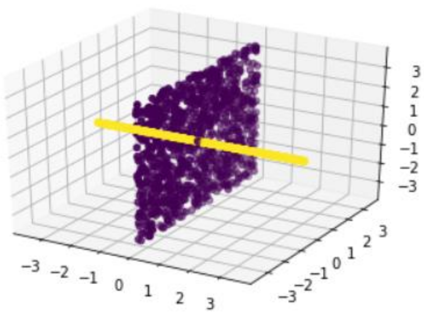
OWL has really good performance in classifying orthogonal objects. In my first model, the line outside the sphere is vertical to the sphere, thus it could be classified completely just through OWL and the speed of convergence is rather quick. Like the following example, the line is vertical to the plane.



However, it could do nearly nothing to the part of line if I only move the line a little bit. OWL tends to distinguish orthogonal elements, like two orthogonal space. And its performance strongly depends on hyperparameters--Δ and λ(in paper). I find this situation really wired. I did experiment on two separate lines. I think the conditions for OWL are really strict and confusing. I tried many combination of Δ and λ, many of them can not even give me reasonable connecting matrix. For other cases, the eigenvalues show that the whole object are distinguished as a single one. And only a single or a few overlapping points.

When comparing two methods, the only difference is the construction of connecting matrix. In OWL, we construct this matrix using coefficients of OWL regression for some points, while in the graph Laplacian we connect points with their nearest neighbors. The knn connecting matrix needs very few assumptions. And the basic assumption for this method is that the points near each other are more likely to be in the same clustering, which is also a basic assumption for clustering. Thus in this case, if there exists a lower dimensional object, it is easier to connect points in this object with other points of this object. So it is not hard to find that follow the process of spectral clustering, no matter how many clustering we would detect(many of these are part of same clustering), we will find a part of this object in one clustering. This is the advantage of knn graph Laplacian, and the next problem is to extend this detected set to observe the whole object.

For OWL, it is a different way to connect points. We do regression as below:

$$min_x \frac{1}{2}||y - Hx||^2 + \Omega_w(x)$$

The coefficient represents the weight between points. And I do some experiment on this interesting model:

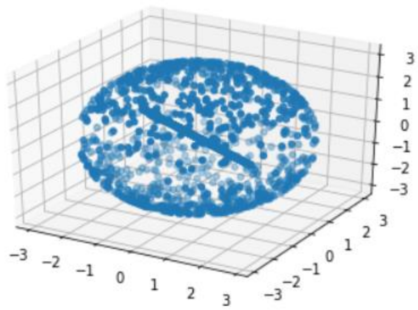


Figure 2: *A sphere and a separate line inside. No interaction*

When I calculate coefficients for the points in line, I found that all other points were given a value which means all other points were connected with this point, but with different weights. The weights are like below:

```
[0. 0,
 1.3985540027553222e-06,
 1.3985540027553222e-06,
 1.3985540027553222e-06,
 1.8690057057132378e-06,
 2.7651591524647097e-06,
 2.7651591524647097e-06,
 2.942212026808166e-06,
 3.948612242110936e-06,
 3.998089959768542e-06,
 4.028888015607741e-06,
 4.13344992566016e-06,
 7.1744294691040524e-06,
 7.1744294691040524e-06,
 7.1744294691040524e-06,
 7.1744294691040524e-06,
 8.473518979442427e-06,
 8.473518979442427e-06,
```

These values seems to be small because I sort the value in ascending order. But the largest value is only 0.002256934821999308 which is still a small value. And other points have similar results. This result means we could not distinguish clustering through this matrix. When I check the second smallest eigenvector, it classified the whole object to only one clustering, just as I thought. However, the knn Laplacian perfectly separated two objects.

To find the reason, I check the paper again. There are some interesting results. As I mentioned before, OWL tends to distinguish orthogonal subspaces, thus it works better if the level of orthogonal is higher(this is a very ambiguous description). A concept named *affinity* is used to describe whether two subspaces are orthogonal or not.

The performance of OWL depends on affinity and weights in OWL regularizer. A theorem in the paper describe that.

If the inequality is satisfied, the performance would be great.

$$max_{k,l} aff(S_k, S_l) \leq \kappa_0 \frac{\overline{w}_{N_l+1}}{w_1} \frac{\sqrt{logN_l/d_l}}{logN}$$

In this inequality, left side describe the level of orthogonal which is decided by data. The w part on the right depends on weights. However, it is not always easy to satisfy this condition. In the model in figure 2, the affinity is almost 1, this means in order to satisfy this inequality, we need to set all w the same value. Even in that case, the inequality may not be right. That is why OWL performed so bad in this model.

Since affinity is a property of dataset and we usually have no knowledge of subspaces in this data set, it is hard to check this property. Comparing with knn Laplacian, OWL is only suitable to a smaller range of dataset.