Quantum Bernoulli factory

Suzie Brown
supervised by Krzysztof Łatuszyński

September 3, 2018

1 Introduction

Loosely speaking, a Bernoulli factory is an algorithm taking p-coin flips as inputs and producing an $f(p)$-coin flip as output. The existence and performance of such algorithms for various choices of p has been a subject of interest since the 1950s when random number generation was gaining popularity, and a subject of focused study since the 1990s.

Bernoulli factories are necessary components of certain Monte Carlo strategies, particularly calculating Metropolis-Hastings acceptance probabilities in pseudo-marginal methods and diffusion models. In both cases, some terms in the acceptance probability cannot be evaluated, but can be Bernoulli-sampled. Luckily the acceptance probability α does not need to be explicitly calculated; the Metropolis-Hastings accept/reject step only uses α to produce a Bernoulli(α) random variable. Choosing whether to accept or reject a proposal therefore comes down to converting the available Bernoulli samples to the desired Bernoulli(α) sample, which is precisely a Bernoulli factory problem.

The fundamental problem of which functions are possible was neatly solved by Keane and O'Brien (1994), who give necessary and sufficient conditions on f for the existence of a Bernoulli factory, but does not address the question of running time (that is, the number of p-coin flips required as inputs). Theorems of Nacu and Peres (2005) determined further necessary conditions on f for the running time to satisfy various properties.

The contribution of Dale et al. (2015) is a theorem analogous to that of Keane and O'Brien (1994), in the context of a "quantum Bernoulli factory": where the inputs are quantum p-coins ("quoins"), and quantum processing is allowed. They show that quantum algorithms can simulate a strictly larger class of functions than classical ones and, for classically simulable algorithms, may consume many fewer input quoins than their classical counterparts do coins.

The remainder of the document is organised as follows. Section 2 provides the background in quantum information theory required to understand the quantum Bernoulli factory. Section 3 introduces the Bernoulli factory problem and summarises previous work on classical solutions, starting with some illustrative examples before laying down the theoretical results. Section 4 gives the context in the quantum information literature of a quantum Bernoulli factory and explains how it works, with comparison to the classical case. Finally, Section 5 discusses the scope of its implications.

2 Quantum information

This section is based on the course by Noah Linden (University of Bristol) and discussion with Thomas Hebdige and David Jennings (Imperial College London). For a comprehensive introduction see for example Nielsen and Chuang (2002) or Wilde (2013).

The basis of quantum mechanics is simply linear algebra in a Hilbert space. The definitions in Sections 2.2 to 2.4 are exactly what you would find in a linear algebra course, except for the notation. The famously strange quantum behaviour arises from the "rules of quantum mechanics" (Section 2.5) pertaining to measurements. The majority of phenomena besides are described perfectly naturally by linear algebra, and are only counter-intuitive when one departs from the safety of mathematical abstraction and attempts to interpret quantum behaviour within the framework of the natural world, as perceived on a human scale. It is not surprising that such attempts should lead to confusion, since quantum mechanics is based on complex numbers and so diverges from human reality even at its very fundamentals.

2.1 Dirac notation

Dirac notation is a convenient way of denoting vectors such that it is easy to visually identify inner and outer products, and thus quickly recognise scalars, vectors and matrices:

- $|v\rangle$ denotes a column vector
- $\langle v|$ denotes a row vector
- $\langle u|v\rangle$ denotes an inner product (resulting in a scalar)
- $|u\rangle\langle v|$ denotes an outer product (resulting in a matrix)

Additionally, $\overline{\alpha}$ denotes the complex conjugate of a scalar α, and $U^\dagger$ denotes the adjoint (conjugate transpose) of an operator U.

2.2 Hilbert space

A *Hilbert space* is a vector space with an inner product $\langle\cdot|\cdot\rangle$ satisfying the following:

- $\langle u|\left(\alpha|v\rangle + \beta|w\rangle\right) = \alpha\langle u|v\rangle + \beta\langle u|w\rangle$
- $\langle u|v\rangle = \overline{\langle v|u\rangle}$
- $\langle v|v\rangle \geq 0$ with equality if and only if $|v\rangle$ is the zero vector.

2.3 Orthonormal bases

An *orthonormal basis* of a Hilbert space $\mathcal{H}$ is a set of vectors $\{v_1, \ldots, v_n\}$ in $\mathcal{H}$ such that:

- $\text{span}\{v_1, \ldots, v_n\} = \mathcal{H}$
- $\langle v_i|v_j\rangle = \delta_{ij}$

We now restrict to the Hilbert space $\mathbb{C}^2$, which is the state space of a single quantum bit (qubit), and is sufficient to describe the basics of quantum information. The *computational basis* $\{|0\rangle, |1\rangle\} = \{(1,0)^T, (0,1)^T\}$ is taken as the canonical basis for $\mathbb{C}^2$ and is henceforth used wherever not specified otherwise. Since it is an orthonormal basis, every vector $|v\rangle$ in $\mathbb{C}^2$ has a unique representation

$$|v\rangle = \alpha|0\rangle + \beta|1\rangle \equiv (\alpha, \beta)^T$$

for some $\alpha, \beta \in \mathbb{C}$. For reasons which will probably not become apparent in this treatment, we restrict ourselves to *normalised* vectors, requiring also $|\alpha|^2 + |\beta|^2 = 1$. We will also consider two vectors equivalent if they differ only by an overall phase, i.e. $|u\rangle \equiv |v\rangle$ if $|u\rangle = e^{i\theta}|v\rangle$ for some θ, since it is impossible to distinguish between two such vectors with any measurement.

To ensure coherency with the properties of the inner product, we have that

$$\langle v| = \overline{\alpha}\langle 0| + \overline{\beta}\langle 1|.$$

The inner product of $|v\rangle = \alpha|0\rangle + \beta|1\rangle$ with $|u\rangle = \gamma|0\rangle + \delta|1\rangle$ is therefore computed as

$$\begin{aligned}\langle v|u\rangle &= (\overline{\alpha}\langle 0| + \overline{\beta}\langle 1|)(\gamma|0\rangle + \delta|1\rangle) \\ &= \overline{\alpha}\gamma\langle 0|0\rangle + \overline{\alpha}\delta\langle 0|1\rangle + \overline{\beta}\gamma\langle 1|0\rangle + \overline{\beta}\delta\langle 1|1\rangle \\ &= \overline{\alpha}\gamma + \overline{\beta}\delta.\end{aligned}$$

One alternative choice of orthonormal basis which is worth mentioning is given by $\{|+\rangle, |-\rangle\}$, consisting of the states

$$\begin{aligned}|+\rangle &:= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ |-\rangle &:= \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle).\end{aligned}$$

2.4 Linear operators

A linear operator is an operator with the property

$$A\left(\alpha|u\rangle + \beta|v\rangle\right) = \alpha A|u\rangle + \beta A|v\rangle.$$

It is therefore fully defined by its action on an orthonormal basis. For instance, the quantum NOT operator (usually denoted X) is defined by

$$X|0\rangle = |1\rangle$$
$$X|1\rangle = |0\rangle$$

Equivalently, X can be expressed as a matrix with respect to the computational basis:

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

In Dirac notation, operators are written in terms of outer products of basis states:

$$X = |0\rangle\langle 1| + |1\rangle\langle 0|$$

This is equivalent to the matrix form:

$$X = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

Then the action of X on a general state $|v\rangle = \alpha|0\rangle + \beta|1\rangle$ can be calculated:

$$\begin{aligned} X|v\rangle &= (|0\rangle\langle 1| + |1\rangle\langle 0|)(\alpha|0\rangle + \beta|1\rangle) \\ &= \alpha|0\rangle\langle 1|0\rangle + \alpha|1\rangle\langle 0|0\rangle + \beta|0\rangle\langle 1|1\rangle + \beta|1\rangle\langle 0|1\rangle \\ &= \alpha|1\rangle + \beta|0\rangle. \end{aligned}$$

Definition 1. Let U be a linear operator.

- U is said to be *self-adjoint* (or *hermitian*) if $U^\dagger = U$.
- U is said to be *unitary* if $UU^\dagger = U^\dagger U = I$.

For example, the operator X is both self-adjoint and unitary. Crucially, unitary operators preserve normalisation, so they map states to states. It is also obvious that unitary transformations are always reversible (i.e. the inverse operator exists).

2.5 Rules of Quantum Mechanics

1. *States* of a quantum mechanical system correspond to normalised vectors in Hilbert space, up to an overall phase.
2. *Evolutions* of the system correspond to unitary operators.
3. *Measurements* on quantum states correspond to self-adjoint operators — see below.

2.5.1 Spectral theorem and measurement

The outcome of a measurement depends on the current state of the system and the type of measurement performed (i.e. which self-adjoint oeprator is applied). The spectral theorem states that every self-adjoint operator A can be represented by its spectral decomposition

$$A = \sum_i \lambda_i P_i \tag{1}$$

where $\{\lambda_1, \ldots, \lambda_k\}$ is the set of *distinct* eigenvalues of A, and P_i is the *projection* operator onto the eigenspace corresponding to eigenvalue λ_i.

When we measure a state $|x\rangle$ using operator A, the measurement outcome we observe is one of the eigenvalues of A. In particular, we observe λ_i with probability $\langle x|P_i|x\rangle$. Making a measurement causes the system to collapse onto the eigenspace corresponding to the observed eigenvalue; that is, the state after measurement is proportional to $P_i|x\rangle$.

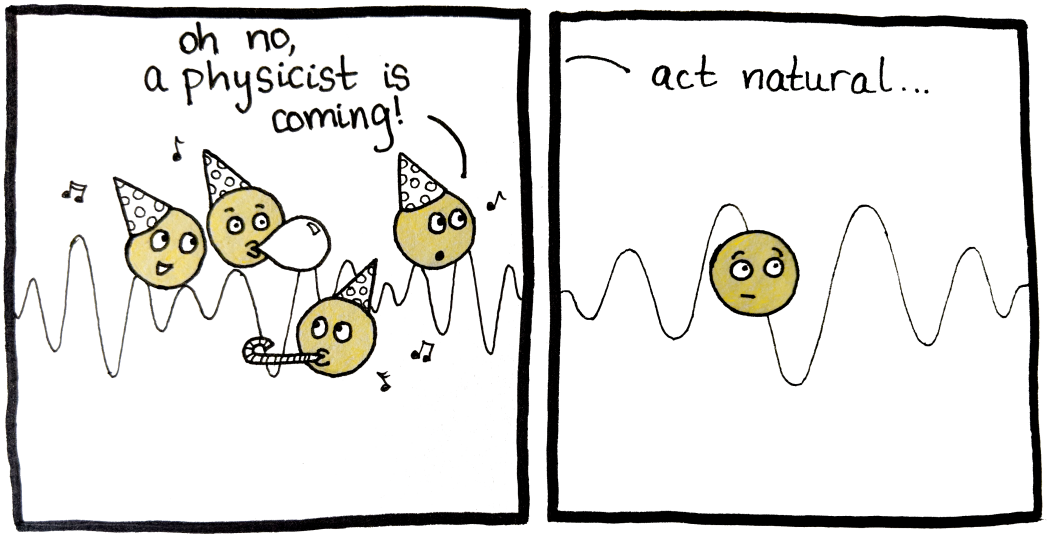


Figure 1: When a quantum system is in a superposition of states, making a measurement causes the system to collapse into the eigenstate corresponding to the observed eigenvalue.

Example 1. To give a concrete example, let us consider again the operator X. This operator has eigenvalues ± 1 corresponding to eigenvectors (or *eigenstates* in quantum mechanics) $|+\rangle$ and $|-\rangle$ respectively. Therefore X admits the diagonal representation

$$X = |+\rangle\langle +| - |-\rangle\langle -|$$

which is of the form (1). Now suppose we make a measurement on the state $|v\rangle = |0\rangle$, using X. The outcome of the measurement will be an eigenvalue of X: either $+1$ or -1. We observe the outcome $+1$ with probability

$$\langle v|P_{+1}|v\rangle = \langle 0|+\rangle \langle +|0\rangle = 1/2$$

in which case the state after measurement is

$$P_{+1}|v\rangle = |+\rangle\langle +|0\rangle \propto |+\rangle.$$

Similarly, with probability $1/2$ we observe the outcome -1 and the state after measurement is $|-\rangle$. △

In this example, the two outcomes are equally likely because the state $|0\rangle$ is 'equidistant' from the two eigenstates of X. In general, outcomes that leave the state after measurement closer to the original state are more likely. This notion is formalised in Section 2.8.

A particularly important type of measurement is measurement in the computational basis. In this case the self-adjoint operator used is $0 \cdot |0\rangle\langle 0| + 1 \cdot |1\rangle\langle 1| = |1\rangle\langle 1|$. Measuring a state $|v\rangle = \alpha|0\rangle + \beta|1\rangle$ in the computational basis returns 0 with probability $|\alpha|^2$ or 1 with probability $|\beta|^2$. This is useful in quantum computing algorithms because it deterministically converts the basis states $|0\rangle$ and $|1\rangle$ to their classical equivalents 0 and 1, ready for classical output.

2.6 Quantum randomness

It is worth remarking at this point on the difference between quantum and classical randomness. As statisticians we are used to dealing with randomness, but we accept that randomness is simply part of a model, and is not purported to exist in nature. We artificially introduce random variables into our models to account for a lack of information, either about the state of the system or about the (presumably deterministic) processes that govern certain phenomena.

On the contrary, our uncertainty about the outcome of a measurement on a quantum system is of a different kind. This uncertainty is not a symptom of our lack of knowledge: even when we know exactly the state of the system, as in Example 1, we are still unable to predict with certainty the outcome of a measurement on the system. The randomness here is intrinsic; the natural processes governing quantum measurement are truly stochastic.

2.7 Pure and mixed states

So far we have only encountered *pure states*, like $|0\rangle$ or $|+\rangle$. But we can also consider probabilistic mixtures of pure states, known as *mixed states*, like

$$|v\rangle = \sum_i p_i |v_i\rangle$$

where $\{p_i\}$ is a probability distribution and the $|v_i\rangle$ are pure states. You can think of this as a 'source' which emits a state $|v_i\rangle$ with probability p_i. A classical analogue is a bit which takes the value 1 with probability p or 0 with probability $1-p$ (a p-coin). In this case, we can say that the expected value of the bit is p.

There is no full ordering on the set of quantum states, so it is not possible to define the expected value of a mixed state. However, we can calculate the expected value of a measurement on a mixed state, using the state's *density operator*.

2.7.1 Density operators

The density operator of a mixed state $|v\rangle = \sum_i p_i |v_i\rangle$ is given by

$$\rho = \sum_i p_i |v_i\rangle\langle v_i|.$$

Suppose we measure this mixed state using the self-adjoint operator $A = \sum_k \lambda_k P_k$, where $\{\lambda_k\}$ are the distinct eigenvalues of A and P_k the projections onto the corresponding eigenspaces. Recall (from Section 2.5.1) that the outcome of the measurement will be an eigenvalue of A. Applying the law of total probability, the probability of observing a particular outcome λ_j is

$$\mathbb{P}(\text{observe } \lambda_j) = \sum_i p_i \langle v_i|P_j|v_i\rangle = tr(\rho P_j)$$

where $tr(\cdot)$ denotes the trace of an operator. Note that $tr(\rho) = 1$ because the states are normalised and $\{p_i\}$ is a probability distribution. The state $\rho = I/2$ is called the *maximally mixed* state; it expresses complete ignorance, like a classical (1/2)-coin.

Different mixtures of states may have the same density operator, but in this case is is impossible to distinguish between them by any measurement.

Example 2. For example, a source $|v\rangle$ which emits $|0\rangle$ or $|1\rangle$ each with probability 1/2 has the same density operator $\rho = I/2$ as a source $|u\rangle$ which emits $|+\rangle$ or $|-\rangle$ each with probability 1/2. Suppose we measured both of these mixed states in the computational basis. The probability of observing the outcome 0 from $|u\rangle$ is

$$\langle u|P_0|u\rangle = \frac{1}{2}\langle 0|0\rangle\langle 0|0\rangle + \frac{1}{2}\langle 1|0\rangle\langle 0|1\rangle = 1/2 \tag{2}$$

while the probability of observing 0 from $|v\rangle$ is

$$\langle v|P_0|v\rangle = \frac{1}{2}\langle +|0\rangle\langle 0|+\rangle + \frac{1}{2}\langle -|0\rangle\langle 0|-\rangle = \frac{1}{2}\cdot\frac{1}{2} + \frac{1}{2}\cdot\frac{1}{2} = 1/2. \tag{3}$$

Therefore measurement in the computational basis can provide no information to distinguish between $|u\rangle$ and $|v\rangle$. And indeed nor can any other measurement — proved easily by exchanging the projection operator $|0\rangle\langle 0|$ for general P in the above — so the two sources are indistinguishable. △

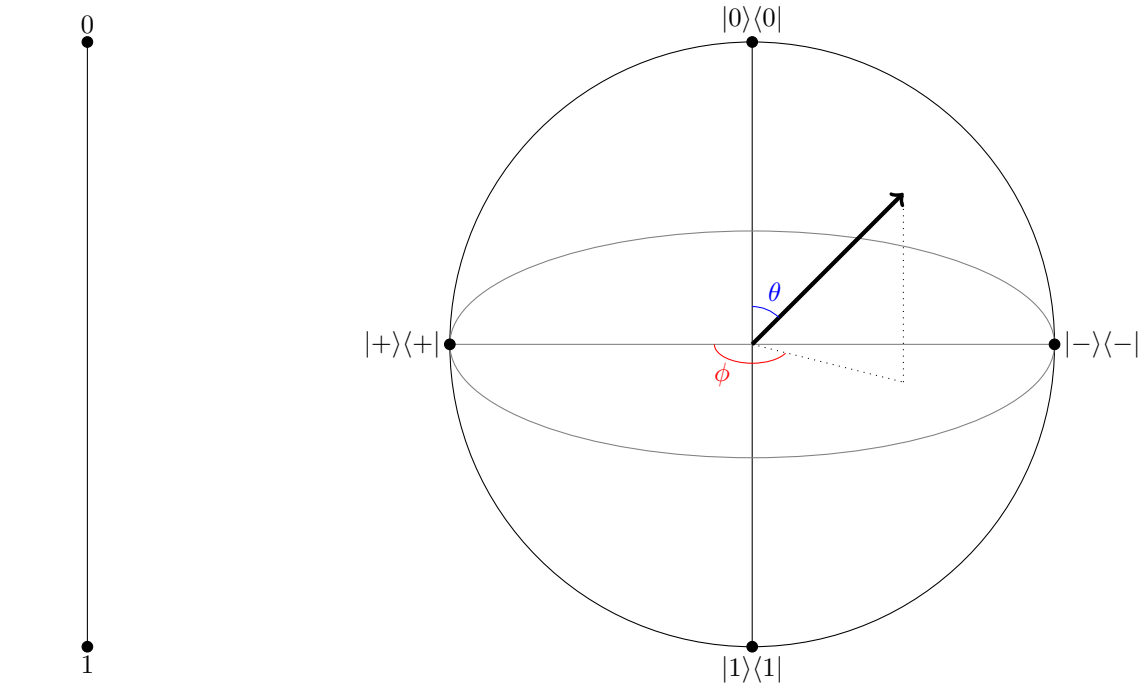
2.8 The Bloch sphere

Density operators have a neat geometrical representation as points in a unit sphere (Figure 2b). This is called the *Bloch sphere*, and can be thought of as the state space of a qubit.

An arbitrary pure state can be written in the form

$$|v\rangle = e^{i\psi}\left[\cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle\right].$$

Since the overall phase $e^{i\psi}$ can be ignored, the state is parametrised by two angles θ and ϕ. Taking these as spherical coordinates with unit radius, we can think of each pure state as a point on the unit sphere (or equivalently, a unit vector). Orthogonal states lie at opposite points on the Bloch sphere.



(a) Classical bit

(b) Qubit

Figure 2: State space of classical bits versus qubits. The state space of a classical bit is the line segment $[0, 1]$*. The 'pure' states 0 and 1 are at the endpoints, and probabilistic mixtures of the two lie inbetween. The state space of a qubit is the Bloch sphere. Pure states lie on the surface of the sphere, and mixed states lie in the interior. The centre of the sphere is the maximally mixed state* $\rho = I/2$*.*

Mixed states can also be represented in the Bloch sphere. The Bloch vector for a mixed state is found by taking the weighted average of the Bloch vectors of its component pure states. Due to the triangle inequality, mixed states therefore lie strictly inside the unit sphere. This illustrates that several different mixtures of states can have the same density operator. Quantum states are distinguishable if and only if they correspond to different points in the Bloch sphere.

There is an analogy here with classical information. The state space of a classical bit is the unit interval: its value could be 0 or 1, or we could take a probabilistic mixture of the two. The 'pure' states then are 0 and 1, which lie on the 'surface' of the interval, while 'mixed' states lie strictly inside the interval (Figure 2a).

Finally, unitary operators correspond to rotations of the Bloch sphere. This illustrates that unitary transformations are reversible.

2.8.1 Geometric view of measurement

Suppose we are going to measure a pure state $|v\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + e^{i\phi}\sin\left(\frac{\theta}{2}\right)|1\rangle$ in the computational basis. There is a simple relationship between the probability of each outcome and the position of $|v\rangle$ in the Bloch sphere.

Figure 3 shows the semicircular slice of the Bloch sphere on which $|v\rangle$ lies. The value of ϕ determines which slice this is, and conditional on that the state only depends on θ — the following calculations are independent of ϕ and apply to an arbitrary pure state. The state $|v\rangle$ is projected onto the line segment between $|0\rangle$ and $|1\rangle$ as shown. Since the sphere has unit radius, the distance labelled x on Figure 3 is

$$x = 1 - \cos\theta.$$

Using the expansion of $|v\rangle$ in the computational basis, we can calculate the probability of observing measurement outcome 0:

$$\mathbb{P}(\text{observe } 0) = \langle v|0\rangle\langle 0|v\rangle = \cos^2\left(\frac{\theta}{2}\right) = \frac{\cos\theta + 1}{2} = 1 - \frac{x}{2}.$$

This relationship between the projected distance and measurement probabilities also applies to mixed states. For instance, if we take any mixture of states all with the same value of θ (but different values of ϕ), the result will hold,

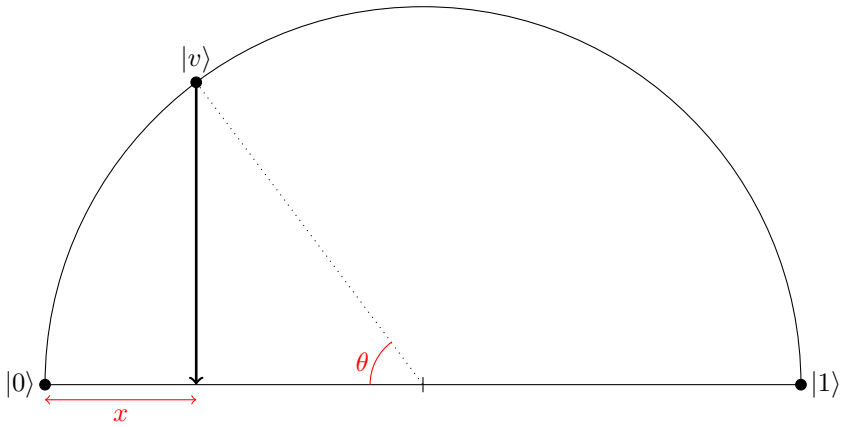


Figure 3: When measuring a state in the computational basis, the probability of observing a certain eigenvalue is related to the projected distance x from the corresponding eigenstate.

because x and $\mathbb{P}(\text{observe } 0)$ will be the same for every component state, and hence for the mixed state too. But the result holds for general mixed states as well. Any point in the sphere whose projection onto the line segment between $|0\rangle$ and $|1\rangle$ lies a distance x from $|0\rangle$ will measure 0 with probability $1 - x/2$.

Furthermore, this illustrates a more general concept, which applies not only to measurement in the computational basis but to any quantum measurement. Any valid measurement operator on one qubit (that is non-trivial in the sense of having more than one possible outcome) has two eigenstates that lie opposite each other on the Bloch sphere. Projecting a state $|v\rangle$ onto the line segment between the two eigenstates gives the probabilities of observing the associated eigenvalues.

2.9 Entanglement

Suppose we now have two qubits and would like to describe their joint state. The joint state space will be $\mathbb{C}^2 \otimes \mathbb{C}^2$, where $\otimes$ denotes the tensor product. This state space has canonical basis

$$\{|0\rangle \otimes |0\rangle, |0\rangle \otimes |1\rangle, |1\rangle \otimes |0\rangle, |1\rangle \otimes |1\rangle\}.$$

We also use the shorthand notations $|0\rangle \otimes |0\rangle = |0\rangle|0\rangle = |00\rangle$. The inner product on $\mathbb{C}^2 \otimes \mathbb{C}^2$ of $|u_1\rangle|u_2\rangle$ with $|v_1\rangle|v_2\rangle$ is defined as $\langle u_1|v_1\rangle \langle u_2|v_2\rangle$.

The joint state of two qubits $|u\rangle = \alpha|0\rangle + \beta|1\rangle$ and $|v\rangle = \gamma|0\rangle + \delta|1\rangle$ can be expanded in the computational basis:

$$|u\rangle \otimes |v\rangle = \alpha\gamma|00\rangle + \alpha\delta|01\rangle + \beta\gamma|10\rangle + \beta\delta|11\rangle.$$

States of this form are called *product states* as they can be written as the tensor product of two states in $\mathbb{C}^2$. However, the set of all product states is only a subset of $\mathbb{C}^2 \otimes \mathbb{C}^2$. In general, a state $a|00\rangle + b|01\rangle + c|10\rangle + d|11\rangle$ may not be expressible as a product of two states in $\mathbb{C}^2$. In this case, it is called an *entangled* state.

For example, consider the state $(|00\rangle + |11\rangle)/\sqrt{2}$. This state is entangled: if we try to write it in the form $|u\rangle \otimes |v\rangle$ as above, we find that either α or δ must be zero, in which case the coefficient of either $|00\rangle$ or $|11\rangle$ must be zero, producing a contradiction.

The set of four entangled states

$$\{|\phi^+\rangle, |\phi^-\rangle, |\psi^+\rangle, |\psi^-\rangle\} := \left\{\frac{|00\rangle + |11\rangle}{\sqrt{2}}, \frac{|00\rangle - |11\rangle}{\sqrt{2}}, \frac{|01\rangle + |10\rangle}{\sqrt{2}}, \frac{|01\rangle - |10\rangle}{\sqrt{2}}\right\}$$

is another important basis for $\mathbb{C}^2 \otimes \mathbb{C}^2$, called the *Bell basis*.

3 Bernoulli factories

A Bernoulli factory is an algorithm that takes Bernoulli(p) samples as inputs and uses them to produce a Bernoulli($f(p)$) sample as output, where f is a known function but the parameter p is unknown (therefore the algorithm must not depend on p). The concept is formalised in Definition 2. The imaginary device for generating Bernoulli(p) random

variables is sometimes referred to as a p-coin or black box; samples thereof may be called flips/tosses or queries. The (possibly random) number of queries required to produce one output is referred to as the running time of the Bernoulli factory.

The first description of a Bernoulli factory (though not under that name) is attributed to Von Neumann (1951), who described an algorithm for simulating $f(p) \equiv 1/2$ (Example 3). Four decades later, Keane and O'Brien (1994) published necessary and sufficient conditions on f under which a Bernoulli factory exists (Theorem 1). They also give an explicit algorithm for simulating f under these conditions, using a decomposition into Bernstein polynomials. However they do not consider the running time, except insofar as fulfilling the standard condition that it must be almost surely finite.

Nacu and Peres (2005) consider the running time more closely, presenting necessary conditions (Theorem 2(b),(c)) on f for the running time to have finite expectation or finite k^{th} moment respectively. Moreover, they give necessary and sufficient conditions (Theorem 2(d)) for the running time to have exponentially decaying tails, in which case they say that f has a 'fast' simulation. This last theorem is underpinned by the result that simulating any real analytic function rests on the ability to simulate $f(p) = 2p$ (Nacu and Peres, 2005, Proposition 14(iii)), affording this function the significance to justify its popularity in the literature.

Huber (2016) provides a generic algorithm for functions of the form $f(p) = kp$ with $k > 1$ (with an appropriate ε-truncation of the domain to satisfy the conditions of the Keane-O'Brien theorem), which includes the $2p$ case. He derives a lower bound on the running time for any kp algorithm, and thus shows that the running time of his algorithm attains the optimal scaling with k and ε.

Definition 2 (Bernoulli factory)**.** Let $S \subseteq [0,1]$ and $f : S \to [0,1]$, and suppose we have access to a sequence of Bernoulli random variables $X_1, X_2, \ldots \overset{iid}{\sim} \text{Bernoulli}(p)$ with unknown parameter $p \in S$. Let $U \in \mathcal{U}$ denote an auxiliary random variable with known distribution, independent of p. Let τ be a stopping time with respect to the natural filtration such that $\mathbb{P}(\tau < \infty) = 1$.
A *Bernoulli factory* is a computable function $\mathcal{A}(U, X_1, \ldots, X_\tau)$ not depending on p, with the property that $Y := \mathcal{A}(U, X_1, \ldots, X_\tau) \sim \text{Bernoulli}(f(p))$ for all $p \in S$.
The stopping time τ, which may depend on the observed values of U and $\{X_i\}$, is the first time at which the value of $\mathcal{A}$ can be computed, and is called the *running time* of the Bernoulli factory.

Definition 3 (simulable)**.** Following the terminology of Keane and O'Brien (1994), a function $f : S \to [0,1]$ is said to be *simulable* if there exists a Bernoulli factory that simulates $f(p)$ for all $p \in S$. Furthermore, it is *strongly simulable* if there exists a Bernoulli factory that simulates $f(p)$ without the need for an auxiliary random variable U.

The first formal treatment of a Bernoulli factory problem was presented by Von Neumann (1951), who described a Bernoulli factory for the constant function $f(p) \equiv 1/2$:

> *To cite a human example, for simplicity, in tossing a coin it is probably easier to make two consecutive tosses independent than to toss heads with probability exactly one-half. If independence of successive tosses is assumed, we can reconstruct a 50-50 chance out of even a badly biased coin by tossing twice. If we get heads-heads or tails-tails, we reject the tosses and try again. If we get head-tails (or tails-heads), we accept the result as heads (or tails). The resulting process is rigorously unbiased, although the amended process is at most 25 percent as efficient as ordinary coin-tossing.*

Most authors now use the slightly more convenient convention with the outcomes inverted, so that once 10 or 01 is observed, the output of the Bernoulli factory is equal to the last coin flip (0 or 1 respectively). It is clear that, as claimed, this procedure produces a (1/2)-coin, because 01 and 10 appear with the same probability, so conditional on observing one or the other, both outcomes have probability 1/2. The procedure is formalised in Example 3.

Example 3. $f(p) \equiv 1/2$.
This function is strongly simulable using the procedure:

$$\tau = \min\{t \in \{2, 4, \ldots\} : X_{t-1} \neq X_t\}$$
$$\mathcal{A}(X_1, X_2, \ldots) = X_\tau.$$

The running time of this algorithm is random and indeed unbounded, but it is almost surely finite, making it a valid Bernoulli factory. In particular, $\tau \overset{d}{=} 2 \times \text{Geom}(2p(1-p))$, with expectation $\mathbb{E}(\tau) = \frac{1}{p(1-p)}$. Figure 4 shows the expected running time for various values of p. The expectation achieves its minimum value of 4 when $p = 1/2$ (that is, the coin was already unbiased), hence von Neumann's claim that "the amended process is at most 25 percent

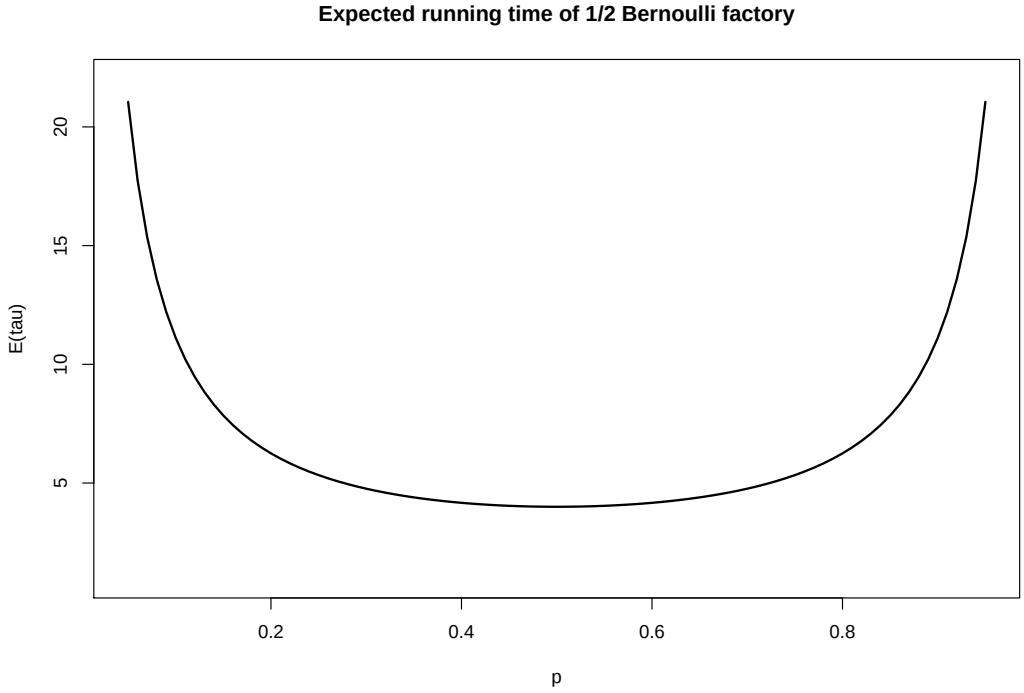


Figure 4: Expected running time $\mathbb{E}(\tau)$ of von Neumann's Bernoulli factory for $f(p) \equiv 1/2$. The expectation attains its minimum $\mathbb{E}(\tau) = 4$ at $p = 1/2$, and is moderate over a large range of p, but grows unboundedly as p approaches 0 and 1.

as efficient as ordinary coin-tossing", although it is possible for this procedure to terminate in just two tosses. The expected running time approaches infinity when p is close to 0 or 1, but is moderate across a large range of values. △

The algorithm given in Example 3 can be extended to simulate other constant functions of the form $f(p) \equiv 1/k$ for $k \in \{2, 3, \dots\}$. The generalised procedure is:

$$\tau = \min\left\{ t \in \{k, 2k, 3k, \dots\} : \sum_{i=t-k+1}^{t} X_i = 1 \right\}$$
$$\mathcal{A}(X_1, X_2, \dots) = X_\tau.$$

However, the running time increases badly with k, even under certain early stopping strategies. Of course, any constant $f(p) \equiv c$ is simulable by employing an auxiliary random variable $U \sim \text{Bernoulli}(c)$, but von Neumann's example shows that $c = 1/2$ is also strongly simulable. In fact, all constant functions are strongly simulable. The $1/2$ procedure is a primitive for a Bernoulli factory for arbitrary constants $f(p) \equiv c \in [0, 1]$, described in Nacu and Peres (2005, Proposition 13).

Example 4. $f(p) \equiv c \in [0, 1]$
We can write c in its binary expansion

$$c = \sum_{n=1}^{\infty} c_n 2^{-n}$$

where $c_n \in \{0, 1\}$. Using the procedure of Example 3 to generate $(1/2)$-coins, we toss $(1/2)$-coins until heads is observed. Call the number of $(1/2)$-coin tosses needed M. Finally we output c_M. $M \sim \text{Geom}(1/2)$, so the total probability of outputting heads is

$$\mathbb{P}(\text{heads}) = \sum_{n=1}^{\infty} c_n \mathbb{P}(M = n) = \sum_{n=1}^{\infty} c_n 2^{-n}$$

as desired. The running time of this Bernoulli factory is $\tau \stackrel{d}{=} \text{Geom}(1/2) \times 2 \times \text{Geom}(2p(1-p))$, independent of c. So on average this procedure for a general constant takes twice as long as the procedure for $1/2$. △

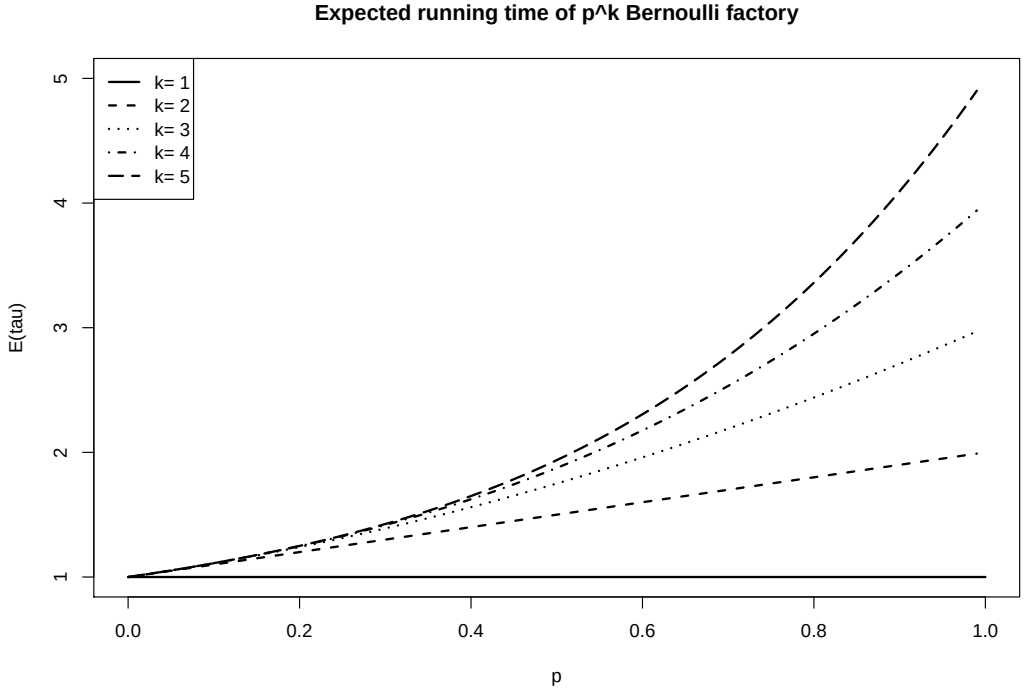


Figure 5: Expected running time for the $f(p) = p^k$ Bernoulli factory described, with early stopping. Without early stopping, the running time is deterministically equal to k. The early stopping procedure significantly increases the efficiency, particularly when p is small or k is large. The minimum number of queries is one for all k and p, and of course the $k = 1$ (identity) Bernoulli factory requires exactly one query for any p.

Another important example is the case of $f(p) = p^2$, which is strongly simulable using exactly two queries, returning heads if and only if both inputs are heads. Example 5 presents a generalisation of this procedure.

Example 5. $f(p) = p^k, p \in [0, 1], k \in \{1, 2, \dots\}$.
This problem is solved easily by flipping the p-coin k times and outputting heads if and only if all k flips return heads. No auxiliary random variable is required.

$$\tau = k$$
$$\mathcal{A}(X_1, \dots, X_T) = \mathbb{I}\{X_1 = 1, X_2 = 1, \dots, X_\tau = 1\}$$

It is clear that this procedure produces heads with probability p^k, since the flips are independent and each gives heads with probability p.

In this case the running time is deterministically equal to k. This can be decreased by an early stopping strategy, whereby we stop the process as soon as a tails input appears (in which case we know immediately that the output will be tails). In this case the running time is random, but still bounded above by k:

$$\tau = k \wedge \min\{t \in \{1, 2, \dots\} : X_t = 0\} \stackrel{d}{=} k \wedge \text{Geom}(1 - p).$$

Figure 5 shows the expected running time of this procedure for a few values of k. We see that early stopping can significantly reduce the expected running time when p is not close to 1, and that the gain in efficiency is greater for larger k. In the case $k = 1$, where f is the identity, the number of coin flips required is deterministically equal to 1, for any p. △

Example 6. $f(p) = 2p(1 - p)$.
This is the probability of obtaining either 01 or 10 with two coin flips. Therefore the Bernoulli factory is simply:

$$\tau = 2$$
$$\mathcal{A}(X_1, X_2) = \mathbb{I}\{(X_1, X_2) \in \{(0, 1), (1, 0)\}\}$$

Of course, many other functions can be similarly obtained by varying the number of p-coin flips used and the set of inputs resulting in a heads output. One example, mentioned in Patel et al. (2018), is the function $f(p) = 3p(1-p)$. Noticing that $3p(1-p) = 3p^2(1-p) + 3p(1-p)^2$, this function is the probability of obtaining any outcome other than 000 or 111 with three coin flips. △

Examples 3–6 cast some light on the sort of manipulations that are possible when constructing Bernoulli factories. Lemma 1 provides some general properties, all of which are trivial to prove.

Lemma 1. *Suppose we have a Bernoulli factory $\mathcal{A}_f$ for the function $f : S_f \to [0,1]$ and a Bernoulli factory $\mathcal{A}_g$ for $g : S_g \to R_g \subseteq [0,1]$. Then*

(a) There exists a Bernoulli factory for $1 - f : S_f \to [0,1]$, given by inverting the output of $\mathcal{A}_f$.

(b) For any $\alpha \in [0,1]$ there exists a Bernoulli factory for $\alpha f + (1-\alpha)g$, given by first sampling from Bernoulli(α) *and, depending on the outcome, using either f or g.*

(c) If $R_g \subseteq S_f$, then there exists a Bernoulli factory for the composition $f \circ g : S_g \to [0,1]$, given by the composition $\mathcal{A}_f \circ \mathcal{A}_g$.

(d) There exists a Bernoulli factory for the product $fg : S_f \cap S_g \to [0,1]$, given by $A_f \wedge A_g$.

Note that (b) is a special case of (c). Additionally, since there is a trivial Bernoulli factory for $f(p) \equiv 0$, (b) implies that αf is simulable for any $\alpha \in [0,1]$. Many more properties, including consideration of their running times, are presented in Nacu and Peres (2005).

It is not immediately obvious that not all functions are simulable. However, one counterexample is the following.

Example 7. $f(p) = 2p, p \in (0, 1/2)$.
This function turns out not to be simulable, although the truncated function $f(p) = 2p \wedge (1-\varepsilon)$ — or equivalently, restricting the domain to $p \in (0, (1-\varepsilon)/2)$ — is simulable for any $\varepsilon > 0$. In this case the number of queries required is $O(\varepsilon^{-1})$ (Huber, 2016).

The problem with the unrestricted function is that $f(p)$ approaches 1 within $p \in (0,1)$, which violates one of the conditions given in the Keane-O'Brien theorem (Theorem 1). △

Lemma 2. *It is not possible to construct a Bernoulli factory for $f(p) = 2p$ where $p \in (0, 1/2)$.*

The following proof is an intuitive one, adapted from Łatuszyński et al. (2011, Corollary 3.3). The intuition behind the proof is that there is no sharp change between say a (1/4)-coin and a (1/2)-coin; but there is a qualitative difference between the resulting (1/2)-coin and 1-coin respectively. The latter is no longer random, but deterministically returns heads every time, regardless of the outcome on the (1/2)-coin that seeds it.

Proof. Let $\mathbb{P}_p$ denote the probability measure conditioned on the input coin having probability p of heads. We will compare the cases $p = 1/4$ and $p = 1/2 - \varepsilon$. First,

$$\frac{1}{2} = \mathbb{P}_{1/4}[\mathcal{A}(X_1, X_2, \dots) = 0] = \lim_{t\to\infty} \mathbb{P}_{1/4}[\mathcal{A}(X_1, \dots, X_T) = 0, T < t]$$

Thus there exists a $t_0 \in \mathbb{N}$ such that

$$\mathbb{P}_{1/4}[\mathcal{A}(X_1, \dots, X_T) = 0, T \le t_0] > \frac{1}{4}$$

Bounding the maximum possible discrepancy between a sequence of t_0 flips from a p-coin and a (1/4)-coin, we have for any $p \in [1/4, 1/2)$:

$$\mathbb{P}_p[\mathcal{A}(X_1, \dots, X_T) = 0, T \le t_0] \ge \left(\frac{1}{2}\right)^{t_0} \mathbb{P}_{1/4}[\mathcal{A}(X_1, \dots, X_T) = 0, T \le t_0] \ge \left(\frac{1}{2}\right)^{t_0+2}.$$

Now

$$\mathbb{P}_{1/2-\varepsilon}[\text{output } 1] = 1 - \mathbb{P}_{1/2-\varepsilon}[\mathcal{A}(X_1, X_2, \dots) = 0] \le 1 - \mathbb{P}_{1/2-\varepsilon}[\mathcal{A}(X_1, \dots, X_T) = 0, T \le t_0] \le 1 - \left(\frac{1}{2}\right)^{t_0+2}$$

Taking the limit as $\varepsilon \to 0$, the LHS is $f(1/2 - \varepsilon) = 1 - 2\varepsilon \to 1$, while the RHS is independent of ε and remains strictly less than 1. This provides the contradiction, and hence such an algorithm $\mathcal{A}$ cannot exist. □

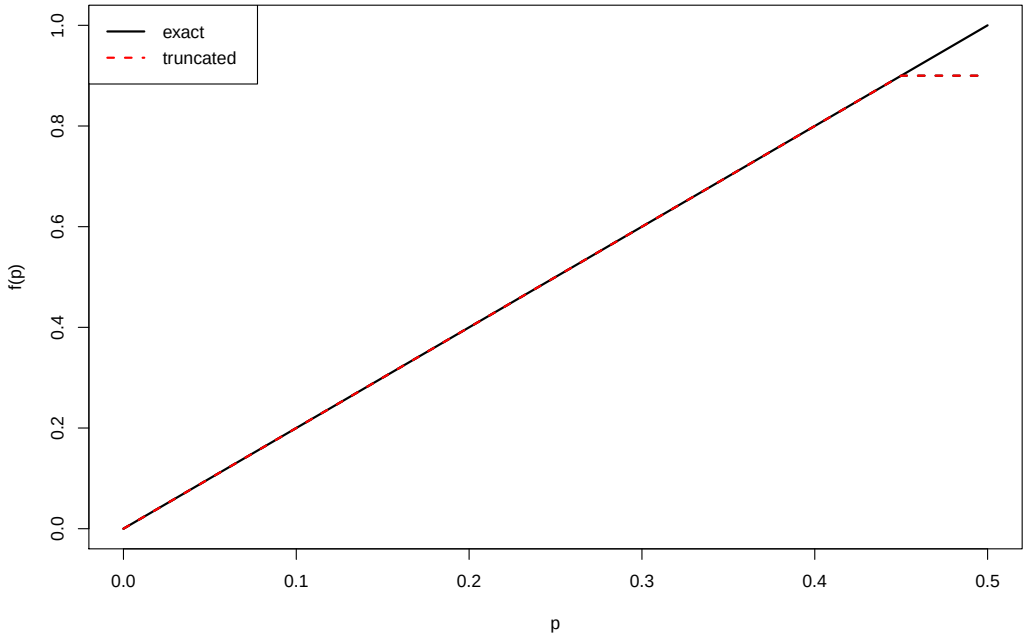


Figure 6: The function $f(p) = 2p$ (black) is not simulable; whereas the truncated function $f(p) = 2p \wedge (1-\varepsilon)$ (red) is simulable for any $\varepsilon > 0$, using $O(\varepsilon^{-1})$ queries.

Lemma 2 is just a particular example of a general class of functions that cannot be simulated by a classical Bernoulli factory. The Keane-O'Brien theorem describes the exact class of classically simulable functions.

Theorem 1 (Keane and O'Brien (1994))**.** *A function $f : S \to [0,1]$ is simulable if and only if f is continuous on S and either*

(a) f is constant on S, or

(b) there exists $n \in \mathbb{N}$ such that $\min\{f(p), 1-f(p)\} \geq \min\{p^n, (1-p)^n\}$ for all $p \in S$.

Roughly speaking, the second condition is "$f(p)$ is polynomially bounded away from 0 and 1 inside its domain" — note that $f(p)$ is allowed to touch 0 and 1 at $p \in \{0,1\}$. A proof is given in Keane and O'Brien (1994), but is not reproduced here. We now see that Lemma 2 follows as a corollary from the Keane-O'Brien theorem.

Corollary 1. *It is not possible to construct a Bernoulli factory for $f(p) = 2p$ where $p \in (0, 1/2)$.*

Proof. Firstly, f is continuous on $S = (0, 1/2)$, so the initial condition of Theorem 1 is satisfied. Since f is not constant, we require that there exists $n \in \mathbb{N}$ such that $\min\{f(p), 1-f(p)\} \geq \min\{p^n, (1-p)^n\}$ for all $p \in S$. Take $p = \frac{1}{2} - \varepsilon$; then $p \in S$ for all $\varepsilon > 0$. We calculate the relevant quantities and, since the inequality should hold for all $\varepsilon > 0$, we can take the limit $\varepsilon \to 0$:

$$\begin{aligned} \min\{f(p), 1-f(p)\} &= 2\varepsilon \quad \stackrel{\varepsilon \to 0}{\longrightarrow} \quad 0 \\ \min\{p^n, (1-p)^n\} &= \left(\frac{1}{2} - \varepsilon\right)^n \quad \stackrel{\varepsilon \to 0}{\longrightarrow} \quad 2^{-n}. \end{aligned}$$

Since $2^{-n} > 0$ for all $n \in \mathbb{N}$, we have that for all n there exists ε such that $\min\{f(p), 1-f(p)\} < \min\{p^n, (1-p)^n\}$; therefore f violates the second condition of Theorem 1. □

Theorem 2. *Let $f : S \to (0,1)$. There exists a Bernoulli factory for f with τ satisfying the following properties, under the corresponding necessary conditions on f:*

(a) (Keane and O'Brien, 1994) $\mathbb{P}(\tau < \infty) = 1$ *only if f is continuous on S.*

Property of τ		Property of f on S
almost surely finite	$\Leftrightarrow$	continuous
finite expectation	$\Leftarrow$	Lipschitz
finite k^{th} moment	$\Leftarrow$	k continuous derivatives
exponentially decaying tails	$\Leftrightarrow$	real analytic

Table 1: Summary of results from Keane and O'Brien (1994) and Nacu and Peres (2005) giving conditions on the function $f : S \to (0, 1)$ for a Bernoulli factory to exist with running time τ satisfying certain properties.

(b) (Nacu and Peres, 2005, Proposition 23) $\sup_{p\in S} \mathbb{E}(\tau) < \infty$*, only if f is Lipschitz over S.*

(c) (Nacu and Peres, 2005, Proposition 22) $\mathbb{E}(\tau^k) < \infty$ *and* $\lim_{n\to\infty} \mathbb{E}(\tau^k)\mathbb{I}\{\tau > t\} = 0$ *uniformly for all $p \in S$, only if f is k times continuously differentiable on S.*

(d) (Nacu and Peres, 2005, Theorem 2) $\mathbb{P}(\tau > t) \leq c\rho^t$*, for some constants $c > 0, \rho < 1$ and for all $p \in S$, only if f is real analytic on every open subset of S.*

The four cases of Theorem 2 are summarised in Table 1. Proofs are given in the original references but are not reproduced here.

4 The quantum advantage

It is known that for certain computational problems, quantum computers provide a theoretical advantage over classical computers. For instance, the Deutsch-Jozsa algorithm (Deutsch and Jozsa, 1992) is an example of a problem (if not a particularly practical one) which a quantum computer can solve in exponentially less time than the best possible classical algorithm.

It is widely believed (at least within the field) that quantum computing really is more powerful than its classical counterpart. Quantum computers are able to do anything that classical computers can, and quantum speed-ups have been proved for a handful of carefully constructed problems (see for another example Grover's algorithm for unstructured search (Grover, 1997)), so from a theoretical viewpoint there is no denying the so-called 'quantum supremacy'.

There is, however, a lack of rigorous results for any computational problems with significant practical applications. One such problem was apparently solved by Kerenidis and Prakash (2016), who came up with a quantum speed-up for the recommendation problem — that is, how Amazon and Netflix estimate which products you will like given yours and other customers' ratings of all products. However, the authors were unable to prove that no comparably fast classical algorithm exists, and one has now been found (Tang, 2018).

4.1 Quantum Bernoulli factories

The Bernoulli factory problem is an example of a practically applicable problem that is nevertheless simple to state and analyse. It was therefore a good candidate in the quest for a practical and provable quantum advantage, and indeed Dale et al. (2015) prove this advantage.

We will use the term *quantum Bernoulli factory* to refer to an algorithm that takes (almost surely finitely many) p-quoins as inputs, and outputs a Bernoulli($f(p)$) sample, where as usual $p \in S$ is unknown and $f : S \to [0, 1]$ is known. A p-quoin is the quantum analogue of a classical p-coin, given by the quantum state

$$|p\rangle = \sqrt{1-p}\,|0\rangle + \sqrt{p}\,|1\rangle.$$

This is a sensible definition since measuring $|p\rangle$ in the computational basis gives output 0 with probability $1 - p$ or 1 with probability p, like a classical p-coin. Note that the quantum Bernoulli factory has a quantum input (quoins) and a classical output (measurement outcome). We will call a function *quantum simulable* if there exists a quantum Bernoulli factory for it.

Dale et al. (2015) prove that quantum computation allows a strictly larger class of functions to be simulated, taking as an example the notorious $f(p) = 2p$. While this function is not simulable in the classical setting, the authors show that it is quantum simulable, and provide an explicit algorithm using two entangled qubits. They also determine necessary and sufficient conditions (Theorem 3) for a function to be quantum simulable, which are weaker than those for classical simulability provided by the Keane-O'Brien theorem (Theorem 1).

Theorem 3 (Dale et al. (2015, Theorem 1)). *A function $f : [0,1] \to [0,1]$ is quantum simulable (using only operations on single qubits) if and only if:*

(a) f is continuous on S

(b) $Z := \{z : f(z) = 0\}$ and $W := \{w : f(w) = 1\}$ are both finite sets

(c) $\forall z \in Z$ there exist constants $c \in \mathbb{R}$, $\delta > 0$, integer $k < \infty$ such that $c(p-z)^{2k} \leq f(p)$ for all $p \in [z-\delta, z+\delta]$

(d) $\forall w \in W$ there exist constants $c \in \mathbb{R}$, $\delta > 0$, integer $k < \infty$ such that $1 - c(p-w)^{2k} \leq f(p)$ for all $p \in [w-\delta, w+\delta]$.

In the quantum case, the function f is allowed to touch 0 and 1 at finitely many points in its domain, as opposed to the classical case where $f(p)$ may only take the values 0 and 1 at $p \in \{0,1\}$ or in the cases of the constant functions $f \equiv 0$ and $f \equiv 1$. The conditions (c) and (d) ensure that f goes to 0 or 1 only polynomially quickly at these points.

Dale et al. (2015) also claim that the same necessary and sufficient conditions apply even when operations on more than one qubit are allowed. That is, taking advantage of quantum entanglement (as in Section 4.2) does not expand the set of simulable functions. It can, however, substantially reduce the consumption of p-quoins, as demonstrated in Patel et al. (2018).

4.2 A quantum Bernoulli factory for 2p

This section describes the quantum algorithm of Dale et al. (2015) for the function $f_\wedge(p) := 2\min\{p, 1-p\}$, $p \in [0,1]$. This function (Figure 7a) coincides with that of Example 7 when the domain is restricted to $p \in (0, 1/2)$, so it is sufficient to find a Bernoulli factory for $f_\wedge$.

Although Dale et al. (2015) also constructs a quantum algorithm using only measurements on single qubits (and both algorithms are implemented by Patel et al. (2018)), we will focus on the two-qubit algorithm which is simpler to describe and gives the best performance.

First note that

$$f_\wedge(p) = 2\min\{p, 1-p\} = 1 - \sqrt{1-4p(1-p)} = \sum_{k=1}^{\infty} \binom{2k}{k} \frac{1}{(2k-1)2^{2k}} (4p(1-p))^k =: \sum_{k=1}^{\infty} q_k (4p(1-p))^k. \quad (4)$$

where q_k are constants not depending on p, and are therefore simulable. Due to property (b) of Lemma 1, $f_\wedge$ is then simulable if there exists a Bernoulli factory for $(4p(1-p))^k$. Since this is a case of Example 5, we need only construct a Bernoulli factory for $4p(1-p)$. This function is not classically simulable since it still touches 1 at $p = 0.5$ (Figure 7b).

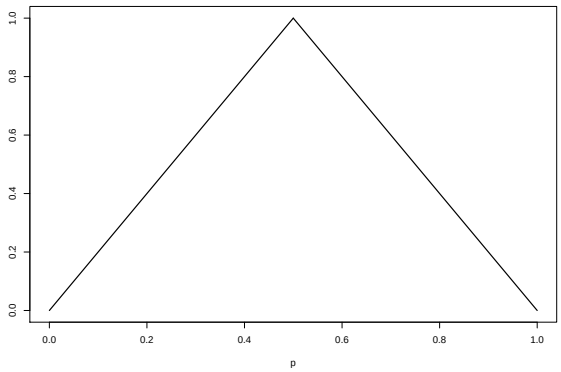
However, Dale et al. (2015) describe the following quantum Bernoulli factory for the function $4p(1-p)$. We first prepare two p-quoins; that is, we have two qubits jointly in the product state $|p\rangle|p\rangle$. We then measure the joint state in the Bell basis, say with the operator

$$0 \cdot |\phi^+\rangle\langle\phi^+| + 1 \cdot |\phi^-\rangle\langle\phi^-| + 2 \cdot |\psi^+\rangle\langle\psi^+| + 3 \cdot |\psi^-\rangle\langle\psi^-|.$$

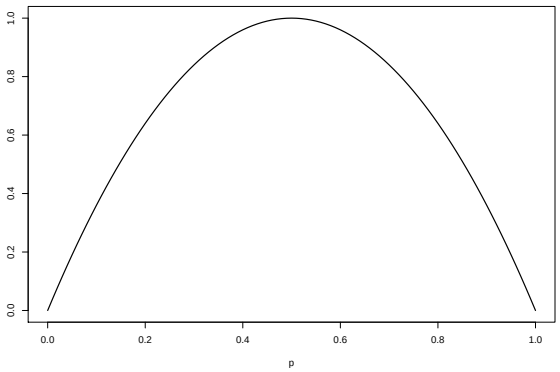
Then the possible measurement outcomes are:

$$\begin{aligned}
\mathbb{P}(\text{observe } 0) &= \langle v|\phi^+\rangle \langle\phi^+|v\rangle = 1/2 \\
\mathbb{P}(\text{observe } 1) &= \langle v|\phi^-\rangle \langle\phi^-|v\rangle = \frac{(1-2p)^2}{2} \\
\mathbb{P}(\text{observe } 2) &= \langle v|\psi^+\rangle \langle\psi^+|v\rangle = 2p(1-p) \\
\mathbb{P}(\text{observe } 3) &= \langle v|\psi^-\rangle \langle\psi^-|v\rangle = 0
\end{aligned}$$

This means that $\mathbb{P}(\text{observe 1 or 2}) = 1/2$, and $\mathbb{P}(\text{observe 2} \mid \text{observe 1 or 2}) = 4p(1-p)$, which is the required function. Thus our Bernoulli factory for $4p(1-p)$ is as follows: measure $|p\rangle|p\rangle$ in the Bell basis until the observed outcome is either 1 or 2, then return heads if the outcome is 2, or tails if the outcome is 1. This procedure has running time $\tau \stackrel{d}{=} \text{Geom}(1/2)$, which is as good as we can hope for, having exponentially decaying tails.



(a) The function $f_\wedge(p)$, which cannot be simulated classically since it touches 1 within its domain.



(b) Simulation of $f_\wedge(p)$ reduces to simulation of this function $4p(1-p)$, which is also classically unobtainable.

Figure 7: The function $f\wedge(p) = 2\min\{p, 1-p\}$ and the component function $4p(1-p)$ to which its quantum Bernoulli factory reduces.

Let us now consider the full procedure for $f_\wedge$. We first choose a k according to the probability distribution $\{q_k\}$ (Figure 9a). We then repeat the the $4p(1-p)$ Bernoulli factory k times to determine the output. The full procedure therefore has running time

$$\tau \stackrel{d}{=} \text{Geom}(1/2) \sum_{k=1}^{\infty} k \cdot q_k$$

which is almost surely finite as required. But since $k \cdot q_k$ is not summable, the *expected* running time is $\mathbb{E}(\tau) = +\infty$ for all p.

However, we can improve the running time by using early stopping as in Example 5, so that we output tails as soon as any of the $4p(1-p)$ procedures outputs tails. In this case the running time becomes

$$\tau \stackrel{d}{=} \text{Geom}(1/2) \sum_{k=1}^{\infty} [\text{Geom}(1 - 4p(1-p)) \wedge k] \cdot q_k.$$

Conditioned on the choice of k, the expectation is

$$\mathbb{E}(\tau \mid k) = 2 \cdot \frac{1 - (4p(1-p))^k}{1 - 4p(1-p)}.$$

The unconditional expectation is not available analytically, but is approximated in Figure 9b. The Geometric early stopping time moderates the overall running time sufficiently that it has finite expectation whenever $p \neq 1/2$, but $\mathbb{E}(\tau) \to +\infty$ as $p \to 1/2$. That is, the algorithm behaves badly around the 'difficult' point $p = 1/2$.

4.2.1 Experimental considerations

Patel et al. (2018) describe a physical implementation of this quantum Bernoulli factory. Since in a physical system infinite quoin consumption cannot be allowed, the authors approximate the infinite series (4) by truncating it after k_{max} terms. The resulting target functions are only approximations of $f_\wedge$, which are rounded at the top and therefore do not touch 1 at $p = 1/2$ (Figure 8).

In fact, the authors admit that with the experimental equipment currently available, it is not possible to achieve a better approximation than that given by $k_{max} \simeq 2000$. For higher-order approximations, the experimental errors become overwhelming. This means that there is a bound on the precision possible with the quantum algorithm, no matter how many quoins we are allowed to use — a shortcoming not suffered by the classical algorithm. Of course, improvements in equipment will allow higher precision, but it is reasonable to suppose that it will always be bounded at some level.

Also, the authors do not run their experiment at the input value $p = 1/2$, presumably since the coin consumption there would ideally have infinite mean and so the experiment would be time-consuming, with many runs using the maximum k_{max} quoins. Patel et al. (2018, Figure 3e) shows a realisation of the quoin consumption, similar to

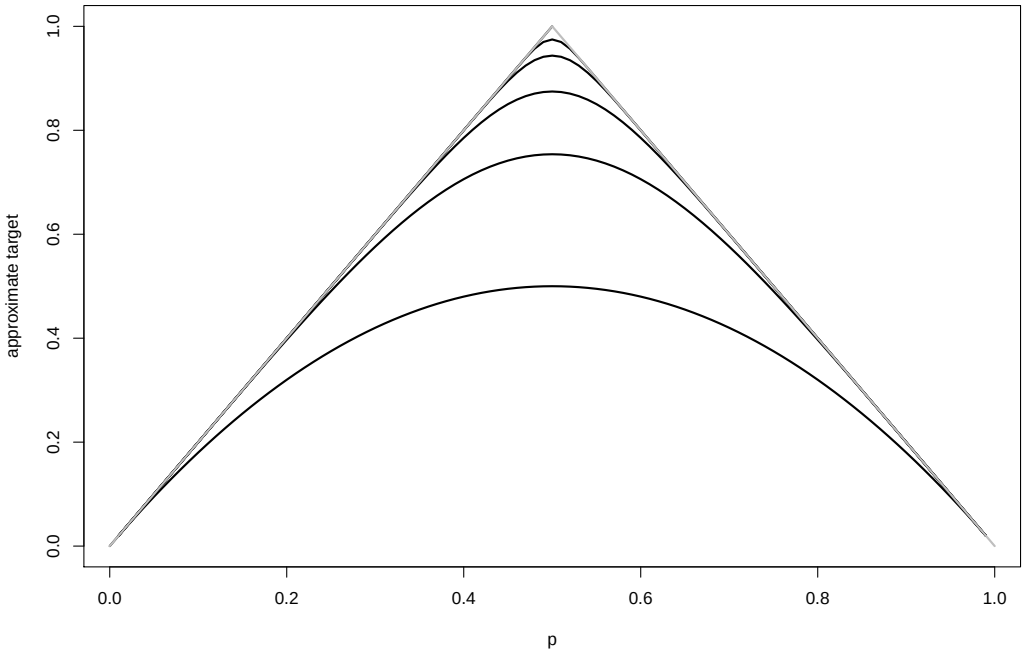
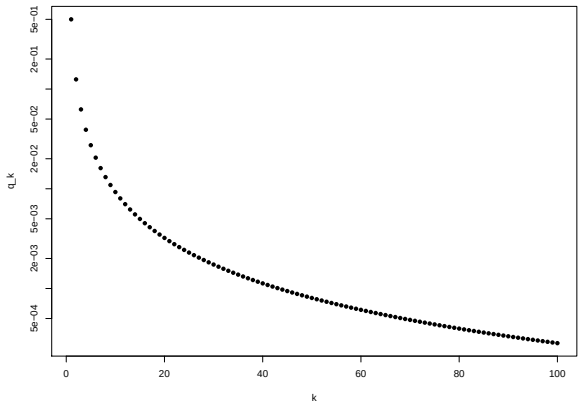
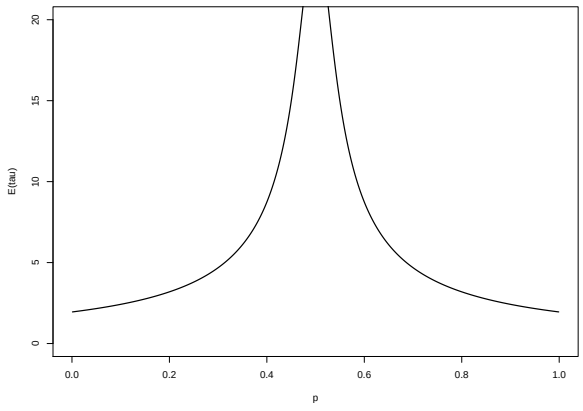


Figure 8: Approximations to $f_\wedge(p)$ when the series expansion (4) is truncated at $k_{max} = 1, 5, 20, 100, 500$. Each subsequent approximation is closer to the target function, but none attains the ideal $f(1/2) = 1$.



(a) Value of q_k (on a logarithmic scale) for various k. The probabilities decay quickly, so it is unlikely to choose a large value of k which would make the QBF slow.



(b) Expected running time for the $f_\wedge(p)$ quantum Bernoulli factory. The running time is almost surely finite for all p, but its expectation goes to infinity when p is close to 1/2.

Figure 9: (a) Value of q_k for $k = 1, \ldots, 100$. (b) Expected running time of the QBF for $f_\wedge(p)$.

Figure 9b, except that in practice the quoin consumption cannot approach infinity, and tops out at a mean of 60 in their experiment, when p is close to $1/2$.

There is a question as to whether the quantum Bernoulli factory for $f_\wedge$ is really an improvement on the classical truncated one, since experimental constraints effectively amount to something like an ε-truncation of the function even in the quantum case (Figure 8). However, Patel et al. (2018) claim that the quantum algorithm given a fixed number of input quoins achieves a much better approximation than the best classical algorithm using the equivalent number of input coins.

4.2.2 Scaling limits

Huber (2016) shows that the best classical Bernoulli factory for $f(p) = 2p \wedge 1 - \varepsilon$ has running time that scales like $1/\varepsilon$, where ε is the error at $p = 1/2$ between the ideal function $f(p) = 2p$ and the approximate (truncated) function. In the quantum case we can similarly define ε as the error at $p = 1/2$ between $f_\wedge$ and the approximation using k_{max} terms in the series expansion (4).

Figure 10 shows the theoretical relationship between expected consumption (running time) and approximation error ε, for p close to $1/2$ (recall that at $p = 1/2$ the running time has infinite expectation). The approximation error is given by

$$\varepsilon = 1 - \sum_{k=1}^{k_{max}} q_k$$

and the expected quoin consumption is

$$\frac{2}{1 - 4p(1-p)} \sum_{k=1}^{\infty} q_k [1 - (4p(1-p))^k].$$

Each black point represents a different value of k_{max}, from 1 (bottom right) to 500 (top left). The value of p used is 0.499; bringing p any closer to 0.5 makes no perceptible difference. The red line shows the fitted $\mathbb{E}(\tau) \propto 1/\varepsilon$ relationship, which appears to fit the data very well.

Patel et al. (2018) claim that the quoin consumption scales like $\varepsilon^{-1/2}$, but their result is obtained by averaging over all values of p. This seems an unfair comparison, since the scaling is worst close to $p = 1/2$. Indeed if the equivalent Figure 10 is produced for different values of p we see much better scaling when p is not close to $1/2$, so averaging over p favourably biases the scaling results. The classical scaling result of Huber (2016, Theorem 1) considers the 'worst-case' of the running time, taking the supremum over p, which is what we have tried to emulate here.

It seems therefore that both the classical and quantum Bernoulli factories for $2p$ have running time that scales with $1/\varepsilon$ when p is close to $1/2$. This suggests that the huge experimental difference in resource usage observed by Patel et al. (2018) must be due only to better constants in the quantum case.

4.3 Another quantum Bernoulli factory

A more impressive, if less fundamentally important, example is obtained by considering the function

$$h_a(p) := \left(\sqrt{p(1-a)} - \sqrt{a(1-p)}\right)^2$$

which is plotted for various values of a in Figure 11. This function is not classically simulable except when $a \in \{0, 1\}$, since it takes the value 0 when $p = a$. (Note that for $a \in \{0, 1\}$ we simply recover the functions p and $1 - p$ respectively, which are trivially simulable using one coin.) Contrarily, $h_a(p)$ is quantum simulable for all $a \in [0, 1]$. Moreover, it can be simulated by a quantum Bernoulli factory using just one quoin. The quantum procedure involves applying the unitary operator

$$\begin{pmatrix} \sqrt{1-a} & \sqrt{a} \\ -\sqrt{a} & \sqrt{1-a} \end{pmatrix}$$

to a p-quoin $|p\rangle := \sqrt{1-p}\,|0\rangle + \sqrt{p}\,|1\rangle$, producing the state

$$\left(\sqrt{(1-a)(1-p)} + \sqrt{ap}\right)|0\rangle + \left(\sqrt{p(1-a)} - \sqrt{a(1-p)}\right)|1\rangle.$$

Measurement in the computational basis then outputs 1 with the desired probability $h_a(p)$.

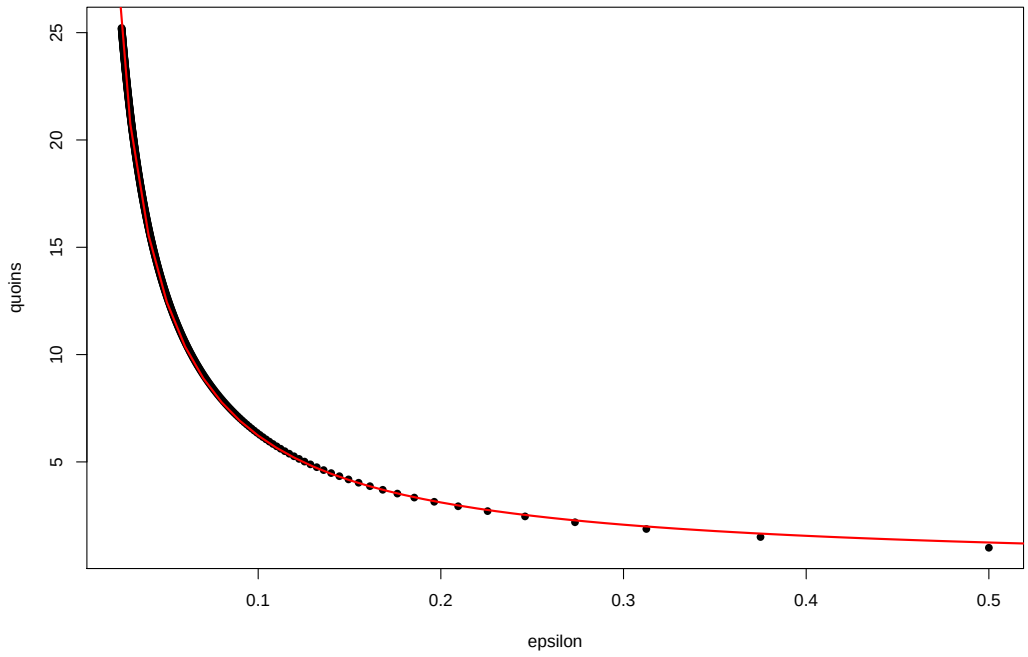


Figure 10: Expected running time (quoin consumption) at $p = 0.499$ of the quantum Bernoulli factory for $f_\wedge$, for various values of k_{max}, against the amount of error ε incurred by truncating at k_{max} terms. A $1/\varepsilon$ relationship (red line) seems to fit very well.

This is an example of how the quantum ability to manipulate quoins prior to measurement yields flexibility that is not available through classical processing. The function $h_a(p)$ is used in Dale et al. (2015) to produce upper and lower bounding functions for $f(p)$, which are required for the proof of Theorem 3.

5 Discussion

We have seen necessary and sufficient conditions describing the exact class of functions that are simulable by classical and quantum Bernoulli factories respectively. The class of classically simulable functions is a strict subset of the class of quantum simulable functions, proving the superiority of quantum computing in this application.

For the purposes of comparison we took a p-quoin, as the quantum analogue of a p-coin, to be equal in value to a classical p-coin flip. However the comparison does not seem particularly fair. Firstly, the running time of the quantum Bernoulli factory is measured as the number of p-quoins consumed, whereas the running time of the classical Bernoulli factory is the number of flips of a p-coin used. In a sense, the classical Bernoulli factory only requires a single coin which can be flipped many times, while in the quantum Bernoulli factory a quoin is produced and then destroyed at every step. But this is still not quite the right way to look at it, since the classical coin is an abstraction and does not really exist; there are only Bernoulli samples which, like the quoins, are 'used up' at each step.

The second and more serious issue is that constructing a p-quoin requires the value of p to be available (though not necessarily known to the experimenters). That is, producing the quantum state $|p\rangle$ requires the quantum computer to take the value of p as an input (Hebdige, 2018) — but the fundamental point of a Bernoulli factory is that p is unknown.

On a philosophical level, we can abstract the problem and imagine that in both the classical and quantum cases p is unknown to the experimenters but known to some third party. That third party is then responsible in the classical case for feeding the value of p into a Bernoulli sample generator, or in the quantum case for feeding p into a p-quoin generator. In this framework, p plays the same role for both factories. This framework also addresses the first problem, because we can then equate classical coins with 'dephased' quoins, which are exactly like quoins except that they lack the ability for quantum coherence and entanglement.

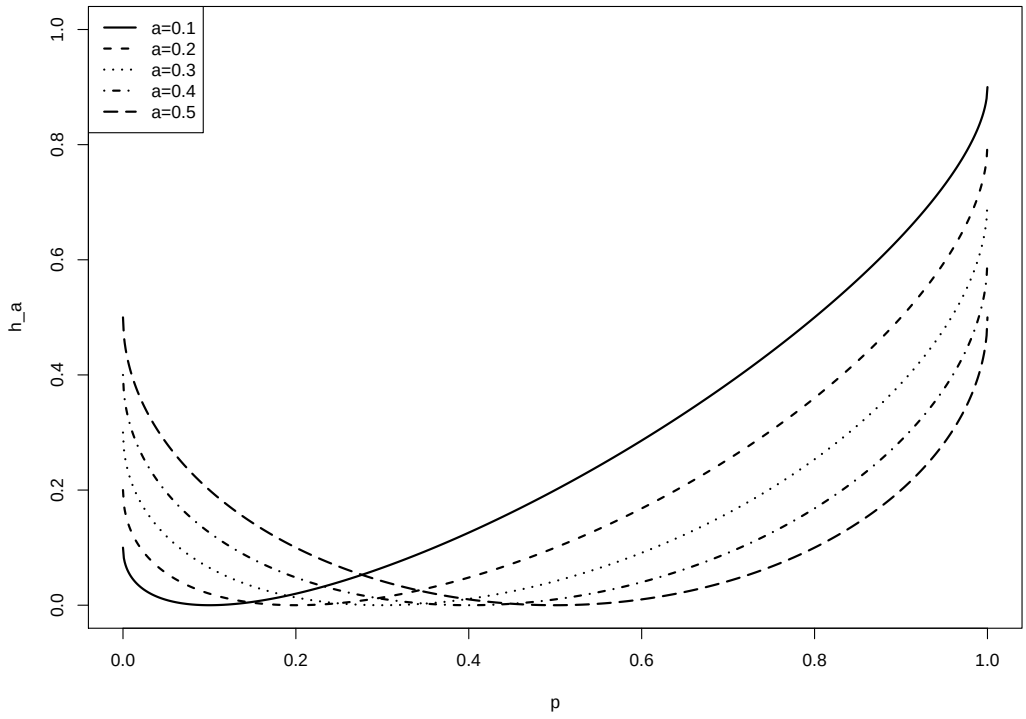


Figure 11: The function $h_a(p)$ for various values of a (taking $a > 0.5$ produces the reflection in $p = 0.5$ of h_{1-a}). These functions are not classically simulable except for $a \in \{0, 1\}$ because they touch 0 at $p = a$. But they can be simulated by a quantum Bernoulli factory using just one p-quoin.

However, this sort of abstraction neglects the original purpose of our Bernoulli factory. Returning to the Monte Carlo setting, we want to use a Bernoulli factory to decide whether to accept or reject a proposal during a particular step of a Metropolis-Hastings algorithm. The objective is to accept with probability exactly equal to α, where α is a function of p. The value of p itself is not available analytically, nor can it be estimated efficiently. We are able to draw samples from Bernoulli(p), although it may be computationally expensive to do so.

Such samples are precisely the inputs required by a classical Bernoulli factory, so in the case of classically simulable functions they are sufficient to produce an exact sample from Bernoulli(α). On the other hand, the quantum Bernoulli factory requires the value of p as an input, which is a much stronger requirement. The only possibility is to use many Bernoulli(p) samples to produce an estimate of p from which to construct quoins, in which case the Bernoulli factory would only be approximate.

In light of these caveats, it seems that the quantum Bernoulli factory is not after all the coveted applicable example of quantum supremacy, but just another toy problem with no important application. The "provable quantum advantage" appears only when the classical coins are taken to be 'dephased' quoins, in which case the algortihms are no longer applicable to Monte Carlo simulation.

References

Dale, H., Jennings, D. and Rudolph, T. (2015), 'Provable quantum advantage in randomness processing', *Nature communications* **6**, 8203.

Deutsch, D. and Jozsa, R. (1992), 'Rapid solution of problems by quantum computation', *Proc. R. Soc. Lond. A* **439**(1907), 553–558.

Grover, L. K. (1997), 'Quantum mechanics helps in searching for a needle in a haystack', *Physical review letters* **79**(2), 325.

Hebdige, T. (2018), 'Quantum bernoulli factories'.

Huber, M. (2016), 'Nearly optimal bernoulli factories for linear functions', *Combinatorics, Probability and Computing* **25**(4), 577–591.

Keane, M. and O'Brien, G. L. (1994), 'A bernoulli factory', *ACM Transactions on Modeling and Computer Simulation (TOMACS)* **4**(2), 213–219.

Kerenidis, I. and Prakash, A. (2016), 'Quantum recommendation systems', *arXiv preprint arXiv:1603.08675* .

Łatuszyński, K., Kosmidis, I., Papaspiliopoulos, O. and Roberts, G. O. (2011), 'Simulating events of unknown probabilities via reverse time martingales', *Random Structures & Algorithms* **38**(4), 441–452.

Nacu, Ş. and Peres, Y. (2005), 'Fast simulation of new coins from old', *The Annals of Applied Probability* **15**(1A), 93–115.

Nielsen, M. A. and Chuang, I. (2002), *Quantum computation and quantum information*, AAPT.

Patel, R. B., Rudolph, T. and Pryde, G. J. (2018), 'An experimental quantum bernoulli factory', *arXiv preprint arXiv:1807.04297* .

Tang, E. (2018), 'A quantum-inspired classical algorithm for recommendation systems', *arXiv preprint arXiv:1807.04271* .

Von Neumann, J. (1951), 'Various techniques used in connection with random digits', *Appl. Math Ser* **12**(36-38), 3.

Wilde, M. M. (2013), *Quantum information theory*, Cambridge University Press.