AN ITERATIVE ALGORITHM FOR COMPUTING THE MEASURES OF GENERALIZED VORONOI REGIONS∗

LISA J. LARSSON†, RUSTUM CHOKSI‡, AND JEAN-CHRISTOPHE NAVE§

Abstract. We present and analyze a fast algorithm for directly computing the measure of a generalized Voronoi region associated with generators of arbitrary co-dimension. The algorithm is based upon solving one Eikonal equation to generate a kernel-based operator whose iteration accumulates “mass” along the closest generator. In particular, the algorithm does not require the computation of the ridge set (Voronoi diagram) nor the gradient of the solution to the Eikonal equation. The algorithm is shown to be first order and converge very quickly. Several illustrations are presented including the computation of measures of influence associated with the Los Angeles County highway system. The method can also be used for the fast computation of the centroid and higher moments of the generalized Voronoi regions.

Key words. Generalized Voronoi Region, Measure of Influence, Eikonal equation, Iteration of a Markov Operator

AMS subject classifications.

1. Introduction. In this article, we present and analyze an algorithm for directly computing the measure of a generalized Voronoi region. Let $\Omega \subset \mathbb{R}^d$ be a bounded, convex domain with a non-negative density function $\mu \in L^1(\Omega)$ that integrates to one. Let $\Gamma_1, \ldots, \Gamma_n \subset \Omega$ be a fixed collection of subsets of $\Omega$ with co-dimension in $\{1, \ldots, d-1\}$. We call the sets $\Gamma_1, \ldots, \Gamma_n$ generators. Each generator has a corresponding generalized Voronoi region, which is simply the set of points in the domain that are closer to that generator than to any other generator. We allow the points that are equidistant to two or more generators to be included in all relevant regions. More precisely, we define the generalized Voronoi region, $V_i$, as follows:

$$V_i := \{ x \in \Omega \mid \text{dist}(x, \Gamma_i) \leq \text{dist}(x, \Gamma_j), \text{ for } j \in \{1, \ldots, n\} \setminus \{i\} \} ,$$

where

$$\text{dist}(x, \Gamma_i) := \min_{y \in \Gamma_i} |x - y| .$$

We define the ridge set associated with the generators to be the set of points which are equidistant to two or more generators, i.e.,

$$\bigcup_{i \neq j} (V_i \cap V_j) .$$

In this special case where $d = 2$, $\mu(x) \equiv 1$, and each generator is a singleton (point), the regions corresponds to the well-studied Voronoi cells which comprise a Voronoi

∗This work was supported by NSERC Canada under their Discovery Grant Program (RC, JC) and Discovery Accelerator Program (JC). LL was also supported by a Schulich Graduate Fellowship at McGill.
†Department of Mathematics and Statistics, McGill University, Montreal, Quebec, CA (lisa.powers@mail.mcgill.ca).
‡Department of Mathematics and Statistics, McGill University, Montreal, Quebec, CA (rchoksi@math.mcgill.ca).
§Department of Mathematics and Statistics, McGill University, Montreal, Quebec, CA (jcnave@math.mcgill.ca).
diagram or Voronoi tessellation. However, this definition extends beyond two dimensions, and is valid for sets \( \Gamma_i \) of co-dimension 1 through \( d - 1 \).

**Definition 1.1.** Given a density \( \mu \in L^1(\Omega) \) we define the **measure of influence** (or simply the **influence**), \( w_i \), corresponding to the \( i \)th generator as follows:

\[
w_i := \int_{V_i} \mu(x) \, dx.
\]  

In this article our goal is:

**Given** \( \Omega, \mu, \) and generators \( \Gamma_i \), **compute** \( w_i \) without finding the \( V_i \) explicitly.

Given the ubiquitous nature of the problem, there are many potential applications where the fast computation of such influences is relevant. One particular application comes from **geographic distributions and urban planning**. For example, given a collection of hospitals (points), a Voronoi diagram describes the geographic region closest to each of the hospitals. Similarly, given a collect of highways or subway lines (curves), a generalized Voronoi diagram describes the geographic region closest to each of the highways. The corresponding values of the influences describe the fraction of land or population that is serviced by a given hospital or highway (cf. Section 5.2).

**Voronoi diagrams** or (**Voronoi tessellations**) have been the subject of intense study in computational geometry ([14, 17, 10]). For a collection of points, the ridge set is simply the boundaries between the Voronoi regions and consists of straight lines. Efficient combinatorial algorithms for generating the Voronoi diagram (a Voronoi tessellation) are readily available via **Delaunay Triangulation** and the “dual” Voronoi diagram (see, for example, [14, 17, 10]). Thus the influence associated with each polygonal region can be computed efficiently. For non-point generators (e.g. curves), analogous algorithms for the Voronoi diagrams are far less developed and exist only for simple ansatzes for the curves (see [1, 3, 20] and the references therein, see also [12]). Indeed, the ridge sets associated with the generalized Voronoi diagram are far more complicated for curve generators. On the other hand, we could directly compute the Voronoi region boundaries via distance functions by solving certain Eikonal equations. For general collections of curves (or curves and points), to find these boundaries one must solve an Eikonal equation for each curve, and have these solutions stored in memory simultaneously in order to calculate the generalized Voronoi boundaries. Once these boundaries are obtained, one must still integrate a density function over these regions to obtain their measure. Whereas computational methods are very well developed for many aspects of this calculation (cf. [18, 22]), this direct approach turns out to be computationally inefficient. We will discuss this at length in the next section.

In this paper, we present an algorithm for computing the influences which by-passes the explicit calculation of the ridge sets. Rather, it relies on the iteration of a Markov kernel operator until the initial density is accumulated in a neighborhood of the generators. This transforms the problem completely: instead of integrating a density over a priori unknown regions, we evolve the initial density to be able to integrate over known regions. Our algorithm

- solves an Eikonal equation (c.f. (3.2)) once;
- uses the Eikonal solution to construct a Markov kernel operator (c.f. (3.1), (3.7)) whose application to an initial density moves “mass” towards the closest
generator;

• iterates the operator until all mass from the initial density is accumulated in a fixed neighborhood of the generators;
• computes \( w_i \) by integrating around this neighborhood.

We remark that the Eikonal solution \( \phi \) essentially gives a distance function associated to the closest generator. Given that the spirit of the algorithm is to move mass towards the closest generator, one can naturally see the analogue with a gradient flow associated with \( \phi \). However, we stress that at no point do we need the explicit calculation of \( \nabla \phi \). The discontinuity set of \( \nabla \phi \) is not a priori known and is in fact larger than the ridge set. Figure 1.1 illustrates this point as the set of discontinuities of \( \nabla \phi \) includes points where the closest distance is attained by more than one point on the same generator.

![Image of ridge set complexity](Fig. 1.1. Complexity of the ridge set (in dashed lines) for generators consisting of three curves (in bold) and two points. Taken from Alt, Cheong, & Vigneron [1].)

Thus the main novelties of our algorithm are

• it applies in any space dimension and to any finite collection of generators consisting of sets of arbitrary co-dimension;
• it requires one solution \( \phi \) of an Eikonal equation associated with all the generators, but never requires the computation of the ridge set (the generalized Voronoi diagram), nor the gradient \( \nabla \phi \).
• The method applies not only to compute influences but to compute the integral of any integrable function over \( V_i \). In particular, one can compute any moment of \( \mu \) over \( V_i \) such as the centroid or moment of inertia of the generalized Voronoi region (see Section 6 for a brief discussion of Centroidal Voronoi Tessellation and possible extensions of our method).

The paper is organized as follows: We first discuss the direct approach to this problem, that is, computing the ridge sets explicitly, and then integrating over the obtained regions. In particular, we write out an explicit algorithm to calculate these boundaries, and describe exactly where the computational inefficiencies arise. Next, we introduce our indirect approach to calculating these measures, and describe how the solution approximates the true (generalized Voronoi) measures of influence. We then introduce the computational algorithm and provide error estimates. We end with several numerical examples.
2. The Direct Approach. We first motivate our approximation algorithm by explaining the inefficiencies of a more direct approach. To calculate the influences directly, the boundaries of the generalized Voronoi regions are calculated, and then the initial density is integrated over each of these regions to obtain the influence of each generator.

2.1. Computing the Boundaries. The boundaries of the generalized Voronoi regions, \( \partial V_i \), are parts of the ridge set. There are only very special cases where the ridge set are available explicitly\(^1\). In most cases, however, the location of the ridges must be computed numerically. To compute the ridge set, we first obtain a distance function, \( \phi_i \), corresponding to each generator \( \Gamma_i \), \( i = 1, \ldots, n \). Using these distance functions, we present an algorithm below (Algorithm 1) to find \( n \) level set functions \( \phi_{\partial V_i}, i = 1, \ldots, n \), whose zero contours are the ridge set. For each \( i \in \{1, \ldots, n\} \), the distance functions, \( \phi_i \), can be obtained by solving the Eikonal equation with \( \Gamma_i \) as the initial contour.

### Algorithm 1 Calculate Boundary Level Set Functions \( \phi_{\partial V_i}, i = 1, \ldots, n \).

```
for i = 1 : n do
    Solve \(|\nabla \phi_i| = 1\) in \( \Omega \),
    such that \( \phi_i(\Gamma_i) = 0 \).
end for
for i = 1 : n do
    for j = i + 1 : n do
        \( \phi_{ij} \leftarrow \phi_i - \phi_j \)
        \( \phi_{ji} \leftarrow -\phi_{ij} \)
    end for
end for
for i = 1 : n do
    \( \phi_{\partial V_i} \leftarrow \min_{j \neq i \in \{1, \ldots, n\}} \phi_{ji} \);
end for
```

To identify the generalized Voronoi boundaries, simply find where \( \phi_{\partial V_i} = 0 \) for \( i = 1, \ldots, n \).

2.1.1. How it works. Let us look a little closer at this algorithm. For any fixed generator, \( \Gamma_i \), the zeros of the functions \( \phi_{ji} \) (where \( \phi_{ji} = \phi_i - \phi_j \), \( j(\neq i) \in \{1, \ldots, n\} \)) are all candidates for the ridge sets. These functions are zero at any point that is equidistant to both \( \Gamma_i \) and \( \Gamma_j \). For each \( i \), the algorithm picks the true solution by computing a minimum among all candidates, \( \phi_{ji}, j \in \{1, \ldots, n\} \setminus \{i\} \). A one-dimensional example of this algorithm is presented in Figure 2.1. For \( i = 1 \), the candidates for the ridge sets are \( \phi_{21} \) and \( \phi_{31} \). By taking the minimum of these two functions, we calculate the ridge to be at \( x = 0.3 \), which is the correct solution. The same characterization is true in higher dimensions.

2.1.2. Computational Complexity. There are very few generators that admit an analytical distance function. One example is circular generators: the distance function is simply a cone. For more general generators, it is necessary to solve the Eikonal equation, using the generator as the initial contour. For each generator, one distance function is required. Assume there are \( M \) grid points in each of \( d \) dimensions.

---

\(^1\)See Appendix A for the equation of the ridge set in the case of circular generators.
That results in \( n \) computations of \( \mathcal{O}(M^d) \) complexity (using a Fast Sweeping Method [23], for example). To compute Algorithm 1 quickly, all distance functions must be retained in memory. Further, \( \mathcal{O}(n^2) \) difference functions \( \phi_{ij} \) must be calculated and retained in memory. For each \( i \), all distance functions are required to calculate \( \phi_{ij} \) for every \( j \). Assume each array element is double-precision. By the end of the first for loop, there are

\[
M^d \cdot \frac{n^2 + n}{2} \cdot 8 \text{ bytes}
\]

in memory. For \( d = 2 \) with \( M = 200 \) and \( n = 100 \), this takes 1.6 GB of memory. For \( M = 300 \), and \( n = 100 \), this takes 3.6 GB of memory. Jumping to \( d = 3 \), for \( M = 200 \) and \( n = 100 \), this takes 323 GB of memory. For \( M = 300 \), and \( n = 100 \), this takes 1.09 TB of memory. Performing these calculations by input/output of the various level set functions may be feasible in some cases memory-wise, but this would be extremely slow.

**2.2. Integration.** The calculation is not complete until we integrate the density function over the generalized Voronoi regions to obtain influences. First find the points where \( \phi \partial V_i = 0 \) for all \( i = 1, \ldots, n \). Then connect the points using interpolation, and integrate over each region. This can be done in a variety of ways, and the accuracy of the approach will depend on the accuracy of the Eikonal solver that was used to obtain the distance functions \( \phi_i \), and the method used for obtaining the zeros of the \( \phi \partial V_i \).

**3. Our Approach.** We now introduce a method of approximating each generator’s influence which does not require the region boundaries. This avoids the computational challenges addressed in the previous section. The idea is simple: Instead of computing an integral over each generalized Voronoi region, \( V_i \), to obtain the influence, accumulate the initial density to a neighborhood of the generator, and then integrate the accumulated density over this neighborhood. This shifts the problem from integrating over an unknown region, to integrating over a known region. The key to this approach is to accumulate the initial density in the correct way: we do this by iterating a Markov operator until a stationary density is reached. The Markov operator moves the initial density towards the closest generator, with some error in a neighborhood of the region boundaries.

**3.1. The Markov Kernel Operator.** Let \( D \) denote the set of densities on \( \Omega \), that is all nonnegative \( \rho \in L^1(\Omega) \) such that \( \|\rho\|_{L^1(\Omega)} = 1 \). A linear operator \( P : L^1(\Omega) \to L^1(\Omega) \) is called a **Markov operator** if \( PD \subset D \). A Markov kernel operator has even more structure. It is defined in terms of a **stochastic kernel**, Compute the next level set functions.
which is a nonnegative function $k : \Omega \times \Omega \to \mathbb{R}$ that satisfies

$$\int_{\Omega} k(x,y) \, dx = 1, \quad y \in \Omega \text{ a.e.}$$

The stochastic kernel is deterministic; it is termed stochastic in the same spirit as a stochastic matrix. In fact, by discretizing a stochastic kernel appropriately, a stochastic matrix is obtained. We define the Markov kernel operator $^2P$ by its action on $\mu \in D$:

$$P[\mu](x) = \int_{\Sigma} k(x,y) \mu(y) \, dy. \quad (3.1)$$

We iterate such a Markov kernel operator to accumulate the initial density to neighborhoods around the generators, and finally integrate over these known regions to obtain influences. All information about how this accumulation is encapsulated in the stochastic kernel, which is constructed next. Note that this method is only valid for generators that do not intersect the boundary of the domain, that is, $\Gamma_i \cap \partial\Omega = \emptyset$. If this intersection is non-empty, the Markov operator we construct may send mass towards the boundary of the domain.

### 3.2. The Stochastic Kernel.

The kernel is constructed by first solving for the minimum distance from all points $x \in \Omega$ to any generator—this is the solution to the Eikonal equation with the generators as initial contours. In particular, we first solve the following Eikonal equation: Find the function $\phi : \Omega \to \mathbb{R}$ such that

$$|\nabla \phi| = 1 \quad \text{in } \Omega$$
$$\phi = 0 \quad \text{on } \bigcup_{i=1}^{n} \Gamma_i. \quad (3.2)$$

We will transform this solution, $\phi$, by changing the sign and scale, to put the generators at a local maximum and convert the “ridges” into “valleys”. We also scale the solution to be in the range of $[0, 1]$. In doing so, the kernel will ensure that the initial density moves from the valleys towards the generators. To this end, we define the transformed Eikonal solution $\bar{\phi} : \Omega \to [0, 1]$ as:

$$\bar{\phi}(x) := 1 - \frac{\text{abs}(\phi(x))}{\max_{y \in \Omega}(\phi(y))} \quad (3.3)$$

Let $\alpha > 0$. We may now define the function that assigns highest value to the generators, and converts the ridges of the Eikonal solution to valleys. We call this function $q : \Omega \to [0, 1 - \alpha]$, and define it as follows:

$$q(x) = \begin{cases} \bar{\phi}(x), & \text{if } \bar{\phi}(x) < 1 - \alpha \\ 1 - \alpha, & \text{else.} \end{cases} \quad (3.4)$$

The parameter $\alpha$ is introduced to guarantee that the stochastic kernel is not singular. This will be discussed more in the following sections. The construction of $q(x)$ is depicted in Figure (3.1) for one spatial dimension.

---

$^2$This equation was obtained by looking at a discrete-time Markov pure jump process whose initial position is generated by $\mu$. $P[\mu]$ describes the probability that the process is at any point in $\Omega$ after one time step.
To define the stochastic kernel we require some notation. Let $\bar{B}_\varepsilon(x)$ be the closed ball of radius $\varepsilon > 0$ around a point $x \in \Omega$. Moreover, let $\beta > 0$; this is another parameter that ensures the regularity of the kernel. We will first define $\tilde{k}(x, y)$, and then normalize it to be stochastic. The form of $\tilde{k}(x, y)$ is:

$$
\tilde{k}(x, y) = \begin{cases} 
q(x) - q(y) + \beta, & \text{for } y \in \bar{B}_\varepsilon(x) \text{ and } q(x) \geq q(y) \\
0 & \text{otherwise.}
\end{cases} \tag{3.5}
$$

This function is positive only for $x$ and $y$ within a small distance $\varepsilon$ of one another. We do not allow the initial density to evolve nonlocally, as we would like the density to accumulate around the closest generator. We constrain the function $\tilde{k}(x, y)$ to be positive only where $q(x)$ is greater than or equal to $q(y)$: that is, $\tilde{k}(x, y) > 0$ when the point $x$ is closer to the generator than the point $y$. In this way $\tilde{k}(x, y)$ will help to propel the initial density towards the closest generator. The stochastic kernel is:

$$
k_{\Gamma}(x, y) = \frac{\tilde{k}(x, y)}{\int_{u \in \Omega} \tilde{k}(u, y) \, du} \tag{3.6}
$$

$$
= \frac{(q(x) - q(y) + \beta) \mathbb{1}_{\{q(x) \geq q(y)\}} \mathbb{1}_{\{y \in \bar{B}_\varepsilon(x)\}}}{\int_{u \in \Omega} (q(u) - q(y) + \beta) \mathbb{1}_{\{q(u) \geq q(y)\}} \mathbb{1}_{\{u \in \bar{B}_\varepsilon(y)\}} \, du}. \tag{3.7}
$$

By construction the kernel in Equation (3.7) is stochastic and the associated Markov kernel operator sends “mass” towards the closest generator. In fact, we show in the next section that when $P$ is iterated, “mass” will accumulate in a neighborhood of the closest generator. The constant $\beta$ ensures there is is positive density associated to moving to a location of equal quality. This parameter should be chosen to be much smaller than $\varepsilon$ so the information afforded by the quality function is not overpowered.

If $\alpha$ and $\varepsilon$ are chosen such that $2\alpha < \varepsilon$, then the set $A := \{x : q(x) \geq 1 - \alpha\}$ is invariant. This means that once the density moves to $A$, it may never move outside this set. We will give these properties a detailed look in the next section. In Figure 3.2, we show two different stochastic kernels for the one-dimensional case. In (a), the generator is at $x = 0.5$. To the left of the generator, the density will move only towards $x = 0.5$. To the right of the generator, the density moves back towards $x = 0.5$. Moreover, once the density has jumped to a point of distance $\alpha$ or less from the generator, the point will stay in the the invariant region $\{y : \|0.5 - y\| \leq \alpha\}$.

In Figure 3.2 (b), we return to the example from Figure 3.1, where we have
Fig. 3.2. Stochastic kernels for 1D problem on $\Omega = [0, 1]$: The probability of moving from a point $x$ in the domain, to another point $y$ in the domain. (a) One generator at $x = 0.5$, $\alpha = 0.067$, $\varepsilon = 0.14$. (b) Three generators, as in Figure 3.1, $\alpha = 0.06$, $\varepsilon = 0.1267$. In both cases, $\beta = 5.5 \times 10^{-4}$.

three generators at $x = 0.1, 0.4, 0.85$. This is to show how the kernel behaves near the ridges. Similar to part (a), we have density moving in the positive direction to the immediate left of a generator, and in the negative direction to the immediate right of a generator. The $\alpha$-neighborhoods of each generator are invariant sets, where the density gets trapped. Moreover, at the ridge sets (at $x = 0.25, 0.625$), there is symmetric probability of moving towards either generator.

Table 3.1 summarizes the role of each parameter in $k_\Gamma$.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\varepsilon$</td>
<td>The maximum distance density can move in one iteration</td>
<td>$\varepsilon &gt; 0$</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>Controls the size of the invariant set around each generator</td>
<td>$0 \leq \alpha \leq \varepsilon/2$</td>
</tr>
<tr>
<td>$\beta$</td>
<td>Allows density to move to areas of equal quality</td>
<td>$0 \leq \beta &lt;&lt; \varepsilon$</td>
</tr>
</tbody>
</table>

3.3. Stationary Densities and Convergence Results. In this section, we prove some simple results concerning the kernel operator. In particular, we discuss invariant sets, how it accumulates the initial density, and how this allows us to calculate region influences.

3.3.1. Invariant sets and Stationary Densities. We define a set $A \subset \Omega$ to be invariant under $P$ if for any density $\mu \in D$ concentrated on $A$, $\int_{x \in A} P[\mu](x) \, dx = 1$. Let $\Gamma_\alpha^i := \{x \in \Omega \mid |x - y| \leq \alpha, \text{ for } y \in \Gamma_i \}$ denote the $\alpha$-neighborhood of the generator $\Gamma_i$, for $i = 1, \ldots, n$.

**Proposition 3.1.** The sets $\Gamma_\alpha^i$ are invariant under $P$.

**Proof.** Let $\mu$ be a density that is concentrated on $\Gamma_\alpha^i$. Then

$$P[\mu](x) = \int_{y \in \Omega} k_\Gamma(x, y)\mu(y) \, dy = \int_{y \in \Gamma_\alpha^i} k_\Gamma(x, y)\mu(y) \, dy.$$
For any $y \in \Gamma_i^\alpha$, the stochastic kernel reduces to:

$$k_\Gamma(x, y) = \begin{cases} (\int_{u \in \Gamma_i^\alpha \cap B_{\varepsilon}(y)} du)^{-1}, & \text{for } x \in \Gamma_i^\alpha \cap \overline{B}_{\varepsilon}(y) \\ 0, & \text{else.} \end{cases}$$

So we have:

$$\int_{x \in \Gamma_i^\alpha} P[\mu](x) \, dx = \int_{x \in \Gamma_i^\alpha} \left( \int_{y \in \Gamma_i^\alpha} k_\Gamma(x, y) \mu(y) \, dy \right) \, dx$$

$$= \int_{x \in \Gamma_i^\alpha} \left( \int_{y \in \Gamma_i^\alpha} \mathbb{1}_{\{x \in \Gamma_i^\alpha \cap B_{\varepsilon}(y)\}} \mu(y) \, dy \right) \, dx$$

$$= \int_{y \in \Gamma_i^\alpha} \frac{\int_{x \in \Gamma_i^\alpha \cap B_{\varepsilon}(y)} dx}{\int_{u \in \Gamma_i^\alpha \cap B_{\varepsilon}(y)} du} \mu(y) \, dy$$

$$= \int_{y \in \Gamma_i^\alpha} \mu(y) \, dy = 1. \quad (3.8)$$

In Equation 3.8, Fubini’s Theorem was applied. \( \square \)

Before looking at how the operator $P$ moves density to the invariant sets, we first consider the special case of point-generators. In this case the operator $P$ has a stationary density of the following form:

$$\gamma_i(x) = v_{\alpha,i} \mathbb{1}_{\{\Gamma_i^\alpha\}}$$

where $v_{\alpha,i} = \left( \int_{\Gamma_i^\alpha} dx \right)^{-1}$ and $\mathbb{1}_{\{\Gamma_i^\alpha\}}$ denotes the indicator function of the set $\Gamma_i^\alpha$.

**Theorem 3.2.** Let $\Gamma_i = \{z_i\}$ for some point $z_i \in \Omega$, $\forall i = 1, \ldots, n$. Further let $2\alpha \leq \varepsilon$, and for $i = 1, \ldots, n$, let $\hat{w}_i \geq 0$ and $\sum_{i=1}^{n} \hat{w}_i = 1$. Then

$$P[\sum_{i=1}^{n} \hat{w}_i \gamma_i(x)] = \sum_{i=1}^{n} \hat{w}_i \gamma_i(x).$$

**Proof.** Since $P$ is a linear operator, it suffices to show

$$P[\mathbb{1}_{\{x \in \Gamma_i^\alpha\}}](x) = \int_{\Omega} k_\Gamma(x, y) \mathbb{1}_{\{y \in \Gamma_i^\alpha\}} \, dy = \mathbb{1}_{\{\Gamma_i^\alpha\}}(x). \quad (3.9)$$

We consider separately the cases $x \not\in \Gamma_i^\alpha$ and $x \in \Gamma_i^\alpha$. For $x \not\in \Gamma_i^\alpha$ and $y \in \Gamma_i^\alpha$, $k_\Gamma(x, y) = 0$. This is because the stochastic kernel $k_\Gamma(x, y)$ never sends mass from a higher $q(y)$ to a lower $q(x)$: $k_\Gamma(x, y) = f(x, y) \cdot \mathbb{1}_{\{q(y) \leq q(x)\}}$. On the other hand, if $y \in \Gamma_i^\alpha$, then $q(y) = 1 - \alpha$. However, for $x \not\in \Gamma_i^\alpha$, $q(x) < 1 - \alpha$. Thus for $x \not\in \Gamma_i^\alpha$, $k_\Gamma(x, y) = 0$, and hence $P[\gamma_i](x) = 0$ on this set.

Suppose $x \in \Gamma_i^\alpha$. Then for any $y \in \Gamma_i^\alpha$,

$$|x - y| = |x - z_i + z_i - y| \leq |x - z_i| + |z_i - y| \leq 2\alpha \leq \varepsilon.$$
That is, \( x \in \Gamma_i^\alpha \) and \( y \in \Gamma_i^\alpha \) imply \( y \in \overline{B_\varepsilon(x)} \), and in particular the kernel is non-zero. For \( x \in \Gamma_i^\alpha \):

\[
P[\mathbb{1}_{\{\Gamma_i^\alpha\}}](x) = \int \Omega k_T(x,y) \mathbb{1}_{\{y \in \Gamma_i^\alpha\}} dy
\]

\[
= \int \Omega \int_{u \in \Omega} \frac{(q(x) - q(y) + \beta) \mathbb{1}_{\{q(y) \leq q(x)\}} \mathbb{1}_{\{y \in B_\varepsilon(x)\}}}{u - q(y) + \beta \mathbb{1}_{\{q(y) \leq q(u)\}} \mathbb{1}_{\{u \in B_\varepsilon(y)\}}} du \mathbb{1}_{\{y \in \Gamma_i^\alpha\}} dy
\]

\[
= \int_{\Gamma_i^\alpha} \int_{u \in \Gamma_i^\alpha} \beta du = 1.
\]

In the last line, we used the fact that for \( x,y \in \Gamma_i^\alpha \), \( q(x) = q(y) = 1 - \alpha \), and that \( q(y) \leq q(u) \) if and only if \( u \in \Gamma_i^\alpha \). Note that for \( u \in \Gamma_i^\alpha \), \( u \) is at most a distance of \( \varepsilon \) from \( y \).

**Remark:** Theorem 3.3.1 holds for any positive constants \( \tilde{\omega}_i \) that sum to one. We will show that for a given initial density \( \mu \) and a given \( \varepsilon \), the Markov operator \( P \) will approach a stationary density such that the constants \( \tilde{\omega}_i \) approximate the generalized Voronoi region influences. For non-point generators, we will demonstrate that the region influences are similarly approximated by

\[
\int_{\Gamma_i^\alpha} P^m[\mu](x) \, dx,
\]

for \( m \) large enough that the density has concentrated on the invariant sets.

**3.3.2. Convergence to Invariant Sets.** In this section we will show that given any starting density, \( \mu \in L^1(\Omega) \), \( P \) will accumulate \( \mu \) onto the invariant sets. That is,

\[
\lim_{m \to \infty} \int_{\Omega \cup \Gamma_i^\alpha} P^m[\mu](x) \, dx \to 0.
\]

First we show that any density concentrated on a tube around the invariant set \( \Gamma_i^\alpha \) will have some mass transported to the invariant set in one iteration of \( P \). To this end, denote the tube around the invariant set by \( R := \Gamma_i^\alpha + \varepsilon \setminus \Gamma_i^\alpha \), that is

\[
R = \{ x \in V_i \big| \alpha < |x - y| \leq \alpha + \varepsilon \text{ for } y \in \Gamma_i \}.
\]

**Lemma 3.3.** Let \( \mu \) be a density supported on \( R \). Then

\[
\int_{\Omega \setminus \Gamma_i^\alpha} P[\mu](x) \, dx < \int_R \mu(x) \, dx.
\]

**Proof.** Consider how \( P \) acts on \( \mu \):

\[
P[\mu](x) = \int_{y \in \overline{B_\varepsilon(x)}} k_T(x,y) \mu(y) \, dy
\]

\[
= \int_{y \in \overline{B_\varepsilon(x) \cap R}} k_T(x,y) \mu(y) \, dy
\]

\[
= 1_{\{x \in \Gamma_i^\alpha\}} \int_{y \in \overline{B_\varepsilon(x) \cap R}} k_T(x,y) \mu(y) \, dy + 1_{\{x \notin \Gamma_i^\alpha\}} \int_{y \in \overline{B_\varepsilon(x) \cap R}} k_T(x,y) \mu(y) \, dy
\]
In the last line, we used the fact that for \( x \notin \Gamma_i^{\alpha} \), the kernel is zero outside \( R \) (in that region \( q(y) > q(x) \)). We claim \( I > 0 \). To this end, note that \( B_{\varepsilon}(x) \cap R \neq \emptyset \) since \( R \) is adjacent to \( \Gamma_i^{\alpha} \). For \( y \in R \) and \( x \in \Gamma_i^{\alpha} \), \( q(y) < 1 - \alpha = q(x) \) and hence \( k_I(x,y) > 0 \). On the other hand, the support of \( \mu \) is \( R \), so for \( y \in R \), \( \mu > 0 \). Thus \( I > 0 \) and hence

\[
\int_{\Omega \setminus \Gamma_i^{\alpha}} P[\mu](x)dx = 1 - \int_{\Gamma_i^{\alpha}} P[\mu](x)dx < 1 = \int_R \mu(x)dx.
\]

\( \square \)

As a Corollary we have that this is true for any tube a certain distance \( d \) from the invariant set: In one iteration of \( P \), the support will include the region distance \( d - \varepsilon \) from the invariant set.

**Corollary 3.4.** Let \( \mu \) be a density with support on the (nontrivial set) \( R := V_i \cap \left( \Gamma_i^{\alpha+d} \setminus \Gamma_i^{\alpha+d-\varepsilon} \right) \). Then

\[
\int_{\Omega \setminus \Gamma_i^{\alpha+d-\varepsilon}} P[\mu](x)dx < \int_R \mu(x)dx.
\]

In fact in each iteration, the support will include regions a distance \( \varepsilon \) closer to the invariant set. Therefore we have the following result:

**Proposition 3.5.** If \( \mu \) is a density concentrated on a set whose minimum distance to \( \Gamma_i^{\alpha} \) is \( d \), then \( \text{supp}(P^m[\mu]) \cap \Gamma_i^{\alpha} \neq \emptyset \) for all \( m \geq \left\lceil \frac{d}{\varepsilon} \right\rceil \).

Before proving the main convergence result, we recall the following definitions.

**Definition 3.6.** A sequence of measures \( \mu_m \) is called **tight** if for any \( \varepsilon > 0 \) there is a compact subset \( K_\varepsilon \subset \Omega \) such that for all \( m \in \mathbb{N} \), \( \mu_m(K_\varepsilon) > 1 - \varepsilon \).

**Definition 3.7.** We say a sequence of measures \( \mu_m \) **converges weakly** to a measure \( \mu^* \) if for all bounded continuous functions \( g \),

\[
\int_{\Omega} g(x)\mu_m(dx) \to \int_{\Omega} g(x)\mu^*(dx).
\]

In this case we write \( \mu_m \overset{w}{\to} \mu^* \).

We now wish to prove that as \( m \to \infty \), \( P^m[\mu] \) loses its support on \( \Omega \setminus \cup \Gamma_i^{\alpha} \). For the proof presented below, we will require the application of \( P \) to potentially singular measures. Our current definition of \( P \) is based upon the kernel \( k_I \) which is discontinuous because of the cut-off \( 1_{\{\|x-y\| \leq \varepsilon\}} \). By including an annular region of thickness \( \lambda \ll \varepsilon \) around \( B_{\varepsilon}(y) \) which in the radial direction linearly decreases to 0, we obtain an analogous kernel \( \tilde{k}_I(x,y) \) which is continuous in \( x \) and \( y \). We define this linearly decreasing function to be:

\[
k^\text{lin}(x,y) = \frac{\tilde{k}(y + \varepsilon \frac{x-y}{\|x-y\|}, y)}{\lambda} \|x-y\| \left(1 - \frac{\varepsilon + \lambda}{\|x-y\|}\right) \|x-y\|.
\]
Then the modified kernel has the following form:

\[ \tilde{k}_T(x, y) = \frac{\tilde{k}(x, y) \mathbb{1}_{\|x - y\| \leq \varepsilon} + k^{\text{lin}}(x, y) \mathbb{1}_{\|x - y\| \leq \varepsilon + \lambda}}{\int_{u \in \Omega} k(u, y) \mathbb{1}_{\|u - y\| \leq \varepsilon} + k^{\text{lin}}(u, y) \mathbb{1}_{\|u - y\| \leq \varepsilon + \lambda} \, du}. \]

We can then define the associated operator \( \tilde{P}^m[\mu] \) for any probability measure \( \mu \) with support on \( \Omega \). All the basic properties hold true, and the resulting algorithm could also be used. In particular, the Lemma 3.3, Corollary 3.4 and Proposition 3.5 hold for \( \tilde{P} \) and any probability measure \( \mu \). Note further that by taking \( \lambda \) less than the grid size, there is no numerical difference between using \( P \) or \( \tilde{P} \).

**Theorem 3.8.** Let \( \mu \in L^1(\Omega) \) be a density. Then,

\[ \lim_{m \to \infty} \int_{\Omega \cup \Gamma_\alpha} \tilde{P}^m[\mu](x) \, dx \to 0. \]

**Proof.** Let

\[ a_0 = \int_{\Omega \cup \Gamma_\alpha} \mu(x) \, dx \quad \ldots \quad a_m = \int_{\Omega \cup \Gamma_\alpha} \tilde{P}^m[\mu](x) \, dx. \]

By the Lemma 3.3, Corollary 3.4, and Proposition 3.5, \( \{a_m\}_{m=0}^\infty \) is a nonincreasing sequence: Indeed, if \( \tilde{P}^m[\mu] \) has support in an \( \alpha + \varepsilon \)-neighborhood of any generator, then by Lemma 3.3, \( a_{m+1} < a_m \). If \( \tilde{P}^m[\mu] \) doesn’t have support in an \( \alpha + \varepsilon \)-neighborhood, but has support in an \( \alpha + \varepsilon + d \)-neighborhood of some generator, then in \( a_{m+\lceil d/\varepsilon \rceil} < a_m \) by Corollary 3.4 and Proposition 3.5. If \( \tilde{P}^m[\mu] \) has no support outside \( \cup \Gamma_\alpha \), we’re done, because \( a_m \equiv 0 \).

Since \( 1 \geq a_m \geq 0 \) for all \( m \), the sequence \( \{a_m\}_{m=0}^\infty \) converges. Our goal is to show that it converges to zero. Suppose not. Then there is some \( C \in (0, 1) \) such that

\[ \lim_{m \to \infty} a_m = c. \]

Because \( \tilde{P} \) is a Markov Operator on a closed, bounded domain, the sequence of measures \( \tilde{P}^m[\mu] \) is tight. By Prokhorov’s Theorem, for every tight sequence of measures, there is a weakly convergent subsequence ([2]). So for some \( m_k, k \in \mathbb{N} \),

\[ \tilde{P}^{m_k}[\mu] \xrightarrow{\text{w}} \mu^*. \]

Note here that we can not a priori guaranteed that the measure \( \mu^* \) is absolutely continuous with respect to Lebesgue measure. This is the reason we are working with \( \tilde{P} \). Since \( \lim_{m \to \infty} a_m = c \), we know that \( \int_{\Omega \cup \Gamma_\alpha} \mu^*(dx) = c \). So there is some set in \( \Omega \setminus \cup \Gamma_\alpha \) to which \( \mu^* \) assigns mass. We claim that for any \( l \in \mathbb{N} \),

\[ \tilde{P}^{m_k+l}[\mu] \xrightarrow{\text{w}} \tilde{P}^l[\mu^*]. \tag{3.10} \]

First let \( l = 1 \). Then for any \( g \in C(\Omega \setminus \cup \Gamma_\alpha) \), it suffices to prove that

\[ \int g(x) \tilde{k}_T(x, y) \, dx \quad \text{is a continuous function of } y. \tag{3.11} \]
As then,
\[
\int g(x) \tilde{P}^m \mu(x) dx = \int g(x) \int \tilde{k}_\Gamma(x, y) \tilde{P}^m \mu(y) dy \, dx
\]
\[
= \int \left( \int g(x) \tilde{k}_\Gamma(x, y) dx \right) \tilde{P}^m \mu(y) dy
\]
\[
\rightarrow \int \left( \int g(x) \tilde{k}_\Gamma(x, y) dx \right) \mu^*(dy)
\]
\[
= \int g(x) \bar{P} \mu^*(dx).
\]

But (3.11) holds true by the continuity of \( \tilde{k}_\Gamma \). Since the induction step is analogous, we have (3.10). But now, \( \mu^* \) assigns positive probability on \( \Omega \setminus \Gamma_i^\alpha \). By Proposition 3.5 (applied to \( \bar{P} \) and any probability measure \( \mu^* \)), for \( l \geq \left[ \frac{d}{2} \right] + 1 \in \mathbb{N} \), \( \bar{P}^l \mu^* \) will send some of this mass to the generators. Therefore \( \int_{\Omega \setminus \Gamma_i^\alpha} \bar{P}^l \mu^*(dx) < c \). But this is a contradiction and \( c = 0 \). \( \square \)

### 3.3.3. Approximating Influences.

In this section we demonstrate what error we obtain from iterating the Markov Operator.

**Theorem 3.9.** Given \( \mu \in L^1(\Omega) \), then

\[
\lim_{m \to \infty} \int_{\Gamma_i^\alpha} P^m \mu(x) \, dx = w_i + O(\varepsilon).
\]

**Proof.** Consider the generalized Voronoi region \( V_i \). Define

\[
V_i^{\varepsilon/2} = \{ x \in \Omega : |x - y| < \varepsilon/2, \ y \in \partial V_i \}.
\]

Furthermore for \( m = 1, 2, \ldots \), define \( k_m(x, y) \) inductively by

\[
k_1(x, y) = k_\Gamma(x, y), \quad \ldots \quad k_m(x, y) = \int k_\Gamma(x, z) k_{m-1}(z, y) dz.
\]

Moreover, let \( \mu(x) = \mu_1(x) + \mu_2(x) + \mu_3(x) \), where \( \text{supp(} \mu_1) \subset V_i \setminus V_i^{\varepsilon/2} \), \( \text{supp(} \mu_2) \subset V_i^{\varepsilon/2} \), and \( \text{supp(} \mu_3) \subset (V_i \cup V_i^{\varepsilon/2})^c \). Then for any \( m \),

\[
\int_{\Gamma_i^\alpha} P^m \mu(x) \, dx = \int_{\Gamma_i^\alpha} P^m \mu_1(x) \, dx + \int_{\Gamma_i^\alpha} P^m \mu_2(x) \, dx + \int_{\Gamma_i^\alpha} P^m \mu_3(x) \, dx
\]
\[
= \int_{\Gamma_i^\alpha} P^m \mu_1(x) \, dx + \int_{\Gamma_i^\alpha} P^m \mu_2(x) \, dx.
\]

For \( y \in \text{supp(} \mu_1) \), \( k_\Gamma(x, y) \) will only be nonzero for \( x \in \text{supp(} \mu_1) \). Therefore \( \text{supp(} P^m \mu_1(x) \) \subset V_i \setminus V_i^{\varepsilon/2} \). But we know that all mass converges to the invariant sets \( \Gamma_i^\alpha \), so \( \text{supp(} P^m \mu_1(x) \) \rightarrow \Gamma_i^\alpha \) as \( m \to \infty \). Since this mass is conserved,
\[
\lim_{m \to \infty} I_1 = \int_{V_i \setminus V_{i/2}} \mu_1(x) \, dx.
\]

Now consider \( I_2 \):

\[
I_2 = \int_{\Gamma_i} \left( \int_{\Omega} k_m(x, y) \mu(y) \mathbb{1}_{y \in V_{i/2}}(y) \, dy \right) \, dx
\]
\[
= \int_{\Omega} \mu(y) \mathbb{1}_{y \in V_{i/2}}(y) \int_{\Gamma_i} k_m(x, y) \, dx \, dy
\]
\[
\leq \int_{\Omega} \mu(y) \mathbb{1}_{y \in V_{i/2}}(y) \, dy
\]
\[
\leq \|\mu\|_{\infty} \cdot \text{measure}(V_{i/2}) = O(\varepsilon).
\]

Here we use the fact that by Fubini, \( k_m \) is a stochastic kernel, and that the measure of the set \( V_{i/2} \) is \( O(\varepsilon) \).

Similarly,

\[
\int_{V_i \cap V_{i/2}} \mu(x) \, dx \leq \|\mu\|_{\infty} \cdot \text{measure}(V_i \cap V_{i/2}) = O(\varepsilon).
\]

Therefore,

\[
\lim_{m \to \infty} \int_{\Gamma_i} P^m[\mu](x) \, dx = \lim_{m \to \infty} (I_1 + I_2)
\]
\[
= \int_{V_i \setminus V_{i/2}} \mu(x) \, dx + O(\varepsilon) + \int_{V_i \cap V_{i/2}} \mu(x) \, dx - \int_{V_i \cap V_{i/2}} \mu(x) \, dx
\]
\[
= \int_{V_i} \mu(x) \, dx + O(\varepsilon)
\]
\[
= w_i + O(\varepsilon)
\]

\[\square\]

**Remark:** For \( d = 2 \), \( \mu = |\Omega|^{-1} \), and point generators,

\[
\lim_{m \to \infty} \int_{\Gamma_i} P^m[\mu](x) \, dx = w_i + O(\varepsilon^2).
\]

For point generators, the region boundaries \( \partial V_i \) are piecewise linear. In the region \( V_{i/2} \), the operator \( P \) sends mass to the "wrong" generator. However, when the region boundaries are linear, there is a symmetry that eliminates this error. For any point \( y \in V_{i/2} \), there is a symmetric point \( y' \in V_{i/2} \) such that \( y' \in V_j \), and \( \int_{V_j} P[\delta_y] \, dx = \int_{V_i} P[\delta_{y'}] \, dx \). To find \( y' \), simply project \( y \) perpendicularly onto \( \partial V_i \). Now continue perpendicularly from \( \partial V_i \) for the same distance into \( V_j \). There you will have \( y' \). This symmetry degrades in \( \varepsilon \)-neighborhoods of the corners of \( \partial V_i \). However, the corners are \( O(\varepsilon^2) \).
4. Numerical Scheme. In this section we discuss the numerical discretization of the Markov Operator, the algorithm to compute influences, and error estimates.

4.1. Computational Domain. We now discuss how to calculate the influences on a discrete computational grid. Because we are interested in the integral of $P^m[\mu]$ on the invariant set, we construct a finite volume method. Using a finite volume approach to discretize the Markov operator, we avoid singularities in the kernel, as we are no longer working with the density, but rather the integrated density. Therefore, we take $\alpha = 0, \beta = 0$, and we set $\epsilon = 2 \cdot h$, for example. We present the discretization of the Markov Operator in two dimensions for ease of notation.

Because $\Omega \subset \mathbb{R}^2$ is a bounded domain, there exist $R_1, R_2, R_3, R_4 \in \mathbb{R}$ such that $\Omega \subset [R_1, R_2] \times [R_3, R_4]$. We take $R_1, R_2, R_3, R_4$ such that $[R_1, R_2] \times [R_3, R_4]$ is the smallest square covering $\Omega$. Let $h = \frac{R_4 - R_3}{N}$. Then the computational domain $\Omega_h$ is defined to be all pairs $(x_i, y_j)$ of the following form:

$x_i = R_1 + (i - 1) \cdot h, \ i = 1, \ldots, N + 1$

$y_j = R_3 + (j - 1) \cdot h, \ j = 1, \ldots, N + 1$

For all functions previously defined on the bounded domain $\Omega$, including the density $\mu \in L^1(\Omega)$ and the kernel $k_\Gamma$, we extend them by zero to all of the computational domain.

Remark: In the previous sections, we carefully defined the stochastic kernel in terms of the regularization parameters $\alpha, \beta$, though we now set them to zero for the numerics. This was for ease of analysis: we first demonstrate that the operator $P$ has the invariant sets $\Gamma^\alpha_i$. Then it is clear that as $\alpha \to 0$, the operator concentrates mass along singular sets. Because we simply need the measure that $\lim_{m \to \infty} P^m[\mu]$ assigns to the invariant sets, numerically we use a finite volume method which allows us to let $\alpha, \beta = 0$.

4.2. Discretization and Error Analysis: One Iteration. To obtain an iterative scheme, we first approximate the kernel by a piecewise constant function, and subsequently discretize all remaining integrals using the Trapezoidal rule. We then analyze the error associated with one iteration, and with multiple iterations.

4.2.1. Approximating the Kernel. We approximate $k_\Gamma(x, y, u, v)$ by a piecewise constant function along each gridbox in $(u, v)$. We choose this constant to be a weighted average of the kernel evaluated in the last two variables at grid points. Let $J \subset \{(i, j) \mid i, j \in \{1, \ldots, N + 1\}\}$ be some index set, which we will next fix. Let $\sum_{(i,j) \in J} \lambda_{ij} = 1$. Then,

$$k_\Gamma(x, y, u, v) \approx \sum_{(i,j) \in J} \lambda_{ij} k_\Gamma(x, y, x_i, y_j).$$

In particular, for $(u, v) \in [x_i, x_{i+1}) \times [y_j, y_{j+1})$, we either approximate the kernel $k_\Gamma(x, y, u, v)$ by it’s value at the left lower endpoint,

$$k_\Gamma(x, y, u, v) = k_\Gamma(x, y, x_i, y_j) + O(h), \quad (4.1)$$

or by the average of the four endpoints,

$$k_\Gamma(x, y, u, v) = \frac{1}{4} k_\Gamma(x, y, x_i, y_j) + \frac{1}{4} k_\Gamma(x, y, x_{i+1}, y_j)$$
\[
+ \frac{1}{4} k_T(x, y, x_i, y_j) + \frac{1}{4} k_T(x, y, x_{i+1}, y_j) + O(h).
\]

We compute the error bounds using the first approximation (Equation 4.1). Let \( \varepsilon = c \cdot h \) for \( c \geq 1 \in \mathbb{Z} \) (but \( c << N \)). We let \( Q_{ij}^{(1)} \) denote the mass moved to the region \([x_i, x_{i+1}) \times [y_j, y_{j+1})\) after one iteration:

\[
Q_{ij}^{(1)} := \int_{x_i}^{x_{i+1}} \int_{y_j}^{y_{j+1}} P[\mu](x, y) dy dx \\
= \int_{x_i}^{x_{i+1}} \int_{y_j}^{y_{j+1}} \left( \int_{x_i-c}^{x_{i+c}} \int_{y_j-c}^{y_{j+c}} k_T(x, y, u, v) \mu(u, v) \, dv \, du \right) dy dx \\
= \int_{x_i}^{x_{i+1}} \int_{y_j}^{y_{j+1}} \sum_{k=1-i}^{1+i-c} \sum_{l=1-j}^{1+j-c} \left( k_T(x, y, x_k, y_l) + O(h) \right) \int_{x_i}^{x_{i+1}} \int_{y_j}^{y_{j+1}} \mu(u, v) \, dv \, du \, dy \, dx \\
\leq \|\mu\|_\infty (2c+1)^2 h^4 O(h)
\]

The summations and integrals above run from \( k = \max(i - c, 1) \) to \( k = \min(i + c - 1, N - 1) \), and similarly for \( l \), to remain in the computational domain. The kernel is only non-zero for \((x, y)\) and \((u, v)\) within a distance \( \varepsilon = c \cdot h \) of each other. For \((x, y) \in [x_i, x_{i+1}) \times [y_j, y_{j+1}), \ (u, v)\) is restricted to \([x_{i-c}, x_{i+c}) \times [y_{j-c}, y_{j+c})\), which is precisely the range of the inner integration.

In Equation 4.4, \( K_{ijkl} \) is an element of a sparse tensor, with at most \((2c+1)^2 \cdot N^2\) non-zero elements (out of \( N^4 \)). By Equation 4.4, we see that iterating the discretized Markov Operator reduces to multiplying a sparse \( N^2 \times N^2 \) matrix \((K)\) by a \( N^2 \times 1 \) vector \((Q)\).

The error from assuming \( k_T(x, y, u, v) \) is piecewise constant in \((u, v)\) is \( O(h^5) \) under one iteration of the Markov operator. We must still discretize the integral of the kernel and the integral of the initial density; these integrals will be discretized using the Trapezoidal rule.

**4.2.2. Integrating the Kernel.** We construct the discretized kernel with elements

\[
K_{ijkl} = \int_{x_i}^{x_{i+1}} \int_{y_j}^{y_{j+1}} k_0(x, y, x_k, y_l) \, dy \, dx \\
= \left( \int_{x_i}^{x_{i+1}} \int_{y_j}^{y_{j+1}} k_0(x, y, x_k, y_l) \, dy \, dx \right) \cdot \left( \int_{x_{i-c}}^{x_{i+c}} \int_{y_{j-c}}^{y_{j+c}} k_0(x, y, x_k, y_l) \, dy \, dx \right)^{-1} := l_1 \cdot l_2
\]
for \( i, j, k, l \in \{1, \ldots, N\} \). These integrals are computed using the Trapezoidal Rule. Let \( \hat{K}_{ijkl} \) denote the discretized tensor element. Then,

\[
\begin{align*}
\text{Trap}_1 &= \frac{h^2}{4} \left( k_0(x_i, y_j, x_k, y_l) + k_0(x_i, y_j, x_{k+1}, y_l) \right) \\
&\quad + \frac{h^2}{4} \left( k_0(x_{i+1}, y_j, x_k, y_l) + k_0(x_{i+1}, y_j, x_{k+1}, y_l) \right),
\end{align*}
\]

\[
\begin{align*}
\text{Trap}_2 &= \sum_{r=k-c}^{k+c-1} \sum_{p=l-c}^{l+c-1} \frac{h^2}{4} \left( k_0(x_r, y_p, x_k, y_l) + k_0(x_r, y_p, x_{k+1}, y_l) \right) \\
&\quad + \sum_{r=k-c}^{k+c-1} \sum_{p=l-c}^{l+c-1} \frac{h^2}{4} \left( k_0(x_{r+1}, y_p, x_k, y_l) + k_0(x_{r+1}, y_p, x_{k+1}, y_l) \right),
\end{align*}
\]

and \( I_1 = \text{Trap}_1 + O(h^3) \) (see Appendix C). Similarly, \( I_2 = \text{Trap}_2 + O(h^3) \). The error from discretizing \( K_{ijkl} \) using the Trapezoidal rule is \( O(h^3) \), that is,

\[
\left| \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} K_{ijkl} Q_{kl} - \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} \hat{K}_{ijkl} Q_{kl} \right| \leq O(h^3).
\]

Observe,

\[
\begin{align*}
E &\leq \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} |K_{ijkl} - \hat{K}_{ijkl}| Q_{kl} \quad (4.6) \\
&= \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} \frac{I_1}{I_2} - \frac{\text{Trap}_1}{\text{Trap}_2} |Q_{kl}| \quad (4.7) \\
&= \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} \frac{I_1 \text{Trap}_2 - \text{Trap}_1 I_2}{\text{Trap}_2 I_2} |Q_{kl}| \quad (4.8) \\
&\leq \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} \left[ \frac{|I_1 - \text{Trap}_1|}{\text{Trap}_1} + \frac{|\text{Trap}_2|}{\text{Trap}_2} \right] |Q_{kl}| \quad (4.9) \\
&\leq O(h^3). \quad (4.10)
\end{align*}
\]

Once we have discretized the final integral, the integral of the initial density, the Markov Operator will be fully discretized and we can analyze the error accumulated in the first step.

**4.2.3. Integrating the Initial Density.** We must take the initial density \( \mu \) and compute the mass matrix \( Q_{ij}^{(0)} \),

\[
Q_{ij}^{(0)} := \int_{x_i}^{x_{i+1}} \int_{y_j}^{y_{j+1}} \mu(u, v) \, dv \, du. \quad (4.11)
\]
These integrals are also computed using the Trapezoidal Rule. Denote \( \hat{Q}^{(0)}_{ij} \) the discretized mass function. Then,

\[
\hat{Q}^{(0)}_{ij} = \frac{h^2}{4} \left( \mu(x_i, y_j) + \mu(x_i, y_{j+1}) + \mu(x_{i+1}, y_j) + \mu(x_{i+1}, y_{j+1}) \right),
\]

and \( Q^{(0)}_{ij} = \hat{Q}^{(0)}_{ij} + O(h^4) \) (see Appendix C). Therefore we have that

\[
\left| i_{c+1} \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} K_{ijkl} Q^{(0)}_{kl} - \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} \hat{K}_{ijkl} \hat{Q}^{(0)}_{kl} \right| \leq O(h^4).
\]

\[= \hat{Q}^{(1)}_{ij}\]

4.2.4. Error Summary for the first iteration. For the first iteration, the error obtained by discretizing the Markov Operator is as follows:

\[
\left| Q^{(1)}_{ij} - \hat{Q}^{(1)}_{ij} \right| \leq \left| Q^{(1)}_{ij} - \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} K_{ijkl} Q^{(0)}_{kl} \right| = O(h^5)
\]

\[+ \left| \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} K_{ijkl} Q^{(0)}_{kl} - \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} \hat{K}_{ijkl} \hat{Q}^{(0)}_{kl} \right| = O(h^3)
\]

\[+ \left| \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} \hat{K}_{ijkl} \hat{Q}^{(0)}_{kl} - \hat{Q}^{(1)}_{ij} \right| = O(h^4)
\]

\[\leq O(h^5).
\]

After one iteration we retain third order accuracy.

4.3. Error Analysis for \( m \) Iterations. Assume that after \( m \) iterations of the numerical scheme, the initial density has concentrated on the invariant sets. Computationally, we find \( m \) to be finite, and in fact \( O(N) \). See Appendix B for more information on the number of iterations until convergence. The error after \( m \) iterations is bounded as follows:

\[
\left| Q^{(m)}_{ij} - Q^{(m)}_{ij} \right| \leq \left| Q^{(m)}_{ij} - \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} K_{ijkl} Q^{(m-1)}_{kl} \right| = E_1
\]

\[+ \left| \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} K_{ijkl} Q^{(m-1)}_{kl} - \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} \hat{K}_{ijkl} \hat{Q}^{(m-1)}_{kl} \right| = E_2
\]

\[\leq O(h^3).
\]
Computing the Measures of Generalized Voronoi Regions

\[ + \left| \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} \hat{K}_{ijkl} Q_{kl}^{(m-1)} - \hat{Q}_{ij}^{(m)} \right| \]

The error \( E_1 \) comes from assuming the kernel is piecewise constant in \((u, v)\). As in Equation (4.3),

\[ E_1 = \mathcal{O}(h) \int_{x_i}^{x_{i+1}} \int_{y_j}^{y_{j+1}} \int_{x_{i-c}}^{x_{i+c}} \int_{y_{j-c}}^{y_{j+c}} P^{m-1[\mu]} dv \, du \, dy \, dx \]

However, \( P^{m-1[\mu]} \) should be almost concentrated along the generators. In the case of point generators, the maximum of the iterated density should be \( \mathcal{O}(N^2) \), as the mass should be concentrated on \( n \) grid boxes of area \( h^2 \) (Recall that \( n \) is the number of generators, which is independent of \( h \)). In the case of curve generators, the iterated density should be \( \mathcal{O}(N) \), as for each generator, the mass should be concentrated on \( \mathcal{O}(N) \) gridboxes of area \( h^2 \). Therefore,

\[ \int_{x_{i-c}}^{x_{i+c}} \int_{y_{j-c}}^{y_{j+c}} P^{m-1[\mu]} dv \, du \leq \begin{cases} \mathcal{O}(h^4) & \text{for curved generators,} \\ \mathcal{O}(h^3) & \text{for point generators.} \end{cases} \]

And in particular,

\[ E_1 \leq \begin{cases} \mathcal{O}(h^4) & \text{for curved generators,} \\ \mathcal{O}(h^3) & \text{for point generators.} \end{cases} \]

The second error comes from discretizing the kernel using the Trapezoidal rule. This calculation is also analogous to the error in the first iteration, adjusting for the concentrated nature of \( P^{m-1[\mu]} \). As in Equation 4.9,

\[ E_2 \leq \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} \left[ I_1 - \text{Trap}_1 \right] \left[ I_2 - \text{Trap}_2 \right] Q_{kl} \]

\[ = \mathcal{O}(h^3) \quad = \mathcal{O}(h^2) = \mathcal{O}(h^3) = \mathcal{O}(h^3) \]

\[ \leq \begin{cases} \mathcal{O}(h^2) & \text{for curved generators,} \\ \mathcal{O}(h) & \text{for point generators.} \end{cases} \]

The third error is as follows:

\[ E_3 = \left| \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} \hat{K}_{ijkl} Q_{kl}^{(m-1)} - \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} \hat{K}_{ijkl} \hat{Q}_{kl}^{(m-1)} \right| \]

\[ \leq \sum_{k=i-c}^{i+c-1} \sum_{l=j-c}^{j+c-1} \left| \hat{K}_{ijkl} \right| \left| Q_{kl}^{(m-1)} - \hat{Q}_{kl}^{(m-1)} \right| \]

This error is the same order as the error of the previous iteration. Since that error will always be dominated by the error from integrating the kernel using the Trapezoidal
rule, in the final step, this error will be:

\[ E_3 \leq \begin{cases} \mathcal{O}(h^2) & \text{for curved generators,} \\ \mathcal{O}(h^1) & \text{for point generators.} \end{cases} \]

4.3.1. Error Summary for \( m \) Iterations. After \( m \) iterations, when the mass has concentrated on the generators, the error is:

\[ |Q_{ij}^{(m)} - \hat{Q}_{ij}^{(m)}| \leq \begin{cases} \mathcal{O}(h^2) & \text{for curved generators,} \\ \mathcal{O}(h^1) & \text{for point generators.} \end{cases} \]

4.4. Obtaining Influences. Given \( \hat{Q}^{(m)} \), and the indices corresponding to each generator, it only remains to sum the values of \( \hat{Q}^{(m)} \) along each generator to obtain the approximated weights. Define \( I_i := \{(j,k) \mid d((x_j,y_k), \Gamma_i) < h\} \). These are the computational points which lie closest to the generator \( \Gamma_i \). Then \( \hat{w}_i = \sum_{(j,k) \in I_i} \hat{Q}_{jk}^{(m)} \). Let the union of these sets be denoted \( \mathcal{I} = \bigcup_{i=1}^n I_i \).

4.4.1. Influences for Curves. In the case of curved generators, this summation will be over \( \mathcal{O}(N) \) points, so an order of accuracy will be lost. The error we obtain by discretizing the Markov operator is

\[
\left| \hat{w}_i - \sum_{(j,k) \in \mathcal{I}} \hat{Q}_{jk}^{(m)} \right| \leq \sum_{(j,k) \in \mathcal{I}} |Q_{jk}^{(m)} - \hat{Q}_{jk}^{(m)}| \\
\leq \mathcal{O}(N)\mathcal{O}(h^2) \\
= \mathcal{O}(h).
\]

Finally, recall that the true influence was denoted \( w_i \). The error for curves between the true influence and the influence obtained by numerically iterating the Markov operator is first order:

\[
\left| w_i - \sum_{(j,k) \in I_i} Q_{jk}^{(m)} \right| \leq \left| w_i - \hat{w}_i \right| + \left| \hat{w}_i - \sum_{(j,k) \in \mathcal{I}} \hat{Q}_{jk}^{(m)} \right| \\
\leq \mathcal{O}(h).
\]

Therefore the method is first order for curved generators.

4.4.2. Influences for Points. In the case of point generators, \( I_i \) consists of one or at most four gridpoints, which is a size \( \mathcal{O}(1) \) set. Then,

\[
\left| \hat{w}_i - \sum_{(j,k) \in I_i} \hat{Q}_{jk}^{(m)} \right| \leq 4 \max_{(j,k) \in I_i} |Q_{jk}^{(m)} - \hat{Q}_{jk}^{(m)}| \leq \mathcal{O}(h).
\]

Moreover,

\[
\left| w_i - \sum_{(j,k) \in I_i} Q_{jk}^{(m)} \right| \leq \left| w_i - \hat{w}_i \right| + \left| \hat{w}_i - \sum_{(j,k) \in \mathcal{I}} \hat{Q}_{jk}^{(m)} \right| \leq \mathcal{O}(h). \]
Therefore the numerical method is first order for point generators as well. These error rates are demonstrated in the next section on numerical results.

Remark: So far in our discussion on errors, we have not addressed the error induced by solving the Eikonal equation numerically. The error of the numerical scheme determines the error in the placement of the ridges. A first order numerical scheme should therefore result in a first order error for the area of the generalized Voronoi regions, which does not degrade the error rate of our scheme. The Eikonal equation can be solved numerically via the Fast Sweeping Method or Fast Marching Method ([23], [21, 22]).

4.5. Algorithm. Implementing this iterative scheme to compute influences involves five steps. The first is to solve the Eikonal equation numerically, we use the Fast Sweeping Method ([23]). In the case of points and circles, the Eikonal solution is the minimum of conic functions, which can be computed exactly. Next, the probability transition tensor $\hat{K}$ must be populated using the Trapezoidal discretization. Third, the initial density must be integrated numerically (via the Trapezoidal rule, exactly, or by any higher order quadrature scheme). Fourth, the scheme must be iterated until the mass is accumulated on the generators. For this, iteration is terminated after the mass outside the set $I$ is less than some tolerance level. The tolerance should be less than the accuracy divided by the number of gridpoints in $I$. For an idea of the accuracy, see Figure 5.4. Finally, the influences are calculated by summing the mass along the gridpoints closest to each generator. These five steps are summarized in Algorithm 2:

Algorithm 2 Calculate Approximate Influences

Given: a bounded domain $\Omega \subset \mathbb{R}^2$, a density $\mu$, and generators $\{\Gamma_i\}_{i=1}^n$.

Set TOL, h.

Find gridpoints $I_i := \{(j,k) \mid d((x_j,y_k),\Gamma_i) < h\}$.

1. Solve Eikonal Equation:
   a. Initialize $\phi(I)$, by interpolation, and set $\phi(\Omega_h \setminus I) = \infty$.
   b. Solve Eikonal equation for $\phi$ on $\Omega_h$ (Equation 3.2).
   c. Obtain the function $q$ on $\Omega_h$ (Equation 3.4).

2. Compute Probability Transition Tensor $\hat{K}$ from $q$ (Equation 4.5).

3. Integrate $\mu$ along gridboxes to obtain $\hat{Q}$ (Equation 4.11).

4. Accumulate mass to $I$:
   while $\|\hat{Q}\|_{L^1(I)} < 1 - TOL$ do
     $\hat{Q} = \hat{K}\hat{Q}$
   end while

5. Obtain influences:
   for $i = 1 : n$ do
     $\hat{w}_i = \sum_{(k,l) \in I_i} \hat{Q}_{kl}$.
   end for

5. Numerical Results. In this section we first provide numerical verification that the numerical scheme is first order for the case of points and curves. In each case, the initial density is uniform. For the case of curves, we use the calculations in Appendix A to compute the influences exactly. We then give an application of the
method to the Los Angeles (LA) County highway system. The method computes the fraction of population that lives closest to each highway.

5.1. Error. Below we present the error rates for the numerical scheme in the case of point and circle generators. In Scheme 1, we use the assumption that the kernel $k(x, y, u, v)$ is approximated by the left-bottom endpoint along each gridbox. In Scheme 2, we use approximate $k(x, y, u, v)$ by the average of its four corner values. The point generators are the white dots shown in Figure 5.1. The circular generators are the blue lines in Figure 5.2. The ridge set for the circular generators were calculated using the formulas of Appendix A.

Scheme 1 results in first order convergence for both points and circles. Scheme 2 results in first order convergence for circles, which is expected as $|w_i - \tilde{w}_i| = \mathcal{O}(h)$ for curved generators. The accuracy is improved by almost an order by using Scheme 2 in the case of circles.

Scheme 2 almost yields second order convergence for the Voronoi case. This is likely because the averaging in effect smooths the kernel. The kernel is $C^0$ along the domain of integration, which results in the trapezoidal error of $\mathcal{O}(h^3)$. The smoothing from averaging the endpoints may result in an error closer to $\mathcal{O}(h^4)$. Using this trapezoidal error in the previous error analysis would give almost an extra order of accuracy for the case of points. Based on these results, we recommend the second scheme.

5.2. Application: Los Angeles County Highways. Next we compute the influences of the highways in Los Angeles (LA) County. The influence describes the
fraction of population living closest to each highway in LA County. The population density data and influences are depicted in Figure 5.2, (a) and (b). The map tiles were obtained from [16, 19]. The highway data and geographic boundary data for LA County zip codes were obtained from the U.S. Census Bureau [5]. It is accurate as of January 1, 2010. The population density for each zip code in Los Angeles county was also obtained from the U.S. Census Bureau [4]. The highway data came in the form of latitude and longitude coordinates. These coordinates were used as the initial contour for solving the Eikonal equation. The solution was obtained in a square domain, where the population density was set to zero outside Los Angeles county. From a purely qualitative perspective, the results are certainly intuitive: there are more people living near the larger interstate freeways, and less people living in the domain of influence of the state highways. We present the numerical results in Table 5.1.

6. Discussion. We have presented an efficient numerical scheme to compute the influence of generalized Voronoi regions. The scheme is first order accurate and can deal with generators of arbitrary co-dimension. The method avoids solving the Eikonal equation numerically for each generator and never requires the computation of the Eikonal gradient. While we have shown experiments in 2D, the method applies in any space dimension. It would be interesting to find novel 3D applications for our algorithm.

As we mentioned in the introduction, one motivation for this work is geographic distributions and urban planning. From this perspective, an important problem is to find suitably optimal placements (positions) of generators. For point generators, there has been much work on centroidal Voronoi tessellation (see, for example, [6, 7]) where given a finite number of points and a domain Ω one seeks to place the points in such a way such that they are the centroids of their respective Voronoi region. To this end, there exists a very successful and simple algorithm – Lloyd’s Algorithm. However simple, only partial rigorous convergence results have been proven for Lloyd’s algorithm (see [9] for a proof of weak global convergence and non-degeneracy). For generators consisting of rigid curves, centroidal type ideas are naturally more complicated (see, for example, [15, 11]). Given that our method allows for the fast computation of the centroids for the general Voronoi regions, it would be interesting to investigate the use of our algorithm to compute suitably optimal placements of rigid curves.

Appendix A. Example of Analytically Available Region Boundaries.

It is worth noting that in some cases the formulas for the boundaries of the generalized Voronoi regions are analytically available. In this example we consider

<table>
<thead>
<tr>
<th>Highway Name</th>
<th>Influence</th>
<th>Highway Name</th>
<th>Influence</th>
</tr>
</thead>
<tbody>
<tr>
<td>I-10</td>
<td>13.05 %</td>
<td>I-710</td>
<td>5.46 %</td>
</tr>
<tr>
<td>I-5</td>
<td>12.41 %</td>
<td>I-605</td>
<td>4.67 %</td>
</tr>
<tr>
<td>I-405</td>
<td>11.64 %</td>
<td>State Rte. 91</td>
<td>3.81 %</td>
</tr>
<tr>
<td>I-210</td>
<td>10.53 %</td>
<td>State Rte. 170</td>
<td>3.21 %</td>
</tr>
<tr>
<td>I-110</td>
<td>10.52 %</td>
<td>State Rte. 118</td>
<td>2.30 %</td>
</tr>
<tr>
<td>U.S. 101</td>
<td>8.07 %</td>
<td>State Rte. 134</td>
<td>1.80 %</td>
</tr>
<tr>
<td>State Rte. 60</td>
<td>5.78 %</td>
<td>State Rte. 2</td>
<td>0.97 %</td>
</tr>
<tr>
<td>I-105</td>
<td>5.49 %</td>
<td>State Rte. 57</td>
<td>0.30 %</td>
</tr>
</tbody>
</table>
circular generators.

In the case of non-overlapping circular generators in two dimensions, we can derive and explicit formula for $\phi_{ij} = 0$. In this case,

$$\phi_i(x, y) = \sqrt{(x - x_i)^2 + (y - y_i)^2} - r_i$$

Solving for $\phi_{ij} = 0$ yields

$$y = \frac{-B \pm \sqrt{B^2 - 4 \cdot A \cdot C}}{2 \cdot A}$$

where $A, B,$ and $C$ are defined as:

$$A = \left( \frac{y_j - y_i}{r_i - r_j} \right)^2 - 1$$

$$B = \frac{(y_j - y_i)}{(r_i - r_j)} \left( y_i^2 - y_j^2 + (x - x_i)^2 - (x - x_j)^2 \right) + y_i + y_j$$

$$C = \left( \frac{y_i^2 - y_j^2 + (x - x_i)^2 - (x - x_j)^2}{2(r_i - r_j)} - \frac{(r_i - r_j)}{2} \right)^2 - (x - x_j)^2 - y_j^2$$

In Equation A.1, the positive or negative root is chosen such that $\phi_i(x, y) = \phi_j(x, y)$. If there is not one unique $y$ value for each $x$, one should use the analogous expression for $x$ as a function of $y$. This equation was used to find the ridges in Figure 5.2. There the generators are shown in black, the region boundaries in blue, and the contour plot of the distance function is in the background.

**Appendix B. CPU Times.**

**B.1. Solving the Eikonal Equation.** The first step in our algorithm is to find the Eikonal solution with the generators as the initial contours. This is done via a Fast Sweeping Method, which is $O(N^2)$ algorithm (see [23]).

**B.2. Constructing the Discretized Kernel.** To construct the discretized kernel, we must populate a matrix with $O(N^2)$ non-zero elements. This takes $O(N^2)$ seconds, which is verified below in Figures B.1 and B.2.

**B.3. Number of Iterations until Convergence.** We conjecture the number of iterations until the mass converges to be $O(N)$. Figures B.3 and B.4 confirm that the number of iterations until convergence is $O(N)$.

**B.4. CPU Time for Convergence to Invariant Sets.** Each iteration of the discretized Markov Operator requires the multiplication of a vector with a sparse $O(N^2)$ matrix. The cpu time required for this is $O(N^2)$. As $O(N)$ iterations are required to obtain convergence, we expect an over cpu time that is $O(N^3)$. This is exactly what is recovered in Figures B.5 and B.6.

**Appendix C. Trapezoidal Rule Error.**

Let $f : \mathbb{R}^2 \to \mathbb{R}$ have two continuous partial derivatives in both $x$ and $y$. Let $h = x_{i+1} - x_i = y_{i+1} - y_j$ for $i, j \in \{1, \ldots, N\}$. Then by integration by parts,
\[ \int_0^h \int_0^h f(x_i + t, y_j + s) dt \, ds = \frac{h^2}{4}(f(x_i, y_j) + f(x_i, y_{j+1}) + f(x_{i+1}, y_j) + f(x_{i+1}, y_{j+1})) \]

\[ + \frac{h}{2} \int_0^h \left( \frac{(s - \frac{h}{2})^2}{2} - \frac{h^2}{8} \right) f_{yy}(x_{i+1}, y_j + s) ds \]

\[ + \frac{h}{2} \int_0^h \left( \frac{(s - \frac{h}{2})^2}{2} - \frac{h^2}{8} \right) f_{yy}(x_i, y_j + s) ds \]

\[ + \int_0^h \int_0^h \left( \frac{(t - \frac{h}{2})^2}{2} - \frac{h^2}{8} \right) f_{xx}(x_i + t, y_j + s) ds \, dt. \]

Then,

\[ \max_{i,j \in \{1, \ldots, N\}} \left| \int_0^h \int_0^h f(x_i + t, y_j + s) dt \, ds - \text{Trap}_{ij} \right| \leq \frac{h^4}{12} (\|f_{yy}\|_{\infty} + \|f_{xx}\|_{\infty}). \]

So for any function with two continuous partial derivatives, we expect the Trapezoidal rule to yield fourth order accuracy. The stochastic kernel we define is only continuous on \( \{(x, y) \mid |x - y| \leq \varepsilon\} \) (this is precisely the domain of integration). In fact, there will be a gridpoint such that the left and right \( x \) derivatives (similarly for the \( y \) derivatives) will not agree. Numerically, we see that for some gridpoint \((x_i, y_j)\)

\[ \|f_{xx}\|_{\infty} = \left| \frac{f_x(x_{i+1}, y_j) - f_x(x_i, y_j)}{h} \right| = \frac{c}{h}, \]

and similarly for \( f_{yy} \). Therefore in the case of the kernel we have defined, we expect the Trapezoidal rule to yield \( \mathcal{O}(h^3) \) convergence.

REFERENCES


Fig. 5.5. (a) Population Density in LA County. (b) Fraction of LA County Population Living Closest to each Highway.
Fig. B.1. (a) Voronoi Case

Fig. B.2. (b) Generalized Voronoi Case

Fig. B.3. (a) Voronoi Case

Fig. B.4. (b) Generalized Voronoi Case

Fig. B.5. (a) Voronoi Case

Fig. B.6. (b) Generalized Voronoi Case