8-27-2012

# Quantum algorithms, symmetry, and Fourier analysis

Aaron Jafar Denney

Aaron Denney

*Candidate*

Physics and Astronomy

*Department*

This dissertation is approved, and it is acceptable in quality and form for publication:

*Approved by the Dissertation Committee:*

Cristopher Moore , Chairperson

Carlton Caves

Ivan Deutsch

Andrew Landahl

# Quantum Algorithms, Symmetry, and Fourier Analysis

by

**Aaron Denney**

B.S., California Institute of Technology, 2000

DISSERTATION

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Doctorate of Philosophy
Physics

The University of New Mexico

Albuquerque, New Mexico

July, 2012

# Acknowledgments

I would like to thank my advisor, Cris Moore, whose combination of patience and prodding has been vital. I would also like to thank the many members of the UNM quantum information groups whose support and insightful conversations have contributed to my graduate experience.

# Quantum Algorithms, Symmetry, and Fourier Analysis

by

**Aaron Denney**

B.S., California Institute of Technology, 2000

Ph.D., Physics, University of New Mexico, 2012

## Abstract

I describe the role of symmetry in two quantum algorithms, with a focus on how that symmetry is made manifest by the Fourier transform. The Fourier transform can be considered in a wider context than the familiar one of functions on $\mathbb{R}^n$ or $\mathbb{Z}/n\mathbb{Z}$; instead it can be defined for an arbitrary group where it is known as *representation theory*.

The first quantum algorithm solves an instance of the hidden subgroup problem—distinguishing conjugates of the Borel subgroup from each other in groups related to $\mathrm{PSL}(2;q)$. I use the symmetry of the subgroups under consideration to reduce the problem to a mild extension of a previously solved problem. This generalizes a result of Moore, Rockmore, Russel and Schulman[33] by switching to a more natural measurement that also applies to prime powers.

In contrast to the first algorithm, the second quantum algorithm is an attempt to use naturally continuous spaces. Quantum walks have proved to be a useful tool for designing quantum algorithms. The natural equivalent to continuous time quantum walks is evolution with the Schrödinger equation, under the kinetic energy Hamiltonian for a massive particle. I take advantage of quantum interference to find

the center of spherical shells in high dimensions. Any implementation would be likely to take place on a discrete grid, using the ability of a digital quantum computer to simulate the evolution of a quantum system.

In addition, I use ideas from the second algorithm on a different set of starting states, and find that quantum evolution can be used to sample from the evolute of a plane curve. The method of stationary phase is used to determine scaling exponents characterizing the precision and probability of success for this procedure.

# Contents

# List of Figures

# Glossary

| | |
|---|---|
| $\mathrm{AGL}(1;q)$ | The 1-dimensional affine group on $\mathbb{F}_q$ |
| $\mathrm{AGL}(d;2)$ | The $d$-dimension affine group on $\mathbb{F}_2$ |
| $\mathrm{AGL}(1;p)$ | The 1-dimensional affine group on $\mathbb{F}_p$ |
| $\mathbb{C}$ | The complex numbers |
| $\mathbb{F}_q$ | The finite field with $q$ elements |
| $G$ | A group |
| $\mathrm{GL}(2;q)$ | The 2-dimensional general linear group on $\mathbb{F}_q$ |
| $\mathrm{GL}(d;2)$ | The $d$-dimensional general linear group on $\mathbb{F}_2$ |
| $H$ | A subgroup, or a Hamiltonian |
| $m$ | An integer |
| $n$ | An integer |
| $p$ | A prime number |
| $\mathrm{P}\mathbb{C}$ | The complex plane completed with a point at infinity, also known as the Riemann sphere, or overly formally as the complex projective line |
| $\mathrm{P}\mathbb{F}_q$ | The projective finite field with $q+1$ elements; $\mathbb{F}_q \cup \infty$ |

| | |
|---|---|
| $\mathrm{PGL}(2;q)$ | The 2-dimensional projective linear group over the finite field with $q$ elements |
| $\mathrm{PSL}(2;q)$ | The 2-dimensional projective special linear group over the finite field with $q$ elements |
| $q$ | A prime power |
| $\mathbb{R}$ | The real numbers |
| $\mathbb{R}^n$ | Real space in $n$ dimensions |
| $\mathrm{SL}(2;q)$ | The 2-dimensional special linear group over the finite field with $q$ elements |
| $\mathbb{Z}$ | The integers |
| $\mathbb{Z}/n\mathbb{Z}$ or $\mathbb{Z}_n$ | The integers mod $n$ |

# Chapter 1

# Introduction

The idea of computations and of algorithms as an abstract class has caused an amazing revolution through the twentieth century by allowing mechanized data processing. The word "computer" has ceased to refer to an occupation and now refers to a machine.

When this abstraction was introduced, it missed a point. Computation is a physical process, and the physical universe in which we live is quantum mechanical in nature. Quantum mechanics was only just being developed at the same time as this notion—and quantum mechanics imposes a different set of restrictions on how it is possible to compute than classical mechanics does.

Because computation is a physical process, it is necessary to apply our knowledge of physics to capture what is relevant in constructing a relevant abstraction. From this necessity has arisen the fields of quantum information and computation. Even once this abstraction is constructed, further use and analysis can be greatly aided by applying tools of physics to these more abstract problems. In this thesis I attempt to demonstrate the utility of a key tool in physics— systematic exploitation of symmetries—to analysis of two quantum algorithms.

The first algorithm, presented and analyzed in chapter 3, has been published as [12] and is a restricted case of the hidden subgroup problem. The hidden subgroup

problem has been a standard framework for expressing the bulk of quantum algorithms in a unified framework. Given a function $f$ on a group $G$, the general problem is to identify the maximal subgroup $H$ that leaves the function unchanged. In order for this to be well posed, we require that there in fact is some group $H$ such that the function is constant and distinct on (left) cosets of $H$. That is:

$$f(c) = f(b) \text{ if and only if } c^{-1}b \in H.$$

In addition, to be able to use this function in a quantum algorithm, we must be able to apply it in a quantum coherent manner. This requirement is usually phrased in terms of an "oracle function", and part of the computational cost is actually evaluating such a function. Implicitly lurking behind many applications is running a classical algorithm to compute this on a general purpose quantum computer (such as fast exponentiation with Shor's factoring algorithm, or testing if an item is the desired one in Grover's search algorithm). In more abstract cases, the implementation of the oracle is ignored.

The groups under consideration here are $\mathrm{PSL}(2;q)$ and some other families with similar structure. I call it a restricted case, because in addition I require that the subgroup $H$ belong to a particular family of subgroups: conjugates of the subgroup of upper triangular matrices, $B$. All subgroups of a given group come in families of mutually conjugate subgroups. Telling which family a given subgroup is in is straightforward given the ability to efficiently Fourier transform. The hard part is telling conjugates apart. $\mathrm{PSL}(2;q)$ is an interesting family in that they are one of the relatively small number of families of simple groups. These are the atomic building blocks by which other, larger groups can be built from semi-direct products of smaller groups. Simple groups cannot be decomposed into smaller groups in this way. This suggests that it would be a fruitful family to have a working algorithm for. The central insight is that $\mathrm{PSL}(2;q)$ acts multiply transitively on elements of the projective field $\mathrm{P}\mathbb{F}_q$. $B$ and its conjugates stabilize elements of $\mathrm{P}\mathbb{F}_q$. This allows a classical reduction to a hidden subgroup problem in $B \cong \mathrm{AGL}(1;q)$. A moderate fix-up of a previously published measurement for solving $\mathrm{AGL}(1;p)$ [33] allows generalizing to $\mathrm{AGL}(1;q)$.

The second algorithm, considered in chapter 4 has not been previously published, and is an attempt to generalize the use of quantum walks on discrete systems to a continuous setting. A continuous-time quantum walk generalizes to evolution under the Schrödinger equation with a free-particle Hamiltonian. To attempt to do interesting computation, I essentially construct a problem to take advantage of interference effects. Constructive interference is realized when there are equal propagation distances to a given point. A set of points equidistant from a given point is, of course, a sphere, having a huge deal of symmetry. I show that for a reasonable family of states concentrated on the surface of a sphere, time evolution concentrates the resulting states near the center.

I further explore these ideas on less symmetric figures in chapter 5. For a plane curve, evolution leads to concentration not at the center, but at the evolute of the plane curve, the curve's set of centers of curvature. While the center of a sphere is a global property corresponding to rotational symmetry, the evolute of a curve is a more local property corresponding to neighborhoods of points, which weaken the interference effects. I hope to extend the analysis to higher dimensions and figures with symmetries weaker than a sphere, but stronger than arbitrary plane curves.

# Chapter 2

# Background

## 2.1 Classical and Quantum algorithms

There are many formal definitions of algorithms. They are all roughly equivalent in capturing the same class of processes, though some formulations are more fruitful for certain purposes than others. Rather than reiterating, we point the interested reader to discussion such as [31].

Algorithms take in some input data, process it in some manner, and then deliver an output. A classical algorithm, by definition, works with classical data, usually considered as bit strings. Though for many purposes classical approximations can be excellent, we do not live in a classical world. Computation is a physical process, and in certain circumstances it is more appropriate to consider quantum algorithms. This matters not just for abstract intellectual rigor, but because it appears that quantum computation offers the ability to fundamentally solve some processes faster than equivalent classical computers.

A quantum algorithm is one that works with quantum data, expressed as quantum states. Many common models restrict these to "quantum bit strings"—Hilbert spaces endowed with the structure $(\mathbb{C}^2)^{\otimes n}$ (though a physically irrelevant phase is "modded out" by identifying all *rays*—complex multiples of a given vector). There is no real

loss to doing so, as other quantum systems can be effectively simulated using such states. Because we ourselves are effectively classical, we can only consider as inputs those states that we can efficiently construct. This can be tightened slightly to requiring the input state be "0" without any loss (by moving initialization into the quantum computer). Our interaction with output states is restricted to collecting measurement statistics in an efficiently implementable basis. This usually means "local" measurements for qubits.

Many quantum algorithms depend crucially on symmetry. Our experience with physics has taught us that for problems with symmetry, it is useful to express our problem (e.g., states and operators) in bases adapted to that symmetry. The Fourier transform is one of the key ways of doing so.

## 2.2 Symmetry and the Fourier transform

A system is described as symmetric if some action leaves it the same. This notion of symmetry naturally leads to the notion of a *group*.

Two operations on a system can be composed. Performing one operation after the other is itself an operation. If both initial operations leave the system invariant, then the the composition must as well.

**Definition 1.** *A group is a set $G$ and a binary operation $\cdot$ such that:*

1. *The set $G$ is closed under the operation $\cdot$: for all elements $x$ and $y$ in $G$, $x \cdot y$ is also in $G$;*

2. *The operation is associative, so that for all elements $x$, $y$, and $z$: $x \cdot (y \cdot z) = (x \cdot y) \cdot z$;*

3. *There is an identity element $e$ such that for all $x \in G$, $e \cdot x = x \cdot e = x$;*

4. *Each element $x$ has an inverse $x^{-1}$ such that $x \cdot x^{-1} = x^{-1} \cdot x = e$.*

Two symmetry groups common in physics are the group of translations, and the group of rotations. The natural basis to work with translationally-invariant systems is the Fourier basis. The Fourier basis functions are of the form $\exp(-\mathrm{i}k \cdot x)$. Translations of these basis functions reduce to scalar multiplication (depending on both the basis function and the translation $\delta x$).

For most groups beyond this, we must generalize to "representation theory". Take rotations, for example: because they are non-abelian, there is no "perfect" basis. If rotations caused all basis functions to be multiplied by a scalar, they would commute. Instead, we choose basis functions that commute as much as possible. The natural basis functions that make rotations have as simple a form as possible are the spherical harmonics. Instead of all rotations acting trivially by scalar multiplications, only some do: those rotations about some picked $z$-axis. Other rotations can no longer act simply. They can intermix two different basis functions, but only ones that have the same $\ell$ value. Each $\ell$ value defines a separate *irreducible representation* that can largely be dealt with separately from the other irreducible representations, even though the entries within a given representation (i.e., the coefficients in a spherical harmonic transform with the same angular momentum $\ell$) cannot be.

The types of subsymmetry a function may have are well-captured by viewing in this basis. Continuing the rotation example, a function that only has non-zero spin-0 components is completely symmetric under rotations, while one that only has spin-1 components is not invariant under arbitrary rotations, but is invariant under rotations for some well-defined axis. Extracting the exact axis of symmetry requires knowing the values of the coefficients.

This picture is essentially the same for other non-abelian groups. There are a set of irreducible representations that are the fundamental building blocks for dealing with symmetries relative to that group. Each representation has a set of coefficients that intermix under action of the group, and capture a portion of the symmetries of states under the group and its subgroups.

# Chapter 3

# Symmetries of discrete spaces: a non-Abelian hidden subgroup problem

In this chapter, we reduce a case of the hidden subgroup problem (HSP) in $\mathrm{SL}(2;q)$, $\mathrm{PSL}(2;q)$, and $\mathrm{PGL}(2;q)$, three related families of finite groups of Lie type, to efficiently solvable HSPs in the affine group $\mathrm{AGL}(1;q)$. These groups act on projective space in an "almost" 3-transitive way, and we use this fact in each group to distinguish conjugates of its Borel (upper triangular) subgroup, which is also the stabilizer subgroup of an element of projective space. Our observation is mainly group-theoretic, and as such breaks little new ground in quantum algorithms. Nonetheless, these appear to be the first positive results on the HSP in finite simple groups such as $\mathrm{PSL}(2;q)$.

## 3.1 Introduction: hidden subgroup problems

One of the key tricks of Shor's factoring algorithm is the construction of a function on $\mathbb{Z}/nZ$ with periodicities that when recovered allow factoring $n$. These periodici-

ties are extracted from the global behavior of the function by the quantum Fourier transform.

This generalizes to one of the principal quantum algorithmic paradigms: handed a function $f$ on some group $G$ that is periodic in some way, our task is to characterize how it is periodic. This is a "black-box" function; we are not told anything about it beyond the restriction that it is suitably periodic. To actually compute values, we are given access to it via an "oracle" that we can query by giving it a quantum state representing a group element, and have it return a state representing its answer. Not only can we give it a state representing a group elements, but states representing coherent superpositions of group elements. The answers must then be returned as coherent superpositions. The symmetry of the function will also be a group—some "hidden" subgroup $H$ of $G$ such that $f$ is precisely invariant under translation by $H$ or, equivalently, $f$ is constant on the cosets of $H$ and takes distinct values on distinct cosets. The *hidden subgroup problem* is the problem of determining the subgroup $H$ (or, more generally, a short description of it, such as a generating set) from such a function.

The standard approach[8] is to use the oracle function $f$ to create *coset states*

$$\rho_H = \frac{1}{|G|} \sum_{c \in G} |cH\rangle \langle cH|$$

where

$$|cH\rangle = \frac{1}{\sqrt{|H|}} \sum_{h \in H} |ch\rangle \ .$$

Different subgroups yield different coset states, which must then be distinguished by some series of quantum measurements.

For *abelian* subgroups, sampling these states in the Fourier basis of the group $G$ is sufficient to completely determine a hidden subgroup in an efficient manner. For *nonabelian* subgroups, the Fourier basis takes the form $\{|\rho, i, j\rangle\}$ where $\rho$ is the name of an irreducible representation and $i$ and $j$ index a row and column in a chosen basis. Although a number of interesting results have been obtained on the

nonabelian HSP, the groups for which efficient solutions are known remain woefully few. Friedl, Ivanyos, Magniez, Santha, and Sen solve a problem they call the Hidden Translation Problem, and thus generalize this further to what they call "smoothly solvable" groups: these are solvable groups whose derived series is of constant length and whose abelian factors are each the direct product of an abelian group of bounded exponent and one of polynomial size [17]. Moore, Rockmore, Russell, and Schulman give an efficient algorithm for the affine groups $\mathrm{AGL}(1;p) = \mathbb{Z}_p \rtimes \mathbb{Z}_p^*$, and more generally $\mathbb{Z}_p \rtimes \mathbb{Z}_q$ where $q = (p-1)/\mathrm{polylog}(p)$. Bacon, Childs, and van Dam derive algorithms for the Heisenberg group and other "nearly abelian" groups of the form $A \rtimes Z_p$, where $A$ is abelian, by showing that the "Pretty Good Measurement" is the optimal measurement for distinguishing the corresponding coset states [2]. Recently, Ivanyos, Sanselme, and Santha [23, 24] give an efficient algorithm for the HSP in nilpotent groups of class 2.

However, for groups of the greatest algorithmic interest, such as the symmetric group $S_n$ for which solving the HSP would solve Graph Isomorphism, the hidden subgroup problem appears to be quite hard. Moore, Russell, and Schulman showed that the standard approach of Fourier sampling individual coset states fails [34]. Hallgren et al. showed under very general assumptions that highly-entangled measurements over many coset states are necessary in any sufficiently nonabelian group [20]. For $S_n$ in particular, Moore, Russell and Śniady showed that the main proposal for an algorithm of this kind, a sieve approach due to Kuperberg [26], cannot succeed [35].

It is tempting to think that the difficulty of the HSP on the symmetric group is partly due to the appearance of the alternating group $A_n$ as a subgroup. For $n \geq 5$, $A_n$ forms one of the families of nonabelian finite simple groups. All known algorithmic techniques for the HSP work by breaking the group down into abelian pieces, as a semidirect product or through its derived series. Since simple groups cannot be broken down this way, it seems that any positive results on the HSP for simple groups is potentially valuable.

We offer a small advance in this direction. We show how to efficiently solve a restricted case of HSP for the family of finite simple groups $\mathrm{PSL}(2;q)$, and for two

related finite groups of Lie type. No new quantum techniques are introduced; instead, we point out a group-theoretic reduction to a mild extension of a previously solved case of the HSP. Unfortunately, this reduction only applies to one set of subgroups, and there is no obvious generalization that covers the other subgroups. On the other hand, we show that a similar reduction works in any group which acts on some set in a sufficiently transitive way, though this is unhelpful in many obvious cases.

## 3.2 Reduction

We start with a trivial observation: suppose we have a restricted case of the hidden subgroup problem where we need to distinguish among a family of subgroups $H_1, ..., H_t \subset G$. If there is a subgroup $F$ whose intersections $K_i = H_i \cap F$ are distinct, then we can reduce the original hidden subgroup problem to the corresponding one on $F$, consisting of distinguishing among the $K_i$, by restricting the oracle to $F$, rather than the original domain $G$.

The subgroups in question will be the stabilizers of one or more elements under a suitably transitive group action. Recall the following definitions:

**Definition 2.** *A* group action *of a group $G$ on a set $\Omega$ is a homomorphism $\phi$ from $G$ to the group of permutations on $\Omega$. In other words,*

$$\phi(g_1 g_2)(x) = \phi(g_1)(\phi(g_2)(x)) \,.$$

*When the group action is understood, we will often write just $g_1(x)$ for $\phi(g_1)(x)$.*

**Definition 3.** *A* transitive *group action on a set $\Omega$ is one such that for any $\alpha, \beta \in \Omega$ there is at least one $g \in G$ such that $g(\alpha) = \beta$. A $k$-*transitive *group action is one such that any $k$-tuple of distinct elements $(\alpha_1, \ldots, \alpha_k)$ can be mapped to any $k$-tuple of distinct elements $(\beta_1, \ldots, \beta_k)$. That is, given that $\alpha_i = \alpha_j$ and $\beta_i = \beta_j$ only when $i = j$, there is at least one $g$ such that $g(\alpha_i) = \beta_i$ for all $i = 1, \ldots, k$ A group is called $k$-transitive if it has a $k$-transitive group action on some set.*

**Definition 4.** *Given an element $\alpha \in \Omega$, the* stabilizer *of $\alpha$ with respect to a given action by a group $G$ is the subgroup $G_\alpha = \{g \in G \mid g(\alpha) = \alpha\}$. Given a subset $S \subseteq \Omega$, the pointwise stabilizer is*

$$G_S = \{g \in G \mid \forall \alpha \in S : g(\alpha) = \alpha\} = \bigcap_{\alpha \in S} G_\alpha \,.$$

*When $S$ is small we will abuse notation by writing, for instance, $G_\alpha$ or $G_{\alpha,\beta}$.*

Let's consider the case of the HSP where we wish to distinguish the one-point stabilizers $G_\alpha$ from each other. If $G$ is transitive, these are conjugates of each other, since $G_\beta = gG_\alpha g^{-1}$ for any $g$ such that $g(\alpha) = \beta$. Conversely, $gG_\alpha g^{-1} = G_{g(\alpha)}$, so any conjugate of a stabilizer is a stabilizer. Similarly, for each $\alpha$, the two-point stabilizers $G_{\alpha,\beta}$ labeled by $\beta$ are conjugate subgroups in $G_\alpha$.

Now suppose we restrict our queries to the oracle to $G_\alpha$. We then get a coset state corresponding to $G_\alpha \cap G_\beta = G_{\alpha,\beta}$:

$$\rho_{G_{\alpha\beta}} = \frac{1}{|G_\alpha|} \sum_{c \in G_\alpha} |cG_{\alpha,\beta}\rangle \langle cG_{\alpha,\beta}| \,.$$

This reduces the problem of distinguishing the one-point stabilizers $G_\beta$, as subgroups of $G$, to that of distinguishing the two-point stabilizers $G_{\alpha,\beta}$ as subgroups of $G_\alpha$—a potentially easier problem. Note that we can test for the possibility that $\alpha = \beta$ with a polynomial number of classical queries, since we just need to check that $f(1) = f(g)$ for a set of $O(\log |G|)$ generators of $G_\alpha$.

Of course, this whole procedure is only useful if $G_{\alpha,\beta}$ are distinct when $G_\beta$ are distinct, or if there are only a (polynomially) small number of one-point stabilizers corresponding to each two-point stabilizer. Below we give sufficient conditions for this to be true, and use this reduction to give an explicit algorithm for distinguishing conjugates of the Borel subgroups in some finite groups of Lie type, including the finite simple groups PSL(2; q). Using the transitivity of the group action we can bound the size of these stabilizers relative to each other and to the original group, and hence show that they are distinct.

**Lemma 1.** *Suppose $G$ has a $k$-transitive group action on a set $\Omega$ where $|\Omega| = s$. Then for any $j \leq k$, if $S \subseteq \Omega$ and $|S| = j$, we have*

$$\frac{|G|}{|G_S|} = \frac{s!}{(s-j)!}\,.$$

*In particular,*

$$|G_\alpha| = \frac{|G|}{s}\,,\ |G_{\alpha,\beta}| = \frac{|G|}{s(s-1)}\,,\ |G_{\alpha,\beta,\gamma}| = \frac{|G|}{s(s-1)(s-2)}\,.$$

*Proof.* The index of $G_S$ in $G$ is the number of cosets. There is one coset for each $j$-tuple to which we can map $S$, and since $G$ is $j$-transitive this includes all $s!/(s-j)!$ ordered $j$-tuples. □

For groups that are at least 3-transitive, the following then holds: the intersection of two subgroups that are single-point stabilizers of $\Omega$ has size $1/(s-1)$ of both of the subgroups. The intersection with a third stabilizer subgroup is $1/(s-2)$ this size again. In particular, this means that when subgroups $G_\beta$ and $G_\gamma$ are distinct, then their intersections $G_\alpha \cap G_\beta = G_{\alpha,\beta}$ and $G_\alpha \cap G_\gamma = G_{\alpha,\gamma}$ are distinct, because their intersection $G_{\alpha,\beta} \cap G_{\alpha,\gamma} = G_{\alpha,\beta,\gamma}$ is smaller than either.

In fact, we don't need full 3-transitivity for this argument to hold. The crucial fact we used was that the number of cosets of $G_{\alpha,\beta,\gamma}$ was greater than $G_{\alpha,\beta}$ or $G_{\alpha,\gamma}$. Consider the following definition:

**Definition 5.** *A group is* almost $k$-transitive *if there is a constant $b$ such that $G$ has an action on a set $\Omega$ which is $(k-1)$-transitive, and such that we can map any $k$-tuple of distinct elements $(\alpha_1, \ldots, \alpha_k)$ to at least a fraction $b$ of all ordered $k$-tuples $(\beta_1, \ldots, \beta_k)$ of distinct elements.*

Strictly speaking, there is a different notion of "almost" for different values of $b$. Obviously, for any group there is some value of $b$ low enough that this definition applies. However, by fixing $b$ and considering a family of groups we still have a useful concept.

As an example, a group action is *k-homogeneous* if any set of points of size $k$ can be mapped (setwise) to any other set of the same size. Since this means that any ordered $k$-tuple can be mapped to at least $1/k!$ of the ordered $k$-tuples, and since all $k$-homogeneous group actions are $(k-1)$-transitive [13], a group with such an action is almost $k$-transitive with $b = 1/k!$ (in fact, with $b = 1/k$).

Applying the above argument to almost 3-transitive groups shows that the stabilizer of 3 distinct elements is smaller than the stabilizer of 2 distinct elements by a factor of $(s-2)b$. So long as $b \geq 1/(s-2)$, two-point stabilizers of distinct elements will be distinct. In the group families we cover, $b = 1/2$, and $s$ grows.

## 3.3 Families of transitive groups

Which families of groups and subgroups have the kind of transitivity that let us take advantage of this idea?

Unfortunately, not many do. We can categorize based on *faithful* group actions, i.e., those that do not map any group element other than the identity to the trivial action. Any non-faithful group action corresponds to a faithful action of a quotient of the group. Even the requirement of 2-transitivity in faithful group actions restricts the choices to a few sporadic groups, or one of eight infinite families [13]: The symmetric group $S_n$, the alternating group $A_n$, and six different families of groups of Lie type.

Obviously the symmetric group $S_n$ is $n$-transitive, and the alternating group $A_n$ is almost $n$-transitive. However, the size $n$ of the set these groups act on is only polynomially large (i.e., polylogarithmic in the size of the groups) so we can distinguish the one-point stabilizers with a polynomial number of classical queries.

The other infinite families are finite groups of Lie type which are defined in terms of matrices over finite fields $\mathbb{F}_q$ subject to some conditions. These groups have natural actions by matrix multiplication on column vectors, or on equivalence classes

of column vectors. The actions of most of these groups are rather complicated to describe; for more details, see [13, §7.7]. Of these, two are 3-transitive: $\mathrm{PSL}(2;q)$, and $\mathrm{AGL}(d;2)$.

There are also a number of sporadic finite groups that are up to 5-transitive, such as the Matthieu groups $M_{11}$, $M_{12}$, $M_{22}$, $M_{23}$, $M_{24}$, built on finite geometries. However, an interesting fact is that if a group action has a threshold of transitivity, then it contains all permutations, or at least all even ones: for $k > 5$, all finite groups with a $k$-transitive action on a set of size $n$ must contain $A_n$ [13].

## 3.4 $\mathbf{PSL}(2;q)$ and some relatives

The most interesting family of simple groups with a faithful almost 3-transitive group action is $\mathrm{PSL}(2;q)$. To discuss it, consider instead $\mathrm{GL}(2;q)$, the group of invertible $2 \times 2$ matrices with entries in the finite field $\mathbb{F}_q$, where $q = p^n$ is the power of some prime $p$. Its elements are of the form

$$\begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix}$$

where $\alpha, \beta, \gamma, \delta \in \mathbb{F}_q$, and $\alpha\delta - \beta\gamma \neq 0$. We will assume that $q$ is odd; some details change when it is a power of 2, but the basic results still hold.

A little thought reveals that $|\mathrm{GL}(2;q)| = (q^2-1)(q^2-q) = (q+1)q(q-1)^2$. The subgroup $\mathrm{SL}(2;q)$ consists of the matrices with determinant 1, so $|\mathrm{SL}(2;q)| = (q+1)q(q-1)$. If we take the quotient of these groups by the normal subgroup consisting of the scalar matrices, we obtain $\mathrm{PGL}(2;q)$ and $\mathrm{PSL}(2;q)$ respectively. For $\mathrm{SL}(2;q)$ the only scalar matrices are $\pm\mathbb{1}$, so $|\mathrm{PSL}(2;q)| = (q+1)q(q-1)/2$.

$\mathrm{GL}(2;q)$ and $\mathrm{SL}(2;q)$ act naturally on nonzero 2-dimensional vectors. However, for $\mathrm{PGL}(2;q)$ and $\mathrm{PSL}(2;q)$, we must identify vectors which are scalar multiples. This identification turns $\mathbb{F}_q^2 - \{0,0\}$ into the projective line $\mathrm{P}\mathbb{F}_q$. Each element of $\mathrm{P}\mathbb{F}_q$ corresponds to a "slope" of a vector: the vector $\begin{pmatrix} x \\ y \end{pmatrix}$ has slope $x/y$, i.e., $xy^{-1}$

if $y \neq 0$ and $\infty$ if $y = 0$. Thus we can think of $\mathrm{P}\mathbb{F}_q$ as $\mathbb{F}_q \cup \{\infty\}$, and it has $q + 1$ elements.

The action of $\mathrm{PGL}(2; q)$ and $\mathrm{PSL}(2; q)$ on $\mathrm{P}\mathbb{F}_q$ is given by

$$\begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \alpha x + \beta y \\ \gamma x + \delta y \end{pmatrix} .$$

This fractional linear transformation is analogous to the Möbius transformation defined by $\mathrm{PGL}_2(\mathbb{C})$:

$$\begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \frac{\alpha x + \beta y}{\gamma x + \delta y} ,$$

which can map any 3 points in the complex projective line $\mathrm{P}\mathbb{C}$ (i.e., the complex plane augmented by the point at infinity, or the Riemann sphere) to any other 3 points. When we replace $\mathbb{C}$ with the finite field $\mathbb{F}_q$, the action of $\mathrm{PGL}(2; q)$ remains 3-transitive. The action of $\mathrm{PSL}(2; q)$ is 2-transitive, but cannot be 3-transitive, since there are half as many elements as there are 3-tuples. However, $\mathrm{PSL}(2; q)$ is almost 3-transitive in the sense defined above with $b = 1/2$, since $1/2$ of all 3-tuples can be reached. $\mathrm{SL}(2; q)$ is also almost 3-transitive: from a given tuple, it reaches the same set of tuples as $\mathrm{PSL}(2; q)$, with each tuple being hit twice. As a result, this action is obviously not faithful, for the kernel is $\pm\mathbb{1}$.

Let $G = \mathrm{PGL}(2; q)$, and consider the one-point stabilizer subgroups of its action on $\mathrm{P}\mathbb{F}_q$. A natural one is the Borel subgroup $B$ of upper-triangular matrices. Such matrices preserve the set of vectors of the form $\begin{pmatrix} x \\ 0 \end{pmatrix}$, so we can write $B = G_\infty$. There are $q + 1$ conjugates of $B$, including itself, one for each element of $\mathrm{P}\mathbb{F}_q$. For instance, if we conjugate by the Weyl element $w = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix}$, we get $wBw^{-1} = G_0$, the subgroup of lower-triangular matrices, which preserves the set of vectors of the form $\begin{pmatrix} 0 \\ y \end{pmatrix}$.

## 3.5 An efficient algorithm for distinguishing the conjugates of the Borel subgroup

Now consider the case of the HSP on these groups where the hidden subgroup is one of $B$'s conjugates, or equivalently, one of the one-point stabilizers $G_s$. As discussed above, we solve this by restricting the oracle to $B$, and distinguishing the two-point stabilizer subgroups $B \cap G_s = G_{s,\infty}$ as subgroups of $B$. To do this, we need to describe the structure of $B$ explicitly. For all three families of matrix groups we discuss, namely $\mathrm{SL}(2;q)$, $\mathrm{PSL}(2;q)$, and $\mathrm{PGL}(2;q)$, $B$ is closely related to the affine group.

In $\mathrm{PGL}(2;q)$ a generic representative of $B$ can be written $\begin{pmatrix} \alpha & \beta \\ 0 & 1 \end{pmatrix}$, $\alpha \neq 0$, so $|B| = q(q-1)$. This is exactly the affine group $\mathrm{AGL}(1;q) \cong \mathbb{F}_q \rtimes \mathbb{F}_q^*$. To see this, recall that $\mathrm{AGL}(1;q)$ consists of the set of affine functions on $\mathbb{F}_q$ of the form $x \mapsto \alpha x + \beta$ under composition. Now consider $B$'s action on $\mathrm{P}\mathbb{F}_q - \{\infty\}$, which we (re)identify with $\mathbb{F}_q$. For $\mathrm{PGL}(2;q)$, we have

$$\begin{pmatrix} \alpha & \beta \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ 1 \end{pmatrix} = \begin{pmatrix} \alpha x + \beta \\ 1 \end{pmatrix} .$$

Obviously these elements compose as $\mathrm{AGL}(1;q)$, so $B \cong \mathrm{AGL}(1;q)$.

The cases of $\mathrm{SL}(2;q)$ and $\mathrm{PSL}(2;q)$ are more complicated. The unit determinant requirement limits $B$ to elements of the form $\begin{pmatrix} \alpha & \beta \\ 0 & \alpha^{-1} \end{pmatrix}$. Thus $|B| = q(q-1)$ again in $\mathrm{SL}(2;q)$. For $\mathrm{PSL}(2;q)$ we identify $\alpha$ with $-\alpha$, so $|B| = q(q-1)/2$.

For $\mathrm{SL}(2;q)$, we can enumerate the elements as:

$$\begin{pmatrix} \alpha & \alpha^{-1}\beta \\ 0 & \alpha^{-1} \end{pmatrix} .$$

Composing two such elements gives us:

$$\begin{pmatrix} \alpha & \alpha^{-1}\beta \\ 0 & \alpha^{-1} \end{pmatrix} \begin{pmatrix} \gamma & \gamma^{-1}\delta \\ 0 & \gamma^{-1} \end{pmatrix} = \begin{pmatrix} \alpha\gamma & \alpha^{-1}\gamma^{-1}\beta + \gamma^{-1}\alpha\delta \\ 0 & \alpha^{-1}\gamma^{-1} \end{pmatrix} = \begin{pmatrix} \alpha\gamma & (\alpha^{-1}\gamma^{-1})(\beta + \alpha^2\delta) \\ 0 & \alpha^{-1}\gamma^{-1} \end{pmatrix} .$$

Here, we still have a semidirect product of the groups $\mathbb{F}_q$ and $\mathbb{F}_q^*$. Unlike the affine group, where the multiplicative group acts directly as an automorphism on the additive group by multiplication, it instead acts "doubly" by multiplying twice, analogous to the "$q$-hedral" groups in [33] (with $q = p/2$, in their notation). Finally, $\mathrm{PSL}(2; q)$ merely forgets the difference between $\pm\alpha$. This quotient group of $\mathrm{SL}(2; q)$ can also be seen as a subgroup of the affine group that can only multiply by the square elements.

In all three cases the HSP on $B$ can be solved efficiently using small generalizations of the algorithms of [33]. We need to generalize slightly as [33] deals only with the case of $\mathbb{Z}_n \ltimes \mathbb{F}_p$ with $p$ prime—not a prime power $q = p^n$, as here. The basic methods remain effective, though we construct and analyze a slightly different final measurement. The number and size of the representations remains the same (with $q$ replacing $p$), and the methods for constructing Gelf'and-Tsetlin adapted bases are similar. As this has not been published in the literature, we describe the details more fully in the next section, though only what is necessary for our purposes.

## 3.6 Generalizing the affine group to prime powers

Although there can be more types of subgroups than the ones covered in [33], we are only concerned about one particular type whose analog was covered there: $H = (a, 0)$ and its conjugates $H^b = (1, b)H(1, -b)$, stabilizing the finite field element $b$. The representation theory is analogous, with $q - 1$ one-dimensional representations (characters) depending only on $a$. As in the prime case, we have $q$ conjugacy classes: the identity, all pure translations, and each multiplication by a different $a$, combined with all translations. This leaves us with one $(q - 1)$ dimensional representation, $\rho$.

In the prime case we had:

$$\rho((a, b))_{j,k} = \begin{cases} \omega_p^{bj} & k = aj \\ 0 & \text{otherwise} \end{cases} \qquad (j, k \in \mathbb{F}_q, \neq 0)\,.$$

where $\omega_p = \exp(2\pi i/p)$. The roots of unity are the non-trivial additive characters of

$\mathbb{F}_p$, indexed by $j$, evaluated at $b$. We can extend this to the prime power case simply by replacing $bj$ with

$$b \cdot j = \operatorname{Tr} bj = \operatorname{Tr}_{\mathbb{F}_{p^n}/\mathbb{F}_p} bj = \sum_{m=1}^{n} (bj)^{p^m} ,$$

which as $b$ varies, exactly covers the full set of non-trivial linear operators from $\mathbb{F}_{p^n}$ to $\mathbb{F}_p$, and $\omega_p^{b \cdot j}$ exactly covers the set of additive characters. Performing weak Fourier sampling (measuring only the representation name) on the coset state yields $\rho$ with probability $P(\rho) = 1 - 1/q$. Conditioned on that outcome, we get the following projection operator:

$$\pi_{H^b}(\rho)_{j,k} = \frac{1}{q-1} \omega_p^{b \cdot (j-k)} .$$

As in [33], we then perform a Fourier transform on the rows, and ignore the columns. There they performed the Fourier transform over $\mathbb{Z}_{p-1}$, as there were $p-1$ rows. However, the structure for general $q$ is not $\mathbb{Z}_q^* \equiv \mathbb{Z}_{q-1}$, but $\mathbb{F}_q^*$. The interaction we want to capture is the additive one, not the multiplicative one. We can still perform the abelian transform over the additive group $\mathbb{F}_q \equiv \mathbb{Z}_p^n$—the zero component we lack is, of course, zero. The probability of observing a frequency $\ell \in \mathbb{Z}_p^n$ is then:

$$\begin{aligned} P(\ell) &= \left| \frac{1}{\sqrt{q(q-1)}} \sum_{j \neq 0} \omega_p^{b \cdot j} \omega_p^{-j \cdot \ell} \right|^2 \\ &= \frac{1}{q(q-1)} \left| -1 + \sum_{j} \omega_p^{b \cdot j} \omega_p^{-j \cdot \ell} \right|^2 \\ &= \frac{1}{q(q-1)} \left| -1 + q \delta_{\ell b} \right|^2 \\ &= \begin{cases} \frac{1}{q(q-1)} & \ell \neq b \\ 1 - \frac{1}{q} & \ell = b \end{cases} . \end{aligned}$$

For the case of $B$ in $\mathrm{PSL}(2; q)$, we can analyze the equivalent measurements via the embedding in the full affine group, just as in the prime case. Let $a$ be a generator of the "even" multiplicative subgroup of $\mathbb{F}_q^*$, consisting of elements that are squares.

$H_a^b$ is then elements of the form $(a^t, (1-a^t)b)$ stabilizing $b$. For these subgroups, the trivial representation, a "sign" representation, and the large representation occur with non-zero probability. The first two have vanishingly small probability, $O(1/q)$.

In the following we use the notation $G(m,a) = \sum_{x\in\mathbb{F}_q^*} m(x)a(x)$ for the Gauss sum of a multiplicative and an additive character, where $\chi_k(j) = \omega_p^{k\cdot j}$ is the additive character of $\mathbb{F}_q$ with frequency $k \in \mathbb{Z}_p^n$. We follow the common convention that non-trivial multiplicative characters vanish at 0. We use the quadratic character $\eta$ of $\mathbb{F}_q^*$, which is 1 for squares, and $-1$ for non-squares, to select rows and columns which differ by values in the "even" subgroup mentioned above.

Weak measurement gives us the representation $\rho$ with overwhelming probability. Conditioning on this event, we get the mixed state

$$\begin{aligned}\rho(H_a^b)_{j,k} &= \frac{\sqrt{2}}{q-1}\sum_{t=1}^{q-1/2} \omega_p^{(1-a^tb)\cdot j}\delta_{k,a^tj} \\ &= \frac{\sqrt{2}}{q-1}\sum_{t=1}^{q-1/2} \omega_p^{b\cdot(j-k)}\delta_{k,a^tj} \\ &= \frac{\sqrt{2}}{q-1}\,\omega_p^{b\cdot(j-k)}(1+\eta(jk))/2\,.\end{aligned}$$

Measuring the column $k$ gives us, up to a phase, $\rho(b)_j = \sqrt{\frac{2}{q-1}}\,\omega_p^{b\cdot j}(1\pm\eta(j))/2$.

We again include the zero component, with zero weight, and perform the abelian Fourier transform over the additive group $\mathbb{F}_q \equiv \mathbb{Z}_p^n$. The probability of measuring

frequency $\ell$ is

$$\begin{aligned}
P(\ell) &= \frac{1}{q}\left|\sum_j \omega_p^{j\cdot\ell}\rho(b)_j\right|^2 \\
&= \frac{2}{q(q-1)}\left|\sum_{j\neq 0}\omega_p^{(b-\ell)\cdot j}(1\pm\eta(j))/2\right|^2 \\
&= \frac{2}{q(q-1)}\left|\sum_{j\neq 0}\chi_{b-\ell}(j)\pm\sum_{j\neq 0}\chi_{b-\ell}(j)\eta(j)\right|^2 \\
&= \frac{1}{2q(q-1)}\left|G(1,\chi_{b-\ell})\pm G(\eta,\chi_{b-\ell})\right|^2 \\
&= \frac{1}{2q(q-1)}\left|q\delta_{b,\ell}-1\pm\eta(b-\ell)G(\eta,\chi_1)\right|^2 \\
&= \frac{1}{2q(q-1)}\left|q\delta_{b,\ell}-1\pm\eta(b-\ell)i^d q^{1/2}\right|^2
\end{aligned}$$

where $d$ is odd for odd $n$ if $p^n \equiv 3 \pmod 4$, and $d$ is even otherwise.

For $\ell = b$ we have $P(\ell) = (q-1)^2/2q(q-1) = (q-1)/2q$. For $\ell \neq b$ we have $P(\ell) = (q+1)/4q(q-1)$ if $d$ is odd. If $\ell \neq b$ and $d$ is even, we have $P(\ell) = (q \pm 2q^{1/2}+1)/4q(q-1)$. In any case, the probability of observing $b$ is

$$P(b) = \frac{q-1}{2q} = \frac{1}{2} - O(1/q)\,,$$

so repeating this measurement will allow us to identify $\ell = b$ with any desired probability. As $\mathrm{SL}(2;q)$ is a small extension of $\mathrm{PSL}(2;q)$, we can handle it similarly, by Theorem 8 in [33].

## 3.7 $\mathbf{AGL}(d;2)$ and its stabilizer subgroups

An interesting question is whether it is useful to apply this approach to the other family of 3-transitive groups. This is the $d$-dimensional affine group $\mathrm{AGL}(d;2)$, consisting of functions on $\mathbb{F}_2^d$ of the form $Av+B$, where $A \in \mathrm{GL}_d(\mathbb{F}_2)$ and $B \in \mathbb{F}_2^d$. It can be expressed as a block matrix of the form $\begin{pmatrix} A & B \\ 0 & 1 \end{pmatrix}$. It is the semidirect product $\mathrm{GL}(d;2) \ltimes \mathbb{F}_2^d$, and hence obviously not simple. That it is triply transitive

can be seen by realizing that the affine geometry it acts on has no three points that are collinear.

The stabilizer subgroups are $2^d$ conjugate subgroups of the original $\mathrm{GL}(d;2)$. Obviously this stabilizes the point 0, and is the largest subgroup that will, as $\mathrm{GL}(d;2)$ has two orbits: the zero vector, and all others. A general point $P$ is stabilized by translating it to 0 with the element $(A,B) = (1,P)$, applying any element of $\mathrm{GL}(d;2)$, and then translating back. To apply our method we need to look at the intersections.

Consider the point $\vec{1} = (0,...,0,1)^T$. Splitting $A$ into two diagonal blocks of size $(d-1)\times(d-1)$ and $1\times 1$ and two off-diagonal blocks of size $(d-1)\times 1$ and $1\times(d-1)$ allows us to see that $\vec{1}$ is stabilized by a (transposed) copy of $\mathrm{A}GL(d-1;2)$ living in $\mathrm{GL}(d;2)$. The last column must be $\vec{1} = (0,...,0,1)^T$ to preserve $\vec{1}$. The large $(d-1)\times(d-1)$ block must be in $\mathrm{GL}(d;2)$ to keep the the entire transformation invertible, and anything in $\mathrm{GL}(d;2)$ will preserve the first $d-1$ 0 bits of $\vec{1}$. The rest of the last row can be arbitrary, resulting in a subgroup isomorphic to $\mathrm{A}GL(d-1;2)$.

As a result, distinguishing the stabilizers of points reduces to distinguishing conjugates of a smaller transposed copy of the affine group in the general linear group. This last reduction does not immediately yield an efficient new quantum algorithm.

## 3.8 Conclusion

It is interesting to note that although we can Fourier sample over $\mathrm{AGL}(d;2)$ efficiently [32], we don't know how to do so in the projective groups. The fastest known classical Fourier transform for $\mathrm{SL}(2;q)$ or $\mathrm{PSL}(2;q)$ takes $\Theta(q^4 \log q)$ time [27], and the natural quantum adaptation of this takes $\Theta(q \log q)$ time [32]. If $q$ is exponentially large, this is polynomial, rather than polylogarithmic, in the size of the group. In the absence of new techniques for the FFT or QFT, this suggests that we need to somehow reduce the HSP in $\mathrm{PSL}(2;q)$ to that in some smaller, simpler group—which was the original motivation for our work.

We conclude by asking whether our analysis of $\mathrm{AGL}(d;2)$ can be extended to give an efficient algorithm distinguishing its stabilizer subgroups, or whether any of the other 2-transitive groups have usable "almost" 3-transitive actions.

## 3.9 Acknowledgment

This work was supported by the NSF under grant CCF-0829931, and by the DTO under contract W911NF-04-R-0009.

# Chapter 4

# Symmetries of continuous spaces: finding the center of a sphere

Quantum walks have been the basis for many interesting quantum algorithms, from evaluating a NAND tree with $n$ leaves in time $O(n^{1/2})$ to recovering hidden nonlinear structures in finite fields. Taking the continuous limit of a continuous-time quantum walk on a lattice yields the time-dependent Schrödinger equation for a single particle. Rather than solving algebraic problems, we use this approach to recover geometric information about an initial state. In $\mathbb{R}^n$, given a state concentrated on a hyperspherical shell, or access to a spherically-symmetric function, we can efficiently locate its center of symmetry using a small number number of samples. This compares favorably to the $O(n)$ samples required classically, and gives a simpler alternative to an algorithm of Liu. The phenomena we exploit are analogous to classical phenomena in diffraction and geometric optics, such as Arago's spot and the brightening of caustics and their cusps.

## 4.1 Introduction

There are many perspectives for viewing quantum algorithms, but fundamentally they are all based on delicately constructing interference such that some property of

an initial state, Hamiltonian, or unitary operator determines which of a set of states will be observed with high probability.

In Grover's algorithm [19], the initial state is a uniform distribution over all elements, and alternating "query" and "diffusion" operators iteratively concentrate amplitude on a marked element. Farhi and Gutmann describe a nice variant [16] that works in continuous time, where the marked element is encoded as the sole eigenvector of a Hamiltonian with nonzero eigenvalue. By turning on an additional interaction that does not depend on which element is marked, time evolution under the new Hamiltonian recovers the marked element.

A similar framework is used by Farhi, Goldstone, and Gutmann [15], in their Hamiltonian algorithm for evaluating NAND trees. As in [16], the input to the algorithm consists of a Hamiltonian, namely the adjacency matrix of a binary tree, where the presence or absence of an additional edge at each leaf represents a truth value. This Hamiltonian is then modified by the addition of a line of ancillary nodes connected at the origin to the root of the tree. Time evolution under this Hamiltonian implements a continuous-time *quantum walk* on this graph. They construct a suitable wave packet that will propagate on the line, and either reflect off the tree or pass through it depending on the truth value of the tree. Measuring which side of the line the wave packet ends up on then gives the truth value.

Childs, Schulman, and Vazirani consider various "hidden nonlinear structure" problems [7]. In their "hidden flat of centers" problem, one key construction is their use of a continuous-time quantum walk on the Winnie Li graph of a finite field. Again, the Hamiltonian is the adjacency matrix of the graph. Starting with a superposition over a "quasi-spherical" shell, this walk shifts probability closer to the center, letting them reconstruct the hidden flat.

Inspired in particular by these last two papers, we explore how interference could be used in the case of continuous spaces. We know from [7] and [15] that continuous-time quantum evolution, i.e. running the time evolution given by Schrödinger's equation gives us useful information when we run it on a "designer Hamiltonian." In [15],

the initial state is prepared by the algorithm, and the Hamiltonian is the input (the NAND tree), while in [7] the initial state is generated by consulting an "oracle" and the Hamiltonian is a carefully-designed part of the algorithm. Thus we can gain algorithmic power by designing the Hamiltonian, the initial state, or both.

We are interested in the computational power of continuous-time quantum walks in continuous spaces. In this paper, we focus on the case where the input to the algorithm consists of the initial state, and the Hamiltonian is simply $H = p^2/2m$, the free propagation of a massive particle in a zero potential, with only a kinetic energy term. Even in this simplest possible setting—perhaps the simplest non-trivial Hamiltonian encountered in quantum mechanics—we find that time evolution with Schrödinger's equation can reveal geometric information about the initial state more efficiently than classical computation can, especially in high dimensions.

Note that this case of Schrödinger's equation can also be seen as the low-energy limit of quantum walks on lattices. The time-dependent Schrödinger equation reads

$$-i\hbar\frac{\partial}{\partial t}\psi = H\psi\,.$$

For a free particle, $H$ is just the kinetic energy,

$$H = \frac{p^2}{2m} = -\hbar^2\frac{\nabla^2}{2m}\,.$$

The standard way to define continuous-time quantum walks on graphs uses the adjacency matrix as the Hamiltonian—or, with a rescaling of the energy, the graph Laplacian. Consider, for example, a 1-dimensional lattice with spacing $d$. The adjacency operator $A_d$ and discrete Laplacian $\nabla_d^2$ are:

$$A_d\,\psi(x) = \psi(x-d) + \psi(x+d)$$
$$\nabla_d^2\,\psi(x) = \psi(x-d) - 2\psi(x) + \psi(x+d)\,.$$

Since $A_d = \nabla_d^2 + 2$, using $A_d$ or $\nabla_d^2$ as the Hamiltonian only affects the time evolution by an overall phase.[1]

[1]For graphs with non-constant degree, $A$ and $\nabla^2$ correspond to different potentials, since they no longer differ by a constant.

In the limit $d \to 0$, where the lattice becomes continuous, or equivalently the low-energy limit where the particle's wavelength is much larger than $d$, we have $\nabla_d^2 \to d^2\nabla^2$. We can can see this by considering the complete eigenbasis $\psi_k(x) = \exp(ikx)$ shared by both operators. For the continuous case, each $\psi_k$ has an eigenvalue of $\lambda = -k^2$, while for the discrete case it is $\lambda_d = 2\cos kd - 2$. In the low-frequency limit $kd \ll 1$ where the wavelength $1/k$ is much larger than the lattice spacing, we have $\lambda_d(k) \to (kd)^2 = d^2\lambda(k)$.

Thus one of the canonical forms of quantum evolution in physics, Schrödinger's equation with $H = -\hbar^2\nabla^2/2m$, corresponds to a limit of quantum walks popular in computer science. But how do we get it to do something useful?

We borrow a problem of Yi-Kai Liu to find the center of spherically symmetric functions [29]. This problem fits into the Hidden Symmetry Subgroup Problem (HSSP) framework recently discussed by Decker et al. in [11]. Given a group $G$ acting on a set $M$, and a black-box function $f$ whose level sets partition $M$, the goal is to find the subgroup of $G$ that does not disturb this partition. If $M = \mathbb{R}^n$ and $G$ is the group of all Euclidean transformations in dimension $n$ (i.e., rotations and translations), then $f$'s level sets partition $M$ into spherical shells around its point of symmetry, and the desired subgroup is the set of rotations around that point.

We treat $f$ as a black-box function using only the ability to evaluate it at points or quantum superpositions of points, rather than analyzing its implementation in an attempt to discern its symmetry. If we prepare a wave function that is constant over a region containing the center (analogous to a uniform superposition in discrete algorithms), and evaluate $f$ and measure its output, we end up with a state analogous to a coset state in hidden subgroup problems. This picks out a particular spherical shell. Measurements are generally not exact, so we should end up with a state with some finite width, but concentrated on a shell, and spherically symmetric.

We analyze the evolution of a particular family of initial states concentrated on a spherical shell whose center and radius are unknown. After evolving for an appropriate amount of time, the state becomes highly concentrated near the center,

giving a probability proportional to $\epsilon$ of observing a position within $\epsilon$ of the center, regardless of the number of dimensions. While we make some assumptions about the initial state for simplicity, nearly any smooth initial state concentrated on a shell will produce similar results, as the main effect is geometric. Combining the information from multiple such shells localizes the center to higher accuracy.

While our entire analysis and inspiration is based on continuum behavior, we expect any concrete implementation to use a quantum computer to simulate the evolution. This gives a readily understandable way to implement a function and measure it. As long as the grid length is shorter than the shell width, a discrete version will not largely disturb the evolution. The evolution can be readily implemented by quantum Fourier transforms, then controlled phase gates, followed by the inverse quantum Fourier transform. Simulating an arena of side $L$ to a grid spacing of size $\epsilon$ is a state space of size $(L/\epsilon)^n$ which requires $O(n \log(L/\epsilon))$ position qudits. The $n$ Fourier transforms and inverses can each be done in depth $O(\log(L/\epsilon))$. The exponentially larger system size makes an implementation in linear optics unfeasible.

Classically, given the ability to evaluate functions, we cannot construct any analogue of the superpositions that we depend on in the quantum case. As we will see later, even adding a variety of reasonable seeming extra abilities to a classical computer does not help much.

As the effect here is due to wave interference, it seems we could do an equivalent job with a classical system supporting waves. This would of course require a system with the right number of dimensions, though these are not readily available. Further we would need to efficiently construct an initial condition concentrated on a spherical shell, given the ability to compute a given function. However, there is no a general method to construct such an initial state short of either separately computing many points and "plotting them" (note that verifying that a given sphere has the same center requires testing $O(n)$ points, and finding a suitable sphere would require many more), or finding the center some other way, letting a wave propagate out from the center, and then fixing the phases so that the evolution does not continue with the wave propagating outward.

## 4.2 States concentrated near surfaces, and their evolution

Our initial states are concentrated on a surface $S$ in $\mathbb{R}^n$, such as a spherical shell. One seemingly natural choice would be to take the wave function to be a Dirac "delta function" on $S$, such as $\delta(|x|-R)$ for a shell of radius $R$, but this is not a well-defined state, as it lies outside the space of $L^2$ functions. We instead want the density to approach a Dirac delta function of the radius, and examine the case where the width of this delta (i.e., the thickness of the shell) is much smaller than any other length scale in the problem. As discussed later in Section 4.4, such states arise naturally as a projective, approximate measurement of a spherically symmetric function applied to a constant wave function over a region containing its center.

We describe these states with a kernel $\varphi_0$ convolved with a Dirac delta function on the shell. Symbolically,

$$\psi_0(\vec{x}) = \frac{1}{\sqrt{N_0}} \int_S \mathrm{d}\vec{y}\, \varphi_0(\vec{x}-\vec{y})\,,$$

where $N_0$ is a normalization constant. This separates the geometry from the concentration, which gives us the flexibility to examine any surface $S$, without arduously constructing wave functions tailored to each surface.

We set $\hbar = 1$ and $m = 1$. For analytic ease we use an $n$-dimensional Gaussian kernel of initial width $w_0$ and take the limit $w_0 \to 0$. As the main effects are essentially geometric, any kernel that decays rapidly enough in both position space and frequency space will have similar behavior. Explicitly, this kernel is

$$\varphi_0(r; w_0) = \frac{1}{(\pi w_0^2)^{n/4}} \exp\left(-\frac{r^2}{2w_0^2}\right)\,. \tag{4.1}$$

This is normalized so that $\int \mathrm{d}\vec{x}\ |\varphi_0|^2 = 1$. However, this normalization will not be preserved when we integrate it over a given surface—hence the normalizing factor $N_0$. (Note that $N_0$ is not simply the area of $S$.)

Expressing the initial state as the integral of a kernel over a surface $S$ enables us to treat the evolution uniformly. By linearity, we can get the state at time $t$ by

integrating the kernel evolved to time $t$ over the surface. Letting $U(t)$ denote the unitary time evolution of Schrödinger's equation,

$$\begin{aligned}\psi_t(\vec{x}) &= U(t)\,\psi_0(\vec{x}) \\ &= \frac{1}{\sqrt{N_0}}\,U(t)\int_S \mathrm{d}\vec{y}\,\varphi_0(|\vec{x}-\vec{y}|\,;w_0) \\ &= \frac{1}{\sqrt{N_0}}\int_S \mathrm{d}\vec{y}\,\varphi_t(|\vec{x}-\vec{y}|\,;w_0)\,, \end{aligned} \tag{4.2}$$

where

$$\varphi_t(|\vec{x}-\vec{y}|\,;w_0) = U(t)\,\varphi_0(|\vec{x}-\vec{y}|\,;w_0)$$

denotes the time-evolved kernel.

Evolving the kernel is simplified by detouring through the Fourier domain. A bit of algebra gives the standard result of a spreading Gaussian wave function. We work here with a rescaled time

$$\tau = t/w_0^2\,,$$

and a final width $w$ where

$$w^2 = w_0^2(1+\tau^2) = w_0^2 + \frac{t^2}{w_0^2}\,.$$

Up to a global phase, the time-evolved kernel is then

$$\varphi_\tau(r;w) = \frac{1}{(\pi w^2)^{n/4}}\,\exp\!\left(-\frac{r^2(1+i\tau)}{2w^2}\right). \tag{4.3}$$

As for $\varphi_0$, this kernel is normalized so that $\int \mathrm{d}\vec{x}\,|\varphi_\tau|^2 = 1$. However, an additional normalization factor will be needed when integrating over a surface.

## 4.3 Spherical shells

We now focus the case where our initial state is concentrated on the surface of an $n$-dimensional hypersphere. As the state evolves, a remarkable thing occurs: the

spherical symmetry forces a large degree of constructive interference in the center, analogous to Arago's spot in two dimensions.

If the resulting probability density were smooth at the center, it would be exponentially unlikely, as a function of $n$, to observe a point within $\epsilon$ of the center, since a ball of radius $\epsilon$ has a volume $O(\epsilon^n)$. Instead, the probability density becomes singular, and scales as $r^{-(n-1)}$ down to a cutoff determined by the thickness of the initial state. Multiplying the density by the volume of a sphere, the distribution of the distance from the center is roughly uniform, and we observe a point within $\epsilon$ of the center with probability proportional to $\epsilon$ rather than $\epsilon^{n-1}$.

### 4.3.1 The initial state and its evolution

We denote the standard hypersphere in $n$ dimensions with radius 1 centered at the origin as $S^{n-1}$. Its surface area is

$$C_n = \left|S^{n-1}\right| = 2\pi^{n/2}/\Gamma(n/2) \ .$$

To obtain the final state we integrate the time-evolved kernel (4.3) over $S^{n-1}$. By symmetry, the result is a function only of the distance $r$ from the origin.

Without loss of generality, we can consider a point $(r, 0, \ldots, 0)$, where $x_1 = r$ and all other coördinates are 0. By symmetry around the $x_1$ axis, we can evaluate the integral over $S^{n-1}$ as a sum of integrals over $S^{n-2}$ shells of radius $\sin\theta$, where $\theta$ is the angle away from the $x_1$ axis. The distance $\ell(r, \theta)$ from this point to a point on such a shell is given by

$$\ell^2(r, \theta) = 1 + r^2 - 2r\cos\theta \, .$$

To simplify our expressions, we define

$$b = \frac{1 + i\tau}{2w^2} \, . \tag{4.4}$$

thus reducing (4.3) to

$$\varphi_\tau(\ell; w) = \frac{1}{(\pi w^2)^{n/4}} \exp\bigl(-b\ell^2\bigr) \ .$$

We also wish to extract the portion of $\exp(-b\ell^2)$ that does not depend on $\theta$, and hence define

$$g(b,r) = \exp\bigl(-b(1+r^2)\bigr)\ . \tag{4.5}$$

We label the coördinates on the surface of the sphere collectively as $\vec{\xi}$. Ignoring normalization, we can express the wave-function at time $\tau$ as follows:

$$\begin{aligned}
&\Psi_\tau(r;w)\\
&= \int\limits_{S^{n-1}} \mathrm{d}\vec{\xi}\,\varphi_\tau(\vec{\xi}-(r,0,\ldots,0);w)\\
&= \frac{1}{(\pi w^2)^{n/4}} \int\limits_{S^{n-1}} \mathrm{d}\vec{\xi}\,\exp\bigl(-b\ell^2(r,\theta)\bigr) &(4.6)\\
&= \frac{C_{n-1}}{(\pi w^2)^{n/4}}\, g(b,r) \int_0^\pi \exp(2br\cos\theta)\sin^{n-2}\theta\,\mathrm{d}\theta &(4.7)\\
&= \frac{C_{n-1}}{(\pi w^2)^{n/4}}\, g(b,r)\,\sqrt{\pi}\,\Gamma\left(\frac{n-1}{2}\right)\left(\frac{1}{br}\right)^{\frac{n}{2}-1} I_{\frac{n}{2}-1}(2br)\\
&= \frac{2\pi^{n/4}}{w^{n/2}}\, g(b,r)\left(\frac{1}{br}\right)^{\frac{n}{2}-1} I_{\frac{n}{2}-1}(2br)\,. &(4.8)
\end{aligned}$$

Here $I_{n/2-1}$ is the modified Bessel function of the first kind. As

$$|b| = \frac{\sqrt{1+\tau^2}}{2w^2} = \frac{1}{2w_0 w}$$

and

$$|g(b,r)|^2 = \exp\bigl(-(2\operatorname{Re} b)(1+r^2)\bigr) = \exp\left(-\frac{1+r^2}{w^2}\right)\ , \tag{4.9}$$

this gives an unnormalized probability density of:

$$|\Psi_\tau(r;w)|^2 = \frac{(4\pi)^{n/2} w_0^{n-2}}{w^2\, r^{n-2}} \exp\left(-\frac{1+r^2}{w^2}\right) \left|I_{\frac{n}{2}-1}(2br)\right|^2\,. \tag{4.10}$$

Integrating $|\Psi_\tau|^2$ over all space to produce a normalization constant does not appear to be easy. Happily, by unitarity we can derive the normalization constant from the initial state. Let

$$N_0 = \int \mathrm{d}V\ |\Psi_\tau|^2 = \int \mathrm{d}V\ |\Psi_0|^2\ .$$

At $\tau = 0$, we have $w = w_0$ and $b = 1/(2w^2)$, so

$$|\Psi_0(r; w_0)|^2 = \frac{(4\pi)^{n/2}\, w_0^{n-4}}{r^{n-2}} \exp\left(-\frac{1+r^2}{w_0^2}\right) I_{\frac{n}{2}-1}\left(\frac{r}{w_0^2}\right)^2 .$$

Then we have

$$\begin{aligned} N_0 &= \int r^{n-1}\, |\Psi_0(r; w_0)|^2 \, \mathrm{d}\vec{\xi}\, \mathrm{d}r \\ &= C_n \int r^{n-1}\, |\Psi_0(r; w_0)|^2 \, \mathrm{d}r \\ &= C_n (4\pi)^{n/2}\, w_0^{n-4} \int \exp\left(-\frac{1+r^2}{w_0^2}\right) I_{\frac{n}{2}-1}\left(\frac{r}{w_0^2}\right)^2 r \, \mathrm{d}r \\ &= C_n (4\pi)^{n/2}\, w_0^{n} \int \exp\left(-\frac{1+u^2 w_0^4}{w_0^2}\right) I_{\frac{n}{2}-1}(u)^2 \, u \, \mathrm{d}u \\ &= \frac{1}{2}\, C_n (4\pi)^{n/2}\, w_0^{n-2} \exp\left(-\frac{1}{2w_0^2}\right) I_{\frac{n}{2}-1}\left(\frac{1}{2w_0^2}\right) . \end{aligned}$$

Since our results are asymptotic, we are only interested in $N_0$ in the limit where the width $w_0$ of the initial shell approaches zero. We use the asymptotic behavior of the Bessel function,

$$I_\nu(x) \simeq \frac{\mathrm{e}^x}{\sqrt{2\pi x}} \text{ as } x \to \infty , \tag{4.11}$$

with $x = 1/(2w_0^2)$, to obtain

$$N_0 \simeq \frac{1}{2\sqrt{\pi}}\, C_n (4\pi)^{n/2}\, w_0^{n-1} . \tag{4.12}$$

As expected, $N_0$ scales as $w_0^{n-1}$, since $w_0$ sets a length scale in each of the $n-1$ directions of integration on the sphere.

### 4.3.2 A singularity at the center

Combining (4.10) and (4.12) gives a normalized probability density, in the limit $w_0 \to 0$, of

$$\begin{aligned} |\psi(r; w)|^2 &= \frac{1}{N_0} |\Psi(r; w)|^2 \\ &\simeq \frac{2\sqrt{\pi}}{C_n w_0 w^2 r^{n-2}} \exp\left(-\frac{1+r^2}{w^2}\right) \left|I_{\frac{n}{2}-1}(2br)\right|^2 . \end{aligned} \tag{4.13}$$

Integrating over spheres of radius $r$, the probability density that we observe a point at a distance $r$ from the origin is

$$\begin{aligned}\rho(r) &= C_n\, r^{n-1}\, |\psi(r;w)|^2 \\ &\simeq \frac{2\sqrt{\pi}r}{w_0 w^2}\, \exp\left(-\frac{1+r^2}{w^2}\right) \left|I_{\frac{n}{2}-1}(2br)\right|^2 .\end{aligned} \tag{4.14}$$

We will examine the asymptotic behavior of this distribution carefully in the next section. But as a preview, let us take $w = O(1)$, so that we evolve the wave function until its width is comparable to the radius $R = 1$ of the sphere. In the limit $w_0 \to 0$, we have

$$\tau \simeq w/w_0 \text{ and } b \simeq \frac{1}{2}\left(\frac{1}{w^2} + \frac{i}{w w_0}\right) . \tag{4.15}$$

Thus $b$ has a large imaginary part. As we move north on the complex plane, the Bessel function $I_\nu(z)$ oscillates as a function of $\operatorname{Im} z$. However, in the limit $y \to \infty$ the average of these oscillations behaves as the following generalization of (4.11),

$$|I_\nu(x+iy)|^2 \simeq \frac{\cosh 2x}{\pi y} . \tag{4.16}$$

In that case we have

$$\left|I_{\frac{n}{2}-1}(2br)\right|^2 \simeq \frac{w w_0}{\pi r}\, \cosh\frac{2r}{w^2}$$

Combining this with (4.14) gives

$$\begin{aligned}\rho(r) &\simeq \frac{2}{\sqrt{\pi}w}\, \exp\left(-\frac{1+r^2}{w^2}\right) \cosh\frac{2r}{w^2} \\ &= \frac{1}{\sqrt{\pi}w}\left(\exp\left(-\frac{(r-1)^2}{w^2}\right) + \exp\left(-\frac{(r+1)^2}{w^2}\right)\right) .\end{aligned} \tag{4.17}$$

Note that $\rho(r)$ is the sum of two Gaussians with variance $w^2/2$, one centered at $r = 1$ and the other at $r = -1$. The total probability is $\int_0^\infty \rho(r)\, \mathrm{d}r \,= 1$.

We will address all of this more precisely, including taking the oscillations into account, in the next section. But we already see that, if we fix $w$ and take the limits $w_0 \to 0$ and $\tau \to \infty$ appropriately, the distribution of observed distances $r$ is independent of $n$, and is smooth (i.e., nearly uniform) even at $r = 0$. To put it

differently, the probability distribution in $\mathbb{R}^n$ becomes singular at the origin, diverging as $r^{-(n-1)}$,

$$|\psi(r)|^2 \simeq \frac{1}{C_n\, r^{n-1}} \frac{2}{\sqrt{\pi} w} \exp\left(-\frac{1+r^2}{w^2}\right) \cosh\frac{2r}{w^2}\,.$$

Thus the probability of observing a point within $\epsilon$ of the origin is proportional to $\epsilon$, rather than to $\epsilon^n$ as it would be if we were dealing with a smooth probability distribution in $\mathbb{R}^n$.

### 4.3.3 Using the method of stationary phase

In this section, we perform a more exact asymptotic analysis of the evolved state. As alluded to in the previous section, we first define an appropriate limit. To be algorithmically relevant, the time evolution must exhibit interesting interference behavior. If $w$ is too small, the state will remain concentrated near the initial shell and have no opportunity for interference. Conversely, if $w$ is too large, most of the probability will propagate away to infinity. Thus to have reasonable interference near the center of the shell, we consider the case where $w$ is constant, but $w_0 \ll 1$. Setting two of $\{w, w_0, \tau\}$ fixes the third, so this requires $\tau \simeq w/w_0 \gg 1$.

Note that while the rescaled time $\tau$ is large, the actual time is small, $t = w_0^2 t \simeq w w_0 \ll 1$. This should not be unexpected: a small $w_0$ means there are large contributions from high-frequency modes. These propagate and interfere quickly, so a small $t$ can have large effect.

The details of the interference effects are hidden inside the behavior of the Bessel function $I_{n/2-1}$ on the complex plane. To expose them, we reconsider the equation that generated $I_{n/2-1}$ by integrating on the sphere. Recall the unnormalized wave function from (4.6),

$$\Psi(r) = \frac{1}{(\pi w^2)^{n/4}} \int\limits_{S^{n-1}} \mathrm{d}\vec{\xi} \exp\bigl(-b\ell^2(r,\theta)\bigr)\,.$$

This has a form vulnerable to attack by the method of stationary phase (see appendix B for a review). For non-zero $r$, this expression has two second-order station-

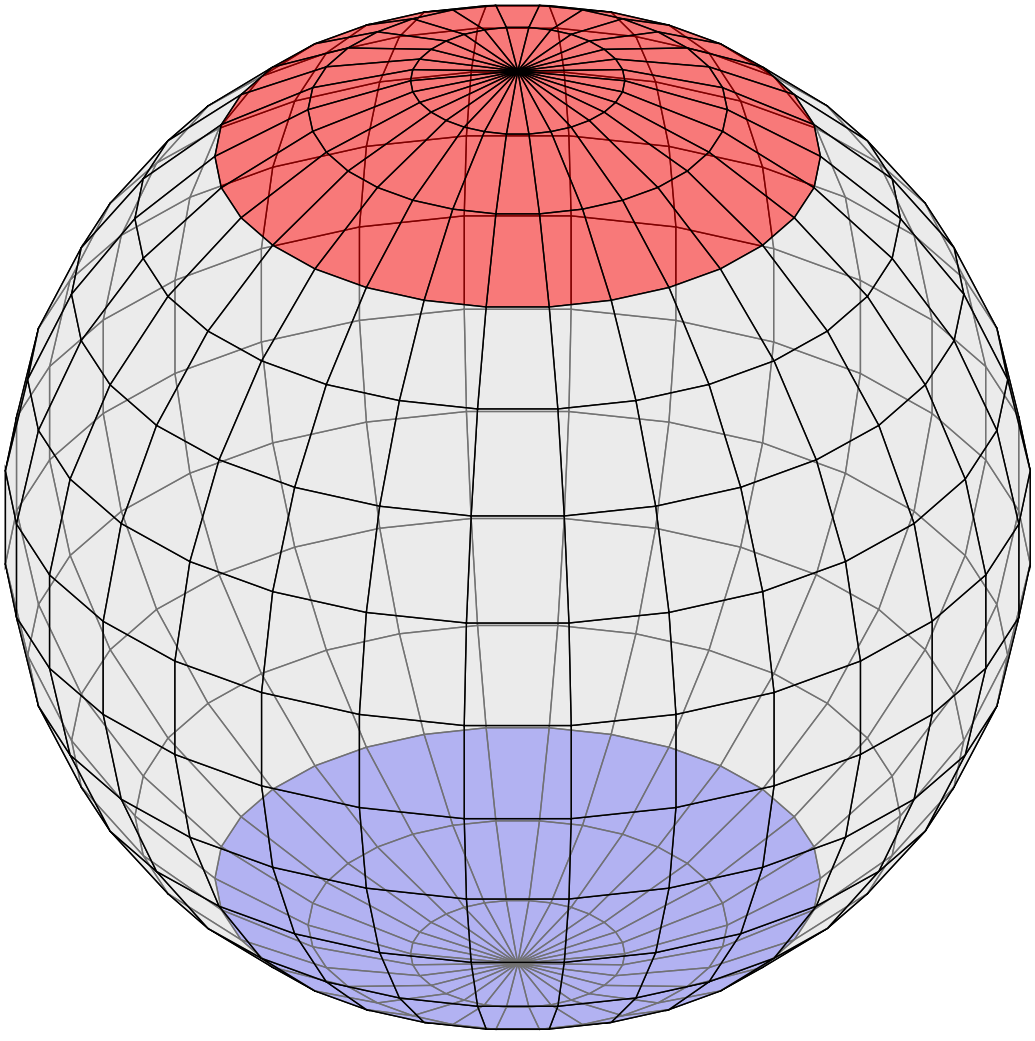

Figure 4.1: Caps on the sphere around the stationary points at the poles with significant contributions towards the integral.

ary points, one for $\vec{\xi}$ at each pole of the $x_1$ axis. There is a cap around each pole that contributes a large amount to the integral, as shown in Figure 4.1. The width of the cap necessary to get a good estimate of $\Psi(r)$ varies with $r$. As $r$ goes to 0, these caps grow until the entire sphere contributes to an *infinite*-order stationary point.

We do not integrate over $S^{n-2}$ shells of radius $\sin\theta$, and then apply the stationary phase method to the integral over $\theta$ appearing in (4.7). Since there are coördinate singularities at the poles $\theta = 0$ and $\theta = \pi$, the differentials $\sin^{n-2}\theta\,\mathrm{d}\theta$, and hence the full integrand, go to zero precisely at the points of interest. While there are forms

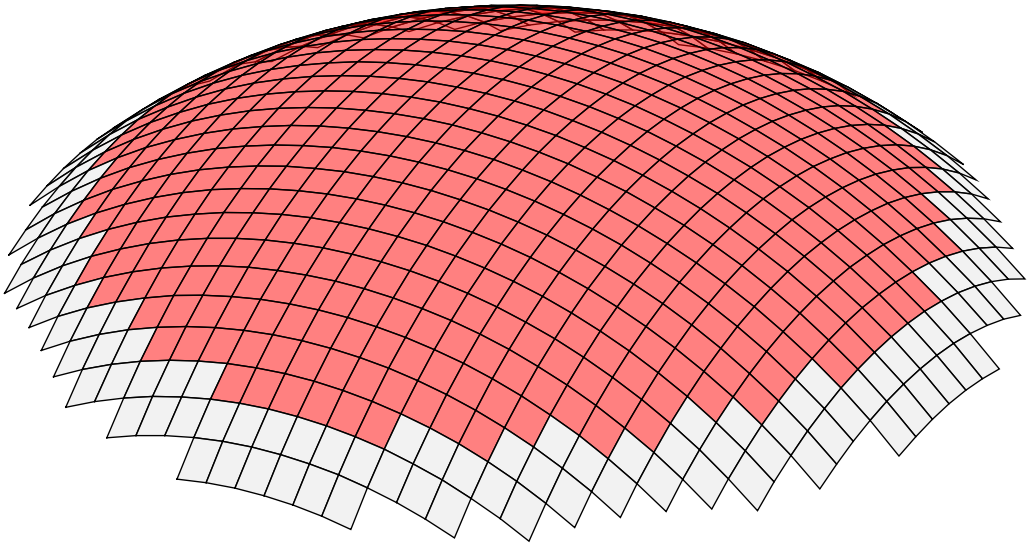

Figure 4.2: The non-singular coördinates $\xi_1, \ldots, \xi_{n-1}$.

of the stationary phase method that can handle this, that boil down to multiple integrations by parts, it is much simpler to rewrite the integral instead.

We choose a different coördinate system $\xi_1, \ldots, \xi_{n-1}$ for the surface of the sphere that is regular and orthonormal at the "north" and "south" poles where $\theta = 0$ and $\theta = \pi$ respectively, as shown in Figure 4.2. (Alternatively, we can think of this as using azimuthal coördinates but where the point $r$ lies beneath the "equator".) The angle away from the north pole is, to first order,

$$\theta \simeq s = \left( \sum_{j=1}^{n-1} \xi_j^2 \right)^{1/2} ,$$

or $\theta = \pi - s$ at the south pole. Thus

$$\begin{aligned} \ell^2 &= 1 + r^2 \mp 2r \cos\theta \\ &\simeq 1 + r^2 \mp r \left( 2 - \sum_{j=1}^{n-1} \xi_j^2 \right) , \end{aligned} \tag{4.18}$$

where $+$ and $-$ refer to the north and south poles respectively. This gives us an approximation for the phase of the integrand that is quadratic in $\vec{\xi}$, which is the essence of the method of stationary phase.

Letting

$$q = r\left(2 - \sum_{j=1}^{n-1} \xi_j^2\right) ,$$

we write $\Psi$ as the sum of two contributions, one from each pole:

$$\Psi \simeq \Psi_+ + \Psi_- ,$$

where (recall the definition of $g(b, r)$ from (4.5))

$$\Psi_\pm = \frac{g(b,r)}{(\pi w^2)^{n/4}} \int\limits_{\text{Patch}\pm} \left(\prod_{j=1}^{n-1} \mathrm{d}\xi_j\right) \exp(\pm bq) .$$

Expanding $\exp(\pm bq)$ gives (with $b$ as in (4.4))

$$\begin{aligned}
\exp(\pm bq) &= \exp\left(\pm br\left(2 - \sum_{j=1}^{n-1} \xi_j^2\right)\right) \\
&= \exp(\pm 2br) \prod_{j=1}^{n-1} \exp\bigl(\mp br\xi_j^2\bigr) \\
&= \exp(\pm 2br) \prod_{j=1}^{nL-1} \exp\left(\mp \frac{r\xi_j^2}{2w^2}\right) \exp\left(\mp \frac{i\tau r\xi_j^2}{2w^2}\right) .
\end{aligned}$$

This lets us rewrite $\Psi_\pm$ as a product of $n-1$ identical one-dimensional Gaussian integrals,

$$\Psi_\pm = \exp(\pm 2br)\, \frac{g(b,r)}{(\pi w^2)^{n/4}} \prod_{j=1}^{n-1} S_{j,\pm} , \tag{4.19}$$

where

$$S_{j,\pm} = \int \mathrm{d}\xi_j \, \exp\left(\mp \frac{i\tau r\xi_j^2}{2w^2}\right) \exp\left(\mp \frac{r\xi_j^2}{2w^2}\right) . \tag{4.20}$$

If we assume that $\xi_j$ ranges from $-\infty$ to $\infty$ then we can compute this integral exactly, giving

$$S_{j,\pm} = \sqrt{\frac{2\pi w^2}{\tau r}}\, \exp(\mp \pi i/4) .$$

This assumption is another ingredient in the method of stationary phase. It incurs an exponentially small error, since the integral is dominated by $\xi_j$ close to the unique stationary point $\xi_j = 0$.

Taking the product (4.19) then gives

$$\Psi_{\pm} = \frac{g(b,r)}{(\pi w^2)^{n/4}} \left(\frac{2\pi w^2}{\tau r}\right)^{(n-1)/2} \exp\left(\pm 2br \mp \frac{(n-1)\pi i}{4}\right) ,$$

and so

$$\Psi \simeq \frac{g(b,r)}{(\pi w^2)^{n/4}} \left(\frac{2\pi w^2}{\tau r}\right)^{(n-1)/2} \cdot 2\cosh\left(2br - \frac{(n-1)\pi i}{4}\right) .$$

Using the identity

$$|2\cosh(x+iy)|^2 = 2(\cosh 2x + \cos 2y) ,$$

and (4.9), we can finally calculate $|\Psi|^2$, the normalized probability density $|\psi|^2$, and the probability density $\rho(r)$ of the distance $r$ of the observed point from the origin:

$$|\Psi|^2 \simeq \frac{2^n \pi^{n/2-1} w^{n-2}}{\tau^{n-1} r^{n-1}} \exp\left(-\frac{1+r^2}{w^2}\right) \left[\cosh\frac{2r}{w^2} + \cos\left(\frac{2\tau r}{w^2} - \frac{(n-1)\pi}{2}\right)\right] \tag{4.21}$$

$$|\psi|^2 = \frac{|\Psi|^2}{N_0} \simeq \frac{1}{C_n\, r^{n-1}} \frac{2}{\sqrt{\pi} w} \exp\left(-\frac{1+r^2}{w^2}\right) \left[\cosh\frac{2r}{w^2} + \cos\left(\frac{2\tau r}{w^2} - \frac{(n-1)\pi}{2}\right)\right] \tag{4.22}$$

$$\rho(r) = C_n\, r^{n-1}\, |\psi|^2 \simeq \frac{2}{\sqrt{\pi} w} \exp\left(-\frac{1+r^2}{w^2}\right) \left[\cosh\frac{2r}{w^2} + \cos\left(\frac{2\tau r}{w^2} - \frac{(n-1)\pi}{2}\right)\right] \tag{4.23}$$

The oscillations in (4.23) come from interference between the contributions to $\Psi$ from the north and south poles. These oscillations modulate the overall probability with a wavelength $O(w^2/\tau)$, so in the limit $w_0 \to 0$, $\tau \to \infty$ and $w$ fixed these modulations become impossible to observe and can be ignored. At that point, we recover the expression (4.17) for $\rho(r)$ that we obtained from the asymptotic behavior of the Bessel function,

$$\rho(r) \simeq \frac{1}{\sqrt{\pi} w} \left(\exp\left(-\frac{(r-1)^2}{w^2}\right) + \exp\left(-\frac{(r+1)^2}{w^2}\right)\right) .$$

We compare (4.17) to (4.23), and to the exact distribution, in Figure 4.3.

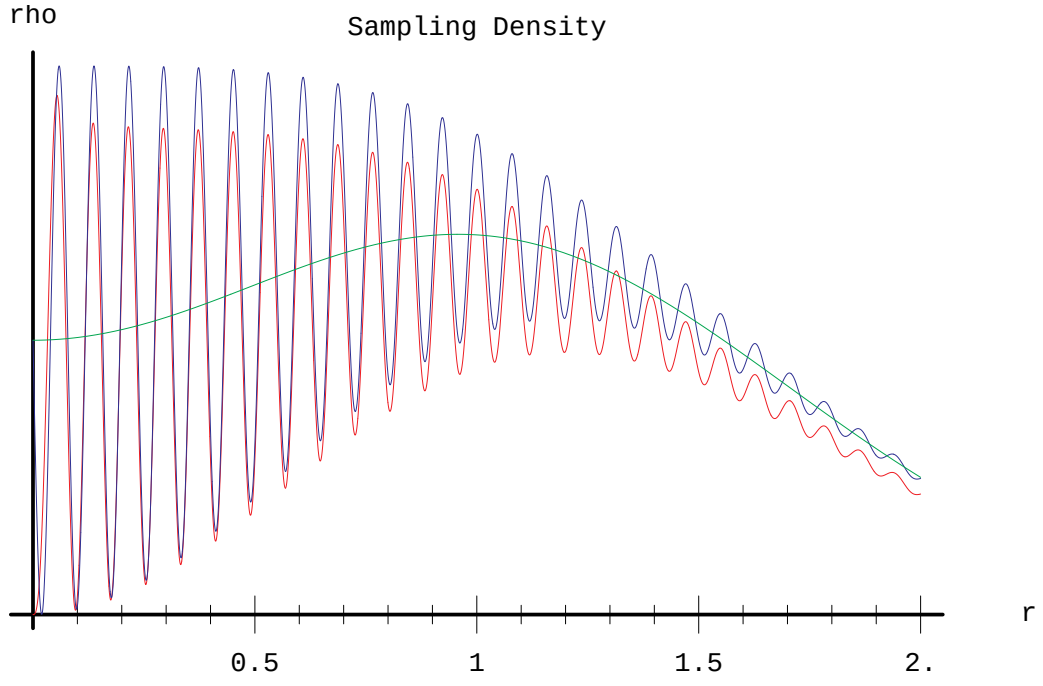


Figure 4.3: The exact probability density $\rho(r)$ (red) and our asymptotic expressions (4.23) (blue) and (4.17) (green) for $n = 4$, $\tau = 40$, $w = 1$, and $w_0 \simeq 1/40$.

### 4.3.4 Behavior near the origin

For small $r$, $\rho(r)$ is essentially constant, so for small $\epsilon$ we observe a point with $r < \epsilon$ with probability

$$P(r < \epsilon) \simeq \frac{2}{\sqrt{\pi} w} \exp\left(-\frac{1}{w^2}\right) \epsilon \,, \tag{4.24}$$

In the next section we will use the fact that $P(r < \epsilon) \propto \epsilon$ in our algorithm for finding the center of a sphere. More generally, we can write $P(r < \epsilon) = \int_0^\epsilon \rho(r)\, \mathrm{d}r$ in terms of the error function.

However, the expression (4.23) does not hold if $r$ is too small as a function of the other parameters. Our stationary phase approximation is equivalent to treating the sphere as a paraboloid at each pole, by writing

$$\cos\theta \approx \prod_{j=1}^{n-1} \cos\xi_j \approx \prod_{j=1}^{n-1} \left(1 - \frac{\xi_j^2}{2}\right) \approx 1 - \frac{1}{2} \sum_{j=1}^{n-1} \xi_j^2 \,.$$

The multiplicative error in the second-order Taylor series for $\cos\xi$ is $1 + O(\xi^2)$. However, taking the product over the $n-1$ coördinates at the cap raises this to $1 + O(n\xi^2)$. Thus the width of the cap that contributes significantly to the integral must obey $\xi \ll n^{-1/2}$ for our approximation to hold. Since the width of the Gaussian $S_{j,\pm}$ in (4.20) is $(\tau r)^{-1/2} \sim (w_0/r)^{1/2}$, this requires that

$$r \gg \frac{n}{\tau} \sim n w_0 \,.$$

To put this differently, if our ambition is to observe points a distance $r < \epsilon$ away from the origin, we need the initial thickness of the shell to be

$$w_0 \ll \frac{\epsilon}{n} \,.$$

This fits with the fact that the asymptotic formulas (4.11), (4.16) for the Bessel function $I_\nu(z)$ in the real and complex-valued case hold when $z \gg \nu^2$ and $y \gg \nu$ respectively.

Ultimately, the probability density $|\psi|^2$ at the origin must take some finite value, so that $\rho(0) = 0$. We can consider its behavior when $r \ll n w_0 \sim n/\tau$ so $|br| \ll n$, using the power series of the modified Bessel function:

$$I_\nu(2z) = z^\nu \sum_{j=0}^{\infty} \frac{z^{2j}}{\Gamma(\nu + 1 + j)\,\Gamma(j+1)} \,.$$

Thus to leading order,

$$I_{n/2-1}(2br) = \frac{(br)^{n/2-1}}{\Gamma(n/2)} \,,$$

and we can approximate (4.13) as

$$\begin{aligned} |\psi(r;w)|^2 &\simeq \frac{2\sqrt{\pi}}{C_n w_0 w^2} \exp\!\left(-\frac{1+r^2}{w^2}\right) \frac{|b|^{n-2}}{\Gamma(n/2)^2} \\ &\simeq \frac{2C_n\sqrt{\pi}}{(2\pi w)^n w_0^{n-1}} \exp\!\left(-\frac{1+r^2}{w^2}\right) . \end{aligned} \tag{4.25}$$

Note that the factors of $r^{n-2}$ cancel out. Thus the peak in the probability density at the origin grows as $w_0 \to 0$, but is finite for any given $w_0$.

## 4.4 Algorithmic applications

We have shown that physical evolution of a certain class of spherically symmetric states concentrates the probability near the center of the sphere. In this section, we show how to use this as a primitive to estimate the center of these spheres within a certain error with a small number of such experiments.

First, scaling arguments imply that for a spherical shell of radius $R$, the probability density (4.17) generalizes to

$$\rho(r; R, w) \simeq \frac{2}{\sqrt{\pi} w} \exp\left(-\frac{R^2 + r^2}{w^2}\right) \cosh \frac{2Rr}{w^2} \,. \tag{4.26}$$

Assume for the moment that $w_0$ and $R$ are known. If we evolve the initial state to time $t = w_0^2(w^2 - w_0^2)$ and measure the position, we have a reasonable probability of getting close to the center with one sample. Namely, for small $\epsilon$, we get within distance of $r$ with probability at least $\epsilon$, where

$$\frac{r}{R} \lesssim \frac{\sqrt{\pi}}{2} \frac{w}{R} \exp(R^2/w^2)\, \epsilon \,.$$

If $w/R$ is a constant—which we can arrange if we know that $R$ lies in some interval $[R_0, CR_0]$ for a constant $C$—then we have

$$\Pr\left[\frac{r}{R} < \epsilon\right] \propto \epsilon \,.$$

Our goal below is to show how to achieve small $r/R$ by combining multiple measurements of this kind.

For comparison, consider classical sampling, where each point is chosen uniformly from the same spherical shell. Each measurement restricts the location of the center, until after $n+1$ measurements we can determine the center by triangulation. (Three points determine a circle, four a sphere, and so on.) This is a somewhat apples-and-oranges comparison, since in our algorithm we do not assume that we have multiple copies of a state, all with the same radius, and our shells have a finite thickness $w_0$. But it is a good illustration of how the "curse of dimensionality"—the dependence of the number of measurements on the number of dimensions—behaves in a classical setting.

### 4.4.1 Combining multiple measurements

We have seen that by evolving the initial state, a single measurement achieves an estimate of the center of the sphere with error $r/R < \epsilon$ with probability proportional to $\epsilon$. Given $s$ such samples, $\vec{x}_1, \ldots, \vec{x}_s$, how can we combine them to achieve a smaller error $\epsilon$, and how does $s$ grow as $\epsilon$ decreases?

One approach is to simply take the mean. However, by the central limit theorem, this gives $\epsilon \sim 1/\sqrt{s}$ or

$$s \sim 1/\epsilon^2 \,.$$

This is no better than leaving the state unevolved, and taking the mean of $s$ samples from the initial shell. While the analysis is more complicated, the same is true for the generalized median (the point that minimizes the total distance to all the samples) or the coördinatewise median (choosing a basis and, along each axis, setting the coördinate of our estimate to the median of the corresponding coördinates of the samples). These methods do no better than the initial state because they do not take advantage of the singularity in the probability distribution. While $\rho(r)$ diverges as $r \to 0$, its variance is not much less than a spherical Gaussian of width $R$, and the error incurred by the mean and median depend primarily on this variance.

Instead, we use a maximum likelihood estimator (MLE) approach. We seek the position of the center $\vec{x}_0$ that would maximize the probability of observing the samples,

$$L(\vec{x}_0) = P(\{\vec{x}_1, \ldots, \vec{x}_s\} \,|\, \vec{x}_0) = \prod_{i=1}^{s} P(\vec{x}_i \,|\, \vec{x}_0) \,, \tag{4.27}$$

where the product holds because the samples are independent. However, in this case the MLE has a very unusual feature. Because the probability distribution of each sample scales roughly as

$$P(\vec{x}_i \,|\, \vec{x}_0) \propto |\vec{x}_i - \vec{x}_0|^{-(n-1)} \,, \tag{4.28}$$

the likelihood $L(\vec{x}_0)$ is not log convex, and has a peak close to each sample. As a result, many of the standard ways of dealing with MLEs do not work here. However,

this doesn't mean there's something wrong with the expression (4.27)—it accurately reflects the information available.

Since the MLE will necessarily be near one of the sample points, we restrict ourselves to considering only the sample points themselves as candidates. For simplicity, we will assume that the scaling (4.28) holds exactly. Since this diverges at $\vec{x}_0 = \vec{x}_i$, we "renormalize" away the term $P(\vec{x}_i \,|\, \vec{x}_i)$. We are then left with the following algorithm: given $s$ samples $\vec{x}_1, \ldots, \vec{x}_s$, choose the $\vec{x}_i$ that maximizes the product

$$\prod_{j \neq i} |\vec{x}_j - \vec{x}_i|^{-(n-1)} \,,$$

or, equivalently, that minimizes the product

$$\prod_{j \neq i} |\vec{x}_j - \vec{x}_i|^2 \,. \tag{4.29}$$

We will show, at least for certain values of $s$, that this algorithm often returns the sample $\vec{x}_i$ that is closest to the true center of the sphere. Since the probability density $\rho(r)$ of the distance from the center is roughly uniform near $r = 0$, it follows that the closest sample typically has

$$\frac{r}{R} \sim \frac{1}{s} \,,$$

so the number of samples it takes to achieve an error $r/R < \epsilon$ grows as

$$s \sim 1/\epsilon \,,$$

as long as $w_0 \ll \epsilon/n$. Thus we remove, or at least reduce, the curse of dimensionality, and achieve a quadratic speedup (measured in terms of the number of samples) over the naive algorithm of sampling $s$ times from the spherical shell and taking the mean.

### 4.4.2 Analyzing the MLE algorithm

To avoid technical details, we analyze the performance of our MLE algorithm in a simplified setting. By spherical symmetry, each sample $\vec{x}_i$ is chosen uniformly from

the spherical shell of radius $r$, where $r$ is chosen according to $\rho(r)$. Recall from (4.17) that, when $R = 1$, $\rho(r)$ is approximately a sum of two Gaussians, with mean 1 and $-1$, and variance $w^2/2 = O(1)$.

Here we instead assume that $\rho(r)$ is uniform in the unit interval. This avoids geometrical factors in the Gaussian distribution, and gives us a reasonable approximation that we can analyze rigorously. Below we report on numerical experiments showing that the behavior when $\rho(r)$ is given by (4.17) is similar.

In this setting, we will prove the following theorem.

**Theorem 1.** *There is are constants $C, D > 0$ such that, for $s < C\sqrt{n}$, the sample point closest to the center has the smallest product* (4.29) *of squared distances to the other samples, with probability at least $D$.*

Thus, up to $O(\sqrt{n})$ samples, the error $\epsilon$ scales as $1/s$. Indeed, numerical experiments show that this scaling persists up to significantly larger values of $s$ (and significantly smaller values of $\epsilon$).

We prove Theorem 1 by bounding the probability that any sample point other than the closest one has a smaller product of squared distances. Let $r_i = |\vec{x}_i|$ be the uniformly chosen radii, and sort them from smallest to largest so that $r_1 \leq r_2 \leq \cdots \leq r_s$. Let $\theta_i$ be the unit vector such that $\vec{x}_i = r_i\theta_i$. We will start by assuming that the radii $r_i$ are fixed and that the sequence $\{r_i\}$ has certain typical properties, and use the fact that the $\theta_i$ are uniform and independent on the sphere of radius 1. Then we will prove that $\{r_i\}$ does indeed have these properties with probability bounded above zero.

First we introduce some notation. Given two sample points $\vec{x}_i$ and $\vec{x}_j$, the distance squared between them is

$$D_{i,j} = |\vec{x}_j - \vec{x}_i|^2 = r_i^2 + r_j^2 + 2r_ir_j\theta_i \cdot \theta_j \,.$$

The product of these for a given point we denote

$$Q_j = \prod_{k \neq j} D_{j,k} \,.$$

Thus our goal is to show that, with probability bounded above zero,

$$Q_1 < Q_j \text{ for all } j > 1\,.$$

The following lemma will be useful:

**Lemma 2.** *If* $\mathbb{E}[Y] > \mathbb{E}[X]$*, then*

$$\Pr[X > Y] \leq \frac{\operatorname{Var}[X - Y]}{(\mathbb{E}[X] - \mathbb{E}[Y])^2}. \tag{4.30}$$

*Proof.* We apply Chebyshev's bound (e.g. [31, §A.3.2]) to the random variable $X - Y$, bounding the probability that it is zero, and hence more than $\mathbb{E}[X - Y]$ away from its mean. □

We define $\eta_j$ as the probability that our MLE algorithm judges $\vec{x}_j$ to be a better estimate than $\vec{x}_1$. Thus

$$\begin{aligned}\Pr[Q_j < Q_1] &= \eta_j \\ &\leq \frac{\operatorname{Var}[Q_j - Q_1]}{(\mathbb{E}[Q_j] - \mathbb{E}[Q_1])^2} \\ &= \frac{\operatorname{Var}[Q_1] + \operatorname{Var}[Q_j] - 2\operatorname{Cov}[Q_1, Q_j]}{(\mathbb{E}[Q_j] - \mathbb{E}[Q_1])^2}\,,\end{aligned} \tag{4.31}$$

where we recall the definition of the covariance of two random variables,

$$\operatorname{Cov}[X, Y] = \mathbb{E}[XY] - \mathbb{E}[X]\,\mathbb{E}[Y]\,.$$

The astute reader may have noticed that $Q_j$ and $Q_1$ share the common term $D_{1,j} = D_{j,1}$. We could instead have used variables defined without this common term, but this complicates the notation for no appreciable gain.

We will, of course, need to know how to work with the moments and correlations of the $D_{i,j}$. Assume the radii $r_i$ are fixed, and that the $\theta_i$ are uniform and independent on the unit sphere. First we prove a simple lemma:

**Lemma 3.** *Let* $\theta$ *be uniform and independent on the* $n$*-sphere of radius* 1*, and let* $y, z$ *be vectors in* $\mathbb{R}^n$*. Then*

$$\mathbb{E}[(\theta \cdot y)(\theta \cdot z)] = \frac{y \cdot z}{n}\,. \tag{4.32}$$

*In particular, for $i \neq j$ we have*

$$\mathbb{E}\left[(\theta_i \cdot \theta_j)^2\right] = \frac{1}{n}\,.$$

*Proof.* Treat $\theta$ as a column vector. By Schur's lemma, the expected outer product of $\theta$ with itself is

$$\mathbb{E}\left[\theta\theta^T\right] = \frac{1}{n}\mathbb{1}\,,$$

so

$$\mathbb{E}\left[(\theta \cdot y)(\theta \cdot z)\right] = y \cdot \mathbb{E}\left[\theta\theta^T\right] \cdot z = \frac{y \cdot z}{n}\,.$$

□

Now note that, for any $j$, the squared distances $D_{j,k}$ for $k \neq j$ are independent of each other, since by symmetry we can take $\theta_j$ to be any fixed vector. Moreover, if $j \neq k$ then $\mathbb{E}\left[\theta_j \cdot \theta_k\right] = 0$. Thus the expectations and second moments of the $Q_j$ are given by

$$\mathbb{E}\left[Q_j\right] = \prod_{k\neq j} \mathbb{E}\left[D_{j,k}\right] = \prod_{k\neq j} (r_j^2 + r_k^2)\,, \tag{4.33}$$

$$\mathbb{E}\left[Q_j^2\right] = \prod_{k\neq j} \mathbb{E}\left[D_{j,k}^2\right] = \prod_{k\neq j} \left((r_j^2 + r_k^2)^2 + \frac{4r_j^2 r_k^2}{n}\right)\,, \tag{4.34}$$

where in (4.34) we used Lemma 3.

The $Q_i$ for various $i$ are not independent. Happily, our next lemma shows that they are positively correlated with each other, which will help us bound the probability that $Q_i < Q_1$.

**Lemma 4.** *Assume the radii $r_i$ are fixed, and that the $\theta_i$ are uniform and independent on the unit sphere. Then for all distinct $i, j$,*

$$\mathrm{Cov}\left[Q_i, Q_j\right] \geq 0\,.$$

*Proof.* Let $\mathbb{E}_{\theta_k}[X]$ denote the expectation over $\theta_k$, while all other $\theta_i$ are held fixed. By Lemma 3,

$$\mathbb{E}_{\theta_k}[D_{i,k}D_{j,k}] = (r_i^2 + r_k^2)(r_j^2 + r_k^2) + 4r_i r_j r_k^2 \frac{\theta_i \cdot \theta_j}{n} .$$

Then

$$\begin{aligned}
\mathbb{E}\left[Q_i Q_j\right] &= \mathbb{E}\left[D_{i,j}^2 \prod_{k\neq i,j} D_{i,k}D_{j,k}\right] \\
&= \mathbb{E}_{\theta_i,\theta_j}\left[D_{i,j}^2 \prod_{k\neq i,j} \mathbb{E}_{\theta_k}[D_{i,k}D_{j,k}]\right] \\
&= \mathbb{E}_{\theta_i,\theta_j}\Bigg[\left(r_i^2 + r_j^2 + 2r_i r_j \theta_i \cdot \theta_j\right)^2 \\
&\quad \times \prod_{k\neq i,j} (r_i^2 + r_k^2)(r_j^2 + r_k^2) + 4r_i r_j r_k^2 \frac{\theta_i \cdot \theta_j}{n}\Bigg] ,
\end{aligned}$$

while

$$\begin{aligned}
\mathbb{E}\left[Q_i\right]\mathbb{E}\left[Q_j\right] &= \mathbb{E}\left[D_{i,j}\right]^2 \prod_{k\neq i,j} \mathbb{E}_{\theta_k}[D_{i,k}]\,\mathbb{E}_{\theta_k}[D_{j,k}] \\
&= (r_i^2 + r_j^2)^2 \prod_{k\neq i,j} (r_i^2 + r_k^2)(r_j^2 + r_k^2) . \qquad (4.35)
\end{aligned}$$

Comparing these expressions, we see that the covariance

$$\mathrm{Cov}\left[Q_i, Q_j\right] = \mathbb{E}\left[Q_i Q_j\right] - \mathbb{E}\left[Q_i\right]\mathbb{E}\left[Q_j\right] .$$

is a sum of terms, with positive coefficients, each of which contains the $m$th moment $\mathbb{E}\left[(\theta_i \cdot \theta_j)^m\right]$ for some integer $m \geq 0$. These moments are zero if $m$ is odd and positive if $m$ is even, so the covariance is non-negative. □

Lemma 4 and (4.31) give us a better bound on the probability $\eta_j$ that our algorithm chooses $\vec{x}_j$ instead of $\vec{x}_1$,

$$\eta_j \leq \frac{\mathrm{Var}\left[Q_1\right] + \mathrm{Var}\left[Q_j\right]}{\left(\mathbb{E}\left[Q_j\right] - \mathbb{E}\left[Q_1\right]\right)^2} . \qquad (4.36)$$

Our goal is to bound $\sum_{j=2}^{s} \eta_j$. We will find two trivial analytic facts helpful. First, if $0 \leq x \leq 1$, then

$$\exp(x) \leq 1 + 2x , \qquad (4.37)$$

since this bound holds at $x = 0$ and $x = 1$, and $\exp(x)$ is convex. Second, if $x, y \neq 0$, the arithmetic-geometric mean inequality (or elementary algebra) gives

$$\sqrt{x^2 y^2} \leq \frac{x^2 + y^2}{2},$$

and squaring both sides gives

$$\frac{4x^2 y^2}{(x^2 + y^2)^2} \leq 1. \tag{4.38}$$

Next we bound the relative variance of $Q_j$.

**Lemma 5.** *If $s \leq n$, then for all $1 \leq j \leq s$*

$$\frac{\operatorname{Var}[Q_j]}{\mathbb{E}[Q_j]^2} < \frac{2s}{n}. \tag{4.39}$$

*Proof.* From (4.33) and (4.34),

$$\begin{aligned}
C_j &= \frac{\mathbb{E}\left[Q_j^2\right]}{\mathbb{E}[Q_j]^2} - 1 \\
&= \prod_{k \neq j} \left(1 + \frac{4r_j^2 r_k^2/n}{(r_j^2 + r_k^2)^2}\right) - 1 \\
&= \exp\left(\sum_{k \neq j} \log\left(1 + \frac{4r_j^2 r_k^2}{n(r_j^2 + r_k^2)^2}\right)\right) - 1 \\
&\leq \exp\left(\frac{1}{n} \sum_{k \neq j} \frac{4r_j^2 r_k^2}{(r_j^2 + r_k^2)^2}\right) - 1 \\
&\leq \frac{2}{n} \sum_{k \neq j} \frac{4r_j^2 r_k^2}{(r_j^2 + r_k^2)^2} \\
&\leq \frac{2(s-1)}{n},
\end{aligned}$$

where we used (4.37), (4.38) and the fact that $s \leq n$ in the fifth line. □

We now wish to argue that $\vec{x}_2$ is the sample most likely to be ranked above $\vec{x}_1$ by the MLE algorithm. Consider the ratios between the expectations $\mathbb{E}[Q_j]$,

$$\gamma_{i,j} = \frac{\mathbb{E}[Q_i]}{\mathbb{E}[Q_j]}.$$

**Lemma 6.** *If $i > j$ then $\gamma_{i,j} > 1$, and $\gamma_{i,1} > \gamma_{j,1}$. In particular, $\gamma_{j,1} > \gamma_{2,1}$ for all $j > 2$.*

*Proof.* Applying (4.33) and dividing out the common term $r_i^2 + r_j^2$, we have

$$\gamma_{i,j} = \prod_{k \neq i,j} \frac{r_i^2 + r_k^2}{r_j^2 + r_k^2} \,.$$

Since $r_i > r_j$, each term is greater than one, so $\gamma_{i,j} > 1$. Then

$$\frac{\gamma_{i,1}}{\gamma_{j,1}} = \gamma_{i,j} > 1 \,.$$

□

Combining (4.36) with Lemmas 5 and 6 gives

$$\begin{aligned} \eta_j &\leq \frac{2s}{n} \frac{\mathbb{E}\left[Q_1\right]^2 + \mathbb{E}\left[Q_j\right]^2}{\left(\mathbb{E}\left[Q_j\right] - \mathbb{E}\left[Q_1\right]\right)^2} \\ &= \frac{2s}{n} \frac{1 + \gamma_{j,1}^2}{\left(\gamma_{j,1} - 1\right)^2} \\ &\leq \frac{2s}{n} \frac{1 + \gamma_{2,1}^2}{\left(\gamma_{2,1} - 1\right)^2} \,, \end{aligned}$$

where we used the fact that $(1 + x^2)/(x - 1)^2$ is monotonically decreasing for $x > 1$. Thus we have the following bound on the probability that the MLE algorithm fails to choose $\vec{x}_1$,

$$\sum_{j=2}^{s} \eta_j \leq \frac{2s^2}{n} \frac{1 + \gamma_{2,1}^2}{\left(\gamma_{2,1} - 1\right)^2} \,. \tag{4.40}$$

Our bound (4.40) shows that the MLE algorithm succeeds with high probability as long as $s \ll \sqrt{n}$ and $\gamma_{2,1}$ is bounded above one. We now use the fact that the radii are random, and prove that the latter condition holds with reasonably large probability.

**Lemma 7.** *Let $r_i$ for $1 \leq i \leq s$ be chosen uniformly and independently from the unit interval, and then sorted so that $r_1 < r_2 < \cdots < r_s$. Then*

$$\Pr[\gamma_{2,1} \geq 1.2] \geq 1/2 \,.$$

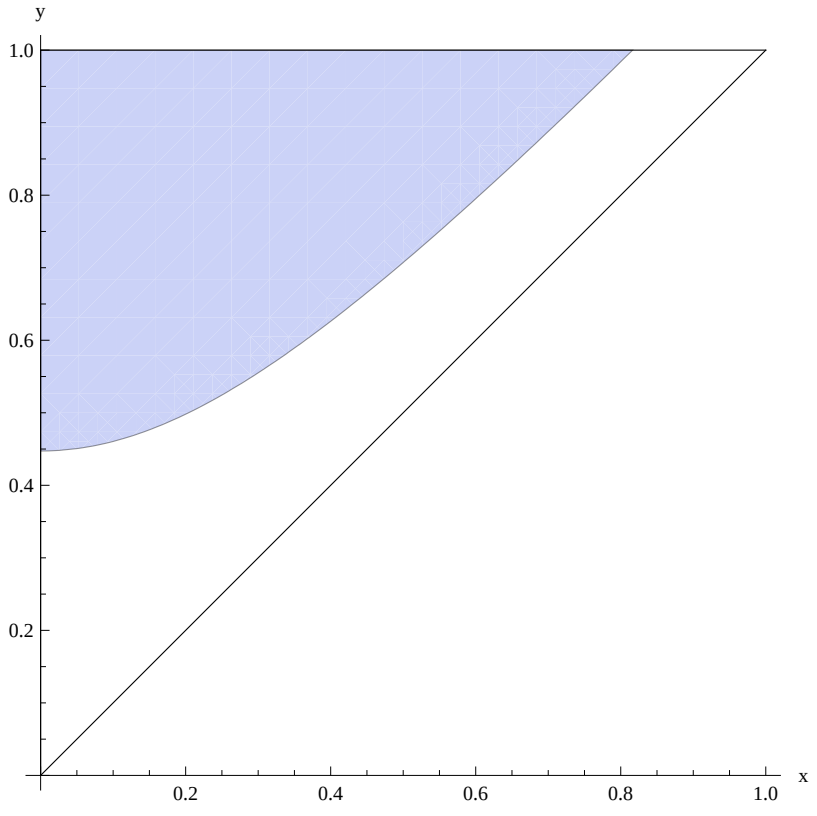


Figure 4.4: The region of the triangle $0 \leq x \leq y \leq 1$ such that $\gamma_{2,1} \geq 1.2$.

*Proof.* Taking just the first term in the product,

$$\gamma_{2,1} \geq \frac{r_2^2 + r_3^2}{r_1^2 + r_3^2} = \frac{1 + (r_2/r_3)^2}{1 + (r_1/r_3)^2}\,.$$

If we condition on the value of $r_3$, then $r_1/r_3$ and $r_2/r_3$ are chosen uniformly and independently from $[0, 1]$, and then sorted. If we call these $x = r_1/r_3$ and $y = r_2/r_3$, then $x$ and $y$ are uniform in the triangle defined by $0 \leq x \leq y \leq 1$. The region in which $\gamma_{2,1} \geq 1.2$ is shown in Figure 4.4, and the fraction of the triangle it occupies is greater than 0.55. □

Putting this together with (4.40), we have that with probability at least 1/2,

$$\begin{aligned}
&\gamma_{2,1} \geq 1.2 \\
&\frac{1+\gamma_{2,1}^2}{(\gamma_{2,1}-1)^2} \leq 61 \\
&\sum_{j=2}^{s} \eta_j \leq 122\,\frac{s^2}{n}\,.
\end{aligned}$$

Thus the total probability that the MLE algorithm fails is bounded by

$$\Pr[\text{MLE fails}] \leq \frac{1}{2}\left(1 + 122\,\frac{s^2}{n}\right)\,. \tag{4.41}$$

Thus, as long as $s/\sqrt{n}$ is sufficiently small, there is a constant probability of picking the point closest to the center with this method, and therefore obtaining an estimate with error

$$\epsilon = r/R \sim 1/s\,.$$

Note that our simplifying assumption that the radii are uniformly distributed enters in only two places: the bound on $\gamma_{2,1}$ in Lemma 7, and our claim that $r_1$ is typically of order $1/s$. Both of these still hold if the $r_i$ are chosen according to the distribution 4.17 rather than the uniform distribution on $[0, 1]$. Since $\rho(r)$ tends to a constant as $r \to 0$, as long as $s$ is reasonably large, with high probability the three smallest radii will be in a region where $\rho(r)$ is effectively uniform, and the geometry of Lemma 7 stays the same. Similarly, the expectation of the smallest radius $r_1$ is $C/s$ where $C$ is a geometrical constant.

However, our analysis is far looser than it could be at several points. Using Chebyshev's bound as in Lemma 2 is very pessimistic, since the variables $Q_j$ are far more concentrated than their variance alone would suggest. In Lemma 5 we treated all $s - 1$ terms in the sum as being close to 1; in reality, they decrease rapidly as $r_k$ gets far from $r_j$, which happens for almost all sets of radii. Finally, in the union bound (4.41) we treated $\eta_j$ for all $j$ as if it were as large as $\eta_2$, while these too rapidly decrease as $j$ increases.

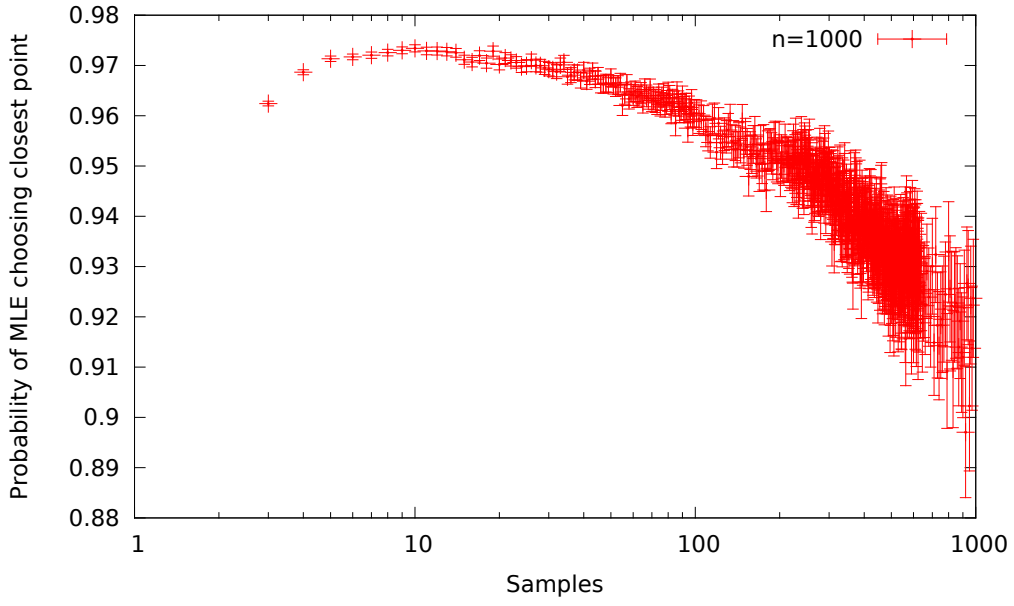


Figure 4.5: The probability that the MLE algorithm selects the closest sample point.

Indeed, we claim that the probability of failure is $o(1)$, as opposed to merely being bounded by a constant, and that this holds for a larger range of $s$ than our proof suggests—in particular, for $s$ well above $\sqrt{n}$. Numerical experiments where the $r_i$ are chosen according to (4.17) support this. Figure 4.5 shows that the MLE algorithm selects the closest sample point with probability close to 1 for $s$ ranging from 1 all the way up to $n$, and Figure 4.6 shows that the error $\epsilon$ scales as $s^{-\alpha}$ for $\alpha \approx 1$ over this entire range. Even when the selected point is not the closest, it is usually among the closest, and the error is still inversely proportional to the number of samples.

### 4.4.3 Iterating estimates to improve accuracy in spherically-symmetric functions

An attractive application of physics producing peaked measurement probability near the center is to find the center of a spherically symmetric function $f(\vec{x})$. By starting with a uniform superposition over a large region in $\mathbb{R}^n$ and measuring the value of

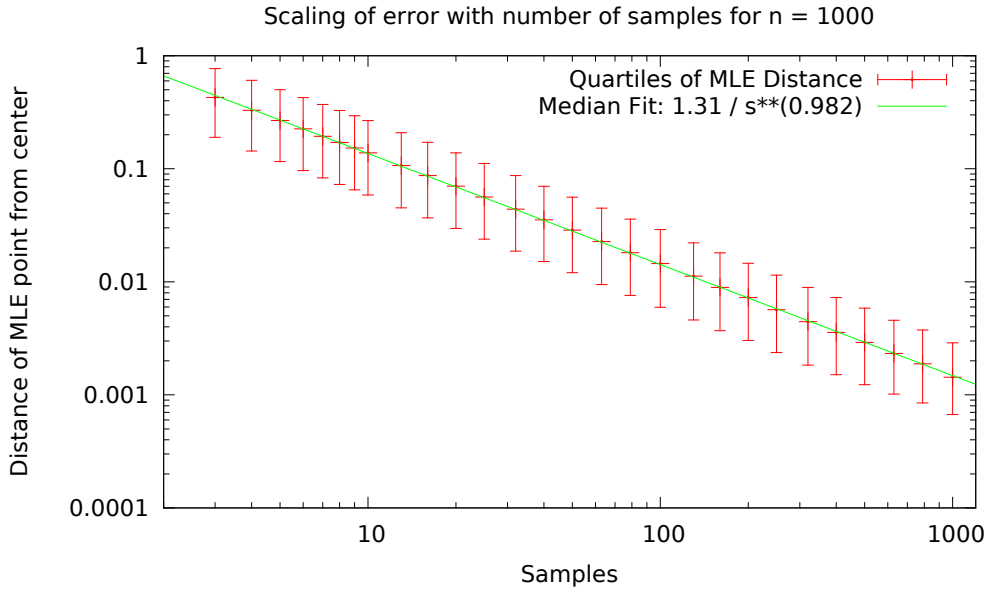


Figure 4.6: The error $\epsilon$ as a function of the number of samples $s$.

$f$, we should end up with a state concentrated on one of $f$'s level sets, i.e., on a spherical shell (or a union of several such shells if $f$ is not monotonic as a function of radius). Evolving for roughly the right amount of time and measuring the position $\vec{x}$ would then give a good estimate of the center with high probability.

Unfortunately, several technical problems arise with this approach. First of all, unlike in discrete Hilbert spaces such as the hidden subgroup problem on finite groups, there is no natural state that is precisely concentrated on a level set. Unless $f$ is constant, an exact measurement of $f$ would lead to an unphysical state, and is prohibited by the uncertainty principle in any case. If the value of $f$ is measured with some small Gaussian error $\nu$, however, we can obtain a state like those analyzed in the previous sections, where the width of the shell depends on $\nu$ and the derivative $\mathrm{d}f/\mathrm{d}r$.

Another difficulty is that we can't, of course, prepare a wave function constant over all of $\mathbb{R}^n$. If $f$ is defined on a compact space, such as a torus or hypersphere, then an analysis similar to the one we carried out in the previous sections should

work. However, if $f$ is defined on $\mathbb{R}^n$, our initial superposition can only be uniform on some finite region. Then another problem arises: since most of the volume of an $n$-sphere is near its boundary, the observed shell will probably be one of those that intersects the boundary of the region. In that case, the initial shell is incomplete, weakening the interference effects that cause the evolved state to concentrate at the origin.

However, we can arrange for these shells to be nearly-complete with high probability if we already know the center $\vec{x}_0$ of $f$ to within some accuracy. Specifically, if we know that $\vec{x}_0$ is at most $\delta$ from some estimate $\vec{x}_{\text{est}}$, we can sample from a ball $B$ centered at $\vec{x}_{\text{est}}$ of radius $R$, where $R$ depends on $\delta$. We can use the results of our algorithm to obtain a new estimate $\vec{x}'_{\text{est}}$ with a new error $\delta'$, and iterate until the error is as small as we desire, as in Liu [29]. Thus we will proceed in a series of rounds, where in each round we sample $s$ times from the evolved distribution, and reduce the error from $\delta$ to $\delta' \sim R/s$.

While it's true that each sample point in a round will be from the evolved version of a different shell, corresponding to a different value observed as output of $f$, the distributions from each run will be quite similar. The probability that we observe a given shell is proportional to its area, so the observed shell will be biased towards the largest possible shell in $B$; as we will see below, the initial radius will typically be $(1 - 1/n)R$. Since the thickness of the shell is small, it has a significant spread of spatial frequencies, and the interference effects at the center are quite robust to small changes in the initial radius.

How large does the radius $R$ of the sampled ball $B$ have to be for all this to work? We can think of the measurement process as choosing a random point $\vec{y}$ in $B$, and then producing the intersection of $B$ with the shell around $\vec{x}_0$ of radius $|\vec{y} - \vec{x}_0|$. First let's ask how often this shell is complete. In the worst case, when $\vec{x}_0$ is $\delta$ away from $\vec{x}_{\text{est}}$, the sphere of radius $R' = R - \delta$ around $\vec{x}_0$ is contained in $B$, and the observed shell is complete whenever $\vec{y}$ is in this sphere (see Figure 4.7). This occurs

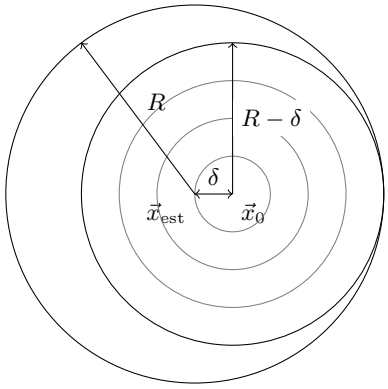


Figure 4.7: All shells around $\vec{x}_0$ with radii smaller than $R - \delta$ are complete.

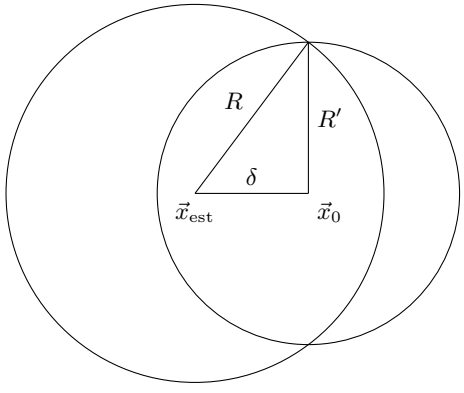


Figure 4.8: Scaling for attaining a half-shell is $O(\sqrt{n})$.

with probability

$$\Pr[\text{complete shell}] = \left(\frac{R'}{R}\right)^n = \left(1 - \frac{\delta}{R}\right)^n .$$

If $R = n\delta$, say, then this probability tends to 1/e.

However, as we showed in the previous section, we need $s \sim 1/\epsilon$ samples in order to obtain a new estimate whose error is $\epsilon R$. If $R = n\delta$, then this means we need more than $n$ samples in each round to make any progress, i.e., to achieve $\delta' < \delta$.

Happily, we don't need the shells to be complete. As long as they have a large enough solid angle, their fidelity with a complete shell will be large enough for the algorithm to work; see the discussion below. To get a sense for how $R$ should scale

to achieve a given solid angle with high probability, consider the case where the solid angle must be at least 1/2 of a complete shell. This occurs whenever $|\vec{y} - \vec{x}_0| \leq R'$, where the shell of radius $R'$ around $\vec{x}_0$ intersects the surface of $B$ at points that form a right angle with $\vec{x}_0$ and $\vec{x}_{\text{est}}$ as in Figure 4.8. In that case, Pythagoras' theorem gives

$$R' = \sqrt{R^2 - \delta^2} \approx R\left(1 - \frac{1}{2}\frac{\delta^2}{R^2}\right).$$

The left half of the sphere of radius $R'$ is contained in $B$, so the probability that $|\vec{y} - \vec{x}_0| \leq R'$ is

$$\Pr[\text{solid angle} \geq 1/2] \geq \frac{1}{2}\left(\frac{R'}{R}\right)^n = \frac{1}{2}\big(1 - O(n\delta^2/R^2)\big).$$

This is bounded above zero whenever $R = \delta\sqrt{n}$.

A constant fraction being good is good enough—our use of an MLE procedure is quite robust to having a few bad samples, because they can't be clustered in such a way to overpower the clustering near the center.

Suppose we seek a fixed $\epsilon$ accuracy. After $r$ rounds, we have:

$$\epsilon = \delta\left(\frac{k}{fs}\right)^r. \tag{4.42}$$

Then,

$$r = \log\epsilon - \frac{\log\delta}{\log(k/fs)}. \tag{4.43}$$

Minimizing the total number of samples $rs$ is minimizing

$$\frac{s}{\log(k/fs)} \tag{4.44}$$

hence

$$\log s = \log(k/f) + 1 \tag{4.45}$$

$$s = \mathrm{e}k/f. \tag{4.46}$$

If we construct a ball of radius $R$ around our estimate $\vec{x}_{\text{est}}$, and the true center $\vec{x}_0$ differ by $\delta$, then the larger the partial shell constructed, the worse the fraction

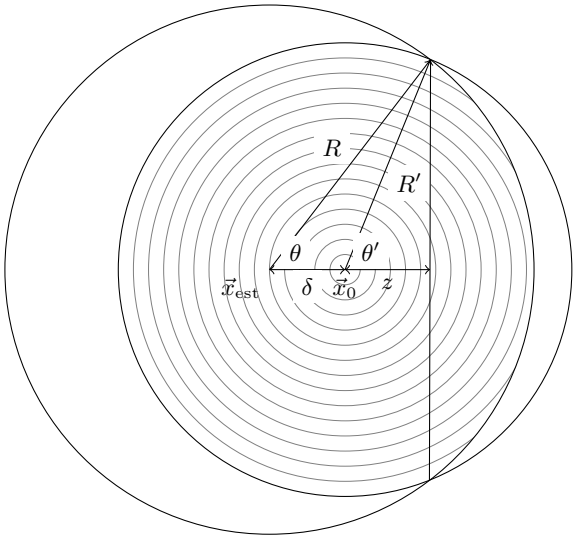


Figure 4.9: Partial shells from off-center sampling.

inside the sampled ball. How large must $R'$ be to still have a good a partial shell in the worst case? And how does the volume of the full ball compare to the volume of "good" shells and partial shells? If by "good" we mean "constant fractional area", then we can improve the scaling to $O(\sqrt{n})$.

The vast bulk of the area of a shell is very close to the equator. As a result, even a shell that is only slightly more than "half," in the sense that it includes latitudes $O(1/\sqrt{n})$ beyond the equator, is nearly complete. For a unit shell in $n$ dimensions, any given coördinate has a Beta distribution that is closely approximated by a Gaussian with variance $1/(n-3)$,

$$\begin{aligned} p(x) &= \frac{\Gamma(n/2)}{\sqrt{\pi}\,\Gamma((n-1)/2)}\,(1-x^2)^{(n-3)/2} \\ &\approx \frac{1}{\sqrt{2\pi/(n-3)}}\,\exp\!\left(-\frac{(n-3)x^2}{2}\right). \end{aligned}$$

Replacing $n-3$ with $n$ when $n$ is large, standard tail bounds on the Gaussian then give

$$\Pr\bigl[x > C/\sqrt{n}\bigr] \le \frac{1}{2}\exp\!\left(\frac{-C^2}{2}\right). \tag{4.47}$$

Now consider Figure 4.9. If $|\vec{y}-\vec{x}_0| \le R'$, then the observed shell is complete

except for points within $\theta'$ of one pole. If $\cos\theta' = C/\sqrt{n}$, then (4.47) tells us that, compared to a complete shell, its missing solid angle is $O(\exp(-C^2/2))$. If $|\phi\rangle$ denotes the uniform superposition on this observed shell and $|\psi\rangle$ denotes the uniform superposition on the complete shell of the same radius, the *fidelity* is the fractional solid angle,

$$\mathcal{F} = |\langle\psi|\phi\rangle|^2 = 1 - O(\exp(-C^2/2)) \,.$$

The probability that a given measurement distinguishes an observed shell from a complete one is at most $\log(1/\mathcal{F}) = O(\exp(-C^2/2))$. [6, 1] By setting $C$ to a large enough constant, we can make this probability as small as we like. By making $C$ slightly larger than a constant, say $C = \log s$, we can even make this probability $o(1/s)$, so that our algorithm will not notice the difference between these shells and complete ones over its entire operation.

The only question remaining is how large $R$ has to be for $\cos\theta' = C/\sqrt{n}$ to hold with high probability. The law of cosines gives the following relationship between $R$, $R'$, and $\delta$,

$$R^2 = R'^2 + \delta^2 + 2\delta R' \cos\theta' \,,$$

Although we have not rigorously proved this, the scaling here strongly suggests that we can still take $k = O(\sqrt{n})$, without having a significant chance of the state being distinguishable, and can thus use this procedure to speed up narrowing down locations of the center.

### 4.4.4 Comparison with previous work

This problem can be seen as a harder version of the hidden subgroup problem on the Euclidean group $E(n)$, the isometries of $\mathbb{R}^n$, with the hidden subgroup promised to be the group of rotations about some point. One difficulty is we are not given a function on $E(n)$, but only on $\mathbb{R}^n$, which $E(n)$ acts on. A standard Fourier transform gives only the representations associated with the translation subgroup, and there is

no more information to extract. This is not the only transform open to us, of course. We can trade off the translation information for other information. Yi-Kai Liu has shown that the quantum curvelet transform lets one find the center of a large class of spherically symmetric functions with a constant number of queries, and conjectures that this generalizes to the discrete case with the same asymptotics.

Liu specifically analyzes three algorithms: algorithm 1 finds the center of of ball given as a quantum sample state; algorithm 2 finds the center of of a radial function; and algorithm 3 recursively applies a variant of algorithm 2 to improve the scaling.

Liu's algorithm 1 uses 1 measurement of a ball state, and with $\Pr[r < \epsilon] \in \Omega(\epsilon^3/R^3)$. The curvelet transform is applied, and measurement then gives a line near which the center must be. This succeeds in giving fine enough angular resolution with probability at least $\Omega(\epsilon^2/R^2)$. Guessing a random point on the line gives a point within $\epsilon$ of the center with probability at least $\Omega(\epsilon/R)$. Liu does not address how having access to multiple copies of the initial state can improve things. Directly copying the techniques in algorithm 2 result in probability $\Omega(\epsilon^4/R^4)$, as both trials must have sufficient angular resolution. With many trials, this can get better, as one can pick out the two best results to combine. This is not directly comparable to our results as we only analyze the case of a spherical shell, not a ball.

Liu's algorithm 2 constructs a spherical shell by sampling from a radially symmetric function and measuring the output. The curvelet transform is applied and measurement yields a line that the center must be near. Repeating this gives another line. If these lines have sufficient angular resolution, points near their closest approach are near the center.

Constructing a region to sample over requires a promise that an estimate of the center is within some distance of the true center. The procedure of sampling creates a shell that is up to a factor of $n$ larger than the initial estimate.

The subprocedure to extract the center of a shell yields an uncertainty $\epsilon$ that depends on the radius of the shell $R$, the dimension $n$, and thickness of the shell $w_0$. With constant probability, $\epsilon < O(\sqrt{w_0 n R})$. Note that $R$ is n times the initial

estimated distance. though the construction of the shell is not heralded.

This is not directly comparable to our results. For a spherical shell, we have a probabilistic result that $\Pr[r < \epsilon] \in O(\epsilon/R)$, for $w_0 \ll \epsilon \ll R$. A constant probability gets us within a constant fraction of the distance to the center,

Liu's algorithm 3 iterates a variant of algorithm 2 in order to shrink the promise in each step, feeding in a better estimate for the next. In order to reduce the chances of constructing a bad shell to $O(1/s)$, he uses a ball of size $S$ larger than $R$, with $S$ tuned to the stage of the overall algorithm.

The present work answers this question in a rather different way. Rather than use a series of quantum gates as a circuit to transform the state to a basis that extracts algebraic information about the problem (lines given as a point on them and a direction, combined to locate the intersection), we use the geometry of the state and let Schrödinger's equation do the work for us.

This process of spherical symmetry in initial conditions causing a peak near the center should also remind one of the phenomenon of Poisson's spot (also known as Arago's spot).[2] A light source is blocked by an opaque circular disk, and the shadow projected onto a screen. At the center of the circular shadow is a bright spot. This is due to constructive interference; by Huygens's principle we can think of each spot on the edge of the disk being a point source of light. At the center, the light travels an equal distance, so has the same phase difference from each point along the edge.

Poisson's analysis was intended to demonstrate the absurdity of a wave theory of light. Arago's experimental verification showed that Poisson's sense of the absurd is not a reliable guide to the workings of the universe.

[2]In accordance with Stigler's law of eponymy, earlier observations were reported, including by Joseph-Nicolas Delisles in 1715, and Giacomo F. Maraldi in 1723, about 100 years before Poisson's derivation, or Arago's subsequent observational confirmation.

## 4.5 Discussion

Although the problems we describe are somewhat artificial, we have shown that even the simplest possible potential—a constant— still allows for interesting quantum information processing under evolution by the Schrödinger equation. These problems have a highly geometric nature, as opposed to the algebraic structures at which most quantum information processing is aimed. Specifically, given states concentrated near a high-dimensional hypersphere we can recover the center of the sphere with a constant number of samples, rather than the classically needed $O(n)$. Given a state concentrated near a plane curve, we can instead sample from its evolute. In both cases optimizing the evolution time requires knowing how concentrated the state is and what the scale of the initial state is, but neither need be known precisely. The finer the initial concentration, the more peaked we can make our final samples. Due to the nature of the Gaussian broadening, a sharper kernel actually requires less real time to evolve to the endpoint. This makes sense given the higher energy Fourier modes available, and is another expression of the standard time-energy tradeoff for continuous-time algorithms.

## Acknowledgments

This work was supported by the NSF under grant CCF-0829931 and the ARO under contract W911NF-04-R-0009. We are grateful to Josh Combes, Kacie Shelton, and Alex Russell for helpful discussions.

# Chapter 5

# Applications to less symmetric spaces: sampling from evolutes

In the previous chapter, we showed that scattering lets us find the center of an initial spherical shell. In this chapter, we explore similar phenomena on more general curves and surfaces. As in the spherical case, the probability distribution becomes singular, but now along the "evolute" of the initial state. We analyze these singularities, and show that various scaling exponents can arise.

This is preliminary work in an attempt to generalize the results of the previous chapter to less symmetric cases. We start in two dimensions, even though this means the computations done in this chapter can easily be classically replicated. In higher dimensions, an ellipsoidal shell where many of the axes are the same has nearly as much symmetry as the sphere. In general, what can be done with that? As we will see here, the scaling exponents for evolutes of arbitrary curves in 2-d are weaker than that for the center of a circle. Even for an ellipse, all the axes are different sizes, so the symmetry is already completely broken. Cases of only partially broken symmetry may prove different in higher dimensions.

## 5.1 Curves in two dimensions

Arago's spot provokes the question of what happens when the object blocking a light source has less symmetry than a circle. Arago's measurements have been extended to ellipses, or nearly equivalently, tilted circular disks, and other fairly simple geometric shapes [10, 9]. With these "shadow masks", the bright spot of the circular disk becomes a bright curve of the *evolute* of the boundary of the shape. The evolute is the set of all "centers of curvature" of a given curve. Indeed, the circular disk is a degenerate case, where the evolute reduces to the central point.

We can readily characterize the results by considering the interference properties of the light diffracted around the edge. The result at any point should be an integral of contributions from such diffractions around the edge. Using the method of stationary phase as a heuristic to guide our analysis, we see that at a generic interior point the interference should be mostly destructive. This occurs locally from most places along the edge, so long as the path length differences vary. However we expect local constructive interference from a few points, where there is a point on the edge such that the path length is stationary with respect to movements along the edge. To gain an understanding of the wave function's behavior we need analyze only these few points. Each such point generating a stationary path length can also be characterized as being a minimum or maximum path length. This in turn implies that the line connecting the point on the edge to the generic interior point being considered must be perpendicular to the edge. Turning this around, for a given point on the edge, the points in the interior to which it greatly contributes are the ones that lie on the perpendicular to the edge at that point. Different points along this line have lesser or greater contributions, peaking at the point that such that the boundary locally resembles a circular boundary with the same radius of curvature $R$. This is the "center of curvature" of the given location on the boundary curve. As the collection of all of these points is the *evolute* of the curve, we should expect brightening precisely on this curve.

As an example, the evolute of an ellipse is approximately shaped like a four-

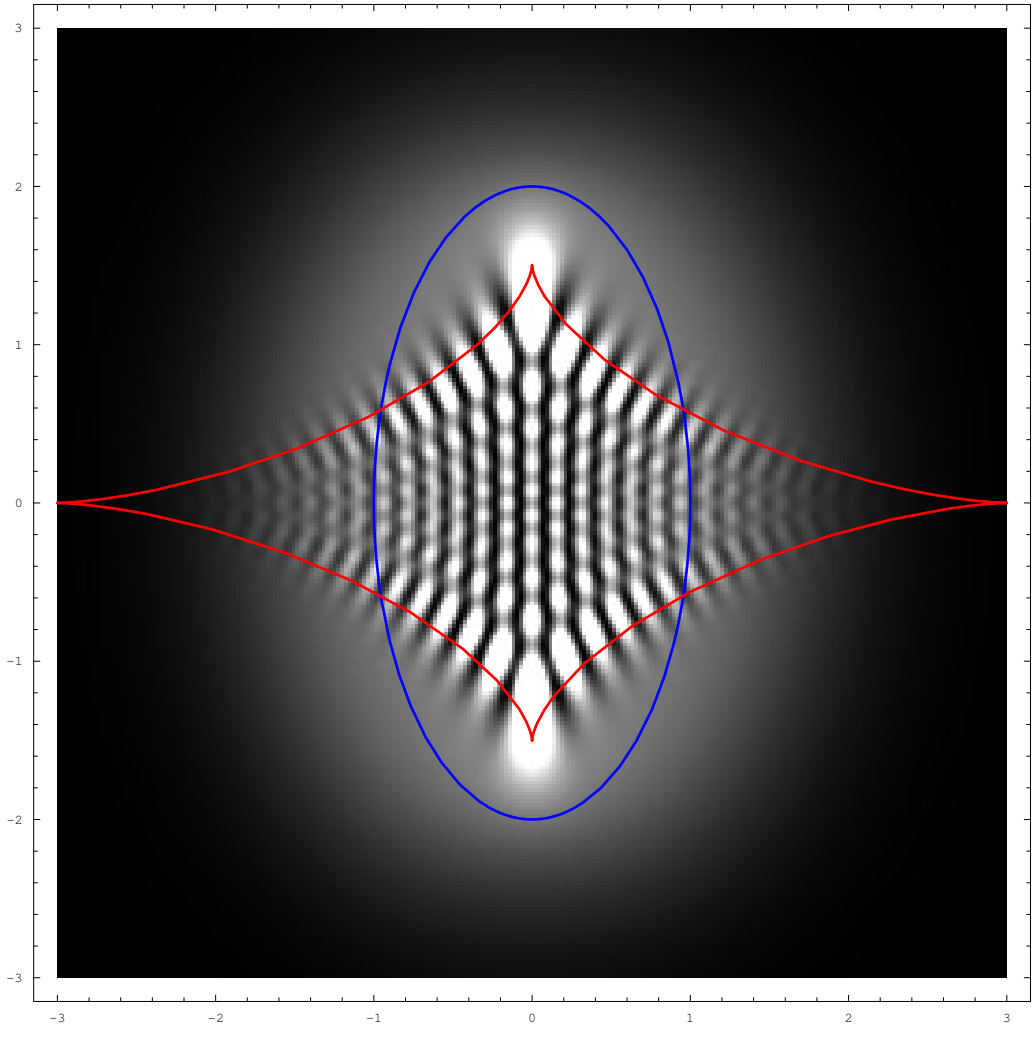


Figure 5.1: An ellipse, its evolute, and the interference pattern it generates, concentrated on the evolute.

pointed star. A point can have up to four stationary distances to the ellipse, with two being maxima and two minima. This occurs inside the evolute. For a point outside the evolute two stationary points exist, one maximum and one minimum. The evolute itself has three fixed points—approaching from the interior, two fixed points have merged producing one of higher order, and hence stronger concentration. Similarly, the cusps, with two fixed points have merged three of the four points creating another higher-order fixed point.

Although we produce similar patterns to diffraction around a connected opaque

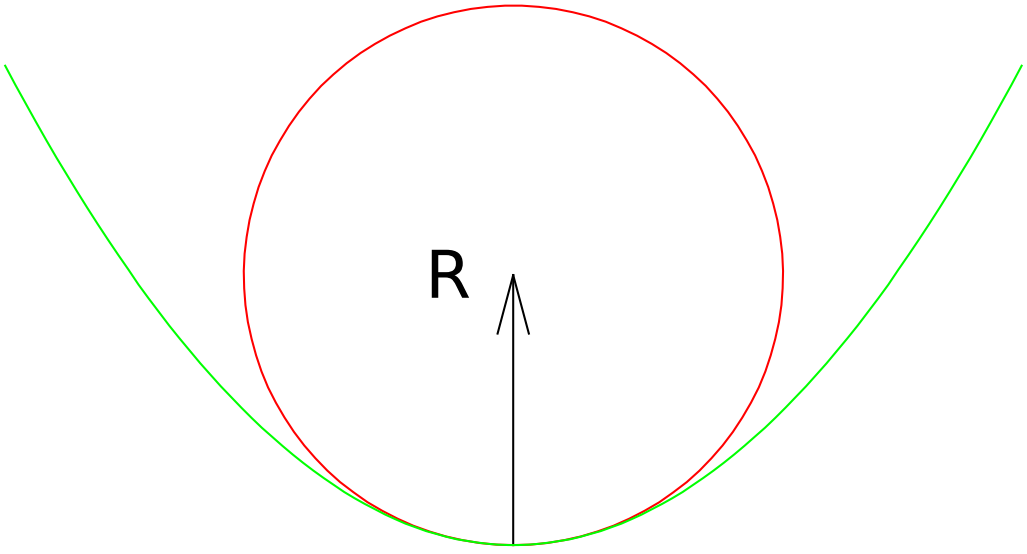


Figure 5.2: Each point on a curve has a circle (the "osculating circle") that will "fit snugly". The center is the point on the evolute generated by that point on the curve.

shape, our setup is more analogous to diffraction through a curved slot on a screen, at various widths. However, because only the boundaries produce strong effects, the analysis essentially carries through similarly. Physically we can understand this with Babinet's principle of complementary screens. Consider a screen with our curve as its edge. By Babinet's principle the complement of this screen produces a similar diffraction problem. Shrinking one slightly and enlarging the other will only slightly change the size of each diffraction pattern. Combining them gives a screen with a curved slot. This will give a diffraction pattern that is a combination of the two, with interference effects.

In the case of shadow masks and optical diffraction, this can be analyzed fairly rigorously with Kirchoff diffraction theory [25], though care must be taken that all the assumed limits are applicable. This would leave us with a frequency-dependent reduced wave equation in the plane perpendicular to propagation.

Rather than examining the steady state behavior of a beam satisfying such a reduced wave equation $\nabla^2\psi + k^2(\omega)\psi = 0$, we directly work in a 2-dimensional space, and examine the evolution of a wave function initially localized near a plane curve.

We attain qualitatively similar behavior to these diffraction effects with a different kernel, evolving under Schrödinger's equation. After a given amount of time, the probability density is concentrated on the evolute of the original curve. Given an initial concentration to a width $w_0$ around a given curve, we have calculated the scaling behavior of the width and probability density of the brightness along the evolute ($w_0^{1/3}$ and $w_0^{-1/3}$, respectively), and the width and probability density at cusps of the evolute ($w_0^{1/2}$ and $w_0^{-1/2}$, respectively), where $w_0$ is the initial width of the wave function around the starting curve.

We consider a curve with a Gaussian intensity profile. Again, rather than directly integrating across the width of the curve, we instead convolve a zero-width curve with a Gaussian kernel of a given width, which gives us the profile we want. Specializing the kernel of the previous sections to $n = 2$ gives

$$\varphi_0(r; w_0) = \frac{1}{w_0 \pi^{1/2}} \exp\left(-\frac{r^2}{2w_0^2}\right).$$

For example, the unnormalized wave function for a uniform line of width $w_0$ is constructed by:

$$\begin{aligned}
\Psi(x, y) &= \int \mathrm{d}x_0 \, \varphi_0(x, y; w_0) \\
&= \frac{1}{w_0 \pi^{1/2}} \int \mathrm{d}x_0 \, \exp\left(-\frac{(x - x_0)^2 + y^2}{2w_0}\right) \\
&= \frac{1}{w_0 \pi^{1/2}} N_2(w_0) \exp\left(-\frac{y^2}{2w_0}\right).
\end{aligned}$$

To represent a curve other than the line $y_0 = 0$, we use three functions $x(s)$, $y(s)$, and a density $\rho(s)$. We parameterize all three with a common variable $s$ which is integrated over to produce the final state. This variable $s$ can locally be considered proportional to the arc length of the curve. We would like to construct a wave function from this that goes to zero density off the curve, and a density proportional to $\rho(s)$ on the curve.

Evolving our state is handled by evolving the kernel inside the integral.

$$\varphi_\tau(r; w_0) = \frac{1}{(\pi w^2)^{1/2}} \exp\left(-\frac{(1 + i\tau) r^2}{2w^2}\right),$$

with $\tau = t/w_0^2$, and a final width $w^2 = w_0^2(1+\tau^2)$. By linearity, the wave function representing the time-evolved curve is just the integral of this kernel around the curve.

We have so far not handled the overall normalization factor. We want to maintain a constant probability integrated over the plane. Although it looks like the the dependence on $w$ would be accounted for in that the kernel has a constant squared integral, this will not work as the integral along the curve is also scaled by $w_0$. As a result, we must also take $N_2(w_0) \propto w_0^{-1/2}$. Strictly speaking, this scaling is exact only for the limit as $w_0 \to 0$, but this is precisely the limit we are interested in.

The exponential drop-off in amplitude taken from the original kernel remains, but with a growing length scale $w$ instead of the fixed $w_0$. In order to observe interesting interference effects, we consider the case where we covary $\tau$ and $w_0$ to keep $w$ fixed at a length scale corresponding to the plane curve. This results in finer and finer concentration along the evolute as the scaled times $\tau$ increases, with $w_0$ decreasing to compensate. (Note that for a fixed $w$, as $\tau$ increases, $t$ decreases. As the initial width $w_0$, the decomposition of the state into plane waves has more support on higher-frequency modes, requiring less unscaled time $t$ to operate.)

### 5.1.1 Analysis of a specific point

In our case, we write

$$\begin{aligned}\psi &= N_2(w_0)\frac{1}{w_0\pi^{1/2}}\int \mathrm{d}s\,\rho(s)\exp(-kq(s)) \\ &= N_2(w_0)\frac{1}{w_0\pi^{1/2}}\int \mathrm{d}s\,f(s)\exp(-i\tau\phi(s))\end{aligned}$$

with

$$\begin{aligned}q(s) &= (x-x(s))^2 + (y-y(s))^2), \\ f(s) &= N_2(w_0)\frac{1}{w_0\pi^{1/2}}\rho(s)\exp(-q(s)/2w),\end{aligned}$$

and

$$\phi(s) = q(s)/2w.$$

For $\phi(s)$ to be stationary, the distance squared $q$ must also be stationary. To set a common framework, we lay down new $u, v$ axes so that the stationary point under consideration is at the origin, with increasing $s$ leading strictly in the $u$ direction. We reparameterize the curve so that $u(s) = s$ and $v(0) = v'(0) = 0$.[1] which puts the point we are evaluating on the $v$ axis, at some $(0, h)$. Calculating the derivatives of $q$ with respect to $s$ gives:

$$\begin{aligned}
q(s) &= s^2 + (h - v(s))^2 & q(0) &= h^2 \\
q'(s) &= 2s - 2(h - v(s))v'(s) & q'(0) &= 0 \\
q''(s) &= 2 - 2(h - v(s))v''(s) + 2v'(s)^2 & q''(0) &= 2 - 2hv''(0) \\
q'''(s) &= 2(v(s) - h)v'''(s) + 6v'(s)v''(s) & q'''(0) &= -2hv'''(0) \\
q^{(4)}(s) &= 2(v(s) - h)v^{(4)}(s) + 8v'(s)v'''(s) + 6v''(s)v''(s) & q^{(4)}(0) &= -2hv^{(4)}(0) + 6v''(0)^2
\end{aligned}$$

So long as $q''(0) = 2 - 2hv''(0) \neq 0$, this is a second-order stationary point (the first derivative is zero, but not the second), and the method of stationary phase gives us the asymptotic value of the integral:

$$S(\tau) \simeq f(0) \exp(i\tau\phi(0)) \sqrt{\frac{2\pi}{\tau \left|\phi''(0)\right|}} \frac{1 \pm i}{\sqrt{2}} + O\left(\frac{1}{\tau}\right)$$

where the sign is that of $\phi''(0)$.

The probability density is the square of the wave function. Except near points with unusual degrees of symmetry, one point will dominate. The lowest order term in $1/\tau$ is

$$\begin{aligned}
|\psi|^2 &\simeq f(0)^2 \frac{2\pi}{\tau \left|\phi''(0)\right|} \\
&\simeq N(w) N_2(w_0) \frac{2\pi w \rho(0)^2}{\tau \left|1 - hv''(0)\right|} \exp(-h^2/w^2)
\end{aligned}$$

[1] Other conventions for the parameterization are possible. A natural one would be to require $s$ to be the path length, but this makes no difference in these arguments until the fourth order.

The overall density in the long time limit of each generic point asymptotes to a constant.

Not all points are generic, however. Note that this expression diverges as $\phi''(0) \to 0$, or as the distance squared becomes a third-order fixed point in $s$. This happens when $v''(0) \to 1/h$, which is just when $(0, h)$ is on the evolute of $(u(s), v(s))$.

If $q'''(0) \neq 0$, the method of stationary phase for a third-order stationary point then gives:

$$S(\tau) \simeq 2\, f(0) \cos\frac{\pi}{6}\Gamma\left(\frac{4}{3}\right)\left(\frac{6}{\tau\left|\phi^{(3)}(0)\right|}\right)^{1/3} + O(\tau^{-2/3}).$$

The density's lowest order term in $1/\tau$ then scales as

$$\begin{aligned}|\psi|^2 &\simeq f(0)^2\Gamma\left(\frac{4}{3}\right)^2 \left(6\tau\left|\phi^{(3)}(0)\right|\right)^{-2/3} \\ &\simeq \rho(0)^2 \exp(-\frac{h^2}{w^2}) N(w) N_2(w_0) \left(\frac{\tau\left|q^{(3)}(0)\right|}{w}\right)^{-2/3},\end{aligned}$$

or as $w_0^{-1/3} \simeq \tau^{1/3}$.

### 5.1.2 Width of region near the evolute

As we have seen, the second-order approximation diverges near the evolute. A reasonable way to characterize the distribution here requires not just an estimate of the probability density, but an estimate of the region over which this estimate should be used—the width of the evolute.

For any given point we evaluate, for a "high enough" $\tau$ value, the method of stationary phase eventually resolves separate stationary points into their individual contributions. However, different points will have different values of $\tau$ that are high enough. For a point that has a second-order stationary phase contribution but that is "nearly" a third-order contribution, it can be appropriate to instead analyze it as a third-order contribution. We set the width of the evolute by analyzing how this border changes with $\tau$.

The contribution to a point a distance $h$ from the curve is from an integral with an oscillatory portion of the form:

$$\int \mathrm{d}s \, \exp(i\tau(s^2 + h^2 + v(s)^2 - 2hv(s)),$$

As $h$ approaches $1/v''(s)$, the second-order contribution diverges, and evaluating the third-order contribution becomes necessary. The phase can be written as:

$$\begin{aligned}\phi &= \tau((1 - hv''(s))s^2 + h^2 - hv'''(s)s^3/3 + O(s^4)) \\ &= \tau((R - h)s^2/R - hv'''(s)s^3/3 + O(s^4)) \\ &= \tau(\delta s^2/R - hv'''(s)s^3/3 + O(s^4)),\end{aligned}$$

giving the integral:

$$\exp(i\tau h^2) \int \mathrm{d}s \, \exp(i\tau(\delta s^2/R - hv'''(s)s^3/3 + O(s^4)).$$

For a given value of $\tau$, what is the value of $h$ (or the width $\delta = R - h$) such that the $s^3$ term is significant compared to the $s^2$ term?

There is a rigorous extension of the stationary phase method dealing with "coalescing saddle points" that can help answer this. The coalescing can be seen the following way: another characterization of the evolute is the envelope of all normals from the curve. From this it is clear that as we approach the evolute, more than one stationary point can contribute, and eventually they must merge into one directly on the stationary point. However, this use does just as much to obscure as to illuminate what happens. Any finite number of stationary points with the same scaling will scale the same way. Instead we turn back to our heuristic explanation. The value contributed to the integral by some critical point is due only to small variations in the phase near that critical point. Large variations past some cut-off can be ignored. We can see that if this cut-off is large enough, the $s^3$ term must dominate and that if this cut-off is small enough, the $s^2$ term must dominate. Where neither dominates is the region we are trying to find. This will be a regime where the phase contribution for both powers of $s$ are of comparable magnitude. That is, for a given $\tau$, we want an $s_0$ satisfying $\phi_0 = \tau\delta s_0^2/R = \tau h v'''(s)s_0^3$. It is important to note that although we

need not be specific what $\phi_0$ is, it must be considered a constant that sets $s_0$—we cannot just divide through by $\tau s_0^2$. Indeed, it need not even be the same for the $s_0^2$ and $s_0^3$ terms—changing the ratio between them alters exactly where the trade-off is placed, but not how it scales. We set $s_0^2 = \phi_0 R/\tau\delta$. Substituting in the other equality gives $s_0^3 = \phi_0^{3/2} R^{3/2}/\tau^{3/2}\delta^{3/2}$ which reduces to $\phi_0 = \tau h v'''(s)\phi_0^{3/2} R^{3/2}/\tau^{3/2}\delta^{3/2}$. In the limit of small $\delta$, $h \simeq R$, resulting in $\delta \propto \tau^{-1/3}$—but with the exact switch-off determined by our choice of $\phi_0$.

This scaling of the width is exactly what we expect from conservation of probability: the probability density of finding the particle near the evolute grows more intense, but the as the probability of any given point off the evolute goes to a constant, then "near the evolute" must shrink accordingly in order to conserve probability.

### 5.1.3 Cusps—fourth order points

All closed curves (and some open ones) have evolutes with "cusps": these cusps are where the point generated on the evolute does not vary to first order in $s$. A bit of algebra shows that this is equivalent to $q'''(s) = 0$, and we must then look at the fourth derivative $q^{(4)}$ to determine the behavior of the contribution from these points.

Exactly analogously to before, the dominant term from the integral must scale as $\tau^{-1/4}$, the square as $\tau^{-1/2}$, and the normalized density as $\tau^{1/2}$.

The size of the region of validity of this growth is complicated to even define: it can be approached from multiple angles, with multiple scaling results. A generic approach has phase coefficients of $s^2$, $s^3$, and $s^4$, giving multiple possible trade-off regimes. However, along the line connecting to the generating point on the curve there are no $s^3$ terms. The argument from the previous section then has $\phi_0 = \tau\delta s_0^2/R = \tau\left(6/R^2 - 2v^{(4)}(0)(R-\delta)\right) s_0^4/12$.

$$
\begin{aligned}
s_0^2 &= \phi_0 R/\delta\tau \\
s_0^4 &= \phi_0^2 R^2/\delta^2\tau^2 \\
\phi_0 &= \tau(3/R^2 - v^{(4)}(0)(R-\delta))\phi_0^2 R^2/6\delta^2\tau^2 \\
6\tau &= (3/R^2 - v^{(4)}(0)(R-\delta))\phi_0 R^2/\delta^2 \\
6\tau\delta^2 &= (3 - v^{(4)}(0)(R-\delta)R^2)
\end{aligned}
$$

For small $\delta$, we have $\delta \simeq \tau^{-1/2}$. This is reassuring, as it fits with the growth in amplitude.

### 5.1.4 Similar work

This work covers similar ground to that of Michael Berry [4], in that it investigates how caustics scale when the length-scale of the producer changes. The set-up is different in that it was examining the steady-state intensity of constant-frequency source, with the wave being perturbed during its propagation by non-homogeneities in media. In our setup, the caustic is not caused by a perturbation, but by the initial conditions, which are a coherent superposition of many frequencies centered at zero (but with a well specified length-scale). We evolve forward a length-scale dependent amount of time to find a caustic, and we examine how the evolute parameters change with scale.

It should be clear that the the phenomenon of the evolute brightening due to constructive interference is ubiquitous—all that is needed is a quickly oscillating phase that varies rapidly with distance. For the scaling results to apply, the form of this must be that the phase varies as the square of the distance.

# Chapter 6

# Further directions

I think the present work speaks for itself regarding the usefulness of physical techniques being applied to the development of quantum algorithms. Going beyond this, we can ask what other sorts of problems we can attack with means like these.

There are many other groups that act as permutation on finite sets, and attacking their stabilizer subgroups seem a plausibly reasonable way to attack them. Indeed, Gábor Ivanyos looks at the subgroups of GL(n) that stabilize "flags" of nested subspaces.[22]

It also seems worthwhile to adapt the techniques of wave propagation from sources concentrated on surfaces to other geometric purposes. Are the reasonable ways to encode problems into potentials or Hamiltonians other than $p^2/2m$?

It also seems reasonable to look at other spaces than $\mathbb{R}^n$. Manifolds in general have less symmetry to work with, but a non-abelian variant would be possible on the surface of an $n$-sphere. Consider the problem of starting with a state concentrated at the "equator", and asking what direction the pole is in. Classical sampling requires $O(n)$ samples to eliminate all directions but one, but the concentration of propagating wave should get us the direction with few samples.

We also looked at states with less symmetry than a sphere, but only in two dimensions. The analogous problem in higher dimensions leaves much more room

for shapes to avoid any useful concentration of probability. Nonetheless, it might be possible to do the same thing in certain settings. If we have an ellipsoid with only a few different semi-axes, we might be able to bootstrap that to finding the center, and hence finding maxima or minima of functions in $\mathbb{R}^n$, as long as we're close enough so that the 2nd-order Taylor series is accurate enough.

# Appendix A

# Derivations

## A.1 Spreading of a Gaussian wave-packet via expansion in the Fourier domain

It's well known that a standard minimum-uncertainty Gaussian wave-packet will spread out over time. Here we reproduce that result.

We start with a Gaussian wave packet of width $w_0$, with standard quantum normalization.

$$\psi(x, w_0) = \frac{\exp(-x^2/2w_0^2)}{(\pi)^{1/4}{w_0}^{1/2}}$$

We Fourier transform it, to get the momentum-space representation.

$$\tilde{\psi}(k, w_0) = \frac{\sqrt{w_0}}{(\pi)^{1/4}} \exp(-{w_0}^2k^2/2)$$

The Hamiltonian and evolution operators are simple in this basis.

$$H = -\nabla^2/2 \simeq k^2/2$$
$$U(t) = \exp(-iHt) = \exp(it\nabla^2/2) \simeq \exp(-itk^2/2)$$

And some routine manipulations give us our desired propagator in momentum-

space.

$$\begin{aligned}\tilde{\psi}(k, w_0, t) &= \frac{\sqrt{w_0}}{\pi^{1/4}} \exp(-{w_0}^2 k^2/2) \exp(-itk^2/2) \\ &= \frac{\sqrt{w_0}}{\pi^{1/4}} \exp(-({w_0}^2 + it)k^2/2)\end{aligned}$$

We change back to position space to show the spreading. By analyticity, we can legitimately use standard Fourier transform tables treating this as a Gaussian with complex width.

$$\begin{aligned}\psi(x, w_0, t) &= \sqrt{w_0} \exp\left(\frac{-x^2}{2({w_0}^2 + it)}\right) / \pi^{1/4}(w_0^2 + it)^{1/2} \\ &= \frac{\sqrt{w_0}}{\pi^{1/4}(w_0^2 + it)^{1/2}} \exp\left(-x^2 \frac{w_0^2 - it}{2(w_0^4 + t^2)}\right) \\ &= \frac{\sqrt{w_0}}{\pi^{1/4}(w_0^2 + it)^{1/2}} \exp\left(-\frac{w_0^2 x^2}{2(w_0^4 + t^2)}\right) \exp\left(i \frac{t x^2}{2({w_0}^4 + t^2)}\right) \\ &= \frac{\sqrt{w_0} \exp(i\theta(w_0, t))}{\pi^{1/4}({w_0}^4 + t^2)^{1/4}} \exp\left(-\frac{w_0^2 x^2}{2(w_0^4 + t^2)}\right) \exp\left(i \frac{t x^2}{2(w_0^4 + t^2)}\right)\end{aligned}$$

Undimensionalizing $t = \tau {w_0}^2$

$$\psi(x, w_0, \tau) = \frac{\exp(i\theta(w_0, \tau))}{\pi^{1/4} {w_0}^{1/2}(1 + \tau^2)^{1/4}} \exp\left(-\frac{x^2}{2w_0^2(1 + \tau^2)}\right) \exp\left(-i \frac{\tau x^2}{2w_0^2(1 + \tau^2)}\right)$$

setting $w^2 = {w_0}^2(1 + \tau^2)$

$$\psi(x, w_0, \tau) = \frac{\exp(i\theta(w_0, \tau))}{\pi^{1/4} w^{1/2}} \exp\left(-\frac{x^2}{2w^2}\right) \exp\left(-i \frac{\tau x^2}{2w^2}\right)$$

Ignoring the physically irrelevant phase-factor, we end with:

$$\psi(x, w, \tau) = \frac{1}{\pi^{1/4} w^{1/2}} \exp\left(-\frac{x^2}{2w^2}\right) \exp\left(-i \frac{\tau x^2}{2w^2}\right)$$

# Appendix B

# Stationary phase method

For high $\tau$ the phase of time-evolved kernel (4.3) varies rapidly with position. When we integrate it over a given surface, the distance to a given point also varies, resulting in rapidly varying contribution to the amplitude $\psi(x;t)$. The rapidly varying phase in the integrand of (4.2) makes numerical integration difficult, and makes the integral difficult or impossible to evaluate exactly.

Fortunately we can turn this seeming liability into a very useful approximation capturing the prominent features of the evolution. We tame the wild behavior by turning to the class of asymptotic methods commonly called the *stationary phase approximation* [5, 14], or method of stationary phase. Similar techniques applied to contour integrals give the *saddle-point approximation*. We review here what kinds of results the method gives, and sketch out how it applies to a simplified integral.

These methods asymptotically approximate oscillatory integrals of the form

$$\int \mathrm{d}s\, f(s) \exp(i\tau\phi(s))$$

in the large $\tau$ limit. This approximation is as a function of the values (and derivatives) of both the phase $\tau\phi$ and envelope $f$ at *stationary points* where the derivative of the phase is zero, expressed as an inverse power series in $\tau$. For integrals of this form, most places have a rapidly varying phase, and nearby regions largely cancel due to destructive interference. Intuitively, the integral is dominated by contributions

from regions where the phase varies slowly; in the large $\tau$ limit these are solely the stationary points.

Much like Laplace's method, we may learn the scaling behavior by considering a region where the phase is nearly constant over a width $\delta s$ given by the parameter $\tau$. In the large $\tau$ limit this region shrinks, so that if $f(s)$ is a slowly-varying pre-factor, it is also effectively constant over it. This region then contributes $f(s)\exp(i\tau\phi(s))\delta s(\tau)$. For the scaling information, the exact definition of "effectively constant" does not really matter.

For concreteness, consider a simple integral such as $\int \exp(i\tau s^2)\,\mathrm{d}s$. Changing variables leads us to a standard form, $\frac{1}{\sqrt{\tau}}\int \exp(iz^2)\,\mathrm{d}z$, immediately demonstrating the scaling based on $\tau$. If there are boundaries, they must be changed to agree with the new variables. For unbounded ends (or a boundary at zero), there is nothing to change. However, even for bounded ends, in practice this works well for large enough $\tau$. The change of variables merely extends the boundaries by a factor of $\sqrt{\tau}$. Consider $S(z_f) = \int_0^{z_f} \exp(iz^2)\,\mathrm{d}z$. As $z_f$ increases, $S(z_f)$ spirals around and into the end point $\sqrt{\pi i}/2$. In this sense, the error of a finite approximation monotonically decreases, and for any given accuracy required, there is some $z_f$ such that $S(z_f)$ is a good approximation to the whole integral $\int_0^{\infty} \exp(iz^2)\,\mathrm{d}z$. Conversely, the whole integral provides a reasonable approximation to the bounded one, for large enough $\tau$.

It should be clear that this scaling argument works for any phase that is locally approximated by a simple power. If a power series is a good approximation, then for high enough $\tau$, the lowest power will dominate, and need be the only one considered. We say that a stationary point $s_0$ is order $m$ if $\phi$ behaves as $(s-s_0)^m$ in its vicinity. [1] In that case, the integral scales as $\tau^{-1/m}$.

Above we focus on the scaling, but the leading value can also be computed. We give only the behavior for a stationary point on a left boundary. Stationary points

[1] This is not a uniform convention. Also common is calling $s^m$ an order $m-1$ stationary point, so that $s^2$ is order 1.

on a right boundary are a trivial modification, and interior stationary points can be handled by splitting the integral in two.

$$\begin{aligned} S(\tau) &= \lim_{\tau\to\infty} \int_0^\infty \mathrm{d}s\, f(s) \exp(i\tau s^m) \\ &\simeq f(0)\Gamma\left(\frac{m+1}{m}\right)\left(\frac{i}{\tau}\right)^{1/m} + O(\tau^{-2/m}). \end{aligned}$$

When the integral has multiple stationary points, it is well approximated by a sum of terms like these, each with their own dominant term and asymptotic series in fractional powers of $\tau^{-1}$.

# Appendix C

# Code

Included here is the code written for numerical experiments related to Chapter 4.

## C.1 Box-Muller for Gaussian sampling

The following code uses the Box-Muller transform on a common random number generator, the Mersenne Twister [30], to generate numbers drawn from a standard Gaussian.

### C.1.1 box_muller.h

```
double box_muller(void);
```

### C.1.2 box_muller.c

```
#include <math.h>
#include "box_muller.h"
#include "mt19937ar.h"

struct box_muller_state {
```

```
        double value;
        int use;
};

void box_muller_state_init(struct box_muller_state * s)
{
        s->use = 0;
}

/* Polar form, with rejection sampling */
double box_muller_r(struct box_muller_state *state)
{
    if (state->use) {
        state->use = 0;
        return state->value;
    } else {
        float u, v, s, q;
        do {
            u = 2*genrand_real3()-1;
            v = 2*genrand_real3()-1;
            s = u*u + v*v;
        } while (s > 1);
        q = sqrt(-2 * log(s)/s);
        state->use = 1;
        state->value = v*q;
        return u*q;
    }
}

double box_muller(void)
{
```

```
        static struct box_muller_state s = { .use = 0 };
        return box_muller_r(&s);
}
```

## C.2 Generating samples

### C.2.1 sampled.c

```
#define _XOPEN_SOURCE
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <unistd.h>
#include "mt19937ar.h"
#include "box_muller.h"


void usage(const char *name)
{
    fprintf(stderr, "Usage:\n");
    fprintf(stderr, "%s [-s seed] -c count -d dimension\n",
        name);
    fprintf(stderr, "seed defaults to current unix time\n");
    exit(EXIT_FAILURE);
}

void gen_and_print_sample(int dimension) {
    double coords[dimension];
    double norm = 0;
    for(int d = 0; d < dimension; d++) {
```

```
            coords[d] = box_muller();
            norm += coords[d]*coords[d];
        }
        norm = sqrt(norm);
        norm /= (1.0 + box_muller()); // Gaussian centered at 1
        norm = fabs(norm);
        for(int d = 0; d < dimension; d++) {
            coords[d] /= norm;
            printf("%.15e ", coords[d]);
        }
        printf("\n");
}

void gen_and_print_samples(int samples, int dimension)
{
        for (int s = 0; s < samples; s++) {
            gen_and_print_sample(dimension);
        }
}

void parse_options(int argc, char **argv, int *dimension,
    unsigned long *seed, int *count)
{
        int c;
        c = getopt(argc, argv, "d:s:c:");
        while (c != -1) {
            switch (c) {
                case 'd':
                    *dimension = atoi(optarg);
                    break;
                case 's':
```

```
                *seed = atoi(optarg);
                break;
            case 'c':
                *count = atoi(optarg);
                break;
            case '?':
                usage(argv[0]);
                break;
            default:
                break;
        }
        c = getopt(argc, argv, "d:s:c:");
    }
    if (!*count || !*dimension) {
        usage(argv[0]);
    }
}

int main(int argc, char **argv)
{
        int samples = 0;
        int dimension = 0;
        unsigned long seed = 0;

        parse_options(argc, argv, &dimension, &seed, &
           samples);

        if (!seed) {
                time_t t = time(NULL);
                seed = t;
```

```
                fprintf(stderr, "No seed supplied.  Using
                    current unix time %lu.\n", seed);
        }

        init_genrand(seed);

        gen_and_print_samples(samples, dimension);

        exit(EXIT_SUCCESS);
}
```

## C.3 Analysis

### C.3.1 make-estimates.c

```
#define _XOPEN_SOURCE
#include <math.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <unistd.h>

/*************************************************/
/* What do we report?  0 means don't report, 1   */
/* means do.  We don't control the order.  The   */
/* last member "need_likelihoods" determines     */
/* whether we do the first two calculations.     */
/*************************************************/

struct which_outputs {
```

```
    unsigned mle:1;
    unsigned bayes_mean:1;
    unsigned mean:1;
    unsigned coordinate_median:1;
    unsigned geometric_median:1;
    unsigned mle_is_smallest_radius:1;
    unsigned need_likelihoods:1;
};

static void usage(const char *name)
{
    fprintf(stderr, "Usage:\n");
    fprintf(stderr, "%s -w <window size> -d <dimension> [ -
        lbmgc ] < samples.dat\n", name);
    fprintf(stderr, "    The -l, -b, -m, -c, and -g flags
        select estimators.\n");
    fprintf(stderr, "    If no flags are specified the
        radius of all estimators is reported.\n");
    fprintf(stderr, "    -l : report the radius of the
        maximum Likelihood estimator.\n");
    fprintf(stderr, "    -b : report the radius of the
        Bayesian mean estimator.\n");
    fprintf(stderr, "    -m : report the radius of the Mean
        .\n");
    fprintf(stderr, "    -c : report the radius of the
        Coordinate median.\n");
    fprintf(stderr, "    -g : report the radius of the
        Geometric median.\n");
    fprintf(stderr, "    ---------- or --\n");
    fprintf(stderr, "%s -w <window size> -d <dimension> -r
        -------- < samples.dat\n", name);
```

```
    fprintf(stderr, "    -r : report the number of times the
        MLE is the closest to the origin.\n");
    exit(EXIT_FAILURE);
}

static void parse_options(int argc, char **argv, int *window
    , int *dimension, struct which_outputs *which)
{
    int output_flags = 0;
    memset(which, 0, sizeof (*which));
    int c;
    do {
        c = getopt(argc, argv, "d:w:lbmcgr");
        switch (c) {
            case 'd':
                *dimension = atoi(optarg);
                break;
            case 'w':
                *window = atoi(optarg);
                break;
            case 'l':
                output_flags = 1;
                which->mle = 1;
                which->need_likelihoods = 1;
                break;
            case 'b':
                output_flags = 1;
                which->bayes_mean = 1;
                which->need_likelihoods = 1;
                break;
            case 'm':
```

```
                output_flags = 1;
                which->mean = 1;
                break;
            case 'c':
                output_flags = 1;
                which->coordinate_median = 1;
                break;
            case 'g':
                output_flags = 1;
                which->geometric_median = 1;
                break;
            case 'r':
                output_flags = 1;
                which->mle_is_smallest_radius = 1;
                which->need_likelihoods = 1;
                break;
            case '?':
                usage(argv[0]);
                break;
            default:
                break;
        }
    } while (c != -1);
    if (!*window) {
        usage(argv[0]);
    }
    if (!*dimension) {
        usage(argv[0]);
    }
    if (!output_flags) {
        which->mle=1;
```

```
            which->bayes_mean=1;
            which->mean=1;
            which->coordinate_median=1;
            which->geometric_median=1;
            which->mle_is_smallest_radius=0;
            which->need_likelihoods=1;
        }

        if ((which->mle || which->bayes_mean || which->mean
                || which->coordinate_median || which->
                   geometric_median) && which->
                   mle_is_smallest_radius) {
            usage(argv[0]);
        }
}

static int read_samples(int window, int dimension, double
   samples[window][dimension])
{
        double temp;
        for (int w = 0; w < window; w++) {
            for (int d = 0; d < dimension; d++) {
                if (scanf("%lf\n", &temp) != 1) {
                    return -1;
                }
                samples[w][d] = temp;
            }
        }
        return 0;
}
```

```
static double distance2(int dimension, const double p1[
    dimension], const double p2[dimension])
{
    double r2 = 0;
    for (int d = 0; d < dimension; d++) {
        r2 += (p1[d] - p2[d])*(p1[d] - p2[d]);
    }
    return r2;
}

static double log_cosh(double x)
{
    return x*log1p(exp(-2*x)) - log(2);
}

static double pairwise_log_likelihood(int dimension, const
    double p1[dimension], const double p2[dimension])
{
    double r2 = distance2(dimension, p1, p2);
    double r = sqrt(r2);
    // return exp(-r2) * cosh(2*r) * pow(r,1-dimension);
    return -r2 + (1-dimension)*r + log_cosh(2*r);
}

static int maximum_index(int window, const double samples[
    window])
{
    int index = 0;
    for (int i = 1; i < window; i++) {
        if (samples[i] > samples[index]) {
            index = i;
```

```
        }
    }

    return index;
}

static double sum(int window, const double likelihoods[
    window]) {
    double s = 0;
    for (int w = 0; w < window; w++) {
        s += likelihoods[w];
    }

    return s;
}

static void normalize_log_likelihoods(int window, double
    likelihoods[window])
{
    double logmax = likelihoods[maximum_index(window,
        likelihoods)];
    for (int w = 0; w < window; w++) {
        likelihoods[w] -= logmax;
    }
}

static void mean(int window, int dimension,
    double samples[window][dimension],
    double out[dimension])
{
    for (int d = 0; d < dimension; d++) {
```

```
        out[d] = 0;
    }

    for (int d = 0; d < dimension; d++) {
        for (int w = 0; w < window; w++) {
                out[d] += samples[w][d];
        }
        out[d] /= window;
    }
}

static int double_compare(const void *a, const void *b)
{
        double da = *(double *)a;
        double db = *(double *)b;
        if (da < db) return -1;
        if (da > db) return 1;
        return 0;
}

static void coordinate_median(int window, int dimension,
    double samples[window][dimension],
    double out[dimension])
{
    double coords[window];
    for (int d = 0; d < dimension; d++) {
        for (int w = 0; w < window; w++) {
            coords[w] = samples[w][d];
        }
        qsort(coords, window, sizeof(double), double_compare
            );
```

```
            out[d] = (coords[window/2] + coords[(window-1)/2])
                /2;
        }
    }

    static void weighted_mean(int window, int dimension,
        double samples[window][dimension],
        const double weights[window], double out[dimension])
    {
        for (int d = 0; d < dimension; d++) {
            out[d] = 0;
        }

        double total = sum(window, weights);

        for (int w = 0; w < window; w++) {
            double weight = weights[w] / total;
            for (int d = 0; d < dimension; d++) {
                out[d] += weight * samples[w][d];
            }
        }
    }

    static void inverse_distances(int window, int dimension,
        double samples[window][dimension],
        const double point[dimension],
        double weights[window])
    {
        for (int w = 0; w < window; w++) {
            double r2 = distance2(dimension, samples[w], point);
            weights[w] = 1/sqrt(r2);
```

```
    }
}

static void weiszfeld(int window, int dimension,
    double samples[window][dimension],
    double out[dimension])
{
    double estimate[2][dimension];
    double weights[window];
    const double epsilon = 1e-15;
    double delta;
    int which = 0;

    /* Initial point */
    mean(window, dimension, samples, estimate[which]);

    do {
        inverse_distances(window, dimension, samples,
            estimate[which], weights);
        which = 1 - which;
        weighted_mean(window, dimension, samples, weights,
            estimate[which]);
        delta = distance2(dimension, estimate[0], estimate
            [1]);
    } while (delta > epsilon);

    memmove(out, estimate[which], dimension * sizeof(double)
        );
}
```

```
static void log_likelihood_at_samples(int window, int
    dimension,
    double samples[window][dimension],
    double renormalized_likelihood[window])
{
    for (int w = 0; w < window; w++) {
        renormalized_likelihood[w] = 0;
    }

    for (int w1 = 0; w1 < window -1 ; w1++) {
        for (int w2 = w1 + 1; w2 < window; w2++) {
            double l = pairwise_log_likelihood(dimension,
                samples[w1], samples[w2]);
            renormalized_likelihood[w1] += l;
            renormalized_likelihood[w2] += l;
        }
    }
}

void dump_samples_and_likelihoods(int window, int dimension,
    double samples[window][dimension],
    const double renormalized_likelihood[window])
{
    for (int w = 0; w < window; w++) {
        for (int d = 0; d < dimension; d++) {
            fprintf(stderr, "%lf ", samples[w][d]);
        }
        fprintf(stderr, "%lf\n", renormalized_likelihood[w])
            ;
    }
}
```

```
static double radius(int dimension, const double p[dimension
    ]) {
    double r2 = 0;

    for (int d = 0; d < dimension; d++) {
        r2 += p[d]*p[d];
    }

    return sqrt(r2);
}

static void map_double_destructive(double (*func)(double),
    int size, double data[size])
{
    for (int i = 0; i < size; i++) {
        data[i] = func(data[i]);
    }
}

static void format_radius(int not_first, double radius)
{
    if (not_first) {
        fputs("\t", stdout);
    }
    printf("%.15le", radius);
}

static void process(int window, int dimension, struct
    which_outputs which)
{
```

```
    double samples[window][dimension];
    double renormalized_log_likelihood[window];
    double point[dimension];
    int good = 0;
    int bad = 0;

    while(!read_samples(window, dimension, samples)) {
        int not_first = 0;
        if (which.need_likelihoods) {
            log_likelihood_at_samples(window, dimension,
               samples, renormalized_log_likelihood);
            normalize_log_likelihoods(window,
               renormalized_log_likelihood);
            if (which.mle) {
                double r_max_l = radius(dimension, samples[
                   maximum_index(window,
                   renormalized_log_likelihood)]);
                format_radius(not_first++, r_max_l);
            }
            if (which.bayes_mean) {
                map_double_destructive(exp, window,
                   renormalized_log_likelihood);
                weighted_mean(window, dimension, samples,
                   renormalized_log_likelihood, point);
                double r_b_mean = radius(dimension, point);
                format_radius(not_first++, r_b_mean);
            }
            if (which.mle_is_smallest_radius) {
                double negative_radii[window];
                for (int s = 0; s < window; s++) {
```

```
                    negative_radii[s] = -radius(dimension,
                        samples[s]);
                }
                int radius_index = maximum_index(window,
                    negative_radii);
                int mle_index = maximum_index(window,
                    renormalized_log_likelihood);
                if (radius_index == mle_index) {
                    good++;
                } else {
                    bad++;
                }
            }
        }
        if (which.mean) {
            mean(window, dimension, samples, point);
            double r_mean = radius(dimension, point);
            format_radius(not_first++, r_mean);
        }
        if (which.coordinate_median) {
            coordinate_median(window, dimension, samples,
                point);
            double r_c_median = radius(dimension, point);
            format_radius(not_first++, r_c_median);
        }
        if (which.geometric_median) {
            weiszfeld(window, dimension, samples, point);
            double r_g_median = radius(dimension, point);
            format_radius(not_first++, r_g_median);
        }
        if (!which.mle_is_smallest_radius) {
```

```
            puts("");
        }
    }
    if (which.mle_is_smallest_radius) {
        printf("%d %d %d\n", good, bad, good+bad);
    }
}

int main(int argc, char **argv)
{
    int window = 0, dimension = 0;
    struct which_outputs which;

    parse_options(argc, argv, &window, &dimension, &which);

    process(window, dimension, which);

    exit(EXIT_SUCCESS);
}
```

# References


[1] K. M. R. Audenaert, J. Calsamiglia, R. Muñoz Tapia, E. Bagan, Ll. Masanes, A. Acin, and F. Verstraete. Discriminating states: The quantum Chernoff bound. *Phys. Rev. Lett.*, 98:160501, Apr 2007.

[2] Dave Bacon, Andrew M. Childs, and Wim van Dam. From optimal measurement to efficient quantum algorithms for the hidden subgroup problem over semidirect product groups. In *FOCS*, pages 469–478. IEEE Computer Society, 2005.

[3] Ethan Bernstein and Umesh Vazirani. Quantum complexity theory (preliminary abstract). In *Proceedings of the 25th Annual ACM Symposium on the Theory of Computing*, pages 11–20. ACM, 1993.

[4] Michael V. Berry. Waves and Thom's theorem. *Advances in Physics*, 25(1):1–26, 1976.

[5] Norman Bleistein and Richard A. Handelsman. *Asymptotic Expansions of Integrals*. Holt, Rinehart, and Winston, 1975.

[6] J. Calsamiglia, R. Muñoz Tapia, Ll. Masanes, A. Acin, and E. Bagan. Quantum chernoff bound as a measure of distinguishability between density matrices: Application to qubit and gaussian states. *Phys. Rev. A*, 77:032311, Mar 2008.

[7] Andrew M. Childs, Leonard J. Schulman, and Umesh. V. Vazirani. Quantum algorithms for hidden nonlinear structures. *Proc. 48th IEEE Symposium on Foundations of Computer Science*, 48:395–404, 2007.

[8] Andrew M. Childs and Wim van Dam. Quantum algorithms for algebraic problems. *Rev. Mod. Phys.*, 82:1–52, Jan 2010.

[9] John Coulson and G. G. Becknell. An extension of the principle of the diffraction evolute and some of its structural detail. *Physical Review*, 20(6):607–612, 1922.

[10] John Coulson and G. G. Becknell. Reciprocal diffraction relations between circular and elliptical plates. *Physical Review*, 20(6):594–600, 1922.

[11] Thomas Decker, Gbor Ivanyos, Miklos Santha, and Pawel Wocjan. Hidden symmetry subgroup problems. 1107.2189.

[12] A. Denney, C. Moore, and A. Russell. Finding conjugate stabilizer subgroups in PSL(2; q) and related groups. *Quantum Information and Computation*, 10:0282–0291, 2010, 0809.2445.

[13] John D. Dixon and Brian Mortimer. *Permutation Groups*. Number 163 in Graduate Texts in Mathematics. Springer-Verlag, New York, 1996.

[14] Arthur Erdélyi. *Asymptotic Expansions*. Dover, New York, 1955.

[15] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum algorithm for the Hamiltonian NAND tree. *Theory of Computing*, 4(1):169–190, 2008.

[16] Edward Farhi and Sam Gutmann. Analog analogue of a digital quantum computation. *Phys. Rev. A*, 57(4):2403–2406, Apr 1998.

[17] Katalin Friedl, Gábor Ivanyos, Frédéric Magniez, Miklos Santha, and Pranab Sen. Hidden translation and orbit coset in quantum computing. In *STOC*, pages 1–9. ACM, 2003.

[18] William Fulton and Joe Harris. *Representation Theory: a First Course*. Number 129 in Graduate Texts in Mathematics. Springer, New York, 2004.

[19] Lov K. Grover. A fast quantum mechanical algorithm for database search. *Proc. 28th ACM Symposium on the Theory of Computing*, 28:212–219, 1996.

[20] Sean Hallgren, Cristopher Moore, Martin Rötteler, Alexander Russell, and Pranab Sen. Limitations of quantum coset states for graph isomorphism. In *Proceedings of the 38th Annual ACM Symposium on Theory of Computing*, pages 604–617, 2006.

[21] Eugene Hecht. *Optics*. Addison Wesley, San Francisco, 2002.

[22] G. Ivanyos. Finding hidden Borel subgroups of the general linear group. *Quantum Information and Computation*, 12:0661–0669, 2012, 1105.4416.

[23] Gábor Ivanyos, Luc Sanselme, and Miklos Santha. An efficient quantum algorithm for the hidden subgroup problem in extraspecial groups. In Wolfgang Thomas and Pascal Weil, editors, *STACS*, volume 4393 of *Lecture Notes in Computer Science*, pages 586–597. Springer, 2007.

[24] Gábor Ivanyos, Luc Sanselme, and Miklos Santha. An efficient quantum algorithm for the hidden subgroup problem in nil-2 groups. In Eduardo Sany Laber, Claudson F. Bornstein, Loana Tito Nogueira, and Luerbio Faria, editors, *LATIN*, volume 4957 of *Lecture Notes in Computer Science*, pages 759–771. Springer, 2008.

[25] John David Jackson. *Classical Electromagnetics*. John Wiley & Sons, Inc., New York, 1999.

[26] Greg Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM J. Comput.*, 35(1):170–188, 2005, quant-ph/0302112.

[27] John D. Lafferty and Daniel Rockmore. Fast Fourier analysis for $sl_2$ over a finite field and related numerical experiments. *Experimental Mathematics*, 1(2):115–139, 1992.

[28] Rudolf Lidl and Harald Niederreiter. *Finite Fields*. Cambridge University Press, 1997. Number 20 in Encyclopedia of Mathematics and its Applications.

[29] Y.-K. Liu. Quantum algorithms using the curvelet transform. *Proc. ACM Symposium on Theory of Computing*, pages 391–400, 2009.

[30] Makoto Matsumoto and Takuji Nishimura. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Trans. Model. Comput. Simul.*, 8(1):3–30, January 1998.

[31] Cristopher Moore and Stephan Mertens. *The Nature of Computation*. Oxford University Press, 2011.

[32] Cristopher Moore, Daniel Rockmore, and Alexander Russell. Generic quantum Fourier transforms. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 778–787, 2004.

[33] Cristopher Moore, Daniel Rockmore, Alexander Russell, and Leonard J. Schulman. The power of basis selection in Fourier sampling: Hidden subgroup problems in affine groups. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1113–1122, 2004.

[34] Cristopher Moore, Alexander Russell, and Leonard Schulman. The symmetric group defies Fourier sampling. In *Proceedings of the 46th Symposium on Foundations of Computer Science*, pages 479–488, 2005.

[35] Cristopher Moore, Alexander Russell, and Piotr Śniady. On the impossibility of a quantum sieve algorithm for graph isomorphism: unconditional results. In *Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, pages 536–545, 2006.