Introduction to Linear Systems

David Fleet

For operator T, input I, and response $R = T[I]$, T satisfies:

- **homogeniety:** *iff* $T[aI] = a\,T[I] \quad \forall a \in \mathcal{C}$
- **additivity:** *iff* $T[I_1 + I_2] = T[I_1] + T[I_2]$
- **superposition:** *iff* $T[aI_1 + bI_2] = a\,T[I_1] + b\,T[I_2] \quad \forall a, b \in \mathcal{C}$

T is linear iff it satisfies superposition.

We'll consider 1D signals for now, typically in one of three forms:

- **continuous:** $I(x)$ for $x \in \mathcal{R}$
- **discrete:** $I[n]$ for $n \in \mathcal{I}$, and often $0 \leq n \leq N-1$.
- **vector form:** $\vec{I} = (I[0], ..., I[N-1])^T$.

We will, where convenient, work with discrete signals.

If an operator T is **linear**, and $R = T[I]$, then there exists a matrix A where

$$\vec{R} = A\vec{I}$$

The response of a linear operator is given by matrix multiplication. The response at a particular sample, such as $R[n]$ is given by the inner product of $\vec{I}$ and the n^{th} row of A.

In the continuous domain, by comparison, the response is given by an integral equation of the form

$$R(x) = \int A(x, \xi)\, I(\xi)\, d\xi$$

In what follows, however, we are going to concentrate mainly on discrete signals and matrices as a consequence.

Shift-Invariance and Toeplitz Matrices

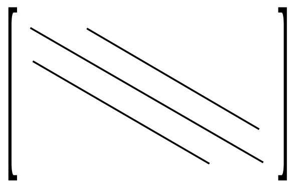
If T is a shift-invariant operator, then $\forall d \in \mathcal{I}$

$$R[n] = T\left[I[n]\right] \qquad iff \qquad R[n-m] = T\left[I[n-m]\right]$$

In words, the operator performs the same operation at every position in the signal. If I shift the input signal, then the response will be shifted a similar amount. There is no preferred origin.

A **linear, shift-invariant** operator is equivalent to a matrix multiplication with a Toeplitz matrix, A:

- each row of A is equal to the previous row, but shifted right by one; so that the elements of A are constant along the diagonal, as in



  You can also see from this that each column is a shifted version of every other column! You might find it useful to prove to yourself that shift-invariance implies a Toeplitz matrix with this structure.

- E.g., let's say we want to compute a weighted average of the input at each point using that point and its two neighbours. Let the weights be 0.25, 0.5, and 0.25 respectively. Then A has the form:

$$\frac{1}{4}\left[\begin{array}{ccccccccc} \ldots & 0 & 1 & 2 & 1 & 0 & \ldots & & \\ & \ldots & 0 & 1 & 2 & 1 & 0 & \ldots & \\ & & \ldots & 0 & 1 & 2 & 1 & 0 & \ldots \end{array}\right]$$

- Often we'll deal with *local* filters; i.e., pixels that contribute to the response at a position are nearby. This produces a banded Toeplitz matrix, like the averaging filter above. The width of nonzero entries in a row is called the *support* of the operator.

Shift-Invariance and Toeplitz Matrices (cont.)

Boundary Issues (failure of shift-invariance?): With finite discrete signals we cannot apply the same operation everywhere because there are no samples beyond the ends of signal. To handle this in a reasonable way:

1. Assume the input is padded with zeros beyond the endpoints out to infinity. Then we can talk about shift-invariance. But in practice, we only have to pad with zeros out to the support of the linear operator. If the operator support is M samples, then we need to pad by $M - 1$ zeros. This means that the length of the output vector will be longer than the input by $M - 1$.

2. If we pad the input as above, but we then truncate the output vector so it has the same length, as is often done, then the transform has the form:

$$\frac{1}{4}\begin{bmatrix} 2 & 1 & 0 & \dots & & & \\ 1 & 2 & 1 & 0 & \dots & & \\ & & \ddots & \ddots & \ddots & & \\ & & \dots & 0 & 1 & 2 & 1 \\ & & & \dots & 0 & 1 & 2 \end{bmatrix}$$

   This case definitely violates shift invariance. But for signals that are much longer than the support of the operator, we're often not too worried about the results at the boundaries.

3. Assume the signal is periodic and that shifts are cyclical. If one had a local operator with a banded matrix, then an assumption of cyclical shifts will introduce nonzero entries in the upper-right and lower-left corners of the matrix. For example:

$$\frac{1}{4}\begin{bmatrix} 2 & 1 & 0 & \dots & & & 1 \\ 1 & 2 & 1 & 0 & \dots & & \\ & & \ddots & \ddots & \ddots & & \\ & & \dots & 0 & 1 & 2 & 1 \\ 1 & & & \dots & 0 & 1 & 2 \end{bmatrix}$$

Convolution and Impulse Response

One can also characterize a linear operator with its impulse response.

- Kronecker delta function (discrete)

$$\delta[n] = \begin{cases} 1 & n = 0 \\ 0 & \text{otherwise} \end{cases}$$

- Dirac delta function (continuous)

$$\delta(x) = 0 \quad \forall x \neq 0, \quad \text{and} \quad \int \delta(x) f(x)\, dx = f(0)$$

  for sufficiently smooth $f(x)$

The impulse response of a linear shift-invariant system is the response to an impulse.

- In the discrete case, multiplying A by $\delta[n-m]$ amounts to extracting the m^{th} column from A.

$$\vec{h} \;=\; A\vec{e}_m, \quad \vec{e}_m \;=\; (0, \ldots, 0, 1, 0, \ldots, 0)^T.$$

  Here, $h[n-m]$ is referred to the impulse response. Note that if we handle boundaries by padding with zeros and truncating the result, then it would be wise to make the origin somewhere in the middle of the vector so that we don't get a truncated impulse response!

- If we applied A to a shifted version of the impulse, then we get another column, which is just a shifted version of the impulse response.

- Therefore, given the impulse response, you could construct the entire matrix! (ie. it contains all the relevant information to define the operator)

Convolution (cont.)

Remarks:

1. Another way to develop the idea of the inpulse response is to express the input signal as a weighted sum of shifted impulses. For example, assume that the signal value at position p was w_p. Then we can write $I[n]$ as

$$I[n] \;=\; \sum_{p=-\infty}^{\infty} w_p \, \delta[n-p]$$

Given T, along with superposition and shift-invariance, we find

$$\begin{aligned} T\,[I[n]] \;&=\; T\left[\sum_{p=-\infty}^{\infty} w_p \, \delta[n-p]\right] \\ &=\; \sum_{p=-\infty}^{\infty} w_p \, T\,[\delta[n-p]] \\ &=\; \sum_{p=-\infty}^{\infty} w_p \, h[n-p] \end{aligned}$$

Notice how the impulse response shows up again. This equation says that operator's output is equal to a weighted sum of shifted impulse responses.

2. We can also rewrite this operation in a slightly different form. Because the value of I at p is equal to w_p, and because $R[n] = T[I[n]]$, we can rewrite this last equation as

$$R[n] \;=\; \sum_{p=-\infty}^{\infty} I[p] \, h[n-p]$$

This is the most common formulation of a linear shift-invariant filter, and this operation is referred to as the convolution of I and h. It's sufficiently important that we have a special operator symbol for it, namely, $*$.

$$I * h \;=\; \sum_{p=-\infty}^{\infty} I[p] \, h[n-p] \;.$$

With continuous signals, $h(x)$ and $I(x)$, convolution is written as

$$I * h \;=\; \int_{p=-\infty}^{\infty} I(\xi) \, h(x-\xi) d\xi$$

3. It can be shown that convolution operator is

   - *commutative:* $I * h \;=\; h * I$
     (for periodic boundary treatment). Not all matrix operations commute, but this one does.
   - *associative:* $(h_1 * h_2) * h_3 \;=\; h_1 * (h_2 * h_3)$
     This is true of all matrix multiplication.
   - *distributive over addition:* $(h_1 + h_2) * h_3 \;=\; h_1 * h_3 + h_2 * h_3$
     This is true of all matrix multiplication.

4. Often the impulse response is relatively limited in its support (its number of nonzero values). Let's say that $h[m]$ is only nonzero for $M/2 \leq m \leq M/2$. In that case, it is convenient to rewrite the convolution equation in yet another way as

$$I * h \;=\; \sum_{p=-M/2}^{M/2} I[n+p]\, h[-p]\;.$$

   In words, this equation describes the following: for each image position, n, center the impulse response at that position, flip the impulse response, and then take its inner product with the image. This describes a much more efficient way to implement the filter than by matrix multiplication!

5. There are a variety of ways of finding an inverse to a convolution operator. One uses the Fourier transform. The other, more expensive but intuitively obvious method is to create the effective Toeplitz matrix, and invert it. One can show that the inverse of a Toepliz matrix (with periodic boundary treatment) is also Toepliz, which shows that the inverse of a linear shift-invariant operator, if it exists, is also linear and shift-invariant.

2D Convolution and Images

In two dimensions, a linear shift-invariant filter computes, at each position in the image, a linear combination of pixel values.
The convolution equation is given by

$$R[n,m] \;=\; \sum_{p=-\infty}^{\infty} \sum_{q=-\infty}^{\infty} I[p,q]\, h[n-p, m-q]$$

Computational Expense: Convolution in two dimensions will require, in the general case, $O(N^2M^2)$ multiplications and additions where N^2 is the number of pixels in the image, and M^2 is the 2d support of the impulse response. If we can decompose the filter into a separable filter, then we can reduce this computational load to $O(N^2M)$.

Separability: If a 2d filter can be expressed as $h(x,y) = h_1(x)\, h_2(y)$ for some $h_1(x)$ and some $h_2(y)$, then h is said to be separable. In the discrete case, an operator $h[n,m]$ is separable if it can be expressed as an outer product:

$$\begin{bmatrix} \\ \quad h[n,m] \quad \\ \\ \end{bmatrix} = \begin{pmatrix} | \\ h_1[n] \\ | \end{pmatrix} \begin{pmatrix} \text{—}\ h_2[m]\ \text{—} \end{pmatrix}$$

With separability, the convolution operation can be decomposed into a cascade of 1d convolutions, first along the rows, and then along the columns. Because convolution is commumtative, it doesn't matter which is done first. Each 1d operation requires only $O(n^2m)$ multiplications and additions. This yields a significant savings when the filter support is larger than 4 or 5 pixels.

Examples:

- The $m \times m$ constant matrix (a crude 2d averaging operator) can be expressed as an outer product of two 1d constant vectors.
- In continuous terms, the Gaussian is the only 2d isotropic function that can be decomposed into a separable product of two 1d Gaussians:

$$\frac{1}{2\pi\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \;=\; \frac{1}{\sqrt{2\pi}\sigma} e^{-x^2/2\sigma^2} \, \frac{1}{\sqrt{2\pi}\sigma} e^{-y^2/2\sigma^2}$$

- Discrete approximations to Gaussians are given by binomial coefficientsi, (e.g, (1, 4, 6, 4, 1)/16)).

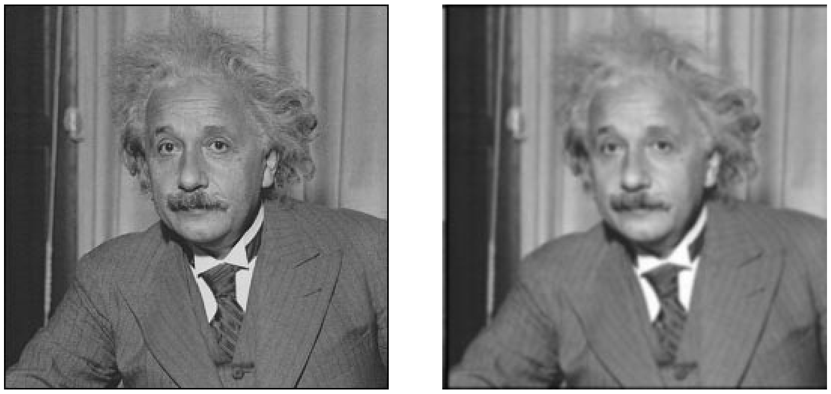


Figure 1: Blurring of Al: The original image of Al and a blurred version are shown. The blurring kernel was simple a separable kernel composed of the outer product of the 5-tap 1d impulse response $\frac{1}{16}(1, 4, 6, 4, 1)$.

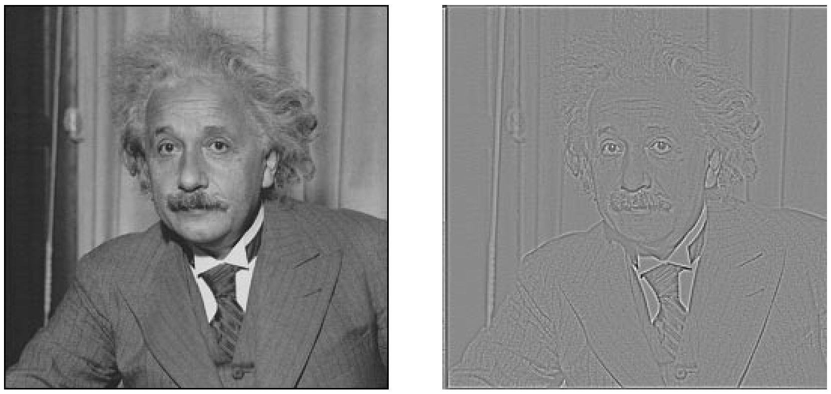


Figure 2: This shows Al and a high-pass filtered version of Al. This impulse response is defined by $\delta[n, m] - h[n, m]$ where $h[n, m]$ is the separable blurring kernel used in the previous figure.

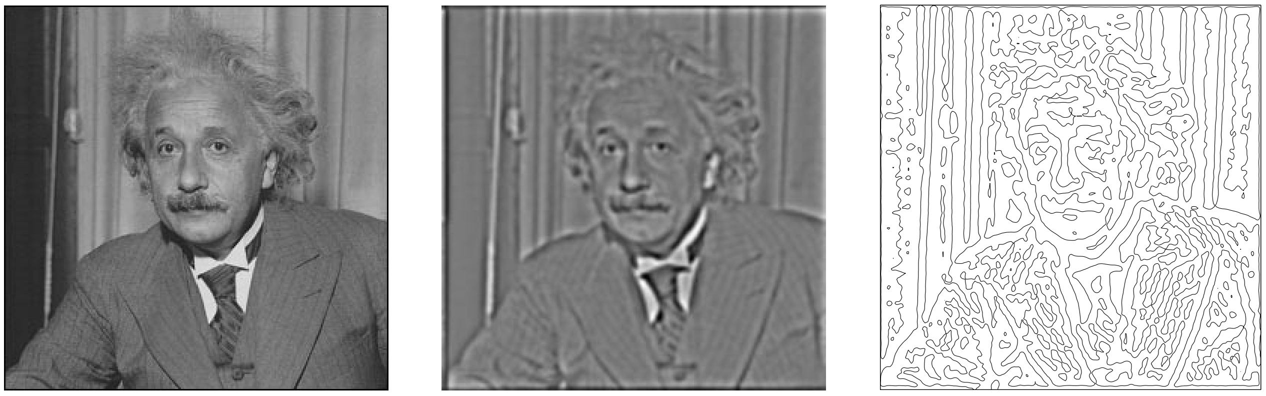


Figure 3: From left to right is the original Al, a band-pass filtered version of Al, and the zero-crossings of the filtered image. This impulse response is defined by the difference of two low-pass filters.

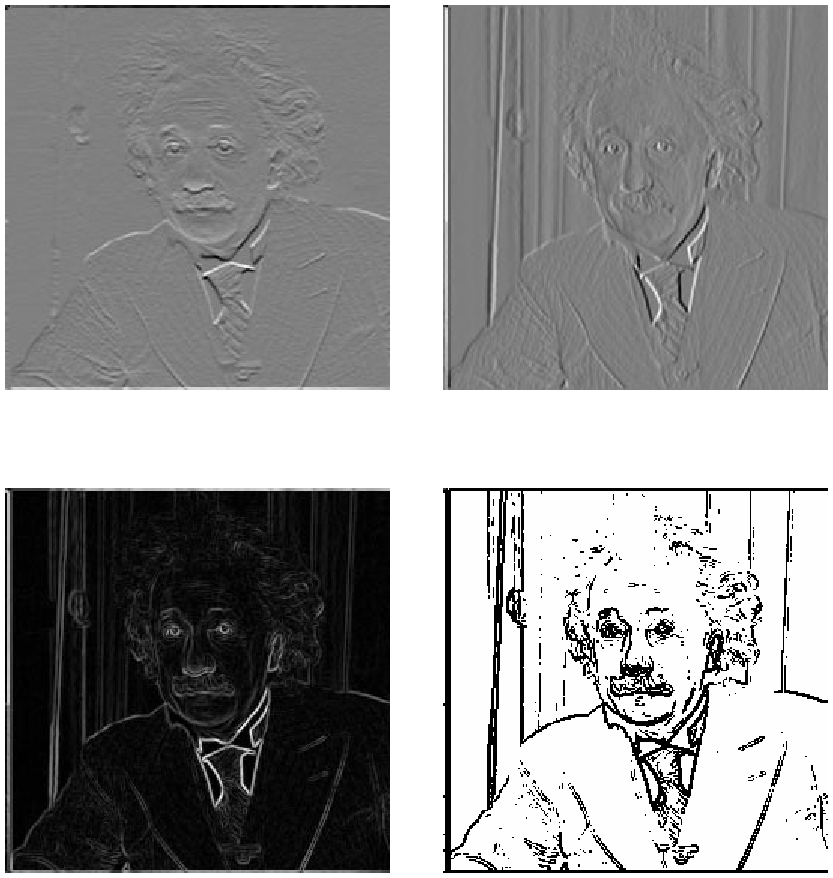


Figure 4: Derivative filters are common in image processing as discussed in more detail below. (top) Crude approximations to a horizontal derivative and a vertical derivative, each of which is separable and composed of an outer product of a smoothing filter in one direction (i.e., $\frac{1}{4}(1, 2, 1)$) and a first-order central difference (i.e., $\frac{1}{2}(-1, 0, 1)$) in the other direction. (bottom) The sum of squared derivative responses gives a measure of the magnitude of the image gradient at each pixel. When clipped, this gives us a rough idea of where edges might be found.