

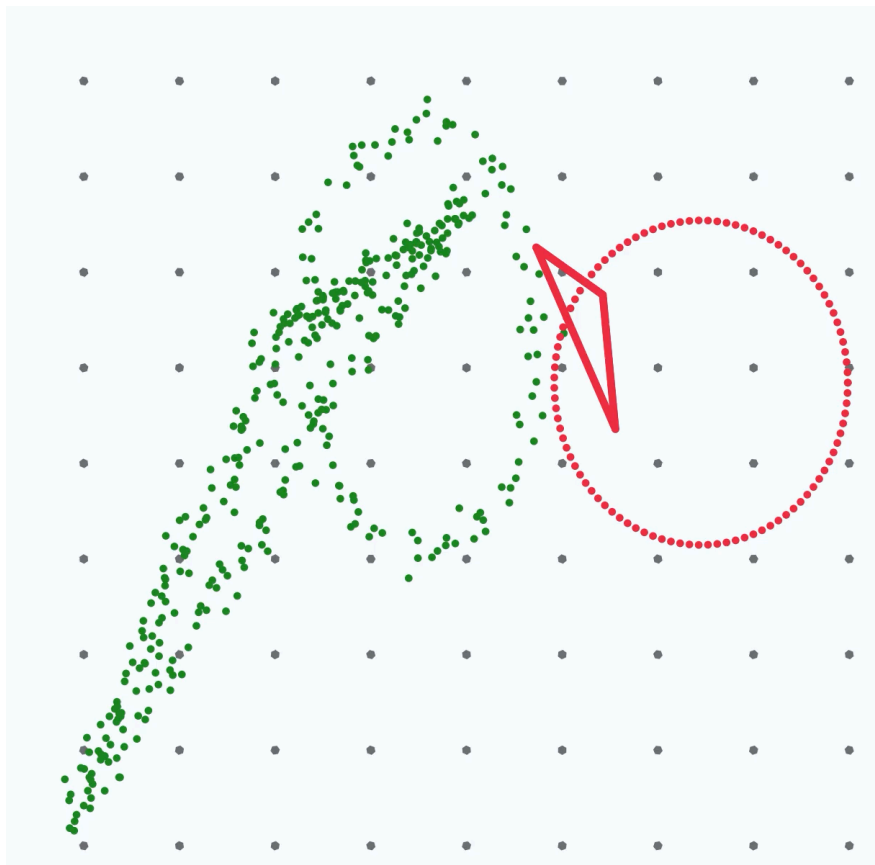
A contour plot of a 2D energy landscape. The plot features several nested, irregular contour lines representing energy levels. The contours are labeled with numerical values: 3.0, 4.5, 6.0, 7.5, and 9.0. The center of the plot is the darkest, indicating the lowest energy state. A path of small, light-colored dots starts from the center and moves outwards, following the contours. The background is a gradient of dark blue to green.

# Neural Estimation of Energy Mover's Distance

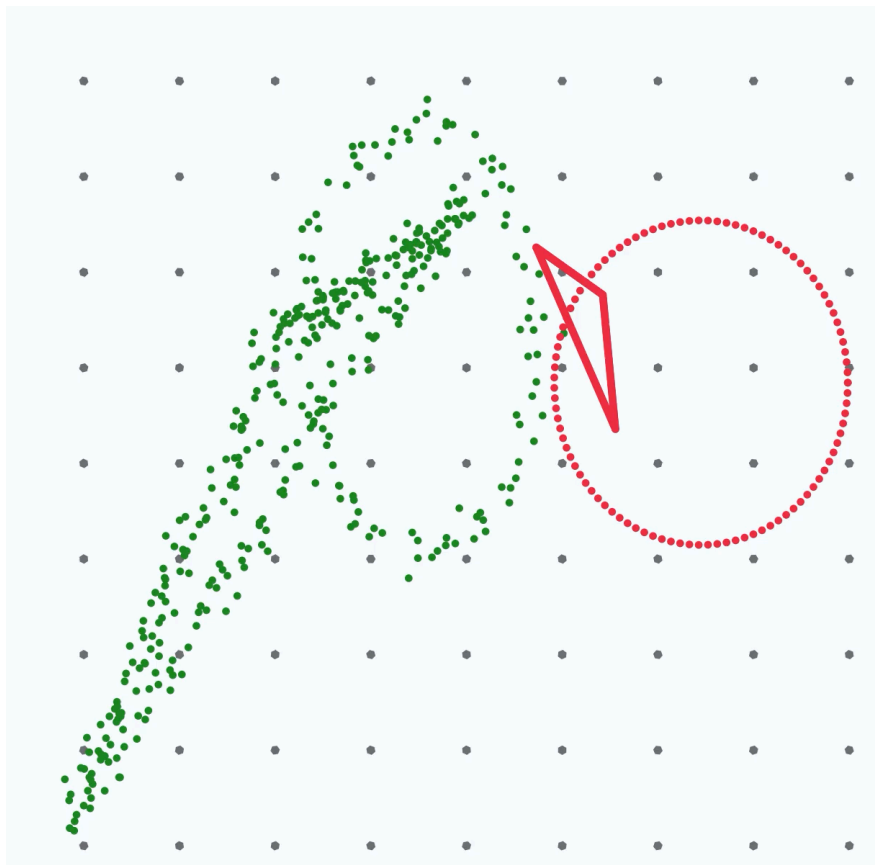
NEEMo

Ouail Kitouni

# Fitting Arbitrary Geometries



# Fitting Arbitrary Geometries



---

# Robust and Provably Monotonic Networks

---

**Ouail Kitouni\*, Niklas Nolte\*, Mike Williams**

NSF AI Institute for Artificial Intelligence and Fundamental Interactions  
Laboratory for Nuclear Science, MIT



Niklas Nolte



Ouail Kitouni

---

# Finding NEEMo: Geometric Fitting using Neural Estimation of the Energy Mover's Distance

---

**Ouail Kitouni, Niklas Nolte, Mike Williams**

The NSF AI Institute for Artificial Intelligence and Fundamental Interactions  
Massachusetts Institute of Technology  
Cambridge, MA 02139, USA  
{kitouni, nnolte, mwill}@mit.edu



Mike Williams

---

# Robust and Provably Monotonic Networks

---

**Ouail Kitouni\*, Niklas Nolte\*, Mike Williams**

NSF AI Institute for Artificial Intelligence and Fundamental Interactions  
Laboratory for Nuclear Science, MIT



Niklas Nolte



Ouail Kitouni

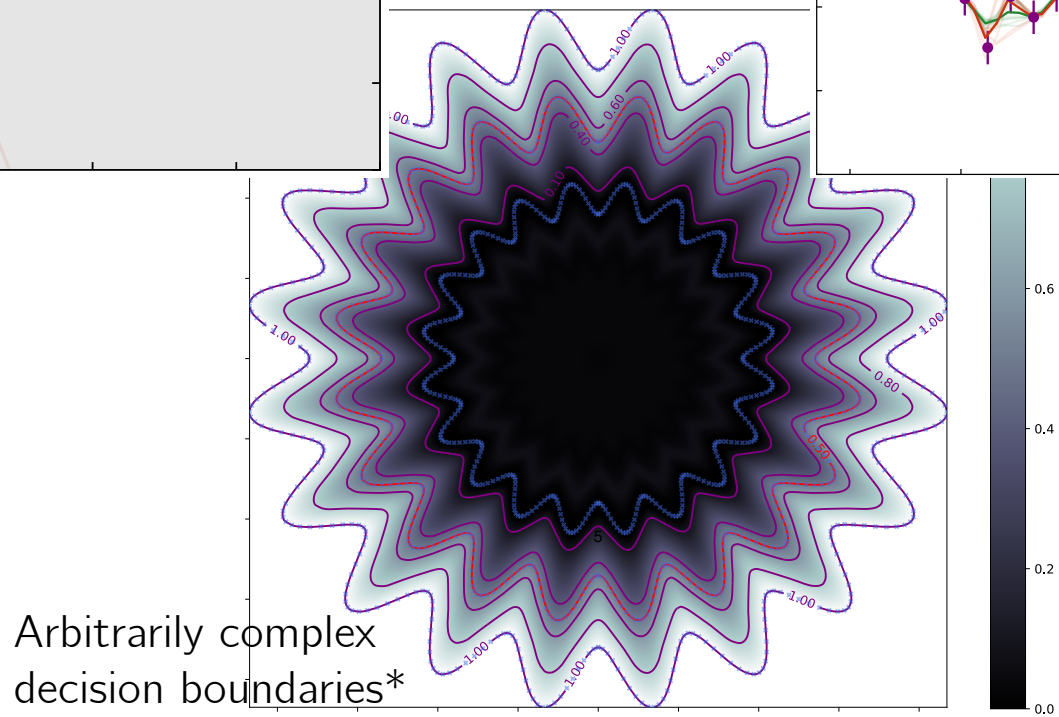
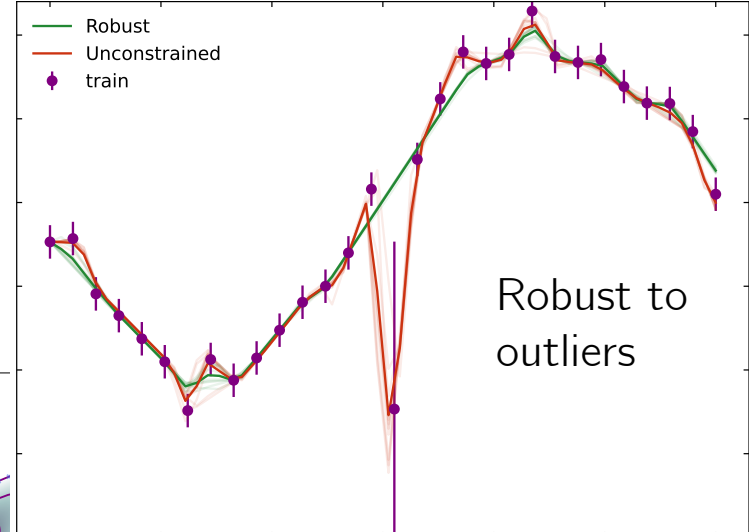
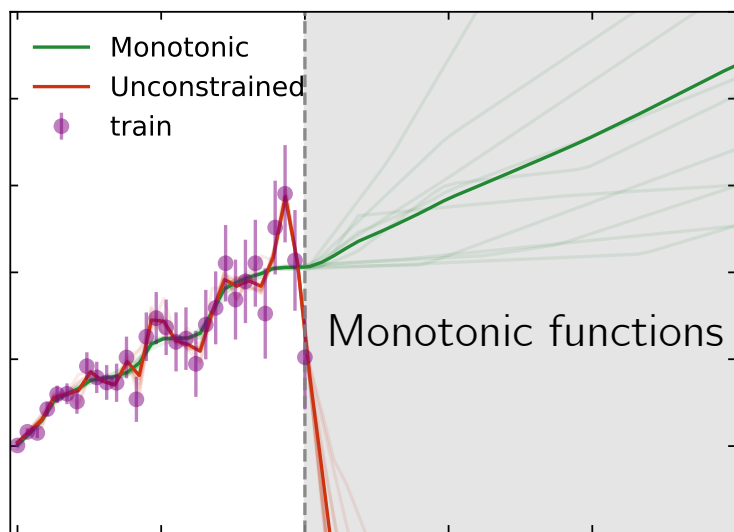


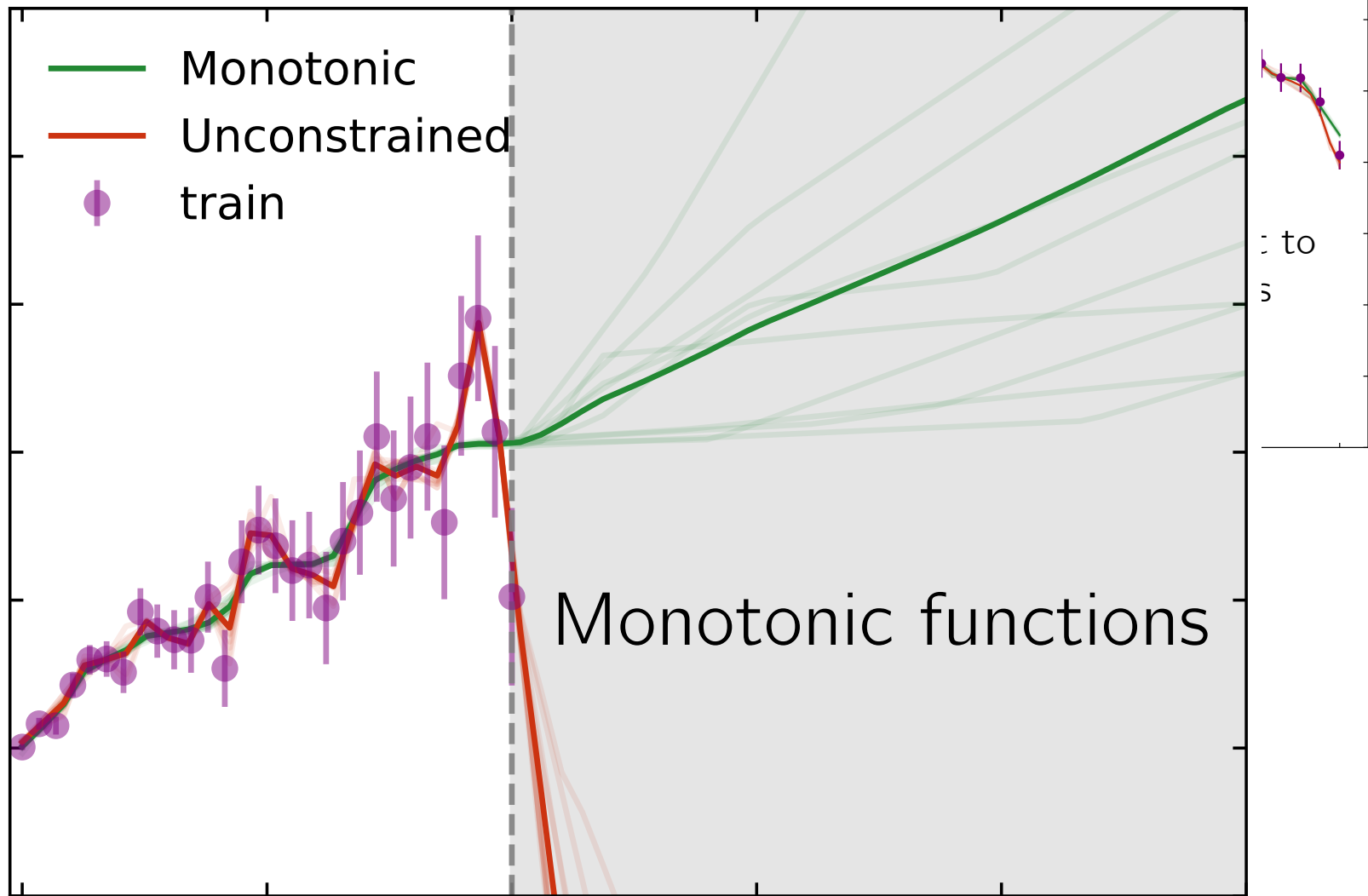
Mike Williams

# **Part 1**

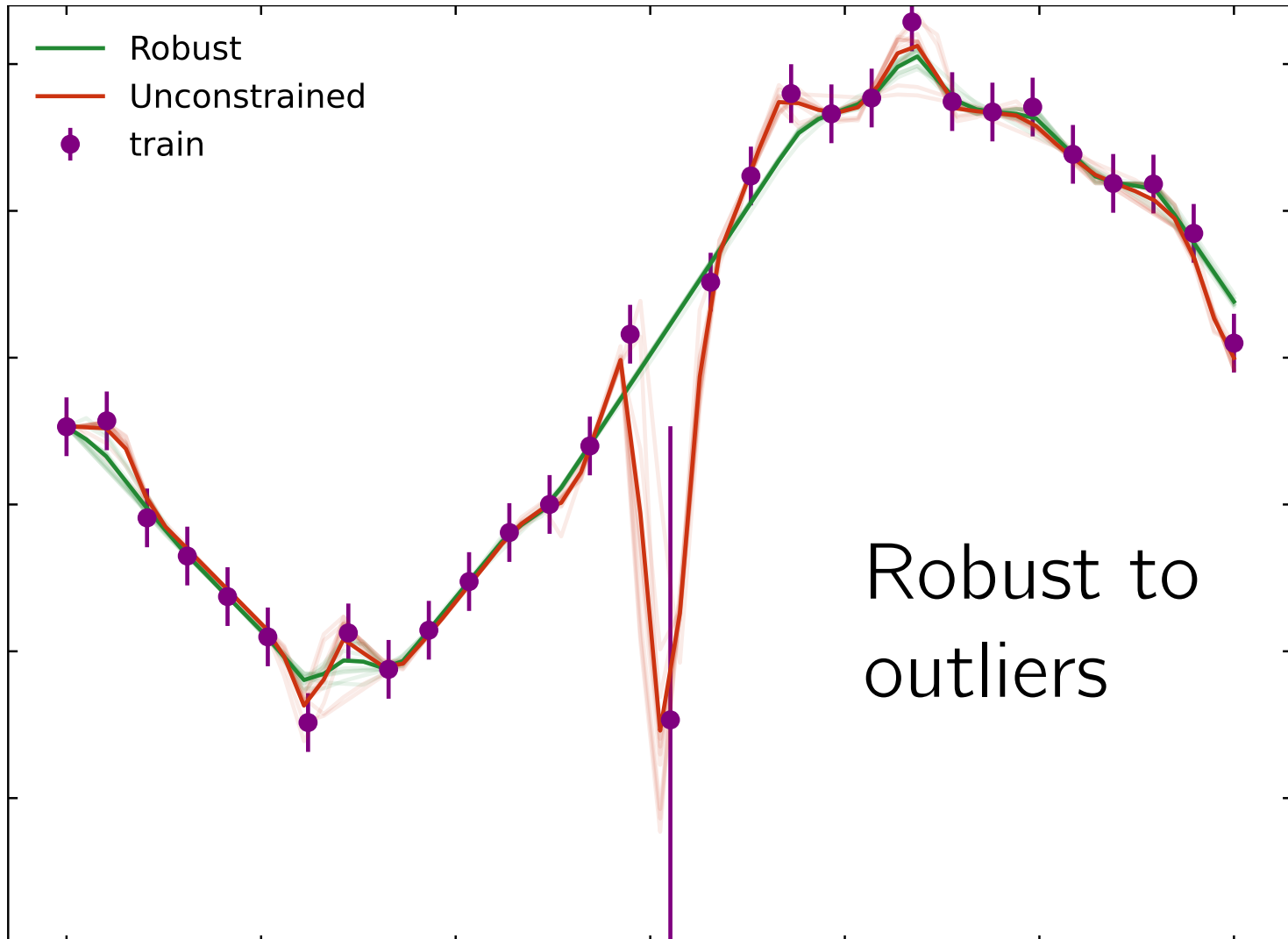
# **Lipschitz**

# **Networks**



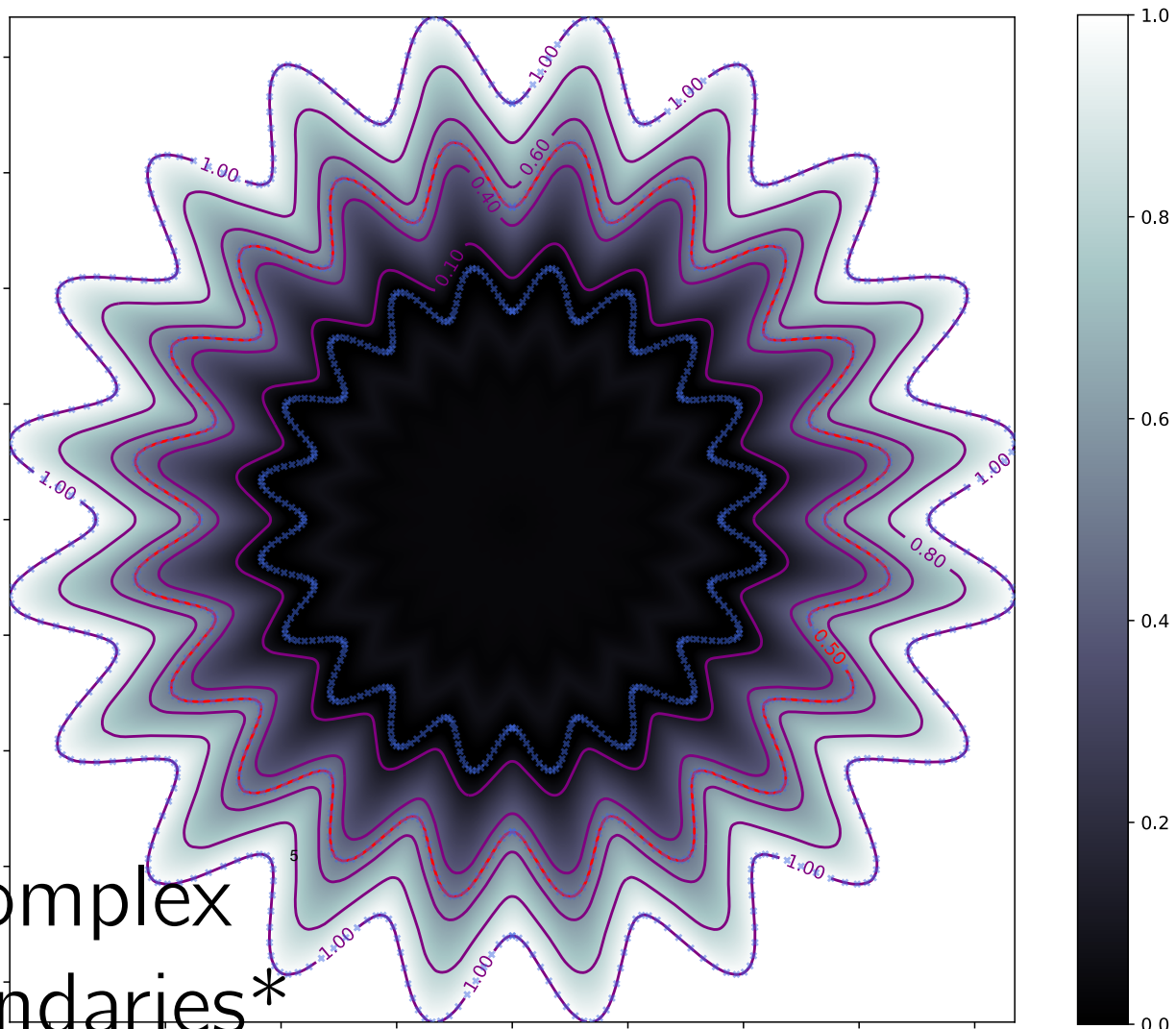






— Robust  
— Unconstrained  
● train

Robust to  
outliers



Arbitrarily complex  
decision boundaries\*

# MontoneNorm

niklasnlte / MonotOneNorm Public

Watch 1 Fork 0

<> Code Issues Pull requests Actions Projects Wiki Security Insights

main 3 branches 0 tags Go to file Add file Code

niklasnlte	fix typo	3ac36a2 19 days ago	🕒 27 commits
Examples	fix typo	19 days ago	
monotonenorm	Added examples and documentation	20 days ago	
.gitignore	initial commit	9 months ago	
README.md	fix typo	19 days ago	
setup.py	renamed project	4 months ago	

About

No description, web provided.

Readme

1 star

1 watching

0 forks

Releases

<https://github.com/niklasnlte/MonotOneNorm>

```
pip install monotonenorm
```

```
conda install monotonenorm -c okitouni
```

How does it work?

## Robustness: Definition (more formally)

Small changes in the input should not lead to large changes in the output:

$$|f(x + \epsilon) - f(x)| \leq \lambda \epsilon \quad \forall \epsilon > 0$$

Thus we would like our Neural Network to represent a Lipschitz continuous function.

# Robustness: Definition (more formally)

Robustness is achieved by constraining the operator 1-norm of the weight matrices of each layer such that

$$\prod_{l=0}^L \|W^l\|_1 \leq \lambda$$

where  $\lambda$  is Lipschitz constant of the resulting network with respect to the  $\infty$ -norm.

Universal Lipschitz- $\lambda$  function approximation requires activations with gradient 1 almost everywhere.

→ **GroupSort\***: reorders inputs

\*Sorting out Lipschitz function approximation [<https://arxiv.org/abs/1811.05381>]

# Monotonicity using weight norm

$g(\mathbf{x})$  is a  $\lambda$ -Lipschitz neural network. Adding the following residual connection

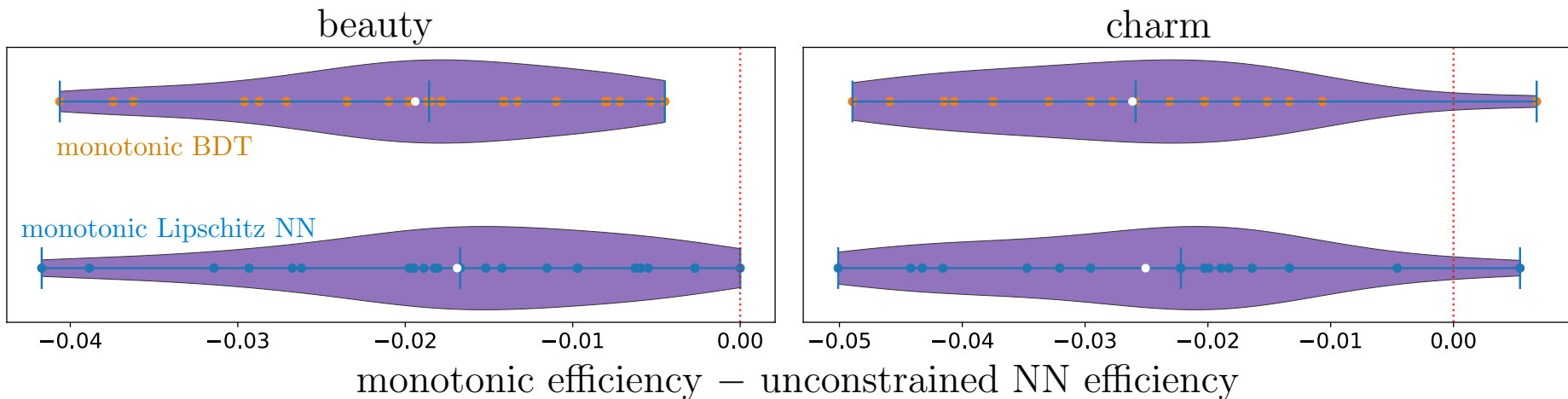
$$f(\mathbf{x}) = g(\mathbf{x}) + \lambda \sum_{i \in I} x_i$$

makes output monotonic since

$$\frac{\partial f}{\partial x_i} = \frac{\partial g}{\partial x_i} g(\mathbf{x}) + \lambda \geq 0 \quad \forall i \in I$$

# Monotonic Lipschitz Networks LHCb RUN 3 trigger

This architecture is being used in the LHCb heavy-flavor RUN 3 trigger.





# **Part 2**

## **Neural Estimation of Energy Mover's distance**

---

## Robust and Provably Monotonic Networks

---

**Ouail Kitouni\*, Niklas Nolte\*, Mike Williams**

NSF AI Institute for Artificial Intelligence and Fundamental Interactions  
Laboratory for Nuclear Science, MIT



Niklas Nolte



Ouail Kitouni

---

## Finding NEEMo: Geometric Fitting using Neural Estimation of the Energy Mover's Distance

---

**Ouail Kitouni, Niklas Nolte, Mike Williams**

The NSF AI Institute for Artificial Intelligence and Fundamental Interactions  
Massachusetts Institute of Technology  
Cambridge, MA 02139, USA  
{kitouni, nnolte, mwill}@mit.edu



Mike Williams

---

# Finding NEEMo: Geometric Fitting using Neural Estimation of the Energy Mover's Distance

---

**Ouail Kitouni, Niklas Nolte, Mike Williams**

The NSF AI Institute for Artificial Intelligence and Fundamental Interactions  
Massachusetts Institute of Technology  
Cambridge, MA 02139, USA  
{kitouni, nnolte, mwill}@mit.edu



Niklas Nolte

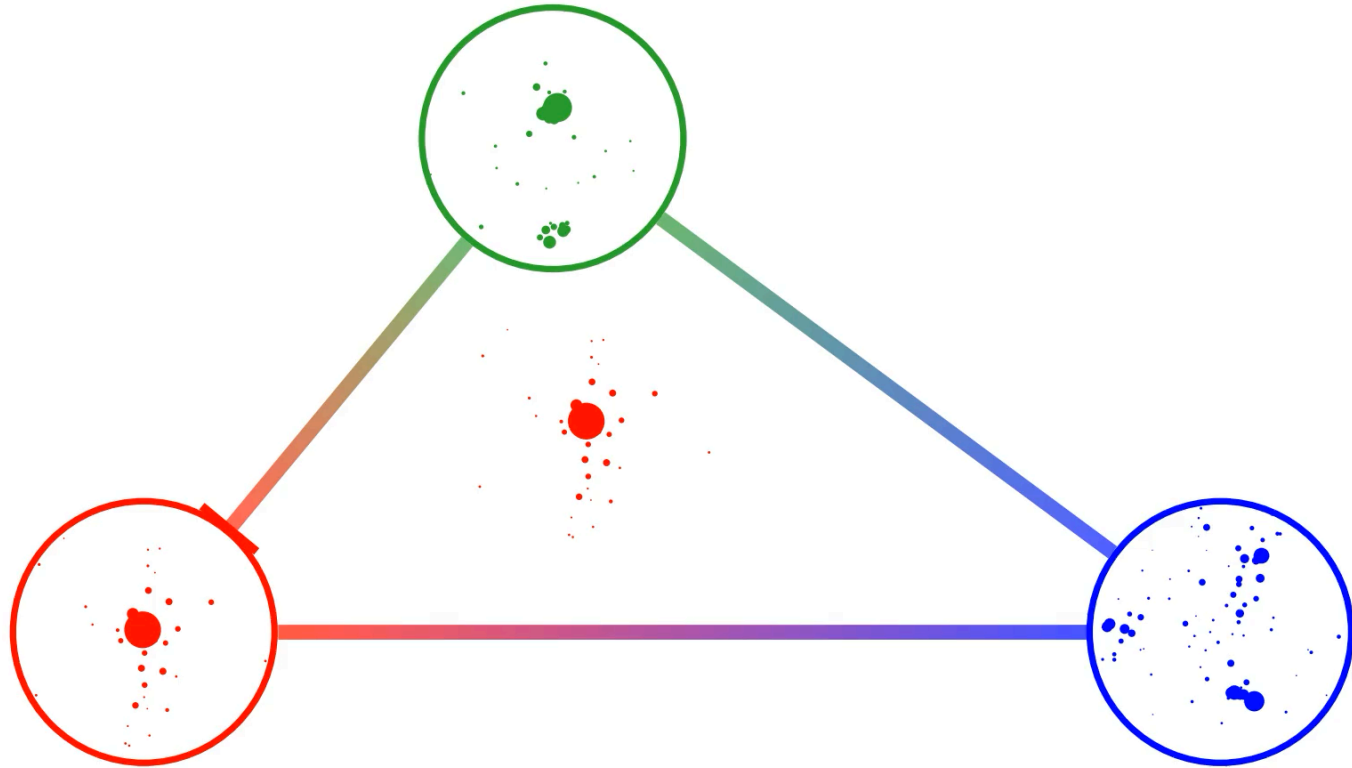


Ouail Kitouni

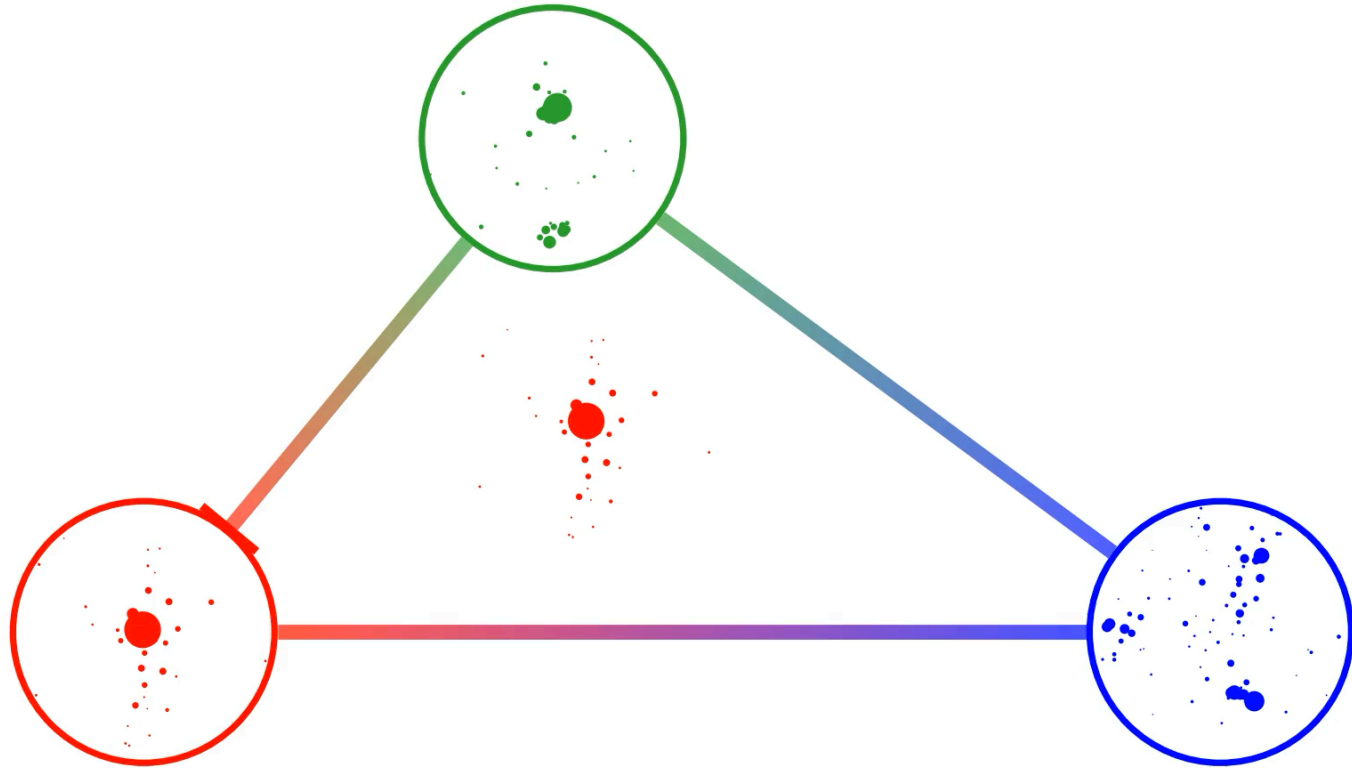


Mike Williams

# Optimal Transport - Energy Mover's Distance



# Optimal Transport - Energy Mover's Distance



# Earth Mover's Distance


The primal formulation of the EMD is an optimization over joint probability distributions

$$\text{EMD}(\mathbb{P}, \mathbb{Q}) = \inf_{\gamma \in \Pi(\mathbb{P}, \mathbb{Q})} \mathbb{E}_{(x,y) \sim \gamma} [\|x - y\|_2],$$

# Earth Mover's Distance

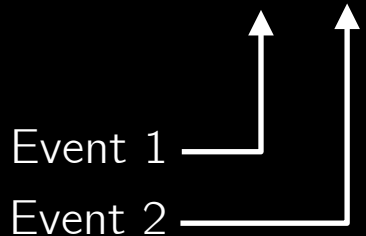
The primal formulation of the EMD is an optimization over joint probability distributions

$$\text{EMD}(\mathbb{P}, \mathbb{Q}) = \inf_{\gamma \in \Pi(\mathbb{P}, \mathbb{Q})} \mathbb{E}_{(x,y) \sim \gamma} [\|x - y\|_2],$$

Event 1 

# Earth Mover's Distance

The primal formulation of the EMD is an optimization over joint probability distributions

$$\text{EMD}(\mathbb{P}, \mathbb{Q}) = \inf_{\gamma \in \Pi(\mathbb{P}, \mathbb{Q})} \mathbb{E}_{(x,y) \sim \gamma} [\|x - y\|_2],$$


The diagram shows two labels, "Event 1" and "Event 2", positioned to the left of the equation. From "Event 1", a horizontal line extends to the right, then a vertical arrow points upwards to the bottom of the "inf" symbol. From "Event 2", a horizontal line extends to the right, then a vertical arrow points upwards to the bottom of the "Q" symbol in the denominator of the fraction.



# Earth Mover's Distance

The primal formulation of the EMD is an optimization over joint probability distributions

$$\text{EMD}(\mathbb{P}, \mathbb{Q}) = \inf_{\gamma \in \Pi(\mathbb{P}, \mathbb{Q})} \mathbb{E}_{(x,y) \sim \gamma} [\|x - y\|_2],$$

Event 1

Event 2

Energies

# Earth Mover's Distance

The primal formulation of the EMD is an optimization over joint probability distributions

$$\text{EMD}(\mathbb{P}, \mathbb{Q}) = \inf_{\gamma \in \Pi(\mathbb{P}, \mathbb{Q})} \mathbb{E}_{(x,y) \sim \gamma} [\|x - y\|_2],$$

The diagram illustrates the components of the EMD formula. On the left, 'Event 1' and 'Event 2' are listed. Arrows from 'Event 1' and 'Event 2' point to the probability distributions  $\mathbb{P}$  and  $\mathbb{Q}$  in the formula. On the right, 'Energies' and 'Spatial Coordinates' are listed. An arrow from 'Energies' points to the joint distribution  $\gamma$  in the formula. An arrow from 'Spatial Coordinates' points to the L2 norm  $\|x - y\|_2$  in the formula.

# Kantorovich-Rubinstein - Dual Formulation

The dual formulation is an optimization over 1-Lipschitz continuous functions

$$\text{EMD}(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim \mathbb{P}}[f(x)] - \mathbb{E}_{x \sim \mathbb{Q}}[f(x)],$$

# Kantorovich-Rubinstein - Dual Formulation

The dual formulation is an optimization over 1-Lipschitz continuous functions

$$\text{EMD}(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_L \leq 1} \mathbb{E}_{x \sim \mathbb{P}}[f(x)] - \mathbb{E}_{x \sim \mathbb{Q}}[f(x)],$$

Kantorovich potential 

# NEEMo Algorithm

Parametrized shape:

$\theta$

Target Distribution

$$\mathbb{Q} = \{e^i, \mathbf{x}^i\}_{i=1}^n$$

Forward pass  $\longrightarrow$

Backward pass  $\cdots\cdots\longrightarrow$

# NEEMo Algorithm

Parametrized shape:

$\theta$



Parametrized Distribution

$$\mathbb{P} = \{w_{\theta}^i, \mathbf{y}_{\theta}^i\}_{i=1}^m$$

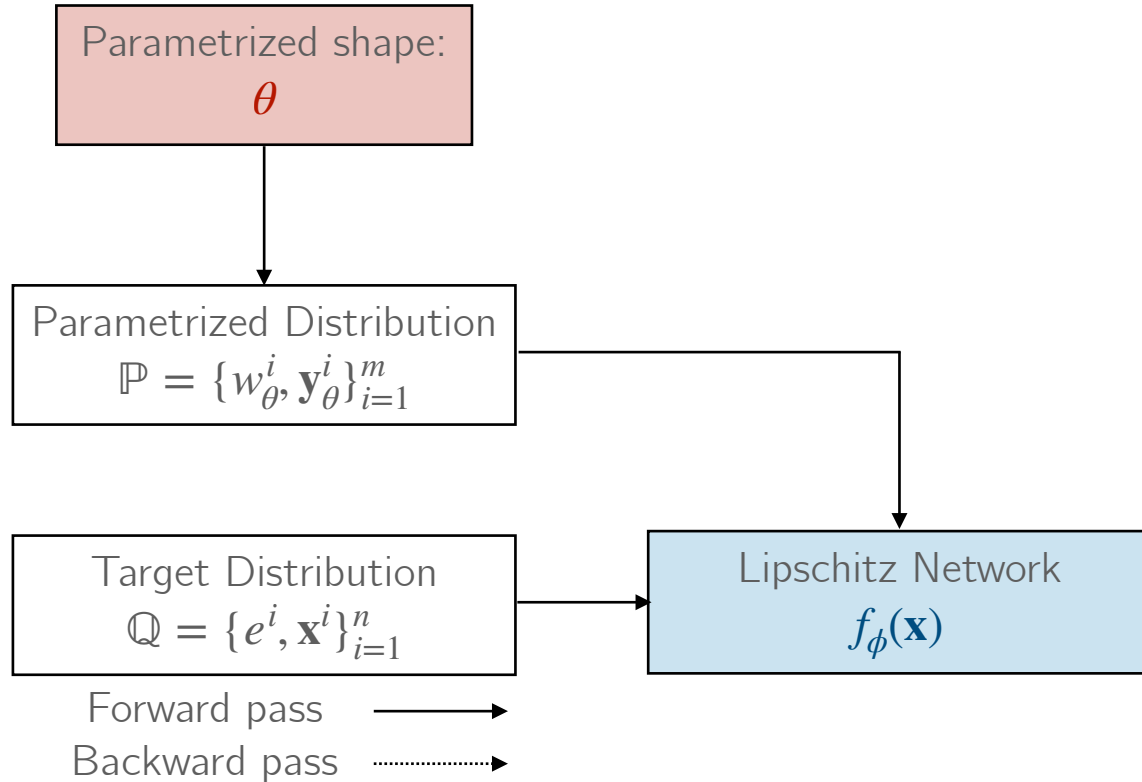
Target Distribution

$$\mathbb{Q} = \{e^i, \mathbf{x}^i\}_{i=1}^n$$

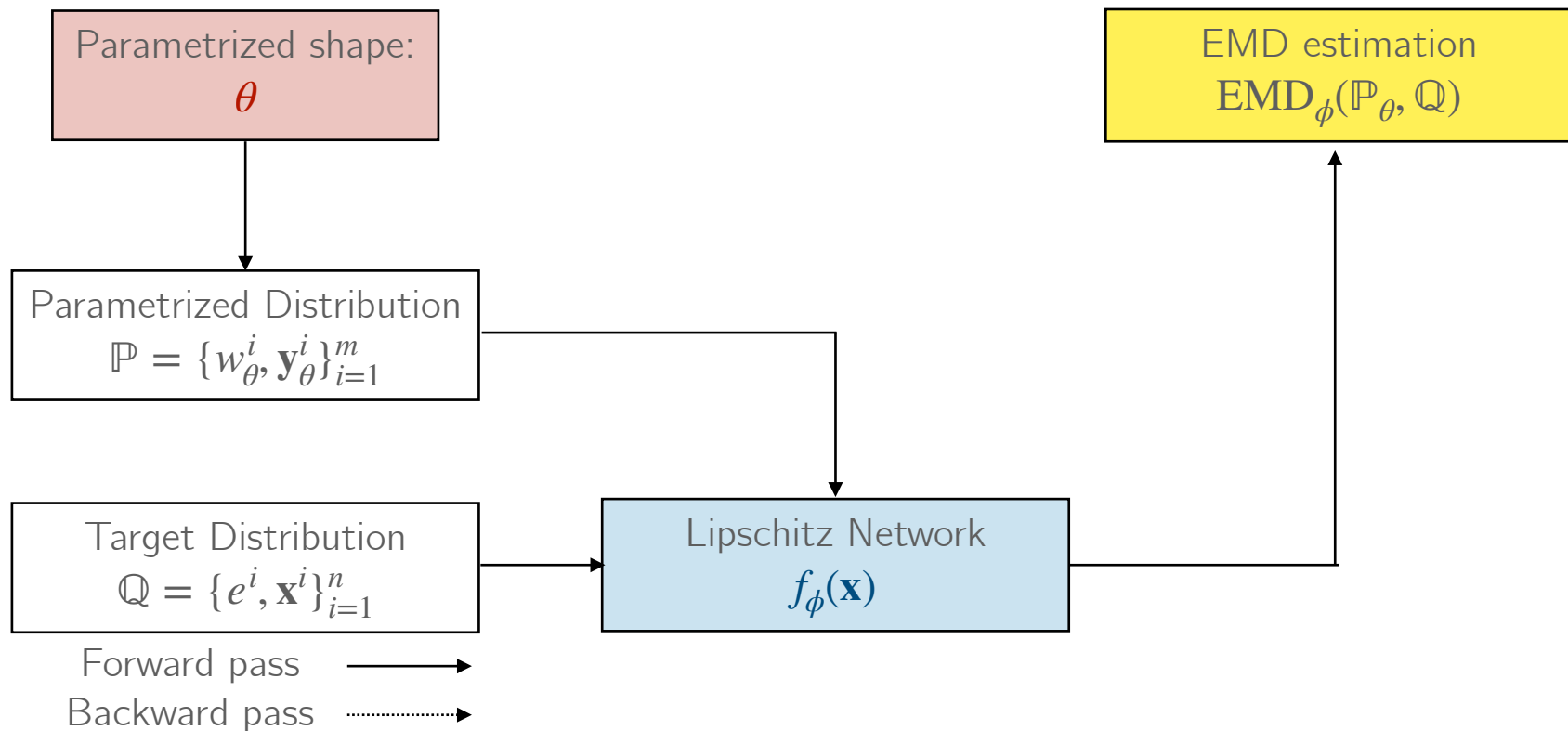
Forward pass  $\longrightarrow$

Backward pass  $\cdots\cdots\longrightarrow$

# NEEMo Algorithm

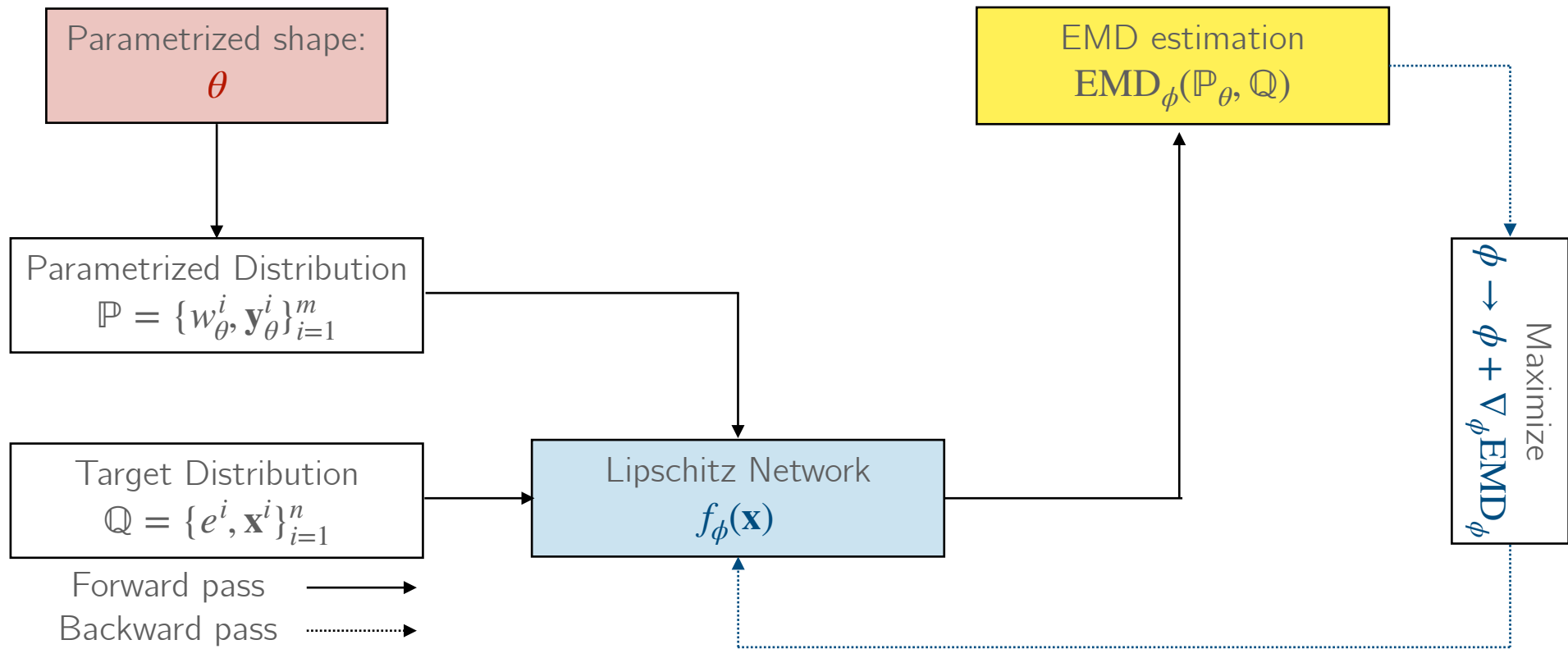


# NEEMo Algorithm

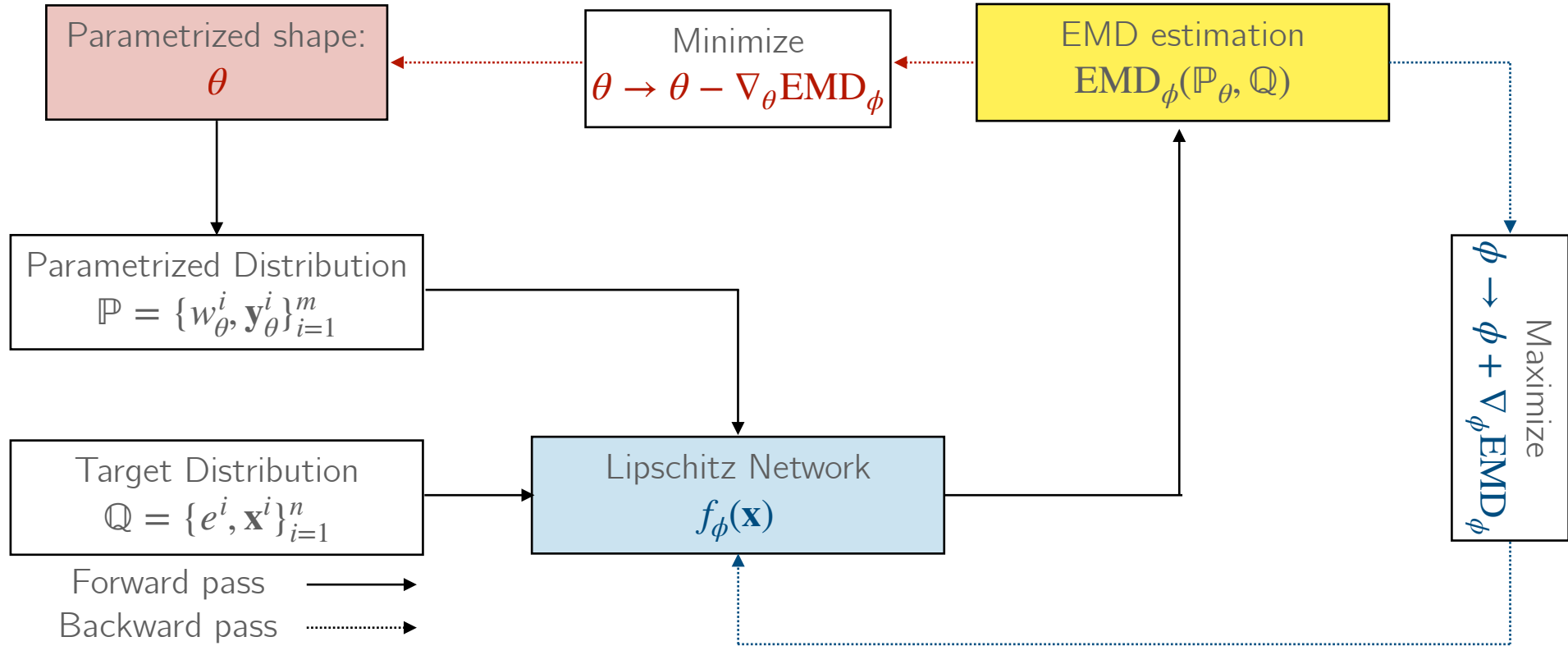




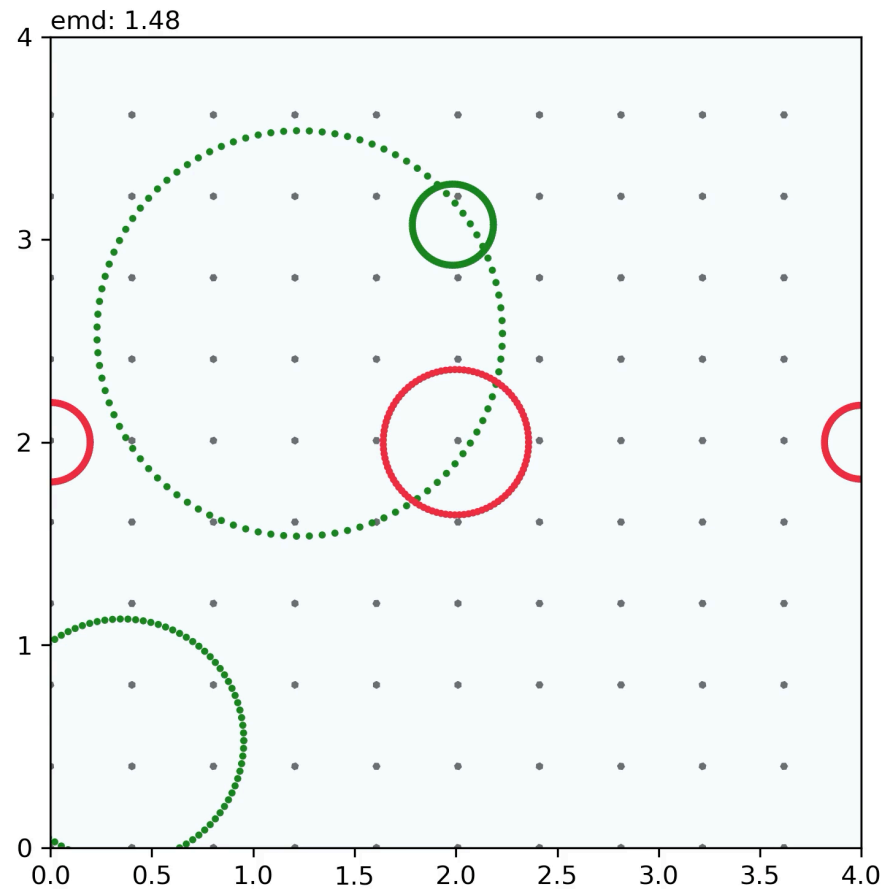
# NEEMo Algorithm



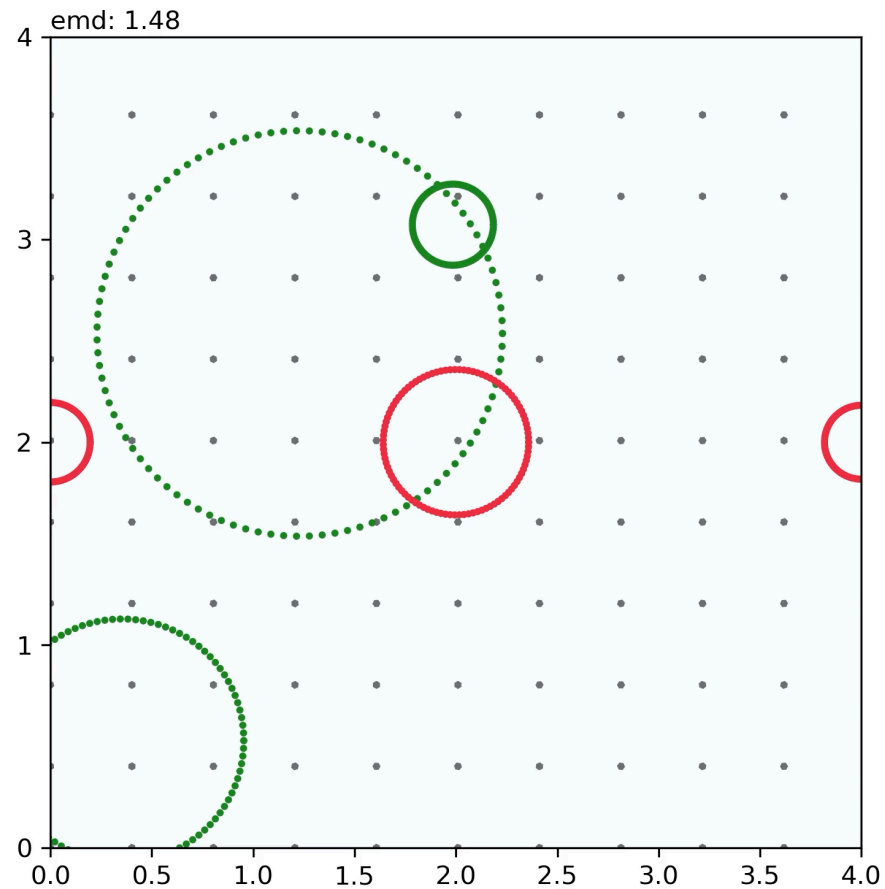
# NEEMo Algorithm



# Fitting Arbitrary Geometries

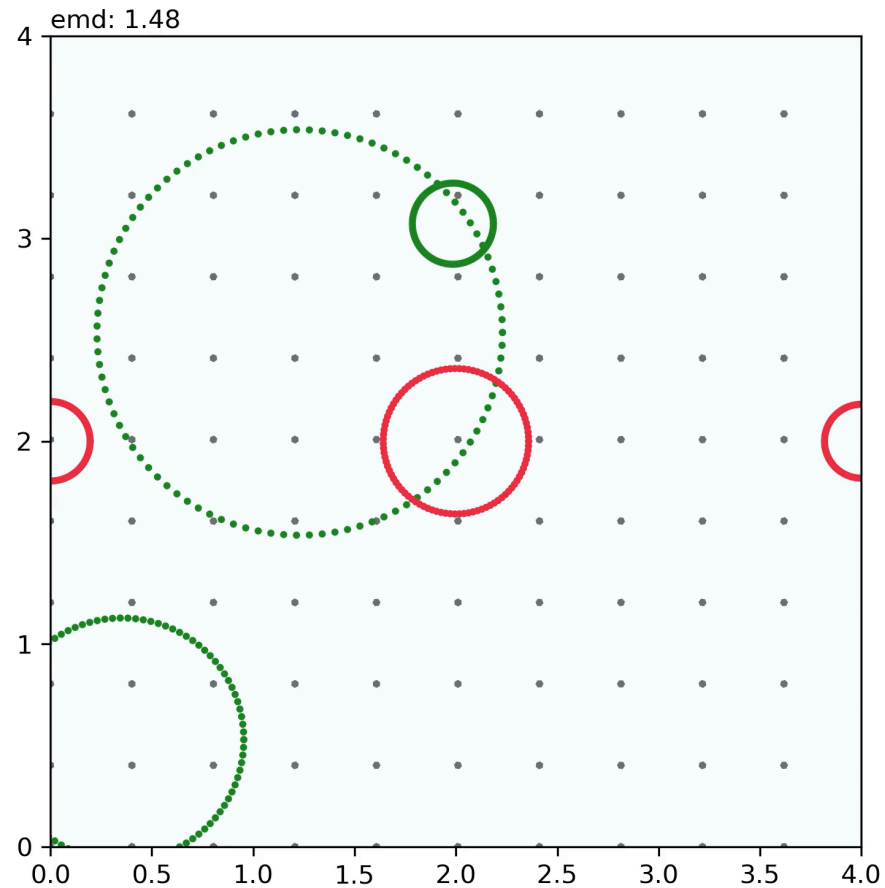


# Fitting Arbitrary Geometries



# Fitting Arbitrary Geometries

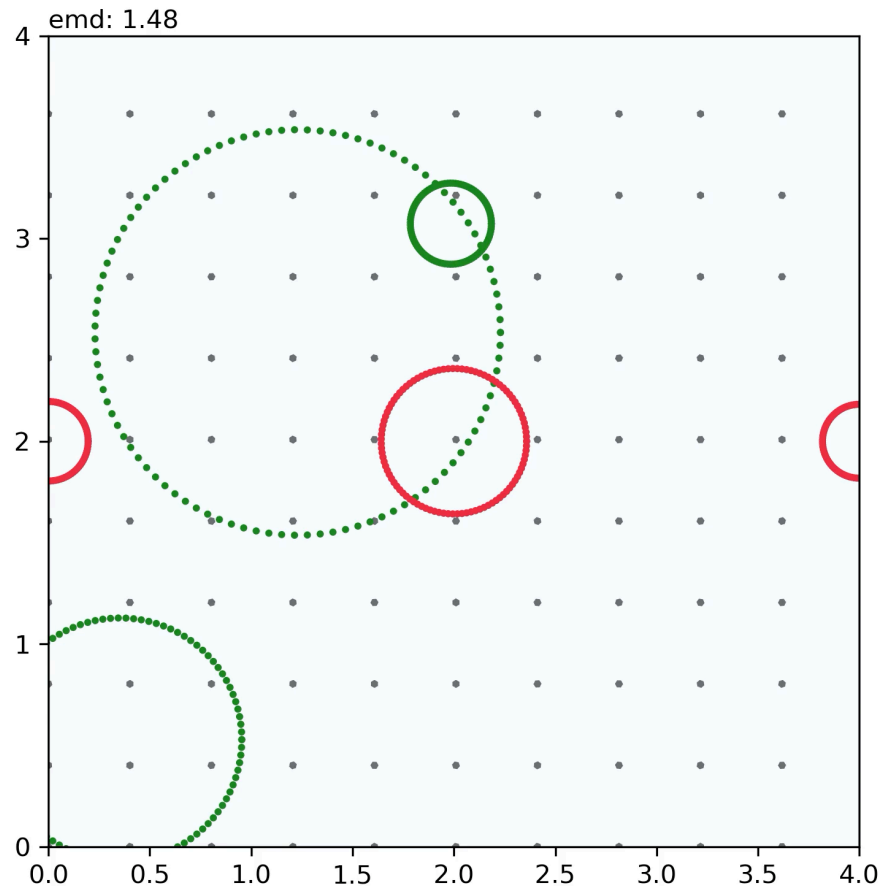
Possible use cases:



# Fitting Arbitrary Geometries

Possible use cases:

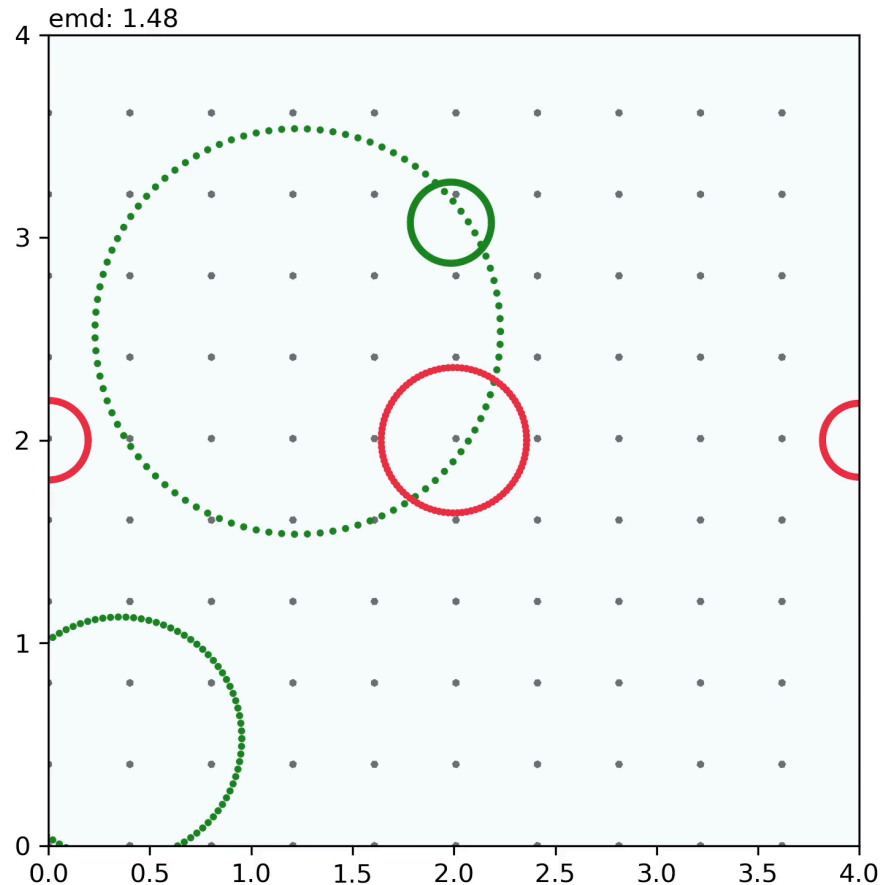
- Floating term for a uniform background to mitigate pileup



# Fitting Arbitrary Geometries

Possible use cases:

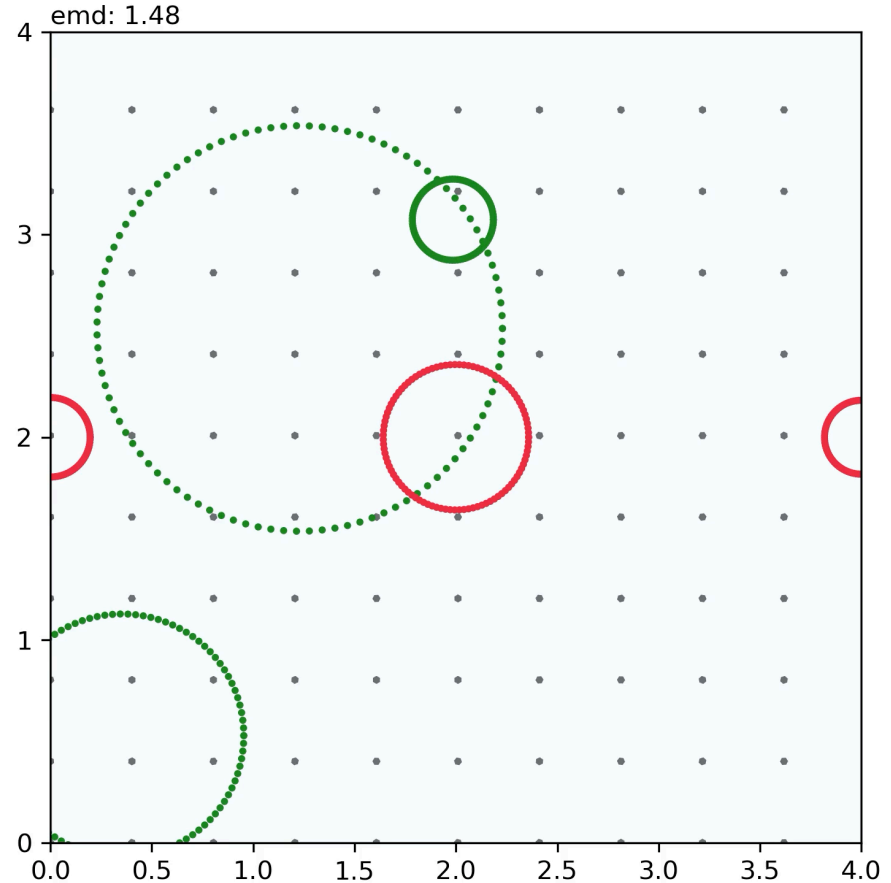
- Floating term for a uniform background to mitigate pileup
- Clustering with jet energy estimation



# Fitting Arbitrary Geometries

Possible use cases:

- Floating term for a uniform background to mitigate pileup
- Clustering with jet energy estimation
- Variable shape clusters (learned radii, ellipses, arbitrary shapes, etc.)

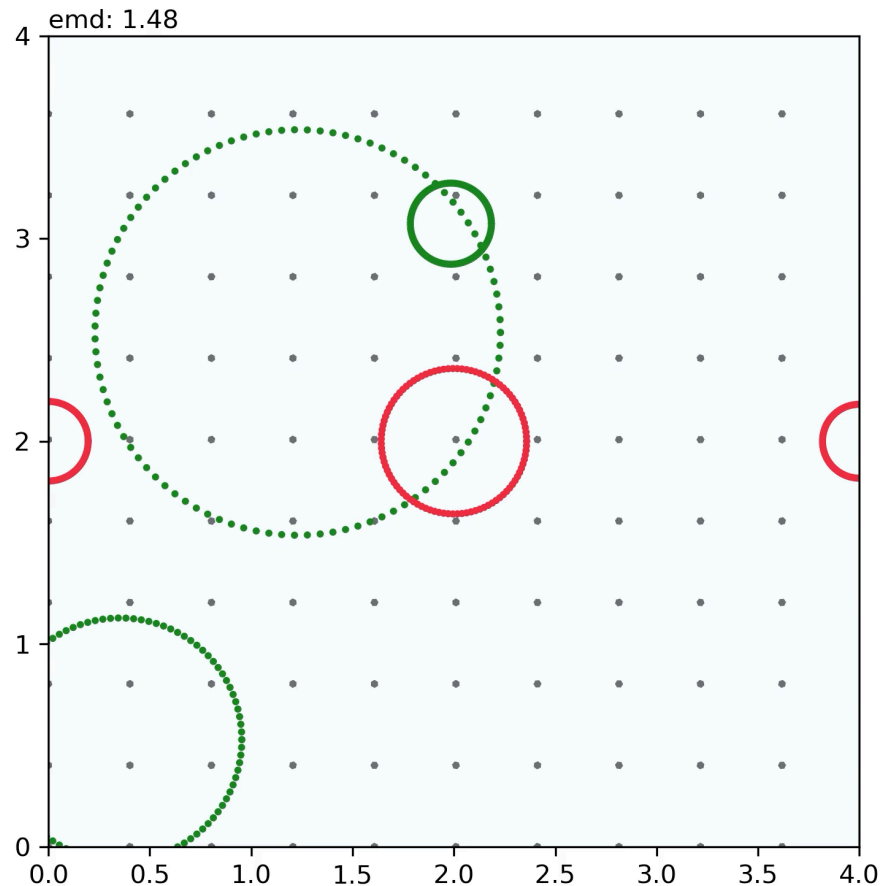




# Fitting Arbitrary Geometries

Possible use cases:

- Floating term for a uniform background to mitigate pileup
- Clustering with jet energy estimation
- Variable shape clusters (learned radii, ellipses, arbitrary shapes, etc.)
- New (and old) observables: N-subjet, N-circle, triangularness, etc.

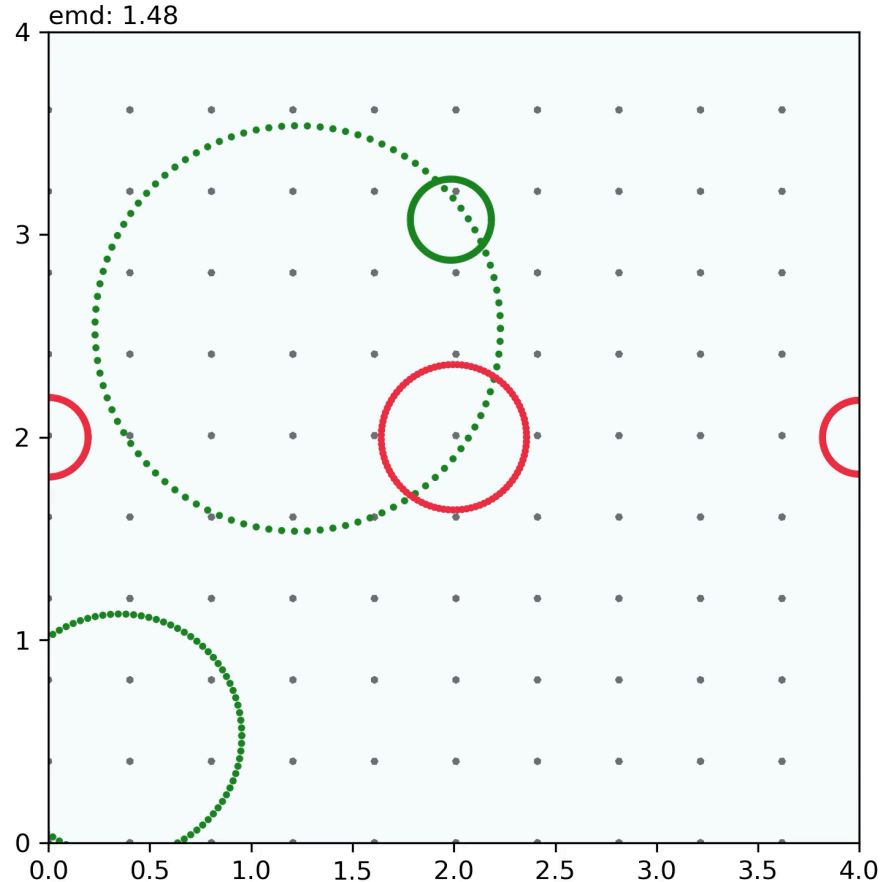


# Fitting Arbitrary Geometries

Possible use cases:

- Floating term for a uniform background to mitigate pileup
- Clustering with jet energy estimation
- Variable shape clusters (learned radii, ellipses, arbitrary shapes, etc.)
- New (and old) observables: N-subjet, N-circle, triangularness, etc.

All in a unified framework given by the Energy Mover's Distance\*



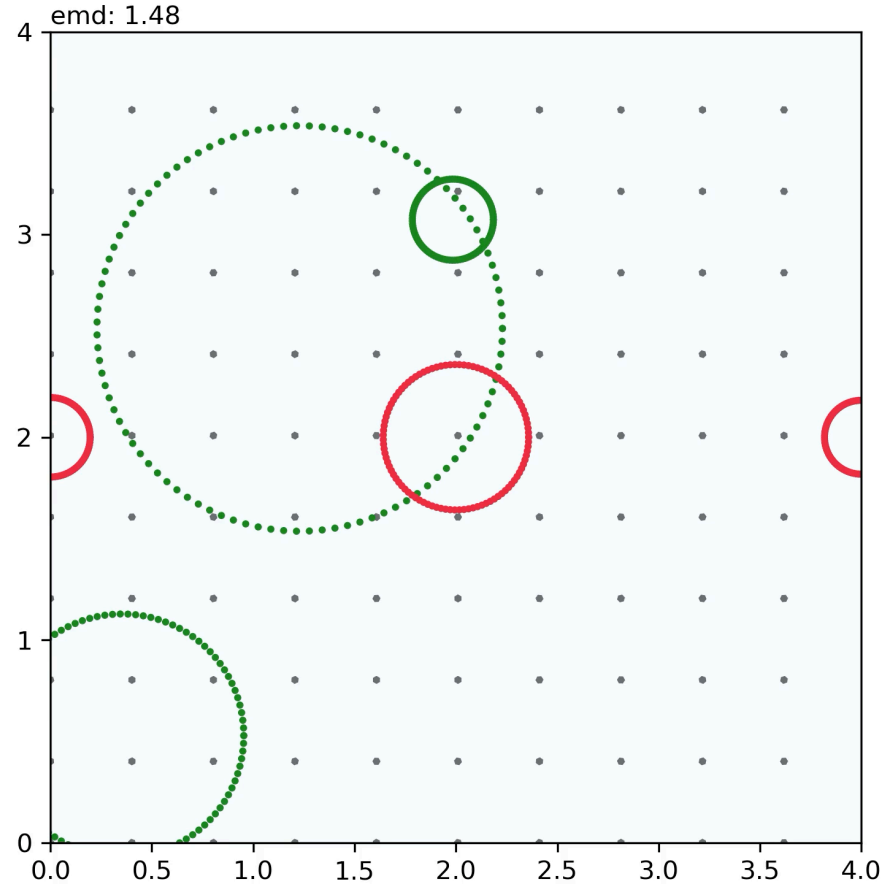
# Fitting Arbitrary Geometries

Possible use cases:

- Floating term for a uniform background to mitigate pileup
- Clustering with jet energy estimation
- Variable shape clusters (learned radii, ellipses, arbitrary shapes, etc.)
- New (and old) observables: N-subjet, N-circle, triangularness, etc.

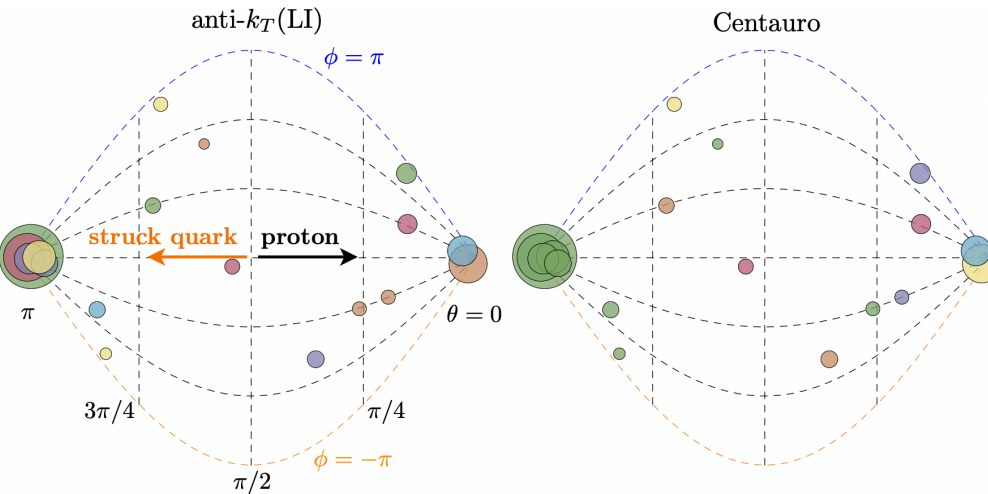
All in a unified framework given by the Energy Mover's Distance\*

\*Can You Hear the Shape of a Jet [<https://indi.to/rbQ5j>]



# EIC - Applications

Hand-designed algorithms

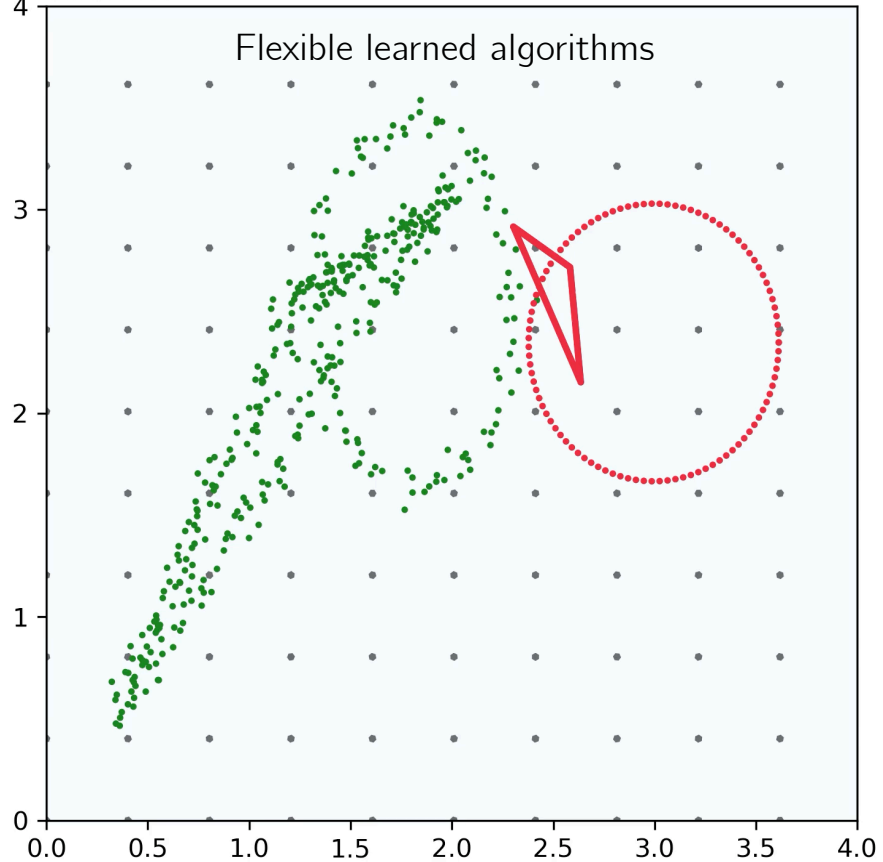


Asymmetric jet clustering in deep-inelastic scattering

[[arxiv.org/pdf/2006.10751.pdf](https://arxiv.org/pdf/2006.10751.pdf)]

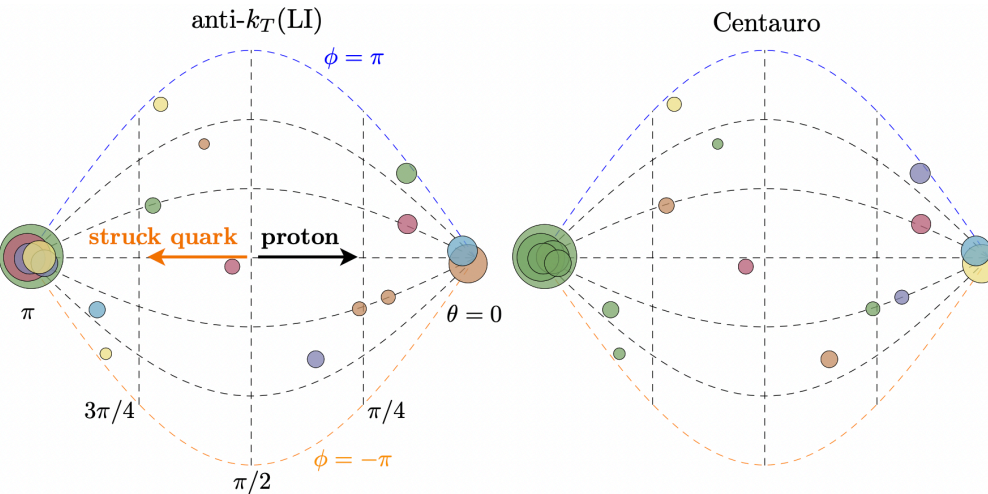
emd: 1.37

Flexible learned algorithms



# EIC - Applications

Hand-designed algorithms

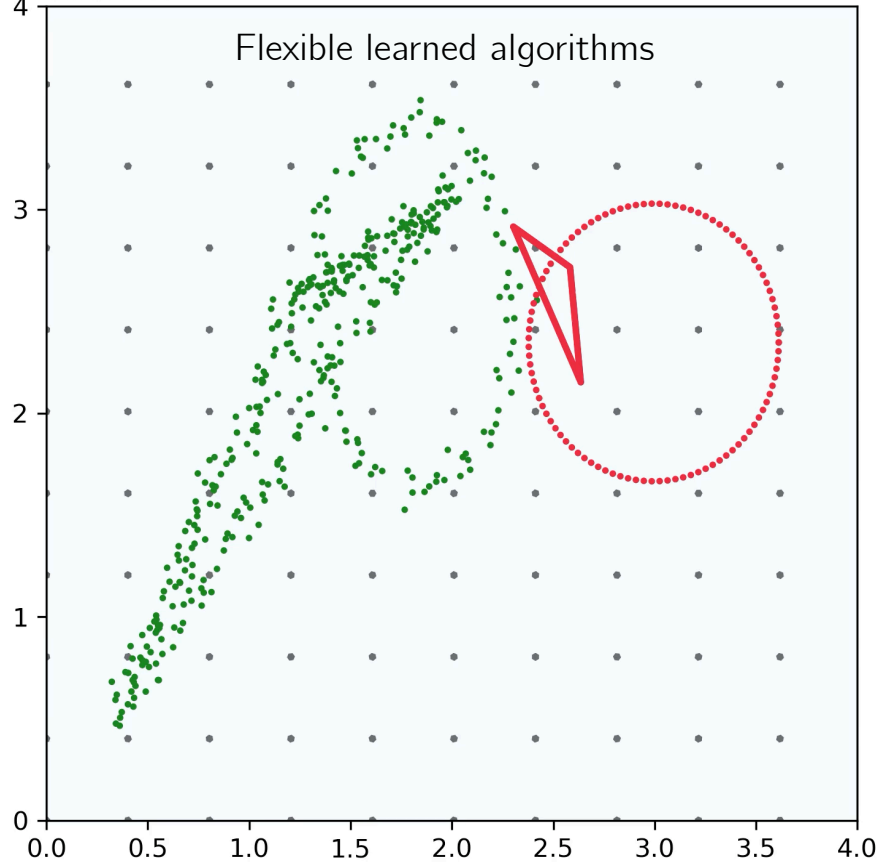


Asymmetric jet clustering in deep-inelastic scattering

[[arxiv.org/pdf/2006.10751.pdf](https://arxiv.org/pdf/2006.10751.pdf)]

emd: 1.37

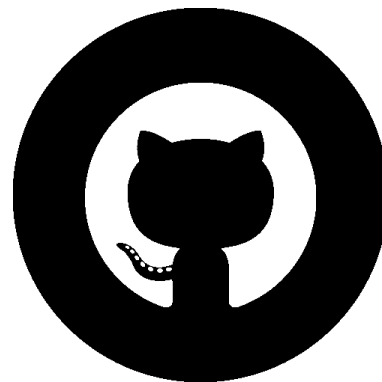
Flexible learned algorithms



# arXiv

<https://arxiv.org/abs/2112.00038>

<https://arxiv.org/abs/2209.15624>



NEEMo

<https://github.com/okitouni/EnergyMover-Dual/tree/neurips2022>

Monotonenorm

<https://github.com/niklasnlte/MonotOneNorm>

`pip install monotonenorm`

`conda install monotonenorm -c okitouni`

PYTORCH