Course: Digital Signal and Data Processing

MOVING AVERAGE FILTERS

Lecture 2

Associate professor Naila Allakhverdiyeva

Moving Average Filter

- The moving average filter operates by averaging a number of points from the input signal to produce each point in the output signal.
- In equation form , this is written:

$$y[i] = \frac{1}{M} \sum_{j=0}^{M-1} x[i+j]$$

- In this equation, x[ ] is the input signal, *y*[ ] is the output signal, and *M* is the number of points used in the moving average.
- This equation only uses points on *one side* of the output sample being calculated.

Moving Average Filter

- For example, in a 5 point moving average filter, point 80 in the output signal is given by:

$$y[80] = \frac{x[80] + x[81] + x[82] + x[83] + x[84]}{5}$$

- As an alternative, the group of points from the input signal can be chosen ***symmetrically*** around the output point:

$$y[80] = \frac{x[78] + x[79] + x[80] + x[81] + x[82]}{5}$$

- This corresponds to changing the summation from: j= 0 to M-1, to: $j = -(M-1)/2 \; to \; (M-1)/2$.
- Symmetrical averaging requires that M be an ***odd*** number.

Moving Average Filter

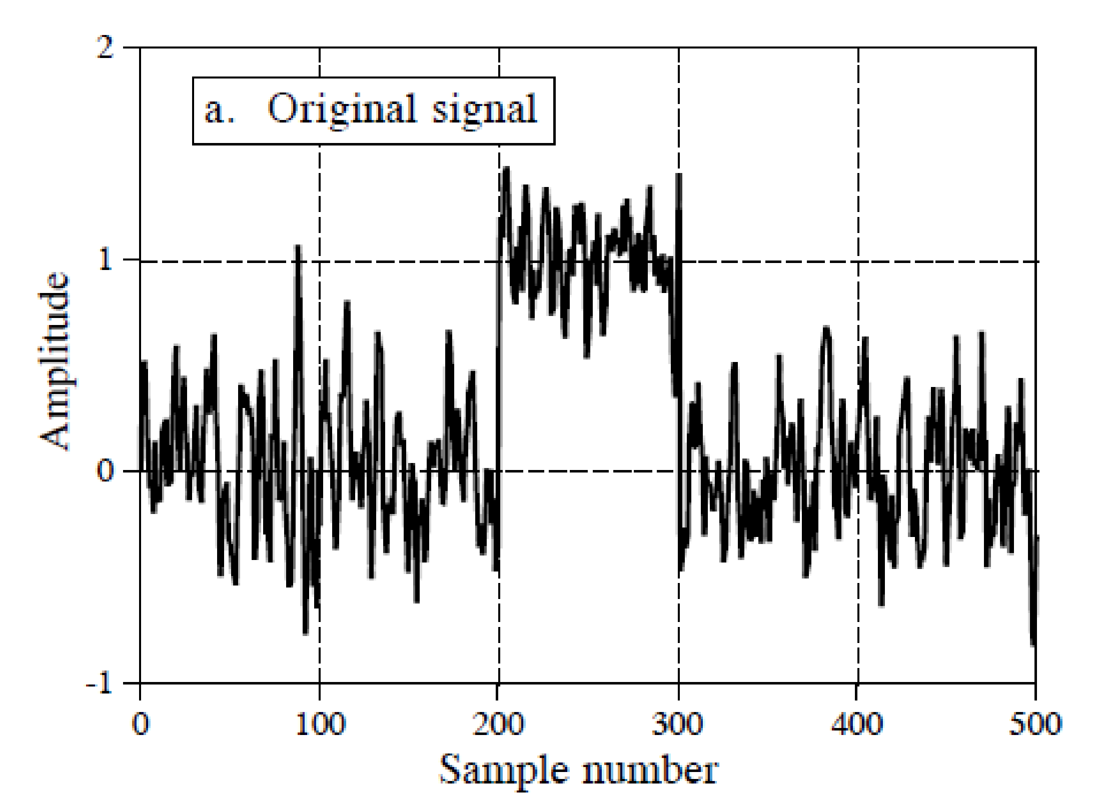
- The moving average filter is a ***convolution*** using a very simple filter kernel.
- For example, a 5 point filter has the filter kernel: …, 0, 0, 1/5, 1/5, 1/5, 1/5, 1/5, 0, 0, …
- The moving average filter is a **convolution of the input signal with a *rectangular pulse* having an area of *one***.

Noise Reduction vs. Step Response

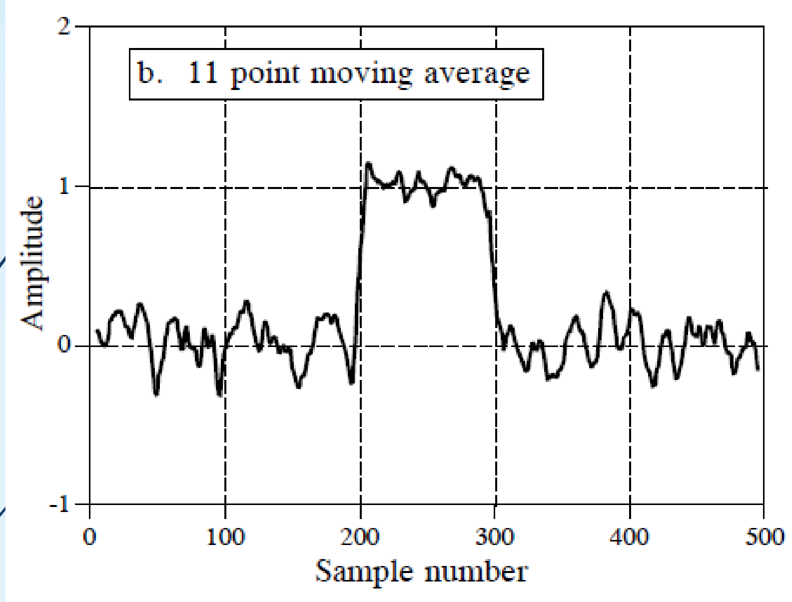
- Moving average filter is very good for many applications.
- It is ***optimal*** for a common problem, reducing random white noise while keeping the sharpest step response.

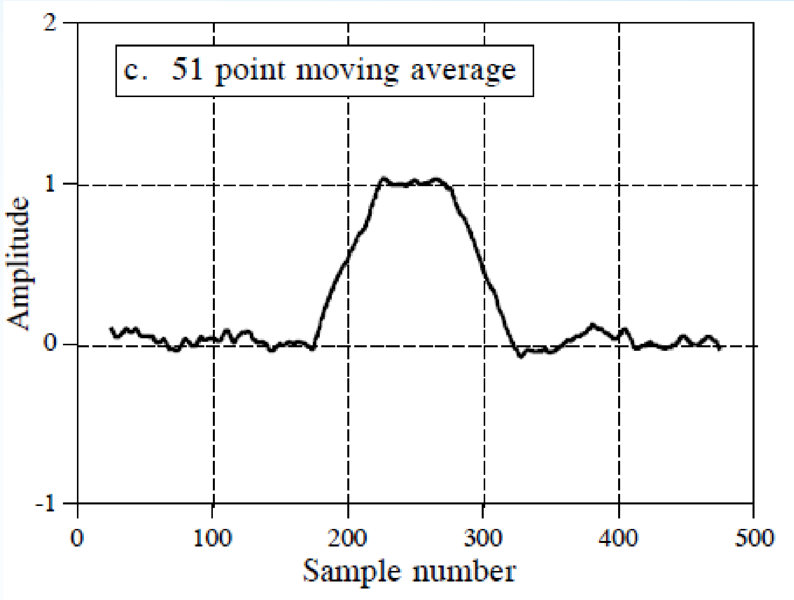
Example of using a moving average filter.

In (a), a rectangular pulse is buried in random noise.



Noise Reduction vs. Step Response



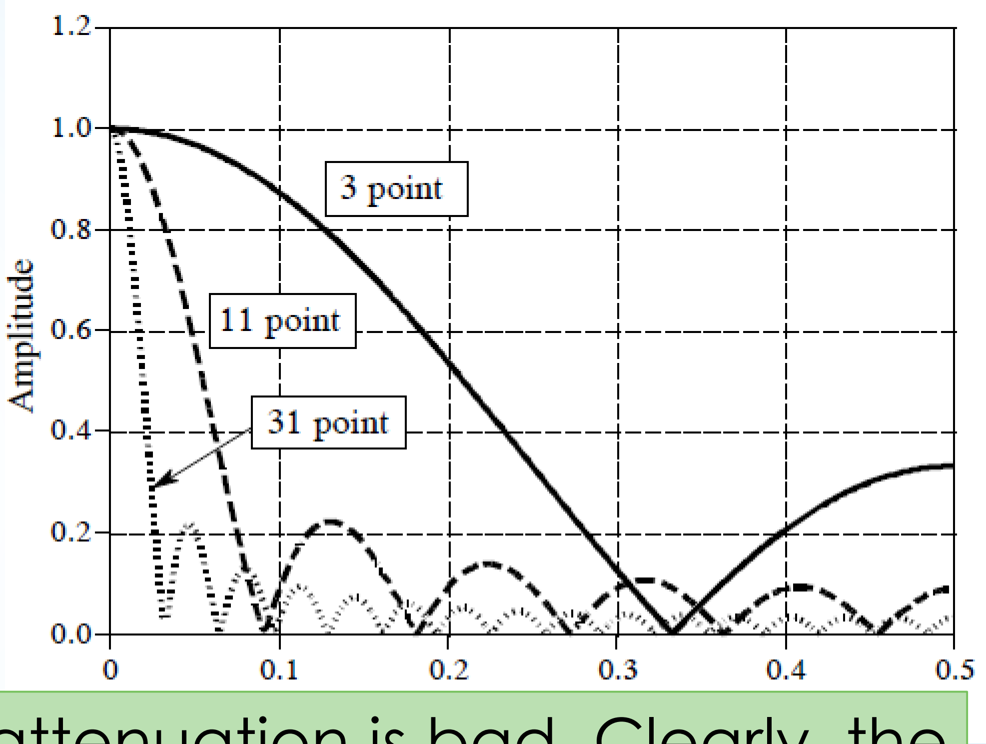


In (b) and (c), the smoothing action of the moving average filter decreases the amplitude of the random noise (good), but also reduces the sharpness of the edges (bad). Of all the possible linear filters that could be used, the **moving average produces the lowest noise for a given edge sharpness**. The amount of noise reduction is equal to the square-root of the number of points in the average. For example, a 100 point moving average filter reduces the noise by a factor of 10.

Frequency Response of MA Filter

Frequency response of an M point moving average filter is mathematically described by the Fourier transform of the rectangular pulse:

$$H[f] = \frac{\sin(\pi f M)}{M \sin(\pi f)}$$



The frequency, f, runs between 0 and 0.5.
For f=0, $H[f] = 1$

The roll-off is very slow and the stopband attenuation is bad. Clearly, the MA filter **cannot separate** one band of frequencies from another. **Good performance in the time domain results in poor performance in the frequency domain, and vice versa**. In short, the MA is an exceptionally ***good smoothing filter*** (time domain), but an exceptionally ***bad low-pass filter*** (frequency domain).

Relatives of the Moving Average Filter

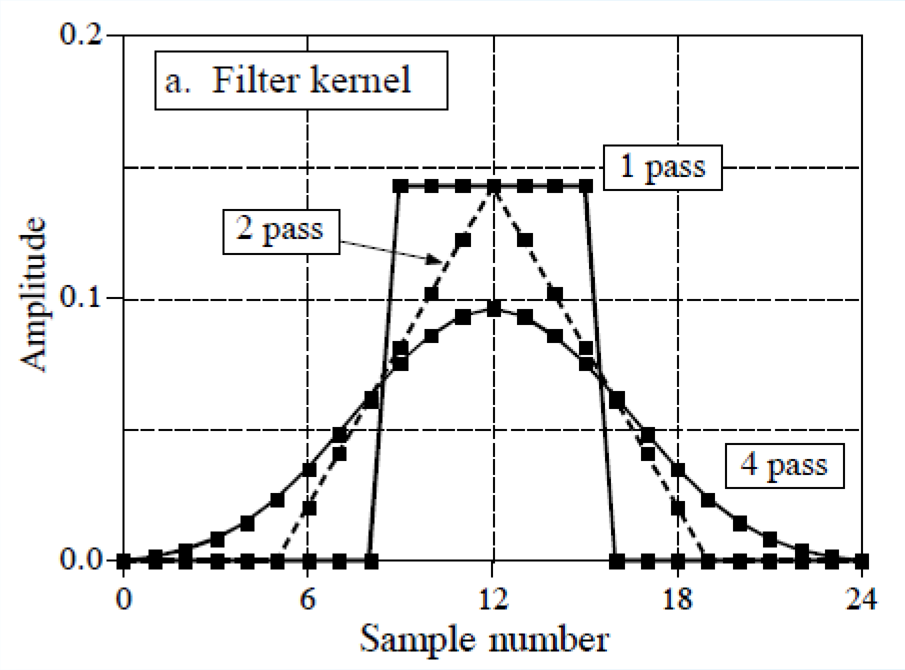
Filter designers would only have to deal with time domain *or* frequency domain encoded information, but never a mixture of the two in the same signal. Unfortunately, there are some applications where **both domains are simultaneously important**.

- Television signals (video information)
- Electromagnetic interference

Relatives of the moving average filter have better frequency domain performance, and can be useful in these mixed domain applications.

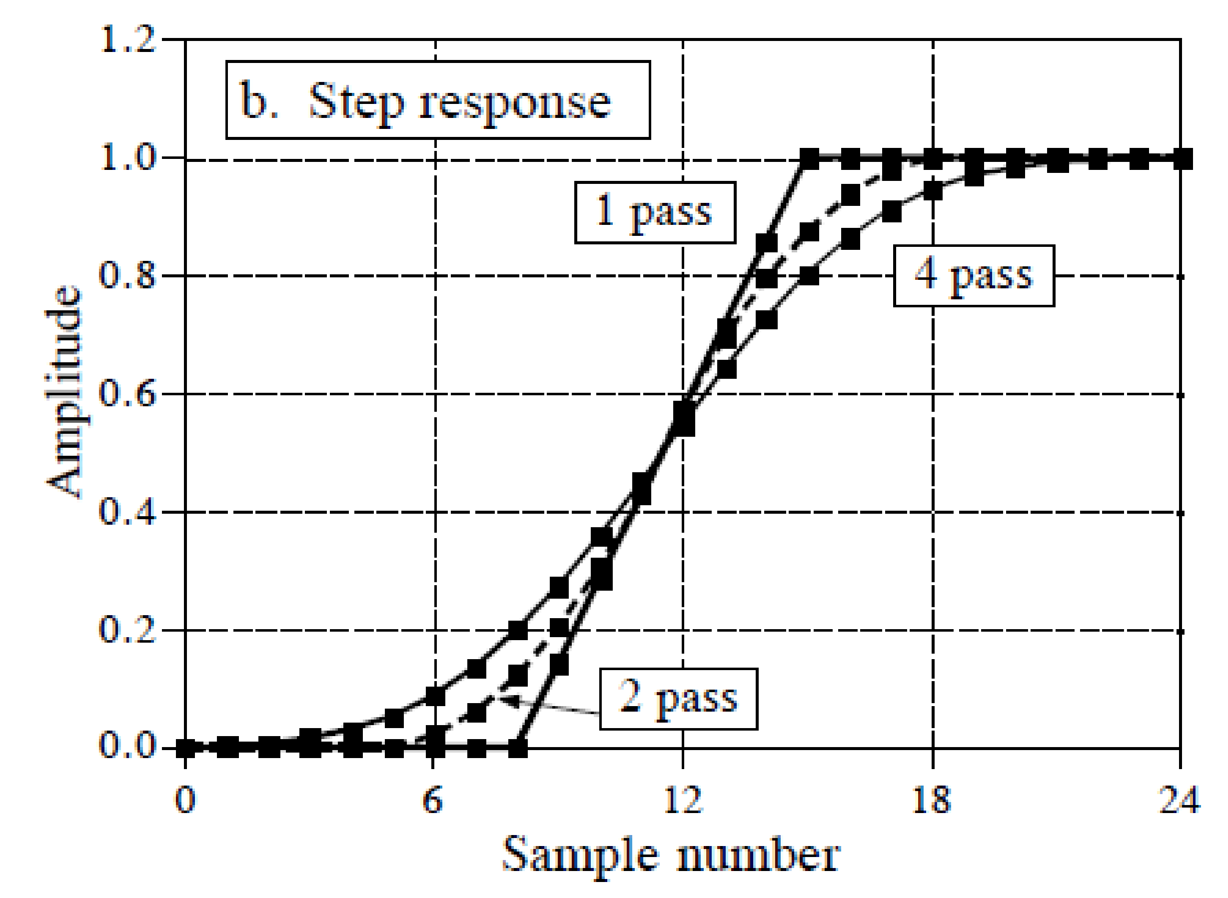
Multiple-pass moving average filter

- **Multiple-pass moving average filters** involve passing the input signal through a moving average filter **two or more times**.
- Figure shows the overall filter kernel resulting from one, two and four passes.
- Two passes are equivalent to using a ***triangular*** filter kernel (a rectangular filter kernel convolved with itself).
- After four or more passes, the equivalent filter kernel looks like a ***Gaussian*** (recall the Central Limit Theorem).



Multiple-pass moving average filter

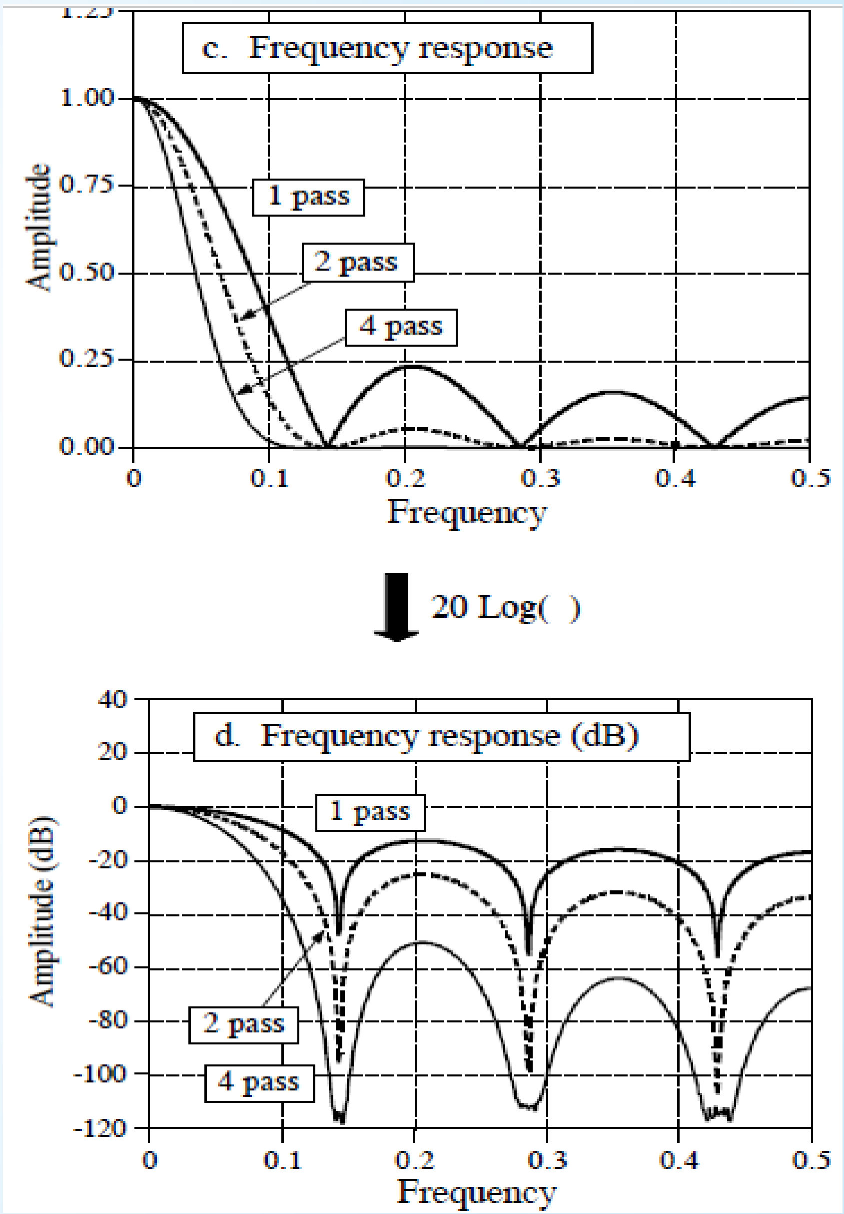
As shown in figure, multiple passes produce an "s" shaped step response, as compared to the straight line of the single pass.



Multiple-pass moving average filter

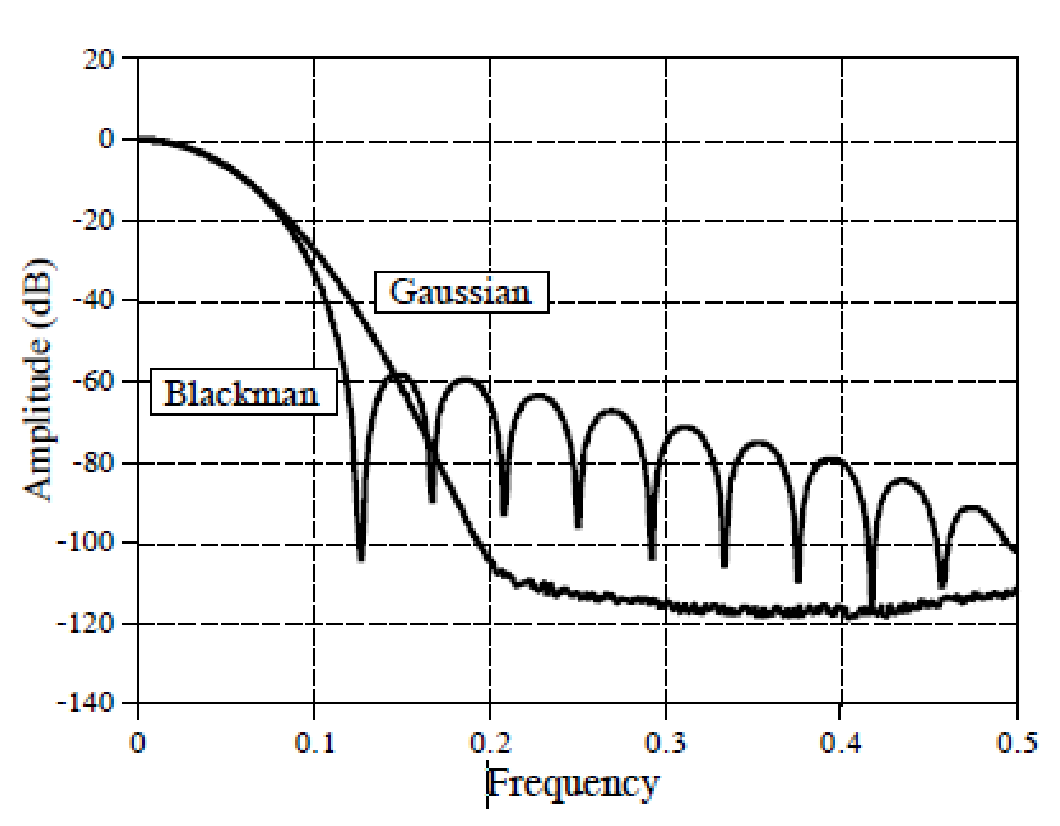
The frequency responses in (c) and (d) are given by H(f) *multiplied* by itself for each pass.

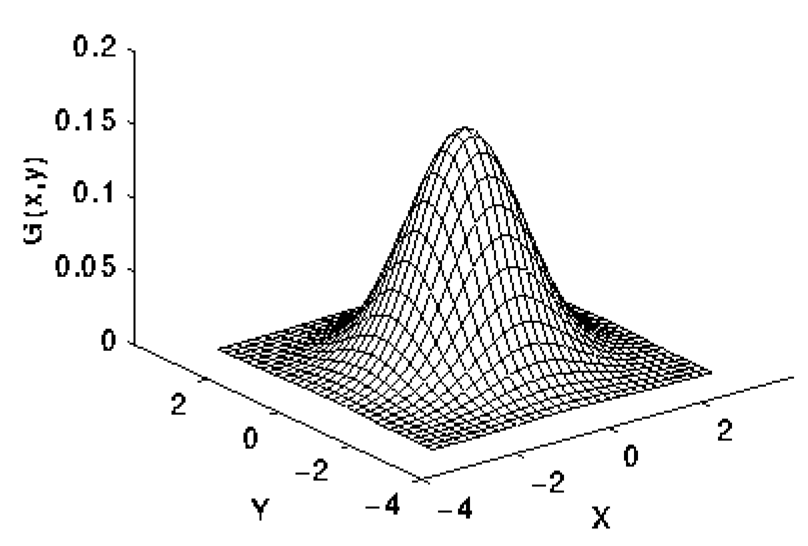
Each time domain **convolution** results in a **multiplication** of the frequency spectra.



Gaussian Filter

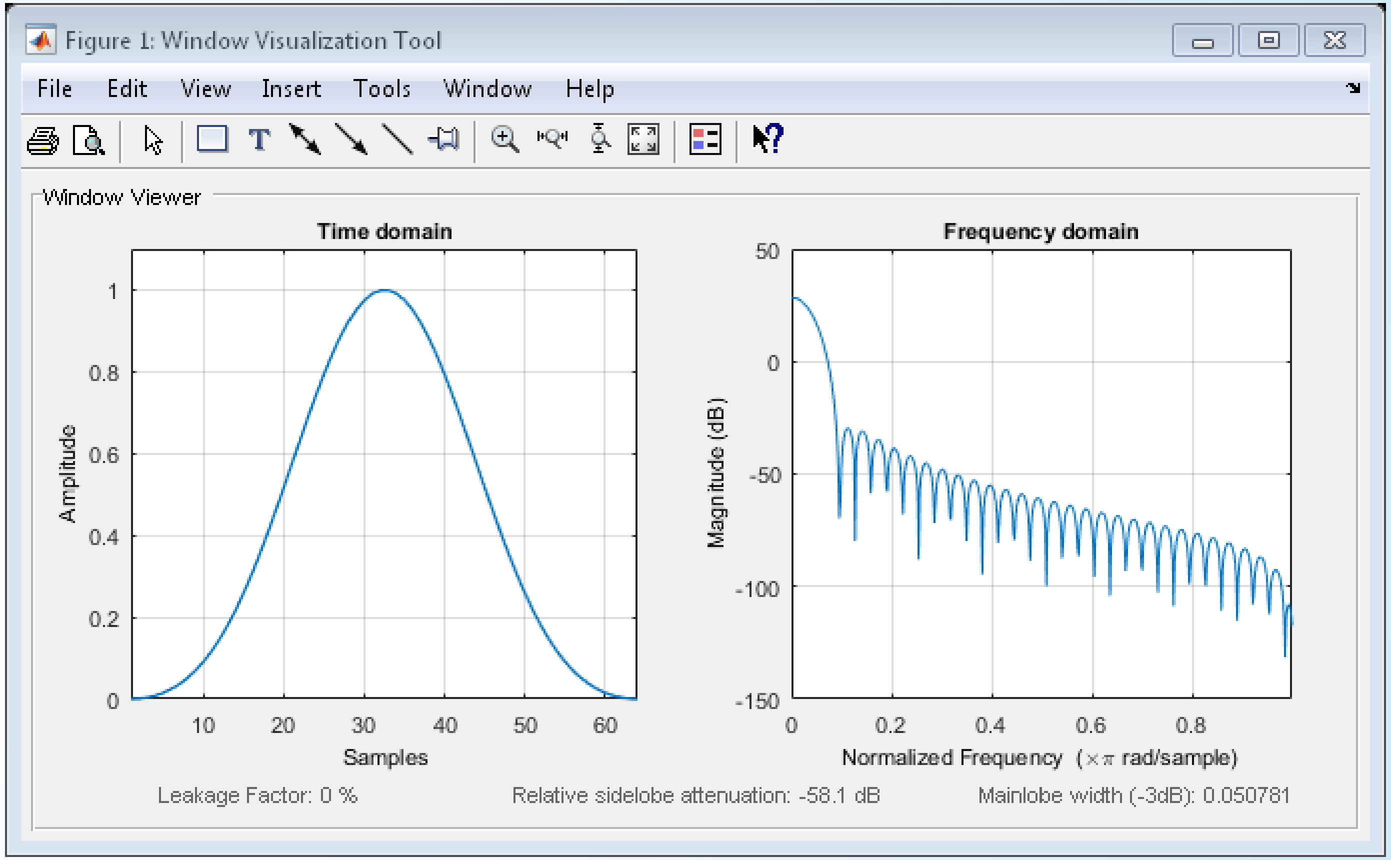
- When a pure **Gaussian** is used as a filter kernel, the frequency response is also a Gaussian
- The Gaussian is important because it is the impulse response of many natural and manmade systems.
- For example, a brief pulse of light entering a long fiber optic transmission line will exit as a Gaussian pulse, due to the different paths taken by the photons within the fiber.
- The Gaussian filter kernel is also used extensively in *image processing* because it has unique properties that allow fast two-dimensional convolutions





Blackman window

- The second frequency response in Figure corresponds to using a **Blackman window** as a filter kernel. (The term *window* has no meaning here; it is simply part of the accepted name of this curve).
- The exact shape of the Blackman window looks much like a Gaussian.



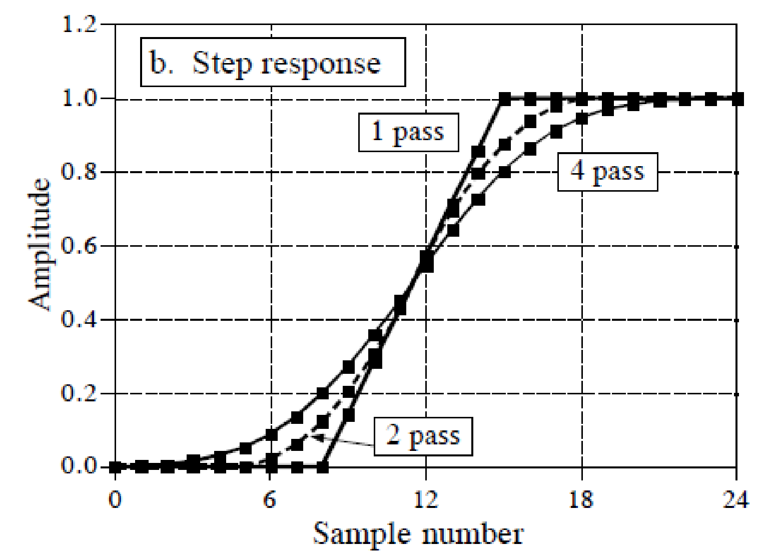
Comparative analysis of MA filters

How are these relatives of the moving average filter better than the moving average filter itself?

1) these filters have **better *stopband attenuation*** than the moving average filter.
2) the filter kernels *have* a **smaller amplitude near the ends.** Recall that each point in the output signal is a weighted sum of a group of samples from the input. If the filter kernel tapers, samples in the input signal that are farther away are given less weight than those close by.
3) the step responses are ***smooth* curves**, rather than the sharp straight line of the moving average.

Comparative analysis of MA filters

- The moving average filter and its relatives are all about the same at reducing random noise.
- The ambiguity lies in how the ***risetime*** of the step response is measured. If the risetime is measured from 0% to 100% of the step, the **moving average filter is the best** you can do, as previously shown.
- In comparison, measuring the risetime from 10% to 90% makes the **Blackman window *better* than the moving average filter**.
- Consider these filters equal in this parameter.



Comparative analysis of MA filters

The biggest difference in these filters is ***execution speed***.

- Using a recursive algorithm, the moving average filter will run like lightning in your computer. In fact, it is the ***fastest* digital filter** available.
- Multiple passes of the moving average will be correspondingly slower, but still very quick.
- In comparison, the **Gaussian and Blackman filters are slow**, because they must use convolution. Think a factor of ten times the number of points in the filter kernel (based on multiplication being about 10 times slower than addition). For example, expect a 100 point Gaussian to be 1000 times slower than a moving average using recursion.

Recursive Implementation

- A tremendous advantage of the moving average filter is that it can be implemented with an algorithm that is **very fast.**
- Imagine passing an input signal, x [ ], through a M=7 point moving average filter to form an output signal, *y* [ ].
- Now look at how two adjacent output points, *y* [50] and *y* [51], are calculated:

$$y[50] = x[47] + x[48] + x[49] + x[50] + x[51] + x[52] + x[53]$$

$$y[51] = x[48] + x[49] + x[50] + x[51] + x[52] + x[53] + x[54]$$

- If *y* [50] has already been calculated, the most *efficient* way to calculate *y* [51] is:

$$y[51] = y[50] + x[54] - x[47]$$

Recursive Implementation

- After the first point is calculated in y [ ], all of the other points can be found with only a single addition and subtraction per point. This can be expressed in the equation:

$$y[i] = y[i-1] + x[i+p] - x[i-q]$$

$$\text{where:} \quad p = (M-1)/2$$

$$q = p + 1$$

- Notice that this equation use two sources of data to calculate each point in the output: points from the **input** *and* **previously calculated points from the output**.
- This is called a **recursive equation**, meaning that the result of one calculation is used in ***future*** calculations.

Recursive Implementation

This algorithm is faster than other digital filters for several reasons.

1) There are only two computations per point, regardless of the length of the filter kernel.
2) Addition and subtraction are the only math operations needed, while most digital filters require time-consuming multiplication.
3) The indexing scheme is very simple. Each index is found by adding or subtracting integer constants that can be calculated before the filtering starts (i.e., p and q).
4) The entire algorithm can be carried out with integer representation. Depending on the hardware used, integers can be more than an order of magnitude faster than floating point.
5) Integer representation works *better* than floating point with this algorithm, in addition to being *faster*. There is **no round-off error** in the arithmetic.