# Intelligent Analysis of Biomedical Images HW1

lachin.naghashyar - 98110179

October 2023

## Q1.1

$$\text{If } y[n] = x[n] * h[n], \text{ then } y[n-1] = x[n-1] * h[n-1]?$$

## S1.1

Given the convolution:

$$y[n] = x[n] * h[n]$$

This means that $y[n]$ is the convolution of $x[n]$ and $h[n]$. The convolution is defined as:

$$y[n] = \sum_{k=-\infty}^{\infty} x[k] \cdot h[n-k]$$

For the value $y[n-1]$ based on the given convolution relationship:

$$y[n-1] = \sum_{k=-\infty}^{\infty} x[k] \cdot h[n-1-k]$$

This is not equal to $x[n-1] * h[n-1]$ since $x[n-1]$ and $h[n-1]$ are just single values at n-1, not entire sequences.

## Q1.2

$$\text{If } y(t) = x(t) * h(t), \text{ then } y(-t) = x(-t) * h(-t)$$

## S1.2

Given

$$y(t) = x(t) * h(t)$$

the convolution integral for continuous-time signals is:

$$(x * h)(t) = \int_{-\infty}^{\infty} x(\tau) \cdot h(t-\tau)\, d\tau$$

Evaluating at $-t$:

$$y(-t) = \int_{-\infty}^{\infty} x(\tau) \cdot h(-t-\tau)\, d\tau$$

This is not the same as $x(-t) * h(-t)$. The expression $x(-t) * h(-t)$ represents the convolution of the time-reversed versions of $x(t)$ and $h(t)$. It's a different operation than simply evaluating $y(t)$ at $-t$.

## Q2

Given two LTI systems. $S_1$ has input $x$ and output $w$. $S_2$ has input $w$ and output $y$. The systems are causal and described by the following difference equations:

$$S_1 : w[n] = \frac{1}{2}w[n-1] + x[n]$$
$$S_2 : y[n] = \alpha y[n-1] + \beta w[n]$$

Given:

$$y[n] = \left(-\frac{1}{8}\right) y[n-2] + \frac{3}{4}y[n-1] + x[n]$$

Find the values of $\alpha$ and $\beta$.

## S2.1

The impulse response of a cascade connection of two systems is given by the convolution of their individual impulse responses. Let the impulse response of system $S_1$ be $h_1[n]$ and that of system $S_2$ be $h_2[n]$.

The impulse response $h[n]$ of the cascade connection is:

$$h[n] = h_1[n] * h_2[n]$$

which is computed using the convolution sum:

$$h[n] = \sum_{k=-\infty}^{\infty} h_1[k] \cdot h_2[n-k]$$

Given the previously computed values of $h_1[n]$ and $h_2[n]$:

$$h_1[0] = 1, \quad h_1[1] = 0.5, \quad h_1[2] = 0.25 \quad (\text{and} \quad h_1[n] = 0 \text{ for } n > 2)$$
$$h_2[0] = 1, \quad h_2[1] = 0.25, \quad h_2[2] = 0.0625 \quad (\text{with values halving for } n > 2)$$

Using the convolution sum, we find the impulse response $h[n]$ for the cascade connection:

$$h[0] = 1, \quad h[1] = 0.75, \quad h[2] = 0.4375, \quad h[3] = 0.09375, \quad h[4] = 0.015625$$

These values represent the impulse response for the cascade connection of the systems $S_1$ and $S_2$. Subsequent values of $h[n]$ will quickly approach zero.

## S2.2

The impulse response of the cascade connection of two systems is the convolution of their individual impulse responses. Given the impulse responses of systems $S_1$ and $S_2$:

$$h_1[0] = 1, \quad h_1[1] = 0.5, \quad h_1[2] = 0.25 \quad (h_1[n] = 0 \text{ for } n > 2)$$
$$h_2[0] = 1, \quad h_2[1] = 0.25, \quad h_2[2] = 0.0625 \quad (h_2[n] \text{ values halve for } n > 2)$$

The impulse response $h[n]$ of the cascade connection can be computed as:

$$h[n] = \sum_{k=-\infty}^{\infty} h_1[k] \cdot h_2[n-k]$$

Computing the convolution for the first few terms, we get:

$$\begin{aligned} h[0] &= 1 \\ h[1] &= 0.75 \\ h[2] &= 0.4375 \\ h[3] &= 0.09375 \\ h[4] &= 0.015625 \end{aligned}$$

## S4.1

Given the convolution:

$$(x * y)[m,n] = \sum_{m'=0}^{N-1} \sum_{n'=0}^{N-1} x[m',n'] \cdot y[m-m', n-n']$$

For each fixed pair $(m,n)$:

1. You iterate over all $m'$ values from 0 to $N-1$. This requires $N$ operations. 2. For each $m'$, you iterate over all $n'$ values from 0 to $N-1$. This requires $N$ operations. 3. Thus, for each $m'$, the inner summation requires $N$ operations. Since there are $N$ possible values of $m'$, the total operations for a fixed $(m,n)$ is $N^2$.

Now, you have to perform this for every possible pair $(m,n)$, and since there are $N \times N$ such pairs, the total number of operations becomes $(N \times N) \times N^2 = N^4$.

So, the time order of this calculation is $O(N^4)$.

## 4.2

Yes, we can use the Convolution Theorem which states that convolution in the spatial domain is equivalent to multiplication in the frequency domain.

We can use this theorem as follows; 1. **Fourier Transform**: Compute the 2D Fourier Transform of both images $x$ and $y$ using the Fast Fourier Transform (FFT).( $O(N^2 \log N)$ ) 2. **Multiplication**: Multiply the two Fourier Transforms element-wise. ( $O(N^2)$ ) 3. **Inverse Fourier Transform**: Compute the Inverse 2D FFT of the product obtained in the previous step to get the convolution result.( $O(N^2 \log N)$ ).

The overall time complexity for the process is:

$$O(N^2 \log N) + O(N^2) + O(N^2 \log N) = O(N^2 \log N)$$

## 5.1

DICOM, standing for Digital Imaging and Communications in Medicine, is a pivotal standard in medical imaging. Here's a distilled overview of a DICOM file's characteristics:

1. **Patient Info**: Carries unique identifiers such as name, ID, birth date, and sex.
2. **Medical Context**: Outlines the study, modality (e.g., CT, MR), and specific medical procedure.
3. **Image Content**: Encompasses the actual image, its size, clarity, and visual settings.
4. **Technical Specs**: Includes details of image acquisition, timestamp of creation, and equipment specifics.
5. **Unique Identifiers**: Assigns a unique code to each image, study, or series.
6. **Annotations**: May contain text or visuals on the image for extra context or emphasis.

7. **Supplementary Data**: Can house additional data like ECG patterns or audio notes.

8. **Compression**: Employs data compression for efficient size management and transfer.

9. **Security Measures**: Ensures data confidentiality through anonymization and encryption protocols.

DICOM files are essential for medical imaging for these reasons:

1. **Consistency**: Ensures images look consistent across various devices and software.

2. **All-in-One**: Stores both the patient info and the image together.

3. **Details**: Contains info about the image's origin and method.

4. **Sharing**: Makes it easy to share images between different places.

5. **Notes**: Allows adding direct notes or marks to the image.

6. **Safety**: Keeps patient data secure and private.

7. **More Views**: Supports moving or 3D images, not just flat ones.

**Rescale Intercept** and **Rescale Slope** are tags in DICOM that help convert the raw pixel values in medical images to values that doctors find more useful.

**Rescale Slope** Multiplier for the raw pixel value, adjusting its scale.

**Rescale Intercept** Value added after scaling, shifting the result.

They help us to have:

1. **Uniform Images**: Ensures images from different sources show standardized data.

2. **Precise Readings**: Makes sure the image values are accurate for diagnosis.

3. **Adaptable Storage**: Allows detailed raw data storage, then converts it to doctor-friendly values.

4. **Easy Comparisons**: Helps in comparing or merging images from different machines.

## 5.2

Anonymizing DICOM files ensures patient privacy while retaining the clinical value of the data. Here's how it's typically done:

1. **Tag Removal**: Erase specific tags holding patient details such as name, ID, and birth date.

2. **Pseudonymization**: Swap identifiers with random codes, preserving data matching without exposing identities.

3. **Date Adjustments**: Modify dates by random intervals to maintain event sequences without revealing actual dates.

4. **Image Alterations**: Mask unique image features, e.g., tattoos.

5. **UID Changes**: Substitute original Unique Identifiers with random ones.

6. **Text Scrubbing**: Cleanse reports or annotations of any identifiable free text.

7. **Activity Logs**: Maintain records of the anonymization changes.

**Additional Considerations**:

- **Encryption**: Add an extra layer of security by locking the files.
- **Validation**: Post-anonymization, ensure compliance with DICOM standards and confirm the clinical data's integrity.

pydicom is a Python library tailored for DICOM file operations. It provides utilities for editing or removing sensitive tags for anonymization.

```
import pydicom

ds = pydicom.dcmread("file_path.dcm")  # Load DICOM
ds.PatientName = "Anonymous"            # Modify tags
ds.PatientID = ""
# ... other tags as needed
ds.save_as("anonymized_file_path.dcm")  # Save changes
```

**Key pydicom Modules**

1. pydicom.dataset: Represents the main structure holding DICOM tags and values.
2. pydicom.dcmread/dcmwrite: Functions for reading and writing DICOM files.
3. pydicom.tag: Manages DICOM tags.
4. pydicom.dataelem: Represents individual DICOM data elements.
5. pydicom.uid: Handles DICOM Unique Identifiers (UIDs).
6. pydicom.pixel_data_handlers: Manages pixel data formats.
7. pydicom.valuerep: Ensures proper DICOM value representations.

While pydicom offers a straightforward approach to basic anonymization, thorough protection may require combining it with other tools or logic.

In summary, pydicom simplifies DICOM operations in Python, but thorough anonymization requires careful consideration.