Notation

$N :=$ number of points sampled

   $N \sim h^{-2}$

   $B :=$ number of basis functions used in vergini method

   $k :=$ wavenumber

   $E = k^2 :=$ energy of eigenfunction

   $h :=$ orthogonal spacing between sampled points

   $\alpha := kh$

   $\Omega \in \mathbb{R}^2$ compact domain

   $\Gamma = \partial\Omega$ domain boundary

Chapter 1

Introduction

1.1 Motivation

Nodal domains characterize regions of a vibrational surface (e.g. a drum head) that move together. The boundaries between nodal domains, known as nodal line are the regions which do not vibrate at all (Fig. 1.1). Understanding the characteristics of nodal domains has potential applications in musical instruments, mechanical engineering, geophysics, astrophysics, and many other areas dealing with wave behavior [8].

The nodal domains studied herein are formulated in terms of quantum mechanical wave functions on Euclidean billiards, a canonical example of quantum chaos. Quantum chaos lies at the intersection of quantum mechanics and chaos theory. The primary signature of chaos in classical systems is a nonlinear (exponential) divergence of paths in the phase space of a system. Quantum mechanics however, is entirely linear and quantum chaos deals with energy eigenfunctions of systems, which are constant in time. Thus chaos in quantum systems is manifest in different ways, one of the most studied being wavefunctions in chaotic domains. Figure 1.2 shows a classical orbit and a quantum eigenfunction on the same billiard.

The goal of this project is to numerically test certain conjectures regarding properties of nodal domains in quantum chaotic eigenfunctions in the high energy, or semiclassical, limit. Efficiently testing these conjectures requires solving new computational challenges. We hope these results will enable further mathematical investigation of eigenfunctions and that the tools developed herein may be applied to related problems in experimental mathematics.

1.2 Classical Chaos in Billiards

A billiard is a compact domain $\Omega \subset \mathbb{R}^2$. A billiard defines a map $f(\mathbf{x}, t)$: $\Omega \times S^1 \times \mathbb{R} \to \Omega \times S^1$ where S^1 is the unit circle and $\mathbf{x} = (q, p) \in \Omega \times S^1$ is a point in phase space where $q \in \Omega$ is a position and $p \in S^1$ is a momentum, or
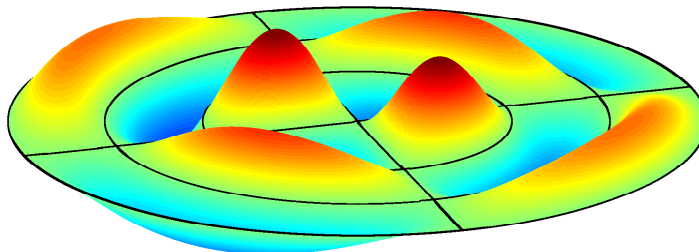
Figure 1.1: A circular vibrational surface with nodal lines shown in black. Nodal domains are regions between black lines

direction of motion. This map describes the motion of a ball bouncing in the domain Ω.

Chaos in classical systems is characterized by the Lyapunov exponent λ of a system, which describes how quickly nearby trajectories diverge. It is computed as the long time ratio of the divergence of two initially close paths:

$$\lambda = \lim_{t \to \infty} \lim_{|\boldsymbol{\epsilon}| \to 0} \frac{1}{t} \frac{|f(\mathbf{x_0}, t) - f(\mathbf{x_0} + \boldsymbol{\epsilon}, t)|}{|\boldsymbol{\epsilon}|}$$

From this definition it follows that

$$|f(\mathbf{x_0}, t) - f(\mathbf{x_0} + \boldsymbol{\epsilon}, t)| \approx e^{\lambda t} |\boldsymbol{\epsilon}|$$

for small $\boldsymbol{\epsilon}$ and large t. Thus a tiny change $\boldsymbol{\epsilon}$ in initial conditions produces a change that grows exponentially over time with growth rate λ. Chaotic systems have a positive Lyapunov exponent and this fact makes it difficult to predict long-term behavior because arbitrarily small errors in measurements of initial conditions eventually become large. As these errors grow to the size of the domain, the position of approaches a uniform distribution over the entire billiard. The property that small subsets of Ω eventually map to all of Ω is known as ergodicity (see appendix A for a formal definition).

1.3   Quantum Chaos in Billiards

A quantum wave-particle in a billiard Ω obeys the Schrödinger equation

$$Eu(\mathbf{r}) = -\frac{\hbar^2}{2m} \Delta u(\mathbf{r}) + V(\mathbf{r})u(\mathbf{r})$$
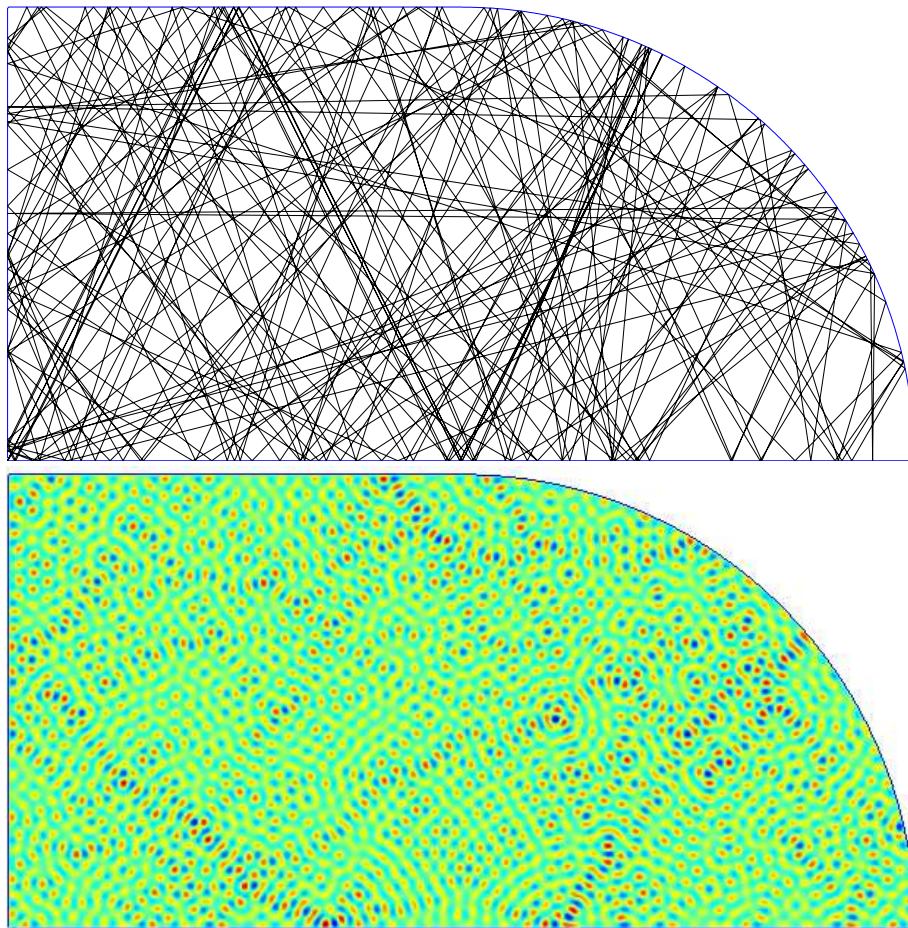
Figure 1.2: Top: A classical orbit in a billiard; bottom: a quantum eigenfunction in the same billiard
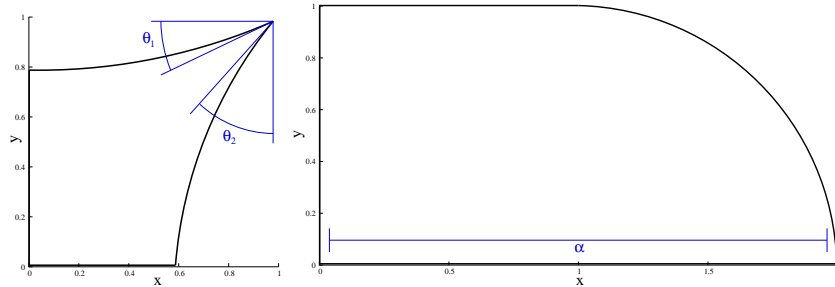
Figure 1.3: Left: quarter generalized rectangular Sinai billiard; right: quarter stadium billiard

Where $\Delta = \nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$ is the Laplacian differential operator in two dimensions. Setting $\hbar = 2m = 1$ and $V(\mathbf{r}) = 0$ for $\mathbf{r} \in \Omega$ while enforcing Dirichlet boundary conditions $u(\mathbf{r}) = 0$ for $\mathbf{r} \in \Gamma = \partial\Omega$ simplifies this to the Helmholtz equation

$$\begin{cases} (\Delta + k^2)u(\mathbf{r}) = 0 & \text{if } \mathbf{r} \in \Omega \\ u(\mathbf{r}) = 0 & \text{if } \mathbf{r} \in \Gamma \end{cases} \tag{1.1}$$

where $k^2 = E$ is the energy of the eigenfunction $u(\mathbf{r})$ and corresponds to the kinetic energy of the quantum wave-particle.

We focus our investigation on two billard shapes: the generalized rectangular Sinai billiard and the Stadium billiard. In both cases we desymmetrize the billard shapes by considering only a quarter of the full shape. This restricts our basis set to functions that are odd as a function of x and y, i.e., $f(-x, y) = f(x, -y) = -f(x, y)$. The desymmetrized billiards are shown in figure 1.3

The Sinai billiard is constructed from circular arcs that meet at $(1, 1)$ and is parameterized by two angles, θ_1 and θ_2, the angles from horizonal and vertical, respectively, of the arcs at $(1, 1)$. The Sinai billiard is said to demonstrate "hard chaos" becase there are no stable orbits.

The stadium billiard is construced from a rectangular region and a quarter circle and is parameterized by the the horizontal length of the billiard α. The stadium billiard contians neutrally stable orbits, specifically those with vertical momentum in the rectangular region. Neutrally stable orbits have zero Lyapunov exponent but any perturbation will cause them to have positive Lyapunov exponenet. The existence of such orbits implies that the stadium does not demonstrate hard chaos but because these orbits occupy a measure zero subset of phase space, the stadium still demonstrates chaotic properties.

1.4   Percolation Model

Bogomolny and Schmit [4] have argued that nodal domains of random functions (which are considered an accurate proxy for eigenfunctions of chaotic systems)
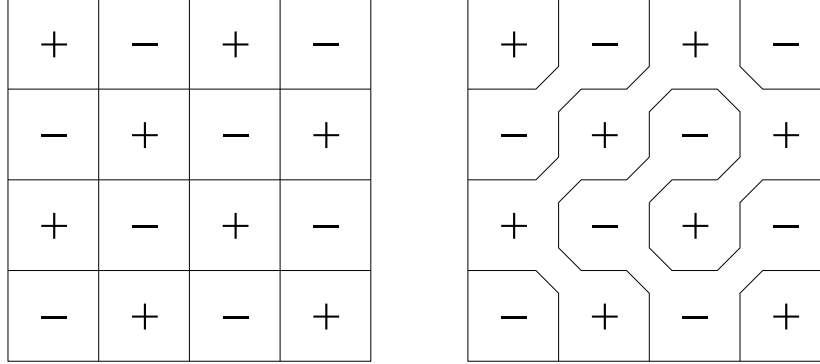
Figure 1.4: Left: checkerboard pattern; right: perturbed checkerboard pattern

can be modeled by nodal domains of a percolation model. Their percolation model is formed by creating a checkerboard of positive and negative regions with a grid size given by the average spacing of zeros of random functions along a particular axis. This checkerboard pattern can be realized as an eigenfunction $\bar{u}(x,y) = sin(\frac{kx}{\sqrt{2}})sin(\frac{ky}{\sqrt{2}})$ of a square billiard $\Omega = [0,1]^2$. A random eigenfunction can be modelled as this mean eigenfunction $\bar{u}(x,y)$ plus another term representing deviation from the mean $u(x,y) = \bar{u}(x,y) + \delta u(x,y)$. The deviation term $\delta u(x,y)$ is modelled by perturbing each nodal line crossing by connecting two diagonal regions (Fig. 1.4). The decision of which nodal domains to connect is made randomly with equal probability for either possibility.

Bogomolny and Schmit apply results from graph theory and statistical physics to compute the distribution of nodal domains in this percolation model. These results form the two primary conjectures we wish to test numerically.

Conjecture 1 (Mean of Nodal Domain Count)**.** *The number of nodal domains $\nu(E)$ in a quantum chaotic eigenfunction with energy E is normally distributed with mean*

$$\frac{\bar{\nu}(E)}{\bar{N}(E)} = \frac{3\sqrt{3} - 5}{\pi} \approx 0.0624 \tag{1.2}$$

Conjecture 2 (Variance of Nodal Domain Count)**.** *The number of nodal domains $\nu(E)$ in a quantum chaotic eigenfunction with energy E is normally distributed with variance*

$$\frac{\sigma^2(\nu(E))}{\bar{N}(E)} = \frac{18}{\pi^2} + \frac{4\sqrt{3}}{\pi} - \frac{25}{2\pi} \approx 0.0502 \tag{1.3}$$

In both conjectures, $\bar{N}(E)$ is the mean number of eigenfunctions with energy less than E which is given by Weyl's law [5] to be

$$\bar{N}(E) = \frac{|\Omega|E}{4\pi}$$

Bogolmony and Schmit also obtain a prediction for the distribution of areas of nodal domains

Conjecture 3 (Area of Nodal Domains). *The area s of nodal domains in a quantum chaotic eigenfunction follows the distribution*

$$f(s) \propto s^{-\tau} \tag{1.4}$$

where $\tau = \frac{187}{91}$

Chapter 2

Methods

2.1 Scaling Method

2.1.1 Theory

Vergini and Saraceno [7] developed a method of computing high energy eigenfunctions of chaotic billiards using a scaling method. The method simultaneously finds all eigenfunctions $\phi_i(\mathbf{r})$ with energy in a given window $[E - \Delta E, E + \Delta E]$ for some $E = k^2$ by scaling each eigenfunction. The scaled eigenfunctions $\chi_i(k, \mathbf{r})$ are computed as

$$\chi_i(k, \mathbf{r}) = \phi_i\left(\frac{k}{k_i}\mathbf{r}\right) = \phi_i\left(\mathbf{r} + \frac{\omega_i}{k_i}\mathbf{r}\right)$$

where $\omega_i = k - k_i$. This scaling causes all eigenfunctions to fall approximately in a linear subspace of a single basis set $\{\xi_l\}_{l=1}^{B}$, that is

$$\chi_i(k, \mathbf{r}) = \sum_{l=1}^{B} X_{li}\xi_l(k, \mathbf{r}) + \epsilon_i(\mathbf{r})$$

where $\epsilon_i(\mathbf{r}) \ll 1$ for sufficiently large B, the number of basis functions used. Values of B of approximately $1.5\frac{k|\Gamma|}{\pi}$ have been shown to produce $\epsilon < 10^{-4}$. This single basis set provides a significant efficiency gain over prior methods because many eigenfunctions can be found by solving a single linear system.

The choice of basis functions $\xi_l(k, \mathbf{r})$ depend on the billiard shape being used. For the quarter generalized rectangular Sinai billiard the basis set consists of irregular Bessel functions (which satisfy $(\Delta + k^2)\xi_l(k, \mathbf{r}) = 0$) placed along Γ^+, the set of points ouside Ω whose nearest distance to Γ is D where kD is taken to be 7 so that D is apprximately one wavelength. For the quarter stadium, the basis set is plane waves with orientations evently spaced around the circle. The scaling method runs in $O(B^3) = O(k^3)$ time. [2]

Billiard	Ratio	k_c
Sinai	75	3860
Stadium	31	550

Table 2.1: Ratios of basis function evaluation to coefficient multiplication for various billiards. Column k_c contains the value of k for which n is equal to the given ratio with $\Delta k = 0.1$ and coefficient multiplications take as long as evaluating basis functions

2.1.2 Run Time

The scaling method only computes coefficients of basis functions, which must be evaluated in order to obtain eigenfunction values at arbitrary $\mathbf{r} \in \Omega$. However, because all eigenfunctions in the energy window use the same basis functions, each basis function only needs to be evaluated once at each point, regardless of the number n of eigenfunctions in the energy window. Thus NB basis function evaluations are required. Computing eigenfunctions is then just a matter of multiplying the basis function values at each point by the coefficients for that eigenfunction, which requires a total of nNB multiplications. Basis evaluation are much more expensive than multiplications however. Table 2.1 summarizes the ratios of these costs.

Errors in eigenfunctions produced by the scaling method scale like $O(\Delta E^3)$ [3] and are independent of E so we use a constant ΔE. We estimate errors by calculating the tension

$$t = \left(\int_\Gamma u(\mathbf{r})^2 d\mathbf{r} \right)^{\frac{1}{2}}$$

ΔE is chosen such that $t \lesssim 10^{-4}$. Using Weyl's law we can obtain an estimate of the number of eigenfunctions in an energy window to be

$$n \approx \frac{|\Omega|}{4\pi}((k + \Delta k)^2 - (k - \Delta k)^2) = \frac{|\Omega|}{\pi} k \Delta k$$

where $k = \sqrt{E}$ and $\Delta k = \sqrt{\Delta E}$.

We choose ΔE to be as large as possible while keeping errors small in order to maximize n, allowing more eigenfunctions to be computed from each evaluation of basis functions.

For both billiards considered here $B \sim k$. Thus, in practice $N \gg n$ and $N \gg B$ so it is primarily N (which scales like α^{-2}) that determines the time to compute eigenfunctions. Computation time is dominated by eigenfunction evaluation, which is dominated by basis function evaluations Additionally, we find that the time to evaluate eigenfunctions dominates total computation time (Fig. 2.1).
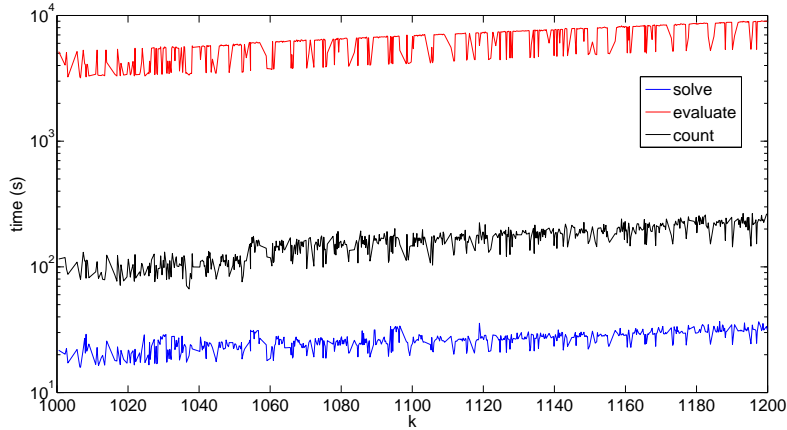
Figure 2.1: Comparison of run times of solving for basis coefficients, evaluating eigenfunctions, and counting nodal domains. Evaluating eigenfunctions dominates runtime, taking approximately 97% of computation time.

2.2 Counting Nodal Domains

Eigenfunctions are sampled on a regular grid; each point in this grid will be referred to as a "pixel." Nodal domains are counted by exploring domains pixel-by-pixel, marking each pixel as "counted" once it has been seen. The searching algorithm usded to explore each domain is a hybrid depth- and breadth-first method where for each pixel, the sign of each neighboring pixel is compared to the sign of the nodal domain and if the sign matches, the neighboring pixel is pushed onto a stack. Exploration then continues by popping a pixel off of the stack. This method was chosen so that the signs of all four neighbors (above, below, right, and left) are known when that pixel is being examined, which is necessary for the adaptive interpolation scheme described below.

10

Algorithm 1 Nodal domain counting algorithm

Require: *grid* is *ny* by *nx* matrix containing eigenfunction values on billiard
Require: $\alpha = kh$, M is highest order Bessel function to interpolate with, ρ is ratio to interpolate by

    function CountNodalDomains($grid, \alpha, M, \rho$)
        for $i \in 1 \ldots ny$ **do**
            for $j \in 1 \ldots nx$ **do**
                $counted[i][j] \leftarrow UNCOUNTED$
            end for
        end for
        $interp \leftarrow CreateInterpMatrix(\alpha, M, \rho)$
        $i, j, domains \leftarrow 0$
        while $i, j \leftarrow FindNextUnseen(counted, i, j)$ **do**
            $domains \leftarrow domains + 1$
            $FindDomain(grid, counted, i, j, domain\_num, interp)$
        end whilereturn *domains*
    end function

    function FindNextUnseen($counted, y, x$)
        for $i \in y \ldots ny$ **do**
            for $j \in 1 \ldots nx$ **do**
                if $i = y$ and $j \leq x$ **then**
                    continue
                end if
                if $counted[i][j] = UNCOUNTED$ **then**
                    return i, j
                end if
            end for
        end for
        return $NULL$
    end function

Algorithm 1 Nodal domain counting algorithm (continued)
---
    function FINDDOMAIN($grid, counted, i, j, interp$)

        $s.push(j, i)$ ▷ s is a stack

        while $x, y \leftarrow s.pop()$ **do**

            $sign \leftarrow sign(grid[y][x])$

            if $InGrid(x - 1, y)$ **then**

                if $sign(grid(x - 1, y)) = sign$ **then**

                    $left \leftarrow TRUE$

                    if $counted[x - 1][y] = UNCOUNTED$ **then**

                        $s.push(x - 1, y)$

                    end if

                end if

            end if

            . . .   ▷ Same for *above*, *right*, and *below*

            if $InGrid(x - 1, y - 1)$ and *above* and *left* **then**

                if $sign(grid(x - 1, y - 1)) = sign$ **then**

                    if not $IsInterpolated(counted, x - 1, y - 1)$ **then**

                        $Interpolate(grid, counted, x - 1, y - 1, interp)$

                    end if

                    if $ConnectedAboveLeft(counted, x, y)$ **then** $s.push(x - 1, y - 1)$

                    end if

                end if

            end if

            . . .   ▷ Same for *below* and *left*, *below* and *right*, and *above* and *right*

            $counted[y][x] = COUNTED$

        end while

    end function

Letting N be the number of points the eigenfunction is sampled at, this method has computational complexity $O(N)$. This is because the method performs a fixed number of comparisons for each pixel plus $O(N)$ total comparisons searching for an unseen nodal domain after a nodal domain has been explored. This algorithm uses an array of N integers to store, for each pixel, whether it has been counted, which nodal domain it is in, whether or not it is within the boundary of the billiard, and additional information relating to the interpolation method described below. In addition, this method uses a dynamically sized array as a stack whose size is (loosely) bounded above by the number of pixels in the nodal domain being explored.

2.3   Interpolation

Because sampling eigenfunctions is expensive (requiring evaluation of B = O(k) basis functions), we are limited by the total number of pixels N, which scales like h^{-2} where h is the distance between adjacent sample points, or the width and
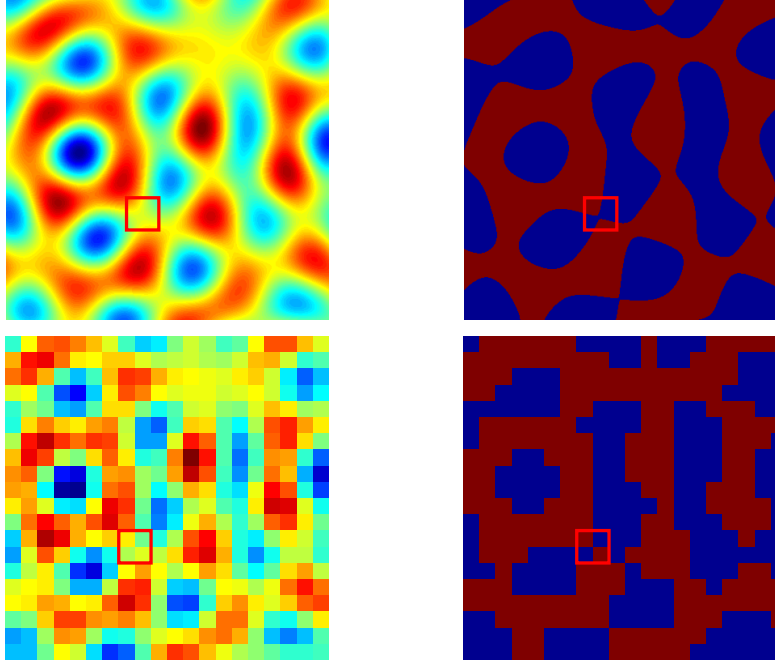
Figure 2.2: An ambiguity in nodal domain connectivity due to coarse sampling. The ambiguous region is shown with a red box. Top left: high resolution image of an eigenfunction; top right: nodal domains of the same eigenfunction; bottom left: low resolution image of the same eigenfunction; bottom right: nodal domains of low resolution eigenfunction

height of a pixel. As a consequence, we must work with relatively coarsely sampled eigenfunctions, causing us to encounter scenarios where the connectivity of nodal domains is ambiguous (Fig. 2.2).

We resolve such ambiguities by performing an interpolation in the ambiguous region. We interpolate with the functions

$$J_n(kr)\sin(n\theta)$$

and

$$J_n(kr)\cos(n\theta)$$

where $J_n(r)$ is a regular Bessel function. These functions form a complete orthonormal basis for solutions of (1.1) (see appendix B). We fix a value M to be the order of the highest order Bessel function, restricting our basis to the

13

finite set

$$\xi_i(r, \theta) = \begin{cases} J_i(kr) & \text{if } i = 0 \\ J_i(kr)\sin(i\theta) & \text{if } 1 \le i \le M \\ J_{i-M}(kr)\cos((i-M)\theta) & \text{if } M+1 \le i \le 2M \end{cases} \tag{2.1}$$

We construct a surrogate function

$$\tilde{u}(\mathbf{r}) = \sum_{i=0}^{2M} c_i \xi_i(\mathbf{r})$$

as a local approximation to the eigenfunction $u(\mathbf{r})$ by fitting the coeffiecients $c_i \in \mathbb{R}$ to minimize the error $\|u - \tilde{u}\|_2$. This function can then be sampled at a higher resolution within the region in question. We define the sampling ratio of this surrogate function to the original eigenfunction to be ρ.

Interpolation occurs only between the central four pixels but uses surrounding values to fit the coefficients c_i. The selection of which surrounding values to use is termed a stencil. After testing various stencil shapes we selected one using 24 points consisting of a four by four square with two additional points on each side of the square. See C for a discussion of the choice of stencil, M, and h.

2.3.1 Numerical Implementation

We can construct a single matrix to transform values on a stencil to interpolated values in the center of the stencil, performing the process described above with a single matrix multiplication. We construct the interpolation matrix as

$$U = LH^+$$

Where L is a matrix which contains evaluations of Bessel functions at low resolution, H contains evalutions of Bessel functions at high resolution and $^+$ denotes the pseudoinverse. Specifically,

$$L_{ij} = \begin{cases} J_j(r_i) & \text{for } j = 0 \text{ and } 0 \le i < T \\ J_j(r_i)\sin(j\theta_i) & \text{for } 1 \le j \le M \text{ and } 0 \le i < T \\ J_{j-M}(r_i)\cos((j-M)\theta_i) & \text{for } M+1 \le j \le 2M+1 \text{ and } 0 \le i < T \end{cases}$$

and

$$U_{ij} = \begin{cases} J_j(r_i) & \text{for } j = 0 \text{ and } 0 \le i < \gamma \\ J_j(r_i)\sin(j\theta_i) & \text{for } 1 \le j \le M \text{ and } 0 \le i < \gamma \\ J_{j-M}(r_i)\cos((j-M)\theta_i) & \text{for } M+1 \le j \le 2M+1 \text{ and } 0 \le i < \gamma \end{cases}$$

where T is the number of points in the stencil, $\gamma = (\rho + 1)^2$ and (r_i, θ_i) are points in the stencil expressed in polar coordinates.
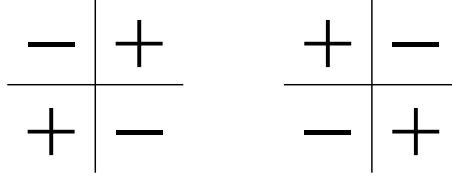
14

Figure 2.3: Configurations that will result in interpolation

The pseudoinverse H^+ is computed using a singular value decomposition as follows,

$$H^+ = V\Sigma^+ U^*$$

where $H = U^*\Sigma V$ is a singular value decompostion of H and

$$\Sigma_{ii}^+ = \begin{cases} \Sigma_{ii}^{-1} & \text{if } \Sigma_{ii} > \epsilon \\ \Sigma_{ii} & \text{otherwise} \end{cases}$$

where $\epsilon = \gamma\epsilon_{double}\Sigma_{11}$ where ϵ_{double} is the difference between 1 and the smallest IEEE double precision floating point number greater than 1. Singular values less than ϵ are considered to be zero within working precision.

2.3.2 Code Implemenation

A region is interpolated if and only if, when counting nodal domains, we encounter a point whose sign matches that of a point diagonally adjacent to it, but differs from the signs of the two points adjacent to both it and its diagonal neighbor (Fig. 2.3). In such a case we fill a vector $\mathbf{v}$ with the eigenfunction values at the stencil points surrounding the four points comprising the ambiguity. We then compute $\mathbf{w} = U\mathbf{v}$ where U is the interpolation matrix described above. This vector $\mathbf{w}$ contains estimated eigenfunction values with a spacing of $\frac{h}{\rho}$ between the four points comprising the ambigious region. We can use these values to determine the connectivity of the nodal domains by traversing pixel-by-pixel from the top-left pixel, in the same manner as above, until we either reach the bottom-right pixel or finish exploring the nodal domain. In the former case the nodal domain containing the top-left pixel connects to the nodal domain containing the bottom-right pixel and in the latter case the nodal domain containing the top-right pixel connects to the nodal domain containing the bottom-left pixel.

2.4 Code

Code implementing Vergini's scaling method to compute eigenfunctions was written by Alex Barnett. Code for counting nodal domains was written by the author. All code used herein is open source and available under the GPL

license [1] [6]. Appendix D contains an overview of the functions provided by
this library.

Chapter 3

Results

3.1 Data Collected

We computed approximately 100,000 eigenfunctions and counted half a billion nodal domains. This data was collected over several weeks running on the Dartmouth Mathematics Condor cluster.

3.2 Mean of number of nodal domains

We find the mean number of nodal domains to be in accordance with conjecture 1.2

3.3 Variance of number of nodal domains

3.4 Interpolation frequency

At $\alpha = 0.5$ interpolation occured on average 0.1512 times per nodal domain and $1.9e - 4$ times per pixel.

3.5 Nodal Domain Size

Billiard	Eigenfucntions	Nodal Domains
Sinai	3e4	1.5e8
Stadium	20	100

Table 3.1: Ratios of basis function evaluation to multiplication for various billiards. Column k_c contains the value of k for which n is equal to the given ratio and coefficient multiplications take as long as evaluating basis functions
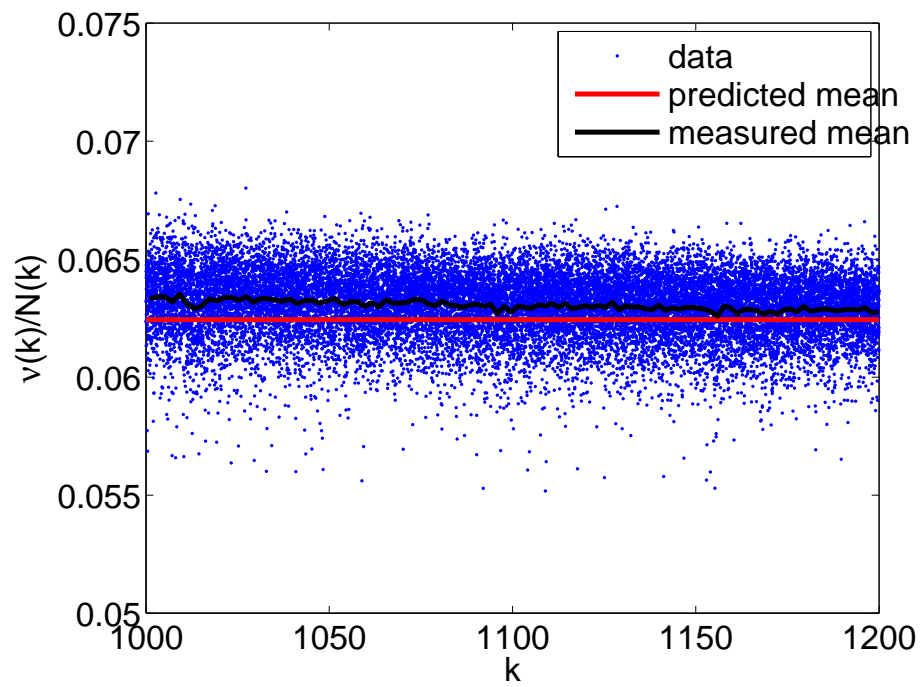
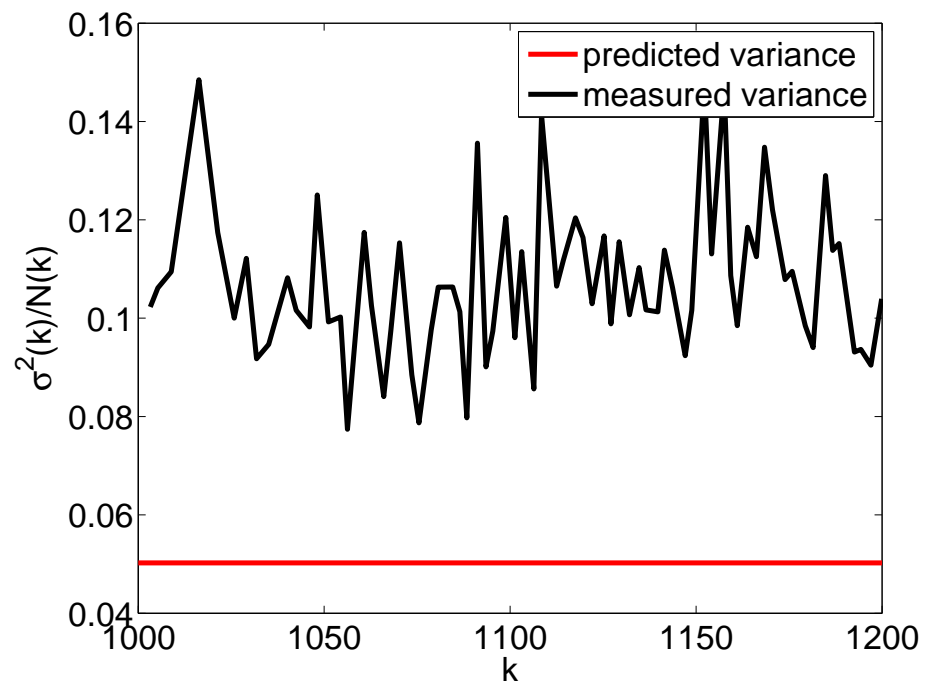Figure 3.1: Mean number of nodal domains
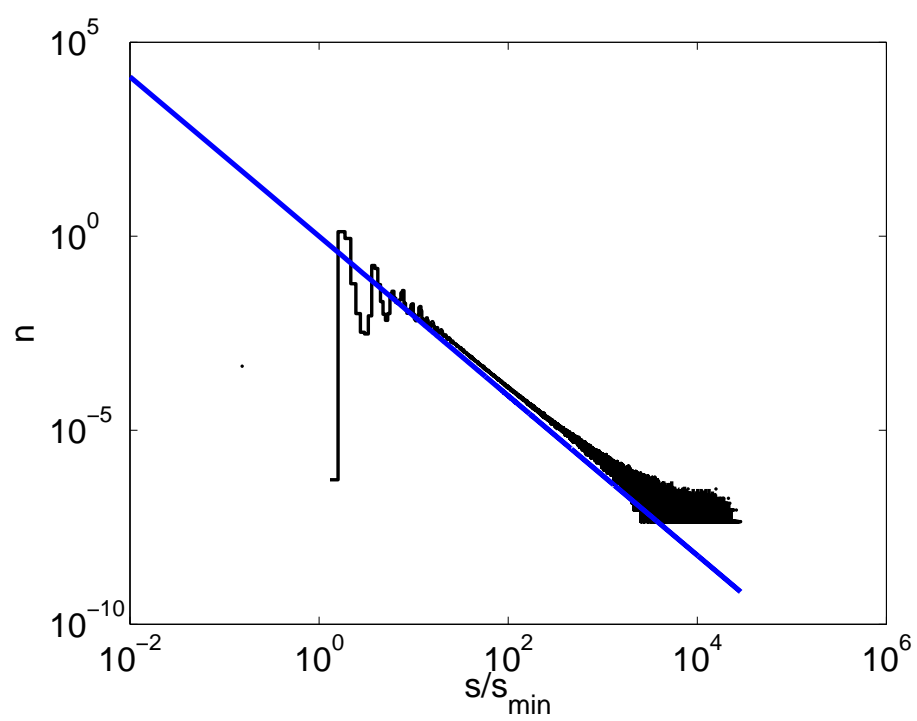
Figure 3.2: Variance number of nodal domains

Figure 3.3: Size of nodal domains

Appendix A

Definition of Ergodicity

Definition 1 (σ-algebra). $\Sigma \subset 2^\Omega$ *is a σ-algebra over Ω if:*

1. $\emptyset, \Omega \in \Sigma$

2. $\Omega \backslash A \in \Sigma \ \forall A \in \Sigma$

3. $\cup_{i=1}^\infty A_i \in \Sigma \ \forall \{A_i\}_{i=1}^\infty$ *such that $A_i \in \Sigma \forall i$*

Definition 2 (Measure). *μ: $\Sigma \to \mathbb{R}$ is a* measure *on Σ if:*

1. $\mu(A) \geq 0 \ \forall A \in \Sigma$

2. $\mu(\emptyset) = 0$

3.
$$\mu\left(\cup_{i=1}^\infty A_i\right) = \sum_{i=1}^\infty \mu(A_i)$$

Definition 3 (Measure space). *A* measure space *is a triple (Ω, Σ, μ) where Σ is a σ-algebra over Ω and μ is a measure on Σ.*

Definition 4 (Probability space). *A* probability space *is a measure space (Ω, Σ, μ) where $\mu(\Omega) = 1$.*

Definition 5 (Ergodicity). *Let (Ω, Σ, μ) be a probability space. A mapping T: $\Sigma \to \Sigma$ is* ergodic *if $T(E) = E \implies \mu(E) = 0$ or $\mu(E) = 1$*

Appendix B

General Solution of the Helmholtz Equation

Here we show that the functions $J_n(kr)\sin(n\theta)$ and $J_n(kr)\cos(n\theta)$ form a complete orthonormal basis of solutions of (1.1).

In polar coordinates,

$$\Delta = \frac{1}{r}\partial_r(r\partial_r) + \frac{1}{r^2}\partial_{\theta\theta}$$

Thus, (1.1) can be expressed as

$$u_{rr}(r,\theta) + \frac{1}{r}u_r(r,\theta) + \frac{1}{r^2}u_{\theta\theta}(r,\theta) + k^2 u(r,\theta) = 0$$

Using separation of variables we attempt solutions of the form

$$u(r,\theta) = R(r)\Theta(\theta)$$

where $\Theta(\theta)$ is periodic with period 2π. This gives

$$\Theta''(\theta) + n^2\Theta(\theta) = 0$$

and

$$r^2 R''(r) + r R'(r) + r^2 k^2 R(r) - n^2 R(r) = 0$$

The periodicity of $\Theta(\theta)$ requires that

$$\Theta(\theta) = \alpha\sin(n\theta) + \beta\cos(n\theta)$$

where $n \in \mathbb{Z}$. The radial differential equation is known as Bessel's equation and has solutions

$$R(r) = J_n(kr)$$

where $k \in \mathbb{R}$ is allowed to take discrete values determined by boundary conditions.

Thus, the general solution of (1.1) can be expressed as a sum of the form

$$b_0 J_0(kr) + \sum_{n=1}^{\infty} a_n J_n(kr) \sin n\theta + b_n J_n kr \cos n\theta \qquad \text{(B.1)}$$

Appendix C

Interpolation Parameter Selection

In this section, we discuss the various stencils, M, and α values that were considered and how we picked our particular values. Only stencil shapes that which are symmetric about the four central points were considered. The four shapes considered are shown in figure C.1. The most important consideration in choosing a stencil was the accuracy of the interpolation. The size of the stencil affects the computational cost of interpolation but this difference is trivial.

The accuracy of interpolation also depends on M and $\alpha = kh$, which must be considered simultaneously with the choice of stencil. Figure C.2 shows a comparison of the infinity norm of the interpolation error over various values of M and α for each of the stencils shown above.
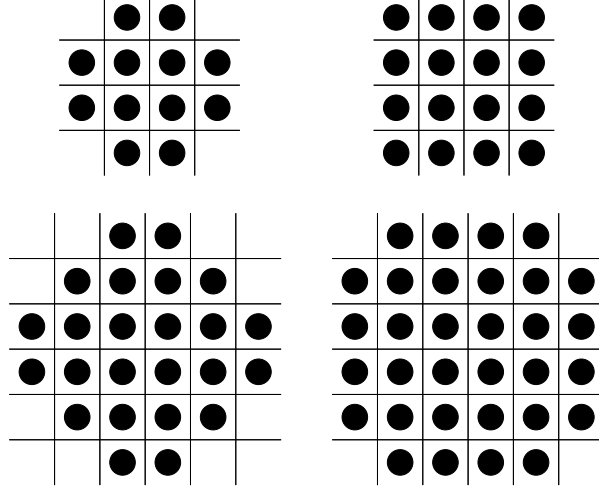
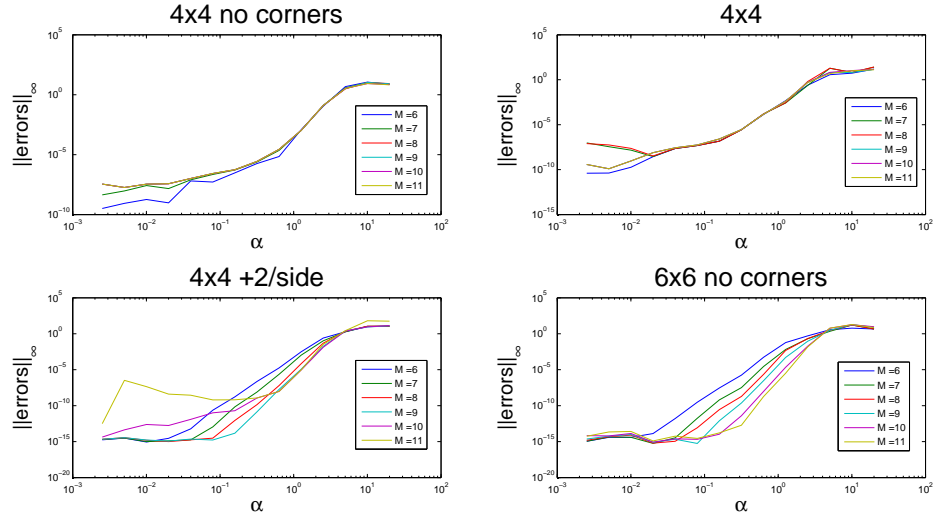Figure C.1: The four stencil shapes considered for interpolation

Figure C.2: Comparison of interpolation error norms over M and α for each stencil.

Appendix D

Code Interface

D.1 Command Line Interface

D.1.1 Verg

```
verg -l qust:2 -b 10 -s vepwoo:1.2:40:1.5 -k 200.01 -V 0.005 -f 0.001 -m
verg -l qugrs:1:0.4:0.7 -s oyooo:1.5:7:1 -u -4 1 -k 200.1 -V 0.005 -f 0.001 -m
```

D.1.2 Count

```
count -f t.sta_bin -m t.mask.sta_bin -l qust:2 -d .001 -k 200.01 -M 8 -u 20
```

D.1.3 Vc

```
vc -n run_2018.450000 -l qugrs:1.0:0.4:0.7 -s oyooo:1.5:7:1 -u -4 1 -k 2018.450000 -V 0.0500
```

D.1.4 Perc

```
perc -r 100:6:1000 -N 100
```

Appendix E

Code Notes

...

Bibliography

[1] Gnu general public license. `http://www.gnu.org/licenses/gpl.html`.

[2] A. Barnett. Asymptotic rate of quantum ergodicity in chaotic euclidean billiards. *Communications in Pure and Applied Mathematics*, 59(10):1457–1488, 2006.

[3] A. Barnett and A. Hassell. Fast computation of high frequency dirichlet eigenmodes via the spectral flow of the interior neumann-to-dirichlet map. 2012.

[4] E. Bogomolny and C. Schmit. Percolation model for nodal domains of chaotic wavefunctions. *Physical Review Letters*, 88(11), 2002.

[5] P. R. Garabedian. *Partial differential equations*. John Wiley & Sons Inc., New York, 1964.

[6] Kyle Konrad. qc. `https://github.com/ktkonrad/qc`.

[7] E. Vergini. Calculation by scaling of highly excited states of billiards. *Physical Review E*, (3):2204–2207, 1995.

[8] I. Wigman. On the nodal line of random and deterministic laplace eigenfunctions. 2011.