EECS C145B / BioE C165 : Image Processing and Reconstruction Tomography

Lecture 2

Jonathan S. Maltz

jon@eecs.berkeley.edu http://muti.lbl.gov/145b
510-486-6744

Topics to be covered

1. Review of linear time/shift invariant systems
2. Review of discrete convolution in 1D
3. 1D convolution as a matrix operation
4. Convolution in 2D
5. 2D convolution as a matrix operation

Reading

1. Gonzalez and Woods pp. 116-123.

Additional reading

1. Oppenheim and Schafer pp. 8-29.
2. Jain pp. 13-15, 26, 49-54.

Linear shift invariant systems

Continuous space:

$$f(x) — \boxed{h(x)} — g(x)$$

$$g(x) = f(x) * h(x)$$

Discrete space:

$$f[n] — \boxed{h[n]} — g[n]$$

$$g[n] = f[n] * h[n]$$

Properties of Linear Shift Invariant (LSI) systems

1. The system is completely characterized by its **impulse response** $h(x)$.
2. Principle of superposition (POS) holds. Thus if:

$$f(x) = f_1(x) + f_2(x)$$

and we feed f_1 and f_2 through the system separately

$$\begin{aligned} g_1(x) &= h(x) * f_1(x) \\ g_2(x) &= h(x) * f_2(x) \end{aligned}$$

then

$$g(x) = h(x) * f(x) = g_1(x) + g_2(x).$$

3. The frequency components in the input signal are merely scaled by the system - no frequencies are present in the output that weren't present in the input.

Discrete convolution by superposition

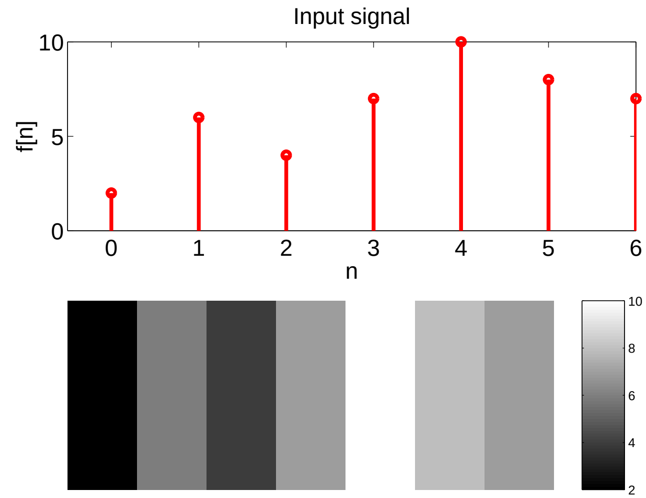
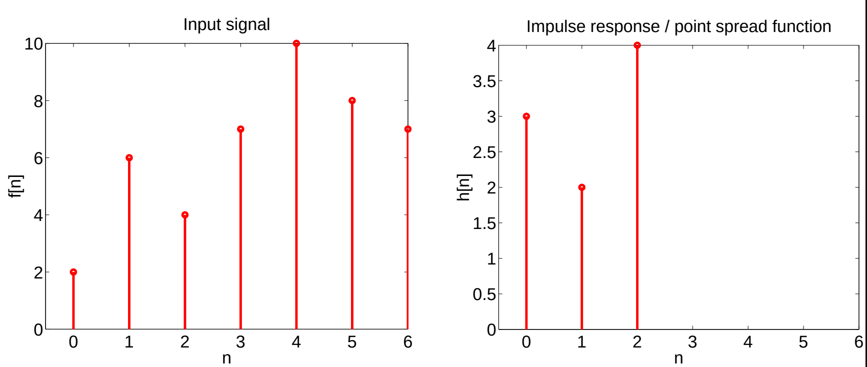
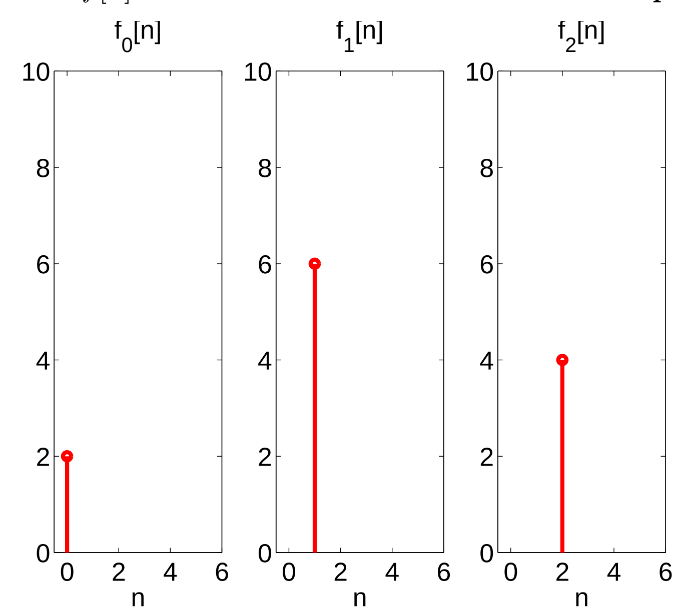


Image representation

Discrete convolution by superposition



Consider $f[n]$ as the sum of shifted and scaled impulses:



Convolution by superposition

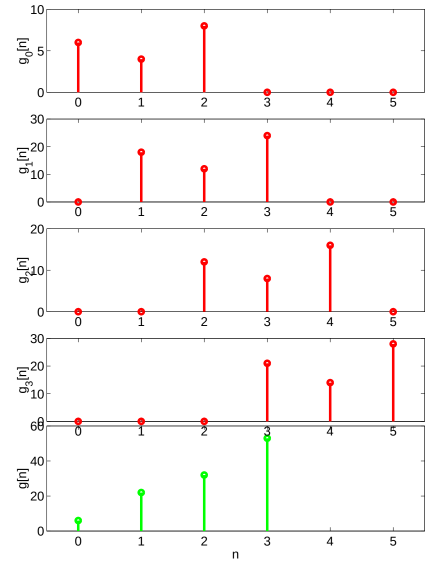
Each $f_i[n]$ is fed into the system separately:

$$f_i[n] \text{---} \boxed{h[n]} \text{---} g_i[n]$$

The outputs $g_i[n]$ are then summed:

$$g_i[n] \text{---} \boxed{\sum_i} \text{---} g[n]$$

Convolution by superposition:



"Rubber stamp" convolution recipe:

1. Initialize partial result $g'[n]$ to zero.
2. Make a rubber stamp out of $h[n]$. This is the point spread function (PSF). It is going to "spread out" each impulse (point) in $f[n]$.
3. Place the rubber stamp's $n = 0$ point (origin) on impulse $f[0]$.
4. Scale (multiply) the entire stamp by the value $f[0]$.
5. Add the result to $g'[n]$.
6. Place rubber stamp at $f[1]$. Scale by $f[1]$. Add result to g'[n].
7. Repeat for all n at which $f[n]$ is defined.
8. When done we have: $g[n] = f[n] * h[n] = g'[n]$.

Because of "point spread" at the ends of $f[n]$, the length of $g[n]$ is always $M + N - 1$, where M is the length of $f[n]$ and N is the length of $h[n]$.

Formalization

Discrete convolution in 1D:

$$g[n] = \sum_{n'=-\infty}^{\infty} f[n-n']h[n']$$

In our example:

$$\begin{aligned}
g[3] &= \sum_{n'=-\infty}^{\infty} f[3-n']h[n'] \\
&= 0 + \ldots + \sum_{n'=-1}^{n'=3} f[3-n']h[n'] + \ldots + 0 \\
&= f[4].h[-1] + f[3].h[0] + f[2].h[1] + f[1].h[2] + f[0].h[3] \\
&= 0 + 21 + 8 + 24 + 0 \\
&= 53
\end{aligned}$$

1D convolution as a matrix operation

$$\begin{bmatrix} g[0] \\ g[1] \\ g[2] \\ g[3] \\ g[4] \\ g[5] \\ g[6] \\ g[7] \\ g[8] \end{bmatrix} = \begin{bmatrix} h[0] & 0 & 0 & 0 & 0 & 0 & 0 \\ h[1] & h[0] & 0 & 0 & 0 & 0 & 0 \\ h[2] & h[1] & h[0] & 0 & 0 & 0 & 0 \\ 0 & h[2] & h[1] & h[0] & 0 & 0 & 0 \\ 0 & 0 & h[2] & h[1] & h[0] & 0 & 0 \\ 0 & 0 & 0 & h[2] & h[1] & h[0] & 0 \\ 0 & 0 & 0 & 0 & h[2] & h[1] & h[0] \\ 0 & 0 & 0 & 0 & 0 & h[2] & h[1] \\ 0 & 0 & 0 & 0 & 0 & 0 & h[2] \end{bmatrix} \begin{bmatrix} f[0] \\ f[1] \\ f[2] \\ f[3] \\ f[4] \\ f[5] \\ f[6] \end{bmatrix}$$

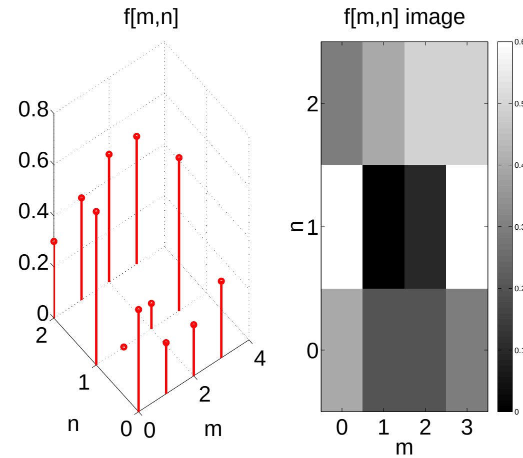
Does this make sense? Check example:

$$g[3] \quad = \quad f[3].h[0] + f[2].h[1] + f[1].h[2]$$

Matrices with constant elements along diagonal and subdiagonals are termed **Toeplitz matrices**.

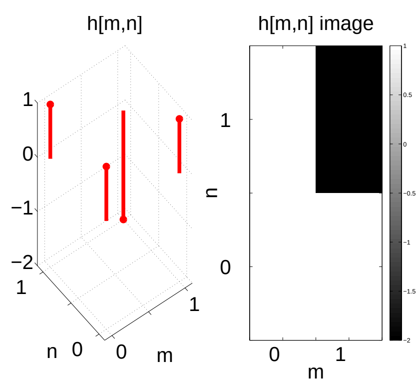
2D linear shift invariant systems

$$f[m,n] \longrightarrow \boxed{h[m,n]} \longrightarrow g[m,n]$$



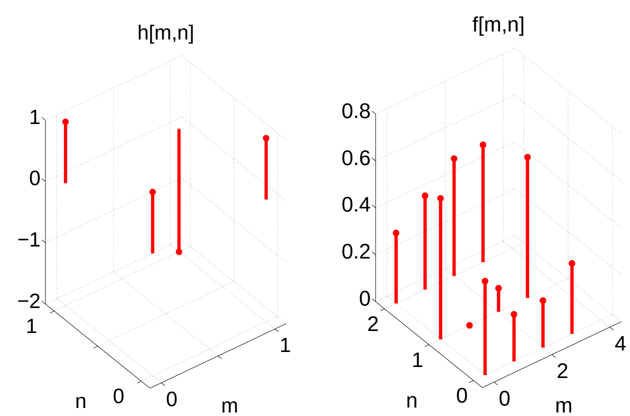
2D convolution kernel

$$\mathbf{h} = \begin{bmatrix} 1 & -2 \\ 1 & 1 \end{bmatrix}$$



Note: Our convention is to place the image origin at the bottom left of the matrix.

Discrete 2D convolution by superposition



$$\mathbf{h} = \begin{bmatrix} 1 & -2 \\ \boxed{1} & 1 \end{bmatrix} \qquad \mathbf{f} = \begin{bmatrix} 0.3 & 0.4 & 0.5 & 0.5 \\ 0.6 & 0.0 & 0.1 & 0.6 \\ \boxed{0.4} & 0.2 & 0.2 & 0.3 \end{bmatrix}$$

2D discrete convolution example (rubber stamp method)

Size of matrix g: $(N_f + N_h - 1) \times (M_f + M_h - 1) = (4 \times 5)$

$$\mathbf{g} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0.4 & -0.8 & 0 & 0 & 0 \\ 0.4 & 0.4 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0.2 & -0.4 & 0 & 0 \\ 0 & 0.2 & 0.2 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0.2 & -0.4 & 0 \\ 0 & 0 & 0.2 & 0.2 & 0 \end{bmatrix} +$$

$$\ldots \quad + \begin{bmatrix} 0 & 0.4 & -0.8 & 0 & 0 \\ 0 & 0.4 & 0.4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0.5 & -1 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0.5 & -1 \\ 0 & 0 & 0 & 0.5 & 0.5 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 0.3 & -0.2 & -0.3 & \underline{\hspace{1em}} & \underline{\hspace{1em}} \\ 0.9 & -0.5 & 1 & 1.4 & -0.7 \\ 1 & 0 & -0.1 & 0.6 & 0 \\ \underline{\hspace{1em}} & \underline{\hspace{1em}} & 0.4 & 0.5 & 0.3 \end{bmatrix}$$

Note: Only the numbers to be filled in are fully determined the terms shown here. How many matrices must be summed to give the full convolution? ____

Can we start the convolution at any point? ____

Formalization

Discrete convolution in 2D:

$$g[n,m] = \sum_{m'=-\infty}^{\infty} \sum_{n'=-\infty}^{\infty} f[m-m', n-n']\, h[m', n']$$

Reminder: The convolution operation is commutative:

$$g = f * h = h * f$$

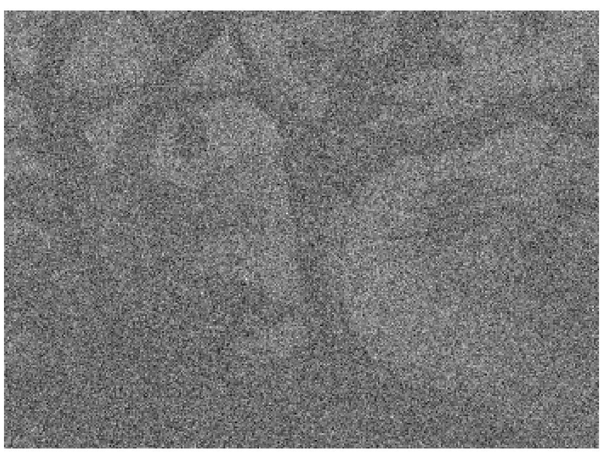
and distributes over addition:

$$h * (f_1 + f_2) = \underline{\hspace{3em}} + \underline{\hspace{3em}}$$

2D convolution example: smoothing a noisy image

Given noisy image $\mathbf{f}$:

Noisy image of trees



Choose a smoothing kernel h:

$$\mathbf{h} = \begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix} /256$$

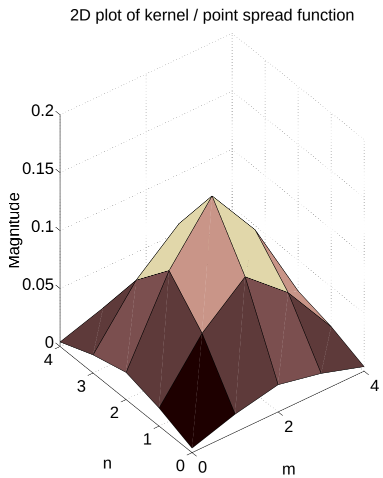
How far must two pixels be apart for one to have no effect on the value of the other after filtering? ____

Why do we divide all elements of the kernel by 256?

____

2D convolution example: smoothing a noisy image

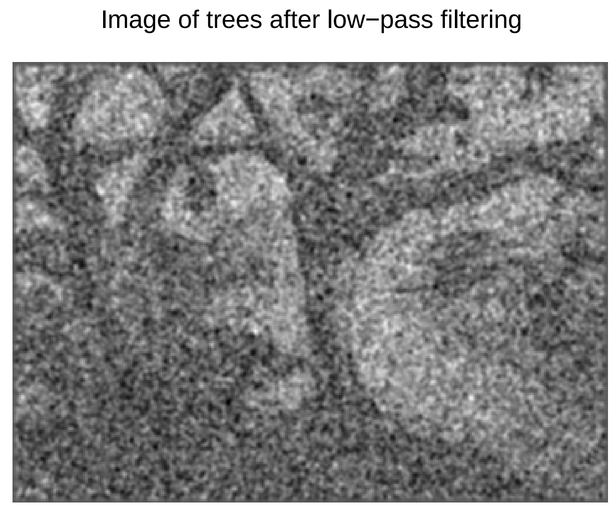
Surface plot of kernel:



This kernel is an approximation of a ____________ function.

2D convolution example: smoothing a noisy image

Filtered image:



2D convolution as a matrix operation

To show how 2D convolution can be expressed as a matrix multiplication, we must reorder an image into vector form.

Row ordering:

$$\mathbf{f} = \begin{bmatrix} 1 & 4 & 1 \\ 2 & 5 & 3 \end{bmatrix}$$

Define:

$$\mathsf{f} = \begin{bmatrix} \mathbf{f}_0 \\ \mathbf{f}_1 \end{bmatrix} = \begin{bmatrix} 1 \\ 4 \\ 1 \\ 2 \\ 5 \\ 3 \end{bmatrix}$$

where $\mathbf{f}_i$ is the $(i+1)$th row of $\mathbf{f}$.

2D convolution as a matrix operation

Similarly if:

$$\mathbf{h} = \begin{bmatrix} 1 & 2 \\ 1 & -1 \end{bmatrix}$$

Then

$$\mathsf{h} = \begin{bmatrix} 1 \\ 2 \\ 1 \\ -1 \end{bmatrix}$$

Define $\mathbf{h}_i$ as the $(i-1)$th row of $\mathbf{h}$.

2D convolution as a matrix operation

Define:

$$\mathbf{H}_0 = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & 2 & 1 \\ 0 & 0 & 2 \end{bmatrix}$$

and

$$\mathbf{H}_1 = \begin{bmatrix} 1 & 0 & 0 \\ -1 & 1 & 0 \\ 0 & -1 & \underline{\quad} \\ 0 & 0 & \underline{\quad} \end{bmatrix}$$

Also

$$\mathbf{H}_{-1} = \mathbf{0}, \quad \mathbf{H}_2 = \mathbf{0}$$

H_0 and H_1 are both Toeplitz matrices.

2D convolution as a matrix operation

The matrix operation equivalent to 2D convolution is then:

$$\mathsf{g} = \begin{bmatrix} \mathbf{g}_0 \\ \mathbf{g}_1 \\ \mathbf{g}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{H}_0 & \mathbf{0} \\ \mathbf{H}_1 & \mathbf{H}_0 \\ \mathbf{0} & \mathbf{H}_1 \end{bmatrix} \triangleq \mathcal{H}\mathsf{f}$$

Note: The matrix matrix $\mathcal{H}$ is termed “doubly Toeplitz”:

2D convolution as a matrix operation: example

$$\begin{bmatrix} 1 \\ 6 \\ 9 \\ 2 \\ 3 \\ 12 \\ 10 \\ 5 \\ 2 \\ 3 \\ -2 \\ -3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ -1 & 1 & 0 & 2 & 1 & 0 \\ 0 & -1 & 1 & 0 & 2 & 1 \\ 0 & 0 & -1 & 0 & 0 & 2 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 4 \\ 1 \\ 2 \\ 5 \\ 3 \end{bmatrix}$$

Formalization

Let the dimension of $\mathbf{f}$ be N_f rows by M_f columns. Then

$$\mathbf{g}_n = \sum_{n'=0}^{N-1} \mathbf{H}_{n-n'}\,\mathbf{f}_{n'}$$

The length of $\mathbf{g}_n$ (a row in the output image) is $(M_f + M_h - 1)$. Therefore, the size of $\mathbf{H}_n$ is $(M_f + M_h - 1) \times M_f$. $\mathbf{H}_n$ is a Toeplitz matrix of this size defined completely by its first column, which is:

$$\begin{bmatrix} \mathbf{h}_n \\ \mathbf{0} \end{bmatrix}$$

Why express convolutions in matrix form?

- Because we can use powerful tools of linear algebra such as the inverse of a matrix to solve many image processing problems.
- Because it will lead to great insight into the power of the Fourier transform to make reconstruction and restoration of large images feasible.