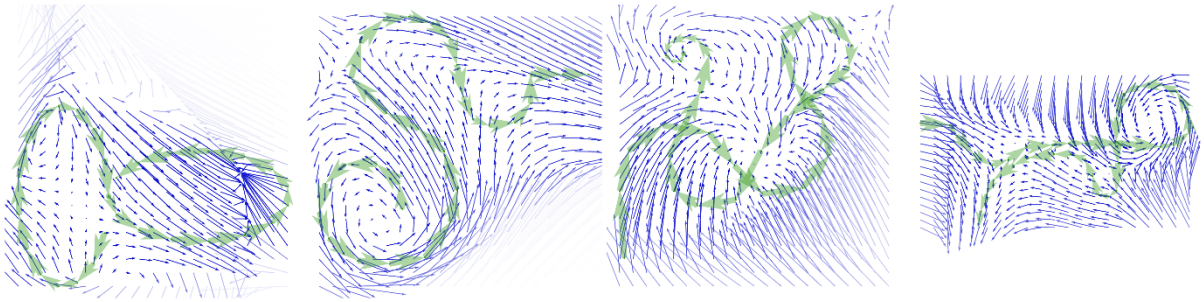
Assignment 2 – Sahil Adwani

1.3)

1) set random seed at 42 for consistency for all

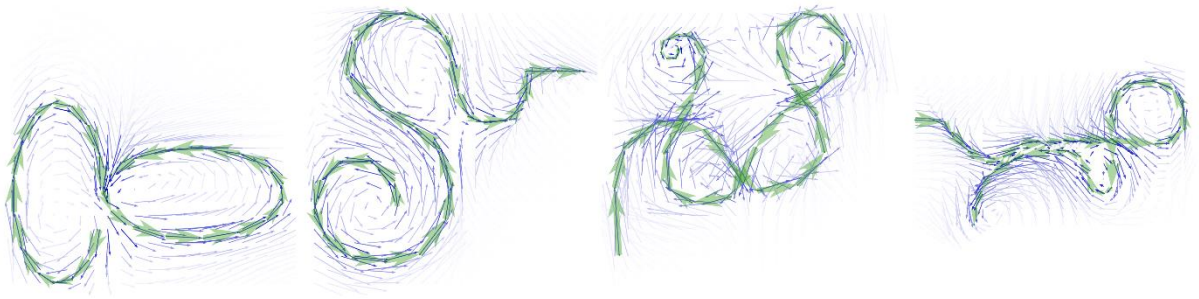
Nhid = 10



The model struggles to effectively capture the complex, non-linear patterns and relationships in the data. The predicted trajectories are less smooth and failed to accurately follow the true paths. This is because the small number of hidden units limits the model's capacity to represent complex, non-linear relationships leading to underfitting.

The predicted trajectories show high uncertainty especially around curves and loops proven by density of blue arrows and colour of the arrows. There are a high number of dark blue arrows everywhere which represent uncertainty and high variability in the model's performance. The model struggles to make accurate prediction shown by the high number of arrows around the green arrow's trajectory.

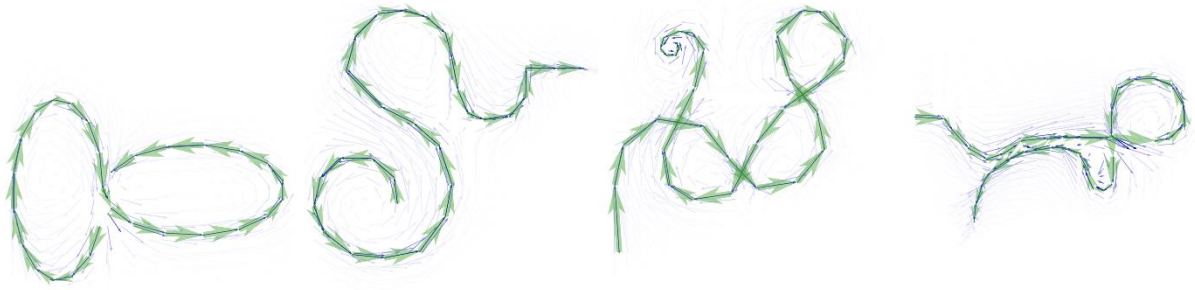
Nhid = 100



Increasing Nhid to 100 improved the predictions. The predictions trajectories are smoother meaning it's a better fit to the true path compared to Nhid of 10. The number of blue arrows has decreased, and the arrows are more aligned with the actual green trajectory which means the model's predictions have become more reliable and decreases uncertainty. The blue predicted arrows along the green trajectory are slightly darker then the ones further from the green trajectory are fainter. This shows decreases in uncertainty and variability in its predictions compared to Nhid = 10. Although we can see faint arrows on the outside so there is still some

uncertainty. The increase to 100 hidden units enhances the model's capacity to capture and represent complex, non-linear relationships in the data.

Nhid = 500



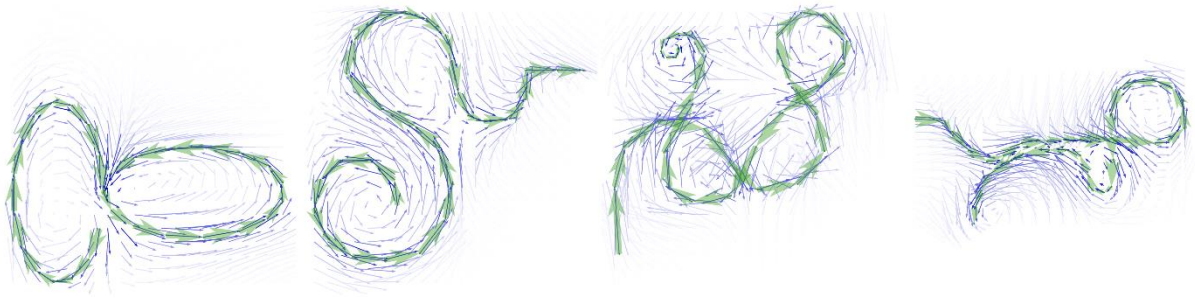
With Nhid = 500, the models' predicted trajectories are even smoother and more accurate, closely following the true path. We can see strong dark predicted arrows along the true green path and has done this better then Nhid = 100. This indicated increase in confidence and accuracy.

Although there are still faint arrows further then the green trajectory, its much fainter and hence better then Nhid =100. This shows that uncertainty is minimal, and the model's confidence has improved.

The number of arrows we can see has also decreased compared to Nhid = 10 which reflects a more accurate model with reduced variability in predictions.

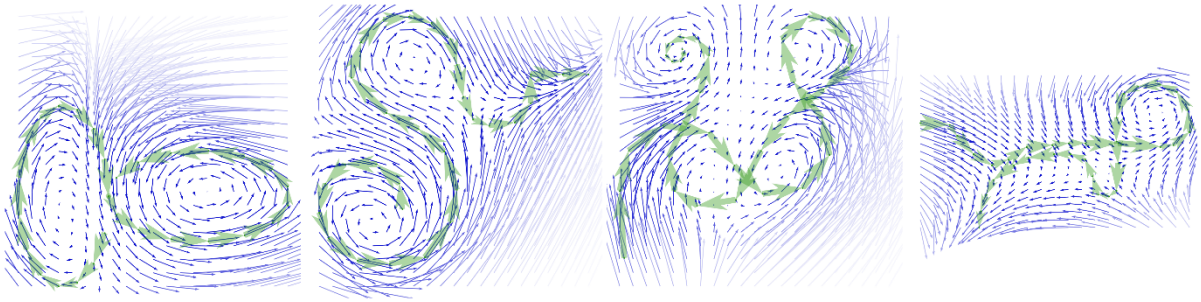
Increasing Nhid to 500 provides the model with a significantly higher capacity to capture and represent the complex, non-linear patterns in the data.

2) using Nhid = 100 for all
ReLU (z = z * (z>0))



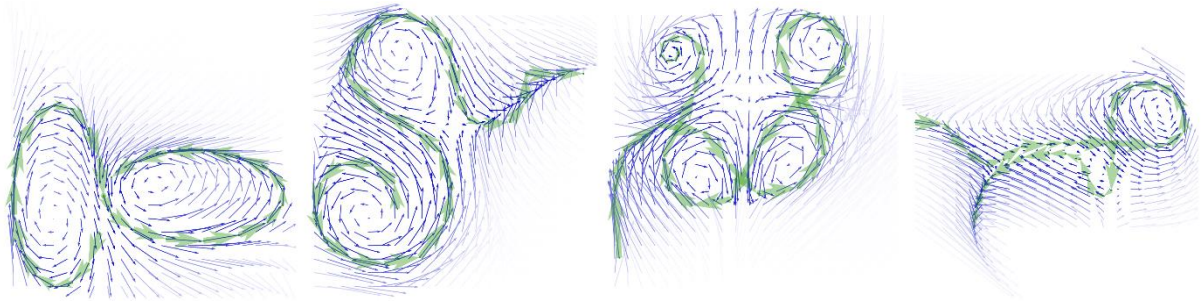
ReLU (Rectified Linear Unit) has shown the best performance in capturing non-linear and complex patterns in the data. The predicted trajectories are smoother, with fewer uncertainties and higher accuracy. The predicted arrows are consistently darker and more aligned with the true trajectory (green path), indicating that the model is confident and accurately following the actual paths. This suggests that ReLU is effective in handling non-linearity and complexity in this context.
ReLU maintains a piecewise linear structure and avoids the vanishing gradient problem, which helps the model learn complex patterns effectively.

Sigmoid (z = 1 / (1 + np.exp(-z)))



The Sigmoid activation function struggled to capture the non-linear and complex patterns effectively. The predicted trajectories exhibit a lot of uncertainty and variability as there are lot of the blue arrows, with a higher density of blue arrows around curves and loops. There is a lack of clear, dark blue arrows along the true trajectory (green path), indicating that the model is less confident and accurate with Sigmoid. The function's limitations in representing complex non-linear relationships contribute to its poorer performance compared to ReLU and Tanh. Sigmoid struggles with more complex non-linear relationships. This saturation limits the model's capacity to accurately follow intricate trajectories.

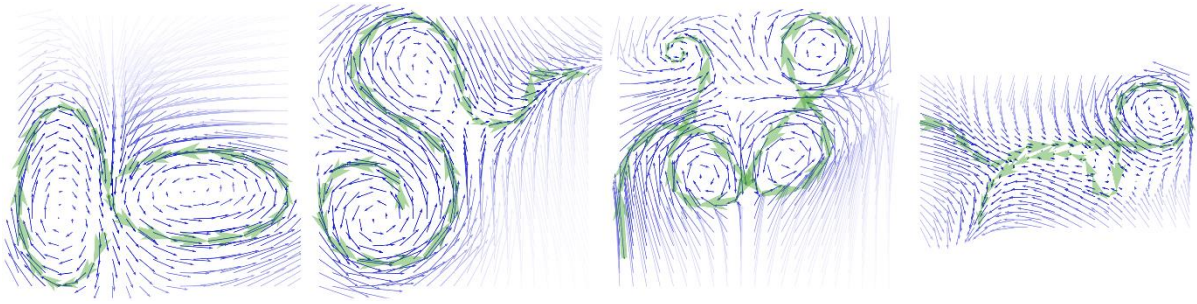
Tanh (z = np.tanh(z))



The Tanh activation function improved upon Sigmoid but still did not match the performance of ReLU. Tanh captured the non-linearity and complexity better than Sigmoid, as indicated by fewer arrows and some dark blue arrows along the green trajectory. However, it still falls short compared to ReLU, as the predictions with Tanh are not as smooth or accurate. Tanh alleviates some issues with gradient saturation compared to Sigmoid by having a broader output range, which helps capture non-linear patterns better. However, it still suffers from some degree of saturation,

ReLU provides more confidence and accuracy with clearer, darker arrows along the green trajectory and fainter arrows outside it, indicating better overall performance in capturing the true paths compared to Sigmoid and Tanh. Sigmoid suffers from the vanishing gradient problem, where gradients become very small for extreme values, hindering the model's ability to learn complex patterns

3) using Nhid = 100 and ReLU for all for consistency
0.1 prior variance:



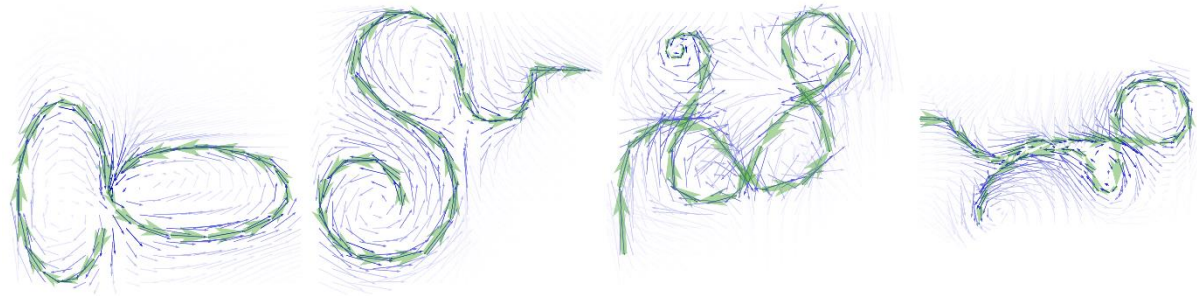
Small prior variance means a strong regularization and strong belief in the prior. This causes less flexibility for the data to influence the weights. Model is more constrained by the prior distribution, so parameters are pulled towards the prior mean (less flexible in adapting to the data).

As we can see that the predicted trajectories are less aligned with the true trajectory (green path). This suggests that the model is too constrained by the prior and fails to capture the complexity of the data.

The presence of many predicted arrows indicates higher uncertainty in the predictions. This is due to the model being too rigid and unable to adapt well to the observed data.

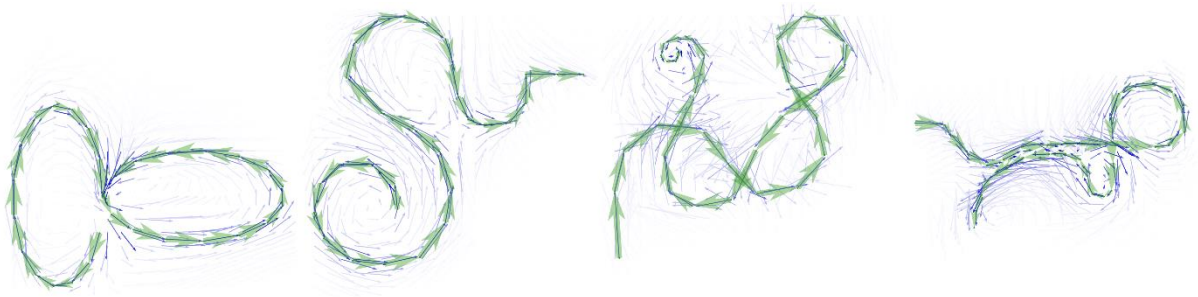
This small prior variance can help prevent overfitting, but it can also lead to underfitting if the prior is too restrictive compared to the actual data which it seems to be doing. The model doesn't capture the data's complexity, leading to poorer performance.

5 prior variance:



This prior variance implies a balanced regularization. The model considered both the prior and the data allowing reasonable flexibility in parameter estimation. There is a balance between prior influence and data fitting. Here we can see that it has predictive accuracy then the prior variance of 5 as we can see a clear dark blue predicted trajectories along the green true path. This suggest that the model is fitting the data more accurately.
While there are still faint arrows around the true path, indicating some uncertainty, it is much less than with the small prior variance. This suggests that the model is less constrained by the prior and more adaptive to the data. The model is not as rigid as with a small prior variance, so it can better capture the complexity of the data while maintaining some level of regularization to avoid overfitting.

25 prior variance:



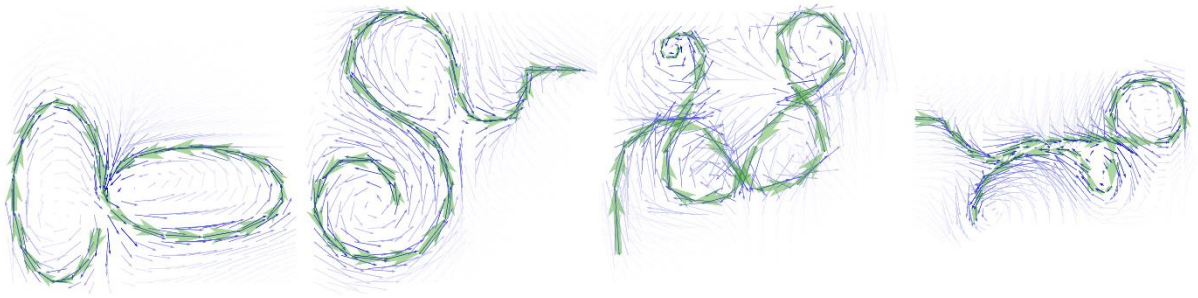
A high prior variance implies weak regularization. With this high prior variance, the model demonstrates its best predictive performance, as indicated by the dark blue arrows closely following the true trajectory (green path). This means the model is highly confident in its predictions, showing a clear fit to the data. The faint arrows (symbolizing uncertainty) are much fewer and less pronounced, indicating reduced uncertainty and increased confidence in the model's predictions.

This shows that the higher the prior variance, the better the model is at predicting as there is weak regularization allowing it to fully adapt to the observed data. This gives the model the flexibility to better capture complex patterns in the data, which leads to higher predictive accuracy. Although saying this, weak regulation could lead to overfitting if the data is noisy.

While the model performs best with high prior variance in this scenario, weak regularization can be risky in the presence of noisy or sparse data. The model might overfit by capturing random fluctuations or noise rather than true underlying patterns.

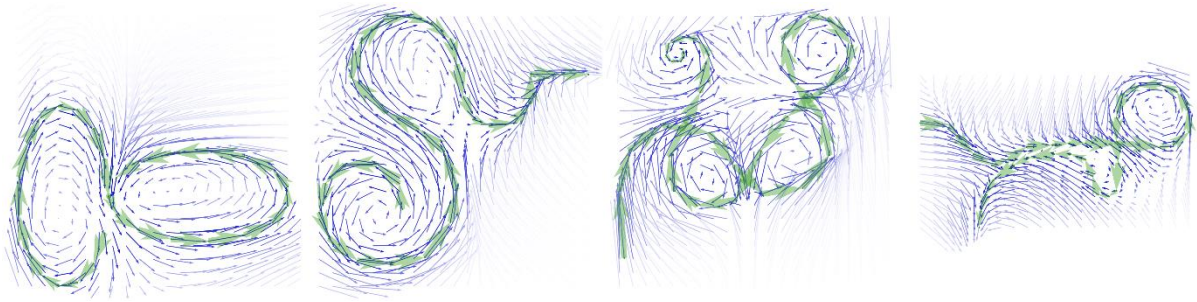
4) using Nhids = 100, ReLU and prior variance = 5

Small Noise variance = 0.01



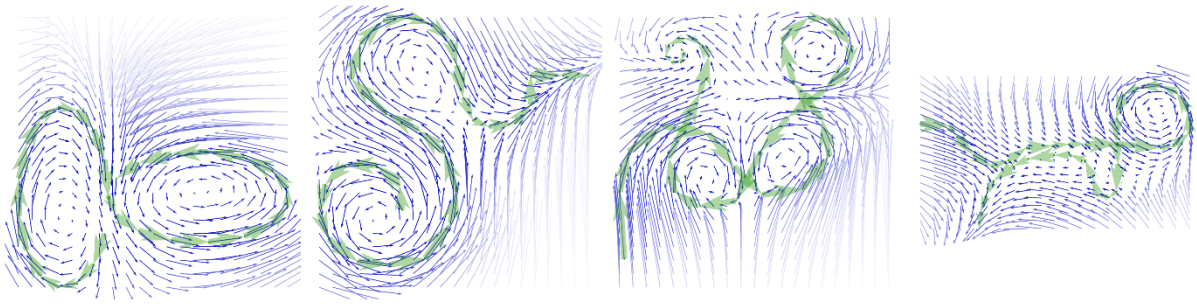
With small noise variance, the model assumes that the observations are very precise with minimal error hence the model is confident in its predictions because it assumes it is very accurate and trustworthy. This is reflected in the visualization by strong, dark predictive arrows along the predicted trajectories and very few faint arrows (uncertainty) outside the path. This shows that in fact that the data has minimal noise. This causes the model to give more weights to the data over any prior assumptions. However, this could lead to the model to overfit if the data contains noise, but I believe that the model hasn't overfit as we can see very dark predicted arrows along the true green path with faint arrows just around the outside of the true path.

Moderate Noise Variance = 0.1



As we increase noise, the model assumes that the data has some among of error in the data. The model is more flexible and accounts for some degree of uncertainty. I do see more predicted arrows from which are more furtherer away from the true trajectory. This means the model is less confident in its predictions and also there is an increase of uncertainty and variability. Although saying this, with moderate noise, the model balances its trust in the data and the assumption that the data might be noisy. This reduces the risk of overfitting, but still keeps a reasonable fit to the true underlying pattern.

Large Noise Variance = 1.0



With large noise variance, the model assumes that the data has a high level of measurement error. The model becomes more cautious in its predictions as it doesn't trust the data as much. Hence the predicted trajectory is the most spread out compared to the other 2 noise variances. This model produces the most number of blue arrow predictions symbolising that it has the most uncertainty and variability out of the 2 noise variance. This does reduce the risk of overfitting, but I believe it has resulted in underfitting and hence why the huge number of blue arrows which means that it hasn't captured the complexity and non-linearity of the data well and so has low confidence predictions.

As we increase noise variance, the model becomes more cautious and less confident, leading to poorer predictions and more variability. For this data, lower noise variance yields better results as the model trusts the data more. However, there is a trade-off: while lower noise variance can improve predictions, it risks overfitting, while higher variance can prevent overfitting but may cause underfitting.

2.3

1)

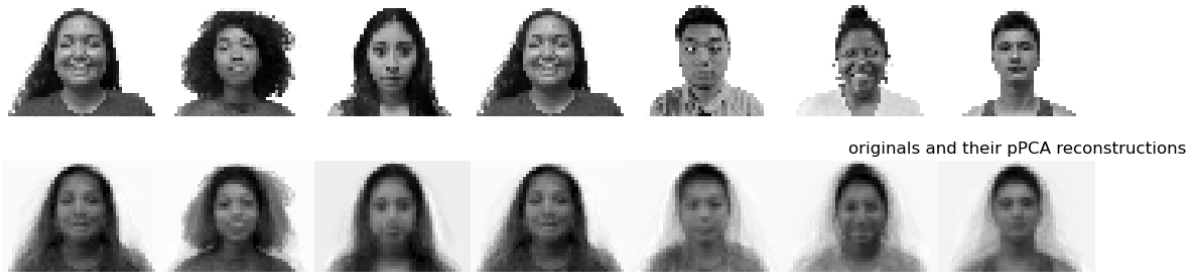
```
Compressing from 1632 to 50 dimensions!

Iteration 1: MSE = 547.8824, sigma^2 = 0.3357
Iteration 2: MSE = 311.1307, sigma^2 = 0.1906
Iteration 3: MSE = 289.5464, sigma^2 = 0.1774
Iteration 4: MSE = 283.0097, sigma^2 = 0.1734
Iteration 5: MSE = 280.2192, sigma^2 = 0.1717
Iteration 6: MSE = 278.7768, sigma^2 = 0.1708
Iteration 7: MSE = 277.9198, sigma^2 = 0.1703
Iteration 8: MSE = 277.3596, sigma^2 = 0.1700
Iteration 9: MSE = 276.9708, sigma^2 = 0.1697
Iteration 10: MSE = 276.6894, sigma^2 = 0.1695
Iteration 11: MSE = 276.4788, sigma^2 = 0.1694
Iteration 12: MSE = 276.3162, sigma^2 = 0.1693
Iteration 13: MSE = 276.1869, sigma^2 = 0.1692
Iteration 14: MSE = 276.0814, sigma^2 = 0.1692
Iteration 15: MSE = 275.9935, sigma^2 = 0.1691
Iteration 16: MSE = 275.9194, sigma^2 = 0.1691
Iteration 17: MSE = 275.8563, sigma^2 = 0.1690
Iteration 18: MSE = 275.8022, sigma^2 = 0.1690
Iteration 19: MSE = 275.7558, sigma^2 = 0.1690
Iteration 20: MSE = 275.7159, sigma^2 = 0.1689
Iteration 21: MSE = 275.6816, sigma^2 = 0.1689
Iteration 22: MSE = 275.6521, sigma^2 = 0.1689
Iteration 23: MSE = 275.6265, sigma^2 = 0.1689
Iteration 24: MSE = 275.6044, sigma^2 = 0.1689
Iteration 25: MSE = 275.5851, sigma^2 = 0.1689
```

The MSE starts at 547.844 and decreases in the first 10 iterations, indicating learning. Afterward, the rate of improvement slows, suggesting the model is nearing convergence. The noise variance (sigma^2) also decreases quickly initially but levels off after the 5th iteration. This gradual reduction suggests the model is stabilizing. The slower changes in MSE and sigma^2 towards the end may indicate the model is close to optimal and could be starting to overfit. The lowest MSE become 275.5851 which shows data was compressed successfully into lower-dimensional latent space while retaining most of the relevant information for reconstruction.

There are most data points within a range of -2 to 2 for both latent dimensions, with some outliers beyond this range nearing -3 for latent dimension 1 and -3 and 3 for latent dimension 2. The majority of points are grouped around 0, indicating that most of the data is centred around the average values for both latent dimensions. This does match my expectations as:

- Data points are centred around zero, which is typical for dimensionality reduction methods like PPCA. This centring helps in providing a balanced and effective representation of the data.
- Data points are evenly distributed between -2 and 2 in both dimensions, indicating that the PPCA model captures the data's variability well.
- The presence of some outliers outside the -2 to 2 range is expected and often indicates data points that are unique or significantly different from the majority.
- The overall shape of the data in the latent space is circular, suggesting that the model is effectively capturing the data's structure.



The trained network produces slightly blurred images, but they still capture the main features well. This blurring is normal because pPCA reduces the data to a lower-dimensional space, which can lead to some loss of detail. Overall, the reconstructions are accurate and show that the model has learned the key features of the face images.

2)

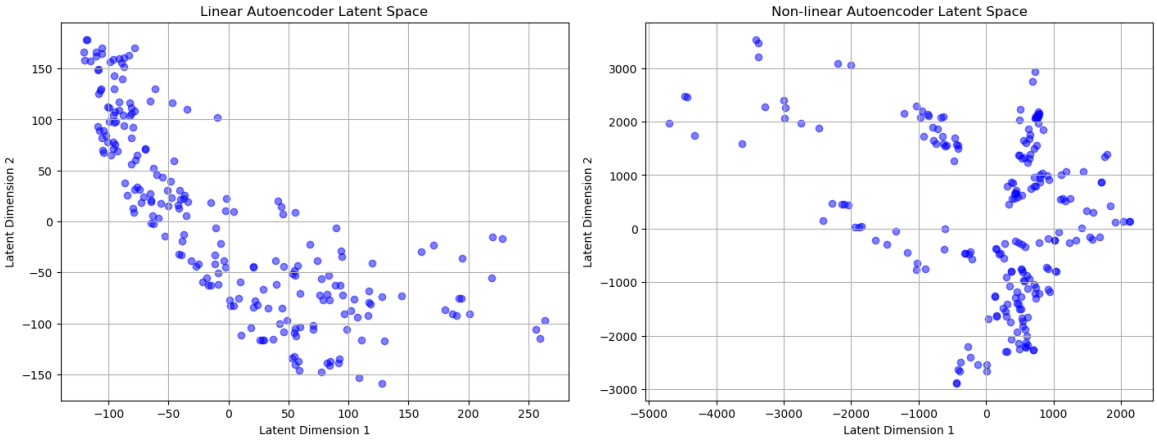
```
Training Linear Autoencoder...
Epoch [100/1000], Loss: 2395.8398
Epoch [200/1000], Loss: 2178.3054
Epoch [300/1000], Loss: 1915.5853
Epoch [400/1000], Loss: 1777.1329
Epoch [500/1000], Loss: 1718.5327
Epoch [600/1000], Loss: 1686.1238
Epoch [700/1000], Loss: 1652.2639
Epoch [800/1000], Loss: 1609.6318
Epoch [900/1000], Loss: 1563.7600
Epoch [1000/1000], Loss: 1524.8264
```

```
Training Non-linear Autoencoder...
Epoch [100/1000], Loss: 1308.4224
Epoch [200/1000], Loss: 1271.8860
Epoch [300/1000], Loss: 1217.7725
Epoch [400/1000], Loss: 1147.7437
Epoch [500/1000], Loss: 1103.1282
Epoch [600/1000], Loss: 1077.0930
Epoch [700/1000], Loss: 1059.1536
Epoch [800/1000], Loss: 1047.5658
Epoch [900/1000], Loss: 1037.7426
Epoch [1000/1000], Loss: 1034.9126
```

```
Compressing from 1632 to 50 dimensions!

Iteration 1: MSE = 547.8824, sigma^2 = 0.3357
Iteration 2: MSE = 311.1307, sigma^2 = 0.1906
Iteration 3: MSE = 289.5464, sigma^2 = 0.1774
Iteration 4: MSE = 283.0097, sigma^2 = 0.1734
Iteration 5: MSE = 280.2192, sigma^2 = 0.1717
Iteration 6: MSE = 278.7768, sigma^2 = 0.1708
Iteration 7: MSE = 277.9198, sigma^2 = 0.1703
Iteration 8: MSE = 277.3596, sigma^2 = 0.1700
Iteration 9: MSE = 276.9708, sigma^2 = 0.1697
Iteration 10: MSE = 276.6894, sigma^2 = 0.1695
Iteration 11: MSE = 276.4788, sigma^2 = 0.1694
Iteration 12: MSE = 276.3162, sigma^2 = 0.1693
Iteration 13: MSE = 276.1869, sigma^2 = 0.1692
Iteration 14: MSE = 276.0814, sigma^2 = 0.1692
Iteration 15: MSE = 275.9935, sigma^2 = 0.1691
Iteration 16: MSE = 275.9194, sigma^2 = 0.1691
Iteration 17: MSE = 275.8563, sigma^2 = 0.1690
Iteration 18: MSE = 275.8022, sigma^2 = 0.1690
Iteration 19: MSE = 275.7558, sigma^2 = 0.1690
Iteration 20: MSE = 275.7159, sigma^2 = 0.1689
Iteration 21: MSE = 275.6816, sigma^2 = 0.1689
Iteration 22: MSE = 275.6521, sigma^2 = 0.1689
Iteration 23: MSE = 275.6265, sigma^2 = 0.1689
Iteration 24: MSE = 275.6044, sigma^2 = 0.1689
Iteration 25: MSE = 275.5851, sigma^2 = 0.1689
```

The linear autoencoder reached a final loss of 1524.8264 which is highest compared to non-linear autoencoder and pPCA. This means that the linear autoencoder was less effective at compressing the data while maintaining reconstruction accuracy. The non-linear autoencoder reached a lower loss of 1034.9126 which is roughly 500 less. This symbolises that the extra hidden layers and ReLU activation functions allowed to capture complex, non-linear patterns in the data as it introduces non-linearity and hence decreasing the loss. However, the non-linear autoencoder performed worse the pPCA. PPCA reached a final MSE of 275.5851 meaning it's the best out of all the others as it's the lowest, and was the best at capture complex, non-linear patterns in the data and hence better at reconstructing. PPCA was the best at maintaining the essential features of the data while compressing it.



For the linear autoencoder latent space, I can see that as latent dimension 1 increase from -100 to 50 , latent dimensions 2 decreases from 150 to -150. After latent dimension 50 , data points stay around -100 latent dimension 2 and there are less data points after 150 latent dimension 1. And possibly outliers after 200 latent dimension 1.

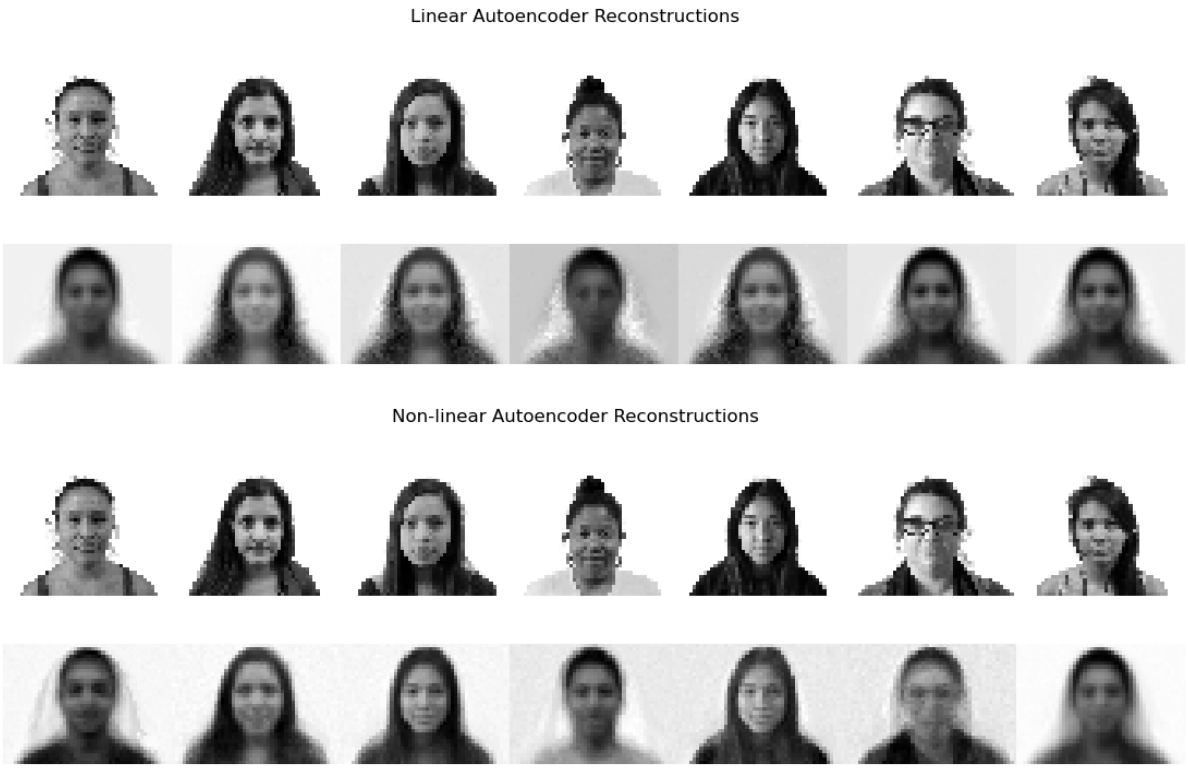
For non-linear autoencoder latent space, I can see most data points being around 0. With many clusters from 0 to 1000 latent dimension 1, ranging from 2000 to -2000 on the y axis (latent

dimension 2). But there are some further away data points around -5000 to -3000 on latent dimension 1 which could be considered as outliers.

I expected this shape as I knew linear autoencoder won't be able to capture complex non-linear patterns as good as the non-linear autoencoder. Non-linear autoencoder does a better job at capturing the complex data. Also, loss for linear autoencoder at the end was higher then loss for non-linear autoencoder hence whey I knew the shape will be more clustered/compact for the non-linear autoencoder latent space.

For linear autoencoder, the shape of the encoded points suggests that the linear model might be struggling to capture the complexity of the data, leading to a less compact and more scattered distribution.

For non-linear autoencoder, the more centralized clustering around 0 and the wider spread on the y-axis indicate that the non-linear model is better at capturing complex relationships in the data. The presence of clusters and some distant outliers aligns with the expectation that a non-linear model can represent more complex data patterns.



For linear autoencoder, the reconstructions appear to be more blurred and don't have much fine details compared to non-linear autoencoder. This was expected as the final loss for linear autoencoder was higher then the final loss for non-linear autoencoder. Linear autoencoder wasn't able to capture more complex/non-linear patterns and features in the data that well compared to non-linear autoencoder.
For non-linear autoencoder, the reconstructions appear to be more visibly better, with more defined features. Especially for 5th image, non-linear autoencoder does a better job then linear autoencoder as its clearer and closer to the actual image. This is because non-linear

autoencoder can capture complex patterns in the data.
But they both are still blurry but expected since dimensionality reduction involves compressing information and can lead to some loss of detail. And for some images such as the 7th image, linear and non-linear reconstructions are very similar so not all reconstructed images for non-linear auto encoder are significantly better.