Lab 5: Convolution

10/9/2020

Roadmap

- Convolution: definition and interpretations
- Finite Dimensional Signals: boundary conditions and padding
- Exercises:
  - The Mechanics
    - How do different padding settings relate to each other
    - Varying the kernel size
  - Motivating Examples
    - Microscopy: the points spread function (PSF)
    - Convolutions in Vision

Linear Shift Invariant Systems

- *Linear* systems all obey the principles of homogeneity and superposition
  - Linear operators *respect certain relationships between input elements, meaning that those relationships are preserved in the output*
    - L(a*x) = a*L(x) → respects/preserves scaling
    - L(x + y) = L(x) + L(y) → respects/preserves addition
- *Shift Invariant* systems respect a different relationship between input elements...“shifting”
  - L(shift(x)) = shift(L(x))
  - What the heck is a shift? Shifts change the position of elements in the input vector
    - shift(x[i]) = x[i + n] for some fixed value of n
    - Shift with n = 2:

$$\text{shift}\left(x = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \right) = \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

What types of relationships to shifts preserve? Ordering. This is a reasonable property to preserve (features stay in the same positions relative to each other).
LSI systems are linear systems with one more defining property/constraint.

LSI Systems as Matrices

- Linear System → We can describe the transformation with a matrix
  - First column is output of system in response to first basis/impulse vector
  - Subsequent columns are outputs in response to the other basis vectors, but these inputs are by definition shifted versions of the same vector → for LSI systems we only need to test the output to one input.

$$L\left(\begin{pmatrix}1\\0\\0\\0\end{pmatrix}\right) = \begin{pmatrix}k1\\k2\\0\\0\end{pmatrix} \quad \Longrightarrow \quad M = \begin{pmatrix}k1 & 0 & 0 & 0\\ k2 & k1 & 0 & 0\\ 0 & k2 & k1 & 0\\ 0 & 0 & k2 & k1\end{pmatrix}$$

We can take our standard “kick the tires”/test with impulse approach to building the matrix, but shift invariance means we only have to kick one tire.

LSI Systems as Matrices

- LSI system/convolution matrices also have a nice interpretation from the "rows," perspective: the output at a given index is the inner product of part of the input with a reversed order copy of the kernel

$$\begin{pmatrix} \text{S\_1} \\ \text{S\_2} \\ \text{S\_3} \\ \text{s\_4} \end{pmatrix}$$

$$\begin{pmatrix} k1 & 0 & 0 & 0 \\ \boxed{k2 \;\; k1} & & 0 & 0 \\ 0 & \boxed{k2 \;\; k1} & & 0 \\ 0 & 0 & \boxed{k2 \;\; k1} & \\ 0 & 0 & 0 & k2 \end{pmatrix} \begin{pmatrix} \text{out\_1} \\ \text{out\_2} \\ \text{out\_3} \\ \text{out\_4} \\ \text{out\_5} \end{pmatrix}$$

- So, in the usual way that we use the dot product to be a similarity measure, we can say that the output of an LSI (or equivalently, *the convolution of the input signal with the characteristic kernel*) at a given position, is a measure of how similar part of the input is to the kernel
- This is the "sliding dot product," interpretation.

Explain format

LSI Systems as Matrices

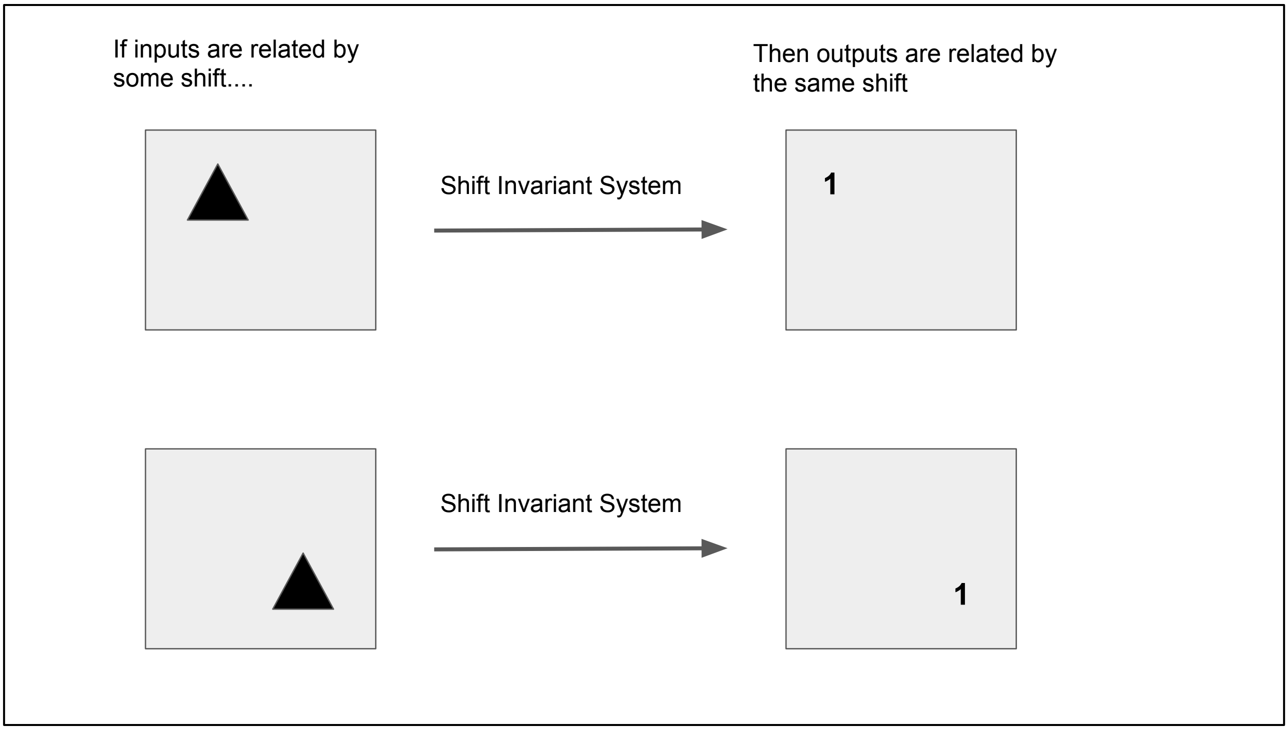
- Terminology:
  - The matrix that represents a LSI system is called a ***"convolution matrix."***
  - The response to the first impulse vector, or the first column of the convolution matrix, is called the ***"impulse response,"*** or ***"kernel,"*** of the operator.
  - We will often use the phraseology: "the output of an LSI system is the convolution of the input signal with the kernel."

Why Bother?

- When there is meaningful structure *between elements of the input signal*, describing the input-output function as an LSI system (or alternatively, describing the output as *the convolution of the input with some fixed kernel*) ensures that same structure is preserved in the output signal.
  - What kinds of signals have the type of structure that convolution respects?
    - Time-series data, images, time-series of images, etc.
- Practical Benefits
  - Reduced degrees of freedom (overfitting/model complexity reduction)

$$M = \begin{pmatrix} k1 & 0 & 0 & 0 \\ k2 & k1 & 0 & 0 \\ 0 & k2 & k1 & 0 \\ 0 & 0 & k2 & k1 \end{pmatrix}$$

Example of (nonlinear) space-shift invariant system: function that takes in an image and outputs object identity as a function of space
Time shift invariance: when you choose to measure a system does not affect its output.

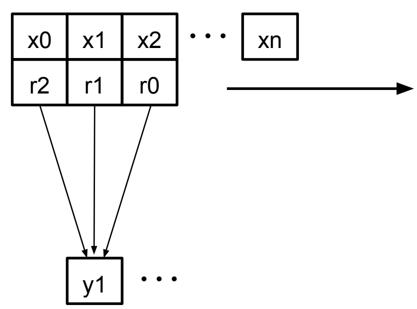

If inputs are related by some shift....
Then outputs are related by the same shift
Shift Invariant System
1
Shift Invariant System
1

Finite Dimensional Signals: Boundary Conditions

- Recall our definition of shift invariance
  - shift(x[i]) = x[i + n] for some fixed value of n
  - Or in general for finite LSI systems the definition of convolution is
  - $y[n] = \sum_m x[m]r[n-m] = \sum_k x[n-k]r[k]$ What happens when k > n?
- For finite dimensional inputs, this definition cannot make sense everywhere…
  - Defining "shift invariance," in these trouble spots ⇔ deciding on a set of boundary conditions
  - 3 most common are
    - Zero Padding ('same', 'valid', 'all')
    - Circular Boundary Conditions
    - Mirrored Boundary Conditions

Zero Padding: ‘full’, ‘same’, and ‘valid’

- ‘valid’ option: only consider the outputs where all terms in definition are known

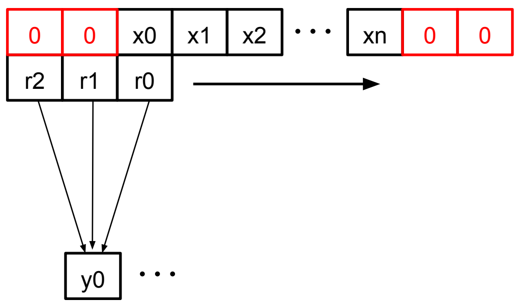


- Only consider outputs where the kernel is entirely within the signal

To start we lean on the “sliding dot product interpretation,” and define valid as the condition without any zero padding. That is, the outputs only exist where the dot product makes sense.

Zero Padding: ‘full’, ‘same’, and ‘valid’

- ‘full’ option: use every possible datapoint, assuming the unknown values are zero.

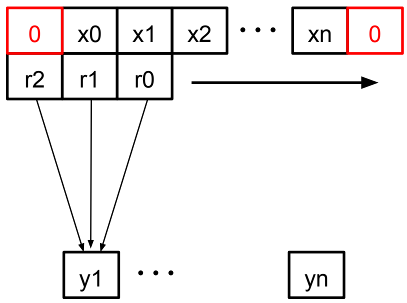


$$\sum_k x[n-k]r[k]$$

Assume x[-1] = x[-2] = 0.

Zero Padding: ‘full’, ‘same’, and ‘valid’

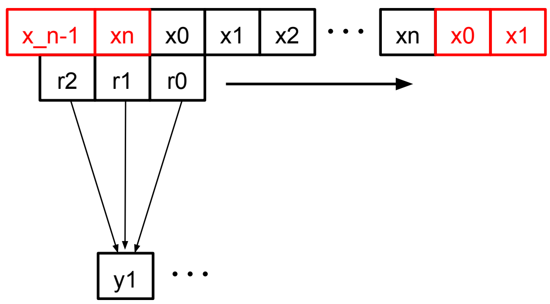
- ‘same’ option: zero pad on each side by an amount that makes the output size equal to the input size.



- It is often convenient for either theoretical or practical reason for the output to have the same dimensionality as the input.

Circular/Periodic Boundary Conditions

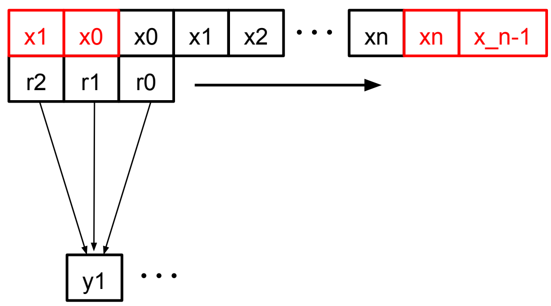
- Rather than assume unknown values are zero, assume values “wrap around”



- Similar to full but with different assumption about missing values

Mirrored Boundary Conditions

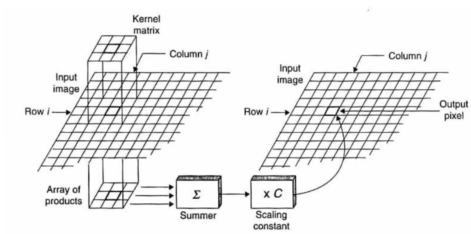
- Padded values are the “reflection,” or values near the boundary



- Similar to full but with different assumption about missing values

2-D Extensions

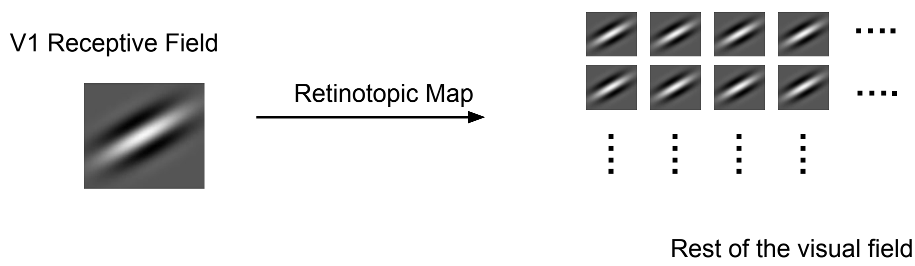
- Everything discussed so far has a direct extension to the two dimensional case:
    - Shift invariance: LSI systems are invariant to shifts in both directions (two dimensional data structures, i.e. images, can be shifted in two directions independently).
    - Padding: zeros surround data structure on both sides
    - Other boundary conditions: can be independently applied to each dimension
        - Visual: a 360 degree panoramic photo might be taken to have circular boundary conditions along one dimension (horizontal), but use zero padding along the other



Exercise Set 1:
The Mechanics of Convolution

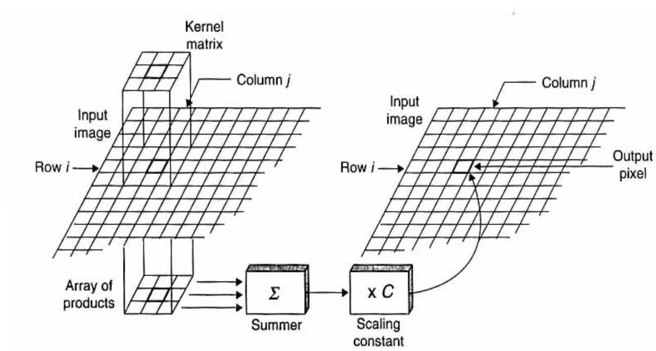
Motivating Example 1: Vision

- Convolutions play an important role in both biological and artificial vision



Motivating Example 1: Vision

- Convolution, especially via its role in convolutional neural networks, is the mathematical principle behind the recent rapid acceleration in computer vision.



Imagine you want to learn a linear transformation between 2 sets of images (i.e. find M s.t. y~=Mx) where y and x are both image data. In general the size of M is huge (bad), but restricting the transformation to be convolutional means the only values that need to be optimized are the elements of the kernel. This idea is called “weight sharing.”

Explain using mouse how 2-D convolution works using sliding dot product interpretation.

Motivating Example 1: Vision

- Convolution, especially via its role in convolutional neural networks, is the mathematical principle behind the recent rapid acceleration in computer vision.

$$M = \begin{pmatrix} k1 & 0 & 0 & 0 \\ k2 & k1 & 0 & 0 \\ 0 & k2 & k1 & 0 \\ 0 & 0 & k2 & k1 \end{pmatrix}$$

Imagine you want to learn a linear transformation between 2 sets of images (i.e. find M s.t. y~=Mx) where y and x are both image data. In general the size of M is huge (bad), but restricting the transformation to be convolutional means the only values that need to be optimized are the elements of the kernel. This idea is called “weight sharing.”

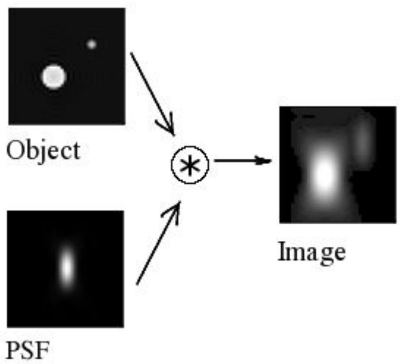
Look Back at convolution matrix examples, note how many values are restricted to be zero, this is what we save with convolution.

Exercise Set 1:
Feature Extraction

Motivating Example 2: Microscopy

- The output of an imaging system can be modeled as the convolution of an underlying signal (which is being measured/imaged) with a “point spread function,” which acts as a kernel.
- The point spread function determines the resolution of the imaging system, among other important properties

Motivating Example 2: Microscopy



- You may not always know ahead of time the value of a systems PSF.
- Can you think of a way, given a set of known inputs and their outputs from the imaging system, to estimate the PSF?
- Look at the example

PSF is the “blob” representing how the imaging system represents what should be a single point.

Exercise 3:
Determining the PSF

Conclusion

- LSI systems are uniquely characterized by their kernel/impulse response. The output of an LSI system is the convolution of the input with its kernel.
  - We need to make some assumptions about the input signal in order to carry out convolutions with finite dimensional signals. We think about the assumptions we make as defining the “boundary conditions.”
- LSI’s are a subset of linear systems with a special additional structure
  - Intuitively this additional structure can be useful when the output of the system is meant to “represent,” the input. Also, this structure naturally occurs in many settings (think time shift invariance for measurements).
- Besides everything discussed here, convolution has a deep connection to the Fourier transform, called the ‘convolution theorem,’ that is very powerful.