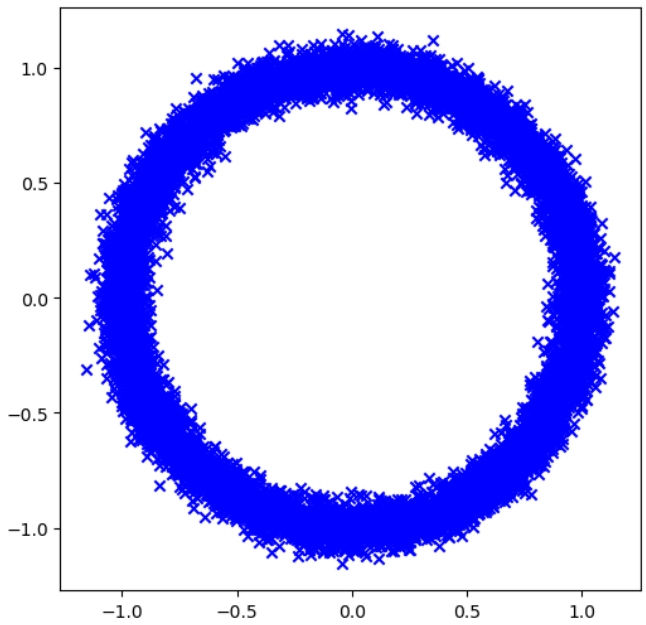
Estimation and Quantization of Expected Persistence Diagrams

Paizhe Xie

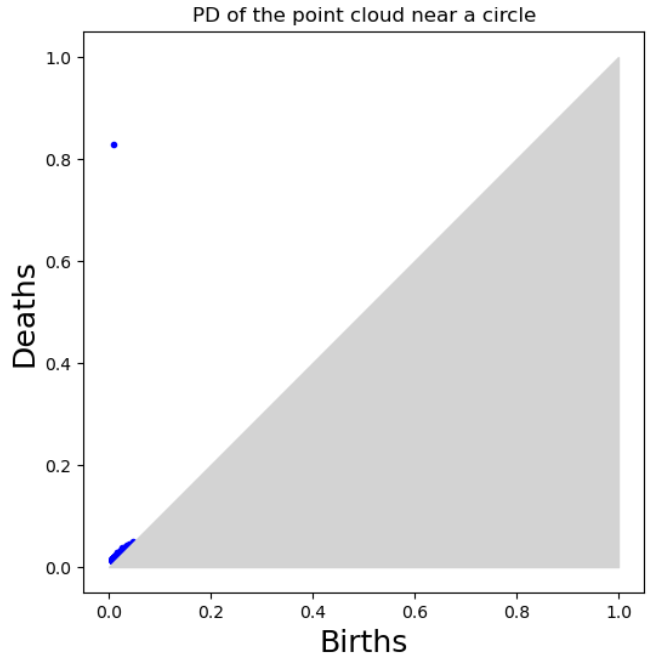
Department of Statistics

May 20, 2024

Given an point cloud in an Euclidean space $\mathbb{R}^d$, we can compute its persistence diagram using Cech filtration, for example, we can compute the persistence diagram of a point cloud sampled near a circle. As we can see, the PD is really 'good'.

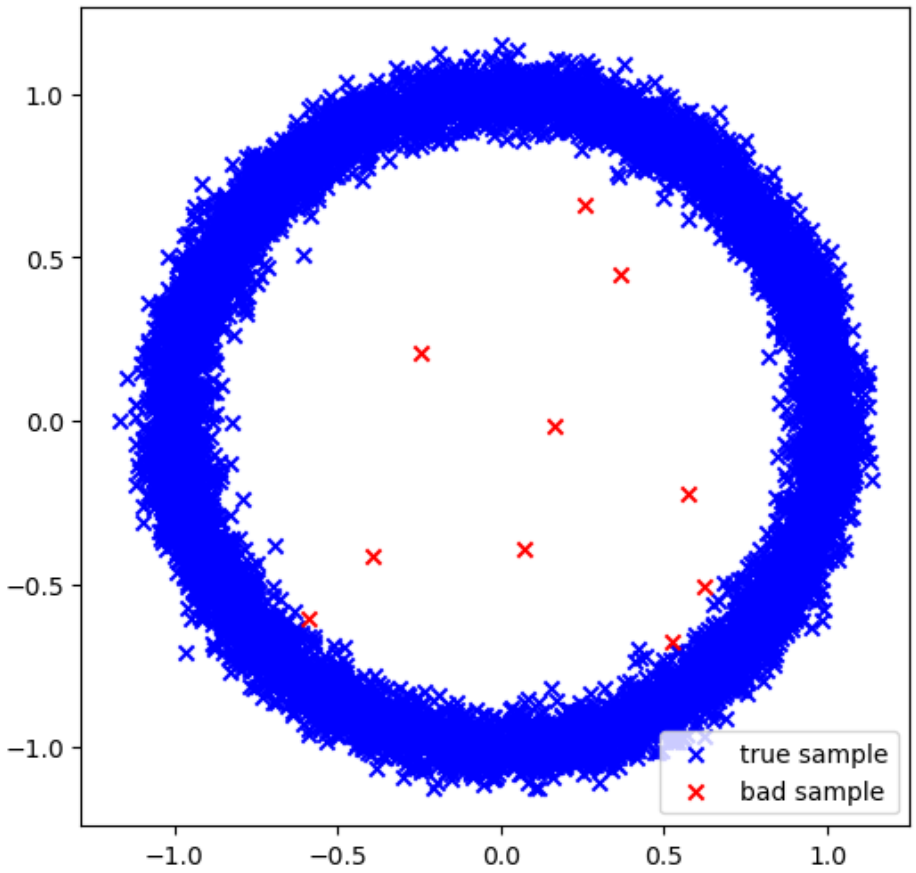


10000 points near unit circle



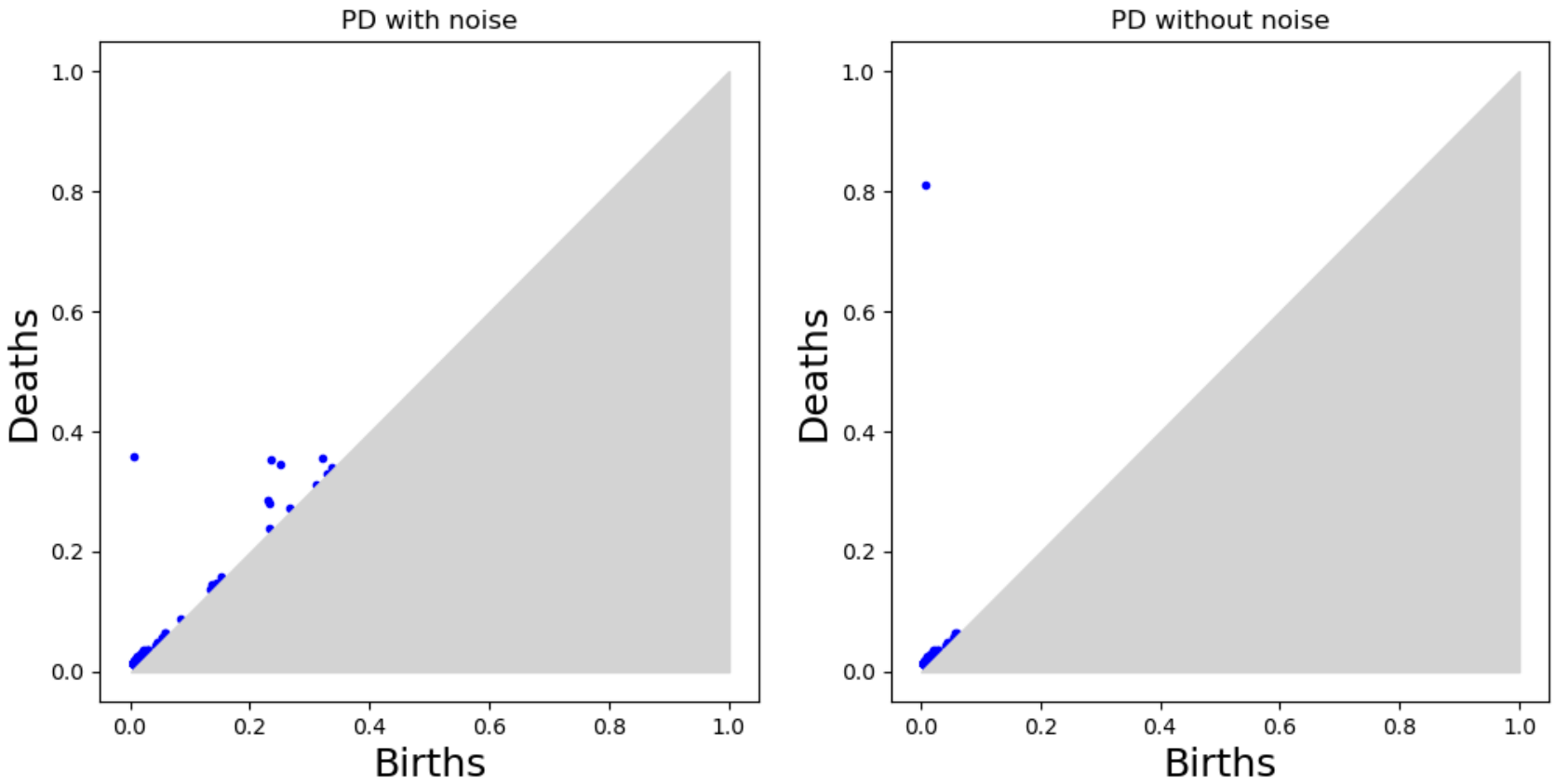
PD of the point cloud

However, if there is some random noise, even a relatively small number of bad samples can cause a big change in the persistence diagram, i.e. the persistence diagram is not robust. In this example we generate 10000 points and 10 of them are 'bad samples'.



9990 points near a circle and 10 'bad samples'

With these random noise, the persistence diagram would be 'chaotic', and it cannot reveal the topology of the true point cloud:



Persistence Diagram with and without noise

So a natural question we would ask is: how to recover the 'true' PD given the presence of noise?

Since we only have 10 bad samples among 10000 samples, if we sample around 100 samples from our dataset, it is likely that there is no bad samples in it and we can obtain a 'good' persistence diagram.

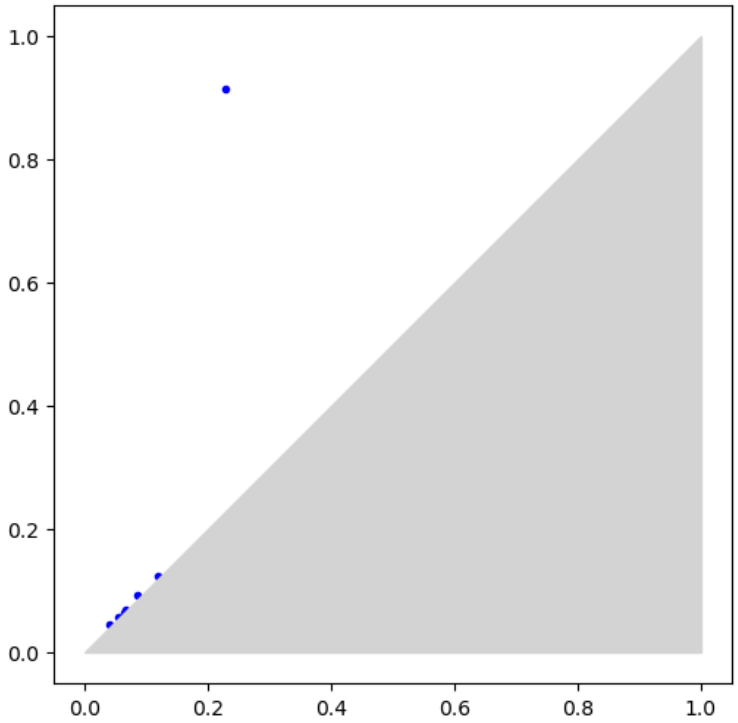


Figure: Persistence diagram of 100 points sampled from the 10000 point dataset

But we cannot use only this PD to 'recover' the true PD. This lead to the idea of expected persistence diagram(EPD).

To obtain the expectation of PDs, we must first formulate the persistence diagrams in a way that we can do computations on. Since PDs are basically points in the region: $\Omega = \{(t_1, t_2)|t_1 < t_2\} \in R^2$, we can view a persistence diagram as a discrete measure supported on Ω: $\mu = \sum_{i \in I} \delta_{x_i}$ where each x_i is a point in the persistence diagram and δ_{x_i} is the Dirac mass on that point. Then given n persistence diagrams $\mu_1, \ldots, \mu_n$, we can compute the empirical EPD by: $\bar{\mu}_n = \frac{1}{n} \sum_{i=1}^{n} \mu_i$. For the example above, we can compute the empirical EPD by sampling from the data.

Here is the resulting EPD from resampling with size 70 and repeat 100 times.

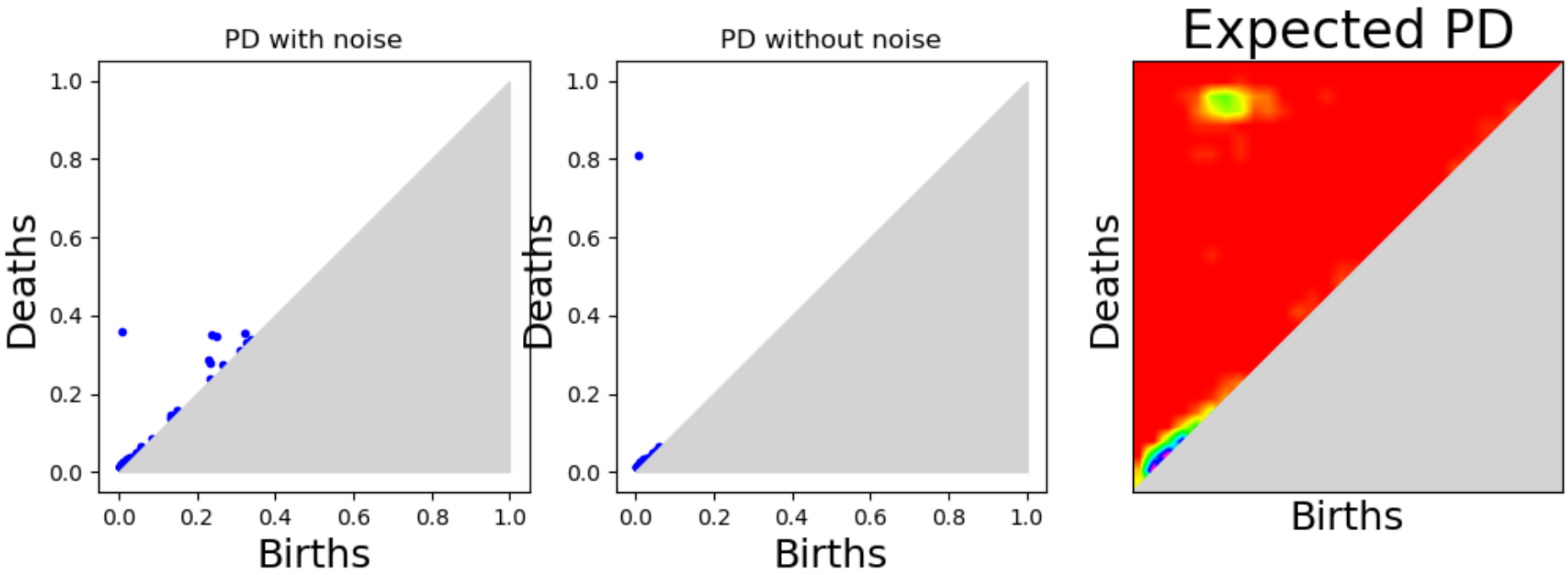
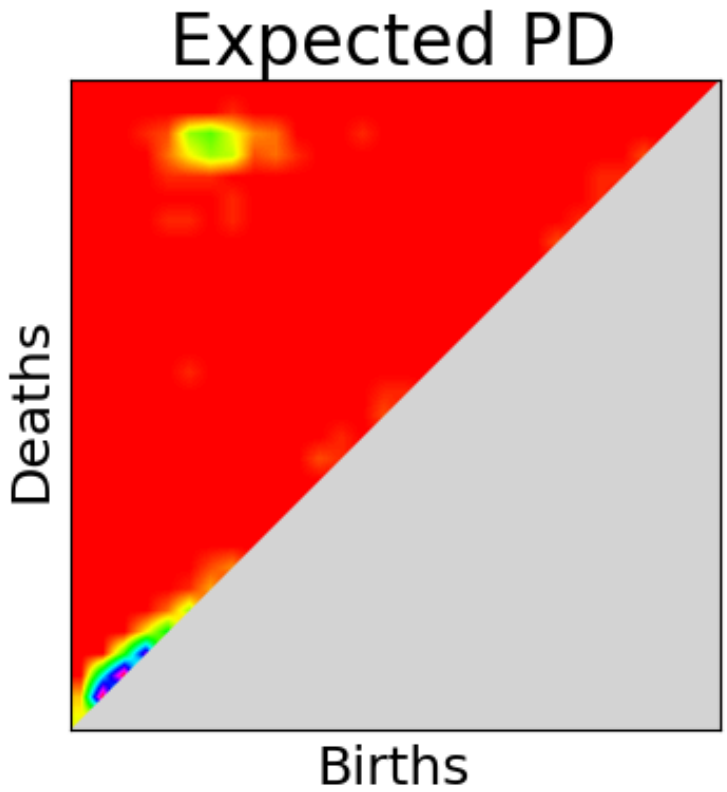


Figure: Persistence Diagram with and without noise and the EPD

As we can see in the figure above, the EPD is quite similar to the true persistence diagram compared to the persistence diagram with random noise.

But the EPD actually has a large support. However, if we wanna do some analysis on the persistence diagrams, usually we would like to have a persistence diagram with a small support. This lead to the idea of the quantization. By doing quantization, we will pick several points from Ω and use it to represent the expected persistence diagram of the dataset. To do so, we need to equip the space of persistence diagrams(persistence measures) with a metric.



The Metric on the Space

Given two measures μ, ν supported on Ω, we can define the distance between μ and ν using the optimal transport metric:

$$\mathrm{OT}_p(\mu,\nu) = \inf_{\pi \in \mathrm{Adm}(\mu,\nu)} \left(\iint_{\bar{\Omega}\times\bar{\Omega}} \|x-y\|^p \mathrm{d}\pi \right)^{\frac{1}{p}}$$

where $\mathrm{Adm}(\mu,\nu)$ is the set of all measures supported on $\bar{\Omega}\times\bar{\Omega}$ whose first marginal is μ and second marginal is ν on Ω (Note that this constrain is on Ω, not $\bar{\Omega}$! This makes it possible for π to transport mass to the boundary, which is also the diagonal $\partial\Omega$ in the graph). When $p=\infty$, the distance OT_∞ is the same as the bottleneck distance.
This gives us the persistence measure space $(\mathcal{M}^p, \mathrm{OT}_p)$.

Formulate Our Goal in a Mathematical Way

Let $\mu \in \mathcal{M}^p$ be a persistence measure and k be a fixed integer. The goal of quantization is to find a measure $v = \sum_{j=1}^{k} m_j \delta_{c_j}$ that approximates the persistence measure μ in an optimal way, i.e. find $((m_1, c_1), \ldots, (m_k, c_k))$ that minimizes $\mathrm{OT}_p(\sum_{j=1}^{k} m_j \delta_{c_j}, \mu)$. Here $c = (c_1, \ldots, c_k) \in \Omega^k$ is called a persistence codebook and $c_j s$ are called centroids.

Codebook and its Partition

So now the problem setting is that given this EPD, we wanna find several points to represent it. This is quite similar to what we're doing in the k-means algorithm. Then given a codebook c, we'll need to define the 'neighborhood' of the centroids.
Let $c = (c_1, \ldots, c_k)$, since we know that the points near the diagonal are not meaningful, thus we define a new 'point' $c_{k+1} = \partial\Omega$, and we can define the partitions:

$$V_j(c) = \{x \in \Omega, \|x - c_j\| \leq \|x - c_{j'}\|, \forall j' < j \text{ and } \|x - c_j\| < \|x - c_{j'}\|, \forall j' > j\}$$

for $1 \leq j \leq k$. Here $V_j(c)$ can be viewed as the cell(neighborhood) of c_j.

For example: the blue points is the codebook and it gives us a partition of Ω.

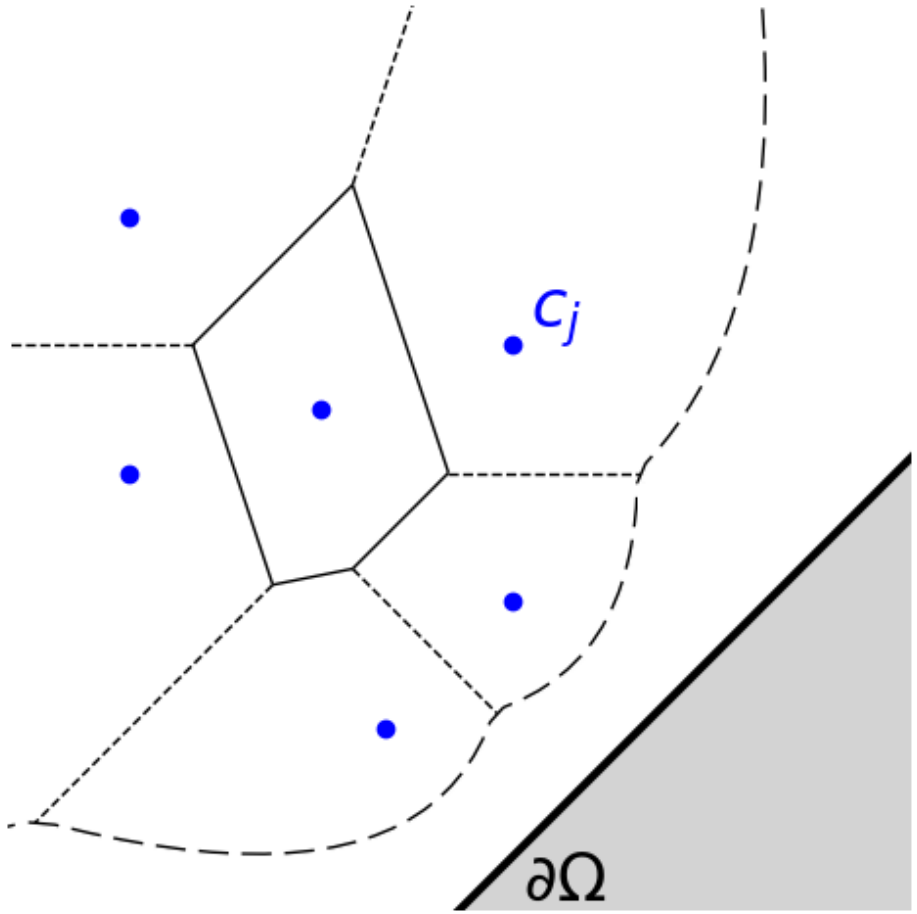


Figure: A codebook and the partition given by this codebook

Given a codebook $c = (c_1, \ldots, c_k)$, we have the following result:

Theorem

Let $\hat{\mu}_c = \sum_{j=1}^{k} \mu(V_j(c))\delta_{c_j}$, then:

$$\mathrm{OT}_p(\hat{\mu}(c), \mu) \leq \mathrm{OT}_p(\nu, \mu)$$

for all $\nu = \sum_{j=1}^{k} m_j \delta_{c_j}$ with $m_1, \ldots, m_k \geq 0$.

This gives us the optimal measure to approximate μ given a codebook c. So the problem of finding the measure ν can be reduced to finding the optimal codebook.

The Quantization Algorithm

In this paper, the author proposed an online algorithm to find the optimal codebook:
Given an initial codebook $c^0 = (c_1^0, \ldots, c_k^0)$, at time t we're given a batch of persistence diagrams: $\mu_i, i \in B_{t+1}$ and the codebook c^t obtain at time t, then we can update the codebook by:

$$c_j^{t+1} = c_j - \frac{c_j - v_p(c, \mu)_j}{t+1}$$

where $\mu = \frac{\sum_{i \in B_{t+1}} \mu_i}{|B_{t+1}|}$, $v_p(c, \mu)_j = \arg\min_y \left(\int_{V_j(c)} \|y - x\|^p \mathrm{d}\mu(x)\right)^{\frac{1}{p}}$ can be considered the p-center mass of μ over the cell $V_j(c)$.
Roughly speaking, the algorithm is pushing c_j toward the p-center mass of μ over $V_j(c)$.

With this algorithm, we can do quantization for the EPD we obtained before and obtain the following result:

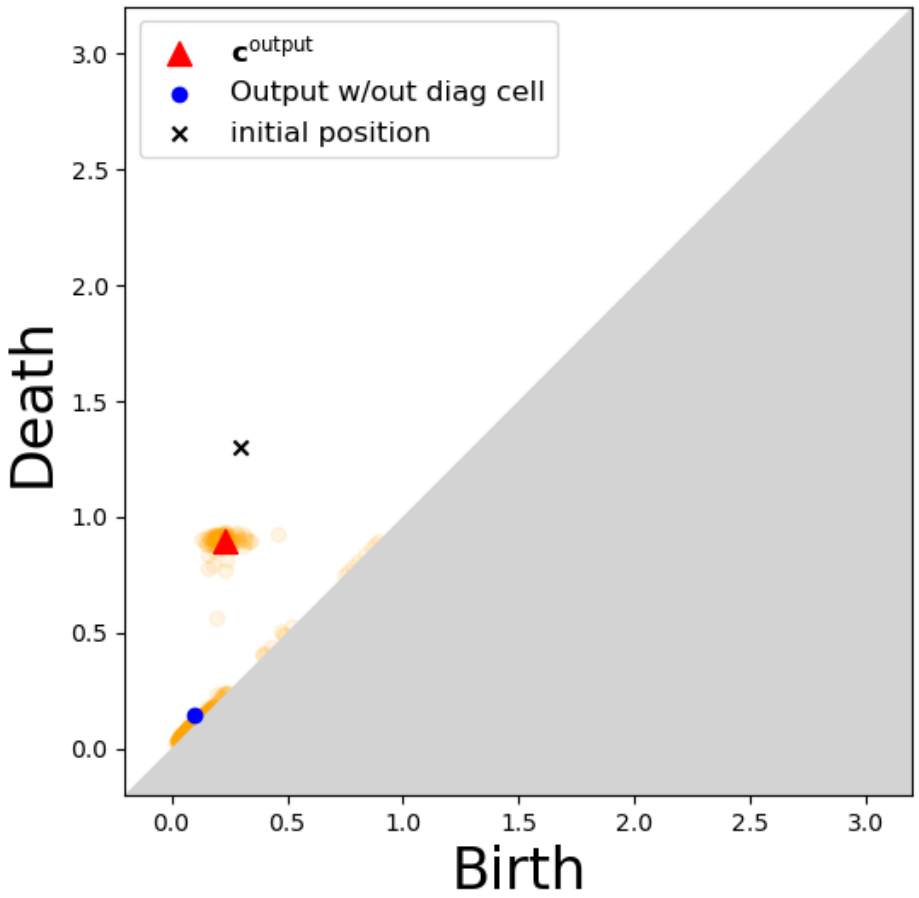


Figure: The result from quantization, the red triangle is the resulting codebook

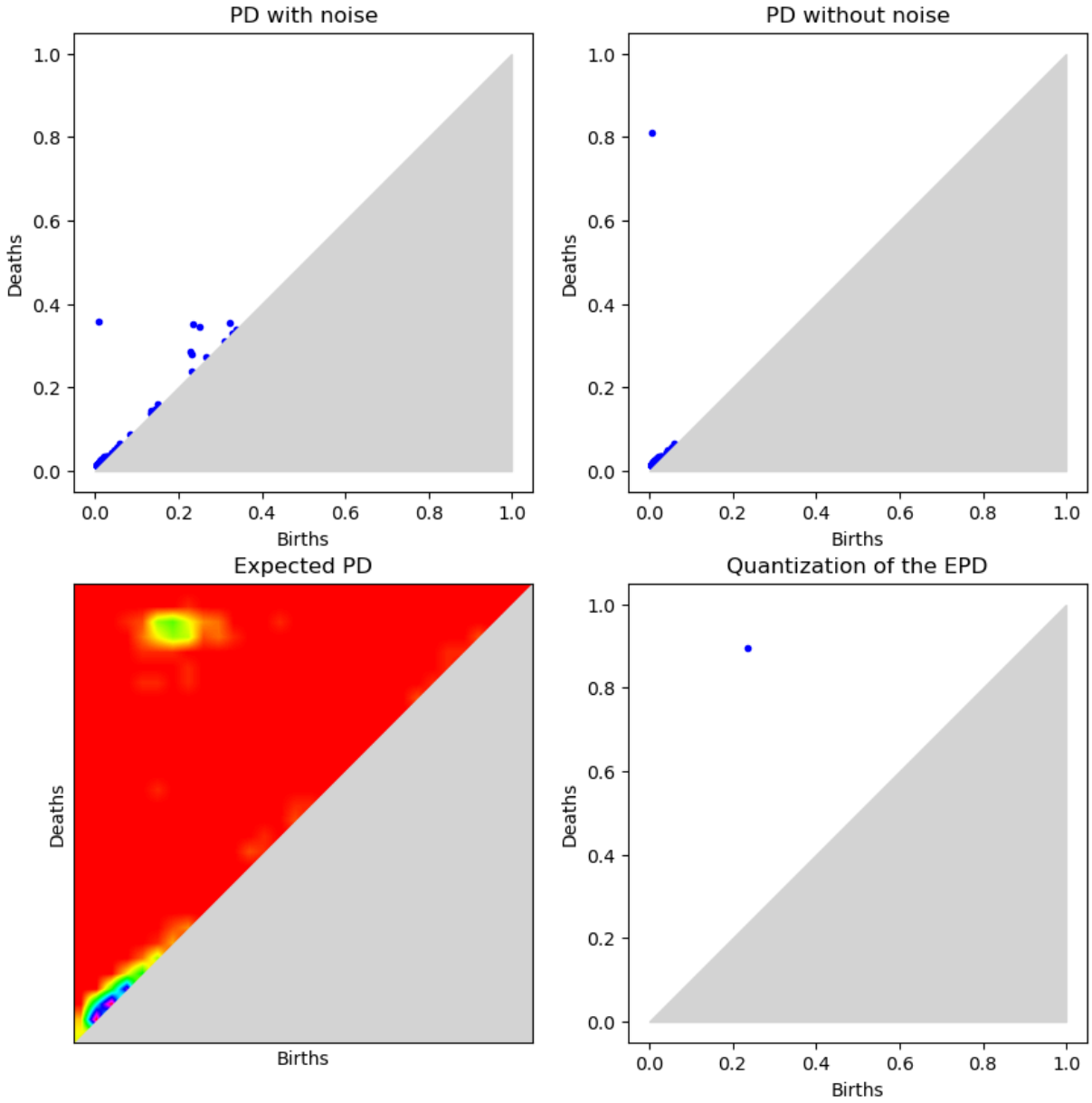


Figure: The PD with noise, without noise, EPD(histogram), Quantization result

Reference

Divol, V. and Lacombe, T., 2021, July. Estimation and quantization of expected persistence diagrams. In International conference on machine learning (pp. 2760-2770). PMLR.

Have a nice day!