Advice Coins for Classical and Quantum Computation

Scott Aaronson[*]   Andrew Drucker[†]

Abstract

We study the power of classical and quantum algorithms equipped with nonuniform advice, in the form of a coin whose bias encodes useful information. This question takes on particular importance in the quantum case, due to a surprising result that we prove: *a quantum finite automaton with just two states can be sensitive to arbitrarily small changes in a coin's bias.* This contrasts with classical probabilistic finite automata, whose sensitivity to changes in a coin's bias is bounded by a classic 1970 result of Hellman and Cover.

Despite this finding, we are able to bound the power of advice coins for space-bounded classical and quantum computation. We define the classes BPPSPACE/coin and BQPSPACE/coin, of languages decidable by classical and quantum polynomial-space machines with advice coins. Our main theorem is that both classes coincide with PSPACE/poly. Proving this result turns out to require substantial machinery. We use an algorithm due to Neff for finding roots of polynomials in NC; a result from algebraic geometry that lower-bounds the separation of a polynomial's roots; and a result on fixed-points of superoperators due to Aaronson and Watrous, originally proved in the context of quantum computing with closed timelike curves.

1   Introduction

1.1   The Distinguishing Problem

The fundamental task of mathematical statistics is to learn features of a random process from empirical data generated by that process. One of the simplest, yet most important, examples concerns a coin with unknown bias. Say we are given a coin which lands "heads" with some unknown probability q (called the *bias*). In the *distinguishing problem*, we assume q is equal either to p or to $p + \varepsilon$, for some known p, ε, and we want to decide which holds.

A traditional focus is the *sample complexity* of statistical learning procedures. For example, if $p = 1/2$, then $t = \Theta\left(\log\left(1/\delta\right)/\varepsilon^2\right)$ coin flips are necessary and sufficient to succeed with probability $1 - \delta$ on the distinguishing problem above. This assumes, however, that we are able to count the number of heads seen, which requires $\log(t)$ bits of memory. From the perspective of computational efficiency, it is natural to wonder whether methods with a much smaller space requirement are possible. This question was studied in a classic 1970 paper by Hellman and Cover [13]. They showed that any (classical, probabilistic) finite automaton that distinguishes with bounded error between a coin of bias p and a coin of bias $p + \varepsilon$, must have $\Omega\left(p\left(1 - p\right)/\varepsilon\right)$ states.[1] Their result holds with *no restriction* on the number of coin flips performed by the automaton. This makes the result especially interesting, as it is not immediately clear how sensitive such machines can be to small changes in the bias.

[1]For a formal statement, see Section 2.5.

Several variations of the distinguishing problem for space-bounded automata were studied in related works by Hellman [12] and Cover [10]. Very recently, Braverman, Rao, Raz, and Yehudayoff [8] and Brody and Verbin [9] studied the power of restricted-width, *read-once branching programs* for this problem. The distinguishing problem is also closely related to the *approximate majority* problem, in which given an n-bit string x, we want to decide whether x has Hamming weight less than $(1/2 - \varepsilon) n$ or more than $(1/2 + \varepsilon) n$. A large body of research has addressed the ability of constant-depth circuits to solve the approximate majority problem and its variants [1, 3, 4, 17, 20, 21].

1.2 The Quantum Case

In this paper, our first contribution is to investigate the power of *quantum* space-bounded algorithms to solve the distinguishing problem. We prove the surprising result that, in the absence of noise, quantum finite automata with a constant number of states can be sensitive to *arbitrarily small* changes in bias:

Theorem 1 (Informal) *For any $p \in [0, 1]$ and $\varepsilon > 0$, there is a quantum finite automaton $M_{p,\varepsilon}$ with just two states (not counting the $|\mathrm{Accept}\rangle$ and $|\mathrm{Reject}\rangle$ states) that distinguishes a coin of bias p from a coin of bias $p + \varepsilon$; the difference in acceptance probabilities between the two cases is at least 0.01. (This difference can be amplified using more states.)*

In other words, the lower bound of Hellman and Cover [13] has no analogue for quantum finite automata. The upshot is that we obtain a natural example of a task that a quantum finite automaton can solve using *arbitrarily* fewer states than a probabilistic finite automaton, not merely exponentially fewer states! Galvao and Hardy [11] gave a related example, involving an automaton that moves continuously through a field φ, and needs to decide whether an integral $\int_0^1 \varphi(x)\, dx$ is odd or even, promised that it is an integer. Here, a quantum automaton needs only a single qubit, whereas a classical automaton cannot guarantee success with any finite number of bits. Naturally, both our quantum automaton and that of Galvao and Hardy only work in the absence of noise.

1.3 Coins as Advice

This unexpected power of quantum finite automata invites us to think further about what sorts of statistical learning are possible using a small number of qubits. In particular, if space-bounded quantum algorithms can detect arbitrarily small changes in a coin's bias, then could a p-biased coin be an incredibly-powerful *information resource* for quantum computation, if the bias p was well-chosen? A bias $p \in (0, 1)$ can be viewed in its binary expansion $p = 0.p_1 p_2 \ldots$ as an infinite sequence of bits; by flipping a p-biased coin, we could hope to access those bits, perhaps to help us perform computations.

This idea can be seen in "Buffon's needle," a probabilistic experiment that in principle allows one to calculate the digits of π to any desired accuracy.[2] It can also be seen in the old speculation that computationally-useful information might somehow be encoded in dimensionless physical constants, such as the fine-structure constant $\alpha \approx 0.0072973525377$ that characterizes the strength of the electromagnetic interaction. But leaving aside the question of which biases $p \in [0, 1]$ can be realized by actual physical processes, let us assume that coins of *any* desired bias are available. We can then ask: what computational problems can be solved efficiently using such coins? This

---

question was raised to us by Erik Demaine (personal communication), and was initially motivated by a problem in computational genetics.

In the model that we use, a Turing machine receives an input x and is given access to a sequence of bits drawn independently from an *advice coin* with some arbitrary bias $p_n \in [0,1]$, which may depend on the input length $n = |x|$. The machine is supposed to decide (with high success probability) whether x is in some language L. We allow p_n to depend only on $|x|$, not on x itself, since otherwise the bias could be set to 0 or 1 depending on whether $x \in L$, allowing membership in L to be decided trivially. We let BPPSPACE/coin be the class of languages decidable with bounded error by polynomial-space algorithms with an advice coin. Similarly, BQPSPACE/coin is the corresponding class for polynomial-space quantum algorithms. We impose no bound on the running time of these algorithms.

It is natural to compare these classes with the corresponding classes BPPSPACE/poly and BQPSPACE/poly, which consist of all languages decidable by BPPSPACE and BQPSPACE machines respectively, with the help of an arbitrary *advice string* $w_n \in \{0,1\}^*$ that can depend only on the input length $n = |x|$. Compared to the standard advice classes, the strength of the coin model is that an advice coin bias p_n can be an arbitrary real number, and so encode infinitely many bits; the weakness is that this information is only accessible indirectly through the observed outcomes of coin flips.

It is tempting to try to simulate an advice coin using a conventional advice string, which simply specifies the coin's bias to poly (n) bits of precision. At least in the classical case, the effect of "rounding" the bias can then be bounded by the Hellman-Cover Theorem. Unfortunately, that theorem (whose bound is essentially tight) is not strong enough to make this work: if the bias p is extremely close to 0 or 1, then a PSPACE machine really *can* detect changes in p much smaller than $2^{-\operatorname{poly}(n)}$. This means that upper-bounding the power of advice coins is a nontrivial problem even in the classical case. In the quantum case, the situation is even worse, since as mentioned earlier, the quantum analogue of the Hellman-Cover Theorem is false.

Despite these difficulties, we are able to show strong limits on the power of advice coins in both the classical and quantum cases. Our main theorem says that PSPACE machines can effectively extract only poly (n) bits of "useful information" from an advice coin:

Theorem 2 (Main) BQPSPACE/coin = BPPSPACE/coin = PSPACE/poly.

The containment PSPACE/poly $\subseteq$ BPPSPACE/coin is easy. On the other hand, proving BPPSPACE/coin $\subseteq$ PSPACE/poly appears to be no easier than the corresponding quantum class containment. To prove that BQPSPACE/coin $\subseteq$ PSPACE/poly, we will need to understand the behavior of a space-bounded advice coin machine M, as we *vary* the coin bias p. By applying a theorem of Aaronson and Watrous [2] (which was originally developed to understand quantum computing with closed timelike curves), we prove the key property that, for each input x, *the acceptance probability $a_x(p)$ of M is a rational function in p of degree at most* $2^{\operatorname{poly}(n)}$. It follows that $a_x(p)$ can "oscillate" between high and low values no more than $2^{\operatorname{poly}(n)}$ times as we vary p. Using this fact, we will show how to identify the "true" bias p^* to sufficient precision with an advice string of poly (n) bits. What makes this nontrivial is that, in our case, "sufficient precision" sometimes means $\exp(n)$ bits! In other words, the rational functions $a_x(p)$ really *can* be sensitive to doubly-exponentially-small changes to p. Fortunately, we will show that this does not happen too often, and can be dealt with when it does.

In order to manipulate coin biases to exponentially many bits of precision—and to interpret our advice string—in polynomial space, we use two major tools. The first is a space-efficient algorithm for finding roots of univariate polynomials, developed by Neff [14] in the 1990s. The second is a

lower bound from algebraic geometry, on the spacing between consecutive roots of a polynomial with bounded integer coefficients. Besides these two tools, we will also need space-efficient linear algebra algorithms due to Borodin, Cook, and Pippenger [7].

2 Preliminaries

We assume familiarity with basic notions of quantum computation. A detailed treatment of space-bounded quantum Turing machines was given by Watrous [22].

2.1 Classical and Quantum Space Complexity

In this paper, it will generally be most convenient to consider an *asymmetric model*, in which a machine M can accept only by halting and entering a special "Accept" state, but can reject simply by never accepting.

We say that a language L is in the class BPPSPACE/poly if there exists a classical probabilistic PSPACE machine M, as well as a collection $\{w_n\}_{n \geq 1}$ of polynomial-size advice strings, such that:

(1) If $x \in L$, then $\Pr[M(x, w_n) \text{ accepts}] \geq 2/3$.

(2) If $x \notin L$, then $\Pr[M(x, w_n) \text{ accepts}] \leq 1/3$.

Note that we do not require M to accept within any fixed time bound. So for example, M could have expected running time that is finite, yet *doubly* exponential in n.

The class BQPSPACE/poly is defined similarly to the above, except that now M is a polynomial-space *quantum* machine rather than a classical one. Also, we assume that M has a designated accepting state, $|\text{Accept}\rangle$. After each computational step, the algorithm is measured to determine whether it is in the $|\text{Accept}\rangle$ state, and if so, it halts.

Watrous [22] proved the following:

Theorem 3 (Watrous [22]) BQPSPACE/poly = BPPSPACE/poly = PSPACE/poly.

Note that Watrous stated his result for *uniform* complexity classes, but the proof carries over to the nonuniform case without change.

2.2 Superoperators and Linear Algebra

We will be interested in S-state quantum finite automata that can include *non-unitary transformations* such as measurements. The state of such an automaton need not be a *pure state* (that is, a unit vector in $\mathbb{C}^S$), but can in general be a *mixed state* (that is, a probability distribution over such vectors). Every mixed state is uniquely represented by an $S \times S$, Hermitian, trace-1 matrix ρ called the *density matrix*. See Nielsen and Chuang [16] for more about the density matrix formalism.

One can transform a density matrix ρ using a *superoperator*, which is any operation of the form

$$\mathcal{E}(\rho) = \sum_j E_j \rho E_j^\dagger,$$

where the matrices $E_j \in \mathbb{C}^{S \times S}$ satisfy $\sum_j E_j^\dagger E_j = I$.[3]

We will often find it more convenient to work with a "vectorized" representation of mixed states and superoperators. Given a density matrix $\rho \in \mathbb{C}^{S \times S}$, let $\mathrm{vec}\,(\rho)$ be a vector in $\mathbb{C}^{S^2}$ containing the S^2 entries of ρ. Similarly, given a superoperator $\mathcal{E}$, let $\mathrm{mat}\,(\mathcal{E}) \in \mathbb{C}^{S^2 \times S^2}$ denote the matrix that describes the action of $\mathcal{E}$ on vectorized mixed states, i.e., that satisfies

$$\mathrm{mat}\,(\mathcal{E}) \cdot \mathrm{vec}\,(\rho) = \mathrm{vec}\,(\mathcal{E}\,(\rho)).$$

We will need a theorem due to Aaronson and Watrous [2], which gives us constructive access to the *fixed-points* of superoperators.

Theorem 4 (Aaronson-Watrous [2]) *Let $\mathcal{E}\,(\rho)$ be a superoperator on an S-dimensional system. Then there exists a second superoperator $\mathcal{E}_{\mathrm{fix}}\,(\rho)$ on the same system, such that:*

(i) $\mathcal{E}_{\mathrm{fix}}\,(\rho)$ is a fixed-point of $\mathcal{E}$ for every mixed state ρ: that is, $\mathcal{E}\,(\mathcal{E}_{\mathrm{fix}}(\rho)) = \mathcal{E}(\rho)$.

(ii) Every mixed state ρ that is a fixed-point of $\mathcal{E}$ is also a fixed-point of $\mathcal{E}_{\mathrm{fix}}$.

(iii) Given the entries of $\mathrm{mat}\,(\mathcal{E})$, the entries of $\mathrm{mat}\,(\mathcal{E}_{\mathrm{fix}})$ can be computed in $\mathrm{polylog}(S)$ space.

The following fact, which we call the "Leaky Subspace Lemma," will play an important role in our analysis of quantum finite automata. Intuitively it says that, if repeatedly applying a linear transformation A to a vector y "leaks" y into the span of another vector x, then there is a uniform lower bound on the rate at which the leaking happens.

Lemma 5 (Leaky Subspace Lemma) *Let $A \in \mathbb{C}^{n \times n}$ and $x \in \mathbb{C}^n$. Suppose that for all vectors y in some compact set $U \subset \mathbb{C}^n$, there exists a positive integer k such that $x^\dagger A^k y \neq 0$. Then*

$$\inf_{y \in U} \max_{k \in [n]} \left| x^\dagger A^k y \right| > 0.$$

Proof. It suffices to prove the following claim: *for all $y \in U$, there exists a $k \in [n]$ such that $x^\dagger A^k y \neq 0$.* For given this claim, Lemma 5 follows by the fact that $f\,(y) := \max_{k \in [n]} \left| x^\dagger A^k y \right|$ is a continuous positive function on a compact set U.

We now prove the claim. Let V_t be the vector space spanned by $\{Ay, A^2 y, \ldots, A^t y\}$, let $V := \bigcup_{t>0} V_t$, and let $d = \dim V$. Then clearly $d \leq n$ and $\dim\,(V_{t-1}) \leq \dim\,(V_t) \leq \dim\,(V_{t-1}) + 1$ for all t. Now suppose $\dim\,(V_t) = \dim\,(V_{t-1})$ for some t. Then it must be possible to write $A^t y$ as a linear combination of $Ay, \ldots, A^{t-1} y$:

$$A^t y = c_1 Ay + \cdots + c_{t-1} A^{t-1} y.$$

But this means that every *higher* iterate ($A^{t+1} y$, $A^{t+2} y$, etc.) is also expressible as a linear combination of the lower iterates: for example,

$$A^{t+1} y = c_1 A^2 y + \cdots + c_{t-1} A^t y.$$

Therefore $d = \dim\,(V_{t-1})$. The conclusion is that $\mathcal{B} := \{Ay, A^2 y, \ldots, A^d y\}$ is a basis for V. But then, if there exists a positive integer k such that $v^\dagger A^k y \neq 0$, then there must also be a $k \leq d$ such that $x^\dagger A^k y \neq 0$, by the fact that $\mathcal{B}$ is a basis. This proves the claim. ∎

---

[3]This condition is necessary and sufficient to ensure that $\mathcal{E}(\rho)$ is a mixed state, for every mixed state ρ.

2.3 Coin-Flipping Finite Automata

It will often be convenient to use the language of finite automata rather than that of Turing machines. We model a coin-flipping quantum finite automaton as a *pair* of superoperators $\mathcal{E}_0, \mathcal{E}_1$. Say that a coin has *bias* p if it lands heads with independent probability p every time it is flipped. (A coin here is just a 0/1-valued random variable, with "heads" meaning a 1 outcome.) Let $\$_p$ denote a coin with bias p. When the automaton is given $\$_p$, its state evolves according to the superoperator

$$\mathcal{E}_p := p\mathcal{E}_1 + (1-p)\,\mathcal{E}_0.$$

In our model, the superoperators $\mathcal{E}_0, \mathcal{E}_1$ both incorporate a "measurement step" in which the automaton checks whether it is in a designated basis state $|\text{Accept}\rangle$, and if so, halts and accepts. Formally, this is represented by a projective measurement with observables $\{\Gamma_{\text{Acc}}, I - \Gamma_{\text{Acc}}\}$, where $\Gamma_{\text{Acc}} := |\text{Accept}\rangle\langle\text{Accept}|$.

2.4 Advice Coin Complexity Classes

Given a Turing machine M, let $M(x, \$_p)$ denote M given input x together with the ability to flip $\$_p$ at any time step. Then BPPSPACE/coin, or BPPSPACE *with an advice coin*, is defined as the class of languages L for which there exists a PSPACE machine M, as well as a sequence of real numbers $\{p_n\}_{n\geq 1}$ with $p_n \in [0, 1]$, such that for all inputs $x \in \{0, 1\}^n$:

(1) If $x \in L$, then $M(x, \$_{p_n})$ accepts with probability at least $2/3$ over the coin flips.

(2) If $x \notin L$, then $M(x, \$_{p_n})$ accepts with probability at most $1/3$ over the coin flips.

Note that there is no requirement for M to halt after at most exponentially many steps, or even to halt with probability 1; also, M may "reject" its input by looping forever. This makes our main result, which bounds the computational power of advice coins, a stronger statement. Also note that M has no source of randomness other than the coin $\$_{p_n}$. However, this is not a serious restriction, since M can easily use $\$_{p_n}$ to generate unbiased random bits if needed, by using the "von Neumann trick."

Let $q(n)$ be a polynomial space bound. Then we model a $q(n)$-space quantum Turing machine M with an advice coin as a $2^{q(n)}$-state automaton, with state space $\{|y\rangle\}_{y\in\{0,1\}^{q(n)}}$ and initial state $|0^{q(n)}\rangle$. Given advice coin $\$_p$, the machine's state evolves according to the superoperator $\mathcal{E}_p = p\mathcal{E}_1 + (1-p)\,\mathcal{E}_0$, where $\mathcal{E}_0, \mathcal{E}_1$ depend on x and n. The individual entries of the matrix representations of $\mathcal{E}_0, \mathcal{E}_1$ are required to be computable in polynomial space.

The machine M has a designated $|\text{Accept}\rangle$ state. In vectorized notation, we let $v_{\text{Acc}} := \text{vec}(|\text{Accept}\rangle\langle\text{Accept}|)$. Since $|\text{Accept}\rangle$ is a computational basis state, v_{Acc} has a single coordinate with value 1 and is 0 elsewhere. As in Section 2.3, the machine measures after each computation step to determine whether it is in the $|\text{Accept}\rangle$ state.

We let ρ_t denote the algorithm's state after t steps, and let $v_t := \text{vec}(\rho_t)$. If we perform a standard-basis measurement after t steps, then the probability $a_{x,t}(p)$ of seeing $|\text{Accept}\rangle$ is given by

$$a_{x,t}(p) = \langle\text{Accept}|\rho_t|\text{Accept}\rangle = v_{\text{Acc}}^\dagger v_t.$$

Note that $a_{x,t}(p)$ is nondecreasing in t.

Let $a_x(p) := \lim_{t\to\infty} a_{x,t}(p)$. Then BQPSPACE/coin is the class of languages L for which there exists a BQPSPACE machine M, as well as a sequence of advice coin biases $\{p_n\}_{n\geq 1}$, such that for all $x \in \{0, 1\}^n$:

(1) If $x \in L$, then $a_x(p_n) \geq 2/3$.

(2) If $x \notin L$, then $a_x(p_n) \leq 1/3$.

2.5 The Hellman-Cover Theorem

In 1970 Hellman and Cover [13] proved the following important result (for convenience, we state only a special case).

Theorem 6 (Hellman-Cover Theorem [13]) *Let $\$_p$ be a coin with bias p, and let $M(\$_p)$ be a probabilistic finite automaton that takes as input an infinite sequence of independent flips of $\$_p$, and can 'halt and accept' or 'halt and reject' at any time step. Let $a_t(p)$ be the probability that $M(\$_p)$ has accepted after t coin flips, and let $a(p) = \lim_{t\to\infty} a_t(p)$. Suppose that $a(p) \leq 1/3$ and $a(p+\varepsilon) \geq 2/3$, for some p and $\varepsilon > 0$. Then M must have $\Omega(p(1-p)/\varepsilon)$ states.*

Let us make two remarks about Theorem 6. First, the theorem is easily seen to be essentially tight: for any p and $\varepsilon > 0$, one can construct a finite automaton with $O(p(1-p)/\varepsilon)$ states such that $a(p+\varepsilon) - a(p) = \Omega(1)$. To do so, label the automaton's states by integers in $\{-K, \ldots, K\}$, for some $K = O(p(1-p)/\varepsilon)$. Let the initial state be 0. Whenever a heads is seen, increment the state by 1 with probability $1-p$ and otherwise do nothing; whenever a tails is seen, decrement the state by 1 with probability p and otherwise do nothing. If K is ever reached, then halt and accept (i.e., guess that the bias is $p+\varepsilon$); if $-K$ is ever reached, then halt and reject (i.e., guess that the bias is p).

Second, Hellman and Cover actually proved a stronger result. Suppose we consider the relaxed model in which the finite automaton M never needs to halt, and one defines $a(p)$ to be the fraction of time that M spends in a designated subset of 'Accepting' states in the limit of infinitely many coin flips (this limit exists with probability 1). Then the lower bound $\Omega(p(1-p)/\varepsilon)$ on the number of states still holds. We will have more to say about finite automata that "accept in the limit" in Section 5.

2.6 Facts About Polynomials

We now collect some useful facts about polynomials and rational functions, and about small-space algorithms for root-finding and linear algebra. First we will need the following fact, which follows easily from L'Hôpital's Rule.

Proposition 7 *Whenever the limit exists,*

$$\lim_{z\to 0} \frac{c_0 + c_1 z + \cdots + c_m z^m}{d_0 + d_1 z + \cdots + d_m z^m} = \frac{c_k}{d_k},$$

where k is the smallest integer such that $d_k \neq 0$.

The next two facts are much less elementary. First, we state a bound on the *minimum spacing* between zeros, for a low-degree polynomial with integer coefficients.

Theorem 8 ([5, p. 359, Corollary 10.22]) *Let $P(x)$ be a degree-d univariate polynomial, with integer coefficients of bitlength at most τ. If $z, z' \in \mathbb{C}$ are distinct roots of P, then*

$$\left| z - z' \right| \geq 2^{-O(d\log d + \tau d)}.$$

In particular, if P is of degree at most $2^{\mathrm{poly}(n)}$, and has integer coefficients with absolute values bounded by $2^{\mathrm{poly}(n)}$, then $|z - z'| \geq 2^{-2^{\mathrm{poly}(n)}}$.

7

We will need to locate the zeros of univariate polynomials to high precision using a small amount of memory. Fortunately, a beautiful algorithm of Neff [14] from the 1990s (improved by Neff and Reif [15] and by Pan [18]) provides exactly what we need.

Theorem 9 ([14, 15, 18]) *There exists an algorithm that*

(i) *Takes as input a triple (P, i, j), where P is a degree-d univariate polynomial with rational[4] coefficients whose numerators and denominators are bounded in absolute value by 2^m.*

(ii) *Outputs the ith most significant bits of the real and imaginary parts of the binary expansion of the jth zero of P (in some order independent of i, possibly with repetitions).*

(iii) *Uses $O(\text{polylog}(d + i + m))$ space.*

We will also need to invert $n \times n$ matrices using $\text{polylog}(n)$ space. We can do so using an algorithm of Borodin, Cook, and Pippenger [7] (which was also used for a similar application by Aaronson and Watrous [2]).

Theorem 10 (Borodin et al. [7, Corollary 4.4]) *There exists an algorithm that*

(i) *Takes as input an $n \times n$ matrix $A = A(p)$, whose entries are rational functions in p of degree $\text{poly}(n)$, with the coefficients specified to $\text{poly}(n)$ bits of precision.*

(ii) *Computes $\det(A)$ (and as a consequence, also the (i, j) entry of A^{-1} for any given coordinates (i, j), assuming that A is invertible).*

(iii) *Uses $\text{poly}(n)$ time and $\text{polylog}(n)$ space.*

Note that the algorithms of [7, 14, 15, 18] are all stated as *parallel* (NC) algorithms. However, any parallel algorithm can be converted into a space-efficient algorithm, using a standard reduction due to Borodin [6].

3 Quantum Mechanics Nullifies the Hellman-Cover Theorem

We now show that the quantum analogue of the Hellman-Cover Theorem (Theorem 6) is false. Indeed, we will show that for any fixed $\varepsilon > 0$, there exists a quantum finite automaton with only 2 states that can distinguish a coin with bias $1/2$ from a coin with bias $1/2 + \varepsilon$, with bounded probability of error independent of ε. Furthermore, this automaton is even a *halting* automaton, which halts with probability 1 and enters either an $|\text{Accept}\rangle$ or a $|\text{Reject}\rangle$ state.

The key idea is that, in this setting, a single qubit can be used as an "analog counter," in a way that a classical probabilistic bit cannot. Admittedly, our result would fail were the qubit subject to noise or decoherence, as it would be in a realistic physical situation.

Let ρ_0 be the designated starting state of the automaton, and let $\rho_1, \rho_2, \ldots,$ be defined as $\rho_{t+1} = \mathcal{E}_p \rho_t$, with notation as in Section 2.4. Let

$$a(p) := \lim_{n \to \infty} \langle \text{Accept} | \mathcal{E}_p^n (\rho_0) | \text{Accept} \rangle$$

be the limiting probability of acceptance. This limit exists, as argued in Section 2.4.

We now prove Theorem 1, which we restate for convenience.

---

[4]Neff's original algorithm assumes polynomials with *integer* coefficients; the result for rational coefficients follows easily by clearing denominators.
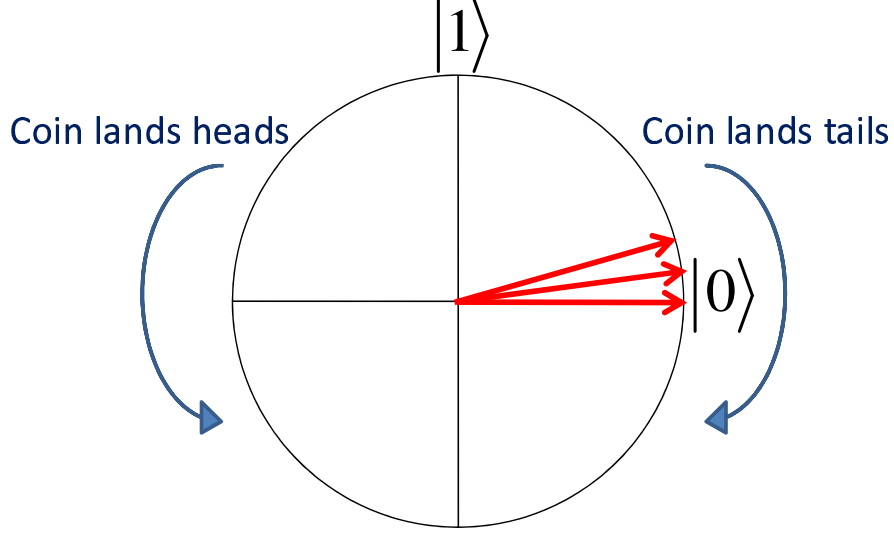
Figure 1: A quantum finite automaton that distinguishes a $p = 1/2$ coin from a $p = 1/2 + \varepsilon$ coin, essentially by using a qubit as an analog counter.

> *Fix $p \in [0, 1]$ and $\varepsilon > 0$. Then there exists a quantum finite automaton M with two states (not counting the $|\text{Accept}\rangle$ and $|\text{Reject}\rangle$ states), such that $a (p + \varepsilon) - a (p) \geq \beta$ for some constant β independent of ε. (For example, $\beta = 0.0117$ works.)*

Proof of Theorem 1. The state of M will belong to the Hilbert space spanned by $\{|0\rangle, |1\rangle, |\text{Accept}\rangle, |\text{Reject}\rangle\}$. The initial state is $|0\rangle$. Let

$$U (\theta) := \left(\begin{array}{cc} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{array} \right)$$

be a unitary transformation that rotates counterclockwise by θ, in the "counter subspace" spanned by $|0\rangle$ and $|1\rangle$. Also, let A and B be positive integers to be specified later. Then the finite automaton M runs the following procedure:

(1) If a 1 bit is encountered (i.e., the coin lands heads), apply $U (\varepsilon (1 - p) / A)$.

(2) If a 0 bit is encountered (i.e., the coin lands tails), apply $U (-\varepsilon p / A)$.

(3) With probability $\alpha := \varepsilon^2 / B$, "measure" (that is, move all probability mass in $|0\rangle$ to $|\text{Reject}\rangle$ and all probability mass in $|1\rangle$ to $|\text{Accept}\rangle$); otherwise do nothing.

We now analyze the behavior of M. For simplicity, let us first consider steps (1) and (2) only. In this case, we can think of M as taking a random walk in the space of possible angles between $|0\rangle$ and $|1\rangle$. In particular, after t steps, M's state will have the form $\cos \theta_t |0\rangle + \sin \theta_t |1\rangle$, for some angle $\theta_t \in \mathbb{R}$. (As we follow the walk, we simply let θ_t increase or decrease without bound, rather than confining it to a range of size 2π.) Suppose the coin's bias is p. Then after t steps,

$$\mathbb{E} [\theta_t] = pt \cdot \frac{\varepsilon}{A} (1 - p) + (1 - p) t \cdot \left(-\frac{\varepsilon}{A} p \right) = 0.$$

On the other hand, suppose the bias is $q = p + \varepsilon$. Then

$$\mathbb{E}\left[\theta_t\right] = qt \cdot \frac{\varepsilon}{A}\left(1 - p\right) + \left(1 - q\right)t \cdot \left(-\frac{\varepsilon}{A}p\right)$$

$$= \frac{\varepsilon}{A}t \cdot \left[q\left(1 - p\right) - p\left(1 - q\right)\right]$$

$$= \frac{\varepsilon^2 t}{A}.$$

So in particular, if $t = K/\varepsilon^2$ for some constant K, then $\mathbb{E}\left[\theta_t\right] = K/A$. However, we also need to understand the variance of the angle, $\mathrm{Var}\left[\theta_t\right]$. If the bias is p, then by the independence of the coin flips,

$$\mathrm{Var}\left[\theta_t\right] = t \cdot \mathrm{Var}\left[\theta_1\right]$$

$$= t \cdot \left[p\left(\frac{\varepsilon}{A}\left(1 - p\right)\right)^2 + \left(1 - p\right)\left(\frac{\varepsilon}{A}p\right)^2\right]$$

$$\leq \frac{\varepsilon^2 t}{A^2},$$

and likewise if the bias is $q = p + \varepsilon$. If $t = K/\varepsilon^2$, this implies that $\mathrm{Var}\left[\theta_t\right] \leq K/A^2$ in both cases. We now incorporate step (3). Let T be the number of steps before M halts (that is, before its state gets measured). Then clearly $\Pr\left[T = t\right] = \alpha\left(1 - \alpha\right)^t$. Also, let $u := K/\varepsilon^2$ for some K to be specified later. Then if the bias is p, we can upper-bound M's acceptance probability $a\left(p\right)$ as

$$a\left(p\right) = \sum_{t=1}^{\infty} \Pr\left[T = t\right] \cdot \mathbb{E}\left[\sin^2\theta_t \mid t\right]$$

$$\leq \Pr\left[T > u\right] + \sum_{t=1}^{u} \Pr\left[T = t\right] \cdot \mathbb{E}\left[\sin^2\theta_t \mid t\right]$$

$$\leq \Pr\left[T > u\right] + \sum_{t=1}^{u} \Pr\left[T = t\right] \cdot \mathbb{E}\left[\theta_t^2 \mid t\right]$$

$$\leq \left(1 - \alpha\right)^u + \mathbb{E}\left[\theta_u^2 \mid u\right]$$

$$\leq \left(1 - \frac{\varepsilon^2}{B}\right)^{B/\varepsilon^2 \cdot K/B} + \frac{\varepsilon^2 u}{A^2}$$

$$\leq e^{-K/B} + \frac{K}{A^2}.$$

Here the third line uses $\sin x \leq x$, while the fourth line uses the fact that $\mathbb{E}\left[\theta_t^2\right]$ is nondecreasing for an unbiased random walk. So long as $A^2 \geq B$, we can minimize the final expression by setting $K := B \ln\left(A^2/B\right)$, in which case we have

$$a\left(p\right) \leq \frac{B}{A^2}\left(1 + \ln\left(\frac{A^2}{B}\right)\right).$$

On the other hand, suppose the bias is $p + \varepsilon$. Set $v := L/\varepsilon^2$ where $L := \pi A/4$. Then for all $t \leq v$,

we have

$$\Pr\left[|\theta_t| > \pi/2 \mid t\right] \le \Pr\left[\left|\theta_t - \frac{\varepsilon^2 t}{A}\right| > \frac{\pi}{2} - \frac{\varepsilon^2 t}{A} \mid t\right]$$

$$< \frac{\varepsilon^2 t/A^2}{(\pi/2 - \varepsilon^2 t/A)^2}$$

$$\le \frac{\varepsilon^2 v/A^2}{(\pi/2 - \varepsilon^2 v/A)^2}$$

$$= \frac{4}{\pi A}$$

where the second line uses Chebyshev's inequality. Also, let $\Delta_t := \theta_t - \varepsilon^2 t/A$. Then for all $t \le v$ we have

$$\mathbb{E}\left[\theta_t^2 \mid t\right] = \mathbb{E}\left[\left(\frac{\varepsilon^2 t}{A} + \Delta_t\right)^2 \mid t\right]$$

$$= \frac{\varepsilon^4 t^2}{A^2} + \mathbb{E}\left[\Delta_t^2 \mid t\right] + 2\frac{\varepsilon^2 t}{A}\mathbb{E}\left[\Delta_t \mid t\right]$$

$$\ge \frac{\varepsilon^4 t^2}{A^2}.$$

Putting the pieces together, we can lower-bound $a\,(p+\varepsilon)$ as

$$a\,(p+\varepsilon) = \sum_{t=1}^{\infty} \Pr\left[T = t\right] \cdot \mathbb{E}\left[\sin^2 \theta_t \mid t\right]$$

$$\ge \sum_{t=1}^{v} \Pr\left[T = t\right] \cdot \mathbb{E}\left[\sin^2 \theta_t \mid t\right]$$

$$\ge \sum_{t=1}^{v} \Pr\left[T = t\right] \cdot \Pr\left[|\theta_t| \le \pi/2 \mid t\right] \cdot \mathbb{E}\left[\theta_t^2/3 \mid t\right]$$

$$\ge \sum_{t=1}^{v} \alpha\,(1-\alpha)^t \cdot \left(1 - \frac{4}{\pi A}\right) \cdot \frac{\varepsilon^4 t^2}{3A^2}$$

$$= \left(1 - \frac{4}{\pi A}\right) \frac{\varepsilon^4 \alpha}{3A^2} \sum_{t=1}^{L/\varepsilon^2} \left(1 - \frac{\varepsilon^2}{B}\right)^t t^2$$

$$\ge \left(1 - \frac{4}{\pi A}\right) \frac{\varepsilon^4 \alpha}{3A^2 e^{L/B}} \sum_{t=1}^{L/\varepsilon^2} t^2$$

$$\ge \left(1 - \frac{4}{\pi A}\right) \frac{\varepsilon^6}{3A^2 B e^{L/B}} \cdot \frac{\left(L/\varepsilon^2\right)^3}{6}$$

$$= \left(1 - \frac{4}{\pi A}\right) \frac{L^3}{18A^2 B e^{L/B}}$$

$$= \left(1 - \frac{4}{\pi A}\right) \frac{\pi^3 A}{1152 B e^{\pi A/4B}}.$$

11

Here the third line uses the fact that $\sin^2 x \geq x^2/3$ for all $|x| \leq \pi/2$. If we now choose (for example) $A = 10000$ and $B = 7500$, then we have $a(p) \leq 0.0008$ and $a(p + \varepsilon) \geq 0.0125$, whence $a(p + \varepsilon) - a(p) \geq 0.0117$. ∎

We can strengthen Theorem 1 to ensure that $a(p) \leq \delta$ and $a(p + \varepsilon) \geq 1 - \delta$ for any desired error probability $\delta > 0$. We simply use standard amplification, which increases the number of states in M to $O(\text{poly}(1/\delta))$ (or equivalently, the number of qubits to $O(\log(1/\delta))$).

4 Upper-Bounding the Power of Advice Coins

In this section we prove Theorem 2, that $\mathsf{BQPSPACE/coin} = \mathsf{BPPSPACE/coin} = \mathsf{PSPACE/poly}$. We start with the easy half:

Proposition 11 $\mathsf{PSPACE/poly} \subseteq \mathsf{BPPSPACE/coin} \subseteq \mathsf{BQPSPACE/coin}$.

Proof. Given a polynomial-size advice string $w_n \in \{0,1\}^{s(n)}$, we encode w_n into the first $s(n)$ bits of the binary expansion of an advice bias $p_n \in [0,1]$. Then by flipping the coin $\$_{p_n}$ sufficiently many times ($O(2^{2s(n)})$ trials suffice) and tallying the fraction of heads, a Turing machine can recover w_n with high success probability. Counting out the desired number of trials and determining the fraction of heads seen can be done in space $O(\log(2^{2s(n)})) = O(s(n)) = O(\text{poly}(n))$. Thus we can simulate a $\mathsf{PSPACE/poly}$ machine with a $\mathsf{BPPSPACE/coin}$ machine. ∎

The rest of the section is devoted to showing that $\mathsf{BQPSPACE/coin} \subseteq \mathsf{PSPACE/poly}$. First we give some lemmas about quantum polynomial-space advice coin algorithms. Let M be such an algorithm. Suppose M uses $s(n) = \text{poly}(n)$ qubits of memory, and has $S = 2^{s(n)}$ states. Let $\mathcal{E}_0, \mathcal{E}_1, \mathcal{E}_p$ be the superoperators for M as described in Section 2.4. Recalling the vectorized notation from Section 2.4, let $B_p := \text{mat}(\mathcal{E}_p)$. Let $\rho_{x,t}(p)$ be the state of M after t coin flips steps on input x and coin bias p, and let $v_{x,t}(p) := \text{vec}(\rho_{x,t}(p))$. Let

$$a_{x,t}(p) := v_{\text{Acc}}^\dagger v_{x,t}(p)$$

be the probability that M is in the $|\text{Accept}\rangle$ state, if measured after t steps. Let $a_x(p) := \lim_{t \to \infty} a_{x,t}(p)$. As discussed in Section 2.4, the quantities $a_{x,t}(p)$ are nondecreasing in t, so the limit $a_x(p)$ is well-defined.

We now show that—except possibly at a finite number of values—$a_x(p)$ is actually a *rational function* of p, whose degree is at most the number of states.

Lemma 12 *There exist polynomials $Q(p)$ and $R(p) \neq 0$, of degree at most $S^2 = 2^{\text{poly}(n)}$ in p, such that*

$$a_x(p) = \frac{Q(p)}{R(p)}$$

holds whenever $R(p) \neq 0$. Moreover, Q and R have rational coefficients that are computable in $\text{poly}(n)$ space given $x \in \{0,1\}^n$ and the index of the desired coefficient.

Proof. Throughout, we suppress the dependence on x for convenience, so that $a(p) = \lim_{t \to \infty} a_t(p)$ is simply the limiting acceptance probability of a finite automaton $M(\$_p)$ given a coin with bias p.

Following Aaronson and Watrous [2], for $z \in (0,1)$ define the matrix $\Lambda_{z,p} \in \mathbb{C}^{S^2 \times S^2}$ by

$$\Lambda_{z,p} := z \left[I - (1-z) B_p \right]^{-1}.$$

The matrix $I - (1-z)B_p$ is invertible, since $z > 0$ and all eigenvalues of B_p have absolute value at most 1.[5] Using Cramer's rule, we can represent each entry of $\Lambda_{z,p}$ in the form $\frac{f(z,p)}{g(z,p)}$, where f

---

[5]For the latter fact, see [19] and [2, p. 10, footnote 1].

and g are bivariate polynomials of degree at most S^2 in both z and p, and $g(z,p)$ is not identically zero. Note that by collecting terms, we can write

$$f(z,p) = c_0(p) + c_1(p)z + \cdots + c_{S^2}(p)z^{S^2}$$
$$g(z,p) = d_0(p) + d_1(p)z + \cdots + d_{S^2}(p)z^{S^2},$$

for some coefficients $c_0, \ldots, c_{S^2}$ and $d_0, \ldots, d_{S^2}$. Now let

$$\Lambda_p := \lim_{z \to 0} \Lambda_{z,p}. \tag{1}$$

Aaronson and Watrous [2] showed that Λ_p is precisely the matrix representation $\mathrm{mat}(\mathcal{E}_{\text{fix}})$ of the superoperator $\mathcal{E}_{\text{fix}}$ associated to $\mathcal{E} := \mathcal{E}_p$ by Theorem 4. Thus we have

$$B_p(\Lambda_p v) = \Lambda_p v$$

for all $v \in \mathbb{C}^{S^2}$.

Now, the entries of $\Lambda_{z,p}$ are bivariate rational functions, which have absolute value at most 1 for all z, p. Thus the limit in equation (1) must exist, and the coeffients c_k, d_k can be computed in polynomial space using Theorem 10.

We claim that every entry of Λ_p can be represented as a rational function of p of degree at most S^2 (a representation valid for all but finitely many p), and that the coefficients of this rational function are computable in polynomial space. To see this, fix some $i, j \in [S]$, and let $(\Lambda_p)_{ij}$ denote the $(i,j)^{th}$ entry of Λ_p. By the above, $(\Lambda_p)_{ij}$ has the form

$$(\Lambda_p)_{ij} = \lim_{z \to 0} \frac{f(z,p)}{g(z,p)} = \lim_{z \to 0} \frac{c_0(p) + c_1(p)z + \cdots + c_{S^2}(p)z^{S^2}}{d_0(p) + d_1(p)z + \cdots + d_{S^2}(p)z^{S^2}}.$$

By Proposition 7, the above limit (whenever it exists) equals $c_k(p)/d_k(p)$, where k is the smallest integer such that $d_k(p) \neq 0$. Now let k^* be the smallest integer such that d_{k^*} is not the identically-zero polynomial. Then $d_{k^*}(p)$ has only finitely many zeros. It follows that $(\Lambda_p)_{ij} = c_{k^*}(p)/d_{k^*}(p)$ except when $d_{k^*}(p) = 0$, which is what we wanted to show. That the coefficients are rational and computable in polynomial space follows by construction: we can loop through all k until we find k^* as above, and then compute the coefficients of $c_{k^*}(p)$ and $d_{k^*}(p)$.

Finally, we claim that we can write A's limiting acceptance probability $a(p)$ as

$$a(p) = v_{\text{Acc}}^\dagger \Lambda_p v_0, \tag{2}$$

where v_0 is the vectorized initial state of A (independent of p). It will follow from equation (2) that $a(p)$ has the desired rational-function representation, since the map $\Lambda_p \to v_{\text{Acc}}^\dagger \Lambda_p v_0$ is linear in the entries of Λ_p and can be performed in polynomial space.

To establish equation (2), consider the Taylor series expansion for $\Lambda_{z,p}$,

$$\Lambda_{z,p} = \sum_{t \geq 0} z(1-z)^t B_p^t,$$

valid for $z \in (0,1)$ (see [2] for details). The equality

$$\sum_{t \geq 0} z(1-z)^t = 1, \quad z \in (0,1),$$

implies that $v_{\text{Acc}}^{\dagger}\Lambda_{z,p}v_0$ is a weighted average of the t-step acceptance probabilities $a_t(p)$, for $t \in \{0, 1, 2, \ldots\}$. Letting $z \to 0$, the weight on each individual step approaches 0. Since $\lim_{t\to\infty} a_t(p) = a(p)$, we obtain equation (2). $\blacksquare$

The next lemma lets us "patch up" the finitely many singularities, and show that $a_x(p)$ is a rational function in the entire open interval $(0, 1)$.[6]

Lemma 13 $a_x(p)$ *is continuous for all $p \in (0, 1)$.*

Proof. Once again we suppress the dependence on x, so that $a(p) = \lim_{t\to\infty} a_t(p)$ is just the limiting acceptance probability of a finite automaton $M(\$_p)$.

To show that $a(p)$ is continuous on $(0, 1)$, it suffices to show that $a(p)$ is continuous on every closed subinterval $[p_1, p_2]$ such that $0 < p_1 < p_2 < 1$. We will prove this by proving the following claim:

(*) *For every subinterval $[p_1, p_2]$ and every $\delta > 0$, there exists a time t (not depending on p) such that $a_t(p) \geq a(p) - \delta$ for all $p \in [p_1, p_2]$.*

Claim (*) implies that $a(p)$ can be uniformly approximated by continuous functions on $[p_1, p_2]$, and hence is continuous itself on $[p_1, p_2]$.

We now prove claim (*). First, call a mixed state ρ *dead for bias p* if $M(\$_p)$ halts with probability 0 when run with ρ as its initial state. Now, the superoperator applied by $M(\$_p)$ at each time step is $\mathcal{E}_p = p\mathcal{E}_1 + (1-p)\mathcal{E}_0$. This means that ρ is dead for any bias $p \in (0, 1)$, if and only if ρ is dead for bias $p = 1/2$. So we can simply refer to such a ρ as *dead*, with no dependence on p.

Recall that $B_p := \text{mat}(\mathcal{E}_p)$. Observe that ρ is dead if and only if

$$v_{\text{Acc}}^{\dagger} B_{1/2}^t \text{vec}(\rho) = 0$$

for all $t \geq 0$. In particular, it follows that there exists a "dead subspace" D of $\mathbb{C}^S$, such that a pure state $|\psi\rangle$ is dead if and only if $|\psi\rangle \in D$. (A mixed state $\rho = \sum_i p_i |\psi_i\rangle\langle\psi_i|$ is dead if and only if $|\psi_i\rangle \in D$ for all i such that $p_i > 0$.) By its definition, D is orthogonal to the $|\text{Accept}\rangle$ state. Define the "live subspace," L, to be the orthogonal complement of $|\text{Accept}\rangle$ and D.

Let P be the projector onto L, and let $v_{\text{Live}} := \text{vec}(P)$. Also, recalling that v_0 is the vectorized initial state of M, let

$$g_t(p) := v_{\text{Live}}^{\dagger} B_p^t v_0$$

be the probability that $M(\$_p)$ is "still alive" if measured after t steps—i.e., that M has neither accepted nor entered the dead subspace. Clearly $a(p) \leq a_t(p) + g_t(p)$.

Thus, to prove claim (*), it suffices to prove that for all $\delta > 0$, there exists a t (not depending on p) such that $g_t(p) \leq \delta$ for all $p \in [p_1, p_2]$. First, let U be the set of all ρ supported only on the live subspace L, and notice that U is compact. Therefore, by Lemma 5 (the "Leaky Subspace Lemma"), there exists a constant $c_1 > 0$ such that, for all $\rho \in U$,

$$\left(v_{\text{Acc}}^{\dagger} + v_{\text{Dead}}^{\dagger}\right) B_{p_1}^{S^2} \text{vec}(\rho) \geq c_1$$

and hence

$$v_{\text{Live}}^{\dagger} B_{p_1}^{S^2} \text{vec}(\rho) \leq 1 - c_1.$$

---

[6]Note that there could still be singularities at $p = 0$ and $p = 1$, and this is not just an artifact of the proof! For example, consider a finite automaton that accepts when and only when it sees 'heads.' The acceptance probability of such an automaton satisfies $a(0) = 0$, but $a(p) = 1$ for all $p \in (0, 1]$.

Likewise, there exists a $c_2 > 0$ such that, for all $\rho \in U$,

$$v_{\text{Live}}^{\dagger} B_{p_2}^{S^2} \text{vec} (\rho) \leq 1 - c_2.$$

Let $c := \min \{c_1, c_2\}$. Then by convexity, for all $p \in [p_1, p_2]$ and all $\rho \in U$, we have

$$v_{\text{Live}}^{\dagger} B_p^{S^2} \text{vec} (\rho) \leq 1 - c,$$

and hence

$$v_{\text{Live}}^{\dagger} B_p^{S^2 t} \text{vec} (\rho) \leq (1 - c)^t$$

for all $t \geq 0$. This means that, to ensure that $g_t (p) \leq \delta$ for all $p \in [p_1, p_2]$ simultaneously, we just need to choose t large enough that $(1 - c)^{t/S^2} \leq \delta$. This proves claim (*). ■

We are now ready to complete the proof of Theorem 2. Let L be a language in BQPSPACE/coin, which is decided by the quantum polynomial-space advice-coin machine $M (x, \$_p)$ on advice coin biases $\{p_n\}_{n \geq 1}$. We will show that $L \in$ BQPSPACE/poly $=$ PSPACE/poly.

It may not be possible to perfectly specify the bias p_n using poly (n) bits of advice. Instead, we use our advice string to simulate access to a second bias r_n that is "almost as good" as p_n. This is achieved by the following lemma.

Lemma 14 *Fixing $L, M, \{p_n\}$ as above, there exists a classical polynomial-space algorithm R, as well as a family $\{w_n\}_{n \geq 1}$ of polynomial-size advice strings, for which the following holds. Given an index $i \leq 2^{\text{poly}(n)}$, the computation $R (w_n, i)$ outputs the ith bit of a real number $r_n \in (0, 1)$, such that for all $x \in \{0, 1\}^n$,*

(i) If $x \in L$, then $\Pr [M (x, \$_{r_n}) \text{ accepts}] \geq 3/5$.

(ii) If $x \notin L$, then $\Pr [M (x, \$_{r_n}) \text{ accepts}] \leq 2/5$.

(iii) The binary expansion of r_n is identically zero, for sufficiently large indices $j \geq h(n) = 2^{\text{poly}(n)}$.

Once Lemma 14 is proved, showing the containment $L \in$ BQPSPACE/poly is easy. First, we claim that using the advice family $\{w_n\}$, we can simulate access to r_n-biased coin flips, as follows. Let $r_n = 0.b_1 b_2 \ldots$ denote the binary expansion of r_n.

```
given w_n
j := 0
while j < h(n)
    let z_j ∈ {0,1} be random
    b_j := R(w_n, j)
    if z_j < b_j then output 1
    else if z_j > b_j then output 0
    else j := j + 1
output 0
```

Observe that this algorithm, which runs in polynomial space, outputs 1 if and only if

$$0.z_1 z_2 \ldots z_{h(n)} < 0.b_1 b_2 \ldots b_{h(n)} = r_n,$$

and this occurs with probability r_n. Thus we can simulate r_n-biased coin flips as claimed.

We define a BQPSPACE/poly machine M' that takes $\{w_n\}$ from Lemma 14 as its advice. Given an input $x \in \{0,1\}^n$, the machine M' simulates $M(x, \$_{r_n})$, by generating r_n-biased coin flips using the method described above. Then M' is a BQPSPACE/poly algorithm for L by parts (i) and (ii) of Lemma 14, albeit with error bounds $(2/5, 3/5)$. The error bounds can be boosted to $(1/3, 2/3)$ by running several independent trials. So $L \in$ BQPSPACE/poly = PSPACE/poly, completing the proof of Theorem 2.

Proof of Lemma 14. Fix an input length $n > 0$, and let $p^* := p_n$. For $x \in \{0,1\}^n$, recall that $a_x(p)$ denotes the acceptance probability of $M(x, \$_p)$. We are interested in the way $a_x(p)$ oscillates as we vary p. Define a *transition pair* to be an ordered pair $(x, p) \in \{0,1\}^n \times (0,1)$ such that $a_x(p) \in \{2/5, 3/5\}$. It will be also be convenient to define a larger set of *potential transition pairs*, denoted $\mathcal{P} \subseteq \{0,1\}^n \times [0,1)$, that contains the transition pairs; the benefit of considering this larger set is that its elements will be easier to enumerate. We defer the precise definition of $\mathcal{P}$.

The advice string w_n will simply specify *the number of distinct potential transition pairs (y, p) such that $p \leq p^*$*. We first give a high-level pseudocode description of the algorithm R; after proving that parts (i) and (ii) of Lemma 14 are met by the algorithm, we will fill in the algorithmic details to show that the pseudocode can be implemented in PSPACE, and that we can satisfy part (iii) of the Lemma.

The pseudocode for R is as follows:

```
given (w_n, i)
for all (y, p) ∈ P
    s := 0
    for all (z, q) ∈ P
        if q ≤ p then s := s + 1
    next (z, q)
    if s = w_n then
        let r_n := p + ε (for some small ε = 2^(-2^poly(n)))
        output the i^th bit of r_n
    end if
next (y, p)
```

We now prove that parts (i) and (ii) of Lemma 14 are satisfied. We call $p \in [0,1)$ a *transition value* if (y, p) is a transition pair for some $y \in \{0,1\}^n$, and we call p a *potential transition value* if $(y, p) \in \mathcal{P}$ for some $y \in \{0,1\}^n$. Then by definition of w_n, the value r_n produced above is equal to $p_0 + \varepsilon$, where $p_0 \in [0,1)$ is the largest potential transition value less than or equal to p^*. (Note that 0 will always be a potential transition value, so this is well-defined.)

When we define $\mathcal{P}$, we will argue that any distinct potential transition values p_1, p_2 satisfy

$$\min\{|p_1 - p_2|, 1 - p_2\} \geq 2^{-2^{\mathrm{poly}(n)}}. \tag{3}$$

It follows that if $\varepsilon = 2^{-2^{\mathrm{poly}(n)}}$ is suitably small, then $r_n < 1$, and there is no potential transition value lying in the range $(p_0, r_n]$. Also, there are no potential transition values in the interval (p_0, p^*).

Now fix any $x \in \{0,1\}^n \cap L$. Since M is a BQPSPACE/coin machine for L with bias p^*, we have $a_x(p^*) \geq 2/3$. If $a_x(r_n) < 3/5$, then Lemma 13 implies that there must be a transition value in the open interval between p^* and r_n. But there are no such transition values. Thus $a_x(r_n) \geq 3/5$. Similarly, if $x \in \{0,1\}^n \setminus L$, then $a_x(r_n) \leq 2/5$. This establishes parts (i) and (ii) of Lemma 14.

Now we formally define the potential transition pairs $\mathcal{P}$. We include $(0^n, 0)$ in $\mathcal{P}$, guaranteeing that 0 is a potential transition value as required. Now recall, by Lemma 12, that for each $x \in \{0, 1\}^n$, the acceptance probability $a_x(p)$ is a rational function $Q_x(p)/R_x(p)$ of degree $2^{\text{poly}(n)}$, for all but finitely many $p \in (0, 1)$. Therefore, the function $(a_x(p) - 3/5)(a_x(p) - 2/5)$ also has a rational-function representation:

$$\frac{U_x(p)}{V_x(p)} = \left(a_x(p) - \frac{3}{5}\right)\left(a_x(p) - \frac{2}{5}\right),$$

valid for all but finitely many p. We will include in $\mathcal{P}$ all pairs (x, p) for which $U_x(p) = 0$. It follows from Lemmas 12 and 13 that $\mathcal{P}$ contains all transition pairs, as desired.

We can now establish equation (3). Fix any distinct potential transition values $p_1 < p_2$ in $[0, 1)$. Since $p_2 \neq 0$ is a potential transition value, there is some x_2 such that $(x_2, p_2) \in \mathcal{P}$. If $p_1 = 0$, then p_1, p_2 are distinct roots of the polynomial $pU_{x_2}(p)$, whence $|p_1 - p_2| \geq 2^{-2^{\text{poly}(n)}}$ by Theorem 8. Similarly, if $p_1 > 0$, then $(x_1, p_1) \in \mathcal{P}$ for some x_1. We observe that p_1, p_2 are common roots of $U_{x_1}(p)U_{x_2}(p)$, from which it again follows that $|p_1 - p_2| \geq 2^{-2^{\text{poly}(n)}}$. Finally, $1 - p_2 \geq 2^{-2^{\text{poly}(n)}}$ follows since 1 and p_2 are distinct roots of $(1 - p)U_{x_2}(p)$. Thus equation (3) holds.

Next we show that the pseudocode can be implemented in PSPACE. Observe first that the degrees of U_x, V_x are $2^{\text{poly}(n)}$, with rational coefficients having numerator and denominator bounded by $2^{\text{poly}(n)}$. Moreover, the coefficients of U_x, V_x are computable in PSPACE from the coefficients of Q_x, R_x, and these coefficients are themselves PSPACE-computable. To loop over the elements of $\mathcal{P}$ as in the for-loops of the pseudocode, we can perform an outer loop over $y \in \{0, 1\}^n$ and an inner loop over the zeros of U_y. These zeros are indexed by Neff's algorithm (Theorem 9) and can be looped over with that indexing. The algorithm of Theorem 9 may return duplicate roots, but these can be identified and removed by comparing each root in turn to all previously visited roots. For each pair of distinct zeros of U_y differ in their binary expansion to a sufficiently large $2^{\text{poly}(n)}$ number of bits (by Theorem 8), and Theorem 9 allows us to compare such bits in polynomial space.

Similarly, if $(y, p), (z, q) \in \mathcal{P}$ then we can determine in PSPACE whether $q \leq p$, as required. The only remaining implementation step is to produce the value r_n in PSPACE, in such a way that part (iii) of Lemma 14 is satisfied. Given the value p chosen by the inner loop, and the index $i \leq 2^{\text{poly}(n)}$, we need to produce the ith bit of a value $r_n \in \left(p, p + 2^{-2^{\text{poly}(n)}}\right)$, such that the binary expansion of r_n is identically zero for sufficiently large $j \geq h(n) = 2^{\text{poly}(n)}$. But this is easily done, since we can compute any desired j^{th} bit of p, for $j \leq 2^{\text{poly}(n)}$, in polynomial space. ∎

5 Distinguishing Problems for Finite Automata

The *distinguishing problem*, as described in Section 1, is a natural problem with which to investigate the power of restricted models of computation. The basic task is to distinguish a coin of bias p from a coin of bias $p + \varepsilon$, using a finite automaton with a bounded number of states. Several variations of this problem have been explored [13, 10], which modify either the model of computation or the mode of acceptance. A basic question to explore in each case is whether the distinguishing task can be solved by a finite automaton whose number of states is independent of the value ε (for fixed p, say).

Variations of interest include:

(1) *Classical vs. quantum finite automata.* We showed in Section 3 that, in some cases, quantum finite automata can solve the distinguishing problem where classical ones cannot.
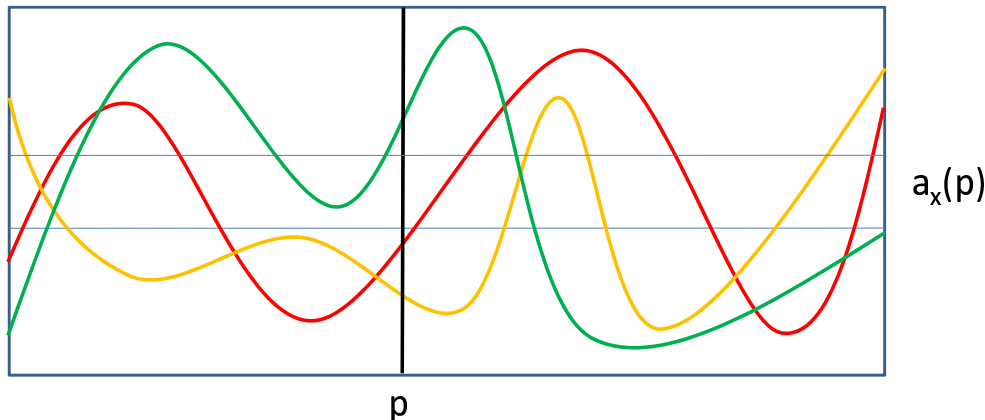
Figure 2: Graphical depiction of the proof of Theorem 2, that $\mathsf{BQPSPACE/coin} = \mathsf{PSPACE/poly}$. For each input $x \in \{0,1\}^n$, the acceptance probability of the $\mathsf{BQPSPACE/coin}$ machine is a rational function $a_x(p)$ of the coin bias p, with degree at most $2^{\mathrm{poly}(n)}$. Such a function can cross the $a_x(p) = 2/5$ or $a_x(p) = 3/5$ lines at most $2^{\mathrm{poly}(n)}$ times. So even considering *all* 2^n inputs x, there can be at most $2^{\mathrm{poly}(n)}$ crossings in total. It follows that, if we want to specify whether $a_x(p) < 2/5$ or $a_x(p) > 3/5$ for all 2^n inputs x simultaneously, it suffices to give only $\mathrm{poly}(n)$ bits of information about p (for example, the total number of crossings to the left of p).

(2) *ε-dependent vs. ε-independent automata.* Can a single automaton M distinguish $\$_p$ from $\$_{p+\varepsilon}$ for every $\varepsilon > 0$, or is a different automaton M_ε required for different ε?

(3) *Bias 0 vs. bias 1/2.* Is the setting $p = 0$ easier than the setting $p = 1/2$?

(4) *Time-dependent vs. time-independent automata.* An alternative, "nonuniform" model of finite automata allows their state-transition function to depend on the *current time step* $t \geq 0$, as well as on the current state and the current bit being read. This dependence on t can be arbitrary; the transition function is not required to be computable given t.

(5) *Acceptance by halting vs. 1-sided acceptance vs. acceptance in the limit.* How does the finite automaton register its final decision? A first possibility is that the automaton halts and enters an $|\mathrm{Accept}\rangle$ state if it thinks the bias is $p + \varepsilon$, or halts and enters a $|\mathrm{Reject}\rangle$ state if it thinks the bias is p. A second possibility, which corresponds to the model considered for most of this paper, is that the automaton halts and enters an $|\mathrm{Accept}\rangle$ state if it thinks the bias is $p+\varepsilon$, but can reject by simply never halting. A third possibility is that the automaton *never* needs to halt. In this third model, we designate some subset of the states as "accepting states," and let a_t be the probability that the automaton would be found in an accepting state, were it measured at the tth time step. Then the automaton is said to *accept in the limit* if $\liminf_{t\to\infty}(a_1+\ldots+a_t)/t \geq 2/3$, and to *reject in the limit* if $\limsup_{t\to\infty}(a_1+\cdots+a_t)/t \leq 1/3$. The automaton solves the distinguishing problem if it accepts in the limit on a coin of bias $p + \varepsilon$, and rejects in the limit on a coin of bias p.

For almost every possible combination of the above, we can determine whether the distinguishing problem can be solved by an automaton whose number of states is independent of ε, by using the results and techniques of [13, 10] as well as the present paper. The situation is summarized in the following two tables.

Classical case   *Coin distinguishing task*

Type of automaton	$\frac{1}{2}$ vs. $\frac{1}{2}+\varepsilon$			0 vs. ε		
	Halt	1-Sided	Limit	Halt	1-Sided	Limit
Fixed	No	No	No	No	Yes (easy)	Yes
ε-dependent	No	No	No [13]	Yes (easy)	Yes	Yes
Time-dependent	No	Yes [10]	Yes	No	Yes	Yes
ε,time-dependent	Yes [10]	Yes	Yes	Yes	Yes	Yes

Quantum case   *Coin distinguishing task*

Type of automaton	$\frac{1}{2}$ vs. $\frac{1}{2}+\varepsilon$			0 vs. ε		
	Halt	1-Sided	Limit	Halt	1-Sided	Limit
Fixed	No	No (here)	?	No	Yes	Yes
ε-dependent	**Yes (here)**	**Yes**	**Yes**	Yes	Yes	Yes
Time-dependent	No	Yes	Yes	No (easy)	Yes	Yes
ε,time-dependent	Yes	Yes	Yes	Yes	Yes	Yes

Let us briefly discuss the possibility and impossibility results.

(1) Hellman and Cover [13] showed that a classical finite automaton needs $\Omega\left(1/\varepsilon\right)$ states to distinguish $p = 1/2$ from $p = 1/2+\varepsilon$, even if the transition probabilities can depend on ε and the automaton only needs to succeed in the limit.

(2) By contrast, Theorem 1 shows that an ε-dependent quantum finite automaton with only *two* states can distinguish $p = 1/2$ from $p = 1/2 + \varepsilon$ for any $\varepsilon > 0$, even if the automaton needs to halt.

(3) Cover [10] gave a construction of a 4-state *time-dependent* (but ε-independent) classical finite automaton that distinguishes $p = 1/2$ from $p = 1/2+\varepsilon$, for any $\varepsilon > 0$, in the limit of infinitely many coin flips. This automaton can even be made to halt in the case $p = 1/2 + \varepsilon$.

(4) It is easy to modify Cover's construction to get, for any *fixed* $\varepsilon > 0$, a time-dependent, 2-state finite automaton that distinguishes $p = 1/2$ from $p = 1/2 + \varepsilon$ with high probability and that halts. Indeed, we simply need to look for a run of $1/\varepsilon$ consecutive heads, repeating this $2^{1/\varepsilon}$ times before halting. If such a run is found, then we guess $p = 1/2 + \varepsilon$; otherwise we guess $p = 1/2$.

(5) If we merely want to distinguish $p = 0$ from $p = \varepsilon$, then even simpler constructions suffice. With an ε-dependent finite automaton, at every time step we flip the coin with probability $1 - \varepsilon$; otherwise we halt and guess $p = 0$. If the coin ever lands heads, then we halt and output $p = \varepsilon$. Indeed, even an ε-*independent* finite automaton can distinguish $p = 0$ from $p = \varepsilon$ in the 1-sided model, by flipping the coin over and over, and accepting if the coin ever lands heads.

(6) It is not hard to show that even a *time-dependent, quantum* finite automaton cannot solve the distinguishing problem, even for $p = 0$ versus $p = \varepsilon$, provided that (i) the automaton has to halt when outputting its answer, and (ii) the same automaton has to work for every ε. The argument is simple: given a candidate automaton M, keep decreasing $\varepsilon > 0$ until M

halts, with high probability, before observing a single heads. This must be possible, since even if $p = 0$ (i.e., the coin *never* lands heads), M still needs to halt with high probability. Thus, we can simply wait for M to halt with high probability—say, after t coin flips—and then set $\varepsilon \ll 1/t$. Once we have done this, we have found a value of ε such that M cannot distinguish $p = 0$ from $p = \varepsilon$, since in both cases M sees only tails with high probability.

6   Open Problems

(1) Our advice-coin computational model can be generalized significantly, as follows. Let BQPSPACE/dice (m, k) be the class of languages decidable by a BQPSPACE machine that can sample from m distributions $\mathcal{D}_1, \ldots, \mathcal{D}_m$, each of which takes values in $\{1, \ldots, k\}$ (thus, these are "k-sided dice"). Note that BQPSPACE/coin = BQPSPACE/dice$(1, 2)$.

We conjecture that

$$\mathsf{BQPSPACE/dice}\,(1, \mathrm{poly}\,(n)) = \mathsf{BQPSPACE/dice}\,(\mathrm{poly}\,(n), 2) = \mathsf{PSPACE/poly}.$$

Furthermore, we are hopeful that the techniques of this paper can shed light on this and similar questions.[7]

(2) Not all combinations of model features in Section 5 are well-understood. In particular, can we distinguish a coin with bias $p = 1/2$ from a coin with bias $p = 1/2 + \varepsilon$ using a quantum finite automaton, not dependent on ε, that only needs to succeed in the limit?

(3) Given *any* degree-d rational function $a\,(p)$ such that $0 \le a\,(p) \le 1$ for all $0 \le p \le 1$, does there exist a d-state (or at least poly(d)-state) quantum finite automaton M such that $\Pr\,[M\,(\$_p)\ \text{accepts}] = a\,(p)$?

7   Acknowledgments

We thank Erik Demaine for suggesting the advice coins problem to us, and Piotr Indyk for pointing us to the Hellman-Cover Theorem.

References

[1] S. Aaronson. BQP and the polynomial hierarchy. In *Proc. ACM STOC*, 2010. arXiv:0910.4698.

[2] S. Aaronson and J. Watrous. Closed timelike curves make quantum and classical computing equivalent. *Proc. Roy. Soc. London*, (A465):631–647, 2009. arXiv:0808.2669.

[3] M. Ajtai. Σ_1^1-formulae on finite structures. *Ann. Pure Appl. Logic*, 24:1–48, 1983.

[4] K. Amano. Bounds on the size of small depth circuits for approximating majority. In S. Albers, A. Marchetti-Spaccamela, Y. Matias, S. E. Nikoletseas, and W. Thomas, editors, *ICALP (1)*, volume 5555 of *Lecture Notes in Computer Science*, pages 59–70. Springer, 2009.

[5] S. Basu, R. Pollack, and M. Roy. *Algorithms in Real Algebraic Geometry*. Springer, 2006.

---

[7]Note that the distinguishing problem for k-sided dice, for $k > 2$, is addressed by the more general form of the theorem of Hellman and Cover [13], while the distinguishing problem for read-once branching programs was explored by Brody and Verbin [9].

[6] A. Borodin. On relating time and space to size and depth. *SIAM J. Comput.*, 6(4):733–744, 1977.

[7] A. Borodin, S. Cook, and N. Pippenger. Parallel computation for well-endowed rings and space-bounded probabilistic machines. *Information and Control*, 58(1-3):113–136, 1983.

[8] M. Braverman, A. Rao, R. Raz, and A. Yehudayoff. Pseudorandom generators for regular branching programs. In *Proc. IEEE FOCS*, 2010.

[9] J. Brody and E. Verbin. The coin problem, and pseudorandomness for branching programs. In *Proc. IEEE FOCS*, 2010.

[10] T. M. Cover. Hypothesis testing with finite statistics. *Ann. Math. Stat.*, 40(3):828–835, 1969.

[11] E. F. Galvao and L. Hardy. Substituting a qubit for an arbitrarily large number of classical bits. *Phys. Rev. Lett.*, 90(087902), 2003. quant-ph/0110166.

[12] M. E. Hellman. *Learning with finite memory*. PhD thesis, Stanford University, Department of Electrical Engineering, 1969.

[13] M. E. Hellman and T. M. Cover. Learning with finite memory. *Ann. of Math. Stat.*, 41:765–782, 1970.

[14] C. A. Neff. Specified precision polynomial root isolation is in NC. *J. Comput. Sys. Sci.*, 48(3):429–463, 1994.

[15] C. A. Neff and J. H. Reif. An efficient algorithm for the complex roots problem. *J. Complexity*, 12(2):81–115, 1996.

[16] M. Nielsen and I. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 2000.

[17] R. O'Donnell and K. Wimmer. Approximation by DNF: examples and counterexamples. In *Proc. Intl. Colloquium on Automata, Languages, and Programming (ICALP)*, pages 195–206, 2007.

[18] V. Y. Pan. Optimal (up to polylog factors) sequential and parallel algorithms for approximating complex polynomial zeros. In *Proc. ACM STOC*, pages 741–750, 1995.

[19] B. Terhal and D. DiVincenzo. On the problem of equilibration and the computation of correlation functions on a quantum computer. *Phys. Rev. A*, 61:022301, 2000. quant-ph/9810063.

[20] E. Viola. On approximate majority and probabilistic time. In *Proc. IEEE Conference on Computational Complexity*, pages 155–168, 2007. Journal version to appear in Computational Complexity.

[21] E. Viola. Randomness buys depth for approximate counting. *Electronic Colloquium on Computational Complexity (ECCC)*, 17:175, 2010.

[22] J. Watrous. Space-bounded quantum complexity. *J. Comput. Sys. Sci.*, 59(2):281–326, 1999.