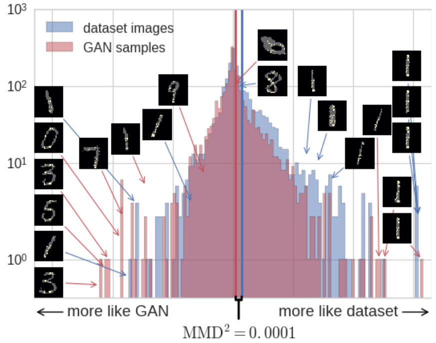
Are these datasets the same?
Learning kernels for efficient and fair two-sample tests

Danica J. Sutherland (she/her)

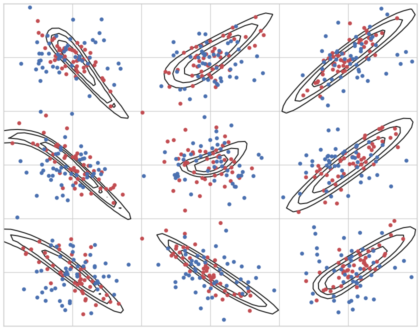
University of British Columbia (UBC) / Alberta Machine Intelligence Institute (Amii)

[ICLR-17]



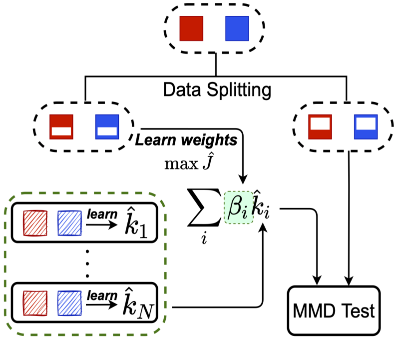
Hsiao-Yu (Fish) Tung
Heiko Strathmann
Soumyajit De
Aaditya Ramdas
Alex Smola
Arthur Gretton

[ICML-20]



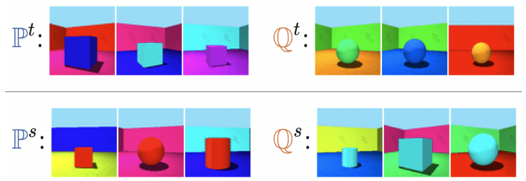
Feng Liu
Wenkai Xu
Jie Lu
Guangquan Zhang
Arthur Gretton

[NeurIPS-21]



Feng Liu
Wenkai Xu
Jie Lu

new!



Namrata Deka

TrustML - 15 Feb 2022

Data drift

- The textbook ML setting:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - Training error $\approx$ test error on $\mathbb{P}$
  - So our model should be good on more samples from $\mathbb{P}$

Data drift

- The textbook ML setting:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - Training error $\approx$ test error on $\mathbb{P}$
  - So our model should be good on more samples from $\mathbb{P}$
- Really:

Data drift

- The textbook ML setting:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - Training error $\approx$ test error on $\mathbb{P}$
  - So our model should be good on more samples from $\mathbb{P}$
- Really:
  - Train on "i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$"

Data drift

- The textbook ML setting:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - Training error $\approx$ test error on $\mathbb{P}$
  - So our model should be good on more samples from $\mathbb{P}$
- Really:
  - Train on "i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$"
  - Training error might vaguely correlate with test error on $\mathbb{P}$

Data drift

- The textbook ML setting:
  - Train on i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$
  - Training error $\approx$ test error on $\mathbb{P}$
  - So our model should be good on more samples from $\mathbb{P}$
- Really:
  - Train on "i.i.d. samples from some distribution, $X_i \sim \mathbb{P}$"
  - Training error might vaguely correlate with test error on $\mathbb{P}$
  - Deploy it on some distribution $\mathbb{Q}$, might be sort of like $\mathbb{P}$
    - and probably changes over time…

This talk

Based on samples $\{X_i\} \sim \mathbb{P}$ and $\{Y_j\} \sim \mathbb{Q}$:

- How is $\mathbb{P}$ different from $\mathbb{Q}$?

This talk

Based on samples $\{X_i\} \sim \mathbb{P}$ and $\{Y_j\} \sim \mathbb{Q}$:

- ~~How is $\mathbb{P}$ different from $\mathbb{Q}$?~~

This talk

Based on samples $\{X_i\} \sim \mathbb{P}$ and $\{Y_j\} \sim \mathbb{Q}$:

- ~~How is $\mathbb{P}$ different from $\mathbb{Q}$?~~
- Is $\mathbb{P}$ close enough to $\mathbb{Q}$ for our model?

This talk

Based on samples $\{X_i\} \sim \mathbb{P}$ and $\{Y_j\} \sim \mathbb{Q}$:

- ~~How is $\mathbb{P}$ different from $\mathbb{Q}$?~~
- ~~Is $\mathbb{P}$ close enough to $\mathbb{Q}$ for our model?~~

This talk

Based on samples $\{X_i\} \sim \mathbb{P}$ and $\{Y_j\} \sim \mathbb{Q}$:

- ~~How is $\mathbb{P}$ different from $\mathbb{Q}$?~~
- ~~Is $\mathbb{P}$ close enough to $\mathbb{Q}$ for our model?~~
- Is $\mathbb{P} = \mathbb{Q}$?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad\qquad Y \sim \mathbb{Q}$$

- Question: is $\mathbb{P} = \mathbb{Q}$?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?
- Do Brits have the same friend network types as Americans?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?
- Do Brits have the same friend network types as Americans?
- When does my laser agree with the one on Mars?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?
- Do Brits have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?
- Do Brits have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?
- Does presence of this protein affect DNA binding? [MMDiff2]

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?
- Do Brits have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?
- Does presence of this protein affect DNA binding? [`MMDiff2`]
- Do these `dob` and `birthday` columns mean the same thing?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?
- Do Brits have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?
- Does presence of this protein affect DNA binding? [`MMDiff2`]
- Do these `dob` and `birthday` columns mean the same thing?
- Does my generative model $\mathbb{Q}_\theta$ match $\mathbb{P}_{\mathbf{data}}$?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Do smokers/non-smokers get different cancers?
- Do Brits have the same friend network types as Americans?
- When does my laser agree with the one on Mars?
- Are storms in the 2000s different from storms in the 1800s?
- Does presence of this protein affect DNA binding? [MMDiff2]
- Do these `dob` and `birthday` columns mean the same thing?
- Does my generative model $\mathbb{Q}_\theta$ match $\mathbb{P}_{\text{data}}$?
- Independence testing: is $P(X, Y) = P(X)P(Y)$?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Question: is $\mathbb{P} = \mathbb{Q}$?

Two-sample testing

- Given samples from two unknown distributions

$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Question: is $\mathbb{P} = \mathbb{Q}$?
- Hypothesis testing approach:

$$H_0 : \mathbb{P} = \mathbb{Q} \qquad H_1 : \mathbb{P} \neq \mathbb{Q}$$

Two-sample testing

- Given samples from two unknown distributions

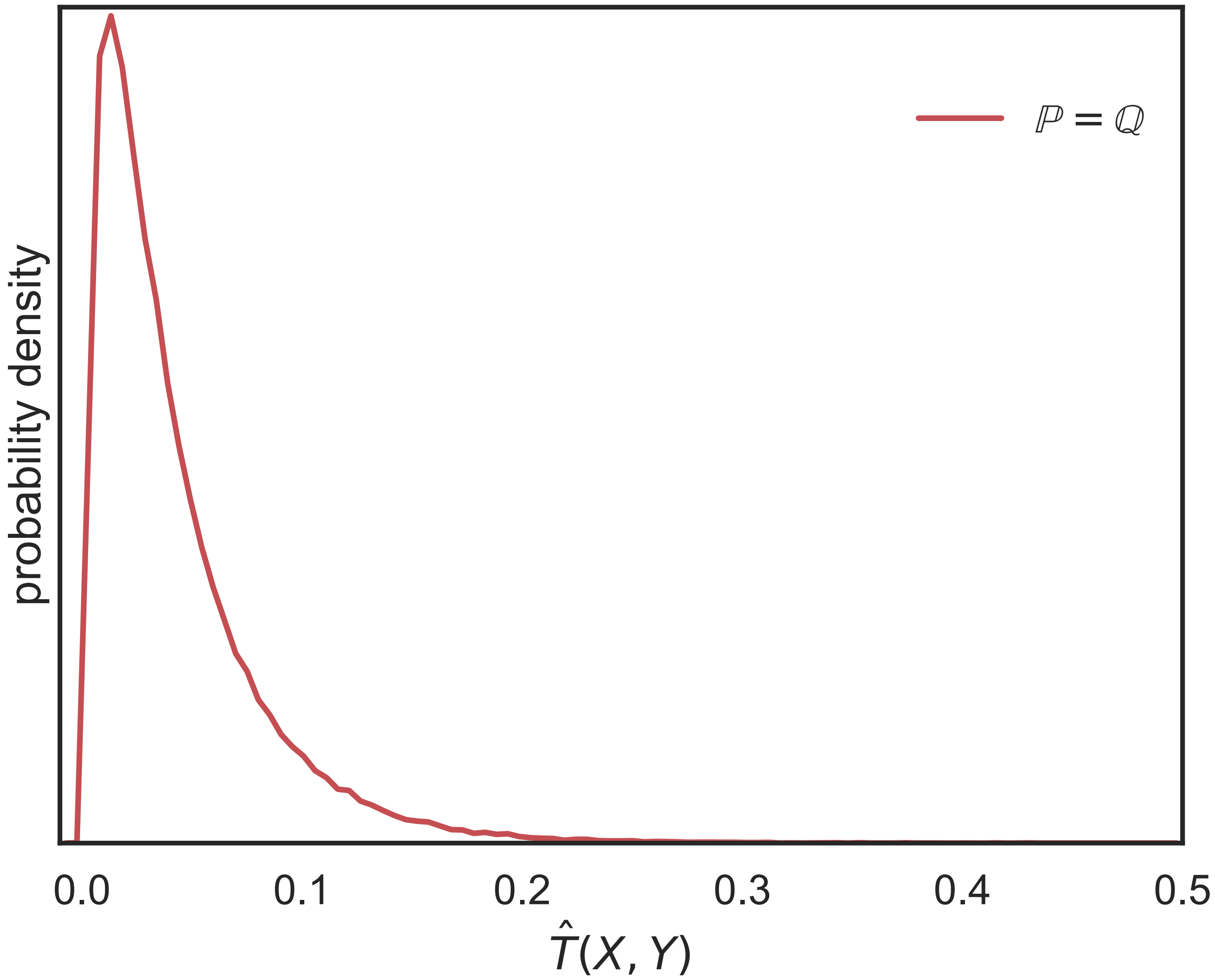
$$X \sim \mathbb{P} \qquad Y \sim \mathbb{Q}$$

- Question: is $\mathbb{P} = \mathbb{Q}$?
- Hypothesis testing approach:

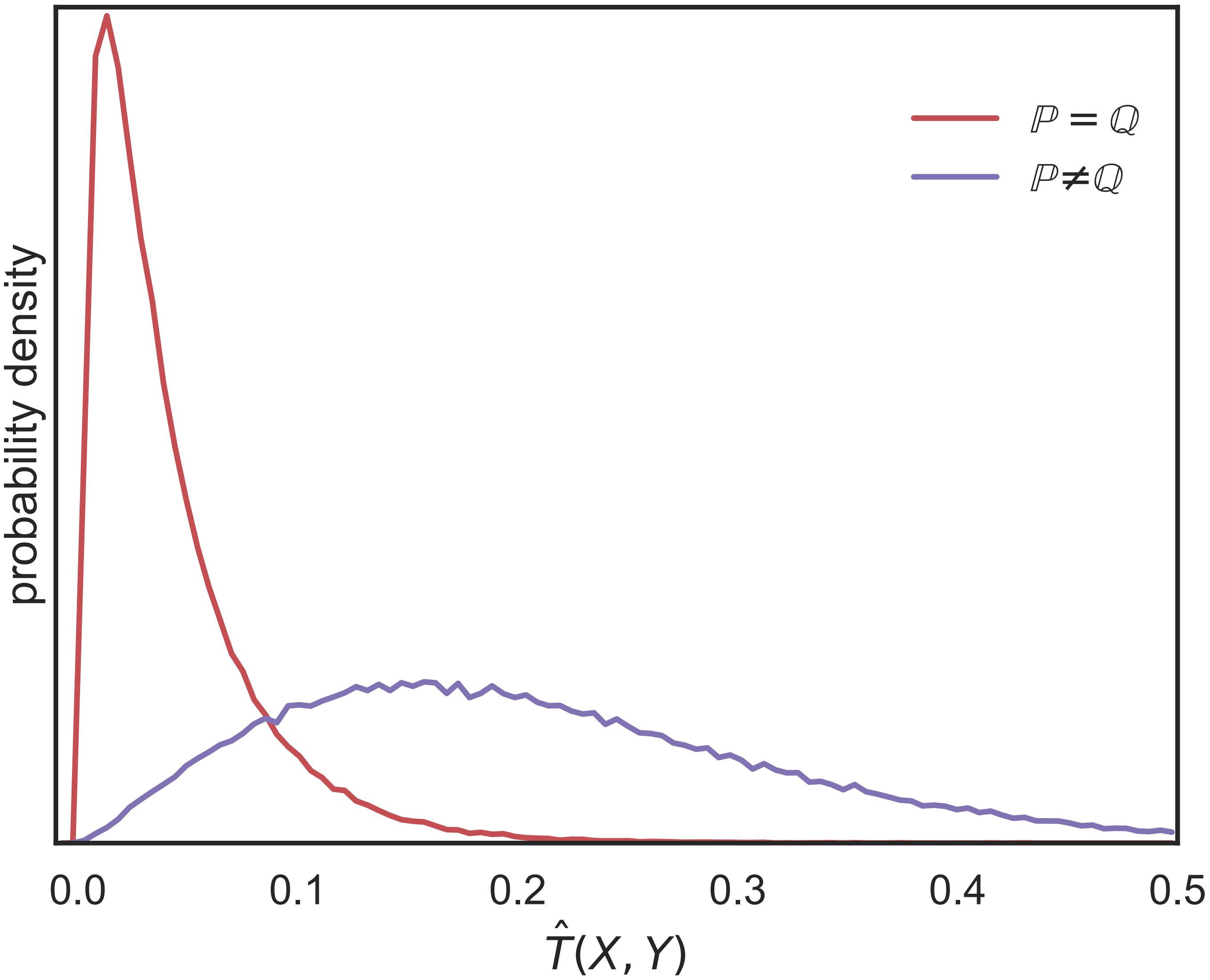
$$H_0 : \mathbb{P} = \mathbb{Q} \qquad H_1 : \mathbb{P} \neq \mathbb{Q}$$

- Reject H_0 if test statistic $\hat{T}(X, Y) > c_\alpha$

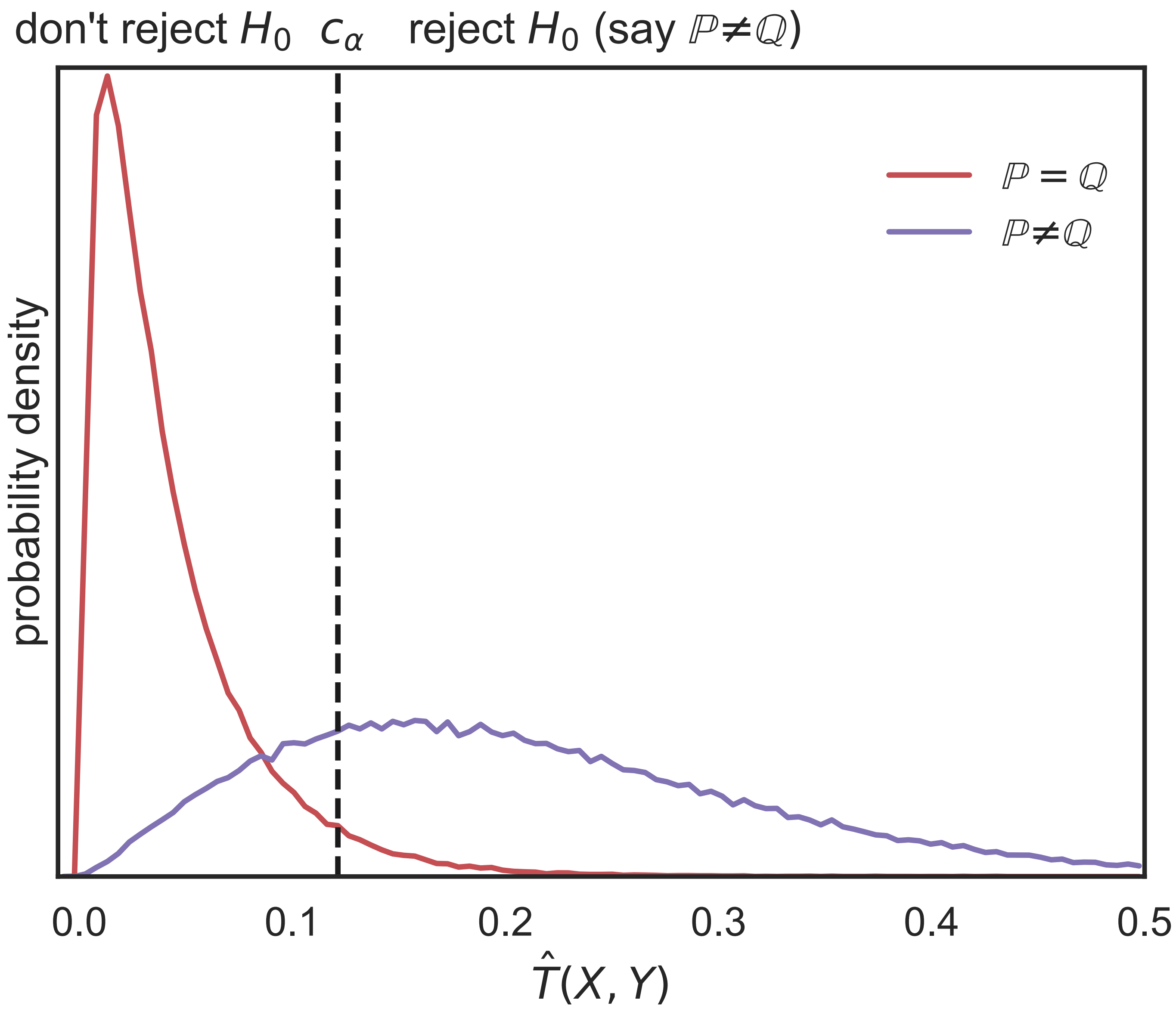
What's a hypothesis test again?



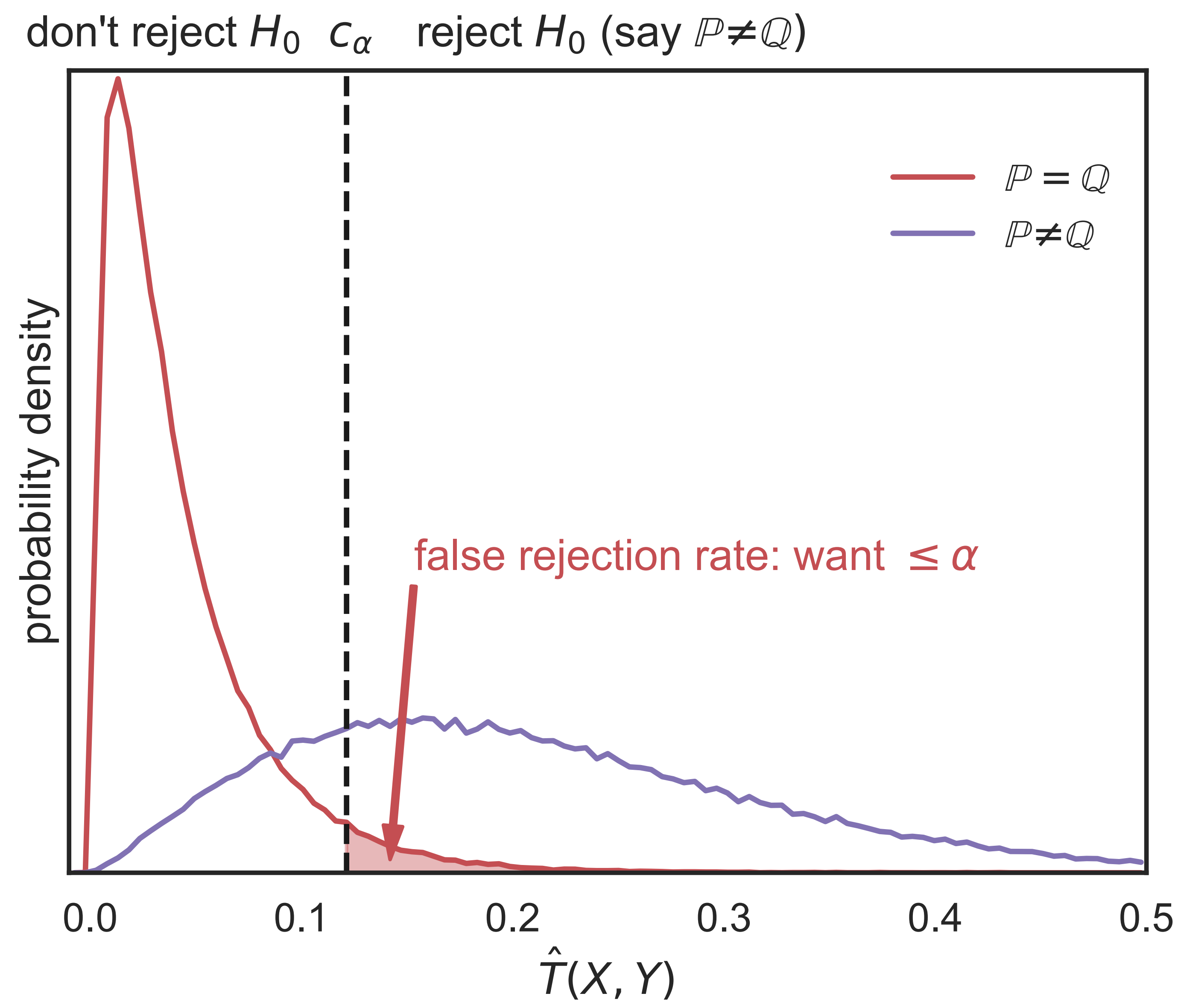
What's a hypothesis test again?



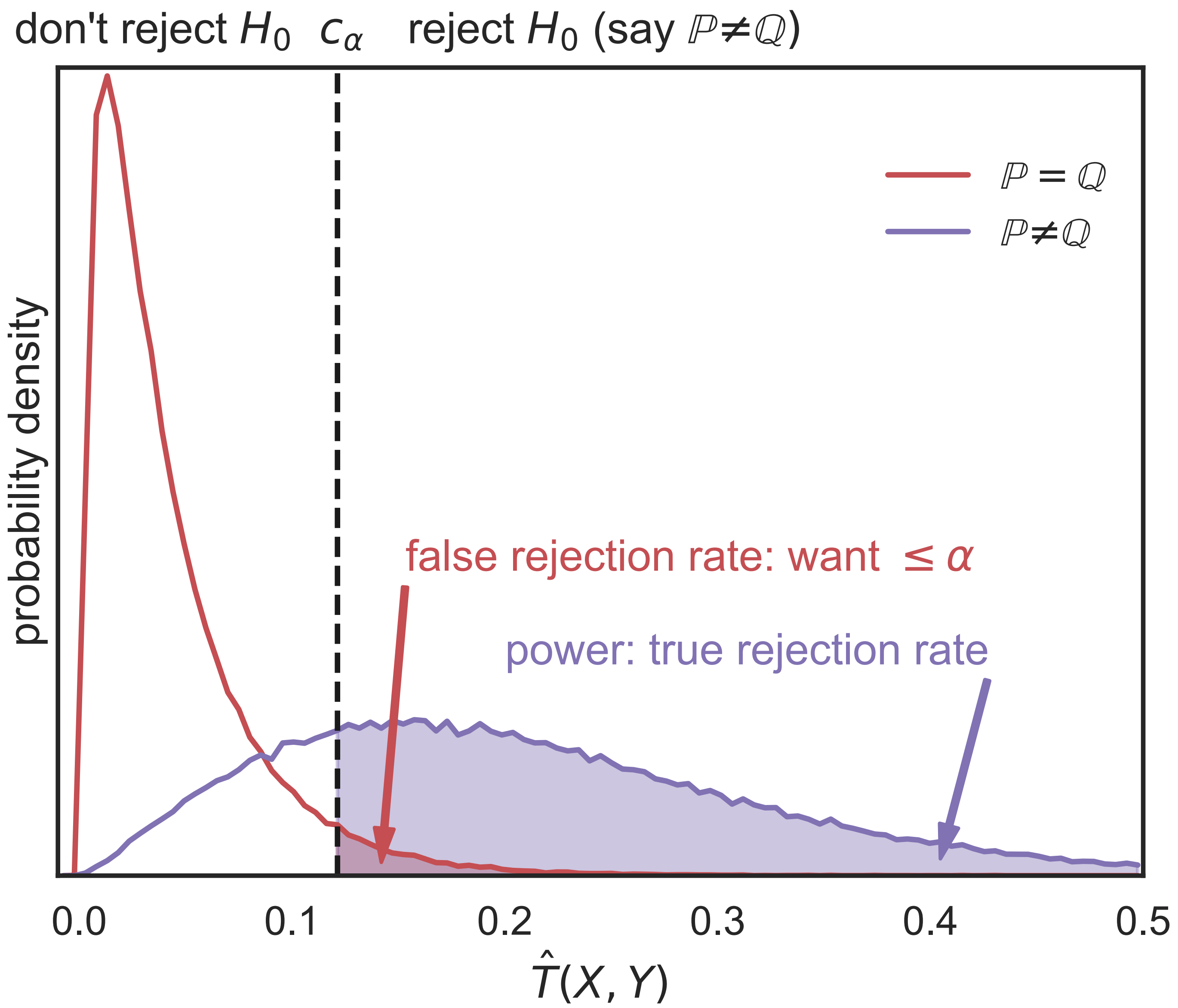
What's a hypothesis test again?



What's a hypothesis test again?



What's a hypothesis test again?



Permutation testing to find c_α

Need $\Pr_{H_0}\left(\hat{T}(X, Y) > c_\alpha\right) \leq \alpha$

$X_1 \quad X_2 \quad X_3 \quad X_4 \quad X_5 \qquad Y_1 \quad Y_2 \quad Y_3 \quad Y_4 \quad Y_5$

c_α: $1-\alpha$th quantile of $\left\{ \qquad \right\}$

Permutation testing to find c_α

Need $\Pr_{H_0}\left(\hat{T}(X, Y) > c_\alpha\right) \leq \alpha$

X_1 X_2 X_3 X_4 X_5 Y_1 Y_2 Y_3 Y_4 Y_5

c_α: $1-\alpha$th quantile of $\left\{ \qquad \right\}$

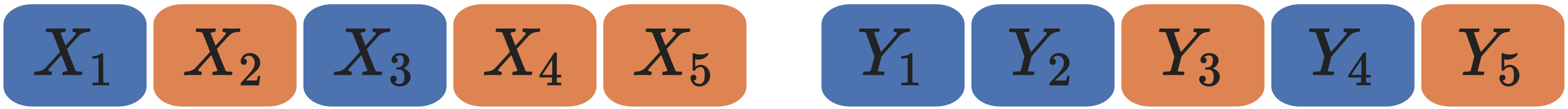
Permutation testing to find c_α

Need $\Pr_{H_0}\left(\hat{T}(X, Y) > c_\alpha\right) \le \alpha$

c_α: $1-\alpha$th quantile of $\left\{\hat{T}(\tilde{X}_1, \tilde{Y}_1), \qquad \right\}$

Permutation testing to find c_α

Need $\Pr_{H_0}\left(\hat{T}(X, Y) > c_\alpha\right) \leq \alpha$



c_α: $1 - \alpha$th quantile of $\left\{\hat{T}(\tilde{X}_1, \tilde{Y}_1),\ \hat{T}(\tilde{X}_2, \tilde{Y}_2), \qquad \right\}$

Permutation testing to find c_α

Need $\Pr_{H_0}\left(\hat{T}(X, Y) > c_\alpha\right) \leq \alpha$

$X_1 \quad X_2 \quad X_3 \quad X_4 \quad X_5 \qquad Y_1 \quad Y_2 \quad Y_3 \quad Y_4 \quad Y_5$

c_α: $1-\alpha$th quantile of $\left\{\hat{T}(\tilde{X}_1, \tilde{Y}_1),\ \hat{T}(\tilde{X}_2, \tilde{Y}_2),\ \cdots\right\}$

Need a $\hat{T}$ to estimate the difference between distributions, based on samples

Need a $\hat{T}$ to estimate the difference between distributions, based on samples

Our choice of $\hat{T}$: the **Maximum Mean Discrepancy** (MMD)

Need a $\hat{T}$ to estimate the difference between distributions, based on samples

Our choice of $\hat{T}$: the **Maximum Mean Discrepancy** (MMD)

This is a *kernel-based* distance between distributions

What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \mathrm{sign}(f(x))$, $f(x) = w^\mathsf{T}\,(x, 1)$

What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \text{sign}(f(x))$, $f(x) = w^\mathsf{T}(x, 1)$

What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \operatorname{sign}(f(x))$, $f(x) = w^{\mathsf{T}} (x, 1)$

What's a kernel again?

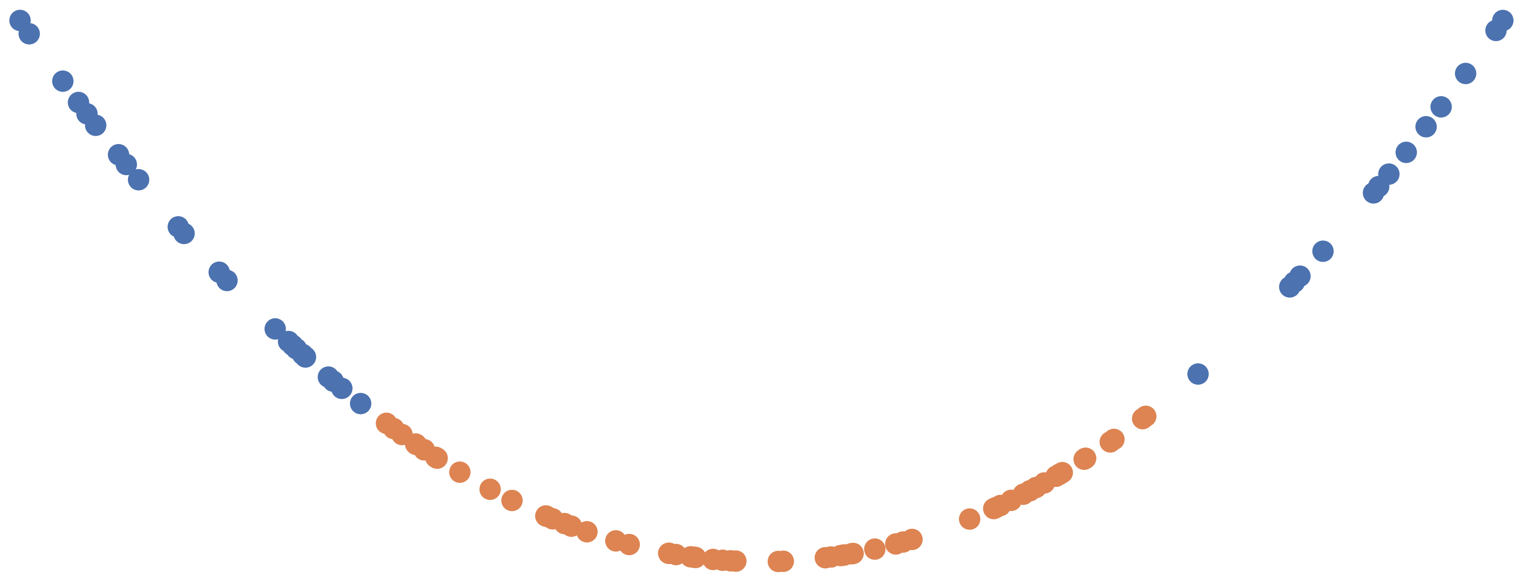
- Linear classifiers: $\hat{y}(x) = \operatorname{sign}(f(x))$, $f(x) = w^\mathsf{T}(x, 1)$
- Use a "richer" x:

$$f(x) = w^\mathsf{T}\left(x, x^2, 1\right) = w^\mathsf{T}\phi(x)$$

What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \text{sign}(f(x))$, $f(x) = w^\mathsf{T}\,(x, 1)$
- Use a "richer" x:

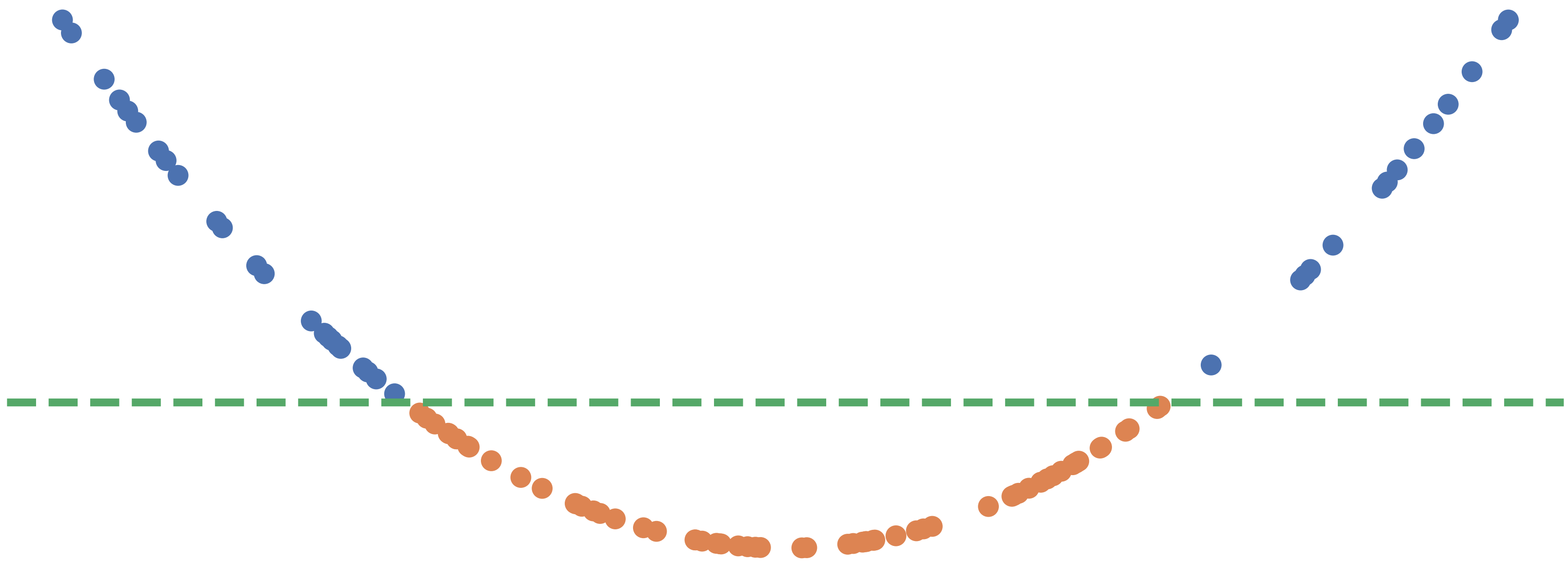
$$f(x) = w^\mathsf{T}\left(x, x^2, 1\right) = w^\mathsf{T}\phi(x)$$



What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \operatorname{sign}(f(x))$, $f(x) = w^\mathsf{T}(x, 1)$
- Use a "richer" x:

$$f(x) = w^\mathsf{T}\left(x, x^2, 1\right) = w^\mathsf{T}\phi(x)$$



What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \operatorname{sign}(f(x))$, $f(x) = w^\mathsf{T} \, (x, 1)$
- Use a "richer" x:

$$f(x) = w^\mathsf{T} \left(x, x^2, 1\right) = w^\mathsf{T} \phi(x)$$

- Can avoid explicit $\phi(x)$; instead $k(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}$

What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \operatorname{sign}(f(x))$, $f(x) = w^\mathsf{T} \, (x, 1)$
- Use a "richer" x:

$$f(x) = w^\mathsf{T} \left(x, x^2, 1\right) = w^\mathsf{T} \phi(x)$$

- Can avoid explicit $\phi(x)$; instead $k(x, y) = \langle \phi(x), \phi(y) \rangle_{\mathcal{H}}$
- "Kernelized" algorithms access data only through $k(x, y)$

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

What's a kernel again?

- Linear classifiers: $\hat{y}(x) = \operatorname{sign}(f(x))$, $f(x) = w^\mathsf{T} (x, 1)$
- Use a "richer" x:

$$f(x) = w^\mathsf{T} \left(x, x^2, 1\right) = w^\mathsf{T} \phi(x)$$

- Can avoid explicit $\phi(x)$; instead $k(x, y) = \langle \phi(x), \phi(y) \rangle_\mathcal{H}$
- "Kernelized" algorithms access data only through $k(x, y)$

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

- $\|f\|_\mathcal{H} = \sqrt{\alpha^\mathsf{T} K \alpha}$ gives kernel notion of smoothness

Reproducing Kernel Hilbert Space (RKHS)

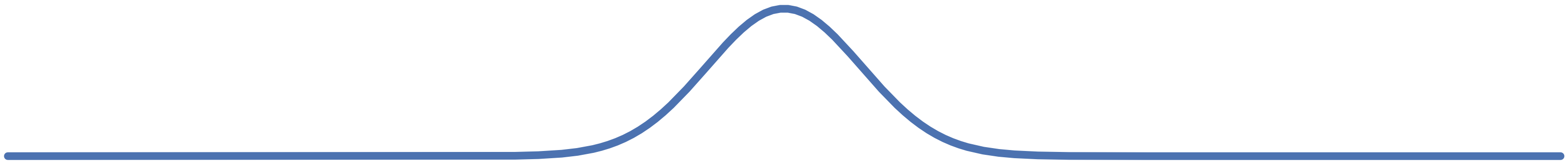
- Ex: Gaussian RBF

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

Reproducing Kernel Hilbert Space (RKHS)

- Ex: Gaussian RBF

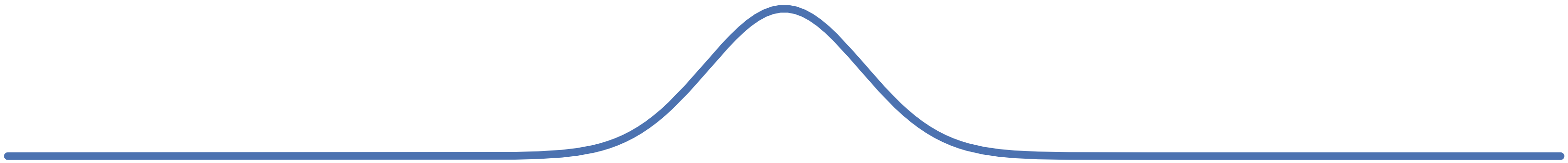
$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right)$$



Reproducing Kernel Hilbert Space (RKHS)

- Ex: Gaussian RBF / exponentiated quadratic / squared exponential / …

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

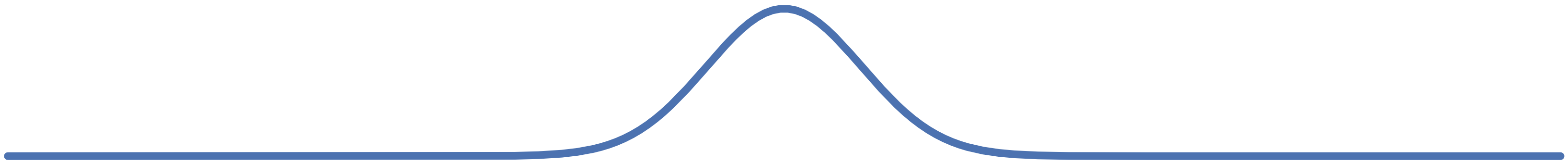


Reproducing Kernel Hilbert Space (RKHS)

- Ex: Gaussian RBF / exponentiated quadratic / squared exponential / …

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

- Some functions with small $\|f\|_{\mathcal{H}}$:



Reproducing Kernel Hilbert Space (RKHS)

- Ex: Gaussian RBF / exponentiated quadratic / squared exponential / …

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

- Some functions with small $\|f\|_{\mathcal{H}}$:

Reproducing Kernel Hilbert Space (RKHS)

- Ex: Gaussian RBF / exponentiated quadratic / squared exponential / ...

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

- Some functions with small $\|f\|_{\mathcal{H}}$:

Reproducing Kernel Hilbert Space (RKHS)

- Ex: Gaussian RBF / exponentiated quadratic / squared exponential / ...

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

- Some functions with small $\|f\|_{\mathcal{H}}$:

Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

The sup is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(t)]$

Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathbb{P}}[f(X)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(Y)]$$

The sup is achieved by $f(t) \propto \mathbb{E}_{X\sim\mathbb{P}}[f(t)] - \mathbb{E}_{Y\sim\mathbb{Q}}[f(t)]$

Maximum Mean Discrepancy (MMD)

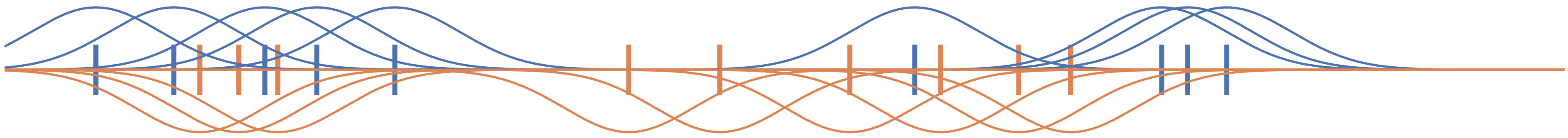
$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \le 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

The sup is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(t)]$

Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{X \sim \mathbb{P}}[f(X)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(Y)]$$

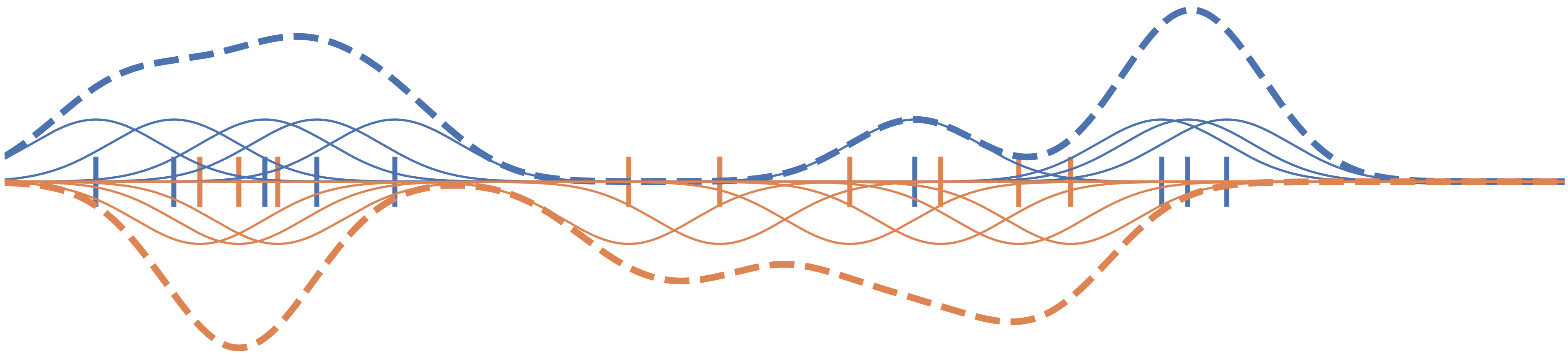
The sup is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(t)]$



Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

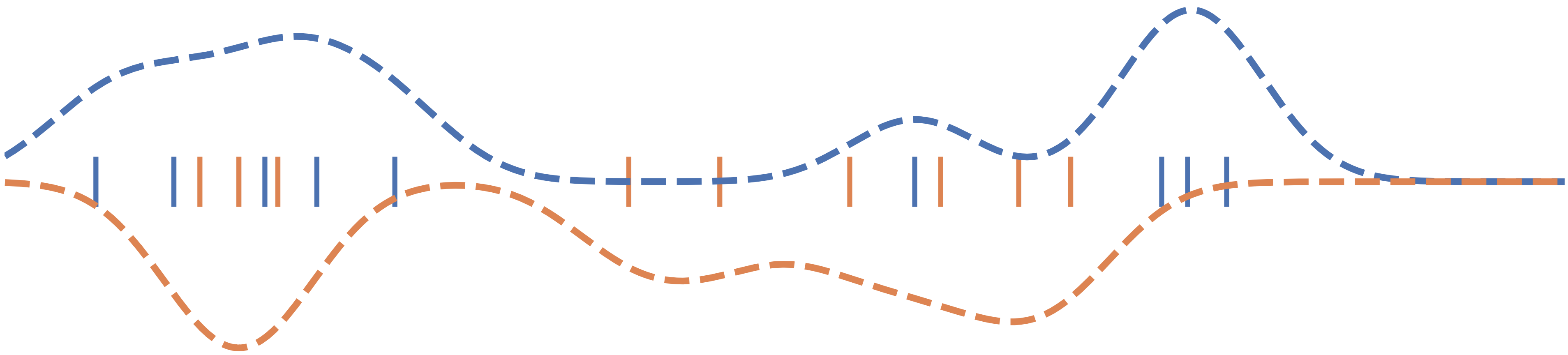
The sup is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(t)]$



Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

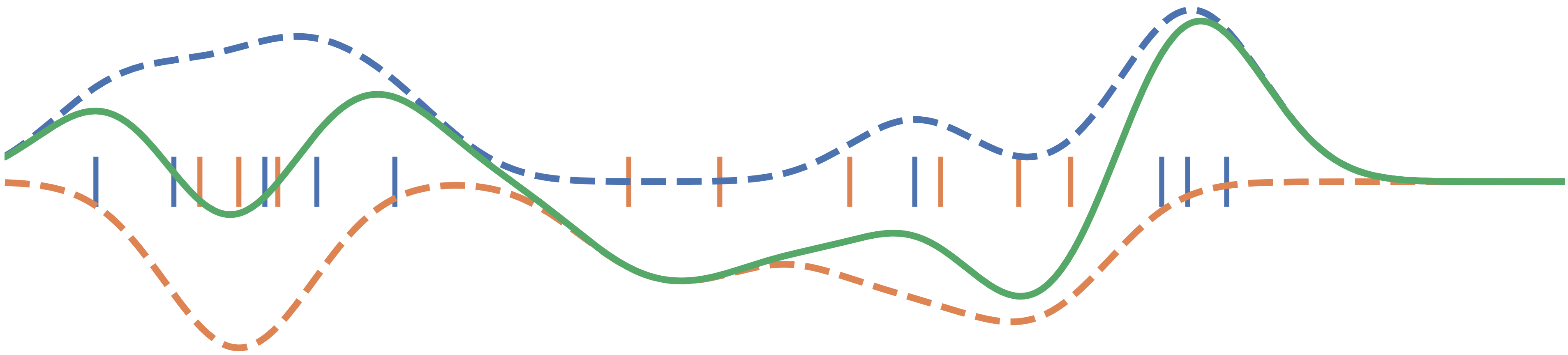
The sup is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(t)]$



Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

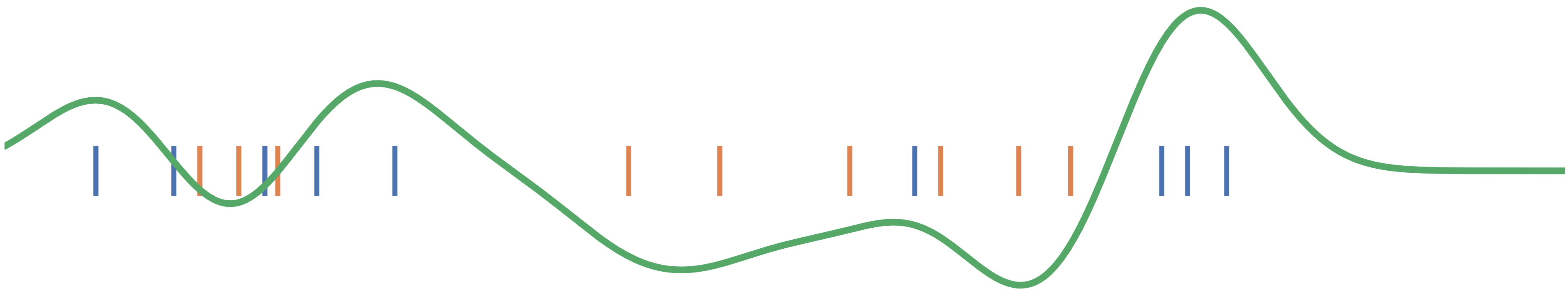
The sup is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(t)]$



Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \le 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

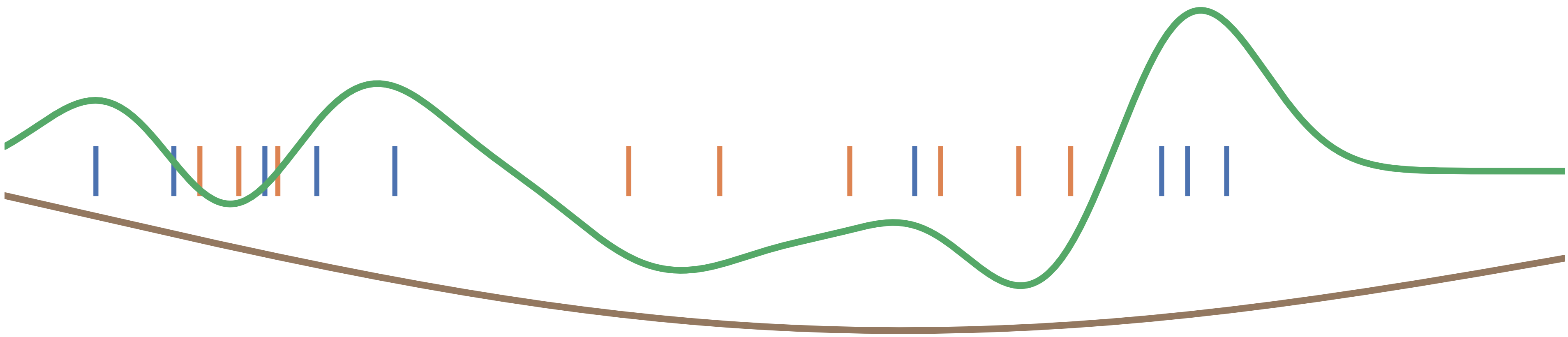
The sup is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(t)]$



Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

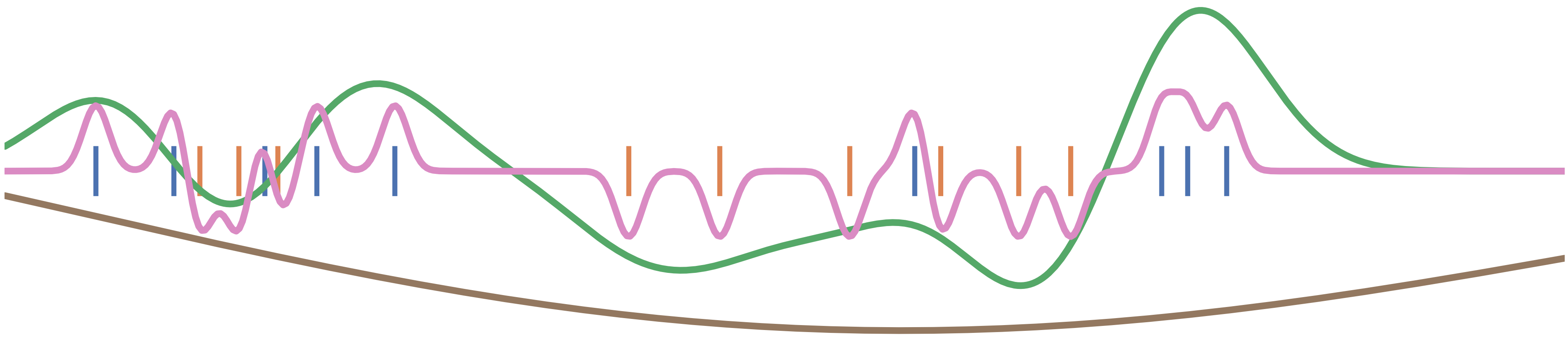
The sup is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(t)]$



Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \le 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

The sup is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(t)]$

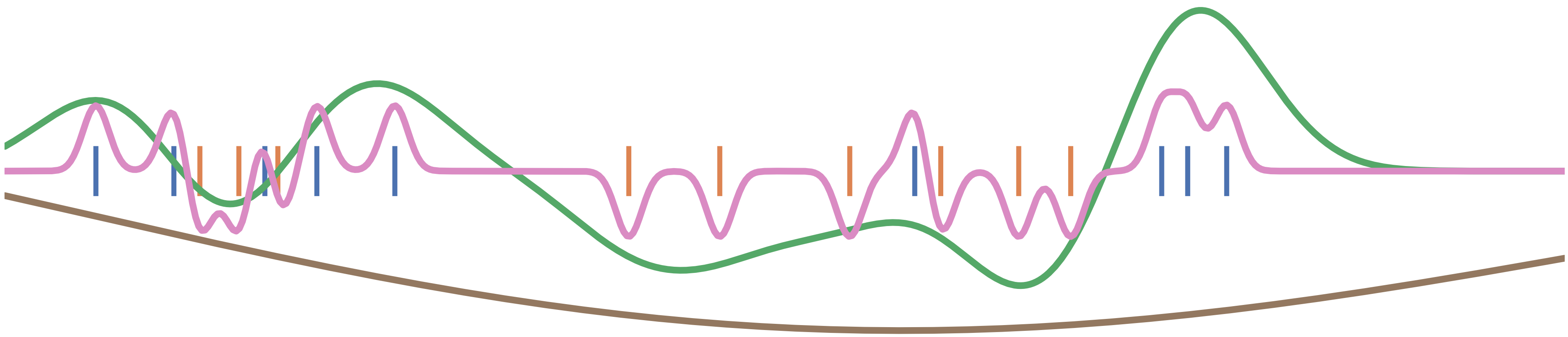


Maximum Mean Discrepancy (MMD)

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)]$$

The sup is achieved by $f(t) \propto \mathbb{E}_{X \sim \mathbb{P}}[f(t)] - \mathbb{E}_{Y \sim \mathbb{Q}}[f(t)]$

$$\mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}) = \mathop{\mathbb{E}}_{\substack{X, X' \sim \mathbb{P} \\ Y, Y' \sim \mathbb{Q}}} [k(X, X') + k(Y, Y') - 2k(X, Y)]$$



$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{\boldsymbol{X} \sim \mathbb{P}}[f(\boldsymbol{X})] - \mathbb{E}_{\boldsymbol{Y} \sim \mathbb{Q}}[f(\boldsymbol{Y})]$$

$$
\begin{aligned}
\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) &= \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)] \\
&= \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[\langle f, \varphi(X) \rangle_{\mathcal{H}}] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[\langle f, \varphi(Y) \rangle_{\mathcal{H}}]
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) &= \sup_{\|f\|_{\mathcal{H}} \le 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)] \\
&= \sup_{\|f\|_{\mathcal{H}} \le 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[\langle f, \varphi(X) \rangle_{\mathcal{H}}] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[\langle f, \varphi(Y) \rangle_{\mathcal{H}}] \\
&= \sup_{\|f\|_{\mathcal{H}} \le 1} \left\langle f, \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[\varphi(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[\varphi(Y)] \right\rangle_{\mathcal{H}}
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{MMD}_k(\mathbb{P},\mathbb{Q}) &= \sup_{\|f\|_{\mathcal{H}}\le 1} \mathop{\mathbb{E}}_{X\sim\mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y\sim\mathbb{Q}}[f(Y)] \\
&= \sup_{\|f\|_{\mathcal{H}}\le 1} \mathop{\mathbb{E}}_{X\sim\mathbb{P}}[\langle f,\varphi(X)\rangle_{\mathcal{H}}] - \mathop{\mathbb{E}}_{Y\sim\mathbb{Q}}[\langle f,\varphi(Y)\rangle_{\mathcal{H}}] \\
&= \sup_{\|f\|_{\mathcal{H}}\le 1} \left\langle f, \mathop{\mathbb{E}}_{X\sim\mathbb{P}}[\varphi(X)] - \mathop{\mathbb{E}}_{Y\sim\mathbb{Q}}[\varphi(Y)] \right\rangle_{\mathcal{H}} \\
&= \sup_{\|f\|_{\mathcal{H}}\le 1} \left\langle f, \mu_{\mathbb{P}}^k - \mu_{\mathbb{Q}}^k \right\rangle_{\mathcal{H}}
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) &= \sup_{\|f\|_{\mathcal{H}} \le 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)] \\
&= \sup_{\|f\|_{\mathcal{H}} \le 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[\langle f, \varphi(X) \rangle_{\mathcal{H}}] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[\langle f, \varphi(Y) \rangle_{\mathcal{H}}] \\
&= \sup_{\|f\|_{\mathcal{H}} \le 1} \left\langle f, \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[\varphi(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[\varphi(Y)] \right\rangle_{\mathcal{H}} \\
&= \sup_{\|f\|_{\mathcal{H}} \le 1} \left\langle f, \mu_{\mathbb{P}}^k - \mu_{\mathbb{Q}}^k \right\rangle_{\mathcal{H}} = \left\| \mu_{\mathbb{P}}^k - \mu_{\mathbb{Q}}^k \right\|_{\mathcal{H}}
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) &= \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[f(Y)] \\
&= \sup_{\|f\|_{\mathcal{H}} \leq 1} \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[\langle f, \varphi(X) \rangle_{\mathcal{H}}] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[\langle f, \varphi(Y) \rangle_{\mathcal{H}}] \\
&= \sup_{\|f\|_{\mathcal{H}} \leq 1} \left\langle f, \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[\varphi(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[\varphi(Y)] \right\rangle_{\mathcal{H}} \\
&= \sup_{\|f\|_{\mathcal{H}} \leq 1} \left\langle f, \mu_{\mathbb{P}}^k - \mu_{\mathbb{Q}}^k \right\rangle_{\mathcal{H}} = \left\| \mu_{\mathbb{P}}^k - \mu_{\mathbb{Q}}^k \right\|_{\mathcal{H}}
\end{aligned}
$$

$$
\langle \mu_{\mathbb{P}}^k, \mu_{\mathbb{Q}}^k \rangle_{\mathcal{H}} = \mathop{\mathbb{E}}_{\substack{X \sim \mathbb{P} \\ Y \sim \mathbb{Q}}} \langle \varphi(X), \varphi(Y) \rangle_{\mathcal{H}} = \mathop{\mathbb{E}}_{\substack{X \sim \mathbb{P} \\ Y \sim \mathbb{Q}}} k(X, Y)
$$

$$
\begin{aligned}
\mathrm{MMD}_k(\mathbb{P},\mathbb{Q}) &= \sup_{\|f\|_{\mathcal{H}}\le 1} \mathop{\mathbb{E}}_{X\sim\mathbb{P}}[f(X)] - \mathop{\mathbb{E}}_{Y\sim\mathbb{Q}}[f(Y)] \\
&= \sup_{\|f\|_{\mathcal{H}}\le 1} \mathop{\mathbb{E}}_{X\sim\mathbb{P}}[\langle f,\varphi(X)\rangle_{\mathcal{H}}] - \mathop{\mathbb{E}}_{Y\sim\mathbb{Q}}[\langle f,\varphi(Y)\rangle_{\mathcal{H}}] \\
&= \sup_{\|f\|_{\mathcal{H}}\le 1} \left\langle f, \mathop{\mathbb{E}}_{X\sim\mathbb{P}}[\varphi(X)] - \mathop{\mathbb{E}}_{Y\sim\mathbb{Q}}[\varphi(Y)] \right\rangle_{\mathcal{H}} \\
&= \sup_{\|f\|_{\mathcal{H}}\le 1} \left\langle f, \mu_{\mathbb{P}}^k - \mu_{\mathbb{Q}}^k \right\rangle_{\mathcal{H}} = \left\| \mu_{\mathbb{P}}^k - \mu_{\mathbb{Q}}^k \right\|_{\mathcal{H}}
\end{aligned}
$$

$$
\langle \mu_{\mathbb{P}}^k, \mu_{\mathbb{Q}}^k \rangle_{\mathcal{H}} = \mathop{\mathbb{E}}_{\substack{X\sim\mathbb{P}\\ Y\sim\mathbb{Q}}} \langle \varphi(X), \varphi(Y) \rangle_{\mathcal{H}} = \mathop{\mathbb{E}}_{\substack{X\sim\mathbb{P}\\ Y\sim\mathbb{Q}}} k(X,Y)
$$

$$
\mathrm{MMD}^2(\mathbb{P},\mathbb{Q}) = \mathop{\mathbb{E}}_{\substack{X,X'\sim\mathbb{P}\\ Y,Y'\sim\mathbb{Q}}} [k(X,X') + k(Y,Y') - 2k(X,Y)]
$$

Estimating MMD

$$\mathrm{MMD}_k^2(\mathbb{P}, \mathbb{Q}) = \mathop{\mathbb{E}}_{X, X' \sim \mathbb{P}}[k(X, X')] + \mathop{\mathbb{E}}_{Y, Y' \sim \mathbb{Q}}[k(Y, Y')] - 2 \mathop{\mathbb{E}}_{\substack{X \sim \mathbb{P} \\ Y \sim \mathbb{Q}}}[k(X, Y)]$$

Estimating MMD

$$\mathrm{MMD}_k^2(\mathbb{P}, \mathbb{Q}) = \mathop{\mathbb{E}}_{X, X' \sim \mathbb{P}}[k(X, X')] + \mathop{\mathbb{E}}_{Y, Y' \sim \mathbb{Q}}[k(Y, Y')] - 2 \mathop{\mathbb{E}}_{\substack{X \sim \mathbb{P} \\ Y \sim \mathbb{Q}}}[k(X, Y)]$$

$$\widehat{\mathrm{MMD}}_k^2(X, Y) = \mathrm{mean}(K_{XX}) + \mathrm{mean}(K_{YY}) - 2\,\mathrm{mean}(K_{XY})$$

Estimating MMD

$$\mathrm{MMD}_k^2(\mathbb{P}, \mathbb{Q}) = \mathop{\mathbb{E}}_{X, X' \sim \mathbb{P}}[k(X, X')] + \mathop{\mathbb{E}}_{Y, Y' \sim \mathbb{Q}}[k(Y, Y')] - 2 \mathop{\mathbb{E}}_{\substack{X \sim \mathbb{P} \\ Y \sim \mathbb{Q}}}[k(X, Y)]$$

$$\widehat{\mathrm{MMD}}_k^2(X, Y) = \mathrm{mean}(K_{XX}) + \mathrm{mean}(K_{YY}) - 2\,\mathrm{mean}(K_{XY})$$

K_{XX}

	1.0	0.2	0.6
	0.2	1.0	0.5
	0.6	0.5	1.0

Estimating MMD

$$\mathrm{MMD}_k^2(\mathbb{P},\mathbb{Q}) = \mathop{\mathbb{E}}_{X,X'\sim\mathbb{P}}[k(X,X')] + \mathop{\mathbb{E}}_{Y,Y'\sim\mathbb{Q}}[k(Y,Y')] - 2\mathop{\mathbb{E}}_{\substack{X\sim\mathbb{P}\\ Y\sim\mathbb{Q}}}[k(X,Y)]$$

$$\widehat{\mathrm{MMD}}_k^2(X,Y) = \mathrm{mean}(K_{XX}) + \mathrm{mean}(K_{YY}) - 2\,\mathrm{mean}(K_{XY})$$

K_{XX}

1.0	0.2	0.6
0.2	1.0	0.5
0.6	0.5	1.0

K_{YY}

1.0	0.8	0.7
0.8	1.0	0.6
0.7	0.6	1.0

Estimating MMD

$$\mathrm{MMD}_k^2(\mathbb{P}, \mathbb{Q}) = \mathbb{E}_{X, X' \sim \mathbb{P}}[k(X, X')] + \mathbb{E}_{Y, Y' \sim \mathbb{Q}}[k(Y, Y')] - 2 \mathbb{E}_{\substack{X \sim \mathbb{P} \\ Y \sim \mathbb{Q}}}[k(X, Y)]$$

$$\widehat{\mathrm{MMD}}_k^2(X, Y) = \mathrm{mean}(K_{XX}) + \mathrm{mean}(K_{YY}) - 2\,\mathrm{mean}(K_{XY})$$

K_{XX}

1.0	0.2	0.6
0.2	1.0	0.5
0.6	0.5	1.0

K_{YY}

1.0	0.8	0.7
0.8	1.0	0.6
0.7	0.6	1.0

K_{XY}

0.3	0.1	0.2
0.2	0.3	0.3
0.2	0.1	0.4

MMD as feature matching

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \left\| \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[\varphi(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[\varphi(Y)] \right\|_{\mathcal{H}}$$

- $\varphi : \mathcal{X} \to \mathcal{H}$ is the *feature map* for $k(x, y) = \langle \varphi(x), \varphi(y) \rangle$

MMD as feature matching

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \left\| \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[\varphi(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[\varphi(Y)] \right\|_{\mathcal{H}}$$

- $\varphi : \mathcal{X} \to \mathcal{H}$ is the *feature map* for $k(x, y) = \langle \varphi(x), \varphi(y) \rangle$
- If $k(x, y) = x^\mathsf{T} y$, $\varphi(x) = x$, then the MMD is distance between means

MMD as feature matching

$$\mathrm{MMD}_k(\mathbb{P}, \mathbb{Q}) = \left\| \mathop{\mathbb{E}}_{X \sim \mathbb{P}}[\varphi(X)] - \mathop{\mathbb{E}}_{Y \sim \mathbb{Q}}[\varphi(Y)] \right\|_{\mathcal{H}}$$

- $\varphi : \mathcal{X} \to \mathcal{H}$ is the *feature map* for $k(x, y) = \langle \varphi(x), \varphi(y) \rangle$
- If $k(x, y) = x^\mathsf{T} y$, $\varphi(x) = x$, then the MMD is distance between means
- Many kernels: **infinite-dimensional** $\mathcal{H}$

MMD-based tests

- If k is *characteristic*, $\mathrm{MMD}(\mathbb{P}, \mathbb{Q}) = 0$ iff $\mathbb{P} = \mathbb{Q}$
- Efficient permutation testing for $\widehat{\mathrm{MMD}}(X, Y)$

MMD-based tests

- If k is *characteristic*, $\mathrm{MMD}(\mathbb{P}, \mathbb{Q}) = 0$ iff $\mathbb{P} = \mathbb{Q}$
- Efficient permutation testing for $\widehat{\mathrm{MMD}}(X, Y)$
  - H_0: $n\widehat{\mathrm{MMD}}^2$ converges in distribution
  - H_1: $\sqrt{n}(\widehat{\mathrm{MMD}}^2 - \mathrm{MMD}^2)$ asymptotically normal

MMD-based tests

- If k is *characteristic*, $\mathrm{MMD}(\mathbb{P}, \mathbb{Q}) = 0$ iff $\mathbb{P} = \mathbb{Q}$
- Efficient permutation testing for $\widehat{\mathrm{MMD}}(X, Y)$
  - H_0: $n\widehat{\mathrm{MMD}}^2$ converges in distribution
  - H_1: $\sqrt{n}(\widehat{\mathrm{MMD}}^2 - \mathrm{MMD}^2)$ asymptotically normal
- Any characteristic kernel gives consistent test

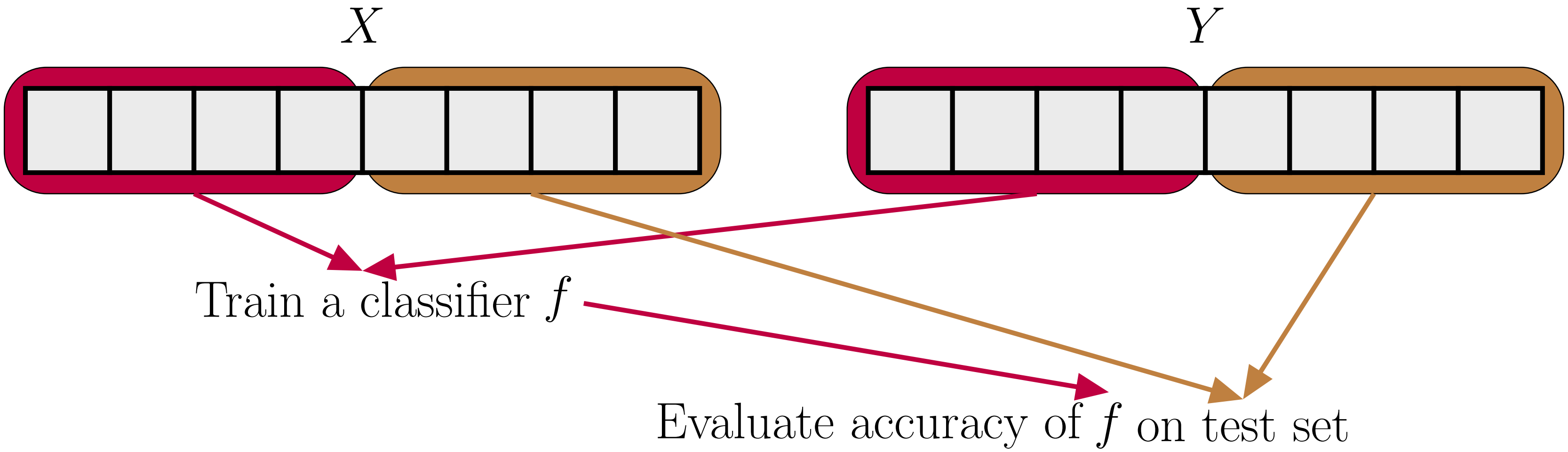
MMD-based tests

- If k is *characteristic*, $\mathrm{MMD}(\mathbb{P},\mathbb{Q}) = 0$ iff $\mathbb{P} = \mathbb{Q}$
- Efficient permutation testing for $\widehat{\mathrm{MMD}}(X, Y)$
  - H_0: $n\widehat{\mathrm{MMD}}^2$ converges in distribution
  - H_1: $\sqrt{n}(\widehat{\mathrm{MMD}}^2 - \mathrm{MMD}^2)$ asymptotically normal
- Any characteristic kernel gives consistent test…eventually

MMD-based tests

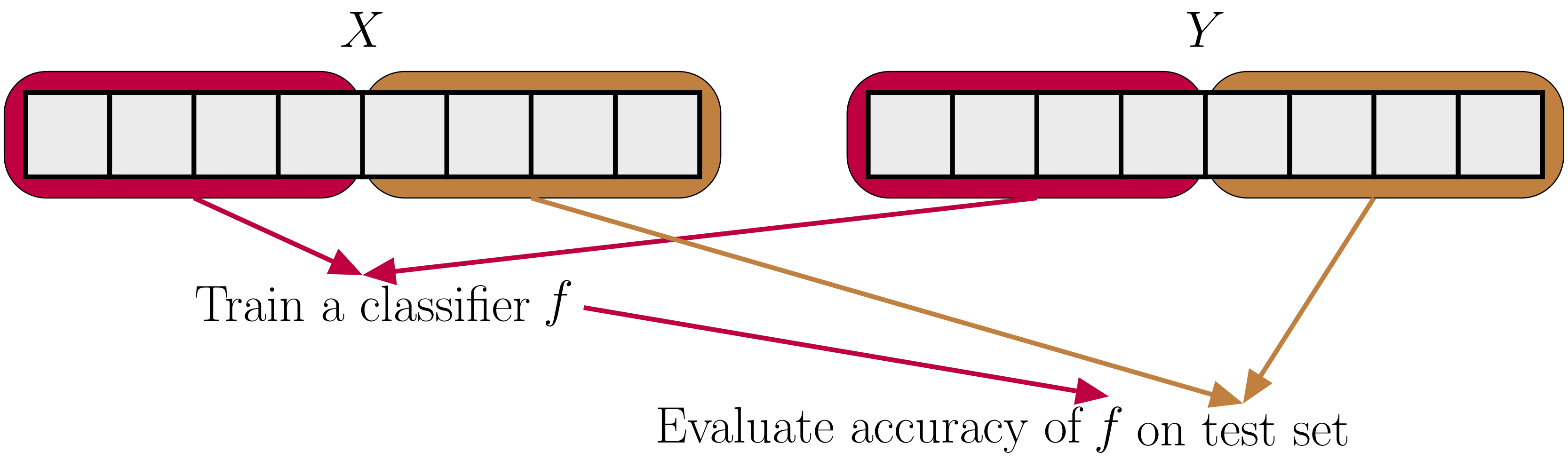
- If k is *characteristic*, $\operatorname{MMD}(\mathbb{P}, \mathbb{Q}) = 0$ iff $\mathbb{P} = \mathbb{Q}$
- Efficient permutation testing for $\widehat{\operatorname{MMD}}(X, Y)$
  - H_0: $n\widehat{\operatorname{MMD}}^2$ converges in distribution
  - H_1: $\sqrt{n}(\widehat{\operatorname{MMD}}^2 - \operatorname{MMD}^2)$ asymptotically normal
- Any characteristic kernel gives consistent test…eventually
- Need enormous n if kernel is bad for problem

Classifier two-sample tests



- $\hat{T}(X, Y)$ is the accuracy of f on the test set
- Under H_0, classification impossible: $\hat{T} \sim \text{Binomial}(n, \frac{1}{2})$

Classifier two-sample tests



- $\hat{T}(X, Y)$ is the accuracy of f on the test set
- Under H_0, classification impossible: $\hat{T} \sim \mathrm{Binomial}(n, \frac{1}{2})$
- With $k(x, y) = \frac{1}{4} f(x) f(y)$ where $f(x) \in \{-1, 1\}$, get $\widehat{\mathrm{MMD}}(X, Y) = \left| \hat{T}(X, Y) - \frac{1}{2} \right|$

Deep learning and deep kernels

- $k(x, y) = \frac{1}{4} f(x) f(y)$ is one form of *deep kernel*

Deep learning and deep kernels

- $k(x, y) = \frac{1}{4} f(x) f(y)$ is one form of *deep kernel*
- Deep models are usually of the form $f(x) = w^{\mathsf{T}} \phi_{\psi}(x)$
  - With a *learned* $\phi_{\psi}(x) : \mathcal{X} \to \mathbb{R}^D$

Deep learning and deep kernels

- $k(x, y) = \frac{1}{4} f(x) f(y)$ is one form of *deep kernel*
- Deep models are usually of the form $f(x) = w^\mathsf{T} \phi_\psi(x)$
  - With a *learned* $\phi_\psi(x) : \mathcal{X} \to \mathbb{R}^D$
- If we fix ψ, have $f \in \mathcal{H}_\psi$ with $k_\psi(x, y) = \phi_\psi(x)^\mathsf{T} \phi_\psi(y)$

Deep learning and deep kernels

- $k(x, y) = \frac{1}{4} f(x) f(y)$ is one form of *deep kernel*
- Deep models are usually of the form $f(x) = w^\mathsf{T} \phi_\psi(x)$
  - With a *learned* $\phi_\psi(x) : \mathcal{X} \to \mathbb{R}^D$
- If we fix ψ, have $f \in \mathcal{H}_\psi$ with $k_\psi(x, y) = \phi_\psi(x)^\mathsf{T} \phi_\psi(y)$
  - Same idea as NNGP approximation

Deep learning and deep kernels

- $k(x,y) = \frac{1}{4} f(x) f(y)$ is one form of *deep kernel*
- Deep models are usually of the form $f(x) = w^\mathsf{T} \phi_\psi(x)$
  - With a *learned* $\phi_\psi(x) : \mathcal{X} \to \mathbb{R}^D$
- If we fix ψ, have $f \in \mathcal{H}_\psi$ with $k_\psi(x,y) = \phi_\psi(x)^\mathsf{T} \phi_\psi(y)$
  - Same idea as NNGP approximation
- Generalize to a **deep kernel**:

$$k_\psi(x,y) = \kappa\left(\phi_\psi(x), \phi_\psi(y)\right)$$

Normal deep learning $\subset$ deep kernels

- Take $k_\psi(x, y) = \frac{1}{4} f_\psi(x) f_\psi(y)$
- Final function in $\mathcal{H}_\psi$ will be $a f_\psi(x)$

Normal deep learning $\subset$ deep kernels

- Take $k_\psi(x, y) = \frac{1}{4} f_\psi(x) f_\psi(y) + 1$
- Final function in $\mathcal{H}_\psi$ will be $a f_\psi(x) + b$

Normal deep learning $\subset$ deep kernels

- Take $k_\psi(x, y) = \frac{1}{4} f_\psi(x) f_\psi(y) + 1$
- Final function in $\mathcal{H}_\psi$ will be $a f_\psi(x) + b$
- With logistic loss: this is Platt scaling

Normal deep learning $\subset$ deep kernels

- Take $k_\psi(x, y) = \frac{1}{4} f_\psi(x) f_\psi(y) + 1$
- Final function in $\mathcal{H}_\psi$ will be $a f_\psi(x) + b$
- With logistic loss: this is Platt scaling

On Calibration of Modern Neural Networks

Chuan Guo [*1] **Geoff Pleiss** [*1] **Yu Sun** [*1] **Kilian Q. Weinberger** [1]

So what?

- This definitely does *not* say that deep learning is (even approximately) a kernel method

So what?

- This definitely does *not* say that deep learning is (even approximately) a kernel method
- ...despite what some people might want you to think

Computer Science > Machine Learning

[Submitted on 30 Nov 2020]

Every Model Learned by Gradient Descent Is Approximately a Kernel Machine

Pedro Domingos

So what?

- This definitely does *not* say that deep learning is (even approximately) a kernel method
- …despite what some people might want you to think

Computer Science > Machine Learning

[Submitted on 30 Nov 2020]

Every Model Learned by Gradient Descent Is Approximately a Kernel Machine

Pedro Domingos

- We know theoretically deep learning can learn some things faster than any kernel method [see Malach+ ICML-21 + refs]

So what?

- This definitely does *not* say that deep learning is (even approximately) a kernel method
- ...despite what some people might want you to think

> **Computer Science > Machine Learning**
>
> *[Submitted on 30 Nov 2020]*
>
> **Every Model Learned by Gradient Descent Is Approximately a Kernel Machine**
>
> Pedro Domingos

- We know theoretically deep learning can learn some things faster than any kernel method [see Malach+ ICML-21 + refs]
- But deep kernel learning ≠ traditional kernel models
  - exactly like how usual deep learning ≠ linear models

Optimizing power of MMD tests

- Asymptotics of $\widehat{\mathrm{MMD}}^2$ give us immediately that

$$\Pr_{H_1}\left(n\widehat{\mathrm{MMD}}^2 > c_\alpha\right) \approx \Phi\left(\frac{\sqrt{n}\,\mathrm{MMD}^2}{\sigma_{H_1}} - \frac{c_\alpha}{\sqrt{n}\sigma_{H_1}}\right)$$

MMD, σ_{H_1}, c_α are constants: first term usually dominates

Optimizing power of MMD tests

- Asymptotics of $\widehat{\mathrm{MMD}}^2$ give us immediately that

$$\Pr_{H_1}\left(n\widehat{\mathrm{MMD}}^2 > c_\alpha\right) \approx \Phi\left(\frac{\sqrt{n}\,\mathrm{MMD}^2}{\sigma_{H_1}} - \frac{c_\alpha}{\sqrt{n}\sigma_{H_1}}\right)$$

  MMD, σ_{H_1}, c_α are constants: first term usually dominates

- Pick k to maximize an estimate of $\mathrm{MMD}^2 / \sigma_{H_1}$

Optimizing power of MMD tests

- Asymptotics of $\widehat{\mathrm{MMD}}^2$ give us immediately that

$$\Pr_{H_1}\left(n\widehat{\mathrm{MMD}}^2 > c_\alpha\right) \approx \Phi\left(\frac{\sqrt{n}\,\mathrm{MMD}^2}{\sigma_{H_1}} - \frac{c_\alpha}{\sqrt{n}\sigma_{H_1}}\right)$$

  MMD, σ_{H_1}, c_α are constants: first term usually dominates

- Pick k to maximize an estimate of $\mathrm{MMD}^2 / \sigma_{H_1}$
- Use $\widehat{\mathrm{MMD}}$ from before, get $\hat{\sigma}_{H_1}$ from U-statistic theory

Optimizing power of MMD tests

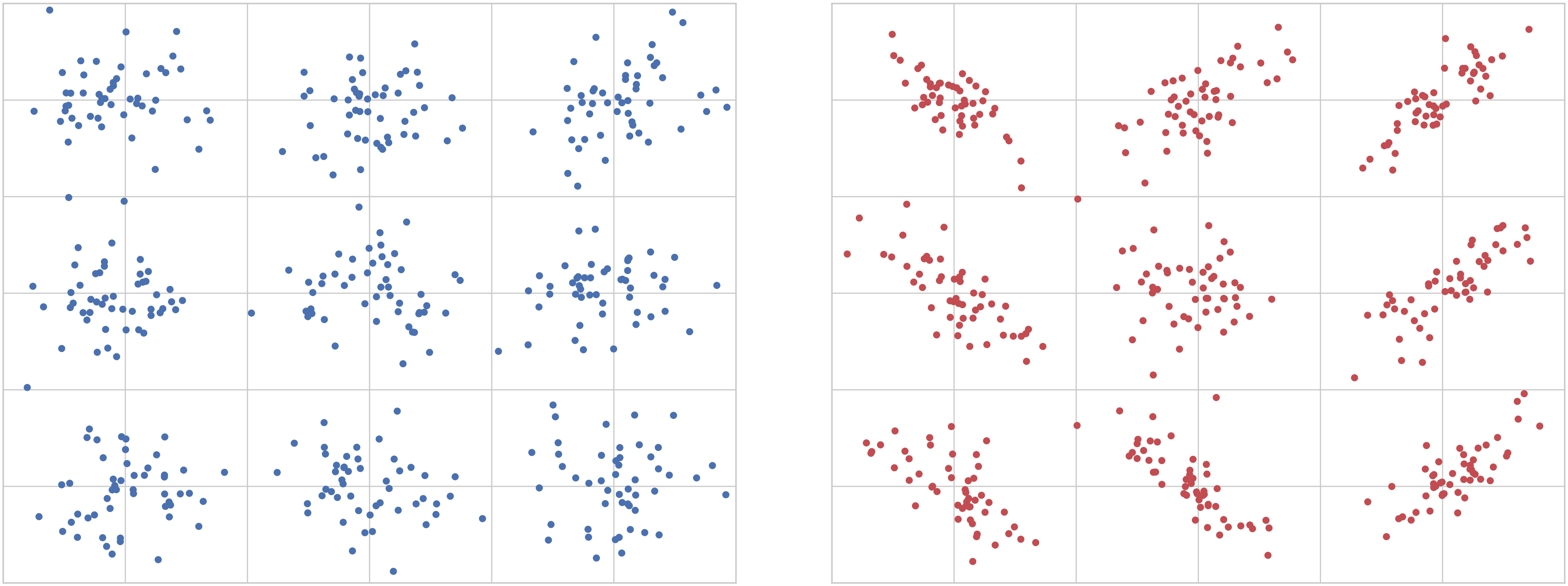
- Asymptotics of $\widehat{\mathrm{MMD}}^2$ give us immediately that

$$\Pr_{H_1}\left(n\widehat{\mathrm{MMD}}^2 > c_\alpha\right) \approx \Phi\left(\frac{\sqrt{n}\,\mathrm{MMD}^2}{\sigma_{H_1}} - \frac{c_\alpha}{\sqrt{n}\sigma_{H_1}}\right)$$

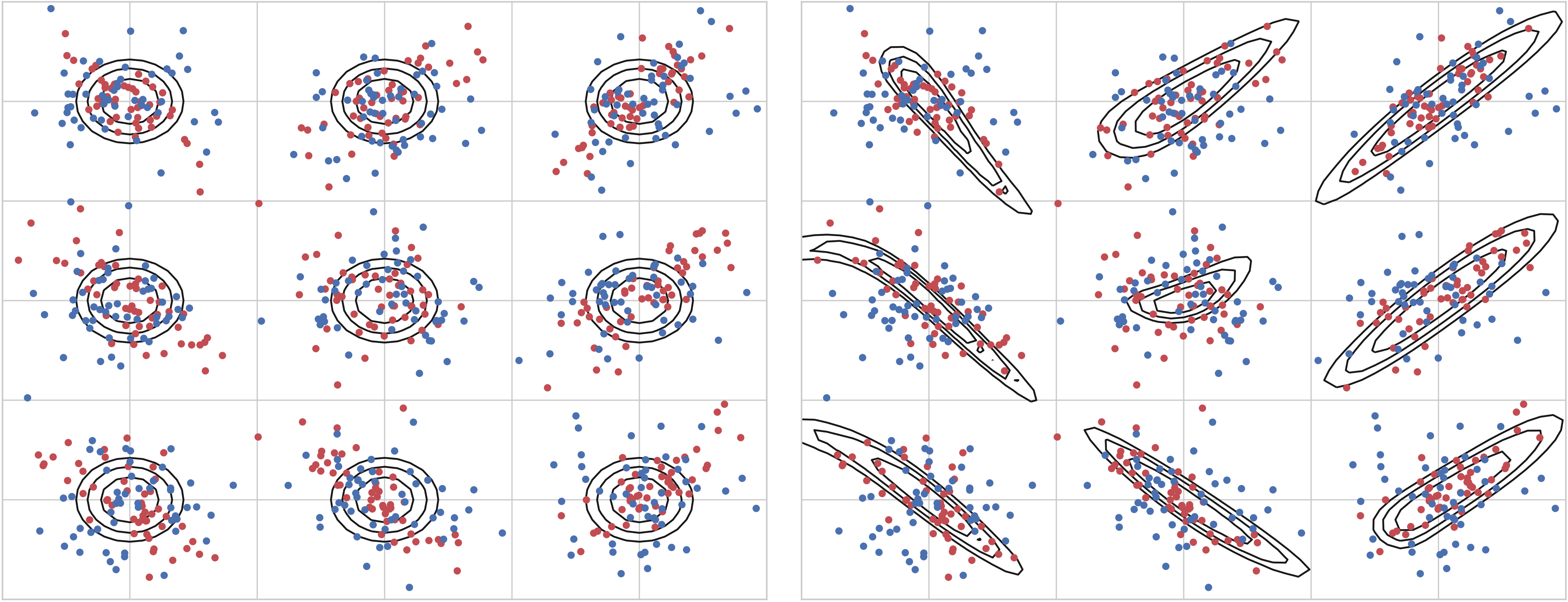
MMD, σ_{H_1}, c_α are constants: first term usually dominates

- Pick k to maximize an estimate of $\mathrm{MMD}^2 / \sigma_{H_1}$
- Use $\widehat{\mathrm{MMD}}$ from before, get $\hat{\sigma}_{H_1}$ from U-statistic theory
- Can show uniform $\mathcal{O}_P(n^{-\frac{1}{3}})$ convergence of estimator

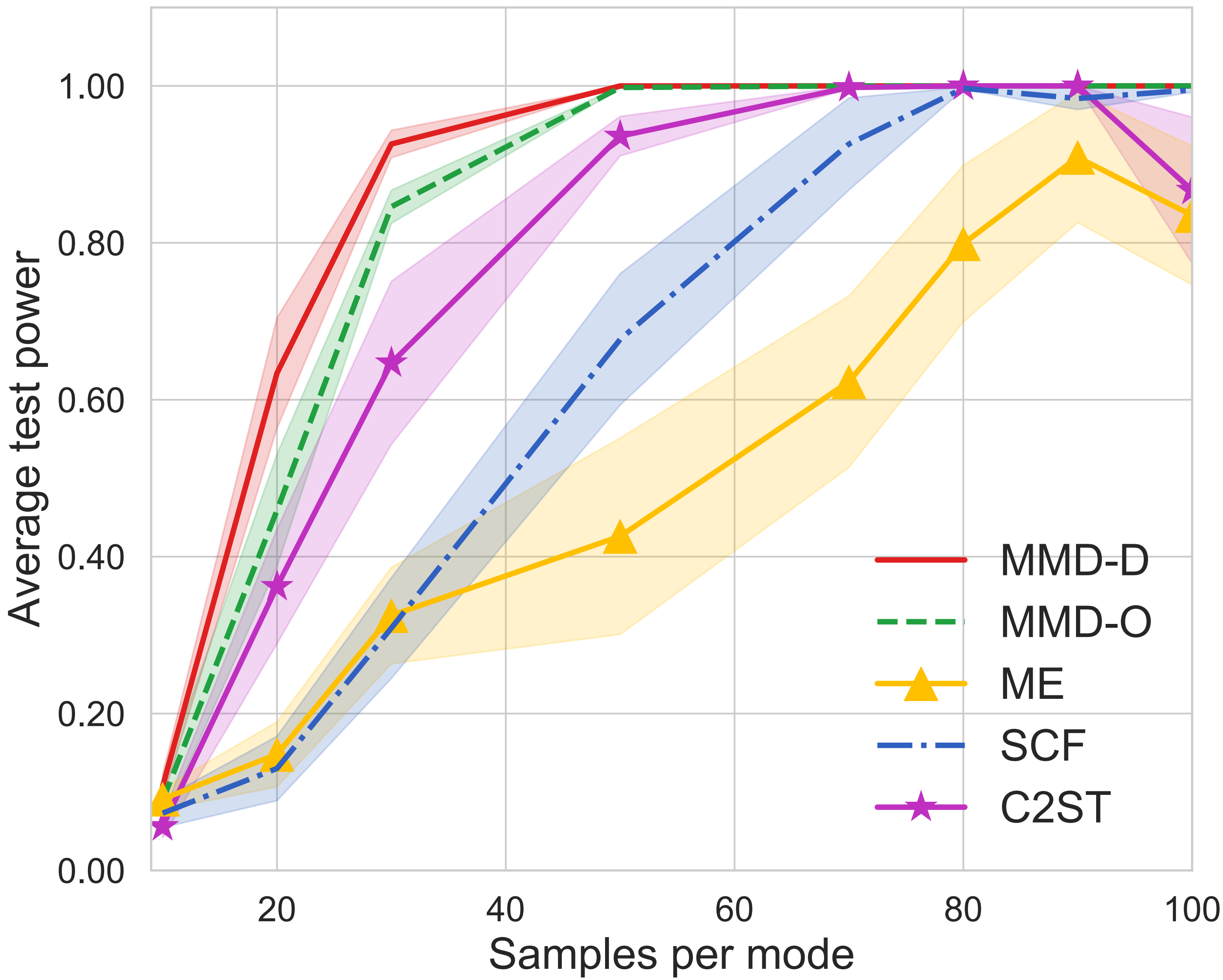
Blobs dataset



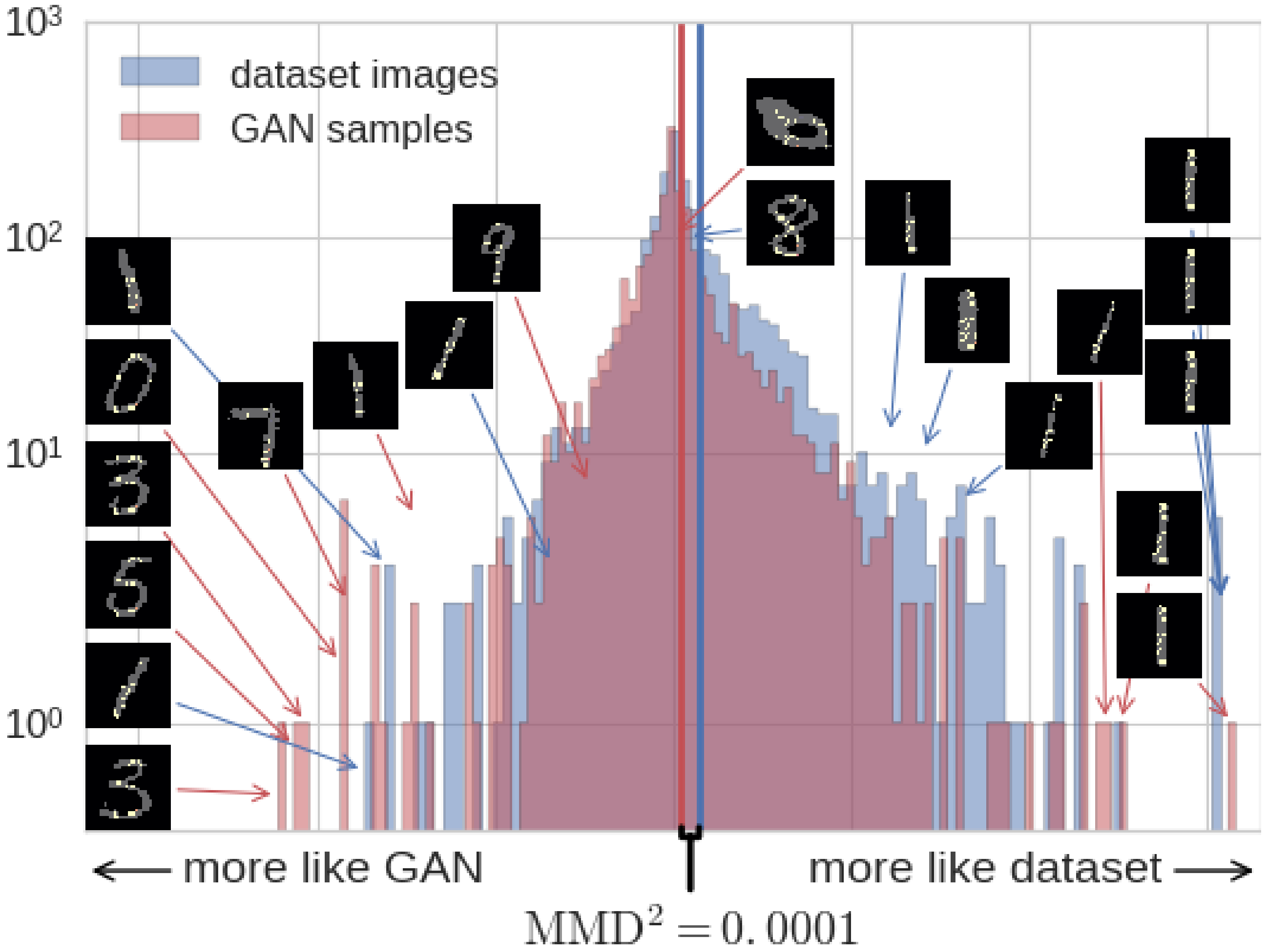
Blobs kernels



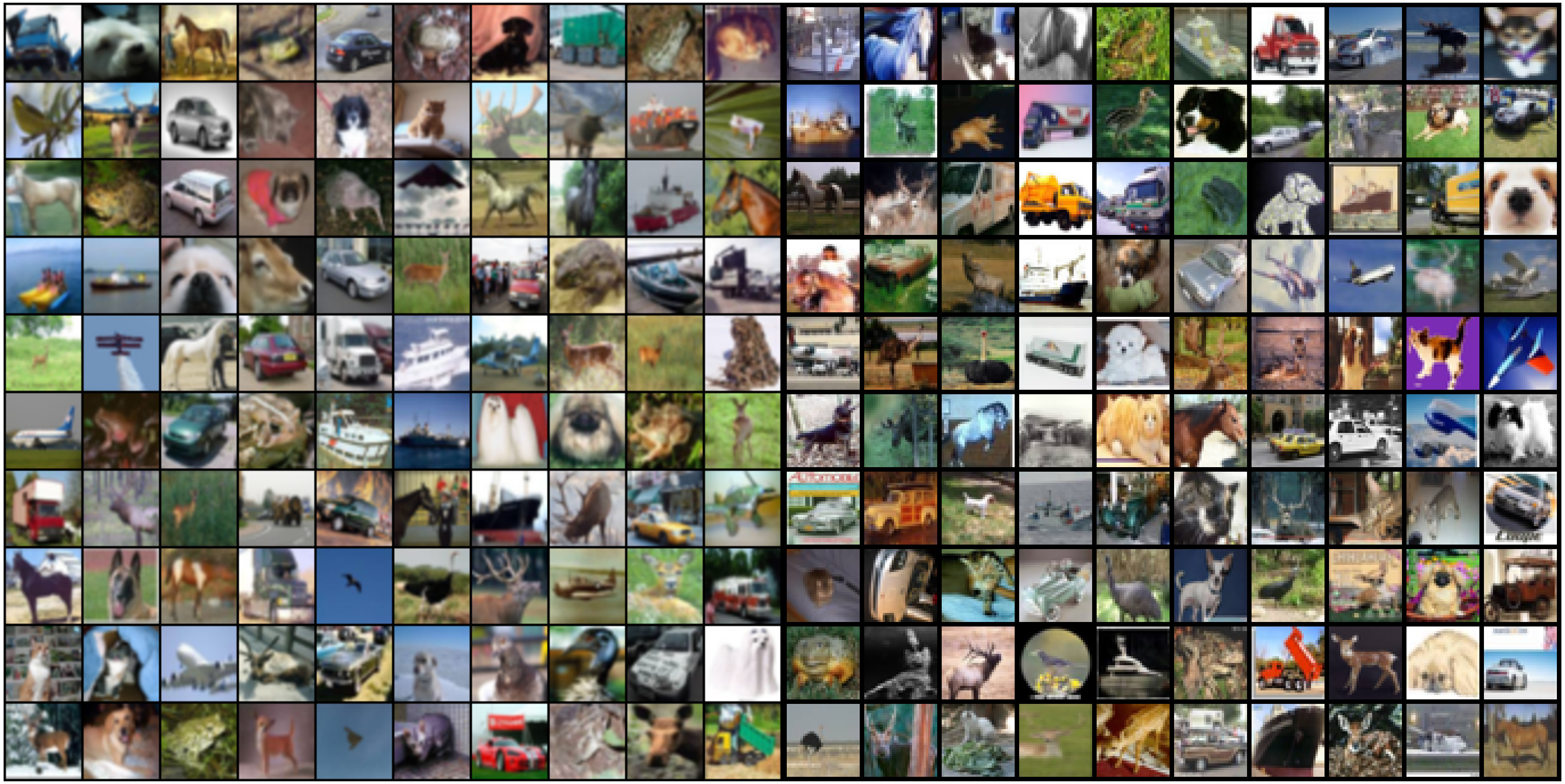
Blobs results



Investigating a GAN on MNIST



CIFAR-10 vs CIFAR-10.1



Train on 1 000, test on 1 031, repeat 10 times. Rejection rates:

ME	SCF	C2ST	MMD-O	MMD-D
0.588	0.171	0.452	0.316	**0.744**

Ablation vs classifier-based tests

	Cross-entropy			Max power		
Dataset	**Sign**	**Lin**	**Ours**	**Sign**	**Lin**	**Ours**
Blobs	0.84	0.94	0.90	–	0.95	0.99
High-d Gauss. mix.	0.47	0.59	0.29	–	0.64	0.66
Higgs	0.26	0.40	0.35	–	0.30	0.40
MNIST vs GAN	0.65	0.71	0.80	–	0.94	1.00

But...

- What if you don't have much data for your testing problem?

But...

- What if you don't have much data for your testing problem?
- Need enough data to pick a good kernel

But...

- What if you don't have much data for your testing problem?
- Need enough data to pick a good kernel
- Also need enough test data to actually detect the difference

But...

- What if you don't have much data for your testing problem?
- Need enough data to pick a good kernel
- Also need enough test data to actually detect the difference
- Best split depends on best kernel's quality / how hard to find

But...

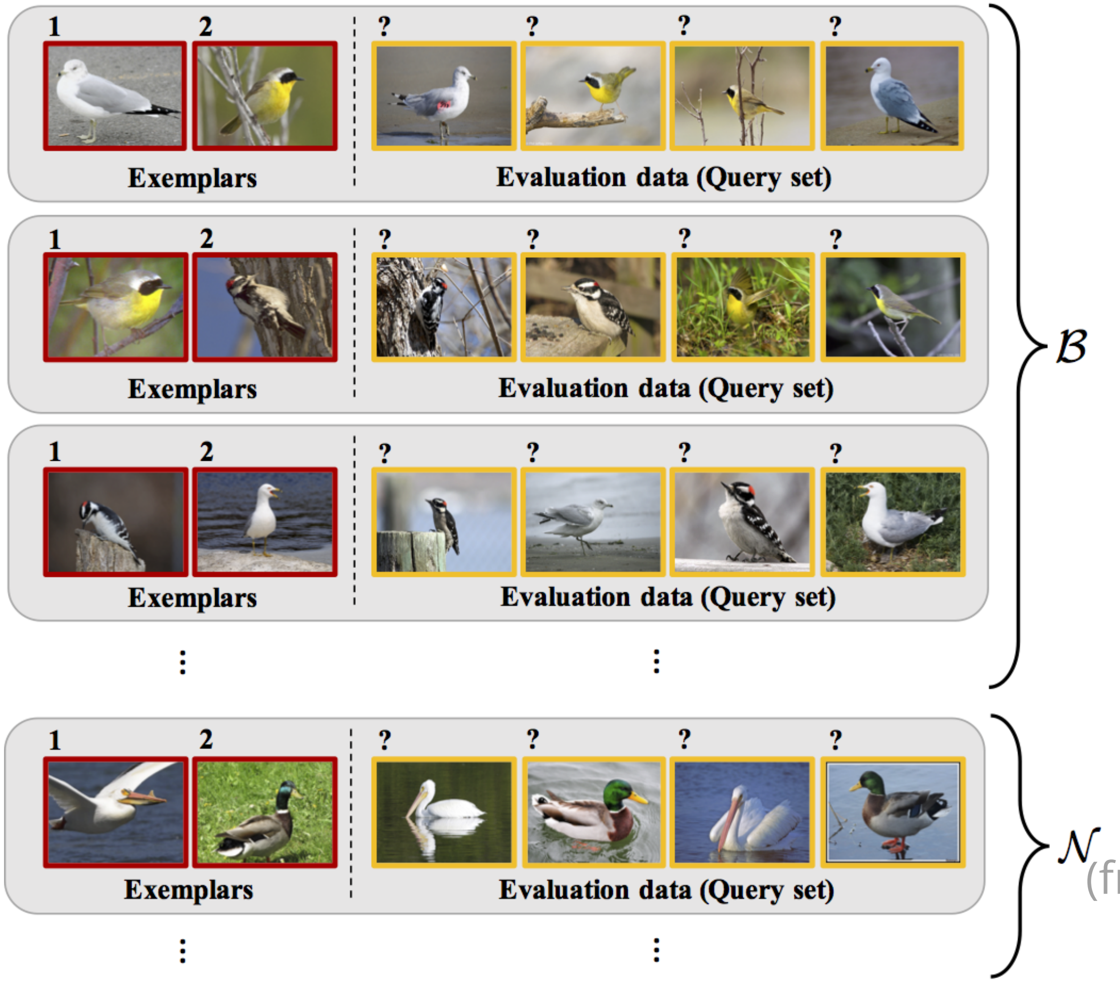
- What if you don't have much data for your testing problem?
- Need enough data to pick a good kernel
- Also need enough test data to actually detect the difference
- Best split depends on best kernel's quality / how hard to find
  - Don't know that ahead of time; can't try more than one

Meta-testing

- One idea: what if we have *related* problems?

Meta-testing

- One idea: what if we have *related* problems?
- Similar setup to meta-learning:



(from Wei+ 2018)

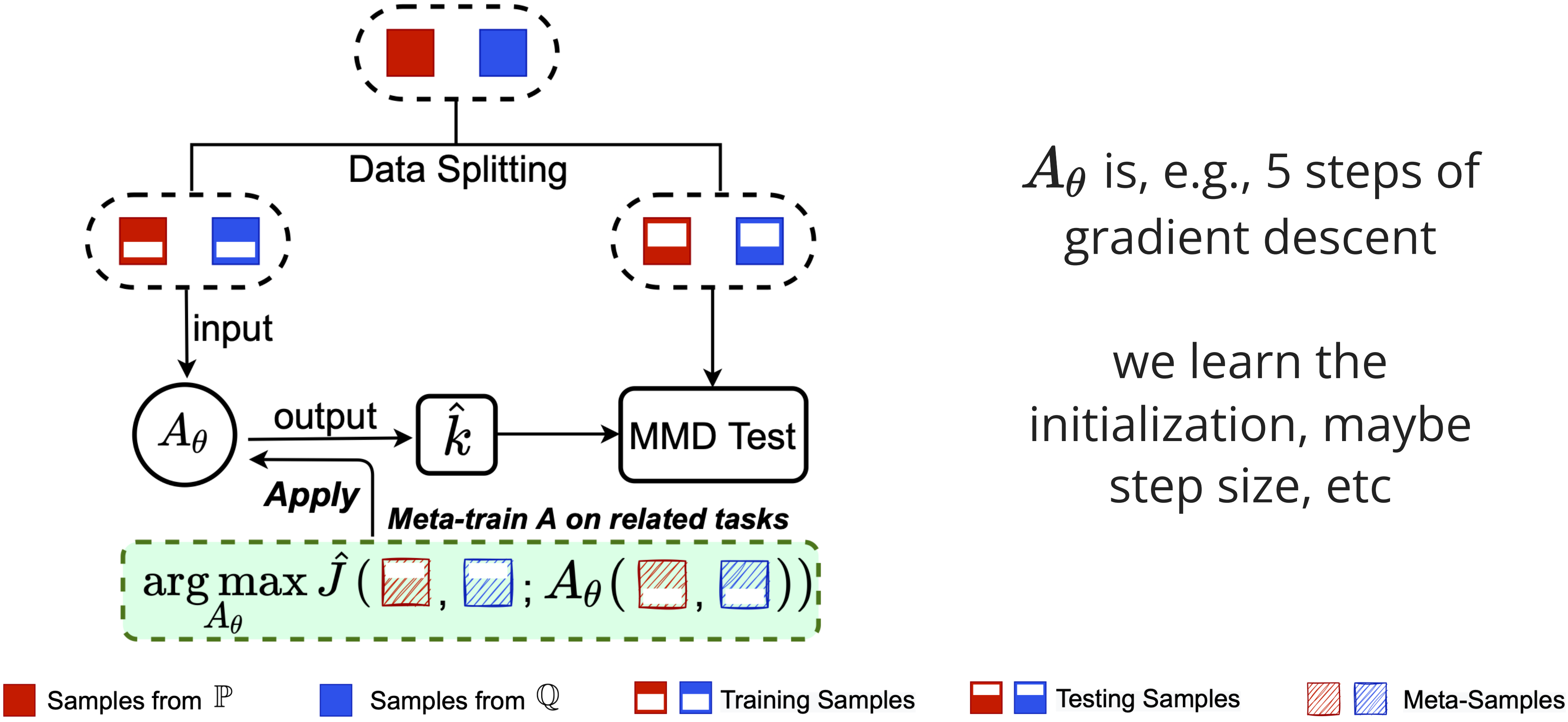
Meta-testing for CIFAR-10 vs CIFAR-10.1

- CIFAR-10 has 60,000 images, but CIFAR-10.1 only has 2,031
- Where do we get related data from?

Meta-testing for CIFAR-10 vs CIFAR-10.1

- CIFAR-10 has 60,000 images, but CIFAR-10.1 only has 2,031
- Where do we get related data from?
- One option: set up tasks to distinguish classes of CIFAR-10 (airplane vs automobile, airplane vs bird, ...)

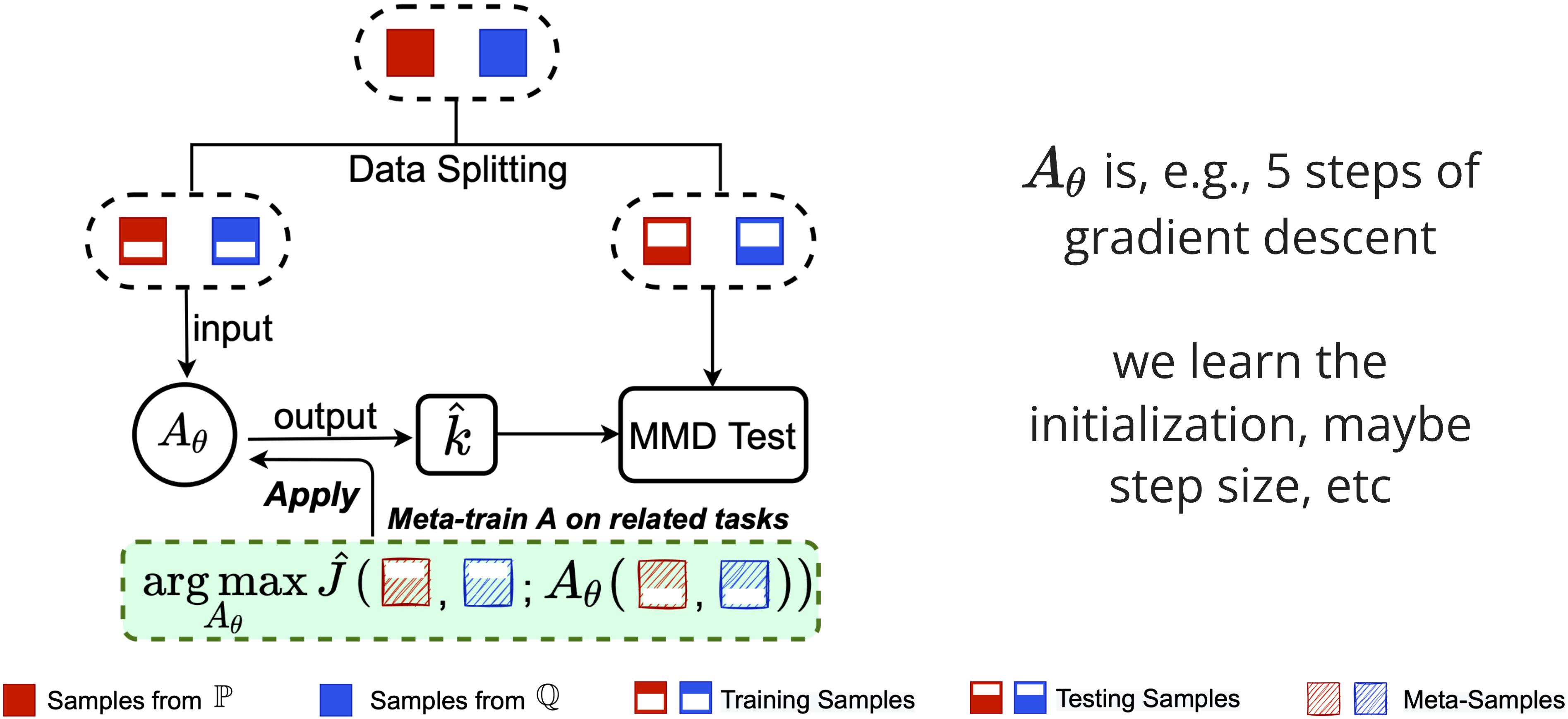
One approach (MAML-like)



A_θ is, e.g., 5 steps of gradient descent

we learn the initialization, maybe step size, etc

One approach (MAML-like)

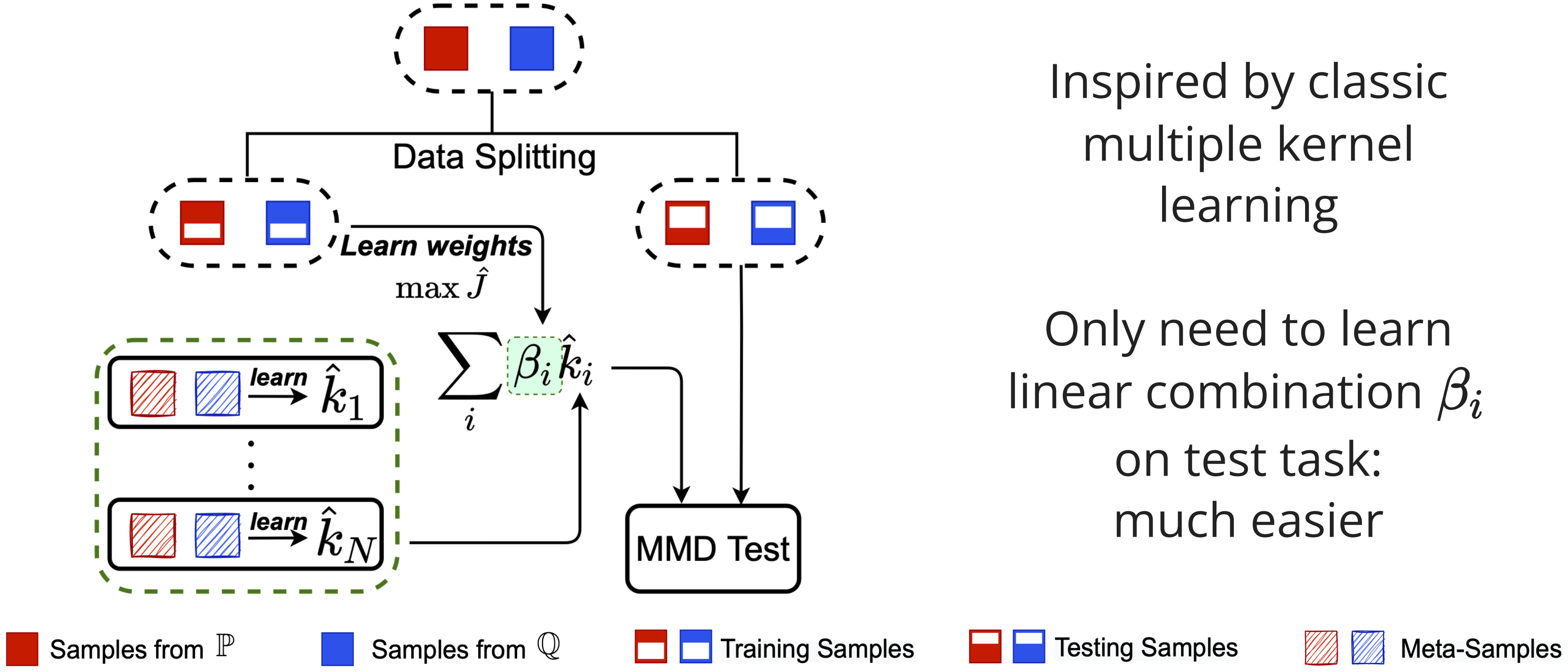


A_θ is, e.g., 5 steps of gradient descent

we learn the initialization, maybe step size, etc

This works, but not as well as we'd hoped...
Initialization might work okay on everything, not really adapt

Another approach: Meta-MKL



Inspired by classic multiple kernel learning

Only need to learn linear combination β_i on test task: much easier

Theoretical analysis for Meta-MKL

- Same big-O dependence on test task size 😐
- But multiplier is *much* better:
  based on number of meta-training tasks, not on network size

Theoretical analysis for Meta-MKL

- Same big-O dependence on test task size 😐
- But multiplier is *much* better:
  based on number of meta-training tasks, not on network size
- Coarse analysis: assumes one meta-tasks is “related” enough
  - We compete with picking the single best related kernel
  - Haven't analyzed meaningfully combining related kernels (yet!)

Results on CIFAR-10.1

Methods	$m_{tr} = 100$			$m_{tr} = 200$		
	$m_{te} = 200$	$m_{te} = 500$	$m_{te} = 900$	$m_{te} = 200$	$m_{te} = 500$	$m_{te} = 900$
ME	$0.084_{\pm 0.009}$	$0.096_{\pm 0.016}$	$0.160_{\pm 0.035}$	$0.104_{\pm 0.013}$	$0.202_{\pm 0.020}$	$0.326_{\pm 0.039}$
SCF	$0.047_{\pm 0.013}$	$0.037_{\pm 0.011}$	$0.047_{\pm 0.015}$	$0.026_{\pm 0.009}$	$0.018_{\pm 0.006}$	$0.026_{\pm 0.012}$
C2ST-S	$0.059_{\pm 0.009}$	$0.062_{\pm 0.007}$	$0.059_{\pm 0.007}$	$0.052_{\pm 0.011}$	$0.054_{\pm 0.011}$	$0.057_{\pm 0.008}$
C2ST-L	$0.064_{\pm 0.009}$	$0.064_{\pm 0.006}$	$0.063_{\pm 0.007}$	$0.075_{\pm 0.014}$	$0.066_{\pm 0.011}$	$0.067_{\pm 0.008}$
MMD-O	$0.091_{\pm 0.011}$	$0.141_{\pm 0.009}$	$0.279_{\pm 0.018}$	$0.084_{\pm 0.007}$	$0.160_{\pm 0.011}$	$0.319_{\pm 0.020}$
MMD-D	$0.104_{\pm 0.007}$	$0.222_{\pm 0.020}$	$0.418_{\pm 0.046}$	$0.117_{\pm 0.013}$	$0.226_{\pm 0.021}$	$0.444_{\pm 0.037}$
AGT-KL	$0.170_{\pm 0.032}$	$0.457_{\pm 0.052}$	$0.765_{\pm 0.045}$	$0.152_{\pm 0.023}$	$0.463_{\pm 0.060}$	$0.778_{\pm 0.050}$
Meta-KL	$0.245_{\pm 0.010}$	$0.671_{\pm 0.026}$	$0.959_{\pm 0.013}$	$0.226_{\pm 0.015}$	$0.668_{\pm 0.032}$	$0.972_{\pm 0.006}$
Meta-MKL	$\mathbf{0.277}_{\pm 0.016}$	$\mathbf{0.728}_{\pm 0.020}$	$\mathbf{0.973}_{\pm 0.008}$	$\mathbf{0.255}_{\pm 0.020}$	$\mathbf{0.724}_{\pm 0.026}$	$\mathbf{0.993}_{\pm 0.003}$

But...

- Sometimes we know ahead of time that there are differences that we don't care about

But...

- Sometimes we know ahead of time that there are differences that we don't care about
  - In the MNIST GAN criticism, initial attempt just picked out that the GAN outputs numbers that aren't one of the 256 values MNIST has

But...

- Sometimes we know ahead of time that there are differences that we don't care about
  - In the MNIST GAN criticism, initial attempt just picked out that the GAN outputs numbers that aren't one of the 256 values MNIST has

- Can we find a kernel that *can* distinguish $\mathbb{P}^t$ from $\mathbb{Q}^t$, but *can't* distinguish $\mathbb{P}^s$ from $\mathbb{Q}^s$?

But...

- Sometimes we know ahead of time that there are differences that we don't care about
  - In the MNIST GAN criticism, initial attempt just picked out that the GAN outputs numbers that aren't one of the 256 values MNIST has
- Can we find a kernel that *can* distinguish $\mathbb{P}^t$ from $\mathbb{Q}^t$, but *can't* distinguish $\mathbb{P}^s$ from $\mathbb{Q}^s$?
- Also useful for **fair representation learning**

But...

- Sometimes we know ahead of time that there are differences that we don't care about
  - In the MNIST GAN criticism, initial attempt just picked out that the GAN outputs numbers that aren't one of the 256 values MNIST has
- Can we find a kernel that *can* distinguish $\mathbb{P}^t$ from $\mathbb{Q}^t$, but *can't* distinguish $\mathbb{P}^s$ from $\mathbb{Q}^s$?
- Also useful for **fair representation learning**
  - e.g. can distinguish “creditworthy” vs not, can't distinguish by race

High on one power, low on another

Choose k with $\min_k \rho_k^s - \rho_k^t$

High on one power, low on another

Choose k with $\min_k \rho_k^s - \rho_k^t$

- First idea: $\rho = \frac{(\mathrm{MMD})^2}{\sigma_{H_1}}$

High on one power, low on another

Choose k with $\min_k \rho_k^s - \rho_k^t$

- First idea: $\rho = \dfrac{(\mathrm{MMD})^2}{\sigma_{H_1}}$
  - No good: doesn't balance power appropriately

High on one power, low on another

Choose k with $\min_k \rho_k^s - \rho_k^t$

- First idea: $\rho = \dfrac{(\mathrm{MMD})^2}{\sigma_{H_1}}$
  - No good: doesn't balance power appropriately
- Second idea: $\rho = \Phi\left(\dfrac{\sqrt{n}(\mathrm{MMD})^2 - c_\alpha}{\sigma_{H_1}}\right)$

High on one power, low on another

Choose k with $\min_k \rho_k^s - \rho_k^t$

- First idea: $\rho = \dfrac{(\mathrm{MMD})^2}{\sigma_{H_1}}$
  - No good: doesn't balance power appropriately
- Second idea: $\rho = \Phi\left(\dfrac{\sqrt{n}(\mathrm{MMD})^2 - c_\alpha}{\sigma_{H_1}}\right)$
  - Can estimate c_α inside the optimization

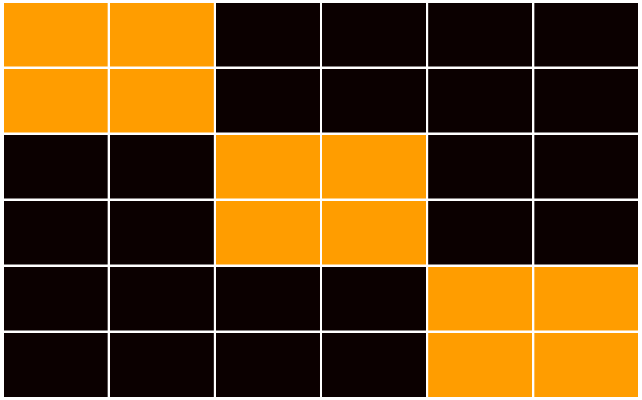
High on one power, low on another

Choose k with $\min_k \rho_k^s - \rho_k^t$

- First idea: $\rho = \dfrac{(\mathrm{MMD})^2}{\sigma_{H_1}}$
  - No good: doesn't balance power appropriately
- Second idea: $\rho = \Phi\left(\dfrac{\sqrt{n}(\mathrm{MMD})^2 - c_\alpha}{\sigma_{H_1}}\right)$
  - Can estimate c_α inside the optimization
  - Better, but tends to “stall out” in minimizing ρ_k^s

Block estimator [Zaremba+ NeurIPS-13]

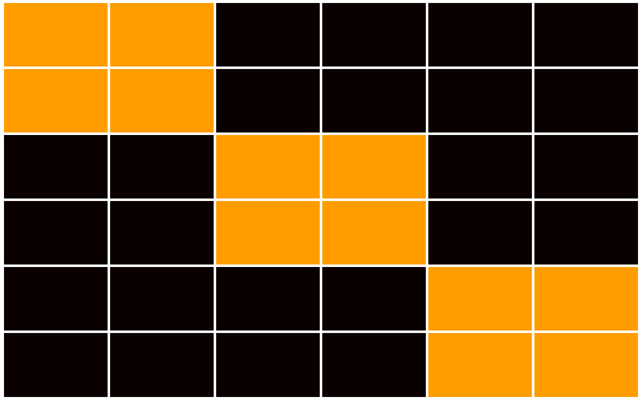
- Use previous $\widehat{\text{MMD}}$ on b blocks, each of size B



- Final estimator: average of each block's estimate

Block estimator [Zaremba+ NeurIPS-13]

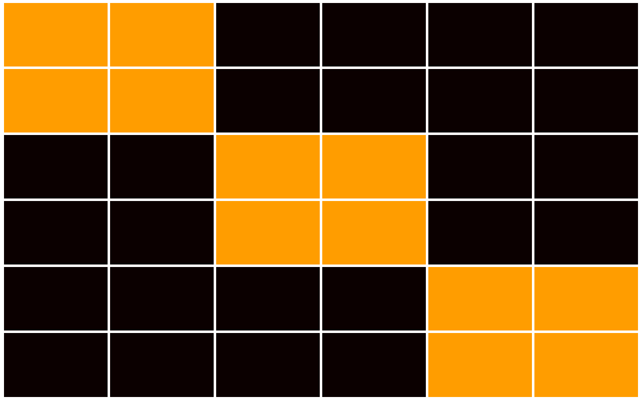
- Use previous $\widehat{\mathrm{MMD}}$ on b blocks, each of size B



- Final estimator: average of each block's estimate
  - Each block has previous asymptotics

Block estimator [Zaremba+ NeurIPS-13]

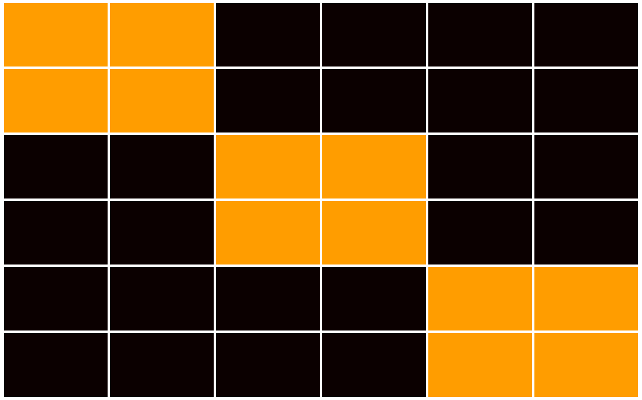
- Use previous $\widehat{\mathrm{MMD}}$ on b blocks, each of size B



- Final estimator: average of each block's estimate
  - Each block has previous asymptotics
  - Central limit theorem across blocks

Block estimator [Zaremba+ NeurIPS-13]

- Use previous $\widehat{\mathrm{MMD}}$ on b blocks, each of size B



- Final estimator: average of each block's estimate
  - Each block has previous asymptotics
  - Central limit theorem across blocks
- Power is $\rho = \Phi\left(\sqrt{bB}\frac{\mathrm{MMD}^2}{\sigma_{H_1}^2} - \Phi^{-1}(1-\alpha)\right)$

MMD-B-Fair

- Choose k as $\min_k \rho_k^s - \rho_k^t$

MMD-B-Fair

- Choose k as $\min_k \rho_k^s - \rho_k^t$
  - ρ is the power of a test with b blocks of size B

MMD-B-Fair

- Choose k as $\min_k \rho_k^s - \rho_k^t$
  - ρ is the power of a test with b blocks of size B
  - We *don't* actually use a block estimator computationally

MMD-B-Fair

- Choose k as $\min_k \rho_k^s - \rho_k^t$
  - ρ is the power of a test with b blocks of size B
  - We *don't* actually use a block estimator computationally
  - b, B have *nothing to do* with minibatch size

MMD-B-Fair

- Choose k as $\min_k \rho_k^s - \rho_k^t$
  - ρ is the power of a test with b blocks of size B
  - We *don't* actually use a block estimator computationally
  - b, B have *nothing to do* with minibatch size
- Representation learning: $\min_\phi \max_\kappa \rho_{\kappa\circ\phi}^s - \rho_{\kappa\circ\phi}^t$

MMD-B-Fair

- Choose k as $\min_k \rho_k^s - \rho_k^t$
  - ρ is the power of a test with b blocks of size B
  - We *don't* actually use a block estimator computationally
  - b, B have *nothing to do* with minibatch size
- Representation learning: $\min_\phi \max_\kappa \rho_{\kappa\circ\phi}^s - \rho_{\kappa\circ\phi}^t$
  - Deep kernel is $[\kappa \circ \phi](x, y) = \kappa(\phi(x), \phi(y))$

MMD-B-Fair

- Choose k as $\min_k \rho_k^s - \rho_k^t$
  - ρ is the power of a test with b blocks of size B
  - We *don't* actually use a block estimator computationally
  - b, B have *nothing to do* with minibatch size
- Representation learning: $\min_\phi \max_\kappa \rho_{\kappa\circ\phi}^s - \rho_{\kappa\circ\phi}^t$
  - Deep kernel is $[\kappa \circ \phi](x, y) = \kappa(\phi(x), \phi(y))$
  - κ could be deep itself, with adversarial optimization

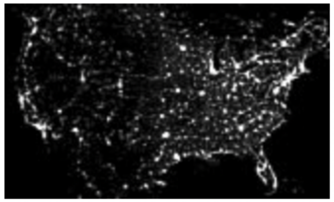
MMD-B-Fair

- Choose k as $\min_k \rho_k^s - \rho_k^t$
  - ρ is the power of a test with b blocks of size B
  - We *don't* actually use a block estimator computationally
  - b, B have *nothing to do* with minibatch size
- Representation learning: $\min_\phi \max_\kappa \rho_{\kappa\circ\phi}^s - \rho_{\kappa\circ\phi}^t$
  - Deep kernel is $[\kappa \circ \phi](x, y) = \kappa(\phi(x), \phi(y))$
  - κ could be deep itself, with adversarial optimization
  - For now, just Gaussians with different lengthscales

Adult

Adult Data Set

Download: Data Folder, Data Set Description



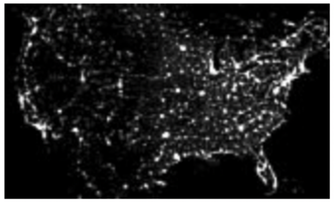
Abstract: Predict whether income exceeds $50K/yr based on census data. Also known as "Census Income" dataset.

Data Set Characteristics:	Multivariate	**Number of Instances:**	48842	**Area:**	Social
Attribute Characteristics:	Categorical, Integer	**Number of Attributes:**	14	**Date Donated**	1996-05-01
Associated Tasks:	Classification	**Missing Values?**	Yes	**Number of Web Hits:**	2390574

Adult

Adult Data Set

Download: Data Folder, Data Set Description



Abstract: Predict whether income exceeds $50K/yr based on census data. Also known as "Census Income" dataset.

Data Set Characteristics:	Multivariate	**Number of Instances:**	48842	**Area:**	Social
Attribute Characteristics:	Categorical, Integer	**Number of Attributes:**	14	**Date Donated**	1996-05-01
Associated Tasks:	Classification	**Missing Values?**	Yes	**Number of Web Hits:**	2390574

Shapes3D

$\mathbb{P}^t$:

$\mathbb{Q}^t$:

$\mathbb{P}^s$:

$\mathbb{Q}^s$:

ci-ratio	Method	Pr(target)↑	Pr(sensitive)↓	Pr(sensitive) fine-tuned↓
$(0.1, 0.1)$	Laftr	0.2500	0.6100	1.000 ($\sigma = 0.111$)
	Cfair	0.2500	0.6071	**0.8929** ($\sigma = 0.087$)
	Ffvae	0.1785	0.6428	1.000 ($\sigma = 0.0695$)
	Ours	**1.000**	**0.2500**	0.9642 ($\sigma = 0.007$)
$(0.33, 0.66)$	Laftr	0.285	0.607	1.000 ($\sigma = 0.237$)
	Cfair	0.2857	0.6071	1.000 ($\sigma = 0.234$)
	Ffvae	0.9642	1.000	1.000 ($\sigma = 0.075$)
	Ours	1.000	**0.5614**	**0.6842** ($\sigma = 0.005$)

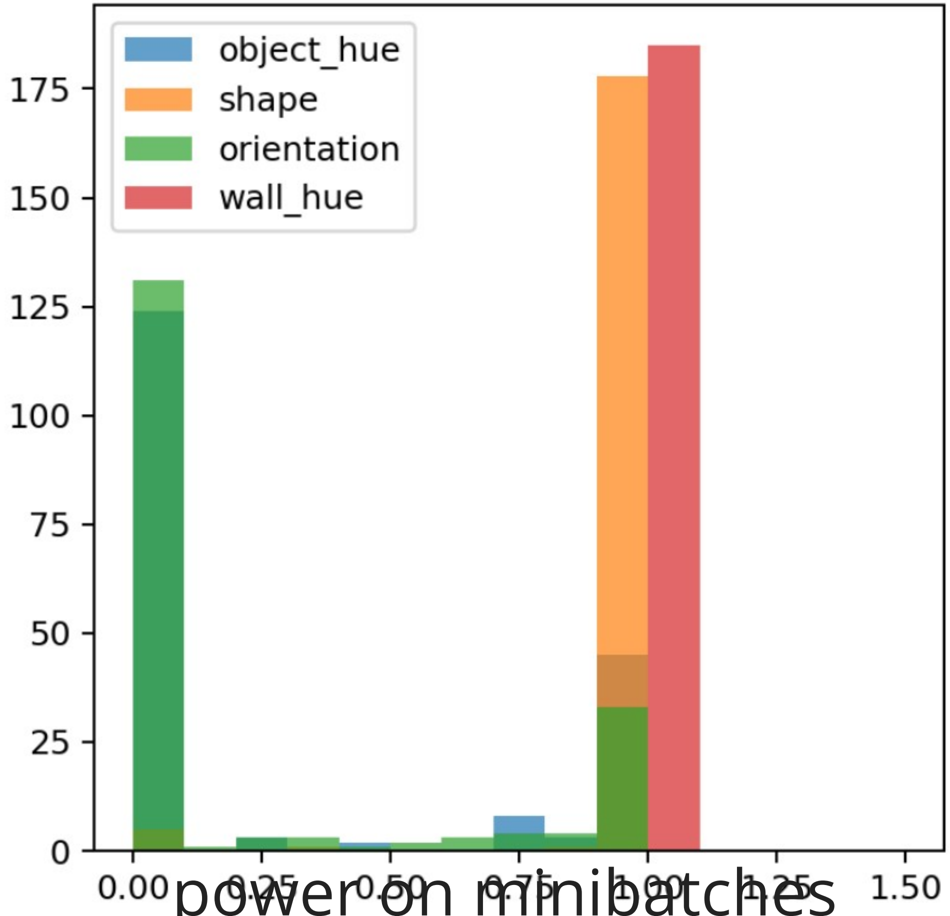
(a) **Adult dataset:** Our method outperforms all others even when additional layers are trained to maximize the sensitive power (albeit with smaller bandwidths in the under-represented scenario).

ci-ratio	Method	Pr(target)↑	Pr(sensitive)↓	Pr(sensitive) fine-tuned↓
$(0.1, 0.1)$	Laftr	1.000	1.000	1.000 ($\sigma = 0.001$)
	Cfair	1.000	1.000	1.000 ($\sigma = 0.003$)
	Ffvae	0.9574	0.9787	1.000 ($\sigma = 0.1002$)
	Ours	**1.000**	**0.0744**	**0.9625** ($\sigma = 0.0205$)
$(0.9, 0.1)$	Laftr	1.000	1.000	1.000 ($\sigma = 0.006$)
	Cfair	1.000	1.000	1.000 ($\sigma = 0.005$)
	Ffvae	**0.8723**	**0.8723**	1.000 ($\sigma = 0.092$)
	Ours	0.1383	1.000	1.000 ($\sigma = 0.006$)

(b) **3DShapes dataset:** Our method is able to outperform others in the under-represented case, but the highly correlated scenario of **ci-ratio**=(0.9,0.1) is a failure case.

Multiple targets / sensitive attributes

$$\max_k \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} \rho_k^t - \frac{1}{|\mathcal{S}|} \sum_{s \in \mathcal{S}} \rho_k^s$$



power on minibatches

Remaining challenges

- MMD-B-Fair:
  - When s and t are very correlated
  - For attributes with many values (use HSIC?)

Remaining challenges

- MMD-B-Fair:
  - When s and t are very correlated
  - For attributes with many values (use HSIC?)
- Meta-testing: more powerful approaches, better analysis

Remaining challenges

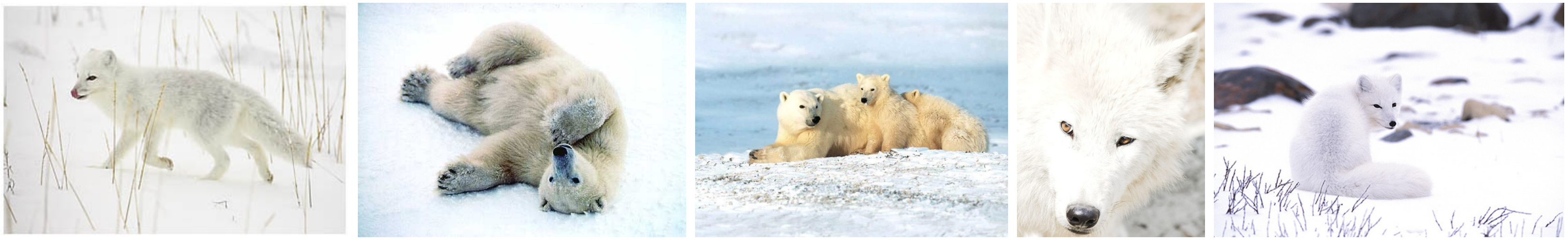
- MMD-B-Fair:
  - When s and t are very correlated
  - For attributes with many values (use HSIC?)
- Meta-testing: more powerful approaches, better analysis
- When $\mathbb{P} \neq \mathbb{Q}$, can we tell *how* they're different?

Remaining challenges

- MMD-B-Fair:
  - When s and t are very correlated
  - For attributes with many values (use HSIC?)
- Meta-testing: more powerful approaches, better analysis
- When $\mathbb{P} \neq \mathbb{Q}$, can we tell *how* they're different?
  - Methods so far: low-d, and/or points w/ large critic value

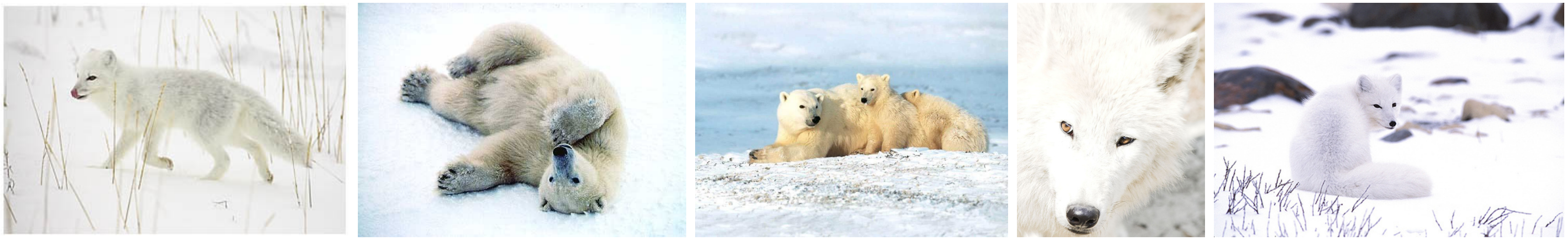
Remaining challenges

- MMD-B-Fair:
  - When s and t are very correlated
  - For attributes with many values (use HSIC?)
- Meta-testing: more powerful approaches, better analysis
- When $\mathbb{P} \neq \mathbb{Q}$, can we tell *how* they're different?
  - Methods so far: low-d, and/or points w/ large critic value



Remaining challenges

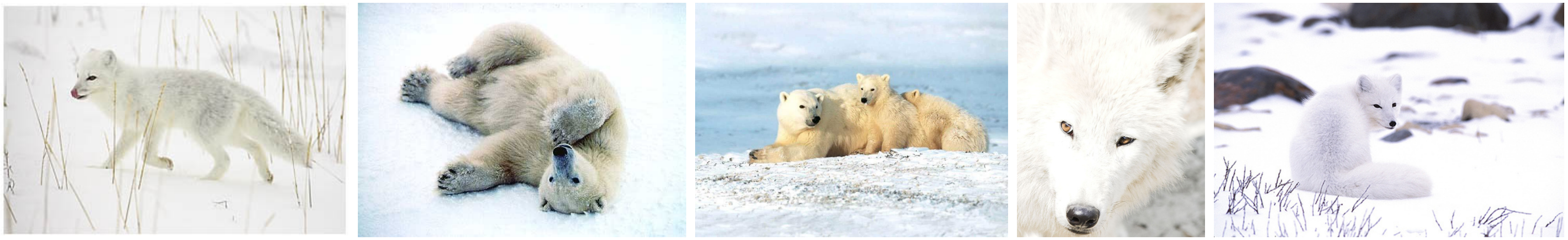
- MMD-B-Fair:
  - When s and t are very correlated
  - For attributes with many values (use HSIC?)
- Meta-testing: more powerful approaches, better analysis
- When $\mathbb{P} \neq \mathbb{Q}$, can we tell *how* they're different?
  - Methods so far: low-d, and/or points w/ large critic value



- Avoid the need for data splitting (selective inference)

Remaining challenges

- MMD-B-Fair:
  - When s and t are very correlated
  - For attributes with many values (use HSIC?)
- Meta-testing: more powerful approaches, better analysis
- When $\mathbb{P} \neq \mathbb{Q}$, can we tell *how* they're different?
  - Methods so far: low-d, and/or points w/ large critic value



- Avoid the need for data splitting (selective inference)
  - Kübler+ NeurIPS-20 gave one method, but very limited

A good takeaway

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 (2007), "Scaling Learning Algorithms towards AI"