Convolution and Correlation

- Correlation
  - New signal by moving desired mask/template around.
  - New values follow the mask

- Convolution
  - Filter is the response to a unit bar (or delta function)
  - Convolution gives response to the entire signal

- Can switch from one to the other by flipping the filter
  - \( h_{\text{corr}}(x) = h_{\text{conv}}(-x) \)
  - \( h_{\text{corr}}(x, y) = h_{\text{conv}}(-x, -y) \)

- No difference if filter is symmetric
Image Filtering Preliminaries

- Denote the image by $F$ (to follow the book).

- Represent weights as a second image, $H$ (the kernel).

- Pretend that images are padded to infinity with zeros (so sums don’t need limits).

- To shift a function $f(x,y)$ up and to the right by $(a,b)$
  - $f(x-a, y-b)$
Correlation

- Denote by \( \circ \)

- Then the definition of discrete 2D correlation is:

\[
R_{i,j} = \sum_{u,v} H_{u-i,v-j} F_{u,v}
\]

Puts filter on \((i,j)\)
Correlation example one
Correlation example two
Correlation example three (switch H and F)
Correlation (H ◦ F versus F ◦ H)

Correlation is not always commutative
(example one is, example two is not)
Convolution

- Denote by $\otimes$
  - Others symbols include * (for 1D) and ** (for 2D).

- Matlab “conv” (1D) and “conv2” (2D)

- The definition of discrete 2D convolution is:

$$R_{i,j} = \sum_{u,v} H_{i-u, j-v} F_{u,v}$$

- Notice weird order of indices (includes the flips)
Convolution example
Properties of  \( R_{i,j} = \sum_{u,v} H_{i-u,j-v} F_{u,v} \)

- Commutative (unlike correlation)
  Compare \( H_{i-u} F_u \) and \( F_{i-u'} H_{u'} \) when subscripts of F are the same.
  In other words, \( u = i-u' \). Plug this into the first H, to get \( i-u = u' \).
  Note that the pairings are always the same. (QED)

- Associative (Can save CPU time!)
  \((A \otimes B) \otimes C = A \otimes (B \otimes C)\)

- Linear  \((aA + bB) \otimes C = a(A \otimes C) + b(B \otimes C)\)
  and
  \(C \otimes (aA + bB) = a(C \otimes A) + b(C \otimes B)\)
Properties of $R_{i,j} = \sum_{u,v} H_{i-u,j-v} F_{u,v}$

- Commutative (contrast with correlation)
- Associative (Can save CPU time!)
- Linear

- Output is a **shift-invariant** function of the input (i.e. shift the input image two pixels to the left, the output is shifted two pixels to the left)

- Converse of above is true: If a system is linear and shift invariant, then it is a convolution.
Shift invariant linear systems (§7.2)

- **Shift invariant**
  - Shift in the input means we simply shift the output
  - Example: Optical system response to a point of light
    - Light moves from center to edge, so does its image

- **Linear shift invariant**
  - Can compute the output due to complex input, based on the response to a single point input
    - Discrete version---function $\text{box}(x,y)$ is zero everywhere except at $(x',y')$ where is is 1.
    - Continuous version---delta function

- $f(x,y)$ is a linear combination of shifted versions of $\text{box}(x',y')$
Rewrite $f(i,j)$ as a sum over its natural basis

$$f(i, j) = \sum_u \sum_v \text{box}(i - u, j - v) f(u, v)$$

Example, if $f(i, j) = \begin{pmatrix} 2 & 3 \\ 4 & 5 \end{pmatrix}$

$$f(i, j) = 2 \cdot \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + 3 \cdot \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} + 4 \cdot \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} + 5 \cdot \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

Also, if $\text{box}(u, v) = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$

$$f(i, j) = 2 \cdot \text{box}(i - 0, j - 0) + 3 \cdot \text{box}(i - 0, j - 1) + 4 \cdot \text{box}(i - 1, j - 0) + 5 \cdot \text{box}(i - 1, j - 1)$$

$$f(i, j) = \sum_{u=0}^{1} \sum_{v=0}^{1} \text{box}(i - u, j - v) f(u, v)$$  
(reverse multiplication order to follow convention.)
Before we derive convolution ...

- Note that we are thinking of images as a linear combination of simple units (a “basis”).

- You have seen this before

\[ \mathbf{v} = (3, -2, 5) = 3 \cdot \mathbf{i} + (-2) \cdot \mathbf{j} + 5 \cdot \mathbf{k} \]

\[ e^x = 1 \cdot x^0 + 1 \cdot x^1 + \left( \frac{1}{2} \right) \cdot x^2 + \left( \frac{1}{2 \cdot 3} \right) \cdot x^3 + \ldots \]

- The notion of basis is an important abstraction because rewriting images (etc) with respect to different bases can provide insight and/or solves problems.
Another example

More useful basis for this data than the usual (1,0) and (0,1).
Rewrite \( f(i,j) \) as a sum over its natural basis

\[
f(i, j) = \sum_u \sum_v box(i-u, j-v) f(u, v)
\]

Box shifted by \((u,v)\). Note subtraction!

Given that

\[
\text{Response}(box(i,j)) = h(i,j)
\]

Shift invariance means that

\[
\text{Response}(box(i-u, j-v)) = h(i-u, j-v)
\]

Linearity means we can bring the response inside the sum.

\[
\text{Response}(f(i,j)) = R_{ij} = \sum_u \sum_v h(i-u, j-v) f(u, v)
\]

(Convolution by \( h \))
In more detail

\[
\text{response}\{f(i,j)\} = \text{response}\left\{\sum_u \sum_v \text{box}(i-u, j-v) \cdot f(u,v)\right\}
\]

\[
= \sum_u \sum_v \text{response}\{\text{box}(i-u, j-v) \cdot f(u,v)\}\quad \text{(linearity)}
\]

\[
= \sum_u \sum_v \text{response}\{\text{box}(i-u, j-v)\} \cdot f(u,v)\quad \text{(linearity)}
\]

\[
= \sum_u \sum_v h(i-u, j-v) \cdot f(u,v)\quad \text{(shift-invariant)}
\]

In the last step, we have used the fact that we can get the response to the shifted filter by shifting the response.

This derives convolution in terms of responses to a unit impulse function (here denoted by box()).
Response as sum of basis functions (§7.2)

- The response is linear combination of shifted versions of the kernel
- The weights are the values of the function being convolved
- The shifted versions of the kernels form a basis over which the result image is constructed
- Thinking of an image as a weighted sum over a basis is a generally useful idea—e.g., Fourier transforms.
Response as sum of basis functions (§7.2)

- Linear shift invariant systems explains the “flip” is in the previous formula
  - Shifting rewrote the function values so that the kernel was flipped
  - Convolution by h() implies a basis of shifted, flipped, h()
  - Getting the right answer requires flips if the kernel is not symmetric
Correlation

- Similar to convolution (no flips)
- Implements convolution (if a flip is used) or vice versa
- Finds things in images that “look like” the kernel
- The kernel is also referred to as a “mask”, especially in application oriented discussion (both in convolution and correlation).
2D convolution example (from MathWorks website)

For example, suppose the image is

\[
A = \begin{bmatrix}
17 & 24 &  1 &  8 & 15 \\
23 &  5 &  7 & 14 & 16 \\
 4 &  6 & 13 & 20 & 22 \\
10 & 12 & 19 & 21 &  3 \\
11 & 18 & 25 &  2 &  9 \\
\end{bmatrix}
\]

and the convolution kernel is

\[
h = \begin{bmatrix}
8 & 1 & 6 \\
3 & 5 & 7 \\
4 & 9 & 2 \\
\end{bmatrix}
\]

\[
R(1,1) = 5*17+3*24+1*23+8*5
\]

To do the complete convolution, set A and h as above in Matlab, and do conv2(A,h,’same’). Try also conv2(A,h) --- make sure you understand the difference!
Filters are templates

- Applying a filter at some **point** can be seen as taking a dot-product between the image and some vector
- Filtering the image yields a set of dot products

- Useful intuition
  - Filters look like the effects they are intended to find.
  - Filters find effects that look like them.
  - Remember to flip your filter if you are implementing correlation using convolution.

Filters for steps in X (left) and Y (right). The step in X goes from high-to-low. Convolving with it finds high-to-low steps due to the flip.
Normalized correlation

- Think of filters of a dot product
  
  - **problem**: brighter parts give bigger results even if the structure is same (often not what you want)
  
  - **normalized** correlation output is filter output, divided by root sum of squares of values over which filter lies
    \[
    \frac{h \cdot f}{|f|}
    \]
    (\(f\) is limited to where \(h\) is non zero)
  
  - Can think in terms of angle between vectors. Recall
    \[
    \cos(\theta) = \frac{h \cdot f}{|h||f|}
    \]
    (\(|h|\) is not relevant to this problem)
Normalized correlation

- Some tricks of the trade
  - Consider template filters that have zero response to a constant region (helps reduce response to irrelevant background).