

Semi-Parametric Manifold Clustering

Estimating Polynomial Curves

Problem Setup

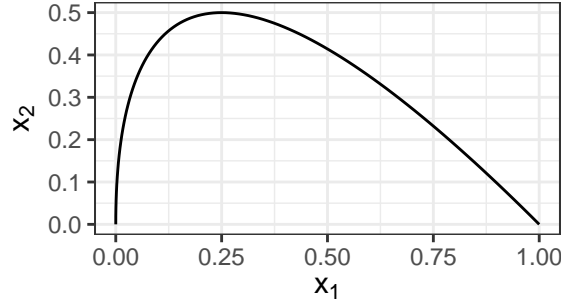
Let:

- $T_1, \dots, T_n \stackrel{\text{iid}}{\sim} F$ with support $[0, 1]$.
- $g(\cdot, \theta) : [0, 1] \mapsto \mathcal{X} \subset \mathbb{R}^d$.
- $X_1, \dots, X_n = g(T_1), \dots, g(T_n)$

Assuming some parametric form of g with parameters θ , we want to find $\hat{\theta}$, some “reasonable” estimate for θ . We observe X_i but not T_i .

For now, we limit $d = 2$ and g to quadratic functions.

Example 1. Let $g(t) = (t^2, 2t(1-t)) = (0 + 0t + t^2, 0 + 2t - 2t^2)$. (This is the first two dimensions of the Hardy-Weinberg curve). Then $\theta = (0, 0, 1, 0, 2, -2)$.



If we observe the T_i 's, then we can use a standard polynomial regression method to obtain $\hat{\theta}$. Since we do not observe them, the proposed iterative method is as follows:

1. Initialize $\hat{\theta}^{(0)}$ (e.g., randomly).
2. Estimate each $\hat{t}_i^{(s)}$ by minimizing $L(t_i, \hat{\theta}^{(s)} | x_i) = L_i = \|x_i - g(t_i | \hat{\theta}^{(s)})\|^2$.
3. Compute each $\hat{x}_i^{(s)} = g(\hat{t}_i^{(s)} | \hat{\theta}^{(s)})$
4. Estimate $\hat{\theta}^{(s+1)}$ by minimizing $L(\{\hat{t}_i^{(s)}\}, \theta | X) = \sum_i \|x_i - g(\hat{t}_i^{(s)} | \theta)\|^2$.
5. Repeat steps 2-4 until convergence.

If we restrict g to be polynomials, then steps (2) and (4) have closed-form solutions. Alternatively, we can estimate g using more general forms, e.g., splines, which may require approximation.

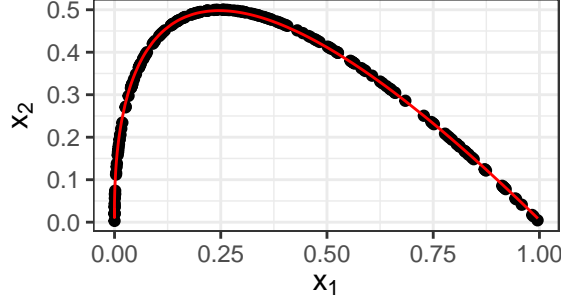
Example 2. Write $g(t|\theta) = (g_1(t|\theta_1), \dots, g_d(t|\theta_d))$ where $g_r(t|\theta_r)$ is the component of g in the r^{th} dimension and θ_r is the vector of parameters for the r^{th} dimension. If g_r are polynomials of degree p , then each θ_r contains up to $p+1$ entries.

Given the observed points $x_1, \dots, x_n \in \mathbb{R}^d$ and their corresponding index points $t_1, \dots, t_n \in \mathbb{R}$, we can find each $\hat{\theta}_r$ individually by $\hat{\theta}_r = A^{-1}b$ where $b \in \mathbb{R}^{p+1}$ and $b_k = \sum_i x_i t_i^k$ and $A \in \mathbb{R}^{(p+1) \times (p+1)}$ and

$$A_{kl} = \sum_i t^{(k-1)(l-1)}.$$

On the other hand, if we have parameters θ but not the index points t_i , we can minimize each t_i individually by finding the roots of a $p + 1$ polynomial with coefficients that depend on x_1, \dots, x_n and θ .

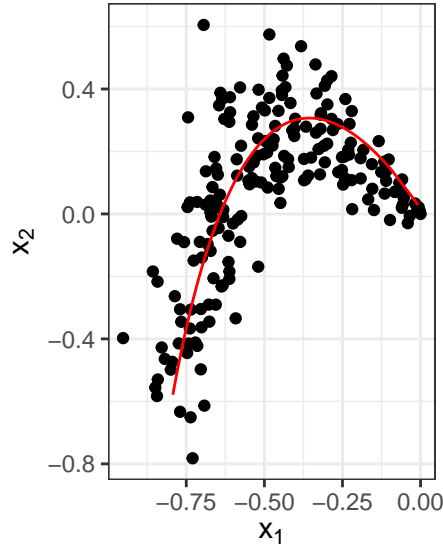
In the following plot, we drew $n = 200$ points from the 2D H-W curve with $T_1, \dots, T_n \stackrel{\text{iid}}{\sim} \text{Uniform}(0, 1)$. The red line is the curve that was fit using the above method.



Note: the parameterization of the curve is not unique.

Estimation with Noise

Example 3. In the next example, we draw $A \sim \text{RDPG}(X)$ using the same H-W curve and sample size as above and estimate the true latent positions (up to rotation).



A modification to this that is possibly more robust is to use Bezier curves for g . This is the same functional form as the polynomial curves used before, but with orthogonal bases:

$$g(t|p) = \sum_{r=0}^R p_r \binom{R}{r} (1-t)^{R-r} t^r$$

where R is the order of the Bezier curve, each p_s is a vector of length d , and, as before, $g : [0, 1] \mapsto \mathbb{R}^d$. Thus, if we fit each p_r , then the procedure is the same as before.

The least squares estimate for $p \in \mathbb{R}^{R \times d}$ is

$$\hat{p} = (T^\top T)^{-1} T^\top X$$

where $T = \begin{bmatrix} t_{\cdot,1} & \cdots & t_{\cdot,R} \end{bmatrix}$ and each $t_{ir} = \binom{R}{r} (1 - t_i)^{R-r} t_i^r$, and $X \in \mathbb{R}^{n \times d}$. The same procedure for estimating t_1, \dots, t_n can be applied here.

The parameterization for a given curve is not unique. In particular, the above procedure will not necessarily provide $t_1, \dots, t_n \in [0, 1]$. One possible remedy for this is to, after estimating the t_i 's, normalize them to the unit interval. If we assume $t_1, \dots, t_n \stackrel{\text{iid}}{\sim} \text{Uniform}(a, b)$, then the UMVUE are

$$\hat{a} = \frac{nt_{(1)} - t_{(n)}}{n - 1}, \hat{b} = \frac{nt_{(n)} - t_{(1)}}{n - 1}$$

which yields the normalization transformation $t \leftarrow (t - \hat{a})/(\hat{b} - \hat{a})$.

Alternatively, we can force the t_i 's to be approximately uniform on the unit interval by the transformation $t \leftarrow \hat{F}(t)$, where \hat{F} is the empirical CDF of t_1, \dots, t_n .

For initialization, we can use a one-dimensional Isomap embedding to estimate the t_i 's and use that to estimate \hat{p} . Experiments suggest that if the data are well-behaved (i.e., it looks like the curve we are trying to fit), this results in much faster convergence.

Theoretical Framework

If we assume that the points are distributed as $X_i \stackrel{\text{ind}}{\sim} \mathcal{N}(f(t_i|p), \Sigma(t_i|\phi))$, then we can write the (incomplete) log likelihood as:

$$\ell(p, \phi) = -\frac{1}{2} \sum_i \log |\Sigma(t_i|\phi)| - \frac{1}{2} \sum_i (x_i - f(t_i|p))^\top (\Sigma(t_i|\phi))^{-1} (x_i - f(t_i|p))$$

In the case $\Sigma(t_i|\phi) = \phi I$ and $f(t_i|p) = p^\top \tilde{t}_i$ where $\tilde{t}_i \in \mathbb{R}^{R+1}$ is the Bezier polynomial expansion of order R , then this becomes

$$\ell(p, \phi) = -\frac{nd}{2} \log \phi - \frac{1}{2\phi} \|X - Tp\|_F^2$$

Then given t_1, \dots, t_n , the MLE of p is again $(T^\top T)^{-1} T^\top X$, and the MLE of ϕ is just $\frac{1}{nd} \|X - T\hat{p}\|_F^2$. And given p , we can estimate each t_i in the same way as before to maximize ℓ (the variance is unnecessary here).

As noted before, the parameterization given by T and p is not unique. We can arbitrarily scale the t_i 's and p to obtain the same value of ℓ . Naively performing the proposed algorithm sometimes results in estimates for p that diverge and estimates for t_i 's that converge to a single value. To remedy this, we can either scale the t_i 's as before, or we can scale p by forcing f to be an arclength parameterization.

Theorem 1. Let $x_1, \dots, x_n \stackrel{\text{ind}}{\sim} \mathcal{N}(p^\top \tilde{t}_i, \phi I_d)$, where each $x_i \in \mathbb{R}^d$, $\tilde{t}_i \in \mathbb{R}^{R+1}$ such that $\tilde{t}_{i,r} = \binom{R}{r} (1 - t_i)^{R-r} t_i^r$ for $i = 1, \dots, n$ and $r = 0, \dots, R$, and $p \in \mathbb{R}^{(R+1) \times d}$.

Then each iteration of the following decreases the negative log likelihood:

1. $p \leftarrow (T^\top T)^{-1} T^\top X$, where $T = \begin{bmatrix} \tilde{t}_1 & \cdots & \tilde{t}_n \end{bmatrix}^\top$ and $X = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix}^\top$.
2. $t_i \leftarrow \arg \min_t \|x_i - p^\top \tilde{t}_i\|^2$, which can be solved by finding the roots of a polynomial of degree $2R - 1$.

Proof (sketch). For $\Sigma(t_i|\phi) = \phi I$, the negative log likelihood is, up to some additive and multiplicative constants:

$$l(p, \{t_i\}) = \|X - Tp\|_F^2 = X^\top X - 2X^\top Tp + p^\top T^\top Tp$$

To find the minimizer \hat{p} given t_1, \dots, t_n :

$$\nabla_p l = -2T^\top X + 2T^\top Tp = 0 \implies \hat{p} = (T^\top T)^{-1} T^\top X$$

To find the minimizer \hat{t}_i given p :

We can rewrite the negative log likelihood as $l(p, \{t_i\}) = \sum_i \|x_i - p^\top \tilde{t}_i\|^2$.

Then each entry of $\nabla_{t_i} l$ depends only on t_i , so each t_i can be optimized independently. Furthermore, each $\frac{\partial}{\partial t_i} \|x_i - p^\top \tilde{t}_i\|^2$ is a polynomial in t_i , so it can be minimized by simply finding the roots of the polynomial.

Thus this algorithm is a coordinate descent algorithm, which reduces the objective function with each iteration. \square

Theorem 2. Let $A^{(n)} \sim \text{RDPG}(F(p, \theta), n)$, and let $X^{(n)}$ be the ASE of $A^{(n)}$. Let $\hat{p}, \{\hat{t}_i\}$ be the global minimizer of l given $X^{(n)}$. Then $l(\hat{p}, \{\hat{t}_i\}) \xrightarrow{a.s.} 0$.

Covariance Structures

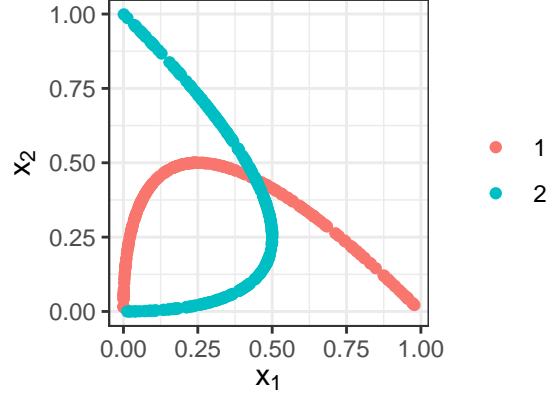
While the CLT property of the ASE allows us to approximate $x_i \sim \mathcal{N}(Wf(t_i|p), W\Sigma(t_i|\phi)W^\top)$, it is unclear what $\Sigma(t_i|\phi)$ looks like, and we cannot in general say $\Sigma(t_i|\phi) = \phi I_d$.

Clustering

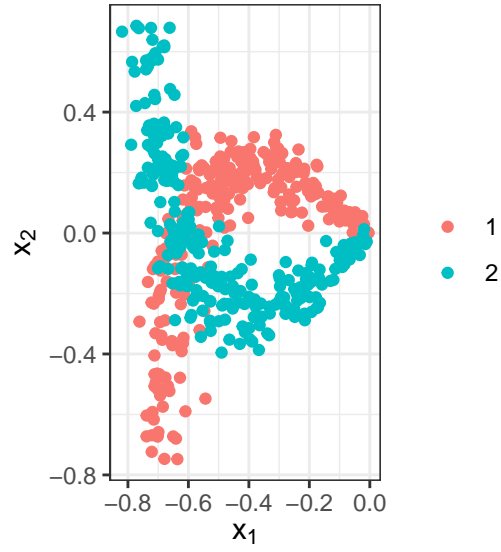
Next, suppose we have K curves parameterized by $g^{(k)}$, with points drawn along these curves. Then one possible clustering technique is as follows:

1. Assign an initial clustering (e.g., via spectral clustering).
2. Estimate the curve for each cluster (using the same curve-fitting procedure as before).
3. Reassign the clusters by proximity to each curve.
4. Repeat 2 and 3 until convergence.

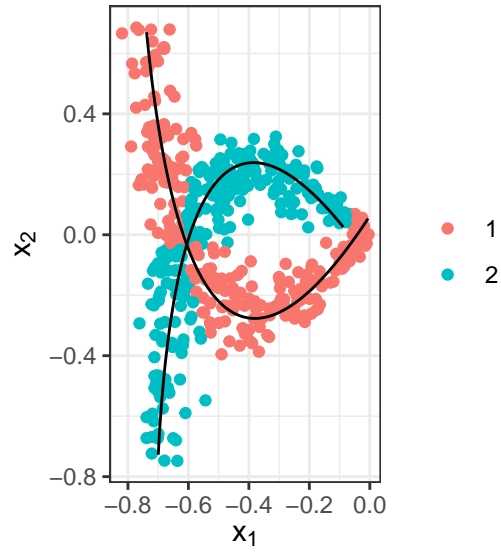
Example 4. We have two intersecting curves, $g_1(t) = \begin{bmatrix} t^2 & 2t(1-t) \end{bmatrix}^\top$ and $g_2(t) = \begin{bmatrix} 2t(1-t) & (1-t)^2 \end{bmatrix}^\top$. $n_1 = n_2 = 256$ points are drawn uniformly from each.



We draw $A \sim \text{RDPG}(X)$ and obtain the following ASE:



Fitting two quadratic Bezier curves to these data yields a community detection error rate of 10%. In the following plot, the points are labeled according to their estimated labels.



The objective function that this method aims to minimize is:

$$\sum_{k=1}^K \sum_{i \in C_k} \|x_i - g_k(t_i | p_k)\|^2$$

Where $g_k : [0, 1] \mapsto \mathbb{R}^d$ with parameters p_k is the curve for community k .

TBD: It can be shown that each iteration decreases the objective unless we are at a stationary point.

Harry Potter Emnity Graph

Example 5. *In this section, we analyze the Harry Potter emnity graph. Rubin-Delanchy et al. suggested applying the GRDPG ASE to this graph with $p = 1$ and $q = 1$, revealing a DCBM-like structure in the latent space. Passino and Heard instead fit Gaussian processes to the embedding. Here, we consider what happens when we apply the Bezier curve clustering method.*

