Graph-Based Learning: Theory and Applications

Jeff Calder

School of Mathematics
University of Minnesota

Mathematics of Machine Learning Course
Brigham Young University
March 5, 2021
Outline

1. Spectral clustering

2. Semi-supervised learning
   - Laplacian regularization
   - Poisson learning

3. Experiments in Python

4. Pointwise consistency for graph Laplacians
Outline

1. Spectral clustering

2. Semi-supervised learning
   - Laplacian regularization
   - Poisson learning

3. Experiments in Python

4. Pointwise consistency for graph Laplacians
Graph-based learning

Let $(\mathcal{X}, \mathcal{W})$ be a graph.

- $\mathcal{X} \subset \mathbb{R}^d$ are the vertices.
- $\mathcal{W} = (w_{xy})_{x, y \in \mathcal{X}}$ are nonnegative edge weights.
- $w_{xy}$ is large when $x$ and $y$ are similar, and small or $w_{xy} = 0$ otherwise.
Some common graph-based learning tasks

1. Clustering (grouping similar datapoints)
2. Semi-supervised learning (propagating labels)
3. Dimension reduction (spectral embeddings)
MNIST (70,000 28 × 28 pixel images of digits 0-9)
MNIST (70,000 $28 \times 28$ pixel images of digits 0-9)

Each image is a datapoint

$$x \in \mathbb{R}^{28 \times 28} = \mathbb{R}^{784}.$$
MNIST (70,000 $28 \times 28$ pixel images of digits 0-9)

- Each image is a datapoint
  \[ x \in \mathbb{R}^{28 \times 28} = \mathbb{R}^{784}. \]
- Geometric weights:
  \[ w_{xy} = \eta \left( \frac{|x - y|}{\varepsilon} \right) \]
MNIST (70,000 $28 \times 28$ pixel images of digits 0-9)

<table>
<thead>
<tr>
<th>5 0 4 1 9 2 1 3 1</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 5 3 6 1 7 2 8 6 9</td>
<td></td>
</tr>
<tr>
<td>4 0 9 1 1 2 4 3 2 7</td>
<td></td>
</tr>
<tr>
<td>3 8 6 9 0 5 6 0 7 6</td>
<td></td>
</tr>
<tr>
<td>1 8 7 9 3 9 8 5 9 3</td>
<td></td>
</tr>
<tr>
<td>3 0 7 4 9 8 0 9 4 1</td>
<td></td>
</tr>
<tr>
<td>4 4 6 0 4 5 6 1 0 0</td>
<td></td>
</tr>
<tr>
<td>1 7 1 6 3 0 2 1 1 7</td>
<td></td>
</tr>
<tr>
<td>9 0 2 6 7 8 3 9 0 4</td>
<td></td>
</tr>
<tr>
<td>6 7 4 6 8 0 7 8 3 1</td>
<td></td>
</tr>
</tbody>
</table>

- Each image is a datapoint
  \[ x \in \mathbb{R}^{28 \times 28} = \mathbb{R}^{784}. \]

- Geometric weights:
  \[ w_{xy} = \eta \left( \frac{|x - y|}{\varepsilon} \right) \]

- $k$-nearest neighbor graph:
  \[ w_{xy} = \eta \left( \frac{|x - y|}{\varepsilon_k(x)} \right) \]
Clustering MNIST

https://divamgupta.com
Graph cuts

**Question:** How do we cluster graph data?

Consider binary clustering (two classes). We can try to minimize a graph cut energy function:

$$\text{Cut}(A) = \sum_{x, y \in X; x \in A, y \not\in A} w_{xy}.$$ 

Tends to produce unbalanced classes (e.g., $A = \{x\}$).
Question: How do we cluster graph data?

Consider binary clustering (two classes). We can try to minimize a graph cut energy

\[
\text{(Min-Cut)} \quad \min_{A \subseteq \mathcal{X}} \text{Cut}(A) := \sum_{\substack{x, y \in \mathcal{X} \\ x \in A, y \notin A}} w_{xy}.
\]
Question: How do we cluster graph data?

Consider binary clustering (two classes). We can try to minimize a graph cut energy

\[(\text{Min-Cut}) \min_{A \subseteq \mathcal{X}} \text{Cut}(A) := \sum_{\substack{x, y \in \mathcal{X} \atop x \in A, y \notin A}} w_{xy}.\]

Tends to produce unbalanced classes (e.g., \(A = \{x\}\)).
Graph cuts

**Question:** How do we cluster graph data?

Consider binary clustering (two classes). We can try to minimize a graph cut energy

\[
(Balanced-Cut) \quad \min_{A \subset \mathcal{X}} \frac{\text{Cut}(A)}{\text{Vol}(A) \text{Vol}(\mathcal{X} \setminus A)},
\]

where

\[
\text{Vol}(A) = \sum_{x \in A} \sum_{y \in \mathcal{X}} w_{xy}.
\]
Question: How do we cluster graph data?

Consider binary clustering (two classes). We can try to minimize a graph cut energy

\[
\text{Balanced-Cut} \quad \min_{A \subseteq \mathcal{X}} \frac{\text{Cut}(A)}{\text{Vol}(A)\text{Vol}(\mathcal{X} \setminus A)},
\]

where

\[
\text{Vol}(A) = \sum_{x \in A} \sum_{y \in \mathcal{X}} w_{xy}.
\]

Gives good clusterings but very computationally hard (NP-hard).
Spectral clustering

For $A \subset \mathcal{X}$ set

$$u(x) = \begin{cases} 
1, & \text{if } x \in A \\
0, & \text{otherwise.}
\end{cases}$$
Spectral clustering

For $A \subset \mathcal{X}$ set

$$u(x) = \begin{cases} 1, & \text{if } x \in A \\ 0, & \text{otherwise.} \end{cases}$$

Then we have

$$\text{Cut}(A) = \sum_{\substack{x, y \in \mathcal{X} \\ x \in A, y \notin A}} w_{xy} = \frac{1}{2} \sum_{x, y \in \mathcal{X}} w_{xy} (u(x) - u(y))^2$$

and

$$\text{Vol}(A) = \sum_{x, y \in \mathcal{X}} w_{xy} u(x).$$
Spectral clustering

For \( A \subset X \) set

\[
u(x) = \begin{cases} 
1, & \text{if } x \in A \\
0, & \text{otherwise.} 
\end{cases}
\]

Then we have

\[
\text{Cut}(A) = \sum_{\substack{x, y \in X \\
x \in A, y \notin A}} w_{xy} = \frac{1}{2} \sum_{x, y \in X} w_{xy} (u(x) - u(y))^2
\]

and

\[
\text{Vol}(A) = \sum_{x, y \in X} w_{xy} u(x).
\]

This allow us to write the balanced cut problem as

\[
\min_{u: X \to \{0, 1\}} \frac{\sum_{x, y \in X} w_{xy} (u(x) - u(y))^2}{\sum_{x, y, x', y' \in X} u(x) w_{xy} (1 - u(y')) w_{x', y'}}.
\]
Spectral clustering

Consider solving the similar, relaxed, problem

$$\min_{u: \mathcal{X} \to \mathbb{R}} \sum_{x, y \in \mathcal{X}} w_{xy} (u(x) - u(y))^2$$

$$\sum_{x \in \mathcal{X}} u(x) \neq 0 \frac{\sum_{x \in \mathcal{X}} u(x)^2}{\sum_{x \in \mathcal{X}} u(x)^2}.$$
Spectral clustering

Consider solving the similar, relaxed, problem

\[
\min_{u: \mathcal{X} \to \mathbb{R}} \sum_{x,y \in \mathcal{X}} w_{xy} (u(x) - u(y))^2 \frac{\sum_{x \in \mathcal{X}} u(x)^2}{\sum_{x \in \mathcal{X}} u(x)^2}.
\]

The solution is the smallest non-trivial eigenvector (Fiedler vector) of the graph Laplacian

\[
\mathcal{L} u(x) = \sum_{y \in \mathcal{X}} w_{xy} (u(x) - u(y)).
\]
Spectral clustering

Consider solving the similar, relaxed, problem

\[
\min_{u: \mathcal{X} \to \mathbb{R}} \sum_{x,y \in \mathcal{X}} w_{xy} (u(x) - u(y))^2 \quad \text{subject to} \quad \sum_{x \in \mathcal{X}} u(x) \neq 0 \quad \text{and} \quad \sum_{x \in \mathcal{X}} u(x)^2 = \sum_{x \in \mathcal{X}} u(x)^2.
\]

The solution is the smallest non-trivial eigenvector (Fiedler vector) of the graph Laplacian

\[
\mathcal{L} u(x) = \sum_{y \in \mathcal{X}} w_{xy} (u(x) - u(y)).
\]

Binary spectral clustering:

1. Compute Fiedler vector \( u : \mathcal{X} \to \mathbb{R} \).
2. Set \( A = \{x \in \mathcal{X} : u(x) > 0\} \).
Spectral clustering: To cluster into $k$ groups:

1. Compute first $k$ eigenvectors of the graph Laplacian $\mathcal{L}$:

   $$u_1, \ldots, u_k : \mathcal{X} \rightarrow \mathbb{R}.$$
Spectral clustering

**Spectral clustering:** To cluster into \( k \) groups:

1. Compute first \( k \) eigenvectors of the graph Laplacian \( \mathcal{L} \):
   \[
   u_1, \ldots, u_k : \mathcal{X} \to \mathbb{R}.
   \]

2. Define the spectral embedding \( \Psi : \mathcal{X} \to \mathbb{R}^k \) by
   \[
   \Psi(x) = (u_1(x), u_2(x), \ldots, u_k(x)).
   \]
Spectral clustering

Spectral clustering: To cluster into $k$ groups:

1. Compute first $k$ eigenvectors of the graph Laplacian $L$:

   $$u_1, \ldots, u_k : \mathcal{X} \rightarrow \mathbb{R}.$$

2. Define the spectral embedding $\Psi : \mathcal{X} \rightarrow \mathbb{R}^k$ by

   $$\Psi(x) = (u_1(x), u_2(x), \ldots, u_k(x)).$$

3. Cluster the point cloud $\mathcal{Y} = \Psi(\mathcal{X})$ with your favorite clustering algorithm (often $k$-means).
Spectral methods in data science

Spectral methods are widely used for dimension reduction and clustering in data science and machine learning.

- Spectral clustering [Shi and Malik (2000)] [Ng, Jordan, and Weiss (2002)]
- Laplacian eigenmaps [Belkin and Niyogi (2003)]
- Diffusion maps [Coifman and Lafon (2006)]
Outline

1. Spectral clustering

2. Semi-supervised learning
   - Laplacian regularization
   - Poisson learning

3. Experiments in Python

4. Pointwise consistency for graph Laplacians
Graph-based semi-supervised learning

Given:

- Graph \((\mathcal{X}, \mathcal{W})\)
- Labeled nodes \(\Gamma \subset \mathcal{X}\) and labels \(g : \Gamma \rightarrow \mathbb{R}^k\),
- The \(i^{th}\) class has label vector \(g(x) = e_i = (0, \ldots, 0, 1, 0, \ldots, 0)\).
Graph-based semi-supervised learning

**Given:**

- Graph \((\mathcal{X}, \mathcal{W})\)
- Labeled nodes \(\Gamma \subset \mathcal{X}\) and labels \(g: \Gamma \rightarrow \mathbb{R}^k\),
- The \(i^{th}\) class has label vector \(g(x) = e_i = (0, \ldots, 0, 1, 0, \ldots, 0)\).

**Task:** Extend the labels to the rest of the graph \(\mathcal{X} \setminus \Gamma\).
Graph-based semi-supervised learning

Given:
- Graph \((\mathcal{X}, \mathcal{W})\)
- Labeled nodes \(\Gamma \subseteq \mathcal{X}\) and labels \(g : \Gamma \rightarrow \mathbb{R}^k\),
- The \(i^{th}\) class has label vector \(g(x) = e_i = (0, \ldots, 0, 1, 0, \ldots, 0)\).

Task: Extend the labels to the rest of the graph \(\mathcal{X} \setminus \Gamma\).

Semi-supervised: Goal is to use both the labeled and unlabeled data to get good performance with far fewer labels than required by fully-supervised learning.
Graph-based semi-supervised learning

**Given:**
- Graph $(\mathcal{X}, \mathcal{W})$
- Labeled nodes $\Gamma \subset \mathcal{X}$ and labels $g : \Gamma \to \mathbb{R}^k$,
- The $i^{th}$ class has label vector $g(x) = e_i = (0, \ldots, 0, 1, 0, \ldots, 0)$.

**Task:** Extend the labels to the rest of the graph $\mathcal{X} \setminus \Gamma$.

**Semi-supervised:** Goal is to use both the labeled and unlabeled data to get good performance with far fewer labels than required by fully-supervised learning.

Applications of semi-supervised learning

1. Speech recognition
2. Classification (images, video, website, etc.)
3. Inferring protein structure from sequencing
Why semi-supervised?
Why semi-supervised?
Why semi-supervised?
Why semi-supervised?
Why semi-supervised?
Why semi-supervised?
Laplacian regularization

Laplacian regularized semi-supervised learning solves the Laplace equation

\[
\begin{aligned}
\mathcal{L}u &= 0 \quad \text{in } \mathcal{X} \setminus \Gamma, \\
    u &= g \quad \text{on } \Gamma,
\end{aligned}
\]

where \( u : \mathcal{X} \to \mathbb{R}^k \), and \( \mathcal{L} \) is the graph Laplacian

\[
\mathcal{L}u(x) = \sum_{y \in \mathcal{X}} w_{xy}(u(x) - u(y)).
\]
Laplacian regularization

Laplacian regularized semi-supervised learning solves the Laplace equation

\[
\begin{cases}
    \mathcal{L}u = 0 & \text{in } \mathcal{X} \setminus \Gamma, \\
    u = g & \text{on } \Gamma,
\end{cases}
\]

where \( u : \mathcal{X} \rightarrow \mathbb{R}^k \), and \( \mathcal{L} \) is the graph Laplacian

\[
\mathcal{L}u(x) = \sum_{y \in \mathcal{X}} w_{xy} (u(x) - u(y)).
\]

The label decision for vertex \( x \in \mathcal{X} \) is determined by the largest component of \( u(x) \)

\[
\ell(x) = \arg\max_{j \in \{1, \ldots, k\}} \{u_j(x)\}.
\]

References:

- Original work [Zhu et al., 2003]
- Learning [Zhou et al., 2005, Ando and Zhang, 2007]
- Manifold ranking [He et al., 2006, Zhou et al., 2011, Xu et al., 2011]
Ill-posed with small amount of labeled data

- Graph is \( n = 10^5 \) i.i.d. random variables uniformly drawn from \([0, 1]^2\).
- \( w_{xy} = 1 \) if \(|x - y| < 0.01\) and \( w_{xy} = 0 \) otherwise.
- Two labels: \( g(x) = 0 \) at the Red point and \( g(x) = 1 \) at the Green point.

[Nadler et al., 2009]
Recent work

The low-label rate problem was originally identified in [Nadler 2009].

A lot of recent work has attempted to address this issue with new graph-based classification algorithms at low label rates.

- Higher-order regularization: [Zhou and Belkin, 2011], [Dunlop et al., 2019]
- $p$-Laplace regularization: [Alaoui et al., 2016], [Calder 2018,2019], [Slepcev & Thorpe 2019]
- Re-weighted Laplacians: [Shi et al., 2017], [Calder & Slepcev, 2019]
- Centered kernel method: [Mai & Couillet, 2018]
- Poisson Learning: [Calder, Cook, Thorpe, Slepcev, ICML 2020]
Poisson learning

At low label rates one should replace Laplace learning

\[
\begin{cases}
    \mathcal{L} u = 0, & \text{in } \mathcal{X}, \\
    u = g, & \text{on } \Gamma,
\end{cases}
\]

with Poisson learning

\[\mathcal{L} u(x) = \sum_{y \in \Gamma} (g(y) - \bar{g}) \delta_{xy},\]

subject to \(\sum_{x \in \mathcal{X}} d(x) u(x) = 0\), where \(\bar{g} = \frac{1}{|\Gamma|} \sum_{y \in \Gamma} g(y)\).
Poisson learning

At low label rates one should replace Laplace learning

\[
\begin{aligned}
\mathcal{L}u &= 0, \quad \text{in } \mathcal{X}, \\
u &= g, \quad \text{on } \Gamma,
\end{aligned}
\]

with Poisson learning

\[
\mathcal{L}u(x) = \sum_{y \in \Gamma} (g(y) - \bar{g}) \delta_{xy},
\]

subject to \(\sum_{x \in \mathcal{X}} d(x) u(x) = 0\), where \(\bar{g} = \frac{1}{|\Gamma|} \sum_{y \in \Gamma} g(y)\).

In both cases, the label decision is the same:

\[
\ell(x) = \arg\max_{j \in \{1, \ldots, k\}} \{u_j(x)\}.
\]

Outline

1. Spectral clustering

2. Semi-supervised learning
   - Laplacian regularization
   - Poisson learning

3. Experiments in Python

4. Pointwise consistency for graph Laplacians
Graph-based Clustering and Semi-Supervised Learning

This python package is devoted to efficient implementations of modern graph-based learning algorithms for both semi-supervised learning and clustering. The package implements many popular datasets (currently MNIST, FashionMNIST, cifar-10, and WEBKB) in a way that makes it simple for users to test out new algorithms and rapidly compare against existing methods.

This package reproduces experiments from the paper


Installation

Install with

```
pip install graphlearning
```
Outline

1 Spectral clustering

2 Semi-supervised learning
   - Laplacian regularization
   - Poisson learning

3 Experiments in Python

4 Pointwise consistency for graph Laplacians
Pointwise consistency on random geometric graphs

Let $X_1, X_2, \ldots, X_n$ be a sequence of i.i.d random variables on $\Omega \subset \mathbb{R}^d$ with density $\rho \in C^2(\Omega)$, where $\Omega$ is open and bounded with a smooth boundary, and $\rho \geq \rho_{\text{min}} > 0$.

The random geometric graph Laplacian applied to $u : \Omega \rightarrow \mathbb{R}$ is

$$
\mathcal{L}u(x) = \sum_{i=1}^{n} \eta \left( \frac{|X_i - x|}{\varepsilon} \right) (u(X_i) - u(x)),
$$

where $\varepsilon > 0$ is the connectivity length scale (bandwidth) and $\eta : \mathbb{R} \rightarrow \mathbb{R}$ is smooth, nonnegative and has compact support in $[0, 1]$. 
Pointwise consistency on random geometric graphs

Let $X_1, X_2, \ldots, X_n$ be a sequence of i.i.d random variables on $\Omega \subset \mathbb{R}^d$ with density $\rho \in C^2(\Omega)$, where $\Omega$ is open and bounded with a smooth boundary, and $\rho \geq \rho_{\text{min}} > 0$.

The random geometric graph Laplacian applied to $u : \Omega \to \mathbb{R}$ is

$$\mathcal{L}u(x) = \sum_{i=1}^{n} \eta \left( \frac{|X_i - x|}{\varepsilon} \right) (u(X_i) - u(x)),$$

where $\varepsilon > 0$ is the connectivity length scale (bandwidth) and $\eta : \mathbb{R} \to \mathbb{R}$ is smooth, nonnegative and has compact support in $[0, 1]$.

Today we'll prove that when $u$ is $C^3$ we have

$$\frac{2}{\sigma \eta n \varepsilon^{d+2}} \mathcal{L}u(x) = \rho^{-1} \text{div} \left( \rho^2 \nabla u \right) + O \left( n^{-1/2} \varepsilon^{-1-d/2} \right) + O(\varepsilon).$$

with high probability, provided $B(x, \varepsilon) \subset \Omega$. 
**Discrete to continuum convergence**

**Manifold assumption:** Let $x_1, \ldots, x_n$ be a sequence of i.i.d. random variables with density $\rho$ supported on a $d$-dimensional compact, closed, and connected Riemannian manifold $\mathcal{M}$ embedded in $\mathbb{R}^D$, where $d \ll D$. Fix a finite set of points $\Gamma \subset \mathcal{M}$ and set

$$
\mathcal{X}_n := \left\{ x_1, \ldots, x_n \right\} \cup \left[ \Gamma \right].
$$

Unlabeled

Labeled
Discrete to continuum convergence

**Manifold assumption:** Let \( x_1, \ldots, x_n \) be a sequence of i.i.d. random variables with density \( \rho \) supported on a \( d \)-dimensional compact, closed, and connected Riemannian manifold \( M \) embedded in \( \mathbb{R}^D \), where \( d \ll D \). Fix a finite set of points \( \Gamma \subset M \) and set

\[
X_n := \{x_1, \ldots, x_n\} \cup \Gamma.
\]

**Conjecture**

Let \( n \to \infty \) and \( \varepsilon = \varepsilon_n \to 0 \) so that \( \lim_{n \to \infty} \frac{n \varepsilon^{d+2}}{\log n} = \infty \). Let \( u_n \) be the solution of the Poisson learning problem

\[
\left( \frac{2}{\sigma \varepsilon_n \varepsilon^{d+2}} \right) \mathcal{L} u_n(x) = \sum_{y \in \Gamma} (g(y) - \overline{g})(n \delta_{xy}) \quad \text{for } x \in X_n.
\]

Then with probability one \( u_n \to u \) locally uniformly on \( M \setminus \Gamma \) as \( n \to \infty \), where \( u \in C^\infty(M \setminus \Gamma) \) is the solution of the Poisson equation

\[
- \text{div}_M (\rho^2 \nabla_M u) = \sum_{y \in \Gamma} (g(y) - \overline{g}) \delta_y \quad \text{on } M.
\]
Concentration of measure

Theorem (Bernstein’s inequality)

Let $Y_1, \ldots, Y_n$ be i.i.d. with mean $\mu = \mathbb{E}[Y_i]$ and variance $\sigma^2 = \mathbb{E}[(Y_i - \mathbb{E}[Y_i])^2]$, and assume $|Y_i| \leq M$ almost surely for all $i$. Then for any $t > 0$

\begin{equation}
\mathbb{P} \left( \left| \frac{1}{n} \sum_{i=1}^{n} Y_i - \mu \right| > t \right) \leq 2 \exp \left( - \frac{nt^2}{2\sigma^2 + 4Mt/3} \right).
\end{equation}
Concentration of measure

Theorem (Bernstein’s inequality)

Let \( Y_1, \ldots, Y_n \) be i.i.d. with mean \( \mu = \mathbb{E}[Y_i] \) and variance \( \sigma^2 = \mathbb{E}[(Y_i - \mathbb{E}[Y_i])^2] \), and assume \( |Y_i| \leq M \) almost surely for all \( i \). Then for any \( t > 0 \)

\[
\Pr \left( \left| \frac{1}{n} \sum_{i=1}^{n} Y_i - \mu \right| > t \right) \leq 2 \exp \left( - \frac{nt^2}{2\sigma^2 + 4Mt/3} \right).
\]

Let \( \delta > 0 \) and choose \( t > 0 \) so that \( \delta = 2 \exp \left( - \frac{nt^2}{2\sigma^2 + 4Mt/3} \right) \). Then we get

\[
\left| \frac{1}{n} \sum_{i=1}^{n} Y_i - \mu \right| \leq \sqrt{\frac{2\sigma^2 |\log \frac{\delta}{2}|}{n}} + \frac{4M |\log \frac{\delta}{2}|}{3n}
\]

with probability at least \( 1 - \delta \).
Concentration of measure

**Theorem (Bernstein’s inequality)**

Let $Y_1, \ldots, Y_n$ be i.i.d. with mean $\mu = \mathbb{E}[Y_i]$ and variance $\sigma^2 = \mathbb{E}[(Y_i - \mathbb{E}[Y_i])^2]$, and assume $|Y_i| \leq M$ almost surely for all $i$. Then for any $t > 0$

\[
\mathbb{P} \left( \left| \frac{1}{n} \sum_{i=1}^{n} Y_i - \mu \right| > t \right) \leq 2 \exp \left( - \frac{nt^2}{2\sigma^2 + 4Mt/3} \right).
\]

Let $\delta > 0$ and choose $t > 0$ so that $\delta = 2 \exp \left( - \frac{nt^2}{2\sigma^2 + 4Mt/3} \right)$. Then we get

\[
\left| \frac{1}{n} \sum_{i=1}^{n} Y_i - \mu \right| \leq \sqrt{\frac{2\sigma^2 |\log \frac{\delta}{2}|}{n}} + \frac{4M |\log \frac{\delta}{2}|}{3n}
\]

with probability at least $1 - \delta$. Provided $M \leq C \sqrt{n} \sigma$ we can write

\[
\frac{1}{n} \sum_{i=1}^{n} Y_i = \mu + O \left( \sqrt{\frac{\sigma^2}{n}} \right) \quad \text{w.h.p.}
\]
Proof of Pointwise consistency

\[ Y \mu(x) = \sum_{i=1}^{n} \mathcal{N} \left( \frac{1}{\varepsilon} \left( x - x_i \right) \right) \left( \mu(x) - \mu(x_i) \right) \]

\[ |Y_i| \leq C \varepsilon = M. \]

\[ \sigma^2 = \operatorname{Var}(Y_i) \sim \int_{B(x, \varepsilon)} \mathcal{N} \left( \frac{1}{\varepsilon} \left( y - x \right) \right) \left( \mu(y) - \mu(x) \right)^2 \, dy \]

\[ \sim \varepsilon^{d+2} \]

\[ \sim \varepsilon \]
Bernstein:

\[ \frac{1}{n} \sum_{i=1}^{n} \left( 1 - \frac{y_i - \tau}{\varepsilon} \right) \left( u(y_i) - u(x) \right) e^{-\frac{(y_i - \tau)^2}{2\varepsilon^2}} \]

\[ \mathcal{B}(x, \varepsilon) \]

\[ + O\left( \frac{\sqrt{\varepsilon}}{n} \right) \]

Variance

Taylor expansions in \( Au(x) \) after \( z = \frac{x - \tau}{\varepsilon} \)

\[ Au(x) = \sum_{i=3}^{d} \int_{\mathcal{B}(0,1)} y_i \left( \frac{12}{11} \left( u(x) - u(y) \right) \right) \rho(y) dy \]
\[ f(x+\varepsilon z) = f(x) + \varepsilon \nabla f(x) \cdot z + O(\varepsilon^2) \]

\[ u(x+\varepsilon z) - u(x) = 3 \nabla u(x) \cdot z + \frac{\varepsilon^2}{2} \nabla^2 u(x) \cdot z + O(\varepsilon^3) \]

1. \( \sum_{j} \nabla u(x) \cdot z \) odd function over \( B(\varepsilon_1) \to 0 \)
2. \( \varepsilon^2 \sum_{j} (\nabla^2 u(x) \cdot z)(\nabla u(x) \cdot z) \)
3. \( \varepsilon (x) \frac{\varepsilon^2}{2} \nabla^2 u(x) \cdot z \)
\[
\sum_{i,j=1}^{n} E_{X_i}(x) \cdot E_{X_j}(x) \int_{B(2i)} \eta(1x_1, z) \, dP
\]

\[
= \sum_{i,j=1}^{n} E_{X_i}(x) \cdot E_{X_j}(x) \int_{B(2i)} \eta(1x_1, z) \, dP
\]

\[
= \sum_{i,j=1}^{n} E_{X_i}(x) \cdot E_{X_j}(x)
\]

\[
= \sum_{i,j=1}^{n} \delta_{i} \cdot \delta_{j}
\]

\[
= \delta_{i} \delta_{j}
\]
\[
\frac{1}{2} e(x) \int \eta(1z) z^T \nabla m(x) z \, dz = \nabla^2 m(x) \quad \text{(3)}
\]

\[
= \frac{1}{2} e(x) \sum_{i=1}^{n} u_{x_i} \nabla x_j(x) \int \eta(1z) z_i z_j \, dz \quad B(1z) \\
= \frac{1}{2} e(x) \sum_{i=1}^{n} u_{x_i} \sigma m \\
= \frac{\sigma m}{2} e(x) \Delta m(x) .
\]
Hence

\[ A_{\mathbf{u}}(\mathbf{x}) = 3^{d+2} \delta_{\mathbf{u}} \left( \frac{1}{2} \rho \Delta \mathbf{u} + \nabla \mathbf{u} \cdot \nabla \rho \right) + O(\varepsilon^{d+3}) \]

\[ = \frac{\delta_{\mathbf{u}}}{2} 3^{d+2} \rho^{-1} \left( \varepsilon^2 \Delta \mathbf{u} + 2 \rho \nabla \rho \cdot \nabla \mathbf{u} \right) + O(\varepsilon^{d+3}) \]

\[ = \frac{\delta_{\mathbf{u}}}{2} 3^{d+2} \varepsilon^{-1} \nabla \mathbf{u} \left( \varepsilon^2 \Delta \mathbf{u} + 2 \rho \nabla \rho \cdot \nabla \mathbf{u} \right) + O(\varepsilon^{d+3}) \]

Therefore
\[ \frac{1}{n} \mathcal{L}(\mathbf{x}) = \frac{3}{2} \delta_m^2 \epsilon^{2+2} e^{-1} \text{div} (\epsilon^2 \mathbf{A}_n) + O(n^{-3+3}) \]

and so

\[ \frac{2}{\delta_m^{3+2}} \mathcal{L}(\mathbf{x}) = \epsilon^{-1} \text{div} (\epsilon^2 \mathbf{A}_n) + O(3) \]

\[ + O \left( \sqrt{\frac{1}{n \epsilon^{2+2}}} \right) \]

Bias

Variance.