Hands-On
DRAFT

1 Problem Definition

The objective is to minimize the mean squared local hedging error using a feed-forward neural network model. The profit/loss of the portfolio for one option i, also known as the local hedging error, is given by

$$\Delta V - \delta \Delta S = (V_{t+1} - V_t) - \delta_t(\Delta S_{t+1} - \Delta S_t), \text{ for any } t \in [0, T]$$

From this equation it is clear that the delta that leads to zero hedging error $\delta_t = \Delta V_{t+1}/\Delta S_{t+1}$ is not observable in time t.
So we aim to solve the following optimization problem,

$$\min_{\delta} \frac{1}{M_1} \sum_i \left(\Delta V^{(i)} - \delta^{(i)} \Delta S^{(i)} \right)^2,$$

by imposing a model such that

$$\delta^{(i)} = \delta^{(i)}_{BS} + f_{NN}(x_i) + \epsilon$$

where $\delta^{(i)}_{BS}$ is the implied Black-Scholes delta, this is the delta of the Black-Scholes model when using the market implied volatility as the volatility parameter, and $f_{NN}(x)$ is a feed-forward neural network output with structure and feature explain later.

It should be noted that a specific option i at time t_1 can be treated differently than the same option i at time t_2, so this implementation is completely time independent and the temporal influence can be fully captured by the Time-to-Maturity (TTM) feature fed in both δ_{BS} and f_{NN}.

$$\min_{f_{NN}} \frac{1}{M_1} \sum_i \left(\Delta V^{(i)} - \left(\delta^{(i)}_{BS} + f_{NN}(x_i) \right) \Delta S^{(i)} \right)^2$$

With some algebraic manipulation, the objective function can be adjusted to

$$\min_{f_{NN}} \frac{1}{M_1} \sum_i \left(\frac{\Delta V^{(i)}}{\Delta S^{(i)}} - \delta^{(i)}_{BS} - f_{NN}(x_i) \right)^2,$$

which leads us to a MSE loss function with target variable $y_i = \frac{\Delta V^{(i)}}{\Delta S^{(i)}} - \delta^{(i)}_{BS}$.
M_1 is then Number of time-steps $\times$ Number of products if we consider the complete dataset. **(We should shuffle though.)**

2 Neural Network Structure

A Feed-forward Neural Network (FNN), also known as a Multi-Layer Perceptron (MLP), is implemented to learn the residual $f_{NN}(x)$. The structure of an FNN with two hidden layers is shown in Figure 1.

Let $\mathbf{X}^{\lfloor 0 \rfloor} \in \mathbb{R}^{m\times d}$ be the vector of input features, where m represents the batch size and d represents the number of input features, and $\mathbf{X}^{\lfloor 3 \rfloor} \in \mathbb{R}^{m\times 1}$ be the final output layer. For a two-hidden-layer FNN, let $\mathbf{X}^{\lfloor 1 \rfloor}$, $\mathbf{X}^{\lfloor 2 \rfloor} \in \mathbb{R}^{m\times h}$, be the hidden layers, where h represents the number of neurons on the hidden layers. All layers are fully connected, with hidden layer 1 weights $\mathbf{W}^{\lfloor 1 \rfloor} \in \mathbb{R}^{d\times h}$ and bias $\mathbf{b}^{\lfloor 1 \rfloor} \in \mathbb{R}^{1\times h}$, hidden layer 2 weights $\mathbf{W}^{\lfloor 2 \rfloor} \in \mathbb{R}^{h\times h}$ and bias $\mathbf{b}^{\lfloor 2 \rfloor} \in \mathbb{R}^{1\times h}$, output layer weights $\mathbf{W}^{\lfloor 3 \rfloor} \in \mathbb{R}^{h\times 1}$ and bias $\mathbf{b}^{\lfloor 3 \rfloor} \in \mathbb{R}$. Using a sigmoid activation function σ, the input-to-output mapping is calculated as follows:

$$\mathbf{X}^{\lfloor l+1 \rfloor} = \sigma\underbrace{\left(\mathbf{X}^{\lfloor l \rfloor}\,\mathbf{W}^{\lfloor l+1 \rfloor} + \mathbf{1}_m\,\mathbf{b}^{\lfloor l+1 \rfloor}\right)}_{\mathbf{Z}^{\lfloor l+1 \rfloor}}, \text{ for } l = \{0, 1\}$$

$$f_{NN} = \mathbf{X}^{\lfloor 3 \rfloor} = \mathbf{X}^{\lfloor 2 \rfloor}\,\mathbf{W}^{\lfloor 3 \rfloor} + \mathbf{1}_m\,\mathbf{b}^{\lfloor 3 \rfloor}\,.$$

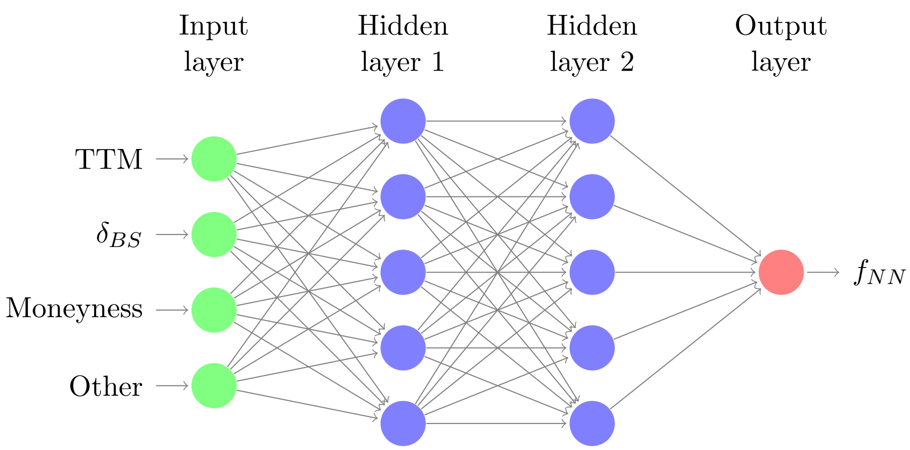

Figure 1: Structure of FNN

3 Back Propagation

Let consider the MSE loss function

$$\mathcal{L} = \frac{1}{2m}\sum_i \left[y_i - f_{NN}(x_i)\right]^2 = \frac{1}{2m}(\mathbf{Y} - \mathbf{X}^{\lfloor 3 \rfloor})^\top(\mathbf{Y} - \mathbf{X}^{\lfloor 3 \rfloor}).$$

If we also consider $g^{\lfloor l \rfloor}$ as the activation function of the layer l and $\odot$ as an element-wise matrix multiplication, then we can describe the backpropagation process as the algorithm in Figure 2. In this particular case $g^{\lfloor 3 \rfloor}$ is just a the identity.
Formally, the gradients of the different layers with respect to the loss function are as follows:

1. Output Layer:
   - $\dfrac{\partial \mathcal{L}}{\partial \mathbf{X}^{\lfloor 3 \rfloor}} = \delta^{\lfloor 3 \rfloor} = \dfrac{-1}{m}(\mathbf{Y} - \mathbf{X}^{\lfloor 3 \rfloor})$
   - $\dfrac{\partial \mathcal{L}}{\partial \mathbf{W}^{\lfloor 3 \rfloor}} = \dfrac{\partial \mathcal{L}}{\partial \mathbf{X}^{\lfloor 3 \rfloor}}\cdot\dfrac{\partial \mathbf{X}^{\lfloor 3 \rfloor}}{\partial \mathbf{W}^{\lfloor 3 \rfloor}} = (\mathbf{X}^{\lfloor 2 \rfloor})^\top\cdot\delta^{\lfloor 3 \rfloor}$

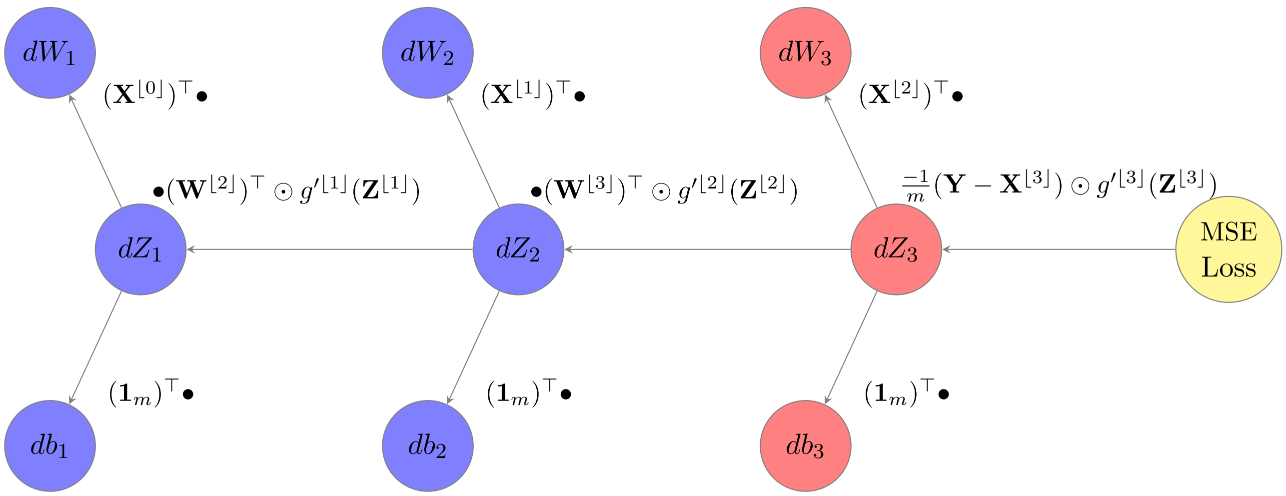

Figure 2: Backpropagation in a MLP

- $\dfrac{\partial \mathcal{L}}{\partial \mathbf{b}^{\lfloor 3 \rfloor}} = \dfrac{\partial \mathcal{L}}{\partial \mathbf{X}^{\lfloor 3 \rfloor}} \cdot \dfrac{\partial \mathbf{X}^{\lfloor 3 \rfloor}}{\partial \mathbf{b}^{\lfloor 3 \rfloor}} = \mathbf{1}_m^\top \cdot \delta^{\lfloor 3 \rfloor}$

2. Second Hidden layer:

- $\dfrac{\partial \mathcal{L}}{\partial \mathbf{Z}^{\lfloor 2 \rfloor}} = \delta^{\lfloor 2 \rfloor} = \delta^{\lfloor 3 \rfloor} \cdot \dfrac{\partial \mathbf{X}^{\lfloor 3 \rfloor}}{\partial \mathbf{X}^{\lfloor 2 \rfloor}} \cdot \dfrac{\partial \mathbf{X}^{\lfloor 2 \rfloor}}{\partial \mathbf{Z}^{\lfloor 2 \rfloor}} = \delta^{\lfloor 3 \rfloor} \cdot (\mathbf{W}^{\lfloor 3 \rfloor})^\top \odot \sigma'(\mathbf{Z}^{\lfloor 2 \rfloor}) = \delta^{\lfloor 3 \rfloor} \cdot (\mathbf{W}^{\lfloor 3 \rfloor})^\top \odot \mathbf{X}^{\lfloor 2 \rfloor} \odot (1 - \mathbf{X}^{\lfloor 2 \rfloor})$
- $\dfrac{\partial \mathcal{L}}{\partial \mathbf{W}^{\lfloor 2 \rfloor}} = \dfrac{\partial \mathcal{L}}{\partial \mathbf{Z}^{\lfloor 2 \rfloor}} \cdot \dfrac{\partial \mathbf{Z}^{\lfloor 2 \rfloor}}{\partial \mathbf{W}^{\lfloor 2 \rfloor}} = (\mathbf{X}^{\lfloor 1 \rfloor})^\top \cdot \delta^{\lfloor 2 \rfloor}$
- $\dfrac{\partial \mathcal{L}}{\partial \mathbf{b}^{\lfloor 2 \rfloor}} = \dfrac{\partial \mathcal{L}}{\mathbf{Z}^{\lfloor 2 \rfloor}} \cdot \dfrac{\partial \mathbf{Z}^{\lfloor 2 \rfloor}}{\partial \mathbf{b}^{\lfloor 2 \rfloor}} = \mathbf{1}_m^\top \cdot \delta^{\lfloor 2 \rfloor}$

3. First Hidden layer:

- $\dfrac{\partial \mathcal{L}}{\partial \mathbf{Z}^{\lfloor 1 \rfloor}} = \delta^{\lfloor 1 \rfloor} = \delta^{\lfloor 2 \rfloor} \cdot \dfrac{\partial \mathbf{Z}^{\lfloor 2 \rfloor}}{\partial \mathbf{X}^{\lfloor 1 \rfloor}} \cdot \dfrac{\partial \mathbf{X}^{\lfloor 1 \rfloor}}{\partial \mathbf{Z}^{\lfloor 1 \rfloor}} = \delta^{\lfloor 2 \rfloor} \cdot (\mathbf{W}^{\lfloor 2 \rfloor})^\top \odot \sigma'(\mathbf{Z}^{\lfloor 1 \rfloor}) = \delta^{\lfloor 2 \rfloor} \cdot (\mathbf{W}^{\lfloor 2 \rfloor})^\top \odot \mathbf{X}^{\lfloor 1 \rfloor} \odot (1 - \mathbf{X}^{\lfloor 1 \rfloor})$
- $\dfrac{\partial \mathcal{L}}{\partial \mathbf{W}^{\lfloor 1 \rfloor}} = \dfrac{\partial \mathcal{L}}{\partial \mathbf{Z}^{\lfloor 1 \rfloor}} \cdot \dfrac{\partial \mathbf{Z}^{\lfloor 1 \rfloor}}{\partial \mathbf{W}^{\lfloor 1 \rfloor}} = (\mathbf{X}^{\lfloor 0 \rfloor})^\top \cdot \delta^{\lfloor 1 \rfloor}$
- $\dfrac{\partial \mathcal{L}}{\partial \mathbf{b}^{\lfloor 1 \rfloor}} = \dfrac{\partial \mathcal{L}}{\mathbf{Z}^{\lfloor 1 \rfloor}} \cdot \dfrac{\partial \mathbf{Z}^{\lfloor 1 \rfloor}}{\partial \mathbf{b}^{\lfloor 1 \rfloor}} = \mathbf{1}_m^\top \cdot \delta^{\lfloor 1 \rfloor}$