Optimization for Data Science
Lecture 02: Image Denoising

Kimon Fountoulakis

School of Computer Science
University of Waterloo

12/09/2019

Last time: preliminaries

- We reviewed basic linear algebra and calculus concepts for Euclidean spaces.
- We will use these for developing and analyzing numerical optimization algorithms
- Please make sure that you are familiar with these concepts. Ask questions on piazza or during the class if something is not clear.

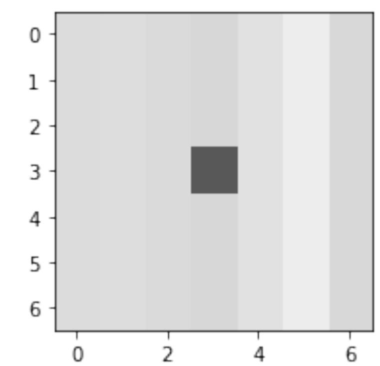
Outline

- Images as matrices
- How to measure horizontal and vertical differences among pixels
- Some basic complex calculus
- Denoising as an optimization problem

Images as matrices

- We are given an n x n image.

7x7 image



- We can think of the image as a n x n matrix

```
[[21., 20., 22., 24., 18., 11., 23.],
 [21., 20., 22., 24., 18., 11., 23.],
 [21., 20., 22., 24., 18., 11., 23.],
 [21., 20., 22., 99., 18., 11., 23.],
 [21., 20., 22., 24., 18., 11., 23.],
 [21., 20., 22., 24., 18., 11., 23.],
 [21., 20., 22., 24., 18., 11., 23.]]
```

- Which can be reshaped (vectorized) into an n^2 vector

```
array([[21.],
       [21.],
       [21.],
       [21.],
       [21.],
       [21.],
       [21.],
       [20.],
       [20.],
       [20.],
       [20.],
       [20.],
       [20.],
       [20.],
         ⋮
```

First column of matrix

Second column of matrix

Source of dataset: https://archive.ics.uci.edu/ml/datasets/Pen-Based+Recognition+of+Handwritten+Digits

Generate Gaussian noise

- We generate a random n x n matrix called “noise”.

```
[[-3.4  -1.83 -0.91  4.29  5.53 -0.55  3.36]
 [-4.94  1.7  -2.43  1.68  2.75 -2.21 -3.37]
 [ 0.14  4.4  -1.5  -9.52 -1.21  3.39 -2.08]
 [ 2.44  0.93 -0.82 -3.    4.49 -2.68  0.72]
 [-0.23 -0.98 -7.04 -4.57 -0.06 -1.51  6.39]
 [-4.14 -4.36 -1.22 -2.09  2.44  2.92  2.56]
 [ 0.3   0.12  5.11 -0.85  1.14  2.67  4.  ]]
```

- Each component of the matrix is a random Gaussian variable with mean equal to zero and some given standard deviation. For example, for our digit image in Python 3 we do

```
mean_ = 0
standard_deviation = 3
dimensions = (7,7)

noise = np.random.normal(mean_,standard_deviation,dimensions)

print(np.round(noise,2))
```

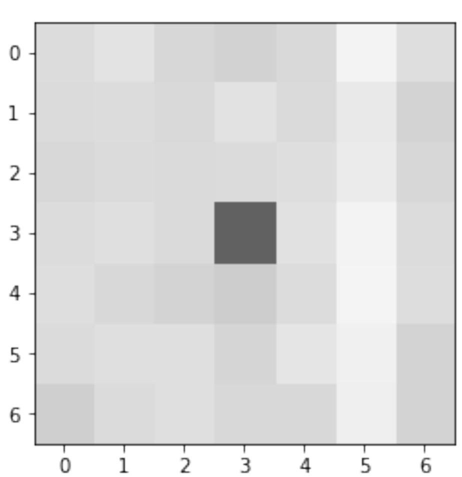
Adding noise to images

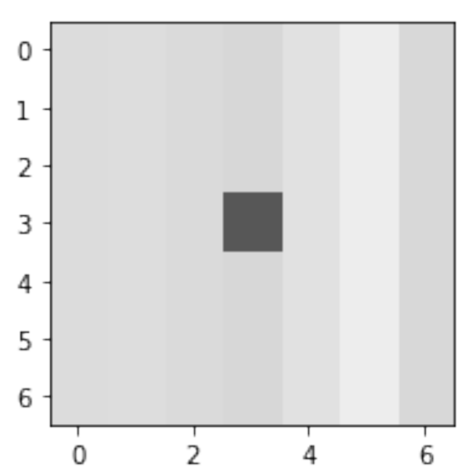
- Let us denote the n x n matrix of our image by “z_clean”
- We can add Gaussian noise to the image by

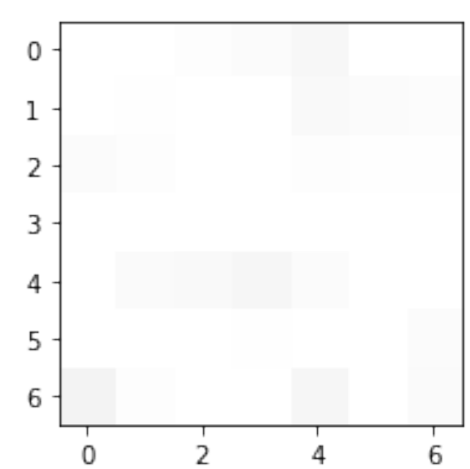
```
[[21., 20., 22., 24., 18., 11., 23.],
 [21., 20., 22., 24., 18., 11., 23.],
 [21., 20., 22., 24., 18., 11., 23.],
 [21., 20., 22., 99., 18., 11., 23.],
 [21., 20., 22., 24., 18., 11., 23.],
 [21., 20., 22., 24., 18., 11., 23.],
 [21., 20., 22., 24., 18., 11., 23.]]
```

```
[[-3.4   -1.83 -0.91  4.29  5.53 -0.55  3.36]
 [-4.94  1.7  -2.43  1.68  2.75 -2.21 -3.37]
 [ 0.14  4.4  -1.5  -9.52 -1.21  3.39 -2.08]
 [ 2.44  0.93 -0.82 -3.    4.49 -2.68  0.72]
 [-0.23 -0.98 -7.04 -4.57 -0.06 -1.51  6.39]
 [-4.14 -4.36 -1.22 -2.09  2.44  2.92  2.56]
 [ 0.3   0.12  5.11 -0.85  1.14  2.67  4.  ]]
```

z_noisy = z_clean + noise





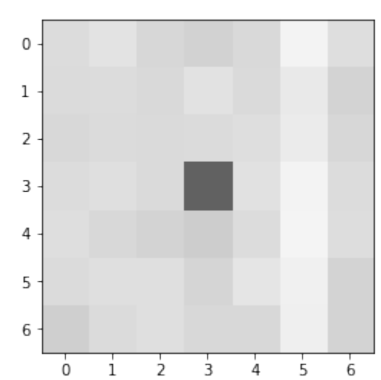


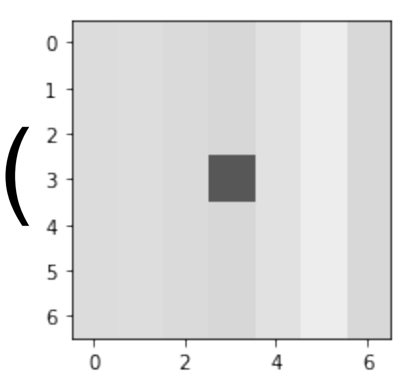
Measuring distance

- Let us denote with z_noisy and z_clean the vectorized noisy and clean images, respectively.

- We measure the distance between the two vectors using the Euclidean norm

$$\|z_{\text{noisy}} - z_{\text{clean}}\|_2$$

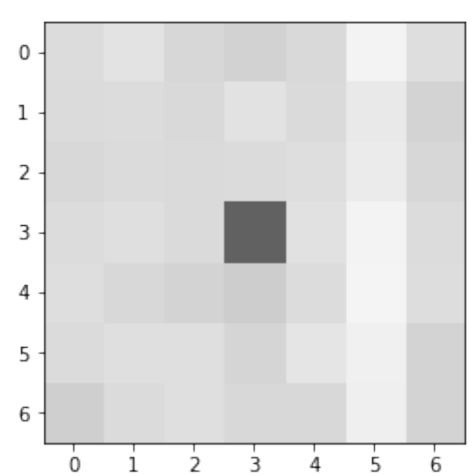
array()

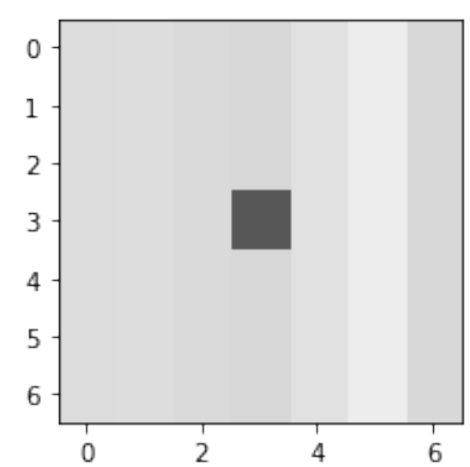
array()

Measuring distance

- Most of the time though we will use the square of the distance

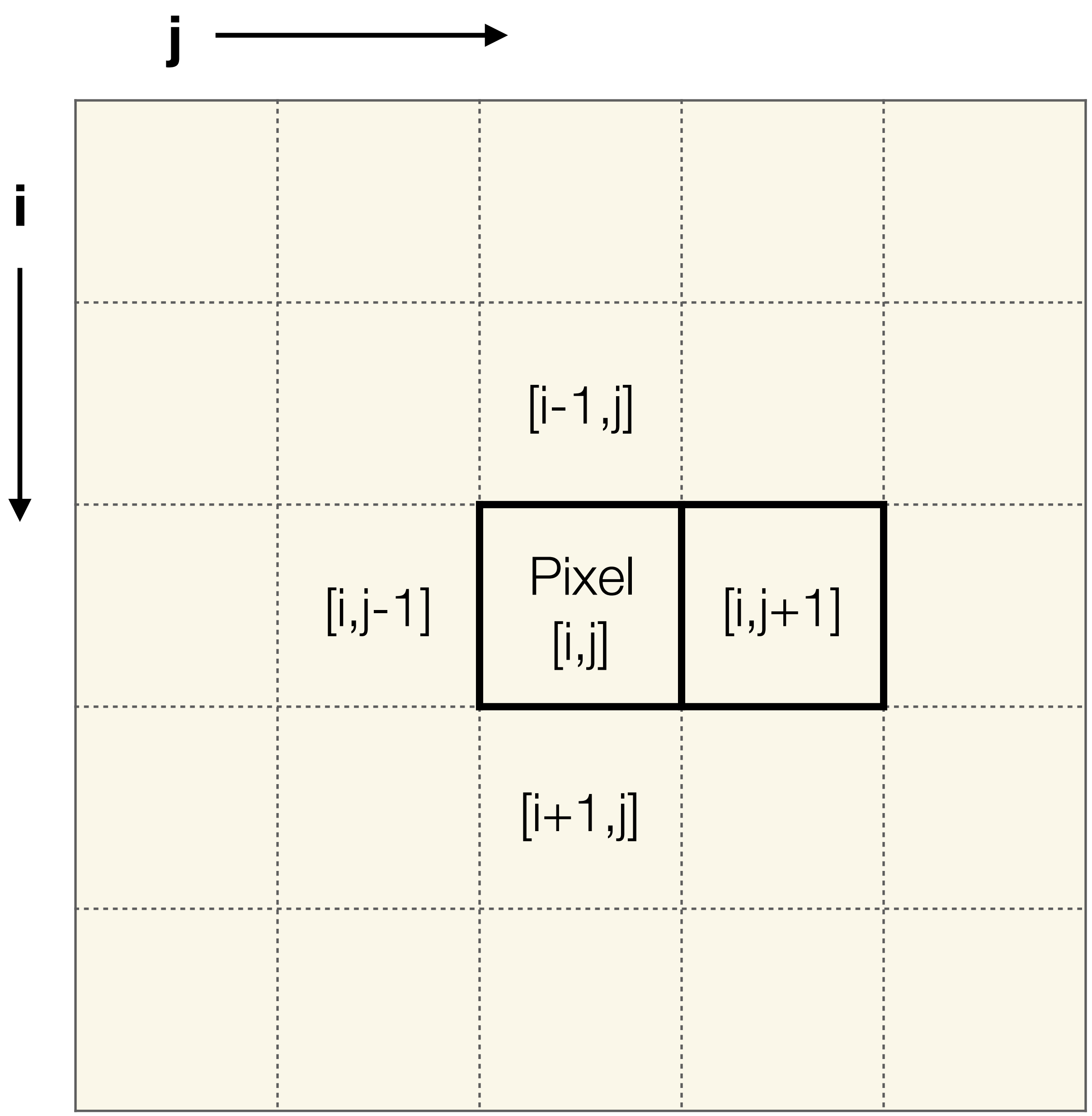
$$\|z_{\text{noisy}} - z_{\text{clean}}\|_2^2$$

array(

) array(

)

Horizontal forward differences



dx[i,j] = pixel [i,j+1] - pixel [i,j]

Horizontal forward differences operator: two new matrices

$$
\begin{array}{cc}
\begin{bmatrix}
-1 & 1 & & & & \\
 & -1 & 1 & & & \\
 & & -1 & 1 & & \\
 & & & . & . & \\
 & & & & . & . \\
 & & & & & .
\end{bmatrix}
&
\begin{bmatrix}
1 & & & & & \\
 & 1 & & & & \\
 & & 1 & & & \\
 & & & . & & \\
 & & & & . & \\
 & & & & & .
\end{bmatrix}
\\
\textbf{n x n matrix} & \textbf{n x n matrix} \\
J & I
\end{array}
$$

Horizontal forward differences operator as a matrix

$$D_h = \underbrace{J \qquad \otimes \qquad I}_{}$$

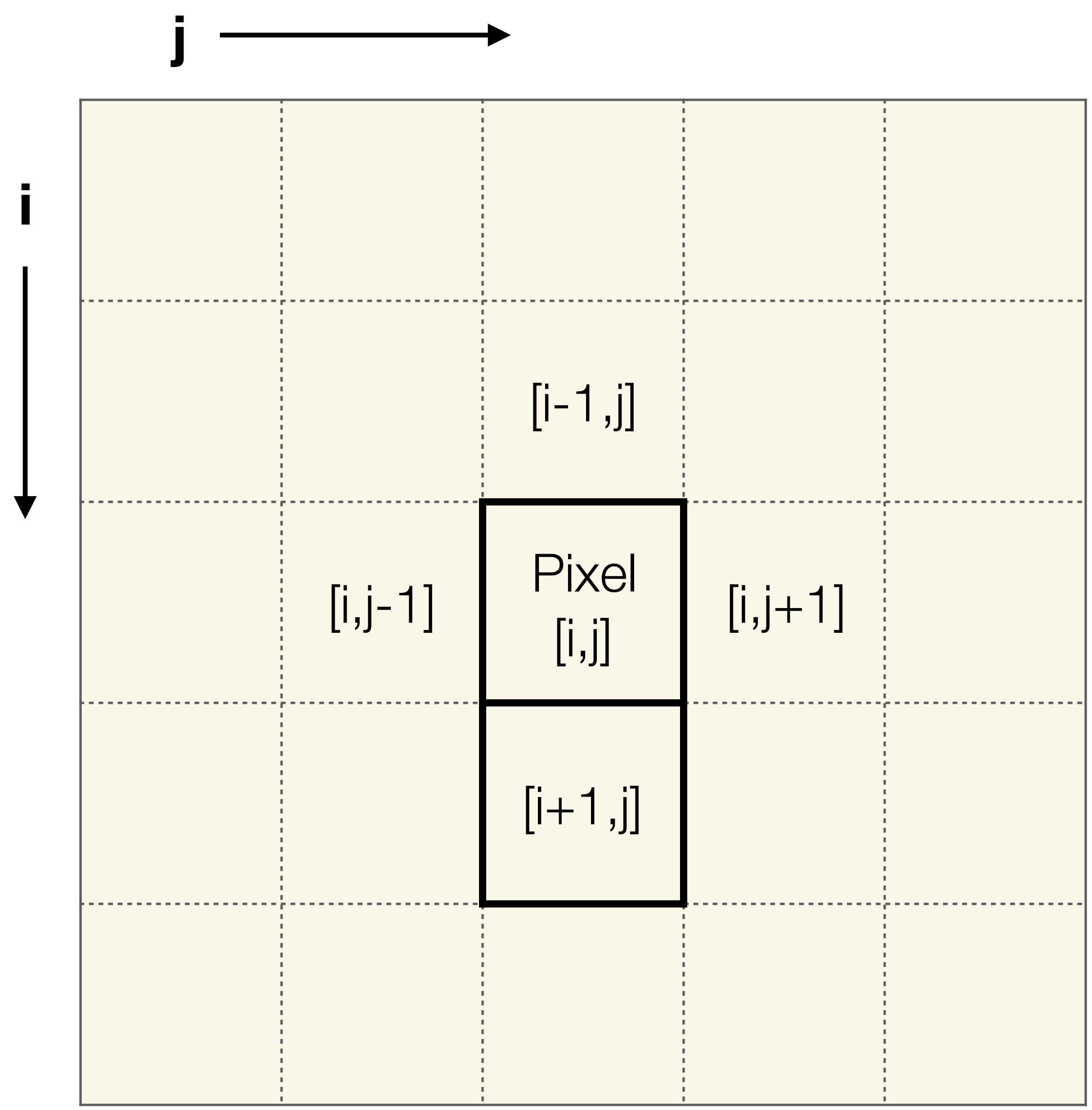
Big sparse matrix
n^2 x n^2

$$\begin{bmatrix} J[1,1]\cdot I & J[1,2]\cdot I & & & \\ & J[2,2]\cdot I & J[2,3]\cdot I & & \\ & & J[3,3]\cdot I & J[3,4]\cdot I & \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ & & & & J[n,n]\cdot I \end{bmatrix}$$

=

$$\begin{bmatrix} -I & I & & & \\ & -I & I & & \\ & & -I & I & \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ & & & & -I \end{bmatrix}$$

Vertical forward differences



dx[i,j] = pixel [i+1,j] - pixel [i,j]

Vertical forward differences operator as a matrix

$$D_v = \underbrace{I \otimes J}$$

Big sparse matrix n^2 x n^2

$$\begin{bmatrix} I[1,1]\cdot J & I[1,2]\cdot J & & & \\ & I[2,2]\cdot J & I[2,3]\cdot J & & \\ & & I[3,3]\cdot J & I[3,4]\cdot J & \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ & & & & I[n,n]\cdot J \end{bmatrix}$$

=

$$\begin{bmatrix} J & & & \\ & J & & \\ & & J & \\ \cdots & \cdots & \cdots & \cdots \\ & & & J \end{bmatrix}$$

Measuring horizontal and vertical differences

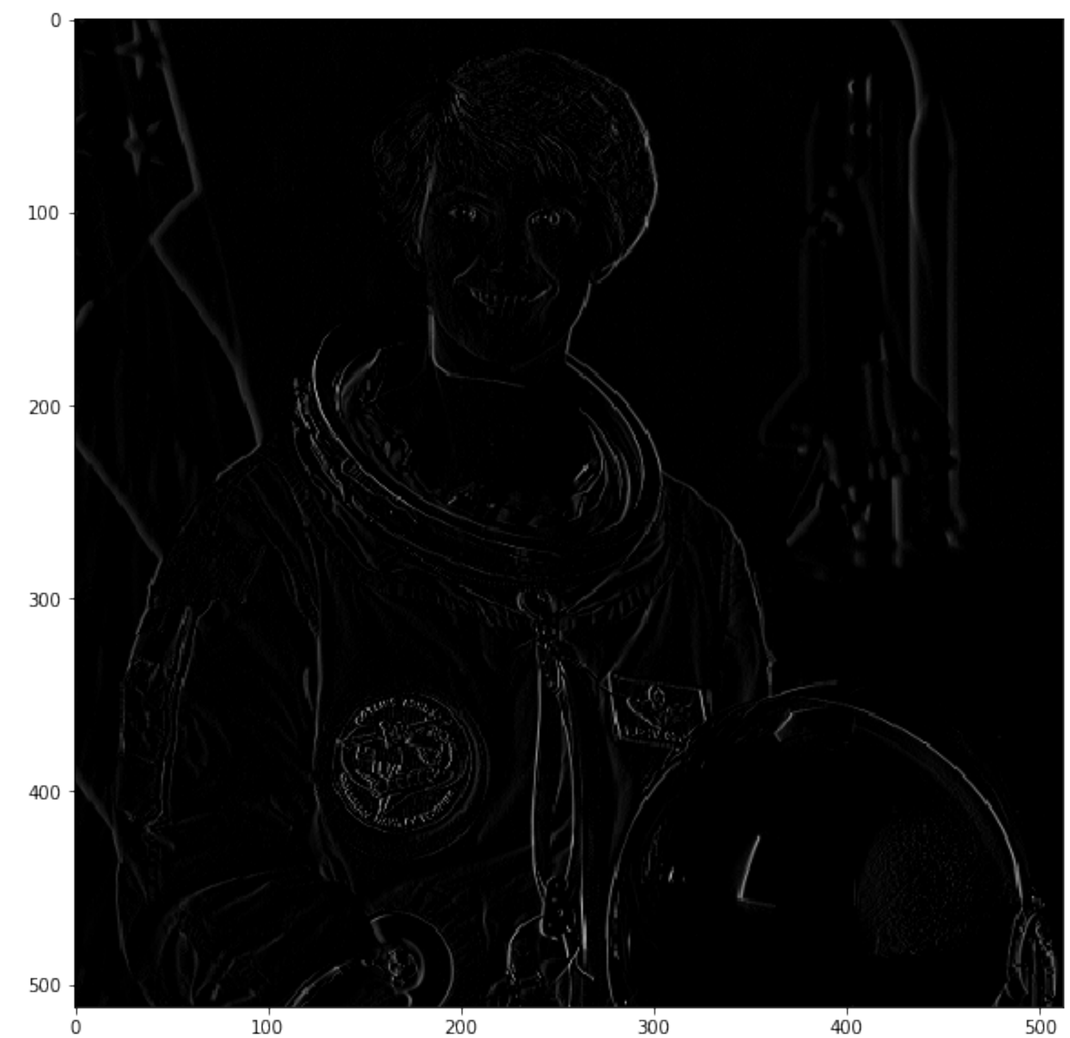
- Let "x" be the vectorized version of the image. The forward horizontal and vertical differences of the image is defined as

$$D_h x \text{ and } D_v x$$

matrix-vector product

Real example

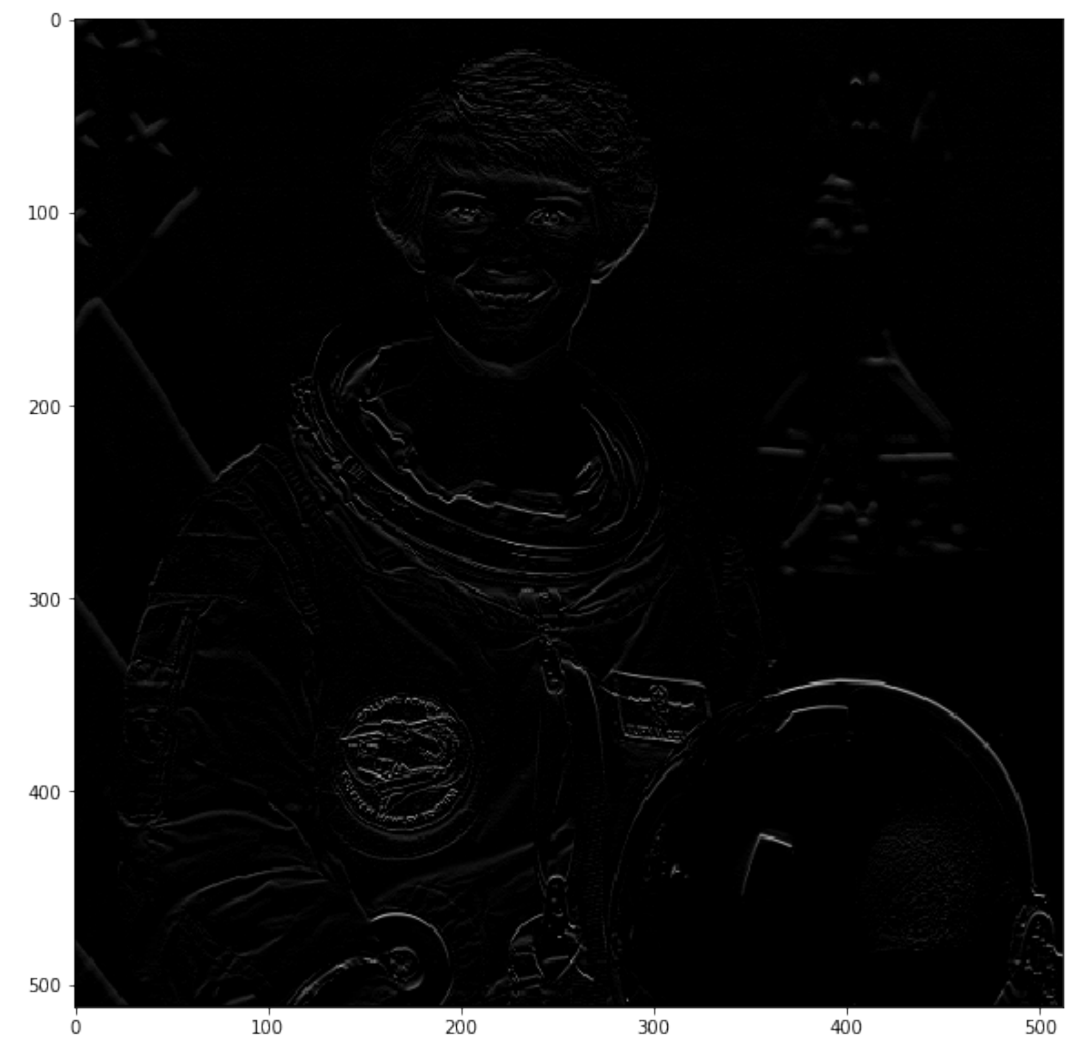
Original



Horizontal forward differences

Real example

Original



Vertical forward differences

Absolute value of complex numbers

$$|x + i \cdot y| = \left(x^2 + y^2\right)^{1/2}$$

Real part

Imaginary part

Euclidean norm for vectors of complex numbers

$$\|x\|_2 = \left(\sum_{i=1}^{n} |x_i|^2\right)^{1/2} = \left(\sum_{i=1}^{n} Re(x_i)^2 + Im(x_i)^2\right)^{1/2}$$

$$x_i = Re(x_i) + i \cdot Im(x_i)$$

Complex differences operator

$$D = D_h + iD_v$$

Complex matrix

Horizontal differences Operator

Vertical differences Operator

The optimization problem

- Let “z_noisy” be a vectorized version of the given image. Then solve the following optimization problem to find the denoised image “x”

$$\text{minimize } \frac{\lambda}{2}\|Dx\|_2^2 + \frac{1}{2}\|x - z_{noisy}\|_2^2$$

- The output “x” is in vectorized format. Remember to reshape “x” to obtain a 2D image.

Parameter tuning

$$\text{minimize } \frac{\lambda}{2}\|Dx\|_2^2 + \frac{1}{2}\|x - z_{noisy}\|_2^2$$

- User defined parameter that controls the balance between regularization and fitting to the noisy image.

$\lambda = 0.1$

$\lambda = 3$

$\lambda = 10$

Optimality conditions for the Denoising problem

$$\left(\lambda \mathrm{real}(D^* D) + I\right) x = z_{noisy}$$

- where real(D* D) is the real part of the matrix-matrix product D* D:

$$\mathrm{real}(D^* D) = D_h^T D_h + D_v^T D_v$$

- and I is the identity matrix of dimension $n^2 \times n^2$

Optimality conditions for the Denoising problem

$$\left(\lambda \mathrm{real}(D^* D) + I\right) x = z_{noisy}$$

- By solving the above linear system we obtain the vectorized denoised image x. Remember to reshape “x” to obtain a 2D image.