Optimal maximum mean discrepancy quantization

With Clustered Lasso on the sparse simplex

Sébastien Da Veiga, ENSAI / CREST — LIKE23 — 06/27/2023

Outline

- Context – Space-filling design of experiments
- Quantization with the maximum mean discrepancy
- New formulation with the Clustered Lasso on the sparse simplex

CONTEXT

SPACE FILLING DESIGN OF EXPERIMENTS

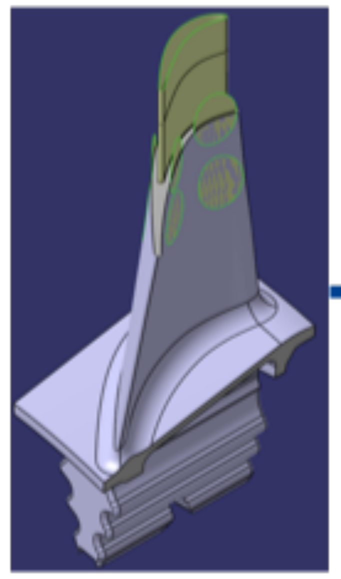
Design of experiments (DOE) — General principle

- **Defining a DOE = choosing points in a pre-defined parameter space**
  - Each point will then be evaluated to collect the corresponding value of the outputs of interest (via an experimental protocol, a production process observation, a numerical simulator, …)
  - **In general this evaluation is costly (time/money), which means that the DOE must be carefully chosen**

- **Objective: explore the output behavior thanks to a limited number of evaluations**
  - Optimize the information: identify regions of interest (safety, optimization), detect influential parameters, quantify their impact, …
  - Generate a DOE to build a regression model

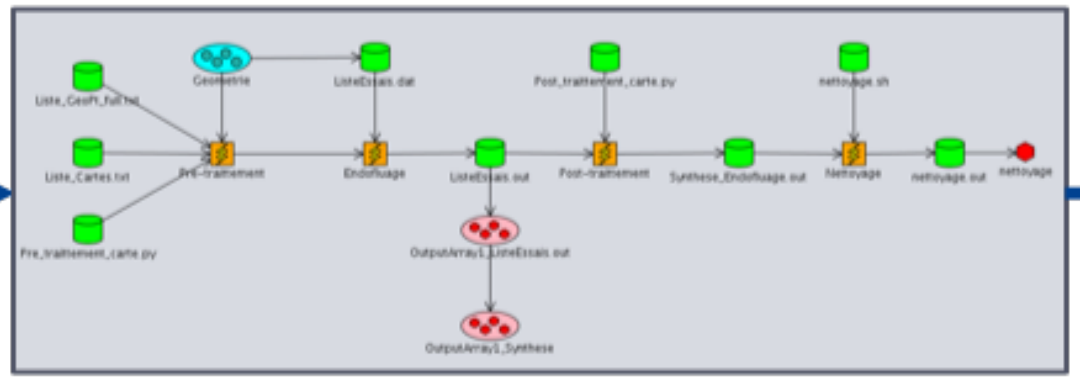
Design of experiments (DOE) — Examples @ Safran

ARRIEL 2 HP blade (SHE)



Vary the geometry to reproduce tolerance intervals

Mechanical Analysis (~2H)



Criteria

Creep damage
Constraints

Exploration: study the impact of tolerances on the mechanical behavior

Variable compressor van (SAE)

Vary the van angles

Expensive experiment on test bench

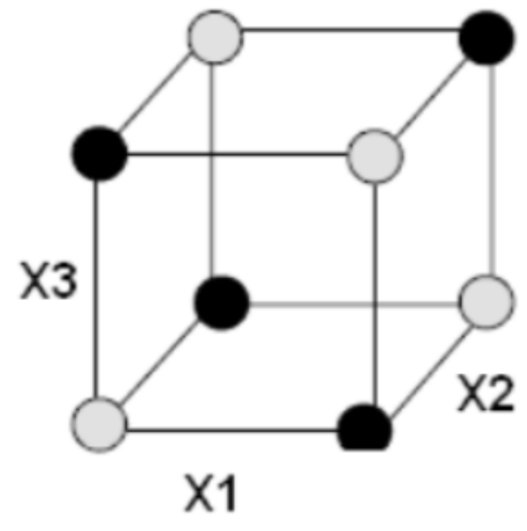
Criteria

Temperature
Margin

Optimization: find the van angles which lead to the best performance

Design of experiments (DOE) — Standard strategies

- **Well-known DOE: factorial design (i.e. a grid)**
  - Parameter discretization on n levels, budget n^d if d parameters (ex: 10 params / 2 lvls = 1024)
  - Extensions to lower the budget (fractional, centered composite, Box-Behnken, …)



- **Limitations: size and underlying model assumption (i.e. linear, 2nd order poly., …)**
  - **If the output does not vary according to the model, the amount of information given by the DOE is poor**
  - Very bad projection properties in general (focus for another talk)

Design of experiments (DOE) — Numerical experiments

- **« Numerical experiments » introduced a fresh point of view**
  - ◉ Main principles
    1. Do not assume an overly simplified model
    2. Ensure some DOE properties w.r.t. some possible output behavior
  - ◉ What we usually expect in numerical experiments
    1. Large input variations which imply nonlinear response for the outputs
    2. Very often the outputs have a low effective dimension

Design of experiments (DOE) — Numerical experiments

- **« Numerical experiments » introduced a fresh point of view**
  - ◉ Main principles

    1. Do not assume an overly simplified model

    2. Ensure some DOE properties w.r.t. some possible output behavior

  - ◉ What we usually expect in numerical experiments

    1. Large input variations which imply nonlinear response for the outputs

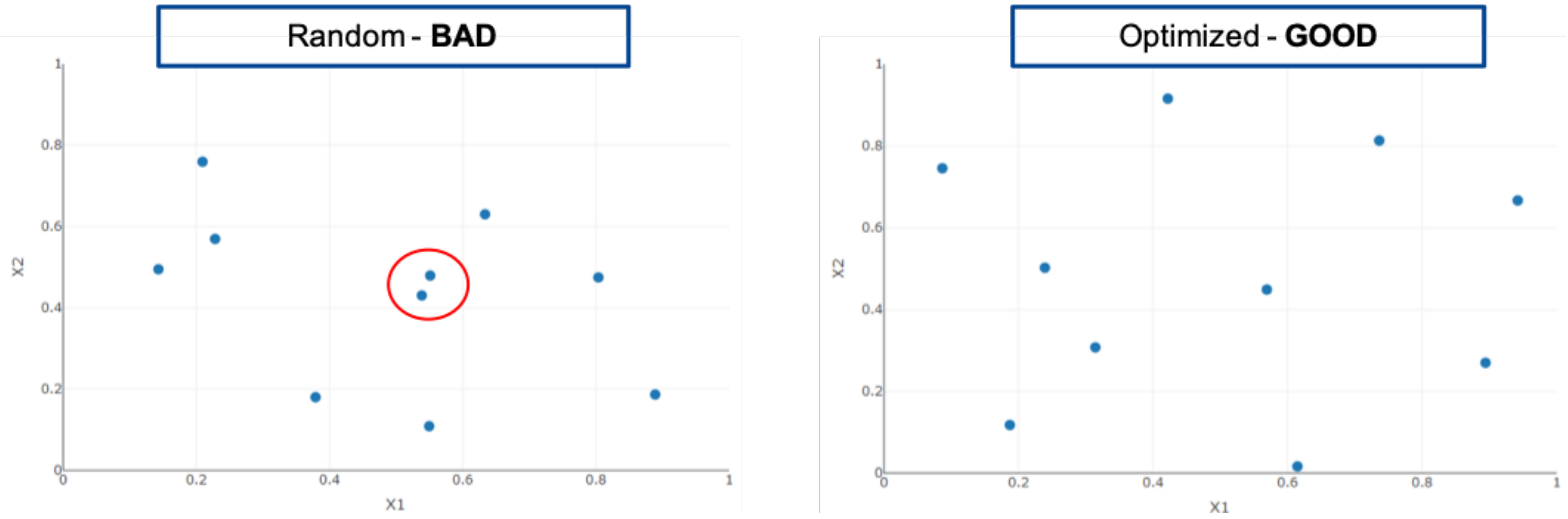
    2. Very often the outputs have a low effective dimension

Property 1
Space-filling

Property 2
Good projection properties

Design of experiments (DOE) — Space filling?

- **Fact: a random sample (Monte-Carlo) is very bad**
  - ◉ Some points are too close, holes in the space
  - ◉ How to mathematically define « space-filling »?



Design of experiments (DOE) — Space filling?

- **Family 1: Geometrical criteria**

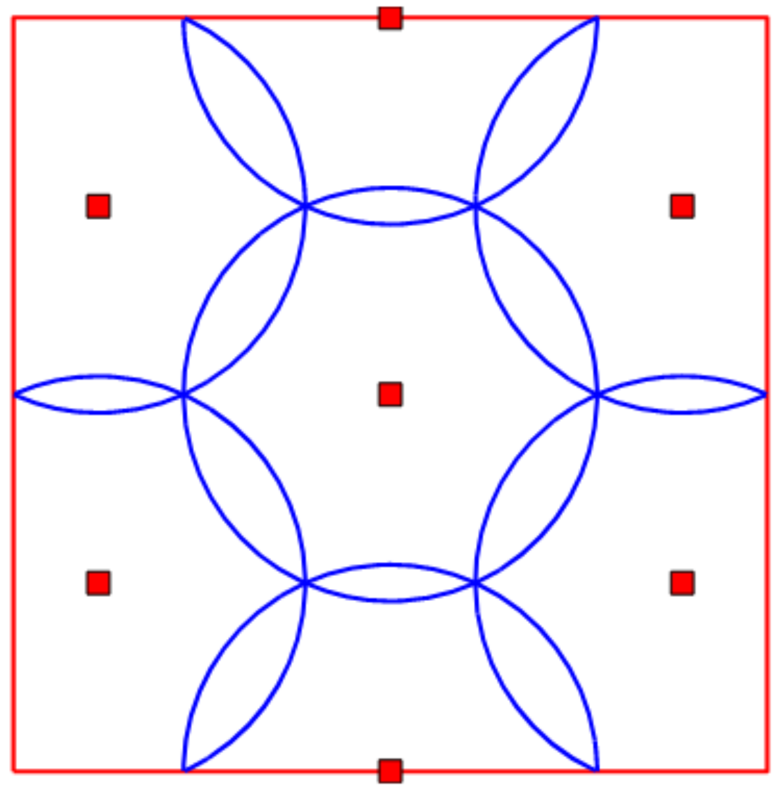
◉ Minimax DOE

- Minimize the maximal distance between any point in the space and the DOE (i.e. smallest possible holes)

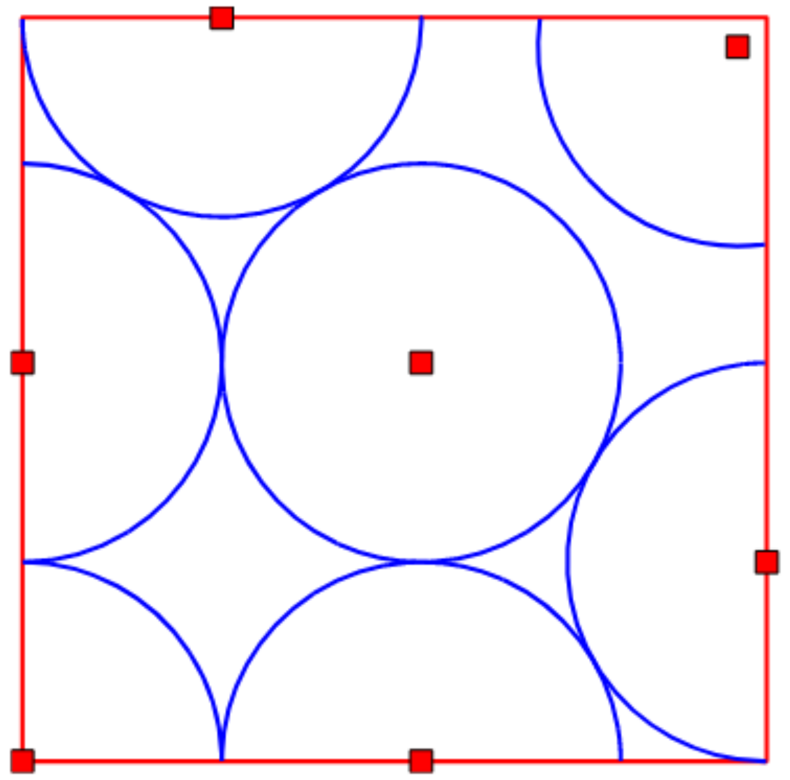
◉ Maximin DOE

- Maximize the minimal distance between points (i.e. limit cluster effect)

① miniMax $d = 2, n = 7$
(radius=$\phi_{mM}(\mathbf{X}_n)$)



② Maximin $d = 2, n = 7$
(radius=$\phi_{Mm}(\mathbf{X}_n)/2$)



Courtesy of L. Pronzato

Design of experiments (DOE) — Space filling?

- **Family 2: Discrepancy criteria**

$$D_n(\mathscr{B}, \mathbf{X}_n) \triangleq \sup_{\mathbb{B} \in \mathscr{B}} \left| \frac{\text{nb. of } \mathbf{x}_i \text{ in } \mathbb{B}}{n} - \text{vol}(\mathbb{B}) \right|$$

with $\mathscr{B}$ a family of subsets of $\mathbb{I}_d$ ($\Rightarrow 0 \leq D_n(\mathscr{B}, \mathbf{X}_n) \leq 1$)

- Goal: have points as close as possible to the uniform distribution
- Changing $\mathscr{B}$ yields different discrepancies
- Point of view justified by QMC integration

Design of experiments (DOE) — Space filling?

- **Koksma-Hlawka inequality (1961)**

$$\left| \int_{\mathbb{I}^d} f(\mathbf{u}) d\mathbf{u} - \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{x}_i) \right| \leq V(f) \; D_n^\star(\mathbf{X}_n)$$

◉ $V(f)$: Hardy-Krause variation, independent of the chosen points

◉ Star-discrepancy defined with subsets $\prod_{l=1}^{d} [0,u_l)$, independent of the function

Design of experiments (DOE) — Space filling?

- **In practice**
  - ◉ Star-discrepancy difficult to compute, bounded by extreme-discrepancy but not practical either
  - ◉ Two available roads:

    1. Use low-discrepancy sequences (Sobol, Halton, Faure, …) $\propto \dfrac{\log(n)^d}{n}$

    2. Change subset family to get analytical expressions

$$D_{Cent,L_2}(\mathbf{X}_n) = \left[\left(\frac{13}{12}\right)^d - \frac{2}{n}\sum_{k=1}^{n}\prod_{i=1}^{d}\left(1+\frac{1}{2}\left|\{\mathbf{x}_k\}_i - \frac{1}{2}\right| - \frac{1}{2}\left|\{\mathbf{x}_k\}_i - \frac{1}{2}\right|^2\right)\right.$$
$$\left. + \frac{1}{n^2}\sum_{k,k'=1}^{n}\prod_{i=1}^{d}\left(1+\frac{1}{2}\left|\{\mathbf{x}_k\}_i - \frac{1}{2}\right| + \frac{1}{2}\left|\{\mathbf{x}_{k'}\}_i - \frac{1}{2}\right| - \frac{1}{2}\left|\{\mathbf{x}_k\}_i - \{\mathbf{x}_{k'}\}_i\right|\right)\right]^{1/2}$$

$$D_{WA,L_2}(\mathbf{X}_n) = \left\{\frac{1}{n^2}\sum_{k,k'=1}^{n}\prod_{i=1}^{d}\left[\frac{3}{2} - \left|\{\mathbf{x}_k\}_i - \{\mathbf{x}_{k'}\}_i\right|\left(1 - \left|\{\mathbf{x}_k\}_i - \{\mathbf{x}_{k'}\}_i\right|\right)\right] - \left(\frac{4}{3}\right)^d\right\}^{1/2}$$

Hickernell 98

Design of experiments (DOE) — Space filling?

- **Discrepancy is nice, but it is only defined for comparing the DOE to the uniform distribution on the unit hypercube**

- **For practical applications**

  ◉ What if we need space-filling properties in more complex parameters spaces?

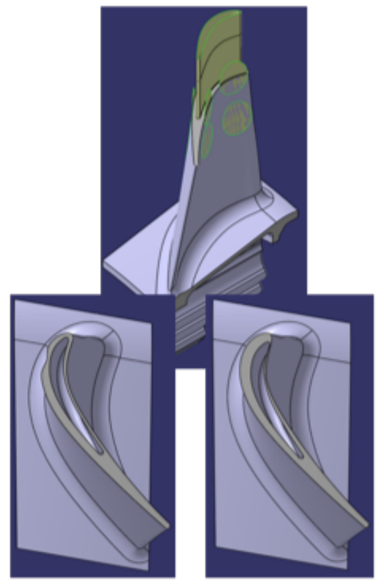
Variable vans design
(CoHP SC - SAE)

Only van values in the cross are admissible

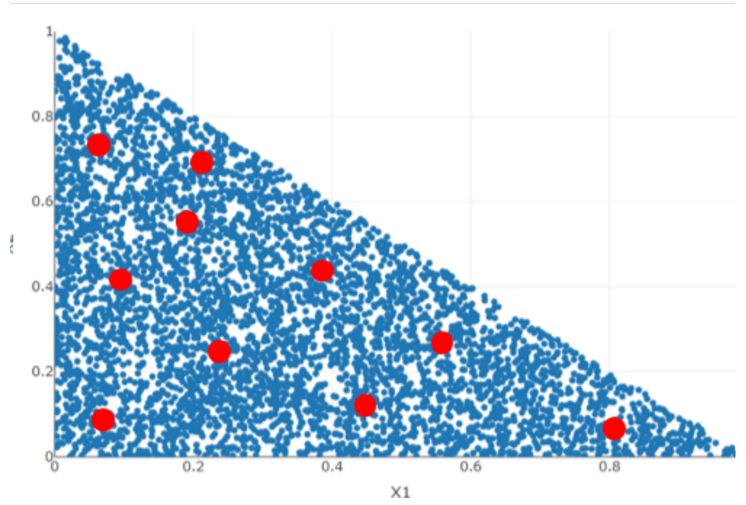
  ◉ What if we the DOE can only be chosen among a given set of points (subsampling)?

  - When the distribution is not known for example (accept/reject)
  - Or up to a constant (MCMC sample, related to optimal thinning)
  - Given database (splitting train/test set, …)

ARRIEL 2 HP blade (SHE)



Only geometrical parameters leading to a « physical » blade are allowed



Design of experiments (DOE) — Space filling?

- **A promising layer of generalization can be achieved with the use of recently introduced kernel-based methods**

  - Distance between probability distributions defined via kernel-embedding of distributions (aka maximum mean discrepancy)

  - Common discrepancies are obtained with specific kernels

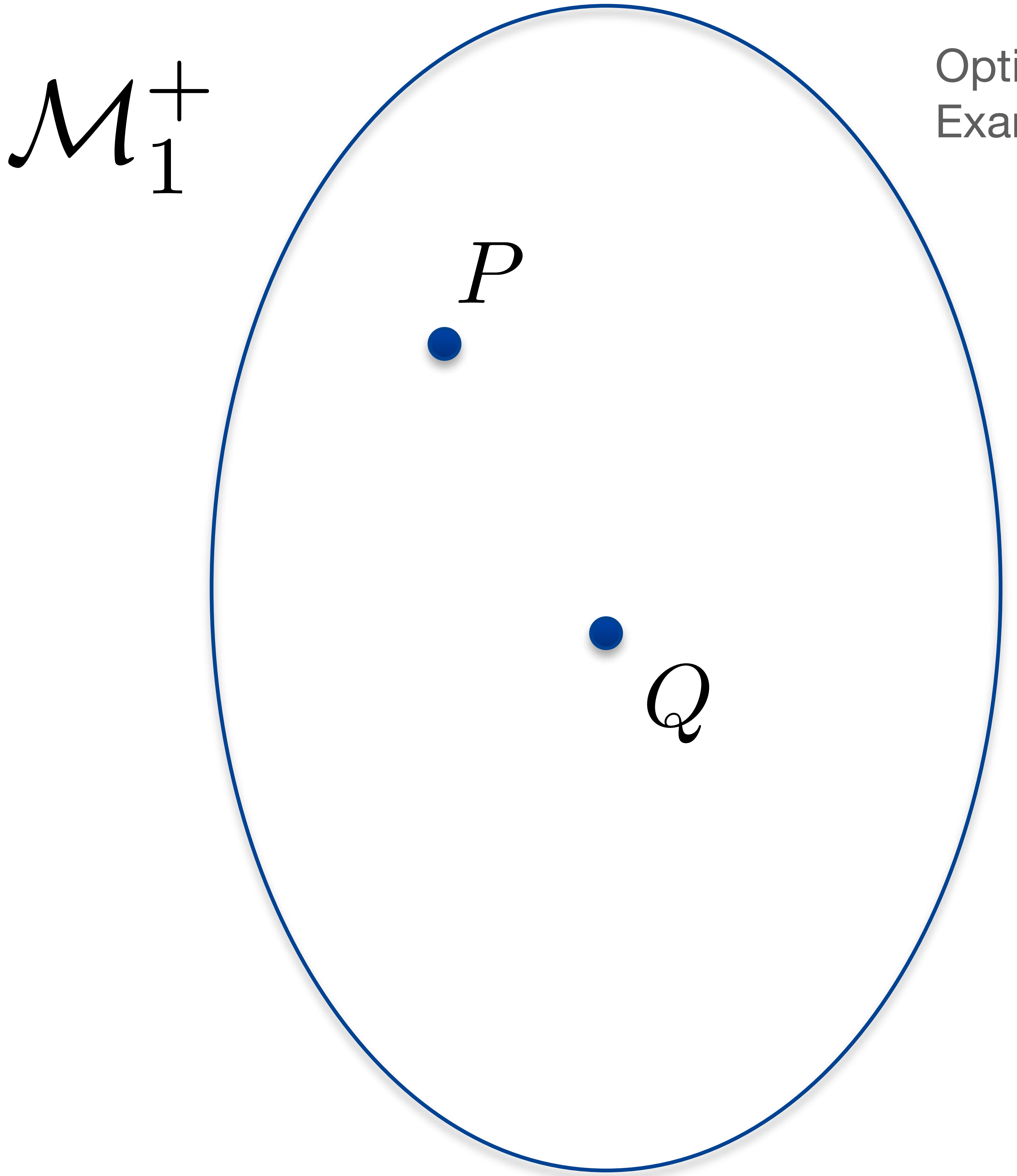
  - No assumption on the distributions

    1. This means we can target other continuous distributions than the uniform

    2. We can also target an empirical distribution (subsampling)

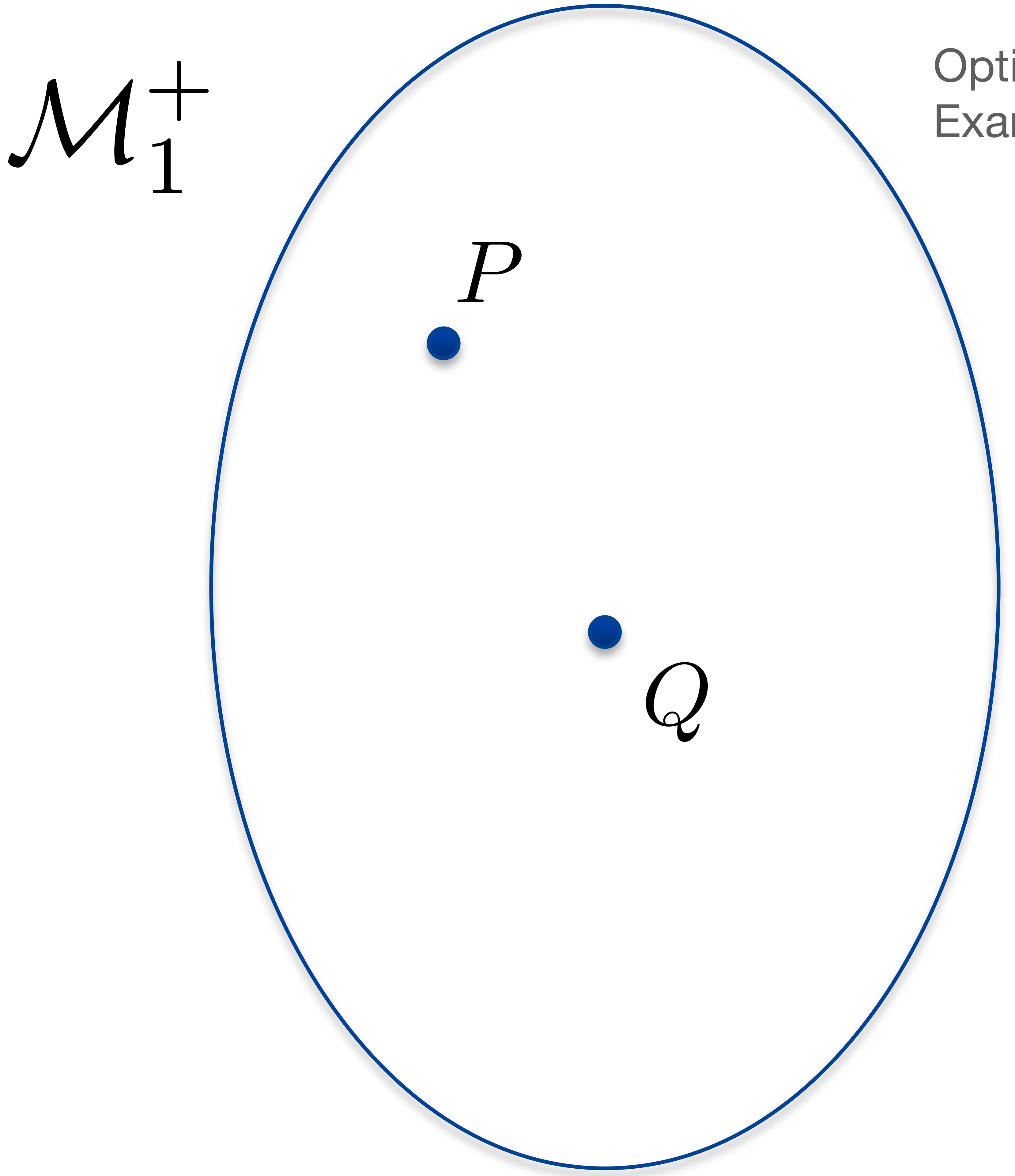
QUANTIZATION WITH THE MAXIMUM MEAN DISCREPANCY

Kernel-embedding of probability distributions



Option 1: work directly in the space of probability measures
Examples: KS, TV, KL, Hellinger, …

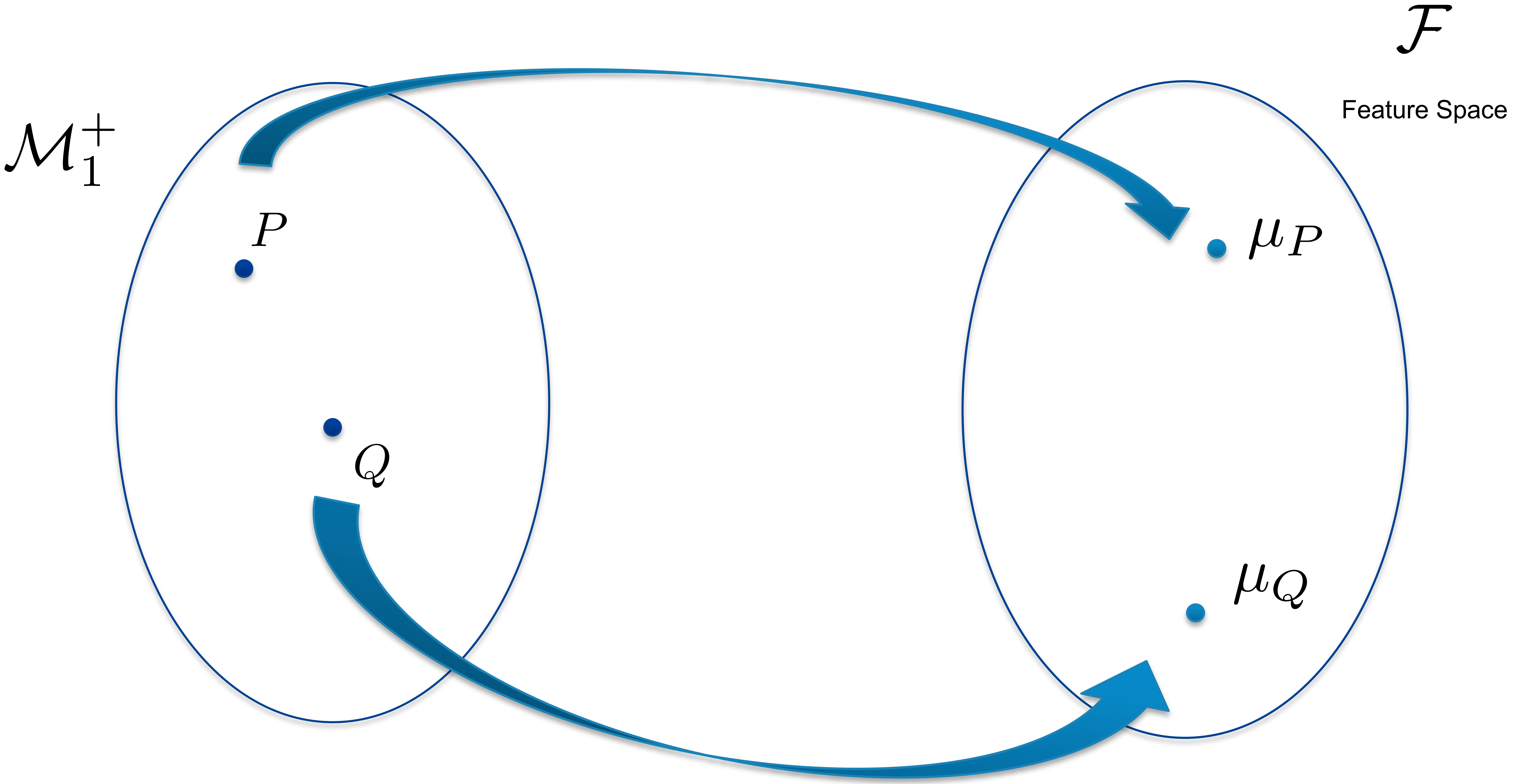
Kernel-embedding of probability distributions



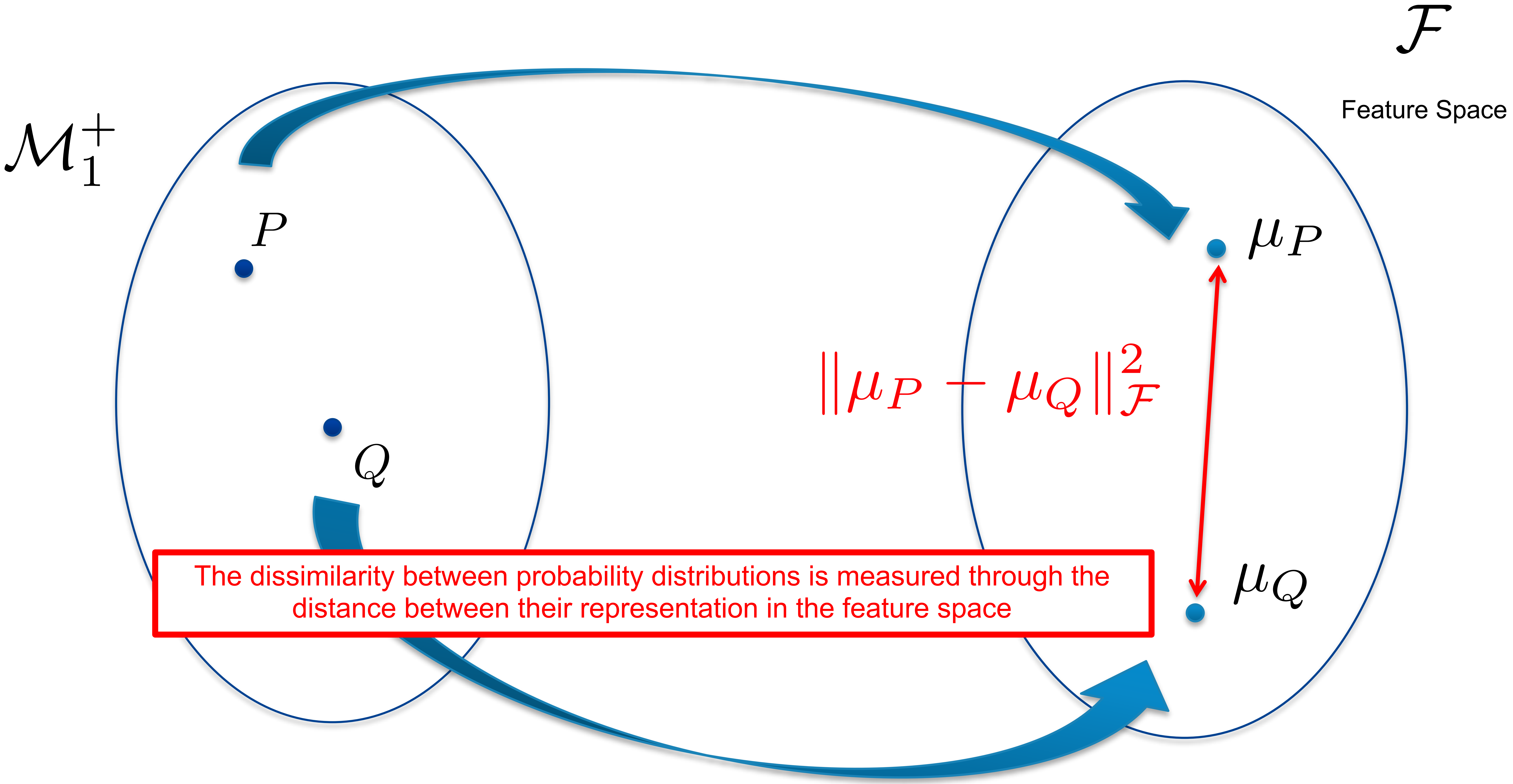
Option 1: work directly in the space of probability measures
Examples: KS, TV, KL, Hellinger, …

Option 2: represent probability measures with some features

Kernel-embedding of probability distributions

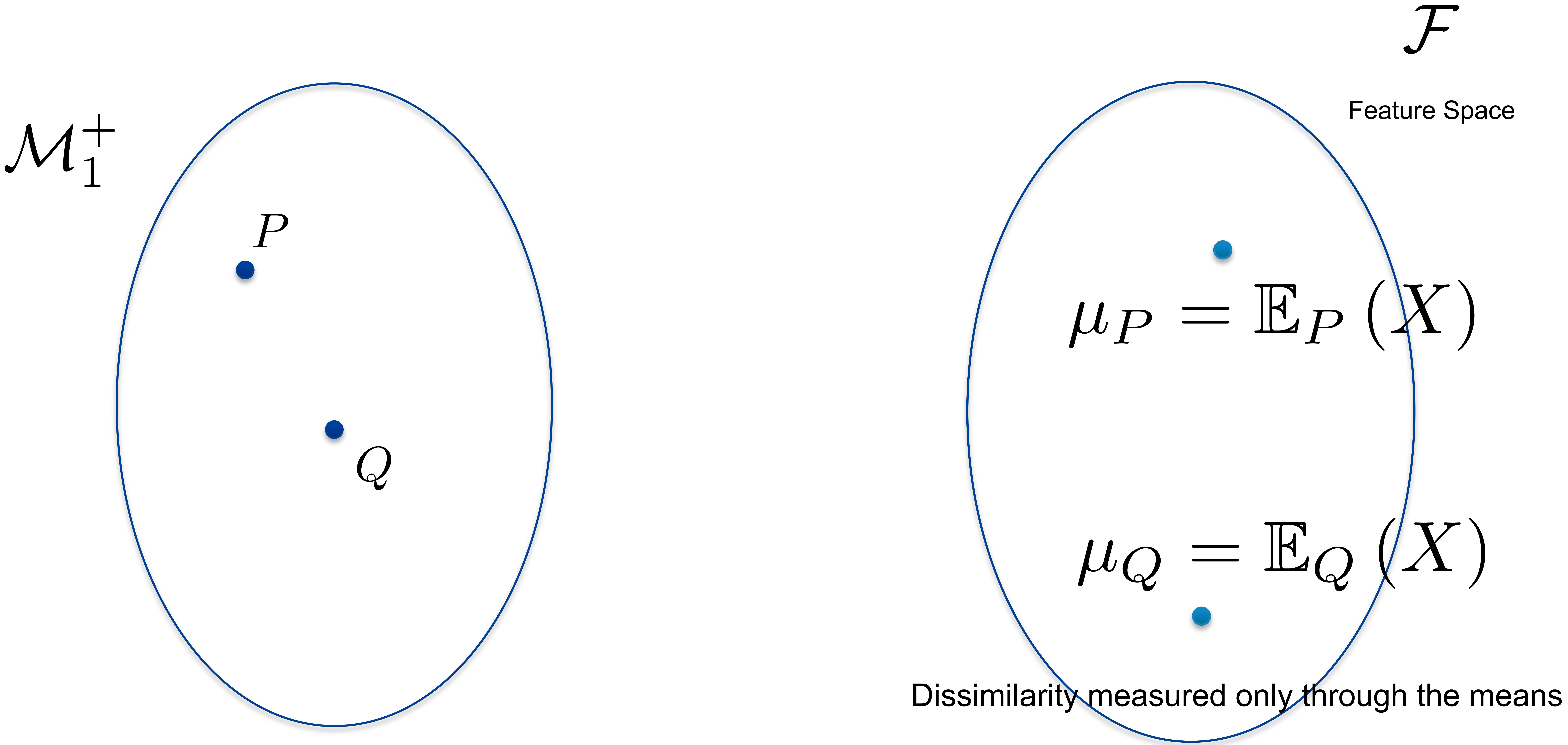


Kernel-embedding of probability distributions

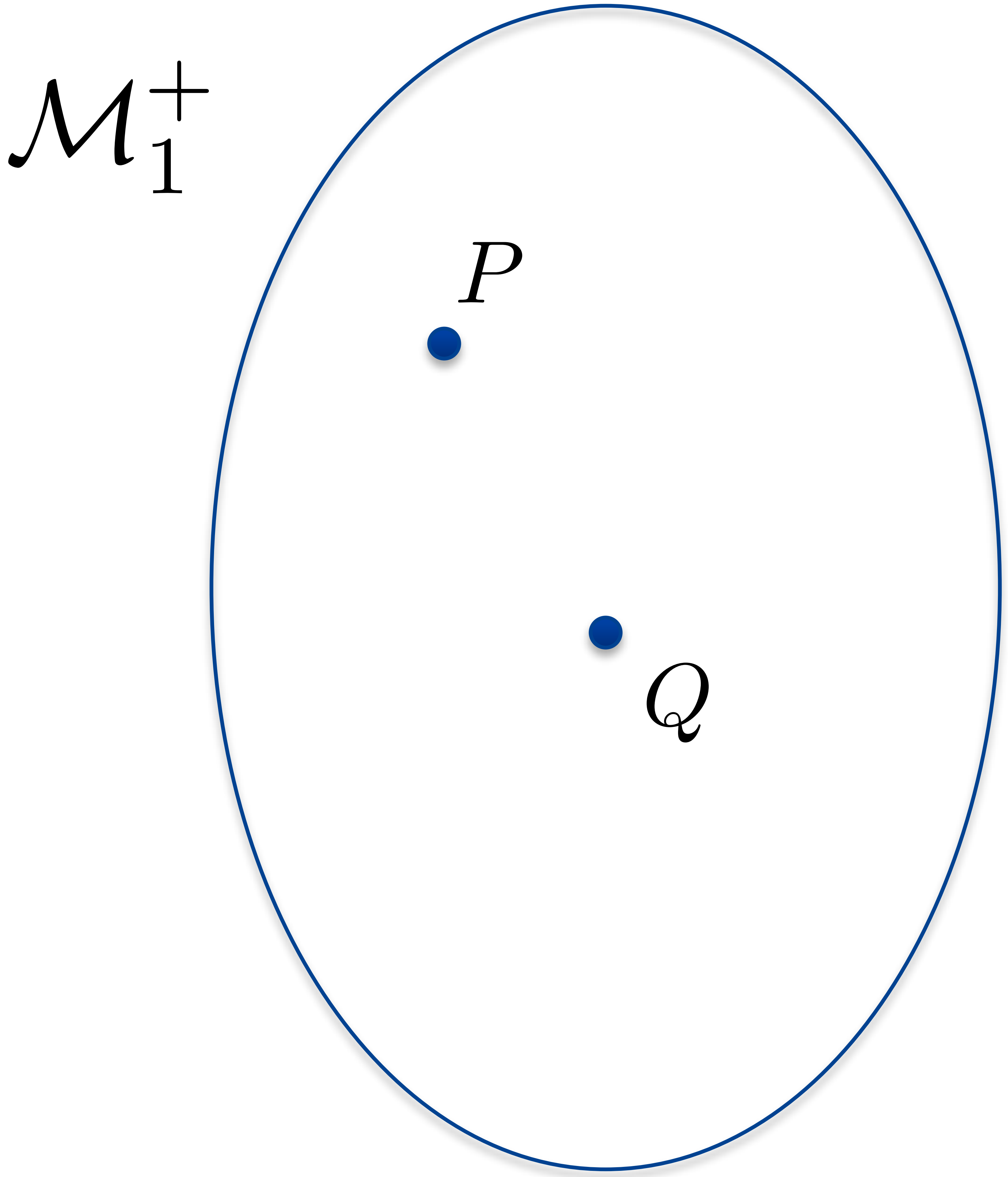


The dissimilarity between probability distributions is measured through the distance between their representation in the feature space

Kernel-embedding of probability distributions



Kernel-embedding of probability distributions



$\mathcal{F}$

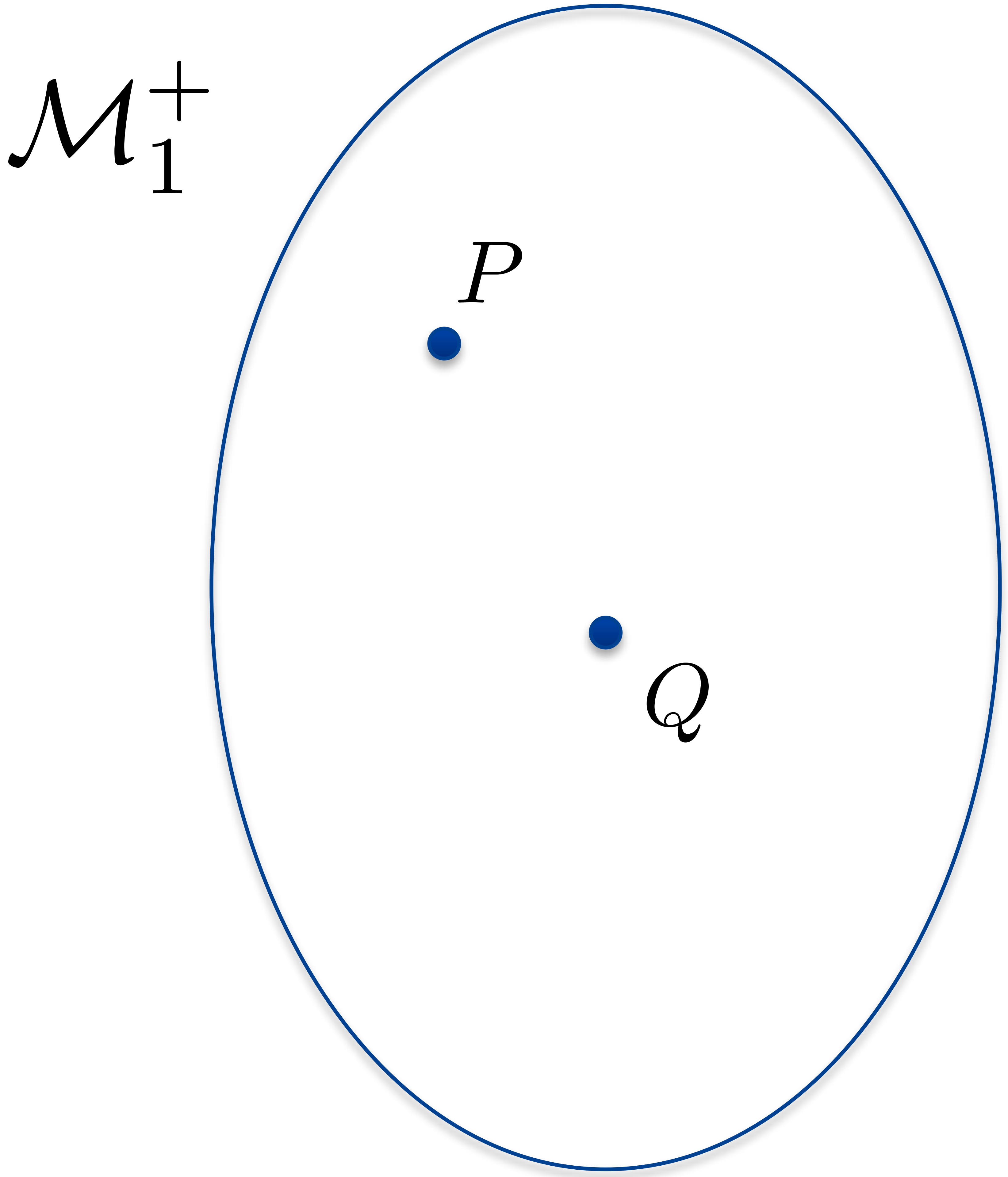
Feature Space

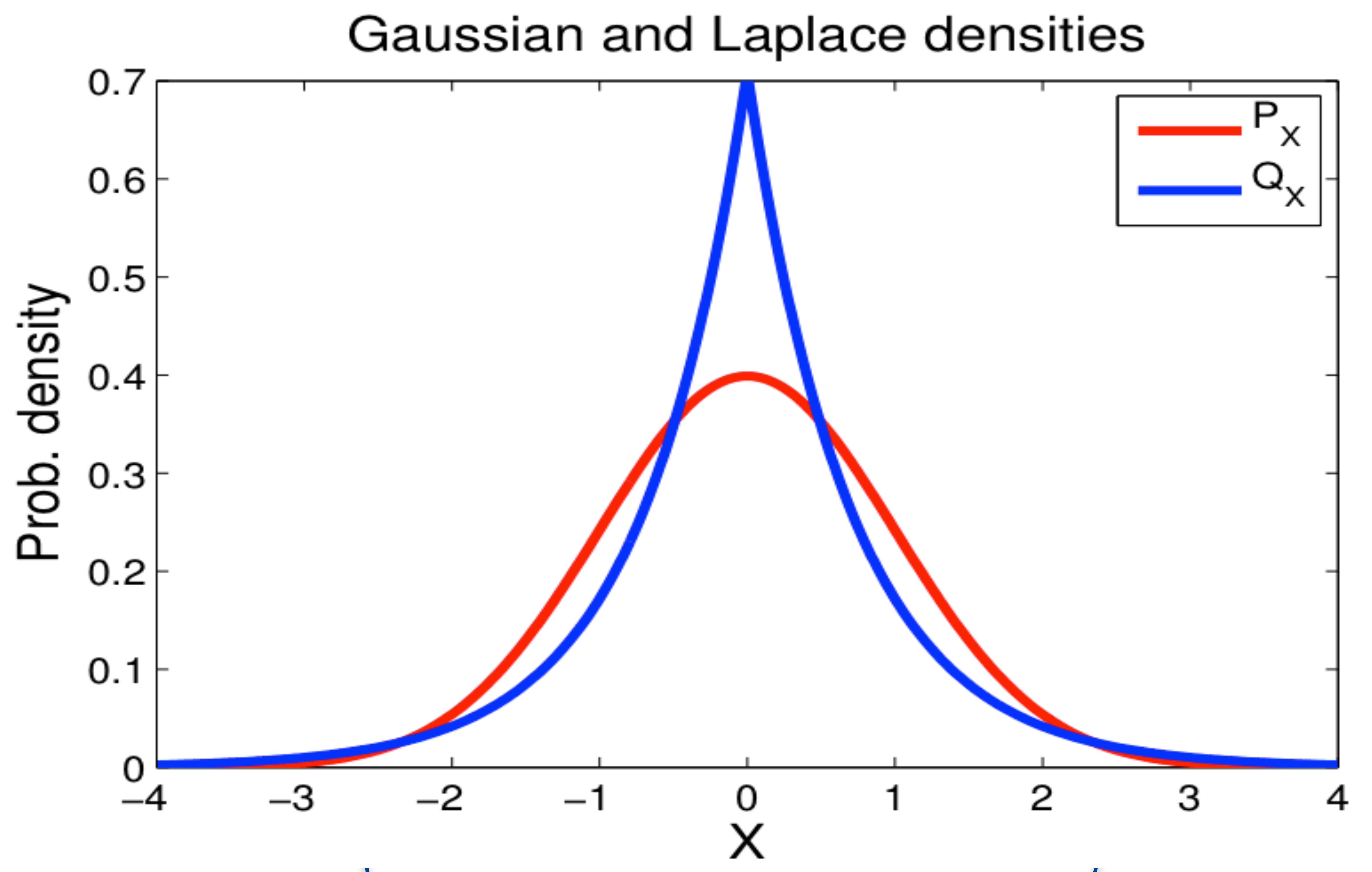
$$\mu_P = \left[\mathbb{E}_P\left(X\right),\ \mathbb{E}_P\left(X^2\right)\right]$$

$$\mu_Q = \left[\mathbb{E}_Q\left(X\right),\ \mathbb{E}_Q\left(X^2\right)\right]$$

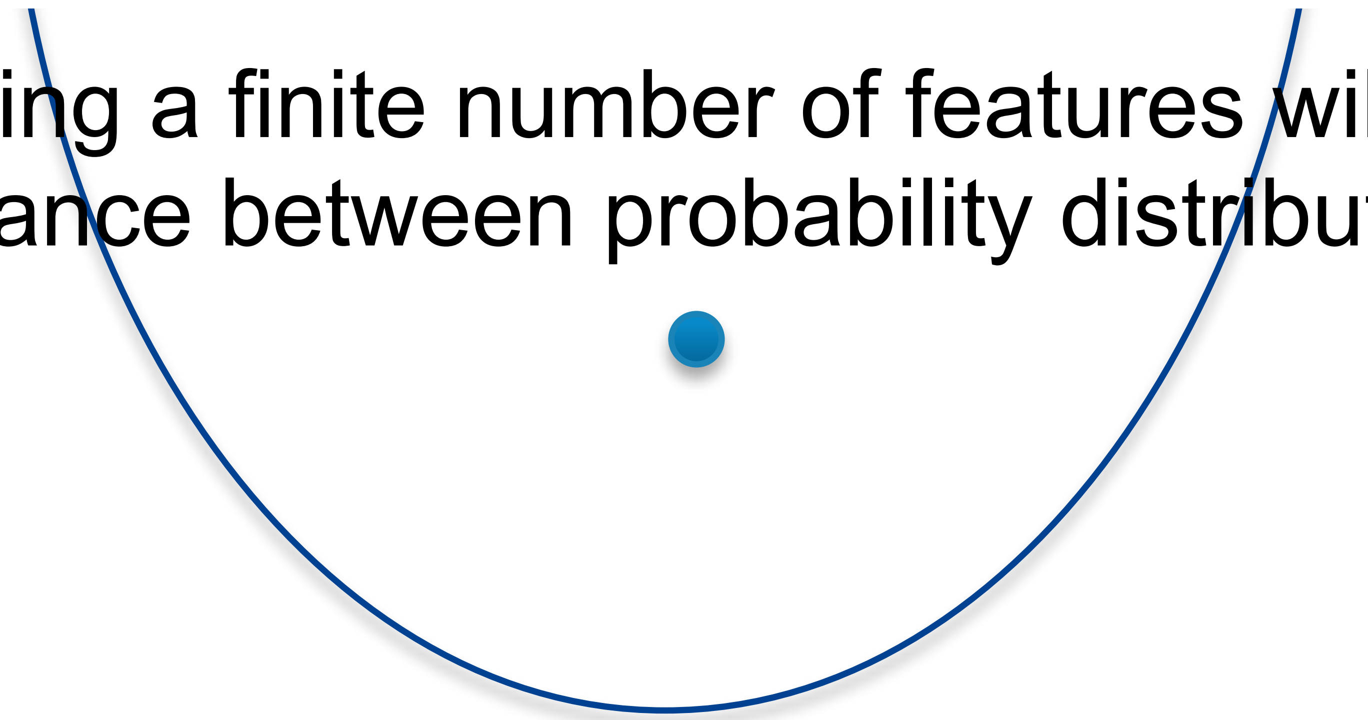
Dissimilarity measured only through first two moments

Kernel-embedding of probability distributions

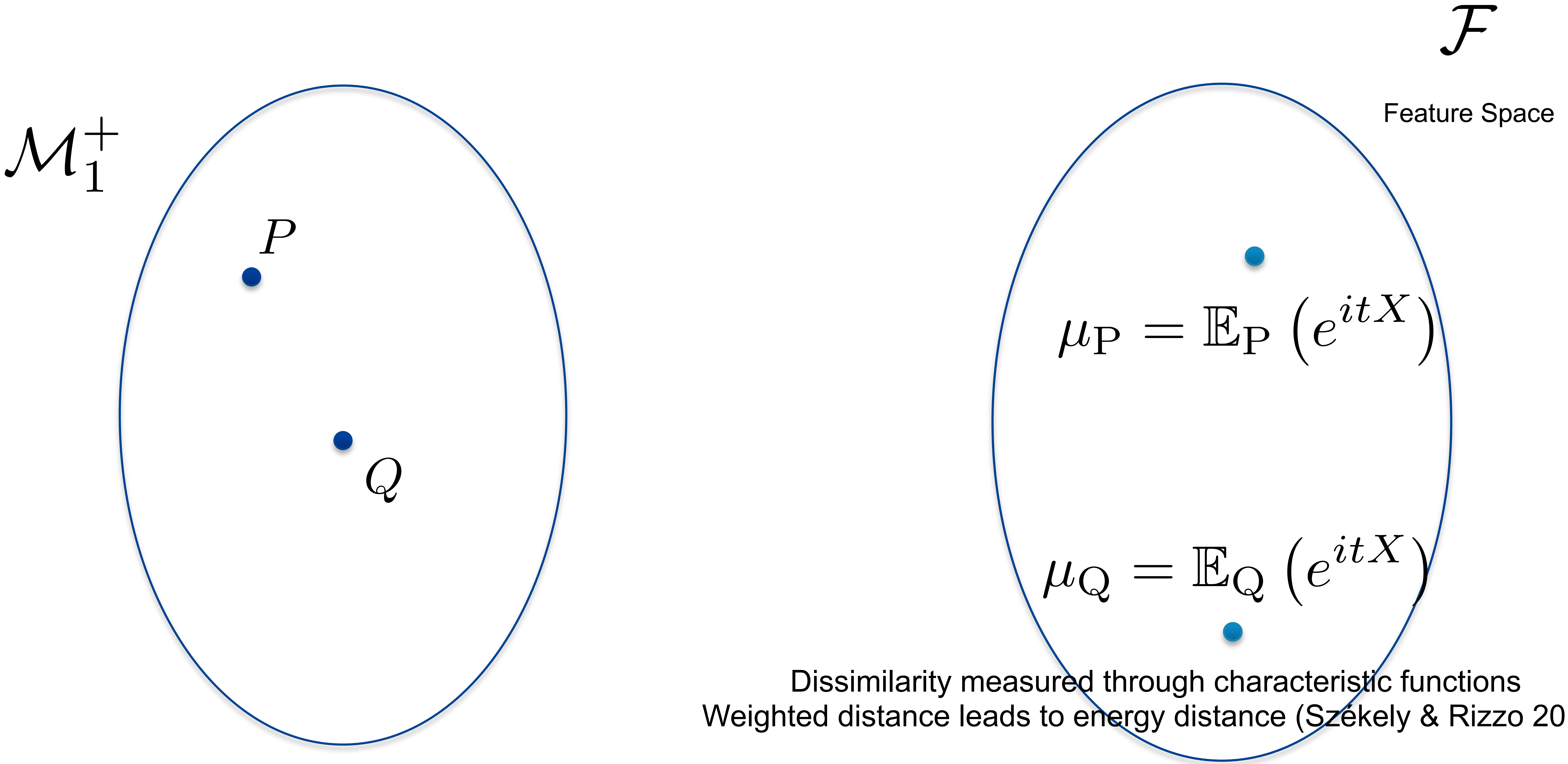




Obviously using a finite number of features will not lead to a distance between probability distributions

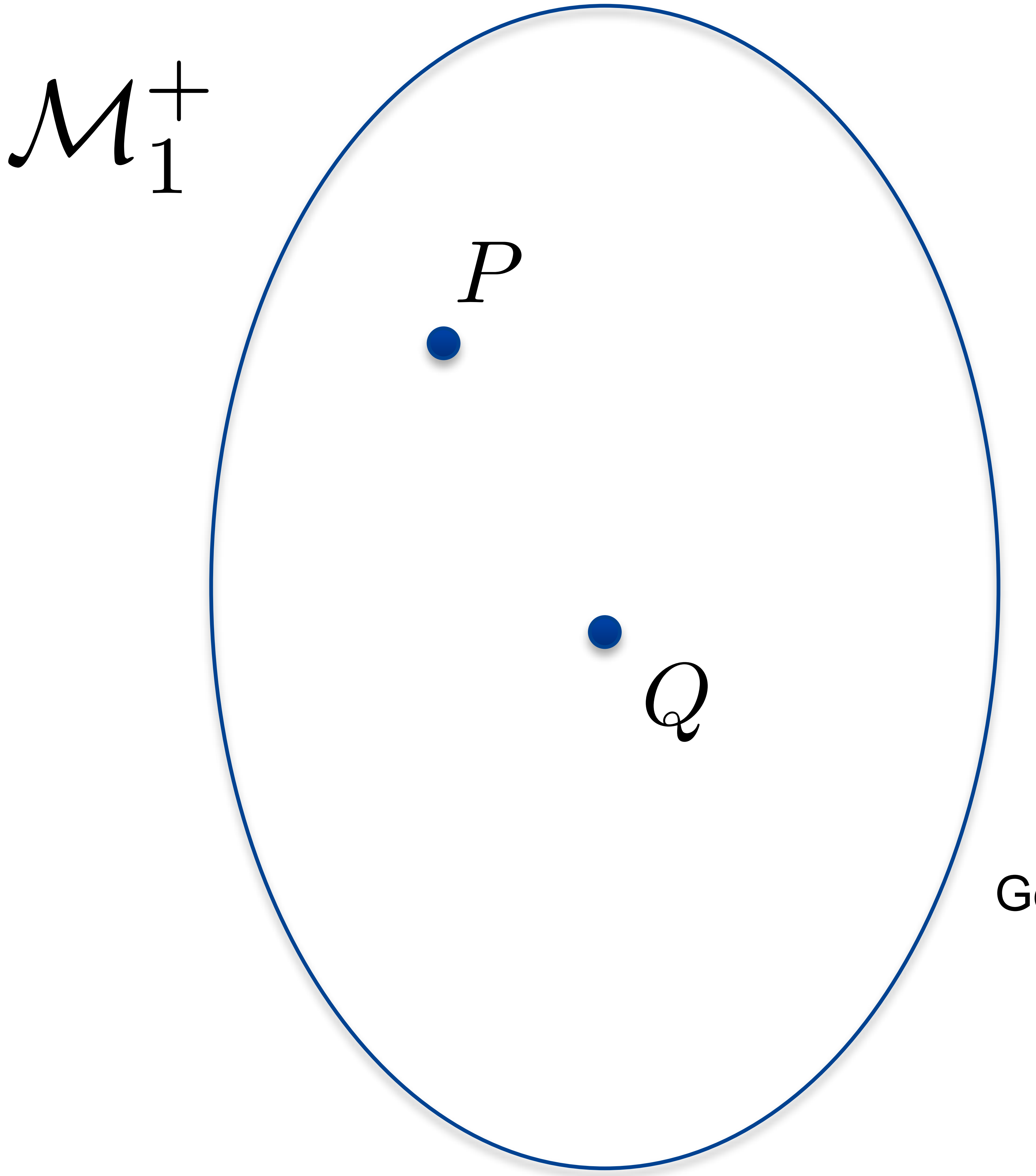


Kernel-embedding of probability distributions



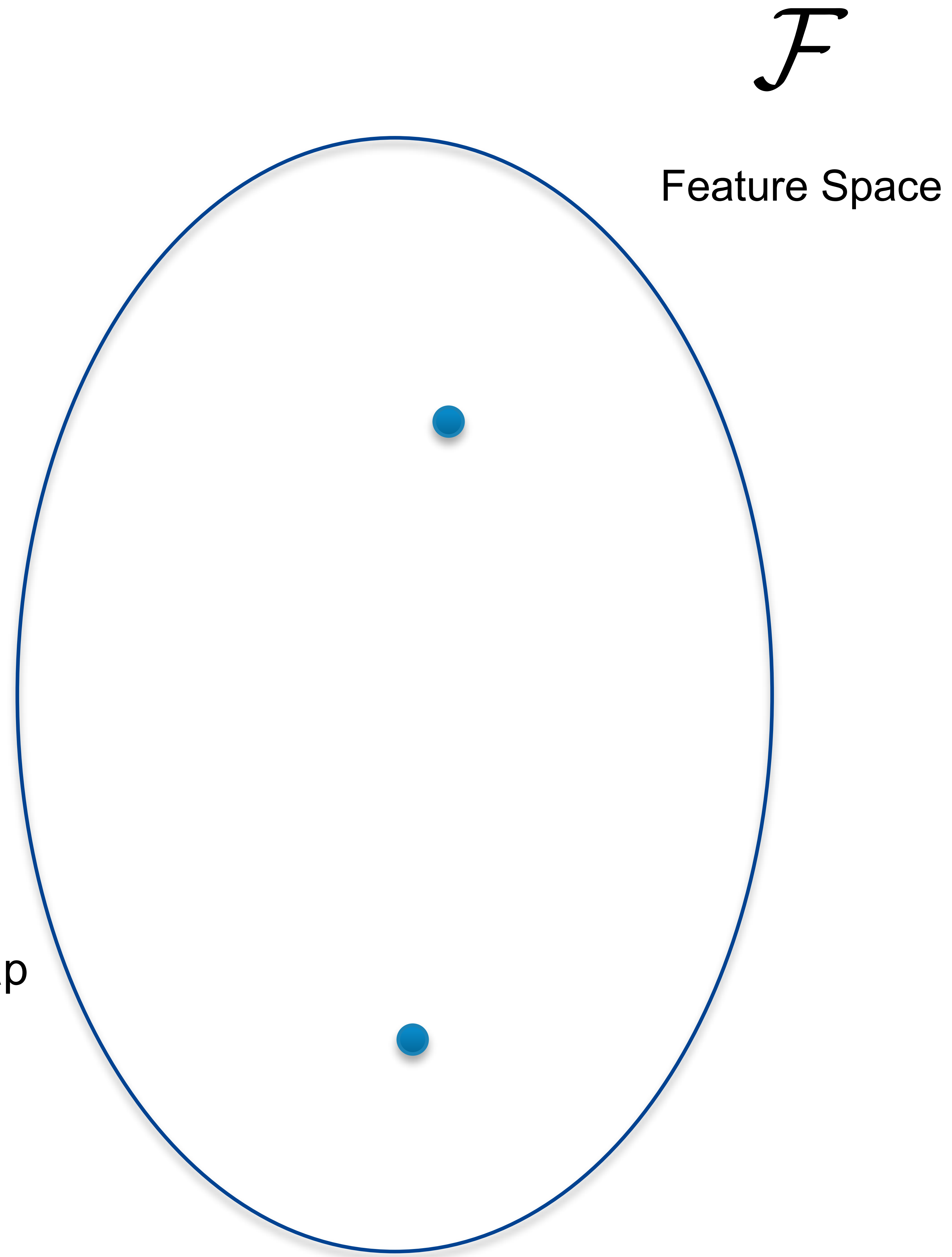
Dissimilarity measured through characteristic functions
Weighted distance leads to energy distance (Székely & Rizzo 2013)

Kernel-embedding of probability distributions

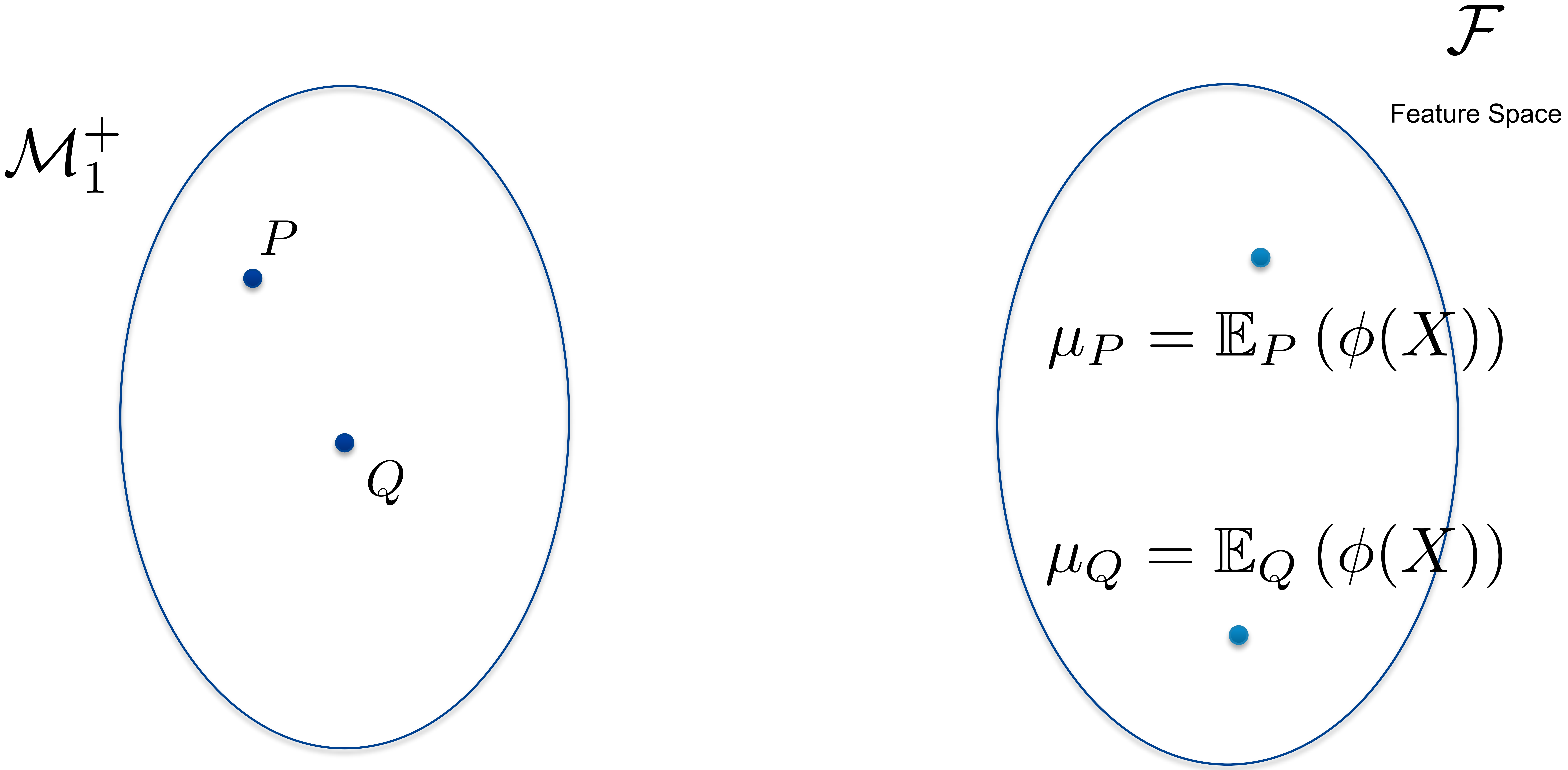


General setting: take a feature map

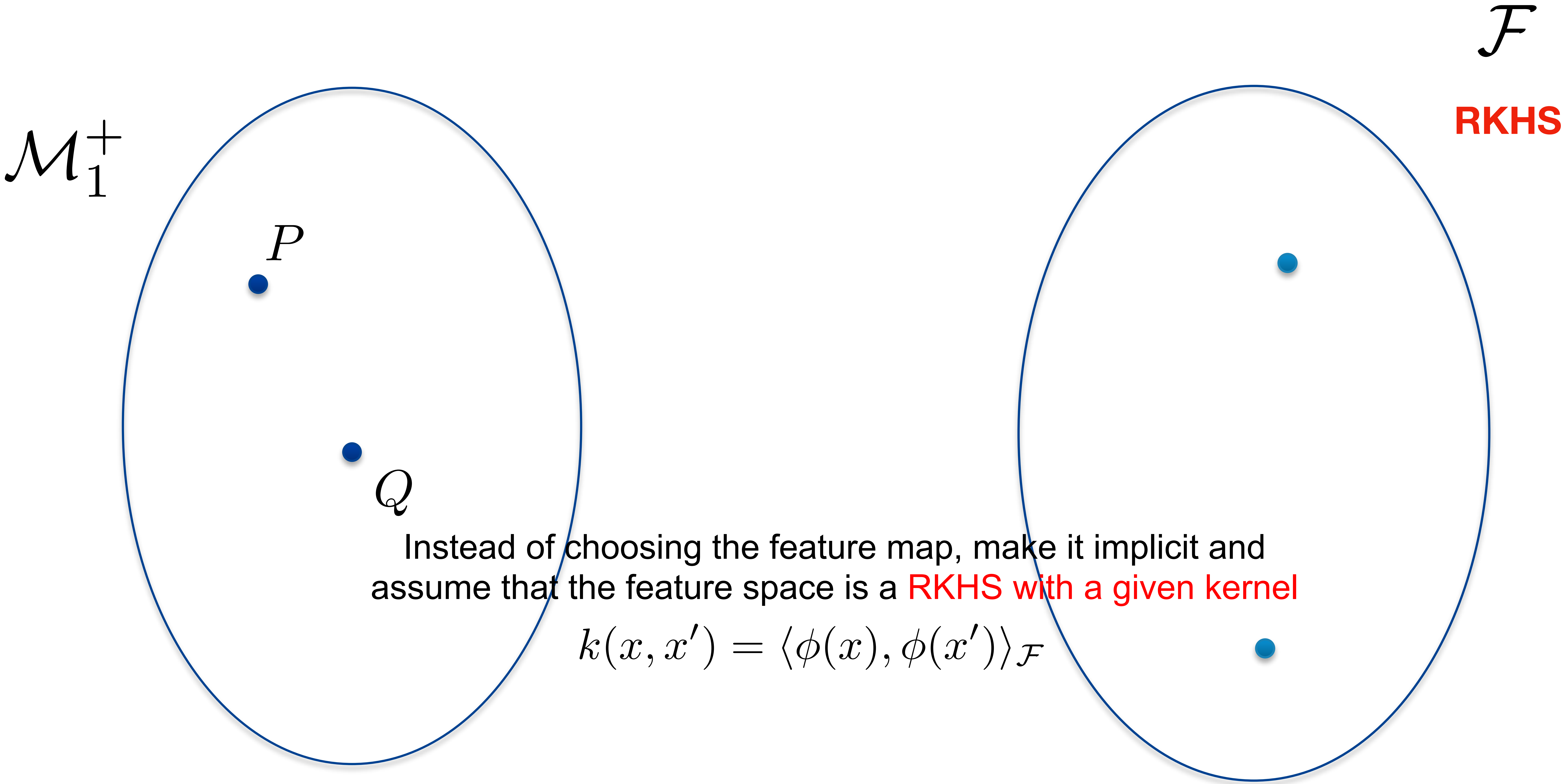
$$\phi : \ \Omega \to \mathcal{F}$$



Kernel-embedding of probability distributions



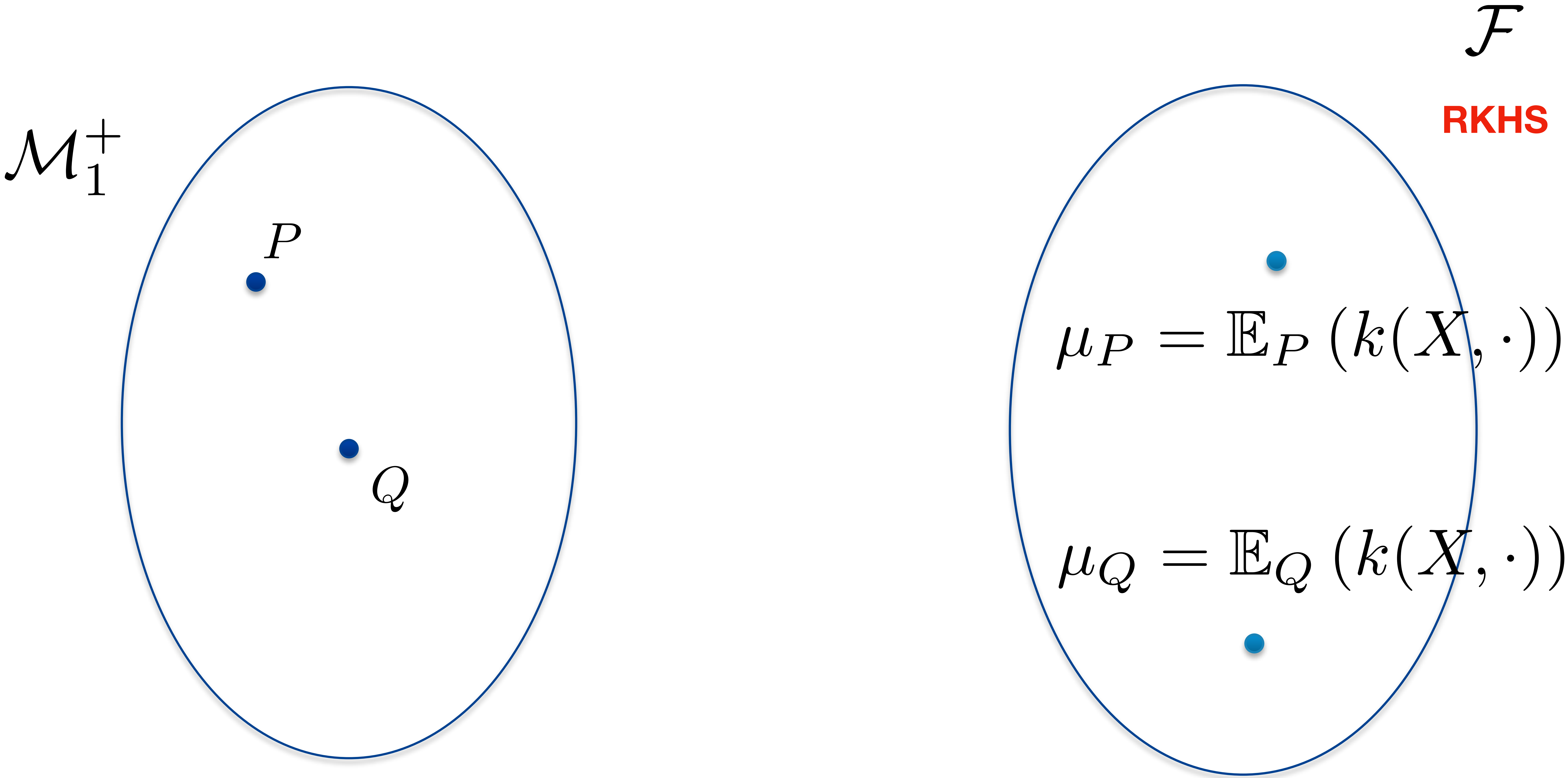
Kernel-embedding of probability distributions



Instead of choosing the feature map, make it implicit and assume that the feature space is a RKHS with a given kernel

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

Kernel-embedding of probability distributions



Kernel-embedding of probability distributions

The kernel mean embedding of a probability measure is defined as

$$\mu_{\mathrm{P}} = \mathbb{E}_{\xi \sim \mathrm{P}} k_{\mathcal{X}}(\xi, \cdot) = \int_{\mathcal{X}} k_{\mathcal{X}}(\xi, \cdot) d\mathrm{P}(\xi)$$

A distance between probability measures is then given by the **Maximum Mean Discrepancy**

$$\mathrm{MMD}(\mathrm{P}_1, \mathrm{P}_2) = \|\mu_{\mathrm{P}_1} - \mu_{\mathrm{P}_2}\|_{\mathcal{H}}$$

The reproducing property in the RKHS gives the central result

$$\mathrm{MMD}^2(\mathrm{P}_1, \mathrm{P}_2) = \mathbb{E}_{\xi,\xi'} k_{\mathcal{X}}(\xi, \xi') - 2\mathbb{E}_{\xi,\zeta} k_{\mathcal{X}}(\xi, \zeta) + \mathbb{E}_{\zeta,\zeta'} k_{\mathcal{X}}(\zeta, \zeta')$$

Smola et al. 2007, Song 2008, Song et al. 2009

Kernel-embedding of probability distributions

Advantages of this distance vs others

- Thanks to the RKHS, only involves **expectations of kernels**
- Less prone to the curse of dimensionality
- **Can easily handle structured objects** (curves, images, graphs, probability measures, sets) by using specific kernels
- (This is a distance only if a *characteristic kernel* is used)

Quantization with the MMD

$$\mathrm{MMD}^2(\mathrm{P}_1, \mathrm{P}_2) = \mathbb{E}_{\xi,\xi'} k_{\mathcal{X}}(\xi, \xi') - 2\mathbb{E}_{\xi,\zeta} k_{\mathcal{X}}(\xi, \zeta) + \mathbb{E}_{\zeta,\zeta'} k_{\mathcal{X}}(\zeta, \zeta')$$

For space-filling designs, we can then just plug the MMD instead of the discrepancy

- Standard case
  - P_1 is the empirical measure supported by the design points
  - P_2 is the uniform distribution on the hypercube

$$\underset{z_1,\ldots,z_n \in \mathbb{R}^d}{\arg\min} \ \mathrm{MMD}^2\left(\frac{1}{n}\sum_{i=1}^{n} \delta_{z_i}, \mu_{\mathcal{U}}\right)$$

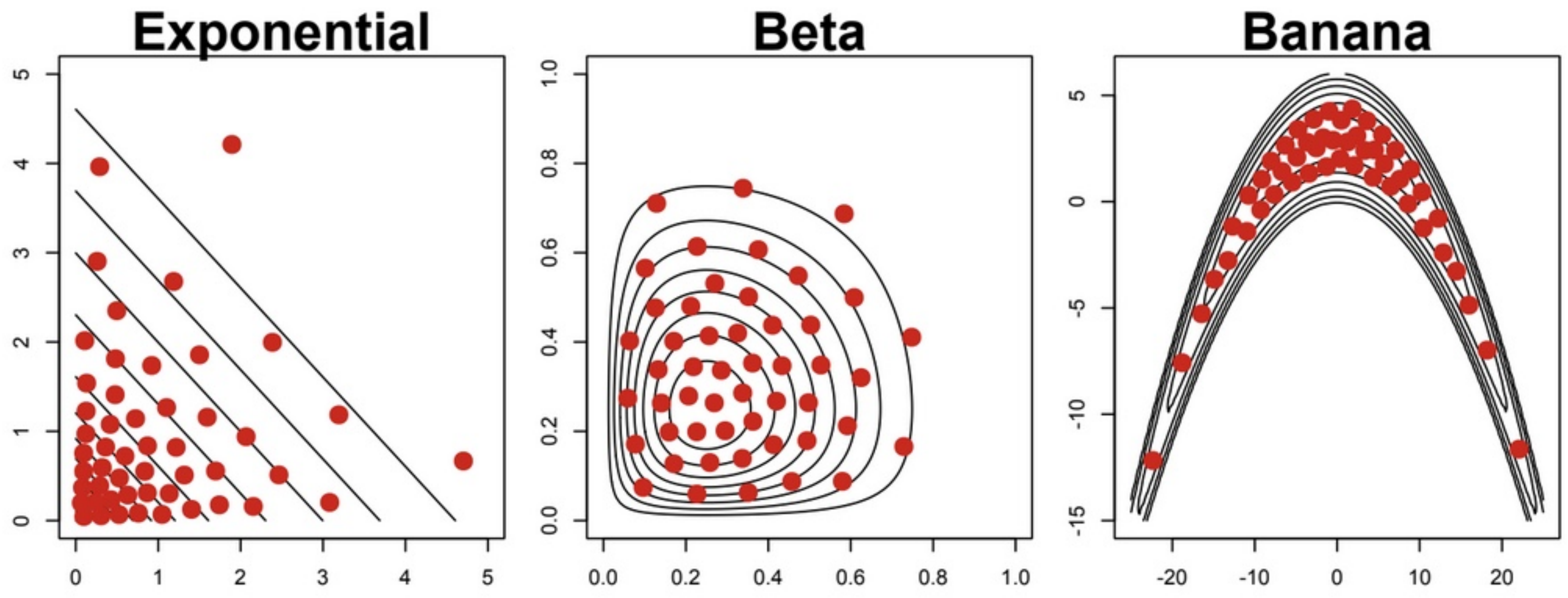
Specific kernels yield usual discrepancies!

Quantization with the MMD

$$\mathrm{MMD}^2(\mathrm{P}_1,\mathrm{P}_2) = \mathbb{E}_{\xi,\xi'} k_{\mathcal{X}}(\xi,\xi') - 2\mathbb{E}_{\xi,\zeta} k_{\mathcal{X}}(\xi,\zeta) + \mathbb{E}_{\zeta,\zeta'} k_{\mathcal{X}}(\zeta,\zeta')$$

For space-filling designs, we can then just plug the MMD instead of the discrepancy

- Standard case
- General continuous case: similar, just need to compute analytically kernel integrals



Mak & Joseph 2018
Kernel = energy distance

Quantization with the MMD

$$\mathrm{MMD}^2(\mathrm{P}_1, \mathrm{P}_2) = \mathbb{E}_{\xi,\xi'} k_{\mathcal{X}}(\xi, \xi') - 2\mathbb{E}_{\xi,\zeta} k_{\mathcal{X}}(\xi, \zeta) + \mathbb{E}_{\zeta,\zeta'} k_{\mathcal{X}}(\zeta, \zeta')$$

For space-filling designs, we can then just plug the MMD instead of the discrepancy

- Standard case
- General continuous case: similar, just need to compute analytically kernel integrals
- Subsampling case (quite common in practice): much harder!

$$\operatorname*{arg\,min}_{z_1,\ldots,z_n \in \mathbb{X}_N} \mathrm{MMD}^2\left(\frac{1}{n}\sum_{i=1}^{n} \delta_{z_i}, \frac{1}{N}\sum_{i=1}^{N} \delta_{x_i}\right)$$

NP-hard!

Quantization with the MMD

The subsampling problems writes

$$\underset{z_1,\ldots,z_n \in \mathbb{X}_N}{\arg\min} \; \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} k(z_i, z_j) - \frac{2}{n} \sum_{i=1}^{n} \sum_{j=1}^{N} k(z_i, x_j)$$

Most used strategy in practice: greedy algorithms

$$z_1^* = \underset{z \in \mathbb{X}_N}{\arg\max} \; \frac{1}{N} \sum_{j=1}^{N} k(z, x_j)$$

Maximize the similarity with the empirical target

$$z_{t+1}^* = \underset{z \in \mathbb{X}_N}{\arg\min} \; \frac{1}{t+1} \sum_{i=1}^{t} k(z, z_t^*) - \frac{1}{N} \sum_{j=1}^{N} k(z, x_j)$$

Minimize the similarity with previous points (« repulsion ») and maximize the similarity with the empirical target

Quantization with the MMD

The subsampling problems writes

$$\arg\min_{z_1,\ldots,z_n \in \mathbb{X}_N} \frac{1}{n^2} \sum_{i=1}^{n} \sum_{j=1}^{n} k(z_i, z_j) - \frac{2}{n} \sum_{i=1}^{n} \sum_{j=1}^{N} k(z_i, x_j)$$

Most used strategy in practice: greedy algorithms

See very nice recent papers: Pronzato 2021, Teymur et al. 2021

Quantization with the MMD

The subsampling problems writes

$$\underset{z_1,\ldots,z_n\in\mathbb{X}_N}{\arg\min}\ \frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}k(z_i,z_j)-\frac{2}{n}\sum_{i=1}^{n}\sum_{j=1}^{N}k(z_i,x_j)$$

Most used strategy in practice: greedy algorithms

See very nice recent papers: Pronzato 2021, Teymur et al. 2021

Here we propose a reformulation which is convex and can be efficiently solved with proximal algorithms

NEW FORMULATION

CLUSTERED LASSO & SPARSE SIMPLEX

Quantization with the MMD — Introducing weights

Instead of a DOE given by a subsample, focus now on a DOE given by a weighted version of the target

$$\operatorname*{arg\,min}_{w_1,\dots,w_N \in \Delta^{N-1}} \mathrm{MMD}^2\left(\sum_{i=1}^{N} w_i \delta_{x_i}, \frac{1}{N}\sum_{i=1}^{N} \delta_{x_i}\right)$$

$\Delta^{N-1} = \left\{\mathbf{w} \in \mathbb{R}^N : \ \mathbf{w} \geq 0, \ 1^T\mathbf{w} = 1\right\}$
is the $N-1$ dimensional canonical simplex

Quantization with the MMD — Introducing weights

Instead of a DOE given by a subsample, focus now on a DOE given by a weighted version of the target

$$\underset{w_1,\dots,w_N \in \Delta^{N-1}}{\arg\min} \quad \mathrm{MMD}^2\left(\sum_{i=1}^{N} w_i \delta_{x_i}, \frac{1}{N}\sum_{i=1}^{N} \delta_{x_i}\right)$$

$\Delta^{N-1} = \left\{\mathbf{w} \in \mathbb{R}^N : \ \mathbf{w} \geq 0,\ 1^T\mathbf{w} = 1\right\}$
is the $N-1$ dimensional
canonical simplex

$$\underset{w_1,\dots,w_N \in \Delta^{N-1}}{\arg\min} \quad \sum_{i=1}^{N}\sum_{j=1}^{N} w_i w_j k(x_i, x_j) - \frac{2}{n}\sum_{i=1}^{N}\sum_{j=1}^{N} w_i k(x_i, x_j)$$

$$\underset{\mathbf{w} \in \Delta^{N-1}}{\arg\min} \quad \mathbf{w}^T K \mathbf{w} - \mathbf{k}^T \mathbf{w}$$

- Of course here the solution is trivial by taking $\forall i,\ w_i = 1/N$

Quantization with the MMD — Introducing weights

$$\arg\min_{\mathbf{w}\in\Delta^{N-1}} \mathbf{w}^T K \mathbf{w} - \mathbf{k}^T \mathbf{w}$$

- The link with subsampling involves two additional ingredients

  1. The weight vector must be sparse

  2. All nonzero weights must be equal

Quantization with the MMD — Introducing weights

$$\underset{\mathbf{w}\in\Delta^{N-1}}{\arg\min}\ \mathbf{w}^T K \mathbf{w} - \mathbf{k}^T \mathbf{w}$$

- The link with subsampling involves two additional ingredients

1. The weight vector must be sparse

Sparsity in the simplex

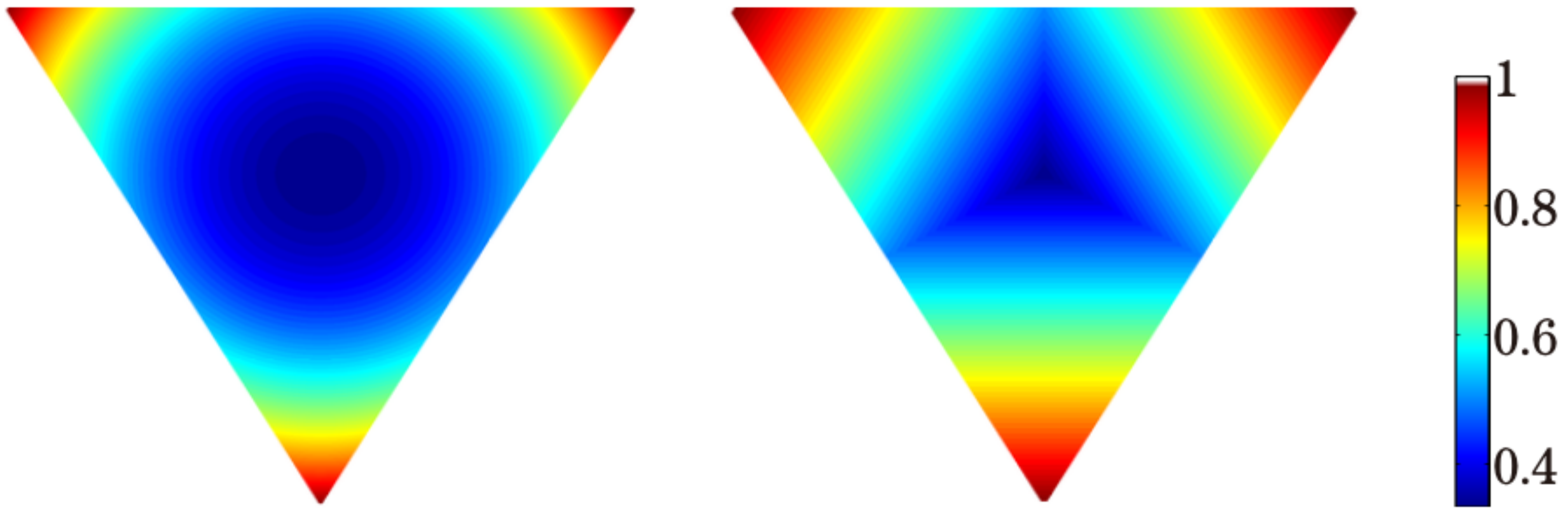
2. All nonzero weights must be equal

Clustering penalty

Quantization with the MMD — Sparsity in the simplex

$$\underset{\mathbf{w}\in\Delta^{N-1}}{\arg\min}\ \mathbf{w}^T K \mathbf{w} - \mathbf{k}^T \mathbf{w}$$

- Here sparsity cannot be achieved with the standard Lasso penalty since $||\mathbf{w}||_1 = 1$ if $\mathbf{w} \in \Delta^{N-1}$
- But on the simplex other norms have interesting features!



Li et al. 2020

Figure 2. Contours of $\beta \mapsto ||\beta||_2^2$ (left) and $\beta \mapsto ||\beta||_\infty$ (right)

Quantization with the MMD — Sparsity in the simplex

$$\arg\min_{\mathbf{w}\in\Delta^{N-1}} \mathbf{w}^T K \mathbf{w} - \mathbf{k}^T \mathbf{w}$$

- Here sparsity cannot be achieved with the standard Lasso penalty since $\|\mathbf{w}\|_1 = 1$ if $\mathbf{w} \in \Delta^{N-1}$
- But on the simplex other norms have interesting features!
- Several papers thus use either $1/\|\mathbf{w}\|_\infty,\ 1/\|\mathbf{w}\|_2^2,\ -\|\mathbf{w}\|_2^2$

Quantization with the MMD — Sparsity in the simplex

$$\arg\min_{\mathbf{w}\in\Delta^{N-1}} \mathbf{w}^T K \mathbf{w} - \mathbf{k}^T \mathbf{w}$$

- Here sparsity cannot be achieved with the standard Lasso penalty since $\|\mathbf{w}\|_1 = 1$ if $\mathbf{w} \in \Delta^{N-1}$
- But on the simplex other norms have interesting features!
- Several papers thus use either $1/\|\mathbf{w}\|_\infty,\ 1/\|\mathbf{w}\|_2^2,\ -\|\mathbf{w}\|_2^2$
- For computational considerations, we follow Li et al. 2020 and use the latter

$$\arg\min_{\mathbf{w}\in\Delta^{N-1}} \quad \mathbf{w}^T K \mathbf{w} - \mathbf{k}^T \mathbf{w} - \lambda_1 \|\mathbf{w}\|_2^2$$

$$= \mathbf{w}^T (K - \lambda_1 I)\mathbf{w} - \mathbf{k}^T \mathbf{w}$$

Still a convex problem if
$\lambda_1 \in [0, \lambda_{\min}(K)]$

Quantization with the MMD — Clustering penalties

- Clustering penalties aim at enforcing the solution of least-squares problem to have identical components

- When the penalty increases, the solution components exhibit a group structure, with all components equal inside a group: this *clusters* the solution vector, hence the name

- In practice, mainly two clustering penalties coexist:

  - The Clustered Lasso

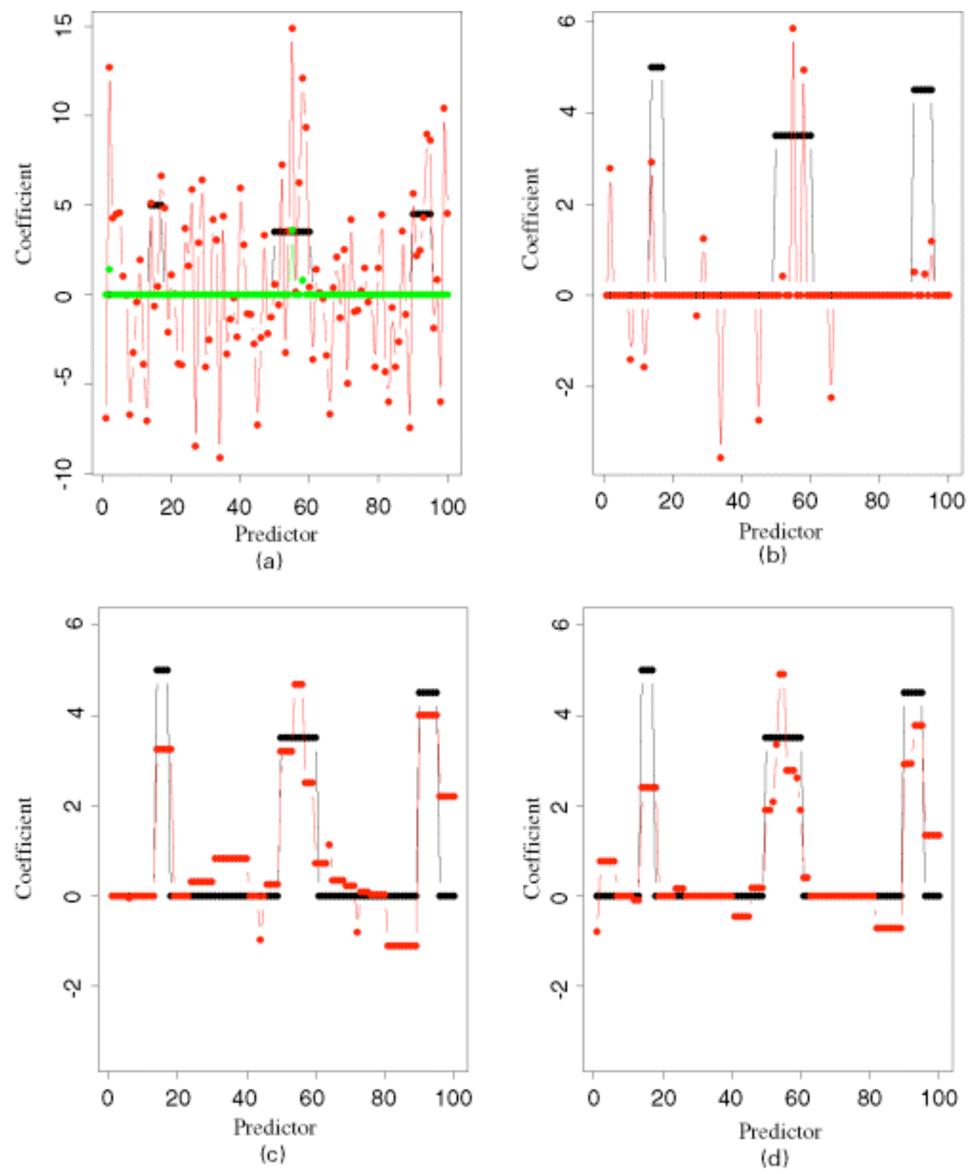
  - The OSCAR norm

Quantization with the MMD — Clustered Lasso

- The Clustered Lasso (She 2010) is an extension of the Fused Lasso (Tibshirani et al. 2005), and a particular case of the Generalized Lasso (Tibshirani & Taylor 2011)

$$\arg\min_{\beta} \|Y - X\beta\|_2^2 + \lambda_1 \|\beta\|_1 + \lambda_2 \sum_{i=1}^{p-1} |\beta_{i+1} - \beta_i|$$

Fused Lasso



$$\arg\min_{\beta} \|Y - X\beta\|_2^2 + \lambda \|D\beta\|_1$$

Generalized Lasso

Quantization with the MMD — Clustered Lasso

- The Clustered Lasso (She 2010) is an extension of the Fused Lasso (Tibshirani et al. 2005), and a particular case of the Generalized Lasso (Tibshirani & Taylor 2011)

- It enforces regression coefficients to be all equal via the penalty

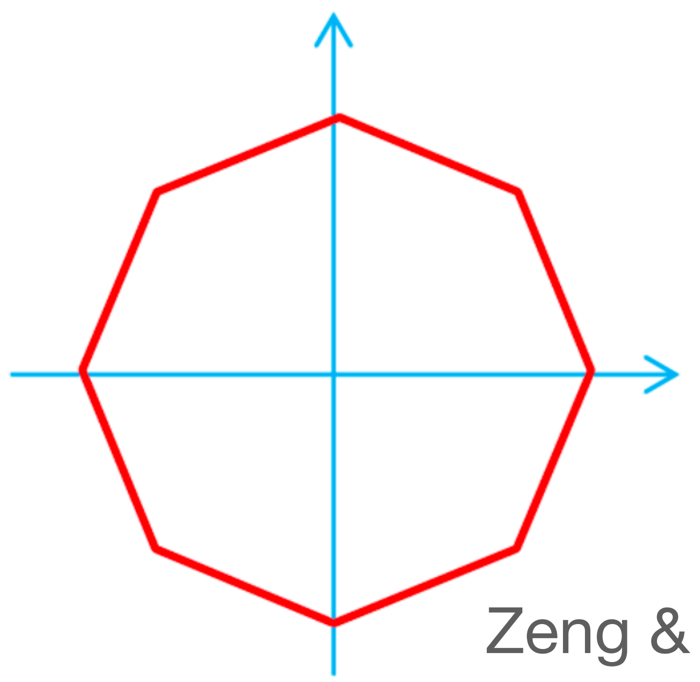
$$\arg\min_{\beta} \|Y - X\beta\|_2^2 + \lambda_1 \|\beta\|_1 + \lambda_2 \sum_{i<j}^{p} |\beta_i - \beta_j|$$

(variables can be clustered into highly correlated groups, and then a single representative covariate can be extracted from each cluster)

Quantization with the MMD — OSCAR penalty

- The OSCAR (Bondel & Reich 2007) penalty stands for Octagonal Shrinkage and Clustering Algorithm for Regression and leads to a penalized problem of the form

$$\arg\min_{\beta} \|Y - X\beta\|_2^2 + \lambda_1 \|\beta\|_1 + \lambda_2 \sum_{i<j}^{p} \max\{|\beta_i|, |\beta_j|\}$$



Zeng & Figueiredo 2015

- Interestingly, the OSCAR norm is a particular case of the Ordered Weighted L1 norm (OWL - Bogdan et al 2015, Zeng & Figueiredo 2014, Zhong & Kwok 2012), which has received a lot of attention since the introduction of the SLOPE algorithm

Quantization with the MMD — Clustered Lasso vs OSCAR?

- Which of these penalties should we use?
- Recall that, contrary to the regression setting, we search for a solution vector in the canonical simplex. In particular, it lies in the nonnegative orthant.

Quantization with the MMD — Clustered Lasso vs OSCAR?

- Which of these penalties should we use?
- Recall that, contrary to the regression setting, we search for a solution vector in the canonical simplex. In particular, it lies in the nonnegative orthant.
- Surprisingly, we have the following result which appears to be new:

Proposition 1. *Let $\mathbf{w} \in \mathbb{R}^N_{\geq 0}$ be a N-dimensional vector lying in the N dimensional nonnegative orthant. Then*

$$\Omega^{\text{oscar}}_{\rho,\lambda}(\mathbf{w}) = \Omega^{\text{classo}}_{\rho+(N-1)\lambda/2,\lambda/2}(\mathbf{w}). \qquad (5)$$

$$\Omega^{\text{oscar}}_{\rho,\lambda}(\mathbf{w}) = \rho\|\mathbf{w}\|_1 + \lambda \sum_{i<j}^{N} \max\{|w_i|, |w_j|\}$$

$$\Omega^{\text{classo}}_{\rho,\lambda}(\mathbf{w}) = \rho\|\mathbf{w}\|_1 + \lambda \sum_{i<j}^{N} |w_i - w_j|$$

Quantization with the MMD — Clustered Lasso vs OSCAR?

- This equivalence result implies that in our setting both clustering penalties are equivalent

- However, when we will rewrite our problem in an amenable form suited for efficient proximal algorithms, this equivalence will be lost

- Decomposition properties of proximal operators (to be detailed in a few moments) lead us to choose the Clustered Lasso over OSCAR

Quantization with the MMD — Putting things together

- The final formulation is obtained by mixing the sparsity penalty in the simplex and the Clustered Lasso penalty:

$$\underset{\mathbf{w}\in\Delta^{N-1}}{\arg\min}\ \mathbf{w}^T(K-\lambda_1 I)\mathbf{w}-\mathbf{k}^T\mathbf{w}+\lambda_2\sum_{i<j}^{N}|w_i-w_j|$$

- This is a convex problem in dimension N
- In practice:
  1. How can we efficiently solve it?
  2. Scaling w.r.t. N?

Quantization with the MMD — Putting things together

- We first reformulate one last time the problem as

$$\arg\min_{\mathbf{w}} \mathbf{w}^T(K - \lambda_1 I)\mathbf{w} - \mathbf{k}^T\mathbf{w} + 1\left(\mathbf{w} \in \Delta^{N-1}\right) + \lambda_2 \sum_{i<j}^{N} |w_i - w_j|$$

Quantization with the MMD — Putting things together

- We first reformulate one last time the problem as

$$\arg\min_{\mathbf{w}} \boxed{\mathbf{w}^T(K-\lambda_1 I)\mathbf{w} - \mathbf{k}^T\mathbf{w}} + \boxed{1\left(\mathbf{w} \in \Delta^{N-1}\right) + \lambda_2 \sum_{i<j}^{N} |w_i - w_j|}$$

Convex, differentiable

Convex, non-differentiable

Quantization with the MMD — Putting things together

- We first reformulate one last time the problem as

$$\arg\min_{\mathbf{w}} \boxed{\mathbf{w}^T(K - \lambda_1 I)\mathbf{w} - \mathbf{k}^T\mathbf{w}} + \boxed{1\left(\mathbf{w} \in \Delta^{N-1}\right) + \lambda_2 \sum_{i<j}^{N} |w_i - w_j|}$$

Convex, differentiable

Convex, non-differentiable

- This form guides us towards the **proximal gradient algorithm**

$$\min_x \; f(x) + g(x)$$

$$x_{k+1} = \mathrm{prox}_{tg}\left(x_k - t\nabla f(x_k)\right)$$

$$\mathrm{prox}_h(x) = \arg\min_u \left(h(u) + \frac{1}{2}\|u - x\|_2^2\right)$$

Quantization with the MMD — Putting things together

- We first reformulate one last time the problem as

$$\arg\min_{\mathbf{w}} \underbrace{\boxed{\mathbf{w}^T(K-\lambda_1 I)\mathbf{w} - \mathbf{k}^T\mathbf{w}}}_{\text{Convex, differentiable}} + \underbrace{\boxed{1\left(\mathbf{w} \in \Delta^{N-1}\right) + \lambda_2 \sum_{i<j}^{N} |w_i - w_j|}}_{\text{Convex, non-differentiable}}$$

- This form guides us towards the **proximal gradient algorithm**

$$\min_x \boxed{f(x)} + g(x)$$

$$x_{k+1} = \mathrm{prox}_{tg}\left(x_k - t\boxed{\nabla f(x_k)}\right)$$

Easy gradient!

$$\mathrm{prox}_h(x) = \arg\min_u \left(h(u) + \frac{1}{2}\|u - x\|_2^2\right)$$

Quantization with the MMD — Putting things together

- We first reformulate one last time the problem as

$$\arg\min_{\mathbf{w}} \boxed{\mathbf{w}^T(K-\lambda_1 I)\mathbf{w} - \mathbf{k}^T\mathbf{w}} + \boxed{1\left(\mathbf{w}\in\Delta^{N-1}\right) + \lambda_2 \sum_{i<j}^{N} |w_i - w_j|}$$

Convex, differentiable

Convex, non-differentiable

- This form guides us towards the **proximal gradient algorithm**

$$\min_x f(x) + \boxed{g(x)}$$

$$x_{k+1} = \boxed{\mathrm{prox}_{tg}}(x_k - t\nabla f(x_k))$$

Can we compute the proximal operator of this?

$$\mathrm{prox}_h(x) = \arg\min_u \left(h(u) + \frac{1}{2}\|u-x\|_2^2\right)$$

Quantization with the MMD — Putting things together

$$1\left(\mathbf{w} \in \Delta^{N-1}\right) + \lambda_2 \sum_{i<j}^{N} |w_i - w_j|$$

- Step1: we need a result on the proximal operator of a sum of functions

Quantization with the MMD — Putting things together

$$\mathbb{1}\left(\mathbf{w} \in \Delta^{N-1}\right) + \lambda_2 \sum_{i<j}^{N} |w_i - w_j|$$

- Step1: we need a result on the proximal operator of a sum of functions

  Fortunately, Yu 2013 Corollary 4 gives, for h permutation invariant

$$\mathrm{prox}_{h+\|\cdot\|_{\mathrm{tv}}} = \mathrm{prox}_h \circ \mathrm{prox}_{\mathrm{tv}}$$

$$\|x\|_{\mathrm{tv}} = \sum_{i,j \in E} \alpha_{i,j} |x_i - x_j|$$

Equivalent result for OSCAR norm, but requires more assumptions on h,
which are not satisfied for the indicator function above!

Quantization with the MMD — Putting things together

$$1\left(\mathbf{w} \in \Delta^{N-1}\right) + \lambda_2 \sum_{i<j}^{N} |w_i - w_j|$$

- Step1: we need a result on the proximal operator of a sum of functions

  Fortunately, Yu 2013 Corollary 4 gives, for h permutation invariant

$$\mathrm{prox}_{h+\|\cdot\|_{\mathrm{tv}}} = \mathrm{prox}_h \circ \mathrm{prox}_{\mathrm{tv}}$$

$$\|x\|_{\mathrm{tv}} = \sum_{i,j \in E} \alpha_{i,j} |x_i - x_j|$$

- Step 2: the first proximal operator is just a projection, given by

$$(\mathcal{P}_{\Delta^{p-1}}(\mathbf{w}))_i = \max(0, w_i - \tau), \quad \tau := \left(\sum_{i=1}^{\rho} w_i - 1\right)/\rho, \quad \rho = \max\{j : \; w_j > (\sum_{i=1}^{j} w_i - 1)/j\}$$

Kyrillidis et al. 2013

Quantization with the MMD — Putting things together

$$1\left(\mathbf{w} \in \Delta^{N-1}\right) + \lambda_2 \sum_{i<j}^{N} |w_i - w_j|$$

- Step 3: the second proximal operator has just recently been computed!

$$\mathrm{prox}_{\lambda_2 \sum_{i<j}^{N} |w_i - w_j|} = \Pi_{\mathbf{w}}^T \mathcal{P}_{\mathcal{D}} \left(\Pi_{\mathbf{w}} \mathbf{w} - \lambda_2 \mathbf{v}\right)$$

$$\mathcal{D} = \{\mathbf{x} \in \mathbb{R}^N : \; B\mathbf{x} \geq 0\}, \; B\mathbf{x} = [x_1 - x_2, \ldots, x_{N-1} - x_N]$$

Lin et al. 2019

$$\Pi_{\mathbf{x}} \mathbf{x} = [x_{(1)}, \ldots, x_{(N)}]$$

$$v_i = N - 2i + 1$$

Computable with the pool-adjacent violation algorithm (isotonic regression)

Quantization with the MMD — Putting things together

$$1\left(\mathbf{w} \in \Delta^{N-1}\right) + \lambda_2 \sum_{i<j}^{N} |w_i - w_j|$$

- Step 3: the second proximal operator has just recently been computed!

$$\mathrm{prox}_{\lambda_2 \sum_{i<j}^{N} |w_i - w_j|} = \Pi_{\mathbf{w}}^{T} \mathcal{P}_{\mathcal{D}}\left(\Pi_{\mathbf{w}} \mathbf{w} - \lambda_2 \mathbf{v}\right)$$

$$\mathcal{D} = \{\mathbf{x} \in \mathbb{R}^N : \ B\mathbf{x} \geq 0\}, \ B\mathbf{x} = [x_1 - x_2, \ldots, x_{N-1} - x_N]$$

Lin et al. 2019

$$\Pi_{\mathbf{x}} \mathbf{x} = [x_{(1)}, \ldots, x_{(N)}]$$

$$v_i = N - 2i + 1$$

Computable with the pool-adjacent violation algorithm (isotonic regression)

- Step 4 (optional): we can also accelerate the proximal gradient algorithm with Nesterov -> $O(1/t^2)$

Quantization with the MMD — Final algorithm

Algorithm 1 Accelerated proximal gradient algorithm for Problem (6)

Require: Gram matrix $K \in \mathbb{R}^{N \times N}$, regularization constants $\lambda_1 \in [0, \lambda_{\min}(K)], \lambda_2 > 0$, initial weights $\mathbf{w}^0 \in \mathbb{R}^N$ and sequence of step sizes t_k, $k = 0, \ldots$.

Set $\mathbf{v}^0 = \mathbf{w}^0$

for $k = 0, 1, \ldots$ **do**

$\mathbf{w}^{k+1} = \mathcal{P}_{\Delta^{N-1}}\left(\mathbf{v}^k - t_k(2(K - \lambda_1 I)\mathbf{v}^k - \mathbf{k}\right)$ ▷ Equation (8)

$\mathbf{w}^{k+1} = \mathrm{prox}_{t_k g_2}(\mathbf{w}^{k+1})$ ▷ Equation (9)

$\mathbf{v}^{k+1} = \mathbf{w}^{k+1} + \frac{k}{k+3}(\mathbf{w}^{k+1} - \mathbf{w}^k)$

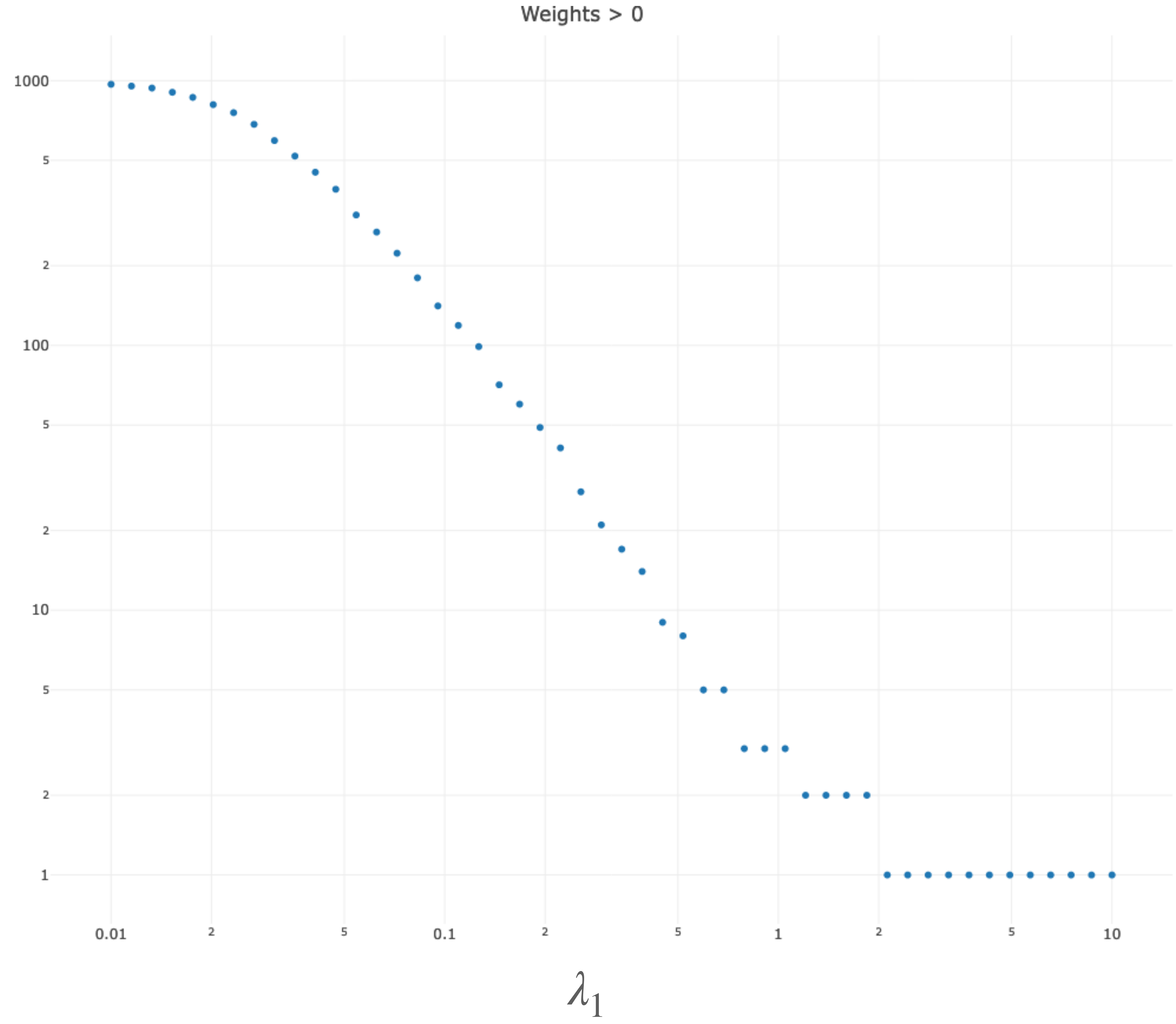
end for

Quantization with the MMD — In practice

$$\arg\min_{\mathbf{w}} \mathbf{w}^T(K - \lambda_1 I)\mathbf{w} - \mathbf{k}^T\mathbf{w} + 1\left(\mathbf{w} \in \Delta^{N-1}\right) + \lambda_2 \sum_{i<j}^{N} |w_i - w_j|$$

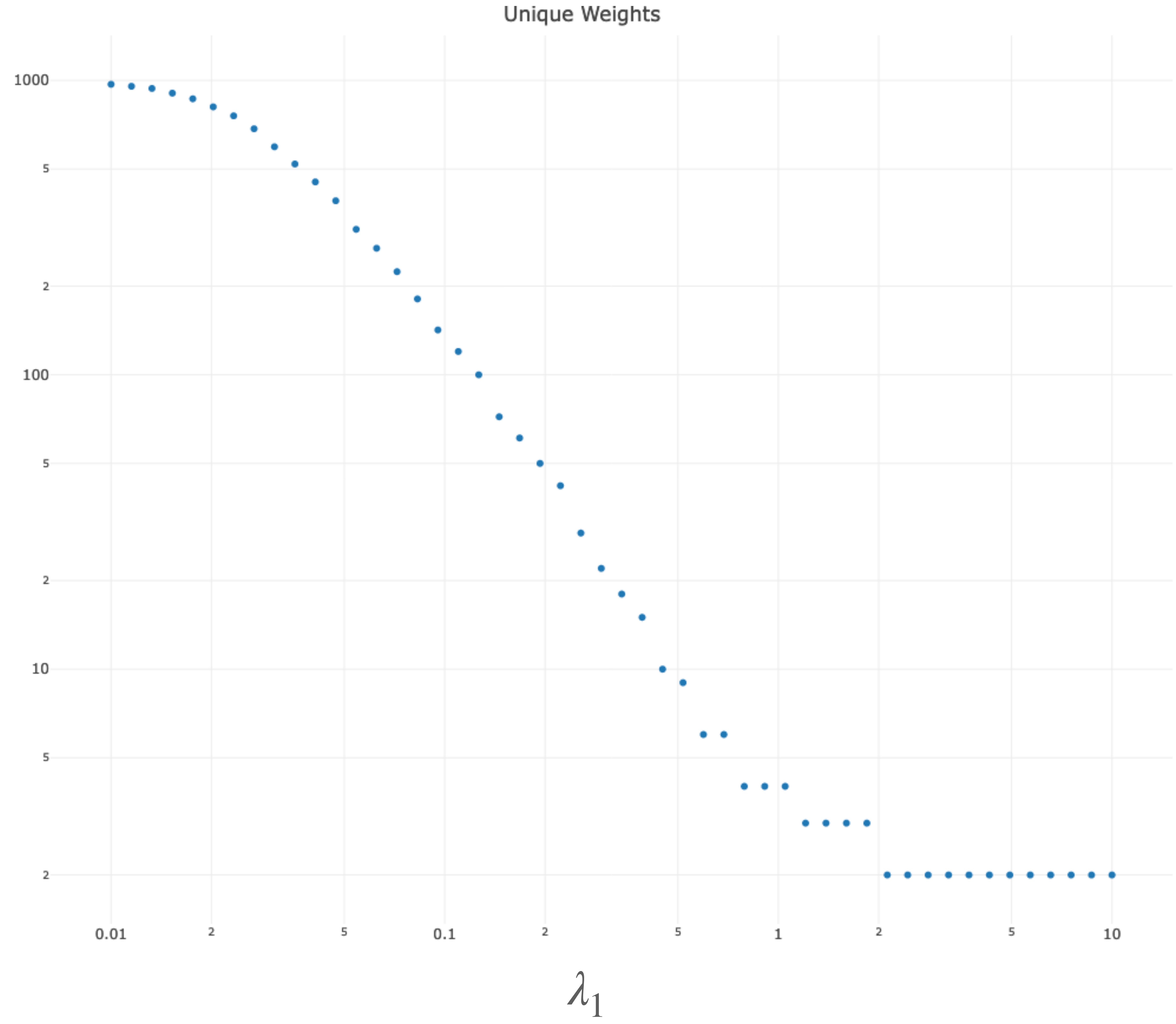
- In practice, we use Nesterov acceleration with fixed step-size
- λ_1 chosen on a grid, $\lambda_2 = C\lambda_1$, post-treatment to reach the desired level of sparsity and such that the weights are equal
- K chosen as the energy-distance kernel
- All implementation in C++ / Rcpp
- Example on a two-dimensional mixture of 3 Gaussians with a sample of size $N = 1000$

Quantization with the MMD — In practice



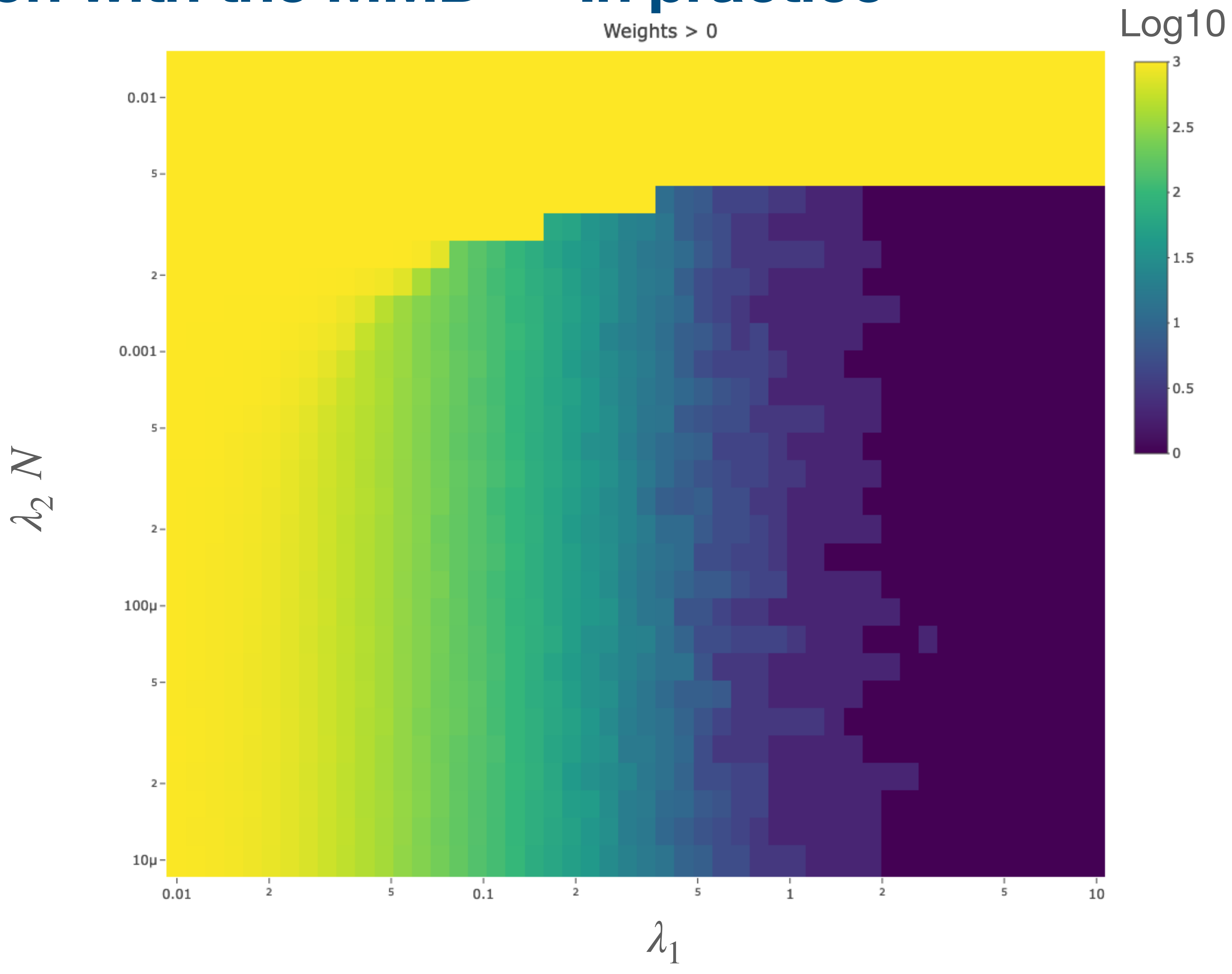
λ_2 fixed

Quantization with the MMD — In practice

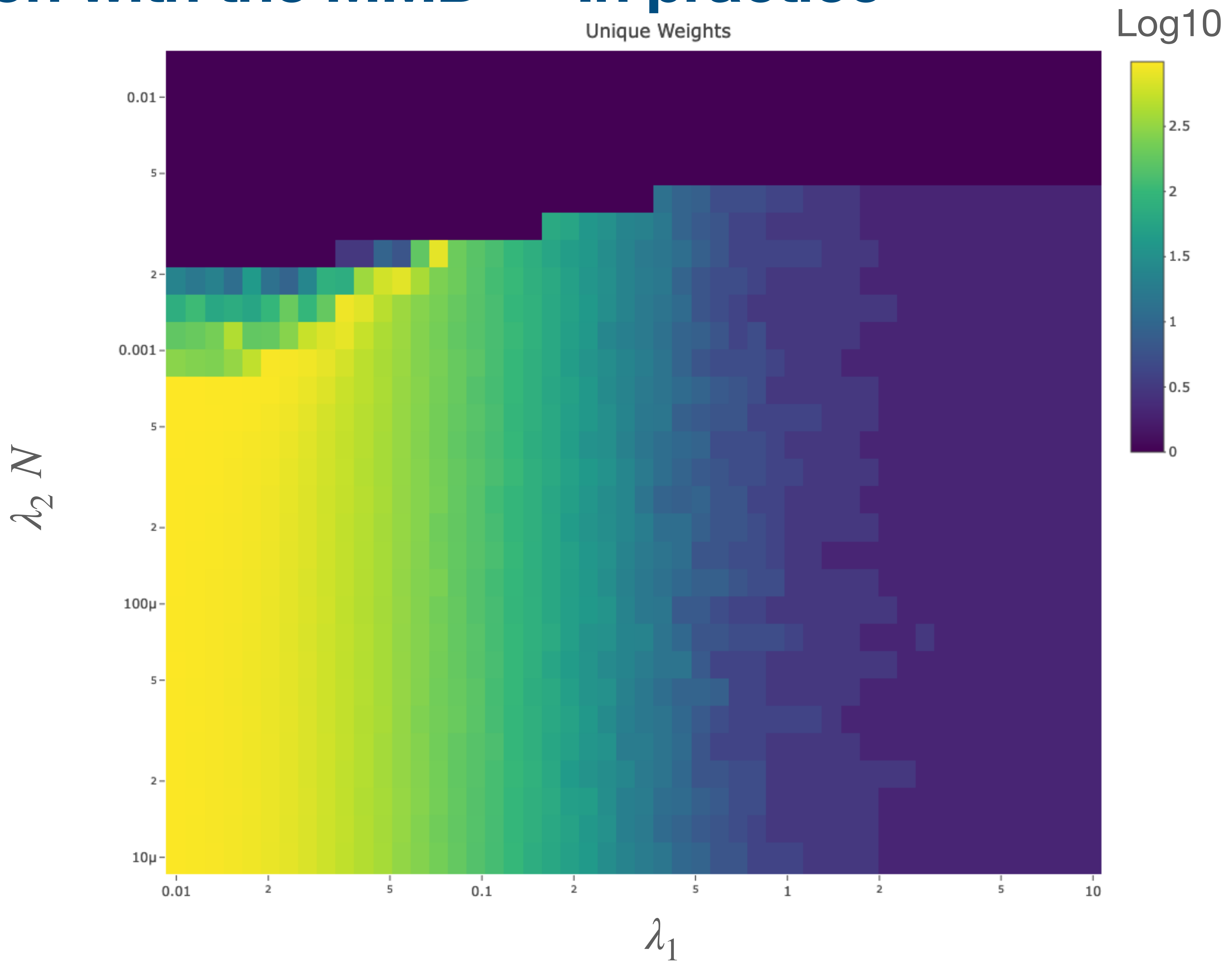


λ_2 fixed

Quantization with the MMD — In practice



Quantization with the MMD — In practice



Quantization with the MMD — Scaling up

- Computation of the proximal operators is cheap $O(N \log N)$
- The computational bottleneck comes from the gradient computation in $O(N^2)$

$$2(K - \lambda_1 I)\mathbf{w} - \mathbf{k}^T$$

- Quantization of very large data sets (big data reduction) is thus out of reach in this setting
- A workaround is tu use a **stochastic proximal gradient** approach instead
  - It will be based on an approximation of the kernel

Quantization with the MMD — Random Fourier Features

- A powerful result for stationary kernels has been proposed by Rahimi & Recht 2007
- It is simply based on Bochner's theorem for stationary kernels which states that

$$
\begin{aligned}
k(\mathbf{x}-\mathbf{y}) &= \int_{\mathbb{R}^d} e^{i\mathbf{u}^T(\mathbf{x}-\mathbf{y})}\hat{k}(\mathbf{u})d\mathbf{u} \\
&= \mathbb{E}_{\mathbf{u}\sim\hat{k}}\left[e^{i\mathbf{u}^T\mathbf{x}}e^{i\mathbf{u}^T\mathbf{y}}\right] \\
&\approx \frac{1}{D}\sum_{j=1}^{D} z_{\mathbf{u}_j}(\mathbf{x})z_{\mathbf{u}_j}(\mathbf{y}) = \mathbf{z}(\mathbf{x})^T\mathbf{z}(\mathbf{y})
\end{aligned}
$$

$$\mathbf{z}(\mathbf{x}) := \left[\cos(\mathbf{u}_1^T\mathbf{x})\ldots\cos(\mathbf{u}_D^T\mathbf{x})\,\sin(\mathbf{u}_1^T\mathbf{x})\ldots\sin(\mathbf{u}_D^T\mathbf{x})\right]^T/\sqrt{D}$$

- We have uniform convergence of the Fourier features (via Hoeffding's inequality)

$$\Pr\left[\sup_{x,y\in\mathcal{M}}|\mathbf{z}(\mathbf{x})'\mathbf{z}(\mathbf{y})-k(\mathbf{x},\mathbf{y})|\geq\epsilon\right]\leq 2^8\left(\frac{\sigma_p\,\mathrm{diam}(\mathcal{M})}{\epsilon}\right)^2\exp\left(-\frac{D\epsilon^2}{4(d+2)}\right)$$

Quantization with the MMD — Random Fourier Features

- This Monte-Carlo estimate of the kernel is unbiased
- It can thus be used inside the gradient to produce a stochastic gradient approximation given by

$$\widehat{K} = Z^T Z, \quad [Z]_{ij} = z_{\mathbf{u}_j}(x_i)$$

thus reducing the complexity to $O(ND)$ since $Z: \quad D \times N$, in practice D is around a few hundreds

- At each iteration, new random features $\mathbf{u}_j$ are generated
- This implies that we must known the Fourier transform of the kernel
  - Readily available for e.g. the popular Gaussian or IMQ kernels

Quantization with the MMD — Stochastic gradient algorithm

Algorithm 2 Accelerated proximal stochastic gradient algorithm for Problem (6)

Require: Dataset $\mathbb{X}_N = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, Fourier transform $\hat{k}$ of kernel k, number of random Fourier features D, regularization constants $\lambda_1 \in [0, \lambda_{\min}(K)], \lambda_2 > 0$, initial weights $\mathbf{w}^0 \in \mathbb{R}^N$ and sequence of step sizes t_k, $k = 0, \ldots$.

Set $\mathbf{v}^0 = \mathbf{w}^0$

for $k = 0, 1, \ldots$ **do**

Draw D i.i.d. samples $\mathbf{u}_1, \ldots, \mathbf{u}_D \in \mathbb{R}^d$ from $\hat{k}$

Assemble matrix $Z \in \mathbb{R}^{D \times N}$ with i-th column equal to

$$\left[\cos(\mathbf{u}_1^T \mathbf{x}_i) \ldots \cos(\mathbf{u}_D^T \mathbf{x}_i) \sin(\mathbf{u}_1^T \mathbf{x}_i) \ldots \sin(\mathbf{u}_D^T \mathbf{x}_i)\right]^T / \sqrt{D}.$$

$$\mathbf{w}^{k+1} = \mathcal{P}_{\Delta^{N-1}}\left(\mathbf{v}^k - 2t_k((Z^T Z - \lambda_1 I)\mathbf{v}^k - Z^T Z \mathbf{1}_N / N\right)$$
$$\mathbf{w}^{k+1} = \operatorname{prox}_{t_k g_2}(\mathbf{w}^{k+1})$$
$$\mathbf{v}^{k+1} = \mathbf{w}^{k+1} + \tfrac{k}{k+3}(\mathbf{w}^{k+1} - \mathbf{w}^k)$$

end for

Conclusion

$$\arg\min_{\mathbf{w}} \mathbf{w}^T(K-\lambda_1 I)\mathbf{w} - \mathbf{k}^T\mathbf{w} + 1\left(\mathbf{w}\in\Delta^{N-1}\right) + \lambda_2 \sum_{i<j}^{N} |w_i - w_j|$$

- New formulation of the MMD quantization problem with sparsity regularization

  1. Convex formulation with a global minimum

  2. Solved with proximal gradient, based on recent results for the Clustered Lasso and sparsity in the simplex

- Stochastic version relying on RFF for large-scale problems

- **The price to pay comes from the tuning of the regularization parameters**

Conclusion

$$\arg\min_{\mathbf{w}} \mathbf{w}^T(K-\lambda_1 I)\mathbf{w} - \mathbf{k}^T\mathbf{w} + 1\left(\mathbf{w}\in\Delta^{N-1}\right) + \lambda_2 \sum_{i<j}^{N} |w_i - w_j|$$

- New formulation of the MMD quantization problem with sparsity regularization
  1. Convex formulation with a global minimum
  2. Solved with proximal gradient, based on recent results for the Clustered Lasso and sparsity in the simplex
- Stochastic version relying on RFF for large-scale problems
- **The price to pay comes from the tuning of the regularization parameters**
- Open questions
  1. Path algorithm for our formulation? (recent results for the Clustered Lasso)
  2. Direct link between the parameters and the sparsity level?
  3. Acceleration via second-order method (recent results for the Clustered Lasso)