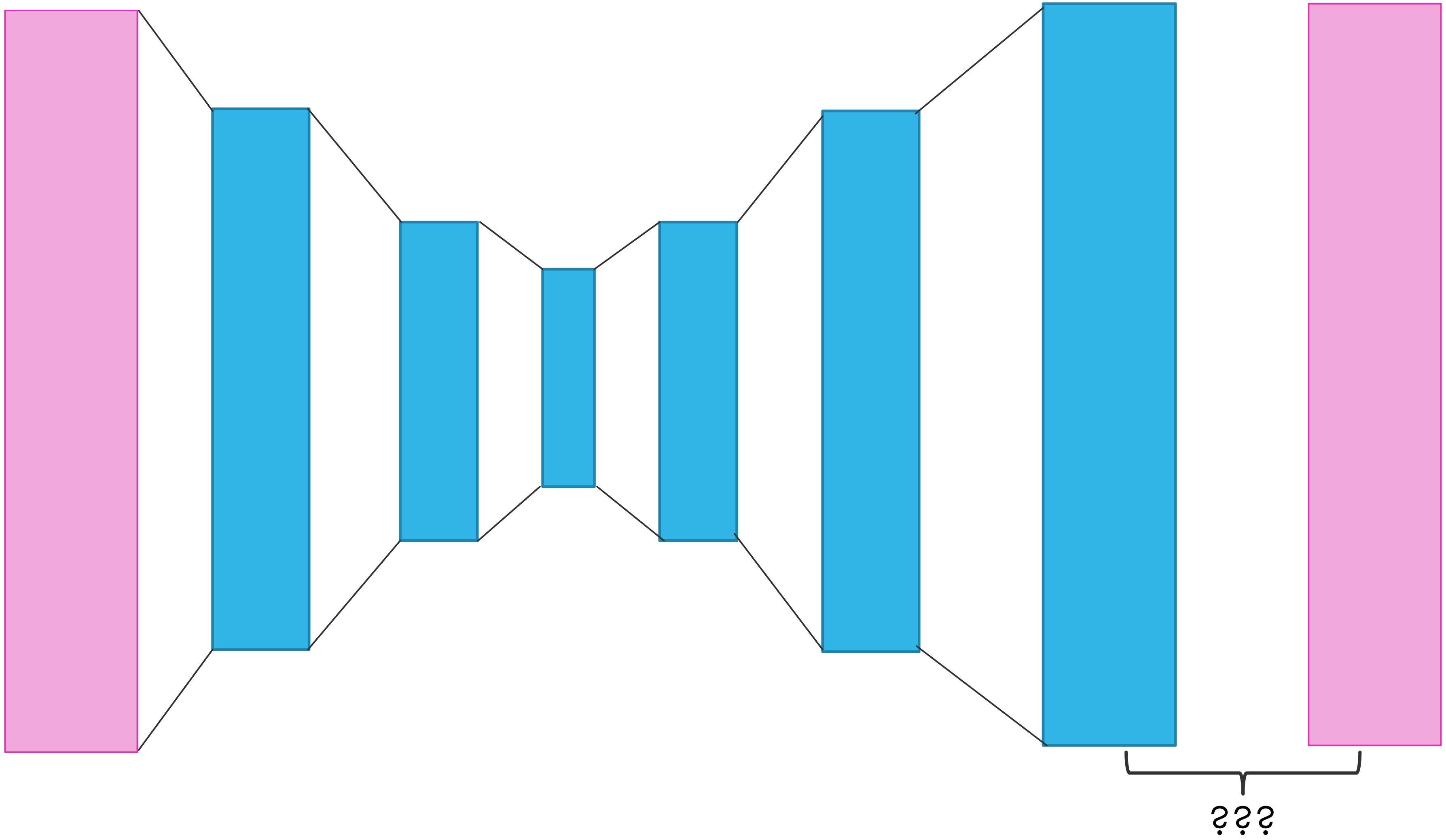
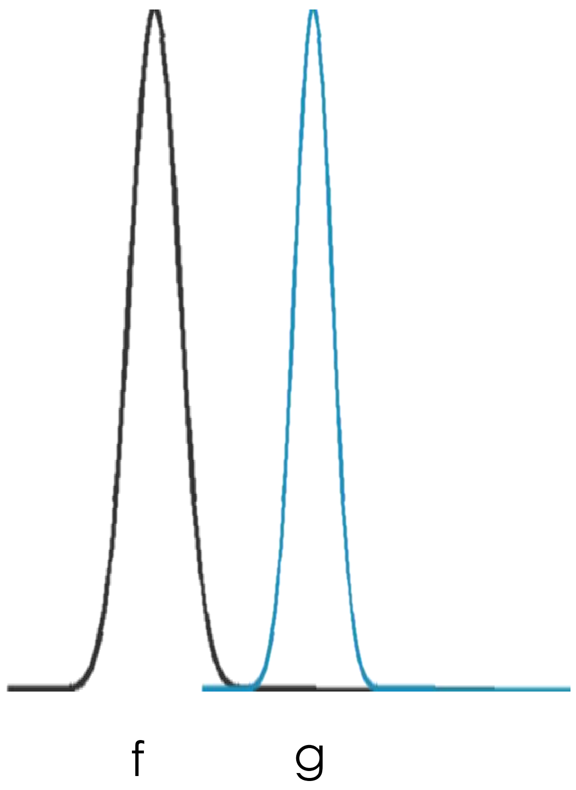
Optimal Transport Principles for Computer Graphics and Machine Learning

Nicolas Bonneel

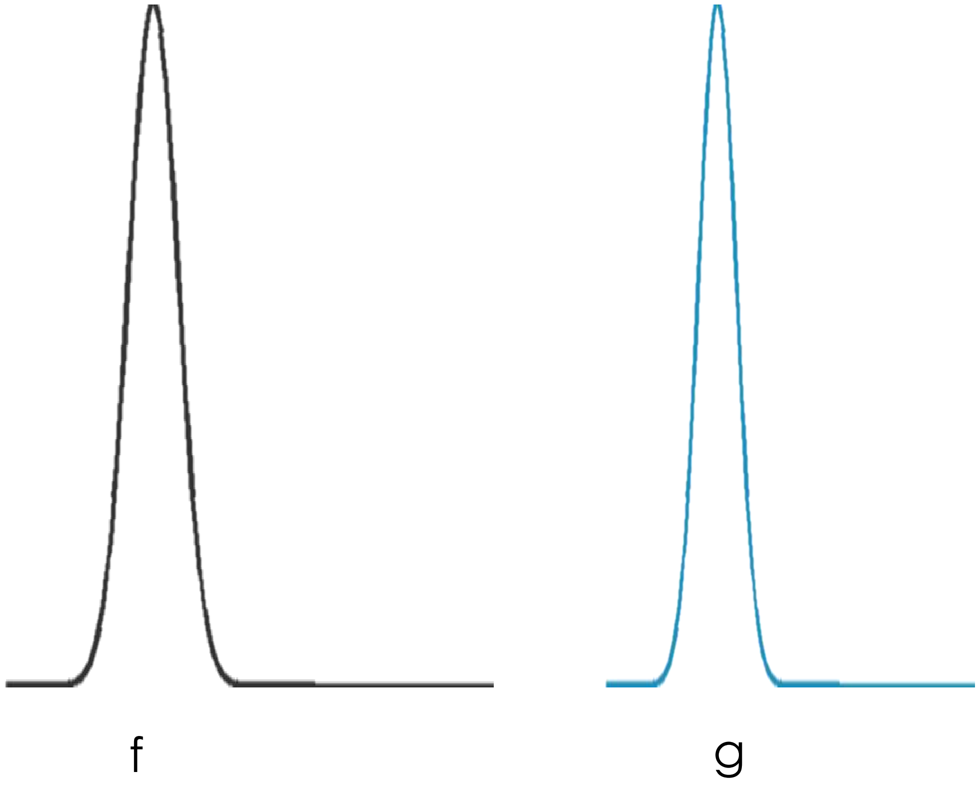
Motivation in neural networks



How to compare functions ?



How to compare functions ?



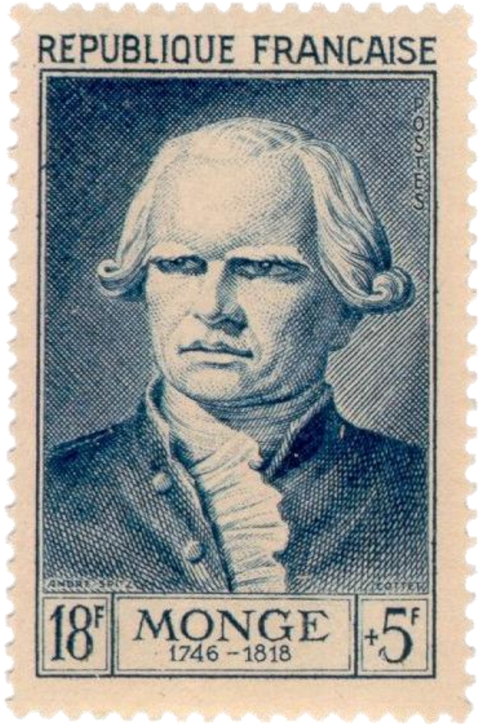
Mémoire sur la théorie des déblais et des remblais (1781)

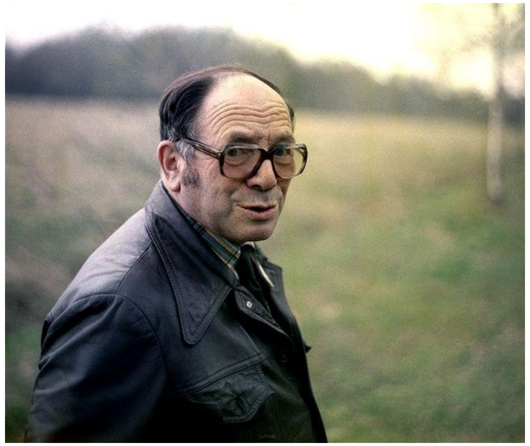
MÉMOIRE
SUR LA
THÉORIE DES DÉBLAIS ET DES REMBLAIS.

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.

Le prix du tranſport d'une molécule étant, toutes choſes d'ailleurs égales, proportionnel à ſon poids & à l'eſpace qu'on lui fait parcourir, & par conſéquent le prix du tranſport total devant être proportionnel à la ſomme des produits des molécules multipliées chacune par l'eſpace parcouru, il s'enſuit que le déblai & le remblai étant donnés de figure & de poſition, il n'eſt pas indifférent que telle molécule du déblai ſoit tranſportée dans tel ou tel autre endroit du remblai, mais qu'il y a une certaine diſtribution à faire des molécules du premier dans le ſecond, d'après laquelle la ſomme de ces produits ſera la moindre poſſible, & le prix du tranſport total ſera un *minimum*.





Leonid Kantorovich

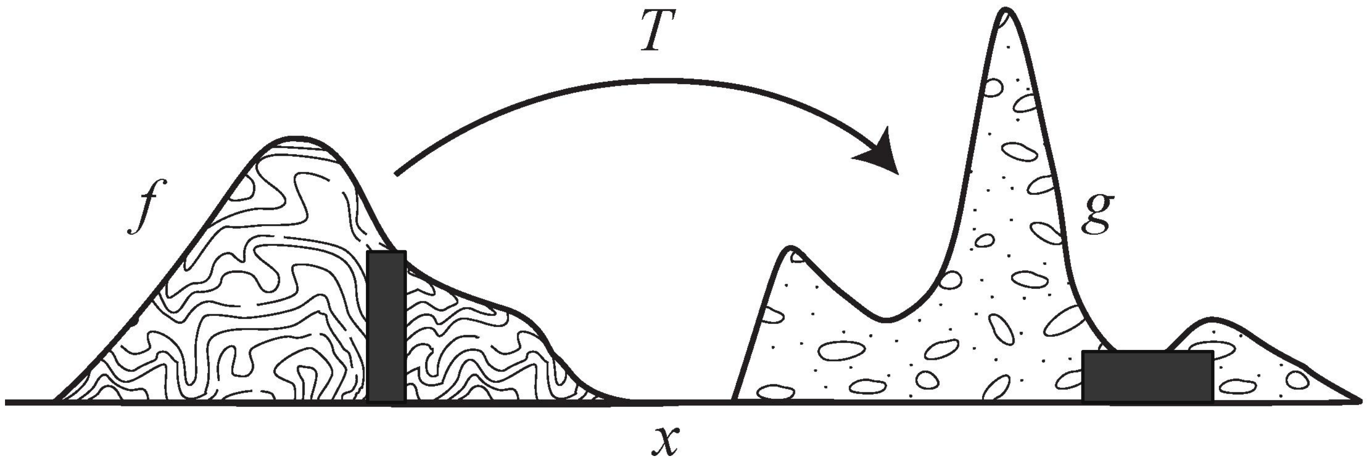
- Nobel prize in economy in 1975, for his "contribution to the theory of resources allocation"

Monge formulation

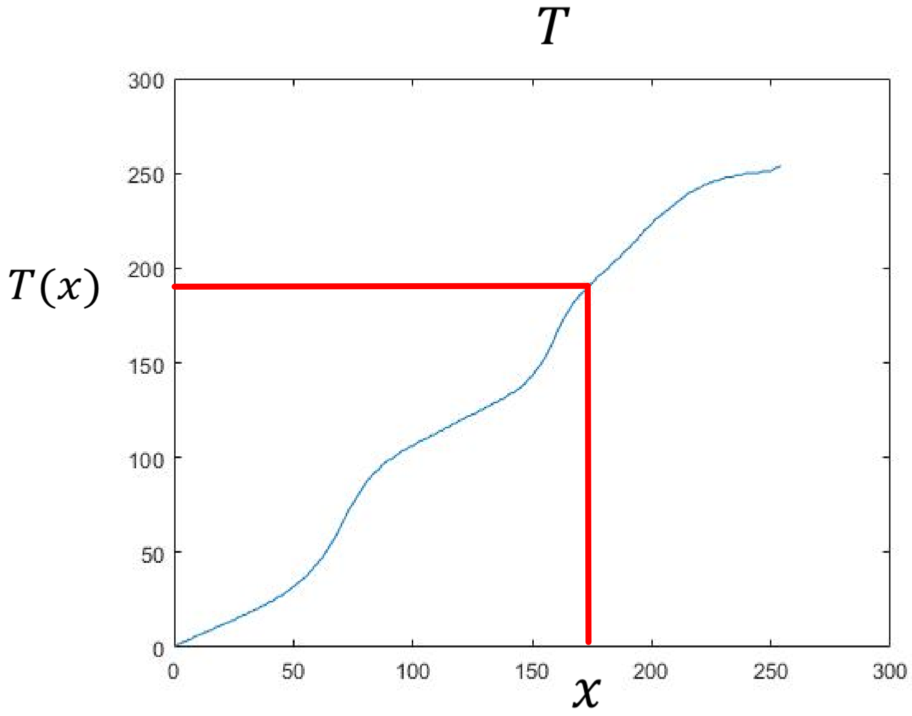
$$W(f,g) = \min_{T} \int_{X} c\left(x,\, T(x)\right) f(x)\, \mathrm{dx}$$

$$\text{s.t.} \qquad f(x) = g(T(x))\, |\det J_T(x)|$$

Monge used $c(x,y) = |x - y|$



Same total mass



"find a good warping between f and g with the change of variable formula"

Kantorovich formulation

Cost

$m_{i\to j}$ particles will move from i to j

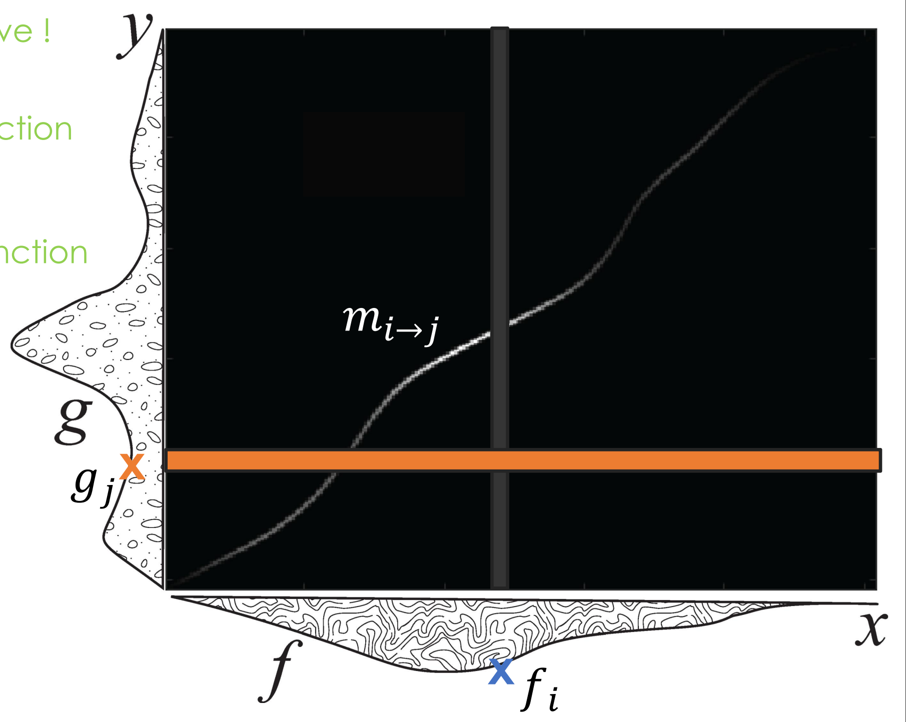
$$\min_m \boxed{\sum_i \sum_j \boxed{c_{i,j}}\,\boxed{m_{i\to j}}}$$

such that: $m_{i\to j} \geq 0$ Nb of particles is positive !

$\sum_i m_{i\to j} = g_j$ Reconstruct target function

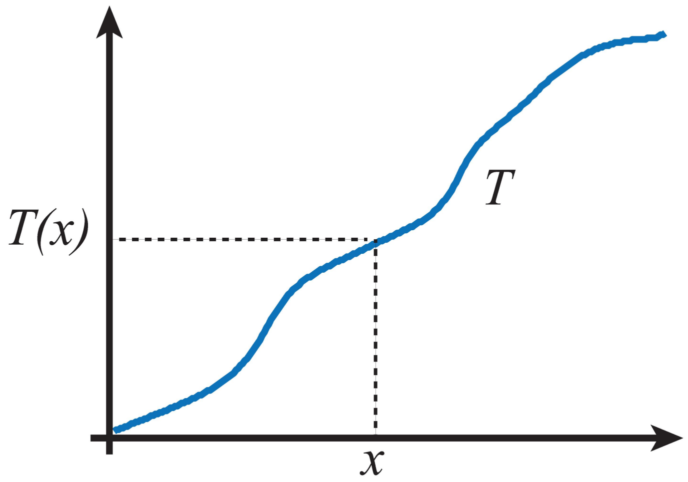
$\sum_j m_{i\to j} = f_i$ Reconstruct source function

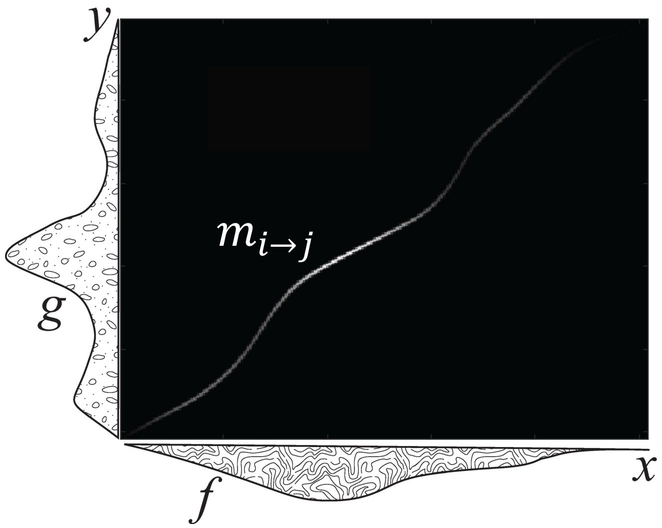
Work for transforming f into g



Discretization of the Kantorovich problem
Earth Mover's Distance

Intuition: comparison





- Finds a "transport map"
- Difficult non-linear problem
- May have no solution (e.g., a Dirac splitting in two)
- Leads to PDEs

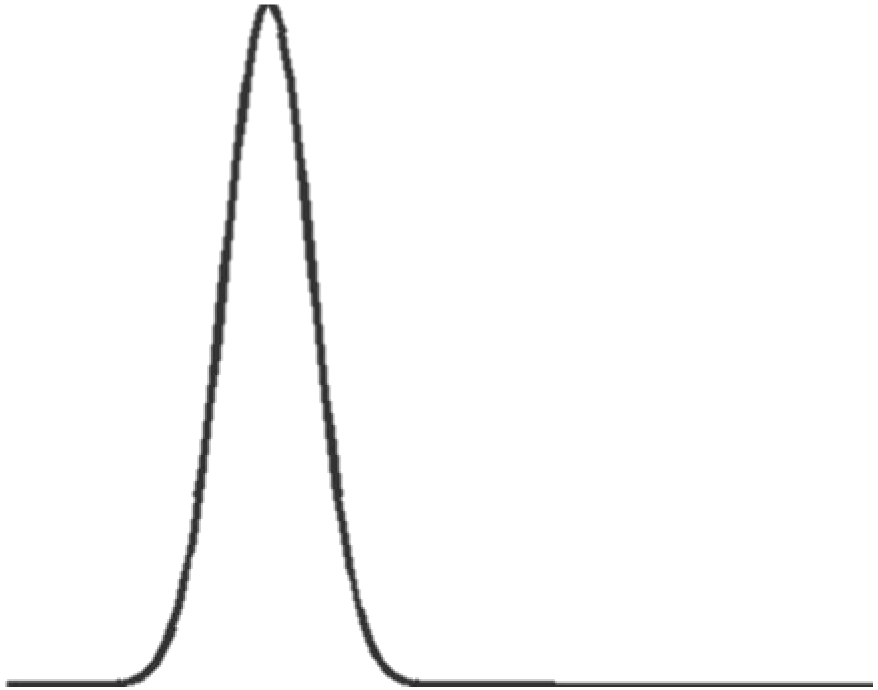
- Finds a "transport plan"
- Linear program
- "Always" has solution (i.e., under reasonable assumptions)
- Also has dual formulation

When it exists, the solution is the same.

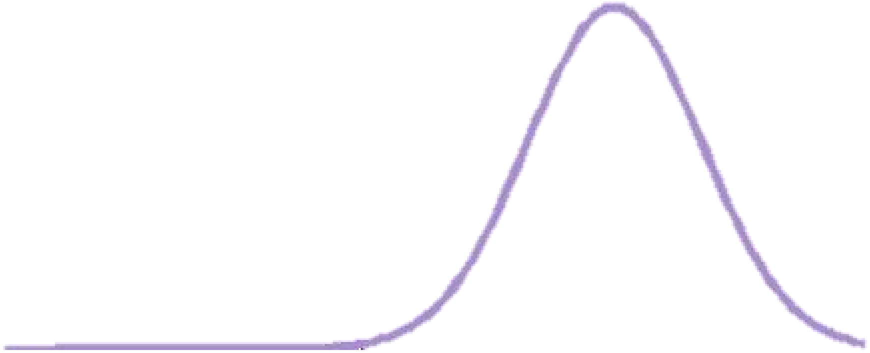
Often, $c(x, y) = \|x - y\|_p^p \Rightarrow W_p^p$

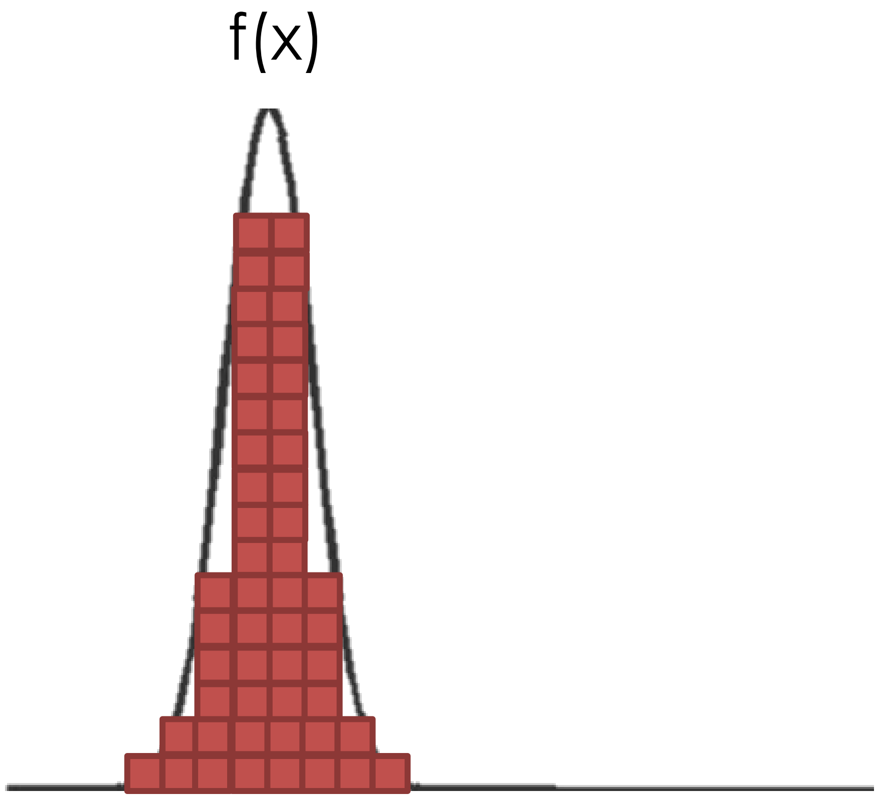
W_p is a distance

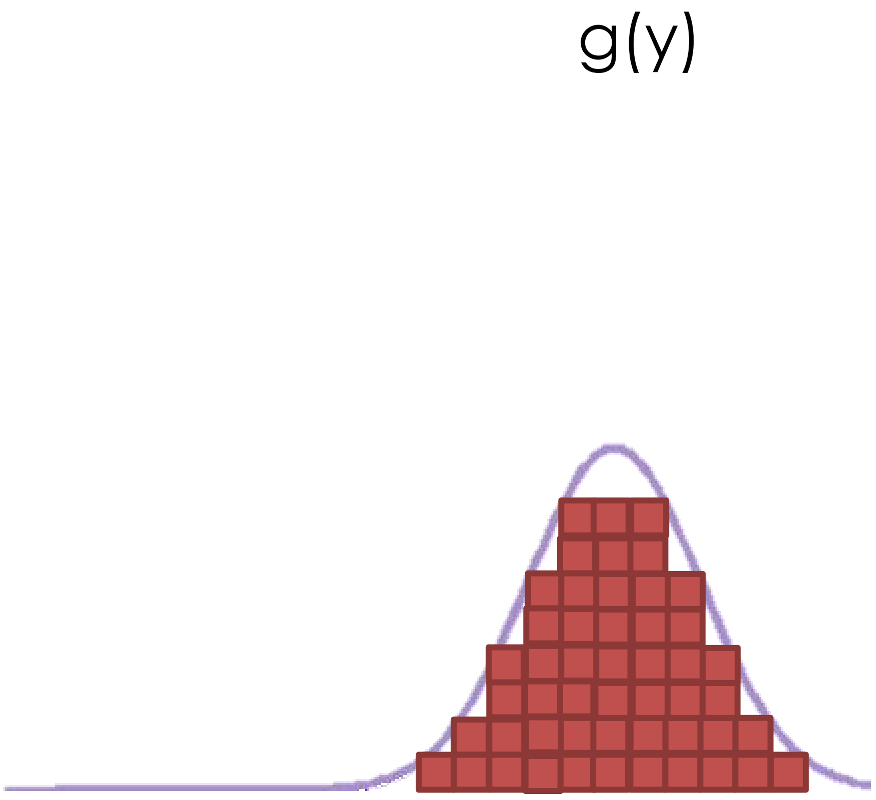
f(x)

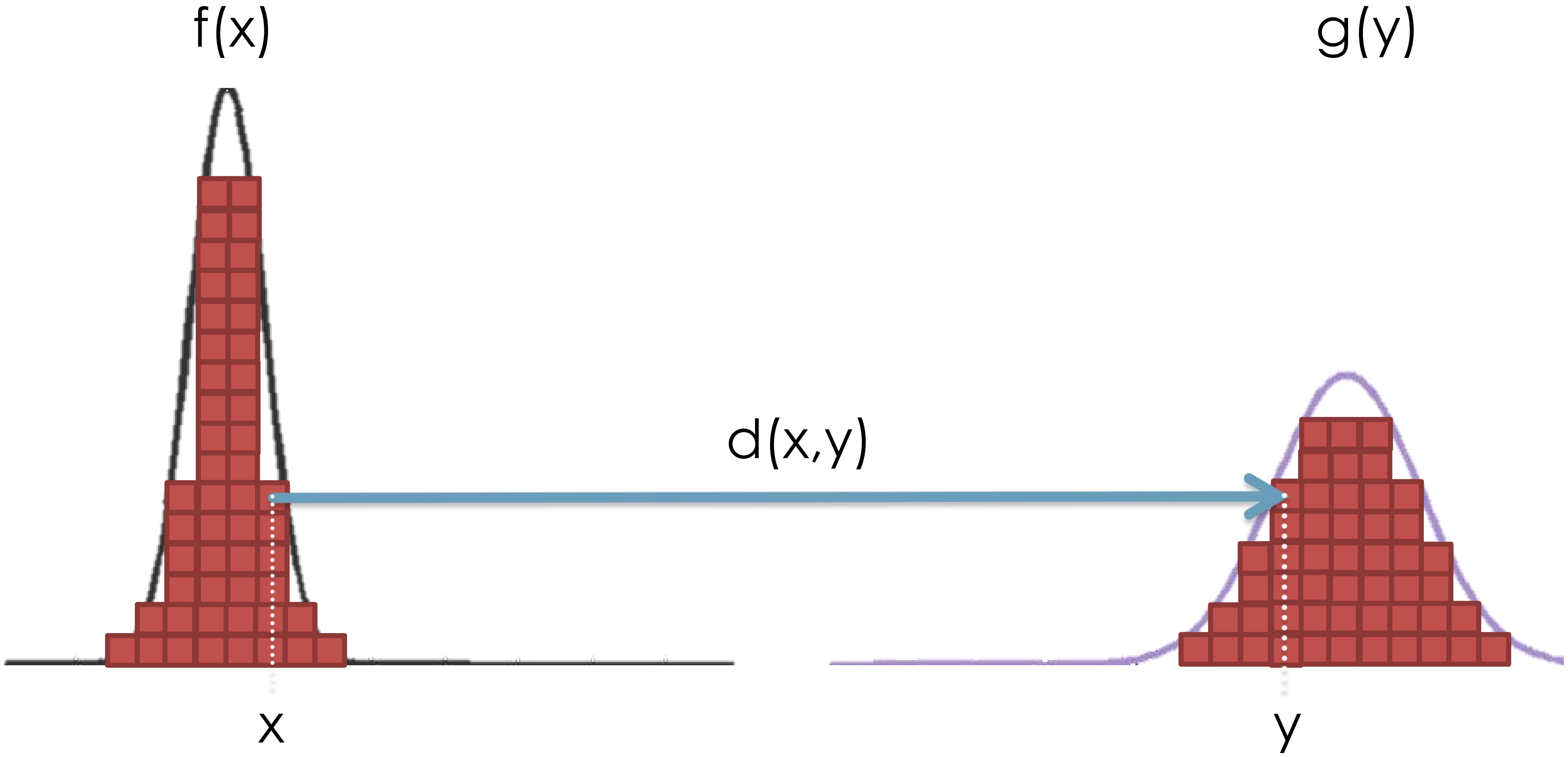


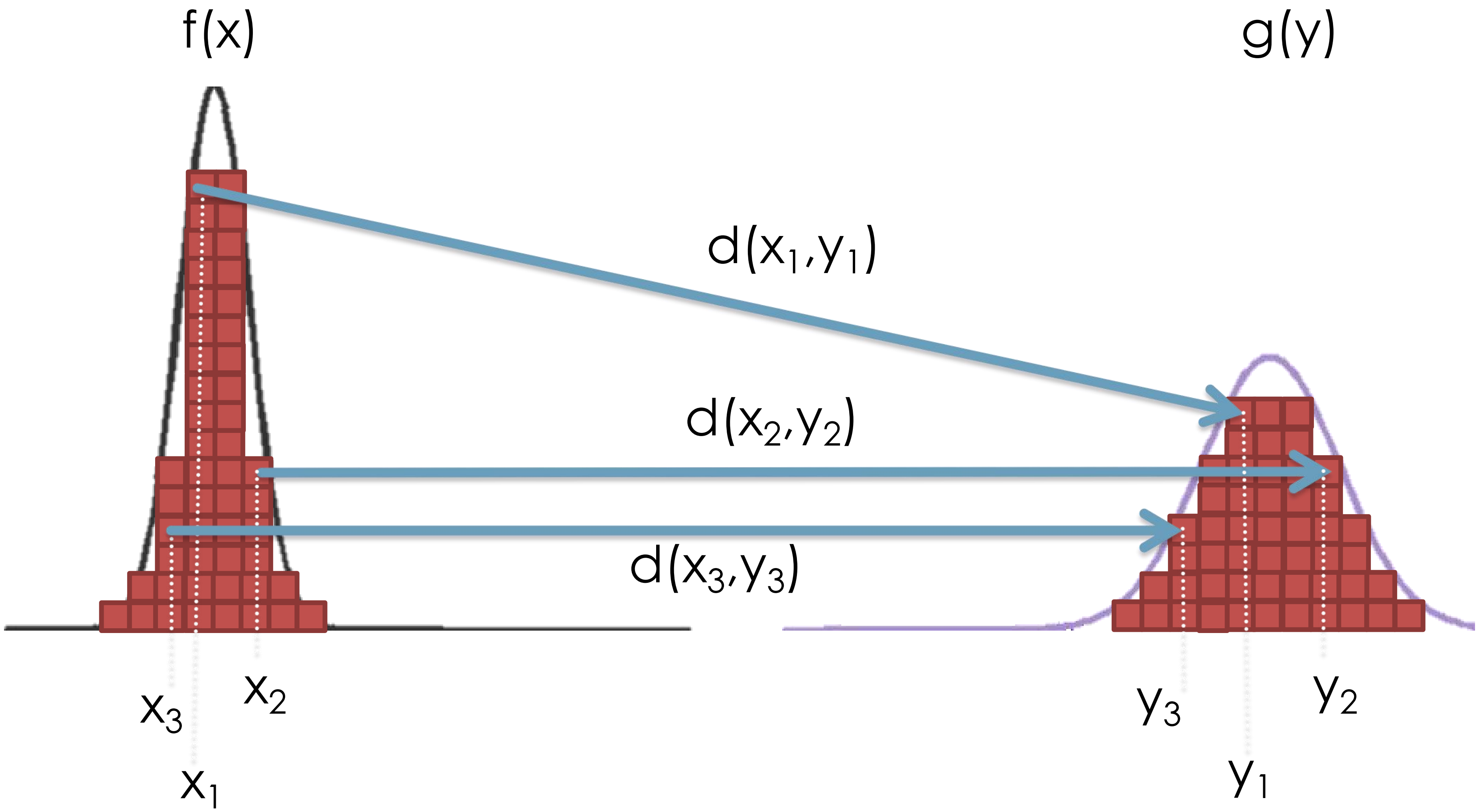
g(y)



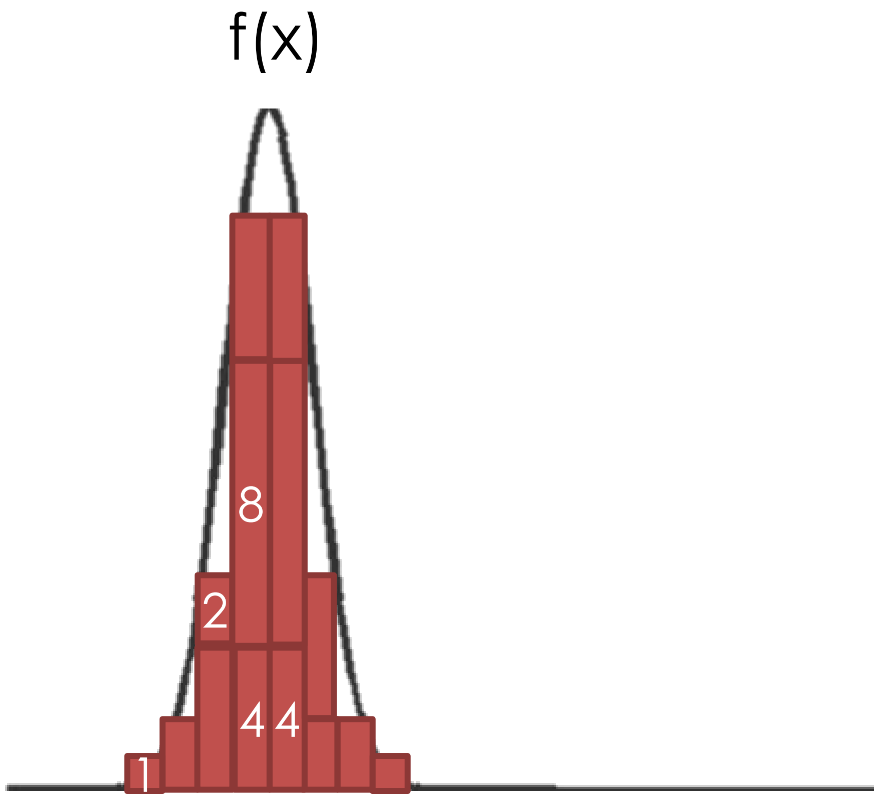

f(x)

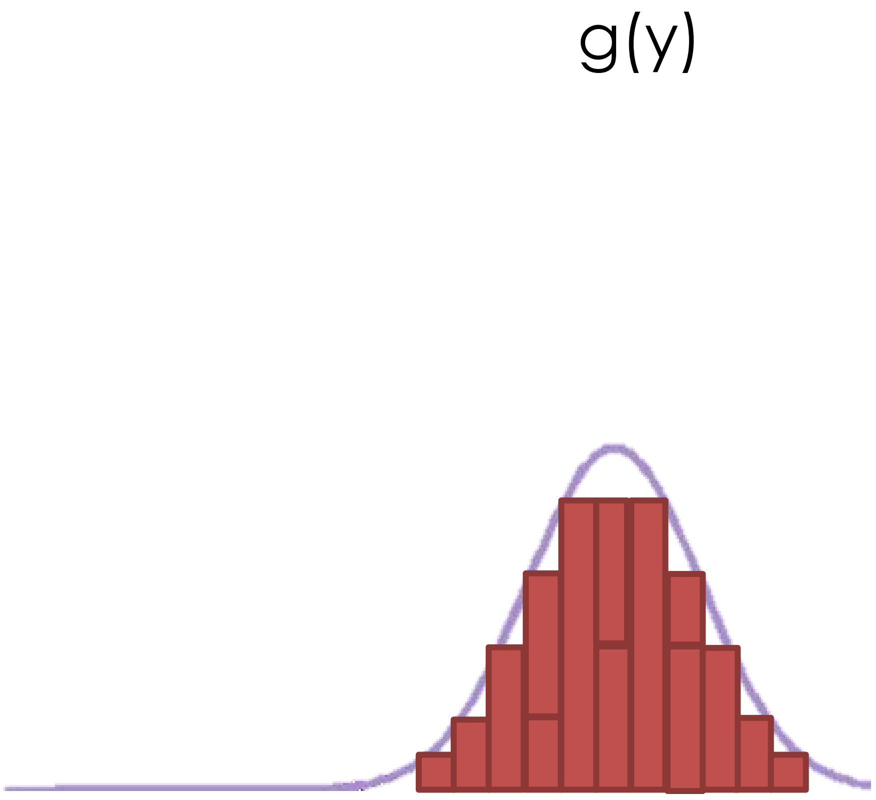

g(y)


f(x)
g(y)
d(x,y)
x
y

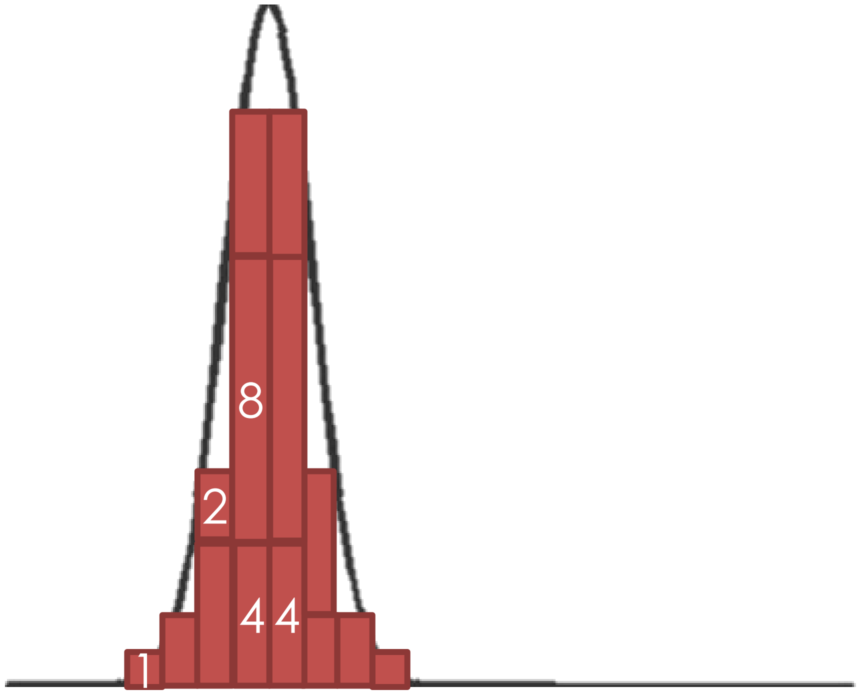


f(x)
g(y)
d(x1,y1)
d(x2,y2)
d(x3,y3)
x3
x1
x2
y3
y1
y2

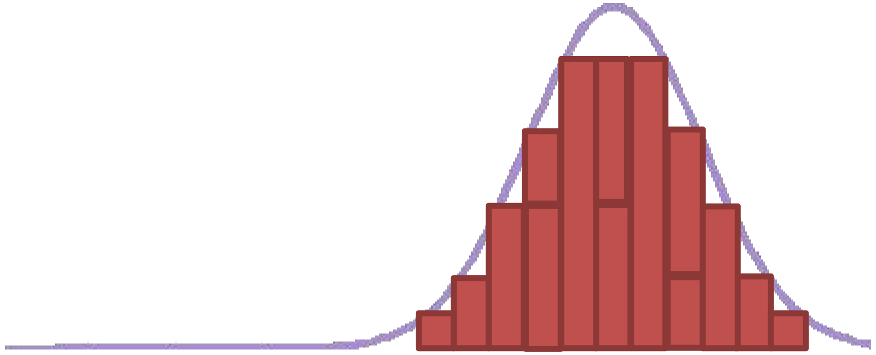

f(x)
8
2
4
4
1

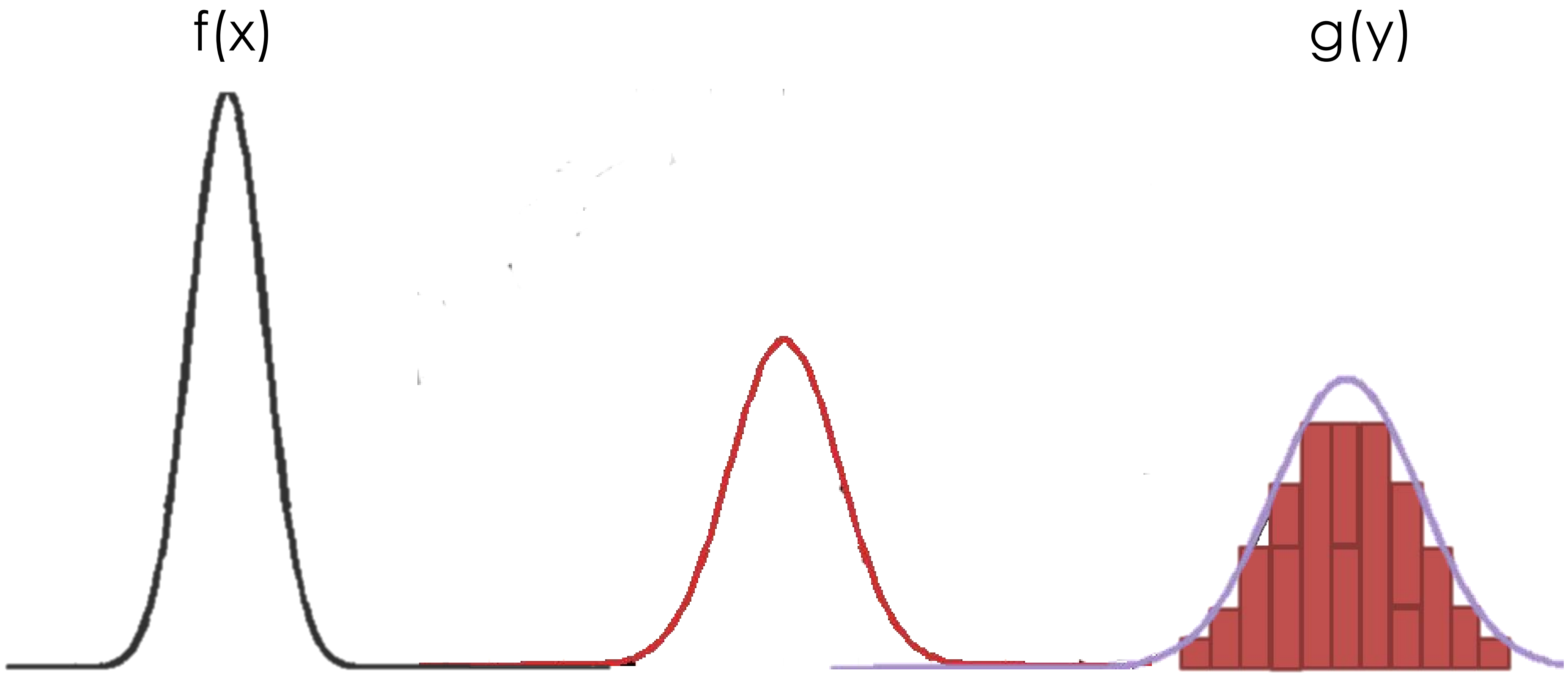

g(y)

f(x)

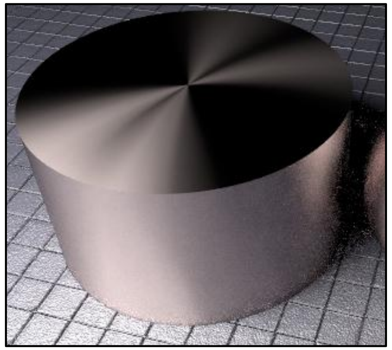


g(y)

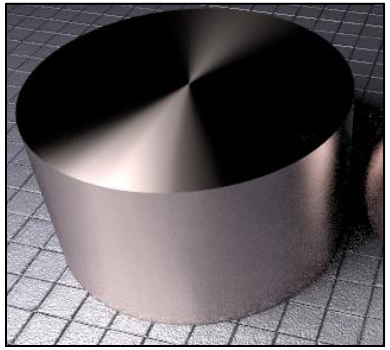



f(x)
g(y)

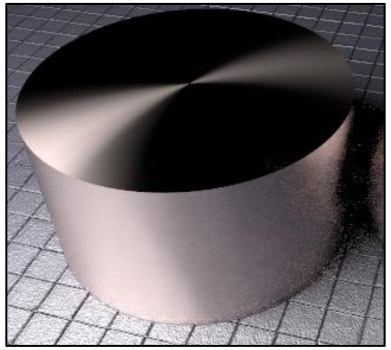
Application: BRDF



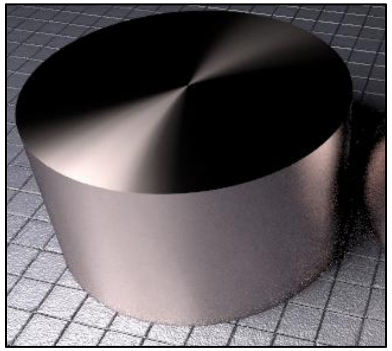
Linear interpolation



Function A



Function B



Displacement interpolation

[Bonneel et al. 2011] *Displacement Interpolation using Lagrangian Mass Transport*, Siggraph Asia

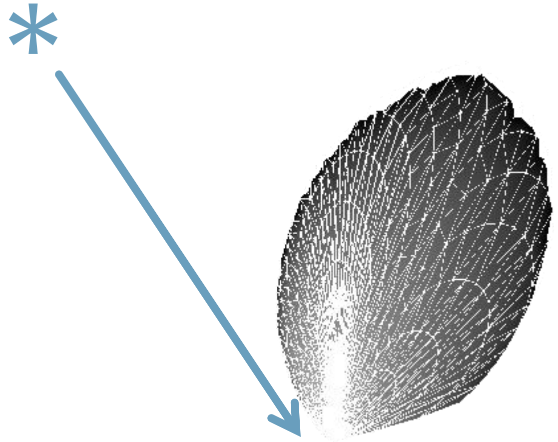
Displacement Interpolation using Lagrangian Mass Transport

Nicolas Bonneel, Michiel van de Panne, Sylvain Paris, Wolfgang Heidrich

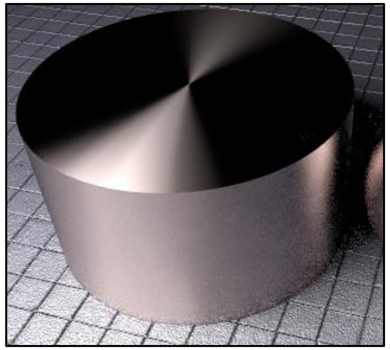
SIGGRAPH Asia 2011

Example: BRDF

- "Bidirectional Reflectance Distribution Function"



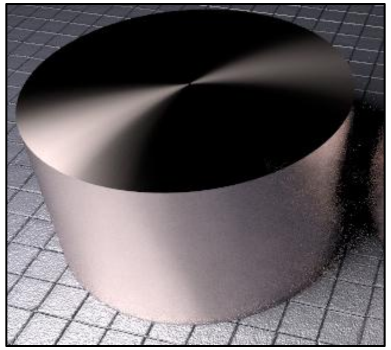
Example: BRDF



Function A

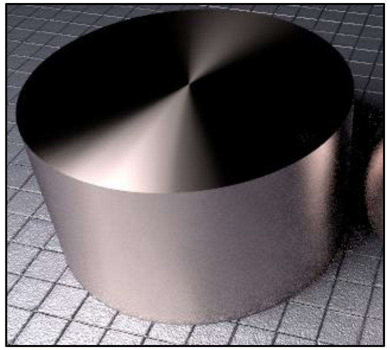
?

Interpolation

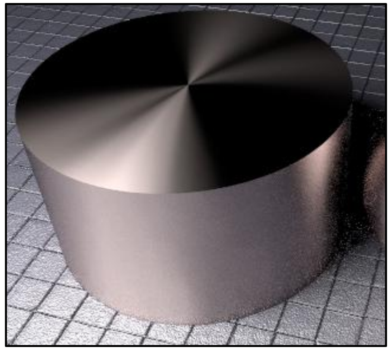


Function B

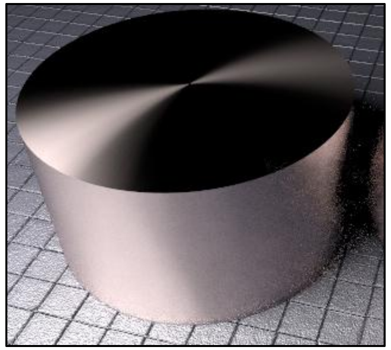
Example: BRDF



Function A

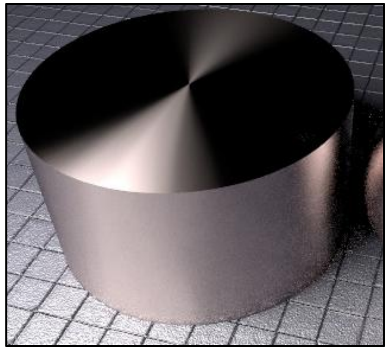


Linear
interpolation

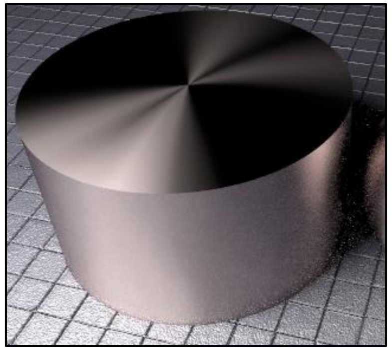


Function B

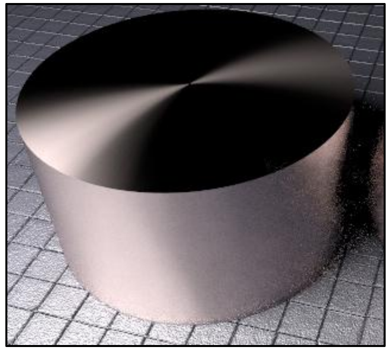
Example: BRDF



Function A

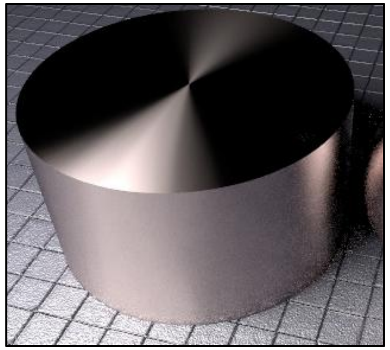


Linear interpolation

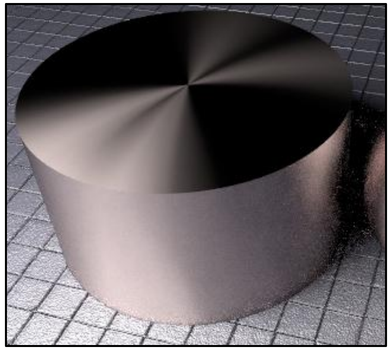


Function B

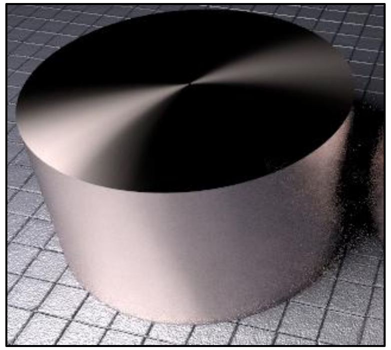
Example: BRDF



Function A

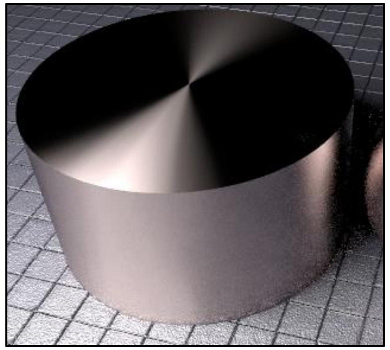


Linear interpolation

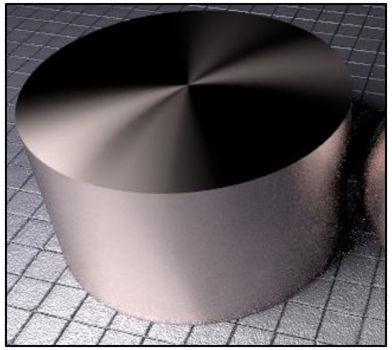


Function B

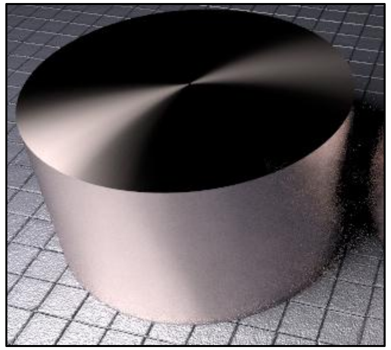
Example: BRDF



Function A

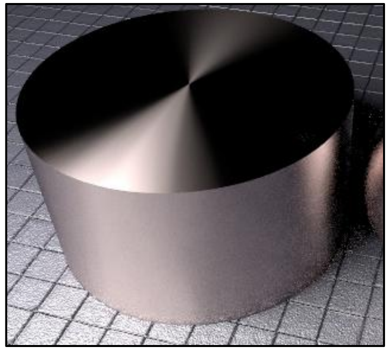


Linear interpolation

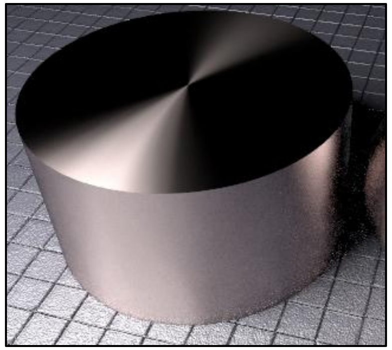


Function B

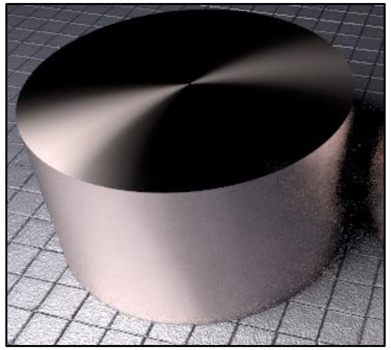
Example: BRDF



Function A

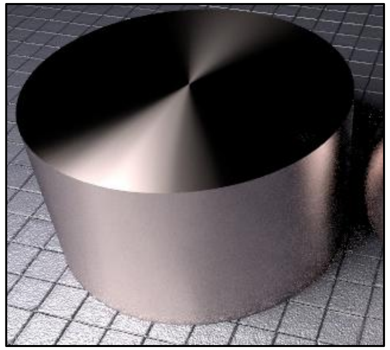


Displacement interpolation

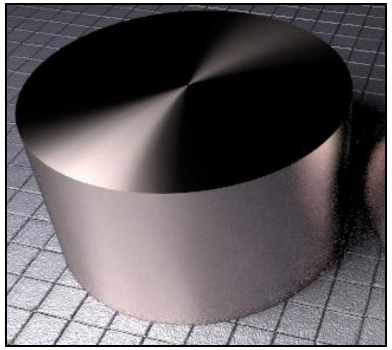


Function B

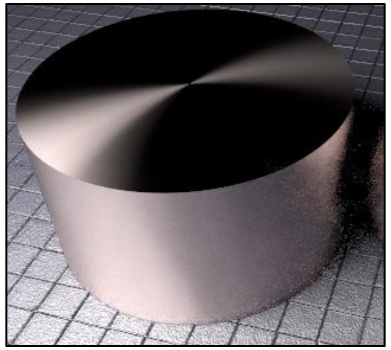
Example: BRDF



Function A

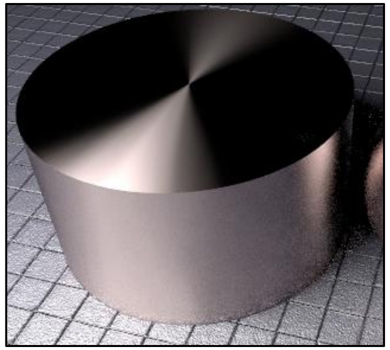


Displacement interpolation

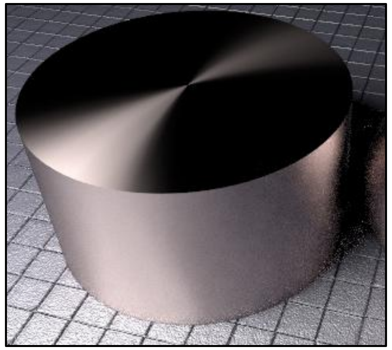


Function B

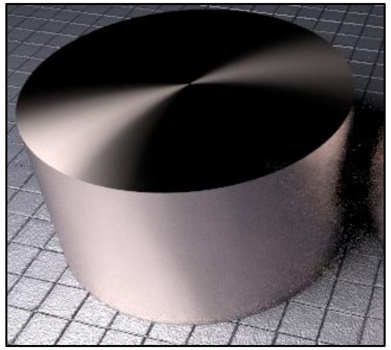
Example: BRDF



Function A

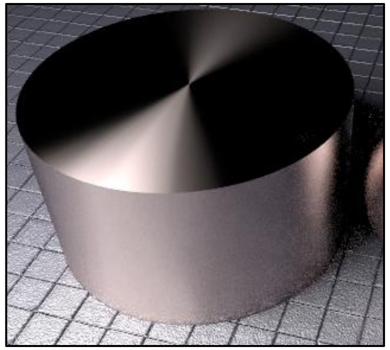


Displacement interpolation

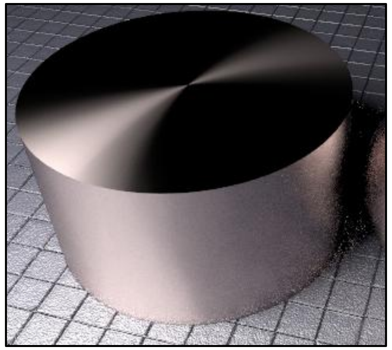


Function B

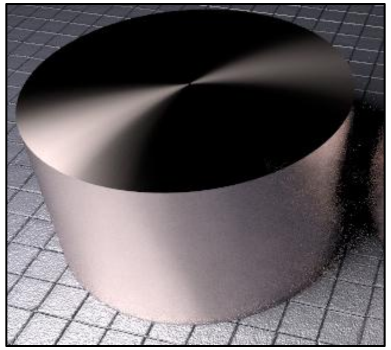
Example: BRDF



Function A



Displacement interpolation



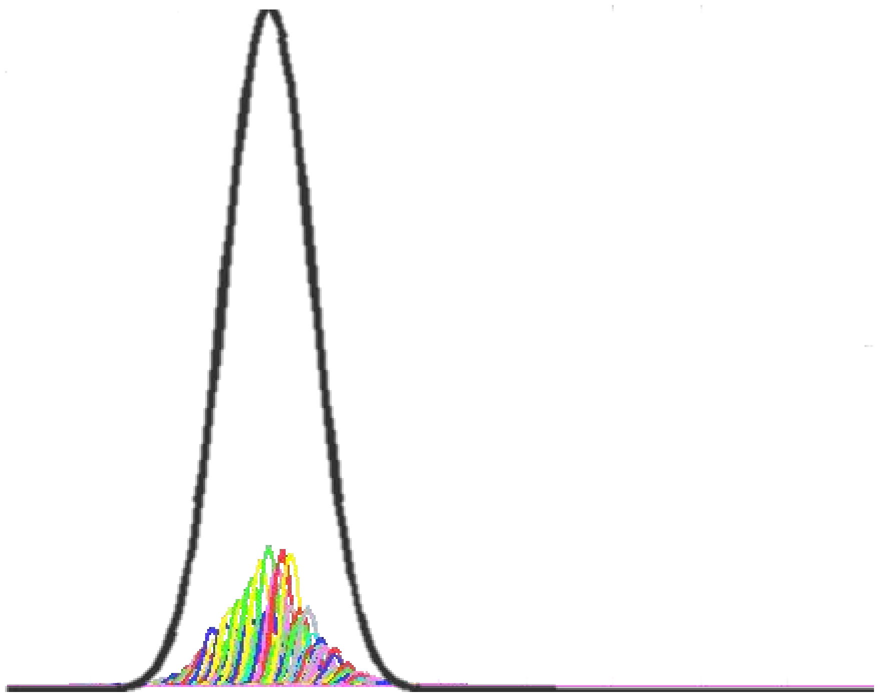
Function B

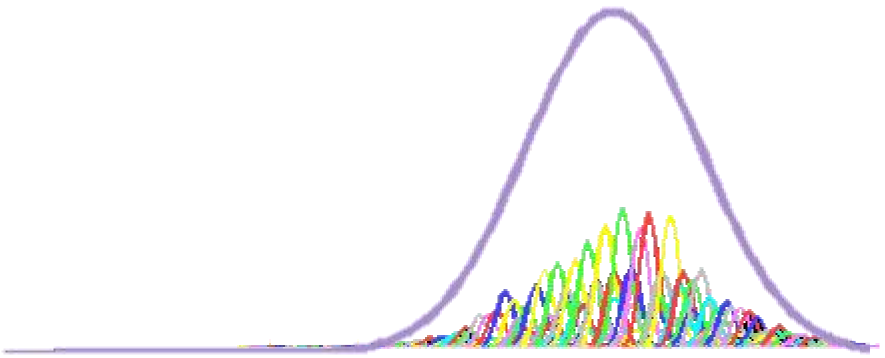
Four steps

- Decompose PDFs into non-negative radial basis functions
- Optimal transport computation
- Partial advection
- Reconstruct interpolated PDF

- (+optional multiscale approach)

Radial Basis Function decomposition





Transport computation

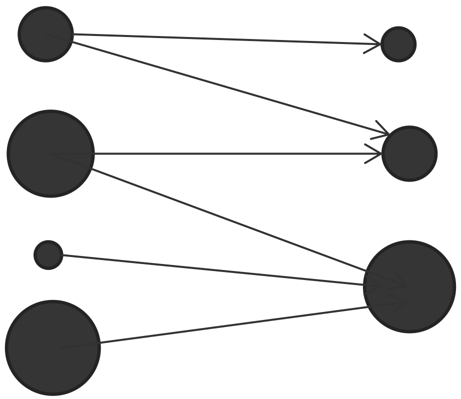
$$\min \sum_{i,j} c_{i,j} m_{i,j}$$

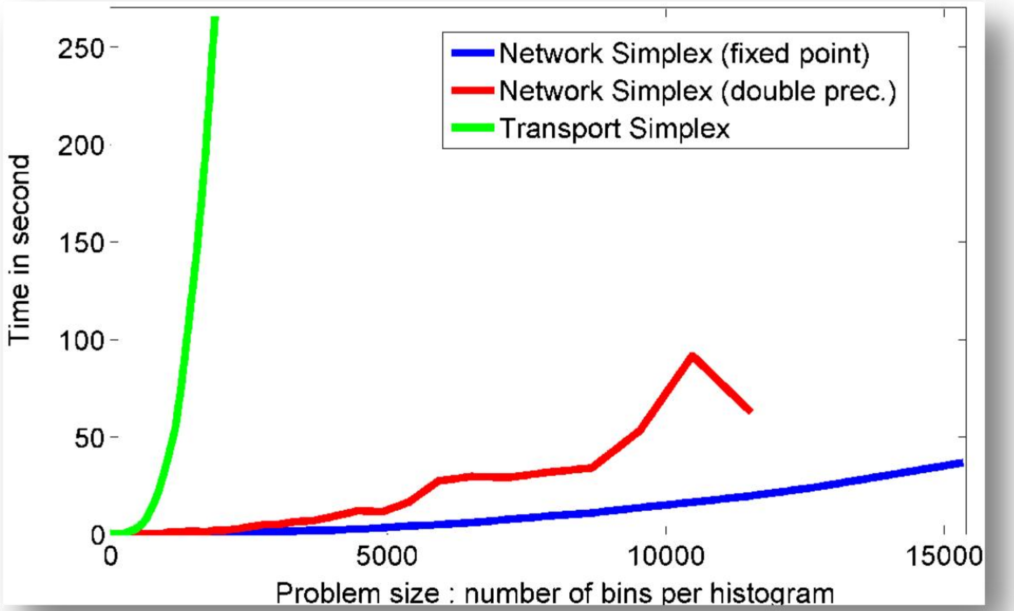
s.t

$$\sum_j m_{i,j} = f_i$$

$$\sum_i m_{i,j} = g_j$$

- Transport RBF weights
- Network simplex > Transportation simplex





Auction algorithm for assignment

- Consider instead: $\max \sum a_{ij}$ over complete assignments $(i, j) \in S$ and $j \in A(i)$
  - a_{ij} : how much person i is ready to pay for object j
    p_j : Price person j will actually pay

Solve dual: $\min_{p,\ \pi} \sum \pi_i + \sum p_j$ s.t. $\pi_i + p_j \geq a_{ij}\ \ \forall i,\ j \in A(i)$

- At optimality $\pi_i = \max_{k \in A(i)} a_{ik} - p_k = a_{ij(i)} - p_{j(i)}$ (saturates constraint)
- Profit of person i: $\pi_i = \max_{j \in A(i)} v_{ij}$
  with benefit for object $j \in A(i)$: $v_{ij} = a_{ij} - p_j$

- Add some slack: $\pi_i - \epsilon = \max_{k \in A(i)} a_{ik} - p_k - \epsilon \leq a_{ij(i)} - p_{j(i)}$ optimal if $\epsilon < \frac{1}{N}$

"The auction algorithm", Bertsekas and Castanon

Auction algorithm for assignment

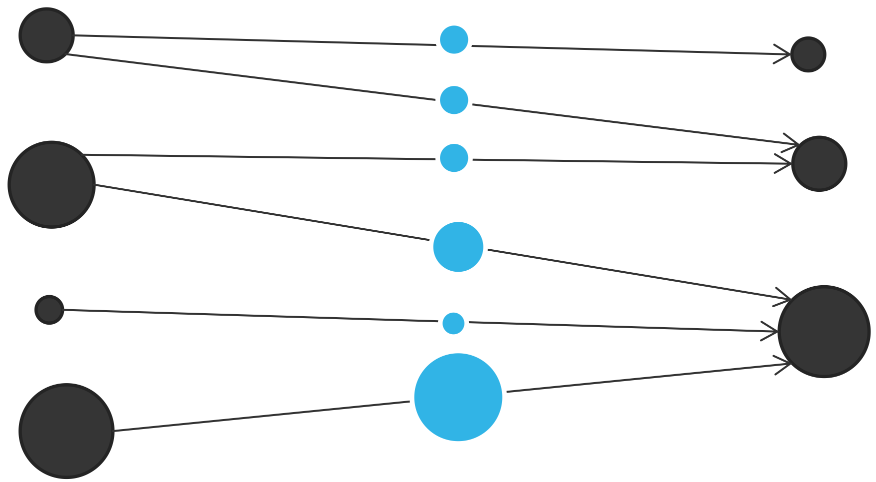
- Start with some assignment S
- For each unassigned person i, find object j^* maximizing benefit, and the benefit w_i of the second best.
  Compute bid : $b_{ij^*} = a_{ij^*} - w_i + \epsilon$
- For each object j : $P(j)$ is the set of persons who bid for j.
  - If $P(j) \neq \emptyset$: $p_j \leftarrow \max_{i \in P(j)} b_{ij}$; remove (i, j) from S, and add (i^*, j) (i^* best bidder)
  - If $P(j) = \emptyset$, p_j unchanged

Auction algorithm for optimal transport (1989)

- In O(N A log(N C))
- Idea: convert problem to assignment with duplicated sources/sinks
- Works on similarity classes
- In the previous algo, replace "second best" by "second best among other classes"

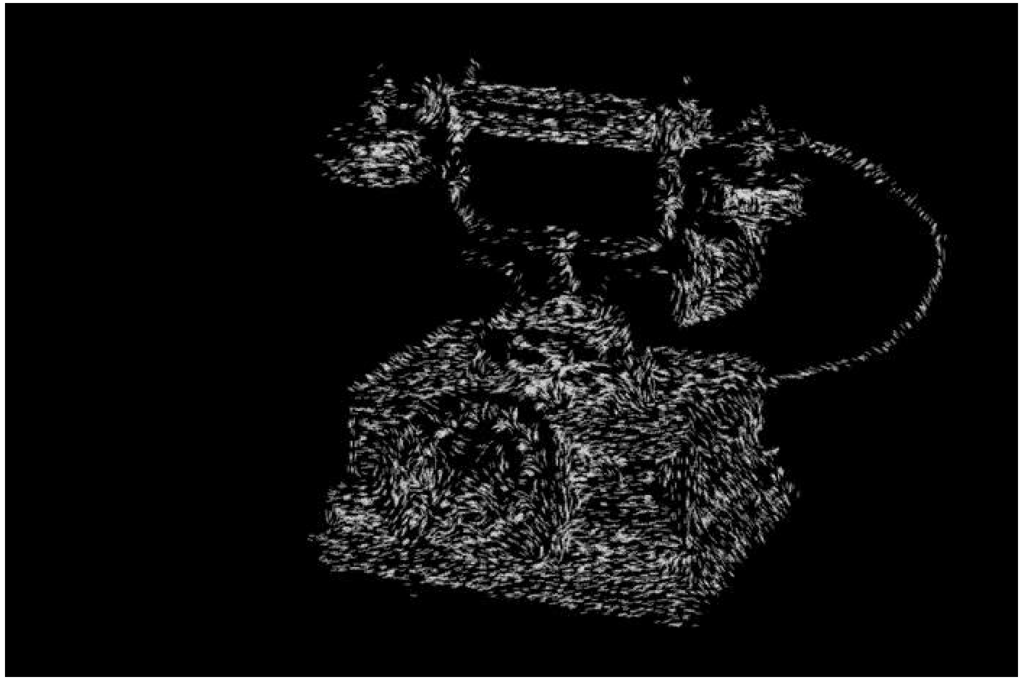
Interpolation

- Divide Gaussian function w.r.t to transported weights
- We advect.

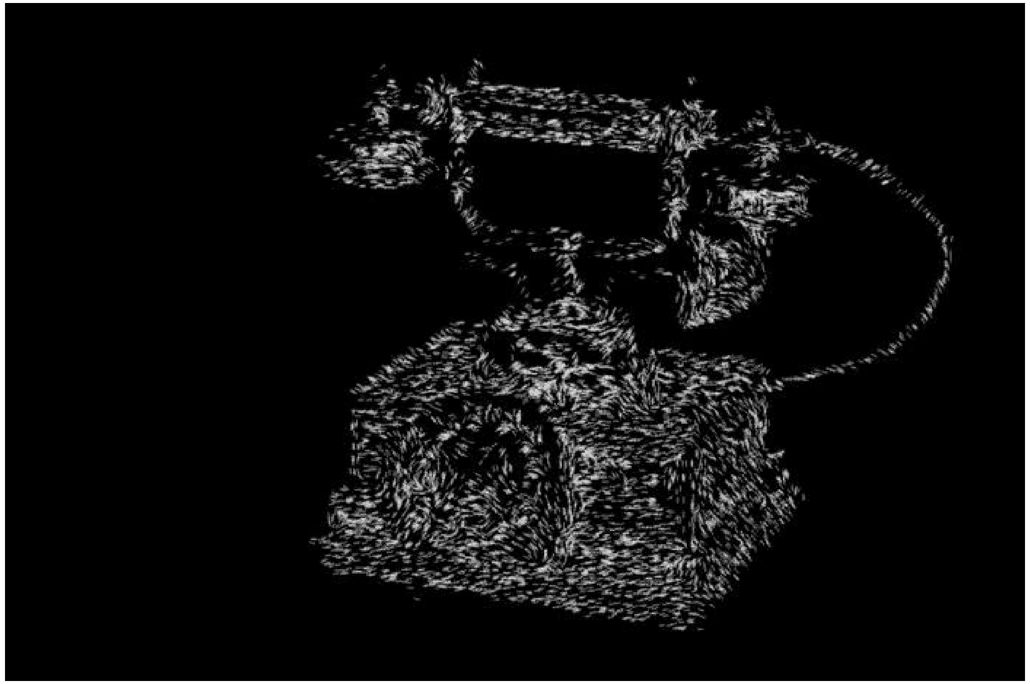


Results

Results

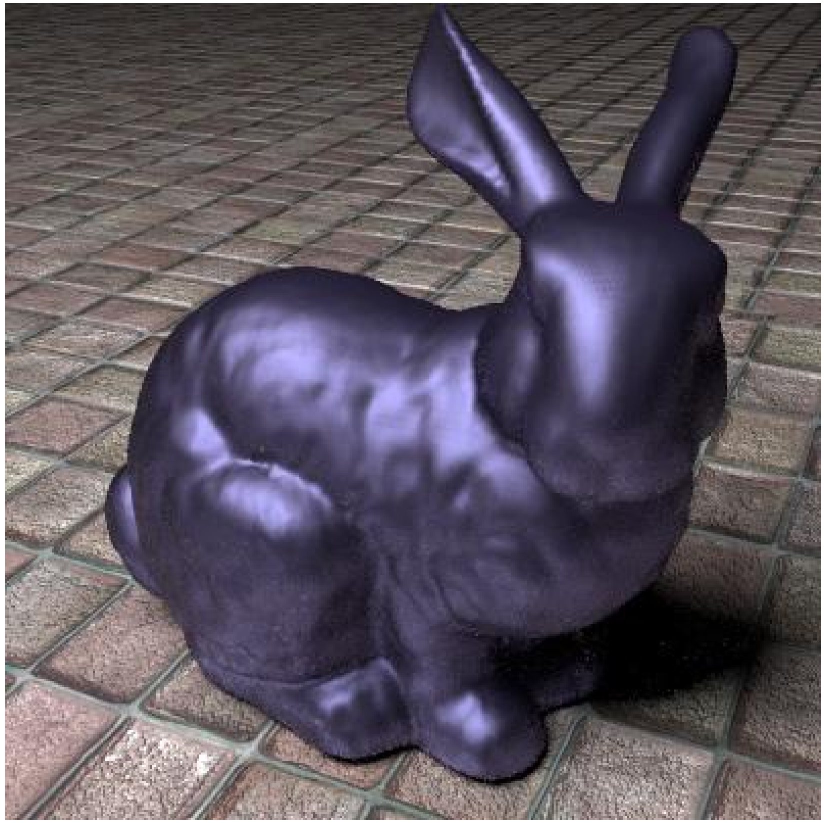


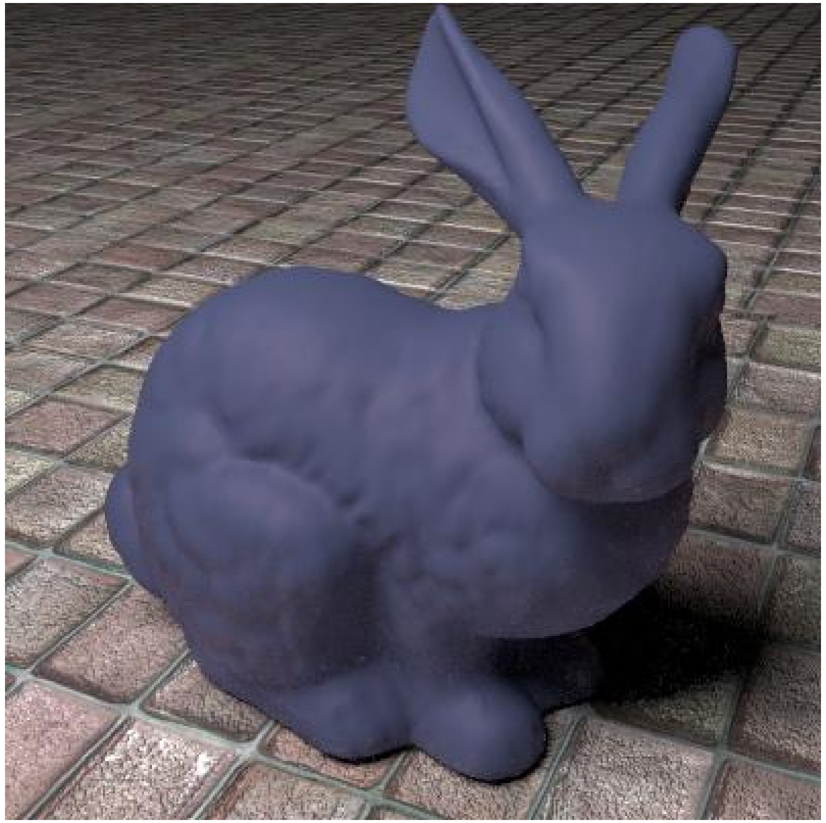
Naive



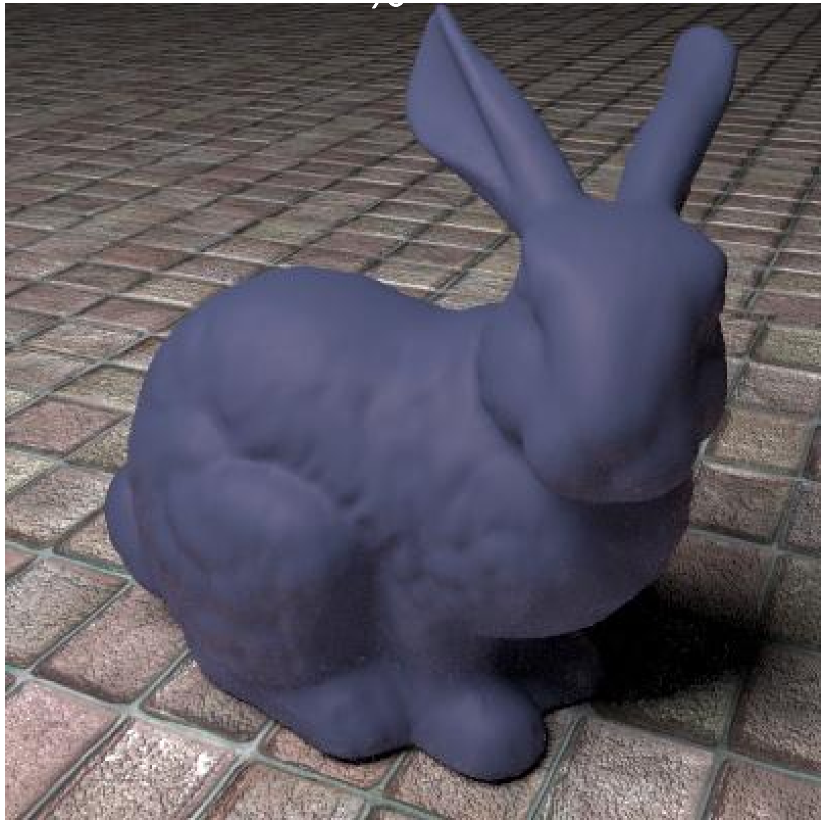
EMD
(minimize kinetic energy)

Results



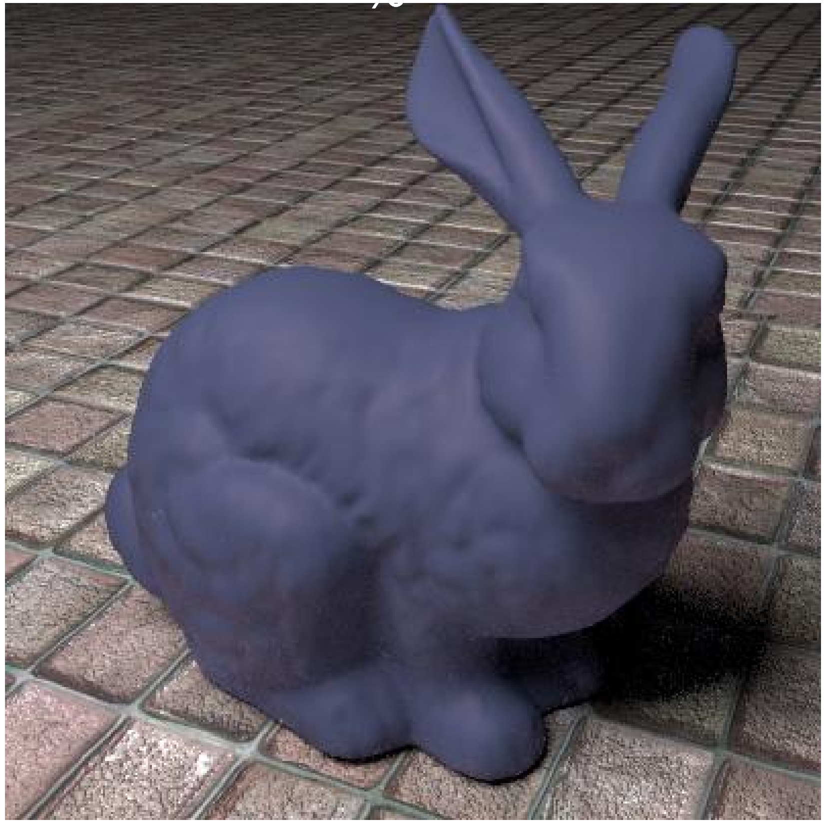


Results



Linear interpolation

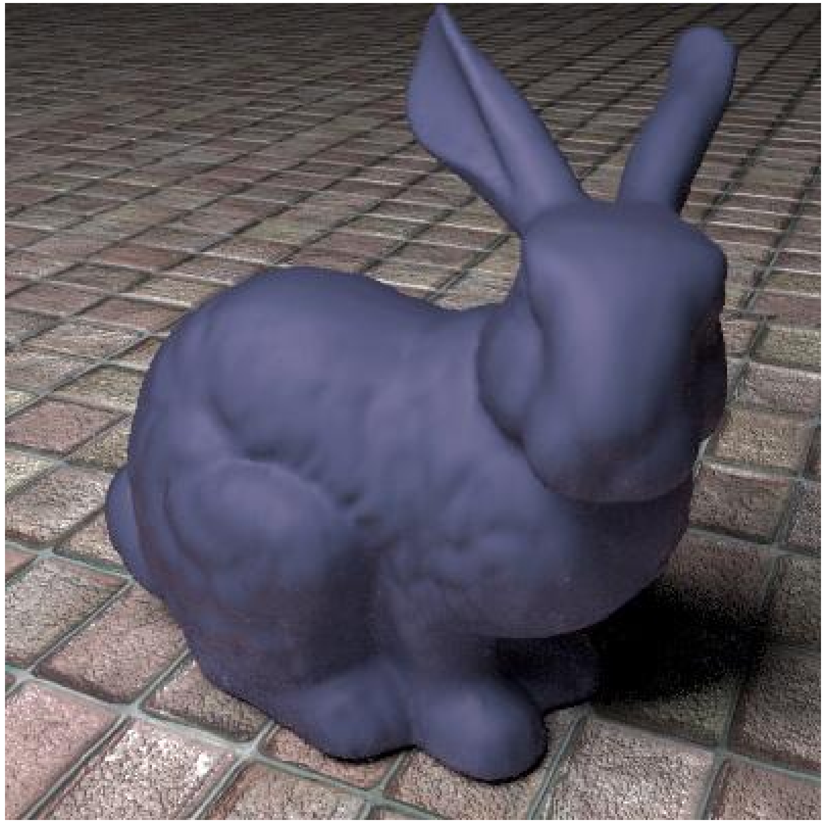
Results



Displacement interpolation

Results

Linear interpolation



Displacement interpolation

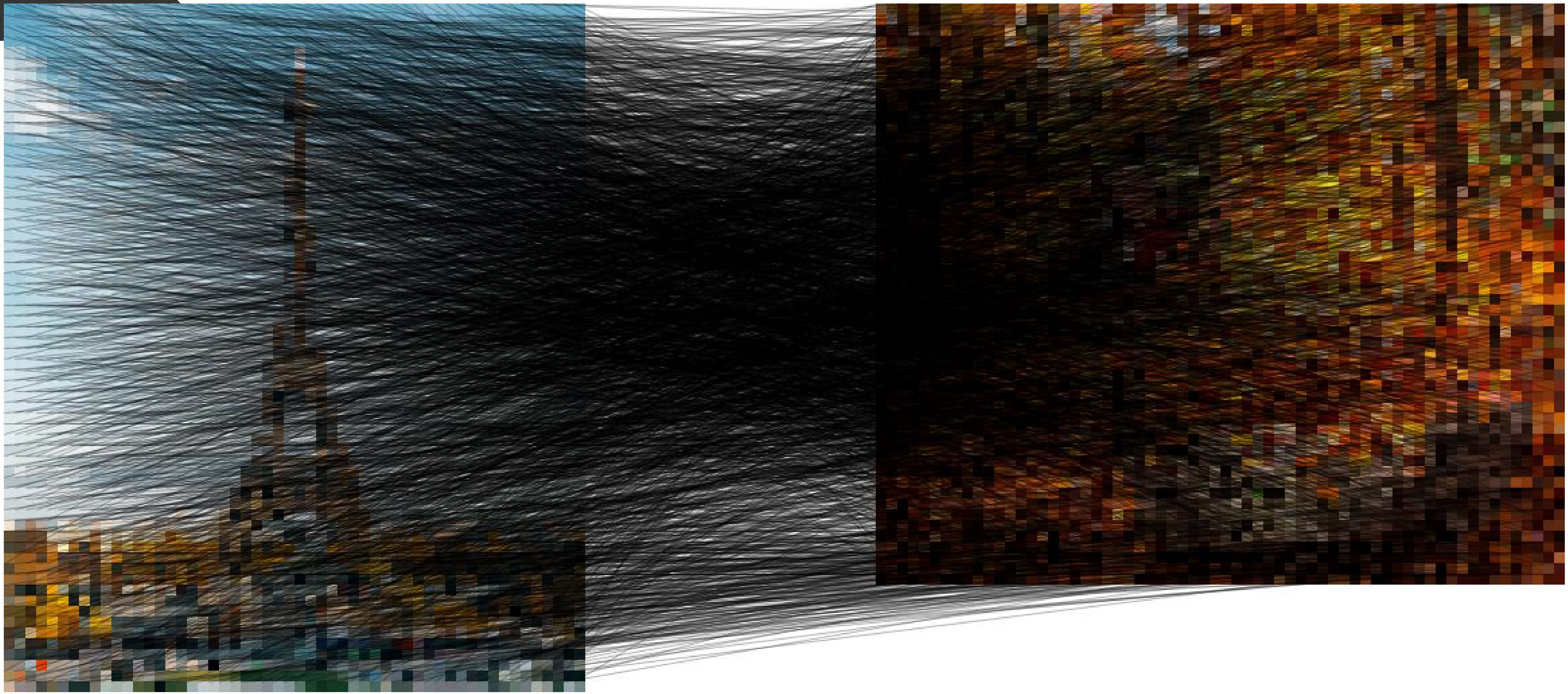
Applications to Color Grading

Input photo

Target style

Input photo

Target style


Input photo
Target style


Input photo


Target style


1733 WORMSLOE

Results

Model

Input

Sliced and Radon Wasserstein Barycenters of Measures

Nicolas Bonneel, Julien Rabin, Gabriel Peyré, Hanspeter Pfister

Journal of Mathematical Imaging and Vision (2014)

Multi-marginal transport

- Two ways transportation :

$$\min \sum_i \sum_j c_{i,j} m_{i\to j}$$

$$m_{i\to j} \geq 0$$
$$\sum_i m_{i\to j} = g_j$$
$$\sum_j m_{i\to j} = f_i$$

Number of non-zeros among M*N variables :

M+N-1

Multi-marginal transport

- Three ways transportation :

$$\min \sum_i \sum_j \sum_k c_{i,j,k} m_{i,j,k}$$

$$m_{i,j,k} \geq 0$$
$$\sum_i \sum_j m_{i,j,k} = h_k$$
$$\sum_i \sum_k m_{i,j,k} = g_j$$
$$\sum_j \sum_k m_{i,j,k} = f_i$$

Number of non-zeros among M*N*P variables :
M*N*P-(M*N+N*P+M*P)+(M+N+P-1)

Simple cases

- Transport 1 Gaussian ↔ 1 Gaussian
- Transport 1 Gaussian ↔ 1 Gaussian ↔ 1 Gaussian […]
- Transport = translation + scaling
- Transport 1D function ↔ 1D function (↔ 1D function […])

Optimal transport is simple for Gaussians

- Optimal transport and barycenters trivially solved for
  - Gaussian distributions with $c(x, y) = \|x - y\|^2$
    - $W_2^2(\mathcal{N}_0, \mathcal{N}_1) = \mathrm{tr}\left(\Sigma_0 + \Sigma_1 - 2\Sigma_{0,1}\right) + \|\mu_0 - \mu_1\|$ with $\Sigma_{0,1} = \left(\Sigma_0^{\frac{1}{2}} \Sigma_1 \Sigma_0^{\frac{1}{2}}\right)^{1/2}$
    - $T(x\ ) = \Sigma_{0,1} x$
    - Barycenter: $\mathcal{N}(\mu, \Sigma)$ with $\mu = \sum_k \lambda_k \mu_k$ and iterations

$$\Sigma^{(n+1)} = \sum_k \lambda_k \left(\sqrt{\Sigma^{(n+1)}} \Sigma_k \sqrt{\Sigma^{(n+1)}}\right)^{1/2}$$

Optimal transport is simple in 1D

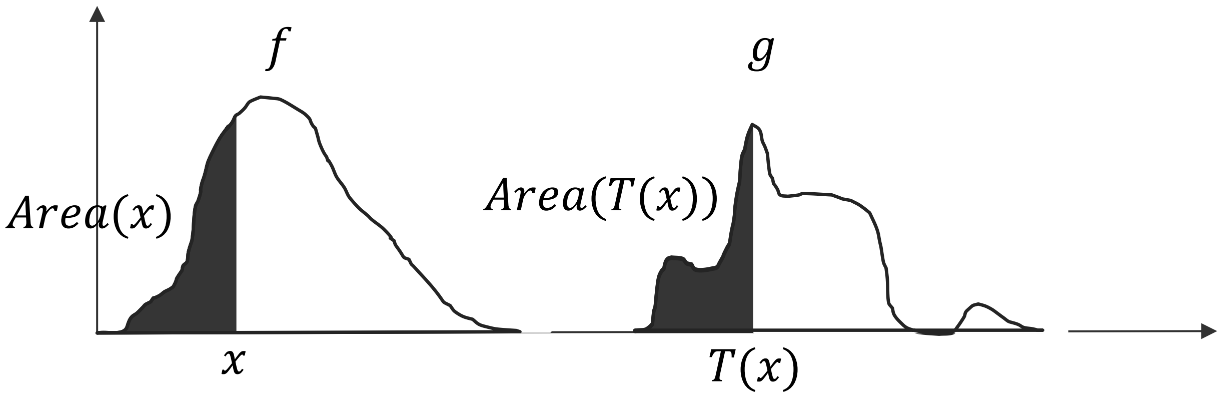
- Continuous case with density, convex cost, $\mu = f\,dx$, $\nu = g\,dy$
  - Need: $\int_{-\infty}^{x} f(x)dx = \int_{-\infty}^{T(x)} g(x)dx$

$$T = G^{-1} \circ F$$

with $F(x) = \int_{-\infty}^{x} f(x)dx$ and $\mathrm{G}(x) = \int_{-\infty}^{x} g(x)dx$

Generalize G^{-1} : $G^{-1}(y) = \min_x \{y = G(x)\}$

Quantile function: e.g.: "what salary corresponds to the first percentile"



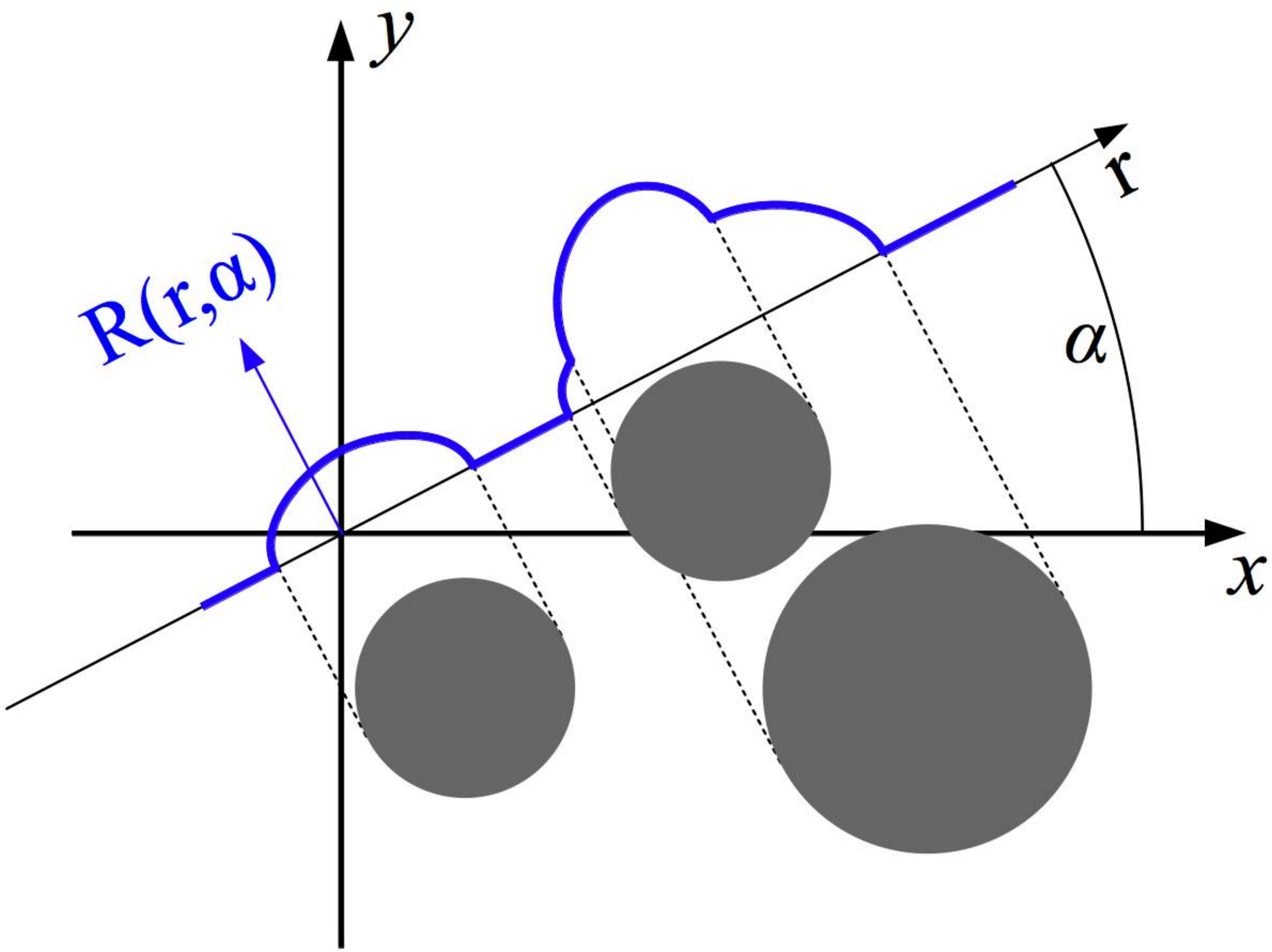
1D Case

OT Map: $T = G^{-1} \circ F$

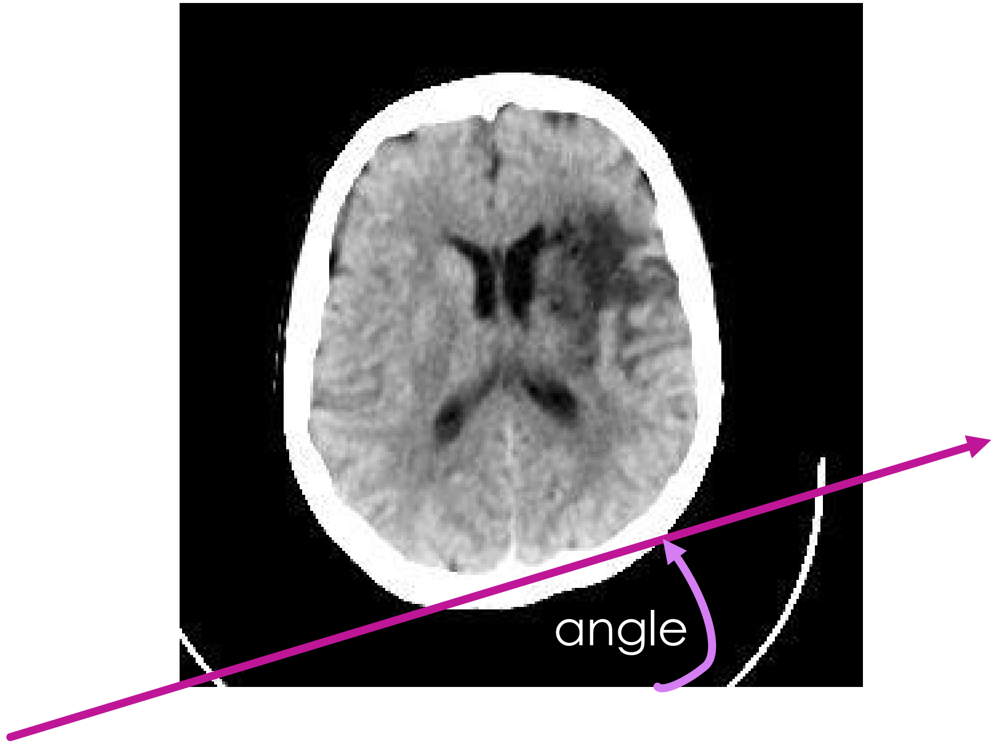
OT cost: $\int_0^1 c\left(F^{-1}(t) - G^{-1}(t)\right) dt$

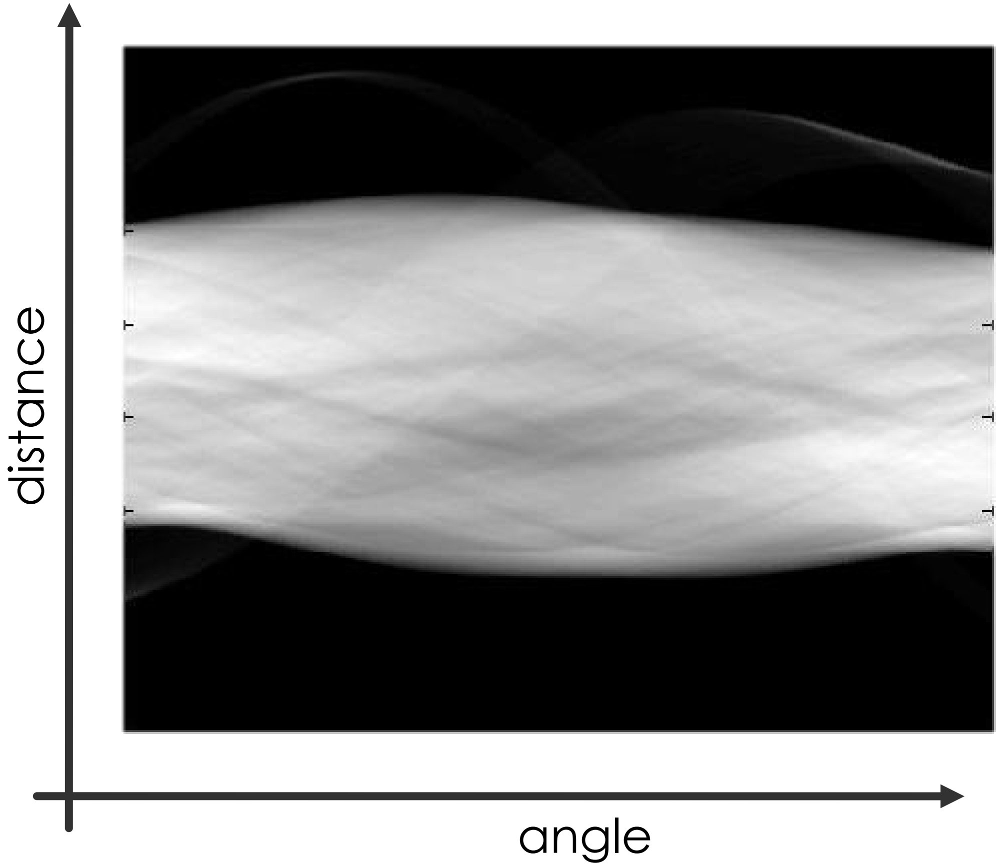
Interpolation: $F_{interp}^{-1}(x) = \sum_i \alpha_i \, F_i^{-1}(x)$

Radon transform

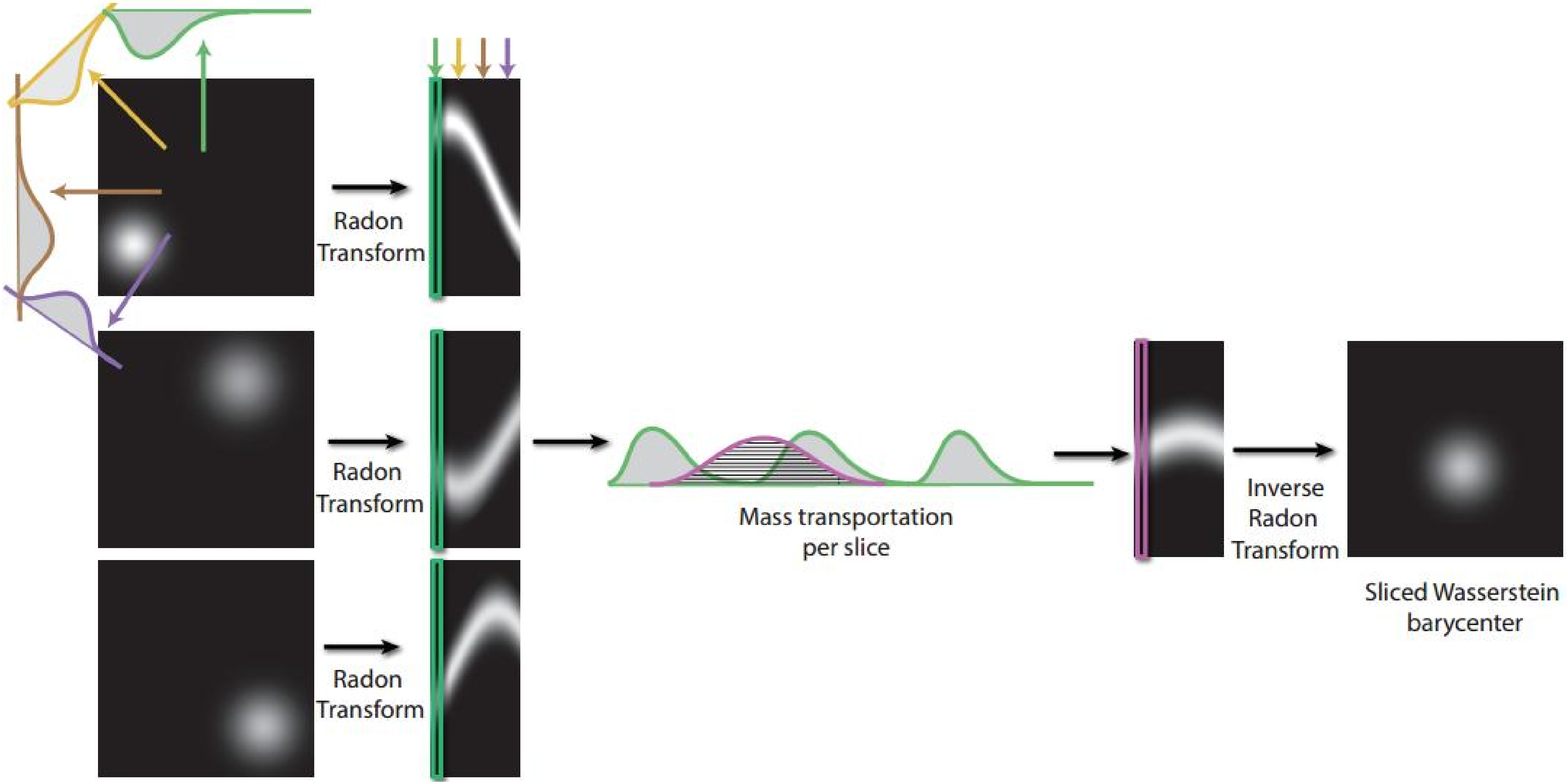


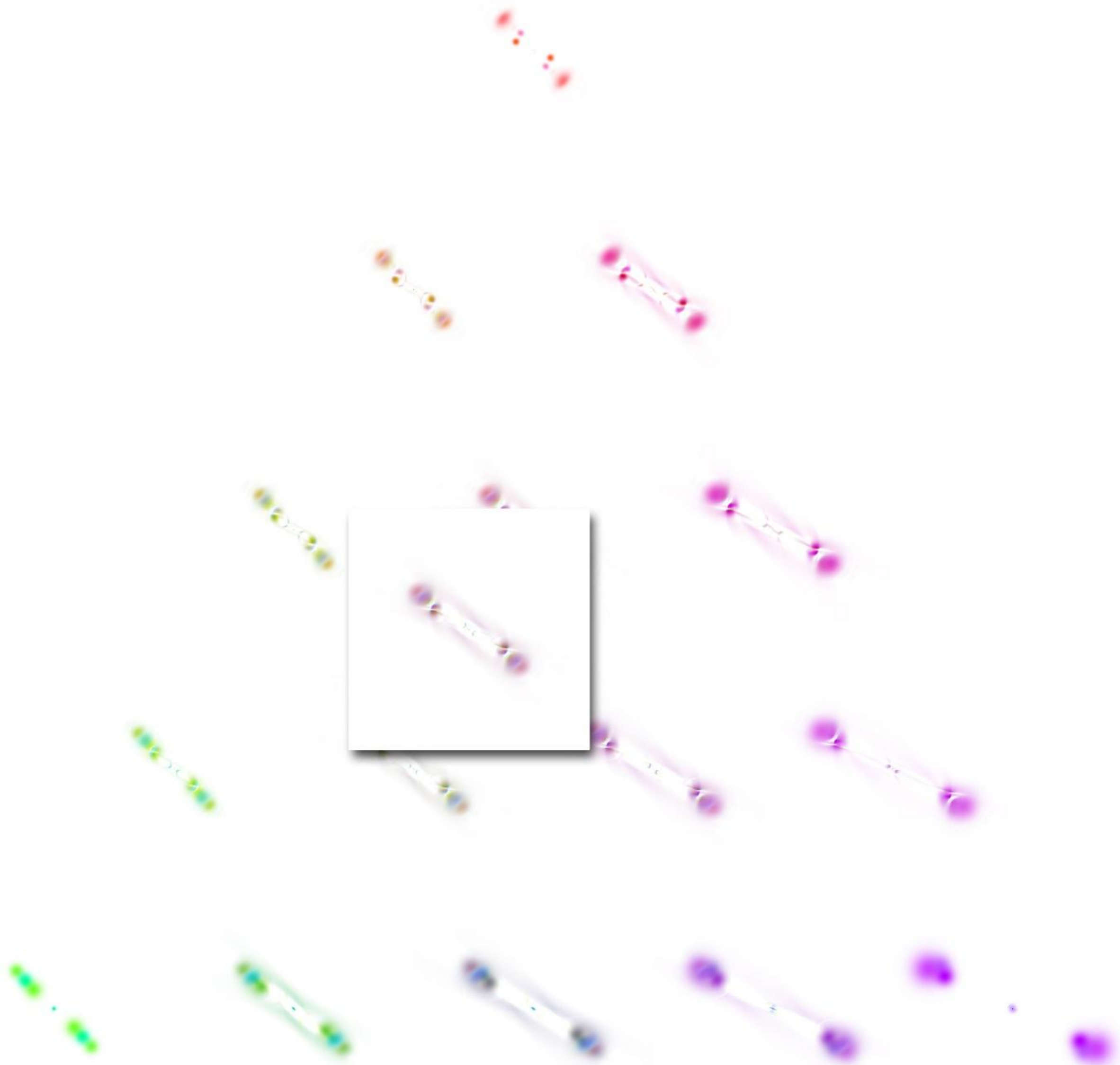
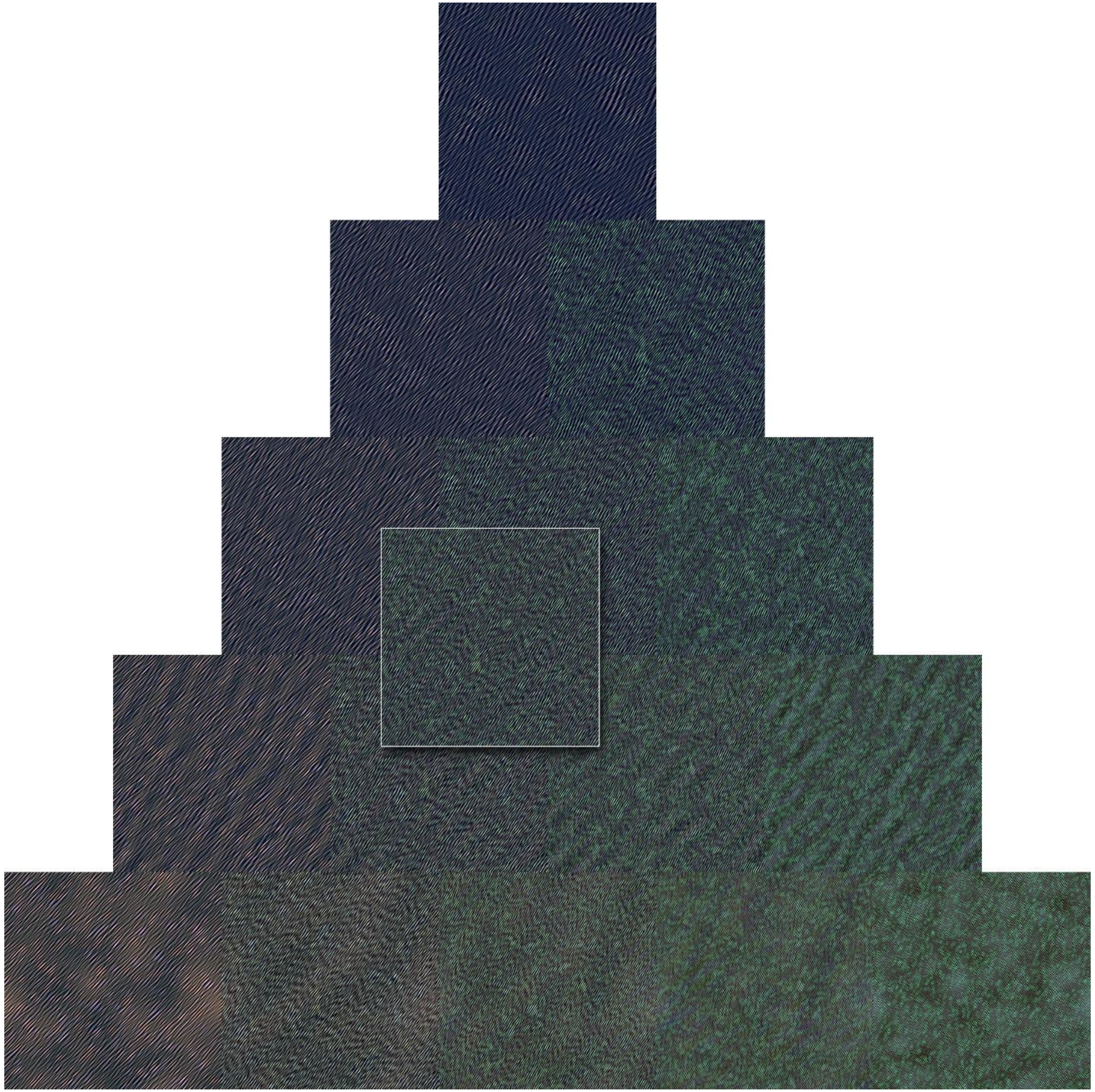
Radon transform





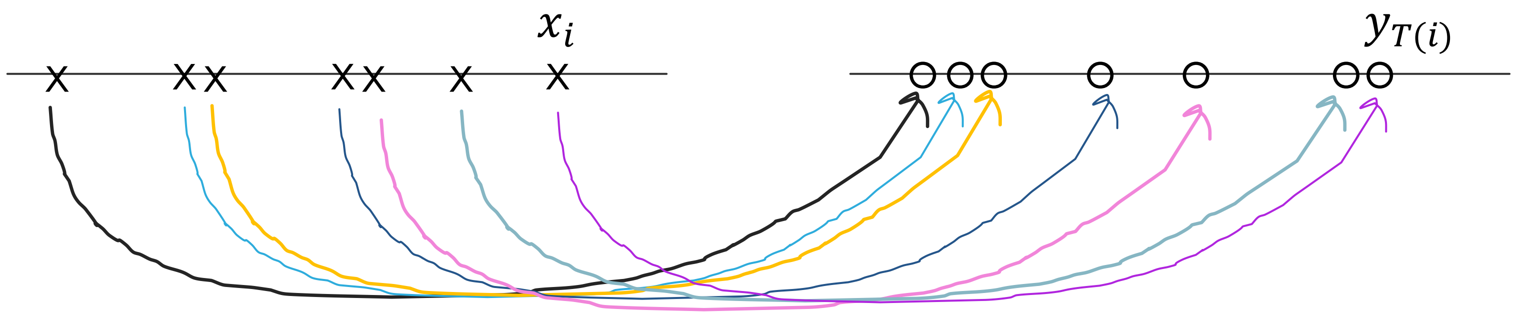
Method


Radon
Transform
Radon
Transform
Radon
Transform
Mass transportation
per slice
Inverse
Radon
Transform
Sliced Wasserstein
barycenter

1D Case, discrete

- Discrete case, $\mu = \sum_{i=1}^{n} \delta_{x_i}$, $\nu = \sum_{i=1}^{n} \delta_{y_i}$ (same for interpolating between more than 2 measures)
- Optimal transport for convex cost = pairing sorted samples



Sliced Wasserstein Distance

- For discrete high-dimensional distributions $\mu = \sum_{i=1}^{n} \delta_{x_i}$ and $\nu = \sum_{i=1}^{n} \delta_{y_i}$

Consider energy

$$SW(\mu, \nu) = \int_S W_2^2\big(\mathrm{proj}(\mu,\ \omega),\ \mathrm{proj}(\nu,\ \omega)\big)\, \mathrm{d}\omega$$

Where $\mathrm{proj}(\mu,\ \omega)$ is the 1-d distribution : $\mathrm{proj}(\mu,\ \omega) = \sum_i \delta_{\langle x_i,\ \omega \rangle}$ (same for ν)

And W_2^2 computes the 1-d squared Wasserstein distance

Sliced Wasserstein Distance

- Take a uniform random direction ω
  - $\omega \leftarrow \left(\mathcal{N}(0,1), \mathcal{N}(0,1), \mathcal{N}(0,1)\right)$ and normalize
- Project samples of μ and ν on ω : μ' = Proj(μ) and ν' = Proj(ν)
- Sort μ' and ν', i.e, find permutations σ_μ and σ_ν
- To compute the Sliced Wasserstein Distance:

$$d^2 \leftarrow d^2 + \sum_i \left|\left\langle x_{\sigma_\mu(i)},\ \omega\right\rangle - \left\langle y_{\sigma_\nu(i)},\ \omega\right\rangle\right|^2$$

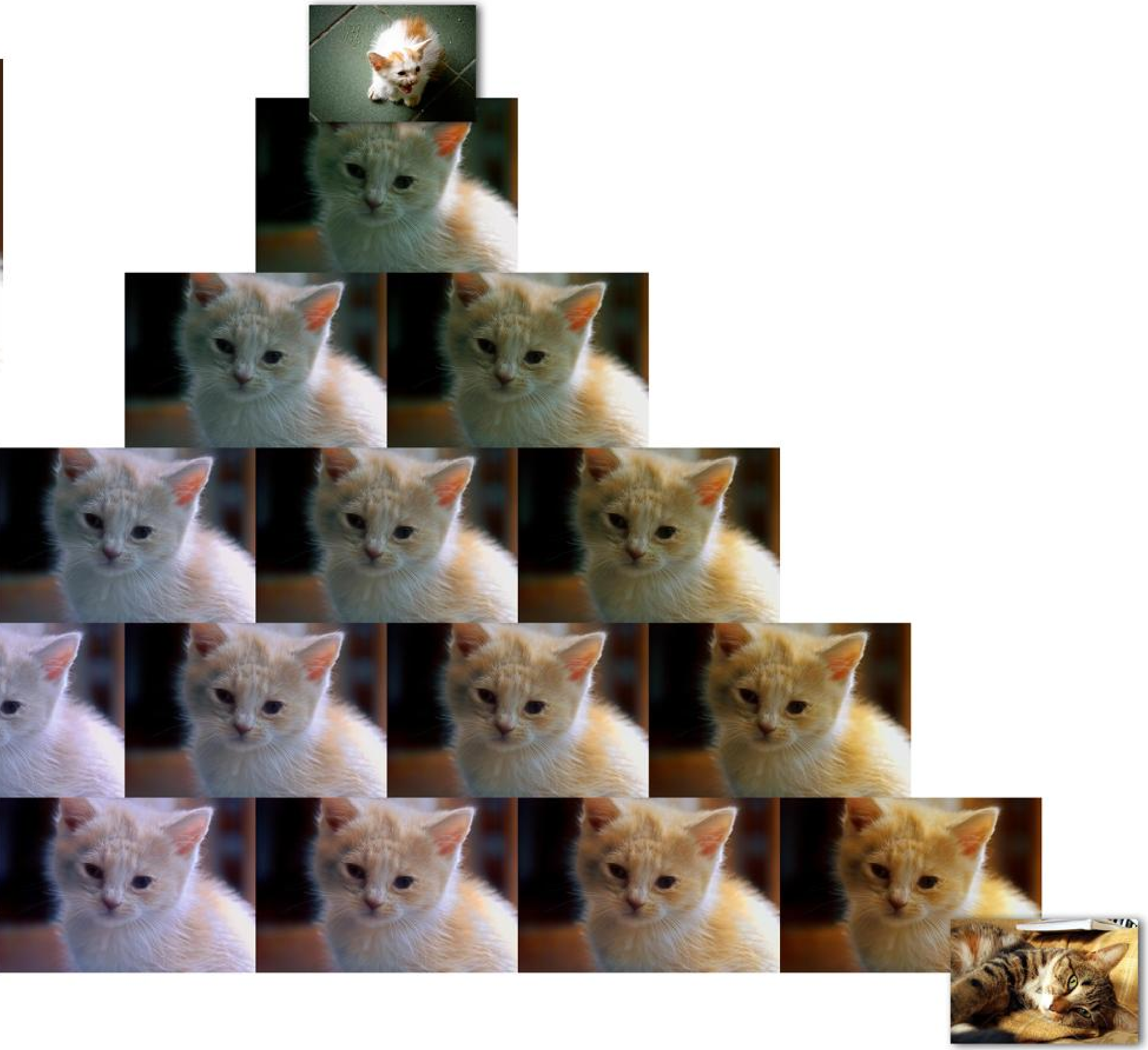
- or, to advect μ towards ν ("gradient flow")
  - Update μ by $x_{\sigma_\mu(i)} \leftarrow x_{\sigma_\mu(i)} + (\langle x_i,\ \omega\rangle - \langle y_i,\ \omega\rangle)\ \omega$

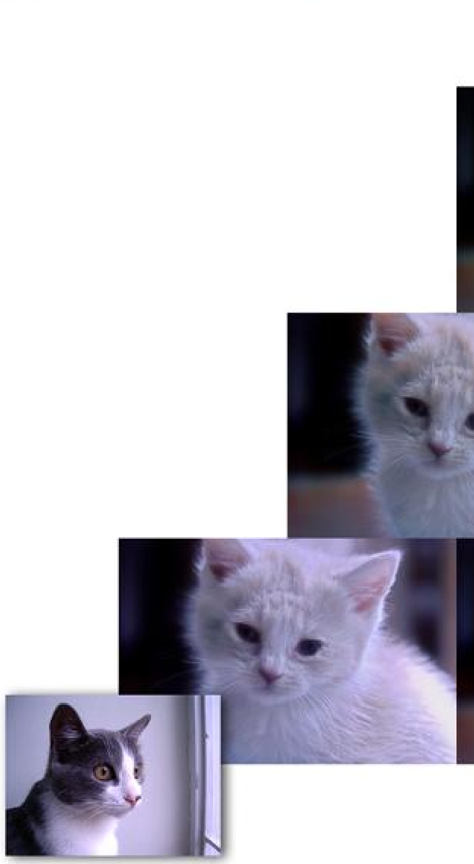
Wasserstein Barycenter and its Application to Texture Mixing. [Rabin et al. 2011]

Sliced Wasserstein Distance

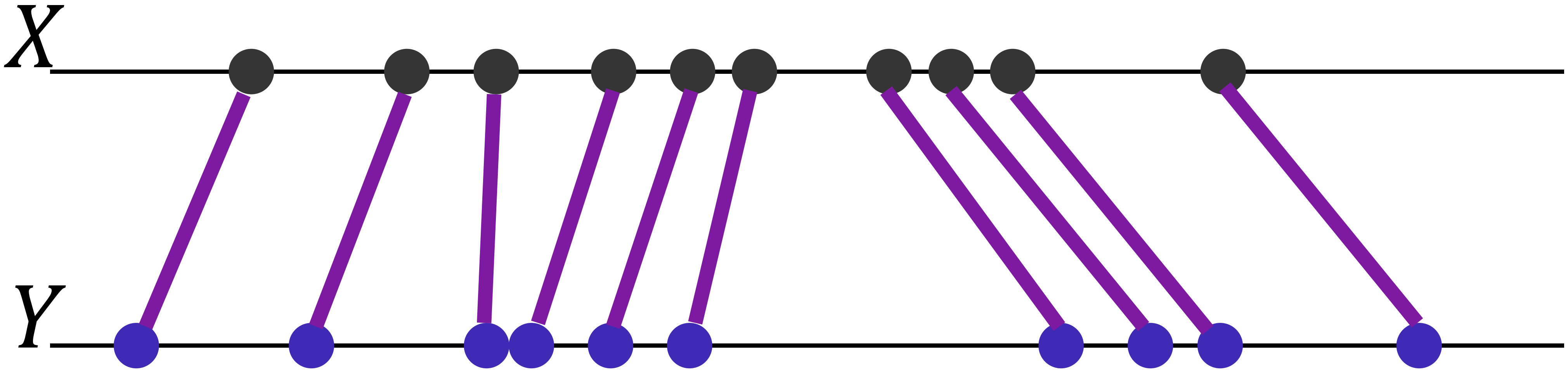
f

g





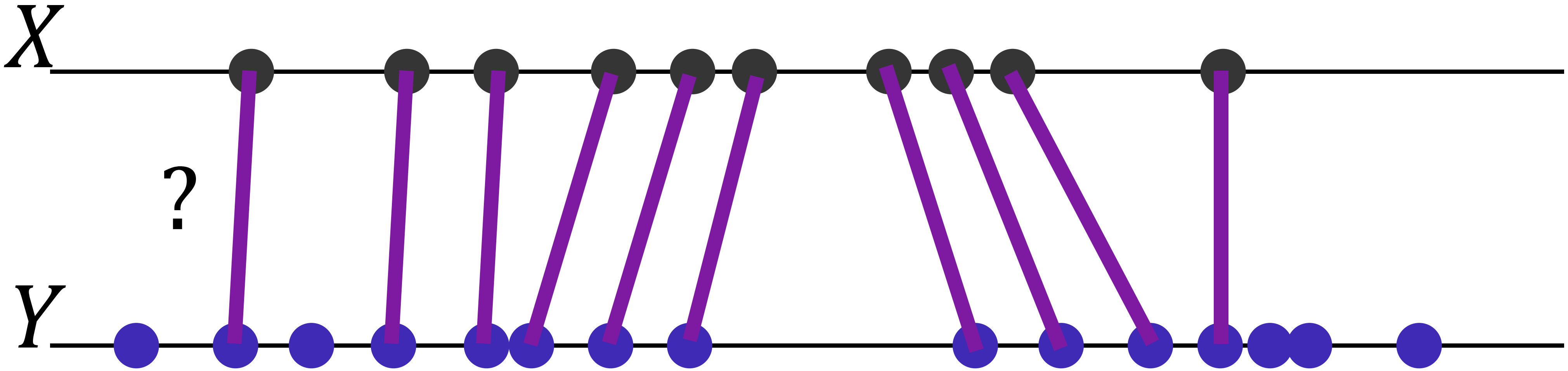
1-d Linear Assignment Problem is trivial*



*assuming the cost c is a convex function of |x-y|

Partial optimal assignment ?

=> Sliced Partial Optimal Transport, [Bonneel and Coeurjolly 2019]

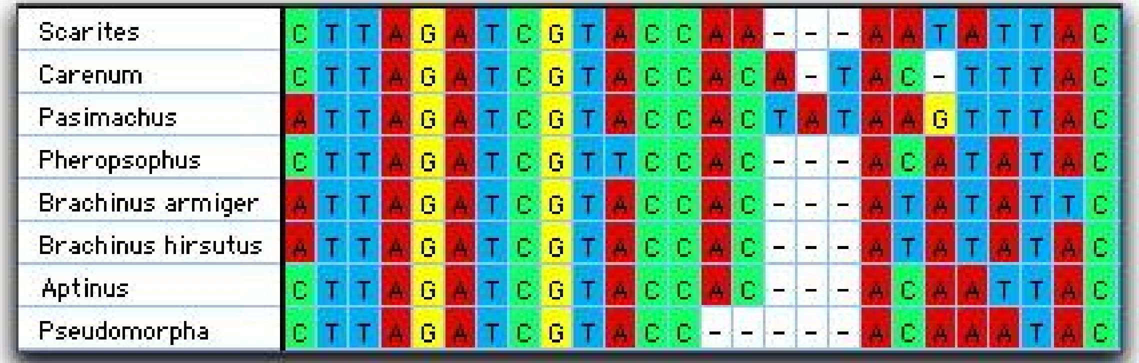


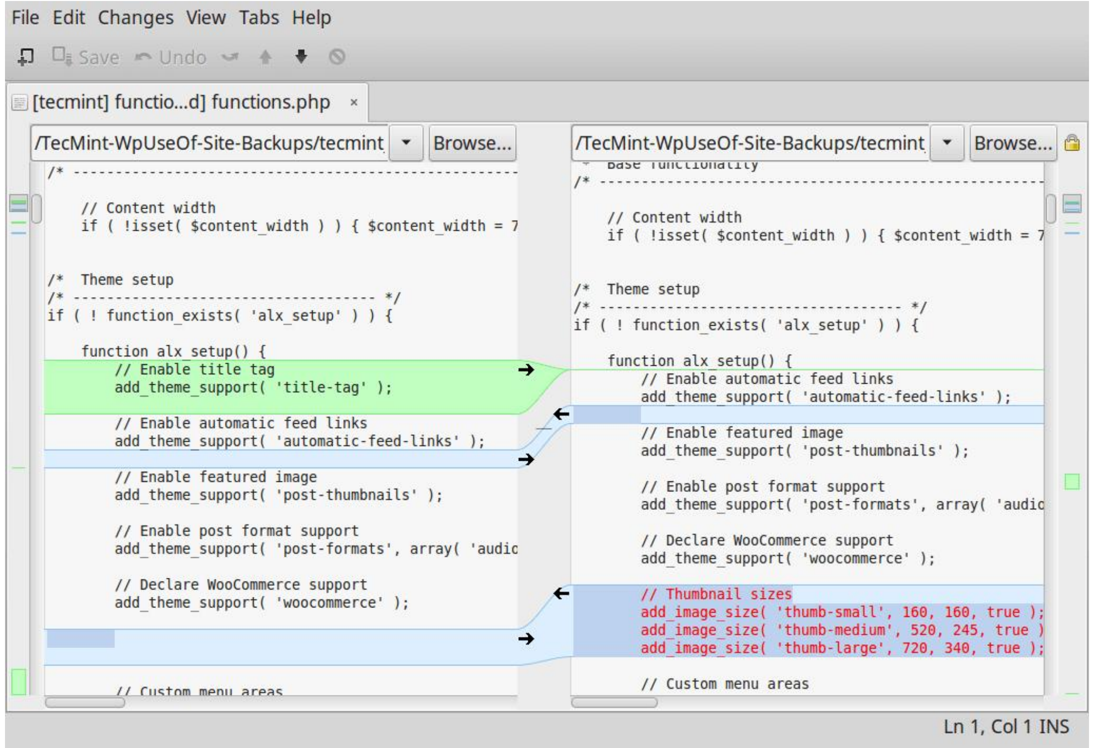
$$W(f,g) = \min \sum_{i,j} c_{i,j}\pi_{i,j} \quad \text{s.t.} \quad \sum_j \pi_{i,j} = 1, \quad \sum_i \pi_{i,j} \boxed{\leq} 1, \quad \pi_{i,j} \geq 0$$

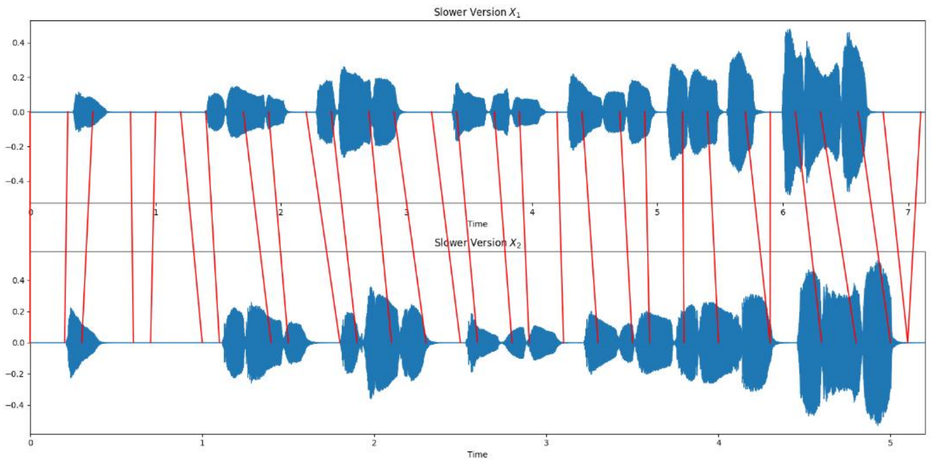
$$\min_{T\ \boxed{\text{injective}}} \sum_i c(x_i,\, y_{T(i)})$$

Similar problems

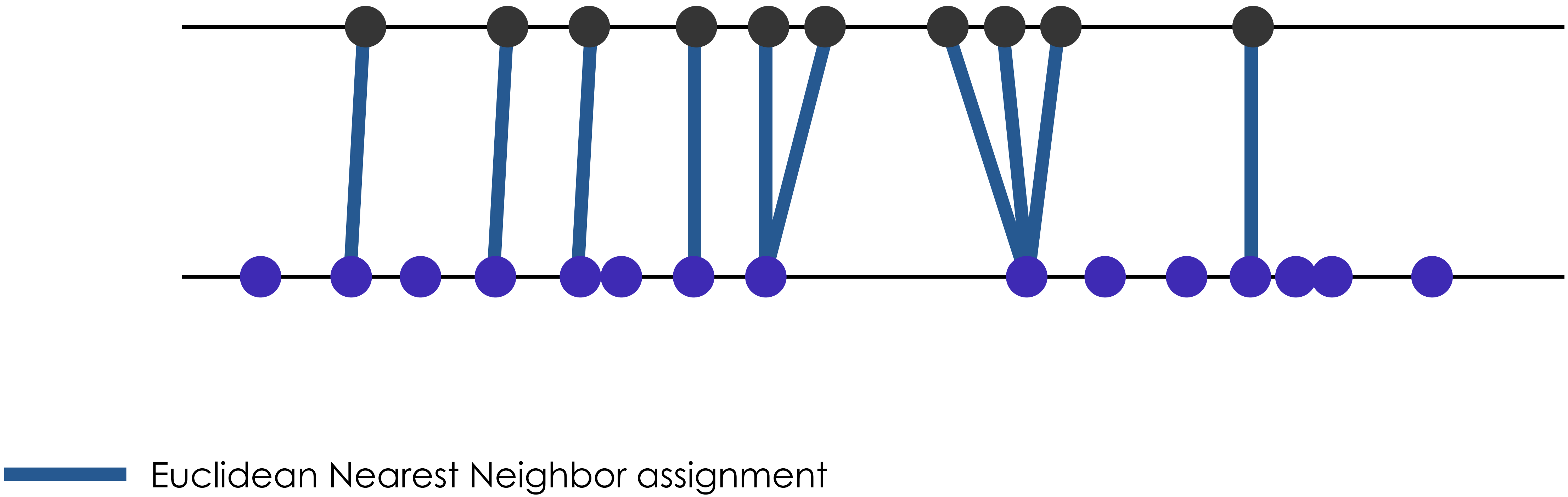
- DNA sequence alignment
- Text alignment
- Music synchronization
- …



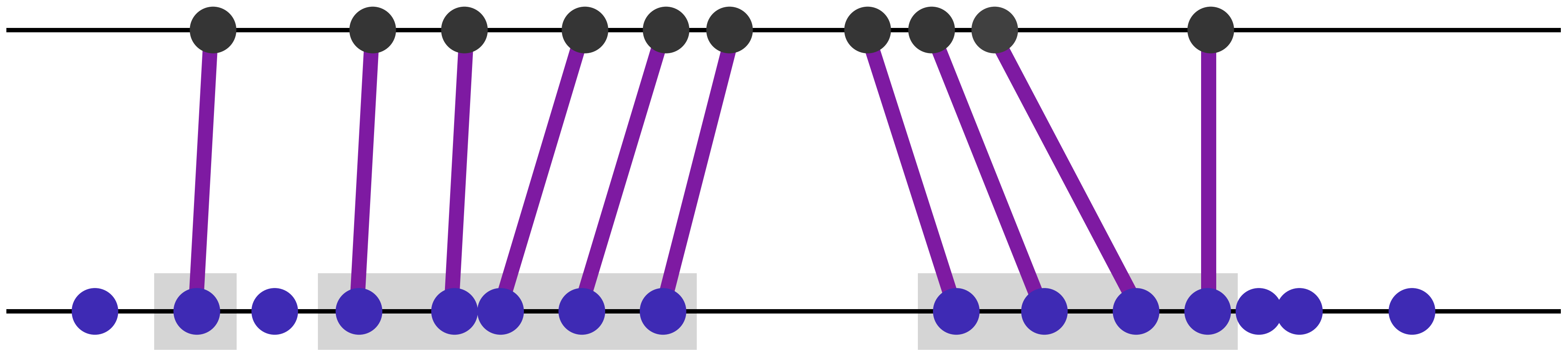




Quadratic time complexity algorithm (linear space)



Quadratic time complexity algorithm (linear space)



Euclidean Nearest Neighbor assignment

Optimal Transport assignment

Intervals of bijective assignments

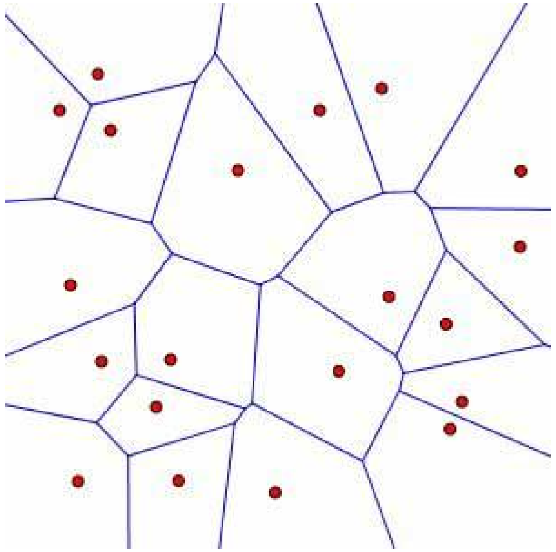
Semi-discrete optimal transport

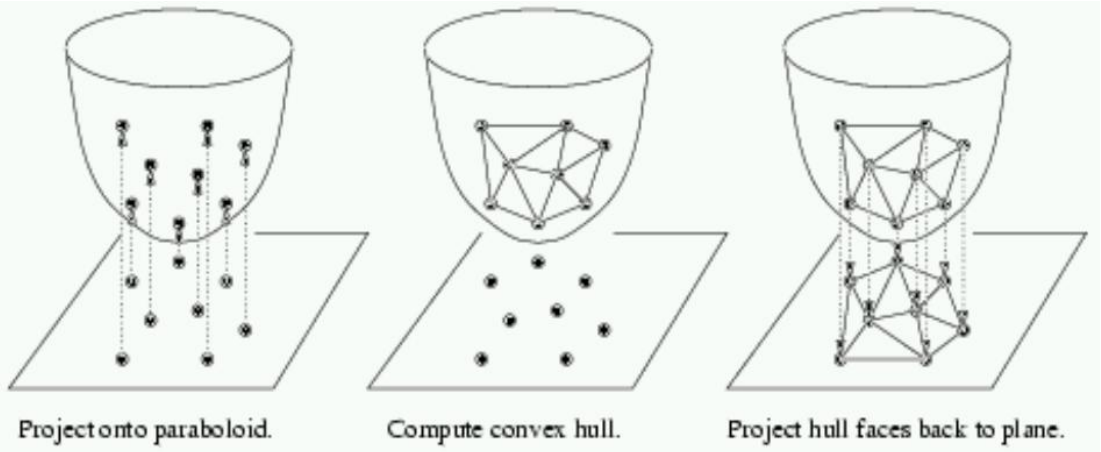
Voronoi diagram

- A partition such that each point x is assigned to its closest site x_i

$$\|x - x_i\|^2 \leq \|x - x_j\|^2 \quad \forall j$$

- The dual of a Delaunay triangulation: a triangulation of the sites such that no other site is encompassed by the circumcircle of a triangle
  - Also: convex hull of a parabolic lifting





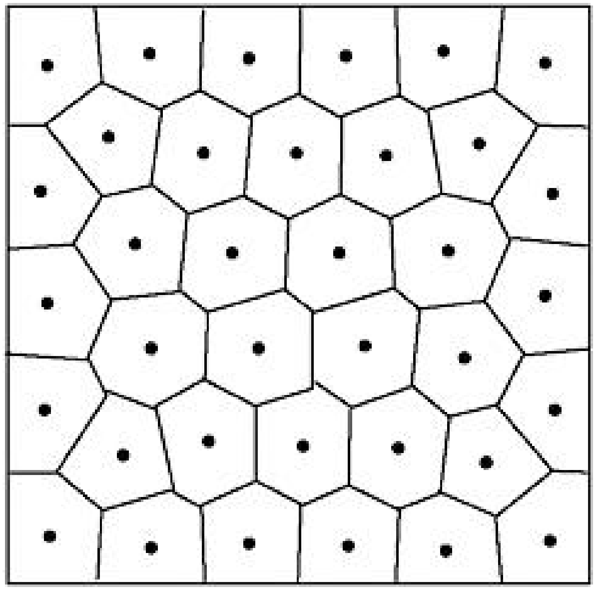
Centroidal Voronoi Diagram

- Can be defined as the solution to a least-square problem

$$\min \int_{Vor_i} \sum_i \|x - x_i\|^2 dx$$

Also says that the centroid of Vor_i is the site x_i

- Can be computed by:
  - A Lloyd clustering algorithm
  - A descent approach on the above energy

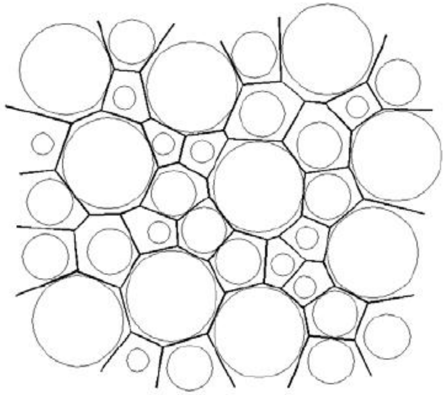
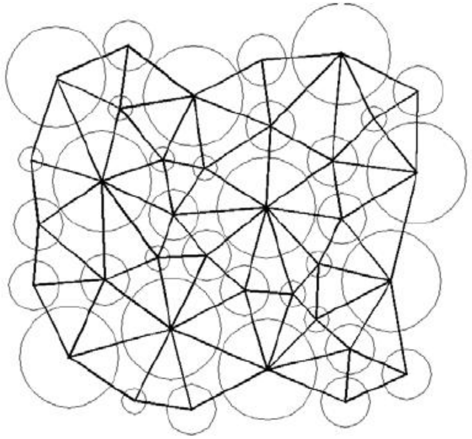


Power diagram (Laguerre diagram)

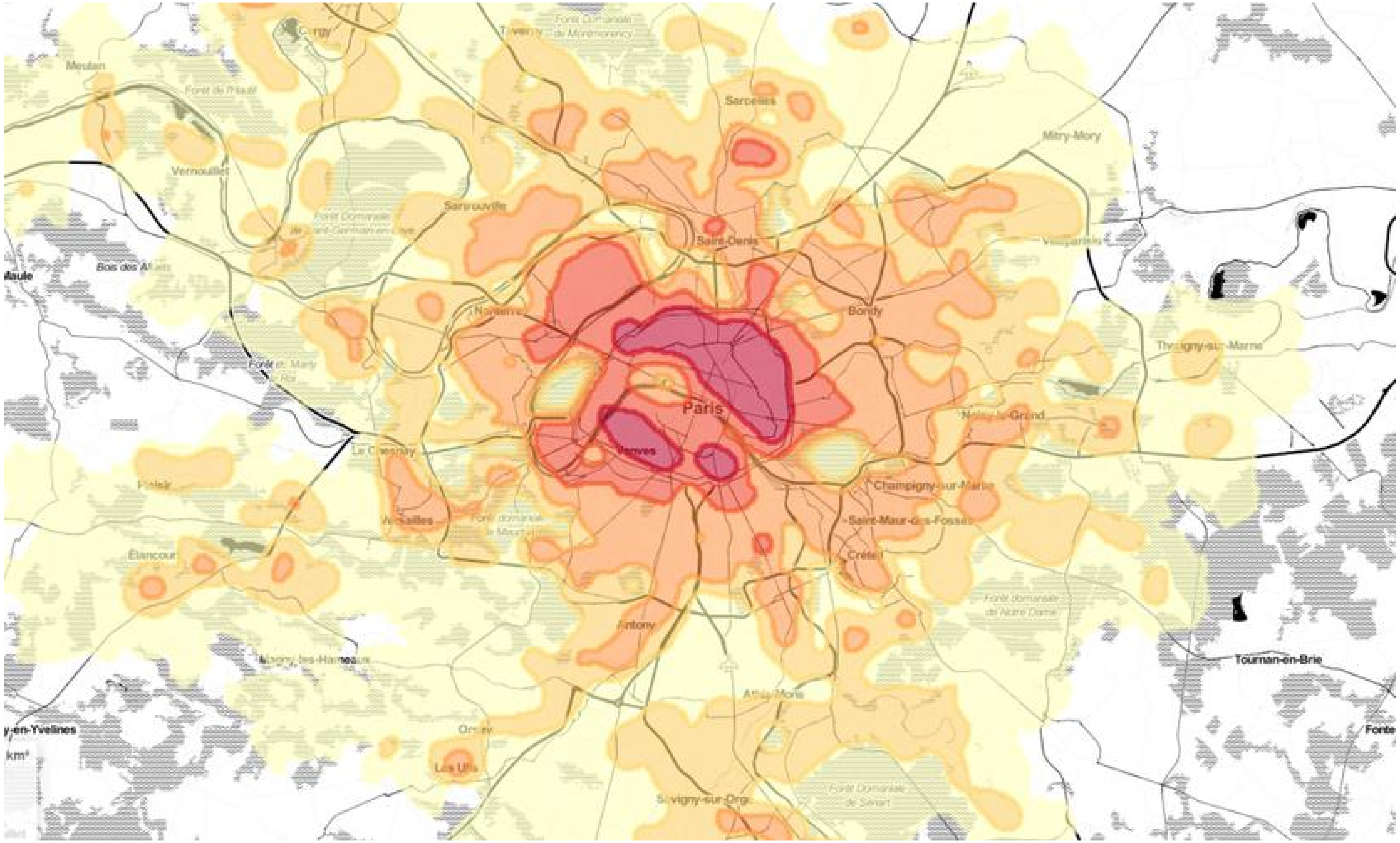
- A partition s.t. each point x is assigned to its closest site x_i with weight w_i

$$\|x - x_i\|^2 - w_i \leq \left\|x - x_j\right\|^2 - w_j \quad \forall j$$

- Can be computed by lifting a Voronoi diagram
  - Consider site coordinates $x_i' = (\, x_i \,;\, \sqrt{c - w_i})$ for large constant c ; $x' = (x\,;0)$
  - Then $\|x' - x_i'\|^2 \leq \left\|x' - x_j'\right\|^2 \;\forall j$
- Any partition into convex polyhedral cells is a power diagram of some sites

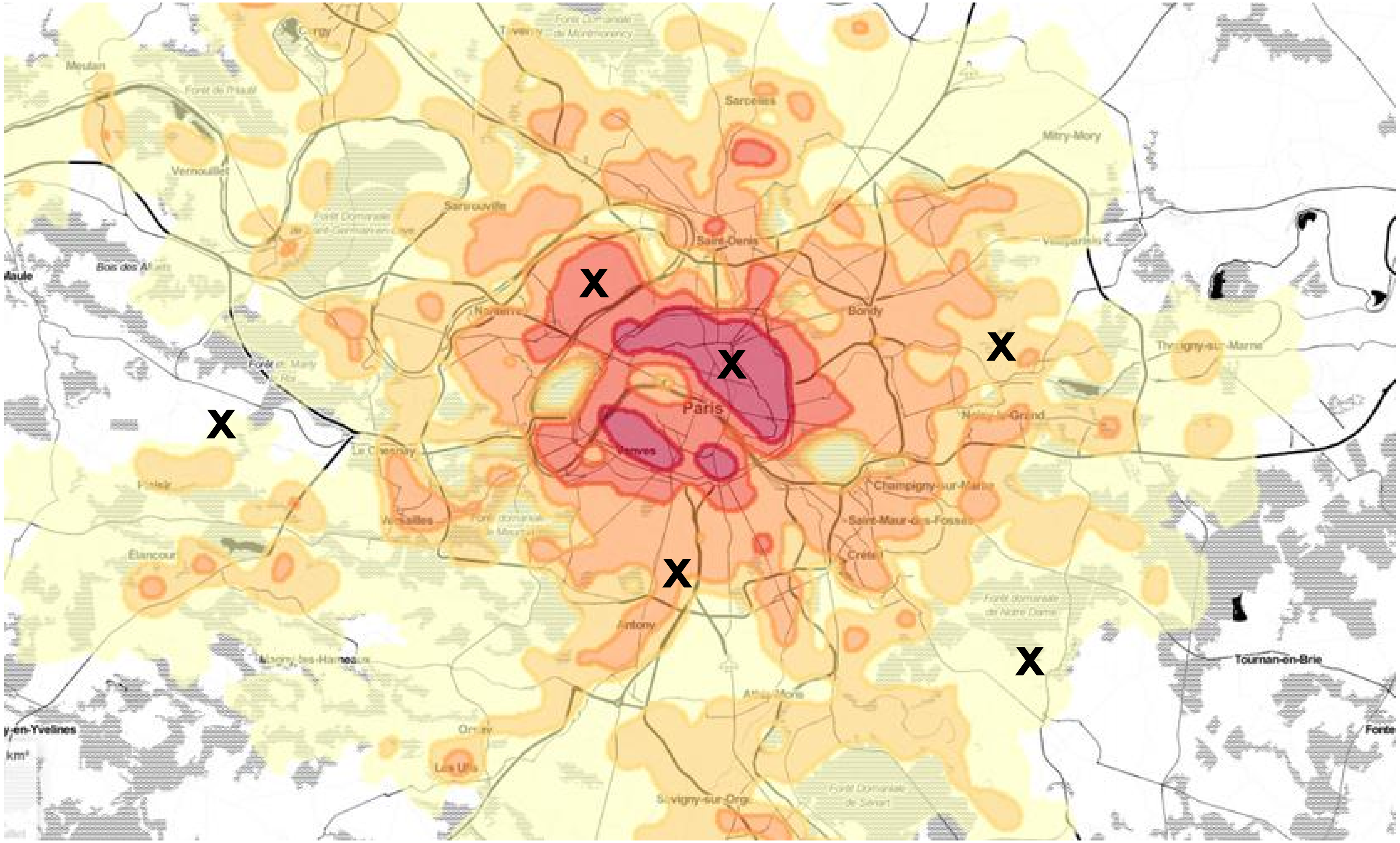



Semi-discrete Optimal Transport



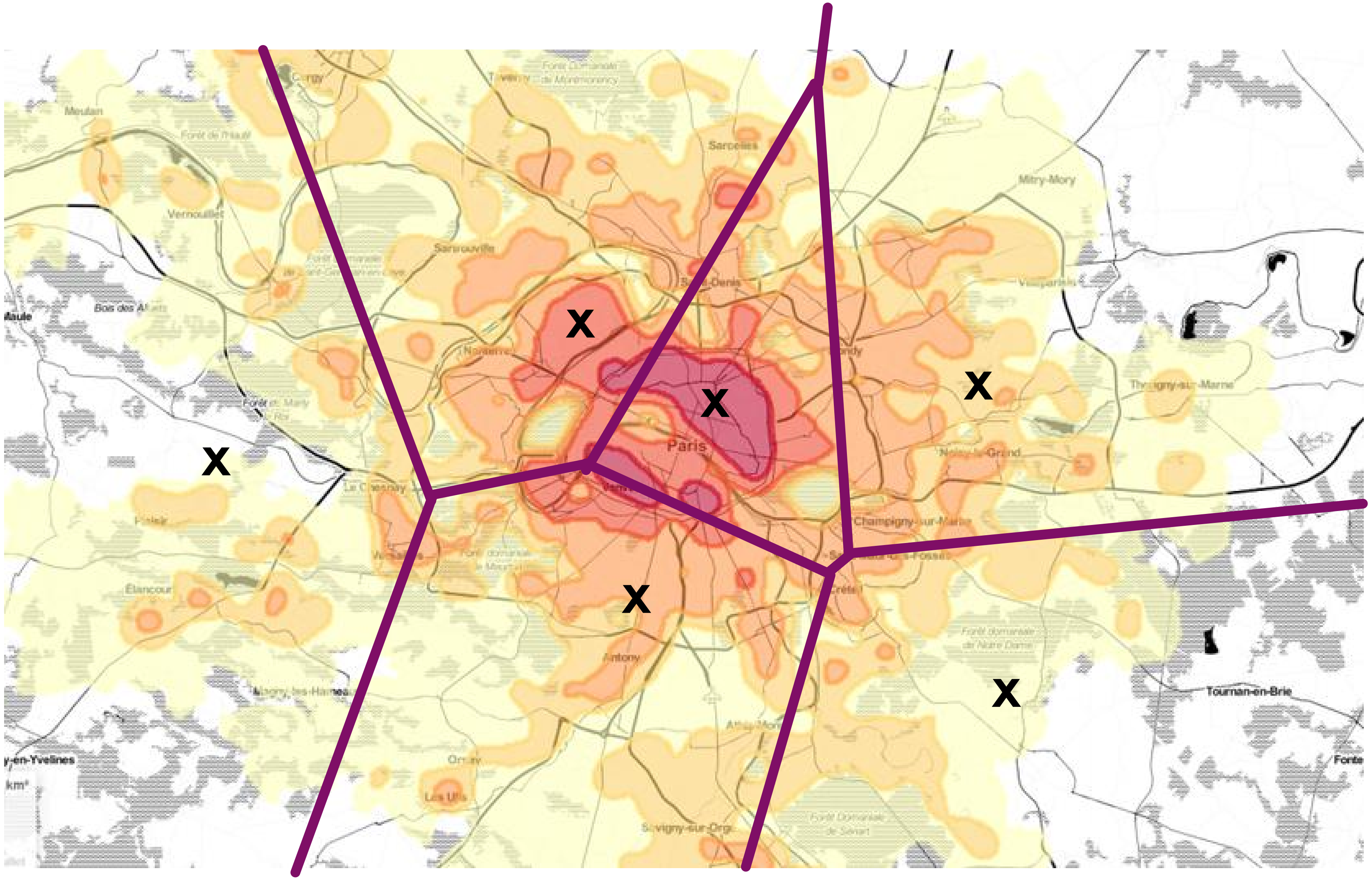
Population density f

Semi-discrete Optimal Transport



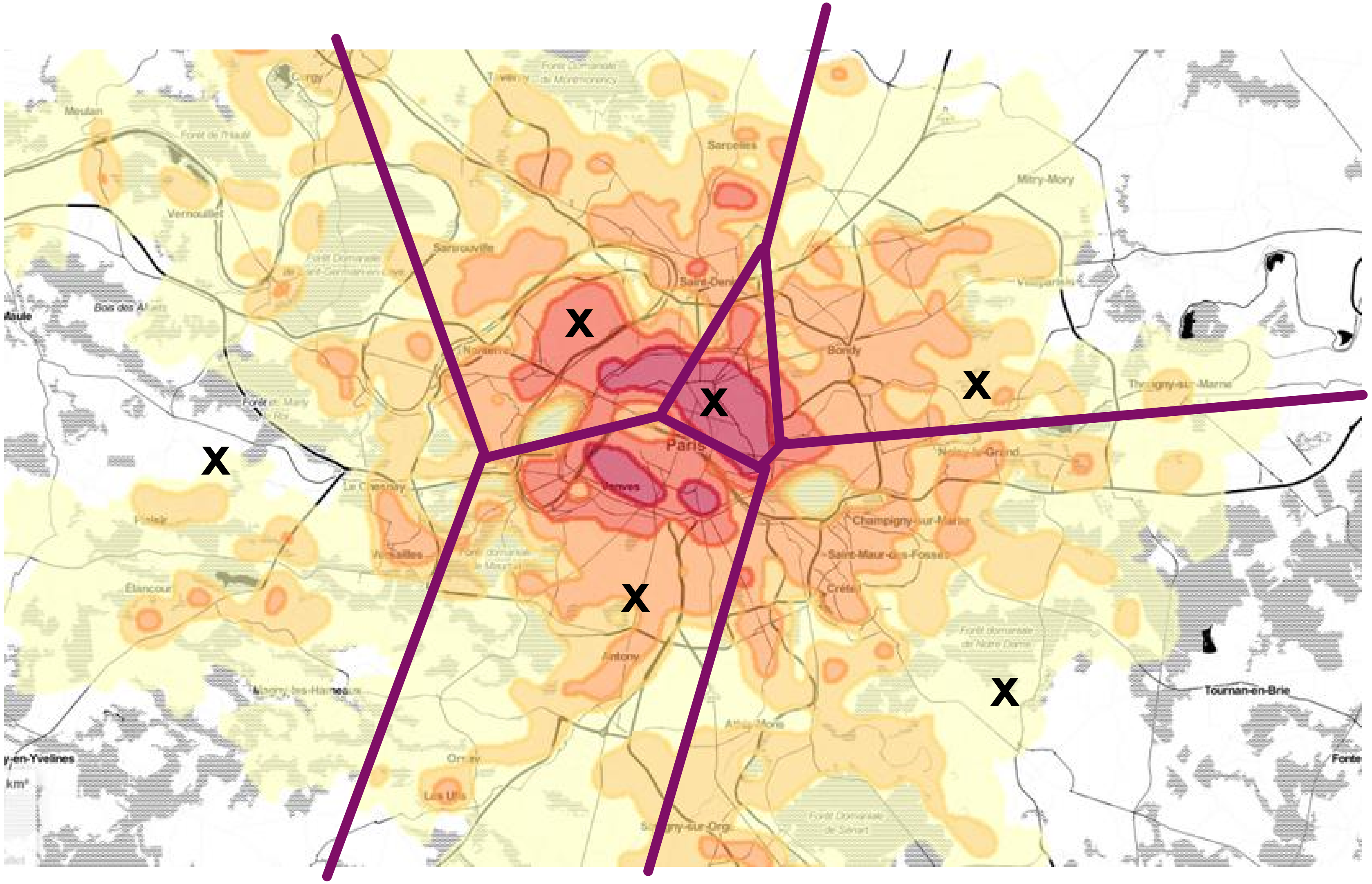
Set of bakeries, factories, …?

Semi-discrete Optimal Transport



No constraint on production: population go to their nearest bakery/factory/… regardless of populat

Semi-discrete Optimal Transport



Limited production: population go to the nearest bakery/factory **with sufficient production**!

Semi-discrete Optimal Transport

Limited production: population go to the nearest bakery/factory **with sufficient production**!

Back to optimal transport

- Optimal transport (Monge version) :

$$\min \int \|x - T(x)\|^2 \, d\mu(x)$$

Considering μ is continuous with density ρ

$$\min \int \|x - T(x)\|^2 \, \rho(x) dx$$

Considering ν (the target measure) discrete: $\nu = \sum \lambda_p \delta_p$

The mass preservation constraint is:

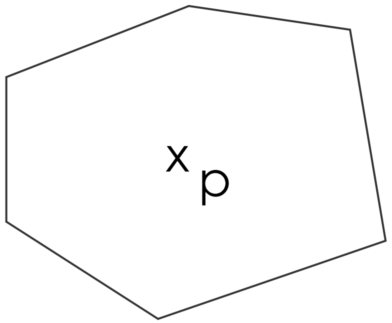
$$\lambda_p = \int_{T^{-1}(\{p\})} \rho(x) dx$$

A Multiscale Approach to Optimal Transport [Mérigot 2011]
Minkowski-Type Theorems and Least-Squares Clustering [Aurenhammer et al. 98]

Back to optimal transport

- In this case : $T^{-1}(\{p\}) = Vor^W(p)$

a power cell for some weight w_p

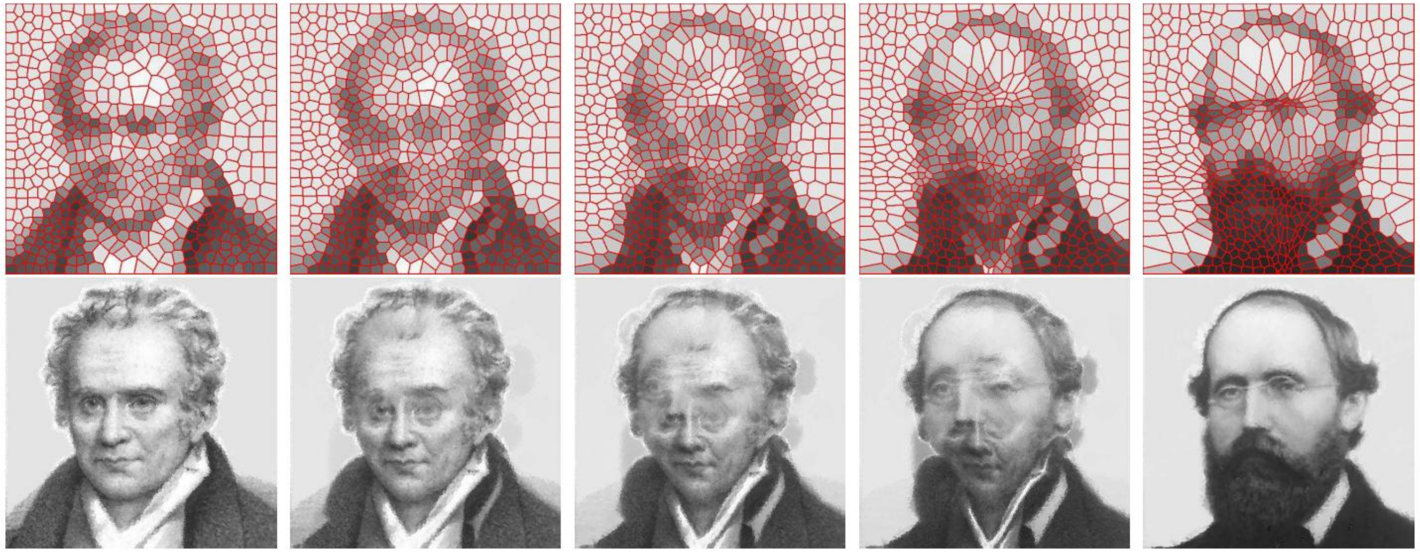


- This determines a partition, so Monge problem is:

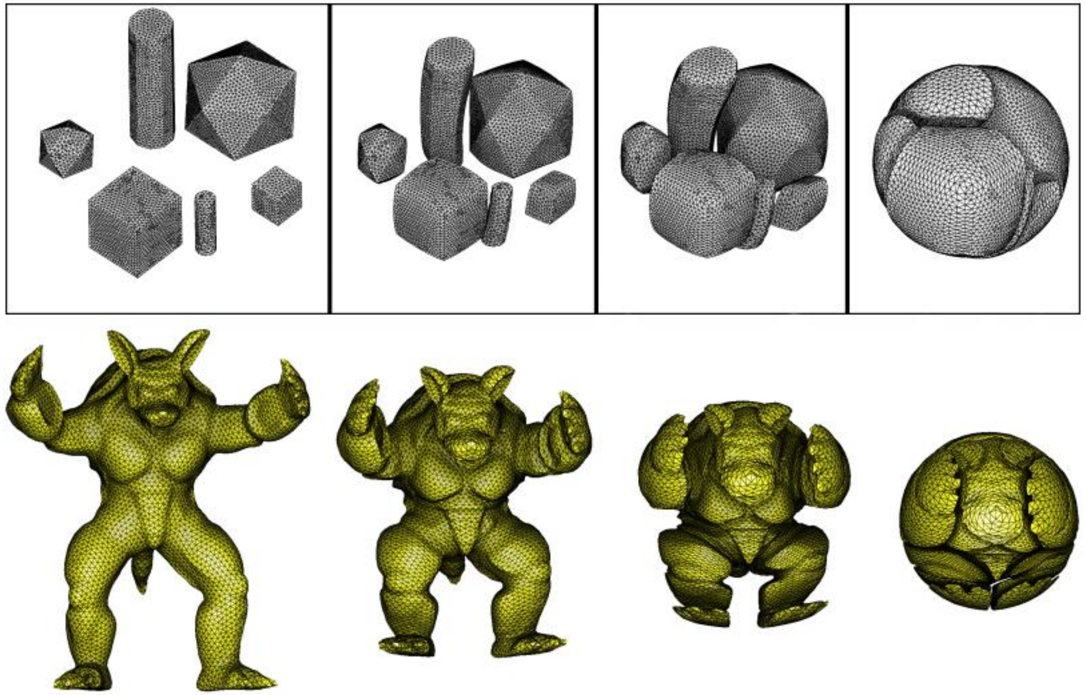
$$\min \sum_p \int_{Vor^W(p)} \|x-p\|^2 \, \rho(x) \, dx$$

- Idea: optimize weights w for each site to grow/shrink power cells until $\lambda_p = \int_{T^{-1}(\{p\})} \rho(x) dx$
- Gradient of appropriate functional given by $\frac{\partial \phi}{\partial w(p)}(w) = \lambda_p - \int_{Vor^W(p)} \rho(x) \, dx$

Back to optimal transport



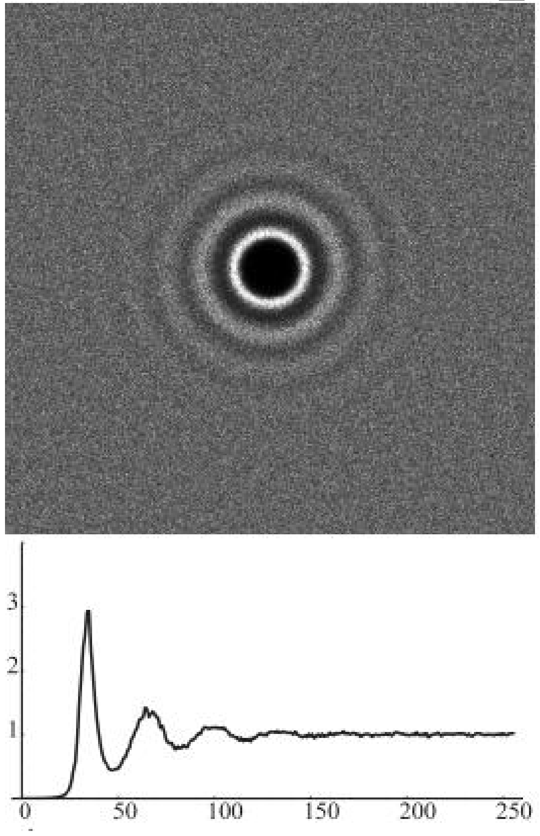
A Multiscale Approach to Optimal Transport [Mérigot 2011]



A Numerical Algorithm for L2 Semi-discrete Optimal Transport in 3D [Lévy 2015]

Application

- Also optimizes for the locations p



Blue Noise through Optimal Transport [de Goes et al. 2012]

Regularized optimal transport

The Sinkhorn algorithm

- Kantorovich optimal transport: $\min_m \sum_i \sum_j c_{i,j}\, m_{i\to j}$ with constraints
- Rewritten as :

$$\min_{M\in\mathcal{U}(r,c)} \langle C,\ M\rangle$$

with $\mathcal{U}(r,c)$ matrices whose rows sum to r and columns to c

- Idea: consider instead

$$\min_{M\in\mathcal{U}(r,c)} \langle C,\ M\rangle - \epsilon E(M)$$

where $E(M) = -\sum M_{ij}(\log\left(M_{ij}\right) - 1)$ is the entropy, ϵ a small constant

Iterative Bregman Projections for Regularized Transportation Problems [Benamou et al. 2014]
Sinkhorn Distances: Lightspeed Computation of Optimal Transport [Cuturi 2013]

The Sinkhorn algorithm

$$\min_{M \in \mathcal{U}(r,c)} \langle C,\ M \rangle - \epsilon E(M)$$

- Can be rewritten as a projection:

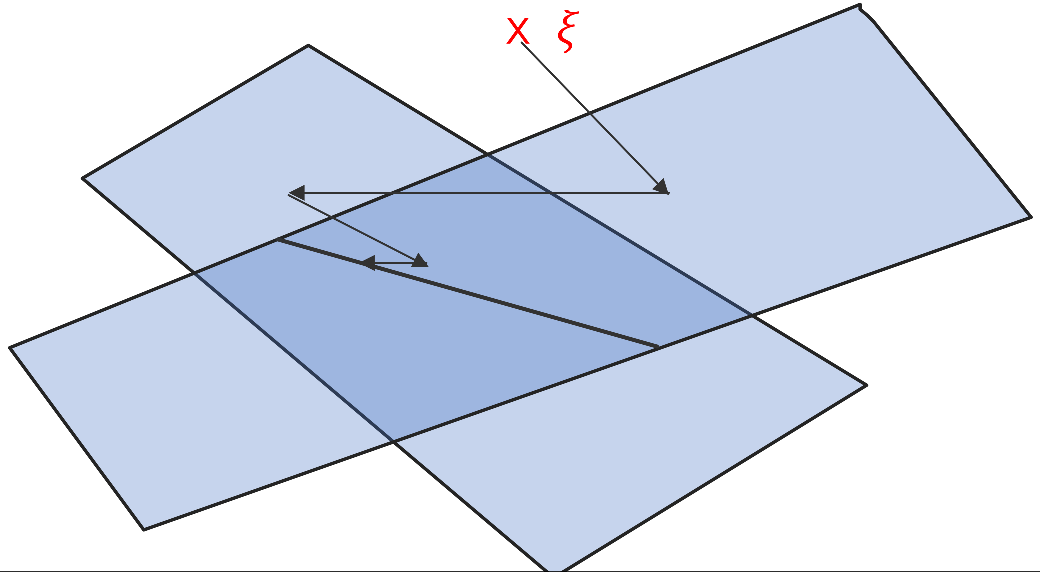
$$\min_{M \in \mathcal{U}(r,c)} KL(M,\ \xi)$$

where $\xi = \exp\left(-\frac{C}{\epsilon}\right)$ and $KL(M,\ \xi) = \sum M_{ij}\left(\log\left(\frac{M_{ij}}{\xi_{ij}}\right) - 1\right)$ the Kullback-Leibler divergence

The Sinkhorn algorithm

$$\min_{M \in \mathcal{U}(r,c)} KL(M,\, \xi)$$

- This is a projection on the intersection of two affine constraints, due to $\mathcal{U}(r, c)$
- We can thus apply Bregman projections: we iteratively project on each constraint



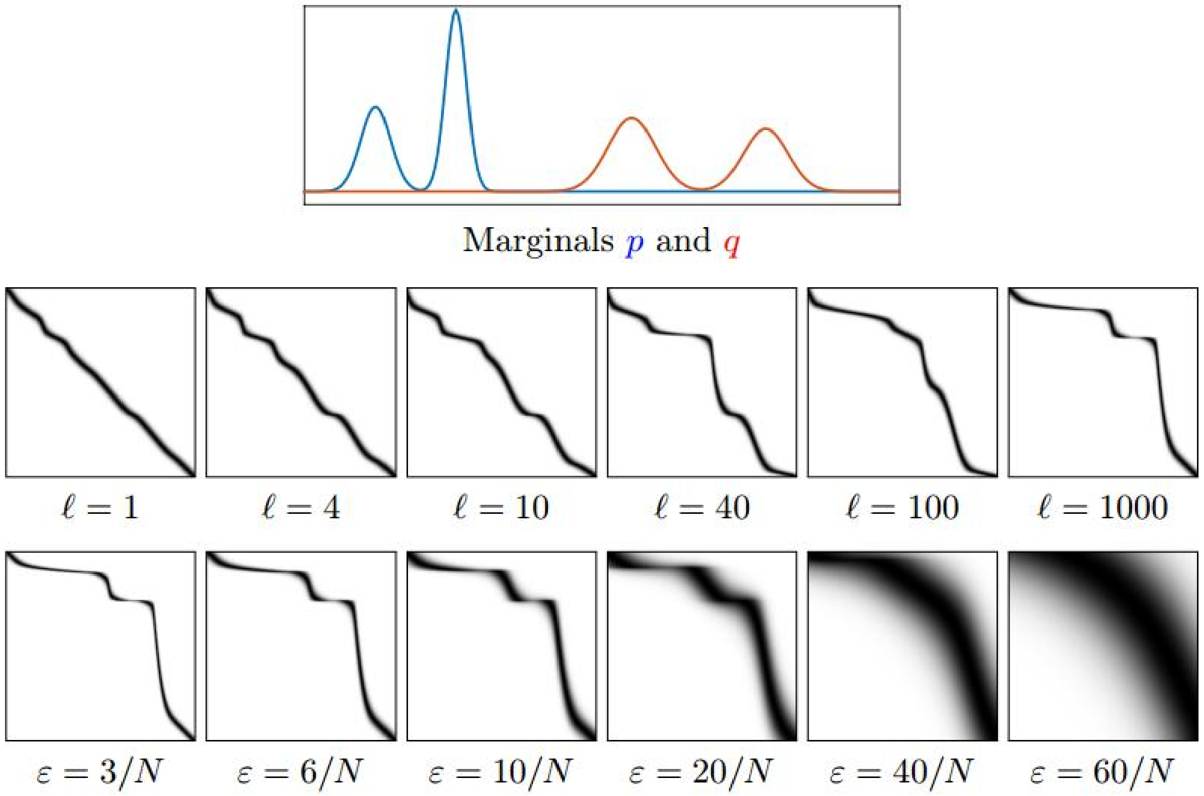
The Sinkhorn algorithm

- Projecting on constraints:
  - Constraints: $\sum_i M_{ij} = r_j$ and $\sum_j M_{ij} = c_i$
  - $M'_{ij} = \frac{M_{ij}}{\sum_i M_{ij}} . r_j$ and $M'_{ij} = \frac{M_{ij}}{\sum_j M_{ij}} . c_i$ corresponds to projection with KL
  - Row/column scaling
  - Corresponds to left/right multiplying M by diagonal matrix

The Sinkhorn algorithm

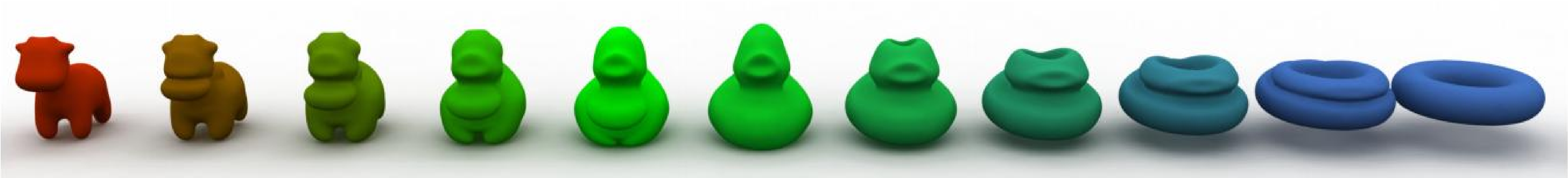
- We can thus apply Bregman projections: we iteratively project on each constraint
- We obtain the algorithm:

- $u^{(n)} = \frac{f}{\xi\, v^{(n)}}$
- $v^{(n+1)} = \frac{g}{\xi^T\, u^{(n)}}$
- $M = diag\big(u^{(n)}\big)\xi\; diag(v^{(n)})$



The Sinkhorn algorithm

- We realize that $\xi\, v^{(n)}$ can be computed efficiently
  - E.g., if $c(x,y) = \|x-y\|^2$, $\xi_{ij} = \exp\left(-\frac{\|x_i - x_j\|^2}{\epsilon}\right)$
  - Then $\xi\, v^{(n)}$ is just a Gaussian convolution
  - So, it is a separable operator, and efficiently done in high-dimension

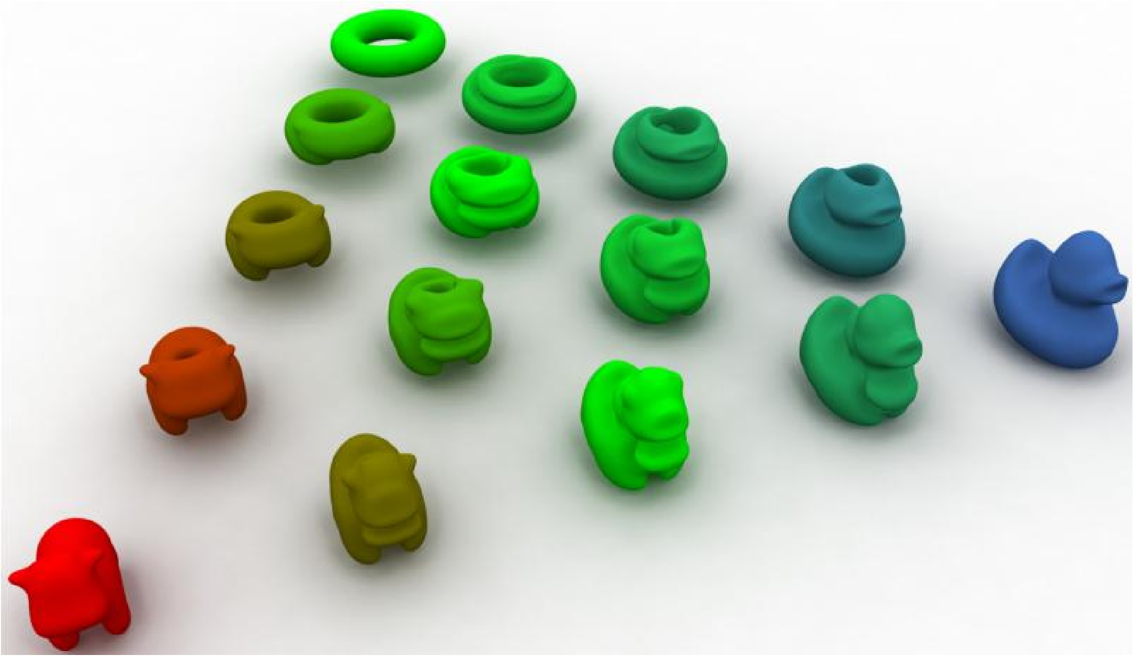


Convolutional Wasserstein Distances: Efficient Optimal Transportation on Geometric Domains [Solomon et al. 2015]

The Sinkhorn algorithm

- Generalized to compute displacement interpolation and barycenters

  - $b_s^{(0)} = 1 \;\; \forall s$
  - for $\ell = 0 \ldots L$
    - $a_s^{(\ell)} = \frac{p_s}{K\, b_s^{(l-1)}} \quad \forall s$
    - $p(\lambda) = \prod_s \left(K^T a_s^{(\ell)}\right)^{\lambda_s}$
    - $b_s^{(\ell)} = \frac{p(\lambda)}{K^T\, a_s^{(\ell)}} \qquad \forall s$

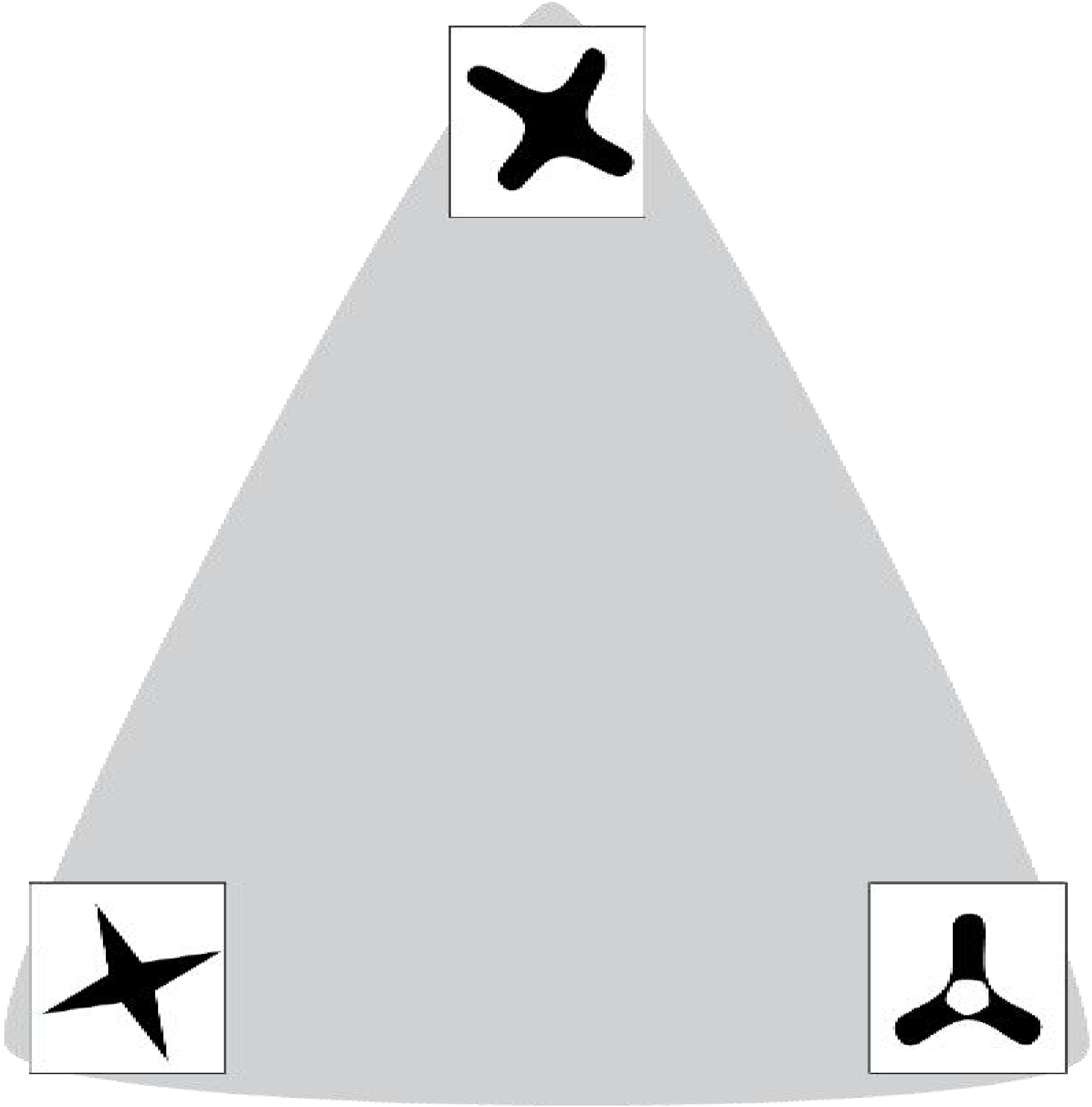


Wasserstein Barycentric Coordinates: Histogram Regression Using Optimal Transport

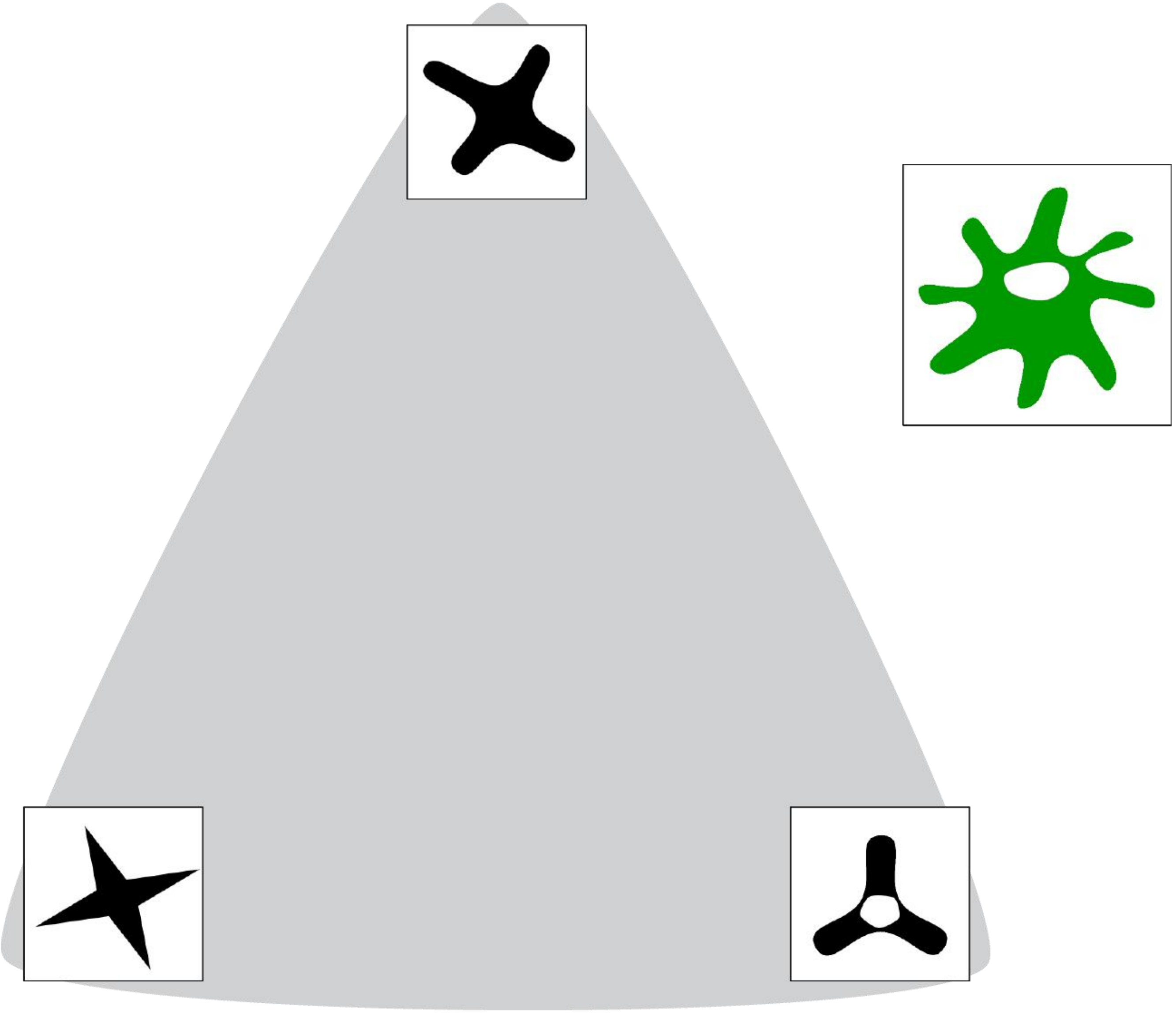
N. Bonneel, G. Peyré, M.Cuturi

SIGGRAPH 2016

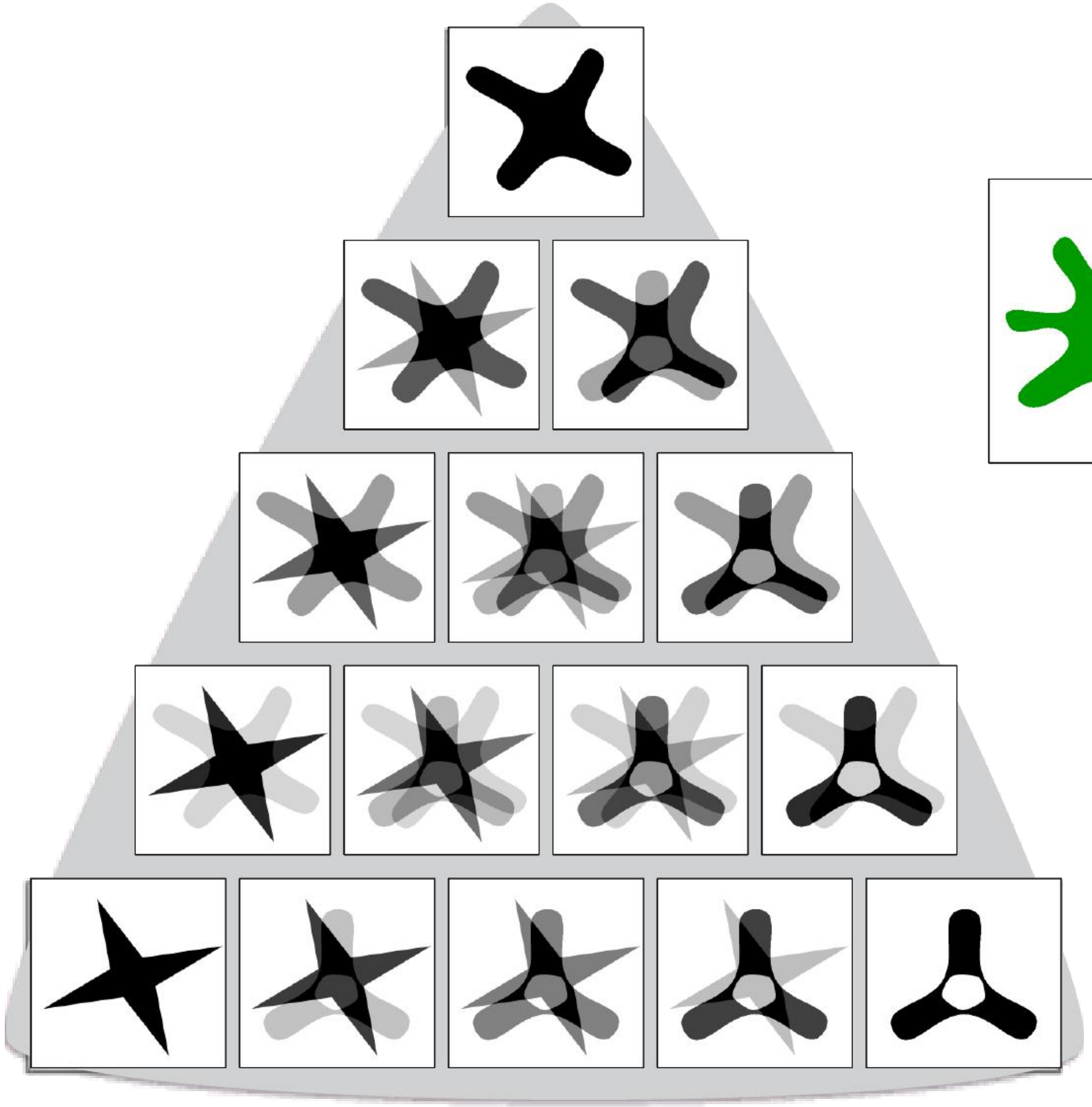
Barycentric coordinates

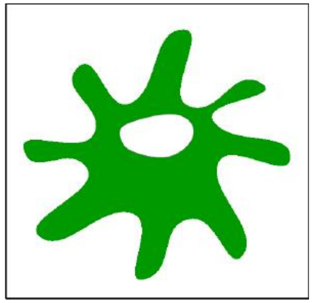


Barycentric coordinates

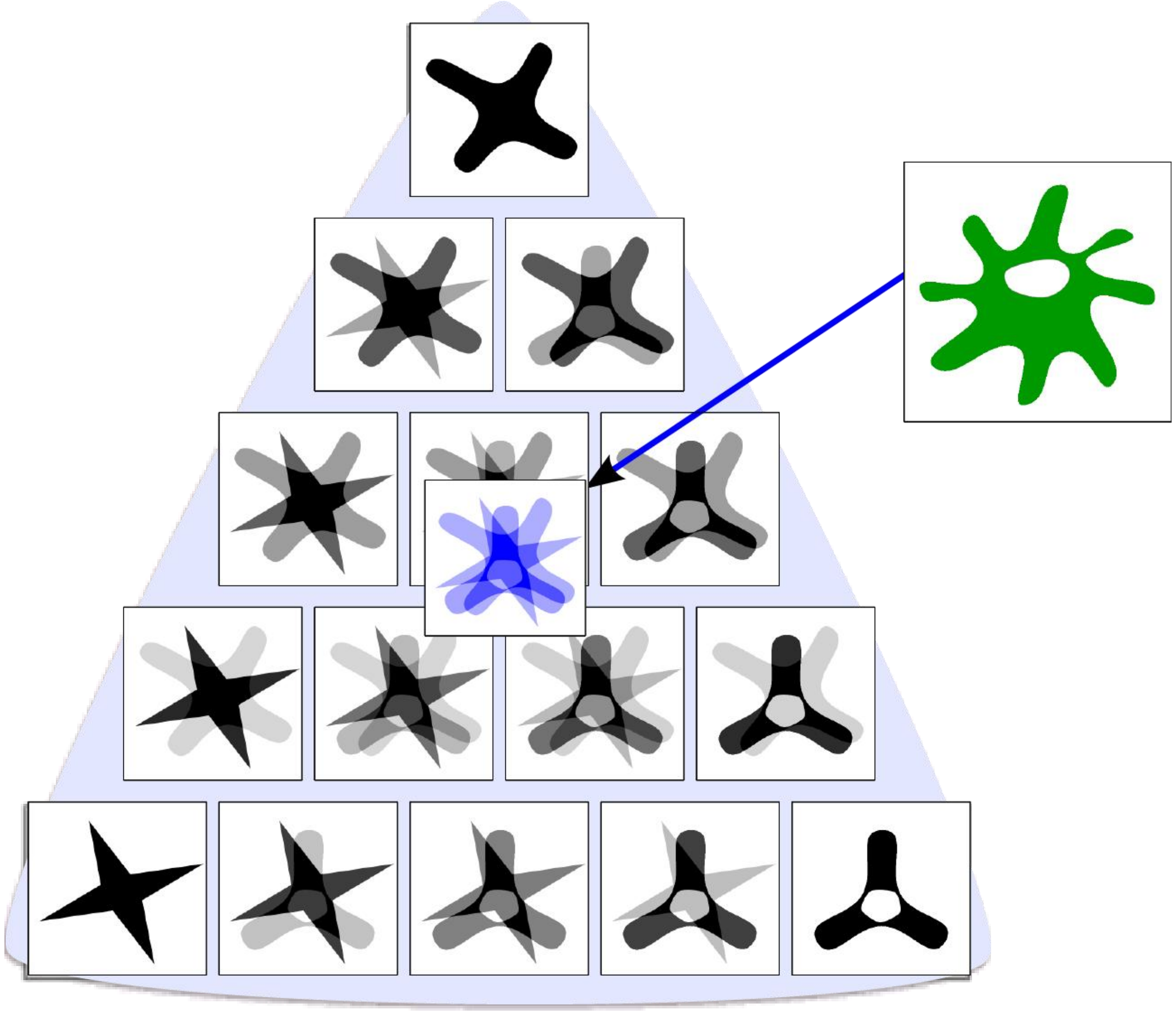


Barycentric coordinates

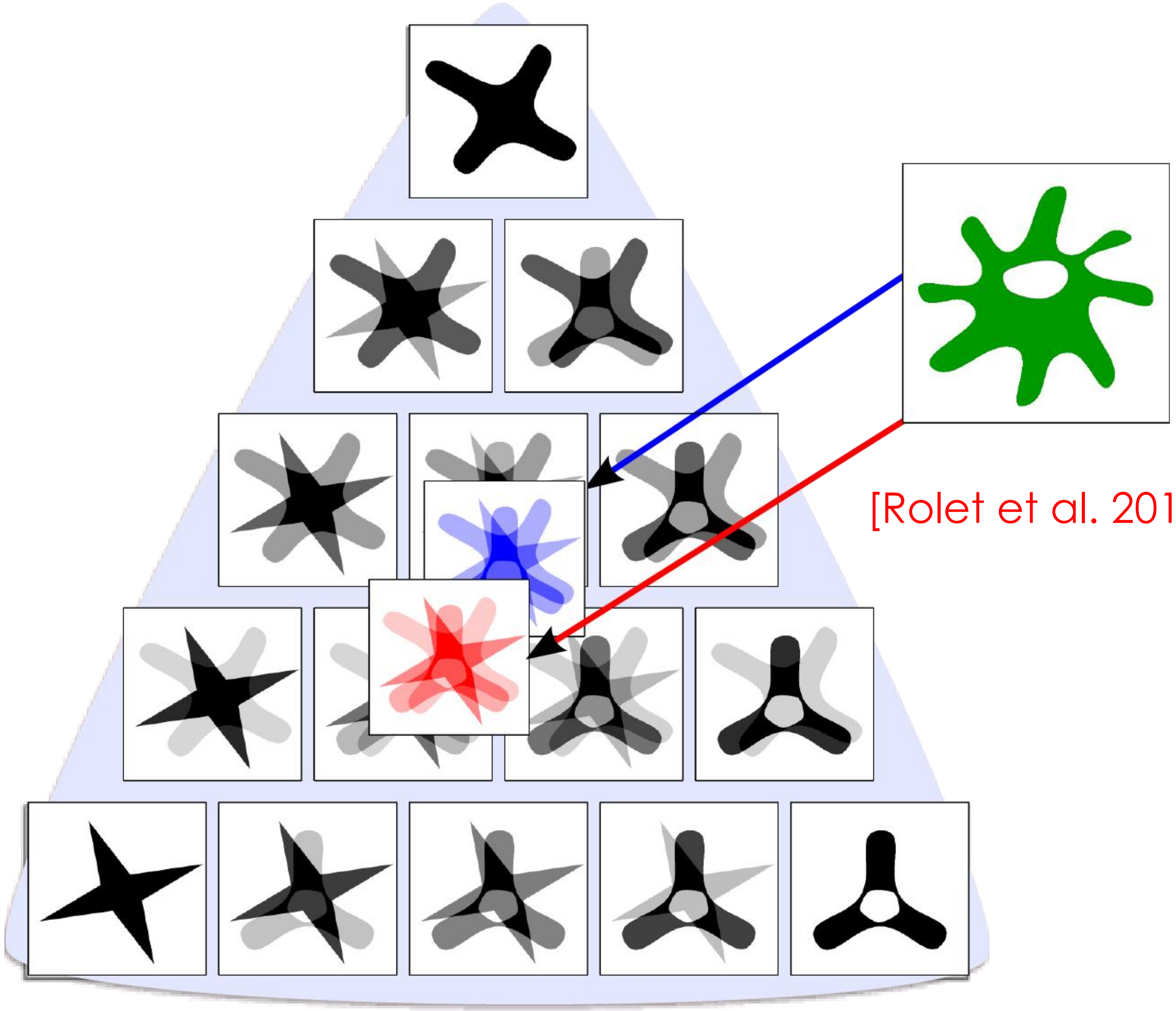




Barycentric coordinates

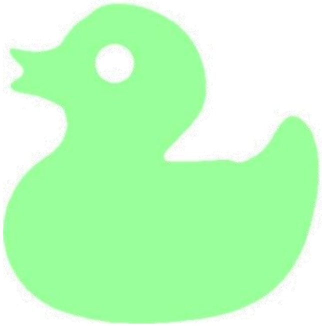


Barycentric coordinates

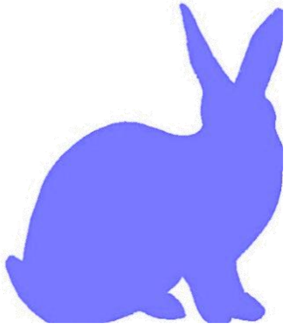


[Rolet et al. 2016]

Optimal Transport

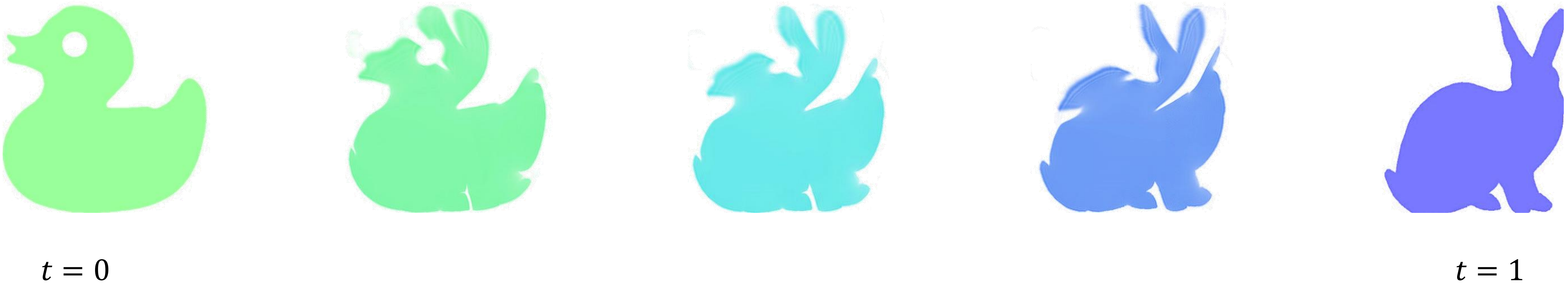


$t = 0$



$t = 1$

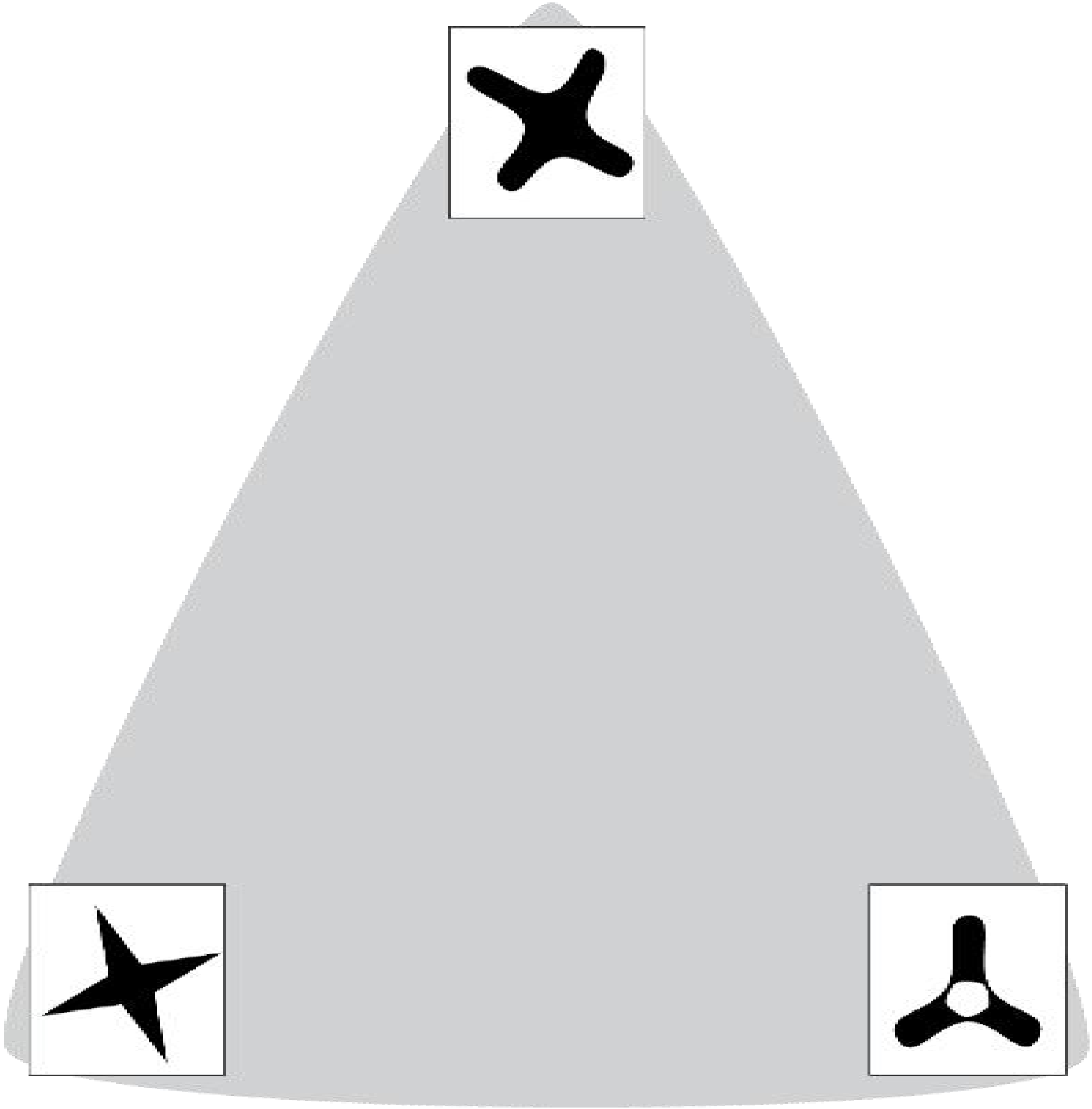
Optimal Transport



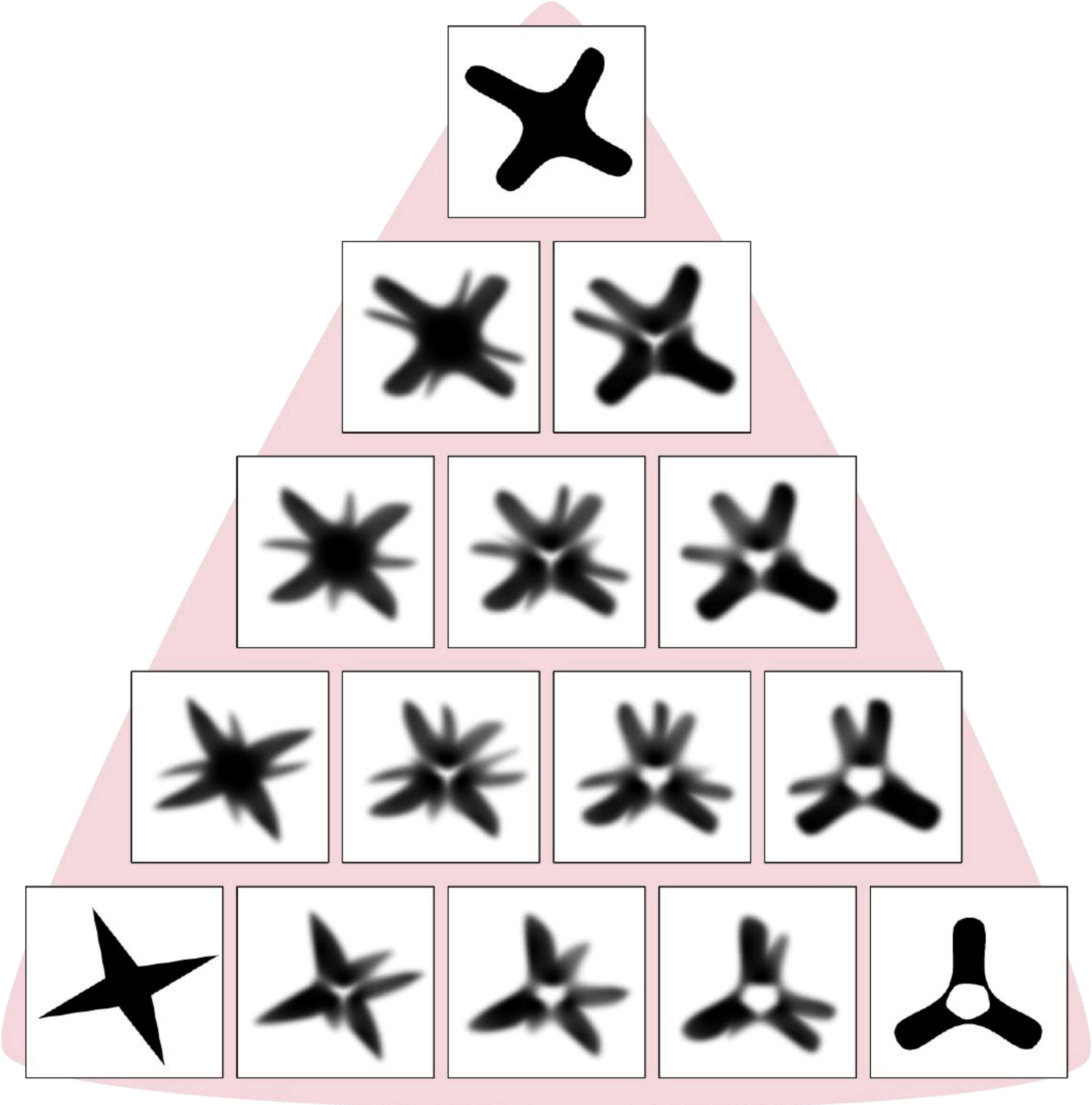
$$W(f, g) = \min \sum\sum \left\| x_i - x_j \right\|^2 m_{ij}$$

$$\text{s.t.} \qquad m_{ij} \geq 0 \ ; \ \textstyle\sum_i m_{ij} = g(x_j) \ ; \ \textstyle\sum_j m_{ij} = f(x_i)$$

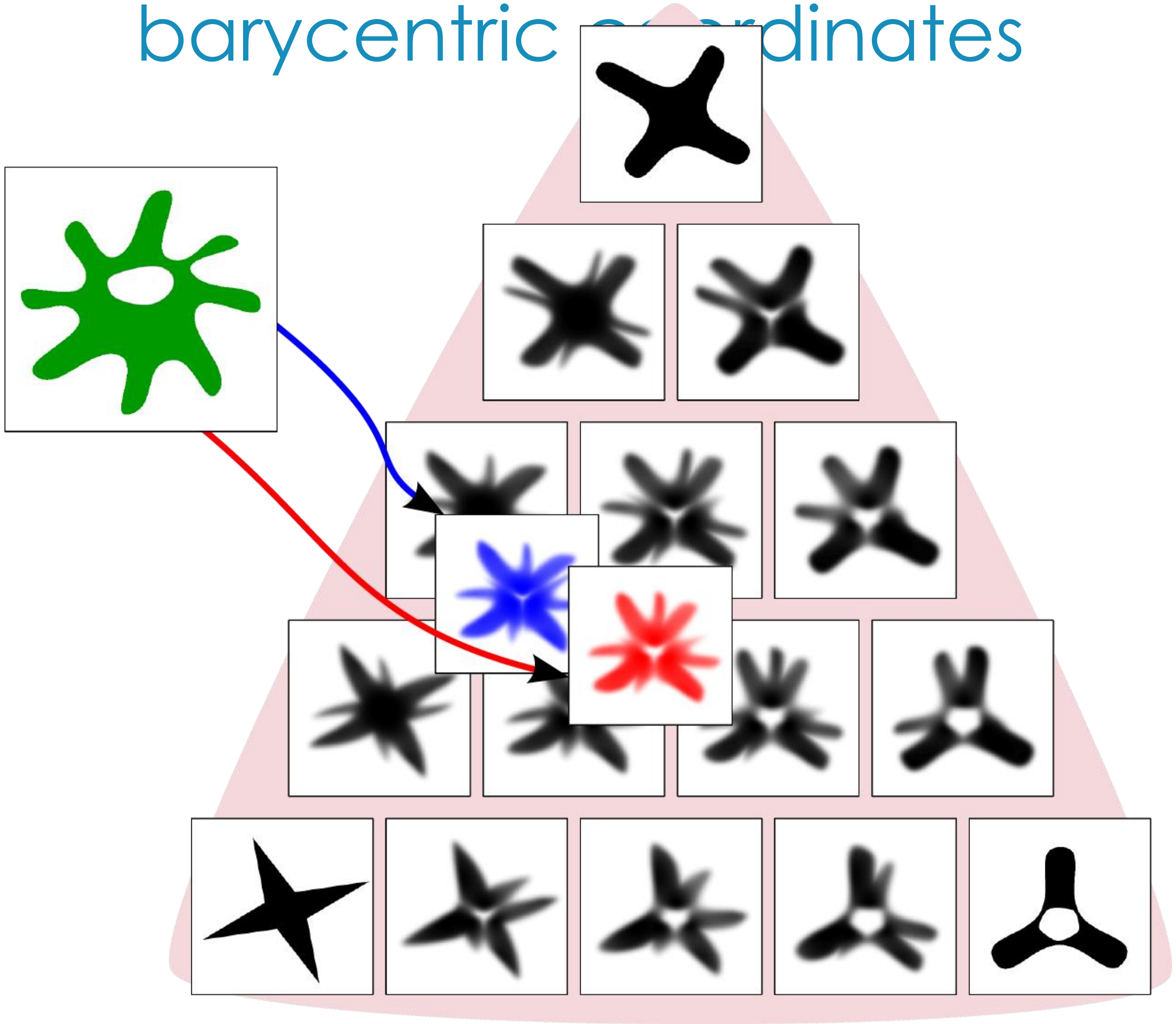
Optimal Transport



Optimal Transport



Optimal transport barycentric coordinates



Formally:

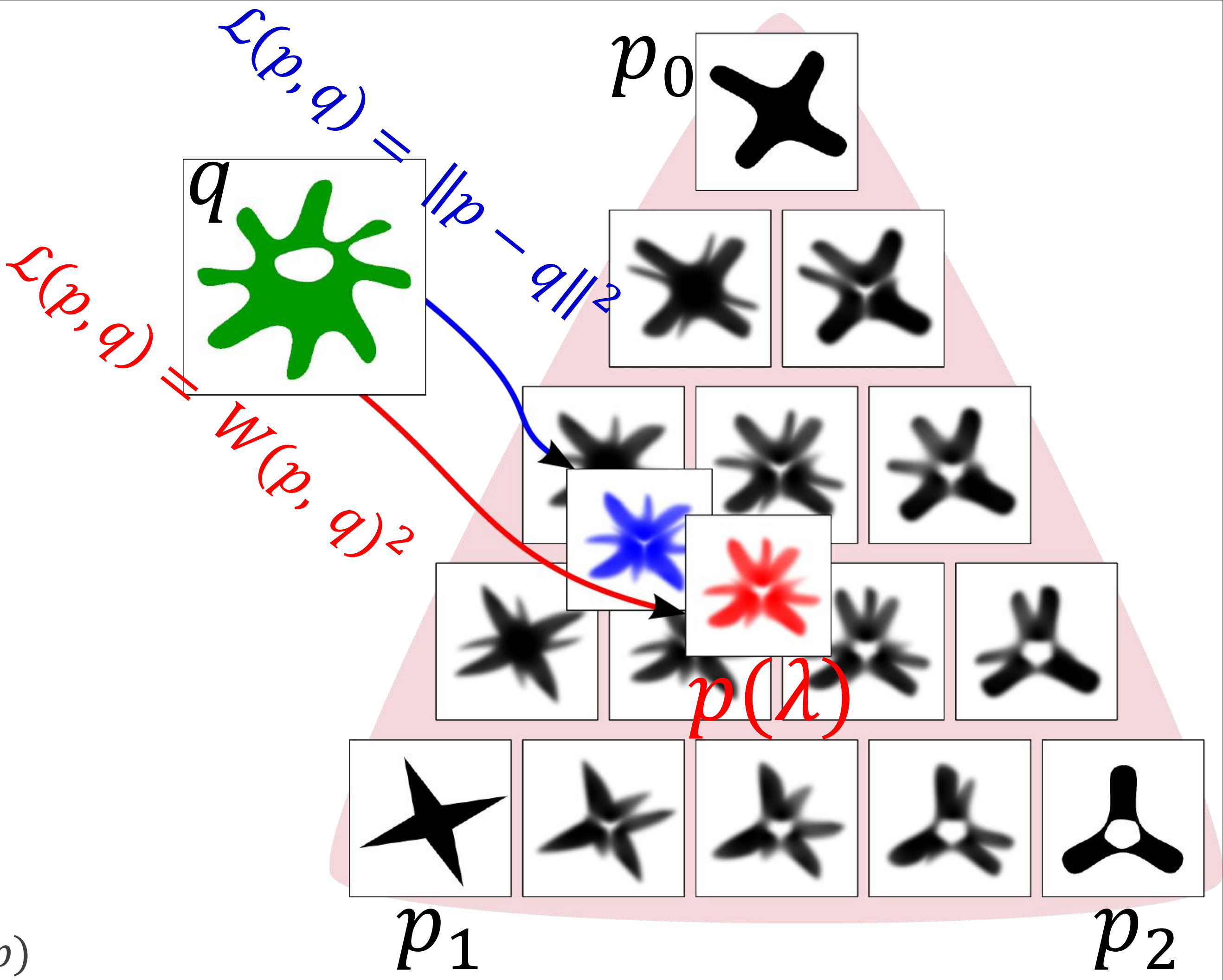
$$\min_{\lambda} \mathcal{L}(p(\lambda),\, q)$$
$$st.\ \sum \lambda_i = 1,\ \lambda_i \geq 0$$

with $p(\lambda)$ a Wasserstein barycenter:

$$p(\lambda) = \mathrm{argmin}_p \sum_s \lambda_s W^2(p_s,\, p)$$

and $\mathcal{L}(p,\, q)$ a cost function :

$$\mathcal{L}(p,\, q) = W(p, q)\,,\ \|p-q\|_2^{\,2}\,, \|p-q\|_1\ \ ,\ KL(p,\, q)$$



Method

$$\min_{\lambda} \mathcal{E}(\lambda) = \mathcal{L}(p(\lambda), q)$$

- We minimize using L-BFGS
- We use $\nabla\mathcal{E}(\lambda) = [\partial p(\lambda)]^T (\nabla\mathcal{L}(p(\lambda), q))$

Hard

Easy

Idea

- $[\partial p(\lambda)]^T$ by deriving the Sinkhorn algorithm [Solomon et al. 2015]

- To compute $p(\lambda)$ given λ, Sinkhorn iterations read:
  - $b_s^{(0)} = 1 \;\; \forall s$
  - for $\ell = 0 \ldots L$
    - $a_s^{(\ell)} = \frac{p_s}{K\, b_s^{(l-1)}} \quad \forall s$
    - $p(\lambda) = \prod_s \left(K^T a_s^{(\ell)}\right)^{\lambda_s}$
    - $b_s^{(\ell)} = \frac{p(\lambda)}{K^T\, a_s^{(\ell)}} \quad \forall s$

Idea

- Automatic differentiation: given an iterative algorithm, apply the chain rule:
  - If

$$p^{(\ell+1)}(\lambda) = f\left(p^{(\ell)}(\lambda),\, \lambda\right)$$

  - Then

$$\frac{\partial p^{(\ell+1)}}{\partial \lambda} = \frac{\partial f}{\partial p^{(\ell)}} \frac{\partial p^{(\ell)}}{\partial \lambda} + \frac{\partial f}{\partial \lambda}$$

$q^{(\ell+1)}$ $q^{(\ell)}$

- We similarly compute the adjoint
- ...formulas in the paper

Gradient computation

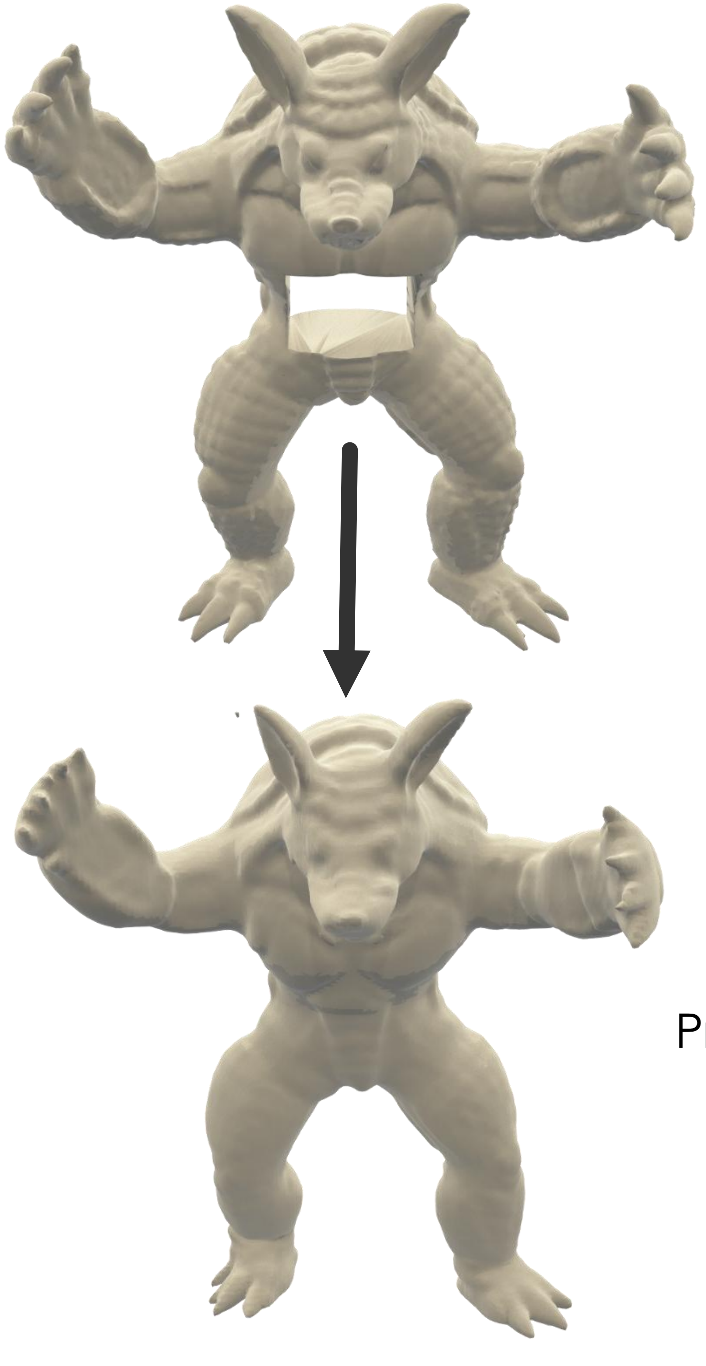
- We obtain:
  - $q_s = 0\,; r_s = 0 \ \ \forall s$
  - $g \leftarrow \nabla\mathcal{L}(p(\lambda),\ q) \odot p(\lambda)$

$\nabla\mathcal{E}(\lambda)$

  - for $\ell = L \ldots 1$
    - $\boxed{q_s} \leftarrow q_s + \left\langle \log K^T a_s^{(\ell)}\ ,\ g \right\rangle \qquad \forall s$
    - $r_s \leftarrow - K^T\left(K\left(\frac{\lambda_s g - r_s}{K^T a_s^{(\ell)}}\right) \odot \frac{p_s}{\left(K b_s^{(\ell-1)}\right)^2}\right) \odot b_s^{(\ell-1)} \qquad \forall s$
    - $g \leftarrow \sum_s r_s$

Applications

Database

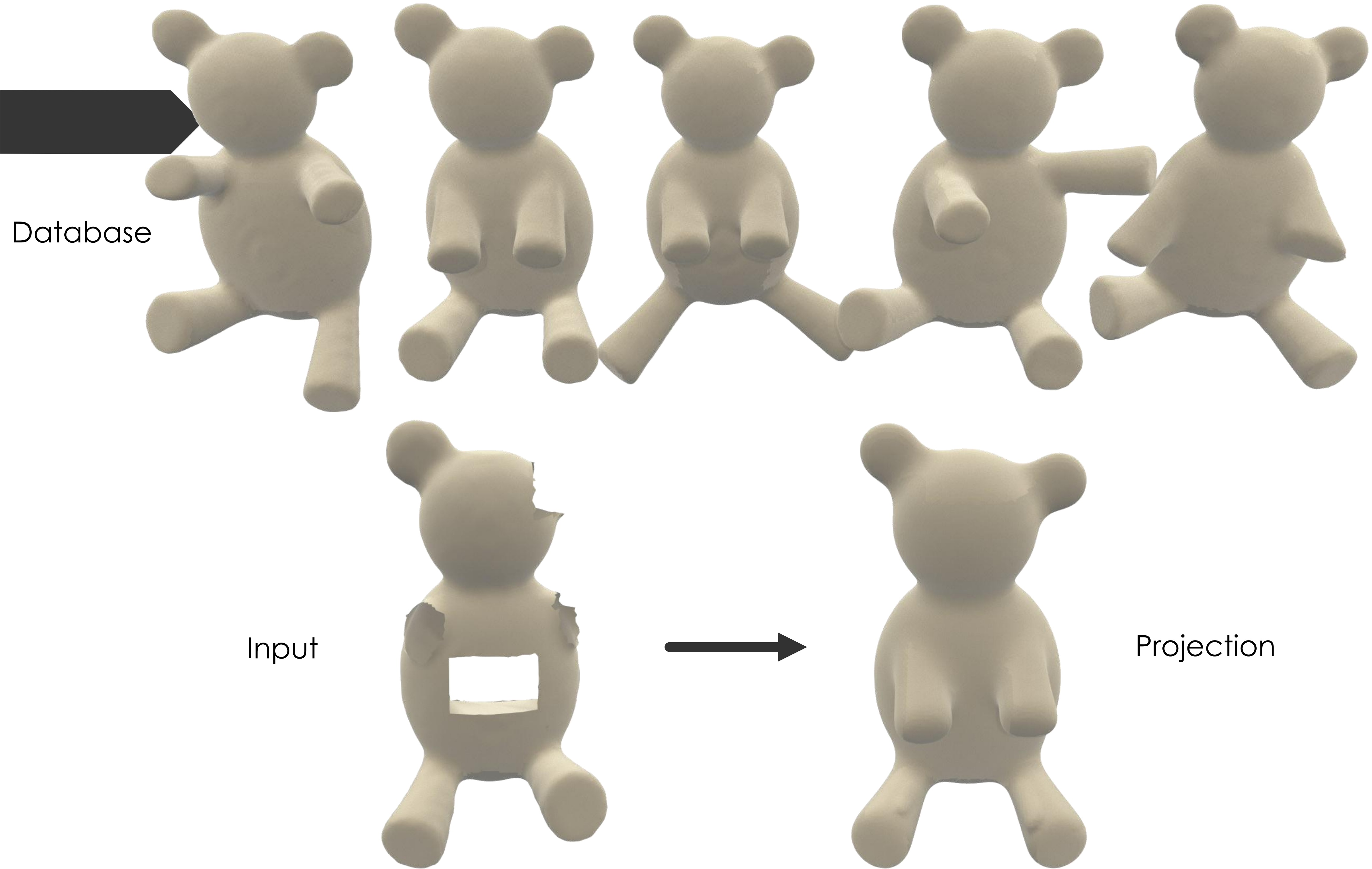


Input

Projection

Euclidean

Wasserstein


Database
Input
Projection

Applications

3E-6

6E-6

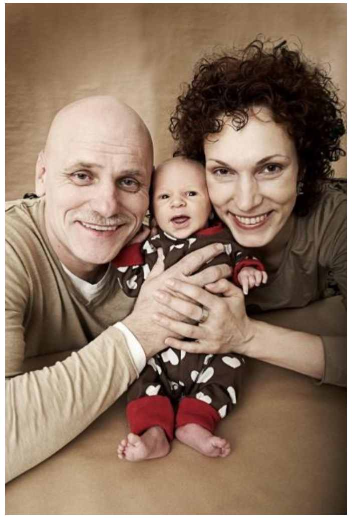
0.23

0.77

Database

Projection

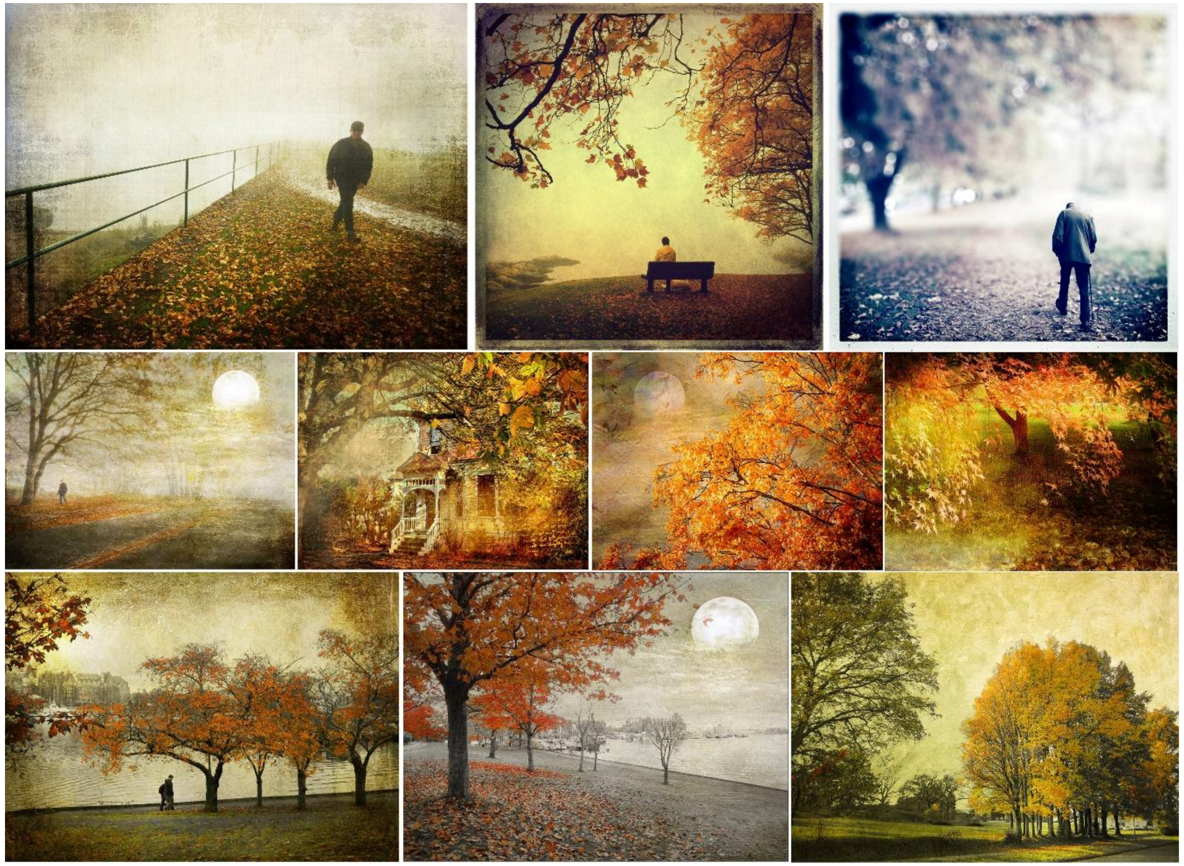
Input



Database

Projection

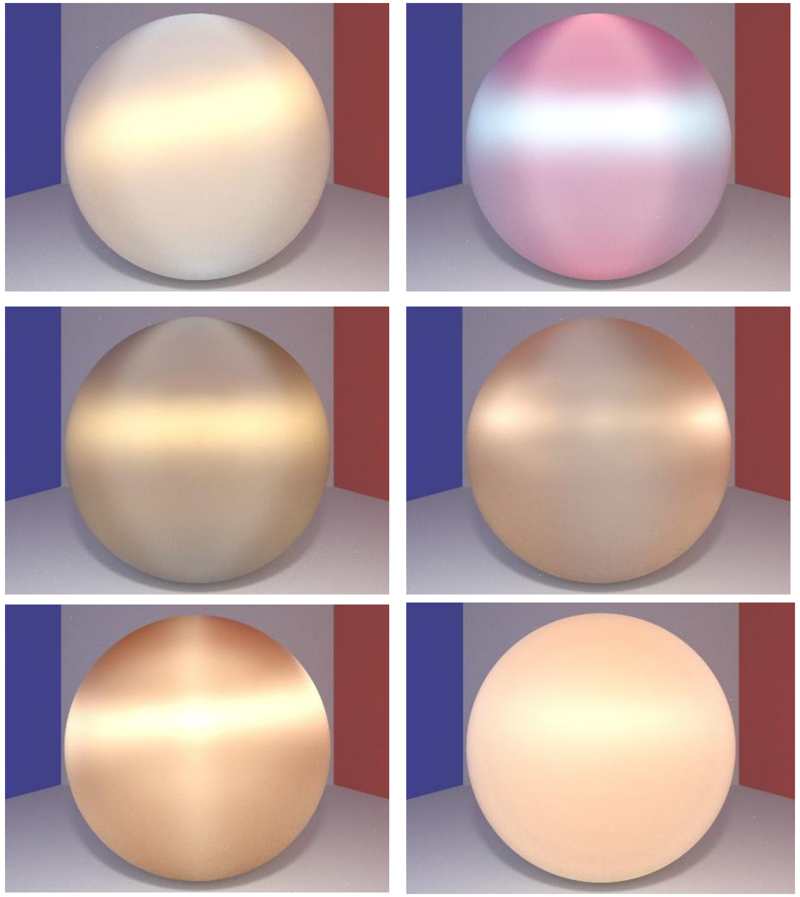
Input



Flickr results for “Autumn”

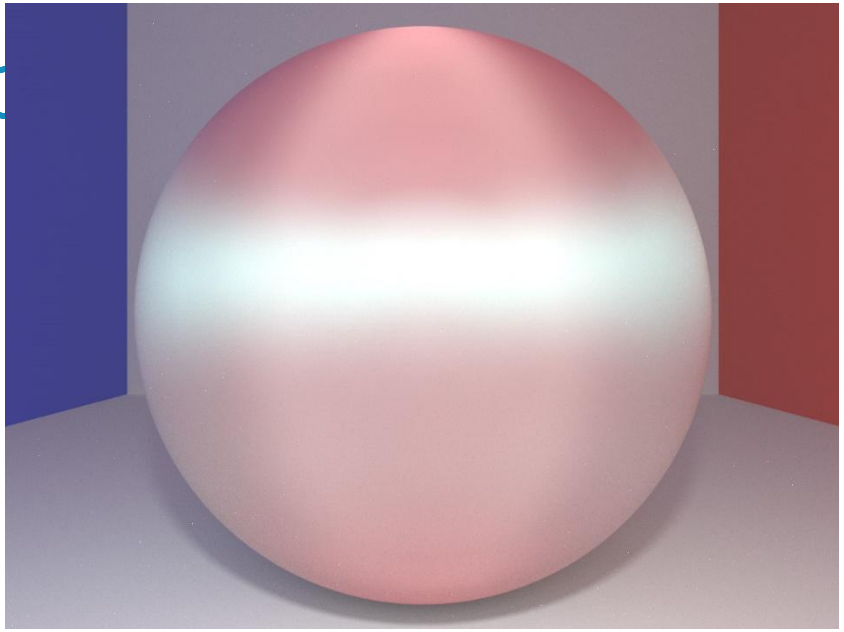
Projection
Input

Applic

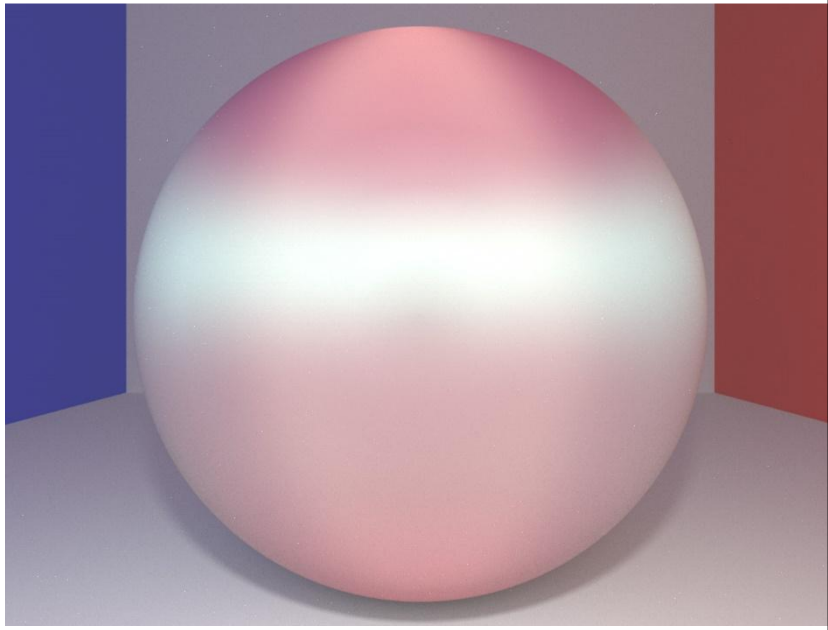


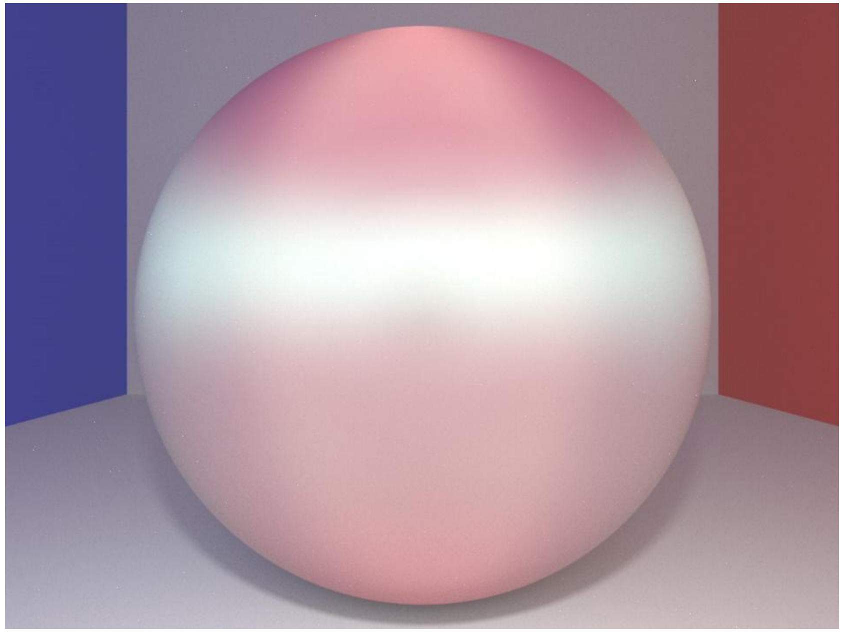
UTIA database

Reference

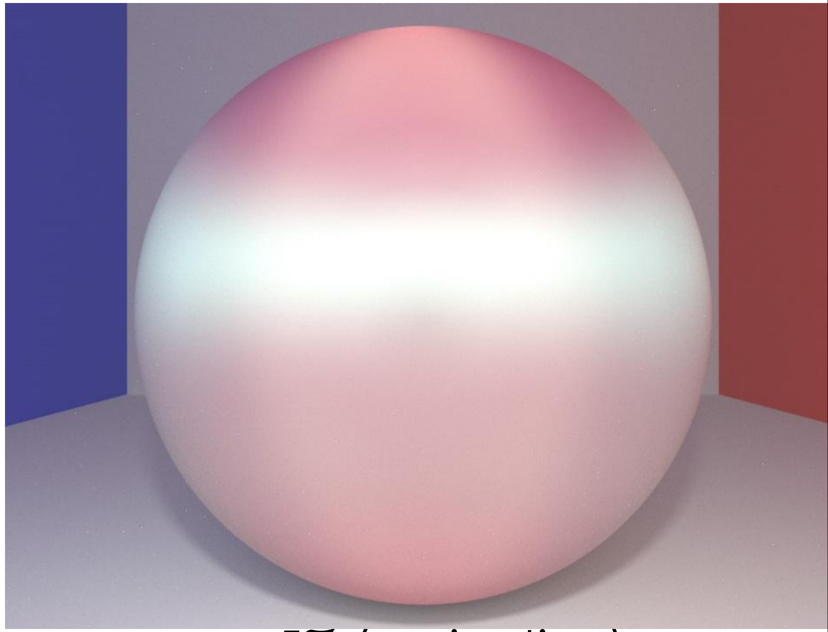


25% (projection)

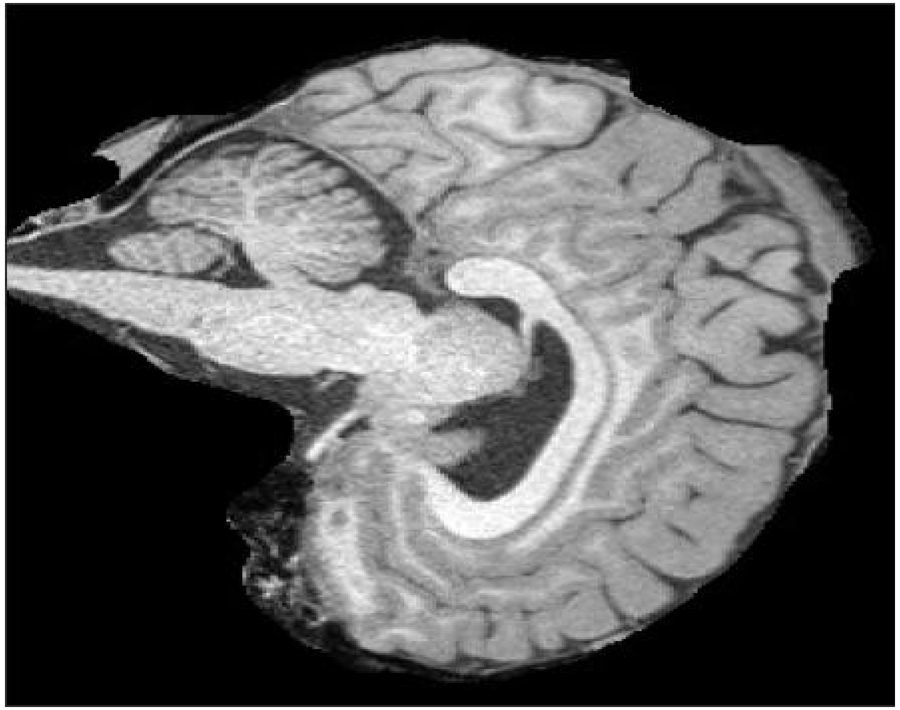




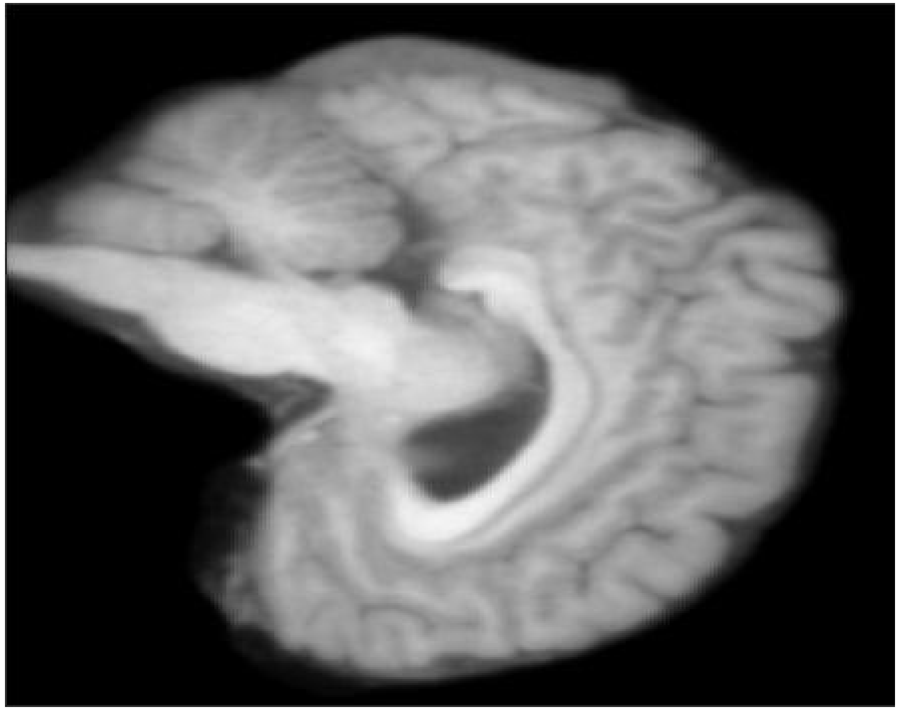
10% (projection)



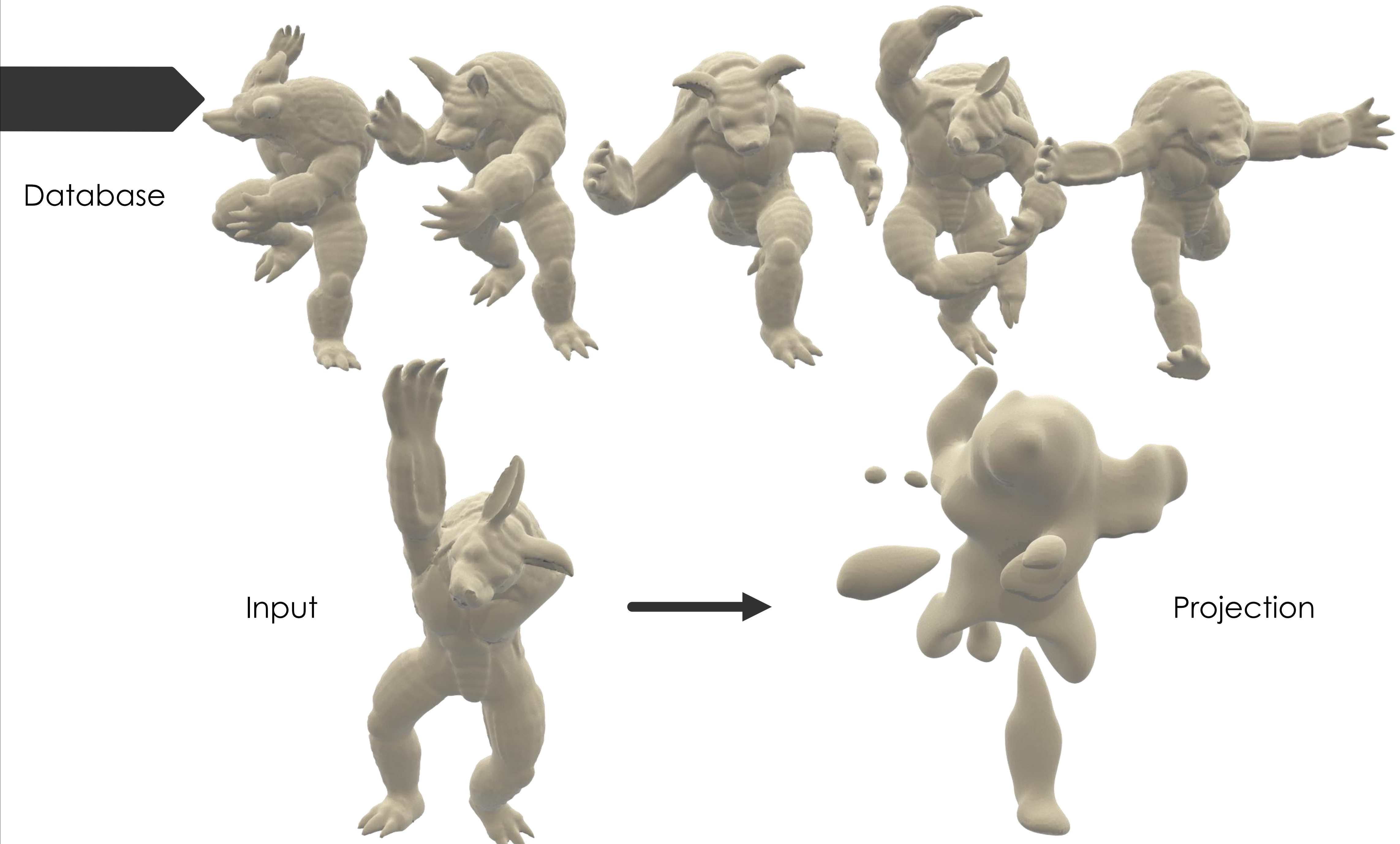
5% (projection)



Input



Projection


Database
Input
Projection

Conclusion

- Notion of barycentric coordinates useful for computer graphics
- Tractable computations
  - Barycenter gradient requires 2x convolutions w.r.t to barycenter alone
  - Relatively large memory footprint
  - Takes between seconds to minutes
- Easy to implement
  - Code available: http://liris.cnrs.fr/~nbonneel/WassersteinBarycentricCoordinates/