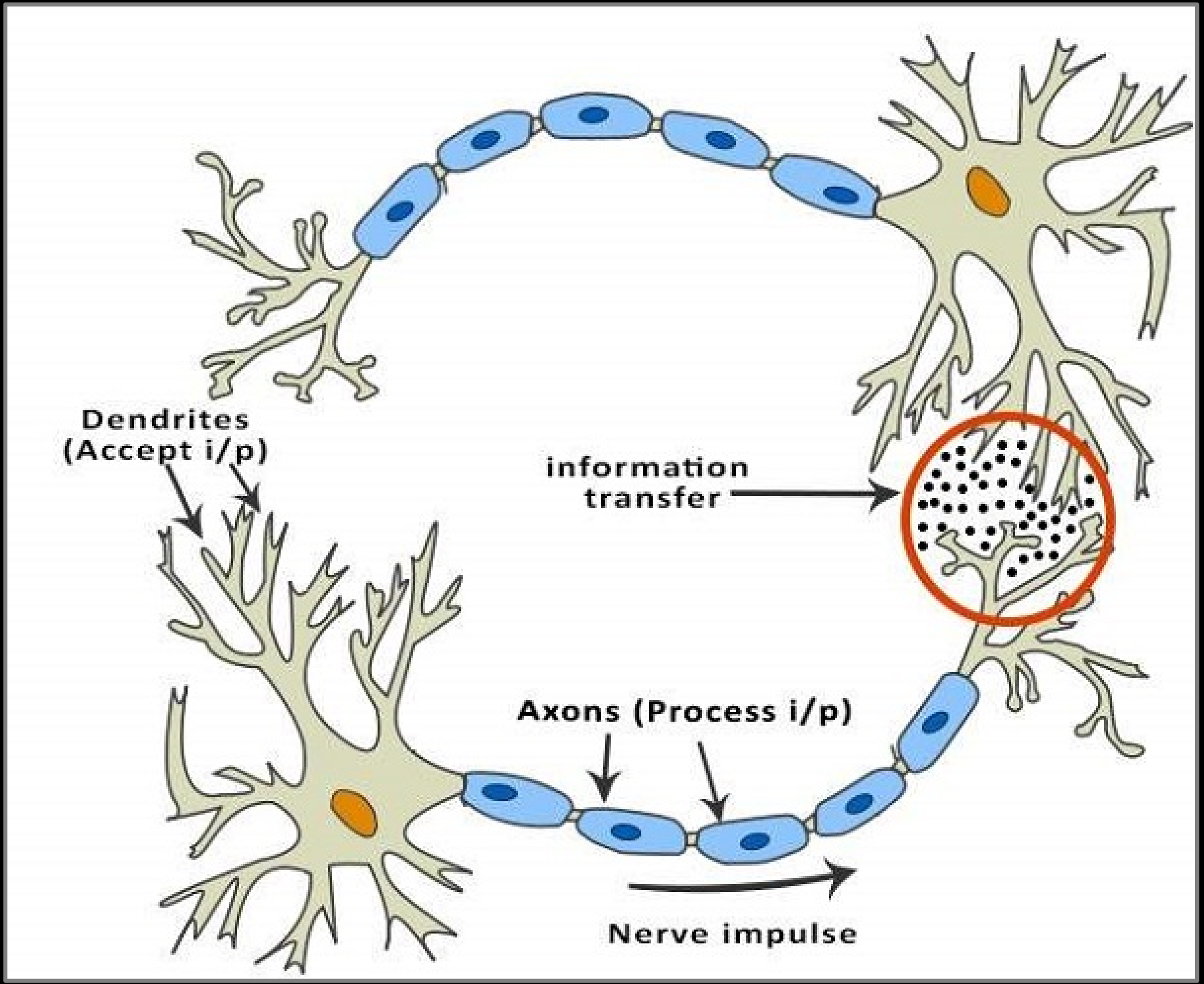
MLP review and development of a simple surrogate model

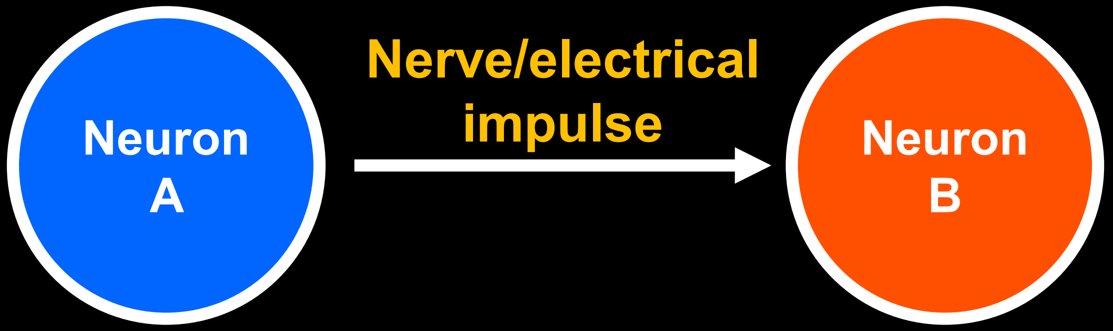
Kathy Breen

March 22, 2019

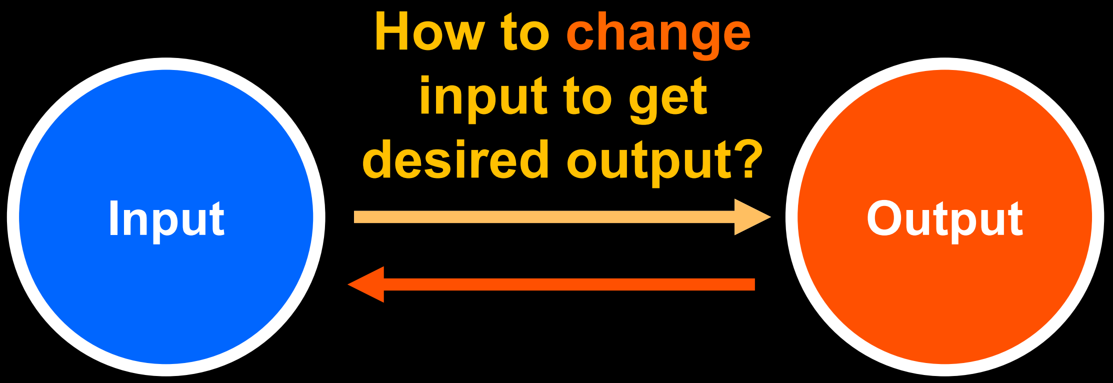
Artificial Neural Networks (ANN)



Anatomy of the Brain:



Artificial Neural Network:



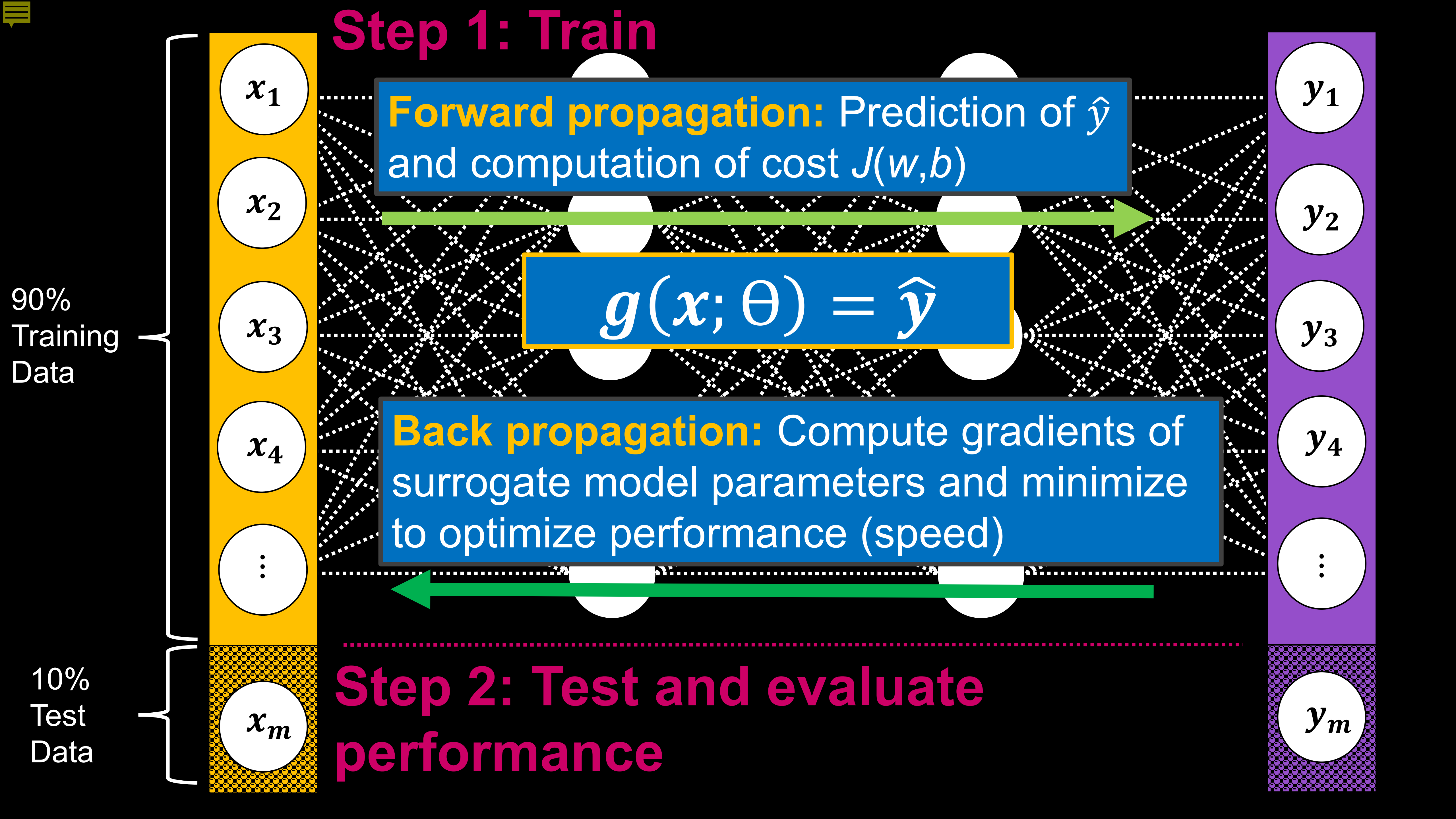

Step 1: Train
90% Training Data
10% Test Data
x_1
x_2
x_3
x_4
$\vdots$
x_m
y_1
y_2
y_3
y_4
$\vdots$
y_m
Forward propagation: Prediction of $\hat{y}$ and computation of cost $J(w,b)$
$g(x;\Theta) = \hat{y}$
Back propagation: Compute gradients of surrogate model parameters and minimize to optimize performance (speed)
Step 2: Test and evaluate performance

Table 1: Data generated using the "known" function.

X	Y
1	3
2	5
3	7
4	9
5	11

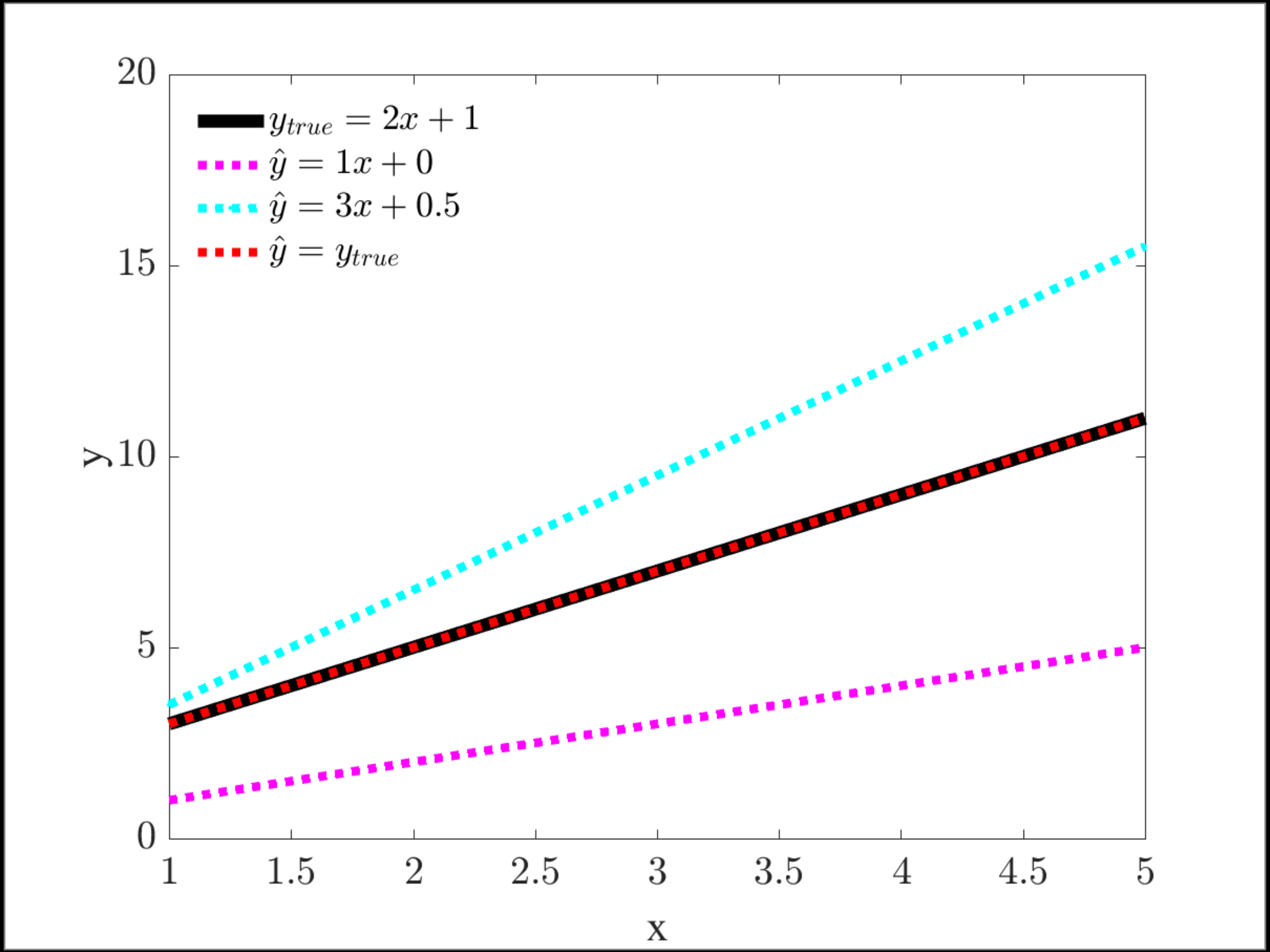
KNOWN:

$$y = 2x + 1$$

TASK: Given the data in Table 1, approximate the "known" function by altering the weight and bias applied to x to get the estimate $\hat{y}$ such that the difference between y and $\hat{y}$ is minimized.

$$\hat{y} = Wx + b$$

Epoch	W	b	RMSE
1	1	0	4.2426
2	3	0.5	2.8723
3	2	1	0

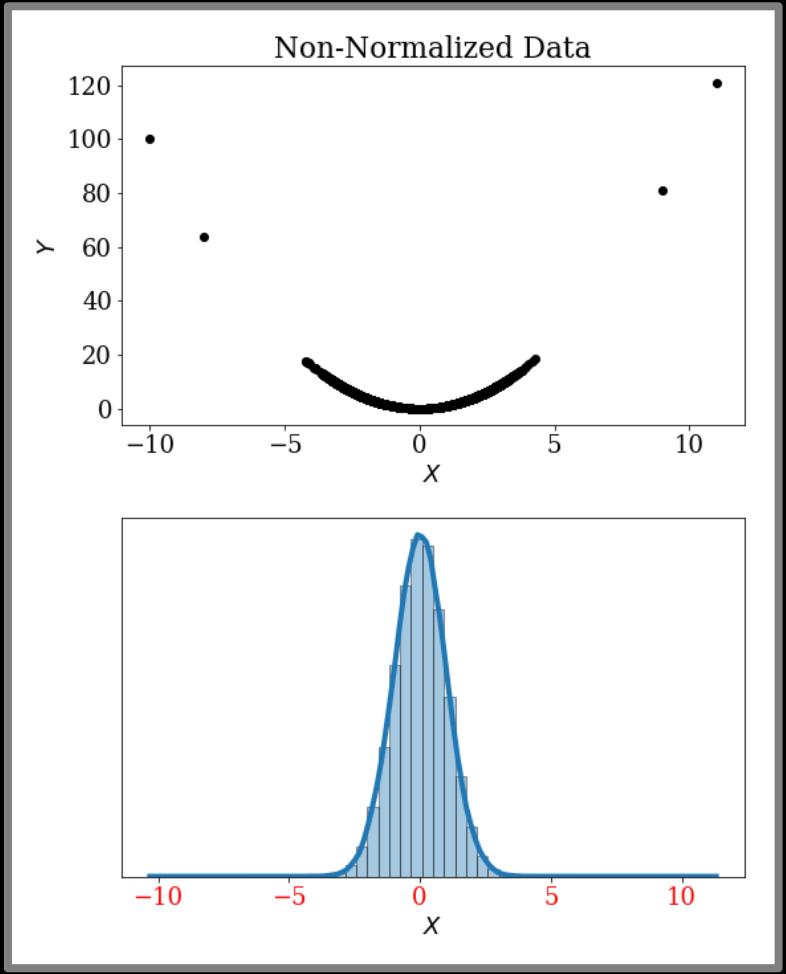


Preprocessing: Why is this important?

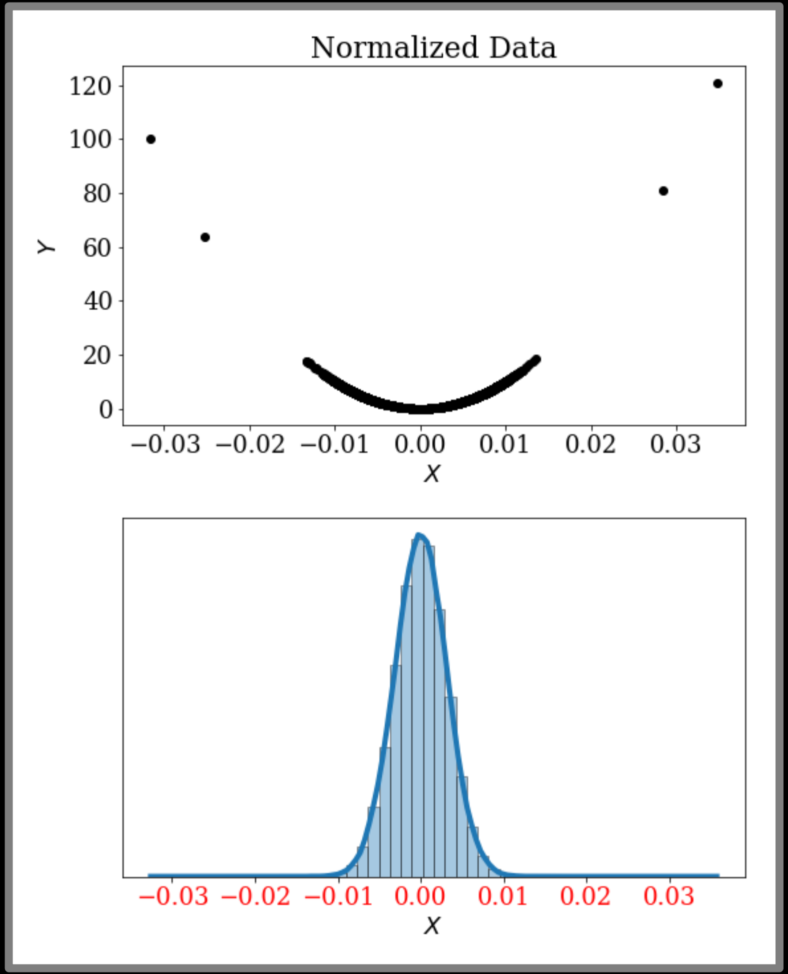
- Your data tell a story – are you communicating this to your DLM?
- Smaller numbers take up less space in memory
- Features transformed to similar scales aid in smooth optimization

Normalization (i.e., Standardization)

- Change the *scale* of the data set *without distorting the relative range*



$$\|X\|_2 = \sqrt{\sum_{k=1}^{n} {x_k}^2}$$

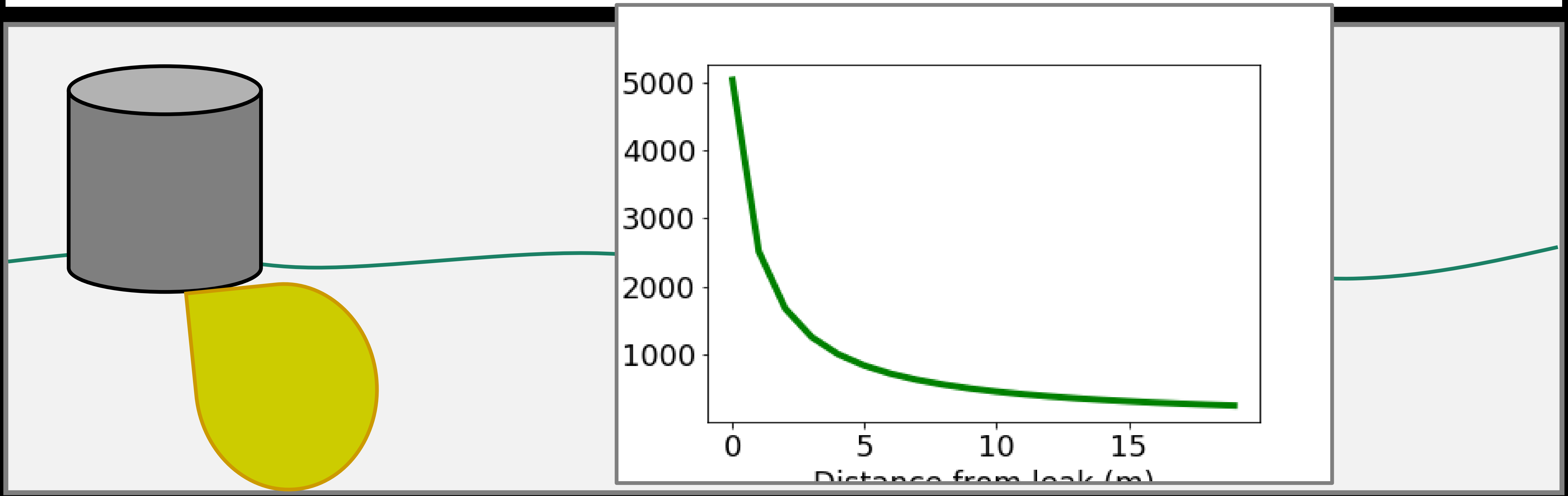


Simple model: pulse2D

Objective: Read model input files, calculate output over model domain, write model output files

This model is based on Example 6.2 of Bedient et. al (1999):

A tank holding chloride at a concentration of 10,000 mg/L accidentally leaks over an area of 10 m^2 into an aquifer. Assume the chloride is a completely conservative tracer, that D_x = 1 m^2/day and D_y = 0.1 m^2/day, and that the seepage velocity (V_w) is 1 m/day.

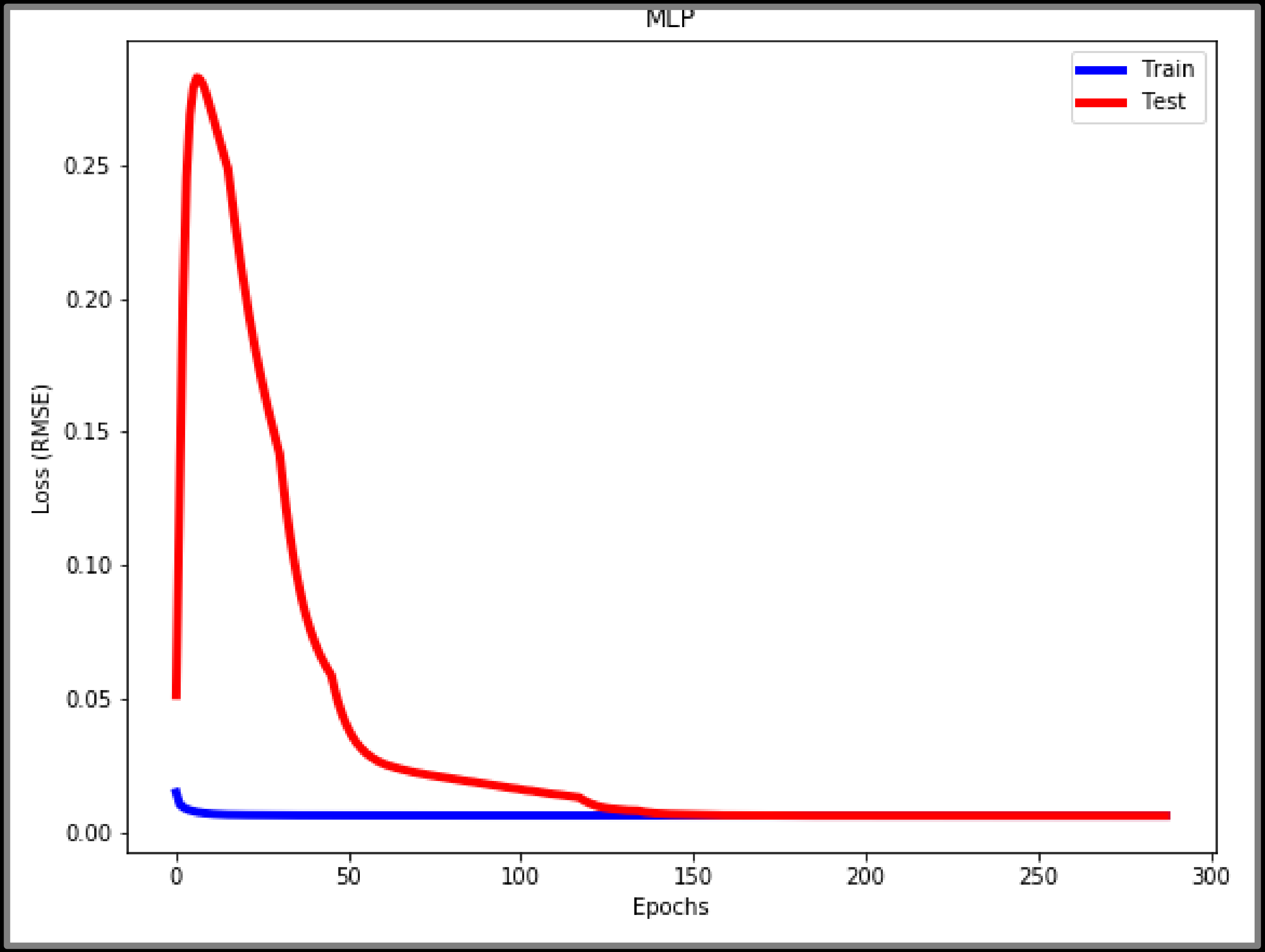


Creating a surrogate model

- Identify a physical model (pulse2D)
- Create N sets of model inputs
- Run the physical model N times
- Use inputs/outputs from N model runs to create training data
- Preprocess data
- Partition data into train/test sets
- Build a DLM
- Train and test DLM

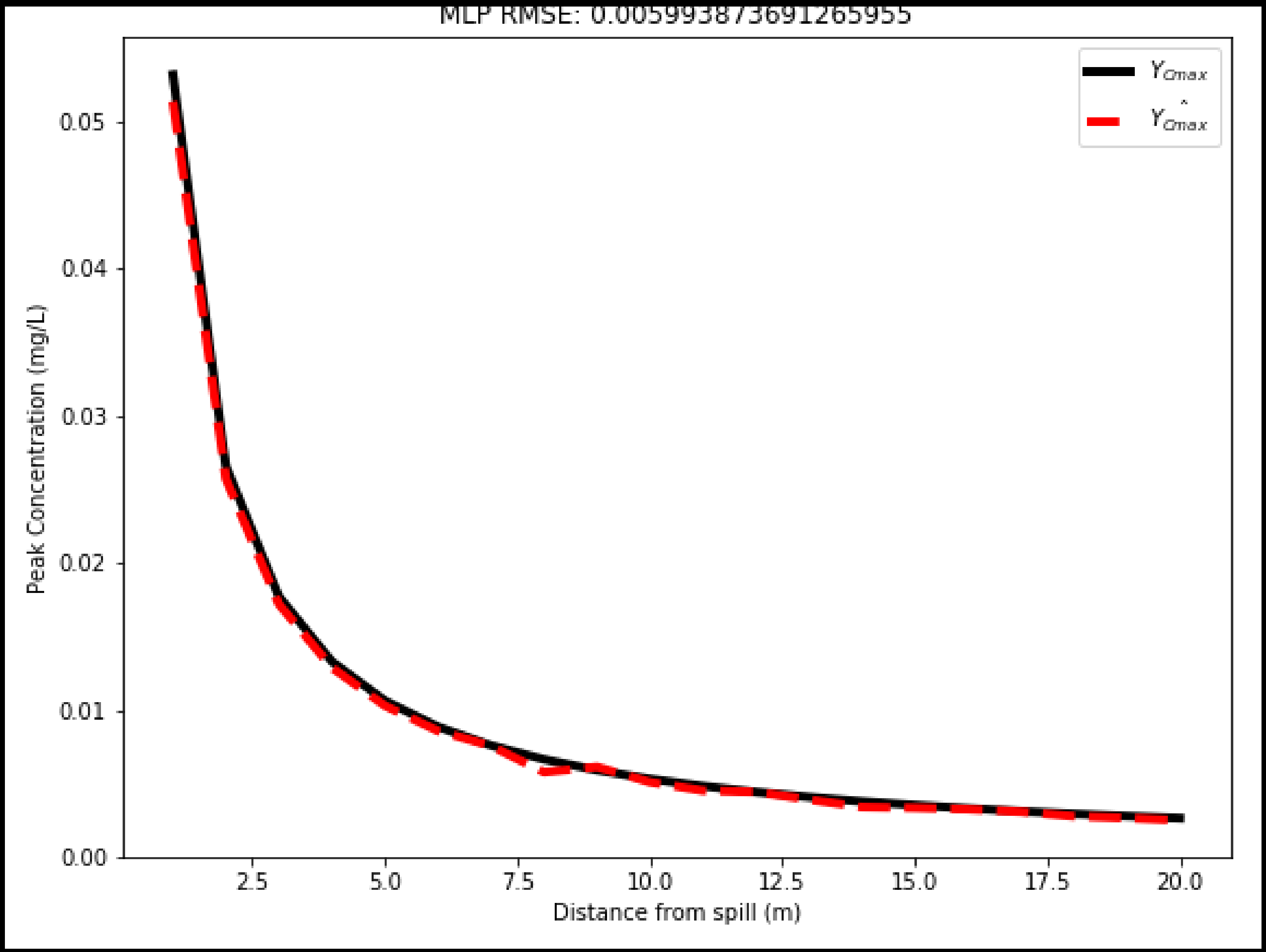
DEMO

Visualize Train and Test Loss



pulse2d_MLP.py

Visualize True data and MLP Prediction



Prediction RMSE: 0.006

pulse2d_MLP.py