CS 583 – Computational Audio -- Fall, 2021

Wayne Snyder
Computer Science Department
Boston University

Lecture 6: Sonic Spatialization and Reverberation

Smoothing and Digital Filters

Convolution

Convolution Reverb

Spatialization (especially stereo)

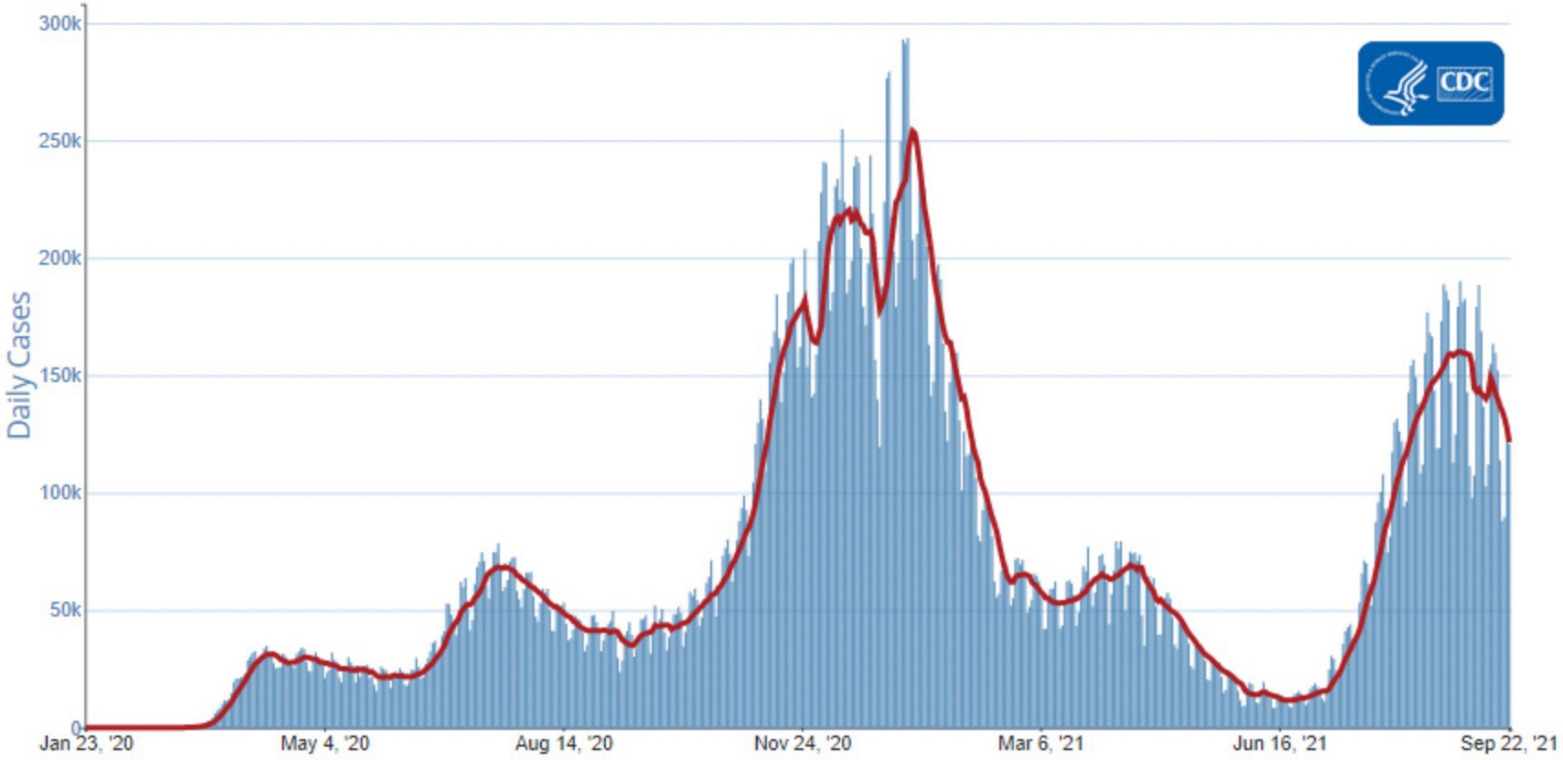
The image part with relationship ID rId2 was not found in the file.

Computer Science

Smoothing Algorithms and Digital Filters

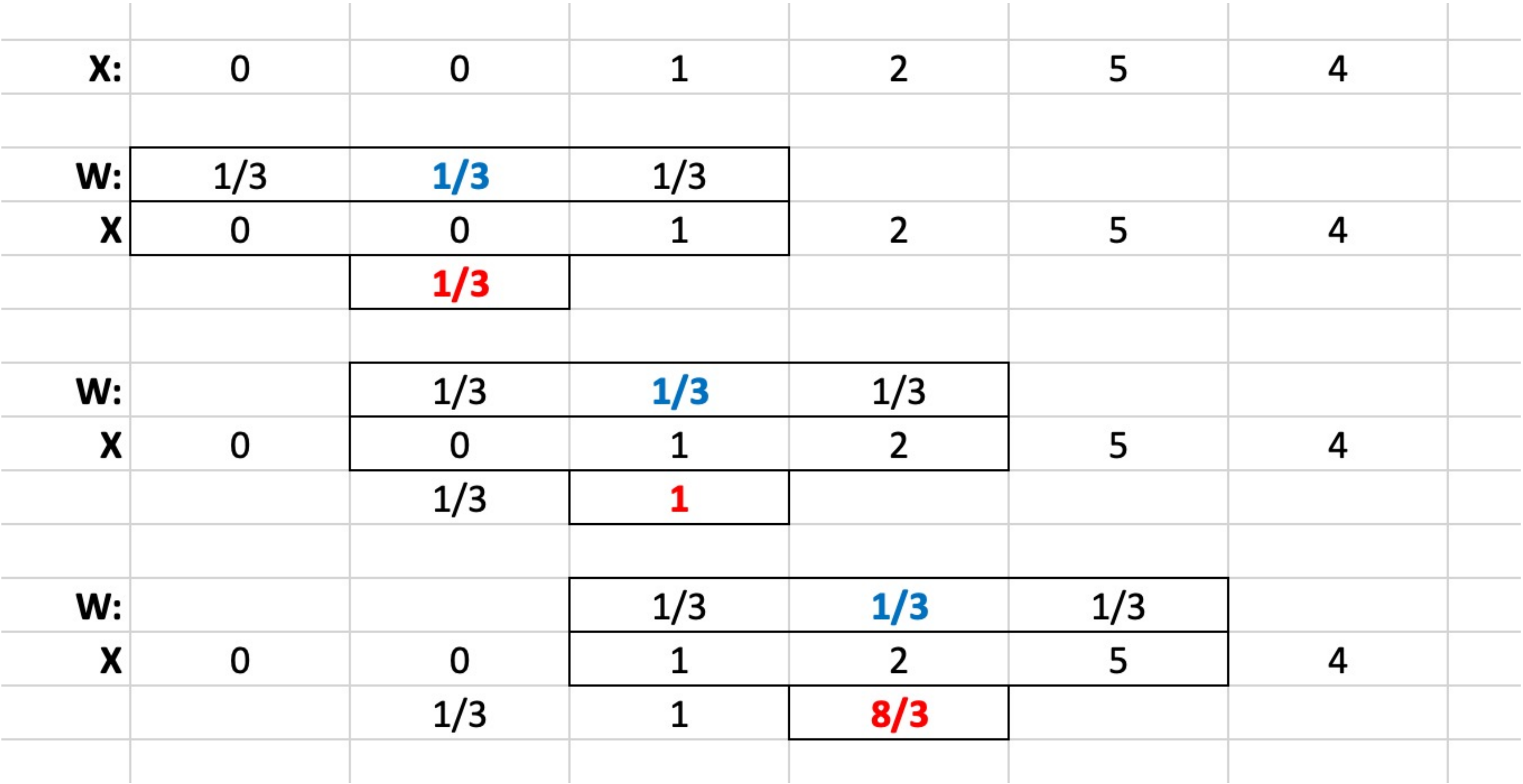
Smoothing refers to the processing of reducing small variations in a signal and emphasizing longer-term trends; in signal processing, this is called a filter, which supresses particular frequencies (here, higher frequencies) in a signal.

For example, Covid data is almost always presented with a “seven-day moving average”:



Smoothing Algorithms and Digital Filters

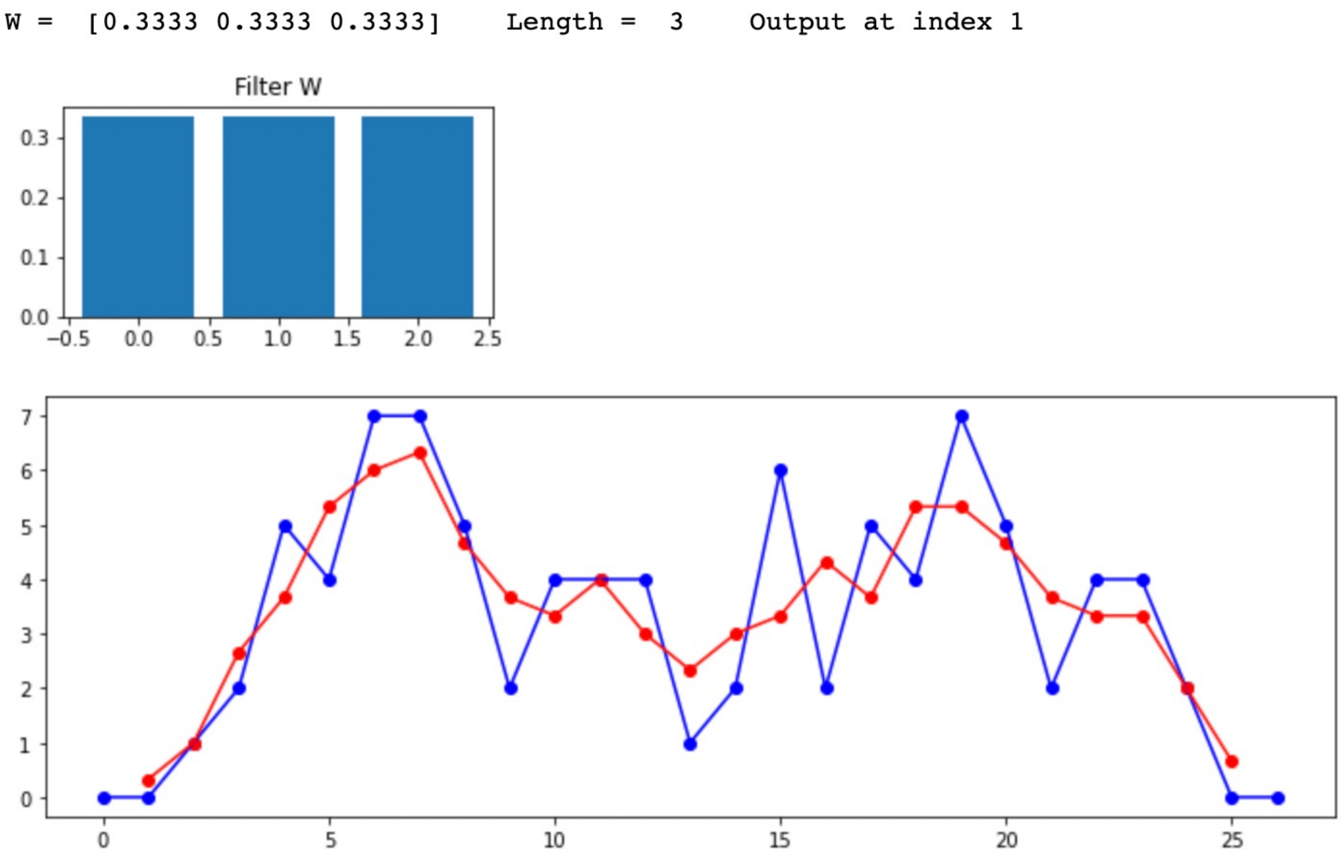
Example: Let’s look at a “3-sample moving average” filter, represented by two signals X (the signal to be smoothed) and W (the “weights” of the filter – just another signal!):



X:	0	0	1	2	5	4
W:	1/3	**1/3**	1/3			
X	0	0	1	2	5	4
		1/3				
W:		1/3	**1/3**	1/3		
X	0	0	1	2	5	4
		1/3	**1**			
W:			1/3	**1/3**	1/3	
X	0	0	1	2	5	4
		1/3	1	**8/3**		

Smoothing Algorithms and Digital Filters

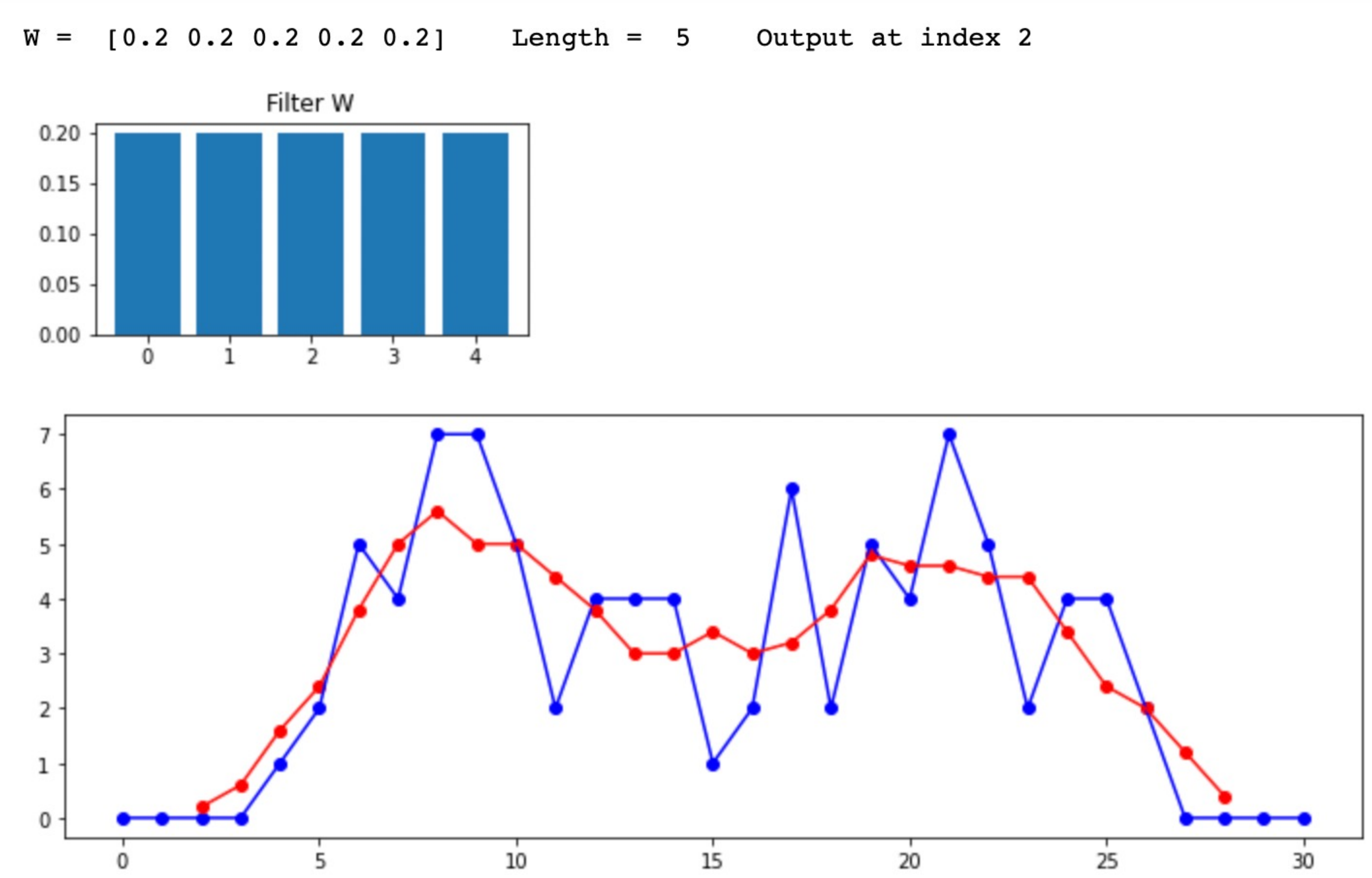
Let’s look at a “3-sample moving average” filter, represented by two signals X (the signal to be smoothed) and W (the “weights” of the filter – just another signal!):



```
W =  [0.3333 0.3333 0.3333]    Length =  3    Output at index 1
```

Smoothing Algorithms and Digital Filters

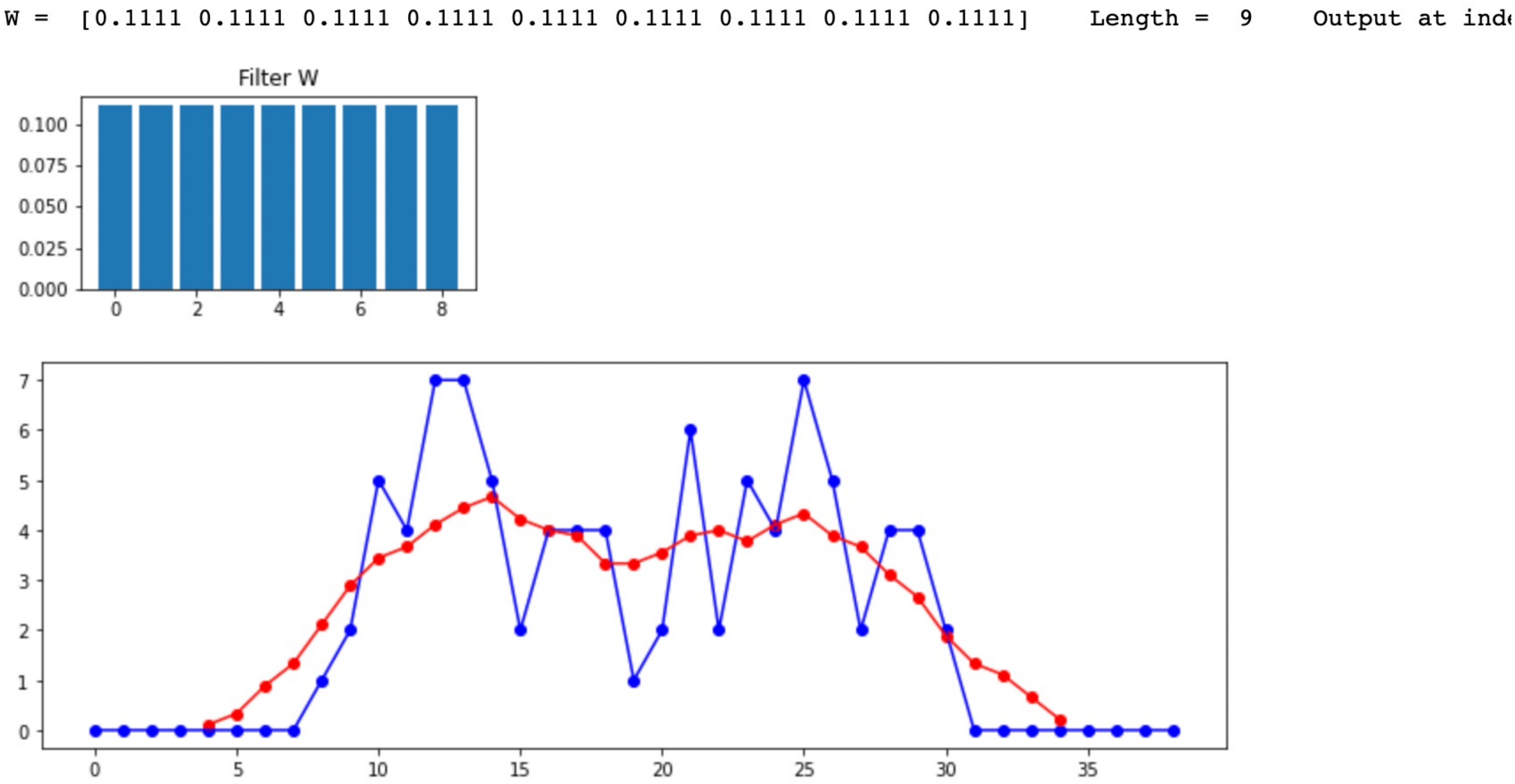
If we increase the length of the filter, we get more smoothing:



```
W =  [0.2 0.2 0.2 0.2 0.2]    Length =  5    Output at index 2
```

Smoothing Algorithms and Digital Filters

If we increase the length of the filter, we get more smoothing:



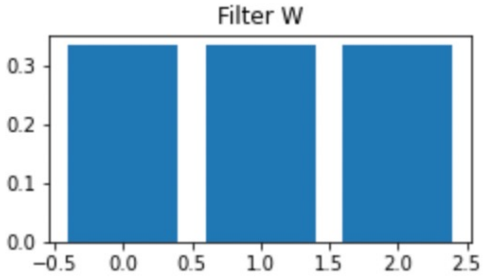
```
W =  [0.1111 0.1111 0.1111 0.1111 0.1111 0.1111 0.1111 0.1111 0.1111]    Length =  9    Output at inde
```

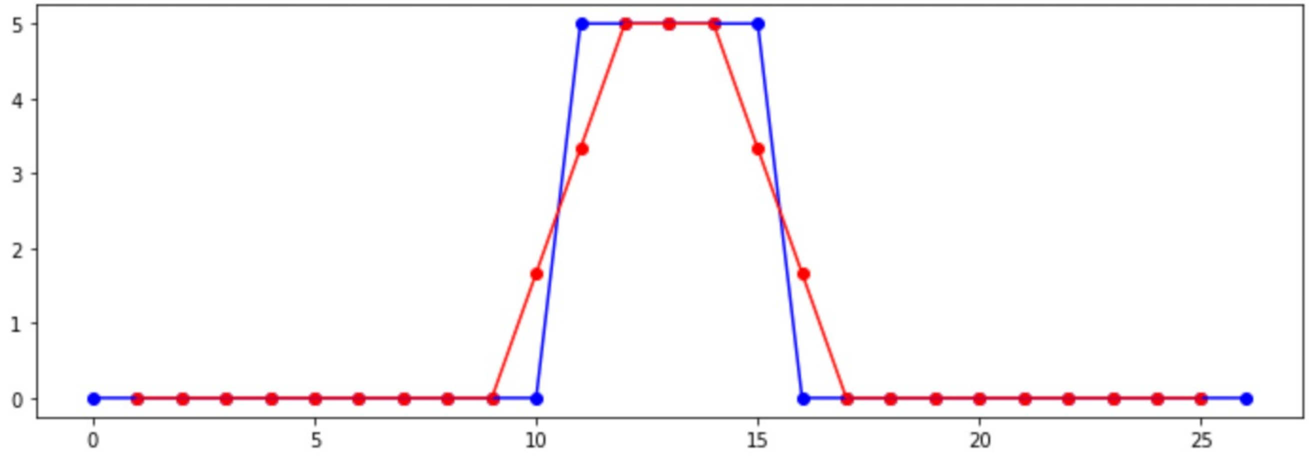
Smoothing Algorithms and Digital Filters

The image part with relationship ID rId15 was not found in the file.

Depending on the values in the filter W = w_0, w_1, …., w_{n-1}, and the location of the "output value," we get different kinds of smoothing:

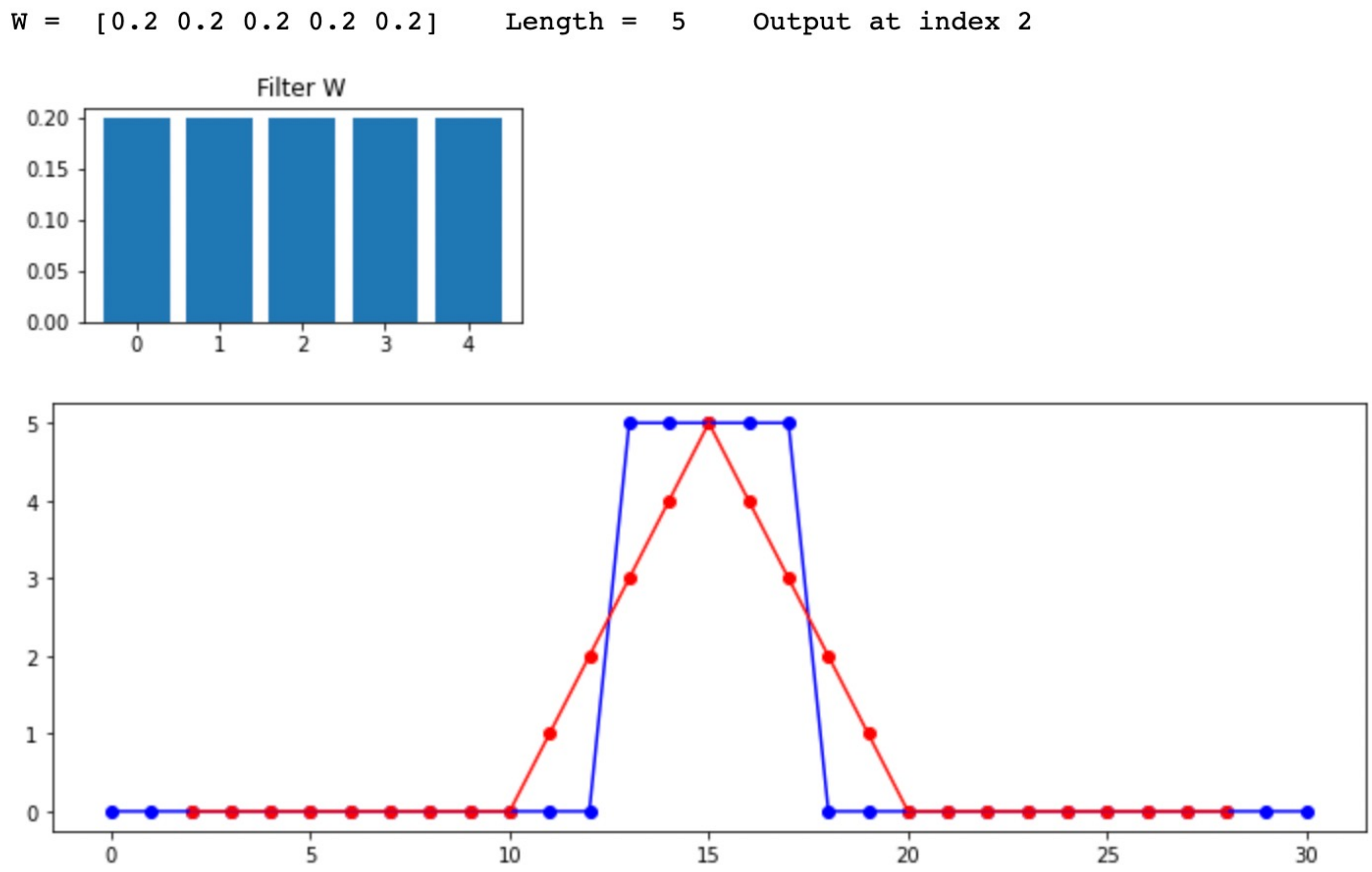
```
W =  [0.3333 0.3333 0.3333]    Length =  3    Output at index 1
```





Smoothing Algorithms and Digital Filters

Depending on the values in the filter $W = w_0, w_1, ..., w_{n-1},$ and the location of the "output value," we get different kinds of smoothing:

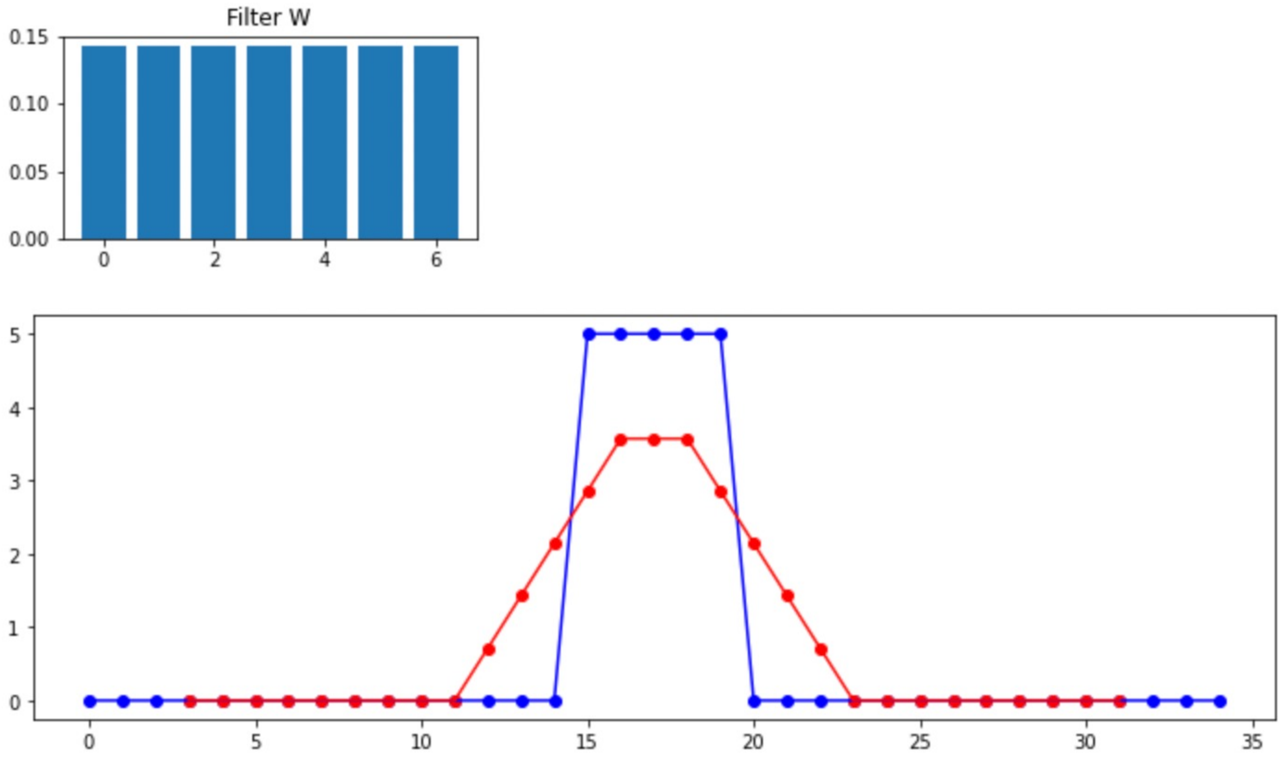


```
W =  [0.2 0.2 0.2 0.2 0.2]    Length =  5    Output at index 2
```

Smoothing Algorithms and Digital Filters

Depending on the values in the filter W = $w_0, w_1, \ldots., w_{n-1},$ and the location of the "output value," we get different kinds of smoothing:

```
W =  [0.1429 0.1429 0.1429 0.1429 0.1429 0.1429 0.1429]    Length =  7    Output at index 3
```



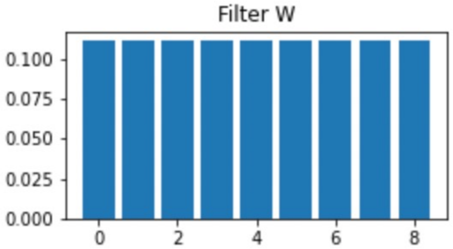
Smoothing Algorithms and Digital Filters

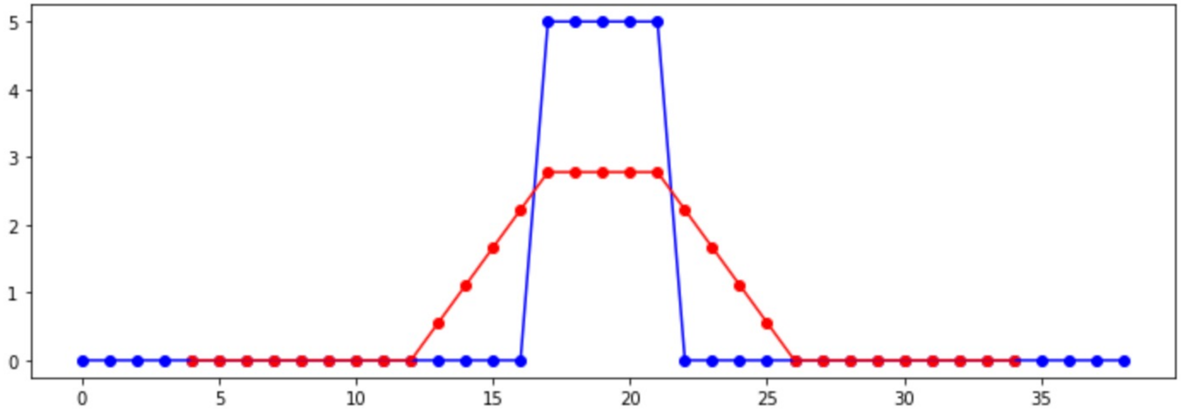
The image part with relationship ID rId15 was not found in the file.

Computer Science

Depending on the values in the filter W = w_0, w_1, …., $w_{n-1,}$ and the location of the "output value," we get different kinds of smoothing:

```
W =  [0.1111 0.1111 0.1111 0.1111 0.1111 0.1111 0.1111 0.1111 0.1111]    Length =  9    Output at index 4
```

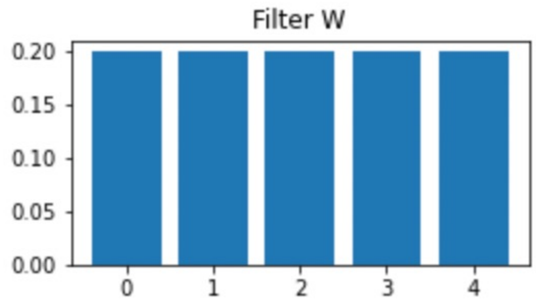


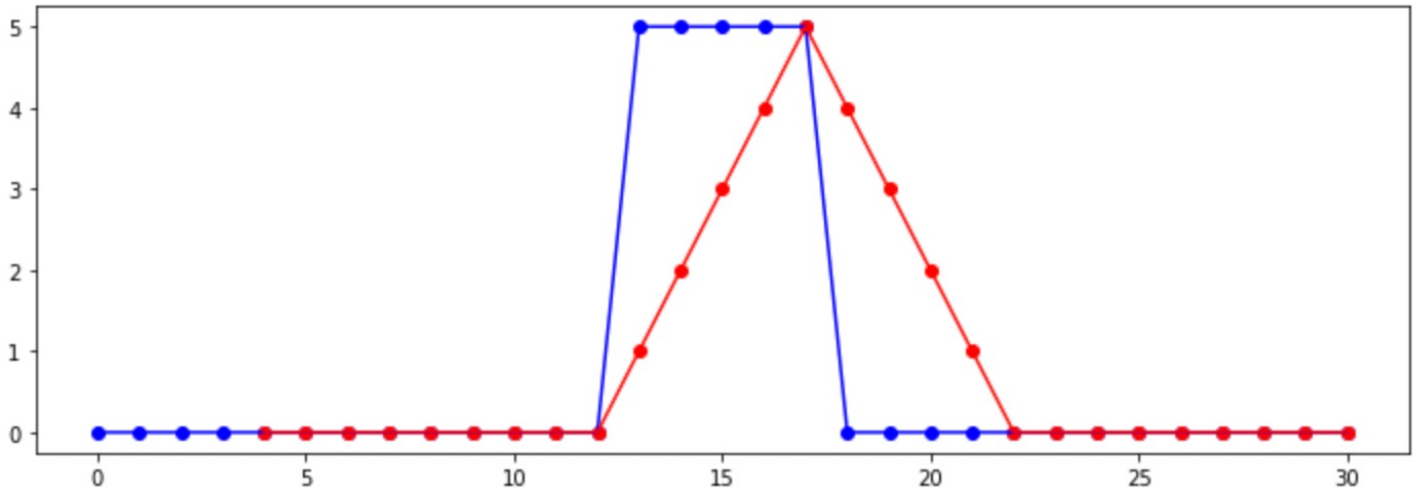


Smoothing Algorithms and Digital Filters

Depending on the values in the filter W = w_0, w_1, …., w_{n-1}, and the location of the "output value," we get different kinds of smoothing:

```
W =  [0.2 0.2 0.2 0.2 0.2]    Length =  5    Output at index 4
```



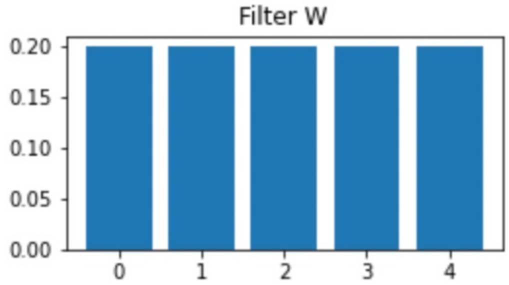


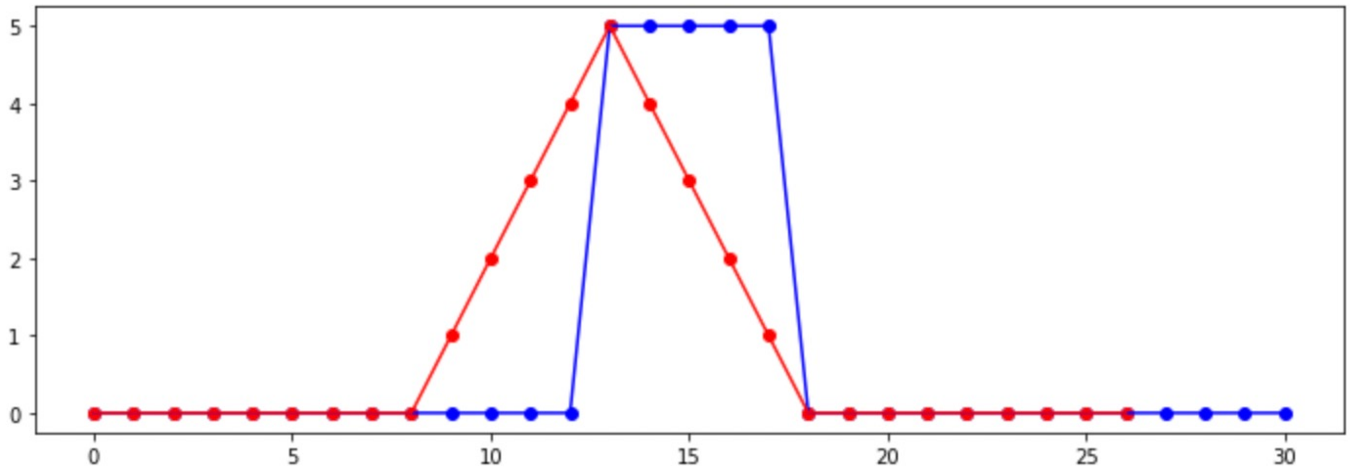
Smoothing Algorithms and Digital Filters

The image part with relationship ID rId15 was not found in the file.

Depending on the values in the filter $W = w_0, w_1, \ldots., w_{n-1},$ and the location of the "output value," we get different kinds of smoothing:

```
W =  [0.2 0.2 0.2 0.2 0.2]    Length =  5    Output at index 0
```

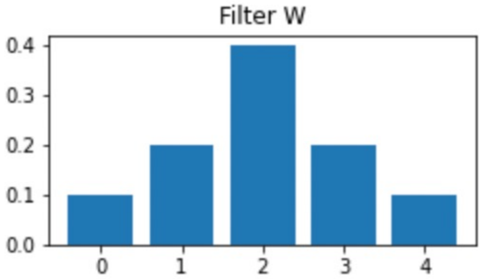


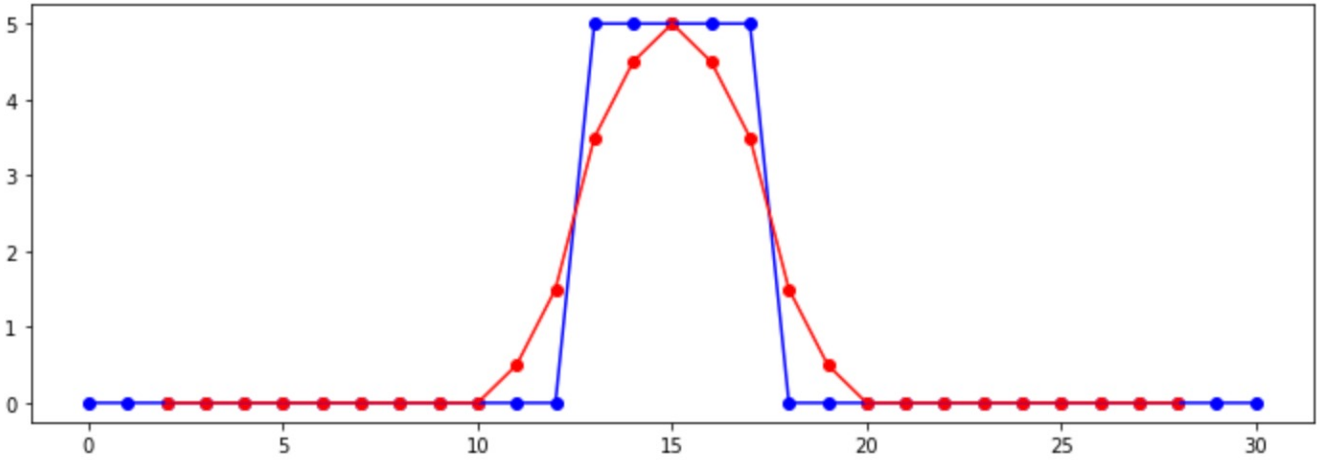


Smoothing Algorithms and Digital Filters

Depending on the values in the filter $W = w_0, w_1, \ldots., w_{n-1},$ and the location of the "output value," we get different kinds of smoothing:

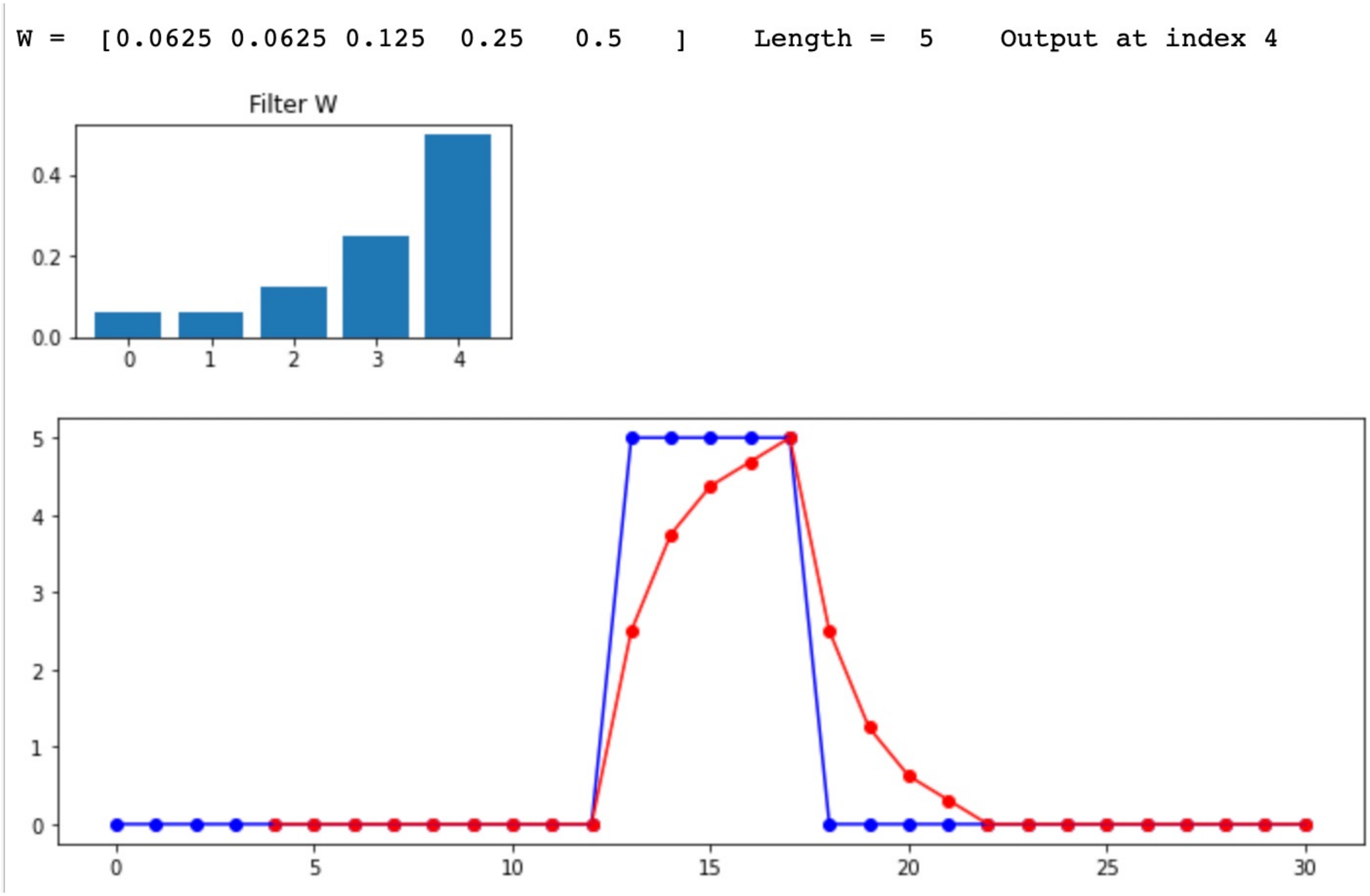
```
W =  [0.1 0.2 0.4 0.2 0.1]    Length =  5    Output at index 2
```





Smoothing Algorithms and Digital Filters

Depending on the values in the filter W = w_0, w_1, …., w_{n-1}, and the location of the “output value,” we get different kinds of smoothing:



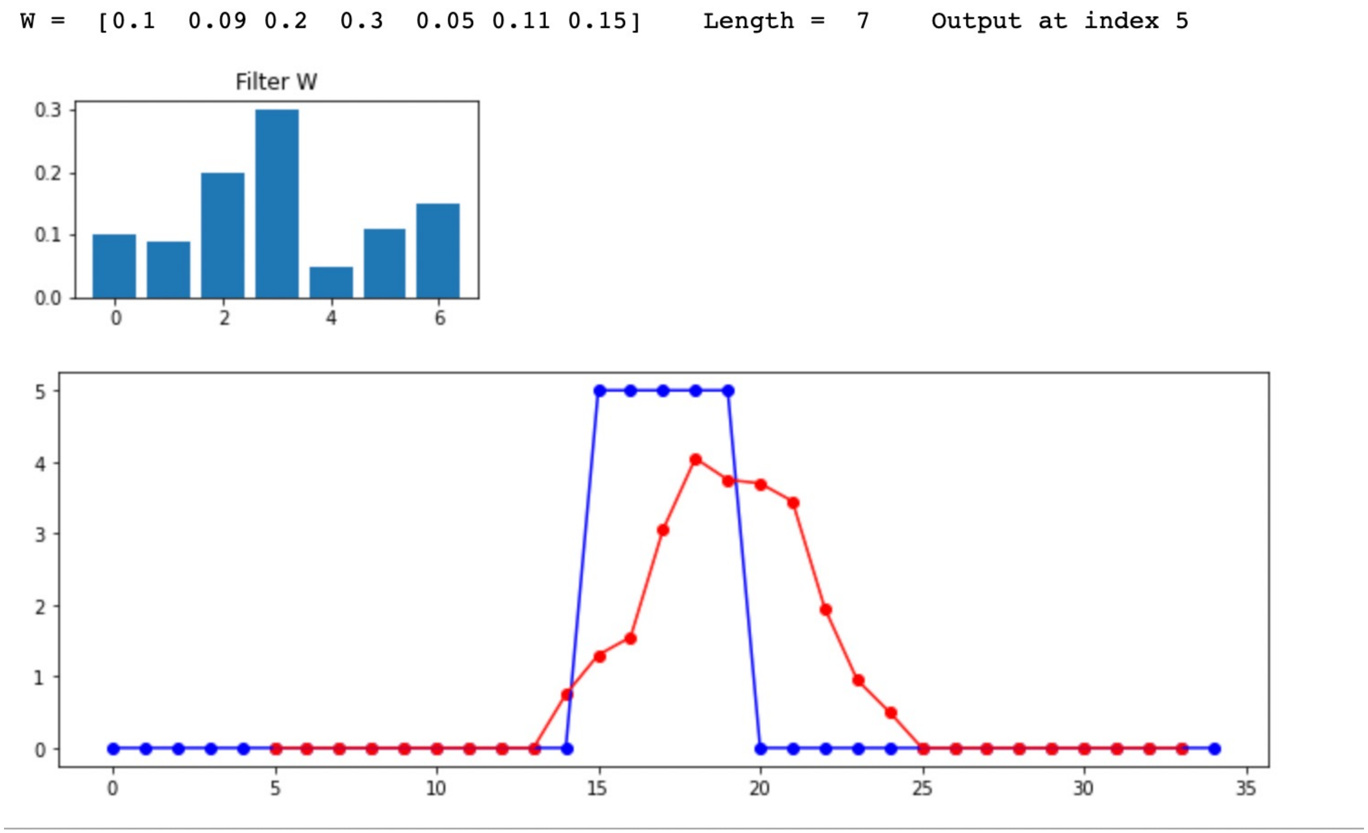
```
W =  [0.0625 0.0625 0.125  0.25   0.5   ]   Length =  5   Output at index 4
```

Smoothing Algorithms and Digital Filters

The image part with relationship ID rId15 was not found in the file.

Computer Science

Depending on the values in the filter W = $w_0, w_1, ..., w_{n-1},$ and the location of the "output value," we get different kinds of smoothing:



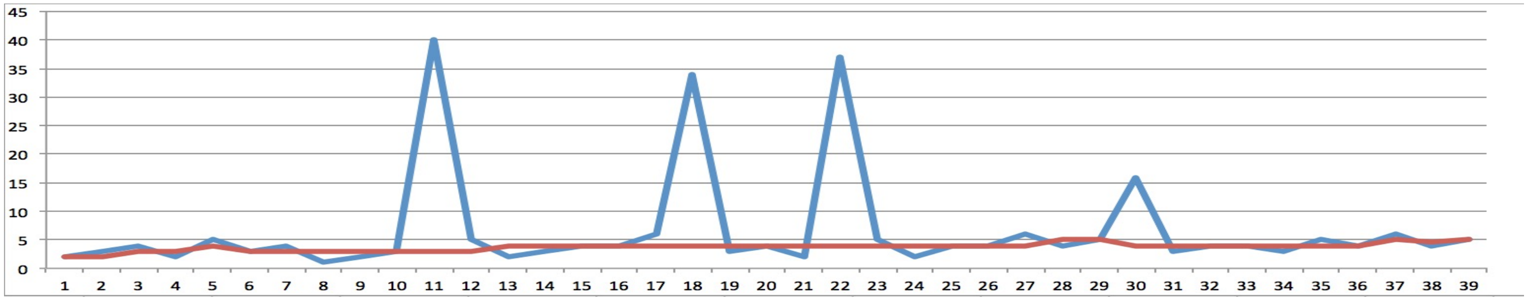
```
W =  [0.1  0.09 0.2  0.3  0.05 0.11 0.15]    Length =  7    Output at index 5
```

Smoothing Algorithms and Digital Filters

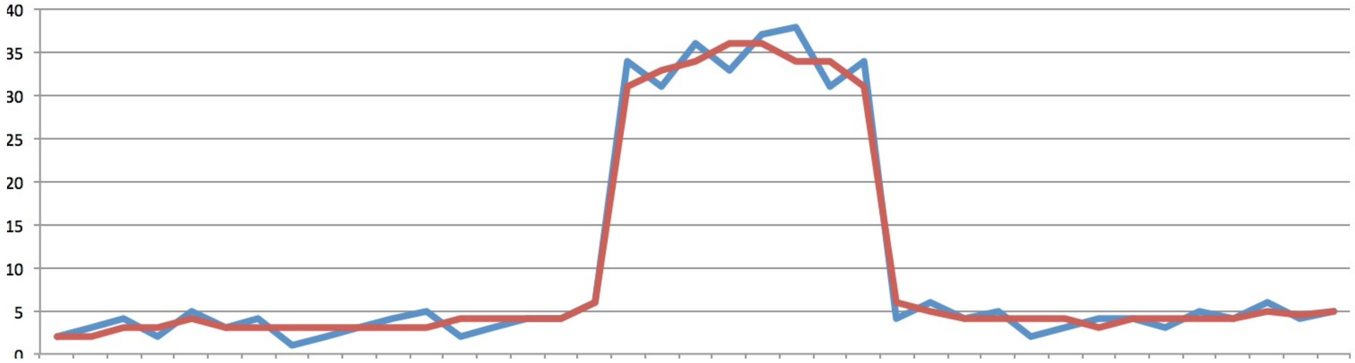
A similar technique which does not exactly fit this “weighted moving average” scheme is median smoothing:

$$Y[i] = \text{median}(X[i-k], ..., X[i], ..., X[i+k])$$

This will ignore extreme outliers:



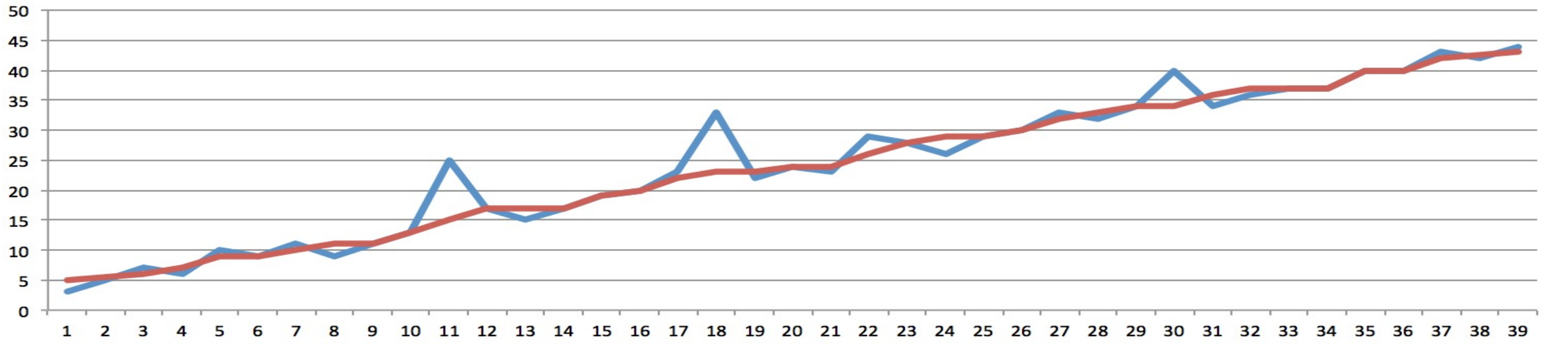
But otherwise acts similarly with rectangular smoothing, but with faster transitions to plateaus:

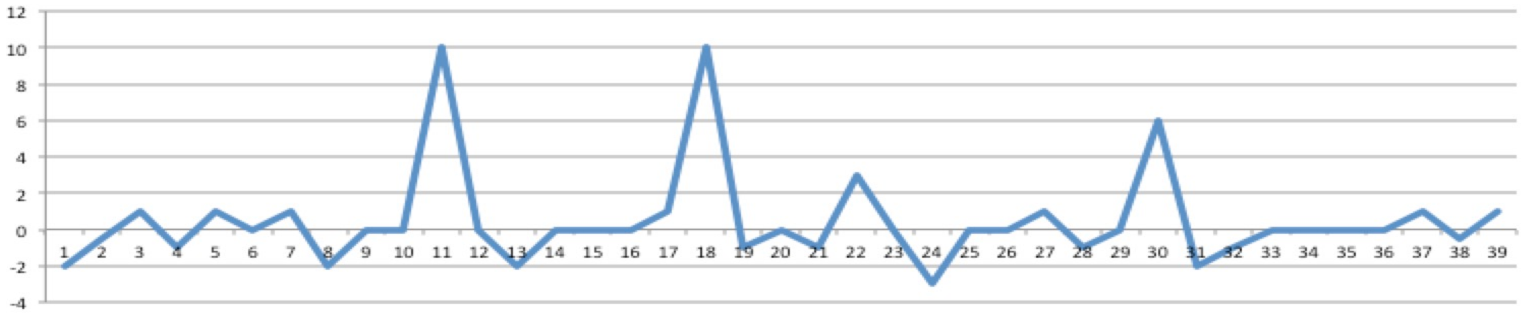


Smoothing Algorithms and Digital Filters

The image part with relationship ID rId15 was not found in the file.

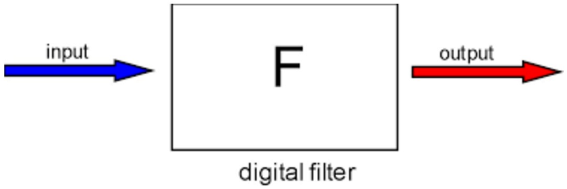
A useful variation of smoothing (sometimes called "detrending" is to **subtract the smoothed signal from the original**; in the case of median smoothing, this will preserve the peaks and ignore the valleys! It can be useful in "peak picking" techniques which we will make use of in a week or two:





Smoothing Algorithms and Digital Filters

In digital signal processing, the general process of modifying an input signal (e.g., by smoothing) is referred to as filtering:



Filters can be used to do a variety of tasks:

Amplitude modification: reduction, increase of amplitude

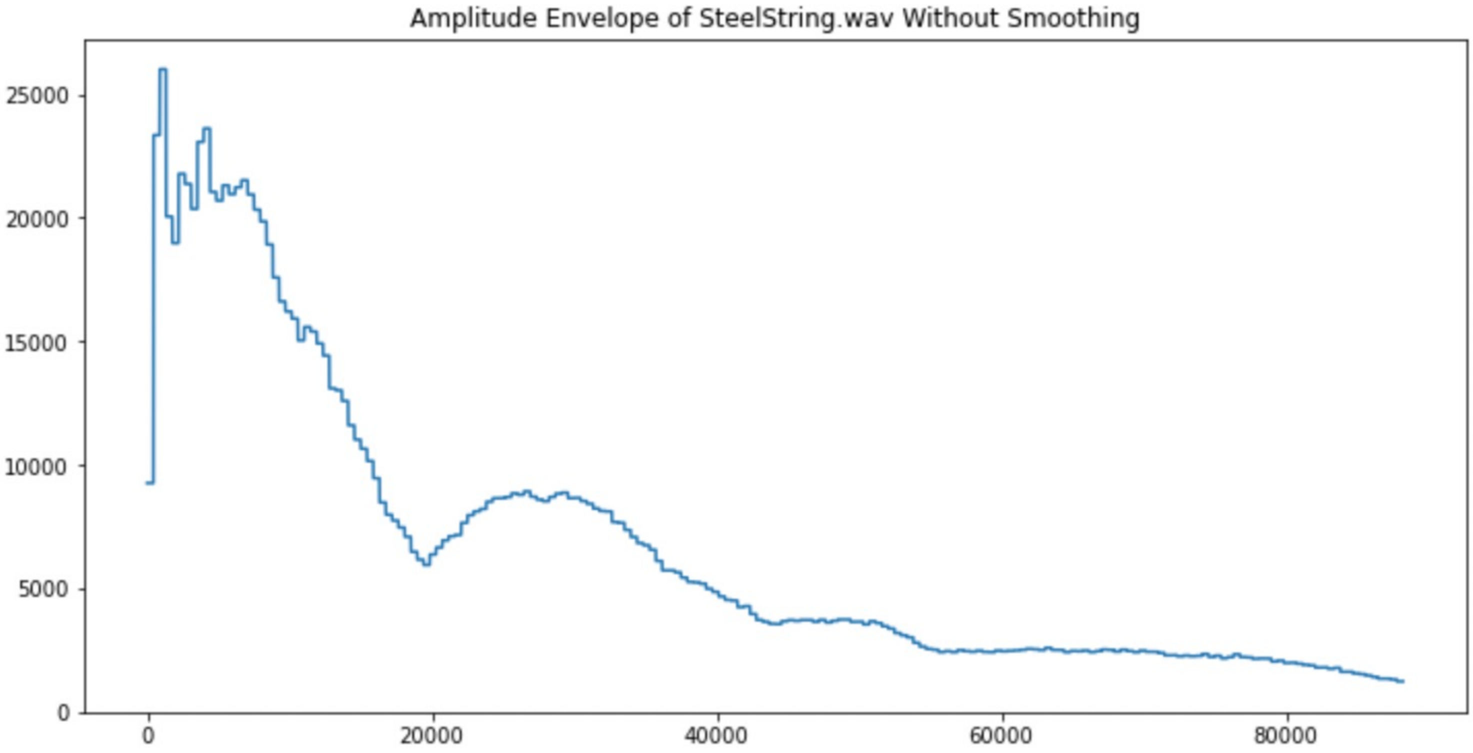
Frequency modification:
Low-pass filter (allow only frequencies below some threshold)
High-pass filter (allow only frequencies above some threshold)
Band-pass filter (allow only frequencies in some band)

Many miscellaneous tasks can be thought of as filtering:

- Noise reduction
- Tone controls
- Reverberation
- Vowel formation in vocal track

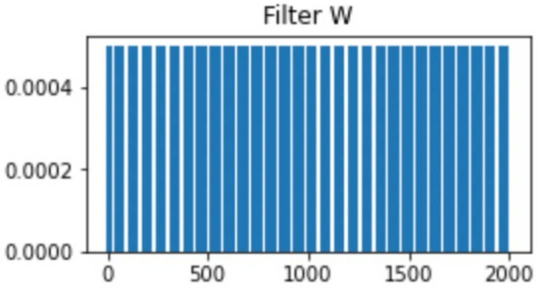
Smoothing Algorithms and Digital Filters

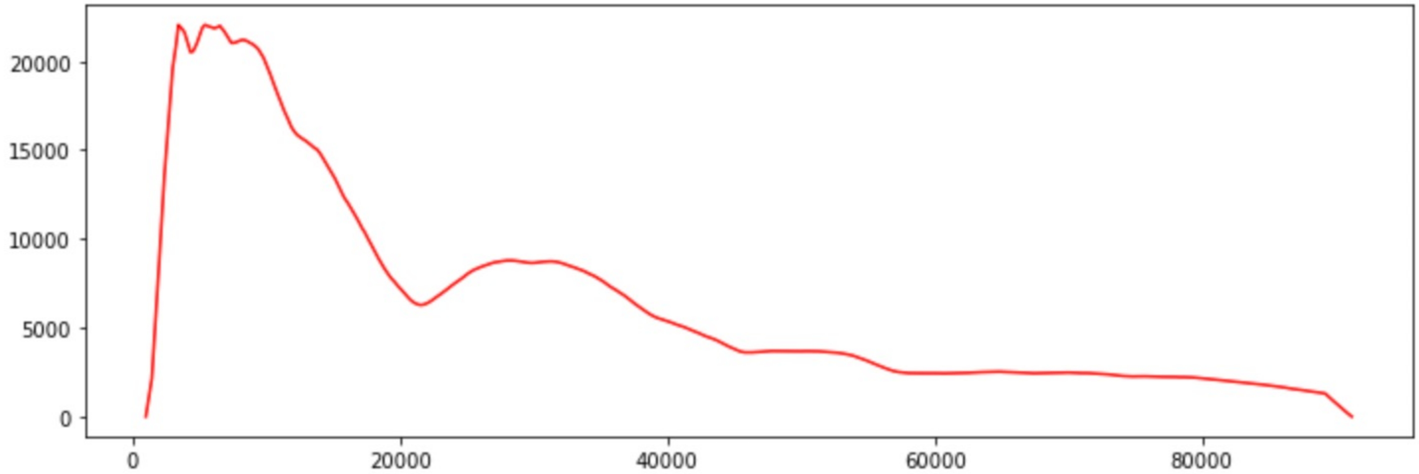
For example, the technique for calculating the amplitude envelope of the SteelString.wav file produces a jagged outline which may not produce the best results:



Smoothing Algorithms and Digital Filters

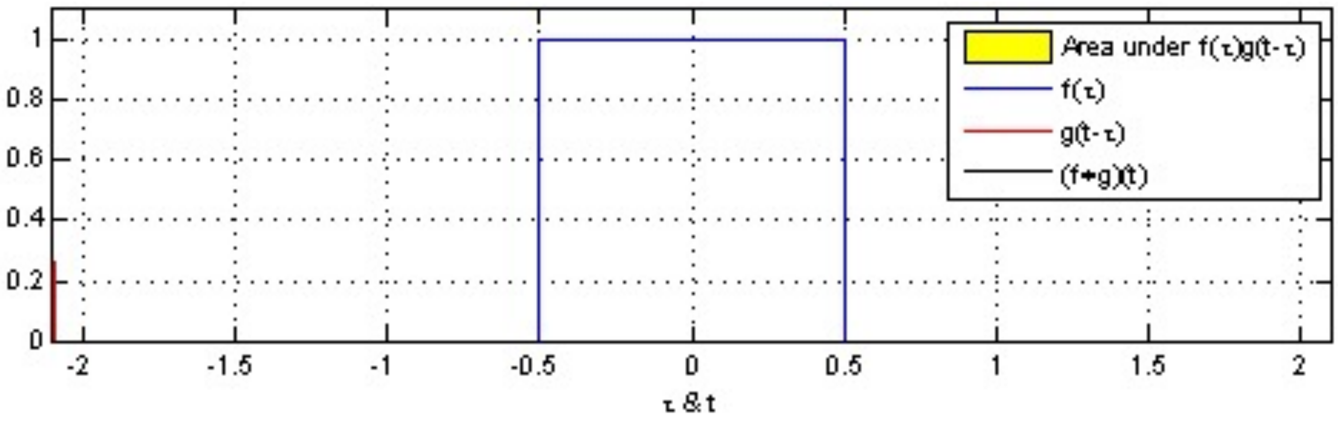
Here is the same envelope, with smoothing, which will produce smoother transitions and less noise in the result:





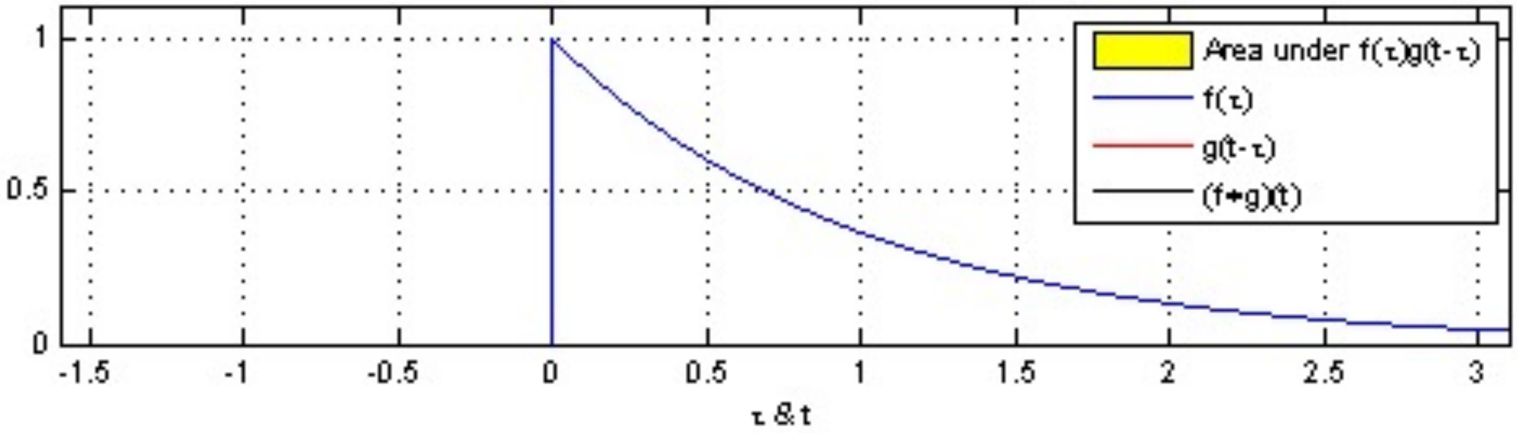
Convolution

Convolution is a very similar operation which shows up in lots of places in audio and video processing and in machine learning. It can be defined on continuous signals as well:



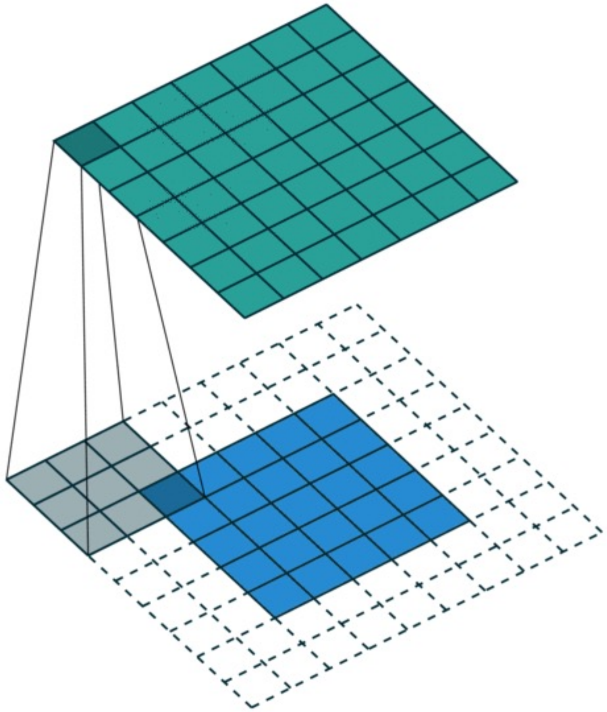
Convolution

Convolution is a very similar operation which shows up in lots of places in audio and video processing and in machine learning:



Convolution

Convolution is a very similar operation which shows up in lots of places in audio and video processing and in machine learning; In image processing, convolution is defined in 2 dimensions, with the “filter” being called a “kernel” but otherwise behaving about as you would expect:



Convolution vs Moving Average Filters

What is the difference between convolution and a moving average filter?

They are the same, except in convolution you REVERSE the filter W before applying it to the signal.

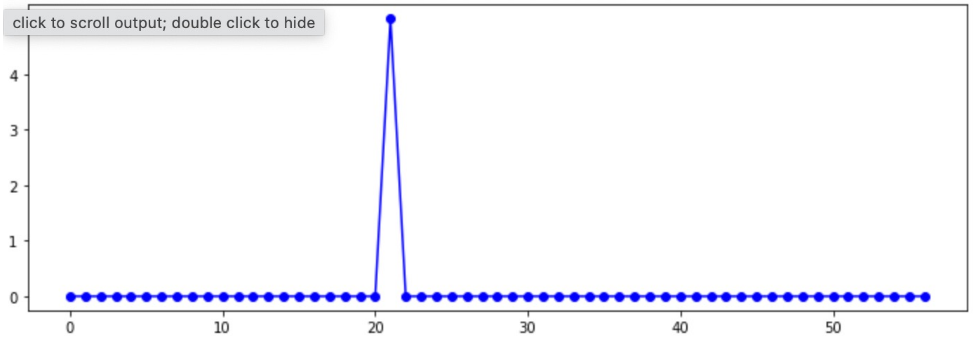
WHY?

The basic idea is that this explains how time series work, and is most obvious when considering the phenomenon of reverberation or echoes.

Convolution vs Moving Average Filters

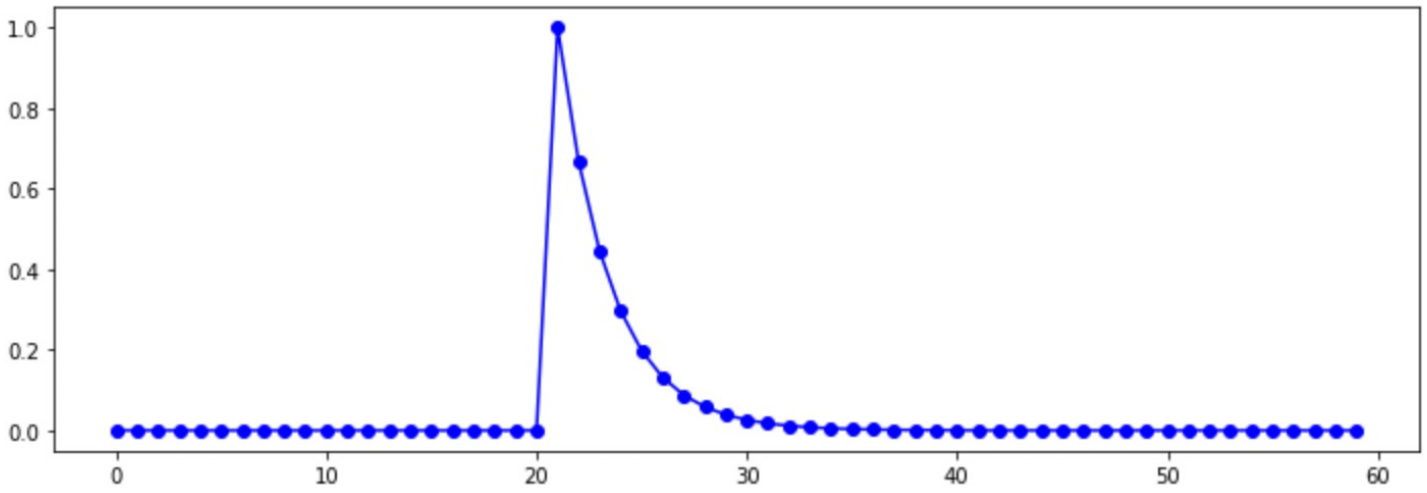
.

Consider a sharp, almost instantaneous sound, like a clap or a fire cracker; outside, you might just hear the single, instantaneous sound:



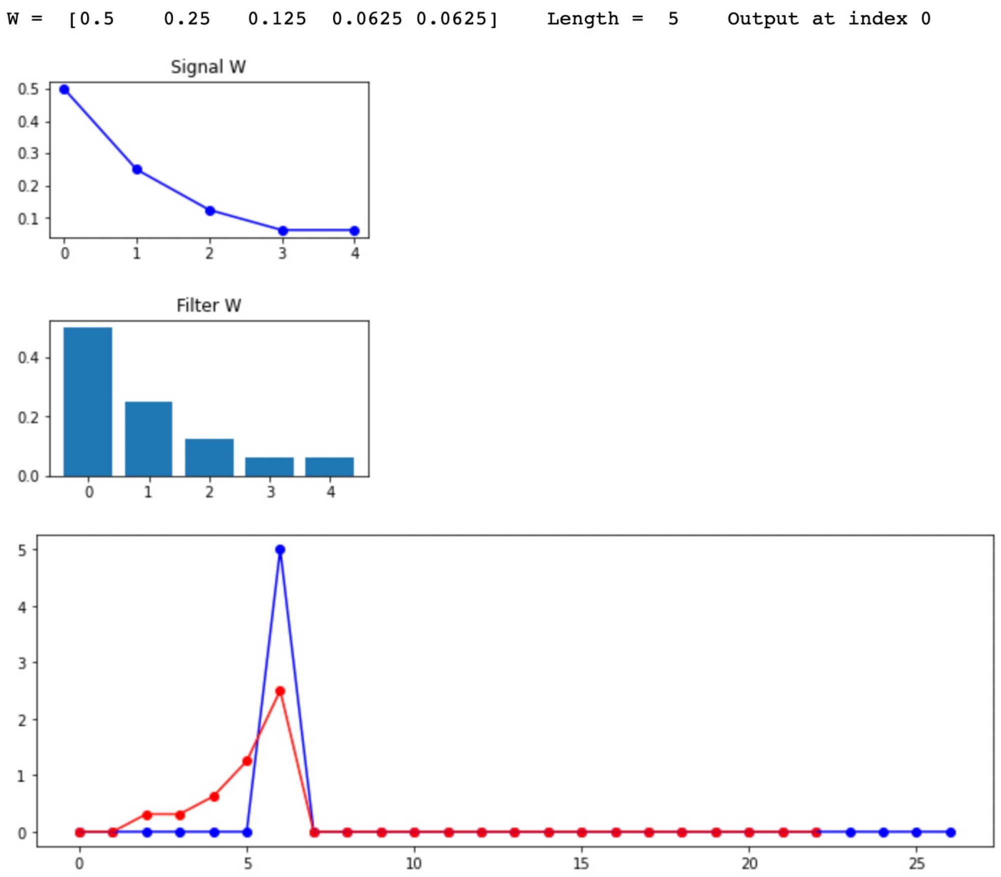
Convolution vs Moving Average Filters

But in a room, you will hear reverberations of the clap as it bounces around the room and returns to your ears, basically following an exponential decrease in the amplitude as the sound dies out:



Convolution vs Moving Average Filters

We can then think of this signal as a pattern for how sound reverberates in that room, and use it as a filter, but to do so, we have to REVERSE the signal, since if you don't, it doesn't model how the reverberation works in time:

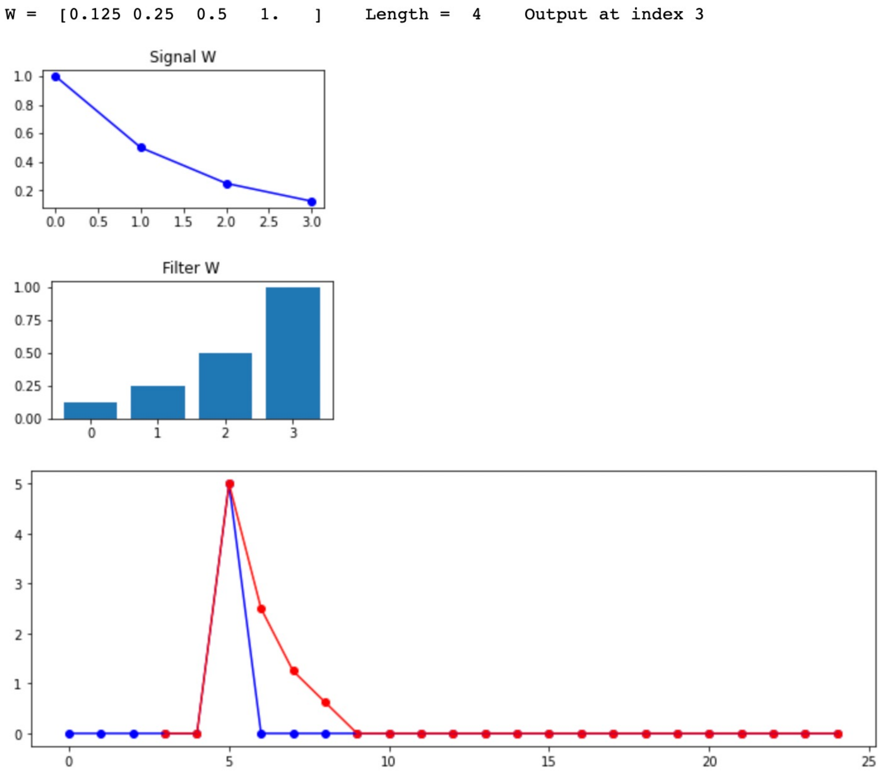


```
W =  [0.5    0.25   0.125  0.0625 0.0625]    Length =  5    Output at index 0
```

Convolution vs Moving Average Filters

We can then think of this signal as a pattern for how sound reverberates in that room, and use it as a filter, but to do so, we have to REVERSE the signal, since if you don't, it doesn't model how the reverberation works in time.

So let's reverse it, so that the moving average filter becomes convolution:

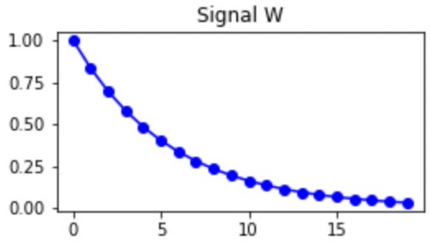


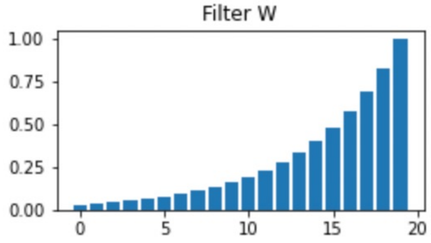
```
W =  [0.125 0.25  0.5   1.   ]    Length =  4    Output at index 3
```

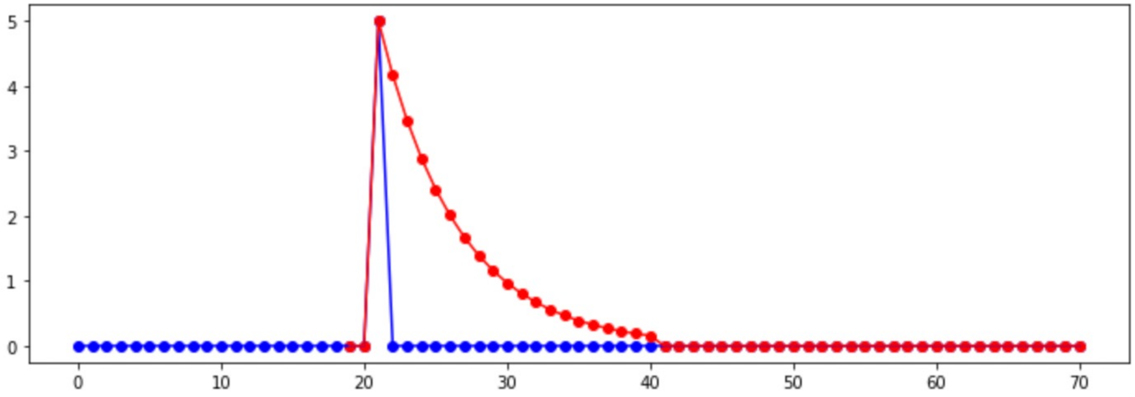
Convolution vs Moving Average Filters

The image part with relationship ID rId15 was not found in the file.

Now we can model how reverberation acts with a clap:

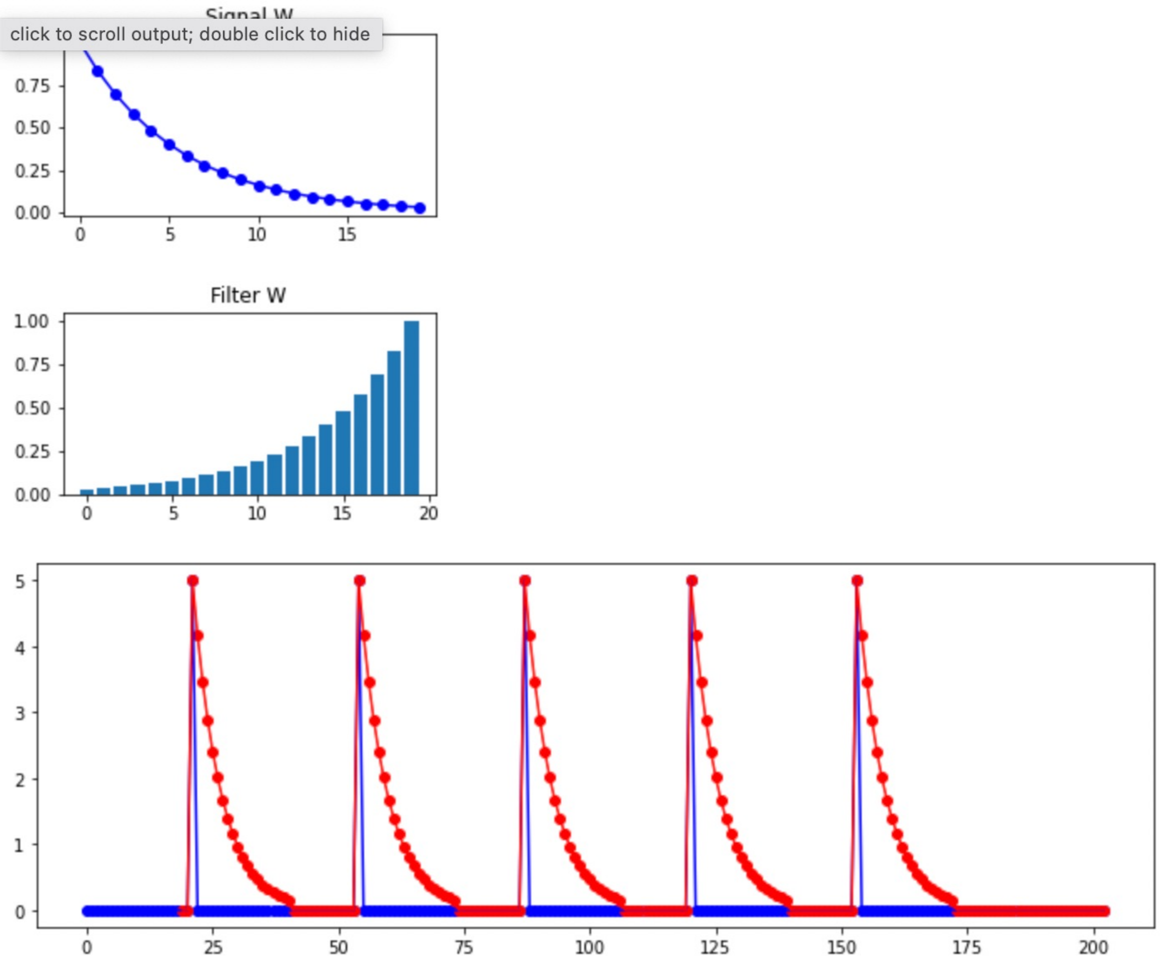






Convolution vs Moving Average Filters

Or repeated claps:

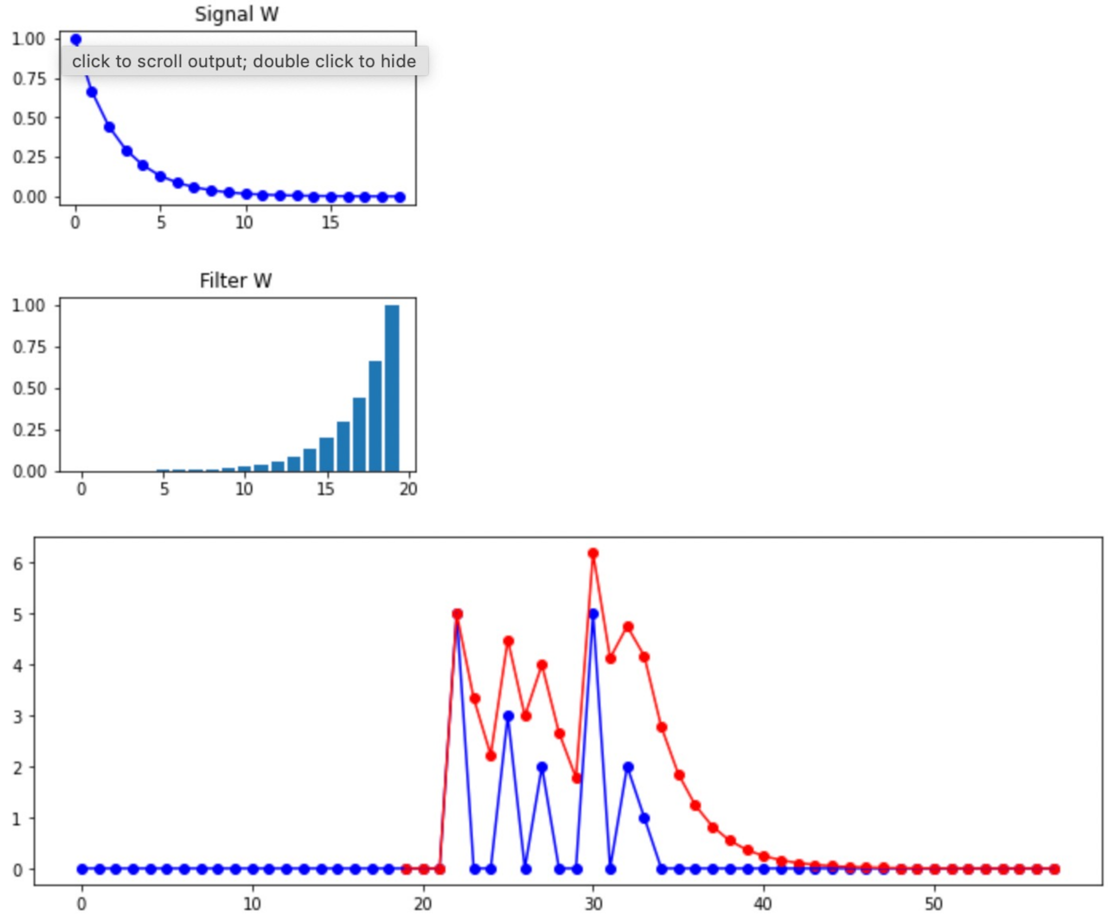


click to scroll output; double click to hide

Convolution vs Moving Average Filters

The image part with relationship ID rId15 was not found in the file.

Or ANY SOUND which is thought of as a collection of individual, instantaneous amplitude values, such as samples in a digital signal (though the same phenomenon exists in the continuous, actual realm of sound:



Convolution Reverb

Therefore, we can take two signals:

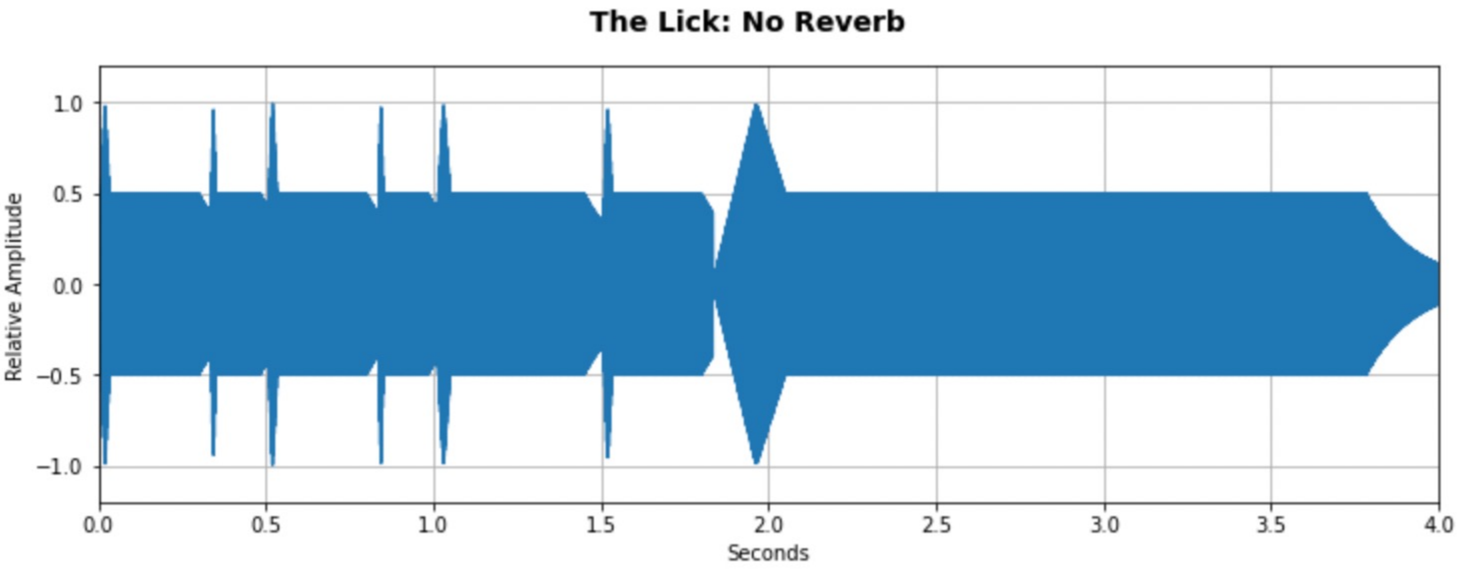
An “impulse response” signal produced by a sharp, instantaneous clap (or starter pistol, or fire cracker, etc.) in a particular environment (such as a concert hall),

A musical signal produced in an environment with NO reverberation (such as a synthesized signal, or one produced in a special ”quiet room” in a studio),

and convolve them to produce a signal representing the musical signal with the reverberation characteristics of the impulse response.

Convolution Reverb: Examples

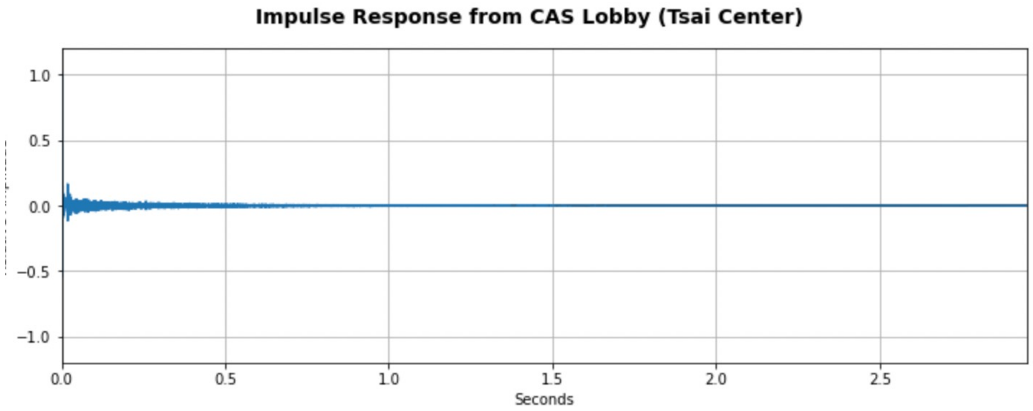
Here is my implementation of “The Lick” from HW 01:



The Lick: No Reverb

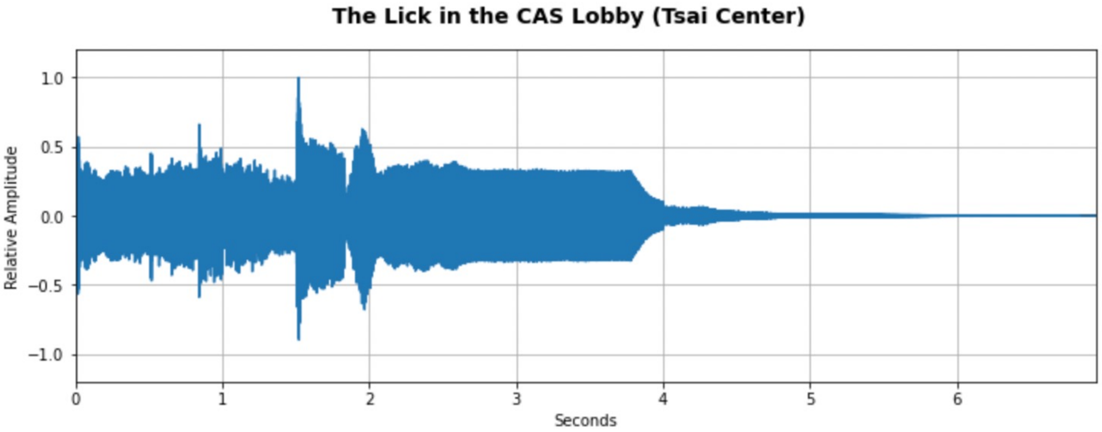
Convolution Reverb: Examples

Here is the impulse response from the CAS Lobby (Tsai Center):



Convolution Reverb: Examples

The Lick convolved with the Tsai Center impulse response:



Convolution Reverb: Examples

The Lick convolved with the Marsh Chapel impulse response:

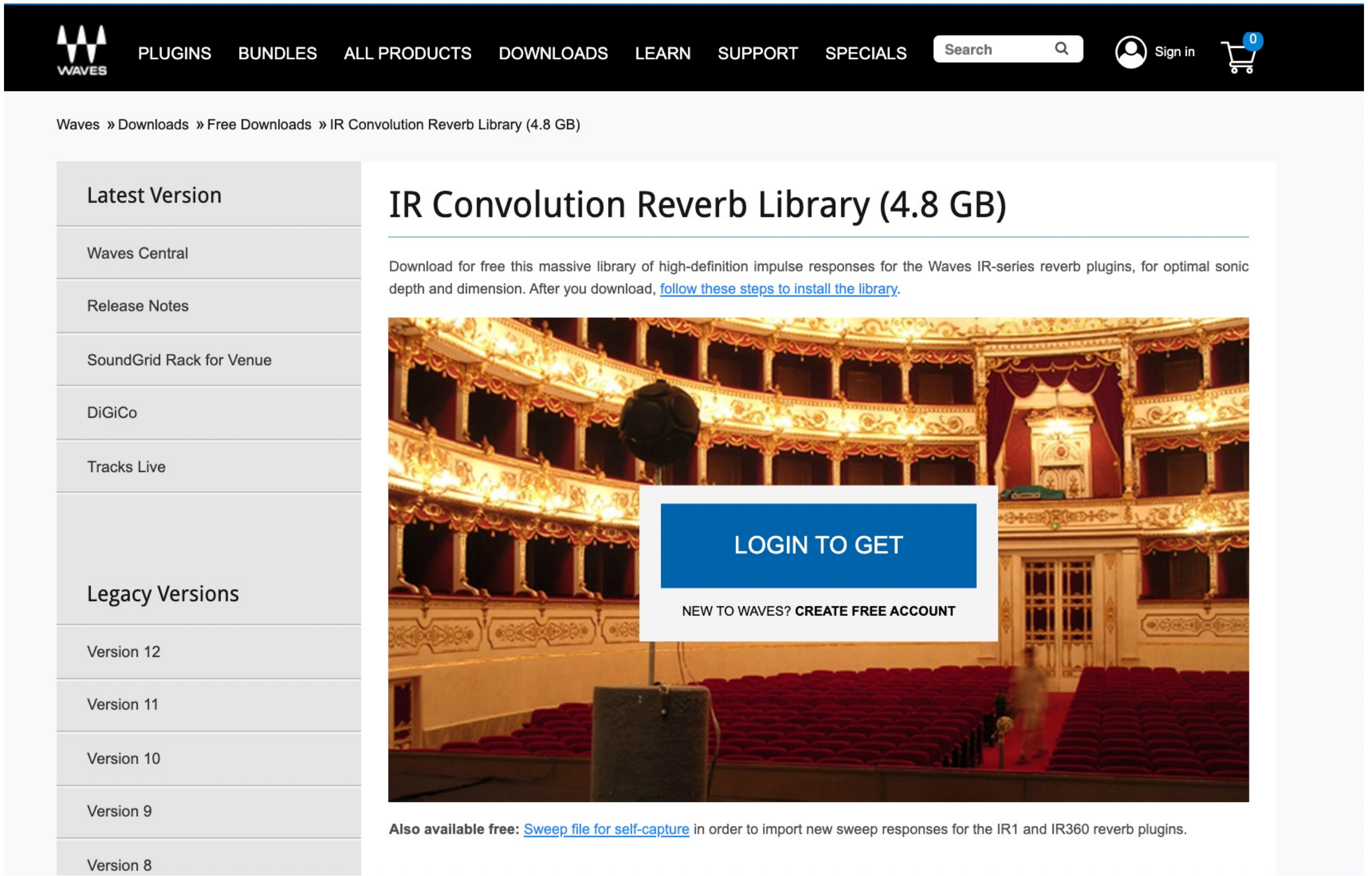
Convolution Reverb: Examples

The Lick convolved with the CAS Bathroom impulse response:

Convolution Reverb: Examples

The image part with relationship ID rId15 was not found in the file.

People have collected (some free, some for $$) a huge variety of impulse responses:



Waves » Downloads » Free Downloads » IR Convolution Reverb Library (4.8 GB)

IR Convolution Reverb Library (4.8 GB)

Download for free this massive library of high-definition impulse responses for the Waves IR-series reverb plugins, for optimal sonic depth and dimension. After you download, follow these steps to install the library.

Also available free: Sweep file for self-capture in order to import new sweep responses for the IR1 and IR360 reverb plugins.

Spatialization and Stereo Panning

Stereo audio systems have two speakers separated, with the listener assume to be in the middle:

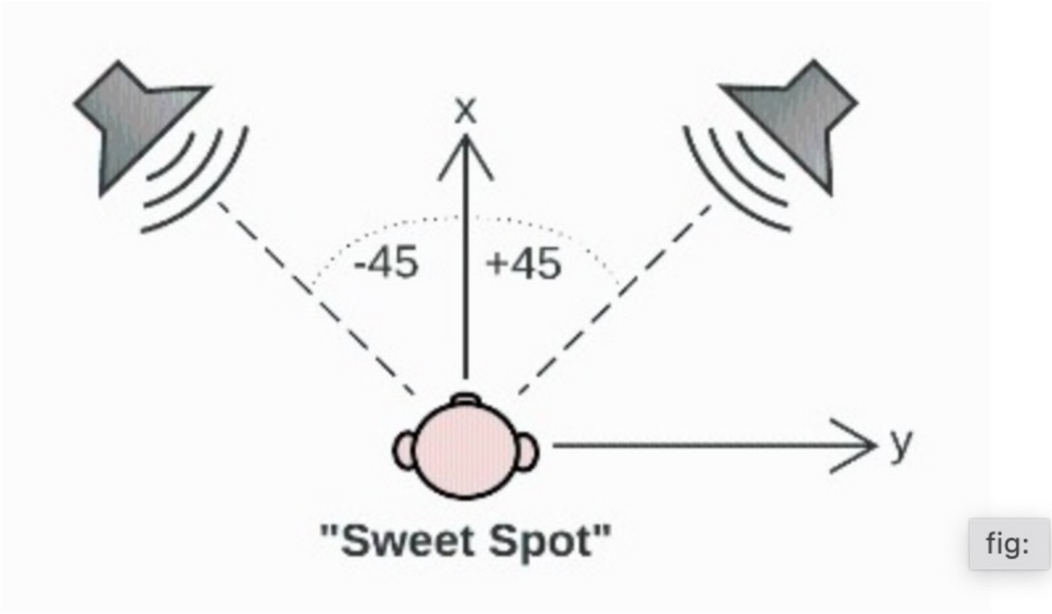


Figure 5: Speaker positioning & sweet spot.

Spatialization and Stereo Panning

The image part with relationship ID rId15 was not found in the file.

Suppose you want to "pan" a sound from left to right, or simply place a sound at some angle with respect to the speakers and the listener. The simplest solution is to simply split the amplitude between left an right.

This is called "Linear Panning" and is what I did in the example in HW 01.

fig:

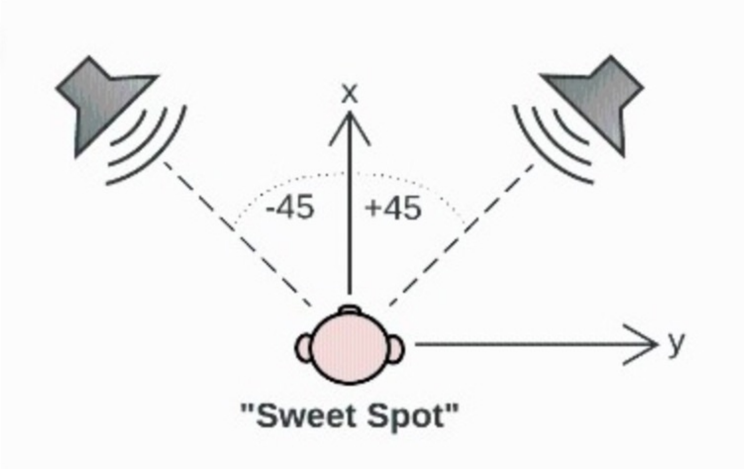


Figure 5: Speaker positioning & sweet spot.

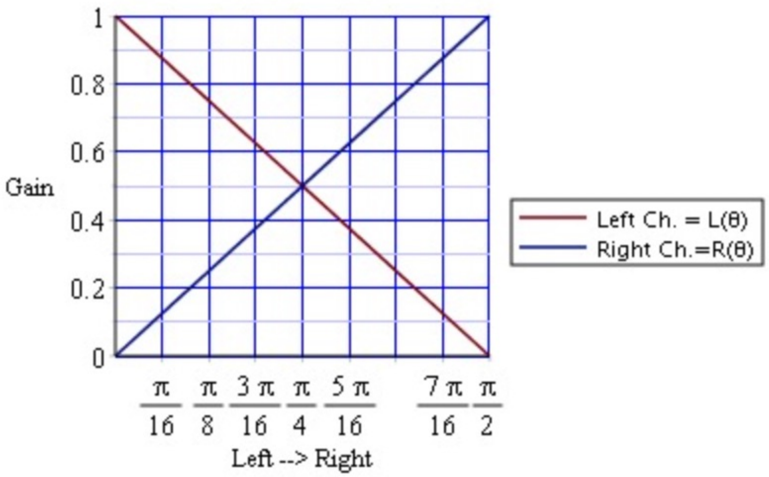


Figure 6: Linear Panning.

Spatialization and Stereo Panning

However, this does not account for the fact that loudness is proportional to the amplitude squared, so it tends to give a “hole in the middle” effect, where the panned sound seems to retreat into the background in the middle.

The solution is to use “constant power panning,” where the formula used boosts the middle to solve the “hole in the middle” problem.

Main idea: Power (and hence perceived loudness) is proportion to the squared amplitude, and sin$(\theta)^2 + cos(\theta)^2$ = 1.0, so the loudness of the left channel should be cos(θ) and the loudness of the right should be sin(θ).

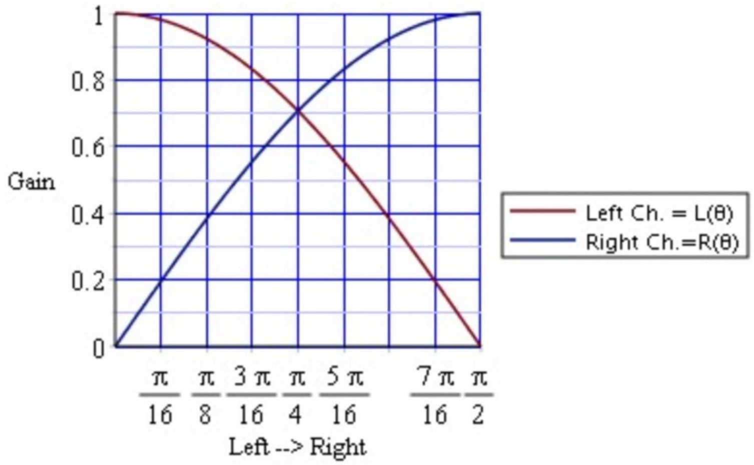


Figure 7: Constant Power Panning.