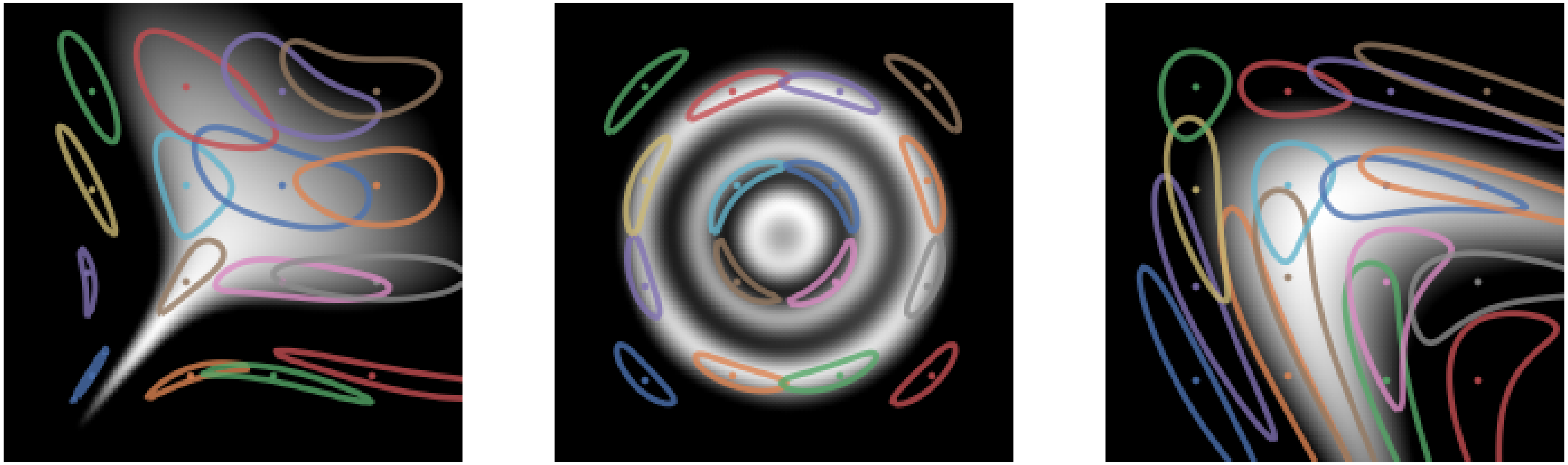
Learning Deep Kernels for Exponential Family Densities

Li K. Wenliang   D. J. Sutherland   H. Strathmann   A. Gretton
Gatsby unit, University College London



Poster #221

Kernel exponential families

- Classic exponential family:

$$p_\eta(\mathbf{x}) = \exp\Big(\big\langle \underbrace{\eta}_{\substack{\text{natural}\\ \text{parameter}}}, \underbrace{T(\mathbf{x})}_{\substack{\text{sufficient}\\ \text{statistic}}} \big\rangle_{\mathbb{R}^s}\Big) \underbrace{q(\mathbf{x})}_{\substack{\text{base}\\ \text{measure}}} \Big/ \underbrace{Z(\eta)}_{\text{normalizer}}$$

  - Gaussian: $T(x) = \begin{bmatrix} x & x^2 \end{bmatrix}$

Kernel exponential families

- Classic exponential family:

$$p_\eta(\mathbf{x}) = \exp\Big(\big\langle \underbrace{\eta}_{\substack{\text{natural}\\\text{parameter}}}, \underbrace{T(\mathbf{x})}_{\substack{\text{sufficient}\\\text{statistic}}} \big\rangle_{\mathbb{R}^s}\Big) \underbrace{q(\mathbf{x})}_{\substack{\text{base}\\\text{measure}}} \;/\; \underbrace{Z(\eta)}_{\text{normalizer}}$$

  - Gaussian: $T(x) = \begin{bmatrix} x & x^2 \end{bmatrix}$
  - Fit depends only on $\mathbb{E}_X\, T(X)$ (and q)

Kernel exponential families

- Classic exponential family:

$$p_\eta(\mathbf{x}) = \exp\big(\langle \underbrace{\eta}_{\substack{\text{natural}\\\text{parameter}}}, \underbrace{T(\mathbf{x})}_{\substack{\text{sufficient}\\\text{statistic}}} \rangle_{\mathbb{R}^s}\big) \; \underbrace{q(\mathbf{x})}_{\substack{\text{base}\\\text{measure}}} \; / \; \underbrace{Z(\eta)}_{\text{normalizer}}$$

  - Gaussian: $T(x) = \begin{bmatrix} x & x^2 \end{bmatrix}$
  - Fit depends only on $\mathbb{E}_X\, T(X)$ (and q)
- Kernel exponential family: $T(\mathbf{x}) = k(\mathbf{x}, \cdot) \in \mathcal{H}$

Kernel exponential families

- Classic exponential family:

$$p_\eta(\mathbf{x}) = \exp\Big(\big\langle \underbrace{\eta}_{\substack{\text{natural}\\\text{parameter}}}, \underbrace{T(\mathbf{x})}_{\substack{\text{sufficient}\\\text{statistic}}} \big\rangle_{\mathbb{R}^s}\Big) \underbrace{q(\mathbf{x})}_{\substack{\text{base}\\\text{measure}}} \Big/ \underbrace{Z(\eta)}_{\text{normalizer}}$$

  - Gaussian: $T(x) = \begin{bmatrix} x & x^2 \end{bmatrix}$
  - Fit depends only on $\mathbb{E}_X\, T(X)$ (and q)

- Kernel exponential family: $T(\mathbf{x}) = k(\mathbf{x}, \cdot) \in \mathcal{H}$
  - Reproducing property: $\langle f, k(\mathbf{x}, \cdot) \rangle_{\mathcal{H}} = f(\mathbf{x})$

Kernel exponential families

- Classic exponential family:

$$p_\eta(\mathbf{x}) = \exp\Big(\big\langle \underbrace{\eta}_{\substack{\text{natural}\\\text{parameter}}}, \underbrace{T(\mathbf{x})}_{\substack{\text{sufficient}\\\text{statistic}}} \big\rangle_{\mathbb{R}^s}\Big) \underbrace{q(\mathbf{x})}_{\substack{\text{base}\\\text{measure}}} \Big/ \underbrace{Z(\eta)}_{\text{normalizer}}$$

  - Gaussian: $T(x) = \begin{bmatrix} x & x^2 \end{bmatrix}$
  - Fit depends only on $\mathbb{E}_X\, T(X)$ (and q)
- Kernel exponential family: $T(\mathbf{x}) = k(\mathbf{x}, \cdot) \in \mathcal{H}$
  - Reproducing property: $\langle f, k(\mathbf{x}, \cdot)\rangle_{\mathcal{H}} = f(\mathbf{x})$
  - So $p_f(\mathbf{x}) = \exp(f(\mathbf{x}))q(\mathbf{x})/Z(f)$

Why kernel exponential families

$$p_f(\mathbf{x}) = \exp\big(f(\mathbf{x})\big)\, q(\mathbf{x}) / Z(f)$$

Why kernel exponential families

$$p_f(\mathbf{x}) = \exp\big(f(\mathbf{x})\big)\, q(\mathbf{x}) / Z(f)$$

- Any density with $\log p - \log q \in \mathcal{H}$

Why kernel exponential families

$$p_f(\mathbf{x}) = \exp\bigl(f(\mathbf{x})\bigr)\, q(\mathbf{x}) / Z(f)$$

- Any density with $\log p - \log q \in \mathcal{H}$
- *Much* richer class; e.g. with Gaussian k, dense in all continuous distributions on compact domains

Why kernel exponential families

$$p_f(\mathbf{x}) = \exp\big(f(\mathbf{x})\big)\, q(\mathbf{x}) / Z(f)$$

- Any density with $\log p - \log q \in \mathcal{H}$
- *Much* richer class; e.g. with Gaussian k, dense in all continuous distributions on compact domains

Why kernel exponential families

$$p_f(\mathbf{x}) = \exp\big(f(\mathbf{x})\big)\, q(\mathbf{x}) / Z(f)$$

- Any density with $\log p - \log q \in \mathcal{H}$
- *Much* richer class; e.g. with Gaussian k, dense in all continuous distributions on compact domains

Why kernel exponential families

$$p_f(\mathbf{x}) = \exp\big(f(\mathbf{x})\big)\, q(\mathbf{x}) / Z(f)$$

- Any density with $\log p - \log q \in \mathcal{H}$
- *Much* richer class; e.g. with Gaussian k, dense in all continuous distributions on compact domains
- Fit with *score matching*

$$\min_f \mathbb{E}\left[\sum_{d=1}^{D} \frac{\partial^2}{\partial X_d^2} \log p_f(X) + \frac{1}{2}\left(\frac{\partial}{\partial X_d} \log p_f(X)\right)^2\right]$$

Choosing a kernel with meta-learning

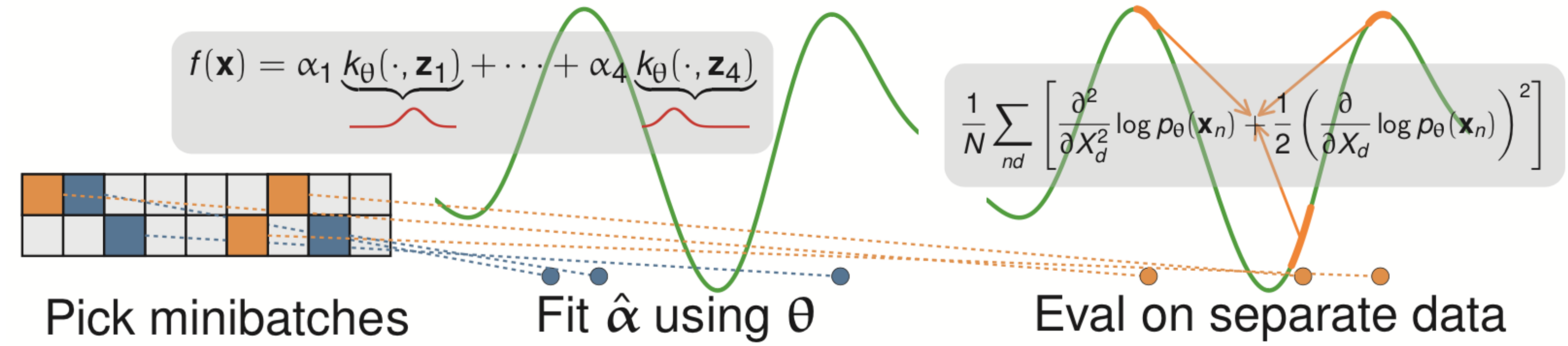
- Fit quality depends a lot on kernel choice
  - Also on the regularization weight
- Need to fit these parameters

Choosing a kernel with meta-learning

- Fit quality depends a lot on kernel choice
  - Also on the regularization weight
- Need to fit these parameters
- … but need to use held-out data to avoid trivially overfitting

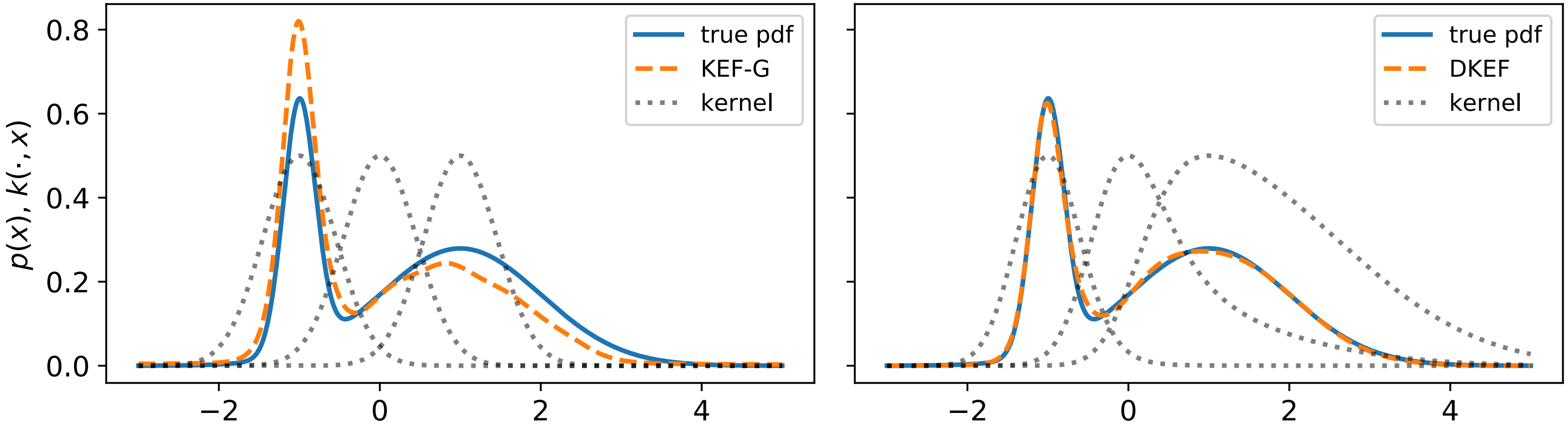
Choosing a kernel with meta-learning

- Fit quality depends a lot on kernel choice
  - Also on the regularization weight
- Need to fit these parameters
- … but need to use held-out data to avoid trivially overfitting
- Meta-learning: take ∇_θ of whole fit on a minibatch



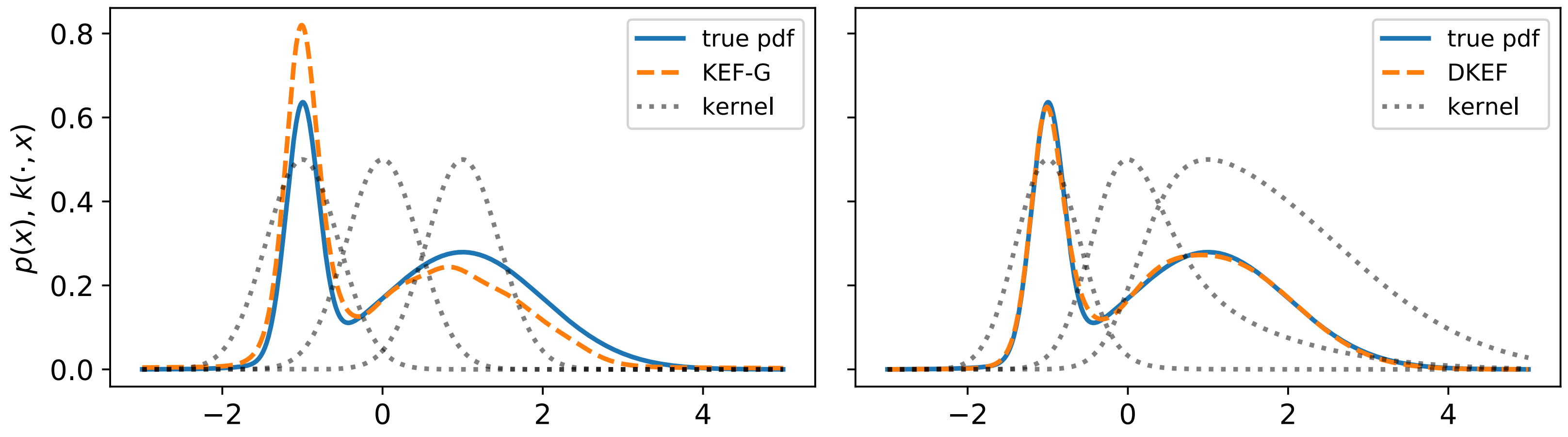
Deep kernels

- Simple kernels, e.g. $\exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x}-\mathbf{y}\|^2\right)$, aren't enough:



Deep kernels

- Simple kernels, e.g. $\exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x}-\mathbf{y}\|^2\right)$, aren't enough:



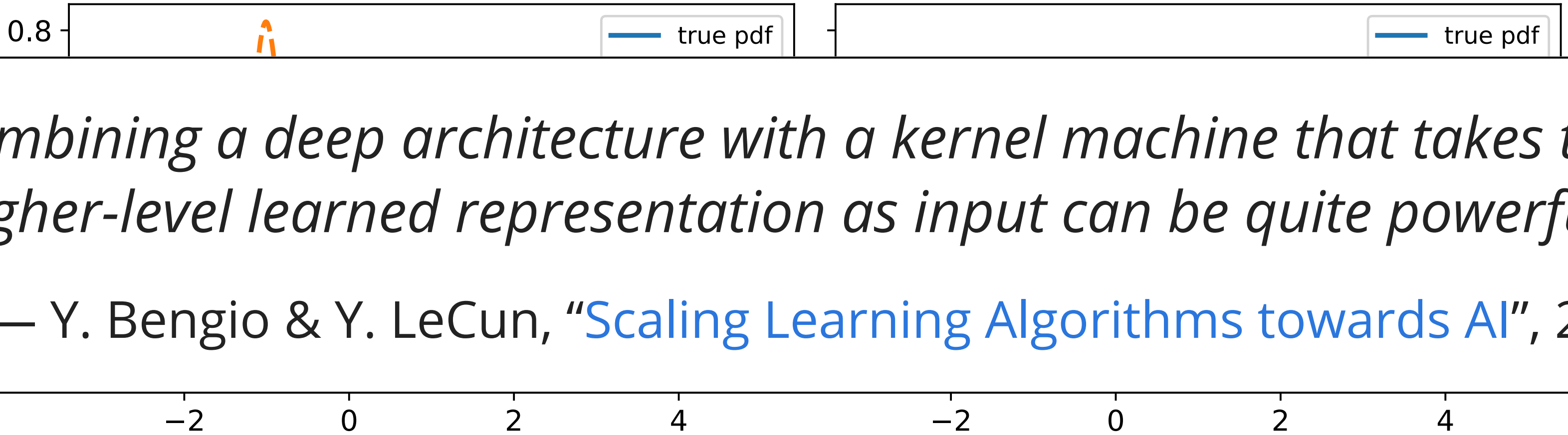
- But we can learn lots of parameters with gradient descent:

$$k(\mathbf{x},\mathbf{y}) = k_{\mathrm{top}}\left(\boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{y})\right)$$

with $\boldsymbol{\phi}$ a neural net, k_{top} something simple

Deep kernels

- Simple kernels, e.g. $\exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x}-\mathbf{y}\|^2\right)$, aren't enough:



> *Combining a deep architecture with a kernel machine that takes the higher-level learned representation as input can be quite powerful.*
>
> — Y. Bengio & Y. LeCun, "Scaling Learning Algorithms towards AI", 2007

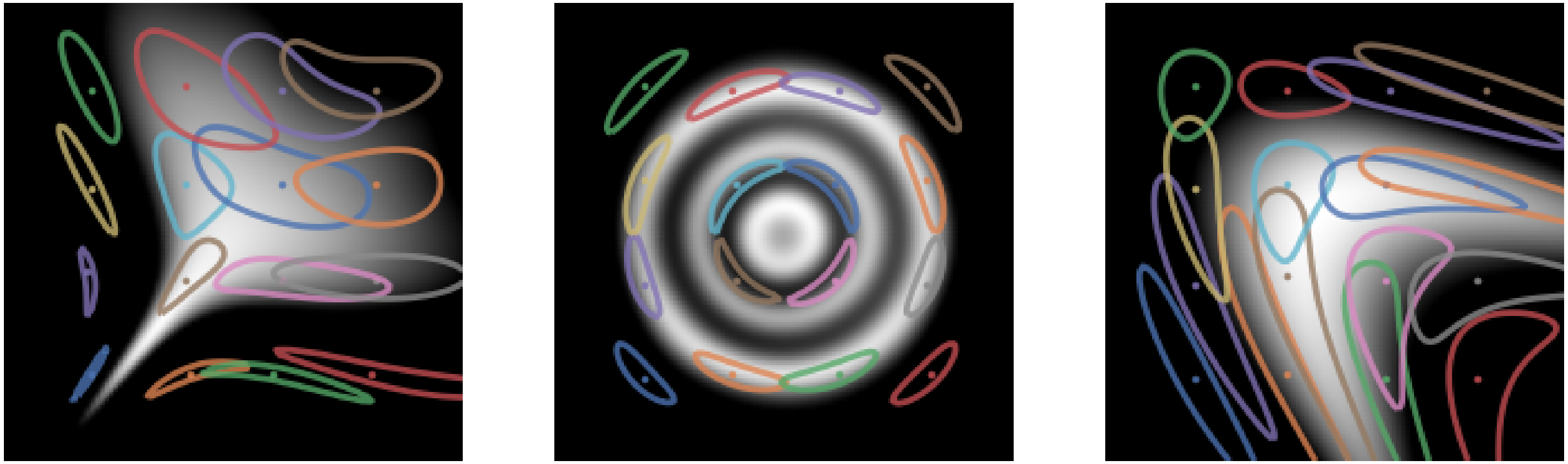
- But we can learn lots of parameters with gradient descent:

$$k(\mathbf{x},\mathbf{y}) = k_{\mathrm{top}}\left(\boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{y})\right)$$

with $\boldsymbol{\phi}$ a neural net, k_{top} something simple

Results

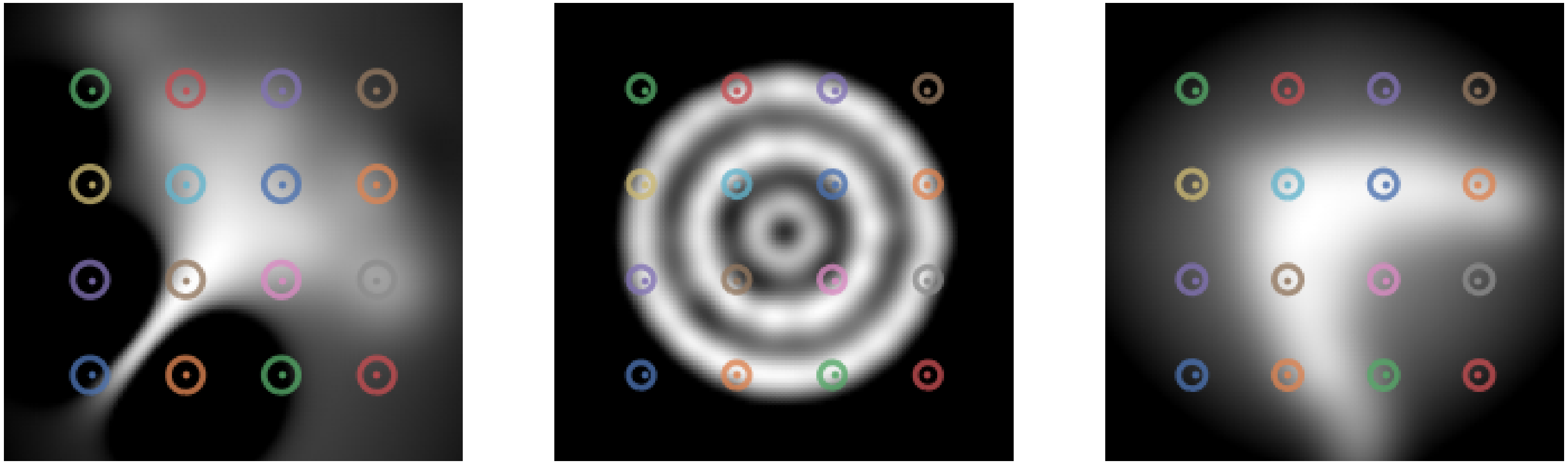
- Learns local dataset geometry: better fits



- On real data: slightly worse likelihoods, maybe better "shapes" than deep likelihood models

Results

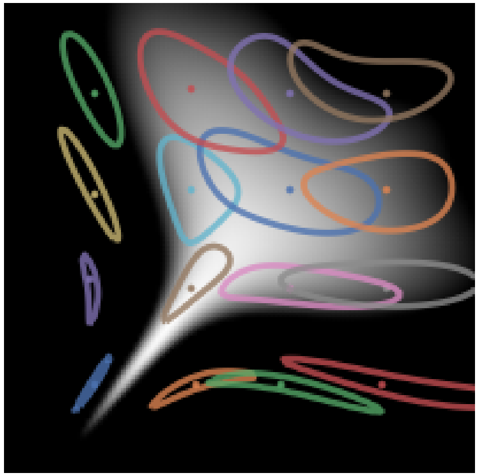
- Learns local dataset geometry: better fits

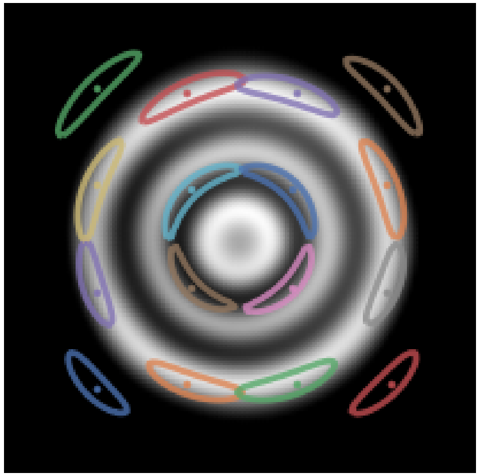


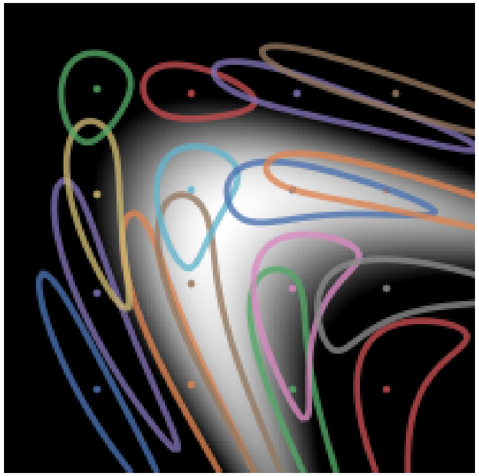
- On real data: slightly worse likelihoods, maybe better "shapes" than deep likelihood models

Results

- Learns local dataset geometry: better fits







- On real data: slightly worse likelihoods, maybe better "shapes" than deep likelihood models