Report for Exercise 5 from Group K

Tasks addressed:	5
Authors:	Yun Fei Hsu (03732414) Jan Watter (03665485) Yaling Shen (03734727)
Last compiled:	2021–01–29
Source code:	`https://github.com/ElsieSHEN/Crowd_Model_WS5`

The work on tasks was divided in the following way:
Equally for all group members and tasks.

Yun Fei Hsu (03732414)	Task 1	33.3%
	Task 2	33.3%
	Task 3	33.3%
	Task 4	33.3%
	Task 5	33.3%
Jan Watter (03665485)	Task 1	33.3%
	Task 2	33.3%
	Task 3	33.3%
	Task 4	33.3%
	Task 5	33.3%
Yaling Shen (03734727)	Task 1	33.3%
	Task 2	33.3%
	Task 3	33.3%
	Task 4	33.3%
	Task 5	33.3%

Report on task 1, Approximating functions

In this task, we are supposed to implement function approximations with linear functions for a linear dataset and radial functions for a nonlinear dataset. The whole task is divided into three parts.

Part One: Approximate the function in dataset (A) with a linear function.

To approximate linear functions, we need to find a linear function between two Euclidean spaces $\mathbb{R}^n$, $\mathbb{R}^d$ with $n, d \in \mathbb{N}$ where $f_{\text{linear}} : \mathbb{R}^n \to \mathbb{R}^d$ is a map, such that for $x \in \mathbb{R}^n$,

$$f_{\text{linear}}(x) = Ax \in \mathbb{R}^d \tag{1}$$

for some matrix $A \in \mathbb{R}^{d \times n}$.

We first load the linear dataset (A) the from `linear_function.txt` as x. Then we use least-squares minimization to solve the problem. The minimal mean squares error problem is defined by the following equation:

$$\min_{\hat{f}} e(\hat{f}) = \min_{\hat{f}} ||F - \hat{f}(X)||^2 = \min_{A} ||F - XA^T||^2 \tag{2}$$

The closed form solution of minimizing the least-squares error is:

$$\hat{A}^T = (X^T X)^{-1} X^T F \tag{3}$$

As for the implementation part, we write the linear approximation and radial approximation (later part) with similar structures as two classes in one `utils.py` file. The function calling is similar to the methods imported from sklearn. As for the approximation, we use `numpy.linalg.lstsq`. The approximation result is shown in figure 1. The original data is plotted with blue dots and the linear approximation function is plotted as an orange line.

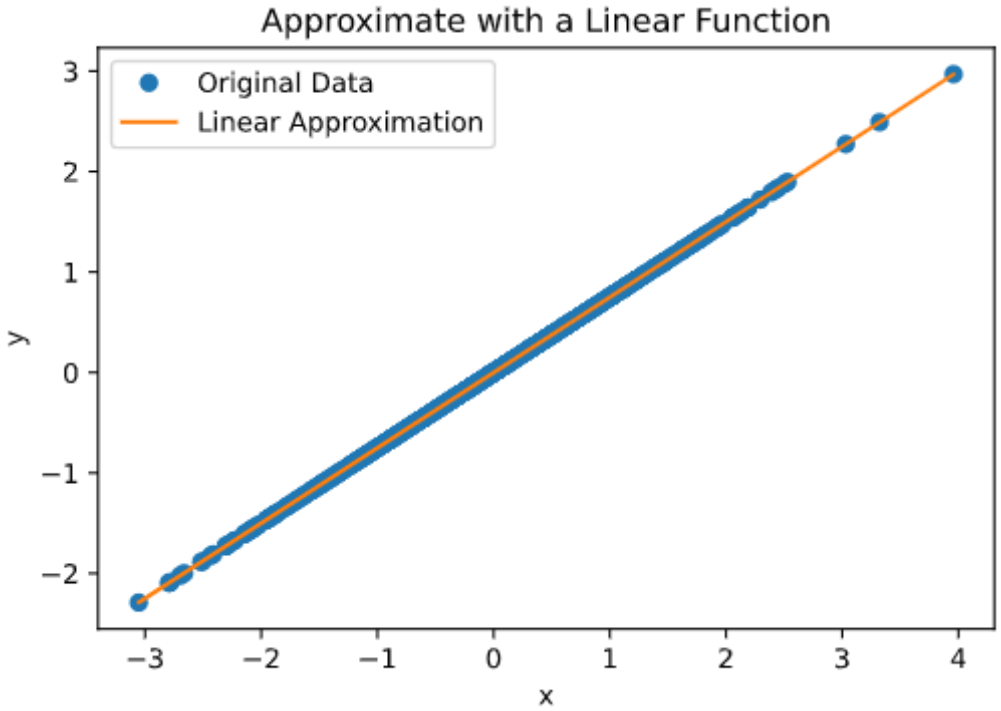

Figure 1: Approximate A Linear Dataset (A) with A Linear Function

Part Two: Approximate the function in dataset (B) with a linear function.

In this part, we are asked to approximate a nonlinear dataset (B) with a linear function. The only difference between part one and two is the loaded dataset. This time, we load our x from `nonlinear_function.txt` from Moodle. The approximation result is shown in Figure 2. Still, the original data is plotted with blue dots. We can see that the approximation result is extremely bad.

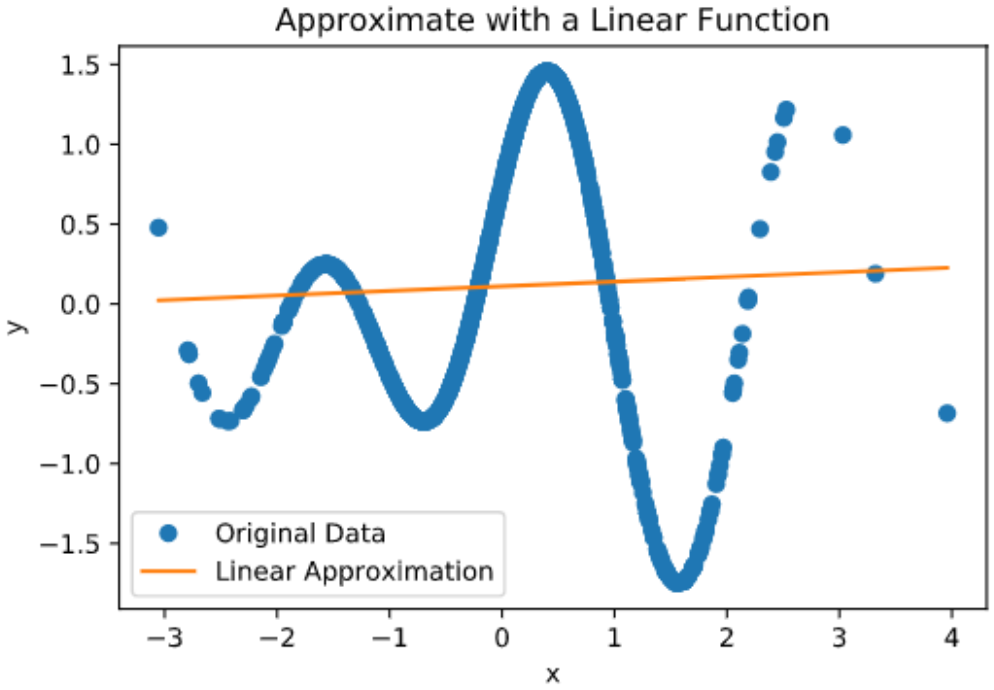

Figure 2: Approximate A Nonlinear Dataset (B) with A Linear Function

Part Three: Approximate the function in dataset (B) with a combination of radial functions.

Apparently, linear functions cannot approximate the nonlinear dataset well. To approximate the nonlinear dataset, we use radial functions instead of linear ones. Here, we use a linear decomposition into nonlinear basis functions. The basic idea is to write the unknown function f as a combination of known functions ϕ, such that

$$f(x) = \sum_{l=1}^{L} c_l \phi_l(x), c_l \in \mathbb{R}^d \tag{4}$$

where the so-called *radial basis functions* ϕ are defined by:

$$\phi_l(x) = \exp(-\|x_l - x\|^2/\epsilon^2) \tag{5}$$

where x_l is the center of the basis function, and the parameter ϵ is the bandwidth.

The minimal least square error problem becomes:

$$\min_{\hat{f}} e(\hat{f}) = \min_{\hat{f}} ||F - \hat{f}(X)||^2 = \min_{C} ||F - \phi(X)C^T||^2 \tag{6}$$

where $C \in \mathbb{R}^{d \times L}$ contains the list of coefficients c_l, and

$$\phi(X) := (\phi_1(X), \phi_2(X), \ldots, \phi_L(X)) \tag{7}$$

We set $L = 15$. In the Diffusion Maps excercise from the last worksheet, the ϵ is calculated by 0.05 times the largest element in distance matrix. In this part, we implement it similar. ϵ is the largest distance in original data multiplied by 0.05. The approximation result is shown in Figure 3. Blue dots are the original nonlinear data and the orange ones are the approximation.

When approximating a linear dataset (A) with a combination of radial functions ($L = 15$), the approximation would be like in Figure 4. Clearly, is is not a good idea to use radial basis functions for dataset (A). Compared with a linear approximator, radial approximation not only cost much time, it cannot approximate well without carefully choosing L and ϵ.

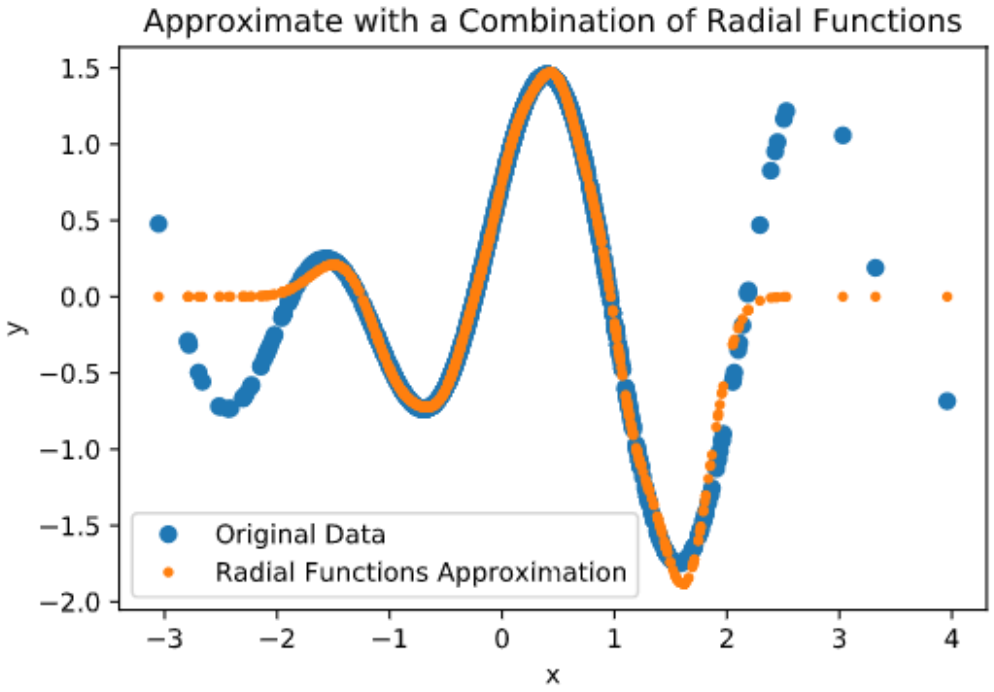

Figure 3: Approximate A Nonlinear Dataset (B) with Radial Functions

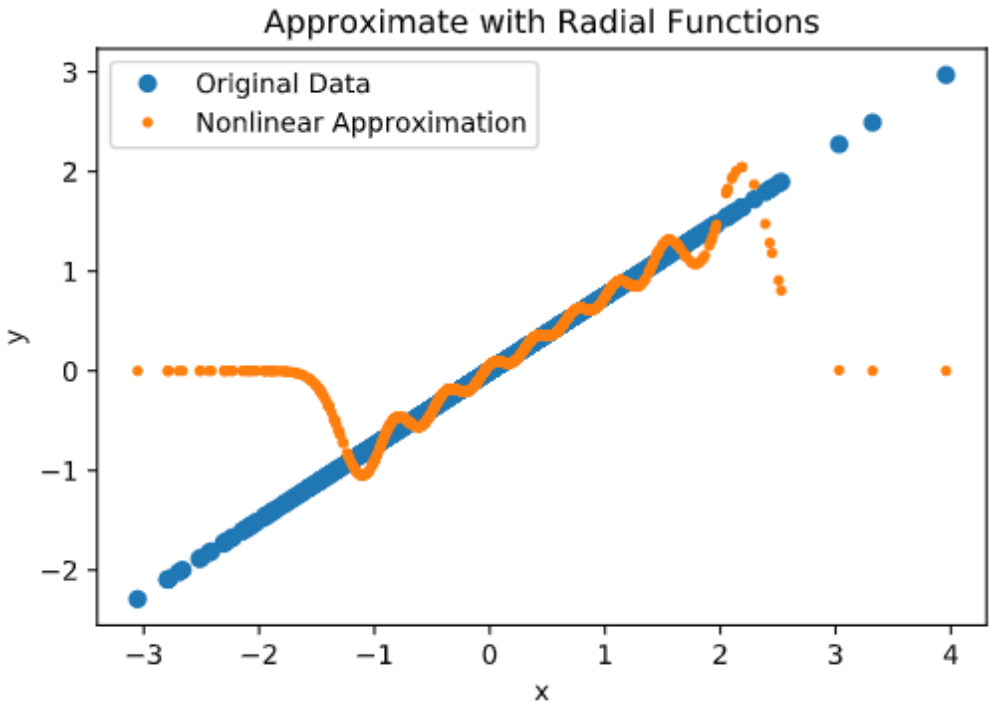

Figure 4: Approximate A Linear Dataset (A) with Radial Functions

Report on task 2, Approximating linear vector fields

Part One: In this part, we are asked to estimate the vector field that was used to generate the points x_1 from the points x_0. We first load data from `linear_vectorfield_data_x0.txt` as our 1000 data points x_0. Then we use the finite-difference formula to estimate the vectors $v^{(k)}$ at all points $x_0^{(k)}$:

$$\hat{v}^{(k)} = \frac{x_1^{(k)} - x_0^{(k)}}{\Delta t} \tag{8}$$

Then we approximate the matrix $A \in \mathbb{R}^{2\times 2}$ with:

$$v(x_0^{(k)}) = v^{(k)} = Ax_0^{(k)} \tag{9}$$

The vector field is linear, so we use linear approximator defined in task1 to get the predicted $\hat{v}^{(k)}$.

Part Two: We have already estimated matrix $\hat{A} \approx A$, and have predicted the $\hat{v}^{(k)}$ for each x_0. We then calculate $\hat{x_1}^{(k)}$ with $\Delta t = 0.1$ by:

$$\hat{x_1}^{(k)} = \hat{v}^{(k)}\Delta t + x_0^{(k)} \tag{10}$$

We then compute the mean square error between $x1^{(k)}$ and $\hat{x1}^{(k)}$ with `np.square.mean`. The final MSE is 5.2660926699020454e-17.

Part Three: Choose the initial point $x_0 = (10, 10)$, again to solve the linear system with matrix approximation for $T = 100$ seconds. With $\Delta t = 0.1$, the total number of iterations is 1000. The trajectory and the phase portrait in domain $[-10, 10]^2$ are shown in Figure 5 and 6, respectively.

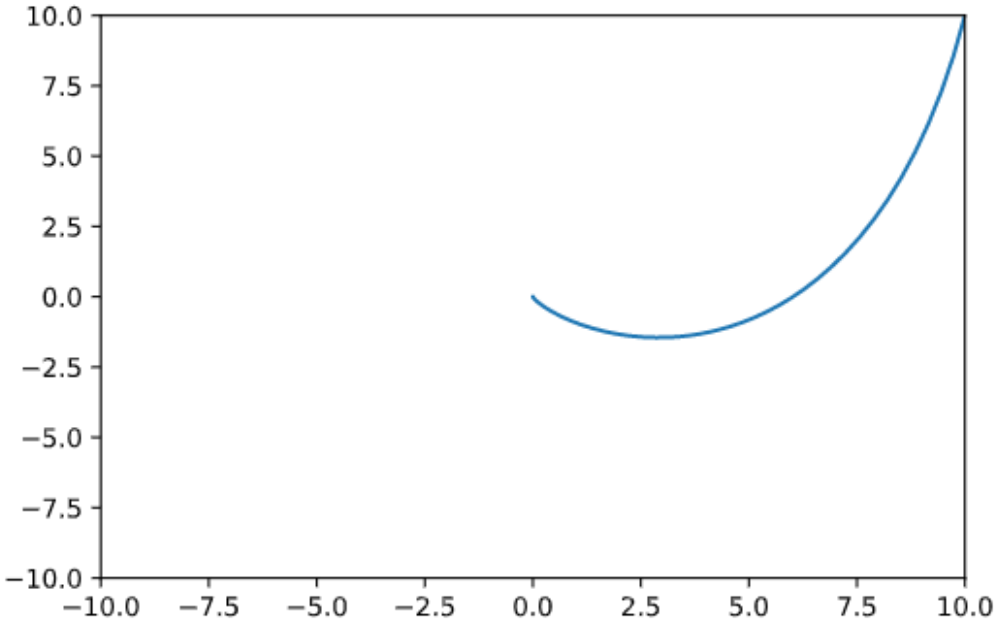

Figure 5: Trajectory of the Motion

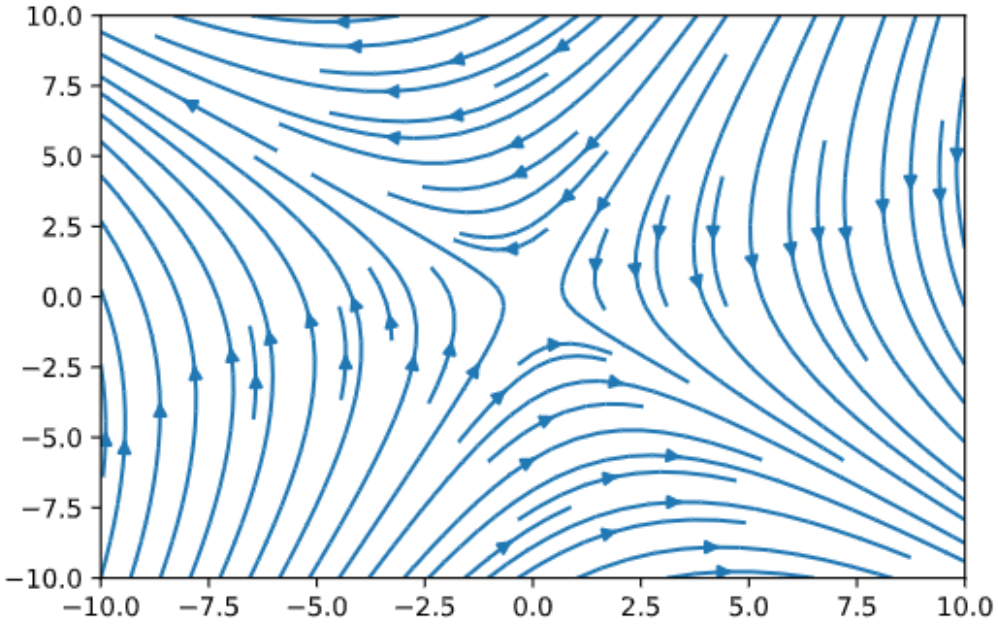

Figure 6: Phase Portrait

Report on task 3, Approximating nonlinear vector fields

Part One: In this part, we are asked to approximate the vector field using linear functions. We fist reuse the linear_approx() function to find the linear operator A and obtain the estimate vector field $\hat{v}^{(k)}$. After that, we calculate the approximate end point $\hat{v}^{(k)}$ with the estimated vector field and a $\Delta t = 0.1$. The mean square error between $x_1^{(k)}$ and $\hat{x_1}^{(k)}$ is 0.018635040587480996.

Part Two: For using the radial basis functions to approximate the vector field, we manually check the MSE of the parameters change and choose the setting with the minimum MSE. We set the number of centers to $L = 800$ and the $\epsilon = 4$, the mean square error between $x_1^{(k)}$ and $\hat{x_1}^{(k)}$ is 1.390180489773158e-15. Compare to the previous result, the error of non-linear approximation is way smaller than the error of the linear approximation, which means using the non-linear function can better to predict the underlying dynamics of this process. Therefore, we can conclude that the vector field of the process is non-linear.

Part Three: To be more precise: in task 2 we plotted the trajectory of one initial point $x_0 = (10, 10)$ whereas now we plot the trajectories for all initial points to find out how the positions of the points will evolve over time. For the first plot we chose a $T = 1$ with $\Delta t = 0.01$ as shown in Figure 7. Each line represents a trajectory although the direction is not visible. Furthermore this plot is confusing since the trajectories cross each other but it also encapsulates some of the interesting behaviour of the vector field therefore we decided to include it.

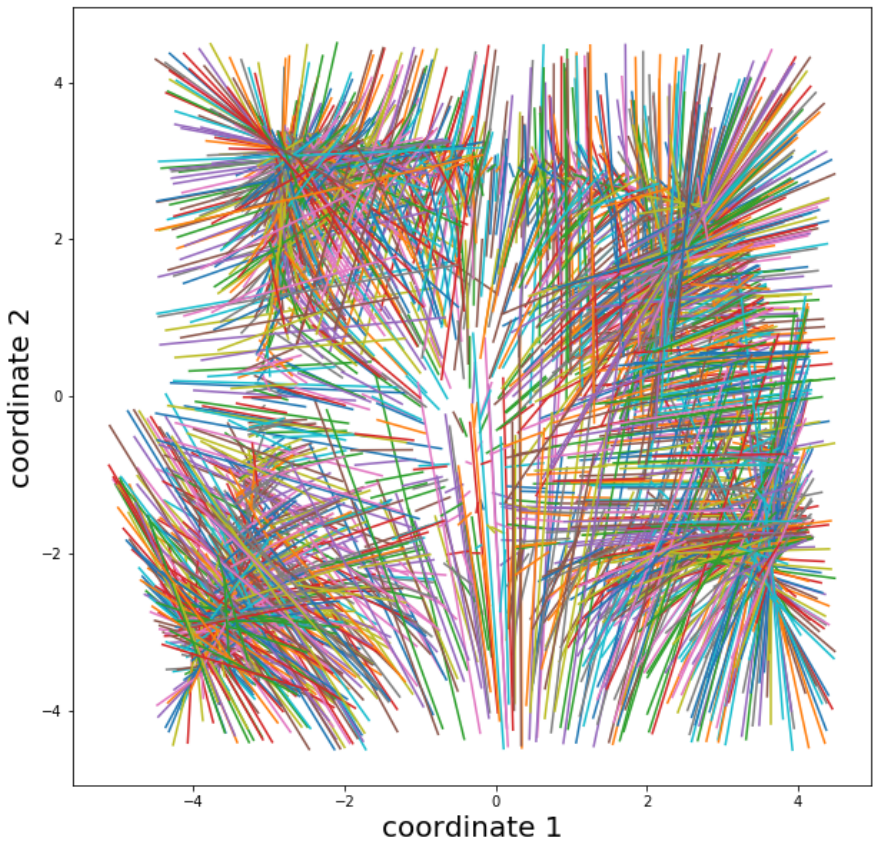

Figure 7: Trajectories with $T = 1$, $\Delta t = 0.01$

A streamplot like in task 2 would be more clear but was not possible due to the dimensionality of the radial basis approximation and the restrict of streamplot to work on evenly spaced points only. However one can clearly see the nonlinear behaviour of the vector field in this plot.

In Figure 8 we plot the same setting but with $T = 100$ and $\Delta t = 0.1$. When T is further increased the trajectories continue to "spread out". Therefore one can deduct that the trajectories "grow" as far as we see, without bounds. This would therefore imply that there are no steady states.

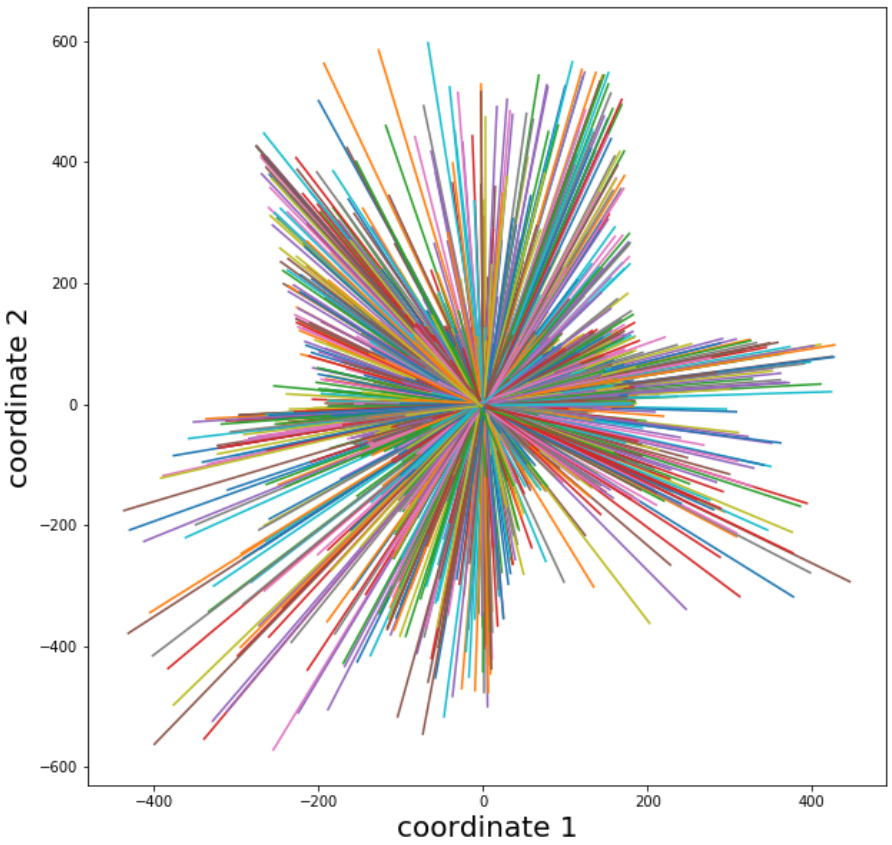

Figure 8: Trajectories with $T = 100$, $\Delta t = 0.1$

In Figure 9 we plot the evolution of the linear vector field as a comparison. One can see that the trajectories tend to grow without bounds as well but do not show such nonlinear behaviour at the start of the evolution.

The question about topological equivalence is certainly not easy. At first, it seems rather difficult to construct a continuous map between the systems. Furthermore we do not even have an analytic description of the system but our approximated estimations only. That being said, the long term behaviour seems to be similar, in the sense that it grows to infinity (no boundaries) and there are no "sinks" in the system. Maybe there is a notion of a "long-term topological equivalence" or there even might be a function that "untangles" the nonlinear behaviour at the start to a linear system. This however seems rather unlikely.

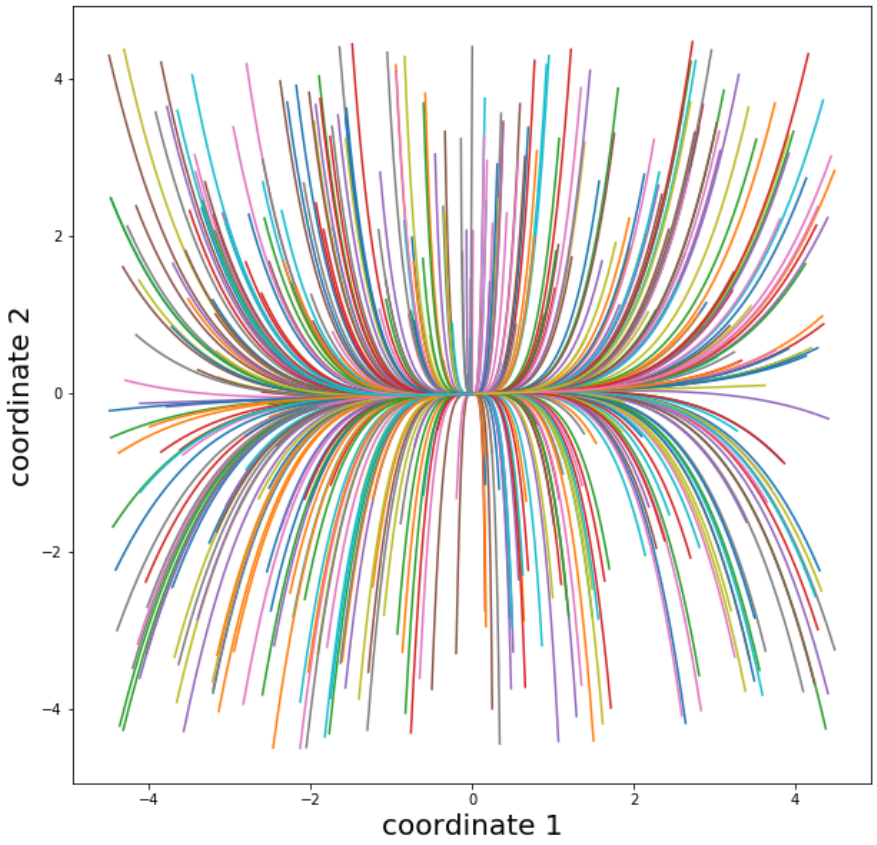

Figure 9: Trajectories with $T = 100$, $\Delta t = 0.1$

Report on task 4, Time-delay embedding

Part One: We are given a dataset with two columns corresponding to two coordinates of a closed, one-dimensional manifold. Figure 10 shows the first coordinate plotted against the line number (the "time"). One can clearly see how the dataset is periodic.

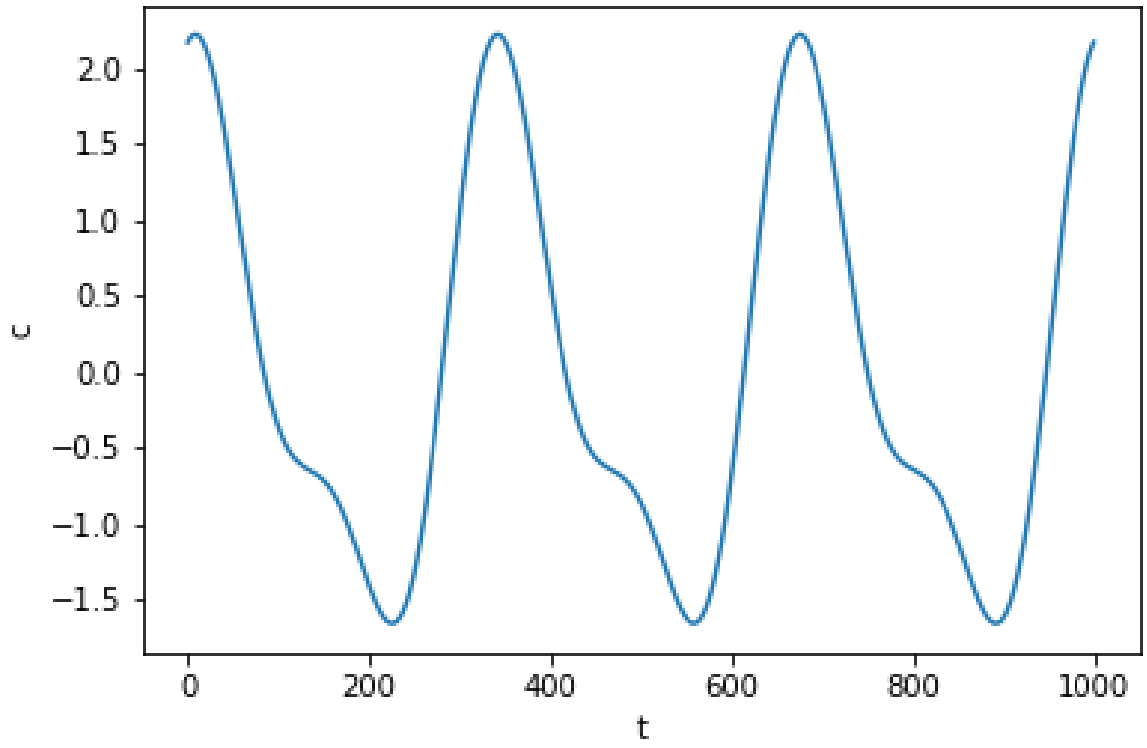

Figure 10: First coordinate plotted against the "time"

Next, we are asked to choose a delay of Δn rows and plot the coordinate against its delayed version. Figures 11 till 14 show these plots with 4 different delays.
According to Takens theorem a single measured quantity is enough to embed the manifold correctly: a time-delayed version of one generic signal can embed the n-dimensional manifold. But one needs $2 * d + 1$ coordinates of this signal to be sure that the manifold is embedded correctly. Since the given manifold is one-dimensional ($d = 1$), we need $2 * d + 1 = 3$ coordinates to embed the periodic manifold correctly.

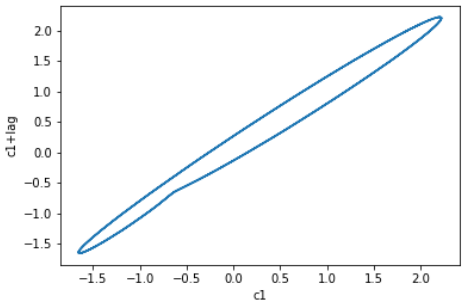

Figure 11: $\Delta n = 5$

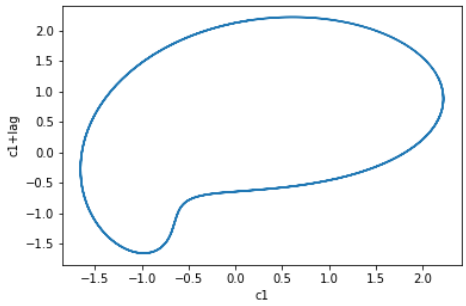

Figure 12: $\Delta n = 50$

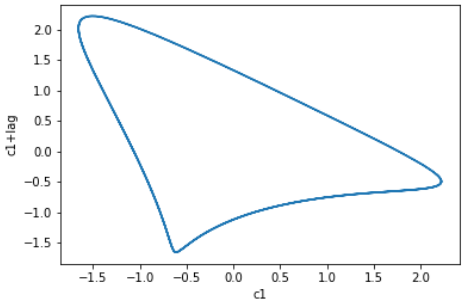

Figure 13: $\Delta n = 100$

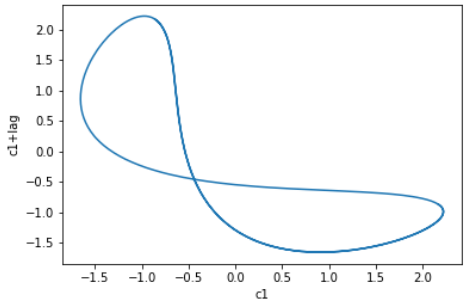

Figure 14: $\Delta n = 500$

Part Two: In part two we are asked to test Takens theorem for the Lorenz attractor. Initially we plot the Lorenz system in its chaotic regime as was done in worksheet 3 shown in Figure 15. We continue to visualize $x_1 = x(t)$ against $x_2 = x(t + \Delta t)$ and $x_3 = x(t + 2 * \Delta t)$ in a three-dimensional plot as shown in Figure 16. We tried different values for Δt and decided to pick $\Delta t = 2$ for this visualization. We performed the same calculations for the y-coordinate as shown in Figure 17. The embedding is correct.
Figure 18 shows the same lagged plot but for the z-coordinate. This time the embedding fails. We assume that the basic requirements, a smooth evolution operator and a smooth operation function are given just as in the first two coordinates. Therefore the problem could arise due to the restriction of genericity in the theorem. Meaning Takens theorem applies to almost all diffeomorphisms but not for "constructed" examples as explained in the video of the lecturer. Constructed in the sense that humans have built an explicit formula for the system and probably built in some symmetries.

Sidenote: This discussion reminds me of this lecture `https://www.youtube.com/watch?v=99hVAu1k6G8` where the speaker argues that theoretical physics is in a crises since theoretical physicists build their theories like String-theories on a lot of symmetries but nature probably does not have these symmetries. By the way, none of us studied physics so we are not sure if this could be linked to our topic.

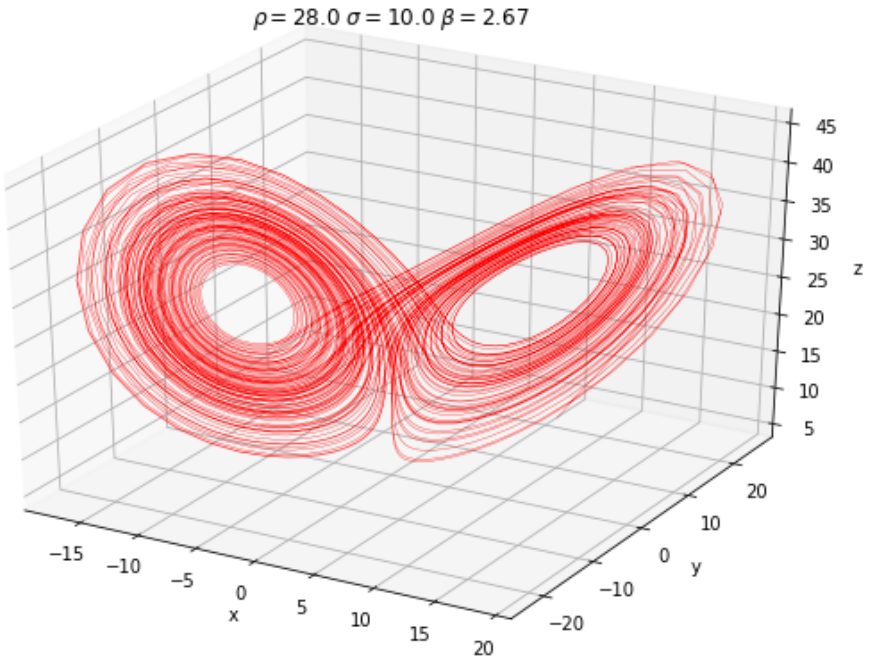

Figure 15: Lorenz system as done in worksheet 3

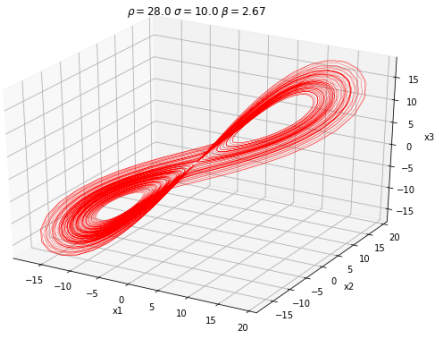

Figure 16: Lagged x-coordinate with $\Delta t = 2$

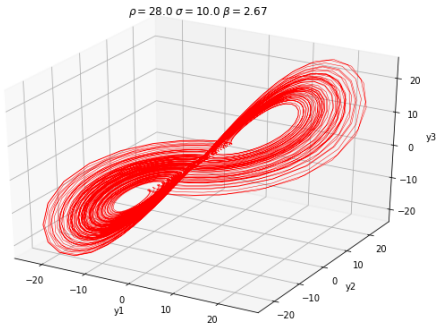

Figure 17: Lagged y-coordinate with $\Delta t = 2$

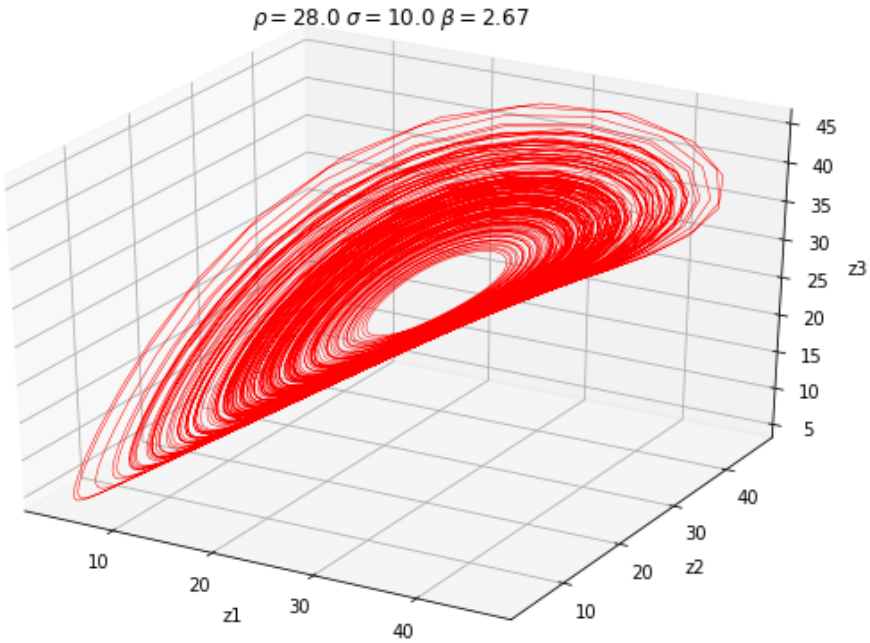

Figure 18: Lagged z-coordinate with $\Delta t = 2$

Report on task 5, Learning crowd dynamics

In this task, we have to learn and predict the utilization of the MI building.

Part One: For the periodic system with a one-dimensional manifold ($d = 1$), we need $2 * d + 1 = 3$ coordinates to embed the periodic manifold according to Takens theorem. For the implementation, we first create the windows dataset for the first 3 measurement areas using a rolling window function which returns 350 delays in each row for a total 13651 rows. Then we normalize the dataset and apply the PCA with 3 components as the embedding space require 3-dimension. The result is shown in Figure 19.

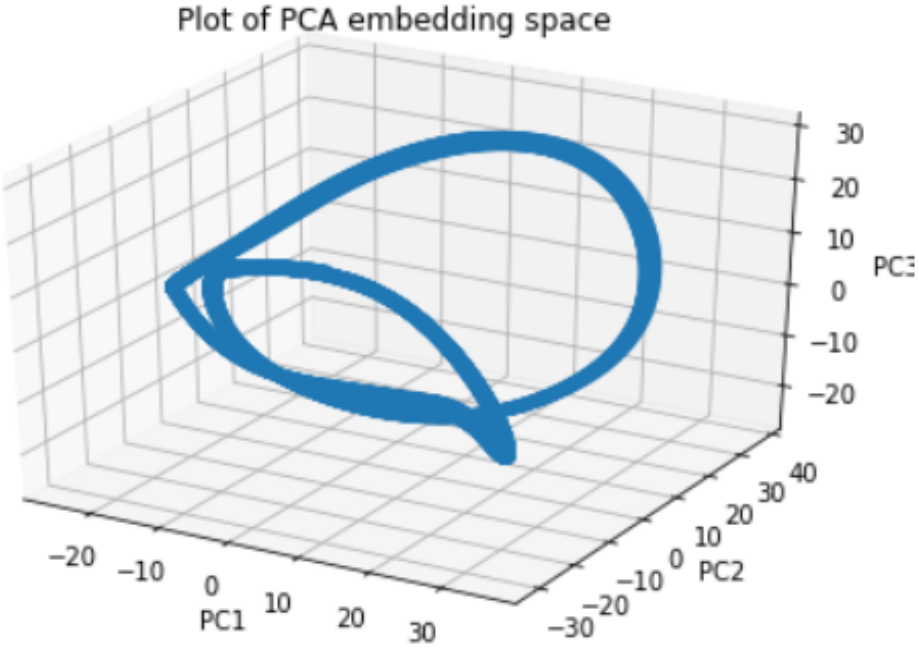

Figure 19: PCA embedding space

Part Two: In this part, we colored the previous embedding space using the utilization of each measurement area. The first measurement area is identical to column 2, then we plot the following graphs from left to right, top to bottom. The brighter color shows higher utilization of the corresponding area. The result is shown in Figure 20.

Part Three: In part 3, we first manually check the loop of the PCA embedding space to prevent over-computation but we use the whole dataset as well to compare the single loop result. We set $\Delta t = 1$ to represent the time step and the ${x_1}^{(k)}$ to the next PCA projected value. Then the vectors $\hat{v}^{(k)}$ would be the arc length of the curve in the PCA space. We use the nonlinear approach to approximate the vector fields on the arc length since the curve in the PCA space is nonlinear. By manually checking the MSE value, we set the number of centers $L = 400$ and $\epsilon = 12$ to obtain the approximate vector field and a relatively small MSE value: 3.006264432303226e-06.

Part Four: After looping over a 14 day time step, the utilization of the MI building in the PCA space is shown in Figure 21.Compare to the results with the given data shown in Figure 20, the predicted value does not have a similar shape with previous plots.

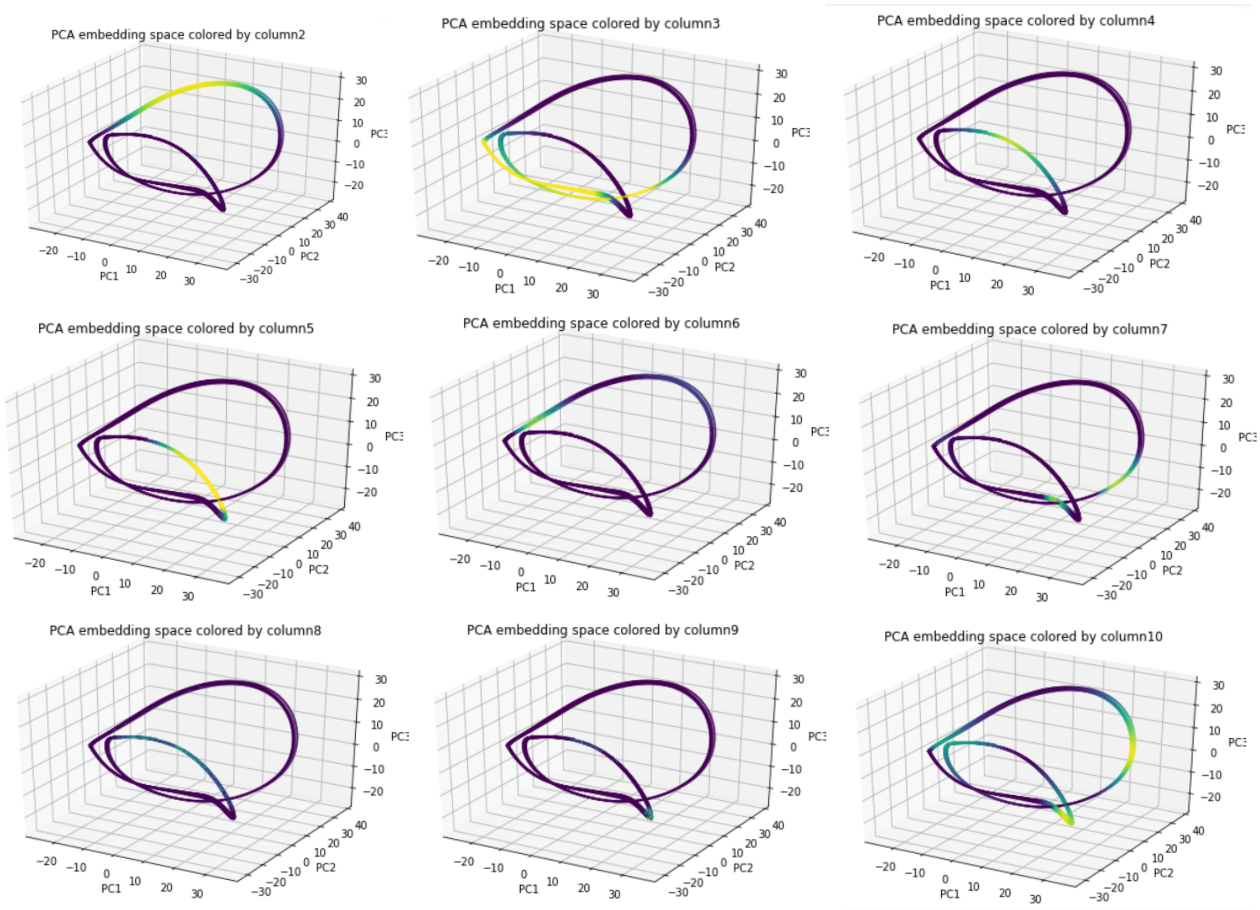

Figure 20: PCA embedding space colored by measurement areas

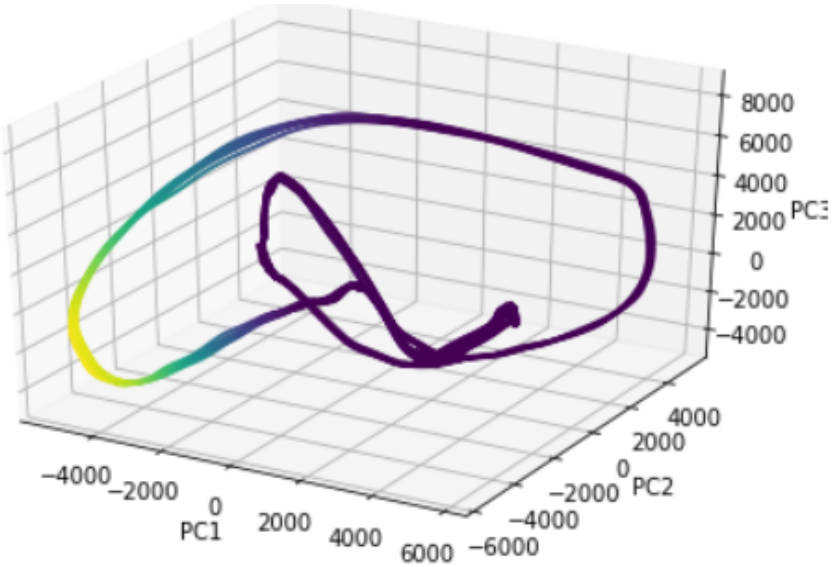

Figure 21: Predicted utilization of MI building after 14 days