Nonlinear PDE continuum limits in data science and machine learning

Jeff Calder

School of Mathematics
University of Minnesota

University of Wisconsin PDE & GA seminar
Monday, April 9, 2018

This research was supported by NSF-DMS grants 0914567, 1500829 and 1713691.

Outline

Outline

Motivating example: Google Goggles

Motivating example: Google Goggles

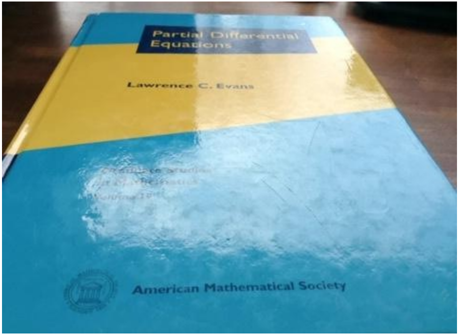


Figure: Query image

Motivating example: Google Goggles

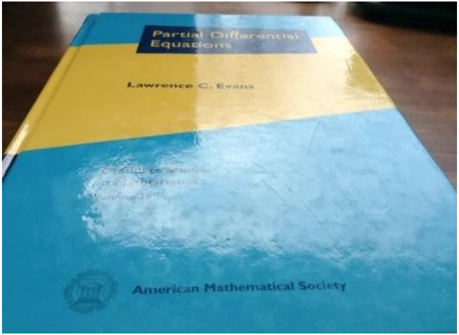


Figure: Query image

Figure: Retrieved images

Multi-query image retrieval

Problem: Find images in a dataset S that are similar to multiple query images.

Pareto method: "Solve" the multi-objective optimization problem

$$\arg\min_{I \in S}(\mathrm{dist}(I, Q_1), \ldots, \mathrm{dist}(I, Q_d)).$$

Query 1

Query 2

Multi-query image retrieval

Problem: Find images in a dataset S that are similar to multiple query images.

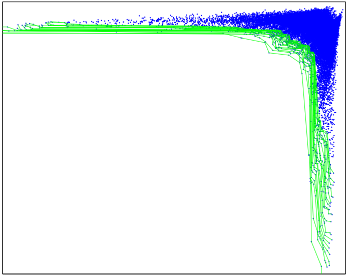
Pareto method: "Solve" the multi-objective optimization problem

$$\arg\min_{I \in S} (\mathrm{dist}(I, Q_1), \ldots, \mathrm{dist}(I, Q_d)).$$

Query 1

Query 2

Pareto points:



Multi-objective optimization

How do we solve the multi-objective optimization problem

$$\operatorname*{arg\,min}_{I \in S} \, (f_1(I), \ldots, f_d(I))?$$

Multi-objective optimization

How do we solve the multi-objective optimization problem

$$\arg\min_{I\in S} \, (f_1(I), \ldots, f_d(I))?$$

Basic approach:

1. Choose some weights $\alpha_i \in [0,1]$ with $\sum_{i=1}^d \alpha_i = 1$ and define

$$f_\alpha(I) = \alpha_1 f_1(I) + \alpha_2 f_2(I) + \cdots + \alpha_d f_d(I).$$

Multi-objective optimization

How do we solve the multi-objective optimization problem

$$\operatorname*{arg\,min}_{I\in S}\,(f_1(I),\dots,f_d(I))?$$

Basic approach:

1. Choose some weights $\alpha_i \in [0,1]$ with $\sum_{i=1}^d \alpha_i = 1$ and define

$$f_\alpha(I) = \alpha_1 f_1(I) + \alpha_2 f_2(I) + \cdots + \alpha_d f_d(I).$$

2. Solve the scalarized optimization problem

$$\operatorname*{arg\,min}_{I\in S}\, f_\alpha(I).$$

Multi-objective optimization

How do we solve the multi-objective optimization problem

$$\operatorname*{arg\,min}_{I\in S}\,(f_1(I),\dots,f_d(I))?$$

Basic approach:

1. Choose some weights $\alpha_i \in [0,1]$ with $\sum_{i=1}^d \alpha_i = 1$ and define

$$f_\alpha(I) = \alpha_1 f_1(I) + \alpha_2 f_2(I) + \cdots + \alpha_d f_d(I).$$

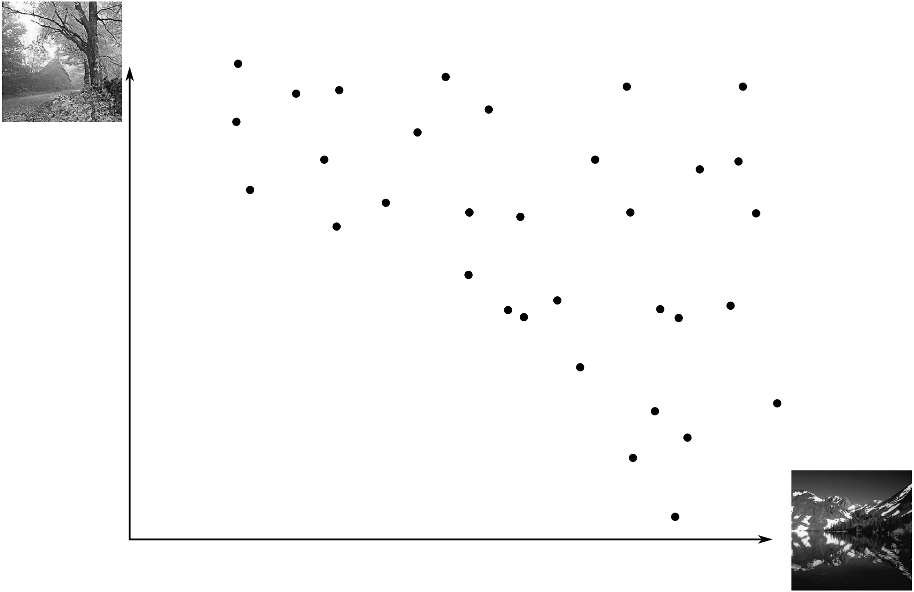
2. Solve the scalarized optimization problem

$$\operatorname*{arg\,min}_{I\in S}\, f_\alpha(I).$$

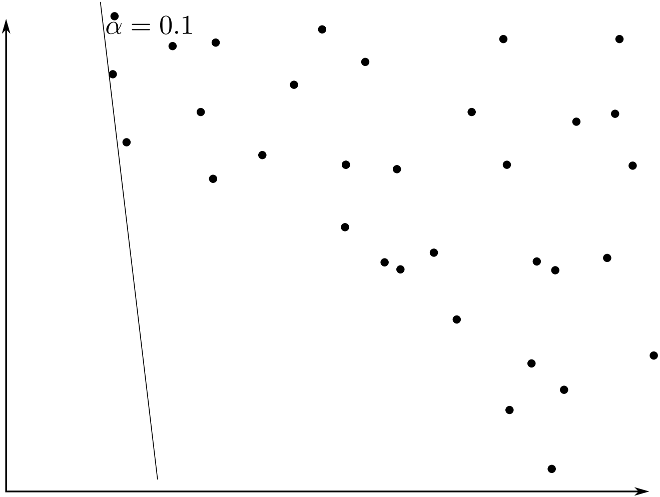
Problems:

1. Difficult to choose weights
2. Ignores relevant solutions

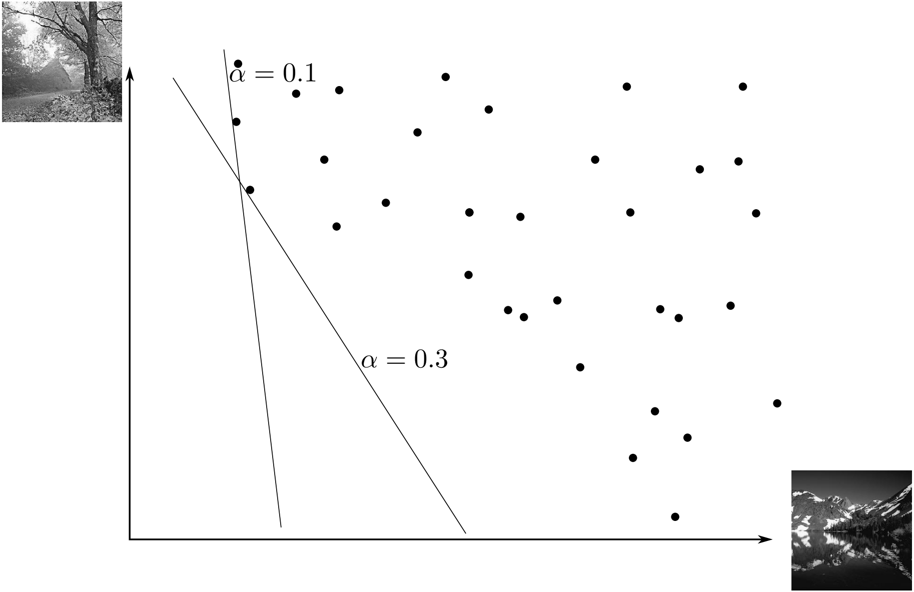
Basic approach



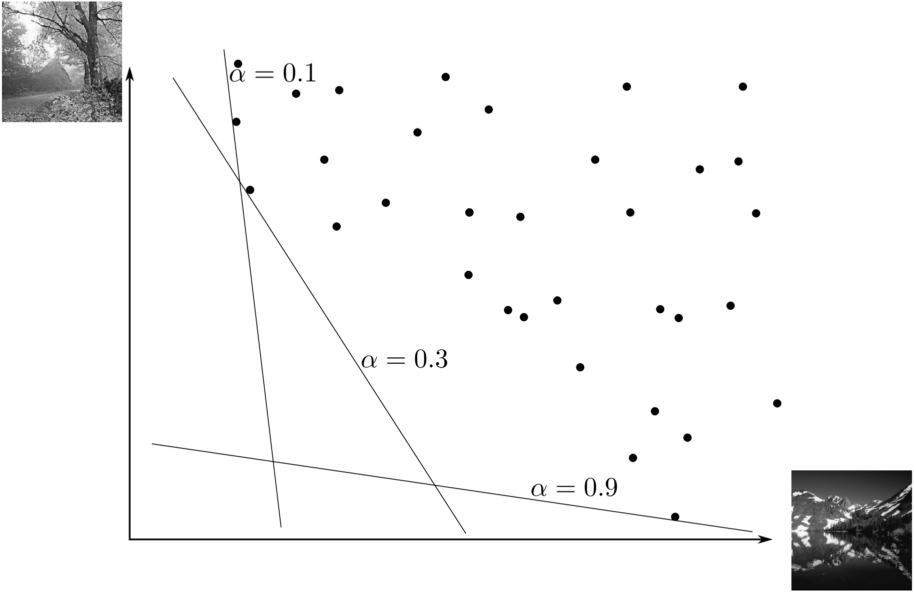
Basic approach



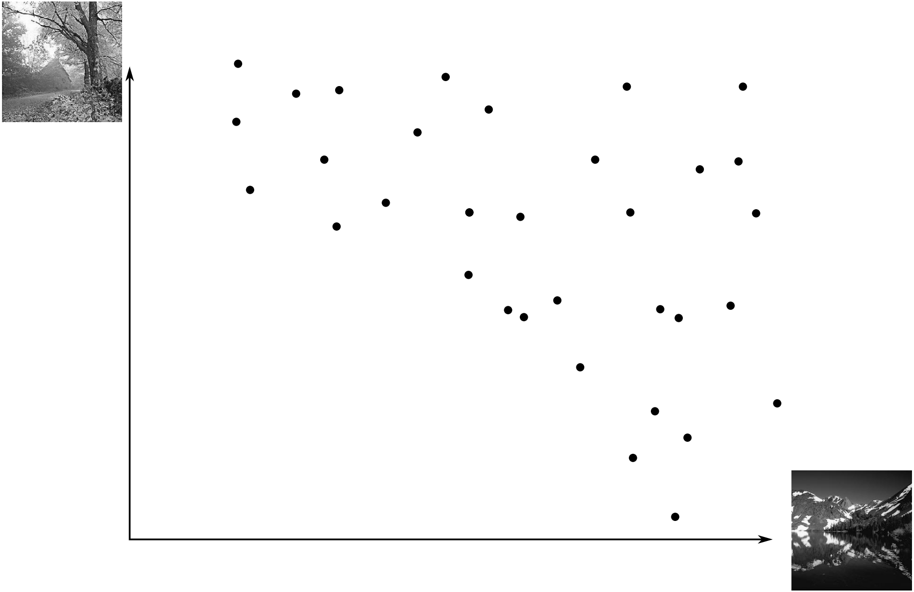
Basic approach



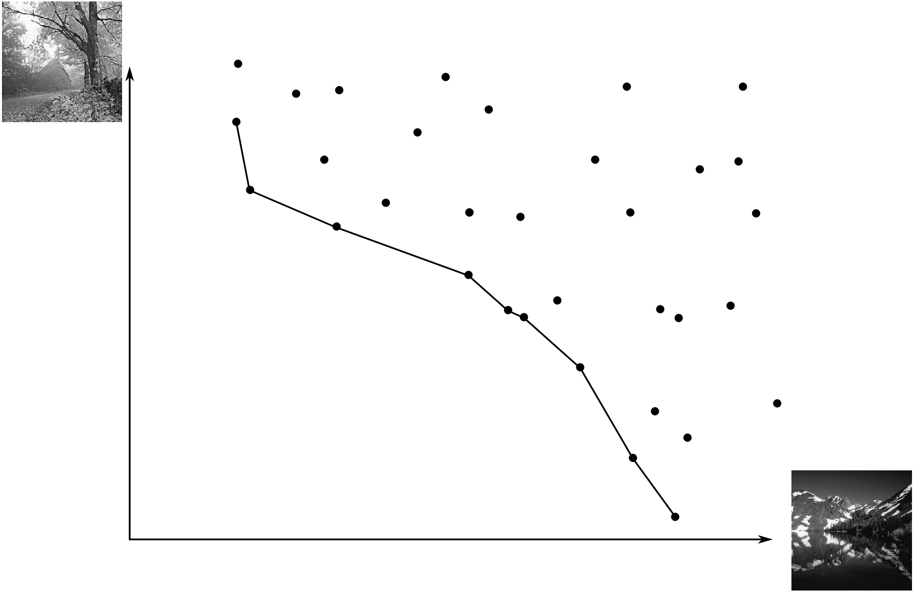
Basic approach



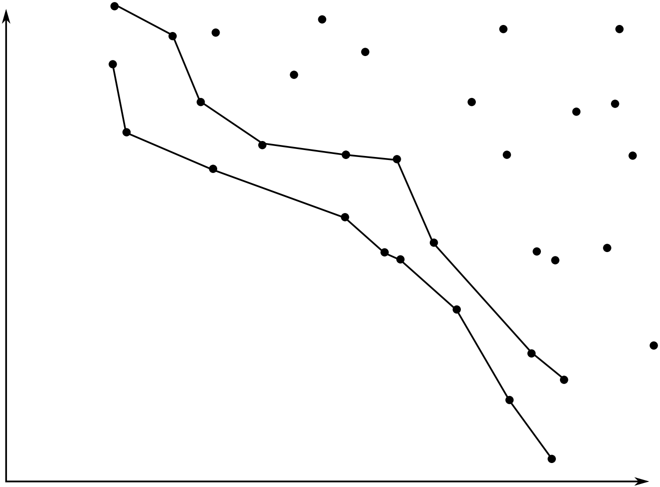
Nondominated solutions



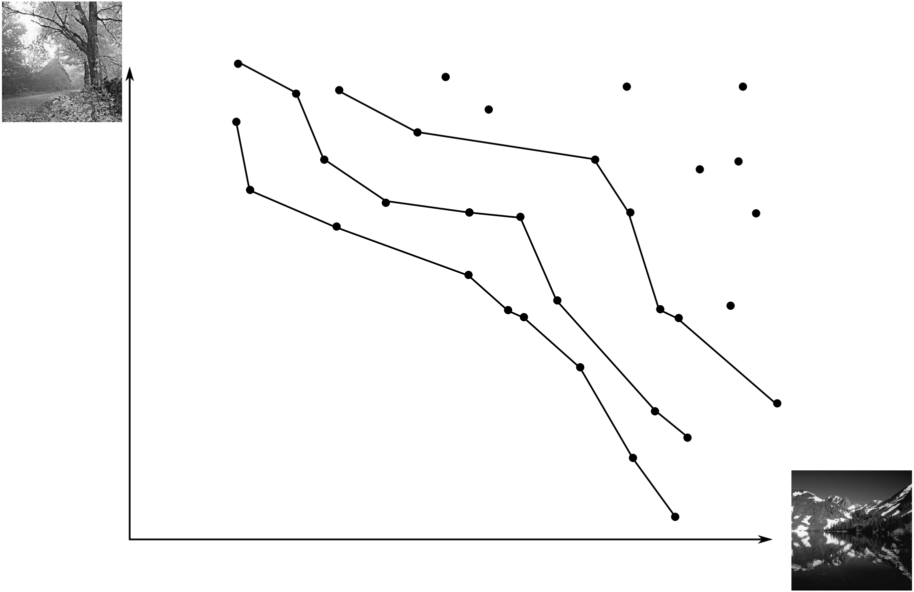
Nondominated solutions



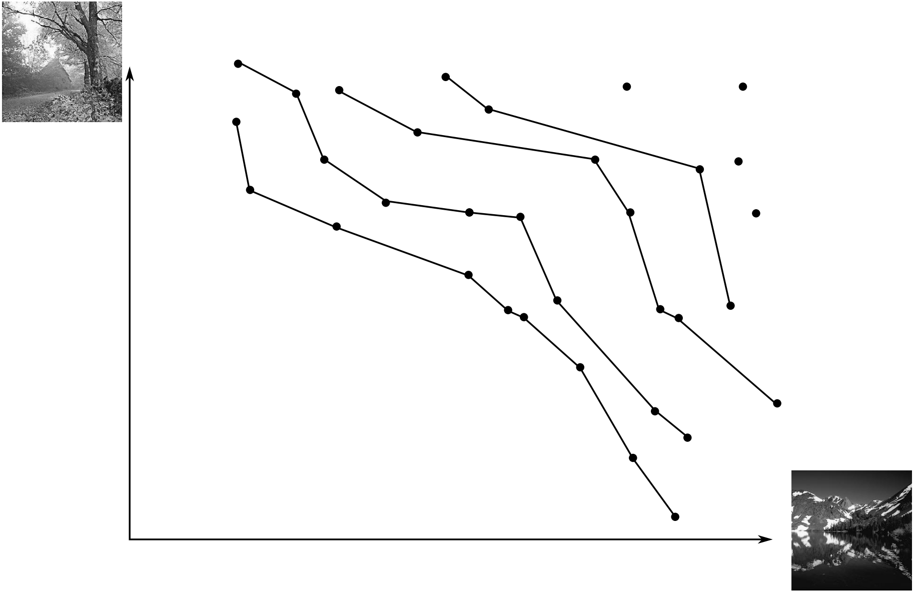
Nondominated solutions



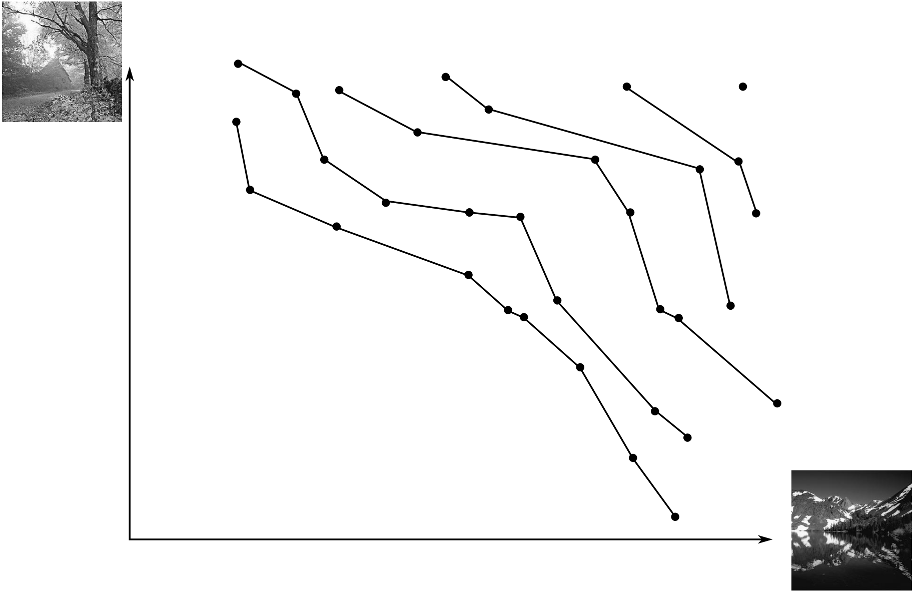
Nondominated solutions



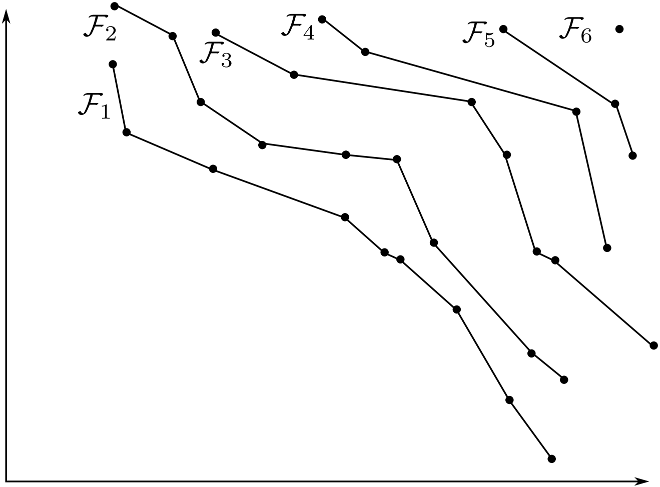
Nondominated solutions



Nondominated solutions



Nondominated solutions



Multi-query image retrieval

First Pareto front:

Query 1

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

Query 2

Hsiao, K.-J., Calder, J., and Hero III, A. O. (2015). Pareto-depth for multiple-query image retrieval. *IEEE Transactions on Image Processing*, 24(2):583–594.

Nondominated sorting

Let $X_1, \ldots, X_n$ be points in $\mathbb{R}^d$ and set $S = \{X_1, \ldots, X_n\}$.

Define the partial order

$$x \leqq y \iff x_i \leq y_i \text{ for all } i \in \{1, \ldots, d\}.$$

Nondominated sorting

Let $X_1, \ldots, X_n$ be points in $\mathbb{R}^d$ and set $S = \{X_1, \ldots, X_n\}$.

Define the partial order

$$x \leqq y \iff x_i \leq y_i \text{ for all } i \in \{1, \ldots, d\}.$$

Definition

Nondominated sorting is the process of arranging S into layers $\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3, \ldots,$ defined by

$$\mathcal{F}_1 = \text{Minimal elements of } S,$$

$$\mathcal{F}_k = \text{Minimal elements of } S \setminus (\mathcal{F}_1 \cup \cdots \cup \mathcal{F}_{k-1}).$$

Applications

Multi-objective optimization

- Genetic algorithms [Deb et al., 2002]
- Gene selection and ranking [Hero, 2003]
- Database systems [Papadias et al., 2005]
- Anomaly detection [Hsiao et al., 2012]
- Image retrieval [Hsiao et al., 2015]

Applications

Multi-objective optimization

- Genetic algorithms [Deb et al., 2002]
- Gene selection and ranking [Hero, 2003]
- Database systems [Papadias et al., 2005]
- Anomaly detection [Hsiao et al., 2012]
- Image retrieval [Hsiao et al., 2015]

Combinatorics and probability

- Longest monotone subsequences [Ulam, 1961]
- Longest chain in Euclidean space [Hammersley, 1972]
- Patience sorting [Aldous and Diaconis, 1999]
- Young Tableaux [Viennot, 1984]
- Graph theory [Lou and Sarrafzadeh, 1993]
- Polynuclear growth (crystals) [Prähofer and Spohn, 2000]

Applications

Multi-objective optimization

- Genetic algorithms [Deb et al., 2002]
- Gene selection and ranking [Hero, 2003]
- Database systems [Papadias et al., 2005]
- Anomaly detection [Hsiao et al., 2012]
- Image retrieval [Hsiao et al., 2015]

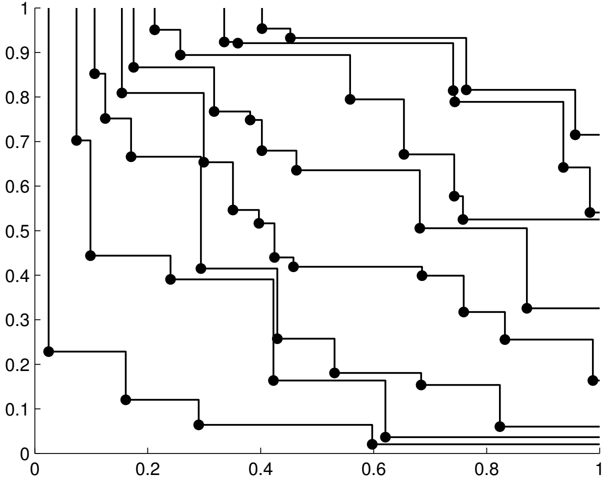
Combinatorics and probability

- Longest monotone subsequences [Ulam, 1961]
- Longest chain in Euclidean space [Hammersley, 1972]
- Patience sorting [Aldous and Diaconis, 1999]
- Young Tableaux [Viennot, 1984]
- Graph theory [Lou and Sarrafzadeh, 1993]
- Polynuclear growth (crystals) [Prähofer and Spohn, 2000]

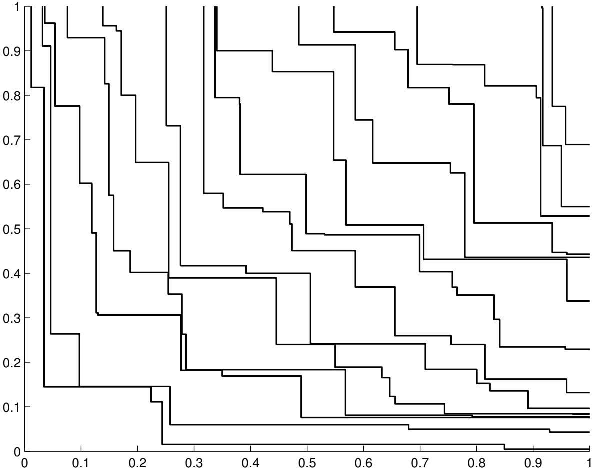
Other applications

- Molecular biology [Pevzner, 2000]
- Integrated circuit design [Adhar, 2007]

Demo: 50 Random samples

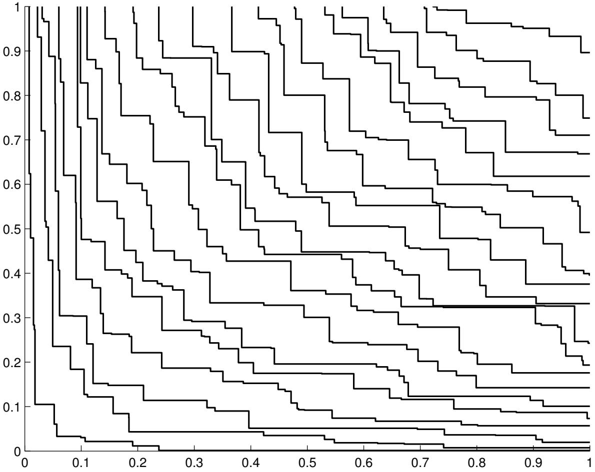


Demo: Uniform distribution



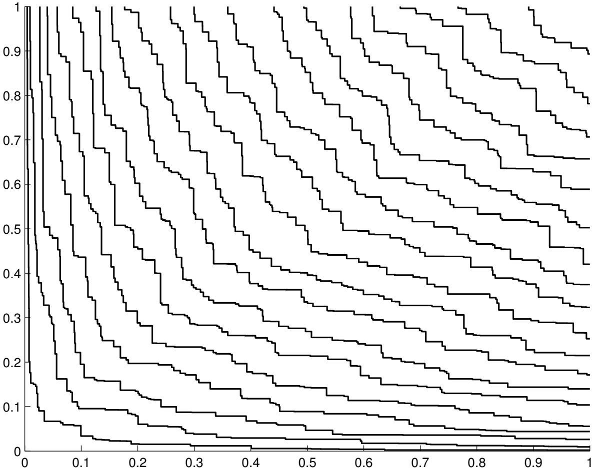
$n = 10^2$ points

Demo: Uniform distribution



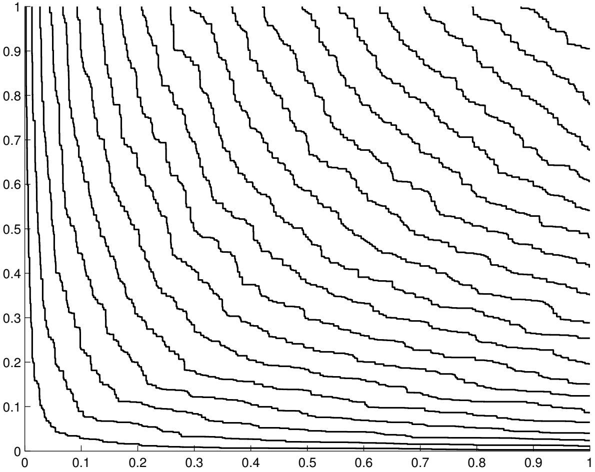
$n = 10^3$ points

Demo: Uniform distribution



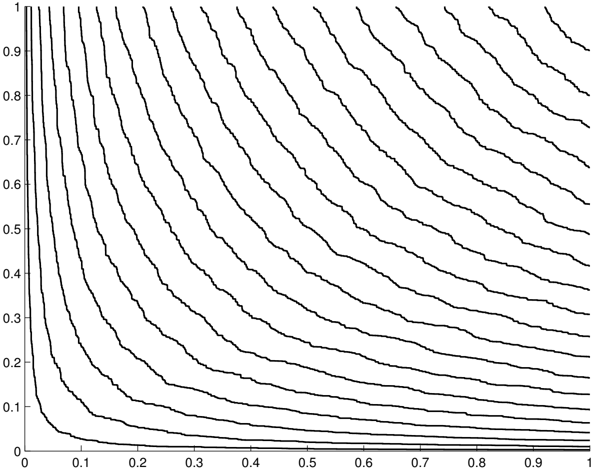
$n = 10^4$ points

Demo: Uniform distribution



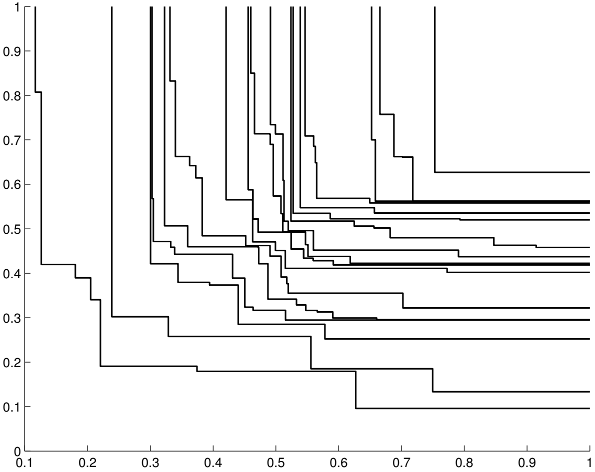
$n = 10^5$ points

Demo: Uniform distribution



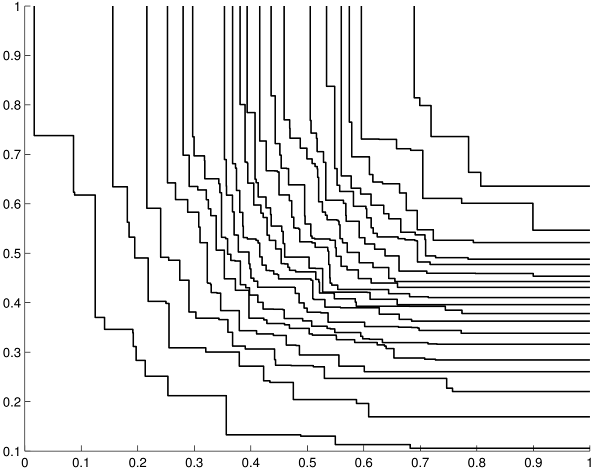
$n = 10^6$ points

Demo: Gaussian distribution



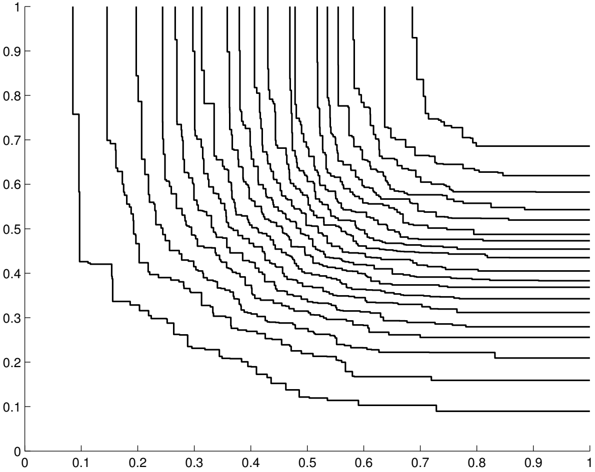
$n = 10^2$ points

Demo: Gaussian distribution



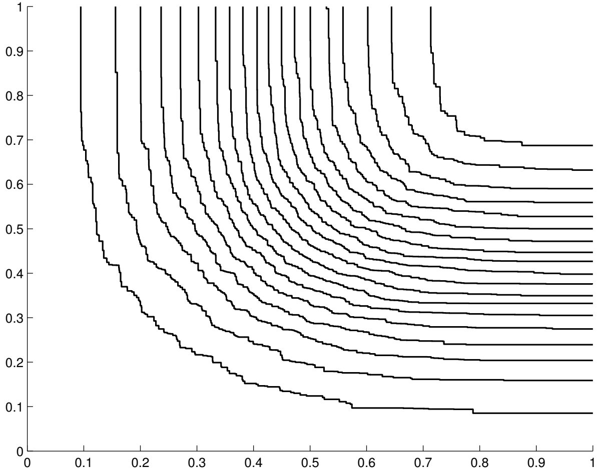
$n = 10^3$ points

Demo: Gaussian distribution



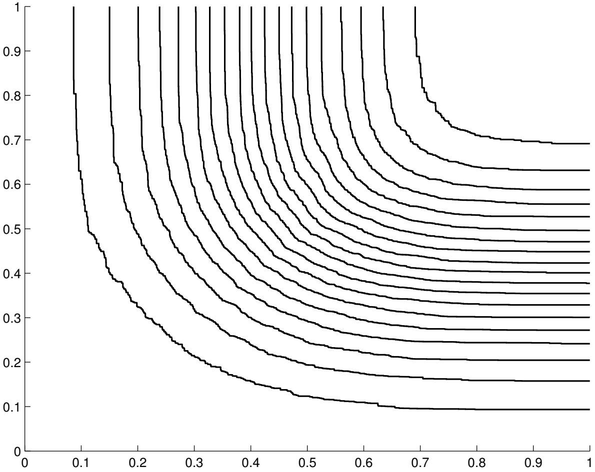
$n = 10^4$ points

Demo: Gaussian distribution



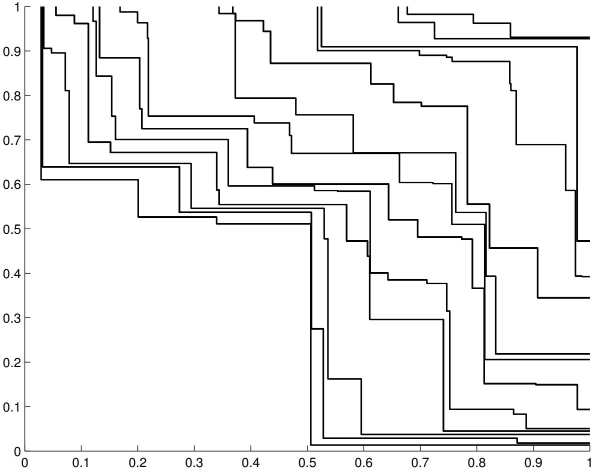
$n = 10^5$ points

Demo: Gaussian distribution



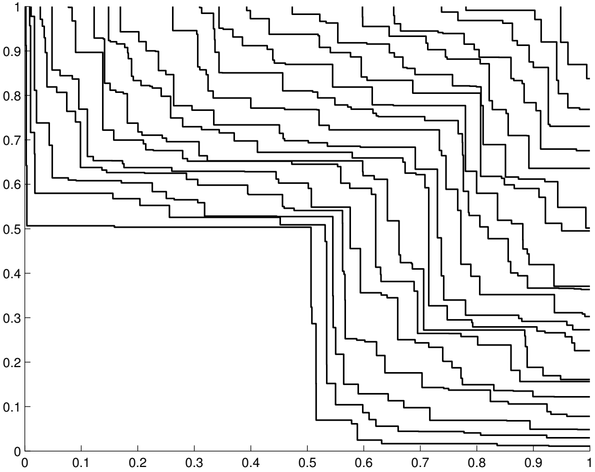
$n = 10^6$ points

Demo: Uniform distribution on $[0,1]^2 \setminus [0,0.5]^2$



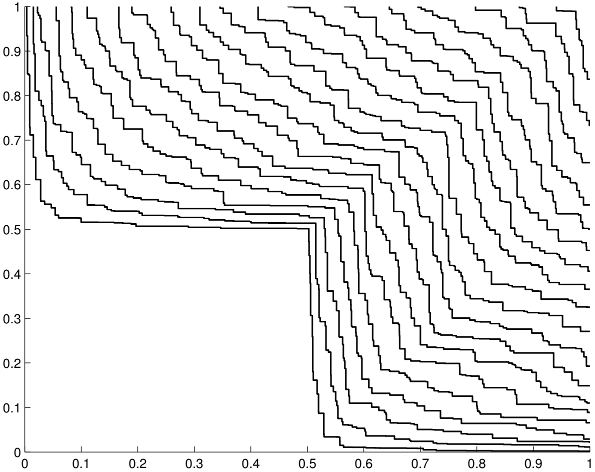
$n = 10^2$ points

Demo: Uniform distribution on $[0,1]^2 \setminus [0,0.5]^2$



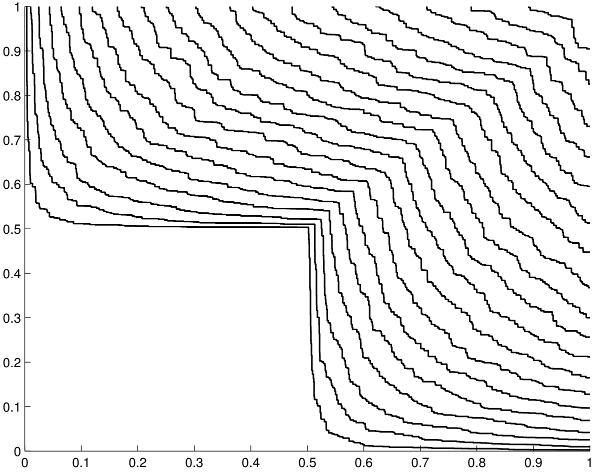
$n = 10^3$ points

Demo: Uniform distribution on $[0,1]^2 \setminus [0,0.5]^2$



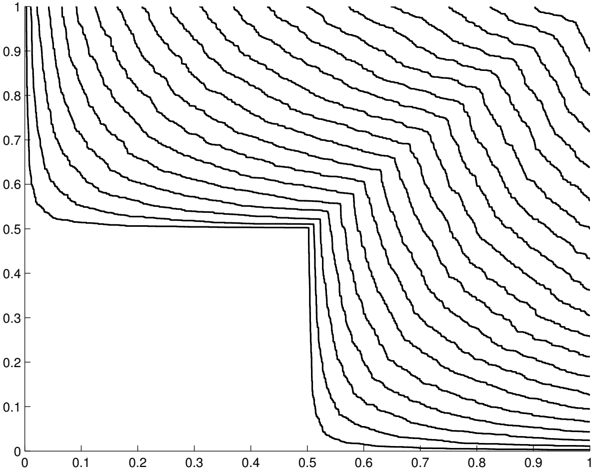
$n = 10^4$ points

Demo: Uniform distribution on $[0,1]^2 \setminus [0,0.5]^2$



$n = 10^5$ points

Demo: Uniform distribution on $[0,1]^2 \setminus [0,0.5]^2$



$n = 10^6$ points

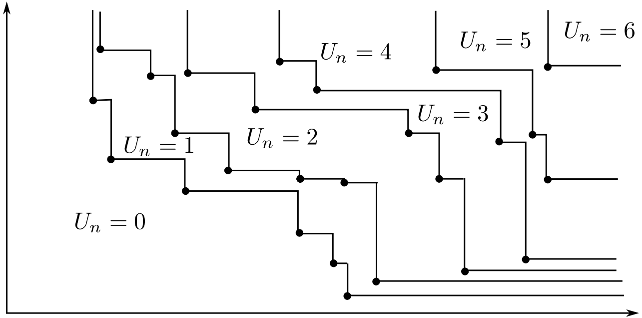
A PDE continuum limit for nondominated sorting

Let $X_1, \ldots, X_n$ be *i.i.d.* random variables in $[0, \infty)^d$ with continuous density f.

A PDE continuum limit for nondominated sorting

Let $X_1, \ldots, X_n$ be *i.i.d.* random variables in $[0,\infty)^d$ with continuous density f.

Let $U_n : \mathbb{R}^d \to \mathbb{N}_0$ be the function that 'counts' the layers $\mathcal{F}_1, \mathcal{F}_2, \ldots$



Theorem (Calder, Esedoḡlu, Hero, 2014)

There exists a universal constant $c_d > 0$ *such that with probability one*

$$n^{-\frac{1}{d}} U_n \longrightarrow c_d u \text{ locally uniformly as } n \to \infty$$

where $u \in C^{0,\frac{1}{d}}([0,\infty)^d)$ *is the unique nondecreasing* ($u_{x_i} \geq 0$) *viscosity solution of*

$$(\mathrm{P}) \begin{cases} u_{x_1} \cdots u_{x_d} = f & \text{in } \mathbb{R}^d_+ := (0,\infty)^d \\ u = 0 & \text{on } \partial\mathbb{R}^d_+. \end{cases}$$

Theorem (Calder, Esedoḡlu, Hero, 2014)

There exists a universal constant $c_d > 0$ *such that with probability one*

$$n^{-\frac{1}{d}} U_n \longrightarrow c_d u \ \textit{ locally uniformly as } n \to \infty$$

where $u \in C^{0,\frac{1}{d}}([0,\infty)^d)$ *is the unique nondecreasing* ($u_{x_i} \geq 0$) *viscosity solution of*

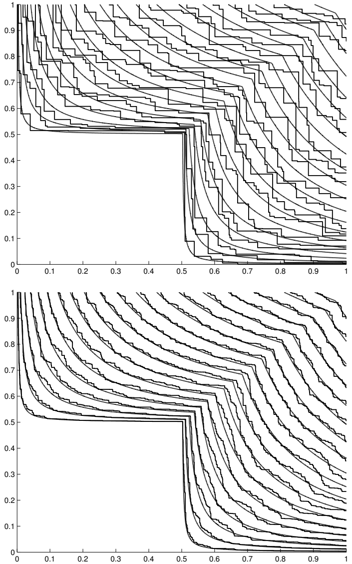
$$\text{(P)} \quad \begin{cases} u_{x_1} \cdots u_{x_d} = f & \textit{in } \mathbb{R}^d_+ := (0,\infty)^d \\ u = 0 & \textit{on } \partial\mathbb{R}^d_+. \end{cases}$$

Calder, J., Esedoḡlu, S., and Hero, A. O. (2014). A Hamilton-Jacobi equation for the continuum limit of non-dominated sorting. *SIAM Journal on Mathematical Analysis*, 46(1):603–638.

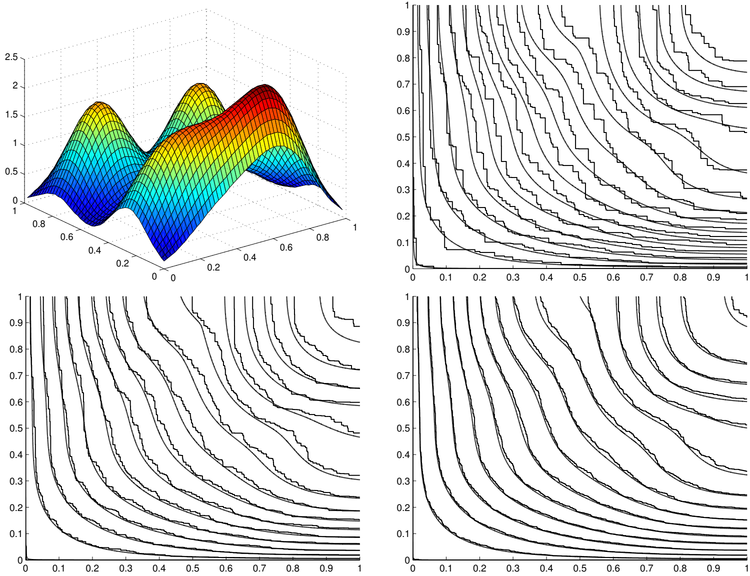
Calder, J. (2016). A direct verification argument for the Hamilton-Jacobi equation continuum limit of nondominated sorting. *Nonlinear Analysis Series A: Methods, Theory & Applications*, 141:88–108

Current work: Rate of convergence (Brendan Cook)

Demo: $f = 1 - \chi_{[0,0.5]^2}$



Demo: Multimodal f



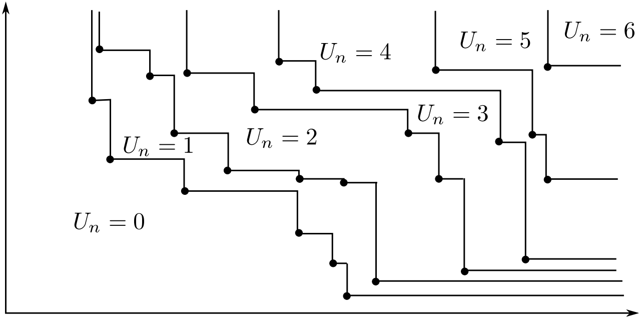
Quick "proof"

Let $X_1, \ldots, X_n$ be *i.i.d.* random variables in $[0, \infty)^d$ with continuous density f.

Quick "proof"

Let $X_1, \ldots, X_n$ be *i.i.d.* random variables in $[0, \infty)^d$ with continuous density f.

Let $U_n : \mathbb{R}^d \to \mathbb{N}_0$ be the function that 'counts' the layers $\mathcal{F}_1, \mathcal{F}_2, \ldots$

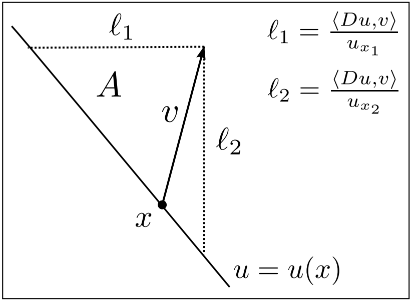


Quick "proof"

Let's suppose that $n^{-\alpha} U_n \longrightarrow u \in C^1$ as $n \to \infty$ for some $\alpha \in [0,1]$.

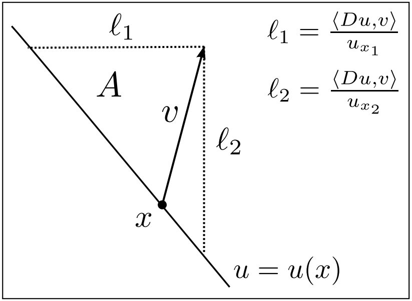
Quick “proof”

Let’s suppose that $n^{-\alpha} U_n \longrightarrow u \in C^1$ as $n \to \infty$ for some $\alpha \in [0, 1]$.



Quick "proof"

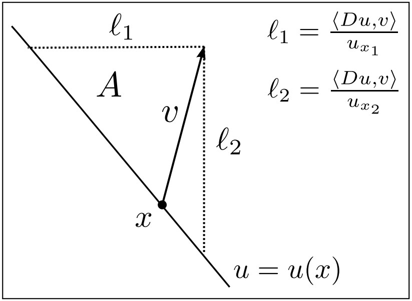
Let's suppose that $n^{-\alpha} U_n \longrightarrow u \in C^1$ as $n \to \infty$ for some $\alpha \in [0, 1]$.



$$\langle Du, v \rangle \quad \approx \quad u(x + v) - u(x)$$

Quick "proof"

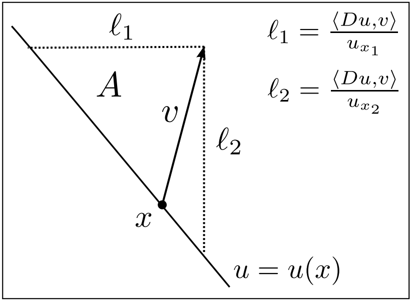
Let's suppose that $n^{-\alpha} U_n \longrightarrow u \in C^1$ as $n \to \infty$ for some $\alpha \in [0,1]$.



$$\begin{aligned} \langle Du, v\rangle &\approx u(x+v) - u(x) \\ &\approx (\#\text{ fronts in } A) n^{-\alpha} \end{aligned}$$

Quick "proof"

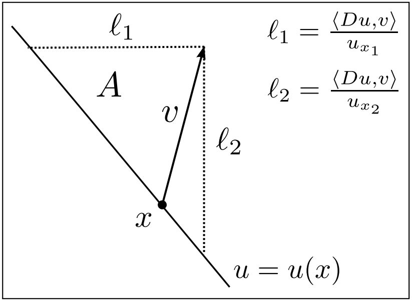
Let's suppose that $n^{-\alpha} U_n \longrightarrow u \in C^1$ as $n \to \infty$ for some $\alpha \in [0,1]$.



$$\begin{aligned}
\langle Du, v\rangle &\approx u(x+v) - u(x) \\
&\approx (\#\text{ fronts in } A)n^{-\alpha} \\
&\approx (\#\text{ samples in } A)^{\alpha} n^{-\alpha}
\end{aligned}$$

Quick "proof"

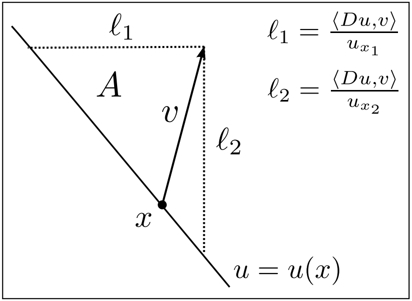
Let's suppose that $n^{-\alpha} U_n \longrightarrow u \in C^1$ as $n \to \infty$ for some $\alpha \in [0,1]$.



$$
\begin{aligned}
\langle Du, v\rangle &\approx u(x+v) - u(x) \\
&\approx (\#\text{ fronts in } A) n^{-\alpha} \\
&\approx (\#\text{ samples in } A)^{\alpha} n^{-\alpha} \\
&\approx (n|A|f(x))^{\alpha} n^{-\alpha}
\end{aligned}
$$

Quick "proof"

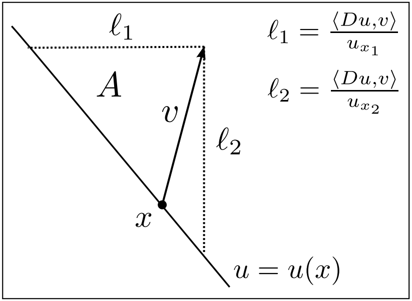
Let's suppose that $n^{-\alpha} U_n \longrightarrow u \in C^1$ as $n \to \infty$ for some $\alpha \in [0,1]$.



$$
\begin{aligned}
\langle Du, v\rangle &\approx u(x+v) - u(x) \\
&\approx (\#\text{ fronts in } A)n^{-\alpha} \\
&\approx (\#\text{ samples in } A)^{\alpha} n^{-\alpha} \\
&\approx (n|A|f(x))^{\alpha} n^{-\alpha} \\
&\approx |A|^{\alpha} f(x)^{\alpha}.
\end{aligned}
$$

Quick "proof"

Let's suppose that $n^{-\alpha} U_n \longrightarrow u \in C^1$ as $n \to \infty$ for some $\alpha \in [0,1]$.

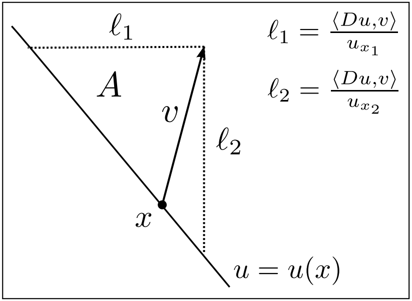


$$\begin{aligned}
\langle Du, v\rangle &\approx u(x+v) - u(x) \\
&\approx (\#\text{ fronts in } A) n^{-\alpha} \\
&\approx (\#\text{ samples in } A)^{\alpha} n^{-\alpha} \\
&\approx (n|A|f(x))^{\alpha} n^{-\alpha} \\
&\approx |A|^{\alpha} f(x)^{\alpha}.
\end{aligned}$$

Use $|A| \approx \frac{\langle Du, v\rangle^d}{u_{x_1} \cdots u_{x_d}}$

Quick "proof"

Let's suppose that $n^{-\alpha} U_n \longrightarrow u \in C^1$ as $n \to \infty$ for some $\alpha \in [0,1]$.



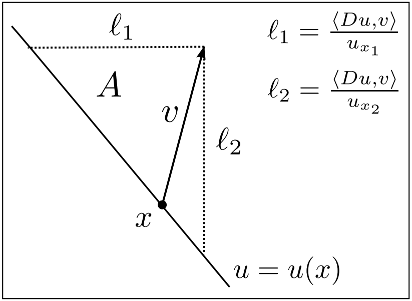
$$\begin{aligned}
\langle Du, v\rangle &\approx u(x+v) - u(x) \\
&\approx (\text{\# fronts in } A) n^{-\alpha} \\
&\approx (\text{\# samples in } A)^{\alpha} n^{-\alpha} \\
&\approx (n|A|f(x))^{\alpha} n^{-\alpha} \\
&\approx |A|^{\alpha} f(x)^{\alpha}.
\end{aligned}$$

Use $|A| \approx \frac{\langle Du, v\rangle^d}{u_{x_1} \cdots u_{x_d}}$ to find

$$\langle Du, v\rangle \approx \left(\frac{f(x)}{u_{x_1} \cdots u_{x_d}}\right)^{\alpha} \langle Du, v\rangle^{\alpha d}$$

Quick "proof"

Let's suppose that $n^{-\alpha} U_n \longrightarrow u \in C^1$ as $n \to \infty$ for some $\alpha \in [0,1]$.



$$\begin{aligned}
\langle Du, v\rangle &\approx u(x+v) - u(x) \\
&\approx (\#\text{ fronts in } A) n^{-\alpha} \\
&\approx (\#\text{ samples in } A)^{\alpha} n^{-\alpha} \\
&\approx (n|A| f(x))^{\alpha} n^{-\alpha} \\
&\approx |A|^{\alpha} f(x)^{\alpha}.
\end{aligned}$$

Use $|A| \approx \frac{\langle Du, v\rangle^d}{u_{x_1} \cdots u_{x_d}}$ to find

$$\langle Du, v\rangle \approx \left(\frac{f(x)}{u_{x_1} \cdots u_{x_d}}\right)^{\alpha} \langle Du, v\rangle^{\alpha d}$$

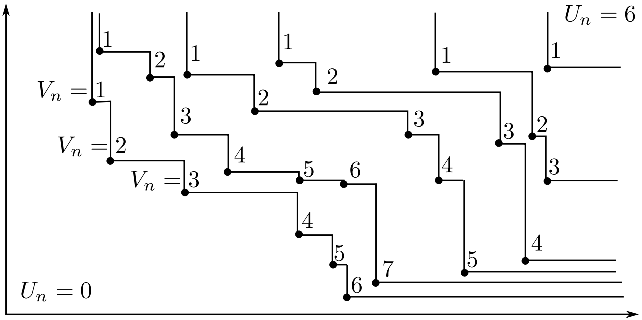
If $\alpha d = 1$, or $\alpha = 1/d$, then

$$\boxed{u_{x_1} \cdots u_{x_d} = f}$$

Ordering within each front

Let $X_1, \ldots, X_n$ be i.i.d. random variables with density f on $[0,1]^2$. Define

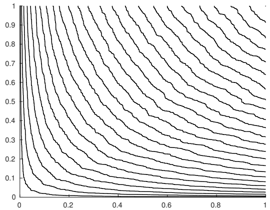
$$V_n(X_i) = \text{Index of } X_i \text{ within its Pareto front.}$$

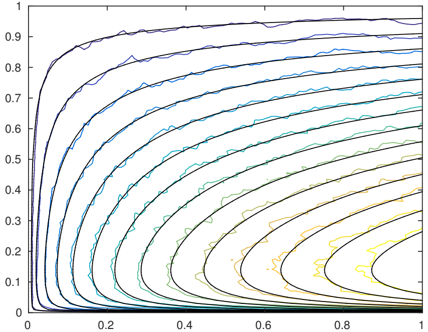


Demo: Uniform distribution on $[0,1]^2$

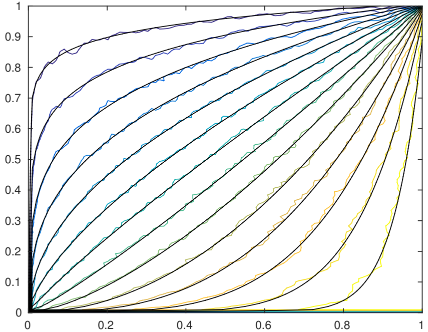
$$\text{(T)} \begin{cases} \langle Dv, D^\perp u\rangle = f & \text{in } (0,1)^2, \\ v = 0 & \text{on } (0,1)\times\{x_2 = 1\}. \end{cases}$$

$$\text{(T')} \begin{cases} \langle Dw, vD^\perp u\rangle = wf & \text{in } (0,1)^2, \\ w = 1 & \text{on } \{x_1 = 1\}\times(0,1). \end{cases}$$





(a) V_n vs. v



(b) W_n vs. w

Fast approximate sorting

Algorithm (PDE-based Ranking)

1. *Select k points from $X_1, \dots, X_n$ at random. Call them $Y_1, \dots, Y_k$. ($k \ll n$)*

Fast approximate sorting

Algorithm (PDE-based Ranking)

1. *Select k points from $X_1, \dots, X_n$ at random. Call them $Y_1, \dots, Y_k$. ($k \ll n$)*
2. *Estimate f with a histogram*

$$\hat{f}(x) = \frac{1}{kh^d} \cdot \#\Big\{ Y_i \,:\, Y_i \in [x, x + h\mathbf{1}] \Big\}.$$

Fast approximate sorting

Algorithm (PDE-based Ranking)

1. *Select* k *points from* $X_1, \ldots, X_n$ *at random. Call them* $Y_1, \ldots, Y_k$. *(*$k \ll n$*)*
2. *Estimate* f *with a histogram*

$$\hat{f}(x) = \frac{1}{kh^d} \cdot \#\Big\{ Y_i \, : \, Y_i \in [x, x + h\mathbf{1}] \Big\}.$$

3. *Compute the numerical solution* $\hat{U}_h$ *of the PDE.*

Fast approximate sorting

Algorithm (PDE-based Ranking)

1. *Select* k *points from* $X_1, \dots, X_n$ *at random. Call them* $Y_1, \dots, Y_k$. *(*$k \ll n$*)*
2. *Estimate* f *with a histogram*

$$\hat{f}(x) = \frac{1}{kh^d} \cdot \#\left\{ Y_i \ : \ Y_i \in [x, x + h\mathbf{1}] \right\}.$$

3. *Compute the numerical solution* $\hat{U}_h$ *of the PDE.*
4. *Evaluate* $\hat{U}_h(X_i)$ *for* $i = 1, \dots, n$ *via interpolation.*

Fast approximate sorting

Algorithm (PDE-based Ranking)

1. *Select k points from $X_1, \ldots, X_n$ at random. Call them $Y_1, \ldots, Y_k$. ($k \ll n$)*
2. *Estimate f with a histogram*

$$\hat{f}(x) = \frac{1}{kh^d} \cdot \#\Big\{ Y_i \,:\, Y_i \in [x, x + h\mathbf{1}] \Big\}.$$

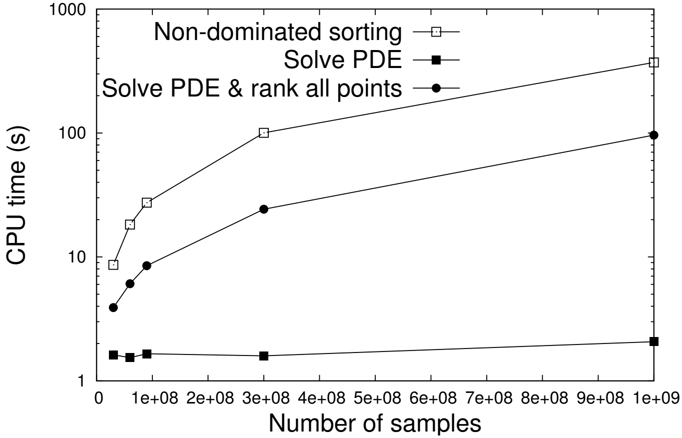
3. *Compute the numerical solution $\hat{U}_h$ of the PDE.*
4. *Evaluate $\hat{U}_h(X_i)$ for $i = 1, \ldots, n$ via interpolation.*

Notes:

- Total complexity is $O(k + h^{-d} + n)$.
- If we fix k and h, independent of n, then Steps 1-3 have $O(1)$ complexity.

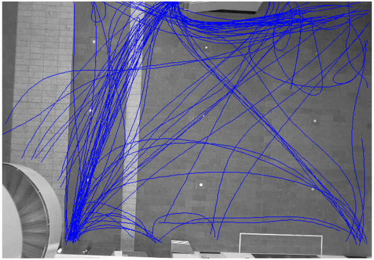
Calder, J., Esedoḡlu, S., and Hero, A. O. (2015). A PDE-based approach to nondominated sorting. *SIAM Journal on Numerical Analysis*, 53(1):82–104.

CPU Time (C/C++)

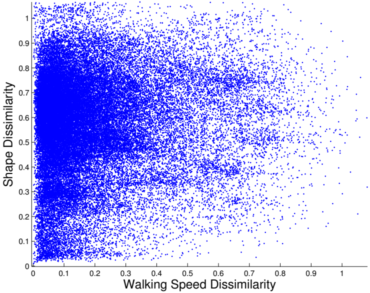


- # Subsamples $= k = 10^7$, Grid for solving PDE $= 250 \times 250$.
- $O(n \log n)$ non-dominated sorting of [Felsner and Wernisch, 1999].

Application in anomaly detection



(a) Example trajectories

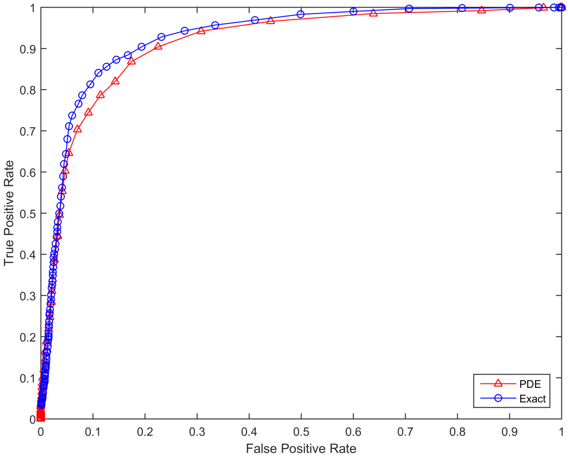


(b) 5×10^5 Pareto points

Abbasi, B., Calder, J., and Oberman, A.M. Anomaly detection and classification for streaming data using PDEs *SIAM Journal on Applied Mathematics*, 78(2), 921–941, 2018.

Results

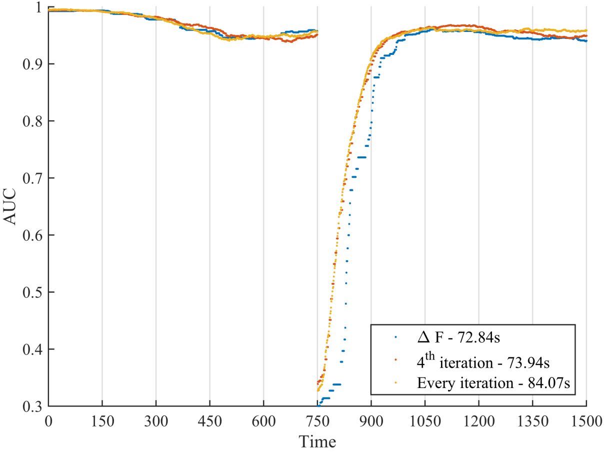
Anomaly detection with PDE-based ranking: Reduces complexity from $O(n^2)$ to $O(n)$.



Abbasi, B., Calder, J., and Oberman, A.M. Anomaly detection and classification for streaming data using PDEs *SIAM Journal on Applied Mathematics*, 78(2), 921–941, 2018.

Results

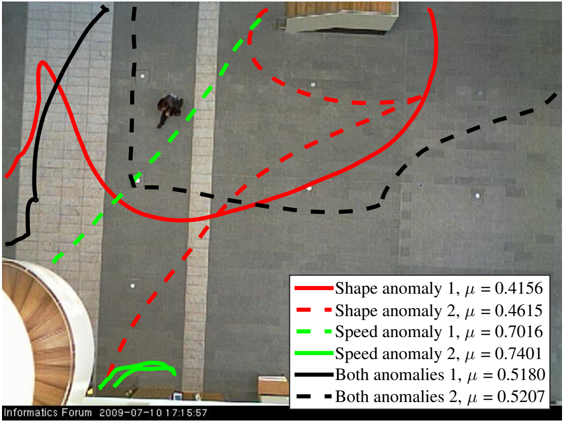
Anomaly detection for streaming data:



Abbasi, B., Calder, J., and Oberman, A.M. Anomaly detection and classification for streaming data using PDEs *SIAM Journal on Applied Mathematics*, 78(2), 921–941, 2018.

Examples of detected anomalies. . .

with classifications using the new transport equations.



Abbasi, B., Calder, J., and Oberman, A.M. Anomaly detection and classification for streaming data using PDEs *SIAM Journal on Applied Mathematics*, 78(2), 921–941, 2018.

Outline

Convex hull peeling

Question: How to define ‘median’ in dimensions $d \geq 2$?

Convex hull peeling

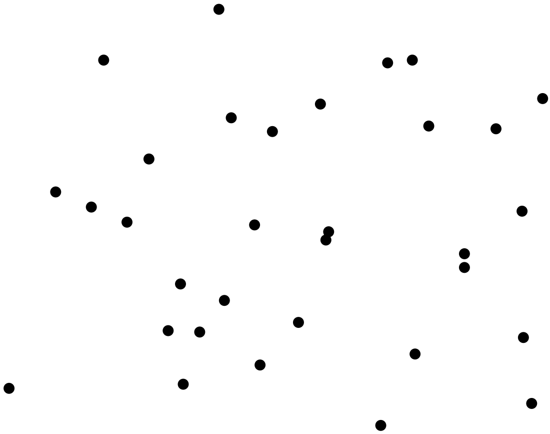
Question: How to define 'median' in dimensions $d \geq 2$?

Barnett [Barnett, 1976]: Convex hull peeling

Convex hull peeling

Question: How to define 'median' in dimensions $d \geq 2$?

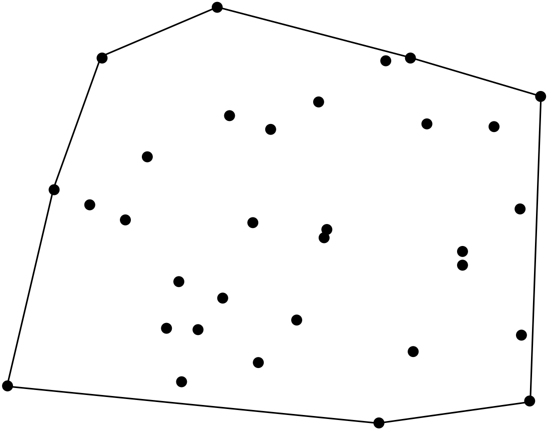
Barnett [Barnett, 1976]: Convex hull peeling



Convex hull peeling

Question: How to define 'median' in dimensions $d \geq 2$?

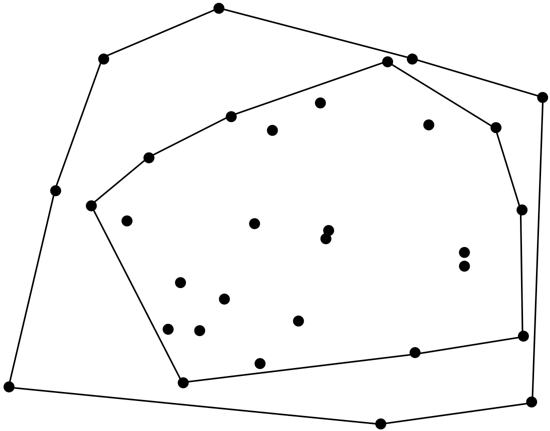
Barnett [Barnett, 1976]: Convex hull peeling



Convex hull peeling

Question: How to define 'median' in dimensions $d \geq 2$?

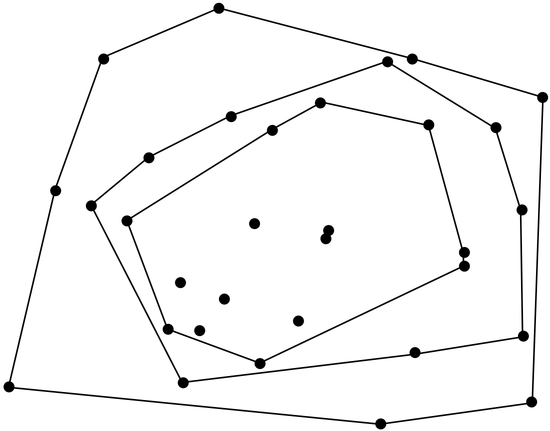
Barnett [Barnett, 1976]: Convex hull peeling



Convex hull peeling

Question: How to define 'median' in dimensions $d \geq 2$?

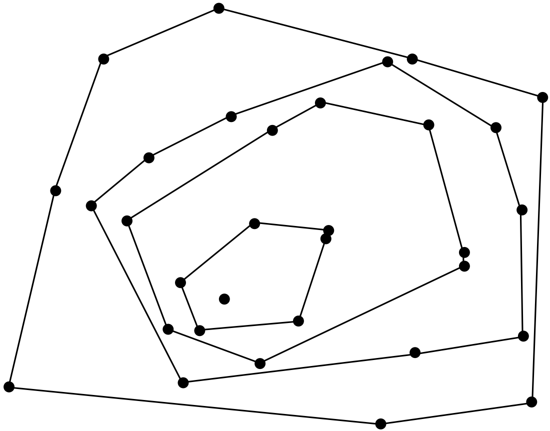
Barnett [Barnett, 1976]: Convex hull peeling



Convex hull peeling

Question: How to define 'median' in dimensions $d \geq 2$?

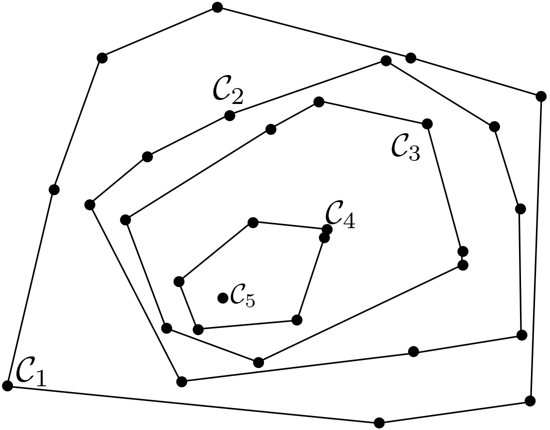
Barnett [Barnett, 1976]: Convex hull peeling



Convex hull peeling

Question: How to define 'median' in dimensions $d \geq 2$?

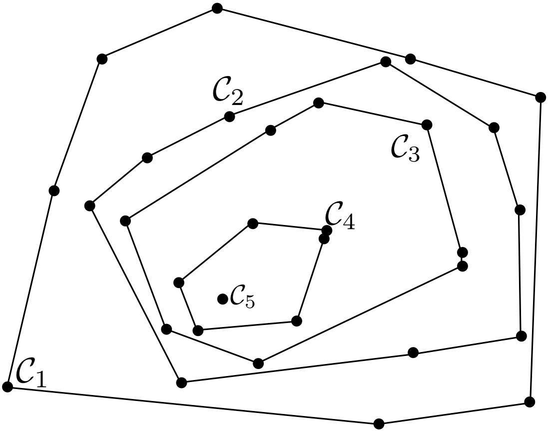
Barnett [Barnett, 1976]: Convex hull peeling



Convex hull peeling

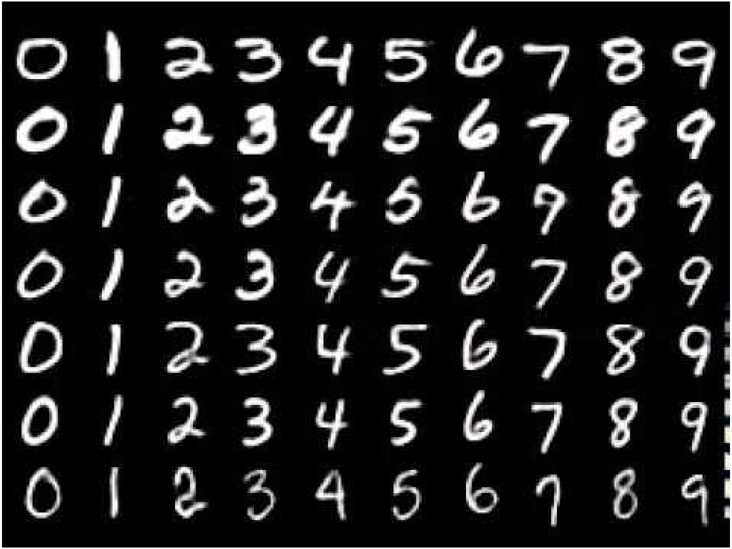
Question: How to define 'median' in dimensions $d \geq 2$?

Barnett [Barnett, 1976]: Convex hull peeling



Convex hull peeling median := Centroid of final layer

MNIST handwritten digit dataset



Convex hull peeling

Let $X_1, \ldots, X_n$ be points in $\mathbb{R}^d$ and set $S = \{X_1, \ldots, X_n\}$.

Convex hull peeling

Let $X_1, \ldots, X_n$ be points in $\mathbb{R}^d$ and set $S = \{X_1, \ldots, X_n\}$.

Definition

Convex hull peeling is the process of arranging S into convex layers $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \ldots,$ defined by

$$\mathcal{C}_1 = \text{Vertices of convex hull of } S,$$

$$\mathcal{C}_k = \text{Vertices of convex hull of } S \setminus (\mathcal{C}_1 \cup \cdots \cup \mathcal{C}_{k-1}).$$

Convex hull peeling

Applications:

- Robust statistics, machine learning, etc.
  - [Rousseeuw and Struyf, 2004],[Donoho and Gasko, 1992], [Hodge and Austin, 2004].

Convex hull peeling

Applications:

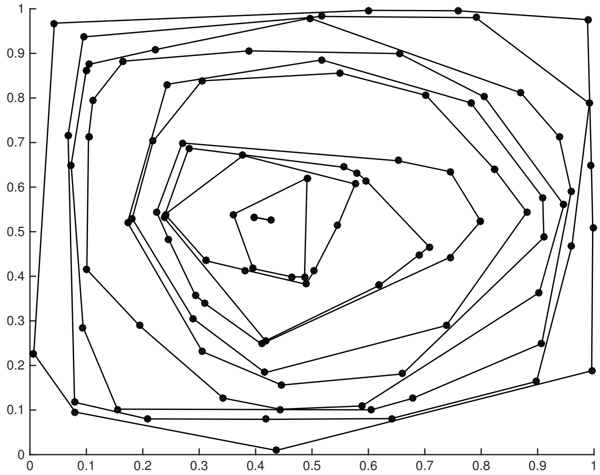
- Robust statistics, machine learning, etc.
  - [Rousseeuw and Struyf, 2004],[Donoho and Gasko, 1992], [Hodge and Austin, 2004].
- Matching of deformed pointclouds [Suk and Flusser, 1999].

Convex hull peeling

Applications:

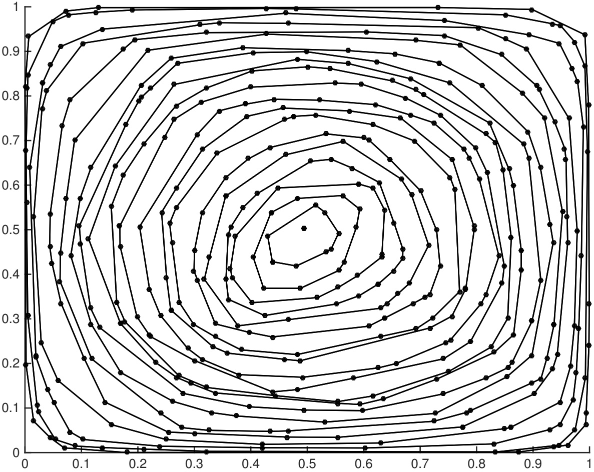
- Robust statistics, machine learning, etc.
  - [Rousseeuw and Struyf, 2004],[Donoho and Gasko, 1992], [Hodge and Austin, 2004].
- Matching of deformed pointclouds [Suk and Flusser, 1999].
- Fingerprint matching [Poulos et al., 2005].

Convex hull peeling: Demo - Uniform distribution



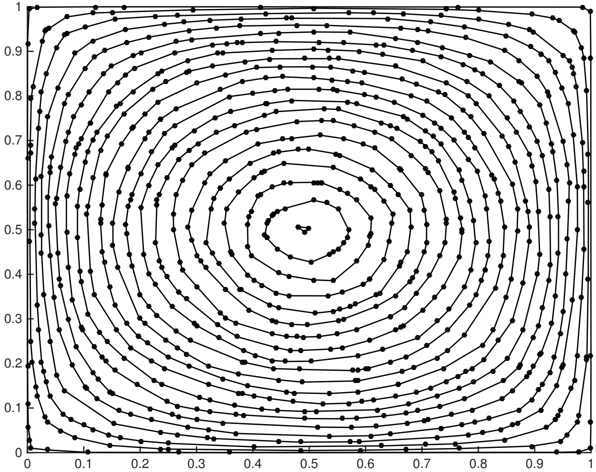
$n = 10^2$ points

Convex hull peeling: Demo - Uniform distribution



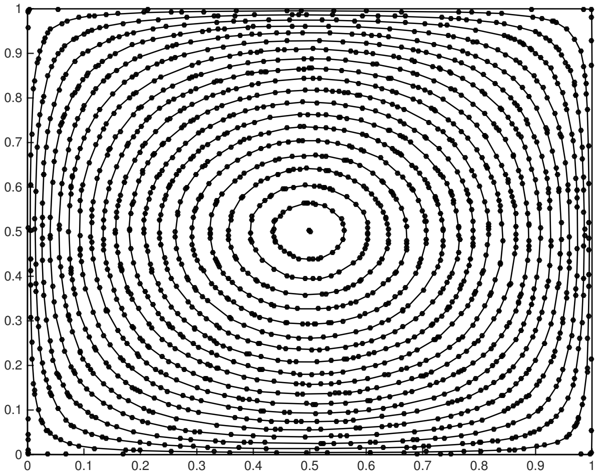
$n = 10^3$ points

Convex hull peeling: Demo - Uniform distribution



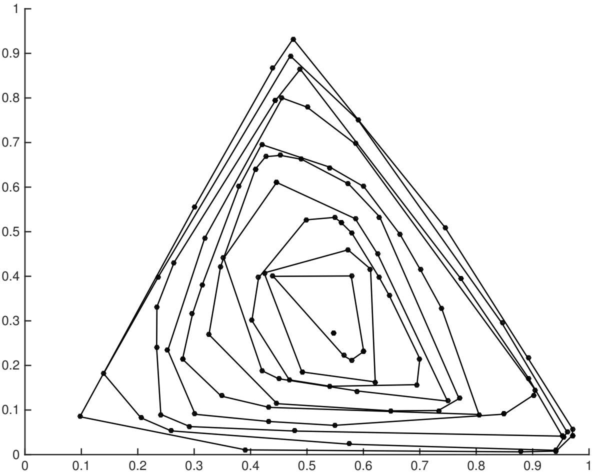
$n = 10^4$ points

Convex hull peeling: Demo - Uniform distribution



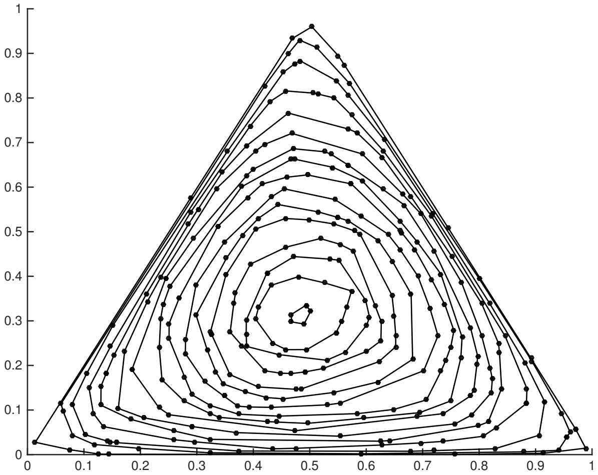
$n = 10^5$ points

Convex hull peeling: Demo - Triangle distribution



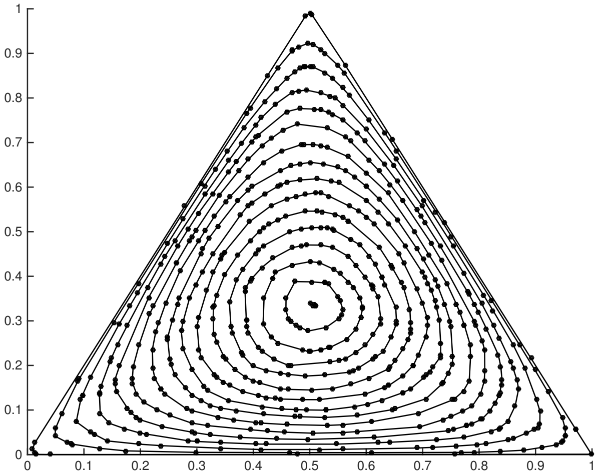
$n = 10^2$ points

Convex hull peeling: Demo - Triangle distribution



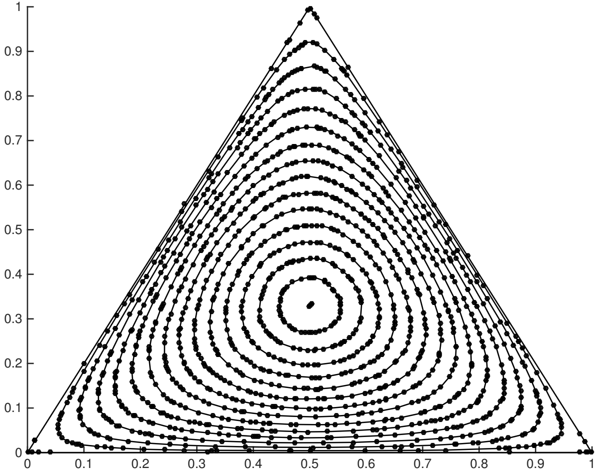
$n = 10^3$ points

Convex hull peeling: Demo - Triangle distribution



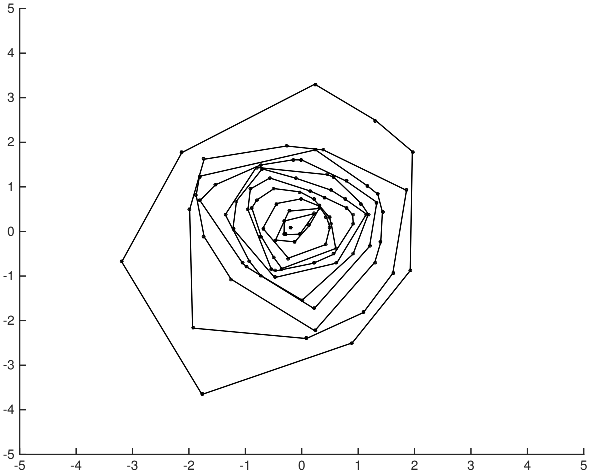
$n = 10^4$ points

Convex hull peeling: Demo - Triangle distribution



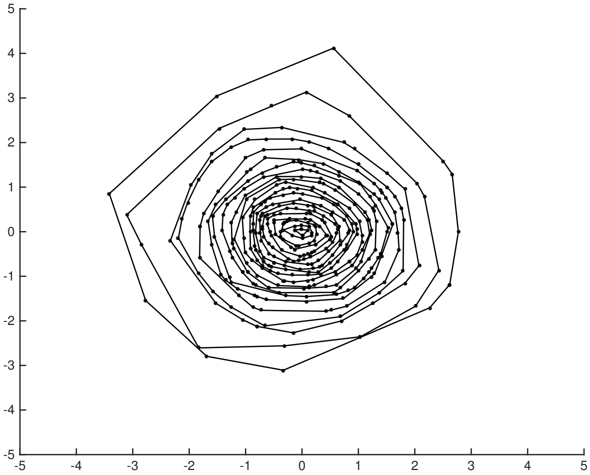
$n = 10^5$ points

Convex hull peeling: Demo - Gaussian distribution



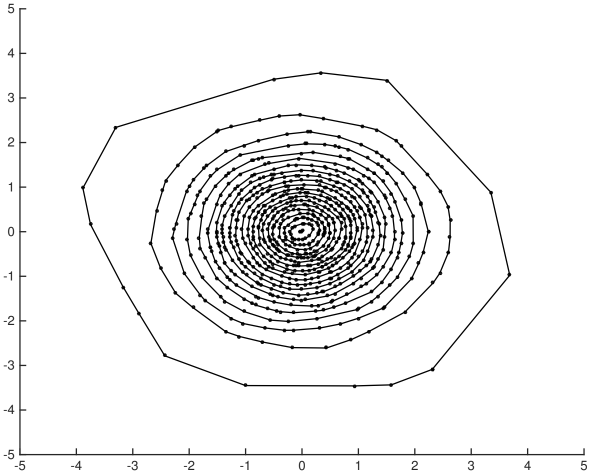
$n = 10^2$ points

Convex hull peeling: Demo - Gaussian distribution



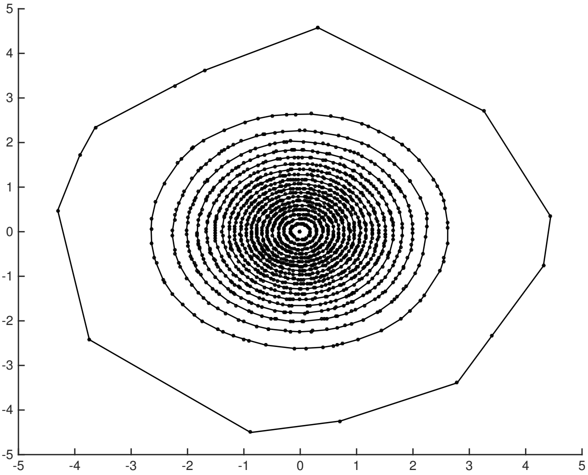
$n = 10^3$ points

Convex hull peeling: Demo - Gaussian distribution



$n = 10^4$ points

Convex hull peeling: Demo - Gaussian distribution



$n = 10^5$ points

A two player game for convex hull peeling

Players: Paul and Carol
State space: $\mathcal{X} := \{X_1, \dots, X_n\}$

A two player game for convex hull peeling

Players: Paul and Carol
State space: $\mathcal{X} := \{X_1, \ldots, X_n\}$

Paul's goal: Reach vertex of convex hull
Carol's goal: Obstruct Paul

A two player game for convex hull peeling

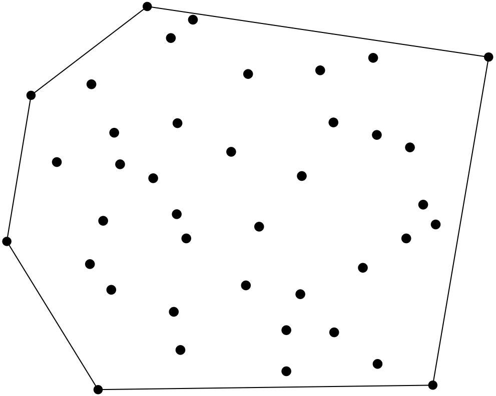
Players: Paul and Carol
State space: $\mathcal{X} := \{X_1, \ldots, X_n\}$

Paul's goal: Reach vertex of convex hull
Carol's goal: Obstruct Paul

Rules of the game: Token starts at $x^0 \in \mathcal{X}$ and is moved according to:

1. Paul picks $v \in \mathbb{S}^{d-1}$
2. Carol moves token to any $x^{k+1} \in \mathcal{X}$ satisfying

$$(x^{k+1} - x^k) \cdot v > 0.$$



A two player game for convex hull peeling

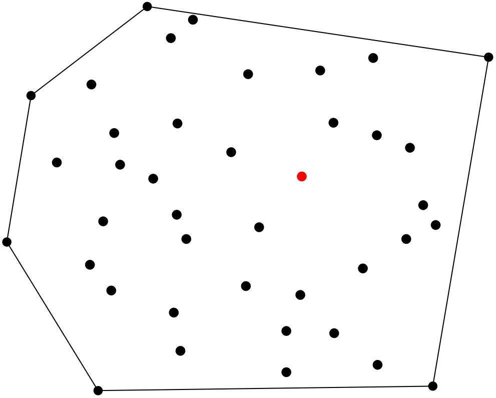
Players: Paul and Carol
State space: $\mathcal{X} := \{X_1, \ldots, X_n\}$

Paul's goal: Reach vertex of convex hull
Carol's goal: Obstruct Paul

Rules of the game: Token starts at $x^0 \in \mathcal{X}$ and is moved according to:

1. Paul picks $v \in \mathbb{S}^{d-1}$
2. Carol moves token to any $x^{k+1} \in \mathcal{X}$ satisfying

$$(x^{k+1} - x^k) \cdot v > 0.$$



A two player game for convex hull peeling

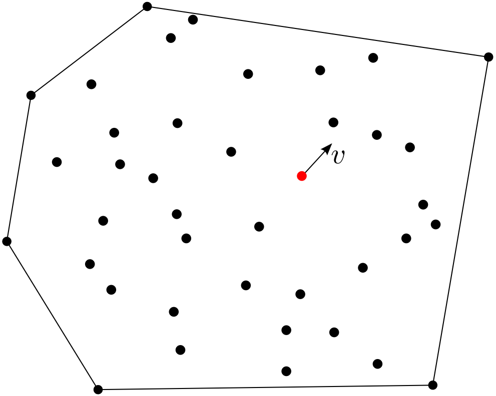
Players: Paul and Carol
State space: $\mathcal{X} := \{X_1, \ldots, X_n\}$

Paul's goal: Reach vertex of convex hull
Carol's goal: Obstruct Paul

Rules of the game: Token starts at $x^0 \in \mathcal{X}$ and is moved according to:

1. Paul picks $v \in \mathbb{S}^{d-1}$
2. Carol moves token to any $x^{k+1} \in \mathcal{X}$ satisfying

$$(x^{k+1} - x^k) \cdot v > 0.$$



A two player game for convex hull peeling

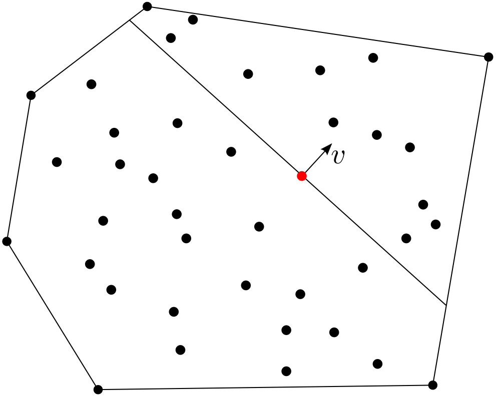
Players: Paul and Carol
State space: $\mathcal{X} := \{X_1, \ldots, X_n\}$

Paul's goal: Reach vertex of convex hull
Carol's goal: Obstruct Paul

Rules of the game: Token starts at $x^0 \in \mathcal{X}$ and is moved according to:

1. Paul picks $v \in \mathbb{S}^{d-1}$
2. Carol moves token to any $x^{k+1} \in \mathcal{X}$ satisfying

$$(x^{k+1} - x^k) \cdot v > 0.$$



A two player game for convex hull peeling

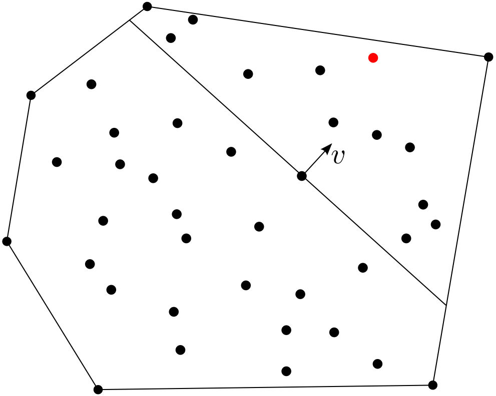
Players: Paul and Carol
State space: $\mathcal{X} := \{X_1, \ldots, X_n\}$

Paul's goal: Reach vertex of convex hull
Carol's goal: Obstruct Paul

Rules of the game: Token starts at $x^0 \in \mathcal{X}$ and is moved according to:

1. Paul picks $v \in \mathbb{S}^{d-1}$
2. Carol moves token to any $x^{k+1} \in \mathcal{X}$ satisfying

$$(x^{k+1} - x^k) \cdot v > 0.$$



A two player game for convex hull peeling

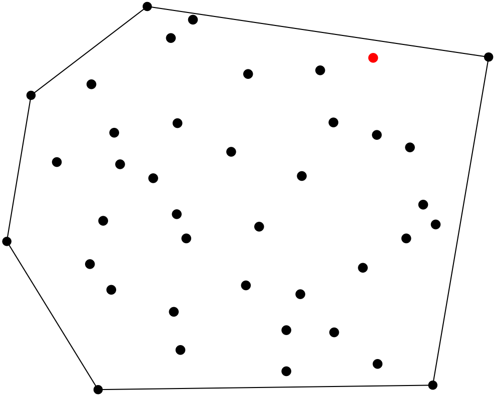
Players: Paul and Carol
State space: $\mathcal{X} := \{X_1, \ldots, X_n\}$

Paul's goal: Reach vertex of convex hull
Carol's goal: Obstruct Paul

Rules of the game: Token starts at $x^0 \in \mathcal{X}$ and is moved according to:

1. Paul picks $v \in \mathbb{S}^{d-1}$
2. Carol moves token to any $x^{k+1} \in \mathcal{X}$ satisfying

$$(x^{k+1} - x^k) \cdot v > 0.$$



A two player game for convex hull peeling

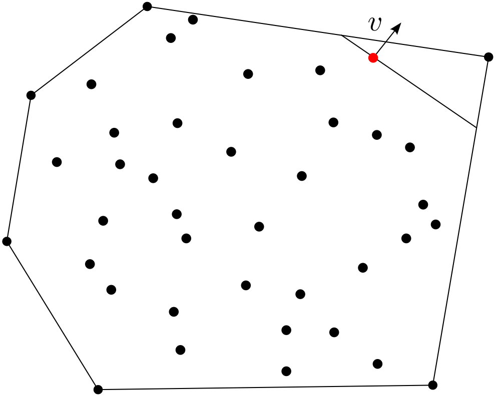
Players: Paul and Carol
State space: $\mathcal{X} := \{X_1, \ldots, X_n\}$

Paul's goal: Reach vertex of convex hull
Carol's goal: Obstruct Paul

Rules of the game: Token starts at $x^0 \in \mathcal{X}$ and is moved according to:

1. Paul picks $v \in \mathbb{S}^{d-1}$
2. Carol moves token to any $x^{k+1} \in \mathcal{X}$ satisfying

$$(x^{k+1} - x^k) \cdot v > 0.$$



A two player game for convex hull peeling

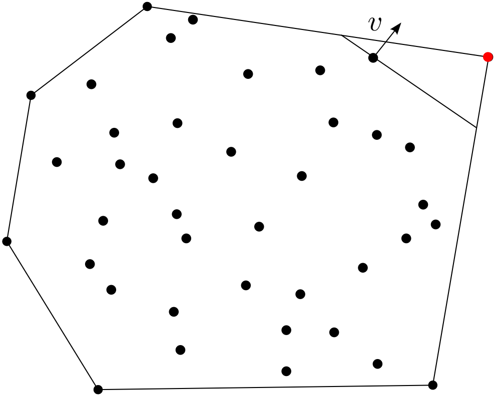
Players: Paul and Carol
State space: $\mathcal{X} := \{X_1, \ldots, X_n\}$

Paul's goal: Reach vertex of convex hull
Carol's goal: Obstruct Paul

Rules of the game: Token starts at $x^0 \in \mathcal{X}$ and is moved according to:

1. Paul picks $v \in \mathbb{S}^{d-1}$
2. Carol moves token to any $x^{k+1} \in \mathcal{X}$ satisfying

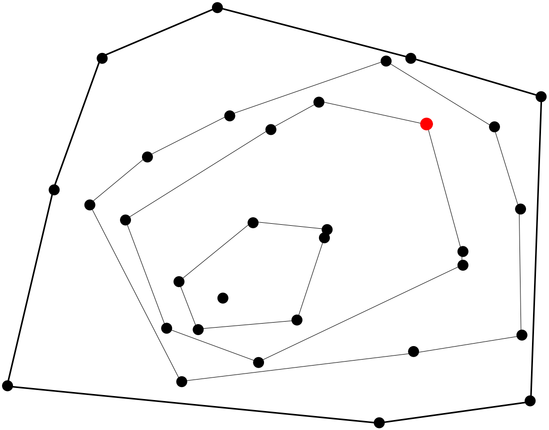
$$(x^{k+1} - x^k) \cdot v > 0.$$



A two player game for convex hull peeling

Paul's optimal choice: Any halfspace supporting current convex layer

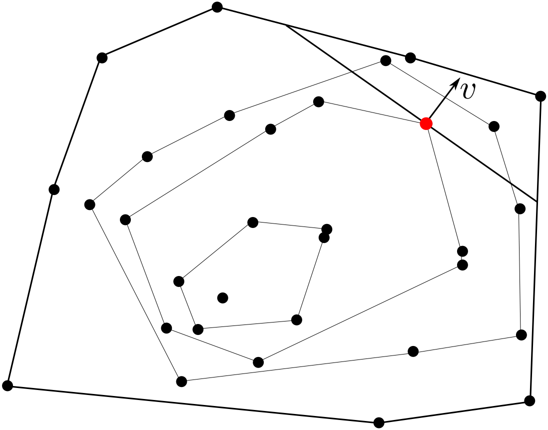
Carol's optimal choice: Any point on the previous convex layer



A two player game for convex hull peeling

Paul's optimal choice: Any halfspace supporting current convex layer

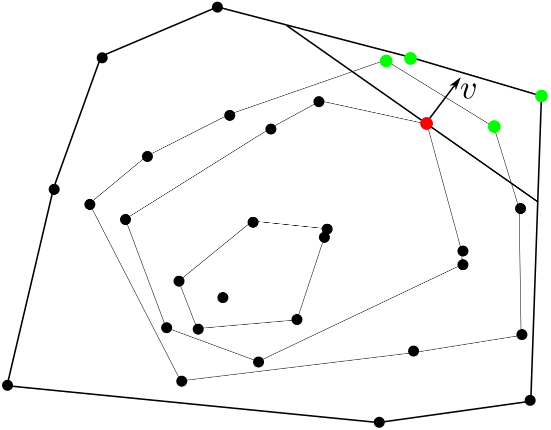
Carol's optimal choice: Any point on the previous convex layer



A two player game for convex hull peeling

Paul's optimal choice: Any halfspace supporting current convex layer

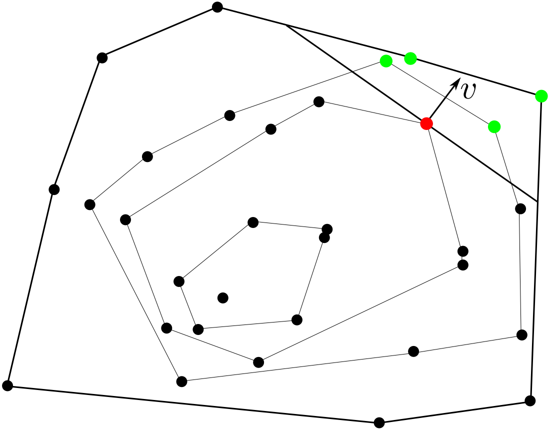
Carol's optimal choice: Any point on the previous convex layer



A two player game for convex hull peeling

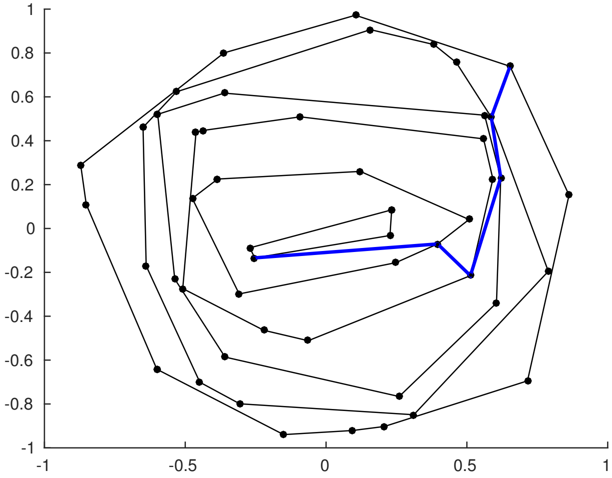
Paul's optimal choice: Any halfspace supporting current convex layer

Carol's optimal choice: Any point on the previous convex layer



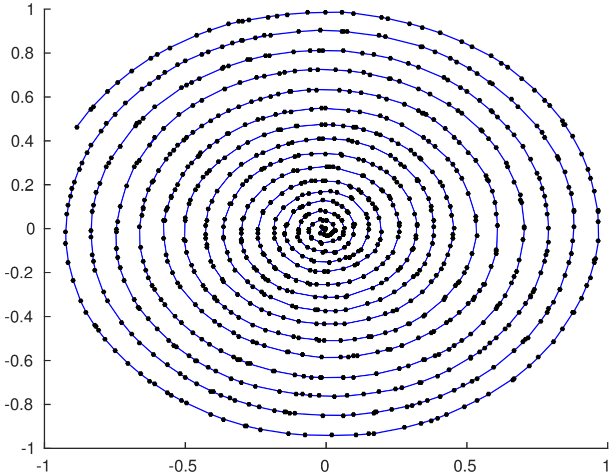
Value function $= U_n(x^0) =$ Convex depth function.

A two player game for convex hull peeling



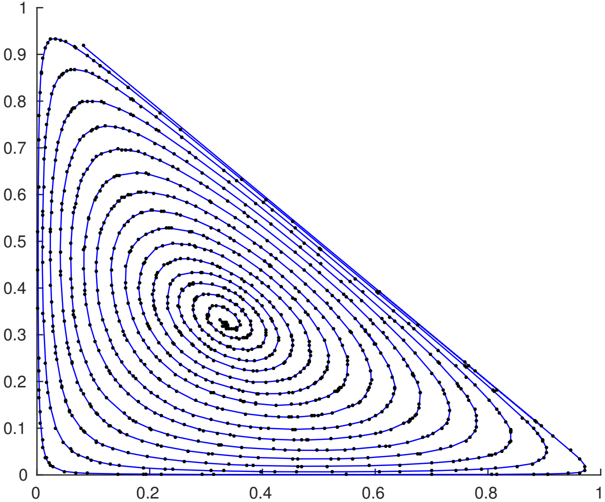
$n = 50$ points

A two player game for convex hull peeling



$n = 10^5$ points

A two player game for convex hull peeling

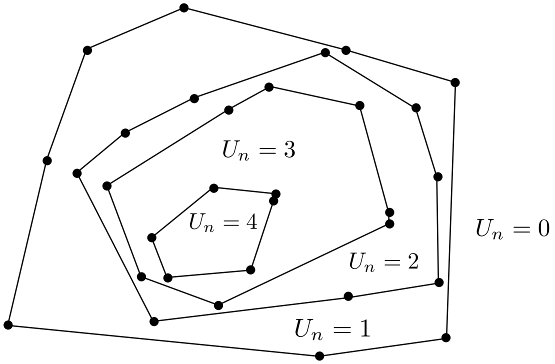


$n = 10^5$ points

A PDE continuum limit for convex hull peeling

Let $X_1, \ldots, X_n$ be i.i.d. with a continuous density f on a convex set $\Omega \subset \mathbb{R}^d$.

Let U_n be the function that 'counts' the associated convex layers $\mathcal{C}_1, \mathcal{C}_2, \ldots$



Partial differential equation (PDE) continuum limit

Theorem (Joint with C. Smart)

There exists a universal constant α_d *such that with probability one*

$$n^{-\frac{2}{d+1}} U_n \longrightarrow \alpha_d u \quad \textit{uniformly on } \Omega,$$

where $u \in C(\overline{\Omega})$ *is the unique viscosity solution of*

$$\begin{cases} \nabla u \cdot \text{cof}(-\nabla^2 u)\nabla u = f^2 & \textit{in } \Omega \\ u = 0 & \textit{on } \partial\Omega. \end{cases} \tag{1}$$

Partial differential equation (PDE) continuum limit

Theorem (Joint with C. Smart)

There exists a universal constant α_d *such that with probability one*

$$n^{-\frac{2}{d+1}} U_n \longrightarrow \alpha_d u \quad \textit{uniformly on } \Omega,$$

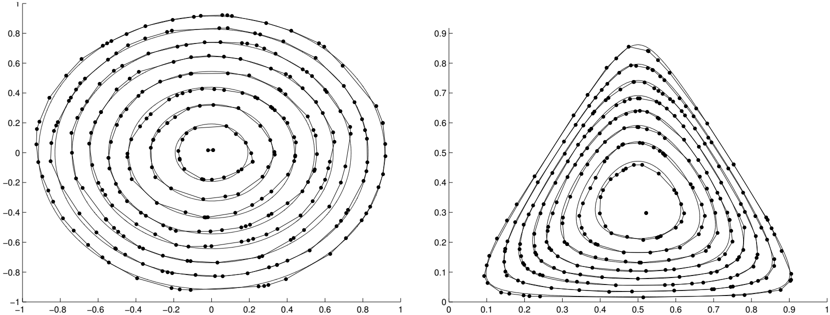
where $u \in C(\overline{\Omega})$ *is the unique viscosity solution of*

$$\begin{cases} \nabla u \cdot \mathrm{cof}(-\nabla^2 u)\nabla u = f^2 & \textit{in } \Omega \\ u = 0 & \textit{on } \partial\Omega. \end{cases} \tag{1}$$

This is just motion by a power of Gauss curvature

$$\frac{dS}{dt} = f^{-2/(d+1)} \kappa_G^{1/(d+1)} \mathbf{n}.$$

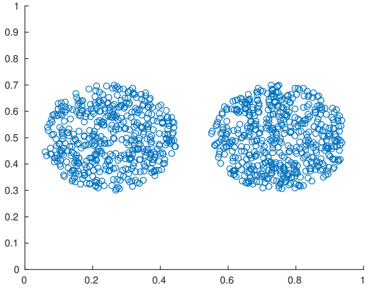
A PDE continuum limit for convex hull peeling



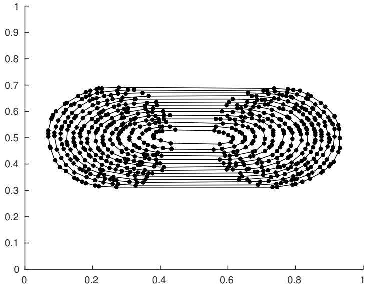
(c) (d)

Figure: Convex layers vs continuum limit for $n = 5 \times 10^3$.

A nonconvex example



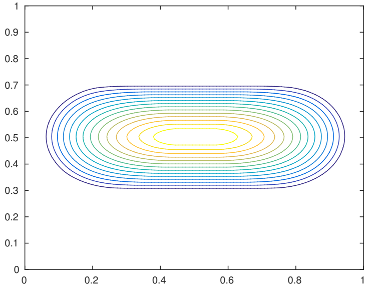
(a) Samples



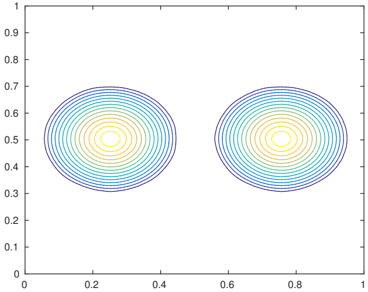
(b) Convex layers

Figure: Convex layers corresponding to disjoint clusters.

A nonconvex example



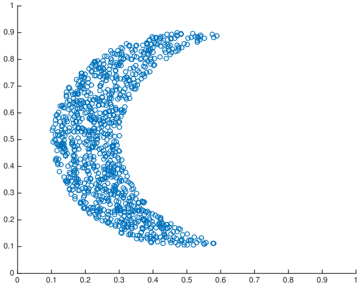
(a) One solution



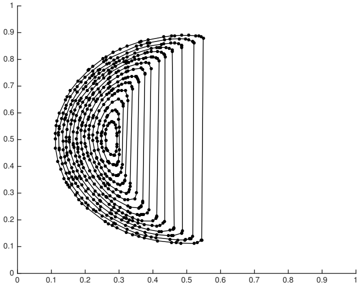
(b) Another solution

Figure: Two different solutions continuum PDE.

The halfmoon



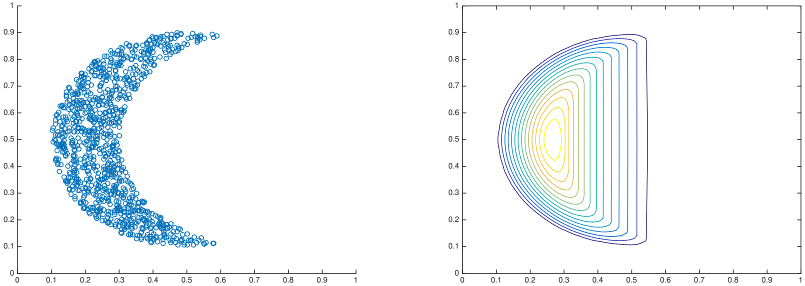
(a) Samples



(b) Convex layers

Figure: Convex layers corresponding to the halfmoon distribution.

The halfmoon



(a) Samples (b) PDE

Figure: Solution of PDE for the halfmoon example.

Outline

Quick intro to learning

Fully supervised: In fully supervised learning, we are given training data (x_i, y_i) for $i = 1, \ldots, n$, where $x_i \in \mathcal{X}$ are the data points and $y_i \in \mathcal{Y}$ are the known labels.

Quick intro to learning

Fully supervised: In fully supervised learning, we are given training data (x_i, y_i) for $i = 1, \ldots, n$, where $x_i \in \mathcal{X}$ are the data points and $y_i \in \mathcal{Y}$ are the known labels. The goal is to learn a function

$$u : \mathcal{X} \to \mathcal{Y} \quad \text{for which } u(x_i) \approx y_i \text{ for } i = 1, \ldots, n. \tag{2}$$

Quick intro to learning

Fully supervised: In fully supervised learning, we are given training data (x_i, y_i) for $i = 1, \ldots, n$, where $x_i \in \mathcal{X}$ are the data points and $y_i \in \mathcal{Y}$ are the known labels. The goal is to learn a function

$$u : \mathcal{X} \to \mathcal{Y} \quad \text{for which } u(x_i) \approx y_i \text{ for } i = 1, \ldots, n. \tag{2}$$

Semi-supervised learning: In semi-supervised learning, we are additionally given a (usually large) amount of unlabeled data $x_{n+1}, \ldots, x_{n+m}$ for $m \geq 1$.

Quick intro to learning

Fully supervised: In fully supervised learning, we are given training data (x_i, y_i) for $i = 1, \ldots, n$, where $x_i \in \mathcal{X}$ are the data points and $y_i \in \mathcal{Y}$ are the known labels. The goal is to learn a function

$$u : \mathcal{X} \to \mathcal{Y} \quad \text{for which } u(x_i) \approx y_i \text{ for } i = 1, \ldots, n. \tag{2}$$

Semi-supervised learning: In semi-supervised learning, we are additionally given a (usually large) amount of unlabeled data $x_{n+1}, \ldots, x_{n+m}$ for $m \geq 1$. Goal is to use the unlabeled data to aid the learning.

Quick intro to learning

Fully supervised: In fully supervised learning, we are given training data (x_i, y_i) for $i = 1, \ldots, n$, where $x_i \in \mathcal{X}$ are the data points and $y_i \in \mathcal{Y}$ are the known labels. The goal is to learn a function

$$u : \mathcal{X} \to \mathcal{Y} \quad \text{for which } u(x_i) \approx y_i \text{ for } i = 1, \ldots, n. \tag{2}$$

Semi-supervised learning: In semi-supervised learning, we are additionally given a (usually large) amount of unlabeled data $x_{n+1}, \ldots, x_{n+m}$ for $m \geq 1$. Goal is to use the unlabeled data to aid the learning.

1. **Inductive learning:** Learn a function

$$u : \mathcal{X} \to \mathcal{Y} \quad \text{for which } u(x_i) \approx y_i \text{ for } i = 1, \ldots, n.$$

Quick intro to learning

Fully supervised: In fully supervised learning, we are given training data (x_i, y_i) for $i = 1, \ldots, n$, where $x_i \in \mathcal{X}$ are the data points and $y_i \in \mathcal{Y}$ are the known labels. The goal is to learn a function

$$u : \mathcal{X} \to \mathcal{Y} \quad \text{for which } u(x_i) \approx y_i \text{ for } i = 1, \ldots, n. \tag{2}$$

Semi-supervised learning: In semi-supervised learning, we are additionally given a (usually large) amount of unlabeled data $x_{n+1}, \ldots, x_{n+m}$ for $m \geq 1$. Goal is to use the unlabeled data to aid the learning.

1. **Inductive learning:** Learn a function

$$u : \mathcal{X} \to \mathcal{Y} \quad \text{for which } u(x_i) \approx y_i \text{ for } i = 1, \ldots, n.$$

2. **Transductive learning:** Learn a function

$$u : \{x_1, x_2, \ldots, x_{n+m}\} \to \mathcal{Y} \quad \text{for which } u(x_i) \approx y_i \text{ for } i = 1, \ldots, n$$

Quick intro to learning

Fully supervised: In fully supervised learning, we are given training data (x_i, y_i) for $i = 1, \dots, n$, where $x_i \in \mathcal{X}$ are the data points and $y_i \in \mathcal{Y}$ are the known labels. The goal is to learn a function

$$u : \mathcal{X} \to \mathcal{Y} \quad \text{for which } u(x_i) \approx y_i \text{ for } i = 1, \dots, n. \tag{2}$$

Semi-supervised learning: In semi-supervised learning, we are additionally given a (usually large) amount of unlabeled data $x_{n+1}, \dots, x_{n+m}$ for $m \geq 1$. Goal is to use the unlabeled data to aid the learning.

1. **Inductive learning:** Learn a function

$$u : \mathcal{X} \to \mathcal{Y} \quad \text{for which } u(x_i) \approx y_i \text{ for } i = 1, \dots, n.$$

2. **Transductive learning:** Learn a function

$$u : \{x_1, x_2, \dots, x_{n+m}\} \to \mathcal{Y} \quad \text{for which } u(x_i) \approx y_i \text{ for } i = 1, \dots, n$$

Classification when $\mathcal{Y}$ finite – Regression when $\mathcal{Y} = \mathbb{R}^d$.

Example: Automated image captioning

Example: Automated image captioning

A woman is throwing a **frisbee** in a park.

A **dog** is standing on a hardwood floor.

A **stop** sign is on a road with a mountain in the background

A little **girl** sitting on a bed with a teddy bear.

A group of **people** sitting on a boat in the water.

A giraffe standing in a forest with **trees** in the background.

[Yann LeCun, Yoshua Bengio, Geoffrey Hinton. Deep learning. Nature, 2015.]

Example: Automated image captioning fail

[Andrej Karpathy's NeuralTalk]

Applications

Why is semi-supervised learning useful?

Applications

Why is semi-supervised learning useful?

It is expensive to label data, and we have an abundance of unlabeled data.

Applications

Why is semi-supervised learning useful?

It is expensive to label data, and we have an abundance of unlabeled data.

Brief list of example applications:

1. Speech recognition

Applications

Why is semi-supervised learning useful?

It is expensive to label data, and we have an abundance of unlabeled data.

Brief list of example applications:

1. Speech recognition
2. Webpage classification

Applications

Why is semi-supervised learning useful?

It is expensive to label data, and we have an abundance of unlabeled data.

Brief list of example applications:

1. Speech recognition
2. Webpage classification
3. Inferring protein structure from sequencing

Applications

Why is semi-supervised learning useful?

It is expensive to label data, and we have an abundance of unlabeled data.

Brief list of example applications:

1. Speech recognition
2. Webpage classification
3. Inferring protein structure from sequencing

A great introductory book [Chapelle et al., 2006].

Graph-based semi-supervised learning

Model:

1. Data (labeled and unlabeled) is a graph $(\mathcal{X}, \mathcal{W})$.

Graph-based semi-supervised learning

Model:

1. Data (labeled and unlabeled) is a graph $(\mathcal{X}, \mathcal{W})$.
   - $\mathcal{X} \subset \mathbb{R}^d$ are the vertices and
   - $\mathcal{W} = (w_{xy})_{x,y\in\mathcal{X}}$ are the nonnegative edge weights.
   - $w_{xy} \approx 1$ if x, y similar, and $w_{xy} \approx 0$ when dissimilar.

Graph-based semi-supervised learning

Model:

1. Data (labeled and unlabeled) is a graph $(\mathcal{X}, \mathcal{W})$.
   - $\mathcal{X} \subset \mathbb{R}^d$ are the vertices and
   - $\mathcal{W} = (w_{xy})_{x,y\in\mathcal{X}}$ are the nonnegative edge weights.
   - $w_{xy} \approx 1$ if x, y similar, and $w_{xy} \approx 0$ when dissimilar.
2. Labeled (or observed) vertices are a subset $\mathcal{O} \subset \mathcal{X}$.

Graph-based semi-supervised learning

Model:

1. Data (labeled and unlabeled) is a graph $(\mathcal{X}, \mathcal{W})$.
   - $\mathcal{X} \subset \mathbb{R}^d$ are the vertices and
   - $\mathcal{W} = (w_{xy})_{x,y \in \mathcal{X}}$ are the nonnegative edge weights.
   - $w_{xy} \approx 1$ if x, y similar, and $w_{xy} \approx 0$ when dissimilar.
2. Labeled (or observed) vertices are a subset $\mathcal{O} \subset \mathcal{X}$.
3. We given a labelling function $g : \mathcal{O} \to \mathbb{R}$.

Graph-based semi-supervised learning

Model:

1. Data (labeled and unlabeled) is a graph $(\mathcal{X}, \mathcal{W})$.
   - $\mathcal{X} \subset \mathbb{R}^d$ are the vertices and
   - $\mathcal{W} = (w_{xy})_{x,y\in\mathcal{X}}$ are the nonnegative edge weights.
   - $w_{xy} \approx 1$ if x, y similar, and $w_{xy} \approx 0$ when dissimilar.
2. Labeled (or observed) vertices are a subset $\mathcal{O} \subset \mathcal{X}$.
3. We given a labelling function $g : \mathcal{O} \to \mathbb{R}$.

Task: Extend the labels from $\mathcal{O}$ to the entire graph $\mathcal{X}$.

Graph-based semi-supervised learning

Model:

1. Data (labeled and unlabeled) is a graph $(\mathcal{X}, \mathcal{W})$.
   - $\mathcal{X} \subset \mathbb{R}^d$ are the vertices and
   - $\mathcal{W} = (w_{xy})_{x,y\in\mathcal{X}}$ are the nonnegative edge weights.
   - $w_{xy} \approx 1$ if x, y similar, and $w_{xy} \approx 0$ when dissimilar.
2. Labeled (or observed) vertices are a subset $\mathcal{O} \subset \mathcal{X}$.
3. We given a labelling function $g : \mathcal{O} \to \mathbb{R}$.

Task: Extend the labels from $\mathcal{O}$ to the entire graph $\mathcal{X}$.

Semi-supervised smoothness assumption

Similar points $x, y \in \mathcal{X}$ in high density regions of the graph should have similar labels.

Laplacian regularization

$$\min_{u:\mathcal{X}\to\mathbb{R}} \sum_{x,y\in\mathcal{X}} w_{xy}^2 (u(x)-u(y))^2 \quad \text{subject to } u(x)=g(x) \text{ for all } x\in\mathcal{O}.$$

Laplacian regularization

$$\min_{u:\mathcal{X}\to\mathbb{R}} \sum_{x,y\in\mathcal{X}} w_{xy}^2 (u(x)-u(y))^2 \quad \text{subject to } u(x) = g(x) \text{ for all } x \in \mathcal{O}.$$

The minimizer $u : \mathcal{X} \to \mathbb{R}$ satisfies the linear system

$$\sum_{y\in\mathcal{X}} w_{xy}^2 (u(x)-u(y)) = 0 \quad \text{for all } x \in \mathcal{X} \setminus \mathcal{O}.$$

Laplacian regularization

$$\min_{u:\mathcal{X}\to\mathbb{R}} \sum_{x,y\in\mathcal{X}} w_{xy}^2 (u(x)-u(y))^2 \quad \text{subject to } u(x)=g(x) \text{ for all } x\in\mathcal{O}.$$

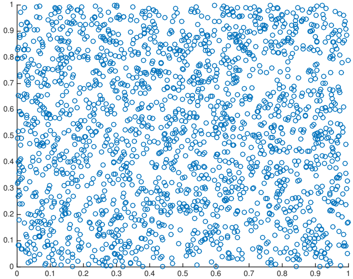
The minimizer $u : \mathcal{X} \to \mathbb{R}$ satisfies the linear system

$$\sum_{y\in\mathcal{X}} w_{xy}^2 (u(x)-u(y)) = 0 \quad \text{for all } x\in\mathcal{X}\setminus\mathcal{O}.$$

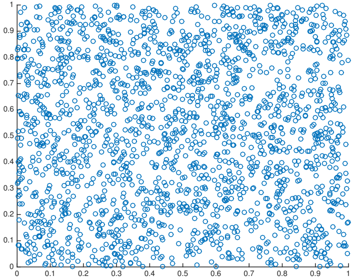
References:

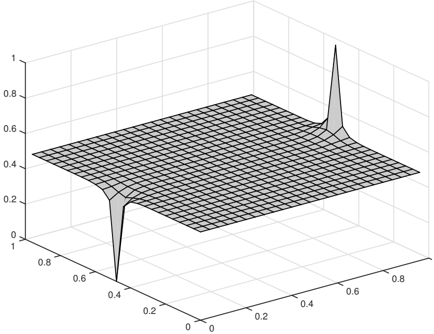
- Original work [Zhu et al., 2003]
- Learning [Zhou et al., 2005][Ando and Zhang, 2007]
- Manifold ranking [He et al., 2006] [Wang et al., 2013] [Yang et al., 2013] [Zhou et al., 2011] [Xu et al., 2011]

Ill-posed with small amount of labeled data



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.
- Over 95% of labels in $[0.4975, 0.5025]$.

[Nadler et al., 2009][El Alaoui et al., 2016]

ℓ_p-based Laplacian regularization

For any $p < \infty$:

$$\min_{u:\mathcal{X}\to\mathbb{R}} \sum_{x,y\in\mathcal{X}} w_{xy}^p |u(x) - u(y)|^p \quad \text{subject to } u(x) = g(x) \text{ for all } x \in \mathcal{O}. \tag{3}$$

ℓ_p-based Laplacian regularization

For any $p < \infty$:

$$\min_{u:\mathcal{X}\to\mathbb{R}} \sum_{x,y\in\mathcal{X}} w_{xy}^p |u(x) - u(y)|^p \quad \text{subject to } u(x) = g(x) \text{ for all } x \in \mathcal{O}. \tag{3}$$

We can send $p \to \infty$:

$$\min_{u:\mathcal{X}\to\mathbb{R}} \max_{x,y\in\mathcal{X}} \{w_{xy}|u(x) - u(y)|\} \quad \text{subject to } u(x) = g(x) \text{ for all } x \in \mathcal{O}. \tag{4}$$

ℓ_p-based Laplacian regularization

For any $p < \infty$:

$$\min_{u:\mathcal{X}\to\mathbb{R}} \sum_{x,y\in\mathcal{X}} w_{xy}^p |u(x) - u(y)|^p \quad \text{subject to } u(x) = g(x) \text{ for all } x \in \mathcal{O}. \tag{3}$$

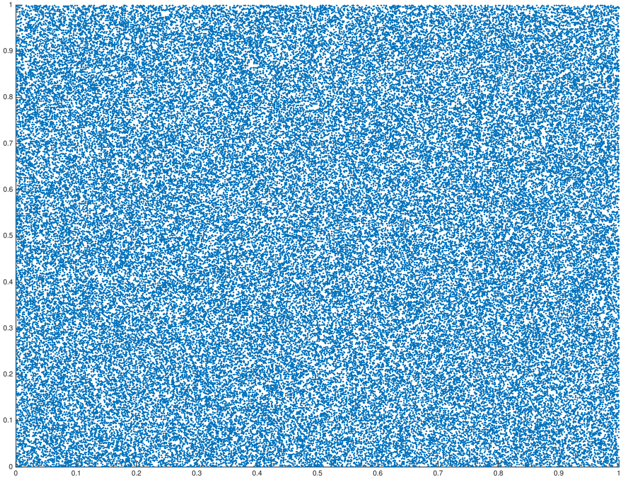
We can send $p \to \infty$:

$$\min_{u:\mathcal{X}\to\mathbb{R}} \max_{x,y\in\mathcal{X}} \{w_{xy}|u(x) - u(y)|\} \quad \text{subject to } u(x) = g(x) \text{ for all } x \in \mathcal{O}. \tag{4}$$

References:

- Finite p: [Bridle and Zhu, 2013][Alamgir and Luxburg, 2011]
- $p = \infty$: [Kyng et al., 2015] [Luxburg and Bousquet, 2004]
- Absolutely minimal Lipschitz extensions: [Aronsson et al., 2004]

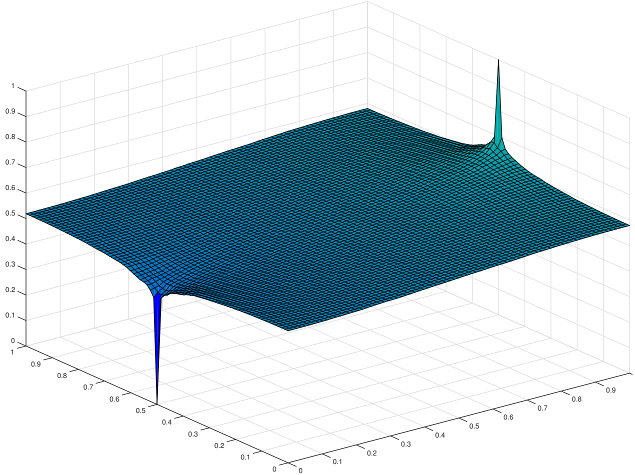
p-Laplacian learning: $n = 10^5$ points, $h = 10^{-2}$



$p = 2$

Simulations are the work of Mauricio Flores (co-supervised by Gilad Lerman).

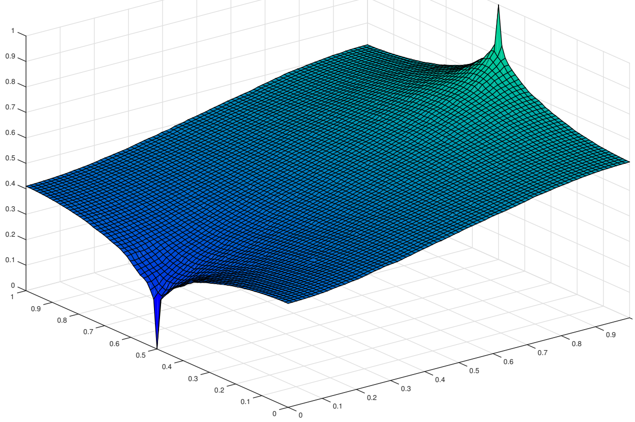
p-Laplacian learning: $n = 10^5$ points, $h = 10^{-2}$



$p = 2$

Simulations are the work of Mauricio Flores (co-supervised by Gilad Lerman).

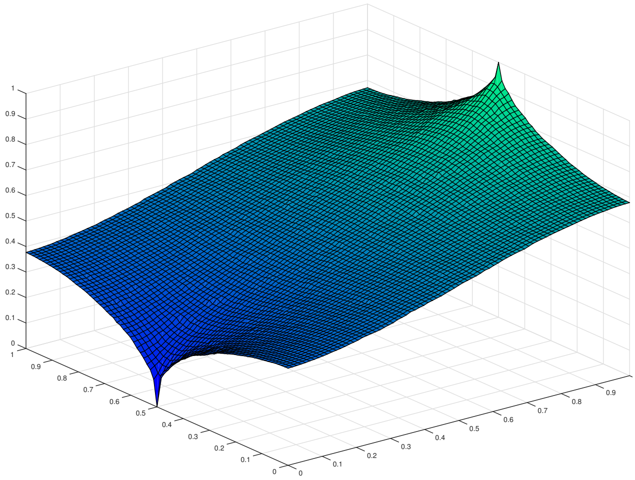
p-Laplacian learning: $n = 10^5$ points, $h = 10^{-2}$



$p = 2.5$

Simulations are the work of Mauricio Flores (co-supervised by Gilad Lerman).

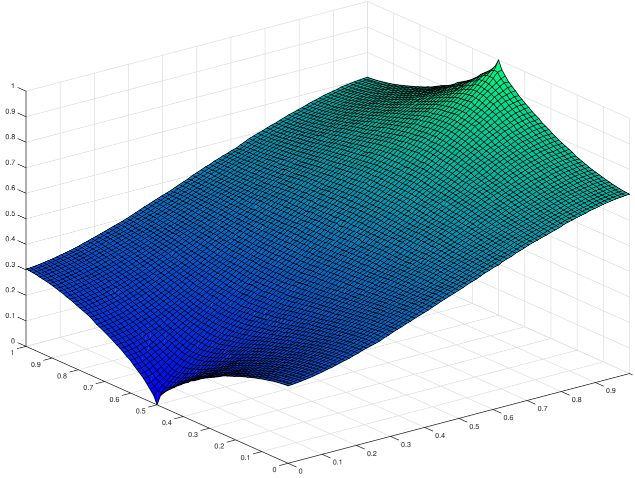
p-Laplacian learning: $n = 10^5$ points, $h = 10^{-2}$



$p = 3$

Simulations are the work of Mauricio Flores (co-supervised by Gilad Lerman).

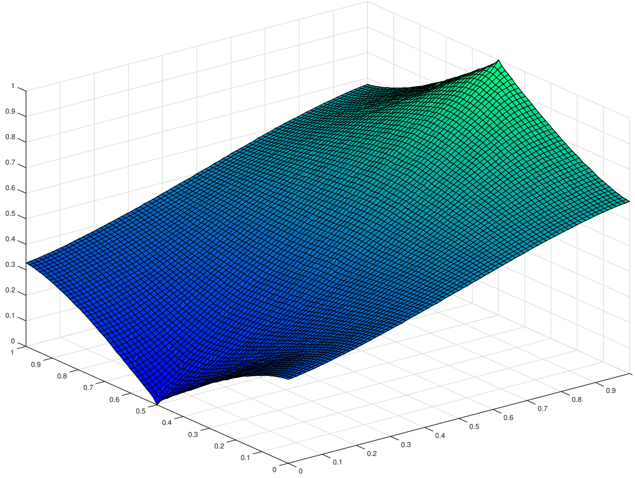
p-Laplacian learning: $n = 10^5$ points, $h = 10^{-2}$



$p = 5$

Simulations are the work of Mauricio Flores (co-supervised by Gilad Lerman).

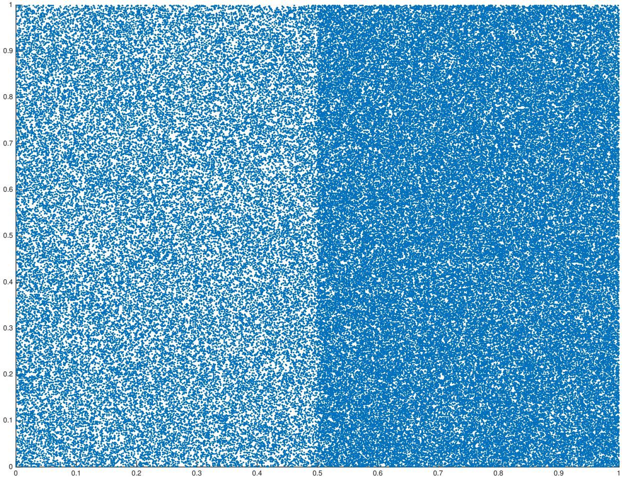
p-Laplacian learning: $n = 10^5$ points, $h = 10^{-2}$



$$p = \infty$$

Simulations are the work of Mauricio Flores (co-supervised by Gilad Lerman).

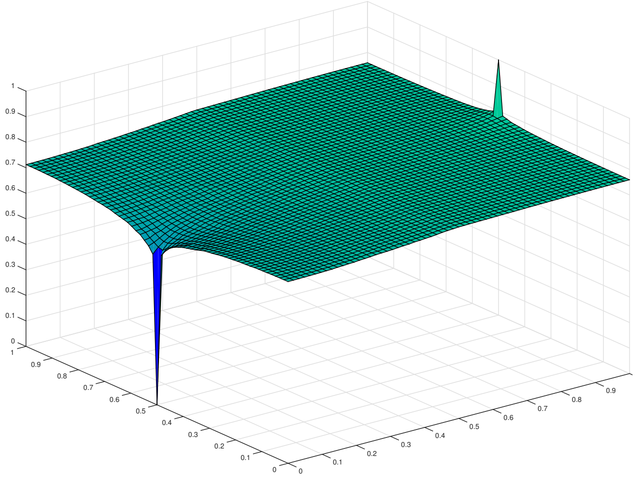
p-Laplacian learning: Varying density



$p = 2$

Simulations are the work of Mauricio Flores (co-supervised by Gilad Lerman).

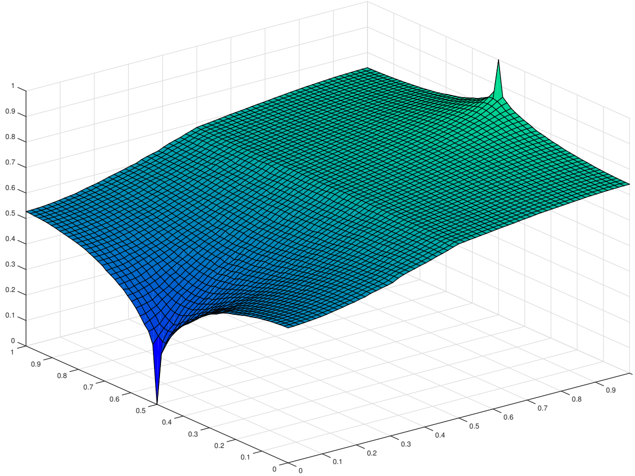
p-Laplacian learning: Varying density



$$p = 2$$

Simulations are the work of Mauricio Flores (co-supervised by Gilad Lerman).

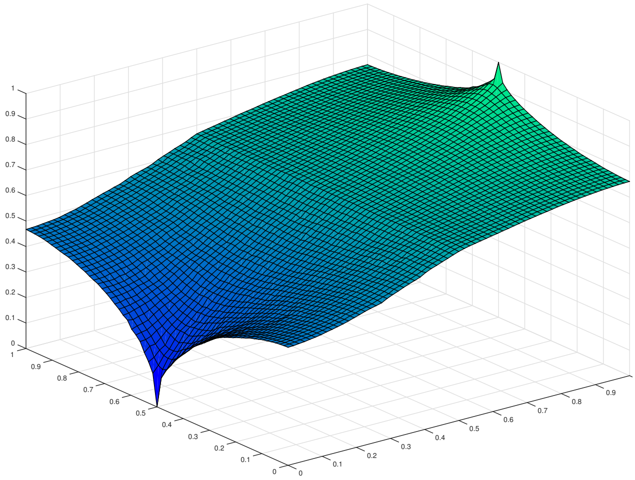
p-Laplacian learning: Varying density



$p = 2.5$

Simulations are the work of Mauricio Flores (co-supervised by Gilad Lerman).

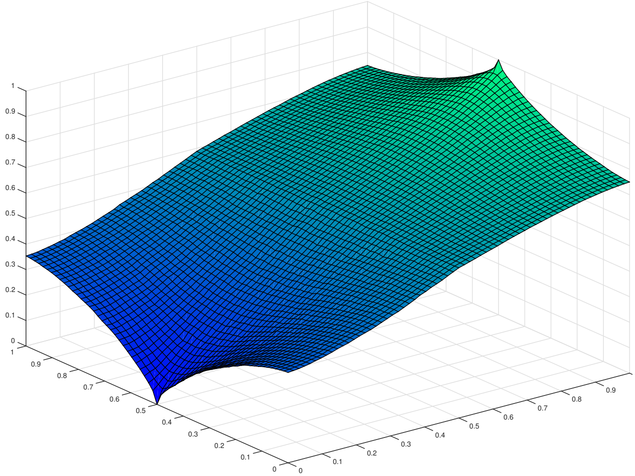
p-Laplacian learning: Varying density



$p = 3$

Simulations are the work of Mauricio Flores (co-supervised by Gilad Lerman).

p-Laplacian learning: Varying density



$$p = 5$$

Simulations are the work of Mauricio Flores (co-supervised by Gilad Lerman).

Random model

- **Labeled data:** The labeled data is a fixed finite collection of N points

$$\mathcal{O} = \{y_1, \ldots, y_N\} \subset U \subset \mathbb{T}^d := \mathbb{R}^d/\mathbb{Z}^d.$$

Random model

- **Labeled data:** The labeled data is a fixed finite collection of N points

$$\mathcal{O} = \{y_1, \ldots, y_N\} \subset U \subset \mathbb{T}^d := \mathbb{R}^d/\mathbb{Z}^d.$$

- **Unlabeled data:** The unlabeled data is a sequence $x_1, x_2, \ldots, x_n$ of i.i.d. random variables with probability density $f : \mathbb{T}^d \to \mathbb{R}$

$$X_{nf} := \{x_1, x_2, \ldots, x_n\}.$$

Random model

- **Labeled data:** The labeled data is a fixed finite collection of N points

$$\mathcal{O} = \{y_1, \ldots, y_N\} \subset U \subset \mathbb{T}^d := \mathbb{R}^d/\mathbb{Z}^d.$$

- **Unlabeled data:** The unlabeled data is a sequence $x_1, x_2, \ldots, x_n$ of i.i.d. random variables with probability density $f : \mathbb{T}^d \to \mathbb{R}$

$$X_{nf} := \{x_1, x_2, \ldots, x_n\}.$$

- **Vertices of graph:** The vertices of the graph are

$$\mathcal{X}_n = X_{nf} \cup \mathcal{O}.$$

Random model

- **Labeled data:** The labeled data is a fixed finite collection of N points

$$\mathcal{O} = \{y_1, \ldots, y_N\} \subset U \subset \mathbb{T}^d := \mathbb{R}^d/\mathbb{Z}^d.$$

- **Unlabeled data:** The unlabeled data is a sequence $x_1, x_2, \ldots, x_n$ of i.i.d. random variables with probability density $f : \mathbb{T}^d \to \mathbb{R}$

$$X_{nf} := \{x_1, x_2, \ldots, x_n\}.$$

- **Vertices of graph:** The vertices of the graph are

$$\mathcal{X}_n = X_{nf} \cup \mathcal{O}.$$

- **Edge weights:** The edge weights are

$$w_{xy} = \Phi\left(\frac{|x-y|}{h}\right),$$

where $h > 0$, and $\Phi : [0,\infty) \to [0,\infty)$.

Random model

For $p < \infty$ we write

$$J_p(u) := \sum_{x,y \in \mathcal{X}_n} w_{xy}^p |u(x) - u(y)|^p,$$

and for $p = \infty$ we write

$$J_\infty(u) := \max_{x,y \in \mathcal{X}_n} \{w_{xy}|u(x) - u(y)|\}.$$

Random model

For $p < \infty$ we write

$$J_p(u) := \sum_{x,y \in \mathcal{X}_n} w_{xy}^p |u(x) - u(y)|^p,$$

and for $p = \infty$ we write

$$J_\infty(u) := \max_{x,y \in \mathcal{X}_n} \{w_{xy}|u(x) - u(y)|\}.$$

For $n \geq 1$, let $u_n : \mathcal{X}_n \to \mathbb{R}$ be the solution of

$$\boxed{\min_{u:\mathcal{X}_n \to \mathbb{R}} J_p(u) \quad \text{subject to } u(x) = g(x) \text{ for all } x \in \mathcal{O}.}$$

Random model

For $p < \infty$ we write

$$J_p(u) := \sum_{x,y \in \mathcal{X}_n} w_{xy}^p |u(x) - u(y)|^p,$$

and for $p = \infty$ we write

$$J_\infty(u) := \max_{x,y \in \mathcal{X}_n} \{w_{xy}|u(x) - u(y)|\}.$$

For $n \geq 1$, let $u_n : \mathcal{X}_n \to \mathbb{R}$ be the solution of

$$\boxed{\min_{u:\mathcal{X}_n \to \mathbb{R}} J_p(u) \quad \text{subject to } u(x) = g(x) \text{ for all } x \in \mathcal{O}.}$$

Question: What can we say about u_n as $n \to \infty$?

Let

$$r_n = \sup \{ s > 0 \,|\, B(x,s) \cap \mathcal{X}_n = \varnothing \text{ for some } x \in U \}. \tag{5}$$

Theorem ($p = \infty$ [Calder, 2017a])

Suppose that $h_n \to 0$ *such that*

$$\lim_{n\to\infty} \frac{r_n^2}{h_n^3} = 0. \tag{6}$$

$$\textit{Then } u_n \longrightarrow u \quad \textit{uniformly as} \quad n \to \infty, \tag{7}$$

where $u \in C(\mathbb{T}^d)$ *is the unique viscosity solution of the* ∞*-Laplace equation*

$$\begin{cases} \Delta_\infty u = 0 & \text{in } \mathbb{T}^d \setminus \mathcal{O} \\ u = g & \text{on } \mathcal{O} \end{cases} \tag{8}$$

Note that (6) holds almost surely when

$$\lim_{n\to\infty} \frac{n h_n^{3d/2}}{\log(n)} = \infty. \tag{9}$$

Theorem (Finite p [Calder, 2017b])

Let $d < p < \infty$, and suppose that $h_n \to 0$ such that

$$\lim_{n\to\infty} nh_n^p = 0 \quad \textit{and} \quad \lim_{n\to\infty} \frac{nh_n^{d+4}}{\log(n)} = \infty. \tag{10}$$

Then with probability one

$$u_n \longrightarrow u \quad \textit{uniformly as} \quad n \to \infty, \tag{11}$$

where $u \in C(\mathbb{T}^d)$ is the unique viscosity solution of the weighted p-Laplace equation

$$\begin{cases} \textit{div}\left(f^2|\nabla u|^{p-2}\nabla u\right) = 0 & \textit{in } \mathbb{T}^d \setminus \mathcal{O} \\ u = g & \textit{on } \mathcal{O} \end{cases} \tag{12}$$

A very similar result appeared recently in [Slepčev and Thorpe, 2017].

Regularity in semi-supervised learning

The PDE-limit can be used to prove Hölder regularity.

Theorem

Assume $p > d$. For every $\alpha < \frac{p-d}{p-1}$ there exists C, δ such that

$$\mathbb{P}\left[\forall x, y \in \mathcal{X}_n,\ |u_n(x) - u_n(y)| \le C(|x-y|^\alpha + n^{\frac{1}{p}} h)\right] \ge 1 - \exp\left(-\delta n h^{d+4} + C\log(n)\right).$$

Graph Laplacians

$$\min_{u:\mathcal{X}_n\to\mathbb{R}} J_p(u) = \sum_{x,y\in\mathcal{X}_n} w_{xy}^p |u(x)-u(y)|^p \quad \text{subject to } u(x)=g(x) \text{ for } x\in\mathcal{O}\subset\mathcal{X}_n$$

Graph Laplacians

$$\min_{u:\mathcal{X}_n\to\mathbb{R}} J_p(u) = \sum_{x,y\in\mathcal{X}_n} w_{xy}^p |u(x)-u(y)|^p \quad \text{subject to } u(x)=g(x) \text{ for } x\in\mathcal{O}\subset\mathcal{X}_n$$

The minimizer $u:\mathcal{X}_n\to\mathbb{R}$ satisfies

$$\begin{cases} \Delta_p^{\mathcal{X}_n} u = 0 & \text{in } \mathcal{X}_n\setminus\mathcal{O}, \\ u = g & \text{on } \mathcal{O}, \end{cases}$$

where $\Delta_p^{\mathcal{X}_n} u : \mathcal{X} \to \mathbb{R}$ is the graph p-Laplacian defined by

$$\Delta_p^{\mathcal{X}_n} u(x) = \sum_{y\in\mathcal{X}_n} w_{xy}^p |u(y)-u(x)|^{p-2}(u(y)-u(x)).$$

Graph Laplacians

$$\min_{u:\mathcal{X}_n\to\mathbb{R}} J_p(u) = \sum_{x,y\in\mathcal{X}_n} w_{xy}^p |u(x)-u(y)|^p \quad \text{subject to } u(x)=g(x) \text{ for } x\in\mathcal{O}\subset\mathcal{X}_n$$

The minimizer $u:\mathcal{X}_n\to\mathbb{R}$ satisfies

$$\begin{cases} \Delta_p^{\mathcal{X}_n} u = 0 & \text{in } \mathcal{X}_n\setminus\mathcal{O}, \\ u = g & \text{on } \mathcal{O}, \end{cases}$$

where $\Delta_p^{\mathcal{X}_n} u : \mathcal{X}\to\mathbb{R}$ is the graph p-Laplacian defined by

$$\Delta_p^{\mathcal{X}_n} u(x) = \sum_{y\in\mathcal{X}_n} w_{xy}^p |u(y)-u(x)|^{p-2}(u(y)-u(x)).$$

References on graph p-Laplacian:

- [Manfredi et al., 2015] [Zhou and Schölkopf, 2005] [Amghibech, 2003] [Bühler and Hein, 2009] [Luo et al., 2010]

Graph Laplacian as $p \to \infty$

Note that solutions of

$$\Delta_p^{\mathcal{X}_n} u(x) = \sum_{y \in \mathcal{X}_n} w_{xy}^p |u(y) - u(x)|^{p-2}(u(y) - u(x)) = 0$$

satisfy

$$\left(\sum_{\substack{y \in \mathcal{X}_n \\ u(y) \geq u(x)}} w_{xy}^p |u(y) - u(x)|^{p-1} \right)^{1/p} = \left(\sum_{\substack{y \in \mathcal{X}_n \\ u(y) < u(x)}} w_{xy}^p |u(y) - u(x)|^{p-1} \right)^{1/p}.$$

Graph Laplacian as $p \to \infty$

Note that solutions of

$$\Delta_p^{\mathcal{X}_n} u(x) = \sum_{y \in \mathcal{X}_n} w_{xy}^p |u(y) - u(x)|^{p-2}(u(y) - u(x)) = 0$$

satisfy

$$\left(\sum_{\substack{y \in \mathcal{X}_n \\ u(y) \geq u(x)}} w_{xy}^p |u(y) - u(x)|^{p-1} \right)^{1/p} = \left(\sum_{\substack{y \in \mathcal{X}_n \\ u(y) < u(x)}} w_{xy}^p |u(y) - u(x)|^{p-1} \right)^{1/p}.$$

Send $p \to \infty$ to get

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

or

$$\Delta_\infty^{\mathcal{X}_n} u(x) := \max_{y \in \mathcal{X}_n} w_{xy}(u(y) - u(x)) + \min_{y \in \mathcal{X}_n} w_{xy}(u(y) - u(x)) = 0.$$

Graph Laplacians

$$\min_{u:\mathcal{X}_n\to\mathbb{R}} J_\infty(u) = \max_{x,y\in\mathcal{X}_n} w_{xy}|u(x)-u(y)| \quad \text{subject to } u(x)=g(x) \text{ for } x\in\mathcal{O}\subset\mathcal{X}_n$$

Graph Laplacians

$$\min_{u:\mathcal{X}_n\to\mathbb{R}} J_\infty(u) = \max_{x,y\in\mathcal{X}_n} w_{xy}|u(x)-u(y)| \quad \text{subject to } u(x)=g(x) \text{ for } x\in\mathcal{O}\subset\mathcal{X}_n$$

The minimizer $u:\mathcal{X}_n\to\mathbb{R}$ satisfies

$$\begin{cases} \Delta_\infty^{\mathcal{X}_n} u = 0 & \text{in } \mathcal{X}_n\setminus\mathcal{O} \\ u = g & \text{in } \mathcal{O}, \end{cases}$$

where $\Delta_\infty^{\mathcal{X}_n} u:\mathcal{X}_n\to\mathbb{R}$ is the graph ∞-Laplacian defined by

$$\Delta_\infty^{\mathcal{X}_n} u(x) = \max_{y\in\mathcal{X}_n} w_{xy}(u(y)-u(x)) + \min_{y\in\mathcal{X}_n} w_{xy}(u(y)-u(x))$$

Graph Laplacians

$$\min_{u:\mathcal{X}_n\to\mathbb{R}} J_\infty(u) = \max_{x,y\in\mathcal{X}_n} w_{xy}|u(x)-u(y)| \quad \text{subject to } u(x)=g(x) \text{ for } x\in\mathcal{O}\subset\mathcal{X}_n$$

The minimizer $u:\mathcal{X}_n\to\mathbb{R}$ satisfies

$$\begin{cases} \Delta_\infty^{\mathcal{X}_n} u = 0 & \text{in } \mathcal{X}_n\setminus\mathcal{O} \\ u = g & \text{in } \mathcal{O}, \end{cases}$$

where $\Delta_\infty^{\mathcal{X}_n} u:\mathcal{X}_n\to\mathbb{R}$ is the graph ∞-Laplacian defined by

$$\Delta_\infty^{\mathcal{X}_n} u(x) = \max_{y\in\mathcal{X}_n} w_{xy}(u(y)-u(x)) + \min_{y\in\mathcal{X}_n} w_{xy}(u(y)-u(x))$$

Reference:

1. [Kyng et al., 2015]

Game theoretic p-Lapacian

We can also consider the game theoretic p-Laplacian for semi-supervised learning:

$$\begin{cases} \dfrac{1}{d_n}\Delta_2^{\mathcal{X}_n} u_n + \lambda(p-2)\Delta_\infty^{\mathcal{X}_n} u_n = 0 & \text{in } \mathcal{X}_n \setminus \mathcal{O} \\ u = g & \text{in } \mathcal{O}, \end{cases}$$

where $d_n(x) = \sum_{y\in\mathcal{X}_n} w_{xy}^2$ and $\lambda = \lambda(\Phi)$.

Game theoretic p-Lapacian

We can also consider the game theoretic p-Laplacian for semi-supervised learning:

$$\begin{cases} \frac{1}{d_n}\Delta_2^{\mathcal{X}_n} u_n + \lambda(p-2)\Delta_\infty^{\mathcal{X}_n} u_n = 0 & \text{in } \mathcal{X}_n \setminus \mathcal{O} \\ u = g & \text{in } \mathcal{O}, \end{cases}$$

where $d_n(x) = \sum_{y \in \mathcal{X}_n} w_{xy}^2$ and $\lambda = \lambda(\Phi)$.

This is likely better conditioned numerically when p is large.

Game theoretic p-Laplacian

Theorem (Finite p [Calder, 2017b])

Let $d < p < \infty$, and suppose that $h \to 0$ such that

$$\lim_{n\to\infty} \frac{nh^q}{\log(n)} = \infty, \tag{13}$$

where $q = \max\{d+4, 3d/2\}$. Then with probability one

$$u_n \longrightarrow u \quad \textit{uniformly as} \quad n \to \infty, \tag{14}$$

where $u \in C(\mathbb{T}^d)$ is the unique viscosity solution of the weighted p-Laplace equation

$$\begin{cases} \textit{div}\left(f^2|\nabla u|^{p-2}\nabla u\right) = 0 & \textit{in } \mathbb{T}^d \setminus \mathcal{O} \\ u = g & \textit{on } \mathcal{O} \end{cases} \tag{15}$$

Notice no upper bound on h (i.e., we don't require $nh^p \to 0$).

Ideas in proof

All graph Laplacians are monotone schemes. We just need consistency and stability.

Ideas in proof

All graph Laplacians are monotone schemes. We just need consistency and stability.

Consistency is straightforward, using concentration of measure and Taylor expansions. For example, for the Graph p-Laplacian

Ideas in proof

All graph Laplacians are monotone schemes. We just need consistency and stability.

Consistency is straightforward, using concentration of measure and Taylor expansions. For example, for the Graph p-Laplacian

$$\Delta_p^{\mathcal{X}_n} u(x) = \sum_{y \in \mathcal{X}_n} w_{xy}^p |u(y) - u(x)|^{p-2}(u(y) - u(x)).$$

we have

$$\mathbb{E}[\Delta_p^{\mathcal{X}_n} \varphi(x)] = nh^d \int_{\mathbb{R}^d} \Phi(|z|)|\varphi(x + zh) - \varphi(x)|^{p-2}(\varphi(x + zh) - \varphi(x))f(x + zh)\, dz.$$

Ideas in proof

All graph Laplacians are monotone schemes. We just need consistency and stability.

Consistency is straightforward, using concentration of measure and Taylor expansions. For example, for the Graph p-Laplacian

$$\Delta_p^{\mathcal{X}_n} u(x) = \sum_{y \in \mathcal{X}_n} w_{xy}^p |u(y) - u(x)|^{p-2}(u(y) - u(x)).$$

we have

$$\mathbb{E}[\Delta_p^{\mathcal{X}_n} \varphi(x)] = nh^d \int_{\mathbb{R}^d} \Phi(|z|)|\varphi(x + zh) - \varphi(x)|^{p-2}(\varphi(x + zh) - \varphi(x))f(x + zh)\, dz.$$

Plug in Taylor expansions and plug away. . .

$$\mathbb{E}[\Delta_p^{\mathcal{X}_n} \varphi(x)] = \frac{1}{2} C_p f^{-1} \mathrm{div}(f^2 |\nabla \varphi|^{p-2} \nabla \varphi) nh^{d+p} + R(x) nh^{d+p+1},$$

where

$$|R(x)| \leq C \|\varphi\|_{C^3(\mathbb{R}^d)}^{p-1}.$$

Hölder continuity for p-Laplace equation

The maximum principle can be used to prove Hölder continuity when $p > d$:

$$\begin{cases} \Delta_p u := \mathrm{div}(|\nabla u|^{p-2}\nabla u) = 0 & \text{in } U \\ u = g & \text{on } \partial U, \end{cases}$$

Hölder continuity for p-Laplace equation

The maximum principle can be used to prove Hölder continuity when $p > d$:

$$\begin{cases} \Delta_p u := \text{div}(|\nabla u|^{p-2}\nabla u) = 0 & \text{in } U \\ u = g & \text{on } \partial U, \end{cases}$$

Let us define

$$v(x) = u(x_0) + C|x - x_0|^\alpha \qquad \text{for} \quad \alpha = \frac{p-d}{p-1}.$$

Hölder continuity for p-Laplace equation

The maximum principle can be used to prove Hölder continuity when $p > d$:

$$\begin{cases} \Delta_p u := \text{div}(|\nabla u|^{p-2}\nabla u) = 0 & \text{in } U \\ u = g & \text{on } \partial U, \end{cases}$$

Let us define

$$v(x) = u(x_0) + C|x - x_0|^\alpha \quad \text{for} \quad \alpha = \frac{p-d}{p-1}.$$

If $B(x_0, r) \subset U$ then for $C = (\max g - \min g)r^{-\alpha}$ we have

$$v(x) \geq u(x) \quad \text{for } |x - x_0| = r.$$

Hölder continuity for p-Laplace equation

The maximum principle can be used to prove Hölder continuity when $p > d$:

$$\begin{cases} \Delta_p u := \text{div}(|\nabla u|^{p-2}\nabla u) = 0 & \text{in } U \\ u = g & \text{on } \partial U, \end{cases}$$

Let us define

$$v(x) = u(x_0) + C|x - x_0|^\alpha \qquad \text{for} \quad \alpha = \frac{p-d}{p-1}.$$

If $B(x_0, r) \subset U$ then for $C = (\max g - \min g)r^{-\alpha}$ we have

$$v(x) \geq u(x) \quad \text{for } |x - x_0| = r.$$

Since $\Delta_p v(x) = 0$ for $x \neq x_0$, we can use the maximum principle to show that

$$u(x) \leq v(x) \ \text{ for all } x \in B(x_0, r).$$

Hölder continuity for p-Laplace equation

The maximum principle can be used to prove Hölder continuity when $p > d$:

$$\begin{cases} \Delta_p u := \text{div}(|\nabla u|^{p-2}\nabla u) = 0 & \text{in } U \\ u = g & \text{on } \partial U, \end{cases}$$

Let us define

$$v(x) = u(x_0) + C|x - x_0|^\alpha \quad \text{for} \quad \alpha = \frac{p-d}{p-1}.$$

If $B(x_0, r) \subset U$ then for $C = (\max g - \min g)r^{-\alpha}$ we have

$$v(x) \geq u(x) \quad \text{for } |x - x_0| = r.$$

Since $\Delta_p v(x) = 0$ for $x \neq x_0$, we can use the maximum principle to show that

$$u(x) \leq v(x) \ \text{ for all } x \in B(x_0, r).$$

It follows that

$$\boxed{u(x) - u(x_0) \leq C|x - x_0|^\alpha.}$$

It is generally not the case that

$$\Delta_p^{\mathcal{X}_n} |x|^{\frac{p-d}{p-1}} = 0.$$

It is generally not the case that

$$\Delta_p^{\mathcal{X}_n} |x|^{\frac{p-d}{p-1}} = 0.$$

Outline of regularity proof:

1. Choose $0 < \alpha < (p-d)/(p-1)$ and set $v(x) = |x-y|^\alpha$

It is generally not the case that

$$\Delta_p^{\mathcal{X}_n} |x|^{\frac{p-d}{p-1}} = 0.$$

Outline of regularity proof:

1. Choose $0 < \alpha < (p-d)/(p-1)$ and set $v(x) = |x-y|^\alpha$
2. Show that $\Delta_p^{\mathcal{X}_n} v(x) \leq 0$ for $|x-y| \geq ch$ with high probability.

It is generally not the case that

$$\Delta_p^{\mathcal{X}_n} |x|^{\frac{p-d}{p-1}} = 0.$$

Outline of regularity proof:

1. Choose $0 < \alpha < (p-d)/(p-1)$ and set $v(x) = |x-y|^\alpha$
2. Show that $\Delta_p^{\mathcal{X}_n} v(x) \leq 0$ for $|x-y| \geq ch$ with high probability.
3. Fill in the gap $|x-y| \leq ch$.

It is generally not the case that

$$\Delta_p^{\mathcal{X}_n} |x|^{\frac{p-d}{p-1}} = 0.$$

Outline of regularity proof:

1. Choose $0 < \alpha < (p-d)/(p-1)$ and set $v(x) = |x-y|^\alpha$
2. Show that $\Delta_p^{\mathcal{X}_n} v(x) \leq 0$ for $|x-y| \geq ch$ with high probability.
3. Fill in the gap $|x-y| \leq ch$.
   1. For the variational graph p-Laplacian

$$|u_n(x) - u_n(y)| \leq Cn^{1/p} h \text{ for } |x-y| \leq h.$$

It is generally not the case that

$$\Delta_p^{\mathcal{X}_n} |x|^{\frac{p-d}{p-1}} = 0.$$

Outline of regularity proof:

1. Choose $0 < \alpha < (p-d)/(p-1)$ and set $v(x) = |x-y|^\alpha$
2. Show that $\Delta_p^{\mathcal{X}_n} v(x) \leq 0$ for $|x-y| \geq ch$ with high probability.
3. Fill in the gap $|x-y| \leq ch$.
   1. For the variational graph p-Laplacian

      $$|u_n(x) - u_n(y)| \leq C n^{1/p} h \text{ for } |x-y| \leq h.$$

   2. For the game theoretic p-Laplacian, we use a different local barrier

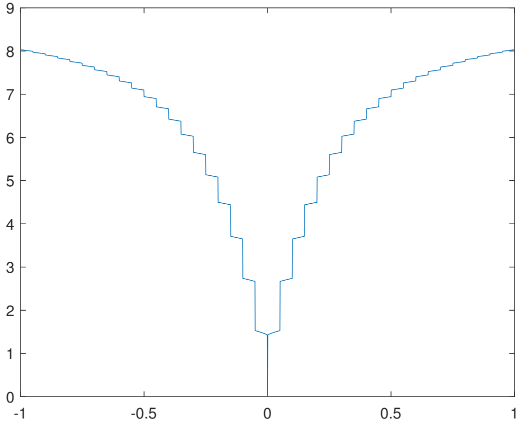
      $$v(x) = |x-y|^\alpha + M h_n^\alpha \sum_{k=1}^{\infty} \beta^k 1_{\{2|x-y| > (k-1)h_n\}}, \text{ where } \beta < 1.$$

The local barrier

$$v(x) = |x-y|^\alpha + Mh_n^\alpha \sum_{k=1}^{\infty} \beta^k 1_{\{2|x-y|>(k-1)h_n\}}$$

exploits the form of the graph ∞-Laplacian

$$\Delta_\infty^{\mathcal{X}_n} u(x) = \max_{y\in\mathcal{X}_n} w_{xy}(u(y)-u(x)) + \min_{y\in\mathcal{X}_n} w_{xy}(u(y)-u(x)).$$



Current/Future work

1. **Fast algorithms:** Primal dual/Nesterov acceleration for pLaplacian learning (Mauricio Flores)

Current/Future work

1. **Fast algorithms:** Primal dual/Nesterov acceleration for pLaplacian learning (Mauricio Flores)

2. **Rates of convergence:**
    1. PageRank algorithm (Amber Yuan)
    2. Nondominated Sorting (Brendan Cook)

Current/Future work

1. **Fast algorithms:** Primal dual/Nesterov acceleration for pLaplacian learning (Mauricio Flores)

2. **Rates of convergence:**
   1. PageRank algorithm (Amber Yuan)
   2. Nondominated Sorting (Brendan Cook)

3. **Infinite labled data:** Suppose the set of labeled data $\mathcal{O}$ grows with n.
   - How fast should $\mathcal{O}$ grow to ensure $p \leq d$ is well-posed?

Current/Future work

1. **Fast algorithms:** Primal dual/Nesterov acceleration for pLaplacian learning (Mauricio Flores)

2. **Rates of convergence:**
   1. PageRank algorithm (Amber Yuan)
   2. Nondominated Sorting (Brendan Cook)

3. **Infinite labled data:** Suppose the set of labeled data $\mathcal{O}$ grows with n.
   - How fast should $\mathcal{O}$ grow to ensure $p \leq d$ is well-posed?
   - What types of models can we take for $\mathcal{O}$?

Current/Future work

1. **Fast algorithms:** Primal dual/Nesterov acceleration for pLaplacian learning (Mauricio Flores)

2. **Rates of convergence:**
   1. PageRank algorithm (Amber Yuan)
   2. Nondominated Sorting (Brendan Cook)

3. **Infinite labled data:** Suppose the set of labeled data $\mathcal{O}$ grows with n.
   - How fast should $\mathcal{O}$ grow to ensure $p \leq d$ is well-posed?
   - What types of models can we take for $\mathcal{O}$?

4. **Soft constraint:** Extend the results to the soft constraint

$$\min_{u:\mathcal{X}_n \to \mathbb{R}} J_p(u) + \lambda \sum_{y \in \mathcal{O}} |u(x) - g(x)|^q.$$

Outline

Adhar, G. (2007).
Parallel algorithms for chains and anti-chains of points on a plane.
In *International Conference on Parallel and Distributed Systems (ICPADS)*, volume 2, pages 1–7.

Alamgir, M. and Luxburg, U. V. (2011).
Phase transition in the family of p-resistances.
In *Advances in Neural Information Processing Systems*, pages 379–387.

Aldous, D. and Diaconis, P. (1999).
Longest increasing subsequences: from patience sorting to the Baik-Deift-Johansson Theorem.
Bulletin of the American Mathematical Society, 36(4):413–432.

Amghibech, S. (2003).
Eigenvalues of the discrete p-Laplacian for graphs.
Ars Combinatoria, 67:283–302.

Ando, R. K. and Zhang, T. (2007).
Learning on graph with laplacian regularization.
Advances in Neural Information Processing Systems, 19:25.

Aronsson, G., Crandall, M., and Juutinen, P. (2004).
A tour of the theory of absolutely minimizing functions.
Bulletin of the American Mathematical Society, 41(4):439–505.

Barnett, V. (1976).
The ordering of multivariate data.
Journal of the Royal Statistical Society. Series A (General), pages 318–355.

Bridle, N. and Zhu, X. (2013).
p-voltages: Laplacian regularization for semi-supervised learning on high-dimensional data.
In *Eleventh Workshop on Mining and Learning with Graphs (MLG2013)*.

Bühler, T. and Hein, M. (2009).
Spectral clustering based on the graph p-Laplacian.
In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 81–88. ACM.

Calder, J. (2016).
A direct verification argument for the Hamilton–Jacobi equation continuum limit of nondominated sorting.
Nonlinear Analysis: Theory, Methods & Applications, 141:88–108.

Calder, J. (2017a).
Consistency of lipschitz learning with infinite unlabeled data and finite labeled data.
arXiv:1710.10364.

Calder, J. (2017b).
The game theoretic p-Laplacian and semi-supervised learning with few labels.
arXiv:1710.10364.

Chapelle, O., Scholkopf, B., and Zien, A. (2006).
Semi-supervised learning.
MIT.

Deb, K., Pratap, A., Agarwal, S., and Meyarivan, T. (2002).
A fast and elitist multiobjective genetic algorithm: NSGA-II.
IEEE Transactions on Evolutionary Computation, 6(2):182–197.

Donoho, D. L. and Gasko, M. (1992).
Breakdown properties of location estimates based on halfspace depth and projected outlyingness.
The Annals of Statistics, pages 1803–1827.

El Alaoui, A., Cheng, X., Ramdas, A., Wainwright, M. J., and Jordan, M. I. (2016).
Asymptotic behavior of lp-based Laplacian regularization in semi-supervised learning.
In *29th Annual Conference on Learning Theory*, pages 879–906.

Felsner, S. and Wernisch, L. (1999).
Maximum k-chains in planar point sets: Combinatorial structure and algorithms.
SIAM Journal on Computing, 28(1):192–209.

Hammersley, J. (1972).
A few seedlings of research.
In *Proceedings of the Sixth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 345–394.

He, J., Li, M., Zhang, H.-J., Tong, H., and Zhang, C. (2006).
Generalized manifold-ranking-based image retrieval.
IEEE Transactions on image processing, 15(10):3170–3177.

Hero, A. (2003).
Gene selection and ranking with microarray data.
In *IEEE International Symposium on Signal Processing and its Applications*, volume 1, pages 457–464.

Hodge, V. J. and Austin, J. (2004).
A survey of outlier detection methodologies.
Artificial Intelligence Review, 22(2):85–126.

Hsiao, K.-J., Calder, J., and Hero III, A. O. (2015).
Pareto-depth for multiple-query image retrieval.
IEEE Transactions on Image Processing, 24(2):583–594.

Hsiao, K.-J., Xu, K., Calder, J., and Hero, A. (2012).
Multi-criteria anomaly detection using Pareto Depth Analysis.
In *Advances in Neural Information Processing Systems 25*, pages 854–862.

Hsiao, K.-J., Xu, K. S., Calder, J., and Hero, A. O. (2016).
Multicriteria similarity-based anomaly detection using pareto depth analysis.
IEEE transactions on neural networks and learning systems, 27(6):1307–1321.

Kyng, R., Rao, A., Sachdeva, S., and Spielman, D. A. (2015).
Algorithms for lipschitz learning on graphs.
In *Proceedings of The 28th Conference on Learning Theory*, pages 1190–1223.

Lou, R. and Sarrafzadeh, M. (1993).
An optimal algorithm for the maximum three-chain problem.
SIAM Journal on Computing, 22(5):976–993.

Luo, D., Huang, H., Ding, C., and Nie, F. (2010).
On the eigenvectors of p-Laplacian.
Machine Learning, 81(1):37–51.

Luxburg, U. v. and Bousquet, O. (2004).
Distance-based classification with lipschitz functions.
Journal of Machine Learning Research, 5(Jun):669–695.

Manfredi, J. J., Oberman, A. M., and Sviridov, A. P. (2015).
Nonlinear elliptic partial differential equations and p-harmonic functions on graphs.
Differential Integral Equations, 28(1–2):79–102.

Nadler, B., Srebro, N., and Zhou, X. (2009).
Semi-supervised learning with the graph Laplacian: The limit of infinite unlabelled data.
In *Neural Information Processing Systems (NIPS)*.

Papadias, D., Tao, Y., Fu, G., and Seeger, B. (2005).
Progressive skyline computation in database systems.
ACM Transactions on Database Systems (TODS), 30(1):41–82.

Pevzner, P. (2000).
Computational Molecular Biology.
The MIT Press.

Poulos, M., Papavlasopoulos, S., and Chrissikopoulos, V. (2005).
An application of the onion peeling algorithm for fingerprint verification purposes.
Journal of Information and Optimization Sciences, 26(3):665–681.

Prähofer, M. and Spohn, H. (2000).
Universal distributions for growth processes in $1+1$ dimensions and random matrices.
Physical review letters, 84(21):4882–4885.

Rousseeuw, P. and Struyf, A. (2004).
Computation of robust statistics: depth, median, and related measures.
In Goodman, J. and O'Rourke, J., editors, *Handbook of Discrete and Compututational Geometry*, Discrete Mathematics and its Applications, pages 1279–1292. Chapman & Hall-CRC Press.

Slepčev, D. and Thorpe, M. (2017).
Analysis of p-laplacian regularization in semi-supervised learning.
arXiv preprint arXiv:1707.06213.

Suk, T. and Flusser, J. (1999).
Convex layers: a new tool for recognition of projectively deformed point sets.
In *Computer Analysis of Images and Patterns*, pages 454–461. Springer.

Ulam, S. (1961).
Monte carlo calculations in problems of mathematical physics.
Modern Mathematics for the Engineers, pages 261–281.

Viennot, G. (1984).
Chain and antichain families, grids and Young tableaux.
In *Orders: description and roles*, volume 99 of *North-Holland Mathematics Studies*, pages 409–463.

Wang, Y., Cheema, M. A., Lin, X., and Zhang, Q. (2013).
Multi-manifold ranking: Using multiple features for better image retrieval.
In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 449–460. Springer.

Xu, B., Bu, J., Chen, C., Cai, D., He, X., Liu, W., and Luo, J. (2011).
Efficient manifold ranking for image retrieval.
In *Proceedings of the 34th international ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 525–534. ACM.

Yang, C., Zhang, L., Lu, H., Ruan, X., and Yang, M.-H. (2013).
Saliency detection via graph-based manifold ranking.
In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 3166–3173.

Zhou, D., Huang, J., and Schölkopf, B. (2005).
Learning from labeled and unlabeled data on a directed graph.
In *Proceedings of the 22nd International Conference on Machine Learning*, pages 1036–1043. ACM.

Zhou, D. and Schölkopf, B. (2005).
Regularization on discrete spaces.
In *Joint Pattern Recognition Symposium*, pages 361–368. Springer.

Zhou, X., Belkin, M., and Srebro, N. (2011).
An iterated graph laplacian approach for ranking on manifolds.
In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 877–885. ACM.

Zhu, X., Ghahramani, Z., Lafferty, J., et al. (2003).
Semi-supervised learning using gaussian fields and harmonic functions.
In *International Conference on Machine Learning*, volume 3, pages 912–919.