Digital Signal Processing

Moving Average Filters

Today's Topics

- Introduce the Moving Average Filter
- Example of how it works
- Describe the frequency response
- Discuss delay issues and mitigations
- Demonstrate the time domain response
- Briefly discuss implementation

Filter Classification

FILTER USED FOR:	FILTER IMPLEMENTED BY: Convolution *Finite Impulse Response (FIR)*	Recursion *Infinite Impulse Response (IIR)*
Time Domain *(smoothing, DC removal)*	Moving average (Ch. 15)	Single pole (Ch. 19)
Frequency Domain *(separating frequencies)*	Windowed-sinc (Ch. 16)	Chebyshev (Ch. 20)
Custom *(Deconvolution)*	FIR custom (Ch. 17)	Iterative design (Ch. 26)

What is a Moving Average Filter?

- A simple digital filter used for:
  - Reducing random noise
  - Retaining sharp step response

- Not as good at separating frequencies
  - Other filter types have better selectivity

How Does the Moving Average Filter Work?

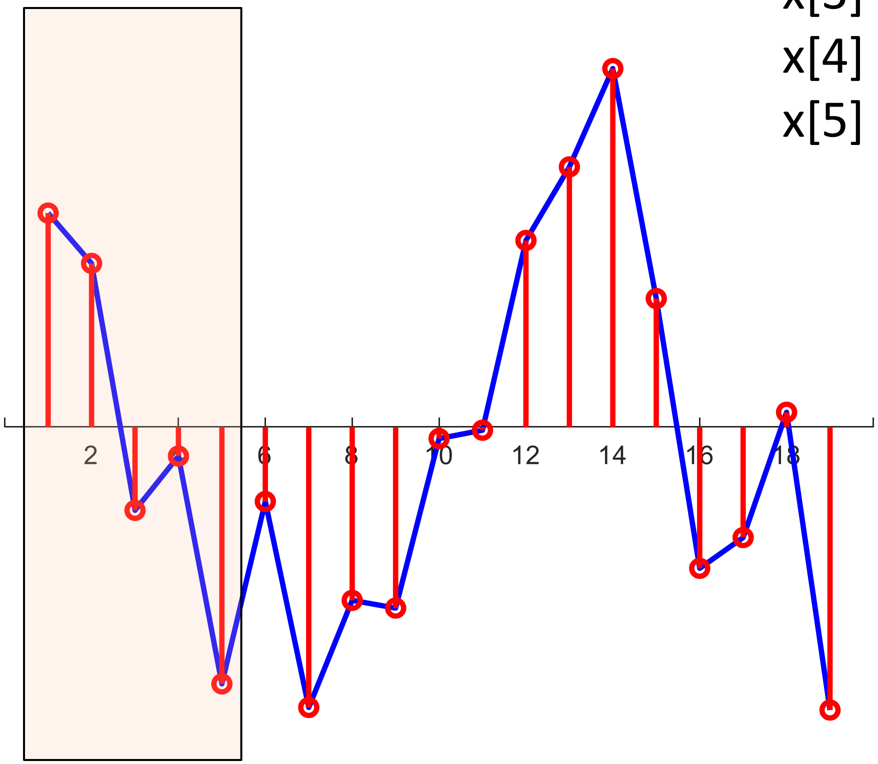
- Create a window of M samples of the signal at a time
- Average the M samples to produce one output sample
- Slide the window over one sample, repeat

Moving Average Filter Example

5-Point Wide Window

x[1] = 1.0937
x[2] = 0.8363
x[3] = -0.4262
x[4] = -0.1487
x[5] = -1.3135

Ave. = 0.0083



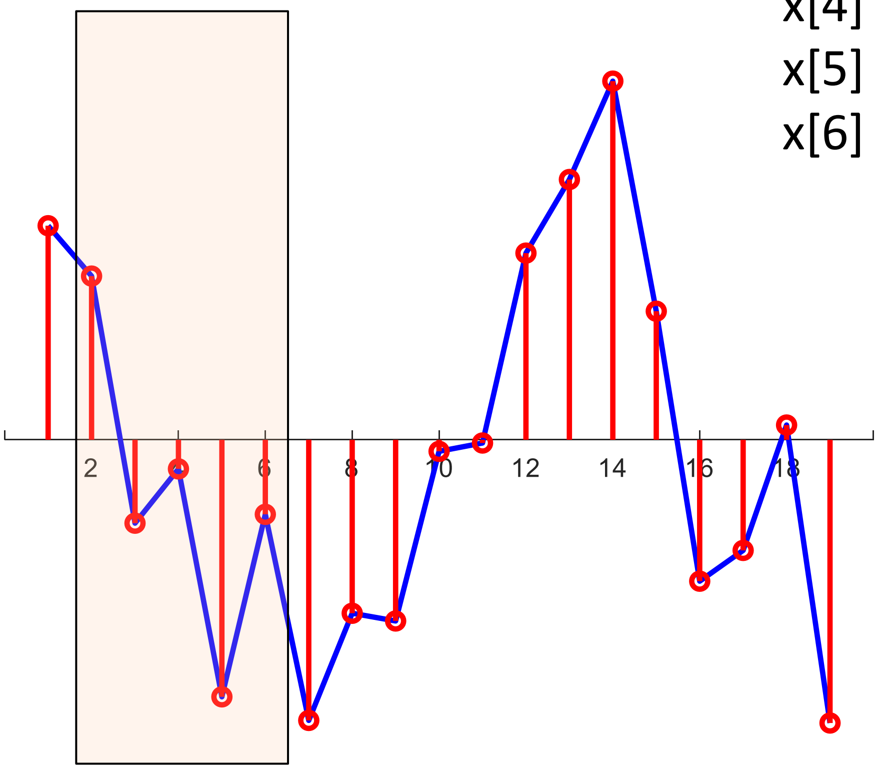
First Output Sample

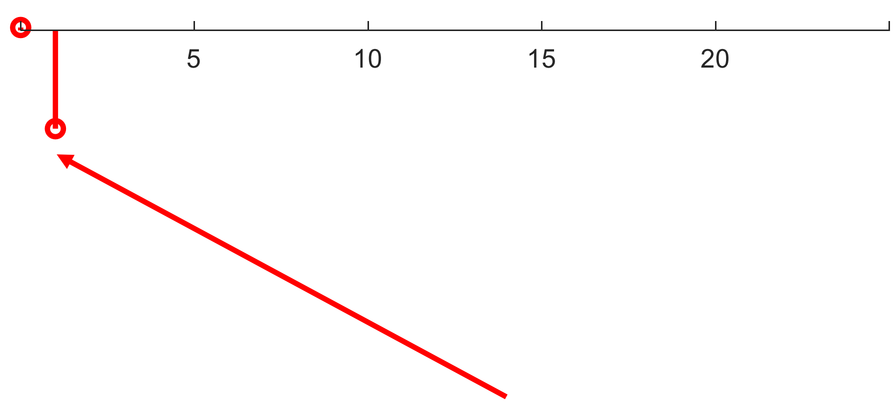
Moving Average Filter Example

Move window
1 sample

x[2] = 0.8363
x[3] = -0.4262
x[4] = -0.1487
x[5] = -1.3135
x[6] = -0.3810

Ave. = -0.2866





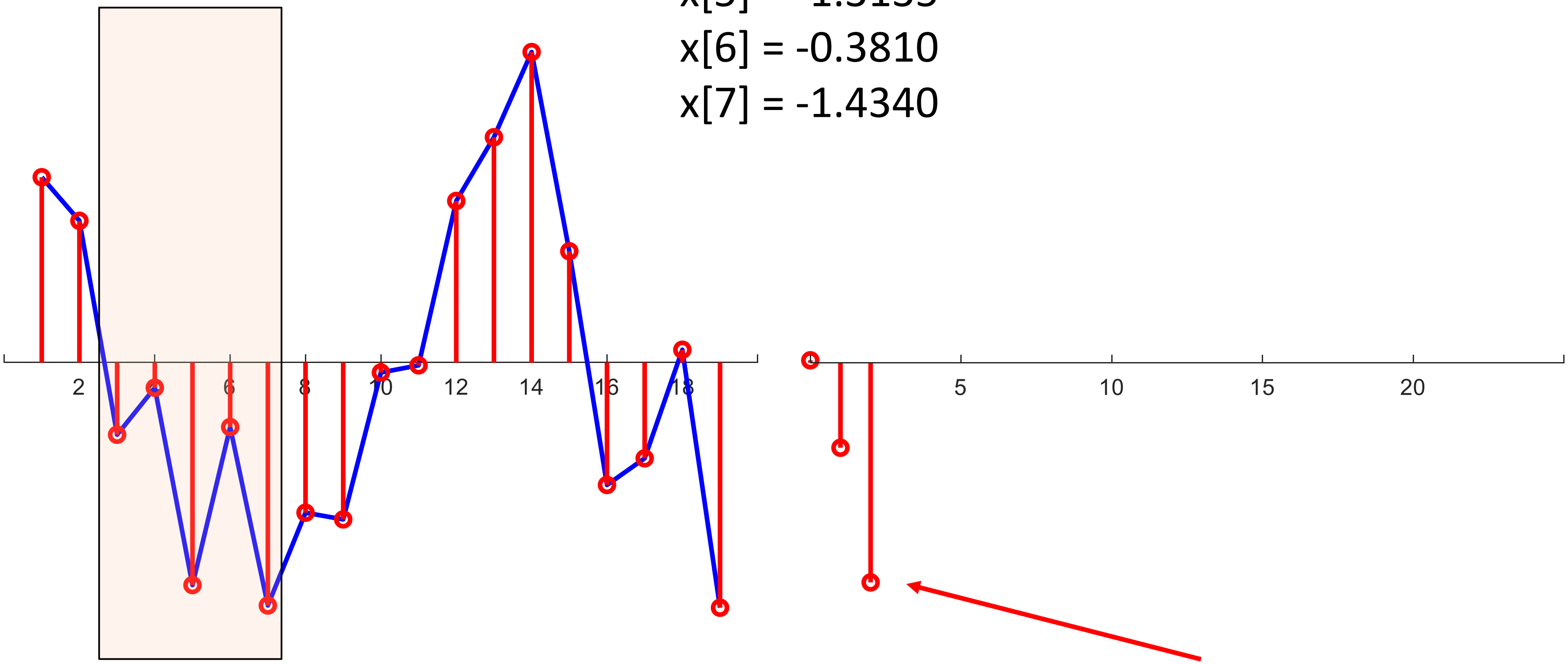
Second Output Sample

Moving Average Filter Example

Move window
1 sample

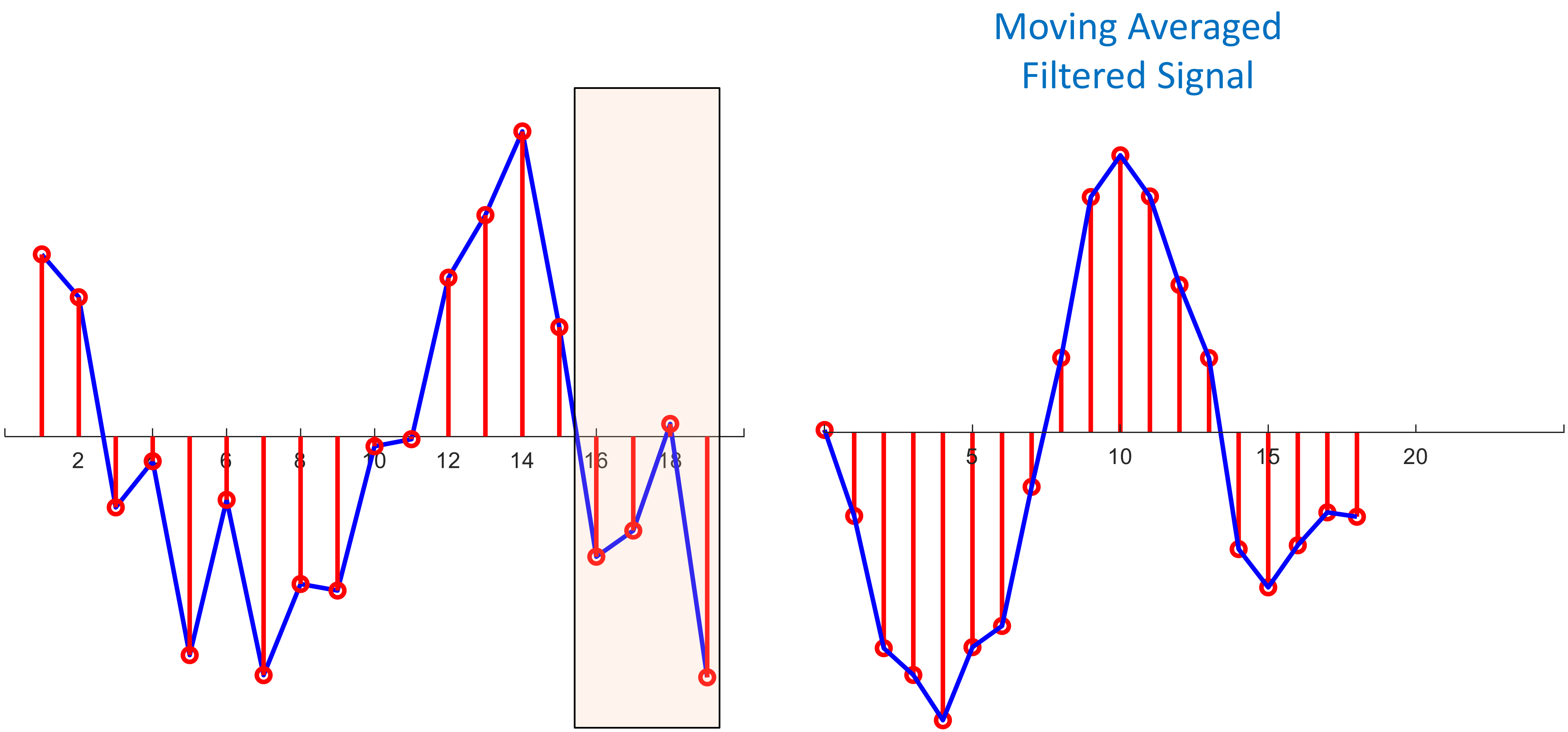
x[3] = -0.4262
x[4] = -0.1487
x[5] = -1.3135
x[6] = -0.3810
x[7] = -1.4340

Ave. = -0.7407



3rd Output Sample

Moving Average Filter Example



Moving Average Filter Advantages

- The moving average filter is helpful for smoothing or reducing noise on a signal

- Easy to implement
  - Can be implemented using convolution
  - Can be implemented using recursion

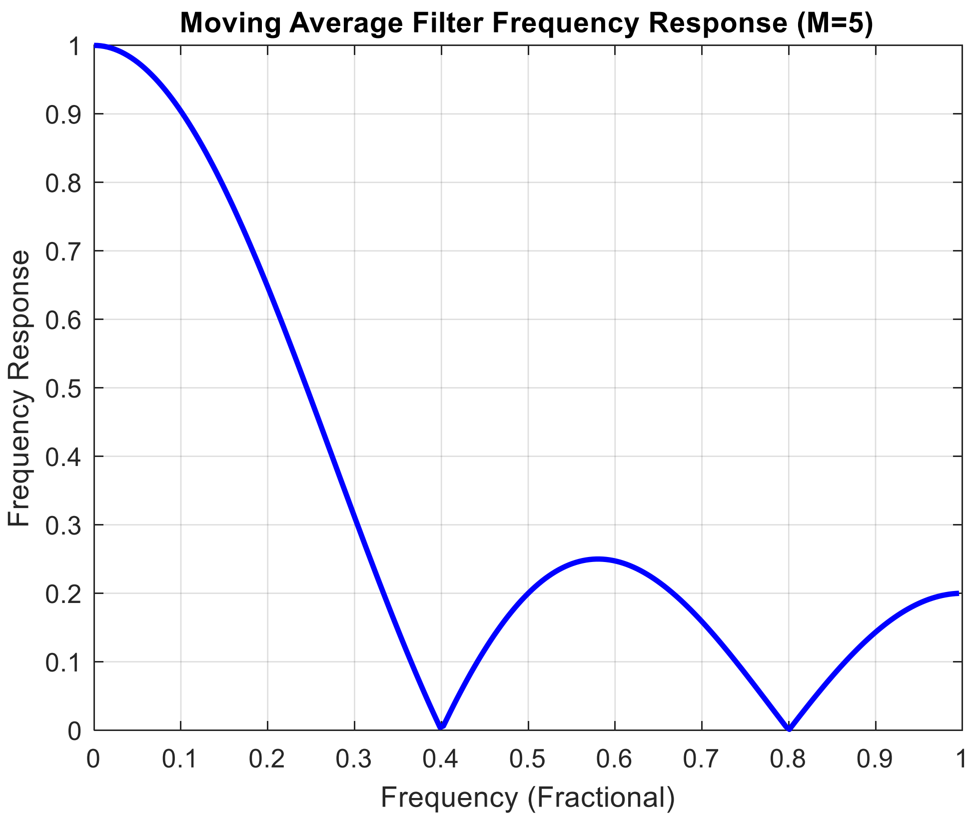
Moving Average Filter Disadvantages

- Frequency response is not very selective
  - May be less useful in separating signals closely spaced in frequency

- There can be a time lag in the filter output
  - Though we can correct for this

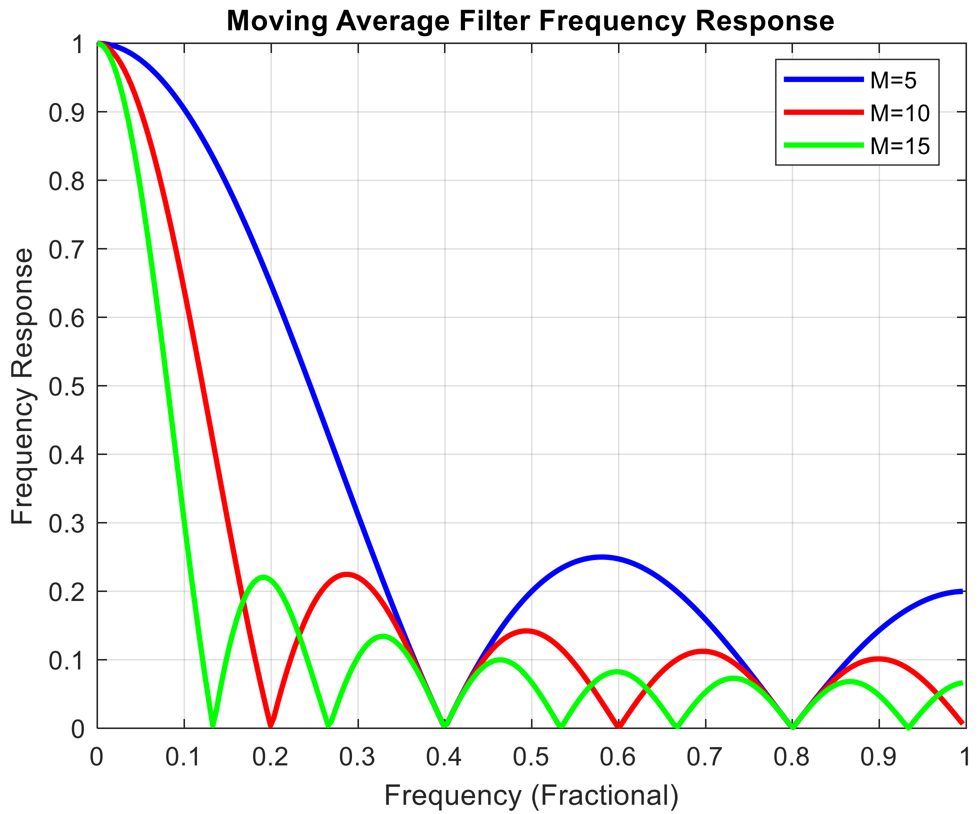
Moving Average Filter Frequency Response

- Frequency response follows a SINC response
- Slow frequency roll-off



Moving Average Filter Frequency Response

- Increasing the filter length improves the roll off
- Lowers the ultimate attenuation
- Rate of change in the transition region stays the same



RIT EEE

Moving Average Filter Mathematically

- Mathematically the filter computes the average of M samples of the input for each output

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

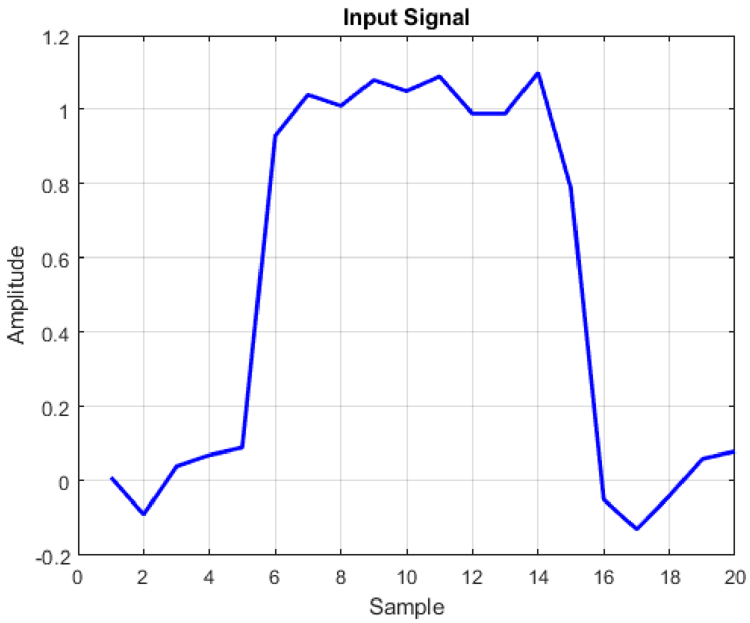
- Can be implemented using convolution and a filter kernel with values

$$h[n] = [\frac{1}{M}, \frac{1}{M}, \frac{1}{M}, \dots \frac{1}{M}]$$

MAV Filter ICP

- Given the sequence of 20 samples shown below, filter the sequence using a moving average filter of length 5

Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Value	0.01	-0.09	0.04	0.07	0.09	0.93	1.04	1.01	1.08	1.05	1.09	0.99	0.99	1.1	0.79	-0.05	-0.13	-0.04	0.06	0.08



MAV Filter ICP

- The impulse response of the MAV filter of length 5 will be:

$$h = [\frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}, \frac{1}{5}]$$

- The output of the filter will be 5 adjacent samples averaged together

- The first 4 samples will be part of the end effect

MAV Filter ICP

- Given the sequence of 20 samples shown below, filter the sequence using a moving average filter of length 5

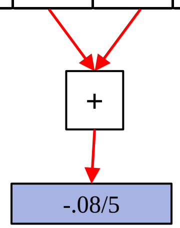
Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Value	0.01	-0.09	0.04	0.07	0.09	0.93	1.04	1.01	1.08	1.05	1.09	0.99	0.99	1.1	0.79	-0.05	-0.13	-0.04	0.06	0.08

.002 .01/5

MAV Filter ICP

- Given the sequence of 20 samples shown below, filter the sequence using a moving average filter of length 5

Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Value	0.01	-0.09	0.04	0.07	0.09	0.93	1.04	1.01	1.08	1.05	1.09	0.99	0.99	1.1	0.79	-0.05	-0.13	-0.04	0.06	0.08



-.016

MAV Filter ICP

- Given the sequence of 20 samples shown below, filter the sequence using a moving average filter of length 5

Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Value	0.01	-0.09	0.04	0.07	0.09	0.93	1.04	1.01	1.08	1.05	1.09	0.99	0.99	1.1	0.79	-0.05	-0.13	-0.04	0.06	0.08

+

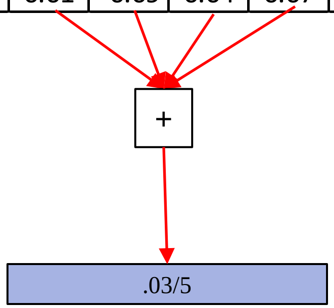
-.04/5

-.008

MAV Filter ICP

- Given the sequence of 20 samples shown below, filter the sequence using a moving average filter of length 5

Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Value	0.01	-0.09	0.04	0.07	0.09	0.93	1.04	1.01	1.08	1.05	1.09	0.99	0.99	1.1	0.79	-0.05	-0.13	-0.04	0.06	0.08

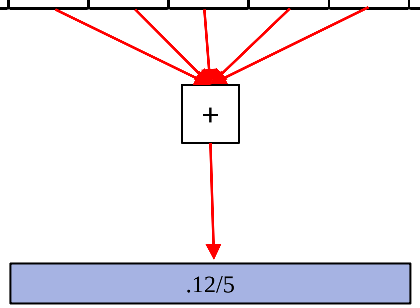


.006

MAV Filter ICP

- Given the sequence of 20 samples shown below, filter the sequence using a moving average filter of length 5

Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Value	0.01	-0.09	0.04	0.07	0.09	0.93	1.04	1.01	1.08	1.05	1.09	0.99	0.99	1.1	0.79	-0.05	-0.13	-0.04	0.06	0.08

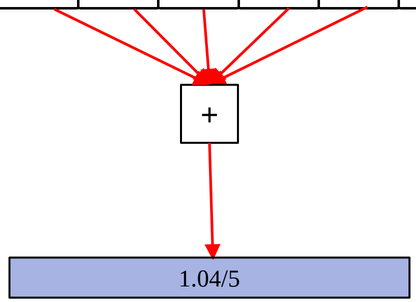


.024

MAV Filter ICP

- Given the sequence of 20 samples shown below, filter the sequence using a moving average filter of length 5

Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Value	0.01	-0.09	0.04	0.07	0.09	0.93	1.04	1.01	1.08	1.05	1.09	0.99	0.99	1.1	0.79	-0.05	-0.13	-0.04	0.06	0.08

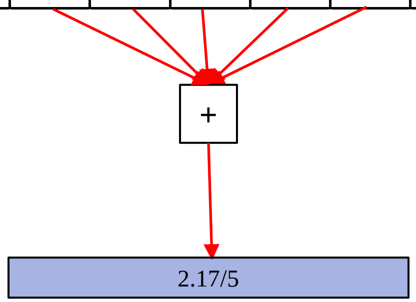


0.208

MAV Filter ICP

- Given the sequence of 20 samples shown below, filter the sequence using a moving average filter of length 5

Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Value	0.01	-0.09	0.04	0.07	0.09	0.93	1.04	1.01	1.08	1.05	1.09	0.99	0.99	1.1	0.79	-0.05	-0.13	-0.04	0.06	0.08

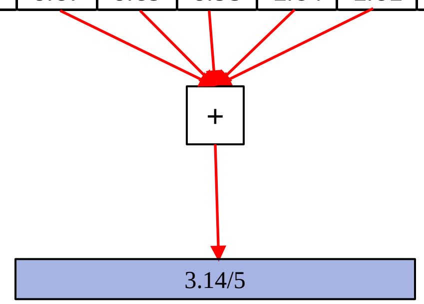


.434

MAV Filter ICP

- Given the sequence of 20 samples shown below, filter the sequence using a moving average filter of length 5

Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Value	0.01	-0.09	0.04	0.07	0.09	0.93	1.04	1.01	1.08	1.05	1.09	0.99	0.99	1.1	0.79	-0.05	-0.13	-0.04	0.06	0.08

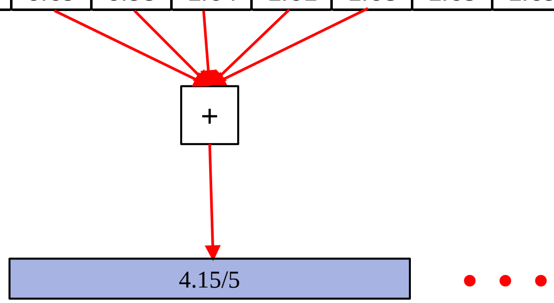


0.628

MAV Filter ICP

- Given the sequence of 20 samples shown below, filter the sequence using a moving average filter of length 5

Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Value	0.01	-0.09	0.04	0.07	0.09	0.93	1.04	1.01	1.08	1.05	1.09	0.99	0.99	1.1	0.79	-0.05	-0.13	-0.04	0.06	0.08

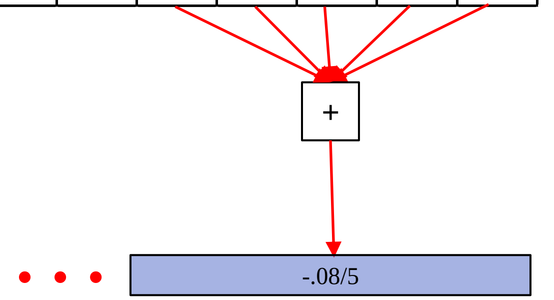


0.830

MAV Filter ICP

- Given the sequence of 20 samples shown below, filter the sequence using a moving average filter of length 5

Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Value	0.01	-0.09	0.04	0.07	0.09	0.93	1.04	1.01	1.08	1.05	1.09	0.99	0.99	1.1	0.79	-0.05	-0.13	-0.04	0.06	0.08

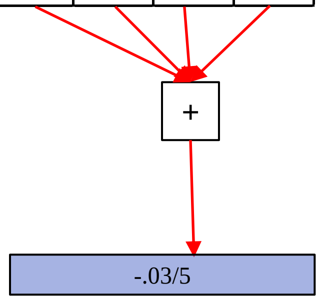


-0.16

MAV Filter ICP

- Given the sequence of 20 samples shown below, filter the sequence using a moving average filter of length 5

Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Value	0.01	-0.09	0.04	0.07	0.09	0.93	1.04	1.01	1.08	1.05	1.09	0.99	0.99	1.1	0.79	-0.05	-0.13	-0.04	0.06	0.08

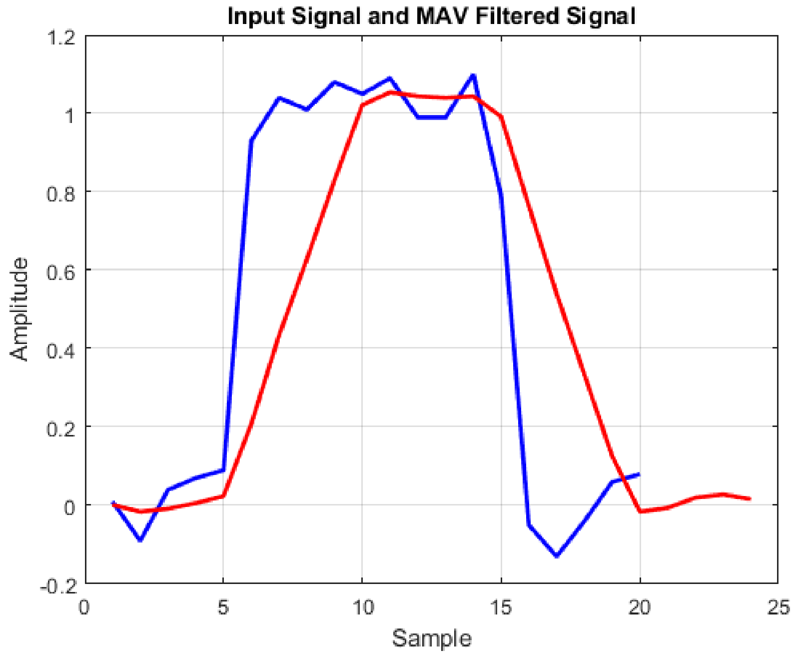


-0.006

Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Output	0.002	-0.016	-0.008	0.006	0.024	0.208	0.434	0.628	0.830	1.022	1.054	1.044	1.040	1.044	0.992	0.764	0.540	0.334	0.126	-0.016	-0.006	0.020	0.028	0.016

MAV Filter ICP

Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Value	0.01	-0.09	0.04	0.07	0.09	0.93	1.04	1.01	1.08	1.05	1.09	0.99	0.99	1.1	0.79	-0.05	-0.13	-0.04	0.06	0.08



Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
Output	0.002	-0.016	-0.008	0.006	0.024	0.208	0.434	0.628	0.830	1.022	1.054	1.044	1.040	1.044	0.992	0.764	0.540	0.334	0.126	-0.016	-0.006	0.020	0.028	0.016

Adjusting the Delay

- If the output sample is computed from 5 input samples

$$y[40] = \frac{1}{5}(\mathrm{x}[40] + \mathrm{x}[41] + \mathrm{x}[42] + \mathrm{x}[43] + \mathrm{x}[44])$$

- The we don’t get an output until 5 samples have entered the filter
  - The output is delayed by 4 samples

Adjusting the Delay

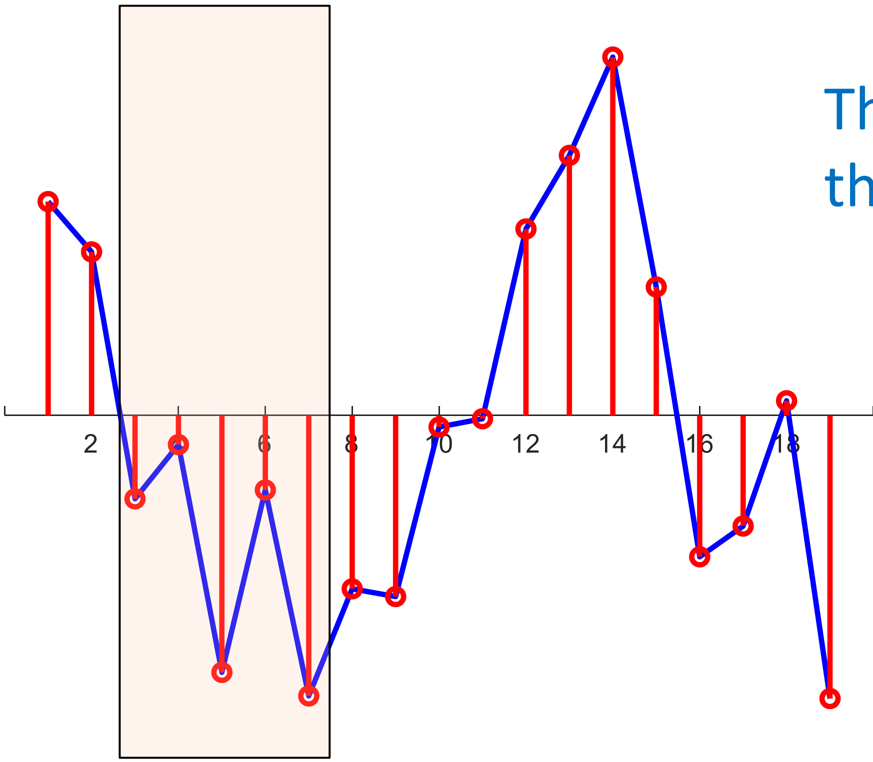
- The delay can be adjusted by using an offset of the input samples

$$y[40] = \frac{1}{5}(x[38] + x[39] + x[40] + x[41] + x[42])$$

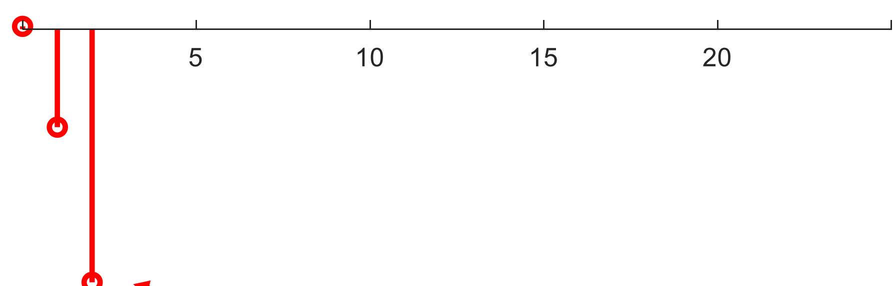
- The output uses past samples and future samples
  - This is non-causal
  - Prevents time lag between the input and the output

Adjusting the Delay

$$y[3] = \frac{1}{5}(\mathrm{x}[3] + \mathrm{x}[4] + \mathrm{x}[5] + \mathrm{x}[6] + \mathrm{x}[7])$$



The output sample is aligned with the oldest sample

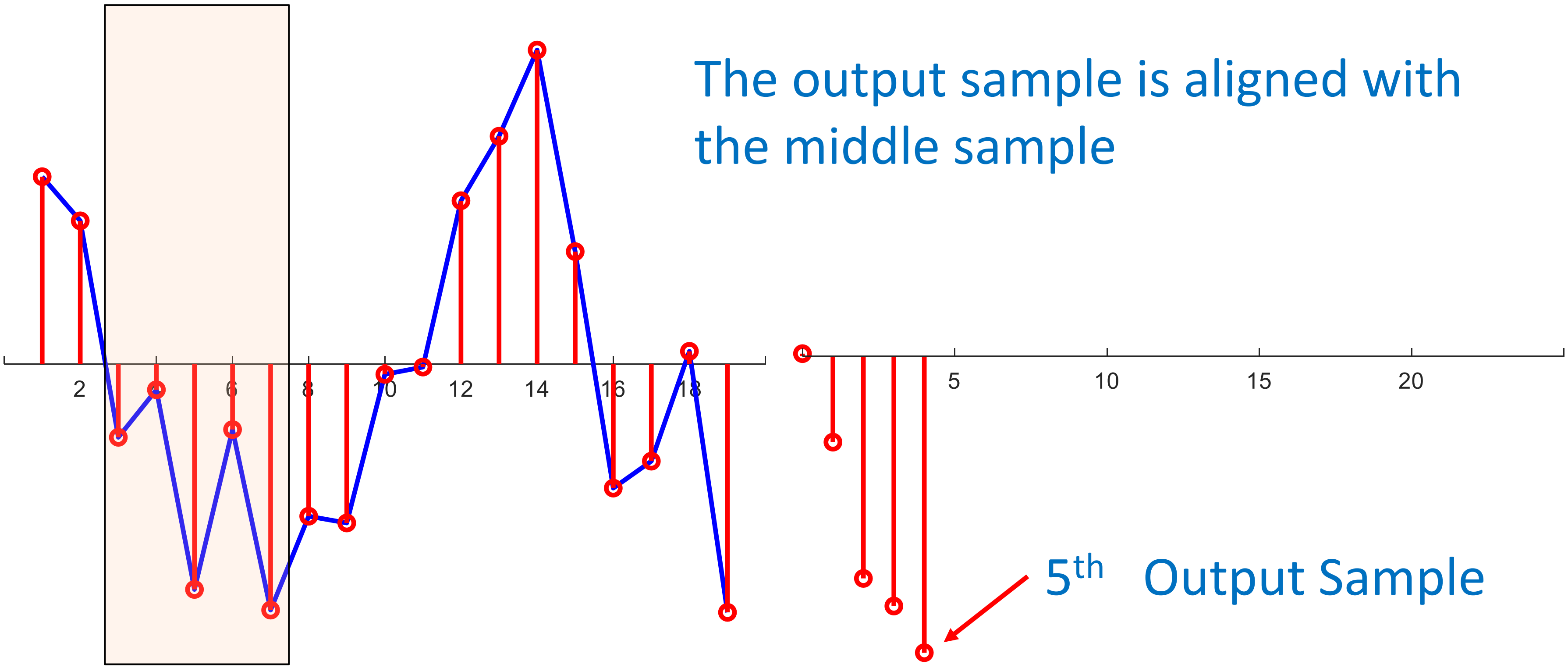


3rd Output Sample

Adjusting the Delay

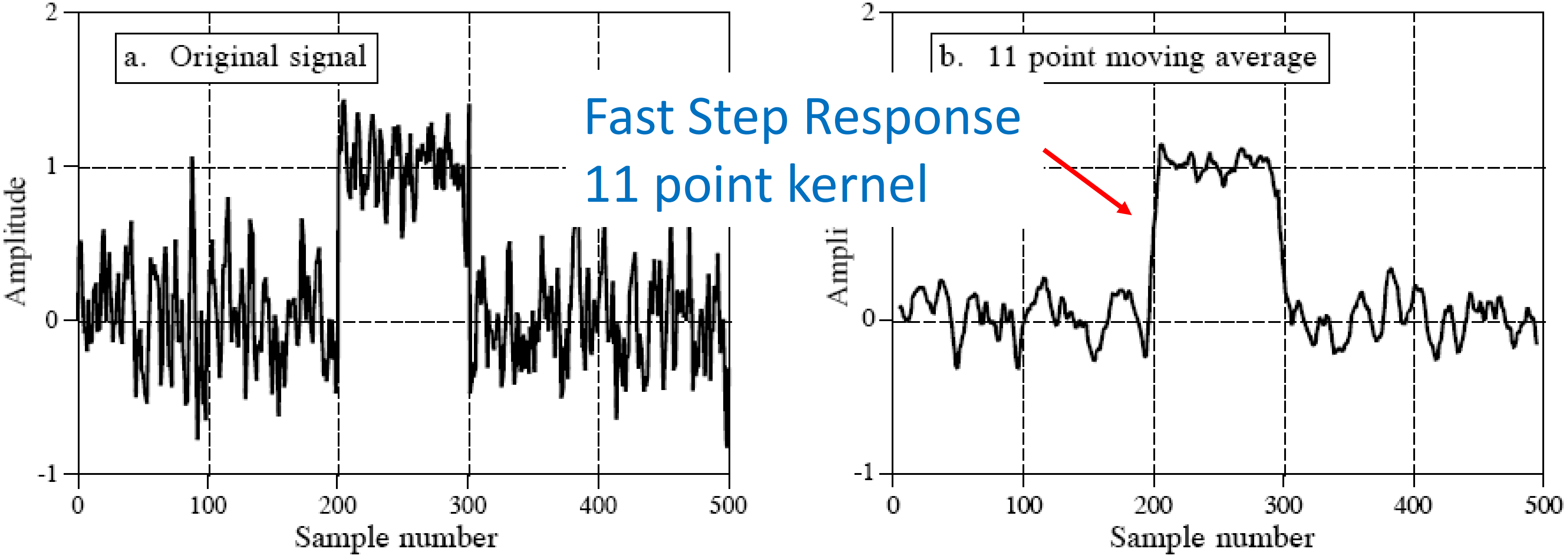
$$y[5] = \frac{1}{5}(\mathrm{x}[3] + \mathrm{x}[4] + \mathrm{x}[5] + \mathrm{x}[6] + \mathrm{x}[7])$$

The output sample is aligned with the middle sample



5th Output Sample

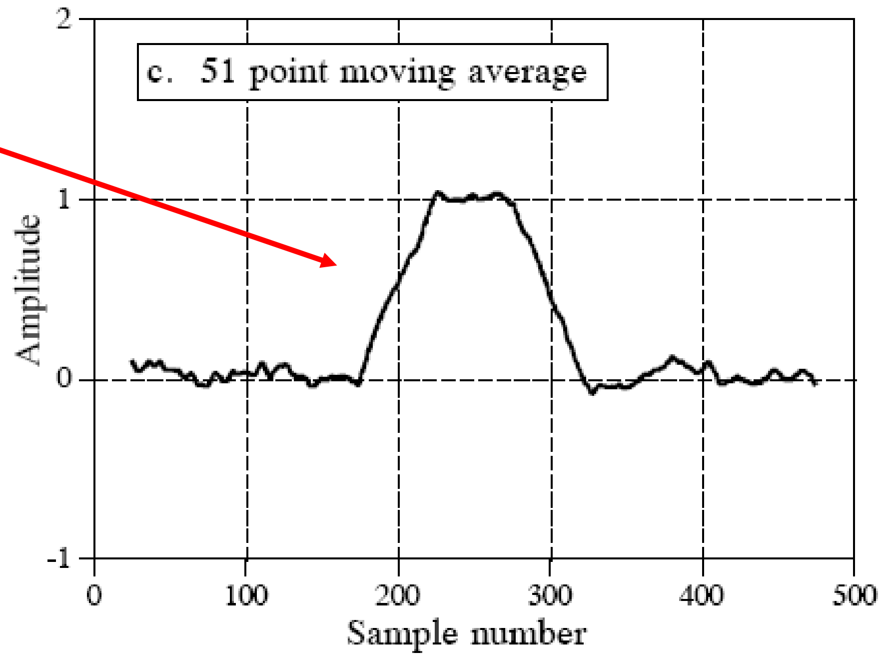
Example Time Domain Response



- The time domain response varies with the length of the filter
- Shorter lengths respond faster

Example Time Domain Response

Slower Step Response with 51 point kernel



- Longer step response for longer filter length

Implementation Convolution

- Use fixed point data types
  - Scale kernel values so that they can be represented using fixed point values
- Example – M = 51 points

$$h = [\frac{1}{51}, \frac{1}{51}, \frac{1}{51}, \dots, \frac{1}{51}]$$

Kernel values are fractions

$$h = [1, 1, 1, \dots, 1]$$

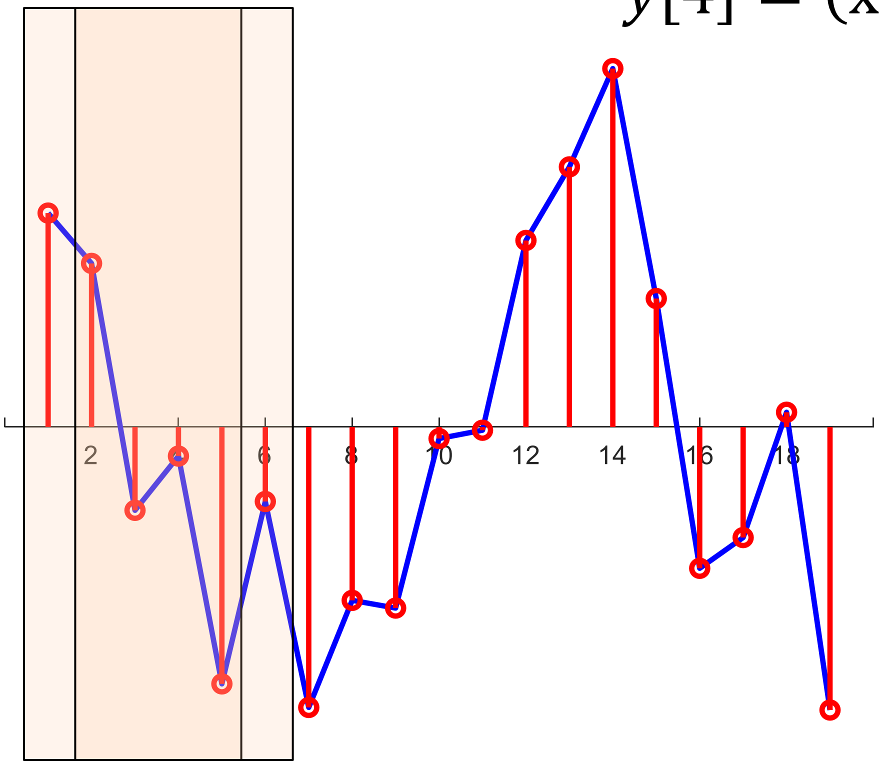
Scale by 51 to fixed point value

$$y_{out} = y_{filter}/51$$

Using floating point to scale output

Implementation by Recursion

5-Point Wide Window



$$y[3] = (x[1] + x[2] + x[3] + x[4] + x[5])$$

$$y[4] = (x[2] + x[3] + x[4] + x[5] + x[6])$$

also

$$y[4] = y[3] - x[1] + x[6]$$

Oldest sample subtracted

Newest sample added

Must accumulate M samples for first output