Mass Transportation Principles for Computer Graphics

Nicolas Bonneel
Mémoire sur la théorie des déblais et des remblais (1781)
Nobel prize in economy in 1975, for his “contribution to the theory of resources allocation”
Monge formulation

\[
\inf \int_{x} c(x, T(x))d\mu(x) \\
\text{s.t. } T_*(\mu) = \nu
\]

(or \( f(x) = |\det J_T(x)| g(T(x)) \) with \( d\mu = f(x)dx \))

(or \( \forall B, \nu[B] = \mu[T^{-1}(B)] \))

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

e.g., variational formulation with Lagrange multipliers (invalid!):

\[
\inf \int_{x} c(x, T(x))d\mu(x) + \lambda(x) |\det J_T(x)| g(T(x))
\]
Particles will move from $i$ to $j$.

Work for transforming $f$ into $g$.

Discretization of the Kantorovich problem.

Earth Mover's Distance

Cost

\[ \min_m \sum_i \sum_j c_{i,j} m_{i \rightarrow j} \]

such that:

\[ m_{i \rightarrow j} \geq 0 \]

\[ \sum_i m_{i \rightarrow j} = g_j \]

\[ \sum_j m_{i \rightarrow j} = f_i \]

Number of particles is positive!
Reconstruct target function
Reconstruct source function
$f(x)$

$g(y)$
\[ f(x) \quad g(y) \]
Application: BRDF

Function A

Linear interpolation

Function B

Displacement interpolation

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

Function B

Linear interpolation
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

- Transport RBF weights
- Network simplex > Transportation simplex

\[
\begin{align*}
\min & \quad \sum_{i,j} c_{i,j} x_{i,j} \\
\text{s.t} & \quad \sum_{j} x_{i,j} = \mu_i \\
& \quad \sum_{i} x_{i,j} = \nu_j
\end{align*}
\]
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 $j$

- Solves the dual $\min \sum r_i + \sum p_j$ s.t. $r_i + p_j \geq a_{ij} \ \forall i, j \in A(i)$
  - Value of object $j \in A(i)$: $v_{ij} = a_{ij} - p_j$
  - Profit of person $i$: $\pi_i = \max_{j \in A(i)} v_{ij}$
  - At optimality $\pi_i = \max_{k \in A(i)} a_{ik} - p_k = a_{ij} - p_j \ \forall (i, j) \in S$

- Add some slack: $\pi_i - \epsilon = \max_{k \in A(i)} a_{ik} - p_k - \epsilon \leq a_{ij} - p_j$ 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 value and the value $w_i$ 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 $O(N^2 \log(N \cdot C))$
- 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
Sliced and Radon Wasserstein Barycenters of Measures
Nicolas Bonneel, Julien Rabin, Gabriel Peyré, Hanspeter Pfister
Multi-way interpolation

- Two ways transportation:

\[ \min \sum_i \sum_j d_{i,j} x_{i \to j} \]

\[ x_{i \to j} \geq 0 \]

\[ \sum_i x_{i \to j} = g_j \]

\[ \sum_j x_{i \to j} = f_i \]

Number of non-zeros among M*N variables:

\[ \text{M} + \text{N} - 1 \]
Multi-way interpolation

- Three ways transportation:

\[
\begin{align*}
\min \sum_i \sum_j \sum_k d_{i,j,k} x_{i,j,k} \\
x_{i,j,k} \geq 0 \\
\sum_i \sum_j x_{i,j,k} = h_k \\
\sum_i \sum_k x_{i,j,k} = g_j \\
\sum_j \sum_k x_{i,j,k} = f_i 
\end{align*}
\]

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 […] )
1D Case

\[ F^{-1}_{\text{interp}}(x) = \sum_i \alpha_i F_i^{-1}(x) \]

with \( F(x) \) the CDF of \( f(x) \):

\[ F(x) = \int_{-\infty}^{t} f(t) dt \]

and \( \sum \alpha_i = 1 \)
Radon transform
Radon transform
Method
Sliced Partial Optimal Transport

Nicolas Bonneel*, David Coeurjolly*

ACM Trans. on Graphics (SIGGRAPH 2019)
Matching points

- Linear Assignment Problem

\[
\min_{T \text{ bijective}} \sum_i c(x_i, y_{T(i)})
\]

- Optimal transport

\[
W(f, g) = \min \sum_{i,j} c_{i,j} \pi_{i,j}
\]

s.t.
\[
\sum_j \pi_{i,j} = 1 \\
\sum_i \pi_{i,j} = 1 \\
\pi_{i,j} \geq 0
\]
1-d Linear Assignment Problem is trivial* assuming the cost $c$ is a convex function of $|x-y|$.
Partial optimal assignment?

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

\[ \min_{T \text{ injective}} \sum_i c(x_i, y_{T(i)}) \]
Similar problems

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

- **Dynamic Time Warping**
  - Solves a dynamic programming problem
  - Smith–Waterman algorithm, Needleman–Wunsch algorithm $O(N^2)$ space and time
  - Hirschberg's algorithm $O(N^2)$ time, $O(N)$ space
  - All end up doing variants of
    - $A_{i,j} = \min(A_{i-1,j-1} + \text{cost}, A_{i-1,j} + \text{cost}', A_{i,j-1} + \text{cost}'')$
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)

Euclidean Nearest Neighbor assignment

Optimal Transport assignment

Intervals of bijective assignments
Quadratic time complexity algorithm (linear space)

Euclidean Nearest Neighbor assignment

Optimal Transport assignment

Intervals of bijective assignments
Quadratic time complexity algorithm (linear space)

- Euclidean Nearest Neighbor assignment
- Optimal Transport assignment
- Intervals of bijective assignments
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)

Euclidean Nearest Neighbor assignment

Optimal Transport assignment

Intervals of bijective assignments
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)
Euclidean Nearest Neighbor assignment
Optimal Transport assignment

Quadratic time complexity algorithm (linear space)

$X$

$Y$

Intervals of bijective assignments
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)

Euclidean Nearest Neighbor assignment

Optimal Transport assignment

Intervals of bijective assignments
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)

Euclidean Nearest Neighbor assignment

Optimal Transport assignment

Intervals of bijective assignments
Quadratic time complexity algorithm (linear space)

Euclidean Nearest Neighbor assignment

Optimal Transport assignment

Intervals of bijective assignments
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)

Euclidean Nearest Neighbor assignment

Optimal Transport assignment

Intervals of bijective assignments
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)

Euclidean Nearest Neighbor assignment

Optimal Transport assignment

Intervals of bijective assignments
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)
Quadratic time complexity algorithm (linear space)

Euclidean Nearest Neighbor assignment
Optimal Transport assignment

Intervals of bijective assignments
Linear time problem decomposition
Problem decomposition
Problem decomposition
Problem decomposition

- Computed in quasi-linear time
- Yields independent subproblems
  - Solvable in parallel
  - That can be further simplified (see paper)
Sliced Partial Optimal Transport (SPOT)
Extension to $d$ dimensions

- Sliced optimal transport

$$E = \int_{S^{d-1}} W(P_\omega X, P_\omega Y) d\omega = \int_{S^{d-1}} \min_T \sum_i (P_\omega x_i - P_\omega y_{T(i)})^2 \ d\omega$$
Gradient flow

- Sliced optimal transport

\[ X^{n+1} = X^n - \nabla E \]

Stochastic descent: \( X^{n+1} = X^n - \nabla W(P_{\omega^n X}, P_{\omega^n Y}). \omega^n \)
Gradient flow

- Sliced optimal transport

\[
\min_{x} \int_{S^{d-1}} \min_{T} \sum_{i} (P_{\omega} x_i - P_{\omega} y_{T(i)})^2 d\omega
\]
Color Transfer Application

Full Transfer | Target 20% larger | Target 40% larger
Color Transfer Application

Full Transfer  Target 20% larger  Target 40% larger
Fast Iterative Sliced Transport
Source: 8k samples
Target: 10k samples

ICP
(0.005 s / iteration)

Iterative Transport with network simplex
(40 s / iteration)

Our FIST algorithm
(0.04 s / iteration)
Source: 90k samples
Target: 100k samples

ICP
(0.05 s / iteration)

Our FIST algorithm
(0.66 s / iteration)

(input too large for iterative transport with network simplex)
Source: 90k samples
Target: 100k samples

ICP
(0.05 s / iteration)

Our FIST algorithm
(0.69 s / iteration)
Source: 150k samples
Target: 200k samples

ICP
(0.09 s / iteration)

Our FIST algorithm
(2.18 s / iteration)

(input too large for iterative transport with network simplex)
Failure case: the transport is optimal only on projections.

Iterative Transport with Network Simplex

Our FIST algorithm
Conclusions

- Fast partial optimal transport in 1d
  - Quadratic-time algorithm (worst case)
  - Quasi-linear time decomposition
- Sliced Partial Optimal Transport
- Fast Iterative Sliced Transport
- Applications: point cloud registration, color matching
Geometric interpretations of optimal transport
The Wasserstein space

- Space of probability measures
The Wasserstein space

- Space of probability measures
- With the Earth Mover’s Distance metric
The Wasserstein space

- Space of probability measures
- With the Earth Mover’s Distance metric
- And actually, seen as a Riemannian manifold
The Wasserstein space

- Space of probability measures
- With the Earth Mover’s Distance metric
- And actually, seen as a Riemannian manifold
  - So, with a tangent space
The Wasserstein space

- Space of probability measures
- With the Earth Mover’s Distance metric
- And actually, seen as a Riemannian manifold
  - So, with a tangent space
  - And geodesics (i.e., action minimizing curves!)
The Wasserstein space

- A tangent space at $\rho$
  - $-\nabla.(\rho v)$ with $v = \nabla u$

- A curvature
  - Zero for $\mathbb{R}^1$
  - Bounded from below if manifold of positive curvature
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

[Diagram of Voronoi diagram]

[Diagram showing steps: Project onto paraboloid, Compute convex hull, Project hull faces back to plane]
Centroidal Voronoi Diagram

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

\[
\min \int_{\text{Vor}_i} \sum_i \|x - x_i\|^2dx
\]

Also says that the centroid of \( \text{Vor}_i \) 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 \( w_i \)

\[
\|x - x_i\|^2 - w_i \leq \|x - x_j\|^2 - w_j \quad \forall j
\]

- 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 \( \|x' - x_i'\|^2 \leq \|x' - x_j'\|^2 \quad \forall j \)

- Any partition into convex polyhedral cells is a power diagram of some sites
Back to optimal transport

Optimal transport (Monge version):

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

Considering $\mu$ is continuous with density $\rho$

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

Considering $\nu$ (the target measure) discrete: $\nu = \sum \lambda_i \delta_{\nu_i}$

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 as 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

\[
\min \sum_p \int_{V_{or\,w(p)}} \|x - p\|^2 \rho(x) \, dx - \sum_p w_p \left( \int_{V_{or\,w(p)}} \rho(x) \, dx - m \right)
\]

- Also optimizes for the locations \( p \)

Enforces cells to have the same mass

Blue Noise through Optimal Transport [de Goes et al. 2012]
Fluid dynamic interpretation
PDE formulation

- Introduce a time variable \( t \)

\[
\min \int_X \int_0^T \rho(t, x) \|v(t, x)\|^2 \, dt \, dx
\]

- Subject to B.C. : \( \rho(0, x) = f \) and \( \rho(T, x) = g \)

- The density \( \rho \) is transported by velocity field \( v \).
  - Continuity equation: \( \partial_t \rho + \nabla \cdot (\rho v) = 0 \)

- Optimality condition: \( v(t, x) = \nabla \phi(t, x) \) and \( \partial_t \phi + \frac{1}{2} \|\nabla \phi\|^2 = 0 \)

- After some rewriting: solved via space-time Poisson equation and projections

A computational fluid mechanics solution to the Monge-Kantorovich mass transfer problem
[Benamou & Brenier 2000]
Simple fluid simulation via semi-discrete OT

- For each time step
  - Compute OT from \( \{p_i\} \) to uniform density
  - For each site \( p_i \)
    - \( \bar{F}_i = \frac{1}{\epsilon^2} (\text{Centroid}_i - p_i) - m \ddot{g} \)
    - \( \bar{V}_i = \bar{V}_i + \frac{dt}{m_i} \bar{F}_i \)
    - \( p_i = p_i + dt \bar{V}_i \)

A Lagrangian scheme à la Brenier for the incompressible Euler equations [Gallouet, Mérigot 2017]
Simple fluid simulation via semi-discrete OT

- Extension to free boundary fluids
  - Store air + fluid particles
  - Impose each fluid particle to have constant mass (e.g., cell area = $0.5 \cdot \frac{1}{N}$ for a fluid of $N$ particles taking half of the space of a unit size domain)
  - Impose the sum of air particles to have constant mass (e.g., $\sum$ cell areas = 0.5 for the example above)
  - Same optimization as before
  - Only move fluid particles
Geodesic computation

- Special case for $L^1$ optimal transport:
  \[
  \min \int_X \|v(x)\| dx \\
  \text{s.t. } \nabla v = g(x) - f(x) \\
  v(x) \cdot n(x) = 0 \text{ on } \partial X
  \]
  - The optimal transport only depends on the difference: can remove shared mass
  - Flow lines of $v$ are geodesics on $X$

- Use Helmholtz-Hodge decomposition:
  \[
  \min \int_X \|\nabla A(x) + \nabla \times B(x) + C(x)\| dx \\
  \text{s.t. } \Delta A(x) = g(x) - f(x) \\
  B(x) = 0 \text{ and } \frac{\partial A(x)}{\partial n} = 0 \text{ on } \partial X \\
  \nabla C(x) = 0 \text{ and } \nabla \times C(x) = 0
  \]

Earth Mover’s Distances on Discrete Surfaces [Solomon et al. 2014]
Geodesic computation

- Solved using an eigen decomposition of the Laplacian as a basis.
- The curve defined by \( \dot{z} = -\frac{v(x)}{(1-t)f+tg} \) is a geodesic.
Regularized optimal transport
The Sinkhorn algorithm

- Kantorovich optimal transport: \( \min_m \sum_i \sum_j c_{i,j} m_{i \to j} \)

- 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 \( E(M) = -\sum M_{ij} (\log(M_{ij}) - 1) \) is the entropy, \( \epsilon \) a small constant

- 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

- This is a projection on two affine constraints due to \( \mathcal{U}(r,c) \)

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

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

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

\[ b_s^{(0)} = 1 \ \forall s \]

\[ \text{for } \ell = 0 \ldots L \]

\[ a_s^{(\ell)} = \frac{p_s}{K b_s^{(\ell-1)}} \ \forall s \]

\[ p(\lambda) = \prod_s \left( K^T a_s^{(\ell)} \right)^{\lambda_s} \]

\[ b_s^{(\ell)} = \frac{p(\lambda)}{K^T a_s^{(\ell)}} \ \forall s \]
Wasserstein Barycentric Coordinates: Histogram Regression Using Optimal Transport

N. Bonneel, G. Peyré, M. Cuturi

SIGGRAPH 2016
Barycentric coordinates
Barycentric coordinates
Barycentric coordinates
Barycentric coordinates
Optimal Transport

$t = 0$

$t = 1$
Optimal Transport

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

s.t.  
$$m_{ij} \geq 0 ; \Sigma_i m_{ij} = g(x_j) ; \Sigma_j m_{ij} = f(x_i)$$
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:

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

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
  \[ \nabla \mathcal{E}(\lambda) = [\partial p(\lambda)]^T (\nabla \mathcal{L}(p(\lambda), q)) \]
Idea

- \([\partial p(\lambda)]^T\) by deriving the Sinkhorn algorithm [Solomon et al. 2015]
- To compute \(p(\lambda)\) given \(\lambda\), Sinkhorn iterations read:


- \(b_s^{(0)} = 1 \ \forall s\)
- for \(\ell = 0 \ldots L\)

- \(a_s^{(\ell)} = \frac{p_s}{K b_s^{(\ell-1)}} \ \forall s\)
- \(p(\lambda) = \prod_s \left(K^T a_s^{(\ell)}\right)^{\lambda_s}\)
- \(b_s^{(\ell)} = \frac{p(\lambda)}{K^T a_s^{(\ell)}} \ \forall s\)
Idea

- Automatic differentiation: given an iterative algorithm, apply the chain rule:
  - If
    \[ p^{(\ell+1)}(\lambda) = f(p^{(\ell)}(\lambda), \lambda) \]
  - Then
    \[ \frac{\partial p^{(\ell+1)}}{\partial \lambda} = \frac{\partial f}{\partial p^{(\ell)}} \times \frac{\partial p^{(\ell)}}{\partial \lambda} + \frac{\partial f}{\partial \lambda} \]

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

- We obtain:

\[ q_s = 0 \; ; \; r_s = 0 \; \forall s \]

\[ g \leftarrow \nabla L(p(\lambda), q) \odot p(\lambda) \]

- for \( \ell = L \; \ldots \; 1 \)

\[ q_s \leftarrow q_s + \left( \log K^T a_s^{(\ell)} , g \right) \; \forall s \]

\[ r_s \leftarrow -K^T \left( K \left( \frac{\lambda_s g - r_s}{K^T a_s^{(\ell)}} \right) \odot \frac{p_s}{(Kb_s^{(\ell-1)})^2} \right) \odot b_s^{(\ell-1)} \; \forall s \]

\[ g \leftarrow \sum_s r_s \]
Applications
Database

Input

Projection
Applications

Database

3E-6

6E-6

0.23

0.77

Projection
Flickr results for “Autumn”
Database

Input

Projection
Wasserstein Dictionary Learning: Optimal Transport-based unsupervised non-linear dictionary learning

SIAM Journal on Imaging Sciences
What is dictionary learning?

- **Linear Dictionary Learning**
  - \( X \): input elements (column vectors)

- **Factorization** \( X \approx D \Lambda \)
  - \( D \): atoms of same dimension as elements of \( X \).
  - \( \Lambda \): weights to reconstruct input elements by **linear combination** of the atoms.

Here, \( \Lambda = [(1.0 \ 0.0), (1.0 \ 0.0), (0.5 \ 0.5), (0.0 \ 1.0), (0.0 \ 1.0)] \)
What is dictionary learning?

- **Wasserstein Dictionary Learning**
  - $X$: input elements (column vectors)

- Factorization $X \approx P(D, \Lambda)$
  - $D$: atoms of same dimension as elements of $X$.
  - $\Lambda$: weights to reconstruct input elements by **Wasserstein combination** of the atoms.

Here, $\Lambda = \{(1.0 1.0), (0.75 0.25), (0.5 0.5), (0.25 0.75), (0.0 1.0)\}$
Wasserstein dictionary learning

\[ D_0 \]

\[ D_1 \]

\[ D_2 \]
Wasserstein dictionary learning

- Now

$$\mathcal{E}'(\lambda, \{D_s\}_s) = \sum_i \mathcal{L}(p(\lambda_i, \{D_s\}_s), X_i)$$

- Idea: differentiate $\mathcal{E}'$ w.r.t the weights and atoms

- $\nabla_\lambda \mathcal{E}'(\lambda, \{D_s\}_s)$ as before

- $\nabla_{\{D_s\}_s} \mathcal{E}'(\lambda, \{D_s\}_s)$ as follows
Wasserstein dictionary learning

- \( c_s = 0; \nu_s = 0; g_s = 0 \ \forall s \)
- \( n \leftarrow \nabla \mathcal{L}(p(\lambda, \{D_s\}_s), X_i) \)
- for \( \ell = L \ldots 1 \)
  - \( c_s \leftarrow K \left( (\lambda_s n - \nu_s) \odot b_s^{(\ell)} \right) \) \ \forall s 
  - \( g_s \leftarrow g_s + \frac{c_s}{K b_s^{(\ell-1)}} \) \ \forall s 
  - \( \nu_s \leftarrow -\frac{1}{K^T a_s^{(\ell-1)}} \odot K^T D_s \odot c_s \) \ \forall s 
  - \( n \leftarrow \sum_s \nu_s \)
Extensions

- Log-domain computations
  - Including separable convolutions in log-domain
- Heavy-ball extrapolation
  - Faster convergence
  - Requires ‘real’ automatic-differentiation
- Unbalanced optimal transport
  - Requires ‘real’ automatic-differentiation
Applications

- PSF learning: PSF varies with wavelength

- Learns 2 atoms
Applications

- “EigenFaces”
Extension to Dictionary Learning
Extension to Dictionary Learning
Applications

$D_\text{Wasserstein}$

$D_\text{PCA}$
Applications

$D_{\text{Wasserstein}}$

$D_{\text{NMF}}$
Conclusion

- Notion of barycentric coordinates useful for computer graphics and tractable
  - Barycenter gradient requires 2x convolutions w.r.t to barycenter alone
  - Relatively large memory footprint
  - Takes between seconds to minutes
- Wasserstein dictionary learning useful for summarizing histogram data
  - Still tractable, though with gradient requiring 2D+2N x convolutions
- Easy to implement
  - Code available: [http://liris.cnrs.fr/~nbonneel/WassersteinBarycentricCoordinates/](http://liris.cnrs.fr/~nbonneel/WassersteinBarycentricCoordinates/)
  - [https://github.com/matthieuheitz/WassersteinDictionaryLearning](https://github.com/matthieuheitz/WassersteinDictionaryLearning)