EENG582: Artificial Neural Networks

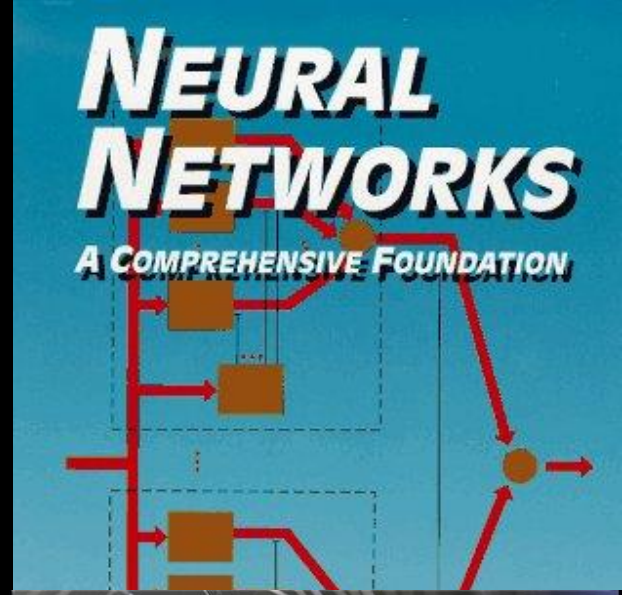
Neural Networks, A Comprehensive Foundation
by S. Haykin
Neural Networks and Deep Learning, by Michael Nielsen
http://neuralnetworksanddeeplearning.com
Introduction to Convolutional Neural Networks, by Vicky Kalogeiton

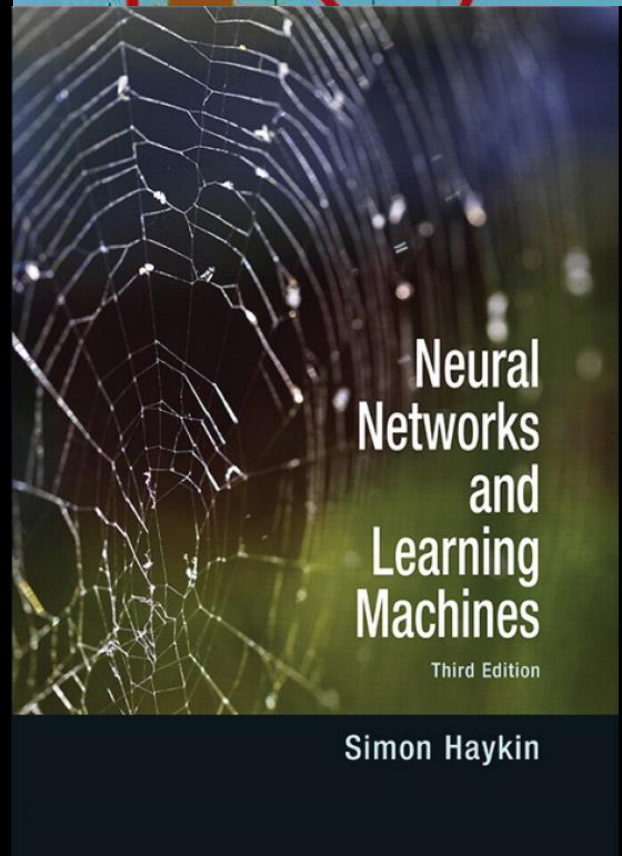
Sections 4.17

4 Multilayer Perceptrons

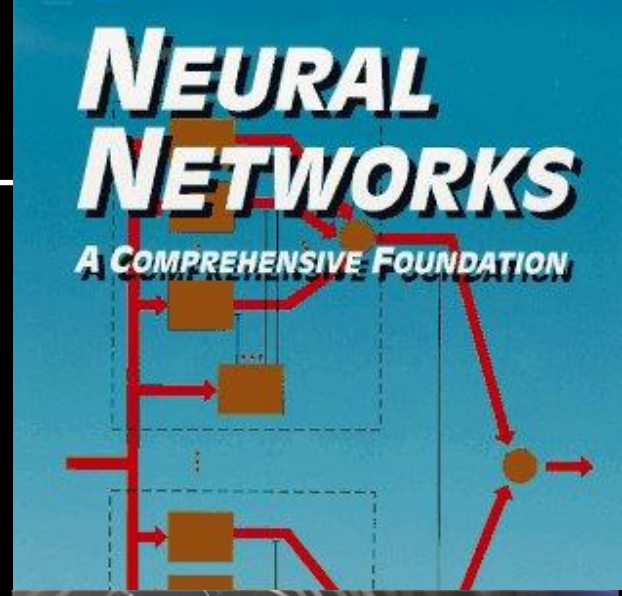
Prof. Dr. Hasan AMCA
Electrical and Electronic Engineering Department
(ee.emu.edu.tr)

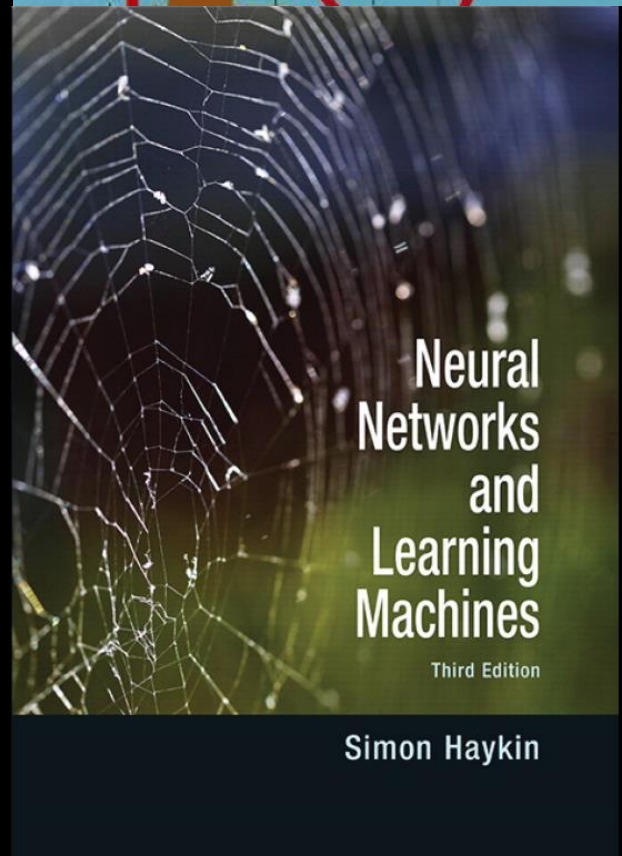
Eastern Mediterranean University
(emu.edu.tr)





4 Multi Layer Perceptrons





4.17 Convolutional Networks

- In this section, we focus on the structural layout of a special class of multilayer perceptrons known collectively as *convolutional networks*, which are well suited for pattern classification.
- The *convolutional networks* are inspired from neurobiologically.
- A *convolutional network* is a multilayer perceptron designed specifically to recognize two-dimensional shapes with a high degree of invariance to translation, scaling, skewing, and other forms of distortion.
- This difficult task is learned in a supervised manner by means of a network whose structure includes the following forms of constraints.

4.17 Convolutional Networks
Structural Constraints

1. *Feature extraction*: Each neuron takes its synaptic inputs from a local receptive field in the previous layer, thereby forcing it to extract local features.

   Once a feature has been extracted, its exact location becomes less important, so long as its position relative to other features is approximately preserved.

2. *Feature Mapping*: Each computational layer of the network is composed of multiple feature maps, with each feature map being in the form of a plane within which the individual neurons are constrained to share the same set of synaptic weights. Feature mapping has the following benefits:

   - shift invariance, forced into the operation of a feature map through the use of convolution with a kernel of small size, followed by a sigmoid function;
   - reduction in the number of free parameters, accomplished through the use of weight sharing.

4.17 Convolutional Networks
Structural Constraints

3. *Subsampling*: Each convolutional layer is followed by a computational layer that performs local averaging and subsampling, whereby the resolution of the feature map is reduced. This operation has the effect of reducing the sensitivity of the feature map's output to shifts and other forms of distortion.

- The multilayer perceptron described in Fig. 4.23 contains approximately 100,000 synaptic connections, but only about 2,600 free parameters.
- This dramatic reduction in the number of free parameters is achieved through the use of weight sharing.
- The capacity of the learning machine is thereby reduced, which in turn improves the machine's generalization ability.

Watch the video at: https://www.youtube.com/watch?v=YRhxdVk_sIs

4.17 Convolutional Networks
Structural Constraints

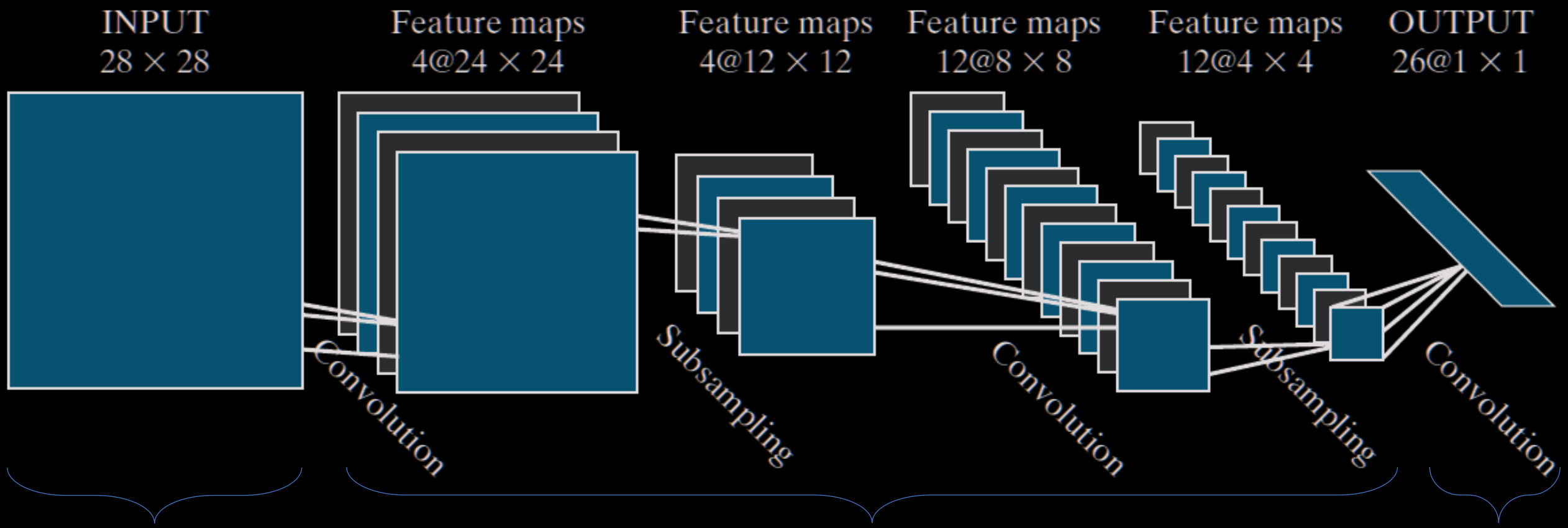


Fig. 4.23 Convolutional network for image processing such as handwriting recognition (Simon Hykin, Neural Networks and Learning Machines, 3rd Ed.).

4.17 Convolutional Networks
Matrix Multiplication

- To multiply a matrix by another matrix, we need to do the product of rows and columns. For example, to do $\mathbf{A} \times \mathbf{B} = \mathbf{C}$, we do the following:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 2 & 3 & 2 \end{bmatrix} \times \begin{bmatrix} 2 & 4 & 6 \\ 4 & 5 & 6 \\ 2 & 3 & 2 \end{bmatrix} = \begin{bmatrix} 16 & 23 & 24 \\ 40 & 63 & 66 \\ 20 & 29 & 34 \end{bmatrix}$$

- The product is

$$(1\ 2\ 3).(2\ 4\ 2) = 1 \times 2 + 2 \times 4 + 3 \times 2 = 16$$

$$(4\ 5\ 6).(4\ 5\ 3) = 4 \times 4 + 5 \times 5 + 6 \times 3 = 63$$

$$(2\ 3\ 2).(6\ 6\ 2) = 2 \times 6 + 3 \times 6 + 2 \times 2 = 34$$

...

- Dimensions => (m × n) . (n × p) = (m × p)

4.17 Convolutional Networks
Matrix Dot Product (Element-by-Element Product)

- The dot product of two matrices is the element-by-element product. Consider the following 2 matrices **A** and **B** with same number of elements. With Dimensions => (m×m) . (m×m) = (m×m), then **C=A. B** is calculated as

Example 1

$$\begin{bmatrix}1 & 1\\ 4 & 6\end{bmatrix}.\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} = 7$$

Hence, the first element of the resulting matrix is $\begin{bmatrix}7 & x\\ x & x\end{bmatrix}$

Element-by-element product is
1×1+1×0+4× 0+6×1= 1+0+0+6=7
Dimensions => (2×2).(2×2)=(2×2)

https://www.bing.com/videos/search?q=convolutional+neural+network+in+matlab&&view=detail&mid=142805453045DED45E82142805453045DED45E82&&FORM=VRDGAR&ru=%2Fvideos%2Fsearch%3Fq%3Dconvolutional%2Bneural%2Bnetwork%2Bin%2Bmatlab%26FORM%3DHDRSC3

Example 2

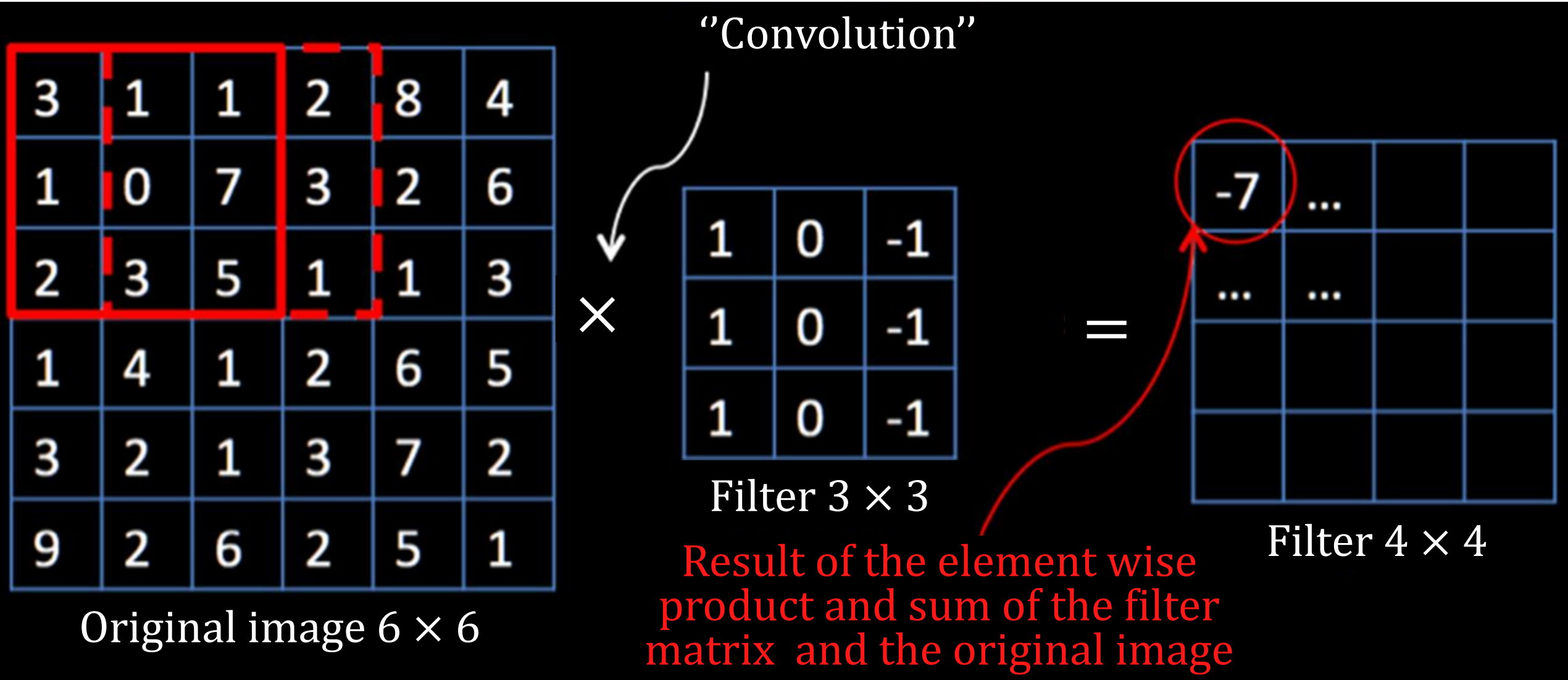
$$\begin{bmatrix}3 & 1 & 1\\ 1 & 0 & 7\\ 2 & 3 & 5\end{bmatrix}.\begin{bmatrix}1 & 0 & -1\\ 1 & 0 & -1\\ 1 & 0 & -1\end{bmatrix} = \begin{bmatrix}-7 & \times & \times\\ \times & \times & \times\\ \times & \times & \times\end{bmatrix}$$

That is, element-by-element product is

3×1+1×0+1×-1+1×1+0×0+7×-1+2×1+3×0+5×-1= 3+0–1+1+0–7+2+0– 5=-7

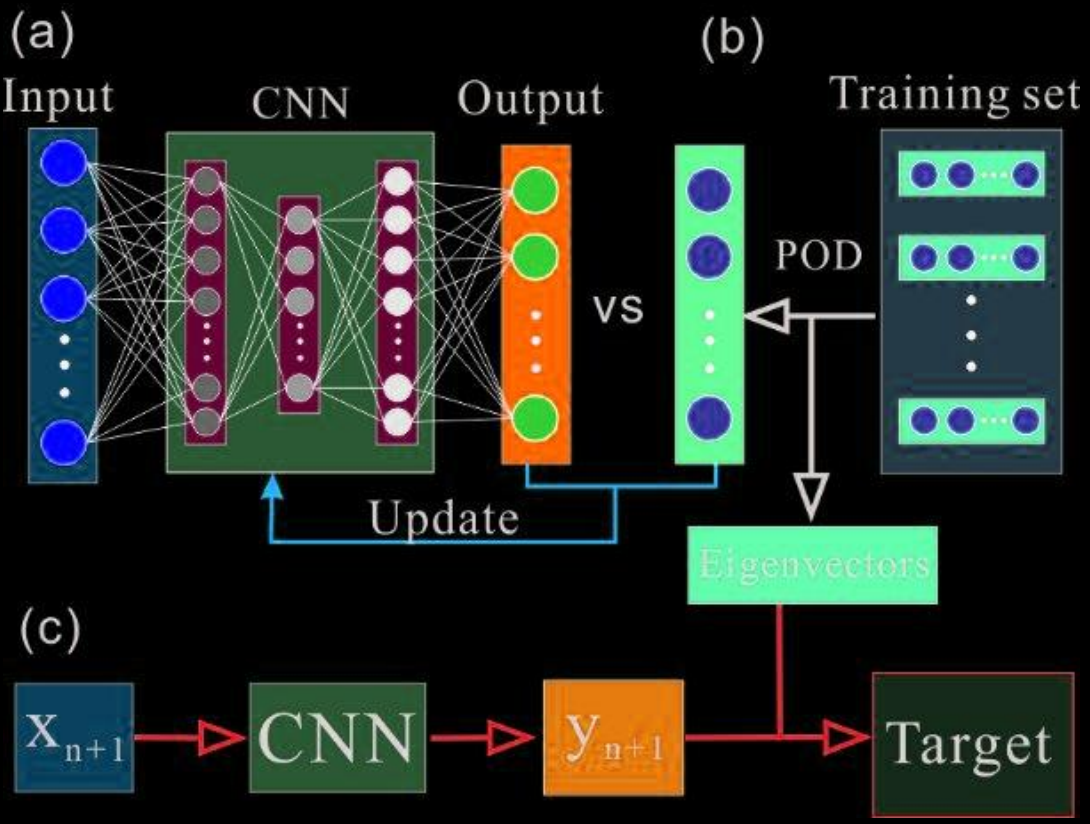
https://laptrinhx.com/convolutional-neural-networks-with-keras-697984035/

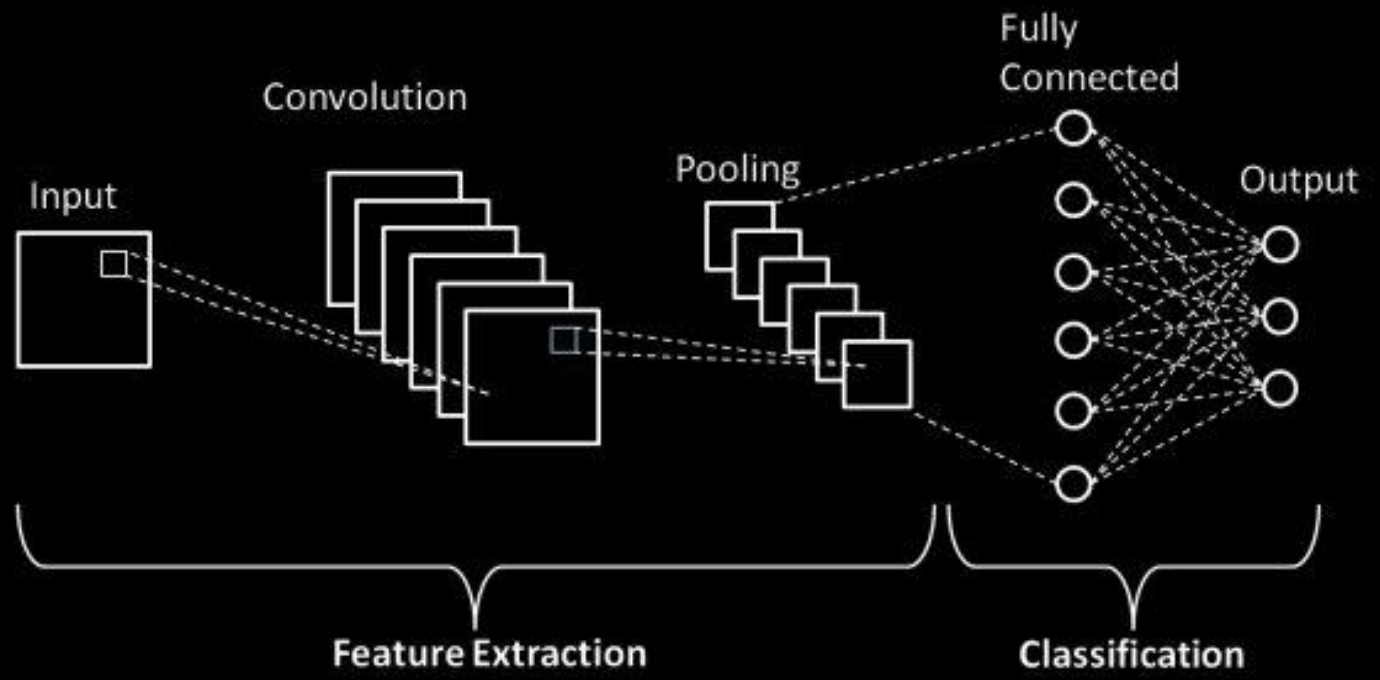
4.17 Convolutional Networks
Matrix Dot Product (Element-by-Element Product)



Source: https://laptrinhx.com/convolutional-neural-networks-with-keras-697984035/

4.17 Convolutional Networks
Structural Constraints





https://www.researchgate.net/publication/335202240_Tomographic_reconstruction_for_3D_flame_imaging_using_convolutional_neural_networks/figures?lo=1

https://www.researchgate.net/publication/336805909_A_High-Accuracy_Model_Average_Ensemble_of_Convolutional_Neural_Networks_for_Classification_of_Cloud_Image_Patches_on_Small_Datasets/figures?lo=1

4.17 Convolutional Networks
Structural Constraints

- Figure 4.23 shows architectural layout of a image processing convolutional network made up of *an input layer*, *four hidden layers*, and *an output layer*.
- **Input layer** is made up of 28 × 28 sensory nodes, receives the images of different characters that have been approximately centered and normalized in size. Thereafter, the computational layouts alternate between convolution and subsampling.
- **1st hidden layer** performs convolution. It consists of four feature maps, with each feature map consisting of 24 × 24 neurons. Each neuron is assigned a receptive field of size 5 × 5.
- **2nd hidden layer** performs subsampling and local averaging. It consists of four feature maps made up of 12 × 12 neurons. Each neuron has a receptive field of size 2 × 2, a trainable coefficient, a trainable bias, and a sigmoid activation function. Trainable coefficient and bias control operating point of the neuron; for example, if the coefficient is small, neuron operates in a quasilinear mode.

4.17 Convolutional Networks
Structural Constraints

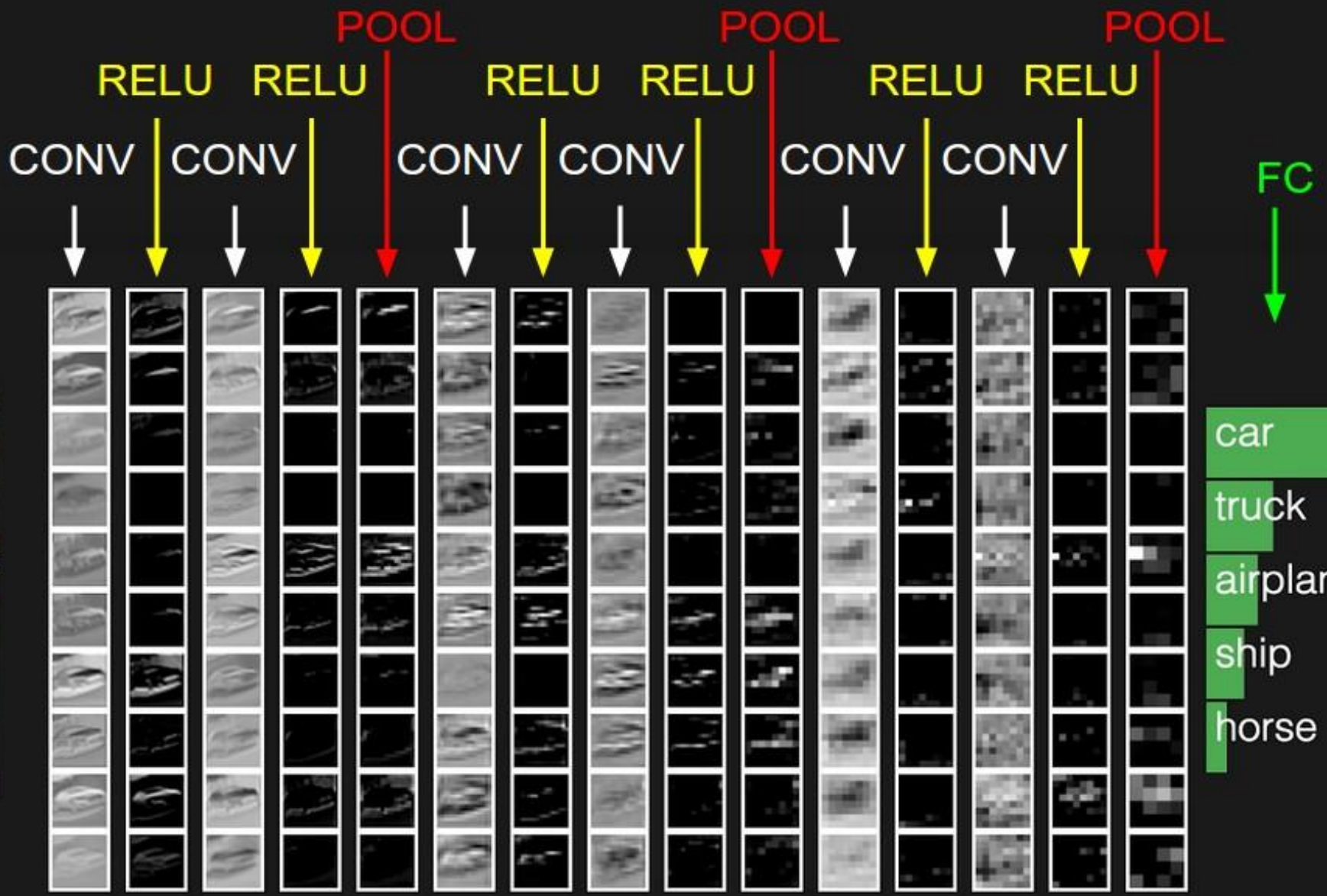
- **3rd hidden layer** performs a second convolution. It consists of 12 feature
- maps, with each feature map consisting of 8 × 8 neurons . Each neuron in this hidden layer may have synaptic connections from several feature maps in the previous hidden layer.
- **4th hidden layer** performs a second subsampling and local averaging. It consists of 12 feature maps, but with each feature map consisting of 4 × 4 neurons. Otherwise, it operates in a manner similar to first subsampling layer.
- **Output layer** performs one final stage of convolution. It consists of 26 neurons, with each neuron assigned to one of 26 possible characters. As before, each neuron is assigned a receptive field of size 4 × 4.

4.17 Convolutional Networks
Structural Constraints

- The activations of an example ConvNet architecture (previous page).
- The initial volume stores the raw image pixels (left) and the last volume stores the class scores (right).
- Each volume of activations along the processing path is shown as a column.
- Since it's difficult to visualize 3D volumes, we lay out each volume's slices in rows.
- The last layer volume holds the scores for each class, but here we only visualize the sorted top 5 scores and print the labels of each one.
- The full web-based demo is shown in the header of our website.
- The architecture shown here is a tiny VGG Net, which we will discuss later.

https://cs231n.github.io/convolutional-networks/

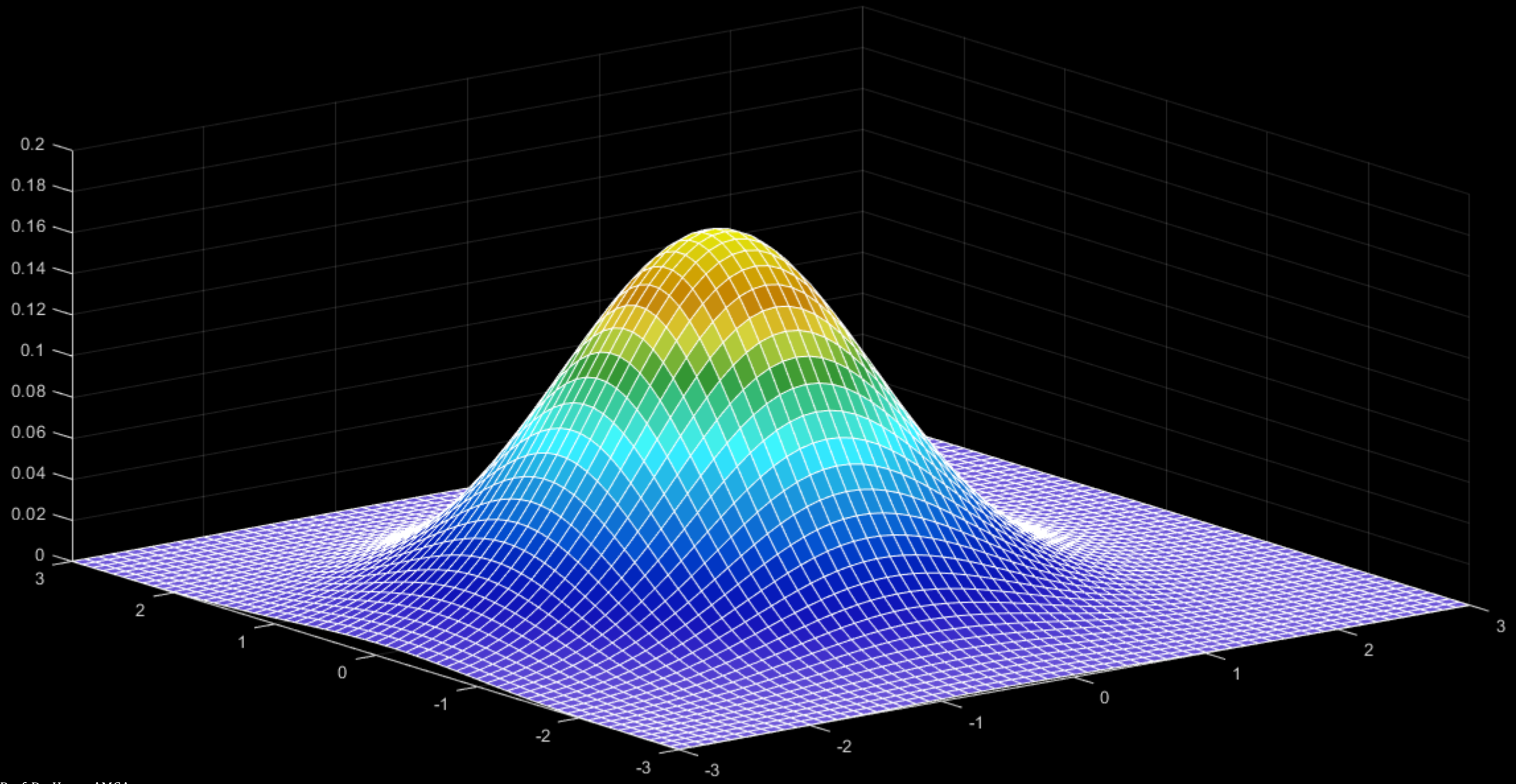
4.17 Convolutional Networks
Structural Constraints



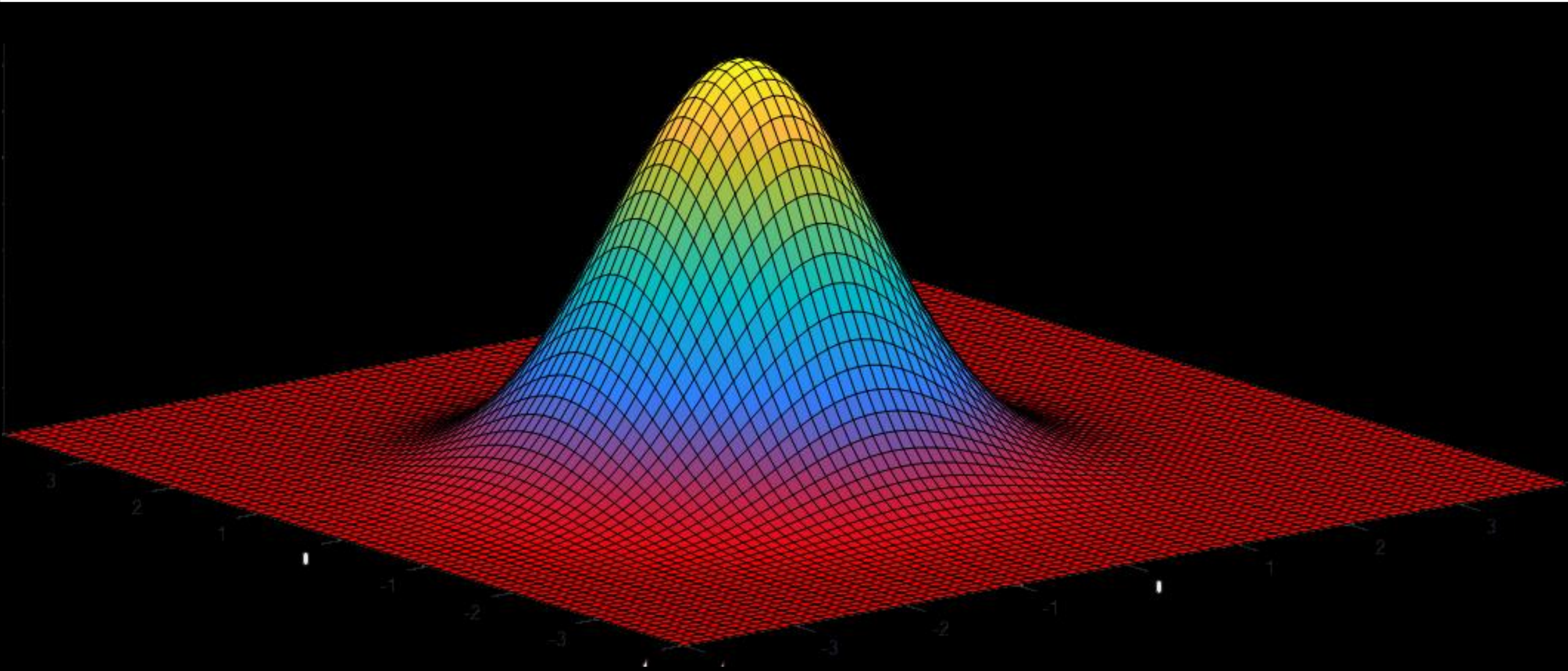
https://cs231n.github.io/convolutional-networks/

End of Section 4.17

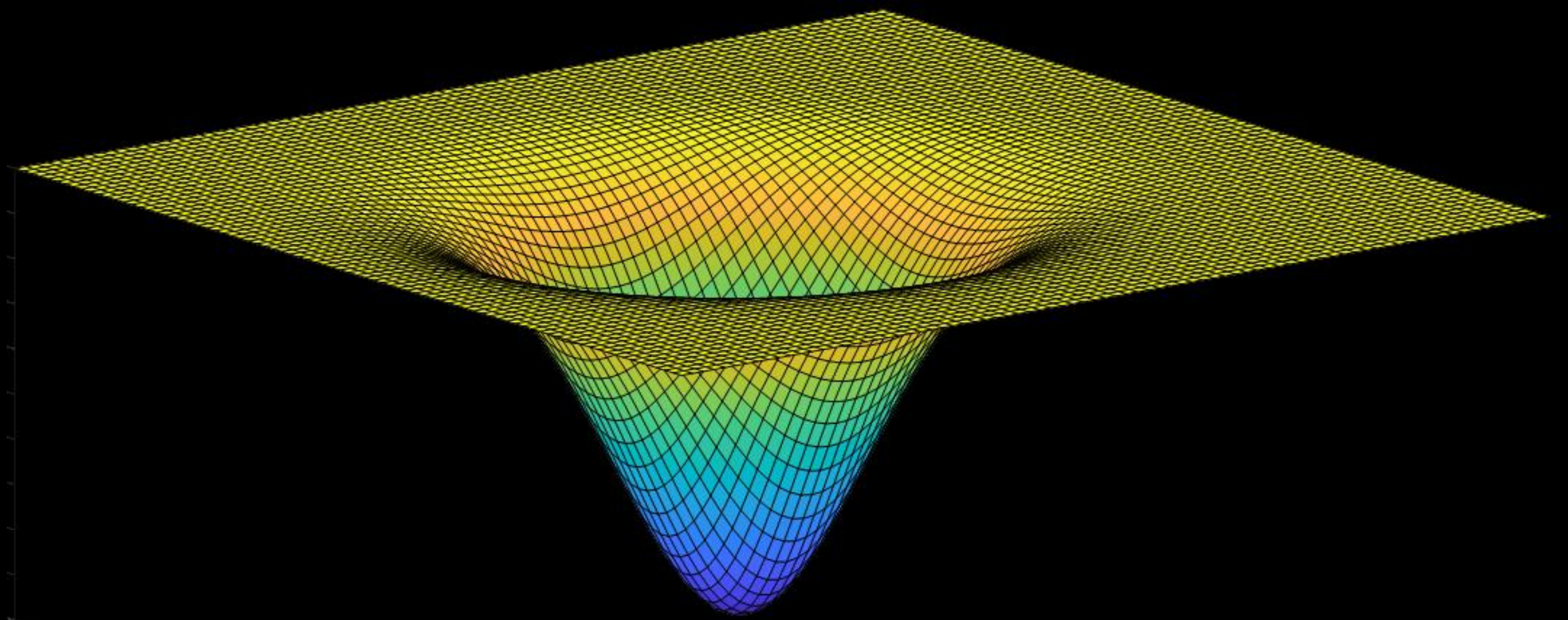
Multivariate Gaussian PDF Plot



Multivariate Gaussian PDF Plot



Multivariate Gaussian PDF Plot



Tips and Tricks:
Getting Started Using Optimization with MATLAB

https://www.mathworks.com/videos/tips-and-tricks-getting-started-using-optimization-with-matlab-81594.html?s_iid=doc_rw_GD_footer