Problem 1

To get $\boldsymbol{w}^*_{LS}$ and $\boldsymbol{w}^*_{LASSO}$, we used the gradient descent to match the minimum of the object.

For linear regression, to fit the shape of the data matrix `(10,4)`, the object function and gradient can be computed as

$$\begin{gathered} L(\boldsymbol{w}) = (X\boldsymbol{w} - \boldsymbol{y})^T(X\boldsymbol{w} - \boldsymbol{y}) \\ \nabla L = 2(X^T\boldsymbol{w}^T - \boldsymbol{y}^T)X \end{gathered} \tag{1}$$

where $\boldsymbol{w}$ is assumed as a column vector with shape `(4,1)`. And the gradient descent

$$\boldsymbol{w}' = \boldsymbol{w} - a\nabla L \tag{2}$$

a here is the learning rate.

The procedure is similar in LASSO algorithm, we have

$$\begin{gathered} L(\boldsymbol{w}) = (X\boldsymbol{w} - \boldsymbol{y})^T(X\boldsymbol{w} - \boldsymbol{y}) + C\|\boldsymbol{w}\|_1 \\ \nabla L = 2(X^T\boldsymbol{w}^T - \boldsymbol{y}^T)X + C\nabla\|\boldsymbol{w}\|_1 \end{gathered} \tag{3}$$

and $\nabla\|\boldsymbol{w}\|_1$ can be calculated as

$$(\nabla\|\boldsymbol{w}\|_1)_i = \begin{cases} 1 & w_i > 0 \\ -1 & w_i < 0 \\ \forall k \in (-1, 1) & w_i = 0 \end{cases} \tag{4}$$

in the program, we just simply take $k = 0$ for computation. And the gradient descent

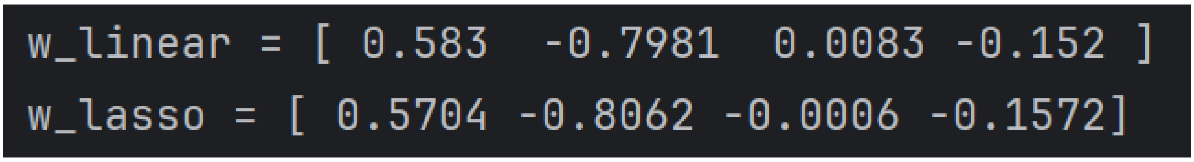
$$\boldsymbol{w}' = \boldsymbol{w} - a\nabla L \tag{5}$$

The gradient descent stops when the iteration number is larger than the threshold `max_iter` or the difference after the iteration $\|\boldsymbol{w}' - \boldsymbol{w}\|$ is smaller than the threshold `epsilon`.

The result of the two algorithms at parameters

```
alpha = 1e-2    #learning rate
epsilon = 1e-2  #tolerance
max_iter = 5000
```

is that



```
w_linear = [ 0.583  -0.7981  0.0083 -0.152 ]
w_lasso = [ 0.5704 -0.8062 -0.0006 -0.1572]
```

herein we find that the result of LASSO has apparently smaller 3rd component than that of linear regression, which weakens the influence of the 3rd feature, and the weight becomes sparse.

Problem 2

For simplification, we just assume that the data are normalized $\mu_x = 0$.

Consider the PCA aiming at maximizing variance of projected data:

$$
\begin{aligned}
U &= \max_U \mathbb{E}[\|U^T \boldsymbol{x}\|^2] \\
&= \frac{1}{n} \max_U \sum_{i=1}^{n} \|U^T \boldsymbol{x}_i\|^2 \\
s.t.\, &U^T U = I_k
\end{aligned}
\tag{6}
$$

herein, $\boldsymbol{x}_i$ is the i-th data point of the data set.

Now consider the view that minimizing the reconstruction error, we have

$$
\begin{gathered}
U = \min_U \|X - UU^T X\|^2 \\
s.t.\ U^T U = I_k
\end{gathered}
\tag{7}
$$

which is equivalent to

$$
\begin{aligned}
U &= \min_U \|(I - UU^T)X\|^2 \\
&= \min_U \sum_{i=1}^{n} \boldsymbol{x}_i^T (I - UU^T)(I - UU^T)\boldsymbol{x}_i \\
&= \min_U \sum_{i=1}^{n} \boldsymbol{x}_i^T (I - UU^T)\boldsymbol{x}_i \\
&= \min_U \sum_{i=1}^{n} \left[\boldsymbol{x}_i^T \boldsymbol{x}_i - \|U^T \boldsymbol{x_i}\|^2\right] \\
s.t.\, &U^T U = I_k
\end{aligned}
\tag{8}
$$

herein $\boldsymbol{x}_i$ are the i-th column of X, and the constraint are used in the 3rd equals sign.

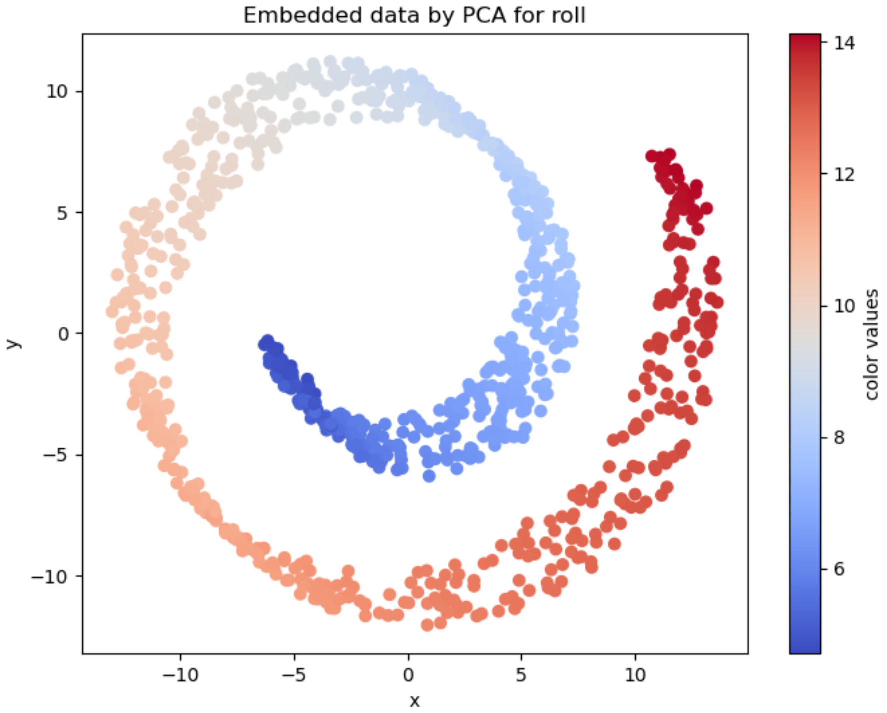
Considering that the first term above is a constant, the optimization now becomes

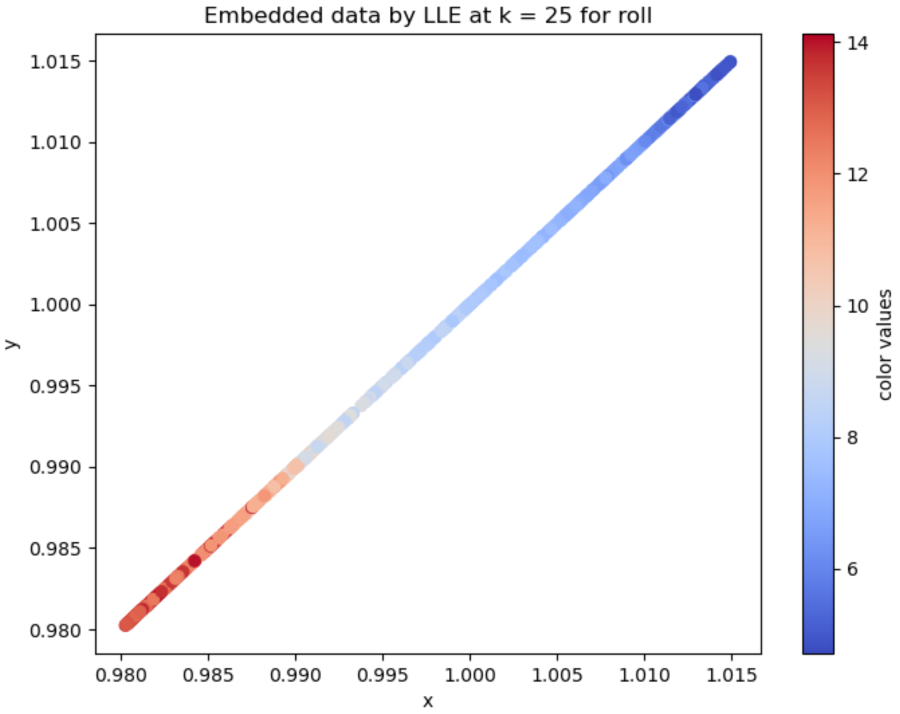
$$
\begin{aligned}
U &= \max_U \sum_{i=1}^{n} \|U^T \boldsymbol{x}_i\|^2 \\
s.t.\, &U^T U = I_k
\end{aligned}
\tag{9}
$$

which is equivalent to the view of maximizing the projected variance. Q.E.D.

Programming

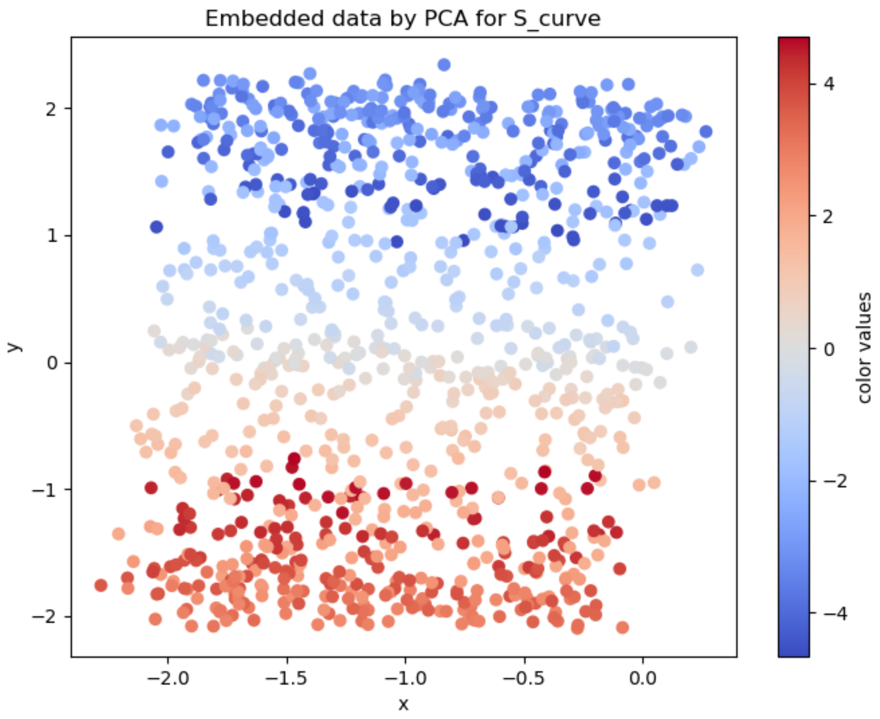
The results obtained by PCA and LLE of Swiss roll case are listed here:

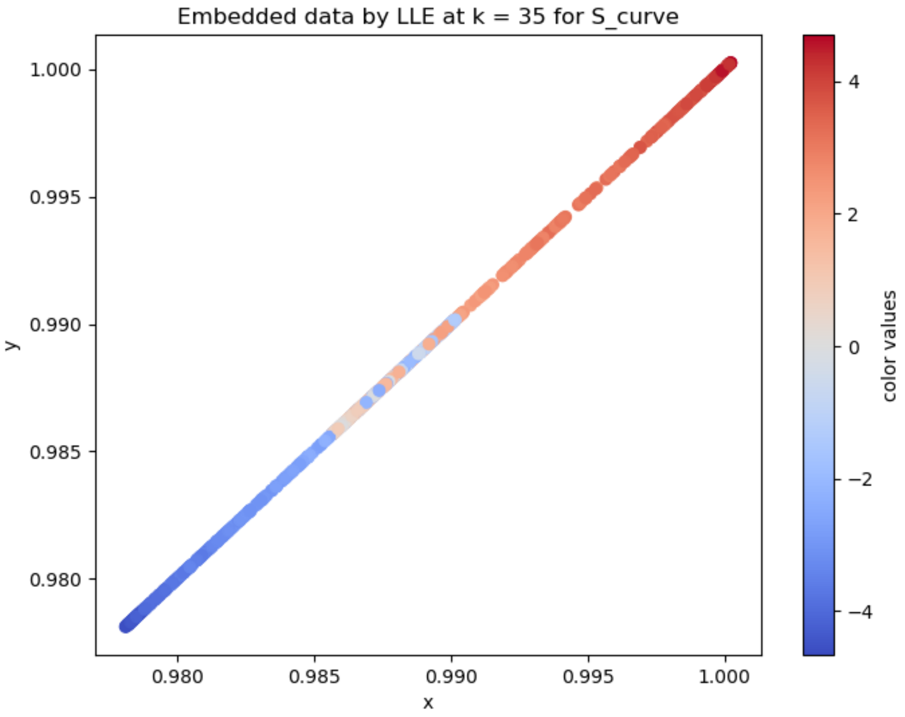




The result given by PCA just projected the Swiss roll in the principal direction, i.e. for the roll, the axial direction. However, the result given by LLE flattened the manifold in 1 direction (for the roll is a 2-D manifold in $\mathbb{R}^3$, and the labels varies only along a particular direction of the geometric coordinates). The difference comes from the different origin of the 2 algorithms: PCA does projection, while LLE shows the manifold structure.

For verification, the other manifold case I choose is the S curve case, and the results of PCA and LLE are as below:





It's clear that in the S curve case, the PCA algorithm performs catastrophically, and the LLE algorithm remains excellence. The features of projection and flattening are remained respectively, which origin from the algorithms.

I's also notable that the choice of neighbor numbers `k` in LLE algorithm is quite subtle, only several particular `k` values can capture the right local structure of the manifold, which means hard searching of the right parameter.

All the parameters can be found in the program.