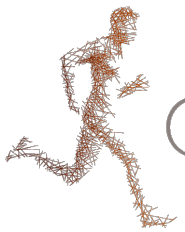
Generative Models and Model Criticism via Optimized Maximum Mean Discrepancy

Dougal J. Sutherland

Gatsby unit, UCL



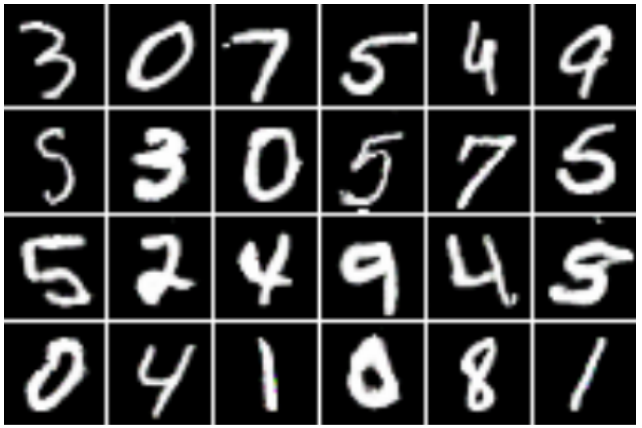
GATSBY

Two-sample testing

Observe two different datasets:

$X \sim \mathbb{P}$

vs



$Y \sim \mathbb{Q}$

Question we want to answer: is $\mathbb{P} = \mathbb{Q}$?

Two-sample testing

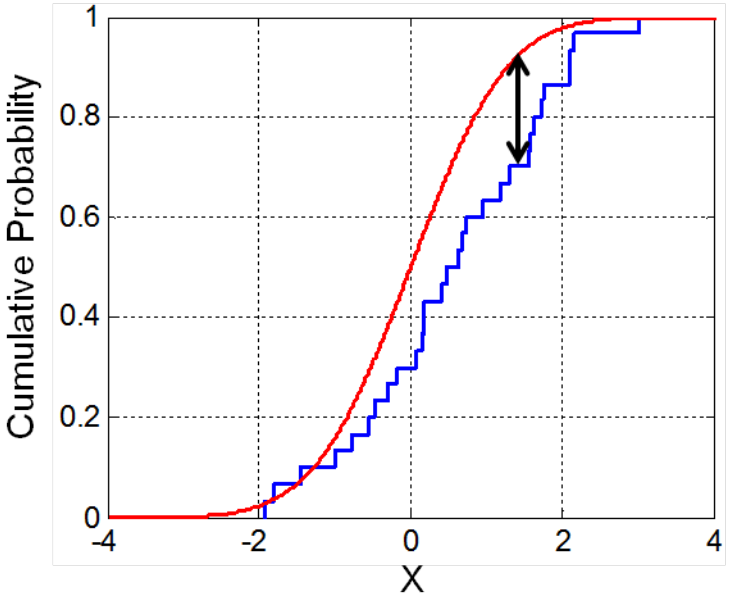
Applications:

- Do cigarette smokers and non-smokers have different distributions of cancers?
- Do these neurons behave differently when the subject is looking at background image A instead of B?
- Do these columns from different databases mean the same thing?
- Did my generative model actually learn the distribution I wanted it to?

Standard approaches

- (Unpaired) *t*-test, Wilcoxon rank-sum test, etc
  - Only test differences in location (mean)

- Kolmogorov-Smirnov test
  - Tests for all differences
  - Nonparametric
  - Hard to extend to > 1d



- Want a test that looks for all possible differences, without parametric assumptions, in multiple dimensions

Defining a two-sample test

1. Choose a distance between distributions $\rho(\mathbb{P}, \mathbb{Q})$
   - Ideally, $\rho(\mathbb{P}, \mathbb{Q}) = 0$ if and only if $\mathbb{P} = \mathbb{Q}$

2. Estimate the distribution distance from data: $\hat{\rho}(X, Y)$

3. Choose a rejection threshold; say $\mathbb{P} \neq \mathbb{Q}$ when $\hat{\rho} > c_\alpha$

$$\Pr_{X,Y \sim \mathbb{P}} (\hat{\rho}(X, Y) > c_\alpha) < \alpha$$

Reminder:

- The *level* of a test is probability of false rejection
- The *power* of a test is probability of true rejection

A Kernel Distance on Distributions

Quick reminder about kernels:

- Our data lives in a space $\mathcal{X}$
- The kernel is a similarity function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$

$$\text{e.g. } k(x, y) = \exp\left(-\frac{1}{2\sigma^2}\|x - y\|^2\right)$$

- Corresponds to a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$, with feature map $\varphi : \mathcal{X} \to \mathcal{H}$, by

$$\langle \varphi(x), \varphi(y) \rangle_{\mathcal{H}} = k(x, y)$$

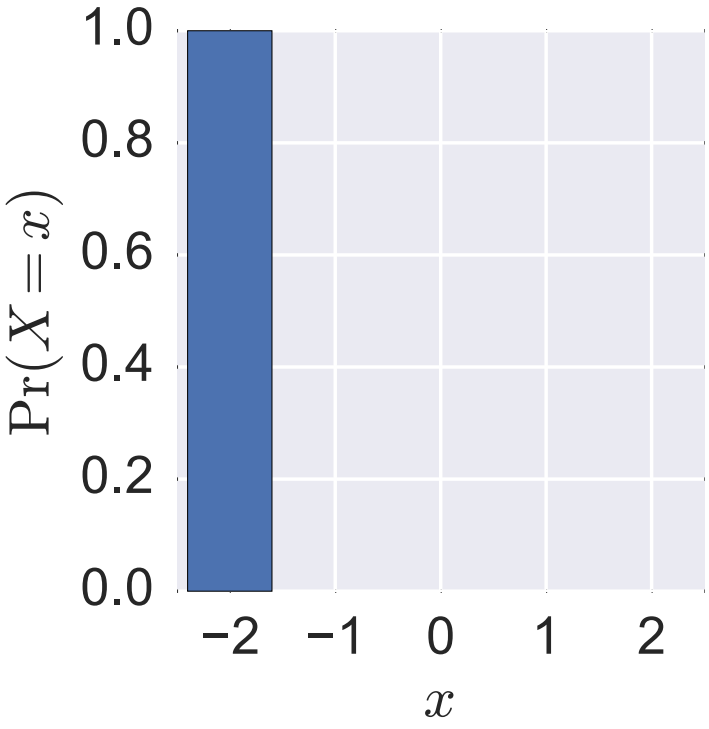
We'll use a *base kernel* on $\mathcal{X}$ to make a distance between distributions over $\mathcal{X}$.

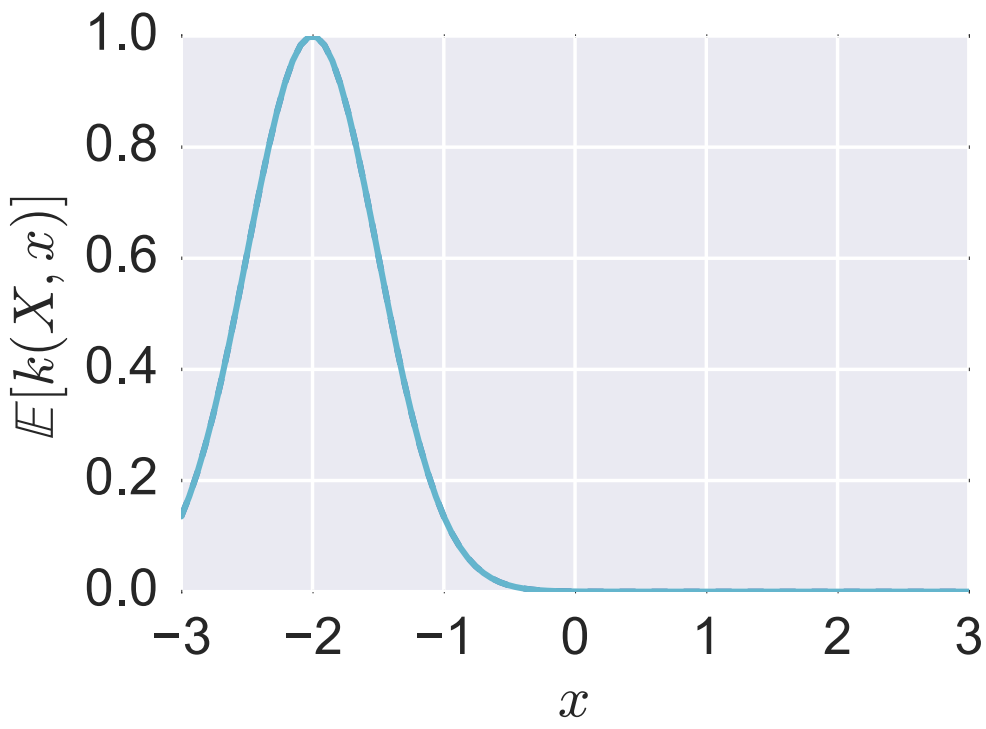
Mean embeddings of distributions

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 $k(x, \cdot)$.



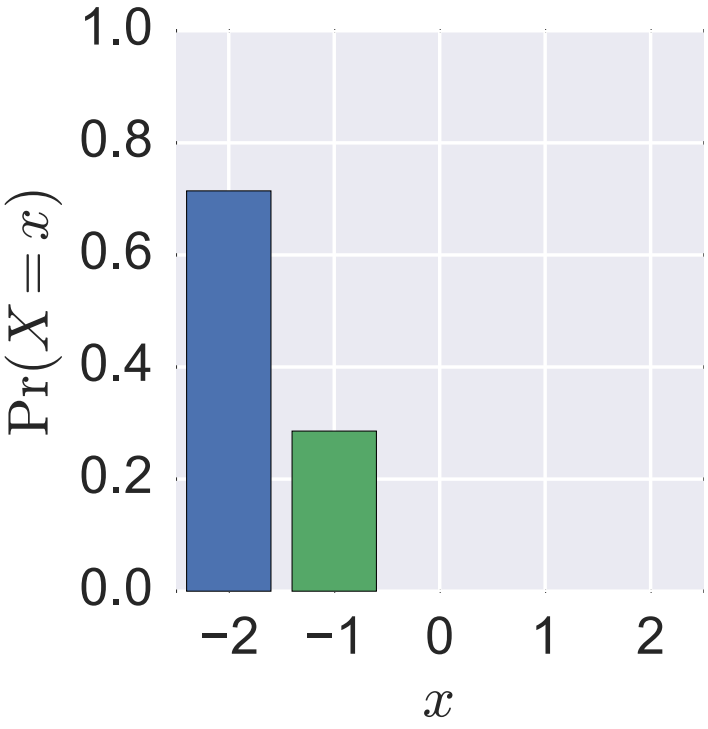


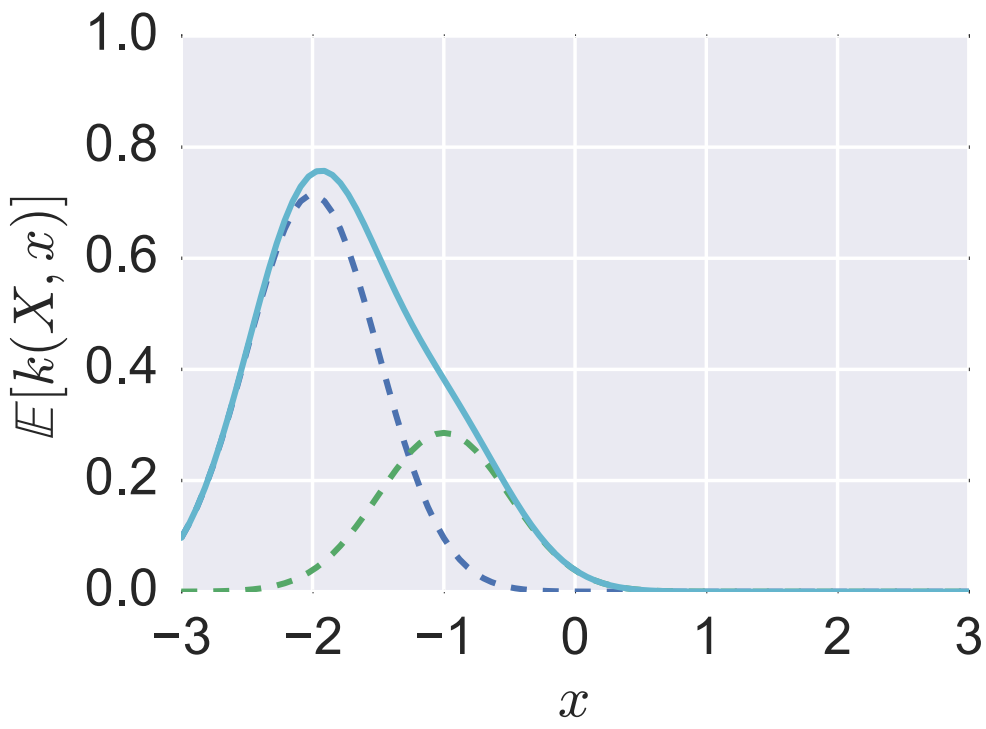
Mean embeddings of distributions

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 $k(x, \cdot)$.



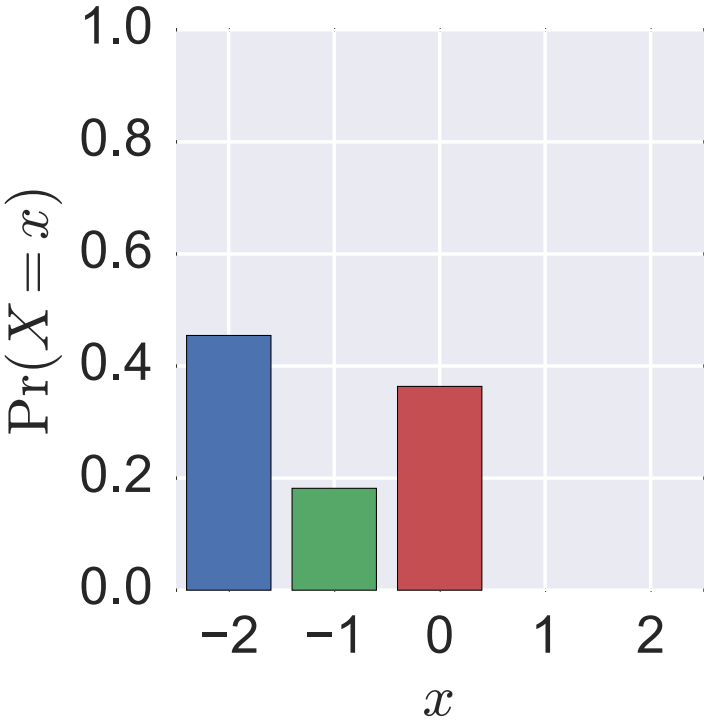


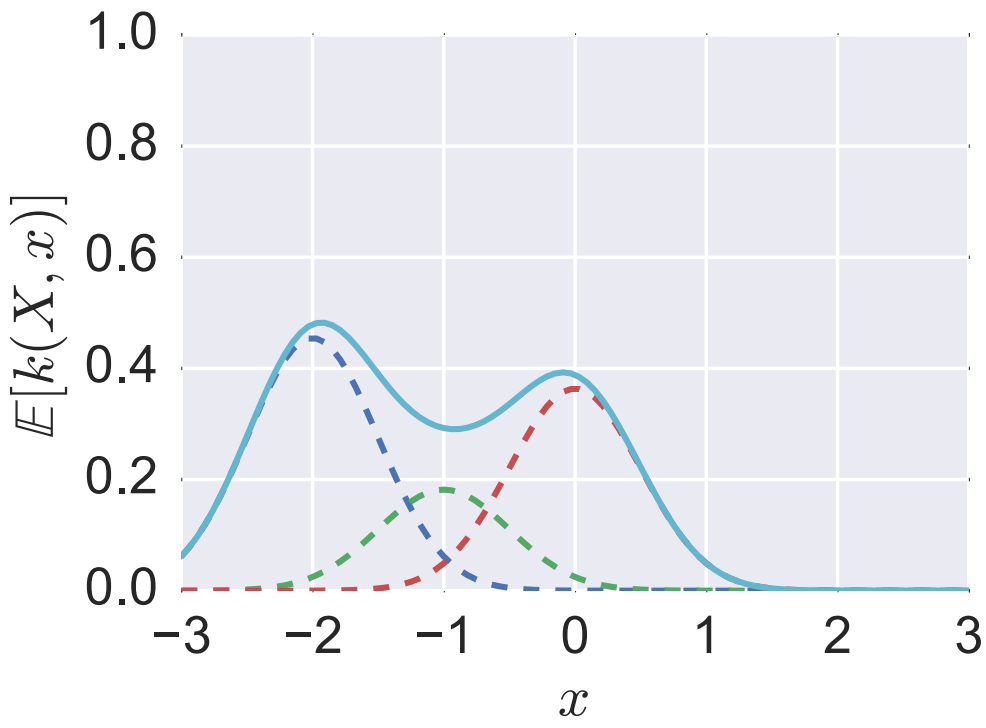
Mean embeddings of distributions

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 $k(x, \cdot)$.



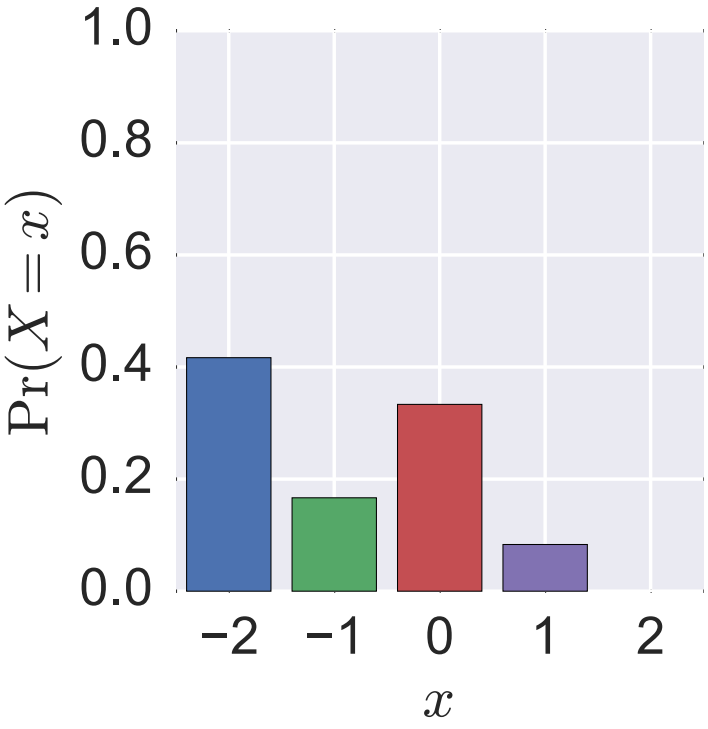


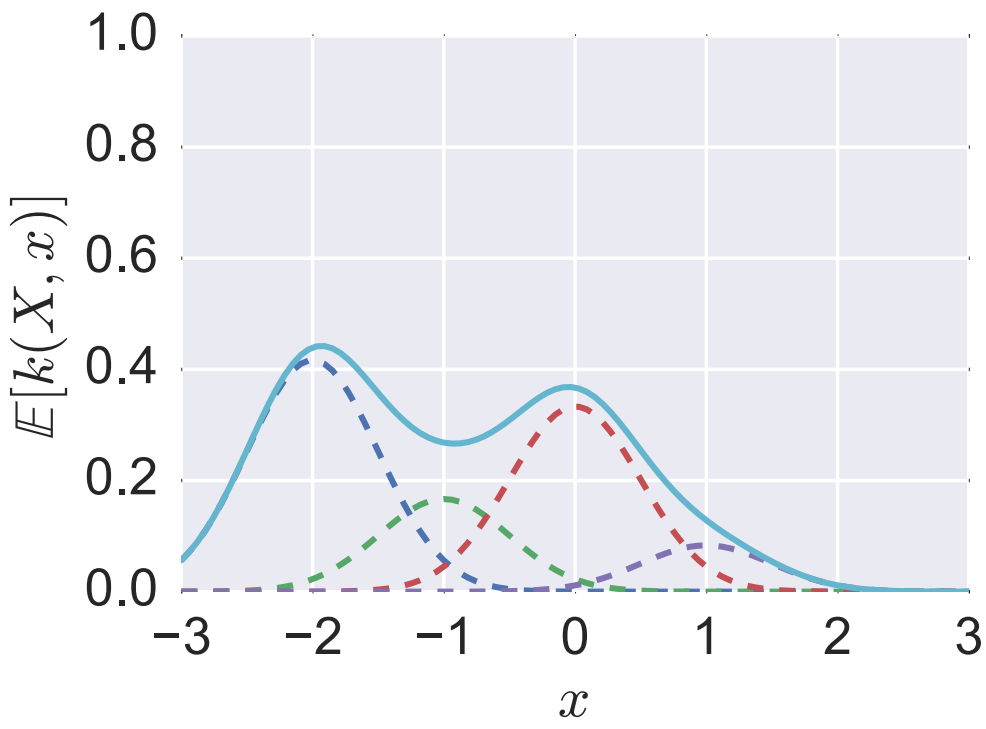
Mean embeddings of distributions

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 $k(x, \cdot)$.



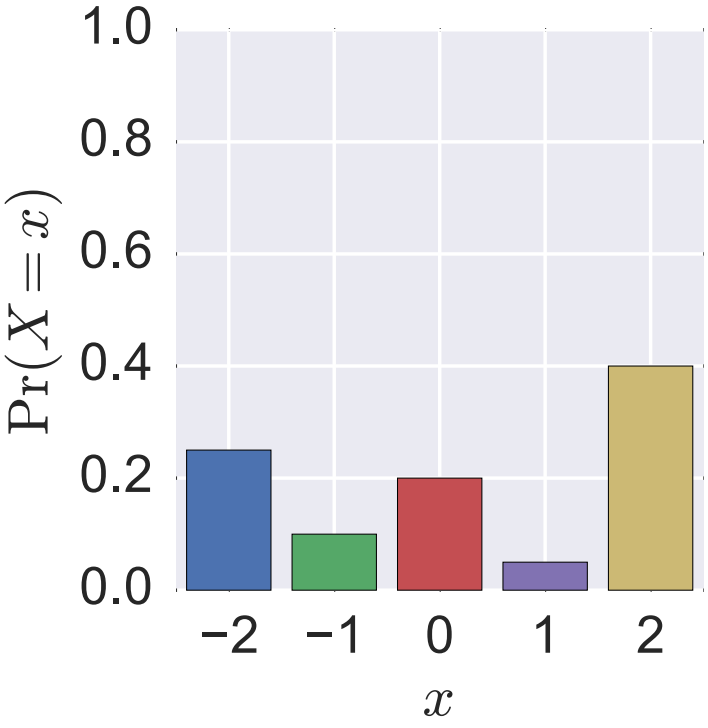


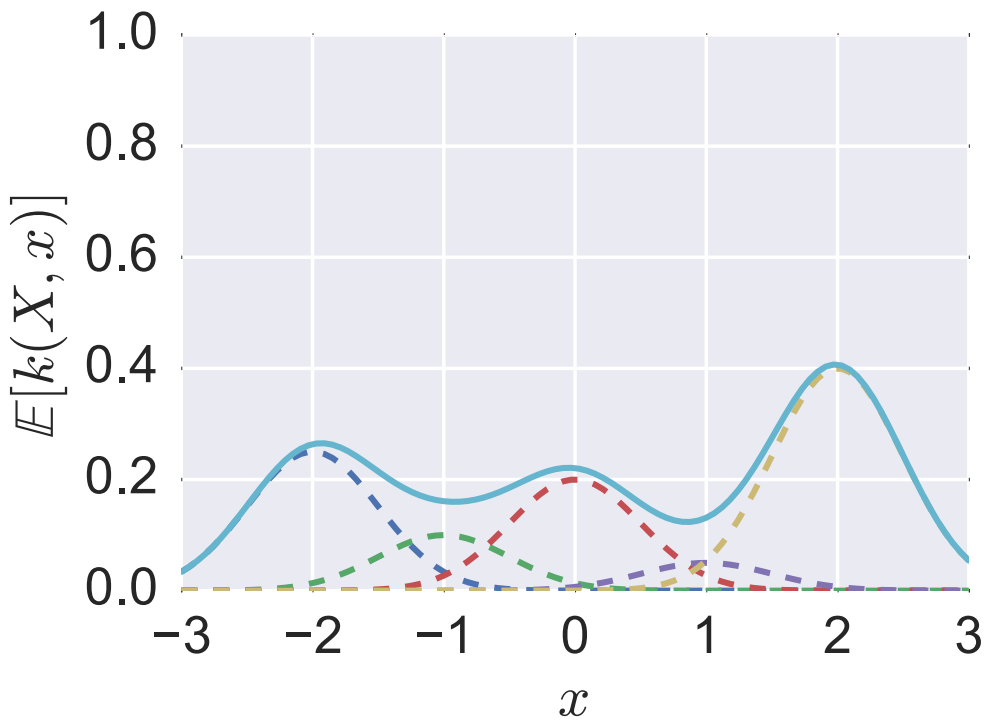
Mean embeddings of distributions

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 $k(x, \cdot)$.

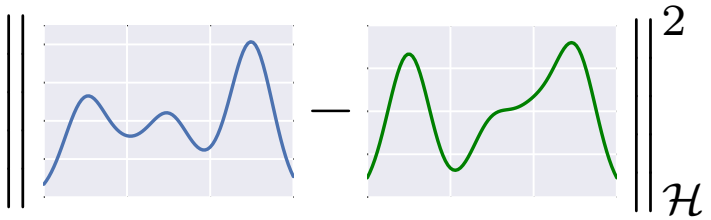




Maximum Mean Discrepancy (MMD)

The MMD is the distance between mean embeddings:

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



$$\begin{aligned}\mathrm{MMD}^2(\mathbb{P},\mathbb{Q}) &= \|\mu_{\mathbb{P}} - \mu_{\mathbb{Q}}\|^2_{\mathcal{H}} \\ &= \langle\mu_{\mathbb{P}},\mu_{\mathbb{P}}\rangle + \langle\mu_{\mathbb{Q}},\mu_{\mathbb{Q}}\rangle - 2\langle\mu_{\mathbb{P}},\mu_{\mathbb{Q}}\rangle\end{aligned}$$

$$\begin{aligned}\langle\mu_{\mathbb{P}},\mu_{\mathbb{Q}}\rangle &= \langle\mathbb{E}_{X\sim P}[\varphi(X)],\mathbb{E}_{Y\sim Q}[\varphi(Y)]\rangle \\ &= \mathbb{E}_{\substack{X\sim P\\Y\sim Q}}[\langle\varphi(X),\varphi(Y)\rangle] \\ &= \mathbb{E}_{\substack{X\sim P\\Y\sim Q}}[k(X,Y)]\end{aligned}$$

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

MMD estimator

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

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

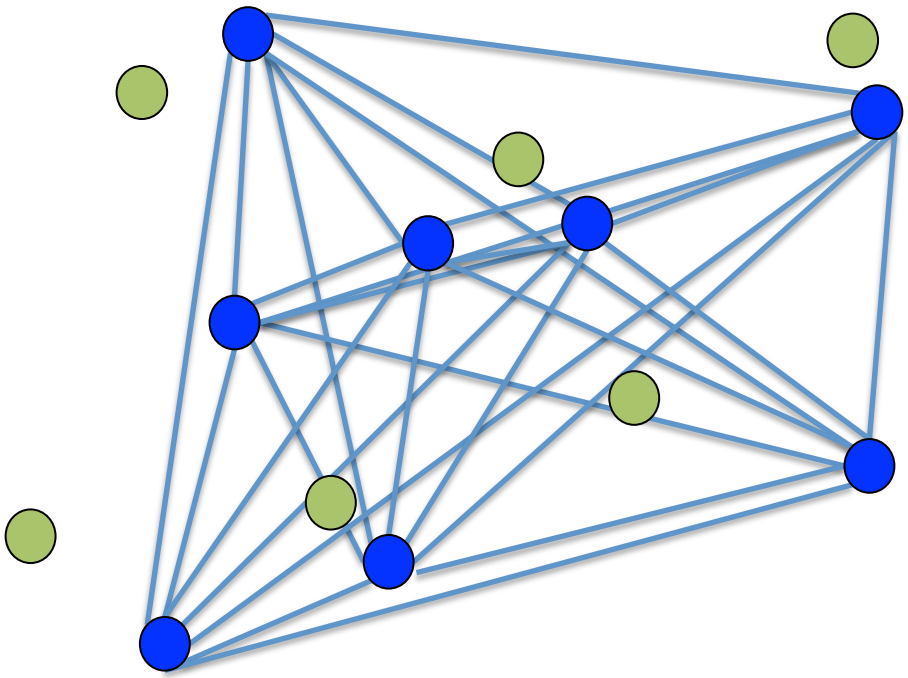
$$\langle \hat{\mu}_{\mathbb{P}}, \hat{\mu}_{\mathbb{Q}} \rangle = \frac{1}{mn} \sum_{ij} k(X_i, Y_j)$$

MMD estimator

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

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

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

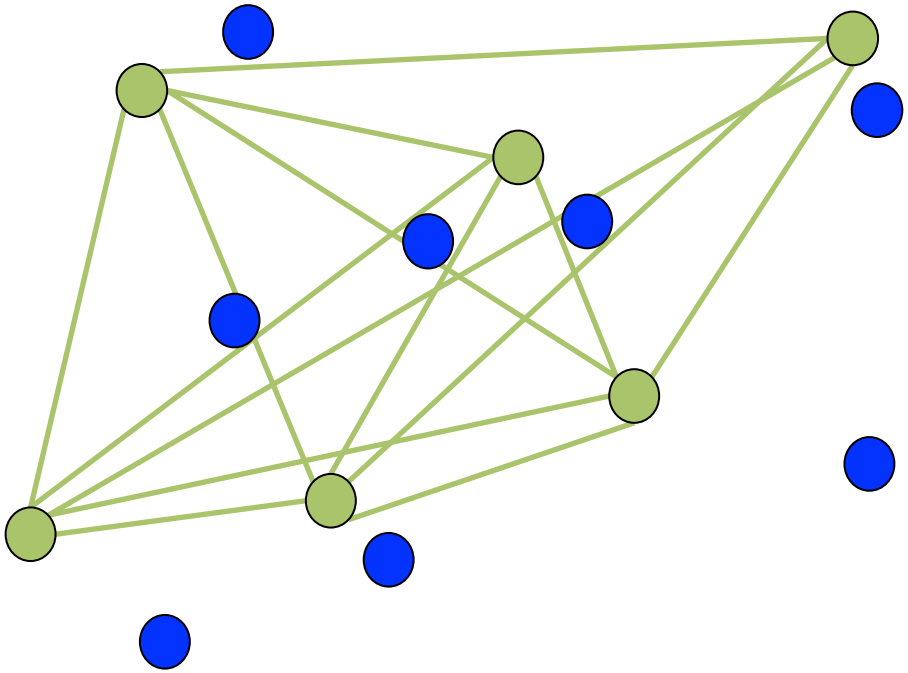


MMD estimator

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

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

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

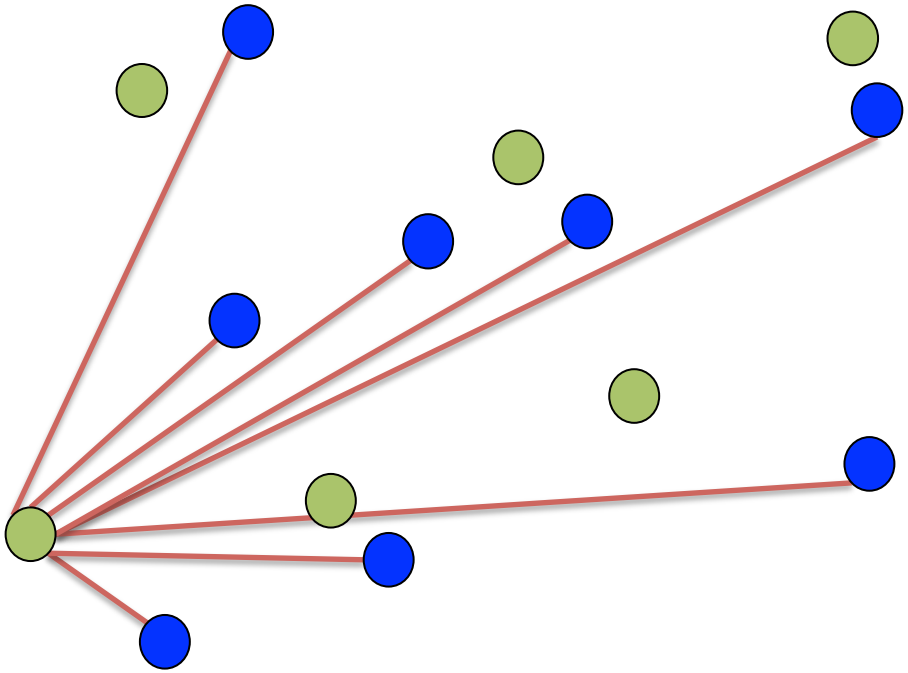


MMD estimator

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

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

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

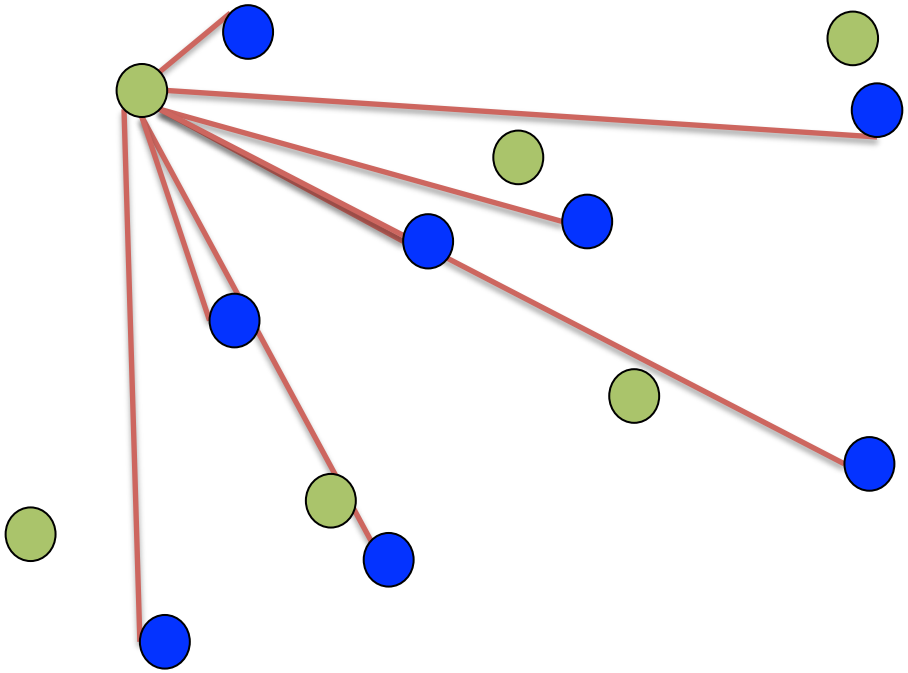


MMD estimator

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

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

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

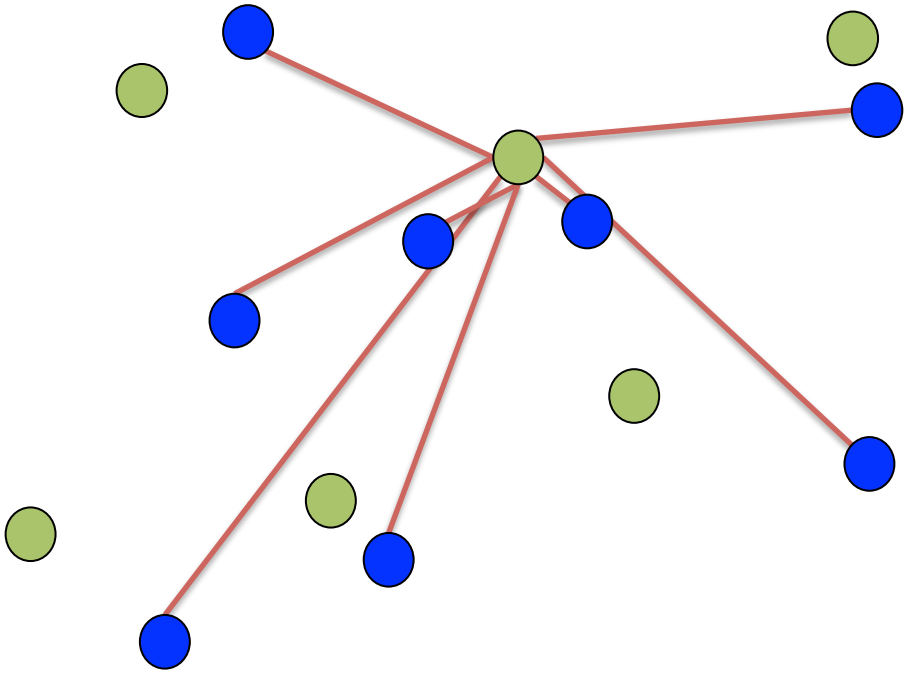


MMD estimator

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

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

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

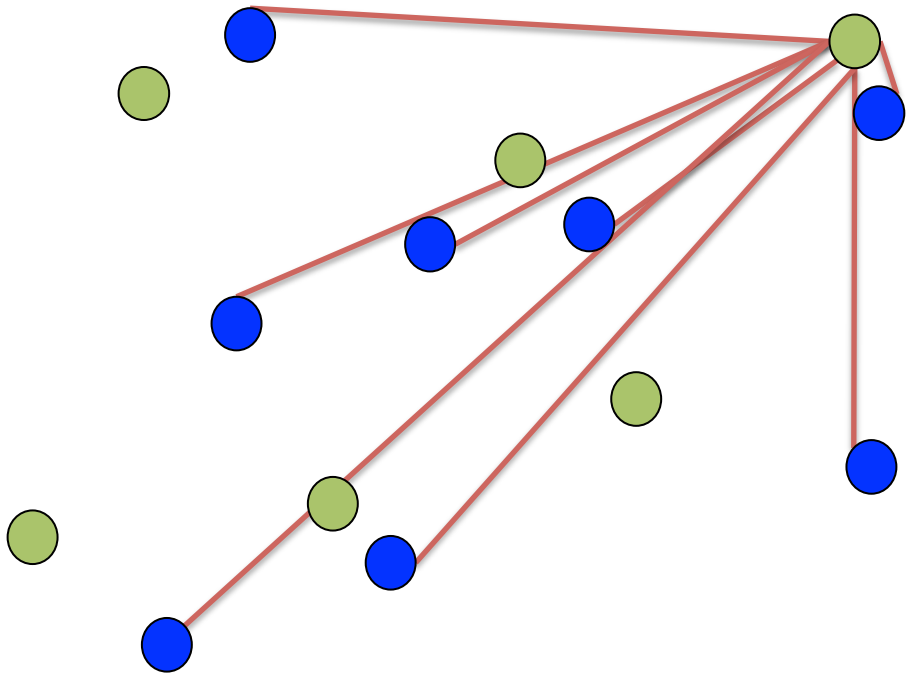


MMD estimator

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

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

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

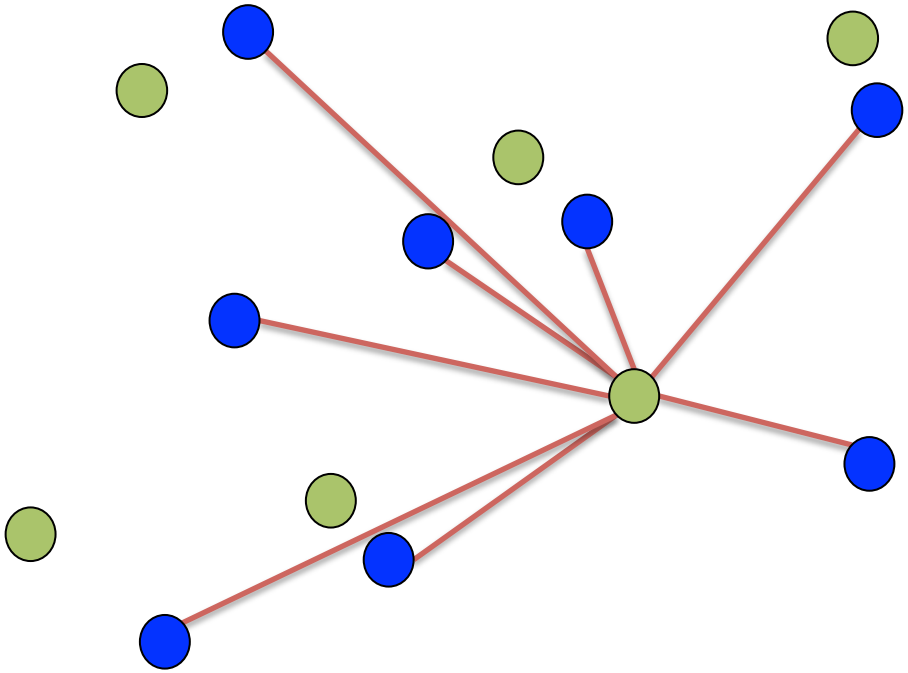


MMD estimator

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

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

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

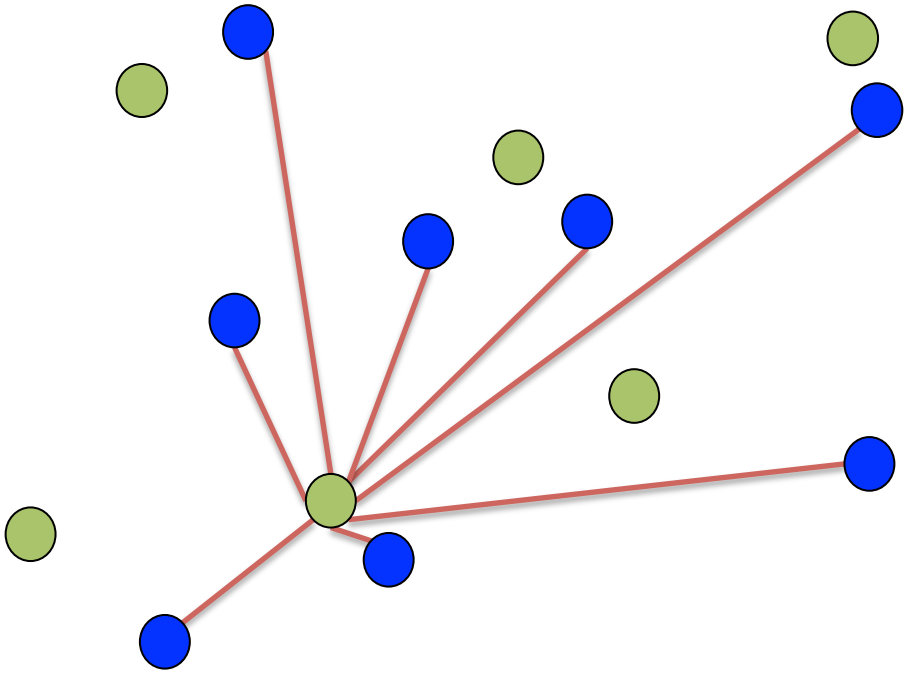


MMD estimator

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

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

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

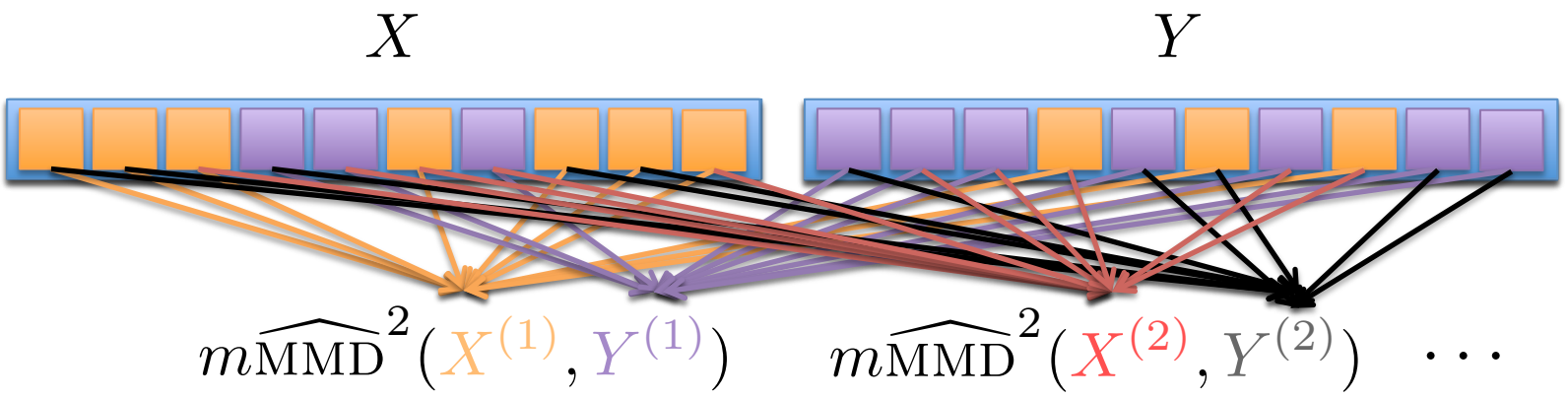


Permutation testing

Still need rejection threshold: $\Pr_{X,Y\sim\mathbb{P}}(\hat{\rho}(X,Y) > c_\alpha) \leq \alpha$

When $\mathbb{P} = \mathbb{Q}$, MMD asymptotics depend on $\mathbb{P}$, so it's hard to find a threshold that way.

Permutation test: split randomly to estimate MMD when $\mathbb{P} = \mathbb{Q}$.

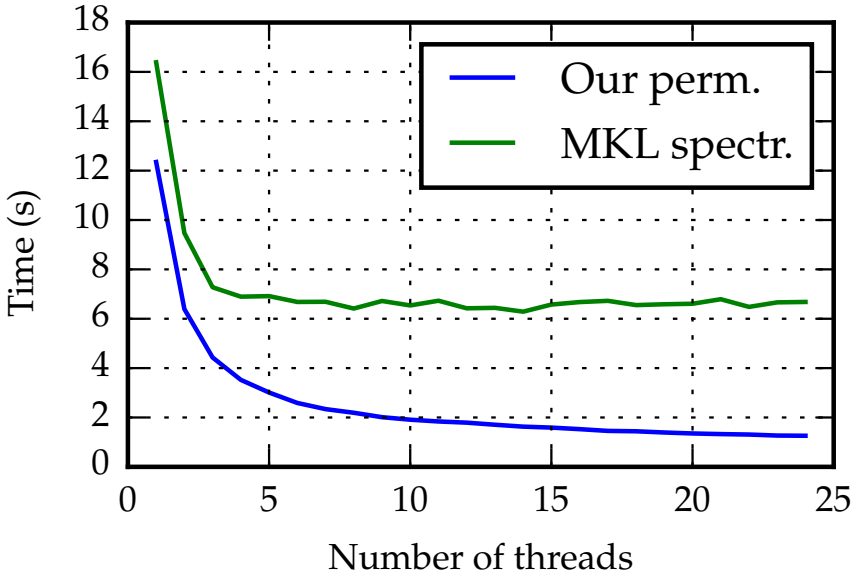


$\hat{c}_\alpha$: (1-α)th quantile of $m\widehat{\mathrm{MMD}}^2(X^{(i)}, Y^{(i)})$

Computing permutation tests

$$K = \begin{bmatrix} k(X_1, X_1) & \dots & K(X_1, X_m) & K(X_1, Y_1) & \dots & K(X_1, Y_m) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ k(X_m, X_1) & \dots & K(X_m, X_m) & K(X_m, Y_1) & \dots & K(X_m, Y_m) \\ k(Y_1, X_1) & \dots & K(Y_1, X_m) & K(Y_1, Y_1) & \dots & K(Y_1, Y_m) \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ k(Y_m, X_1) & \dots & K(Y_m, X_m) & K(Y_m, Y_1) & \dots & K(Y_m, Y_m) \end{bmatrix}$$

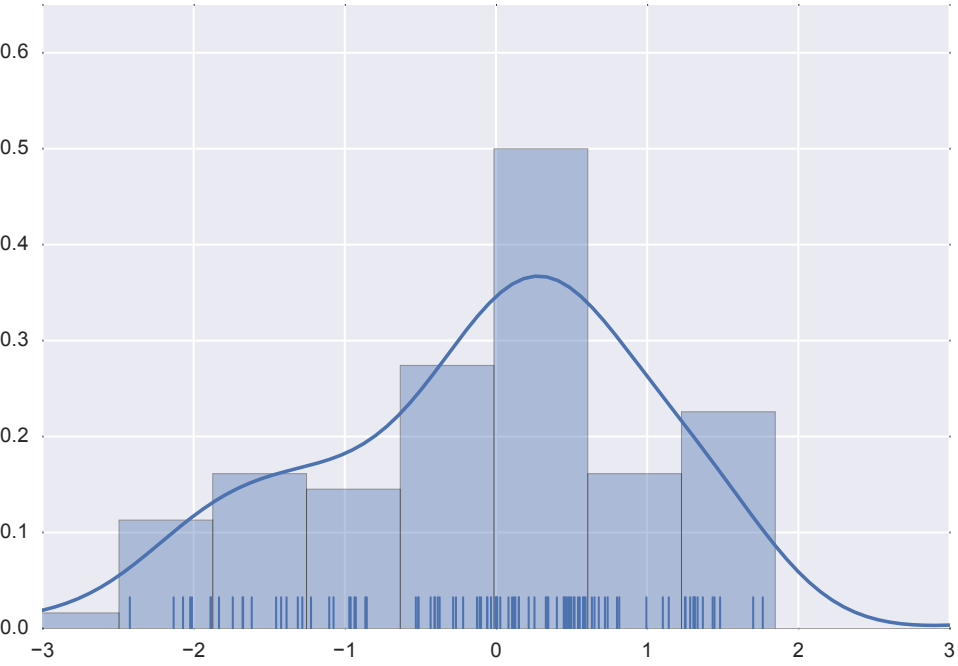
Each element of K is added or subtracted to a term of each permutation estimate. So, do it all in one pass.



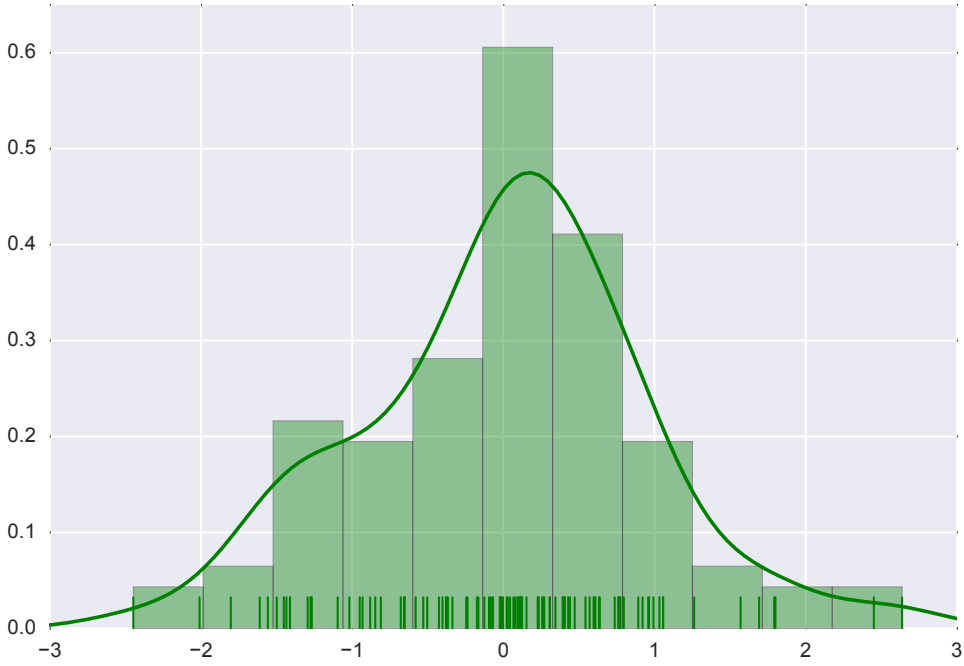
Original Matlab code: 381s

Better Python code: 182s

Example two-sample test



$$X \sim \mathbb{P} = \mathcal{N}(0, 1)$$



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

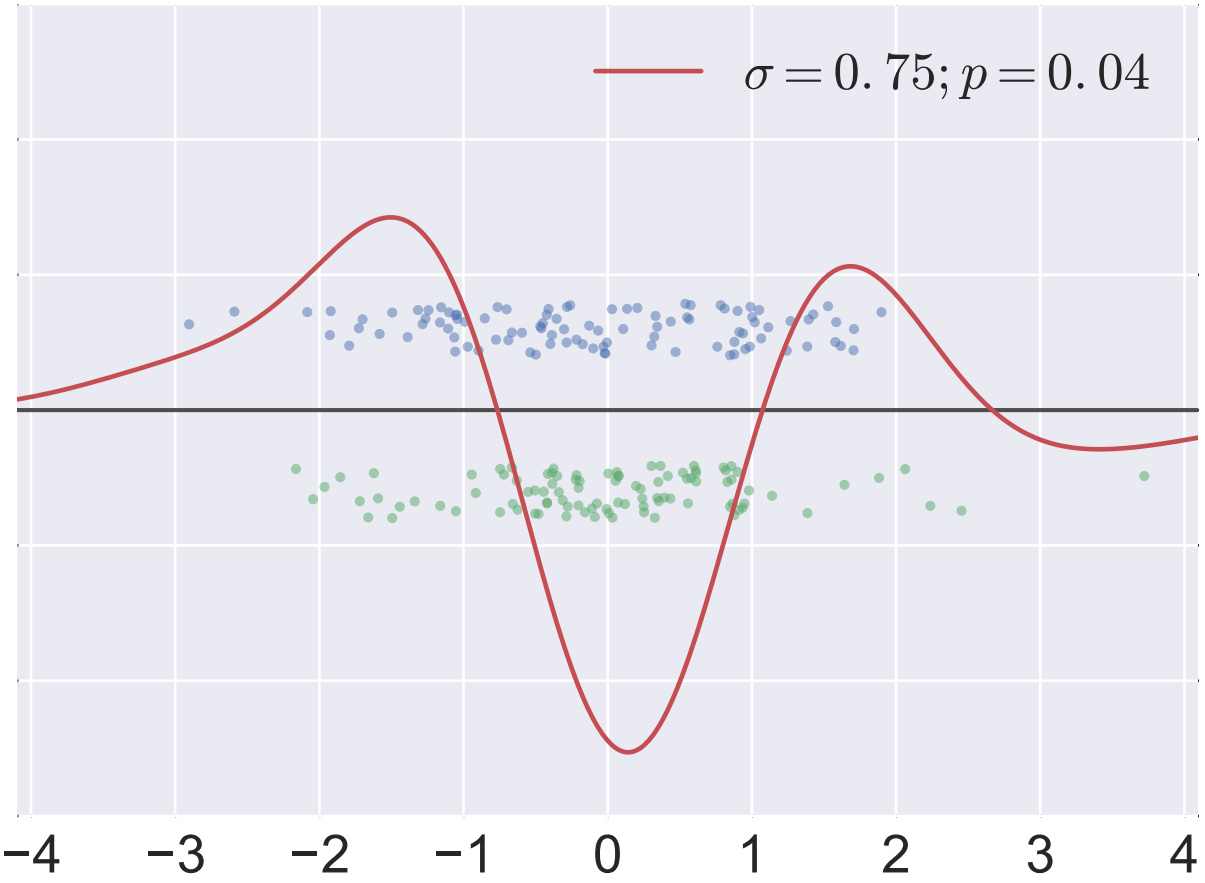
1. Choose a kernel *k*
2. Estimate MMD for true division and many permutations
3. Reject if $m\,\widehat{\text{MMD}}_k^2(X, Y) > c_\alpha$

The kernel matters!

Witness function f helps compare samples:

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

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

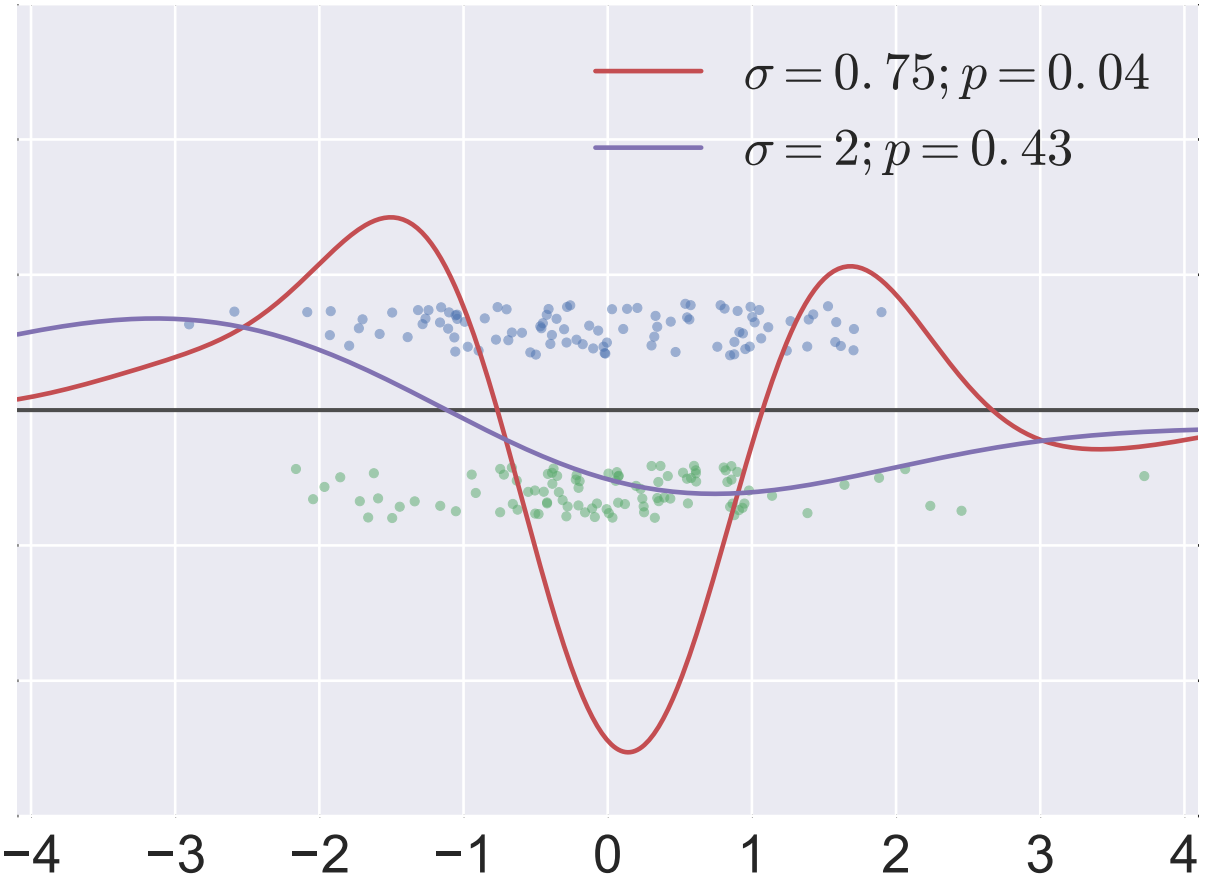


The kernel matters!

Witness function f helps compare samples:

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

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

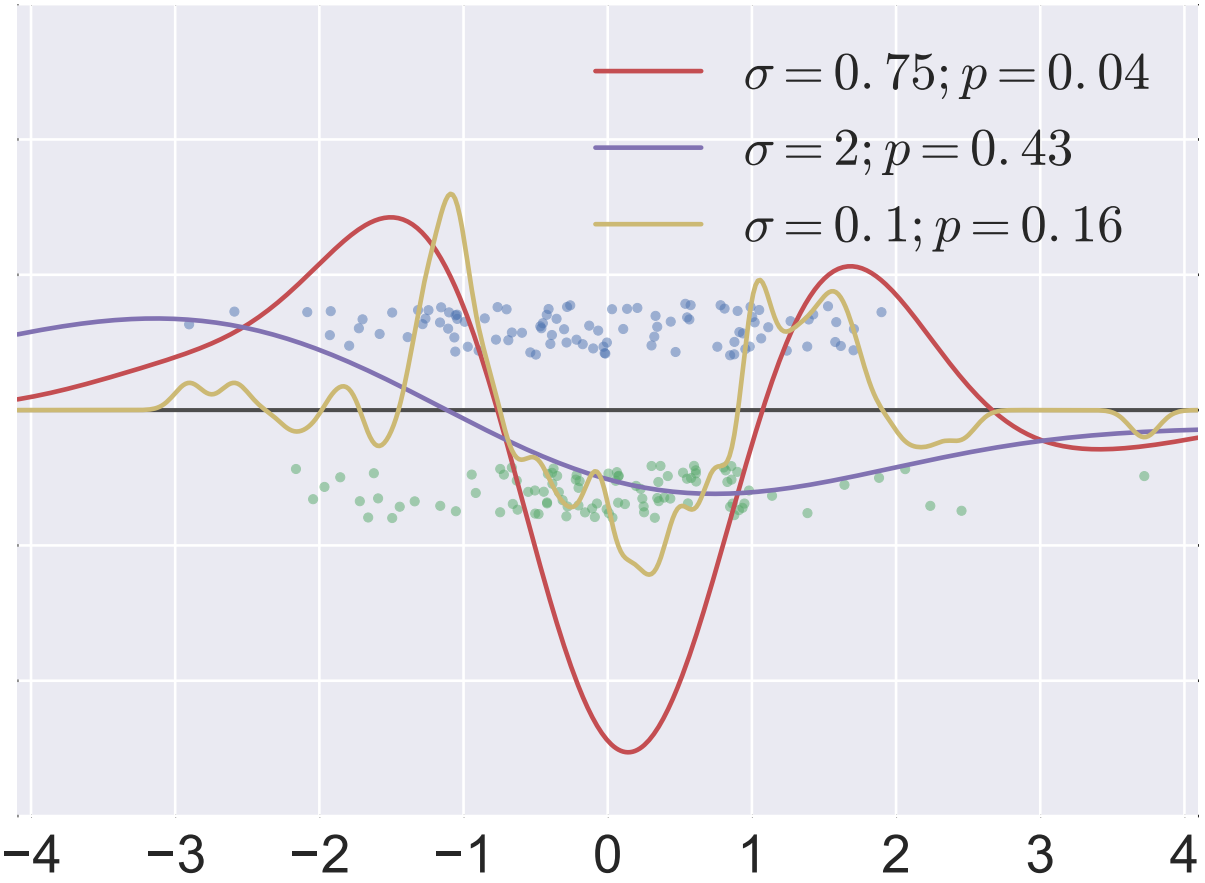


The kernel matters!

Witness function f helps compare samples:

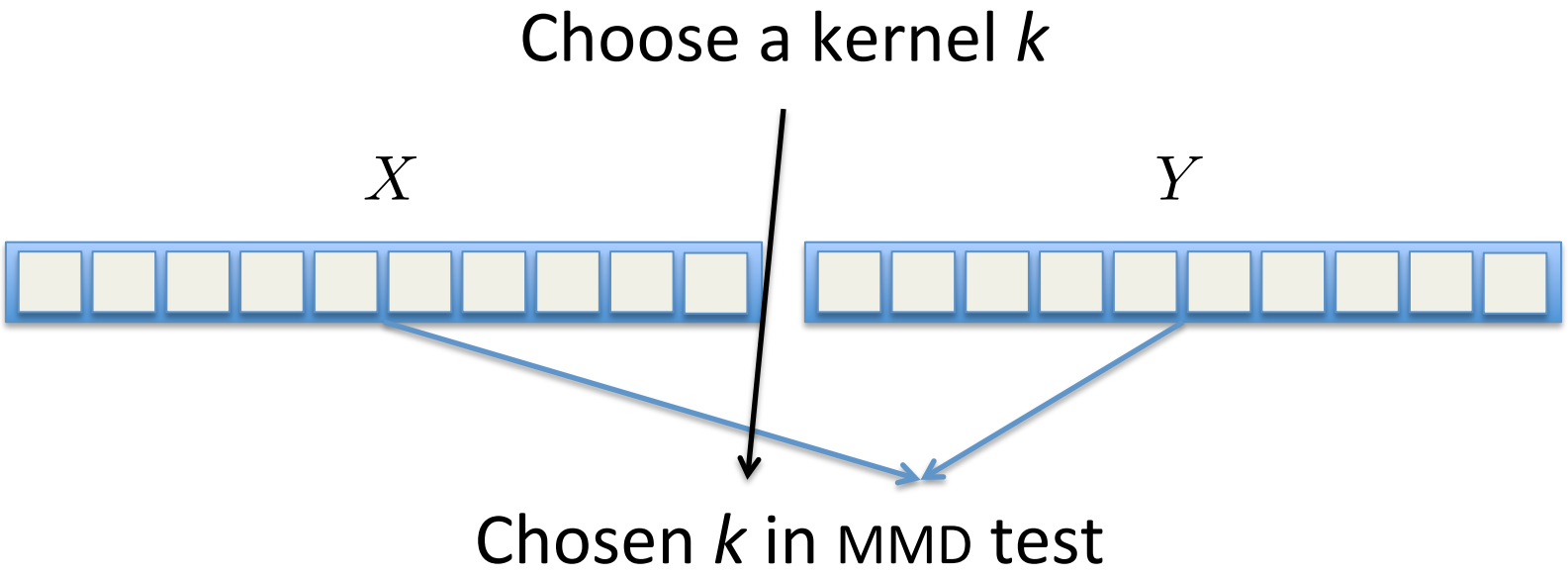
$$\mathrm{MMD}(\mathbb{P}, \mathbb{Q}) = \mathbb{E}_{X \sim \mathbb{P}} f(X) - \mathbb{E}_{Y \sim \mathbb{Q}} f(Y)$$

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



Choosing a kernel

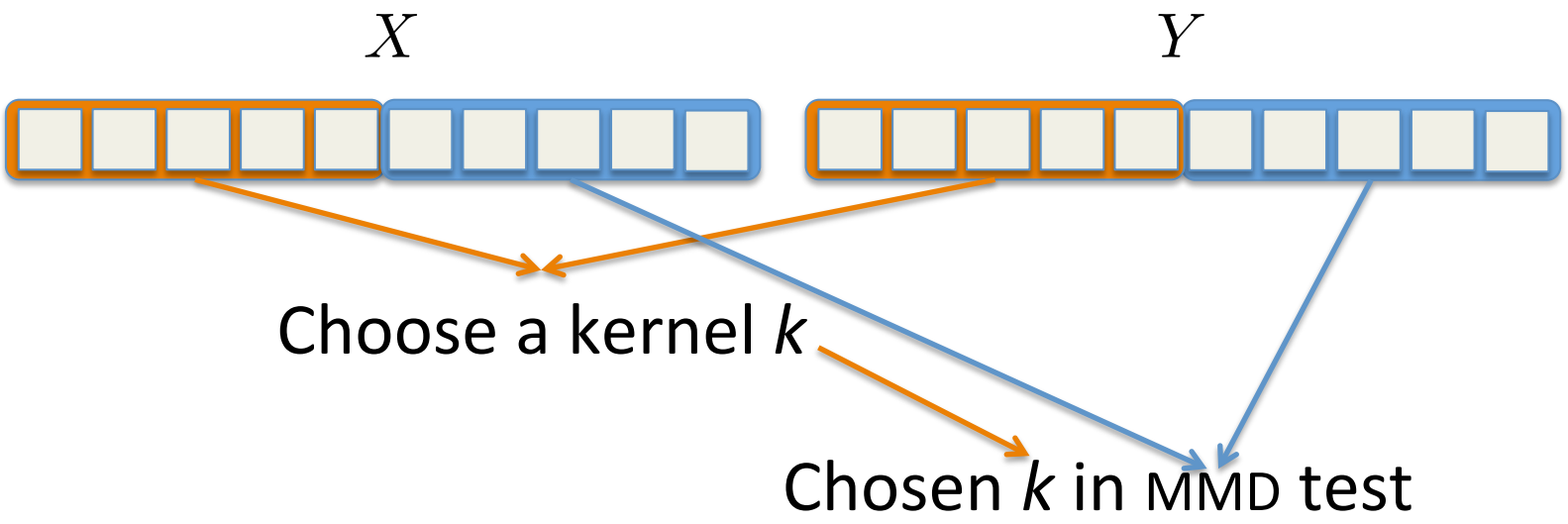
So we need a way to pick a kernel to do the test.



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 $\mathbb{P} \neq \mathbb{Q}$, the MMD estimator 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:

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

$$\to \Phi\left(\frac{\mathrm{MMD}^2}{\sqrt{V_m}} - \frac{c_\alpha}{m\sqrt{V_m}}\right)$$

MMD *t*-statistic

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

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

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

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

$$\begin{aligned}
\widehat{V}_m := {} & \frac{2}{m^2(m-1)^2} \left(2\|\tilde{K}_{XX} e\|^2 - \|\tilde{K}_{XX}\|_F^2 + 2\|\tilde{K}_{YY} e\|^2 - \|\tilde{K}_{YY}\|_F^2 \right) \\
& - \frac{4m-6}{m^3(m-1)^3} \left[\left(e^\mathsf{T} \tilde{K}_{XX} e \right)^2 + \left(e^\mathsf{T} \tilde{K}_{YY} e \right)^2 \right] + \frac{4(m-2)}{m^3(m-1)^2} \left(\|K_{XY} e\|^2 + \|K_{XY}^\mathsf{T} e\|^2 \right) \\
& - \frac{4(m-3)}{m^3(m-1)^2} \|K_{XY}\|_F^2 - \frac{8m-12}{m^5(m-1)} \left(e^\mathsf{T} K_{XY} e \right)^2 \\
& + \frac{8}{m^3(m-1)} \left(\frac{1}{m} \left(e^\mathsf{T} \tilde{K}_{XX} e + e^\mathsf{T} \tilde{K}_{YY} e \right) \left(e^\mathsf{T} K_{XY} e \right) - e^\mathsf{T} \tilde{K}_{XX} K_{XY} e - e^\mathsf{T} \tilde{K}_{YY} K_{XY}^\mathsf{T} e \right)
\end{aligned}$$

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

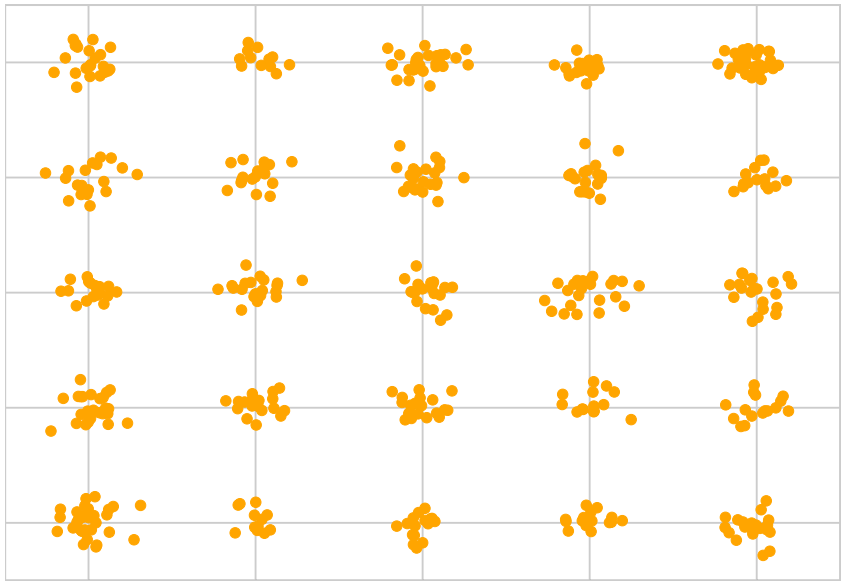
t-statistic estimator

Can even get gradients of $\hat{t}_U$ and (with some more effort) $\hat{\tau}_U$, to maximize it.

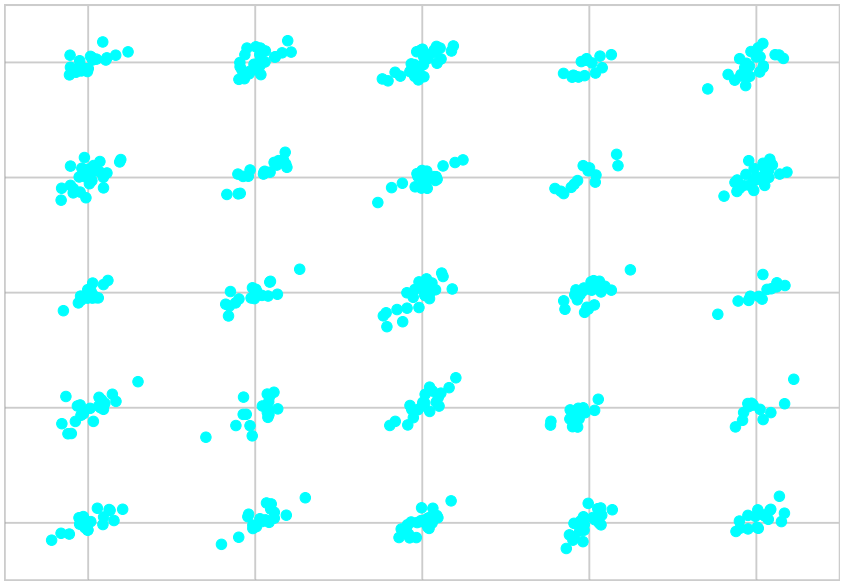
(automatic differentiation is your friend)

Kernel choice on Blobs

Blobs dataset:



vs



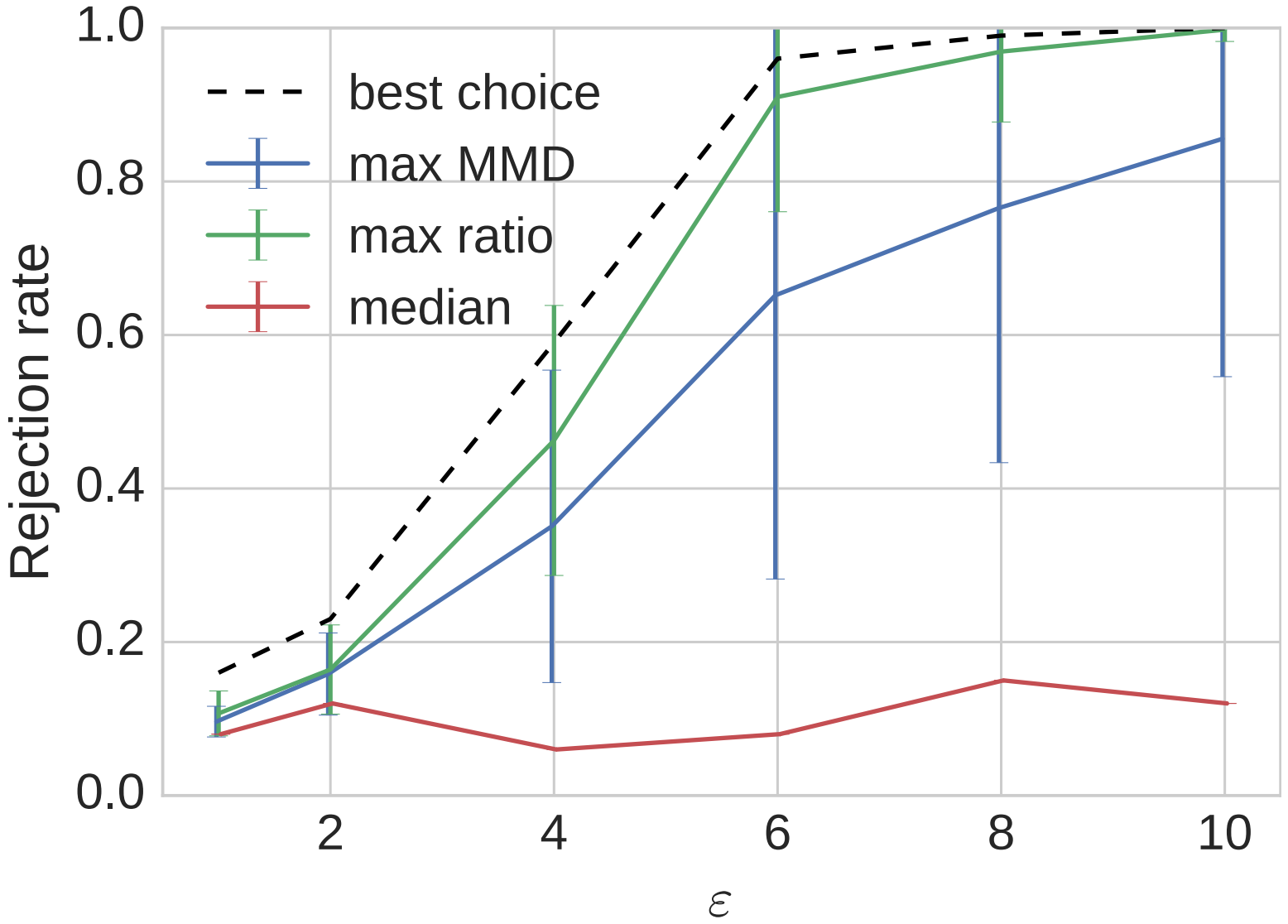
Mixture of $\mathcal{N}\left(\mu_{ij}, \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}\right)$

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.

Only consider choosing the bandwidth of a Gaussian kernel.

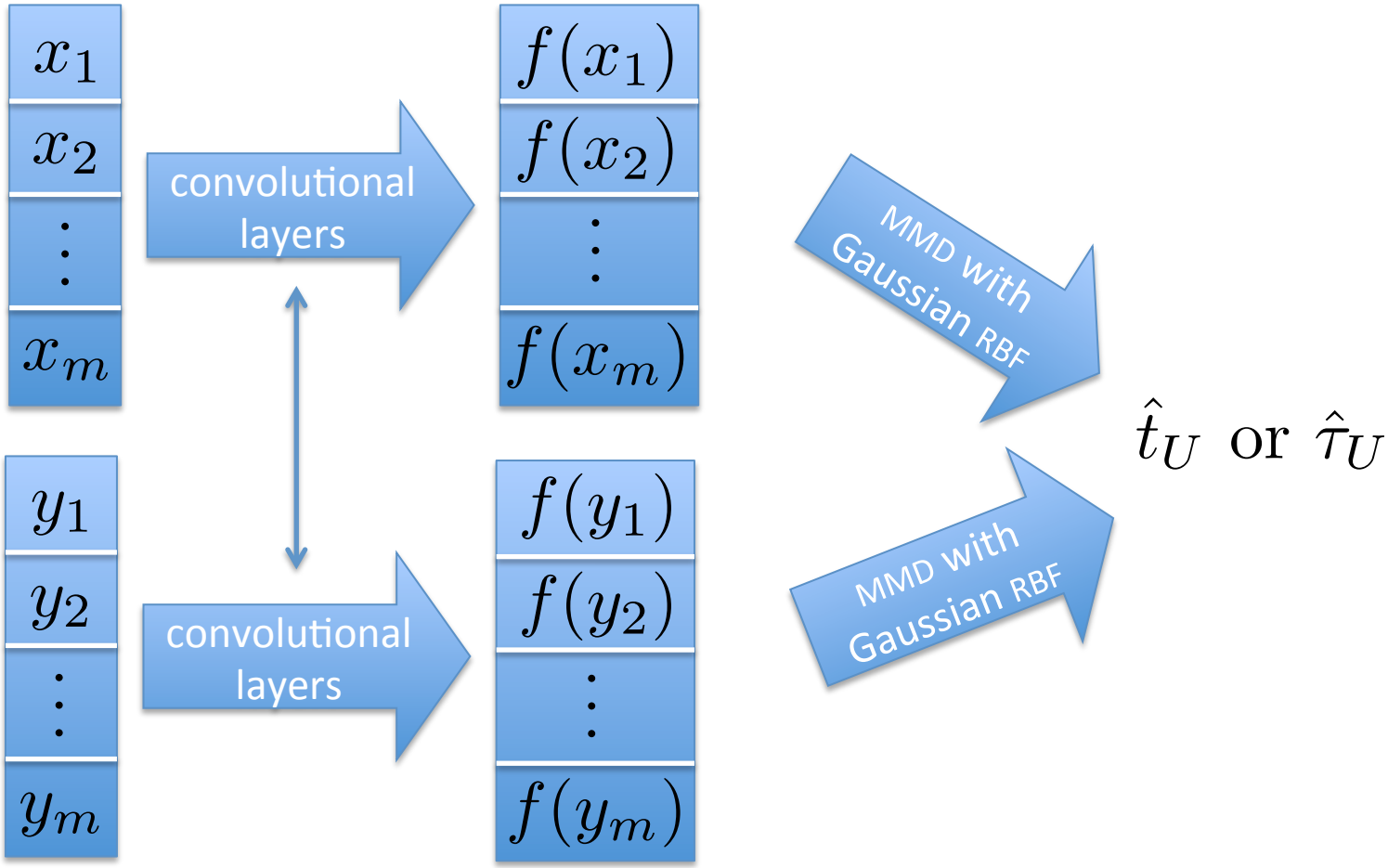
Kernel choice on Blobs



$m = 500$

Deep kernels

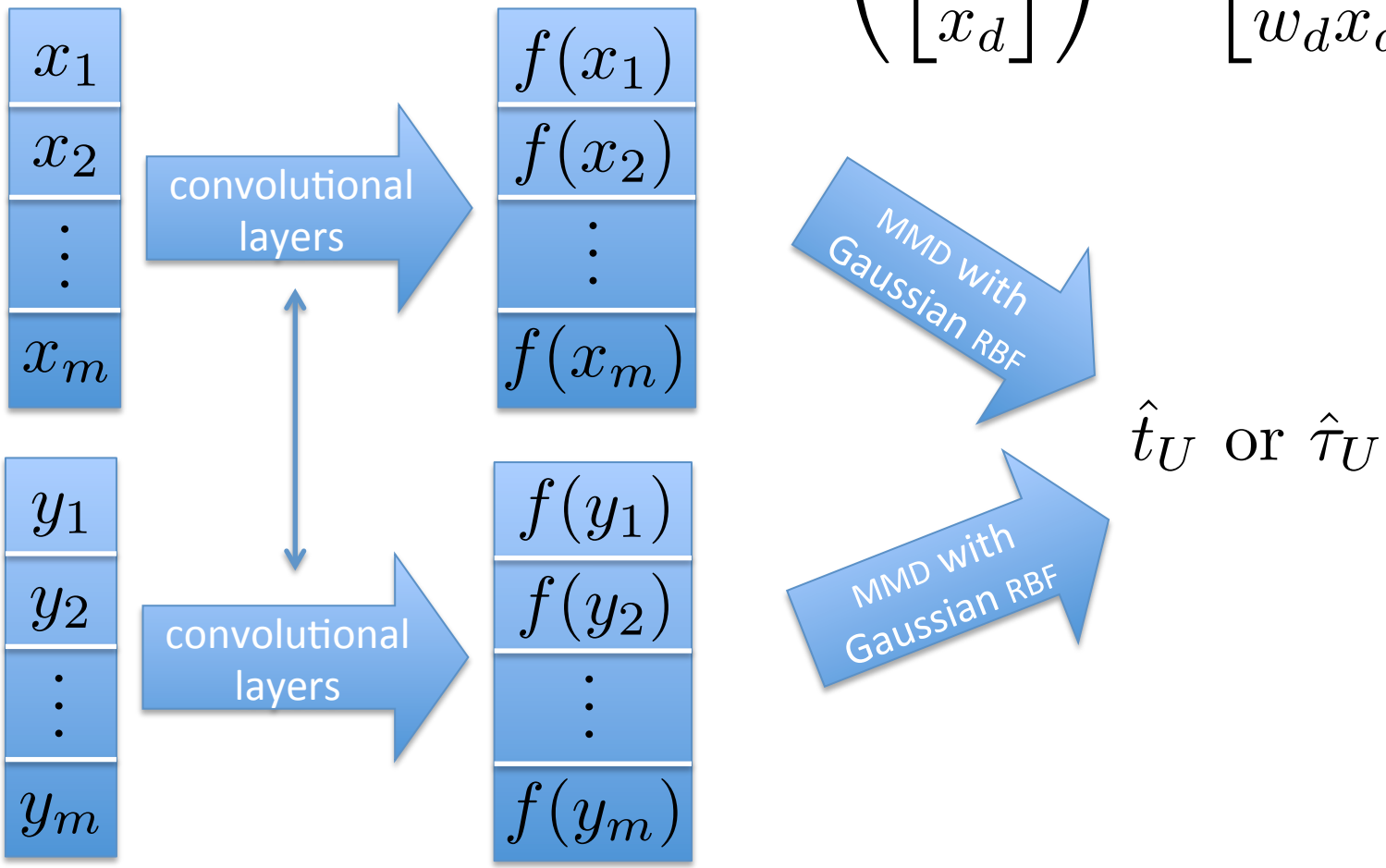
Map through layers of a deep network:



ARD kernel

Simple scaling function:

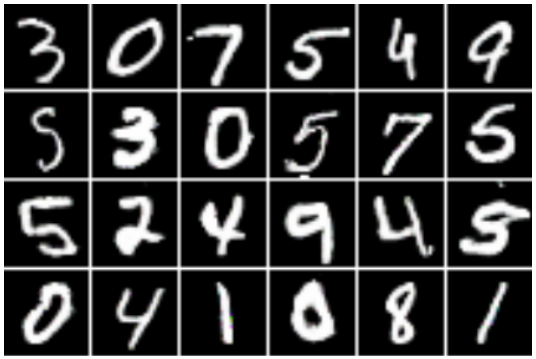
$$f\left(\begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix}\right) = \begin{bmatrix} w_1 x_1 \\ \vdots \\ w_d x_d \end{bmatrix}$$



Generative model criticism

MNIST digits

vs



model samples

Improved Techniques for Training GANs

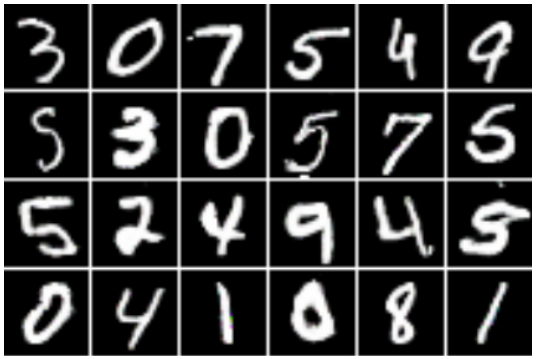
Samples generated by the generator during semi-supervised learning using feature matching (Section 3.1) do not look visually appealing (left Fig. 3). By using minibatch discrimination instead (Section 3.2) we can improve their visual quality. MNIST dataset. *(Right)* Samples generated with minibatch discrimination. Samples are completely indistinguishable from dataset images.

On MTurk, annotators were able to distinguish samples in 52.4% of cases (2000 votes total), where 50% would be obtained by random guessing. Similarly, researchers in our institution were not able to find any artifacts that would allow them to distinguish samples. However, semi-supervised learning with minibatch discrimination does not produce as good a classifier as does feature matching.

Generative model criticism

MNIST digits

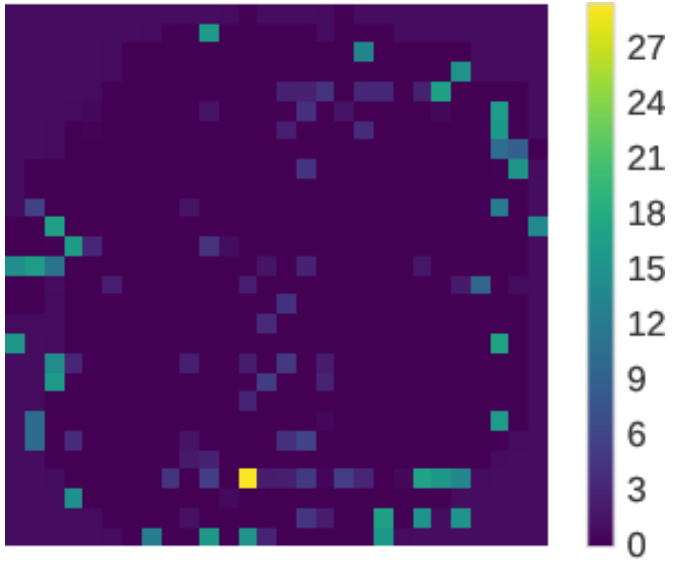
vs



model samples

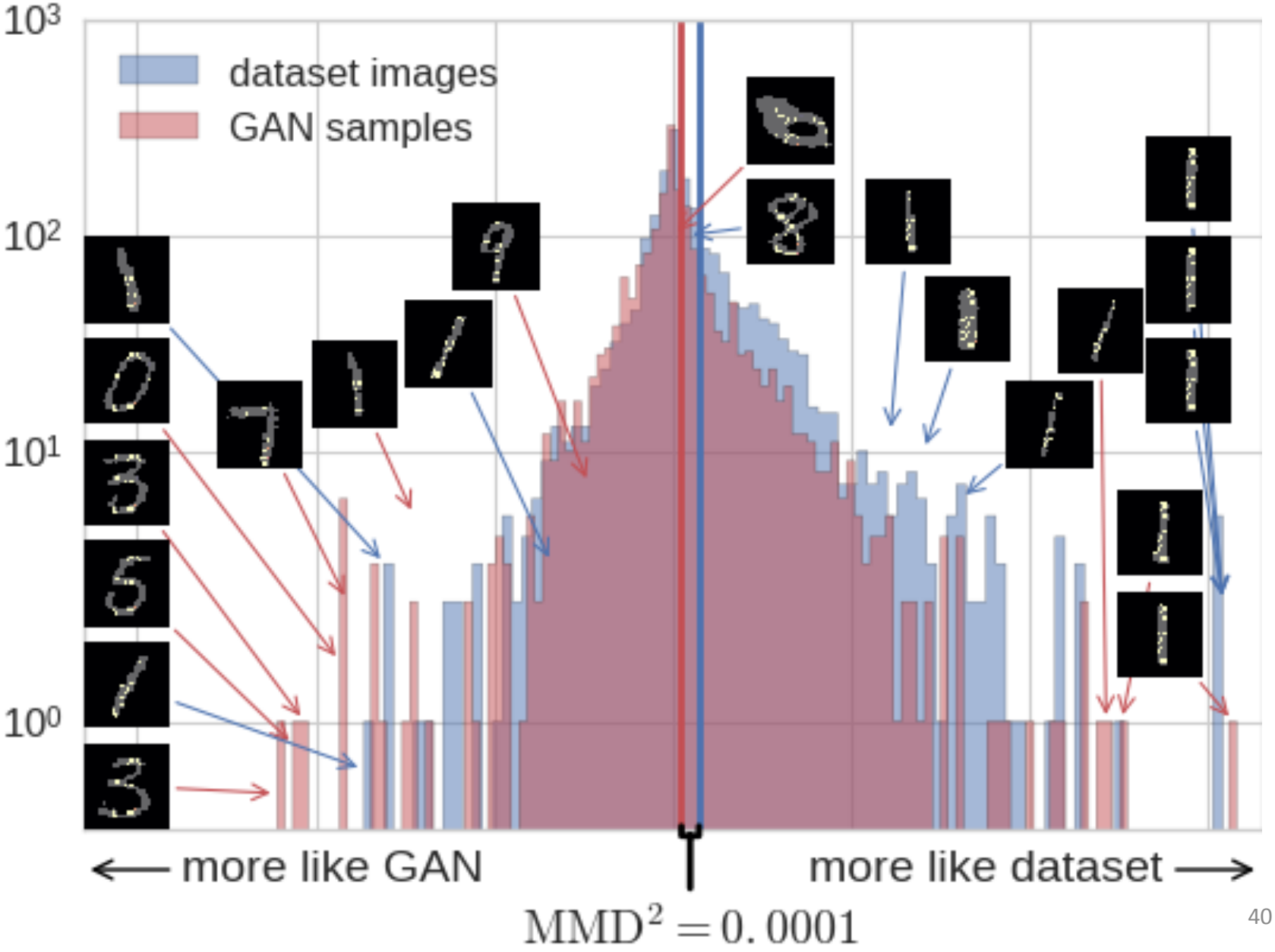
100 times: 2k samples, 1k permutations:

- ARD with $\hat{t}_U$: 98 times p = .000
  2 times p = .001
- Just bandwidth with $\hat{t}_U$:
  43 times p > .01, max = .135
- Median bandwidth:
  58 times p > .01
  3 times p = 1.000



ARD weights

Generative model criticism



GANS

Generative Adversarial Networks

Generator: $z \sim \text{Unif}\left([0,1]^{100}\right) \quad G(z) = x \sim P_G$

Trained to trick the classifier

Discriminator: $x \sim P_G$, $x' \sim P_{\text{data}}$ $\quad D(x) = \Pr\left(x \text{ came from } G\right)$

Classifier, trained on samples

With optimal discriminator, minimizes $2\text{JS}(P_G, P_{\text{data}}) - 2\log 2$

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

GAN example on MNIST

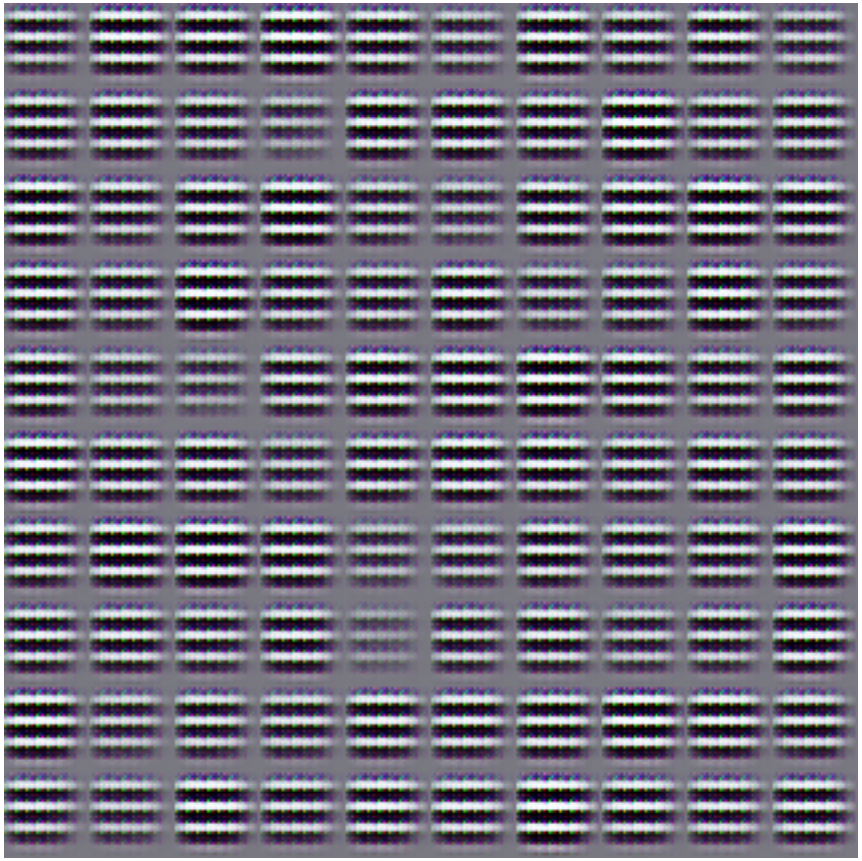
Epoch 1

P_G

P_{data}

GAN example on MNIST

Epoch 2

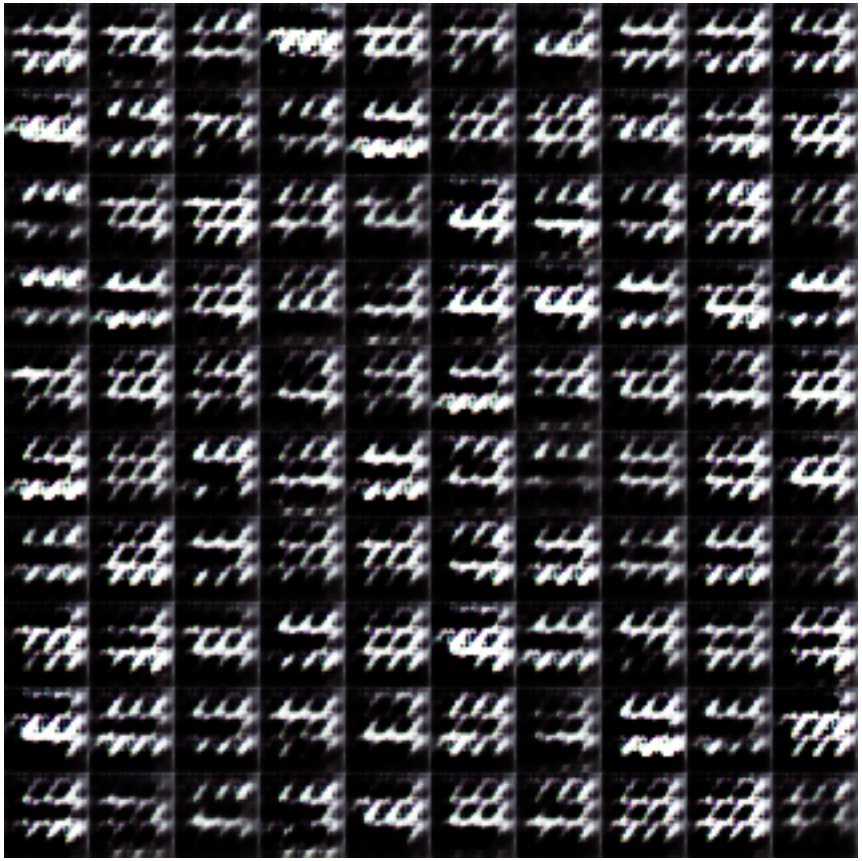


P_G

P_{data}

GAN example on MNIST

Epoch 3

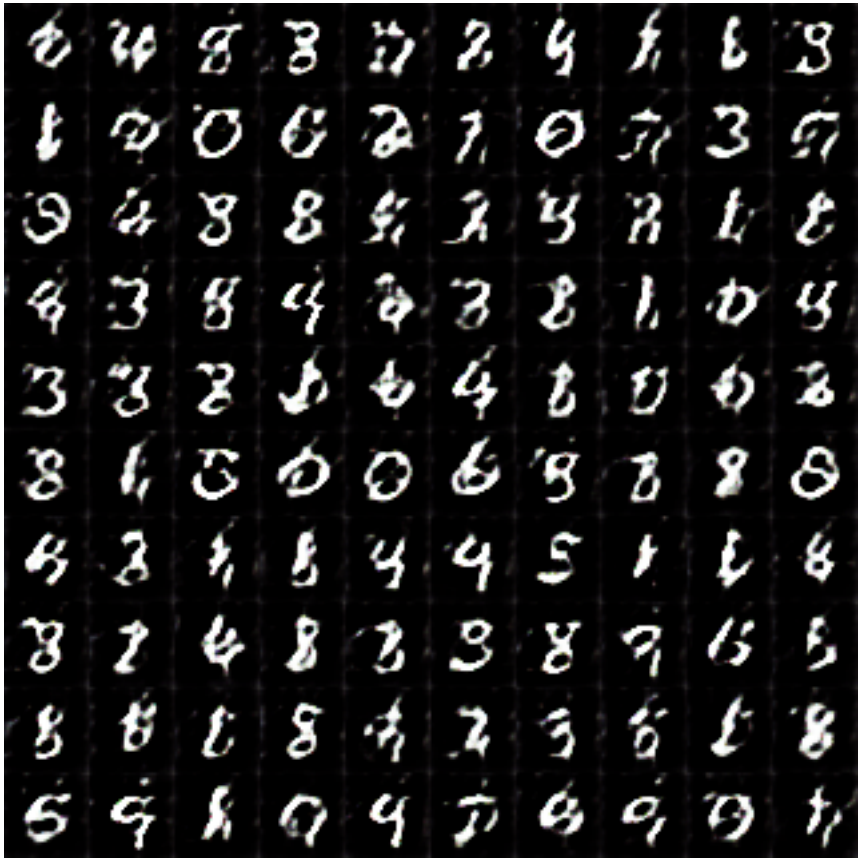


P_G

P_{data}

GAN example on MNIST

Epoch 4



P_G

P_{data}

GAN example on MNIST

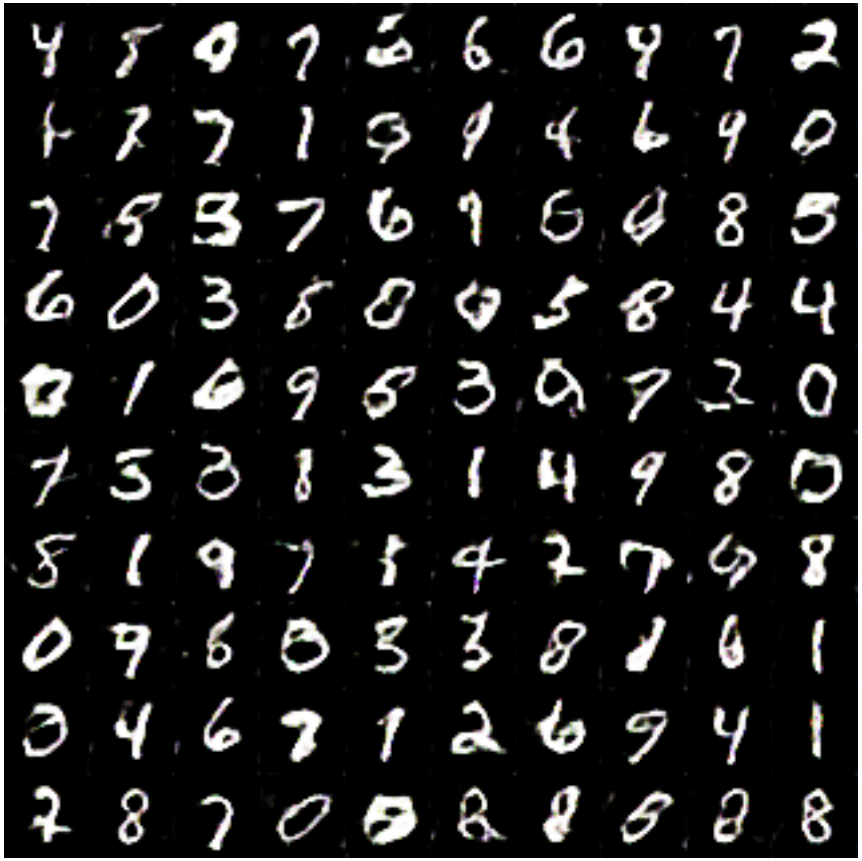
Epoch 5

P_G

P_{data}

GAN example on MNIST

Epoch 6

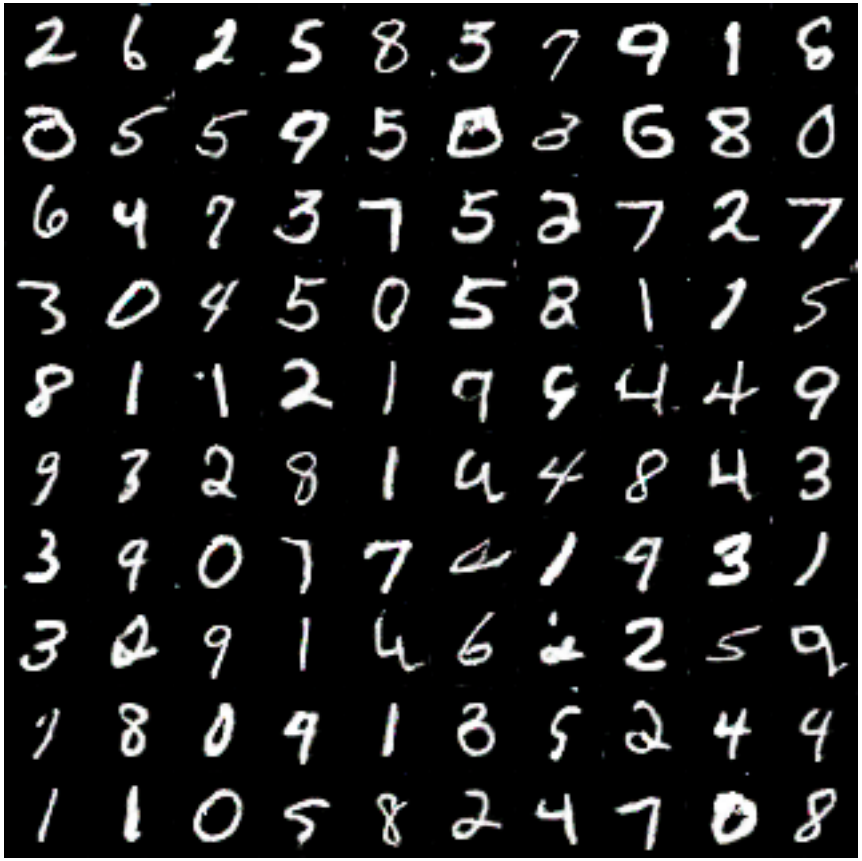


P_G

P_{data}

GAN example on MNIST

Epoch 100

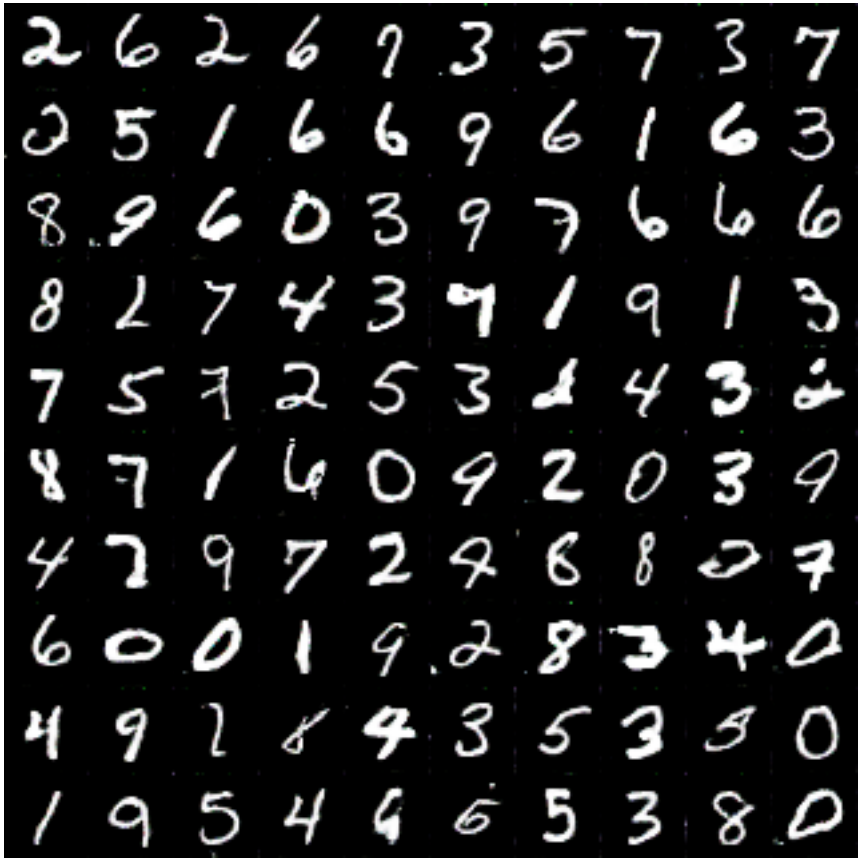


P_G

P_{data}

GAN example on MNIST

Epoch 200

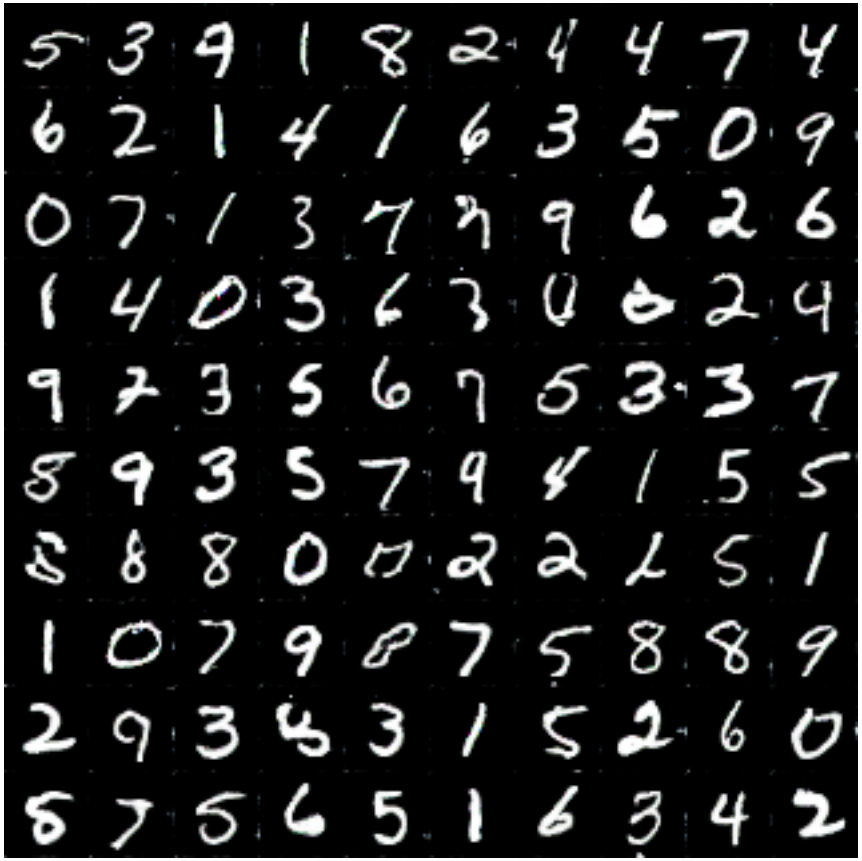


P_G

P_{data}

GAN example on MNIST

Epoch 500

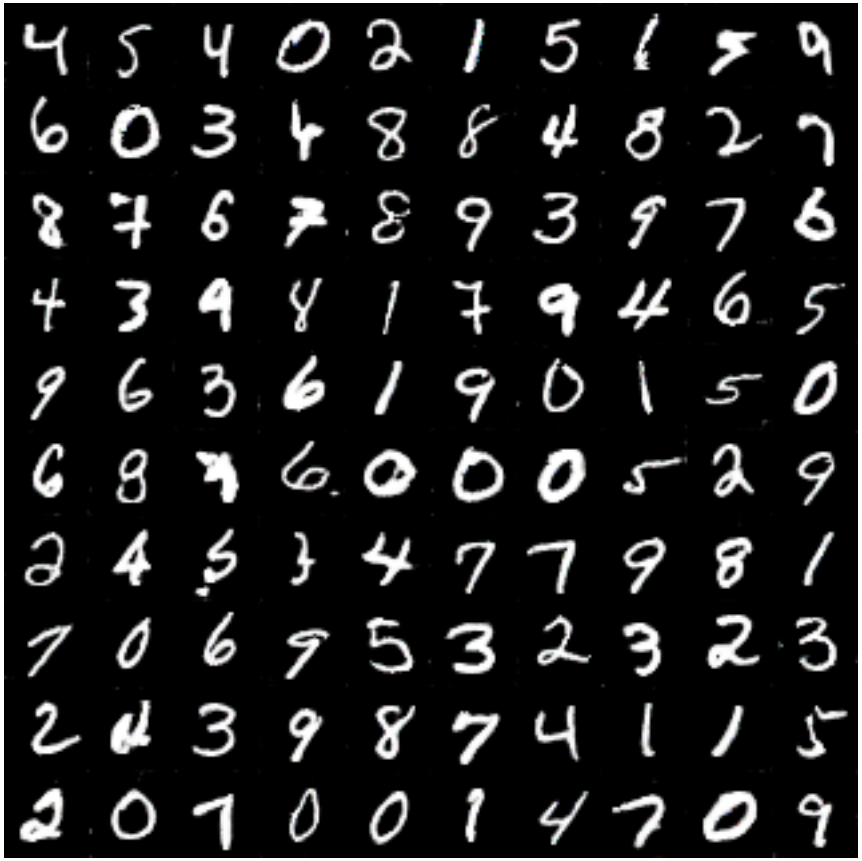


P_G

P_{data}

GAN example on MNIST

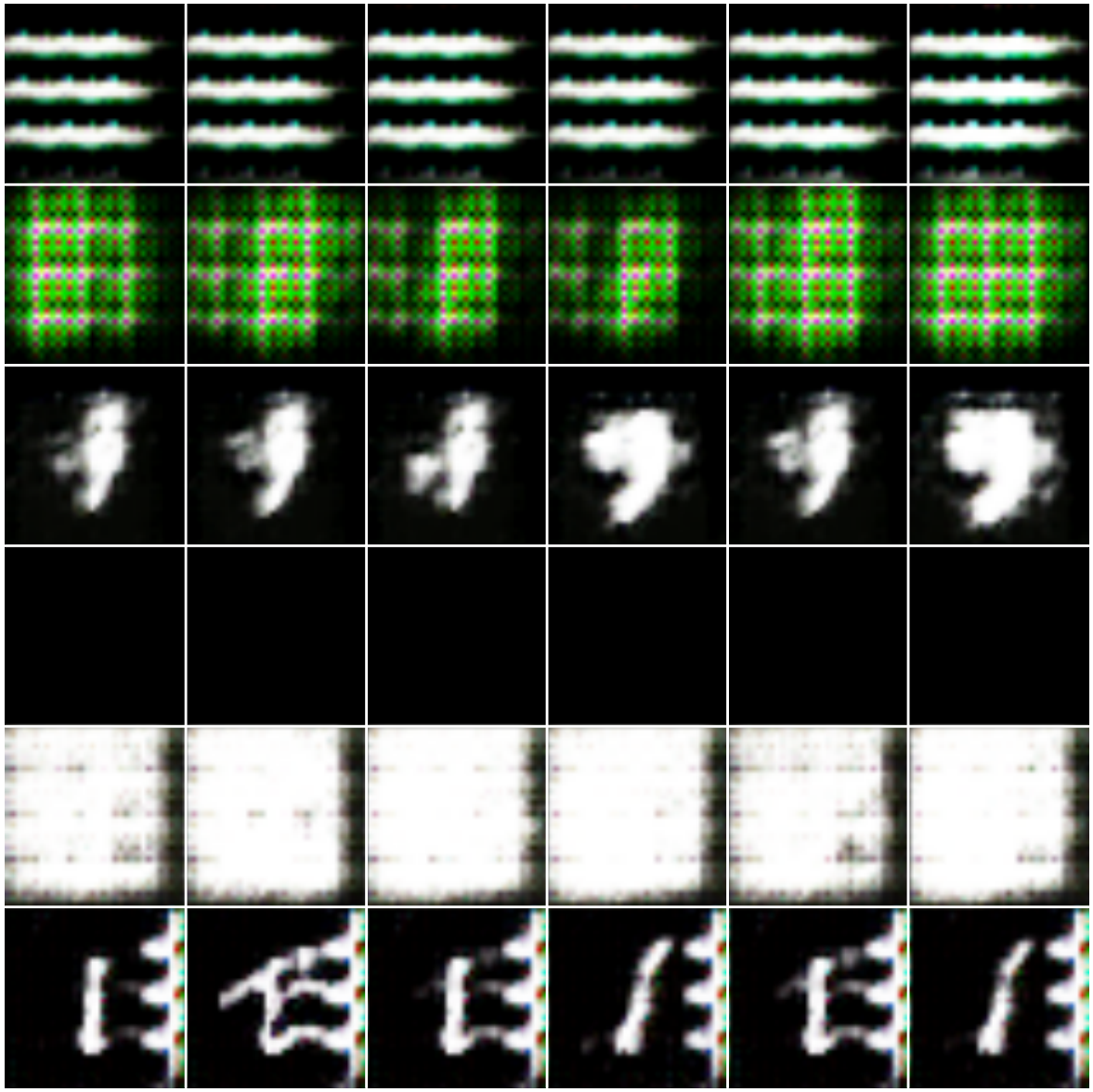
Epoch 900



P_G

P_{data}

Problems with GANs



Problems with GANs

Some possible reasons:

1. For a fixed discriminator, the optimal generator puts a point mass at one point the discriminator thinks is good.

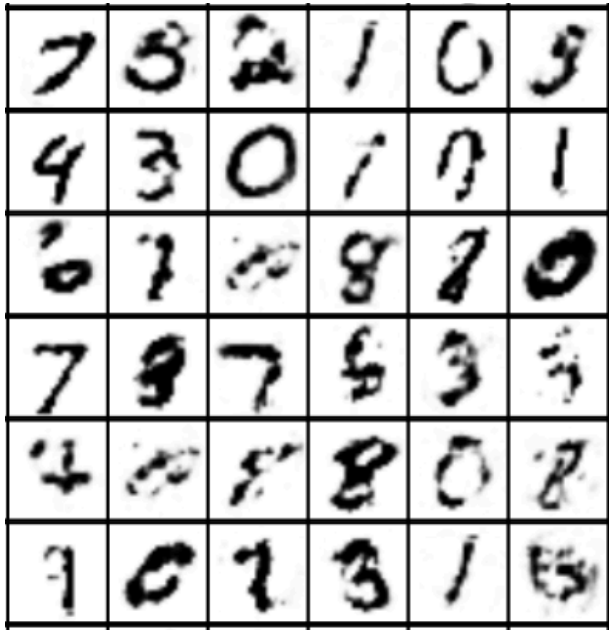
2. For a fixed generator, with probability 1 there is a perfect discriminator with flat gradients (Arjovsky and Bottou 2017).

Generative Moment-Matching Networks

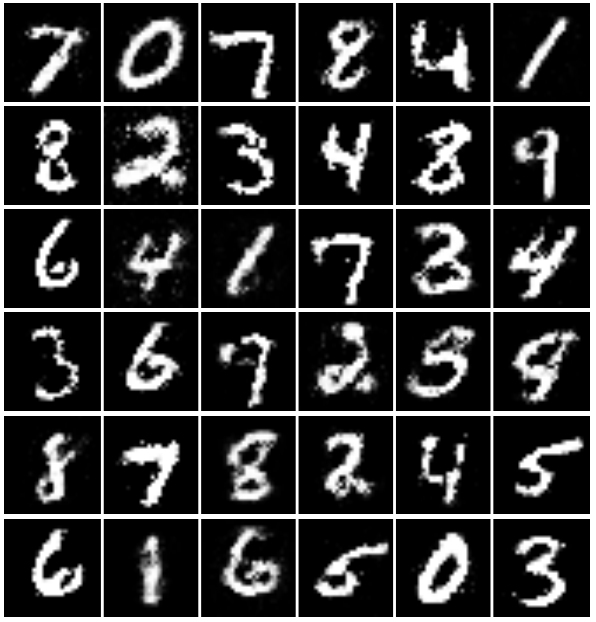
We can solve problem 1 by caring about *sets* at a time…

Replace the discriminator with an MMD two-sample test!

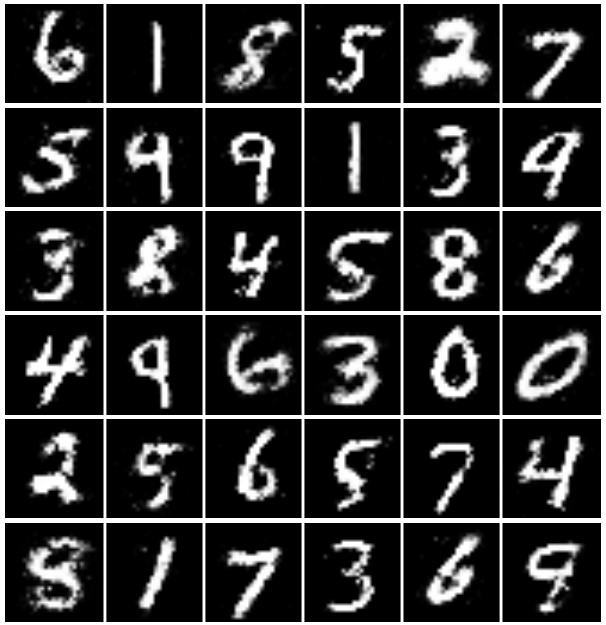
Li, Swersky, & Zemel, UAI 2015; Dziugaite, Roy, & Ghahramani, UAI 2015



Single Gaussian kernel, minimize MMD



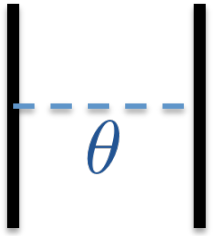
Mix of Gaussian kernels, minimize MMD



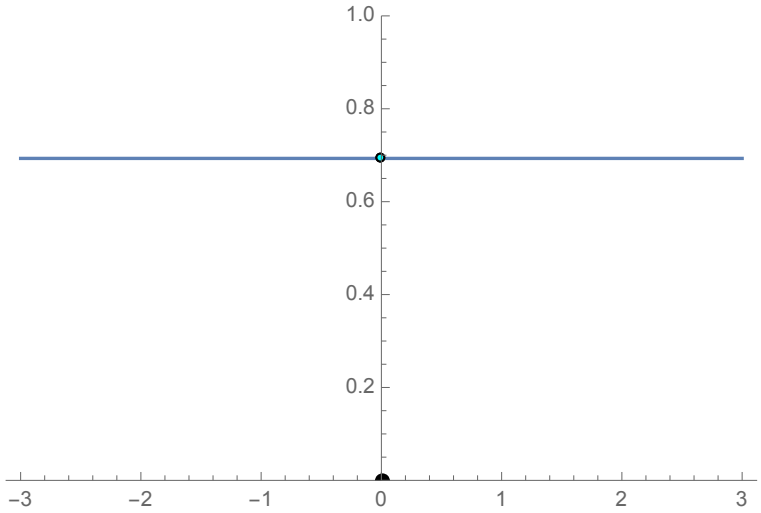
Mix of Gaussian kernels, minimize t statistic

Generative Moment-Matching Networks

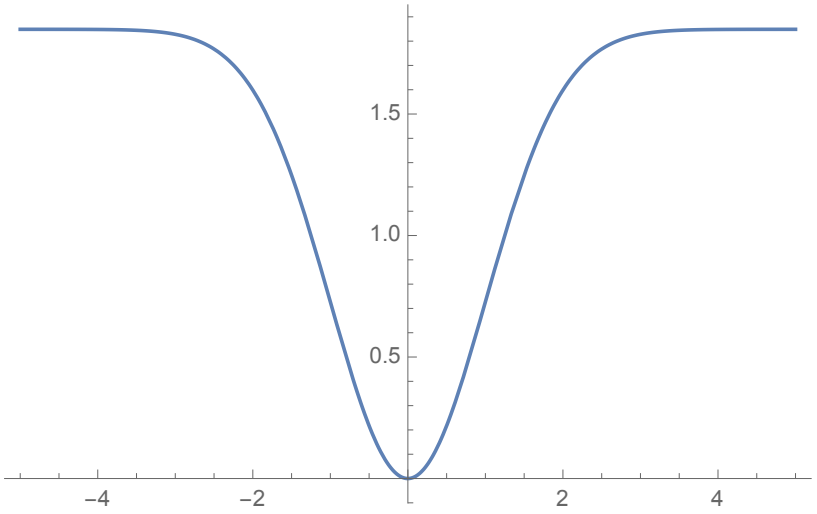
This also solves problem 2, for a fixed kernel:



$P_G \quad P_{\text{data}}$



$JS(P_G, P_{\text{data}})$



$\text{MMD}^2(P_G, P_{\text{data}})$

But once we try optimizing the kernel, it breaks again.

Generative Models and Model Criticism via Optimized Maximum Mean Discrepancy

Dougal J. Sutherland[*†] **Hsiao-Yu Tung**[†] **Heiko Strathmann**[*] **Soumyajit De**[*]
Aaditya Ramdas[‡] **Alex Smola**[†] **Arthur Gretton**[*]
[*] Gatsby Computational Neuroscience Unit, University College London
[†] School of Computer Science, Carnegie Mellon University
[‡] Departments of EECS and Statistics, University of California at Berkeley
dougal@gmail.com htung@cs.cmu.edu