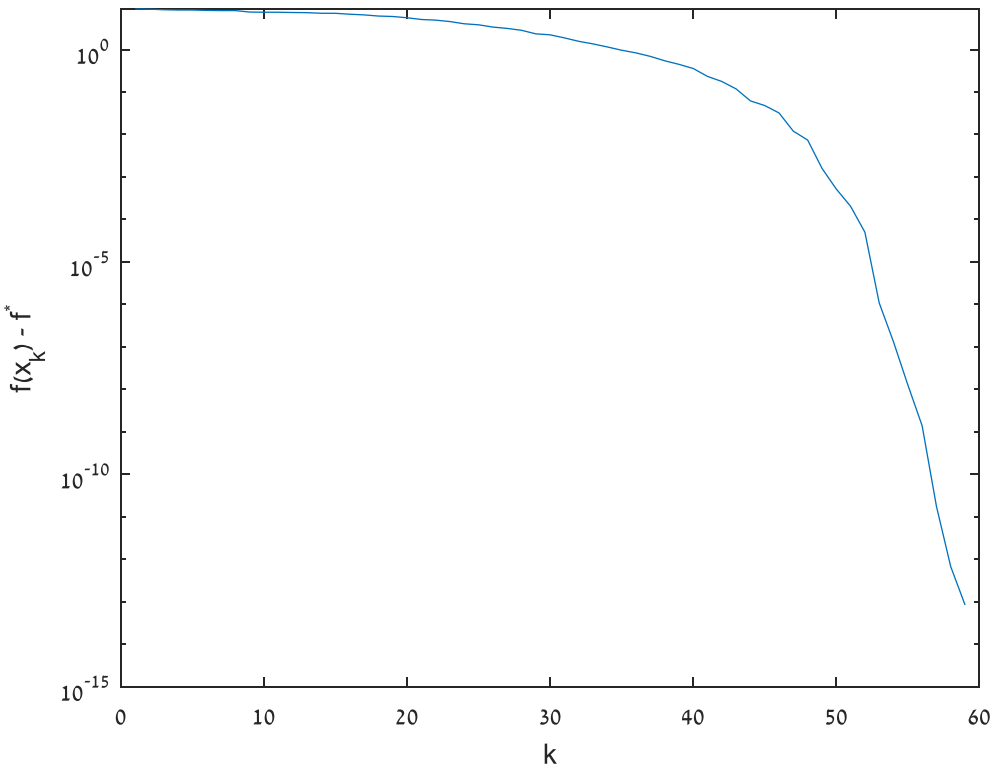
## 1.2 Find the Minimum of the Rosenbrock Function [Code and Report]

Use the BFGS method to find the minimum point of the Rosenbrock function for the case of $n = 10$ and starting point $x_0 = (0, 0, \ldots, 0) \in \mathbb{R}^{10}$. As a reminder, the Rosenbrock function is given as follows:

$$f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n-1} \left( (1 - x_i)^2 + 100 \left( x_{i+1} - x_i^2 \right)^2 \right) \tag{1}$$

plot the convergence curve $f(x_k) - f^*$ (the y-axis) as function of the iteration number $k$ (the x-axis), where $f^*$ is the optimal/minimal value of the Rosenbrock function. Use logarithmic scale for the y-axis.

From the following graph, we can see that the convergence happens swiftly using BFGS, where the number of iterations is relatively small and thus the running time for rosenbrock function is significantly less

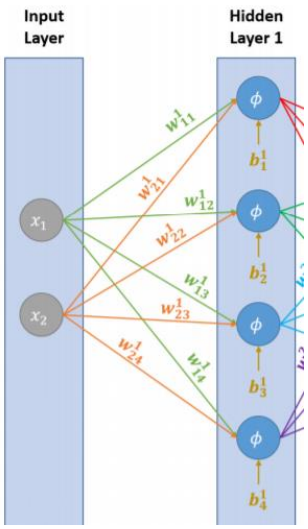


### 1.3.4 Explicit Expression of the Neural Network Model [Report]

Complete the following:

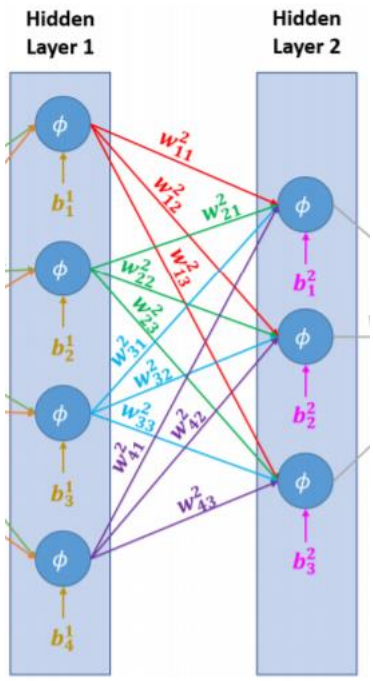
- Write an explicit vectorized expression for the neural network given in section 1.3.1. In other words, you should write the explicit expression of $F(x|\mathcal{W})$. Please note, $\mathcal{W}$ represent the set of model parameters. That is, in our feed-froward neural network context, $\mathcal{W}$ consists of all weight matrices $\{W_i\}$ and all bias vectors $\{b_i\}$ across all layers. Even though your explicit expression for $F(x|\mathcal{W})$ should be written in terms of $x$ (the input vector), $\{W_i\}$, $\{b_i\}$ and the activation function $\phi$, you can also think of $\mathcal{W}$, for implementation purposes, as a single long vector which is made of stacking the columns/rows of the matrices $\{W_i\}$ and the vectors $\{b_i\}$ one on top of the other. For guidance, please refer to figures 5 and 6.

the output of the following first hidden layer would be:




$$\emptyset\left(\begin{bmatrix} w^1_{11} & w^1_{21} \\ w^1_{12} & w^1_{22} \\ w^1_{13} & w^1_{23} \\ w^1_{14} & w^1_{24} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} b_{!^1} \\ b^1_2 \\ b^1_3 \\ b^1_4 \end{bmatrix}\right) = \begin{bmatrix} \emptyset(w^1_{11}x_1 + w^1_{21}x_2 + b^1_1) \\ \emptyset(w^1_{12}x_1 + w^1_{22}x_2 + b^1_2) \\ \emptyset(w^1_{13}x_1 + w^1_{23}x_2 + b^1_3) \\ \emptyset(w^1_{14}x_1 + w^1_{24}x_2 + b^1_4) \end{bmatrix} = \emptyset({w^1}^T x_i + b^1_i)$$

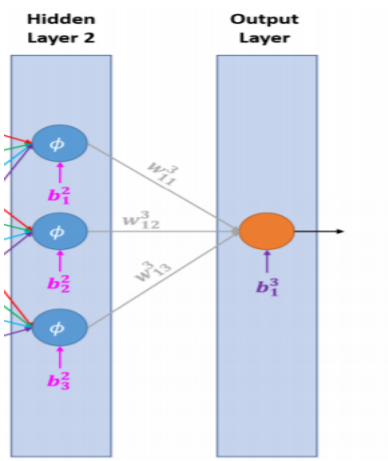
the output of the following second hidden layer would be:




$$\emptyset\left(\begin{bmatrix} w_{11}^2 w_{21}^2 w_{31}^2 w_{41}^2 \\ w_{12}^2 w_{22}^2 w_{32}^2 w_{42}^2 \\ w_{13}^2 w_{23}^2 w_{33}^2 w_{43}^2 \end{bmatrix} \begin{bmatrix} \emptyset(w_{11}^1 x_1 + w_{21}^1 x_2 + b_{1^1}) \\ \emptyset(w_{12}^1 x_1 + w_{22}^1 x_2 + b_2^1) \\ \emptyset(w_{13}^1 x_1 + w_{23}^1 x_2 + b_3^1) \\ \emptyset(w_{14}^1 x_1 + w_{24}^1 x_2 + b_4^1) \end{bmatrix} + \begin{bmatrix} b_!^2 \\ b_2^2 \\ b_3^2 \end{bmatrix}\right) = \emptyset(w^{2^T} \emptyset\left(w^{1^T} x_i + b_i^1\right) + b_i^2)$$

$$\begin{bmatrix} \emptyset(w_{11}^2 \cdot \emptyset(w_{11}^1 x_1 + w_{21}^1 x_2 + b_{1^1}) + w_{21}^2 \cdot \emptyset(w_{12}^1 x_1 + w_{22}^1 x_2 + b_2^1) + w_{31}^2 \emptyset(w_{13}^1 x_1 + w_{23}^1 x_2 + b_3^1) + w_{41}^2 \emptyset(w_{14}^1 x_1 + w_{24}^1 x_2 + b_4^1) + b_1^2) \\ \emptyset(w_{12}^2 \cdot \emptyset(w_{11}^1 x_1 + w_{21}^1 x_2 + b_{1^1}) + w_{22}^2 \cdot \emptyset(w_{12}^1 x_1 + w_{22}^1 x_2 + b_2^1) + w_{32}^2 \emptyset(w_{13}^1 x_1 + w_{23}^1 x_2 + b_3^1) + w_{42}^2 \emptyset(w_{14}^1 x_1 + w_{24}^1 x_2 + b_4^1) + b_2^2) \\ \emptyset(w_{13}^2 \cdot \emptyset(w_{11}^1 x_1 + w_{21}^1 x_2 + b_{1^1}) + w_{23}^2 \cdot \emptyset(w_{12}^1 x_1 + w_{22}^1 x_2 + b_2^1) + w_{33}^2 \emptyset(w_{13}^1 x_1 + w_{23}^1 x_2 + b_3^1) + w_{43}^2 \emptyset(w_{14}^1 x_1 + w_{24}^1 x_2 + b_4^1) + b_3^2) \end{bmatrix}$$

the output of the following output  layer would be:



$$\left( [w_{11}^3 \quad w_{12}^3 \quad w_{13}^3] \begin{bmatrix} \emptyset(w_{11}^2 \cdot \emptyset(w_{11}^1 x_1 + w_{21}^1 x_2 + b_{1^1}) + w_{21}^2 \cdot \emptyset(w_{12}^1 x_1 + w_{22}^1 x_2 + b_2^1) + w_{31}^2 \emptyset(w_{13}^1 x_1 + w_{23}^1 x_2 + b_3^1) + w_{41}^2 \emptyset(w_{14}^1 x_1 + w_{24}^1 x_2 + b_4^1) + b_1^2) \\ \emptyset(w_{12}^2 \cdot \emptyset(w_{11}^1 x_1 + w_{21}^1 x_2 + b_{1^1}) + w_{22}^2 \cdot \emptyset(w_{12}^1 x_1 + w_{22}^1 x_2 + b_2^1) + w_{32}^2 \emptyset(w_{13}^1 x_1 + w_{23}^1 x_2 + b_3^1) + w_{42}^2 \emptyset(w_{14}^1 x_1 + w_{24}^1 x_2 + b_4^1) + b_2^2) \\ \emptyset(w_{13}^2 \cdot \emptyset(w_{11}^1 x_1 + w_{21}^1 x_2 + b_{1^1}) + w_{23}^2 \cdot \emptyset(w_{12}^1 x_1 + w_{22}^1 x_2 + b_2^1) + w_{33}^2 \emptyset(w_{13}^1 x_1 + w_{23}^1 x_2 + b_3^1) + w_{43}^2 \emptyset(w_{14}^1 x_1 + w_{24}^1 x_2 + b_4^1) + b_3^2) \end{bmatrix} + b_1^3 \right) \boxed{= \left( w^{3^T} \emptyset \left( w^{2^T} \emptyset \left( w^{1^T} x_i + b_i^1 \right) + b_i^2 \right) + b^3 \right)} =$$

$$[w_{11}^3 \cdot \emptyset(w_{11}^2 \cdot \emptyset(w_{11}^1 x_1 + w_{21}^1 x_2 + b_{1^1}) + w_{21}^2 \cdot \emptyset(w_{12}^1 x_1 + w_{22}^1 x_2 + b_2^1) + w_{31}^2 \emptyset(w_{13}^1 x_1 + w_{23}^1 x_2 + b_3^1) + w_{41}^2 \emptyset(w_{14}^1 x_1 + w_{24}^1 x_2 + b_4^1) + b_1^2) +$$

$$w_{12}^3 \cdot \emptyset(w_{12}^2 \cdot \emptyset(w_{11}^1 x_1 + w_{21}^1 x_2 + b_{1^1}) + w_{22}^2 \cdot \emptyset(w_{12}^1 x_1 + w_{22}^1 x_2 + b_2^1) + w_{32}^2 \emptyset(w_{13}^1 x_1 + w_{23}^1 x_2 + b_3^1) + w_{42}^2 \emptyset(w_{14}^1 x_1 + w_{24}^1 x_2 + b_4^1) + b_2^2) +$$

$$w_{13}^3 \cdot \emptyset(w_{13}^2 \cdot \emptyset(w_{11}^1 x_1 + w_{21}^1 x_2 + b_{1^1}) + w_{23}^2 \cdot \emptyset(w_{12}^1 x_1 + w_{22}^1 x_2 + b_2^1) + w_{33}^2 \emptyset(w_{13}^1 x_1 + w_{23}^1 x_2 + b_3^1) + w_{43}^2 \emptyset(w_{14}^1 x_1 + w_{24}^1 x_2 + b_4^1) + b_3^2) + b_1^3]$$

To summarize:

- $W^1: 2x4$ ; $b^1: 4x1$
- $W^2: 4x3$ ; $b^2: 3x1$
- $W^3: 3x1$ ; $b^3: 1x1$
- Our output function expression would be:

$$\boxed{F(x|w) = w^{3^T} \emptyset_2 \left( w^{2^T} \emptyset_1 \left( w^{1^T} x_i + b_i^1 \right) + b_i^2 \right) + b^3}$$

When:

$$\phi(x) = \tanh(x)\,;\ \phi'(x) = \frac{4e^{-2x}}{(1+e^{-2x})^2}$$

$$\varphi_1 \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} = \begin{pmatrix} \frac{e^{u_1} - e^{-u_!}}{e^{u_1} + e^{-u_1}} \\ \frac{e^{u_2} - e^{-u_2}}{e^{u_2} + e^{-u_2}} \\ \frac{e^{u_3} - e^{-u_3}}{e^{u_3} + e^{-u_3}} \end{pmatrix} \varphi_2 \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} \frac{e^{u_1} - e^{-u_!}}{e^{u_1} + e^{-u_1}} \\ \frac{e^{u_2} - e^{-u_2}}{e^{u_2} + e^{-u_2}} \\ \frac{e^{u_3} - e^{-u_3}}{e^{u_3} + e^{-u_3}} \end{pmatrix}$$

$$\varphi_1' \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{pmatrix} = \begin{pmatrix} \phi'(u_1) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \phi'(u_4) \end{pmatrix};\ \varphi_2' \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} \phi'(u_1) & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \phi'(u_3) \end{pmatrix}$$

### 1.3.7 Evaluating the Loss Function's Derivative with Respect to Network's Output of Single Training Example [Code] (Updated: 29/05/2021)

In this section we will treat the loss function as a standalone module. Write a routine that calculates the derivative of the loss function with respect to the neural network's output. That is, you should derive and implement $\frac{\partial L}{\partial F(x_i, W)}$.

$$L = \frac{1}{n}\sum_{i=1}^{n}\left(\left(F\left(x^i, W\right) - y_i\right)^2\right); \nabla_w L = \frac{1}{n}\sum_{i=1}^{n}\nabla_w\left(\left(F\left(x^i, W\right) - y_i\right)^2\right)$$

$$\frac{\partial L}{\partial F(x_i, w)} = \frac{\partial \frac{1}{n}\sum_{i=1}^{n}\left(\left(F\left(x^i, W\right) - y_i\right)^2\right)}{\partial F(x_i, w)} = \frac{\frac{1}{n}\sum_{i=1}^{n}\partial\left(\left(F\left(x^i, W\right) - y_i\right)^2\right)}{\partial F(x_i, w)}$$

$$= \frac{\frac{1}{n}\sum_{i=1}^{n}\partial(\psi(r))}{\partial r_i} = \frac{1}{n}\sum_{i=1}^{n}\frac{\partial\left(\psi(r_i)\right)}{\partial r_i}$$

We notice we want output of a single training example:

$$F(x^i|w) = {w^3}^T \emptyset_2\left({w^2}^T \emptyset_1\left({w^1}^T x_i + b_i^1\right) + b_i^2\right) + b^3$$

$$r_i = F\left(x^i, W\right) - y_i$$

$$\psi(r) = r^2$$

$$\psi'(r) = 2rdr$$

$$\frac{\partial L}{\partial F(x_i, w)} = \frac{\left(F\left(x^i, W\right) - y_i\right)^2}{\partial F(x_i, w)} = \frac{\partial(\psi(r))}{\partial r_i}$$

$$\frac{\partial L}{\partial F(x_i, w)} = 2rdr = 2(F\left(x^i, W\right) - y_i)$$

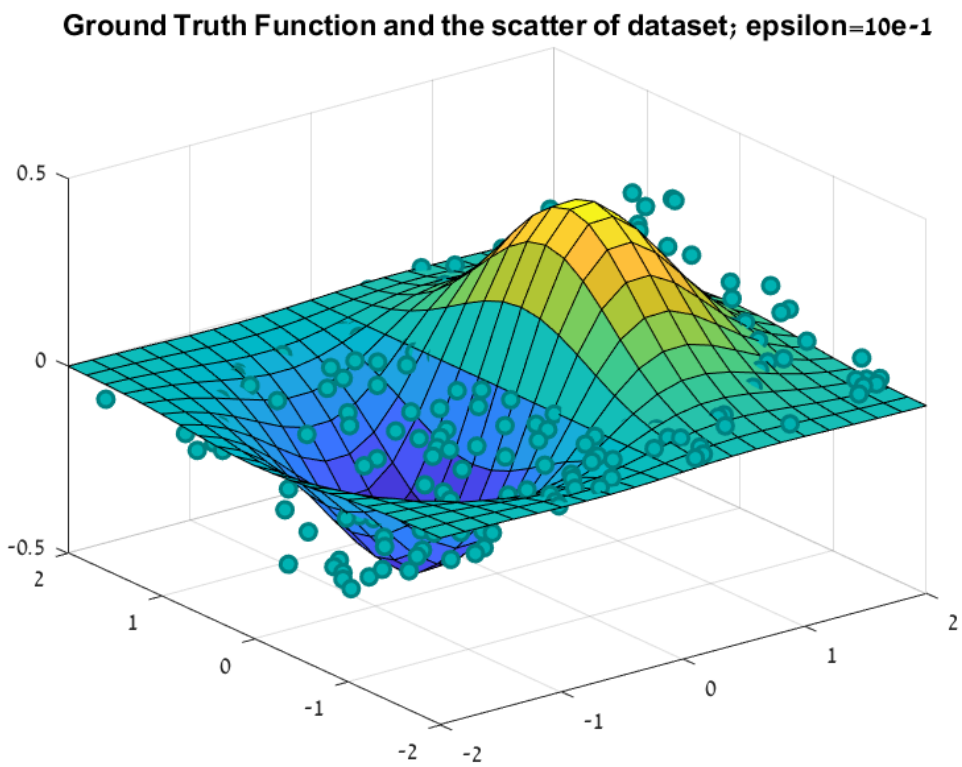
### 1.3.14 Combining it All Together [Code and Report] (Updated: 27/05/2021)

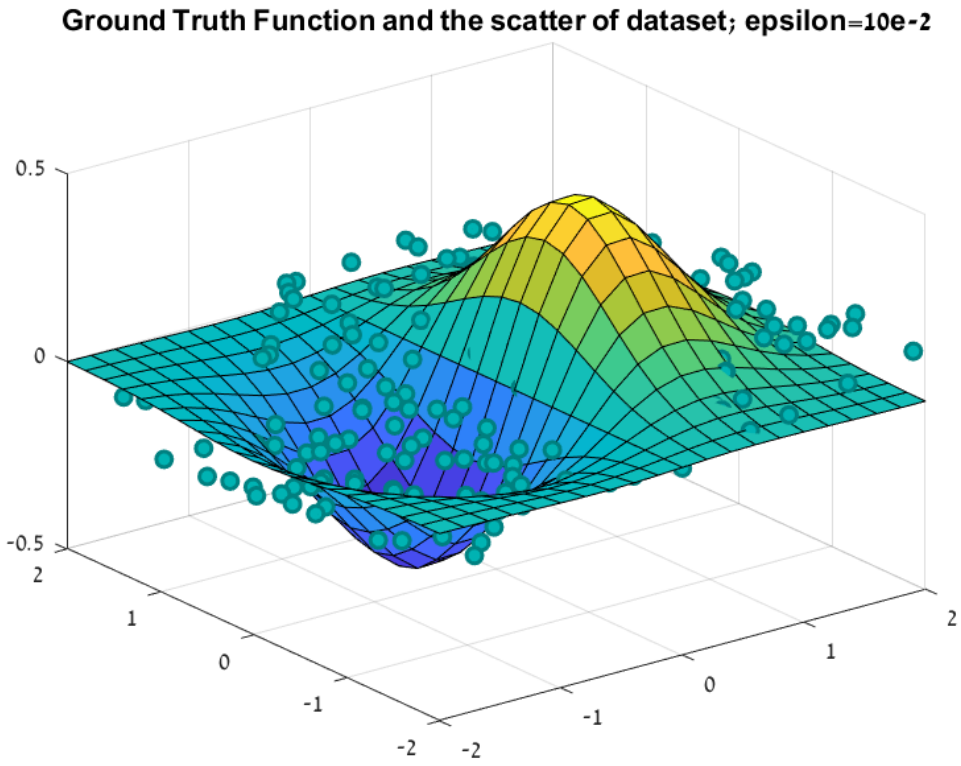
In order to train the neural network and plot the results, follow the steps bellow:

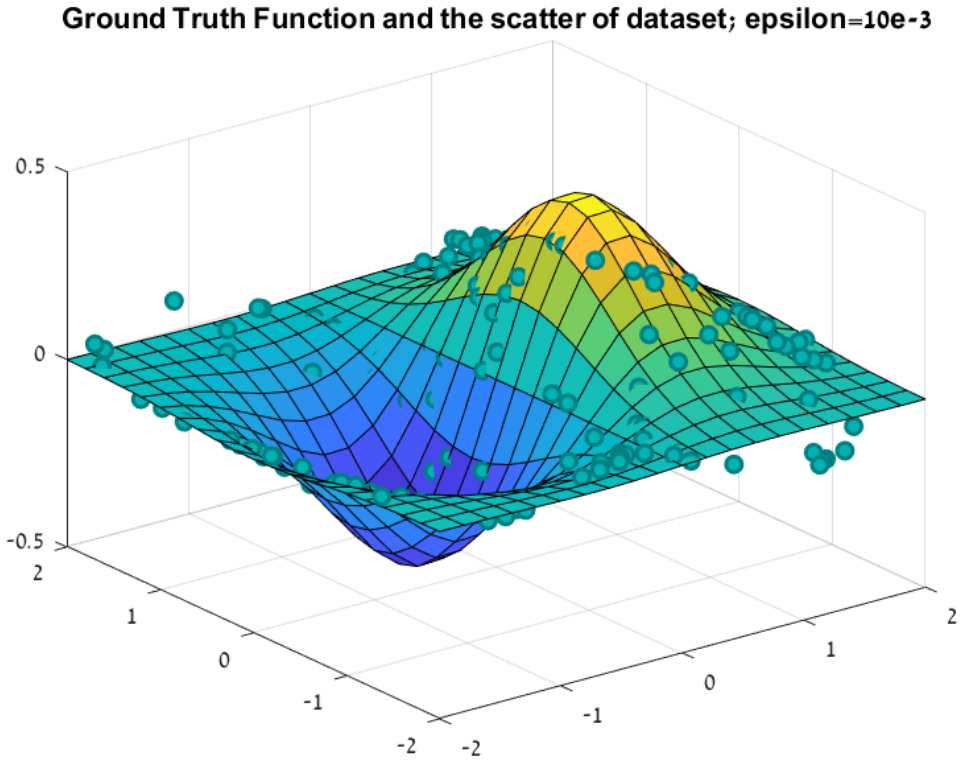
In the plots we see the ground truth function and upon it, a scatter of the reconstruction of the test set that was estimated using our trained network, both of which are limited to the subspace [-2,2]x[-2,2]. Each plot shows the scattering and the ground function for a different epsilon which is the stopping condition on the gradient in the BFGS algorithm.

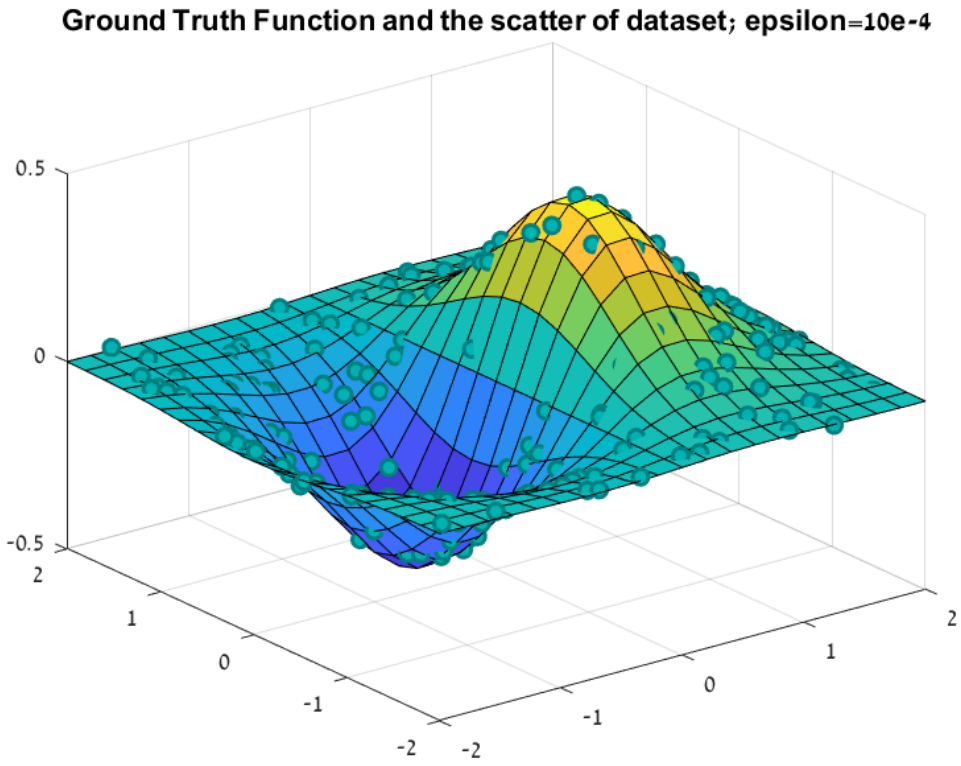
Conclusions:

We can see from the graphs that the smaller the epsilon the more accurate the scatter matches the ground truth function, meaning that the reconstruction of the test set using the trained network works best when the stopping condition is as close to zero as possible.




Ground Truth Function and the scatter of dataset; epsilon=10e-2
0.5
0
-0.5
2
1
0
-1
-2
-2
-1
0
1
2


Ground Truth Function and the scatter of dataset; epsilon=10e-3
0.5
0
-0.5
2
1
0
-1
-2
-2
-1
0
1
2


Ground Truth Function and the scatter of dataset; epsilon=10e-4
0.5
0
-0.5
2
1
0
-1
-2
-2
-1
0
1
2