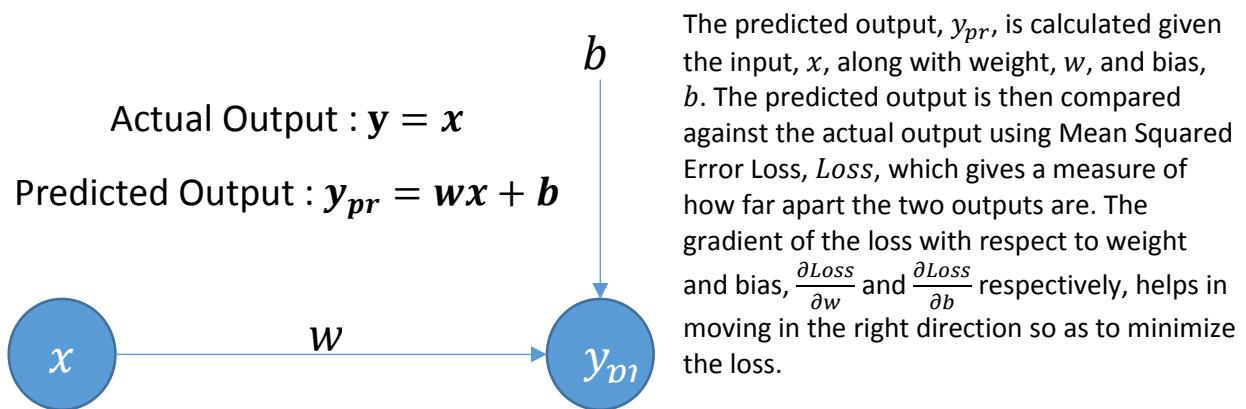


# Realization Of Identity Function Using Two Node Neural Network

A simple function, the Identity Function  $y = x$ , is used to explore Stochastic, Mini-Batch and Batch processing with Gradient Descent(GD), Gradient Descent With Momentum(GDM), RMSProp and Adam optimizer.

## Network



The general flow is defined as below:

Forward Propagation	: $y_{pr} = wx + b$
Loss	: $Loss = \frac{1}{2} * (y_{pr} - y)^2$
Gradients	: $\frac{\partial Loss}{\partial w} = (y_{pr} - y) \frac{\partial y_{pr}}{\partial w} = (y_{pr} - y)x$ : $\frac{\partial Loss}{\partial b} = (y_{pr} - y) \frac{\partial y_{pr}}{\partial b} = (y_{pr} - y)$
Updates	: $w = w - \eta * gradient_w$ : $b = b - \eta * gradient_b$

The data processing method, Stochastic, Mini-Batch or Batch processing, determines how often is the weight and bias updated. The optimizer method, GD, GDM, RMSProp or Adam, determines how to use the gradient information.

## Input Sample Processing

### Stochastic Processing

$$w = w - \eta * \frac{\partial Loss}{\partial w}$$

The weight and bias is updated for every input sample.

$$b = b - \eta * \frac{\partial Loss}{\partial b}$$

### Mini-Batch Processing

The gradient is averaged over M(Mini-Batch Size) samples. The weight and bias is updated after every M samples.

$$\frac{\partial Loss_{avg}}{\partial w} = \frac{1}{M} \sum_{i=1}^M \frac{\partial Loss_i}{\partial w}, w = w - \eta * \frac{\partial Loss_{avg}}{\partial w}$$

$$\frac{\partial Loss_{avg}}{\partial b} = \frac{1}{M} \sum_{i=1}^M \frac{\partial Loss_i}{\partial b}, b = b - \eta * \frac{\partial Loss_{avg}}{\partial b}$$

### Batch Processing

$$\frac{\partial Loss_{avg}}{\partial w} = \frac{1}{N} \sum_{i=1}^N \frac{\partial Loss_i}{\partial w}, w = w - \eta * \frac{\partial Loss_{avg}}{\partial w}$$

$$\frac{\partial Loss_{avg}}{\partial b} = \frac{1}{N} \sum_{i=1}^N \frac{\partial Loss_i}{\partial b}, b = b - \eta * \frac{\partial Loss_{avg}}{\partial b}$$

The gradient is averaged over all the N samples. The weight and bias is updated after N samples.

## Optimizer

$\frac{\partial L}{\partial w}$  and  $\frac{\partial L}{\partial b}$  will be used in the below equations and they can be either Stochastic, Mini-Batch or Batch processing output.

### Gradient Descent

$$\begin{aligned} w &= w - \eta * \frac{\partial L}{\partial w} \\ b &= b - \eta * \frac{\partial L}{\partial b} \end{aligned}$$

The gradient is used as it is to update the weight and bias.

### Gradient Descent With Momentum

The exponential moving average of the gradient is used to update the weight and bias.

$$\begin{aligned} A_w &= \beta A_w + (1 - \beta) \frac{\partial L}{\partial w}, w = w - \eta * A_w \\ A_b &= \beta A_b + (1 - \beta) \frac{\partial L}{\partial b}, b = b - \eta * A_b \end{aligned}$$

### Root Mean Square Prop

$$\begin{aligned} S_w &= \beta S_w + (1 - \beta) \frac{\partial L^2}{\partial w}, w = w - \eta \frac{1}{\sqrt{S_w + \epsilon}} \frac{\partial L}{\partial w} \\ S_b &= \beta S_b + (1 - \beta) \frac{\partial L^2}{\partial b}, b = b - \eta \frac{1}{\sqrt{S_b + \epsilon}} \frac{\partial L}{\partial b} \end{aligned}$$

The exponential moving average of the squared gradient is used to update the weight and bias.

### Adam

The exponential moving average of both the gradient and squared gradient is used to update the weight and bias.

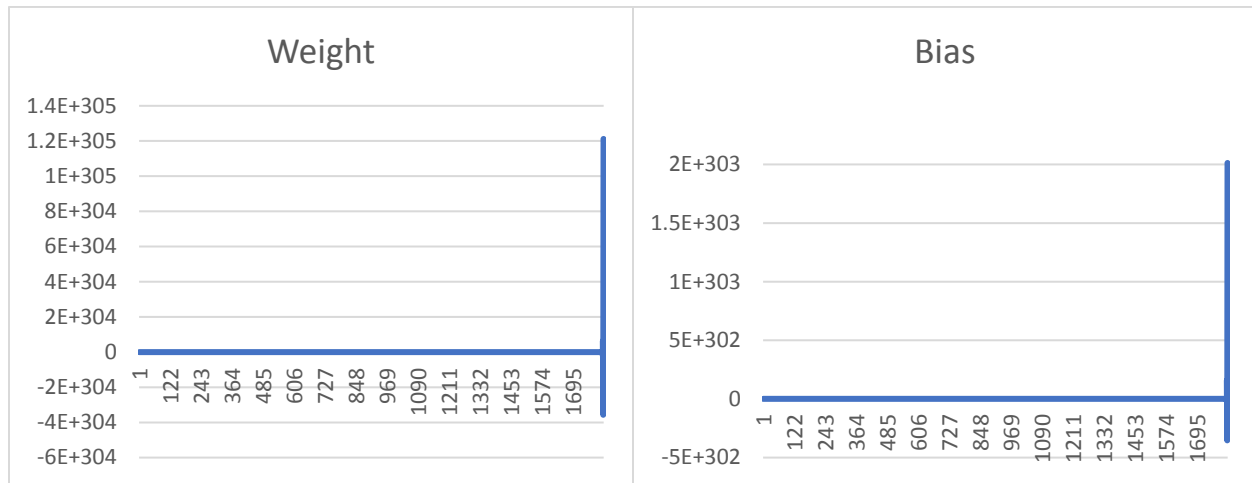
$$\begin{aligned} A_w &= \beta_1 A_w + (1 - \beta_1) \frac{\partial L}{\partial w}, S_w = \beta_2 S_w + (1 - \beta_2) \frac{\partial L^2}{\partial w}, \\ w &= w - \eta \frac{A_w}{\sqrt{S_w + \epsilon}} \\ A_b &= \beta_1 A_b + (1 - \beta_1) \frac{\partial L}{\partial b}, S_b = \beta_2 S_b + (1 - \beta_2) \frac{\partial L^2}{\partial b}, \\ b &= b - \eta \frac{A_b}{\sqrt{S_b + \epsilon}} \end{aligned}$$

## Performance

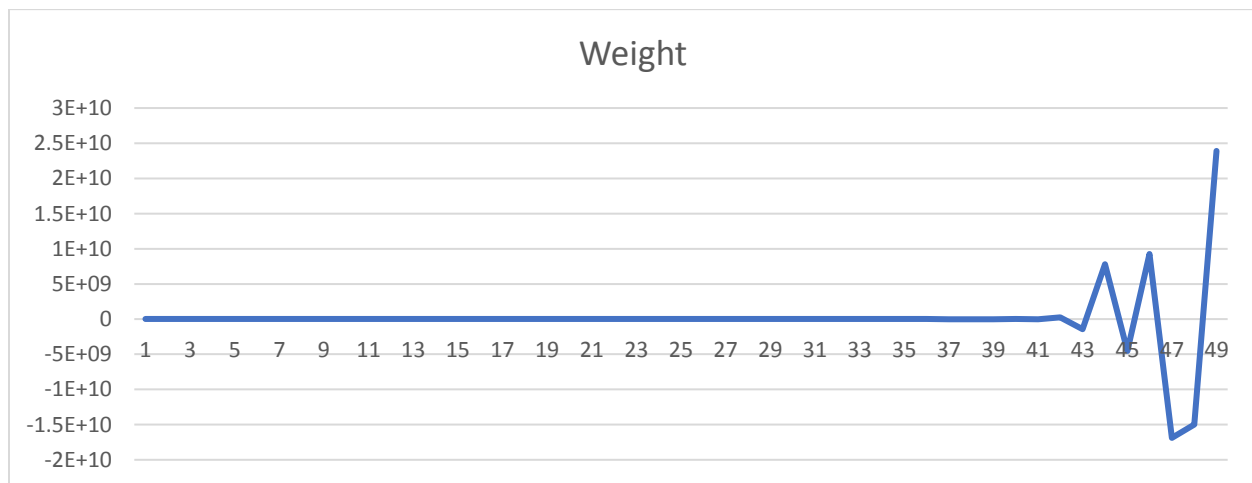
As this is an Identity Function, we know that the weight should be 1 and bias 0. The training is stopped when both the weight and bias reach within  $10E-4$  of the expected value.

### Stochastic Processing with Gradient Descent

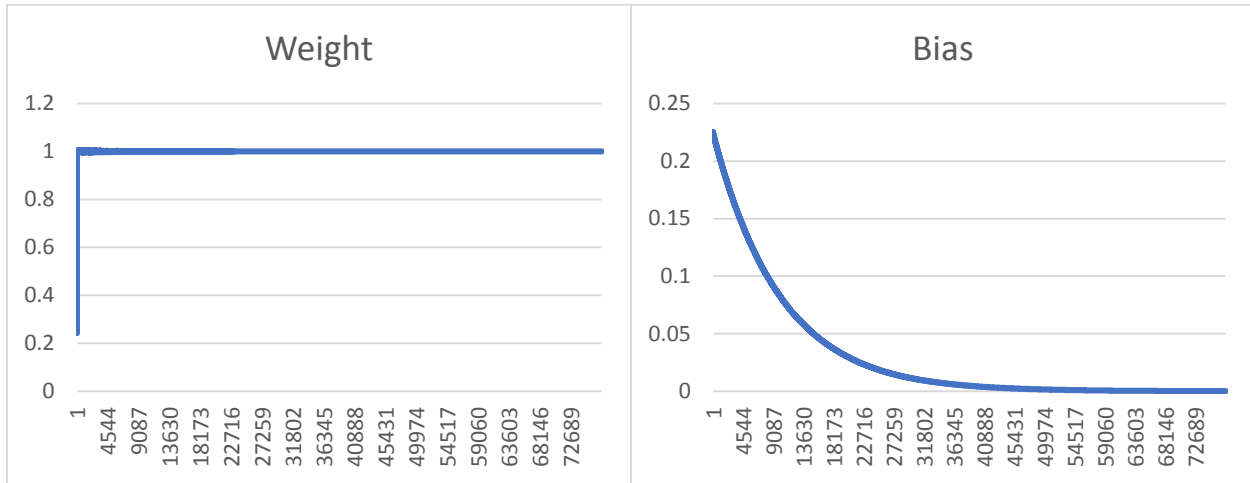
With learning rate,  $\eta$ , set to 0.001, the weight and bias explodes (obtain high magnitude) immediately thereby making the network useless.



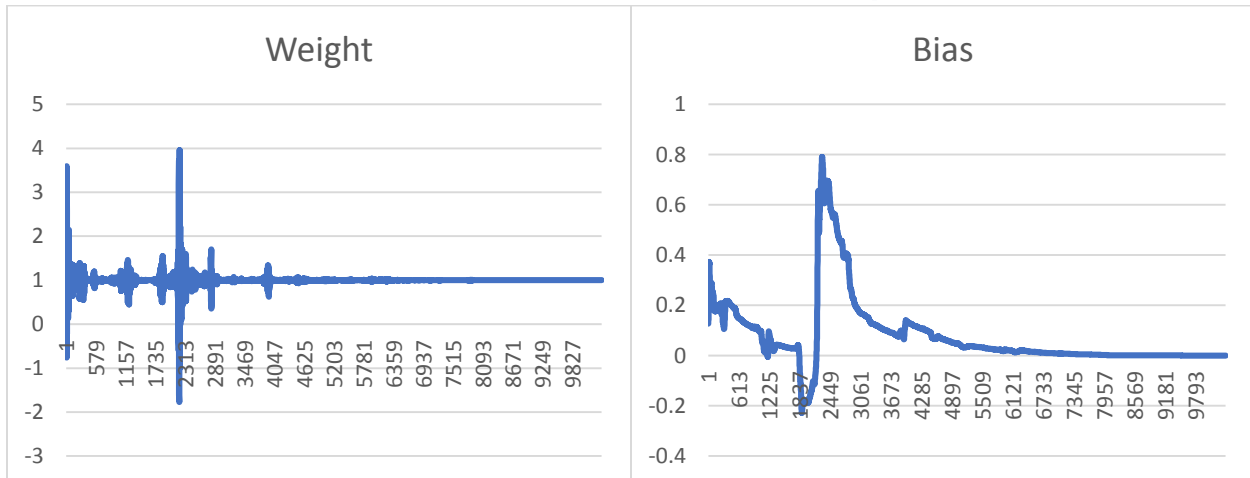
Within 1815 samples, the weight and bias approach infinity. Below graph shows the high oscillation in weight within just ~50 samples.



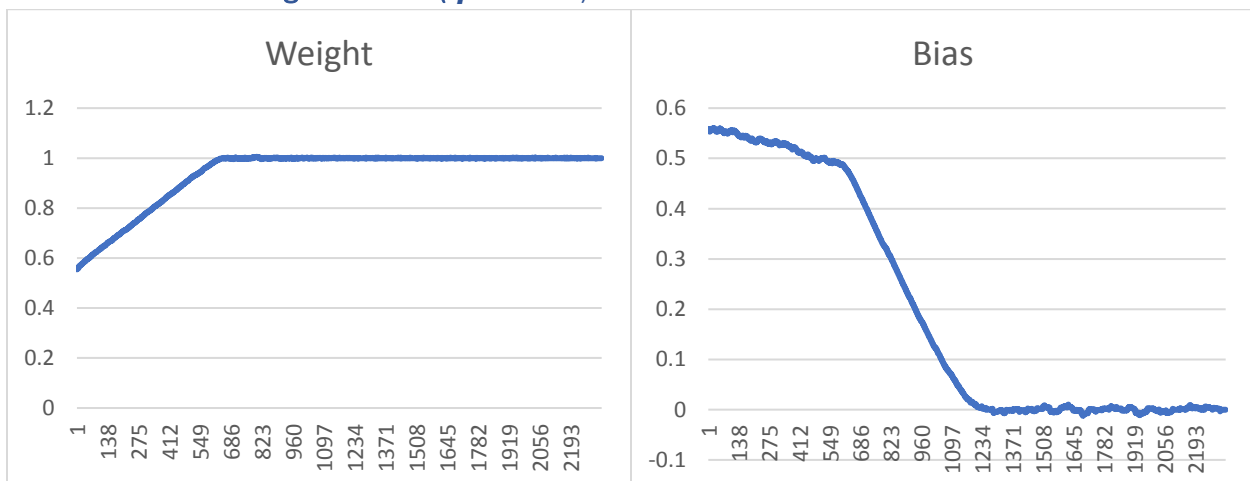
Lowering  $\eta$  to 0.0001 prevents the weights from exploding and we get a usable network.



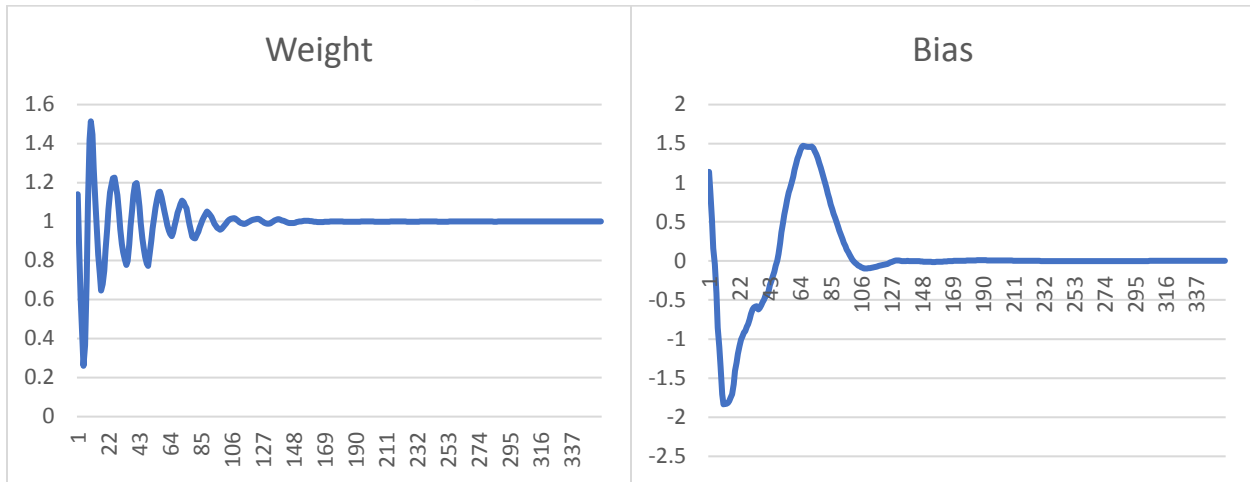
Stochastic Processing with Gradient Descent with Momentum( $\eta = 0.0001$ )



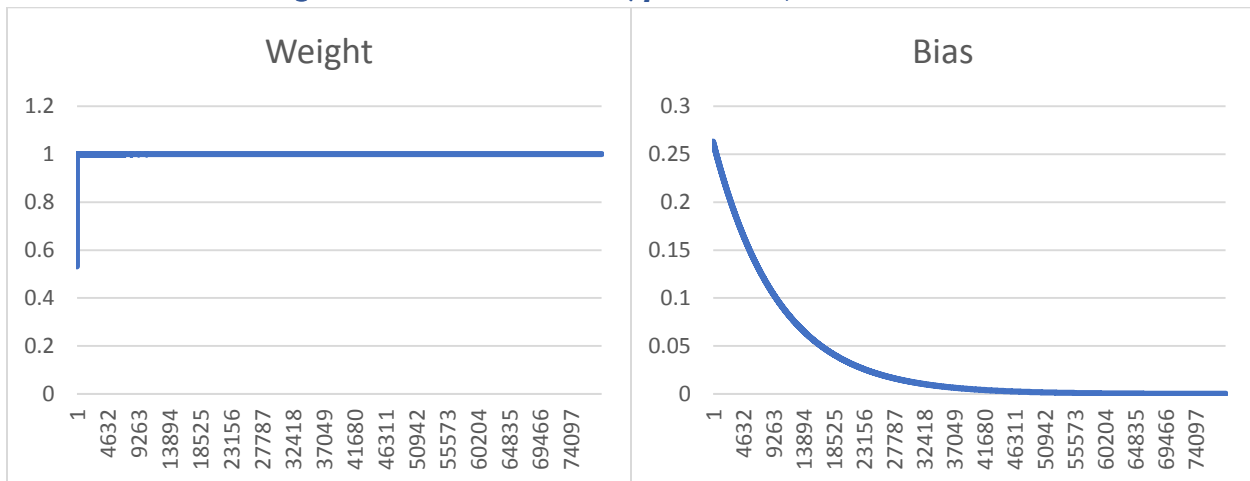
Stochastic Processing with RMS( $\eta = 0.001$ )



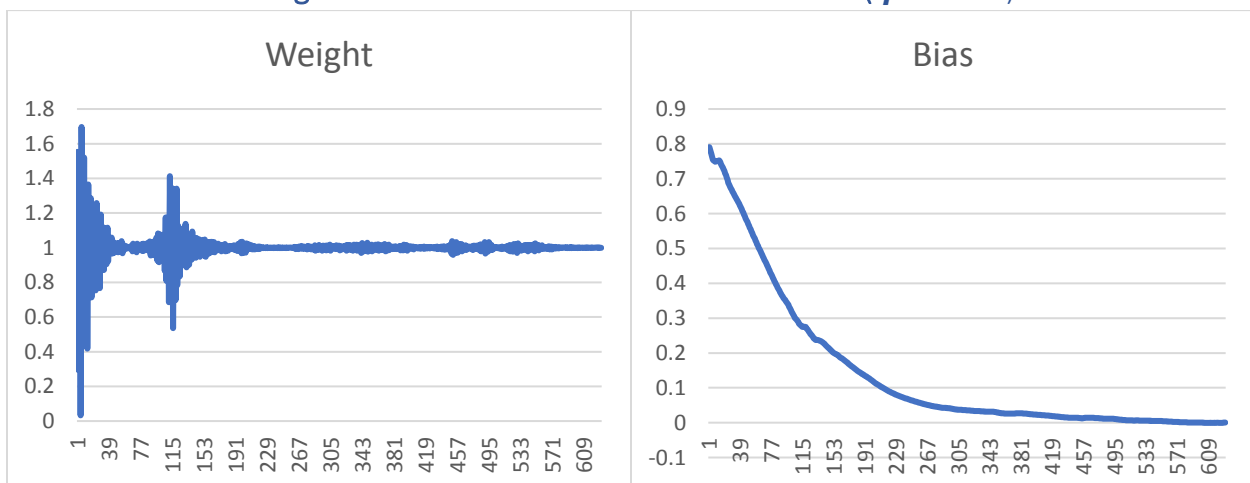
### Stochastic Processing with Adam( $\eta = 0.1$ )



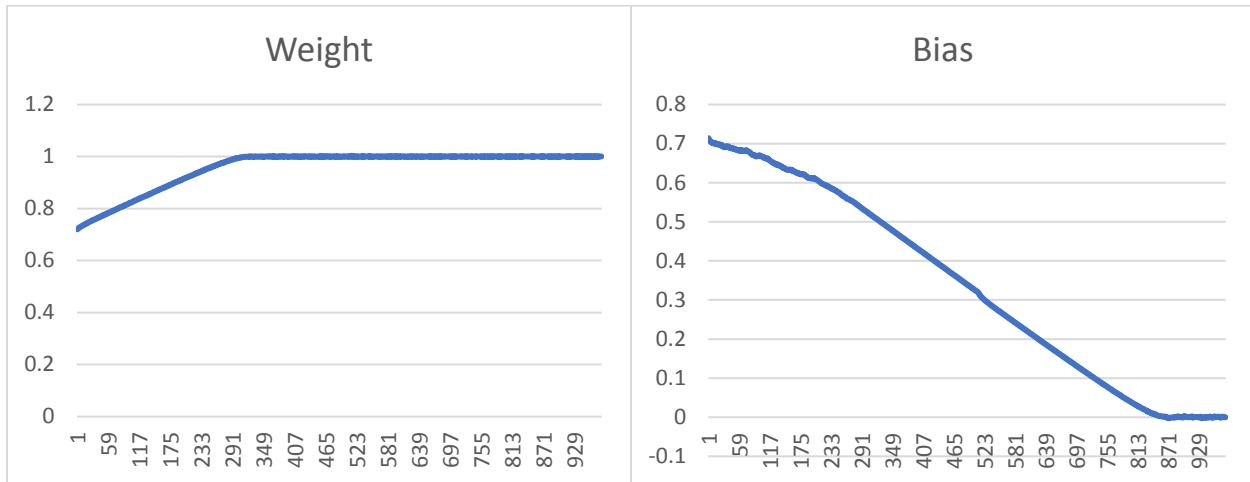
### Mini-Batch Processing with Gradient Descent( $\eta = 0.0001$ )



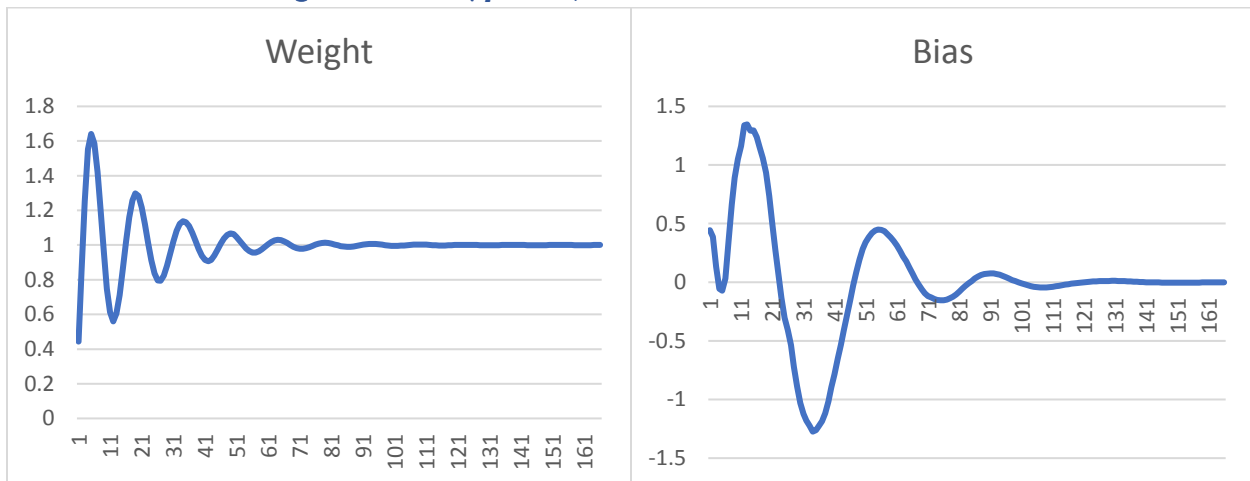
### Mini-Batch Processing with Gradient Descent with Momentum( $\eta = 0.001$ )



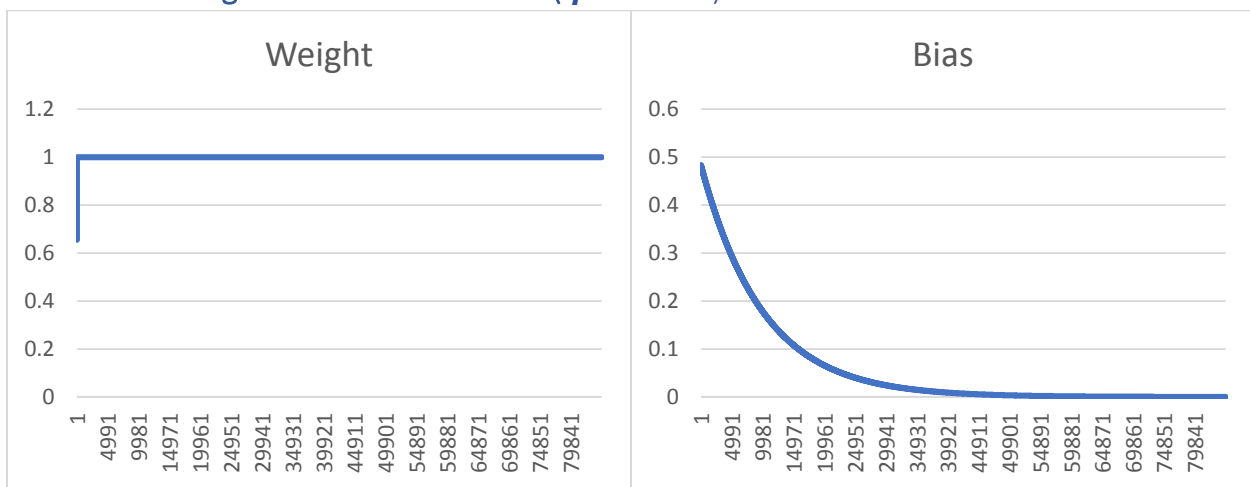
### Mini-Batch Processing with RMS( $\eta = 0.001$ )



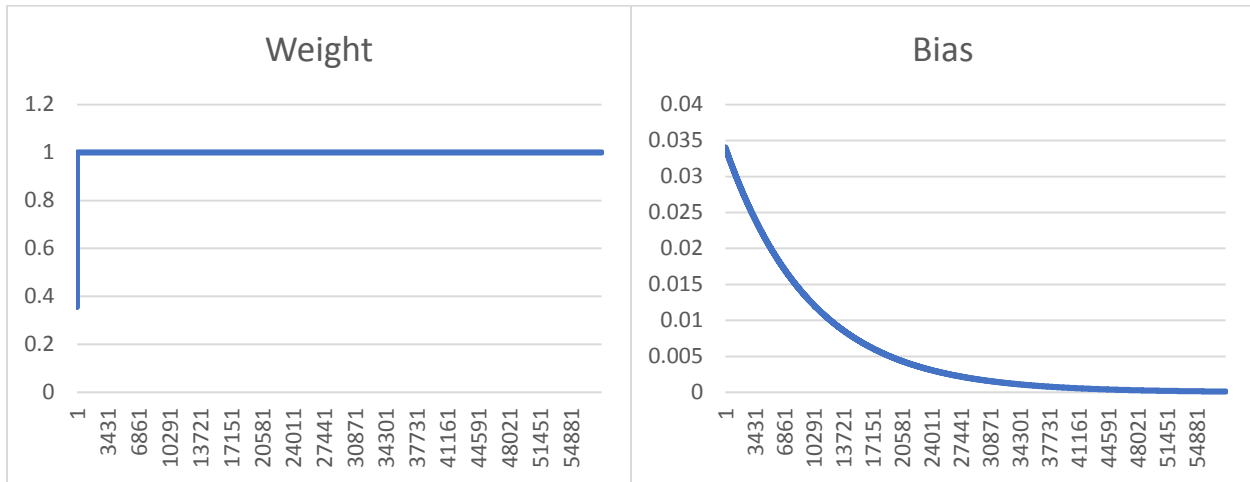
### Mini-Batch Processing with Adam( $\eta = 0.1$ )



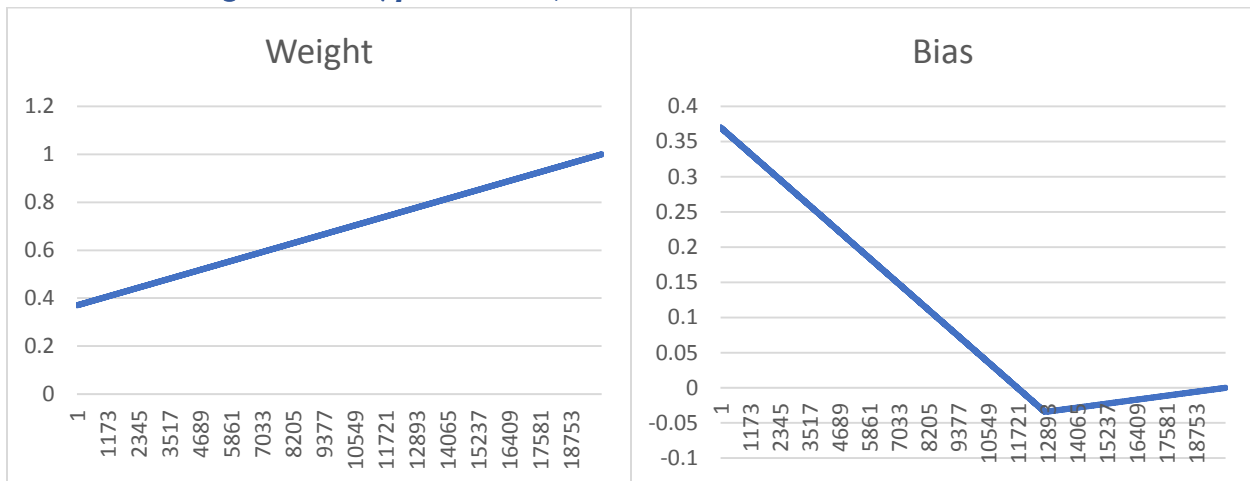
### Batch Processing with Gradient Descent( $\eta = 0.0001$ )



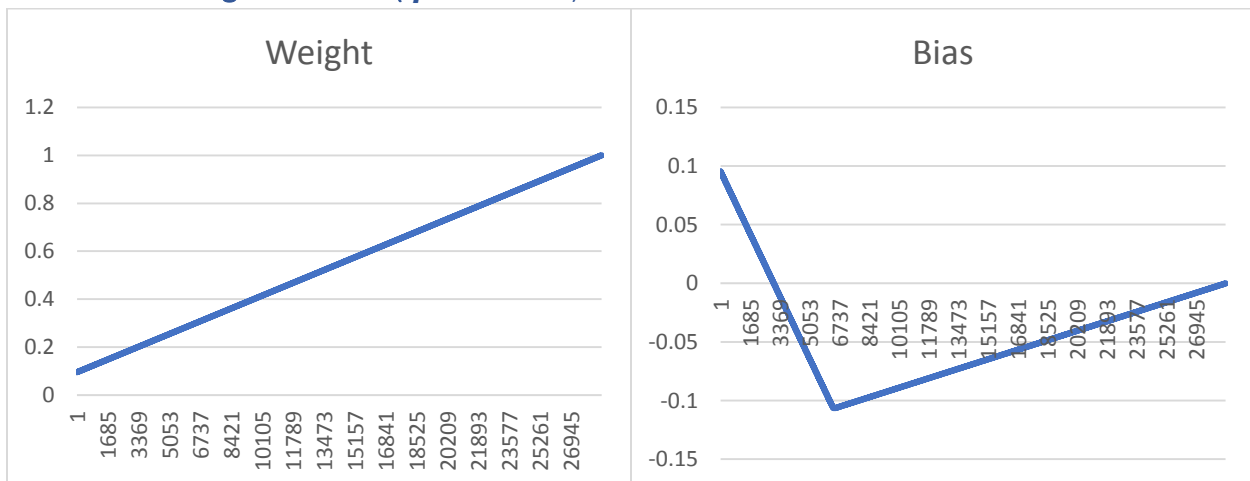
### Batch Processing with Gradient Descent with Momentum( $\eta = 0.0001$ )



### Batch Processing with RMS( $\eta = 0.00001$ )



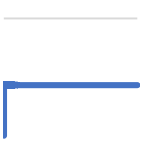
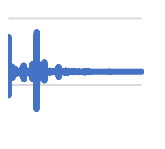


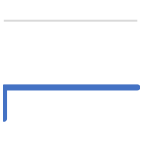







### Batch Processing with Adam( $\eta = 0.00001$ )



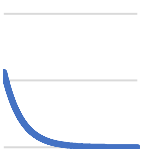



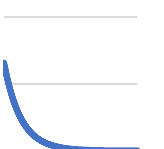
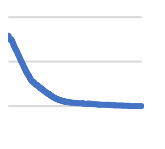
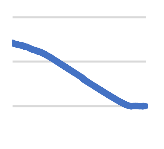

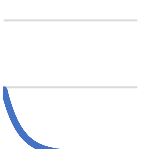
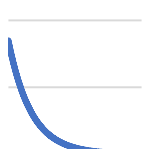

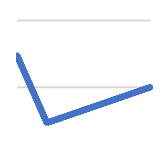


## Comparative Graph

Below table shows the weight profile under different methods:

Weight	GD	GDM	RMSProp	Adam
Stochastic				
Mini-Batch				
Batch				

Below table shows the bias profile under different methods:

Bias	GD	GDM	RMSProp	Adam
Stochastic				
Mini-Batch				
Batch				

Below table shows the maximum learning rate used (above which the weights explode or there is no learning) to achieve the required accuracy under different methods:

Weight	GD	GDM	RMSProp	Adam
Stochastic	0.0001	0.0001	0.001	0.1
Mini-Batch	0.0001	0.001	0.001	0.1
Batch	0.0001	0.0001	0.00001	0.00001

One can observe that the oscillations reduce when moving from Stochastic to Batch. This is because the weights are updated frequently, after every sample, in Stochastic processing whereas the weights are updated only once for the entire batch of data in Batch processing. Also the amount by which the weights are updated in Batch Processing is very small due to the need of very small learning rate.

## Timing

Training time of the model, for weight and bias to reach an accuracy of 0.0001.

### Absolute Value

Training Time(s)	GD	GDM	RMSProp	Adam
Stochastic	1.2531726	0.1571339	0.0252543	0.0081627
Mini-Batch	1.4777563	0.0142058	0.0204918	0.0049806
Batch	180.1724363	157.7865816	50.1907914	48.8349409

### Relative Value

Training Time	GD	GDM	RMSProp	Adam
Stochastic	252	32	6	2
Mini-Batch	297	3	5	1
Batch	36175	31681	10078	9806

## Conclusion

Mini-Batch Processing with Adam Optimizer is the best model and Batch Processing with traditional Gradient Descent optimizer is the worst model taking 36175 times more time to reach the required weight and bias accuracy.

Model parameters used in Mini-Batch Processing with Adam Optimizer:

Epochs taken	1
Mini-Batch Size, M	64
Learning Rate, $\eta$	0.1
Beta1, $\beta_1$	0.9
Beta2, $\beta_2$	0.999