Adaptive two-sample testing

Arthur Gretton

Gatsby Computational Neuroscience Unit,
Google Deepmind

Cambridge, 2023
Comparing two samples

- Given: Samples from unknown distributions $P$ and $Q$.
- Goal: do $P$ and $Q$ differ?
A real-life example: two-sample tests

Goal: do $P$ and $Q$ differ?

CIFAR 10 samples

Cifar 10.1 samples

Significant difference?

Feng, Xu, Lu, Zhang, G., Sutherland, Learning Deep Kernels for Non-Parametric Two-Sample Tests, ICML 2020
Sutherland, Tung, Strathmann, De, Ramdas, Smola, G., ICLR 2017.
Two-sample problem

- Samples $X_m := (x_1, \ldots, x_m)$, $x_i \overset{iid}{\sim} p$ in $\mathbb{R}^d$

- Samples $Y_n := (y_1, \ldots, y_n)$, $y_i \overset{iid}{\sim} q$ in $\mathbb{R}^d$

where $m \leq n$ and $n \leq Cm$.

Hypothesis test: function $\Delta_\alpha(X_m, Y_n)$

$\mathcal{H}_0$: $p = q$ against $\mathcal{H}_1$: $p \neq q$

$\Delta_\alpha(X_m, Y_n) = 1$ $\iff$ reject $\mathcal{H}_0$

$\Delta_\alpha(X_m, Y_n) = 0$ $\iff$ fail to reject $\mathcal{H}_0$

Type II error $\beta$

$\mathbb{P}_{p \times q}(\Delta(X_m, Y_n) = 0) \leq \beta$

Type I error: controlled by $\alpha$ by design

$\mathbb{P}_{p \times p}(\Delta(X_m, Y_n) = 1) \leq \alpha$
Outline

Two sample testing

- Test statistic: **Maximum Mean Discrepancy (MMD)**...
  - ...as a difference in feature means
  - ...as an integral probability metric

- **Statistical testing** with the MMD

- “**How to choose the best kernel**”
  - using aggregation (no sample splitting)
  - minimax guarantees with Sobolev smoothness assumption
Maximum Mean Discrepancy
Kernel methods, feature representation

Kernels: dot products of features

Feature map \( \varphi(x) \in \mathcal{F} \),

\[
\varphi(x) = [\ldots \varphi_i(x) \ldots] \in \ell_2
\]

For positive definite \( k \),

\[
k(x, x') = \langle \varphi(x), \varphi(x') \rangle_{\mathcal{F}}
\]

Infinitely many features \( \varphi(x) \), dot product in closed form!
Kernel methods, feature representation

Kernels: dot products of features

Feature map \( \varphi(x) \in \mathcal{F} \),

\[
\varphi(x) = [\ldots \varphi_i(x) \ldots] \in l_2
\]

For positive definite \( k \),

\[
k(x, x') = \langle \varphi(x), \varphi(x') \rangle_{\mathcal{F}}
\]

Infinitely many features \( \varphi(x) \), dot product in closed form!

Exponentiated quadratic kernel

\[
k(x, x') = \exp \left( -\gamma \| x - x' \|^2 \right)
\]

Features: Gaussian Processes for Machine learning, Rasmussen and Williams, Ch. 4.
Kernels and bandwidths

Kernel: \( k_\lambda(x, y) := \prod_{i=1}^{d} \frac{1}{\lambda_i} K_i \left( \frac{x_i - y_i}{\lambda_i} \right) \)

Bandwidth: \( \lambda \in (0, \infty)^d \)

Assumptions: \( K_1, \ldots, K_d \) integrable (to 1) and square integrable

Eg: Gaussian (\( K_i(u) \propto e^{-u^2} \)), Laplace (\( K_i(u) \propto e^{-|u|} \)), Matérn,…

Gaussian kernel: \( k_\lambda(x - y) := \frac{1}{(\pi)^{d/2}} \prod_{i=1}^{d} \frac{1}{\lambda_i} \exp \left( -\frac{(x_i - y_i)^2}{\lambda_i^2} \right) \)
Kernel mean embedding

Function evaluation in an RKHS:

\[ f(x) = \langle f, \varphi_x \rangle_F \]

Expectation evaluation in an RKHS:

\[ E_P(f(X)) = E_P \langle f, \varphi_X \rangle_F = \langle f, E_P \varphi_X \rangle_F =: \langle f, \mu_P \rangle_F \]

as long as feature map Bochner integrable: \( E_P \| \varphi_X \| = E_P \sqrt{k_\lambda(X, X)} < \infty \).

\[ \mu_P \text{ gives you expectations of all RKHS functions} \]

“Kernel trick” for mean embeddings:

\[ \langle \mu_P, \mu_Q \rangle_F = E_{P, Q}k_\lambda(X, Y) \]

for \( X \sim P \) and \( Y \sim Q \).
The maximum mean discrepancy

The maximum mean discrepancy is the distance between feature means:

\[
MMD^2_{\lambda}(p, q) = \|\mu_P - \mu_Q\|^2_{\mathcal{F}}
\]

\[
= E_P k_{\lambda}(X, X') + E_Q k_{\lambda}(Y, Y') - 2E_{P, Q} k_{\lambda}(X, Y)
\]

(a) = within distrib. similarity, (b) = cross-distrib. similarity.
Characteristic kernels on $\mathbb{R}^d$

**Characteristic kernel:** \( \text{MMD}^2_\lambda(p, q) = 0 \iff p = q \)  

Fukumizu et al. [NIPS07b], Sriperumbudur et al. [COLT08]

When are **translation invariant** kernels \( k_\lambda(x, y) = k_\lambda(x - y) \) characteristic on $\mathbb{R}^d$?
Characteristic kernels on $\mathbb{R}^d$

Characteristic kernel: $\text{MMD}^2_\lambda(p, q) = 0$ iff $p = q$ \cite{Fukumizu}, \cite{Sriperumbudur}

When are translation invariant kernels $k_\lambda(x, y) = k_\lambda(x - y)$ characteristic on $\mathbb{R}^d$?

Bochner’s theorem:

$$k_\lambda(x - y) = \int_{\mathbb{R}^d} e^{-i(x-y)^\top \omega} \, d\Lambda(\omega)$$

$\Lambda(\omega)$ finite non-negative Borel measure.

Characteristic function of $P$ via Fourier transform

$$\varphi_P(\omega) = \int_{\mathbb{R}^d} e^{ix^\top \omega} \, dP(x)$$
Characteristic kernels on $\mathbb{R}^d$

Fourier representation of MMD on $\mathbb{R}^d$:

$$MMD_{\lambda}(p, q) = \int |\varphi_P(\omega) - \varphi_Q(\omega)|^2 \, d\Lambda(\omega)$$

Proof:

$$MMD_{\lambda}(p, q)$$

$:= E_P k_\lambda(x - x') + E_Q k_\lambda(y - y') - 2 E_{P, Q} k_\lambda(x, y)$

$= \int \int [k_\lambda(s - t) \, d(P - Q)(s)] \, d(P - Q)(t)$

$= \int_{\mathbb{R}^d} |\phi_P(\omega) - \phi_Q(\omega)|^2 \, d\Lambda(\omega)$
Characteristic kernels on $\mathbb{R}^d$

Fourier representation of MMD on $\mathbb{R}^d$:

$$\text{MMD}^2_{\lambda}(\mathcal{P}, \mathcal{Q}) = \int |\varphi_{\mathcal{P}}(\omega) - \varphi_{\mathcal{Q}}(\omega)|^2 \, d\Lambda(\omega)$$

Proof:

$$\text{MMD}^2_{\lambda}(\mathcal{P}, \mathcal{Q})$$

$$:= \mathbb{E}_{\mathcal{P}} k_\lambda(x - x') + \mathbb{E}_{\mathcal{Q}} k_\lambda(y - y') - 2 \mathbb{E}_{\mathcal{P}, \mathcal{Q}} k_\lambda(x, y)$$

$$= \int \int \left[ k_\lambda(s - t) \, d(\mathcal{P} - \mathcal{Q})(s) \right] \, d(\mathcal{P} - \mathcal{Q})(t)$$

$$= \int_{\mathbb{R}^d} |\phi_{\mathcal{P}}(\omega) - \phi_{\mathcal{Q}}(\omega)|^2 \, d\Lambda(\omega)$$
Characteristic kernels on $\mathbb{R}^d$

Fourier representation of MMD on $\mathbb{R}^d$:

$$MMD_\lambda^2(p, q) = \int |\varphi_P(\omega) - \varphi_Q(\omega)|^2 \, d\Lambda(\omega)$$

Proof:

$$MMD_\lambda^2(p, q)$$
$$:= E_P k_\lambda(x - x') + E_Q k_\lambda(y - y') - 2E_{P,Q} k_\lambda(x, y)$$
$$= \int \int [k_\lambda(s - t) \, d(P - Q)(s)] \, d(P - Q)(t)$$
$$= \int_{\mathbb{R}^d} |\phi_P(\omega) - \phi_Q(\omega)|^2 \, d\Lambda(\omega)$$
Summary: characteristic kernels on $\mathbb{R}^d$

A translation invariant $k_\lambda$ is characteristic for prob. measures on $\mathbb{R}^d$ if and only if

$$\text{supp}(\Lambda) = \mathbb{R}^d$$

(i.e. support zero on at most a countable set) Sriperumbudur et al. [COLT08, JMLR10]

Corollary: any continuous, compactly supported $k_\lambda$ characteristic (since Fourier spectrum $\Lambda(\omega)$ cannot be zero on an interval).

1-D proof sketch from [Mallat, 99, Theorem 2.6], proof on $\mathbb{R}^d$ via distribution theory in Sriperumbudur et al. [JMLR10, Corollary 10 p. 1535]
A translation invariant $k_\lambda$ is characteristic for prob. measures on $\mathbb{R}^d$ if and only if

$$\text{supp}(\Lambda) = \mathbb{R}^d$$

(i.e. support zero on at most a countable set) Sriperumbudur et al. [COLT08, JMLR10]

Corollary: any continuous, compactly supported $k_\lambda$ characteristic (since Fourier spectrum $\Lambda(\omega)$ cannot be zero on an interval).

1-D proof sketch from [Mallat, 99, Theorem 2.6], proof on $\mathbb{R}^d$ via distribution theory in Sriperumbudur et al. [JMLR10, Corollary 10 p. 1535]
Two-Sample Testing with MMD
A statistical test using MMD

The empirical MMD:

$$\widehat{\text{MMD}}^2_\lambda(X_m, Y_n) = \frac{1}{m(m-1)} \sum_{i \neq j} k_\lambda(x_i, x_j) + \frac{1}{n(n-1)} \sum_{i \neq j} k_\lambda(y_i, y_j)$$

$$- \frac{2}{mn} \sum_{i,j} k_\lambda(x_i, y_j)$$

Two-sample MMD test:

$$\Delta^\lambda_\alpha(X_m, Y_n) := \mathbb{1} \left( \widehat{\text{MMD}}^2_\lambda(X_m, Y_n) > q^{\lambda}_{1-\alpha} \right)$$
A statistical test using MMD

The empirical MMD:

$$\hat{\text{MMD}}^2_{\lambda}(X_m, Y_n) = \frac{1}{m(m-1)} \sum_{i \neq j} k_\lambda(x_i, x_j) + \frac{1}{n(n-1)} \sum_{i \neq j} k_\lambda(y_i, y_j) \quad - \frac{2}{mn} \sum_{i,j} k_\lambda(x_i, y_j)$$

Two-sample MMD test:

$$\Delta^\lambda_{\alpha}(X_m, Y_n) := \mathbb{1}\left(\hat{\text{MMD}}^2_{\lambda}(X_m, Y_n) > q_{1-\alpha}^\lambda\right)$$

Want threshold $q_{1-\alpha}^\lambda$ for test $\hat{\text{MMD}}^2_{\lambda}(X_m, Y_n)$ to get false positive rate $\alpha$
Asymptotics of $\hat{MMD}^2$ when $P = Q$

$P = Q = \mathcal{N}(0, 1)$, statistic has asymptotic distribution

$$(m + n)\hat{MMD}_\lambda^2(X_m, Y_n) \sim \sum_{l=1}^{\infty} \lambda_l \left[ z_l^2 - 2 \right]$$

where

$$\lambda_i \psi_i(x') = \int_{\chi} \tilde{k}_\lambda(x, x') \psi_i(x) dP(x)$$

$z_l \sim \mathcal{N}(0, 2)$ i.i.d.
How do we get the test threshold $q^\lambda_{1-\alpha}$?

Original empirical MMD for dogs and fish:

$$X = \begin{bmatrix} \text{dog} & \text{dog} & \ldots \end{bmatrix}$$

$$Y = \begin{bmatrix} \text{fish} & \text{fish} & \ldots \end{bmatrix}$$

$$\overrightarrow{\text{MMD}}^2 = \frac{1}{n(n-1)} \sum_{i \neq j} k(x_i, x_j)$$
$$+ \frac{1}{n(n-1)} \sum_{i \neq j} k(y_i, y_j)$$
$$- \frac{2}{n^2} \sum_{i,j} k(x_i, y_j)$$
How do we get test threshold $q_{1-\alpha}^\lambda$?

Permuted dog and fish samples (merdogs):

\[
\tilde{X} = \begin{bmatrix}
\text{fish} & \text{dog} & \text{fish} & \ldots
\end{bmatrix}
\]

\[
\tilde{Y} = \begin{bmatrix}
\text{dog} & \text{fish} & \text{dog} & \ldots
\end{bmatrix}
\]
How do we get test threshold \( q_{1-\alpha} \)?

Permuted dog and fish samples (merdogs):

\[
\tilde{X} = [\text{fish} \quad \text{dog} \quad \ldots]
\]

\[
\tilde{Y} = [\text{dog} \quad \text{fish} \quad \ldots]
\]

\[
\overline{\text{MMD}}^2_\lambda (X_m, Y_n) = \\
\frac{1}{m(m-1)} \sum_{i \neq j} k(\tilde{x}_i, \tilde{x}_j) + \\
\frac{1}{n(n-1)} \sum_{i \neq j} k(\tilde{y}_i, \tilde{y}_j) + \\
- \frac{2}{mn} \sum_{i,j} k(\tilde{x}_i, \tilde{y}_j)
\]
MMD test thresholds: permutation, wild bootstrap

Two-sample MMD test:

\[ \Delta_{\alpha}^B(X_m, Y_n) := \mathbb{1}\left(\hat{\text{MMD}}^2_{\lambda}(X_m, Y_n) > \hat{q}_{1-\alpha}^{\lambda}\right) \]

Quantile: \( \hat{q}_{1-\alpha}^{\lambda} \) is the \([(B + 1)(1 - \alpha)]\)-th largest value of \( \hat{\text{MMD}}^2_{\lambda}(X_m, Y_n) \) and \( B \mathcal{H}_0 \)-simulated test statistics

Permutations: \( \hat{\text{MMD}}^2_{\lambda}(X_{m}^{\sigma}, Y_{n}^{\sigma}) \) where \( (X_{m}^{\sigma}, Y_{n}^{\sigma}) = \sigma(X_m \cup Y_n) \)
MMD test thresholds: permutation, wild bootstrap

Two-sample MMD test:

\[ \Delta_{\alpha}^B(X_m, Y_n) := 1 \left( \hat{\text{MMD}}^2_\lambda(X_m, Y_n) > \hat{q}^{\lambda}_{1-\alpha} \right) \]

Quantile: \( \hat{q}^{\lambda}_{1-\alpha} \) is the \( [(B + 1)(1 - \alpha)]\)-th largest value of \( \hat{\text{MMD}}^2_\lambda(X_m, Y_n) \) and \( B \) \( \mathcal{H}_0 \)-simulated test statistics

Permutations: \( \hat{\text{MMD}}^2_\lambda(X^\sigma_m, Y^\sigma_n) \) where \( (X^\sigma_m, Y^\sigma_n) = \sigma(X_m \cup Y_n) \)

Non-asymptotic level (permutation): \( \mathbb{P}_{p \times p}(\Delta_{\alpha}^B(X_m, Y_n) = 1) \leq \alpha \)

Time complexity: \( \mathcal{O}(B (m + n)^2) \)
Approx. null distribution of $MMD^2$ via permutation

Null distribution estimated from 500 permutations

Example: $P = Q = \mathcal{N}(0, 1)$
Kernel choice: MMD as an IPM

Maximum mean discrepancy: smooth function for $P$ vs $Q$

$$\text{MMD}_\lambda(p, q) := \sup_{\|f\|_F \leq 1} \left[ E_P f(X) - E_Q f(Y) \right]$$

Legend:
- $\text{f}$
- Gauss
- Laplace

Witness $f$ for Gauss and Laplace densities

<table>
<thead>
<tr>
<th>$f$</th>
<th>$-2$</th>
<th>$0$</th>
<th>$2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gauss</td>
<td>0.8</td>
<td>0.6</td>
<td>0.4</td>
</tr>
<tr>
<td>Laplace</td>
<td>0.2</td>
<td>0.4</td>
<td>0.6</td>
</tr>
</tbody>
</table>

Axes:
- $X$ (horizontal axis)
- Probability density and $f$ (vertical axis)
Kernel Choice: MMD as an IPM

- Simple choice: Gaussian

\[ k_\lambda(x, y) = \frac{1}{(\pi)^{d/2}} \prod_{i=1}^{d} \frac{1}{\lambda_i} \exp \left( -\frac{(x_i - y_i)^2}{\lambda_i^2} \right) \]

- Characteristic: for any \( \sigma \): for any \( P \) and \( Q \), power \( \rightarrow 1 \) as \( n \rightarrow \infty \)
Kernel Choice: MMD as an IPM

- Simple choice: Gaussian

\[
k_\lambda(x, y) = \frac{1}{(\pi)^{d/2}} \prod_{i=1}^{d} \frac{1}{\lambda_i} \exp\left( -\frac{(x_i - y_i)^2}{\lambda_i^2} \right)
\]

- \textit{Characteristic}: for any \(\sigma\): for any \(P\) and \(Q\), power \(\to 1\) as \(n \to \infty\)

- But choice of \(\lambda_1 \cdots \lambda_d\) is very important for finite \(m, n\)...
Kernel Choice: MMD as an IPM

- Simple choice: Gaussian

\[ k_\lambda(x, y) = \frac{1}{(\pi)^{d/2}} \prod_{i=1}^{d} \frac{1}{\lambda_i} \exp \left( -\frac{(x_i - y_i)^2}{\lambda_i^2} \right) \]

- **Characteristic:** for any \( \sigma \): for any \( P \) and \( Q \), power \( \rightarrow 1 \) as \( n \rightarrow \infty \)

- But choice of \( \lambda_1 \cdots \lambda_d \) is very important for finite \( m, n \ldots \)
Kernel Choice: MMD as an IPM

- Simple choice: Gaussian

\[ k_\lambda(x, y) = \frac{1}{(\pi)^{d/2}} \prod_{i=1}^{d} \frac{1}{\lambda_i} \exp \left( -\frac{(x_i - y_i)^2}{\lambda_i^2} \right) \]

- **Characteristic:** for any \( \sigma \): for any \( P \) and \( Q \), power \( \to 1 \) as \( n \to \infty \)
- But choice of \( \lambda_1 \cdots \lambda_d \) is very important for finite \( m, n \)...
Kernel Choice: MMD as an IPM

- Simple choice: Gaussian

\[ k_\lambda(x, y) = \frac{1}{(\pi)^{d/2}} \prod_{i=1}^{d} \frac{1}{\lambda_i} \exp \left( -\frac{(x_i - y_i)^2}{\lambda_i^2} \right) \]

- *Characteristic:* for any \( \sigma \): for any \( P \) and \( Q \), power \( \rightarrow 1 \) as \( n \rightarrow \infty \)
- But choice of \( \lambda_1 \cdots \lambda_d \) is very important for finite \( m, n \)...

![Graph showing kernel function](image-url)
Test power for known smoothness of $p - q$
Sobolev balls

Regularity/smoothness assumption: $p - q \in S_d^s(R)$

Sobolev balls:

$$S_d^s(R) := \left\{ f \in L^1(\mathbb{R}^d) \cap L^2(\mathbb{R}^d) : \int_{\mathbb{R}^d} \|\xi\|_2^{2s} |\hat{f}(\xi)|^2 \, d\xi \leq (2\pi)^d R^2 \right\}$$

radius $R > 0$

smoothness parameter $s > 0$

dimension $d$

Fourier transform $\hat{f}(\xi) := \int_{\mathbb{R}^d} f(x) e^{-ix^T \xi} \, dx$
**Sobolev balls**

Regularity/smoothness assumption: \( p - q \in S^s_d(R) \)

Sobolev balls:

\[
S^s_d(R) := \left\{ f \in L^1(\mathbb{R}^d) \cap L^2(\mathbb{R}^d) : \int_{\mathbb{R}^d} \|\xi\|_2^s |\hat{f}(\xi)|^2 \, d\xi \leq (2\pi)^d R^2 \right\}
\]

radius \( R > 0 \)

smoothness parameter \( s > 0 \)

dimension \( d \)

Fourier transform \( \hat{f}(\xi) := \int_{\mathbb{R}^d} f(x) e^{-ix^T\xi} \, dx \)
Theorem (MMD test minimax optimality)

For known smoothness $s$, assuming $p - q \in S^s_d(R)$ and setting

$$\lambda^*_i \equiv (m + n)^{-2/(4s+d)}$$

for $i = 1, \ldots, d$, the condition

$$\|p - q\|_2 \geq \frac{C}{\sqrt{\beta}} (m + n)^{-2s/(4s+d)}$$

 guarantees control of the type II error of the MMD test

$$\mathbb{P}_{p \times q \times r} \left( \Delta_{\alpha}^{B}(X_m, Y_n) = 0 \right) \leq \beta.$$

Minimax rate over Sobolev balls: $(m + n)^{-2s/(4s+d)}$
Proof of theorem 1 (next few slides)

For $\alpha, \beta \in (0, 1)$, Type II error control

$$\mathbb{P}_{p \times q \times r} \left( \Delta_{\lambda}^{B}(X_m, Y_n) = 0 \right) \leq \beta$$

is implied by (Chebyshev)

$$\mathbb{P}_{p \times q \times r} \left( \text{MMD}^2_{\lambda}(p, q) \geq \sqrt{\frac{2}{\beta} \text{var}_{p \times q} \left( \widehat{\text{MMD}}^2_{\lambda}(X_m, Y_n) \right) + \widehat{q}_{1-\alpha}^{\lambda}} \right) \geq 1 - \frac{\beta}{2}$$
Proof of theorem 1 (next few slides)

\[
\text{MMD}^2_{\lambda}(p, q) \geq \sqrt{\frac{2}{\beta}} \text{var}_{p \times q} \left( \widehat{\text{MMD}}^2_{\lambda}(X_m, Y_n) \right) + \hat{q}^\lambda_{1-\alpha}
\]

We address each of the three terms (A), (B), (C) in turn.
Breakdown of the MMD (A)

The MMD can be decomposed

$$\text{MMD}^2_{\lambda}(p, q) = \langle p - q, k_{\lambda} * (p - q) \rangle_2$$

$$= \frac{1}{2} \left( \|p - q\|_2^2 + \|k_{\lambda} * (p - q)\|_2^2 \right.$$

$$- \|k_{\lambda} * (p - q) - (p - q)\|_2^2 \right)$$
Breakdown of the MMD (A)

The MMD can be decomposed

\[
MMD^2_\lambda(p, q) = \langle p - q, k_\lambda \ast (p - q) \rangle_2
\]

\[
= \frac{1}{2} \left( \|p - q\|_2^2 + \|k_\lambda \ast (p - q)\|_2^2 - \|k_\lambda \ast (p - q) - (p - q)\|_2^2 \right)
\]

- Keep the first term (test “radius” for power 1 – \(\beta\)): \(\|p - q\|_2^2\)
- Get rid of second term using variance (next slides): \(\|k_\lambda \ast (p - q)\|_2^2\)
- Bound the final term: if \(p - q \in S^s_\delta(R)\), then \(\exists S \in (0, 1)\) such that

\[
\|k_\lambda \ast (p - q) - (p - q)\|_2^2 - S^2\|p - q\|_2^2 \leq C_0(d, s, R) \sum_{i=1}^{d} \lambda_i^{2s}
\]
Updating the power condition after (A)

The power condition (which needs to hold with probability $1 - \beta/2$)

$$\underbrace{\text{MMD}^2_\lambda(p, q)}_{(A)} \geq \sqrt{\frac{2}{\beta}} \underbrace{\text{var}_{p \times q} \left( \text{MMD}^2_\lambda(X_m, Y_n) \right)}_{(B)} + \underbrace{q^{1-\alpha}}_{(C)}$$
Updating the power condition after (A)

The power condition (which needs to hold with probability $1 - \beta/2$)

$$\text{MMD}_\lambda^2(p, q) \geq \sqrt{\frac{2}{\beta} \text{var}_{p \times q} \left( \text{MMD}_\lambda^2(X_m, Y_n) \right)} + \hat{q}_{1-\alpha}^\lambda,$$

after updating (A), then becomes

$$(1 - S^2) \|p - q\|_2^2 \geq C_0 \sum_{i=1}^{d} \lambda_i^{2s} - \|k_\lambda \ast (p - q)\|_2^2$$

$$+ 2 \sqrt{\frac{2}{\beta} \text{var}_{p \times q} \left( \text{MMD}_\lambda^2(X_m, Y_n) \right)} + 2 \hat{q}_{1-\alpha}^\lambda$$
Bound on the variance (B)

Assume that $\max(\|p\|_\infty, \|q\|_\infty) \leq M$ for some $M > 0$.

$$\text{var}_{p \times q} \left( \widehat{\text{MMD}}_{\lambda}^2(X_m, Y_n) \right)$$

$$\leq C_1(M, d) \left( \frac{\|k_\lambda * (p - q)\|_2^2}{m + n} + \frac{1}{(m + n)^2 \lambda_1 \cdots \lambda_d} \right).$$
Bound on the variance \((B)\)

Assume that \(\max(\|p\|_\infty, \|q\|_\infty) \leq M\) for some \(M > 0\).

\[
\begin{aligned}
\text{var}_{p \times q} \left( \widehat{\text{MMD}}_\lambda^2(X_m, Y_n) \right) \\
\leq C_1(M, d) \left( \frac{||k_\lambda * (p - q)||_2^2}{m + n} + \frac{1}{(m + n)^2 \lambda_1 \cdots \lambda_d} \right).
\end{aligned}
\]

Assuming \(\lambda_1 \cdots \lambda_d \leq 1\),

\[
(B) = 2 \sqrt{\frac{2}{\beta}} \text{var}_{p \times q} \left( \widehat{\text{MMD}}_\lambda^2(X_m, Y_n) \right) \\
\leq ||k_\lambda * (p - q)||_2^2 + \frac{C_1'}{\sqrt{\beta (m + n) \sqrt{\lambda_1 \cdots \lambda_d}}}.
\]

Term \(||k_\lambda * (p - q)||_2^2\) will cancel in the power condition.
Updating the power condition after (A), (B)

The power condition (which needs to hold with probability $1 - \beta / 2$)

$$(1 - S^2) \| p - q \|_2^2 \geq C_0 \sum_{i=1}^{d} \lambda_i^{2s} - \| k_\lambda \ast (p - q) \|_2^2$$

$$+ 2 \sqrt{\frac{2}{\beta}} \text{var}_{p \times q} \left( \overbrace{\text{MMD}^2_\lambda(X_m, Y_n)}^{(B)} \right) + 2 \widehat{q}^\lambda_{1-\alpha}.$$

$$(C)$$
Updating the power condition after (A), (B)

The power condition (which needs to hold with probability \(1 - \beta/2\))

\[
(1 - S^2)\|p - q\|_2^2 \geq C_0 \sum_{i=1}^{d} \lambda_i^{2s} - \|k_\lambda \ast (p - q)\|_2^2
\]

\[
+ 2\sqrt{\frac{2}{\beta}} \text{var}_{p \times q} \left( \widehat{\text{MMD}}_\lambda^2(X_m, Y_n) \right) + 2\hat{q}_1^{\lambda 1-\alpha}.
\]

(B)

After bounding (B), becomes

\[
(1 - S^2)\|p - q\|_2^2 \geq \frac{C'_1}{\sqrt{\beta} (m + n) \lambda_1 \cdots \lambda_d} + C_0 \sum_{i=1}^{d} \lambda_i^{2s} + 2\hat{q}_1^{\lambda 1-\alpha}.
\]

(C)
Bound on estimated $1 - \alpha$ quantile (C)

Assume that $\max(||p||_\infty, ||q||_\infty) \leq M$ for some $M > 0$. We have

$$
\mathbb{P}_{p \times q \times r} \left( \tilde{q}_{1-\alpha}^\lambda \leq C_2(M, d) \frac{\ln\left(\frac{1}{\alpha}\right)}{\sqrt{\beta (m + n) \sqrt{\lambda_1 \cdots \lambda_d}}} \right) \geq 1 - \frac{\beta}{2}
$$

for $B \geq \frac{3}{\alpha^2} \left( \ln\left(\frac{8}{\beta}\right) + \alpha(1 - \alpha) \right)$ and $\alpha \in (0, 0.5)$. 

[Schrab et al., 2023, Proposition 4]
Updating the power condition after (A), (B), (C)

The power condition (which needs to hold with probability $1 - \beta / 2$)

$$(1 - S^2)\|p - q\|_2^2 \geq \frac{C_1'}{\sqrt{\beta}(m + n)\sqrt{\lambda_1 \cdots \lambda_d}} + C_0 \sum_{i=1}^{d} \lambda_i^{2s} + 2\hat{q}_{1-\alpha}^{\lambda}.$$

Fine print: $\alpha \in (0, e^{-1})$, $B \geq \frac{3}{\alpha^2} \left( \ln\left(\frac{8}{\beta}\right) + \alpha(1 - \alpha) \right)$, and $\lambda_1 \cdots \lambda_d \leq 1$. 
Updating the power condition after (A), (B), (C)

The power condition (which needs to hold with probability $1 - \beta/2$)

$$(1 - S^2)\|p - q\|_2^2 \geq \frac{C_1'}{\sqrt{\beta} (m + n) \sqrt{\lambda_1 \cdots \lambda_d}} + C_0 \sum_{i=1}^{d} \lambda_i^{2s} + 2\hat{q}^{\lambda}_{1-\alpha}.$$  

After updating (C)

$$\|p - q\|_2^2 \geq \frac{C_4(M, d, s, S, R)}{\sqrt{\beta}} \left( \sum_{i=1}^{d} \lambda_i^{2s} + \frac{\ln\left(\frac{1}{\alpha}\right)}{(m + n) \sqrt{\lambda_1 \cdots \lambda_d}} \right).$$

Fine print: $\alpha \in (0, e^{-1})$, $B \geq \frac{3}{\alpha^2} (\ln\left(\frac{8}{\beta}\right) + \alpha(1 - \alpha))$, and $\lambda_1 \cdots \lambda_d \leq 1$. 
Updating the power condition after (A), (B), (C)

The power condition (which needs to hold with probability $1 - \beta / 2$)

$$(1 - S^2)\|p - q\|_2^2 \geq \frac{C'_1}{\sqrt{\beta}(m + n)\sqrt{\lambda_1 \cdots \lambda_d}} + C_0 \sum_{i=1}^{d} \lambda_i^{2s} + 2\tilde{q}_{1 - \alpha}^\lambda.$$  

After updating (C)

$$\|p - q\|_2^2 \geq \frac{C_4(M, d, s, S, R)}{\sqrt{\beta}} \left( \sum_{i=1}^{d} \lambda_i^{2s} + \frac{\ln\left(\frac{1}{\alpha}\right)}{(m + n)\sqrt{\lambda_1 \cdots \lambda_d}} \right).$$

Picking $\lambda_i^* := (m + n)^{-2/(4s + d)}$ controls the Type II error when

$$\|p - q\|_2 \geq \frac{C}{\sqrt{\beta}}(m + n)^{-2s/(4s + d)}.$$

Fine print: $\alpha \in (0, e^{-1})$, $B \geq \frac{3}{\alpha^2}\left(\ln\left(\frac{8}{\beta}\right) + \alpha(1 - \alpha)\right)$, and $\lambda_1 \cdots \lambda_d \leq 1$. 
Optimizing kernel parameters: aggregation
MMDAgg for a collection of bandwidths $\Lambda$

MMDAgg (MMD Aggregation): non-asymptotic level $\alpha$

$$\Delta^\Lambda_{\alpha}(X_m, Y_n) := 1 \left( \max_{\lambda \in \Lambda} (\tilde{\text{MMD}}^2_{\lambda}(X_m, Y_n) - \hat{q}^\lambda_{1-u}\bar{w}_\lambda) > b_{\alpha} \right)$$

positive weights $(\bar{w}_\lambda)_{\lambda \in \Lambda}$ satisfying $\sum_{\lambda \in \Lambda} w_\lambda \leq 1$

Correction $u_\alpha$ defined as

$$\sup\left\{ u > 0 : \mathbb{P}_{p \times p} \left( \max_{\lambda \in \Lambda} (\tilde{\text{MMD}}^2_{\lambda}(X_m, Y_n) - \hat{q}^\lambda_{1-u}\bar{w}_\lambda) > 0 \right) \leq \alpha \right\}$$

more powerful than Bonferroni correction as $u_\alpha \geq \alpha$

Time complexity $O\left( |\Lambda| (B_1 + B_2) (m + n)^2 \right)$
**MMDAgg for a collection of bandwidths $\Lambda$**

*MMDAgg* (MMD Aggregation): non-asymptotic level $\alpha$

$$\Delta^\Lambda_\alpha(X_m, Y_n) := 1 \left( \overline{\text{MMD}^2_{\lambda}(X_m, Y_n)} > \tilde{q}^\lambda_{1-u_\alpha w_\lambda} \text{ for some } \lambda \in \Lambda \right)$$

positive weights $(w_\lambda)_{\lambda \in \Lambda}$ satisfying $\sum_{\lambda \in \Lambda} w_\lambda \leq 1$

Correction $u_\alpha$ defined as

$$\sup\left\{ u > 0 : \mathbb{P}_{p \times p} \left( \max_{\lambda \in \Lambda} \left( \overline{\text{MMD}^2_{\lambda}(X_m, Y_n)} - \tilde{q}^\lambda_{1-u w_\lambda} \right) > 0 \right) \leq \alpha \right\}$$

more powerful than Bonferroni correction as $u_\alpha \geq \alpha$

Time complexity $O\left( |\Lambda| (B_1 + B_2) (m + n)^2 \right)$
MMDAgg for a collection of bandwidths \( \Lambda \)

MMDAgg (MMD Aggregation): non-asymptotic level \( \alpha \)

\[
\Delta_{\alpha}^{\Lambda}(X_m, Y_n) := 1\left( \widehat{\text{MMD}}_\lambda^2(X_m, Y_n) > \widehat{q}_{1-u\alpha}^\lambda w_\lambda \text{ for some } \lambda \in \Lambda \right)
\]

positive weights \((w_\lambda)_{\lambda \in \Lambda}\) satisfying \(\sum_{\lambda \in \Lambda} w_\lambda \leq 1\)

Correction \(u_\alpha\) defined as

\[
\sup \left\{ u > 0 : \mathbb{P}_{p \times p} \left( \max_{\lambda \in \Lambda} \left( \widehat{\text{MMD}}_\lambda^2(X_m, Y_n) - \widehat{q}_{1-u\alpha}^\lambda w_\lambda \right) > 0 \right) \leq \alpha \right\}
\]

more powerful than Bonferroni correction as \(u_\alpha \geq \alpha\)

Time complexity \(\mathcal{O}\left( |\Lambda| (B_1 + B_2)(m + n)^2 \right)\)
MMDAgg for a collection of bandwidths $\Lambda$

MMDAgg (MMD Aggregation): non-asymptotic level $\alpha$

$$
\Delta^\Lambda_{\alpha}(X_m, Y_n) := \mathbb{1}\left(\overline{\text{MMD}}_\lambda^2(X_m, Y_n) > \hat{q}^\lambda_{1-u_\alpha} w_\lambda \text{ for some } \lambda \in \Lambda\right)
$$

positive weights $(w_\lambda)_{\lambda \in \Lambda}$ satisfying $\sum_{\lambda \in \Lambda} w_\lambda \leq 1$

Correction $u_\alpha$ defined as

$$
\sup \left\{ u > 0 : \mathbb{P}_{p \times p} \left( \max_{\lambda \in \Lambda} \left( \overline{\text{MMD}}_\lambda^2(X_m, Y_n) - \hat{q}^\lambda_{1-u} w_\lambda \right) > 0 \right) \leq \alpha \right\}
$$

more powerful than Bonferroni correction as $u_\alpha \geq \alpha$

Time complexity $O\left(|\Lambda| (B_1 + B_2) (m + n)^2\right)$
Multiple testing correction comparison

Simple example: 3-d Gaussians with different means

\[ \Lambda(i) := \mathcal{2}^l \lambda_{\text{med}} : l \in \{-i, \ldots, i\} \] for \( i \in \{0, 10, 20, 30, 40, 50\} \)

\[ w_{\lambda} := \frac{1}{|\Lambda|} \]
MMDAgg test power guarantee

**Theorem (MMDAgg minimax adaptivity)**

\[ \Lambda^* := \left\{ 2^{-\ell} \mathbb{1}_d : \ell \in \{1, \ldots, \left\lceil \frac{2}{d} \log_2 \left( \frac{m + n}{\ln(\ln(m + n))} \right) \right\rceil \} \right\}, \quad \omega_\lambda := \frac{6}{\pi^2 \ell^2} \]

Assuming \( p - q \in S_d^s(R) \), the condition

\[ \|p - q\|_2 \geq \frac{C}{\sqrt{\beta}} \left( \frac{m + n}{\ln(\ln(m + n))} \right)^{-2s/(4s+d)} \]

guarantees control of the type II error of MMDAgg

\[ \mathbb{P}_{p \times q} \left( \Delta^\Lambda^*_\alpha (X_m, Y_n) = 0 \right) \leq \beta. \]

Minimax rate over Sobolev balls: \((m + n)^{-2s/(4s+d)}\)

Minimax adaptive over \(\{S_d^s(R) : s > 0, R > 0\}\)

Unlike the MMD test \(\Delta^\lambda^*_\alpha\), MMDAgg \(\Delta^\Lambda^*_\alpha\) is independent of \(s\)
MMDAgg parameter-free user-friendly implementation

Radial basis function (RBF) kernel: \( k_\lambda(x, y) := K\left(\frac{x - y}{\lambda}\right) \)

Collection of bandwidths \( \Lambda \): discretisation of the interval \([\lambda_{\text{min}}, \lambda_{\text{max}}]\)
where \( \lambda_{\text{min}} \) and \( \lambda_{\text{max}} \) are the (robust) minimum and maximum of
\[
\left\{ \|x - y\| : x \in X_m, y \in Y_n \right\}
\]

Possible to aggregate several kernels each with multiple bandwidths

Uniform weights: \( w_\lambda := 1 / |\Lambda| \)

Number of permutations / wild bootstraps: \( B_1 = B_2 = 2000 \)

**JAX**: runs on either CPU or GPU (significant speed improvements)

- JAX GPU runs 100 times faster than Numpy CPU

**mmdagg package**: [github.com/antoninschrab/mmdagg](https://github.com/antoninschrab/mmdagg)

```python
from mmdagg import mmdagg
output = mmdagg(X, Y) # 0 or 1
```

# X shape (m, d)
# Y shape (n, d)
Radial basis function (RBF) kernel: \( k_{\lambda}(x, y) := K\left(\frac{||x - y||}{\lambda}\right) \)

Collection of bandwidths \( \Lambda \): discretisation of the interval \([\lambda_{\text{min}}, \lambda_{\text{max}}]\) where \( \lambda_{\text{min}} \) and \( \lambda_{\text{max}} \) are the (robust) minimum and maximum of \( \{||x - y|| : x \in \mathbb{X}_m, y \in \mathbb{Y}_n\} \)

Possible to aggregate several kernels each with multiple bandwidths

Uniform weights: \( w_{\lambda} := 1 / |\Lambda| \)

Number of permutations / wild bootstraps: \( B_1 = B_2 = 2000 \)

JAX: runs on either CPU or GPU (significant speed improvements)
- JAX GPU runs 100 times faster than Numpy CPU

mmdagg package: github.com/antoninschrab/mmdagg

```python
from mmdagg import mmdagg
output = mmdagg(X, Y) ≠ 0 or 1
```
MMDAgg parameter-free user-friendly implementation

Radial basis function (RBF) kernel: \( k_\lambda(x, y) := K\left(\frac{||x - y||}{\lambda}\right) \)

Collection of bandwidths \( \Lambda \): discretisation of the interval \([\lambda_{\text{min}}, \lambda_{\text{max}}]\)
where \( \lambda_{\text{min}} \) and \( \lambda_{\text{max}} \) are the (robust) minimum and maximum of
\[
\left\{ ||x - y|| : x \in X_m, y \in Y_n \right\}
\]

Possible to aggregate several kernels each with multiple bandwidths

Uniform weights: \( w_\lambda := 1 / |\Lambda| \)

Number of permutations / wild bootstraps: \( B_1 = B_2 = 2000 \)

JAX: runs on either CPU or GPU (significant speed improvements)

JAX GPU runs 100 times faster than Numpy CPU

mmdagg package: github.com/antoninschrab/mmdagg

```python
from mmdagg import mmdagg
output = mmdagg(X, Y) # 0 or 1 # Y shape (n, d)
```
MMDAgg parameter-free user-friendly implementation

Radial basis function (RBF) kernel: $k_\lambda(x, y) := K\left(\frac{x - y}{\lambda}\right)$

Collection of bandwidths $\Lambda$: discretisation of the interval $[\lambda_{\text{min}}, \lambda_{\text{max}}]$ where $\lambda_{\text{min}}$ and $\lambda_{\text{max}}$ are the (robust) minimum and maximum of

$$\left\{|x - y| : x \in X_m, y \in Y_n\right\}$$

Possible to aggregate several kernels each with multiple bandwidths

Uniform weights: $w_\lambda := 1 / |\Lambda|$

Number of permutations / wild bootstraps: $B_1 = B_2 = 2000$

JAX: runs on either CPU or GPU (significant speed improvements)

JAX GPU runs 100 times faster than Numpy CPU

mmdagg package: github.com/antoninschrab/mmdagg

```python
from mmdagg import mmdagg
output = mmdagg(X, Y)  # 0 or 1
```

# X shape (m, d)
# Y shape (n, d)
MMDAgg parameter-free user-friendly implementation

Radial basis function (RBF) kernel: \( k_\lambda(x, y) := K\left(\frac{x - y}{\lambda}\right) \)

Collection of bandwidths \( \Lambda \): discretisation of the interval \([\lambda_{\text{min}}, \lambda_{\text{max}}]\)
where \( \lambda_{\text{min}} \) and \( \lambda_{\text{max}} \) are the (robust) minimum and maximum of
\[
\left\{|x - y| : x \in X_m, y \in Y_n\right\}
\]
Possible to aggregate several kernels each with multiple bandwidths

Uniform weights: \( w_\lambda := 1 / |\Lambda| \)

Number of permutations / wild bootstraps: \( B_1 = B_2 = 2000 \)

JAX: runs on either CPU or GPU (significant speed improvements)

JAX GPU runs 100 times faster than Numpy CPU

mmdagg package: github.com/antoninschrab/mmdagg

```python
from mmdagg import mmdagg
output = mmdagg(X, Y) # 0 or 1
```

# X shape (m, d)
# Y shape (n, d)
MMDAgg parameter-free user-friendly implementation

Radial basis function (RBF) kernel: \( k_\lambda(x, y) := K\left(\frac{\|x - y\|}{\lambda}\right) \)

Collection of bandwidths \( \Lambda \): discretisation of the interval \([\lambda_{\text{min}}, \lambda_{\text{max}}]\) where \( \lambda_{\text{min}} \) and \( \lambda_{\text{max}} \) are the (robust) minimum and maximum of

\[
\left\{\|x - y\| : x \in \mathbb{X}_m, y \in \mathbb{Y}_n\right\}
\]

Possible to aggregate several kernels each with multiple bandwidths

Uniform weights: \( w_\lambda := 1 / |\Lambda| \)

Number of permutations / wild bootstraps: \( B_1 = B_2 = 2000 \)

JAX: runs on either CPU or GPU (significant speed improvements)

- JAX GPU runs 100 times faster than Numpy CPU

mmdagg package: github.com/antoninschrab/mmdagg

```python
from mmdagg import mmdagg  # X shape (m, d)
output = mmdagg(X, Y)  # 0 or 1  # Y shape (n, d)
```
Radial basis function (RBF) kernel: $k_\lambda(x, y) := K\left(\left\| \frac{x - y}{\lambda} \right\| \right)$

Collection of bandwidths $\Lambda$: discretisation of the interval $[\lambda_{\text{min}}, \lambda_{\text{max}}]$ where $\lambda_{\text{min}}$ and $\lambda_{\text{max}}$ are the (robust) minimum and maximum of

$$\left\{ \| x - y \| : x \in X_m, y \in Y_n \right\}$$

Possible to aggregate several kernels each with multiple bandwidths

Uniform weights: $w_\lambda := 1 / |\Lambda|$

Number of permutations / wild bootstraps: $B_1 = B_2 = 2000$

**JAX**: runs on either CPU or GPU (significant speed improvements)

- JAX GPU runs 100 times faster than Numpy CPU

mmdagg package: [github.com/antoninschrab/mmdagg](https://github.com/antoninschrab/mmdagg)

``` python
from mmdagg import mmdagg
output = mmdagg(X, Y) # 0 or 1
```

# X shape (m, d)

# Y shape (n, d)
MMDAgg parameter-free user-friendly implementation

Radial basis function (RBF) kernel: \( k_\lambda(x, y) := K\left(\frac{x - y}{\lambda}\right) \)

Collection of bandwidths \( \Lambda \): discretisation of the interval \([\lambda_{\min}, \lambda_{\max}]\) where \( \lambda_{\min} \) and \( \lambda_{\max} \) are the (robust) minimum and maximum of \( \{\|x - y\| : x \in X_m, y \in Y_n\} \)

Possible to aggregate several kernels each with multiple bandwidths

Uniform weights: \( w_\lambda := 1 / |\Lambda| \)

Number of permutations / wild bootstraps: \( B_1 = B_2 = 2000 \)

JAX: runs on either CPU or GPU (significant speed improvements)

- JAX GPU runs 100 times faster than Numpy CPU

mmdagg package: github.com/antoninschrab/mmdagg

```python
from mmdagg import mmdagg
output = mmdagg(X, Y)  # 0 or 1
```

\# X shape (m, d)

\# Y shape (n, d)
Experiment on perturbed uniform $d = 1$
### Experiment on perturbed uniform \( d = 1 \)

<table>
<thead>
<tr>
<th>Number of perturbations</th>
<th>Power</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>0.25</td>
<td></td>
</tr>
<tr>
<td>0.50</td>
<td></td>
</tr>
<tr>
<td>0.75</td>
<td></td>
</tr>
<tr>
<td>1.00</td>
<td></td>
</tr>
</tbody>
</table>

**Legend:**
- MMDAgg
- MMD median
- MMD split
- MMD extra data
- AutoML

![Graph showing power changes with number of perturbations]
Experiment on perturbed uniform $d = 2$
Experiment on perturbed uniform $d = 2$
Experiment on MNIST digits
Experiment on MNIST digits
Experiment on MNIST digits

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>7</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>9</td>
</tr>
</tbody>
</table>
Experiment on MNIST digits
Experiment on MNIST digits
Experiment on MNIST digits
Experiment on MNIST digits

Number of digits missing from one sample

0.00
0.25
0.50
0.75
1.00

Power

MMDAgg
MMD median
MMD split
MMD extra data
AutoML

43/49
Experiment on image shifts on MNIST & CIFAR-10

Failing Loudly Benchmark: Rabanser et al., 2019
Experiment on image shifts on MNIST & CIFAR-10

Failing Loudly Benchmark: Rabanser et al., 2019
Experiment on image shifts on MNIST & CIFAR-10

Number of samples

Power

MMDAgg

AutoML
MMD kernel choice without data splitting

MMD Aggregated Two-Sample Test (JMLR 2023):

[Submitted on 28 Oct 2021 (v1), last revised 29 May 2023 (this version, v3)]

MMD Aggregated Two-Sample Test

Antonin Schrab, Ilmun Kim, Mélisande Albert, Béatrice Laurent, Benjamin Guedj, Arthur Gretton

Code:

https://github.com/antoninschrab/mmdagg-paper
Research support

Work supported by:

The Gatsby Charitable Foundation

Google Deepmind
Questions?