Introduction to Deep Learning (CS474)

Lecture 14

Outline

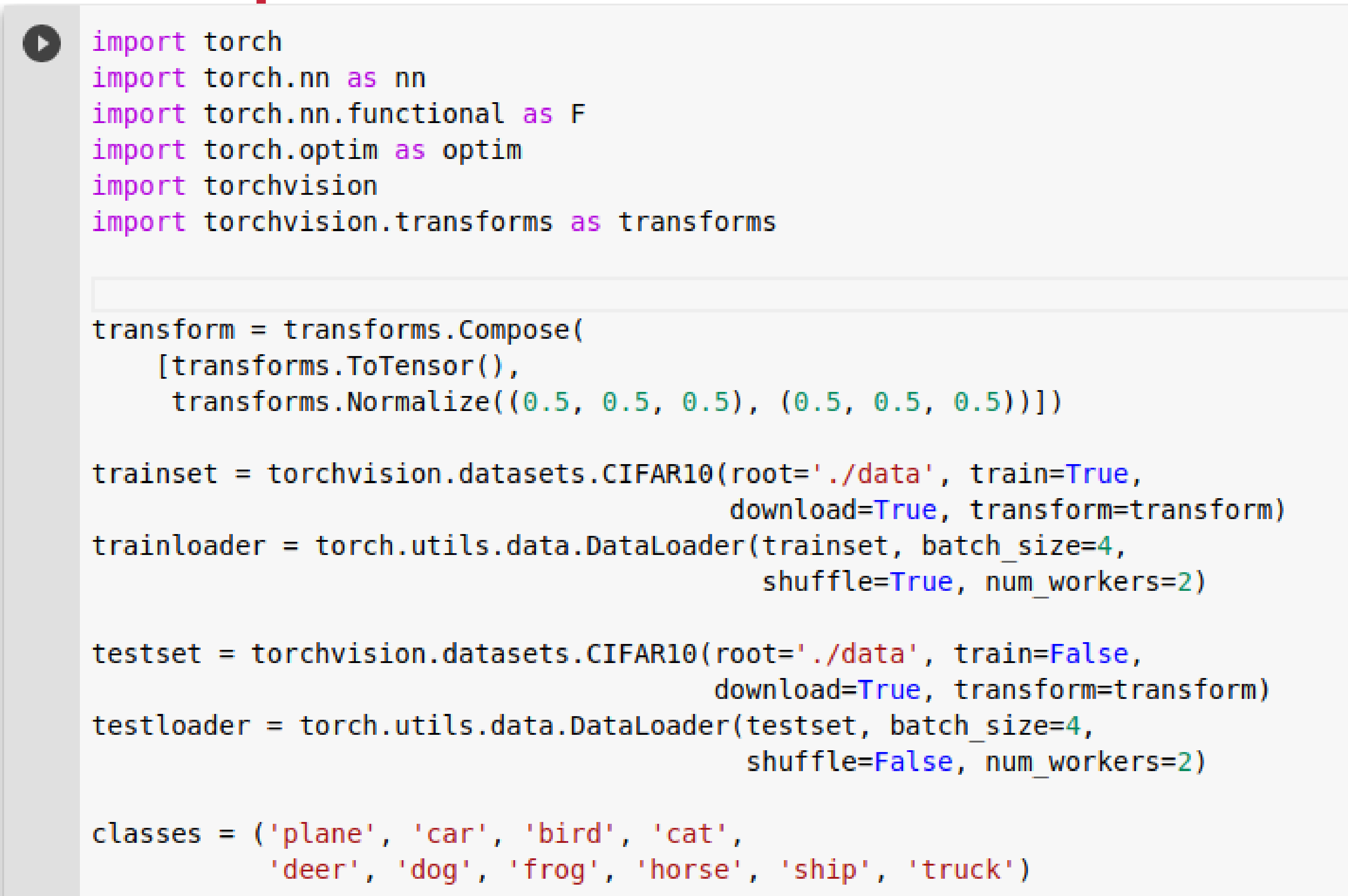
- **Module 2**
  - Brief discussion on *Convolution* through *PyTorch*

Introduction

- We will use another dataset that is simple and a bit more fun. It’s called **CIFAR-10**.
- It has been a computer vision classic for a decade.
- CIFAR-10 consists of 60,000 tiny **32 × 32** color (RGB) images, labeled with an integer corresponding to 1 of 10 classes: airplane (0), automobile (1), bird (2), cat (3), deer (4), dog (5), frog (6), horse (7), ship (8), and truck (9).
- We will use the **torchvision** module to automatically download the dataset and *load* it as a collection of PyTorch tensors.

Slide credit: E. STEVENS, L. ANTIGA, and T. VIEHMANN

Example

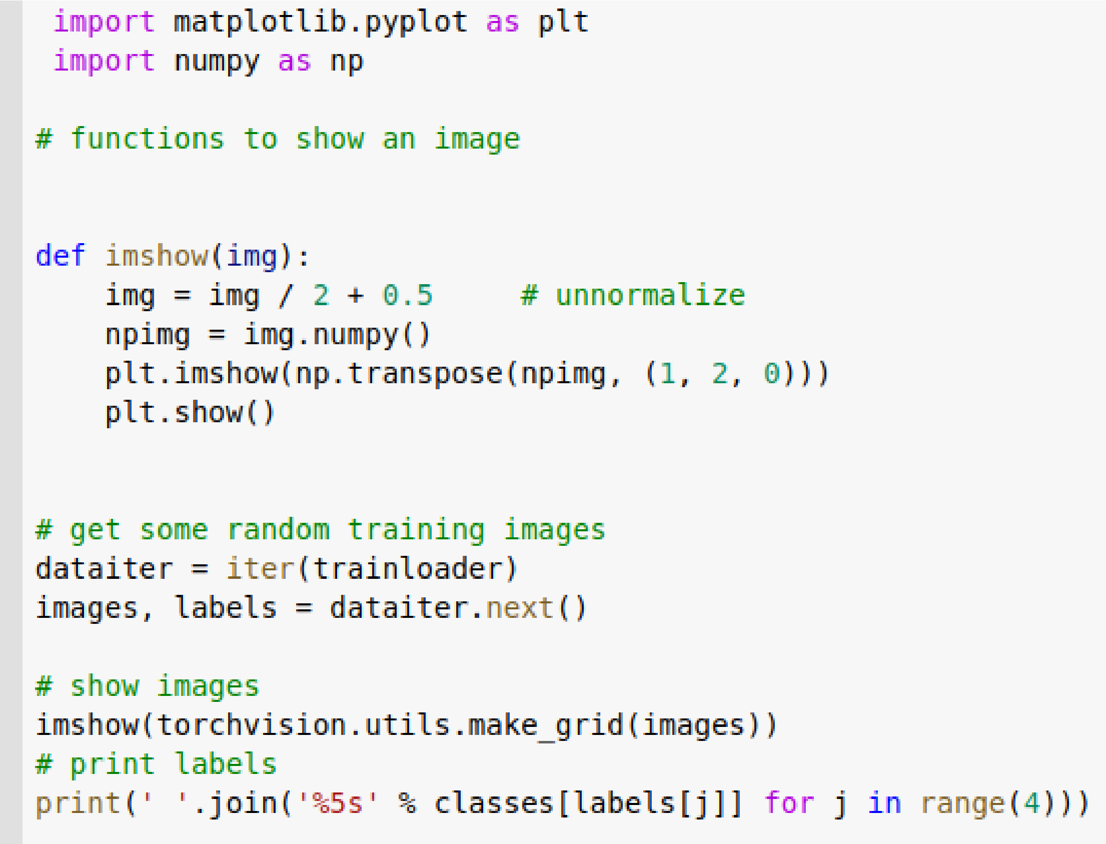


```python
import torch
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim
import torchvision
import torchvision.transforms as transforms


transform = transforms.Compose(
    [transforms.ToTensor(),
     transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))])

trainset = torchvision.datasets.CIFAR10(root='./data', train=True,
                                        download=True, transform=transform)
trainloader = torch.utils.data.DataLoader(trainset, batch_size=4,
                                          shuffle=True, num_workers=2)

testset = torchvision.datasets.CIFAR10(root='./data', train=False,
                                       download=True, transform=transform)
testloader = torch.utils.data.DataLoader(testset, batch_size=4,
                                         shuffle=False, num_workers=2)

classes = ('plane', 'car', 'bird', 'cat',
           'deer', 'dog', 'frog', 'horse', 'ship', 'truck')
```

Slide credit: https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html

Visualizing CIFAR-10

Continuing with the earlier notebook:

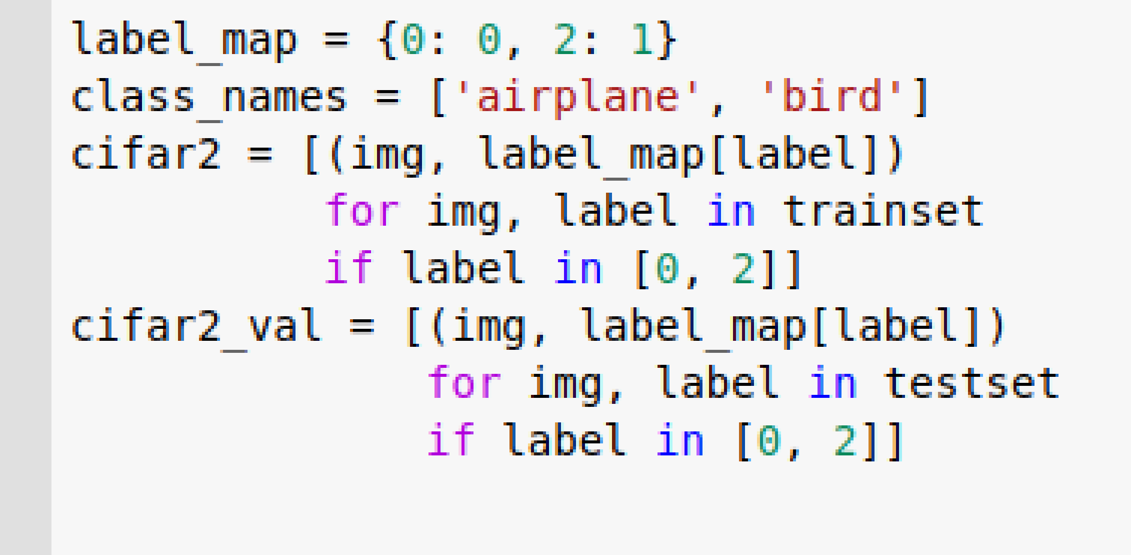


```
 import matplotlib.pyplot as plt
 import numpy as np

# functions to show an image


def imshow(img):
    img = img / 2 + 0.5     # unnormalize
    npimg = img.numpy()
    plt.imshow(np.transpose(npimg, (1, 2, 0)))
    plt.show()


# get some random training images
dataiter = iter(trainloader)
images, labels = dataiter.next()

# show images
imshow(torchvision.utils.make_grid(images))
# print labels
print(' '.join('%5s' % classes[labels[j]] for j in range(4)))
```

Slide credit: https://pytorch.org/tutorials/beginner/blitz/cifar10_tutorial.html

Building the Dataset

Continuing with the earlier notebook:

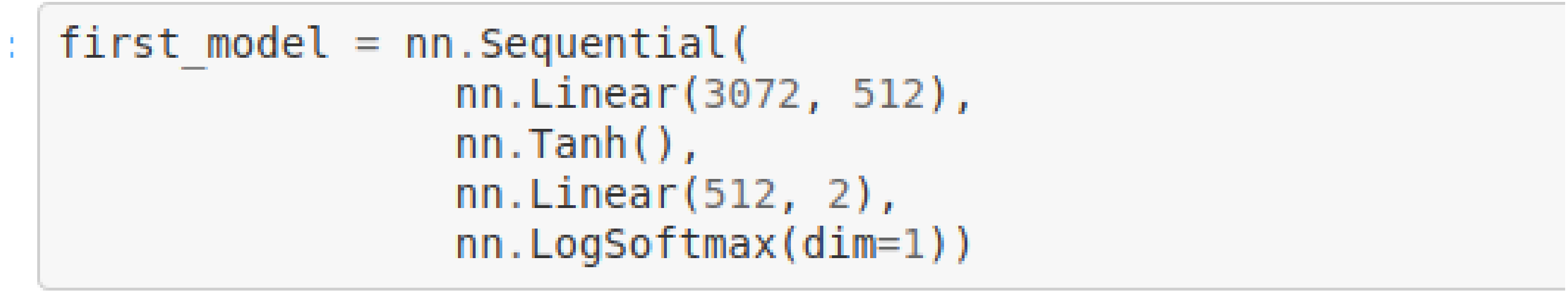


```
label_map = {0: 0, 2: 1}
class_names = ['airplane', 'bird']
cifar2 = [(img, label_map[label])
          for img, label in trainset
          if label in [0, 2]]
cifar2_val = [(img, label_map[label])
              for img, label in testset
              if label in [0, 2]]
```

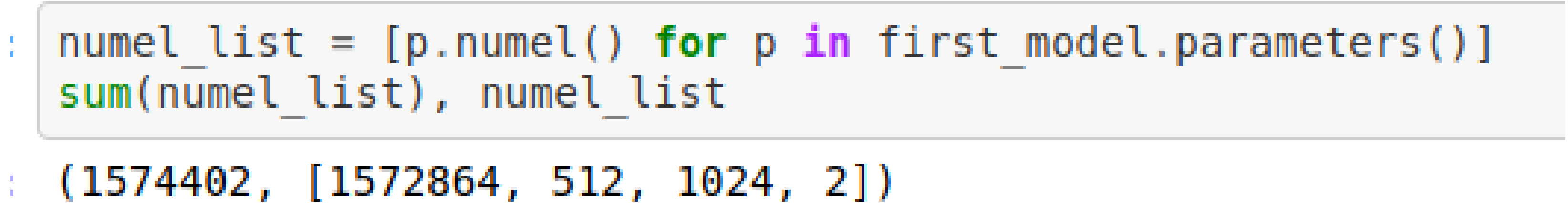
Slide credit: E. STEVENS, L. ANTIGA, and T. VIEHMANN

Initial *Model*

Continuing with the earlier notebook:



```
first_model = nn.Sequential(
                nn.Linear(3072, 512),
                nn.Tanh(),
                nn.Linear(512, 2),
                nn.LogSoftmax(dim=1))
```

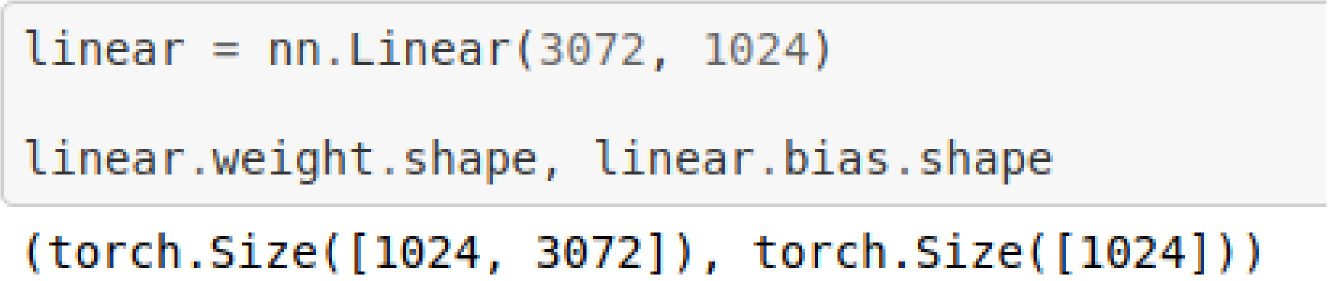


```
numel_list = [p.numel() for p in first_model.parameters()]
sum(numel_list), numel_list
```

```
(1574402, [1572864, 512, 1024, 2])
```

Slide credit: E. STEVENS, L. ANTIGA, and T. VIEHMANN

Check *Model parameters*



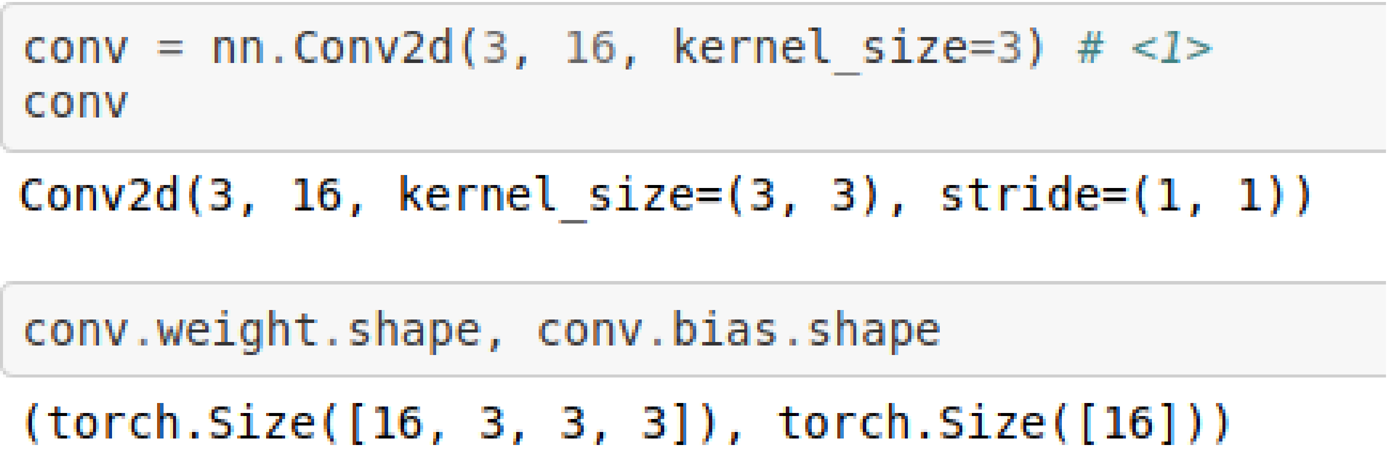
```
linear = nn.Linear(3072, 1024)

linear.weight.shape, linear.bias.shape
```

```
(torch.Size([1024, 3072]), torch.Size([1024]))
```

Slide credit: E. STEVENS, L. ANTIGA, and T. VIEHMANN

Convolution in Action

Continuing with the earlier notebook:



```
conv = nn.Conv2d(3, 16, kernel_size=3) # <1>
conv
```

```
Conv2d(3, 16, kernel_size=(3, 3), stride=(1, 1))
```

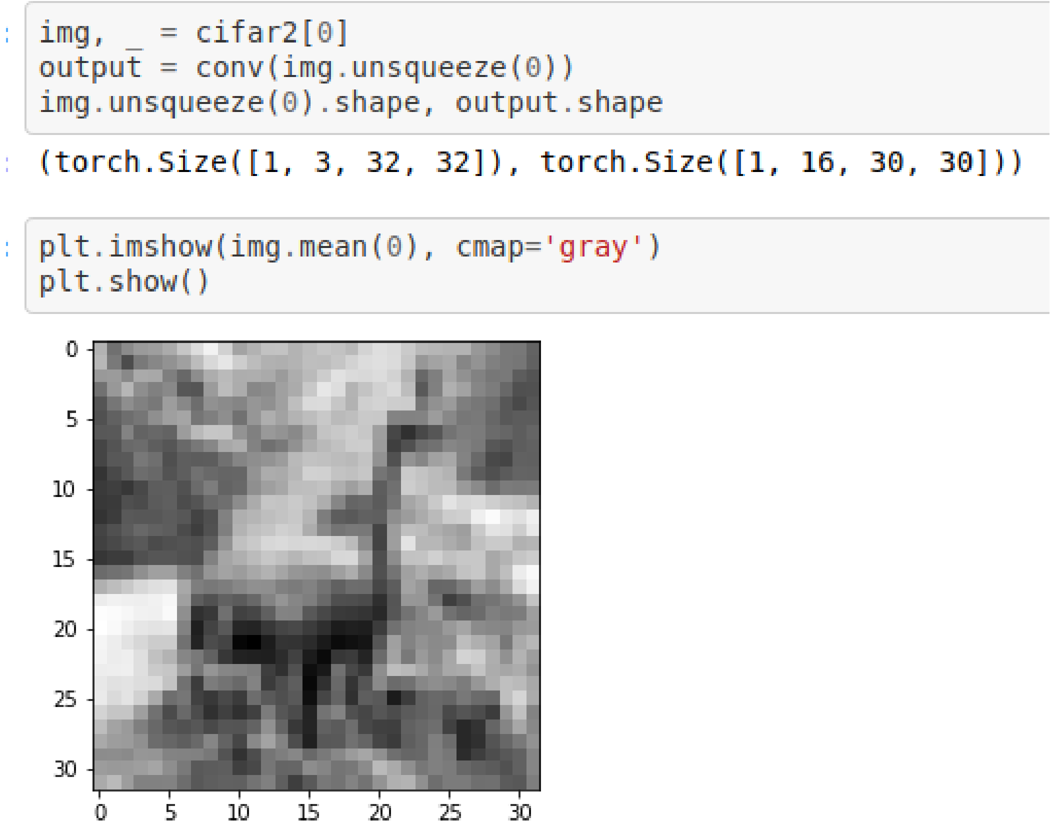
```
conv.weight.shape, conv.bias.shape
```

```
(torch.Size([16, 3, 3, 3]), torch.Size([16]))
```

Slide credit: E. STEVENS, L. ANTIGA, and T. VIEHMANN

Convolution in Action

Continuing with the earlier notebook:



```
img, _ = cifar2[0]
output = conv(img.unsqueeze(0))
img.unsqueeze(0).shape, output.shape
```

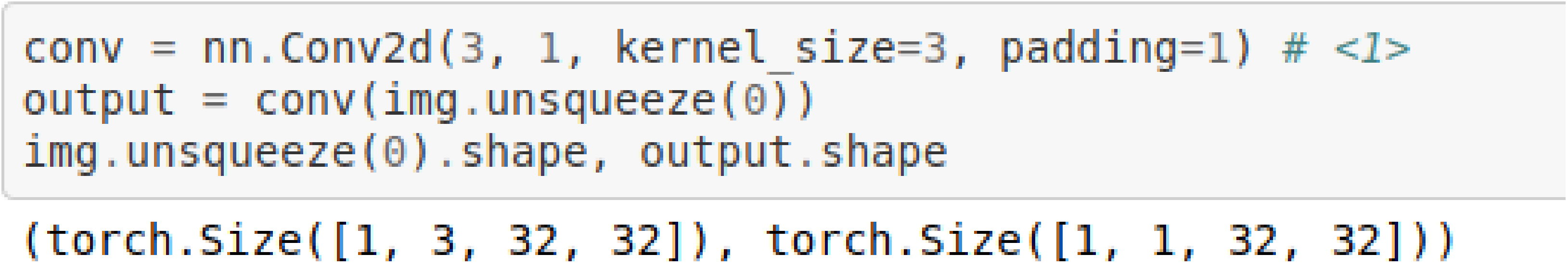
```
(torch.Size([1, 3, 32, 32]), torch.Size([1, 16, 30, 30]))
```

```
plt.imshow(img.mean(0), cmap='gray')
plt.show()
```

Slide credit: E. STEVENS, L. ANTIGA, and T. VIEHMANN

Convolution in Action

Continuing with the earlier notebook:



```
conv = nn.Conv2d(3, 1, kernel_size=3, padding=1) # <1>
output = conv(img.unsqueeze(0))
img.unsqueeze(0).shape, output.shape
```

```
(torch.Size([1, 3, 32, 32]), torch.Size([1, 1, 32, 32]))
```

Slide credit: E. STEVENS, L. ANTIGA, and T. VIEHMANN

References

- All the contents present in the slides are taken from various online resources. Due credit is given in the respective slides.These slides are used for *academic* purposes only.