

A CONVERGENCE OF THE FCM CLUSTERING ALGORITHM

We first need to consider our separate updates as a single update procedure. Let $F : \mathbb{M} \mapsto R$ be defined by (1) and $G : R \mapsto \mathbb{M}$ be defined by (2), and for $R = \{r_{jk}\}$ and $\mathbb{M} = \{\mathbb{M}_k\}$ consider the sequence

$$\left\{ T^{(l)}(R, \mathbb{M}) : l = 0, 1, \dots \right\}, \text{ where } T(R, M) = (F \circ G(R), G(R)).$$

We wish to show convergence of the iterates of T to a local minimum or saddle point of the cost function

$$J(R, \mathbb{M}) = \sum_{j=1}^n \sum_{k=1}^c r_{jk}^2 W_2(\mathbb{M}_k, \mathbb{D}_j)^2.$$

The two stage update process of T is too complicated to use standard fixed point theorems, so as in [Bezdek \(1980\)](#) we shall use the following result, which is proven in [Zangwill \(1969\)](#).

Theorem 3 (Zangwill’s Convergence Theorem). *Let $A : X \rightarrow 2^X$ be a point-to-set algorithm acting on X . Given $x_0 \in X$, generate a sequence $\{x_k\}_{k=1}^\infty$ such that $x_{k+1} \in A(x_k)$ for every k . Let $\Gamma \subset X$ be a solution set, and suppose that the following hold.*

- (i) *The sequence $\{x_k\} \subset S \subset X$ for a compact set S .*
- (ii) *There exists a continuous function Z on X such that if $x \notin \Gamma$ then $Z(y) < Z(x)$ for all $y \in A(x)$, and if $x \in \Gamma$ then $Z(y) \leq Z(x)$ for all $y \in A(x)$. The function Z is called a descent function.*
- (iii) *The algorithm A is closed on $X \setminus \Gamma$.*

Then every convergent subsequence of $\{x_k\}$ tends to a point in the solution set Γ .

Our algorithm is the update function T . We define our solution set as

$$\Gamma = \left\{ (R, \mathbb{M}) : J(R, \mathbb{M}) < J(\hat{R}, \hat{\mathbb{M}}) \forall (\hat{R}, \hat{\mathbb{M}}) \in B((R, \mathbb{M}), r) \right\}$$

for some $r > 0$, where the ball surrounding R is the Euclidean ball in \mathbb{R}^{nc} and the ball surrounding \mathbb{M} is $\cup_{k=1}^c B_{W_2}(\mathbb{M}_k, r)$. This set contains the local minima and saddle points of the cost function [\(Bezdek et al., 1987\)](#). We wish to show that our cost function $J(R, \mathbb{M})$ is the descent function Z . We proceed by verifying each of the requirements for Zangwill’s Convergence Theorem.

Lemma 4. *Every iterate $T^{(l)}(R, \mathbb{M}) \in [0, 1]^{nc} \times \text{conv}(\mathbb{D})^c$, where*

$$\text{conv}(\mathbb{D}) = \bigcup_{k=1}^c \bigcup_{\gamma_j}^m \text{conv}\{\gamma_j(y^{(i)}) : j = 1, \dots, n\},$$

with γ_j a bijection $\mathbb{M}_k \rightarrow \mathbb{D}_j$ and $\text{conv}\{\gamma_j(y^{(i)}) : j = 1, \dots, n\}$ the ordinary convex hull in the plane. Furthermore, $[0, 1]^{nc} \times \text{conv}(\mathbb{D})^c$ is compact.

Proof. By construction, every $r_{jk} \in [0, 1]$. Since $j = 1, \dots, n$ and $k = 1, \dots, c$, we can view R as a point in $[0, 1]^{nc}$, and so every iterate of R is in $[0, 1]^{nc}$. We shall show that for a fixed k and a fixed bijection $\gamma_j : \mathbb{M}_k \rightarrow \mathbb{D}_j$, each updated $y^{(i)}$ is contained in a convex combination of $\{\gamma_j(y^{(i)}) : j = 1, \dots, n\}$. Where $\gamma_j(y^{(i)}) = \Delta$, let $\gamma_j(y^{(i)}) = w_\Delta$ as defined in (4), as this is the update point we use for the diagonal. Since there are a finite number of off-diagonal points, each updated \mathbb{M}_k is therefore contained in the union over all bijections and all points $y^{(i)}$ of the convex combination of $\{\gamma_j(y^{(i)}) : j = 1, \dots, n\}$. By also taking the union over each k , we show that every iterate of \mathbb{M} must be contained in the finite triple-union of the convex combination of each possible bijection. To show that each updated $y^{(i)}$ is contained in a convex combination of $\{\gamma_j(y^{(i)}) : j = 1, \dots, n\}$, recall that $y^{(i)} = \left(\sum_{j=1}^n r_{jk}^2 \right)^{-1} \sum_{j=1}^n r_{jk}^2 \gamma_j(y^{(i)})$. Letting $t_j^{(i)} = r_{jk}^2 \left(\sum_{j=1}^n r_{jk}^2 \right)^{-1}$, clearly each $t_j^{(i)} > 0$ and $\sum_{j=1}^n t_j^{(i)} = 1$. Since $y^{(i)} = \sum_{j=1}^n t_j^{(i)} \gamma_j(y^{(i)})$, each $y^{(i)}$ is contained in the convex combination. Therefore $T^{(l)}(R, \mathbb{M}) \in [0, 1]^{nc} \times \text{conv}(\mathbb{D})^c$ for each $l = 0, 1, \dots$

Now, $[0, 1]$ is closed and bounded, so is compact. The convex hull of points in the plane is closed and bounded, so $\text{conv}\{\gamma_j(y^{(i)}) : j = 1, \dots, n\}$ is compact. Since finite unions and finite direct products of compact sets are compact, $[0, 1]^{nc} \times \text{conv}(\mathbb{D})^c$ is also compact. \square

Lemma 5. *The cost function $J(R, \mathbb{M})$ is a descent function, as defined in Theorem 3(ii).*

Proof. The cost function J is continuous, as it's a sum, product and composition of continuous functions. Furthermore, we have that for any $(R, \mathbb{M}) \notin \Gamma$,

$$J(T(R, \mathbb{M})) = J(F \circ G(R), G(R)) < J(R, G(R)) < J(R, M),$$

where the first inequality is due to Proposition 1 in [Bezdek \(1980\)](#), and the second inequality comes from the definition of the Fréchet mean. If $(R, \mathbb{M}) \in \Gamma$ then the strict inequalities include equality throughout. \square

Theorem 6. *For any (R, \mathbb{M}) , every convergent subsequence of $\{T^{(l)}(R, \mathbb{M}) : l = 0, 1, \dots\}$ tends to a local minimum or saddle point of the cost function J .*

Proof. We proceed with Zangwill's Convergence Theorem. Our algorithm is the update function T , our solution set is Γ , and our descent function is the cost function $J(R, \mathbb{M})$. By Lemma 4, every iterate is contained within a compact set. By Lemma 5, J is a descent function. Finally, since our function T only maps points in the plane to points in the plane, it is a closed map. The theorem follows by applying Theorem 3. \square

B CONVERGENCE OF THE FRÉCHET MEAN ALGORITHM

Recall that the Fréchet mean is computed by finding the arg min of

$$F(\hat{\mathbb{D}}) = \sum_{j=1}^n r_{jk}^2 F_j(\hat{\mathbb{D}}), \text{ with } F_j(\hat{\mathbb{D}}) = W_2(\hat{\mathbb{D}}, \mathbb{D}_j)^2, \quad (5)$$

for fixed k . We start by recounting work in [Turner et al. \(2012\)](#), which this section adapts for the weighted Fréchet mean.⁴ The proofs we're adapting use a gradient descent technique to prove local convergence. In order to use their techniques, we need to define a differential structure on the space of persistence diagrams.

By Theorem 2.5 in [Turner et al. \(2012\)](#), the space of persistence diagrams $\mathcal{D}_{L^2} = \{\mathbb{D} : W_2(\mathbb{D}, \Delta) < \infty\}$ is a non-negatively curved Alexandrov space. An optimal bijection $\gamma : \mathbb{D}_1 \rightarrow \mathbb{D}_2$ induces a unit-speed geodesic $\phi(t) = \{(1-t)x + t\gamma(x) : x \in \mathbb{D}_1, 0 \leq t \leq 1\}$. For a point $\mathbb{D} \in \mathcal{D}_{L^2}$ we define the tangent cone $T_{\mathbb{D}}$. Define $\hat{\Sigma}_{\mathbb{D}}$ as the set of all non-trivial unit-speed geodesics emanating from \mathbb{D} . Let $\phi, \eta \in \hat{\Sigma}_{\mathbb{D}}$ and define the angle between them as

$$\angle_{\mathbb{D}}(\phi, \eta) = \arccos \left(\lim_{s, t \downarrow 0} \frac{s^2 + t^2 - W_2(\phi(s), \eta(t))^2}{2st} \right) \in [0, \pi]$$

when the limit exists. Then the space of directions $(\Sigma_{\mathbb{D}}, \angle_{\mathbb{D}})$ is the completion of $\hat{\Sigma}_{\mathbb{D}} / \sim$ with respect to $\angle_{\mathbb{D}}$, with $\phi \sim \eta \iff \angle_{\mathbb{D}}(\phi, \eta) = 0$. We now define the tangent cone as

$$T_{\mathbb{D}} = (\Sigma_{\mathbb{D}} \times [0, \infty)) / (\Sigma_{\mathbb{D}} \times \{0\}).$$

Given $u = (\phi, s), v = (\eta, t)$, we define an inner product on the tangent cone by

$$\langle u, v \rangle = st \cos \angle_{\mathbb{D}}(\phi, \eta).$$

Now, for $\alpha > 0$ denote the space $(\mathcal{D}_{L^2}, \alpha W_2)$ as $\alpha \mathcal{D}_{L^2}$ and define the map $i_{\alpha} : \alpha \mathcal{D}_{L^2} \rightarrow \mathcal{D}_{L^2}$. For an open set $\Omega \subset \mathcal{D}_{L^2}$ and a function $f : \Omega \rightarrow \mathbb{R}$, the differential of f at $\mathbb{D} \in \Omega$ is defined by $d_{\mathbb{D}}f = \lim_{\alpha \rightarrow \infty} \alpha(f \circ i_{\alpha} - f(\mathbb{D}))$. Finally, we say that $s \in T_{\mathbb{D}}$ is a supporting vector of f at \mathbb{D} if $d_{\mathbb{D}}f(x) \leq -\langle s, x \rangle$ for all $x \in T_{\mathbb{D}}$.

⁴In [Turner et al. \(2012\)](#), the Fréchet mean is defined as the arg min of the Fréchet function $F(\hat{\mathbb{D}}) = \int W_2(\mathbb{D}, \mathbb{D}_j)^2 d\rho(\mathbb{D})$ with the empirical measure $\rho = n^{-1} \sum_{j=1}^n \delta_{\mathbb{D}_j}$. We are using the empirical measure $\rho = \left(\sum_{j=1}^n r_{jk}^2 \right)^{-1} \sum_{j=1}^n r_{jk}^2 \delta_{\mathbb{D}_j}$, but for ease we drop the scalar $\left(\sum_{j=1}^n r_{jk}^2 \right)^{-1}$ as it is positive, so doesn't affect the minimum of the function.

Lemma 7. *The following two results are proven in [Turner et al. \(2012\)](#).*

- (i) *Let $\mathbb{D} \in \mathcal{D}_{L^2}$. Let $F_j(\hat{\mathbb{D}}) = W_2(\hat{\mathbb{D}}, \mathbb{D}_j)^2$. Then if ϕ is a distance-achieving geodesic from \mathbb{D} to $\hat{\mathbb{D}}$, then the tangent vector to ϕ at \mathbb{D} of length $2W_2(\hat{\mathbb{D}}, \mathbb{D})$ is a supporting vector at \mathbb{D} of f .*
- (ii) *If \mathbb{D} is a local minimum of f and s is a supporting vector of f at \mathbb{D} , then $s = 0$.*

If there is a unique optimal matching $\gamma_{\mathbb{D}_1}^{\mathbb{D}_3} : \mathbb{D}_1 \rightarrow \mathbb{D}_3$, we say that it is induced by an optimal matching $\gamma_{\mathbb{D}_1}^{\mathbb{D}_2} : \mathbb{D}_1 \rightarrow \mathbb{D}_2$ if there exists a unique optimal matching $\gamma_{\mathbb{D}_2}^{\mathbb{D}_3} : \mathbb{D}_2 \rightarrow \mathbb{D}_3$ such that $\gamma_{\mathbb{D}_1}^{\mathbb{D}_3} = \gamma_{\mathbb{D}_2}^{\mathbb{D}_3} \circ \gamma_{\mathbb{D}_1}^{\mathbb{D}_2}$. Proposition 3.2 from [Turner et al. \(2012\)](#) states that an optimal matching at a point is also locally optimal. In particular, it states the following.

Lemma 8. *Let $\mathbb{D}_1, \mathbb{D}_2 \in \mathcal{D}_{L^2}$ such that there is a unique optimal matching from \mathbb{D}_1 to \mathbb{D}_2 . Then there exists an $r > 0$ such that for every $\mathbb{D}_3 \in B_{W_2}(\mathbb{D}_2, r)$, there is a unique optimal pairing from \mathbb{D}_2 to \mathbb{D}_3 that is induced by the matching from \mathbb{D}_1 to \mathbb{D}_2 .*

The following theorem proves that our algorithm converges to a local minimum of the Fréchet function.

Theorem 9. *Given diagrams \mathbb{D}_j , membership values r_{jk} , and the Fréchet function F defined in (5), then $\mathbb{M}_k = \{y^{(i)}\}_{i=1}^m$ is a local minimum of F if and only if there is a unique optimal pairing from \mathbb{M}_k to each of the \mathbb{D}_j , denoted γ_j , and each $y^{(i)}$ is updated via (4).*

Proof. First assume that γ_j are optimal pairings from \mathbb{M}_k to each \mathbb{D}_j , and let s_j be the vectors in $T_{\mathbb{M}_k}$ that are tangent to the geodesics induced by γ_j and are distance-achieving. Then by Lemma 7(i), each $2s_j$ is a supporting vector for the function F_j . Furthermore, $2 \sum_{j=1}^n r_{jk}^2 s_j$ is a supporting vector for F , as for any $\hat{\mathbb{D}}$,

$$\begin{aligned} d_{\mathbb{M}_k} F(\hat{\mathbb{D}}) &= d_{\mathbb{M}_k} \left(\sum_{j=1}^n r_{jk}^2 F_j(\hat{\mathbb{D}}) \right) = \sum_{j=1}^n r_{jk}^2 d_{\mathbb{M}_k} F_j(\hat{\mathbb{D}}) \\ &\leq \sum_{j=1}^n -r_{jk}^2 \langle 2s_j, \hat{\mathbb{D}} \rangle = - \left\langle 2 \sum_{j=1}^n r_{jk}^2 s_j, \hat{\mathbb{D}} \right\rangle. \end{aligned}$$

By Lemma 7(ii), $2 \sum_{j=1}^n r_{jk}^2 s_j = 0$. Putting $s_j = \gamma_j(y^{(i)}) - y^{(i)}$ and rearranging gives that $y^{(i)}$ updates via (4), as required. Note that when $\gamma_j(y^{(i)}) = \Delta$, we let $\gamma_j(y^{(i)}) = w_\Delta$ as defined in (4), because this minimises the transportation cost to the diagonal. Now suppose that γ_j and $\tilde{\gamma}_j$ are both optimal pairings. Then by the above argument $\sum_{j=1}^n r_{jk}^2 s_j = \sum_{j=1}^n r_{jk}^2 \tilde{s}_j = 0$, implying that $s_j = \tilde{s}_j$ and so $\gamma_j = \tilde{\gamma}_j$. Therefore the optimal pairing is unique.

To prove the opposite direction, assume that $\mathbb{M}_k = \{y^{(i)}\}$ locally minimises the Fréchet function F . Observe that for a fixed bijection γ_j , we have that

$$\begin{aligned} F(\mathbb{M}_k) &= \sum_{j=1}^n r_{jk}^2 W_2(\mathbb{M}_k, \mathbb{D}_j)^2 \\ &= \sum_{j=1}^n r_{jk}^2 \left(\inf_{\gamma_j : \mathbb{M} \rightarrow \mathbb{D}_j} \sum_{y \in \mathbb{M}_k} \|y - \gamma_j(y)\|^2 \right) \\ &= \sum_{j=1}^n r_{jk}^2 \sum_{i=1}^m \|y^{(i)} - x_j^{(i)}\|^2 \\ &= \sum_{i=1}^m \left(\sum_{j=1}^n r_{jk}^2 \|y^{(i)} - x_j^{(i)}\|^2 \right). \end{aligned}$$

The final term in brackets is non-negative, and minimised exactly when $y^{(i)}$ is updated via (4). Furthermore, the unique optimal pairing from \mathbb{M}_k to each of the \mathbb{D}_j 's is the same for every $\hat{\mathbb{M}}$ within the ball $B_{W_2}(\mathbb{M}_k, r)$ for some $r > 0$, by Lemma 8. Therefore, if \mathbb{M}_k is a local minimum of F , then the $y^{(i)}$'s are equal to the values found by taking the optimal pairings γ_j and calculating the weighted means of $\gamma_j(y^{(i)})$ with the weights r_{jk}^2 , as required. It will remain a minimum as long as the matching stays the same, which happens in the ball $B_{W_2}(\mathbb{M}_k, r)$, so we are done. \square

Algorithm 2 WFrechetMean

Input Diagrams $\mathbb{D} = \{\mathbb{D}_j\}_{j=1}^n$, Weights $R_k = \{r_{jk}\}_{j=1}^n$ (fixed k)
Output Weighted Fréchet mean $\mathbb{M}_k = \{y^{(i)}\}_{i=1}^m$

```

1:  $m \leftarrow \max_{1 \leq j \leq n} |\mathbb{D}_j|$ 
2: for  $j$  in  $1..n$  do
3:    $\left[ x_j^{(i)} \right]_{i=1}^m \leftarrow \text{HUNGARIAN}(\mathbb{M}_k, \mathbb{D}_j)$ 
4: end for
5: while  $\left\{ \left[ x_j^{(i)} \right]_{i=1}^m \right\}_{j=1}^n \neq \left\{ \left[ \hat{x}_j^{(i)} \right]_{i=1}^m \right\}_{j=1}^n$ 
6:   do
7:     for  $i$  in  $1..m$  do
8:        $\mathcal{J}_{\text{OD}}^{(i)} = \{j : x_j^{(i)} \neq \Delta\}$ 
9:        $\mathcal{J}_{\text{D}}^{(i)} = \{j : x_j^{(i)} = \Delta\}$ 
10:      if  $\mathcal{J}_{\text{OD}}^{(i)} = \emptyset$  then
11:         $y^{(i)} \leftarrow \Delta$ 
12:      else
13:         $w = \left( \sum_{j \in \mathcal{J}_{\text{OD}}^{(i)}} r_{jk}^2 \right)^{-1} \sum_{j \in \mathcal{J}_{\text{OD}}^{(i)}} r_{jk}^2 x_j^{(i)}$ 
14:        if  $\mathcal{J}_{\text{D}}^{(i)} = \emptyset$  then
15:           $y^{(i)} \leftarrow w$ 
16:        else
17:           $y^{(i)} \leftarrow \frac{\sum_{j \in \mathcal{J}_{\text{OD}}^{(i)}} r_{jk}^2 x_j^{(i)} + \sum_{j \in \mathcal{J}_{\text{D}}^{(i)}} r_{jk}^2 w \Delta}{\sum_{j=1}^n r_{jk}^2}$ 
18:        end if
19:      end for
20:       $\left\{ \left[ \hat{x}_j^{(i)} \right]_{i=1}^m \right\}_{j=1}^n \leftarrow \left\{ \left[ x_j^{(i)} \right]_{i=1}^m \right\}_{j=1}^n$ 
21:    for  $j$  in  $1..n$  do
22:       $\left[ x_j^{(i)} \right]_{i=1}^m \leftarrow \text{HUNGARIAN}(\mathbb{M}_k, \mathbb{D}_j)$ 
23:    end for
24:  end while
25: return  $\{y^{(i)}\}_{i=1}^m$ 

```

C EXPERIMENTAL DETAILS

C.1 SYNTHETIC DATA

Membership values. The membership values for the synthetic datasets are in Table 2. Datasets 1-3 are the datasets of noise, datasets 4-6 are the datasets with one ring, and datasets 7-9 are the datasets with two rings. We ran our algorithm for 20 iterations.

Table 2: Membership values for the synthetic dataset

Dataset	1	2	3	4	5	6	7	8	9
Cluster 1	0.6336	0.5730	0.5205	0.2760	0.2503	0.1974	0.2921	0.2128	0.2292
Cluster 2	0.1768	0.2057	0.2327	0.5361	0.5329	0.6371	0.2452	0.2291	0.1822
Cluster 3	0.1900	0.2212	0.2468	0.1879	0.2169	0.1655	0.4627	0.5580	0.5885

Timing experiments. For the timing experiments we divide the total number of points equally between four distributions, two of which are noise and two of which are shaped in a ring. Each clustering algorithm was run for five iterations on one core of a 2019 MacBook Pro with a 1.4GHz

Table 3: Seconds per clustering iteration

Points	100	200	300	400	500	600	700	800	900	1000
FPDCluster	0.01552	0.1975	0.9358	2.229	5.694	12.29	19.27	34.50	53.20	77.81
ADMM	5.622	34.86	161.3	617.6	-	-	-	-	-	-
BADMM	0.2020	2.188	26.38	112.6	-	-	-	-	-	-
SubGD	0.4217	2.273	22.17	103.4	-	-	-	-	-	-
IterBP	0.3825	2.226	21.57	108.9	-	-	-	-	-	-
LP	0.3922	2.031	22.32	117.3	-	-	-	-	-	-

Intel Core i5. We included the time taken to compute the persistence diagrams in the running times for our algorithm.

We also use synthetic data to empirically compare the running time of our algorithm to other dataset clustering algorithms available. Computing the Wasserstein distance has super-cubic time complexity Ling & Okada (2007), so is a significant bottleneck both for our algorithm and comparable Wasserstein barycentre clustering algorithms Benamou et al. (2015); Cuturi & Doucet (2014); Li & Wang (2008); Ye & Li (2014); Ye et al. (2017). Persistence diagrams generally reduce both the dimensionality and number of points in a dataset,⁵ so we in turn reduce the computational bottleneck. To demonstrate this, we evaluated the average time per iteration of our persistence diagram clustering algorithm, as well as the average iteration time for comparable Wasserstein barycentre clustering algorithms. We included the time taken to compute the persistence diagrams from the datasets when timing our clustering algorithm. We give the results in Table 3, leaving an entry blank where it became unpractical to run a test (e.g. it takes too long to return a solution and the algorithm becomes unresponsive). We show at least an order of magnitude improvement in performance over comparable Wasserstein barycentre clustering algorithms.

C.2 LATTICE STRUCTURES

The results obtained are in Tables 5-8. The fuzzy values for FPDCluster are given as floats, although in each case they converged to an absolute cluster. The Wasserstein barycentre clustering algorithms each have discrete labels. The correct labellings are for 1-3 and 4-6 to be clustered together in each case. We clustered the 2-PH diagrams. We denote a label as having been assigned by 1, or not assigned by 0. We ran each algorithm for five iterations. We obtained our datasets as cif files, converted them to xyz files, and then to csv files, producing a list of the coordinates of each atom in \mathbb{R}^3 . We create three copies of each structure. For rotation, we rotate two of them by 180° around different axes. For reflection, we reflect two of them in different axes. For translation, we translate them up or down by the length of the unit-cell. We use our own python implementation of FPDCluster, available in the supplementary materials. For each of the other algorithms, we use the implementation provided at https://github.com/bobye/WBC_Matlab, a copy of which is in the supplementary materials. We do not limit the number of points in the diagram when clustering.

C.3 DECISION BOUNDARIES

Why hard clustering doesn’t work. In order to assign each task to the top-ranked models, we need to have a path from a task to the nearest cluster centre, then from that cluster centre to the k -nearest models (note that when we refer to models/tasks, we’re implicitly referring to the persistence diagram of their decision boundary). We can always find that route when fuzzy clustering, as the fractional membership values mean that we have information about the proximity of every model/task with every cluster centre. However, with hard clustering we cannot always find that route. Firstly, the hard labelling means that you lose a lot of information about the proximity of models/tasks to cluster centres. Therefore, in order to find a route, we need a every task to be assigned to a cluster centre

⁵Persistence diagrams are always planar, so if the data is in \mathbb{R}^d , $d > 2$, then there is a dimensionality reduction. For $p > 0$, the persistence diagram of p -PH always has less points than the dataset when computed with the Rips complex.

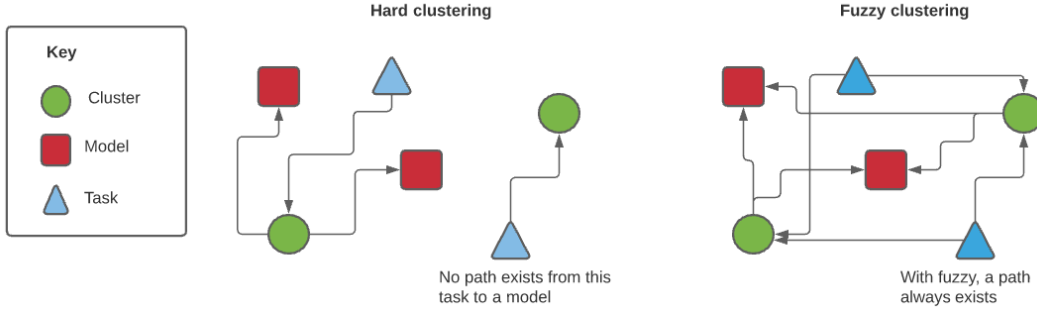


Figure 4: With hard clustering, we cannot always find a path from a task to a model.

that also has a model assigned to it. However, there are no guarantees that will happen. We show an example where no path exists in Figure 4.

Experimental details. All code used for computation is available in the supplementary materials. For models, we trained the standard Pytorch CNN available at <https://github.com/pytorch/examples/blob/master/mnist/main.py>. We trained them on MNIST, FashionMNIST, and KMNIST, each obtained using the Torchvision.datasets package. We split the data into 9 binary datasets for classification, class 0 vs each of the remaining classes. We trained three of each model, seeded with 0, 1, and 2 respectively. MNIST and KMNIST were each trained for five epochs, FashionMNIST was trained for 14 epochs. We used Ripser to compute the 1-persistence diagrams using default settings. We limited the number of points in the diagram to the 25 most persistent when clustering. Our percentage improvement values use the membership values after 16 iterations. We compute the standard error bounds when calculating the percentage improvement.

Table 4: Clustering results after transformation

	Cubic Structures				Carbon Allotropes			
	None	Rotate	Reflect	Translate	None	Rotate	Reflect	Translate
FPDCluster	✓	✓	✓	✓	✓	✓	✓	✓
ADMM	✓	✗	✓	✗	✓	✗	✗	✗
BADMM	✓	✗	✓	✗	✓	✗	✗	✗
SubGD	✓	✗	✓	✗	✓	✗	✗	✗
IterBP	✓	✗	✓	✗	✓	✗	✗	✗
LP	✓	✗	✓	✗	✓	✗	✗	✗

Table 5: Membership values for non-transformed datasets

		Cubic Structure Datasets						Carbon Allotrope Datasets					
		1	2	3	4	5	6	1	2	3	4	5	6
FPDCluster	Cluster 1	1.000	1.000	1.000	0.000	0.000	0.000	1.000	1.000	1.000	0.000	0.000	0.000
	Cluster 2	0.000	0.000	0.000	1.000	1.000	1.000	0.000	0.000	0.000	1.000	1.000	1.000
ADMM	Cluster 1	1	1	1	0	0	0	1	1	1	0	0	0
	Cluster 2	0	0	0	1	1	1	0	0	0	1	1	1
BADMM	Cluster 1	1	1	1	0	0	0	1	1	1	0	0	0
	Cluster 2	0	0	0	1	1	1	0	0	0	1	1	1
SubGD	Cluster 1	1	1	1	0	0	0	1	1	1	0	0	0
	Cluster 2	0	0	0	1	1	1	0	0	0	1	1	1
IterBP	Cluster 1	1	1	1	0	0	0	1	1	1	0	0	0
	Cluster 2	0	0	0	1	1	1	0	0	0	1	1	1
LP	Cluster 1	1	1	1	0	0	0	1	1	1	0	0	0
	Cluster 2	0	0	0	1	1	1	0	0	0	1	1	1

Table 6: Membership values for rotated datasets

		Cubic Structure Datasets						Carbon Allotrope Datasets					
		1	2	3	4	5	6	1	2	3	4	5	6
FPDCluster	Cluster 1	1.000	1.000	1.000	0.000	0.000	0.000	1.000	1.000	1.000	0.000	0.000	0.000
	Cluster 2	0.000	0.000	0.000	1.000	1.000	1.000	0.000	0.000	0.000	1.000	1.000	1.000
ADMM	Cluster 1	0	0	1	0	0	1	1	0	1	1	0	1
	Cluster 2	1	1	0	1	1	0	0	1	0	0	1	0
BADMM	Cluster 1	0	1	1	0	1	1	1	1	0	1	1	0
	Cluster 2	1	0	0	1	0	0	0	0	1	0	0	1
SubGD	Cluster 1	0	1	1	0	1	1	1	0	0	1	0	0
	Cluster 2	1	0	0	1	0	0	0	1	1	0	1	1
IterBP	Cluster 1	0	1	0	0	1	0	0	1	1	0	1	1
	Cluster 2	1	0	1	1	0	1	1	0	0	1	0	0
LP	Cluster 1	1	0	1	1	0	1	0	1	0	0	1	0
	Cluster 2	0	1	0	0	1	0	1	0	1	1	0	1

Table 7: Membership values for reflected datasets

		Cubic Structure Datasets						Carbon Allotrope Datasets					
		1	2	3	4	5	6	1	2	3	4	5	6
FPDCluster	Cluster 1	1.000	1.000	1.000	0.000	0.000	0.000	1.000	1.000	1.000	0.000	0.000	0.000
	Cluster 2	0.000	0.000	0.000	1.000	1.000	1.000	0.000	0.000	0.000	1.000	1.000	1.000
ADMM	Cluster 1	1	1	1	0	0	0	0	0	0	1	1	0
	Cluster 2	0	0	0	1	1	1	1	1	1	0	0	1
BADMM	Cluster 1	1	1	1	0	0	0	0	0	0	1	1	0
	Cluster 2	0	0	0	1	1	1	1	1	1	0	0	1
SubGD	Cluster 1	1	1	1	0	0	0	1	1	1	1	1	0
	Cluster 2	0	0	0	1	1	1	0	0	0	0	0	1
IterBP	Cluster 1	1	1	1	0	0	0	0	0	0	0	0	1
	Cluster 2	0	0	0	1	1	1	1	1	1	1	1	0
LP	Cluster 1	1	1	1	0	0	0	0	0	0	0	1	1
	Cluster 2	0	0	0	1	1	1	1	1	1	1	0	0

Table 8: Membership values for translated datasets

		Cubic Structure Datasets						Carbon Allotrope Datasets					
		1	2	3	4	5	6	1	2	3	4	5	6
FPDCluster	Cluster 1	1.000	1.000	1.000	0.000	0.000	0.000	1.000	1.000	1.000	0.000	0.000	0.000
	Cluster 2	0.000	0.000	0.000	1.000	1.000	1.000	0.000	0.000	0.000	1.000	1.000	1.000
ADMM	Cluster 1	0	0	1	0	0	1	0	0	1	0	0	1
	Cluster 2	1	1	0	1	1	0	1	1	0	1	1	0
BADMM	Cluster 1	0	1	0	1	1	0	1	1	0	1	1	0
	Cluster 2	1	0	1	0	0	1	0	0	1	0	0	1
SubGD	Cluster 1	0	0	1	0	0	1	0	1	0	0	1	0
	Cluster 2	1	1	0	1	1	0	1	0	1	1	0	1
IterBP	Cluster 1	0	1	0	0	1	0	0	0	1	0	0	1
	Cluster 2	1	0	1	1	0	1	1	1	0	1	1	0
LP	Cluster 1	0	0	1	0	0	1	1	0	1	1	0	1
	Cluster 2	1	1	0	1	1	0	0	1	0	0	1	0