Random walks and PDEs in graph-based learning

Jeff Calder

School of Mathematics
University of Minnesota

Mathematics of Machine Learning
One World Seminar Series
March 24, 2021

Joint work with:
Brendan Cook (UMN), Peter J. Olver (UMN), Dejan Slepčev (CMU), Matthew Thorpe (Manchester), and Katrina Yezzi-Woodley (UMN)

Research supported by NSF-DMS 1713691, 1944925, and the Alfred P. Sloan Foundation

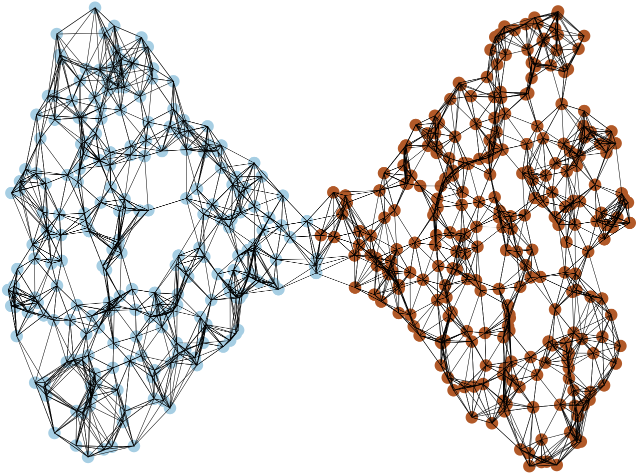
Outline

Outline

Graph-based learning

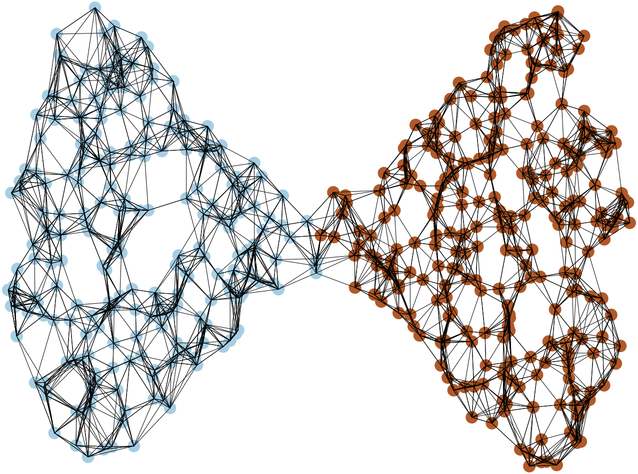
Let $(\mathcal{X}, \mathcal{W})$ be a graph.

- $\mathcal{X} \subset \mathbb{R}^d$ 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.

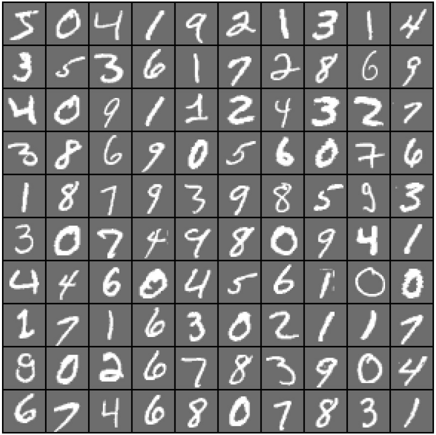


Some common graph-based learning tasks

1. Clustering
   - Grouping similar datapoints
2. Semi-supervised learning.
   - Propagating labels on a graph.



MNIST (70,000 28×28 pixel images of digits 0-9)



- Each image is a datapoint

$$x \in \mathbb{R}^{28\times 28} = \mathbb{R}^{784}.$$

- Geometric weights:

$$w_{xy} = \eta\left(\frac{|x-y|}{\varepsilon}\right)$$

- k-nearest neighbor graph:

$$w_{xy} = \eta\left(\frac{|x-y|}{\varepsilon_k(x)}\right)$$

Clustering MNIST

https://divamgupta.com

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 $g(x) = e_i = (0, , \dots, 0, 1, 0, \dots, 0)$.

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

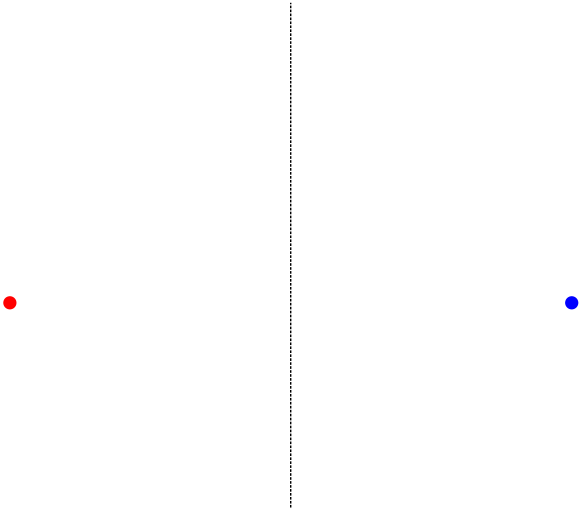
Semi-supervised: Goal is to use both the labeled and unlabeled data to get good performance with far fewer labels than required by fully-supervised learning.

Applications of semi-supervised learning

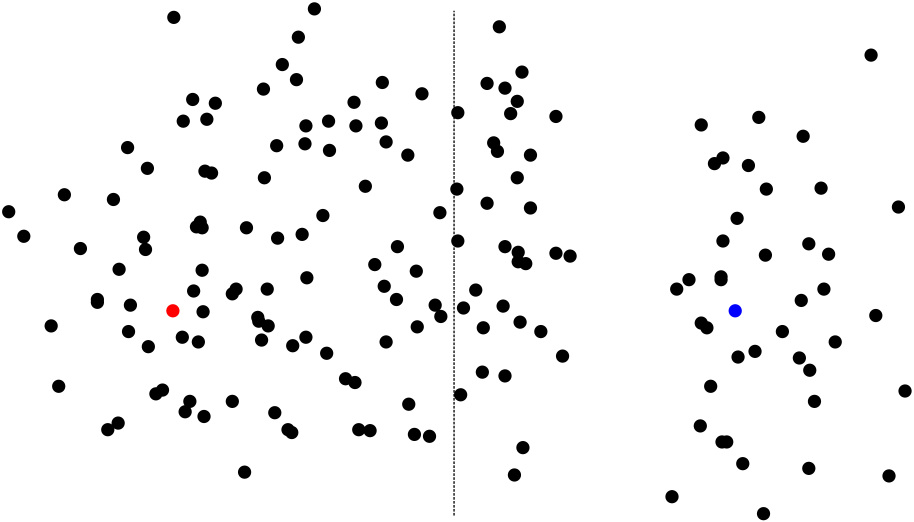
1. Speech recognition
2. Classification (images, video, website, etc.)
3. Inferring protein structure from sequencing

Why semi-supervised?

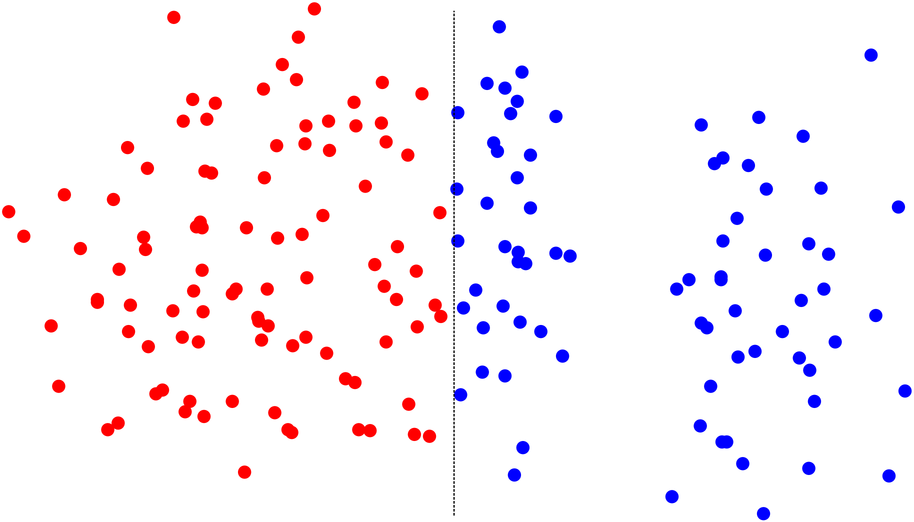
Why semi-supervised?



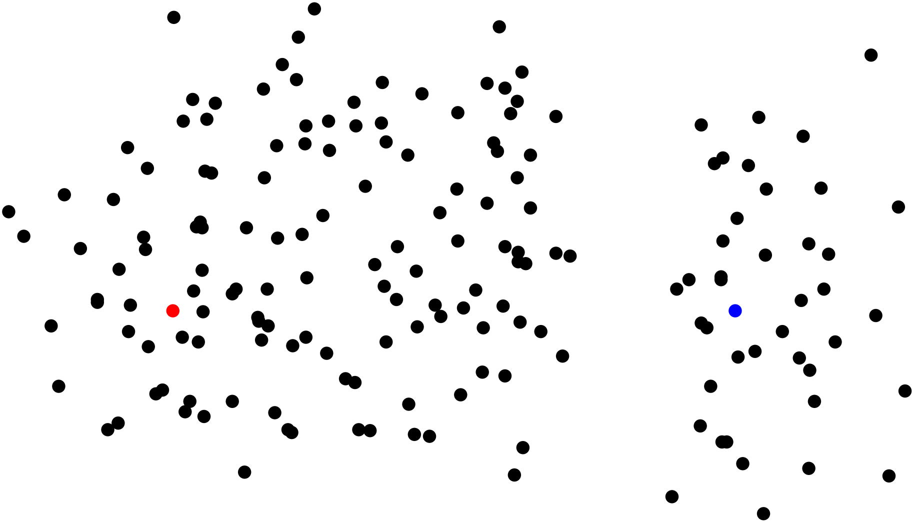
Why semi-supervised?



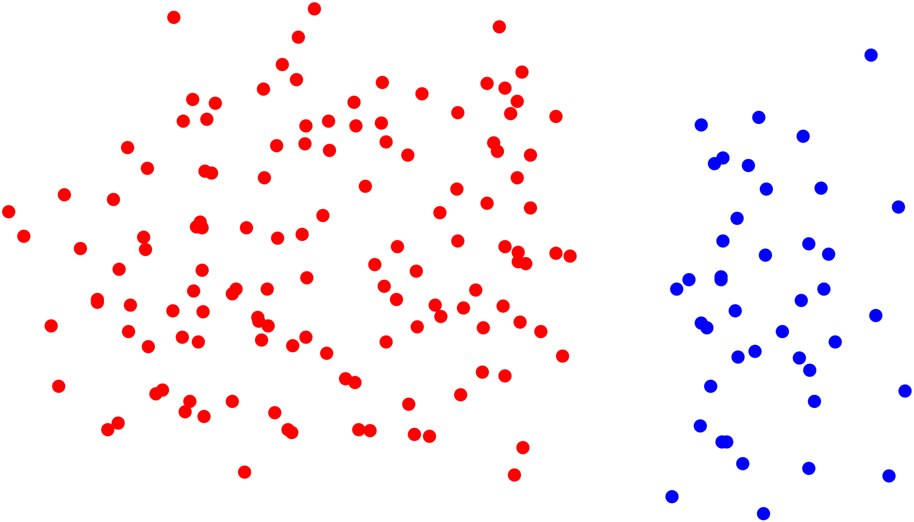
Why semi-supervised?



Why semi-supervised?



Why semi-supervised?



Laplacian regularization

Laplacian regularized semi-supervised learning solves the Laplace equation

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

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

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

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

$$\ell(x) = \operatorname*{argmax}_{j \in \{1,\ldots,k\}} \{u_j(x)\}.$$

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]

Label propagation

The solution of Laplace learning satisfies

$$\mathcal{L}u(x) = \sum_{y \in \mathcal{X}} w_{xy}(u(x) - u(y)) = 0. \qquad (y \in \mathcal{X} \setminus \Gamma)$$

Re-arranging, we see that u satisfies the mean-value property

$$u(x) = \frac{\sum_{y \in \mathcal{X}} w_{xy} u(y)}{\sum_{y \in \mathcal{X}} w_{xy}}.$$

Label propagation [Zhu 2005] iterates

$$u^{k+1}(x) = \frac{\sum_{y \in \mathcal{X}} w_{xy} u^k(y)}{\sum_{y \in \mathcal{X}} w_{xy}}.$$

and at convergence is equivalent to Laplace learning.

Variational interpretation

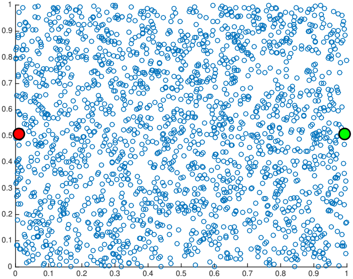
Laplace learning is equivalent to the variational problem

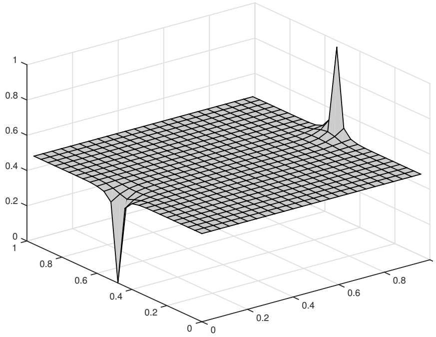
$$\min_{u:\mathcal{X}\to\mathbb{R}^k} \left\{ \sum_{x,y\in\mathcal{X}} w_{xy}|u(x)-u(y)|^2 \,:\, u(x)=g(x) \text{ for all } x\in\Gamma \right\}.$$

Many soft-constrained versions have been proposed

$$\min_{u:\mathcal{X}\to\mathbb{R}^k} \left\{ \sum_{x,y\in\mathcal{X}} w_{xy}|u(x)-u(y)|^2 + \lambda \sum_{x\in\Gamma} \ell(u(x),g(x)) \right\}.$$

Ill-posed with small amount of labeled data

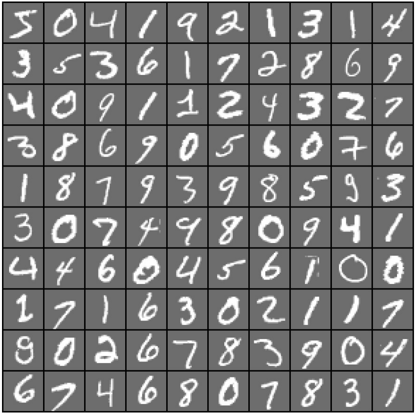




- Graph is $n = 10^5$ i.i.d. random variables uniformly drawn from $[0, 1]^2$.
- $w_{xy} = 1$ if $|x - y| < 0.01$ and $w_{xy} = 0$ otherwise.
- Two labels: $g(x) = 0$ at the Red point and $g(x) = 1$ at the Green point.

[Nadler et al., 2009]

MNIST (70,000 28×28 pixel images of digits 0-9)



[Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE, 86(11):2278-2324, November 1998.]

Laplace learning on MNIST at low label rates

# Labels per class	**1**	**2**	**3**	**4**	**160**
Laplace Learning	16.1 (6.2)	28.2 (10.3)	42.0 (12.4)	57.8 (12.3)	97.0 (0.1)
Nearest Neighbor	65.4 (5.2)	74.2 (3.3)	77.8 (2.6)	80.7 (2.0)	92.4 (0.2)

- Average accuracy over 100 trials with standard deviation in brackets.
- Nearest neighbor is geodesic graph-nearest neighbor.

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.

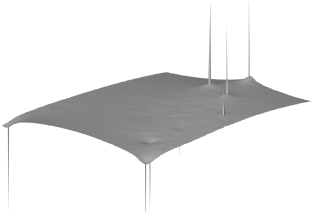
- Higher-order regularization: [Zhou and Belkin, 2011], [Dunlop et al., 2019]
- p-Laplace regularization: [Alaoui et al., 2016], [Calder 2018,2019], [Slepcev & Thorpe 2019]
- Re-weighted Laplacians: [Shi et al., 2017], [Calder & Slepcev, 2019]
- Centered kernel method: [Mai & Couillet, 2018]

While we have lots of new models, the problem with Laplace learning at low label rates was still not well-understood.

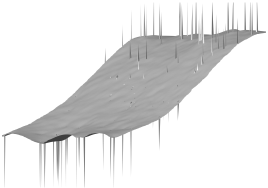
Outline

Spikes in Laplacian regularized learning

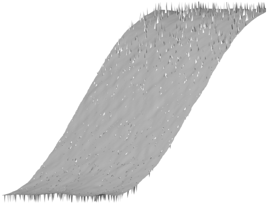
Label function: $g(x) = \cos(x_1)$.



10 labels



100 labels



1000 labels

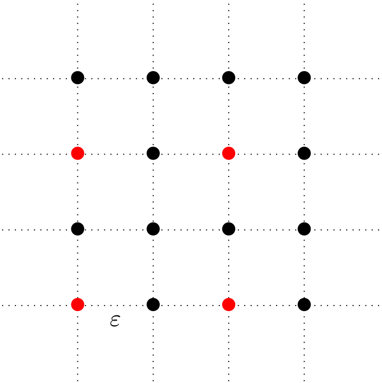
Q1 How many labels do we need to ensure that spikes do not form?

Q2 Why does Laplace learning perform poorly at low label rates?

- Are the spikes too localized? Do they propagate information globally?

Q3 How should we propagate labels in a stable and informative way?

A related numerical analysis problem



Discrete Laplace equation:

$$\begin{cases} \Delta_\varepsilon u_\varepsilon = 0 & \text{in } \mathbb{Z}^d_\varepsilon \setminus \mathbb{Z}^d_{m\varepsilon} \\ u_\varepsilon = g & \text{on } \mathbb{Z}^d_{m\varepsilon}. \end{cases}$$

$$\Delta_\varepsilon u(x) = \sum_{i=1}^{d} \sum_{b=\pm 1} (u(x+b\varepsilon e_i) - u(x)).$$

Dirichlet energy:

$$J_\varepsilon u(x) = \sum_{i=1}^{d} \sum_{b=\pm 1} (u(x+b\varepsilon e_i) - u(x))^2.$$

Label rate is $\beta = m^{-d}$. By energy balancing arguments

$$\text{Energy of smooth part } \sim 2d\varepsilon^2, \qquad \text{Energy of spikes } \sim 2d\beta |u_\varepsilon - g|^2_\infty.$$

Conjecture: $|u_\varepsilon - g|_\infty \sim \frac{C\varepsilon}{\sqrt{\beta}}$. We can prove $|u_\varepsilon - g|_\infty \leq \frac{C\varepsilon}{\beta^{\frac{1}{2}+\frac{1}{d}}}$.

Random geometric graph

Random Geometric Graph: Assume the vertices of the graph are

$$\mathcal{X}_n = \{x_1, \ldots, x_n\}$$

where $x_1, \ldots, x_n$ is a sequence of i.i.d. random variables on $\Omega \subset \mathbb{R}^d$ with positive density ρ, and the weights are given by

$$w_{xy} = \eta\left(\frac{|x-y|}{\varepsilon}\right), \tag{1}$$

where $\eta : [0,\infty) \to [0,1]$ is smooth with compact support.

Pointwise consistency of the graph Laplacian

The graph Laplacian is defined as

$$\mathcal{L}u(x) = \sum_{y \in \mathcal{X}_n} \eta\left(\frac{|x-y|}{\varepsilon}\right)(u(x) - u(y)).$$

In the large data $n \to \infty$ and sparse graph $\varepsilon \to 0$ limit, $\mathcal{L}$ is consistent with

$$\Delta_\rho u = -\rho^{-1}\mathrm{div}(\rho^2 \nabla u).$$

In particular, it is a standard result [Hein et al., 2007] that

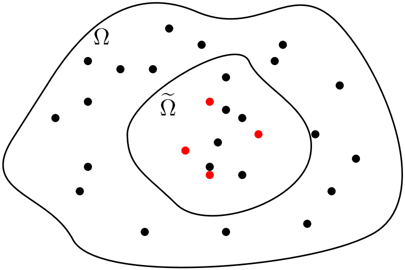
$$\left|\frac{1}{n\varepsilon^{d+2}}\mathcal{L}u(x) - \sigma_\eta \Delta_\rho u(x)\right| \leq C(\lambda + \varepsilon)$$

holds for any $u \in C^3(\Omega)$ with probability at least $1 - 2\exp\left(-cn\varepsilon^{d+2}\lambda^2\right)$.

Note: The density ρ acts as an edge detector encouraging sharp changes in u between clusters.

Model for labeled data

Model 1. Let $\beta \in (0,1]$ and $\widetilde{\Omega} \subset\subset \Omega$. Each $x_i \in \widetilde{\Omega}$ is selected as training data independently with probability β. Let Γ_n = training data.



The Laplacian learning problem is

$$(2) \quad \begin{cases} \mathcal{L}u_n(x) = 0, & \text{if } x \in \mathcal{X}_n \setminus \Gamma_n \\ u_n(x) = g(x), & \text{if } x \in \Gamma_n, \end{cases}$$

where $g : \Omega \to \mathbb{R}$ is Lipschitz and

$$\mathcal{X}_n = \{x_1, x_2, \ldots, x_n\}.$$

Main result

The continuum PDE is

$$\begin{cases} \text{div}(\rho^2 \nabla u) = 0 & \text{in } \Omega \setminus \widetilde{\Omega} \\ u = g & \text{on } \widetilde{\Omega} \\ \nabla u \cdot \mathbf{n} = 0 & \text{on } \partial\Omega. \end{cases} \tag{3}$$

Theorem (C.-Slepcev-Thorpe, 2020)

Let $u_n : \mathcal{X}_n \to \mathbb{R}$ *be the solution of* (2), *and let* $u \in C^3(\overline{\Omega})$ *be the solution of* (3). *If* $\beta \geq \varepsilon^2$ *and* $\varepsilon \leq \lambda \leq c$ *then*

$$\max_{x \in \mathcal{X}_n} |u_n(x) - u(x)| \leq C \left(\frac{\varepsilon}{\sqrt{\beta}} \log\left(\frac{\sqrt{\beta}}{\varepsilon} \right) + \lambda \right) \tag{4}$$

holds with probability at least $1 - Cn\exp\left(-cn\varepsilon^{d+2}\lambda^2\right)$.

"Proof:" Let $X_0, X_1, X_2, \ldots$ be a random walk on $\mathcal{X}_n$ with transition probabilities

$$\mathbb{P}\left(X_{k+1} = y \,|\, X_k = x\right) = \frac{w_{xy}}{\sum_{z \in \mathcal{X}_n} w_{xz}}.$$

Define the stopping time

$$\tau = \inf\{k \geq 0 \,:\, X_k \in \Gamma_n\}.$$

Then $u_n(X_k)$ is a martingale up to the stopping time, and so

$$u_n(x) = \mathbb{E}[g(X_\tau) \,|\, X_0 = x].$$

Therefore

$$|u_n(x) - g(x)| \leq \mathrm{Lip}(g)\mathbb{E}[|X_\tau - X_0| \,|X_0 = x].$$

Each step has probability $O(\beta)$ of hitting a labeled point, so

$$\tau \leq \frac{C}{\beta} \quad \text{with high probability (w.h.p.)}$$

In k steps, the walk moves at most $C\varepsilon\sqrt{k}$ from X_0, w.h.p., and so

$$|X_\tau - X_0| \leq \frac{C\varepsilon}{\sqrt{\beta}} \quad \text{w.h.p.} \quad \square$$

The negative result

Theorem (C.-Slepcev-Thorpe, 2020)

Assume that $\beta = \beta_n \to 0^+$ *and* $\varepsilon = \varepsilon_n \to 0^+$ *satisfy*

(5) $$\beta_n \ll \varepsilon_n^2, \quad \textit{and} \quad n\varepsilon_n^d \gg \log(n).$$

Then, with probability one, the sequence u_n *is pre-compact in* TL^2 *and any convergent subsequence converges to a constant.*

Summary: Laplace learning propagates labels well when

$$\text{Label rate } = \beta \gg \varepsilon^2.$$

Below this label rate, spikes form and the solution is degenerate.

Error on MNIST

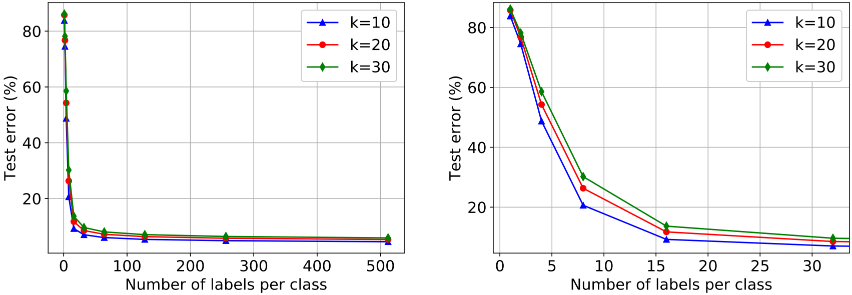


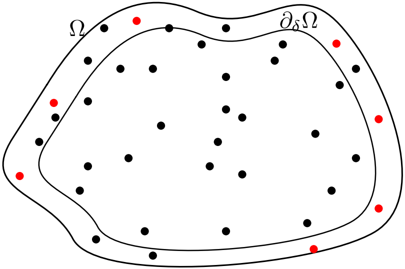
Figure: Error plots for MNIST experiment showing testing error versus number of labels, averaged over 100 trials.

Fits very well to the error rate $\beta^{-1/2}$.

A numerical analysis-inspired model

Model 2. Let $\beta \in (0,1)$, $\delta \in (0,\varepsilon]$. Each $x_i \in \partial_\delta \Omega$ is selected as training data independently with probability β, where

$$\partial_\delta \Omega = \{x \in \Omega : \mathrm{dist}(x, \partial\Omega) < \delta\}.$$



Here, the continuum PDE is

$$\begin{cases} \mathrm{div}(\rho^2 \nabla u) = 0 & \text{in } \Omega \\ u = g & \text{on } \partial\Omega. \end{cases} \tag{6}$$

J. Calder, D. Slepčev, D., and M. Thorpe. **Rates of convergence for Laplacian semi-supervised learning with low label rates.** *arXiv:2006.02765*, 2020.

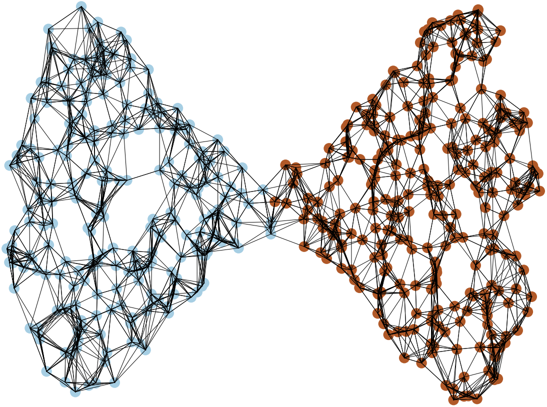
Outline

Random Walk Perspective

Suppose $u : \mathcal{X} \to \mathbb{R}^k$ solves the Laplace learning equation

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

The random walk interpretation $u(x) = \mathbb{E}[g(X_\tau) \mid X_0 = x]$ can help us understand the degeneracy at low label rates.



Random Walk Perspective

Suppose $u : \mathcal{X} \to \mathbb{R}^k$ solves the Laplace learning equation

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

The random walk interpretation $u(x) = \mathbb{E}[g(X_\tau) \,|\, X_0 = x]$ can help us understand the degeneracy at low label rates.

MNIST Classification Example

# Labels per class	**1**	**2**	**3**	**4**	**160**
Laplace Learning	16.1 (6.2)	28.2 (10.3)	42.0 (12.4)	57.8 (12.3)	97.0 (0.1)
Average # Steps	220,409	22,980	16,403	14,145	51

The random walk perspective

At low label rates, the random walk mixes before hitting a label, and the distribution of the random walker is close to the invariant distribution π, given by

$$\pi(x) = \frac{d(x)}{\sum_{y\in\mathcal{X}} d(x),}$$

where the degree is $d(x) = \sum_{y\in\mathcal{X}} w_{xy}$. Thus, the solution of Laplace learning is

$$u(x) = \mathbb{E}[g(X_\tau) \,|\, X_0 = x] \approx \frac{\sum_{y\in\Gamma} d(y)g(y)}{\sum_{y\in\Gamma} d(y)} =: c \in \mathbb{R}^k.$$

To test this, we considered a shifted label decision

$$\ell(x) = \operatorname*{argmax}_{j\in\{1,\dots,k\}} \{u_j(x) - c_j\}.$$

# Labels/class	**1**	**2**	**3**	**4**	**5**
Laplace	16.1 (6.2)	28.2 (10)	42.0 (12)	57.8 (12)	69.5 (12)
Shift Laplace	88.3 (5.7)	92.6 (2.4)	94.3 (1.4)	94 (1.5)	95 (0.6)

A related Poisson equation

If the solution to Laplace learning u is roughly constant $u \approx c$, then at a labeled node $x \in \Gamma$ we can compute

$$\begin{aligned}\mathcal{L}u(x) &= \sum_{y\in\mathcal{X}} w_{xy}(u(x)-u(y)) \\ &\approx \sum_{y\in\mathcal{X}} w_{xy}(g(x)-c) \quad \text{(since } u\approx c\text{)} \\ &= d(x)(g(x)-c).\end{aligned}$$

At unlabeled nodes we have $\mathcal{L}u = 0$. Thus, u approximately solves

$$\boxed{\mathcal{L}u(x) = \sum_{y\in\Gamma} d(y)(g(y)-c)\delta_{xy}, \quad c = \frac{\sum_{y\in\Gamma} d(y)g(y)}{\sum_{y\in\Gamma} d(y)},}$$

where $\delta_{xy} = 1$ if $x = y$ and $\delta_{xy} = 0$ otherwise.

Poisson learning

We propose to replace Laplace learning

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

with Poisson learning

$$\boxed{\mathcal{L}u(x) = \sum_{y \in \Gamma} (g(y) - \overline{g})\delta_{xy},}$$

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

In both cases, the label decision is the same:

$$\ell(x) = \operatorname*{argmax}_{j \in \{1,\ldots,k\}} \{u_j(x)\}.$$

Poisson learning

We propose to replace Laplace learning

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

with Poisson learning

$$\boxed{\mathcal{L}u(x) = \sum_{y\in\Gamma}(g(y) - \overline{g})\delta_{xy},}$$

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

For Poisson learning, unbalanced class sizes can be incorporated:

$$\ell(x) = \operatorname*{argmax}_{j\in\{1,\dots,k\}} \left\{ \frac{p_j}{n_j} u_j(x) \right\},$$

p_j = Fraction of data in class j

n_j = Fraction of training data from class j.

The random walk interpretation

Let X_0^x, X_1^x, X_2^x be a random walk on $\mathcal{X}$ starting from $x \in \mathcal{X}$, and define

$$u_T(x) := \mathbb{E}\left[\sum_{k=0}^{T} \frac{1}{d(x)} \sum_{y\in\Gamma} (g(y) - \overline{g}) \mathbb{1}_{\{X_k^y = x\}}\right], \quad \text{where } \overline{g} = \frac{1}{|\Gamma|}\sum_{y\in\Gamma} g(y).$$

Theorem (C.-Cook-Thorpe-Slepcev, 2020)

For every $T \geq 0$ we have

$$u_{T+1}(x) = u_T(x) + \frac{1}{d(x)}\left(\sum_{y\in\Gamma}(g(y) - \overline{g})\delta_{xy} - \mathcal{L}u_T(x)\right).$$

If the graph G is connected and the Markov chain induced by the random walk is aperiodic, then $u_T \to u$ as $T \to \infty$, where $u : \mathcal{X} \to \mathbb{R}$ is the solution of

$$\mathcal{L}u(x) = \sum_{y\in\Gamma}(g(y) - \overline{g})\delta_{xy},$$

satisfying $\sum_{x\in\mathcal{X}} d(x)u(x) = 0$.

The variational interpretation

Consider the variational problem

$$\min_{u\in\ell_0^2(\mathcal{X})}\left\{\sum_{x,y\in\mathcal{X}} w_{xy}|u(x)-u(y)|^2-\sum_{x\in\Gamma}(g(x)-\overline{g})\cdot u(x)\right\}, \tag{7}$$

where $\overline{g}=\frac{1}{|\Gamma|}\sum_{x\in\Gamma} g(x)$.

Theorem (C.-Cook-Thorpe-Slepcev, 2020)

Assume G is connected. Then there exists a unique minimizer $u\in\ell_0^2(\mathcal{X})$ of (7), *and furthermore, u satisfies the Poisson equation*

$$\mathcal{L}u(x)=\sum_{y\in\Gamma}(g(y)-\overline{g})\delta_{xy}.$$

J. Calder, B. Cook, M. Thorpe, and D. Slepčev. **Poisson Learning: Graph based semi-supervised learning at very low label rates.** *International Conference on Machine Learning (ICML), PMLR 119:1306–1316*, 2020.

The continuum perspective

Manifold assumption: Let $x_1, \ldots, x_n$ be a sequence of i.i.d. random variables with density ρ 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

$$\mathcal{X}_n := \underbrace{\{x_1, \ldots, x_n\}}_{\text{Unlabeled}} \cup \underbrace{\Gamma}_{\text{Labeled}}.$$

Conjecture

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

$$\left(\frac{2}{\sigma_\eta n \varepsilon_n^{d+2}}\right) \mathcal{L} u_n(x) = \sum_{y\in\Gamma} (g(y) - \overline{g})(n\delta_{xy}) \quad \text{for } x \in \mathcal{X}_n.$$

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

$$-\mathrm{div}_{\mathcal{M}} \left(\rho^2 \nabla_{\mathcal{M}} u\right) = \sum_{y\in\Gamma} (g(y) - \overline{g})\delta_y \quad \text{on } \mathcal{M}.$$

Spectral representation

Theorem

The solution of the Poisson learning equation

$$\mathcal{L}u(x) = \sum_{y\in\Gamma}(g(y) - \overline{g})\delta_{xy}$$

is given by

$$u(x) = \sum_{y\in\Gamma}\sum_{k=2}^{n}(g(y) - \overline{g})\lambda_k^{-1}v_k(x)v_k(y),$$

where $v_1, v_2, \dots, v_n$ *are the normalized eigenvectors of* $\mathcal{L}$*, with corresponding eigenvalues* $0 = \lambda_1 < \lambda_2 \leq \cdots \leq \lambda_n$.

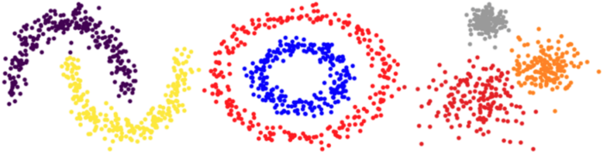
Proof of the conjecture reduces to spectral convergence. We proved $O(\varepsilon)$ spectral convergence rates in the $C^{0,1}$ sense:

J. Calder, N. Garcia Trillos, and M. Lewicka, **Lipschitz regularity of graph Laplacians on random data clouds**, *arXiv:2007.06679*, 2020.

Outline

GraphLearning Python Package

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

Calder, Cook, Thorpe, Slepcev. Poisson Learning: Graph Based Semi-Supervised Learning at Very Low Label Rates., Proceedings of the 37th International Conference on Machine Learning, PMLR 119:1306-1316, 2020.

Installation

Install with

```
pip install graphlearning
```

https://github.com/jwcalder/GraphLearning

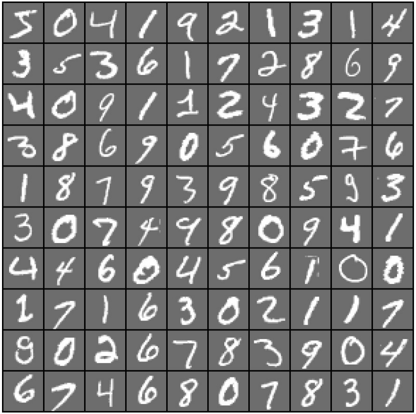
Algorithmic details

Algorithm 1 Poisson Learning

Input: $\mathbf{W}, \mathbf{F} = [y_1, y_2, \ldots, y_m], T$
$\mathbf{D} \leftarrow \mathrm{diag}(\mathbf{W}\mathbb{1})$
$\mathbf{L} \leftarrow \mathbf{D} - \mathbf{W}$
$\mathbf{c} \leftarrow \frac{1}{m}\mathbf{F}\mathbb{1}$
$\mathbf{B} \leftarrow [\mathbf{F} - \mathbf{c}, \textbf{zeros}(k, n-m)]$
$\mathbf{U} \leftarrow \textbf{zeros}(n, k)$
for $i = 1$ **to** T **do**
    $\mathbf{U} \leftarrow \mathbf{U} + \mathbf{D}^{-1}(\mathbf{B}^T - \mathbf{L}\mathbf{U})$
end for
$\ell_i \leftarrow \operatorname*{argmax}_{1 \leq j \leq k} \mathbf{U}_{ij}$
return: $\boldsymbol{\ell} := [\ell_1, \ell_2, \ldots, \ell_n]$

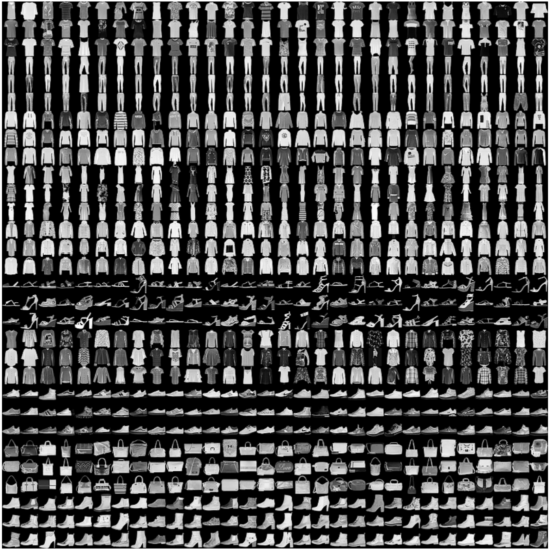
We only need about $T = 100$ iterations on MNIST, FashionMNIST, CIFAR-10, to get good results. CPU Time: 4 seconds on CPU, 1 second on GPU.

MNIST (70,000 28×28 pixel images of digits 0-9)



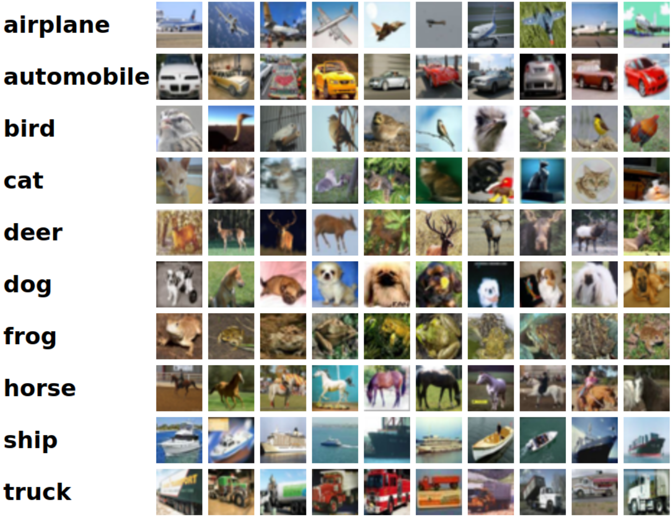
[Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition." Proceedings of the IEEE, 86(11):2278-2324, November 1998.]

FashionMNIST (70,000 28×28 images of fashion items)



[Xiao, Han, Kashif Rasul, and Roland Vollgraf. "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms." arXiv:1708.07747 (2017).]

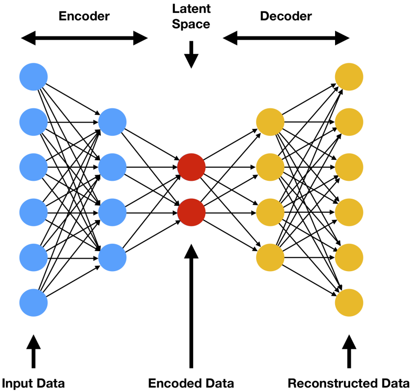
CIFAR-10



[Krizhevsky, Alex, and Geoffrey Hinton. "Learning multiple layers of features from tiny images." (2009).]

Autoencoders

For each dataset, we build the graph by training autoencoders.



www.compthree.com

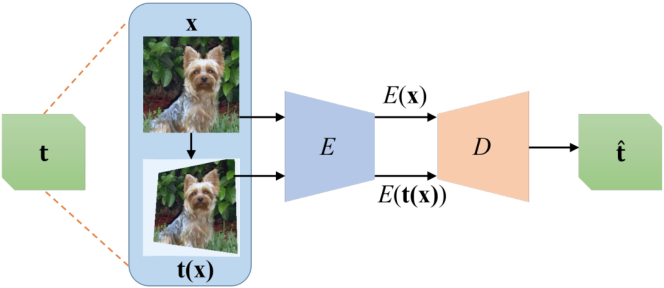
Autoencoders are "Nonlinear versions of PCA"

Building graphs from autoencoders

For MNIST and FashionMNIST, we use a 4-layer variational autoencoder with 30 latent variables:

[Kingma and Welling. Auto-encoding variational Bayes. ICML 2014]

For CIFAR-10, we use the autoencoding framework from [Zhang et al. AutoEncoding Transformations (AET), CVPR 2019] with 12,288 latent variables.



First comparison

We compared against many other graph-based learning algorithms

- Laplace/Label propagation: [Zhu et al., 2003]
- Graph nearest neighbor (using Dijkstra)
- Lazy random walks: [Zhou et al., 2004]
- Mutli-class MBO: [Garcia-Cardona et al., 2014]
- Centered kernel method: [Mai & Couillet, 2018]
- Sparse Label Propagation: [Jung et al., 2016]
- Weighted Nonlocal Laplacian (WNLL): [Shi et al., 2017]
- p-Laplace regularization: [Flores et al. 2019]

MNIST results

Table: Average (standard deviation) classification accuracy over 100 trials.

# Labels per class	**1**	**2**	**3**	**4**	**5**
Laplace/LP	16.1 (6.2)	28.2 (10.3)	42.0 (12.4)	57.8 (12.3)	69.5 (12.2)
Nearest Neighbor	65.4 (5.2)	74.2 (3.3)	77.8 (2.6)	80.7 (2.0)	82.1 (2.0)
Random Walk	66.4 (5.3)	76.2 (3.3)	80.0 (2.7)	82.8 (2.3)	84.5 (2.0)
MBO	19.4 (6.2)	29.3 (6.9)	40.2 (7.4)	50.7 (6.0)	59.2 (6.0)
Centered Kernel	19.1 (1.9)	24.2 (2.3)	28.8 (3.4)	32.6 (4.1)	35.6 (4.6)
Sparse Label Prop.	14.0 (5.5)	14.0 (4.0)	14.5 (4.0)	18.0 (5.9)	16.2 (4.2)
WNLL	55.8 (15.2)	82.8 (7.6)	90.5 (3.3)	93.6 (1.5)	94.6 (1.1)
p-Laplace	72.3 (9.1)	86.5 (3.9)	89.7 (1.6)	90.3 (1.6)	91.9 (1.0)
Poisson	**90.2 (4.0)**	**93.6 (1.6)**	**94.5 (1.1)**	**94.9 (0.8)**	**95.3 (0.7)**

FashionMNIST results

Table: Average (standard deviation) classification accuracy over 100 trials.

# Labels per class	**1**	**2**	**3**	**4**	**5**
Laplace/LP	18.4 (7.3)	32.5 (8.2)	44.0 (8.6)	52.2 (6.2)	57.9 (6.7)
Nearest Neighbor	46.6 (4.7)	53.5 (3.6)	57.2 (3.0)	59.3 (2.6)	61.1 (2.8)
Random Walk	49.0 (4.4)	55.6 (3.8)	59.4 (3.0)	61.6 (2.5)	63.4 (2.5)
MBO	15.7 (4.1)	20.1 (4.6)	25.7 (4.9)	30.7 (4.9)	34.8 (4.3)
Centered Kernel	11.8 (0.4)	13.1 (0.7)	14.3 (0.8)	15.2 (0.9)	16.3 (1.1)
Sparse Label Prop.	14.1 (3.8)	16.5 (2.0)	13.7 (3.3)	13.8 (3.3)	16.1 (2.5)
WNLL	44.6 (7.1)	59.1 (4.7)	64.7 (3.5)	67.4 (3.3)	70.0 (2.8)
p-Laplace	54.6 (4.0)	57.4 (3.8)	65.4 (2.8)	68.0 (2.9)	68.4 (0.5)
Poisson	**60.8 (4.6)**	**66.1 (3.9)**	**69.6 (2.6)**	**71.2 (2.2)**	**72.4 (2.3)**

Compare to clustering result of 67.2% [McConville et al., 2019]

CIFAR-10 results

Table: Average (standard deviation) classification accuracy over 100 trials.

# Labels per class	**1**	**2**	**3**	**4**	**5**
Laplace/LP	10.4 (1.3)	11.0 (2.1)	11.6 (2.7)	12.9 (3.9)	14.1 (5.0)
Nearest Neighbor	33.1 (4.3)	37.3 (4.1)	39.7 (3.0)	41.7 (2.8)	43.0 (2.5)
Random Walk	36.4 (4.9)	42.0 (4.4)	45.1 (3.3)	47.5 (2.9)	49.0 (2.6)
MBO	14.2 (4.1)	19.3 (5.2)	24.3 (5.6)	28.5 (5.6)	33.5 (5.7)
Centered Kernel	15.4 (1.6)	16.9 (2.0)	18.8 (2.1)	19.9 (2.0)	21.7 (2.2)
Sparse Label Prop.	11.8 (2.4)	12.3 (2.4)	11.1 (3.3)	14.4 (3.5)	11.0 (2.9)
WNLL	16.6 (5.2)	26.2 (6.8)	33.2 (7.0)	39.0 (6.2)	44.0 (5.5)
p-Laplace	26.0 (6.7)	35.0 (5.4)	42.1 (3.1)	48.1 (2.6)	49.7 (3.8)
Poisson	**40.7 (5.5)**	**46.5 (5.1)**	**49.9 (3.4)**	**52.3 (3.1)**	**53.8 (2.6)**

Compare to clustering result of 41.2% [Mukherjee et al., ClusterGAN, CVPR 2019].

Volume constrained semi-supervised learning

ELSEVIER

Journal of Computational Physics

Volume 354, 1 February 2018, Pages 288-310

Auction dynamics: A volume constrained MBO scheme

Matt Jacobs, Ekaterina Merkurjev, Selim Esedoḡlu

Show more

https://doi.org/10.1016/j.jcp.2017.10.036

Get rights and content

Classification results can be improved by incorporating prior knowledge of class sizes through volume constraints.

PoissonMBO: Volume constrained Poisson learning

Observation 1: The Poisson learning iteration with a fixed time step

$$u_{T+1}(x) = u_T(x) + dt\left(\sum_{y\in\Gamma}(g(y)-\overline{g})\delta_{ij} - \mathcal{L}u_T(x)\right)$$

is **volume preserving**. That is $\sum_{x\in\mathcal{X}} u_{T+1}(x) = \sum_{x\in\mathcal{X}} u_T(x)$.

Observation 2: We can easily perform a volume constrained label projection

$$\ell(x_i) = \operatorname*{argmax}_{j\in\{1,\dots,k\}} \{s_j u_j(x)\}.$$

We adjust the weights s_j to grow/shrink each region to achieve the correct class sizes.

Named after the Merriman-Bence-Osher (MBO) scheme for curvature motion, which has been used before in graph-based learning [Garcia, et al., 2014, Jacobs et al., 2018].

MNIST results

Table: Average (standard deviation) classification accuracy over 100 trials.

# Labels per class	**1**	**2**	**3**	**4**	**5**
Laplace/LP	16.1 (6.2)	28.2 (10.3)	42.0 (12.4)	57.8 (12.3)	69.5 (12.2)
WNLL	55.8 (15.2)	82.8 (7.6)	90.5 (3.3)	93.6 (1.5)	94.6 (1.1)
p-Laplace	72.3 (9.1)	86.5 (3.9)	89.7 (1.6)	90.3 (1.6)	91.9 (1.0)
VolumeMBO	89.9 (7.3)	95.6 (1.9)	96.2 (1.2)	96.6 (0.6)	96.7 (0.6)
Poisson	90.2 (4.0)	93.6 (1.6)	94.5 (1.1)	94.9 (0.8)	95.3 (0.7)
PoissonMBO	**96.5 (2.6)**	**97.2 (0.1)**	**97.2 (0.1)**	**97.2 (0.1)**	**97.2 (0.1)**
# Labels per class	**10**	**20**	**40**	**80**	**160**
Laplace/LP	91.3 (3.7)	95.8 (0.6)	96.5 (0.2)	96.8 (0.1)	97.0 (0.1)
WNLL	95.6 (0.5)	96.1 (0.3)	96.3 (0.2)	96.4 (0.1)	96.3 (0.1)
p-Laplace	94.0 (0.8)	95.1 (0.4)	95.5 (0.1)	96.0 (0.2)	96.2 (0.1)
VolumeMBO	96.9 (0.2)	97.0 (0.1)	97.1 (0.1)	97.2 (0.1)	**97.3 (0.1)**
Poisson	95.9 (0.4)	96.3 (0.3)	96.6 (0.2)	96.8 (0.1)	96.9 (0.1)
PoissonMBO	**97.2 (0.1)**	**97.2 (0.1)**	**97.2 (0.1)**	**97.2 (0.1)**	97.2 (0.1)

FashionMNIST results

Table: Average (standard deviation) classification accuracy over 100 trials.

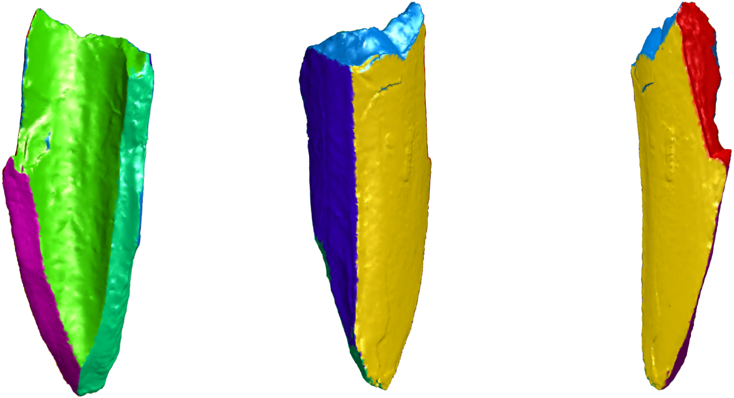
# Labels per class	**1**	**2**	**3**	**4**	**5**
Laplace/LP	18.4 (7.3)	32.5 (8.2)	44.0 (8.6)	52.2 (6.2)	57.9 (6.7)
WNLL	44.6 (7.1)	59.1 (4.7)	64.7 (3.5)	67.4 (3.3)	70.0 (2.8)
p-Laplace	54.6 (4.0)	57.4 (3.8)	65.4 (2.8)	68.0 (2.9)	68.4 (0.5)
VolumeMBO	54.7 (5.2)	61.7 (4.4)	66.1 (3.3)	68.5 (2.8)	70.1 (2.8)
Poisson	60.8 (4.6)	66.1 (3.9)	69.6 (2.6)	71.2 (2.2)	72.4 (2.3)
PoissonMBO	**62.0 (5.7)**	**67.2 (4.8)**	**70.4 (2.9)**	**72.1 (2.5)**	**73.1 (2.7)**
# Labels per class	**10**	**20**	**40**	**80**	**160**
Laplace/LP	70.6 (3.1)	76.5 (1.4)	79.2 (0.7)	80.9 (0.5)	**82.3 (0.3)**
WNLL	74.4 (1.6)	77.6 (1.1)	79.4 (0.6)	80.6 (0.4)	81.5 (0.3)
p-Laplace	73.0 (0.9)	76.2 (0.8)	78.0 (0.3)	79.7 (0.5)	80.9 (0.3)
VolumeMBO	74.4 (1.5)	77.4 (1.0)	79.5 (0.7)	**81.0 (0.5)**	82.1 (0.3)
Poisson	75.2 (1.5)	77.3 (1.1)	78.8 (0.7)	79.9 (0.6)	80.7 (0.5)
PoissonMBO	**76.1 (1.4)**	**78.2 (1.1)**	**79.5 (0.7)**	80.7 (0.6)	81.6 (0.5)

CIFAR-10 results

Table: Average (standard deviation) classification accuracy over 100 trials.

# Labels per class	**1**	**2**	**3**	**4**	**5**
Laplace/LP	10.4 (1.3)	11.0 (2.1)	11.6 (2.7)	12.9 (3.9)	14.1 (5.0)
WNLL	16.6 (5.2)	26.2 (6.8)	33.2 (7.0)	39.0 (6.2)	44.0 (5.5)
p-Laplace	26.0 (6.7)	35.0 (5.4)	42.1 (3.1)	48.1 (2.6)	49.7 (3.8)
VolumeMBO	38.0 (7.2)	46.4 (7.2)	50.1 (5.7)	53.3 (4.4)	55.3 (3.8)
Poisson	40.7 (5.5)	46.5 (5.1)	49.9 (3.4)	52.3 (3.1)	53.8 (2.6)
PoissonMBO	**41.8 (6.5)**	**50.2 (6.0)**	**53.5 (4.4)**	**56.5 (3.5)**	**57.9 (3.2)**
# Labels per class	**10**	**20**	**40**	**80**	**160**
Laplace/LP	21.8 (7.4)	38.6 (8.2)	54.8 (4.4)	62.7 (1.4)	66.6 (0.7)
WNLL	54.0 (2.8)	60.3 (1.6)	64.2 (0.7)	66.6 (0.6)	68.2 (0.4)
p-Laplace	56.4 (1.8)	60.4 (1.2)	63.8 (0.6)	66.3 (0.6)	68.7 (0.3)
VolumeMBO	59.2 (3.2)	61.8 (2.0)	63.6 (1.4)	64.5 (1.3)	65.8 (0.9)
Poisson	58.3 (1.7)	61.5 (1.3)	63.8 (0.8)	65.6 (0.6)	67.3 (0.4)
PoissonMBO	**61.8 (2.2)**	**64.5 (1.6)**	**66.9 (0.8)**	**68.7 (0.6)**	**70.3 (0.4)**

Application: Segmenting broken bone fragments

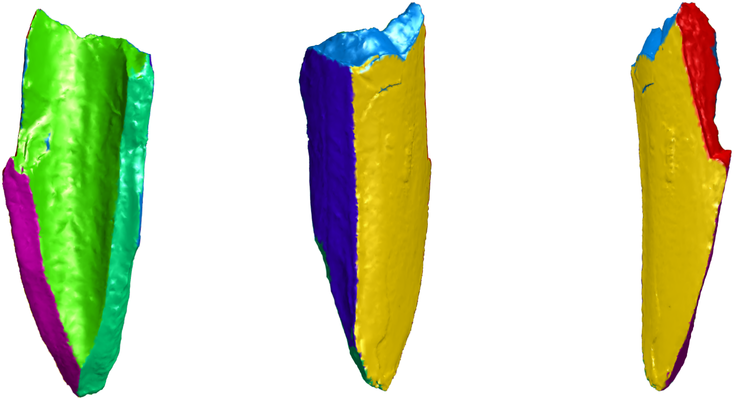


AMAAZE consortium for mathematics and anthropology: `https://amaaze.umn.edu/`

Main collaborators: Peter J. Olver and Katrina Yezzi-Woodley (Anthropology)

REU students: Math: David Floeder, Anthropology: Paige Cody, Chloe Siewert
Math Graduate students: Riley O'Neill, Brendan Cook

Application: Segmenting broken bone fragments

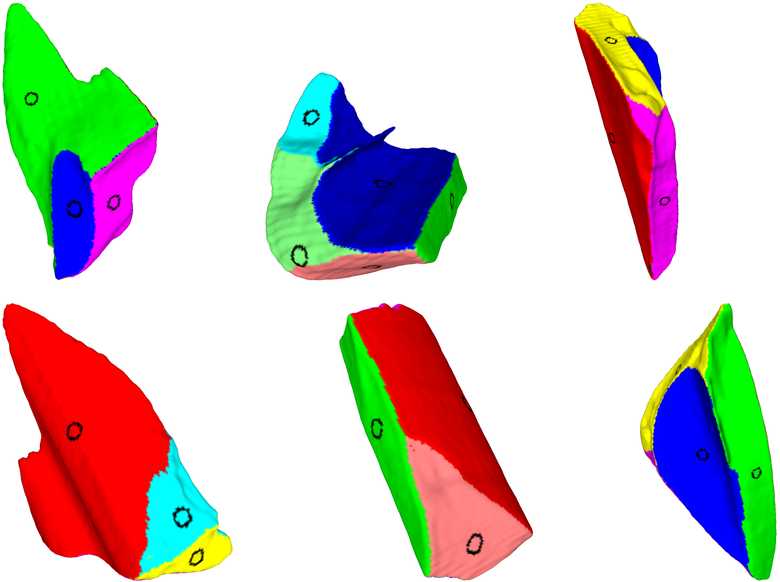


Graph-based clustering with weights

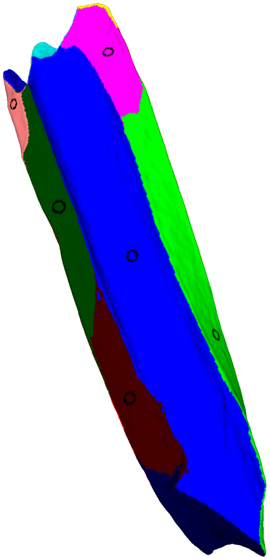
$$w_{ij} = \exp\left(-C|\mathbf{n}_i - \mathbf{n}_j|^p\right).$$

between nearby points on the mesh, where $\mathbf{n}_i$ is the outward normal vector at vertex i.

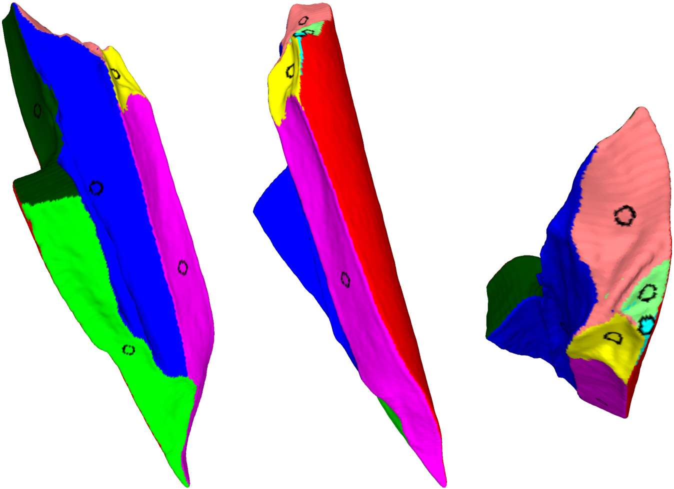
Mesh Segmentation via Poisson Learning



Mesh Segmentation via Poisson Learning

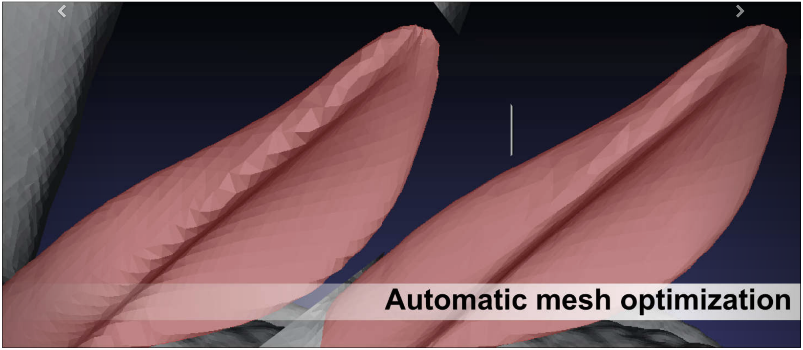


Mesh Segmentation via Poisson Learning



AMAAZE MeshLab plugins

MeshLab the open source system for processing and editing 3D triangular meshes.
It provides a set of tools for editing, cleaning, healing, inspecting, rendering, texturing and converting meshes. It offers features for processing raw data produced by 3D digitization tools/devices and for preparing models for 3D printing.



https://amaaze.umn.edu/software

Outline

Current/Future Work

1. Poisson learning
    - Directed graphs, clustering
    - Continuum limit
    - Asymptotic consistency
2. Rates of convergence for p-Laplacian regularization
    - Including other graphs, like stochastic block models
3. Graph convolutional networks for semi-supervised learning
    - [Kipf & Welling, ICLR 2017]
4. Few-shot semi-supervised learning
    - H. Huang, J. Zhang, J. Zhang, Q. Wu, C. Xu. **PTN: A Poisson Transfer Network for Semi-supervised Few-shot Learning.** To appear in proceedings of AAAI 2021 (arXiv preprint:2012.10844).

References

References:

1. J. Calder, D. Slepčev, D., and M. Thorpe. **Rates of convergence for Laplacian semi-supervised learning with low label rates.** *arXiv:2006.02765*, 2020.
2. J. Calder, B. Cook, M. Thorpe, and D. Slepčev. **Poisson Learning: Graph based semi-supervised learning at very low label rates.** *International Conference on Machine Learning (ICML), PMLR 119:1306–1316*, 2020.

Code: `https://github.com/jwcalder/GraphLearning` (pip install graphlearning)