An Idea of Optimal Transportation in Curve Fitting

Bohao Tang: bhtang@jhu.edu

Feb. 2019

1 Introduction

Here we explore using the idea of optimal transportation in curve fitting problem. The setting is that you have a picture, here we normalize it to add to 1 and therefore consider it a 2 dimensional distribution $Q(x, y)$. Our task is to represent it by a curve

$$x = f(t) \tag{1}$$
$$y = g(t) \tag{2}$$

For $t \in [0, 1]$.
The idea here is that: Suppose there is a uni-variate distribution on $[0, 1]$: $P(t)$, then through curve representation 1, we actually get a distribution in $\mathbb{R}^2$, denote by $P(\mathcal{F}(t))$. Then we can just minimize the distance of these two distributions Q and $P\mathcal{F}$ to get our curve representation.
But the major challenge is that $P\mathcal{F}$ concentrates on a lower dimensional manifold and therefore has no density, we can't easily measure the distance of it to a continuous distribution Q.
To solve this, **principle curve method** actually gave up in measuring distance of distributions, but to measure expected Euclidean distance from $\mathbf{x} \sim Q$ to curve 1, which is then equivalent to some local self-consistent properties.
But we argue that actually we can efficiently and meaningfully measure the distance of Q and $P\mathcal{F}$ through the idea of optimal transportation, which is using **Wasserstein distance**. Through this idea (detail in following section), we are actually trying find a distribution on a curve, which costs you the least to move all the mass on picture to the curve.

2 Method

Here we use a spline method to represent the curve, suppose the basis on point t is B_t. Then the Wasserstein distance of two distribution is defined as:

$$d_w^2(P, Q) = \inf \left\{ \int c(\mathbf{x}, \mathbf{y}) dF(\mathbf{x}, \mathbf{y}) : \ F|_{\mathbf{x}} = P, F|_{\mathbf{y}} = Q \right\} \tag{3}$$

Where $F|_{\mathbf{x}}$ is the marginal distribution of F to $\mathbf{x}$.
The joint distribution F here is a plan for transporting P to Q, and $c(\mathbf{x}, \mathbf{y})$ is the cost to move mass from $\mathbf{x}$ to $\mathbf{y}$. Here we use $c(\mathbf{x}, \mathbf{y}) = ||\mathbf{x} - \mathbf{y}||_2^2$.

This definition seems very hard to compute, but its dual form can be approximate well from adversarial learning, which we may try in the future. Here we just do a discrete approximation. Suppose the coefficients of curves are θ_1, θ_2. Then approximate the picture distribution by

$$Q(x, y) = \sum_i q_i \delta(x - x_i, y - y_i)$$

and approximate curve distribution by

$$P(x, y) = \sum_j p_j \delta(x - B_{t_j}\theta_1, y - B_{t_j}\theta_2)$$

Then after simple calculation we know that $d_w^2(P, Q)$ is:

$$\min_\gamma \sum_{i,j} \{(x_i - B_{t_j}\theta_1)^2 + (y_i - B_{t_j}\theta_2)^2\}\gamma_{ij} \tag{4}$$

$$s.t. \quad \sum_i \gamma_{ij} = p_j \tag{5}$$

$$\sum_j \gamma_{ij} = q_i \tag{6}$$

$$\gamma_{ij} \geq 0 \tag{7}$$

Which is therefore simply a linear program and can be compute efficiently.
Combine the coefficients, we know that we need to solve the problem:

$$\min_{\gamma, p_j, \theta_1, \theta_2} \sum_{i,j} \{(x_i - B_{t_j}\theta_1)^2 + (y_i - B_{t_j}\theta_2)^2\}\gamma_{ij} + \lambda(\theta_1^T R \theta_1 + \theta_2^T R \theta_2) \tag{8}$$

$$s.t. \quad \sum_i \gamma_{ij} = p_j \tag{9}$$

$$\sum_j \gamma_{ij} = q_i \tag{10}$$

$$\gamma_{ij} \geq 0 \tag{11}$$

Given $B_t, t_j, q_i, x_i, y_i, \lambda$. Here R is a regularization term for θ_1 and θ_1
Unfortunately optimization 8 is not convex. But here we try a simple idea, given θ_1, θ_2, above question is a linear program and given γ, p_j, above question is weighted least square, which can both be solved efficiently. Therefore we iteratively optimize for these two sets of parameters.
The algorithm is:

1. Choose t_j and basis, then Initialize θ_1, θ_2 to let the curve be a straight line pass $(E_Q X, E_Q Y)$ and the direction be first principle component of Q.

2. Get p_j, γ_{ij} by solving linear programming:

$$\min_{\gamma, p_j} \sum_{i,j} \{(x_i - B_{t_j}\theta_1)^2 + (y_i - B_{t_j}\theta_2)^2\}\gamma_{ij} \tag{12}$$

$$s.t. \quad \sum_i \gamma_{ij} = p_j \tag{13}$$

$$\sum_j \gamma_{ij} = q_i \tag{14}$$

$$\gamma_{ij} \geq 0 \tag{15}$$

3. Denote $\Lambda = diag(\sum_i \gamma_{ij})$ and $\Gamma = (\gamma_{ij})$, then solving weighted least square:

$$\min_{\theta_1,\theta_2} \sum_{i,j} \{(x_i - B_{t_j}\theta_1)^2 + (y_i - B_{t_j}\theta_2)^2\}\gamma_{ij} + \lambda(\theta_1^T R\theta_1 + \theta_2^T R\theta_2)$$

Where the solution are:

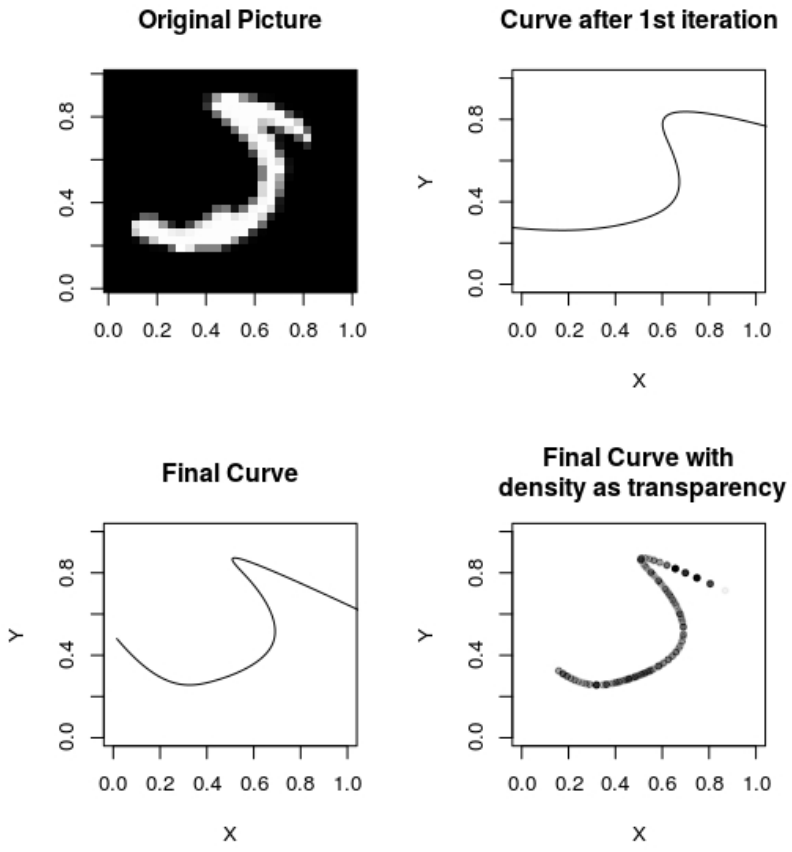
$$\theta_1 = (B^T\Lambda B + \lambda R)^{-1}B^T\Gamma^T\mathbf{x} \tag{16}$$
$$\theta_2 = (B^T\Lambda B + \lambda R)^{-1}B^T\Gamma^T\mathbf{y} \tag{17}$$

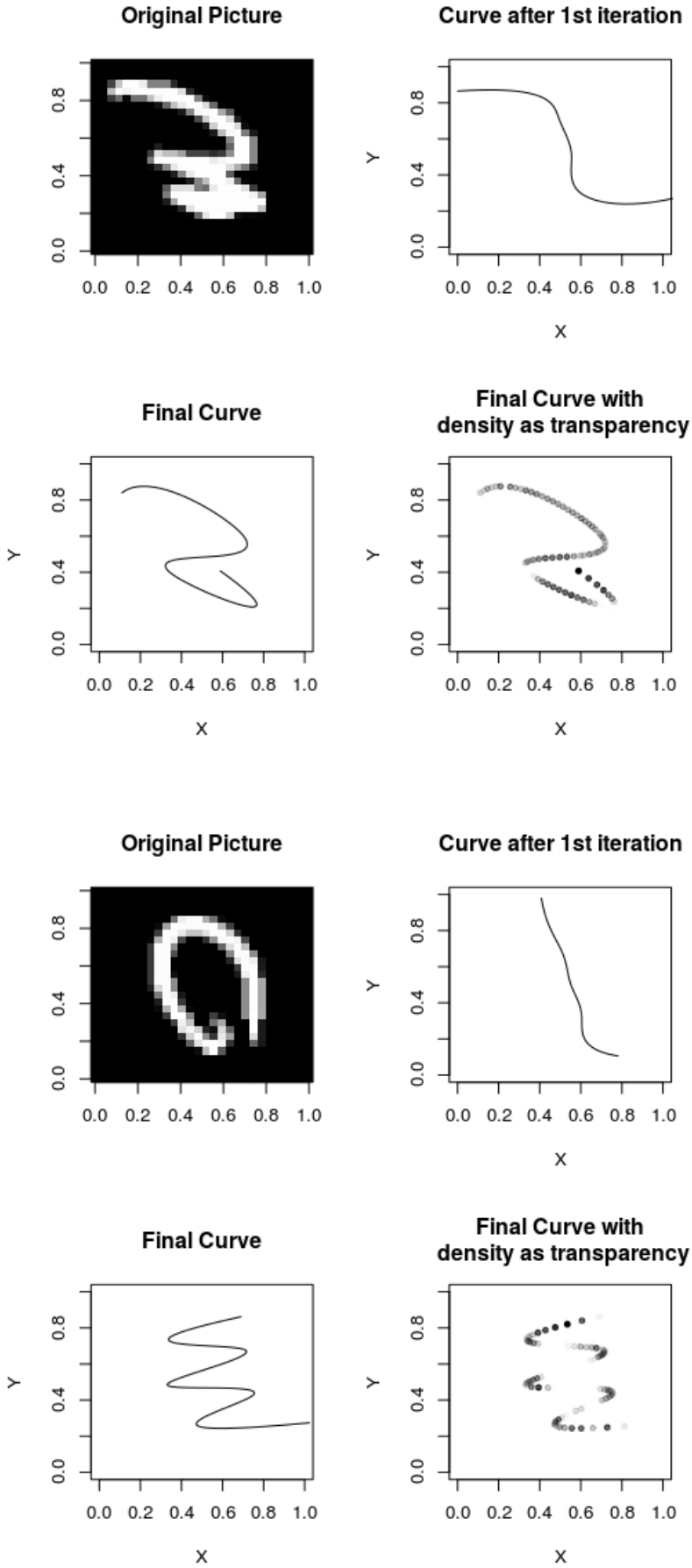
4. Iterate 2. and 3. until convergent. Then output coefficients.

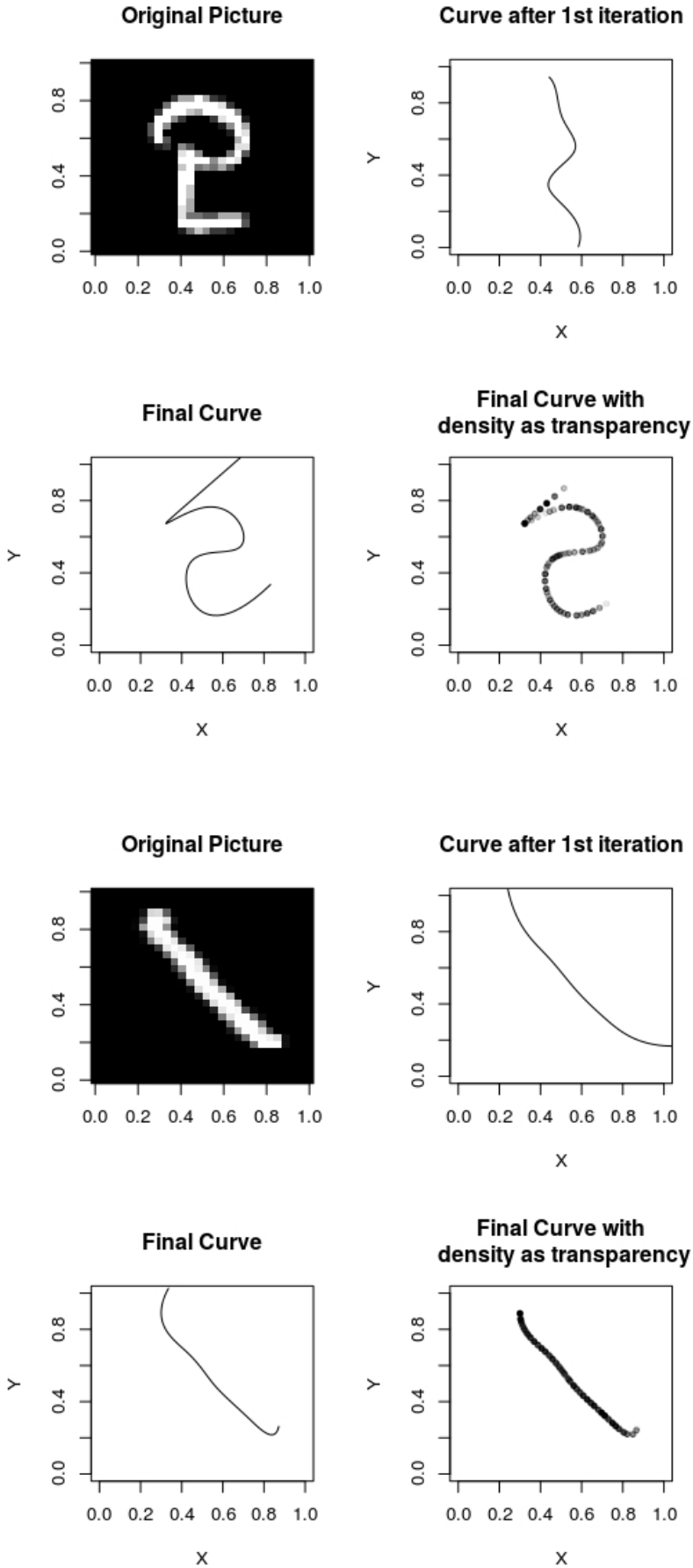
We should try later for other ways to solve this.

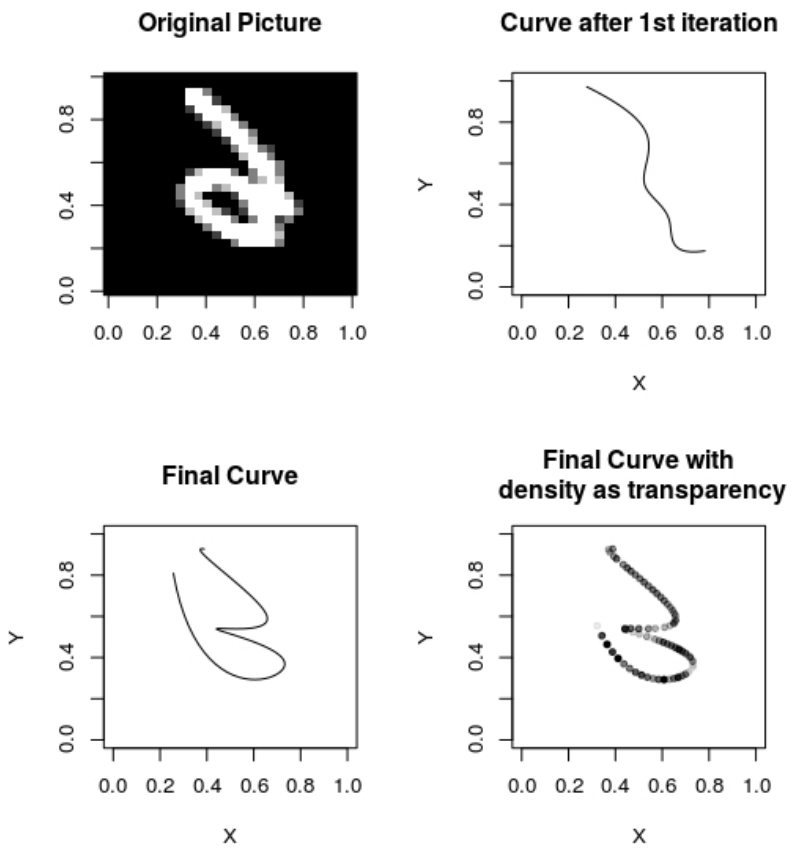
3 Experiment

Here again we use MNIST dataset and we choose t_j uniform 100 points from $[0, 1]$, the basis is from bspline with intercept and df $= 7$. And for R, we choose to let $\theta^T R\theta = \sum_i(\theta_{i+1} - \theta_i)^2$ to simulate length penalty.
Some plot are shown below (we argue that although sometimes the final curve is not good, but the distribution we get on the curve is somehow close to the picture and therefore we can find ways to modify curve to be more real):




Original Picture
Curve after 1st iteration
Final Curve
Final Curve with
density as transparency
Original Picture
Curve after 1st iteration
Final Curve
Final Curve with
density as transparency
X
Y
0.0
0.2
0.4
0.6
0.8
1.0


Original Picture
Curve after 1st iteration
Final Curve
Final Curve with
density as transparency
Original Picture
Curve after 1st iteration
Final Curve
Final Curve with
density as transparency
X
Y
0.0
0.2
0.4
0.6
0.8
1.0


Original Picture
Curve after 1st iteration
Final Curve
Final Curve with
density as transparency
0.0
0.4
0.8
0.0 0.2 0.4 0.6 0.8 1.0
X
Y