Regularized Optimal Transport

Marco Cuturi

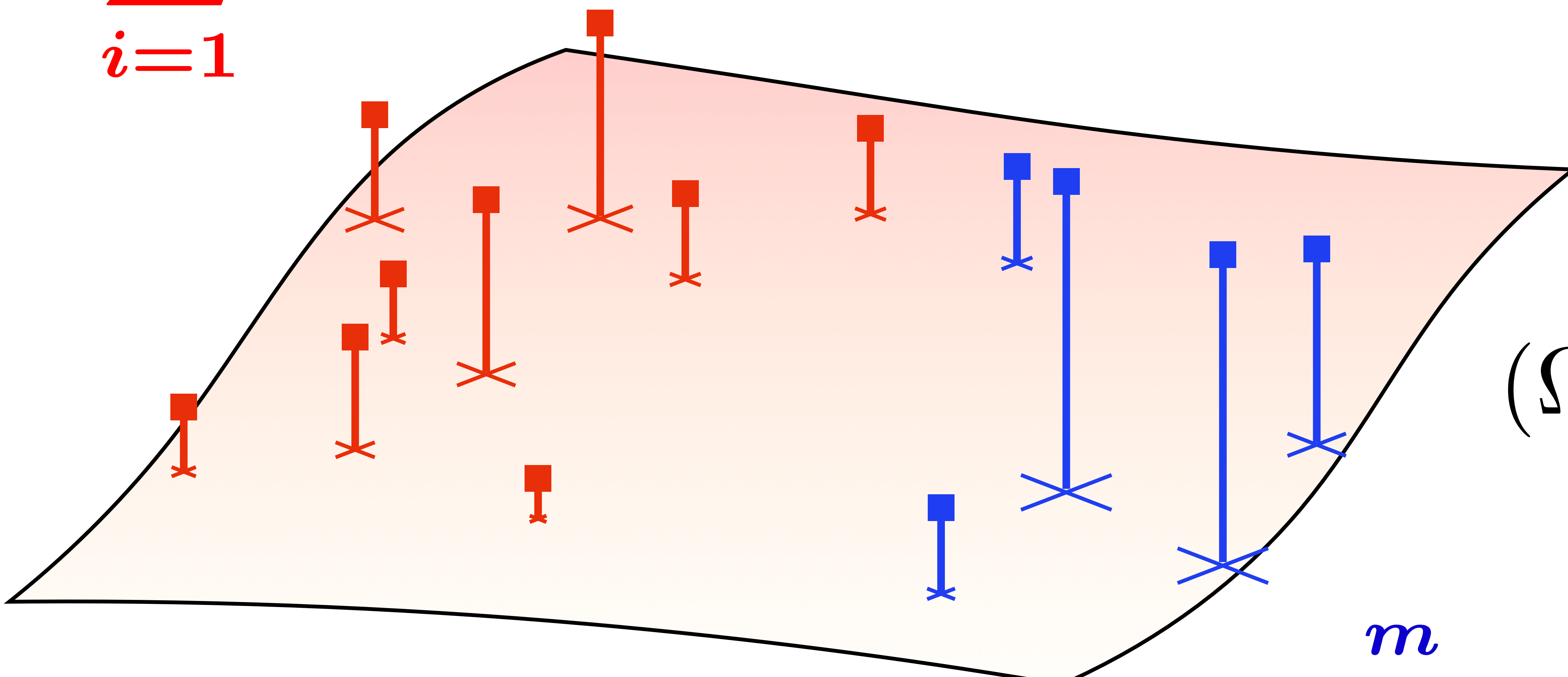
book with Gabriel Peyré

https://optimaltransport.github.io/

Joint works with G. Peyré, F. Bach, N. Bonneel, A. Genevay, L. Chizat, A. Rolet, J. Solomon, G. Carlier, JD Benamou, L. Nenna, M. Heitz, and many others.

Optimal Transport

$$\mu = \sum_{i=1}^{n} a_i \delta_{x_i}$$

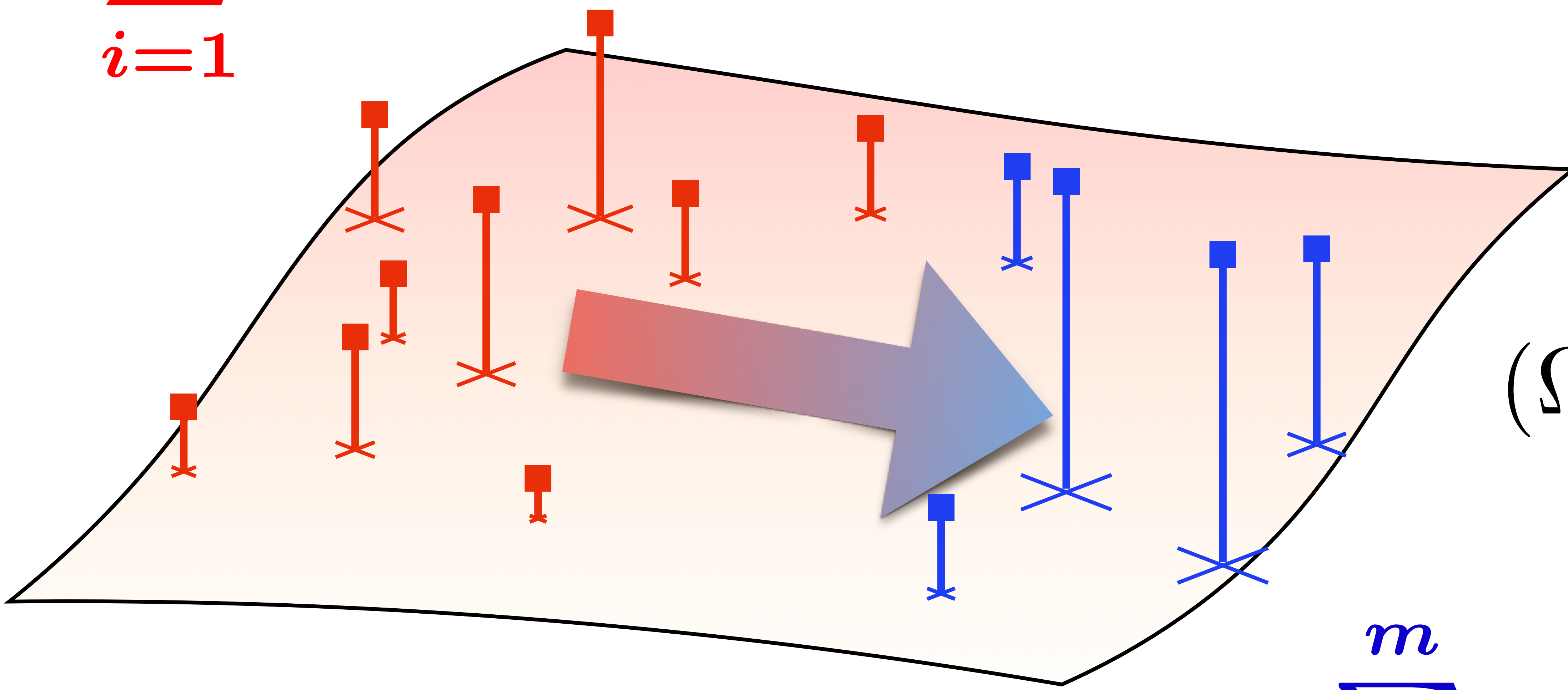


$$(\Omega, D)$$

$$\nu = \sum_{j=1}^{m} b_j \delta_{y_j}$$

Optimal Transport

$$\boldsymbol{\mu} = \sum_{i=1}^{n} \boldsymbol{a_i \delta_{x_i}}$$

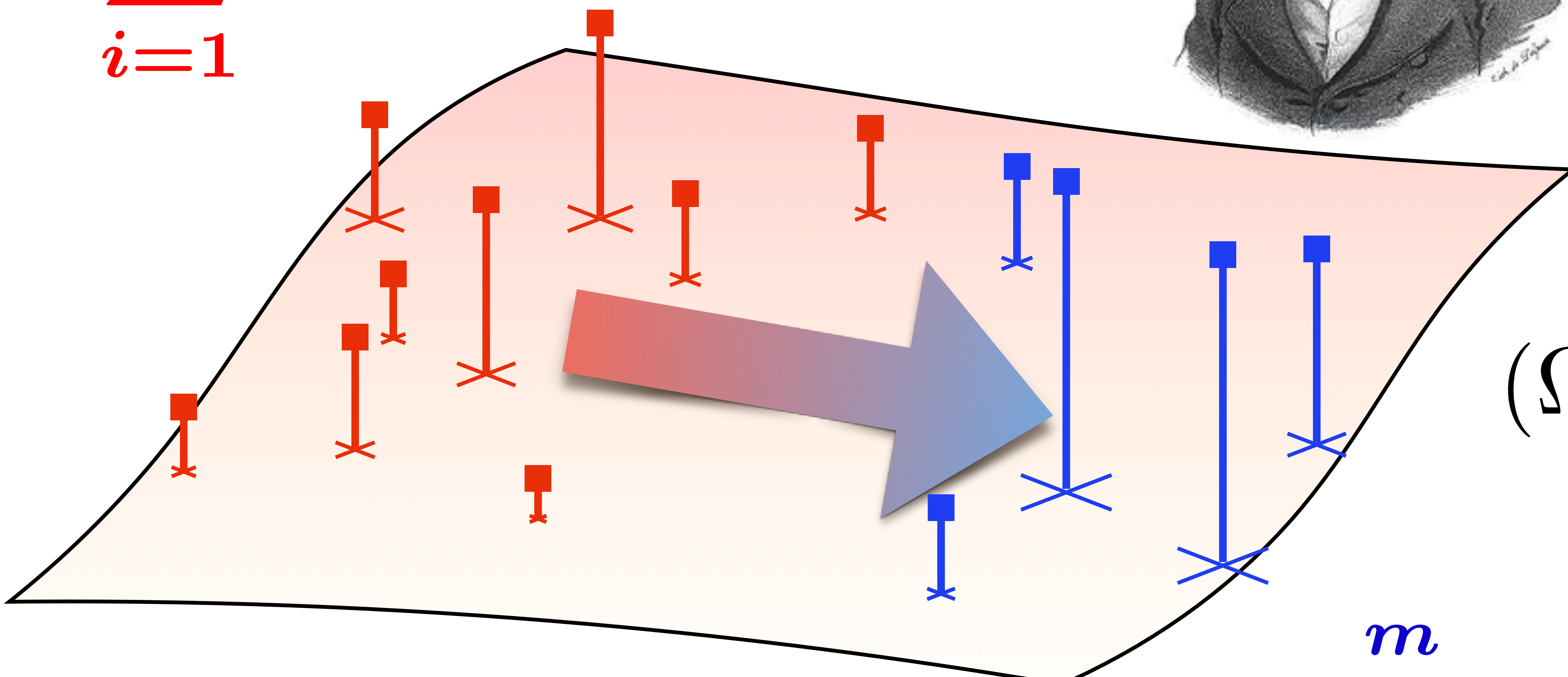


$$(\Omega, \boldsymbol{D})$$

$$\boldsymbol{\nu} = \sum_{j=1}^{m} \boldsymbol{b_j \delta_{y_j}}$$

Optimal Transport

$$\mu = \sum_{i=1}^{n} a_i \delta_{x_i}$$

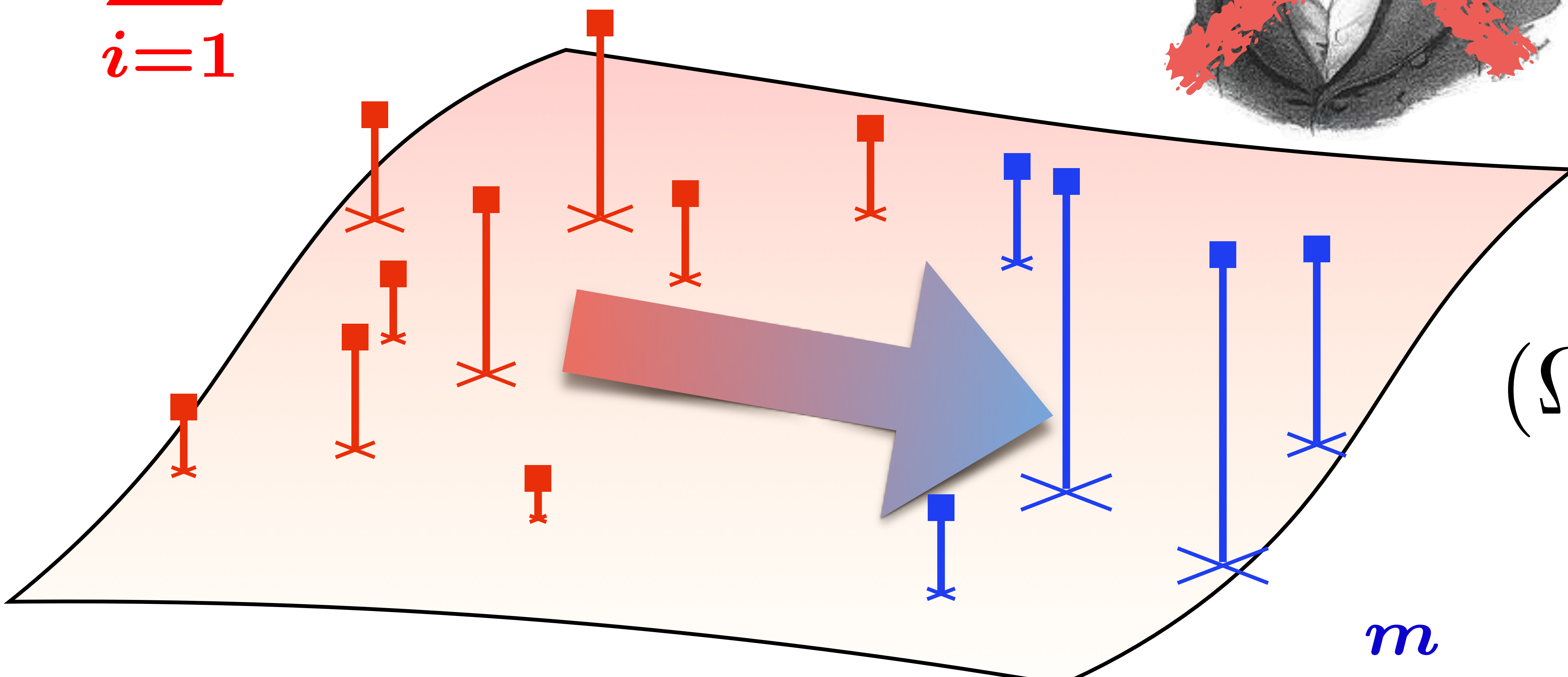


$$(\Omega, D)$$

$$\nu = \sum_{j=1}^{m} b_j \delta_{y_j}$$

Optimal Transport

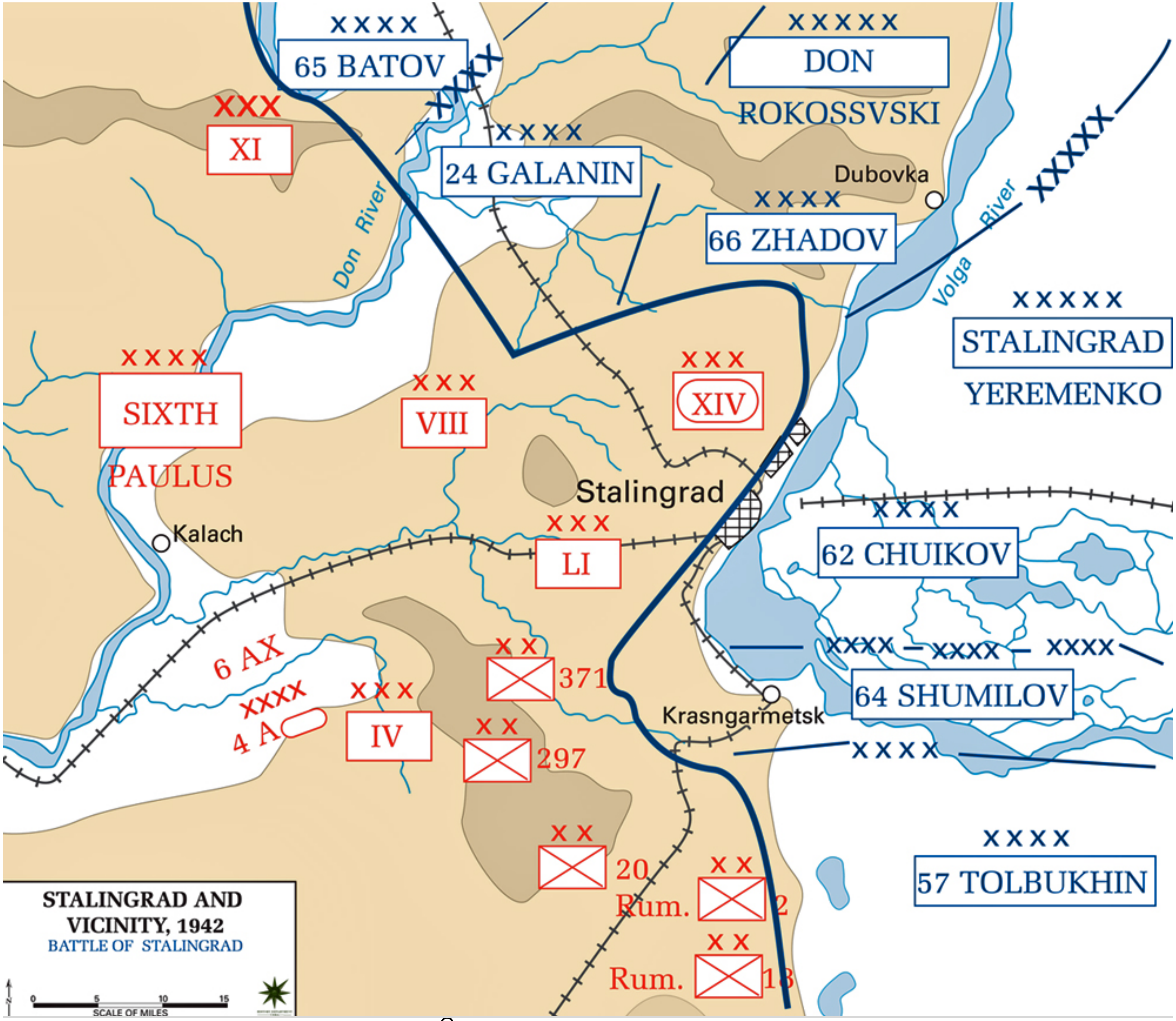
$$\mu = \sum_{i=1}^{n} a_i \delta_{x_i}$$



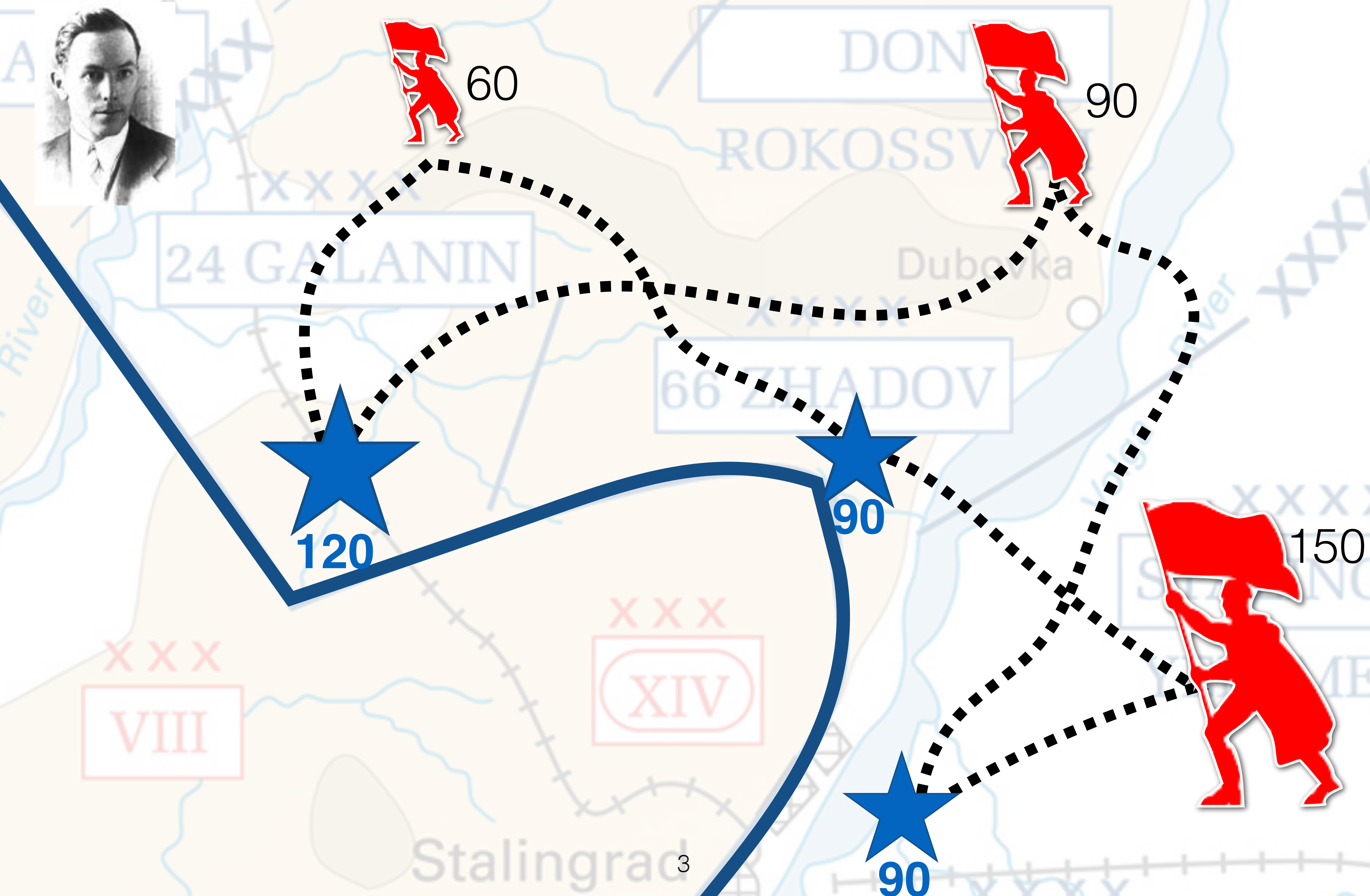
(Ω, D)

$$\nu = \sum_{j=1}^{m} b_j \delta_{y_j}$$

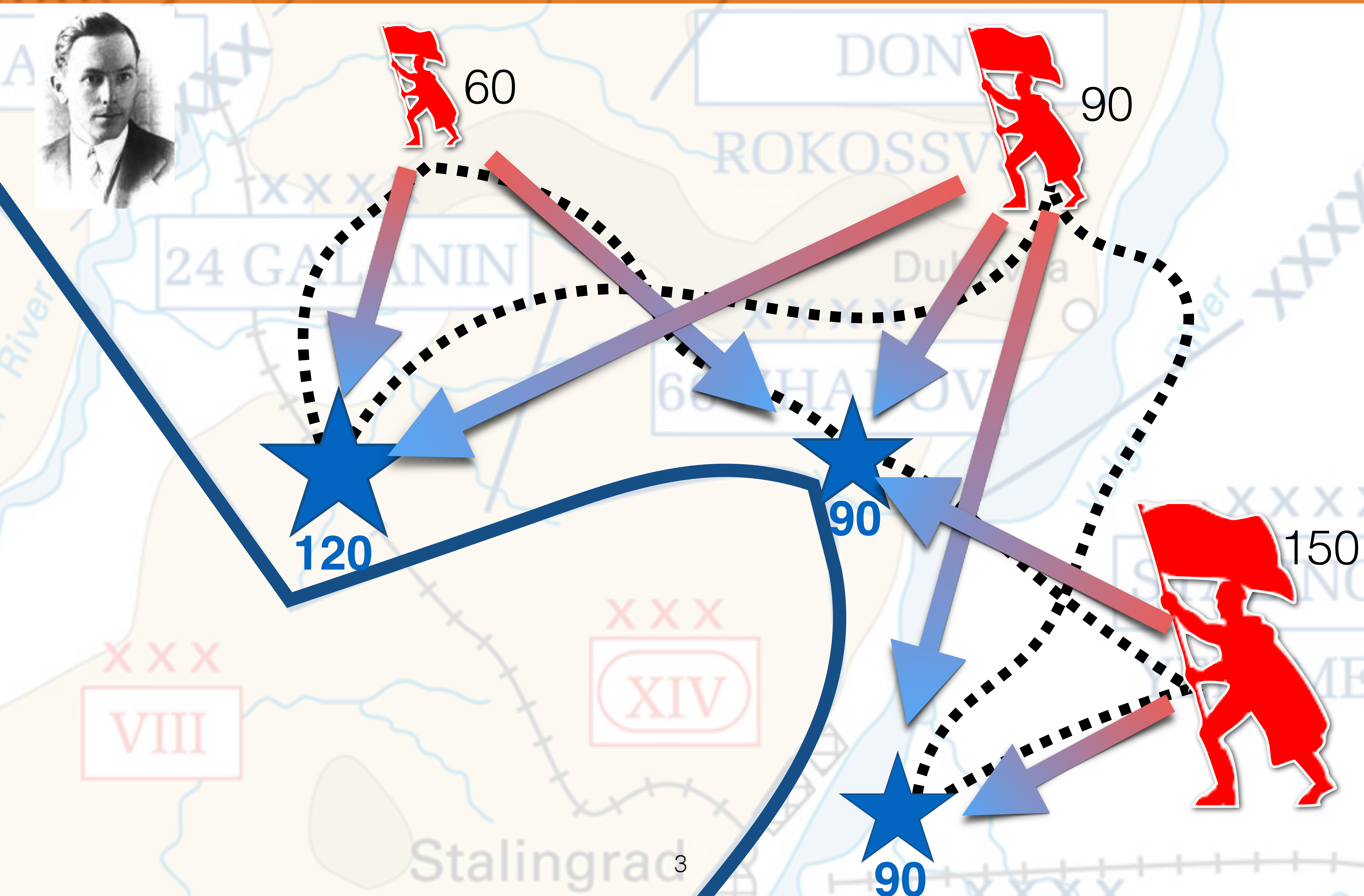
Kantorovich Problem



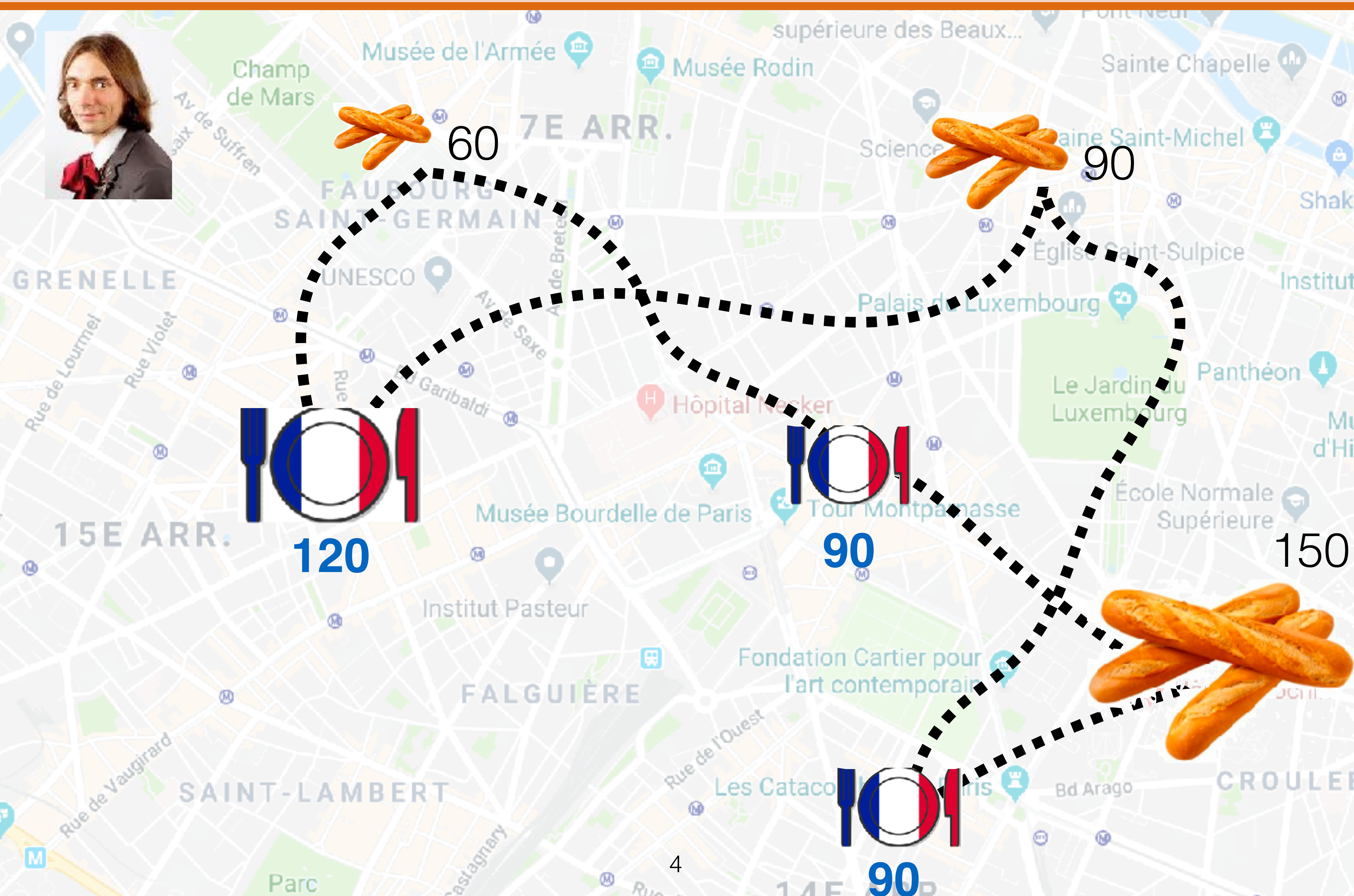
Kantorovich Problem



Kantorovich Problem

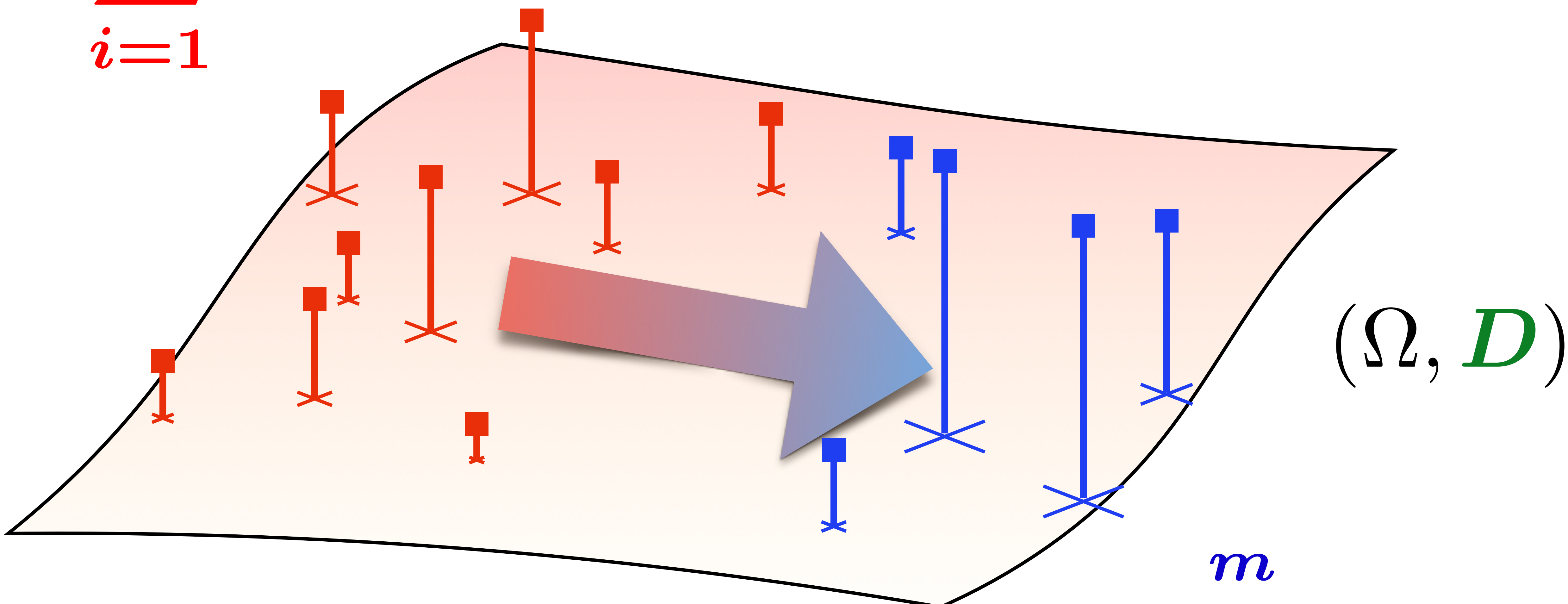


Kantorovich Problem *à la française*



Optimal Transport

$$\mu = \sum_{i=1}^{n} a_i \delta_{x_i}$$



$$\nu = \sum_{j=1}^{m} b_j \delta_{y_j}$$

Wasserstein on Discrete Measures

Consider $\boldsymbol{\mu} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{a_i \delta_{x_i}}$ and $\boldsymbol{\nu} = \sum_{\boldsymbol{j=1}}^{\boldsymbol{m}} \boldsymbol{b_j \delta_{y_j}}$.

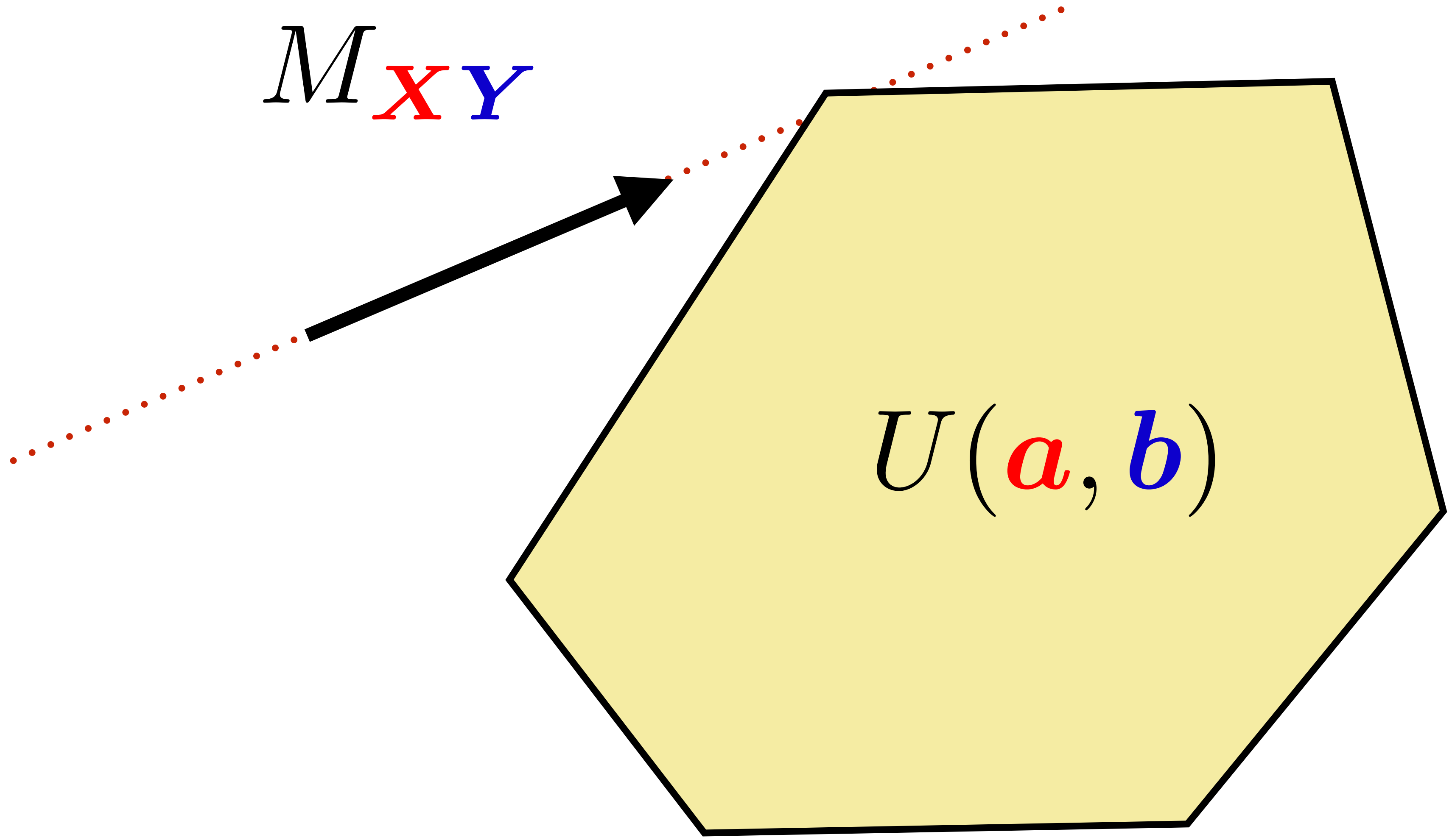
$$M_{\boldsymbol{XY}} \overset{\text{def}}{=} [D(\boldsymbol{x}_i, \boldsymbol{y}_j)^p]_{ij}$$

$$U(\boldsymbol{a}, \boldsymbol{b}) \overset{\text{def}}{=} \{\boldsymbol{P} \in \mathbb{R}_+^{n \times m} \,|\, \boldsymbol{P}\mathbf{1}_m = \boldsymbol{a}, \boldsymbol{P}^T\mathbf{1}_n = \boldsymbol{b}\}$$

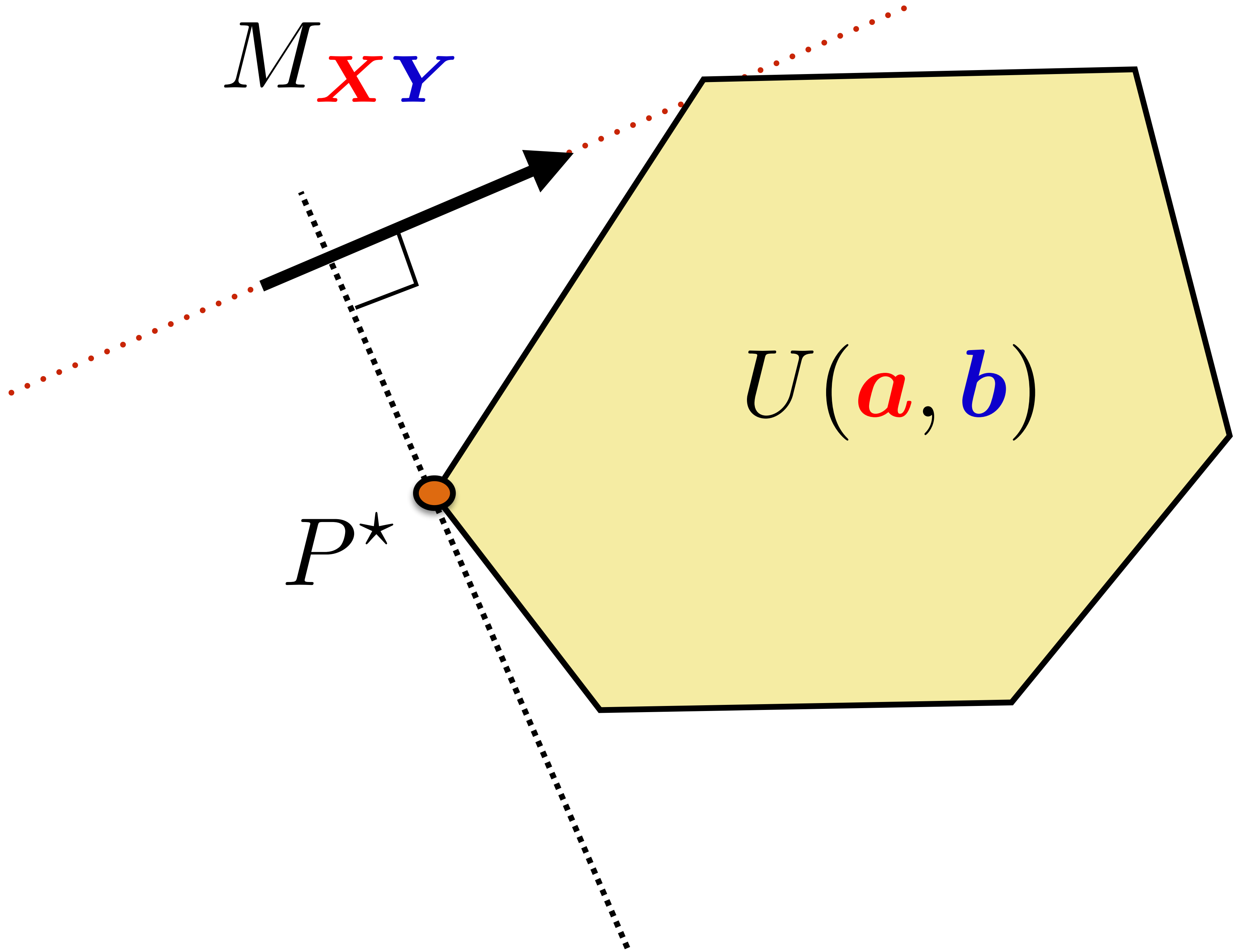
Def. Optimal Transport Problem

$$W_p^p(\boldsymbol{\mu}, \boldsymbol{\nu}) = \min_{\boldsymbol{P} \in U(\boldsymbol{a}, \boldsymbol{b})} \langle \boldsymbol{P}, M_{\boldsymbol{XY}} \rangle$$

Solving the OT Problem



Solving the OT Problem



Early application: Earth Mover's

Early application: Earth Mover’s

Early application: Earth Mover's

[Rubner'98] $\mathrm{dist}(I_1, I_2) = W_1(\mu, \nu)$

Word Mover's Distance

Word Mover's Distance

[Kusner'15] $\quad \mathrm{dist}(D_1, D_2) = W_2(\boldsymbol{\mu}, \boldsymbol{\nu})$

Variational OT Problems in ML

Up to 2010: OT solvers used mostly for retrieval in databases of histograms

$$W_p(\boldsymbol{\mu}, \boldsymbol{\nu}) = ?$$

$$W_p(\boldsymbol{\mu}, \boldsymbol{\nu}) \leq \cdots ?$$

The field has now transitioned to OT as a **loss or fidelity** term

$$\operatorname*{argmin}_{\boldsymbol{\mu} \in \mathcal{P}(\Omega)} F(W_p(\boldsymbol{\mu}, \boldsymbol{\nu_1}), W_p(\boldsymbol{\mu}, \boldsymbol{\nu_2}), \dots, \boldsymbol{\mu}) = ?$$

$$\text{``}\nabla_{\boldsymbol{\mu}}\text{''} W_p(\boldsymbol{\mu}, \boldsymbol{\nu_1}) = ?$$

Recent spike in interest for **[Ambrosio'05]**

"Wasserstein + Data" Problems

- Quantization: k-means problem **[Lloyd'82]**

$$\min_{\substack{\boldsymbol{\mu}\in\mathcal{P}(\mathbb{R}^d)\\ |\operatorname{supp}\boldsymbol{\mu}|=k}} W_2^2(\boldsymbol{\mu},\boldsymbol{\nu}_{\text{data}})$$

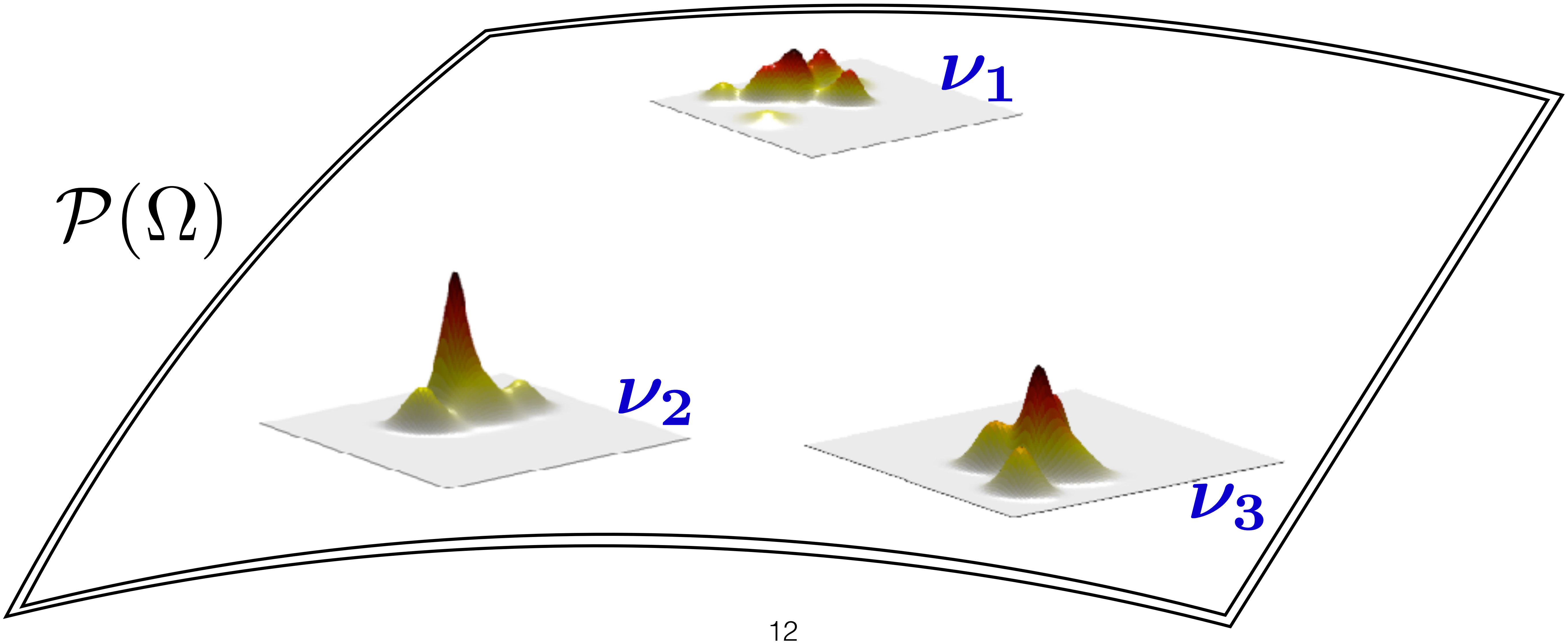
- **[McCann'95]** Interpolant

$$\min_{\boldsymbol{\mu}\in\mathcal{P}(\Omega)} (1-t)W_2^2(\boldsymbol{\mu},\boldsymbol{\nu_1}) + tW_2^2(\boldsymbol{\mu},\boldsymbol{\nu_2})$$

- **[JKO'98]** PDE's as "gradient" flows in $(\mathcal{P}(\Omega), W)$.

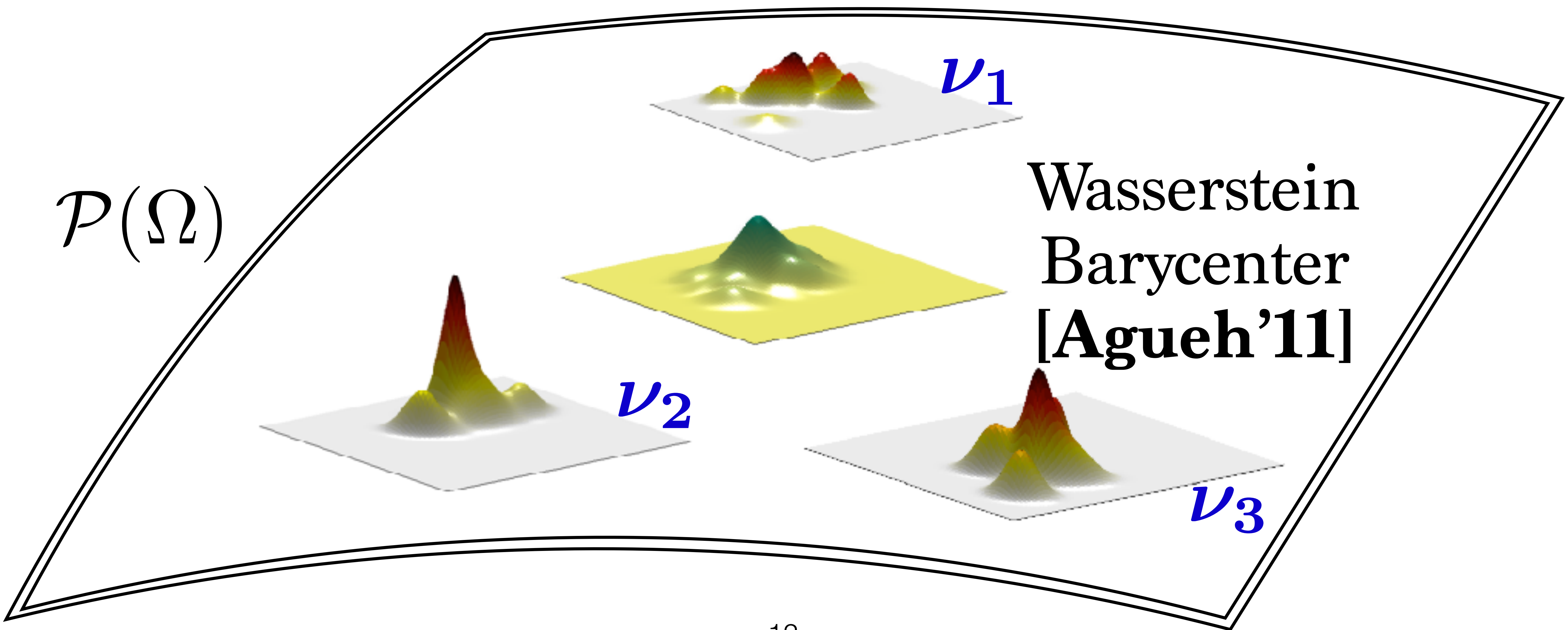
$$\mu_{t+1} = \operatorname*{argmin}_{\boldsymbol{\mu}\in\mathcal{P}(\Omega)} J(\boldsymbol{\mu}) + \lambda_t W_p^p(\boldsymbol{\mu},\mu_t)$$

Example: Barycenters

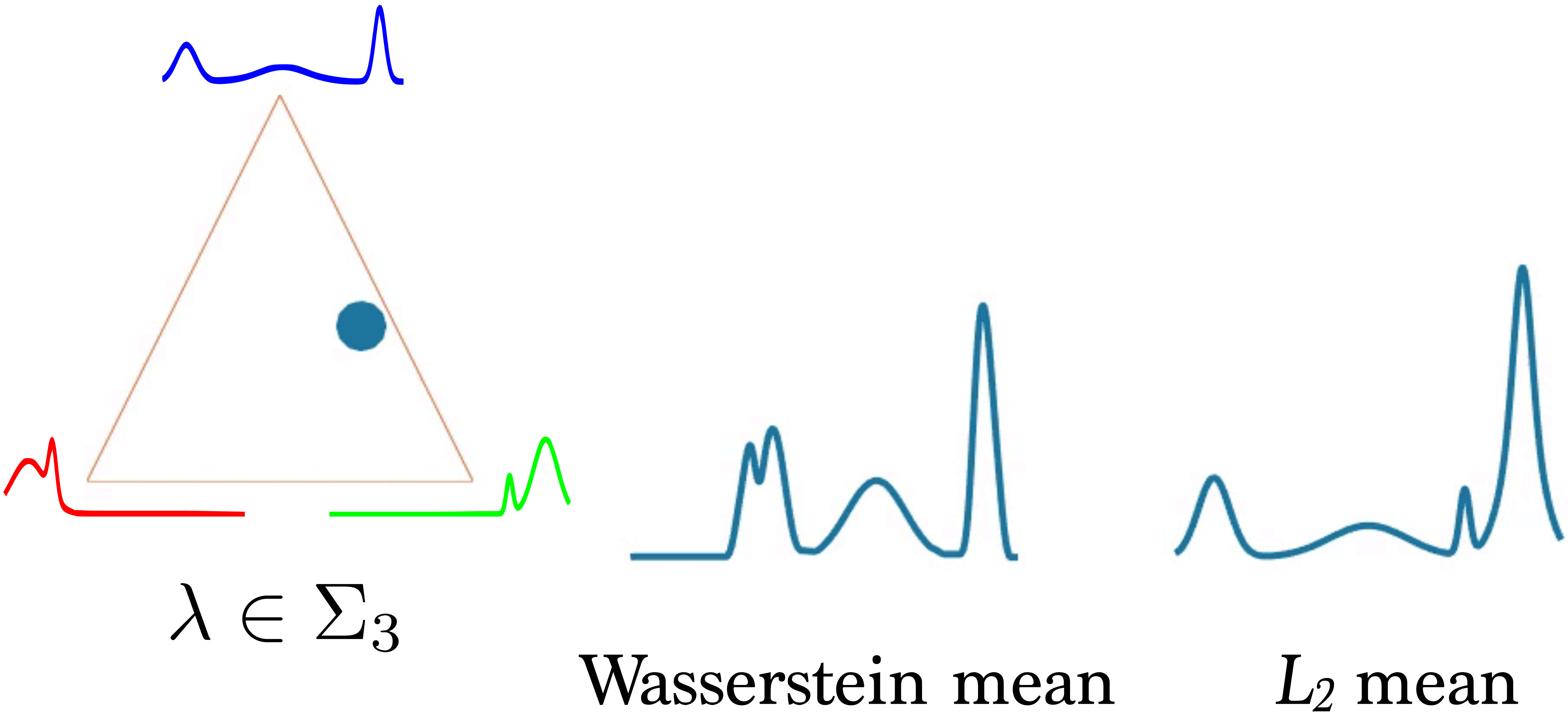


Example: Barycenters

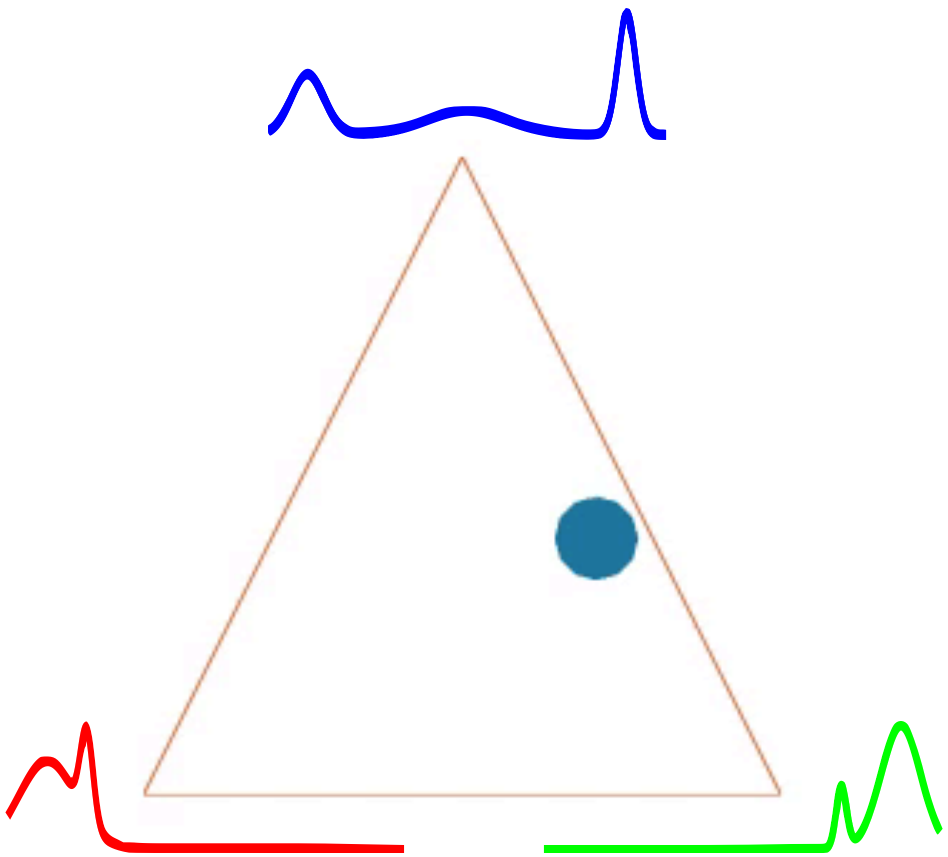
$$\min_{\mu \in \mathcal{P}(\Omega)} \sum_{i=1}^{N} \lambda_i W_p^p(\boldsymbol{\mu}, \boldsymbol{\nu_i})$$



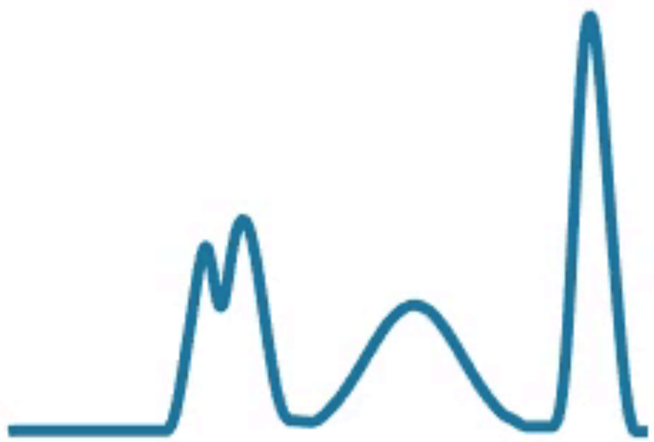
Example: Barycenters



Example: Barycenters



$\lambda \in \Sigma_3$



Wasserstein mean

L_2 mean

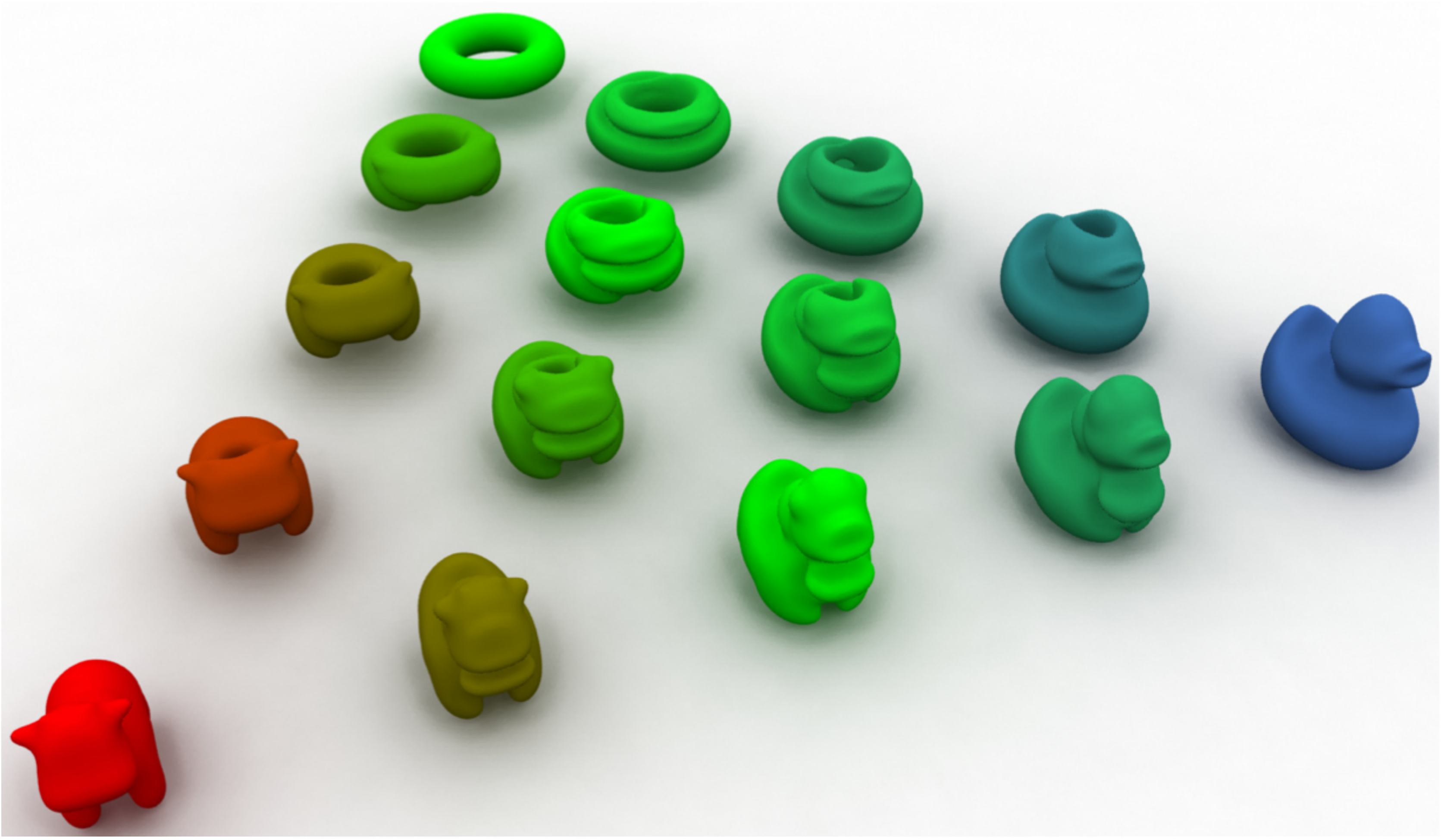
Ex: Barycenters for shapes

Graphics: simple testing ground for relevance of Wasserstein geometry

[SDPC..'15]

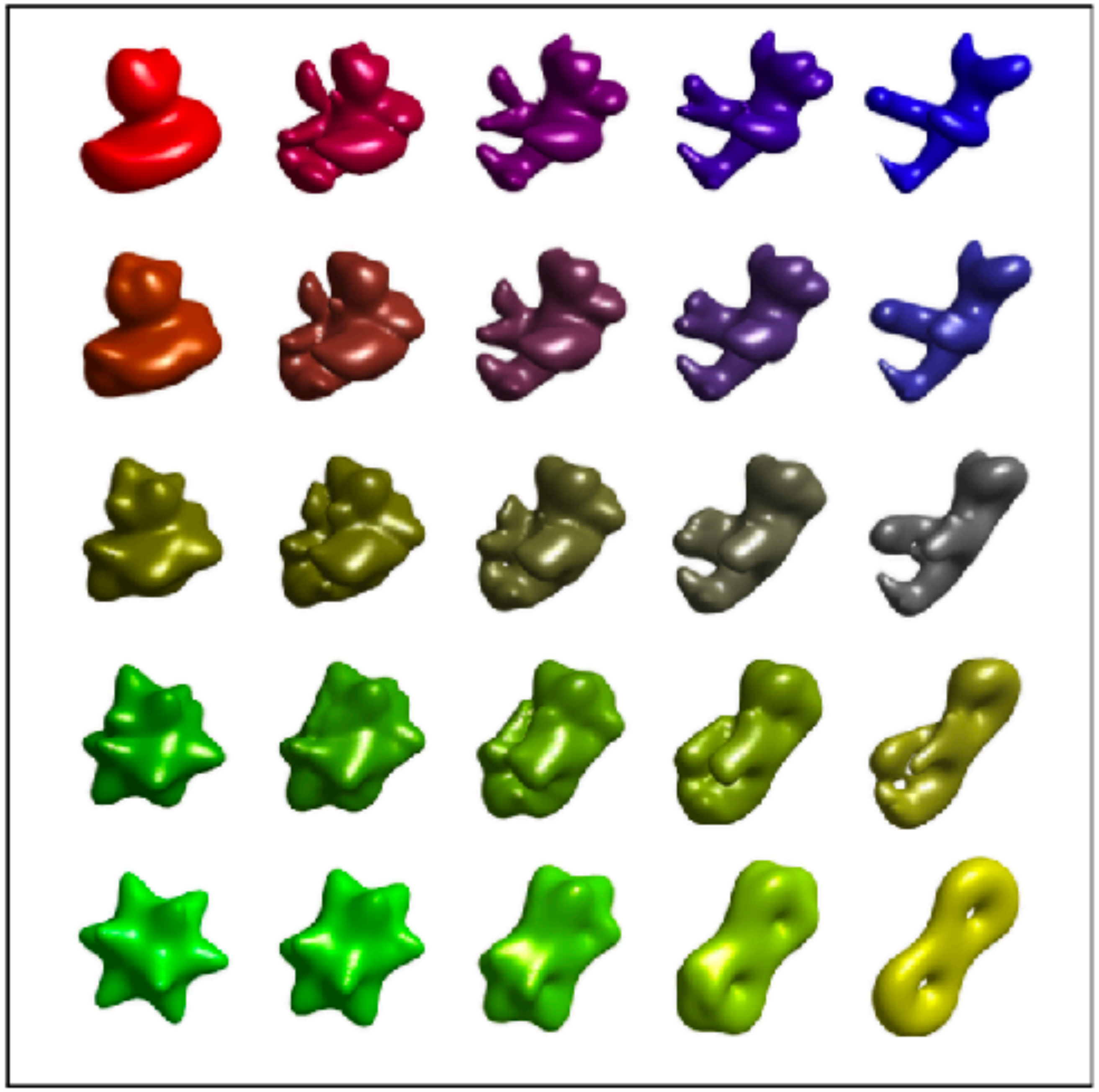
Ex: Barycenters for shapes

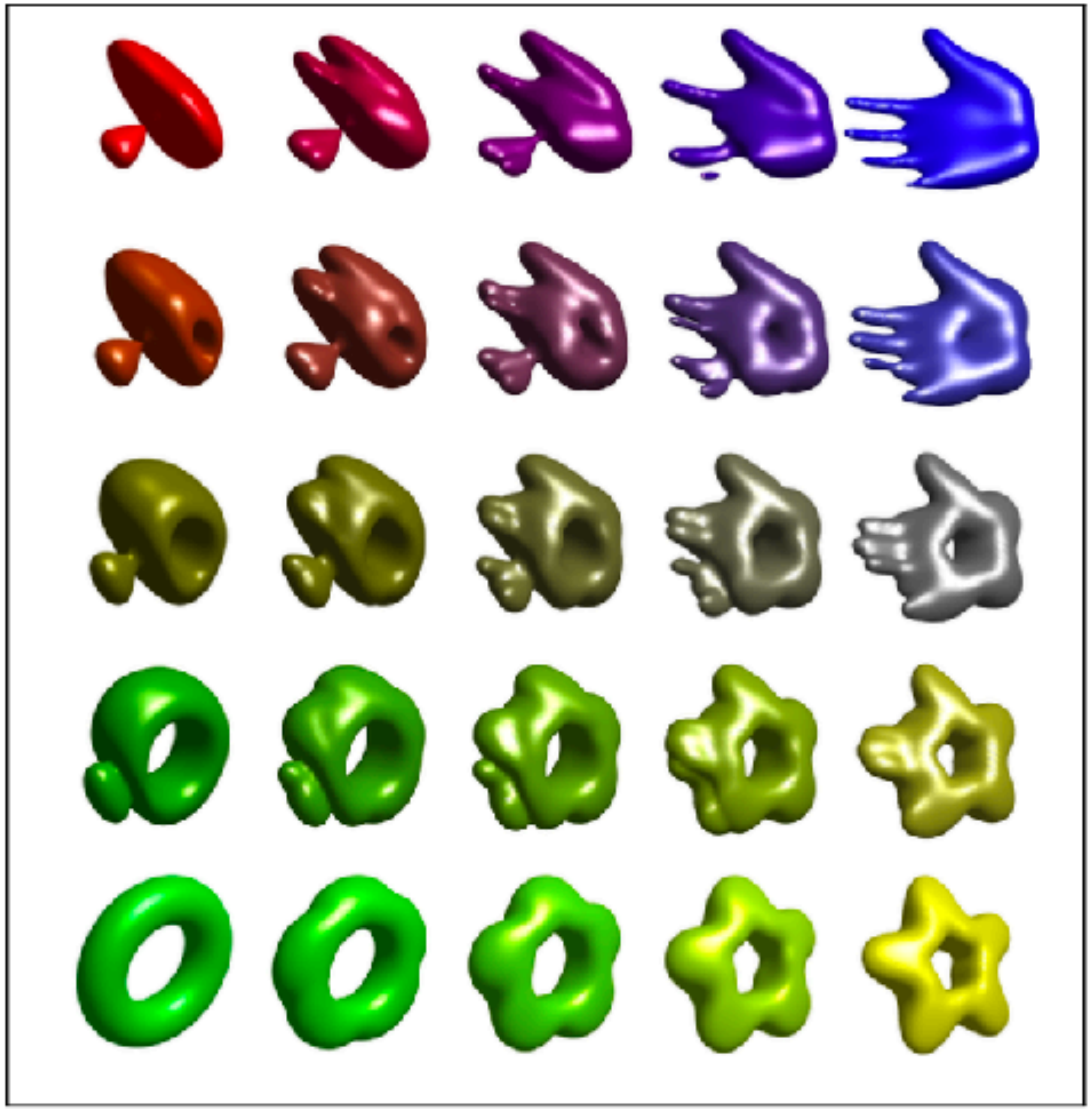
Graphics: simple testing ground for relevance of Wasserstein geometry



[SDPC..'15]

Ex: Barycenters for shapes





[PC'18]

Example: Learning with a W Loss

Dataset $\{(x_i, y_i)\}, x_i \in \mathbb{R}^p, y_i \in \mathbb{R}^n_+$

x_i

husky
snow
sled
slope
men

y_i

Goal is to find $f_{\boldsymbol{\theta}}$: Images $\mapsto$ Labels

Example: Learning with aW Loss

$$\min_{\boldsymbol{\theta} \in \Theta} \sum_{i=1}^{N} \mathcal{L}(f_{\boldsymbol{\theta}}(x_i), y_i)$$

x_i

husky
snow
sled
slope
men

y_i

Example: Learning with a W Loss

$$\min_{\boldsymbol{\theta} \in \Theta} \sum_{i=1}^{N} \mathcal{L}(f_{\boldsymbol{\theta}}(x_i), y_i)$$

$f_{\boldsymbol{\theta}}(x_i)$	y_i
dog	husky
driver	snow
winter	sled
ice	slope
	men

Example: Learning with a W Loss

$$\min_{\boldsymbol{\theta} \in \Theta} \sum_{i=1}^{N} \mathcal{L}(f_{\boldsymbol{\theta}}(x_i), y_i)$$

dog driver winter ice	husky snow sled slope men
$f_{\boldsymbol{\theta}}(x_i)$	y_i

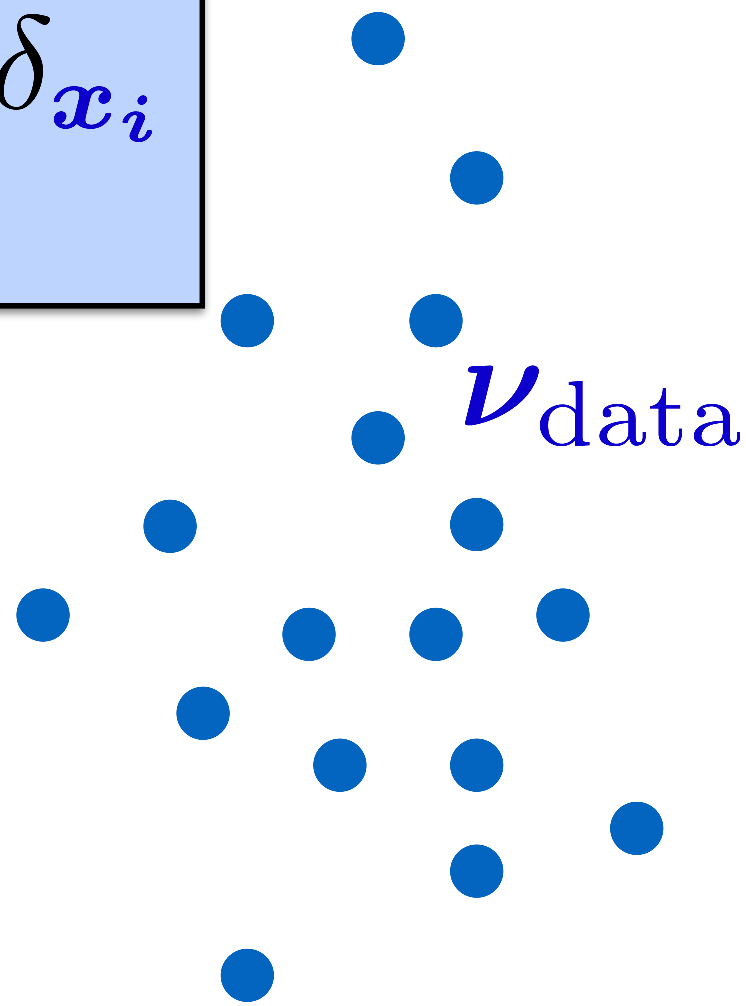
Use for $\mathcal{L}$ a Wasserstein type loss.

[Frogner'15]

Example: Generative Models

We collect data

$$\nu_{\text{data}} = \frac{1}{N} \sum_{i=1}^{N} \delta_{\boldsymbol{x}_i}$$



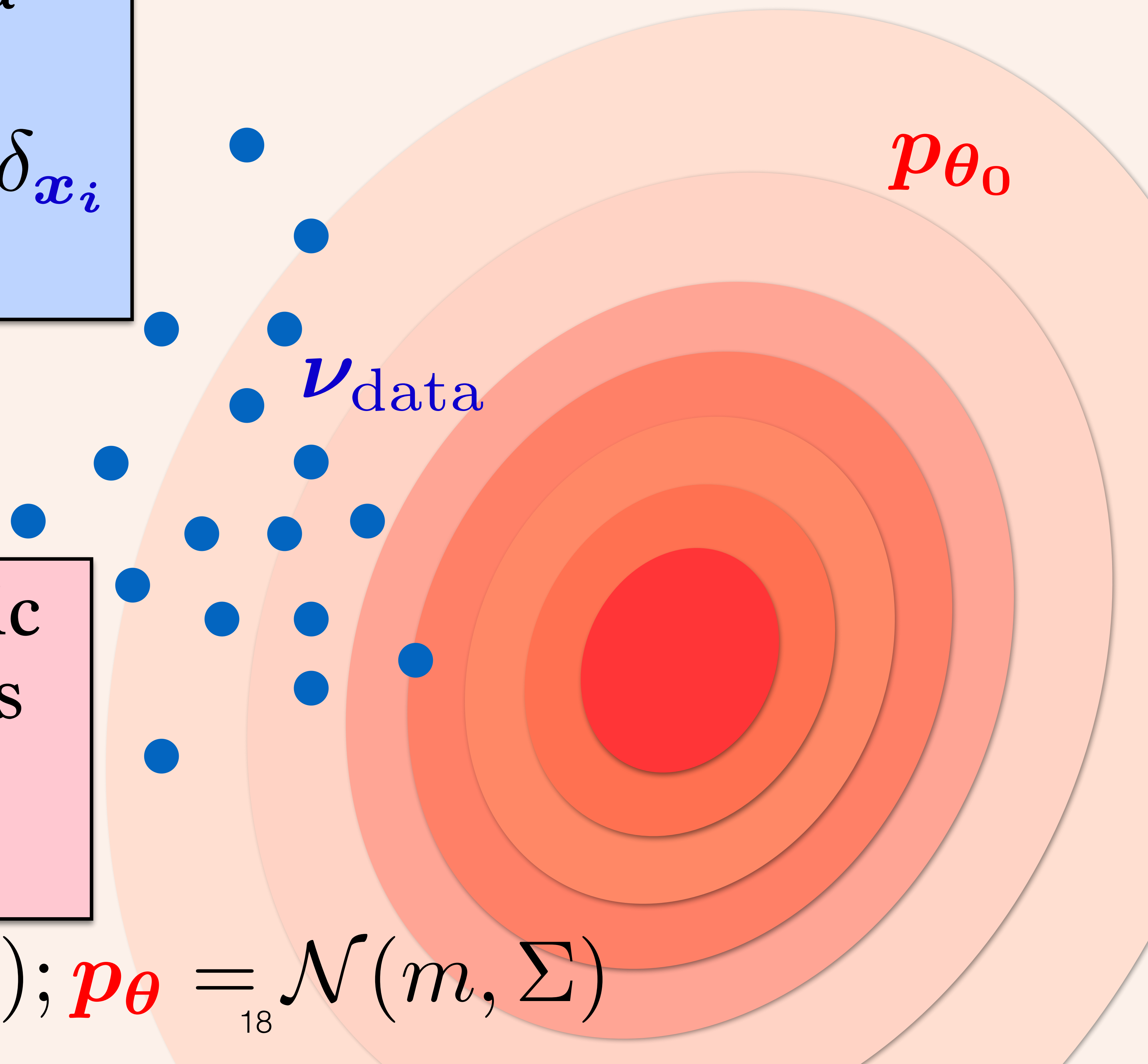
Example: Generative Models

We collect data

$$\boldsymbol{\nu}_{\text{data}} = \frac{1}{N}\sum_{i=1}^{N}\delta_{\boldsymbol{x_i}}$$

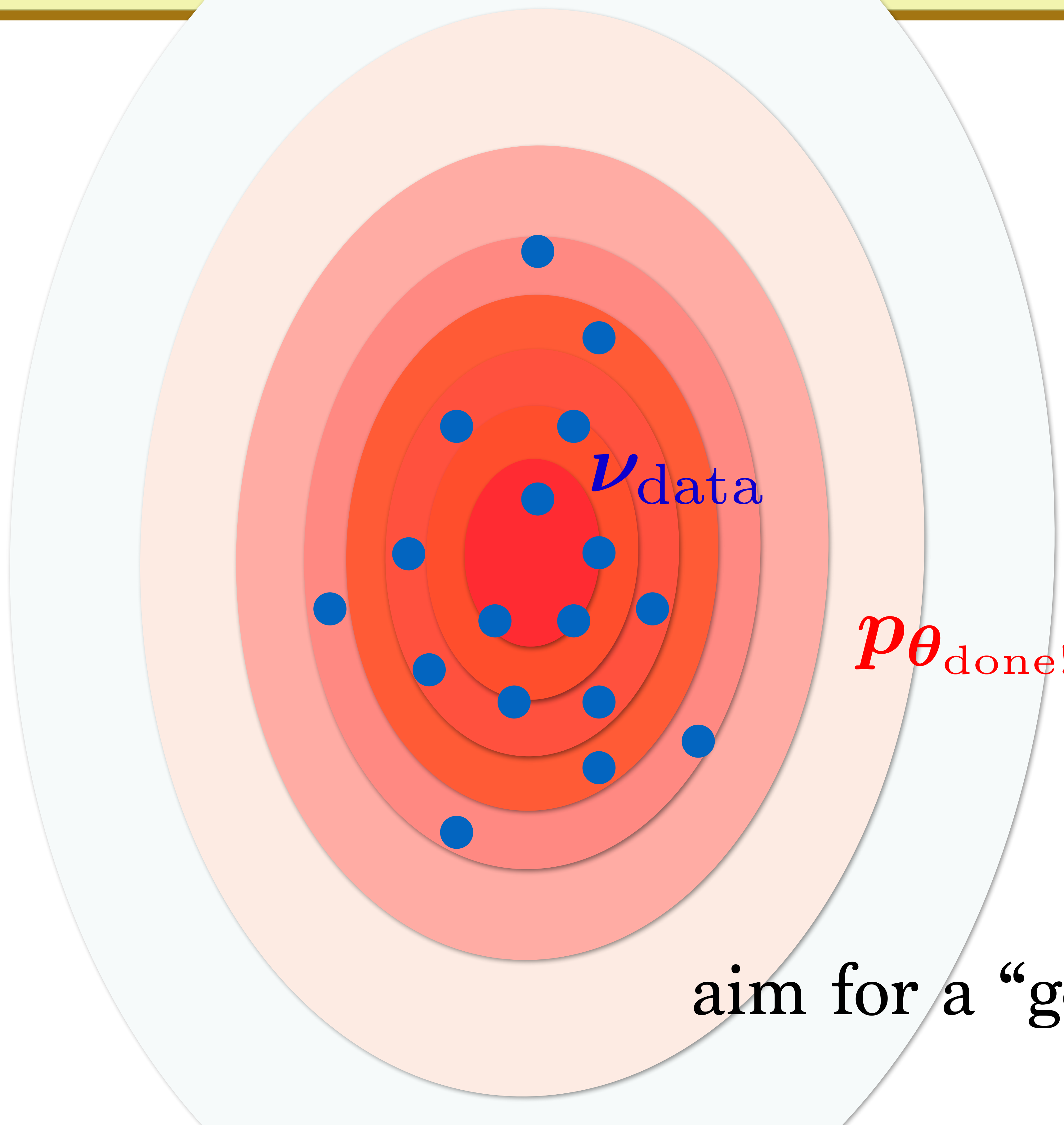
We fit a parametric family of densities

$$\{\boldsymbol{p_\theta}, \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$$



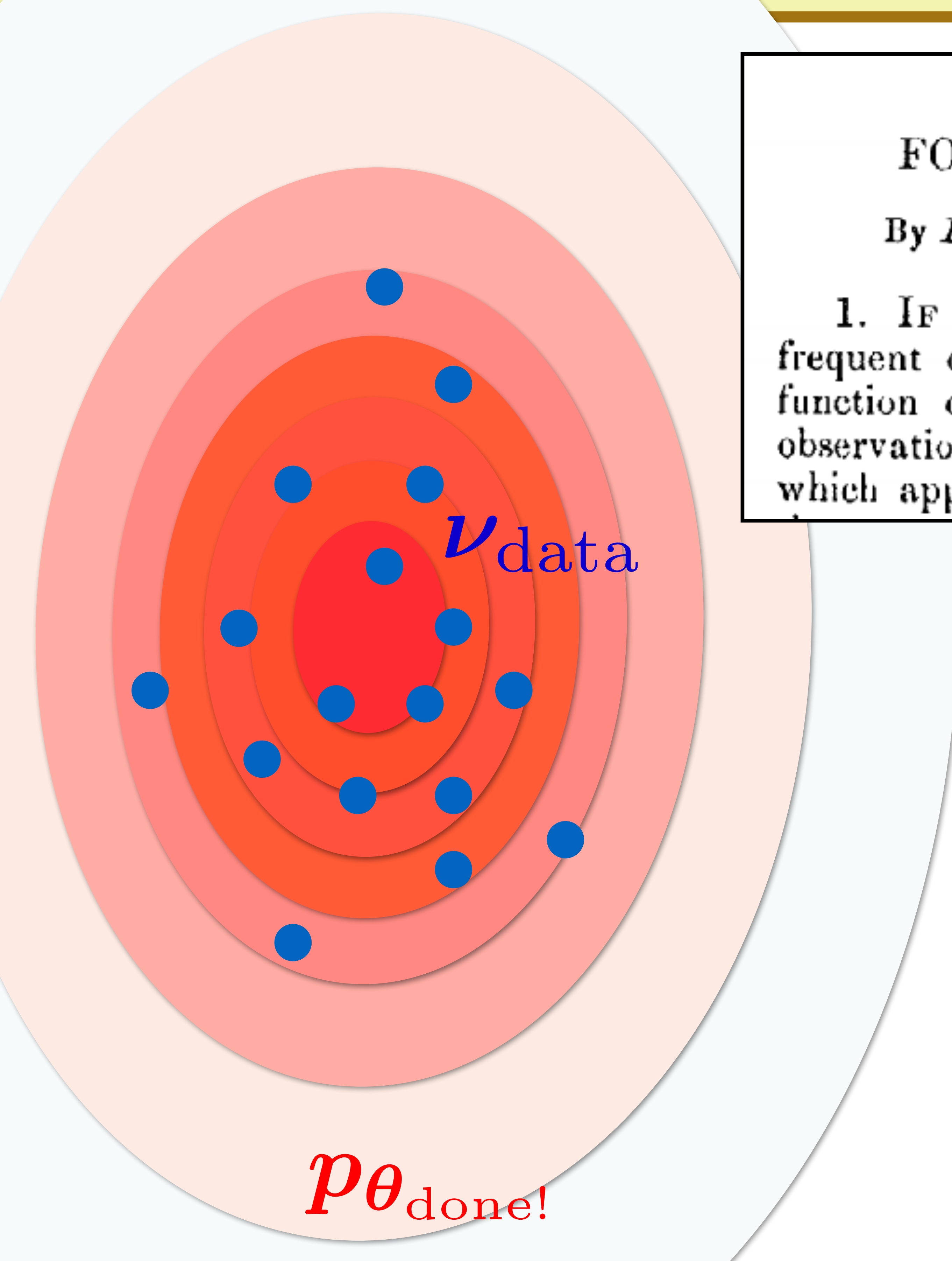
$$e.g.\ \boldsymbol{\theta} = (m, \Sigma); \boldsymbol{p_\theta} = \mathcal{N}(m, \Sigma)$$

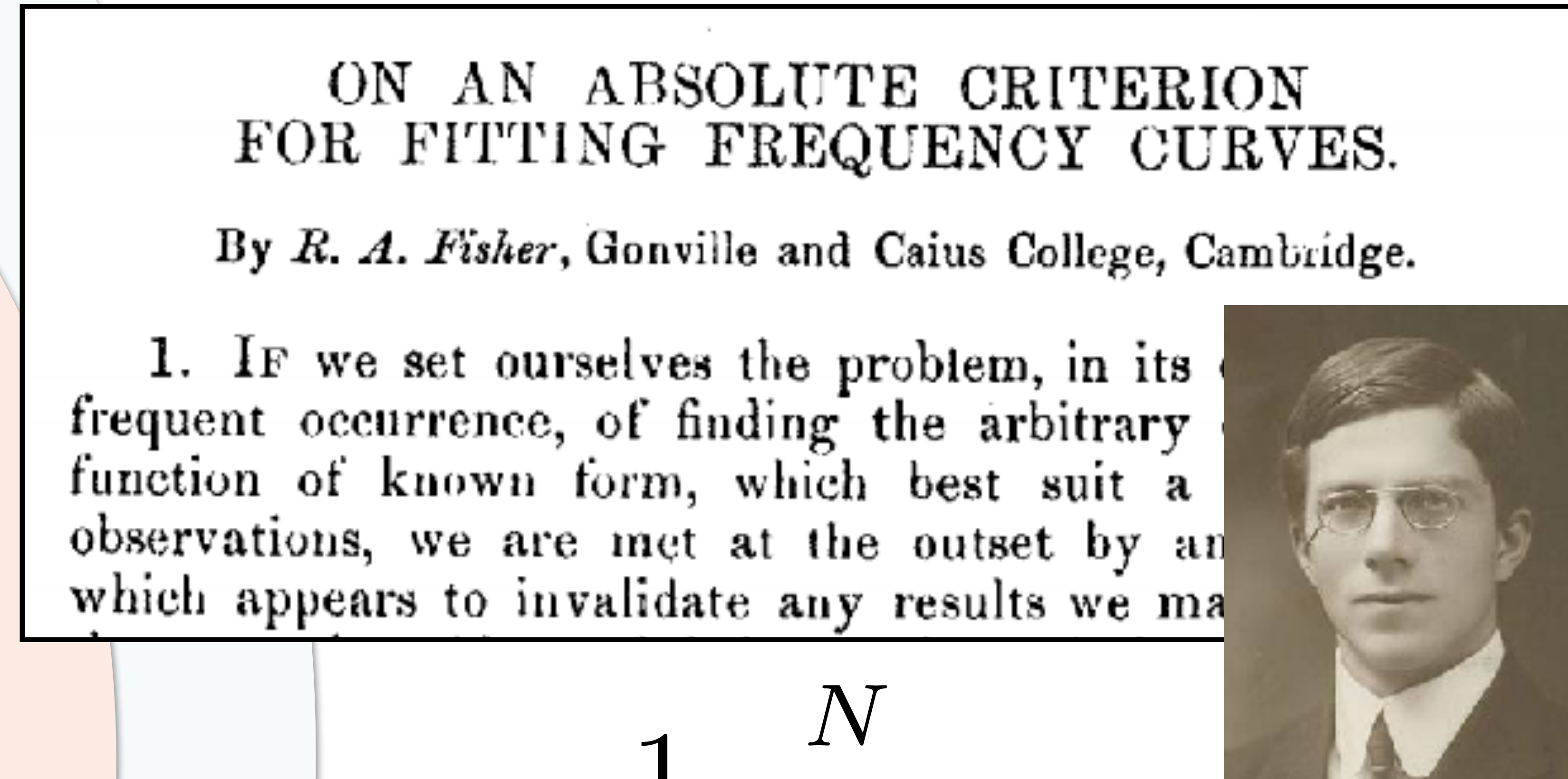
Statistics 0.1: Density Fitting



aim for a “good” fit

Maximum Likelihood Estimation





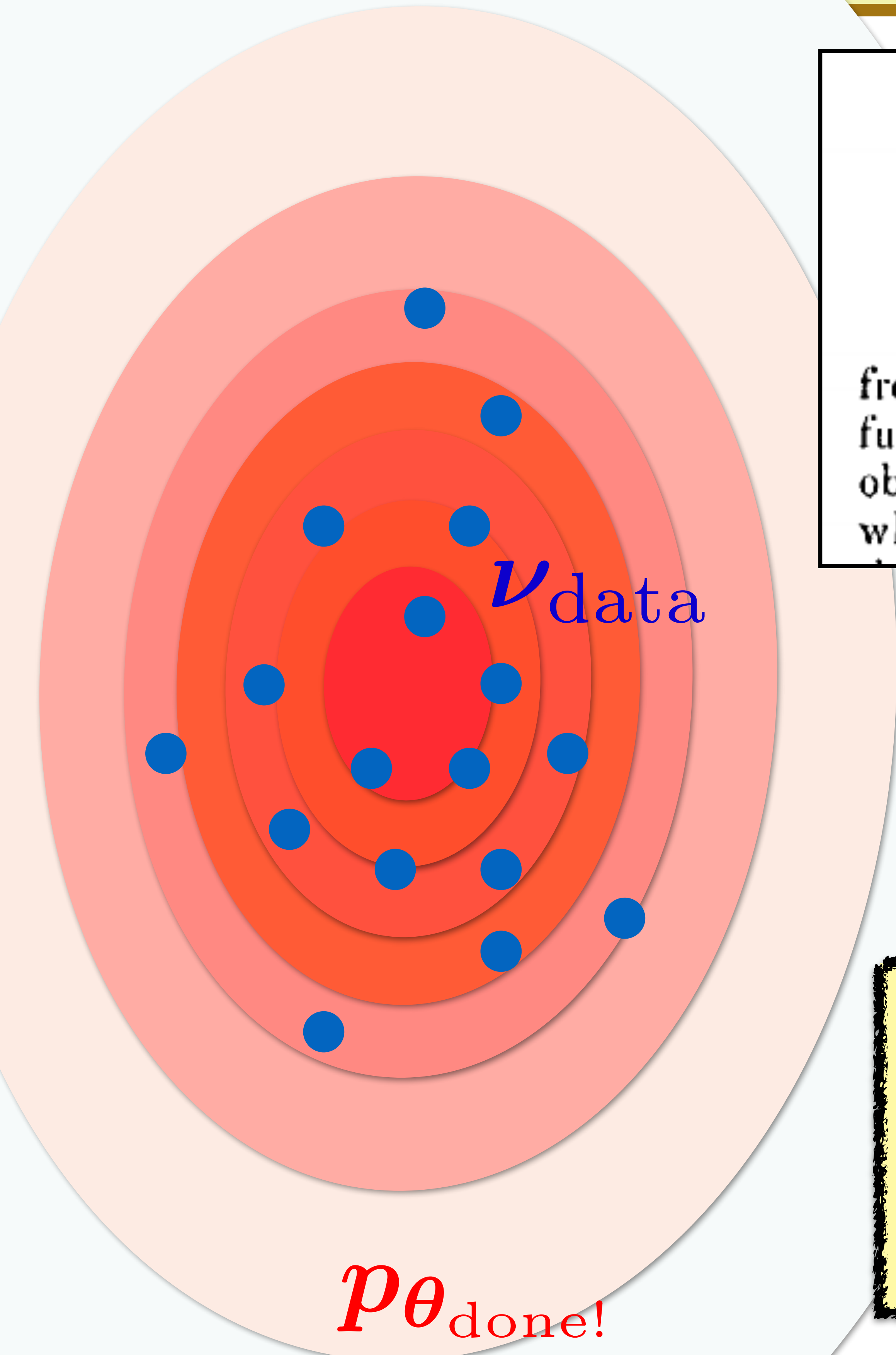
ON AN ABSOLUTE CRITERION
FOR FITTING FREQUENCY CURVES.

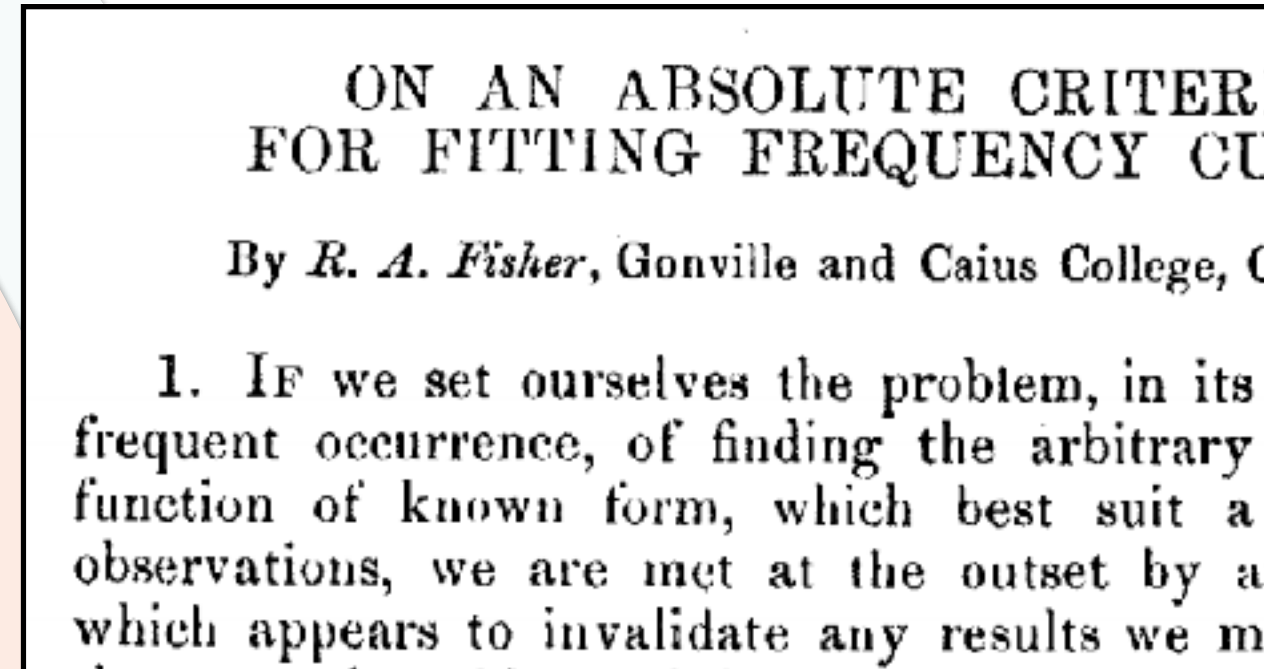
By *R. A. Fisher*, Gonville and Caius College, Cambridge.

1. If we set ourselves the problem, in its
frequent occurrence, of finding the arbitrary
function of known form, which best suit a
observations, we are met at the outset by an
which appears to invalidate any results we ma

$$\max_{\theta \in \Theta} \frac{1}{N} \sum_{i=1}^{N} \log p_{\theta}(x_i)$$

Maximum Likelihood Estimation





ON AN ABSOLUTE CRITERION
FOR FITTING FREQUENCY CURVES.

By *R. A. Fisher*, Gonville and Caius College, Cambridge.

1. If we set ourselves the problem, in its
frequent occurrence, of finding the arbitrary
function of known form, which best suit a
observations, we are met at the outset by an
which appears to invalidate any results we ma

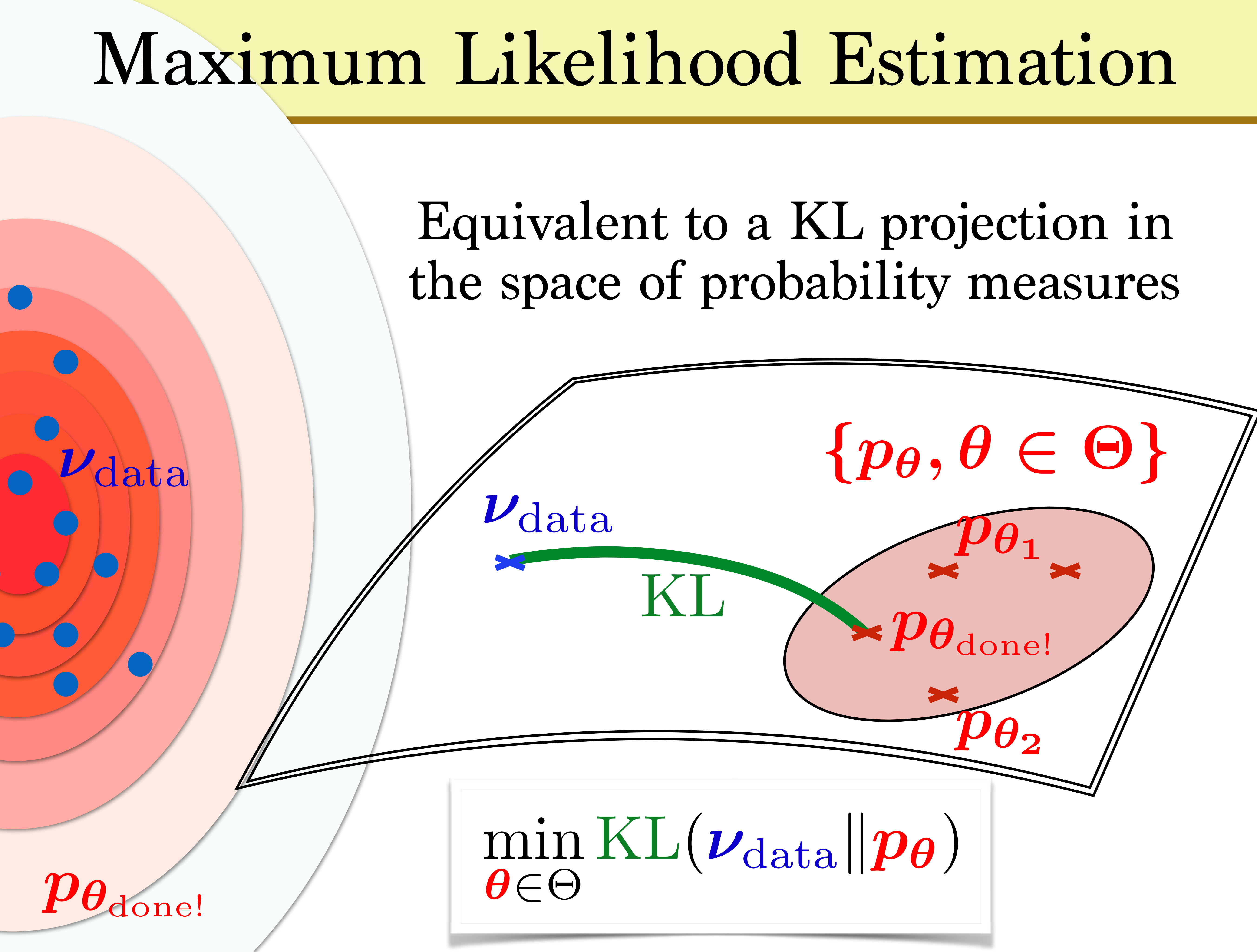
$$\max_{\theta \in \Theta} \frac{1}{N} \sum_{i=1}^{N} \log p_{\theta}(x_i)$$

$$\log 0 = -\infty$$

$p_{\theta}(x_i)$ **must** be > 0

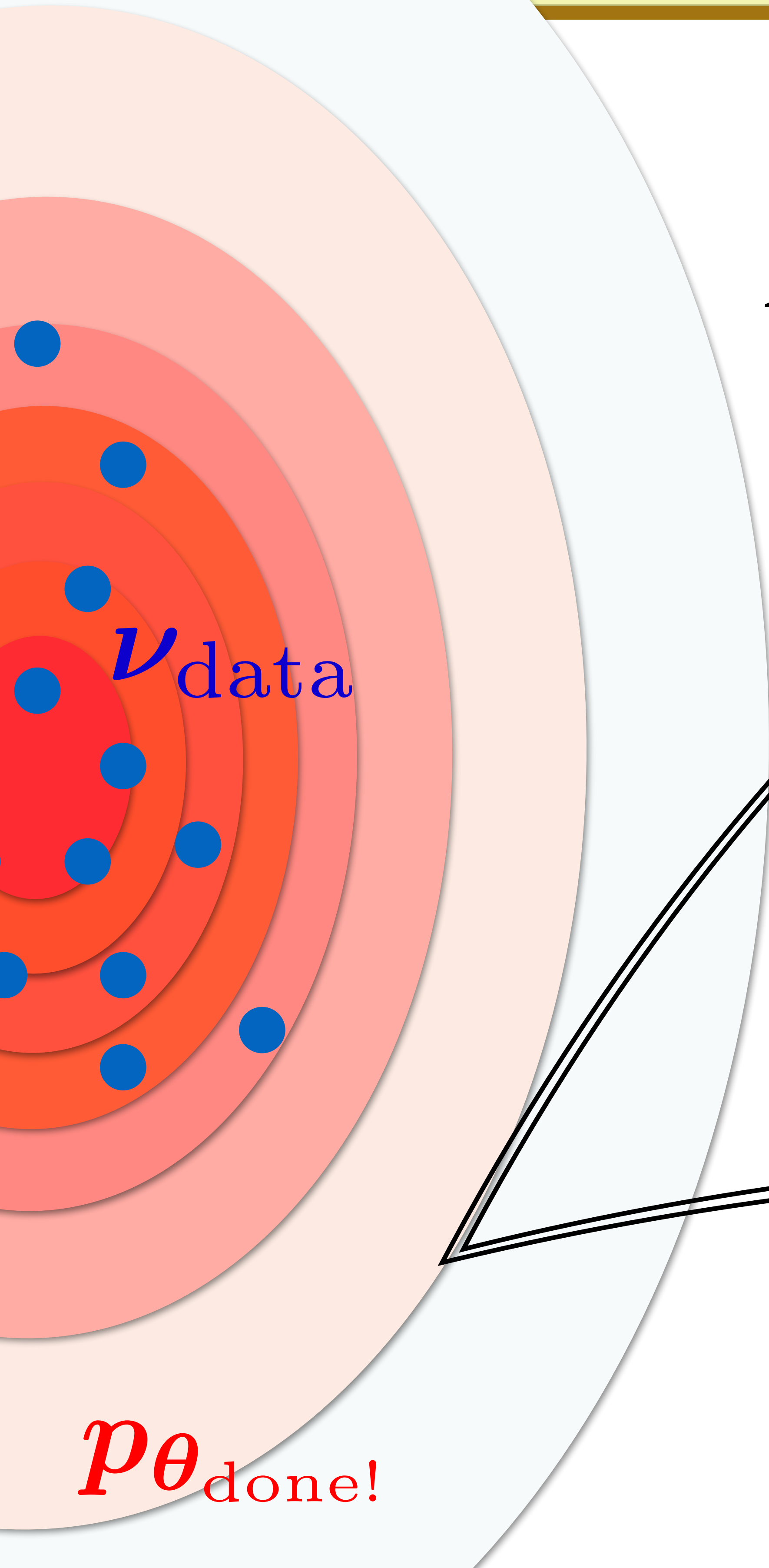
Maximum Likelihood Estimation

Equivalent to a KL projection in the space of probability measures

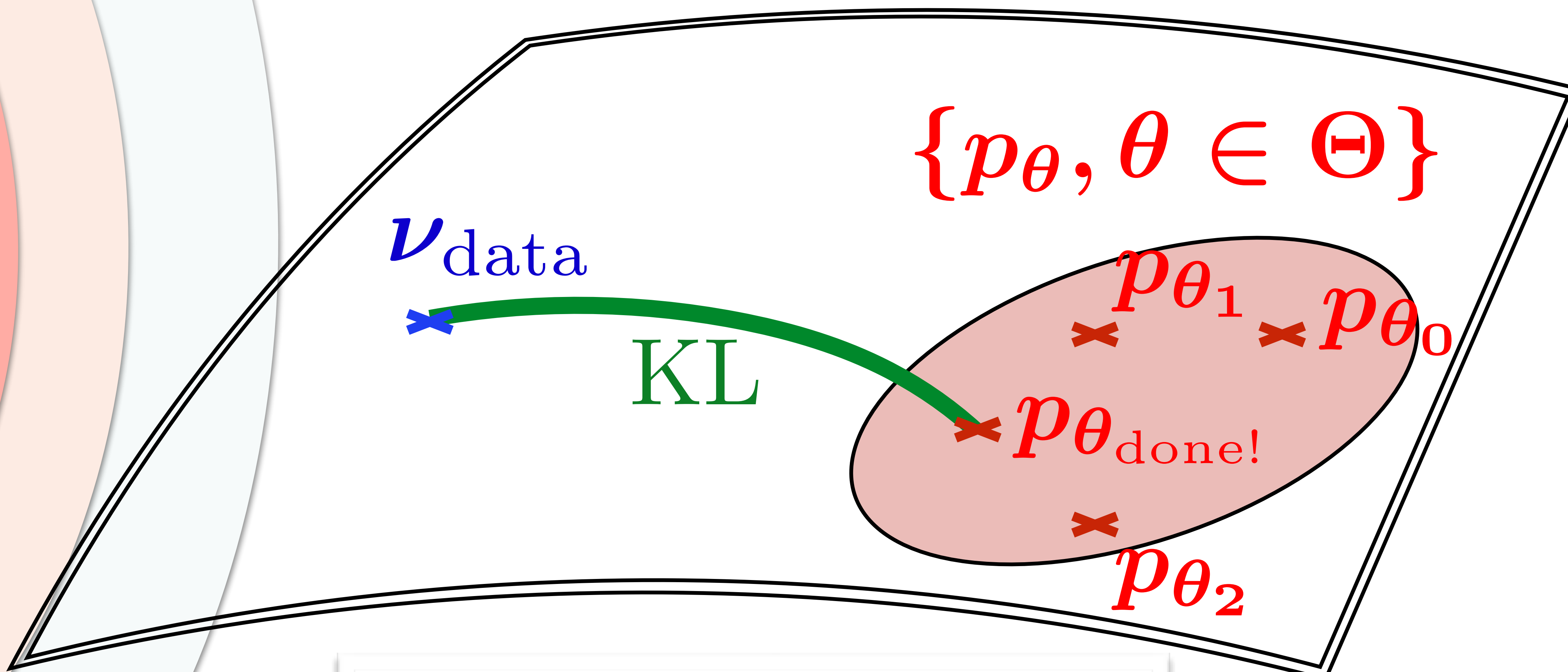


$$\min_{\boldsymbol{\theta} \in \Theta} \mathrm{KL}(\boldsymbol{\nu}_{\text{data}} \| p_{\boldsymbol{\theta}})$$

Maximum Likelihood Estimation

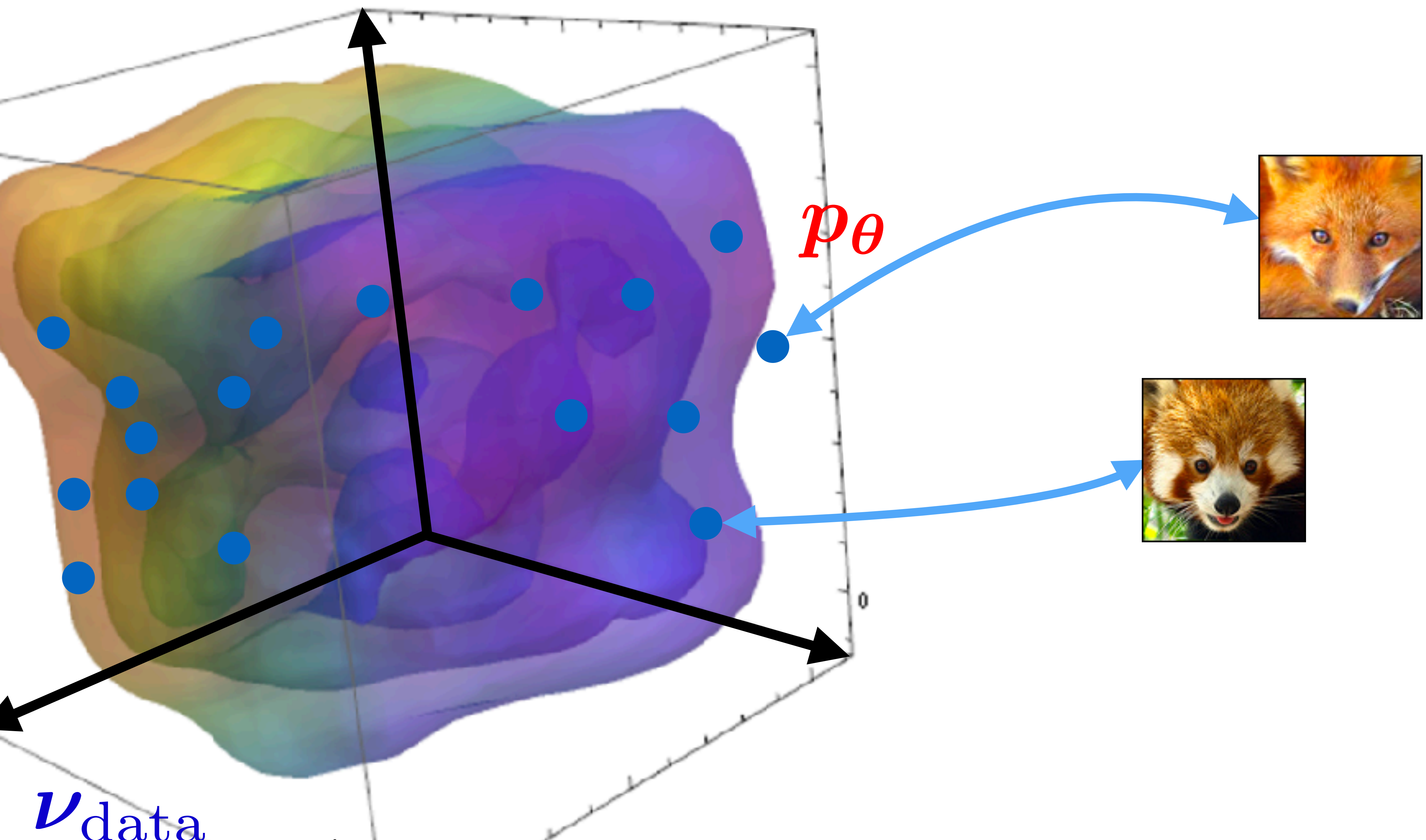


Equivalent to a KL projection in the space of probability measures



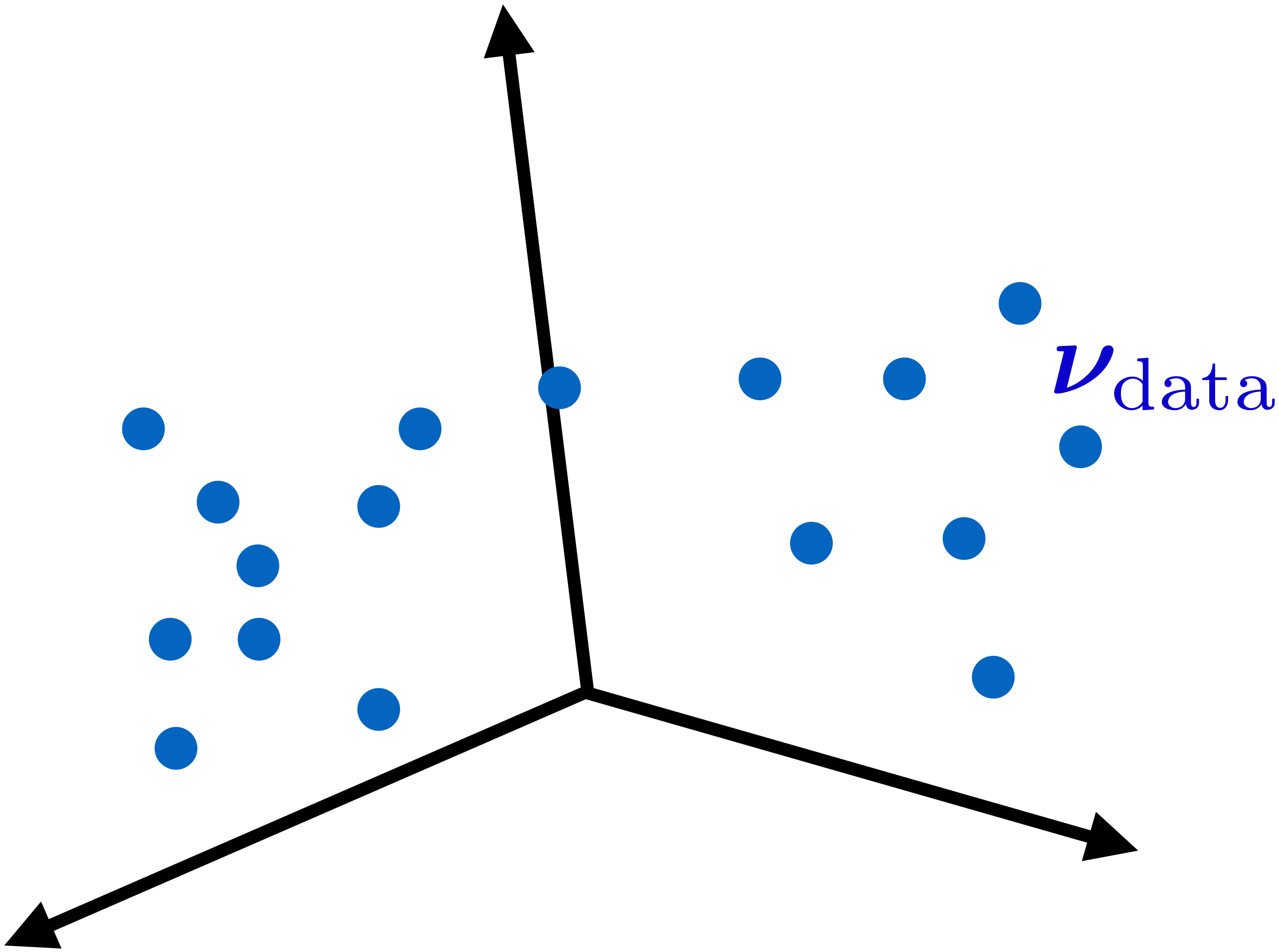
$$\min_{\boldsymbol{\theta} \in \Theta} \mathrm{KL}(\boldsymbol{\nu}_{\mathrm{data}} \| p_{\boldsymbol{\theta}})$$

In higher dimensional spaces...

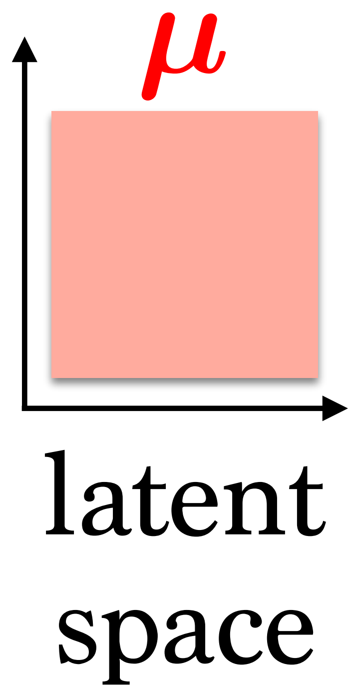


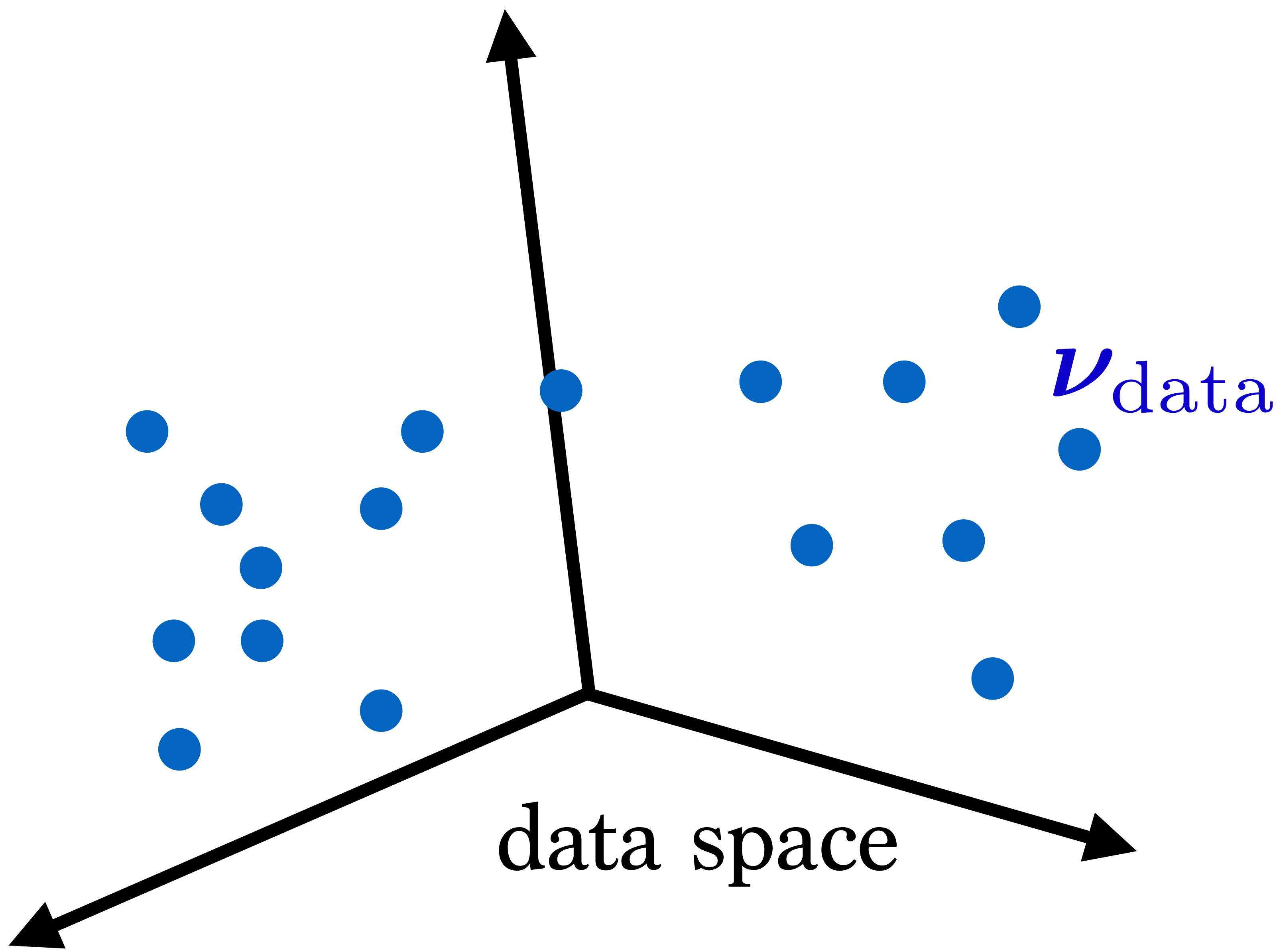
Ambient space: hypercube $d = 30.000$

Generative Models

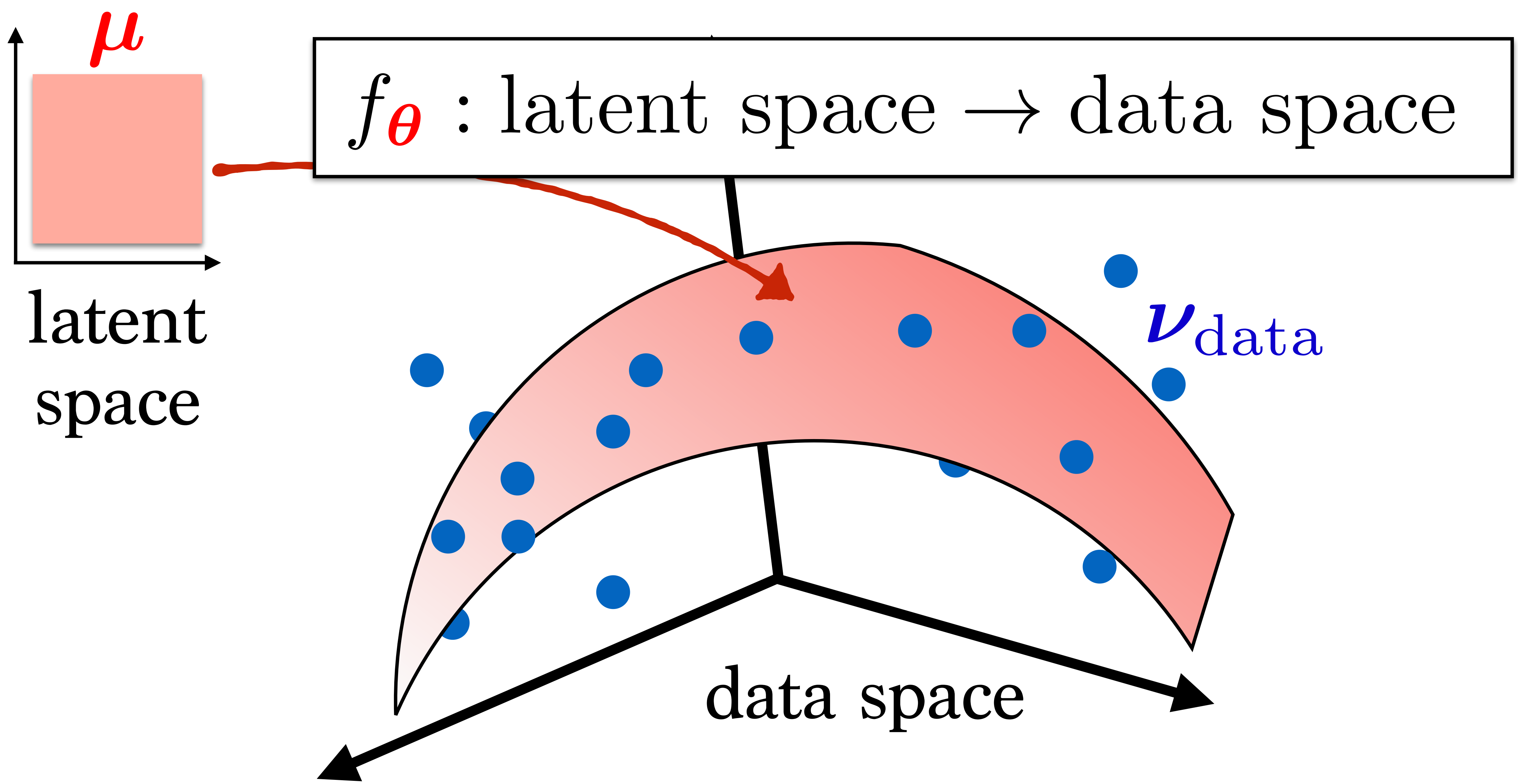


Generative Models

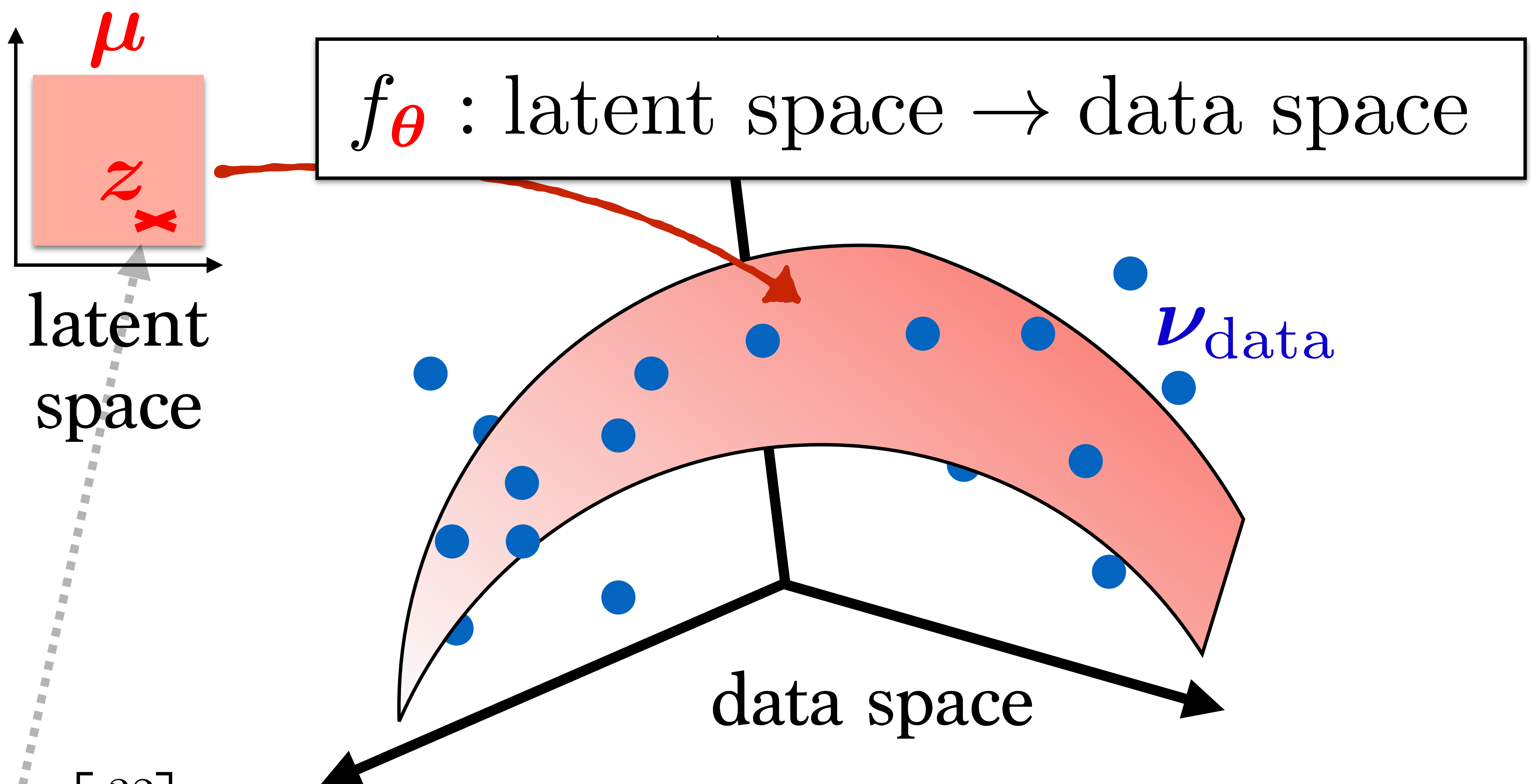




Generative Models

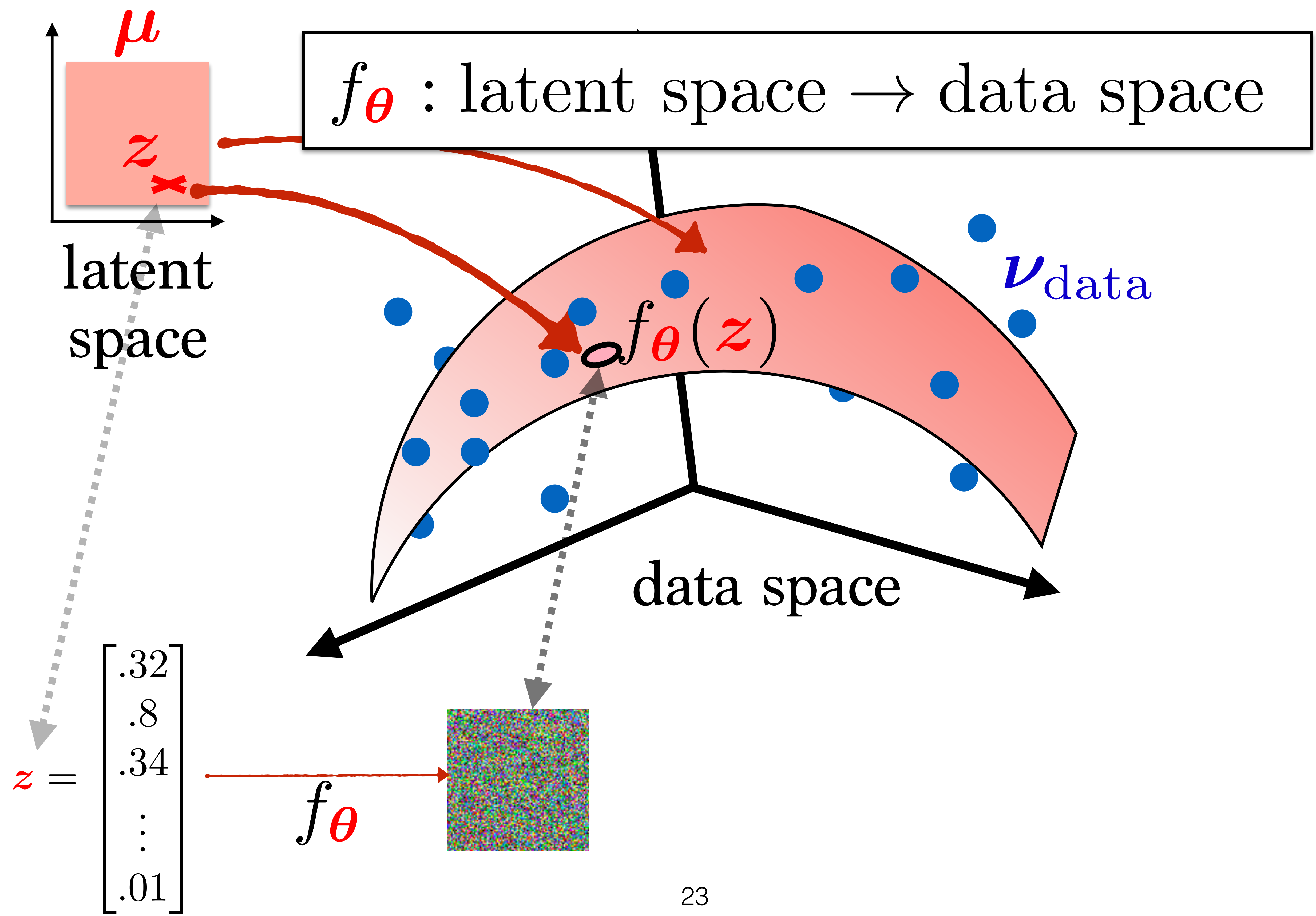


Generative Models

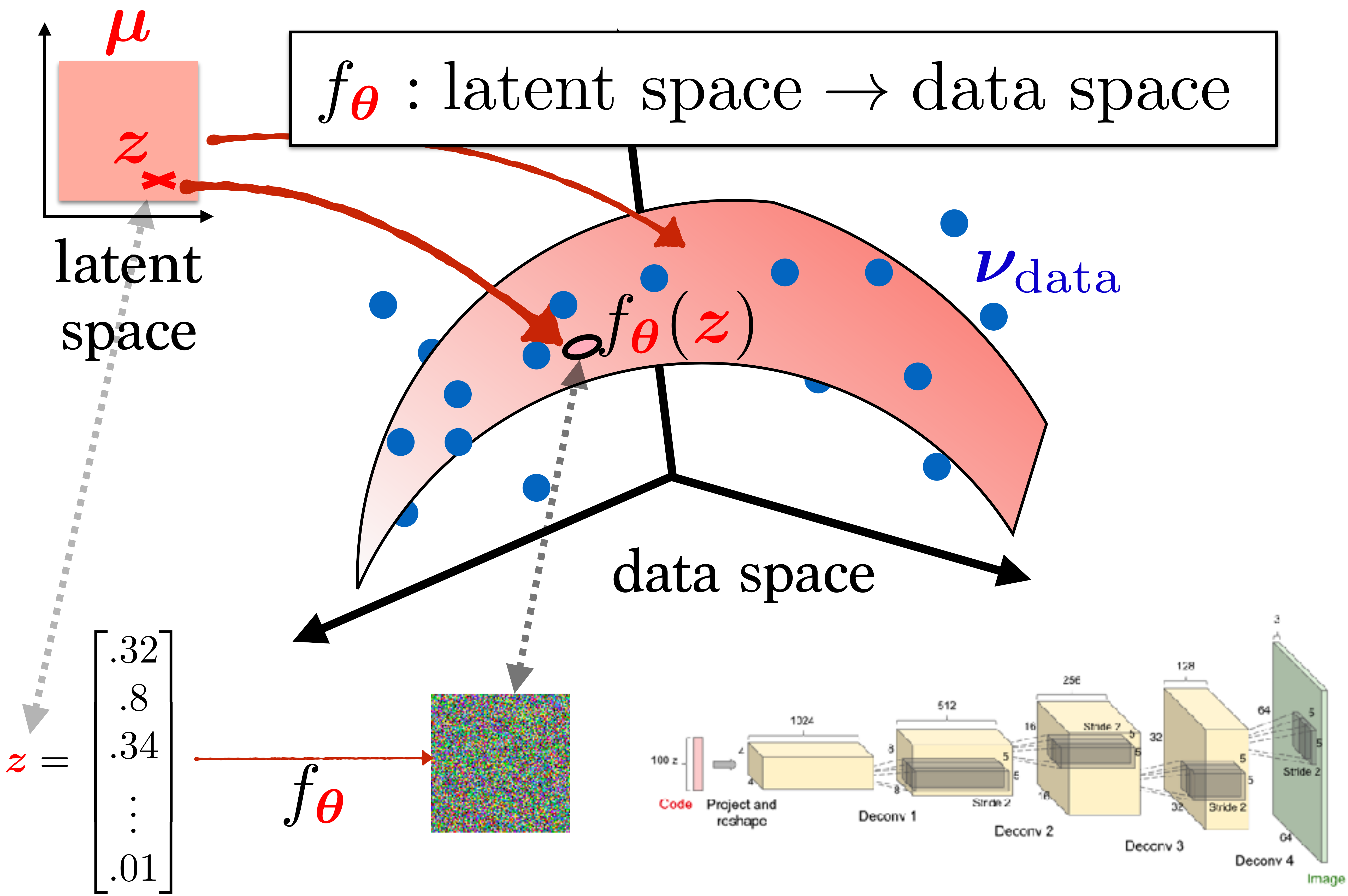


$$z = \begin{bmatrix} .32 \\ .8 \\ .34 \\ \vdots \\ .01 \end{bmatrix}$$

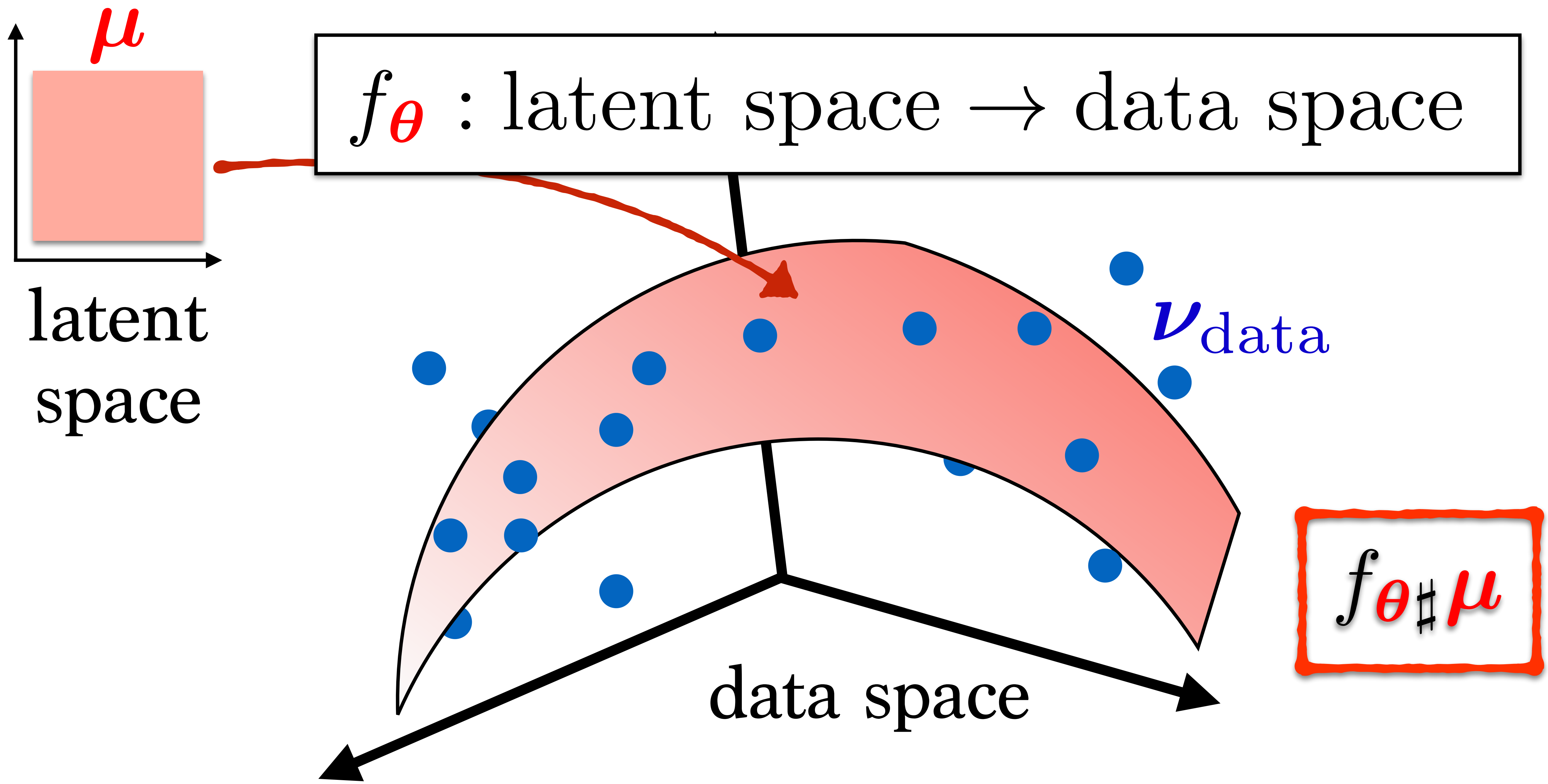
Generative Models



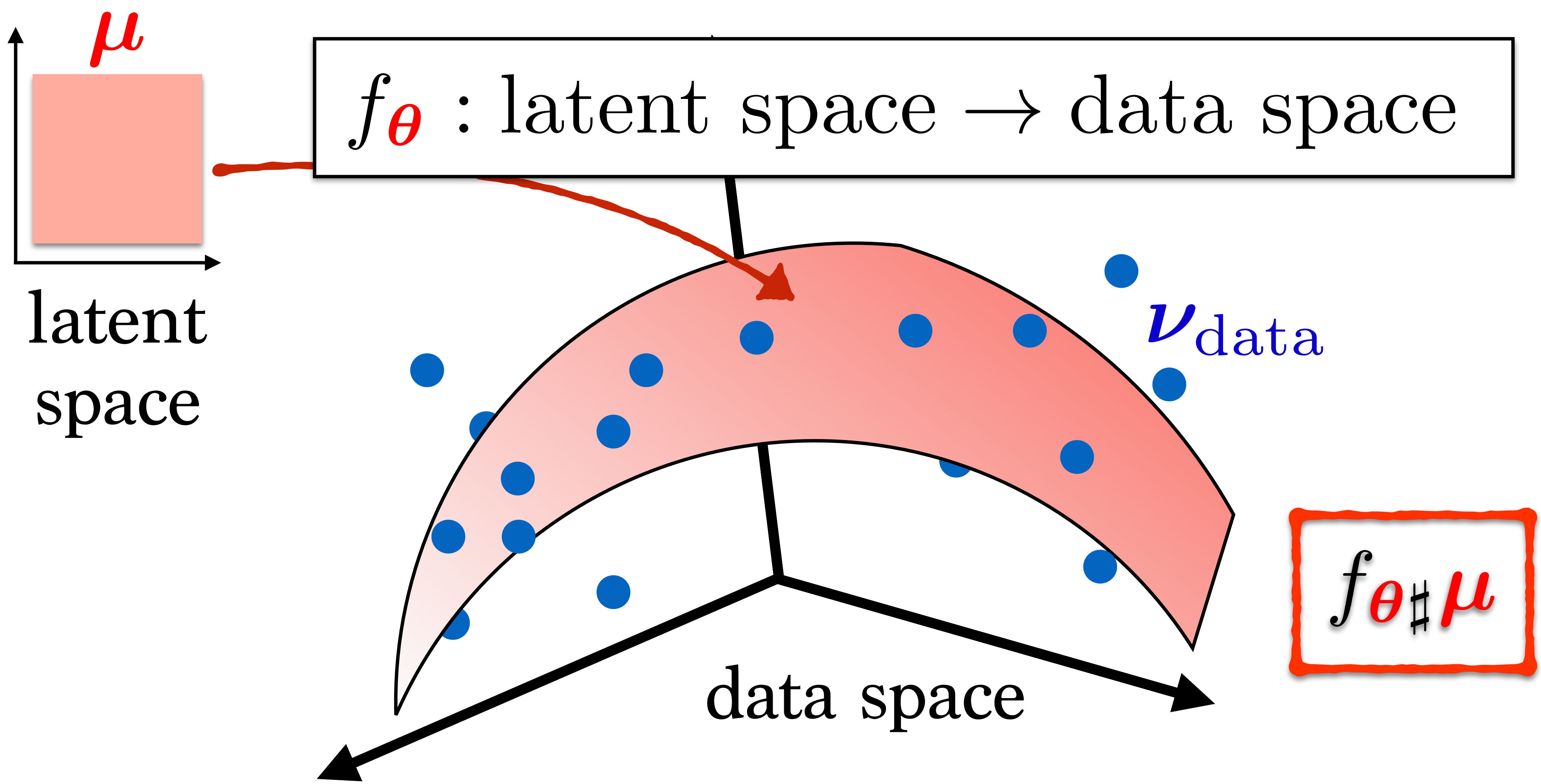
Generative Models



Generative Models

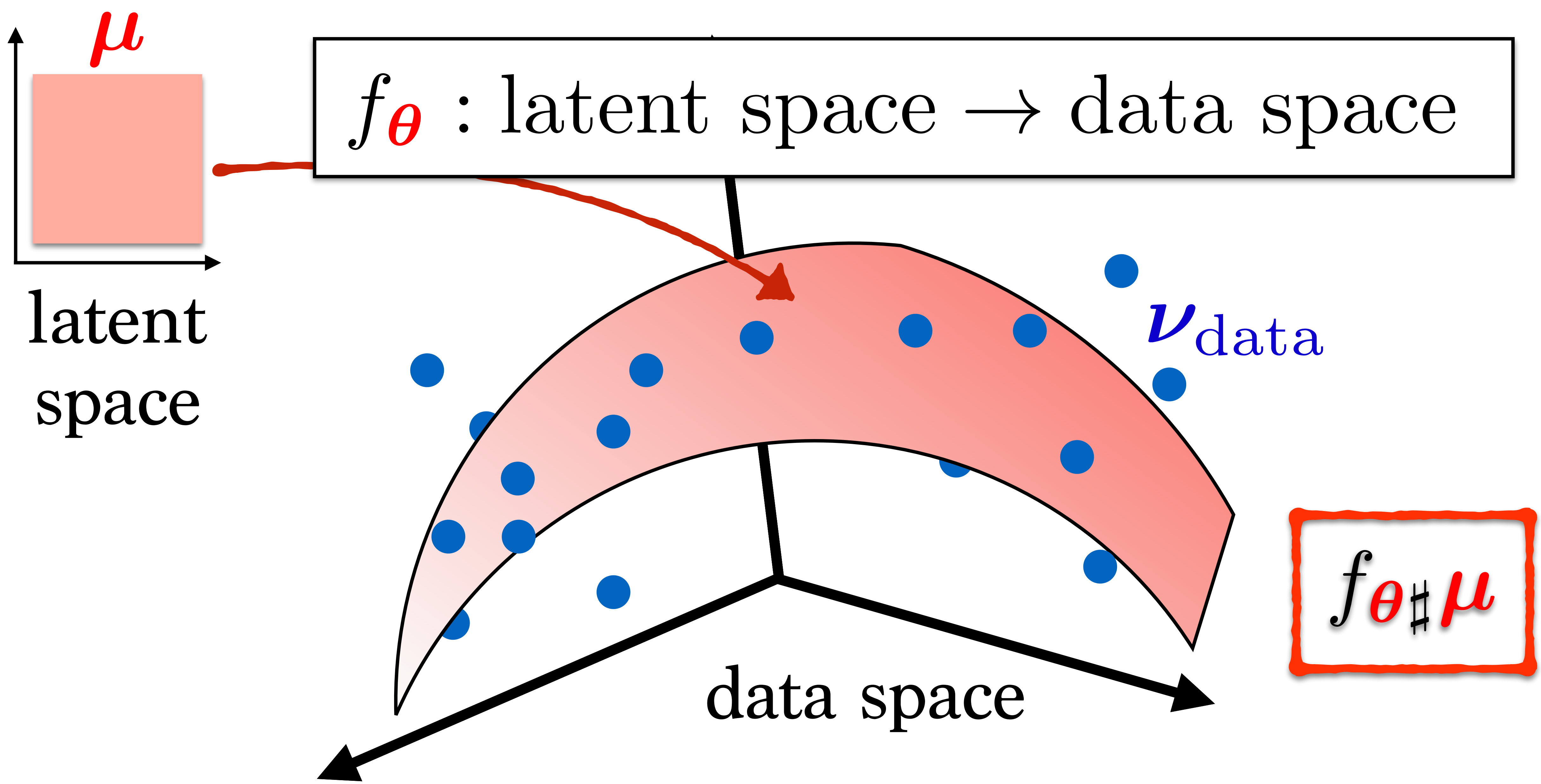


Generative Models



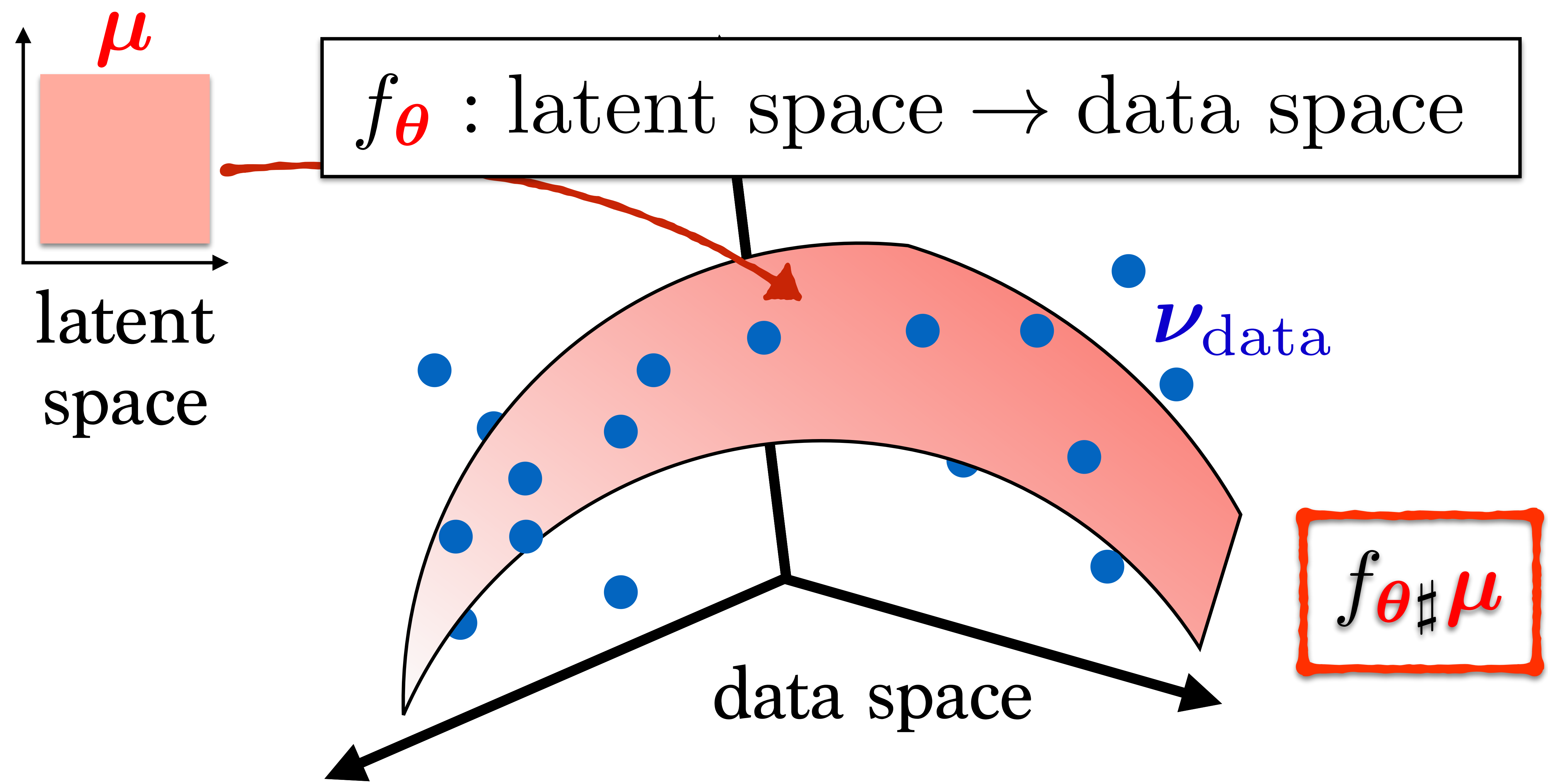
Goal: find θ such that $f_{\theta\sharp}\mu$ fits ν_{data}

Generative Models



Goal: find $\boldsymbol{\theta}$ such that $f_{\boldsymbol{\theta}\sharp}\boldsymbol{\mu}$ fits $\boldsymbol{\nu}_{\text{data}}$

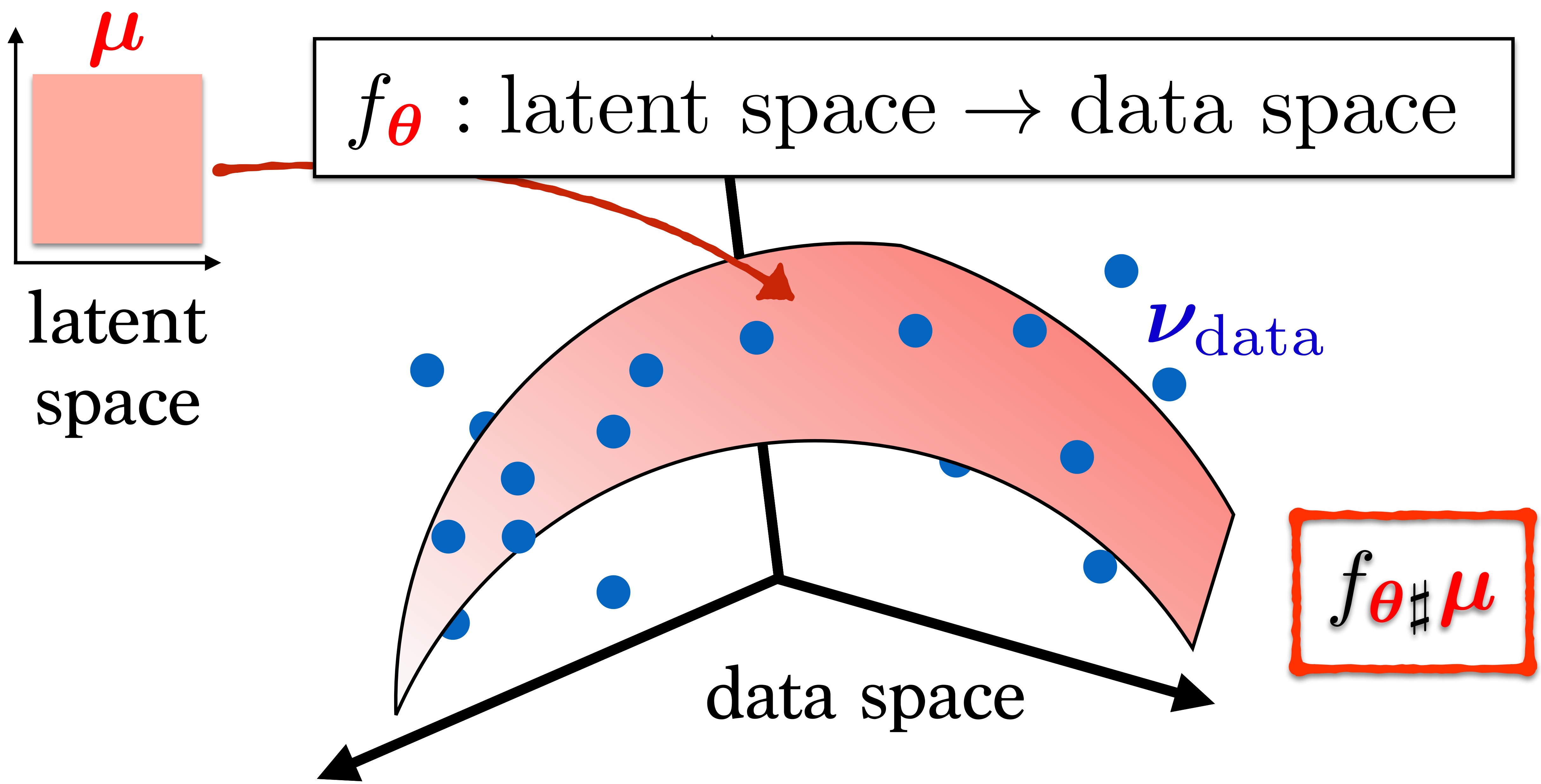
Generative Models



MLE

$$\max_{\boldsymbol{\theta}\in\Theta} \frac{1}{N}\sum_{i=1}^{N} \log \boldsymbol{p}_{\boldsymbol{\theta}}(\boldsymbol{x}_i) \quad = \quad \min_{\boldsymbol{\theta}\in\Theta} \mathrm{KL}(\boldsymbol{\nu}_{\text{data}} \| \boldsymbol{p}_{\boldsymbol{\theta}})$$

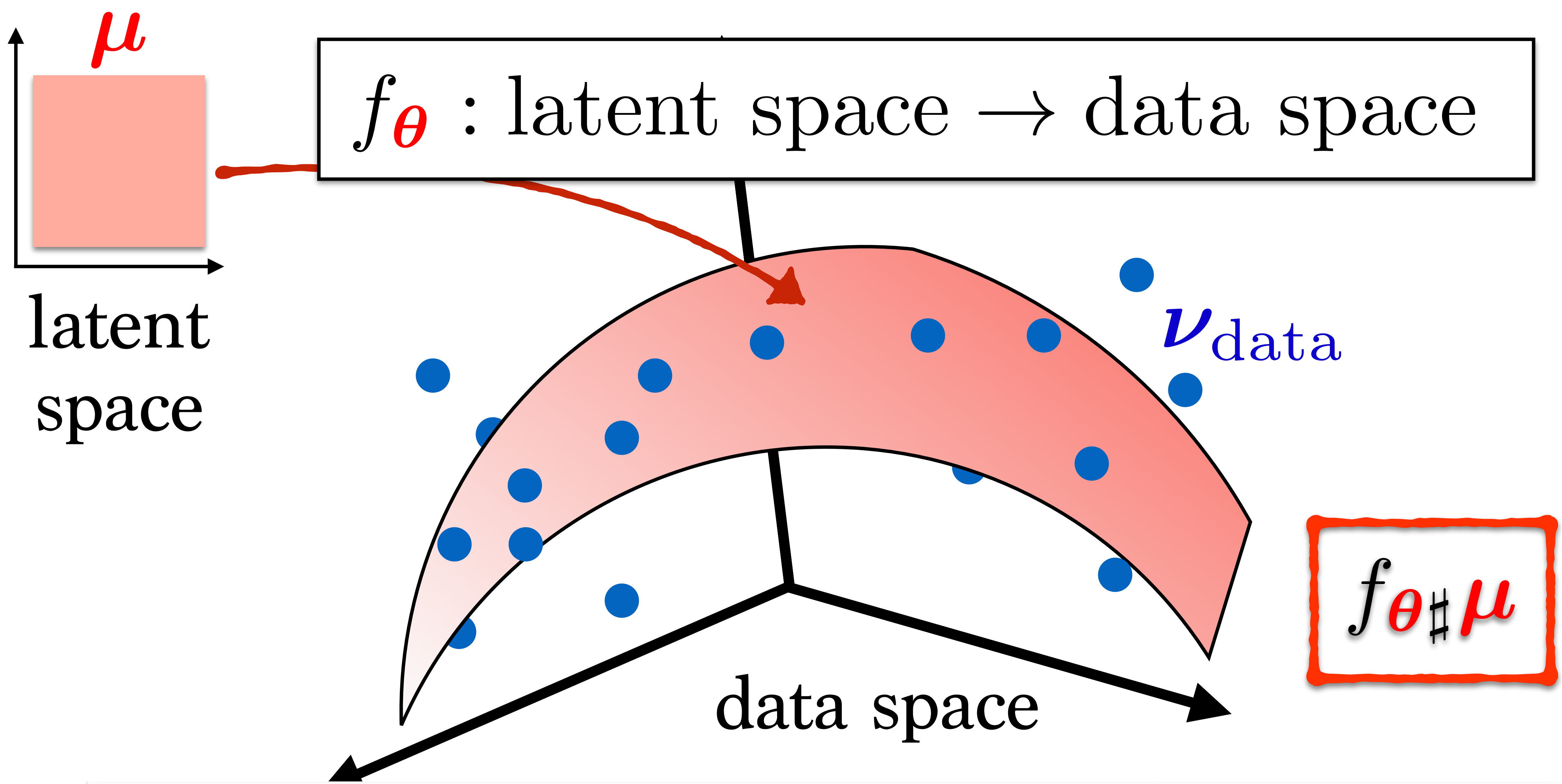
Generative Models



~~MLE~~

$$\max_{\theta \in \Theta} \frac{1}{N} \sum_{i=1}^{N} \log f_{\theta\sharp}\mu(x_i) \qquad \min_{\theta\in\Theta} \mathrm{KL}(\nu_{\text{data}} \| f_{\theta\sharp}\mu)$$

Generative Models

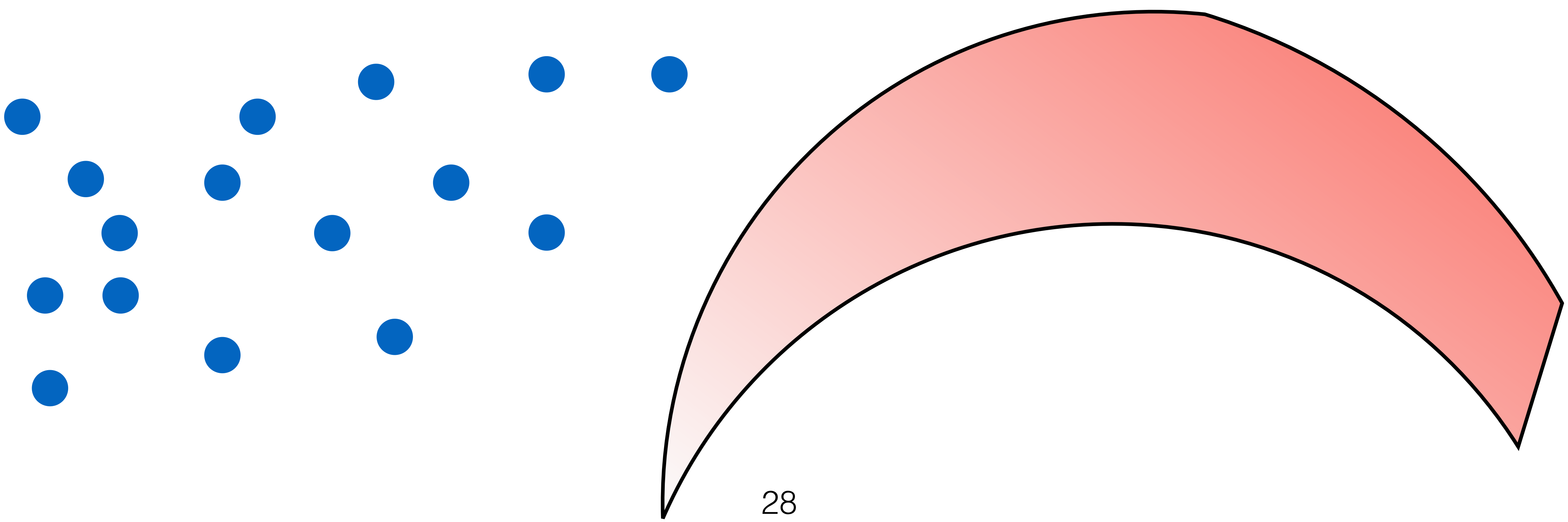


Need a more flexible **discrepancy function** to compare ν_{data} and $f_{\theta\sharp}\mu$

Workarounds?

- Formulation as **adversarial problem** **[GPM…'14]**

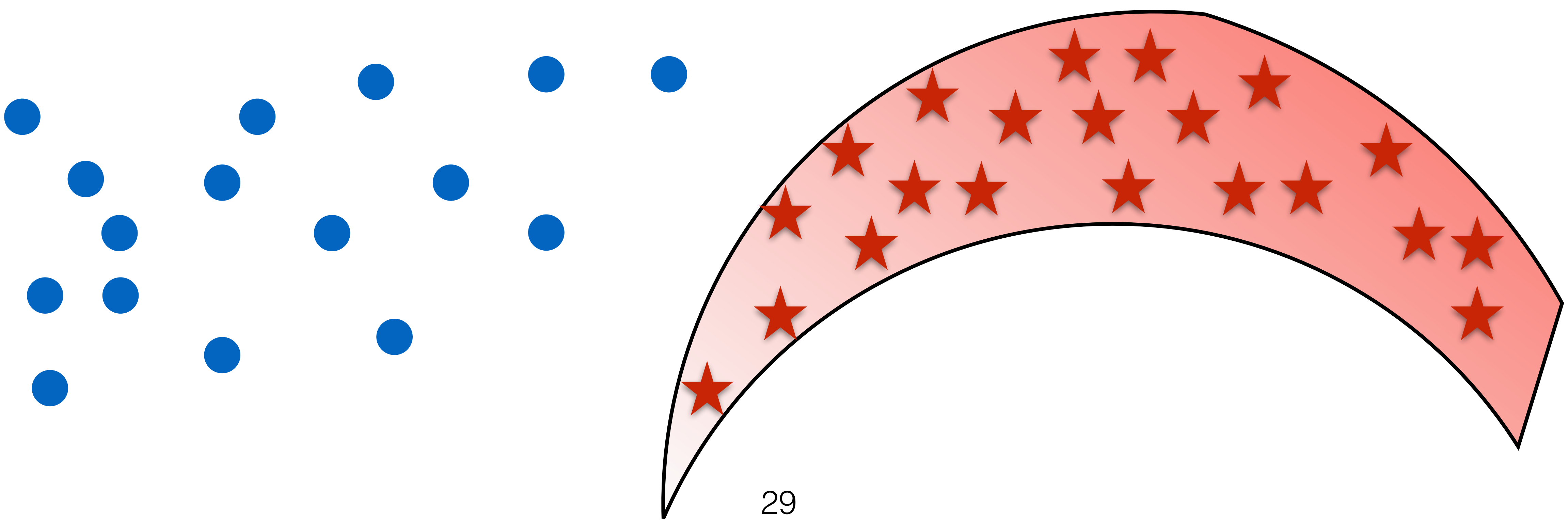
$$\min_{\boldsymbol{\theta}\in\Theta} \max_{\text{classifiers } \boldsymbol{g}} \text{Accuracy}_{\boldsymbol{g}}\left(\left(f_{\boldsymbol{\theta}\sharp}\boldsymbol{\mu}, +1\right), \left(\boldsymbol{\nu}_{\text{data}}, -1\right)\right)$$



Workarounds?

- Formulation as **adversarial problem** **[GPM…'14]**

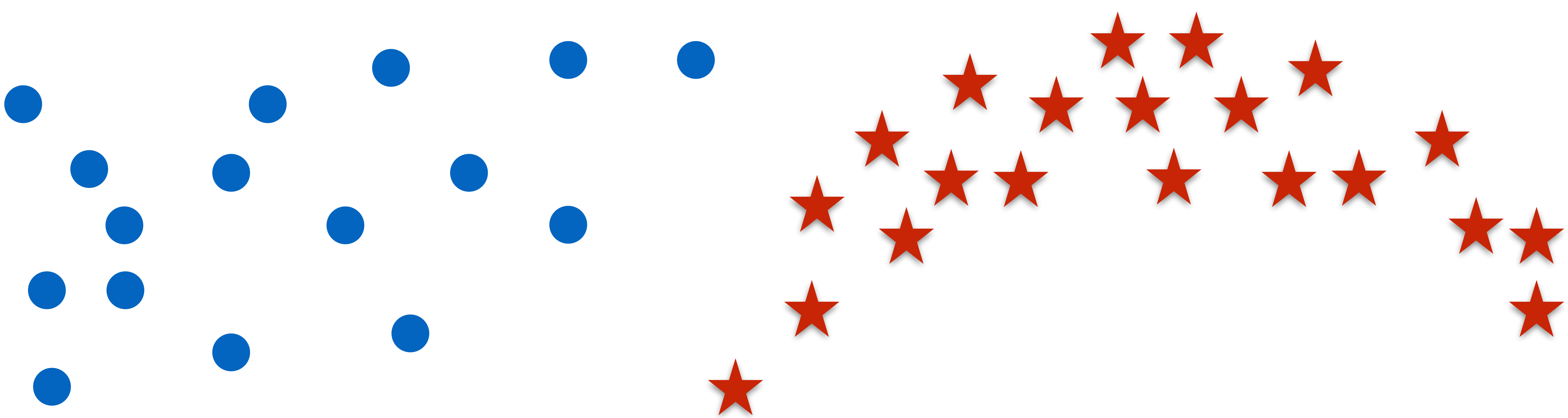
$$\min_{\boldsymbol{\theta}\in\Theta} \max_{\text{classifiers } \boldsymbol{g}} \text{Accuracy}_{\boldsymbol{g}}\left(\left(\boldsymbol{f}_{\boldsymbol{\theta}\sharp}\boldsymbol{\mu}, +1\right), \left(\boldsymbol{\nu}_{\text{data}}, -1\right)\right)$$



Workarounds?

- Formulation as **adversarial problem** **[GPM…'14]**

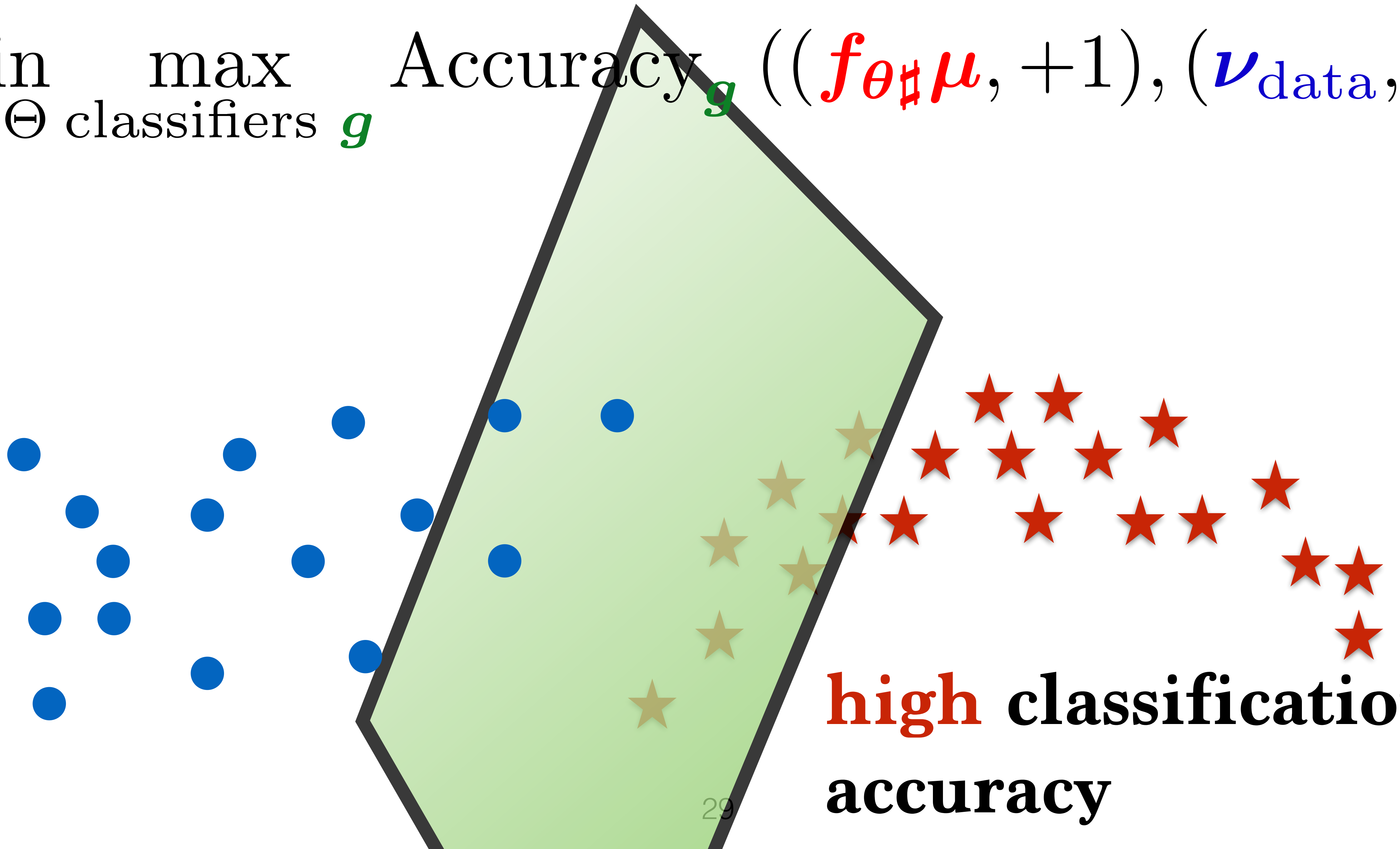
$$\min_{\boldsymbol{\theta}\in\Theta} \max_{\text{classifiers } \boldsymbol{g}} \text{Accuracy}_{\boldsymbol{g}}\left(\left(f_{\boldsymbol{\theta}\sharp}\boldsymbol{\mu}, +1\right), \left(\boldsymbol{\nu}_{\text{data}}, -1\right)\right)$$



Workarounds?

- Formulation as **adversarial problem** **[GPM…'14]**

$$\min_{\boldsymbol{\theta}\in\Theta} \max_{\text{classifiers } \boldsymbol{g}} \text{Accuracy}_{\boldsymbol{g}}\left(\left(\boldsymbol{f_{\theta\sharp}\mu}, +1\right), \left(\boldsymbol{\nu}_{\text{data}}, -1\right)\right)$$

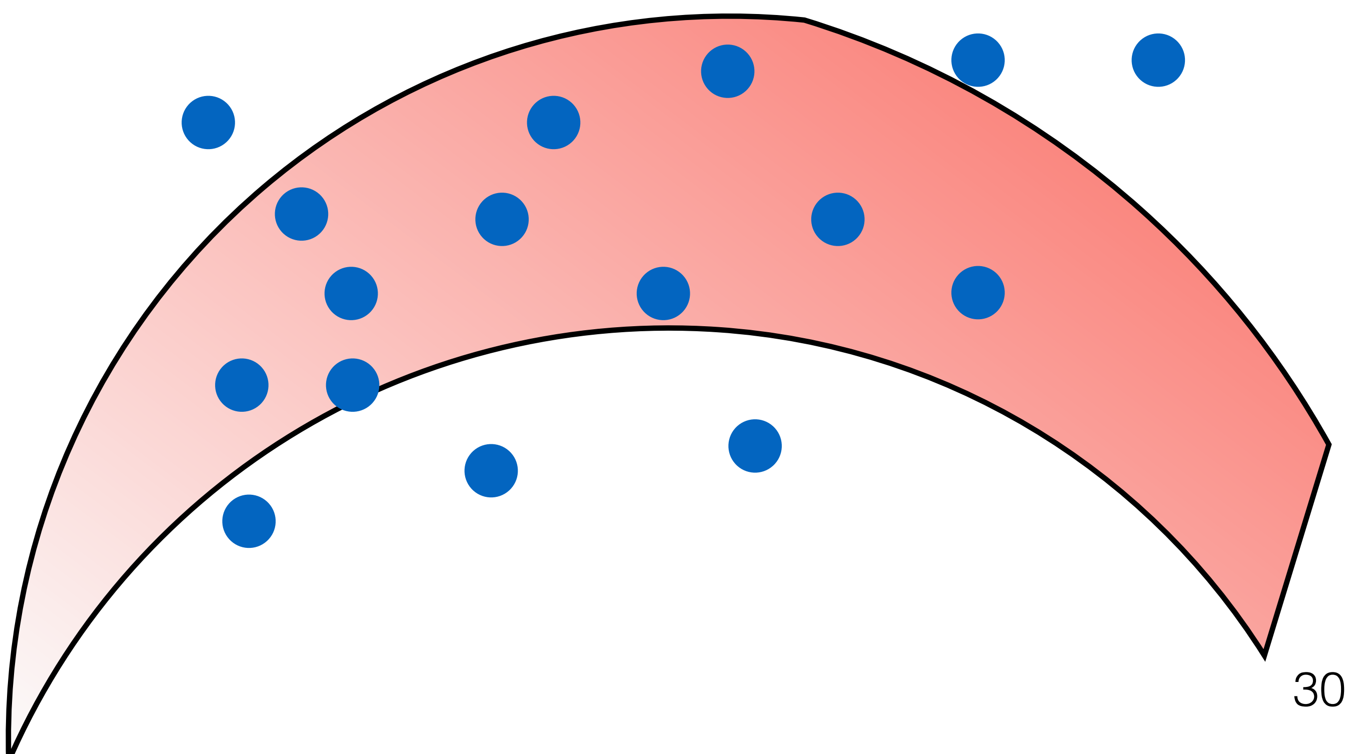


high classification accuracy

Workarounds?

- Formulation as **adversarial problem** **[GPM…'14]**

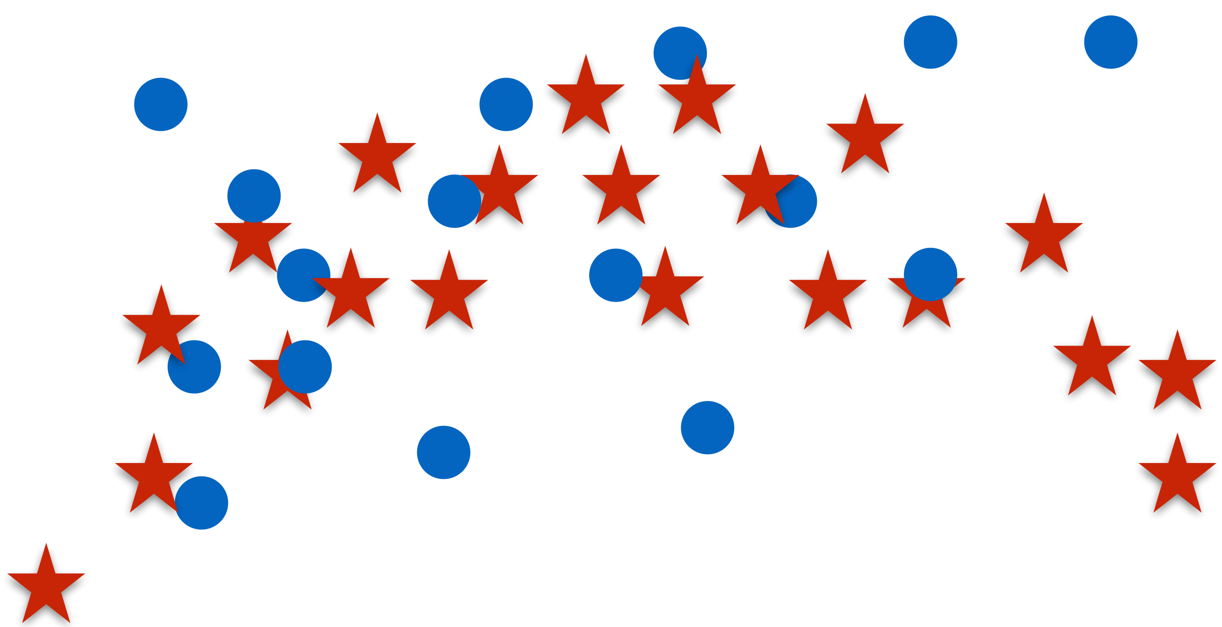
$$\min_{\theta\in\Theta}\ \max_{\text{classifiers } g}\ \text{Accuracy}_g\left(\left(f_{\theta\sharp}\mu, +1\right), \left(\nu_{\text{data}}, -1\right)\right)$$



Workarounds?

- Formulation as **adversarial problem** **[GPM…'14]**

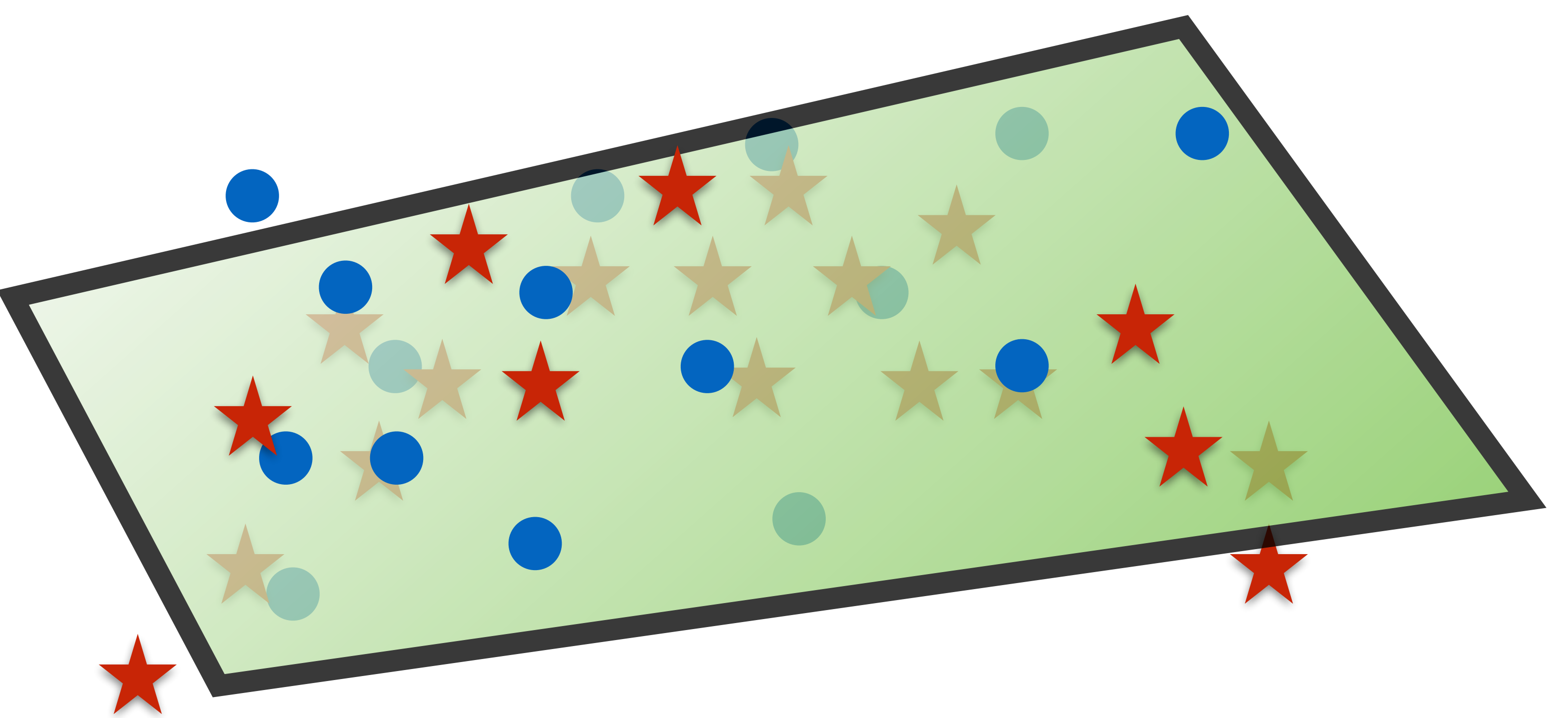
$$\min_{\theta \in \Theta} \max_{\text{classifiers } g} \text{Accuracy}_g\left(\left(f_{\theta\sharp}\mu, +1\right), \left(\nu_{\text{data}}, -1\right)\right)$$



Workarounds?

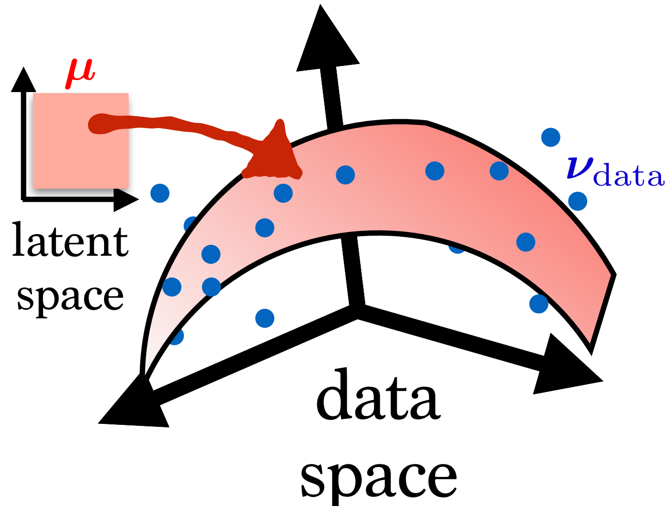
- Formulation as **adversarial problem** **[GPM…'14]**

$$\min_{\theta\in\Theta}\ \max_{\text{classifiers } g}\ \text{Accuracy}_g\left(\left(f_{\theta\sharp}\mu, +1\right), \left(\nu_{\text{data}}, -1\right)\right)$$



low classification accuracy…
is the goal.

Another idea?



- Use a **metric** Δ for probability measures, that can handle measures with non-overlapping supports:

$$\min_{\boldsymbol{\theta}\in\Theta} \Delta(\boldsymbol{\nu}_{\text{data}}, \boldsymbol{p}_{\boldsymbol{\theta}}), \qquad \textbf{not} \min_{\boldsymbol{\theta}\in\Theta} \mathrm{KL}(\boldsymbol{\nu}_{\text{data}} \| \boldsymbol{p}_{\boldsymbol{\theta}})$$

- The original GAN paper can be interpreted in that light using the Jensen-Shannon divergence.

Minimum △ Estimation

The Annals of Statistics
1980, Vol. 8, No. 3, 457–487

MINIMUM CHI-SQUARE, NOT MAXIMUM LIKELIHOOD!

BY JOSEPH BERKSON
Mayo Clinic, Rochester, Minnesota

ELSEVIER Computational Statistics & Data Analysis 29 (1998) 81–103
COMPUTATIONAL STATISTICS & DATA ANALYSIS

Springer Series in Statistics
Luc Devroye
Gábor Lugosi
Combinatorial Methods in Density Estimation

l_1

Minimum Hellinger distance estimation for Poisson mixtures

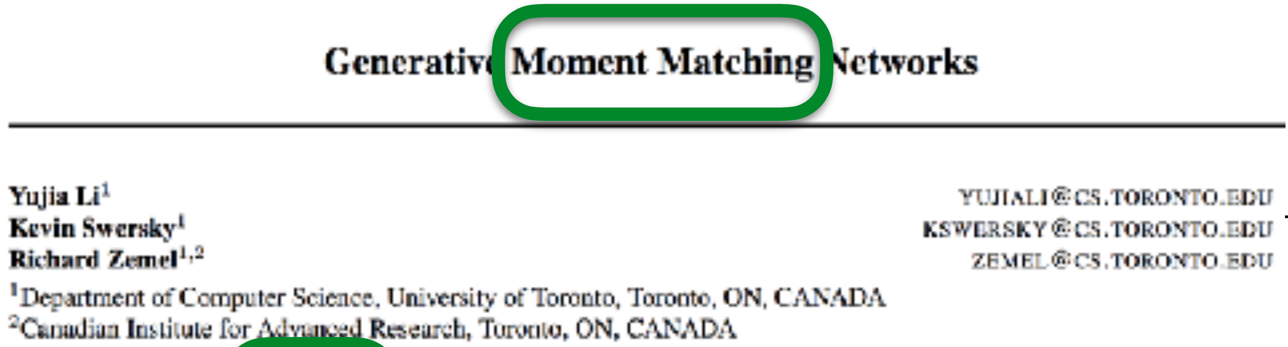
Dimitris Karlis, Evdokia Xekalaki*
Department of Statistics, Athens University of Economics and Business, 76 Patission Str., 104 34 Athens, Greece

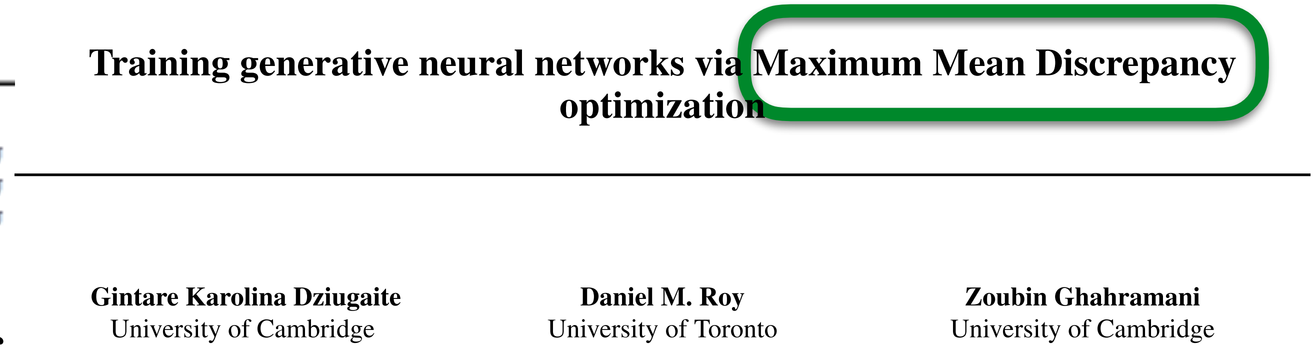
ELSEVIER
Available online at www.sciencedirect.com
SCIENCE DIRECT
Statistics & Probability Letters 76 (2006) 1298–1302
STATISTICS & PROBABILITY LETTERS
www.elsevier.com/locate/stapro

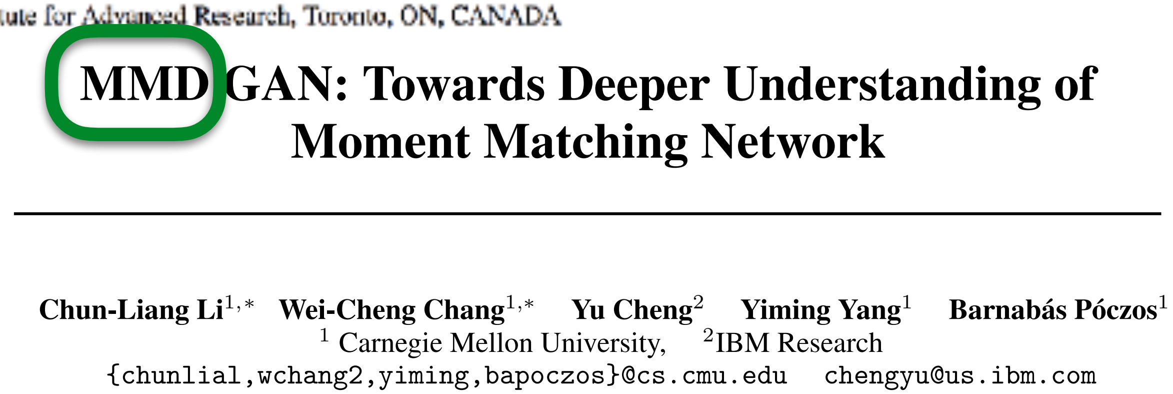
On minimum Kantorovich distance estimators

Federico Bassetti[a], Antonella Bodini[b], Eugenio Regazzini[a,*]

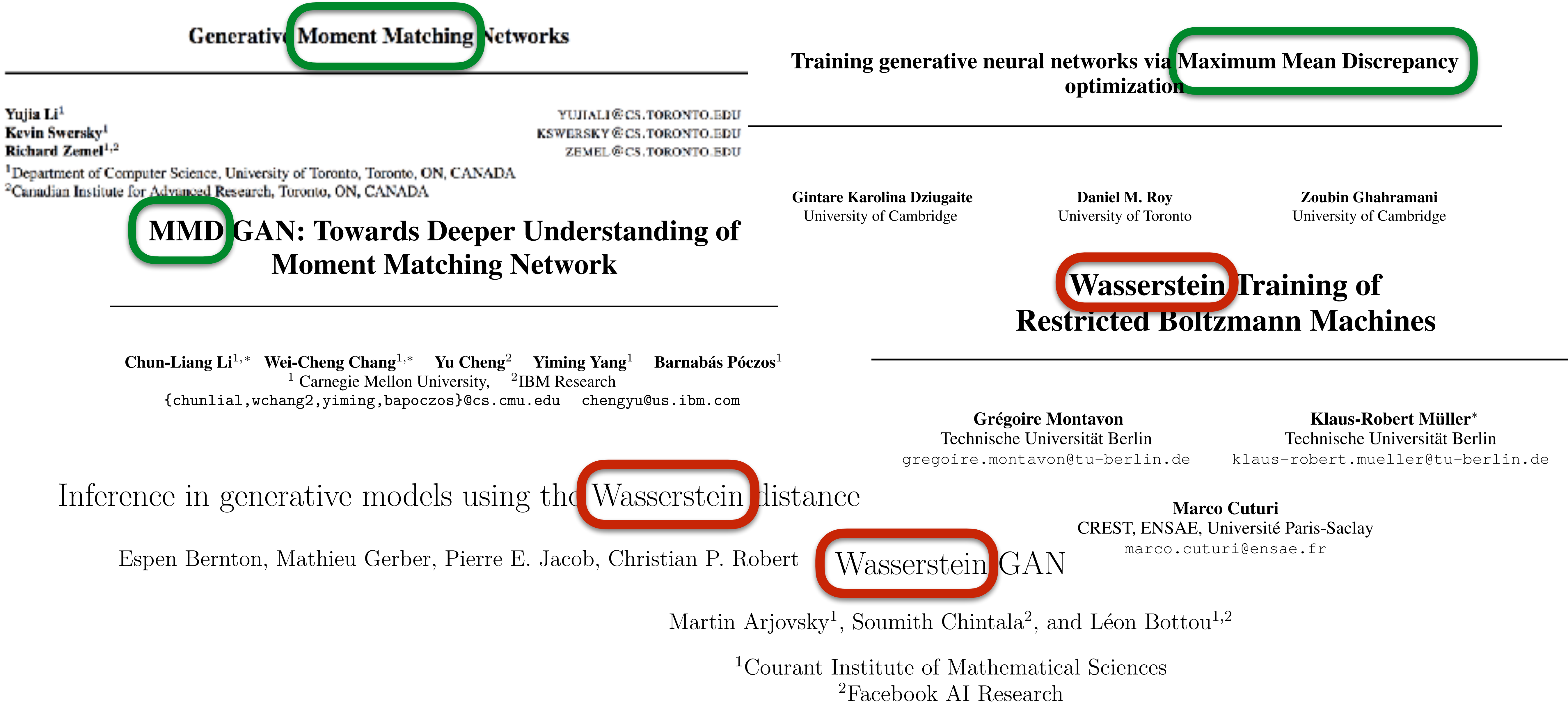
△ Generative Model Estimation







△ Generative Model Estimation



△ Generative Model Estimation

Generative Moment Matching Networks

Yujia Li[1] YUJIALI@CS.TORONTO.EDU
Kevin Swersky[1] KSWERSKY@CS.TORONTO.EDU
Richard Zemel[1,2] ZEMEL@CS.TORONTO.EDU
[1]Department of Computer Science, University of Toronto, Toronto, ON, CANADA
[2]Canadian Institute for Advanced Research, Toronto, ON, CANADA

Training generative neural networks via Maximum Mean Discrepancy optimization

Gintare Karolina Dziugaite, University of Cambridge
Daniel M. Roy, University of Toronto
Zoubin Ghahramani, University of Cambridge

MMD GAN: Towards Deeper Understanding of Moment Matching Network

Chun-Liang Li[1,*] Wei-Cheng Chang[1,*] Yu Cheng[2] Yiming Yang[1] Barnabás Póczos[1]
[1] Carnegie Mellon University, [2]IBM Research
{chunlial,wchang2,yiming,bapoczos}@cs.cmu.edu chengyu@us.ibm.com

Wasserstein Training of Restricted Boltzmann Machines

Grégoire Montavon, Technische Universität Berlin, gregoire.montavon@tu-berlin.de
Klaus-Robert Müller*, Technische Universität Berlin, klaus-robert.mueller@tu-berlin.de
Marco Cuturi, CREST, ENSAE, Université Paris-Saclay, marco.cuturi@ensae.fr

Inference in generative models using the Wasserstein distance

Espen Bernton, Mathieu Gerber, Pierre E. Jacob, Christian P. Robert

Wasserstein GAN

Martin Arjovsky[1], Soumith Chintala[2], and Léon Bottou[1,2]
[1]Courant Institute of Mathematical Sciences
[2]Facebook AI Research

Learning Generative Models with Sinkhorn Divergences

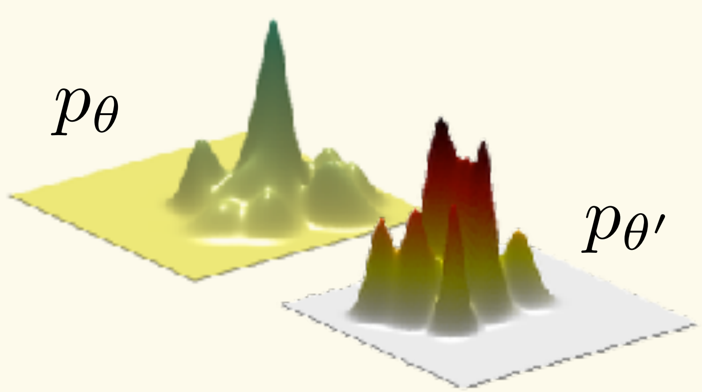
Aude Genevay, CEREMADE, Université Paris-Dauphine
Gabriel Peyré, CNRS and DMA, École Normale Supérieure
Marco Cuturi, ENSAE CREST, Université Paris-Saclay

Improving GANs Using Optimal Transport

Tim Salimans*, OpenAI, tim@openai.com
Han Zhang[†], Rutgers University, han.zhang@cs.rutgers.edu
Alec Radford, OpenAI, alec@openai.com
Dimitris Metaxas, Rutgers University, dnm@cs.rutgers.edu

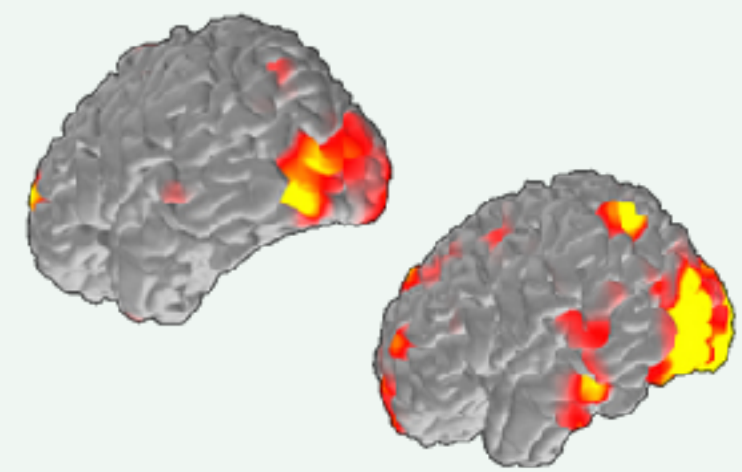
Optimal Transport in ML

OT is establishing itself as a generic toolbox to handle probability measures in ML tasks



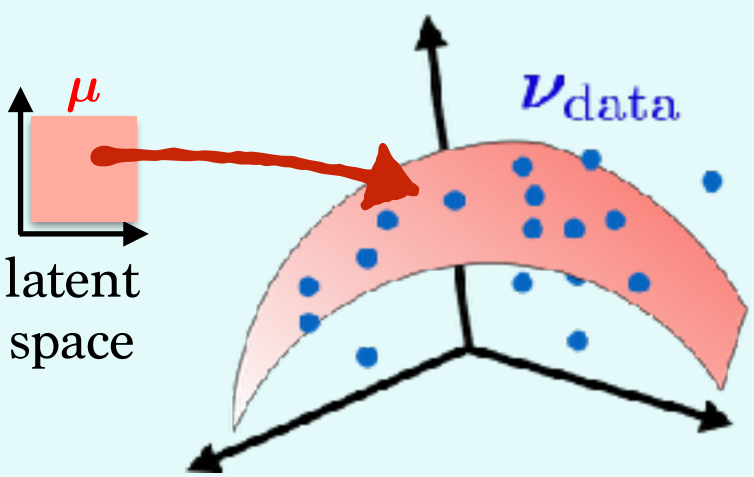
Statistical Models

Bags of features



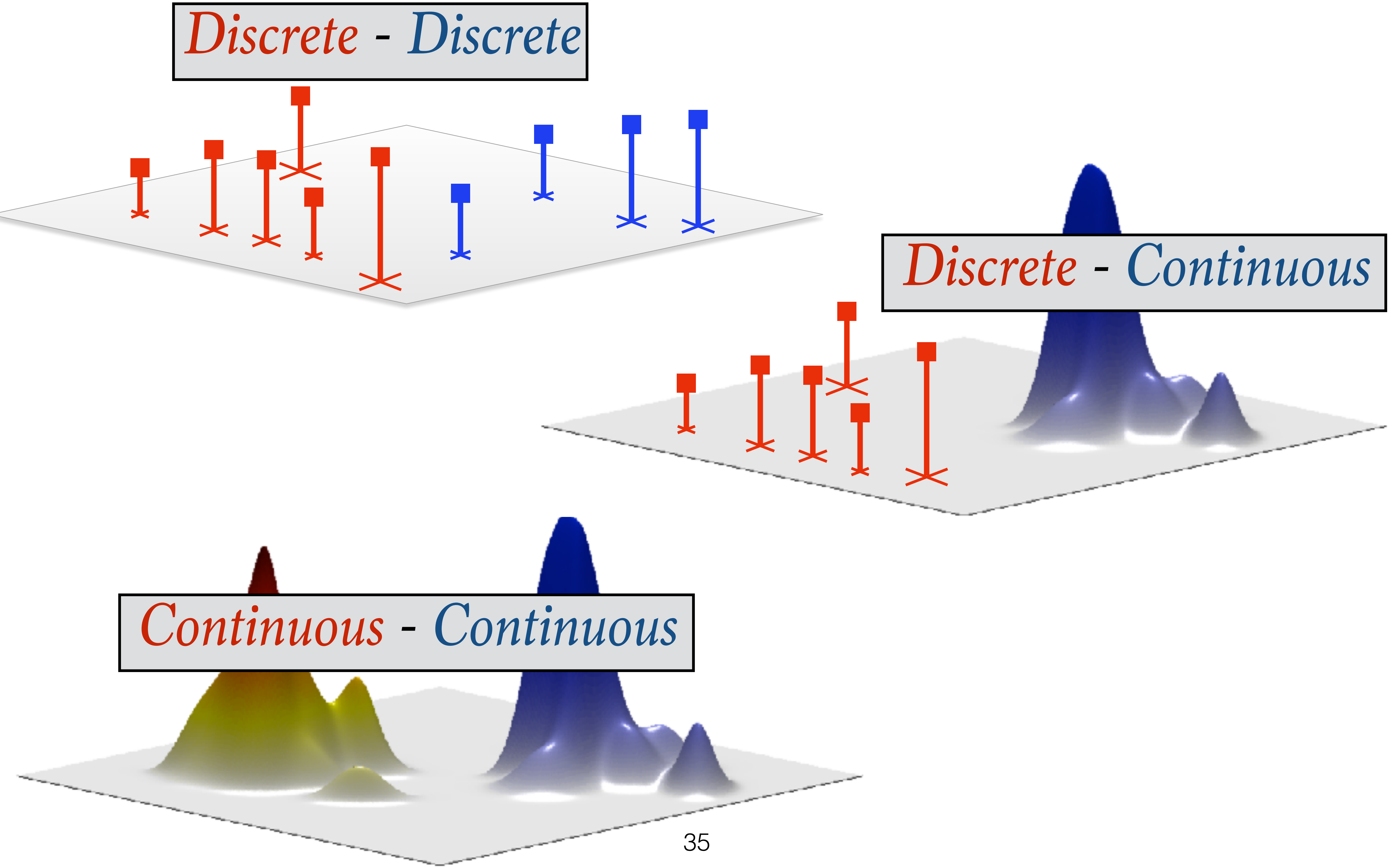
Brain Activation Maps

Generative Models vs. data



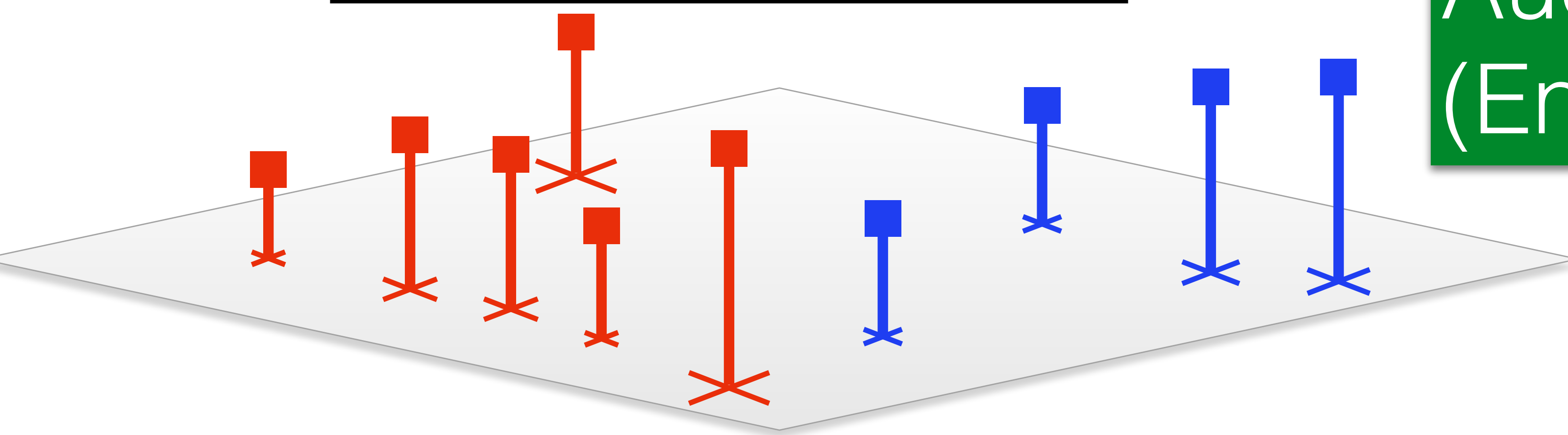
Color Histograms

OT Computations



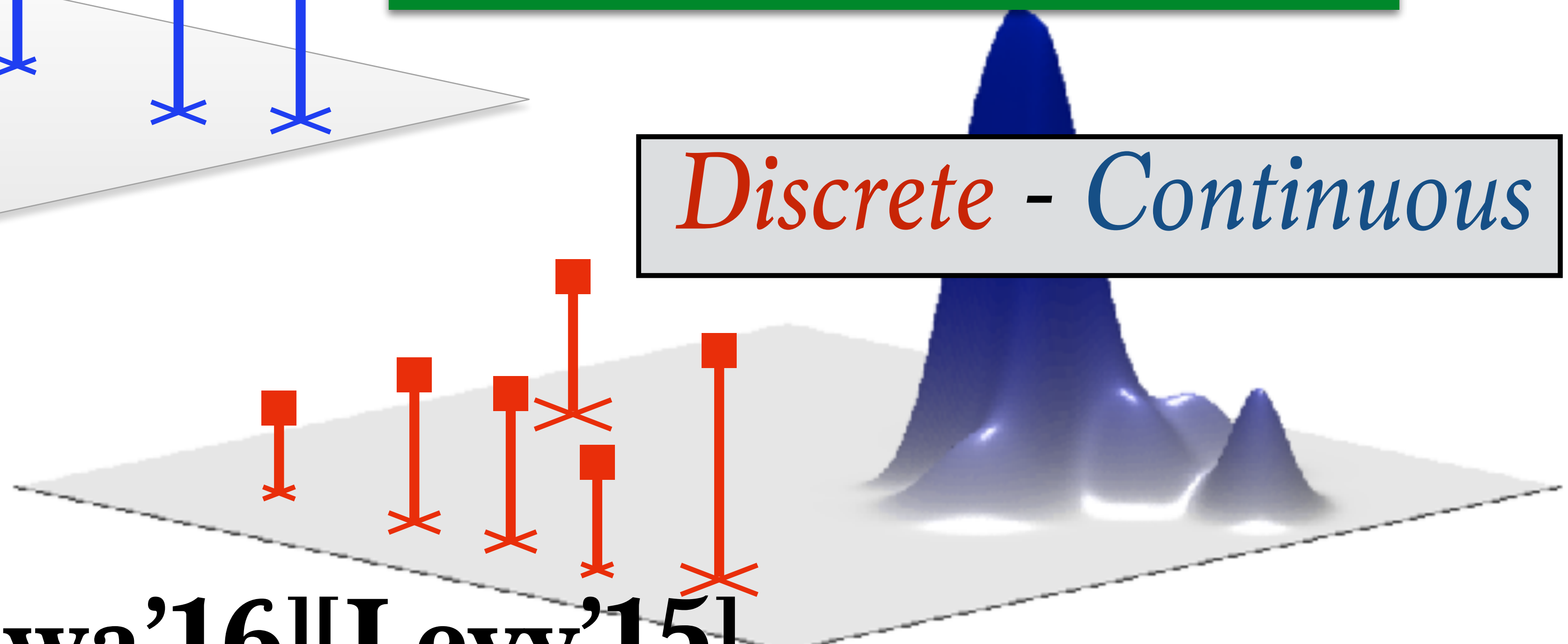
OT Computations

Discrete - *Discrete*



Network flow solvers
Auction algorithm
(Entropic) regularization

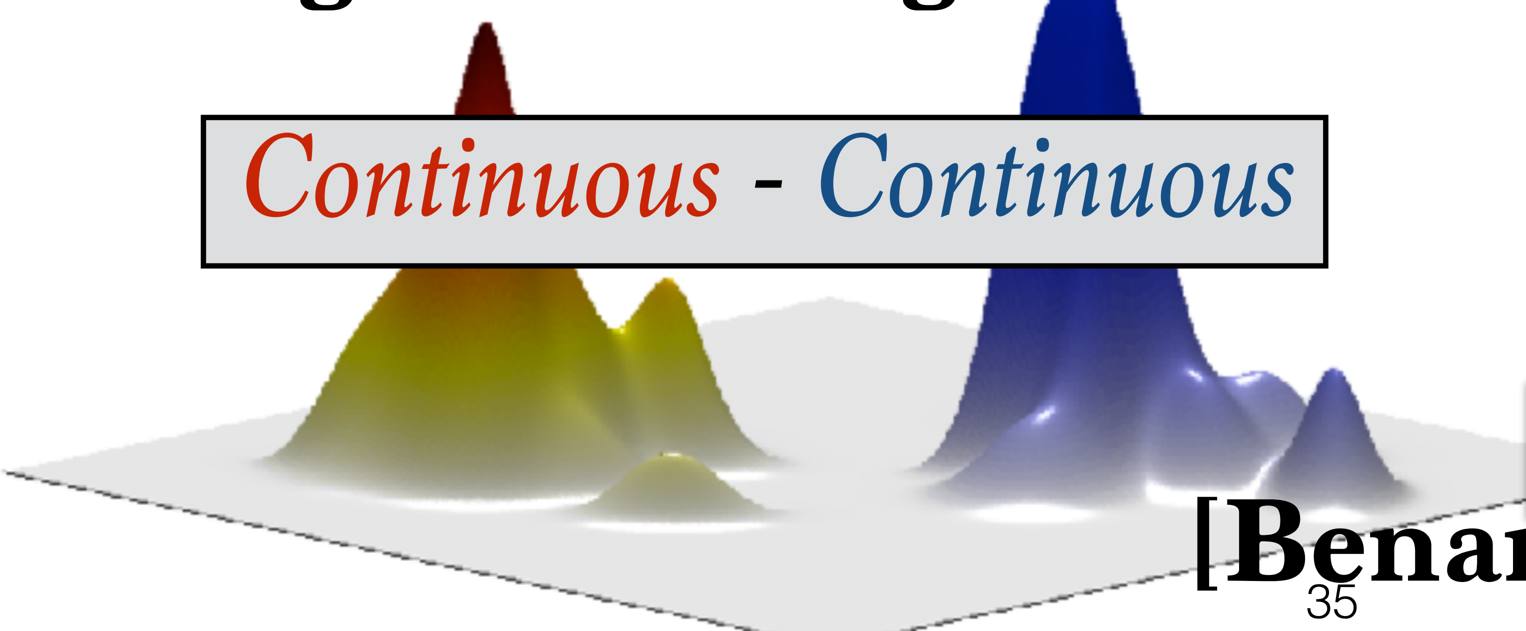
Discrete - *Continuous*



low dim.

[Mérigot'11][Kitagawa'16][Levy'15]

Continuous - *Continuous*



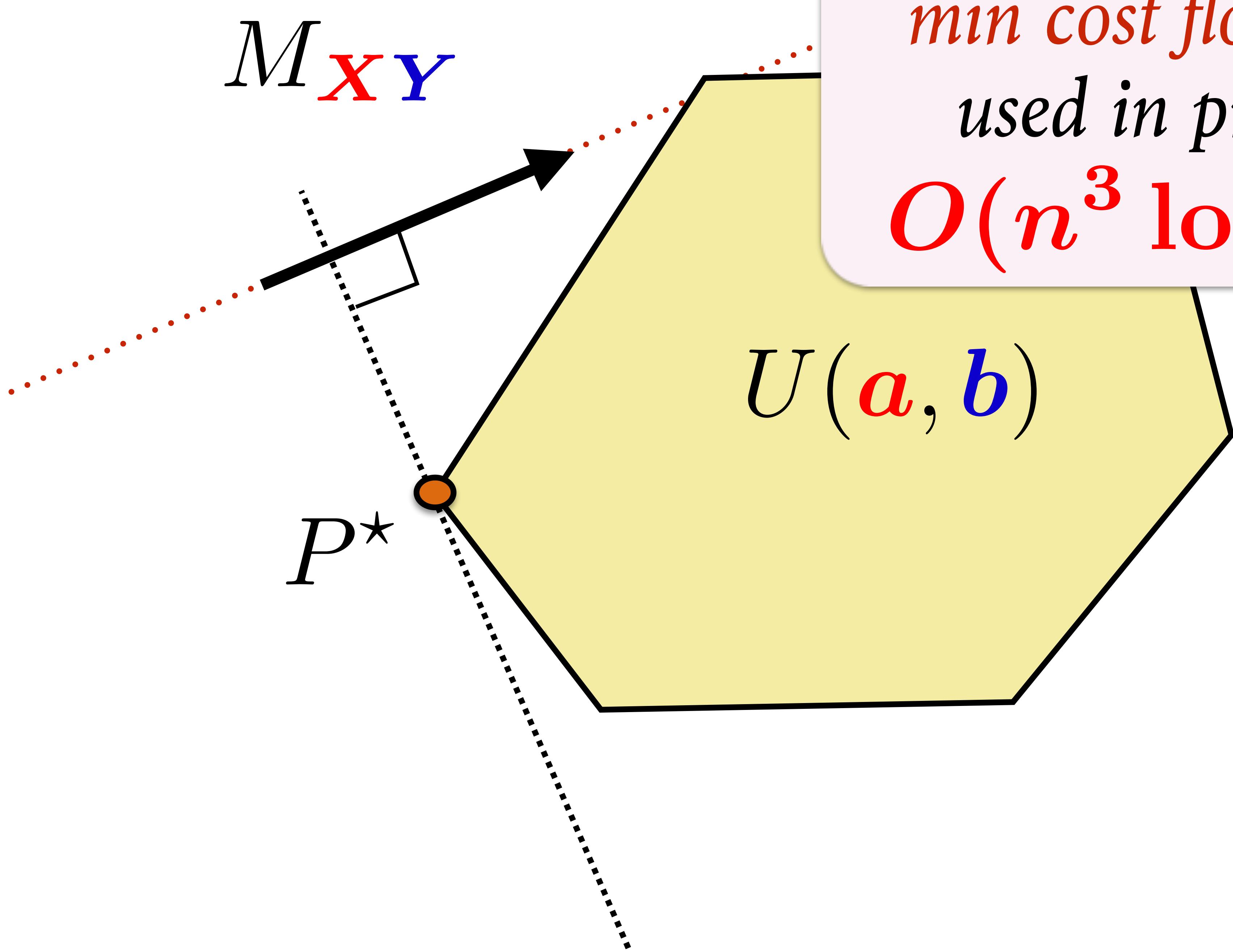
PDE

[Benamou'98]

Stochastic
Optimization

[Genevay'16]

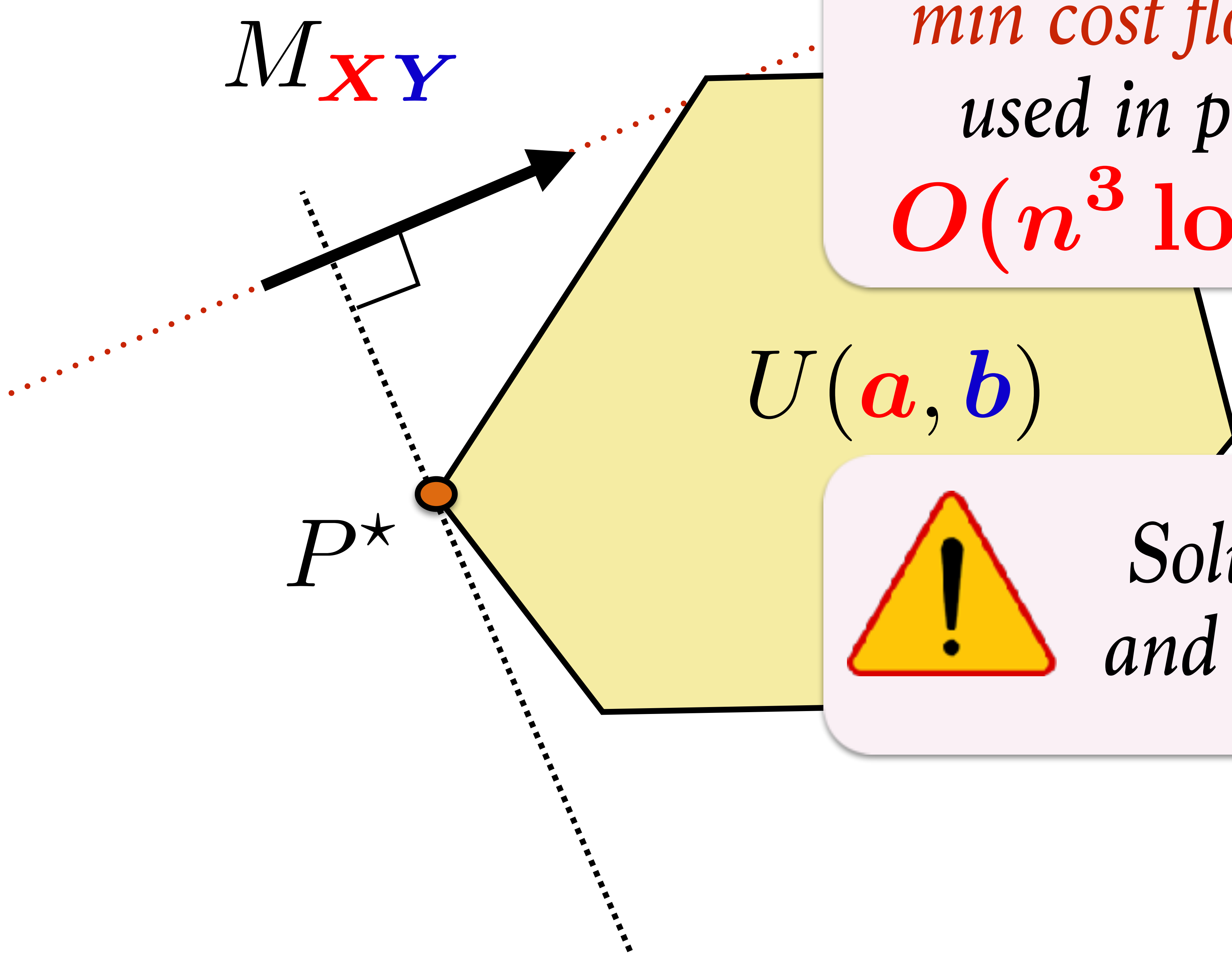
Solving the OT Problem



min cost flow solver
used in practice.

$\boldsymbol{O(n^3 \log(n))}$

Solving the OT Problem

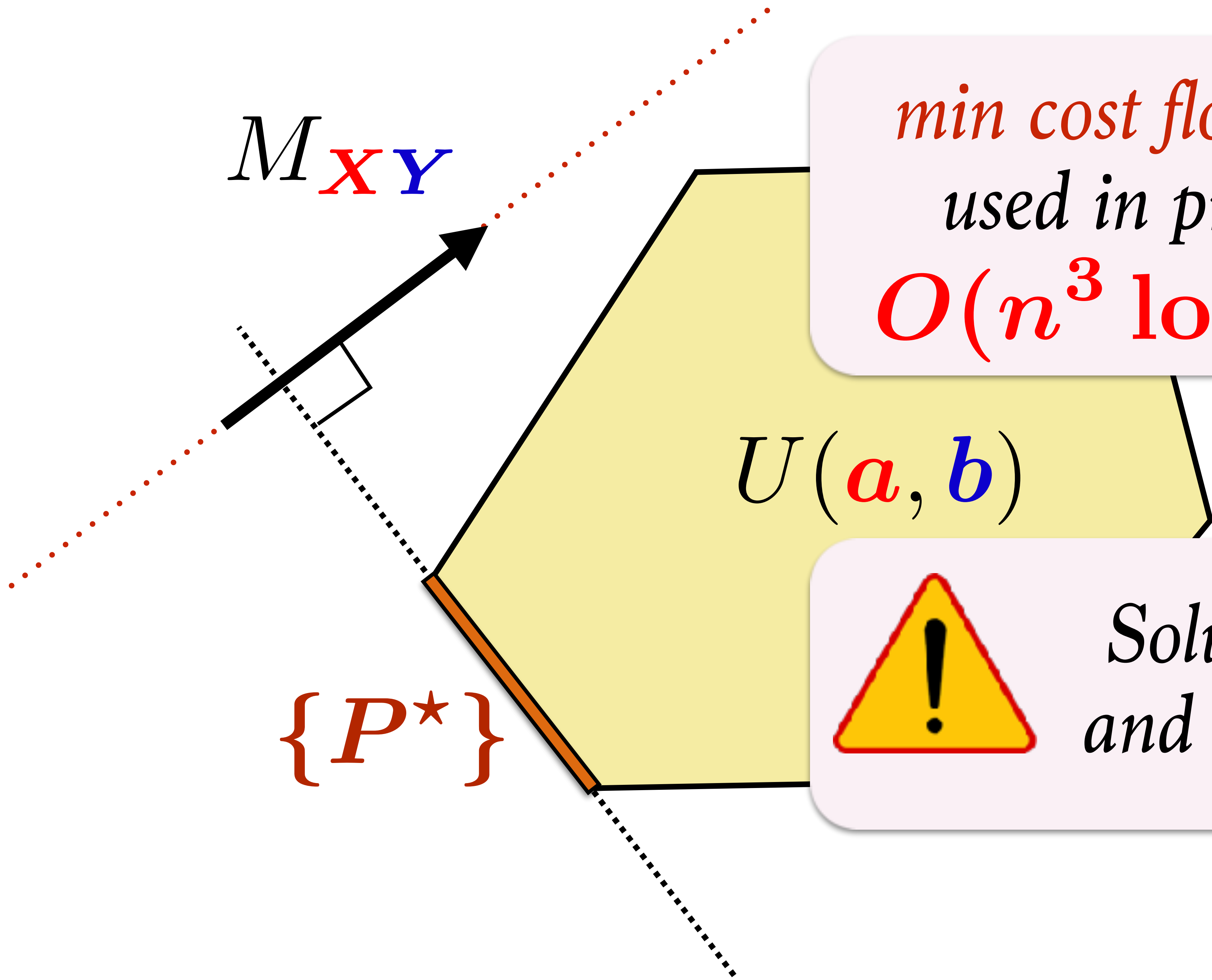


min cost flow solver used in practice.

$O(n^3 \log(n))$

Solution $P^\star$ unstable and not always unique.

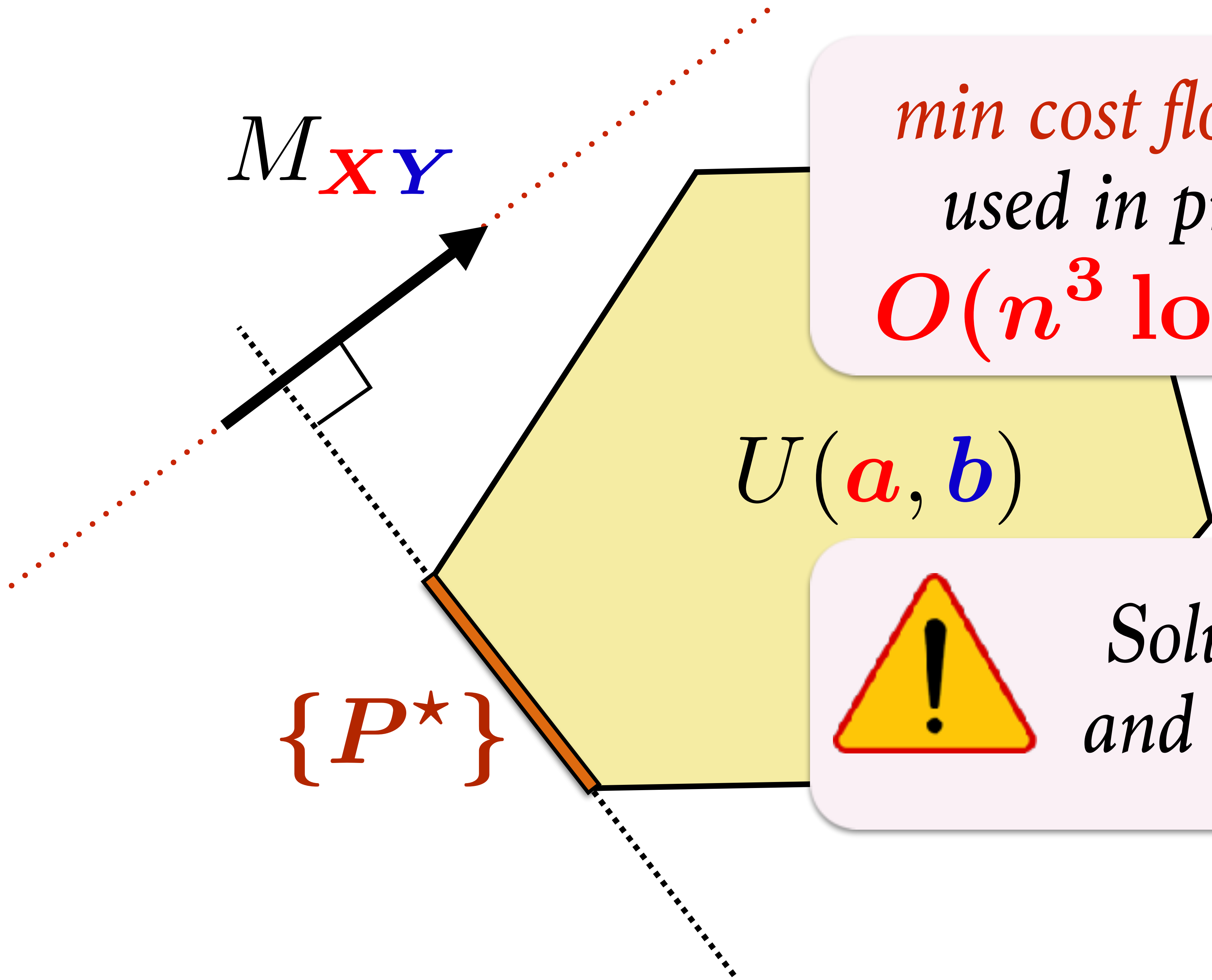
Solving the OT Problem



min cost flow solver *used in practice.* $O(n^3 \log(n))$

Solution $P^\star$ unstable and not always unique.

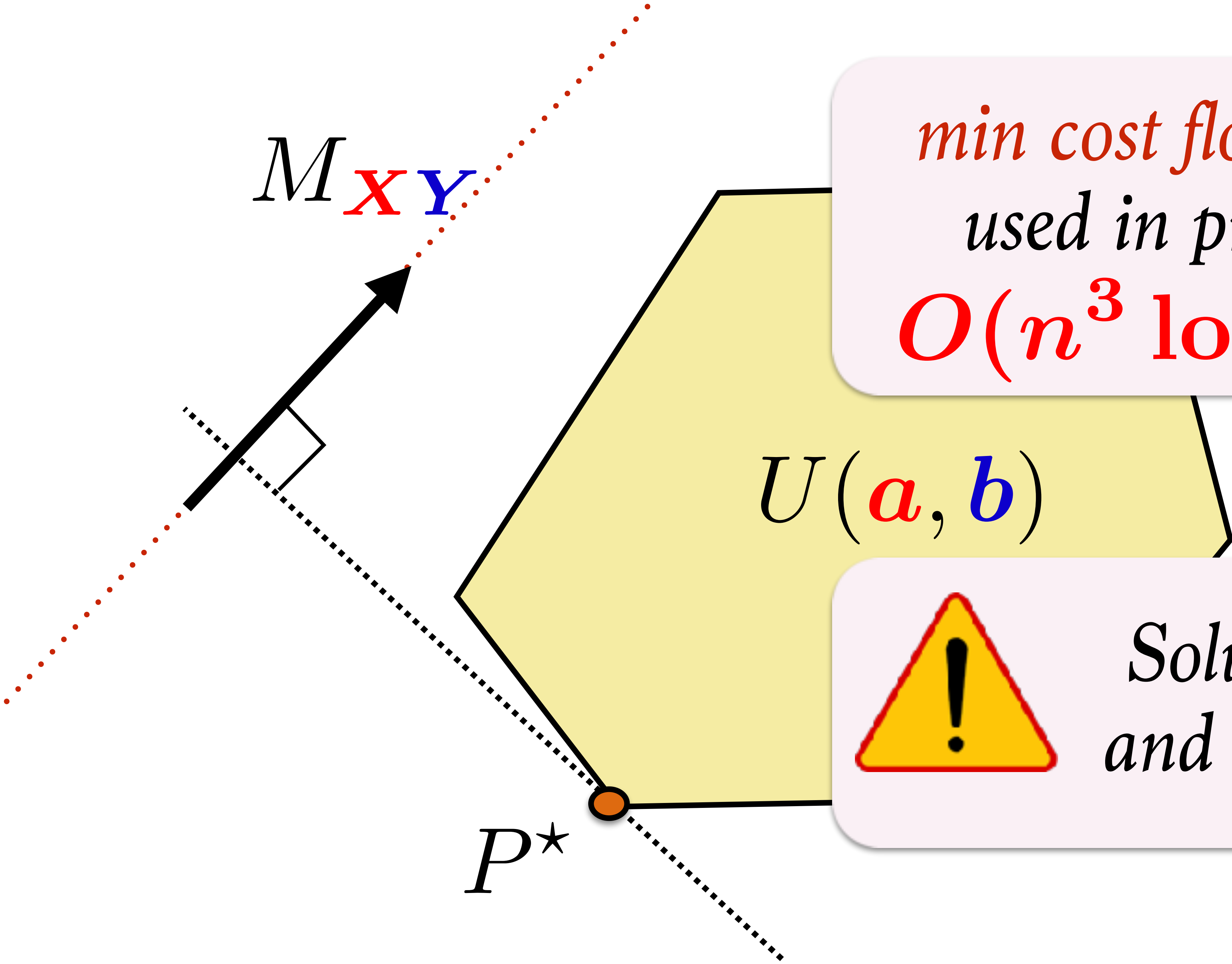
Solving the OT Problem



min cost flow solver *used in practice.*
$O(n^3 \log(n))$

Solution $P^\star$ unstable and not always unique.

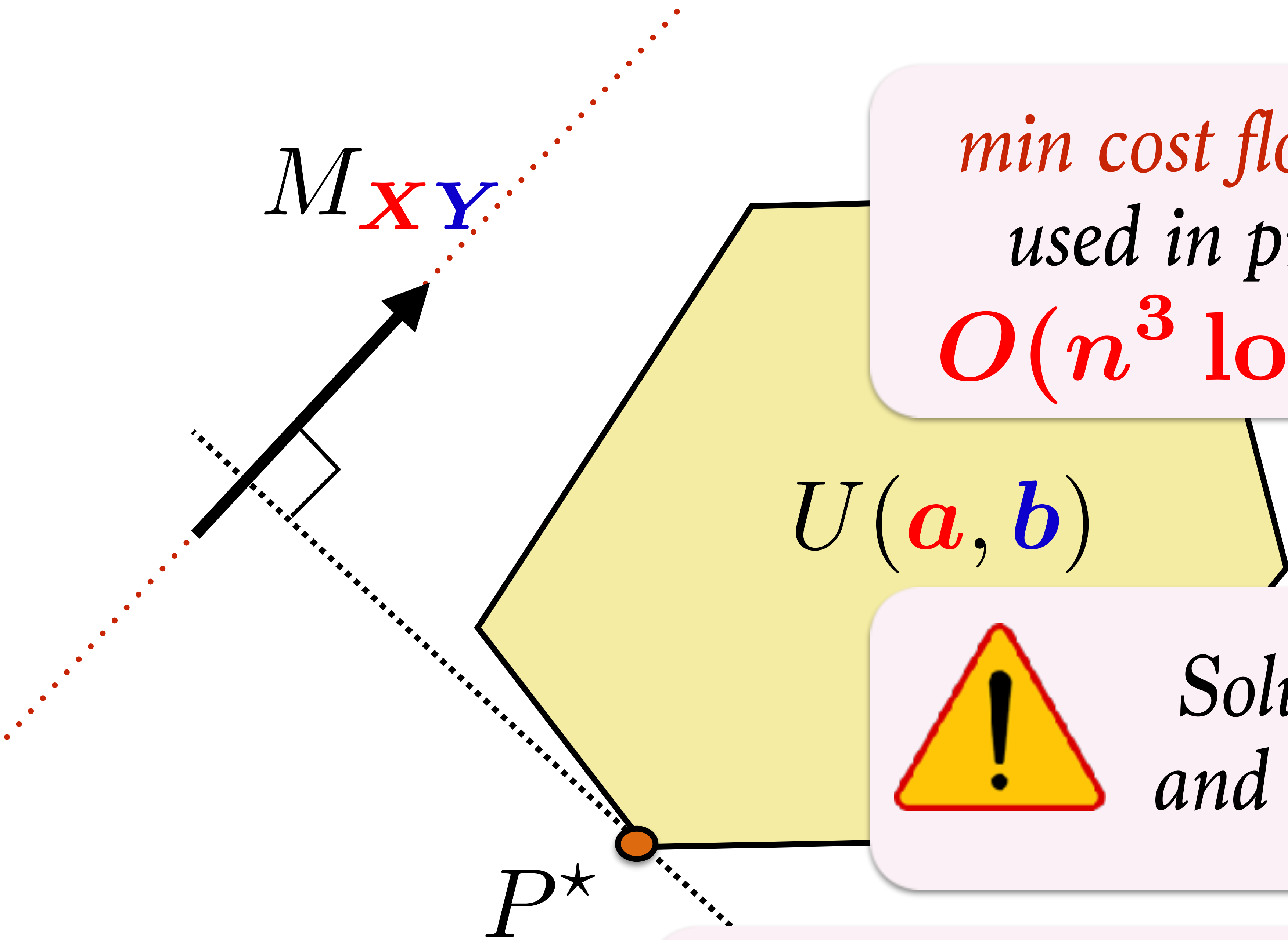
Solving the OT Problem



min cost flow solver *used in practice.* $O(n^3\log(n))$

Solution $P^\star$ unstable and not always unique.

Solving the OT Problem



min cost flow solver
used in practice.
$\boldsymbol{O(n^3 \log(n))}$

Solution $P^\star$ unstable
and not always unique.

$W_p^p(\boldsymbol{\mu}, \boldsymbol{\nu})$ not differentiable.

Discrete OT Problem

emd.c

```c
/*
    emd.c

    Last update: 3/14/98

    An implementation of the Earth Movers Distance.
    Based of the solution for the Transportation problem as described in
    "Introduction to Mathematical Programming" by F. S. Hillier and
    G. J. Lieberman, McGraw-Hill, 1990.

    Copyright (C) 1998 Yossi Rubner
    Computer Science Department, Stanford University
    E-Mail: rubner@cs.stanford.edu   URL: http://vision.stanford.edu/~rubner
*/

/*#include <stdio.h>
#include <stdlib.h>*/
#include <math.h>

#include "emd.h"

#define DEBUG_LEVEL 0
/*
 DEBUG_LEVEL:
   0 = NO MESSAGES
   1 = PRINT THE NUMBER OF ITERATIONS AND THE FINAL RESULT
   2 = PRINT THE RESULT AFTER EVERY ITERATION
   3 = PRINT ALSO THE FLOW AFTER EVERY ITERATION
   4 = PRINT A LOT OF INFORMATION (PROBABLY USEFUL ONLY FOR THE AUTHOR)
*/


#define MAX_SIG_SIZE1  (MAX_SIG_SIZE+1)  /* FOR THE POSIBLE DUMMY FEATURE */

/* NEW TYPES DEFINITION */

/* node1_t IS USED FOR SINGLE-LINKED LISTS */
typedef struct node1_t {
  int i;
  double val;
  struct node1_t *Next;
} node1_t;

/* node1_t IS USED FOR DOUBLE-LINKED LISTS */
typedef struct node2_t {
  int i, j;
  double val;
  struct node2_t *NextC;               /* NEXT COLUMN */
  struct node2_t *NextR;               /* NEXT ROW */
} node2_t;



/* GLOBAL VARIABLE DECLARATION */
static int _n1, _n2;                          /* SIGNATURES SIZES */
static float _C[MAX_SIG_SIZE1][MAX_SIG_SIZE1];/* THE COST MATRIX */
static node2_t _X[MAX_SIG_SIZE1*2];            /* THE BASIC VARIABLES VECTOR */
```

Discrete OT Problem

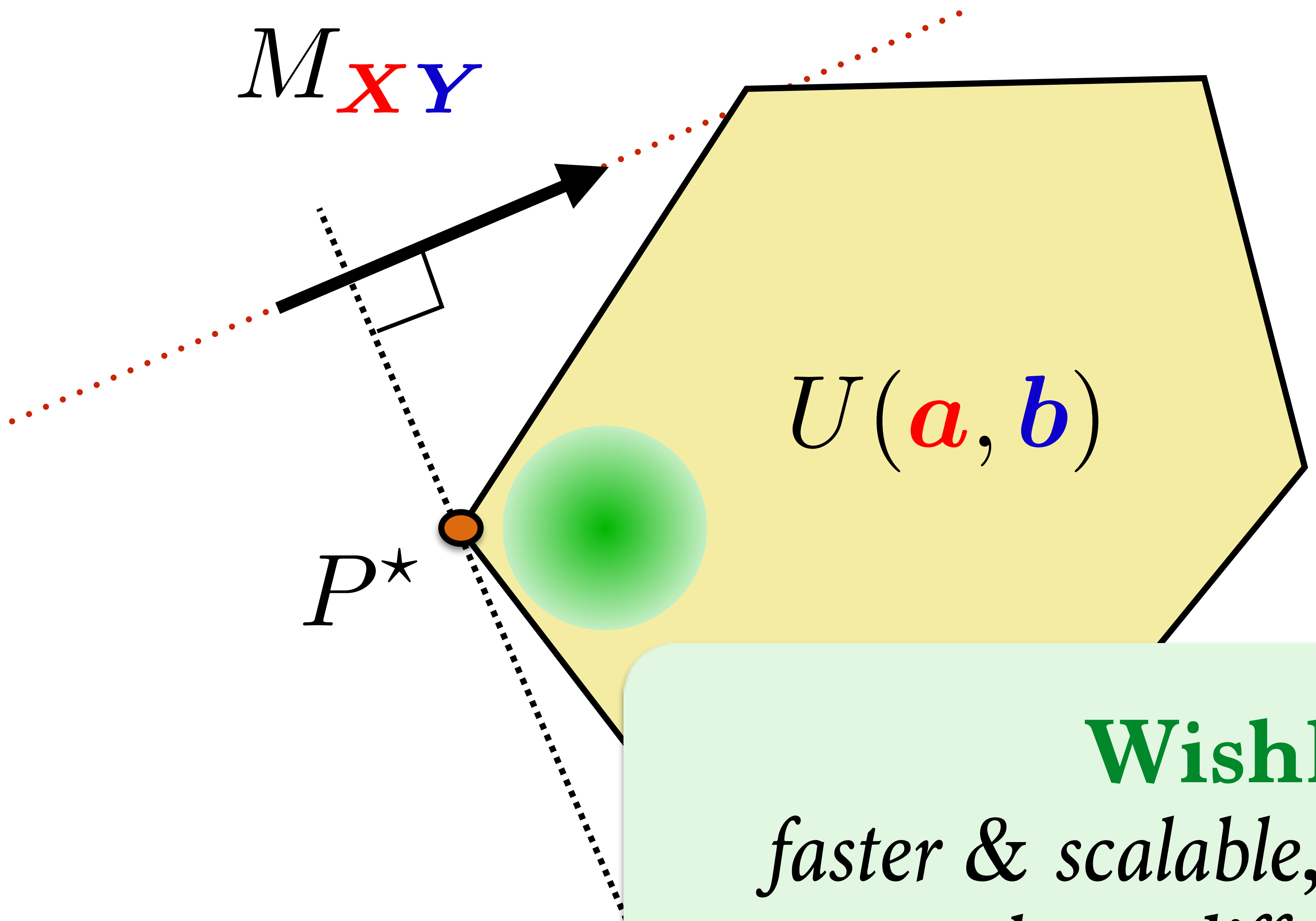
emd.c

```
/*
    emd.c

    Last update: 3/14/98

    An implementation of the Earth Movers Distance.
    Based of the solution for the Transportation problem as described in
    "Introduction to Mathematical Programming" by F. S. Hillier and
    G. J. Lieberman, McGraw-Hill, 1990.

    Copyright (C) 1998 Yossi Rubner
    Computer Science Department, Stanford University
    E-Mail: rubner@cs.stanford.edu   URL: http://vision.stanford.edu/~rubner
*/

/*#include <stdio.h>
#include <stdlib.h>*/
#include <math.h>

#include "emd.h"

#define DEBUG_LEVEL 0
/*
 DEBUG_LEVEL:
   0 = NO MESSAGES
   1 = PRINT THE NUMBER OF ITERATIONS AND THE FINAL RESULT
   2 = PRINT THE RESULT AFTER EVERY ITERATION
   3 = PRINT ALSO THE FLOW AFTER EVERY ITERATION
   4 = PRINT A LOT OF INFORMATION (PROBABLY USEFUL ONLY FOR THE AUTHOR)
*/


#define MAX_SIG_SIZE1  (MAX_SIG_SIZE+1)  /* FOR THE POSIBLE DUMMY FEATURE */

/* NEW TYPES DEFINITION */

/* node1_t IS USED FOR SINGLE-LINKED LISTS */
typedef struct node1_t {
  int i;
  double val;
  struct node1_t *Next;
} node1_t;

/* node1_t IS USED FOR DOUBLE-LINKED LISTS */
typedef struct node2_t {
  int i, j;
  double val;
  struct node2_t *NextC;               /* NEXT COLUMN */
  struct node2_t *NextR;               /* NEXT ROW */
} node2_t;



/* GLOBAL VARIABLE DECLARATION */
static int _n1, _n2;                          /* SIGNATURES SIZES */
static float _C[MAX_SIG_SIZE1][MAX_SIG_SIZE1];/* THE COST MATRIX */
static node2_t _X[MAX_SIG_SIZE1*2];            /* THE BASIC VARIABLES VECTOR */
```

Discrete OT Problem

```
/*
    emd.c

    Last update: 3/14/98

    An implementation of the Earth Movers Distance.
    Based of the solution for the Transportation problem as described in
    "Introduction to Mathematical Programming" by F. S. Hillier and
    G. J. Lieberman, McGraw-Hill, 1990.

    Copyright (C) 1998 Yossi Rubner
    Computer Science Department, Stanford University
    E-Mail: rubner@cs.stanford.edu   URL: http://vision.stanford.edu/~rubner
*/

/*#include <stdio.h>
#include <stdlib.h>*/
#include <math.h>

#include "emd.h"

#define DEBUG_LEVEL 0
/*
 DEBUG_LEVEL:
   0 = NO MESSAGES
   1 = PRINT THE NUMBER OF ITERATIONS AND THE FINAL RESULT
   2 = PRINT THE RESULT AFTER EVERY ITERATION
   3 = PRINT ALSO THE FLOW AFTER EVERY ITERATION
   4 = PRINT A LOT OF INFORMATION (PROBABLY USEFUL ONLY FOR THE AUTHOR)
*/


#define MAX_SIG_SIZE1  (MAX_SIG_SIZE+1)  /* FOR THE POSIBLE DUMMY FEATURE */

/* NEW TYPES DEFINITION */

/* node1_t IS USED FOR SINGLE-LINKED LISTS */
typedef struct node1_t {
  int i;
  double val;
  struct node1_t *Next;
} node1_t;

/* node1_t IS USED FOR DOUBLE-LINKED LISTS */
typedef struct node2_t {
  int i, j;
  double val;
  struct node2_t *NextC;               /* NEXT COLUMN */
  struct node2_t *NextR;               /* NEXT ROW */
} node2_t;



/* GLOBAL VARIABLE DECLARATION */
static int _n1, _n2;                          /* SIGNATURES SIZES */
static float _C[MAX_SIG_SIZE1][MAX_SIG_SIZE1];/* THE COST MATRIX */
static node2_t _X[MAX_SIG_SIZE1*2];            /* THE BASIC VARIABLES VECTOR */
```

Solution: Regularization



Wishlist:
faster & *scalable*, more *stable*,
robust, *differentiable*.

Entropic Regularization [**Wilson'62**]

Def. Regularized Wasserstein, $\gamma \geq 0$

$$W_\gamma(\boldsymbol{\mu}, \boldsymbol{\nu}) \stackrel{\text{def}}{=} \min_{\boldsymbol{P} \in U(\boldsymbol{a}, \boldsymbol{b})} \langle \boldsymbol{P}, M_{\boldsymbol{XY}} \rangle - \gamma E(\boldsymbol{P})$$

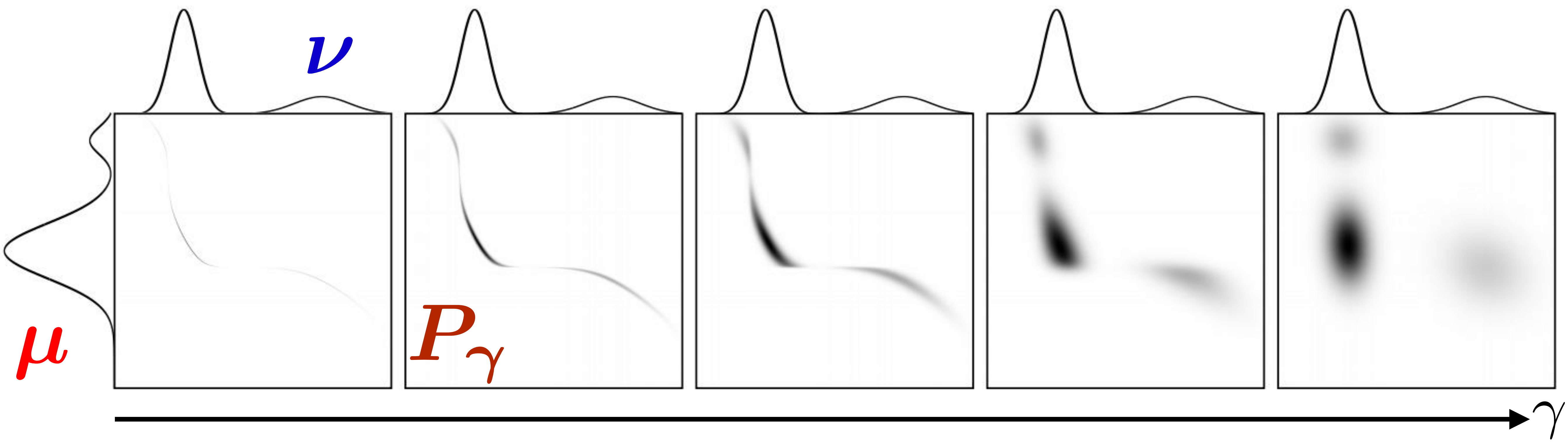
$$E(P) \stackrel{\text{def}}{=} - \sum_{i,j=1}^{nm} P_{ij} (\log P_{ij} - 1)$$

Note: Unique optimal solution because of strong concavity of entropy

Entropic Regularization [Wilson'62]

Def. Regularized Wasserstein, $\gamma \geq 0$

$$W_\gamma(\boldsymbol{\mu}, \boldsymbol{\nu}) \stackrel{\text{def}}{=} \min_{\boldsymbol{P} \in U(\boldsymbol{a}, \boldsymbol{b})} \langle \boldsymbol{P}, M_{\mathbf{XY}} \rangle - \gamma E(\boldsymbol{P})$$

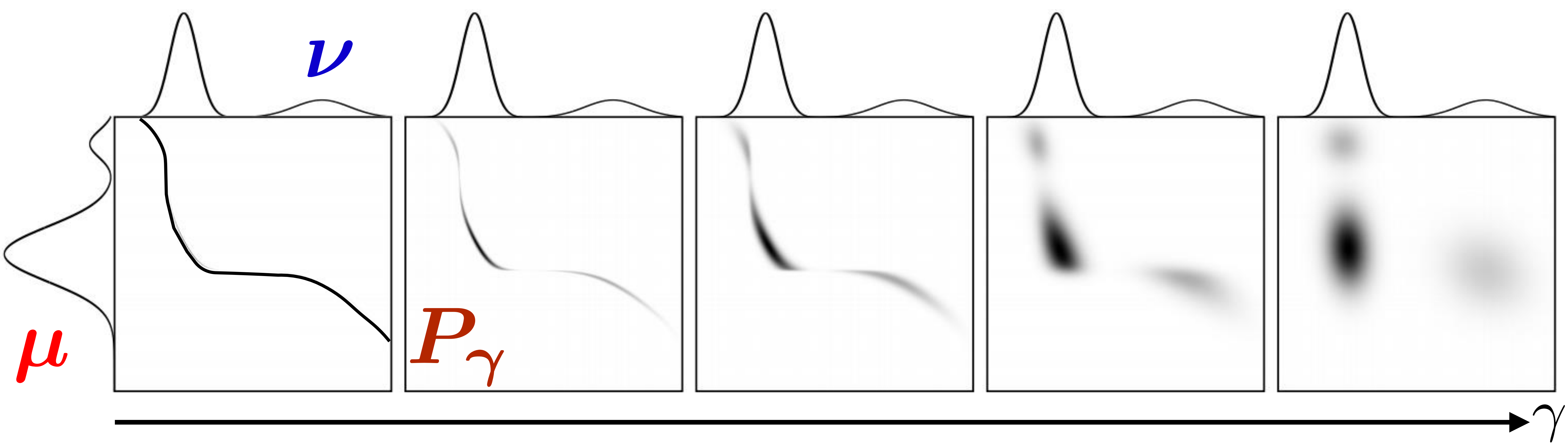


Note: Unique optimal solution because of strong concavity of entropy

Entropic Regularization [Wilson'62]

Def. Regularized Wasserstein, $\gamma \geq 0$

$$W_\gamma(\boldsymbol{\mu}, \boldsymbol{\nu}) \stackrel{\text{def}}{=} \min_{\boldsymbol{P} \in U(\boldsymbol{a}, \boldsymbol{b})} \langle \boldsymbol{P}, M_{\mathbf{XY}} \rangle - \gamma E(\boldsymbol{P})$$

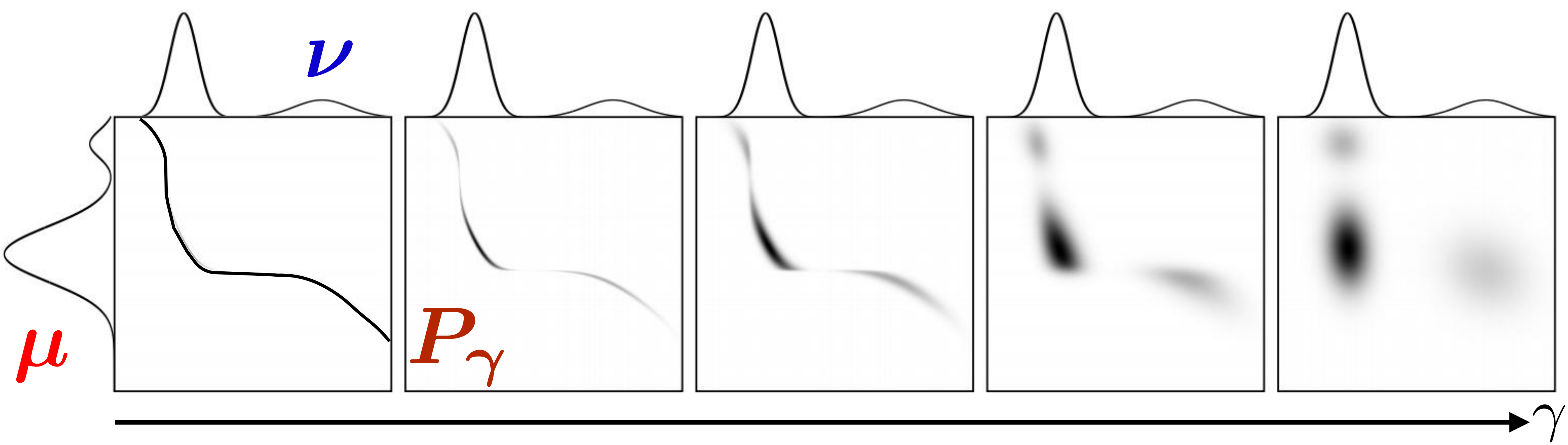


Note: Unique optimal solution because of strong concavity of entropy

Entropic Regularization [**Wilson'62**]

Def. Regularized Wasserstein, $\gamma \geq 0$

$$W_\gamma(\boldsymbol{\mu}, \boldsymbol{\nu}) \stackrel{\text{def}}{=} \min_{\boldsymbol{P} \in U(\boldsymbol{a}, \boldsymbol{b})} \langle \boldsymbol{P}, M_{\mathbf{X}\mathbf{Y}} \rangle - \gamma E(\boldsymbol{P})$$



"static" problem associated with **Schrödinger problem**

Fast & Scalable Algorithm

Prop. If $P_\gamma \stackrel{\text{def}}{=} \underset{\boldsymbol{P}\in U(\boldsymbol{a},\boldsymbol{b})}{\operatorname{argmin}} \langle \boldsymbol{P}, M_{\boldsymbol{XY}} \rangle - \gamma E(\boldsymbol{P})$

then $\exists! \boldsymbol{u} \in \mathbb{R}_+^n, \boldsymbol{v} \in \mathbb{R}_+^m$, such that

$$P_\gamma = \mathbf{diag}(\boldsymbol{u}) K \mathbf{diag}(\boldsymbol{v}), \quad K \stackrel{\text{def}}{=} e^{-M_{\boldsymbol{XY}}/\gamma}$$

Fast & Scalable Algorithm

Prop. If $P_\gamma \overset{\text{def}}{=} \underset{\boldsymbol{P} \in U(\boldsymbol{a},\boldsymbol{b})}{\operatorname{argmin}} \langle \boldsymbol{P}, M_{\mathbf{XY}} \rangle - \gamma E(\boldsymbol{P})$

then $\exists! \boldsymbol{u} \in \mathbb{R}_+^n, \boldsymbol{v} \in \mathbb{R}_+^m$, such that

$$P_\gamma = \mathbf{diag}(\boldsymbol{u}) K \mathbf{diag}(\boldsymbol{v}), \quad K \overset{\text{def}}{=} e^{-M_{\mathbf{XY}}/\gamma}$$

$$L(P, \alpha, \beta) = \sum_{ij} P_{ij} M_{ij} + \gamma P_{ij} (\log P_{ij} - 1) + \alpha^T (P\mathbf{1} - \boldsymbol{a}) + \beta^T (P^T \mathbf{1} - \boldsymbol{b})$$

$$\partial L / \partial P_{ij} = M_{ij} + \gamma \log P_{ij} + \alpha_i + \beta_j$$

$$(\boldsymbol{\partial L / \partial P_{ij}} = 0) \Rightarrow P_{ij} = \boldsymbol{e}^{\frac{\boldsymbol{\alpha_i}}{\gamma}} e^{-\frac{M_{ij}}{\gamma}} \boldsymbol{e}^{\frac{\boldsymbol{\beta_j}}{\gamma}} = \boldsymbol{u_i} K_{ij} \boldsymbol{v_j}$$

Fast & Scalable Algorithm

Prop. If $P_\gamma \overset{\text{def}}{=} \underset{\boldsymbol{P}\in U(\boldsymbol{a},\boldsymbol{b})}{\operatorname{argmin}} \langle \boldsymbol{P}, M_{\mathbf{XY}} \rangle - \gamma E(\boldsymbol{P})$
then $\exists! \boldsymbol{u} \in \mathbb{R}^n_+, \boldsymbol{v} \in \mathbb{R}^m_+$, such that

$$P_\gamma = \mathbf{diag}(\boldsymbol{u}) K \mathbf{diag}(\boldsymbol{v}), \quad K \overset{\text{def}}{=} e^{-M_{\mathbf{XY}}/\gamma}$$

$$P_\gamma \in U(\boldsymbol{a},\boldsymbol{b}) \Leftrightarrow \begin{cases} \mathbf{diag}(\boldsymbol{u}) K \mathbf{diag}(\boldsymbol{v}) \mathbf{1}_m & = \boldsymbol{a} \\ \mathbf{diag}(\boldsymbol{v}) K^T \mathbf{diag}(\boldsymbol{u}) \mathbf{1}_n & = \boldsymbol{b} \end{cases}$$

Fast & Scalable Algorithm

Prop. If $P_\gamma \stackrel{\text{def}}{=} \underset{\boldsymbol{P} \in U(\boldsymbol{a},\boldsymbol{b})}{\operatorname{argmin}} \langle \boldsymbol{P}, M_{\mathbf{XY}} \rangle - \gamma E(\boldsymbol{P})$

then $\exists! \boldsymbol{u} \in \mathbb{R}^n_+, \boldsymbol{v} \in \mathbb{R}^m_+$, such that

$$P_\gamma = \mathbf{diag}(\boldsymbol{u}) K \mathbf{diag}(\boldsymbol{v}), \quad K \stackrel{\text{def}}{=} e^{-M_{\mathbf{XY}}/\gamma}$$

$$P_\gamma \in U(\boldsymbol{a}, \boldsymbol{b}) \Leftrightarrow \begin{cases} \mathbf{diag}(\boldsymbol{u}) K \mathbf{diag}(\boldsymbol{v}) \mathbf{1}_m & = \boldsymbol{a} \\ \mathbf{diag}(\boldsymbol{v}) K^T \underbrace{\mathbf{diag}(\boldsymbol{u}) \mathbf{1}_n}_{\boldsymbol{u}} & = \boldsymbol{b} \end{cases}$$

Fast & Scalable Algorithm

Prop. If $P_\gamma \stackrel{\text{def}}{=} \operatorname*{argmin}_{\boldsymbol{P} \in U(\boldsymbol{a},\boldsymbol{b})} \langle \boldsymbol{P}, M_{\mathbf{XY}} \rangle - \gamma E(\boldsymbol{P})$

then $\exists ! \boldsymbol{u} \in \mathbb{R}_+^n, \boldsymbol{v} \in \mathbb{R}_+^m$, such that

$$P_\gamma = \mathbf{diag}(\boldsymbol{u}) K \mathbf{diag}(\boldsymbol{v}), \quad K \stackrel{\text{def}}{=} e^{-M_{\mathbf{XY}}/\gamma}$$

$$P_\gamma \in U(\boldsymbol{a}, \boldsymbol{b}) \Leftrightarrow \begin{cases} \mathbf{diag}(\boldsymbol{u}) K \overbrace{\mathbf{diag}(\boldsymbol{v}) \mathbf{1}_m}^{\boldsymbol{v}} & = \boldsymbol{a} \\ \mathbf{diag}(\boldsymbol{v}) K^T \underbrace{\mathbf{diag}(\boldsymbol{u}) \mathbf{1}_n}_{\boldsymbol{u}} & = \boldsymbol{b} \end{cases}$$

Fast & Scalable Algorithm

Prop. If $P_\gamma \stackrel{\text{def}}{=} \underset{\boldsymbol{P}\in U(\boldsymbol{a},\boldsymbol{b})}{\operatorname{argmin}} \langle \boldsymbol{P}, M_{\boldsymbol{XY}} \rangle - \gamma E(\boldsymbol{P})$
then $\exists! \boldsymbol{u} \in \mathbb{R}_+^n, \boldsymbol{v} \in \mathbb{R}_+^m$, such that

$$P_\gamma = \mathbf{diag}(\boldsymbol{u}) K \mathbf{diag}(\boldsymbol{v}), \quad K \stackrel{\text{def}}{=} e^{-M_{\boldsymbol{XY}}/\gamma}$$

$$P_\gamma \in U(\boldsymbol{a}, \boldsymbol{b}) \Leftrightarrow \begin{cases} \mathbf{diag}(\boldsymbol{u}) K \boldsymbol{v} & = \boldsymbol{a} \\ \mathbf{diag}(\boldsymbol{v}) K^T \boldsymbol{u} & = \boldsymbol{b} \end{cases}$$

Fast & Scalable Algorithm

Prop. If $P_\gamma \stackrel{\text{def}}{=} \operatorname*{argmin}_{\boldsymbol{P} \in U(\boldsymbol{a},\boldsymbol{b})} \langle \boldsymbol{P}, M_{\mathbf{XY}} \rangle - \gamma E(\boldsymbol{P})$

then $\exists! \boldsymbol{u} \in \mathbb{R}_+^n, \boldsymbol{v} \in \mathbb{R}_+^m$, such that

$$P_\gamma = \mathbf{diag}(\boldsymbol{u}) K \mathbf{diag}(\boldsymbol{v}), \quad K \stackrel{\text{def}}{=} e^{-M_{\mathbf{XY}}/\gamma}$$

$$P_\gamma \in U(\boldsymbol{a}, \boldsymbol{b}) \Leftrightarrow \begin{cases} \boldsymbol{u} \odot K \boldsymbol{v} & = \boldsymbol{a} \\ \boldsymbol{v} \odot K^T \boldsymbol{u} & = \boldsymbol{b} \end{cases}$$

Fast & Scalable Algorithm

Prop. If $P_\gamma \stackrel{\text{def}}{=} \underset{\boldsymbol{P} \in U(\boldsymbol{a},\boldsymbol{b})}{\operatorname{argmin}} \langle \boldsymbol{P}, M_{\mathbf{XY}} \rangle - \gamma E(\boldsymbol{P})$

then $\exists! \boldsymbol{u} \in \mathbb{R}_+^n, \boldsymbol{v} \in \mathbb{R}_+^m$, such that

$$P_\gamma = \mathbf{diag}(\boldsymbol{u}) K \mathbf{diag}(\boldsymbol{v}), \quad K \stackrel{\text{def}}{=} e^{-M_{\mathbf{XY}}/\gamma}$$

$$P_\gamma \in U(\boldsymbol{a}, \boldsymbol{b}) \Leftrightarrow \begin{cases} \boldsymbol{u} = \boldsymbol{a} / K\boldsymbol{v} \\ \boldsymbol{v} = \boldsymbol{b} / K^T \boldsymbol{u} \end{cases}$$

Fast & Scalable Algorithm

Sinkhorn's Algorithm : Repeat

1. $\boldsymbol{u} = \boldsymbol{a} / K \boldsymbol{v}$
2. $\boldsymbol{v} = \boldsymbol{b} / K^T \boldsymbol{u}$

Fast & Scalable Algorithm

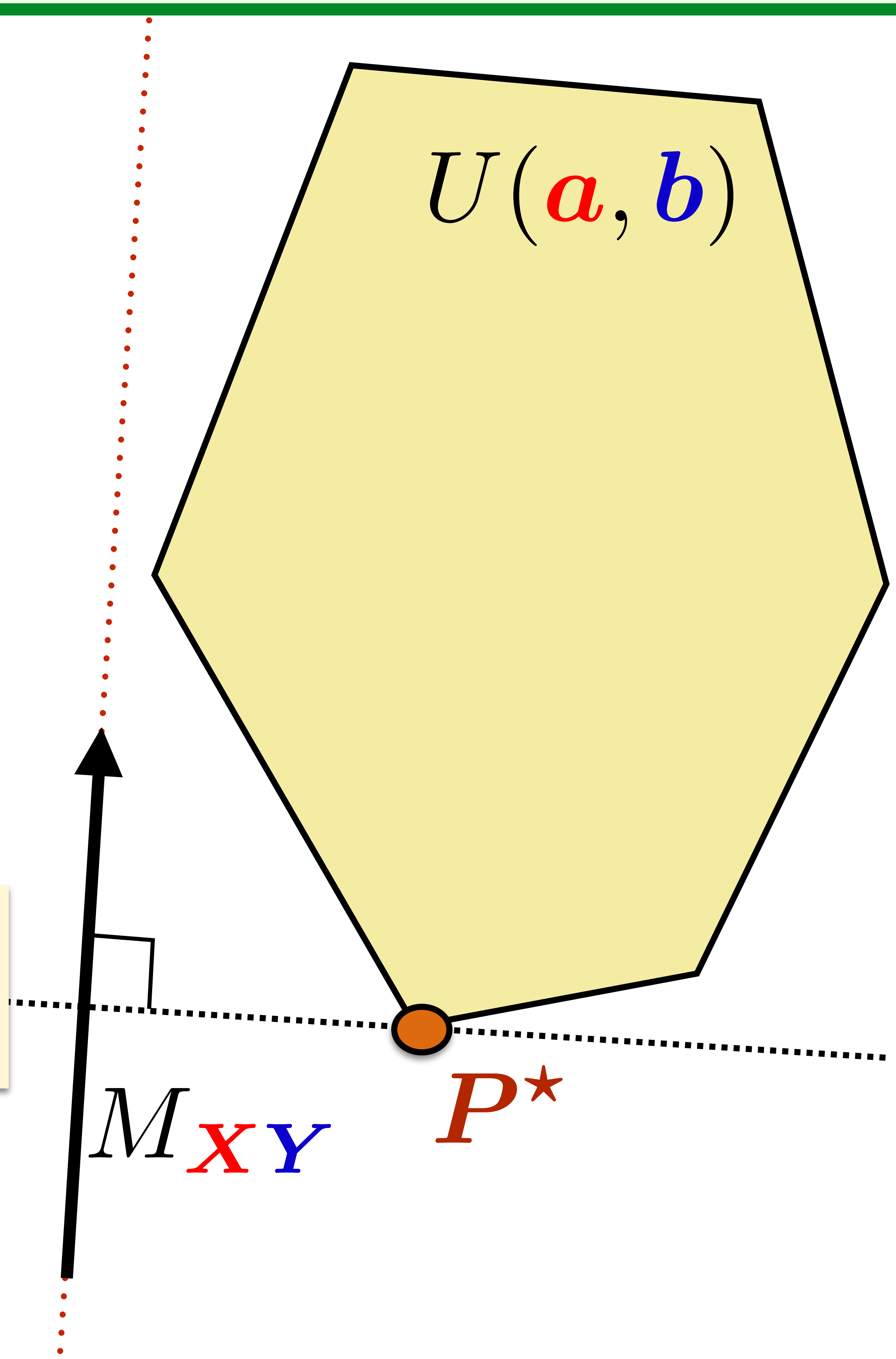
Sinkhorn's Algorithm : Repeat

1. $\boldsymbol{u} = \boldsymbol{a}/K\boldsymbol{v}$
2. $\boldsymbol{v} = \boldsymbol{b}/K^T\boldsymbol{u}$

- **[Sinkhorn'64]** proved convergence for the first time.
- **[Lorenz'89]** linear convergence, see **[Altschuler'17]**
- $O(nm)$ complexity, GPGPU parallel **[Cuturi'13]** .
- $O(n \log n)$ on gridded spaces using convolutions. **[Solomon'15]**

Sinkhorn in between W and MMD

$$\boldsymbol{\mu} = \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{\delta}_{\boldsymbol{x}_i} \qquad \boldsymbol{\nu} = \sum_{j=1}^{m} \boldsymbol{b}_j \boldsymbol{\delta}_{\boldsymbol{y}_j}$$



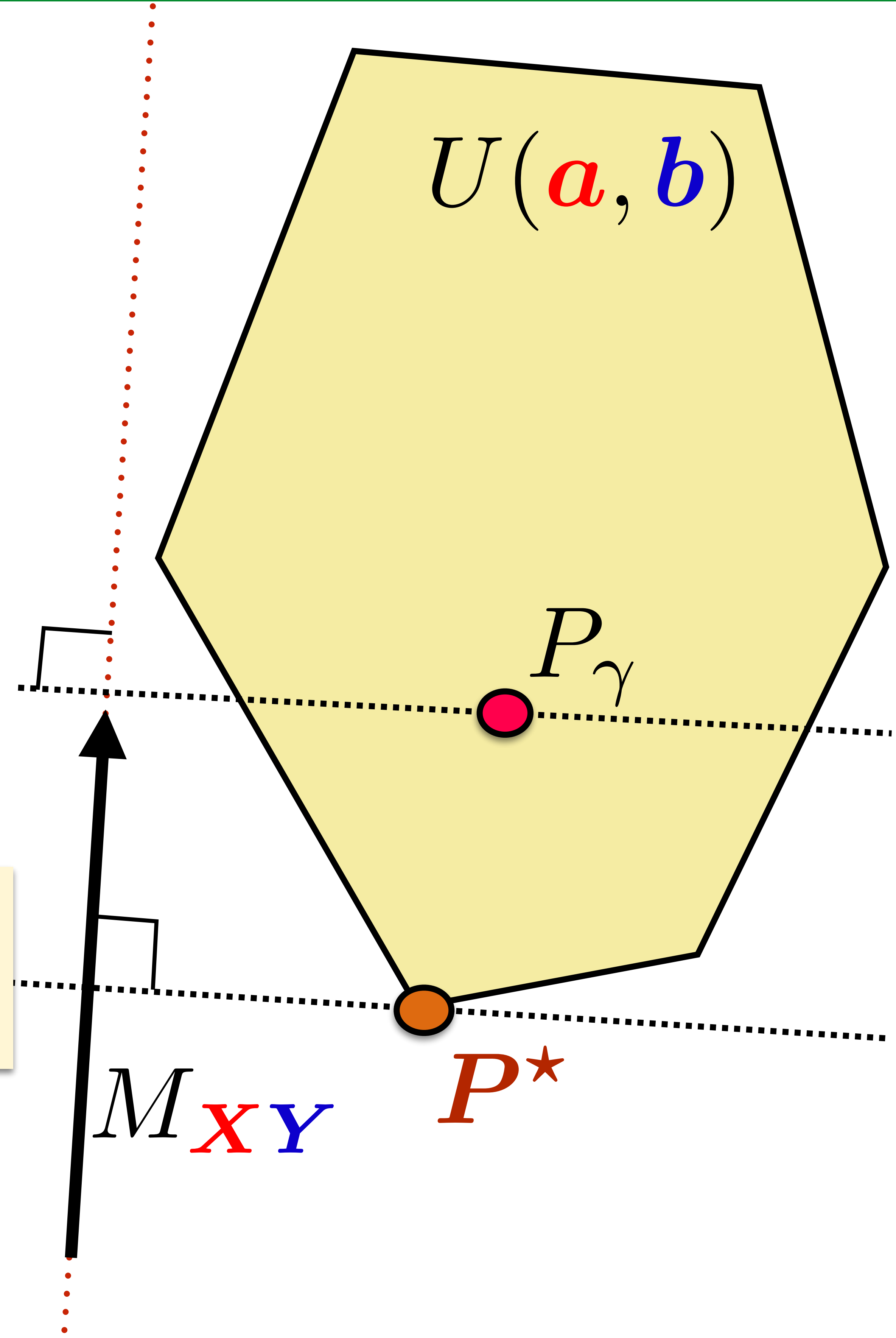
$$W^p(\boldsymbol{\mu}, \boldsymbol{\nu}) = \langle \boldsymbol{P}^\star, M_{\mathbf{XY}} \rangle$$

Sinkhorn in between W and MMD

$$\boldsymbol{\mu} = \sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{\delta}_{\boldsymbol{x}_i} \qquad \boldsymbol{\nu} = \sum_{j=1}^{m} \boldsymbol{b}_j \boldsymbol{\delta}_{\boldsymbol{y}_j}$$

$$W_\gamma(\boldsymbol{\mu}, \boldsymbol{\nu}) = \langle P_\gamma, M_{\mathbf{XY}} \rangle$$

$$W^p(\boldsymbol{\mu}, \boldsymbol{\nu}) = \langle \boldsymbol{P}^\star, M_{\mathbf{XY}} \rangle$$



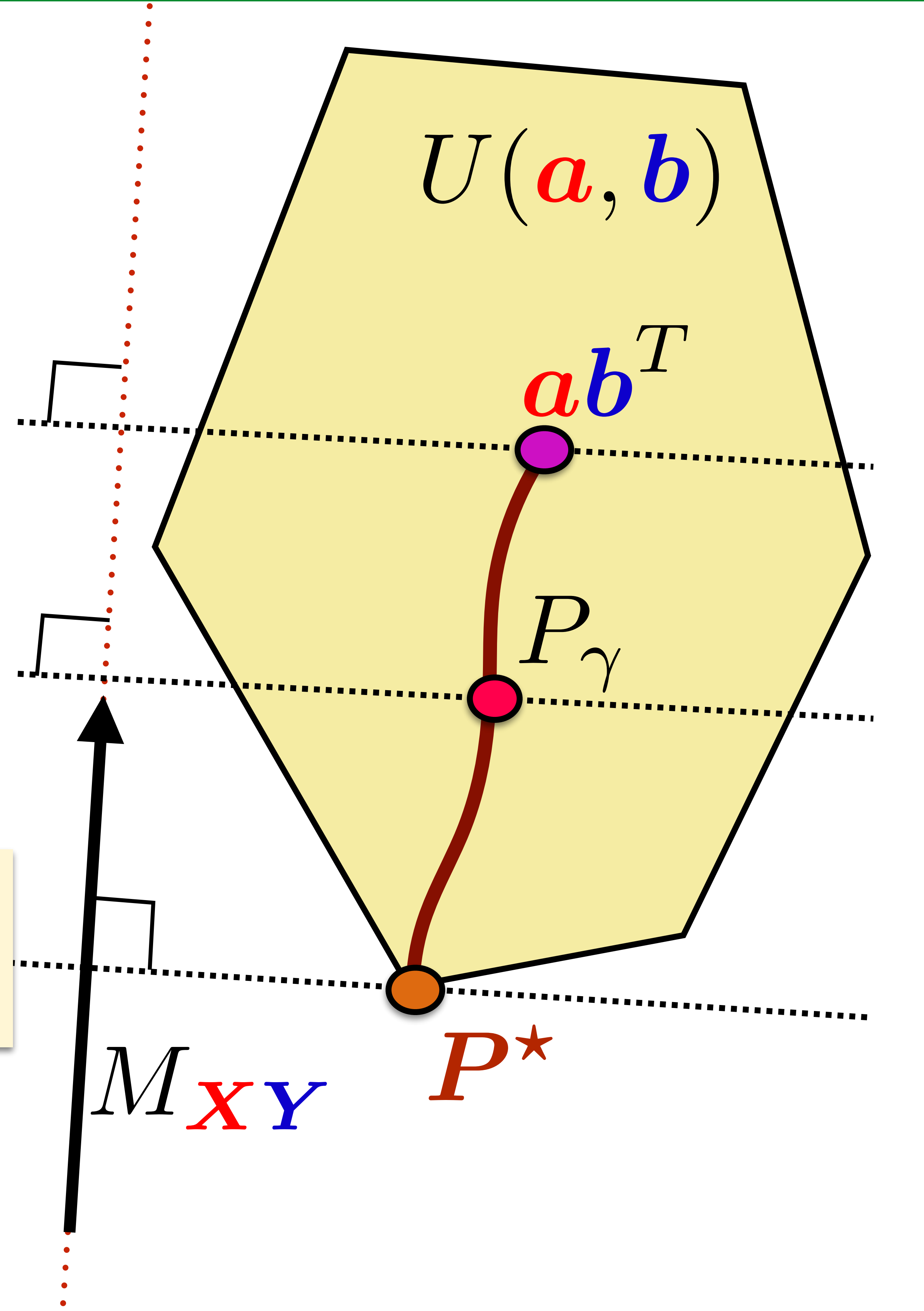
Sinkhorn in between W and MMD

$$\boldsymbol{\mu} = \sum_{i=1}^{n} \boldsymbol{a}_i \delta_{\boldsymbol{x}_i} \qquad \boldsymbol{\nu} = \sum_{j=1}^{m} \boldsymbol{b}_j \delta_{\boldsymbol{y}_j}$$

$$\mathcal{E}(\boldsymbol{\mu}, \boldsymbol{\nu}) = \langle \boldsymbol{a}\boldsymbol{b}^T, M_{\boldsymbol{XY}} \rangle$$

$$W_\gamma(\boldsymbol{\mu}, \boldsymbol{\nu}) = \langle P_\gamma, M_{\boldsymbol{XY}} \rangle$$

$$W^p(\boldsymbol{\mu}, \boldsymbol{\nu}) = \langle \boldsymbol{P}^\star, M_{\boldsymbol{XY}} \rangle$$



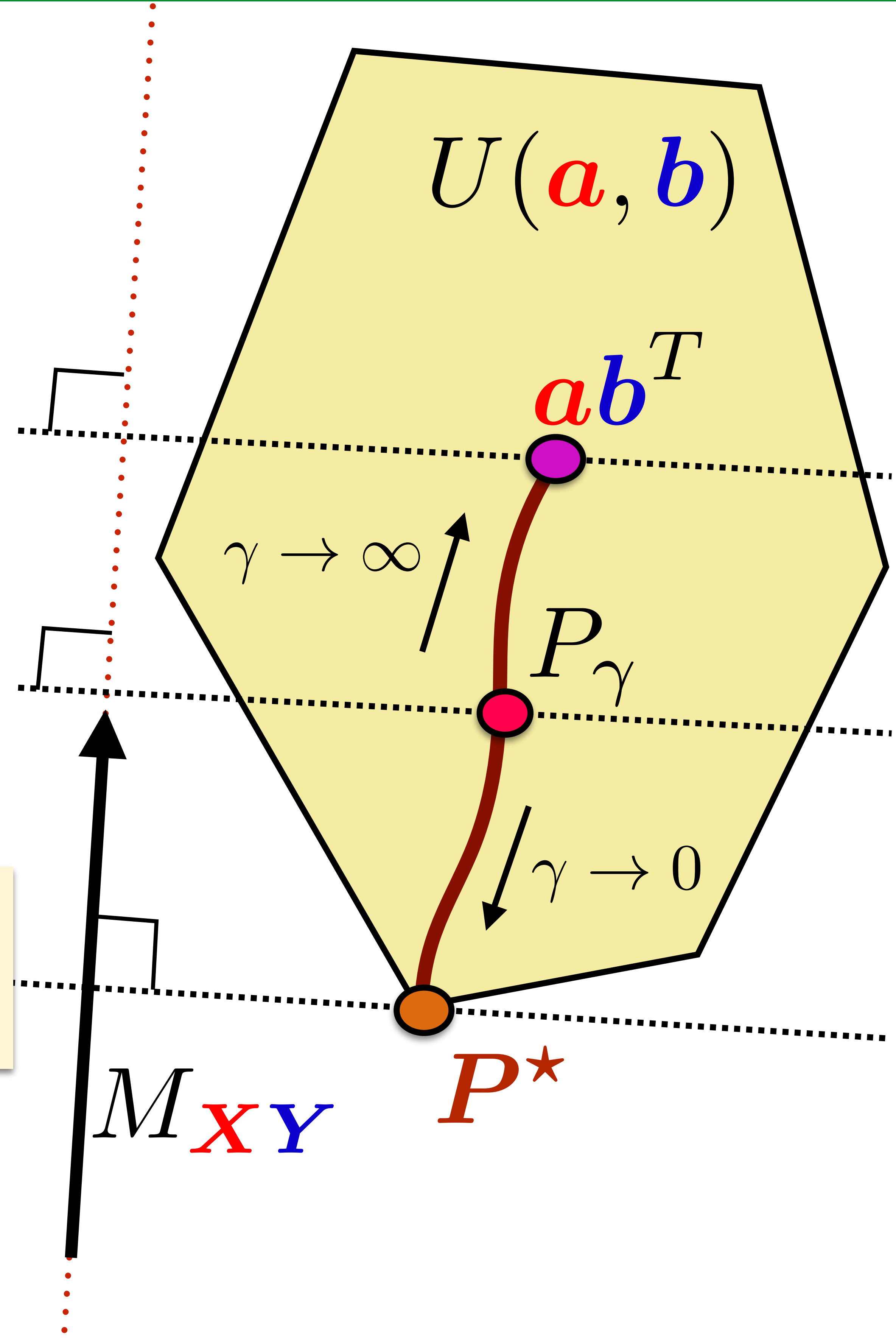
Sinkhorn in between W and MMD

$$\mu = \sum_{i=1}^{n} a_i \delta_{x_i} \qquad \nu = \sum_{j=1}^{m} b_j \delta_{y_j}$$

$$\mathcal{E}(\mu, \nu) = \langle \mathbf{a}\mathbf{b}^T, M_{\mathbf{XY}} \rangle$$

$$W_\gamma(\mu, \nu) = \langle P_\gamma, M_{\mathbf{XY}} \rangle$$

$$W^p(\mu, \nu) = \langle P^\star, M_{\mathbf{XY}} \rangle$$



Sinkhorn in between W and MMD

$$\mathcal{E}(\boldsymbol{\mu}, \boldsymbol{\nu}) = \langle \boldsymbol{a}\boldsymbol{b}^T, M_{\mathbf{XY}} \rangle$$

$$\mathcal{MMD}(\boldsymbol{\mu}, \boldsymbol{\nu}) = \mathcal{E}(\boldsymbol{\mu}, \boldsymbol{\nu}) - \frac{1}{2}(\mathcal{E}(\boldsymbol{\mu}, \boldsymbol{\mu}) + \mathcal{E}(\boldsymbol{\nu}, \boldsymbol{\nu}))$$

$$W_\gamma(\boldsymbol{\mu}, \boldsymbol{\nu}) = \langle P_\gamma, M_{\mathbf{XY}} \rangle$$

$$\bar{W}_\gamma(\boldsymbol{\mu}, \boldsymbol{\nu}) = W_\gamma(\boldsymbol{\mu}, \boldsymbol{\nu}) - \frac{1}{2}(W_\gamma(\boldsymbol{\mu}, \boldsymbol{\mu}) + W_\gamma(\boldsymbol{\nu}, \boldsymbol{\nu}))$$

$$W^p(\boldsymbol{\mu}, \boldsymbol{\nu}) = \langle \boldsymbol{P}^\star, M_{\mathbf{XY}} \rangle$$

Sinkhorn in between W and MMD

$$\mathcal{MMD}(\boldsymbol{\mu},\boldsymbol{\nu}) = \mathcal{E}(\boldsymbol{\mu},\boldsymbol{\nu}) - \frac{1}{2}(\mathcal{E}(\boldsymbol{\mu},\boldsymbol{\mu}) + \mathcal{E}(\boldsymbol{\nu},\boldsymbol{\nu}))$$

$$\gamma \to \infty \quad \uparrow$$

$$\bar{W}_\gamma(\boldsymbol{\mu},\boldsymbol{\nu}) = W_\gamma(\boldsymbol{\mu},\boldsymbol{\nu}) - \frac{1}{2}(W_\gamma(\boldsymbol{\mu},\boldsymbol{\mu}) + W_\gamma(\boldsymbol{\nu},\boldsymbol{\nu}))$$

SINKHORN DIVERGENCE

$$\gamma \to 0 \quad \downarrow$$

$$W^p(\boldsymbol{\mu},\boldsymbol{\nu}) = \langle \boldsymbol{P}^\star, M_{\boldsymbol{XY}} \rangle$$

How to compare them?

i.i.d samples $\boldsymbol{x_1, \ldots, x_n} \sim \boldsymbol{\mu}$, $\boldsymbol{y_1, \ldots, y_m} \sim \boldsymbol{\nu}$,

$$\hat{\boldsymbol{\mu}}_{\boldsymbol{n}} \stackrel{\text{def}}{=} \frac{1}{n}\sum_i \delta_{\boldsymbol{x_i}}, \hat{\boldsymbol{\nu}}_{\boldsymbol{m}} \stackrel{\text{def}}{=} \frac{1}{m}\sum_j \delta_{\boldsymbol{y_j}}$$

Computational properties

Effort to compute/approximate $\boldsymbol{\Delta}(\hat{\boldsymbol{\mu}}_{\boldsymbol{n}}, \hat{\boldsymbol{\nu}}_{\boldsymbol{m}})$?

Statistical properties

$$|\boldsymbol{\Delta}(\boldsymbol{\mu}, \boldsymbol{\nu}) - \boldsymbol{\Delta}(\hat{\boldsymbol{\mu}}_{\boldsymbol{n}}, \hat{\boldsymbol{\nu}}_{\boldsymbol{n}})| \leq f(n)?$$

Sinkhorn in between W and MMD

$$\mathcal{MMD}(\boldsymbol{\mu},\boldsymbol{\nu}) = \mathcal{E}(\boldsymbol{\mu},\boldsymbol{\nu}) - \frac{1}{2}(\mathcal{E}(\boldsymbol{\mu},\boldsymbol{\mu}) + \mathcal{E}(\boldsymbol{\nu},\boldsymbol{\nu}))$$

$(n+m)^2$

$O(1/\sqrt{n})$

$$W^p(\boldsymbol{\mu},\boldsymbol{\nu}) = \langle \boldsymbol{P}^\star, M_{\boldsymbol{XY}} \rangle$$

$O((n+m)nm\log(n+m))$

$O(1/n^{1/d})$

Sinkhorn in between W and MMD

$$\mathcal{MMD}(\boldsymbol{\mu},\boldsymbol{\nu}) = \mathcal{E}(\boldsymbol{\mu},\boldsymbol{\nu}) - \frac{1}{2}(\mathcal{E}(\boldsymbol{\mu},\boldsymbol{\mu}) + \mathcal{E}(\boldsymbol{\nu},\boldsymbol{\nu}))$$

$\boxed{(n+m)^2}$ $\boxed{O(1/\sqrt{n})}$

$$\bar{W}_\gamma(\boldsymbol{\mu},\boldsymbol{\nu}) = W_\gamma(\boldsymbol{\mu},\boldsymbol{\nu}) - \frac{1}{2}(W_\gamma(\boldsymbol{\mu},\boldsymbol{\mu}) + W_\gamma(\boldsymbol{\nu},\boldsymbol{\nu}))$$

$\boxed{O((n+m)^2)}$ $\boxed{O\left(\frac{1}{\gamma^{d/2}\sqrt{n}}\right)}$

[GCBCP'18]

[FSVATP'18]

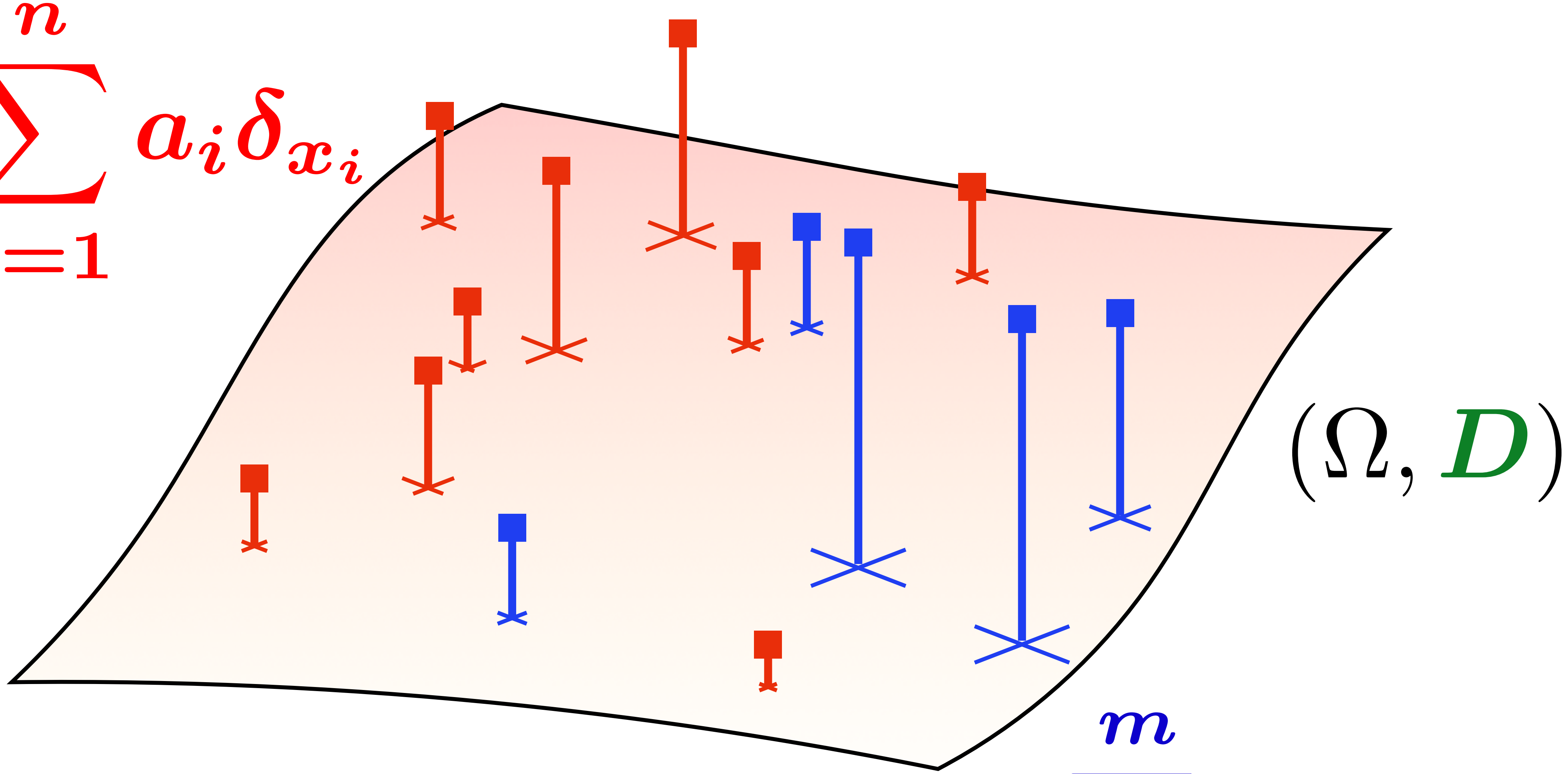
$$W^p(\boldsymbol{\mu},\boldsymbol{\nu}) = \langle \boldsymbol{P}^\star, M_{\boldsymbol{XY}} \rangle$$

$\boxed{O((n+m)nm\log(n+m))}$ $\boxed{O(1/n^{1/d})}$

Differentiability of W

$$W((\boldsymbol{a}, \boldsymbol{X}), (\boldsymbol{b}, \boldsymbol{Y}))$$

$$\boldsymbol{\mu} = \sum_{i=1}^{n} \boldsymbol{a_i \delta_{x_i}}$$

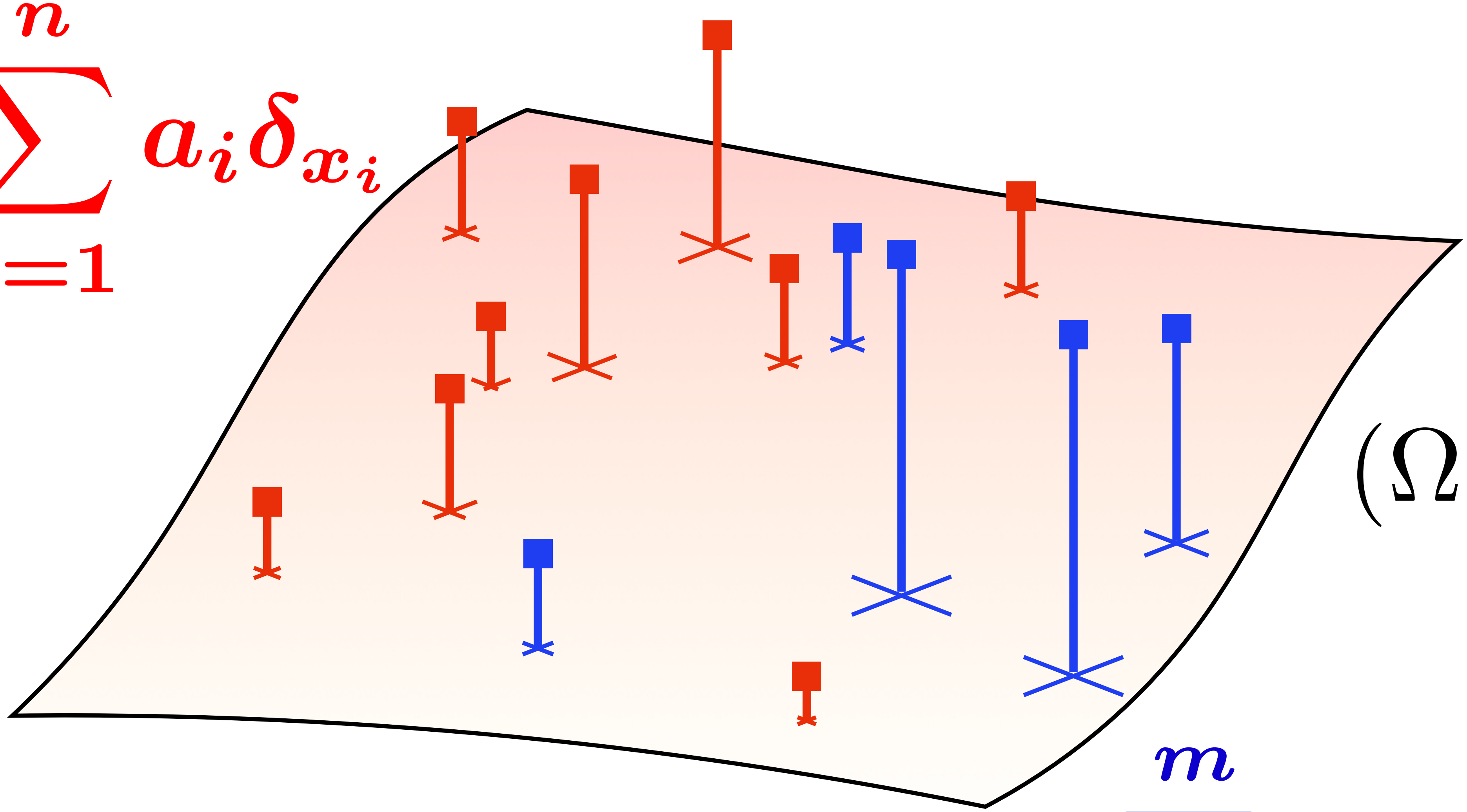


$$\boldsymbol{\nu} = \sum_{j=1}^{m} \boldsymbol{b_j \delta_{y_j}}$$

Differentiability of W

$$W((\boldsymbol{a} + \boldsymbol{\Delta a}, \boldsymbol{X}), (\boldsymbol{b}, \boldsymbol{Y})) = W((\boldsymbol{a}, \boldsymbol{X}), (\boldsymbol{b}, \boldsymbol{Y})) + ??$$

$$\boldsymbol{\mu = \sum_{i=1}^{n} a_i \delta_{x_i}}$$



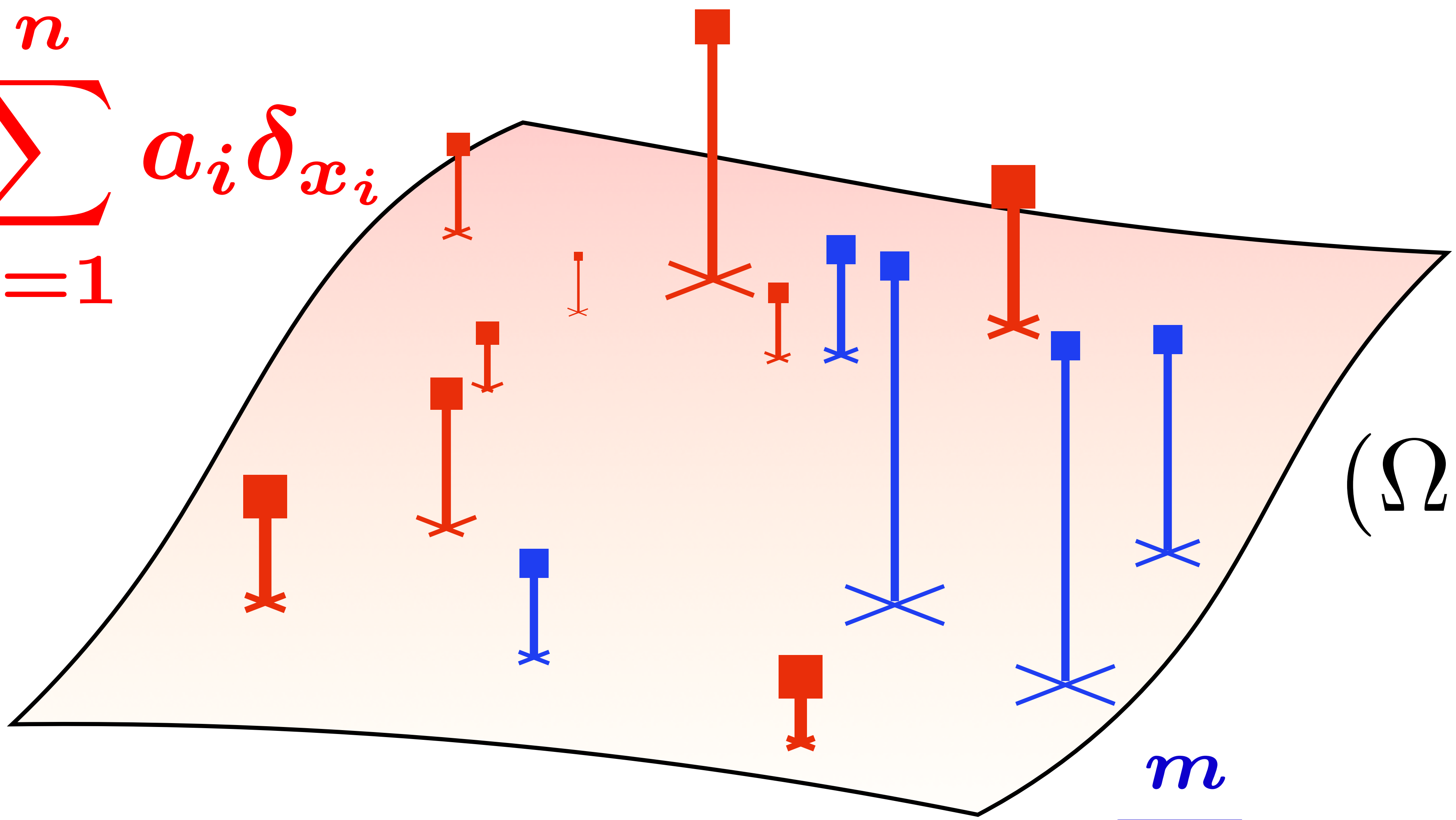
$$(\Omega, \boldsymbol{D})$$

$$\boldsymbol{\nu = \sum_{j=1}^{m} b_j \delta_{y_j}}$$

Differentiability of W

$$W((\boldsymbol{a} + \boldsymbol{\Delta a}, \boldsymbol{X}), (\boldsymbol{b}, \boldsymbol{Y})) = W((\boldsymbol{a}, \boldsymbol{X}), (\boldsymbol{b}, \boldsymbol{Y})) + ??$$

$$\boldsymbol{\mu} = \sum_{i=1}^{n} \boldsymbol{a_i \delta_{x_i}}$$

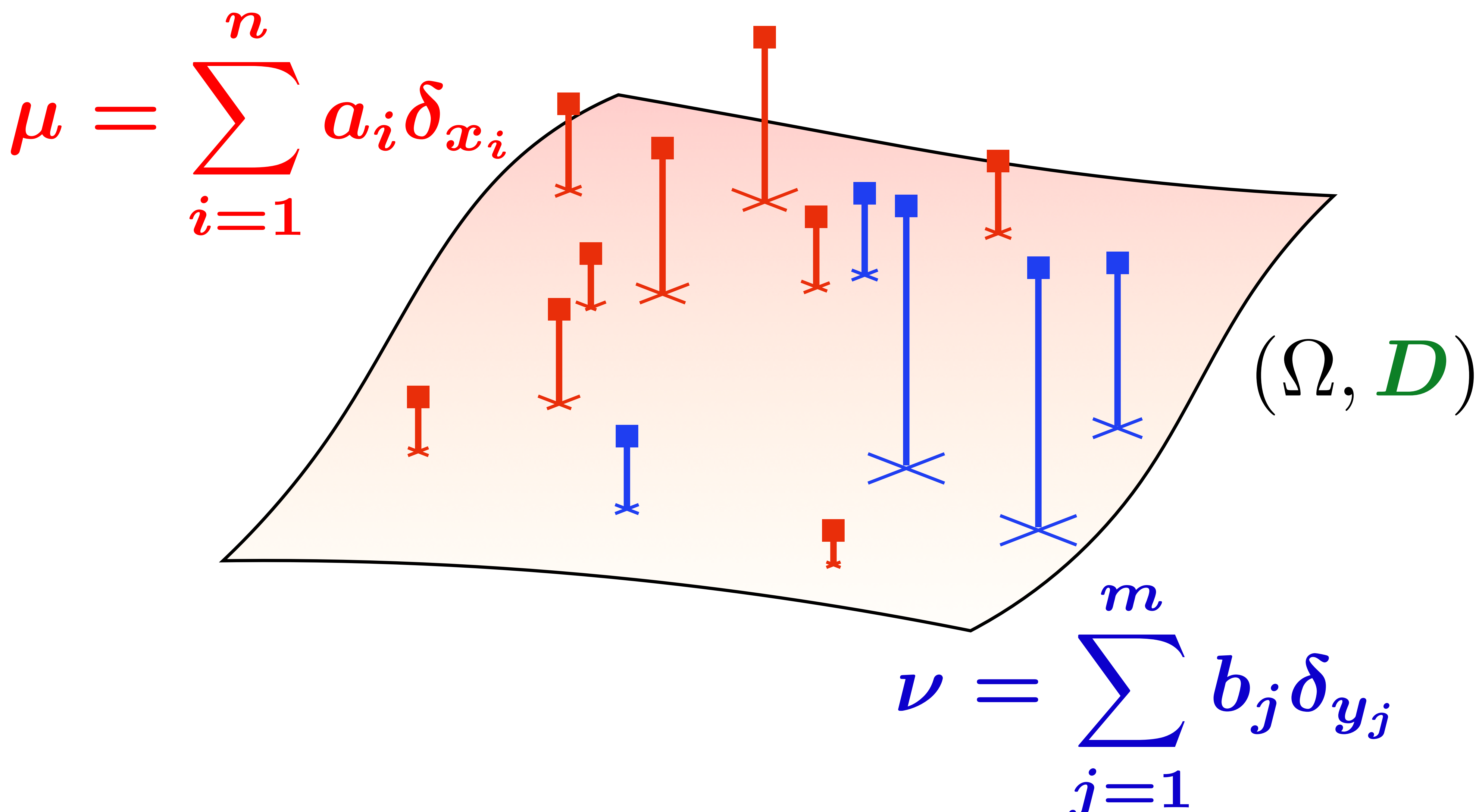


$$\boldsymbol{\nu} = \sum_{j=1}^{m} \boldsymbol{b_j \delta_{y_j}}$$

$$\boldsymbol{a} \leftarrow \boldsymbol{a} + \boldsymbol{\Delta a}$$

Sinkhorn $\dashrightarrow$ *Differentiability*

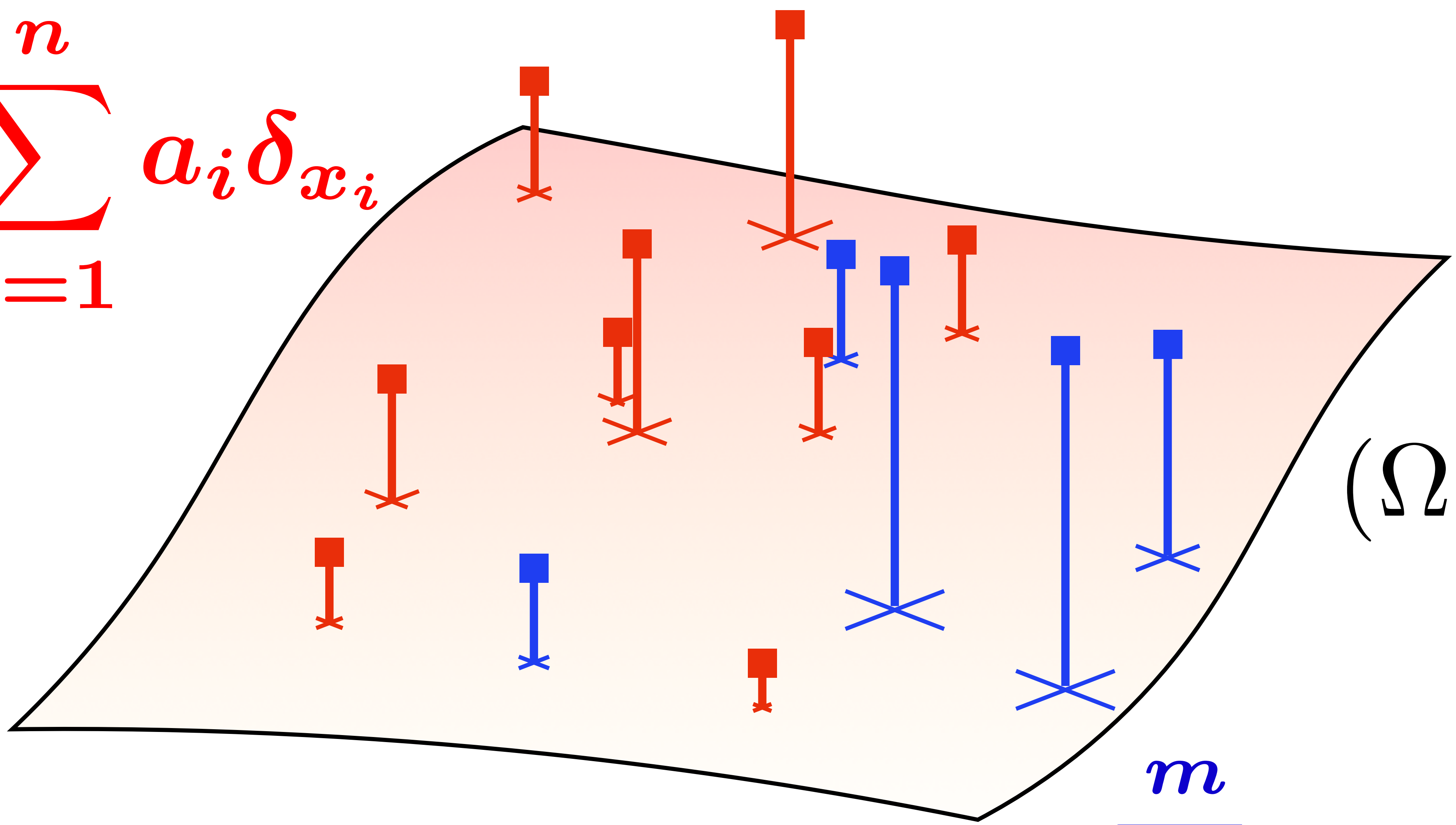
$$W((\boldsymbol{a}, \boldsymbol{X} + \boldsymbol{\Delta X}), (\boldsymbol{b}, \boldsymbol{Y})) = W((\boldsymbol{a}, \boldsymbol{X}), (\boldsymbol{b}, \boldsymbol{Y})) + ??$$



Sinkhorn ⇝ *Differentiability*

$$W((\boldsymbol{a}, \boldsymbol{X} + \boldsymbol{\Delta X}), (\boldsymbol{b}, \boldsymbol{Y})) = W((\boldsymbol{a}, \boldsymbol{X}), (\boldsymbol{b}, \boldsymbol{Y})) + ??$$

$$\boldsymbol{\mu} = \sum_{i=1}^{n} \boldsymbol{a_i \delta_{x_i}}$$



$(\Omega, \boldsymbol{D})$

$$\boldsymbol{X} \leftarrow \boldsymbol{X} + \boldsymbol{\Delta X}$$

$$\boldsymbol{\nu} = \sum_{j=1}^{m} \boldsymbol{b_j \delta_{y_j}}$$

Sinkhorn: A Programmer View

Def. For $L \geq 1$, define

$$W_L(\boldsymbol{\mu}, \boldsymbol{\nu}) \stackrel{\text{def}}{=} \langle \boldsymbol{P_L}, M_{\boldsymbol{XY}} \rangle,$$

where $\boldsymbol{P_L} \stackrel{\text{def}}{=} \mathbf{diag}(\boldsymbol{u_L}) K \mathbf{diag}(\boldsymbol{v_L}),$

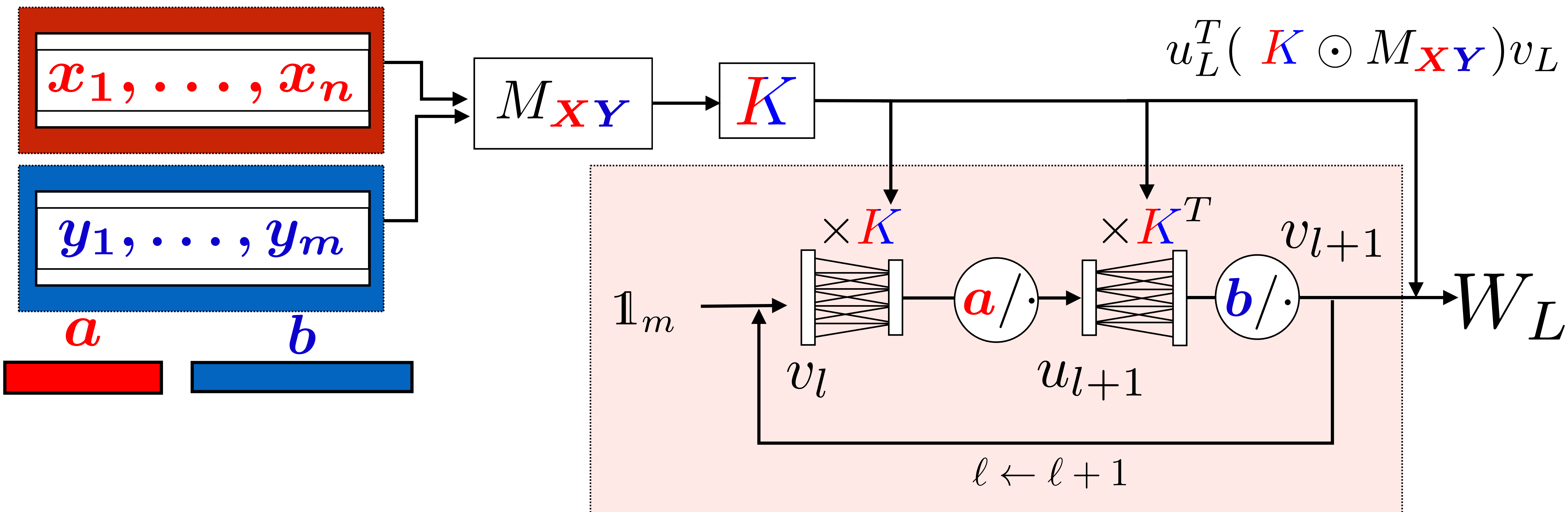
$$\boldsymbol{v_0} = \mathbf{1}_m; l \geq 0, \boldsymbol{u_l} \stackrel{\text{def}}{=} \boldsymbol{a}/K\boldsymbol{v_l}, \boldsymbol{v_{l+1}} \stackrel{\text{def}}{=} \boldsymbol{b}/K^T\boldsymbol{u_l}.$$

Prop. $\frac{\partial W_L}{\partial \boldsymbol{X}}, \frac{\partial W_L}{\partial \boldsymbol{a}}$ can be computed recursively, in $O(L)$ kernel $K\times$vector products.

Sinkhorn: A Programmer View

Def. For $L \geq 1$, define

$$W_L(\boldsymbol{\mu}, \boldsymbol{\nu}) \stackrel{\text{def}}{=} \langle \boldsymbol{P_L}, M_{\boldsymbol{XY}} \rangle,$$



Sinkhorn $\ell = 1, \dots, L-1$

Sinkhorn: A Programmer View

Def. For $L \geq 1$, define

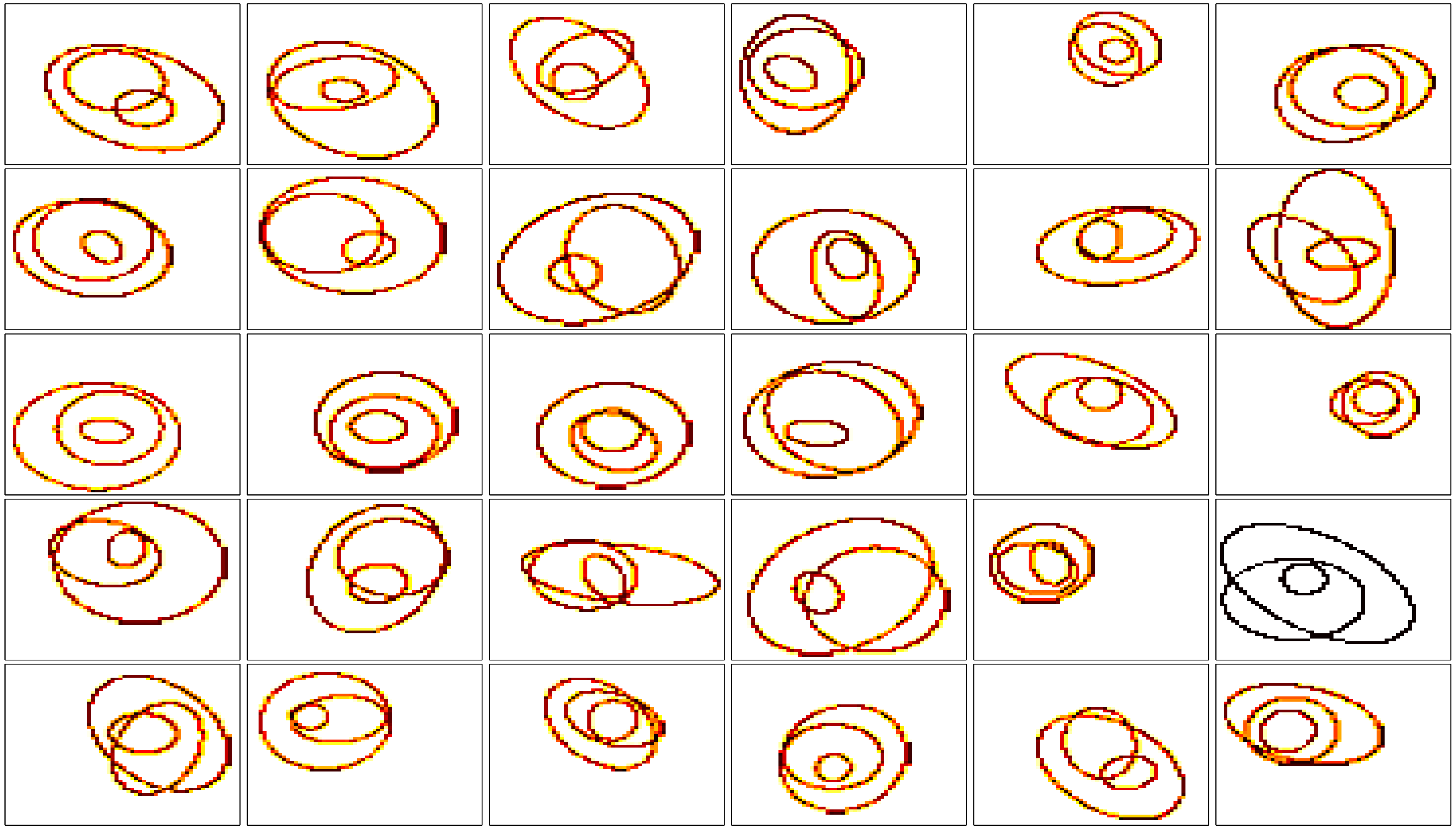
$$W_L(\boldsymbol{\mu}, \boldsymbol{\nu}) \stackrel{\text{def}}{=} \langle \boldsymbol{P_L}, M_{\boldsymbol{XY}} \rangle,$$

Prop. $\frac{\partial W_L}{\partial \boldsymbol{X}}, \frac{\partial W_L}{\partial \boldsymbol{a}}$ can be computed recursively, in $O(L)$ kernel $K \times$vector products.

[Hashimoto'16] [Bonneel'16][Shalit'16]

Primal Descent on Regularized W

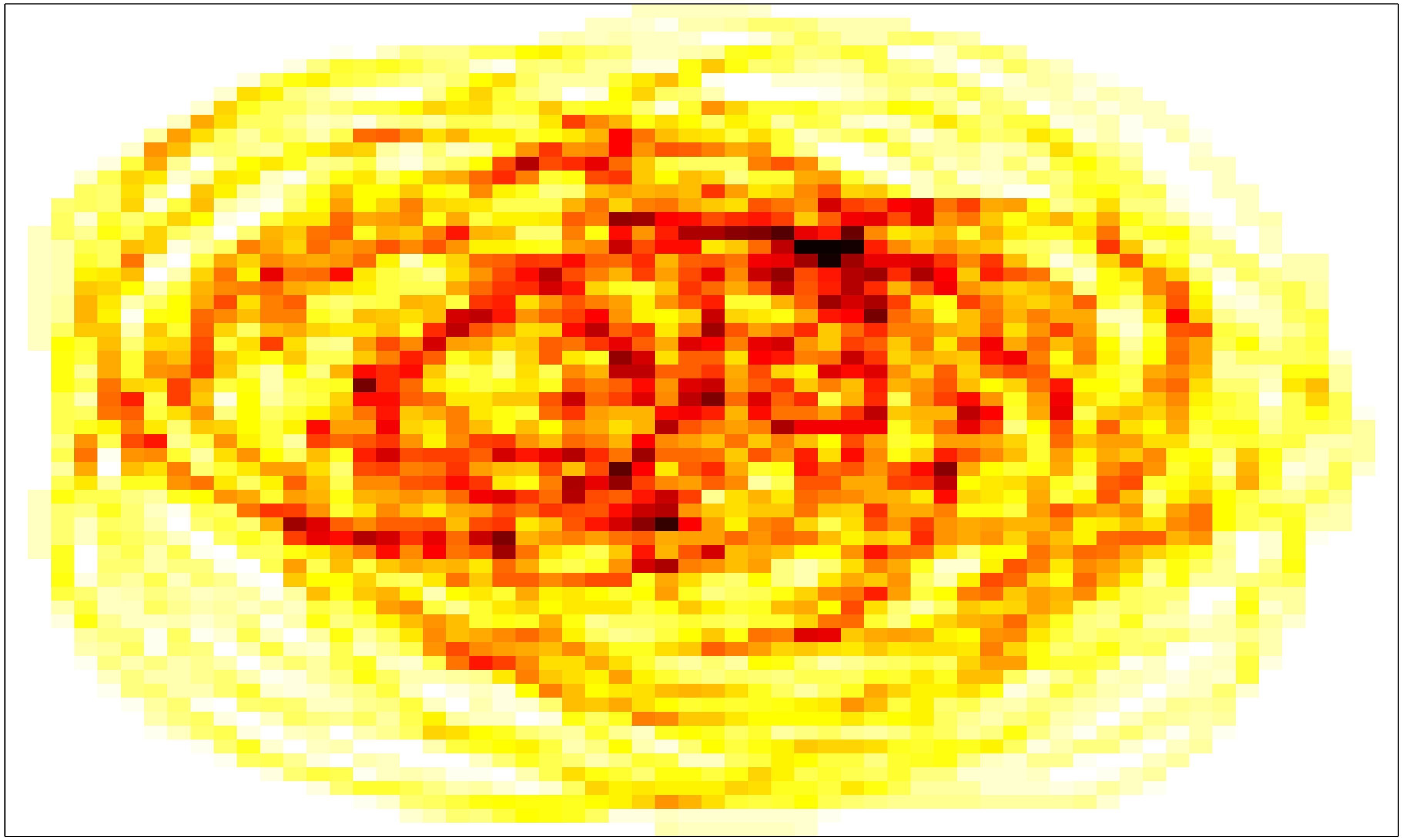
$$\min_{\boldsymbol{a} \in \Sigma_{h \times h}} \sum_{i=1}^{N} \lambda_i W_{\gamma}(\boldsymbol{a}, \boldsymbol{b_i})$$



[Cuturi'14]

Primal Descent on Regularized W

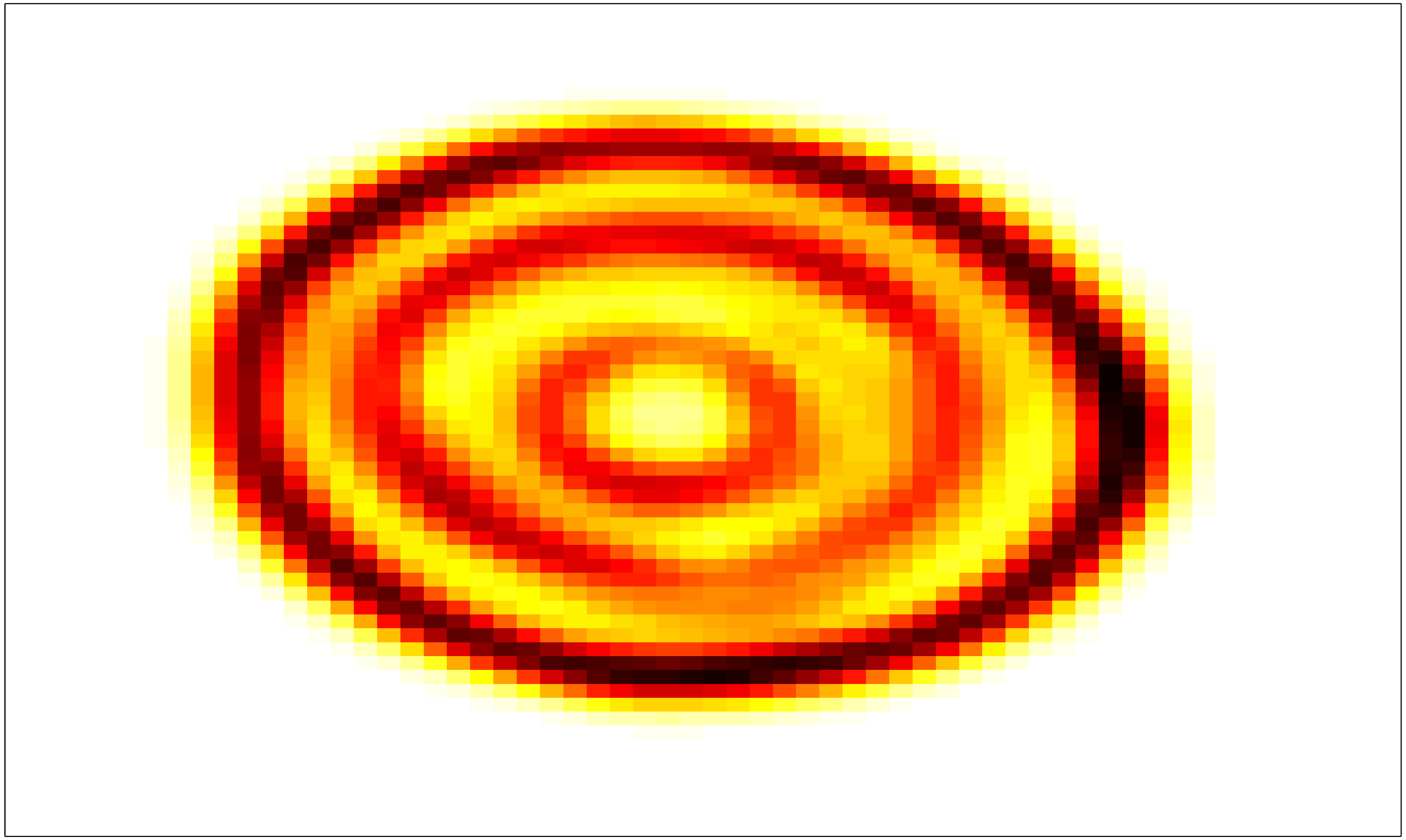
$$\min_{\boldsymbol{a} \in \Sigma_{h \times h}} \sum_{i=1}^{N} \lambda_i W_{\gamma}(\boldsymbol{a}, \boldsymbol{b_i})$$



[Cuturi'14]

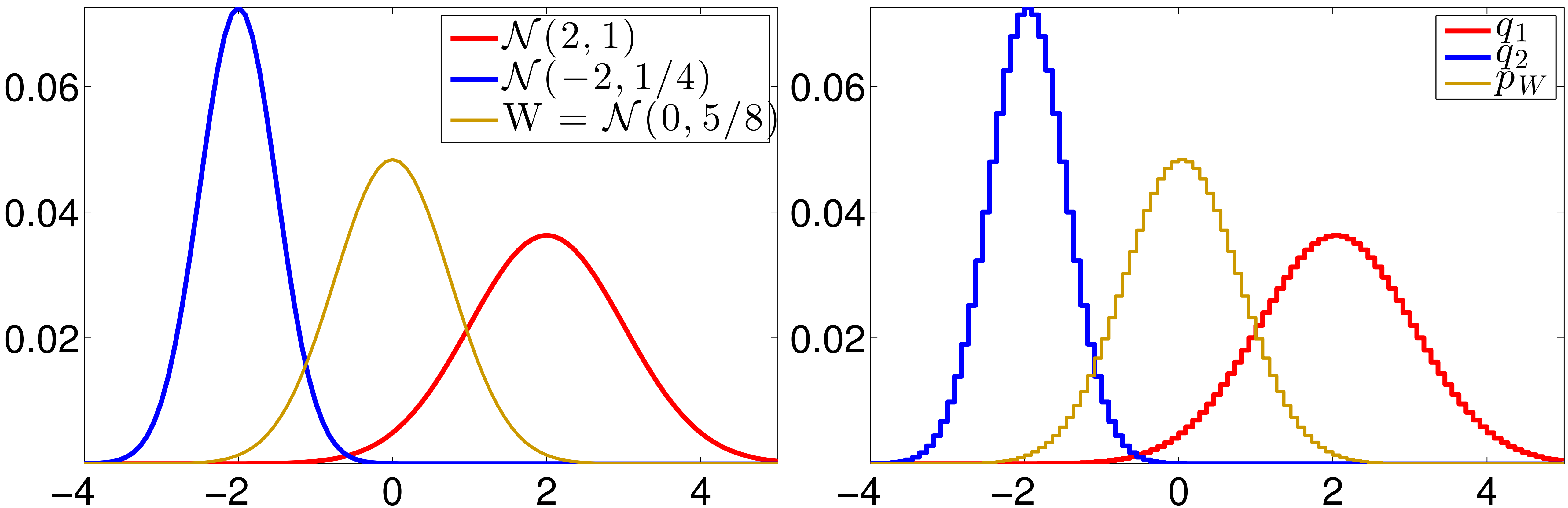
Primal Descent on Regularized W

$$\min_{\boldsymbol{a}\in\Sigma_{h\times h}} \sum_{i=1}^{N} \lambda_i W_{\gamma}(\boldsymbol{a}, \boldsymbol{b_i})$$

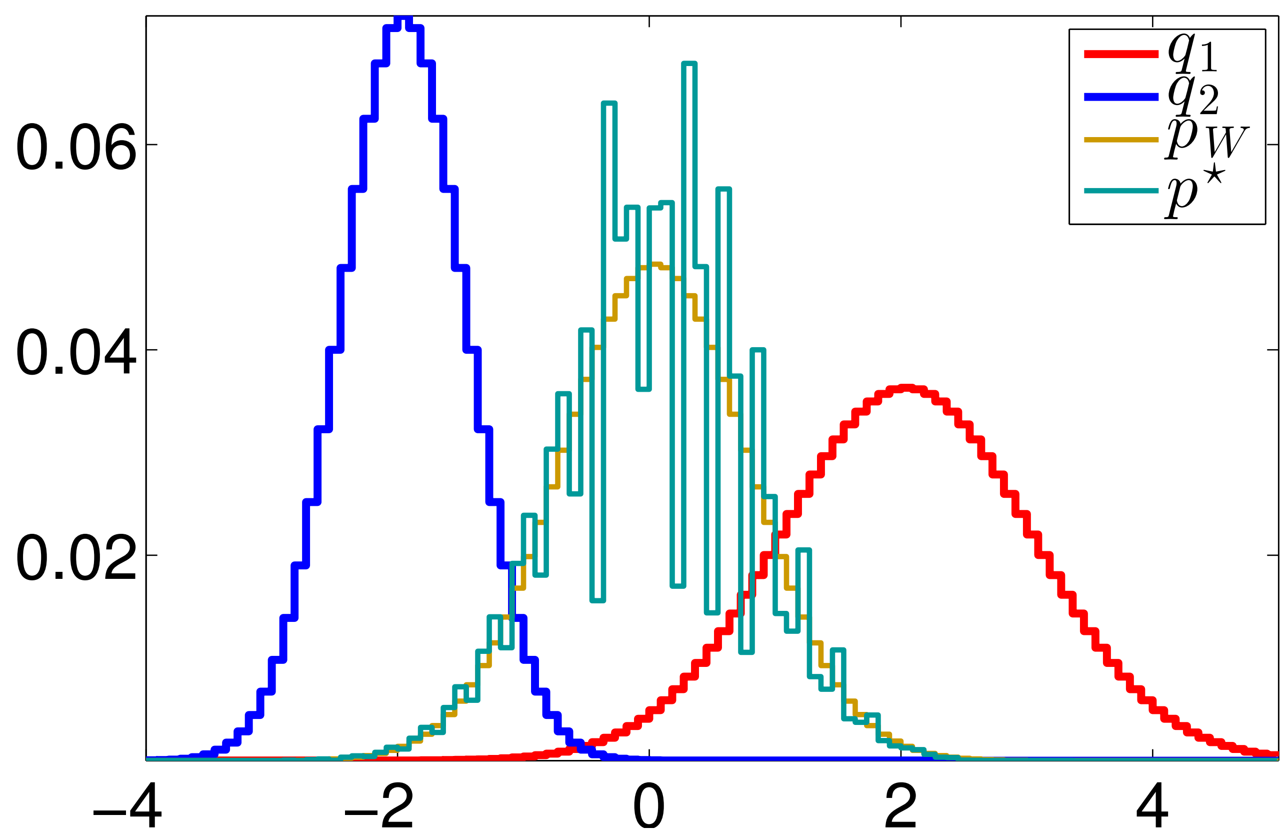


[Cuturi'14]

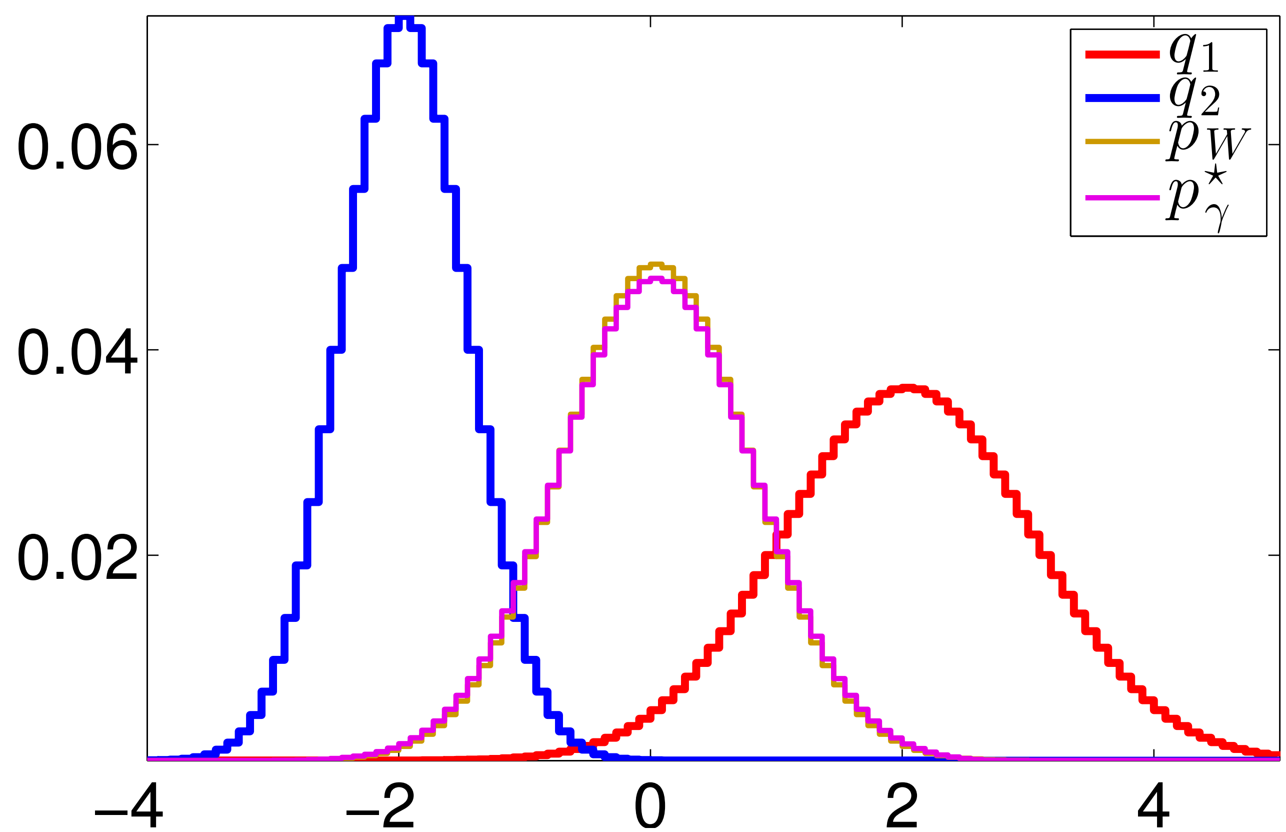
On Regularizing or Not



On Regularizing or Not

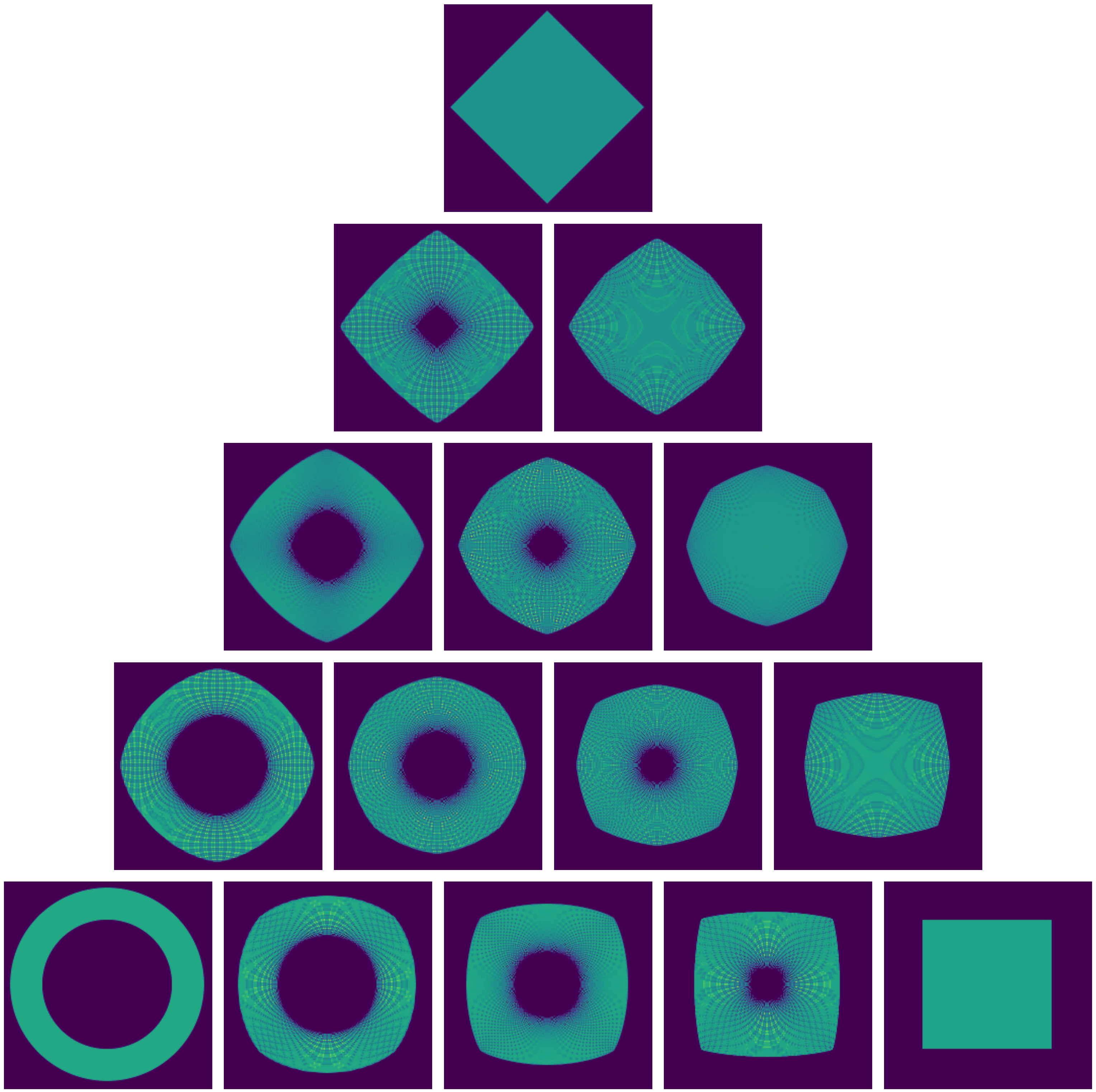


True barycenter



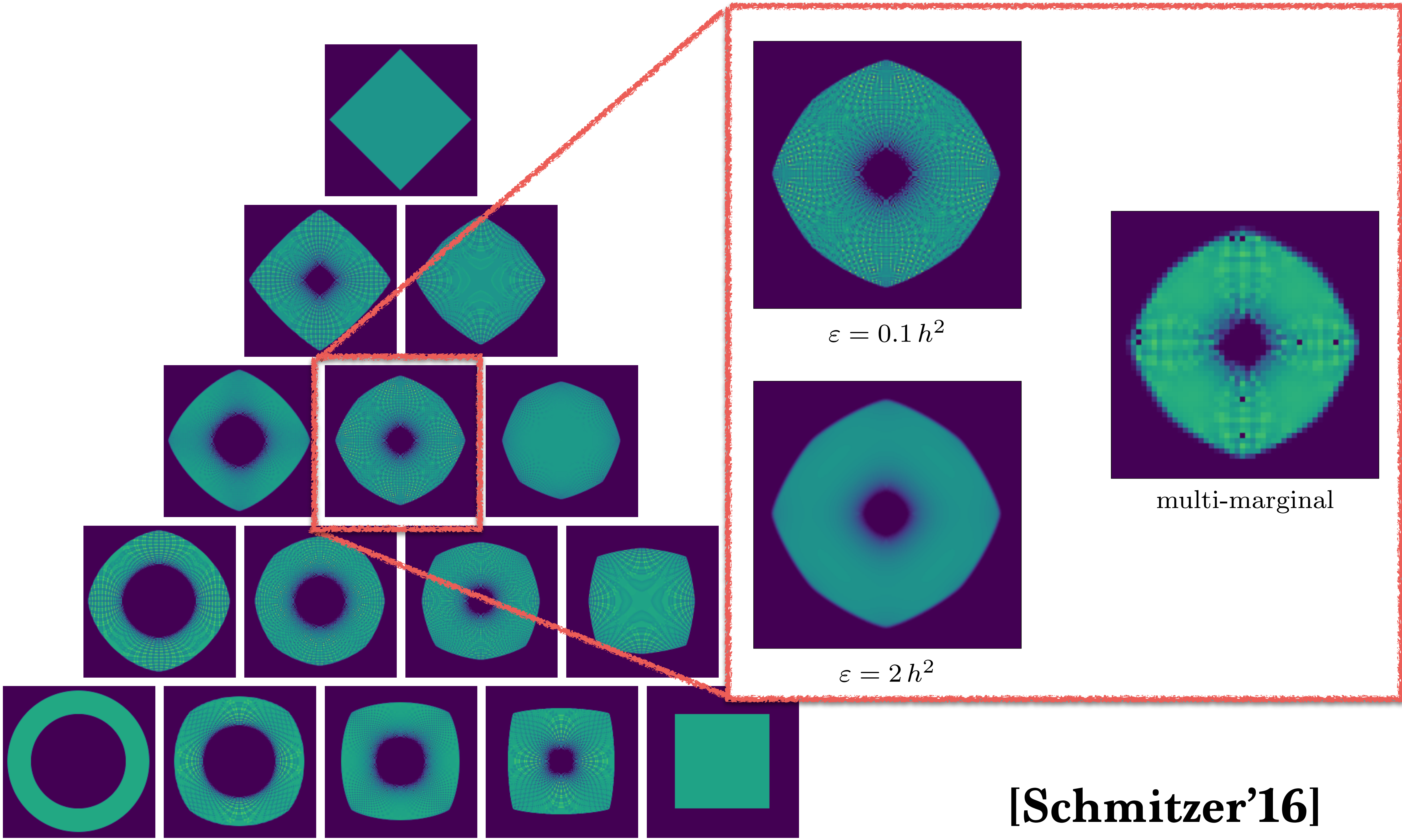
Barycenter using regularized OT

On Regularizing or Not



[Schmitzer'16]

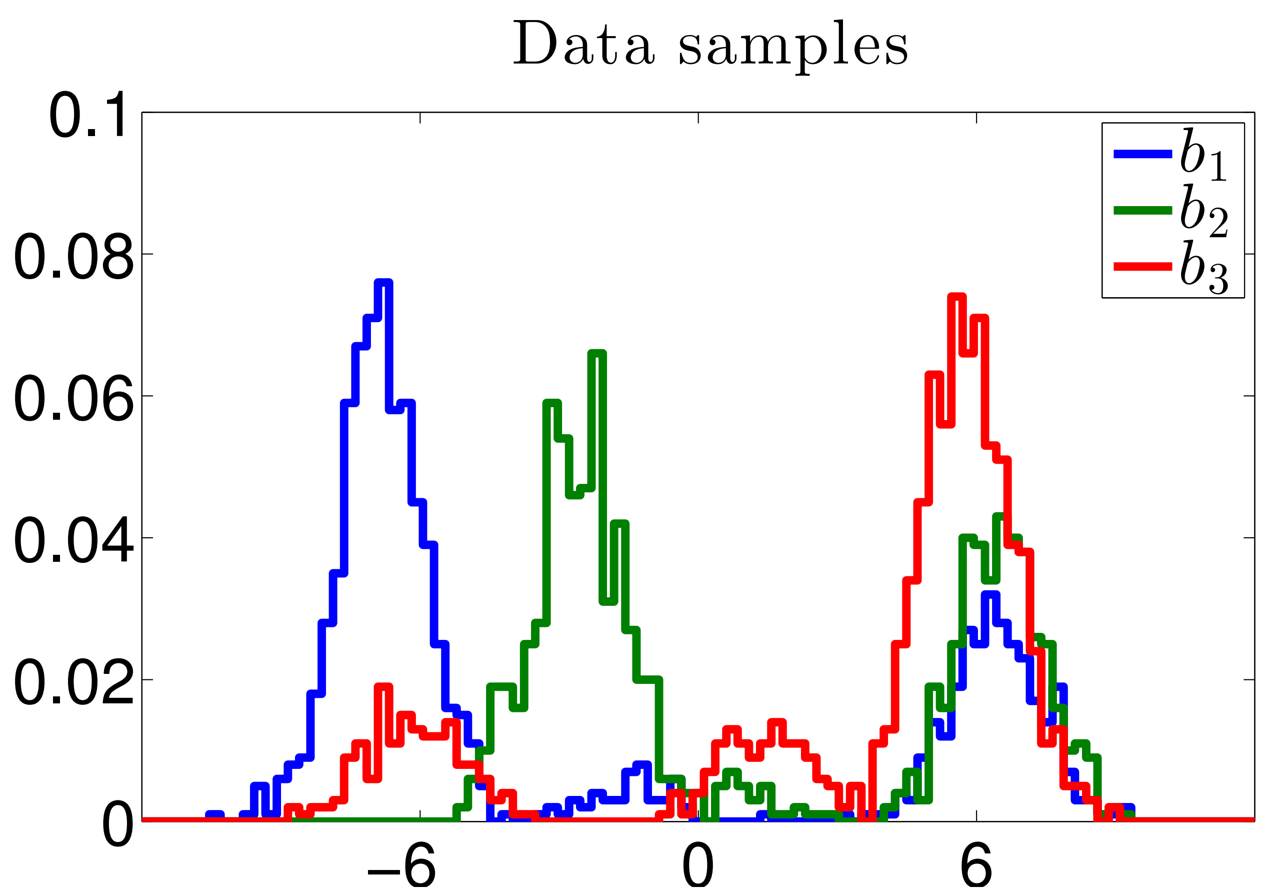
On Regularizing or Not

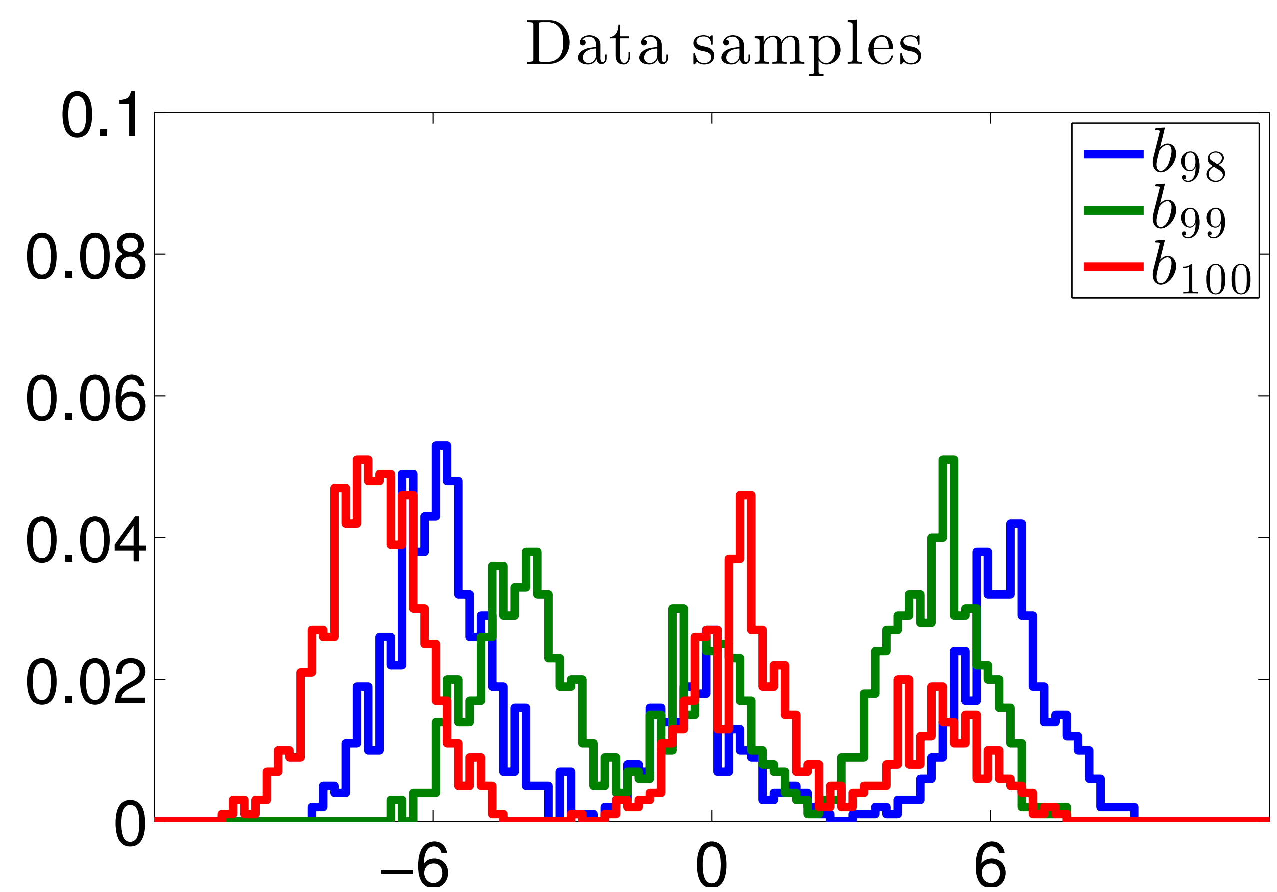


[Schmitzer'16]

Dictionary Learning

$$\min_{\boldsymbol{A}\in(\Sigma_n)^K,\boldsymbol{\Lambda}\in(\Sigma_K)^N} \sum_{i=1}^N W\left(\boldsymbol{b_i}, \sum_{k=1}^K \boldsymbol{\Lambda_k^i a_k}\right)$$

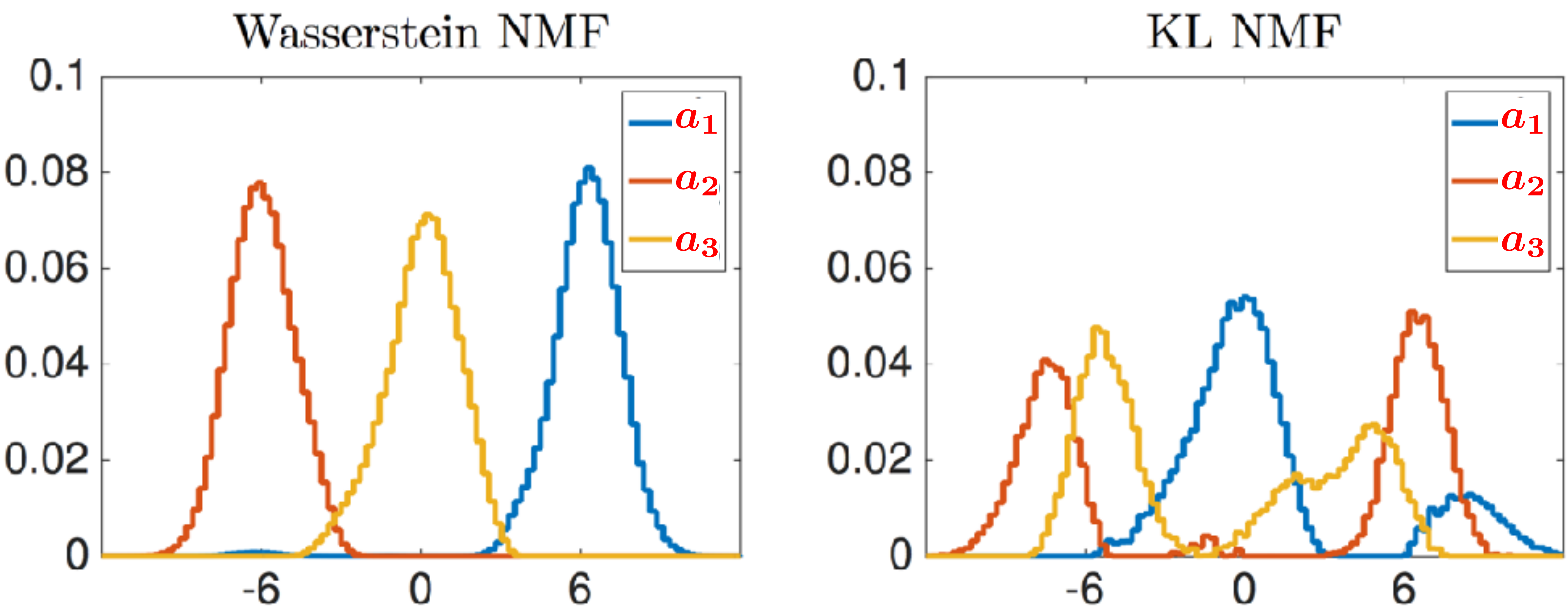




[Sandler'11] [Zen'14] [Rolet'16]

Dictionary Learning

$$\min_{A \in (\Sigma_n)^K, \Lambda \in (\Sigma_K)^N} \sum_{i=1}^{N} W\left(b_i, \sum_{k=1}^{K} \Lambda_k^i a_k\right)$$



[Sandler'11] [Zen'14] [Rolet'16]

OT Dictionary Learning

- **[Hoffman'98]** proposed to learn dictionaries (topics) for text, seen as histograms-of-words.

$$\Omega = \{\text{words}\}, \quad |\Omega| \approx 13,000$$

- Vector embeddings for words **[Mikolov'13] [Pennington'14]** defines geometry:

$$\boldsymbol{D}(\text{public}, \text{car}) = \|x_{\text{public}} - x_{\text{car}}\|^2$$

- Data: 7,034 Reuters, 737 BBC sports news articles

Topic Models

[Rolet'16]

Inverse Wasserstein Problems

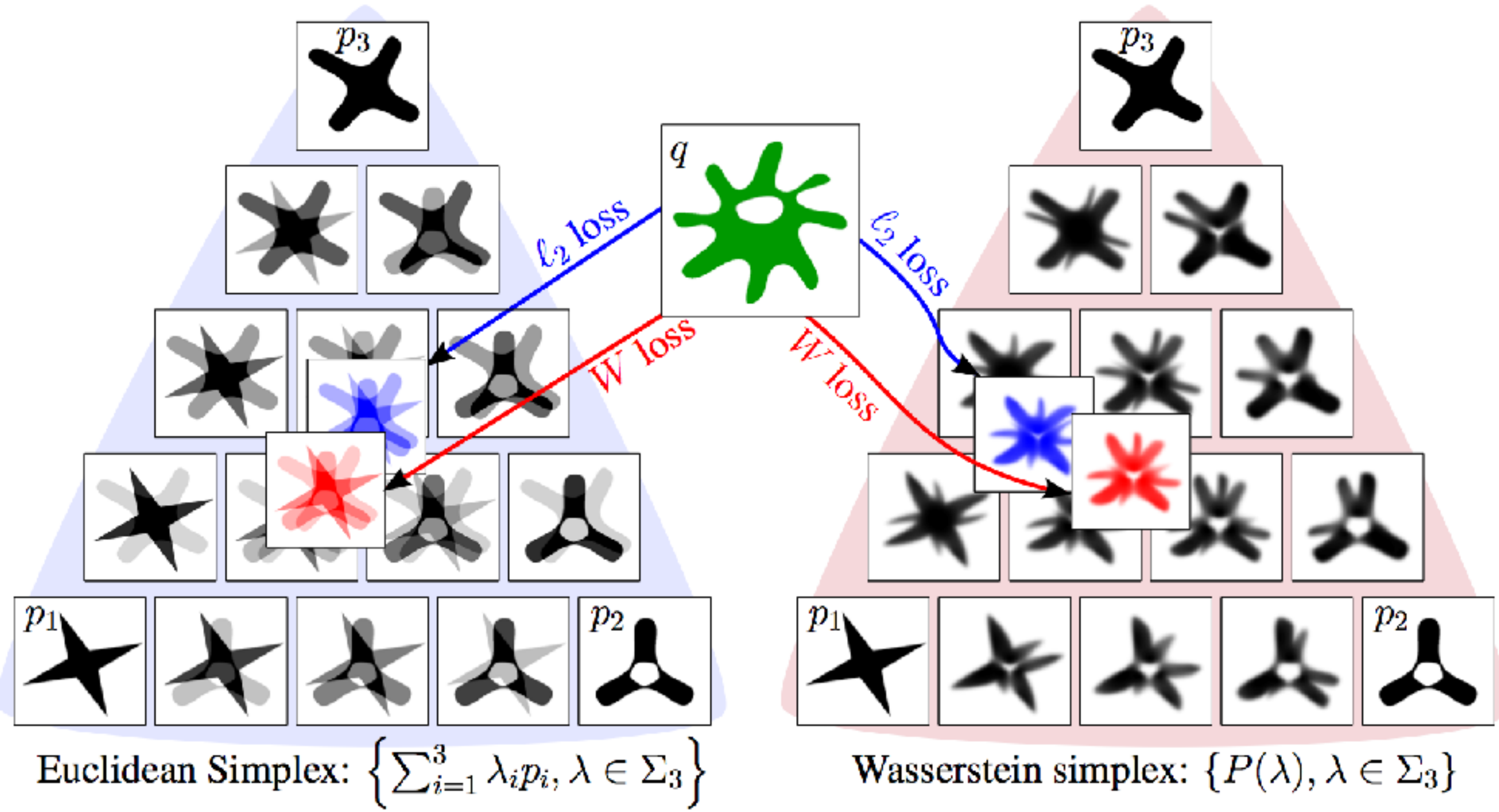
- consider Barycenter operator:

$$\boldsymbol{b}(\lambda) \stackrel{\text{def}}{=} \underset{\boldsymbol{a}}{\operatorname{argmin}} \sum_{i=1}^{N} \lambda_i W_\gamma(\boldsymbol{a}, \boldsymbol{b_i})$$

- address now **Wasserstein inverse problems**:

$$\text{Given } \boldsymbol{a}, \ \text{ find } \ \underset{\lambda \in \Sigma_N}{\operatorname{argmin}} \, \mathcal{E}(\lambda) \stackrel{\text{def}}{=} \operatorname{Loss}(\boldsymbol{a}, \boldsymbol{b}(\lambda))$$

Wasserstein Inverse Problems



Barycenters = Fixed Points

Prop. **[BCCNP'15]** Consider $\boldsymbol{B} \in \Sigma_d^N$ and let $\boldsymbol{U_0} = \mathbf{1}_{\boldsymbol{d\times N}}$, and then for $l \geq 0$:

$$\boldsymbol{b}^l \overset{\text{def}}{=} \exp\left(\log\left(K^T \boldsymbol{U_l}\right)\lambda\right); \begin{cases} \boldsymbol{V_{l+1}} \overset{\text{def}}{=} \dfrac{\boldsymbol{b}^l \mathbf{1}_N^T}{K^T \boldsymbol{U_l}}, \\ \boldsymbol{U_{l+1}} \overset{\text{def}}{=} \dfrac{\boldsymbol{B}}{K \boldsymbol{V_{l+1}}}. \end{cases}$$

Using Truncated Barycenters

- instead of using the exact barycenter

$$\underset{\lambda \in \Sigma_N}{\operatorname{argmin}} \, \mathcal{E}(\lambda) \overset{\text{def}}{=} \text{Loss}(\boldsymbol{a}, \boldsymbol{b}(\lambda))$$

- use instead the L-iterate barycenter

$$\underset{\lambda \in \Sigma_N}{\operatorname{argmin}} \, \mathcal{E}^{(L)}(\lambda) \overset{\text{def}}{=} \text{Loss}(\boldsymbol{a}, \boldsymbol{b}^{(L)}(\lambda))$$

- Differente using **the chain rule.**

$$\nabla \mathcal{E}^{(L)}(\lambda) = [\partial \boldsymbol{b}^{(L)}]^T(\boldsymbol{g}), \;\; \boldsymbol{g} \overset{\text{def}}{=} \nabla \text{Loss}(\boldsymbol{a}, \cdot)|_{\boldsymbol{b}^{(L)}(\lambda)}.$$

Gradient / Barycenter Computation

function SINKHORN-DIFFERENTIATE($(p_s)_{s=1}^S, q, \lambda$)

$\forall\, s, b_s^{(0)} \leftarrow \mathbb{1}$

$(w, r) \leftarrow (0^S, 0^{S\times N})$

for $\ell = 1, 2, \dots, L$ *// Sinkhorn loop*

$\forall\, s, \varphi_s^{(\ell)} \leftarrow K^\top \frac{p_s}{K b_s^{(\ell-1)}}$

$p \leftarrow \prod_s \left(\varphi_s^{(\ell)}\right)^{\lambda_s}$

$\forall\, s, b_s^{(\ell)} \leftarrow \frac{p}{\varphi_s^{(\ell)}}$

$g \leftarrow \nabla \mathcal{L}(p, q) \odot p$

for $\ell = L, L-1, \dots, 1$ *// Reverse loop*

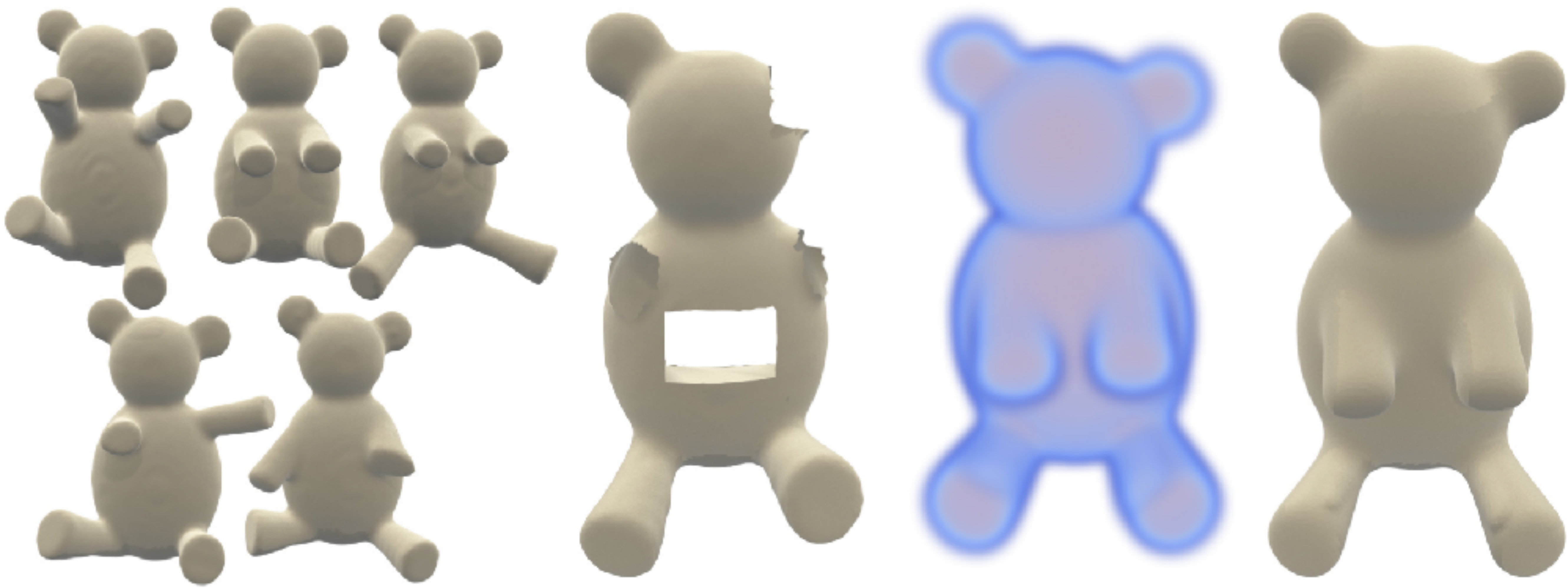
$\forall\, s, w_s \leftarrow w_s + \langle \log \varphi_s^{(\ell)}, g \rangle$

$\forall\, s, r_s \leftarrow -K^\top (K(\frac{\lambda_s g - r_s}{\varphi_s^{(\ell)}}) \odot \frac{p_s}{(K b_s^{(\ell-1)})^2}) \odot b_s^{(\ell-1)}$

$g \leftarrow \sum_s r_s$

return $P^{(L)}(\lambda) \leftarrow p, \nabla \mathcal{E}_L(\lambda) \leftarrow w$

Application: Volume Reconstruction



Shape database $(p_1, \ldots, p_5)$ | Input shape q | Projection $P(\lambda)$ | Iso-surface

[Bonneel'16]

Application: Color Grading

Application: Color Grading

$\lambda_0 = 0.03$

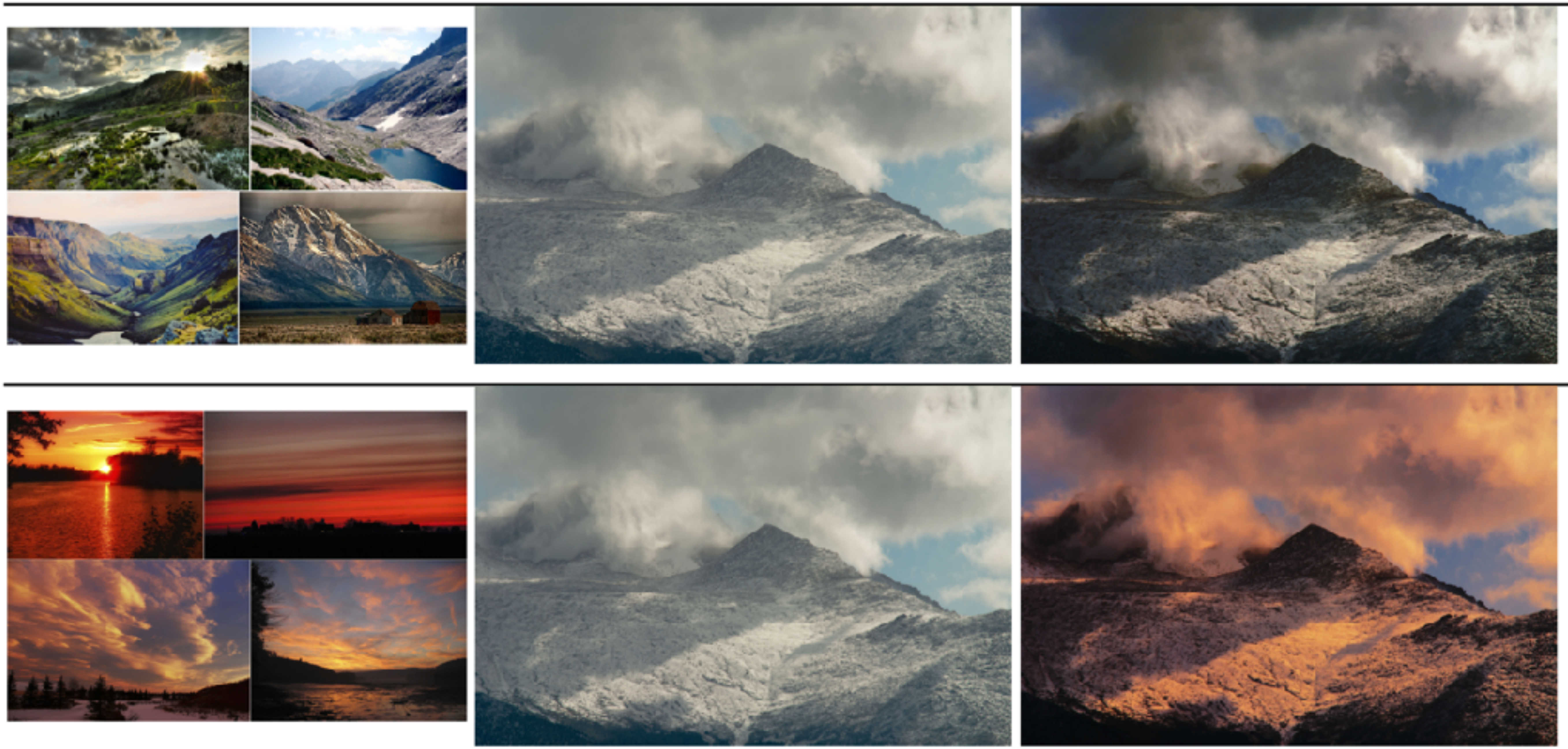
$\lambda_1 = 0.12$

$\lambda_2 = 0.40$

$\lambda_3 = 0.43$

Application: Color Grading

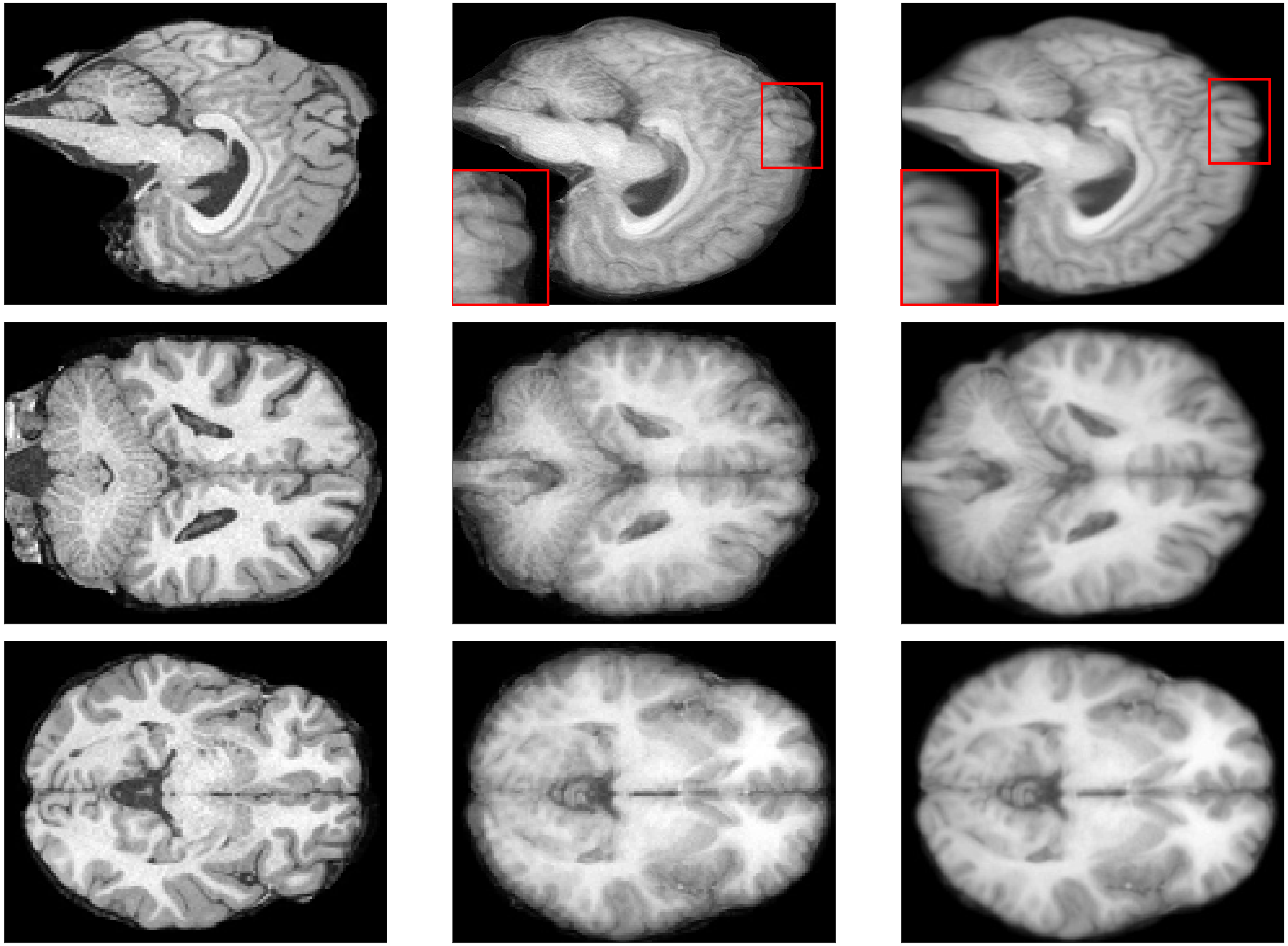
Application: Color Grading



Wasserstein Barycentric Coordinates: Histogram Regression using Optimal Transport, **SIGGRAPH'16**

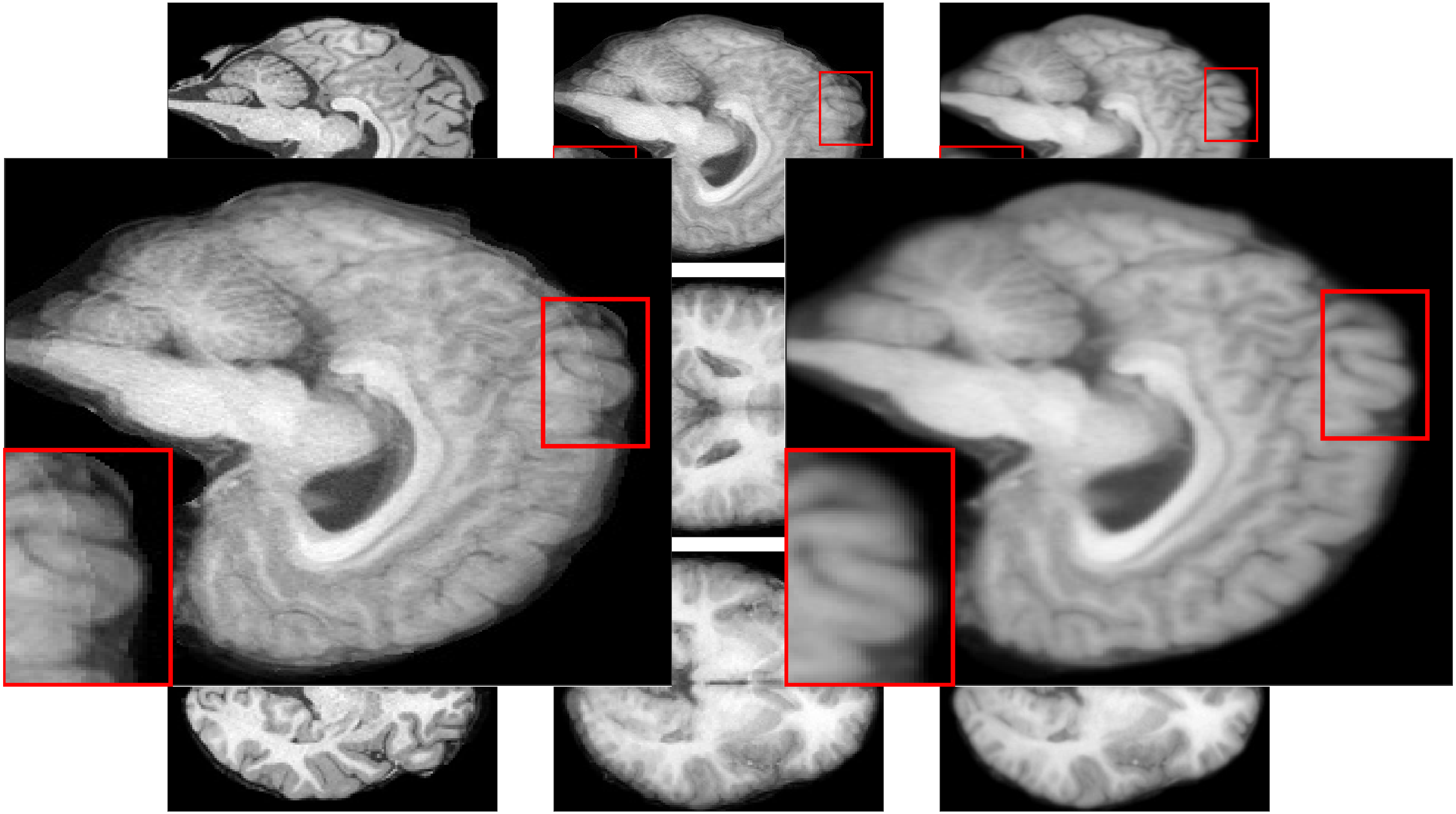
[BPC'16]

Application: Brain Mapping



Original | Euclidean projection | Wasserstein projection

Application: Brain Mapping



Original

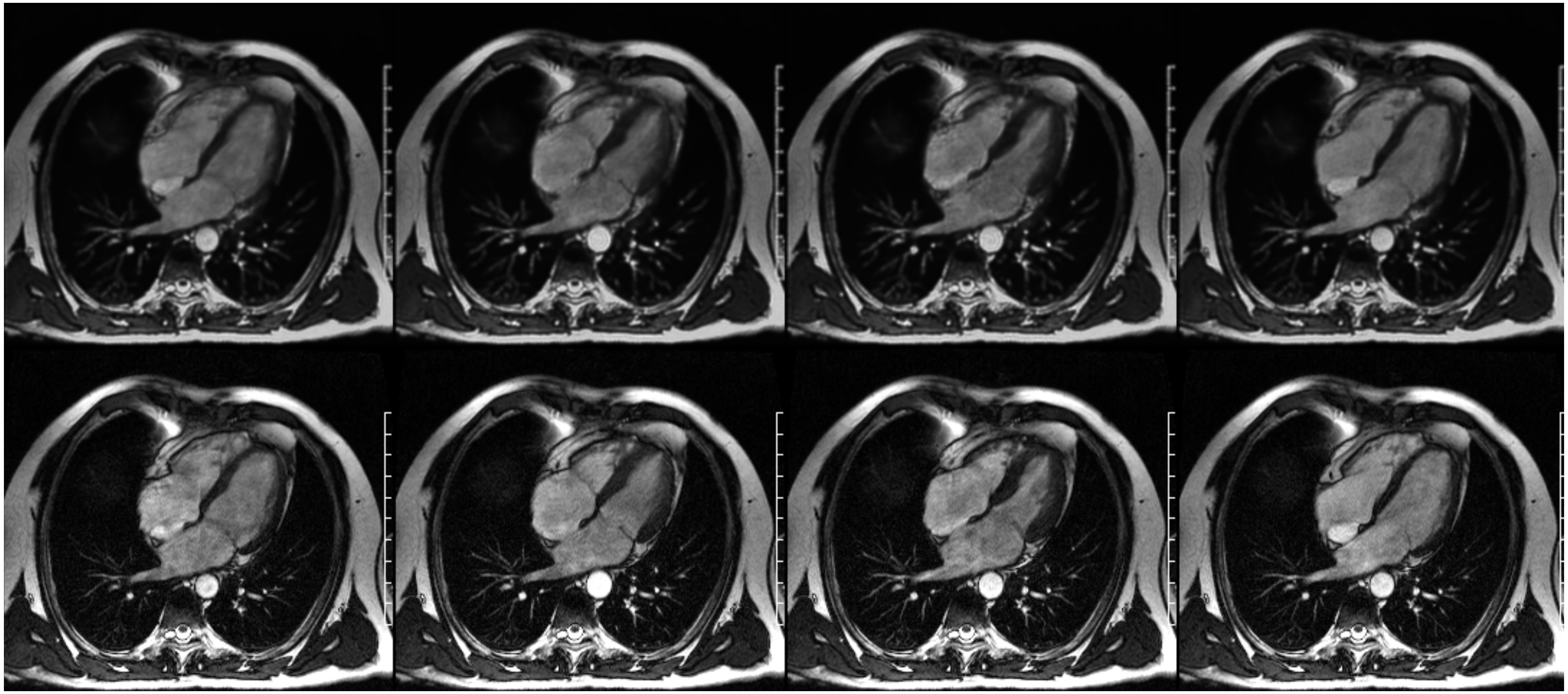
Euclidean projection

Wasserstein projection

end-to-end W Dictionary Learning

$$\min_{\boldsymbol{A}\in(\Sigma_n)^K \boldsymbol{\Lambda}\in(\Sigma_K)^N} \sum_{i=1}^{N} \mathcal{L}\left(\boldsymbol{b_i}, \boldsymbol{a}(\boldsymbol{\lambda_i})\right)$$

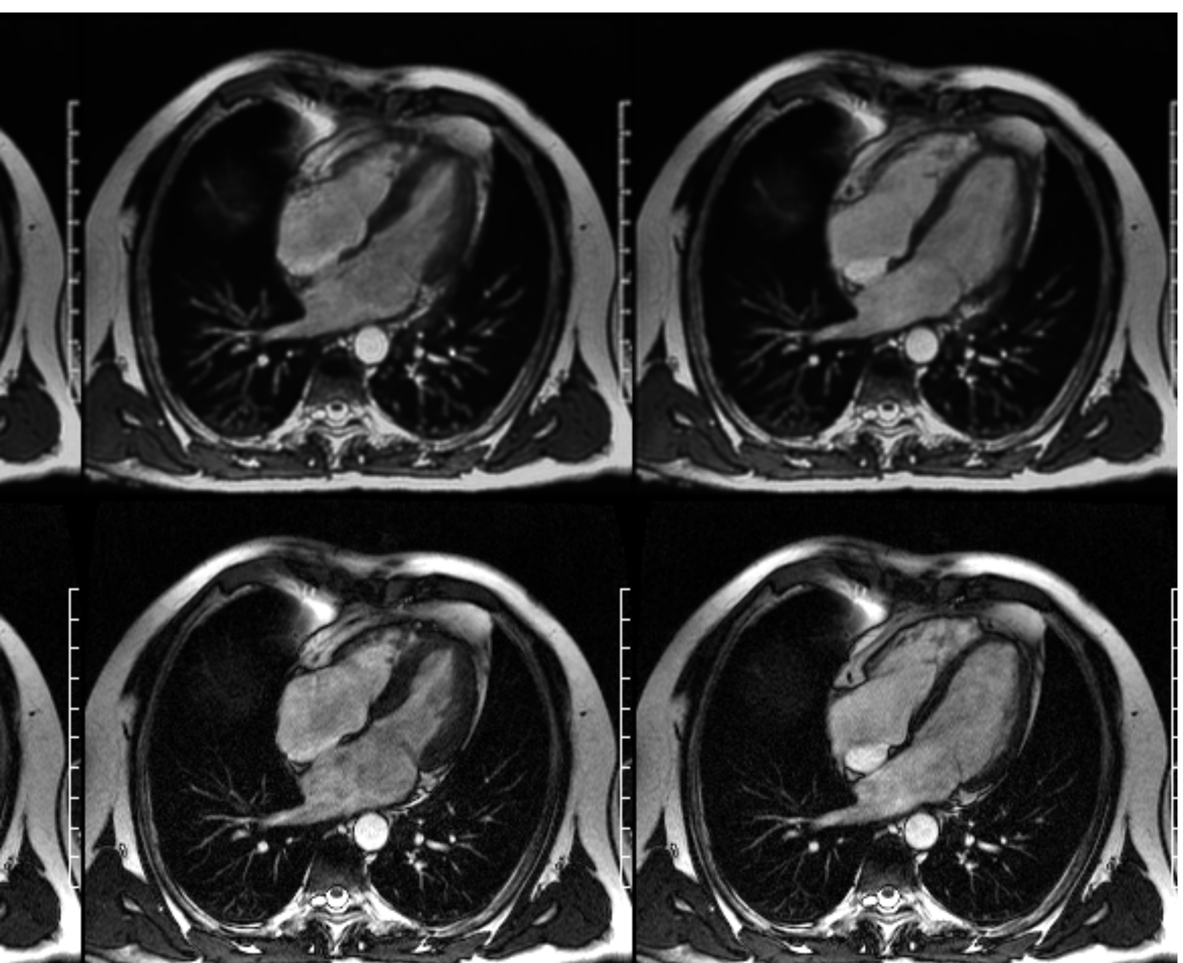
[Schmitz'18]

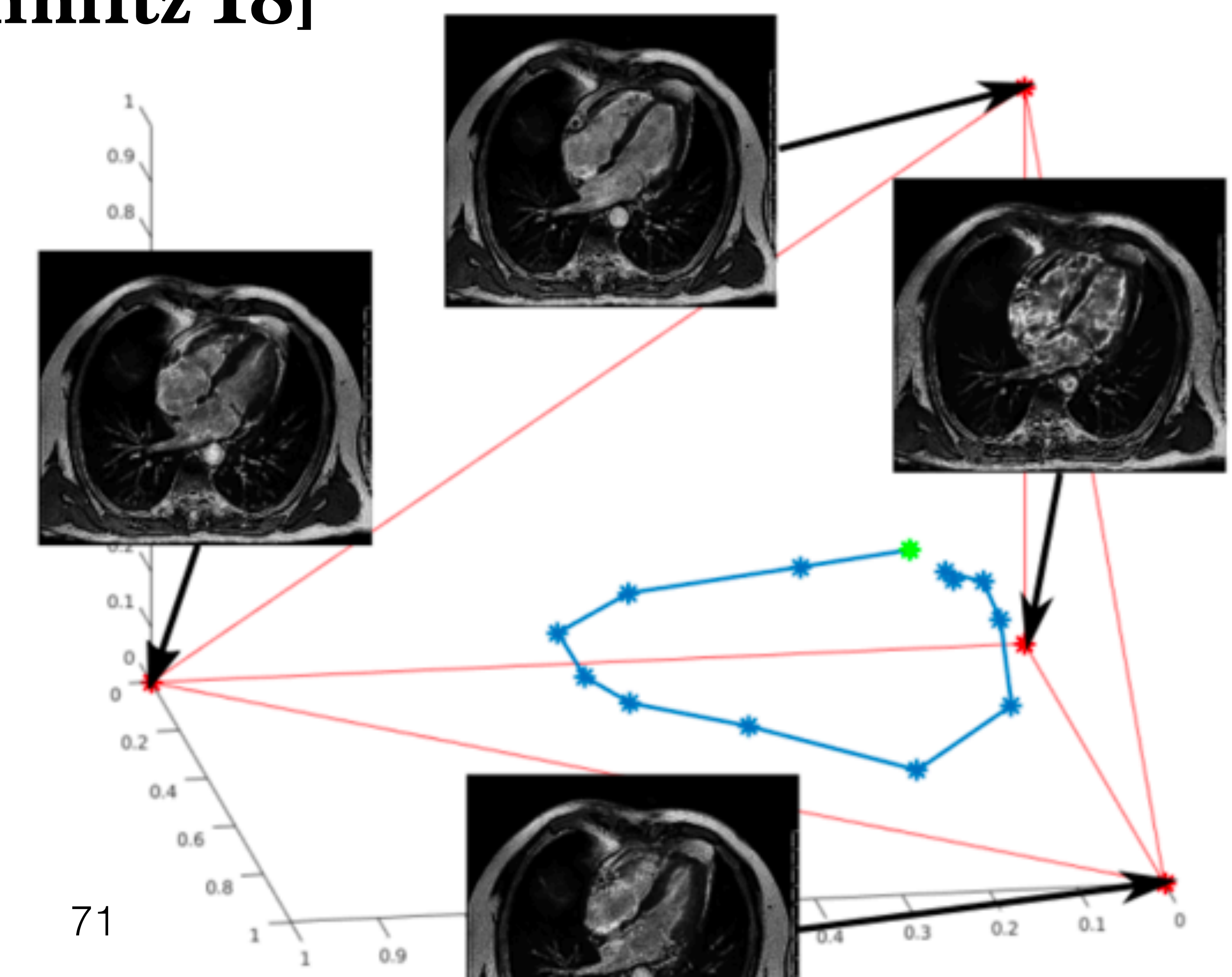


end-to-end W Dictionary Learning

$$\min_{\boldsymbol{A}\in(\Sigma_n)^K \boldsymbol{\Lambda}\in(\Sigma_K)^N} \sum_{i=1}^{N} \mathcal{L}\left(\boldsymbol{b_i}, \boldsymbol{a}(\boldsymbol{\lambda_i})\right)$$

[Schmitz'18]





Minimum Kantorovich Estimators

$$\min_{\boldsymbol{\theta}\in\Theta} W(\boldsymbol{\nu}_{\text{data}}, f_{\boldsymbol{\theta}\sharp}\boldsymbol{\mu})$$

[Bassetti'06] 1st reference discussing this approach.

$$\text{Challenge: } \nabla_{\boldsymbol{\theta}} W(\boldsymbol{\nu}_{\text{data}}, f_{\boldsymbol{\theta}\sharp}\boldsymbol{\mu})?$$

[Montavon'16] use regularized OT in a finite setting.

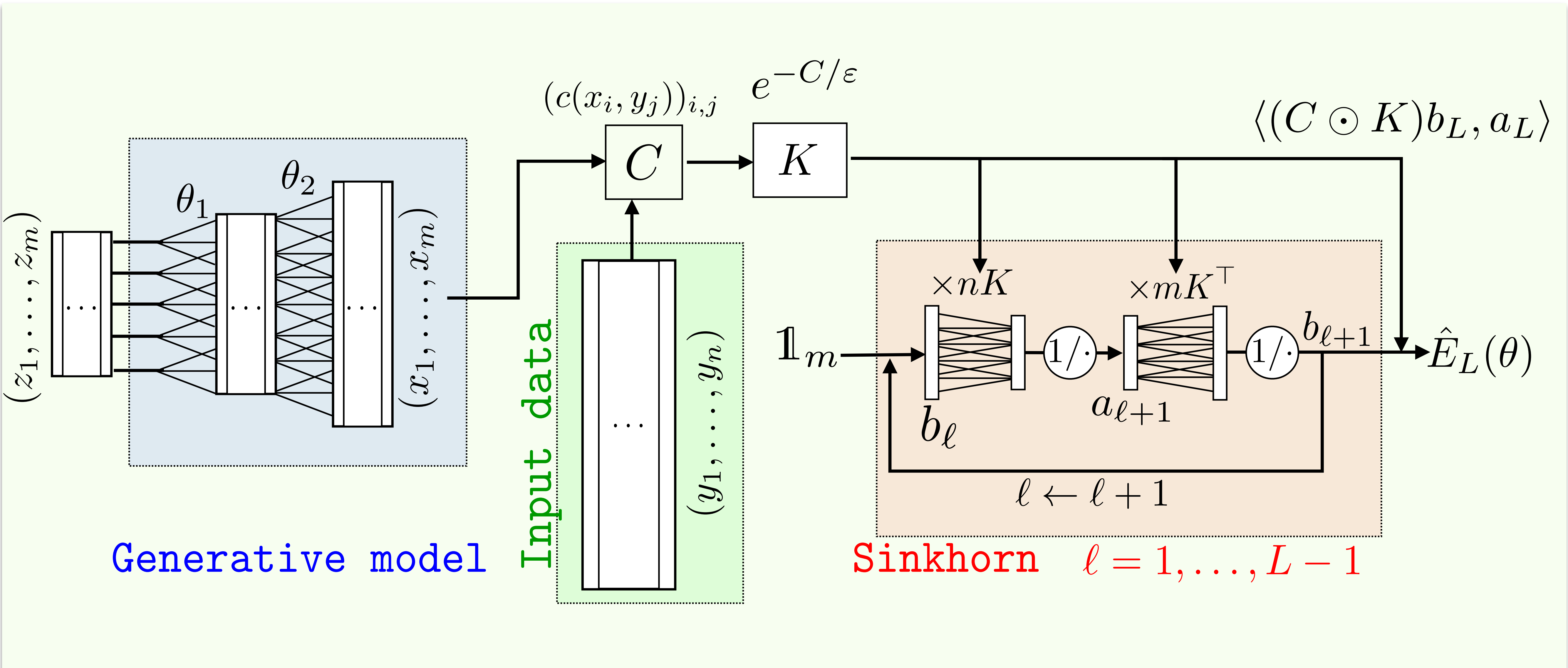
[Arjovsky'17] *(WGAN)* uses a NN to approximate dual solutions and recover gradient w.r.t. parameter

[Bernton'17] *(Wasserstein ABC)*

[Genevay'17, Salimans'17] *(Sinkhorn approach)*

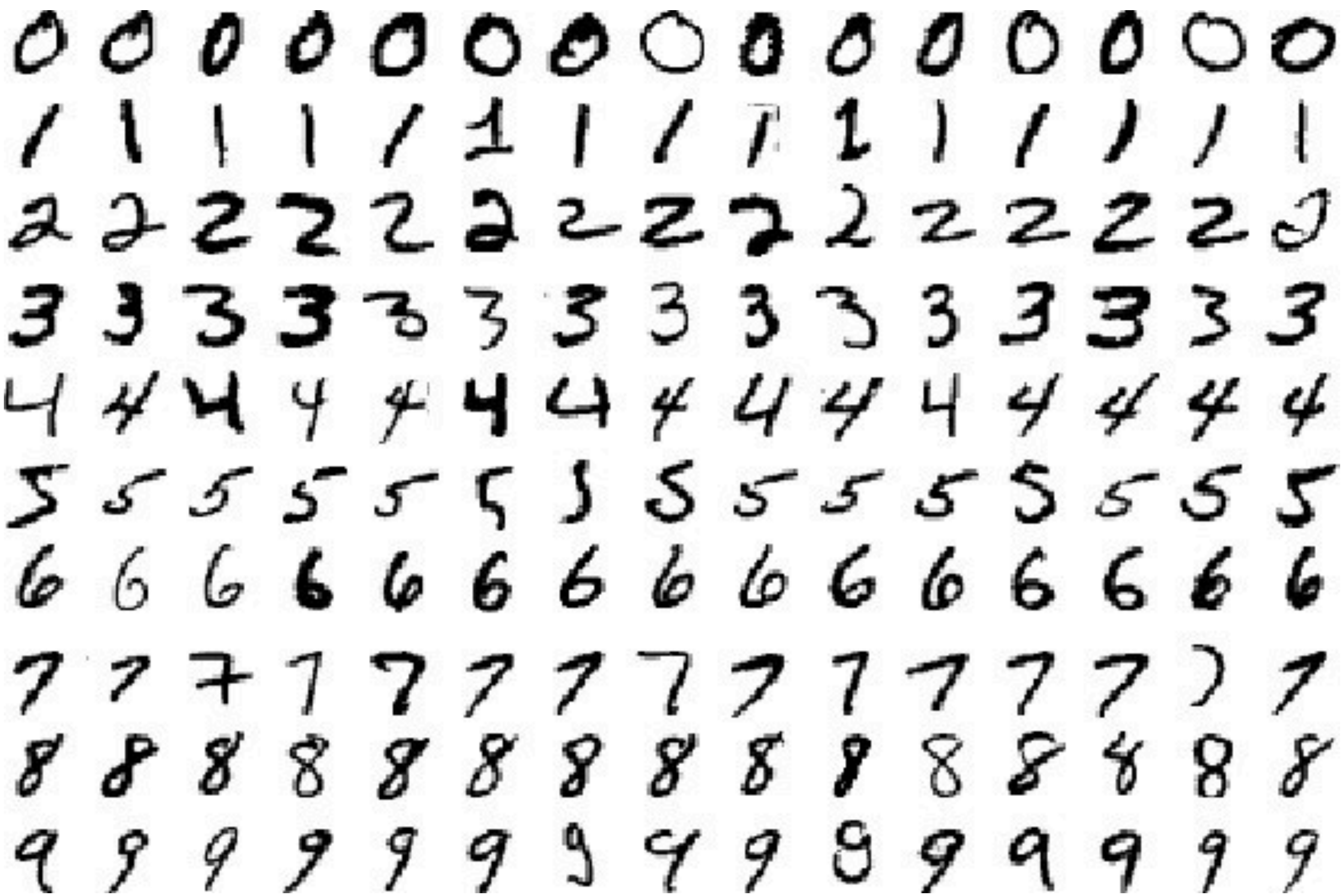
Proposal: Autodiff OT using Sinkhorn

Approximate W loss by the transport cost $\bar{W}_L$ after L Sinkhorn iterations.



[GPC'17]

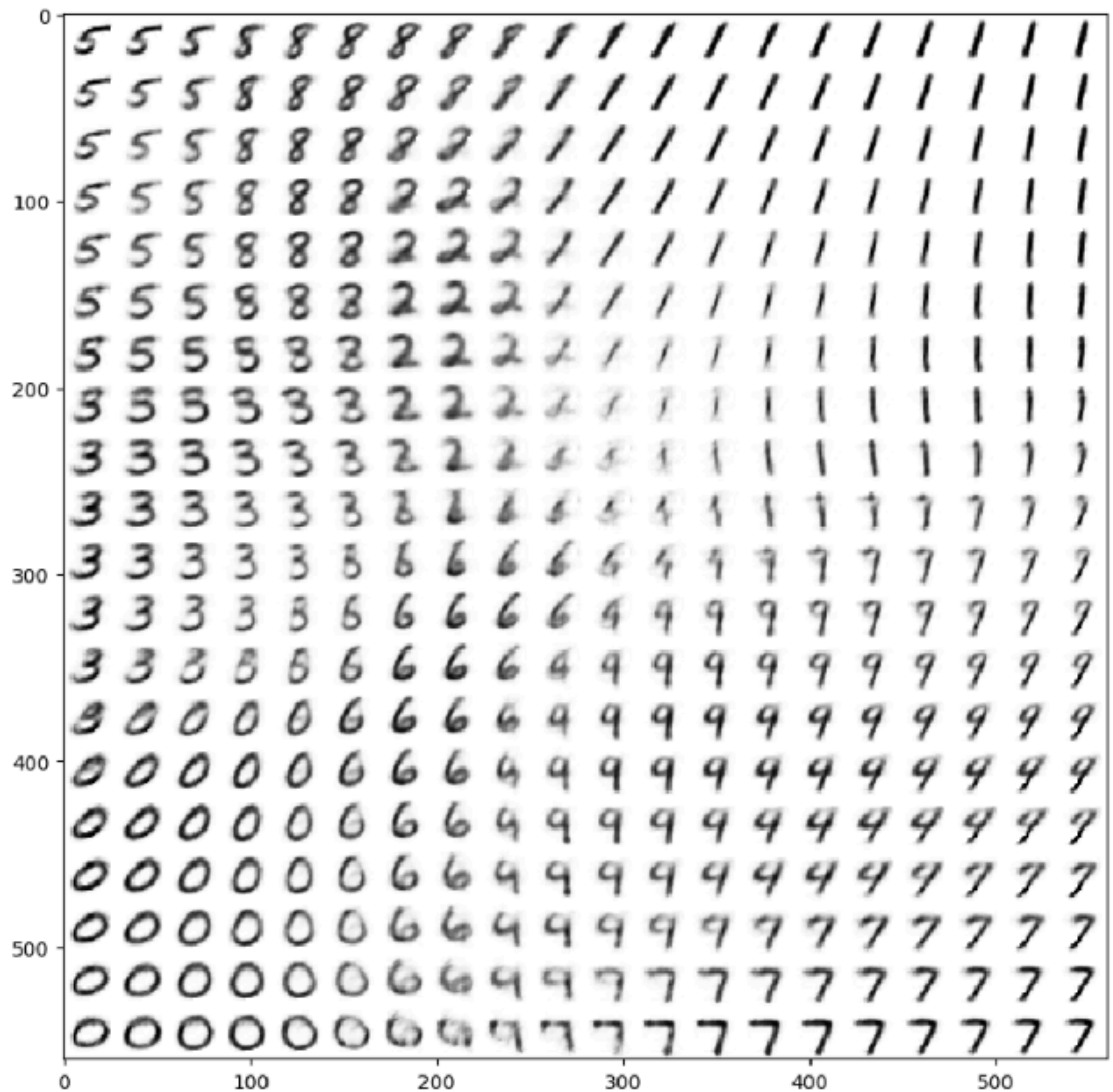
Example: MNIST, Learning $f_{\boldsymbol{\theta}}$



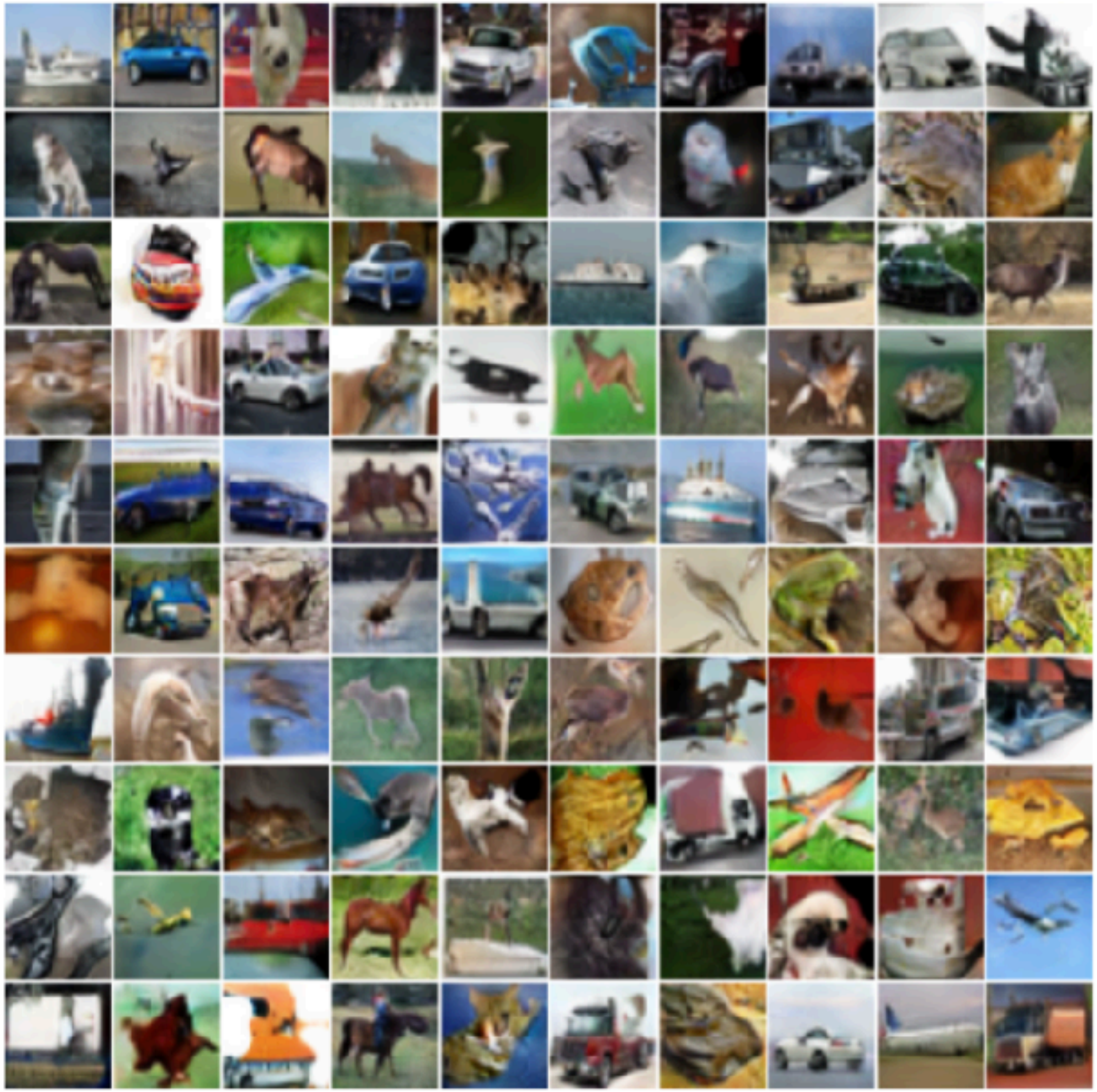
Example: MNIST, Learning $f_{\boldsymbol{\theta}}$

Latent space

$[0, 1]^2$



Example: Generation of Images



arxiv.org/pdf/1710.05488

[Salimans'18]

Example: Generation of Images

arxiv.org/pdf/1710.05488

[Salimans'18]

Concluding Remarks

- *Regularized* OT is much faster than OT when handled in full generality.
- *Regularized* OT can interpolate between W and the *MMD / Energy distance (MMD)* metrics.
- The solution of *regularized OT* is *"auto-differentiable"*.
- **Many open problems remain in ML that can be addressed with OT.**