Lesson 2: PyTorch for the Impatient

2.1 What Is PyTorch?

2.2 The PyTorch Layer Cake

2.3 The Deep Learning Software Trilemma

2.4 What Are Tensors Really?

2.5 Tensors in PyTorch

2.6 Introduction to Computational Graphs

2.7 Backpropagation Is Just the Chain Rule

2.8 Effortless Backpropagation with torch.autograd

2.9 PyTorch's Device Abstraction (i.e., GPUs)

Lesson 2: PyTorch for the Impatient

2.10 Working with Devices

2.11 Components of a Learning Algorithm

2.12 Introduction to Gradient Descent

2.13 Getting to Stochastic Gradient Descent (SGD)

2.14 Comparing Gradient Descent and SGD

2.15 Linear Regression with PyTorch

2.16 Perceptrons and Neurons

2.17 Layers and Activations with torch.nn

2.18 Multi-layer Feedforward Neural Networks (MLP)

2.1

What is PyTorch?

What is PyTorch?

Deep Learning Library

What is PyTorch?

Neural network pseudocode

What is PyTorch?

`numpy` like CUDA interface + **Automatic Differentiation** = **PyTorch**

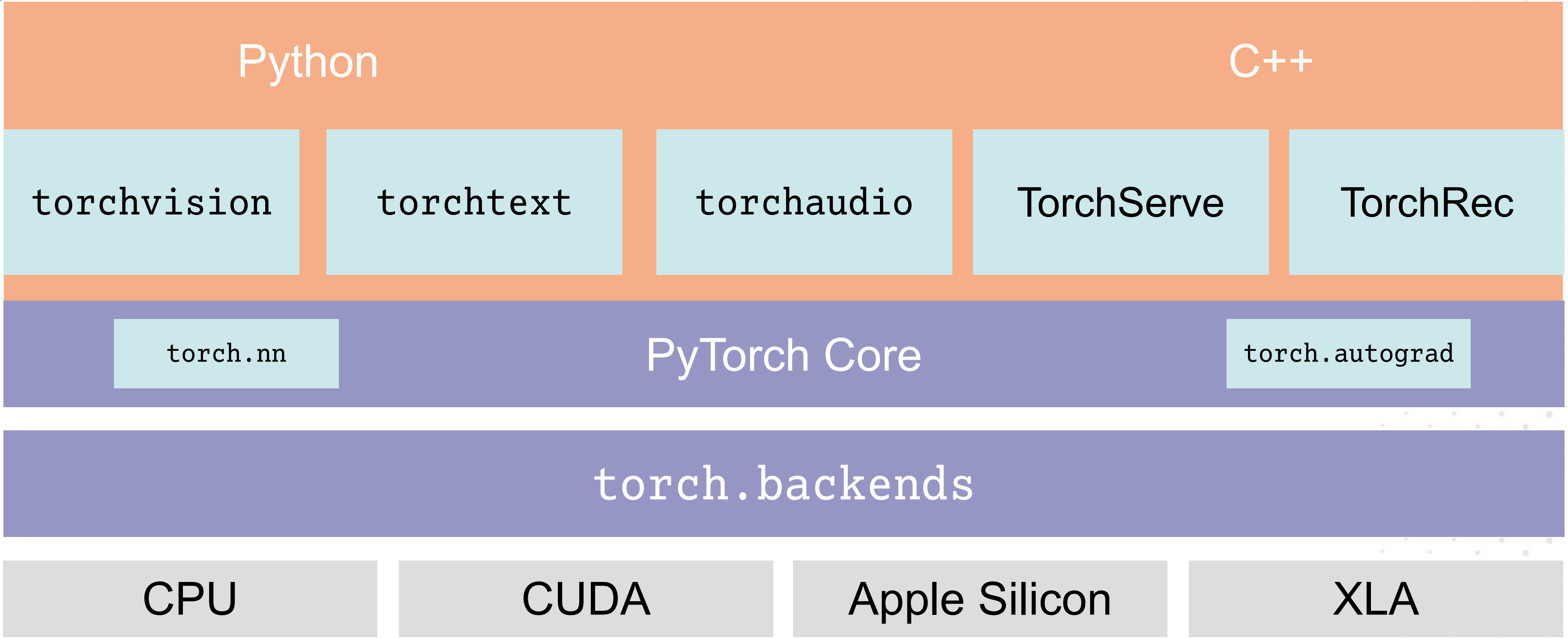
What is PyTorch?

Really only necessary for accelerating deep learning algorithms on GPUs (i.e. not scikit-learn or data science)

2.2

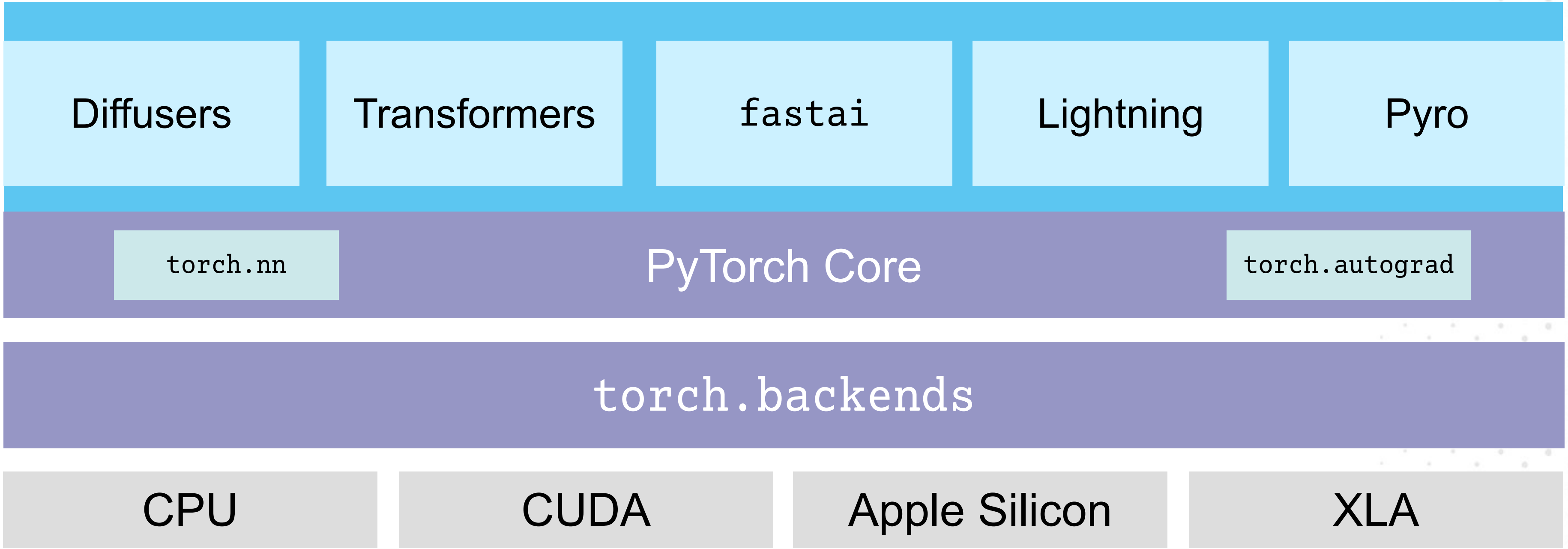
The PyTorch Layer Cake

PyTorch Layer Cake



Pearson

PyTorch Ecosystem

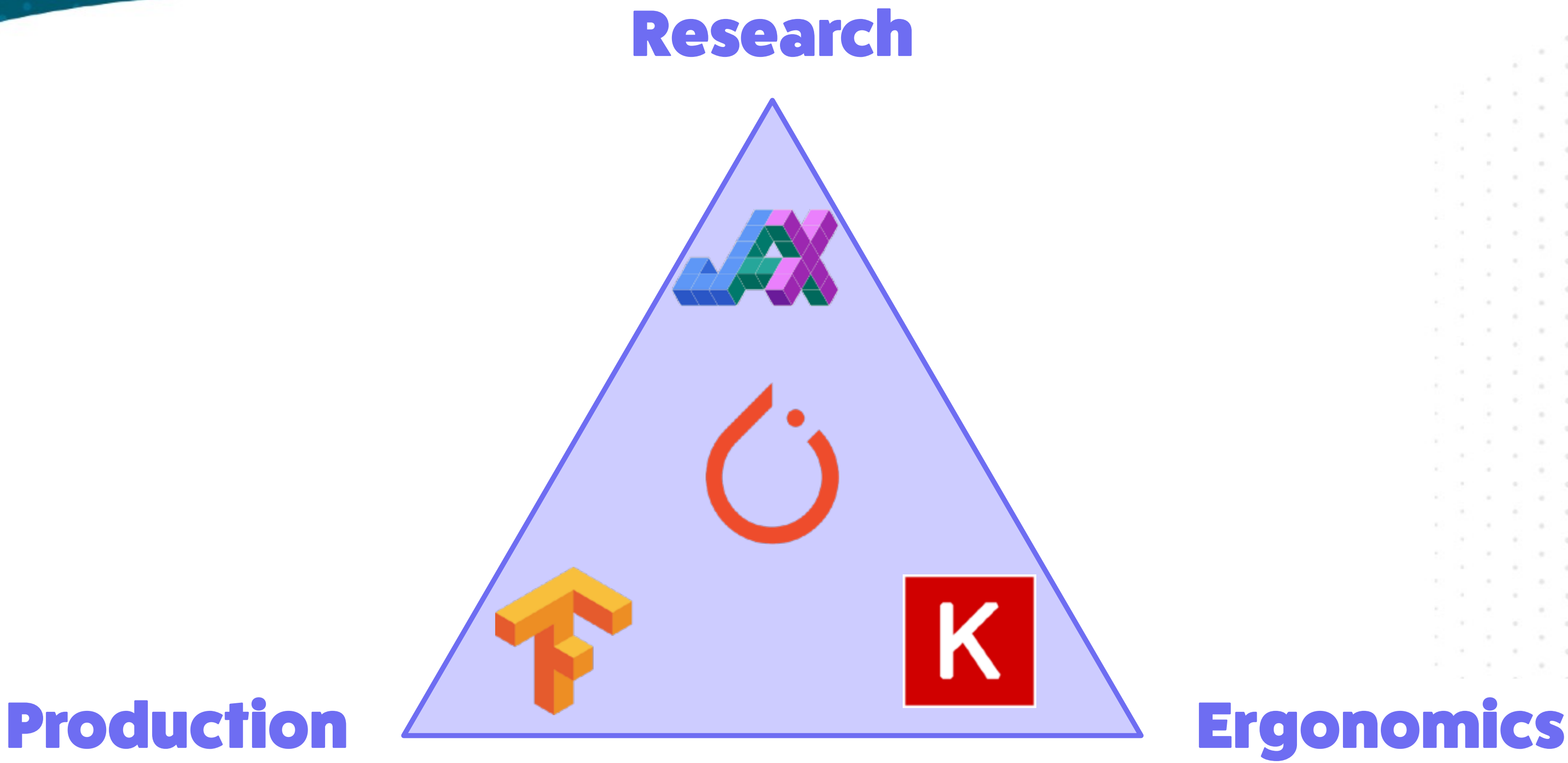


2.3

The Deep Learning Software Trilemma

Pearson

Deep Learning Software Trilemma



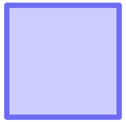
2.4

What are Tensors Really?

What is a Tensor?

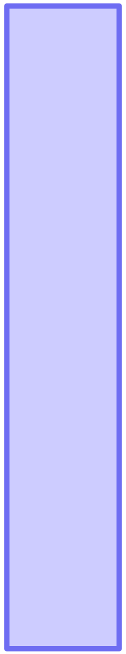
Generalization of arrays

What is a Tensor?



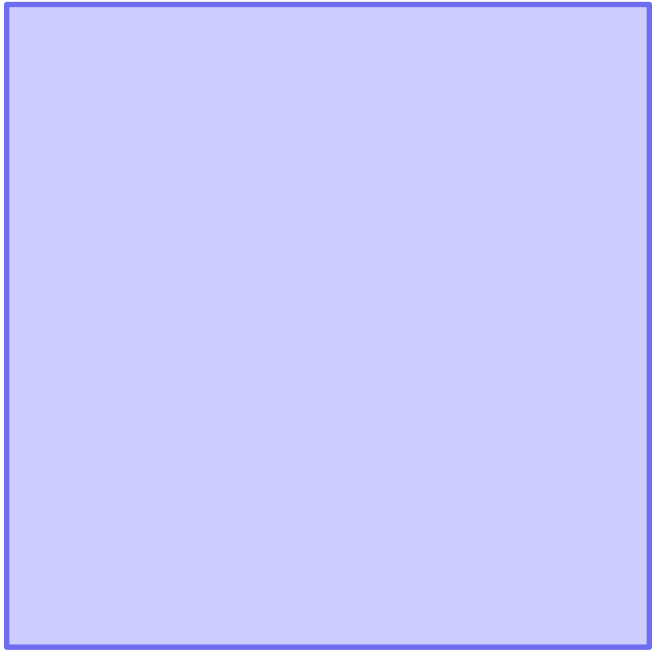
Rank 0 (scalar)

What is a Tensor?



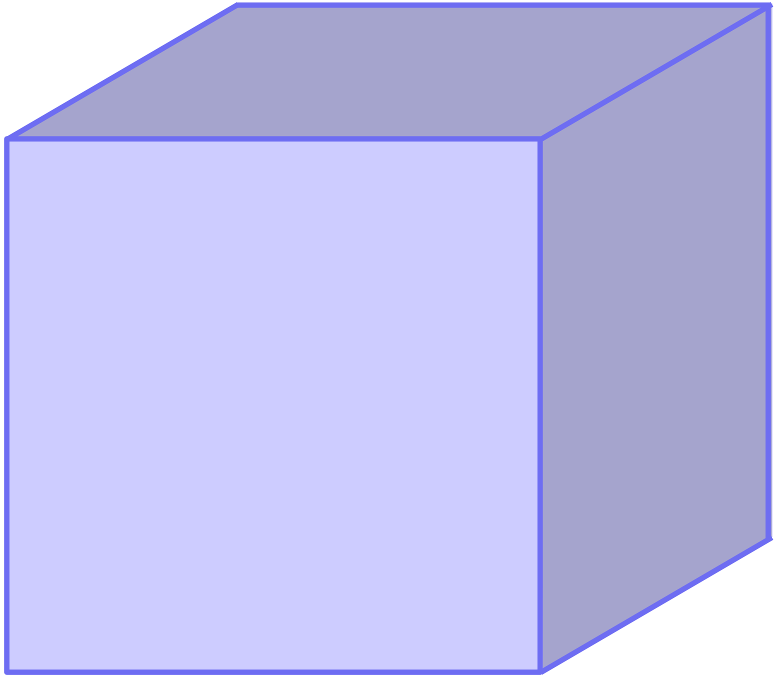
Rank 1 (vector)

What is a Tensor?



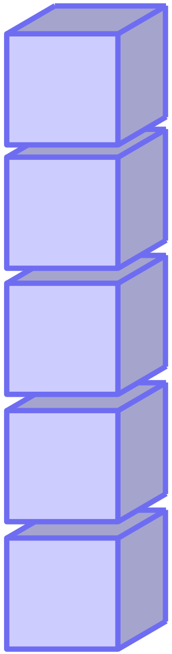
Rank 2 (matrix)

What is a Tensor?



Rank 3 (tensor)

What is a Tensor?



Rank 4 (tensor)

Pearson

2.5

Tensors in PyTorch

Live Coding

2.6

Introduction to Computational Graphs

Computational Graphs

Abstraction for representing mathematical operations

Computational Graphs

Uniquely suited for automatic differentiation (and by extension programming neural networks)

Computational Graphs

$$f(x, y) = 3x^2 + 2y$$

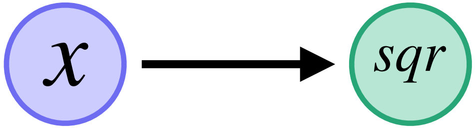
Pearson

Computational Graphs

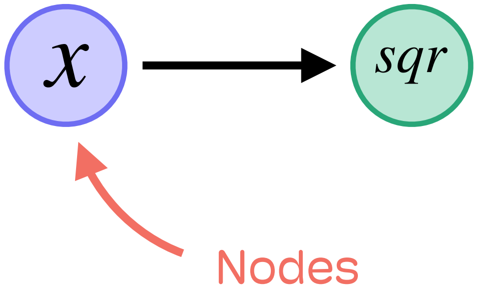
$$f(x, y) = (3 \times (x^2)) + (2 \times y)$$

Pearson

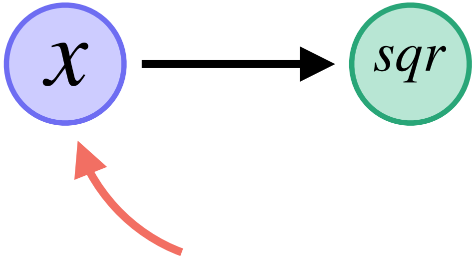
Computational Graphs



Computational Graphs

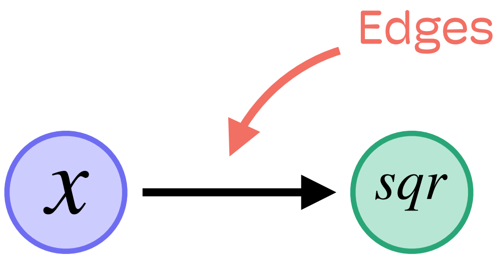


Computational Graphs



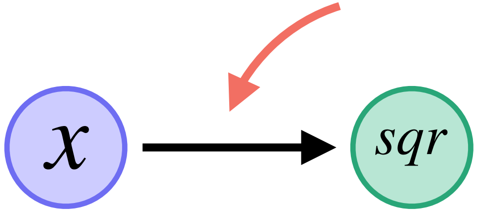
Represent
{variables, values, and operations}

Computational Graphs

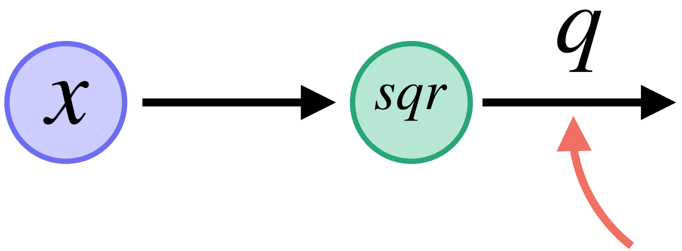


Computational Graphs

Represent
{dependencies and data flow }



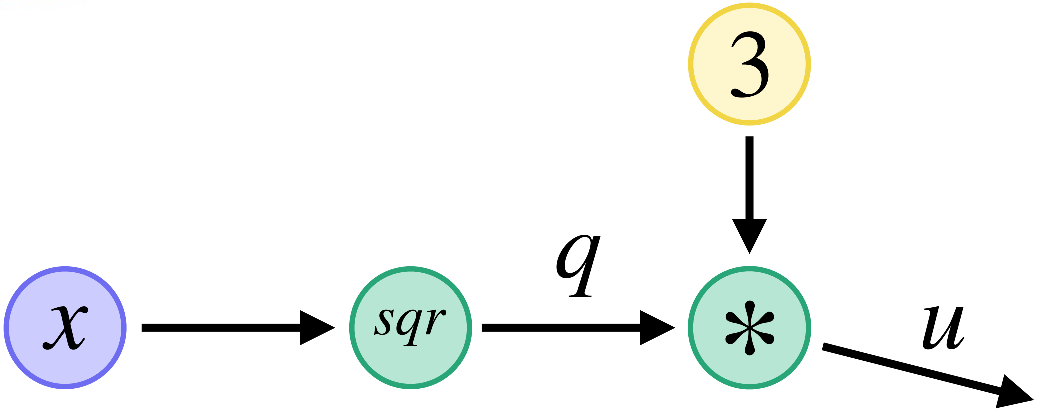
Computational Graphs



i.e. outputs and intermediate computations

Pearson

Computational Graphs



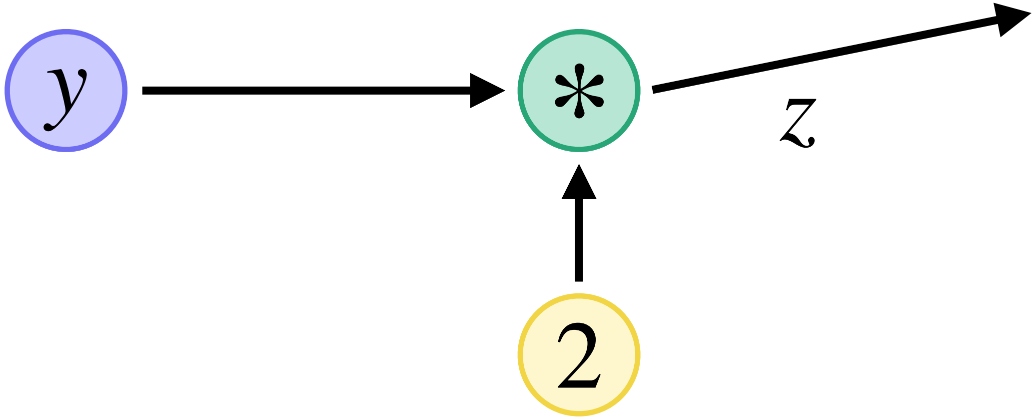
Pearson

Computational Graphs

$$f(x, y) = (3 \times (x^2)) + (2 \times y)$$

Pearson

Computational Graphs

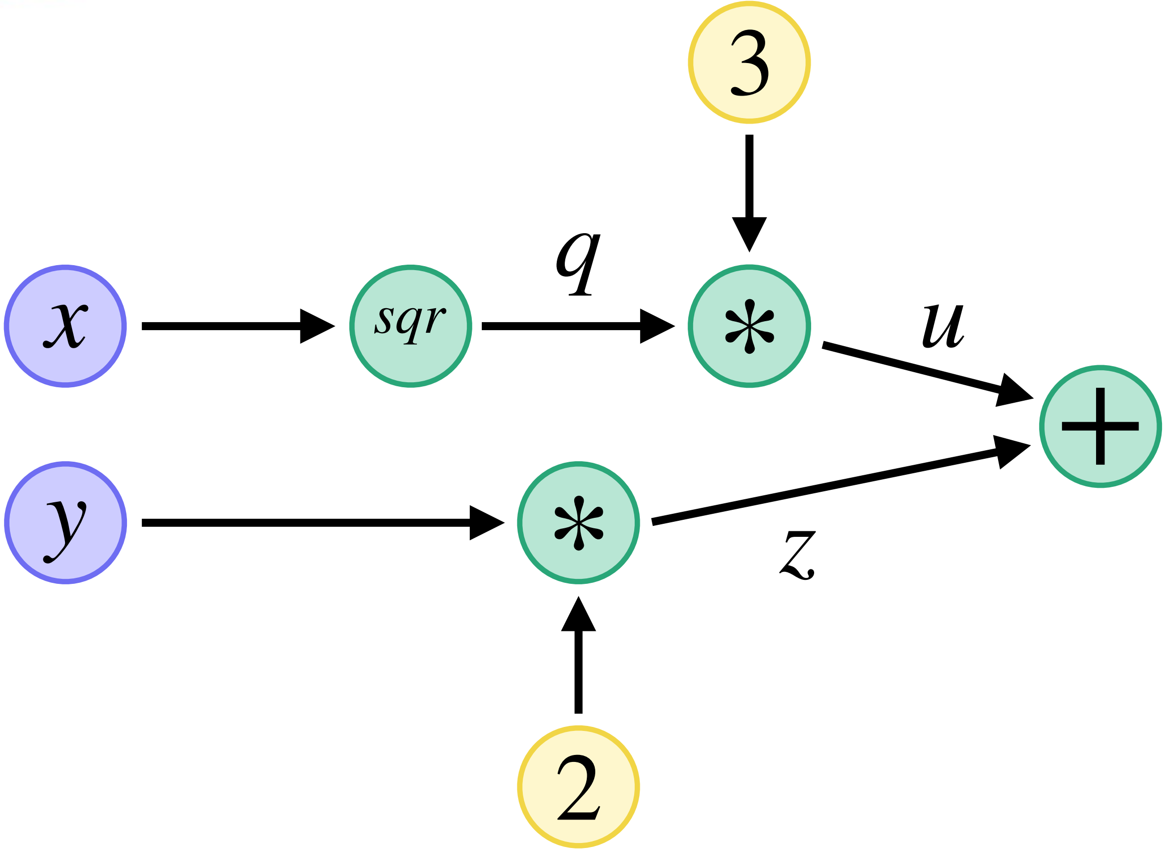


Computational Graphs

$$f(x, y) = (3 \times (x^2)) + (2 \times y)$$

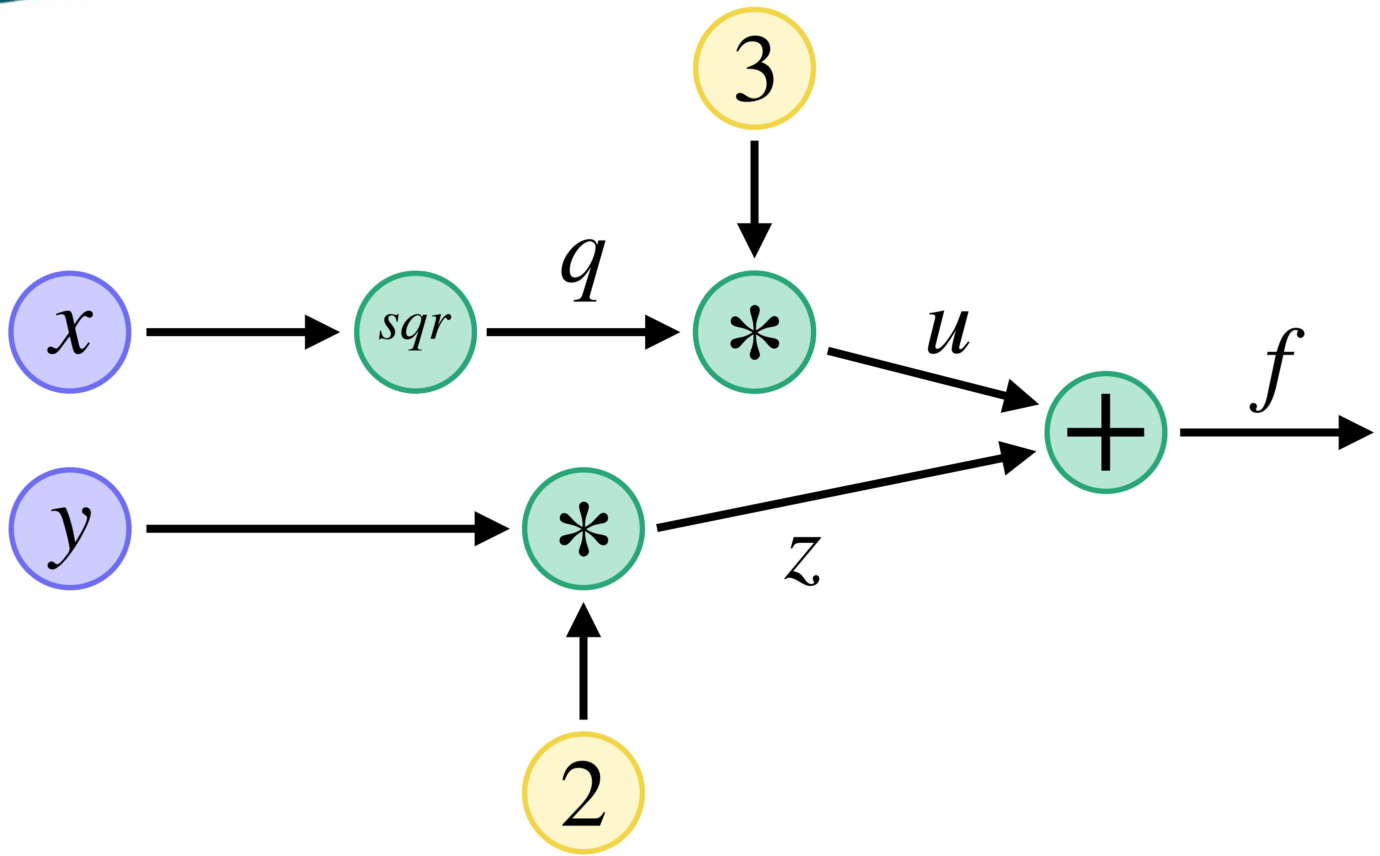
Pearson

Computational Graphs



Pearson

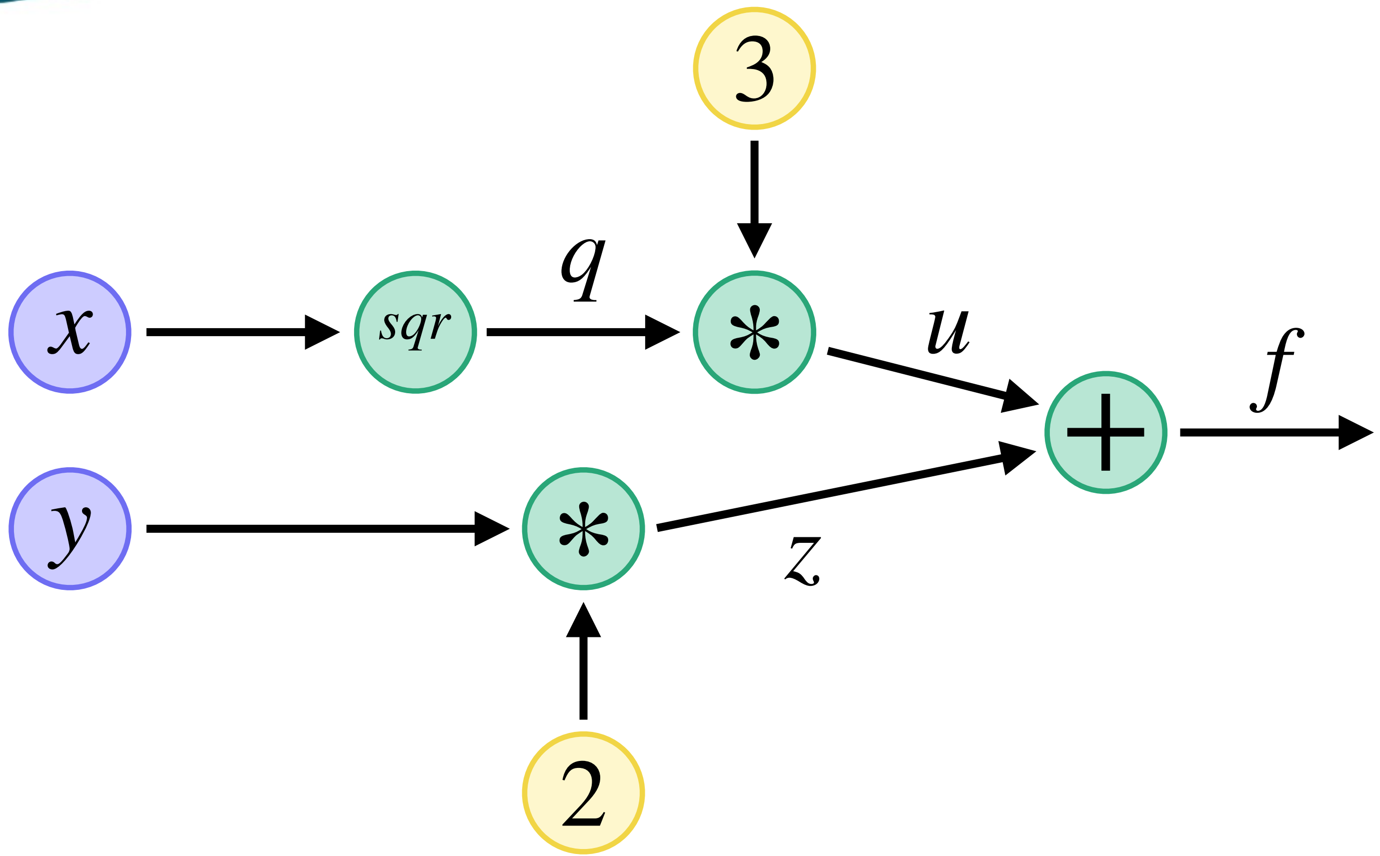
Computational Graphs



2.7

Backpropagation is just the chain rule

Computational Graphs



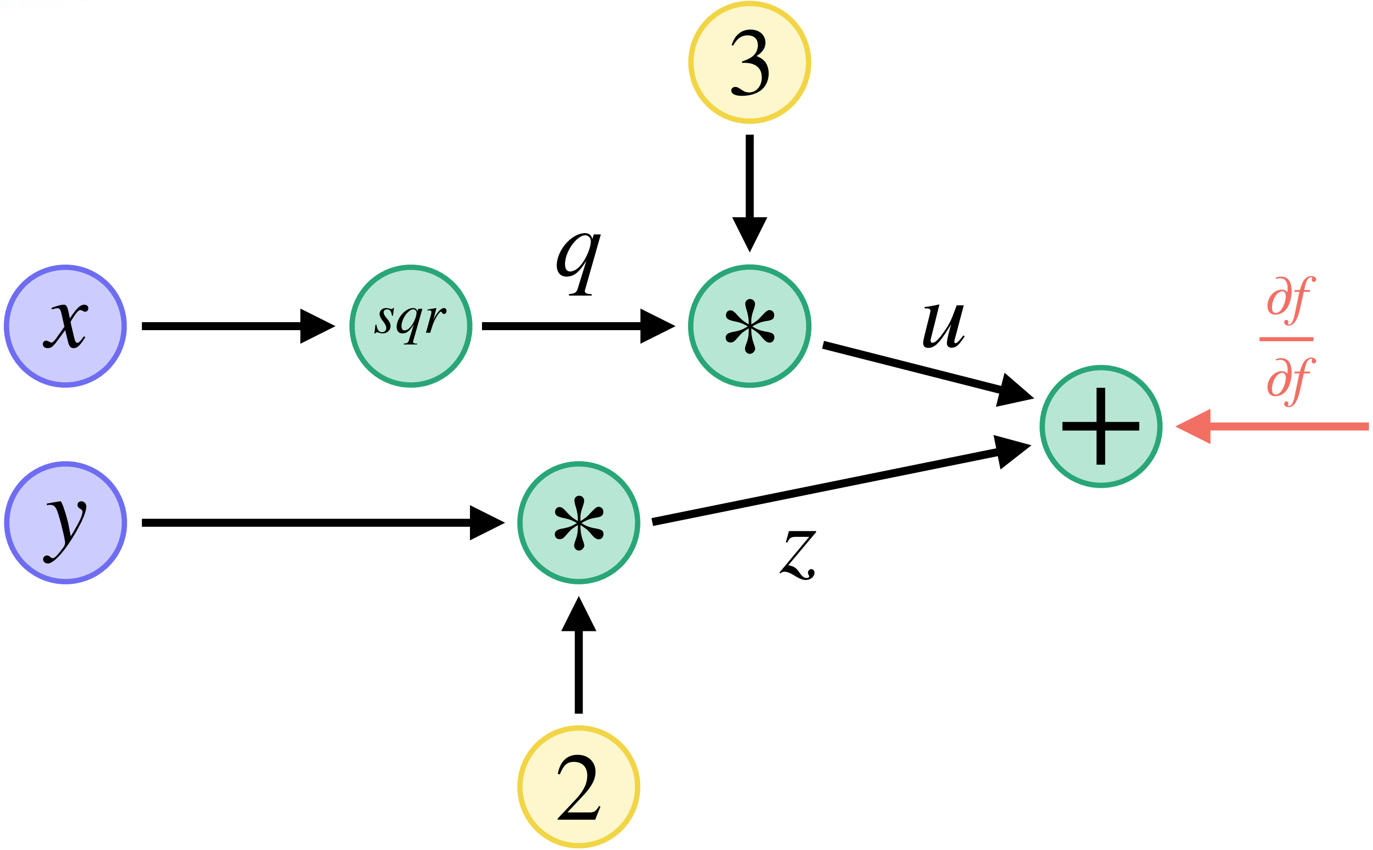
Computational Graphs

$$\left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

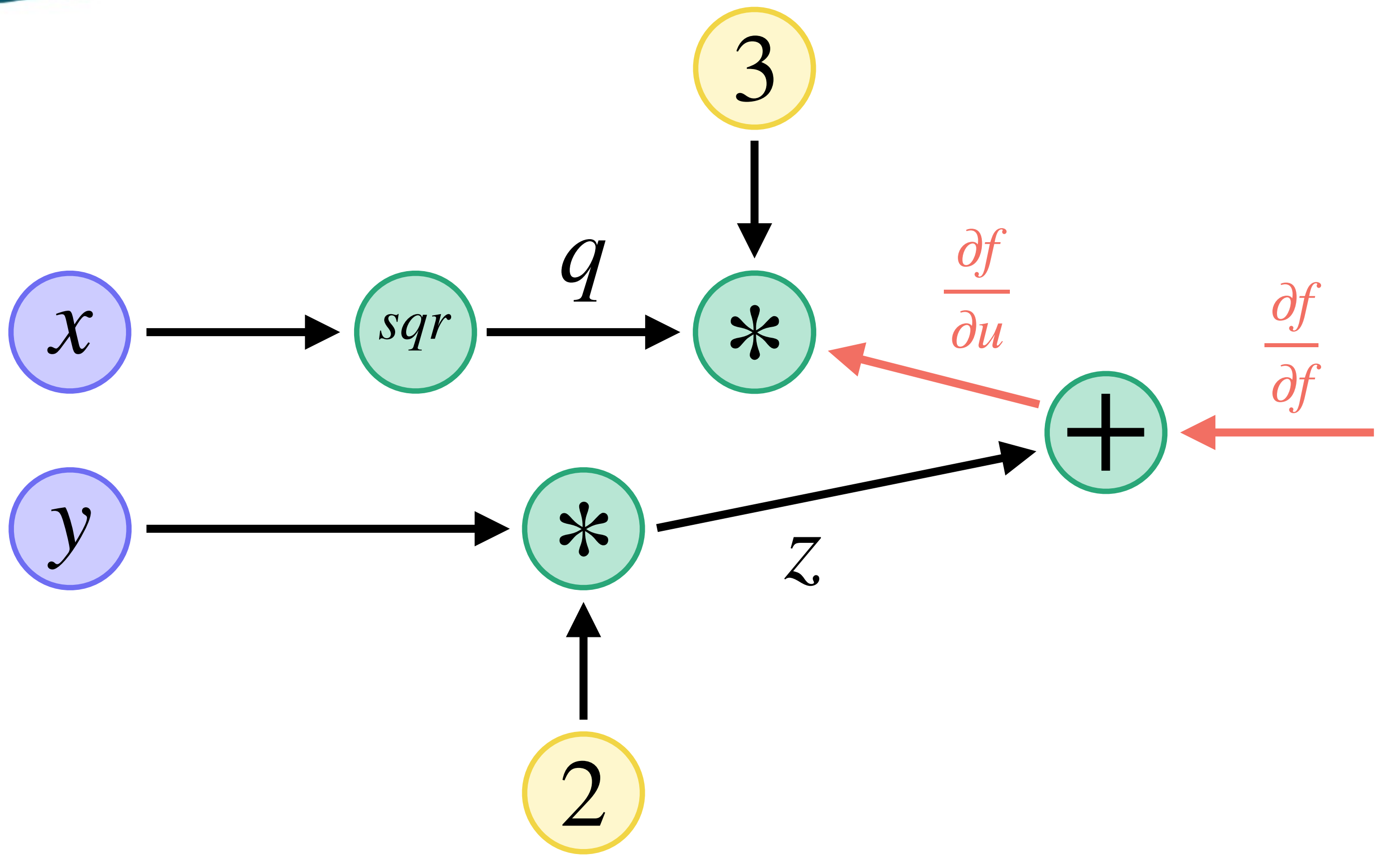
Computational Graphs

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial x}$$

Computational Graphs

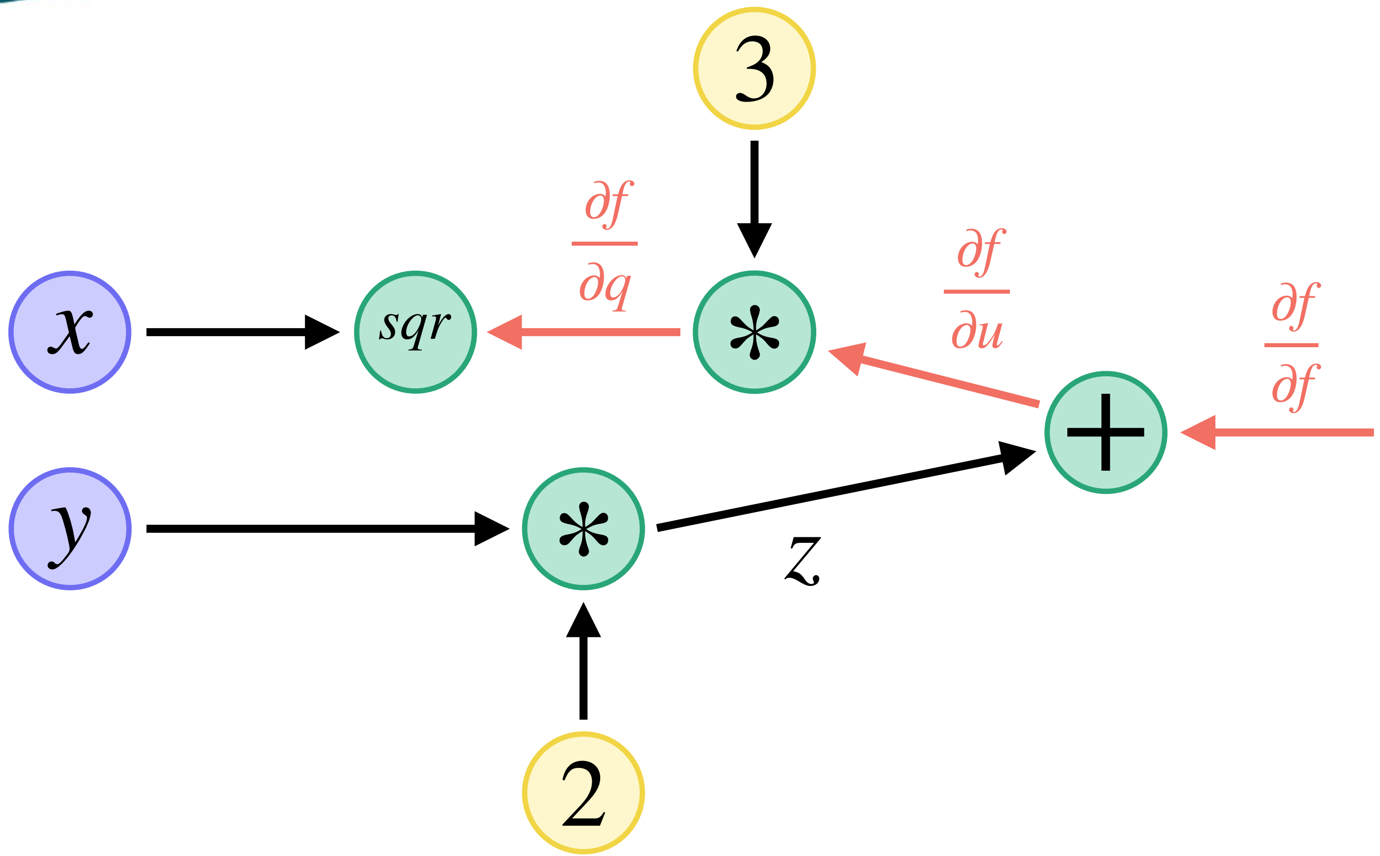


Computational Graphs

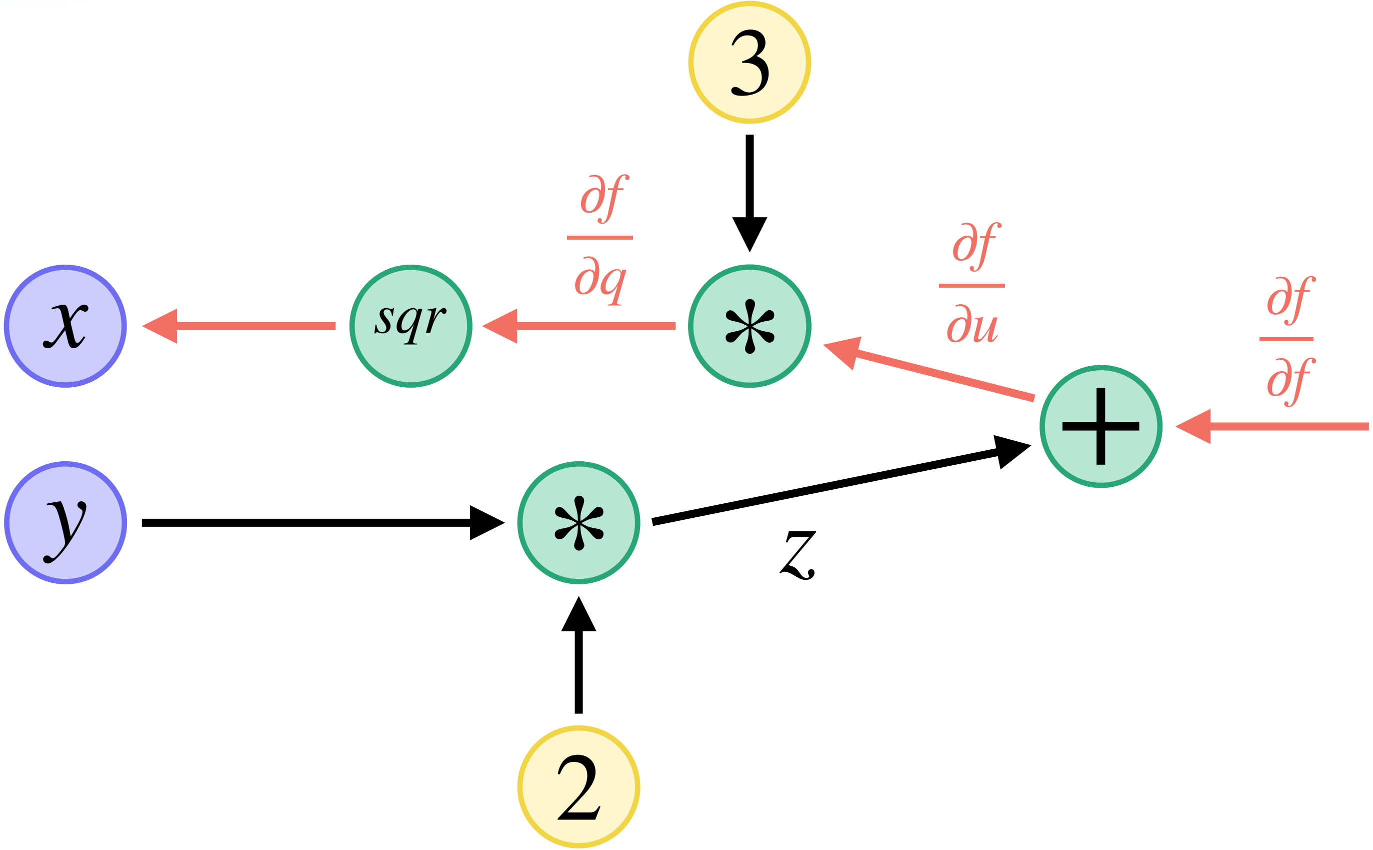


Pearson

Computational Graphs

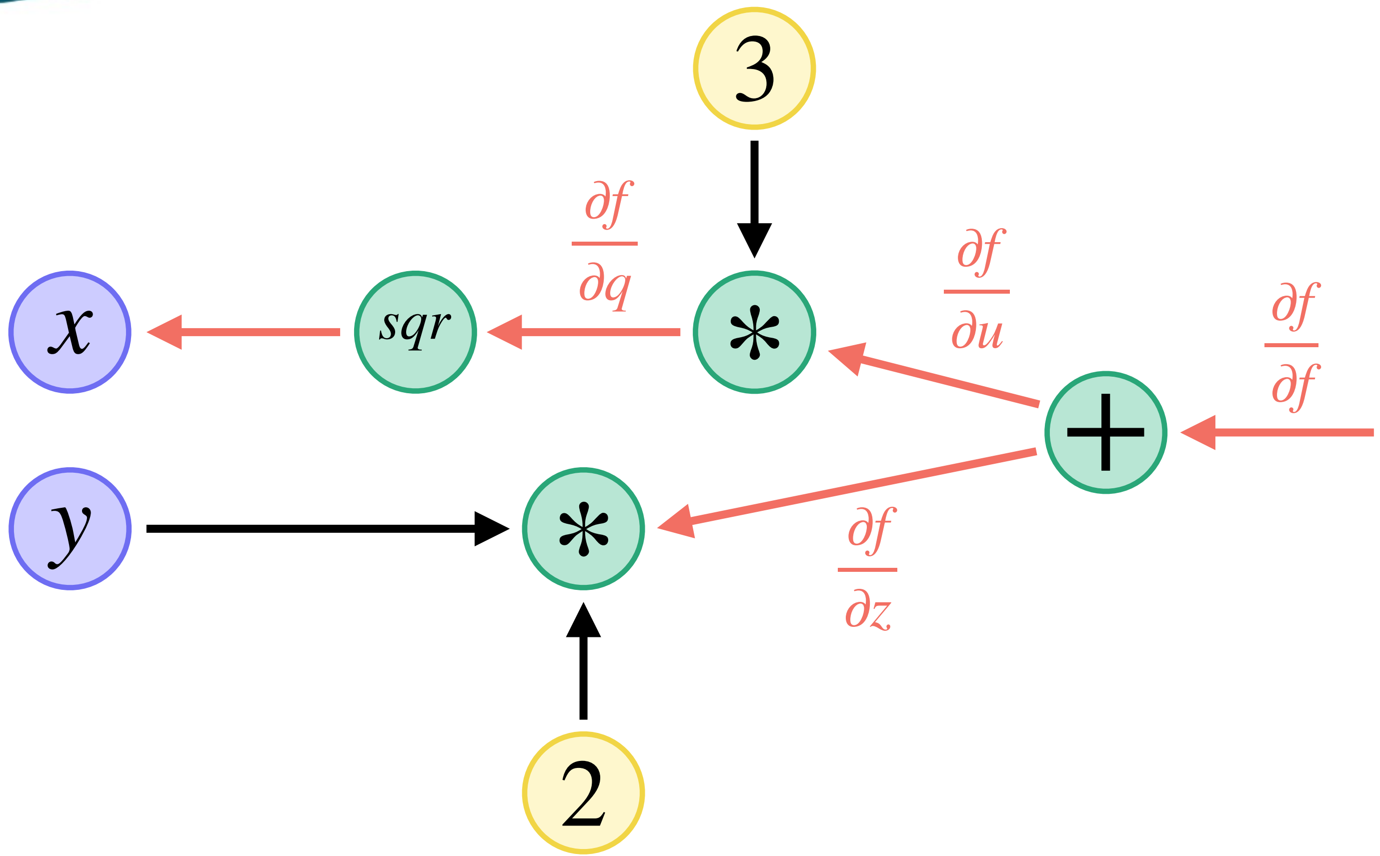


Computational Graphs



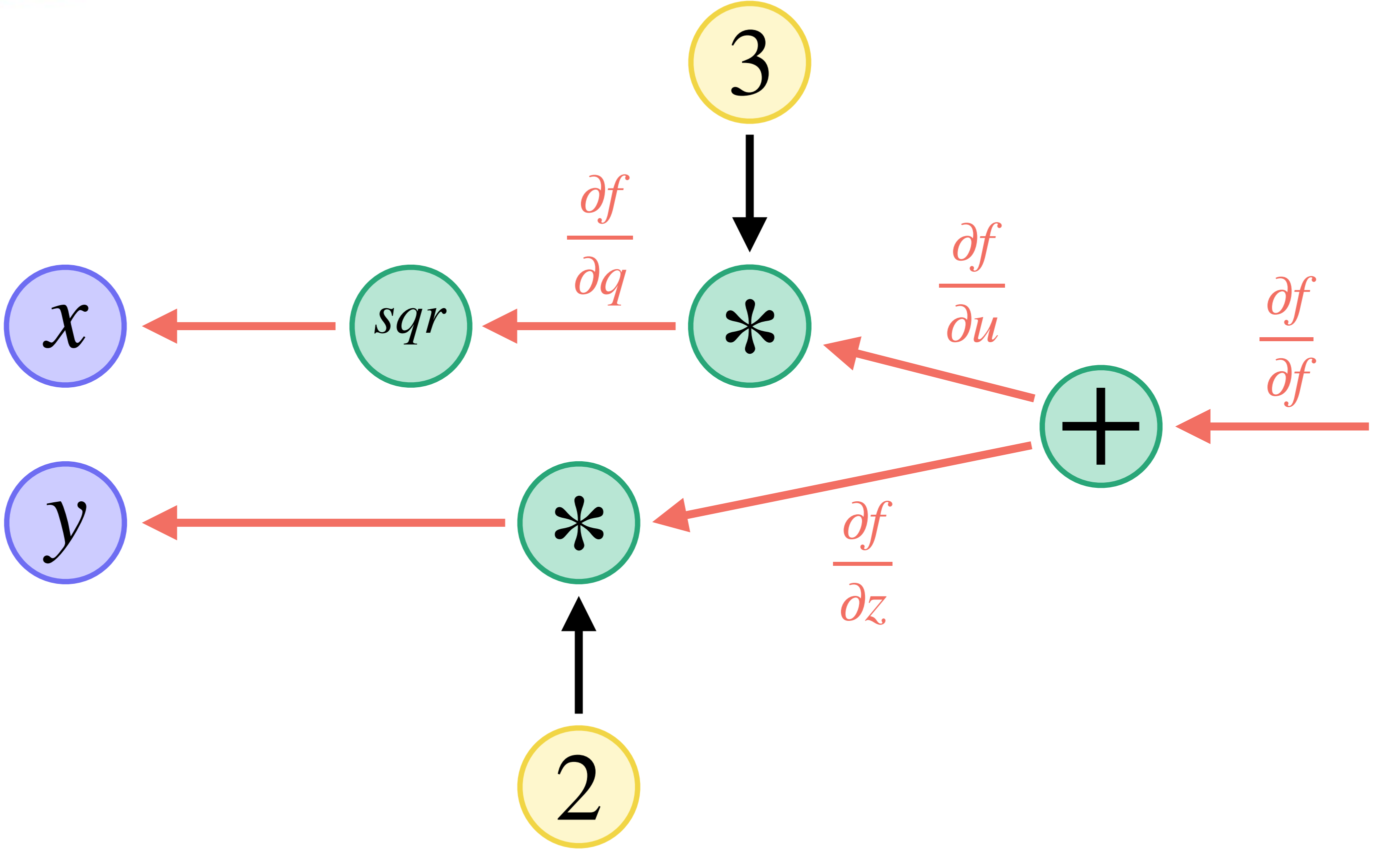
Pearson

Computational Graphs



Pearson

Computational Graphs



Computational Graphs

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial x}$$

Computational Graphs

$$\frac{\partial f}{\partial q} = \frac{\partial f}{\partial u}\frac{\partial u}{\partial q}$$

Computational Graphs

$$\frac{\partial f}{\partial q} = \frac{\partial f}{\partial u}\frac{\partial u}{\partial q}$$

$$\frac{\partial q}{\partial x} = 2x$$

Computational Graphs

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial x}$$

Pearson

Computational Graphs

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial u}\frac{\partial u}{\partial q}2x$$

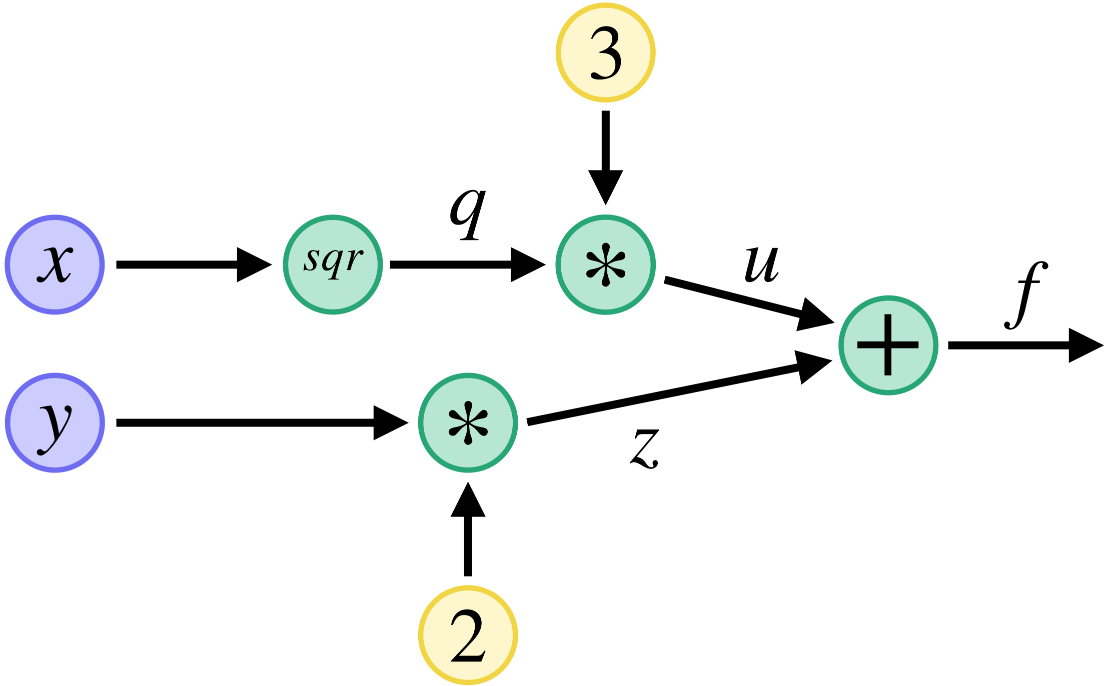
Pearson

Computational Graphs

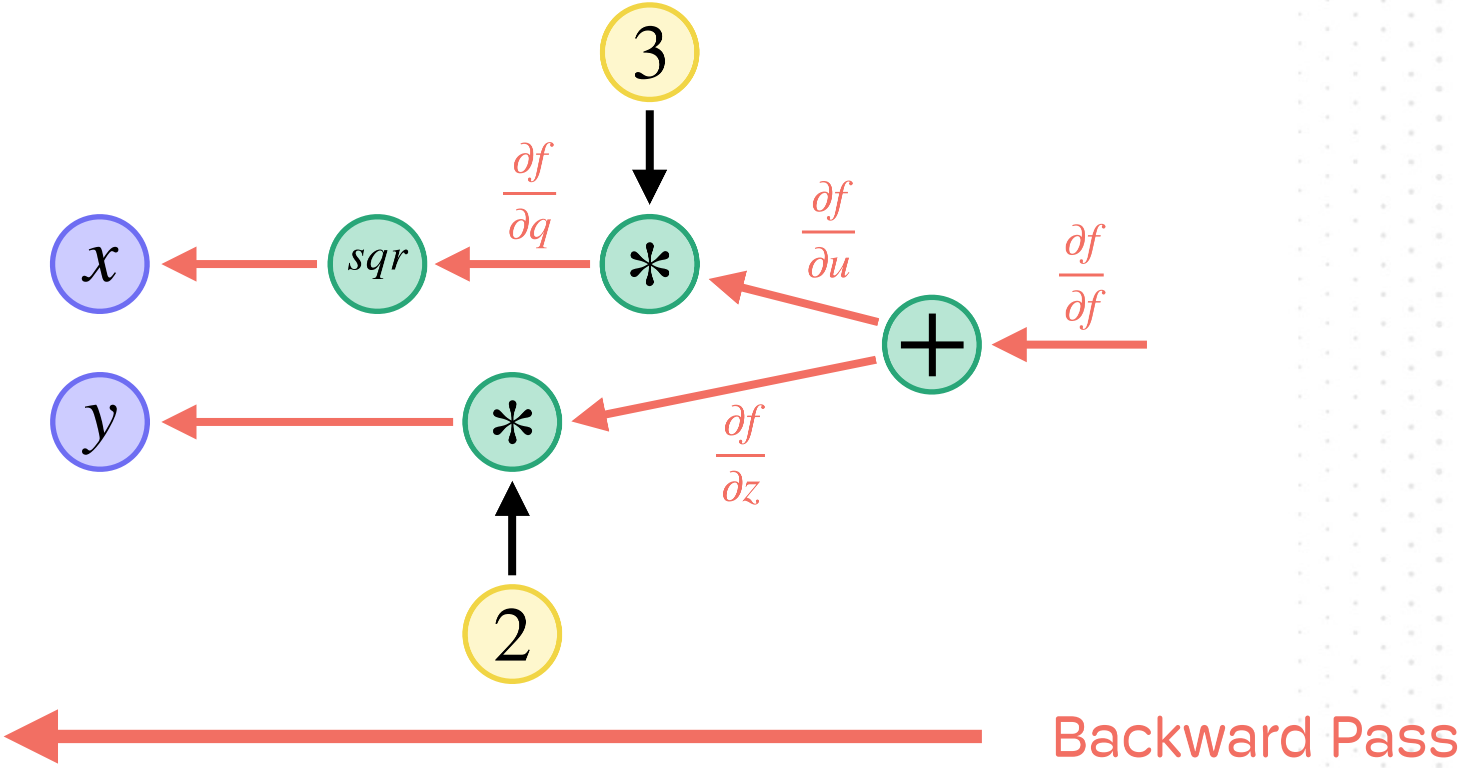
• • •

Computational Graphs

Forward Pass



Computational Graphs



Pearson

Computational Graphs

Each node only needs to know how to:

compute its output (forward)

AND

compute its derivative (backwards)

2.8

Effortless Backpropagation with `torch.autograd`

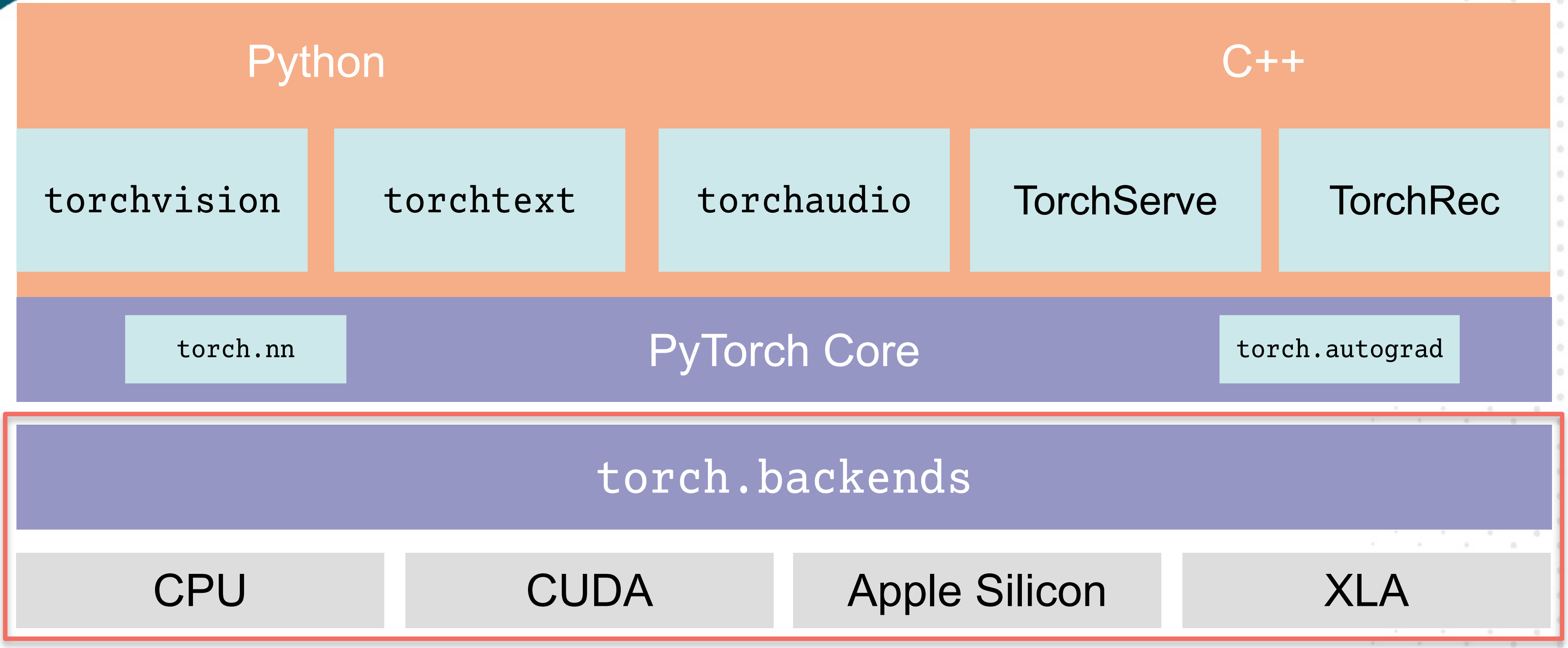
Pearson

Live Coding

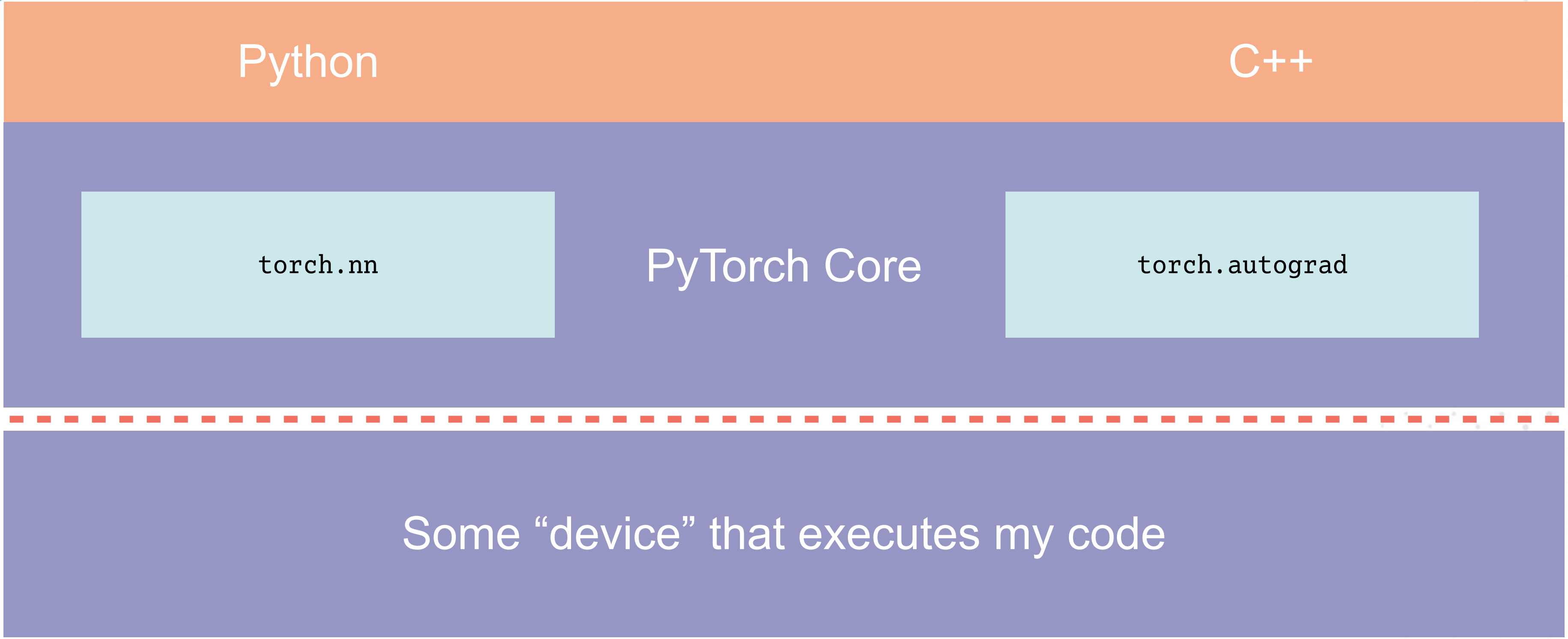
2.9

PyTorch’s Device Abstraction (i.e. GPUs)

PyTorch Layer Cake



PyTorch Layer Cake



2.10

Working with Devices (i.e. GPUs)

Live Coding

2.11

Components of a Learning Algorithm

Live Lecture

2.12

Introduction to Gradient Descent

Pearson

Live Lecture

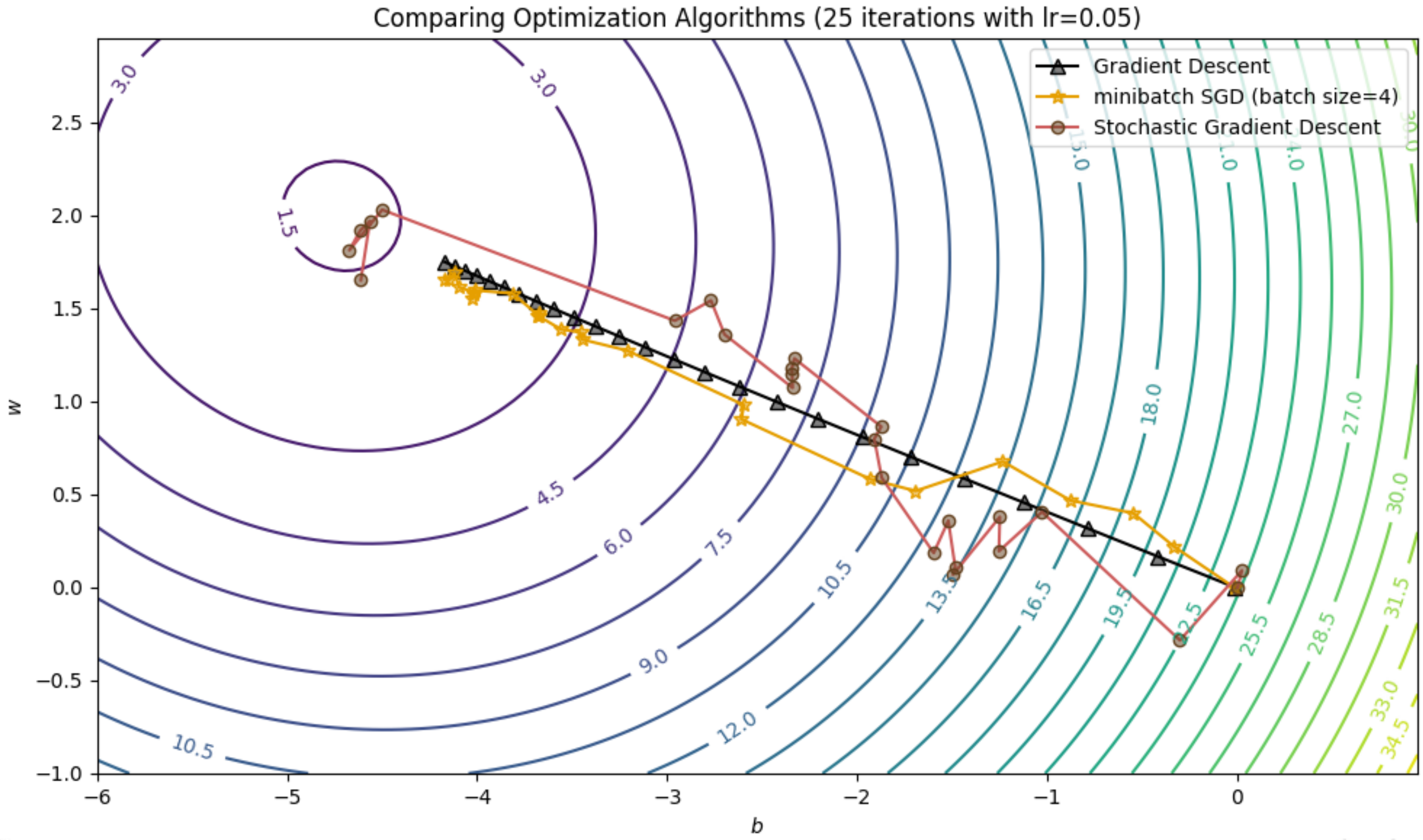
2.13

Getting to Stochastic Gradient Descent (SGD)

Live Lecture

2.14

Comparing Gradient Descent and SGD



Comparing Optimization Algorithms (25 iterations with lr=0.05)
Gradient Descent
minibatch SGD (batch size=4)
Stochastic Gradient Descent
w
b

2.15

Linear Regression with PyTorch

Live Coding

2.16

Perceptrons and Neurons

Live Lecture

2.17

Layers and Activations with torch.nn

Live Coding

2.18

Multi-layer Feedforward Neural Networks (MLP)

Live Coding