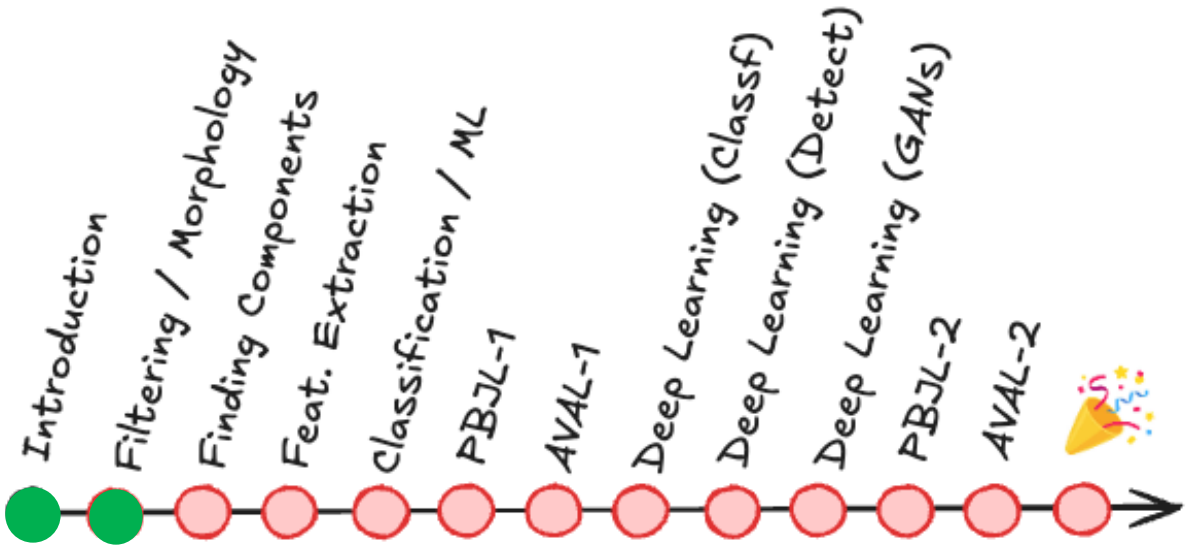
# Lecture 02 - Image Filtering

Prof. André Gustavo Hochuli

gustavo.hochuli@pucpr.br
aghochuli@ppgia.pucpr.br

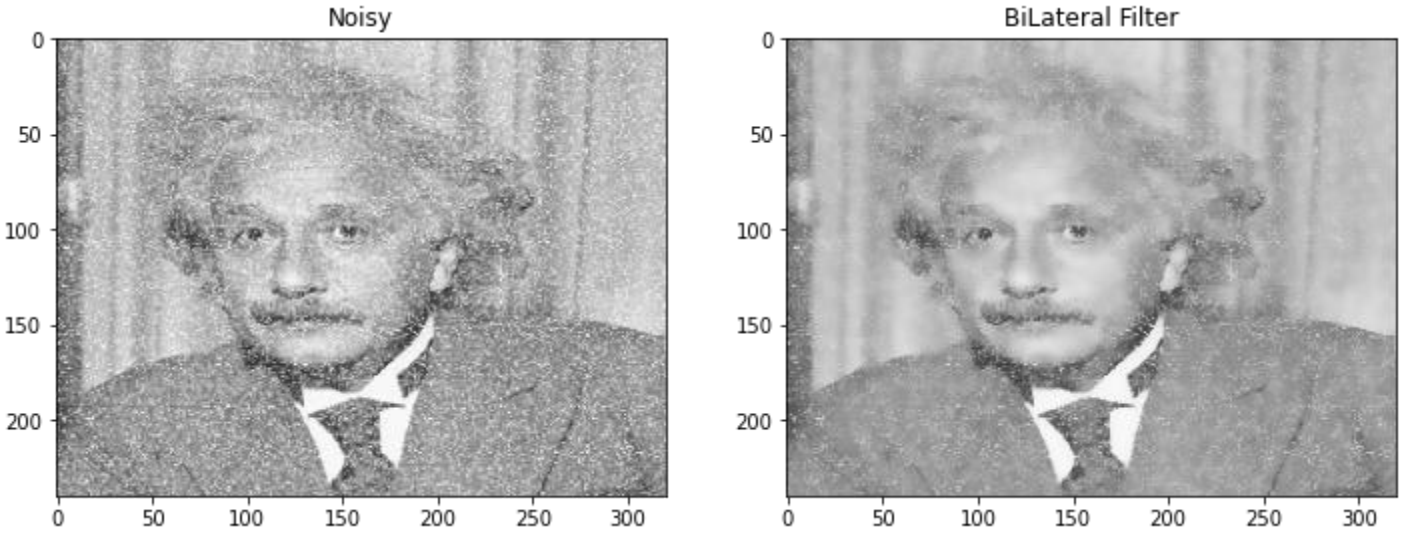
# Topics

- Discussion of Practice 01
- Image Filtering
  - Convolution
  - Mean, Median, Gaussian Filters
- Practice



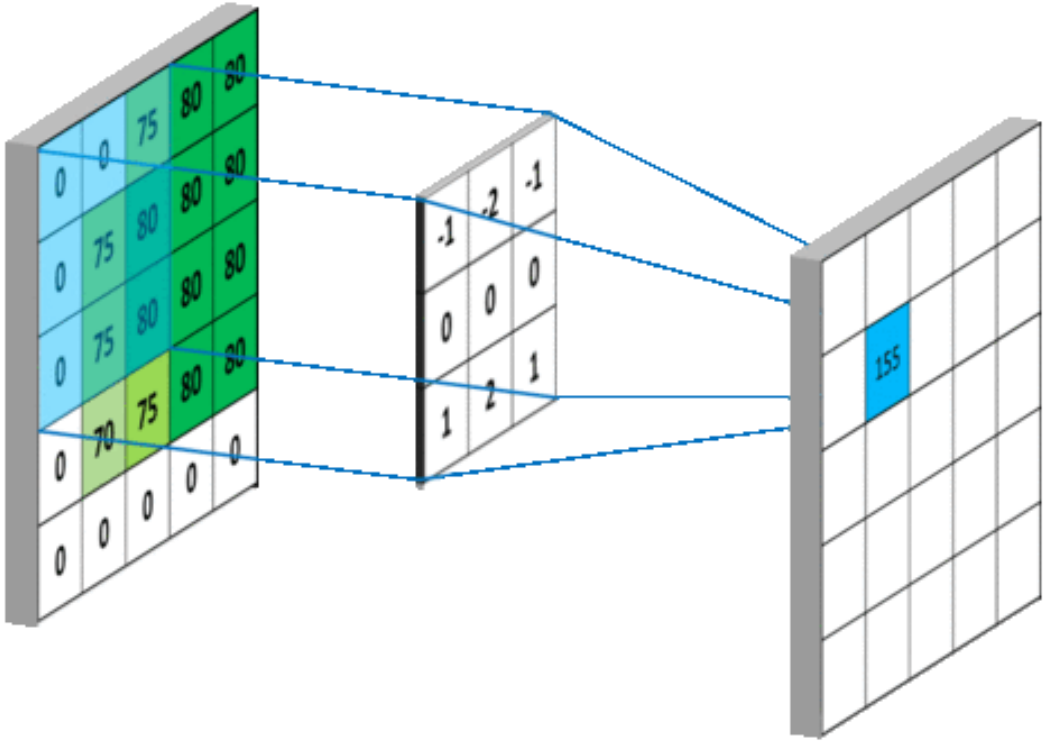
# Image Filtering

- **Image Enhacement**
- **Noise Reduction**
- **Mathematical Operations**

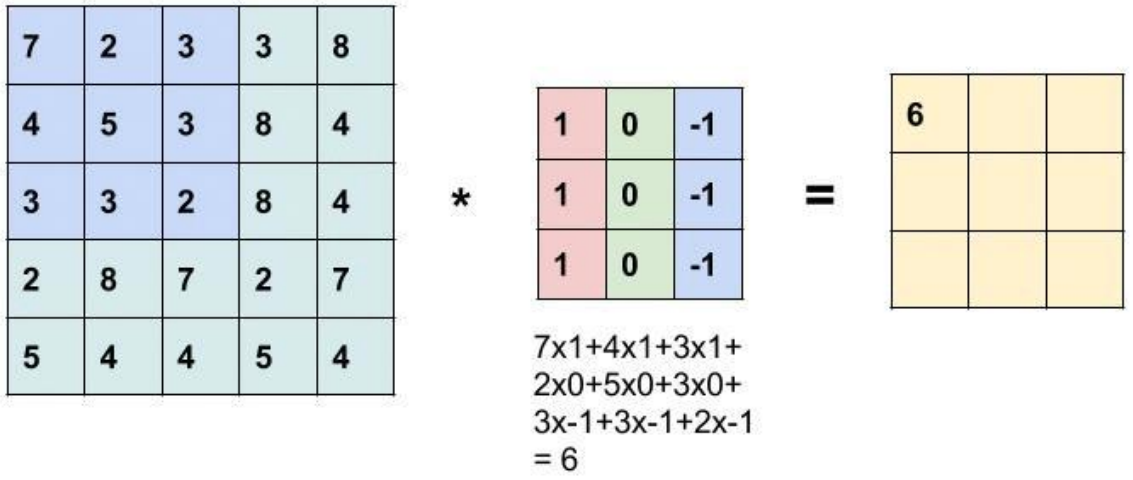


# Convolution

- Slides a kernel (a.k.a convolution filter) in the entire image
- Transforming the pixel in the center of the kernel by the weights of its neighbors

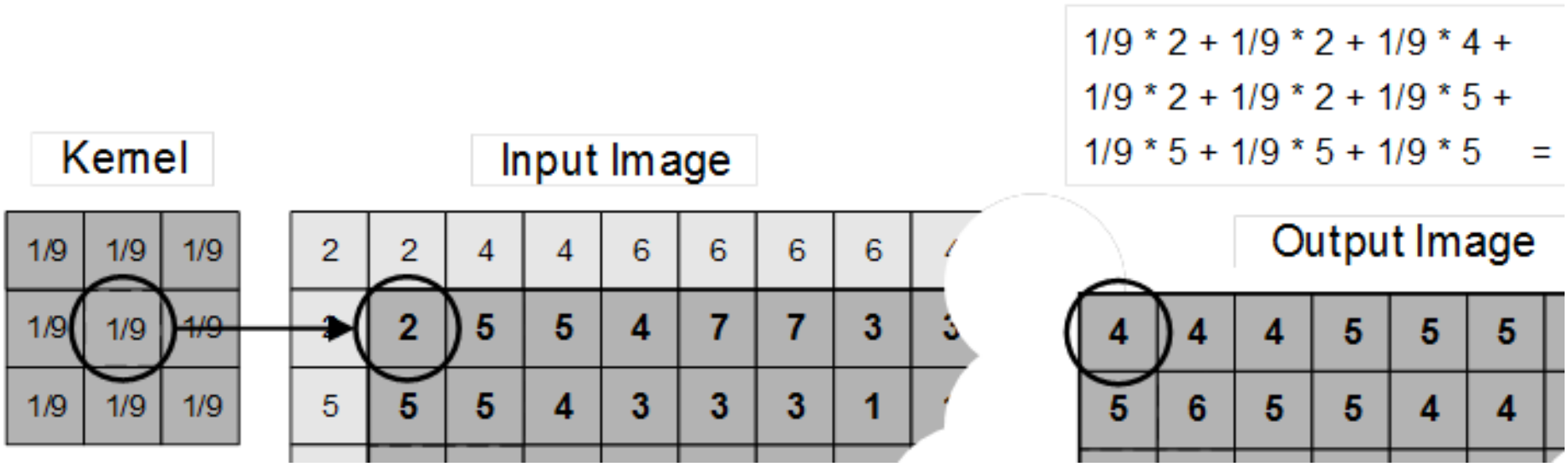


# Convolution



# Mean Filter

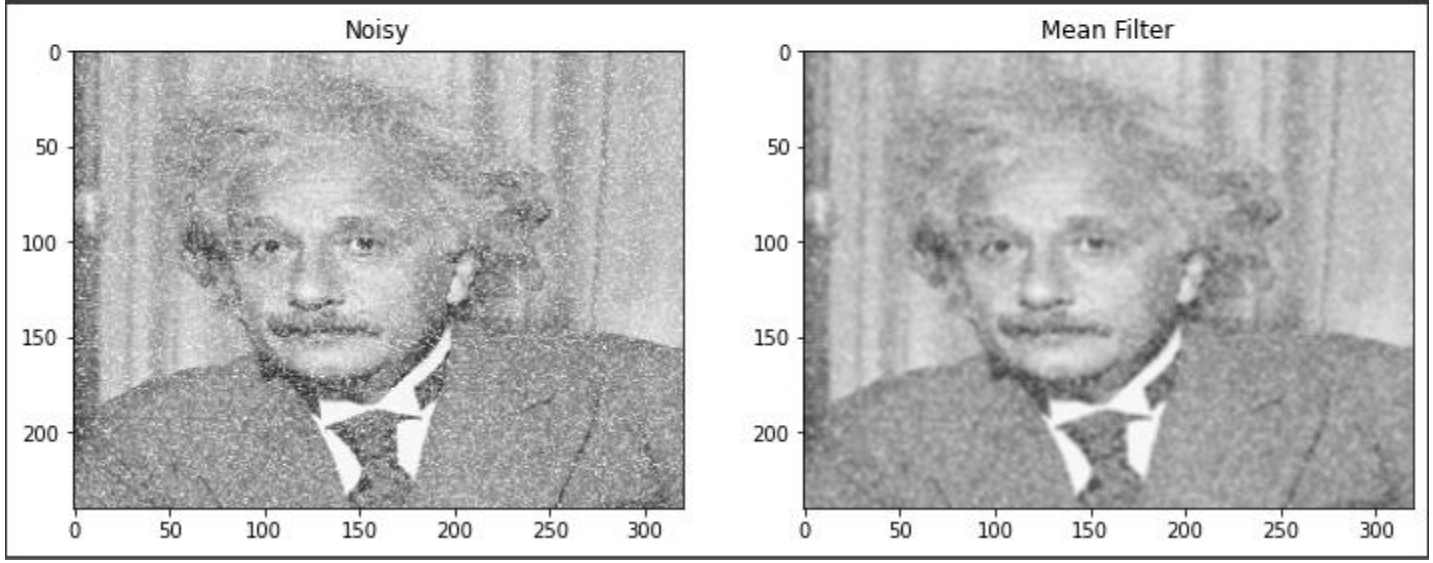
- Replaces the center pixel with the mean of its neighborhood
- Spreads the outlier value to its neighbors
- Details are smoothed



# Mean Filter

- Replaces the center pixel with the mean of its neighborhood
- Spreads the outlier value to its neighbors
- Details are smoothed

```
kernel_size = (3,3)
im_blur = cv2.blur(im_einstein,kernel_size)
plot_sidebyside([im_einstein,im_blur],['Noisy','Mean Filter'],colormap='gray')
```



# Median Filter

- Replaces the center pixel with by a median of its neighborhood
- Preserves more details when compared to the mean filter

**Input**

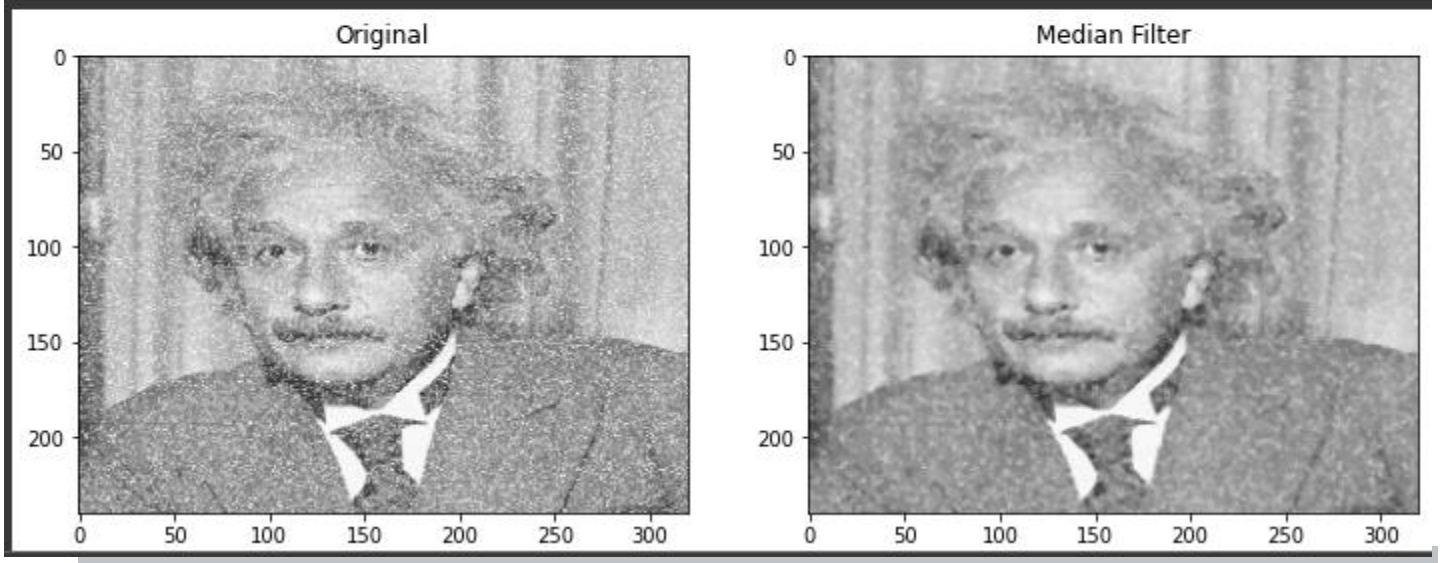
| | | | | | |
|---|---|---|---|---|---|
| 1 | 4 | 0 | 1 | 3 | 1 |
| 2 | 2 | 4 | 2 | 2 | 3 |
| 1 | 0 | 1 | 0 | 1 | 0 |
| 1 | 2 | 1 | 0 | 2 | 2 |
| 2 | 5 | 3 | 1 | 2 | 5 |
| 1 | 1 | 4 | 2 | 3 | 0 |

Sorted:0,0,1,1,**1**,2,2,4,4

**Output**

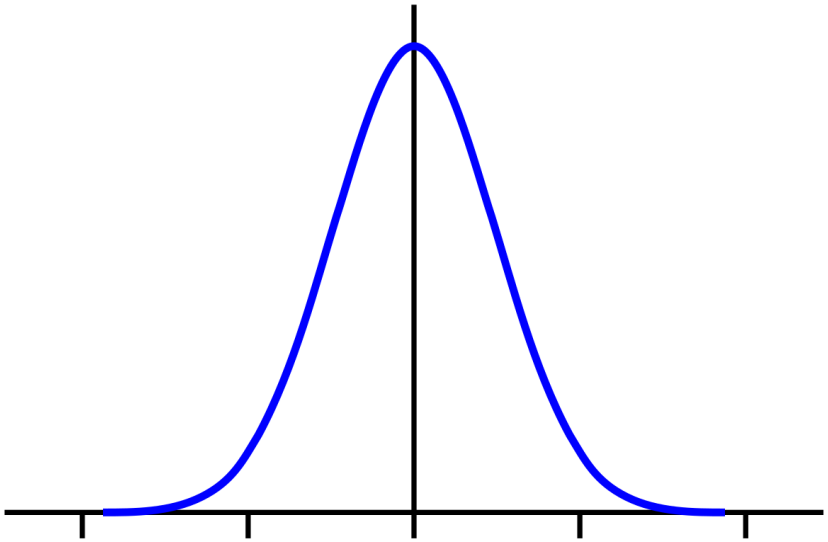
| | | | | | |
|---|---|---|---|---|---|
| 1 | 4 | 0 | 1 | 3 | 1 |
| 2 | 1 | 1 | 1 | 1 | 3 |
| 1 | 1 | 1 | 1 | 2 | 0 |
| 1 | 1 | 1 | 1 | 1 | 2 |
| 2 | 2 | 2 | 2 | 2 | 5 |
| 1 | 1 | 4 | 2 | 3 | 0 |

```
im_blur = cv2.medianBlur(im_einstein,3)
plot_sidebyside([im_einstein,im_blur],['Noisy','Median Filter'],colo
```



# Gaussian Filter

- Gaussian distribution of pixels
- The kernel is composed of probabilities
- Weighted Mean
- The standard deviation determines the blur degree



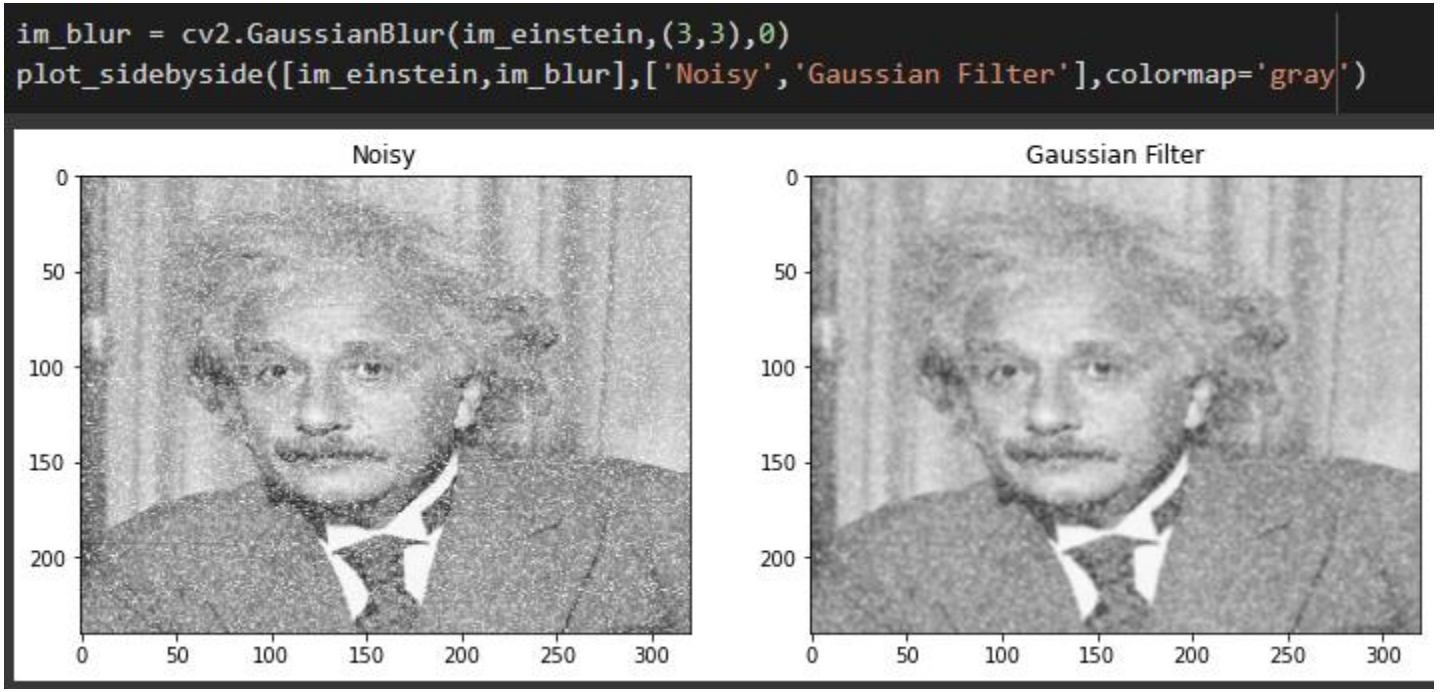
$$\frac{1}{16} \times \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

$$\frac{1}{273} \times \begin{bmatrix} 1 & 4 & 7 & 4 & 1 \\ 4 & 16 & 26 & 16 & 4 \\ 7 & 26 & 41 & 26 & 7 \\ 4 & 16 & 26 & 16 & 4 \\ 1 & 4 & 7 & 4 & 1 \end{bmatrix}$$

# Gaussian Filter

- Gaussian distribution of pixels
- The kernel is composed of probabilities
- Weighted Mean
- The standard deviation determines the blur degree

```
im_blur = cv2.GaussianBlur(im_einstein,(3,3),0)
plot_sidebyside([im_einstein,im_blur],['Noisy','Gaussian Filter'],colormap='gray')
```



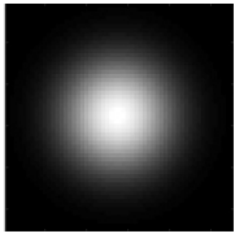
# Bilateral Filter

- **Gaussian Distribution based (spacial and pixel intensity)**
- **Add Normalization Factors and Range Weight**
- **Preserve details**

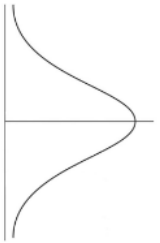
$$BF[I]_p = \frac{1}{W_p} \sum_{q \epsilon S} G_{\sigma_s}(||p-q||) G_{\sigma_r}(|I_p - I_q|) I_q$$

Normalization Factor

Space Weight



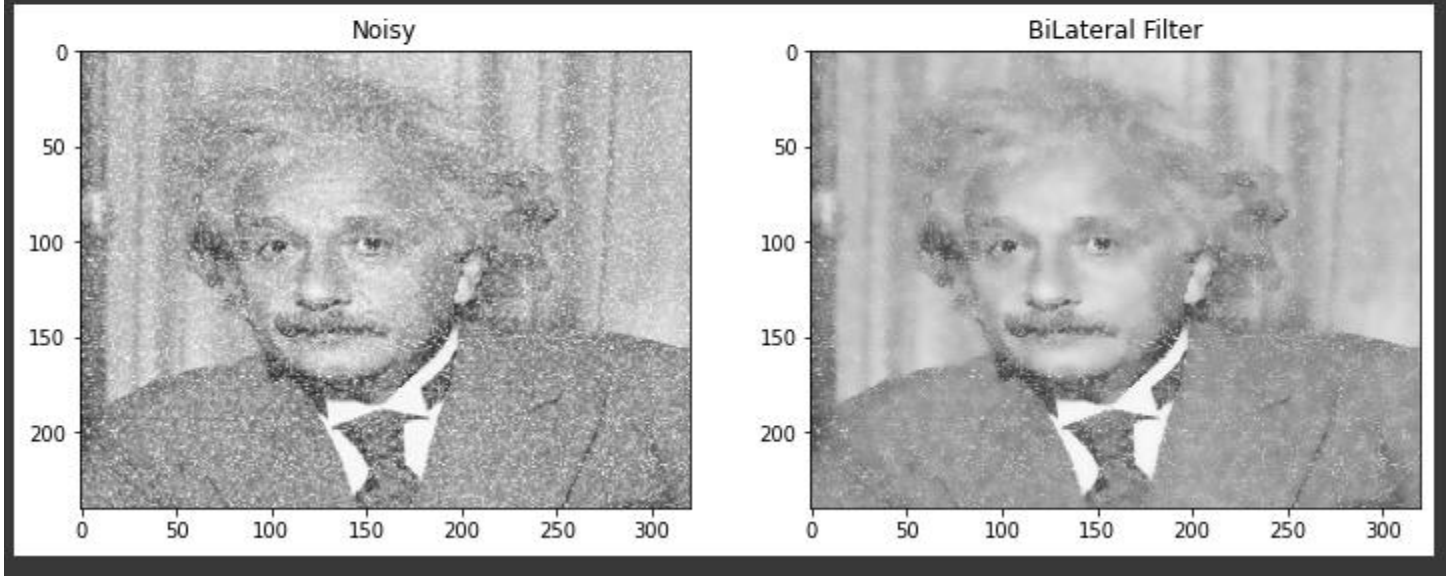
Range Weight



# Bilateral Filter

- Gaussian Distribution based (spacial and pixel intensity)
- Add Normalization Factors and Range Weight
- Preserve details

```
# Apply bilateral filter with d = 15,
# sigmaColor = sigmaSpace = 75.
im_blur = cv2.bilateralFilter(im_einstein,9,75,75)
plot_sidebyside([im_einstein,im_blur],['Noisy','BiLateral Filter'],colormap='gray')
```



# Practice

[Link: Practice 02]