1- Intro to deep learning and population dynamics

Data Science and AI for Neuroscience Summer School 2022, Chen Institute at Caltech

Chethan Pandarinath, Ph.D.
(he/him)

Assistant Professor
Coulter Department of Biomedical Engineering
Department of Neurosurgery
Emory University & Georgia Tech

@chethan

Christopher Versteeg, Ph.D.

Postdoctoral Fellow
Coulter Department of Biomedical Engineering
Emory University & Georgia Tech

SYSTEMS NEURAL ENGINEERING LABORATORY

snel.gatech.edu

EMORY UNIVERSITY

Georgia Tech

What you'll hear about in this lecture

What you'll hear about in this lecture

Neural network basics

What you'll hear about in this lecture

Neural network basics

Deep autoencoders

What you'll hear about in this lecture

Neural network basics

Deep autoencoders

Intro to neural population dynamics

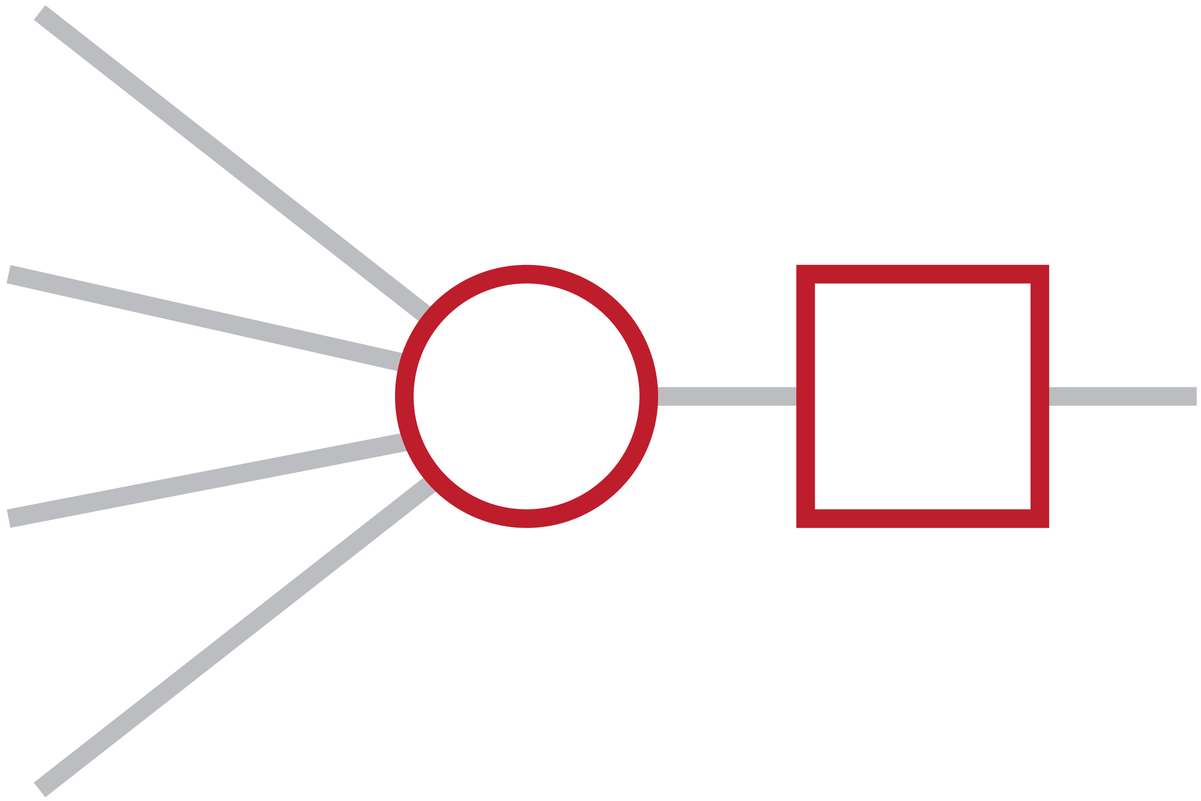
What you'll hear about in this lecture

Neural network basics

Deep autoencoders

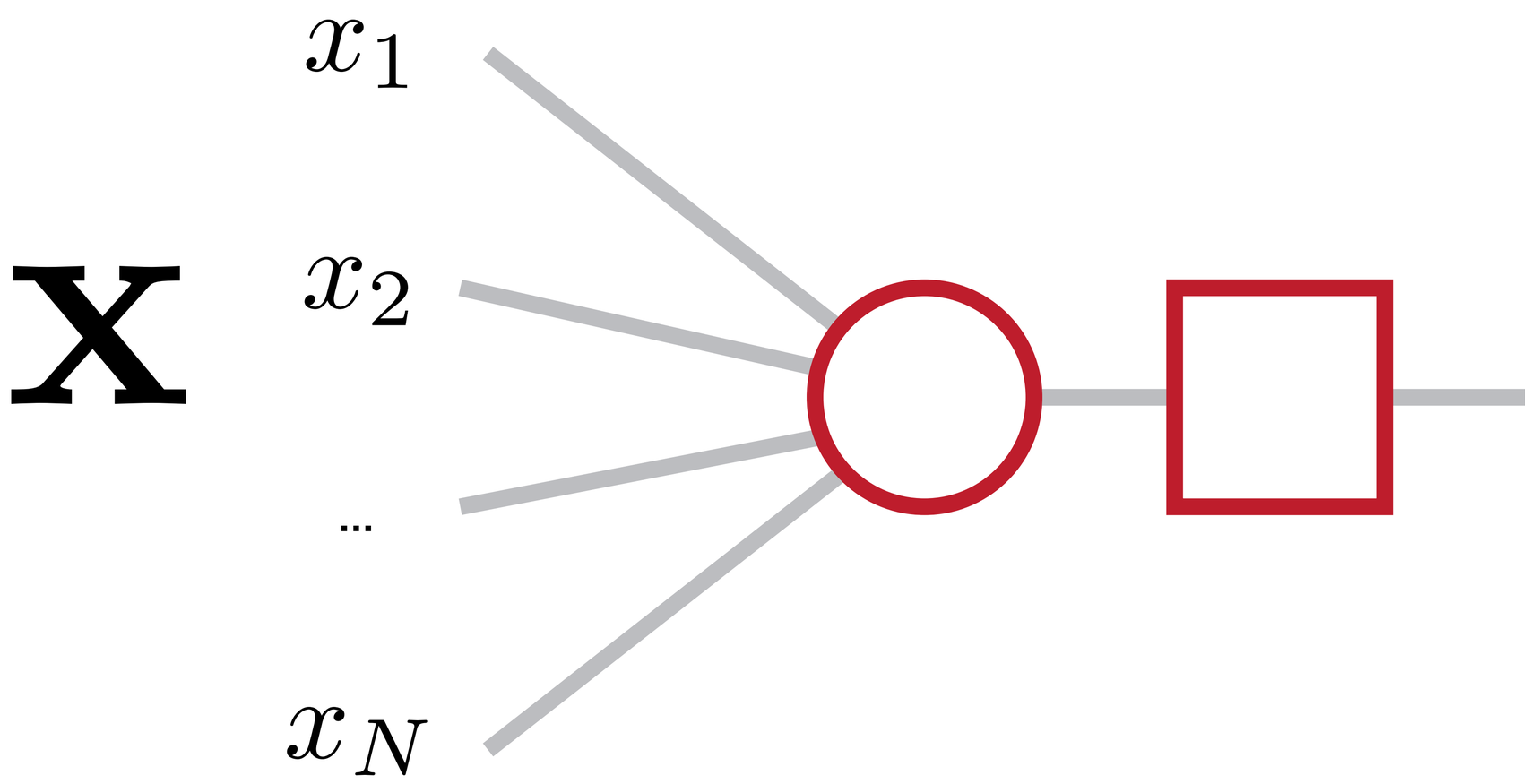
Intro to neural population dynamics

Artificial neurons



Artificial neurons

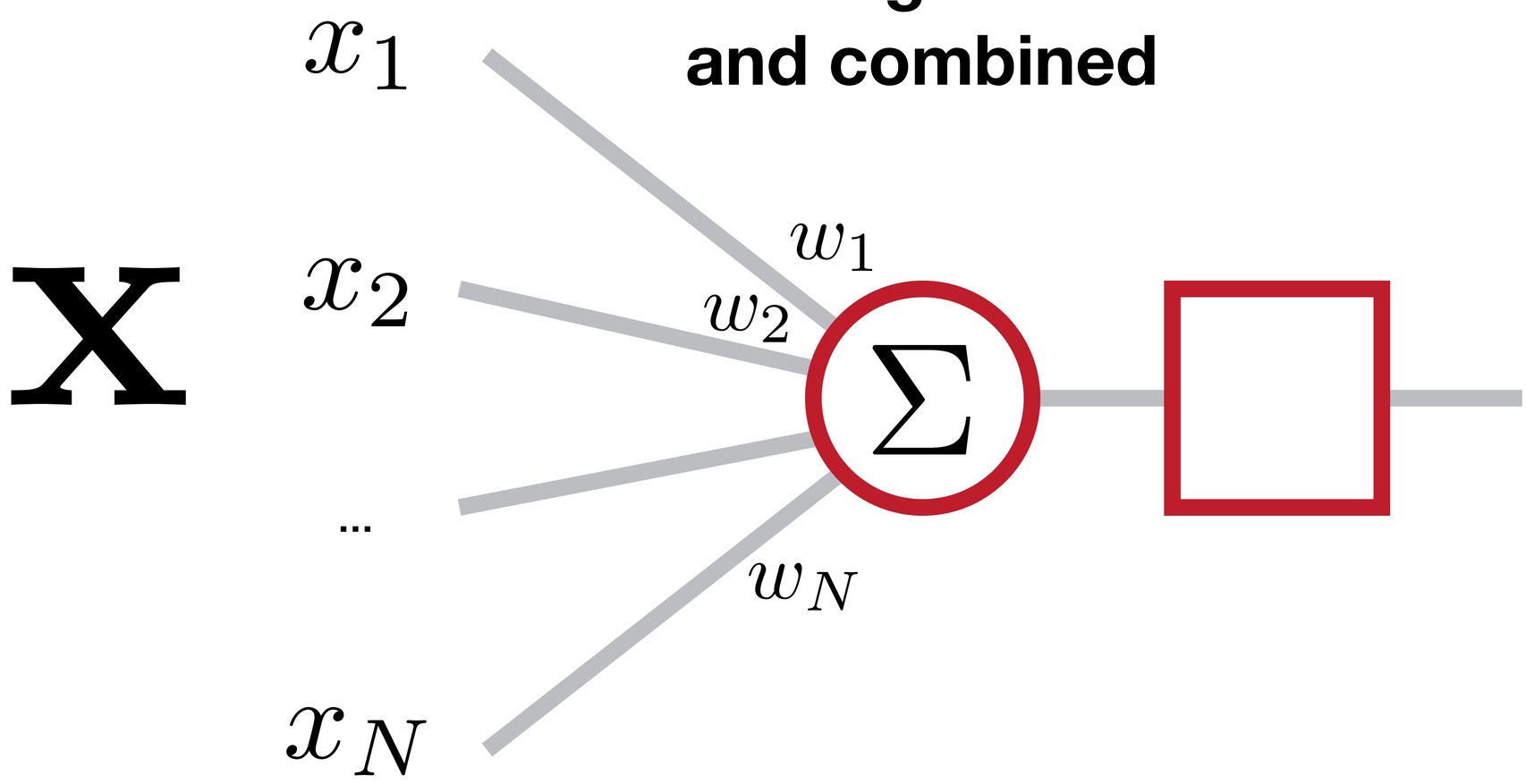
Inputs



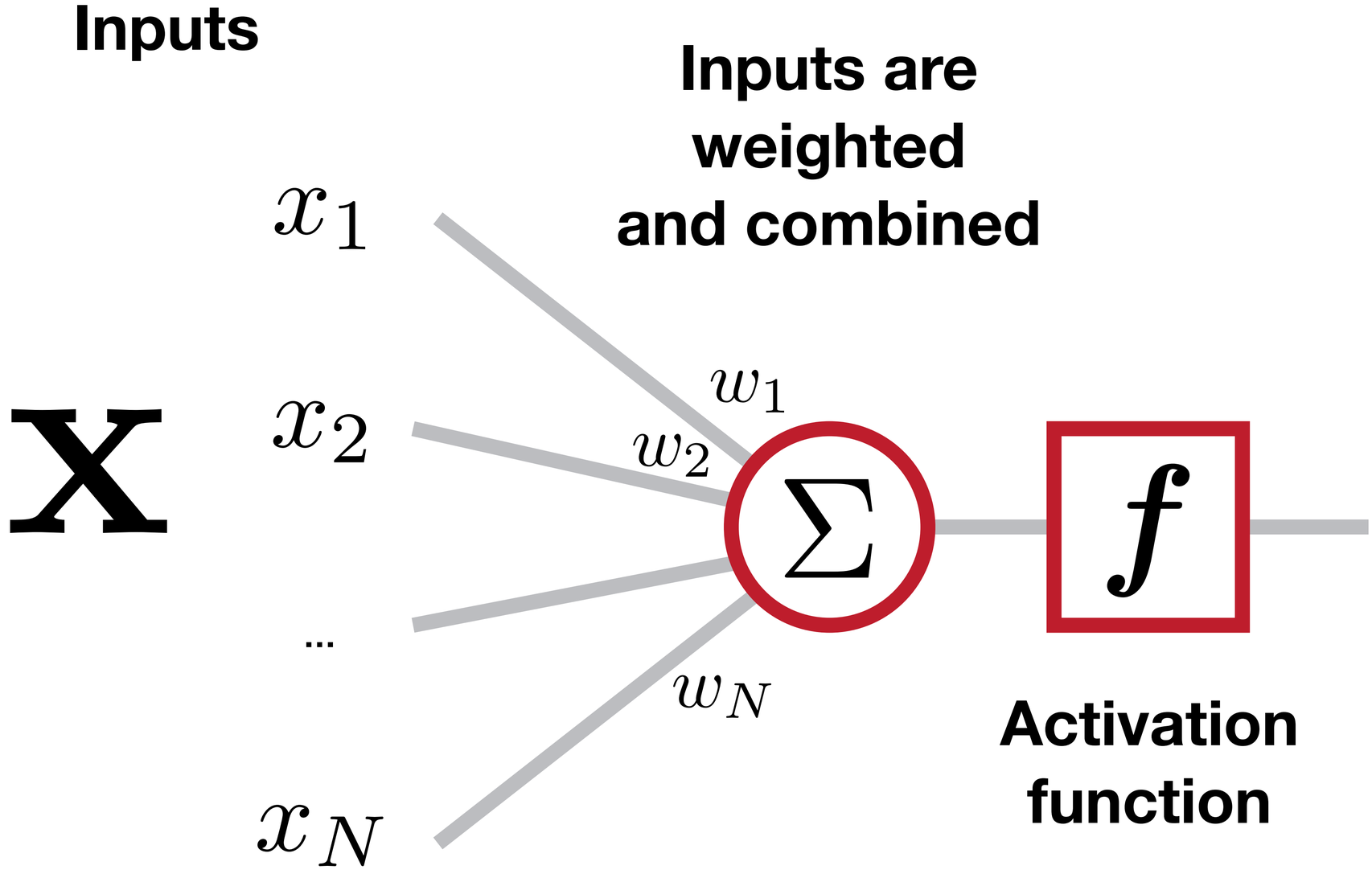
Artificial neurons

Inputs

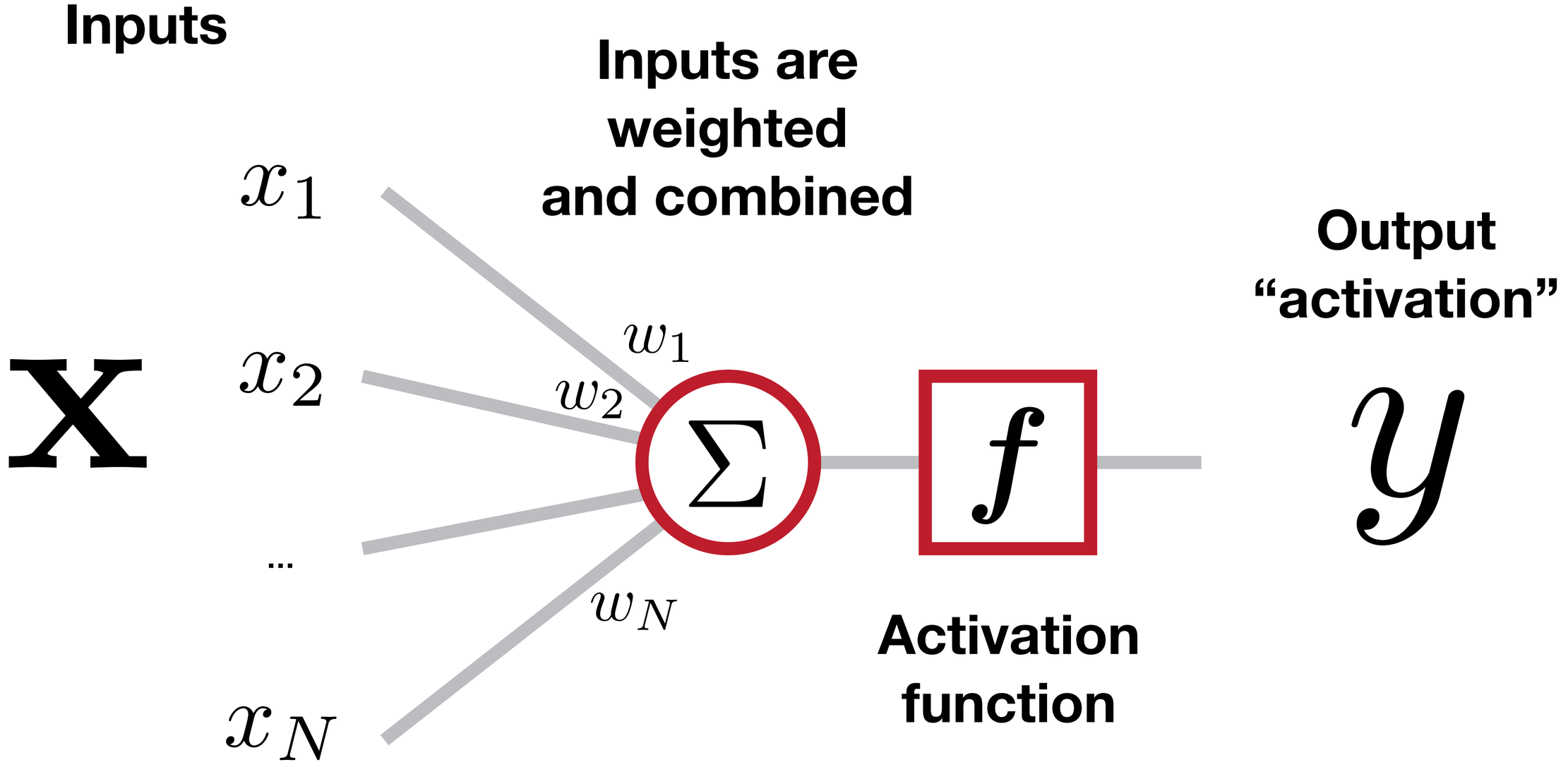
Inputs are weighted and combined



Artificial neurons



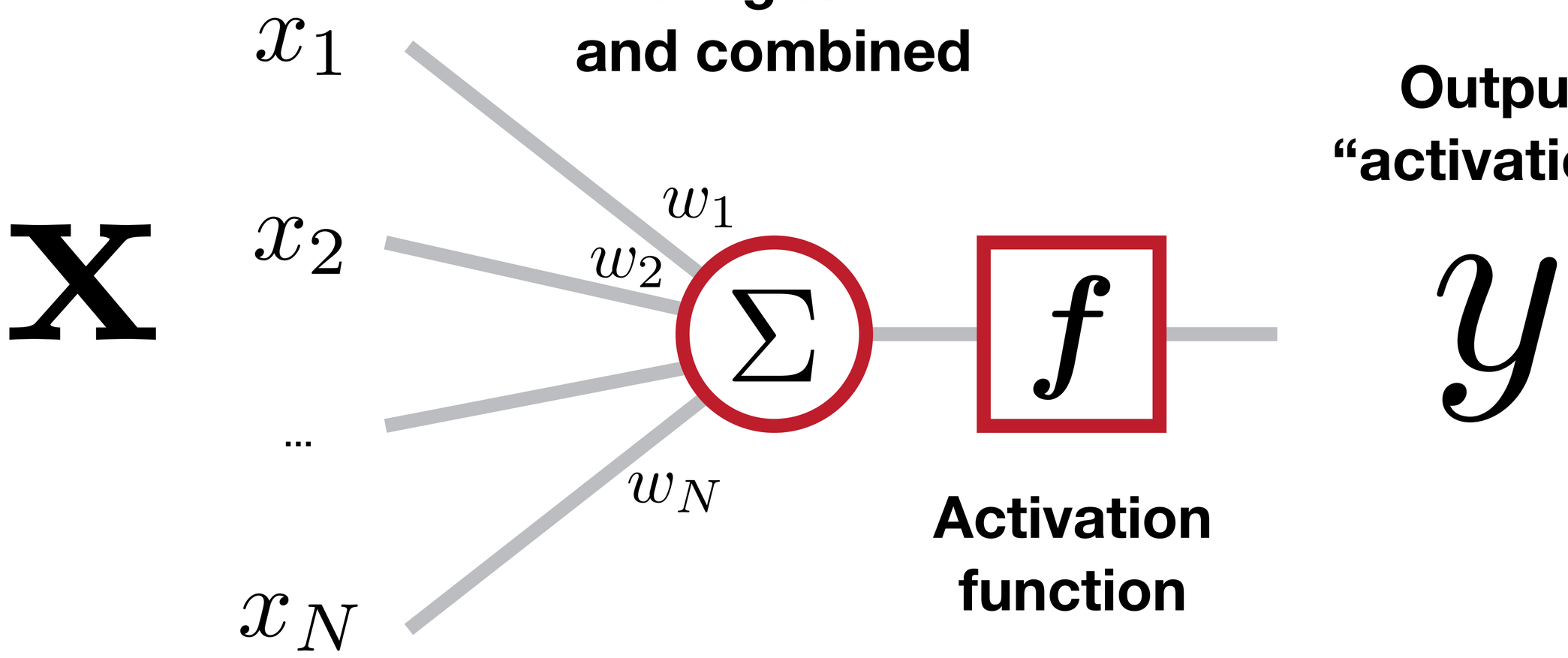
Artificial neurons



Artificial neurons

Inputs

Inputs are weighted and combined

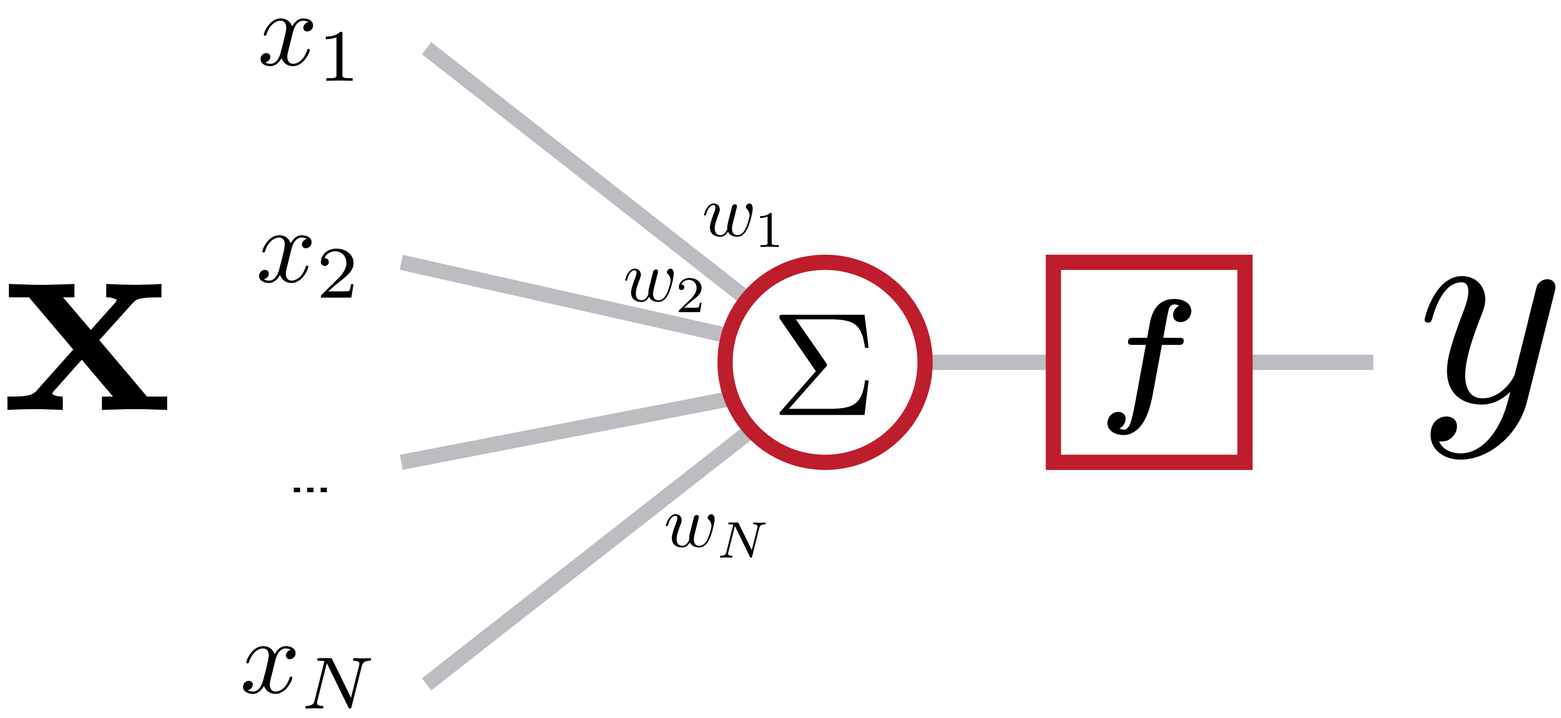


Output "activation"

Activation function

Warren S. McCulloch and Walter Pitts (1943)
"A logical calculus of the ideas immanent in nervous activity."
Bulletin of Mathematical Biophysics 5:115-133

Artificial neurons



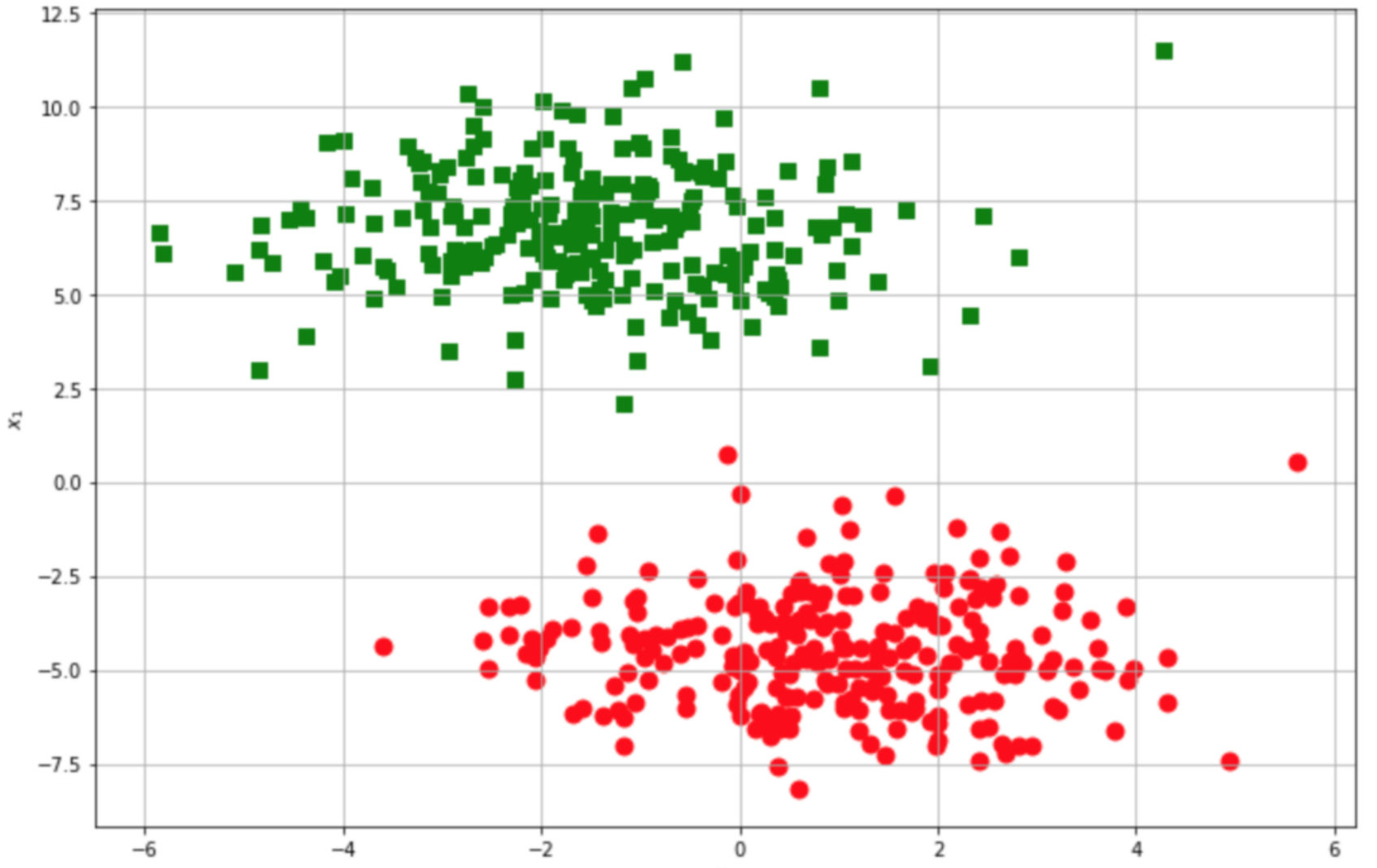
$$y = \boldsymbol{f}(w_1 * x_1 + w_2 * x_2 + ... + b)$$

Simple problem - 2 dimensional classification

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Green: class 0

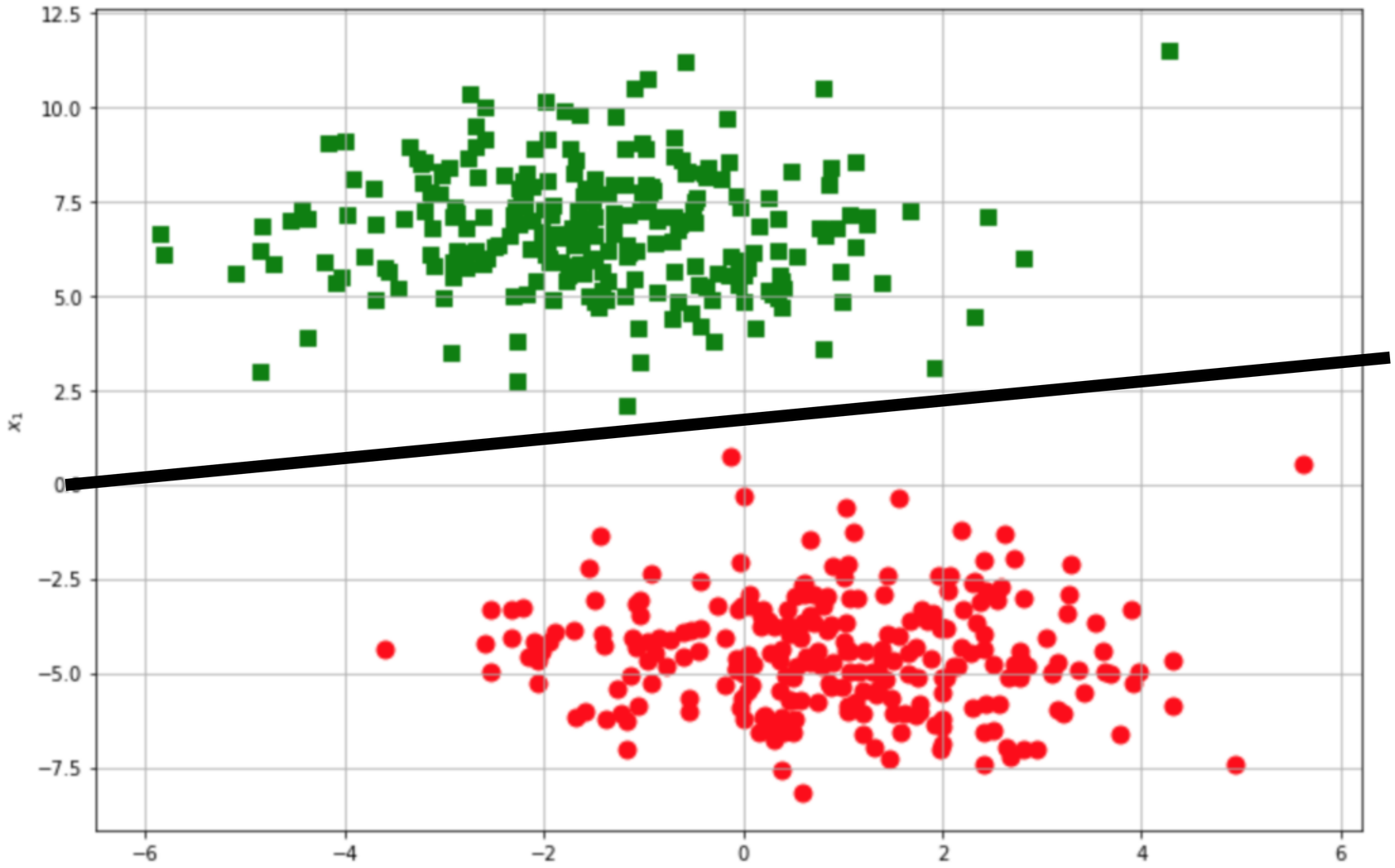
Red: class 1



https://www.bonaccorso.eu/2017/09/28/linearly-separable-no-for-me-it-is-a-brief-introduction-to-kernel-methods/

Simple problem - 2 dimensional classification

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$



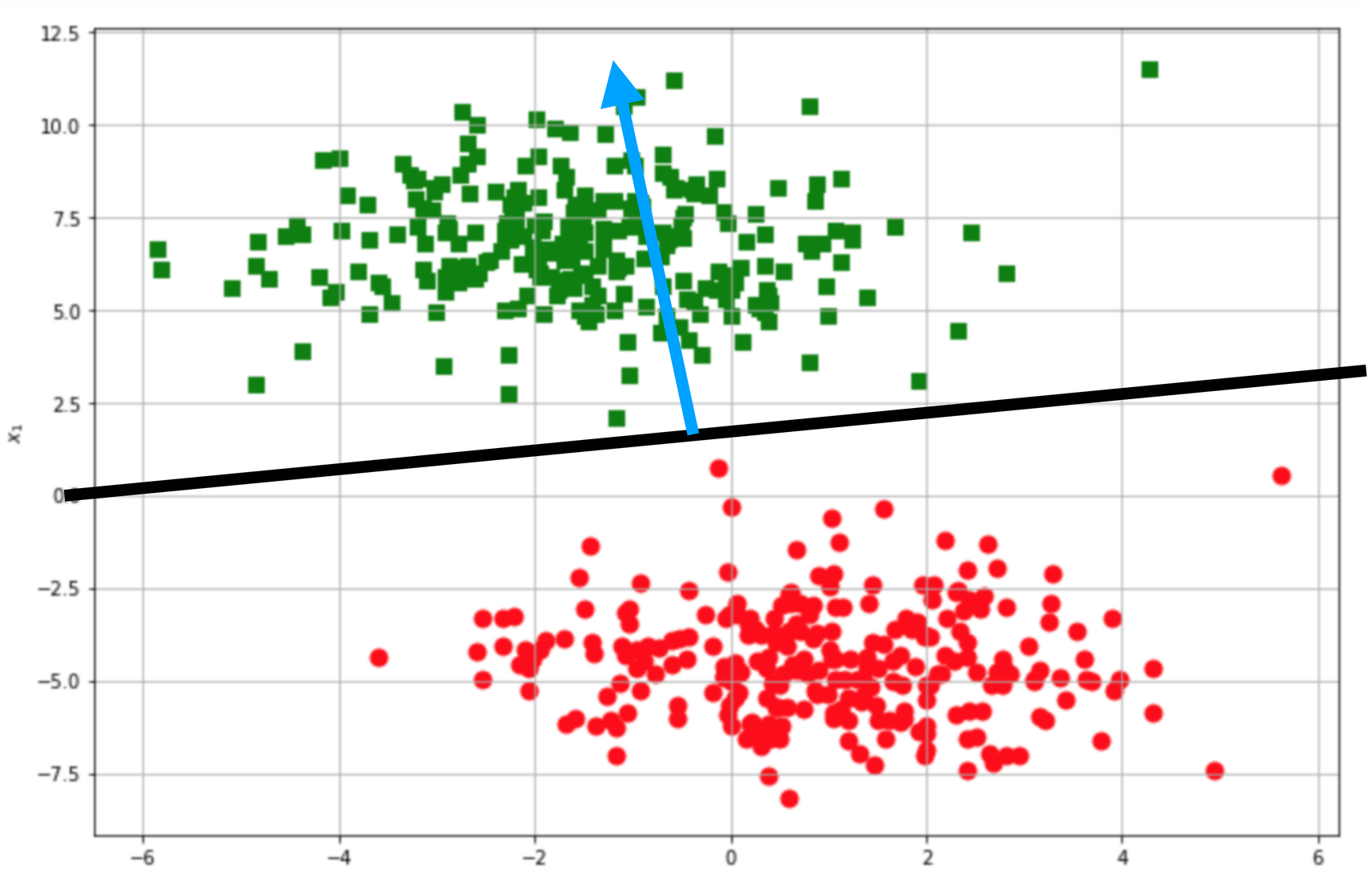
Green: class 0

Red: class 1

https://www.bonaccorso.eu/2017/09/28/linearly-separable-no-for-me-it-is-a-brief-introduction-to-kernel-methods/

Simple problem - 2 dimensional classification

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

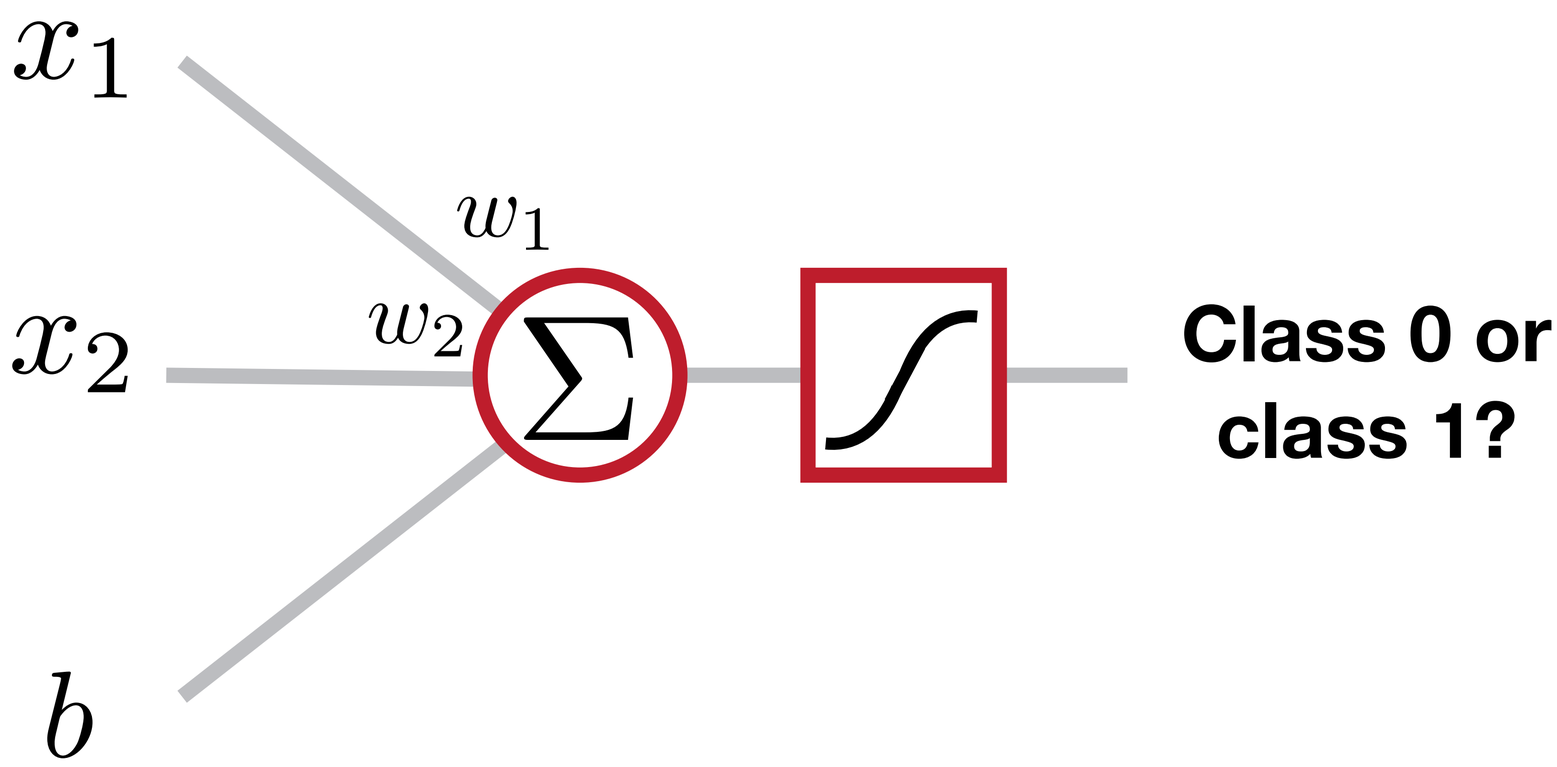


Green: class 0

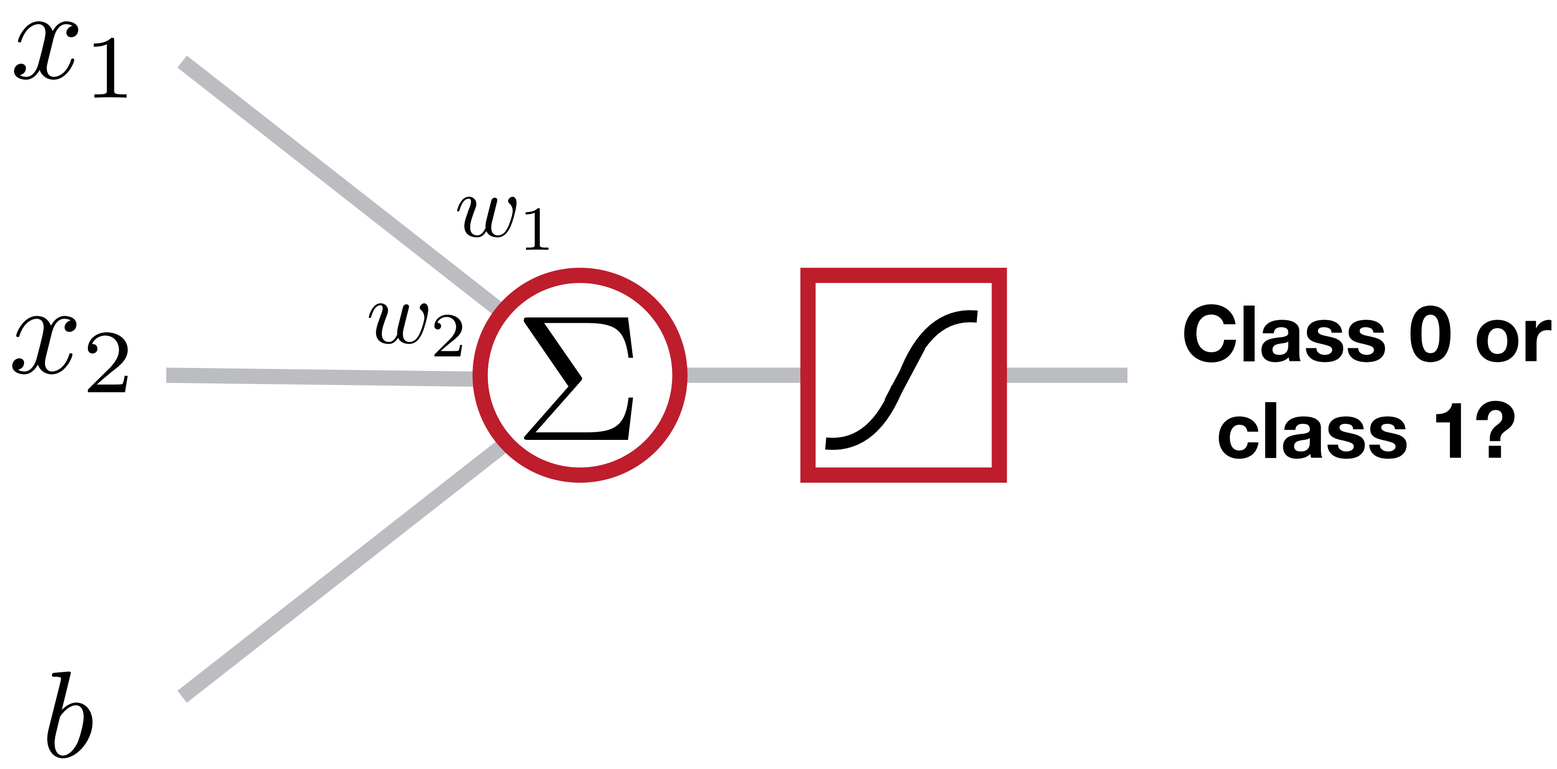
Red: class 1

https://www.bonaccorso.eu/2017/09/28/linearly-separable-no-for-me-it-is-a-brief-introduction-to-kernel-methods/

Simple classifier - Perceptron

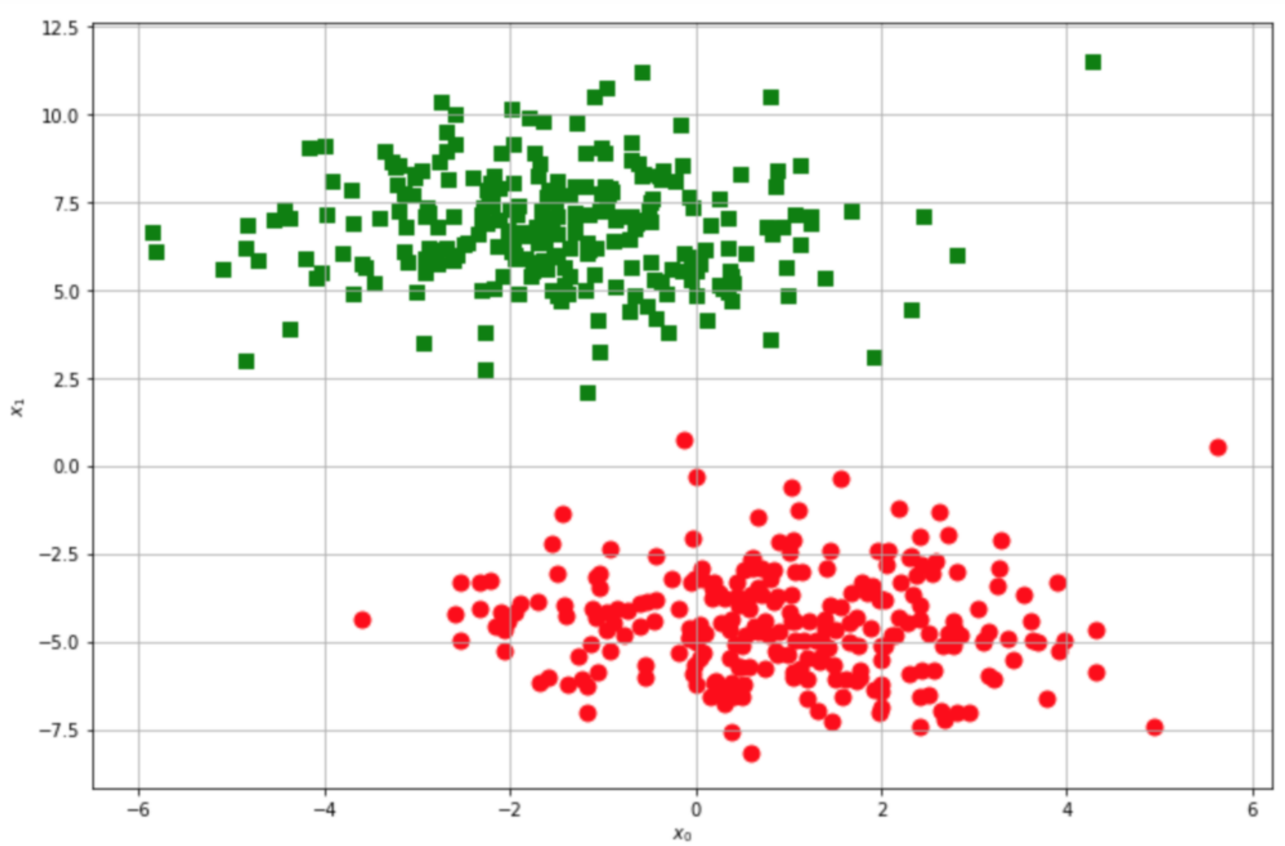


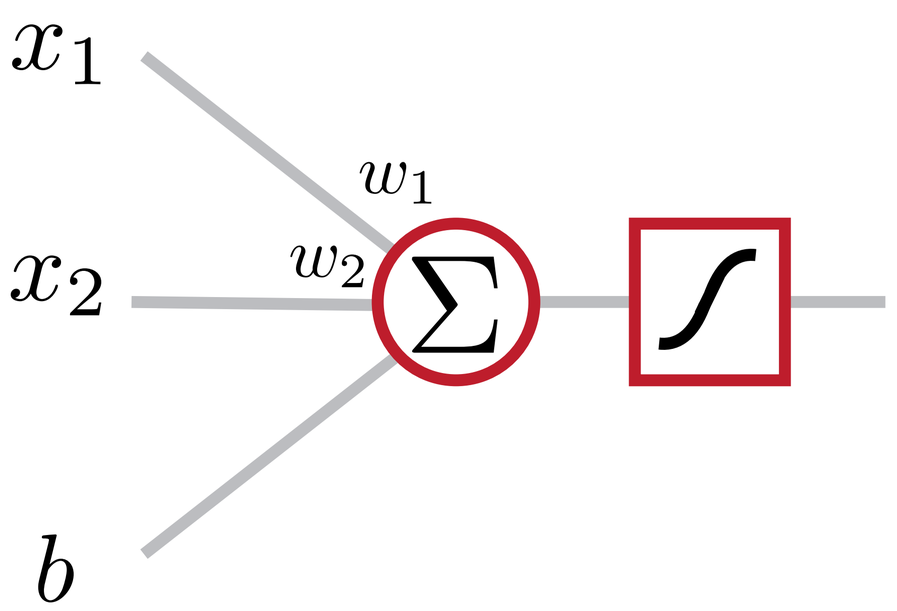
Simple classifier - Perceptron



Frank Rosenblatt (1957)
“The Perceptron — a perceiving and recognizing automaton.”
Report 85–460–1, Cornell Aeronautical Laboratory.

Classification with the perceptron

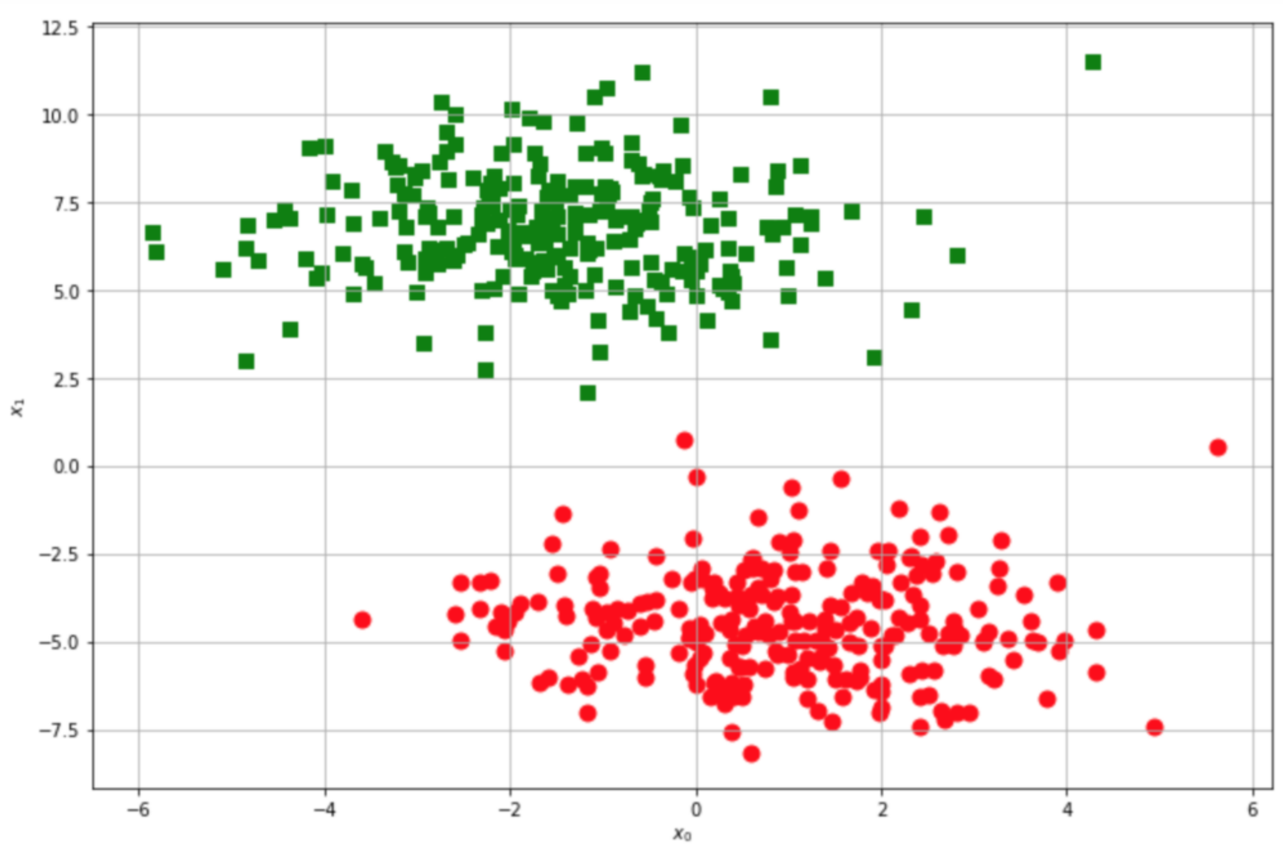


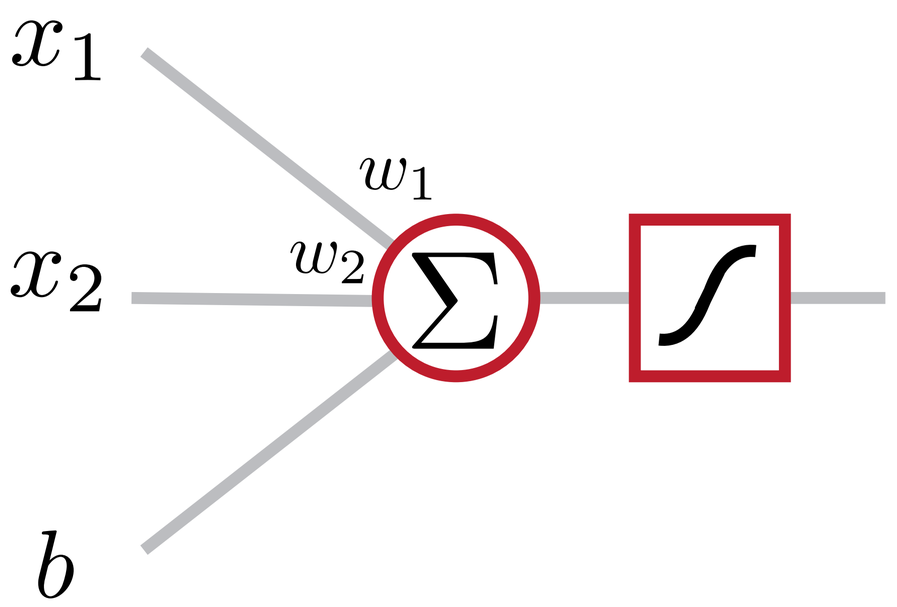


class 0 or class 1?

https://www.bonaccorso.eu/2017/09/28/linearly-separable-no-for-me-it-is-a-brief-introduction-to-kernel-methods/

Classification with the perceptron



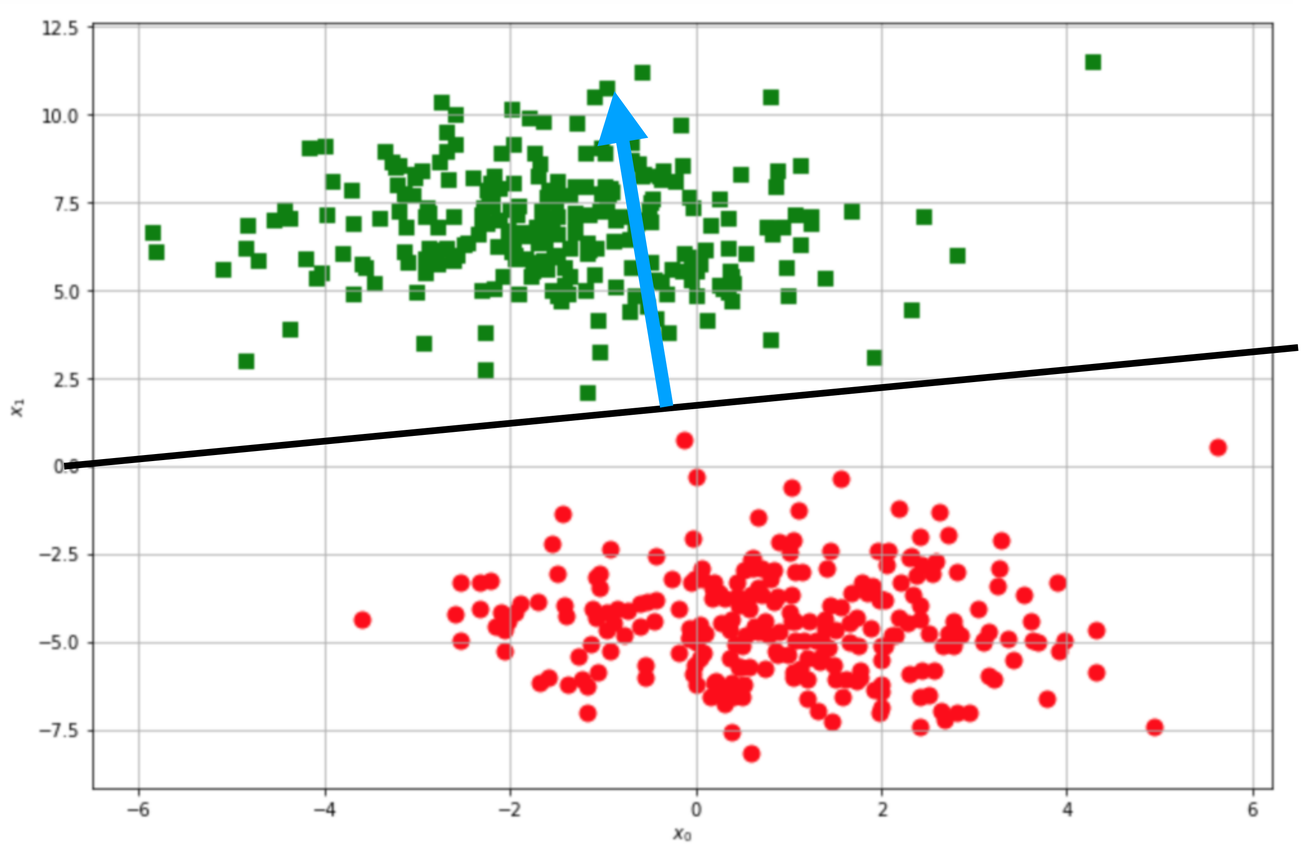


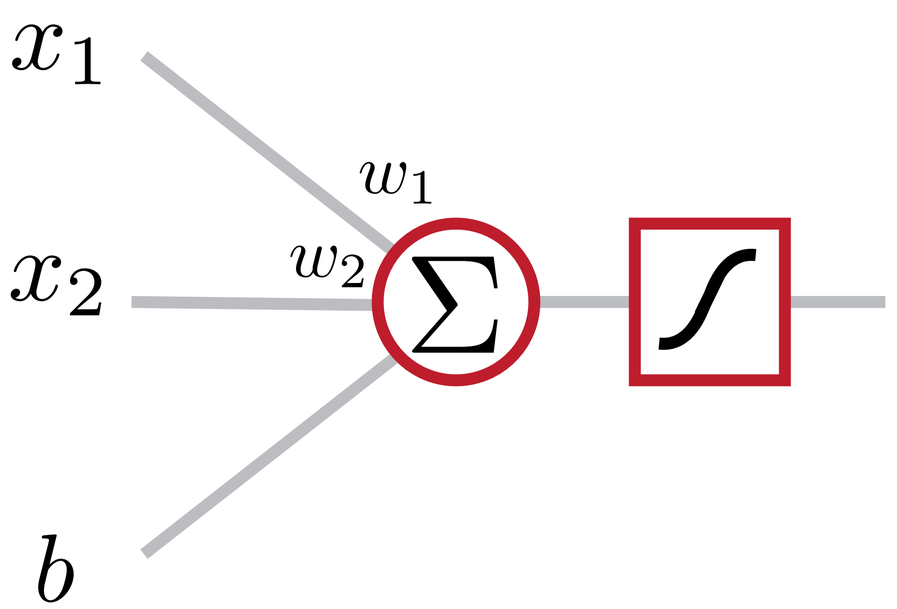
class 0 or class 1?

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \bullet \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} + b$$

https://www.bonaccorso.eu/2017/09/28/linearly-separable-no-for-me-it-is-a-brief-introduction-to-kernel-methods/

Classification with the perceptron



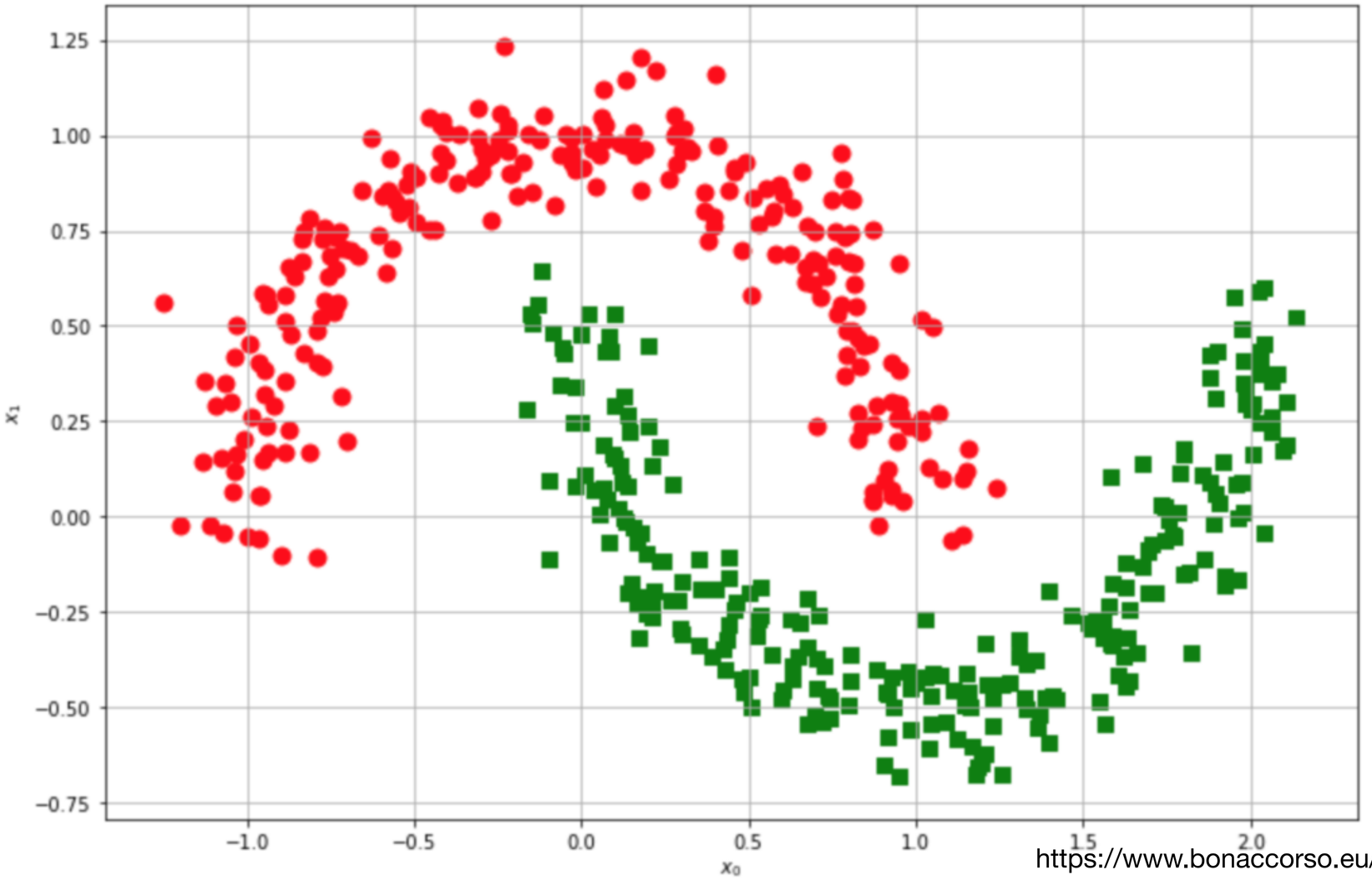


class 0 or class 1?

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \bullet \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} + b$$

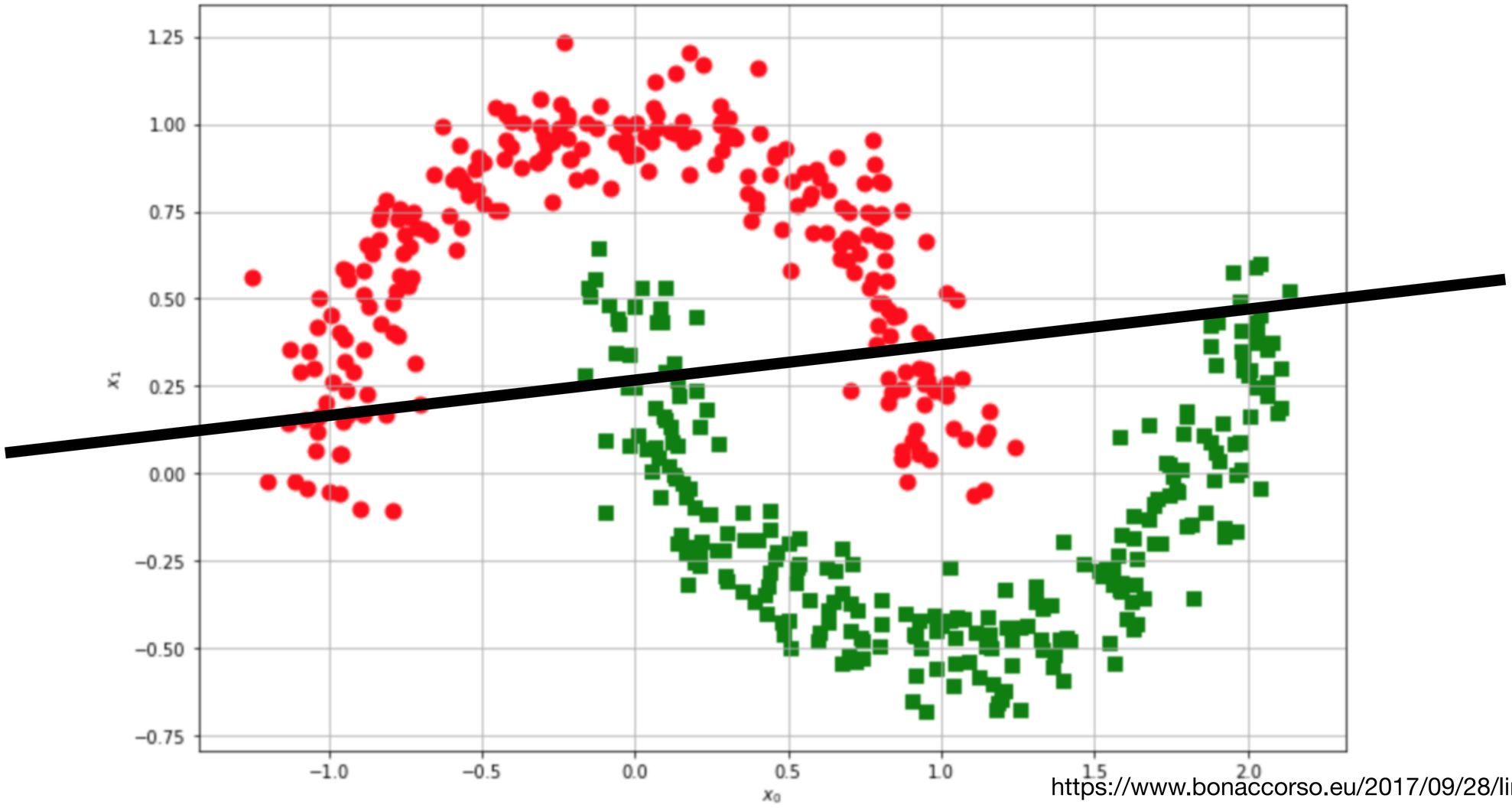
https://www.bonaccorso.eu/2017/09/28/linearly-separable-no-for-me-it-is-a-brief-introduction-to-kernel-methods/

Harder problem



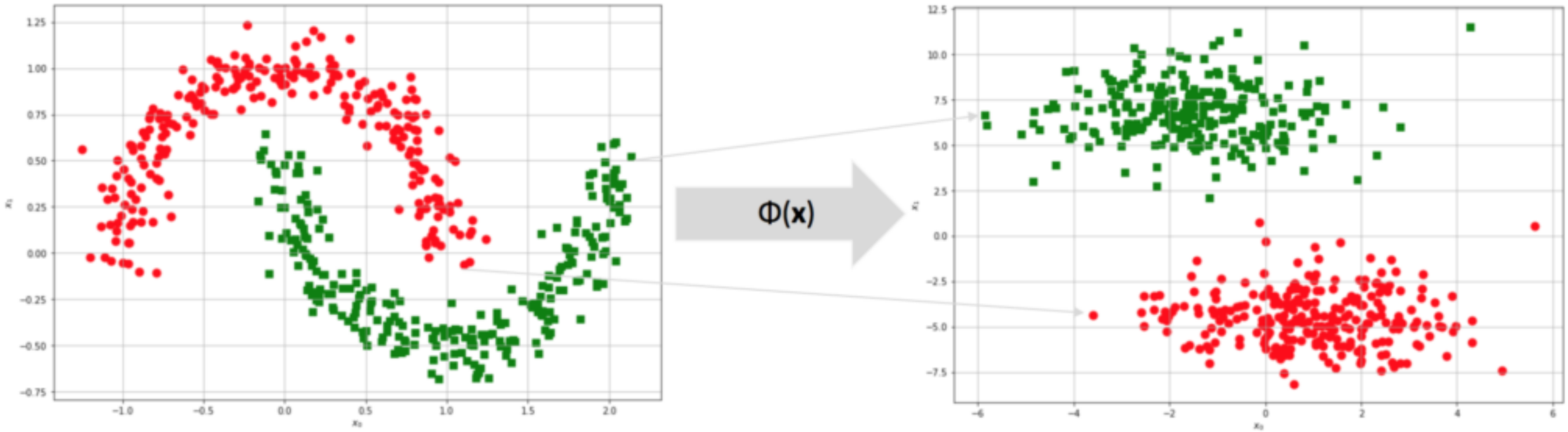
https://www.bonaccorso.eu/2017/09/28/linearly-separable-no-for-me-it-is-a-brief-introduction-to-kernel-methods/

Harder problem



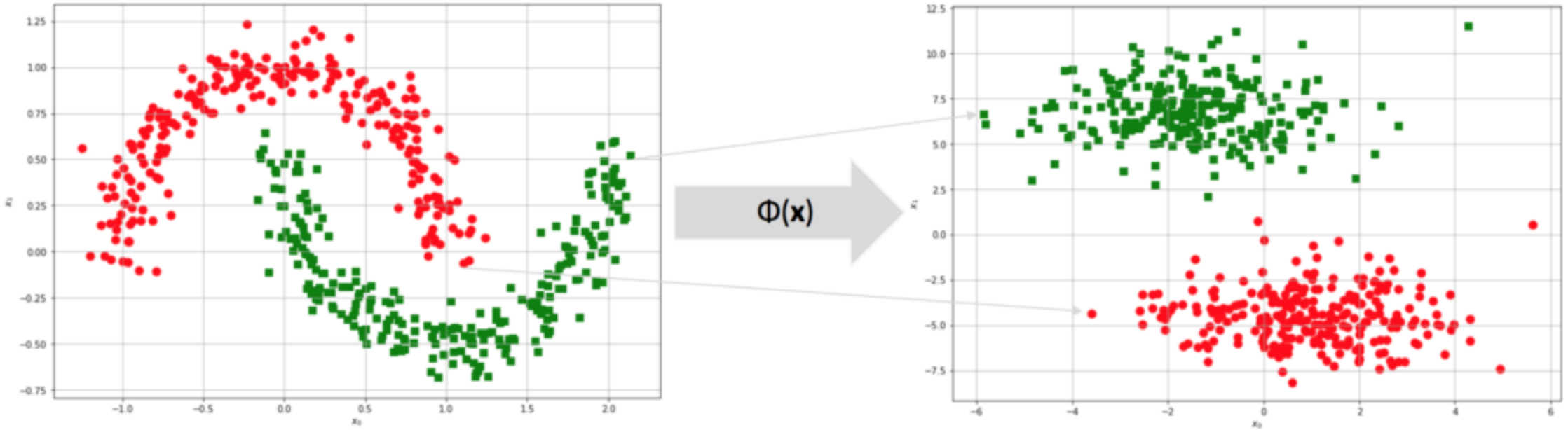
https://www.bonaccorso.eu/2017/09/28/linearly-separable-no-for-me-it-is-a-brief-introduction-to-kernel-methods/

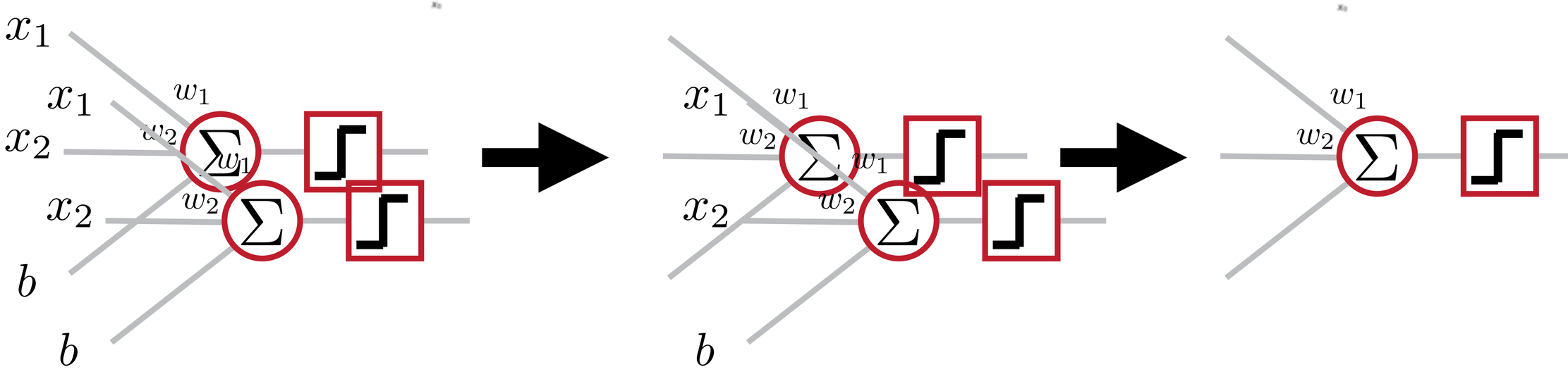
Strategy - multiple transformations



https://www.bonaccorso.eu/2017/09/28/linearly-separable-no-for-me-it-is-a-brief-introduction-to-kernel-methods/

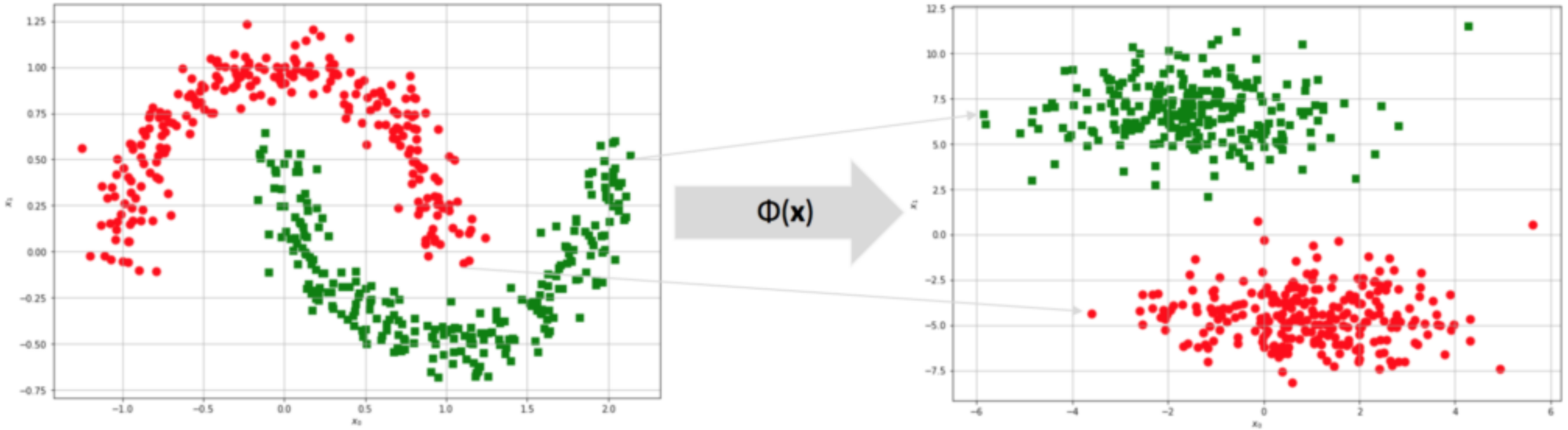
Multi-layer Perceptron (MLP)

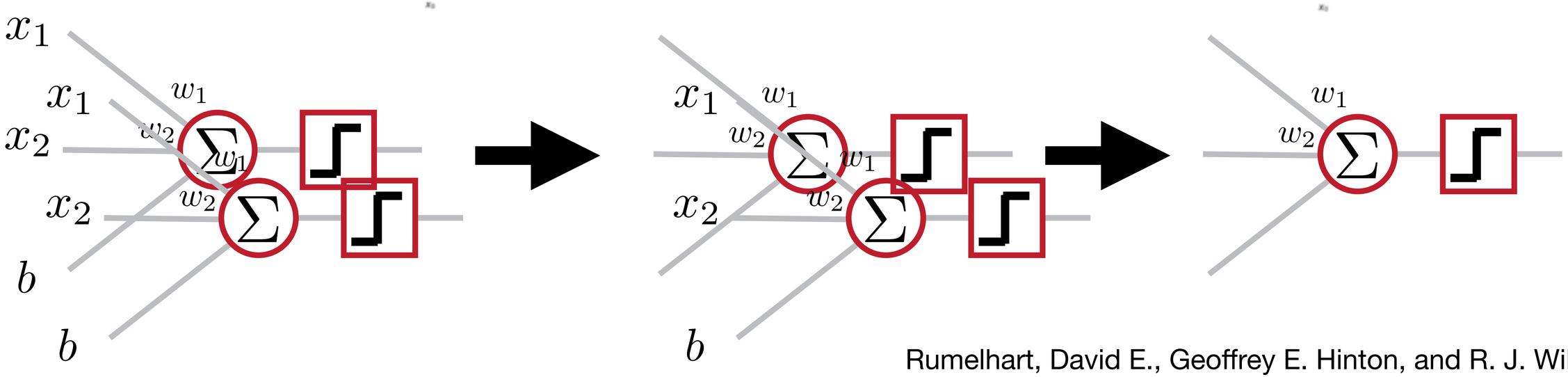




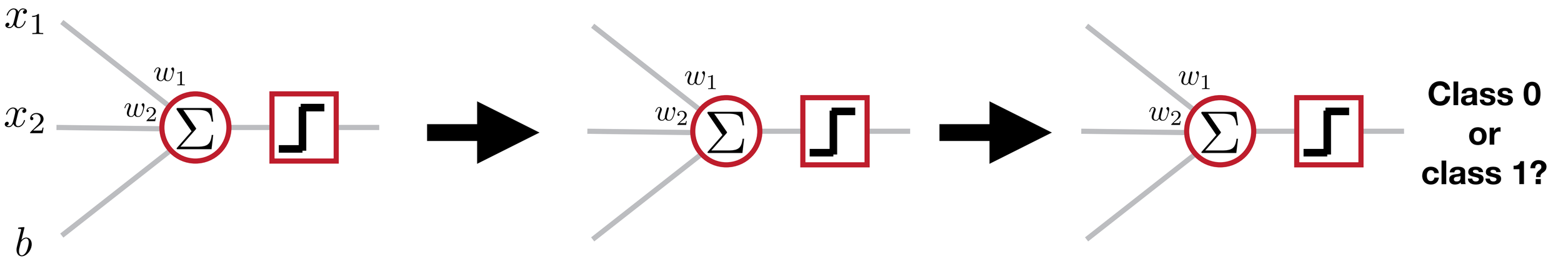
Class 0 or class 1?

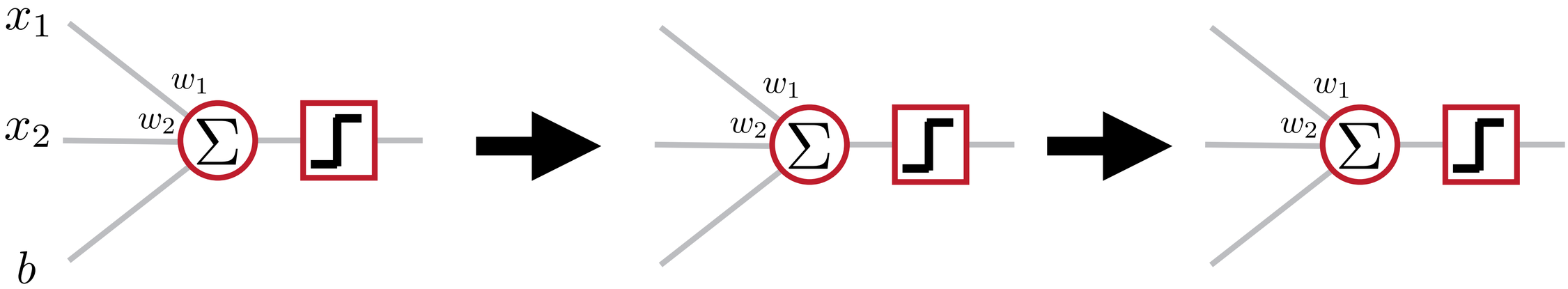
Multi-layer Perceptron (MLP)



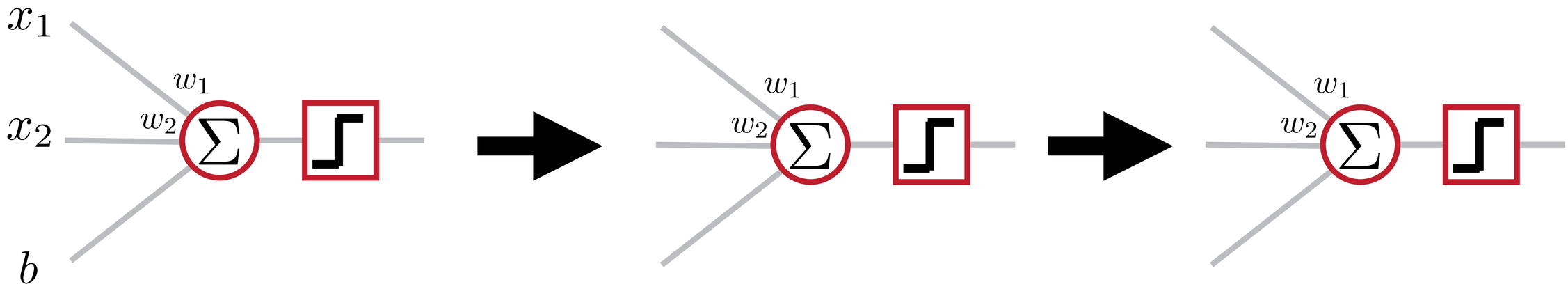


Rumelhart, David E., Geoffrey E. Hinton, and R. J. Williams. (1986)
“Learning Internal Representations by Error Propagation". *Parallel distributed processing*


x_1
x_2
b
w_1
w_2
$\sum$
w_1
w_2
$\sum$
w_1
w_2
$\sum$
Class 0
or
class 1?

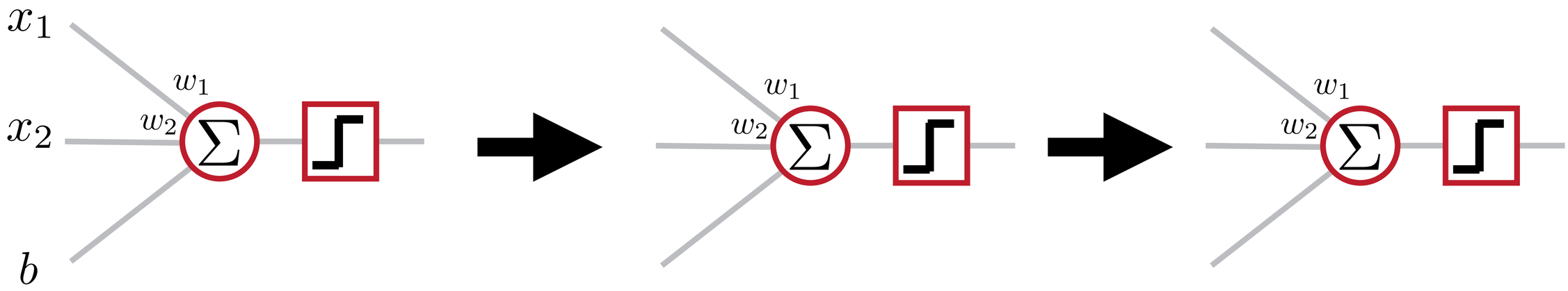


What are the parameters?



What are the parameters?

What is the objective (loss)?



What are the parameters?

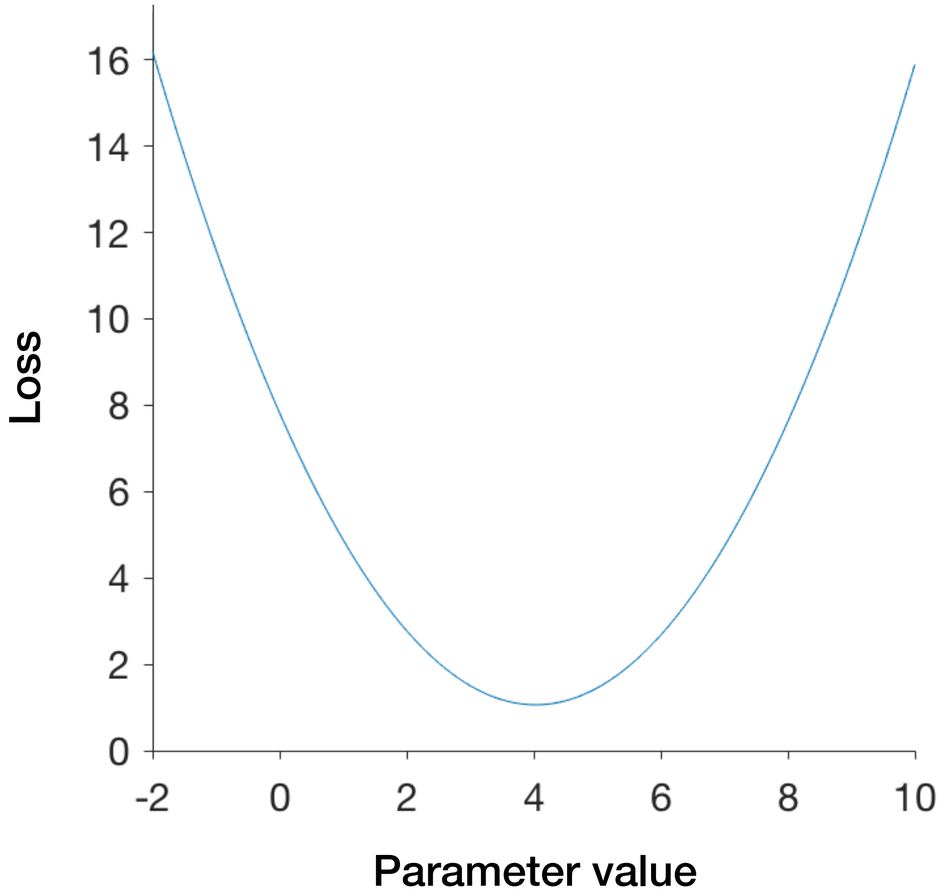
What is the objective (loss)?

How do we find the parameters?

Gradient descent for one parameter

(a.k.a., “try something, then keep trying to make it better”)

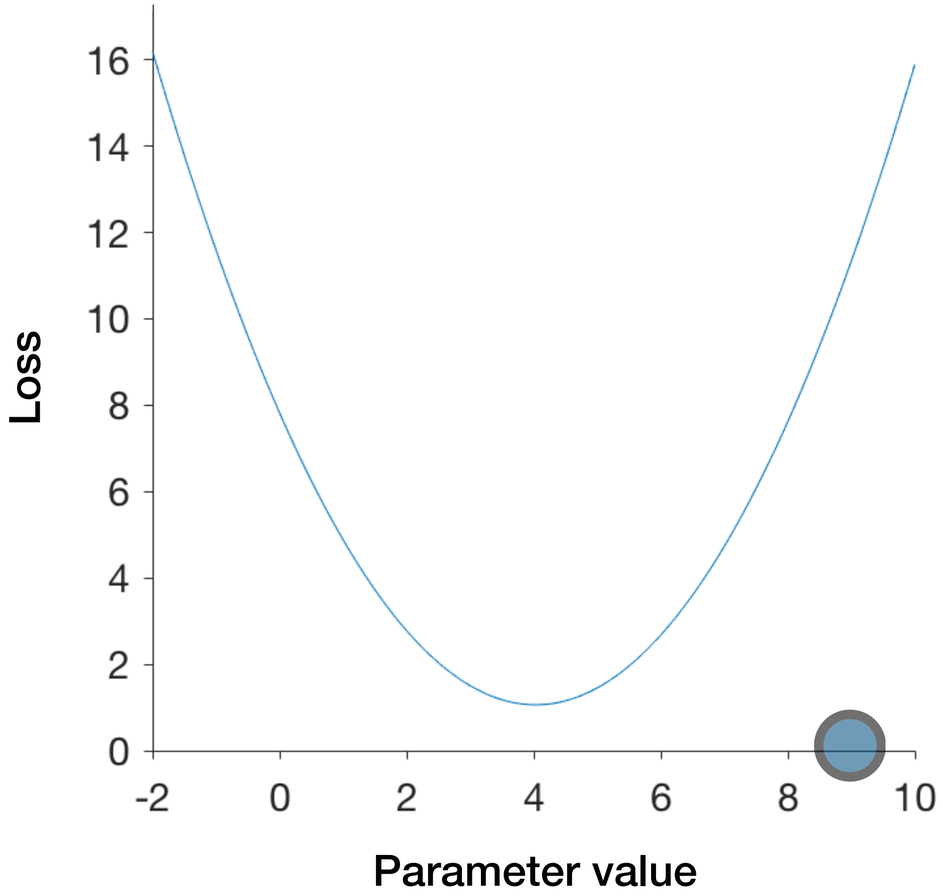
Loss landscape = loss for different parameter values



Gradient descent for one parameter

(a.k.a., “try something, then keep trying to make it better”)

Loss landscape = loss for different parameter values

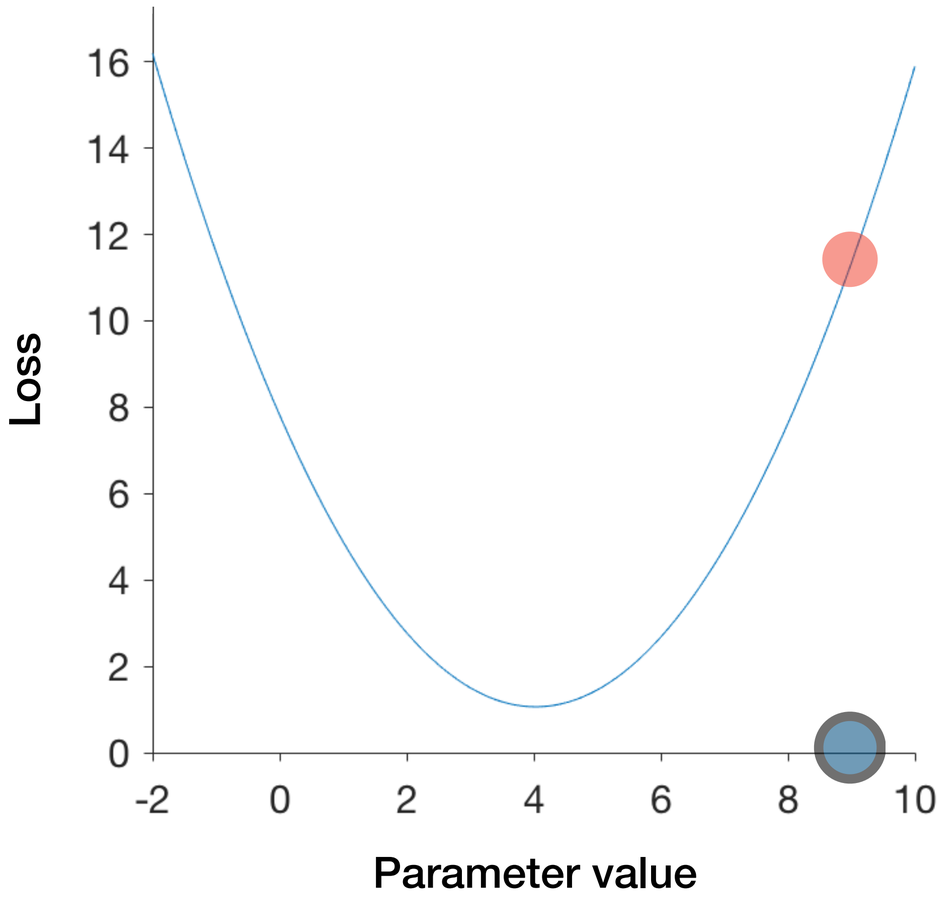


1. Guess a parameter value

Gradient descent for one parameter

(a.k.a., “try something, then keep trying to make it better”)

Loss landscape = loss for different parameter values

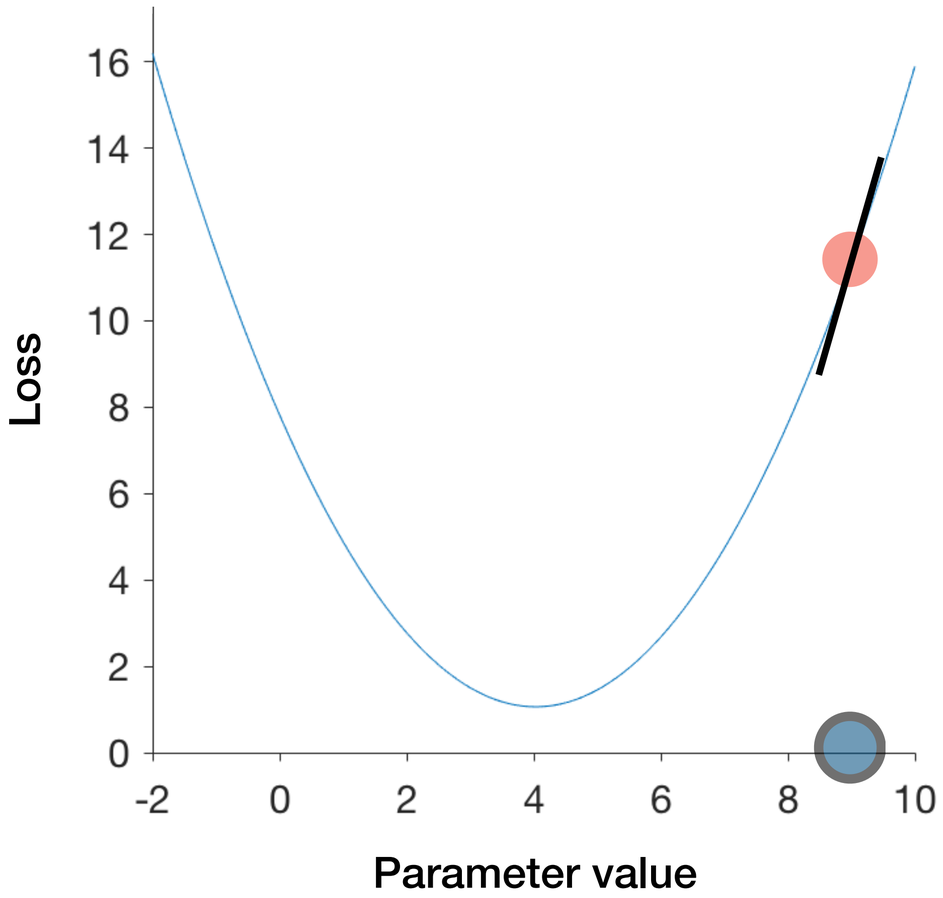


1. Guess a parameter value
2. Evaluate the loss

Gradient descent for one parameter

(a.k.a., “try something, then keep trying to make it better”)

Loss landscape = loss for different parameter values

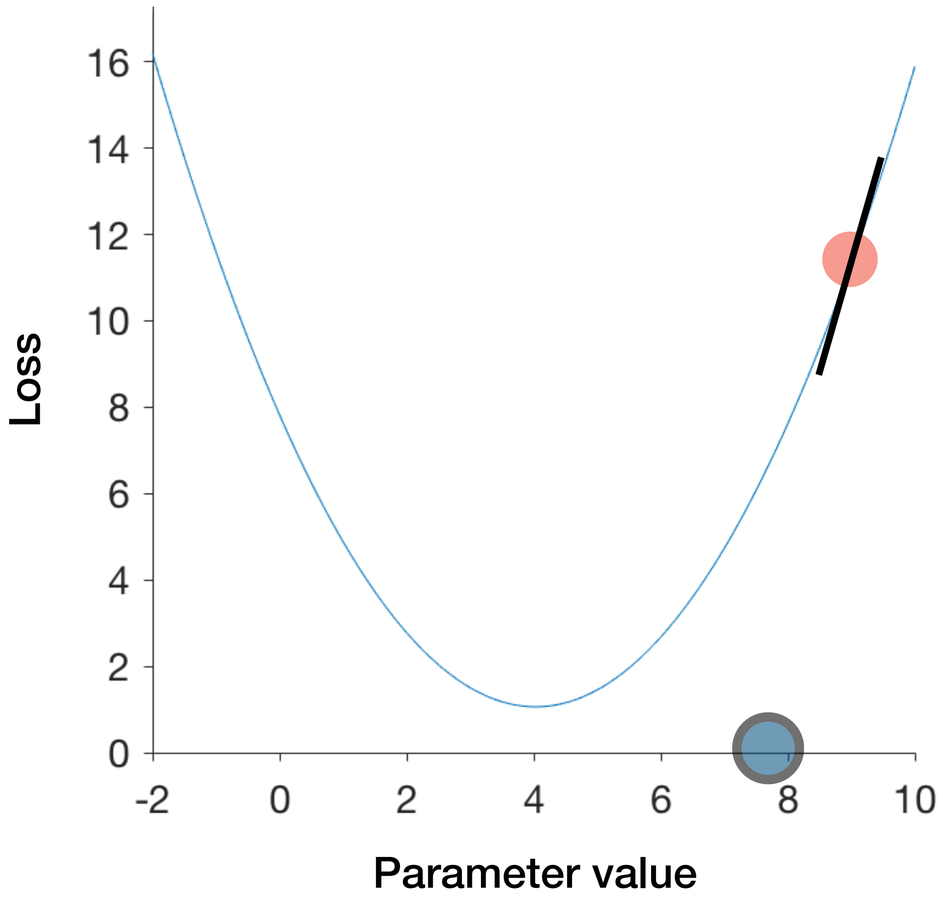


1. Guess a parameter value
2. Evaluate the loss
3. Calculate loss derivative w.r.t. parameter (gradient)

Gradient descent for one parameter

(a.k.a., “try something, then keep trying to make it better”)

Loss landscape = loss for different parameter values

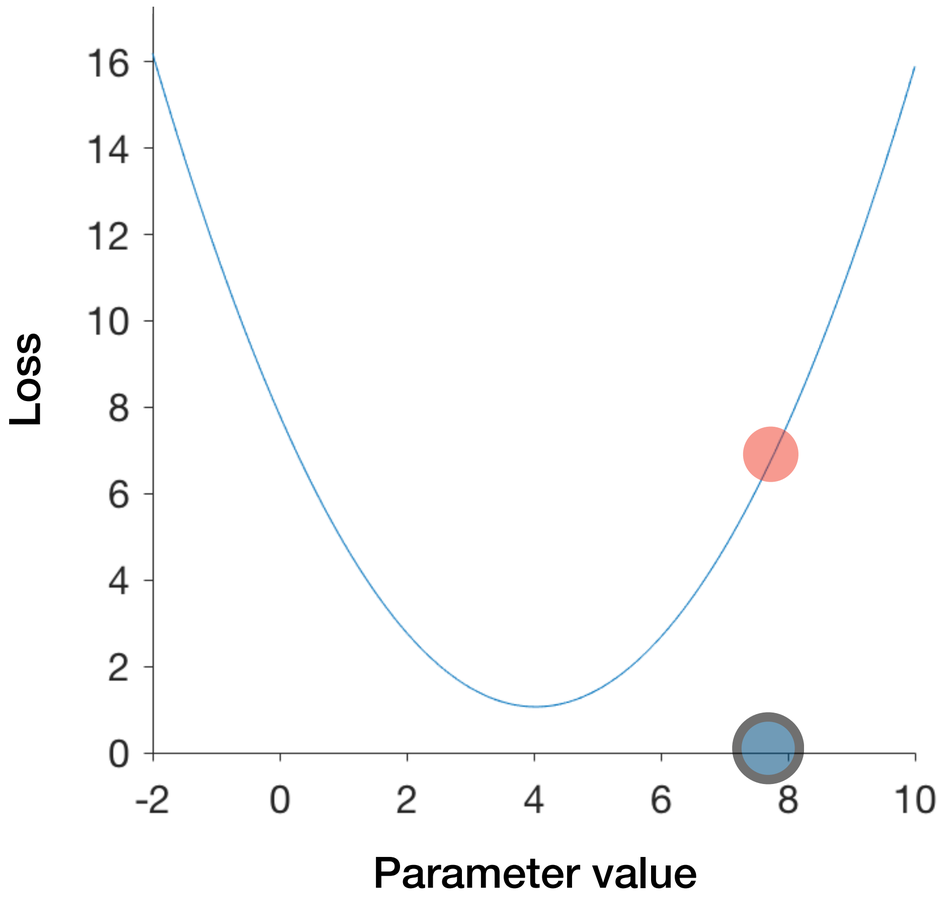


1. Guess a parameter value
2. Evaluate the loss
3. Calculate loss derivative w.r.t. parameter (gradient)
4. Step down the gradient

Gradient descent for one parameter

(a.k.a., “try something, then keep trying to make it better”)

Loss landscape = loss for different parameter values

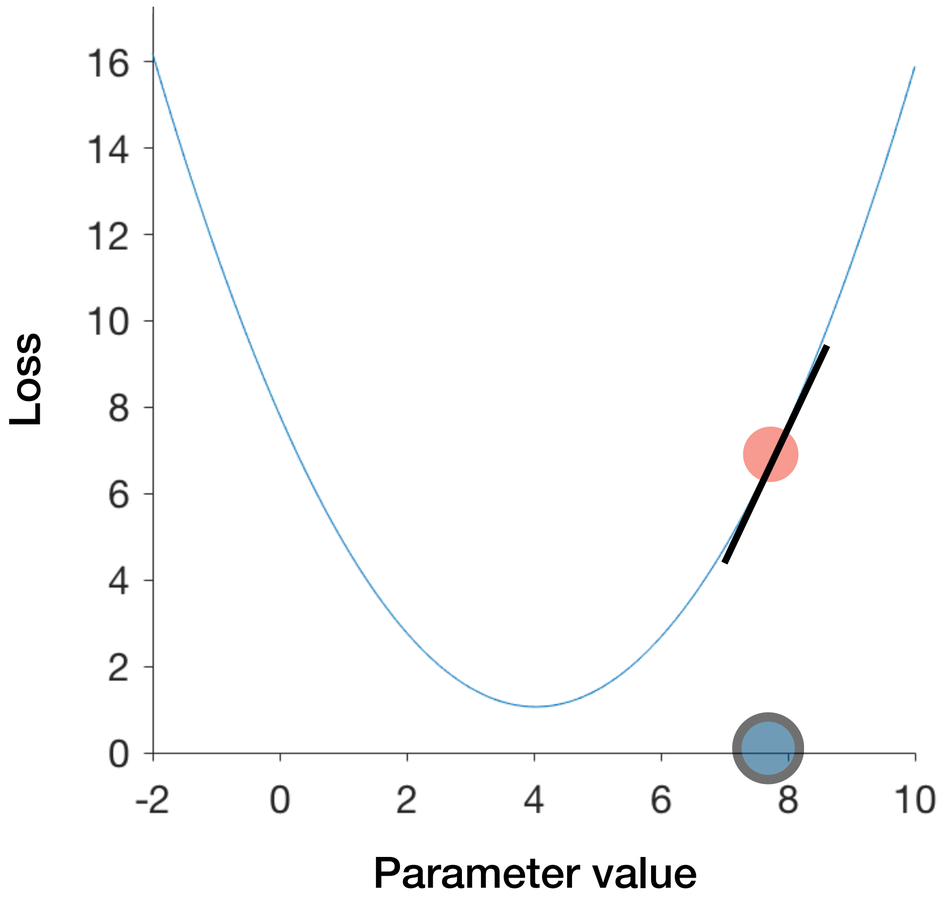


1. Guess a parameter value
2. Evaluate the loss
3. Calculate loss derivative w.r.t. parameter (gradient)
4. Step down the gradient
5. Repeat!

Gradient descent for one parameter

(a.k.a., “try something, then keep trying to make it better”)

Loss landscape = loss for different parameter values

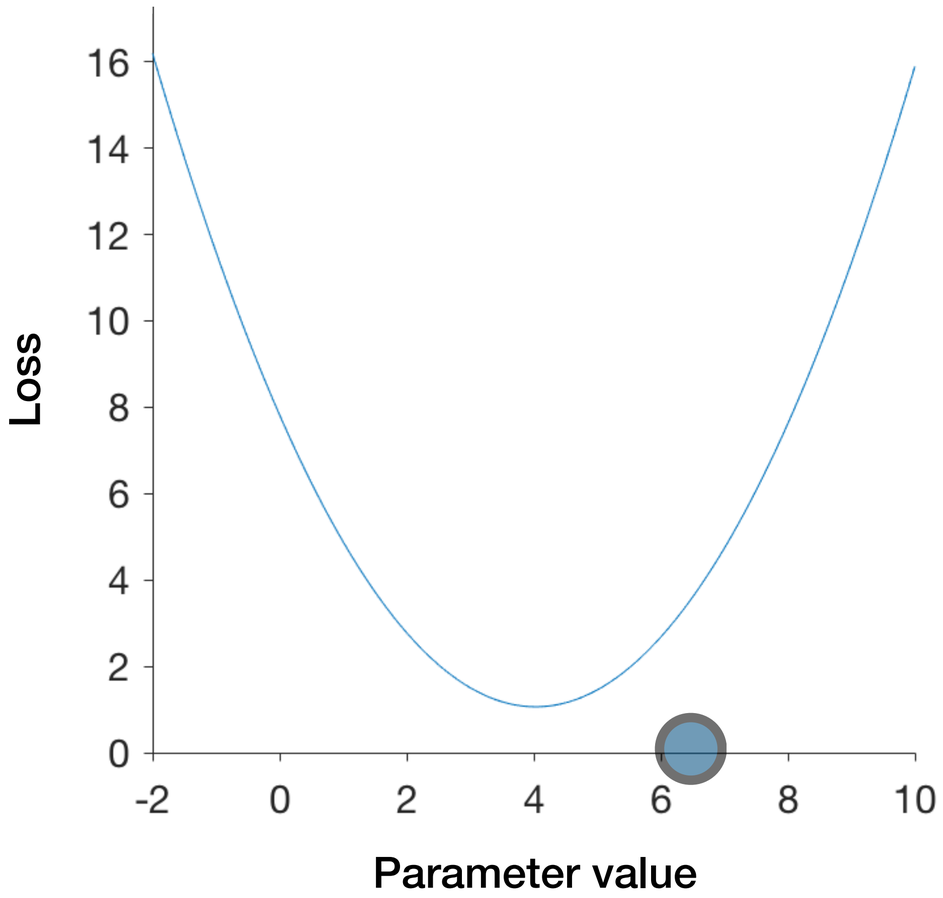


1. Guess a parameter value
2. Evaluate the loss
3. Calculate loss derivative w.r.t. parameter (gradient)
4. Step down the gradient
5. Repeat!

Gradient descent for one parameter

(a.k.a., “try something, then keep trying to make it better”)

Loss landscape = loss for different parameter values

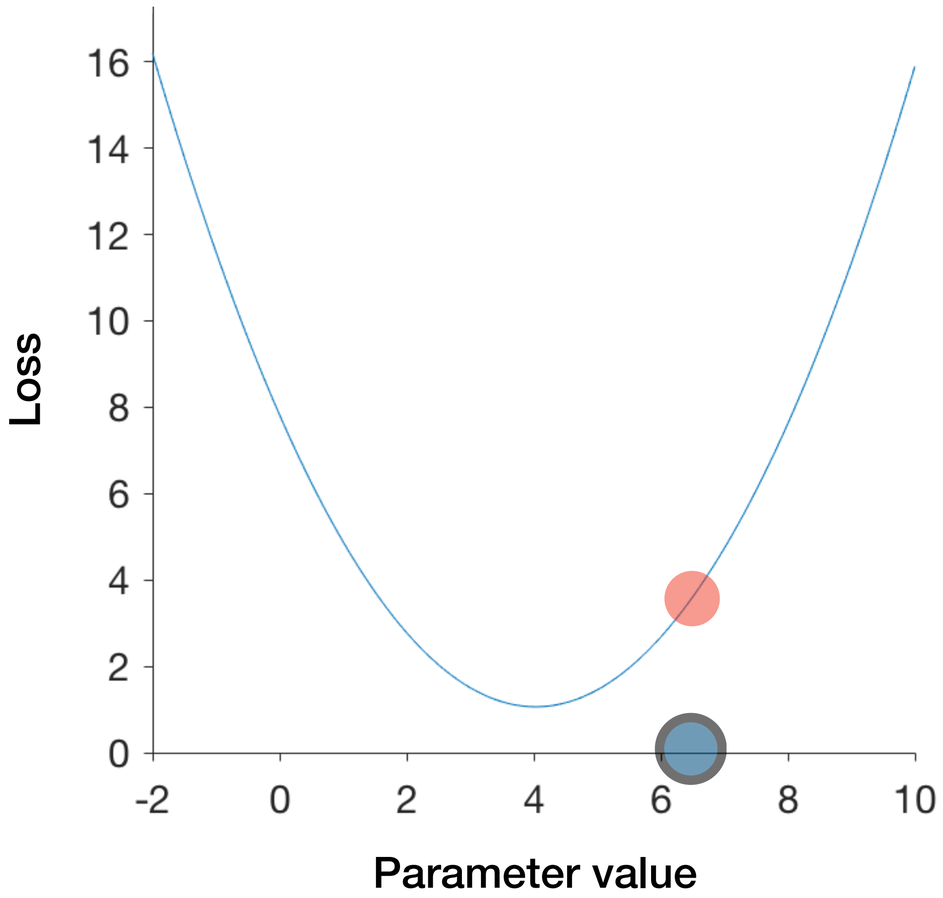


1. Guess a parameter value
2. Evaluate the loss
3. Calculate loss derivative w.r.t. parameter (gradient)
4. Step down the gradient
5. Repeat!

Gradient descent for one parameter

(a.k.a., “try something, then keep trying to make it better”)

Loss landscape = loss for different parameter values

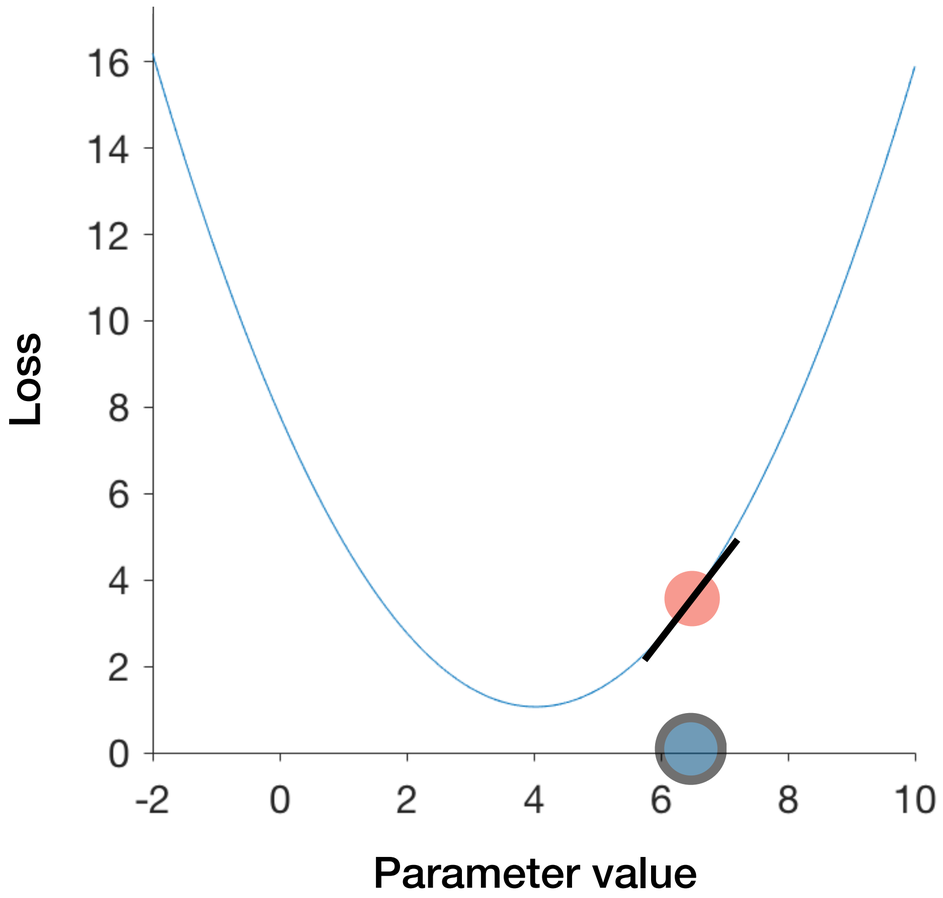


1. Guess a parameter value
2. Evaluate the loss
3. Calculate loss derivative w.r.t. parameter (gradient)
4. Step down the gradient
5. Repeat!

Gradient descent for one parameter

(a.k.a., “try something, then keep trying to make it better”)

Loss landscape = loss for different parameter values

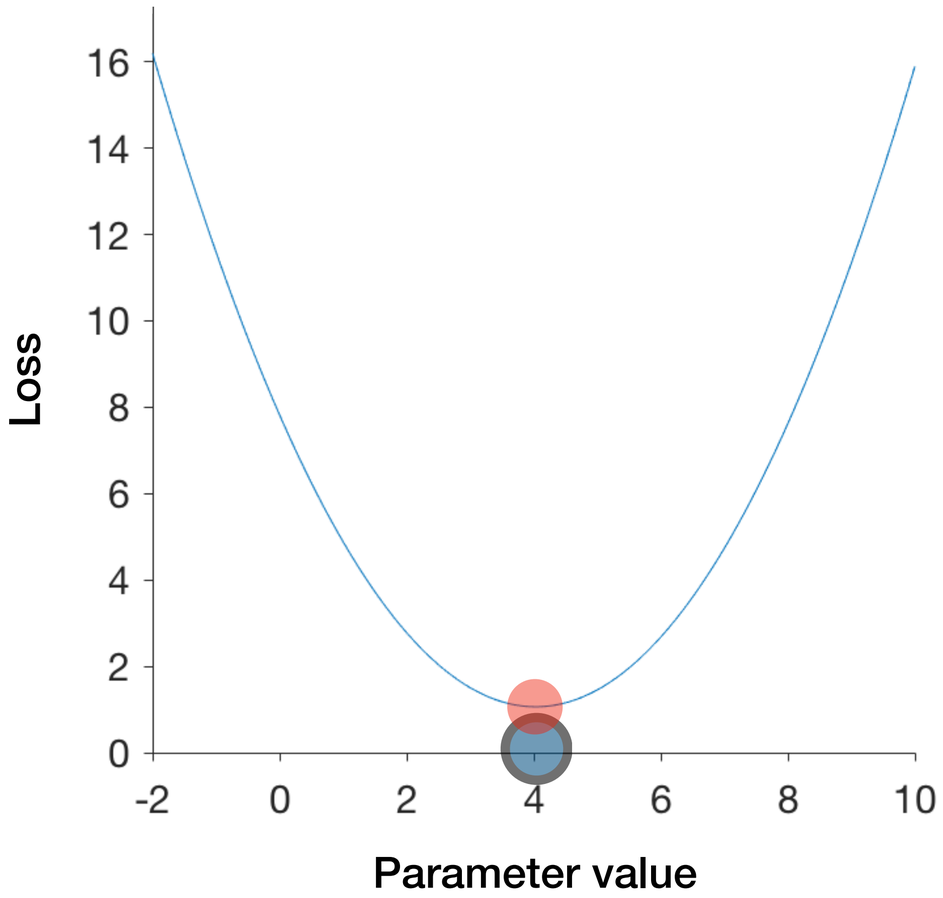


1. Guess a parameter value
2. Evaluate the loss
3. Calculate loss derivative w.r.t. parameter (gradient)
4. Step down the gradient
5. Repeat!

Gradient descent for one parameter

(a.k.a., “try something, then keep trying to make it better”)

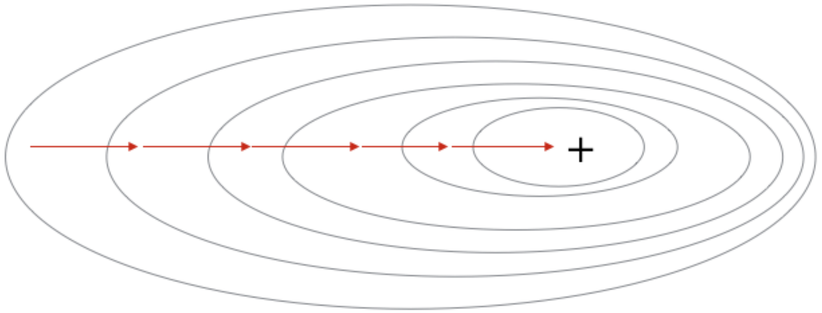
Loss landscape = loss for different parameter values



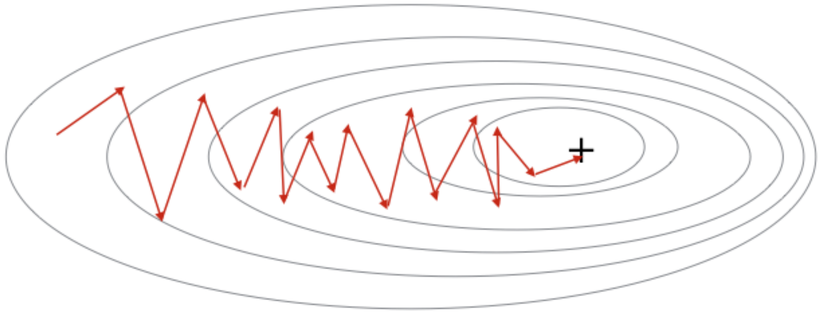
1. Guess a parameter value
2. Evaluate the loss
3. Calculate loss derivative w.r.t. parameter (gradient)
4. Step down the gradient
5. Repeat!

Gradient Descent and Stochastic Gradient Descent

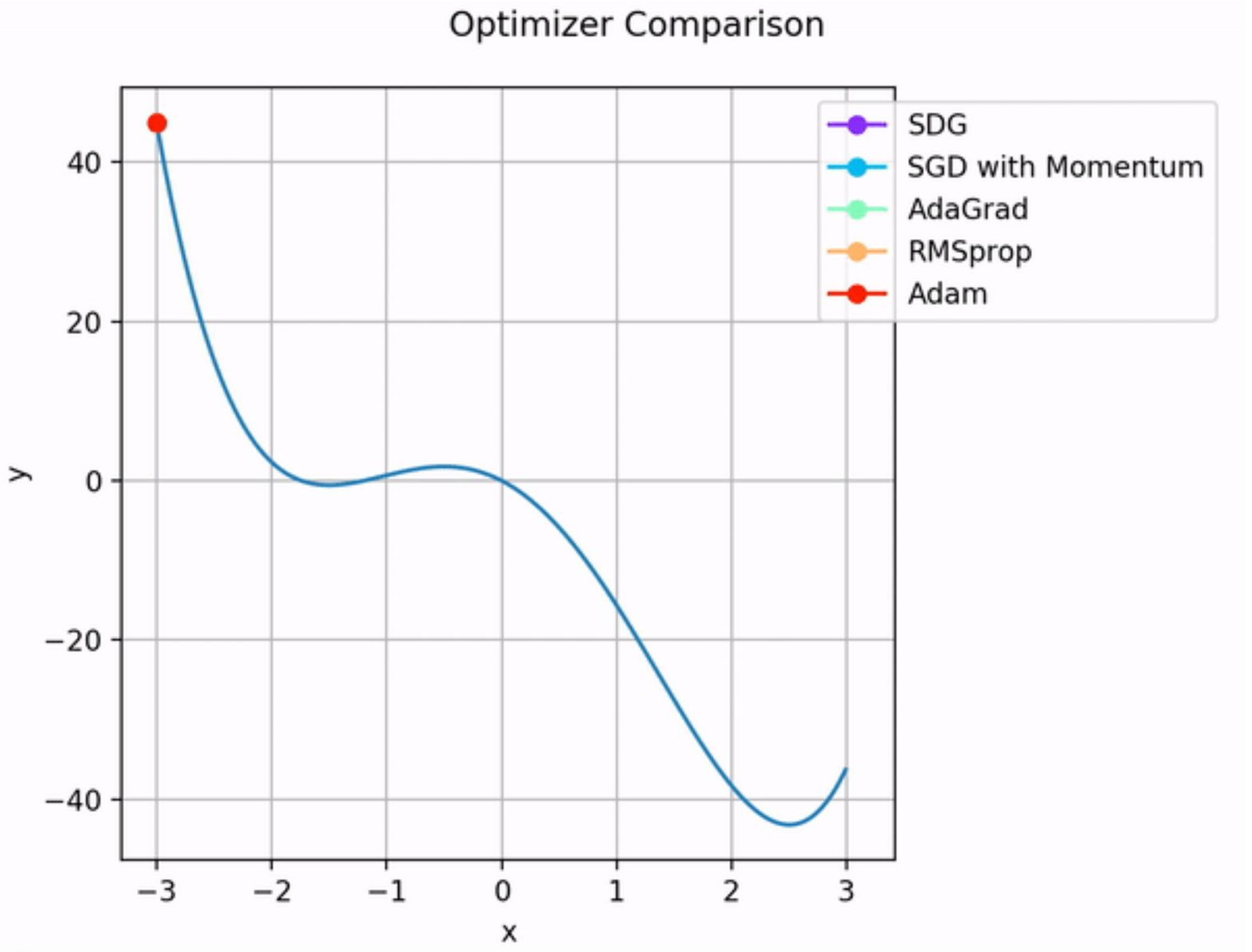
Gradient Descent



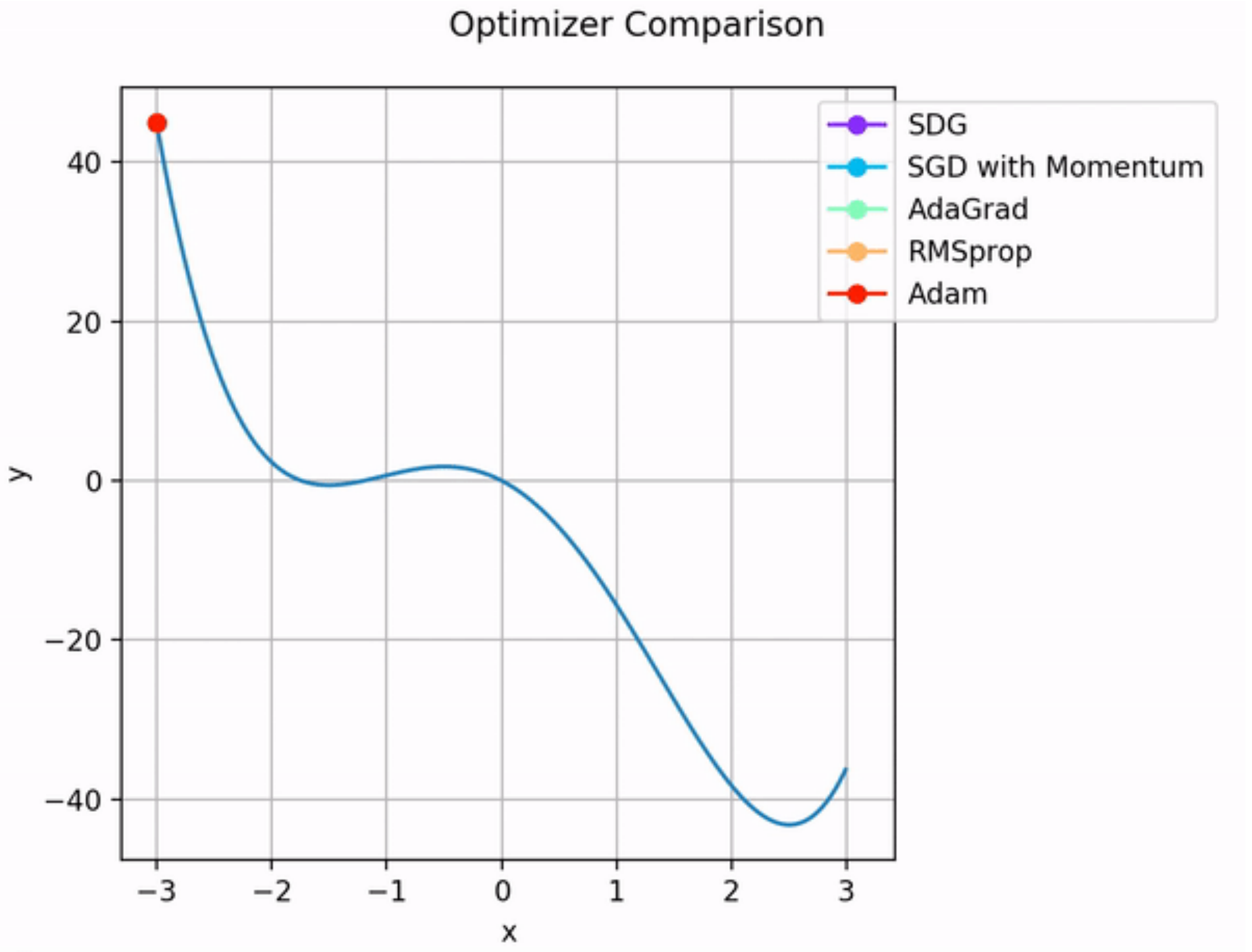
Stochastic Gradient Descent



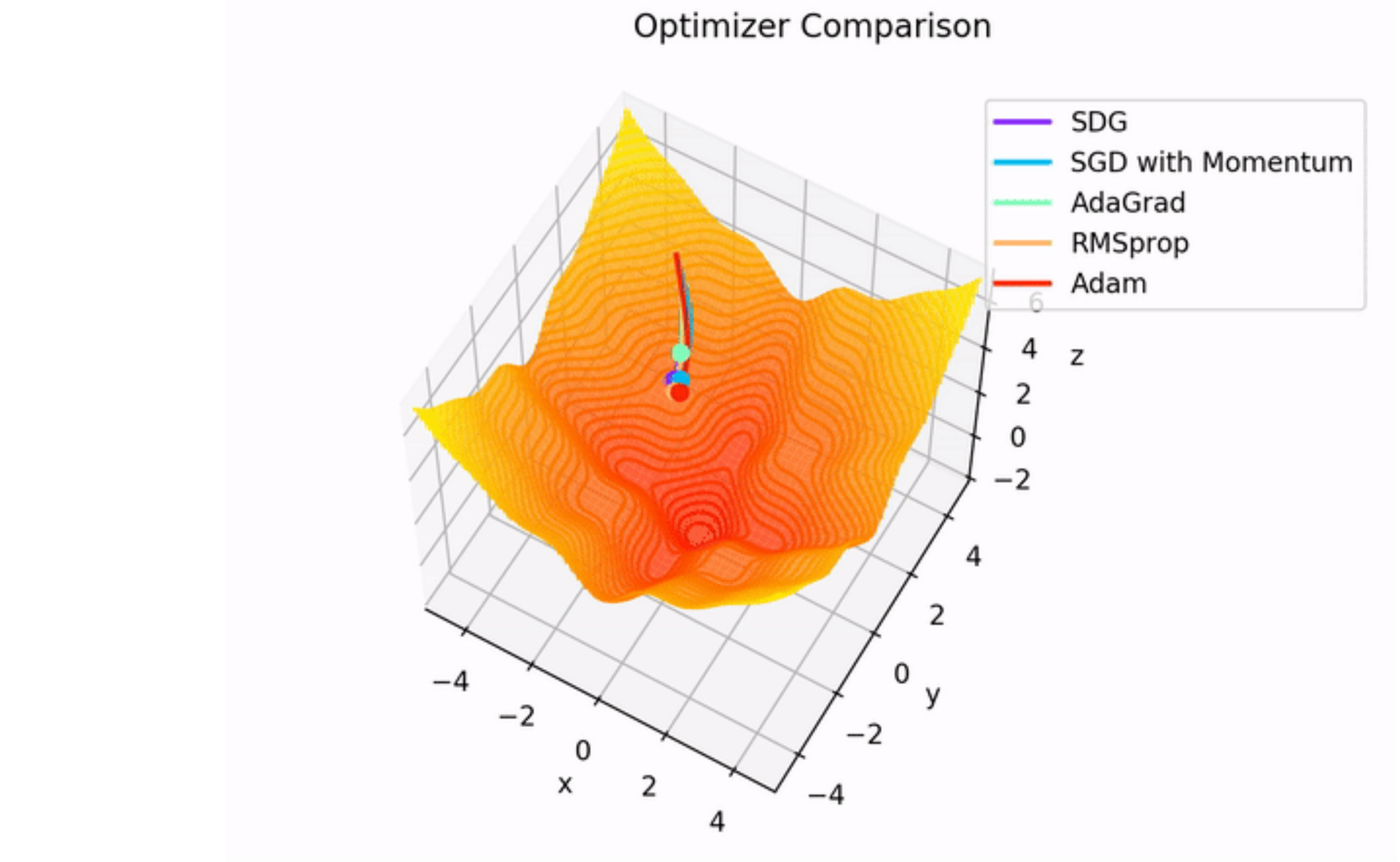
https://towardsdatascience.com/complete-guide-to-adam-optimization-1e5f29532c3d



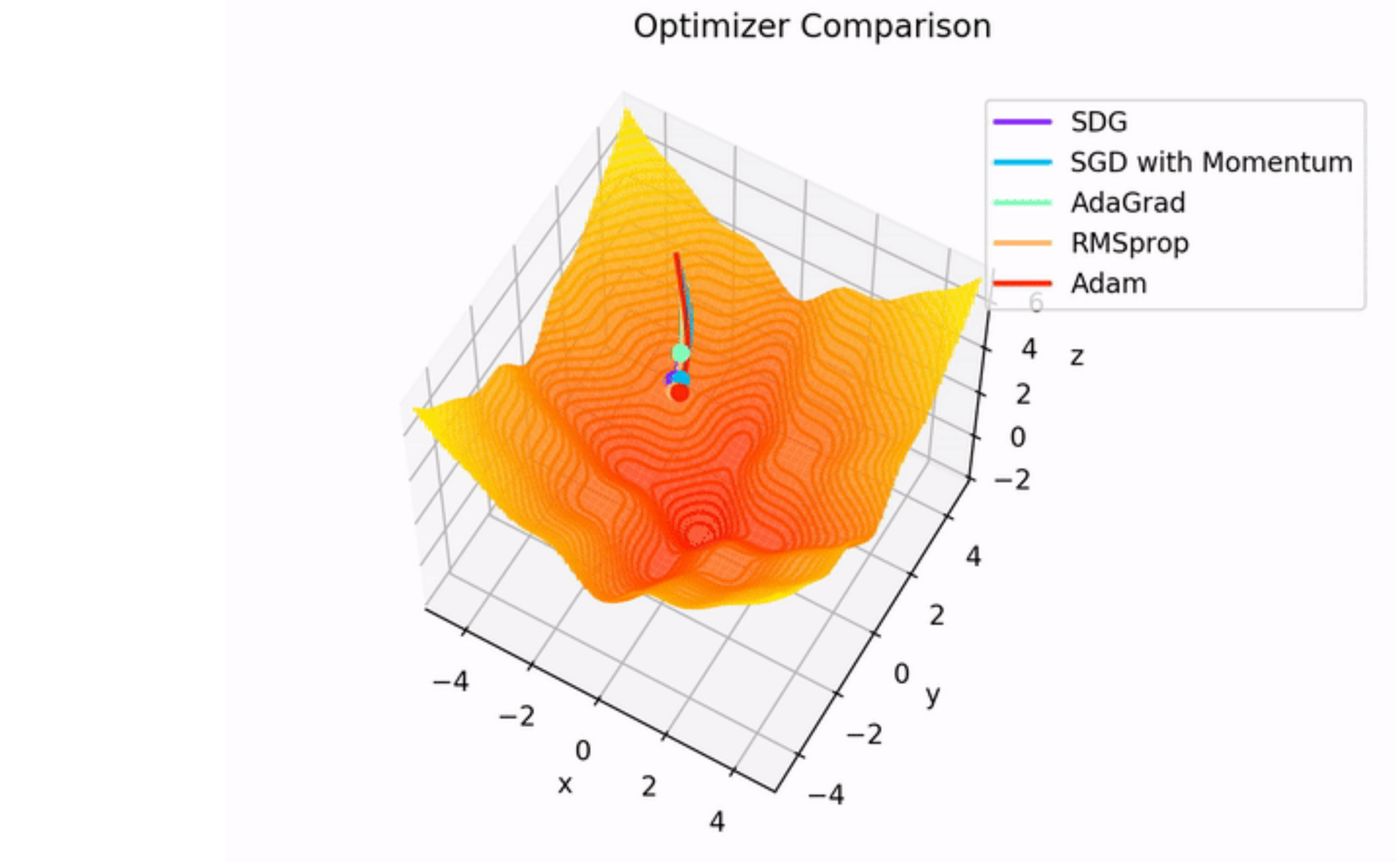
https://towardsdatascience.com/complete-guide-to-adam-optimization-1e5f29532c3d



https://towardsdatascience.com/complete-guide-to-adam-optimization-1e5f29532c3d



https://towardsdatascience.com/complete-guide-to-adam-optimization-1e5f29532c3d



https://towardsdatascience.com/complete-guide-to-adam-optimization-1e5f29532c3d

High-D problems

→ 3

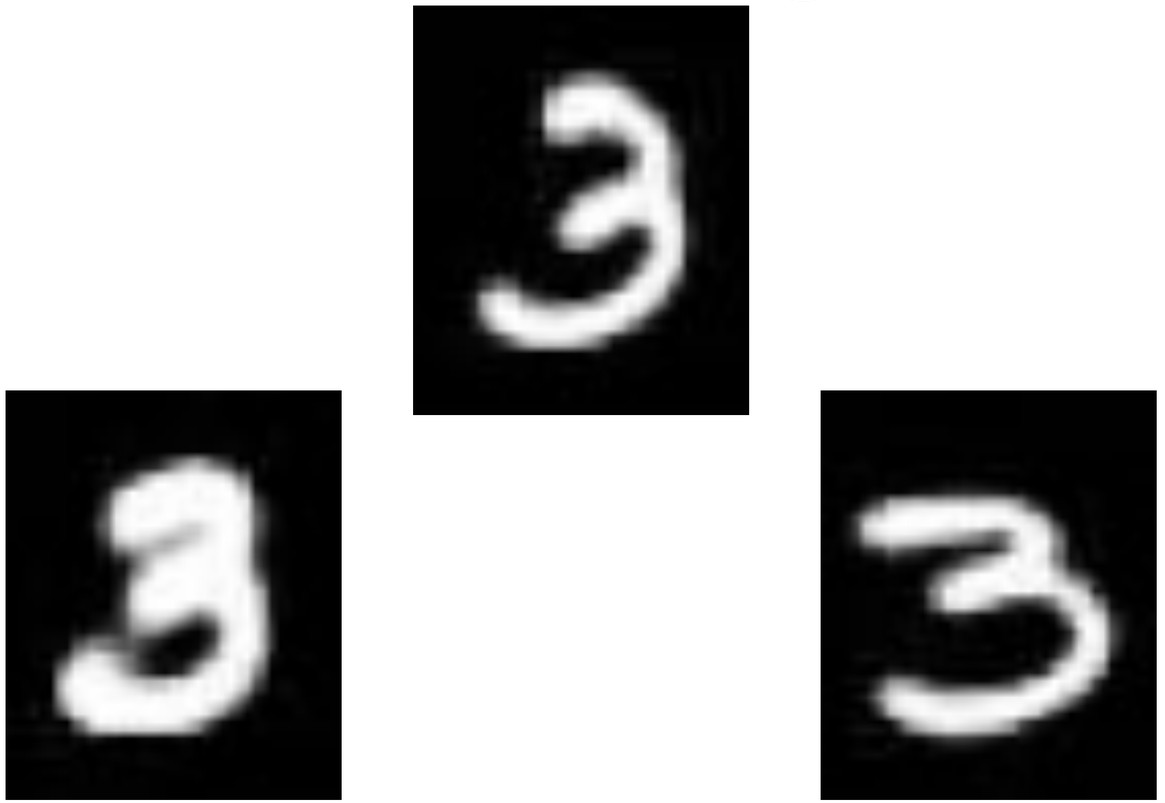
MNIST digits-
http://yann.lecun.com/exdb/mnist/

High-D problems

→ 3

MNIST digits-
http://yann.lecun.com/exdb/mnist/

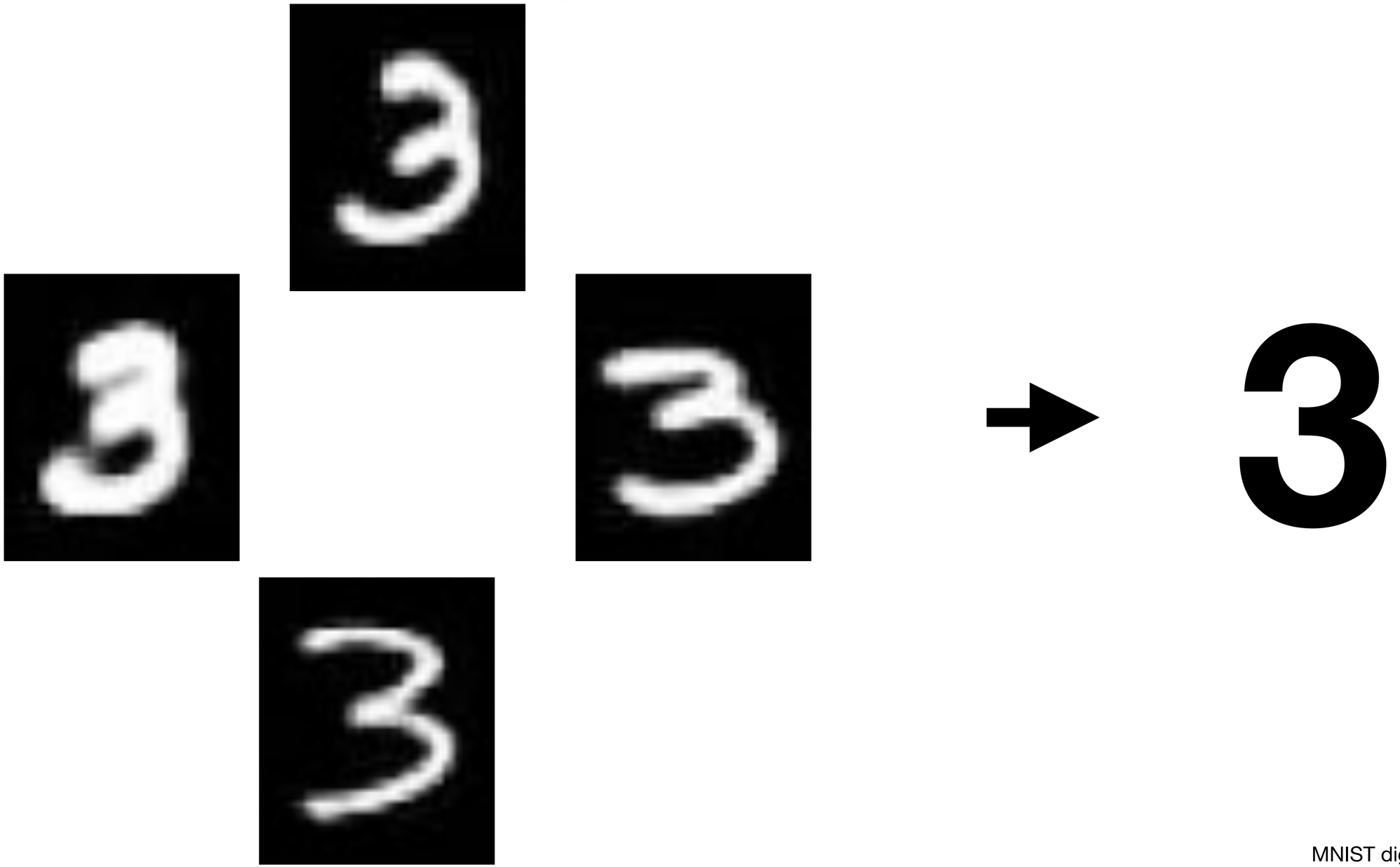
High-D problems



→ 3

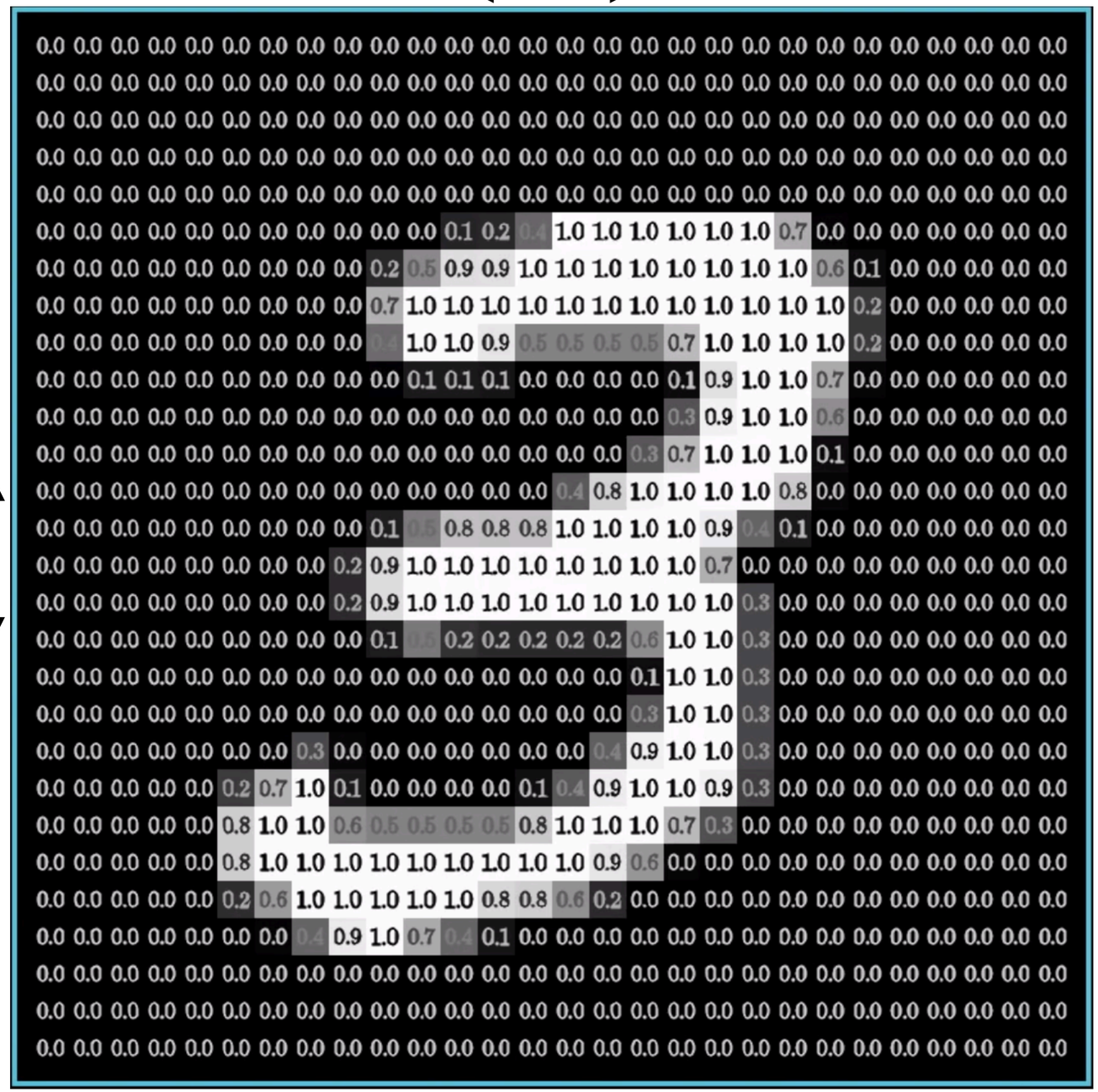
MNIST digits-
http://yann.lecun.com/exdb/mnist/

High-D problems



MNIST digits-
http://yann.lecun.com/exdb/mnist/

28 pixels

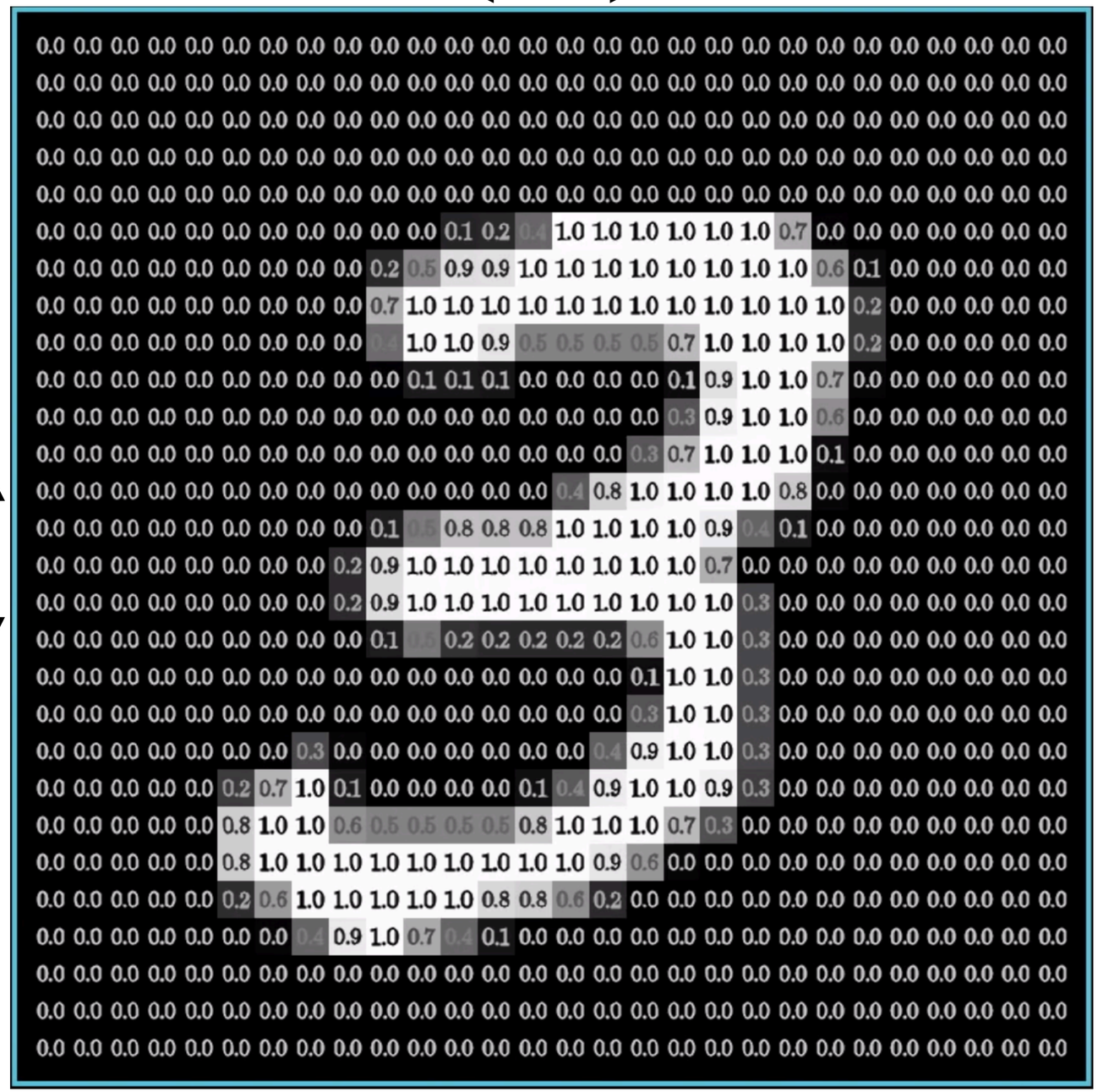


28 pixels

MNIST - 784 total pixels
each ranges from 0 to 1

28 pixels

28 pixels

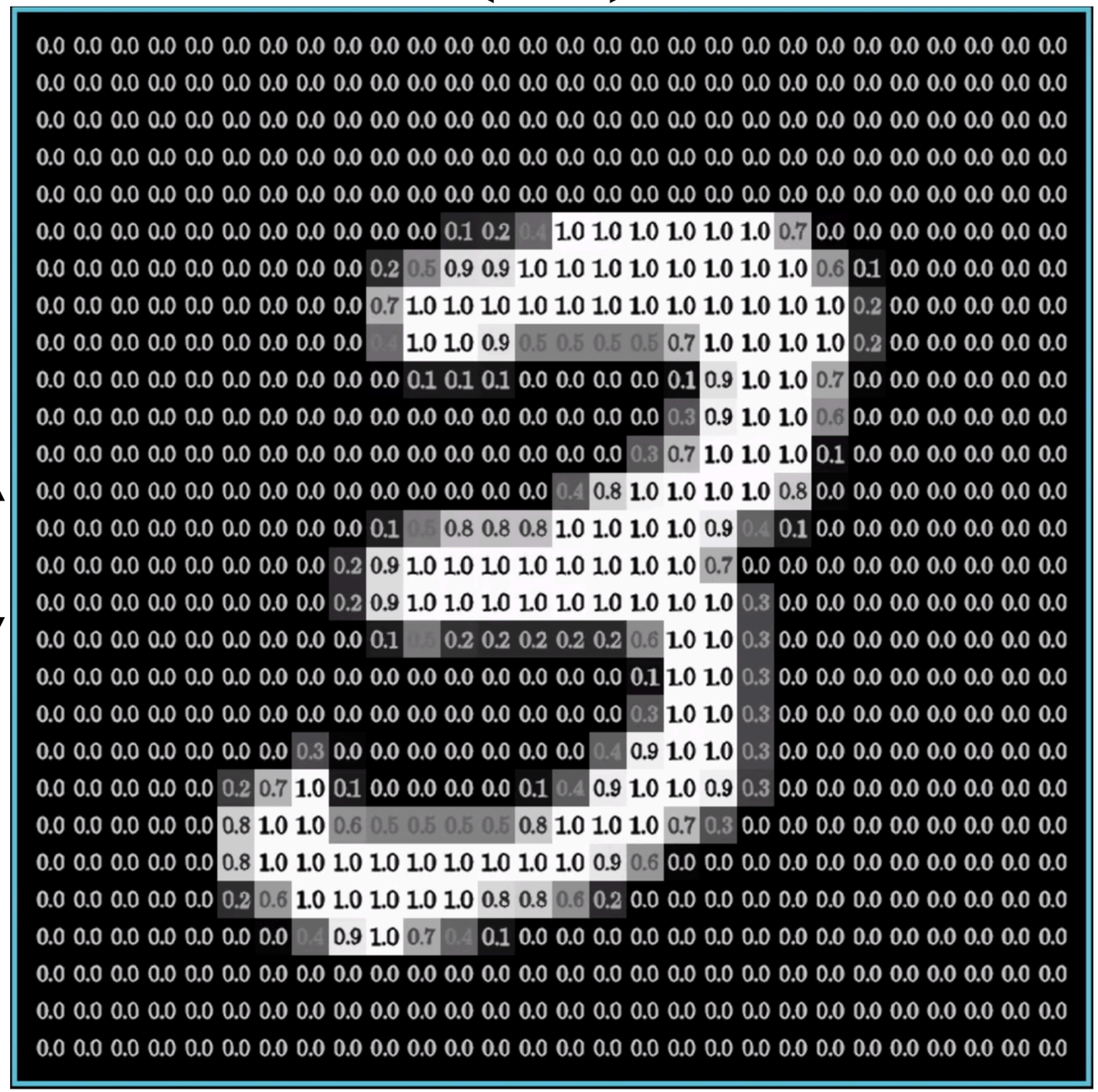


MNIST - 784 total pixels
each ranges from 0 to 1

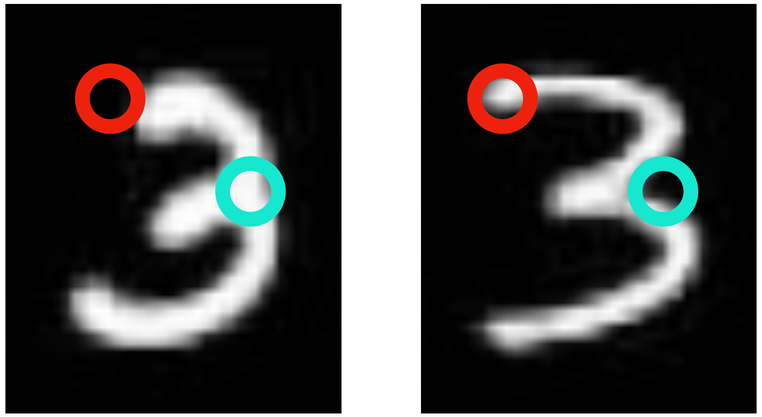
3Blue1Brown (YouTube)
But what is a Neural Network? | Deep learning, chapter 1

28 pixels

28 pixels



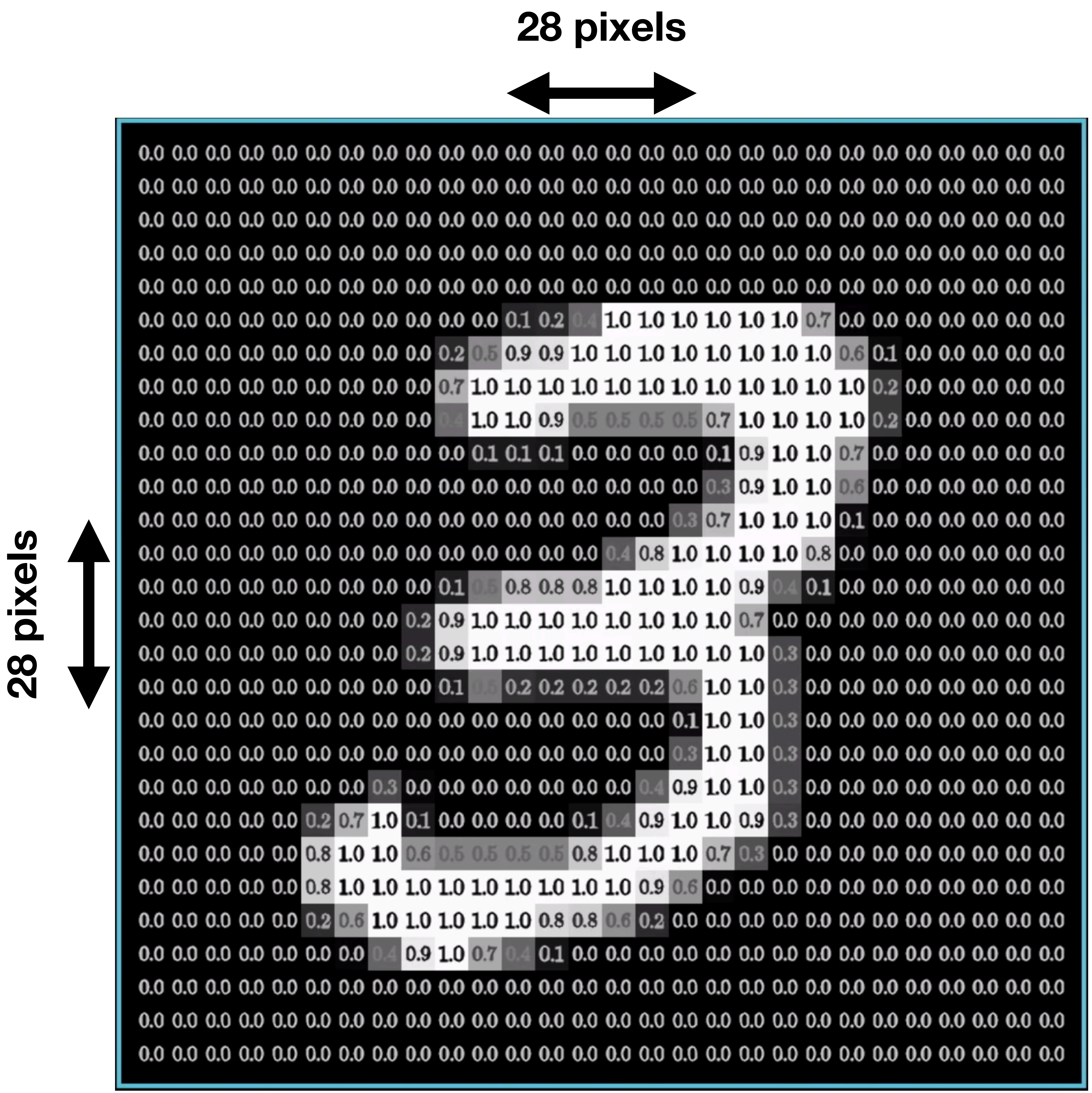
MNIST - 784 total pixels each ranges from 0 to 1



Specific pixel values are very different between images

3Blue1Brown (YouTube)

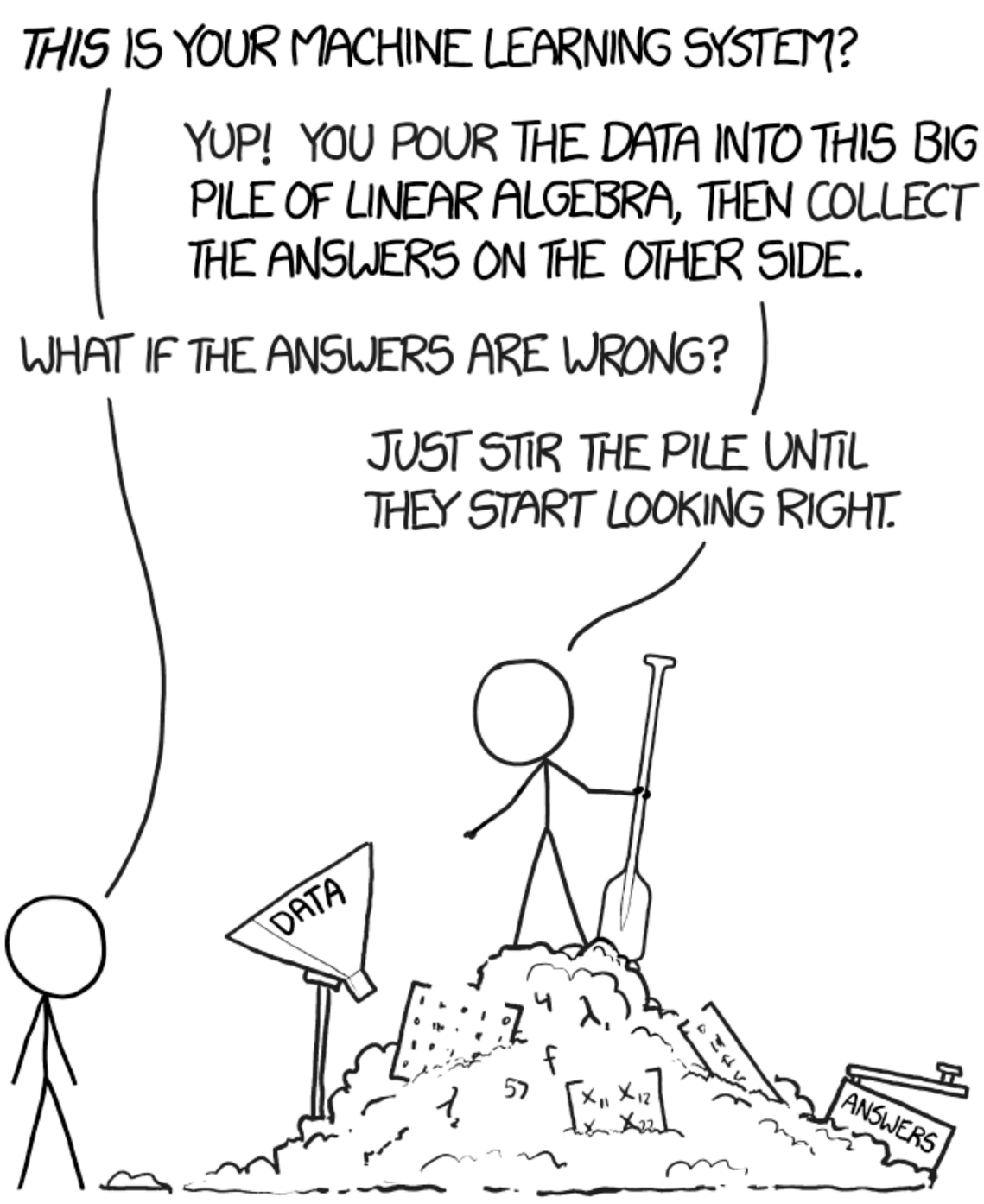
But what is a Neural Network? | Deep learning, chapter 1



MNIST - 784 total pixels each ranges from 0 to 1

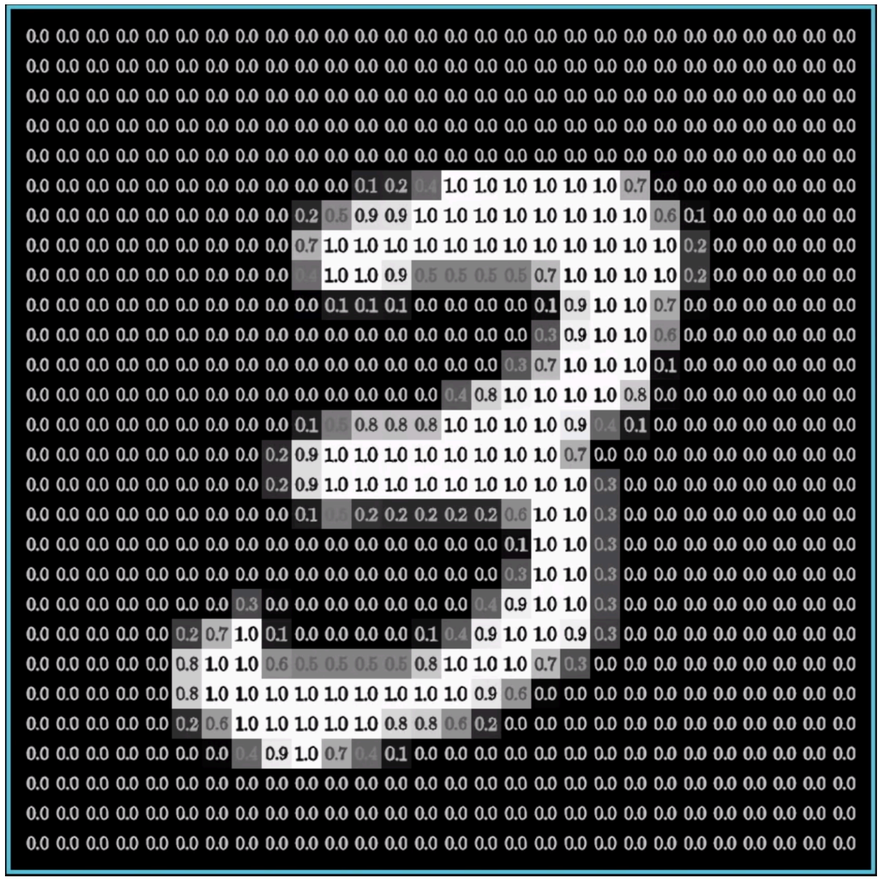
Specific pixel values are very different between images

3Blue1Brown (YouTube)
But what is a Neural Network? | Deep learning, chapter 1



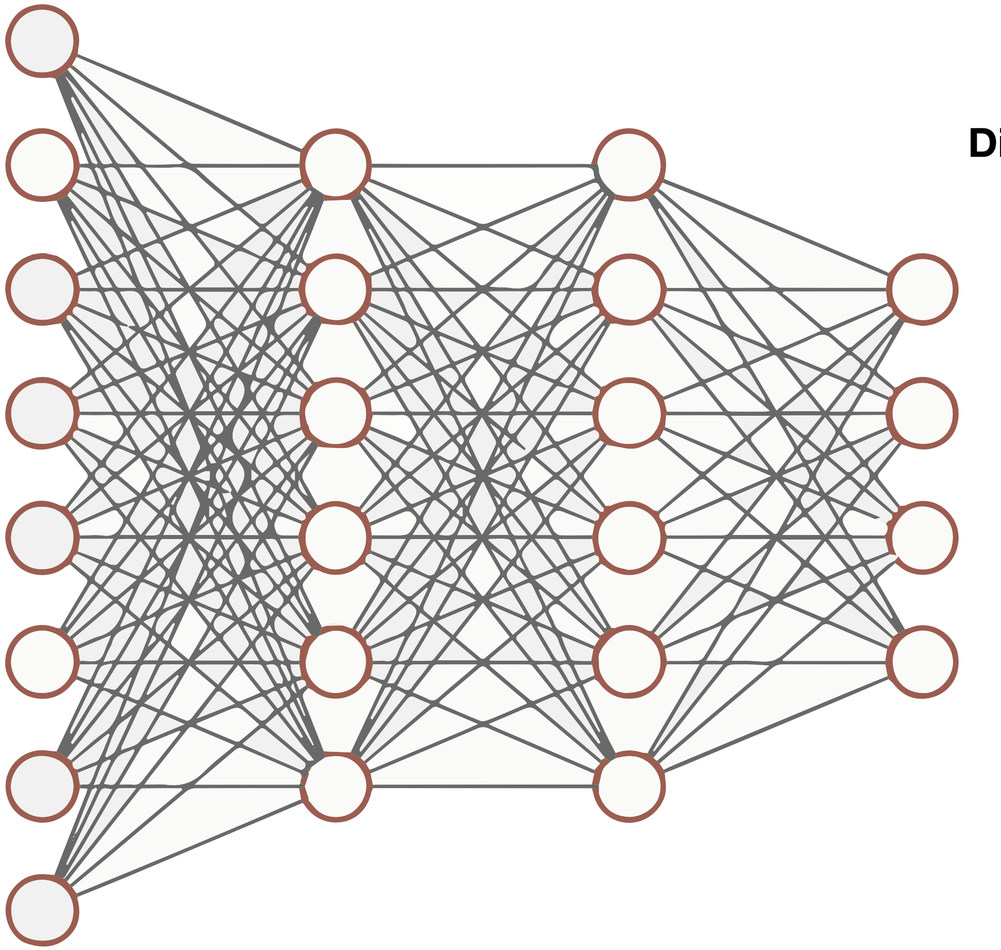
https://xkcd.com/1838/

MLP for digit classification



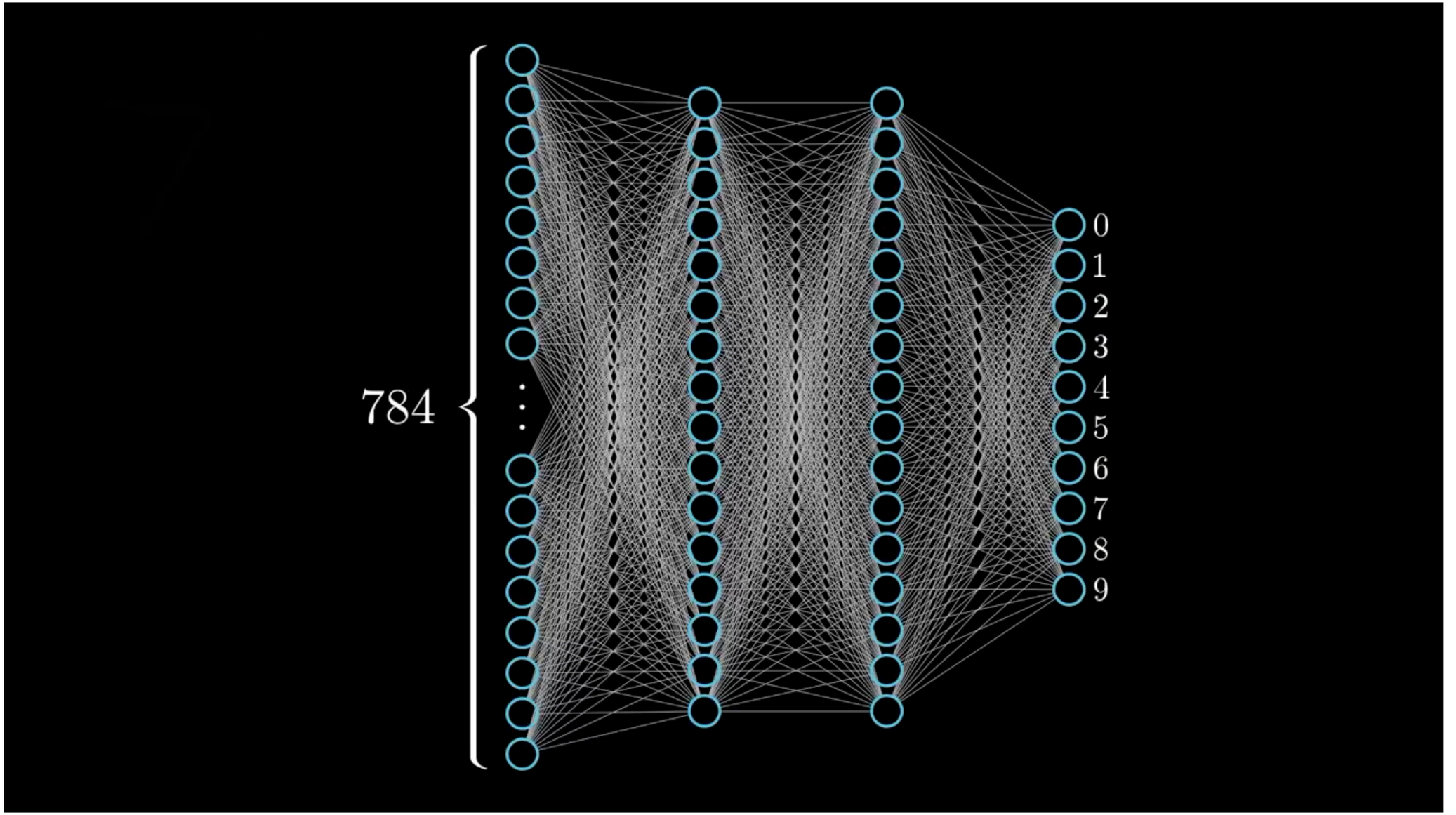
Pixel values

0.0
0.0
0.5
...
1.0
0.3
0.2



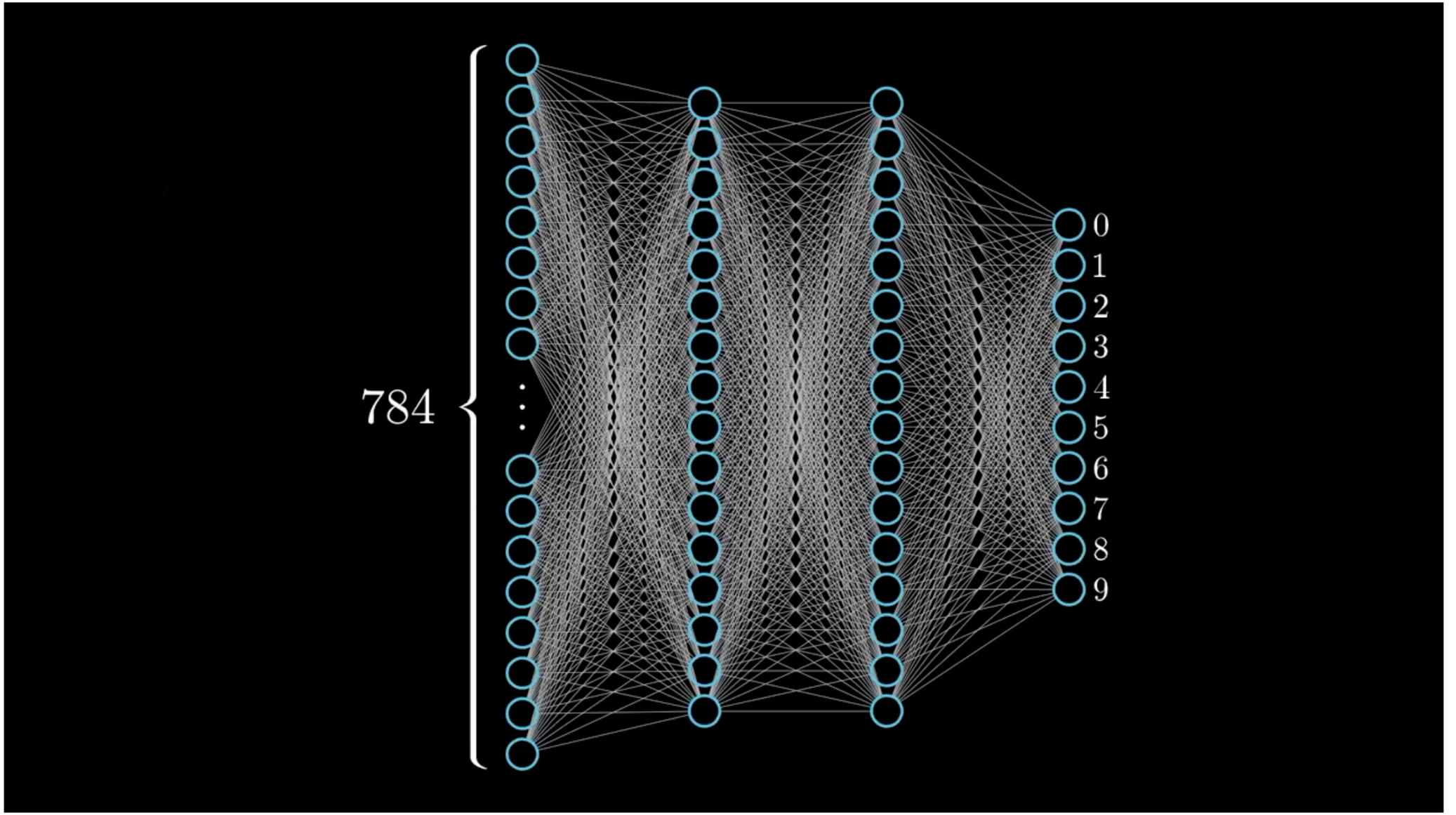
Digit probability

0 ?
1 ?
2 ?
3 ?
...
9 ?



3Blue1Brown (YouTube)

But what is a Neural Network? | Deep learning, chapter 1

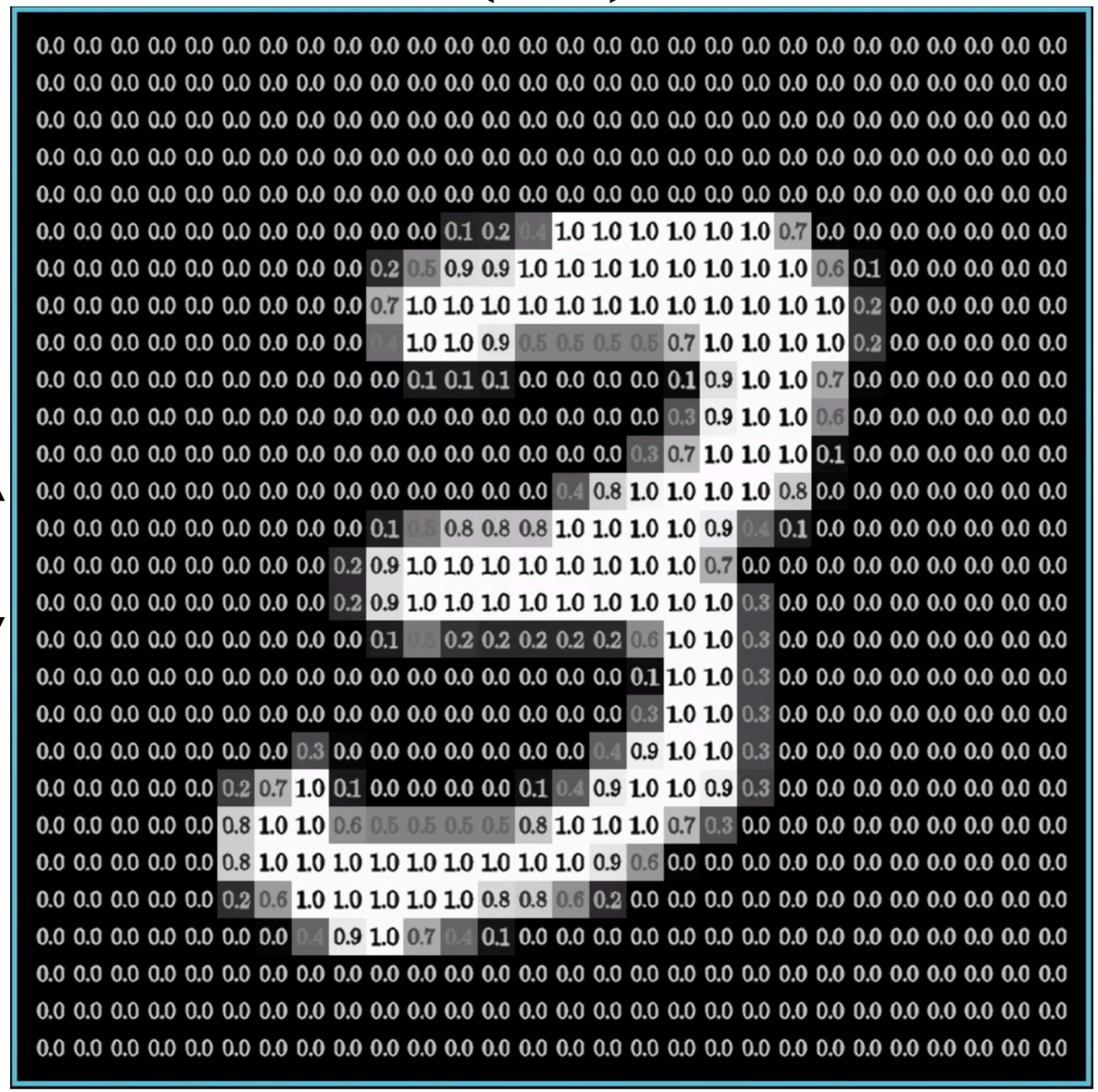


3Blue1Brown (YouTube)

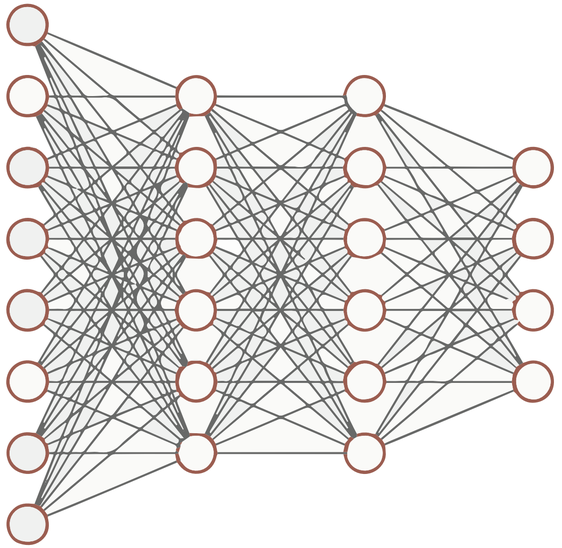
But what is a Neural Network? | Deep learning, chapter 1

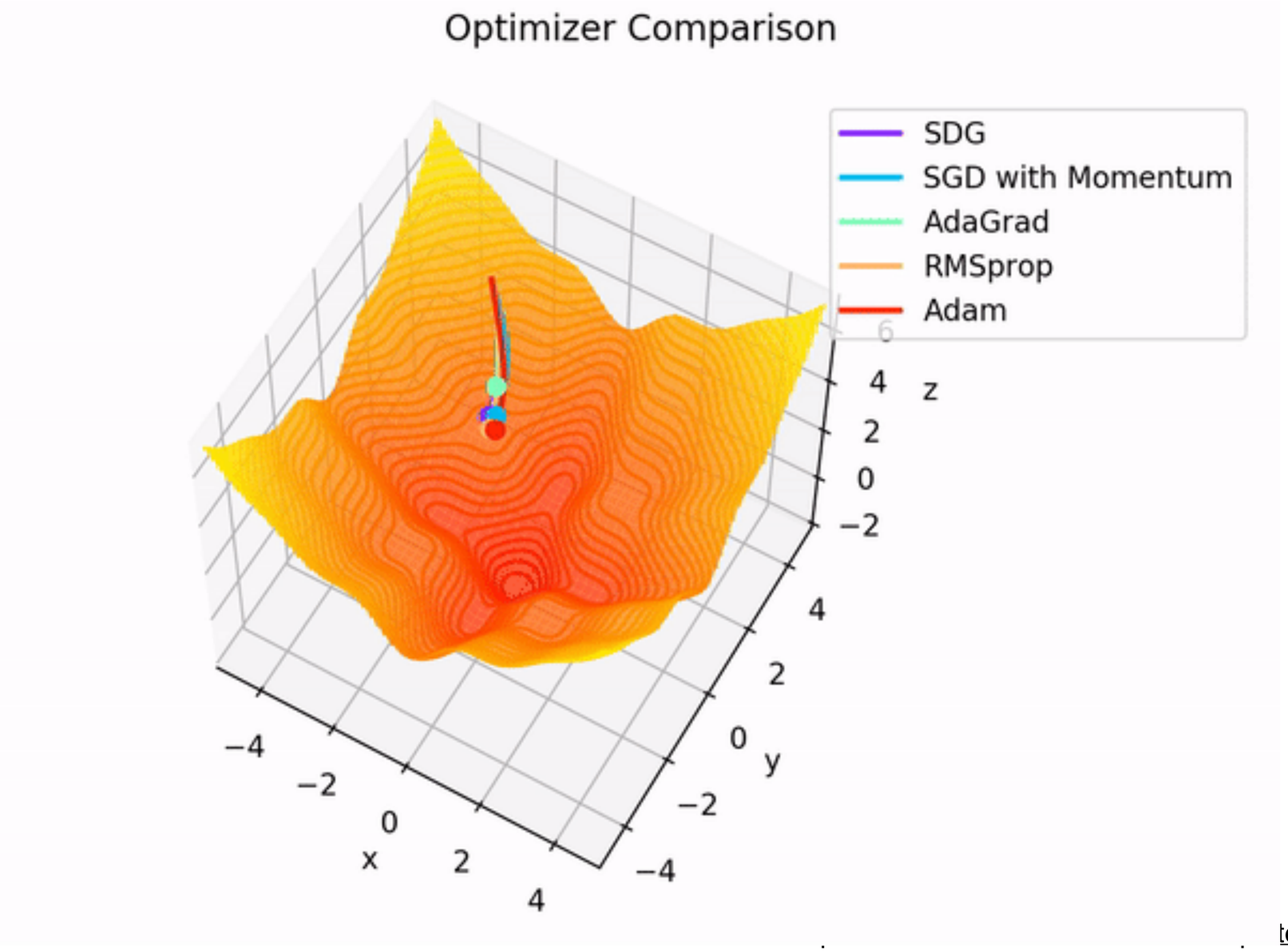
28 pixels

28 pixels

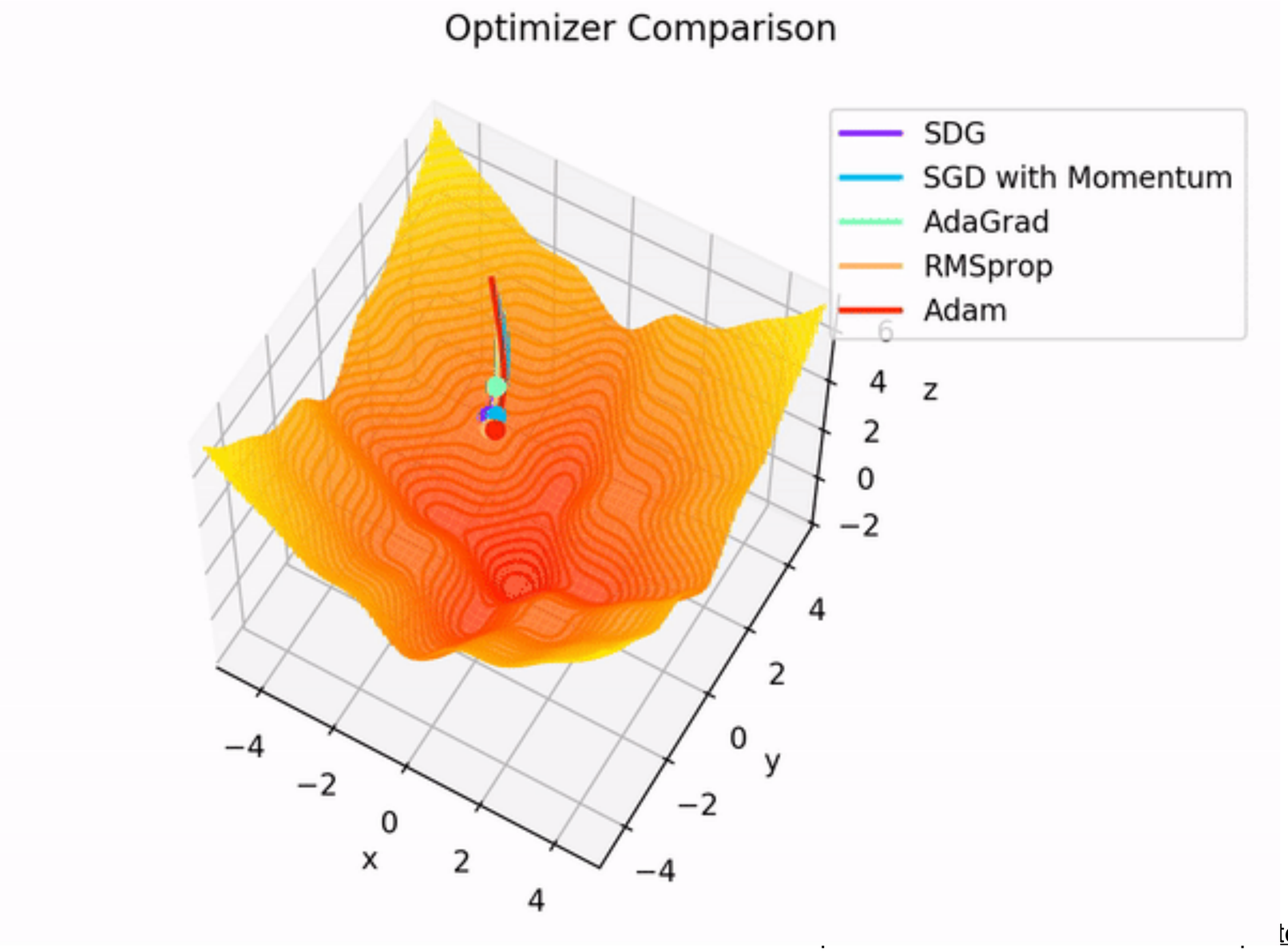


784-dimensional input





te-guide-to-adam-optimization-1e5f29532c3d



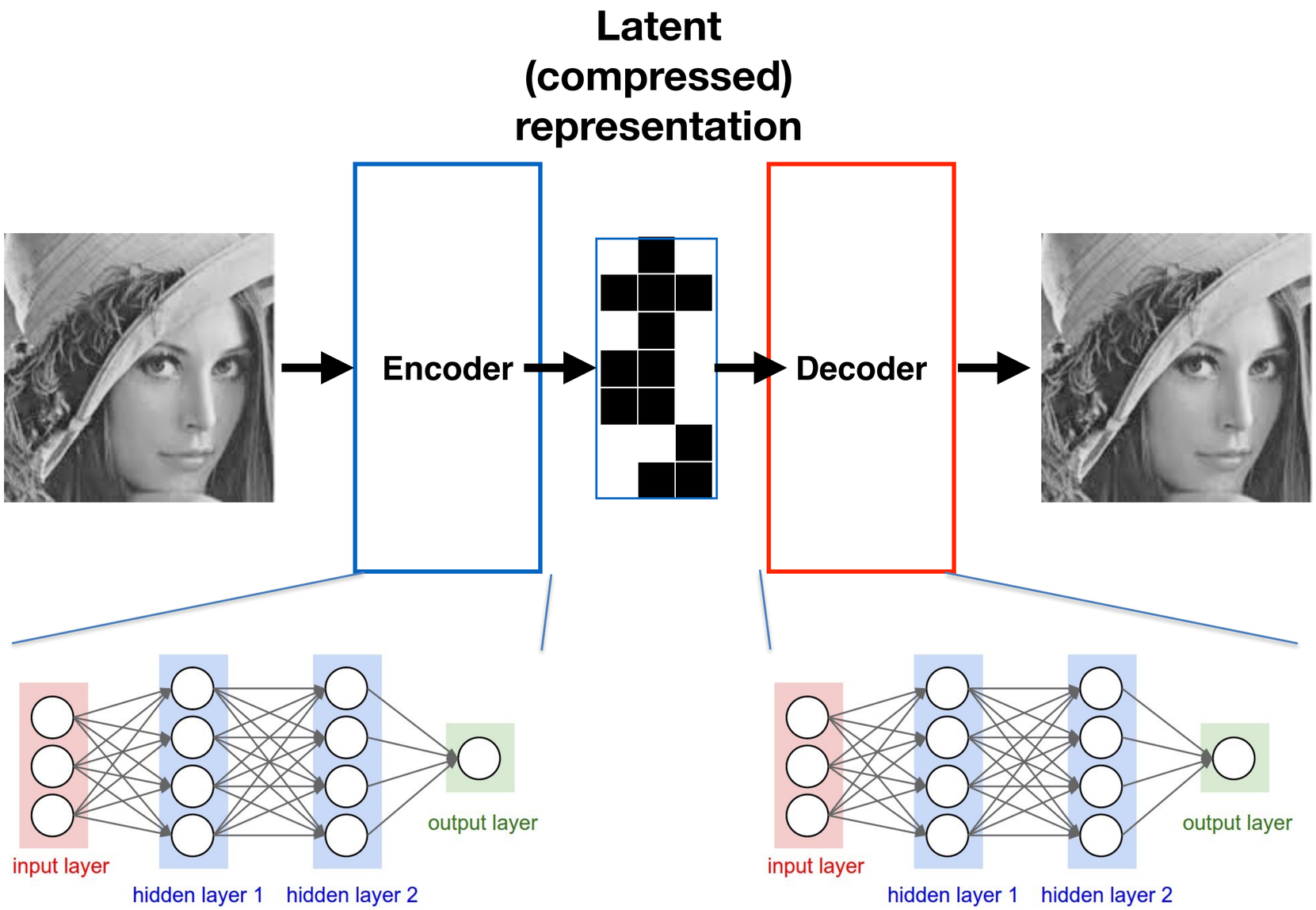
te-guide-to-adam-optimization-1e5f29532c3d

What you'll hear about in this lecture

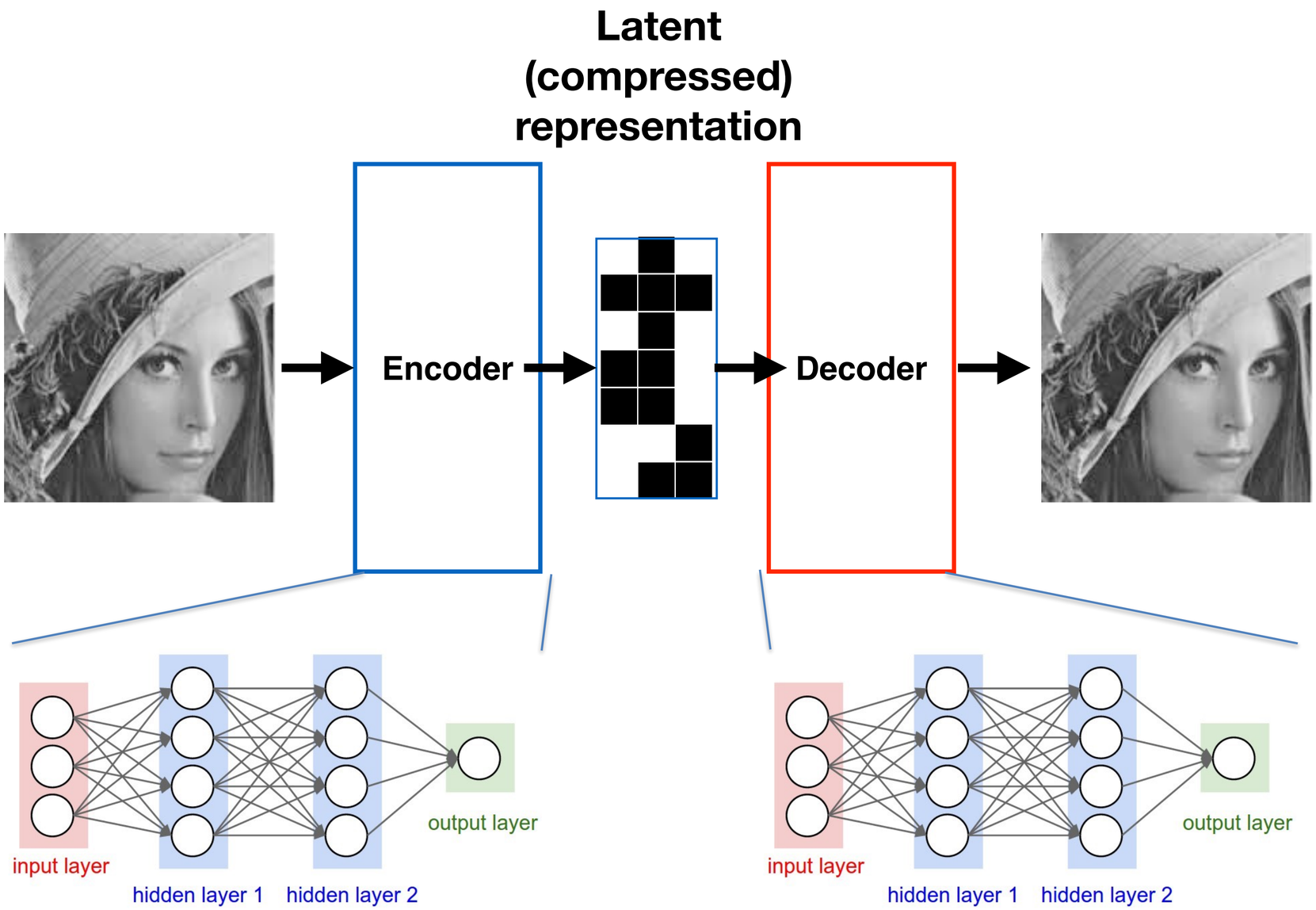
Neural network basics

Deep autoencoders

Intro to neural population dynamics

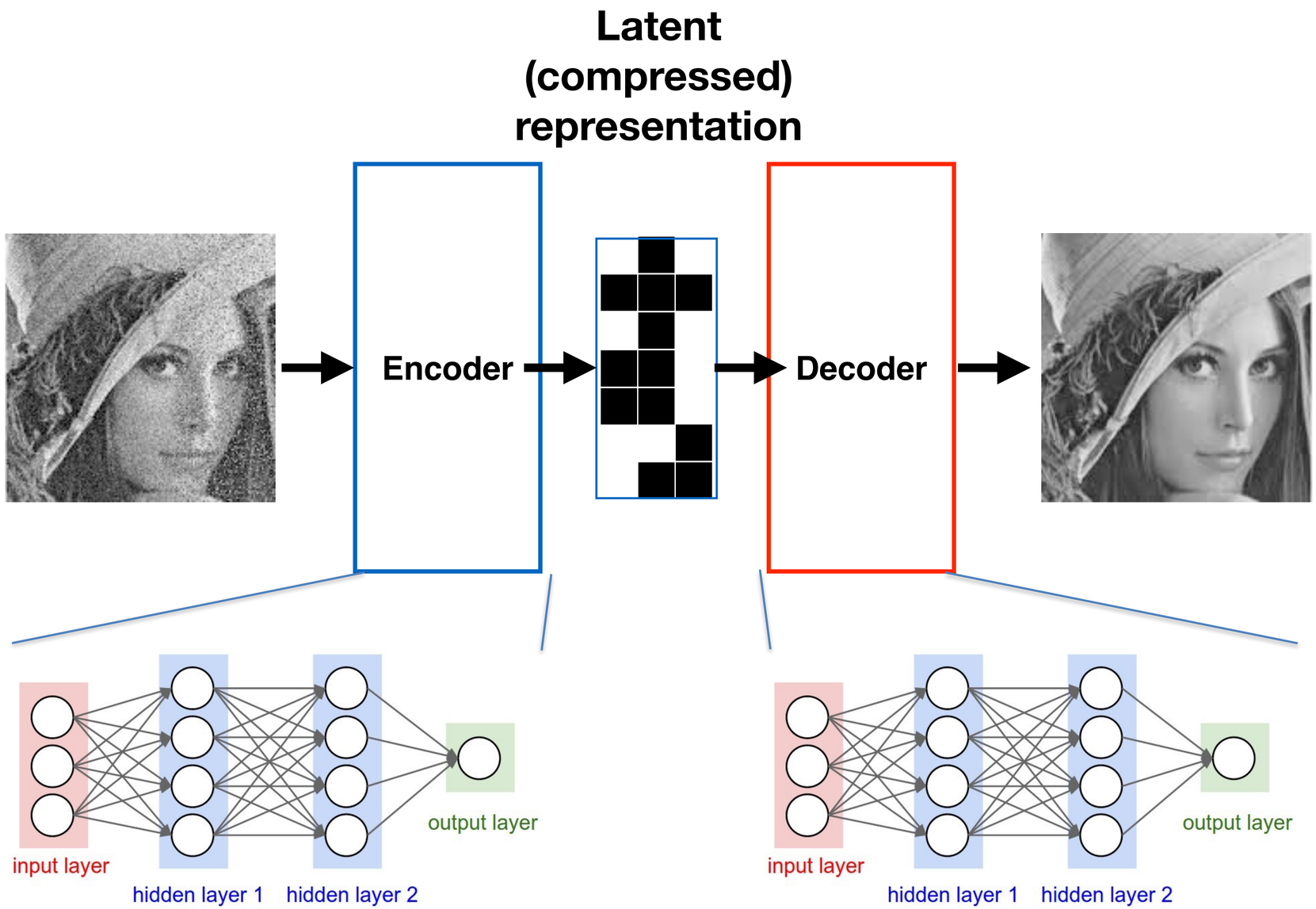


- Autoencoders consist of a pair of networks: an “encoder” and a “decoder”
- The encoder compresses the input down to a low-D representation. The decoder attempts to reconstruct the same input based on the low-D representation



- The compressed representation is a bottleneck - the encoder is forced to throw out irrelevant information and extract the “true” structure underlying the data

- Completely unsupervised - there are no data labels

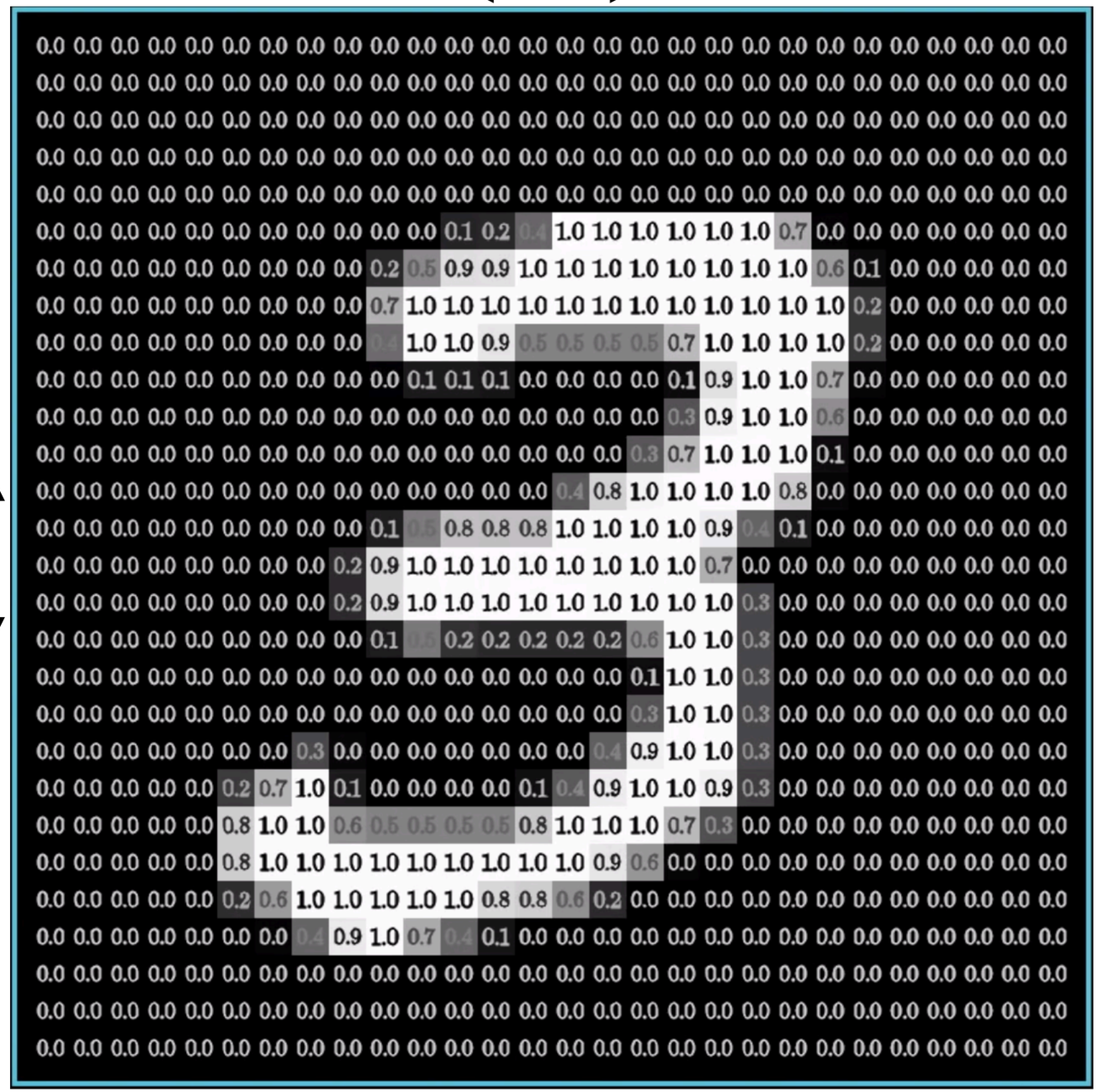


Applications

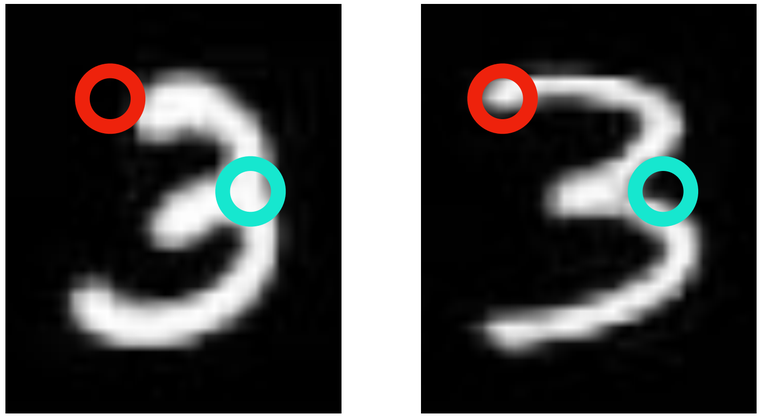
- An autoencoder can reject noise, that is, it filters out signals that don't correspond to the structure underlying the data
- The decoder must also learn to generate the data, i.e., it is a a generative model

28 pixels

28 pixels



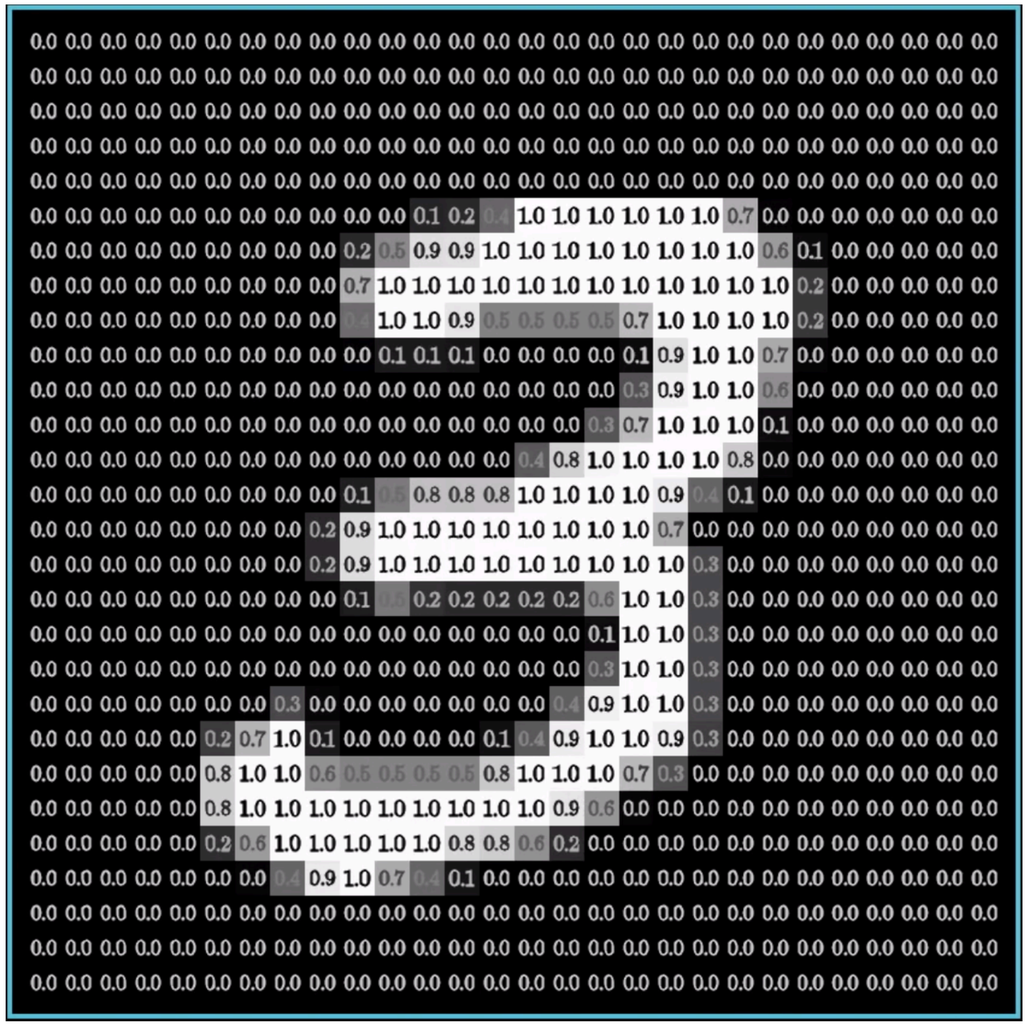
MNIST - 784 total pixels each ranges from 0 to 1

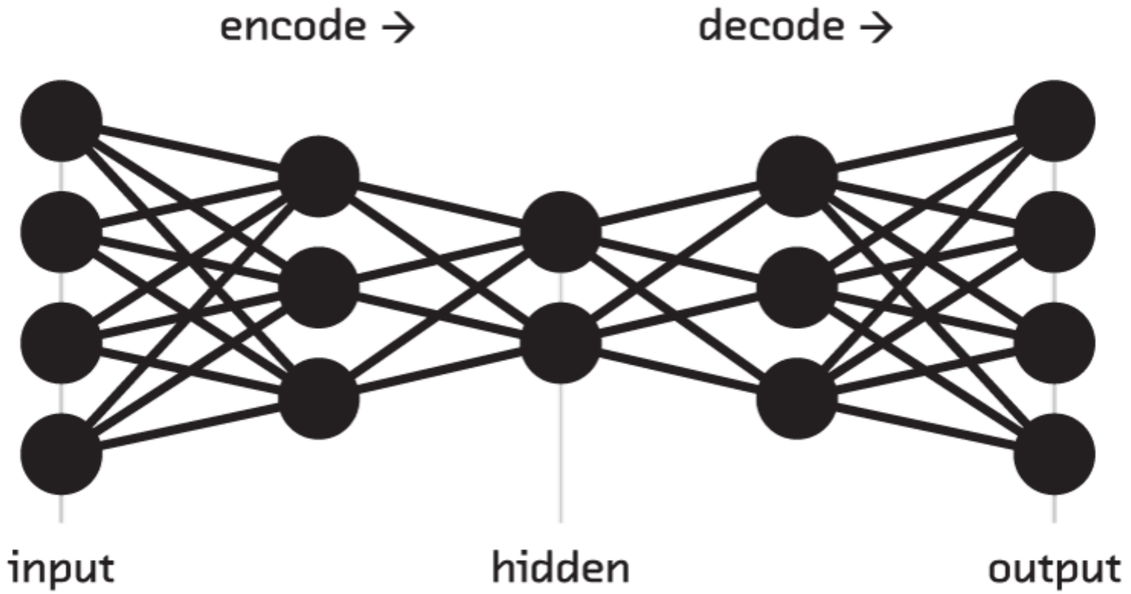


Specific pixel values are very different between images

3Blue1Brown (YouTube)

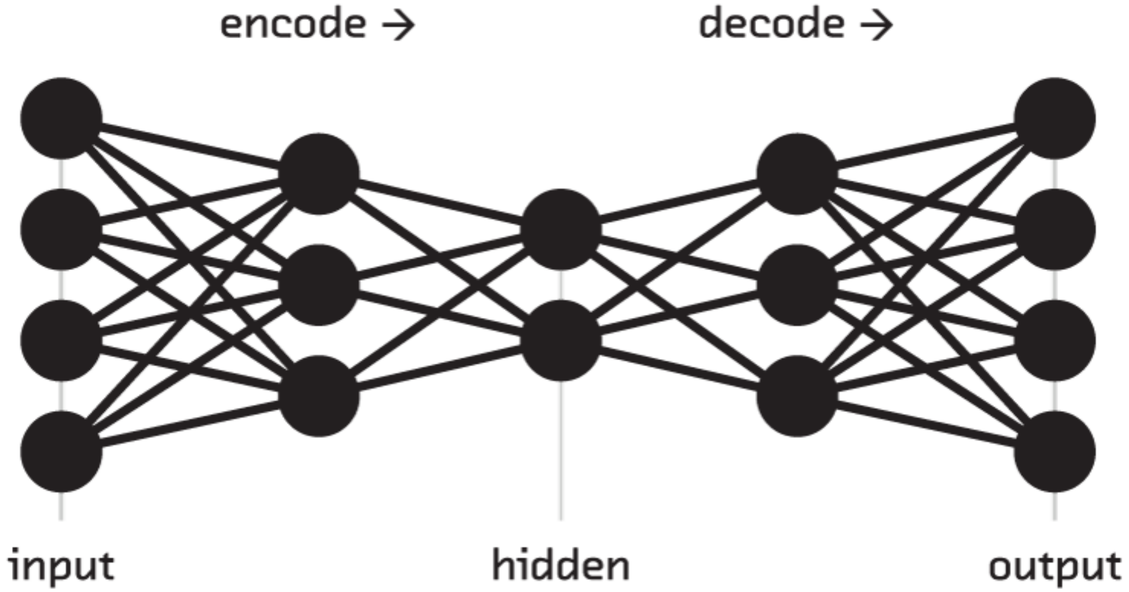
But what is a Neural Network? | Deep learning, chapter 1

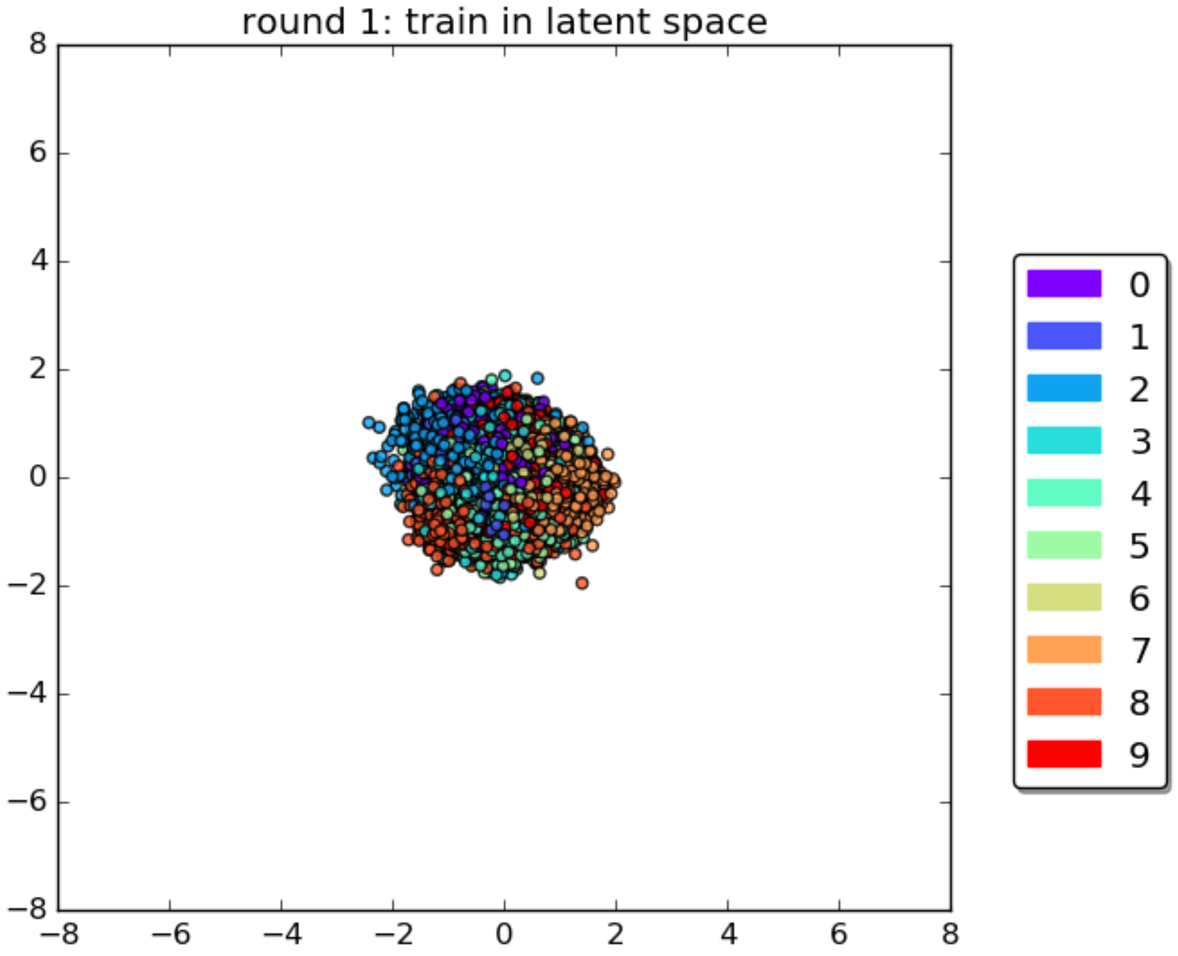




- Network weights are typically initialized completely randomly
- Through the course of training, the latent representation begins to exhibit structure that matches the structure in the data

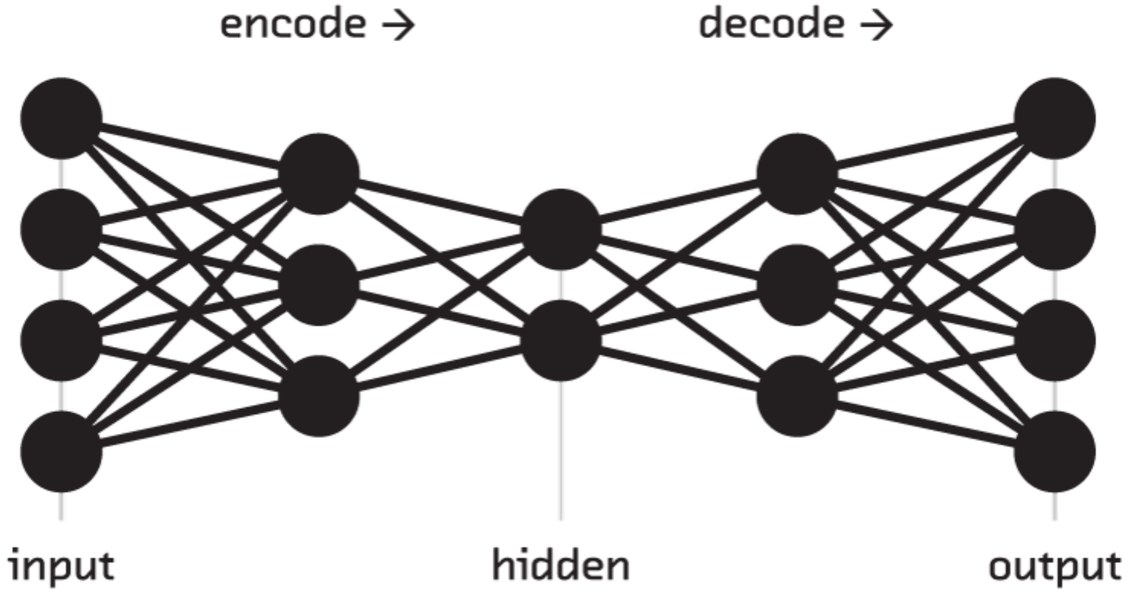
https://blog.fastforwardlabs.com/2016/08/12/introducing-variational-autoencoders-in-prose-and.html

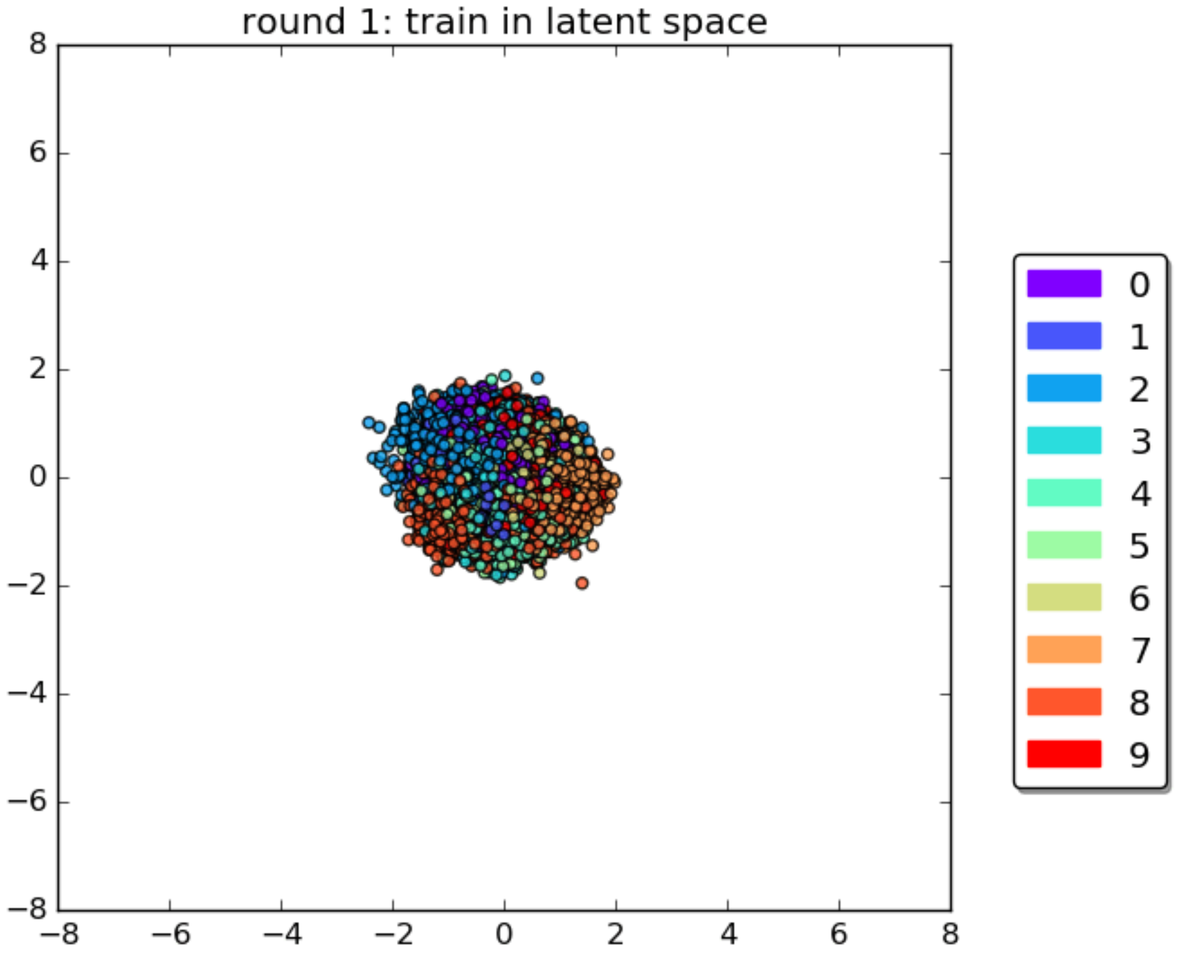




- Through the course of training, the latent representation begins to exhibit structure that matches the structure in the data

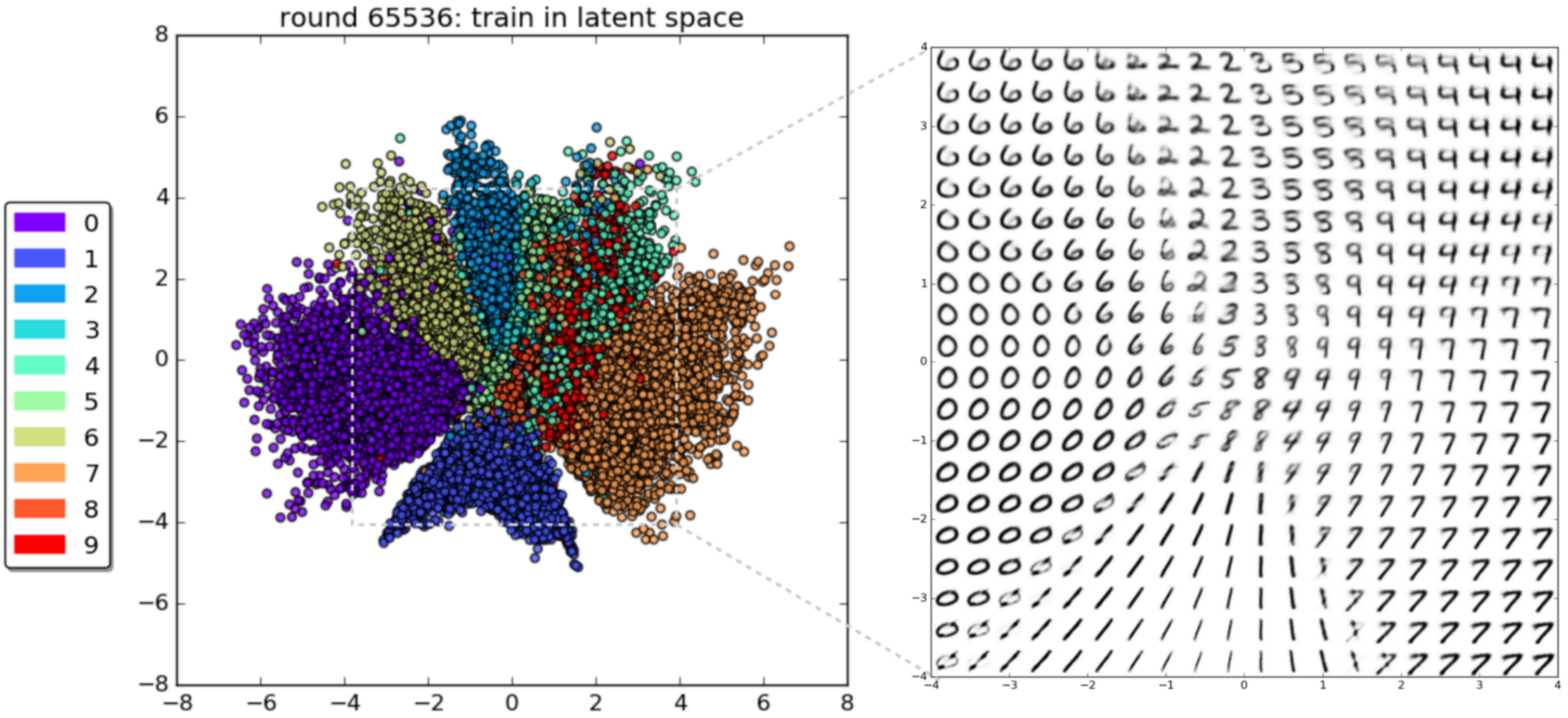
https://blog.fastforwardlabs.com/2016/08/12/introducing-variational-autoencoders-in-prose-and.html





- Through the course of training, the latent representation begins to exhibit structure that matches the structure in the data

https://blog.fastforwardlabs.com/2016/08/12/introducing-variational-autoencoders-in-prose-and.html



https://blog.fastforwardlabs.com/2016/08/12/introducing-variational-autoencoders-in-prose-and.html

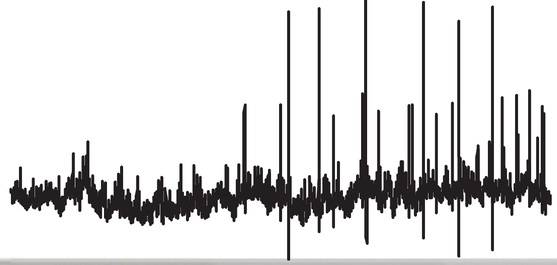
What you'll hear about in this lecture

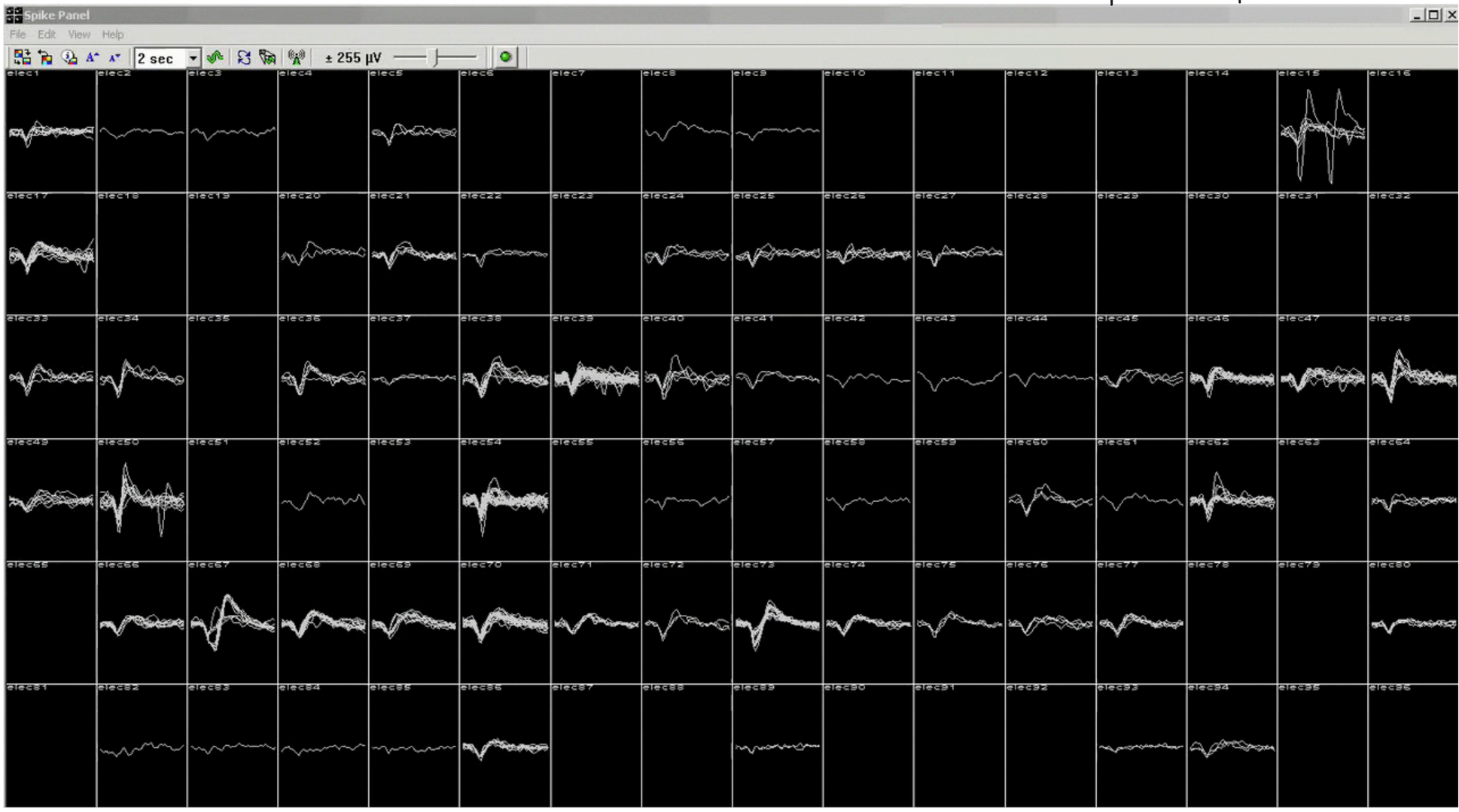
Neural network basics

Deep autoencoders

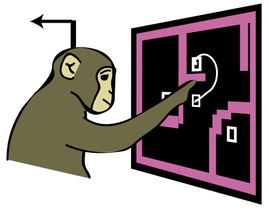
Intro to neural population dynamics

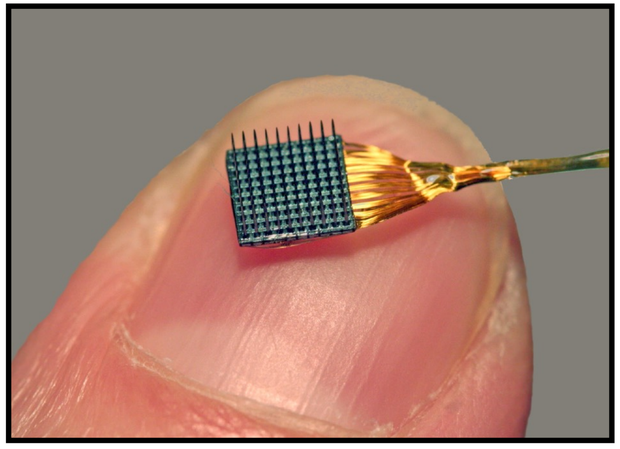
Intracortical electrophysiological recordings



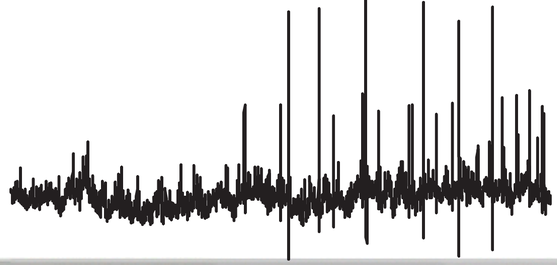


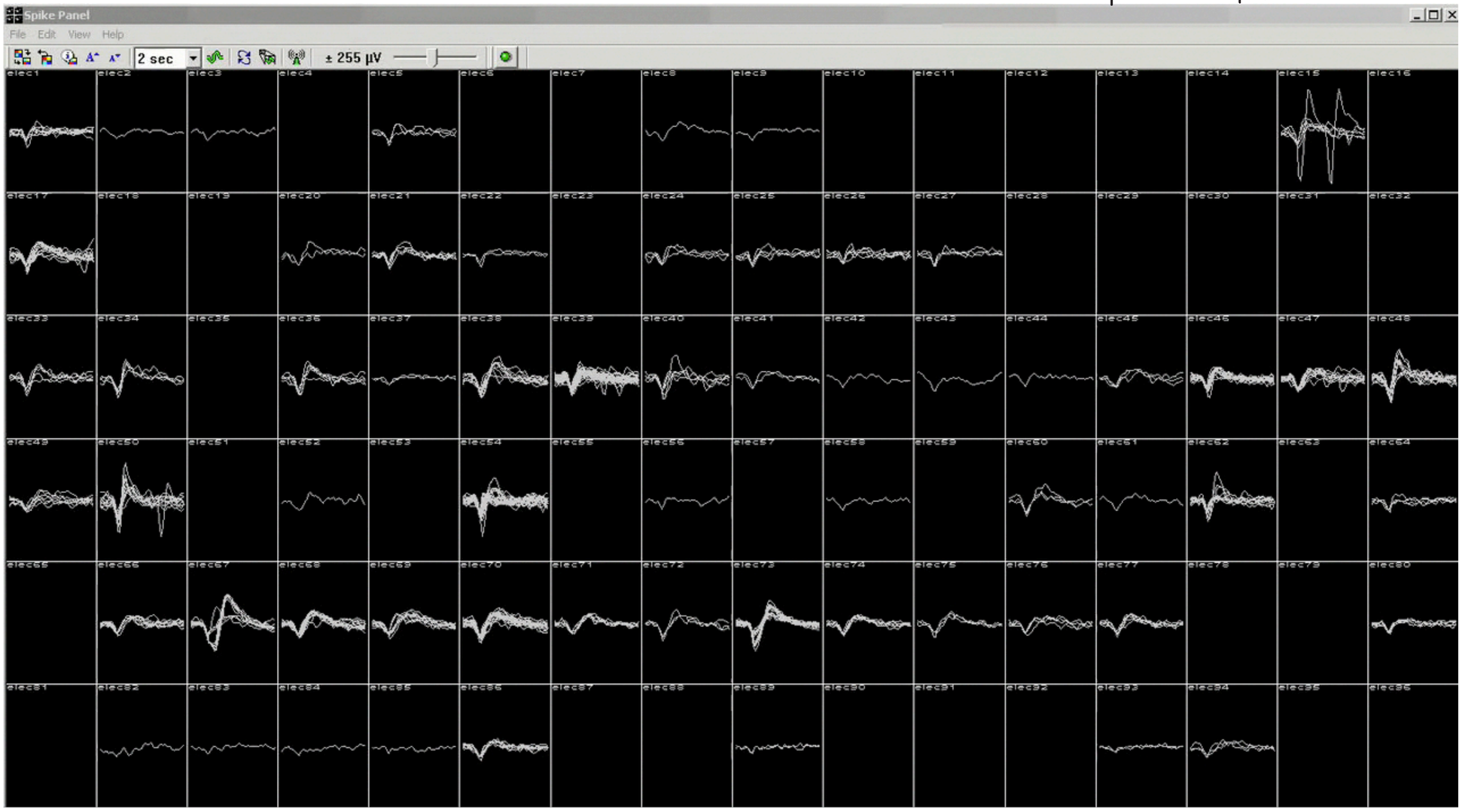
M1/
PMd



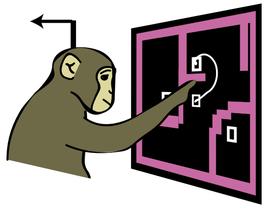


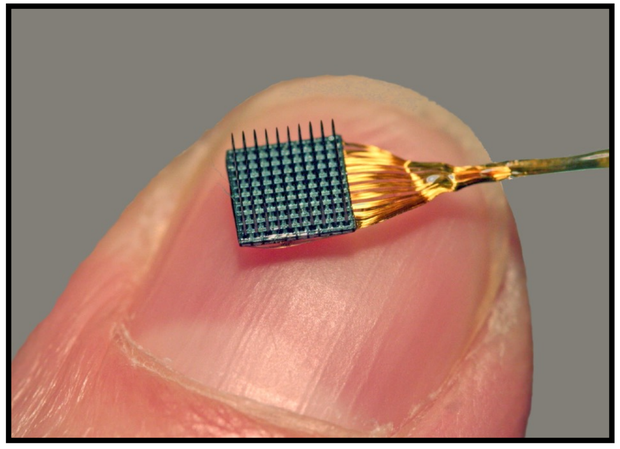
Intracortical electrophysiological recordings



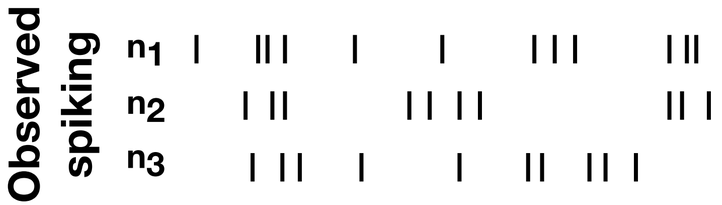


M1/
PMd

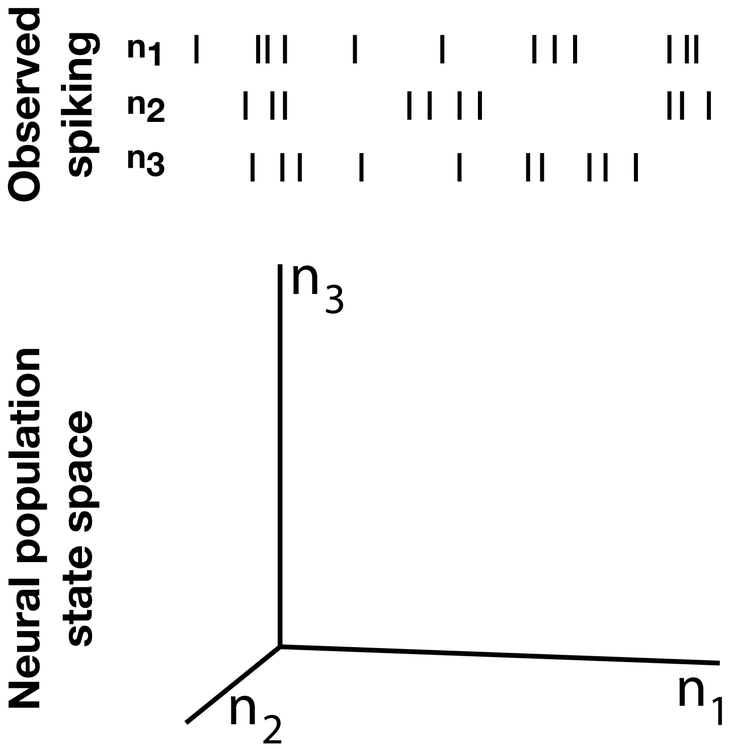




Neural population state spaces

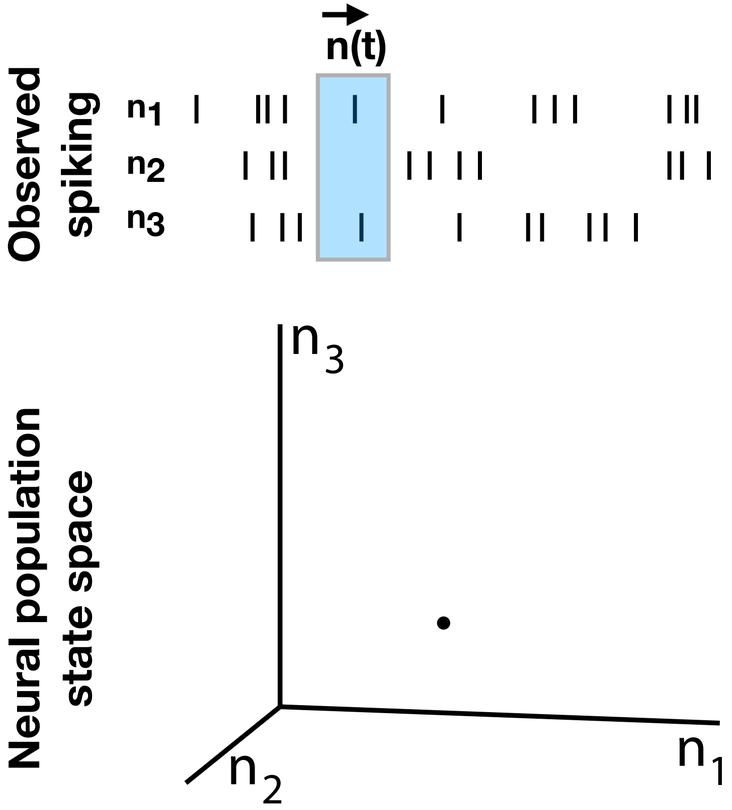


Neural population state spaces



Adapted from Cunningham & Yu, *Nat Neuro* 2014
Dimensionality reduction for large-scale neural recordings

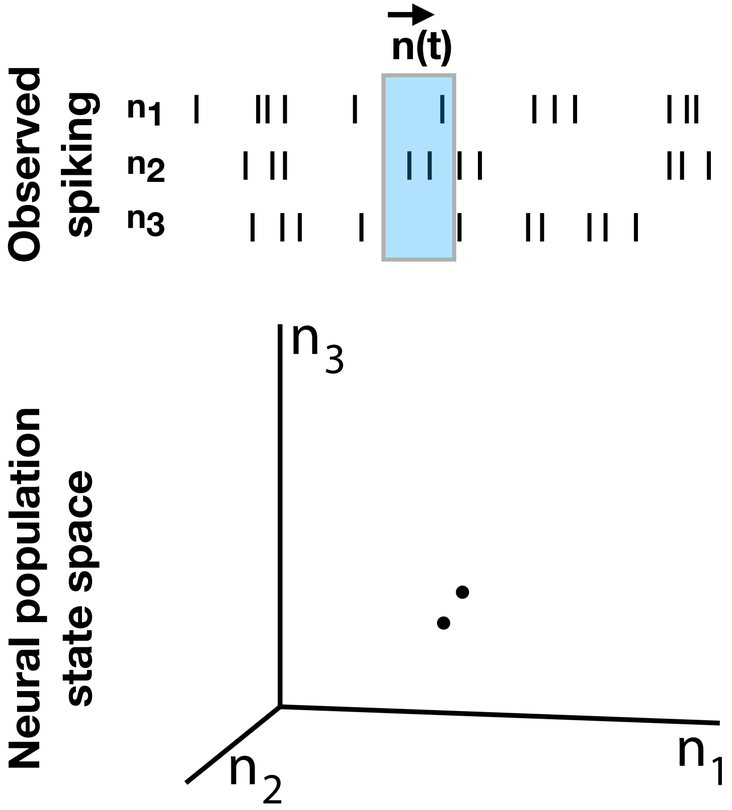
Neural population state spaces



- The current state of the system is captured by the firing rates of all of its neurons

Adapted from Cunningham & Yu, *Nat Neuro* 2014
Dimensionality reduction for large-scale neural recordings

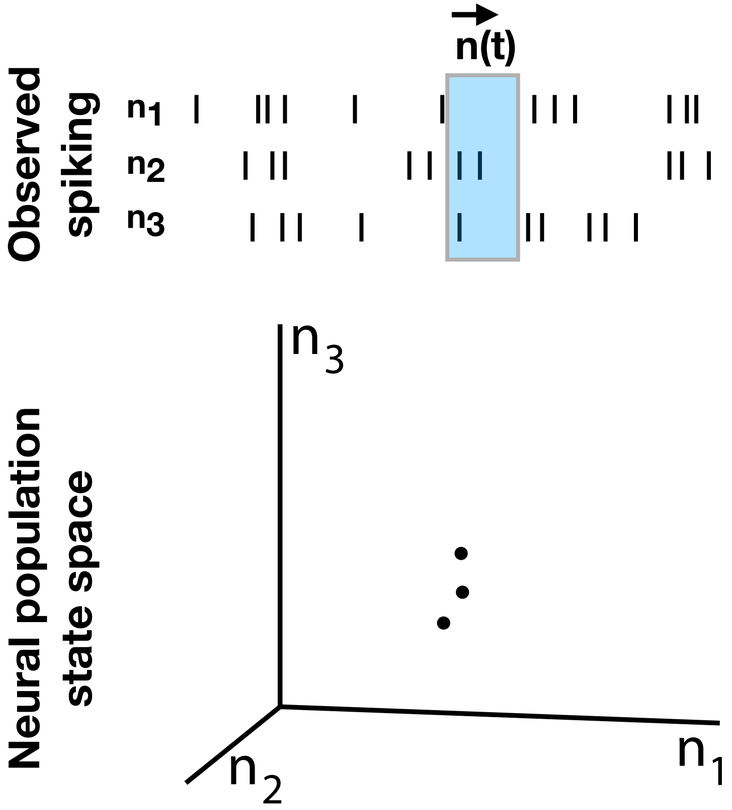
Neural population state spaces



- The current state of the system is captured by the firing rates of all of its neurons

Adapted from Cunningham & Yu, *Nat Neuro* 2014
Dimensionality reduction for large-scale neural recordings

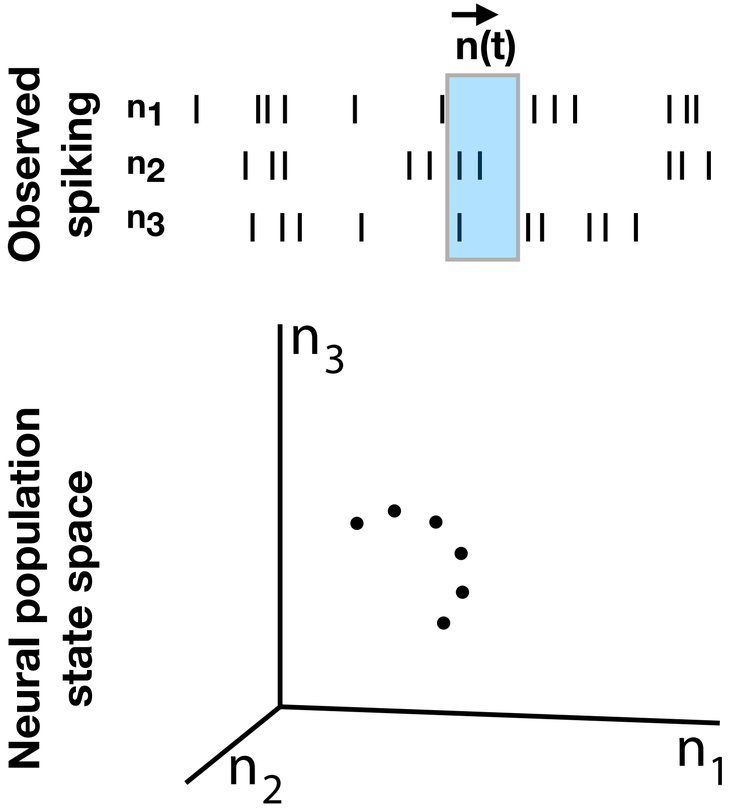
Neural population state spaces



- The current state of the system is captured by the firing rates of all of its neurons

Adapted from Cunningham & Yu, *Nat Neuro* 2014
Dimensionality reduction for large-scale neural recordings

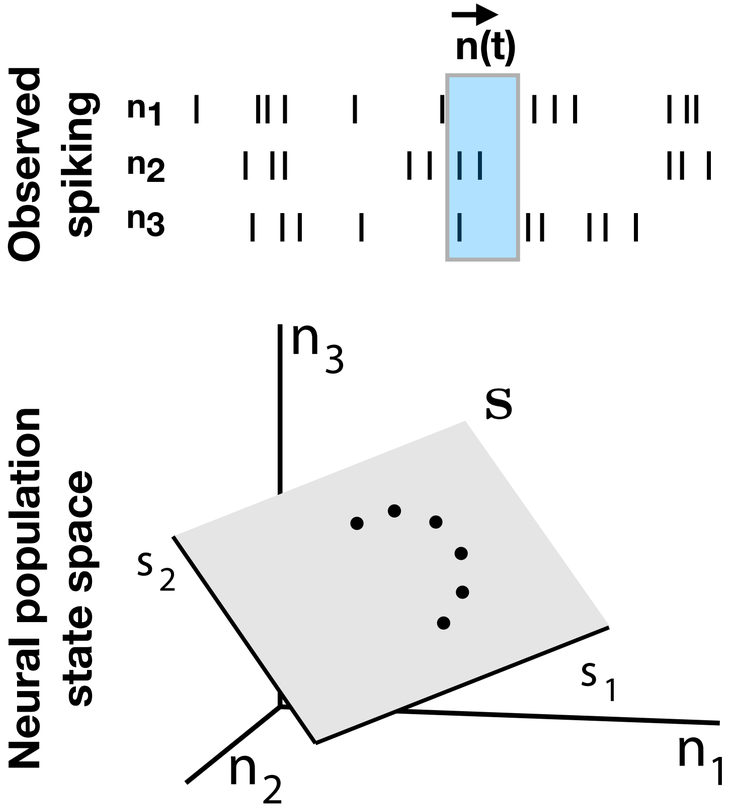
Neural population state spaces



Adapted from Cunningham & Yu, *Nat Neuro* 2014
Dimensionality reduction for large-scale neural recordings

- The current state of the system is captured by the firing rates of all of its neurons
- Activity does not uniformly fill the space - i.e., not all patterns of neural firing rates are equally likely

Neural population state spaces

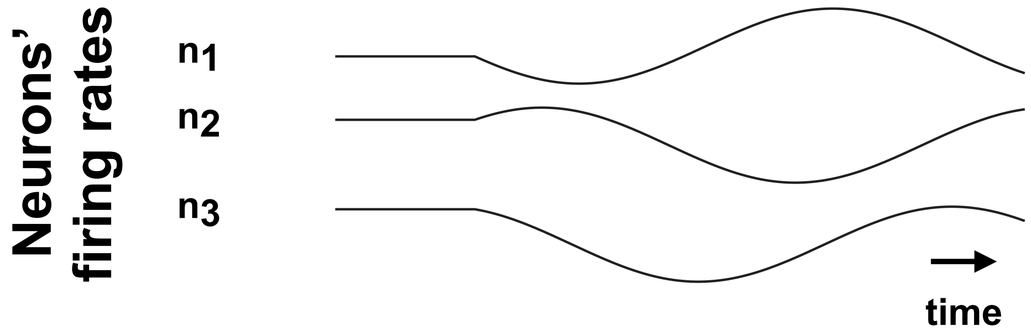


Adapted from Cunningham & Yu, *Nat Neuro* 2014
Dimensionality reduction for large-scale neural recordings

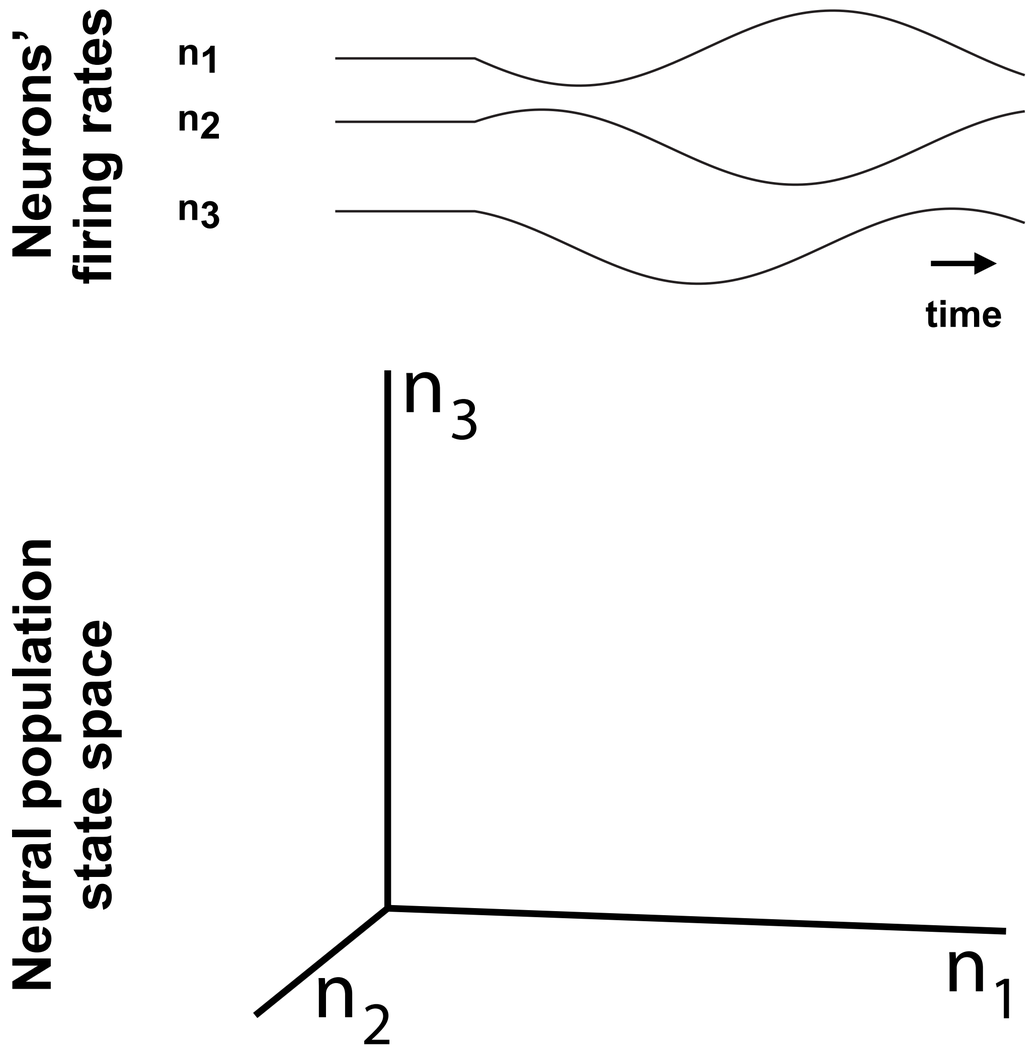
- The current state of the system is captured by the firing rates of all of its neurons
- Activity does not uniformly fill the space - i.e., not all patterns of neural firing rates are equally likely
- We can usually identify shared structure underlying the firing rates of the recorded neurons

Changes in neurons’ firing rates are coordinated

Changes in neurons' firing rates are coordinated

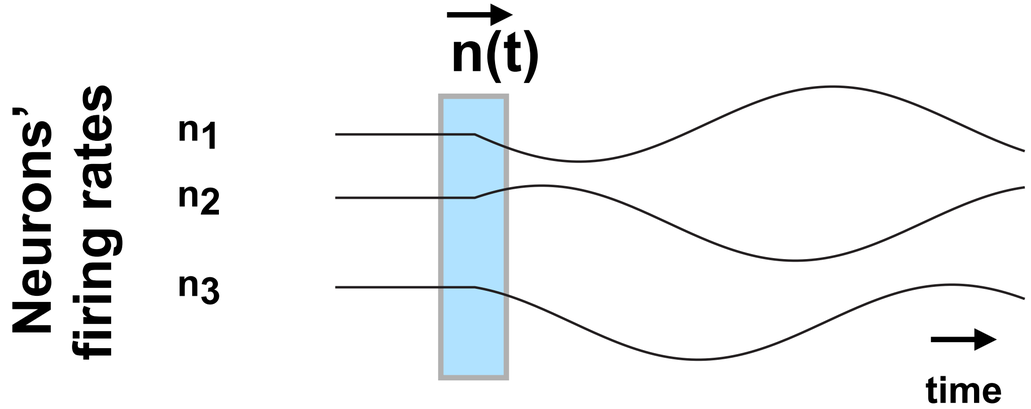


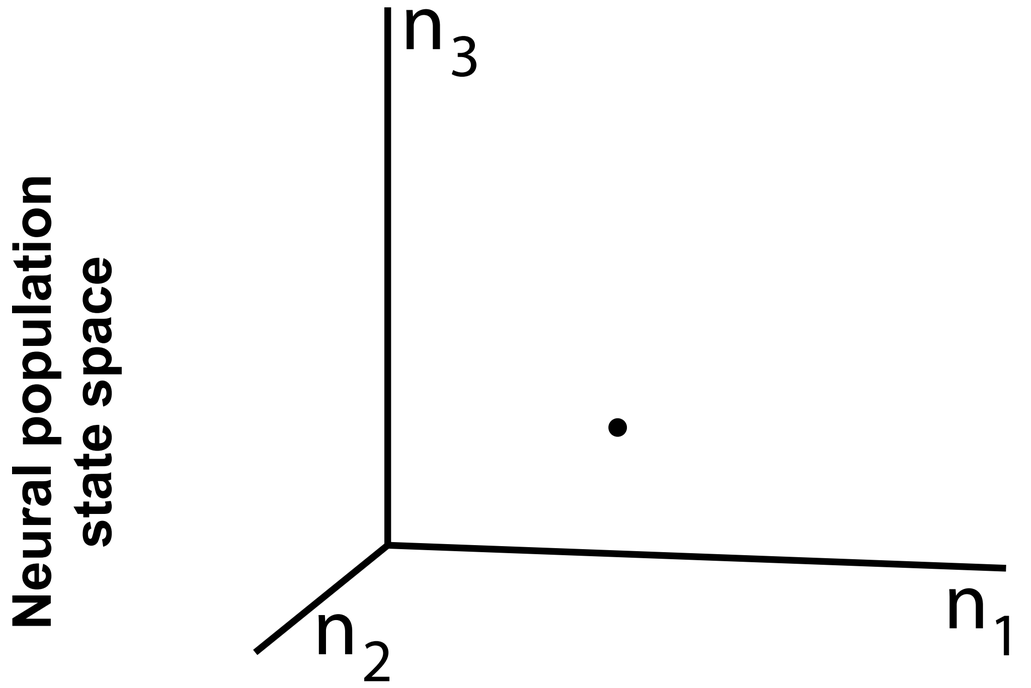
Changes in neurons' firing rates are coordinated



Adapted from Cunningham & Yu, *Nat Neuro* 2014

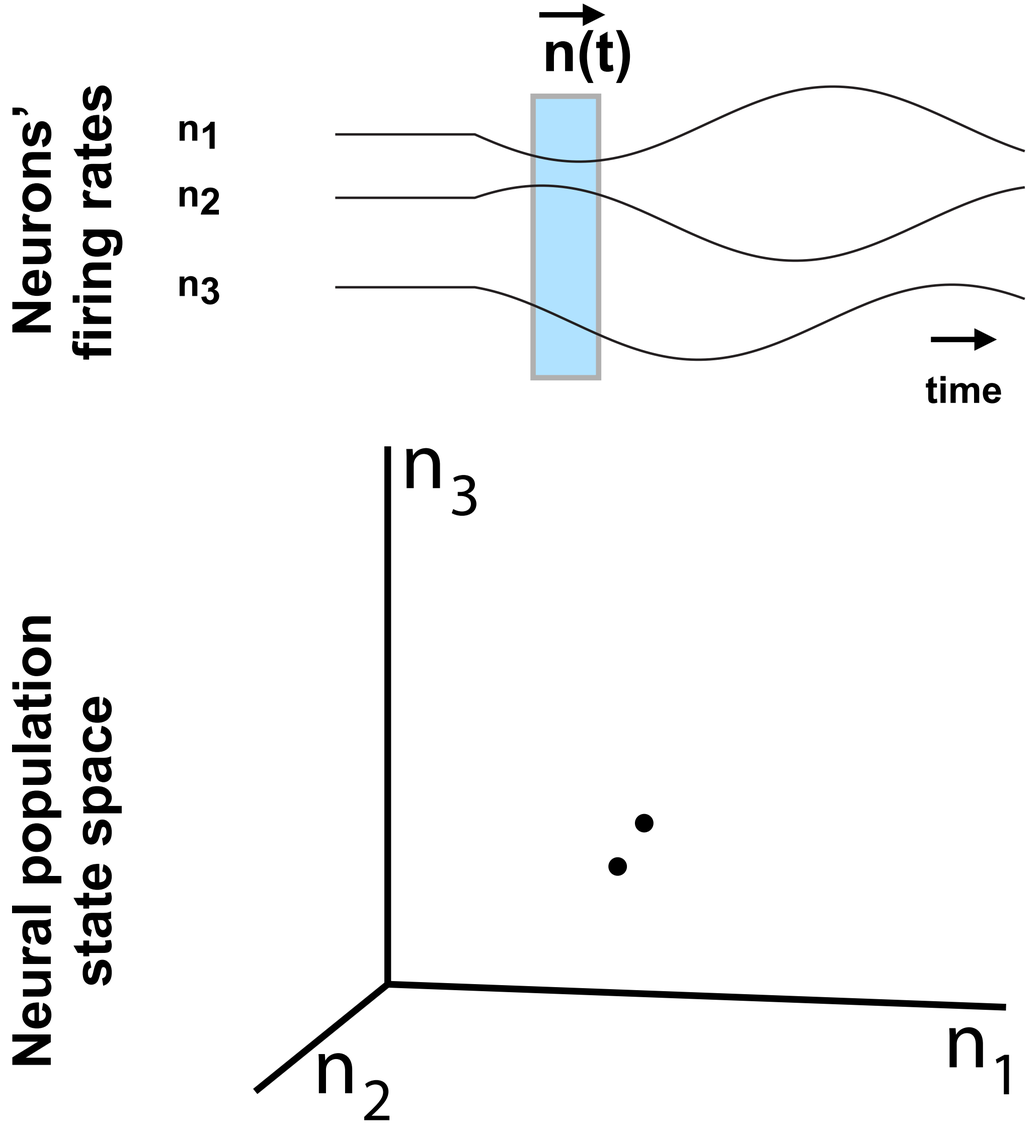
Changes in neurons' firing rates are coordinated





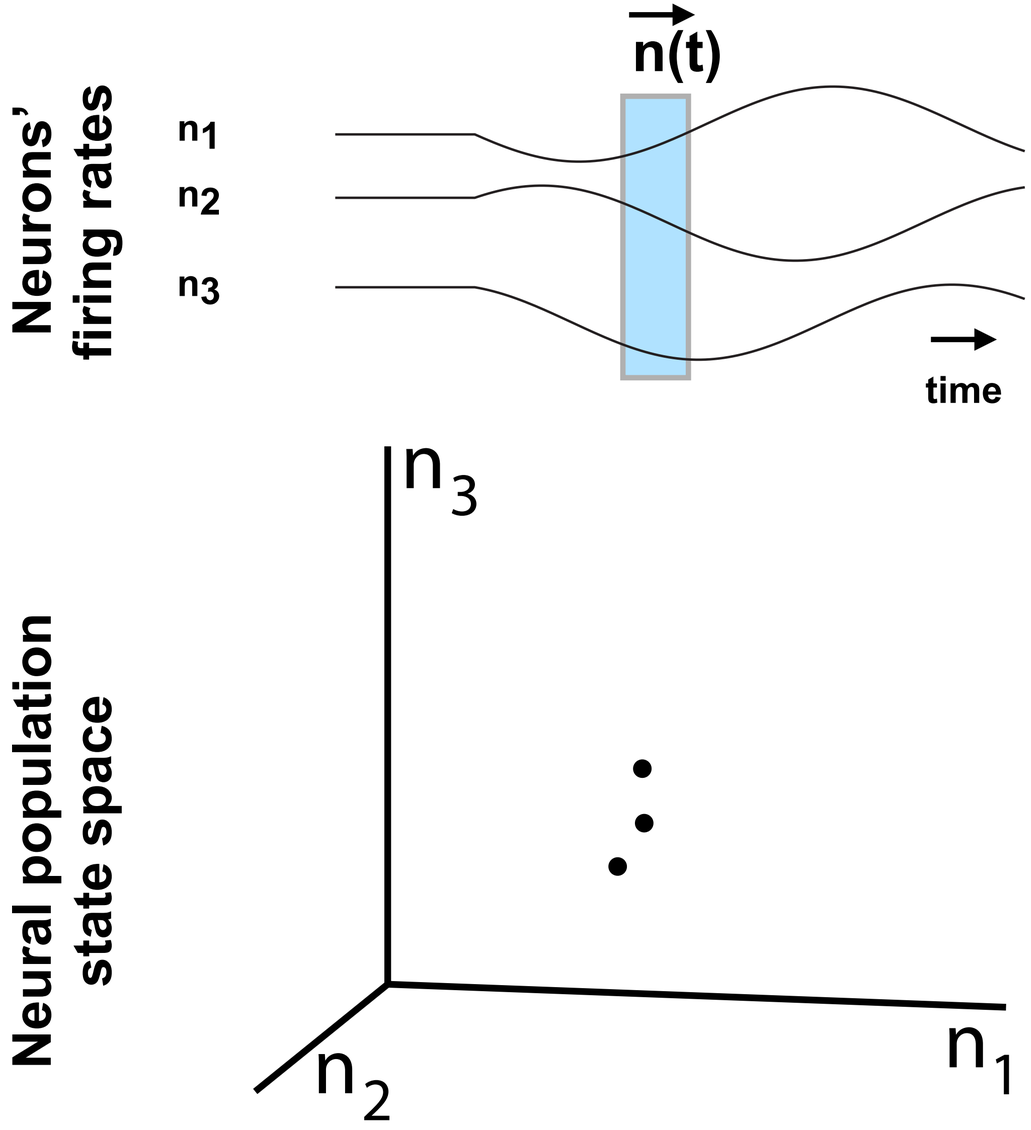
Adapted from Cunningham & Yu, *Nat Neuro* 2014

Changes in neurons' firing rates are coordinated



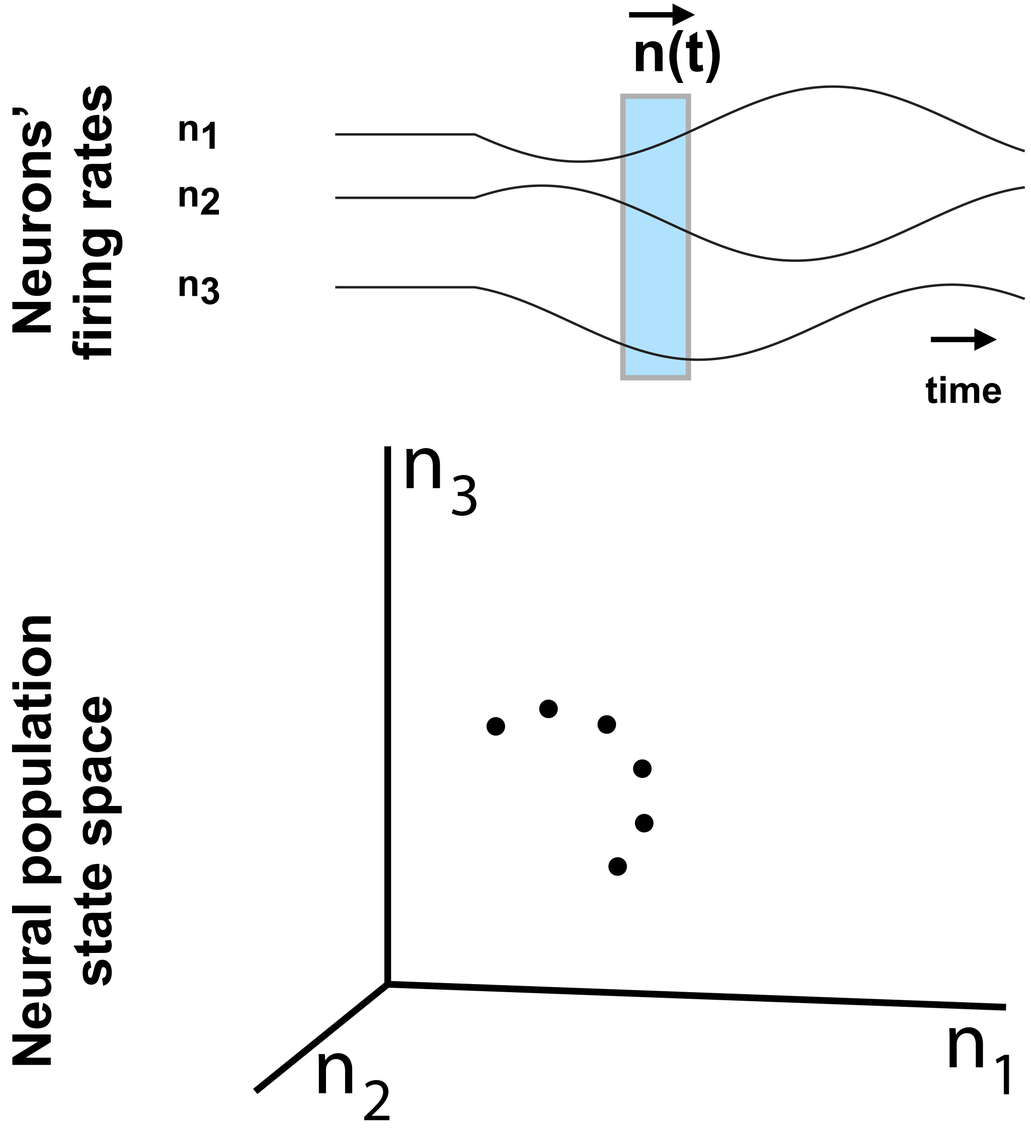
Adapted from Cunningham & Yu, *Nat Neuro* 2014

Changes in neurons' firing rates are coordinated



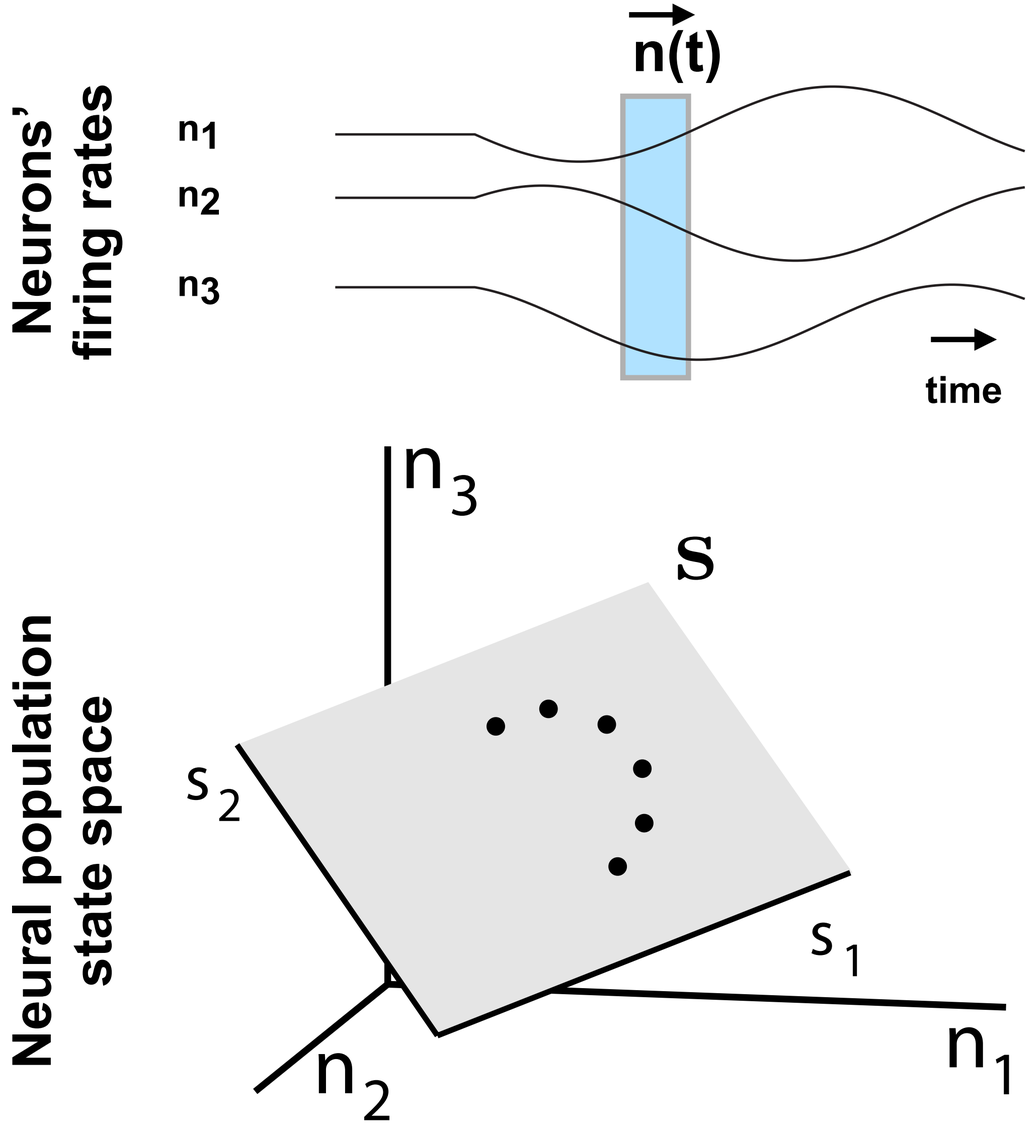
Adapted from Cunningham & Yu, *Nat Neuro* 2014

Changes in neurons' firing rates are coordinated



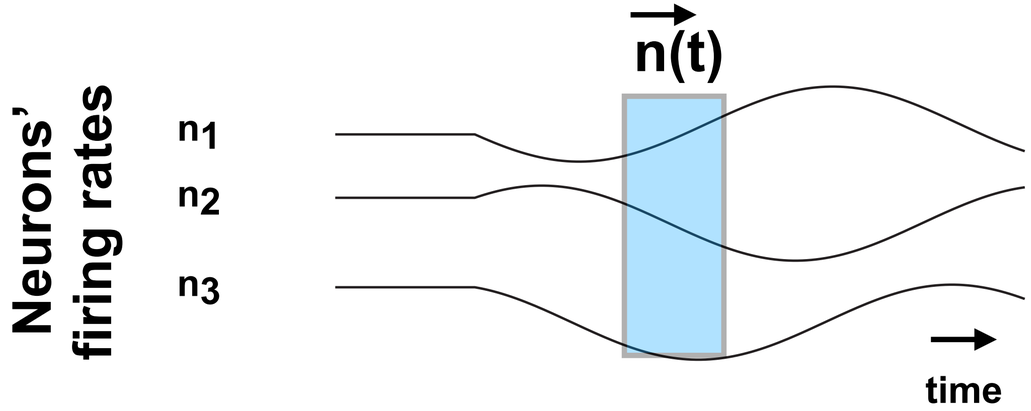
Adapted from Cunningham & Yu, *Nat Neuro* 2014

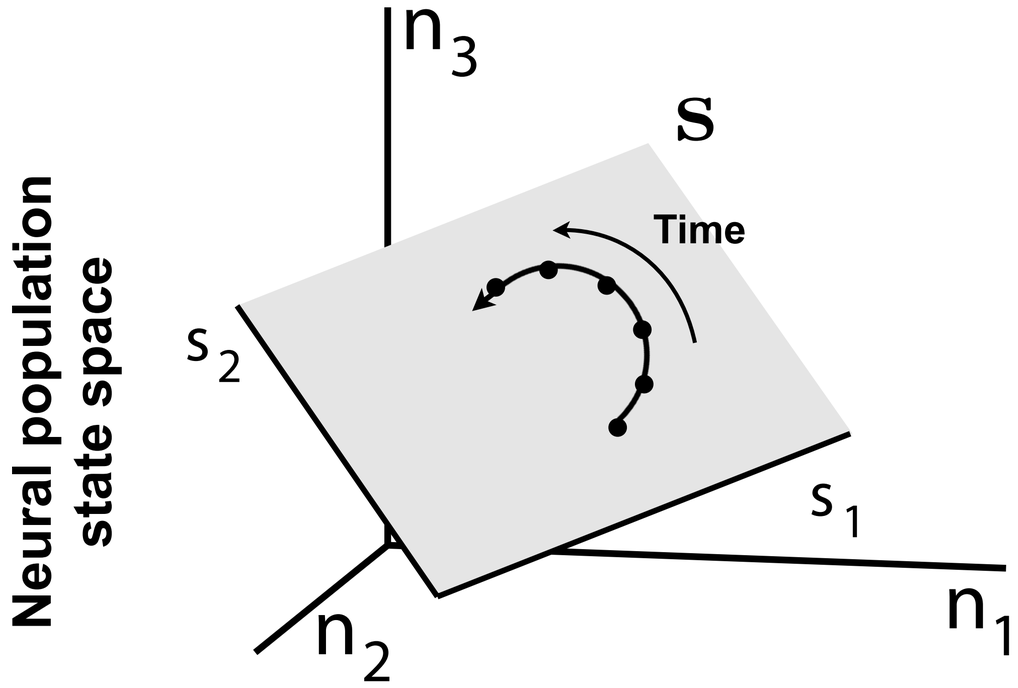
Changes in neurons' firing rates are coordinated



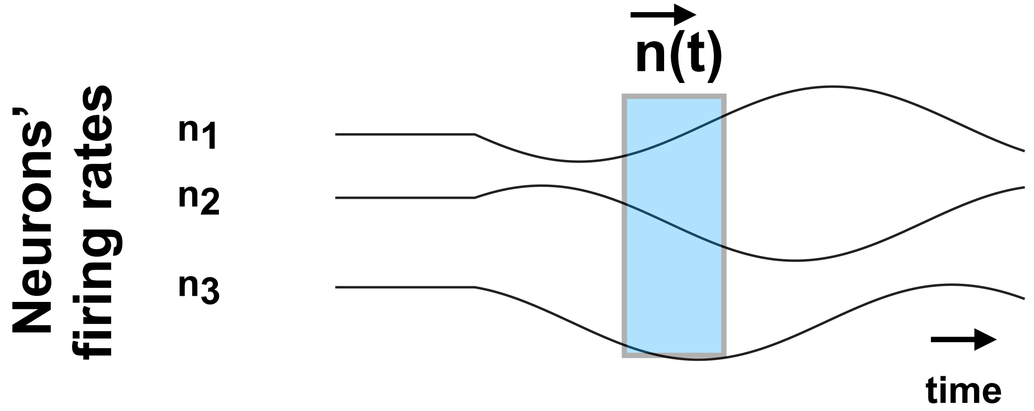
Adapted from Cunningham & Yu, *Nat Neuro* 2014

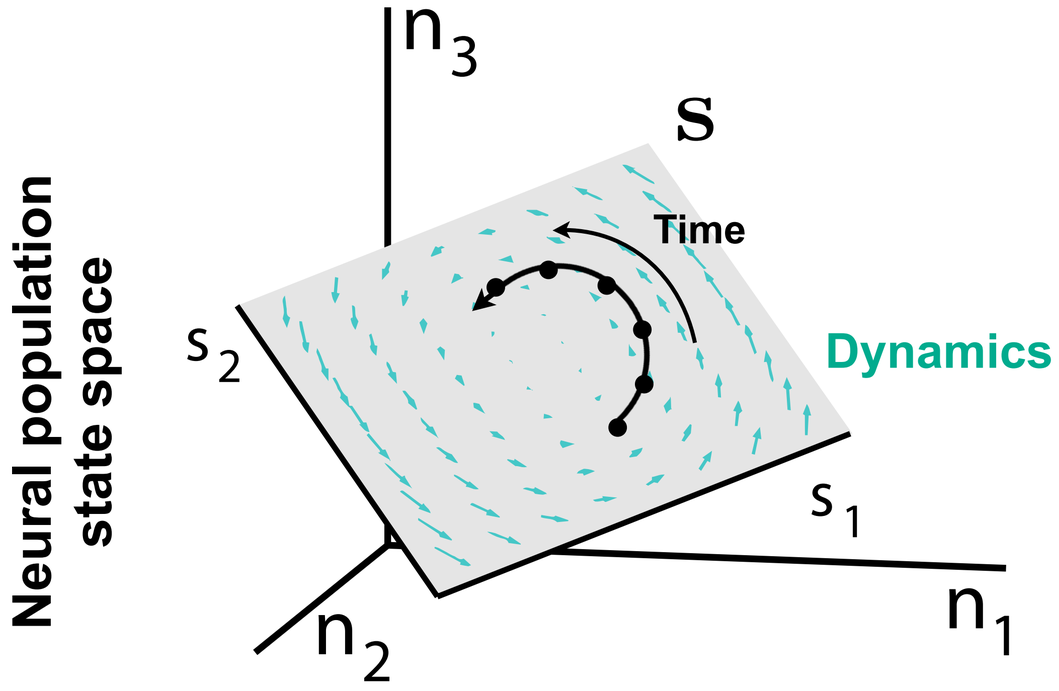
Neural population dynamics



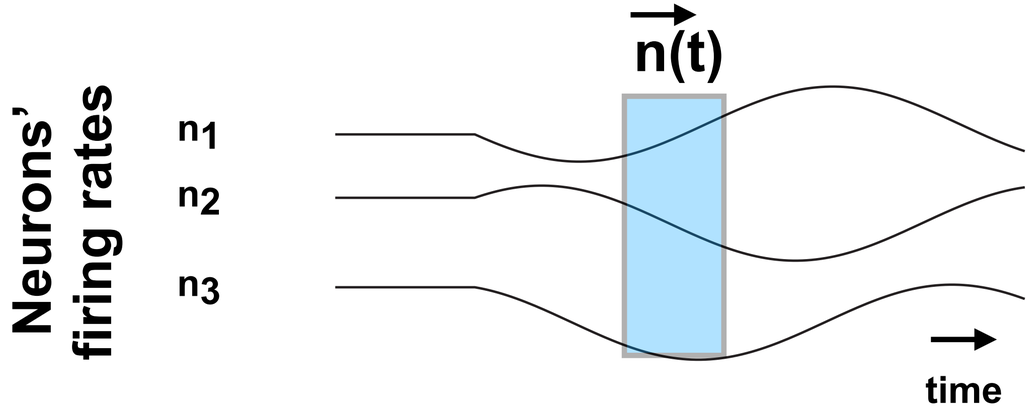


Neural population dynamics



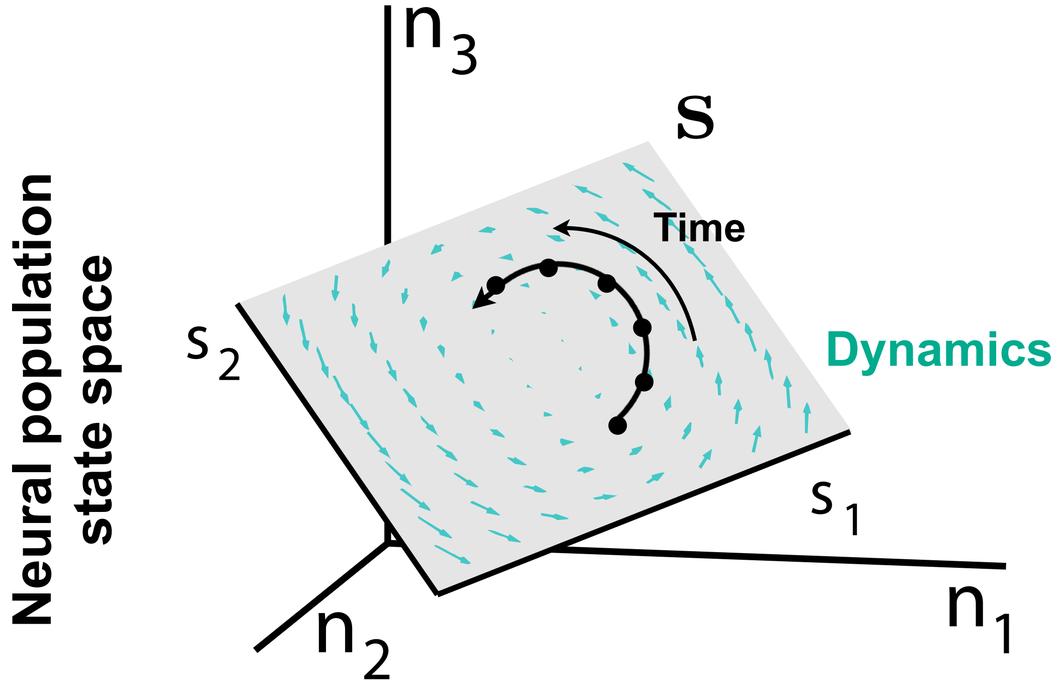


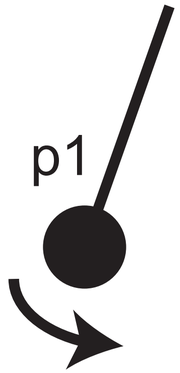
Neural population dynamics

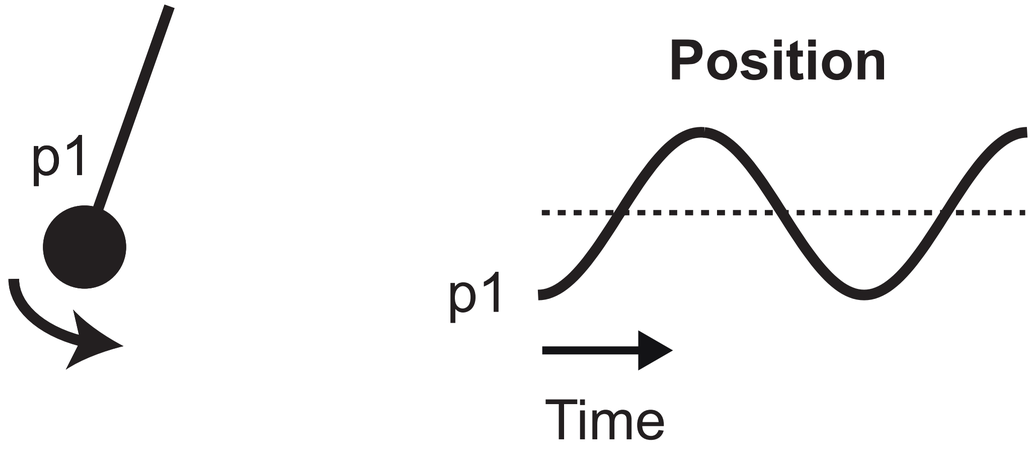


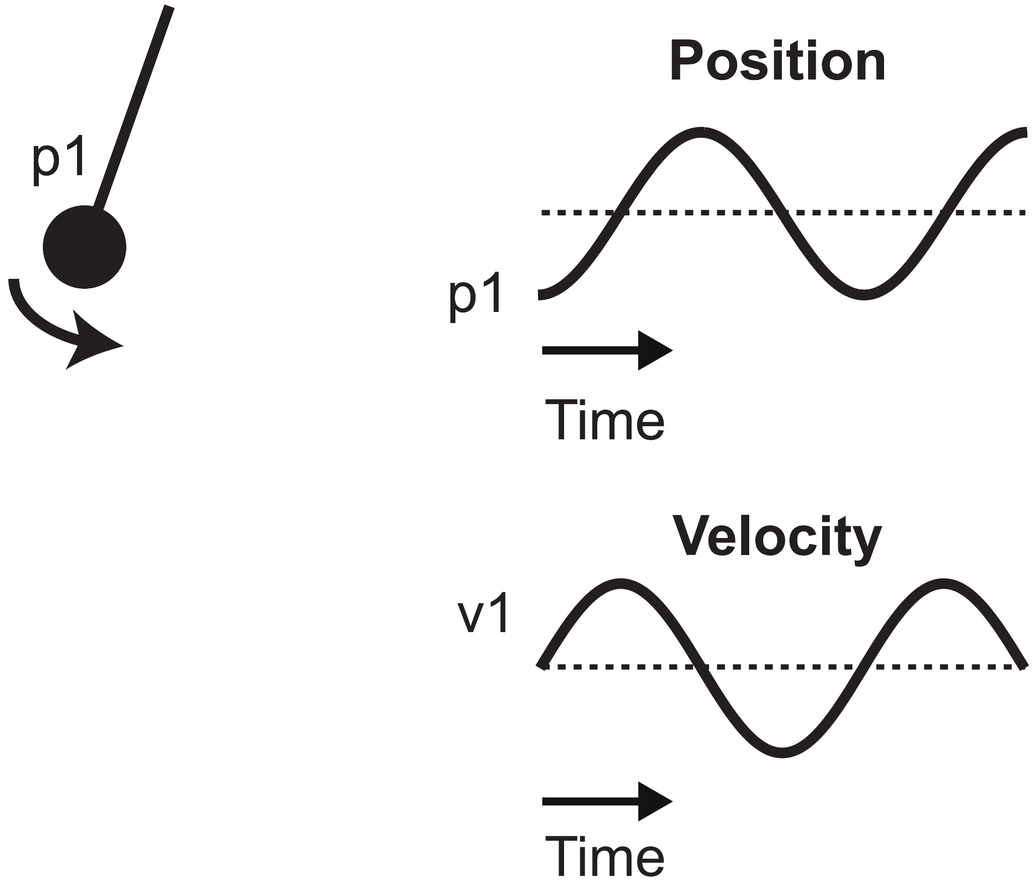
Predictable activity - modeled by *autonomous* dynamics

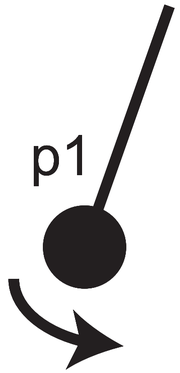
$$\frac{d\mathbf{s}}{dt} = f(\mathbf{s})$$

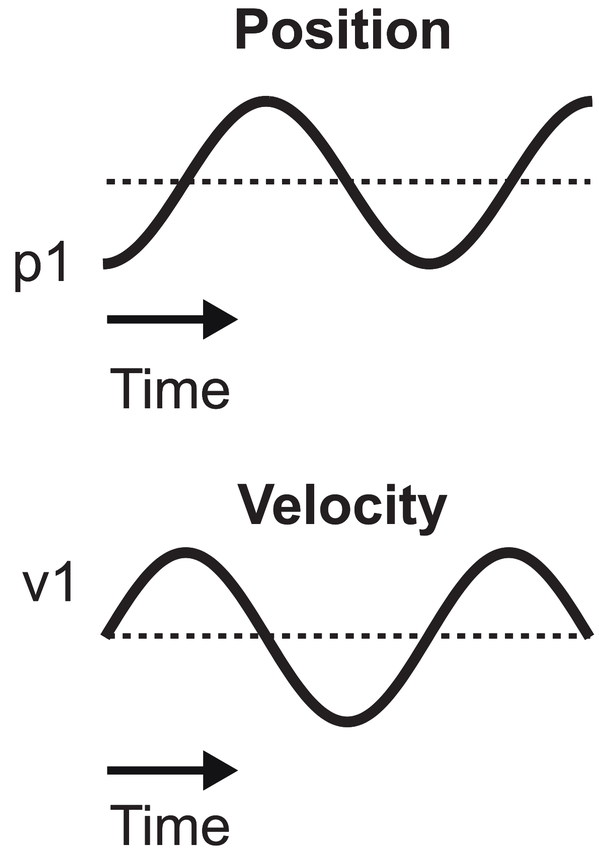



p1

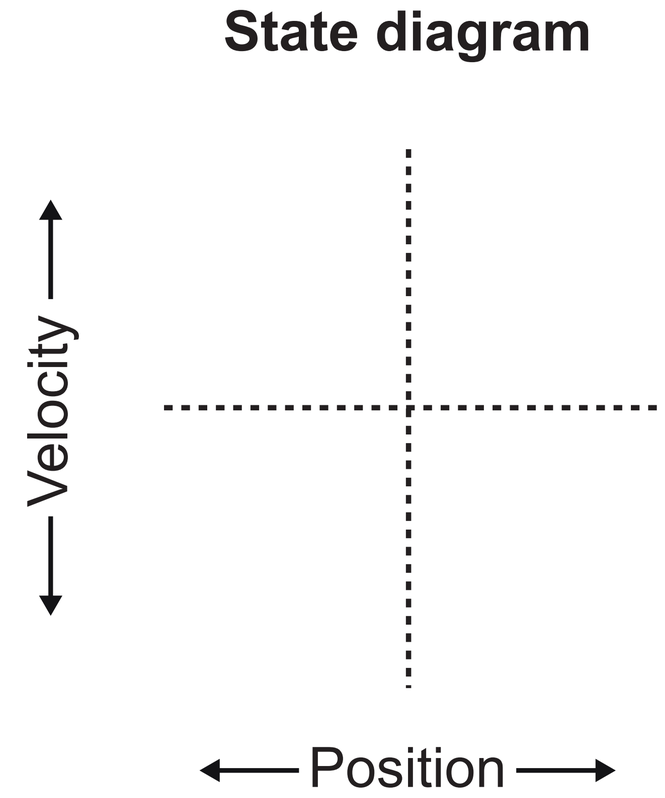

p1
Position
p1
Time

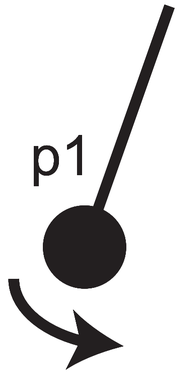

p1
Position
p1
Time
Velocity
v1
Time

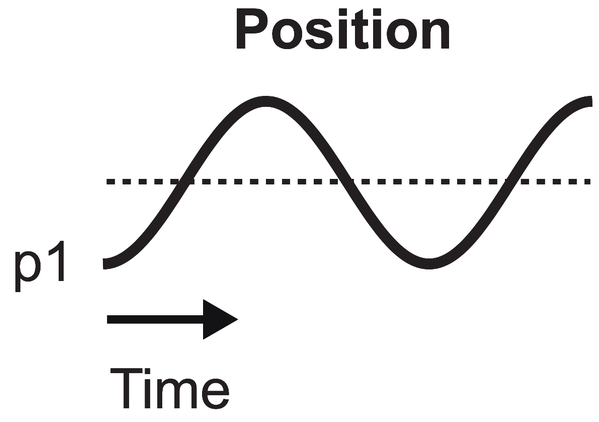


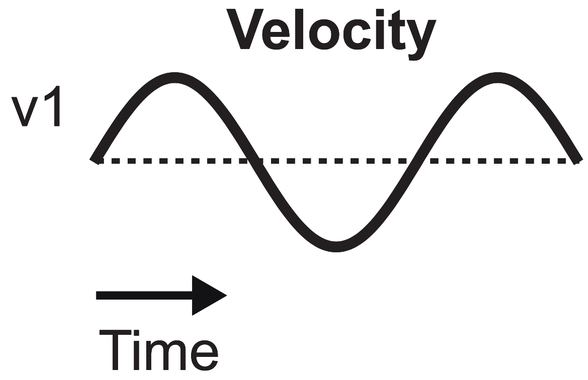


$$\mathbf{x} = \begin{bmatrix} p \\ v \end{bmatrix}$$

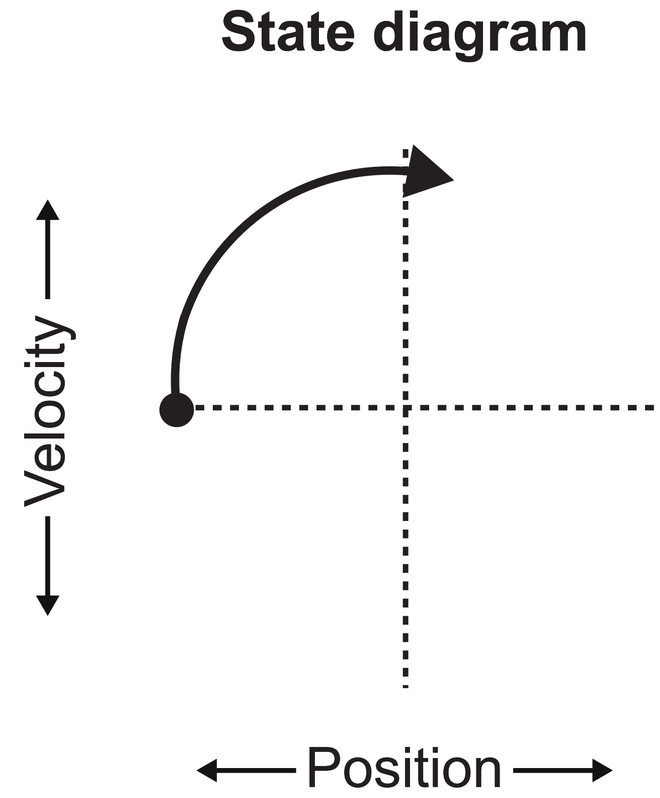


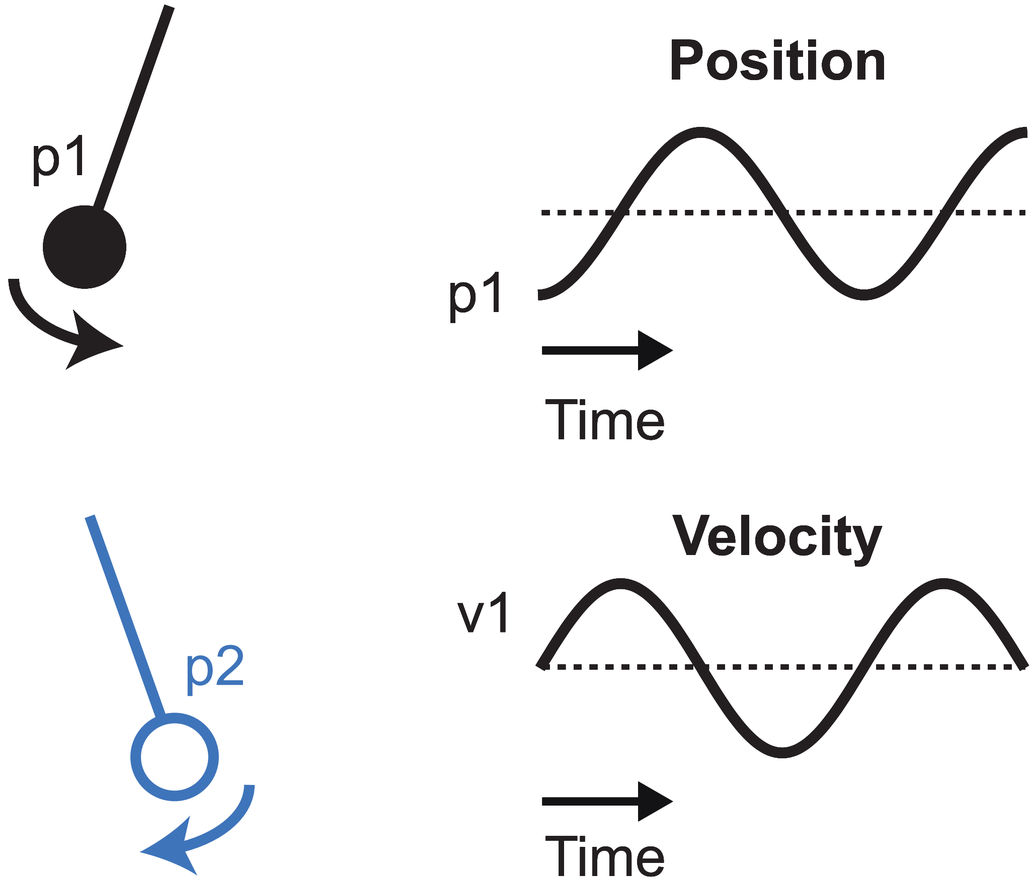




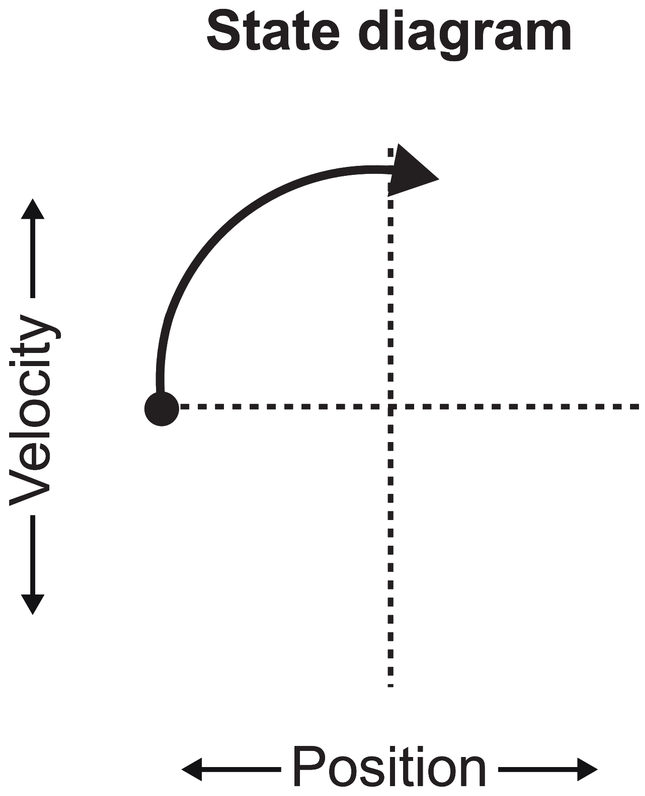


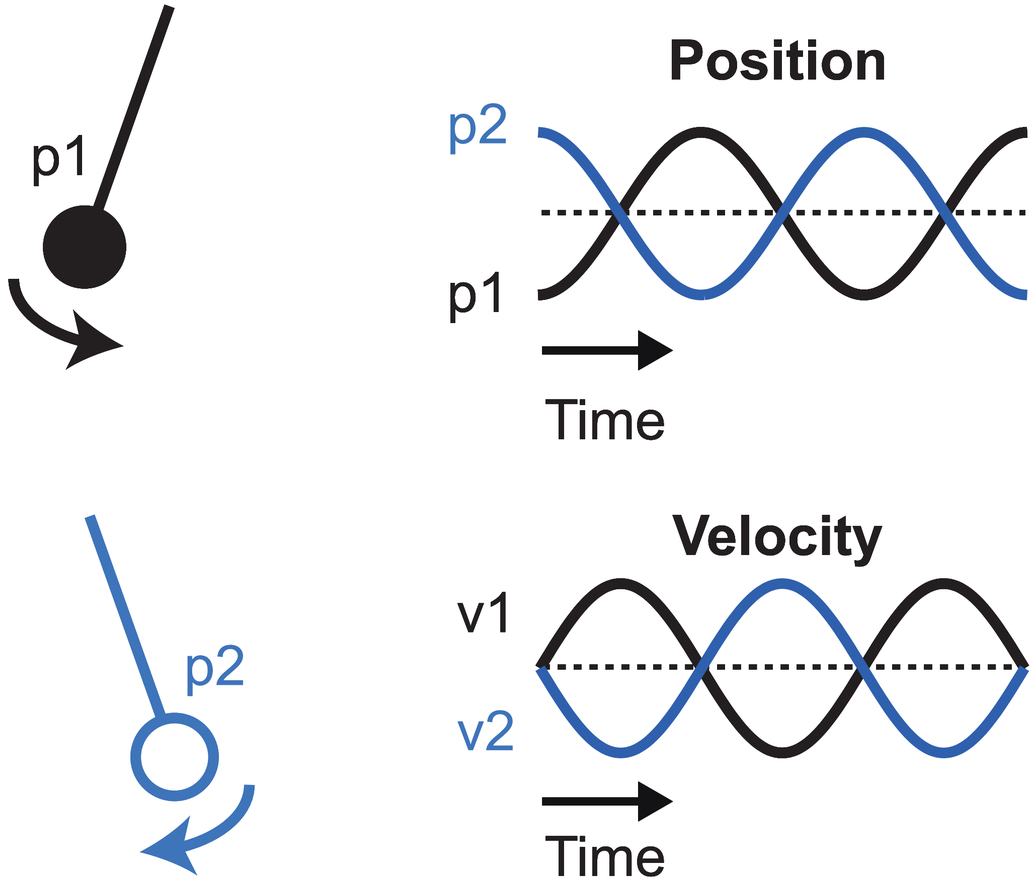
$$\mathbf{x} = \begin{bmatrix} p \\ v \end{bmatrix}$$



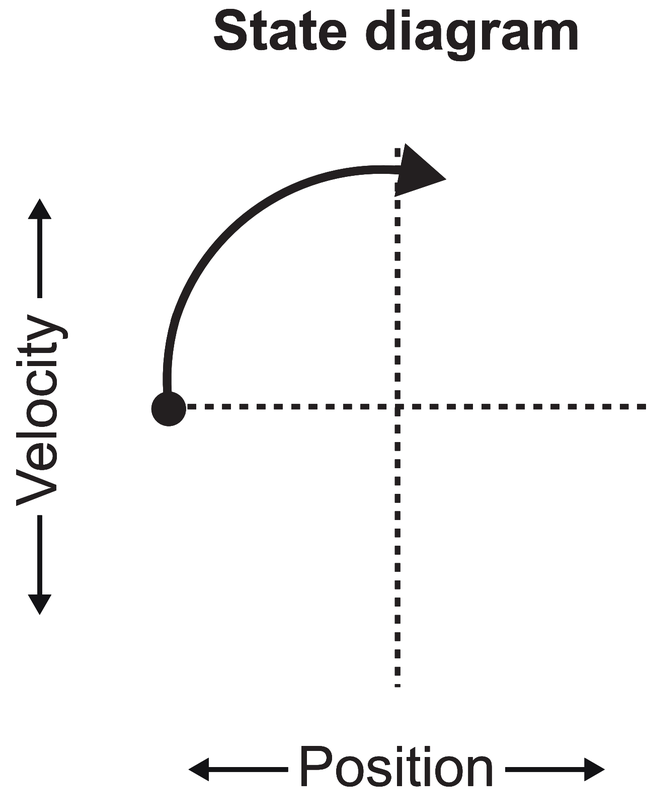


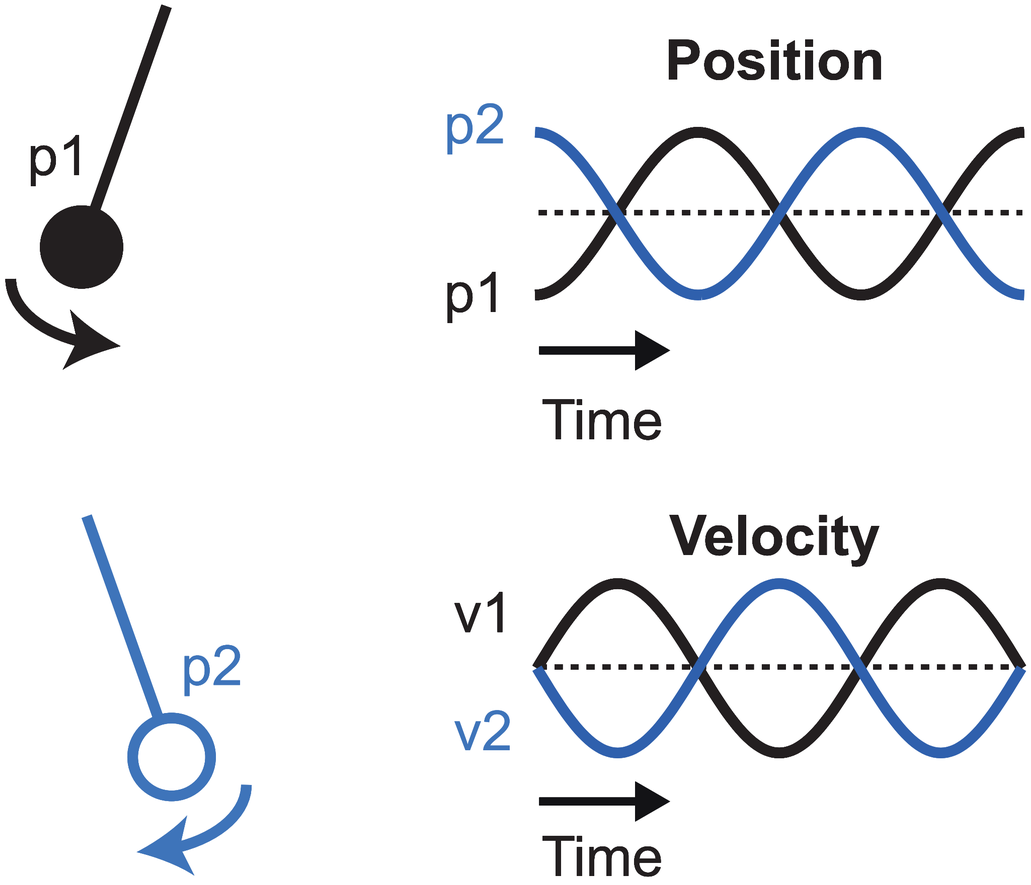
$$\mathbf{x} = \begin{bmatrix} p \\ v \end{bmatrix}$$



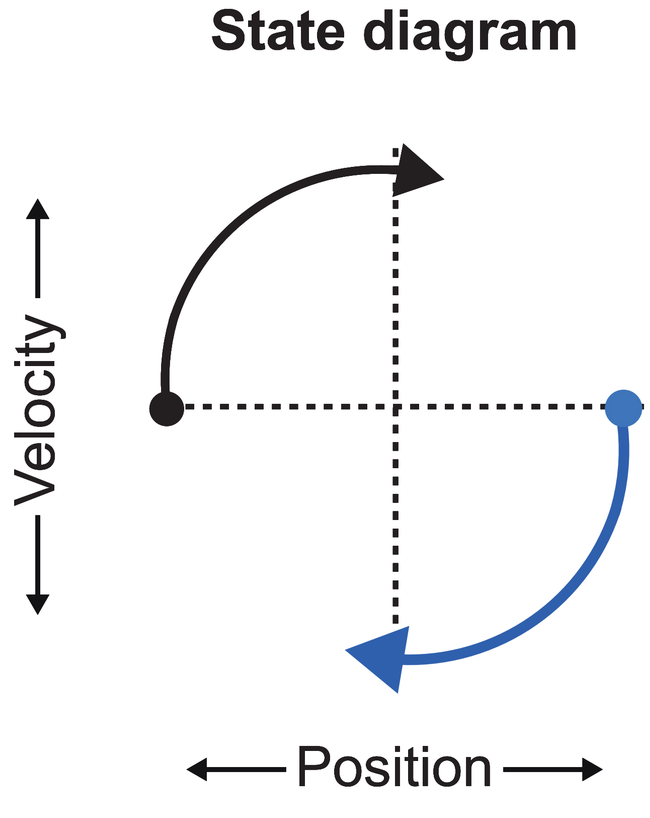


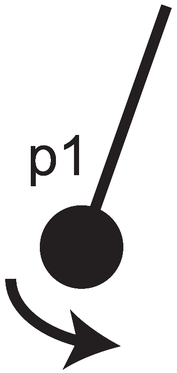
$$\mathbf{x} = \begin{bmatrix} p \\ v \end{bmatrix}$$

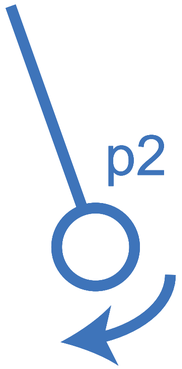


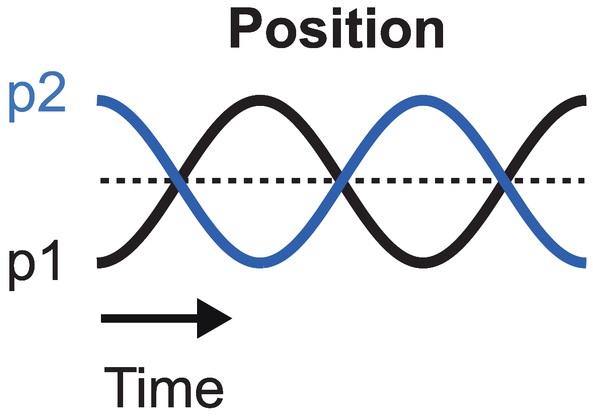


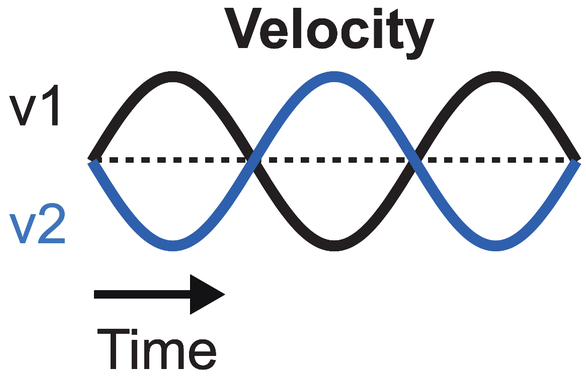
$$\mathbf{x} = \begin{bmatrix} p \\ v \end{bmatrix}$$





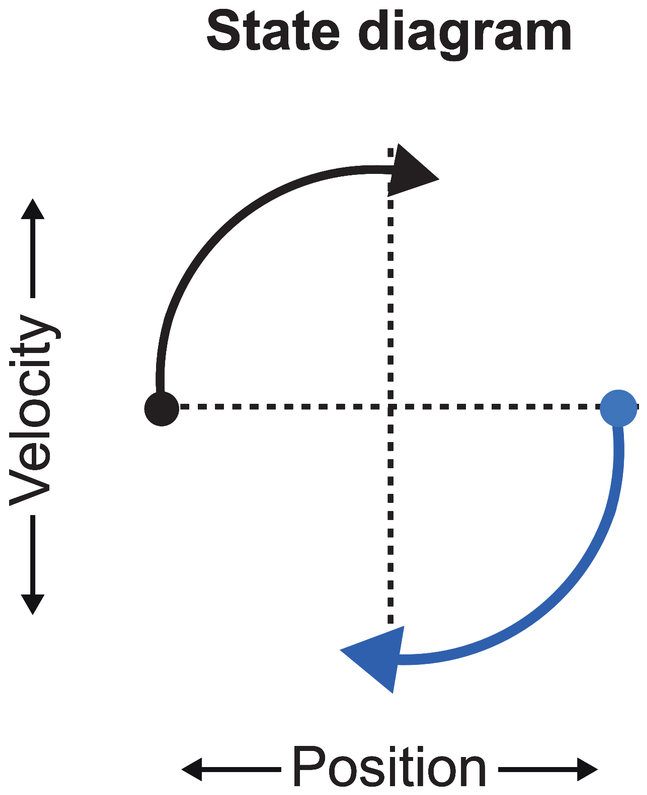


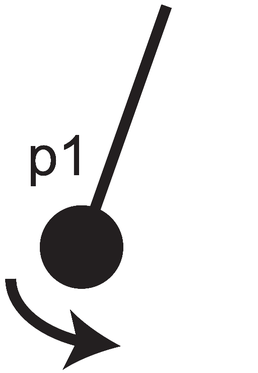


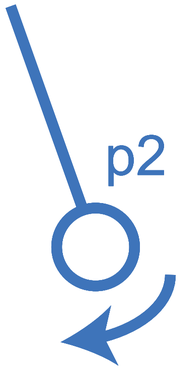


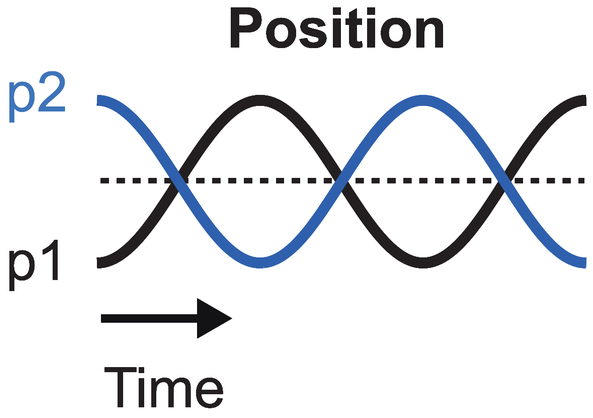
$$\mathbf{x} = \begin{bmatrix} p \\ v \end{bmatrix}$$

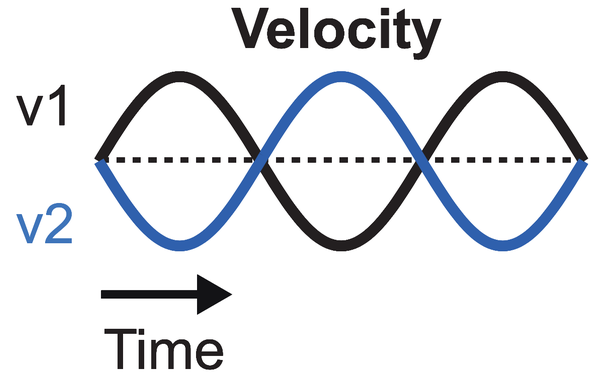
$$\dot{\mathbf{x}} = f(\mathbf{x})$$





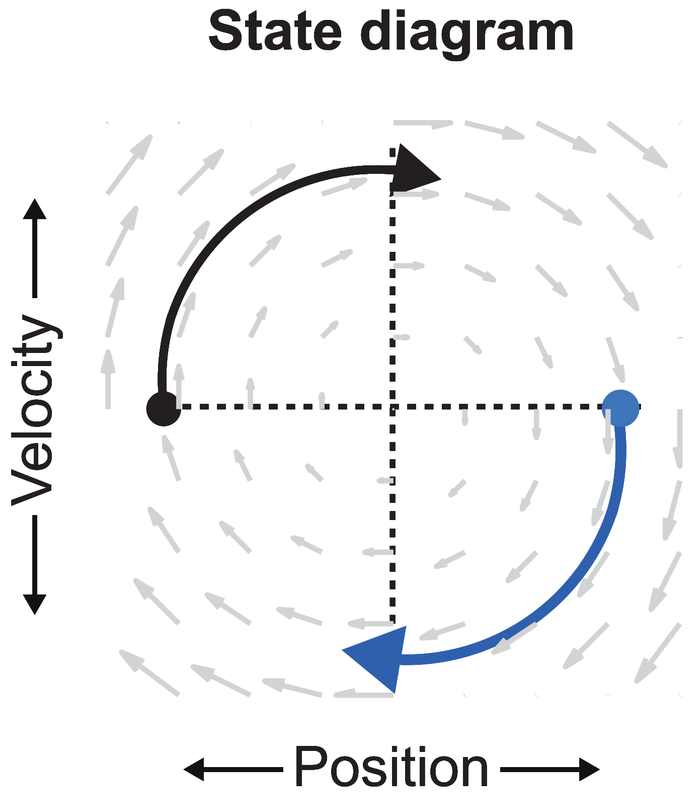






$$\mathbf{x} = \begin{bmatrix} p \\ v \end{bmatrix}$$

$$\dot{\mathbf{x}} = f(\mathbf{x})$$

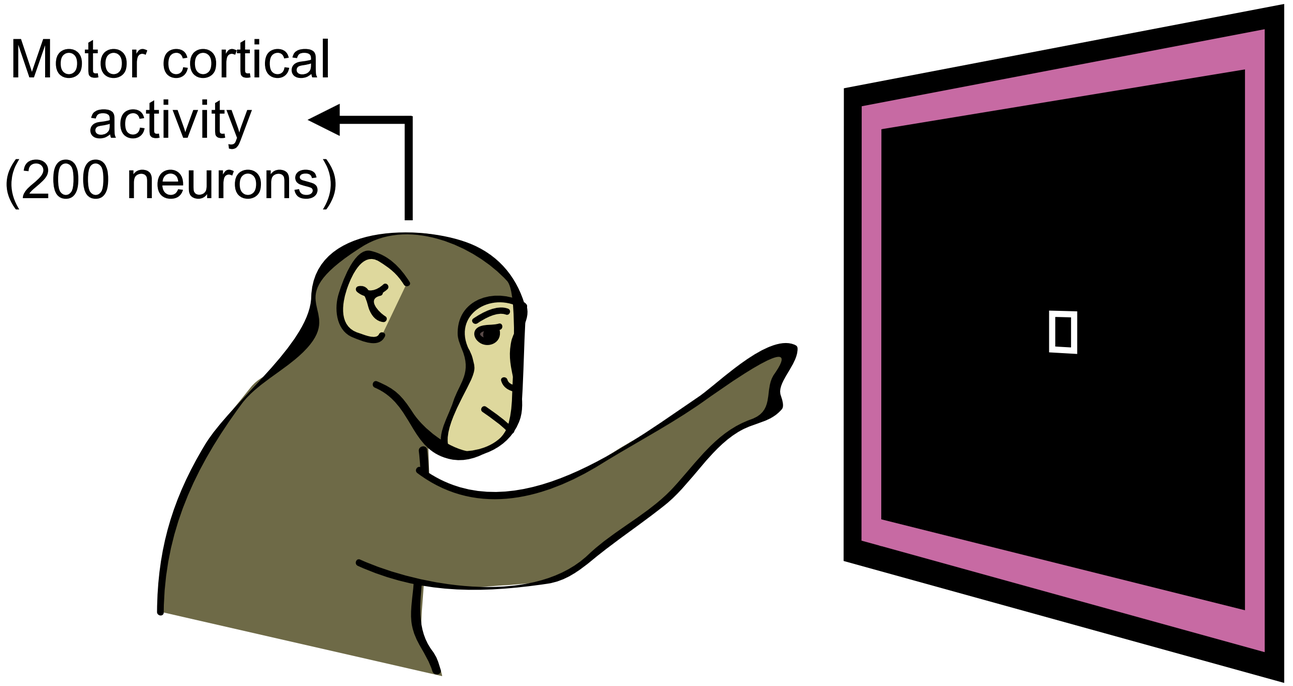


Predictable activity: delayed-reaching

"Maze" task

108 total conditions

- Motor cortex is set to an initial state during the preparatory phase
- Activity during movement execution is highly predictable based on initial state



Churchland*, Cunningham* … Shenoy, *Nature* 2012

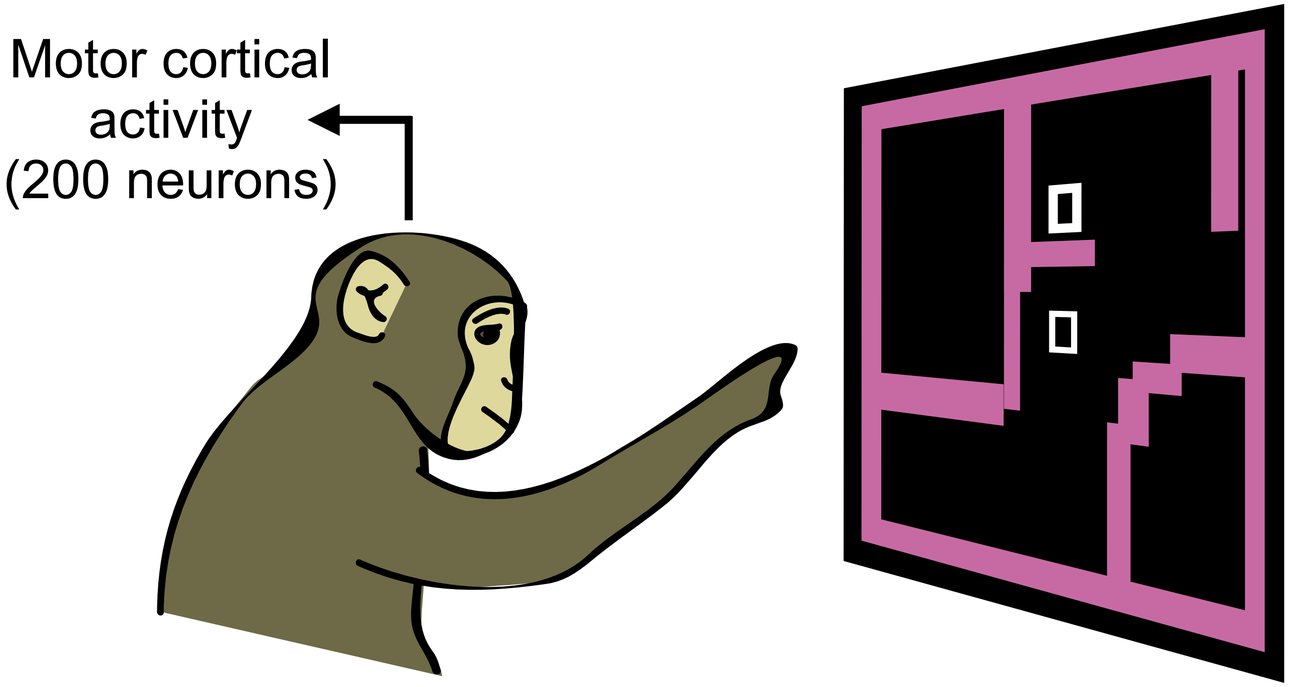
Shenoy, Sahani, Churchland, *Ann Rev Neuro* 2013

Predictable activity: delayed-reaching

"Maze" task

108 total conditions

- Motor cortex is set to an initial state during the preparatory phase
- Activity during movement execution is highly predictable based on initial state



Condition 1

Churchland*, Cunningham* … Shenoy, *Nature* 2012

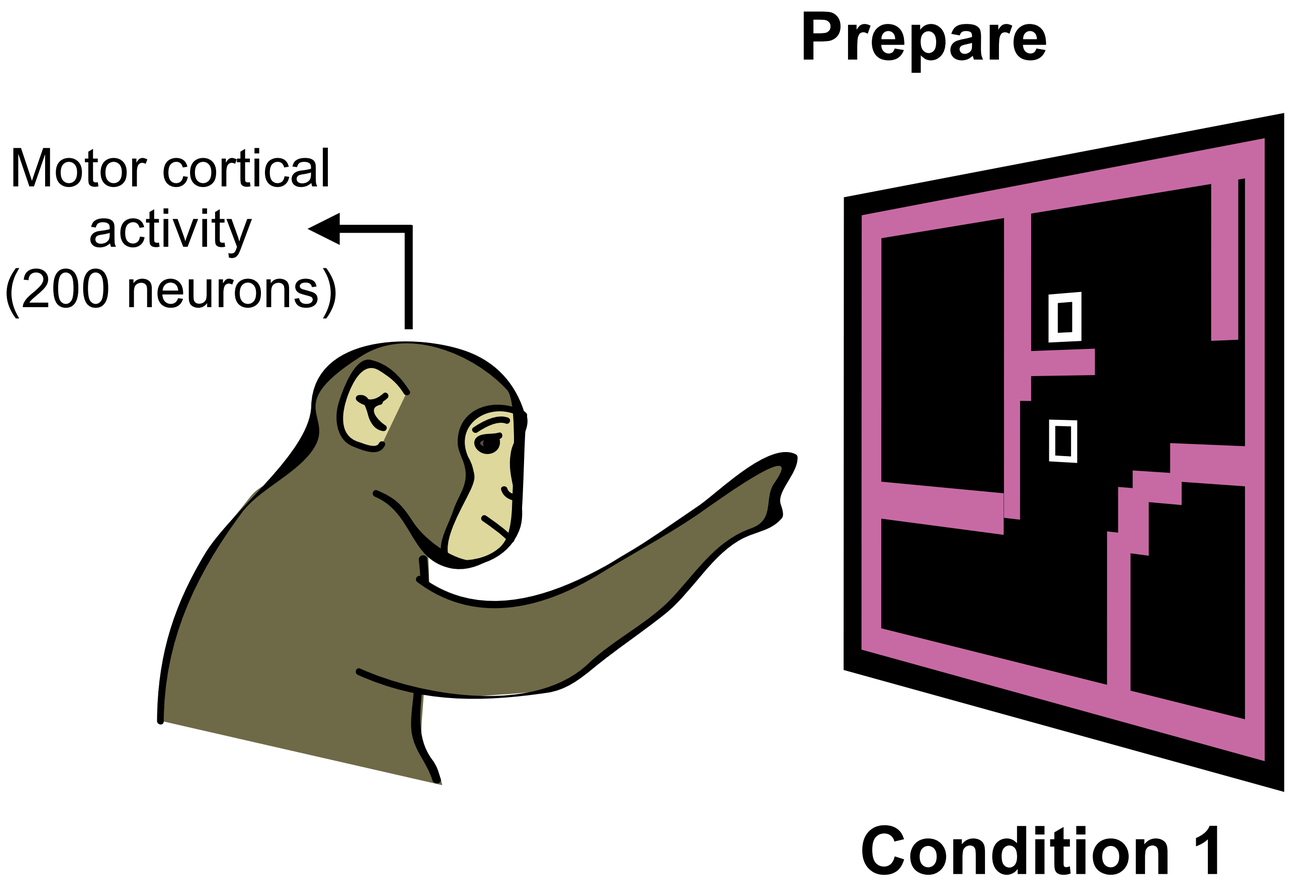
Shenoy, Sahani, Churchland, *Ann Rev Neuro* 2013

Predictable activity: delayed-reaching

"Maze" task

108 total conditions

- Motor cortex is set to an initial state during the preparatory phase
- Activity during movement execution is highly predictable based on initial state



Churchland*, Cunningham* … Shenoy, *Nature* 2012

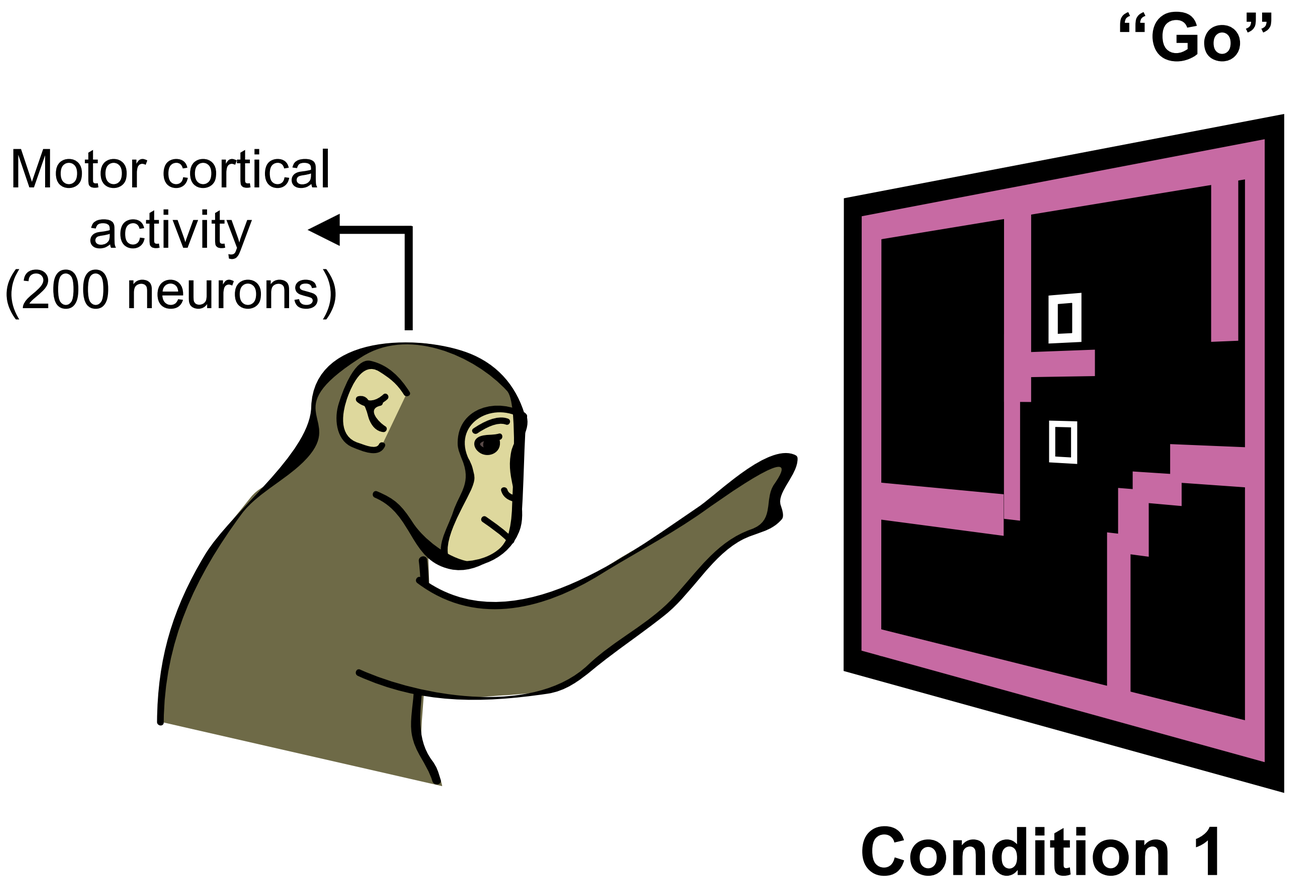
Shenoy, Sahani, Churchland, *Ann Rev Neuro* 2013

Predictable activity: delayed-reaching

"Maze" task

108 total conditions

- Motor cortex is set to an initial state during the preparatory phase
- Activity during movement execution is highly predictable based on initial state



Churchland*, Cunningham* … Shenoy, *Nature* 2012

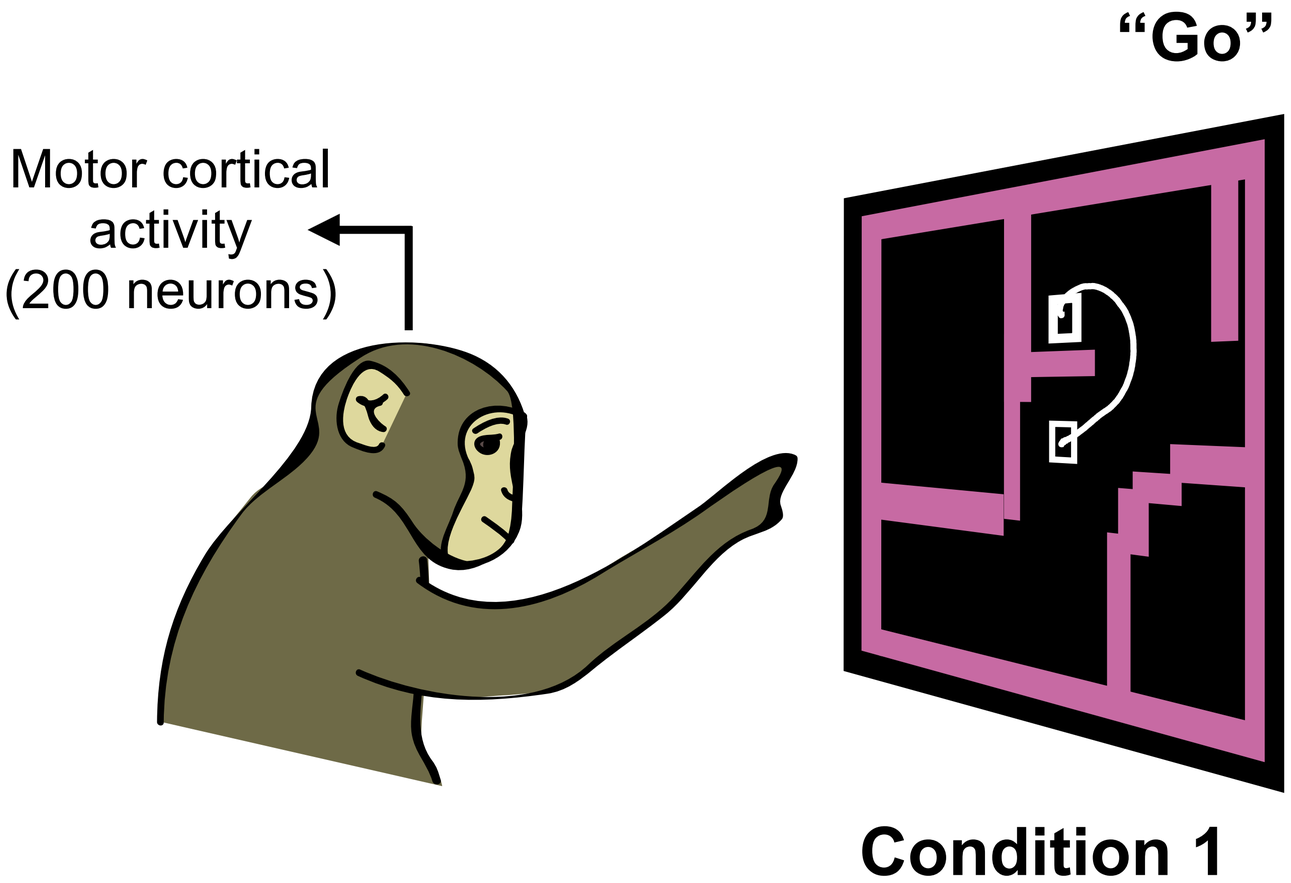
Shenoy, Sahani, Churchland, *Ann Rev Neuro* 2013

Predictable activity: delayed-reaching

"Maze" task

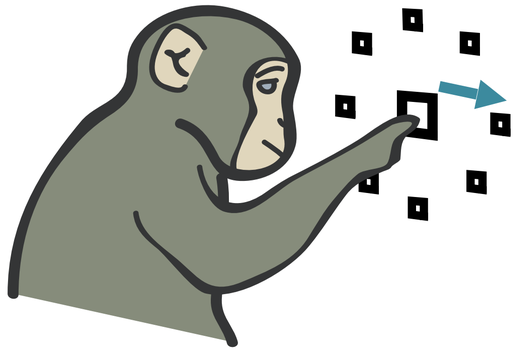
108 total conditions

- Motor cortex is set to an initial state during the preparatory phase
- Activity during movement execution is highly predictable based on initial state

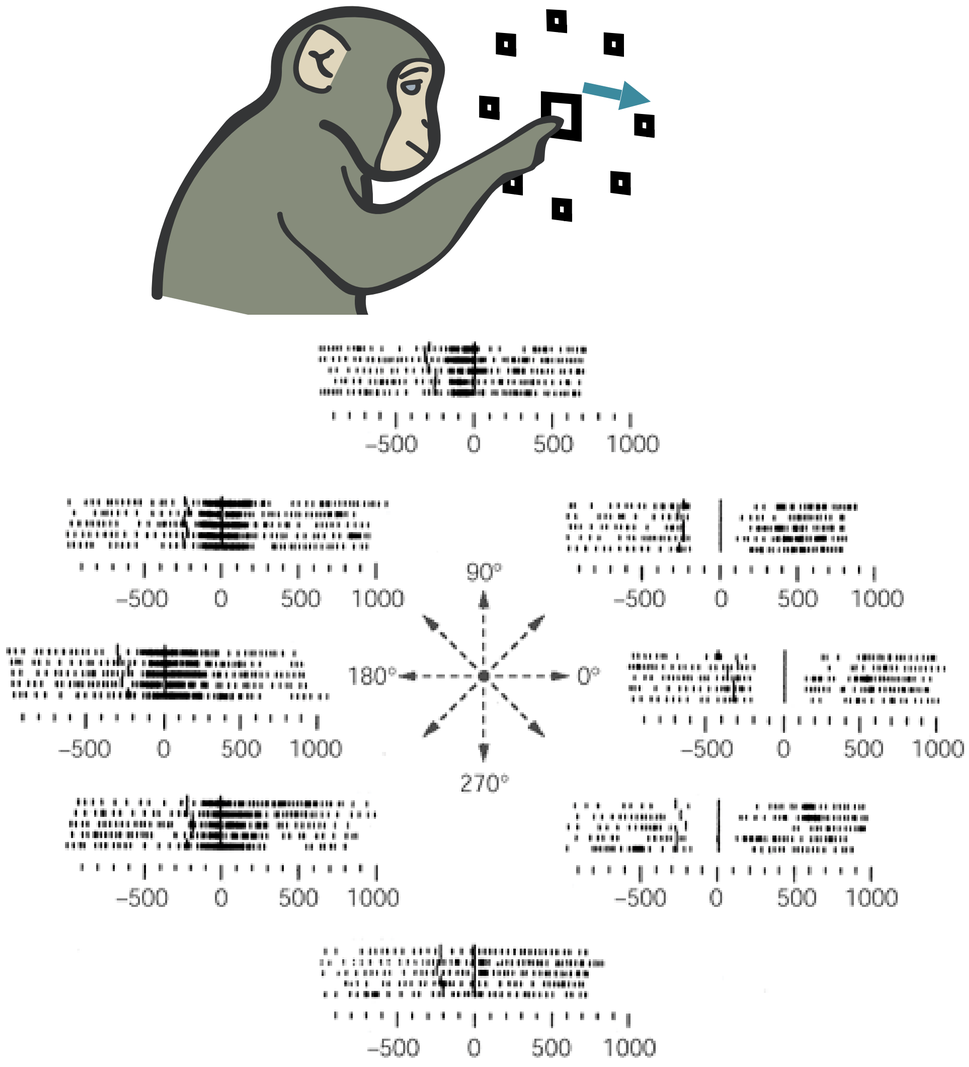


Churchland*, Cunningham* … Shenoy, *Nature* 2012

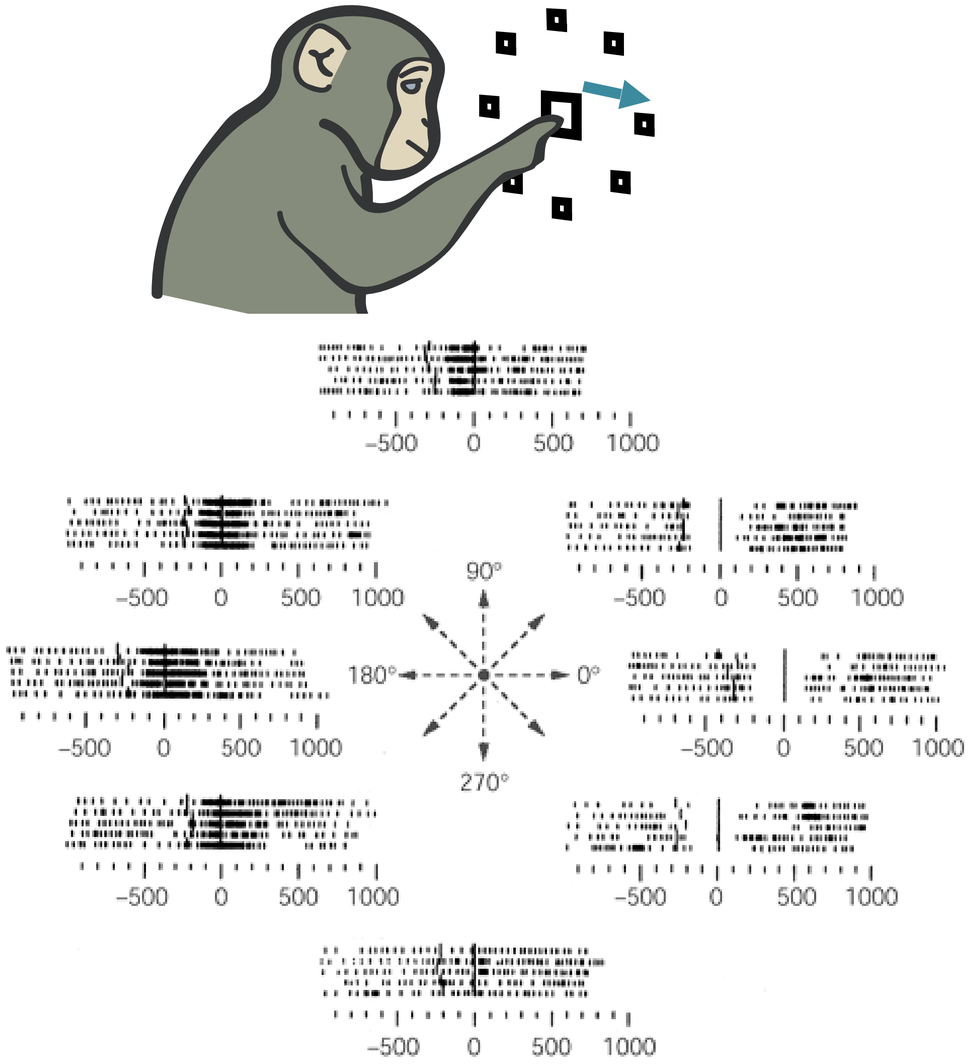
Shenoy, Sahani, Churchland, *Ann Rev Neuro* 2013

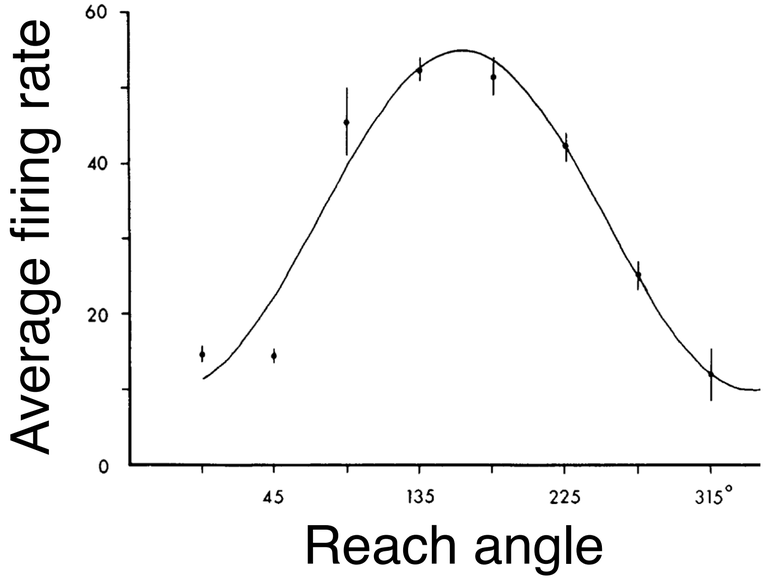


Georgopoulous et al. (1982)
Schwartz et al. (1988)

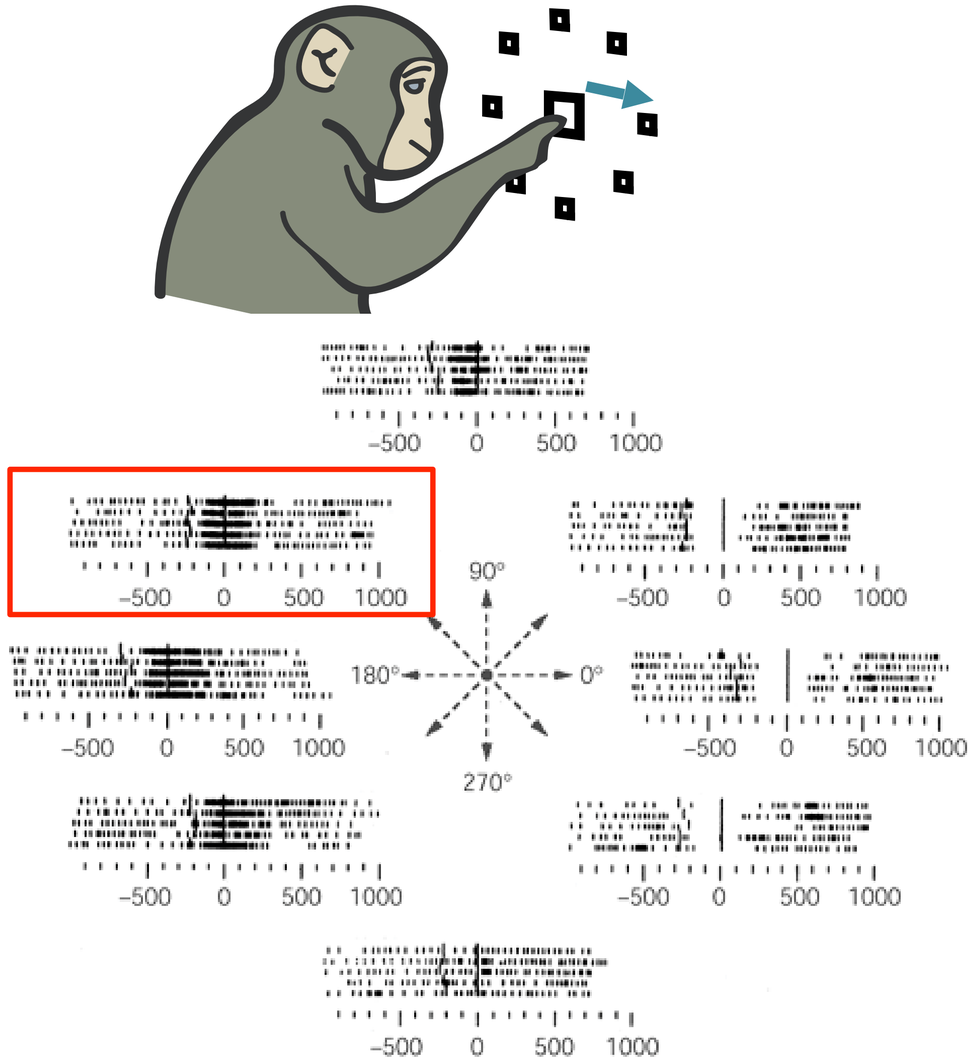


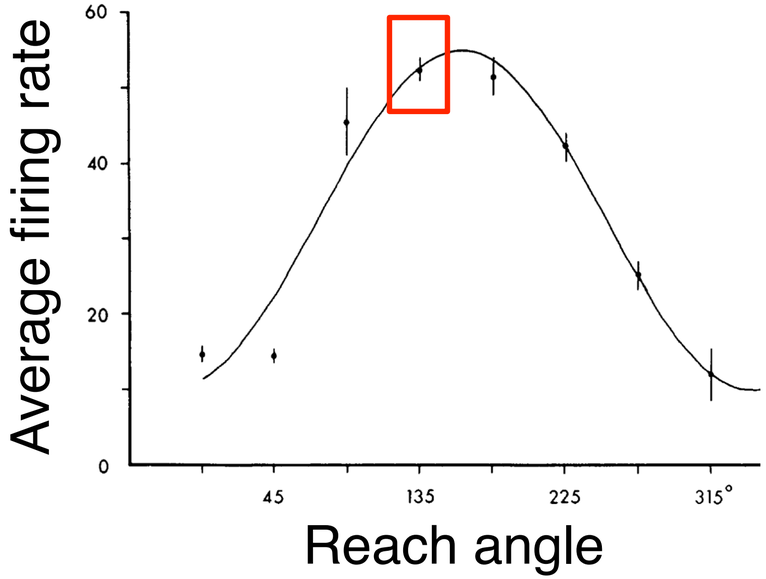
Georgopoulous et al. (1982)
Schwartz et al. (1988)



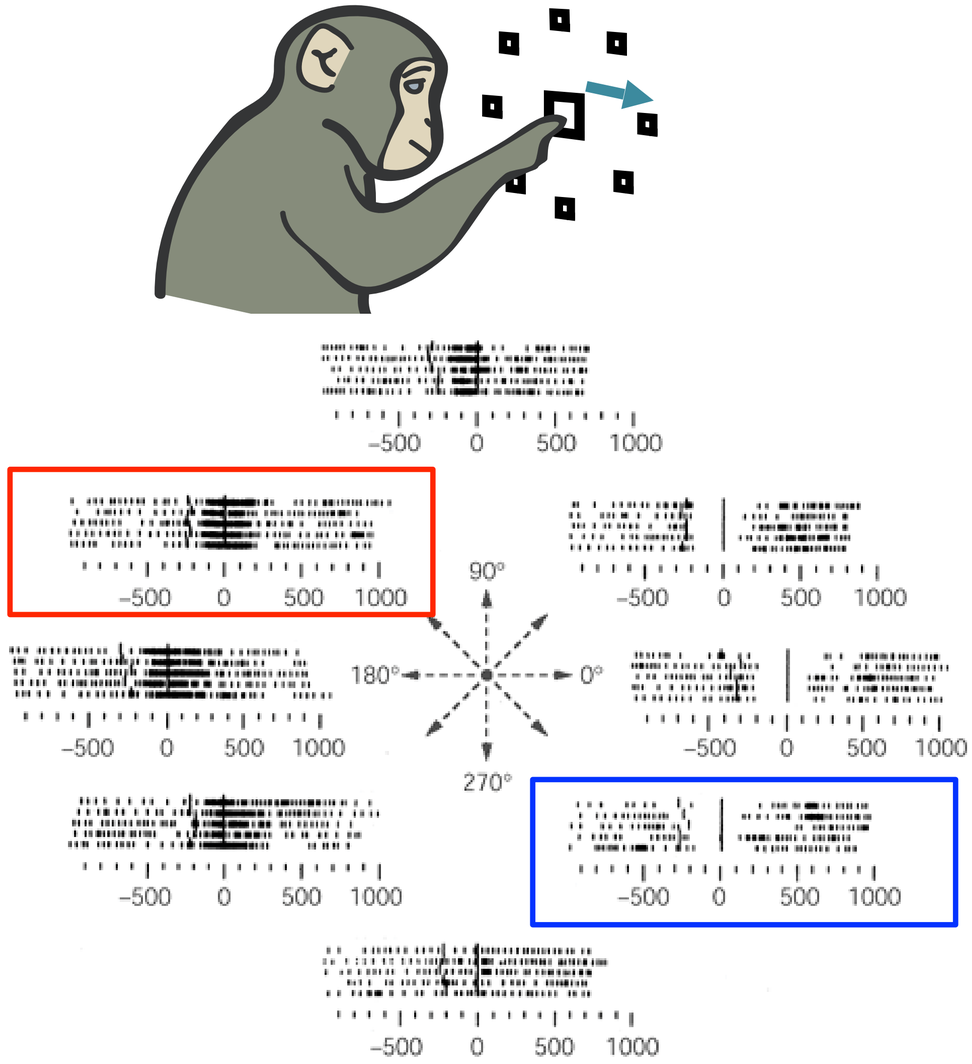


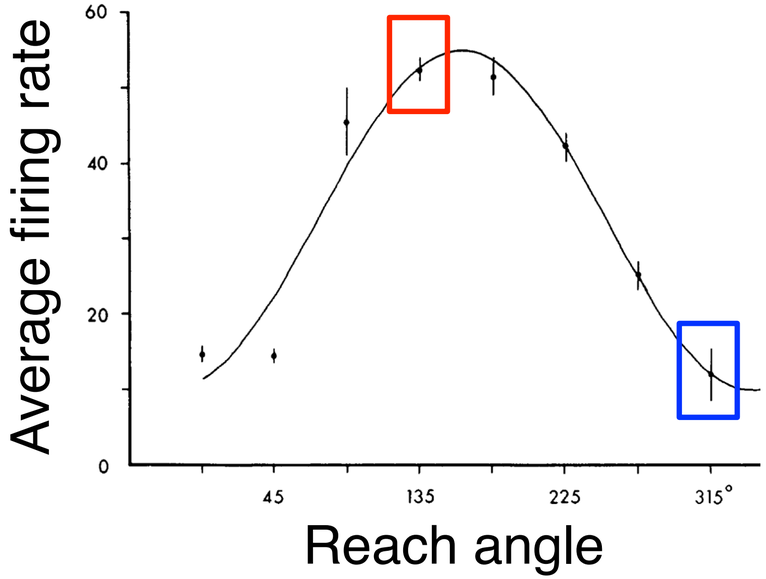
Georgopoulous et al. (1982)
Schwartz et al. (1988)



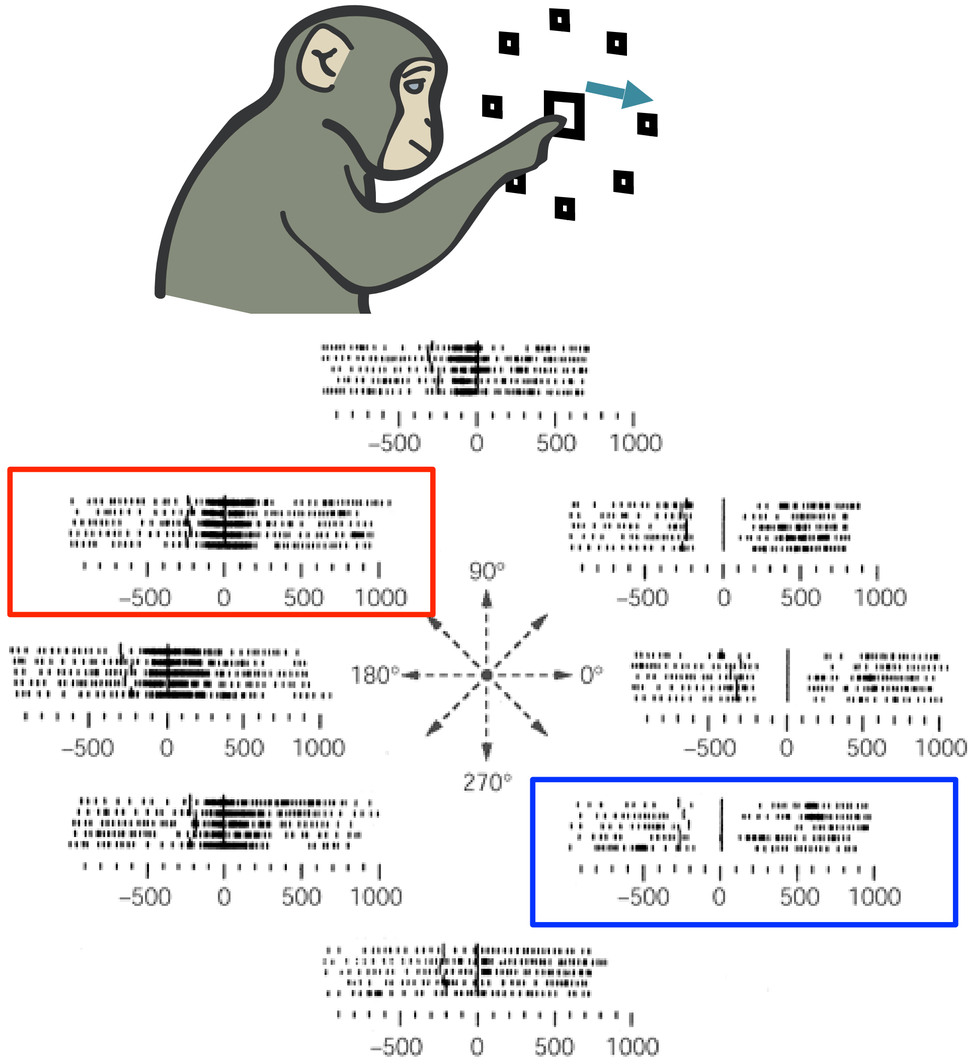


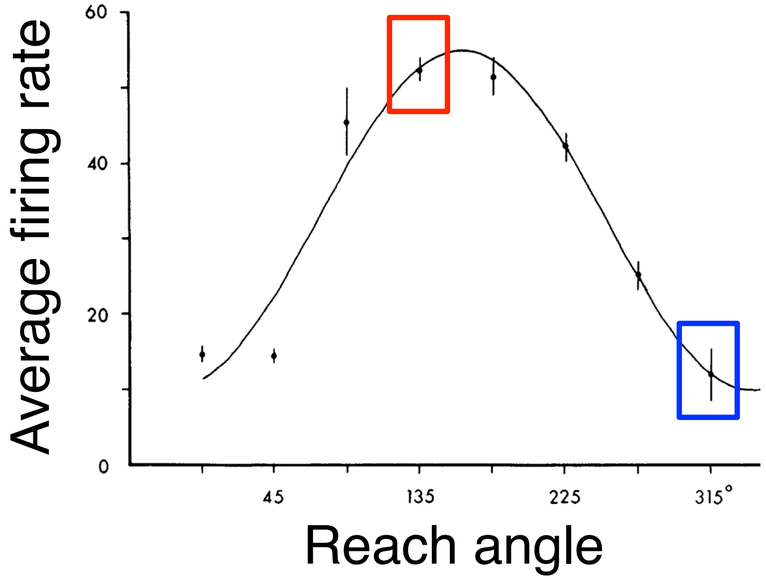
Georgopoulous et al. (1982)
Schwartz et al. (1988)





Georgopoulous et al. (1982)
Schwartz et al. (1988)





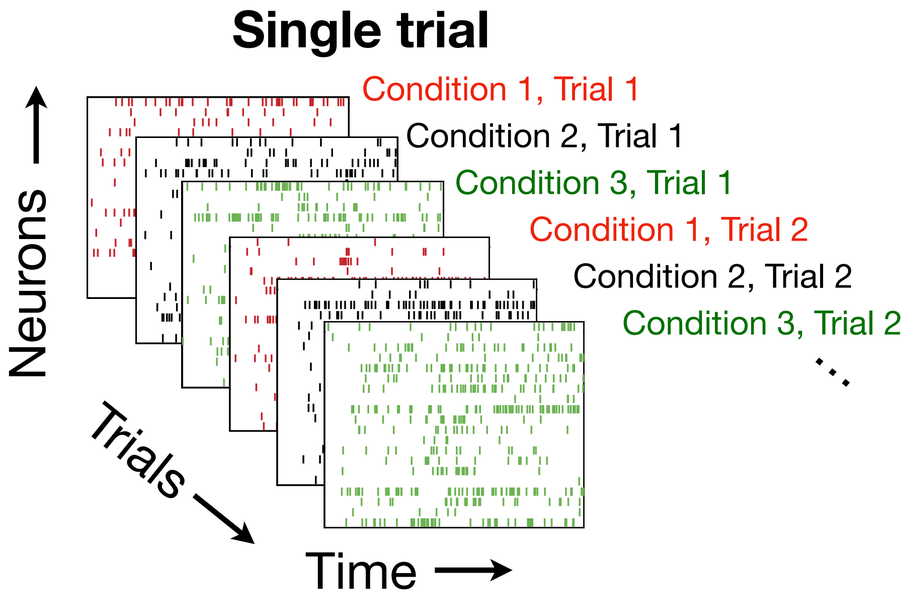
$$\mathbf{z} = f(\mathbf{k})$$

Neurons' firing rate

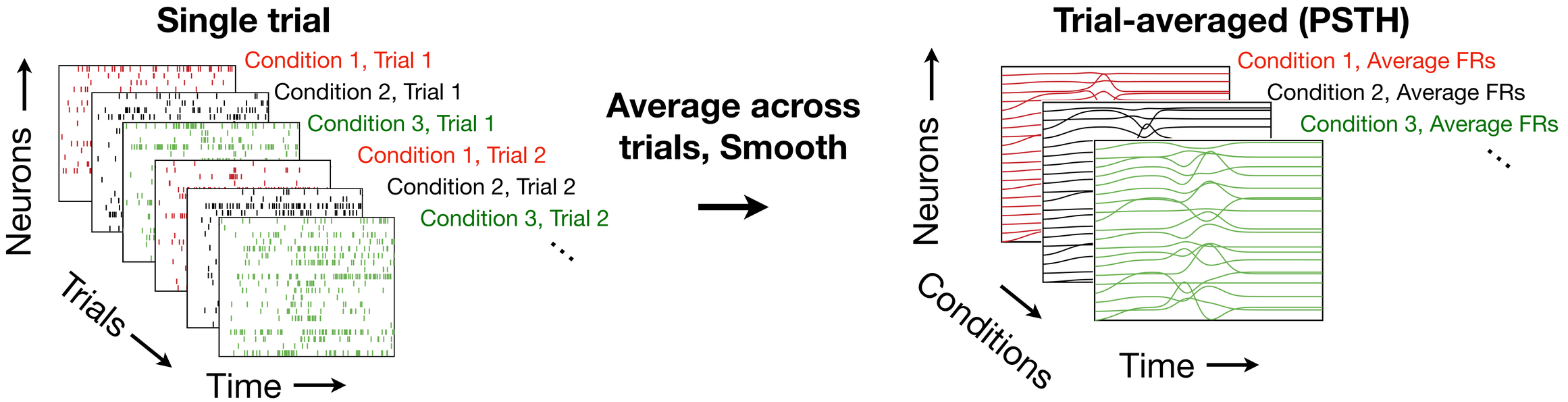
Kinematics or kinetics

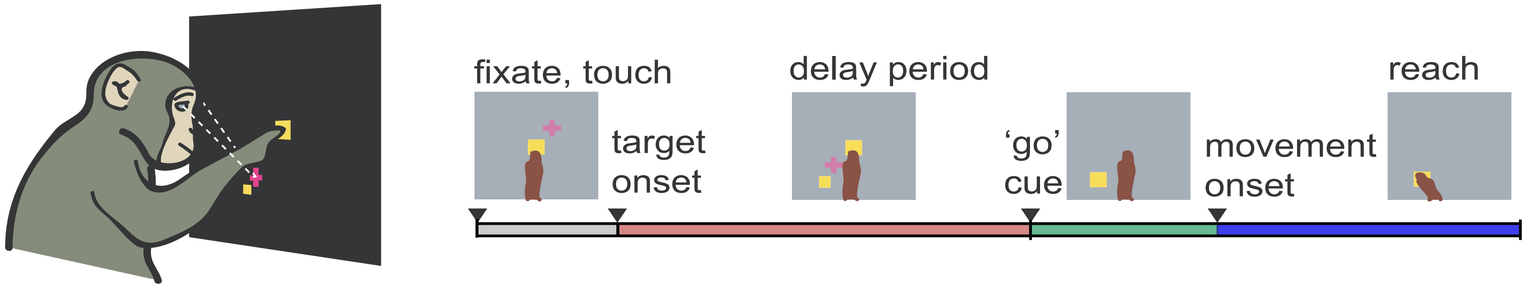
Georgopoulous et al. (1982)
Schwartz et al. (1988)

Visualizing the data

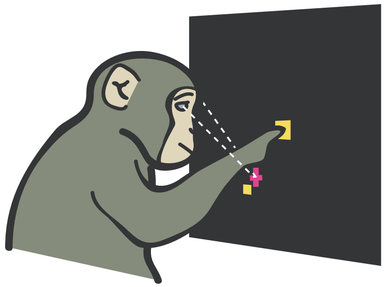


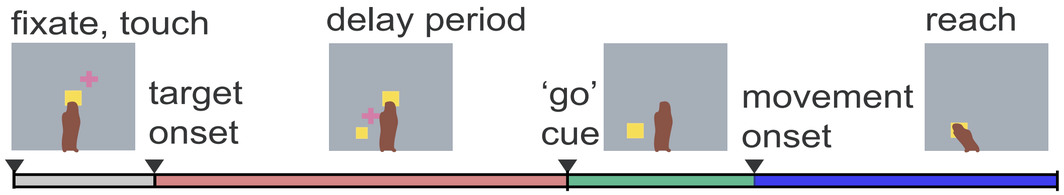
Visualizing the data





Churchland*, Cunningham* … Shenoy, *Nature* (2012)

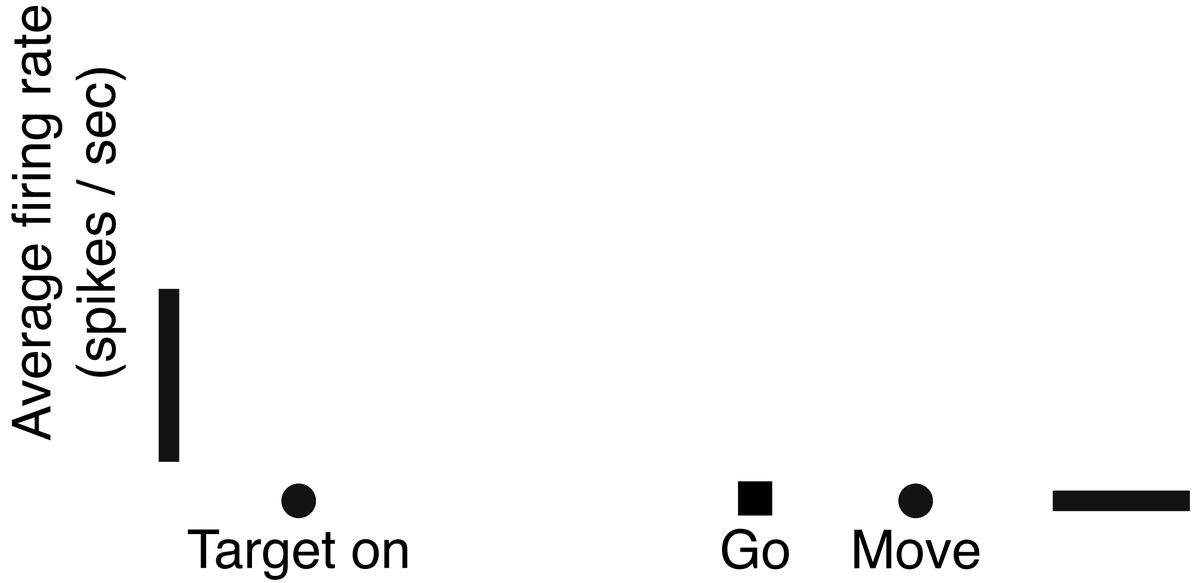




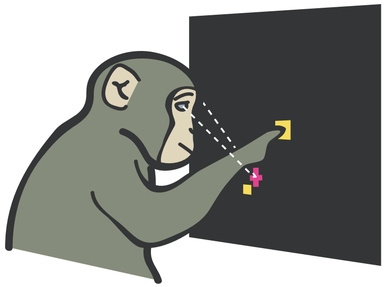
Reach trajectories

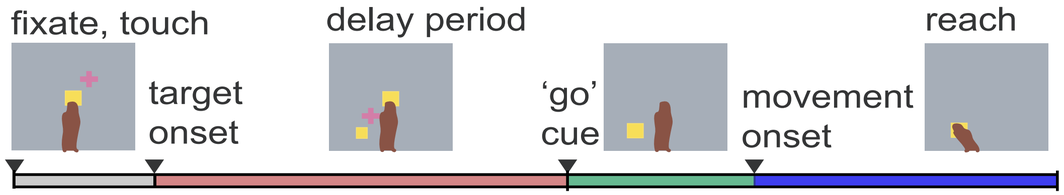
Single neuron firing rate (PSTH)

Cell 12
Monkey B

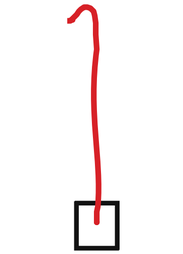


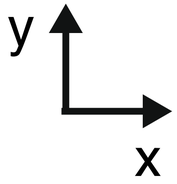
Churchland*, Cunningham* … Shenoy, *Nature* (2012)





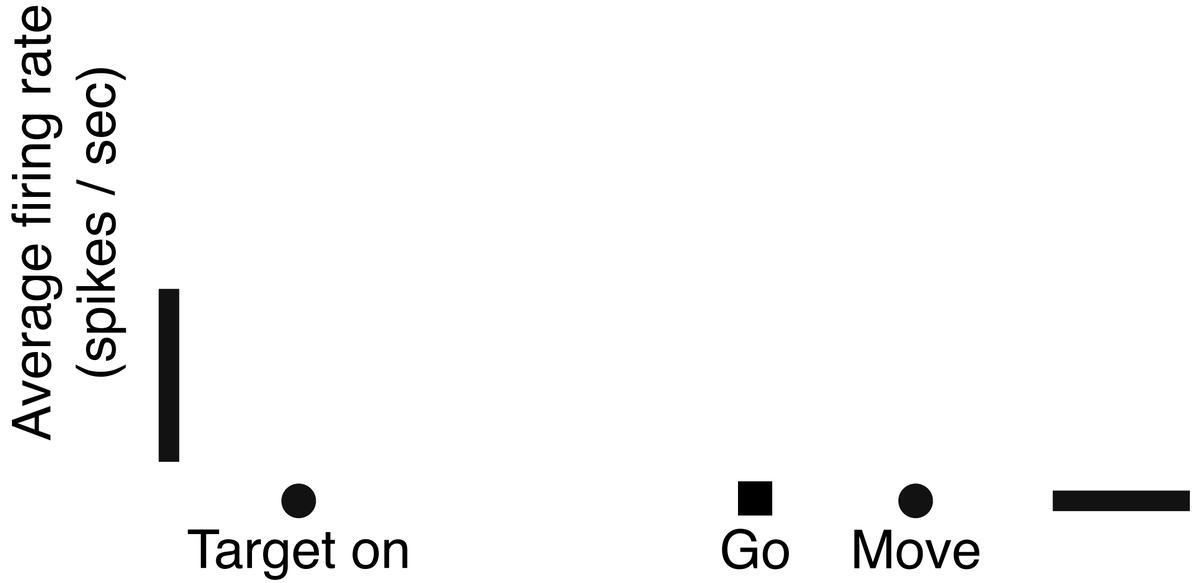
Reach trajectories



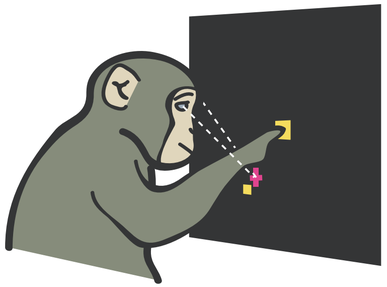


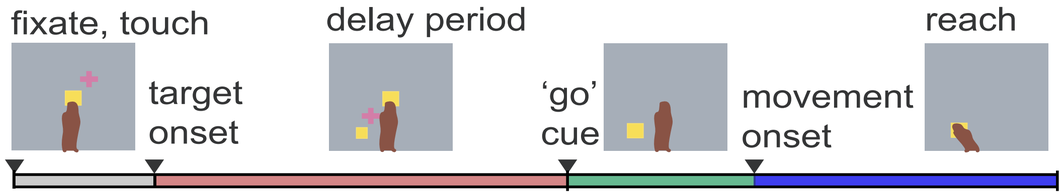
Single neuron firing rate (PSTH)

Cell 12
Monkey B

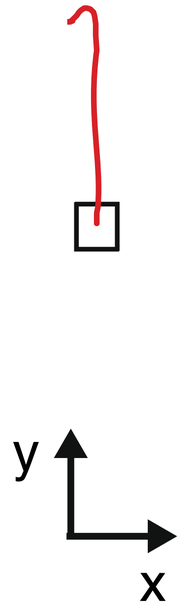


Churchland*, Cunningham* … Shenoy, *Nature* (2012)



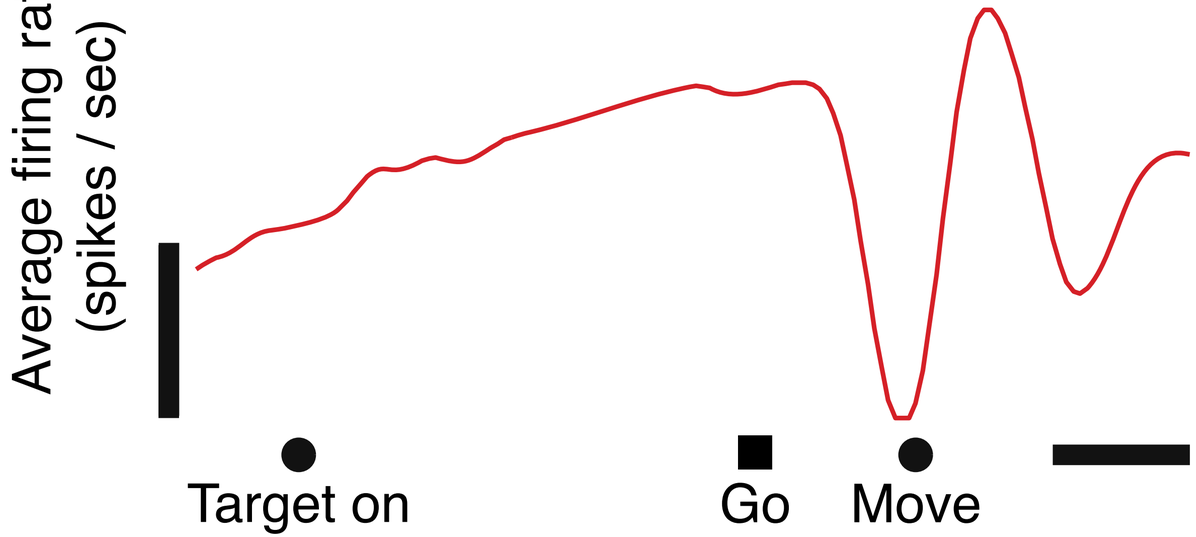


Reach trajectories

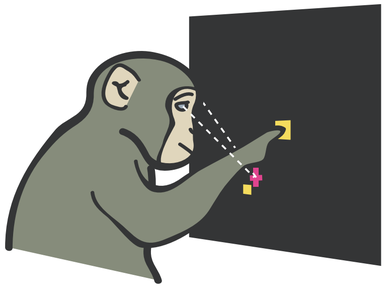


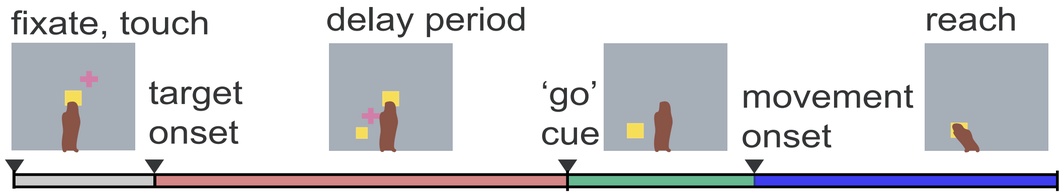
Single neuron firing rate (PSTH)

Cell 12
Monkey B

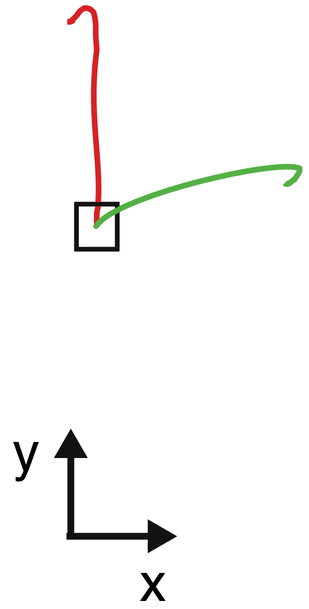


Churchland*, Cunningham* … Shenoy, *Nature* (2012)



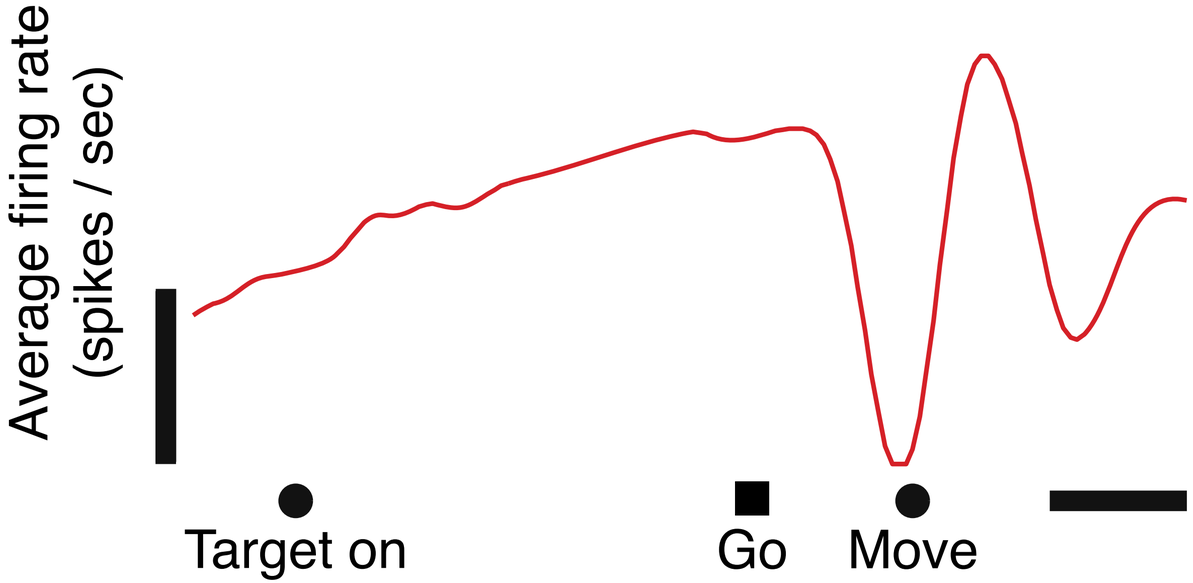


Reach trajectories

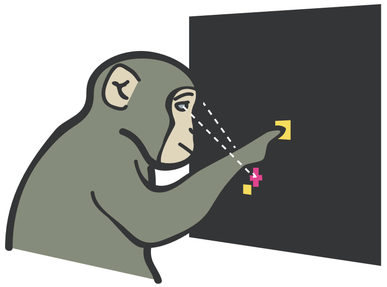


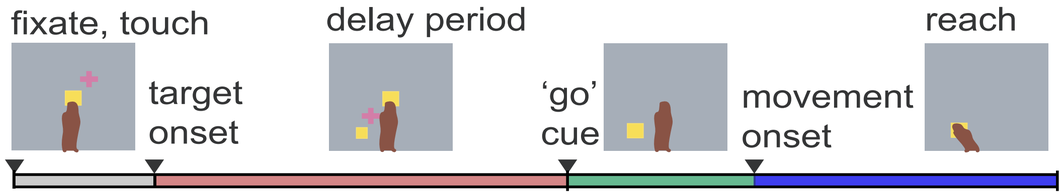
Single neuron firing rate (PSTH)

Cell 12
Monkey B

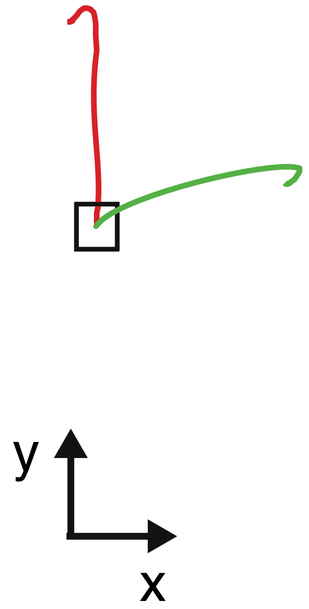


Churchland*, Cunningham* … Shenoy, *Nature* (2012)

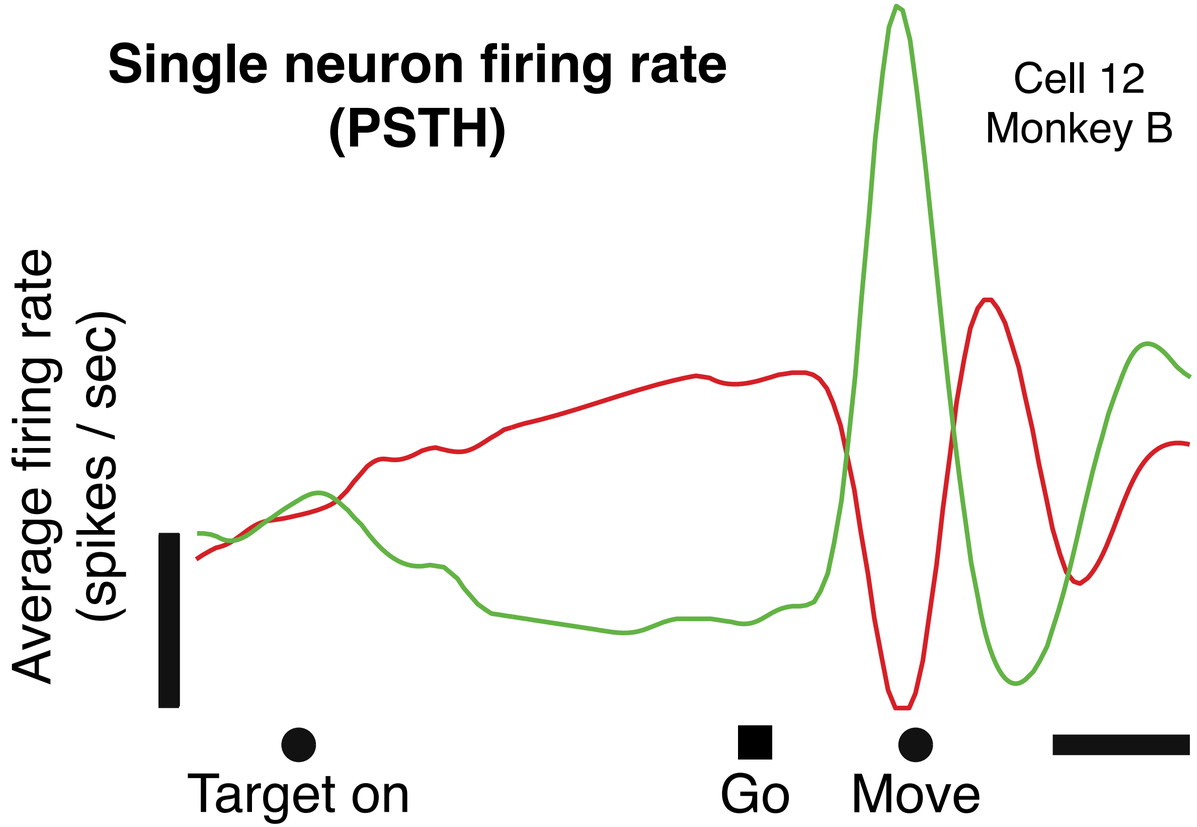




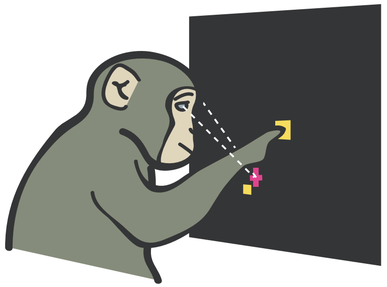
Reach trajectories

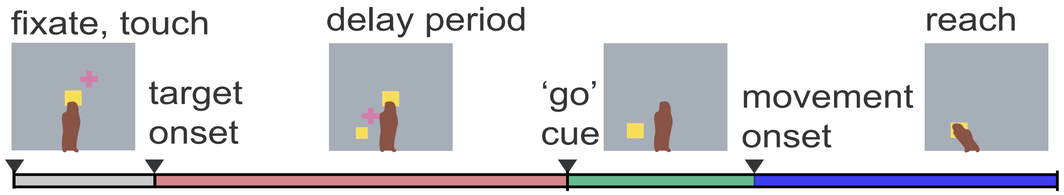


Single neuron firing rate (PSTH)

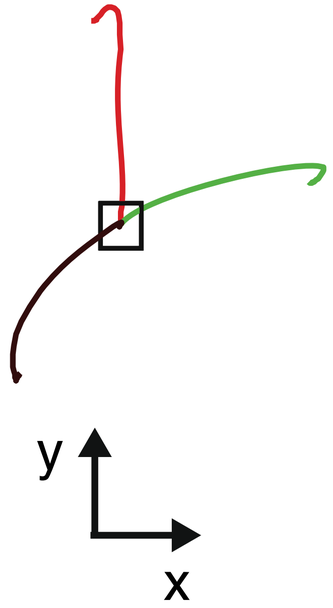


Churchland*, Cunningham* … Shenoy, *Nature* (2012)

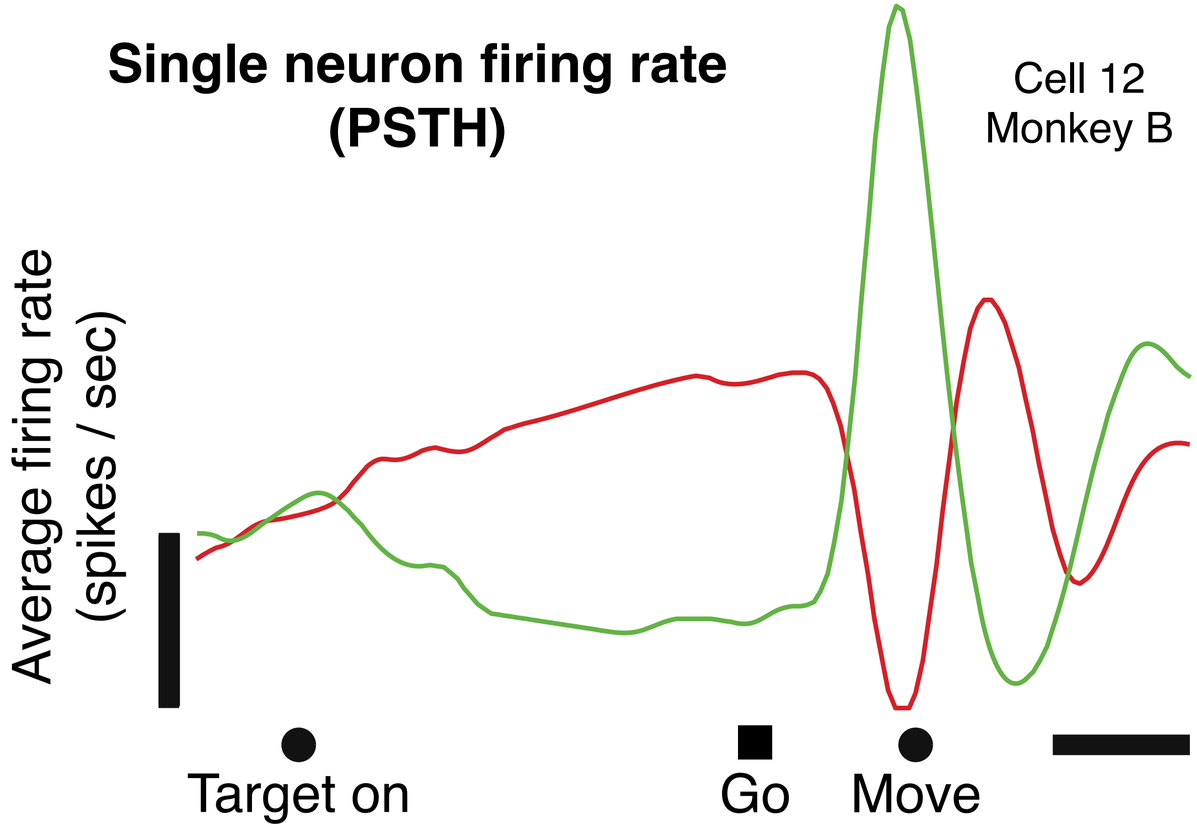




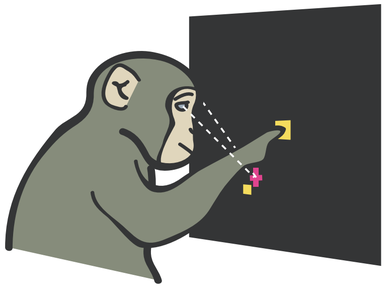
Reach trajectories

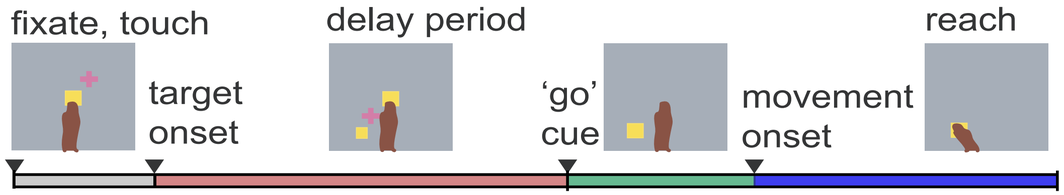


Single neuron firing rate (PSTH)

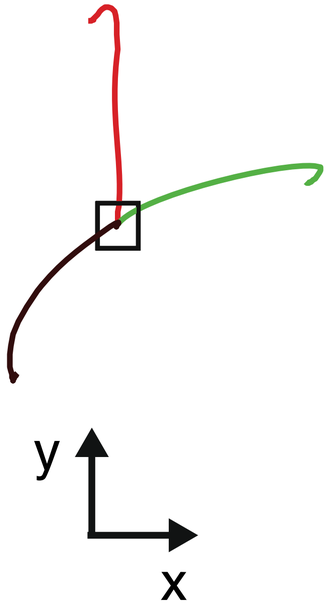


Churchland*, Cunningham* … Shenoy, *Nature* (2012)

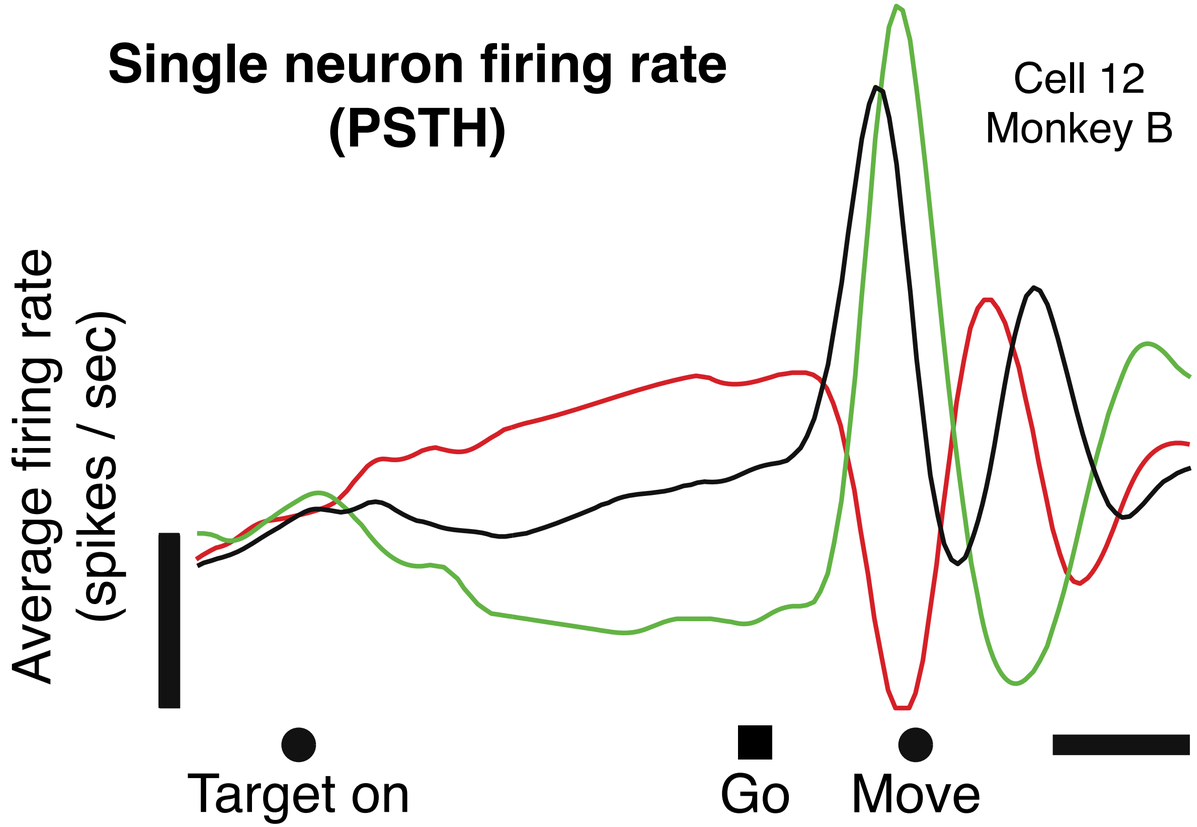




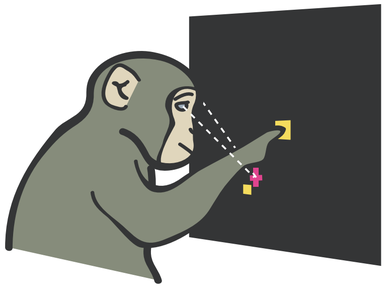
Reach trajectories

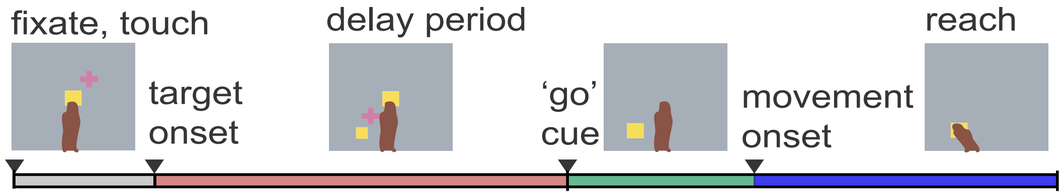


Single neuron firing rate (PSTH)

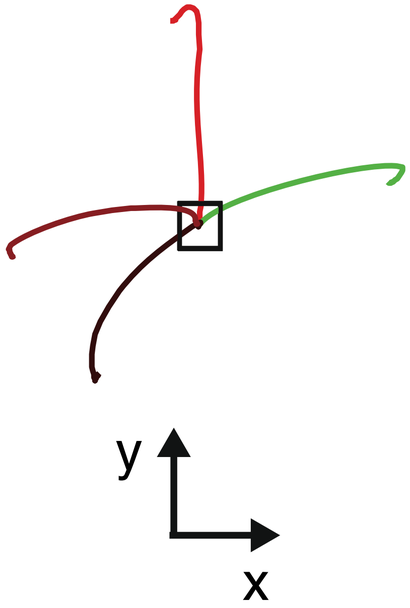


Churchland*, Cunningham* … Shenoy, *Nature* (2012)

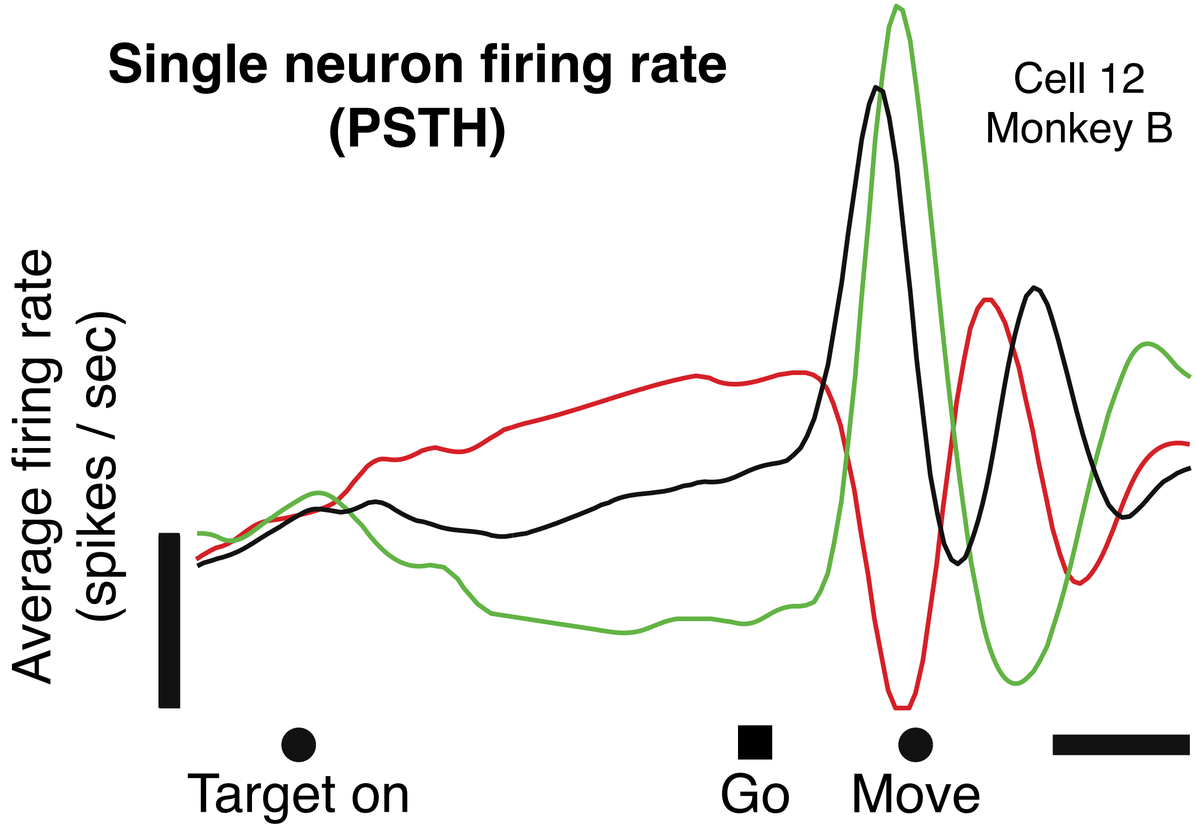




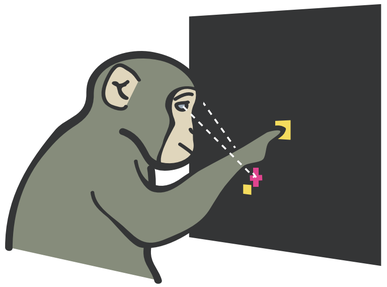
Reach trajectories

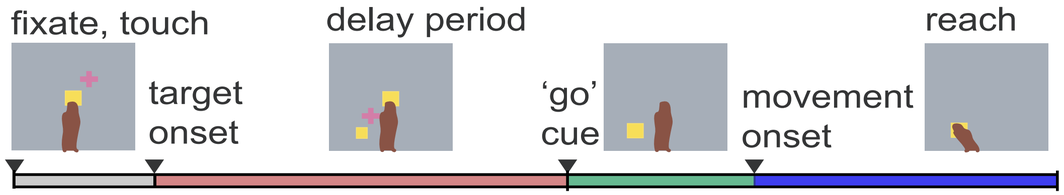


Single neuron firing rate (PSTH)

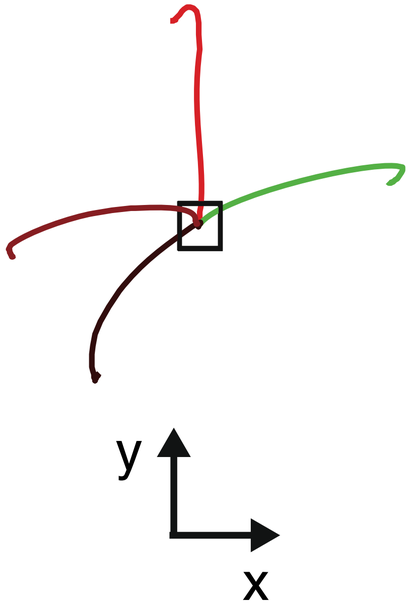


Churchland*, Cunningham* … Shenoy, *Nature* (2012)

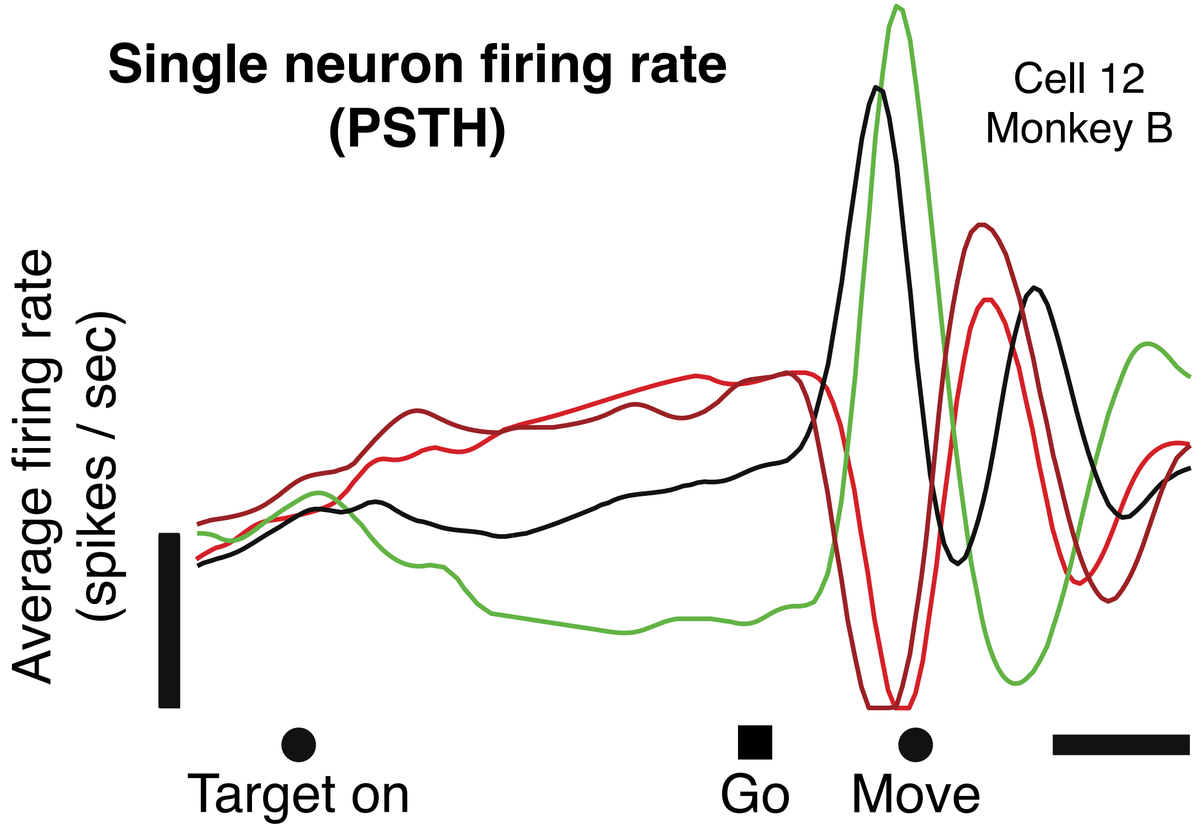




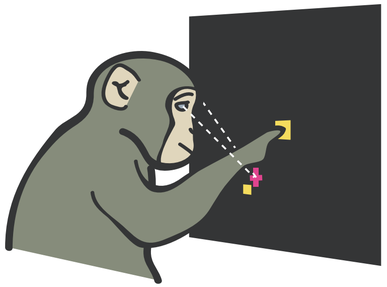
Reach trajectories

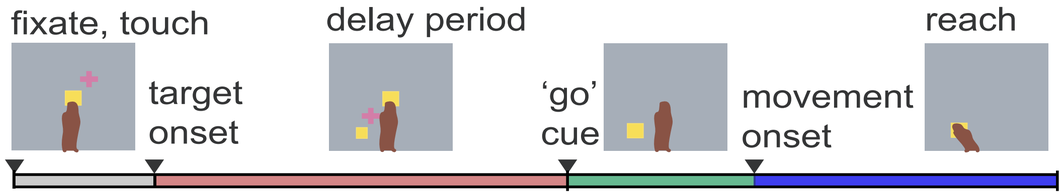


Single neuron firing rate (PSTH)

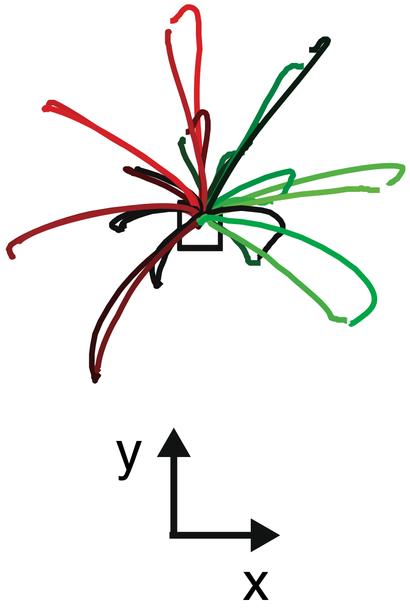


Churchland*, Cunningham* … Shenoy, *Nature* (2012)

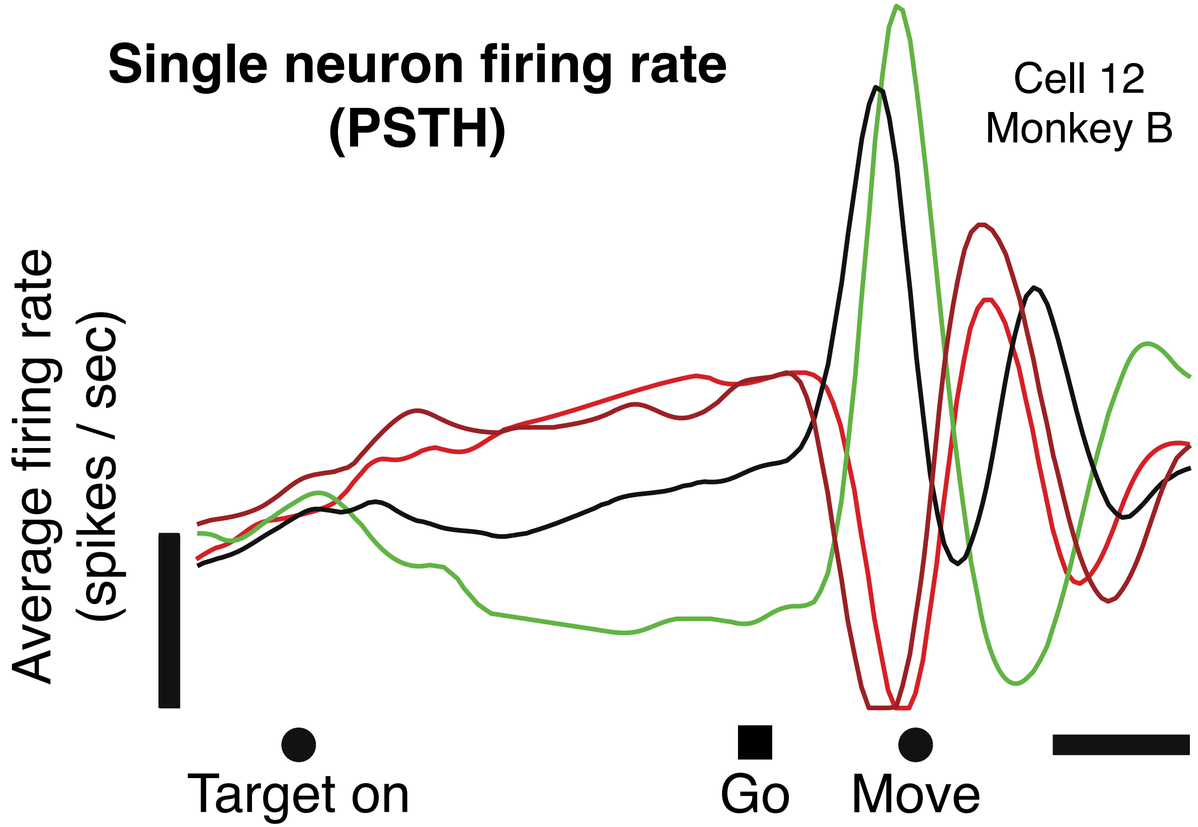




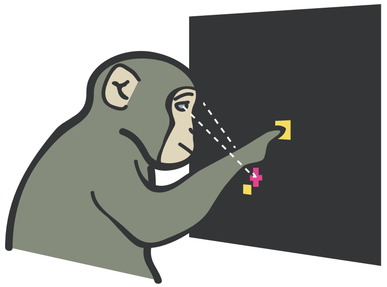
Reach trajectories

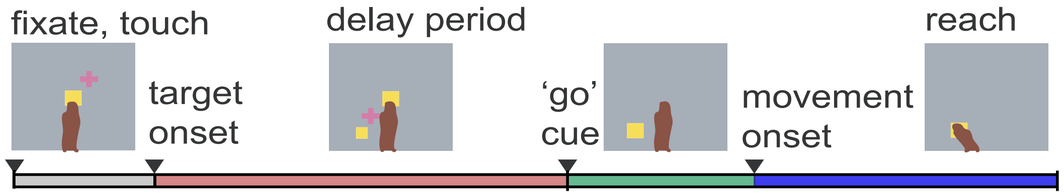


Single neuron firing rate (PSTH)

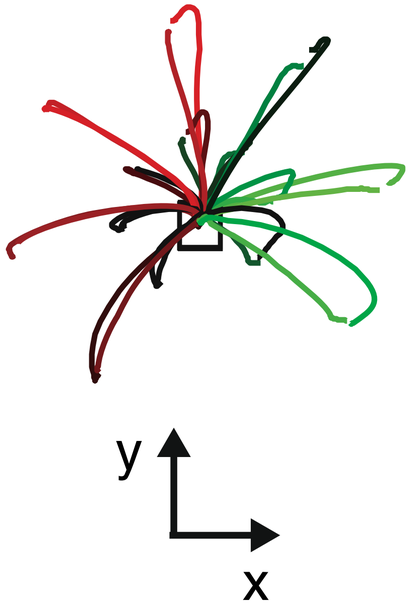


Churchland*, Cunningham* … Shenoy, *Nature* (2012)

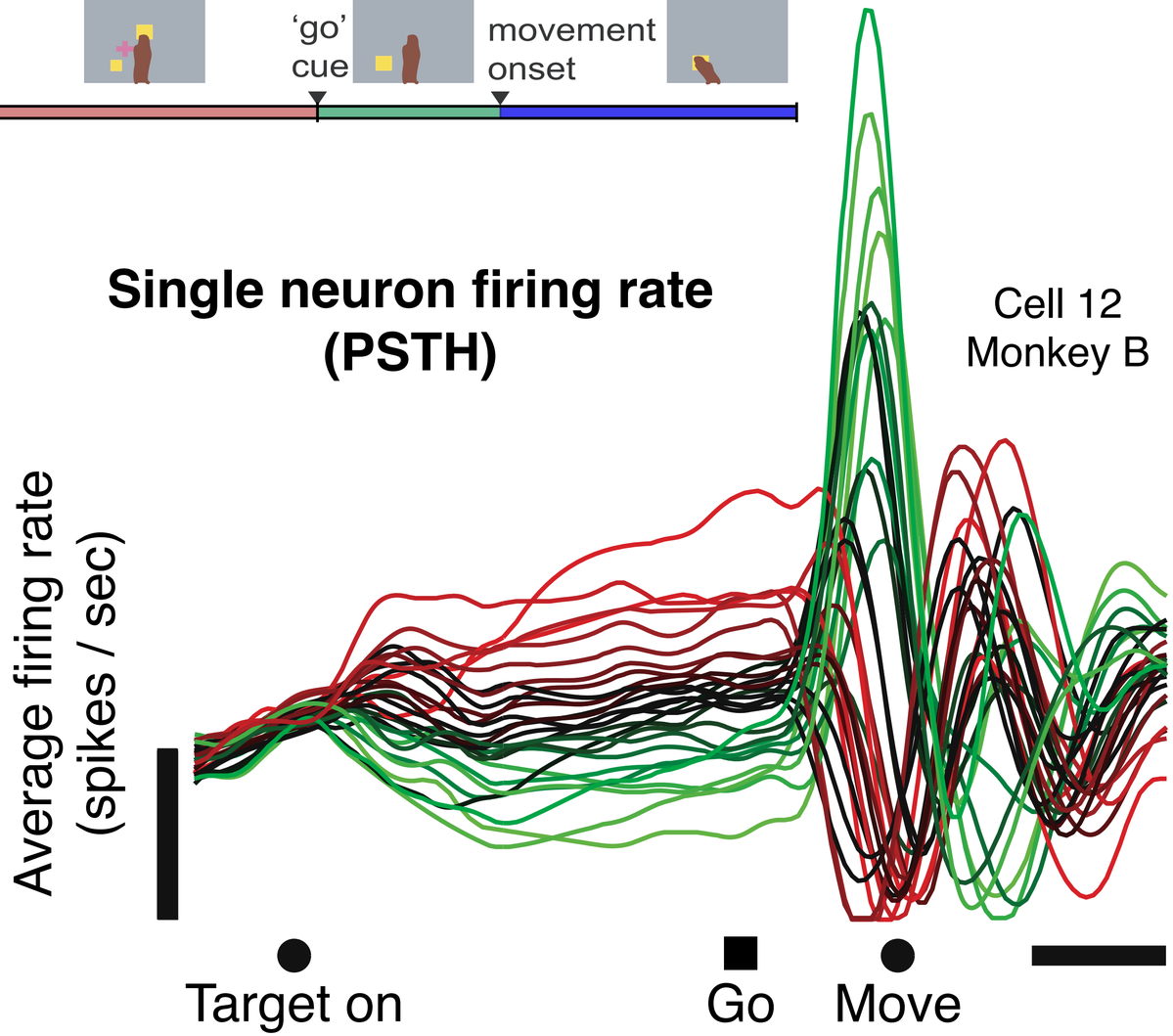




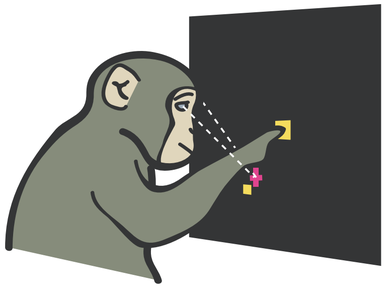
Reach trajectories

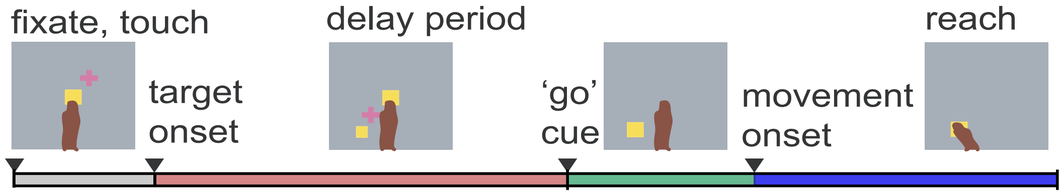


Single neuron firing rate (PSTH)

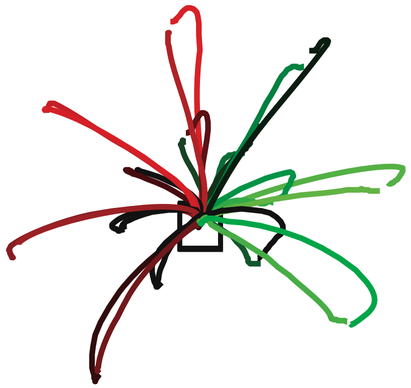


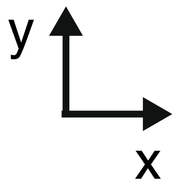
Churchland*, Cunningham* … Shenoy, *Nature* (2012)

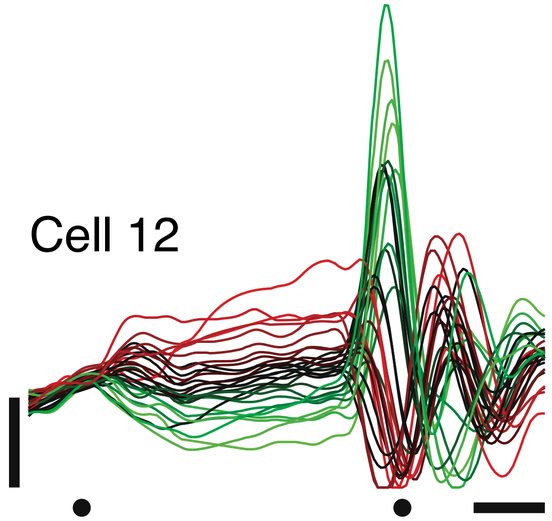


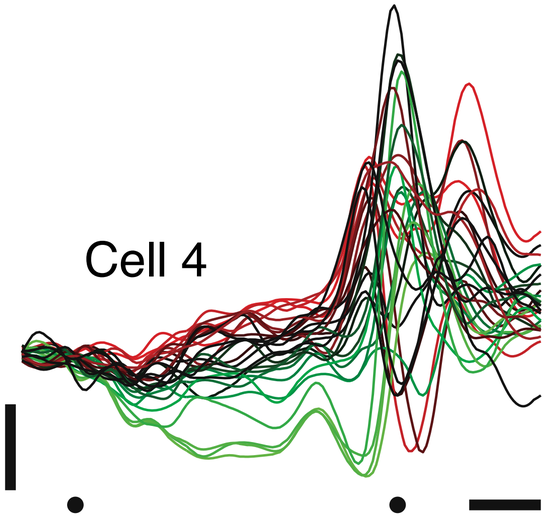


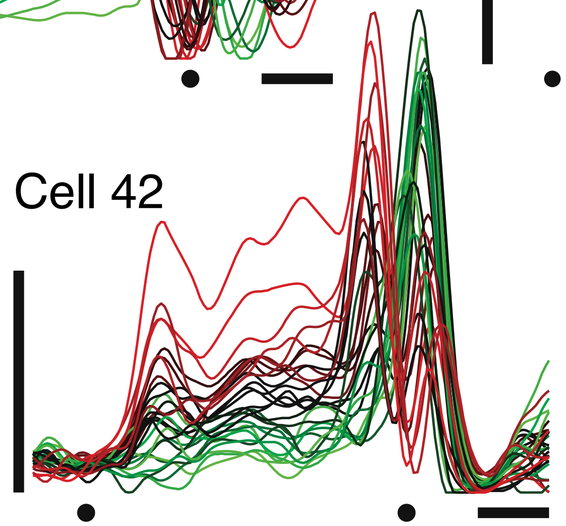
Reach trajectories





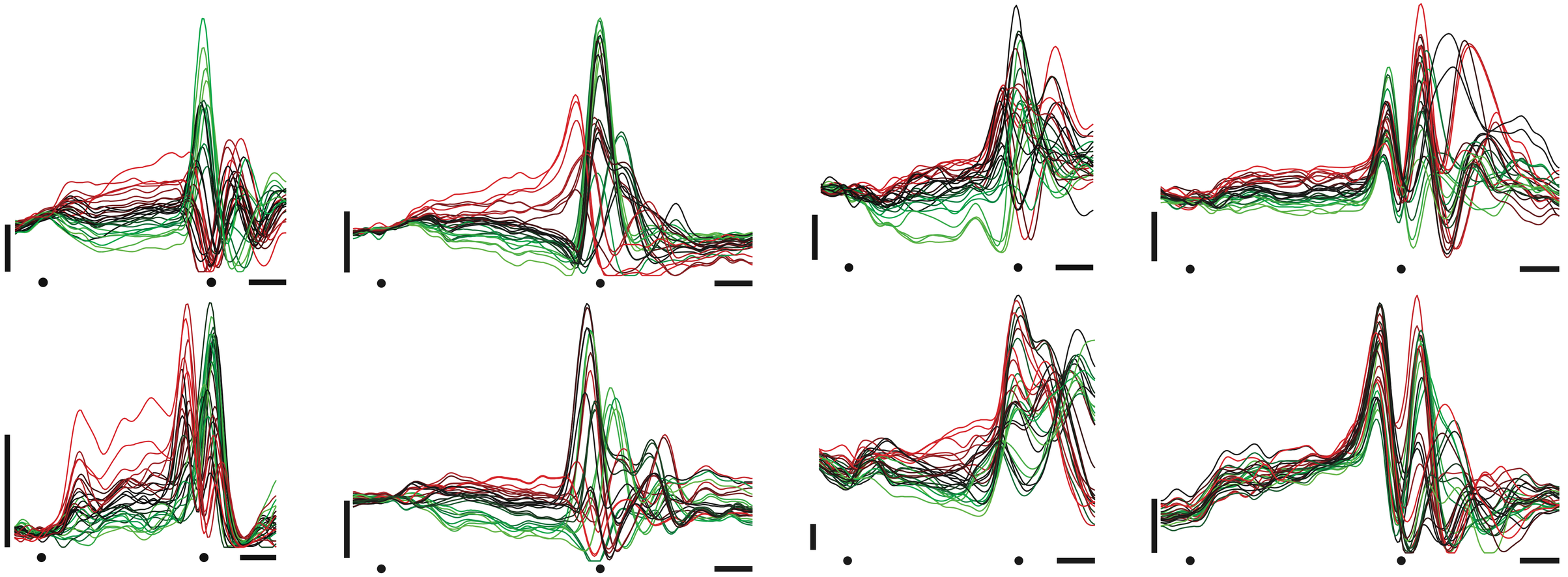






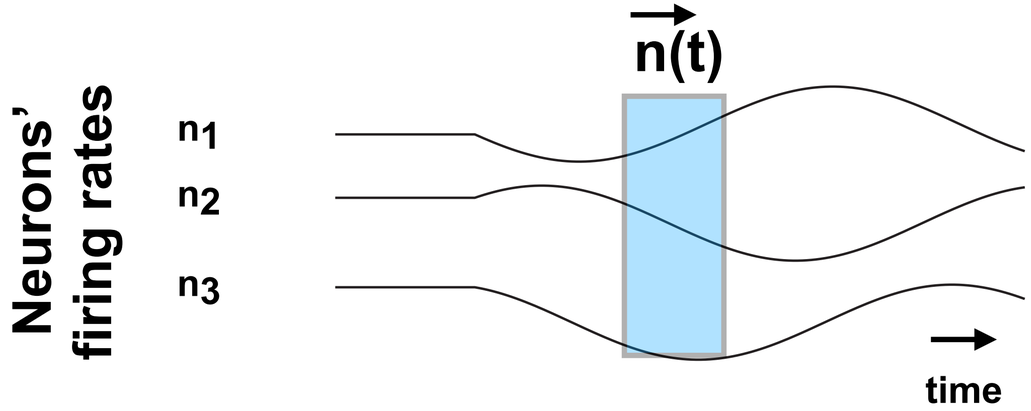
Churchland*, Cunningham* … Shenoy, *Nature* (2012)

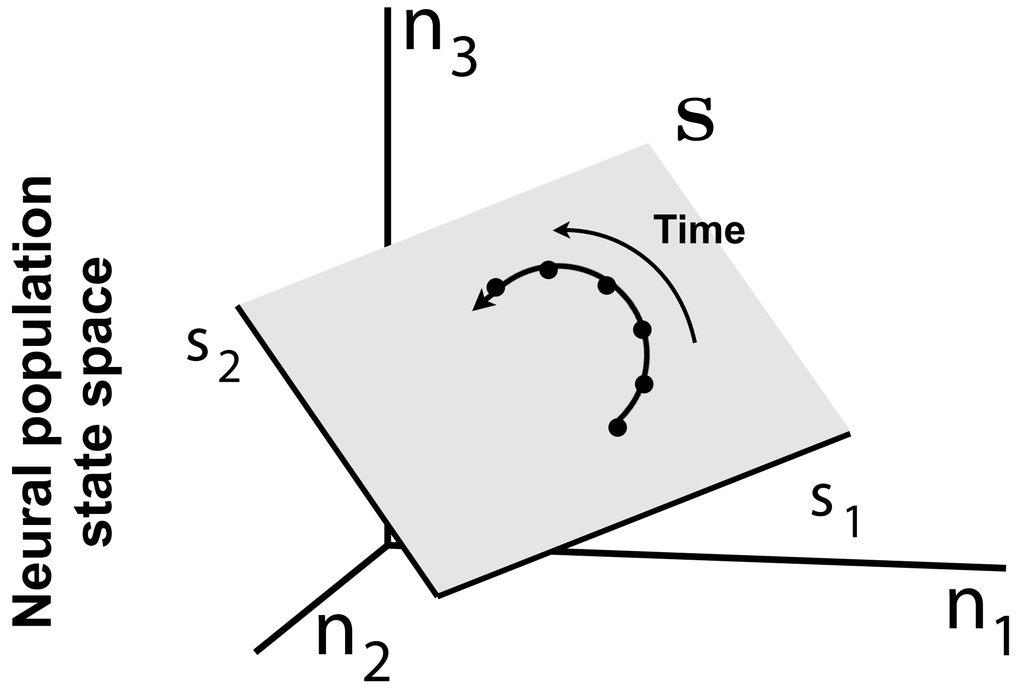
Single-neuron data is confusing



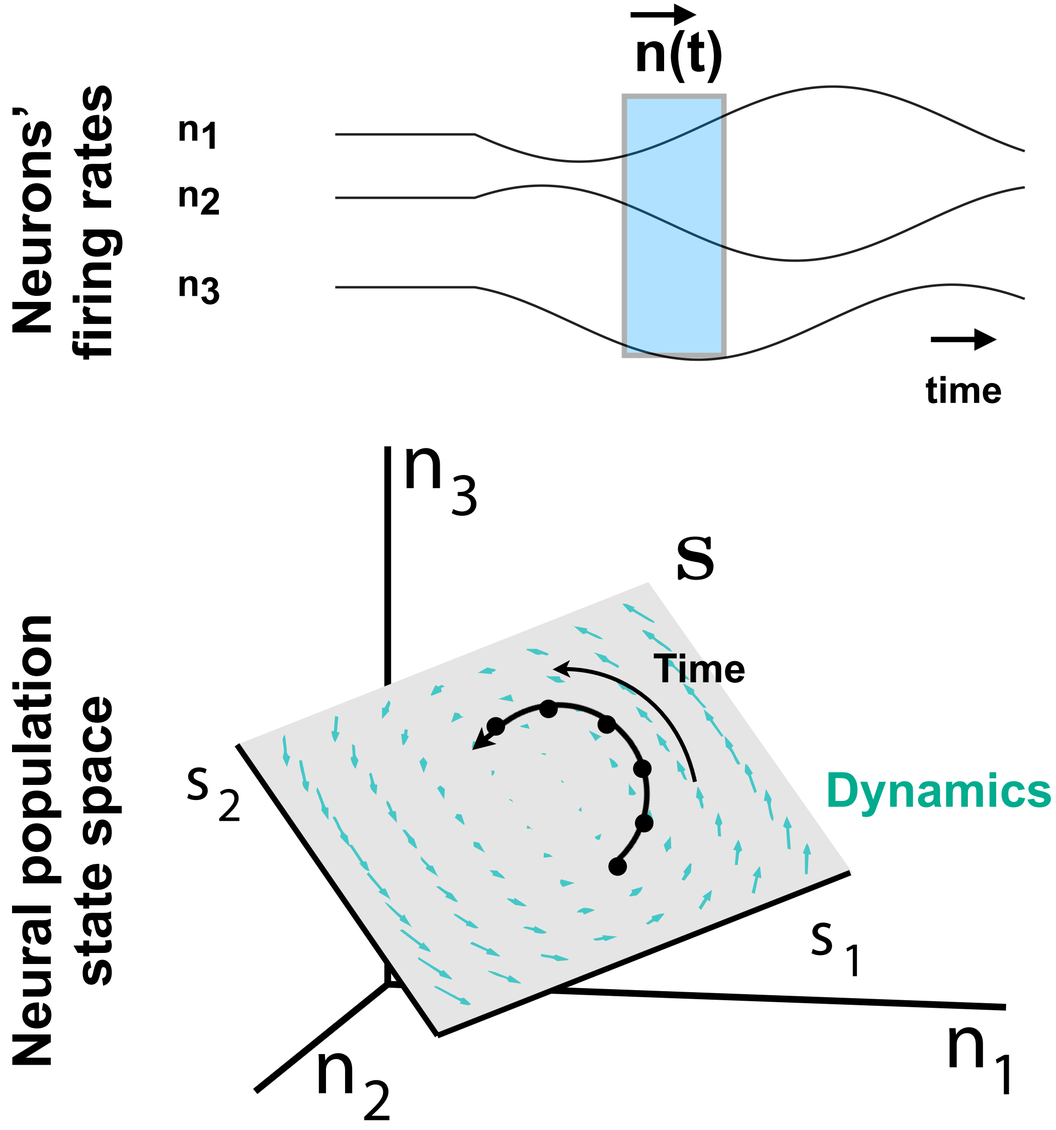
Churchland*, Cunningham* … Shenoy, *Nature* (2012)

Uncovering neural population dynamics

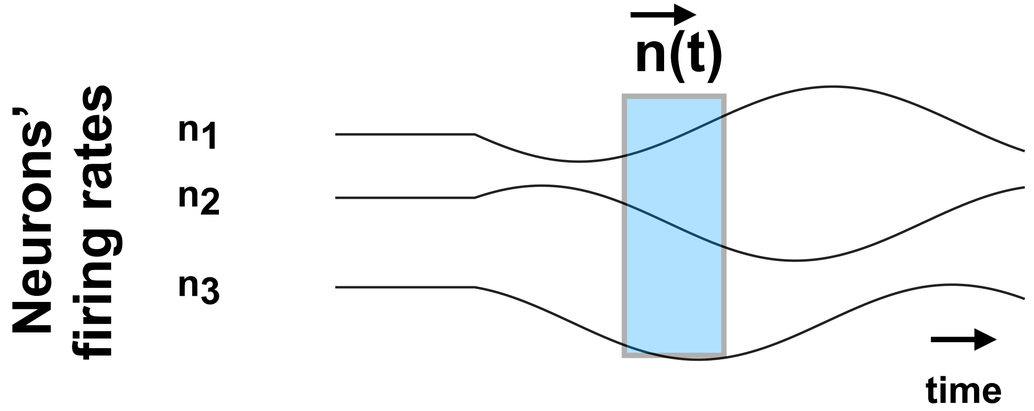




Uncovering neural population dynamics

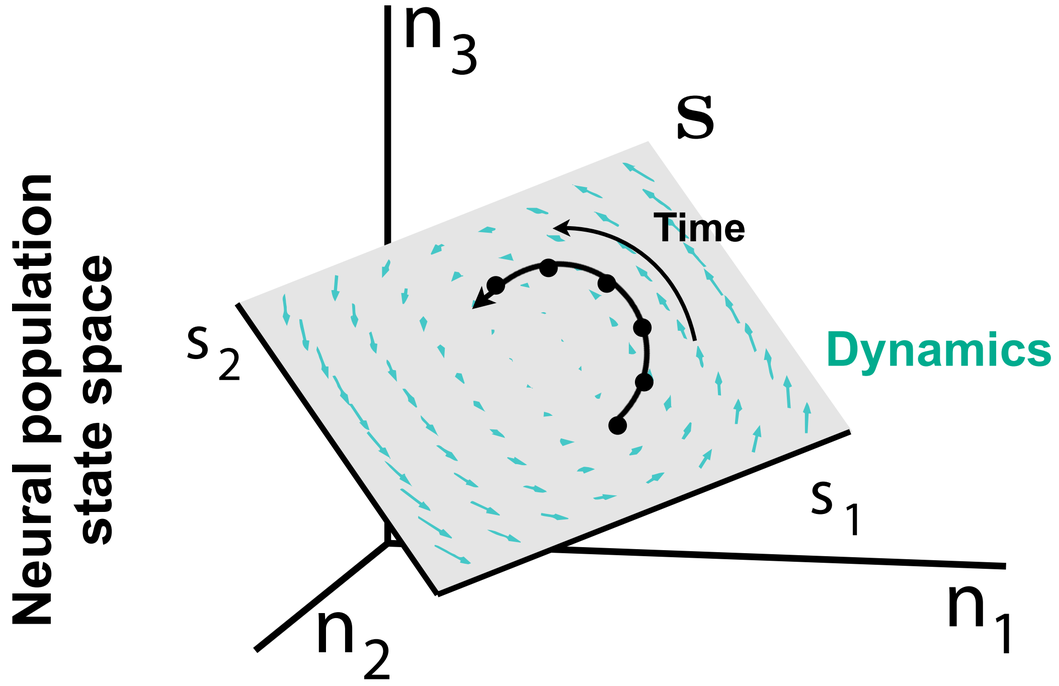


Uncovering neural population dynamics



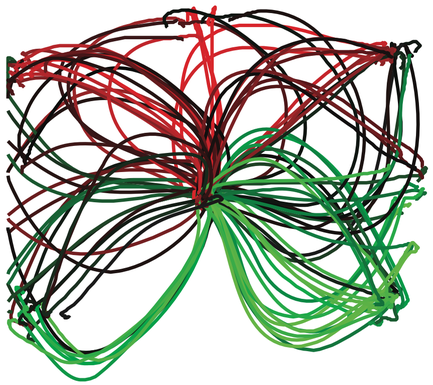
Predictable activity - modeled by *autonomous* dynamics

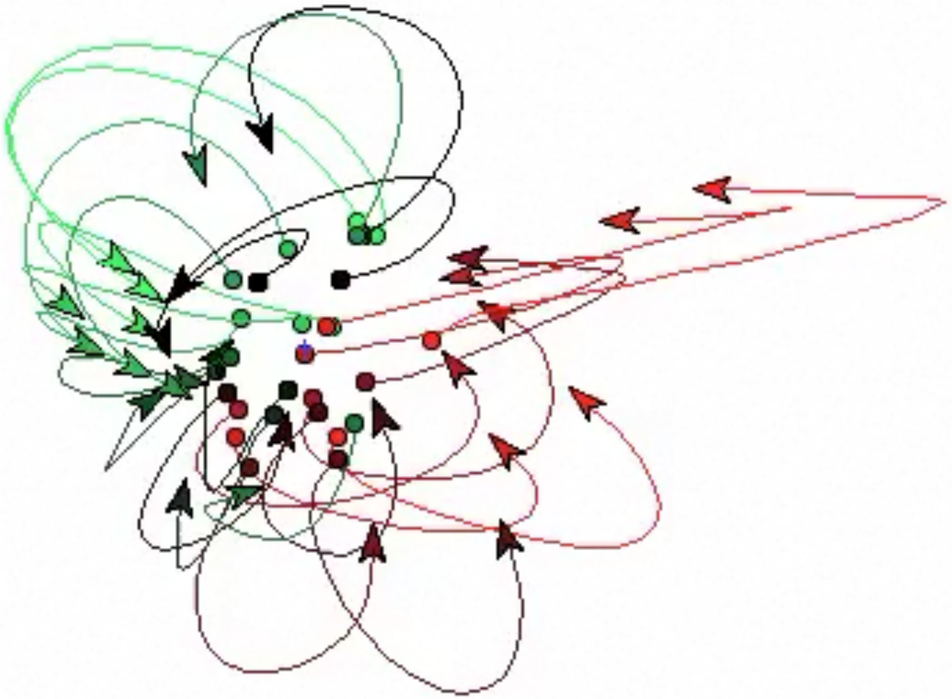
$$\frac{d\mathbf{s}}{dt} = f(\mathbf{s})$$



PCA / jPCA reveals underlying structure

Reach
trajectories

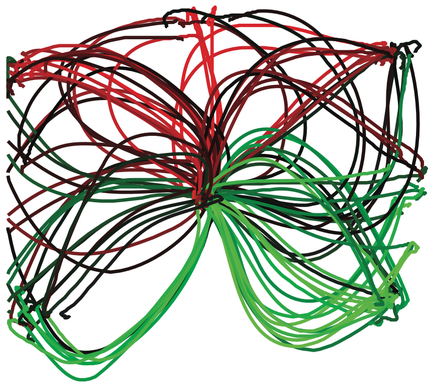


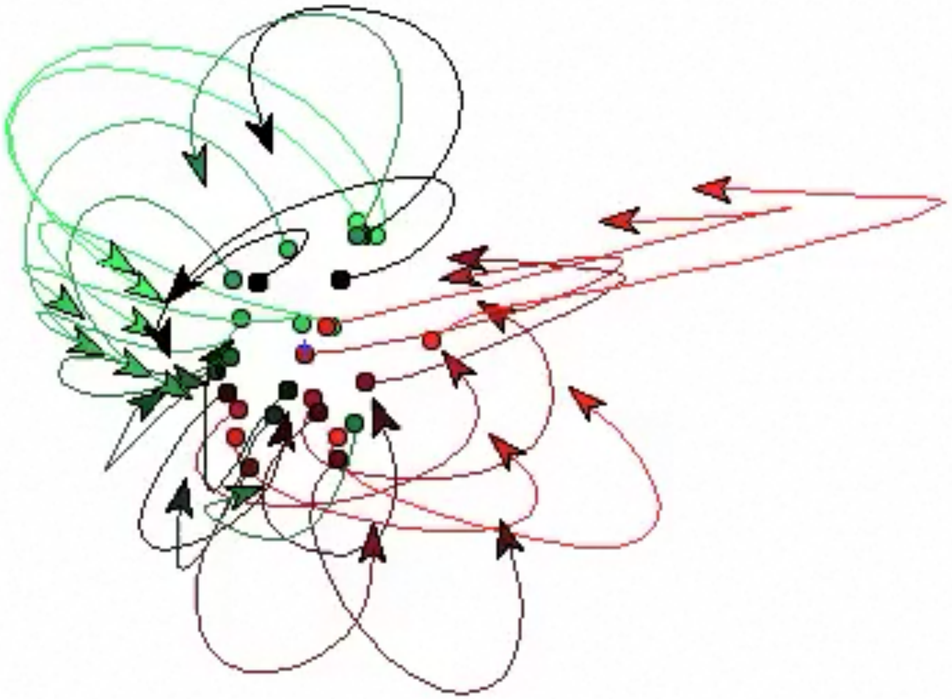


Churchland*, Cunningham* … Shenoy, *Nature* (2012)

PCA / jPCA reveals underlying structure

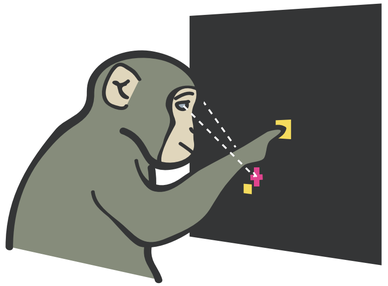
Reach
trajectories



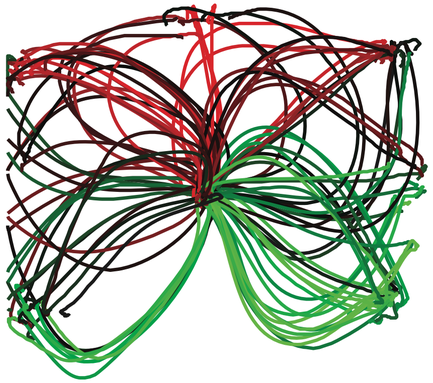


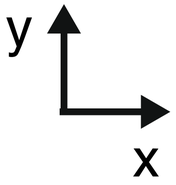
Churchland*, Cunningham* … Shenoy, *Nature* (2012)

Consistent rotational dynamics



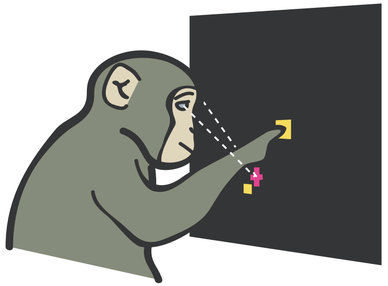
Reach
trajectories



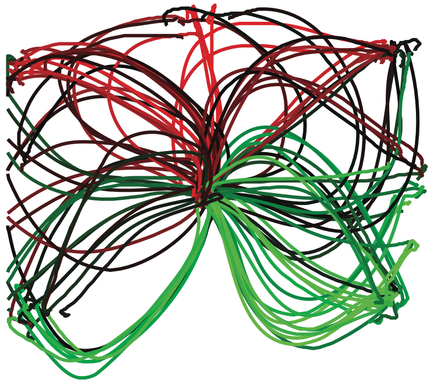


Churchland*, Cunningham* … Shenoy, *Nature* (2012)

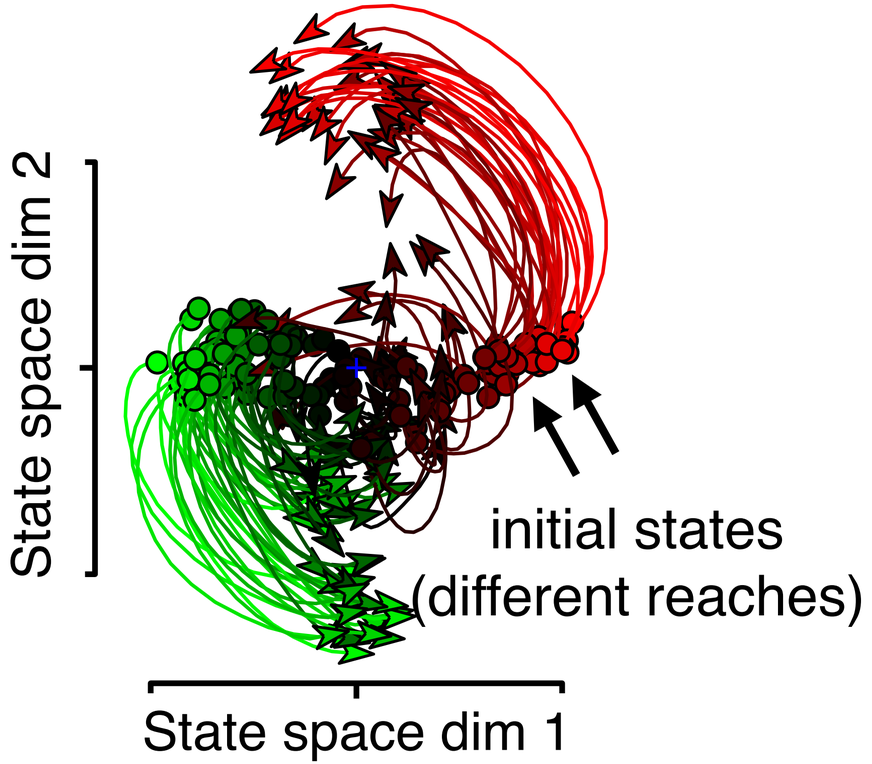
Consistent rotational dynamics



Reach
trajectories

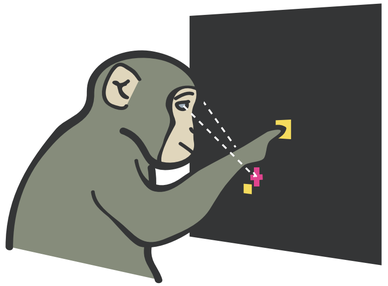


Plane 1

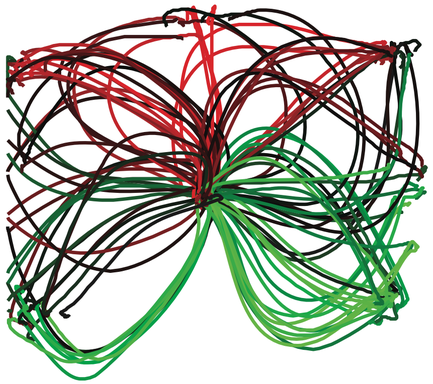


Churchland*, Cunningham* … Shenoy, *Nature* (2012)

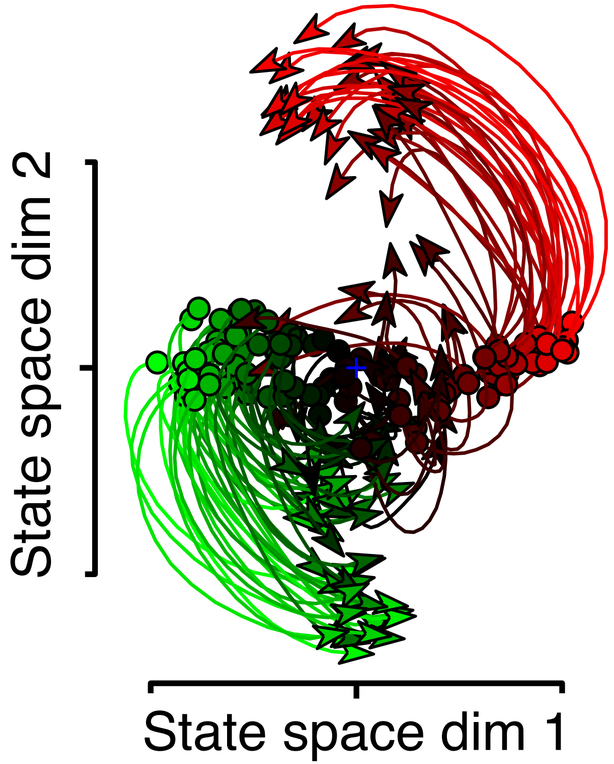
Consistent rotational dynamics



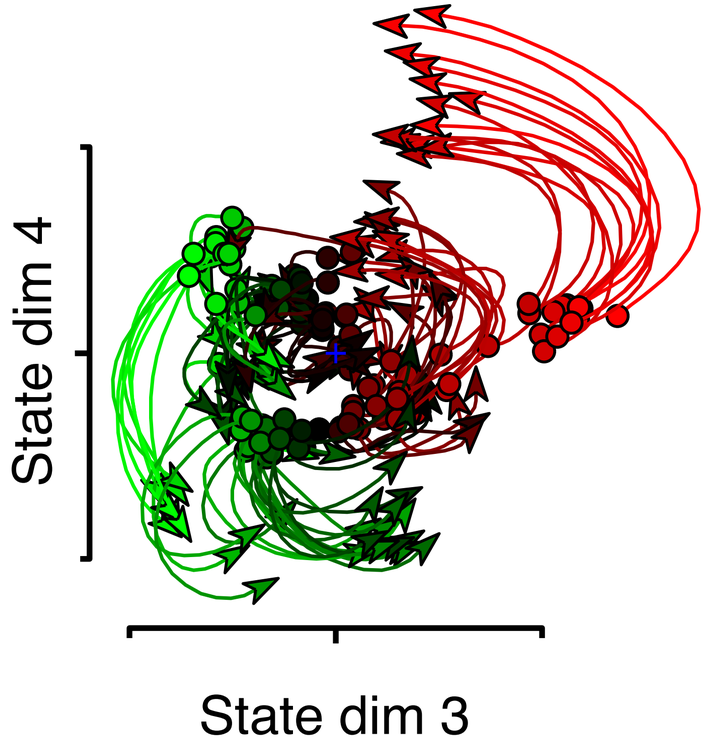
Reach trajectories



Plane 1

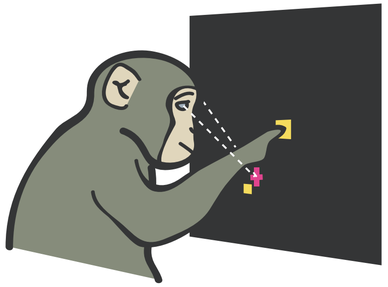


Plane 2

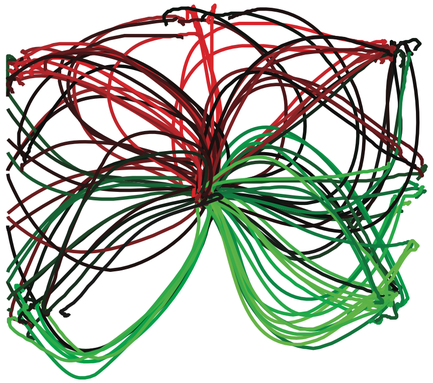


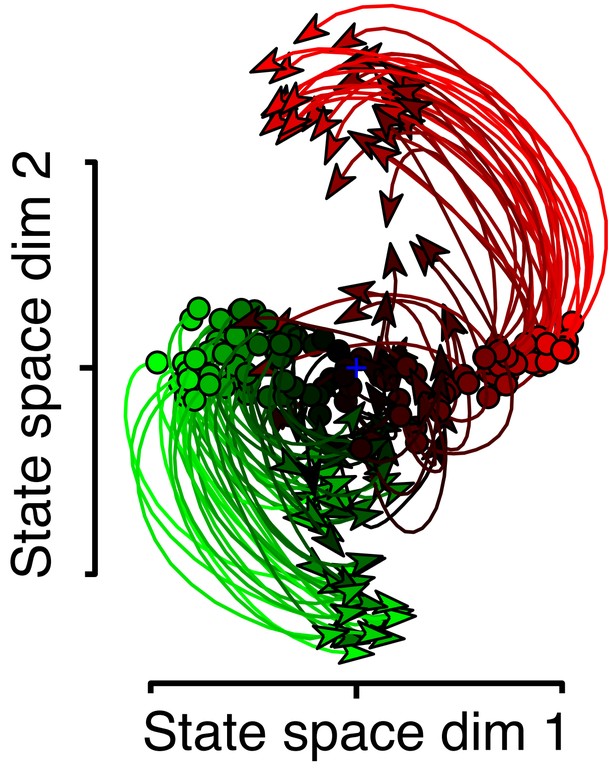
Churchland*, Cunningham* … Shenoy, *Nature* (2012)

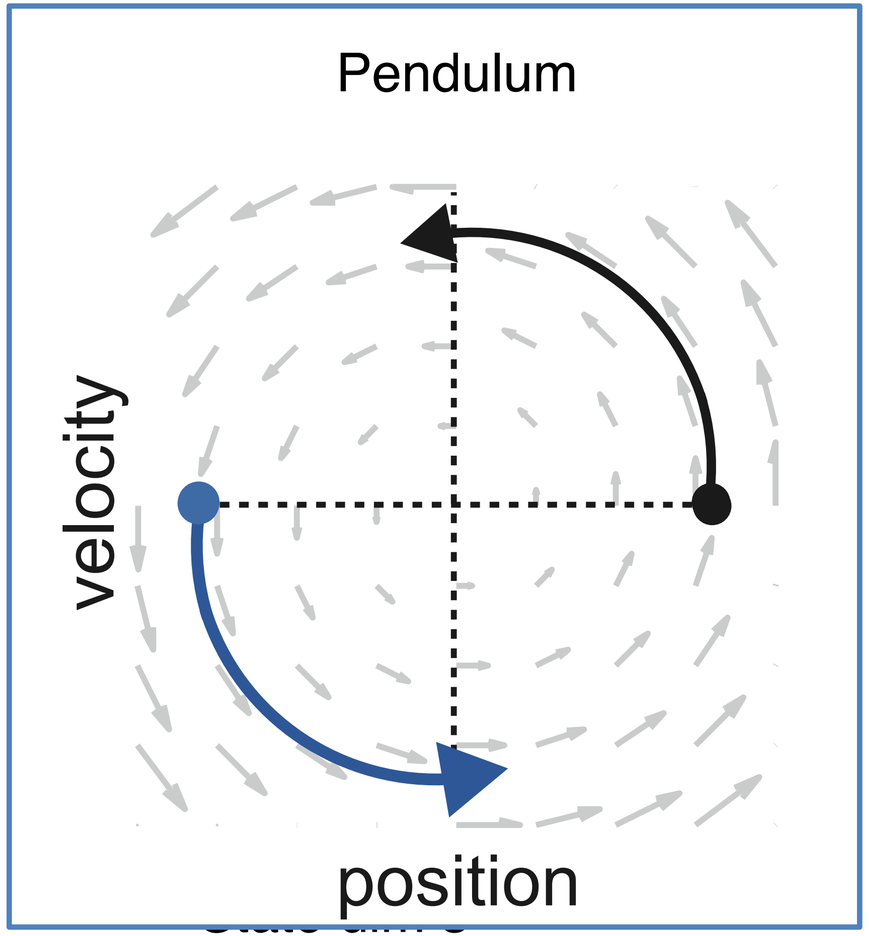
Consistent rotational dynamics



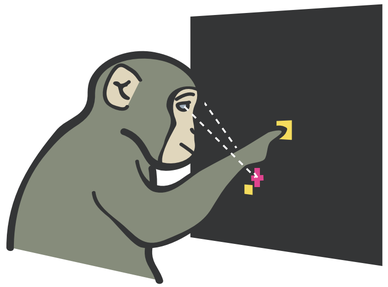
Reach trajectories



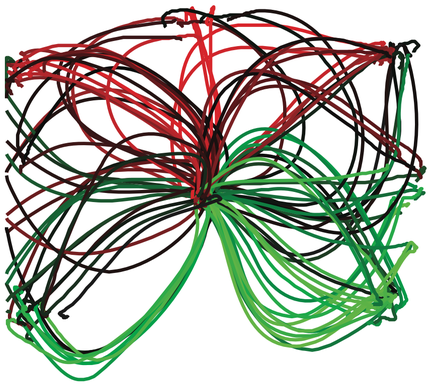


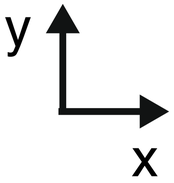


Churchland*, Cunningham* … Shenoy, *Nature* (2012)

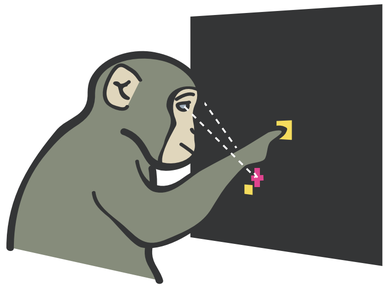


Reach
trajectories

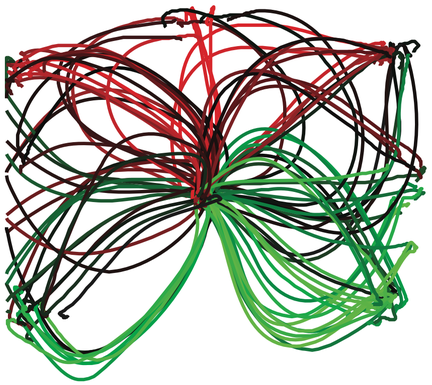


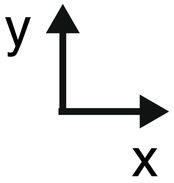


Churchland*, Cunningham* … Shenoy, *Nature* (2012)



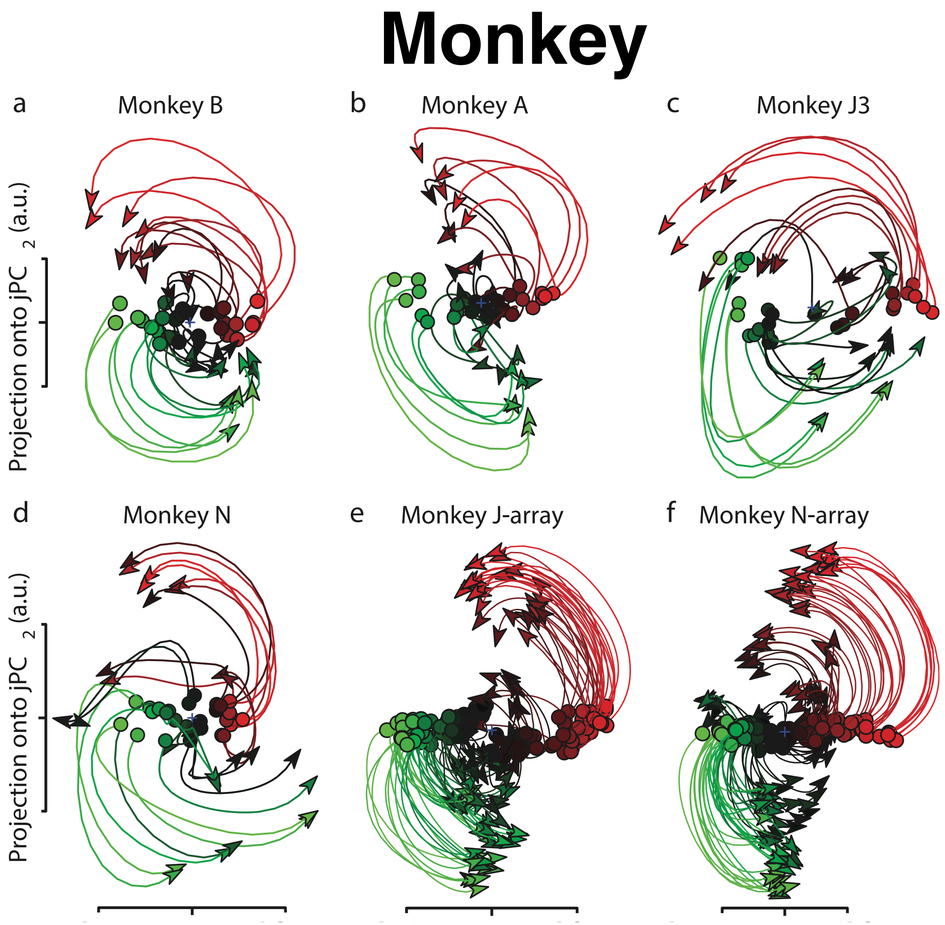
Reach
trajectories





Churchland*, Cunningham* … Shenoy, *Nature* (2012)

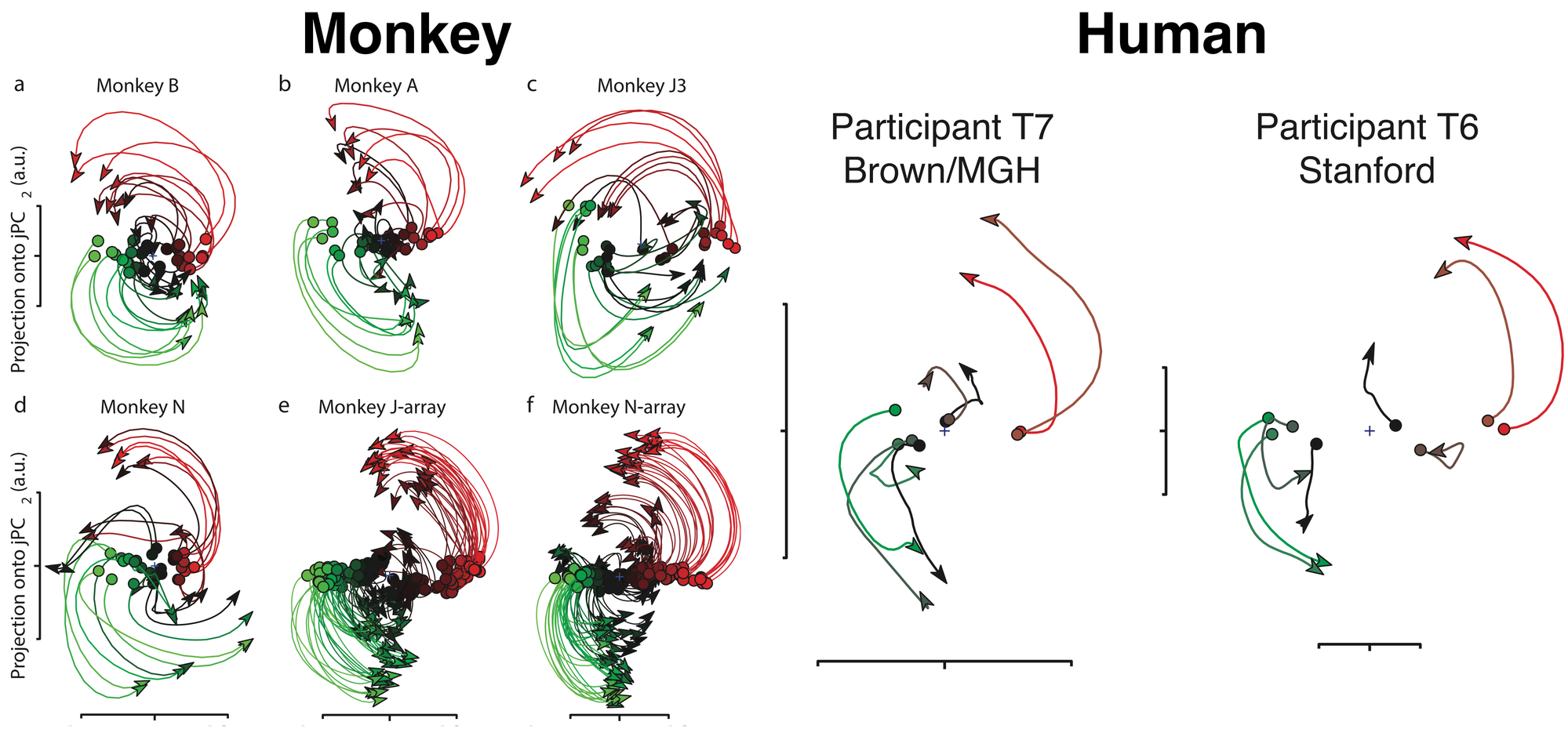
Rotational dynamics in monkeys & humans



Churchland*, Cunningham* … Shenoy, *Nature* (2012)

Pandarinath, … Hochberg, Henderson*, Shenoy*, *eLife* (2015)

Rotational dynamics in monkeys & humans

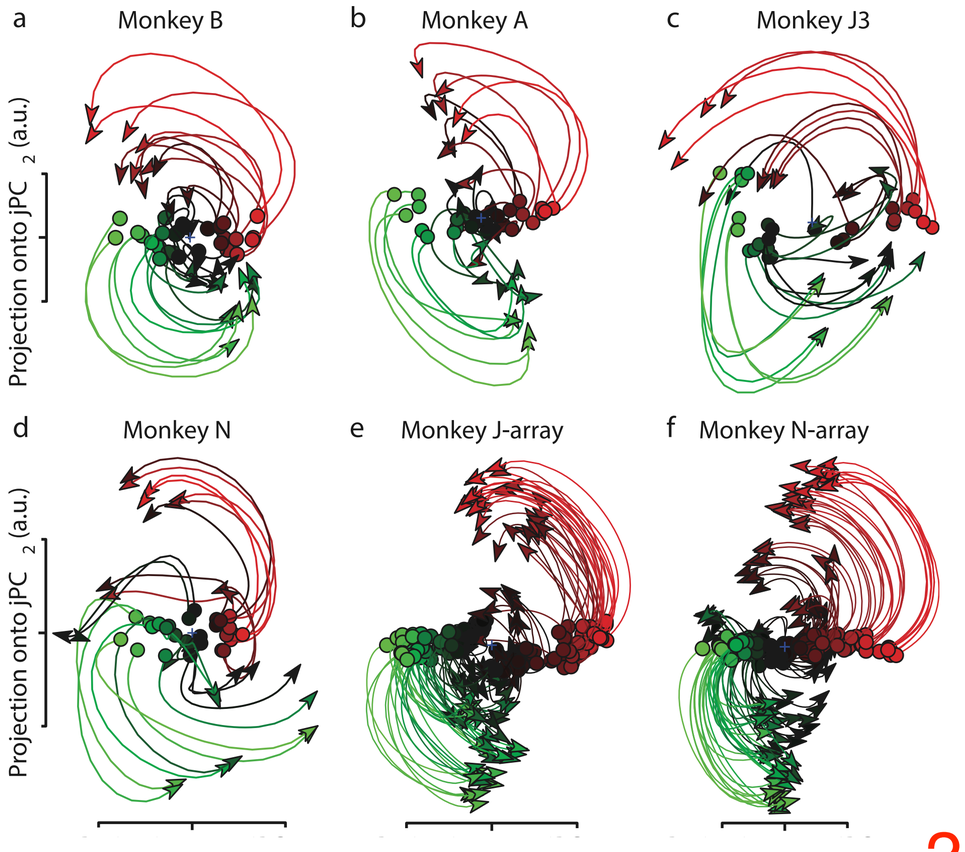


Churchland*, Cunningham* … Shenoy, *Nature* (2012)

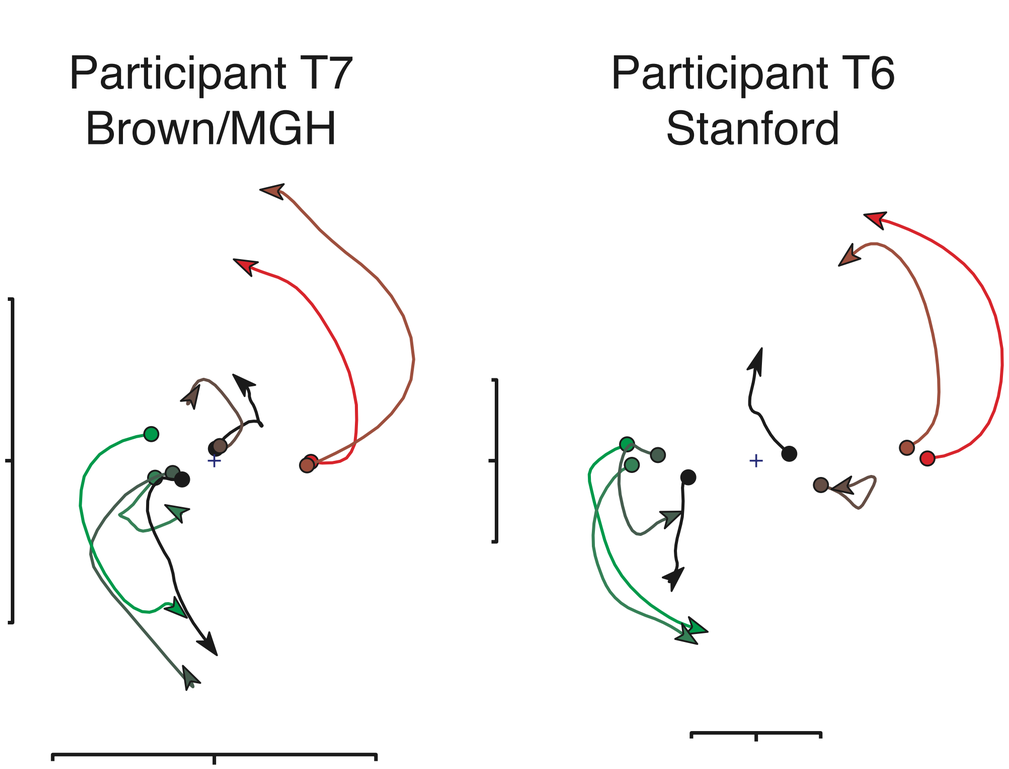
Pandarinath, … Hochberg, Henderson*, Shenoy*, *eLife* (2015)

Rotational dynamics in monkeys & humans

Monkey



Human



2-D state-space represents ***30-50%*** of data variance

Churchland*, Cunningham* … Shenoy, *Nature* (2012)

Pandarinath, … Hochberg, Henderson*, Shenoy*, *eLife* (2015)

What you'll hear about in this lecture

Neural network basics

Deep autoencoders

Intro to neural population dynamics