Optimal Transport in Data Sciences: Why and How?

Marco Cuturi

ENSAE ParisTech
École nationale de la statistique et de l'administration économique
université PARIS-SACLAY

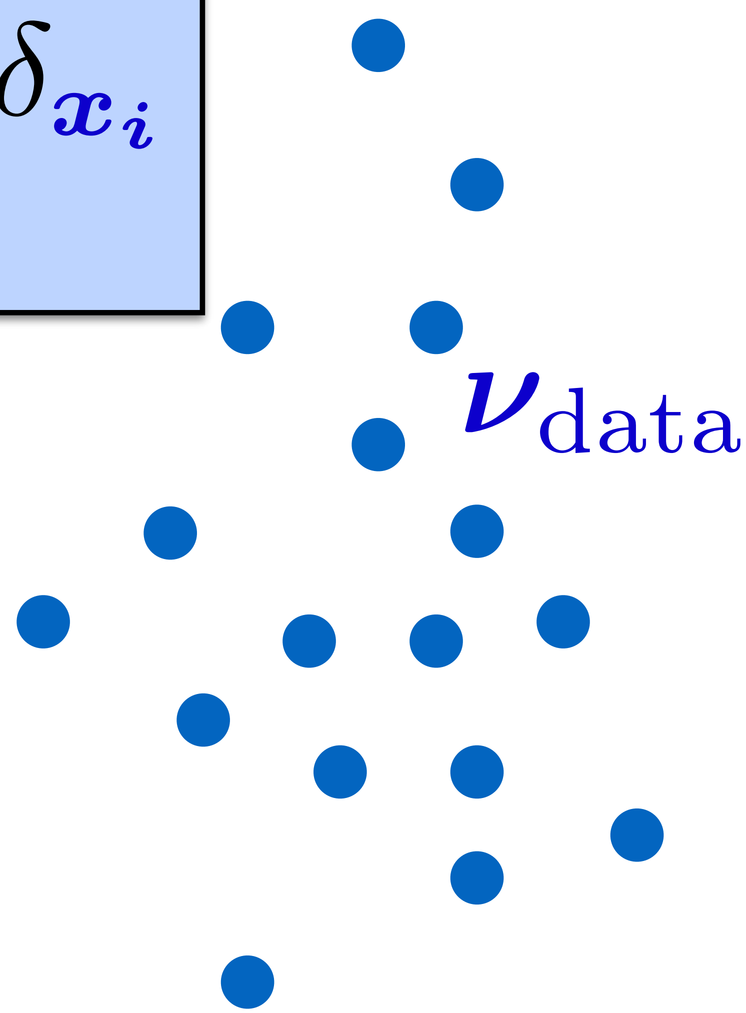
Joint work with
G. Peyré, F. Bach, A. Genevay *(ENS)*
N. Bonneel (*INRIA*) A. Rolet *(Kyoto)* J. Solomon *(MIT)*

https://optimaltransport.github.io/

Statistics 0.1 : Density Fitting

We collect data

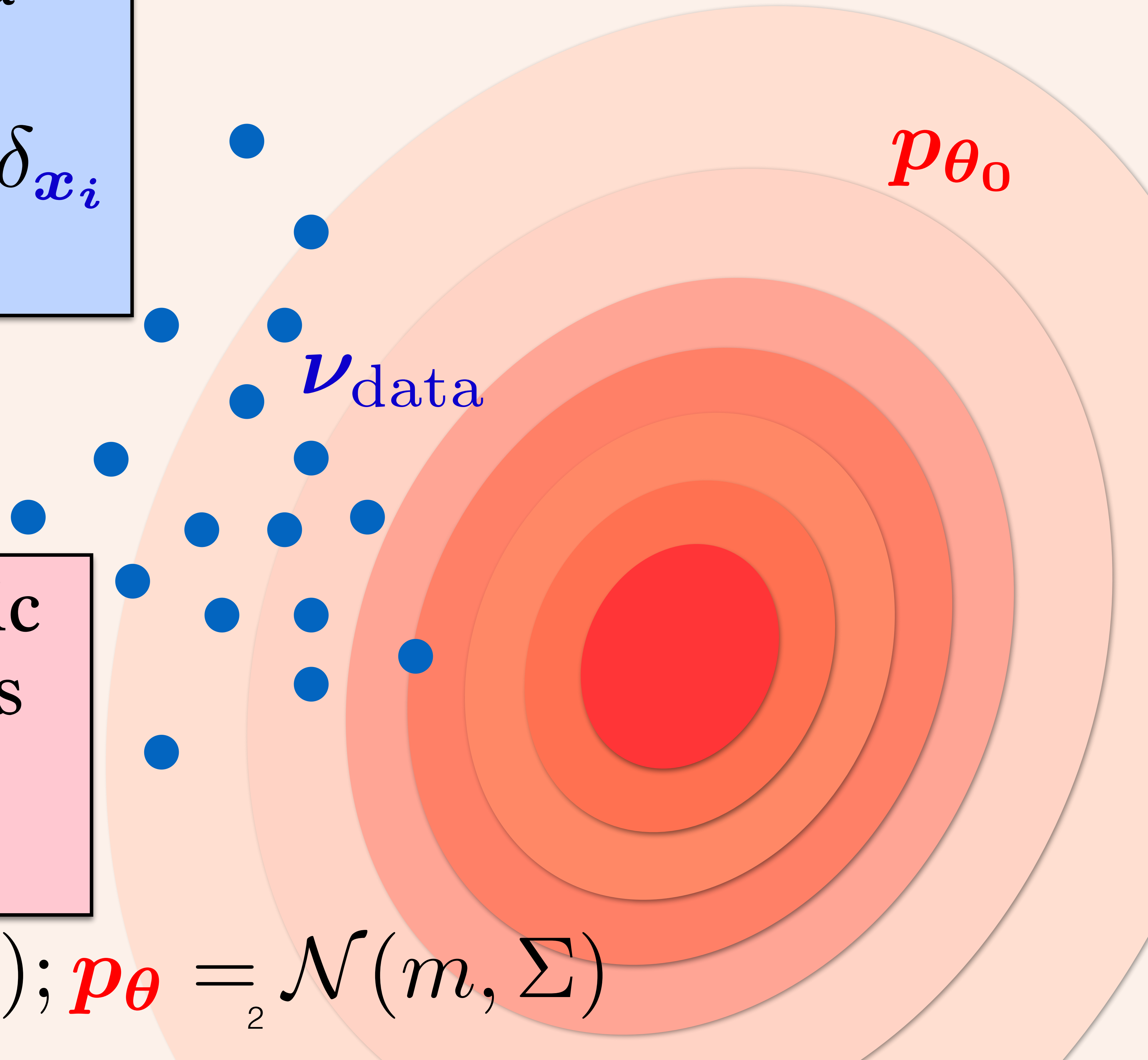
$$\boldsymbol{\nu}_{\text{data}} = \frac{1}{N}\sum_{i=1}^{N}\delta_{\boldsymbol{x_i}}$$



Statistics 0.1 : Density Fitting

We collect data

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

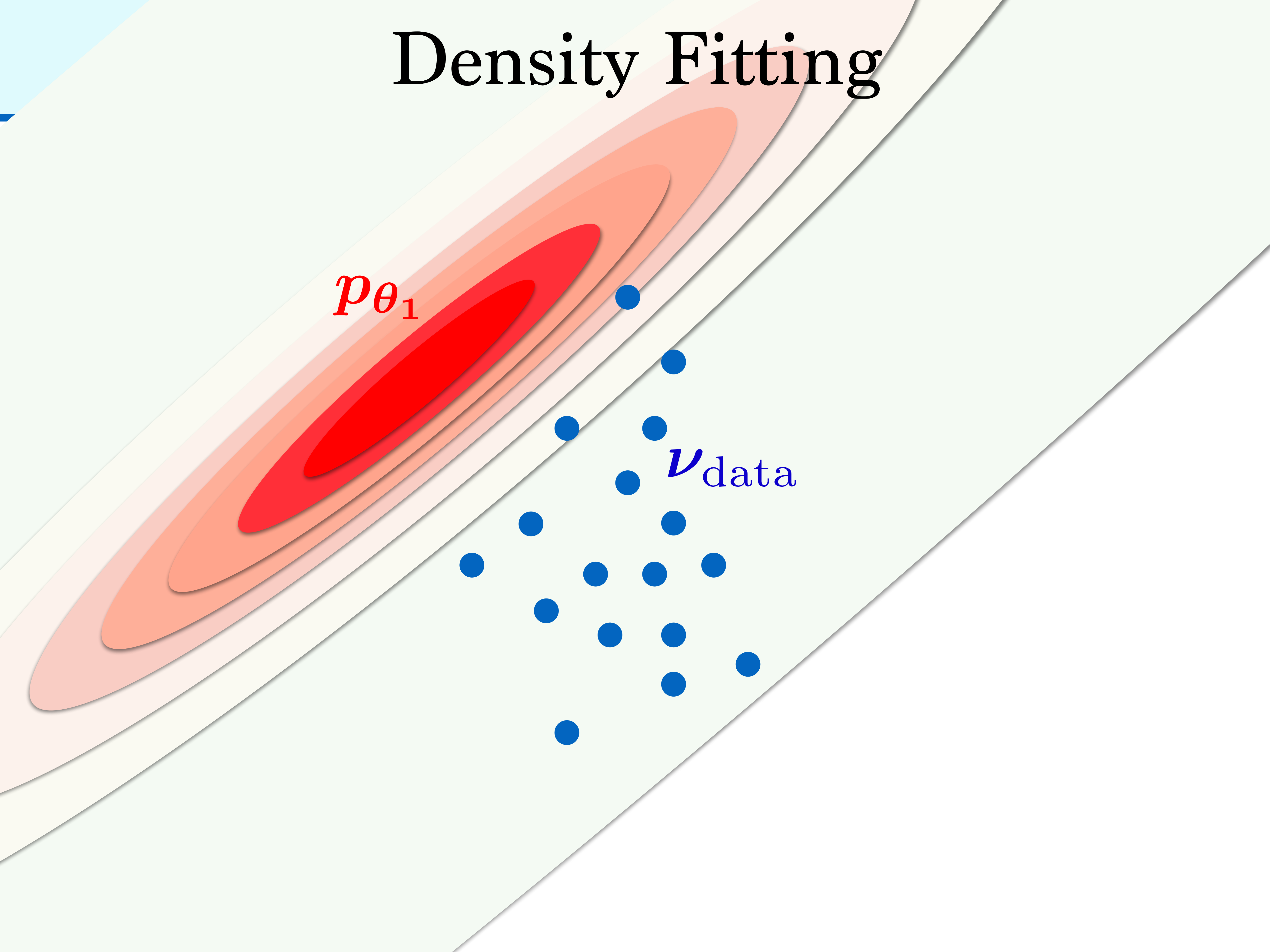


We fit a parametric family of densities

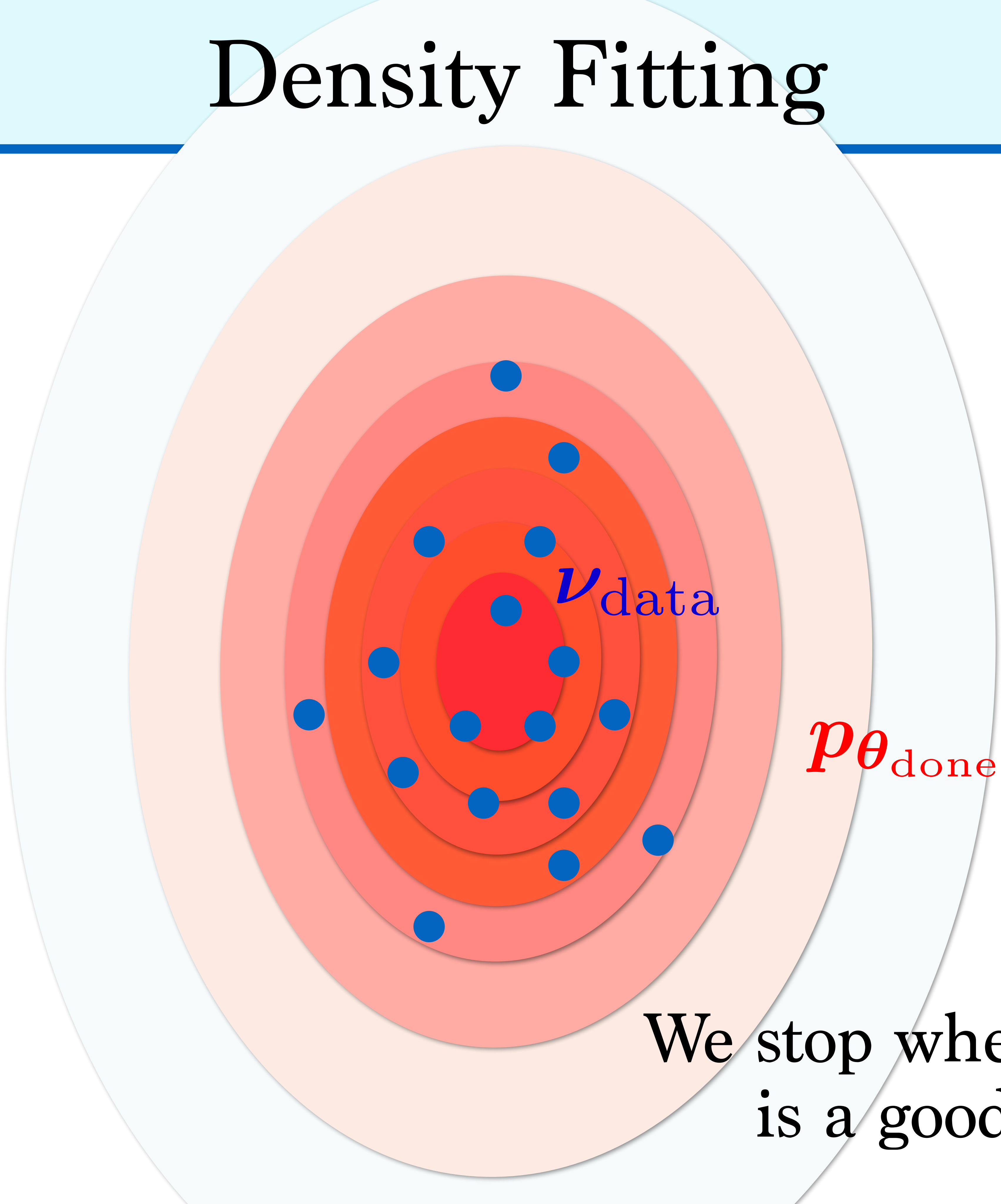
$$\{\boldsymbol{p_\theta}, \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$$

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

Density Fitting

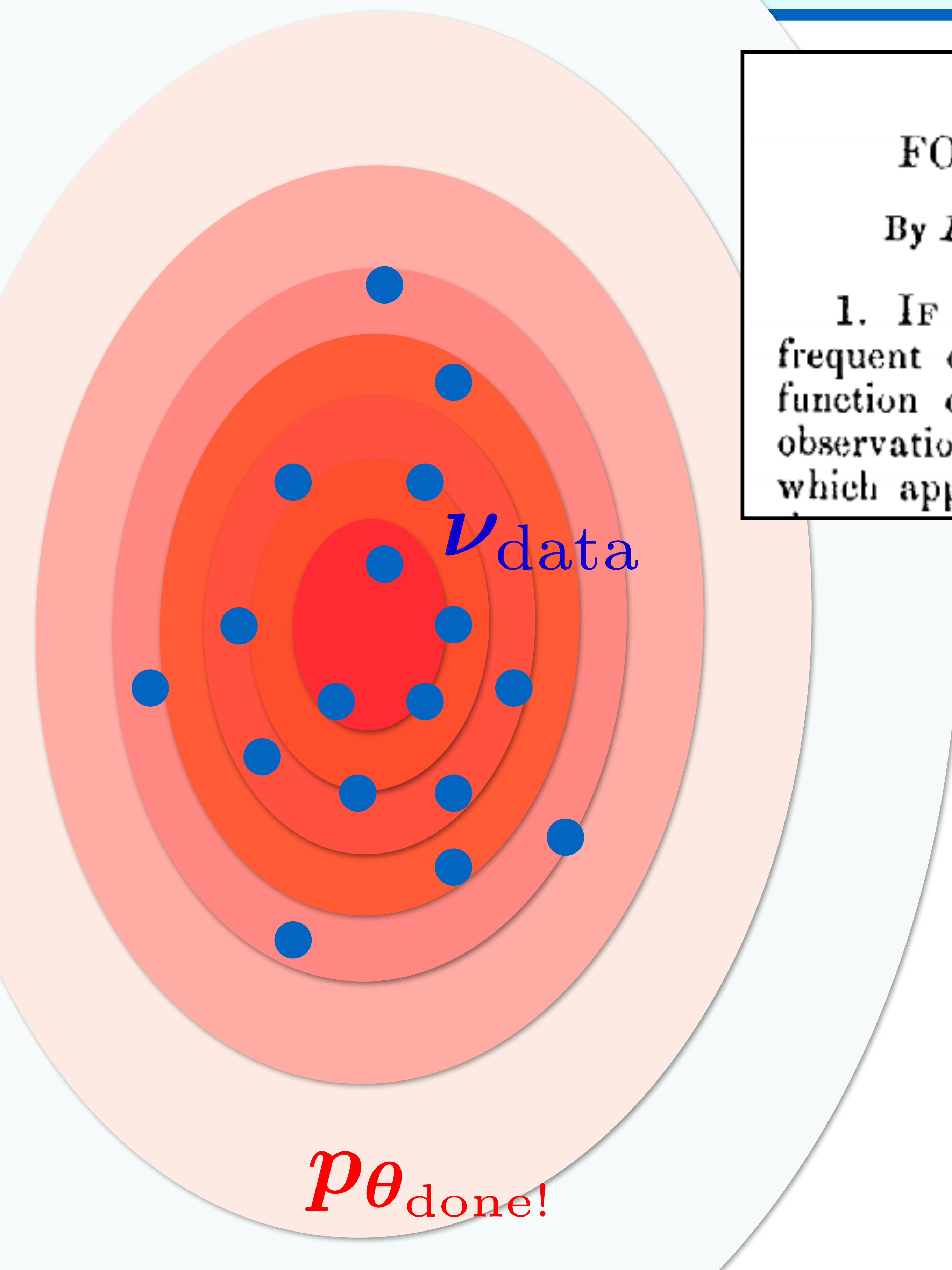


Density Fitting



We stop when there is a good fit.

Maximum Likelihood Estimation



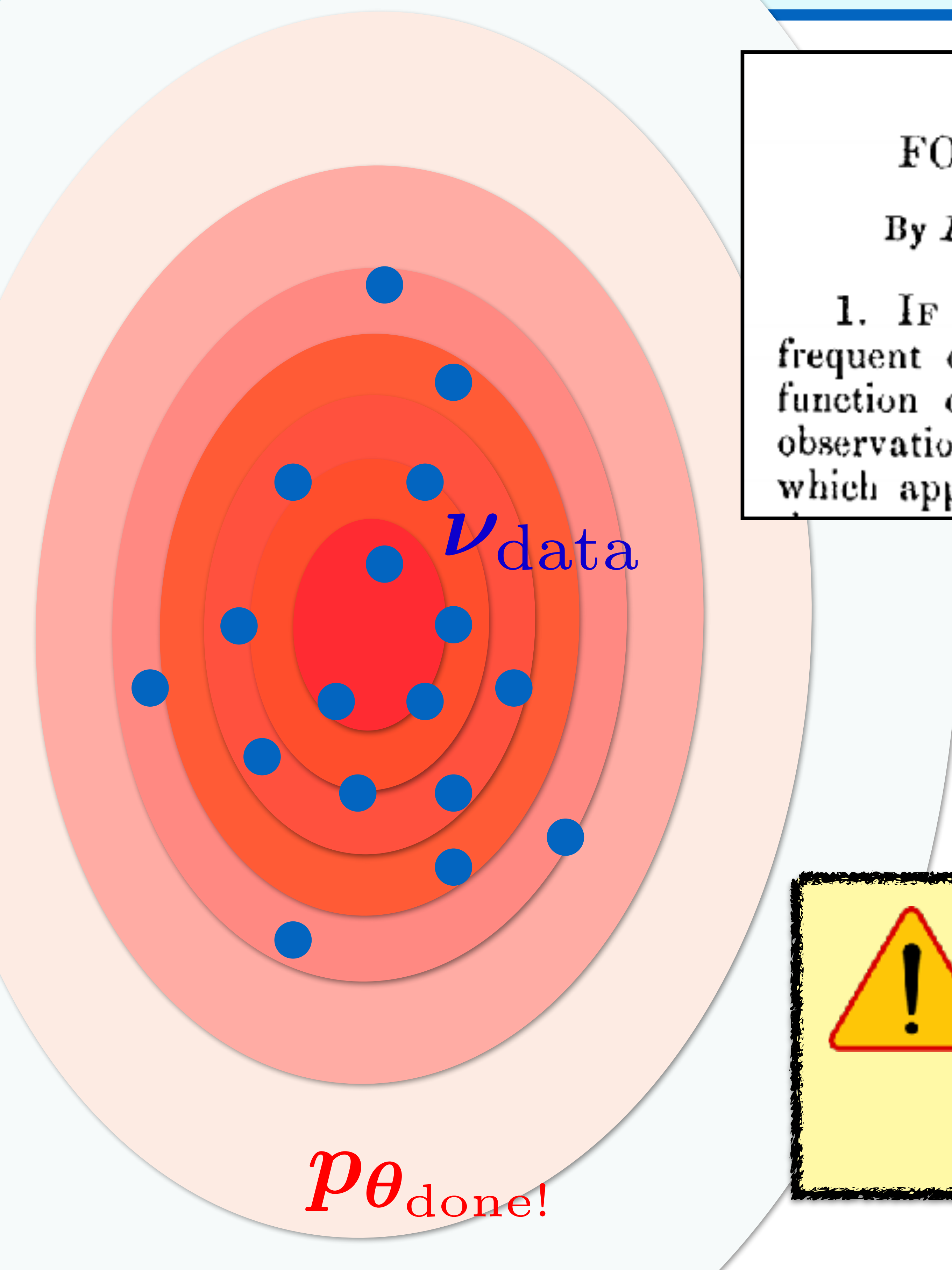
ON AN ABSOLUTE CRITERION FOR FITTING FREQUENCY CURVES.

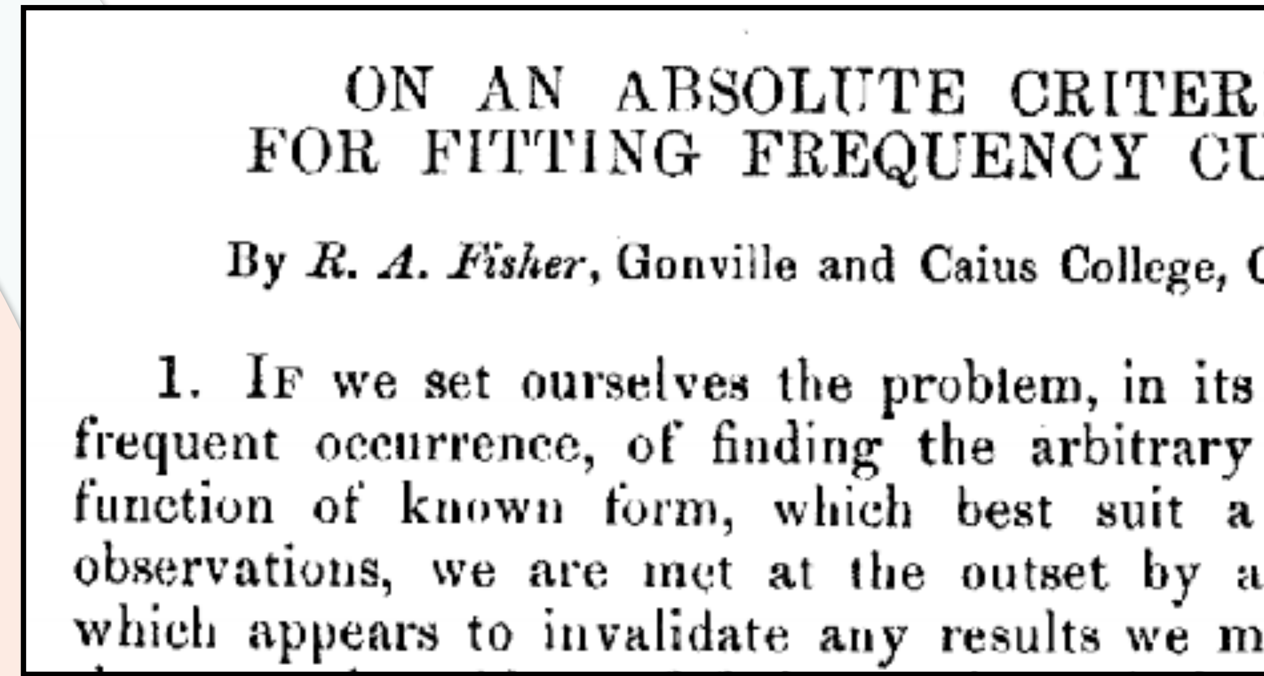
By *R. A. Fisher*, Gonville and Caius College, Cambridge.

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.

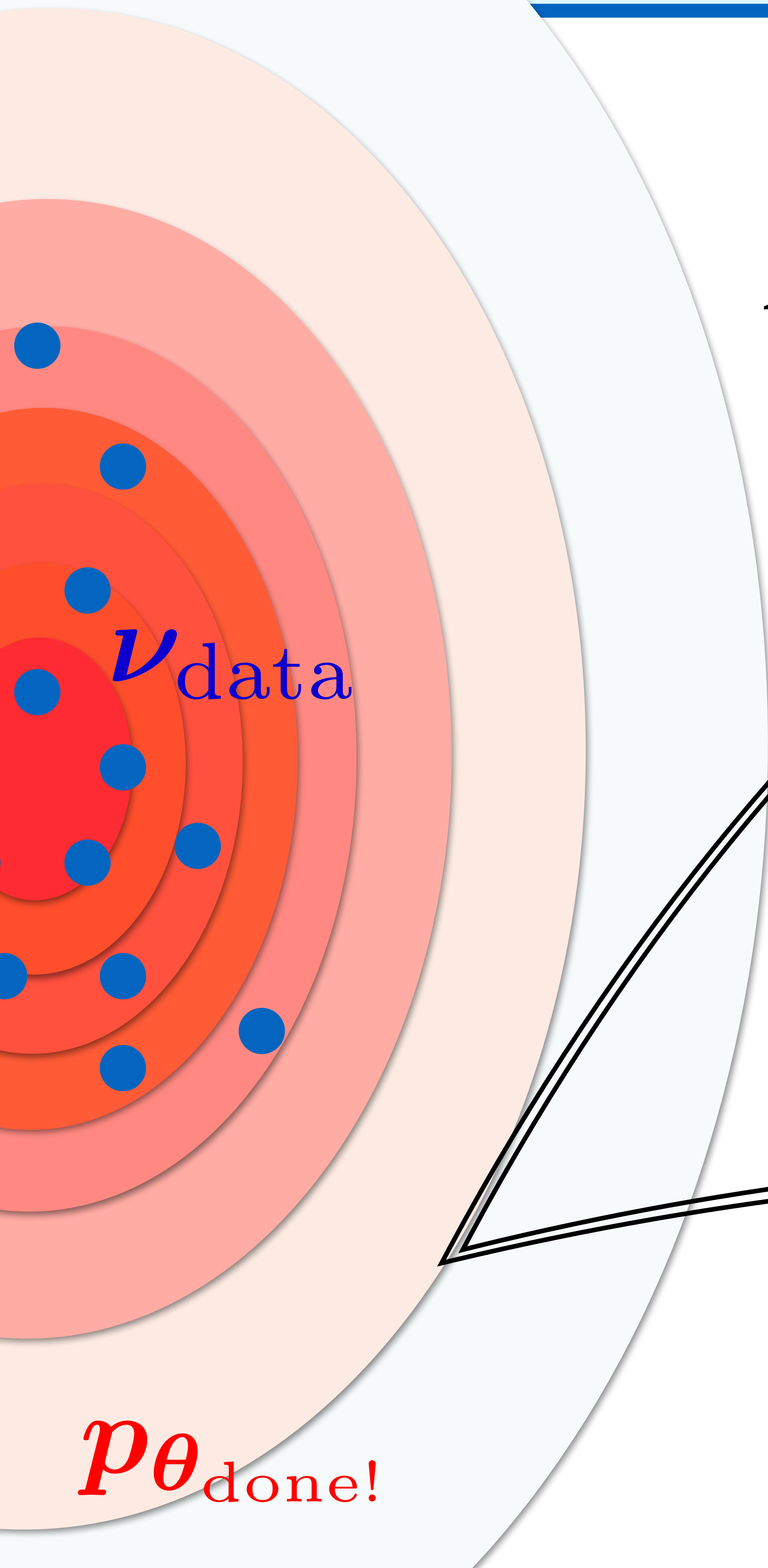
1. If we set ourselves the problem, in its
frequent occurrence, of finding the arbitrary
function of known form, which best suit a
observations, we are met at the outset by an
which appears to invalidate any results we ma

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

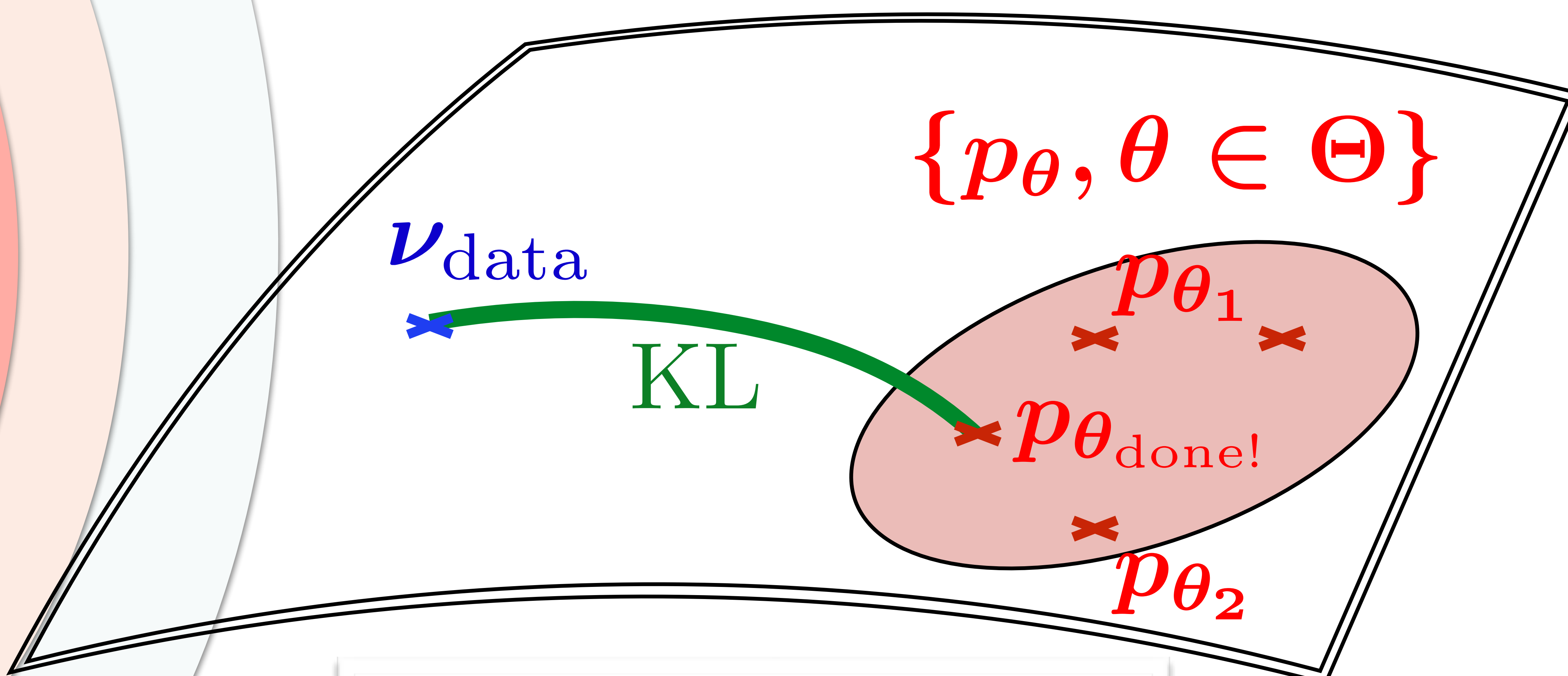
$$\log 0 = -\infty$$

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

Maximum Likelihood Estimation



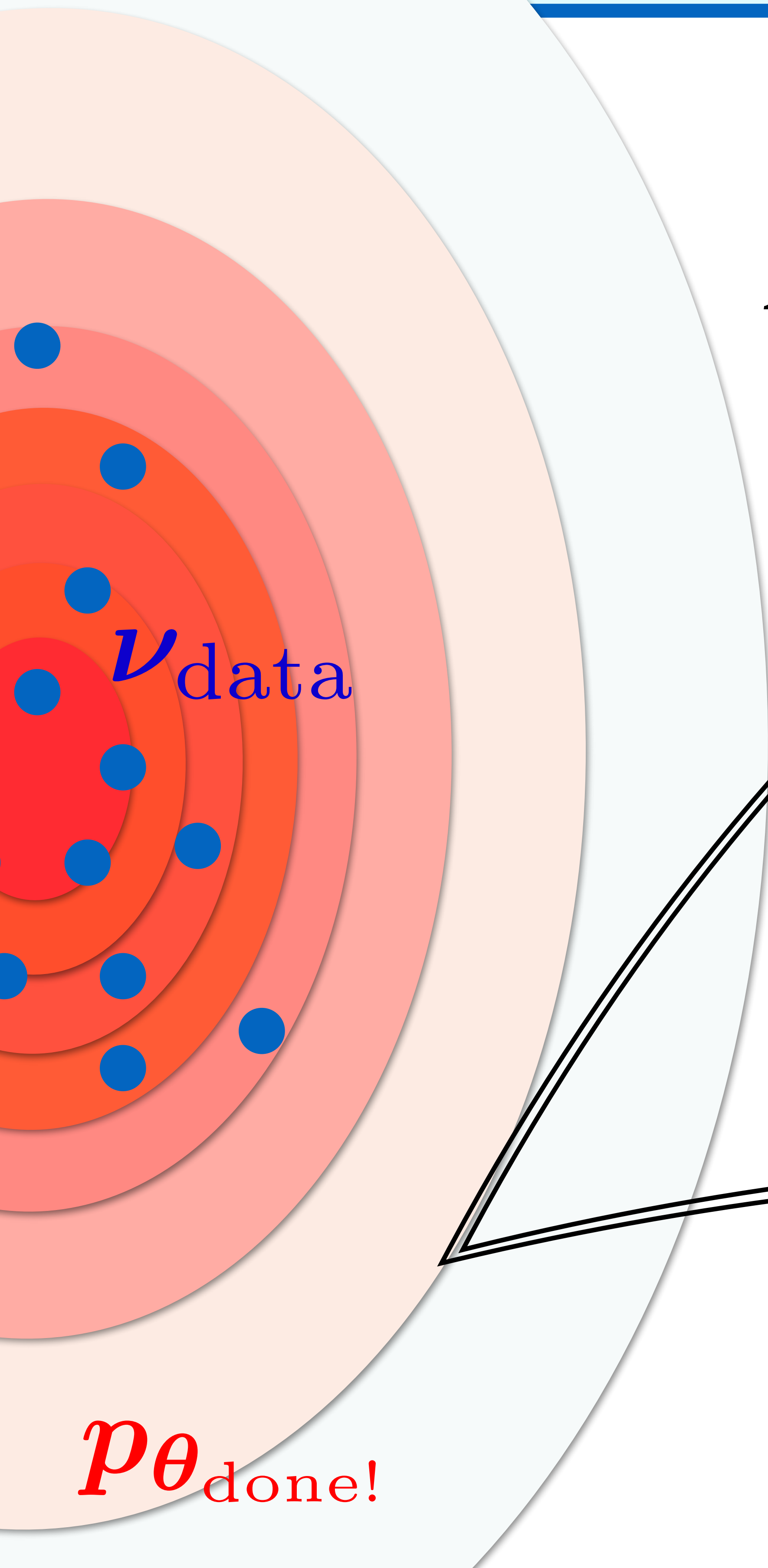
Equivalent to a KL projection in the space of probability measures

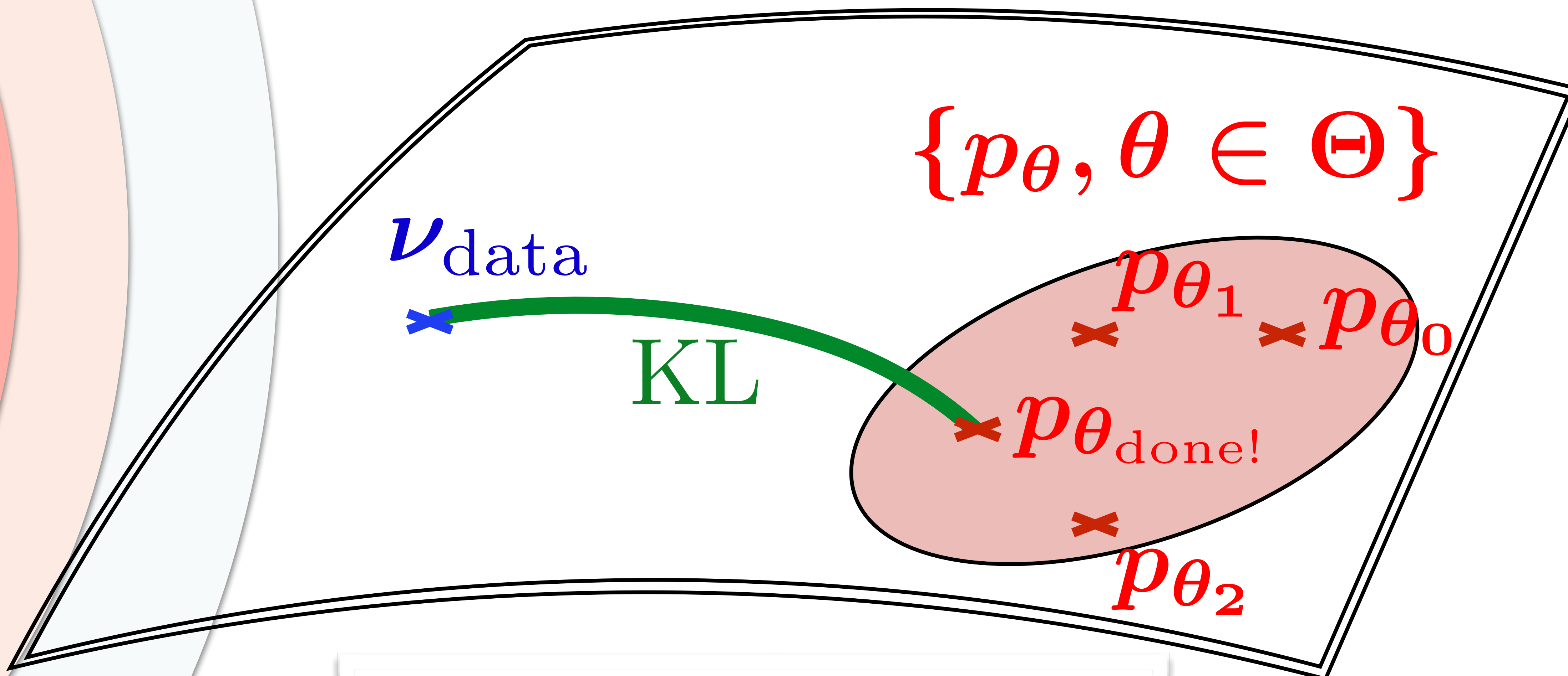


$$\min_{\theta \in \Theta} \mathrm{KL}(\nu_{\text{data}} \| p_\theta)$$

Maximum Likelihood Estimation

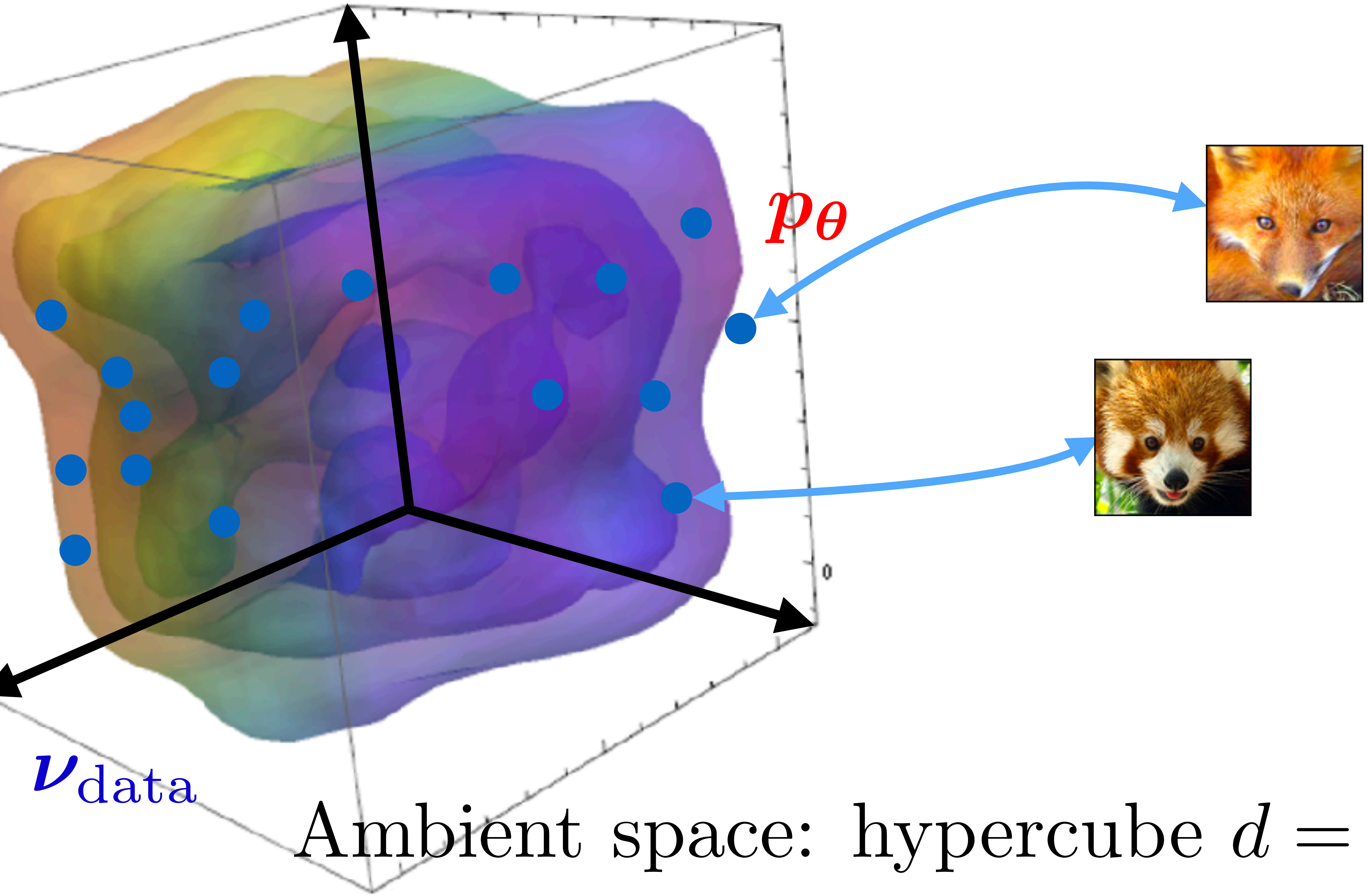
Equivalent to a KL projection in the space of probability measures





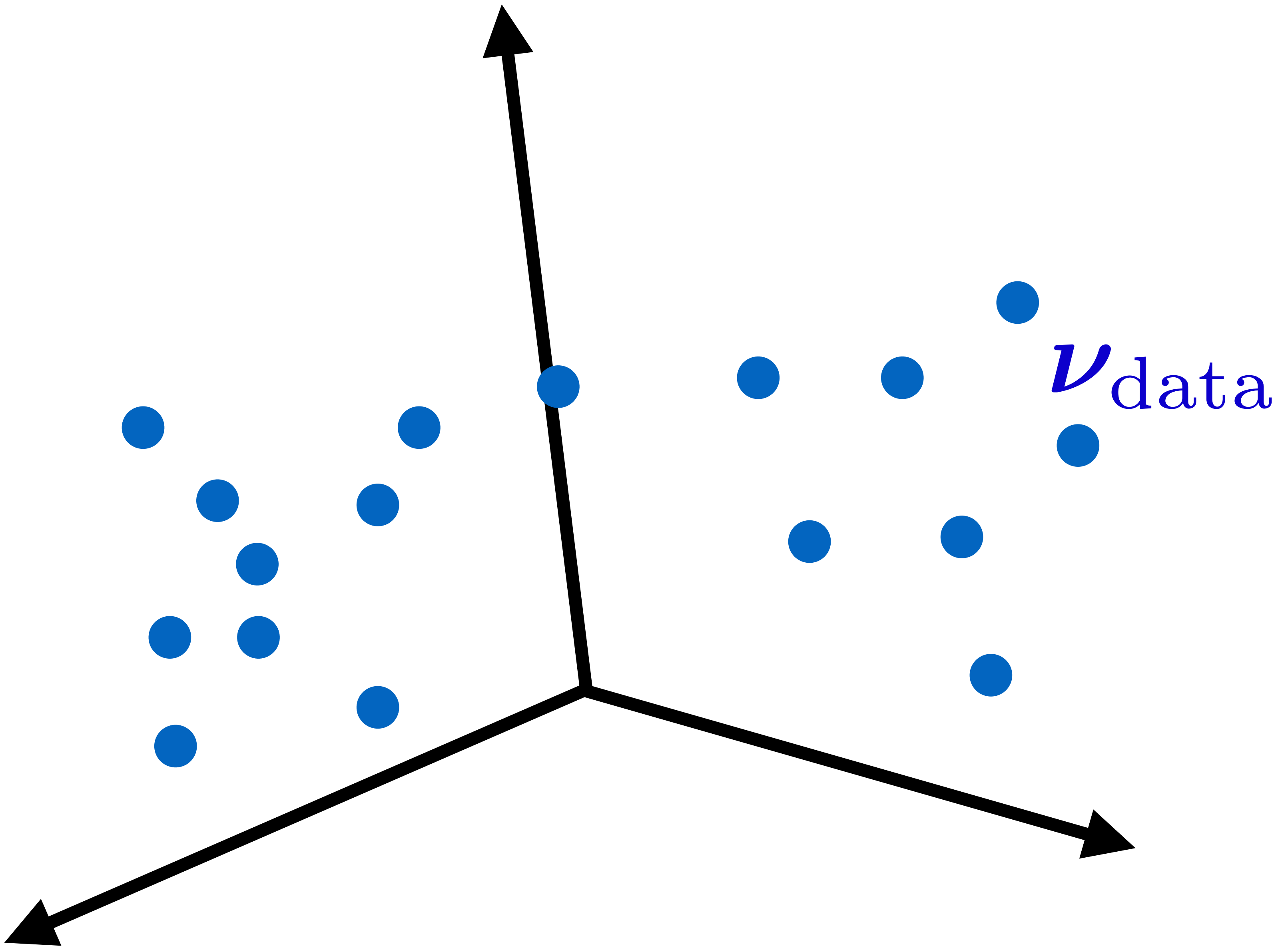
$$\min_{\theta \in \Theta} \mathrm{KL}(\nu_{\text{data}} \| p_\theta)$$

In higher dimensional spaces...

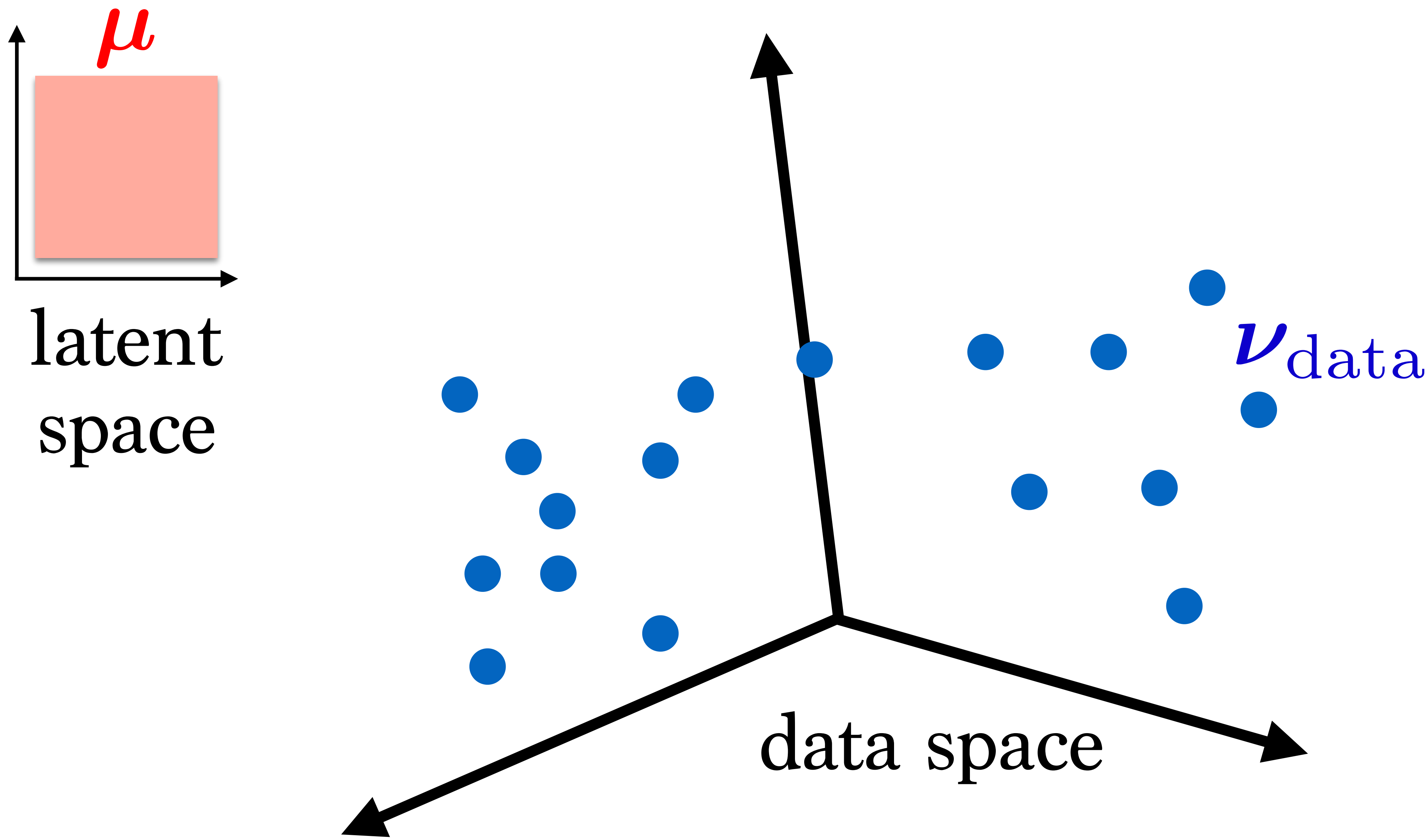


Ambient space: hypercube $d = 30.000$

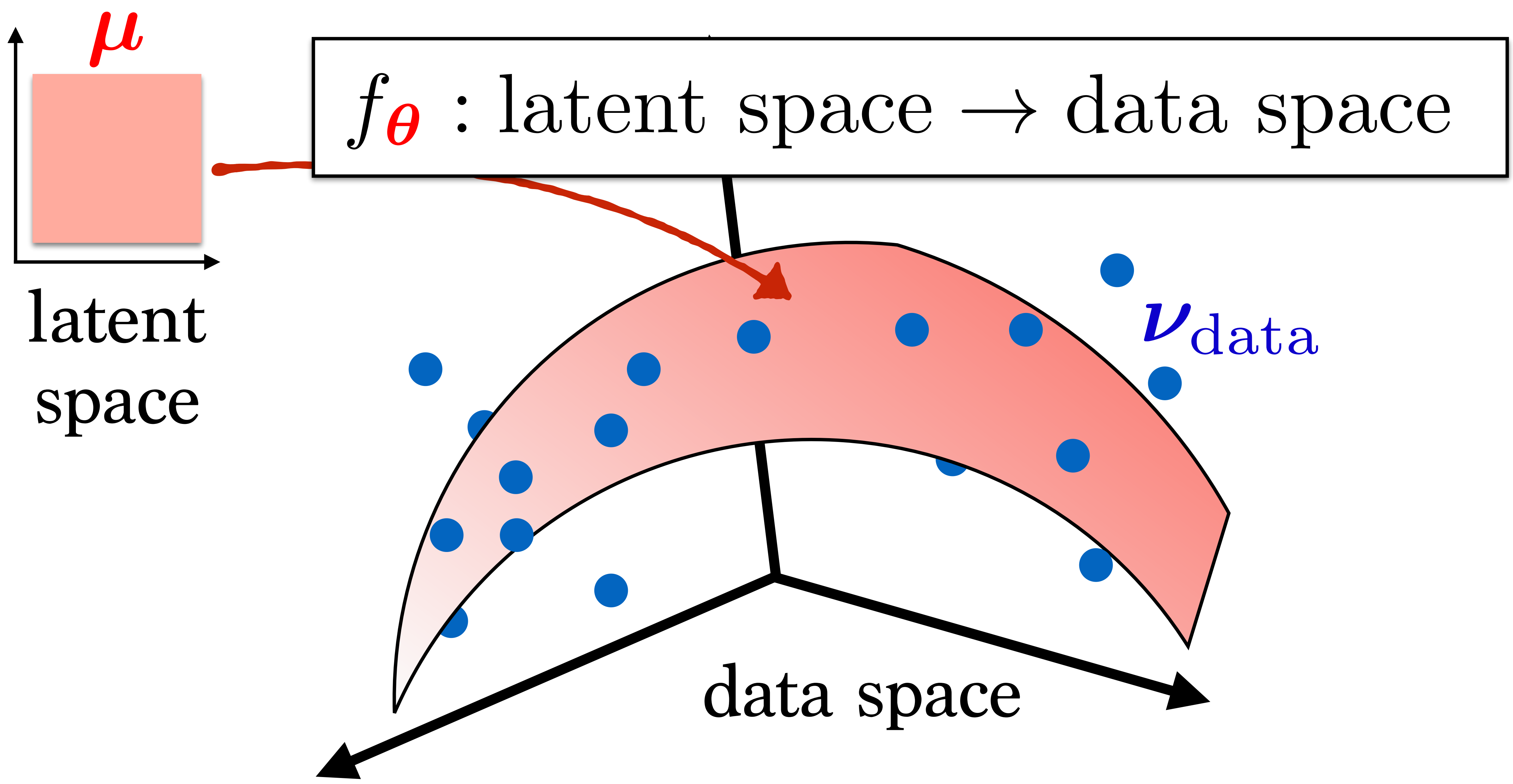
Generative Models



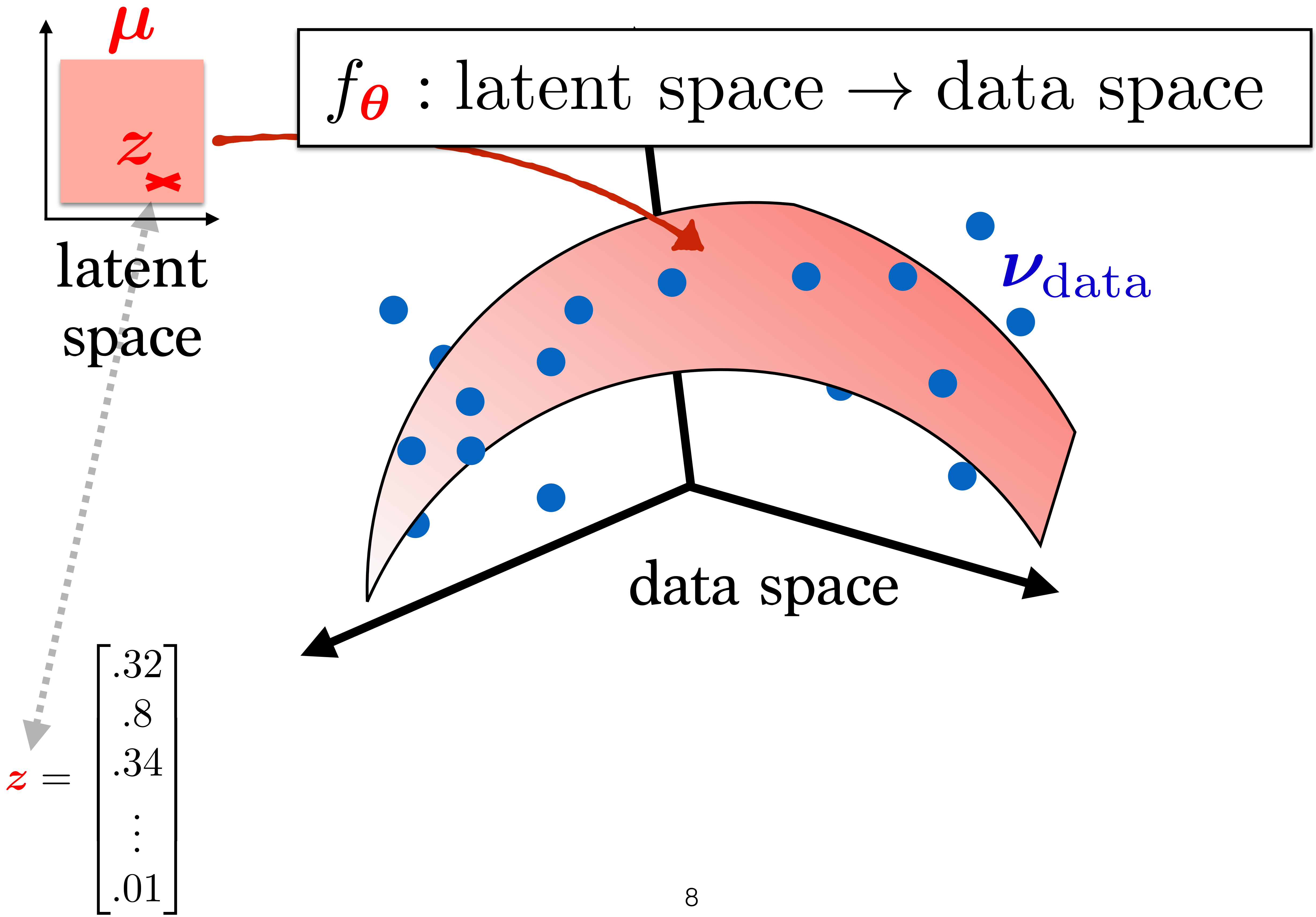
Generative Models



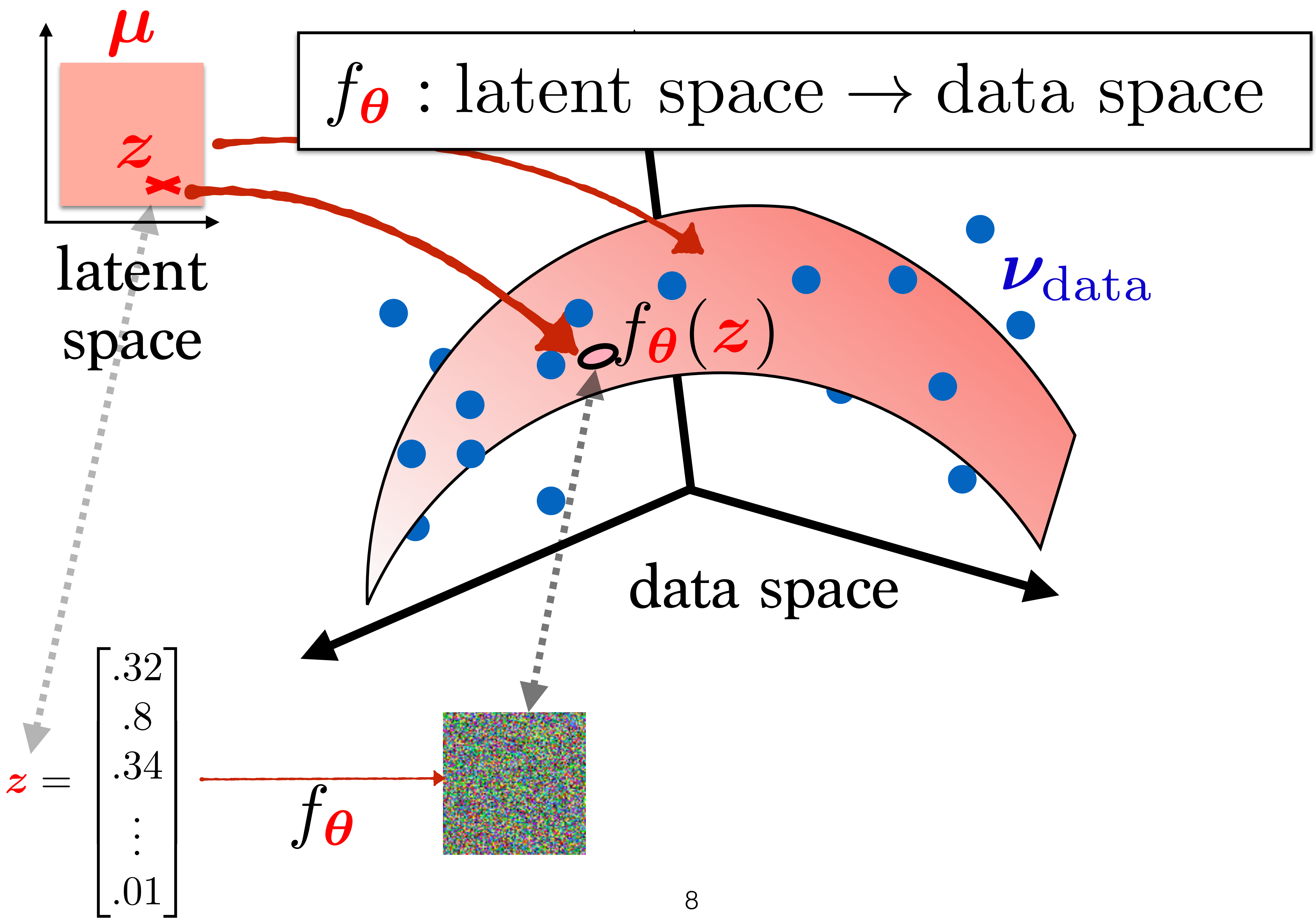
Generative Models



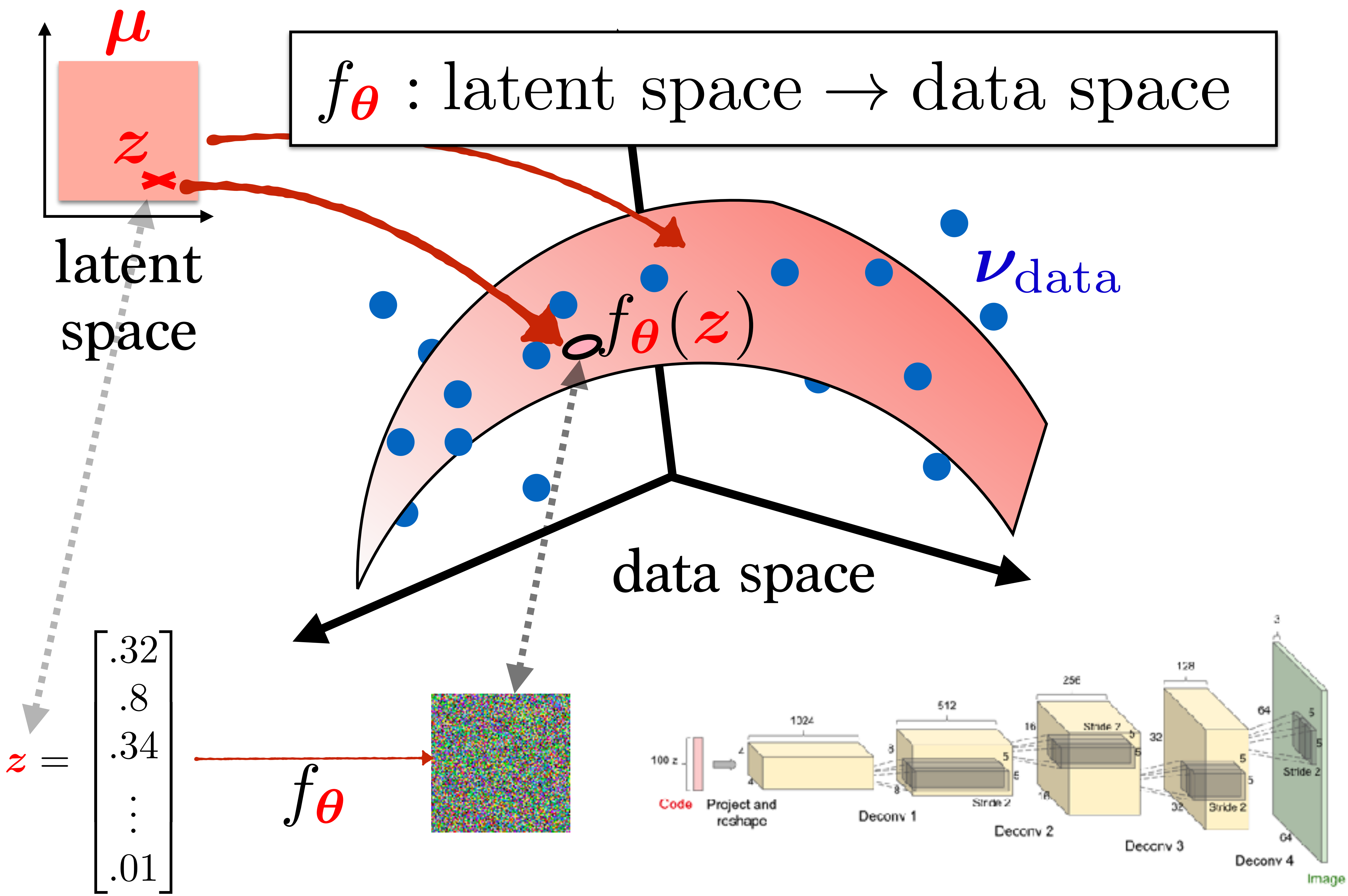
Generative Models



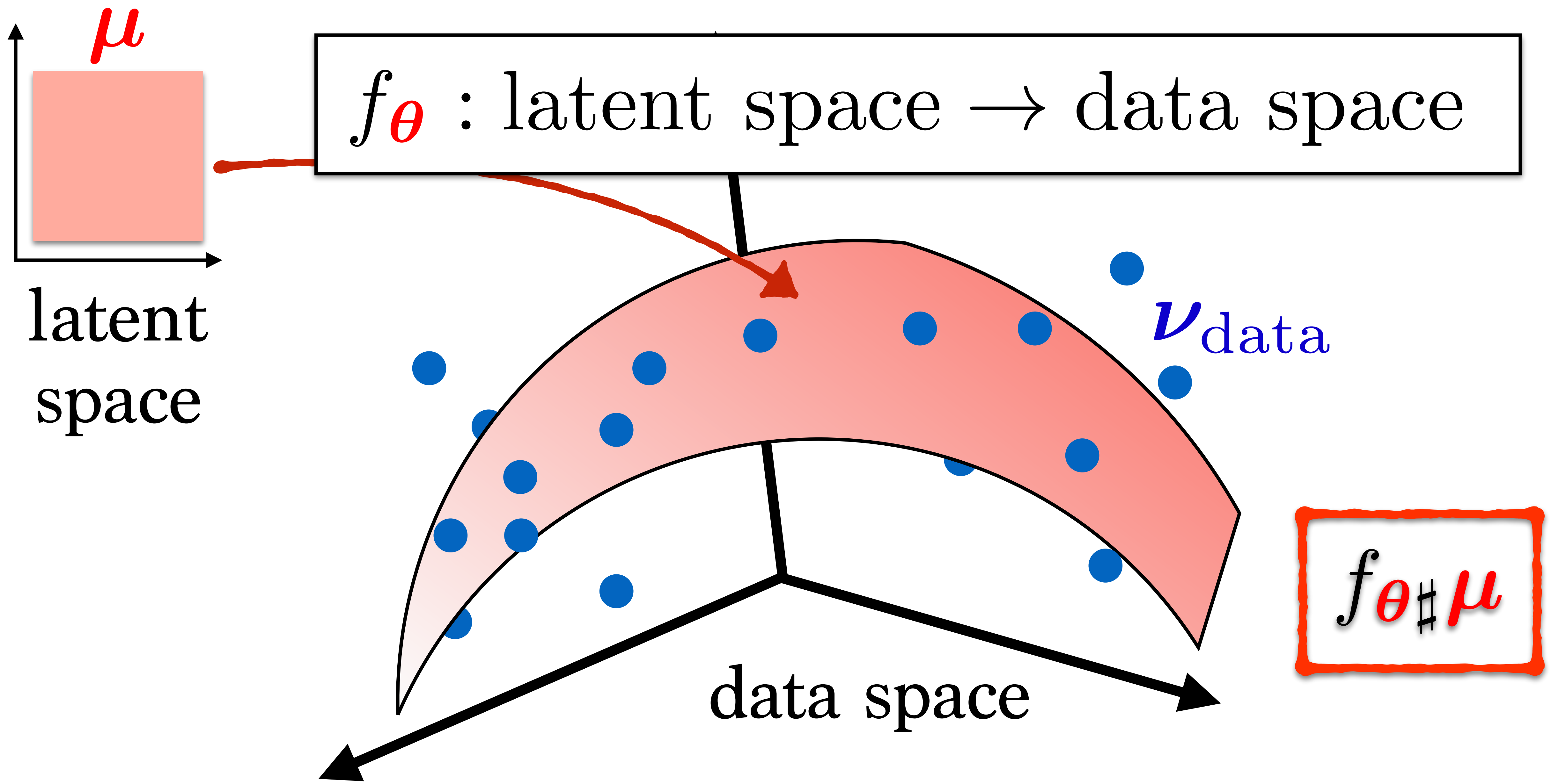
Generative Models



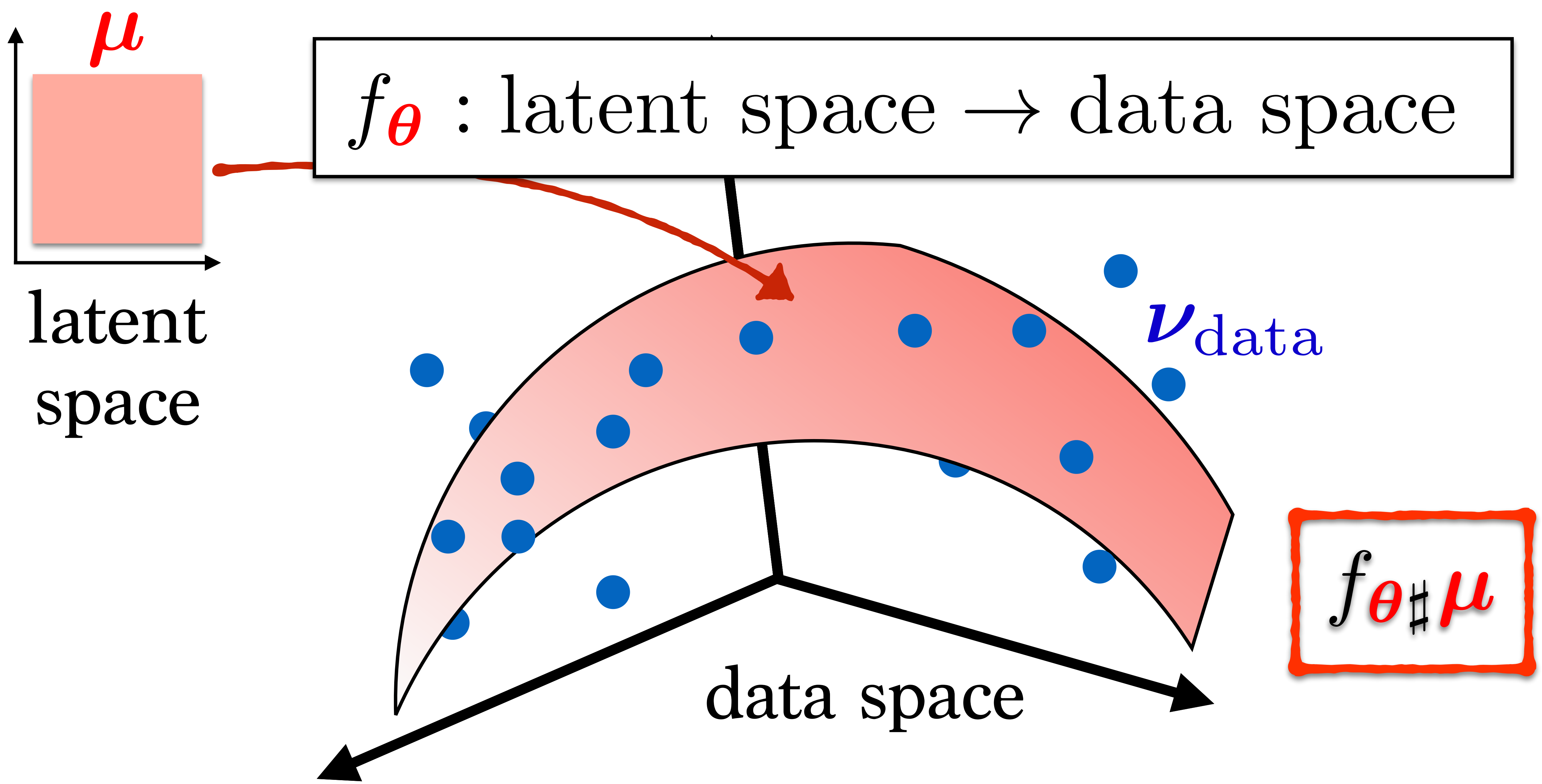
Generative Models



Generative Models

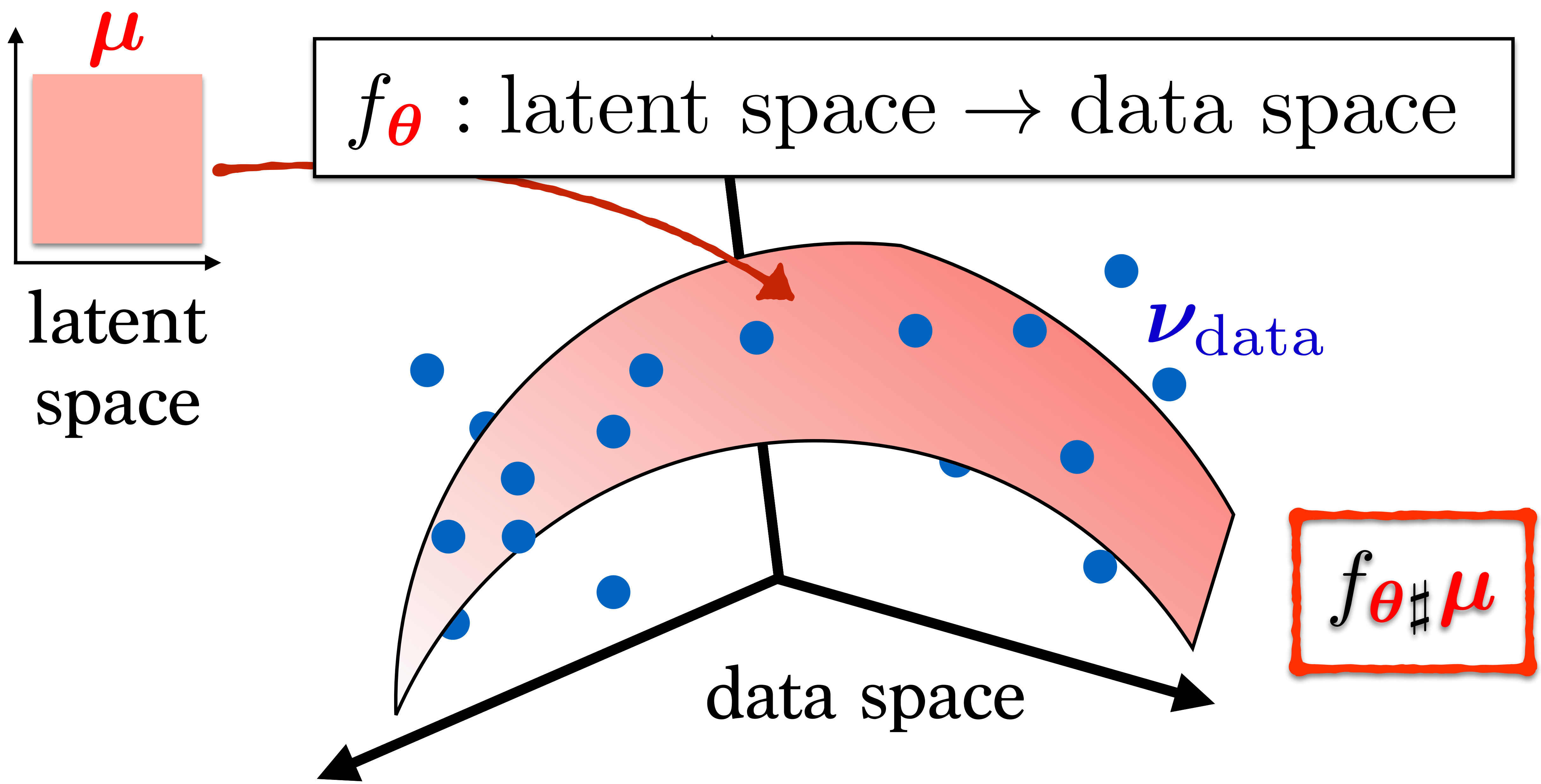


Generative Models



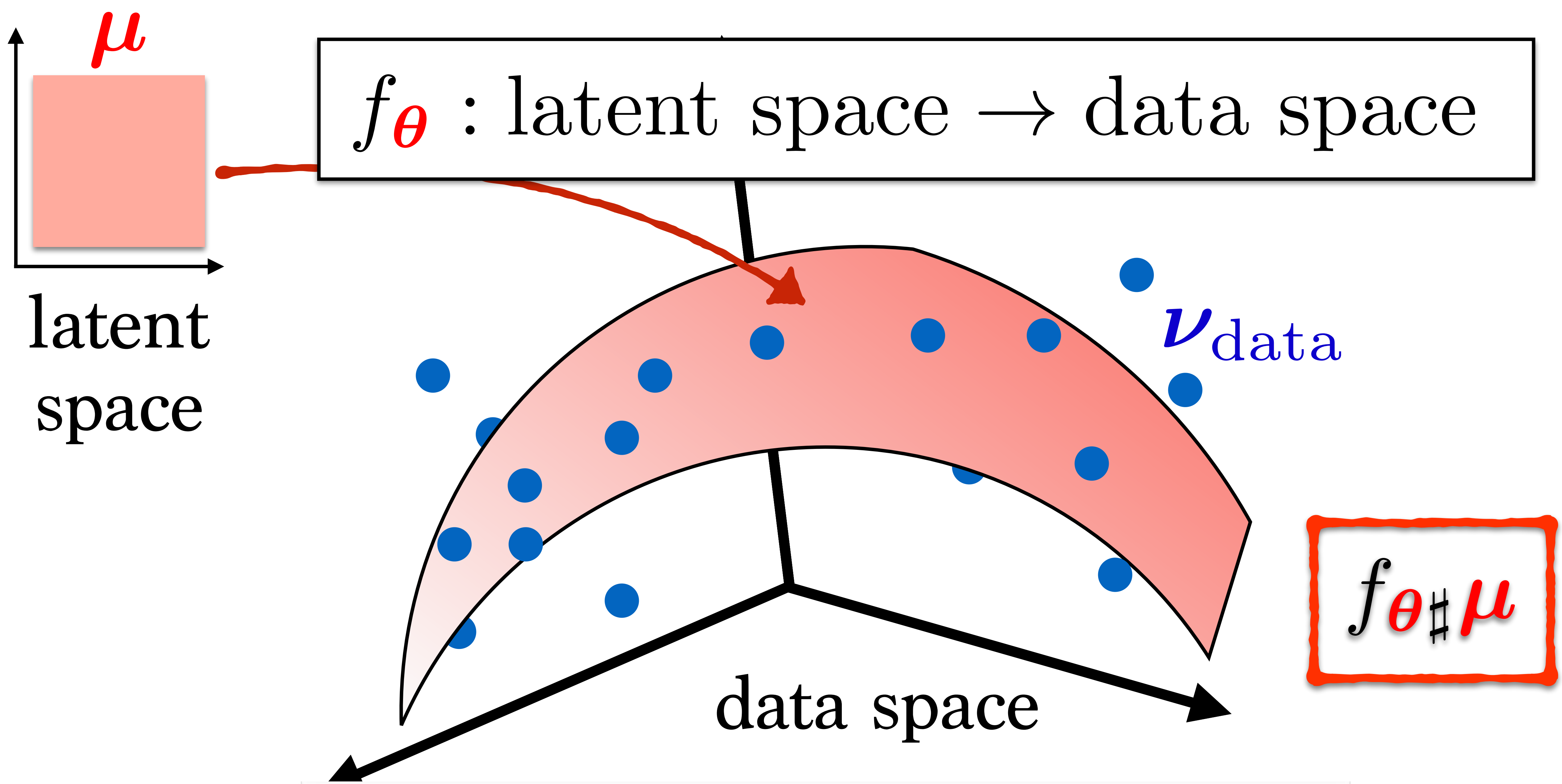
Goal: find θ such that $f_{\theta\sharp}\mu$ fits ν_{data}

Generative Models



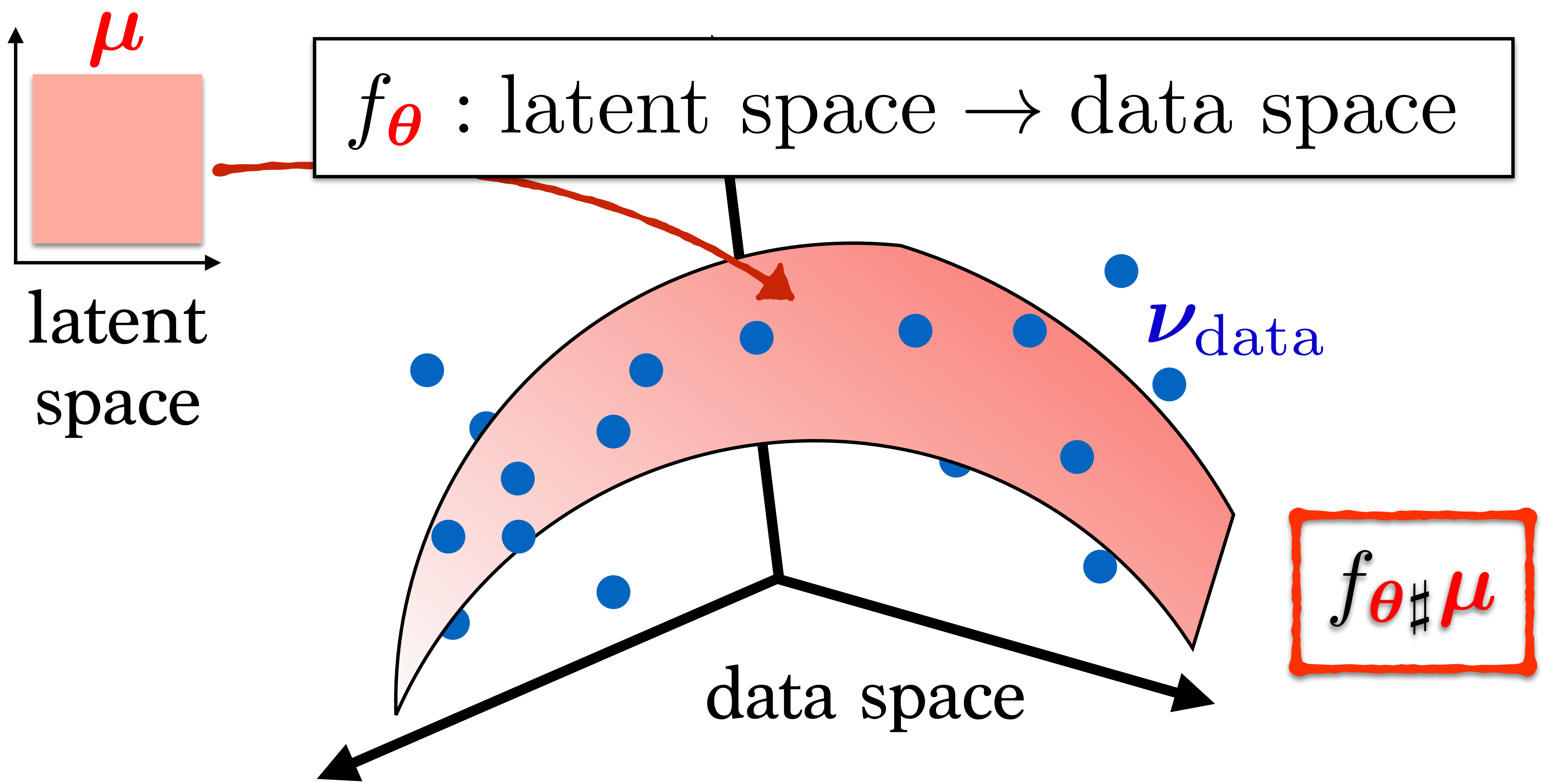
Goal: find θ such that $f_{\theta\sharp}\mu$ fits ν_{data}

Generative Models



Difference between fitting $f_{\theta\sharp}\mu$ *vs.* a density p_{θ}?

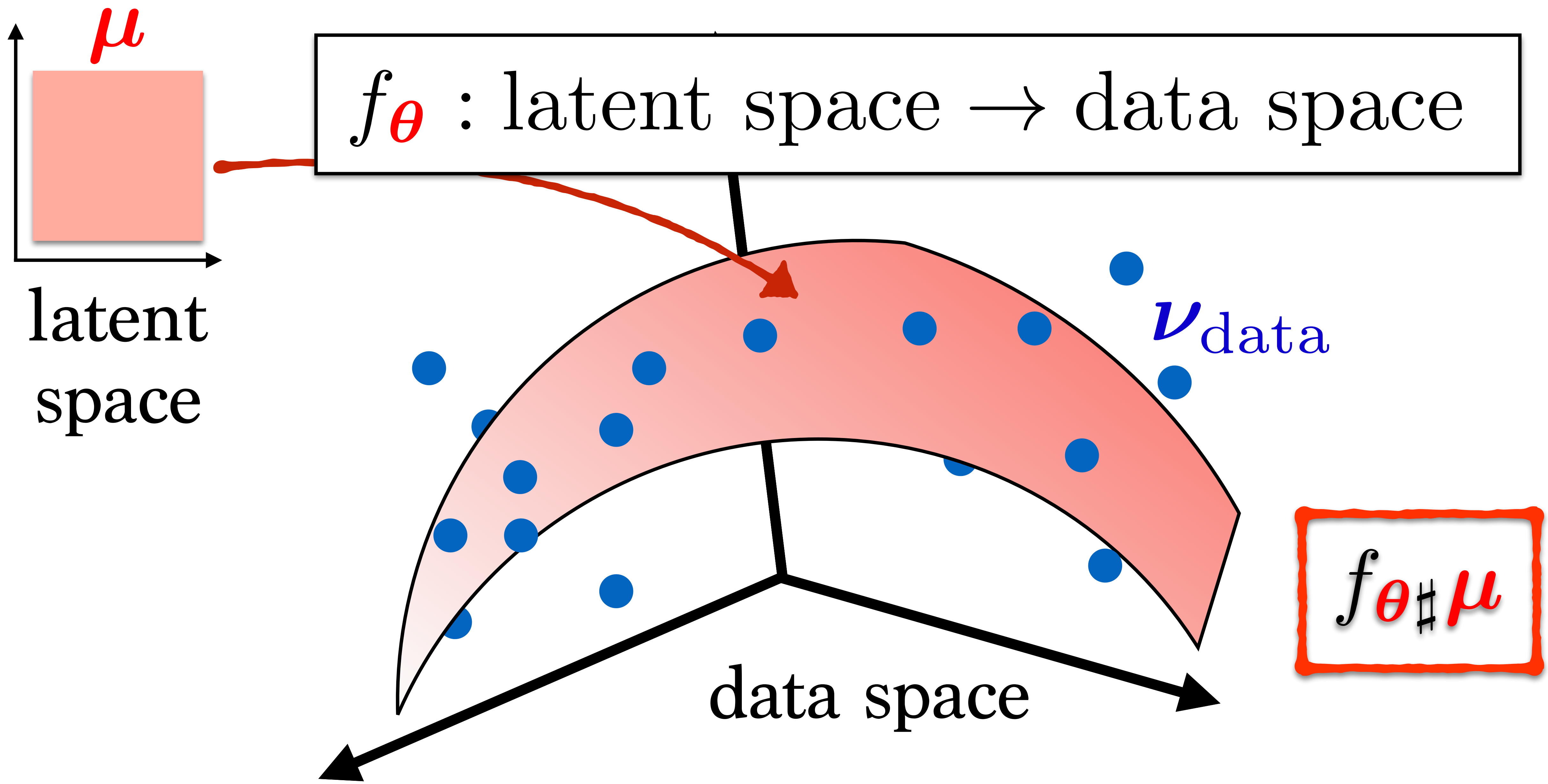
Generative Models



MLE

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

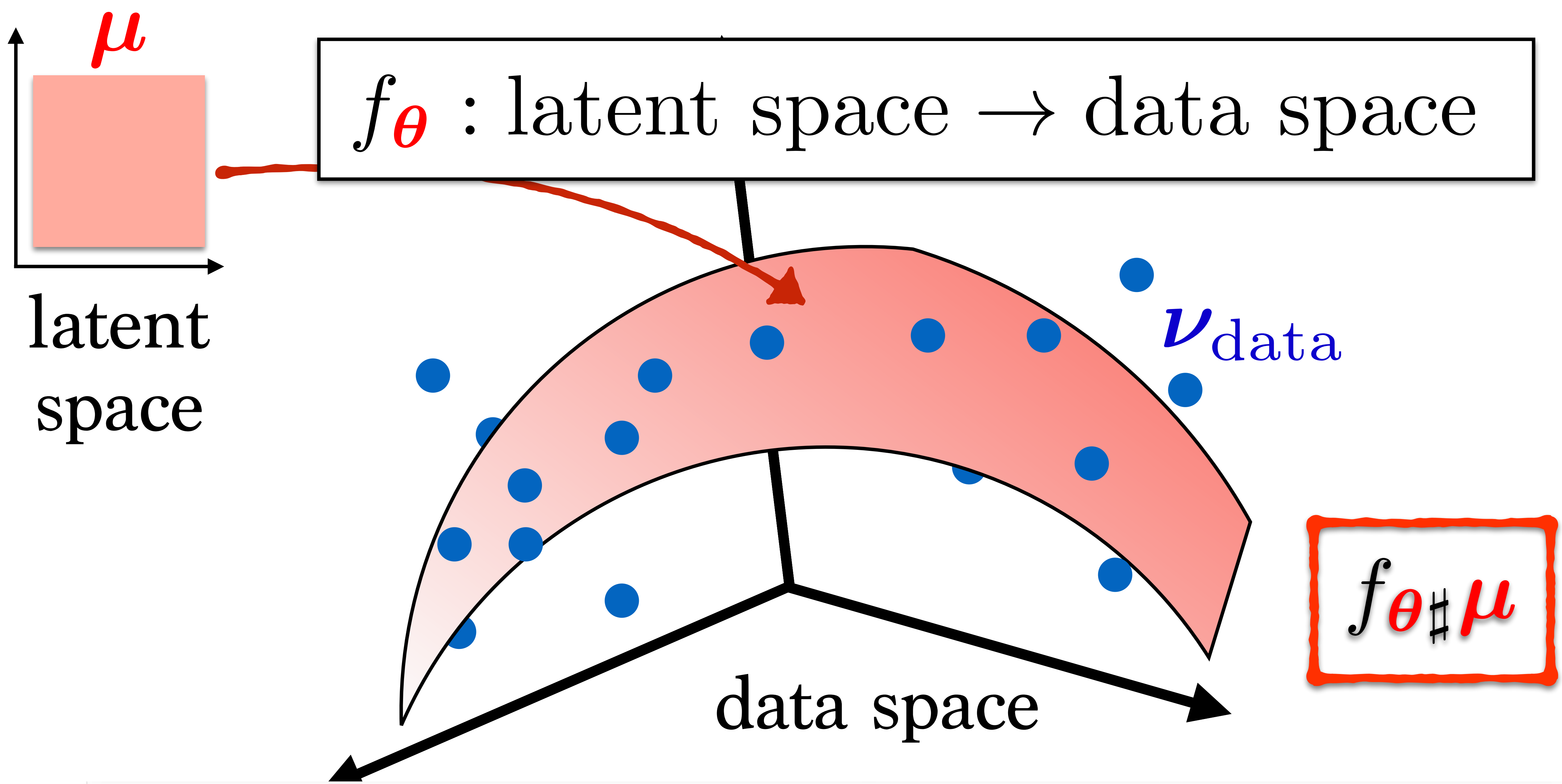
Generative Models



~~MLE~~

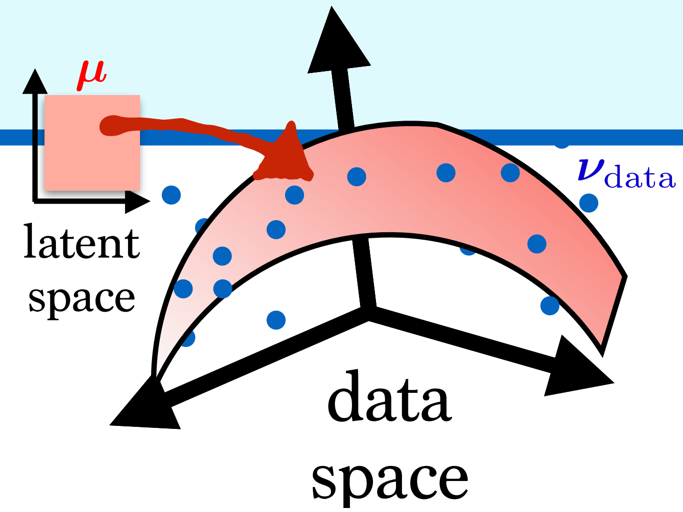
$$\max_{\theta\in\Theta} \frac{1}{N}\sum_{i=1}^{N} \log f_{\theta\sharp}\mu(x_i) \qquad \min_{\theta\in\Theta} \mathrm{KL}(\nu_{\text{data}} \| f_{\theta\sharp}\mu)$$

Generative Models



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

Workarounds?



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

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

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

$$\min_{\theta\in\Theta}\Delta(\nu_{\text{data}},p_\theta), \qquad \textbf{not} \min_{\theta\in\Theta}\text{KL}(\nu_{\text{data}}\|p_\theta)$$

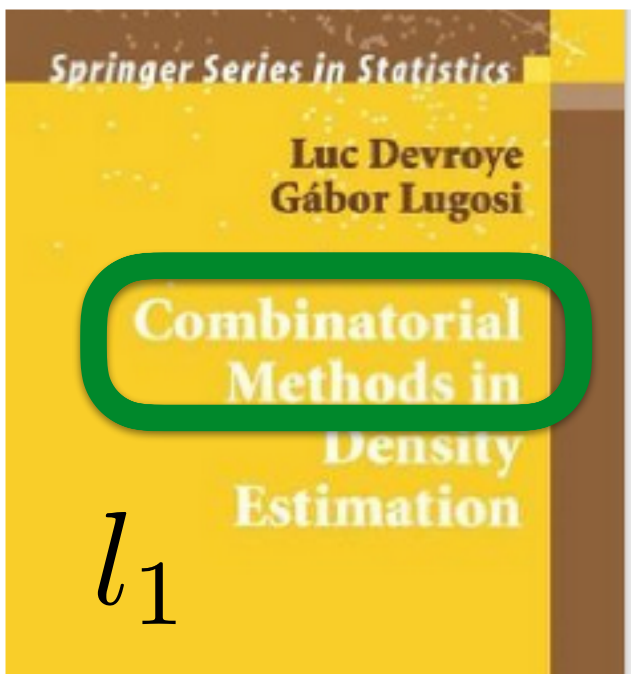
Minimum △ Estimation

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



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 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

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

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

△ 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

△ 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

Minimum Kantorovich Estimation

- Use optimal transport theory, namely *Wasserstein distances* to define discrepancy Δ.

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

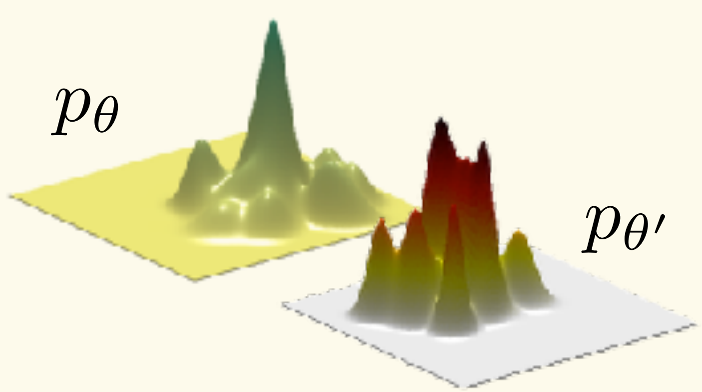
- Optimal transport? fertile field in mathematics.

Monge Kantorovich Koopmans Dantzig Brenier Otto McCann Villani

Nobel '75 Fields '10

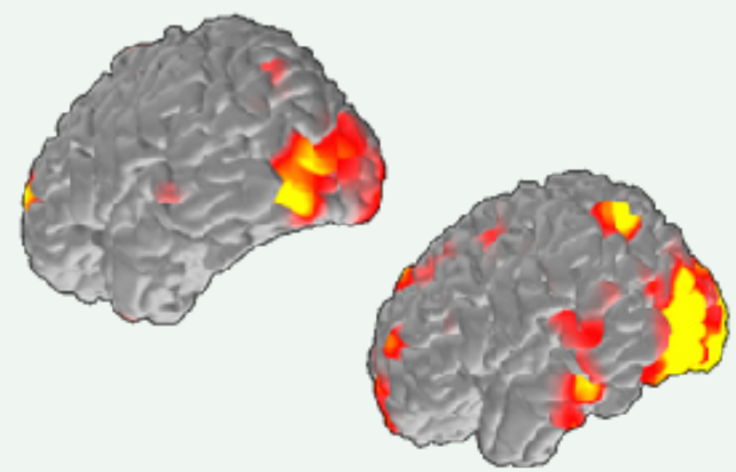
What is Optimal Transport?

The natural geometry for probability measures



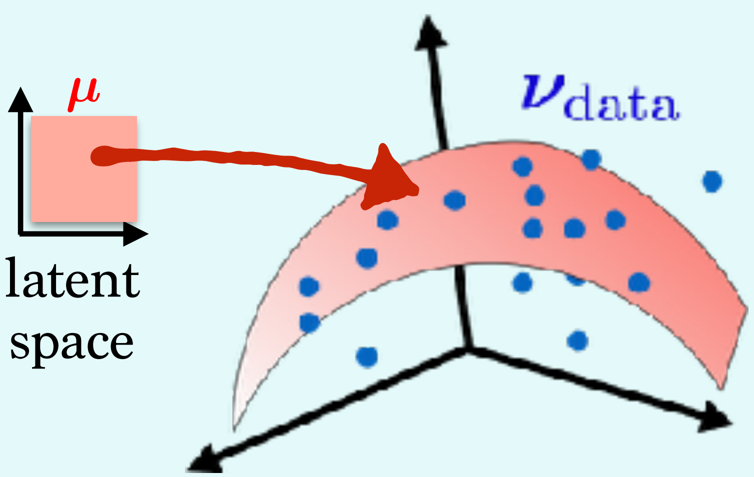
Statistical Models

Bags of features



Brain Activation Maps

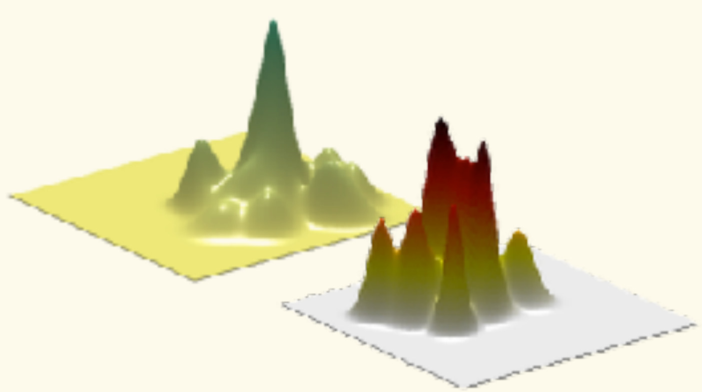
Generative Models vs. data



Color Histograms

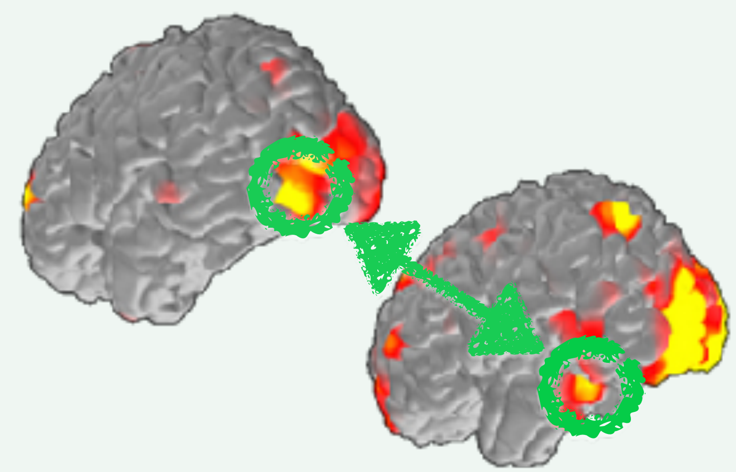
What is Optimal Transport?

The natural geometry for probability measures supported on a geometric space.



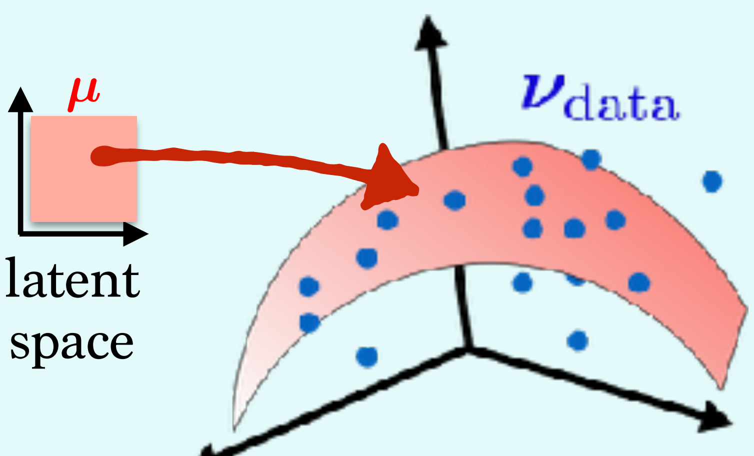
Statistical Models

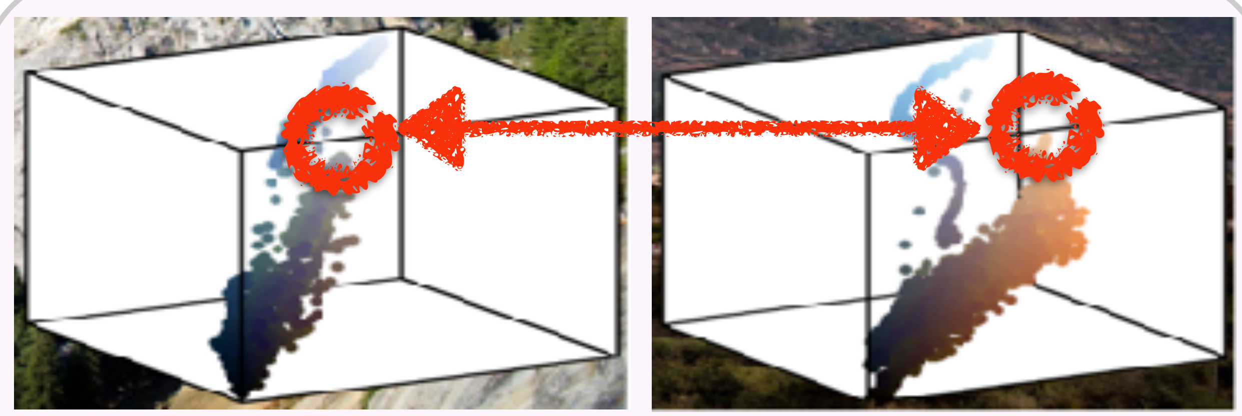
Bags of Features



Brain Activation Maps

Generative Models vs. Data

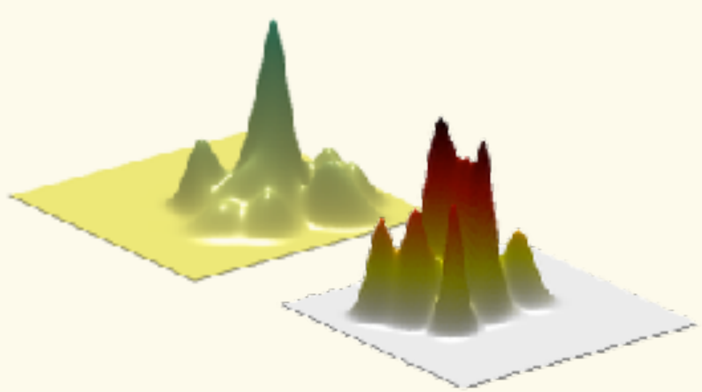




Color Histograms

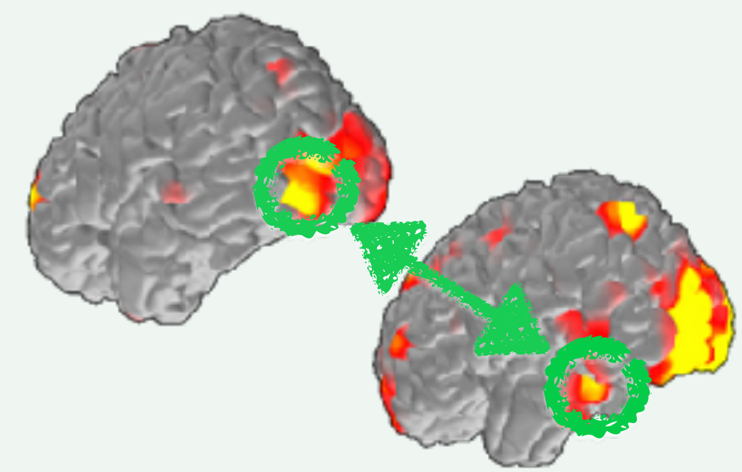
What is Optimal Transport?

The natural geometry for probability measures supported on a geometric space.



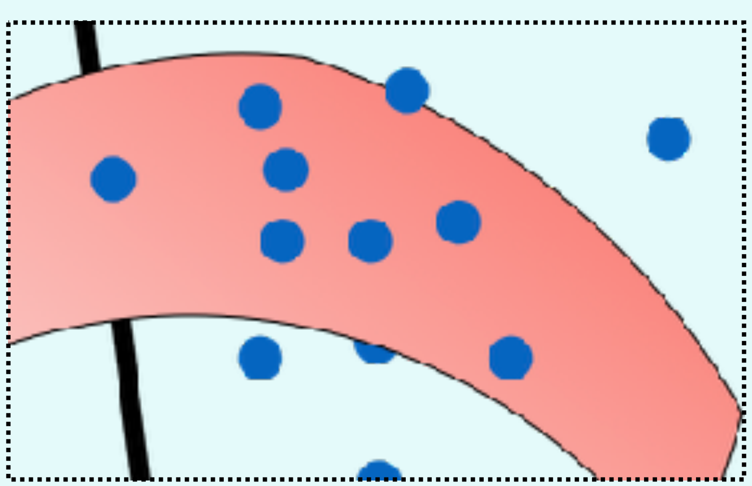
Statistical Models

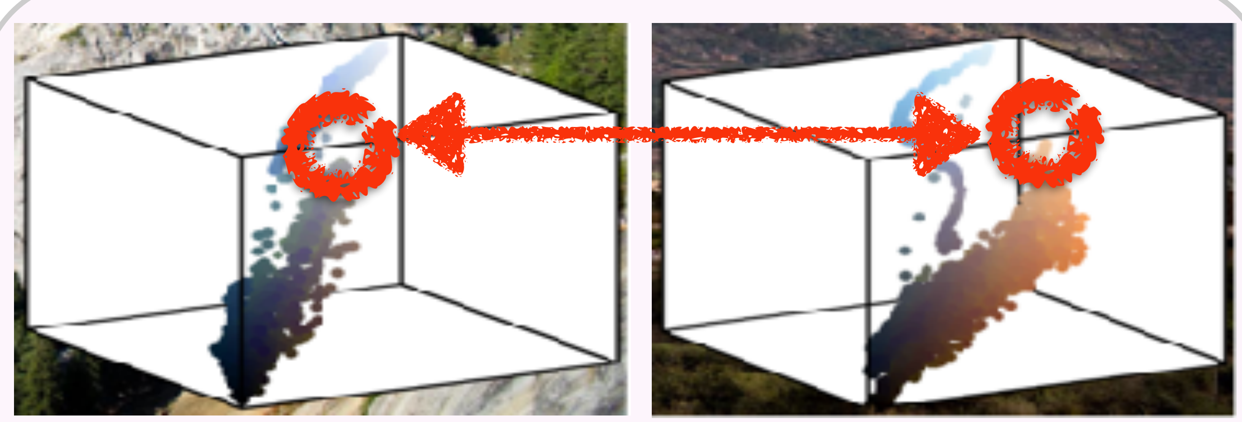
Bags of Features



Brain Activation Maps

Generative Models vs. Data

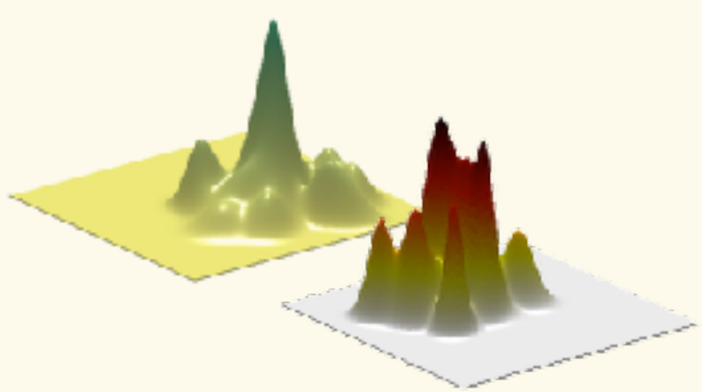




Color Histograms

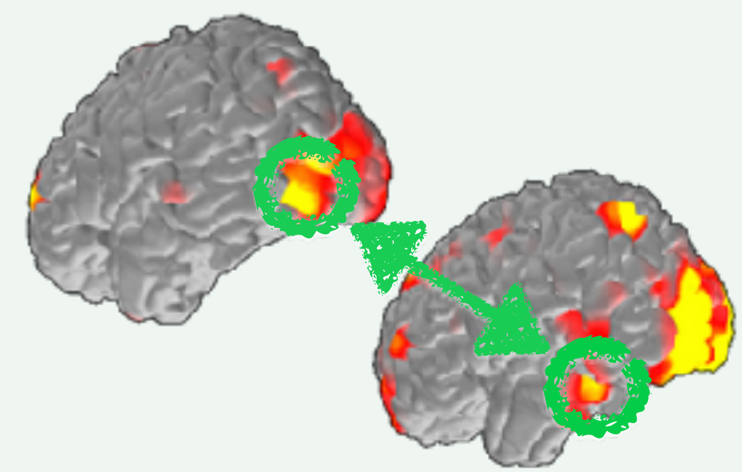
What is Optimal Transport?

The natural geometry for probability measures supported on a geometric space.



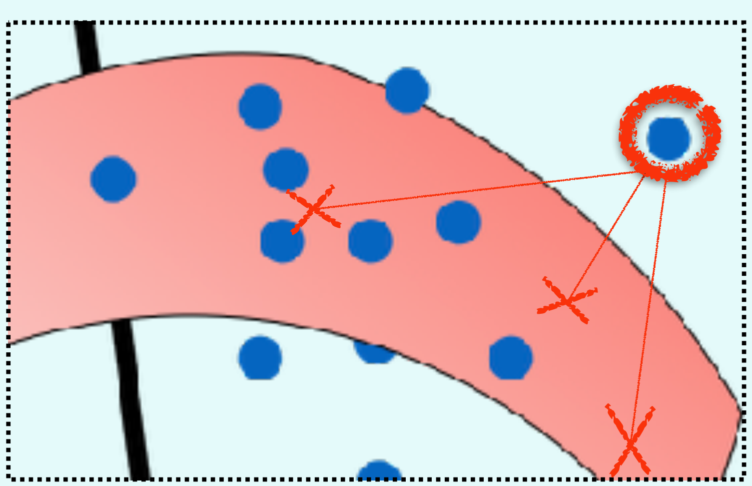
Statistical Models

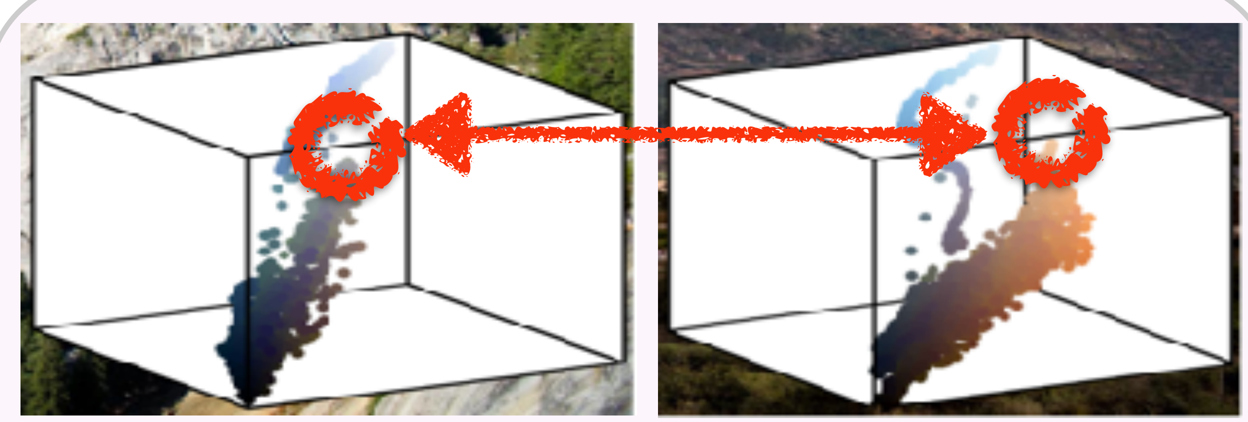
Bags of Features



Brain Activation Maps

Generative Models vs. Data





Color Histograms

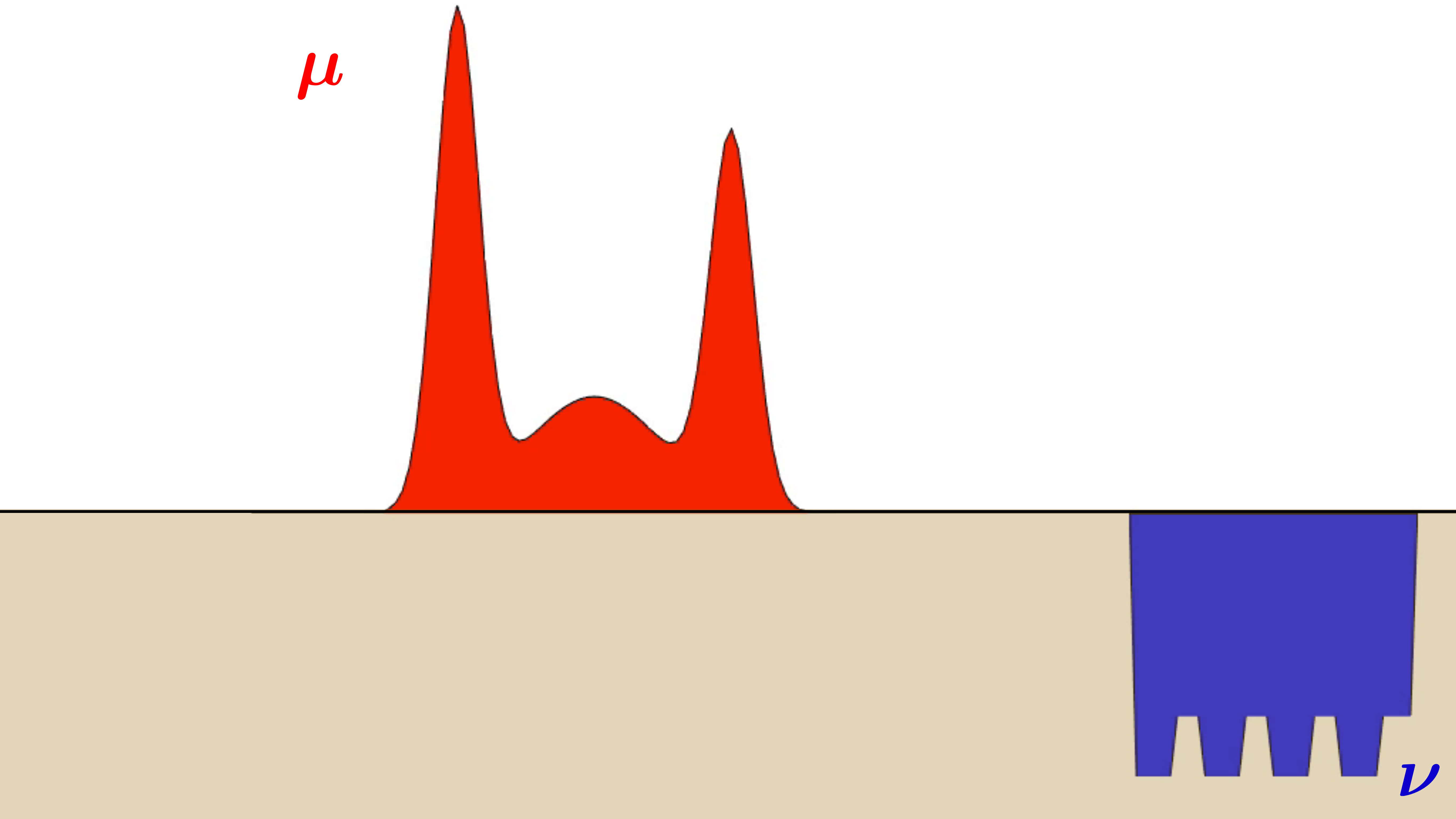
Origins: Monge Problem (1781)

MÉMOIRE
SUR LA
THÉORIE DES DÉBLAIS ET DES REMBLAIS.

Par M. MONGE.

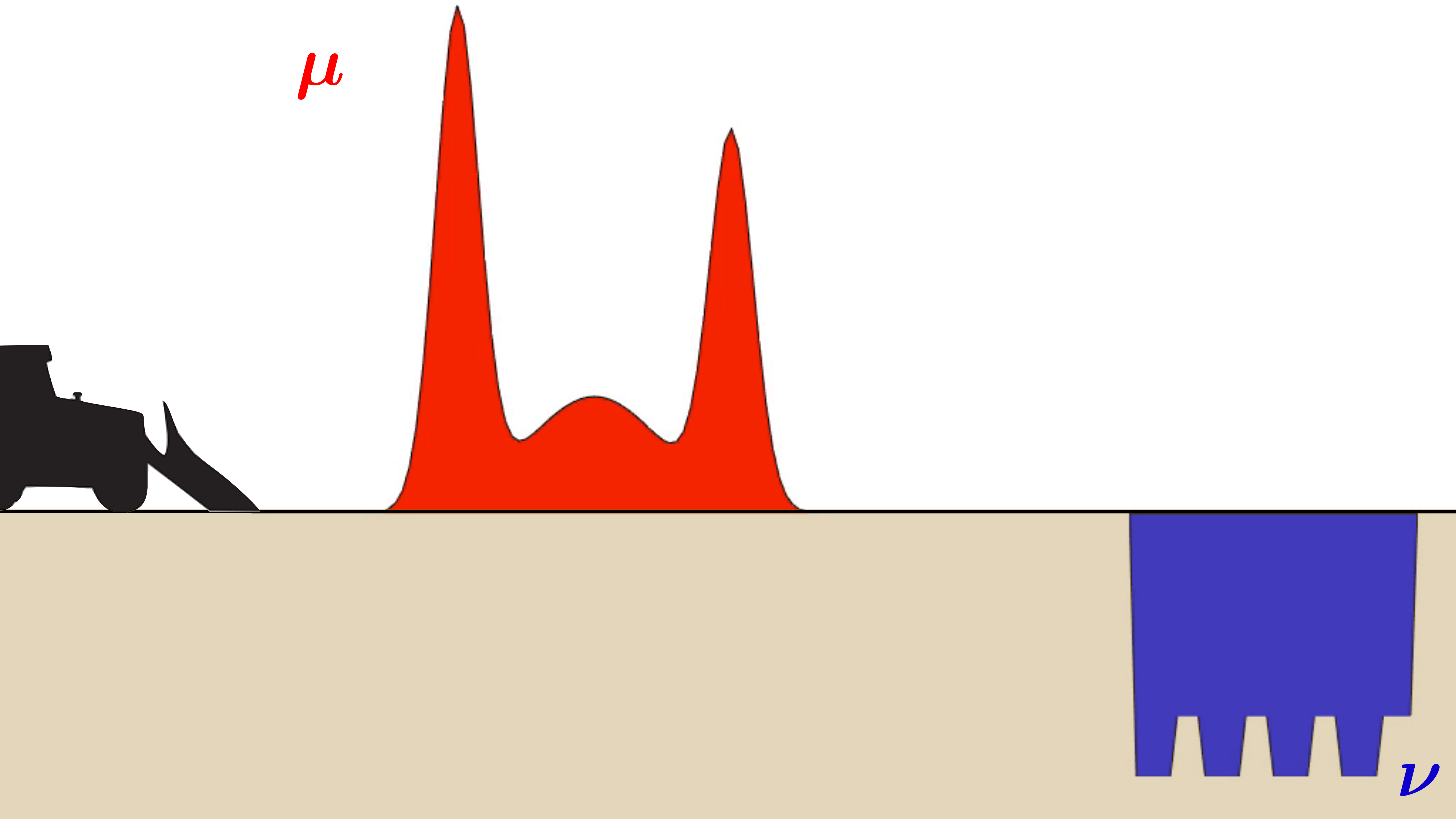
LORSQU'ON doit tranſporter des terres d'un lieu dans un autre, on a coutume de donner le nom de *Déblai* au volume des terres que l'on doit tranſporter, & le nom de *Remblai* à l'eſpace qu'elles doivent occuper après le tranſport.

Origins: Monge Problem



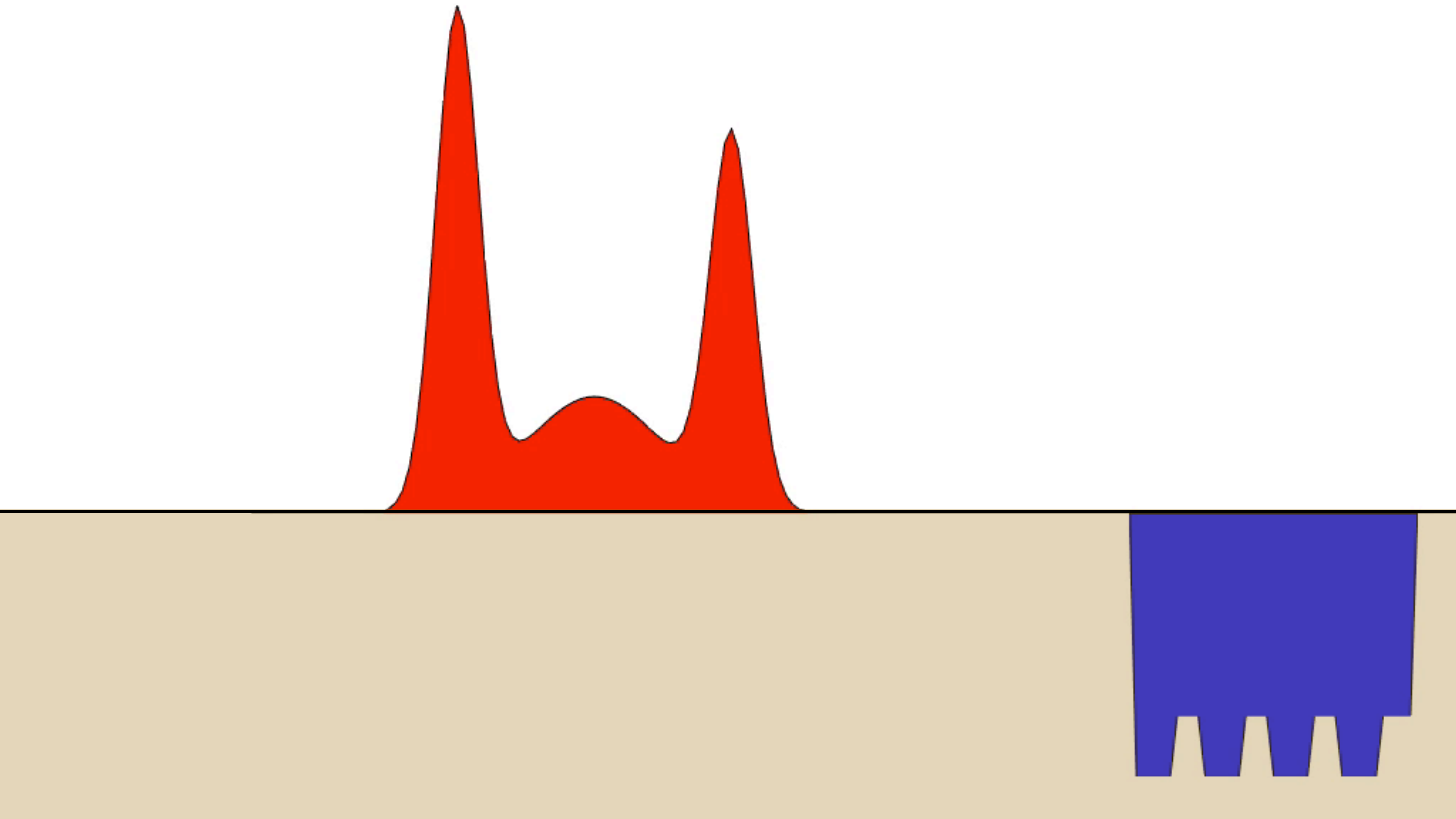
Origins: Monge Problem

In the 21st Century...



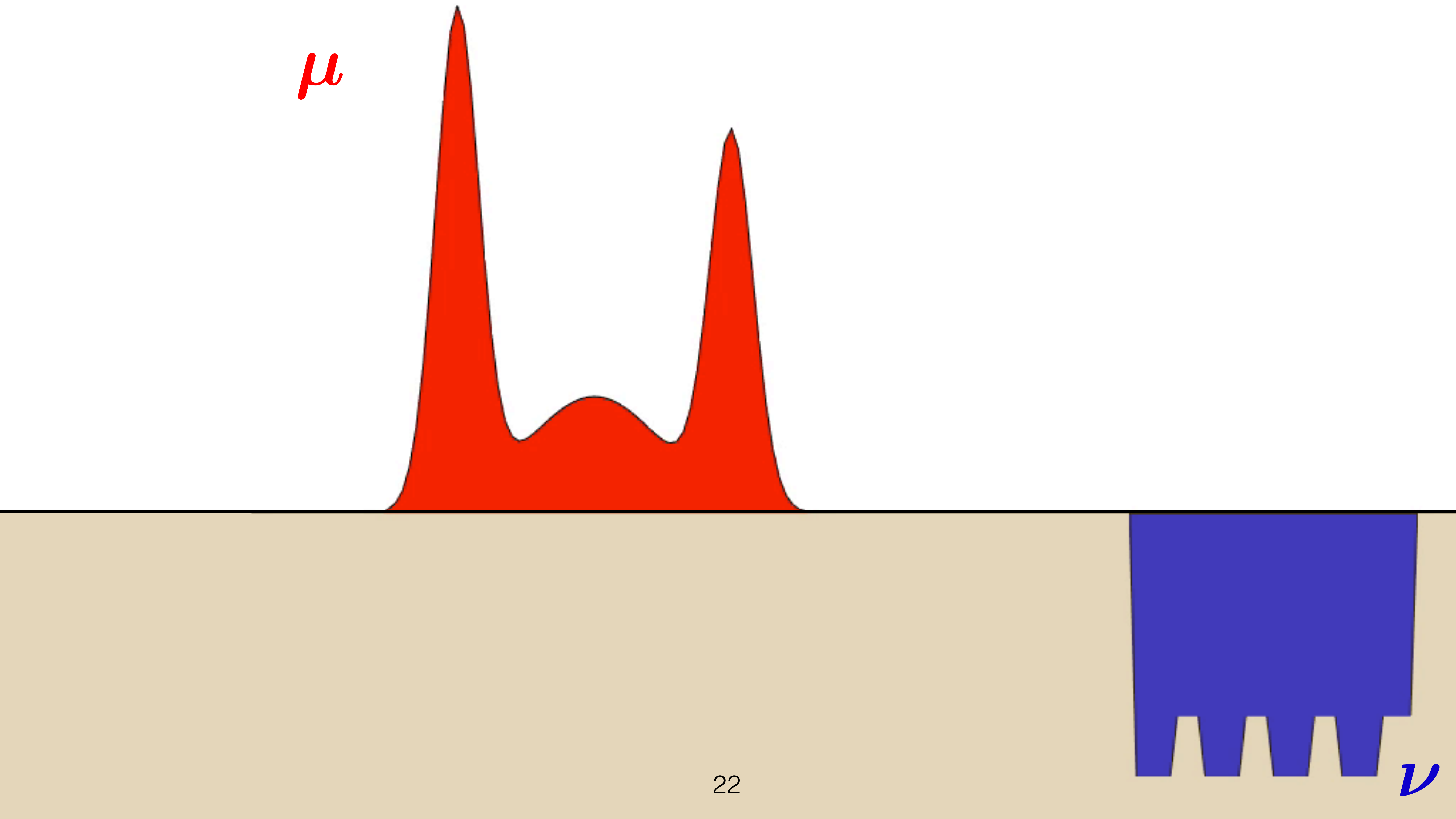
Origins: Monge Problem

In the 21st Century...



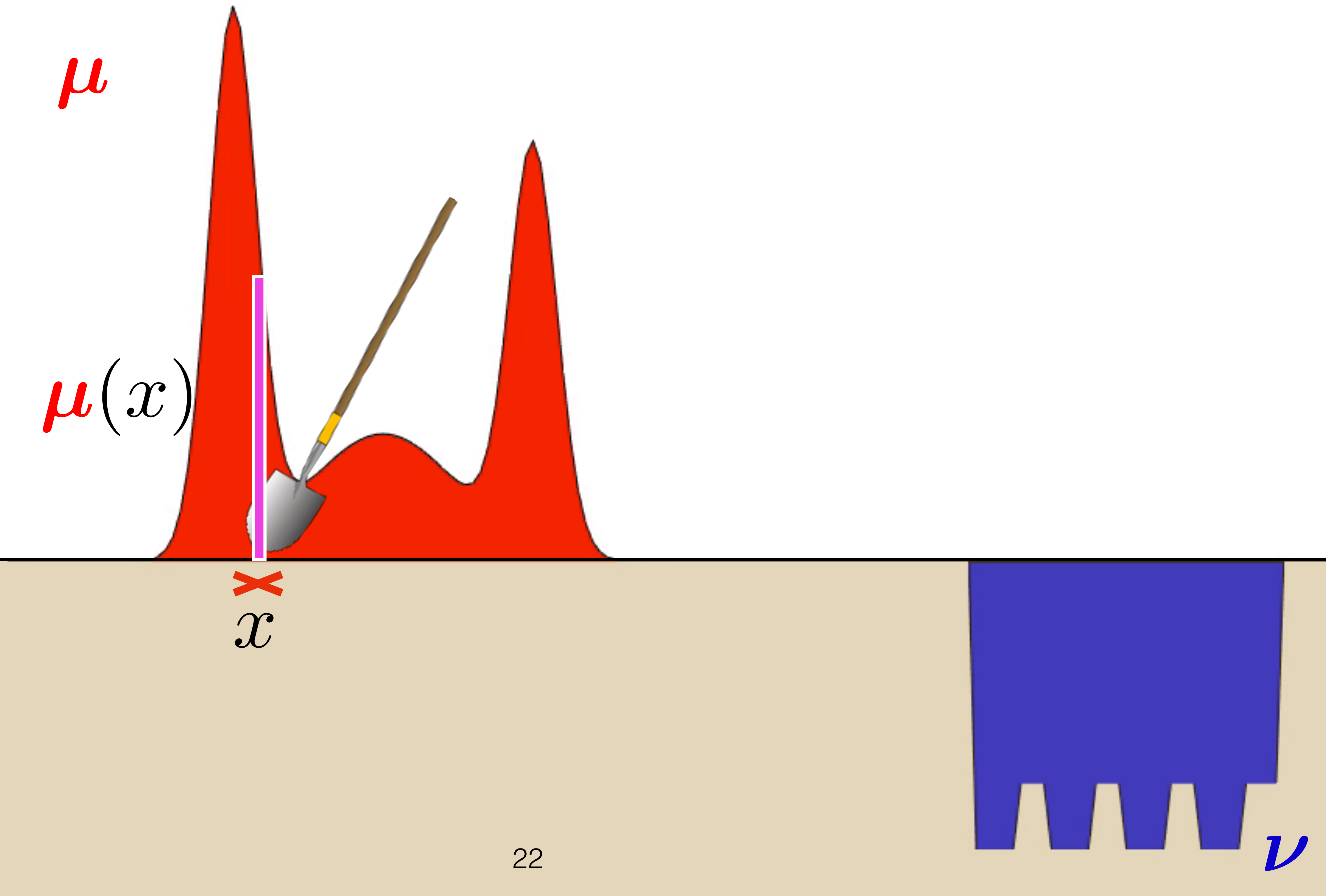
Origins: Monge's Problem

In 1781 however…



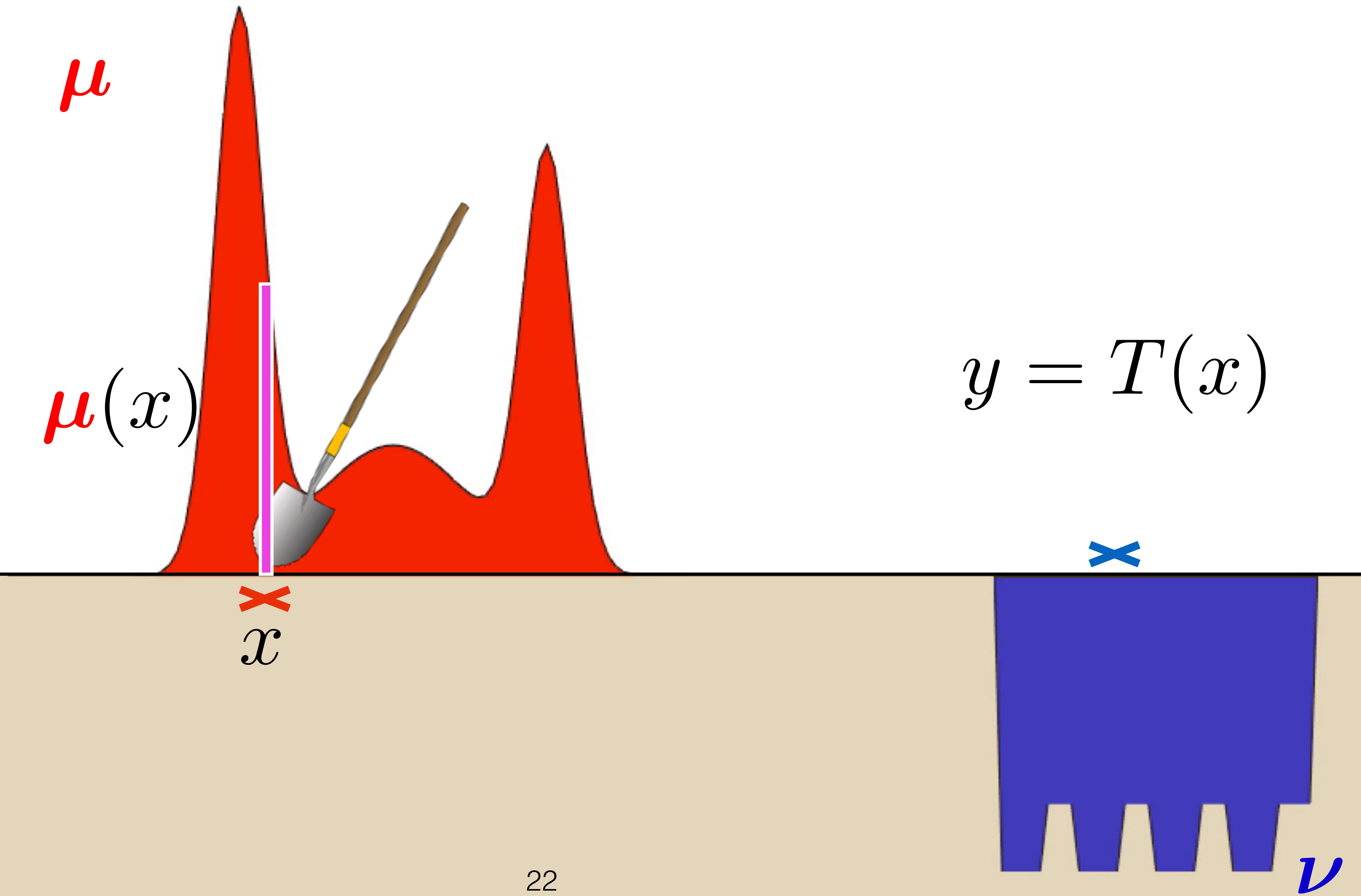
Origins: Monge's Problem

In 1781 however…



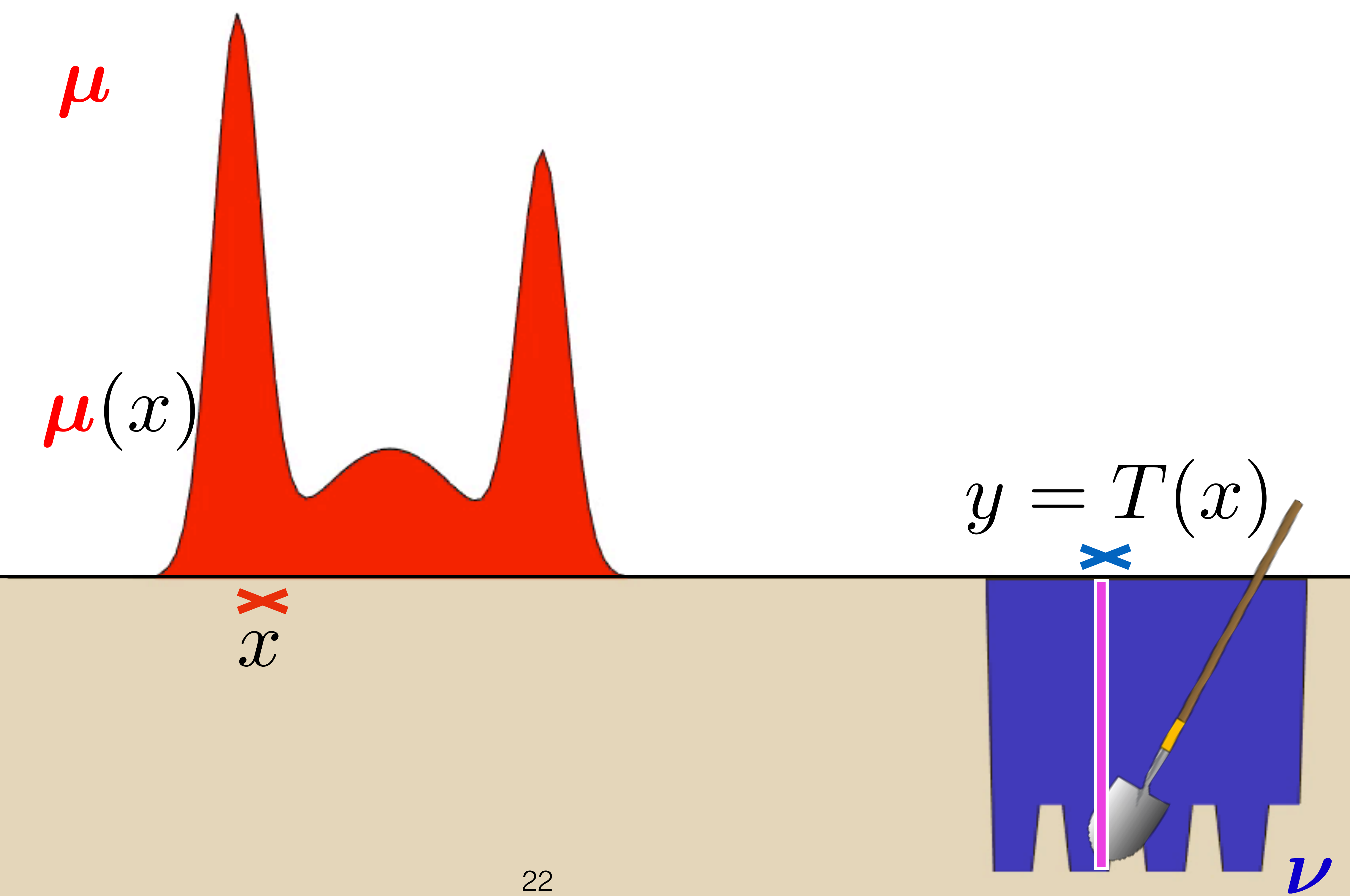
Origins: Monge's Problem

In 1781 however…



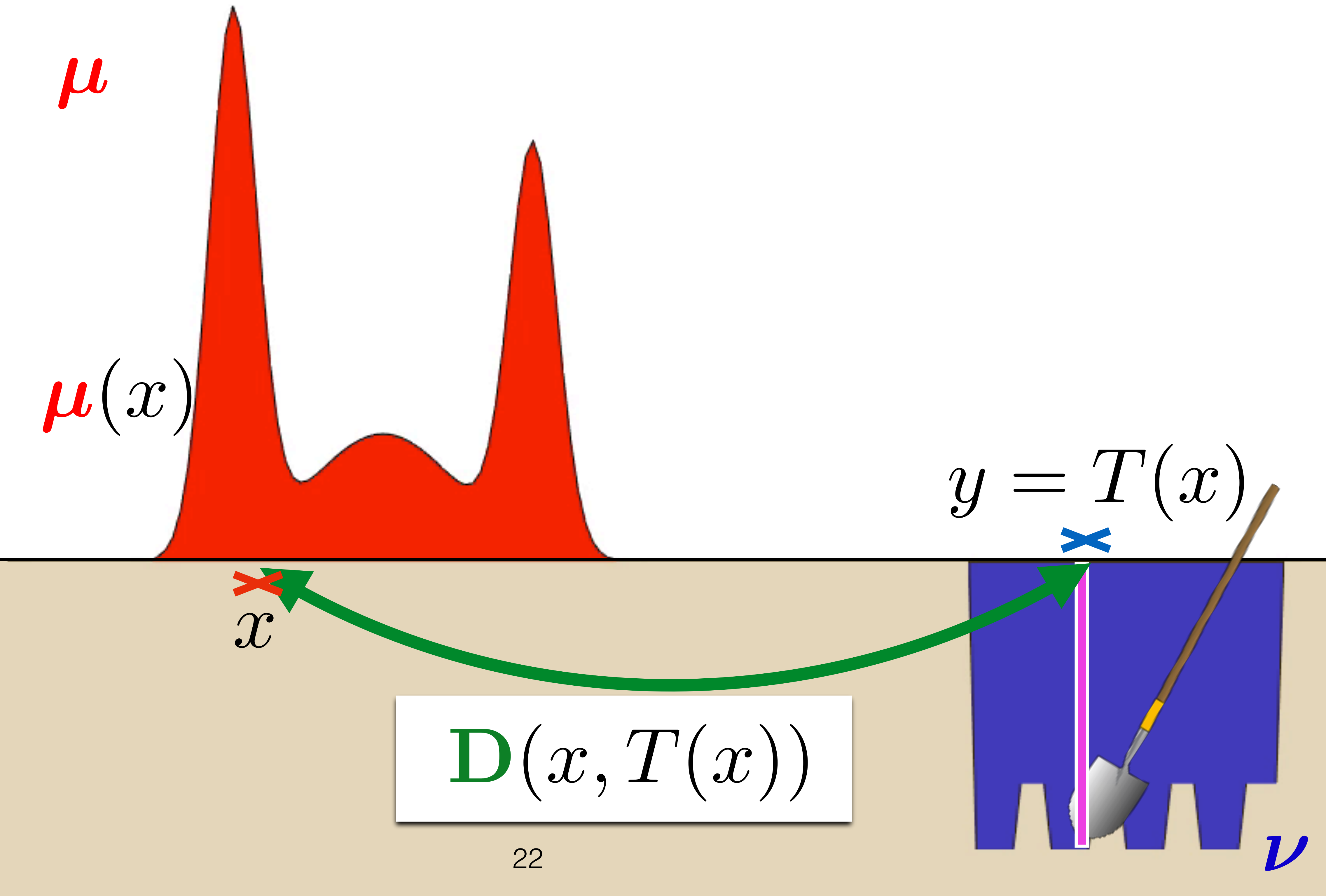
Origins: Monge's Problem

In 1781 however…



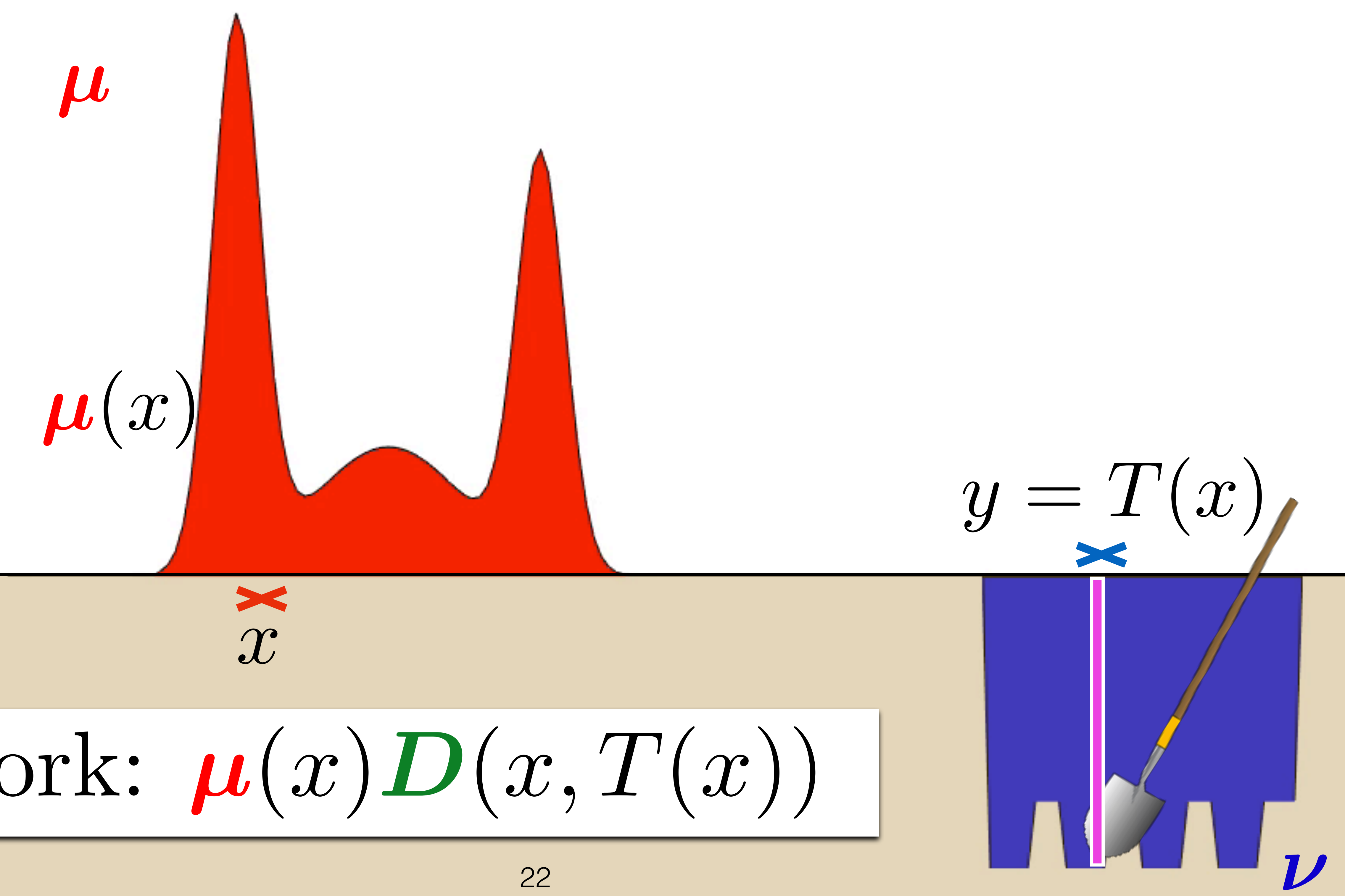
Origins: Monge's Problem

In 1781 however...



Origins: Monge's Problem

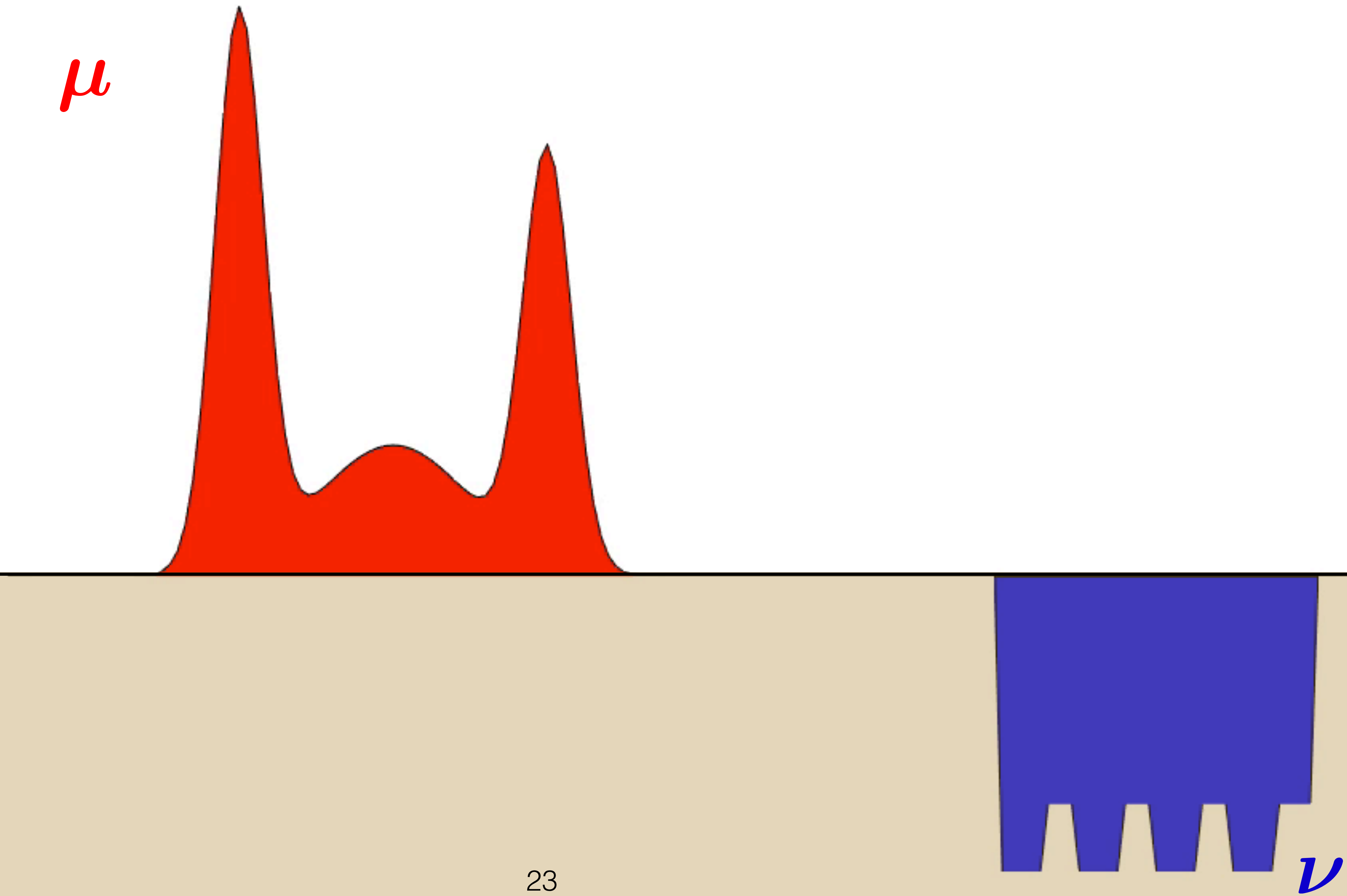
In 1781 however...



work: $\mu(x) D(x, T(x))$

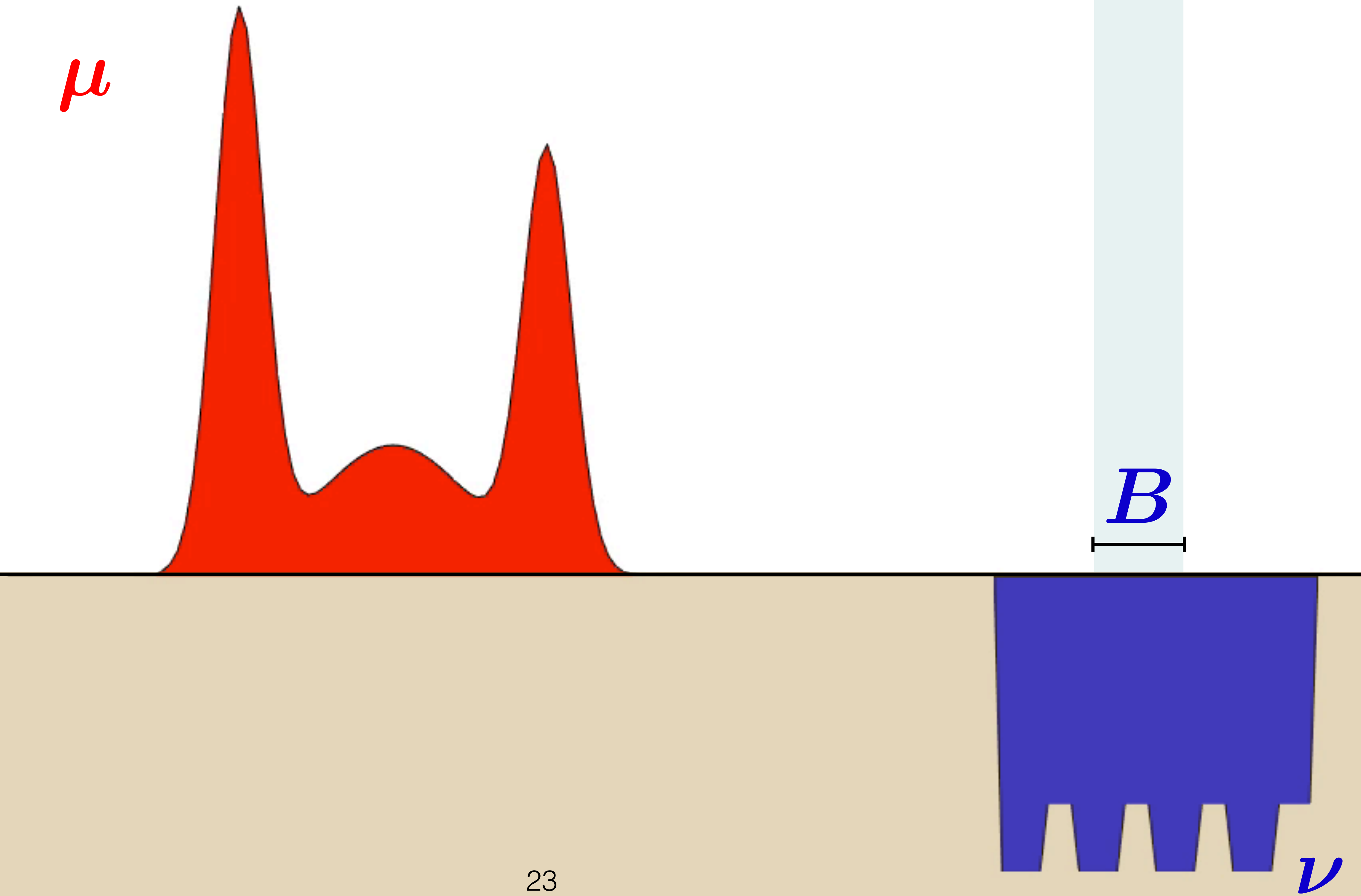
Origins: Monge's Problem

T must map red to blue.



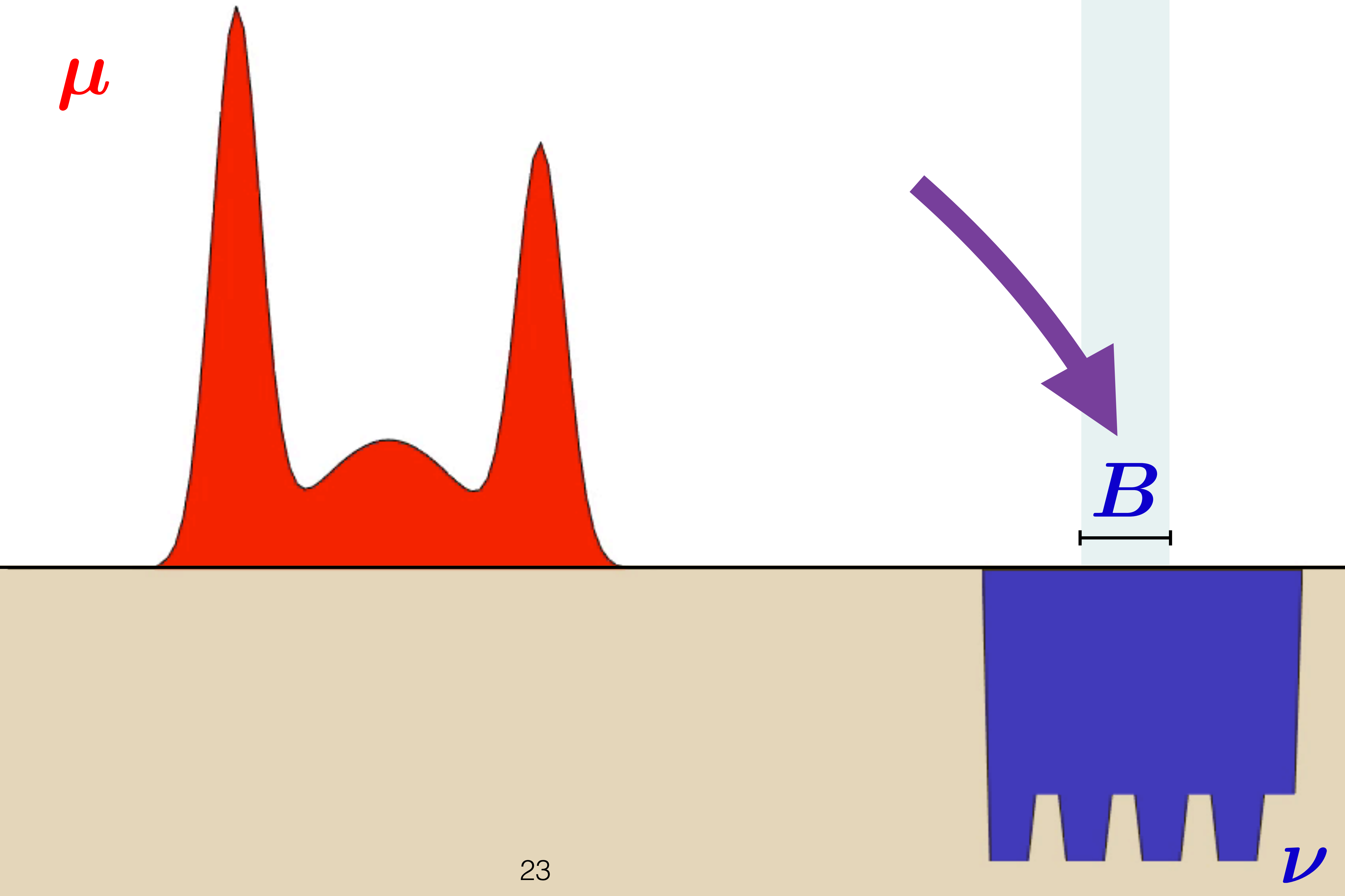
Origins: Monge's Problem

T must map red to blue.



Origins: Monge's Problem

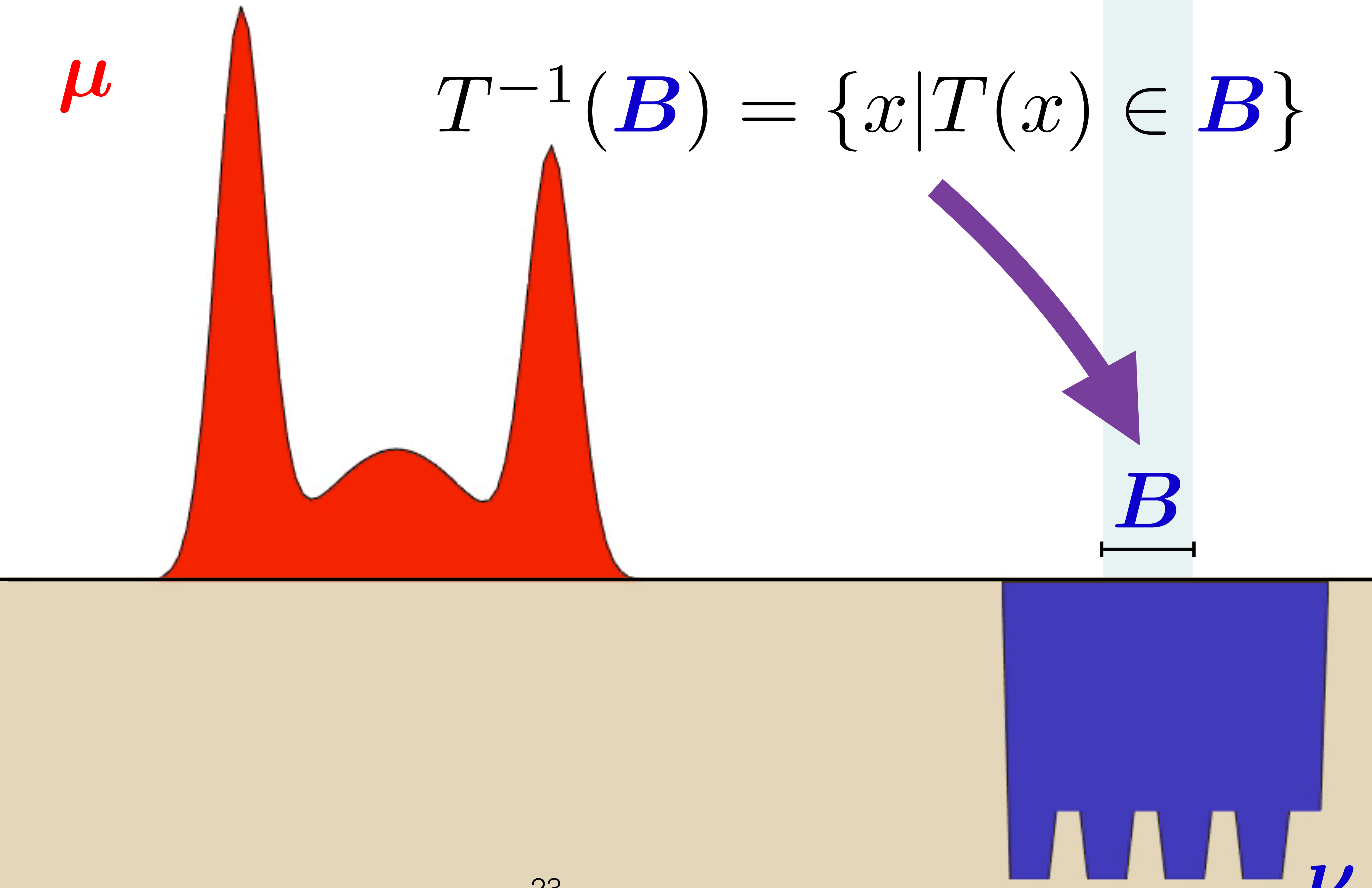
T must map red to blue.



Origins: Monge's Problem

T must map red to blue.

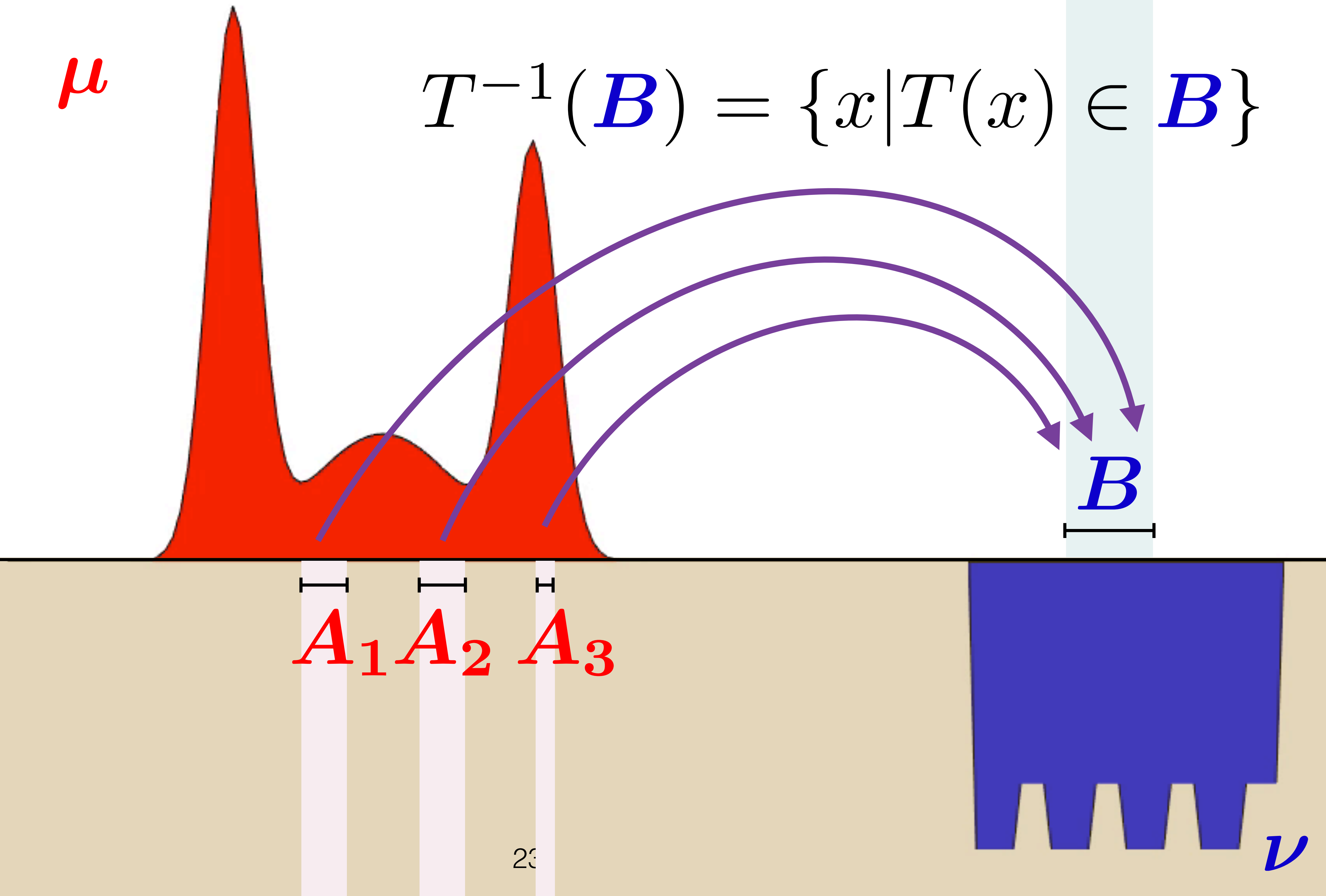
$$T^{-1}(\boldsymbol{B}) = \{x | T(x) \in \boldsymbol{B}\}$$



Origins: Monge's Problem

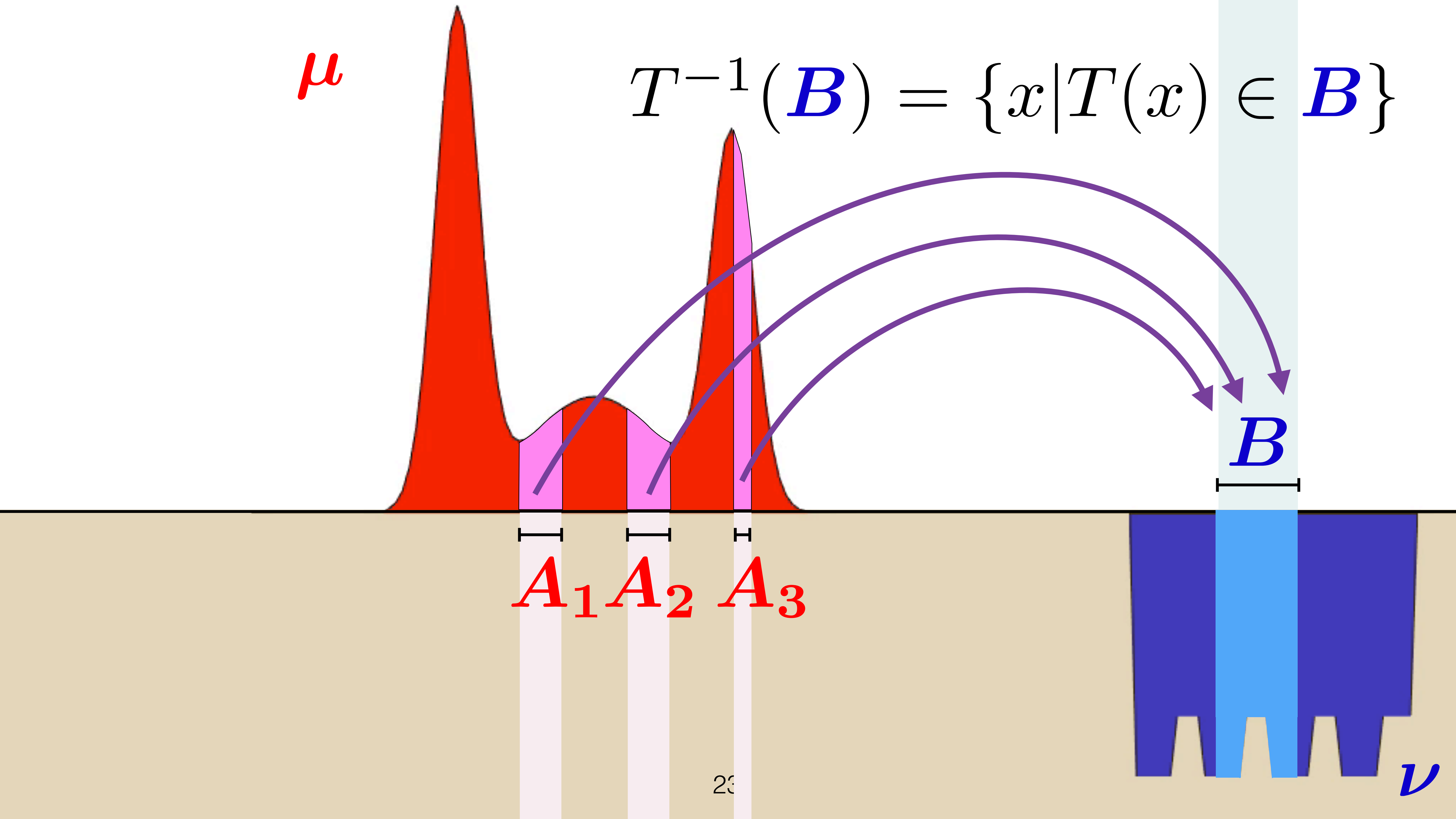
T must map red to blue.

$$T^{-1}(\boldsymbol{B}) = \{x | T(x) \in \boldsymbol{B}\}$$



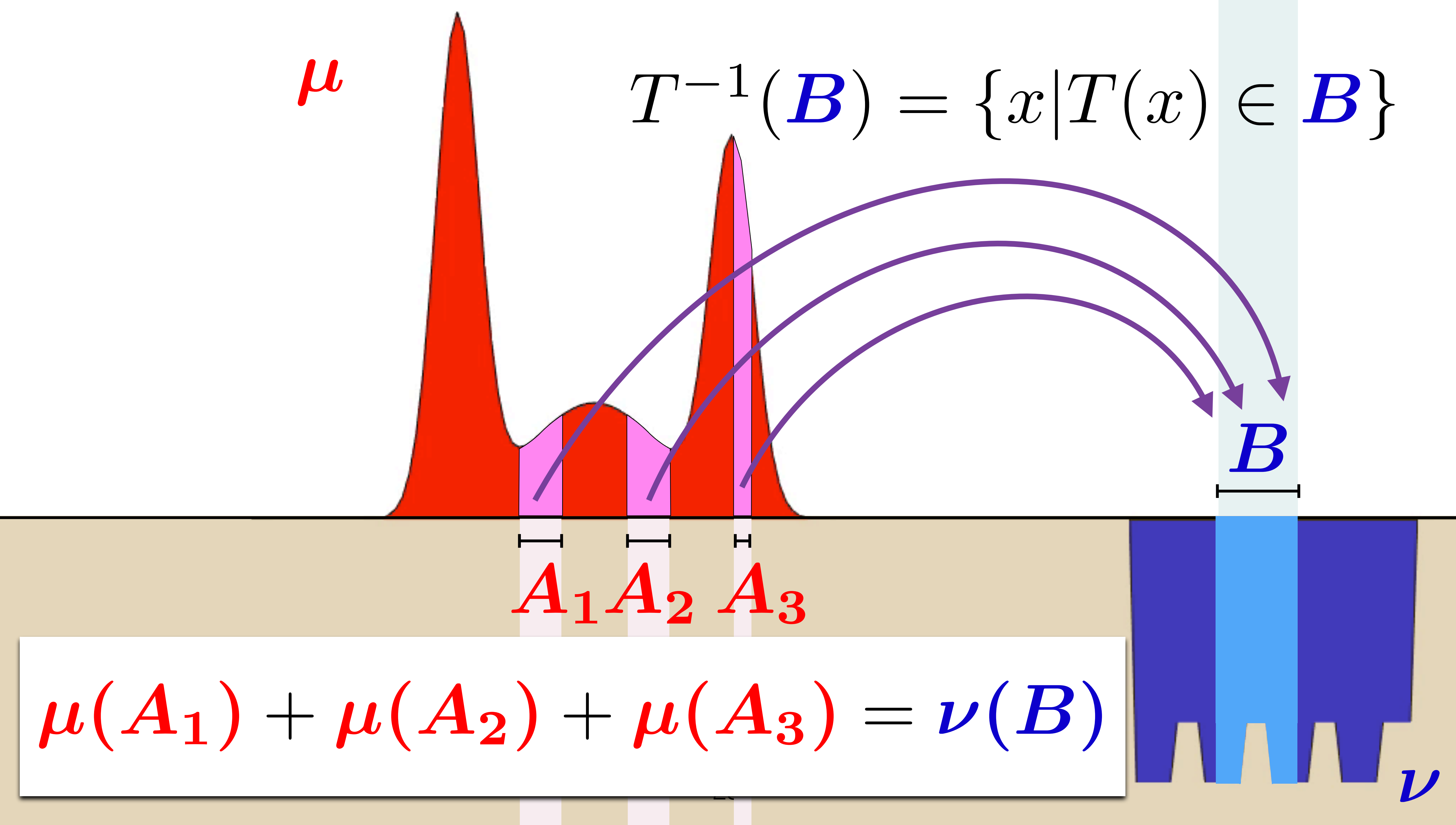
Origins: Monge's Problem

T must map red to blue.



Origins: Monge's Problem

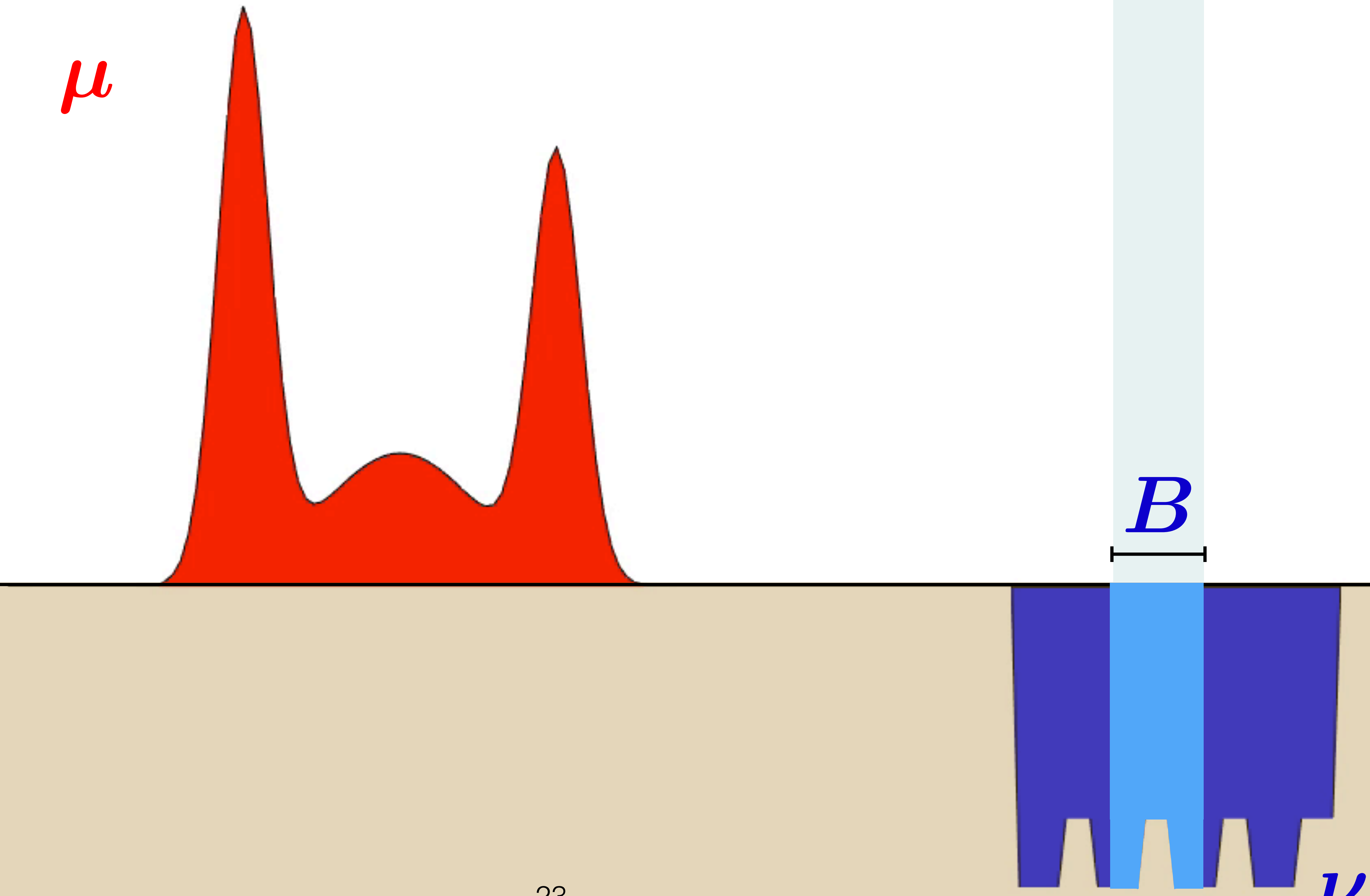
T must map red to blue.



$$\mu(A_1) + \mu(A_2) + \mu(A_3) = \nu(B)$$

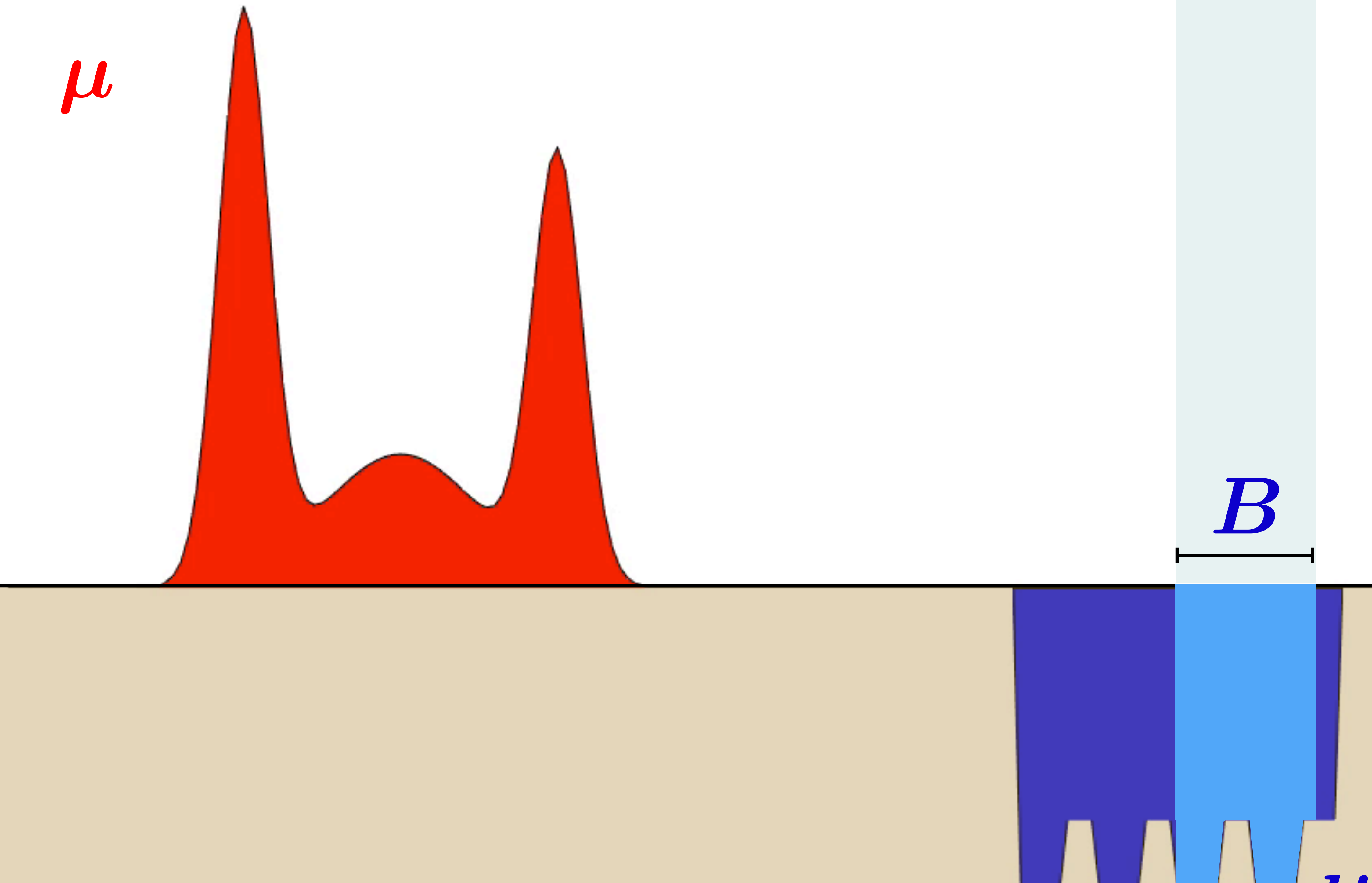
Origins: Monge's Problem

T must map red to blue.



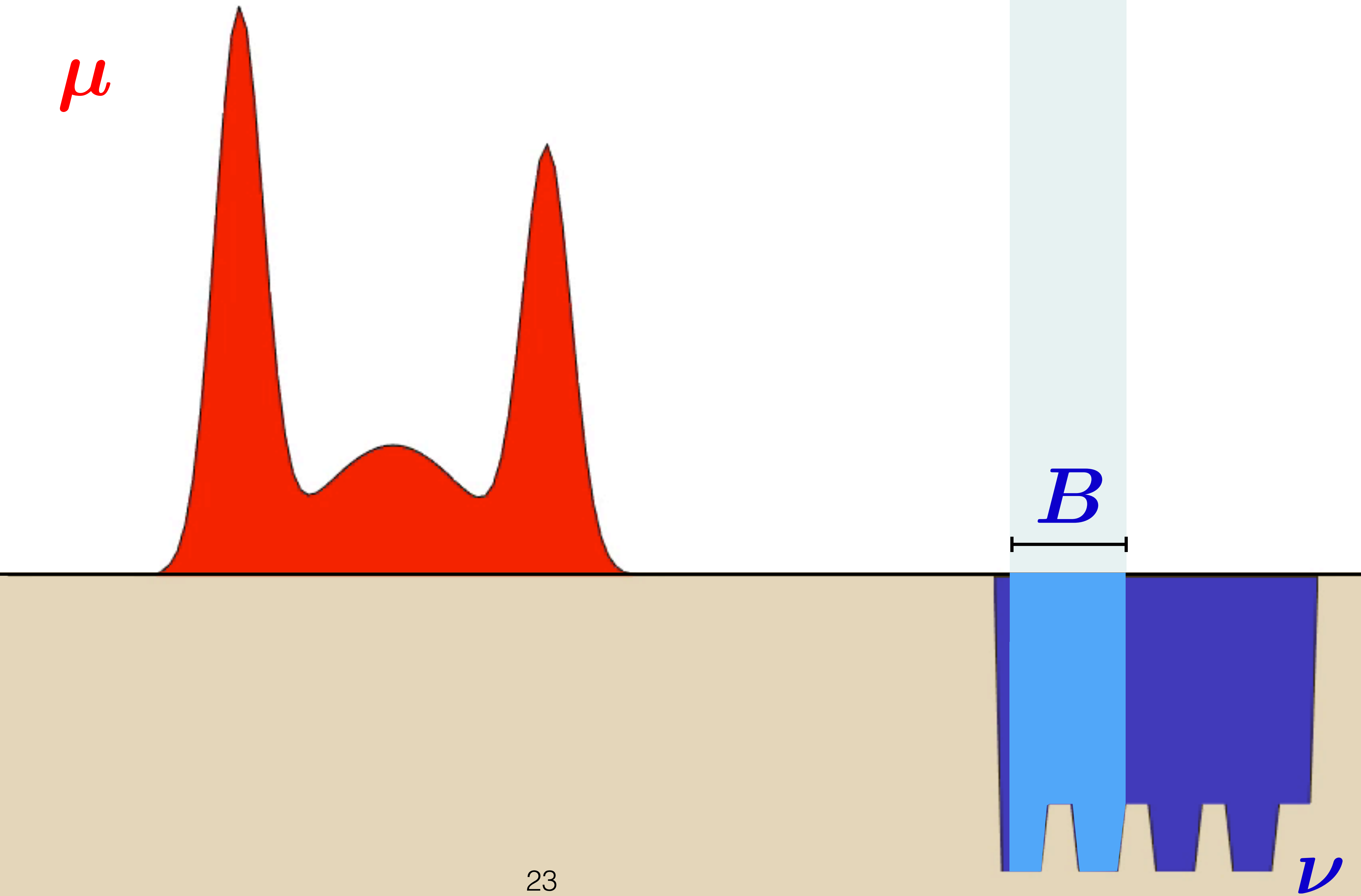
Origins: Monge's Problem

T must map red to blue.



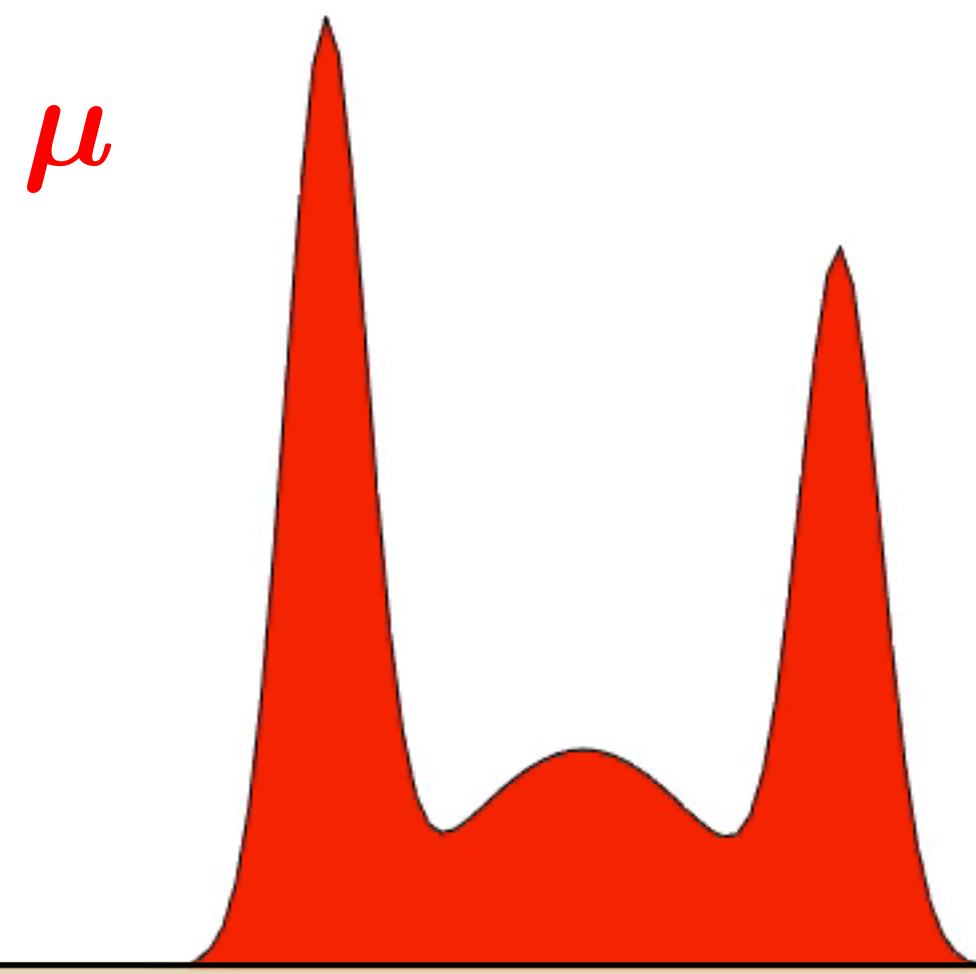
Origins: Monge's Problem

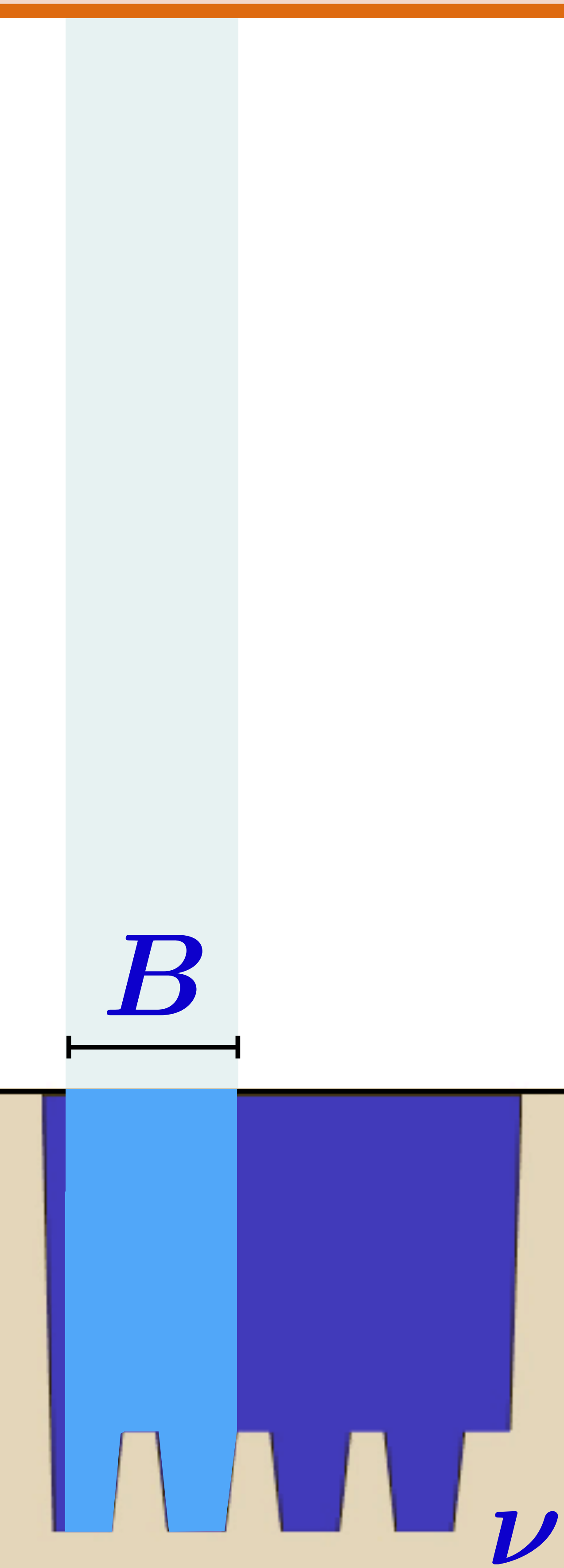
T must map red to blue.



Origins: Monge's Problem

T must map red to blue.

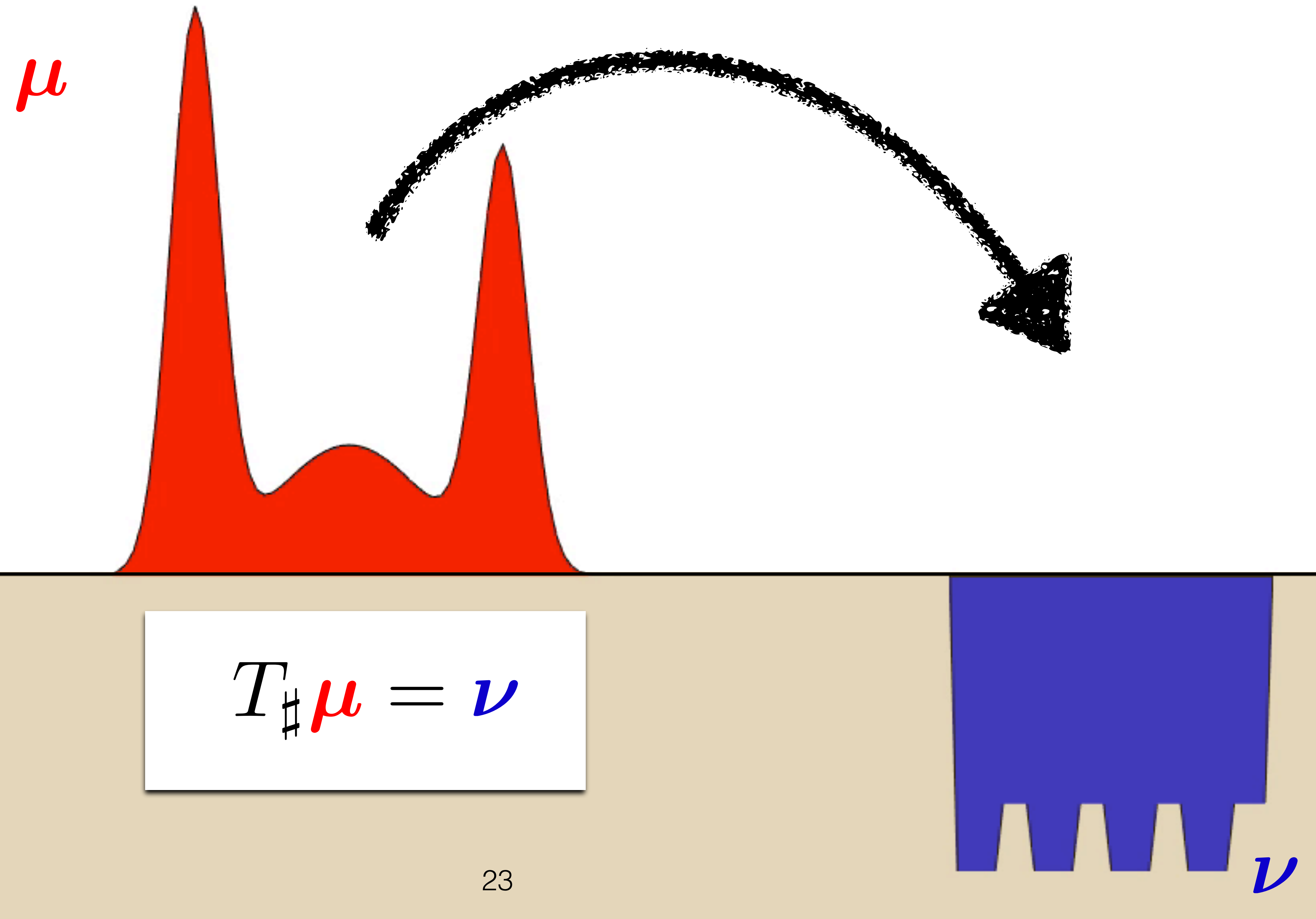




$$\forall B, \mu(T^{-1}(B)) = \nu(B)$$

Origins: Monge's Problem

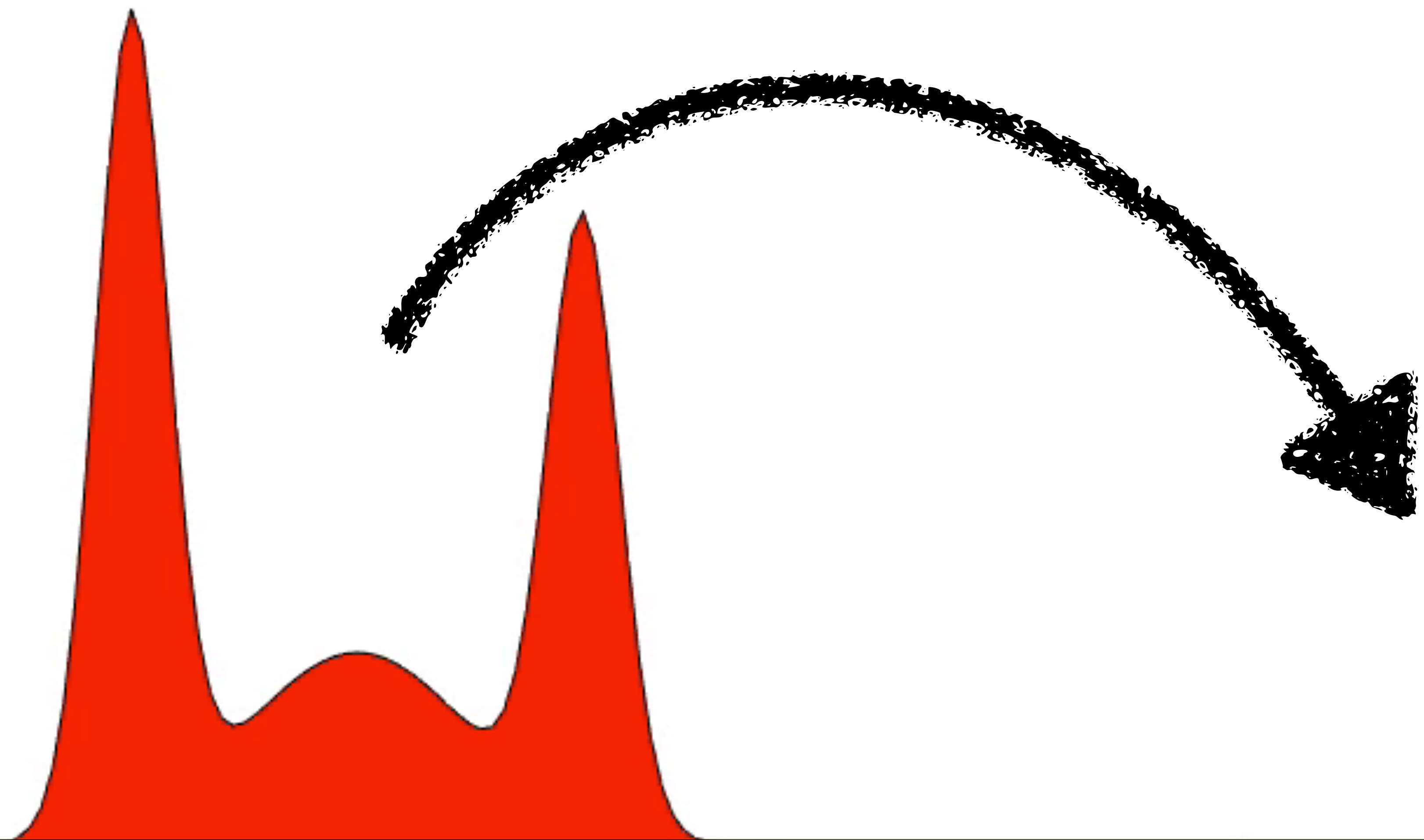
T must **push-forward** *the red measure towards the blue*



$$T_\sharp \mu = \nu$$

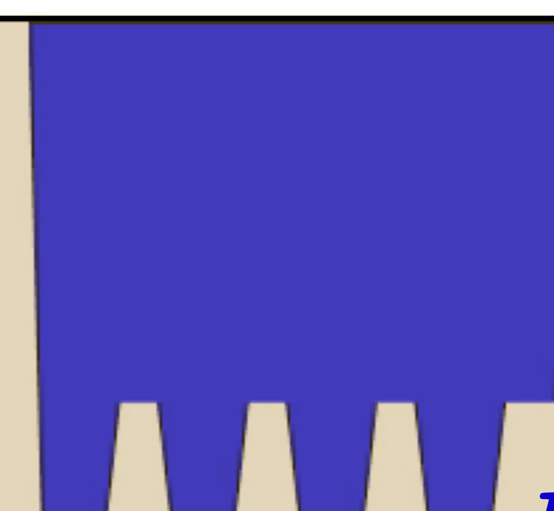
Origins: Monge's Problem

T *must* **push-forward** *the red measure towards the blue*



What T s.t. $T_\sharp \mu = \nu$

minimizes $\int D(x, T(x)) \mu(dx)$?

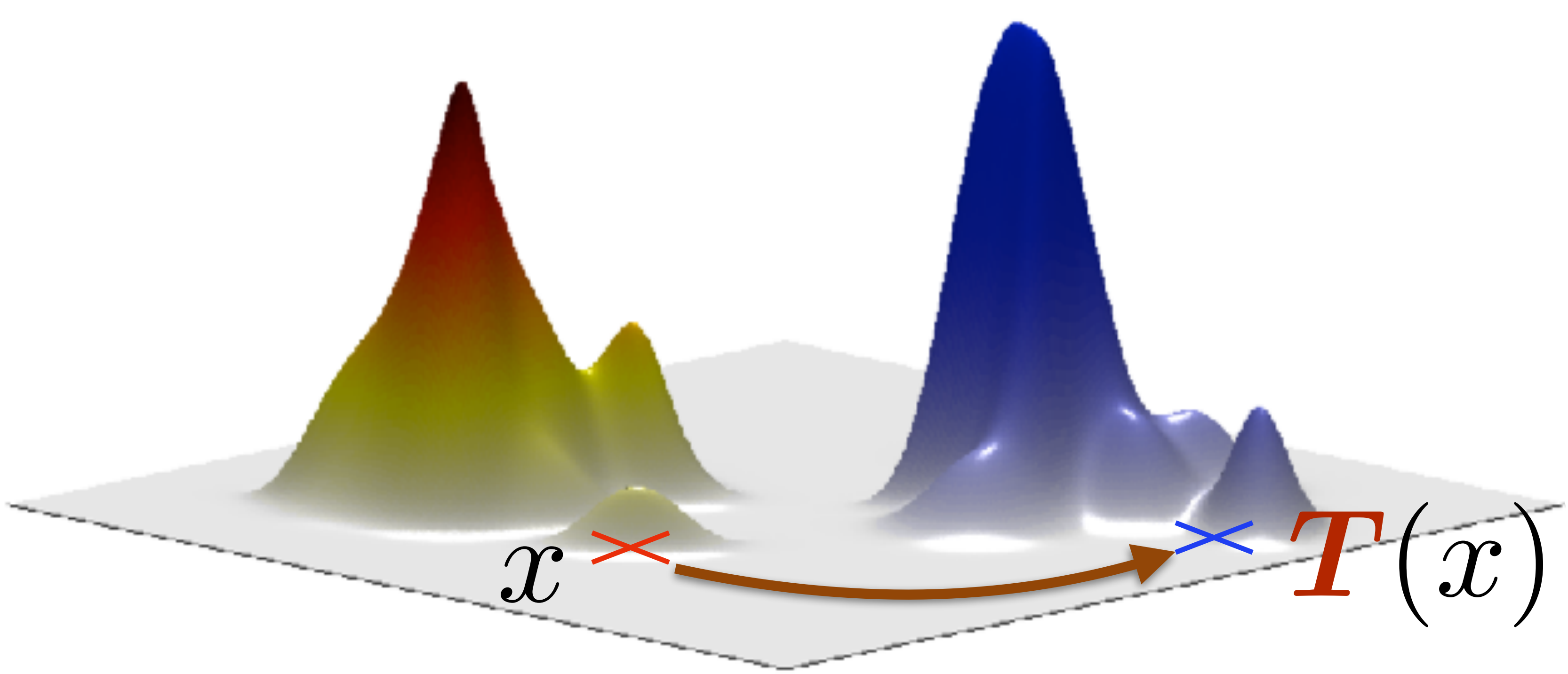


Monge Problem

Ω a probability space, $\boldsymbol{c} : \Omega \times \Omega \to \mathbb{R}$.
$\boldsymbol{\mu}, \boldsymbol{\nu}$ two probability measures in $\mathcal{P}(\Omega)$.

[Monge'81] problem: find a map $\boldsymbol{T} : \Omega \to \Omega$

$$\inf_{\boldsymbol{T}_\sharp \boldsymbol{\mu} = \boldsymbol{\nu}} \int_\Omega \boldsymbol{c}(x, \boldsymbol{T}(x)) \boldsymbol{\mu}(dx)$$

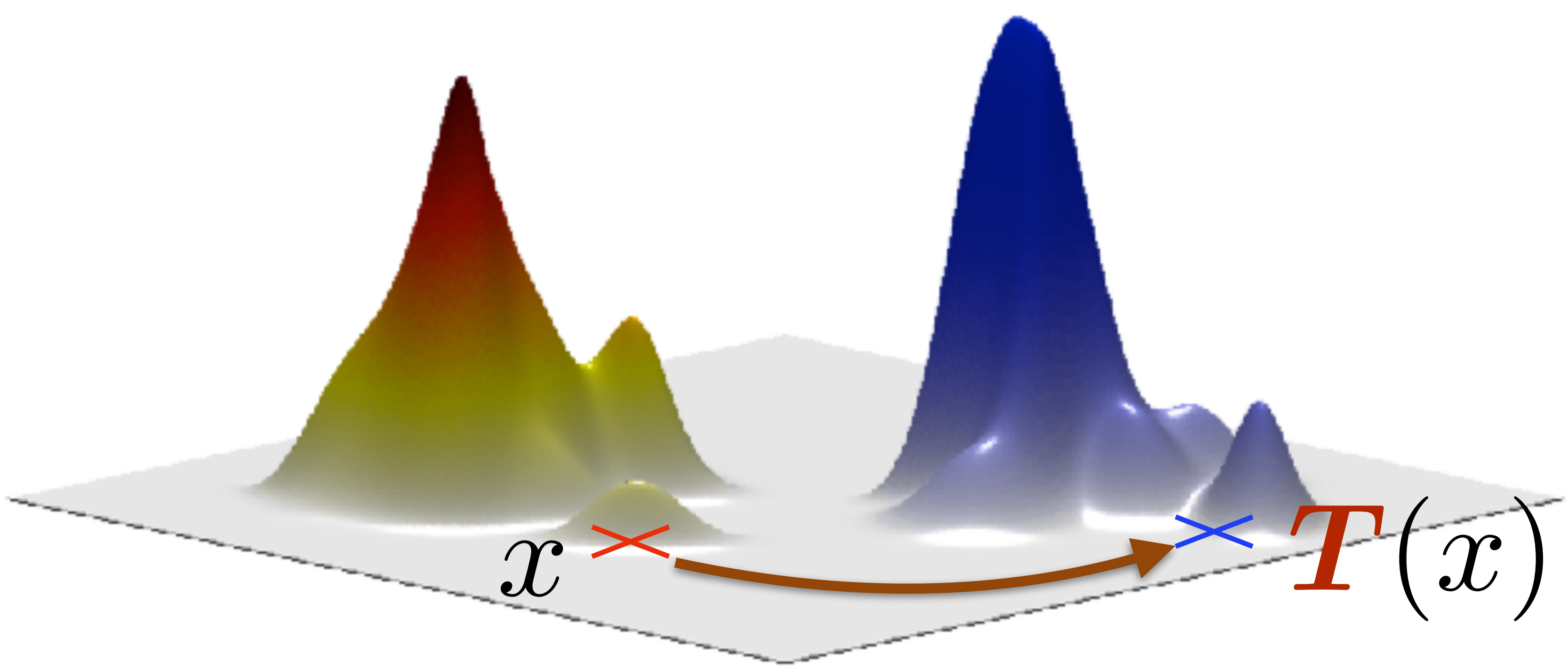


Monge Problem

Ω a probability space, $\boldsymbol{c} : \Omega \times \Omega \to \mathbb{R}$.
$\boldsymbol{\mu}, \boldsymbol{\nu}$ two probability measures in $\mathcal{P}(\Omega)$.

[Monge'81] problem: find a map $\boldsymbol{T} : \Omega \to \Omega$

[Brenier'87] If $\Omega = \mathbb{R}^d$, $\boldsymbol{c} = \|\cdot - \cdot\|^2$,
$\boldsymbol{\mu}, \boldsymbol{\nu}$ a.c., then $\boldsymbol{T} = \nabla \boldsymbol{u}$, $\boldsymbol{u}$ **convex**.

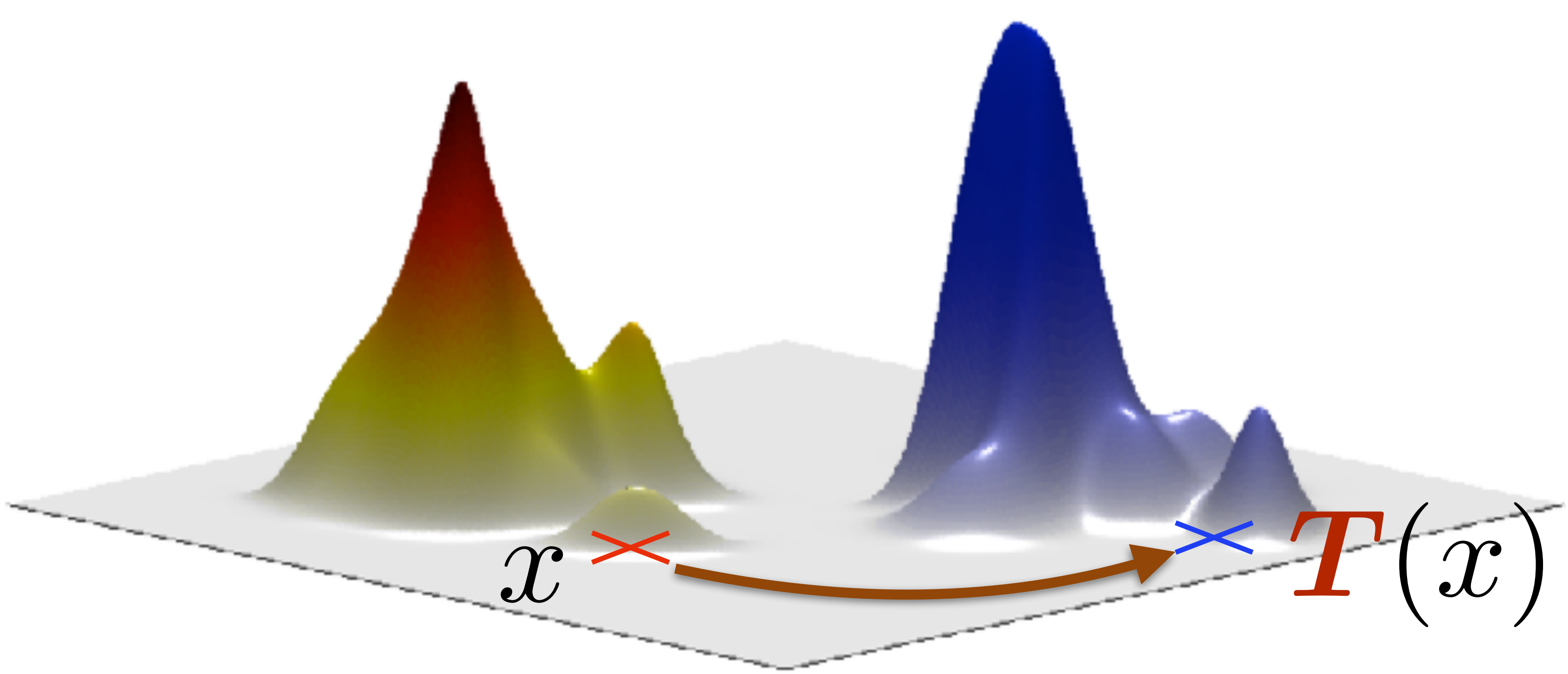


Monge Problem

Ω a probability space, $\boldsymbol{c} : \Omega \times \Omega \to \mathbb{R}$.
$\boldsymbol{\mu}, \boldsymbol{\nu}$ two probability measures in $\mathcal{P}(\Omega)$.

[Monge'81] problem: find a map $\boldsymbol{T} : \Omega \to \Omega$

$$\inf_{\boldsymbol{T}_\sharp \boldsymbol{\mu} = \boldsymbol{\nu}} \int_\Omega \boldsymbol{c}(x, \boldsymbol{T}(x)) \boldsymbol{\mu}(dx)$$

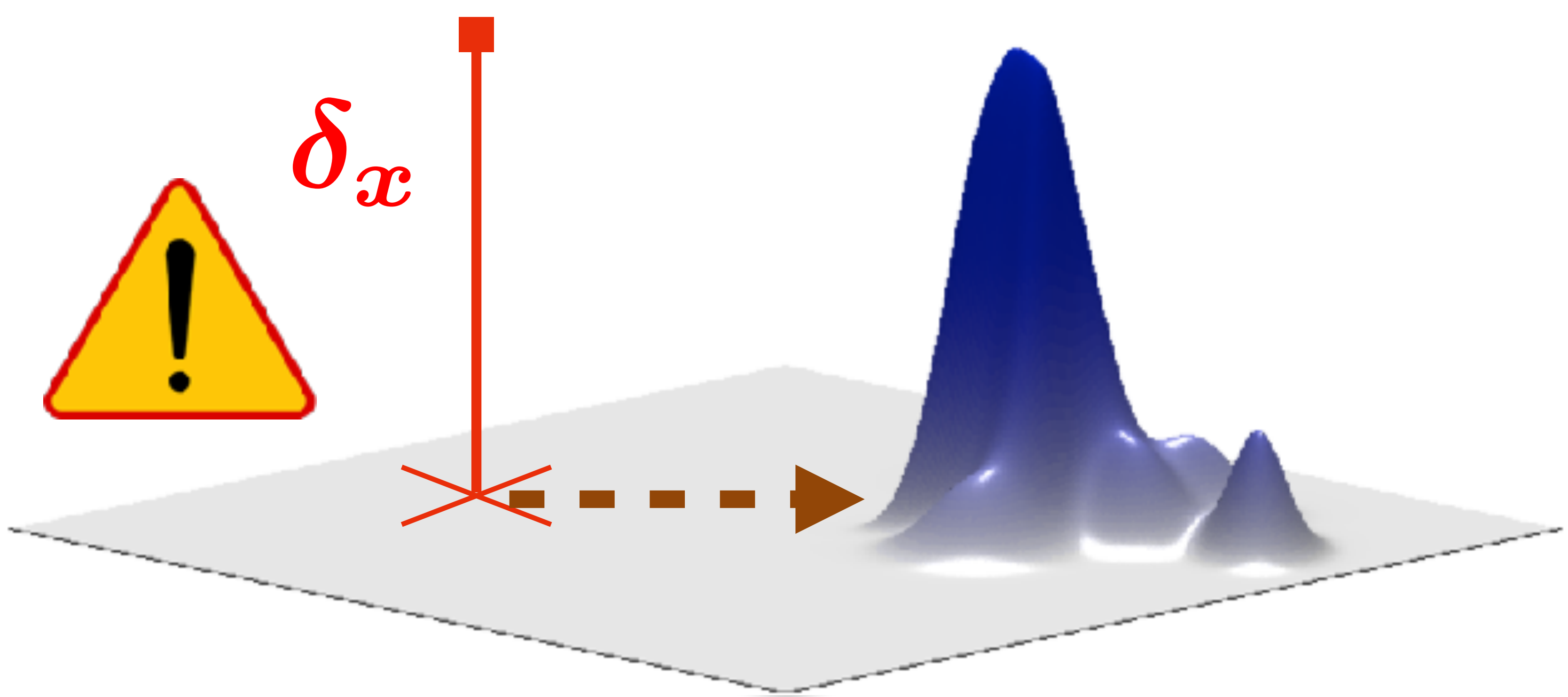


Monge Problem

Ω a probability space, $\boldsymbol{c} : \Omega \times \Omega \to \mathbb{R}$.
$\boldsymbol{\mu}, \boldsymbol{\nu}$ two probability measures in $\mathcal{P}(\Omega)$.

[Monge'81] problem: find a map $\boldsymbol{T} : \Omega \to \Omega$

$$\inf_{\boldsymbol{T}_\sharp \boldsymbol{\mu} = \boldsymbol{\nu}} \int_\Omega \boldsymbol{c}(x, \boldsymbol{T}(x)) \boldsymbol{\mu}(dx)$$



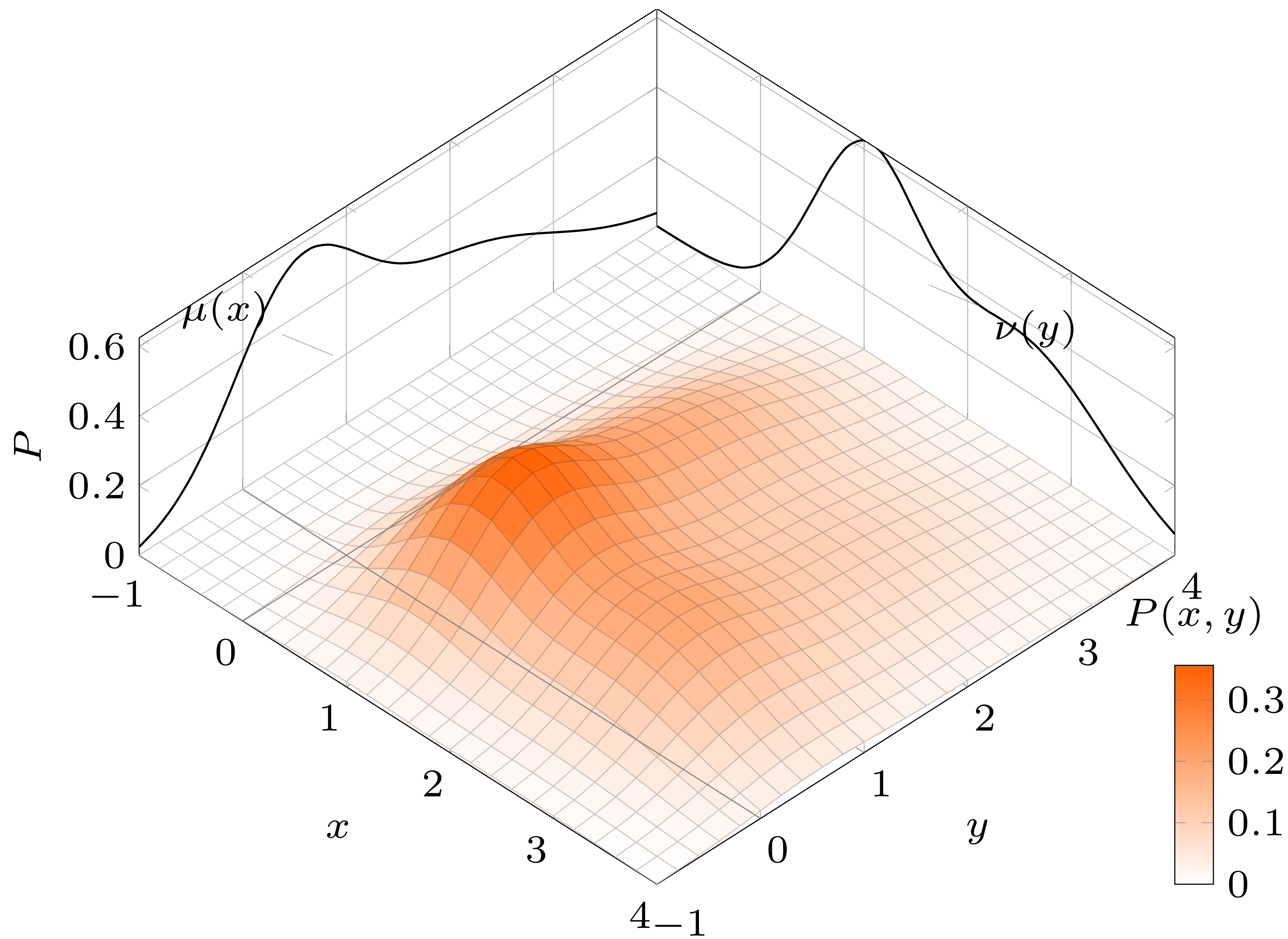
[Kantorovich'42] Relaxation

- Instead of maps $T : \Omega \to \Omega$, consider probabilistic maps, i.e. **couplings** $P \in \mathcal{P}(\Omega \times \Omega)$:

$$\Pi(\mu, \nu) \stackrel{\text{def}}{=} \{P \in \mathcal{P}(\Omega \times \Omega) \,|\, \forall A, B \subset \Omega, \\ P(A \times \Omega) = \mu(A), \\ P(\Omega \times B) = \nu(B)\}$$

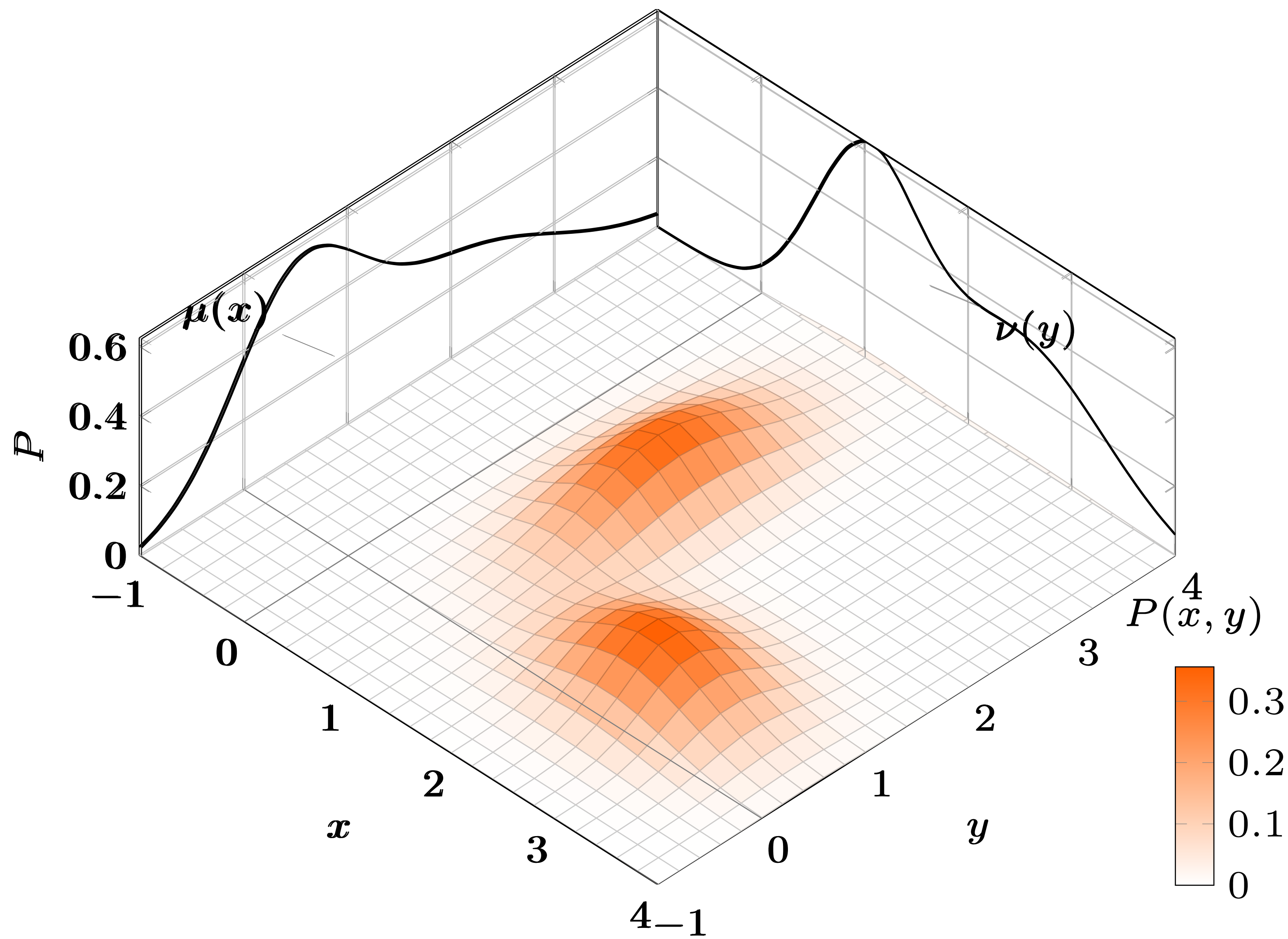
[Kantorovich'42] Relaxation

$$\Pi(\boldsymbol{\mu},\boldsymbol{\nu}) \stackrel{\text{def}}{=} \{\boldsymbol{P} \in \mathcal{P}(\Omega\times\Omega) \,|\, \forall \boldsymbol{A},\boldsymbol{B} \subset \Omega,\ \boldsymbol{P}(\boldsymbol{A}\times\Omega) = \boldsymbol{\mu}(\boldsymbol{A}), \boldsymbol{P}(\Omega\times\boldsymbol{B}) = \boldsymbol{\nu}(\boldsymbol{B})\}$$



[Kantorovich'42] Relaxation

$$\Pi(\mu, \nu) \overset{\text{def}}{=} \{P \in \mathcal{P}(\Omega \times \Omega) | \forall A, B \subset \Omega, P(A \times \Omega) = \mu(A), P(\Omega \times B) = \nu(B)\}$$



Kantorovich Problem

Def. Given $\boldsymbol{\mu}, \boldsymbol{\nu}$ in $\mathcal{P}(\Omega)$; a cost function $\boldsymbol{c}$ on $\Omega \times \Omega$, the Kantorovich problem is

$$\inf_{\boldsymbol{P} \in \Pi(\boldsymbol{\mu}, \boldsymbol{\nu})} \iint \boldsymbol{c}(x, y) \boldsymbol{P}(dx, dy).$$

PRIMAL

Kantorovich Problem

Def. Given μ, ν in $\mathcal{P}(\Omega)$; a cost function c on $\Omega \times \Omega$, the Kantorovich problem is

$$\inf_{P \in \Pi(\mu,\nu)} \iint c(x,y) P(dx, dy).$$

PRIMAL

$$\sup_{\substack{\varphi \in L_1(\mu), \psi \in L_1(\nu) \\ \varphi(x)+\psi(y) \leq c(x,y)}} \int \varphi \, d\mu + \int \psi \, d\nu.$$

DUAL

(Kantorovich) Wasserstein Distances

Let $p \geq 1$. Let $\boldsymbol{c} := \boldsymbol{D}$, a metric.

Def. The p-**Wasserstein distance** between $\boldsymbol{\mu}, \boldsymbol{\nu}$ in $\mathcal{P}(\Omega)$ is

$$W_p(\boldsymbol{\mu}, \boldsymbol{\nu}) \stackrel{\text{def}}{=} \left(\inf_{\boldsymbol{P} \in \Pi(\boldsymbol{\mu}, \boldsymbol{\nu})} \iint \boldsymbol{D}(\boldsymbol{x}, \boldsymbol{y})^p \boldsymbol{P}(dx, dy) \right)^{1/p}.$$

(Kantorovich) Wasserstein Distances

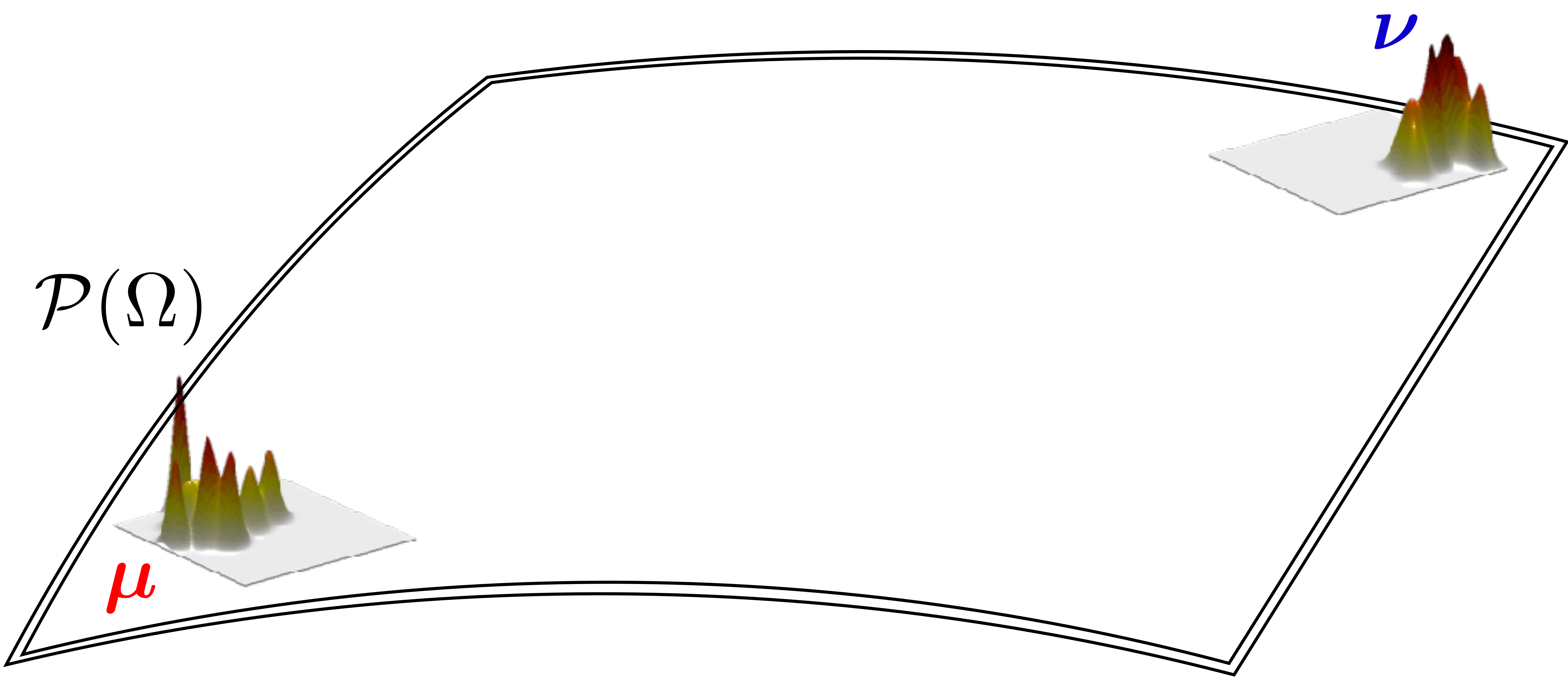
Let $p \geq 1$. Let $c := D$, a metric.

Def. The p-**Wasserstein distance** between μ, ν in $\mathcal{P}(\Omega)$ is

$$W_p^p(\mu, \nu) \stackrel{\text{def}}{=} \left(\inf_{P \in \Pi(\mu,\nu)} \iint D(x, y)^p P(dx, dy) \right)^{\cancel{1/p}}.$$

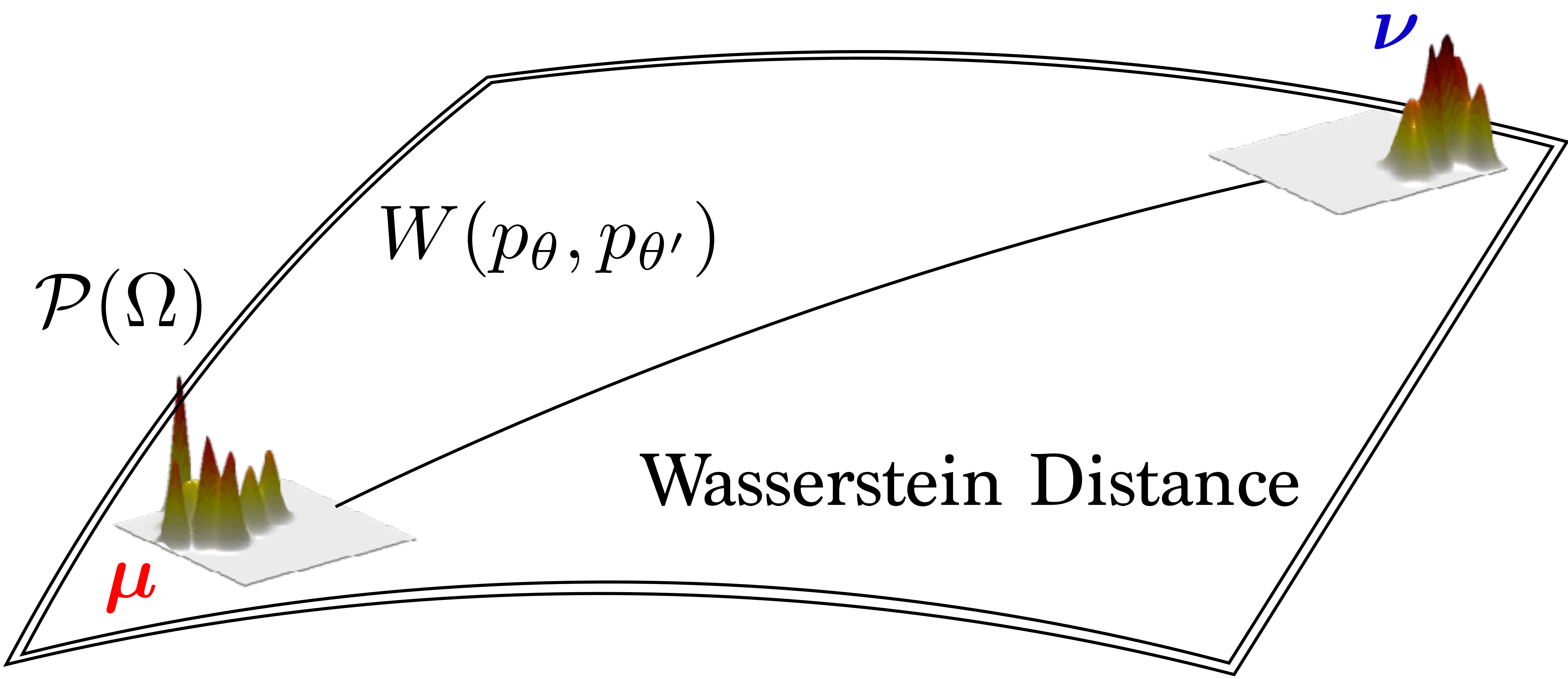
Optimal Transport Geometry

Very different geometry than standard information divergences (*KL*, Euclidean)



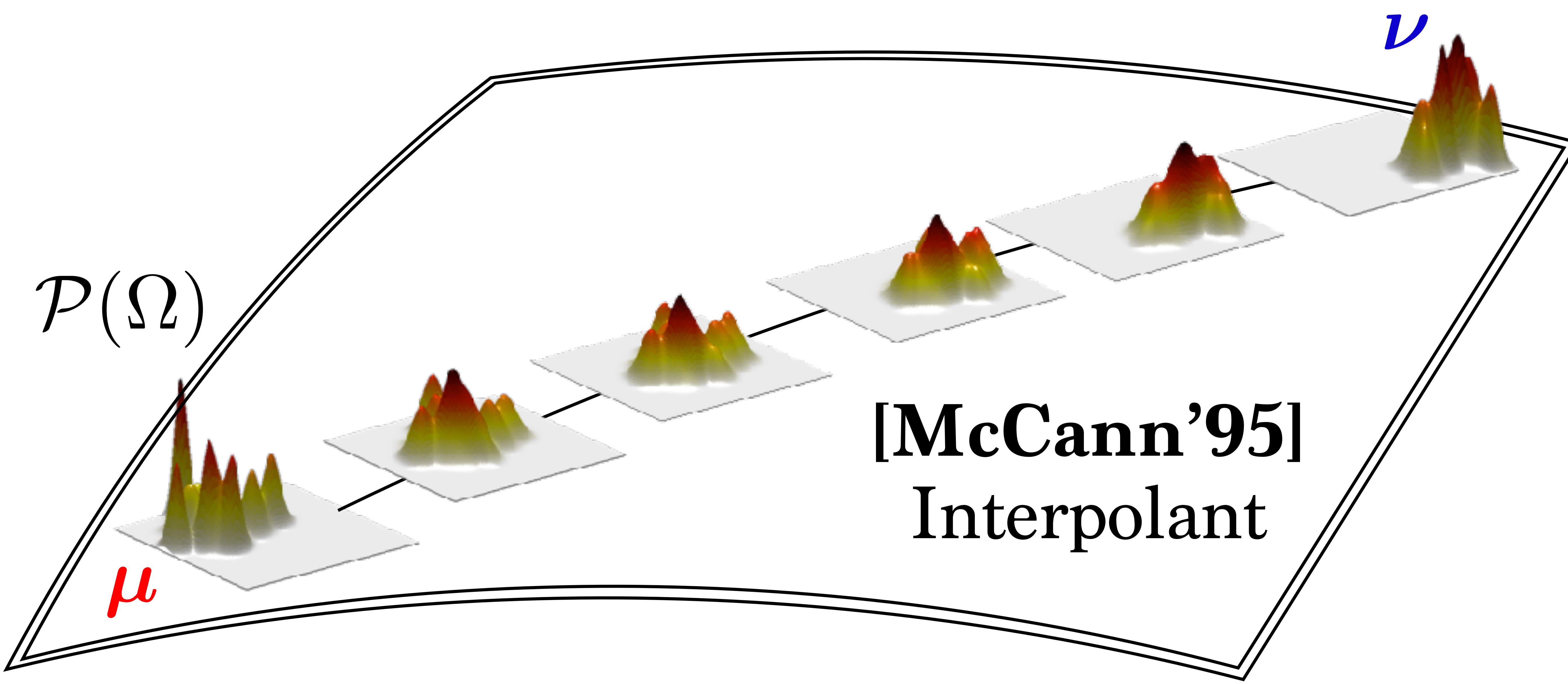
Optimal Transport Geometry

Very different geometry than standard information divergences (*KL*, Euclidean)



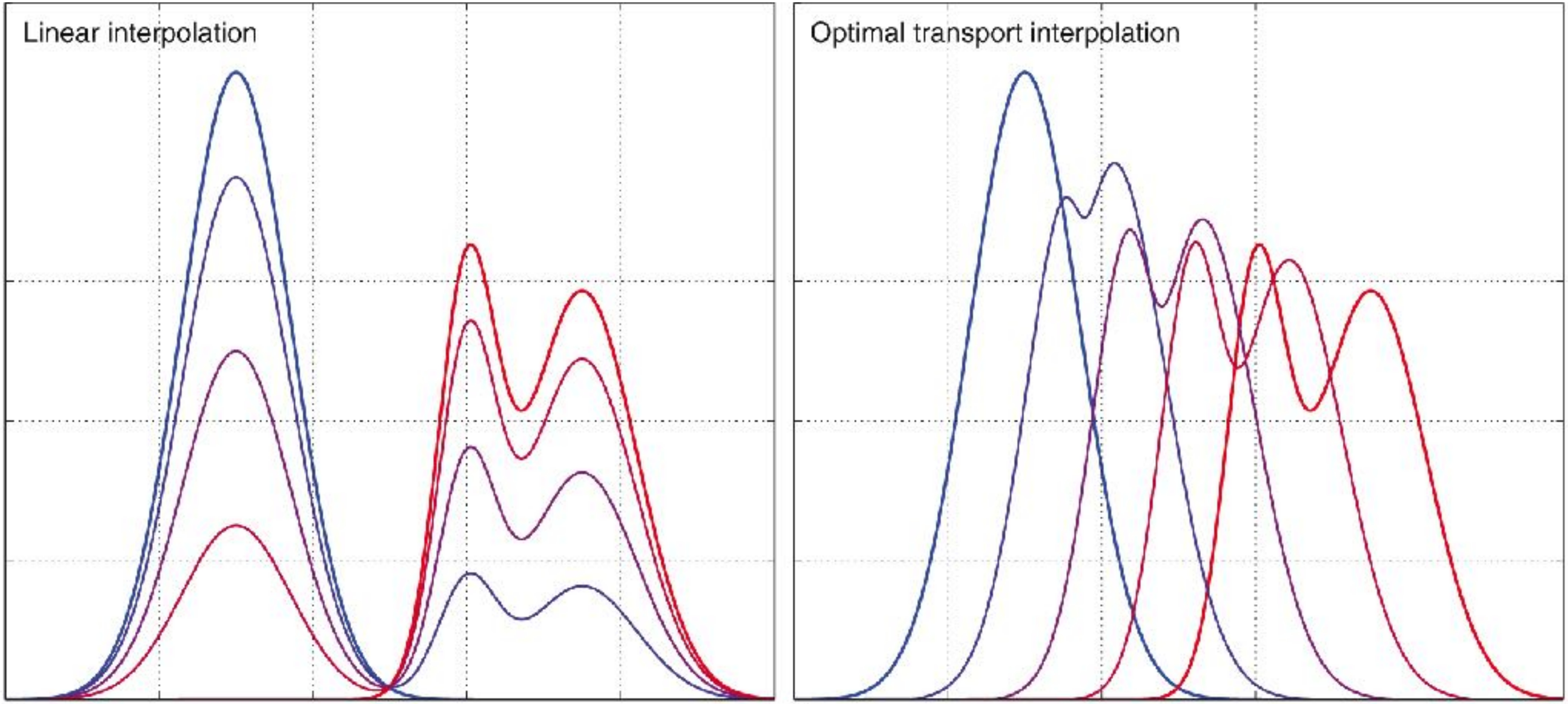
Optimal Transport Geometry

Very different geometry than standard information divergences (KL, Euclidean)



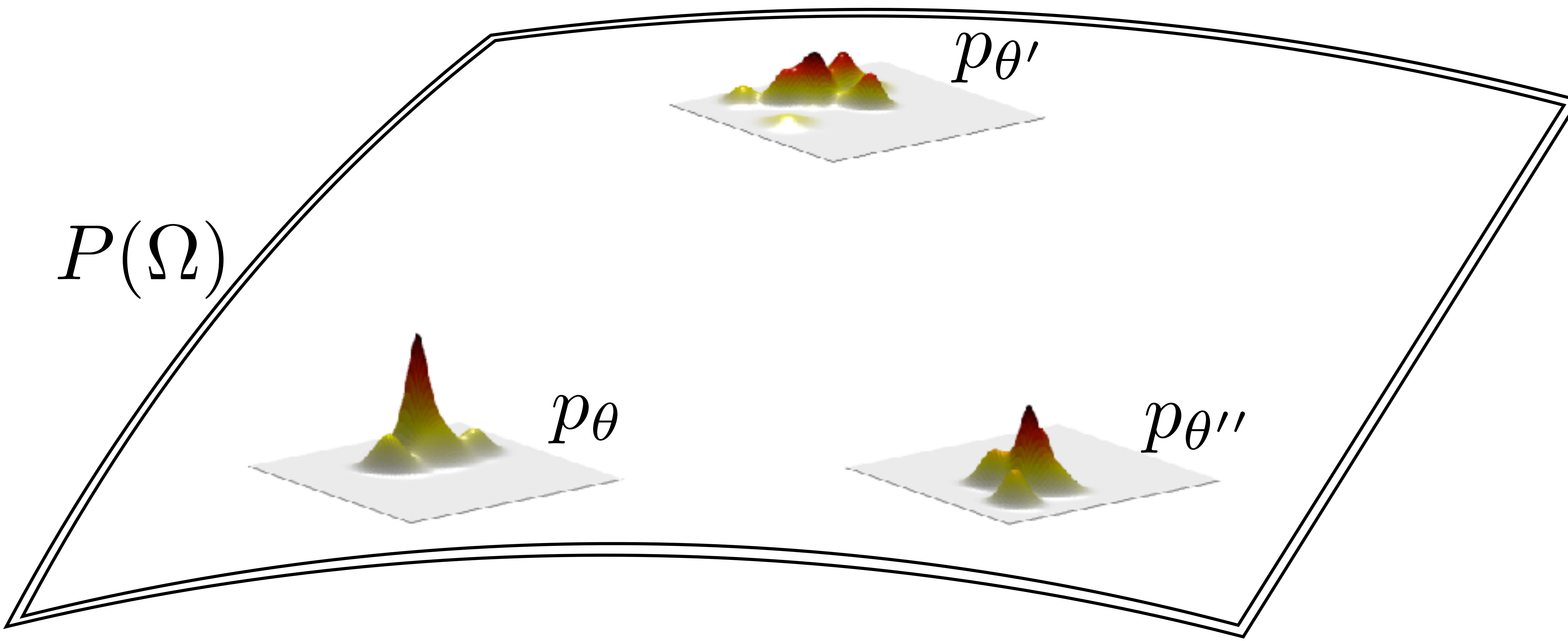
Optimal Transport Geometry

Very different geometry than standard information divergences (KL, Euclidean)



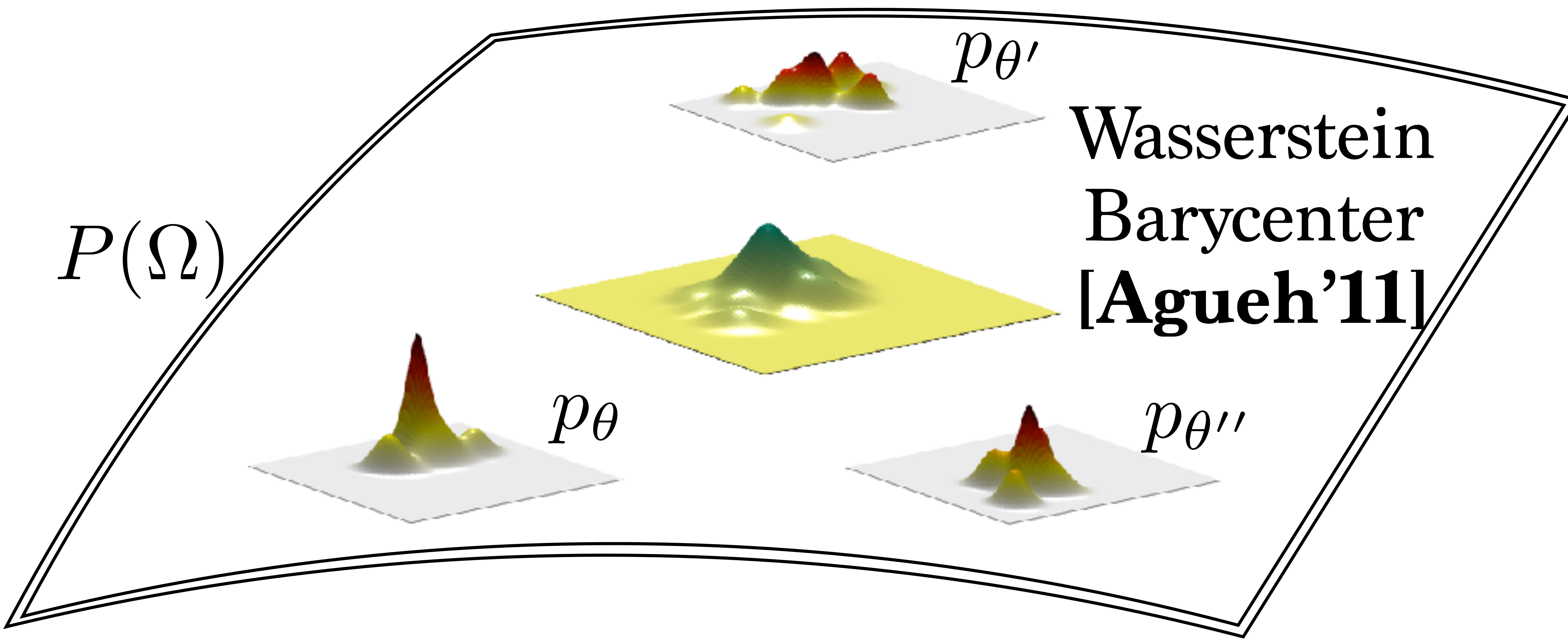
Optimal Transport Geometry

Very different geometry than standard information divergences (*KL*, Euclidean)

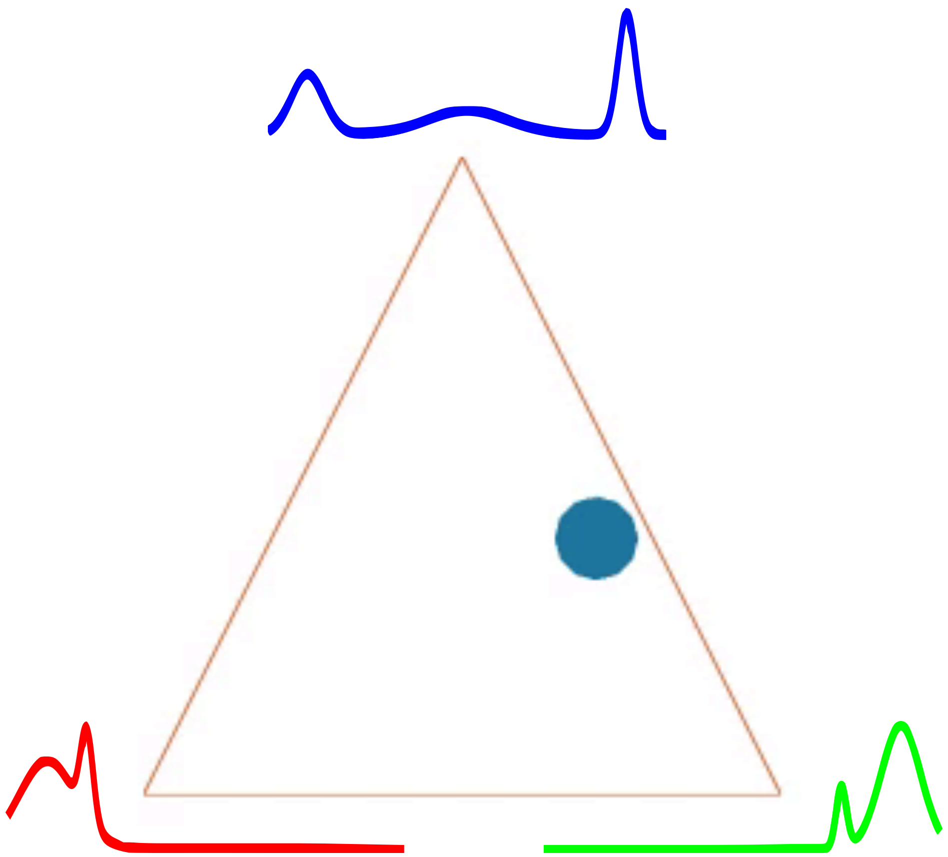


Optimal Transport Geometry

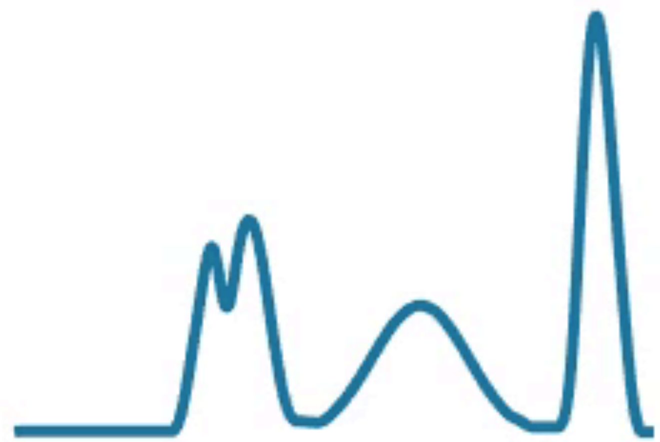
Very different geometry than standard information divergences (*KL*, Euclidean)



Optimal Transport Geometry



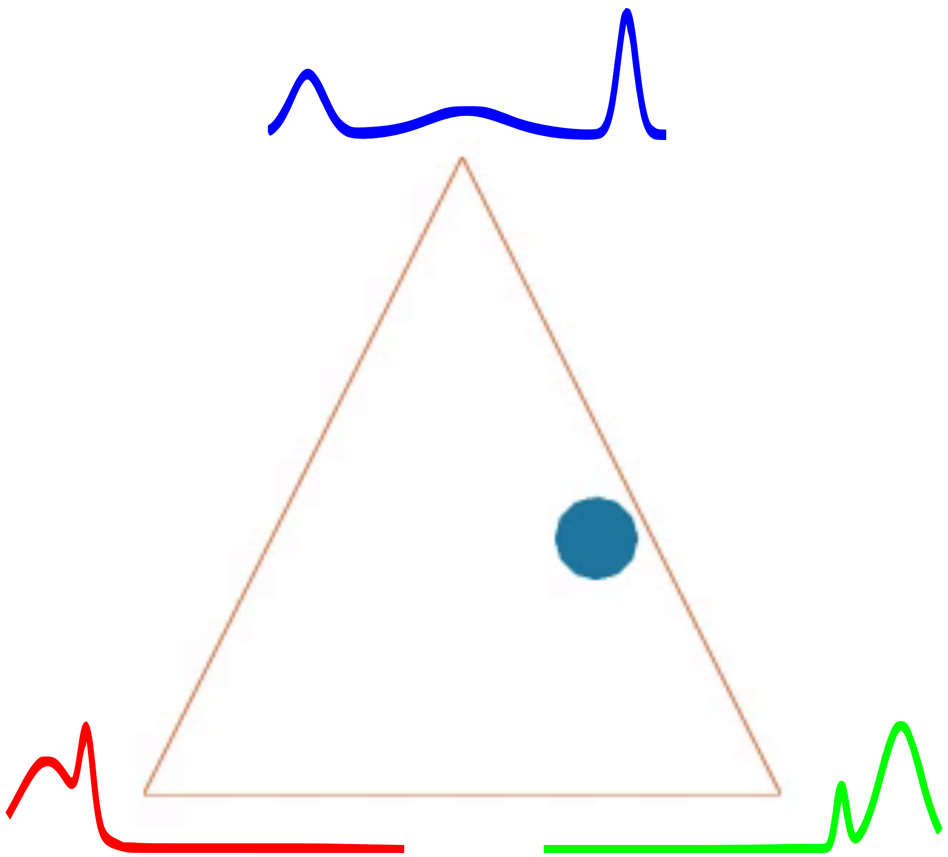
$\lambda \in \Sigma_3$



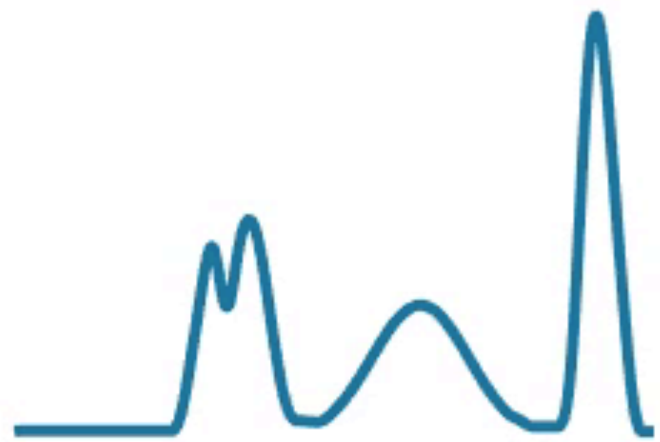
Wasserstein mean

L_2 mean

Optimal Transport Geometry



$\lambda \in \Sigma_3$



Wasserstein mean

L_2 mean

Computational OT

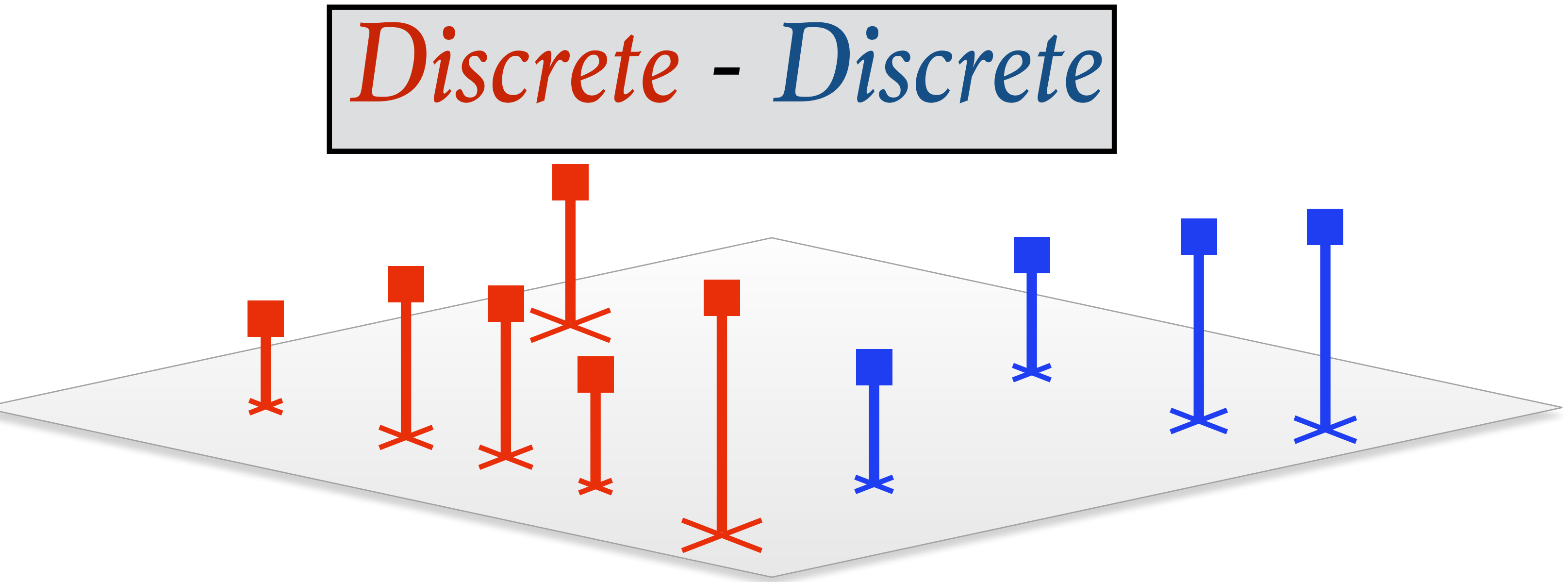
Up to 2010: OT solvers $W_p(\boldsymbol{\mu}, \boldsymbol{\nu}) = ?$

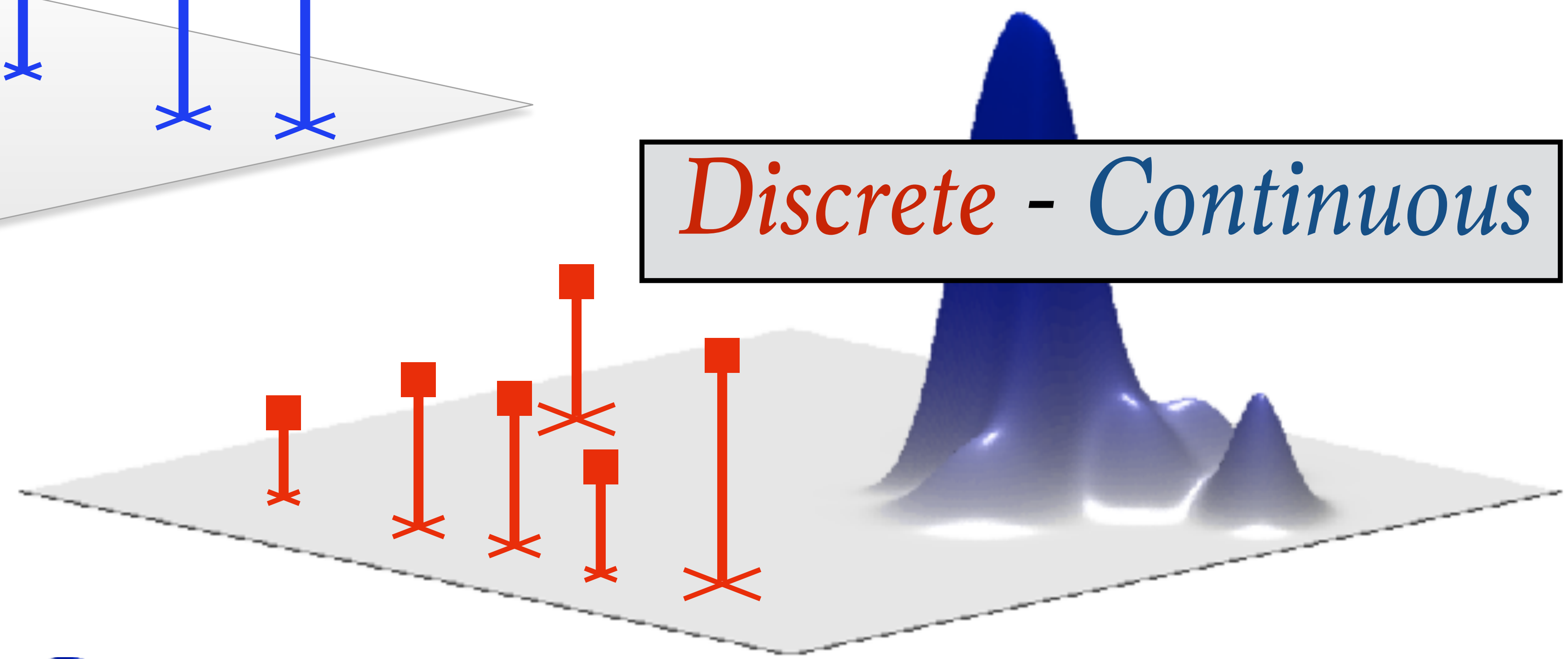
Goal now: use OT as a **loss or fidelity** term

$$\underset{\boldsymbol{\mu} \in \mathcal{P}(\Omega)}{\operatorname{argmin}} \; F(W_p(\boldsymbol{\mu}, \boldsymbol{\nu_1}), W_p(\boldsymbol{\mu}, \boldsymbol{\nu_2}), \ldots, \boldsymbol{\mu}) = ?$$

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

How can we compute OT?

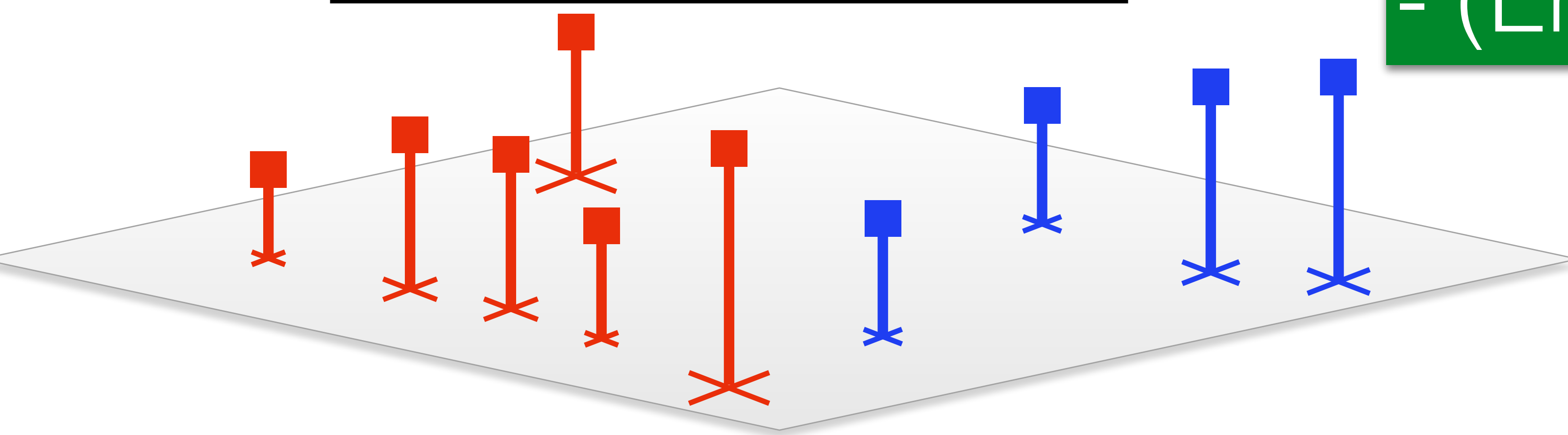




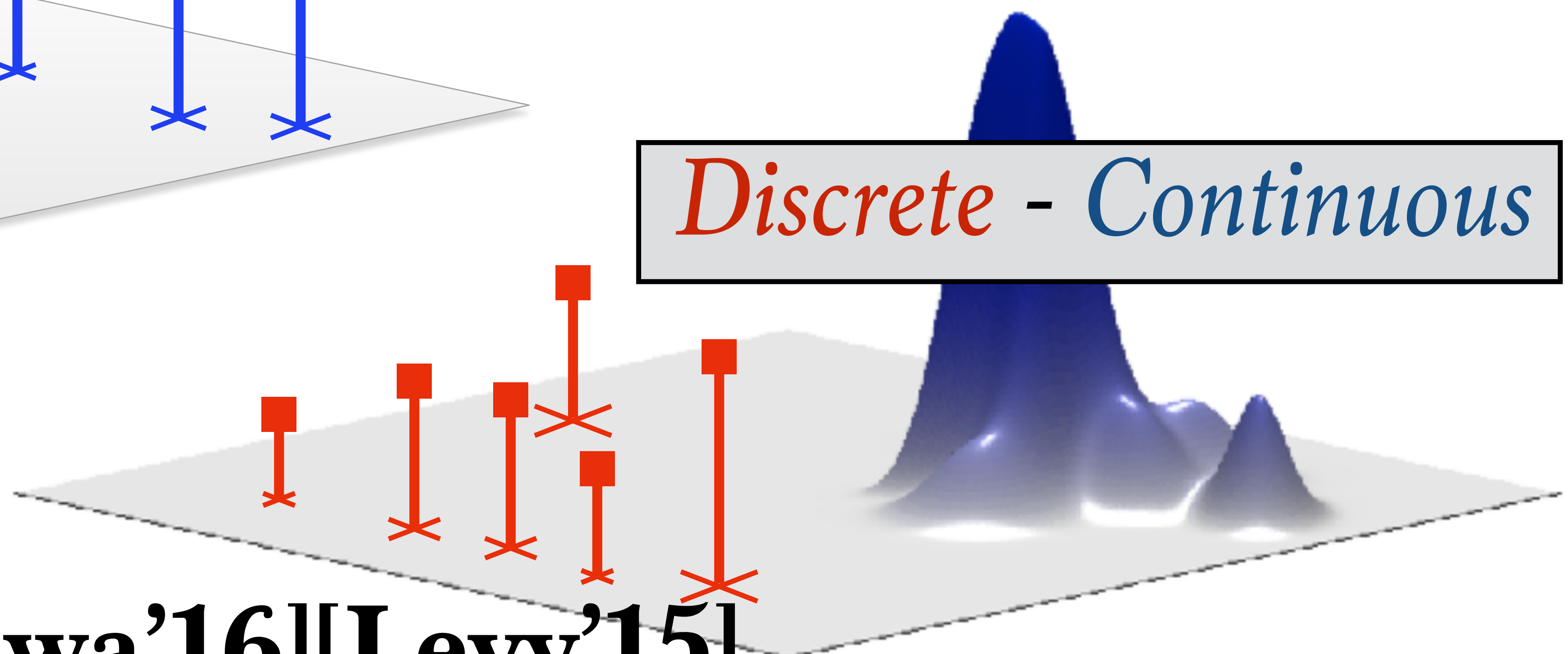
How can we compute OT?

Discrete - Discrete

- Network flow solvers
- (Entropic) regularization



Discrete - Continuous



low dim.

[Mérigot'11][Kitagawa'16][Levy'15]

Stochastic Optimization

[Genevay'16]

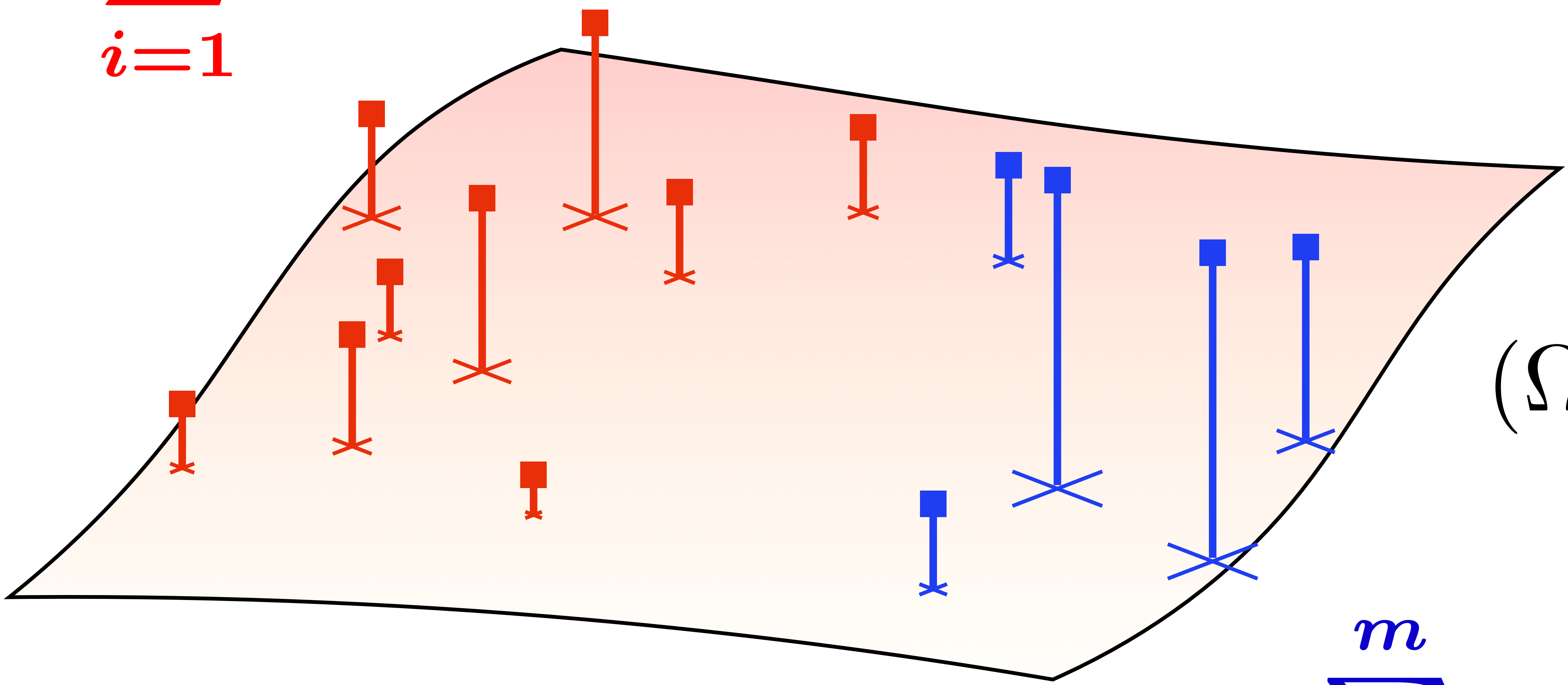
Continuous - Continuous

PDE's

[Benamou'98]

OT on Two Discrete Measures

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

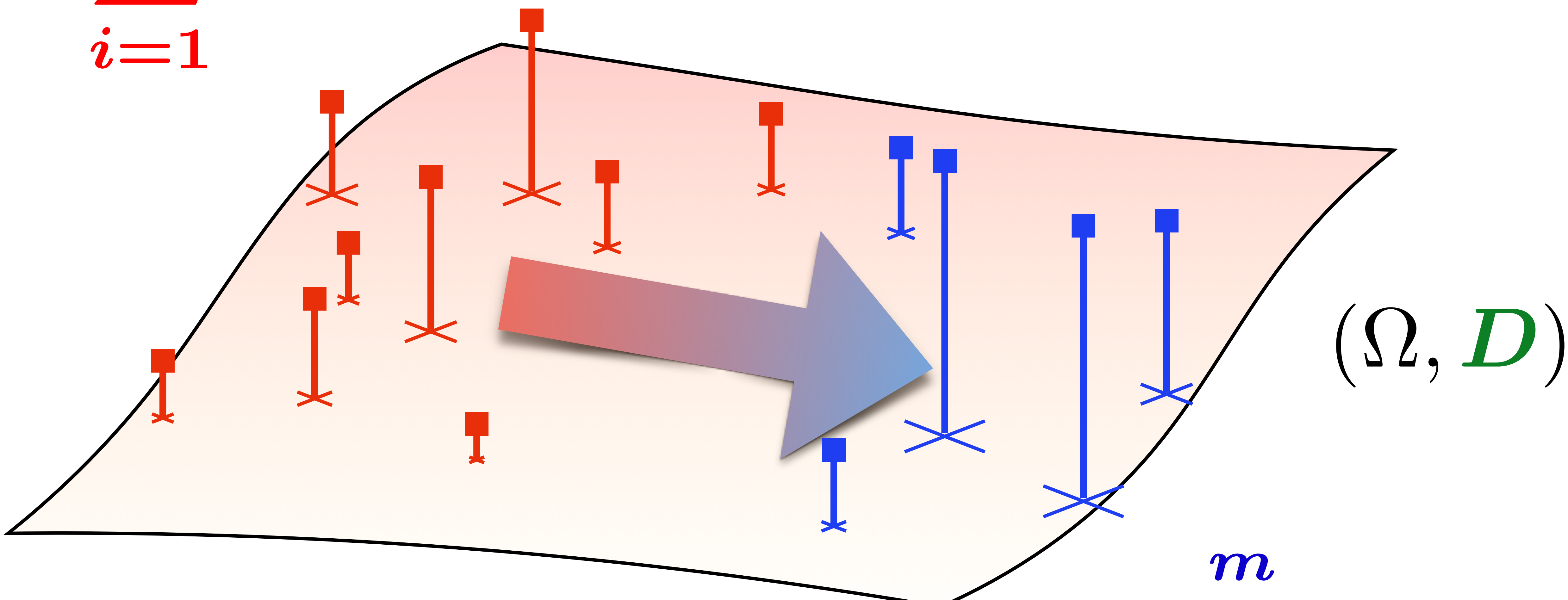


$$(\Omega, D)$$

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

OT on Two Discrete Measures

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



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

Wasserstein on Discrete Measures

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

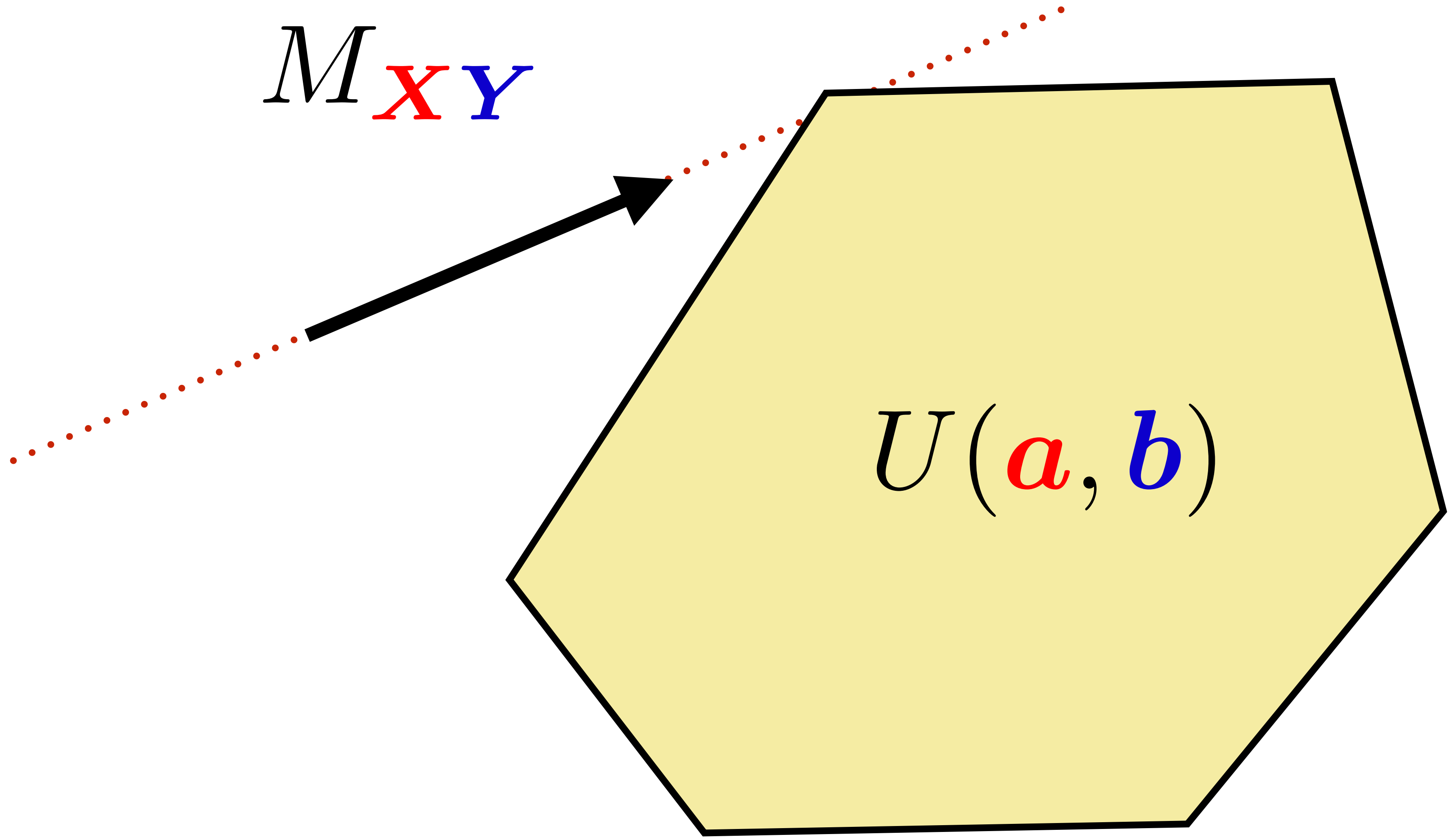
$$M_{\mathbf{XY}} \stackrel{\text{def}}{=} [D(\mathbf{x}_i, \mathbf{y}_j)^p]_{ij}$$

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

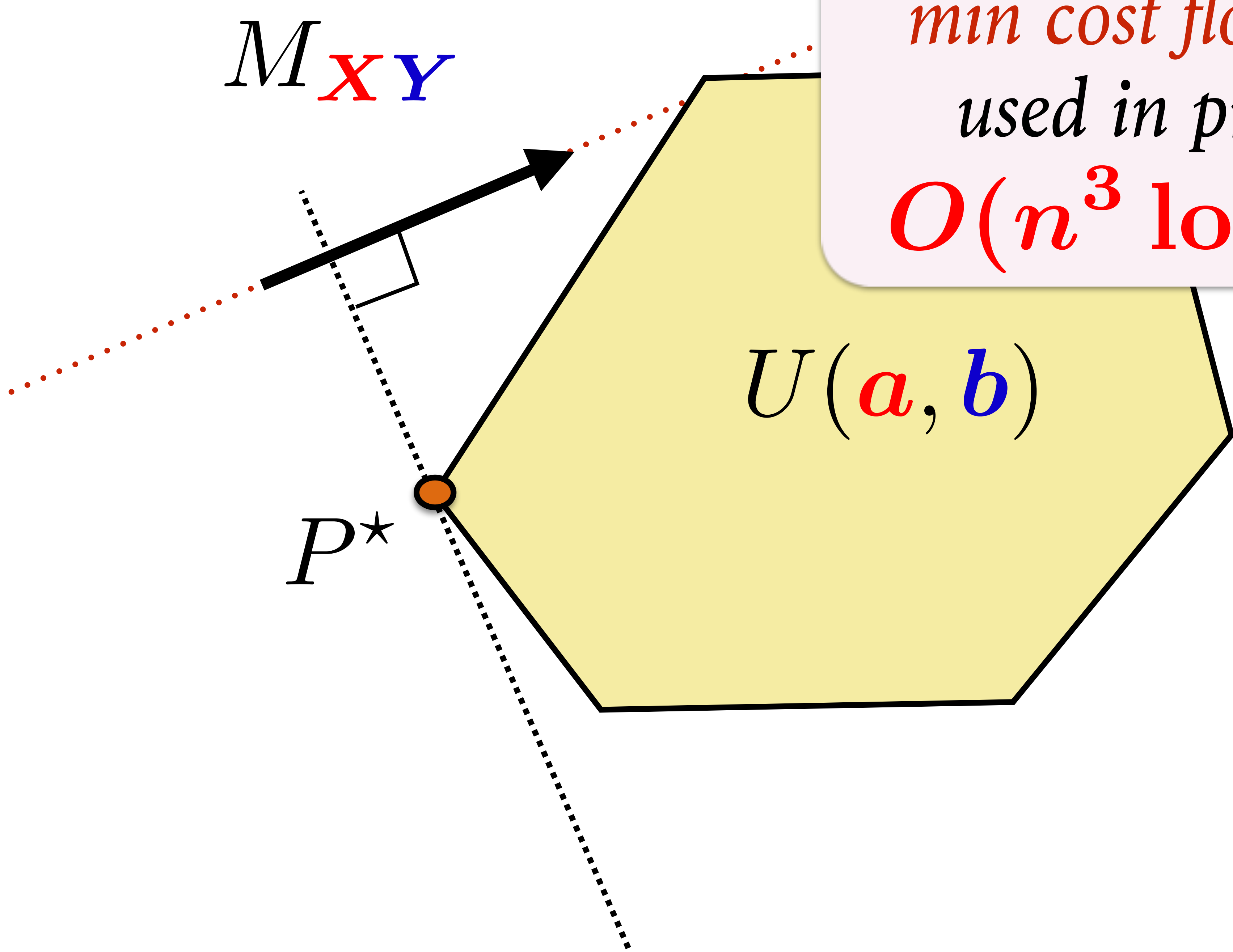
Def. Optimal Transport Problem

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

Solving the OT Problem

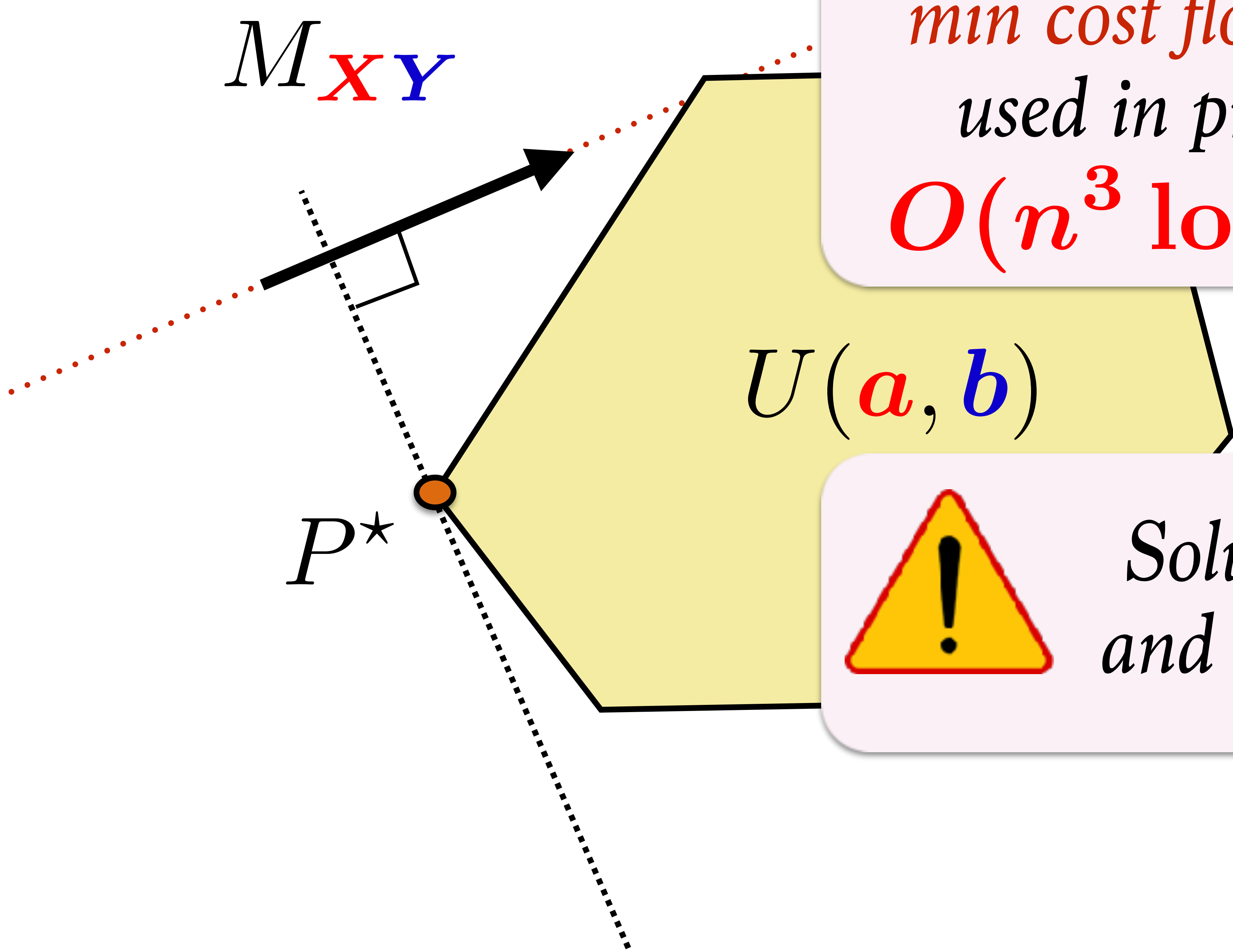


Solving the OT Problem



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

Solving the OT Problem

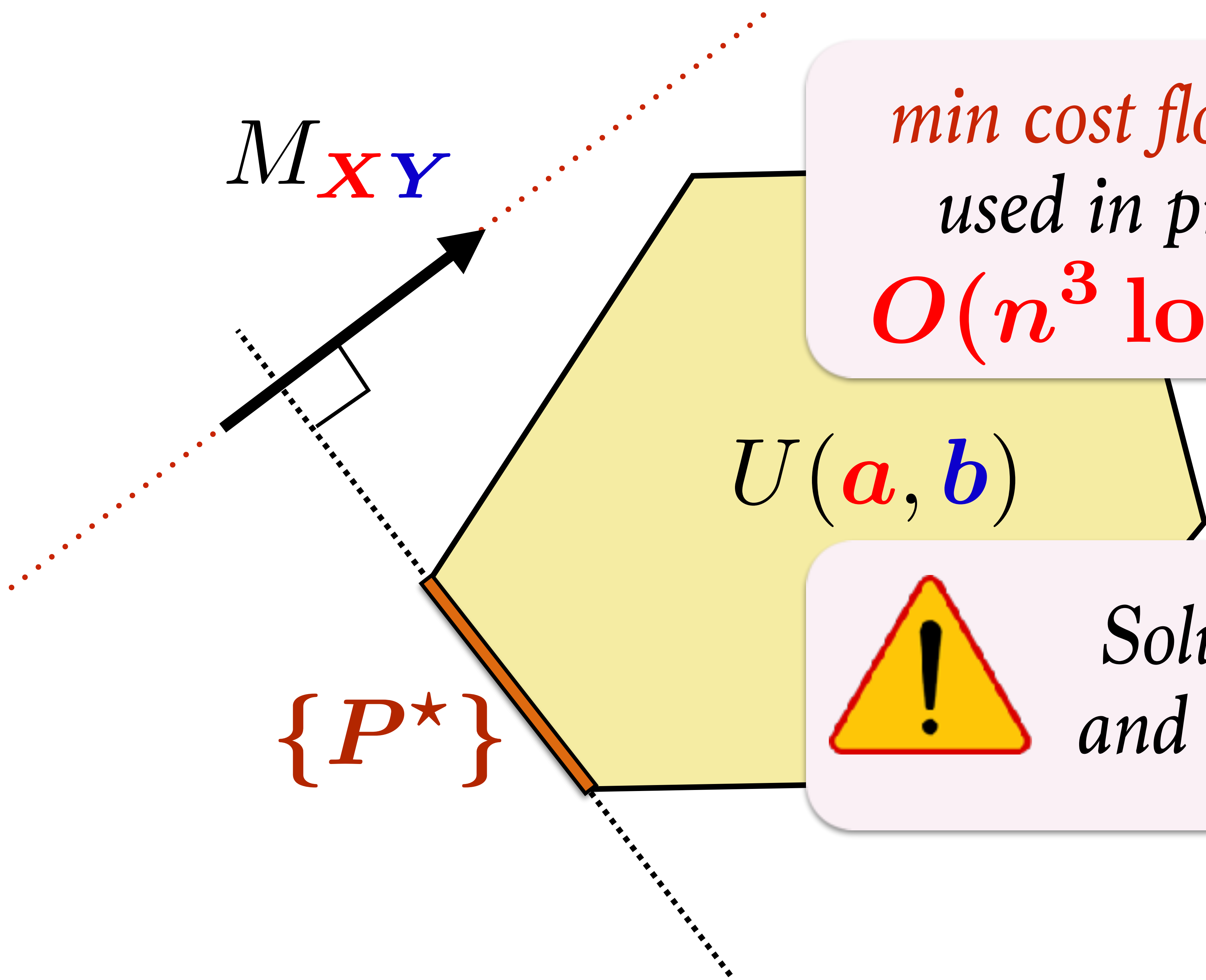


min cost flow solver *used in practice.*

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

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

Solving the OT Problem

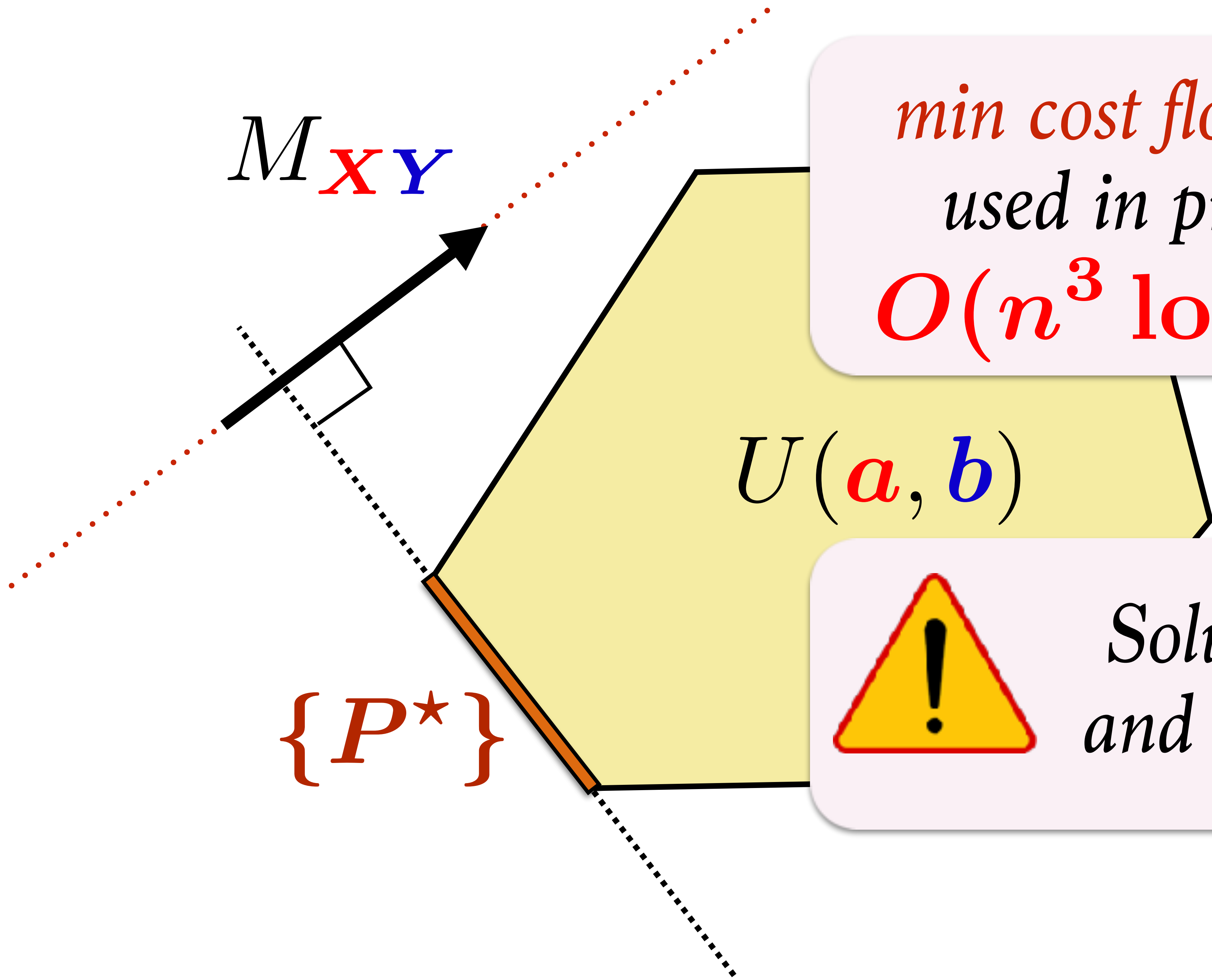


min cost flow solver used in practice.

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

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

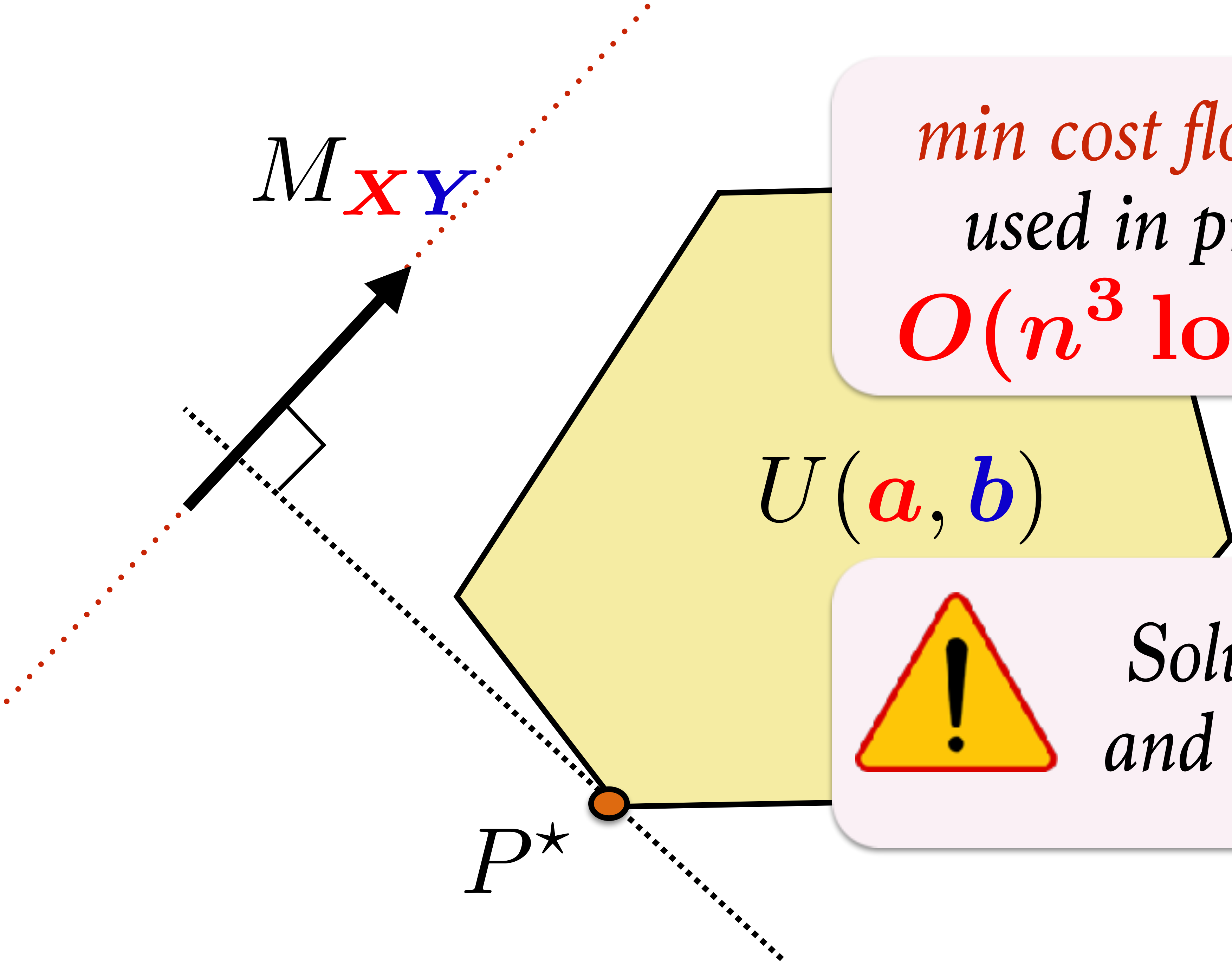
Solving the OT Problem



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

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

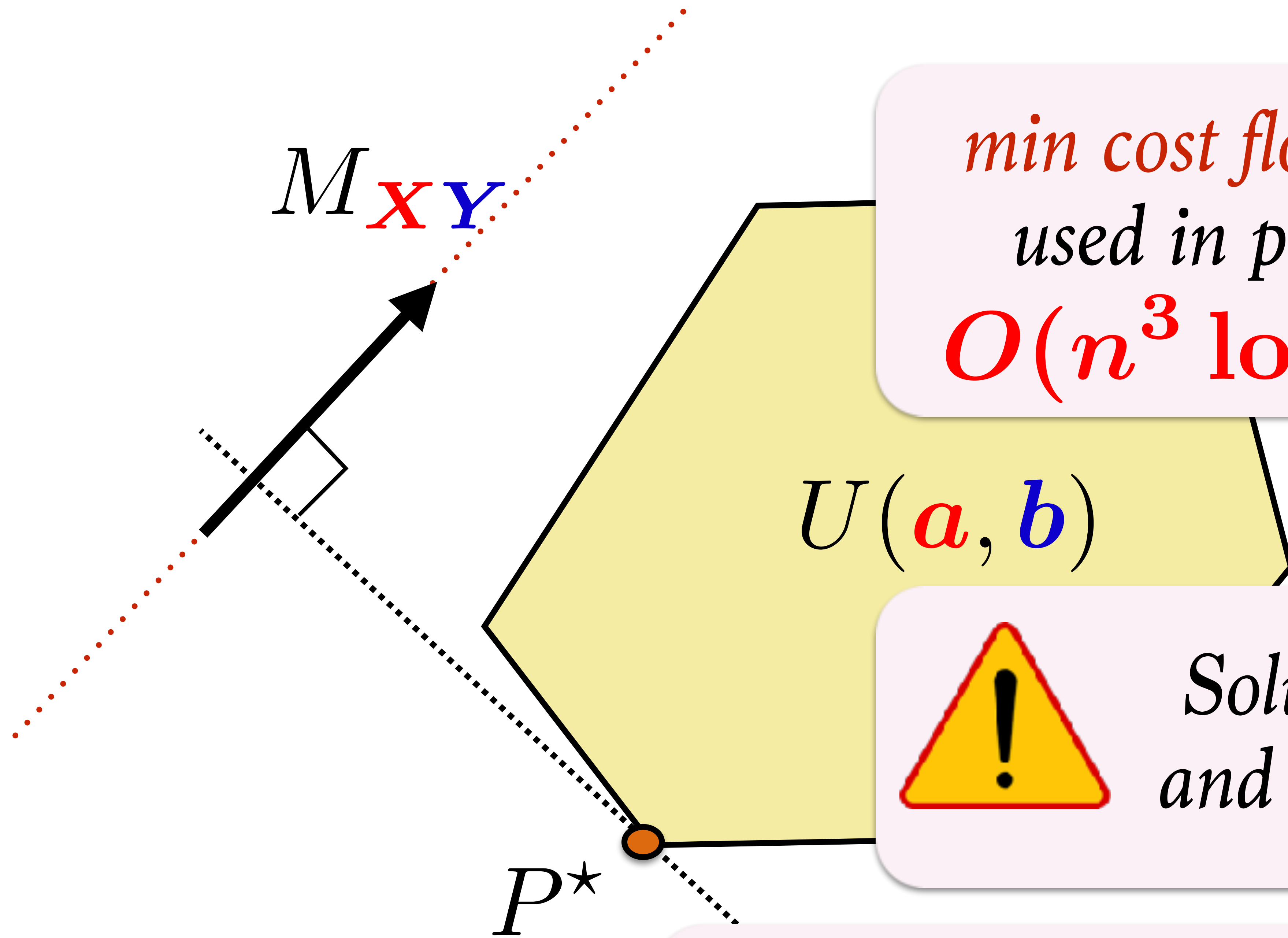
Solving the OT Problem



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

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

Solving the OT Problem

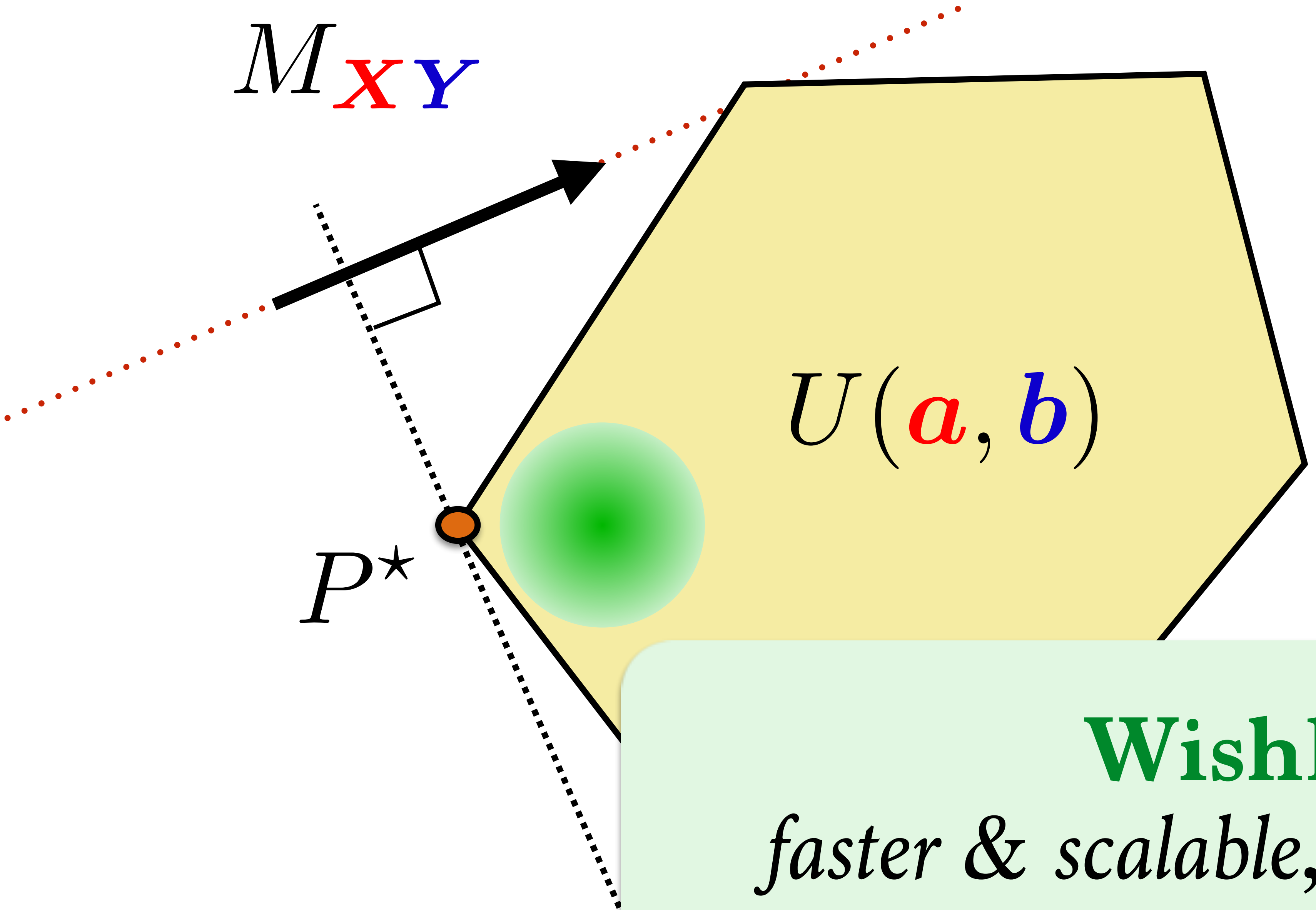


min cost flow solver *used in practice.* $\boldsymbol{O(n^3 \log(n))}$

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

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

Solution: Regularization



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

Entropic Regularization [Wilson'62]

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

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

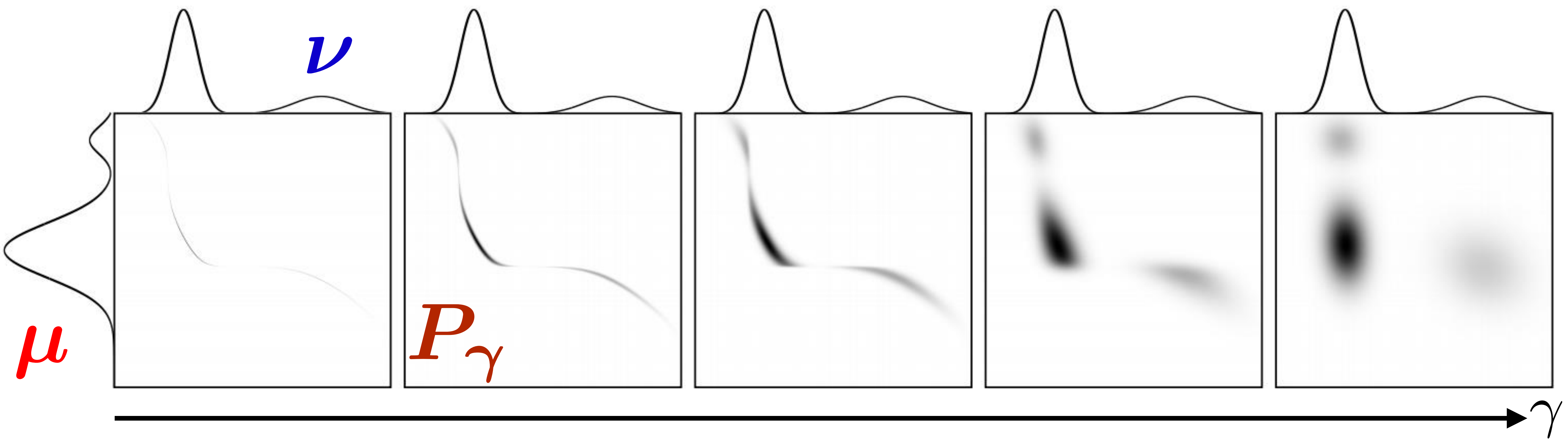
$$E(P) \stackrel{\text{def}}{=} - \sum_{i,j=1}^{nm} P_{ij} (\log P_{ij} - 1)$$

Note: Unique optimal solution because of strong concavity of entropy

Entropic Regularization [Wilson'62]

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

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

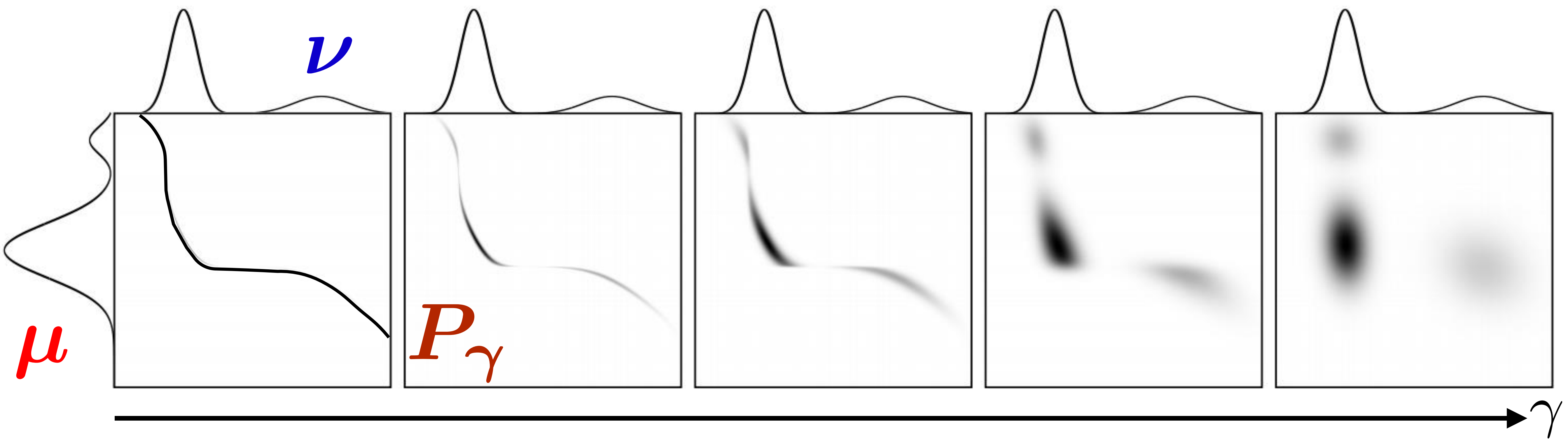


Note: Unique optimal solution because of strong concavity of entropy

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

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

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

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

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

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

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

Fast & Scalable Algorithm

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

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

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

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

Fast & Scalable Algorithm

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

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

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

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

Fast & Scalable Algorithm

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

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

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

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

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

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

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

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

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

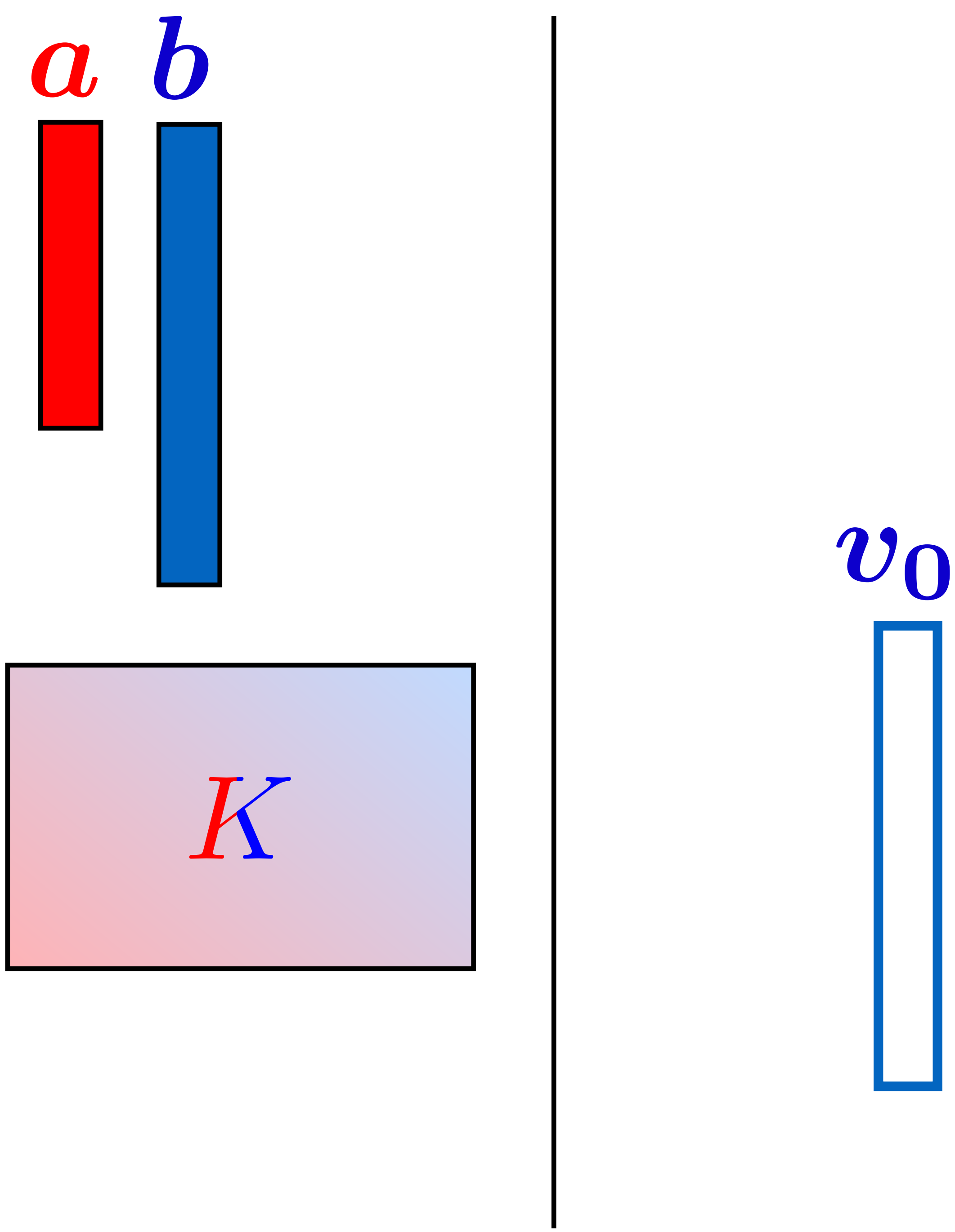
1. $\boldsymbol{u} = \boldsymbol{a}/K\boldsymbol{v}$
2. $\boldsymbol{v} = \boldsymbol{b}/K^T\boldsymbol{u}$

- [**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**]

Fast & Scalable Algorithm

- **[Sinkhorn'64]** fixed-point iterations for $(\boldsymbol{u}, \boldsymbol{v})$

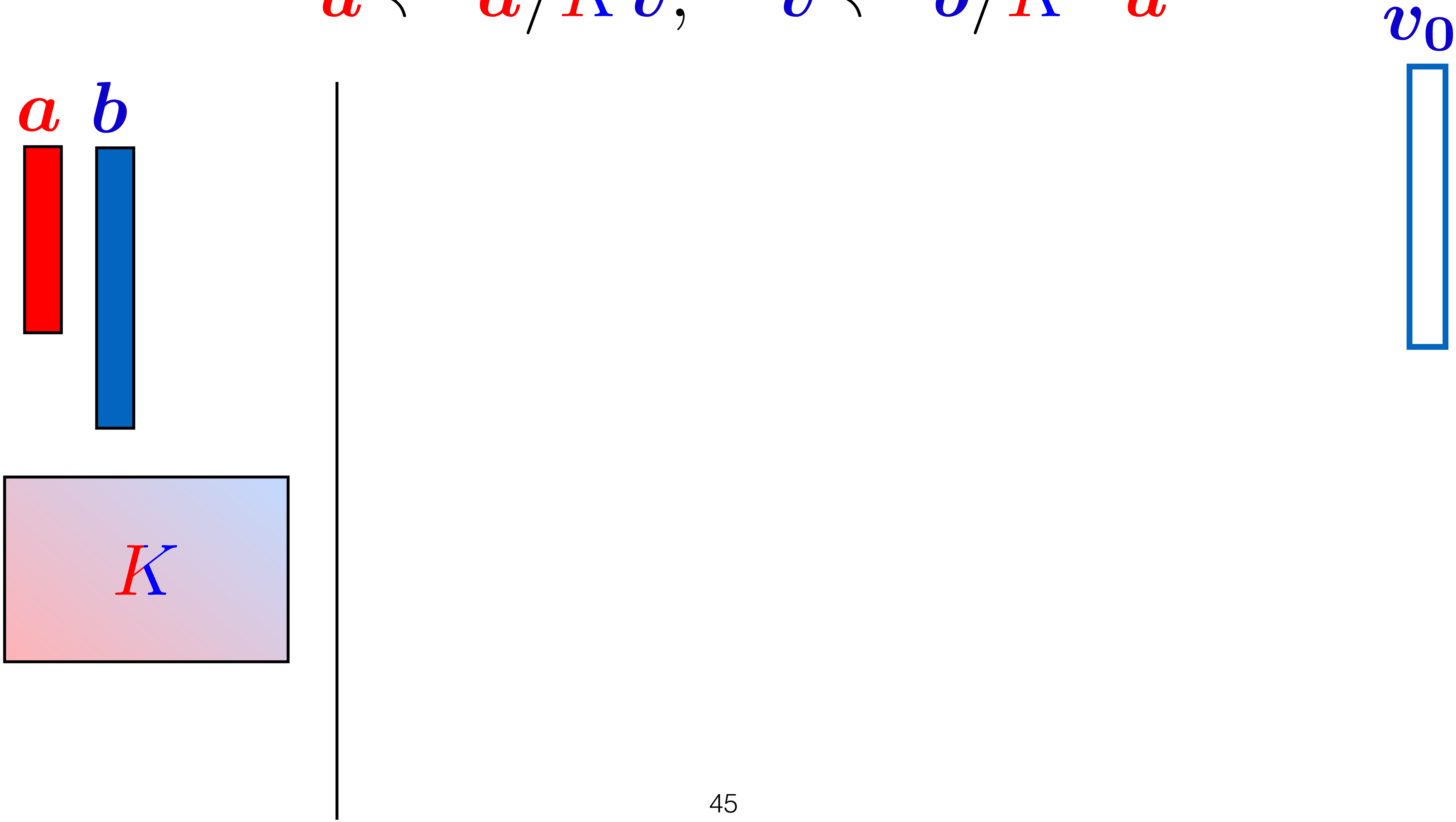
$$\boldsymbol{u} \leftarrow \boldsymbol{a}/K\boldsymbol{v}, \quad \boldsymbol{v} \leftarrow \boldsymbol{b}/K^T\boldsymbol{u}$$



Fast & Scalable Algorithm

- **[Sinkhorn'64]** fixed-point iterations for $(\boldsymbol{u}, \boldsymbol{v})$

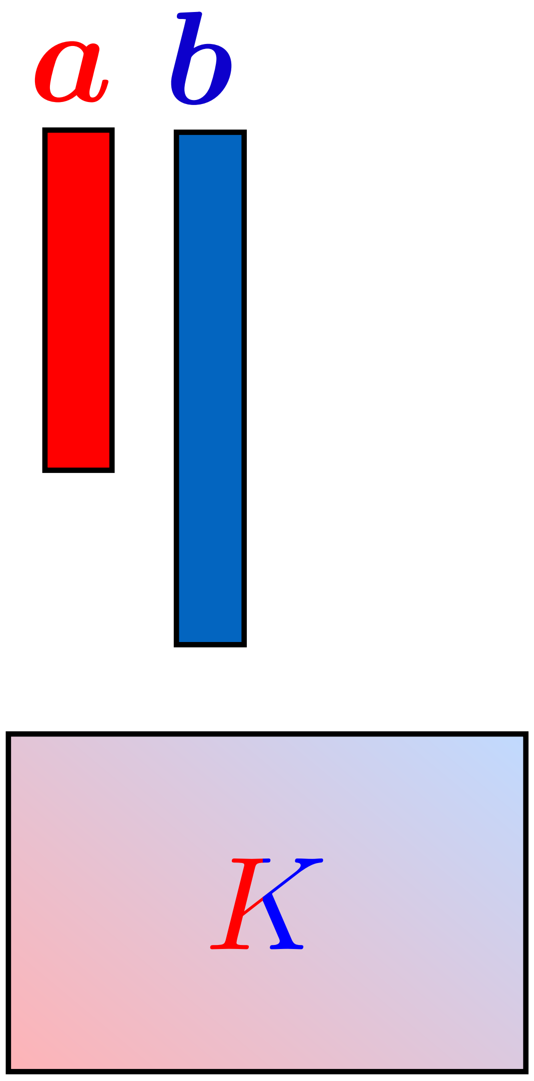
$$\boldsymbol{u} \leftarrow \boldsymbol{a}/K\boldsymbol{v}, \quad \boldsymbol{v} \leftarrow \boldsymbol{b}/K^T\boldsymbol{u}$$

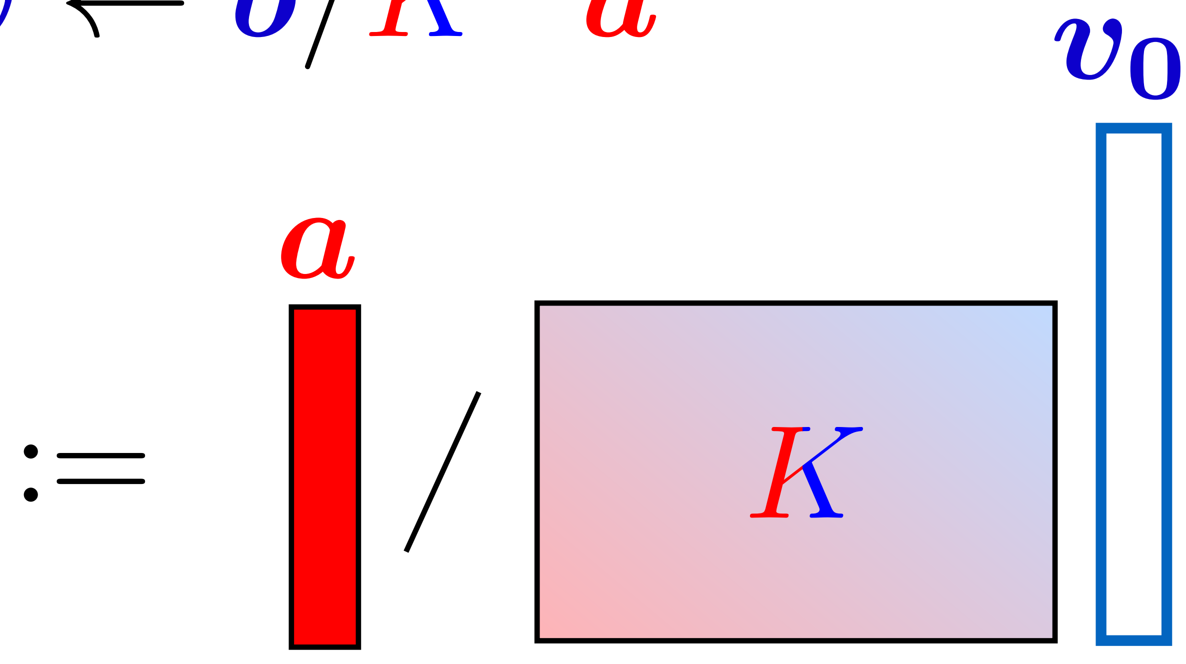


Fast & Scalable Algorithm

- **[Sinkhorn'64]** fixed-point iterations for $(\boldsymbol{u}, \boldsymbol{v})$

$$\boldsymbol{u} \leftarrow \boldsymbol{a}/K\boldsymbol{v}, \quad \boldsymbol{v} \leftarrow \boldsymbol{b}/K^T\boldsymbol{u}$$

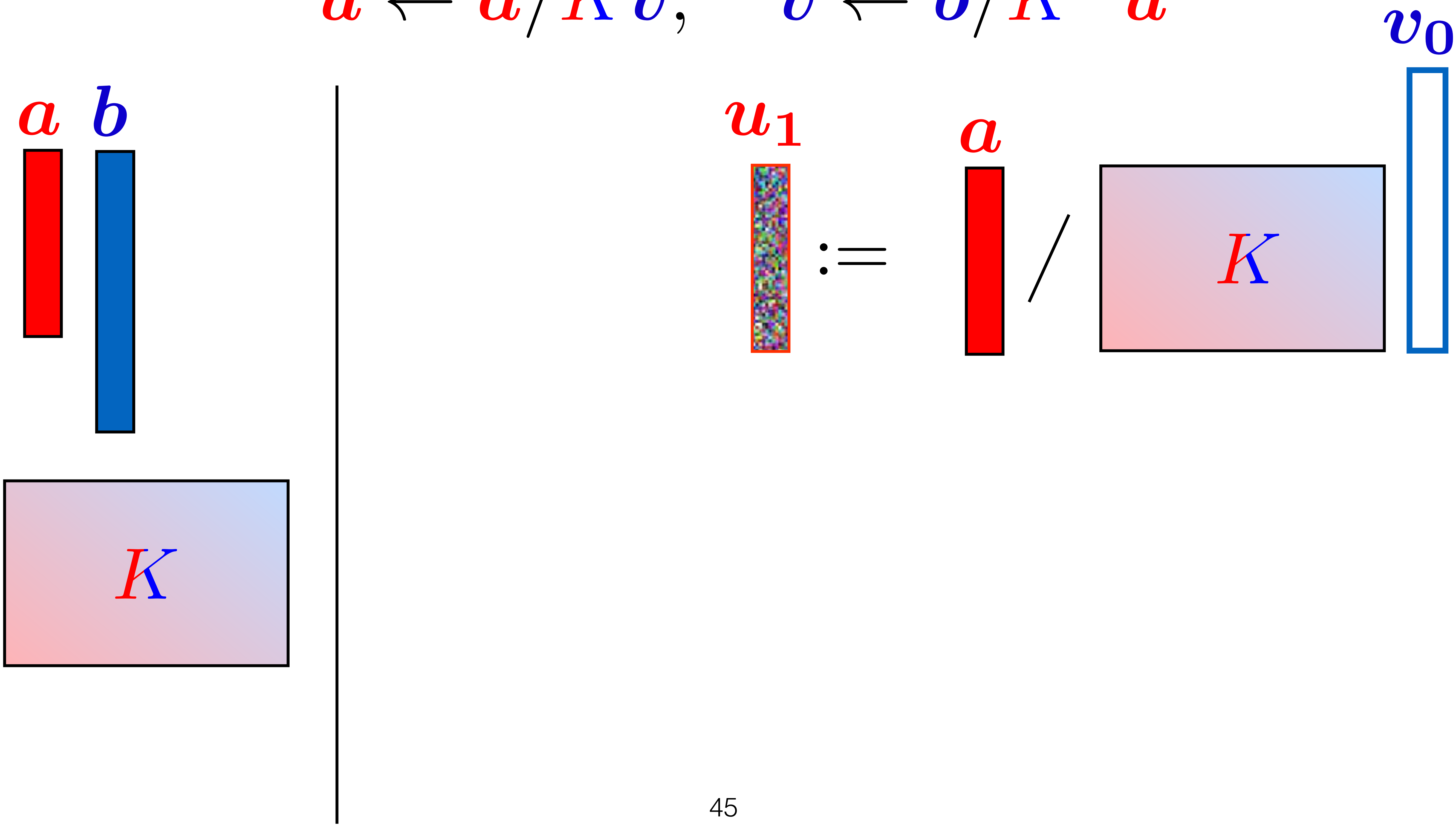




Fast & Scalable Algorithm

- **[Sinkhorn'64]** fixed-point iterations for $(\boldsymbol{u}, \boldsymbol{v})$

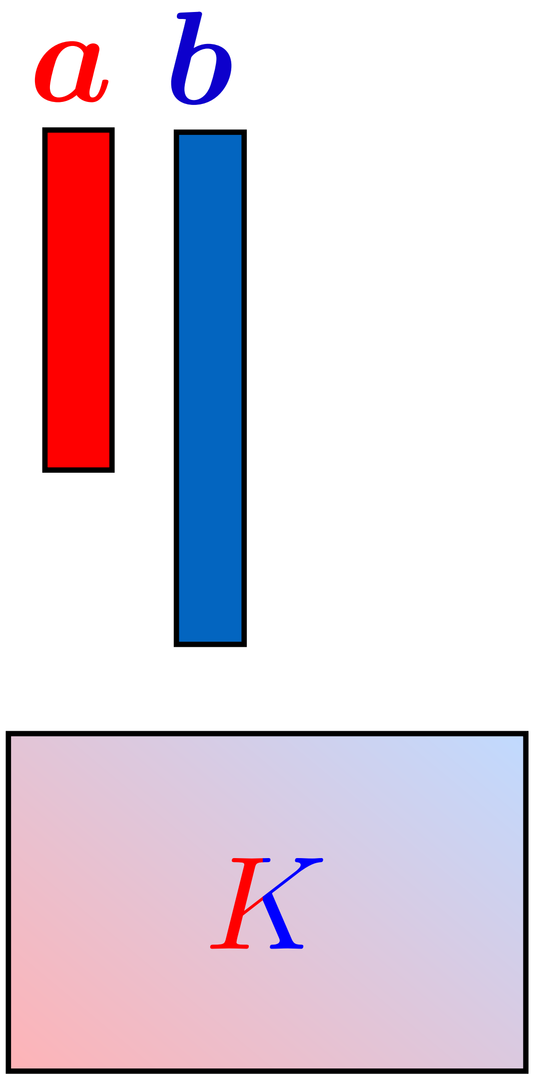
$$\boldsymbol{u} \leftarrow \boldsymbol{a}/K\boldsymbol{v}, \quad \boldsymbol{v} \leftarrow \boldsymbol{b}/K^T\boldsymbol{u}$$

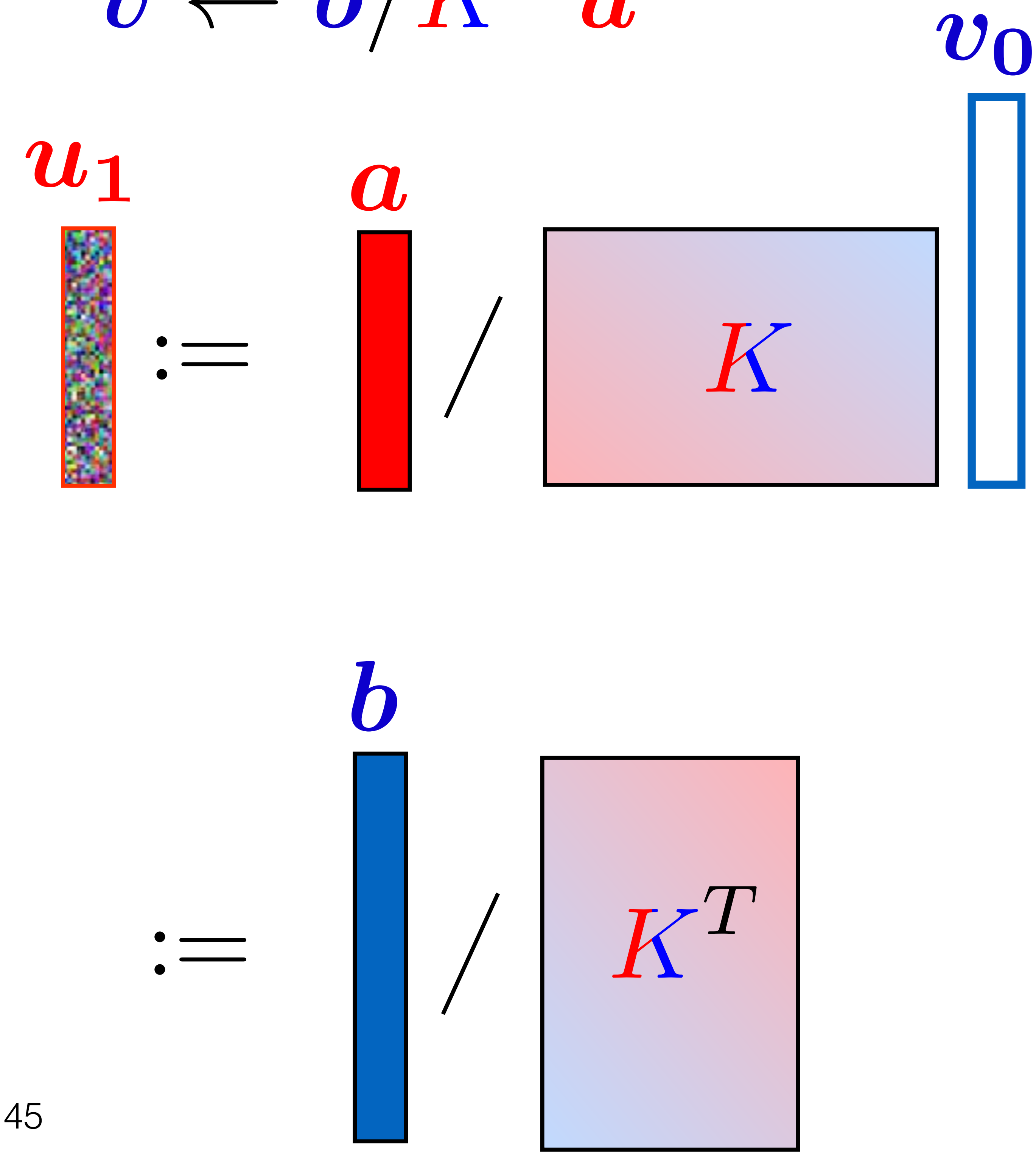


Fast & Scalable Algorithm

- **[Sinkhorn'64]** fixed-point iterations for $(\boldsymbol{u}, \boldsymbol{v})$

$$\boldsymbol{u} \leftarrow \boldsymbol{a}/K\boldsymbol{v}, \quad \boldsymbol{v} \leftarrow \boldsymbol{b}/K^T\boldsymbol{u}$$

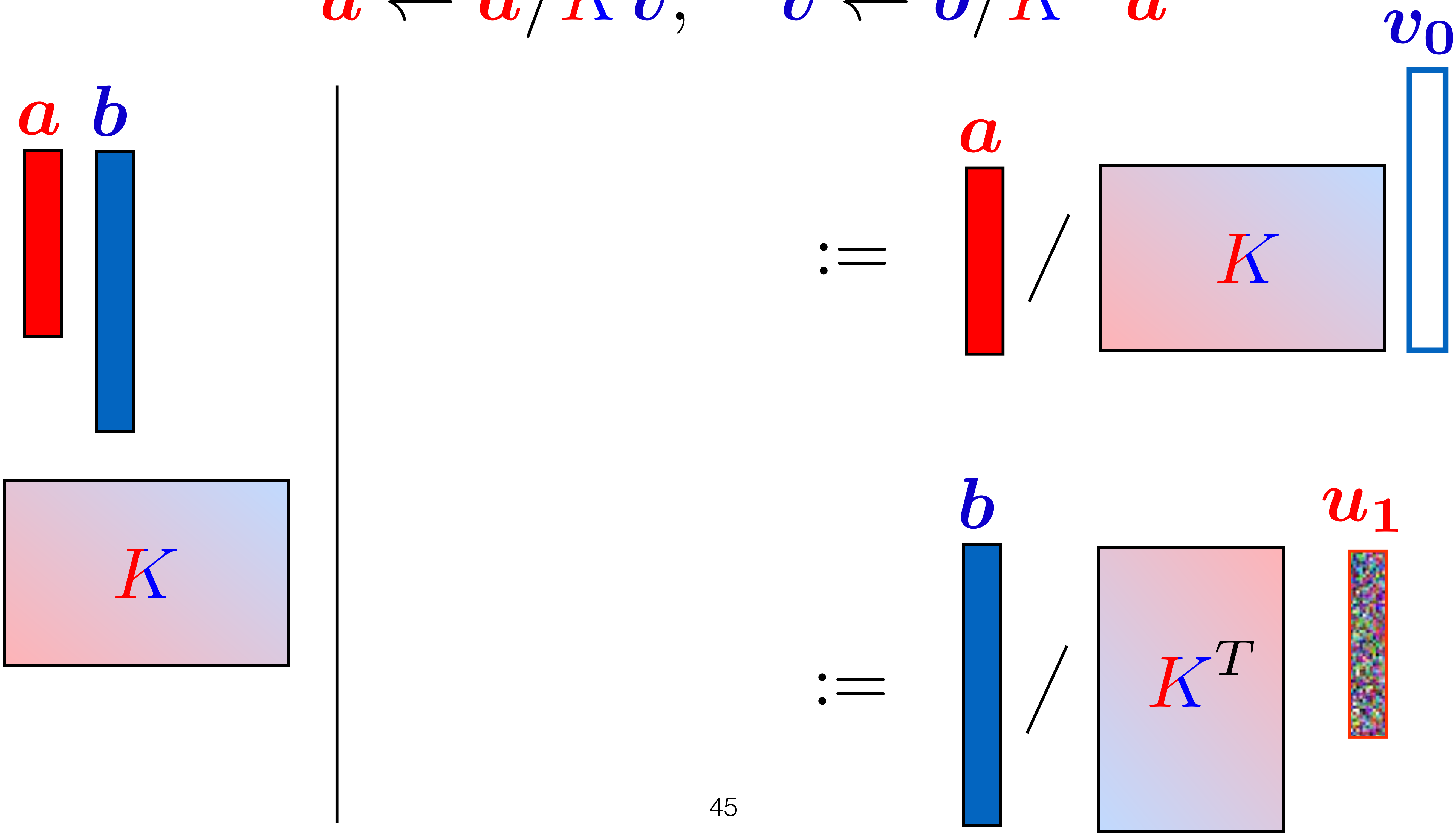




Fast & Scalable Algorithm

- **[Sinkhorn'64]** fixed-point iterations for $(\boldsymbol{u}, \boldsymbol{v})$

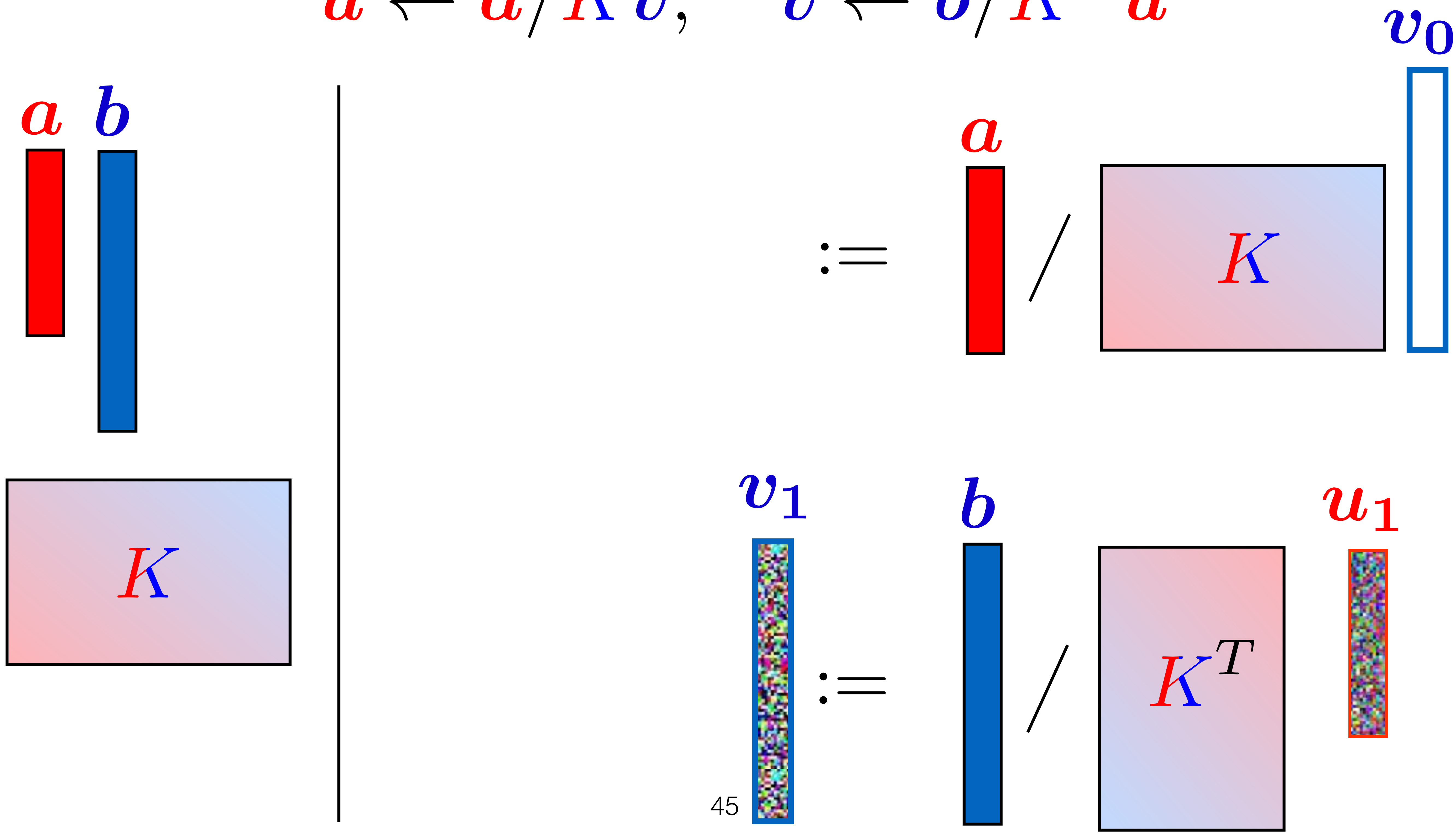
$$\boldsymbol{u} \leftarrow \boldsymbol{a}/K\boldsymbol{v}, \quad \boldsymbol{v} \leftarrow \boldsymbol{b}/K^T\boldsymbol{u}$$



Fast & Scalable Algorithm

- **[Sinkhorn’64]** fixed-point iterations for $(\boldsymbol{u}, \boldsymbol{v})$

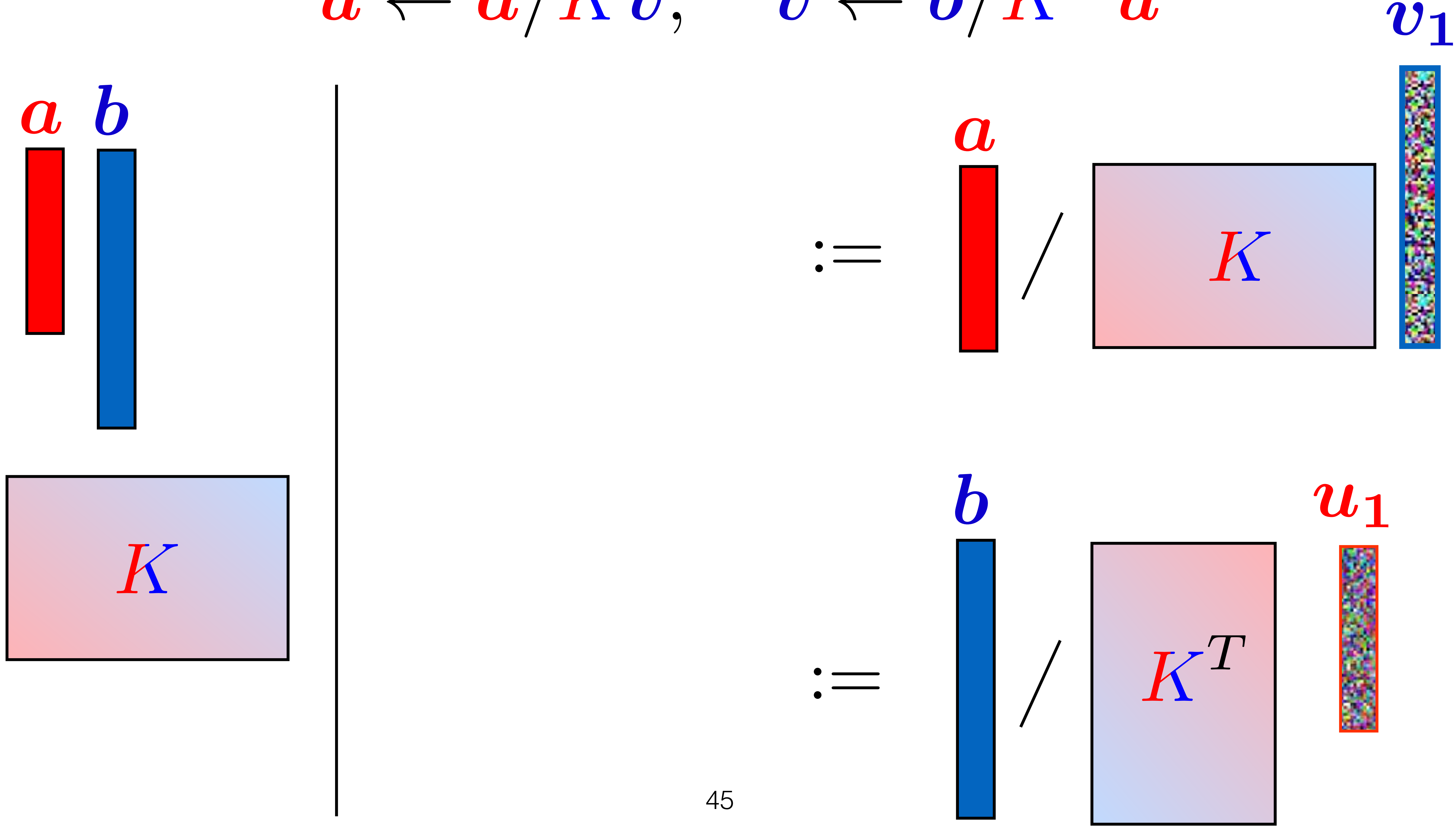
$$\boldsymbol{u} \leftarrow \boldsymbol{a}/K\boldsymbol{v}, \quad \boldsymbol{v} \leftarrow \boldsymbol{b}/K^T\boldsymbol{u}$$



Fast & Scalable Algorithm

- **[Sinkhorn'64]** fixed-point iterations for $(\boldsymbol{u}, \boldsymbol{v})$

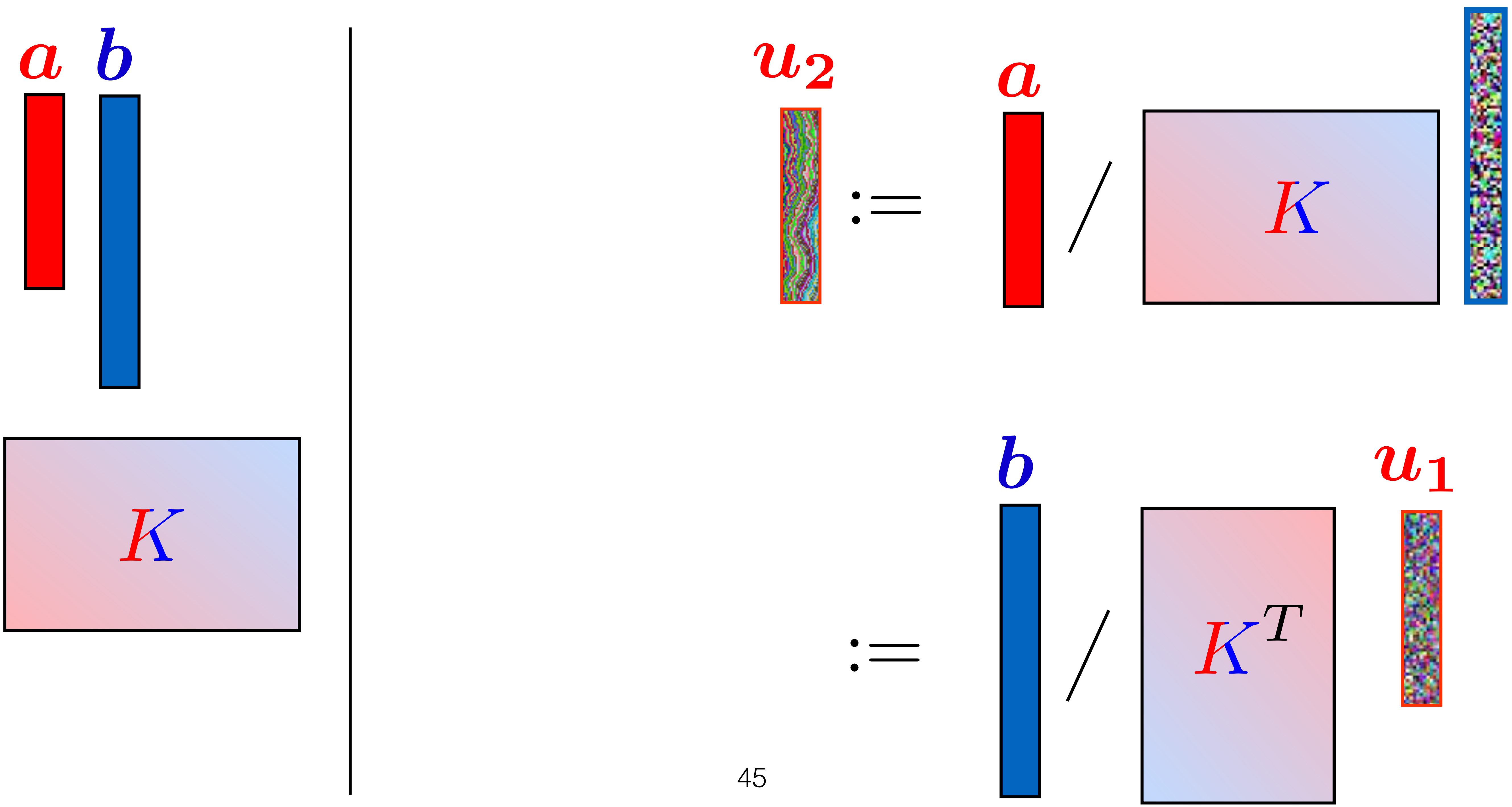
$$\boldsymbol{u} \leftarrow \boldsymbol{a}/K\boldsymbol{v}, \quad \boldsymbol{v} \leftarrow \boldsymbol{b}/K^T\boldsymbol{u}$$



Fast & Scalable Algorithm

- **[Sinkhorn'64]** fixed-point iterations for $(\boldsymbol{u}, \boldsymbol{v})$

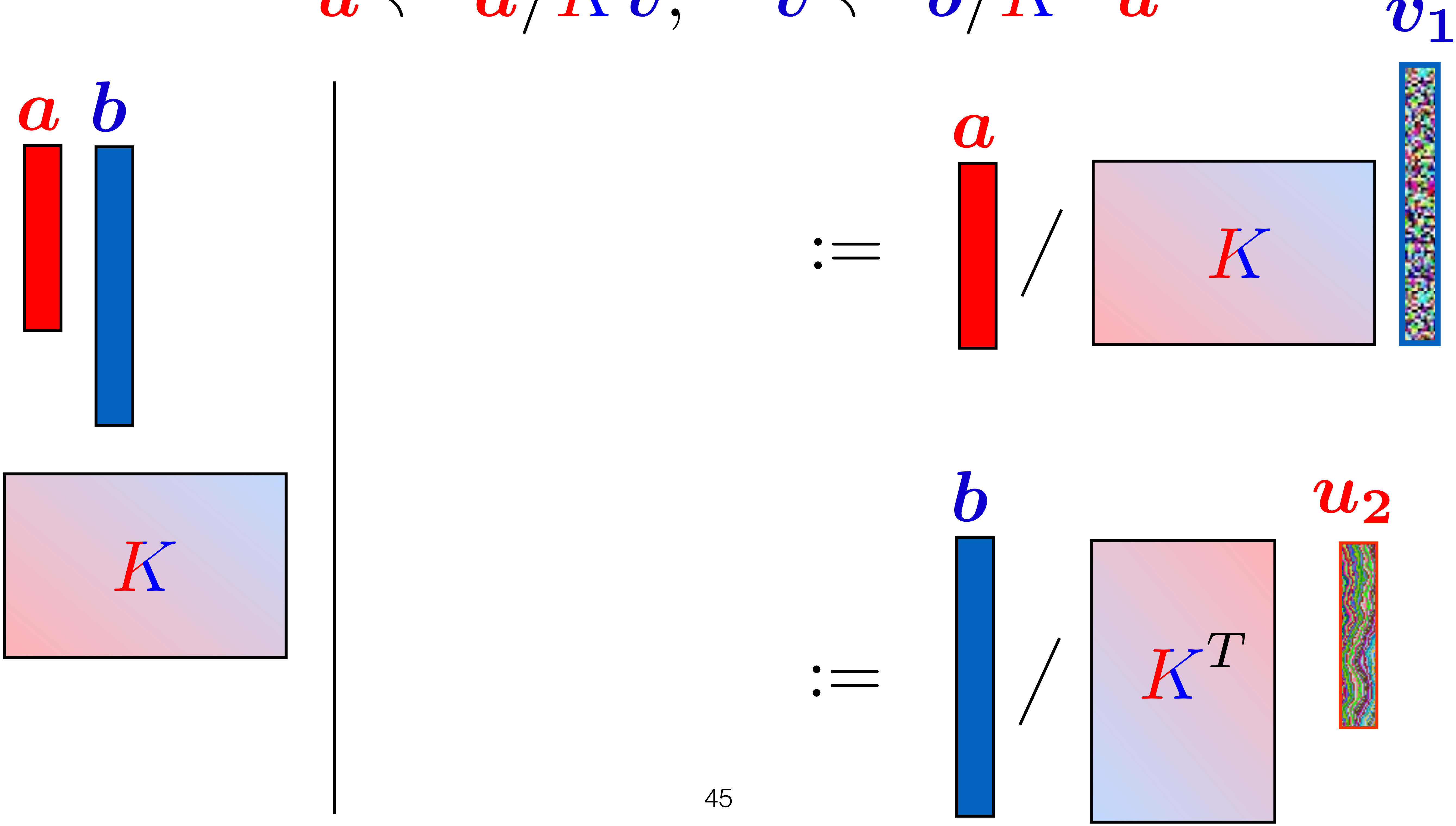
$$\boldsymbol{u} \leftarrow \boldsymbol{a}/K\boldsymbol{v}, \quad \boldsymbol{v} \leftarrow \boldsymbol{b}/K^T\boldsymbol{u}$$



Fast & Scalable Algorithm

- **[Sinkhorn'64]** fixed-point iterations for (u, v)

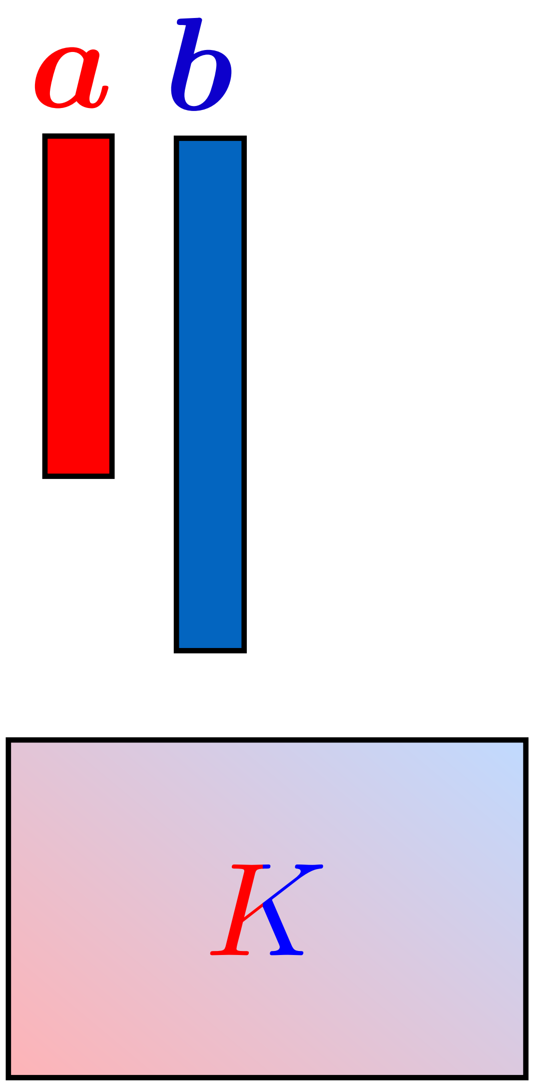
$$u \leftarrow a / K v, \quad v \leftarrow b / K^T u$$

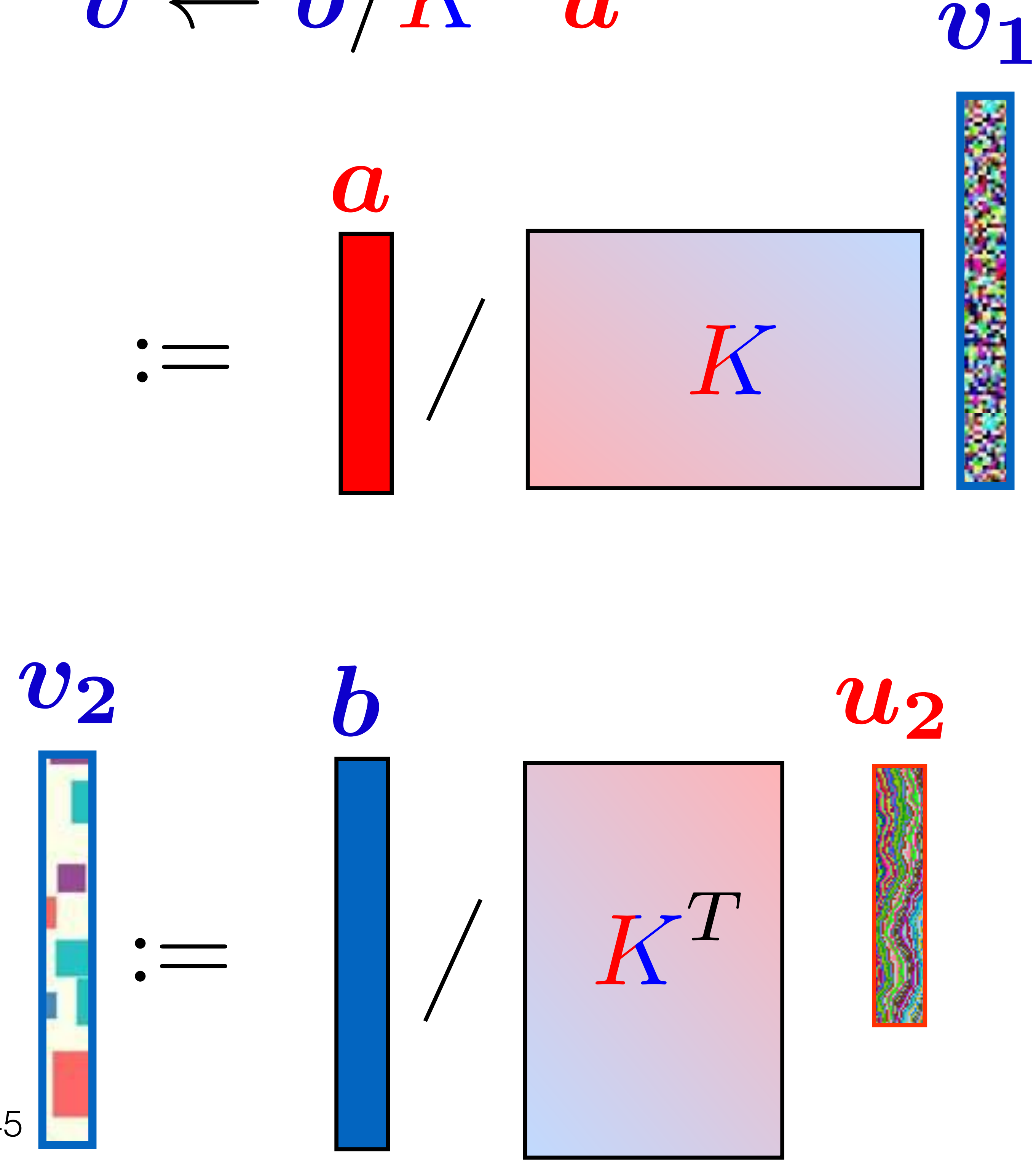


Fast & Scalable Algorithm

- **[Sinkhorn'64]** fixed-point iterations for $(\boldsymbol{u}, \boldsymbol{v})$

$$\boldsymbol{u} \leftarrow \boldsymbol{a}/K\boldsymbol{v}, \quad \boldsymbol{v} \leftarrow \boldsymbol{b}/K^T\boldsymbol{u}$$

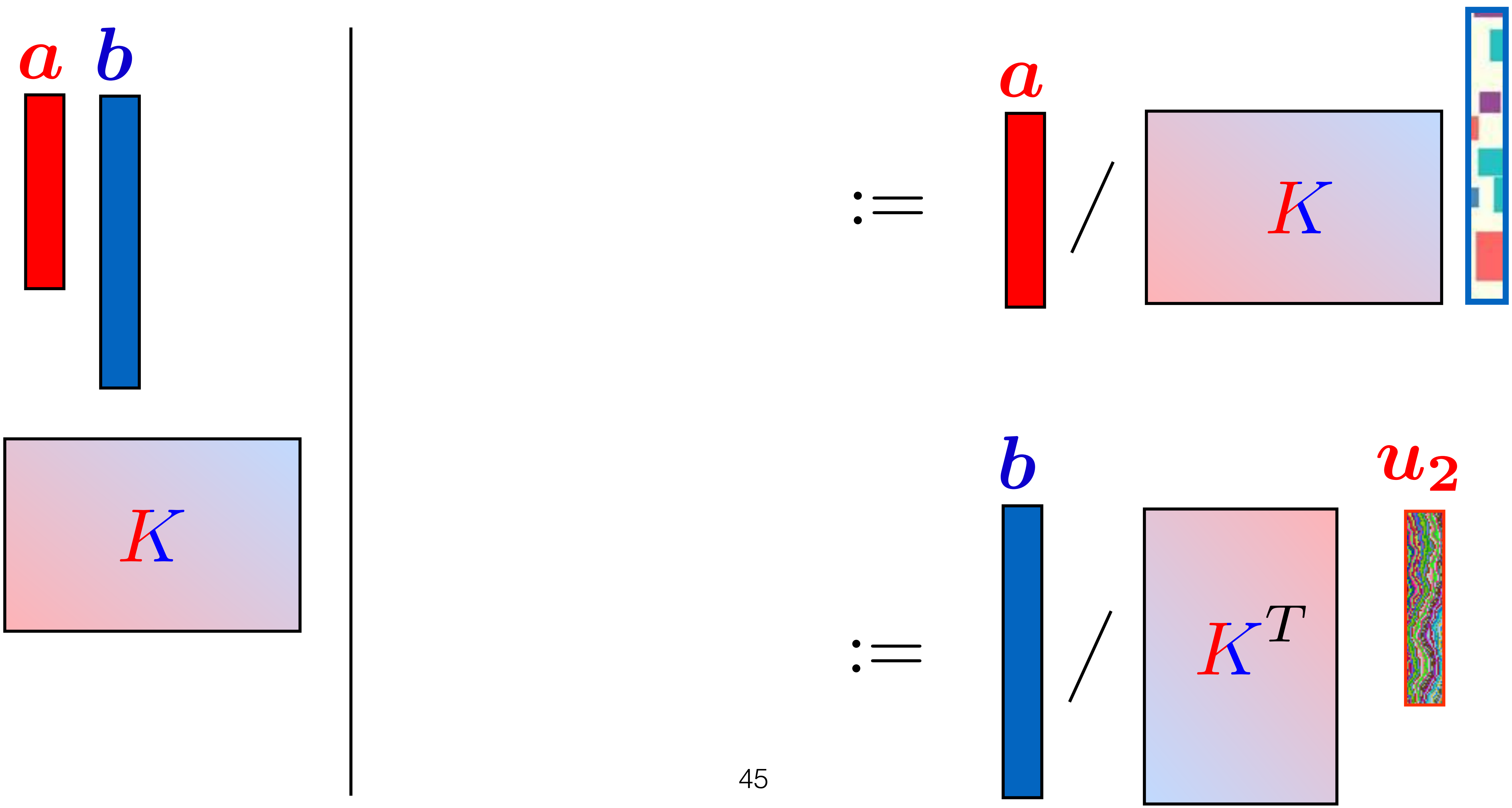




Fast & Scalable Algorithm

- **[Sinkhorn'64]** fixed-point iterations for $(\boldsymbol{u}, \boldsymbol{v})$

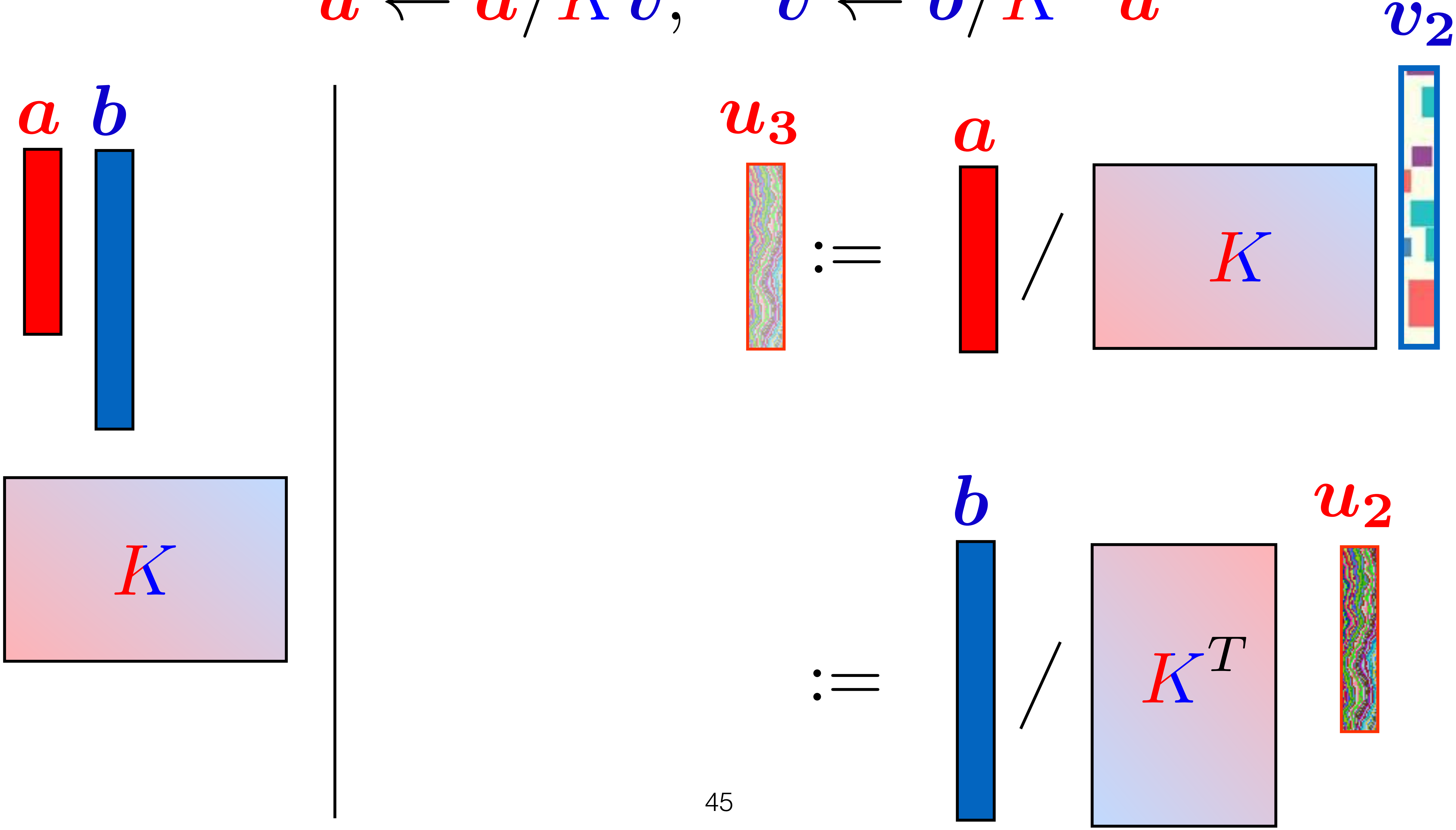
$$\boldsymbol{u} \leftarrow \boldsymbol{a}/K\boldsymbol{v}, \quad \boldsymbol{v} \leftarrow \boldsymbol{b}/K^T\boldsymbol{u}$$



Fast & Scalable Algorithm

- **[Sinkhorn'64]** fixed-point iterations for $(\boldsymbol{u}, \boldsymbol{v})$

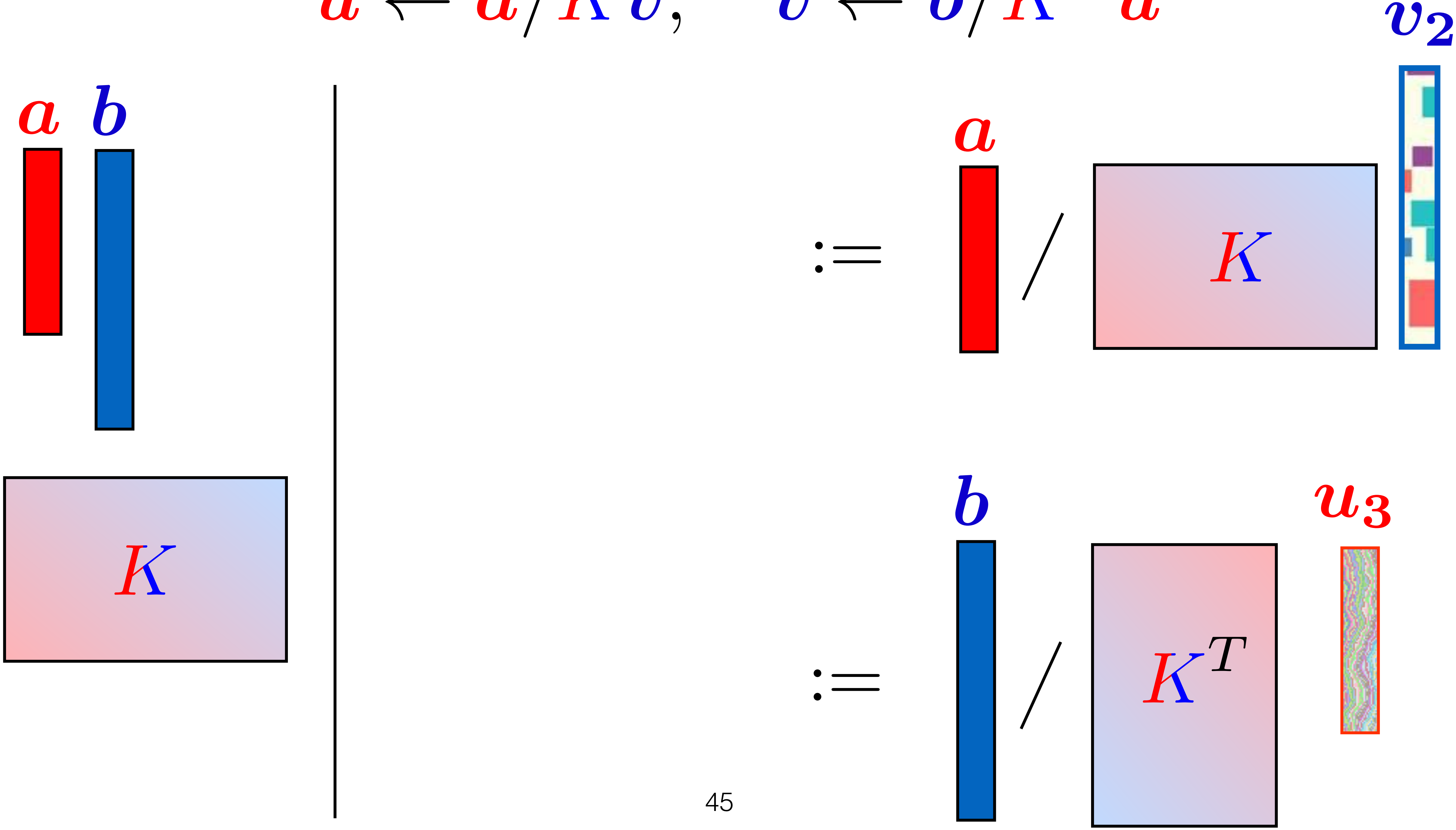
$$\boldsymbol{u} \leftarrow \boldsymbol{a}/K\boldsymbol{v}, \quad \boldsymbol{v} \leftarrow \boldsymbol{b}/K^T\boldsymbol{u}$$



Fast & Scalable Algorithm

- **[Sinkhorn'64]** fixed-point iterations for $(\boldsymbol{u}, \boldsymbol{v})$

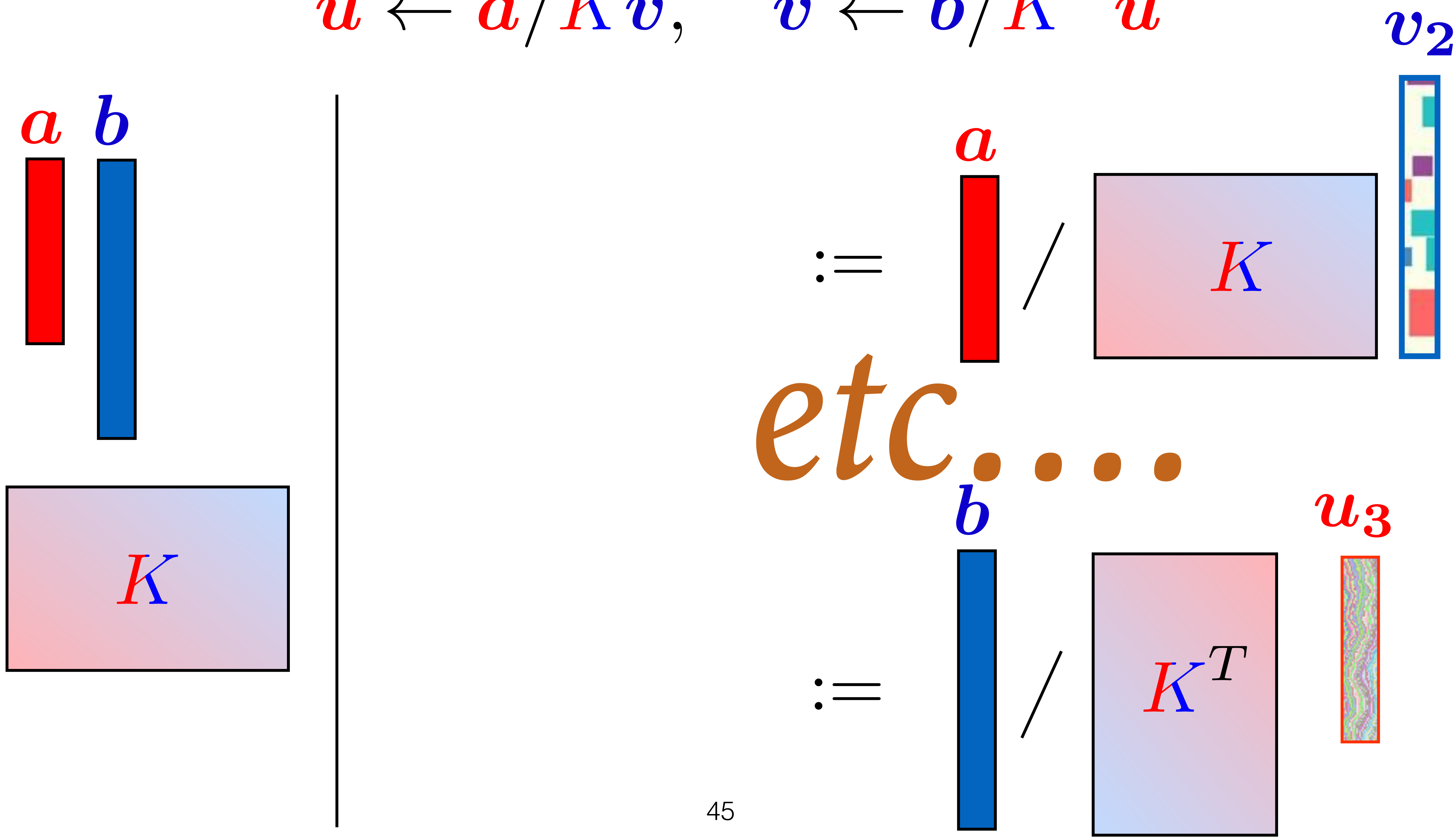
$$\boldsymbol{u} \leftarrow \boldsymbol{a}/K\boldsymbol{v}, \quad \boldsymbol{v} \leftarrow \boldsymbol{b}/K^T\boldsymbol{u}$$



Fast & Scalable Algorithm

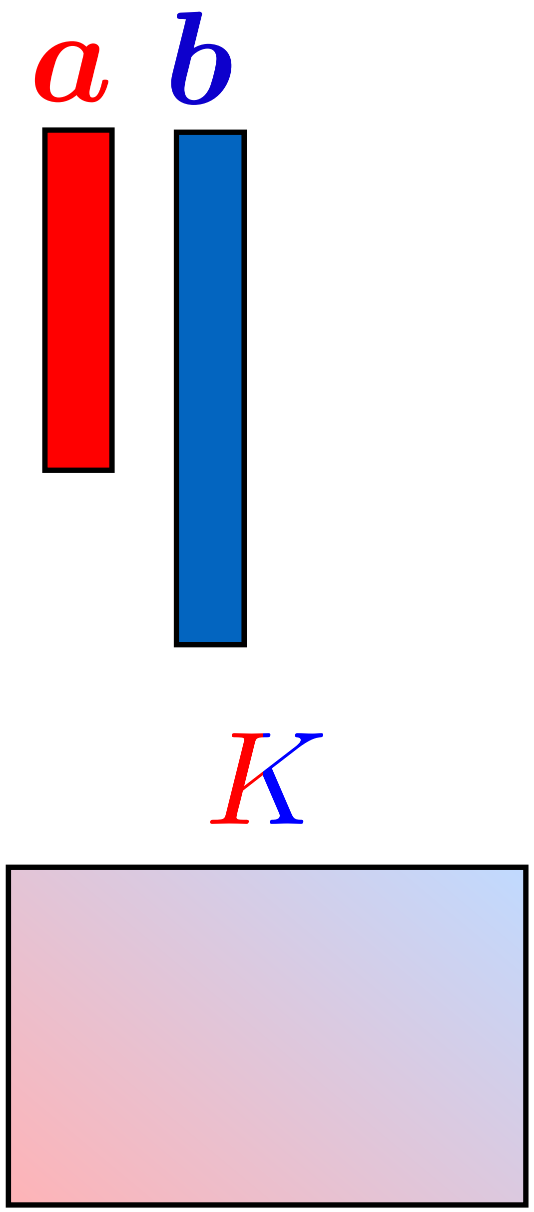
- **[Sinkhorn'64]** fixed-point iterations for $(\boldsymbol{u}, \boldsymbol{v})$

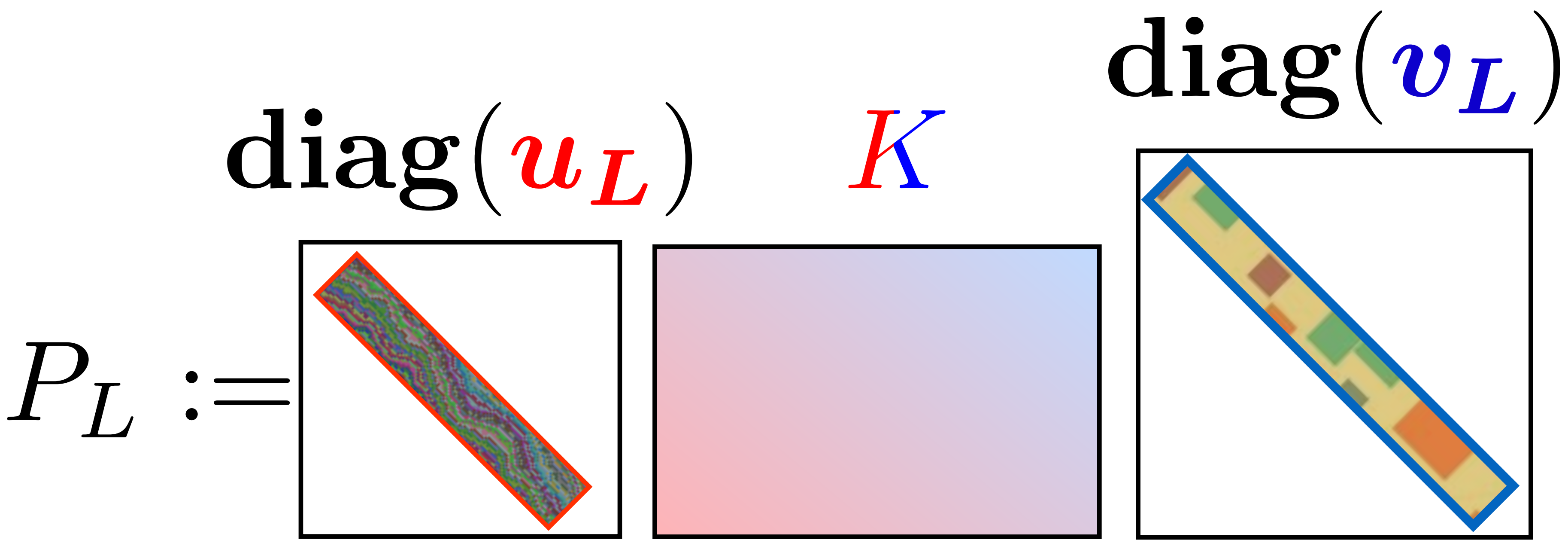
$$\boldsymbol{u} \leftarrow \boldsymbol{a} / K \boldsymbol{v}, \quad \boldsymbol{v} \leftarrow \boldsymbol{b} / K^T \boldsymbol{u}$$



Fast & Scalable Algorithm

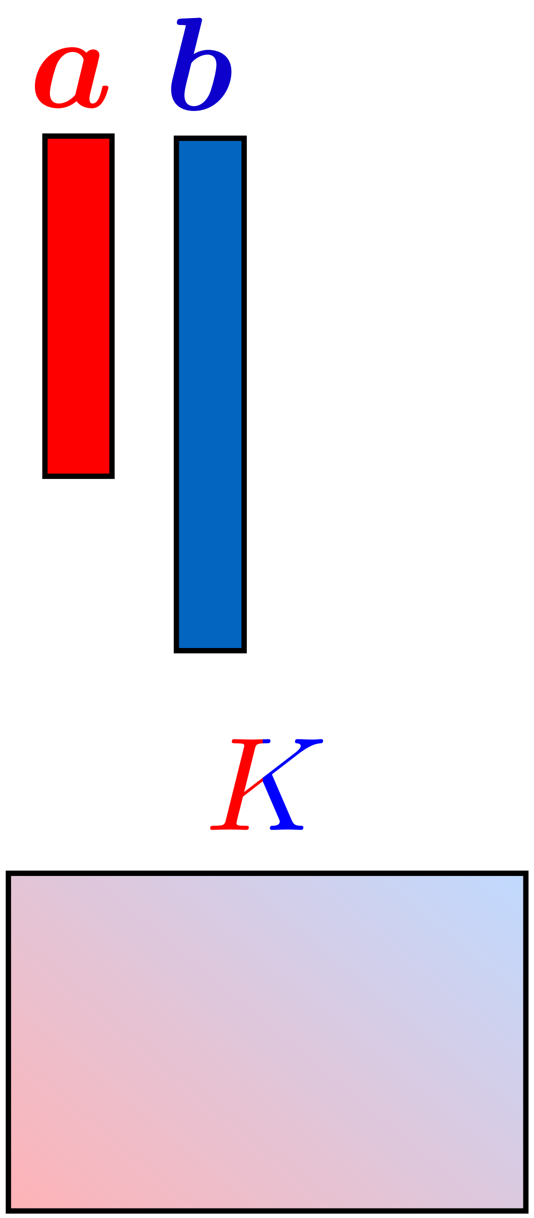
- **[Sinkhorn'64]** fixed-point iterations.



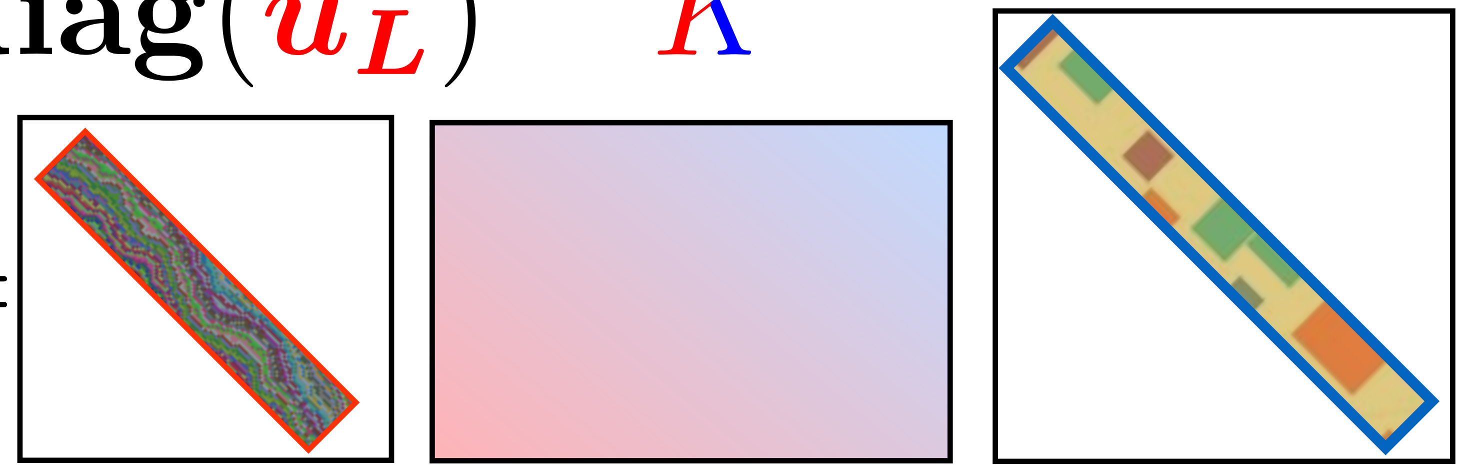


Fast & Scalable Algorithm

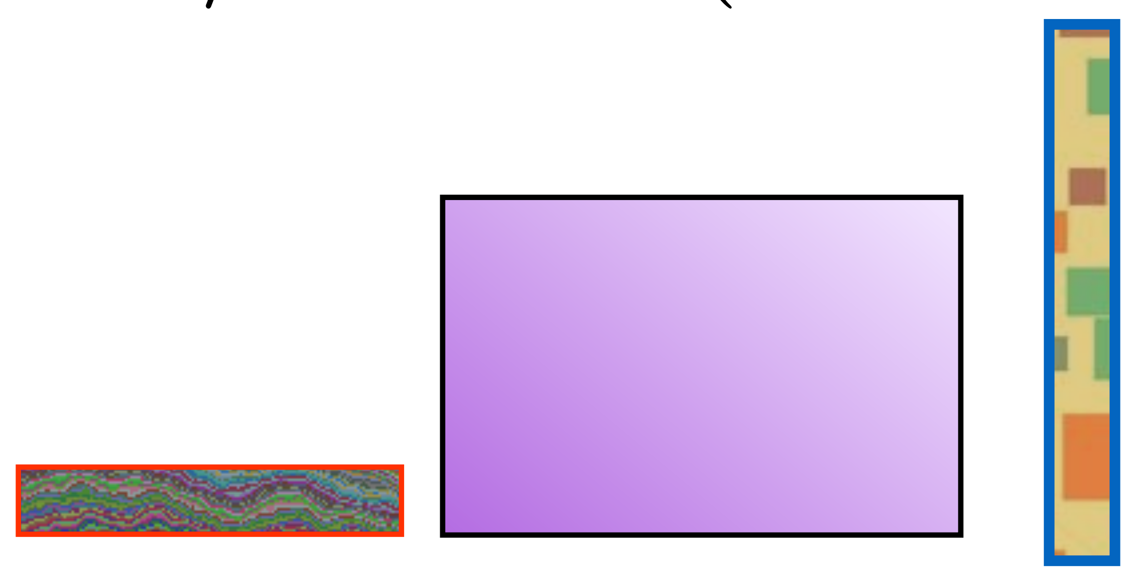
- **[Sinkhorn'64]** fixed-point iterations.



$$P_L := \mathbf{diag}(\boldsymbol{u_L})\, K\, \mathbf{diag}(\boldsymbol{v_L})$$

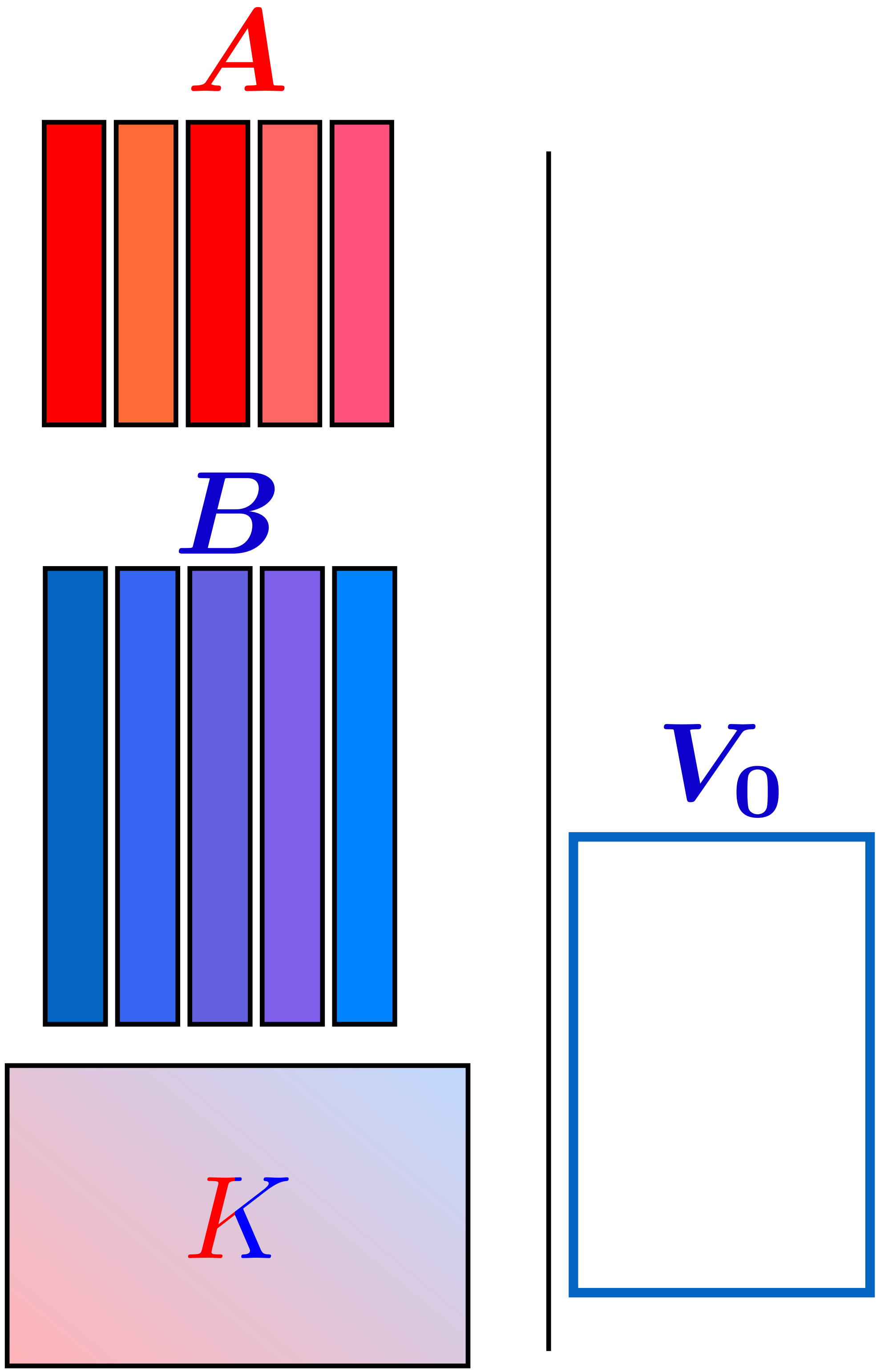


$$\langle P_L, M_{\boldsymbol{XY}} \rangle = \boldsymbol{u_L}^T (K \odot M_{\boldsymbol{XY}}) \boldsymbol{v_L}$$



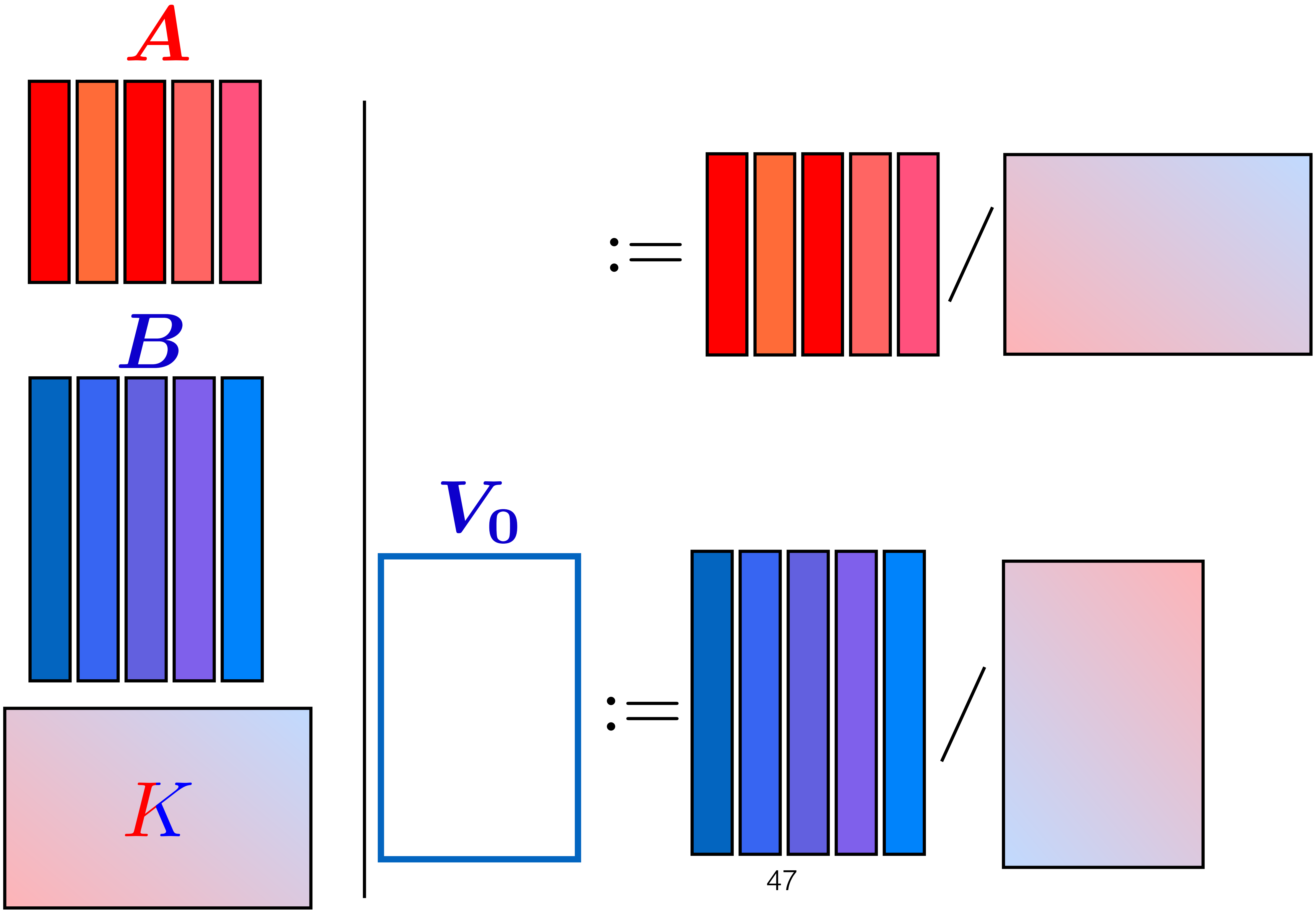
Also embarrassingly parallel

- **[Sinkhorn'64]** with *matrix* fixed-point iterations



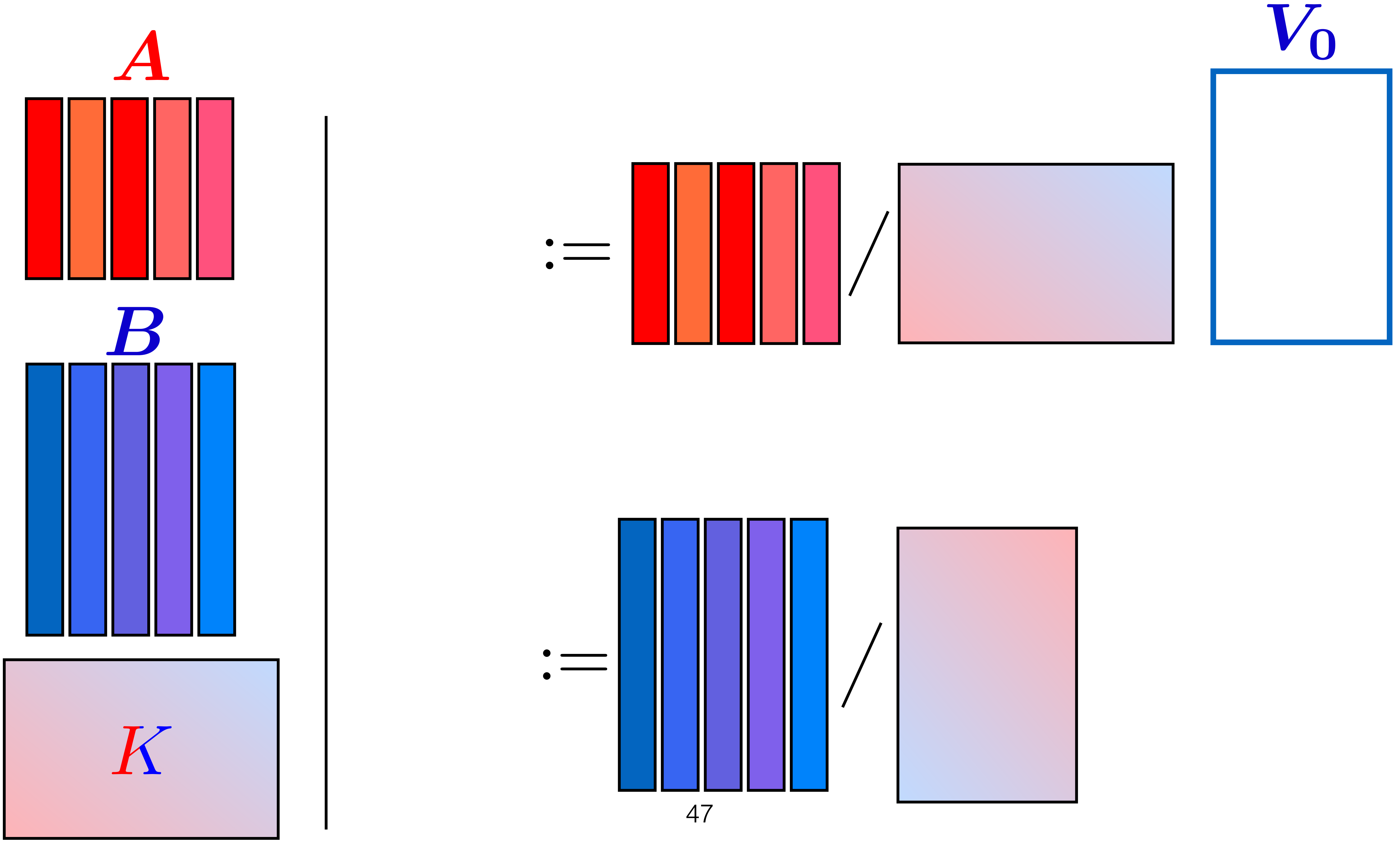
Also embarrassingly parallel

- **[Sinkhorn'64]** with *matrix* fixed-point iterations



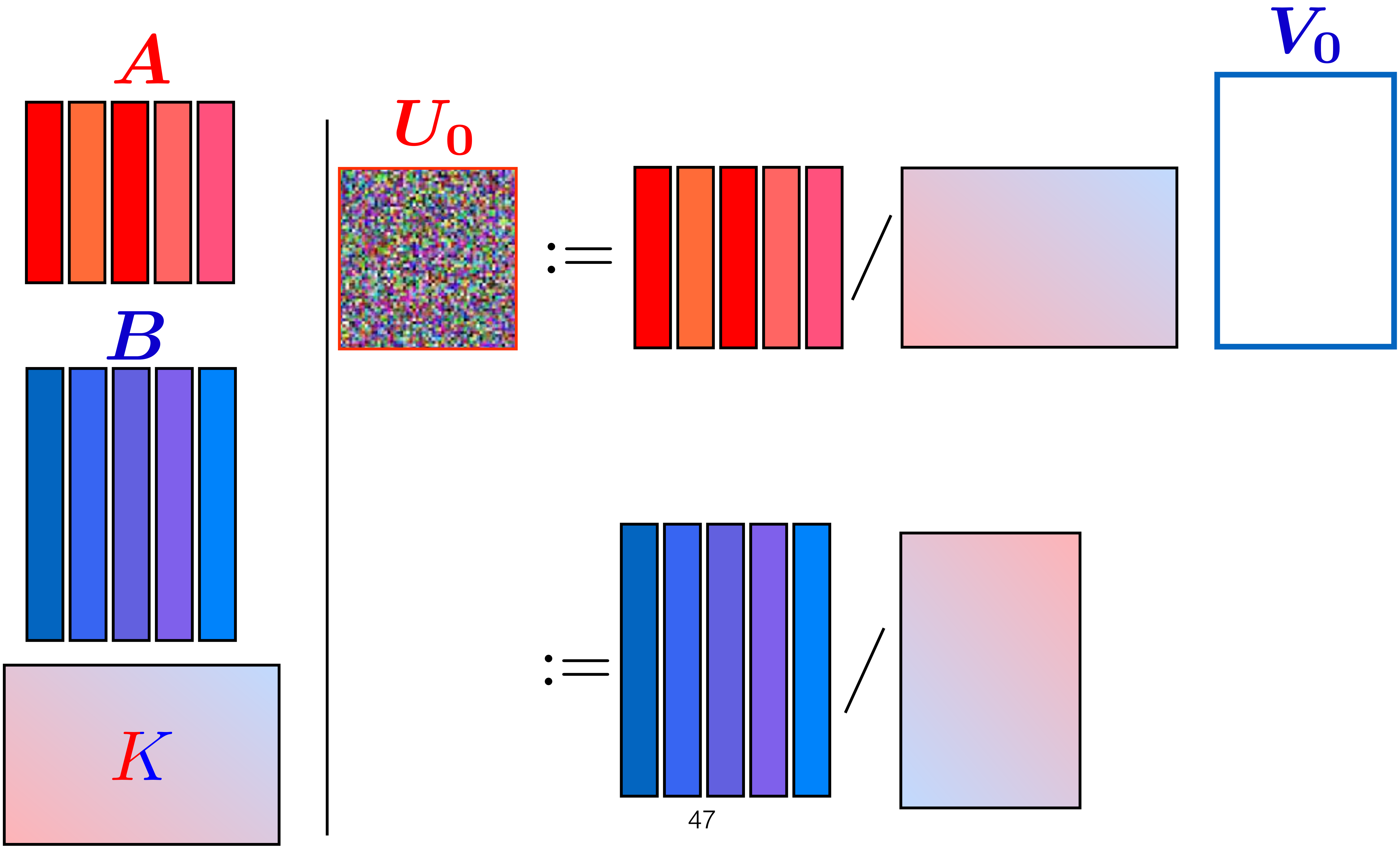
Also embarrassingly parallel

- **[Sinkhorn'64]** with *matrix* fixed-point iterations



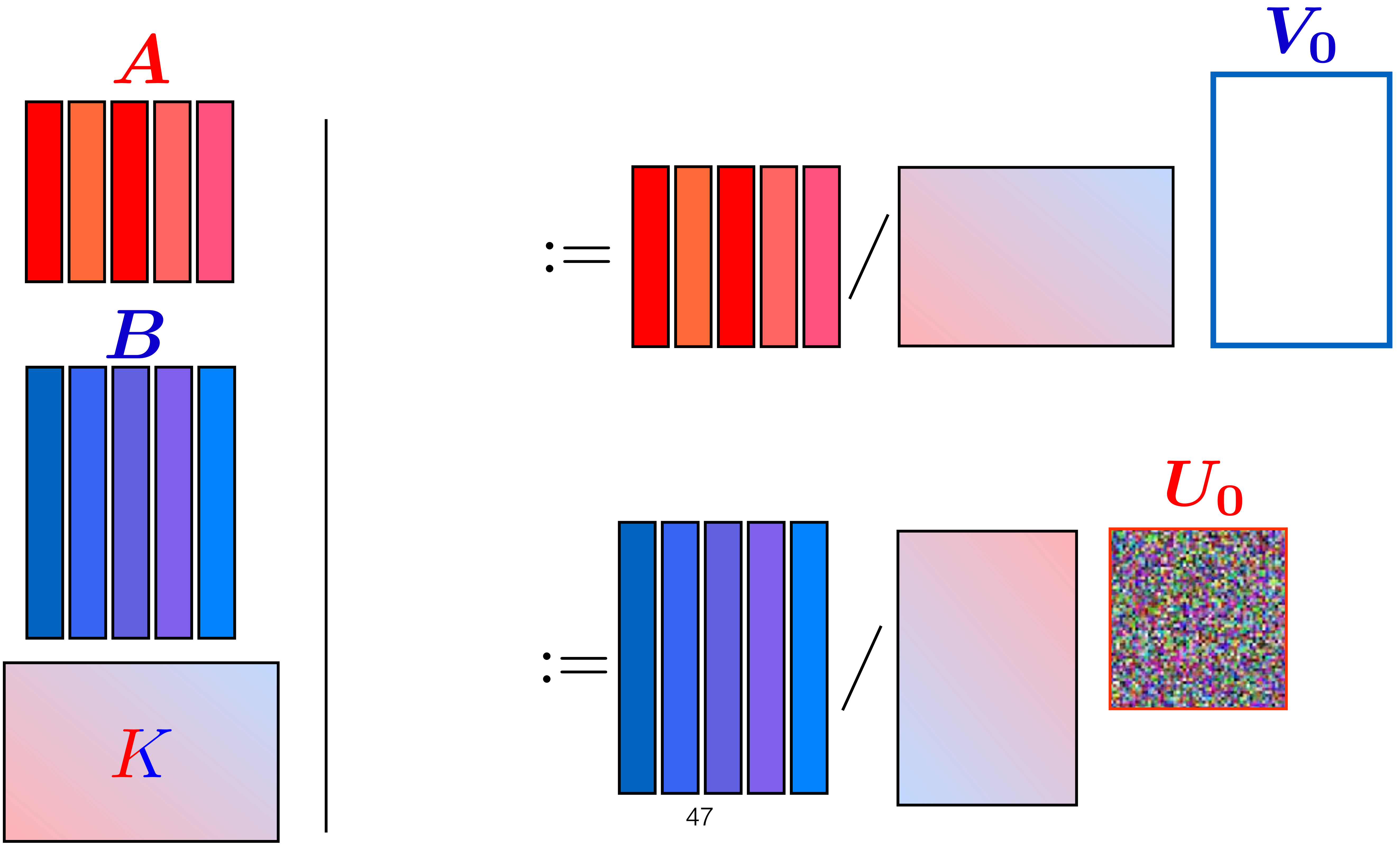
Also embarrassingly parallel

- **[Sinkhorn'64]** with *matrix* fixed-point iterations



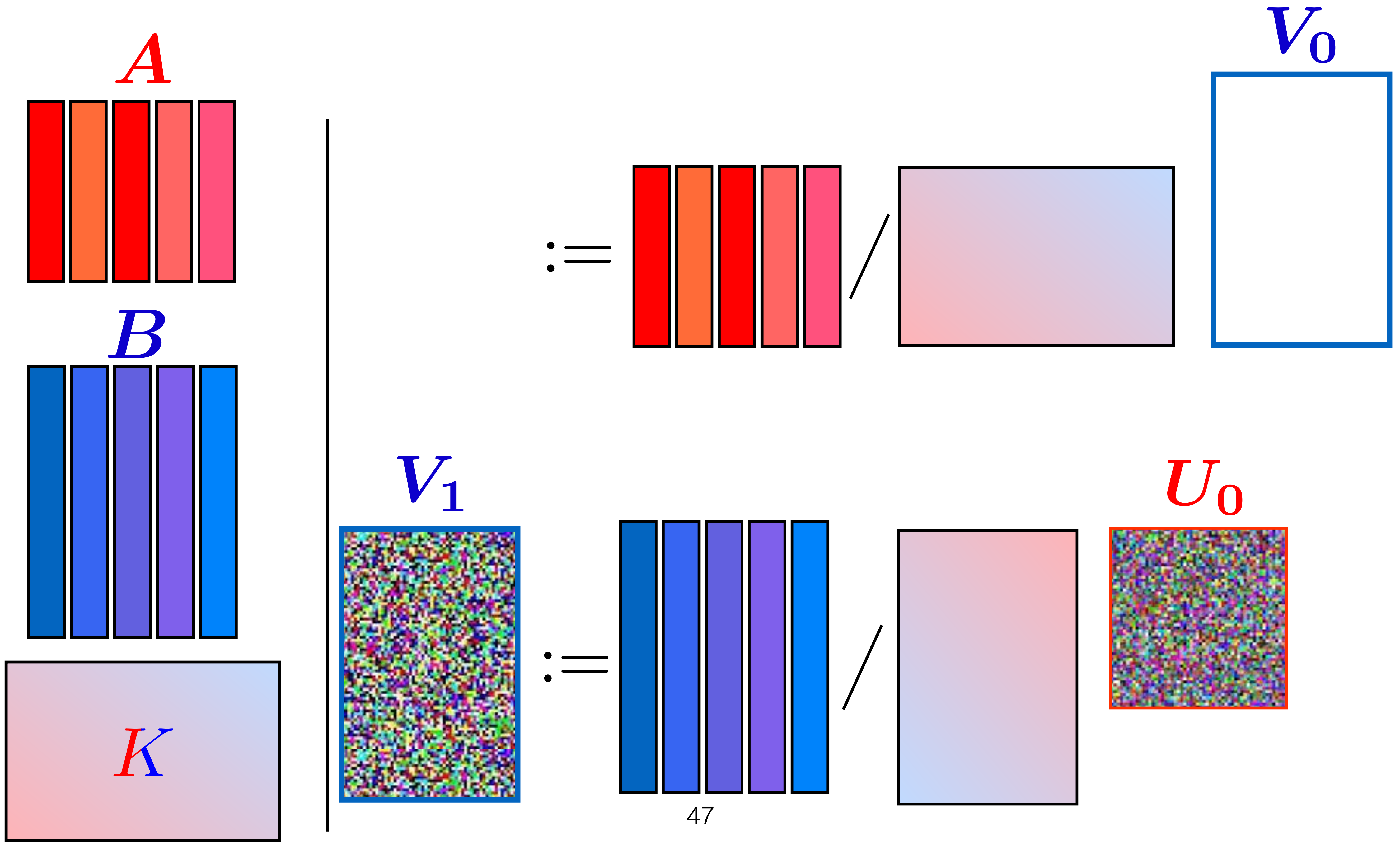
Also embarrassingly parallel

- **[Sinkhorn'64]** with *matrix* fixed-point iterations



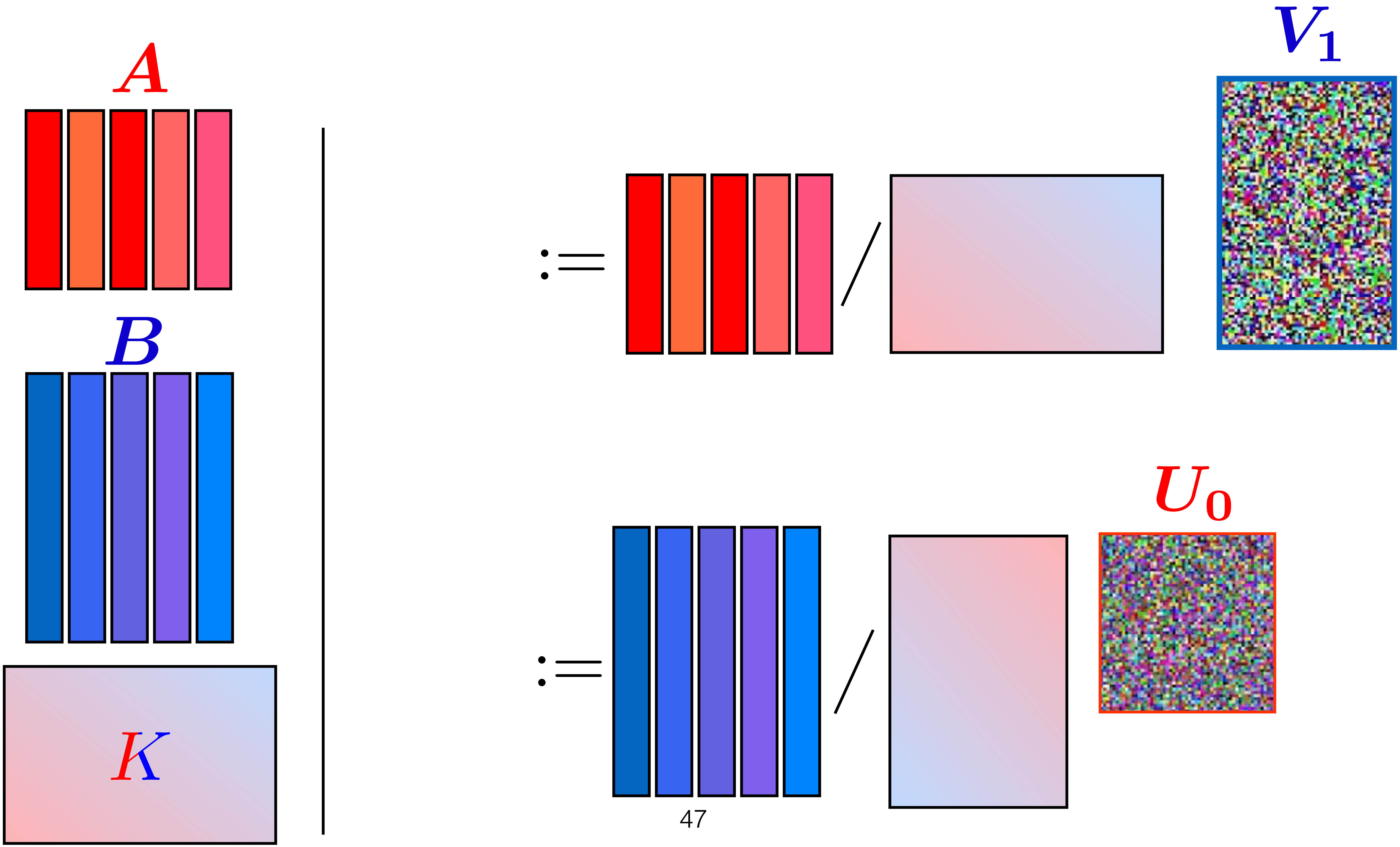
Also embarrassingly parallel

- **[Sinkhorn'64]** with *matrix* fixed-point iterations



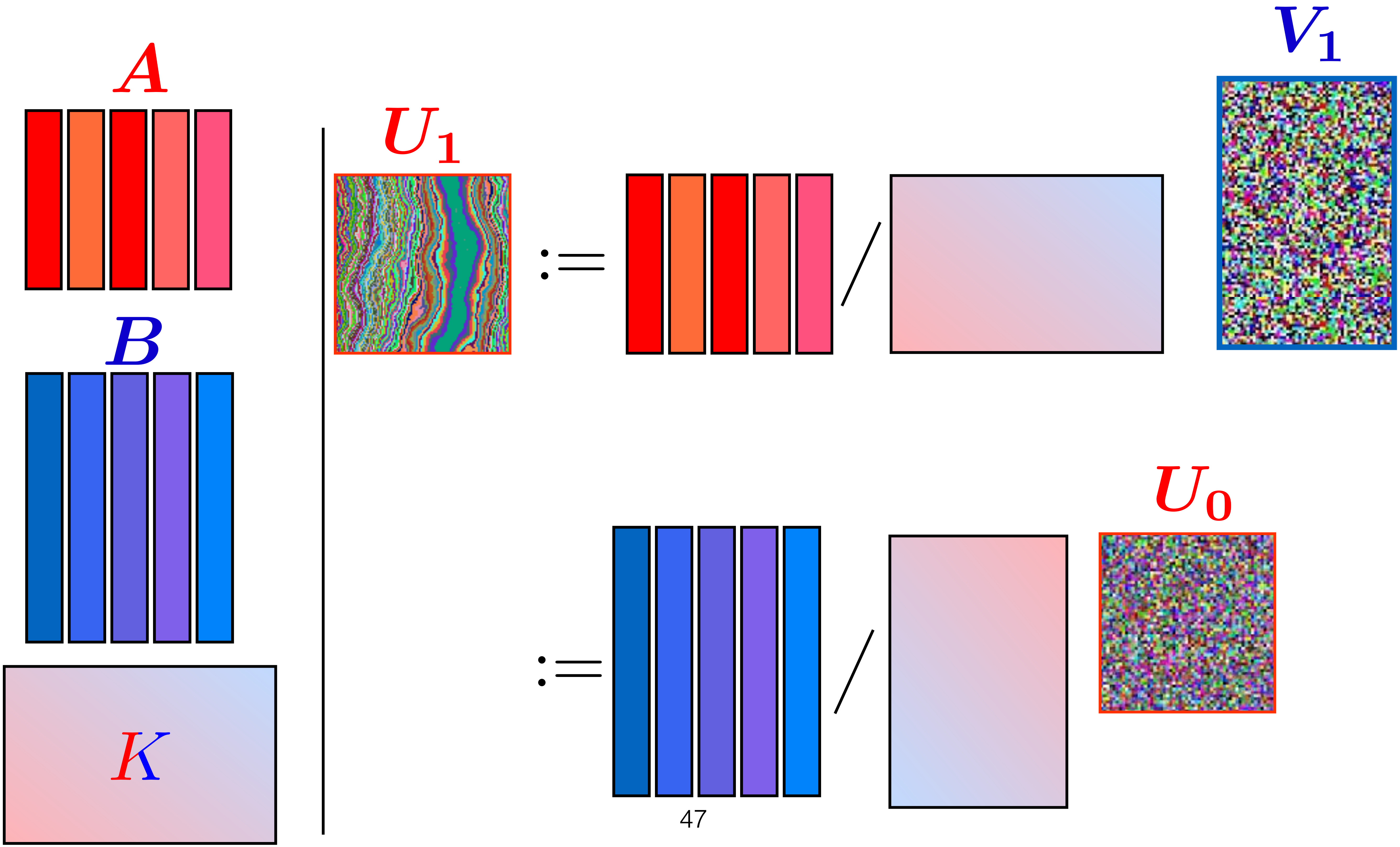
Also embarrassingly parallel

- **[Sinkhorn'64]** with *matrix* fixed-point iterations



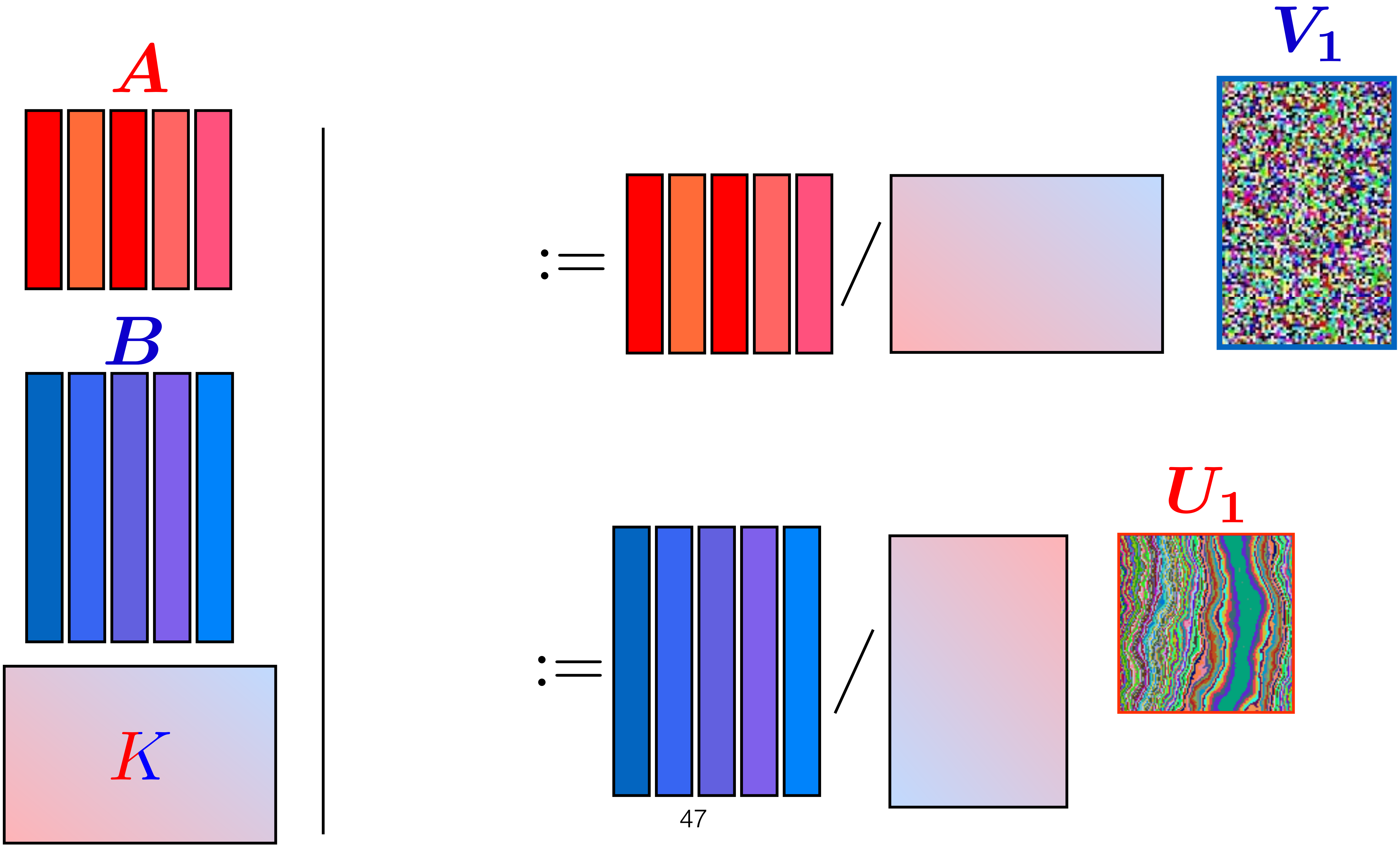
Also embarrassingly parallel

- **[Sinkhorn'64]** with *matrix* fixed-point iterations



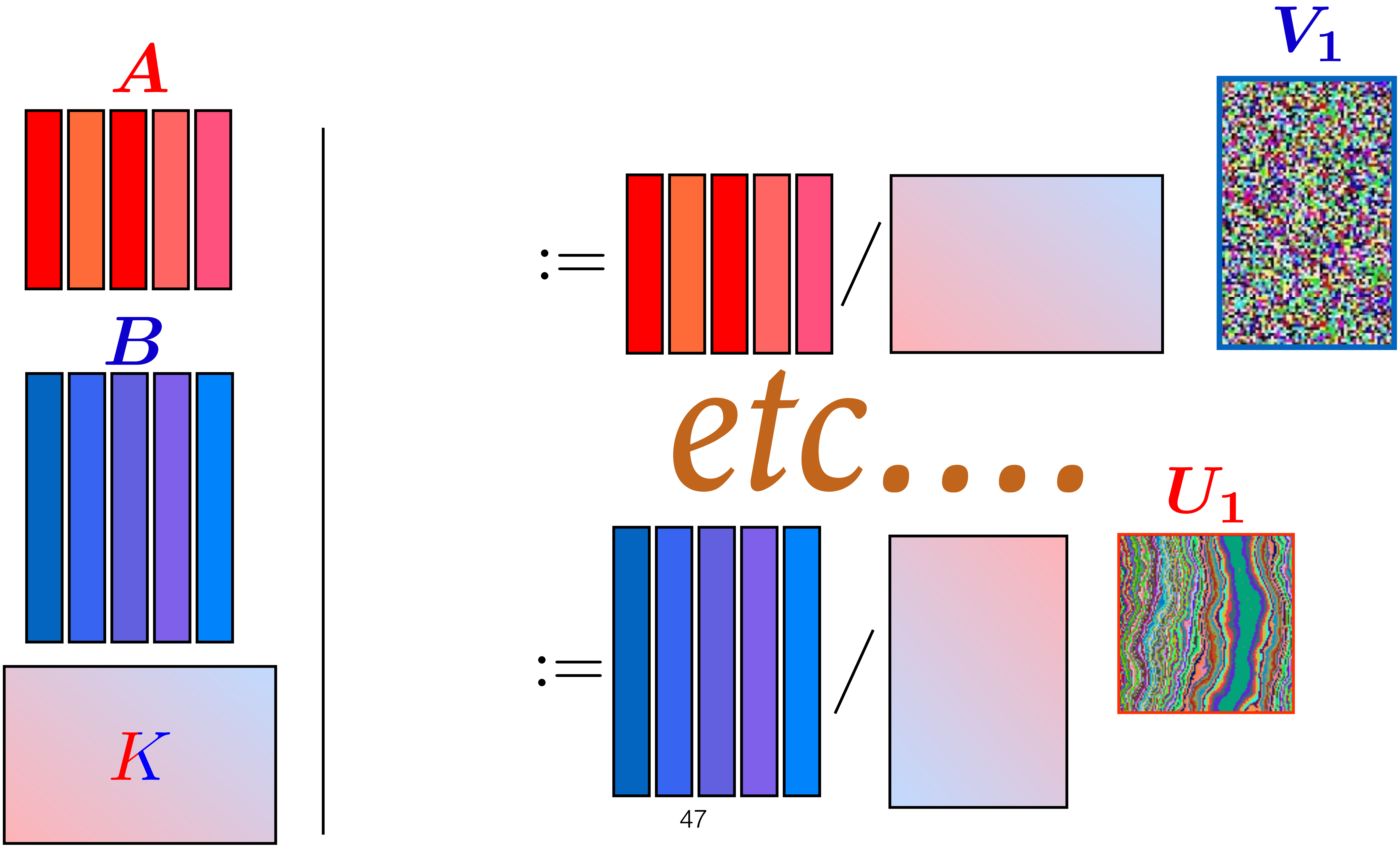
Also embarrassingly parallel

- **[Sinkhorn'64]** with *matrix* fixed-point iterations



Also embarrassingly parallel

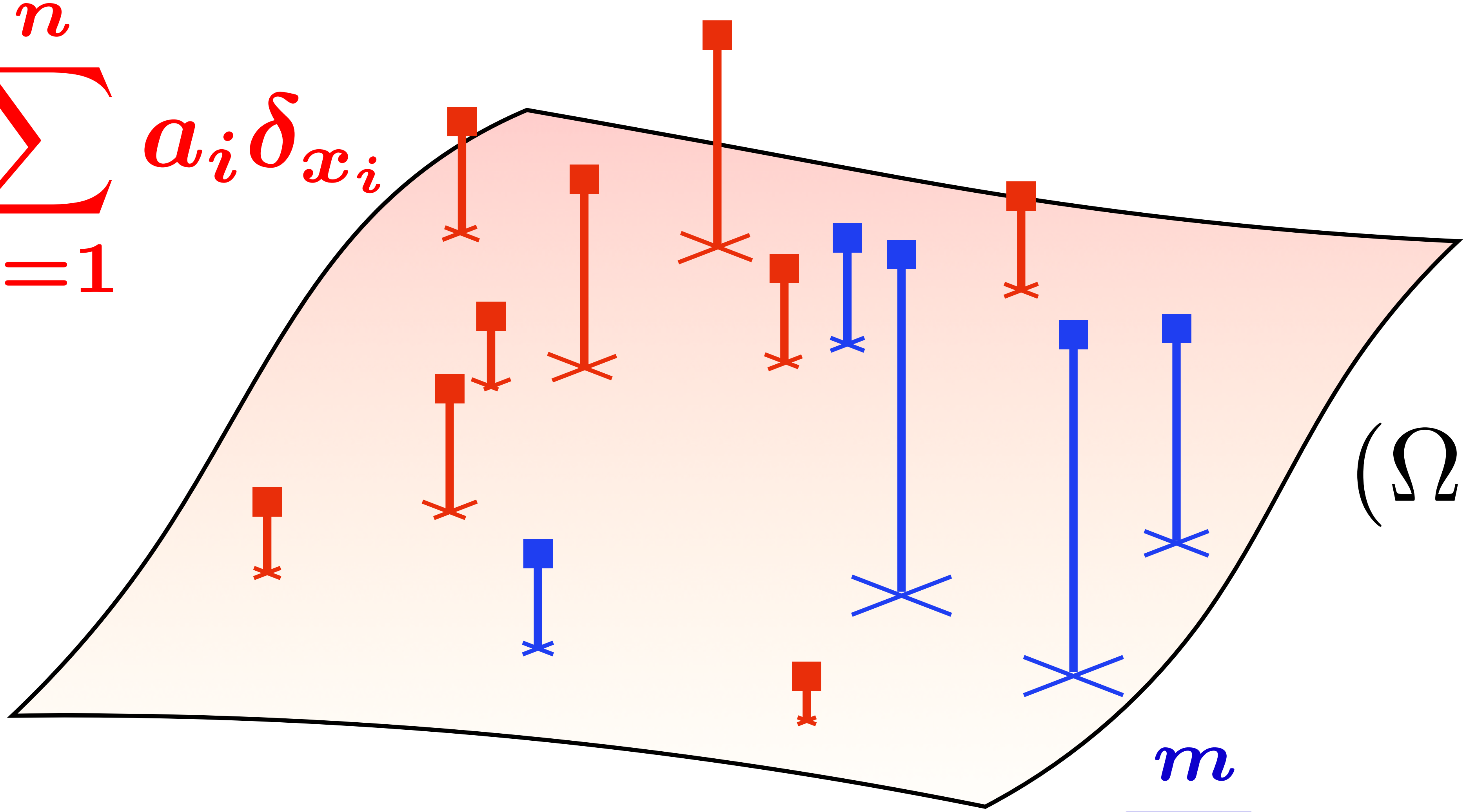
- **[Sinkhorn'64]** with *matrix* fixed-point iterations



Differentiability of W

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

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



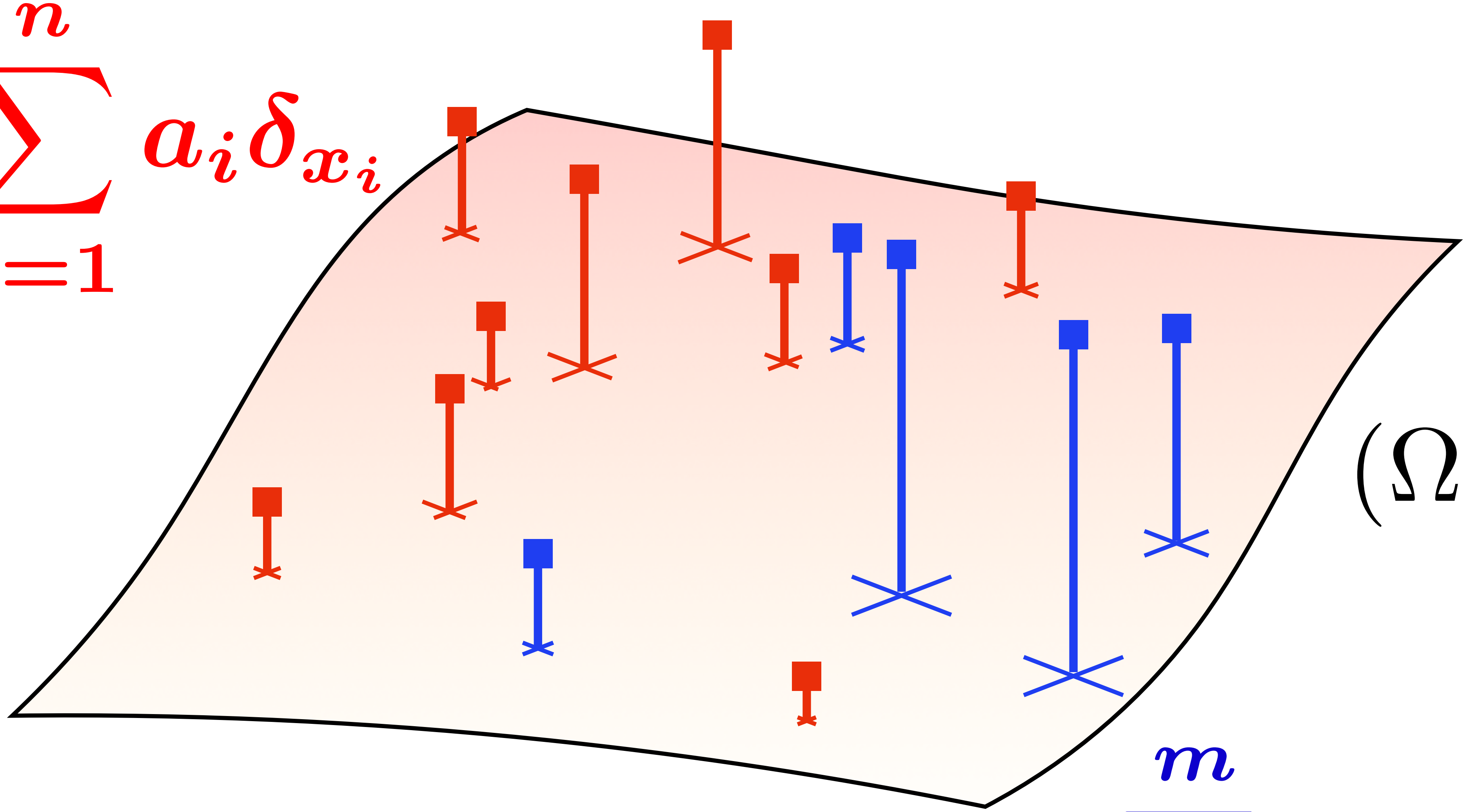
$$(\Omega, \boldsymbol{D})$$

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

Differentiability of W

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

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



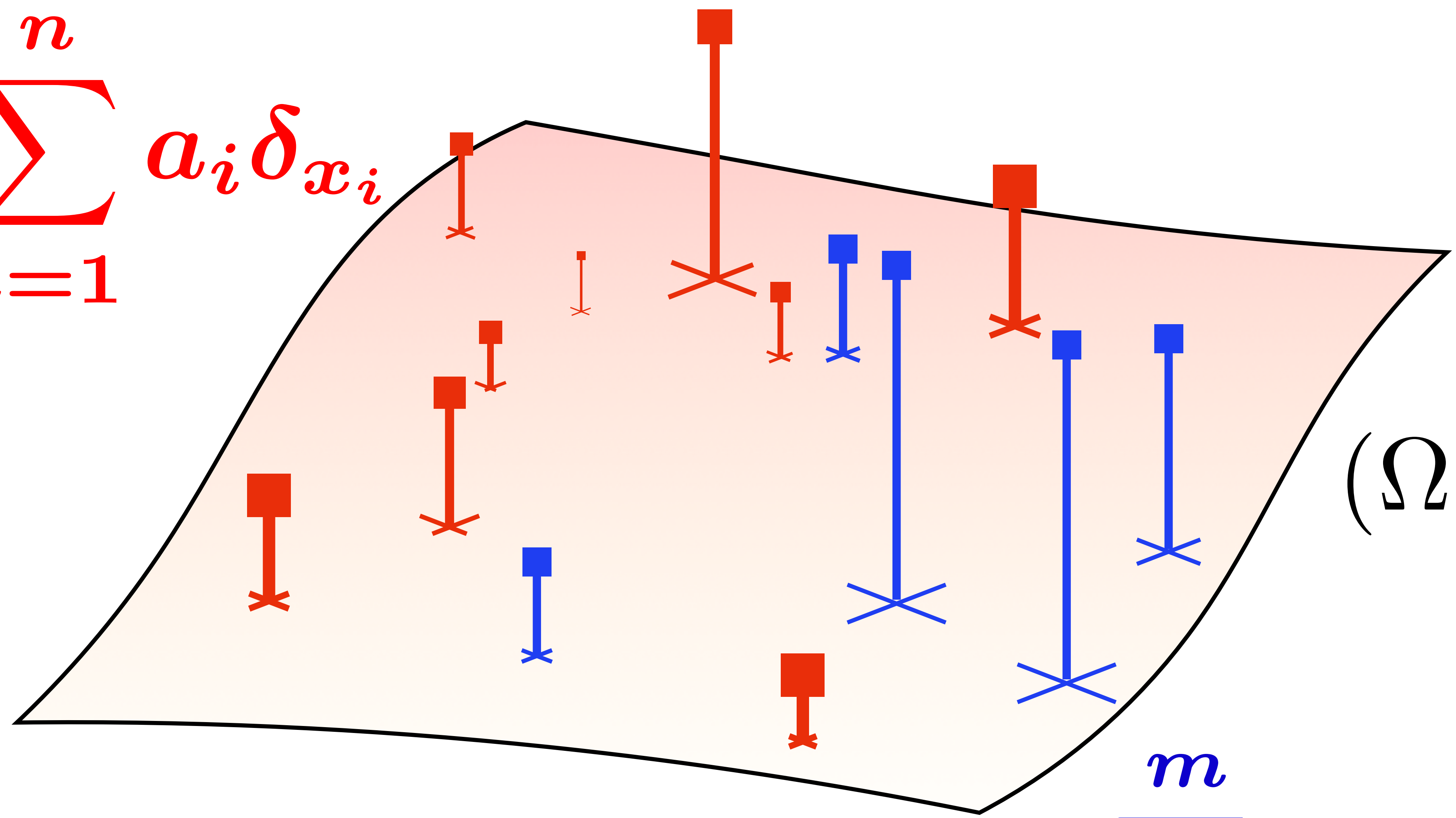
$(\Omega, \boldsymbol{D})$

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

Differentiability of W

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

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



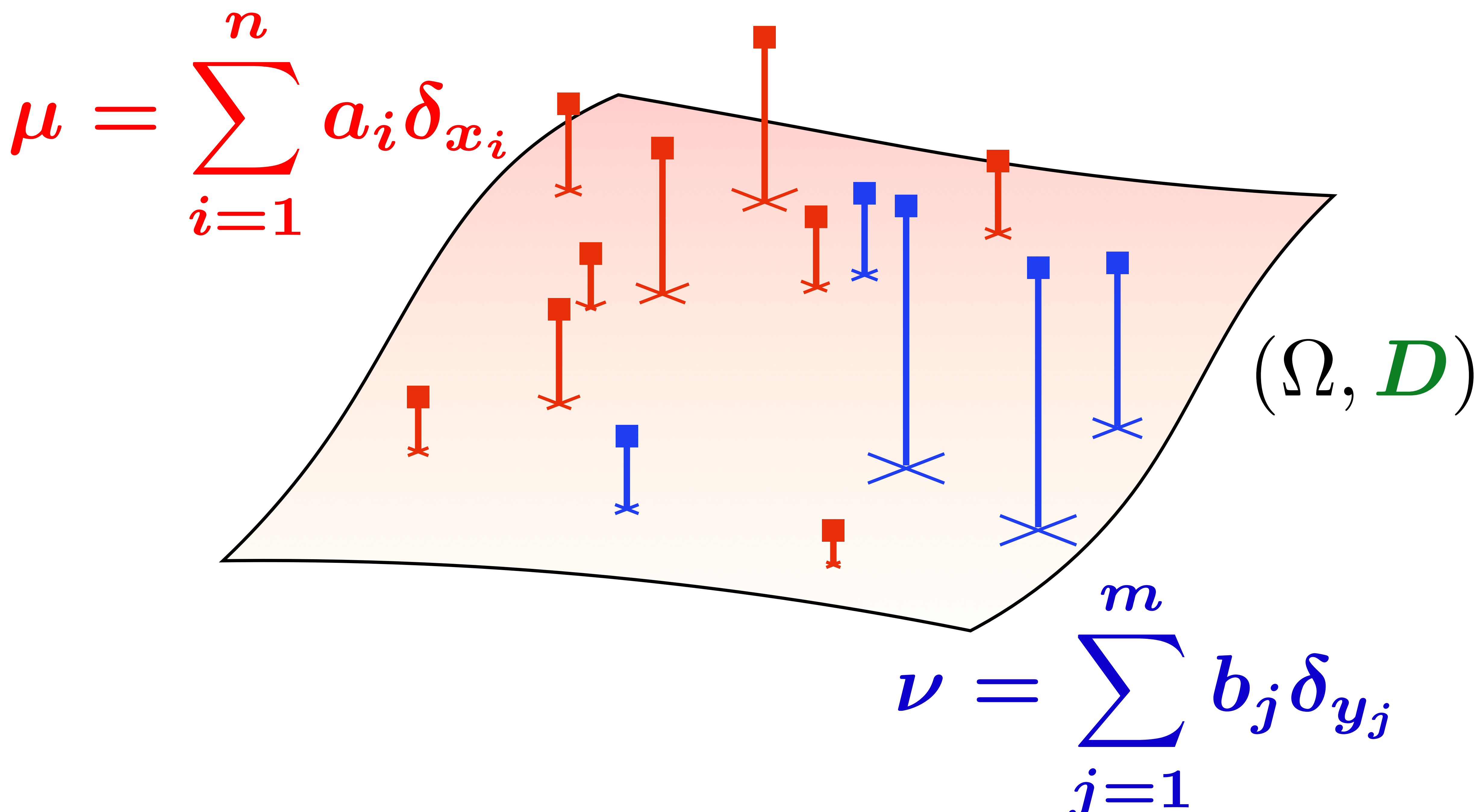
(Ω, D)

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

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

Sinkhorn ⇢ *Differentiability*

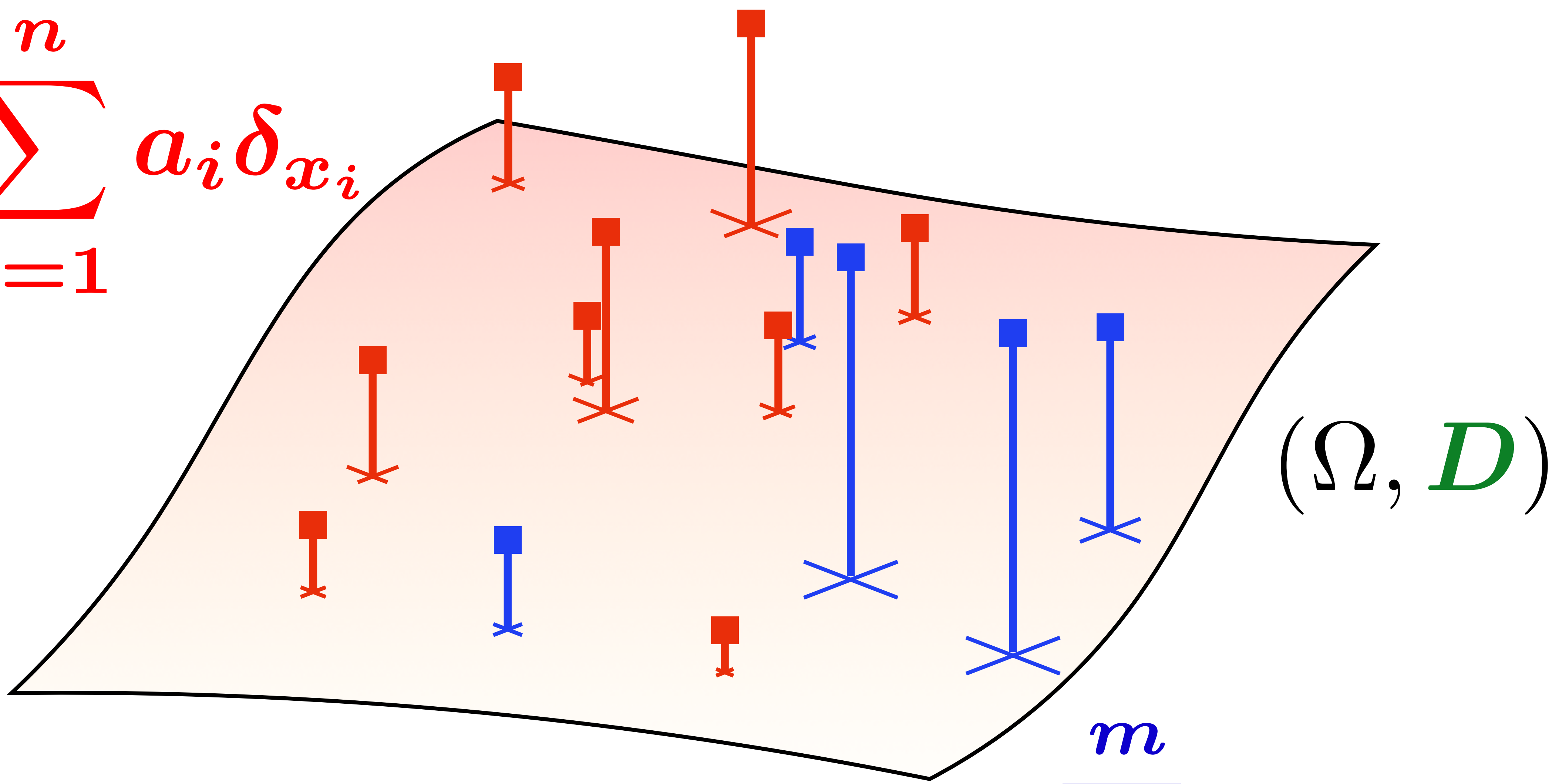
$$W((\mathbf{a}, \mathbf{X} + \mathbf{\Delta X}), (\mathbf{b}, \mathbf{Y})) = W((\mathbf{a}, \mathbf{X}), (\mathbf{b}, \mathbf{Y})) + ??$$



Sinkhorn ⇝ *Differentiability*

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

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



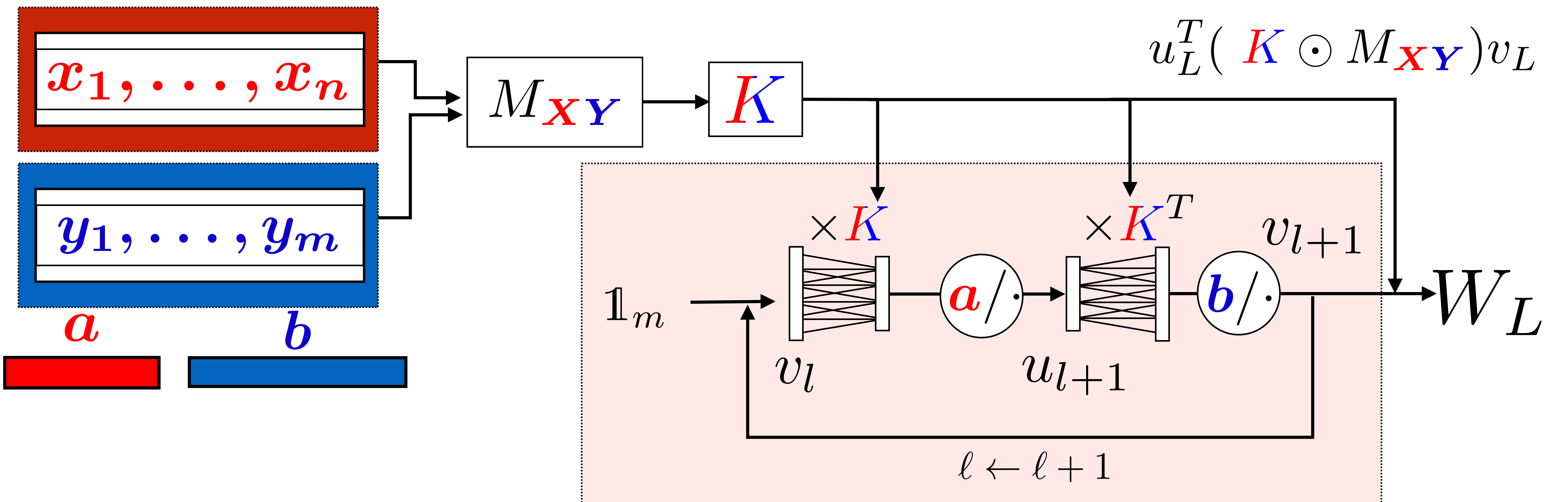
$$\boldsymbol{X} \leftarrow \boldsymbol{X} + \boldsymbol{\Delta X}$$

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

Sinkhorn: A Programmer View

Def. For $L \geq 1$, define

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



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

Sinkhorn: A Programmer View

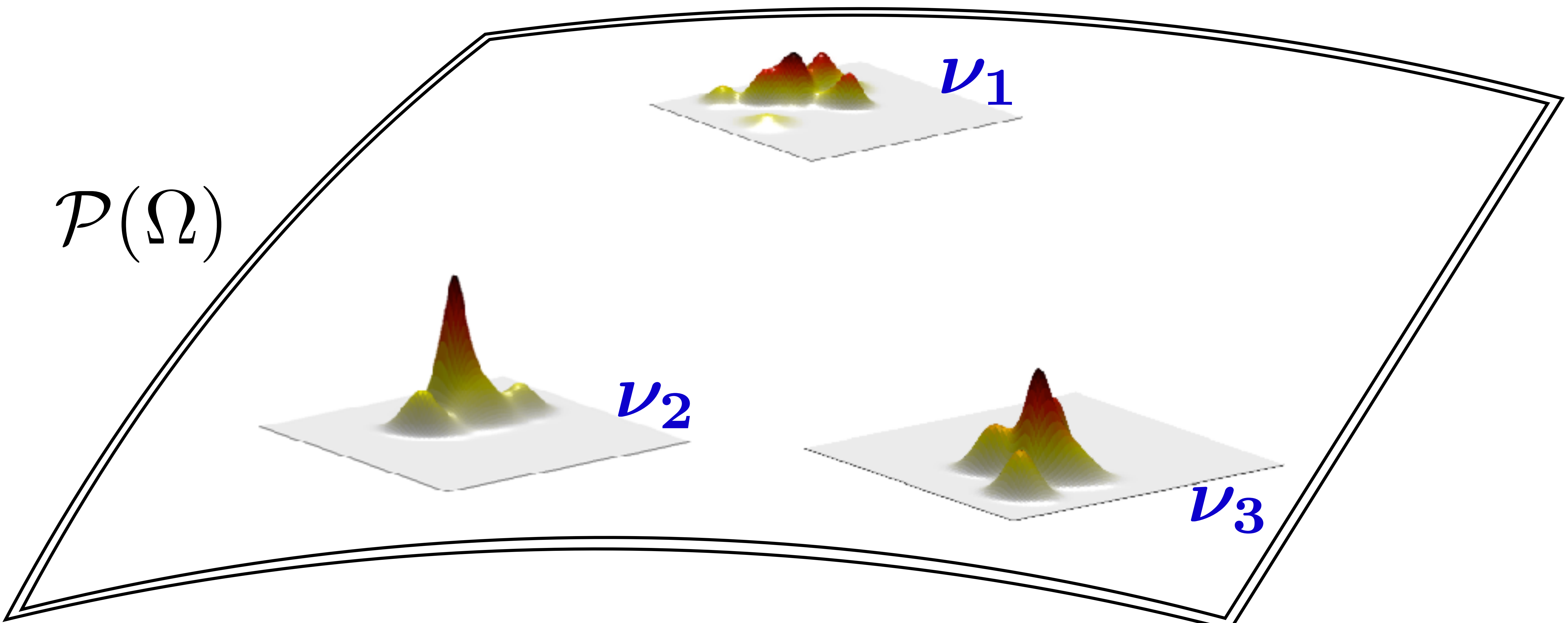
Def. For $L \geq 1$, define

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

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.

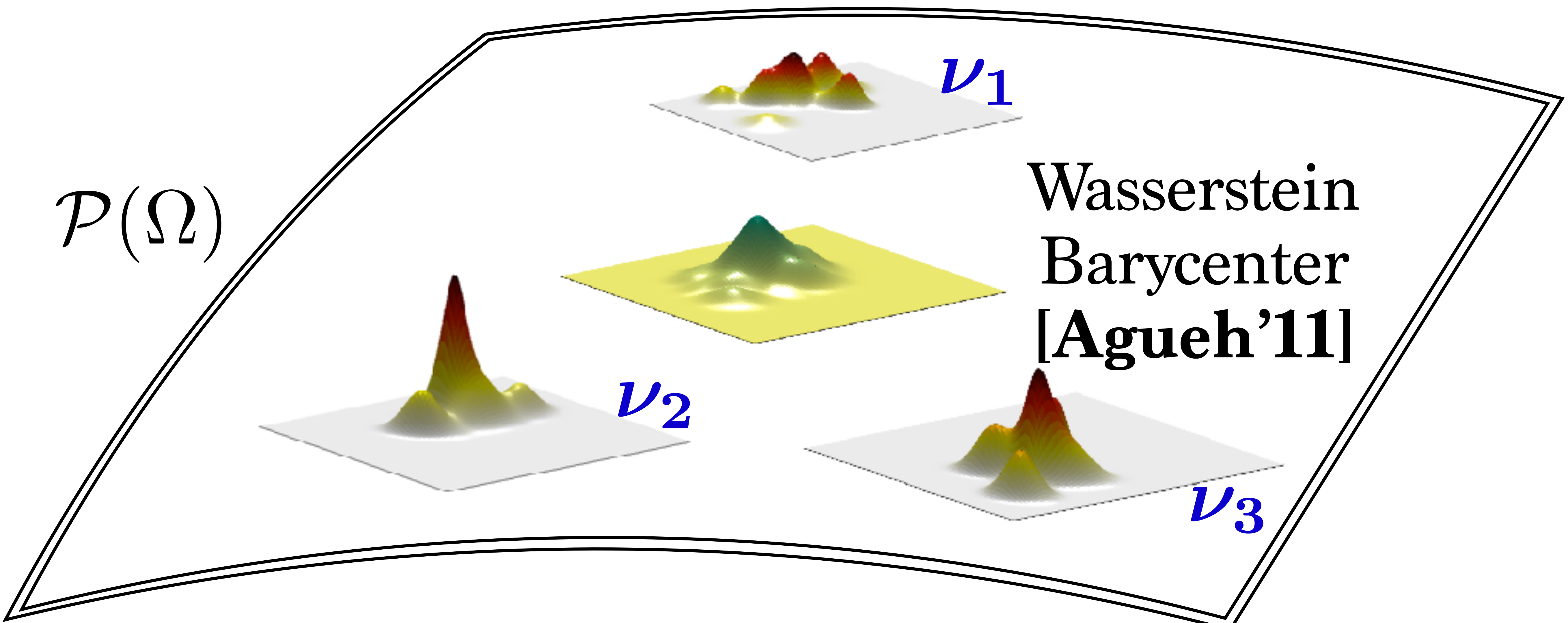
[Hashimoto'16][Bonneel'16][Shalit'16][Flammary'16]

OT: Barycenters



OT: Barycenters

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



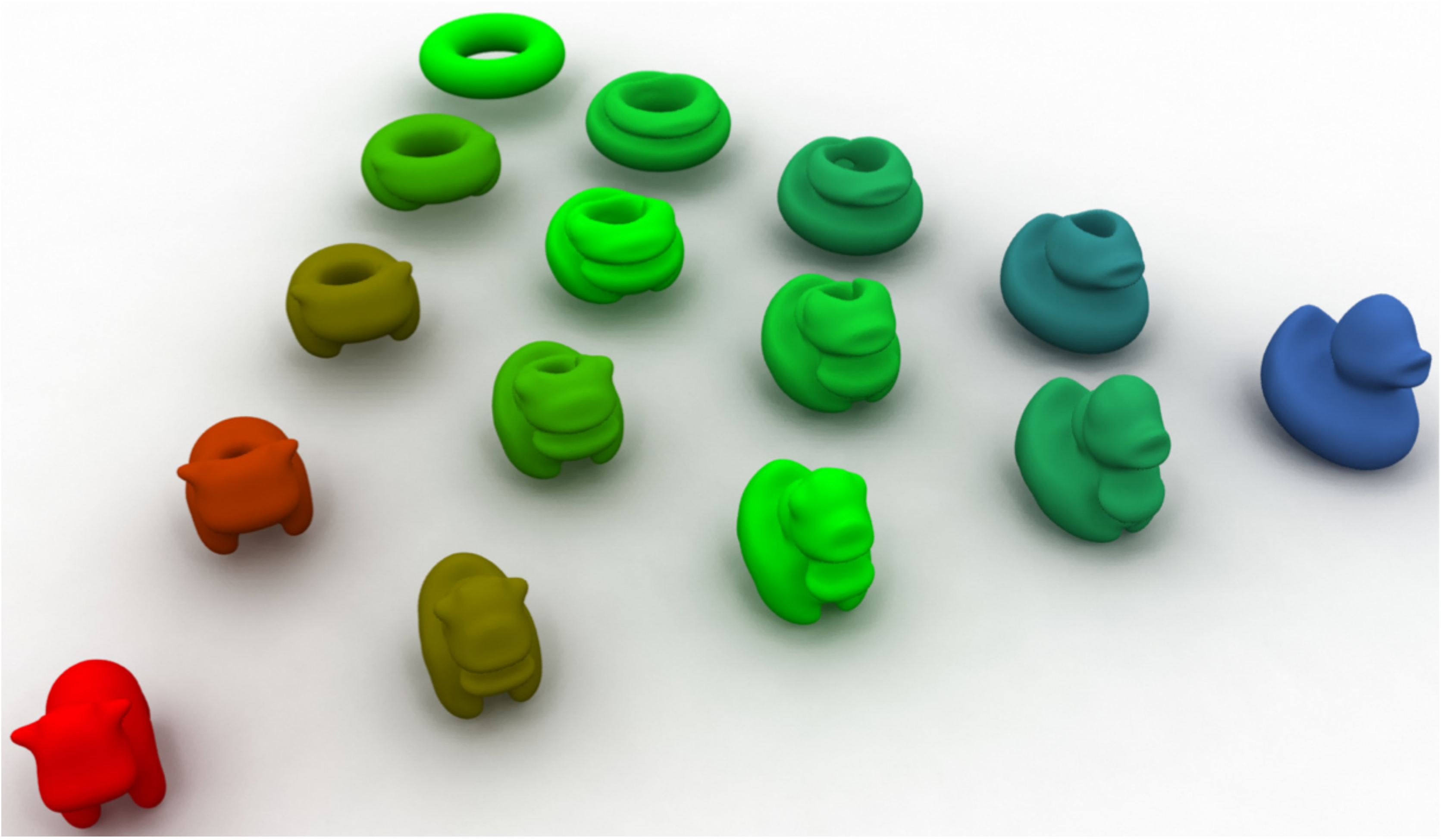
OT: Barycenters

Very different geometry than standard information divergences (*KL*, Euclidean)

[Solomon'15]

OT: Barycenters

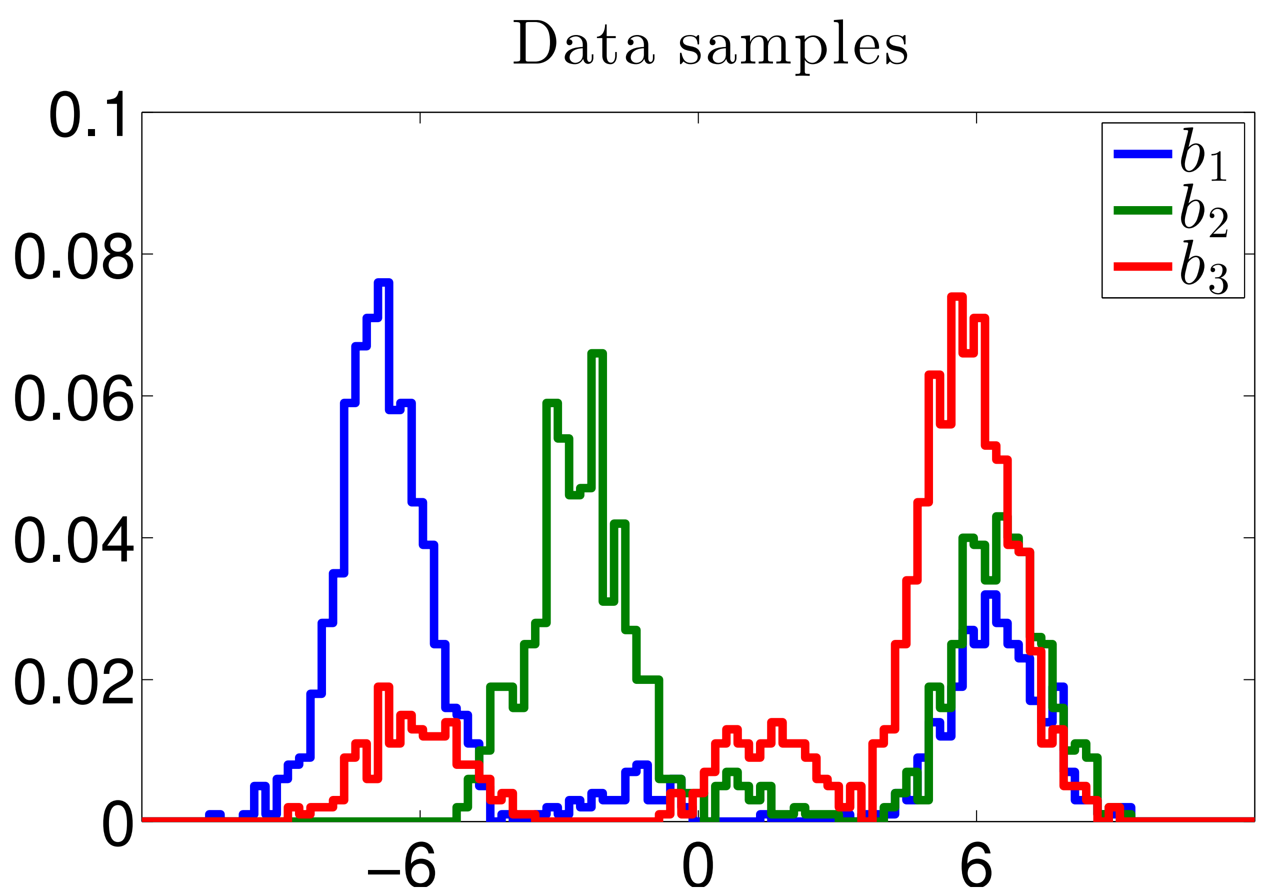
Very different geometry than standard information divergences (*KL*, Euclidean)

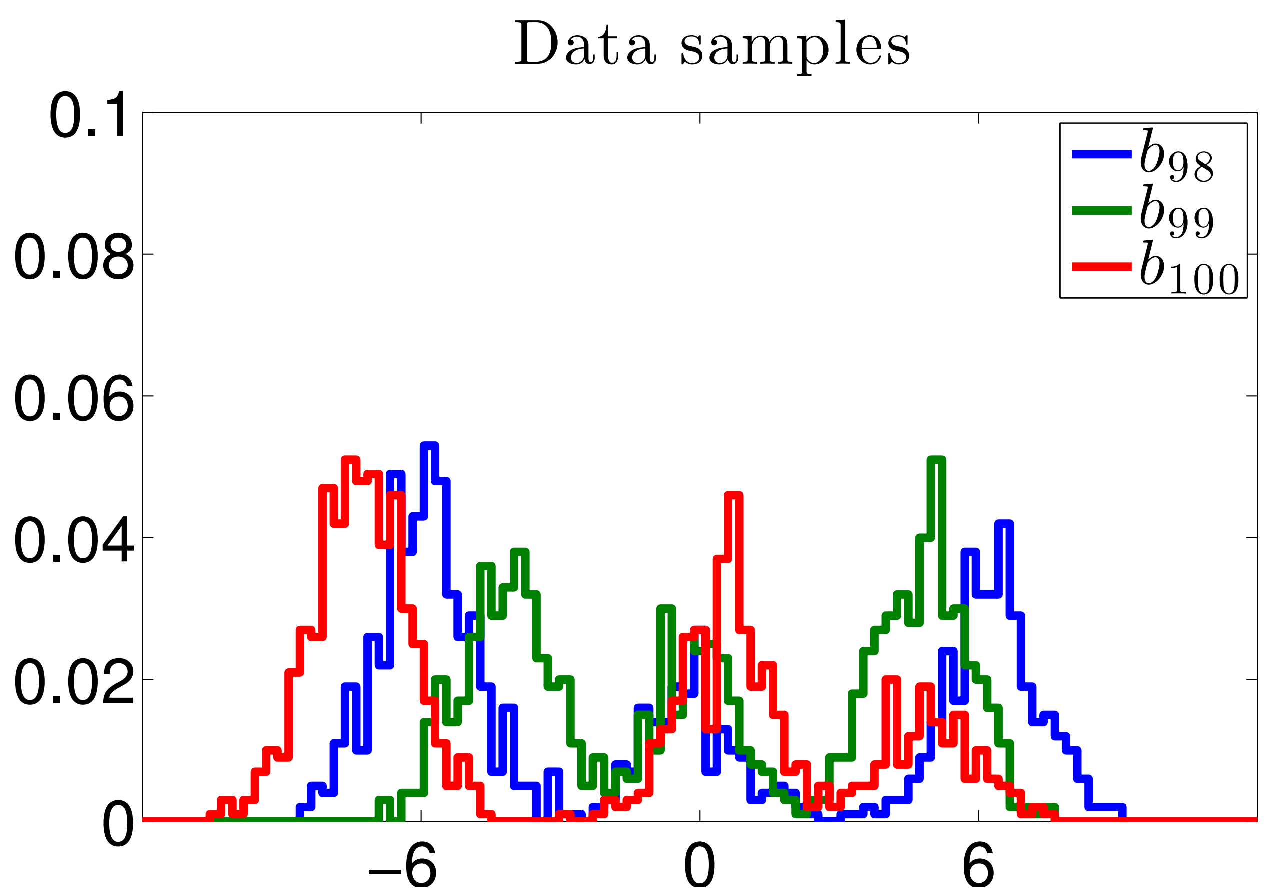


[Solomon'15]

OT: Dictionary Learning

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

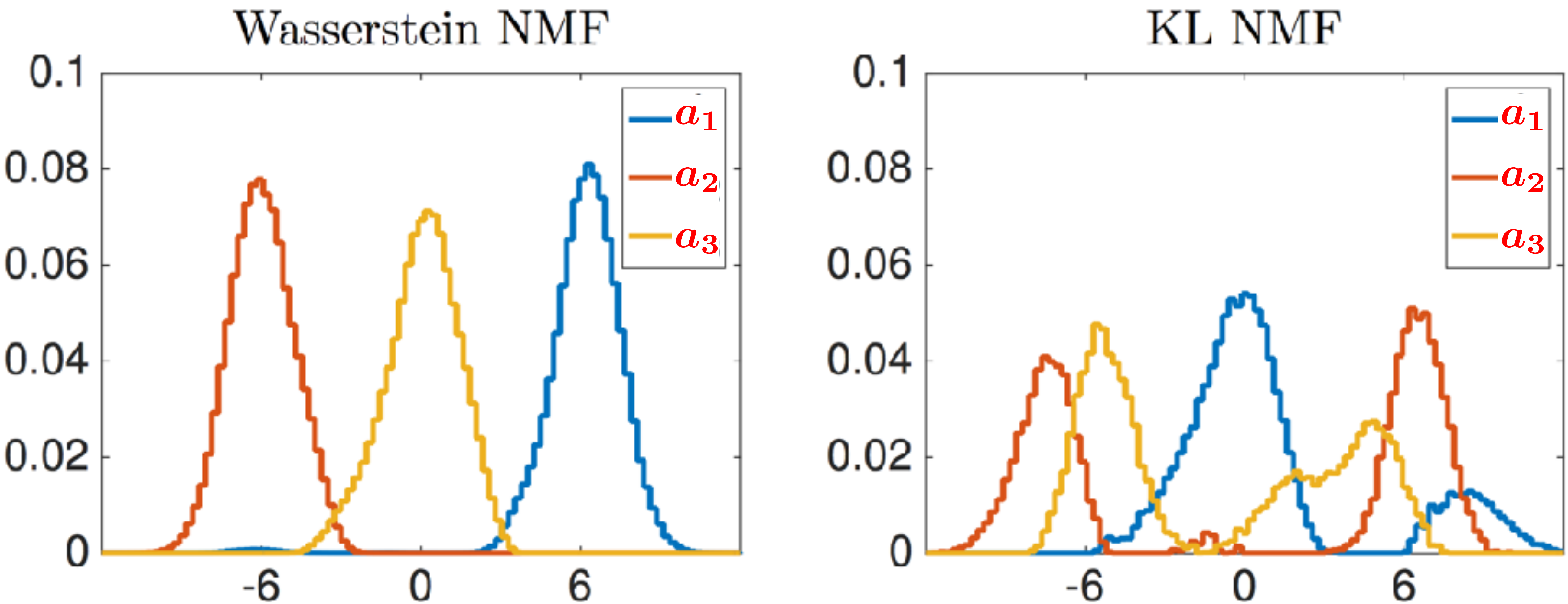




[Sandler'11] [Zen'14] [Rolet'16]

OT: Dictionary Learning

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



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

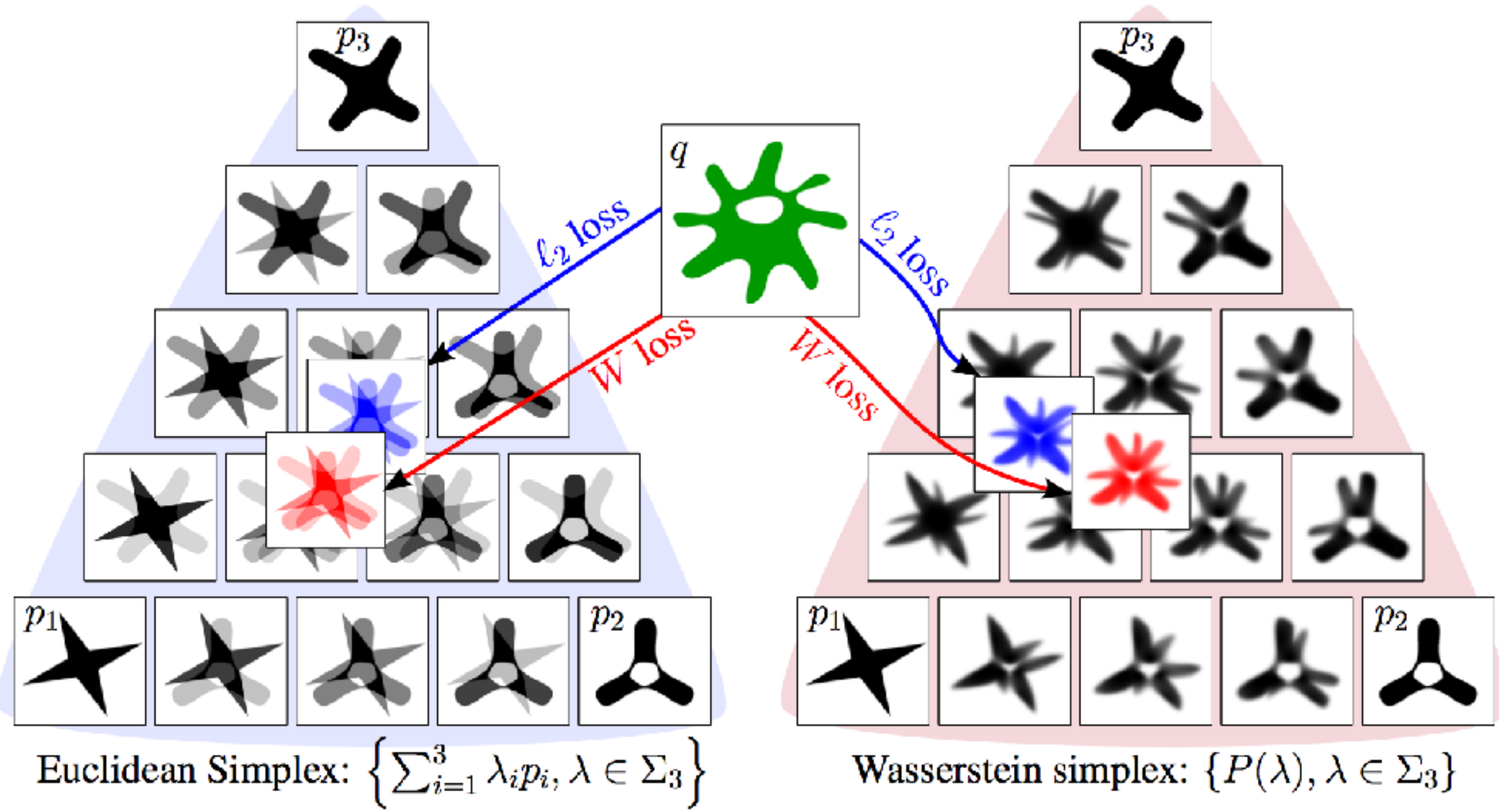
(Word Mover's Distance)

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

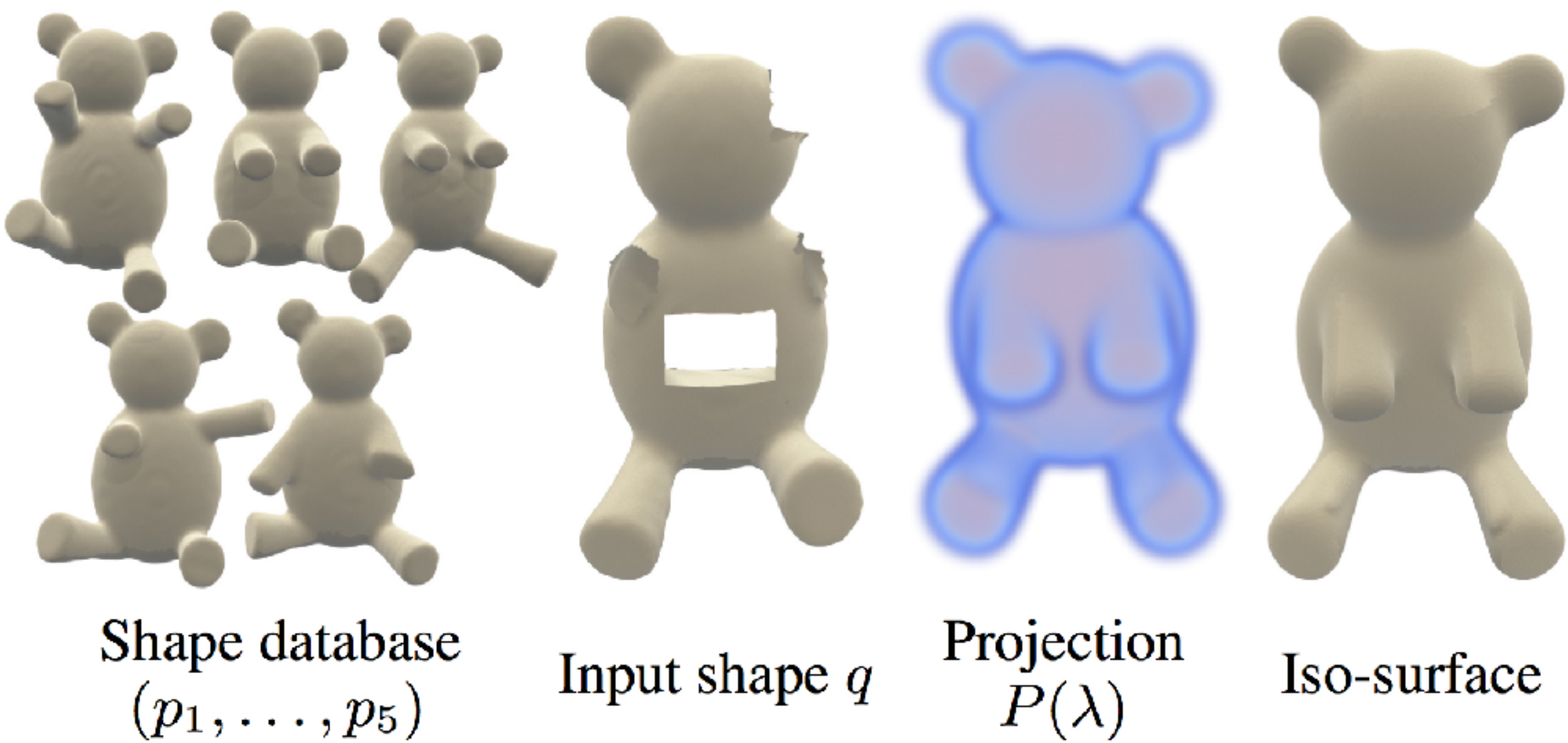
Topic Models

[Rolet'16]

Wasserstein Inverse Problems

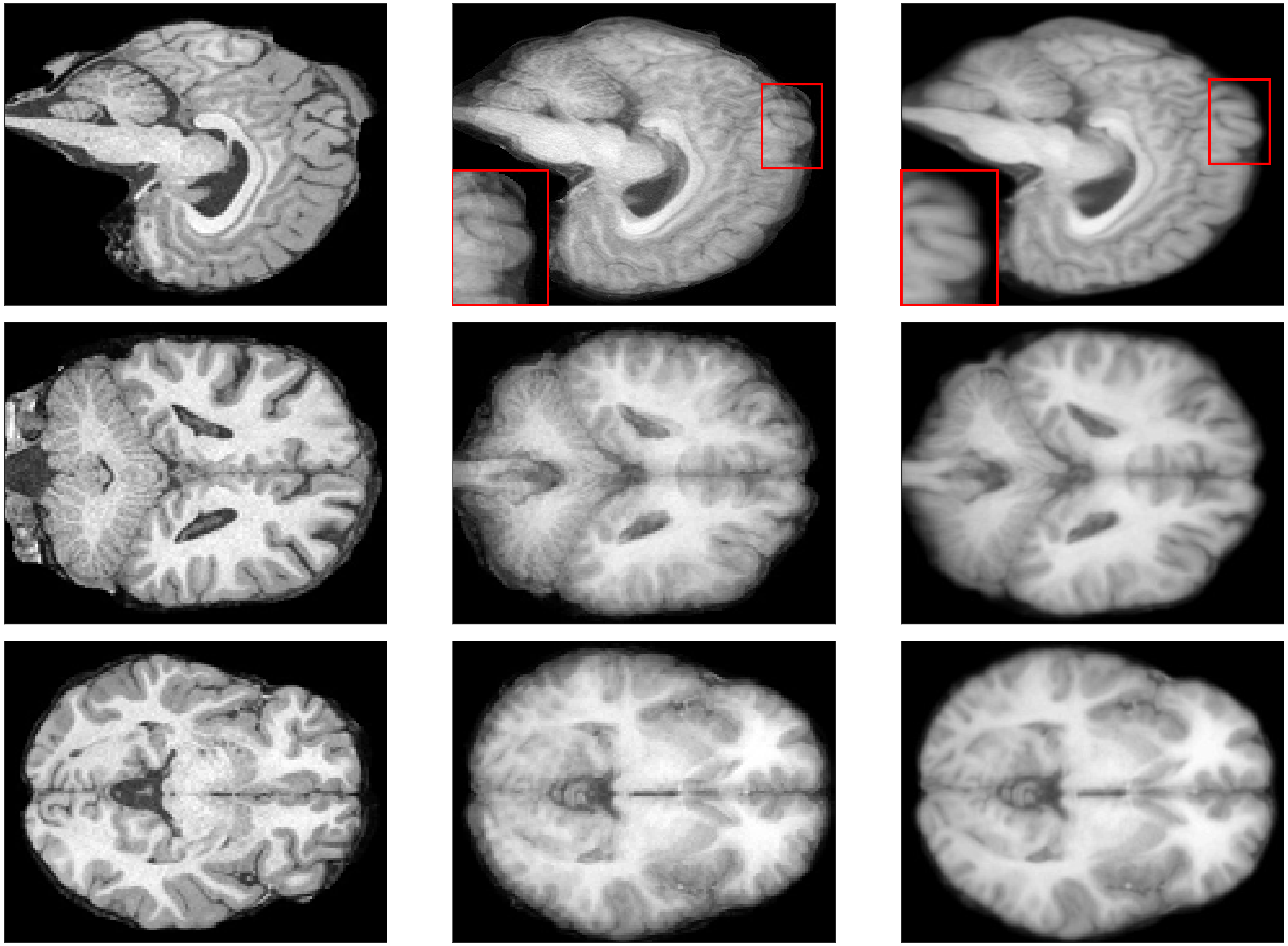


Application: Volume Reconstruction



[Bonneel'16]

Application: Brain Mapping

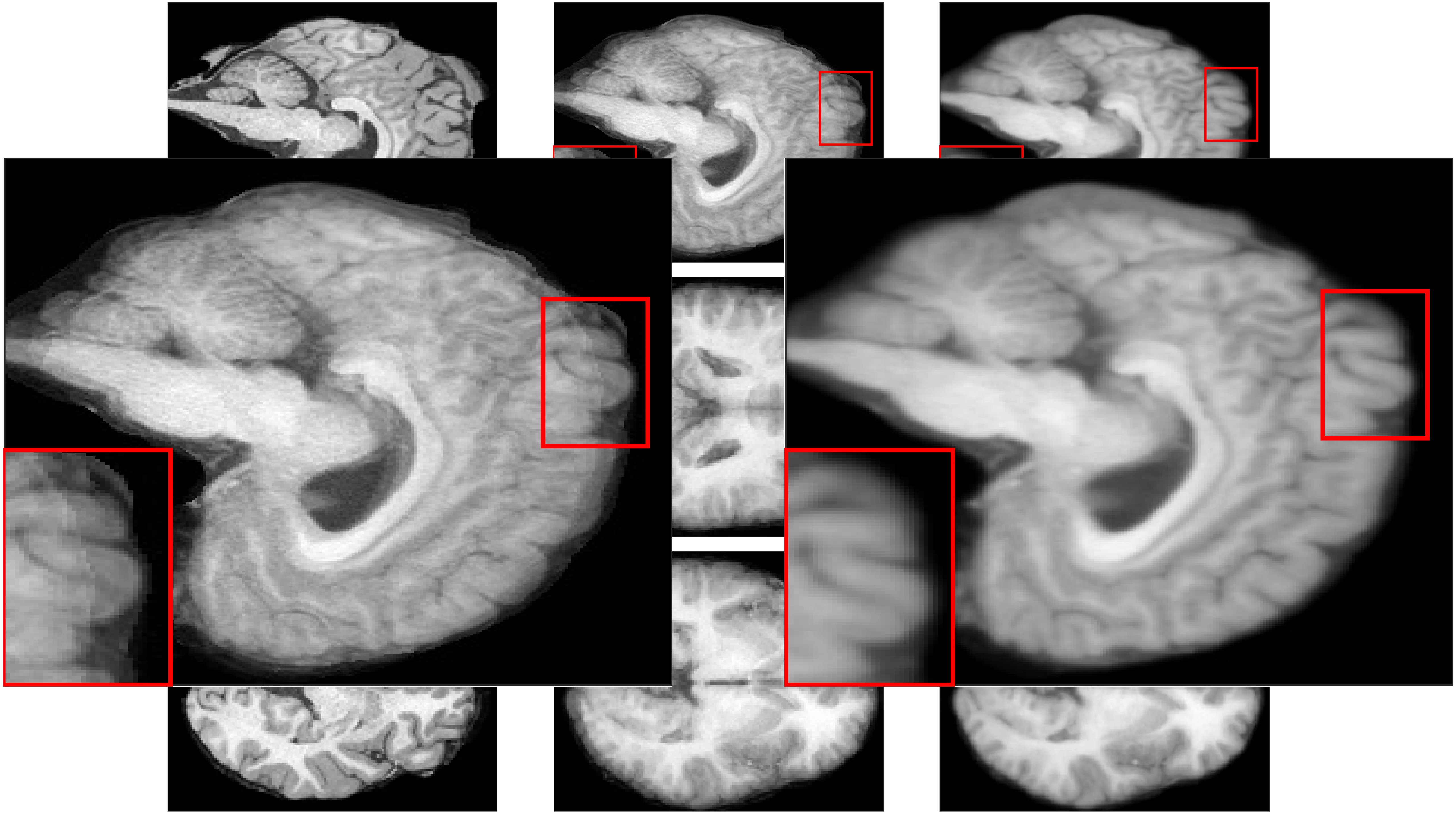


Original

Euclidean projection

Wasserstein projection

Application: Brain Mapping



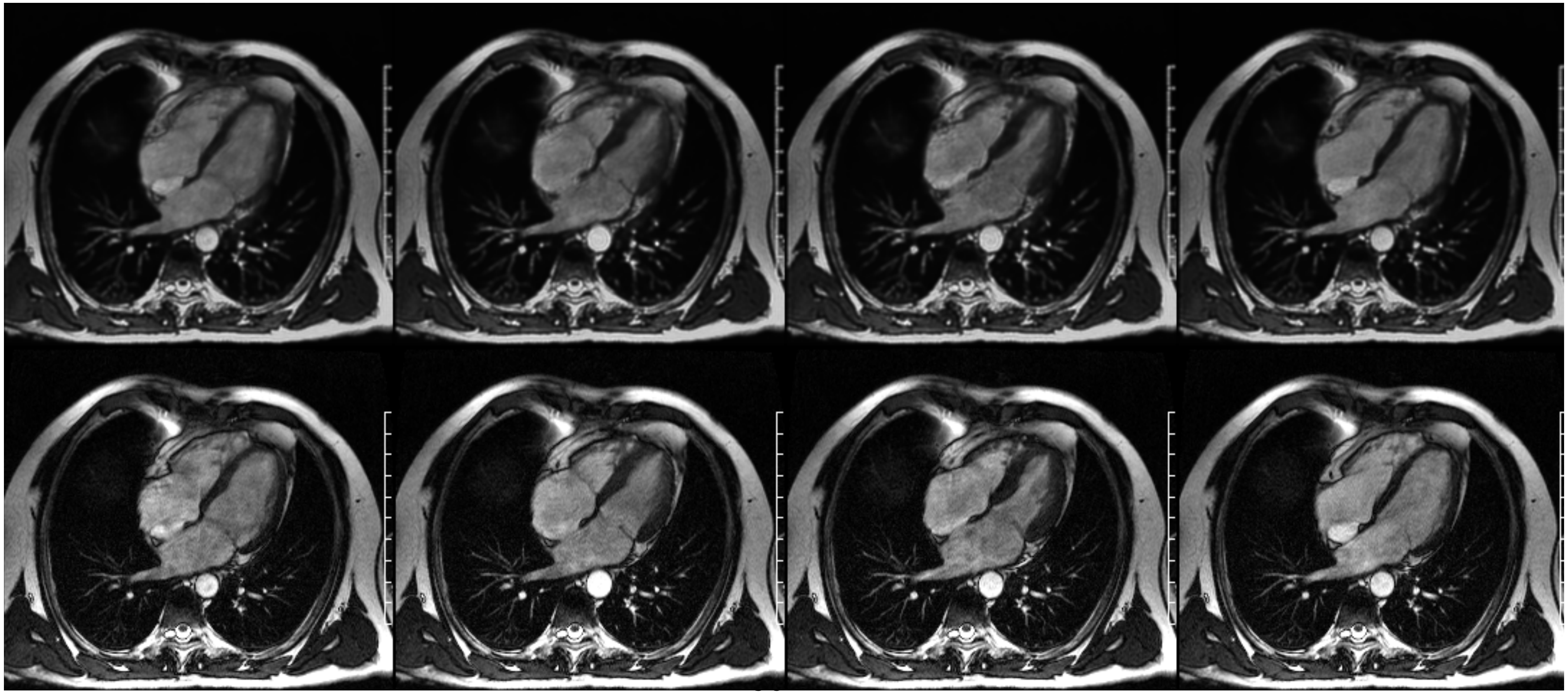
Original

Euclidean projection

Wasserstein projection

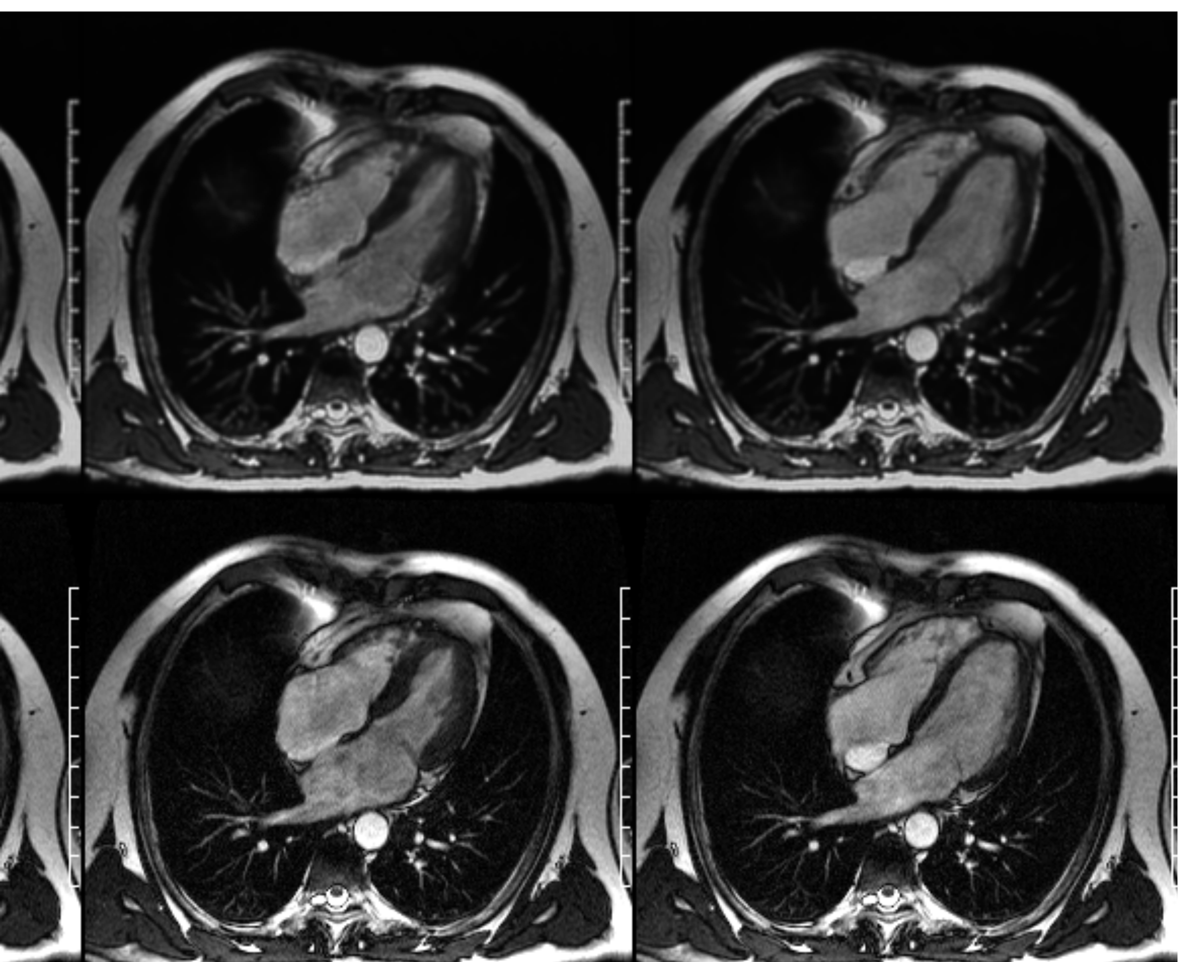
Application: W Dictionary Learning

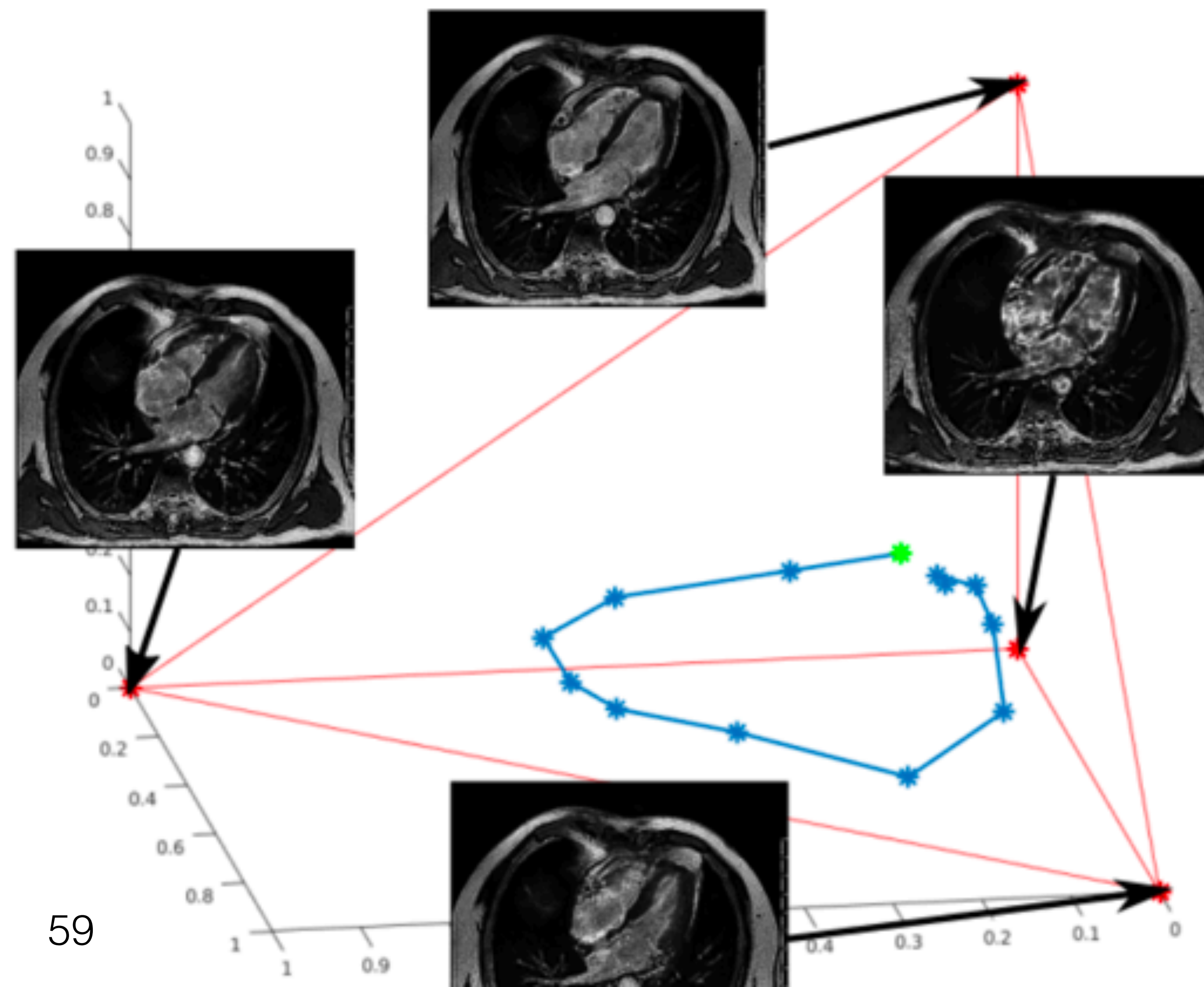
$$\min_{\boldsymbol{A}\in(\Sigma_n)^K\,\boldsymbol{\Lambda}\in(\Sigma_K)^N} \sum_{i=1}^{N} \mathcal{L}\left(\boldsymbol{b_i}, \mathcal{B}_{\boldsymbol{\Lambda}_{\boldsymbol{k}}^{\boldsymbol{i}}}(\boldsymbol{a_k})\right)$$



Application: W Dictionary Learning

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





Learning with a Wasserstein Loss

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

x_i

husky
snow
sled
slope
men

y_i

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

Learning with a Wasserstein 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

Which loss $\mathcal{L}$ could we use?

Learning with a Wasserstein 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

Which loss $\mathcal{L}$ could we use?

Learning with a Wasserstein Loss

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

$$\mathcal{L}(\boldsymbol{a}, \boldsymbol{b}) = \min_{\boldsymbol{P}\in\mathbb{R}^{nm}} \langle \boldsymbol{P}, M \rangle + \varepsilon \mathrm{KL}(\boldsymbol{P}\mathbf{1}, \boldsymbol{a}) + \varepsilon \mathrm{KL}(\boldsymbol{P}^T\mathbf{1}, \boldsymbol{b}) - \gamma E(\boldsymbol{P})$$

1. Generalizes Word Mover's to label clouds
2. Sinkhorn algorithm can be generalized

[Frogner'15] [Chizat'15][Chizat'16]

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.

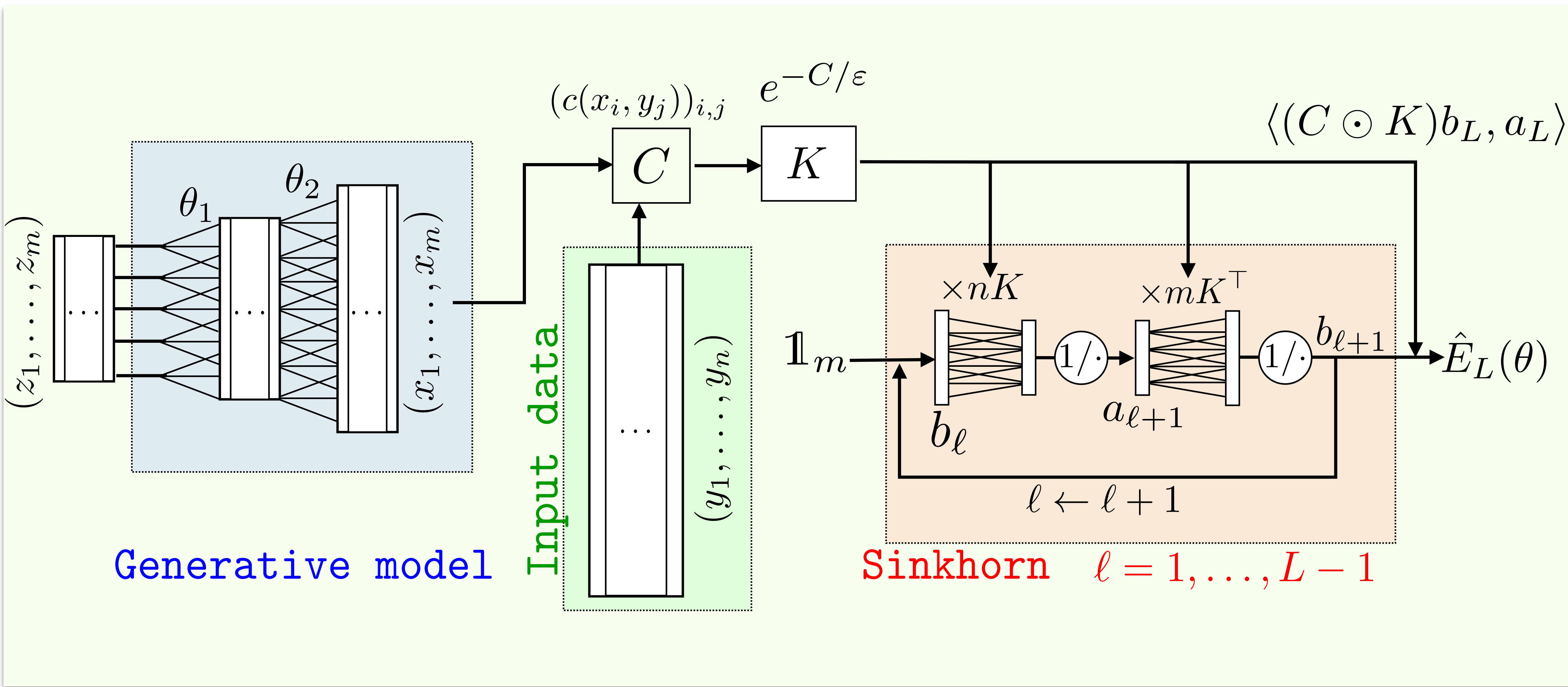
[Arjovsky'17] *(WGAN)* uses a NN to approximate dual solutions and recover gradient w.r.t. parameter

[Bernton'17] reject mechanism *W*(sample, data)

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

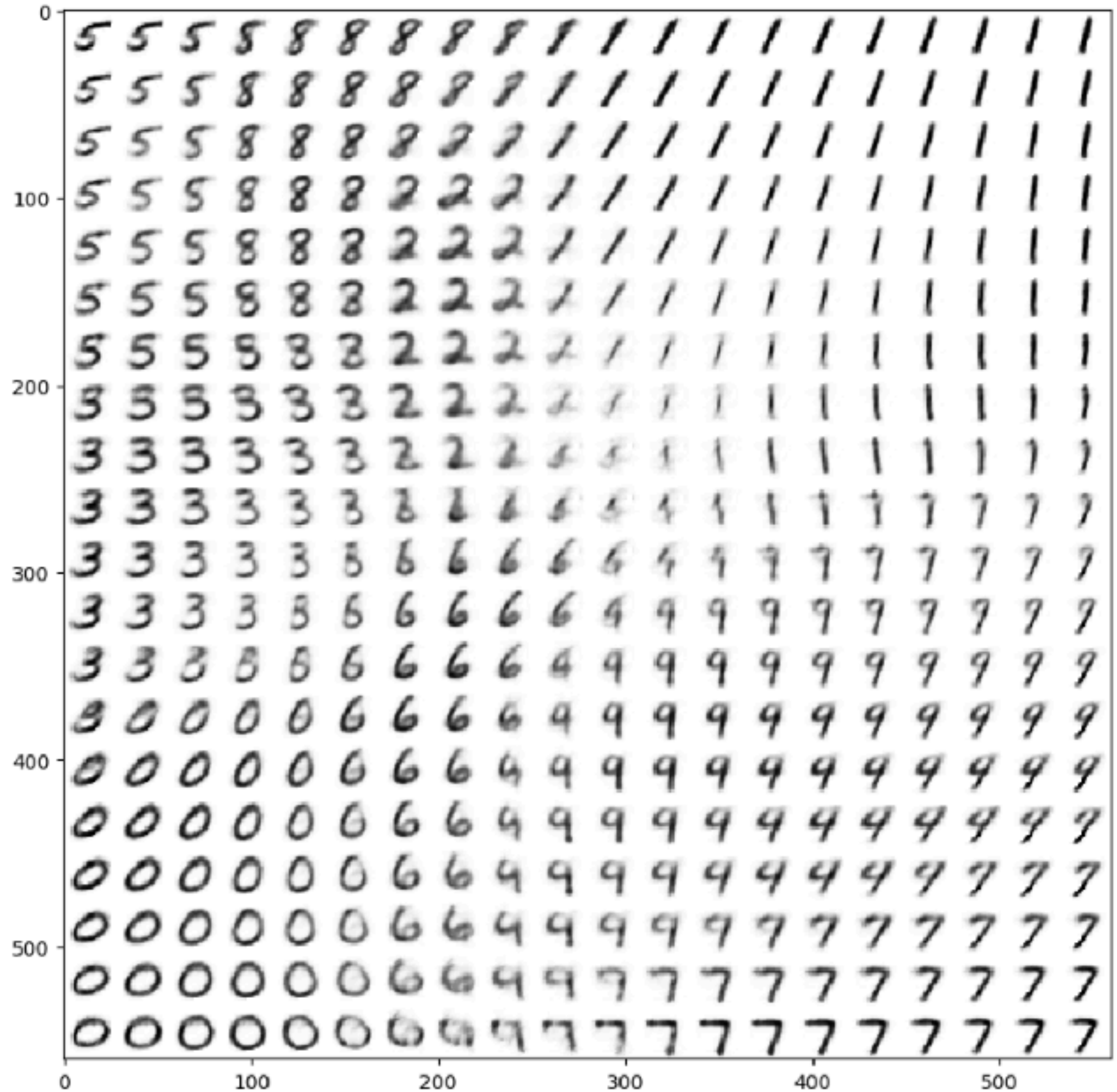
Proposal: Autodiff OT using Sinkhorn

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

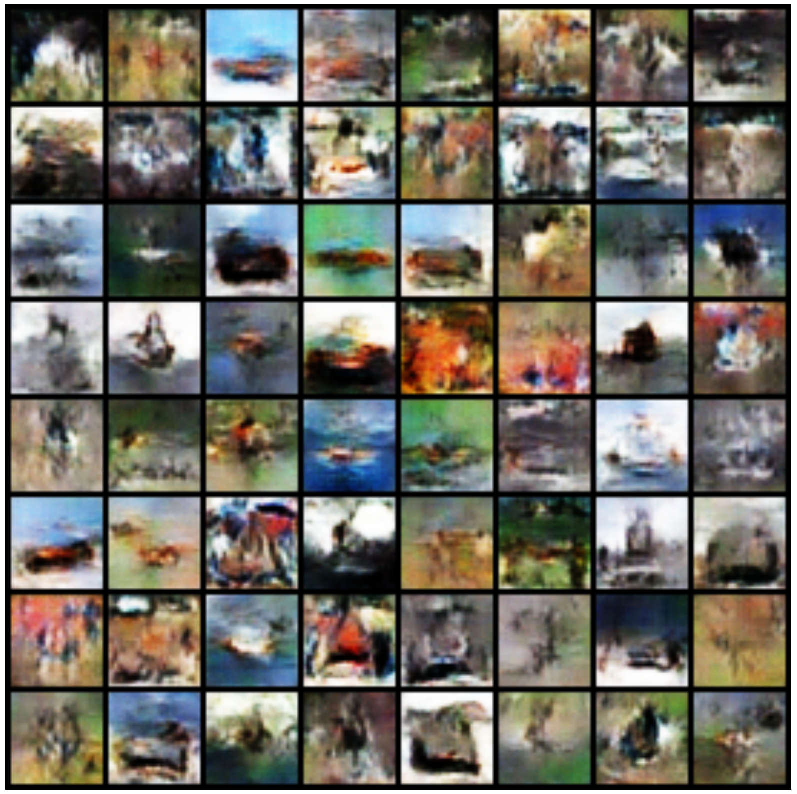


[GPC'17]

Example: MNIST, Learning f_{θ}

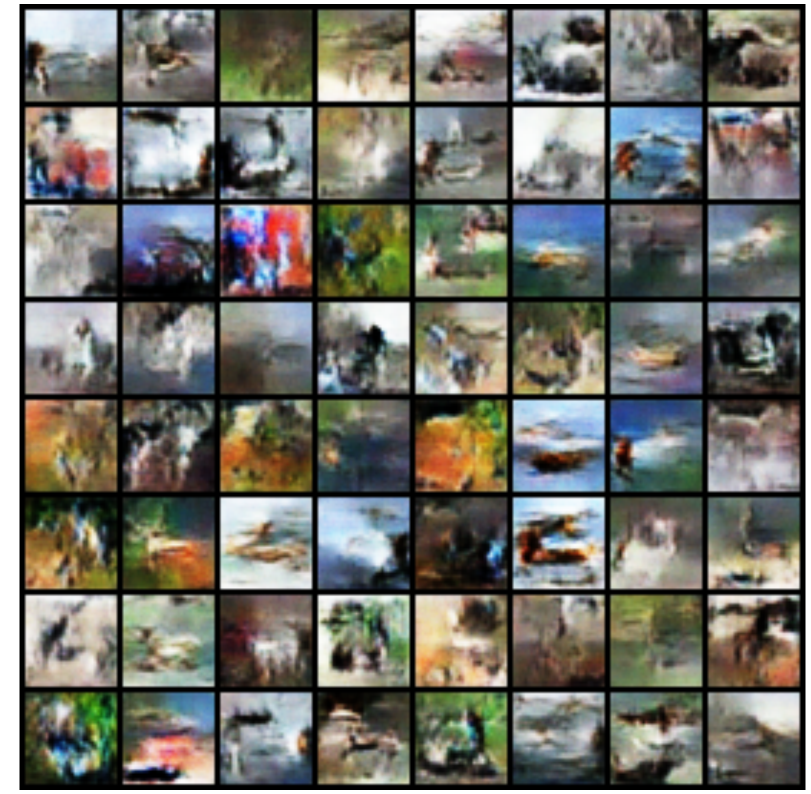


Example: Generation of Images



MMD-GAN

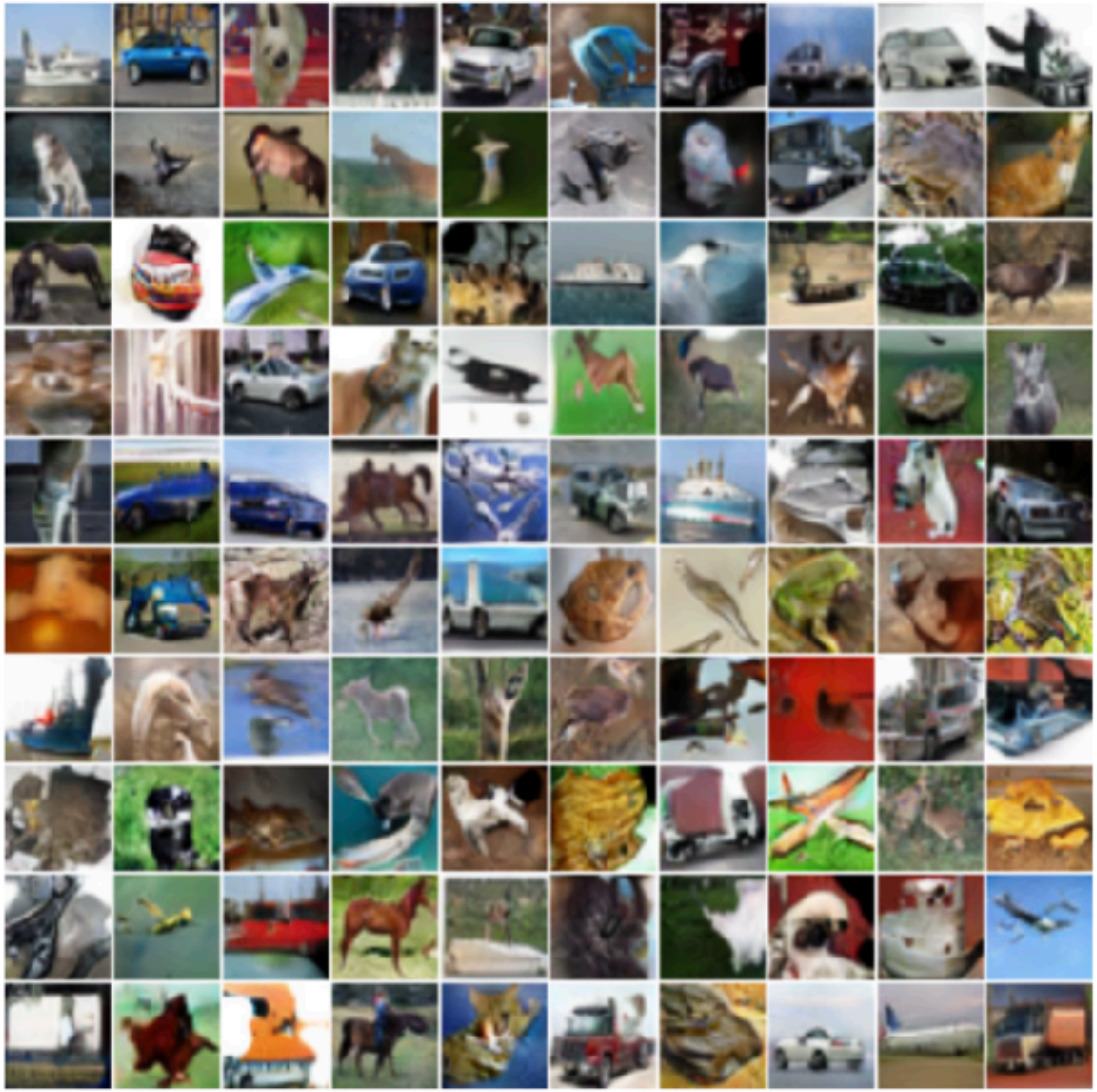
$\tau = 1000$



$\tau=10$

- Learning with CIFAR-10 images
- In these examples the cost function is also learned adversarially, as a NN mapping onto feature vectors.

Example: Generation of Images



arxiv.org/pdf/1710.05488

Example: Generation of Images

arxiv.org/pdf/1710.05488

Concluding Remarks

- *Regularized* OT is much faster than OT.
- *Regularized* OT can interpolate between W and the *MMD* / *Energy distance* metrics.
- The solution of *regularized OT* is *"auto-differentiable"*.
- **Many open problems remain!**