Title: Machine Learning Lecture

Speakers: Mohamed Hibat Allah

Collection: Machine Learning 2023/24

Date: April 09, 2024 - 11:30 AM

URL: https://pirsa.org/24040048

Lecture 4

Last Lecture:

* Architecture of feed-forward Neural Networks (FFNNs) for supervised learning (SL)
  - → Weights $W_{ij}^{(l)}$, biases b_i^l, activation functions
  - → Expressivity of FFNNs
  - → Basics of NNs training: cost functions

Let's continue our discussion on how NNs learn:

* Outline for today:

↳ More on NN training ("learning")

→ Learning algorithms

→ Backpropagation

↳ Overfitting

* Learning algorithm

We use a learning algorithm (such as gradient descent) to minimize a cost function with respect to all weights and biases in each layer l such that:

① $\frac{\partial C}{\partial W_{ij}^{(l)}} = 0$

② $\frac{\partial C}{\partial b_j^{(l)}} = 0$

$\Rightarrow$ for $\forall i, j, l$

Unlike linear regression, # of parameters (weights & biases) can quickly increase (#P can be $\gg$ dim)

↳ It becomes nearly impossible to solve all equations ① and ② to get W & $\vec{b}$

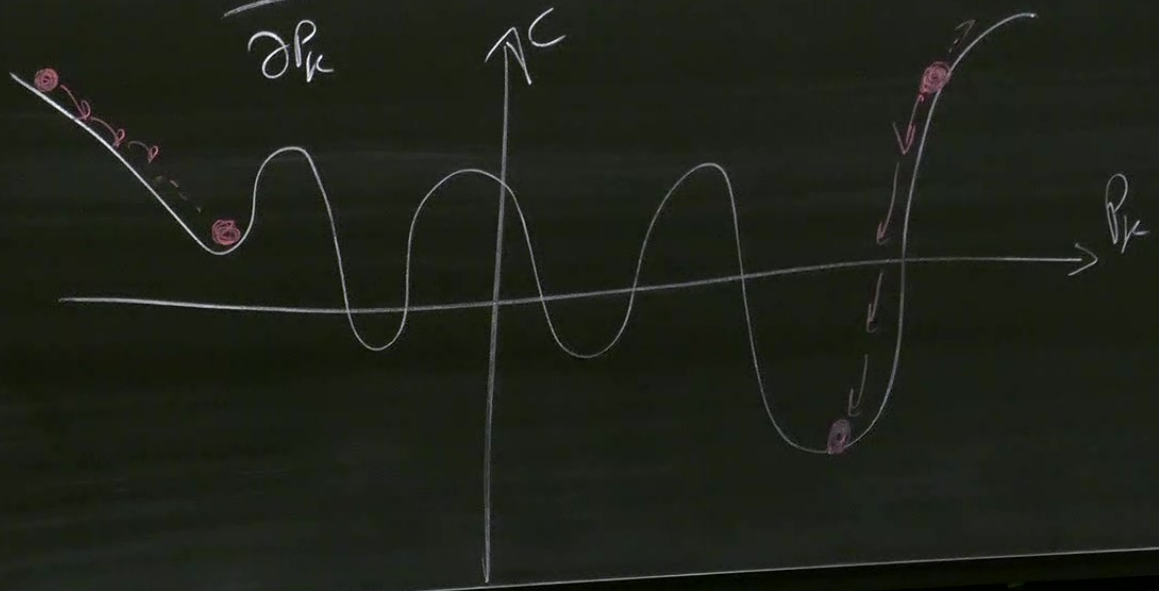
$\vec{p} = (p_1, p_2, \cdots, p_N)$

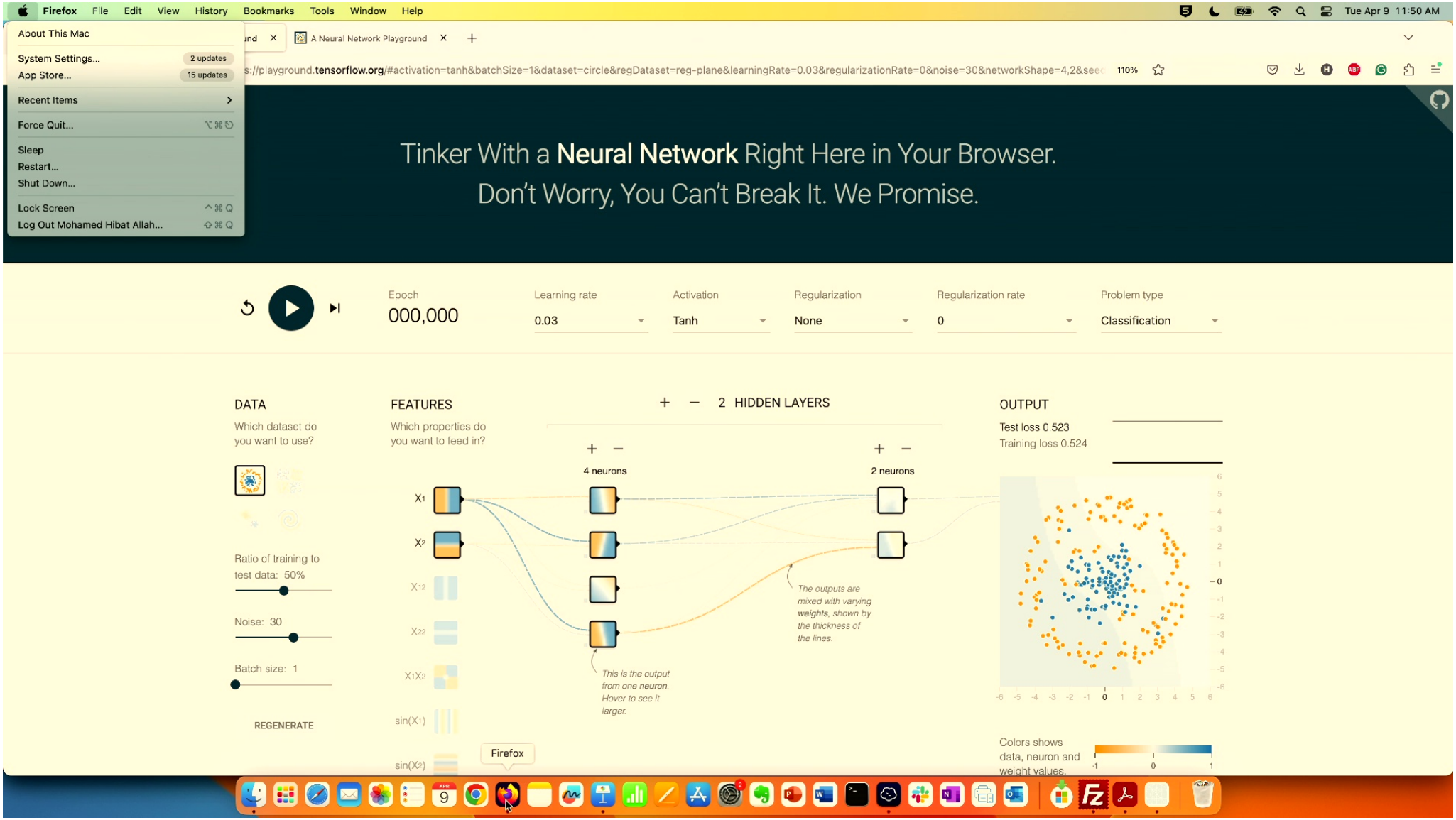
$\frac{\partial C}{\partial p_k} = 0 \quad \forall (1 \leq k \leq N)$

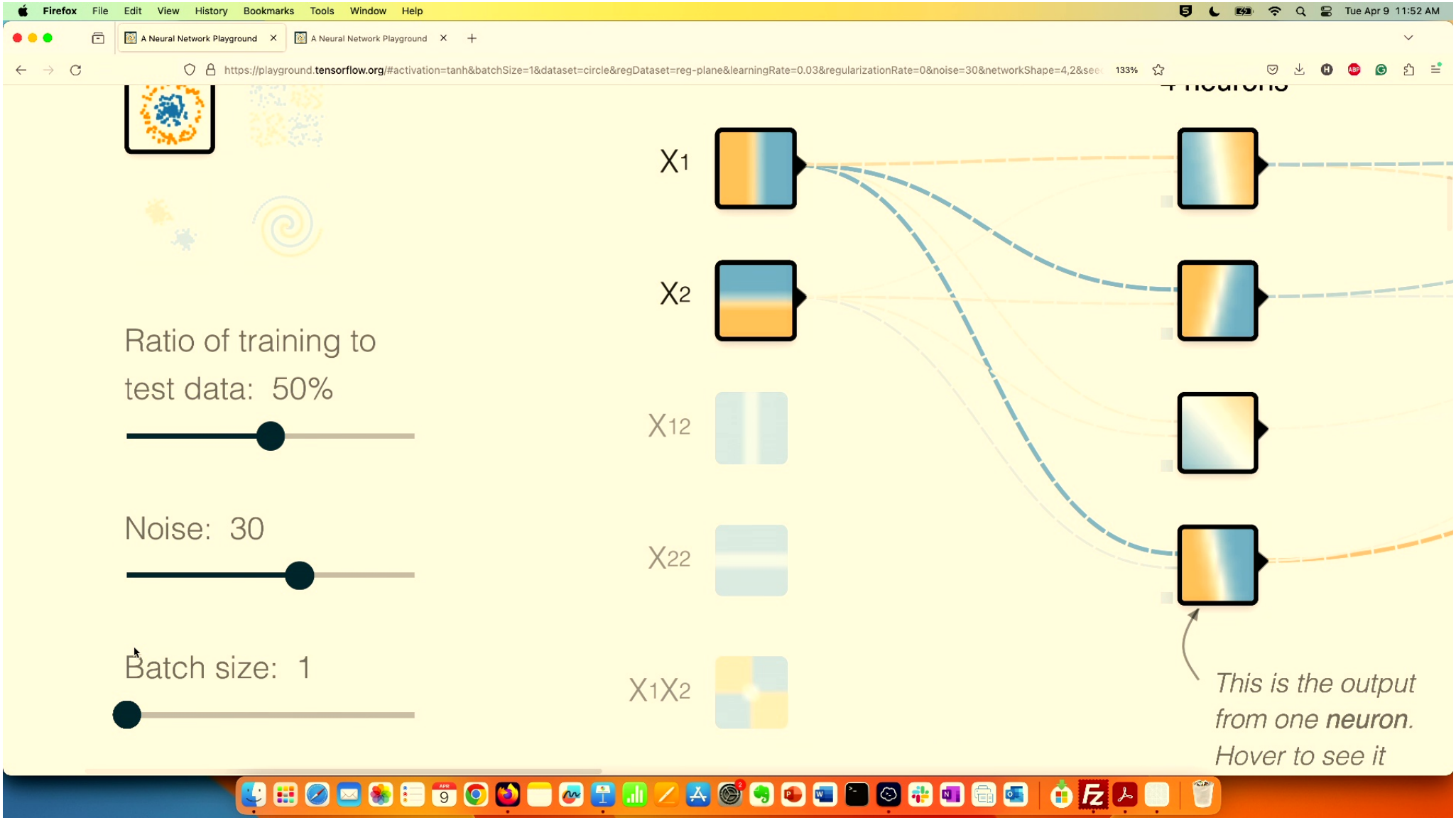
Gradient descent (GD) update.

$p_k \leftarrow p_k - \eta \frac{\partial}{\partial p_k} C$

η: Learning rate.




Firefox File Edit View History Bookmarks Tools Window Help
Tue Apr 9 11:50 AM
About This Mac
System Settings... 2 updates
App Store... 15 updates
Recent Items
Force Quit...
Sleep
Restart...
Shut Down...
Lock Screen
Log Out Mohamed Hibat Allah...
A Neural Network Playground
s://playground.tensorflow.org/#activation=tanh&batchSize=1&dataset=circle&regDataset=reg-plane&learningRate=0.03&regularizationRate=0&noise=30&networkShape=4,2&see
110%
Tinker With a Neural Network Right Here in Your Browser.
Don't Worry, You Can't Break It. We Promise.
Epoch
000,000
Learning rate
0.03
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 50%
Noise: 30
Batch size: 1
REGENERATE
FEATURES
Which properties do you want to feed in?
X1
X2
X12
X22
X1X2
sin(X1)
sin(X2)
2 HIDDEN LAYERS
4 neurons
2 neurons
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Test loss 0.523
Training loss 0.524
Colors shows data, neuron and weight values.
Firefox


Firefox File Edit View History Bookmarks Tools Window Help
Tue Apr 9 11:52 AM
A Neural Network Playground
A Neural Network Playground
https://playground.tensorflow.org/#activation=tanh&batchSize=1&dataset=circle&regDataset=reg-plane&learningRate=0.03&regularizationRate=0&noise=30&networkShape=4,2&see
133%
Ratio of training to test data: 50%
Noise: 30
Batch size: 1
X1
X2
X12
X22
X1X2
This is the output from one neuron. Hover to see it

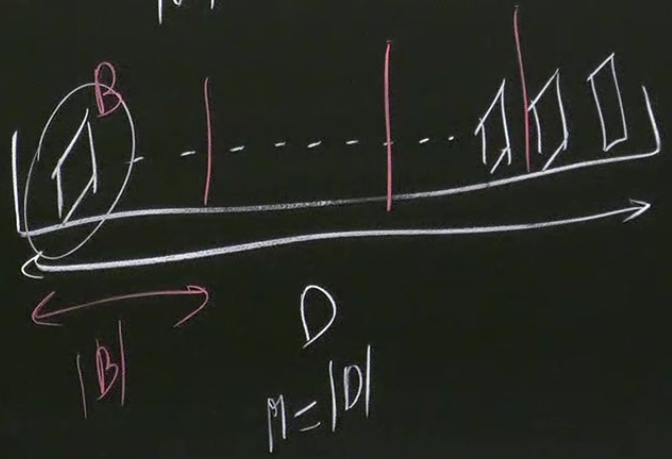
Stochastic Gradient Descent (SGD) & Mini-batch Gradient Descent

$$C = \frac{1}{|D|} \sum_{\vec{x} \in D} C(\vec{x})$$

* SGD $|B| = 1$

$\vec{x}^{(i)} \in D$ — Random order — $i = 1, \dots M$

$$P_k \leftarrow P_k - \eta \partial_k C(\vec{x}^{(i)})$$

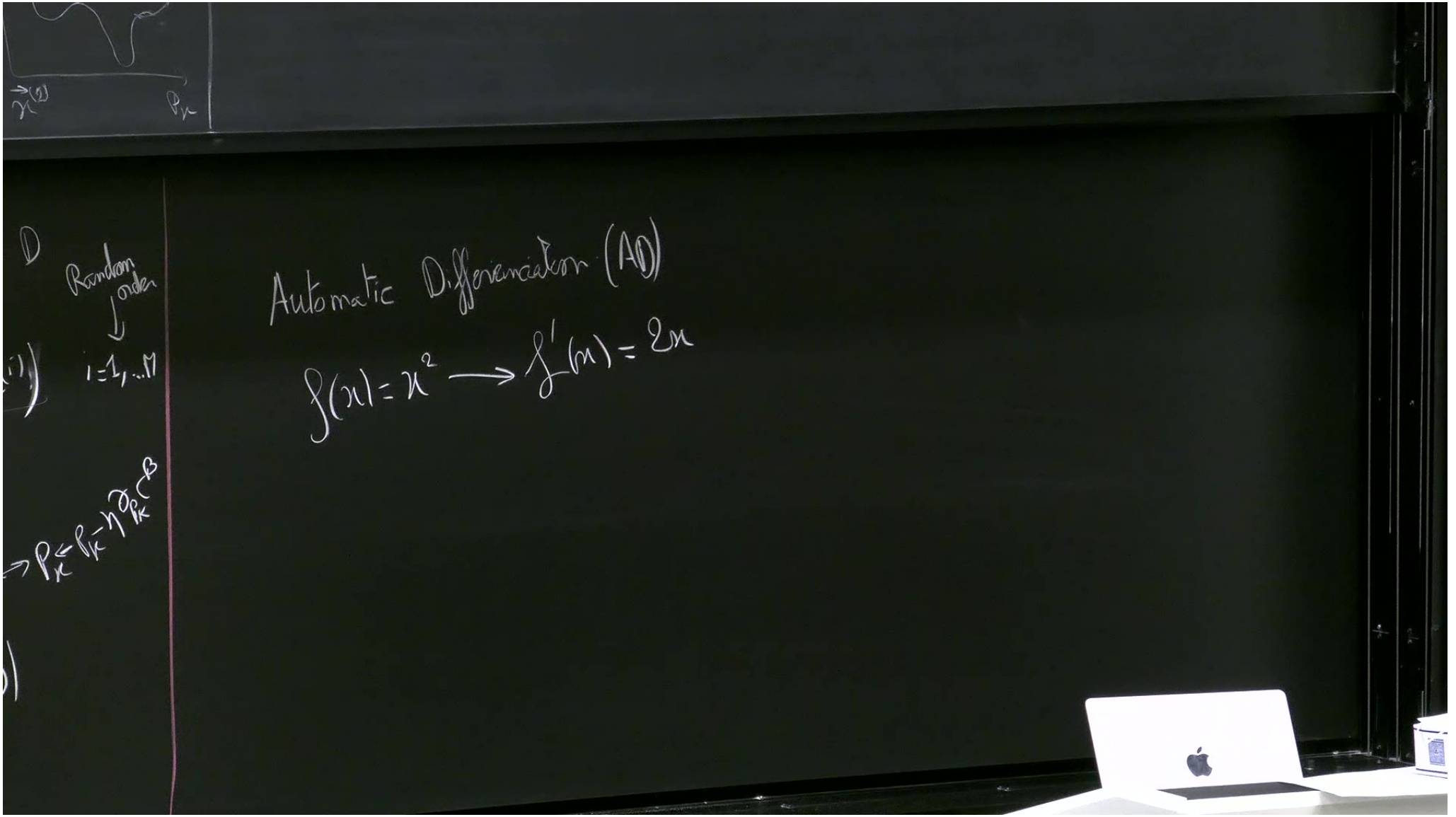


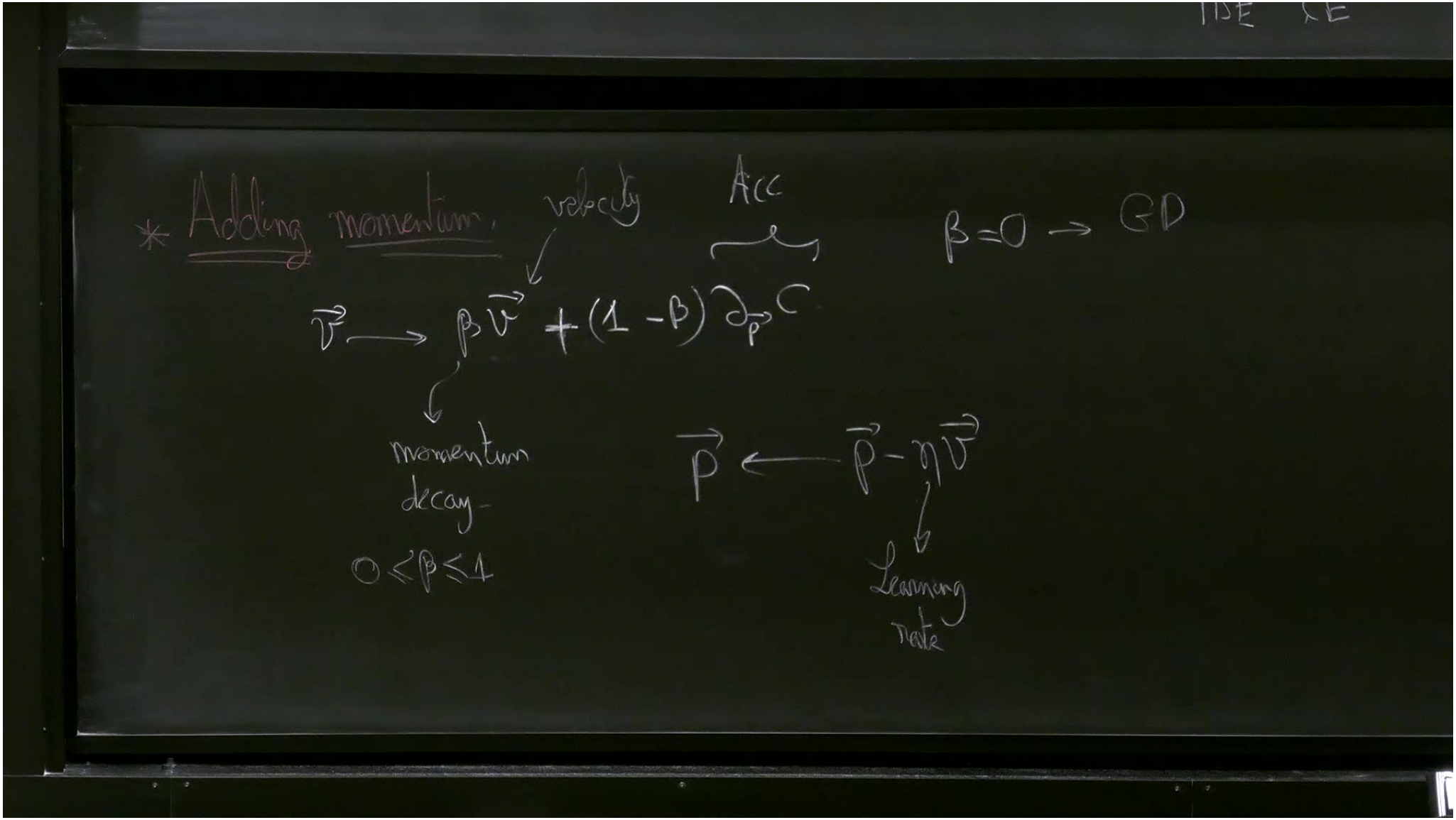
* Mini-batch

$$C^B = \frac{1}{|B|} \sum_{\vec{x} \in B} C(\vec{x}) \rightarrow P_k \leftarrow P_k - \eta \partial_{P_k} C^B$$

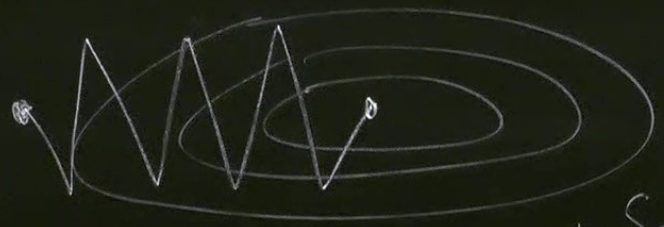
$|B| \equiv O(100)$

$|B| = 2 \rightarrow |D|$


Random order
i=1,...M
Automatic Differentiation (AD)
$f(x) = x^2 \longrightarrow f'(x) = 2x$


* Adding momentum
velocity
Acc
$\beta = 0 \rightarrow GD$
$\vec{v} \longrightarrow \beta \vec{v} + (1-\beta) \nabla_{\vec{p}} C$
momentum decay
$0 < \beta \leq 1$
$\vec{p} \longleftarrow \vec{p} - \eta \vec{v}$
Learning rate

No momentum



Momentum

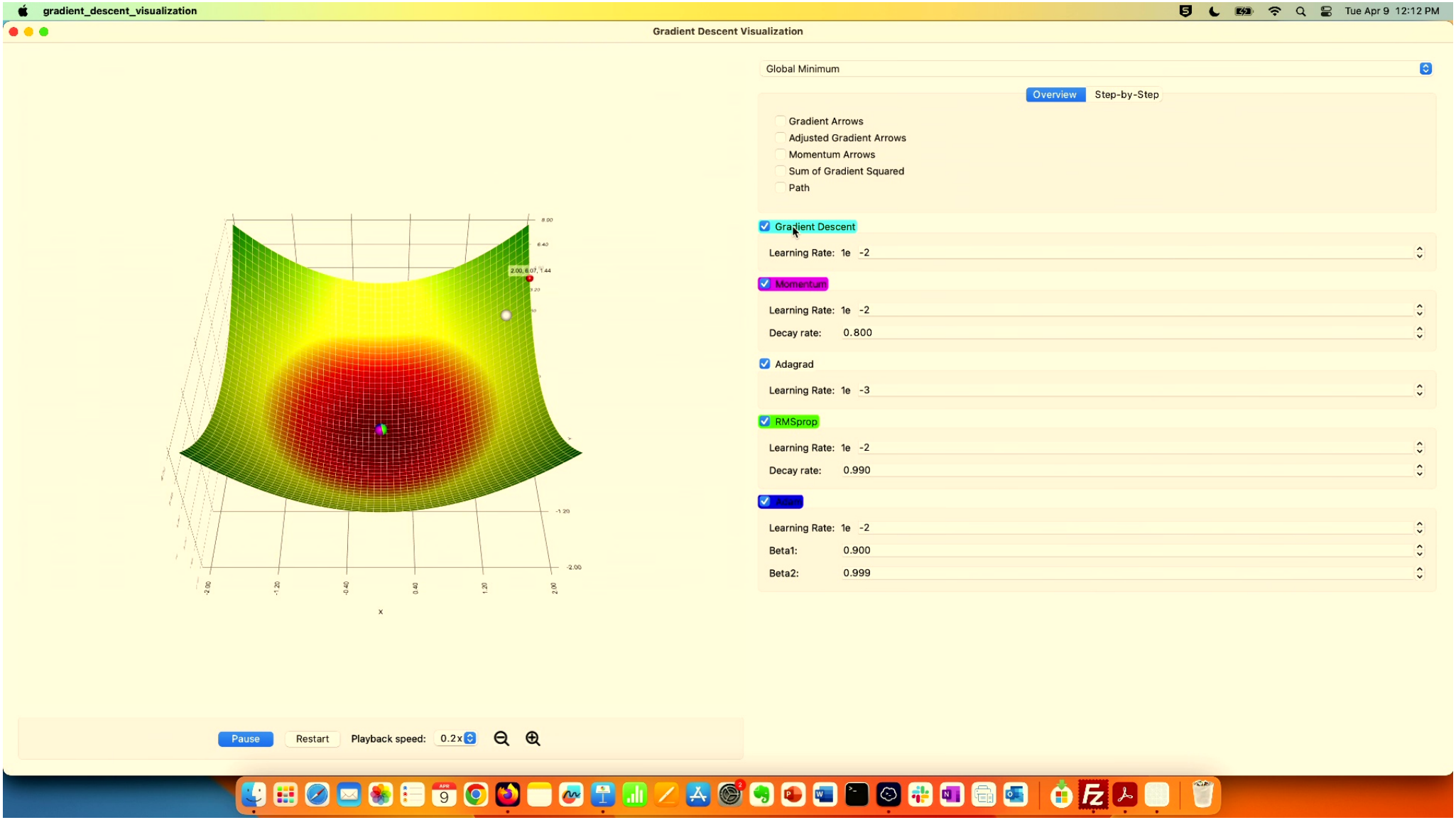
Oscillations

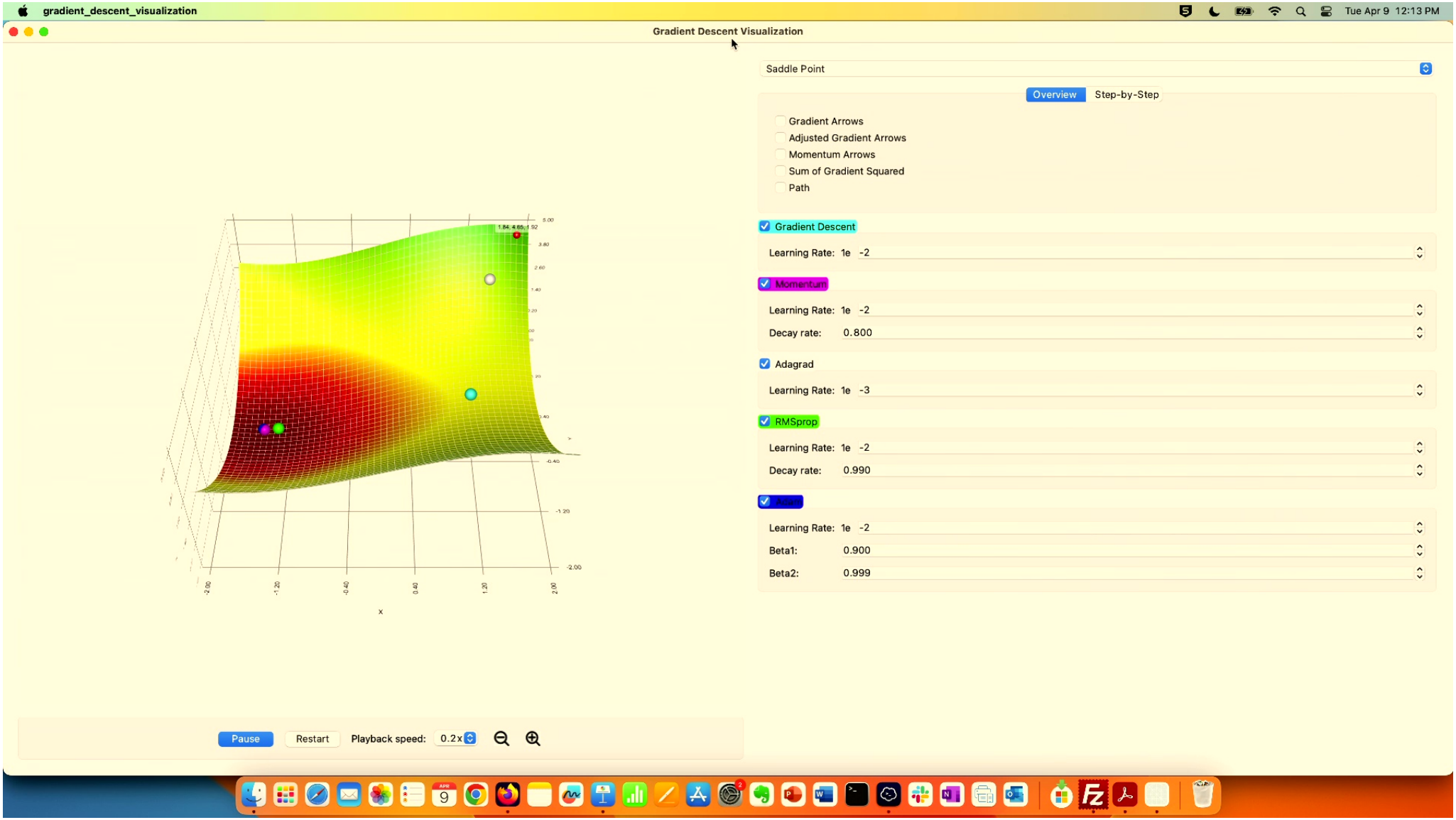
=

Slow learning.

* Second Order optimizers.
* Adaptive learning rate:
  * RMS Prop
  * Adam.

→ Pytorch (torch.optim)


gradient_descent_visualization
Tue Apr 9 12:12 PM
Gradient Descent Visualization
Global Minimum
Overview
Step-by-Step
Gradient Arrows
Adjusted Gradient Arrows
Momentum Arrows
Sum of Gradient Squared
Path
Gradient Descent
Learning Rate: 1e -2
Momentum
Learning Rate: 1e -2
Decay rate: 0.800
Adagrad
Learning Rate: 1e -3
RMSprop
Learning Rate: 1e -2
Decay rate: 0.990
Adam
Learning Rate: 1e -2
Beta1: 0.900
Beta2: 0.999
x
Pause
Restart
Playback speed: 0.2x

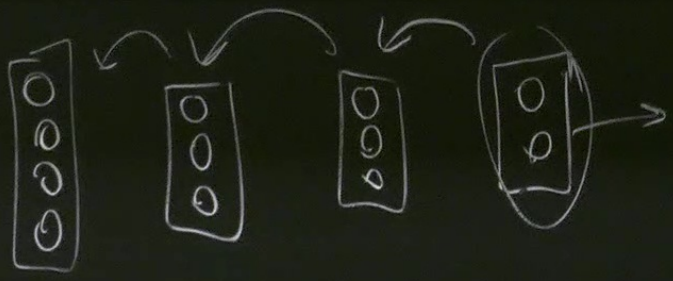

gradient_descent_visualization
Tue Apr 9 12:13 PM
Gradient Descent Visualization
Saddle Point
Overview
Step-by-Step
Gradient Arrows
Adjusted Gradient Arrows
Momentum Arrows
Sum of Gradient Squared
Path
Gradient Descent
Learning Rate: 1e -2
Momentum
Learning Rate: 1e -2
Decay rate: 0.800
Adagrad
Learning Rate: 1e -3
RMSprop
Learning Rate: 1e -2
Decay rate: 0.990
Adam
Learning Rate: 1e -2
Beta1: 0.900
Beta2: 0.999
x
Pause
Restart
Playback speed: 0.2x

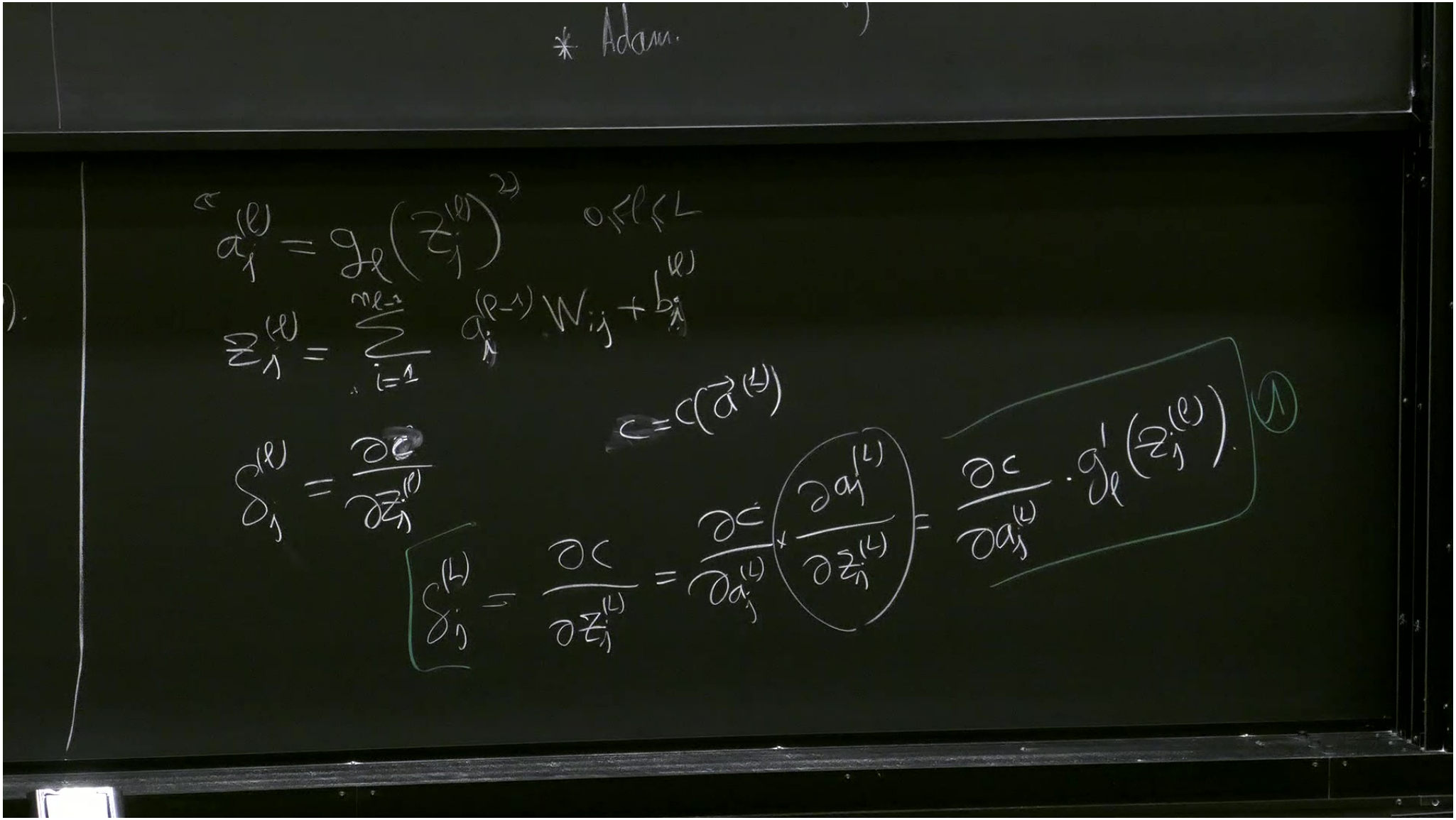
Back propagation in FFNNs

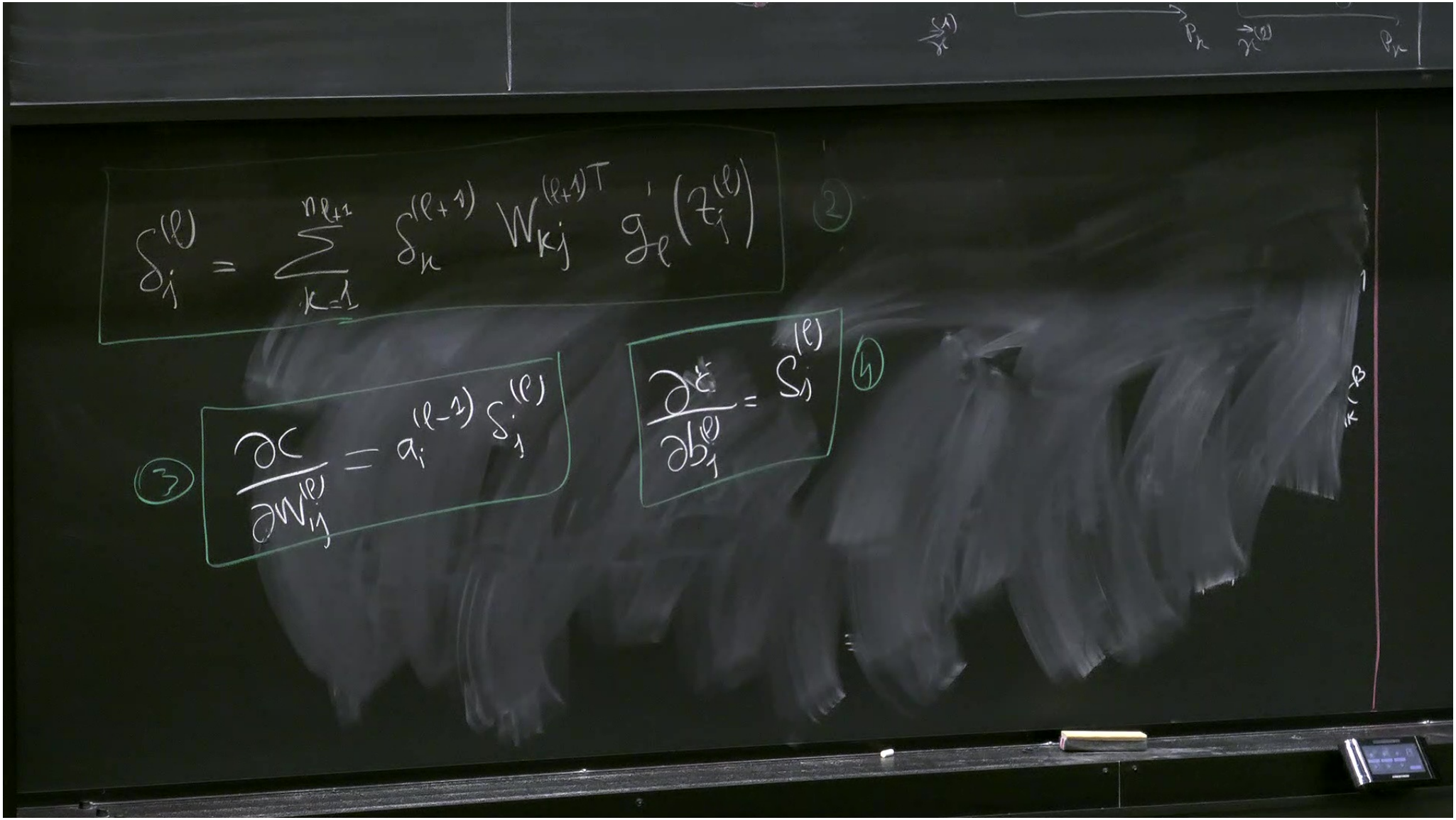
$$\frac{\partial C}{\partial W_{ij}^{(\ell)}}, \frac{\partial C}{\partial b_i^{(\ell)}}$$

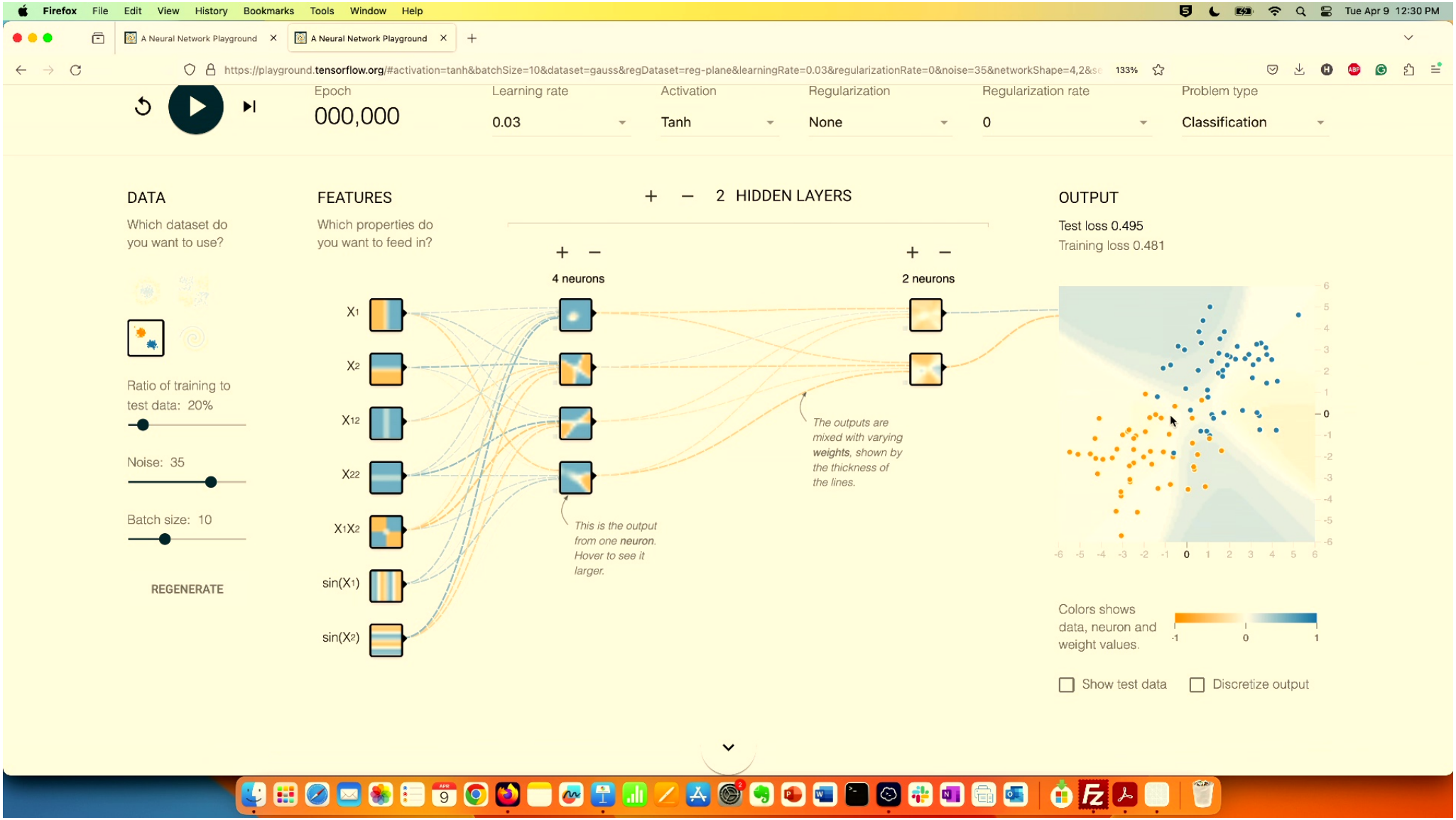
$$C = \frac{1}{|D|} \sum_{x \in D} C(\vec{x})$$

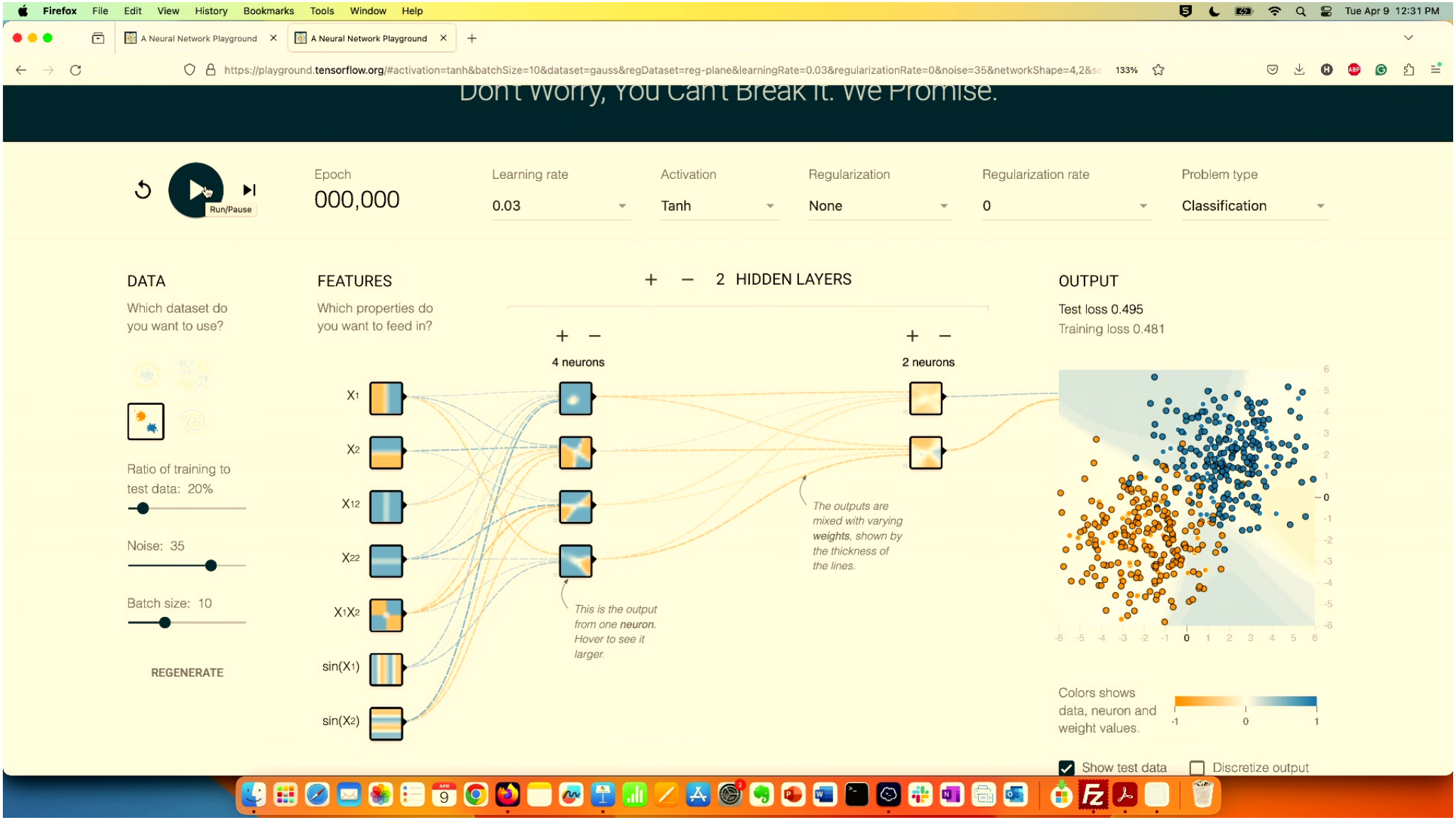
One training sample $\vec{x}$.

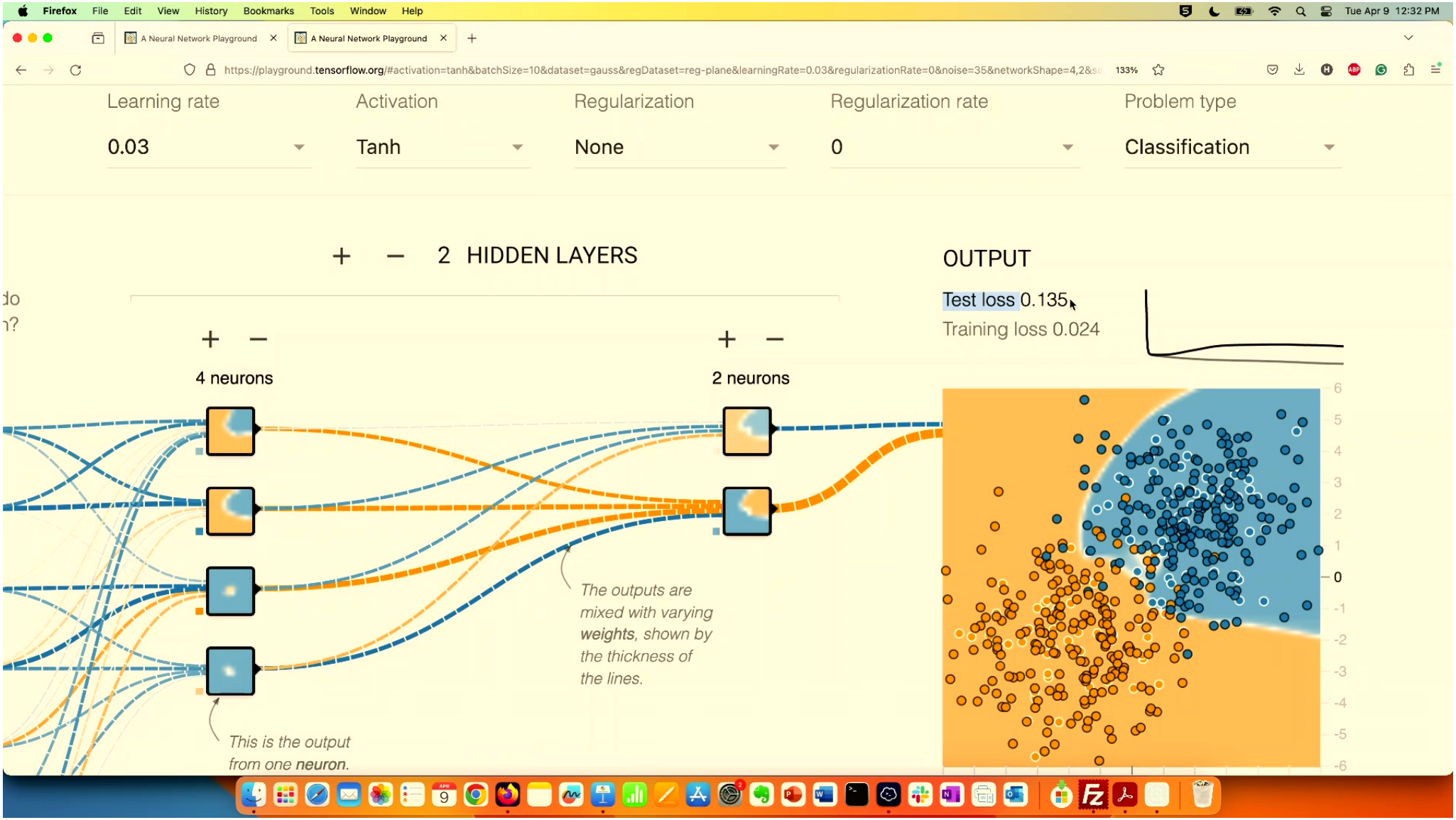



* Adam.
$a_j^{(l)} = g_l(z_j^{(l)})$
$0 < l < L$
$z_j^{(l)} = \sum_{i=1}^{n_{l-1}} a_i^{(l-1)} W_{ij} + b_j^{(l)}$
$C = C(\vec{a}^{(L)})$
$\delta_j^{(l)} = \frac{\partial C}{\partial z_j^{(l)}}$
$\delta_j^{(L)} = \frac{\partial C}{\partial z_j^{(L)}} = \frac{\partial C}{\partial a_j^{(L)}} \times \frac{\partial a_j^{(L)}}{\partial z_j^{(L)}} = \frac{\partial C}{\partial a_j^{(L)}} \cdot g_l'(z_j^{(l)})$
(1)


$\delta_j^{(l)} = \sum_{k=1}^{n_{l+1}} \delta_k^{(l+1)} W_{kj}^{(l+1)T} g_l'(z_j^{(l)})$ (2)
$\frac{\partial C}{\partial W_{ij}^{(l)}} = a_i^{(l-1)} \delta_j^{(l)}$ (3)
$\frac{\partial C}{\partial b_j^{(l)}} = \delta_j^{(l)}$ (4)


Epoch
000,000
Learning rate
0.03
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 20%
Noise: 35
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
2 HIDDEN LAYERS
4 neurons
2 neurons
X1
X2
X12
X22
X1X2
sin(X1)
sin(X2)
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Test loss 0.495
Training loss 0.481
Colors shows data, neuron and weight values.
Show test data
Discretize output


Don't Worry, You Can't Break It. We Promise.
Run/Pause
Epoch
000,000
Learning rate
0.03
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
DATA
Which dataset do you want to use?
Ratio of training to test data: 20%
Noise: 35
Batch size: 10
REGENERATE
FEATURES
Which properties do you want to feed in?
X1
X2
X12
X22
X1X2
sin(X1)
sin(X2)
2 HIDDEN LAYERS
4 neurons
2 neurons
This is the output from one neuron. Hover to see it larger.
The outputs are mixed with varying weights, shown by the thickness of the lines.
OUTPUT
Test loss 0.495
Training loss 0.481
Colors shows data, neuron and weight values.
Show test data
Discretize output


Learning rate
0.03
Activation
Tanh
Regularization
None
Regularization rate
0
Problem type
Classification
2 HIDDEN LAYERS
4 neurons
2 neurons
OUTPUT
Test loss 0.135
Training loss 0.024
The outputs are mixed with varying weights, shown by the thickness of the lines.
This is the output from one neuron.