Scalable, Flexible and Active Learning on Distributions

Dougal J. Sutherland

Thesis Committee:

Jeff Schneider (Chair)

Nina Balcan

Barnabás Póczos

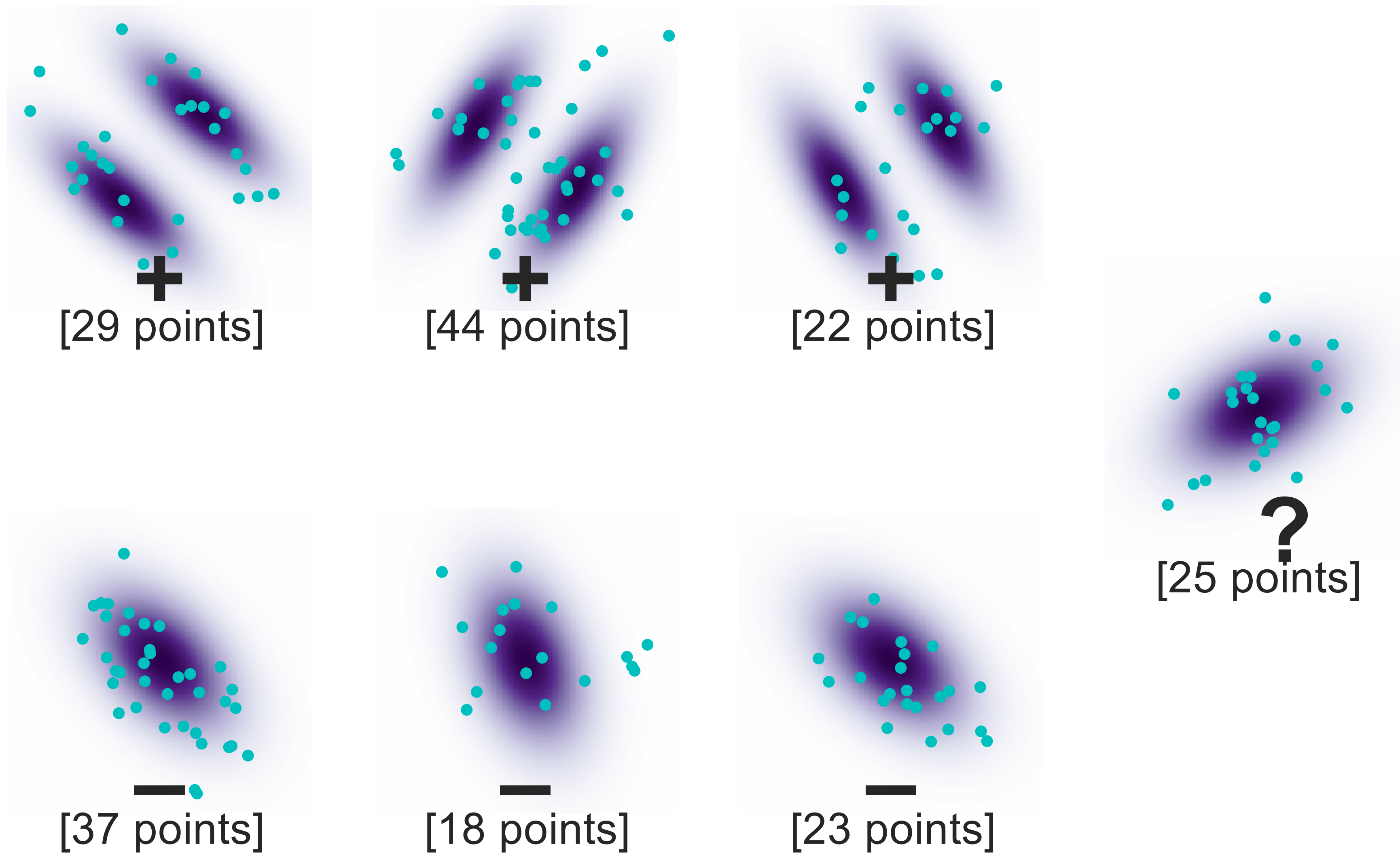
Arthur Gretton (UCL)

Auton Lab

Learning on Distributions

We want to learn a distribution classification function....

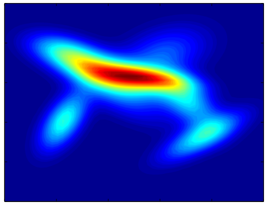
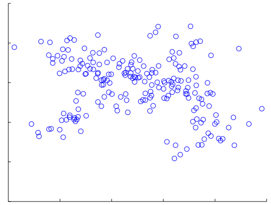
Learning on Distributions



...based on sample sets.

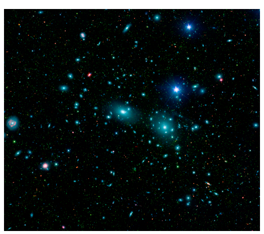
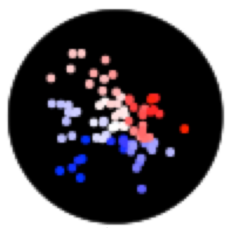
Learning on Distributions

distribution | observed sample | label

9 components


“seaside city”

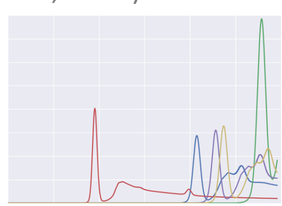



Ntampaka et al. (ApJ 2015, 2016)

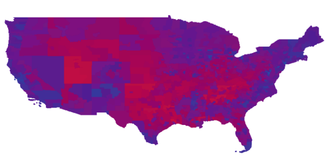
Mass 7×10^{14} $M_\odot$

and more…

Jin et al. (NSS 2016)



no Cs137 present



	AGEP	SEX	...	RACSOR	RACWHT
0	75	1	...	0	1
1	25	0	...	0	1
...	...	...	...	...	...

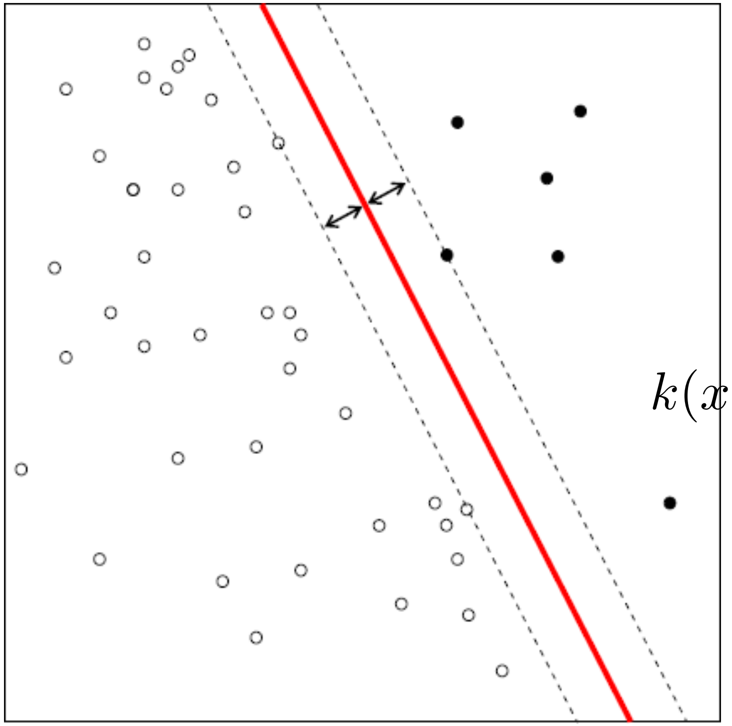
Flaxman et al. (KDD 2015)

county voted
54% for Obama

Contributions

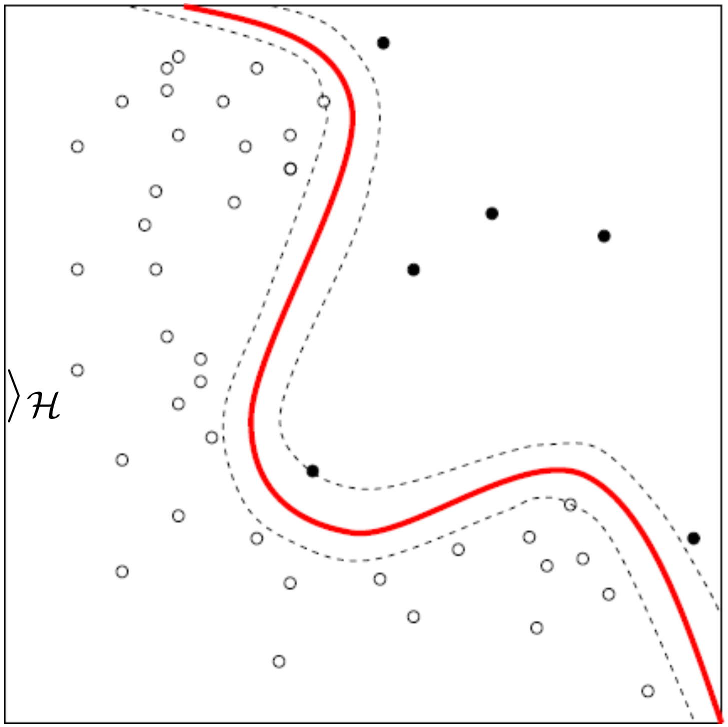
- **Learning on distributions** with nonparametric kernels
- **Scalable** approximate kernel embeddings
  - Random Fourier features analysis
  - New embeddings for distribution kernels
- **Flexible** distribution kernels
  - Deep mean maps in computer vision
  - MMD kernel learning for testing
- **Active** pointillistic pattern search

Kernel Methods



$$\phi : \mathbb{R}^d \to \mathcal{H}$$

$$k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$$



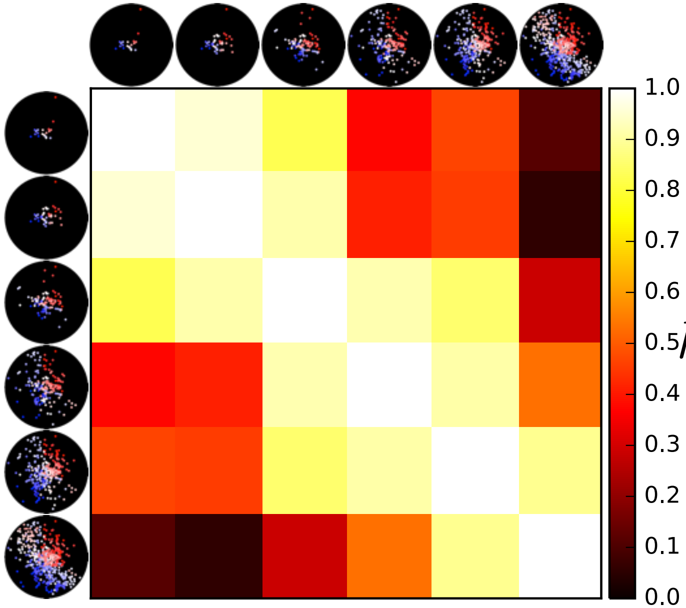
$$f(x) = w^{\mathsf{T}} x + b$$

$$f(x) = \langle w, \phi(x) \rangle_{\mathcal{H}} + b$$
$$= \sum_{i=1}^{n} \alpha_i \, y_i \, k(x_i, x) + b$$

Linear models…

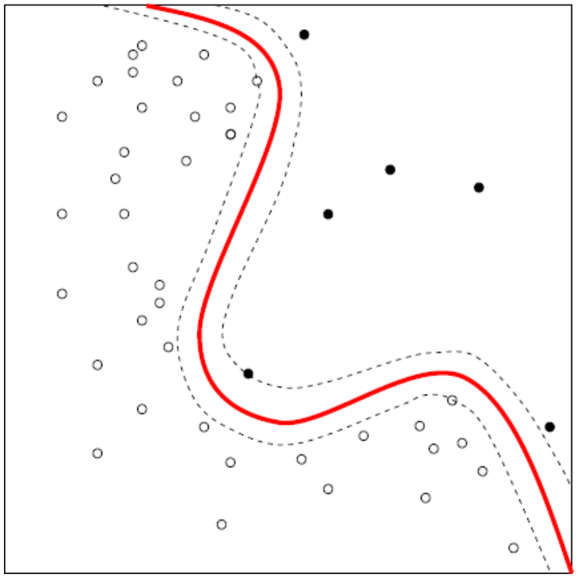
…in Hilbert space.

Kernel Methods



$$\phi : \mathcal{X} \to \mathcal{H}$$

$$k(x, x') = \langle \phi(x), \phi(x') \rangle_{\mathcal{H}}$$



Can use a kernel on any domain.

$$f(x) = \langle w, \phi(x) \rangle_{\mathcal{H}} + b = \sum_{i=1}^{n} \alpha_i \, y_i \, k(x_i, x) + b$$

Linear models…in Hilbert space.

Kernels on Distributions

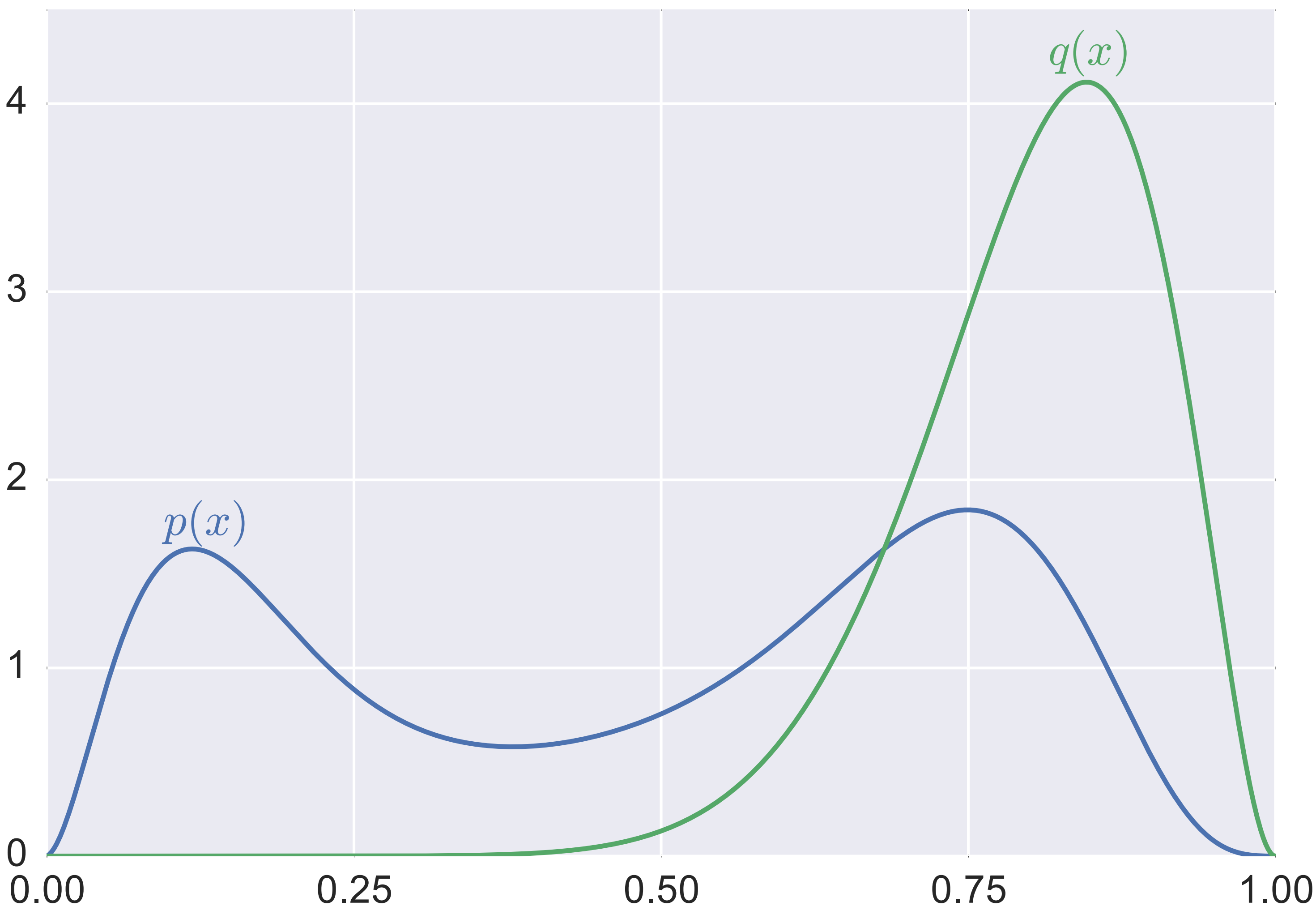
We'll use a kernel on distributions based on a distance ρ:

$$K(\mathbb{P}, \mathbb{Q}) = \exp\left(-\frac{1}{2\sigma^2}\rho^2(\mathbb{P}, \mathbb{Q})\right)$$

The popular Gaussian RBF kernel has this form, with ρ the Euclidean distance between vectors.

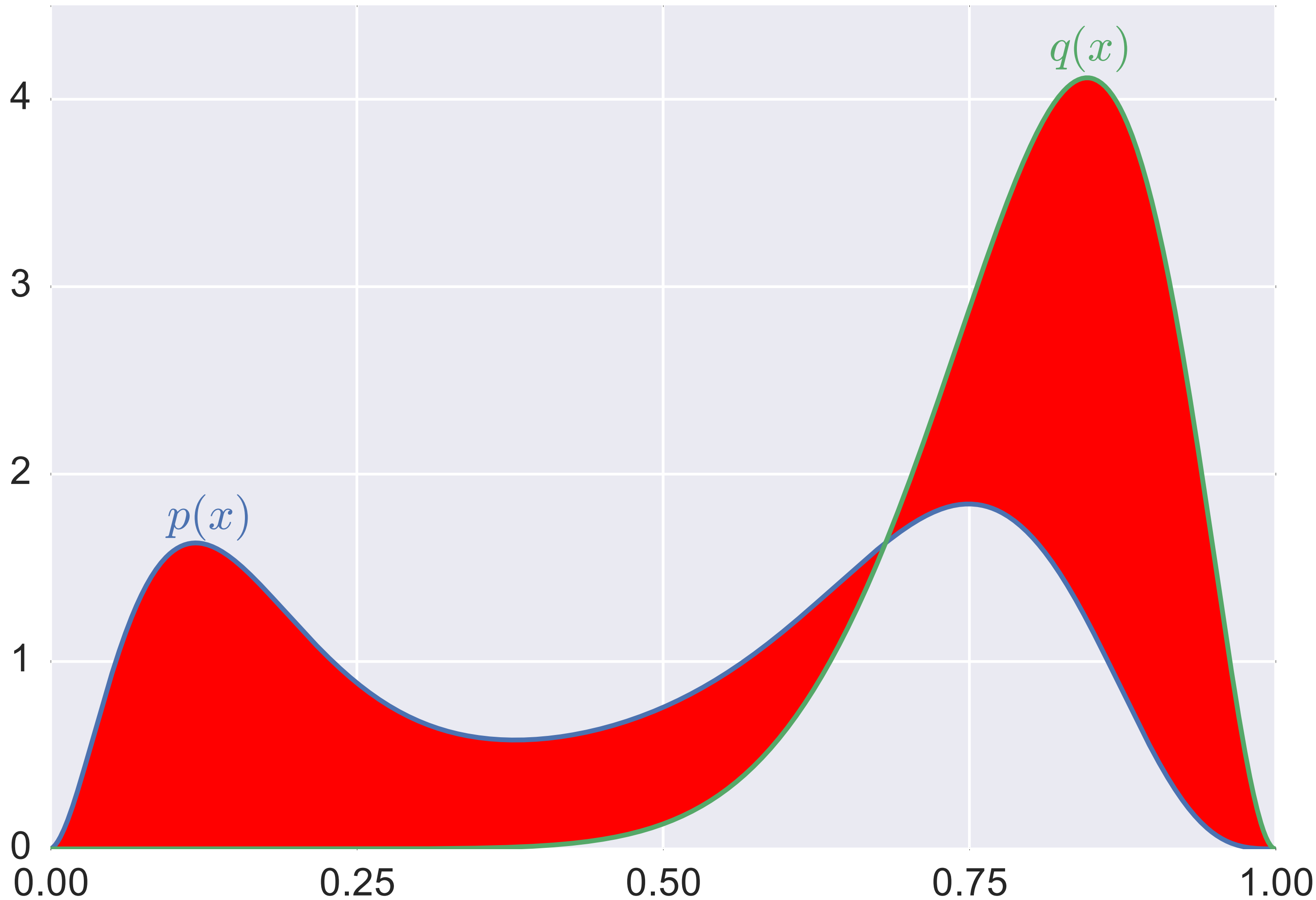
A valid kernel as long as ρ is *Hilbertian*.

Distances on Distributions



Distances on Distributions

$$\mathrm{TV}(\mathbb{P}, \mathbb{Q}) = \int \frac{1}{2} |p(x) - q(x)| \, \mathrm{d}x$$



Distances on Distributions

Total Variation

$$\mathrm{TV}(p,q) = \int \tfrac{1}{2}|p(x) - q(x)|\,\mathrm{d}x$$

L_2

$$L_2^2(p,q) = \int (p(x) - q(x))^2\,\mathrm{d}x$$

Hellinger

$$\mathrm{H}^2(p,q) = \int \tfrac{1}{2}\left(\sqrt{p(x)} - \sqrt{q(x)}\right)^2\,\mathrm{d}x$$

Kullback-Leibler

$$\mathrm{KL}(p\|q) = \int p(x)\log\frac{p(x)}{q(x)}\,\mathrm{d}x$$

Rényi-α

$$\mathrm{R}_\alpha(p\|q) = \frac{1}{\alpha-1}\int p(x)^\alpha q(x)^{1-\alpha}\,\mathrm{d}x$$

Jensen-Shannon

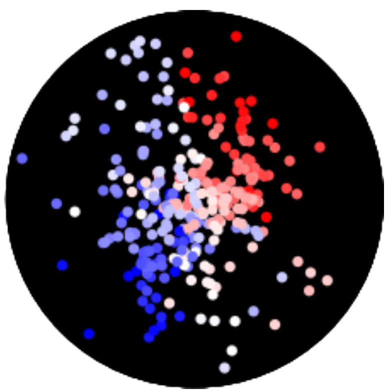
$$\mathrm{JS}(p,q) = \tfrac{1}{2}\mathrm{KL}\left(p\middle\|\frac{p+q}{2}\right) + \tfrac{1}{2}\mathrm{KL}\left(q\middle\|\frac{p+q}{2}\right)$$

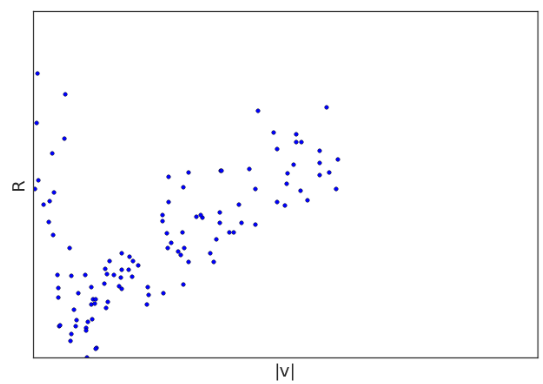
Maximum Mean Discrepancy

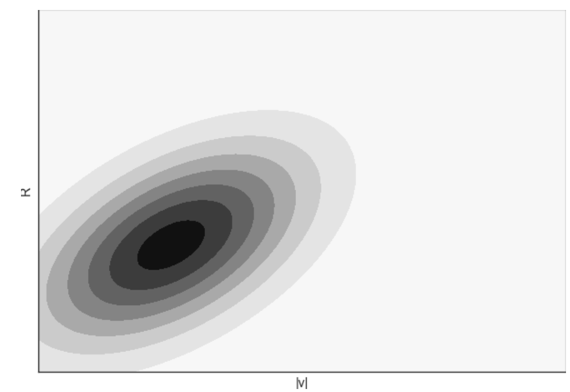
$$\mathrm{MMD}_k(\mathbb{P},\mathbb{Q}) = \sup_{f\in\mathcal{H}_k} \mathbb{E}_{X\sim\mathbb{P}}\, f(X) - \mathbb{E}_{Y\sim\mathbb{Q}}\, f(Y)$$

Estimators of Distributional Distances

- Fit a parametric model and compute distances.
  - Some distances have closed form for some models.
  - Model introduces approximation error







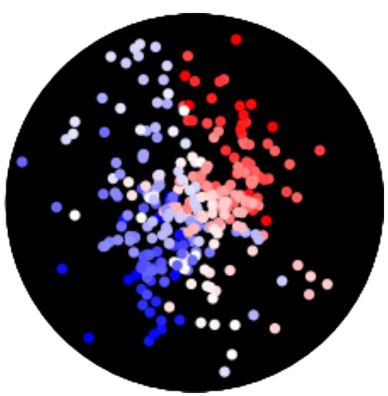
μ, Σ

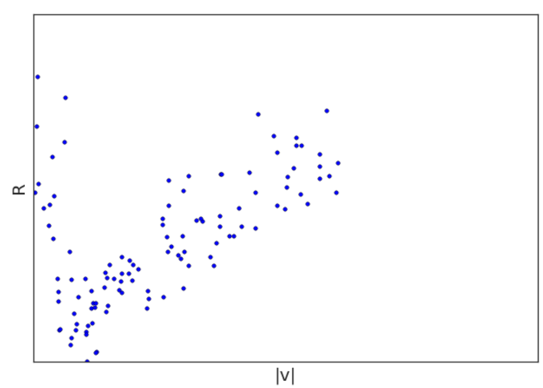
$$L_2(\mathcal{N}(\mu,\Sigma),\mathcal{N}(\mu',\Sigma')) = \frac{1}{|4\pi\Sigma|^{\frac{1}{2}}} + \frac{1}{|4\pi\Sigma'|^{\frac{1}{2}}} - 2\frac{\exp\left(-\frac{1}{2}(\mu-\mu')^T(\Sigma+\Sigma')^{-1}(\mu-\mu')\right)}{|2\pi(\Sigma+\Sigma')|^{\frac{1}{2}}}$$

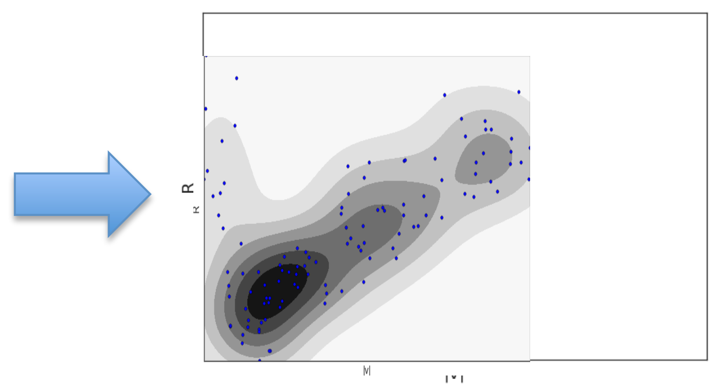
Jebara et al. 2004; Moreno et al. 2004

Estimators of Distributional Distances

- Fit a nonparametric model and compute distances.
  - Histograms
  - Kernel density estimation
  - *k*-nearest neighbor density estimation
  - Basis function projections
  - Empirical distribution (for MMD)







Algorithm: Learning on Distributions

Given sample sets $X_i \sim \mathbb{P}_i$, a distance ρ, and a b.w. σ:

1. Estimate $\hat{\rho}(X_i, X_j)$ for all *i*, *j* nonparametrically.

2. Assemble into a kernel matrix

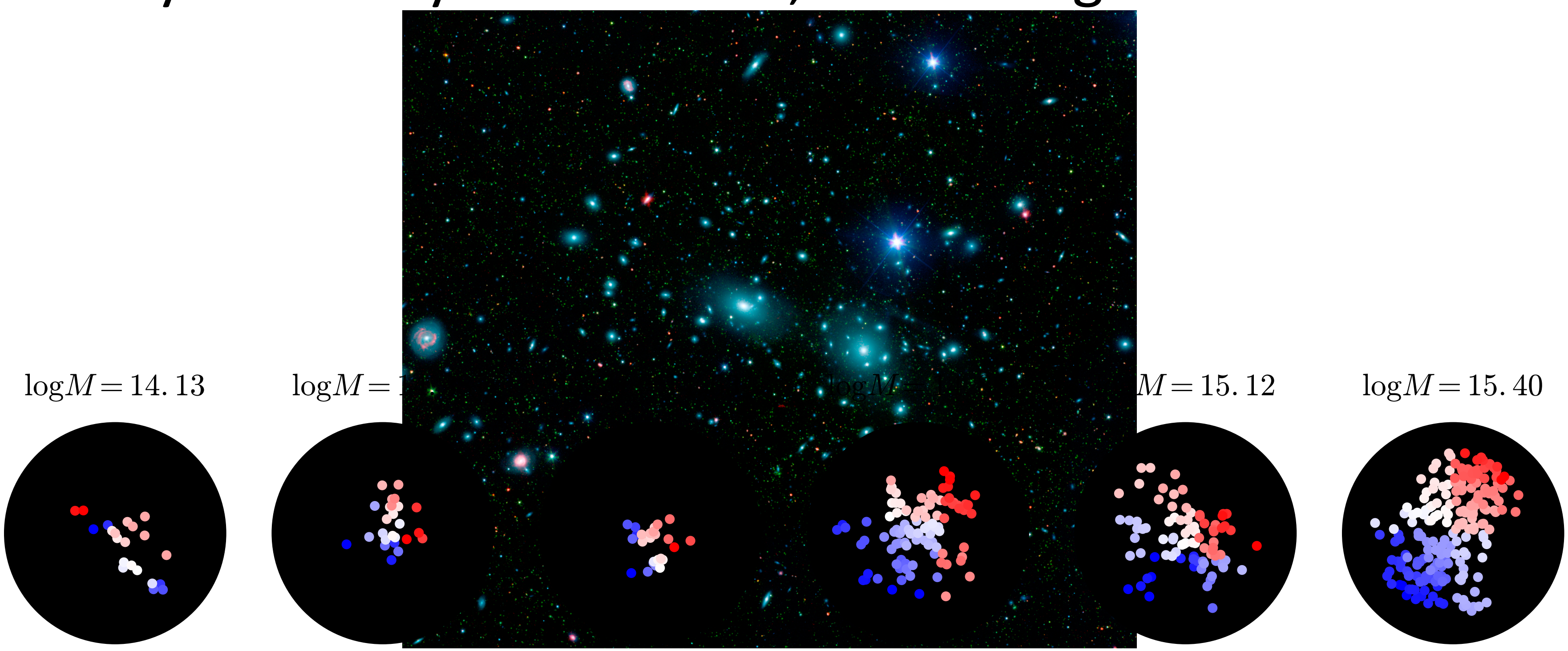
$$K_{ij} = \exp\left(-\frac{1}{2\sigma^2}\hat{\rho}(X_i, X_j)\right).$$

3. Run an SVM / GP / ridge regression / … with *K*.

Application: Galaxy Cluster Mass

Galaxy clusters are fundamental in the study the universe. Their properties can tell us a lot about cosmology.

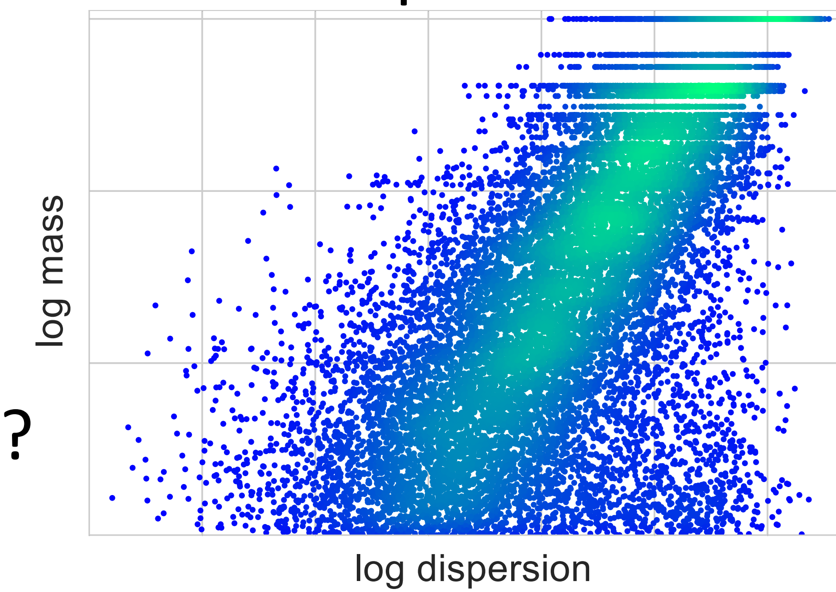
But they're mostly dark matter; measuring their mass is hard.



Coma cluster, NASA

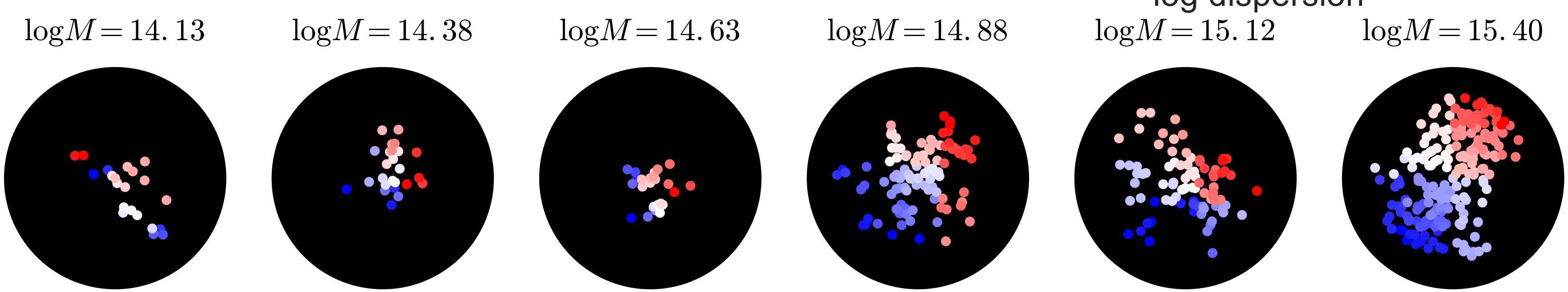
Application: Galaxy Cluster Mass

Fritz Zwicky (1933): Under reasonable assumptions, the velocity dispersion has power-law relationship to total mass.



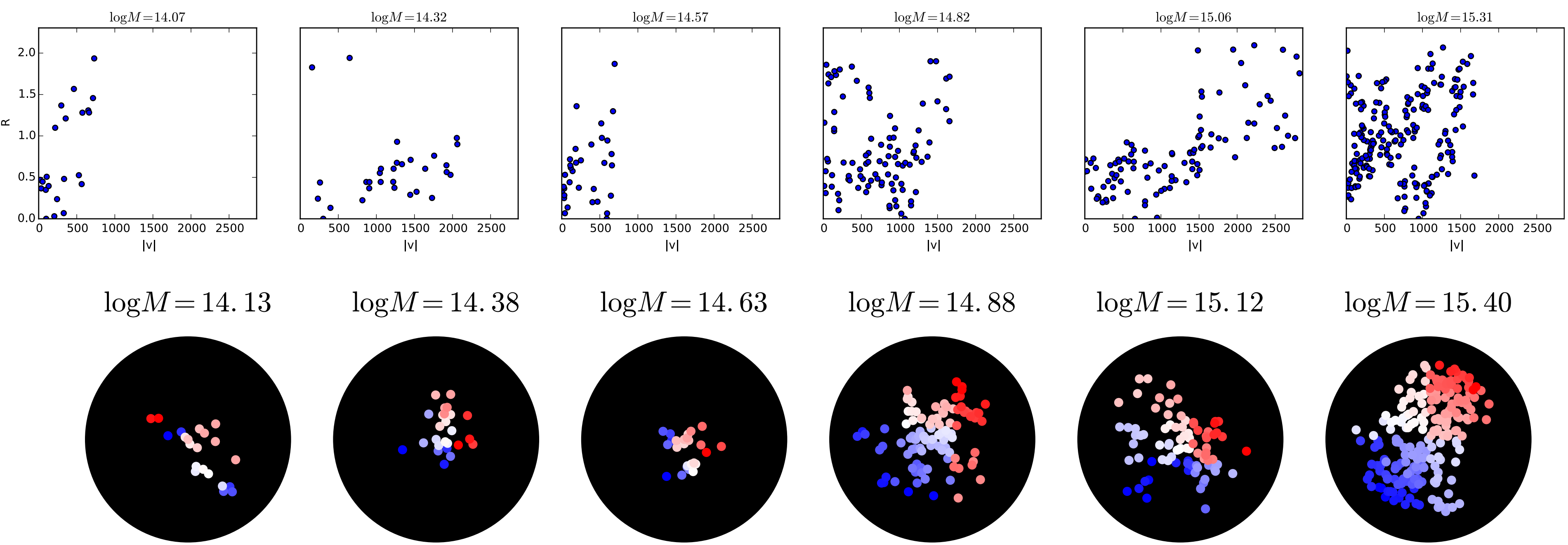
Not so great…

- Assumptions violated
- Which galaxies are in cluster?



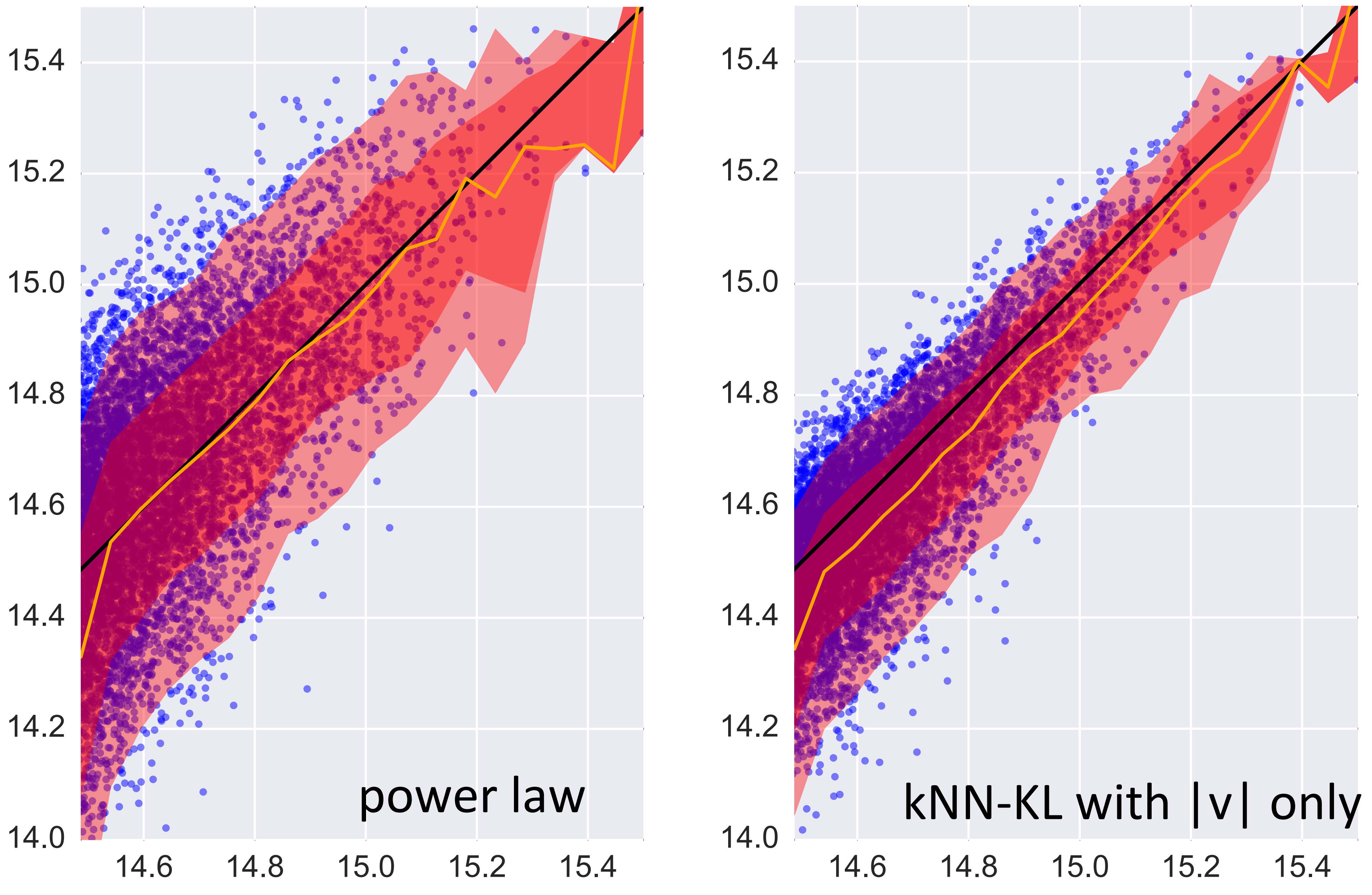
Application: Galaxy Cluster Mass

Alternative approach: consider each cluster as a *distribution* of galaxy features, and regress from these distributions to total mass.

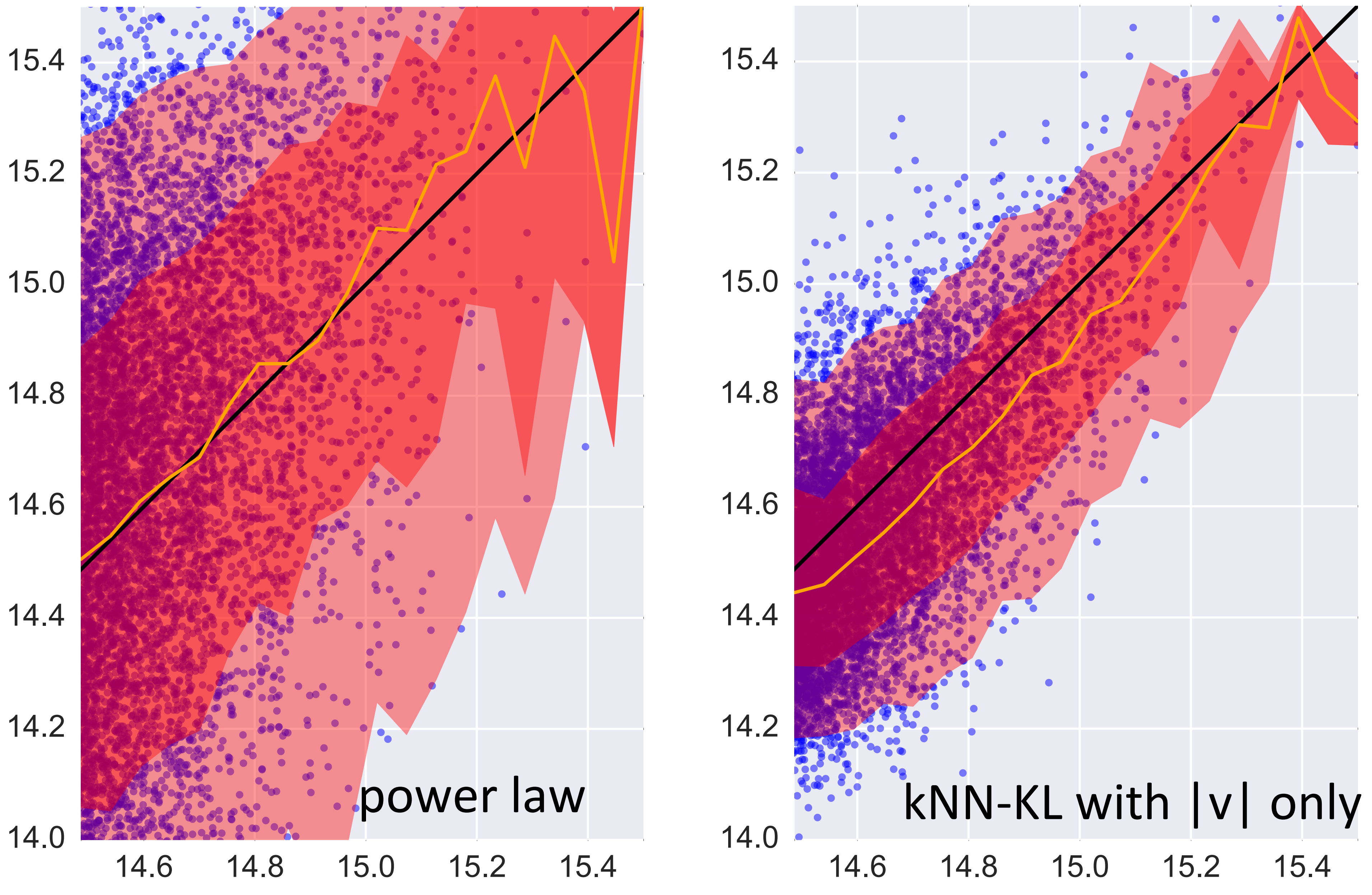


Ntampaka, Trac, Sutherland et al., *ApJ* 2015; Ntampaka, Trac, Sutherland et al., in press at *ApJ*.

Cluster Mass: Known Membership



Cluster Mass: With Interlopers



Contributions

- **Learning on distributions** with nonparametric kernels
- **Scalable** approximate kernel embeddings
  - Random Fourier features analysis
  - New embeddings for distribution kernels
- **Flexible** distribution kernels
  - Deep mean maps in computer vision
  - MMD kernel learning for testing
- **Active** pointillistic pattern search

Scalability

These methods need an *n x n* Gram matrix:

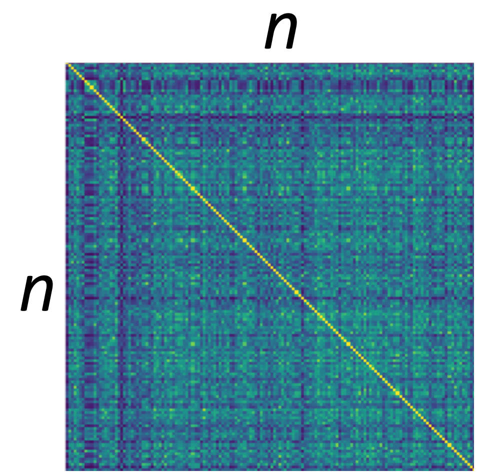
$$\mathbf{K} = \begin{bmatrix} K(x_1, x_1) & K(x_1, x_2) & \dots & K(x_1, x_n) \\ K(x_2, x_1) & K(x_2, x_2) & \dots & K(x_2, x_n) \\ \vdots & \vdots & \ddots & \vdots \\ K(x_n, x_1) & K(x_n, x_2) & \dots & K(x_n, x_n) \end{bmatrix}$$

The matrix is n^2; operations can be as slow as $O(n^3)$.

Linear-kernel models usually scale like $O(n)$.

Approximate Embeddings

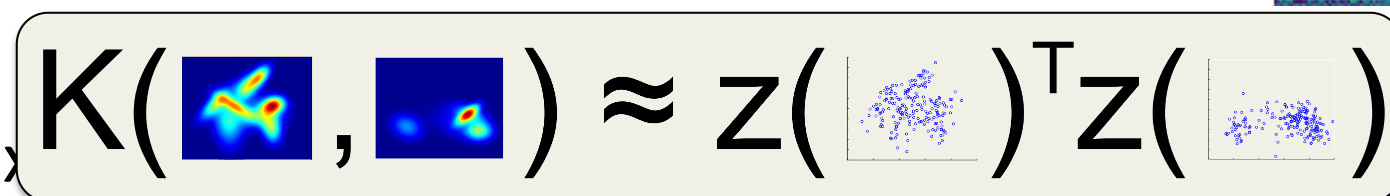
Traditional kernel methods:



$\{x_i\}$ $\{\varphi(x_i)\} \subset \mathcal{H}$

$\langle \varphi(x_i), \varphi(x_j) \rangle_{\mathcal{H}} = k(x_i, y_j)$

$f(x) = \langle w, \varphi(x) \rangle + b = \sum^{n} \alpha_i y_i k(x_i, x) + b$

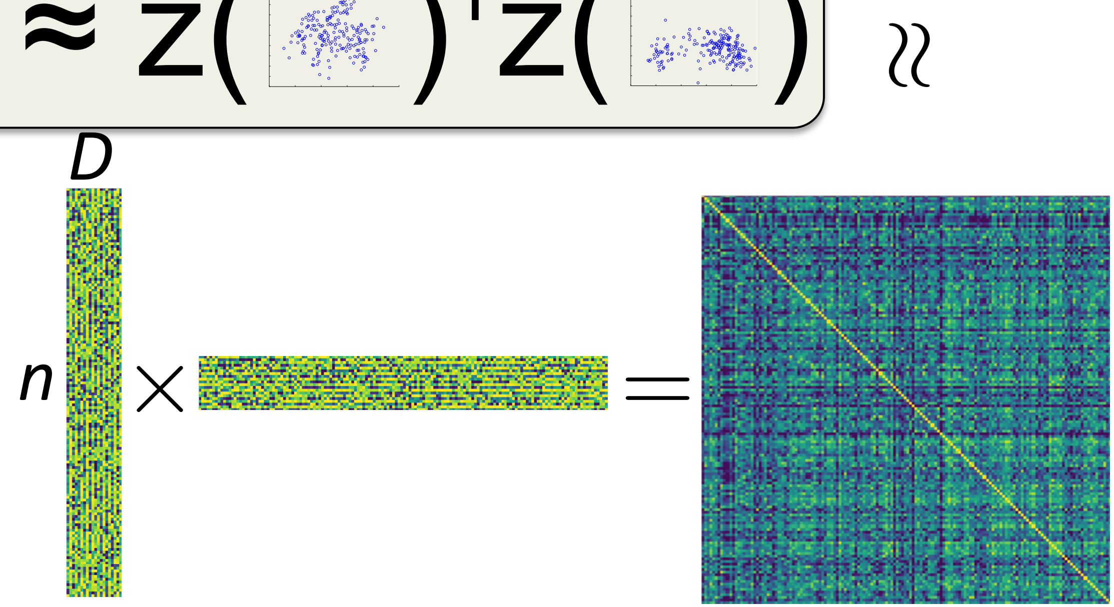


Approx

$\{x_i\}$ $\{z(x_i)\} \subset \mathbb{R}^D$

$z(x_i)^T z(x_j)$

$f(x) = w^T z(x) + b$



Random Fourier Features

Rahimi and Recht (2007) developed random Fourier features, a.k.a. "random kitchen sinks":

$$k(x,y) = \underline{k}(\overbrace{\Delta}^{x-y}) = \exp\left(-\frac{1}{2\sigma^2}\|\Delta\|^2\right)$$

$$= \langle \varphi(x), \varphi(y) \rangle_{\mathcal{H}} \qquad (\dim \mathcal{H} = \infty)$$

$$\approx z(x)^\mathsf{T} z(y) \qquad (z : \mathbb{R}^d \to \mathbb{R}^D)$$

Random Fourier Features

Rahimi and Recht (2007) developed random Fourier features, a.k.a. "random kitchen sinks":

$$k(x, y) = \underline{k}(\overbrace{\Delta}^{x-y}) = \exp\left(-\frac{1}{2\sigma^2}\|\Delta\|^2\right) \approx z(x)^\mathsf{T} z(y)$$

$$\Omega(\omega) := \int \underline{k}(\Delta) \exp\left(-\mathbb{i}\omega^\mathsf{T}\Delta\right) \mathrm{d}\Delta \propto \exp\left(-\frac{\sigma^2}{2}\|\Delta\|^2\right)$$

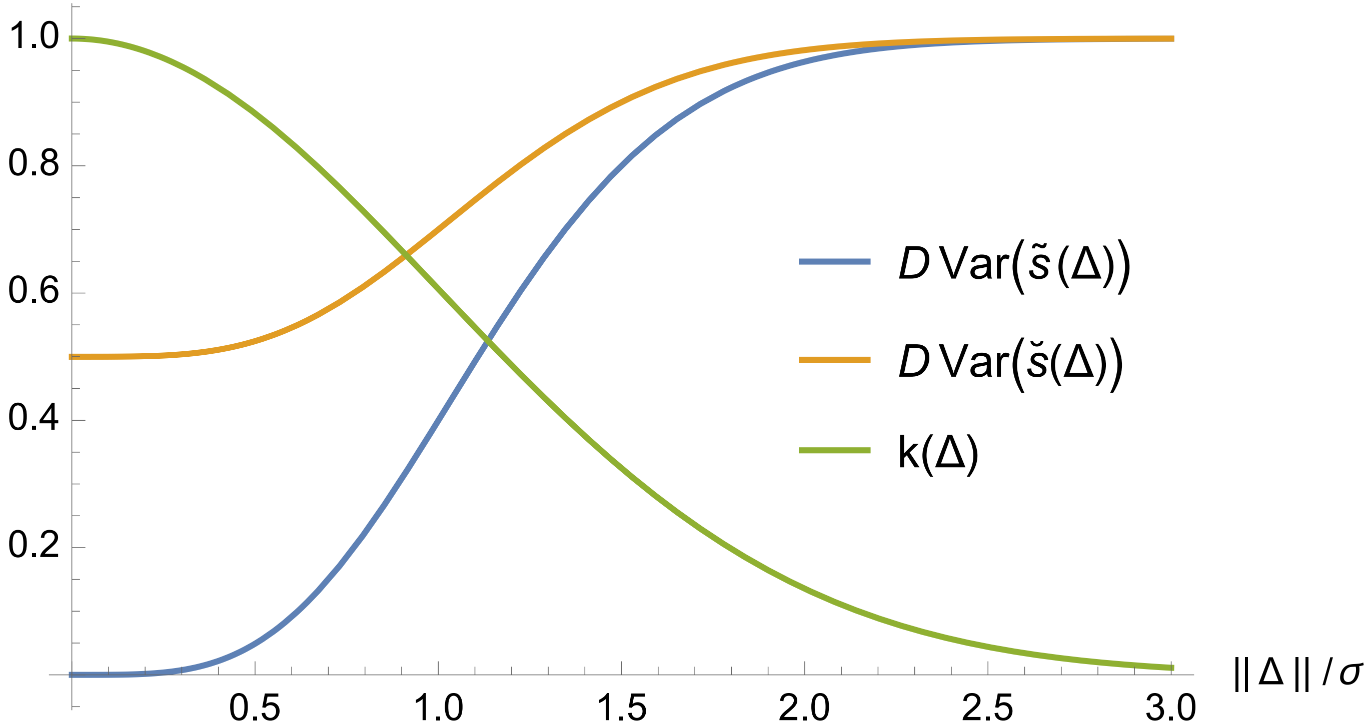
Bochner's theorem: *Ω* is a (scaled) probability distribution.

$$\omega_i \sim \Omega$$

$$\tilde{z}(x) = \sqrt{\frac{2}{D}} \begin{bmatrix} \sin(\omega_1^\mathsf{T} x) & \cos(\omega_1^\mathsf{T} x) & \dots & \sin(\omega_{D/2}^\mathsf{T} x) & \cos(\omega_{D/2}^\mathsf{T} x) \end{bmatrix}^\mathsf{T}$$

$$\breve{z}(x) = \sqrt{\frac{2}{D}} \begin{bmatrix} \cos(\omega_1^\mathsf{T} x + b_1) & \dots & \cos(\omega_D^\mathsf{T} x + b_D) \end{bmatrix}^\mathsf{T} \quad b_i \sim \mathrm{Unif}(0, 2\pi)$$

Random Fourier Features



$$\omega_i \sim \Omega$$

$$\tilde{z}(x) = \sqrt{\frac{2}{D}} \left[\sin(\omega_1^\mathsf{T} x) \quad \cos(\omega_1^\mathsf{T} x) \quad \dots \quad \sin(\omega_{D/2}^\mathsf{T} x) \quad \cos(\omega_{D/2}^\mathsf{T} x) \right]^\mathsf{T}$$

$$\breve{z}(x) = \sqrt{\frac{2}{D}} \left[\cos(\omega_1^\mathsf{T} x + b_1) \quad \dots \quad \cos(\omega_D^\mathsf{T} x + b_D) \right]^\mathsf{T} \; b_i \sim \mathrm{Unif}(0, 2\pi)$$

The L_∞ error is also tighter, in bounds and empirically.

Contributions

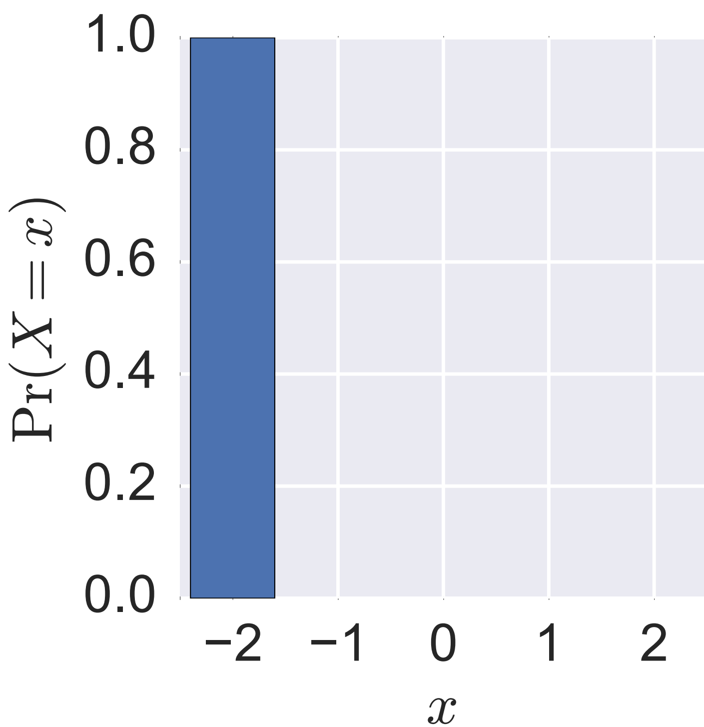
- **Learning on distributions** with nonparametric kernels
- **Scalable** approximate kernel embeddings
  - Random Fourier features analysis
  - New embeddings for distribution kernels
- **Flexible** distribution kernels
  - Deep mean maps in computer vision
  - MMD kernel learning for testing
- **Active** pointillistic pattern search

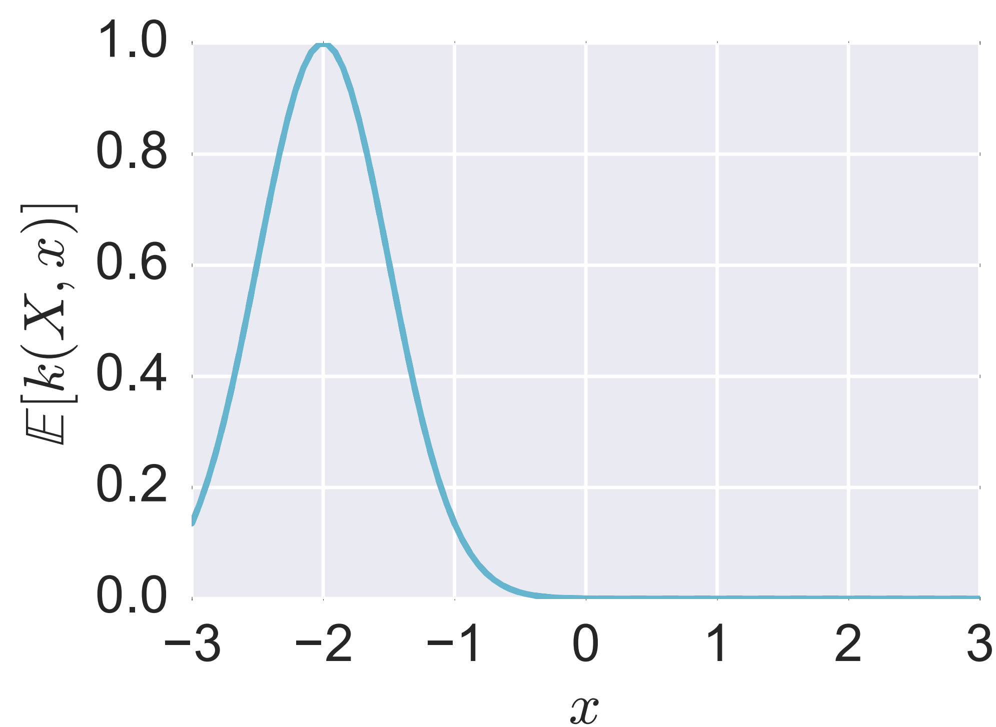
Mean embeddings of distributions

The *mean embedding* of a distribution in an RKHS:

$$\mu_{\mathbb{P}} = \mathbb{E}_{x \sim \mathbb{P}}[\varphi(x)]$$

Remember $\langle \varphi(x), \varphi(y) \rangle = k(x, y)$, so we can think of $\varphi(x)$ as $k(x, \cdot)$.



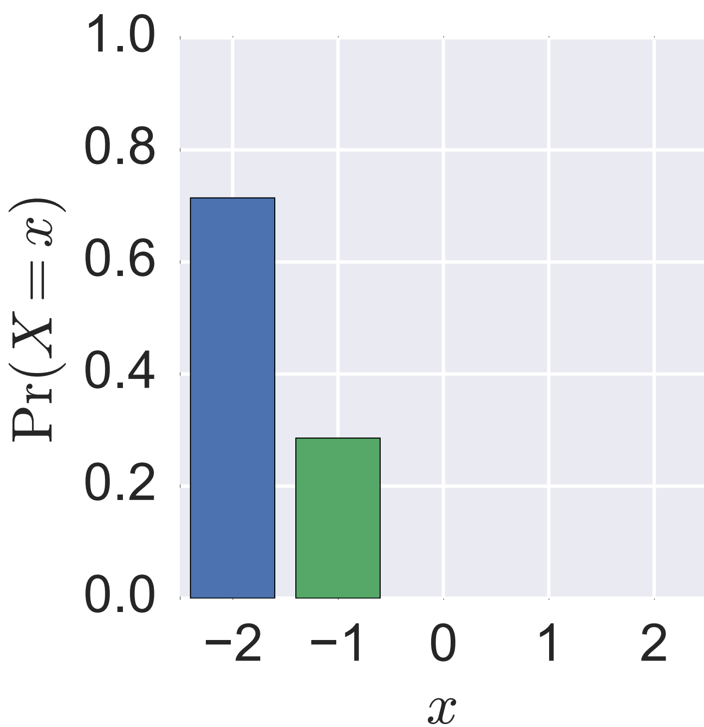


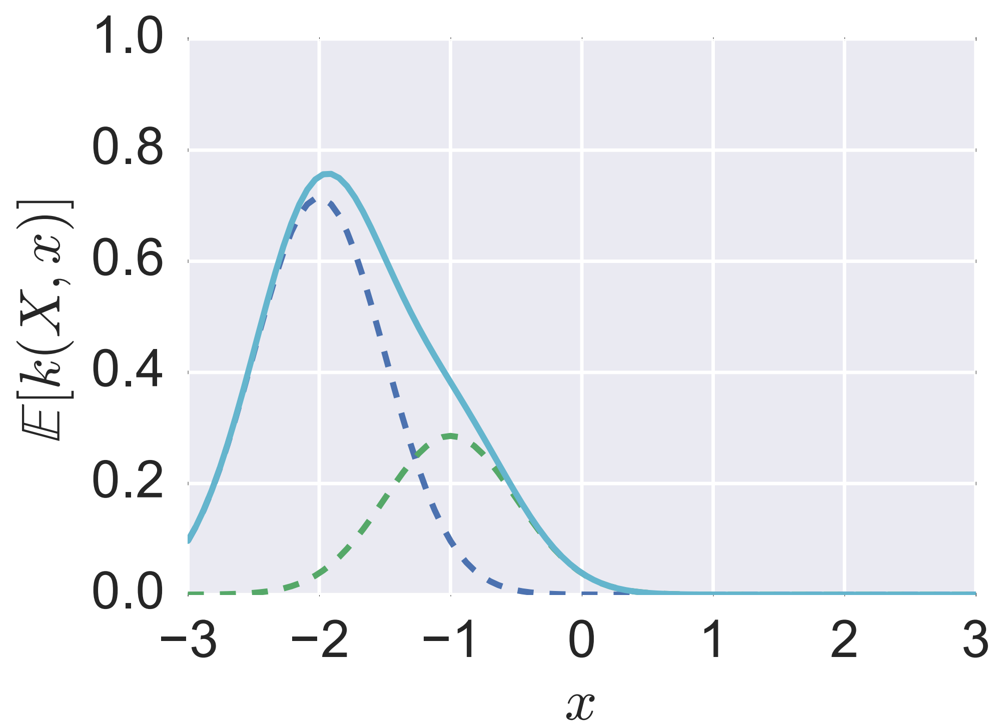
Mean embeddings of distributions

The *mean embedding* of a distribution in an RKHS:

$$\mu_{\mathbb{P}} = \mathbb{E}_{x \sim \mathbb{P}}[\varphi(x)]$$

Remember $\langle \varphi(x), \varphi(y) \rangle = k(x, y)$,
so we can think of $\varphi(x)$ as $k(x, \cdot)$.



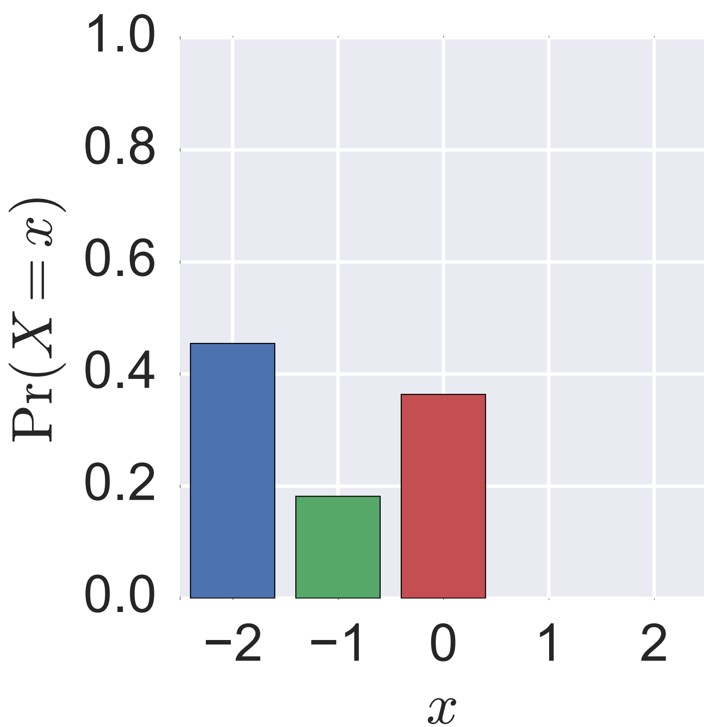


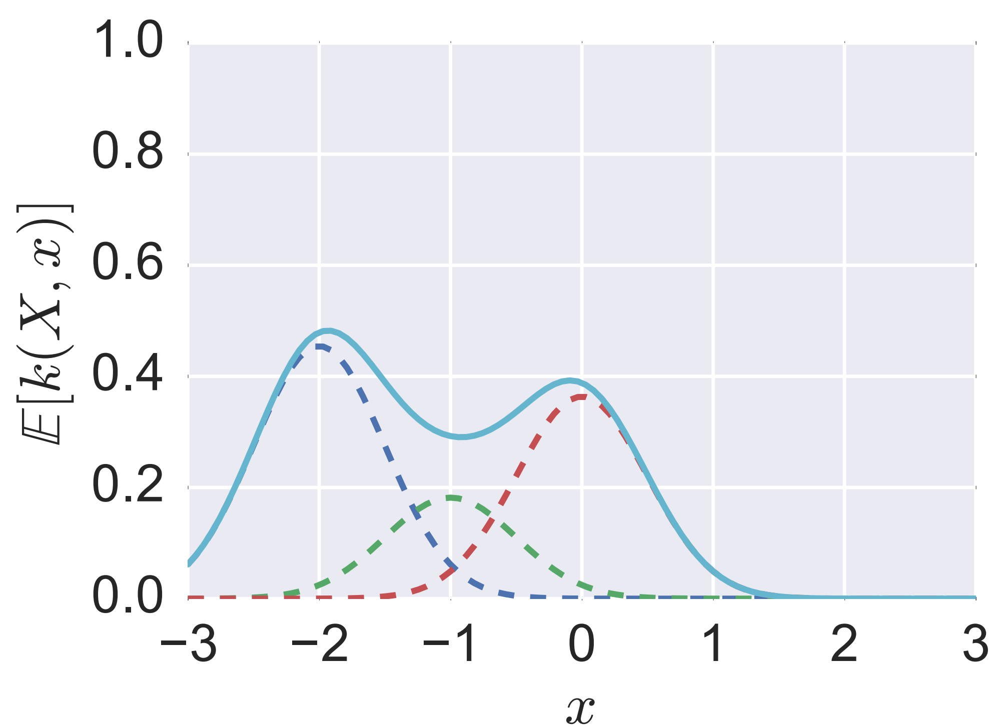
Mean embeddings of distributions

The *mean embedding* of a distribution in an RKHS:

$$\mu_{\mathbb{P}} = \mathbb{E}_{x \sim \mathbb{P}}[\varphi(x)]$$

Remember $\langle \varphi(x), \varphi(y) \rangle = k(x, y)$,
so we can think of $\varphi(x)$ as $k(x, \cdot)$.



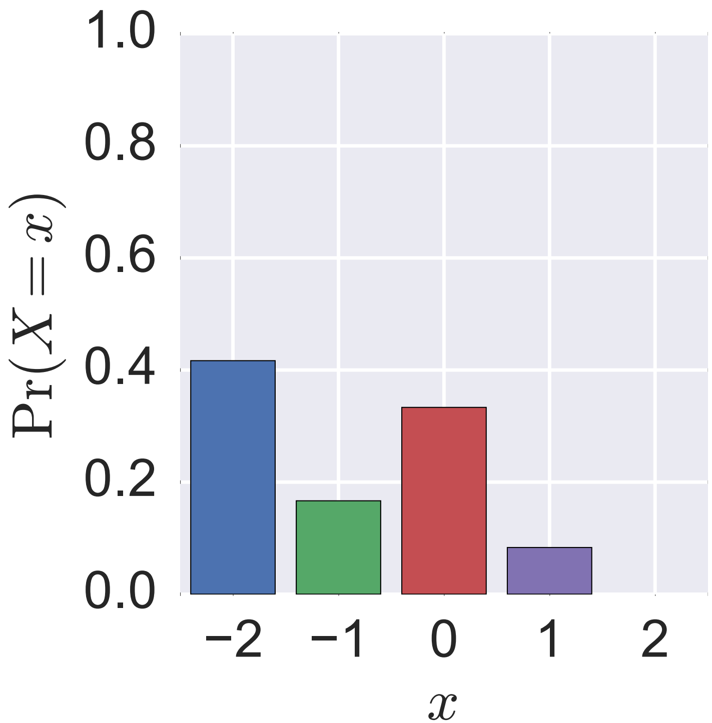


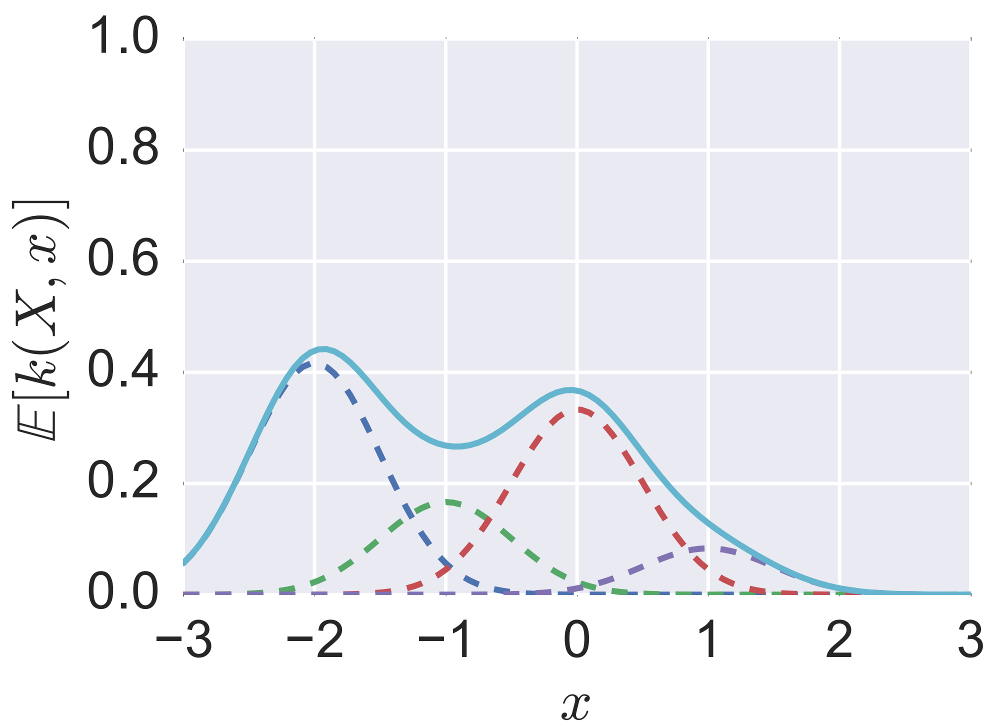
Mean embeddings of distributions

The *mean embedding* of a distribution in an RKHS:

$$\mu_{\mathbb{P}} = \mathbb{E}_{x \sim \mathbb{P}}[\varphi(x)]$$

Remember $\langle \varphi(x), \varphi(y) \rangle = k(x, y)$, so we can think of $\varphi(x)$ as $k(x, \cdot)$.



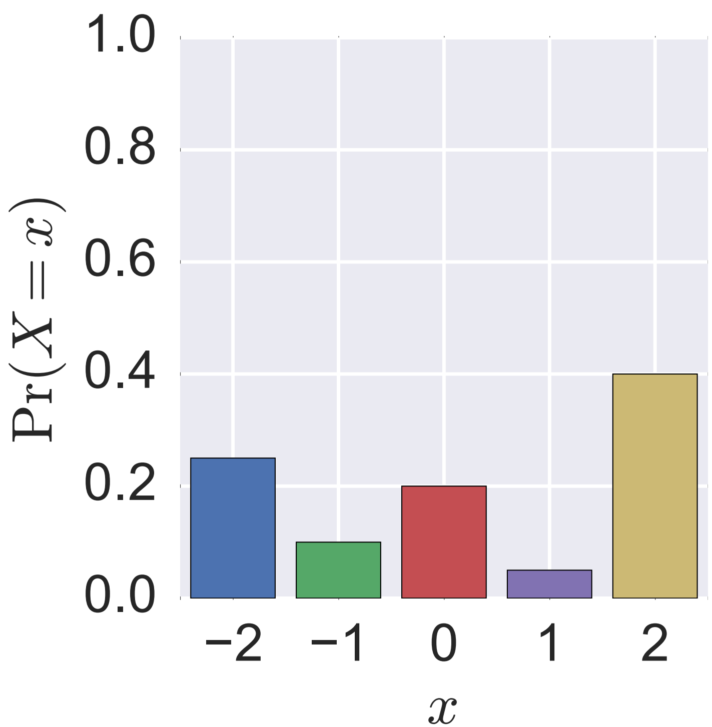


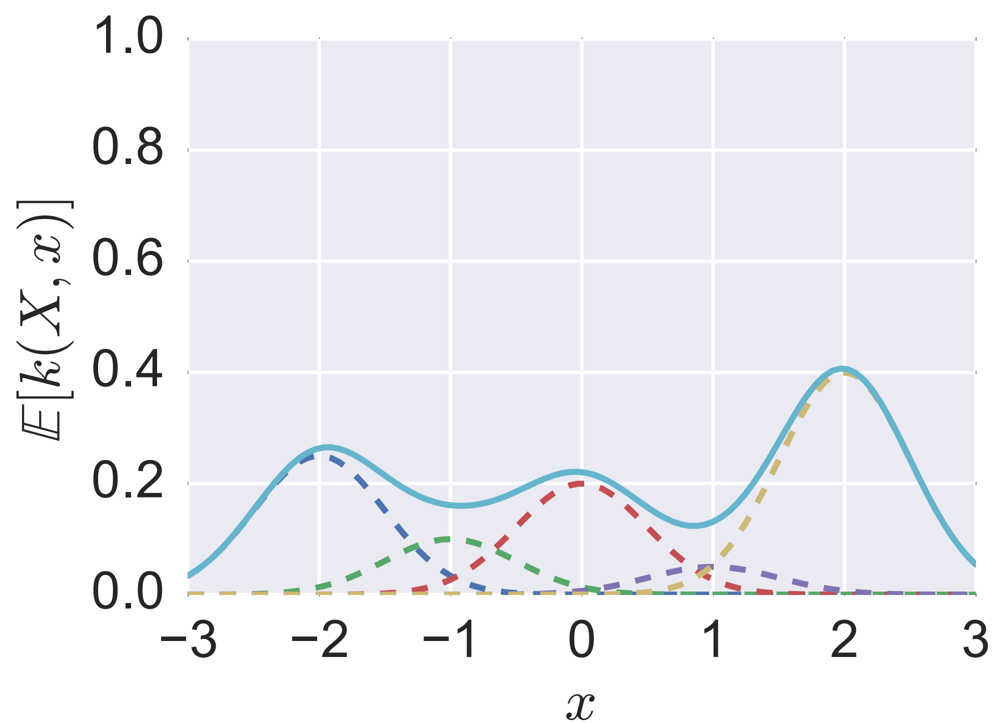
Mean embeddings of distributions

The *mean embedding* of a distribution in an RKHS:

$$\mu_{\mathbb{P}} = \mathbb{E}_{x \sim \mathbb{P}}[\varphi(x)]$$

Remember $\langle \varphi(x), \varphi(y) \rangle = k(x, y)$,
so we can think of $\varphi(x)$ as $k(\cdot, x)$.



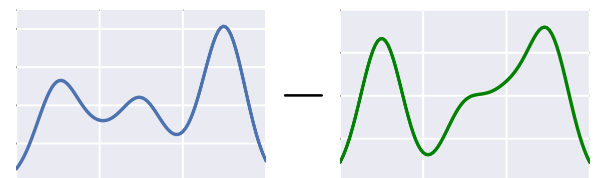


Maximum Mean Discrepancy (MMD)

The MMD is the distance between mean embeddings:

$$\mu_{\mathbb{P}} = \mathbb{E}_{X\sim\mathbb{P}}[\varphi(X)]$$

$$\mathrm{MMD}^2(\mathbb{P},\mathbb{Q}) = \|\mu_{\mathbb{P}} - \mu_{\mathbb{Q}}\|^2_{\mathcal{H}}$$

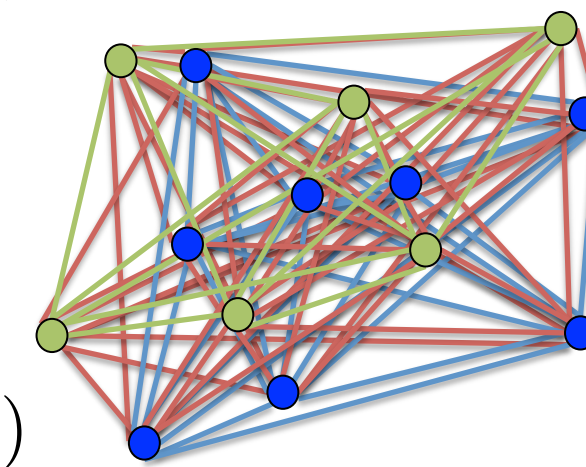
$\Bigg\|$  $\Bigg\|^2_{\mathcal{H}}$

$$= \langle \mu_{\mathbb{P}}, \mu_{\mathbb{P}} \rangle + \langle \mu_{\mathbb{Q}}, \mu_{\mathbb{Q}} \rangle - 2\langle \mu_{\mathbb{P}}, \mu_{\mathbb{Q}} \rangle$$

$$\langle \mu_{\mathbb{P}}, \mu_{\mathbb{Q}} \rangle = \langle \mathbb{E}_{X\sim P}[\varphi(X)], \mathbb{E}_{Y\sim Q}[\varphi(Y)] \rangle$$

$$= \mathbb{E}_{\substack{X\sim P\\ Y\sim Q}}\left[\langle \varphi(X), \varphi(Y) \rangle\right]$$

$$= \mathbb{E}_{\substack{X\sim P\\ Y\sim Q}}\left[k(X,Y)\right]$$

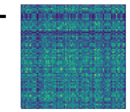


Estimator: $\langle \hat{\mu}_{\mathbb{P}}, \hat{\mu}_{\mathbb{Q}} \rangle = \frac{1}{mn}\sum_{ij} k(X_i, Y_j)$

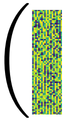
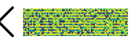
Embedding MMD

$$\widehat{\mathrm{MMD}}(X, Y) = \|\hat{\mu}_{\mathbb{P}} - \hat{\mu}_{\mathbb{Q}}\|_{\mathcal{H}}$$

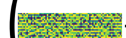
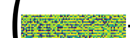
$$\langle \hat{\mu}_{\mathbb{P}}, \hat{\mu}_{\mathbb{Q}} \rangle = \frac{1}{mn} \sum_{ij} k(X_i, Y_j)$$

$= \frac{1}{m} \mathbb{1}^{\mathsf{T}}$  $\frac{1}{n} \mathbb{1}$

$$\approx \frac{1}{mn} \sum_{ij} z(X_i)^{\mathsf{T}} z(Y_j)$$

$= \frac{1}{m} \mathbb{1}^{\mathsf{T}} \Big($  $\times$  $\Big) \frac{1}{n} \mathbb{1}$

$$= \left[\frac{1}{m} \sum_{i} z(X_i)\right]^{\mathsf{T}} \left[\frac{1}{n} \sum_{j} z(Y_j)\right]$$

$= \Big($  $\frac{1}{m} \mathbb{1}\Big)^{\mathsf{T}} \Big($  $\frac{1}{n} \mathbb{1}\Big)$

$$= \bar{z}(X)^{\mathsf{T}} \bar{z}(Y)$$

$=$  $^{\mathsf{T}}$ 

$$\widehat{\mathrm{MMD}}_z(X, Y) = \|\bar{z}(X) - \bar{z}(Y)\|$$

$= \Big\|$  $-$  $\Big\|$

$$\exp\left(-\frac{1}{2\sigma^2} \widehat{\mathrm{MMD}}_z(X, Y)\right) \approx z(\bar{z}(X))^{\mathsf{T}} z(\bar{z}(Y))$$

Embedding L_2

Oliva, Neiswanger, Póczos, Schneider, Xing (AISTATS 2014) gave an embedding for

$$K(\hat{p}, \hat{q}) = \exp\left(-\frac{1}{2\sigma^2}\|\hat{p} - \hat{q}\|_{L_2}^2\right)$$

based on projection coefficients onto an orthonormal basis for L_2.

Embedding HDDS

$$\rho^2(p,q) = \int_{\mathcal{X}} \kappa(p(x), q(x))\, \mathrm{d}x$$

Homogeneous Density Distances

ρ^2 can be:

- Jensen-Shannon
- Total Variation
- Squared Hellinger

Algorithm: HDD Embedding

$$\rho^2(p,q) = \int_{\mathcal{X}} \kappa(p(x), q(x))\, \mathrm{d}x$$ Homogeneous Density Distances

1. Approximately embed ρ into L_2
   - by embedding $\varkappa$ into L_2 (Fuglede 2005).

$$\rho(p,q) \approx \|\psi(p) - \psi(q)\|_{L_2^{2M}}$$

ρ on distributions ≈ L_2 distance on random ψ functions

2. Approximately embed L_2 into $\mathbf{R}^m$.

$$\|\psi(p) - \psi(q)\|_{L_2^{2M}} \approx \|A(p) - A(q)\|_{\mathbb{R}^{2M|V|}}$$

L_2 distance on random ψ functions ≈ Euclidean distance on A vectors

3. Use random Fourier features to embed K on $\mathbf{R}^m$ into $\mathbf{R}^D$.

$$\exp\left(-\frac{1}{2\sigma^2}\rho^2(p,q)\right) \approx z(A(p))^\mathsf{T} z(A(q))$$

RBF kernel with ρ ≈ random Fourier features of A vectors

HDD estimator convergence

The embedding is an estimator for the kernel.

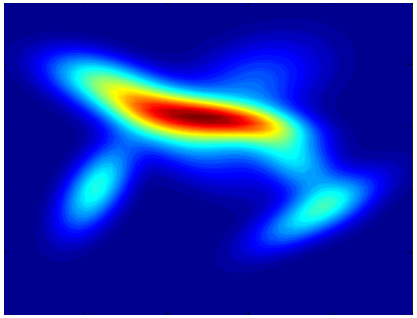
For fixed *p* and *q*, we have a finite-sample bound on

$$\Pr\left(\left|K(p,q) - z(\hat{A}(\hat{p}))^\mathsf{T} z(\hat{A}(\hat{q}))\right| \geq \varepsilon\right)$$

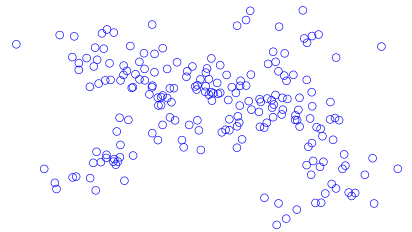
which behaves as expected:

- Lower for smoother, lower-dimensional densities
- Lower for more samples
- More projection coefficients / samples from μ:
  - better approximation, harder integration

Application: Gaussian Mixtures



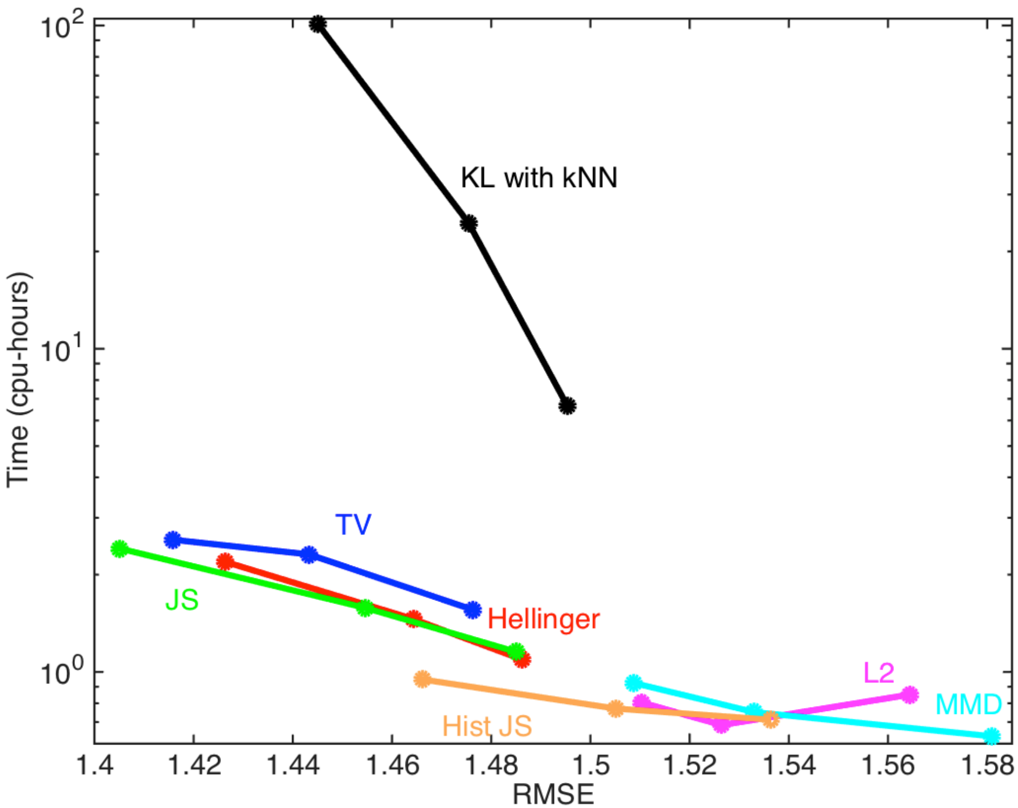
Sample



Regression

9

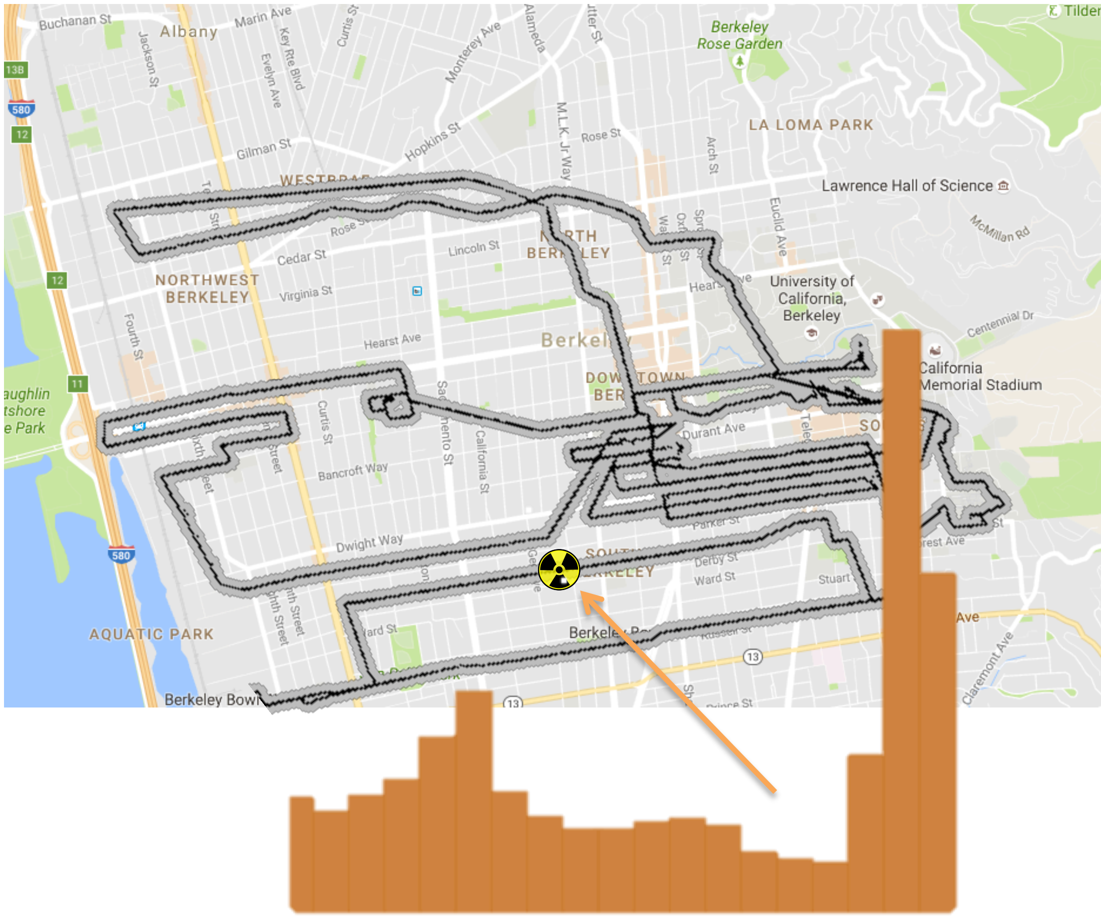
Underlying density | Observed sample | Prediction

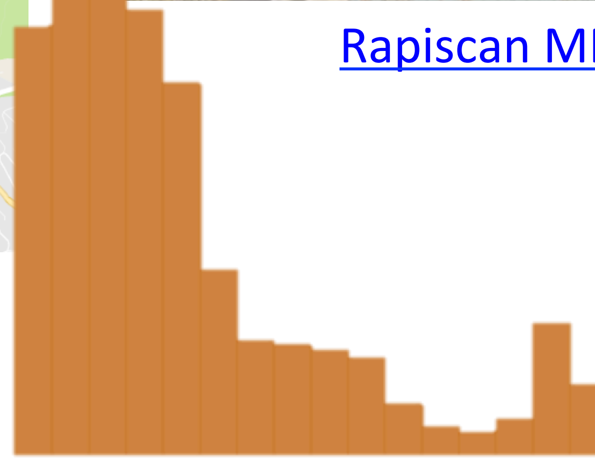


Train with 4K, 8K, 16K distributions; test on 2K. 200 iid points each.

AIC: 2.7
BIC: 3.8
Mean: 2.8

Application: Nuclear Threat Detection

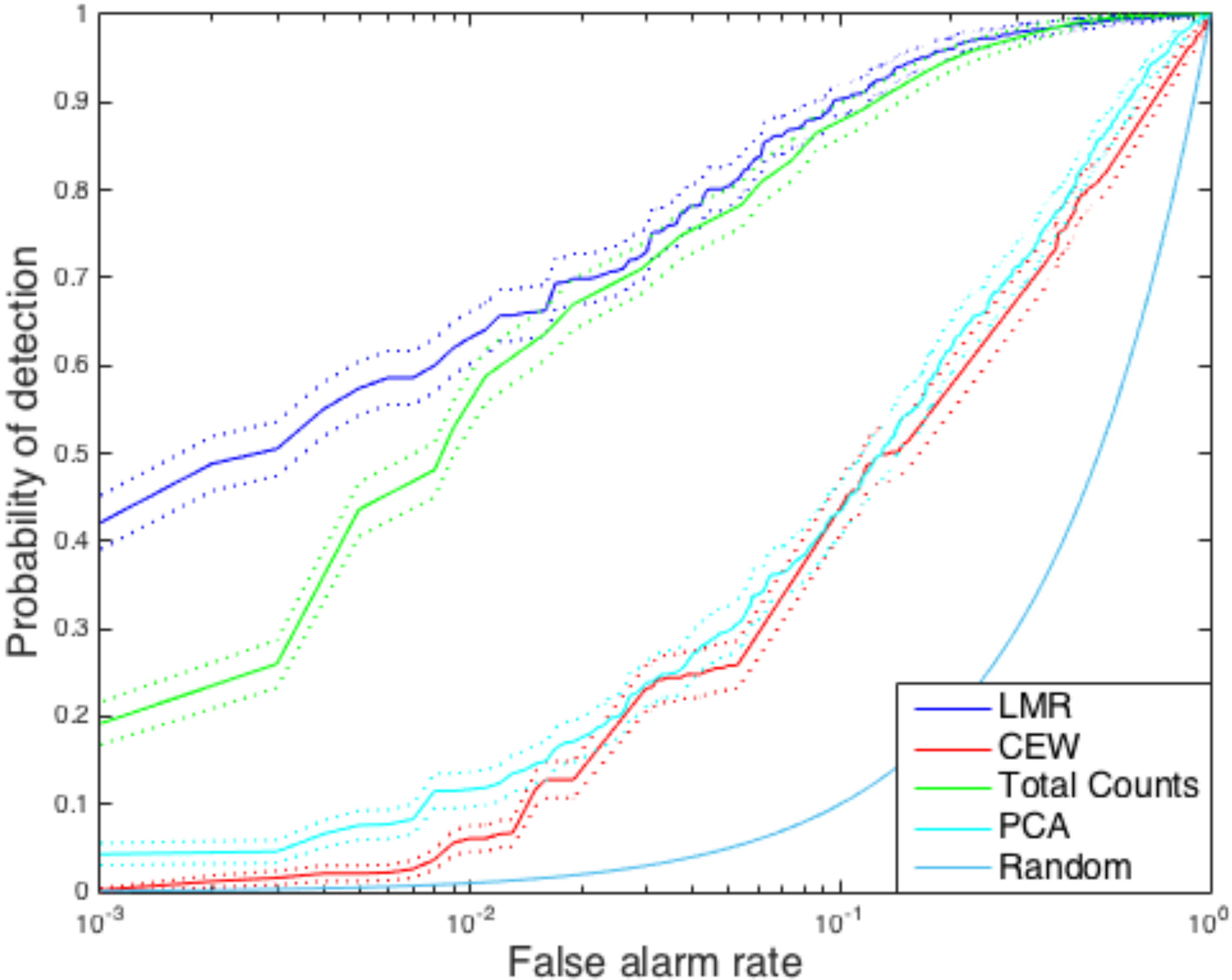




Rapiscan MP100

Small sensors and cluttered environments make lots of challenges for traditional detection algorithms.

Application: Nuclear Threat Detection

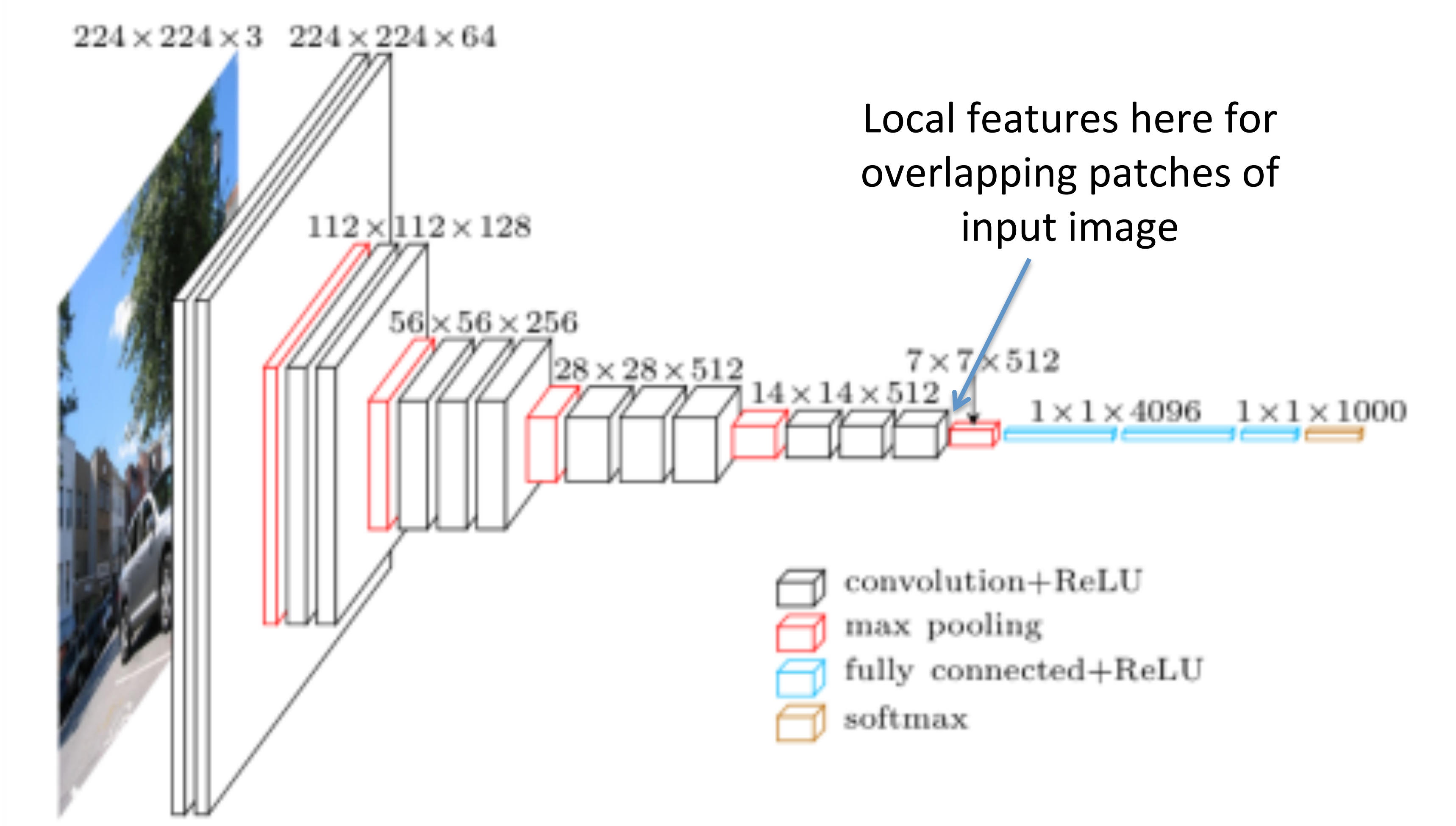


Application: Scene Classification

Scene-15 dataset (4 485 images)

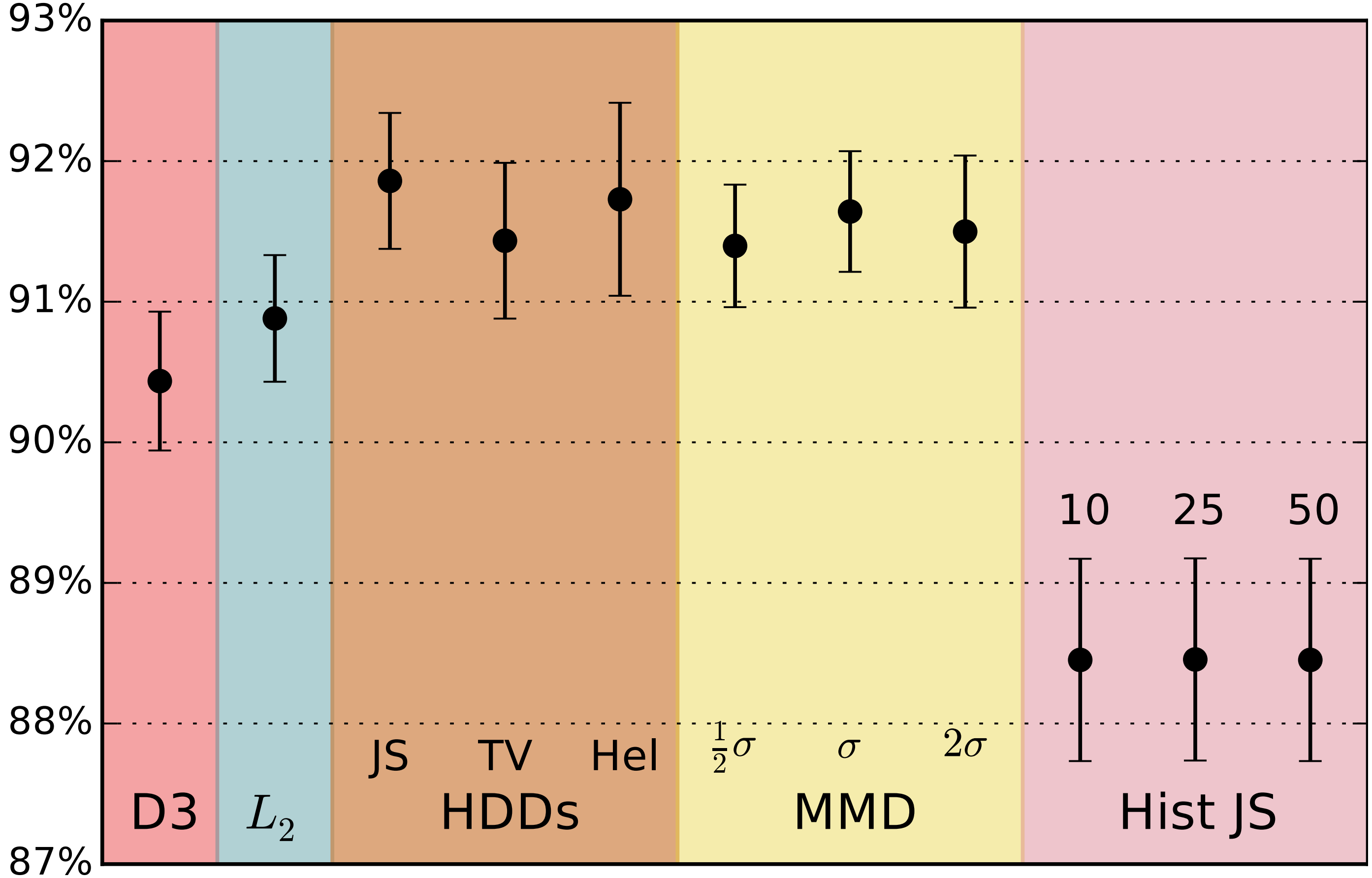
Application: Scene Classification

Features from a deep convnet (16 layers):



Davi Frossard's [blog]

Application: Scene Classification



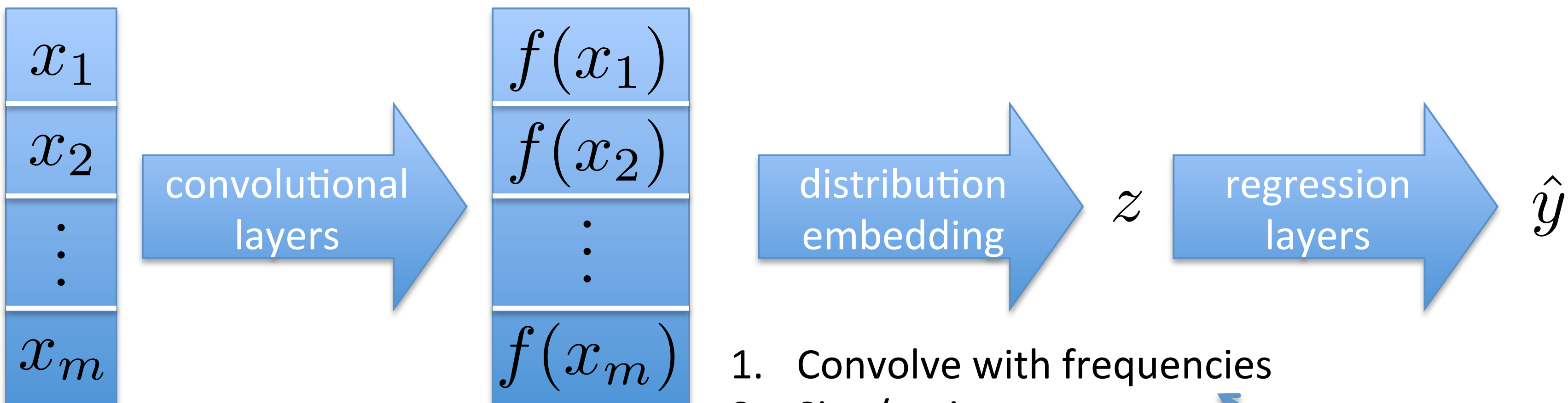
Using $A(p)$ features

Contributions

- **Learning on distributions** with nonparametric kernels
- **Scalable** approximate kernel embeddings
  - Random Fourier features analysis
  - New embeddings for distribution kernels
- **Flexible** distribution kernels
  - Deep mean maps in computer vision
  - MMD kernel learning for testing
- **Active** pointillistic pattern search

Deep distribution kernel learning

We can put distribution embeddings in a deep network:



1. Convolve with frequencies
2. Sine/cosines
3. Global pooling

Fixed frequencies
Learn scale
Learn frequencies

Initial results for scene classification: small but consistent improvement.

Contributions

- **Learning on distributions** with nonparametric kernels
- **Scalable** approximate kernel embeddings
  - Random Fourier features analysis
  - New embeddings for distribution kernels
- **Flexible** distribution kernels
  - Deep mean maps in computer vision
  - MMD kernel learning for testing
- **Active** pointillistic pattern search

Two-sample testing

Observed samples: $X \sim \mathbb{P}$ $Y \sim \mathbb{Q}$

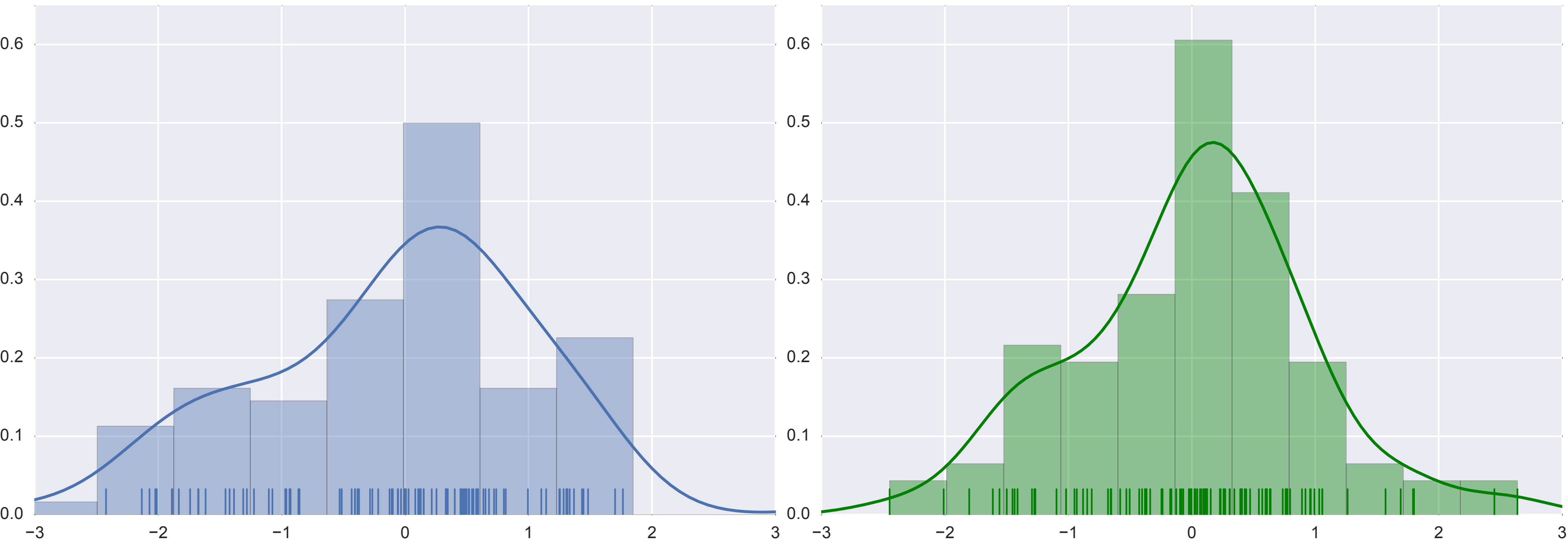
Hypothesis test on unobserved distributions: $\mathbb{P} \stackrel{?}{=} \mathbb{Q}$

Applications:

- Neuroscience: do these areas of the brain behave differently under different conditions?
- Schema alignment: do these two database columns mean the same thing?
- Many more…

Two-sample testing

Test with MMD:



$X \sim \mathbb{P} = \mathcal{N}(0,1)$ $\quad$ $Y \sim \mathbb{Q} = \text{Laplace}\left(0, \frac{1}{\sqrt{2}}\right)$

Test: reject if $m\,\widehat{\text{MMD}}^2(\mathbb{P},\mathbb{Q}) > c_\alpha$.

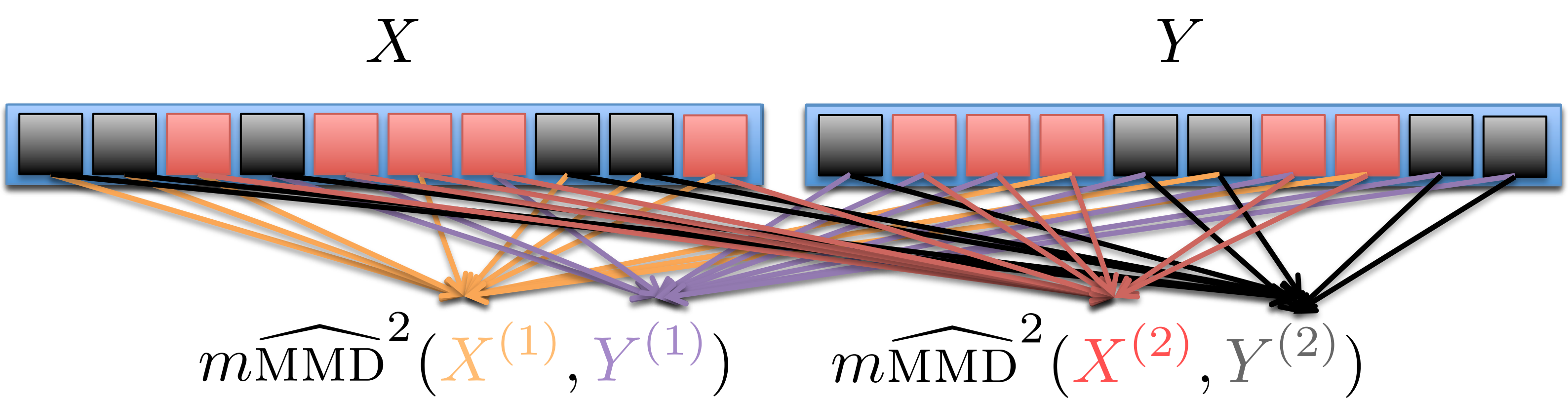
Permutation testing

When $\mathbb{P} = \mathbb{Q}$, MMD is asymptotically gross, so hard to find a threshold c_α that way.

Use a permutation test:

$$X = \{x_1, x_2, \ldots, x_m\} \qquad Y = \{y_1, y_2, \ldots, y_m\}$$

Split randomly to estimate MMD when $\mathbb{P} = \mathbb{Q}$:



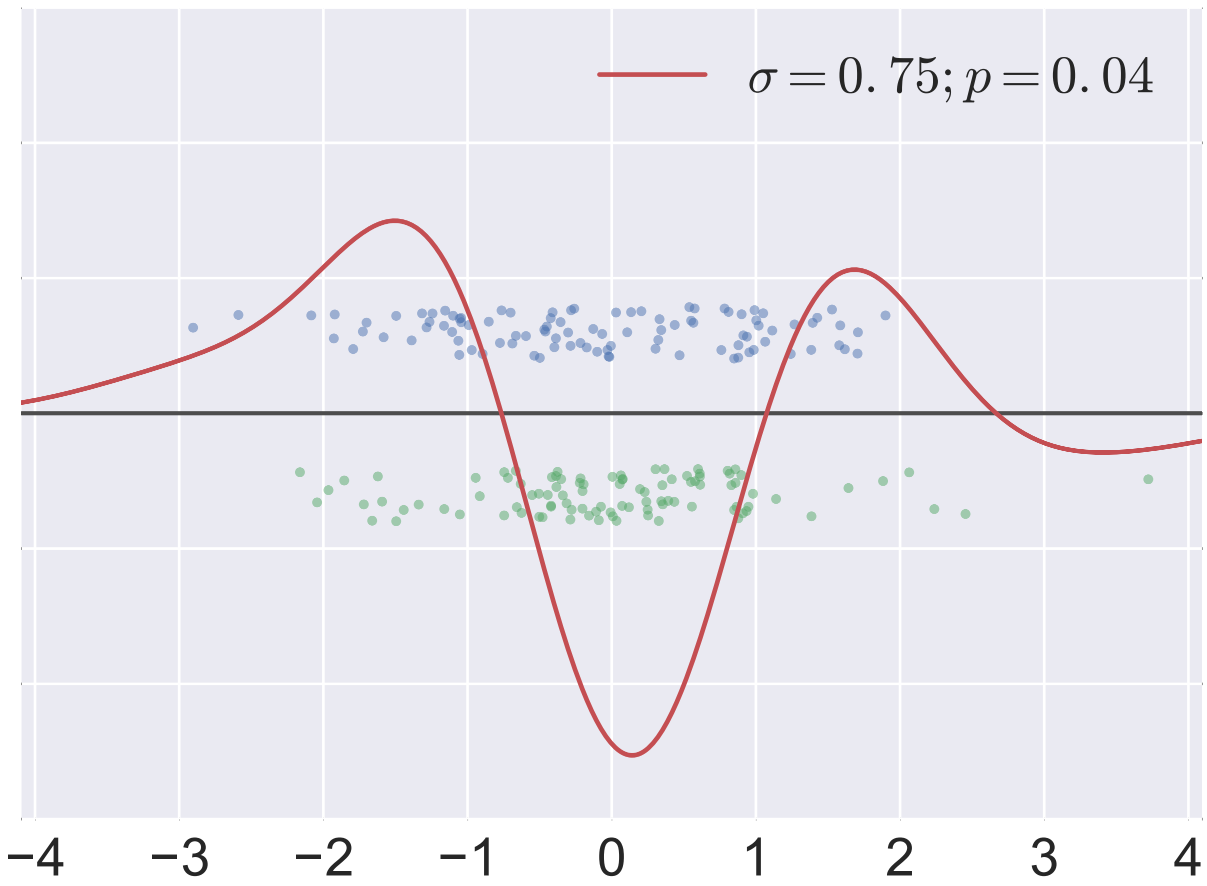
$\hat{c}_\alpha$: (1-α)th quantile of $m\widehat{\text{MMD}}^2(X^{(i)}, Y^{(i)})$

The kernel matters!

Witness function f helps compare samples:

$$\text{MMD}(\mathbb{P}, \mathbb{Q}) = \mathbb{E}_{X \sim \mathbb{P}}\, f(X) - \mathbb{E}_{Y \sim \mathbb{Q}}\, f(Y)$$

$$f(x) = \mathbb{E}_{X \sim \mathbb{P}}\, k(x, X) - \mathbb{E}_{Y \sim \mathbb{Q}}\, k(x, Y)$$

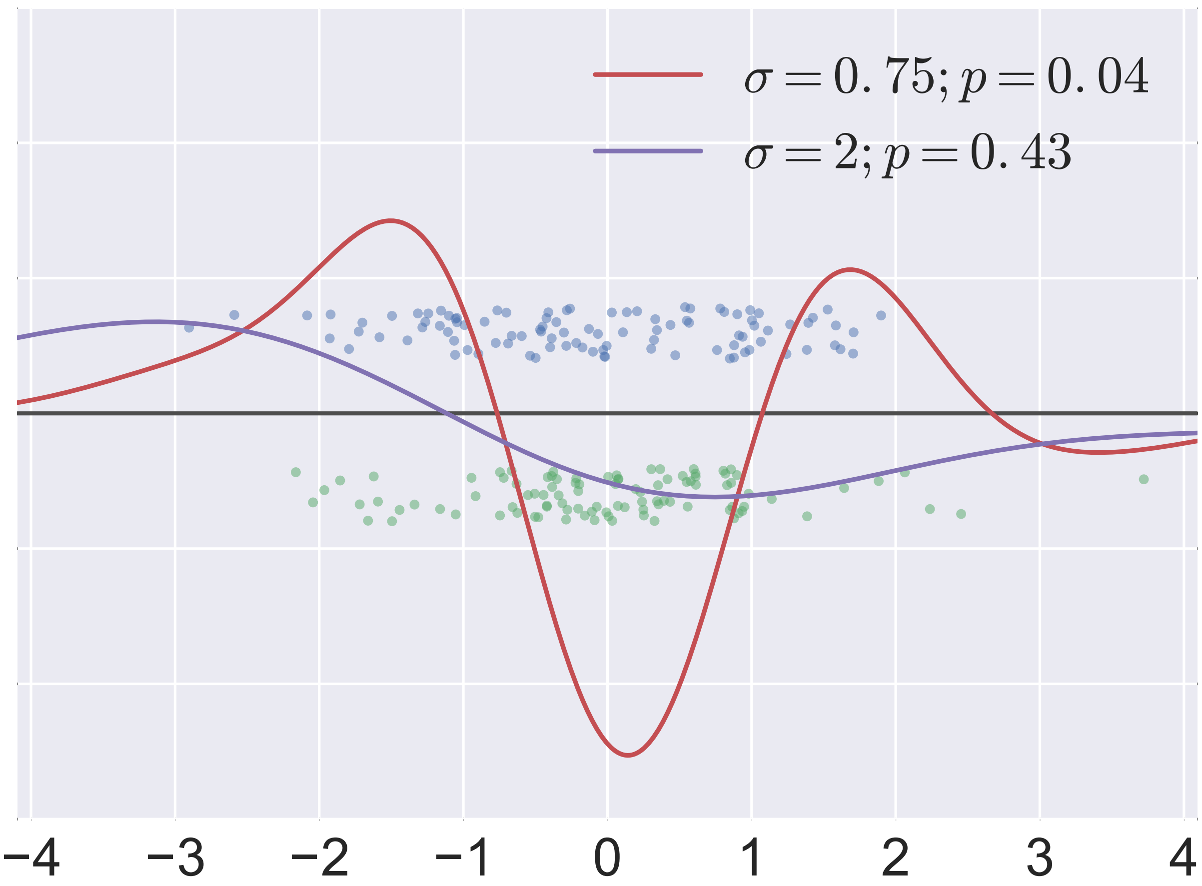


The kernel matters!

Witness function f helps compare samples:

$$\mathrm{MMD}(\mathbb{P}, \mathbb{Q}) = \mathbb{E}_{X\sim\mathbb{P}}\, f(X) - \mathbb{E}_{Y\sim\mathbb{Q}}\, f(Y)$$

$$f(x) = \mathbb{E}_{X\sim\mathbb{P}}\, k(x, X) - \mathbb{E}_{Y\sim\mathbb{Q}}\, k(x, Y)$$

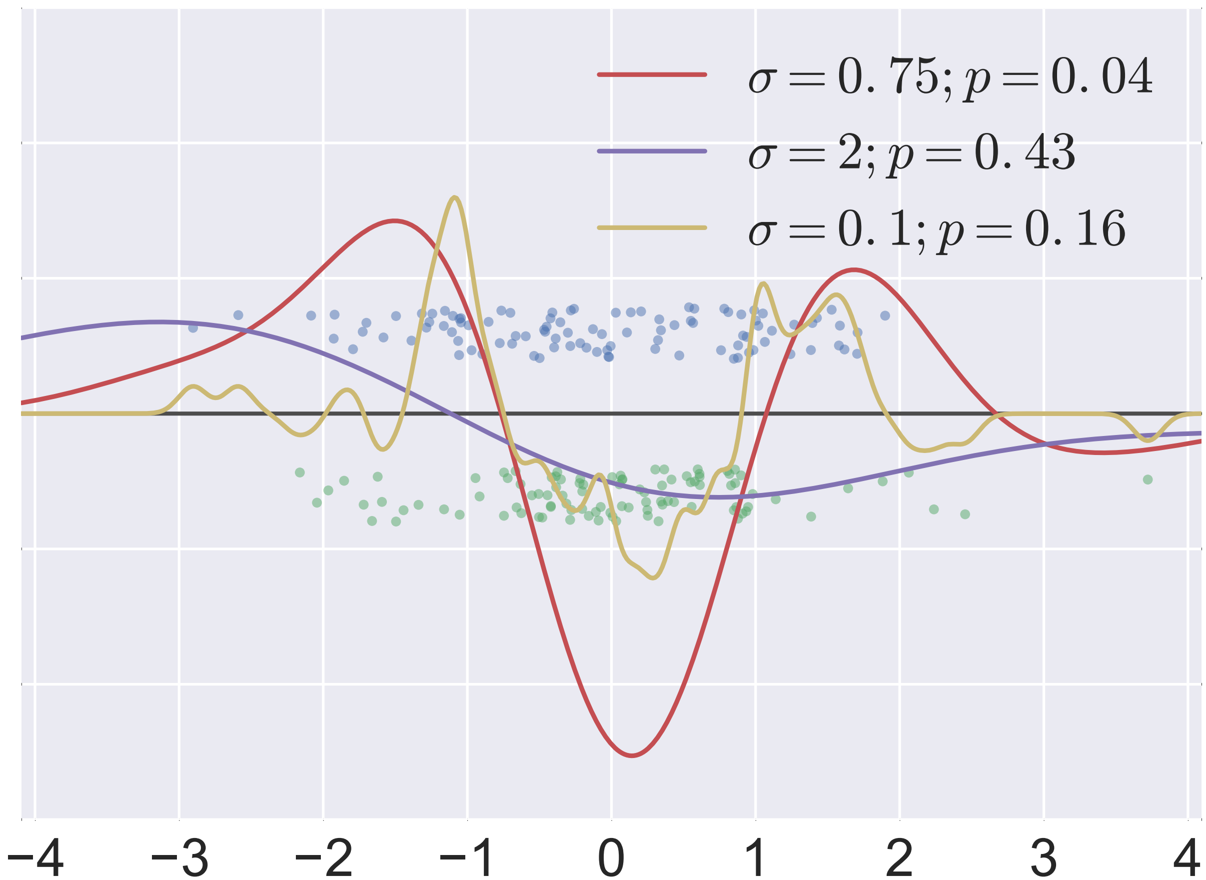


The kernel matters!

Witness function f helps compare samples:

$$\mathrm{MMD}(\mathbb{P}, \mathbb{Q}) = \mathbb{E}_{X \sim \mathbb{P}} f(X) - \mathbb{E}_{Y \sim \mathbb{Q}} f(Y)$$

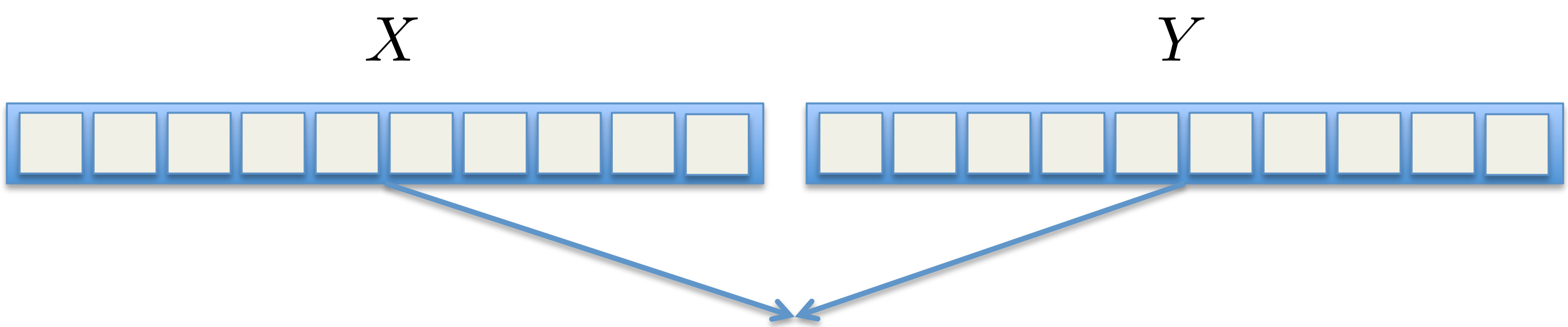
$$f(x) = \mathbb{E}_{X \sim \mathbb{P}} k(x, X) - \mathbb{E}_{Y \sim \mathbb{Q}} k(x, Y)$$



Choosing a kernel

So we need a way to pick a kernel to do the test.

Before:

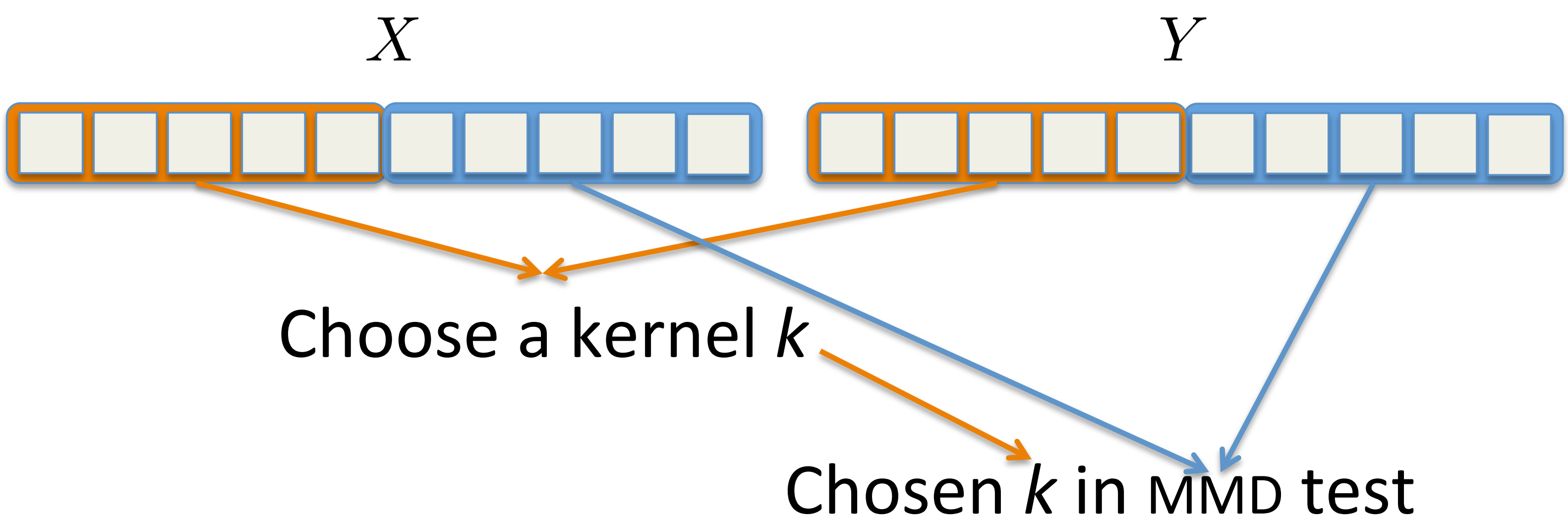


MMD with fixed k

Choosing a kernel

So we need a way to pick a kernel to do the test.

Split data:



How to pick k? Typically: maximize MMD.

But we want the (asymptotically) *most powerful test*.

Asymptotic power of MMD

When $\mathbb{P} \neq \mathbb{Q}$, MMD is asymptotically normal:

$$\frac{\widehat{\mathrm{MMD}^2} - \mathrm{MMD}^2}{\sqrt{V_m}} \xrightarrow{D} \mathcal{N}(0,1) \qquad V_m = \mathrm{Var}_{\substack{X \sim P^m \\ Y \sim Q^m}} \left[\widehat{\mathrm{MMD}^2}(X,Y) \right]$$

and we can analyze the power:

$$\mathrm{Pr}_{H_1} \left(m \widehat{\mathrm{MMD}}^2 > \hat{c}_\alpha \right)$$

MMD *t*-statistic

$$\Pr_{H_1}\left(m\widehat{\mathrm{MMD}}^2 > \hat{c}_\alpha\right) \to 1 - \Phi\left(\frac{c_\alpha}{m\sqrt{V_m}} - \frac{\mathrm{MMD}^2}{\sqrt{V_m}}\right)$$

So we can maximize the power by maximizing

$$\tau_U = \frac{\mathrm{MMD}^2}{\sqrt{V_m}} - \frac{c_\alpha}{m\sqrt{V_m}} \qquad \hat{\tau}_U = \frac{\widehat{\mathrm{MMD}^2}}{\sqrt{\widehat{V}_m}} - \frac{\hat{c}_\alpha}{m\sqrt{\widehat{V}_m}}$$

But V_m is $O(1/m)$, so the first term dominates for large m, and we should be able to get away with maximizing

$$t_U = \frac{\mathrm{MMD}^2}{\sqrt{V_m}} \qquad \hat{t}_U = \frac{\widehat{\mathrm{MMD}^2}}{\sqrt{\widehat{V}_m}}$$

t-statistic estimator

$$\hat{\tau}_U = \frac{\widehat{\mathrm{MMD}^2}}{\sqrt{\widehat{V}_m}} - \frac{\hat{c}_\alpha}{m\sqrt{\widehat{V}_m}}$$

$$\widehat{\mathrm{MMD}^2} := \frac{1}{\binom{m}{2}} \sum_{i \neq j} k(X_i, X_j) + k(Y_i, Y_j) - k(X_i, Y_j) - k(X_j, Y_i)$$

$\hat{c}_\alpha$ is from a permutation test, so the average of a bunch of MMD estimates

t-statistic estimator

$$\widehat{V}_m = \frac{4(m-2)}{m(m-1)}\hat{\zeta}_1 + \frac{2}{m(m-1)}\hat{\zeta}_2$$

$$\begin{aligned}
\hat{\zeta}_1 = & \frac{1}{m(m-1)(m-2)}\left(\mathbf{1}^\intercal \tilde{K}_{XX}\tilde{K}_{XX}\mathbf{1} - \|\tilde{K}_{XX}\|_F^2\right) - \left(\frac{1}{m(m-1)}\mathbf{1}^\intercal \tilde{K}_{XX}\mathbf{1}\right)^2 \\
& - \frac{2}{m^2(m-1)}\mathbf{1}^\intercal \tilde{K}_{XX}K_{XY}\mathbf{1} + \frac{2}{m^3(m-1)}\mathbf{1}^\intercal \tilde{K}_{XX}\mathbf{1}\mathbf{1}^\intercal K_{XY}\mathbf{1} \\
& + \frac{1}{m(m-1)(m-2)}\left(\mathbf{1}^\intercal \tilde{K}_{YY}\tilde{K}_{YY}\mathbf{1} - \|\tilde{K}_{YY}\|_F^2\right) - \left(\frac{1}{m(m-1)}\mathbf{1}^\intercal \tilde{K}_{YY}\mathbf{1}\right)^2 \\
& - \frac{2}{m^2(m-1)}\mathbf{1}^\intercal \tilde{K}_{YY}K_{XY}^\intercal\mathbf{1} + \frac{2}{m^3(m-1)}\mathbf{1}^\intercal \tilde{K}_{YY}\mathbf{1}\mathbf{1}^\intercal K_{XY}\mathbf{1} \\
& + \frac{1}{m^2(m-1)}\left(\mathbf{1}^\intercal K_{XY}^\intercal K_{XY}\mathbf{1} - \|K_{XY}\|_F^2\right) - 2\left(\frac{1}{m^2}\mathbf{1}^\intercal K_{XY}\mathbf{1}\right)^2 \\
& + \frac{1}{m^2(m-1)}\left(\mathbf{1}^\intercal K_{XY}K_{XY}^T\mathbf{1} - \|K_{XY}\|_F^2\right)
\end{aligned}$$

$$\hat{\zeta}_2 = \frac{1}{m(m-1)}\left\|\tilde{K}_{XX} + \tilde{K}_{YY} - \tilde{K}_{XY} - \tilde{K}_{XY}^\intercal\right\|_F^2$$

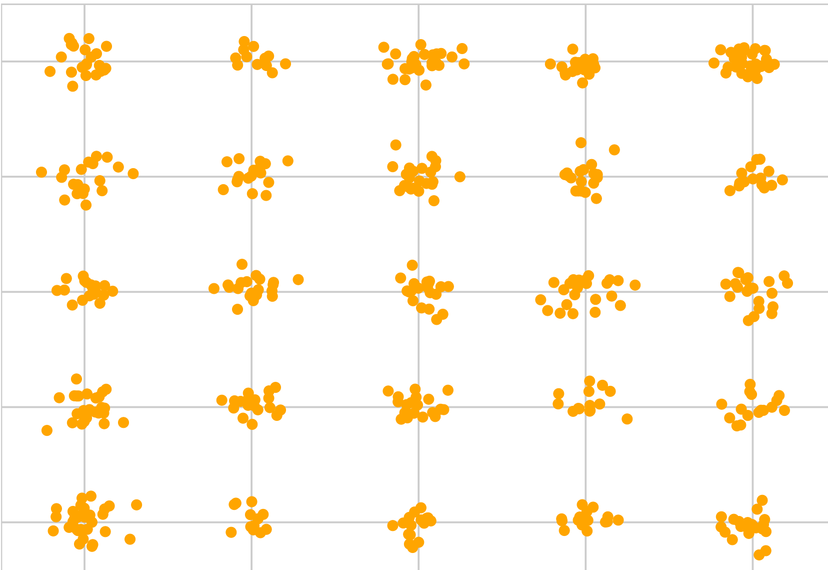
t-statistic estimator

Can even get gradients of t_U and (with some more effort) τ_U, to help maximize it.

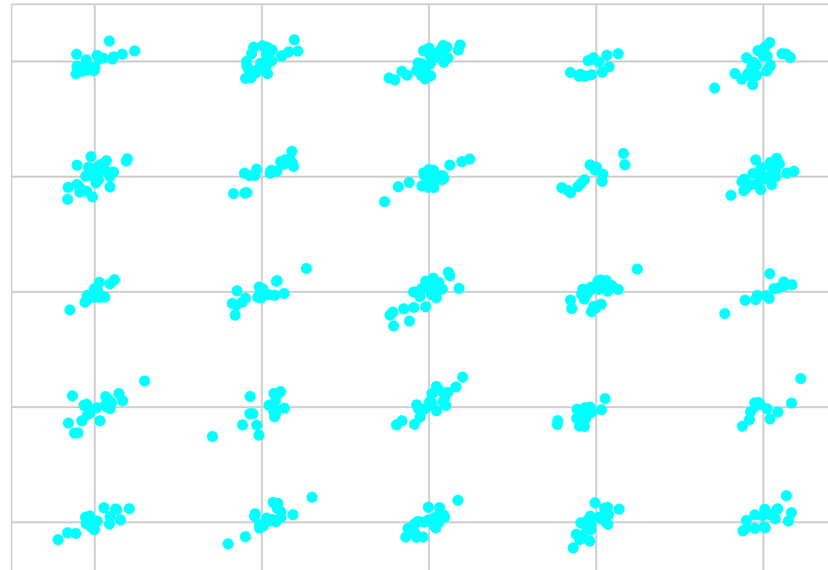
(automatic differentiation is your friend)

Kernel choice on Blobs

Blobs dataset:



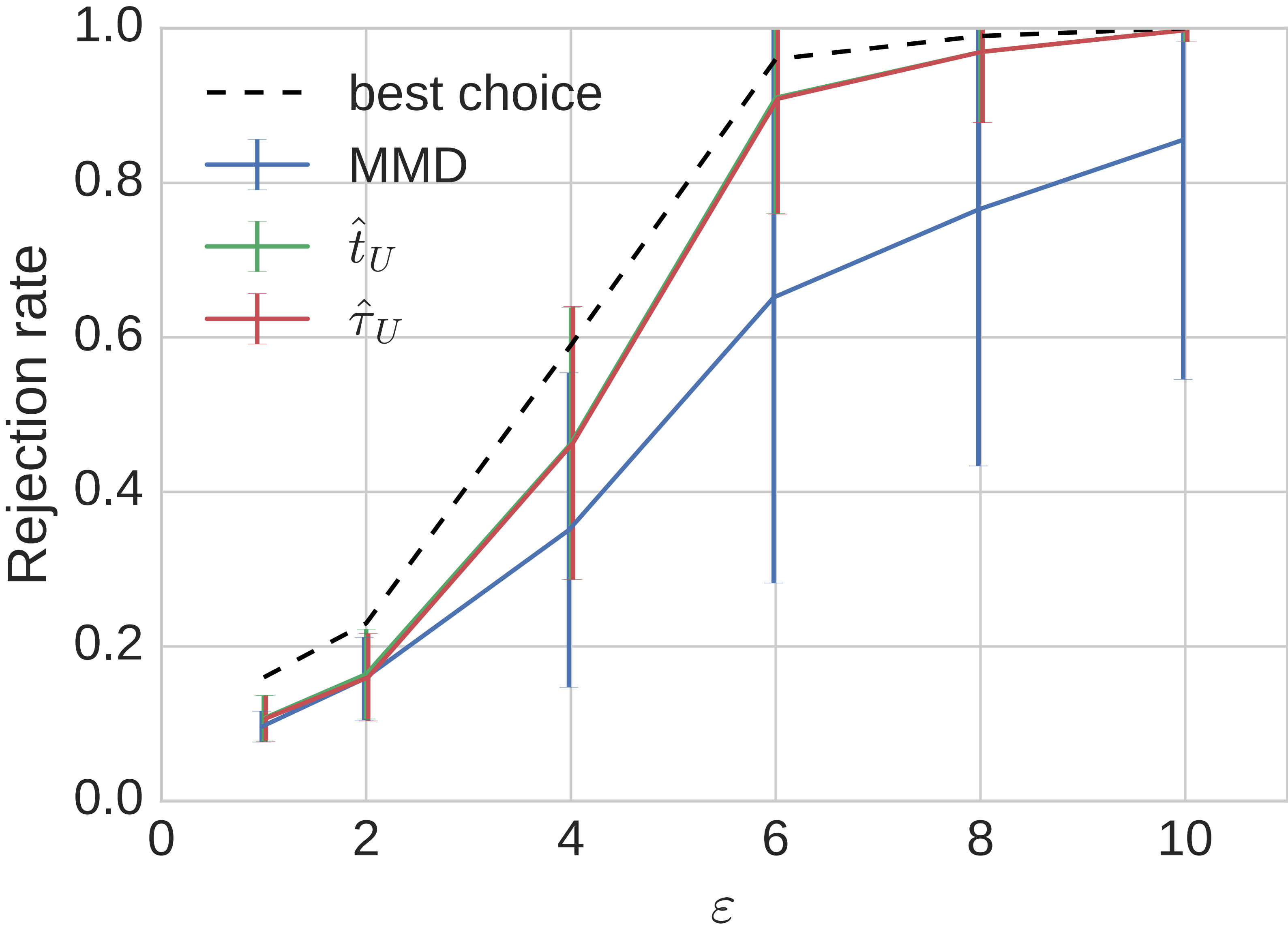
vs



Mixture of $\mathcal{N}\left(\mu_{ij}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$ Mixture of $\mathcal{N}\left(\mu_{ij}, \begin{bmatrix} 1 & \frac{\varepsilon-1}{\varepsilon+1} \\ \frac{\varepsilon-1}{\varepsilon+1} & 1 \end{bmatrix}\right)$

When ε=1, $P = Q$; this picture has ε=6.

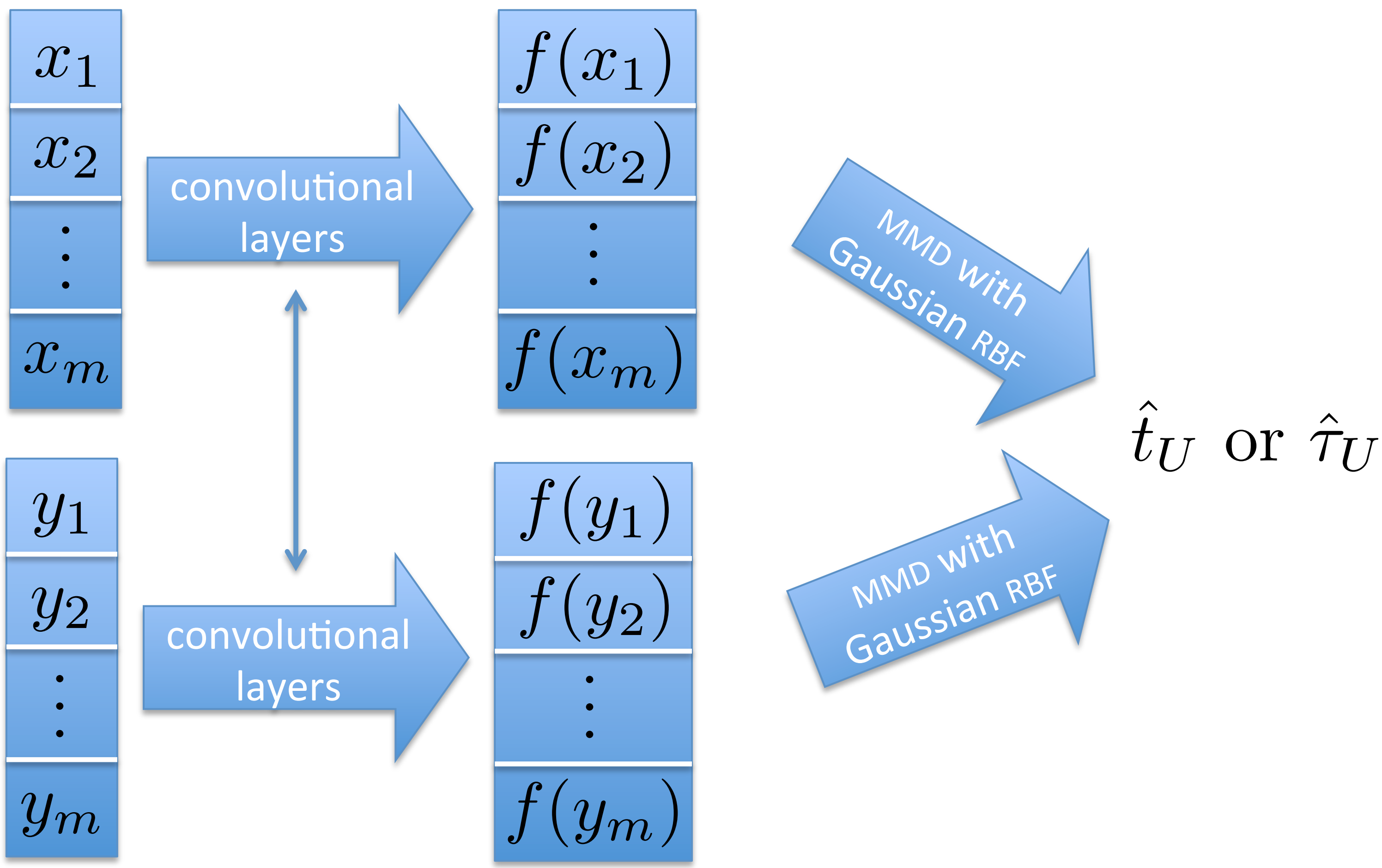
Kernel choice on Blobs



m = 500

Deep Kernels

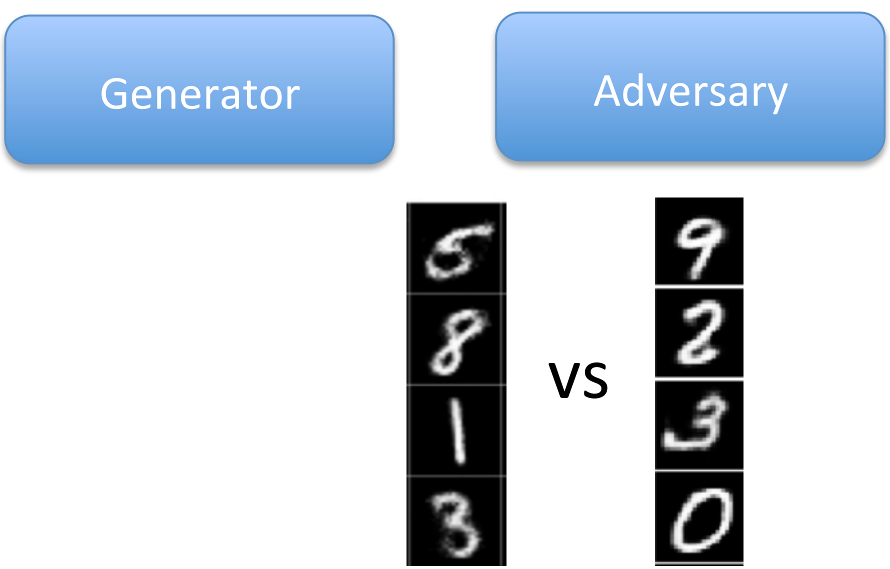
Map through layers of a deep network:



Generative Models

Generative adversarial networks:

- *Generator* comes up with samples; trained to trick the adversary.
- *Adversary* tries to distinguish between generator sample and true data; trained to beat the generator.



Generated Real

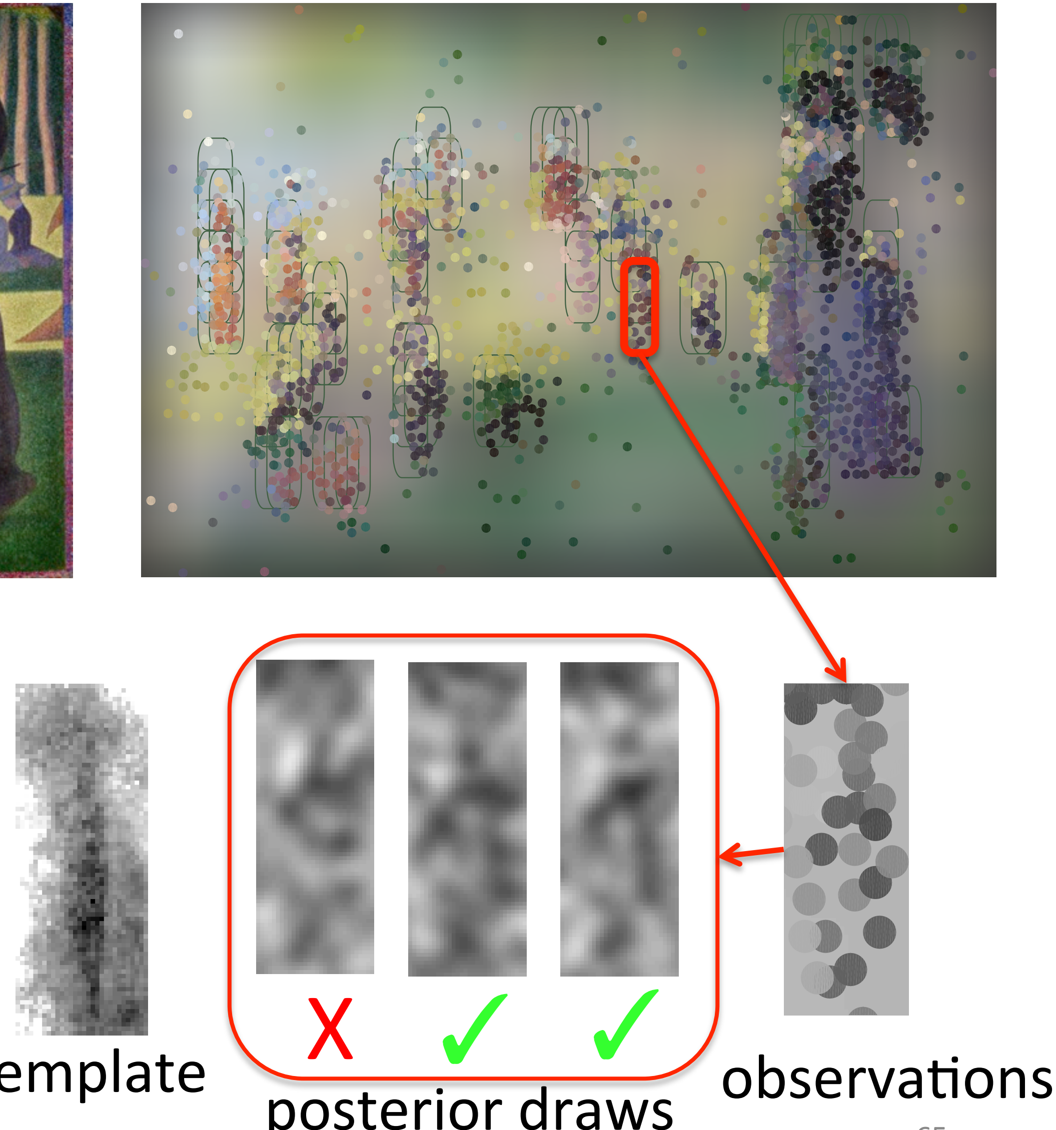
But adversary is really just a two-sample test.

Kernel learning helps prevent the generator's tricks.

Contributions

- **Learning on distributions** with nonparametric kernels
- **Scalable** approximate kernel embeddings
  - Random Fourier features analysis
  - New embeddings for distribution kernels
- **Flexible** distribution kernels
  - Deep mean maps in computer vision
  - MMD kernel learning for testing
- **Active** pointillistic pattern search

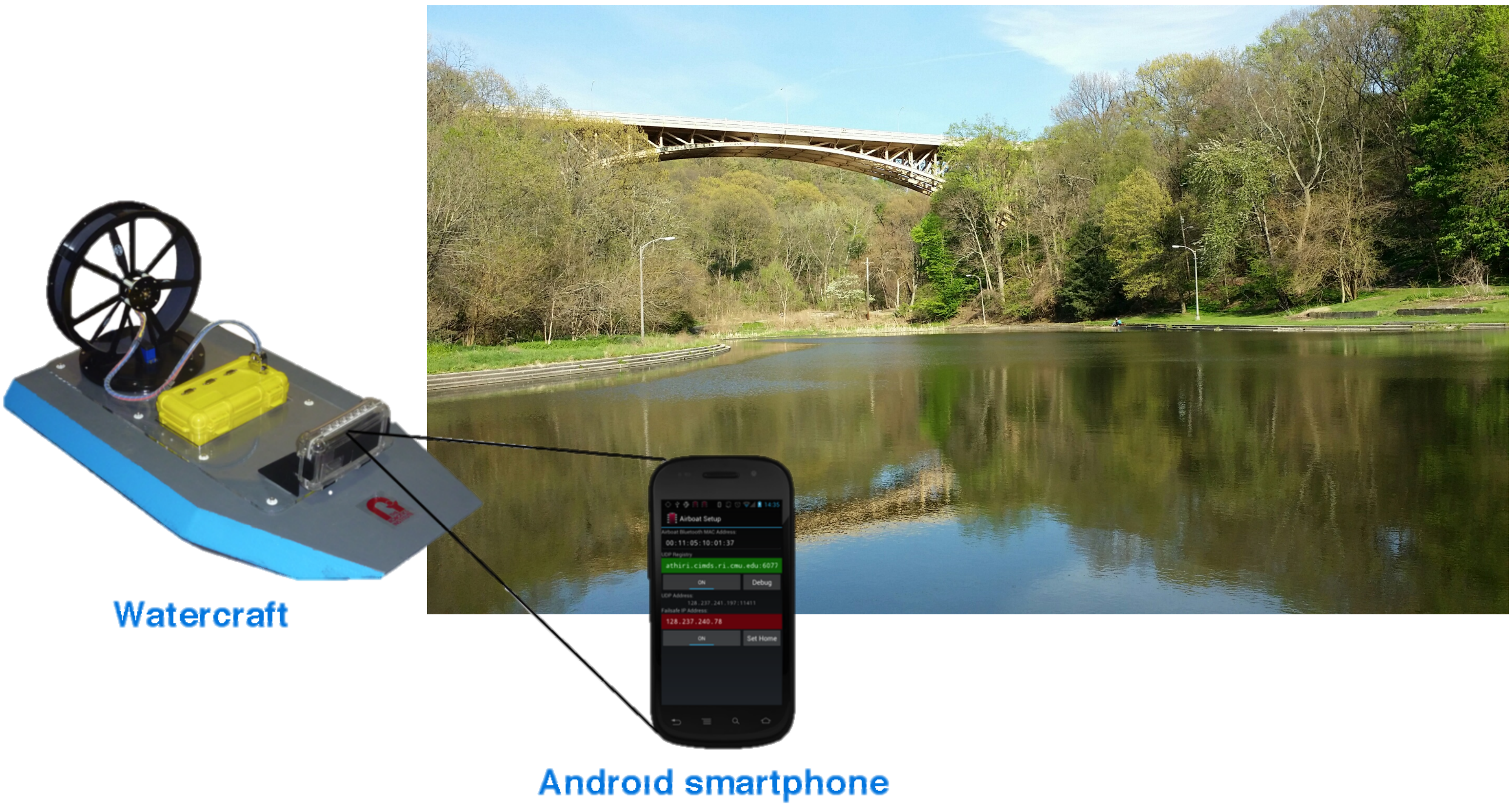
Active Pointillistic Pattern Search



Search for *region patterns* with *point observations*.

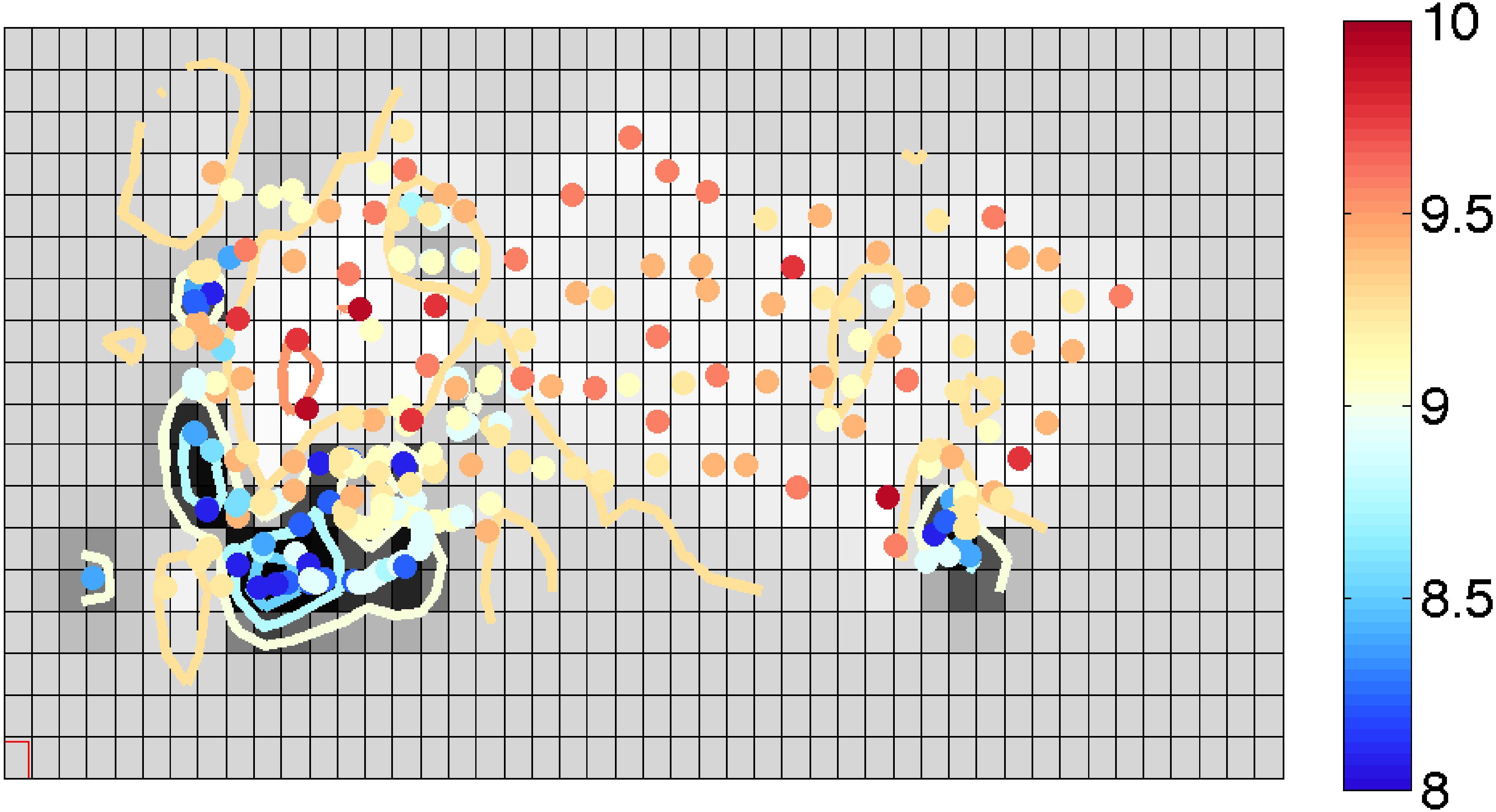
Ma*, Sutherland*, Garnett, Schneider, AISTATS 2015. (*: equal contribution)

Active Pointillistic Pattern Search



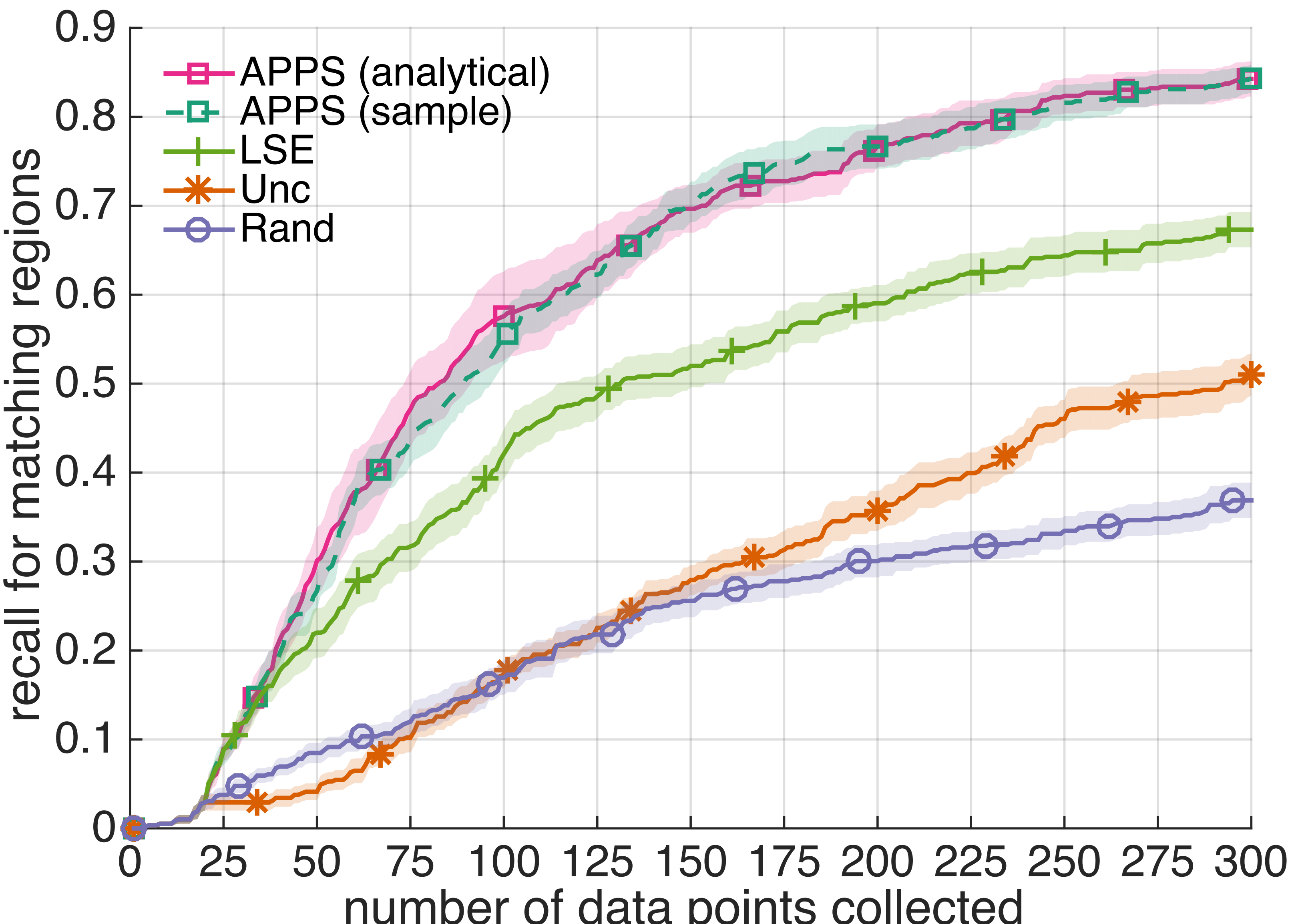
Ma*, Sutherland*, Garnett, Schneider, AISTATS 2015. (*: equal contribution)

Active Pointillistic Pattern Search



Ma*, Sutherland*, Garnett, Schneider, AISTATS 2015. (*: equal contribution)

Active Pointillistic Pattern Search



Ma*, Sutherland*, Garnett, Schneider, AISTATS 2015. (*: equal contribution)

Take-Home Messages

- Think about how you model your data.
- Distributions and sets can work pretty well.
  - Cosmology, nuclear threat detection, scene classification, parametric statistical inference, polling, autonomous sensing…
- Random embeddings can help with scalability...
  - if you use random Fourier features, use the right one
- ...and with flexibility
  - Plug the MMD embedding into deep learning and go crazy

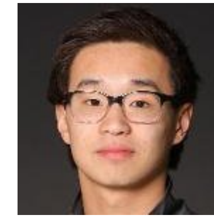
Things Still to Do

- Deep kernel learning
  - Different parameterizations of kernels
- More applications!
  - Word and document embeddings
  - Kernel-learning two-sample test as adversary in a GAN
- Active learning on distributions

Thanks!

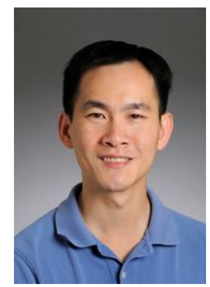
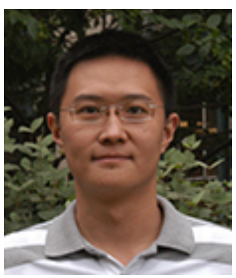











Rahul De

Ben Johnson

Contributions

- **Learning on distributions** with nonparametric kernels
- **Scalable** approximate kernel embeddings
  - Random Fourier features analysis
  - New embeddings for distribution kernels
- **Flexible** distribution kernels
  - Deep mean maps in computer vision
  - MMD kernel learning for testing
- **Active** pointillistic pattern search