# Deconvolution

## Reverse of Convolution

$$x_t = w_t * e_t \rightarrow e_t = x_t * w^{-1}_t$$

=> Inverse Filtering

- **Aim of Deconvolution**
  1. **Theoretical: Reconstruction of the Reflectivity function**
  2. **Practical:**
  - **Shorting of the Signal**
  - **Suppression of Noise**
  - **Suppression of Multiples**

**An overview From Wikipedia, the free encyclopedia**

Jump to: navigation, search

In mathematics, **deconvolution** is an algorithm-based process used to reverse the effects of convolution on recorded data.[1] The concept of deconvolution is widely used in the techniques of signal processing and image processing. Because these techniques are in turn widely used in many scientific and engineering disciplines, deconvolution finds many applications.

In general, the object of deconvolution is to find the solution of a convolution equation of the form:

$$f * g = h$$

Usually, *h* is some recorded signal, and *ƒ* is some signal that we wish to recover, but has been convolved with some other signal *g* before we recorded it. The function *g* might represent the transfer function of an instrument or a driving force that was applied to a physical system. If we know *g*, or at least know the form of *g*, then we can perform deterministic deconvolution. However, if we do not know *g* in advance, then we need to estimate it. This is most often done using methods of statistical estimation[*citation needed*].

In physical measurements, the situation is usually closer to

$$(f * g) + \varepsilon = h$$

In this case *ε* is noise that has entered our recorded signal. If we assume that a noisy signal or image is noiseless when we try to make a statistical estimate of *g*, our estimate will be incorrect. In turn, our estimate of *ƒ* will also be incorrect. The lower the signal-to-noise ratio, the worse our estimate of the deconvolved signal will be. That is the reason why usually inverse filtering the signal is not a good solution. However, if we have at least some knowledge of the type of noise in the data (for example, white noise), we may be able to improve the estimate of *ƒ* through techniques such as Wiener deconvolution.

The foundations for deconvolution and time-series analysis were largely laid by Norbert Wiener of the Massachusetts Institute of Technology in his book *Extrapolation, Interpolation, and Smoothing of Stationary Time Series* (1949).[2] The book was based on work Wiener had done during World War II but that had been classified at the time. Some of the early attempts to apply these theories were in the fields of weather forecasting and economics.

## *Applications of Deconvolution*

### Seismology

The concept of Deconvolution had an early application in reflection seismology. In 1950, Enders Robinson was a graduate student at MIT. He worked with others at MIT, such as Norbert Wiener, Norman Levinson, and economist Paul Samuelson, to develop the "convolutional model" of a reflection seismogram. This model assumes that the recorded seismogram *s*(*t*) is the convolution of an Earth-reflectivity function *e*(*t*) and a seismic wavelet *w*(*t*) from a point source, where *t* represents recording time. Thus, our convolution equation is

$$s(t) = (e * w)(t).$$

The seismologist is interested in *e*, which contains information about the Earth's structure. By the convolution theorem, this equation may be Fourier transformed to

$$S(\omega) = E(\omega)W(\omega)$$

in the frequency domain. By assuming that the reflectivity is white, we can assume that the power spectrum of the reflectivity is constant, and that the power spectrum of the seismogram is the spectrum of the wavelet multiplied by that constant. Thus,

$$|S(\omega)| \approx k|W(\omega)|.$$

If we assume that the wavelet is minimum phase, we can recover it by calculating the minimum phase equivalent of the power spectrum we just found. The reflectivity may be recovered by designing and applying a Wiener filter that shapes the estimated wavelet to a Dirac delta function (i.e., a spike). The result may be seen as a series of scaled, shifted delta functions (although this is not mathematically rigorous):

$$e(t) = \sum_{i=1}^{N} r_i \delta(t - \tau_i),$$

where *N* is the number of reflection events, $\tau_i$ $\tau_i$ are the reflection times of each event, and $r_i$ are the reflection coefficients.

In practice, since we are dealing with noisy, finite bandwidth, finite length, discretely sampled datasets, the above procedure only yields an approximation of the filter required to deconvolve the data. However, by formulating the problem as the solution of a Toeplitz matrix and using Levinson recursion, we can relatively quickly estimate a filter with the smallest mean squared error possible. We can also do deconvolution directly in the frequency domain and get similar results. The technique is closely related to linear prediction.

## Optics and other imaging

In optics and imaging, the term "deconvolution" is specifically used to refer to the process of reversing the optical distortion that takes place in an optical microscope, electron microscope, telescope, or other imaging instrument, thus creating clearer images. It is usually done in the digital domain by a software algorithm, as part of a suite of microscope image processing techniques. Deconvolution is also practical to sharpen images that suffer from fast motion or jiggles during capturing. Early Hubble Space Telescope images were distorted by a flawed mirror and could be sharpened by Deconvolution.

The usual method is to assume that the optical path through the instrument is optically perfect, convolved with a point spread function (PSF), that is, a mathematical function that describes the distortion in terms of the pathway a theoretical point source of light (or other waves) takes through the instrument.[3] Usually, such a point source contributes a small area of fuzziness to the final image. If this function can be determined, it is then a matter of computing its inverse or complementary function, and convolving the acquired image with that. The result is the original, undistorted image.

In practice, finding the true PSF is impossible, and usually an approximation of it is used, theoretically calculated[4] or based on some experimental estimation by using known probes. Real optics may also have different PSFs at different focal and spatial locations, and the PSF may be non-linear. The accuracy of the approximation of the PSF will dictate the final result. Different algorithms can be employed to give better results, at the price of being more computationally intensive. Since the original convolution discards data, some algorithms use additional data acquired at nearby focal points to make up some of the lost information. Regularization in iterative algorithms (as in expectation-maximization algorithms) can be applied to avoid unrealistic solutions.

When the PSF is unknown, it may be possible to deduce it by systematically trying different possible PSFs and assessing whether the image has improved. This procedure is called *blind deconvolution*.[3] Blind deconvolution is a well-established image restoration technique in astronomy, where the point nature of the objects photographed exposes the PSF thus making it more feasible. It is also used in fluorescence microscopy for image restoration, and in fluorescence spectral imaging for spectral separation of multiple unknown fluorophores. The most common iterative algorithm for the purpose is the Richardson–Lucy deconvolution algorithm; the Wiener deconvolution (and approximations) are the most common non-iterative algorithms.

**Radio astronomy**

When performing image synthesis in radio interferometry, a specific kind of radio astronomy, one step consists of deconvolving the produced image with the "dirty beam", which is a different name for the point spread function. A common used method is the CLEAN algorithm.

### *References*

1. ^ O'Haver T. "Intro to Signal Processing - Deconvolution". University of Maryland at College Park. http://www.wam.umd.edu/~toh/spectrum/Deconvolution.html. Retrieved 2007-08-15.
2. ^ Wiener N (1964). *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. Cambridge, Mass: MIT Press. ISBN 0-262-73005-7.
3. ^ [a] [b] Cheng PC (2006). "The Contrast Formation in Optical Microscopy". *Handbook of Biological Confocal Microscopy (Pawley JB, ed.)* (3rd ed. ed.). Berlin: Springer. pp. 189–90. ISBN 038725921x.
4. ^ Nasse M. J., Woehl J. C. (2010). "Realistic modeling of the illumination point spread function in confocal scanning optical microscopy". *J. Opt. Soc. Am. A* **27** (2): 295–302. doi:10.1364/JOSAA.27.000295.

### *External links*

Presentations

- Flash presentation of blind deconvolution and source separation problem
- 3D simulations of deconvolution applied to Digital Room Correction
- Deconvolution Explanation and Examples

Tutorials and techniques

- Deconvolution in optical microscopy
- A summary of blind deconvolution techniques.
- Biophotonics article on deconvolution (PDF)
- Deconvolution Tutorial

Software

- Focus Magic, Software that uses deconvolution to fix out-of-focus blur and motion blur in an image
- Unshake, a blind deconvolver Java program for terrestrial photographs (currently freeware)
- SVI-wiki on 3D microscopy and deconvolution
- Tria Image Processing, Applies deconvolution and blind deconvolution to quickly remove blur from images

- DeblurMyImage from ADPTools - uses state-of-art deconvolution methods for out-of-focus correction and motion correction

Other

- Enders Robinson Oral History at IEEE

Retrieved from "http://en.wikipedia.org/wiki/Deconvolution"

- **Wiener deconvolution:**

***Definition***

Given a system:

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

where * denotes convolution. and:

- $x(t)$ is some input signal (unknown) at time $t$.
- $h(t)$ is the known impulse response of a linear time-invariant system
- $v(t)$ is some unknown additive noise, independent of $x(t)$
- $y(t)$ is our observed signal

Our goal is to find some $g(t)$ so that we can estimate $x(t)$ as follows:

$$\hat{x}(t) = g(t) * y(t)$$

Where $\hat{x}(t)$ is an estimate of $x(t)$ that minimizes the mean square error.

The Wiener deconvolution filter provides such a $g(t)$. The filter is most easily described in the frequency domain:

$$G(f) = \frac{H^*(f)S(f)}{|H(f)|^2 S(f) + N(f)}$$

Where:

- $G(f)$ and $H(f)$ are the Fourier transforms of $g$ and $h$, respectively at frequency $f$.
- $S(f)$ is the mean power spectral density of the input signal $x(t)$
- $N(f)$ is the mean power spectral density of the noise $v(t)$
- the superscript $^*$ denotes complex conjugation.

The filtering operation may either be carried out in the time-domain, as above, or in the frequency domain:

$$\hat{X}(f) = G(f)Y(f)$$

(where $\hat{X}(f)$ is the Fourier transform of $\hat{x}(t)$) and then performing an inverse Fourier transform on $\hat{X}(f)$ to obtain $\hat{x}(t)$.

Note that in the case of images, the arguments $t$ and $f$ above become two-dimensional; however the result is the same.

***Interpretation***

The operation of the Wiener filter becomes apparent when the filter equation above is rewritten:

$$G(f) = \frac{1}{H(f)}\left[\frac{|H(f)|^2}{|H(f)|^2 + \frac{N(f)}{S(f)}}\right]$$
$$= \frac{1}{H(f)}\left[\frac{|H(f)|^2}{|H(f)|^2 + \frac{1}{\mathrm{SNR}(f)}}\right]$$

Here, $1/H(f)$ is the inverse of the original system, and $\mathrm{SNR}(f) = S(f)/N(f)$ is the signal-to-noise ratio. When there is zero noise (i.e. infinite signal-to-noise), the term inside the square brackets equals 1, which means that the Wiener filter is simply the inverse of the system, as we might expect. However, as the noise at certain frequencies increases, the signal-to-noise ratio drops, so the term inside the square brackets also drops. This means that the Wiener filter attenuates frequencies dependent on their signal-to-noise ratio.

The Wiener filter equation above requires us to know the spectral content of a typical image, and also that of the noise. Often, we do not have access to these exact quantities, but we may be in a situation where good estimates can be made. For instance, in the case of photographic images, the signal (the original image) typically has strong low frequencies and weak high frequencies, and in many cases the noise content will be relatively flat with frequency.

***Derivation***

As mentioned above, we want to produce an estimate of the original signal that minimizes the mean square error, which may be expressed:

$$\epsilon(f) = \mathbb{E}\left|X(f) - \hat{X}(f)\right|^2$$

Where $\mathbb{E}$ denotes expectation.

If we substitute in the expression for $\hat{X}(f)$, the following rearrangements can be made:

$$\begin{aligned}\epsilon(f) &= \mathbb{E}\,|X(f) - G(f)Y(f)|^2 \\ &= \mathbb{E}\,|X(f) - G(f)\,[H(f)X(f) + V(f)]|^2 \\ &= \mathbb{E}\big|\,[1 - G(f)H(f)]\,X(f) - G(f)V(f)\big|^2\end{aligned}$$

If we expand the quadratic, we get the following:

$$\begin{aligned}\epsilon(f) = &\Big[1 - G(f)H(f)\Big]\Big[1 - G(f)H(f)\Big]^*\,\mathbb{E}|X(f)|^2 \\ &+ \Big[1 - G(f)H(f)\Big]G^*(f)\,\mathbb{E}\Big\{X(f)V^*(f)\Big\} \\ &+ G(f)\Big[1 - G(f)H(f)\Big]^*\,\mathbb{E}\Big\{V(f)X^*(f)\Big\} \\ &+ G(f)G^*(f)\,\mathbb{E}|V(f)|^2\end{aligned}$$

However, we are assuming that the noise is independent of the signal, therefore:

$$\mathbb{E}\Big\{X(f)V^*(f)\Big\} = \mathbb{E}\Big\{V(f)X^*(f)\Big\} = 0$$

Also, we are defining the power spectral densities as follows:

$$\begin{aligned}S(f) &= \mathbb{E}|X(f)|^2 \\ N(f) &= \mathbb{E}|V(f)|^2\end{aligned}$$

Therefore, we have:

$$\begin{aligned}\epsilon(f) = &\Big[1 - G(f)H(f)\Big]\Big[1 - G(f)H(f)\Big]^*S(f) \\ &+ G(f)G^*(f)N(f)\end{aligned}$$

To find the minimum error value, we differentiate with respect to $G(f)$ and set equal to zero. As this is a complex value, $G^*(f)$ acts as a constant.

$$\frac{d\epsilon(f)}{dG(f)} = G^*(f)N(f) - H(f)\Big[1 - G(f)H(f)\Big]^* S(f) = 0$$

This final equality can be rearranged to give the Wiener filter.

- ## Richardson–Lucy deconvolution

The **Richardson–Lucy algorithm**, also known as **Lucy-Richardson** **deconvolution**, is an iterative procedure for recovering a latent image that has been blurred by a known point spread function.[1][2]

Pixels in the observed image can be represented in terms of the point spread function and the latent image as

$$d_i = \sum_j p_{ij} u_j$$

where $p_{ij}$ is the point spread function (the fraction of light coming from true location $j$ that is observed at position $i$), $u_j$ is the pixel value at location $j$ in the latent image, and $d_i$ is the observed value at pixel location $i$. The statistics are performed under the assumption that $u_j$ are Poisson distributed, which is appropriate for photon noise in the data.

The basic idea is to calculate the most likely $u_j$ given the observed $d_i$ and known $p_{ij}$. This leads to an equation for $u_j$ which can be solved iteratively according to

$$u_j^{(t+1)} = u_j^{(t)} \sum_i \frac{d_i}{c_i} p_{ij}$$

Where

$$c_i = \sum_j p_{ij} u_j^{(t)}.$$

It has been shown empirically that if this iteration converges, it converges to the maximum likelihood solution for $u_j$.[3]

In problems where the point spread function $p_{ij}$ is dependent on one or more unknown parameters, the Richardson–Lucy algorithm cannot be used. A later and more general class of algorithms, the expectation-maximization algorithms,[4] have been applied to this type of problem with great success

***References***

1. ^ Richardson, William Hadley (1972). "Bayesian-Based Iterative Method of Image Restoration". *JOSA* **62** (1): 55–59. doi:10.1364/JOSA.62.000055. http://www.opticsinfobase.org/abstract.cfm?id=54565.
2. ^ Lucy, L. B. (1974). "An iterative technique for the rectification of observed distributions". *Astronomical Journal* **79** (6): 745–754. doi:10.1086/111605.
3. ^ Shepp, L. A.; Vardi, Y. (1982), "Maximum Likelihood Reconstruction for Emission Tomography", *IEEE Transactions on Medical Imaging* **1**: 113, doi:10.1109/TMI.1982.4307558
4. ^ A.P. Dempster, N.M. Laird, D.B. Rubin, 1977, *Maximum likelihood from incomplete data via the EM algorithm*, J. Royal Stat. Soc. Ser. B, **39** (1), pp. 1–38

Retrieved from "http://en.wikipedia.org/wiki/Richardson%E2%80%93Lucy_deconvolution"

- **Digital room correction**

**Digital room correction** (or **DRC**) is a process in the field of acoustics where digital filters designed to ameliorate unfavorable effects of a room's acoustics are applied to the input of a sound reproduction system. Modern room correction systems produce substantial improvements in the time domain and frequency domain response of the sound reproduction system.

### *History*

The use of analog filters, such as equalizers, to normalize the frequency response of a playback system has a long history; however, analog filters are very limited in their ability to correct the distortion found in many rooms. Although digital implementations of the equalizers have been available for some time, digital room correction is usually used to refer to the construction of filters which attempt to invert the impulse response of the room and playback system, at least in part. Digital correction systems are able to use acausal filters,[*dubious* – *discuss*] and are able to operate with optimal time resolution, optimal frequency resolution, or any desired compromise along the gabor limit. Digital room correction is a fairly new area of study which has only recently been made possible by the computational power of modern CPUs and DSPs.

### *Operation*

The configuration of a digital room correction system begins with measuring the impulse response of the room at the listening location for each of the loudspeakers. Then, computer software is used to compute a FIR filter, which reverses the effects of the room and linear distortion in the loudspeakers. Finally, the calculated filter is loaded into a computer or other room correction device which applies the filter in real time. Because most room correction filters are acausal, there is some delay. Most DRC systems allow the operator to control the added delay through configurable parameters.

### *Challenges*

DRC systems are not normally used to create a perfect inversion of the room's response because a perfect correction would only be valid at the location where it was measured: a few millimeters away the arrival times from various reflections will differ and the inversion will be imperfect. The imperfectly corrected signal may end up sounding worse than the uncorrected signal because the acausal filters used in digital room correction may cause pre-echo. Room correction filter calculation systems employ sophisticated processing to attempt to produce an inverse filter which will work over a usably large area, and which avoid producing bad-sounding artifacts outside of that area, at the expense of peak accuracy at the measurement location.

- **Free convolution**

**Free convolution** is the free probability analog of the classical notion of convolution of probability measures. Due to the non-commutative nature of free probability theory, one has talk separately about additive and multiplicative free convolution, which arise from addition and multiplication of free random variables (see below; in the classical case, what would be the analog of free multiplicative convolution can be reduced to additive convolution by passing to logarithms of random variables). The notion of free convolution was introduced by Voiculescu in early 80s

***Free additive convolution***

Let μ and v be two probability measures on the real line, and assume that *X* is a random variable with law μ and *Y* is a random variable with law v. Assume finally that *X* and *Y* are freely independent. Then the **free additive convolution** $\mu \boxplus \nu$ is the law of *X* + *Y*.

In many cases, it is possible to compute the probability measure $\mu \boxplus \nu$ explicitly by using complex-analytic techniques and the R-transform of the measures μ and v.

### *Free multiplicative convolution*

Let μ and v be two probability measures on the interval $[0, +\infty)$, and assume that *X* is a random variable with law μ and *Y* is a random variable with law v. Assume finally that *X* and *Y* are freely independent. Then the **free multiplicative convolution** $\mu \boxtimes \nu$ is the law of $X^{1/2}YX^{1/2}$ (or, equivalently, the law of $Y^{1/2}XY^{1/2}$.

A similar definition can be made in the case of laws μ,v supported on the unit circle {*z*: | *z* | = 1}.

Explicit computations of multiplicative free convolution can be carried out using complex-analytic techniques and the S-transform.

### *Applications of free convolution*

- Free convolution can be used to give a proof of the free central limit theorem.
- Free convolution can be used to compute the laws and spectra of sums or products of random variables which are free. Such examples include: random walk operators on free groups (Kesten measures); and asymptotic distribution of eigenvalues of sums or products of independent random matrices.

Through its applications to random matrices, free convolution has some strong connections with other works on G-estimation of Girko.

The applications in wireless communications, finance and biology have provided a useful framework when the number of observations is of the same order as the dimensions of the system.

### *References*

1. ^ Voiculescu, D., Addition of certain non-commuting random variables, J. Funct. Anal. 66 (1986), 323–346

2. ^ Voiculescu, D., Multiplciation of certain noncommuting random variables , J. Operator Theory 18 (1987), 2223–2235.

- "Free Deconvolution for Signal Processing Applications", O. Ryan and M. Debbah, ISIT 2007, pp. 1846–1850

## *External links*

- Alcatel Lucent Chair on Flexible Radio
- http://www.cmapx.polytechnique.fr/~benaych
- http://folk.uio.no/oyvindry

Retrieved from "http://en.wikipedia.org/wiki/Free_convolution"

- **Numerical evaluation of Cross-correlation**

$$\phi_{xy}(\tau) = \sum_{i=1}^{n-\tau} x_{i+\tau} y_i$$

$x_i$: (i=0 ... n)

$y_i$: (i= 0 ... n)

$\phi_{xy}(\tau)$ : ($-m < \tau < +m$)

m = max. displacement

**In Fourier domain:**

**Cross-correlation = Multiplication of Amplitude spectrum and Subtraction of Phase spectrum**

# Autocorrelation

**Cross-correlation of a Function with itself**

$$\phi_{xx}(\tau) = \sum_{i=1}^{n-\tau} x_{i+\tau} x_i$$

$x_i$ = (i=0 ... n)

$\phi_{xx}(\tau)$ = $(-m < \tau < +m)$

m = max. displacement

# Normalization of correlation

**Auto-correlation**

$$\phi_{xx}(\tau)_{norm} = \frac{\phi_{xx}(\tau)}{\phi_{xx}(0)}$$

**Cross-correlation**

$$\phi_{xy}(\tau)_{norm} = \frac{\phi_{xy}(\tau)}{\sqrt{\phi_{xx}(0)\phi_{yy}(0)}}$$

# Circular Convolution

## 1. Introduction

You should be familiar with Discrete-Time Convolution1, which tells us that given two discrete-time signals x [n], the system's input, and h [n], the system's response, we define the output of the system as

$$\begin{aligned} y[n] &= x[n] * h[n] \\ &= \sum_{k=-\infty}^{\infty} (x[k]\, h[n-k]) \end{aligned}$$

When we are given two DFTs (finite-length sequences usually of length N), we cannot just multiply them together as we do in the above convolution formula, often referred to as linear convolution. Because the DFTs are periodic, they have nonzero values for n $\geq$ N and thus the multiplication of these two DFTs will be nonzero for n $\geq$ N. We need to define a new type of convolution operation that will result in our convolved signal being zero outside of the range n = {0, 1. . . N − 1}. This idea led to the development of circular convolution, also called cyclic or periodic convolution.

## 2. Circular Convolution Formulas

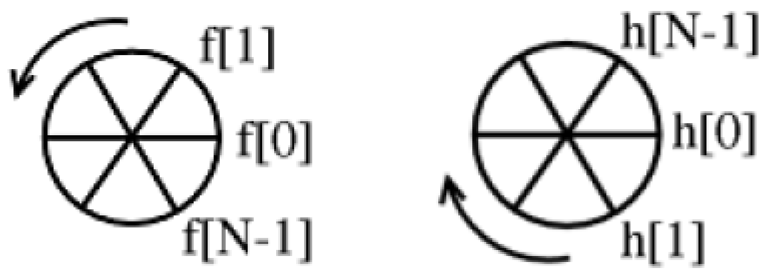
What happens when we multiply two DFT's together, where Y [k] is the DFT of y [n]?

$$Y[k] = F[k]\, H[k]$$

When $0 \leq k \leq N-1$

Using the DFT synthesis formula for y [n]

$$y[n] = \frac{1}{N} \sum_{k=0}^{N-1} \left( F[k]\, H[k]\, e^{j \frac{2\pi}{N} kn} \right)$$




(a) (b)

**Figure 1:** Step 1

And then applying the analysis formula $F[k] = \sum_{m=0}^{N-1} \left( f[m]\, e^{(-j) \frac{2\pi}{N} kn} \right)$

$$\begin{aligned} y[n] &= \frac{1}{N} \sum_{k=0}^{N-1} \left( \left( \sum_{m=0}^{N-1} \left( f[m]\, e^{(-j) \frac{2\pi}{N} kn} \right) \right) H[k]\, e^{j \frac{2\pi}{N} kn} \right) \\ &= \sum_{m=0}^{N-1} \left( f[m] \left( \frac{1}{N} \sum_{k=0}^{N-1} \left( H[k]\, e^{j \frac{2\pi}{N} k(n-m)} \right) \right) \right) \end{aligned}$$

where we can reduce the second summation found in the above equation into $h[((n-m))_N] = \frac{1}{N} \sum_{k=0}^{N-1} \left( H[k]\, e^{j \frac{2\pi}{N} k(n-m)} \right)$

$$y[n] = \sum_{m=0}^{N-1} \left( f[m]\, h[((n-m))_N] \right)$$

which equals circular convolution! When we have $0 \leq n \leq N-1$ in the above, then we get:

$$y[n] \equiv (f[n] \circledast h[n])$$

NOTE: The notation $\circledast$ represents cyclic convolution "mod N".

## 2.1 Steps for Cyclic Convolution

Steps for cyclic convolution are the same as the usual convo, except all index calculations are done "mod N" = "on the wheel"

Steps for Cyclic Convolution

- Step 1: "Plot" f [m] and h [((−m))N]
- Step 2: "Spin" h [((−m))N] n notches ACW (counter-clockwise) to get h [((n − m))N] (i.e. Simply rotate the sequence, h [n], clockwise by n steps).
- Step 3: Pointwise multiply the f [m] wheel and the h [((n − m))N] wheel.

Sum = y [n]

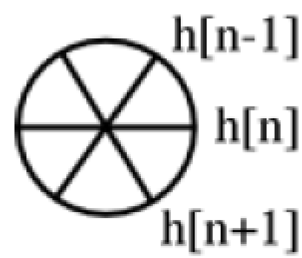



**Figure 2:** Step 2

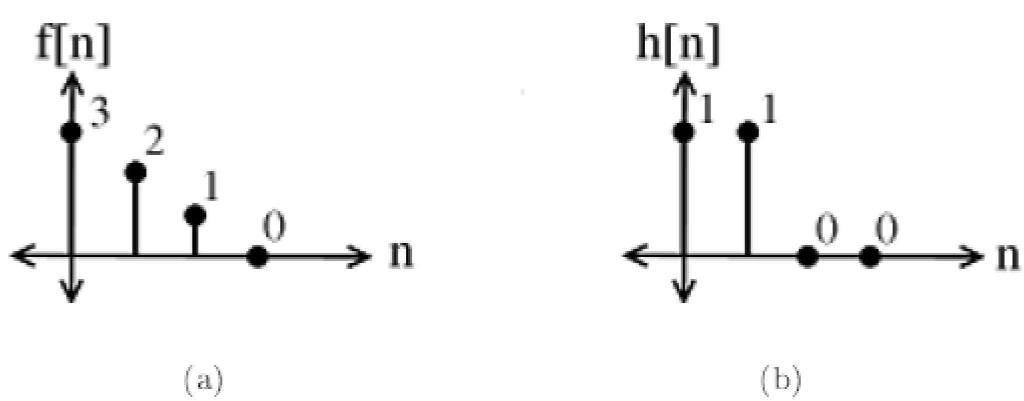



**Figure 3: Two discrete-time signals to be convolved.**

- Step 4: Repeat for all $0 \leq n \leq N-1$

**Example 1: Convolve (n = 4)**

- $h[((-m))_N]$

Multiply $f[m]$ and sum to yield: $y[0] = 3$

- $h[((1-m))_N]$

Multiply $f[m]$ and sum to yield: $y[1] = 5$

- $h[((2-m))_N]$

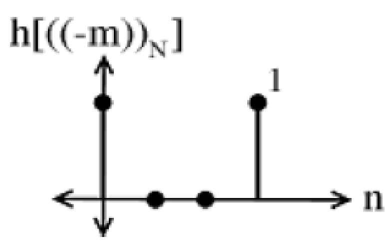



**Figure 4**

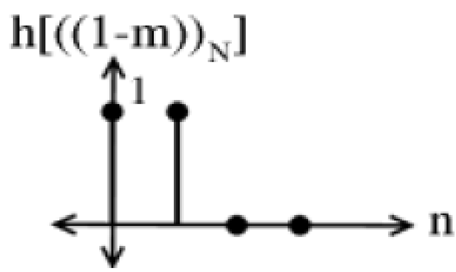



**Figure 5**

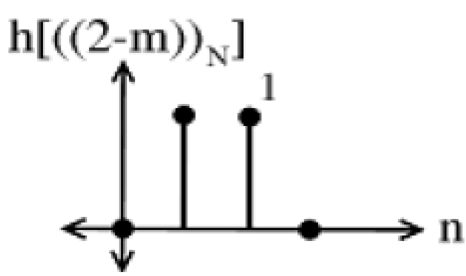



**Figure 6**

Multiply f [m] and sum to yield: y [2] = 3

• h [((3 − m))N]

Multiply f [m] and sum to yield: y [3] = 1

## 2.2 Alternative Algorithm

Alternative Circular Convolution Algorithm

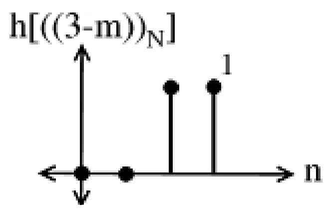



Figure 7

• Step 1: Calculate the DFT of f [n] which yields F [k] and calculate the DFT of h [n] which yields H [k].

• Step 2: Pointwise multiply Y [k] = F [k]H [k]

• Step 3: Inverse DFT Y [k] which yields y [n]

Seems like a roundabout way of doing things, but it turns out that there are extremely fast ways to calculate the DFT of a sequence.

To circularily convolve 2 N-point sequences:

$$y[n] = \sum_{m=0}^{N-1} \left( f[m]\, h[((n-m))_N] \right)$$

For each $n$ : $N$ multiples, $N-1$ additions

$N$ points implies $N^2$ multiplications, $N(N-1)$ additions implies $O(N^2)$ complexity.