EE327 Digital Signal Processing Moving Average Filters

Yasser F. O. Mohammad

REMINDER 1: How to Represent a Filter

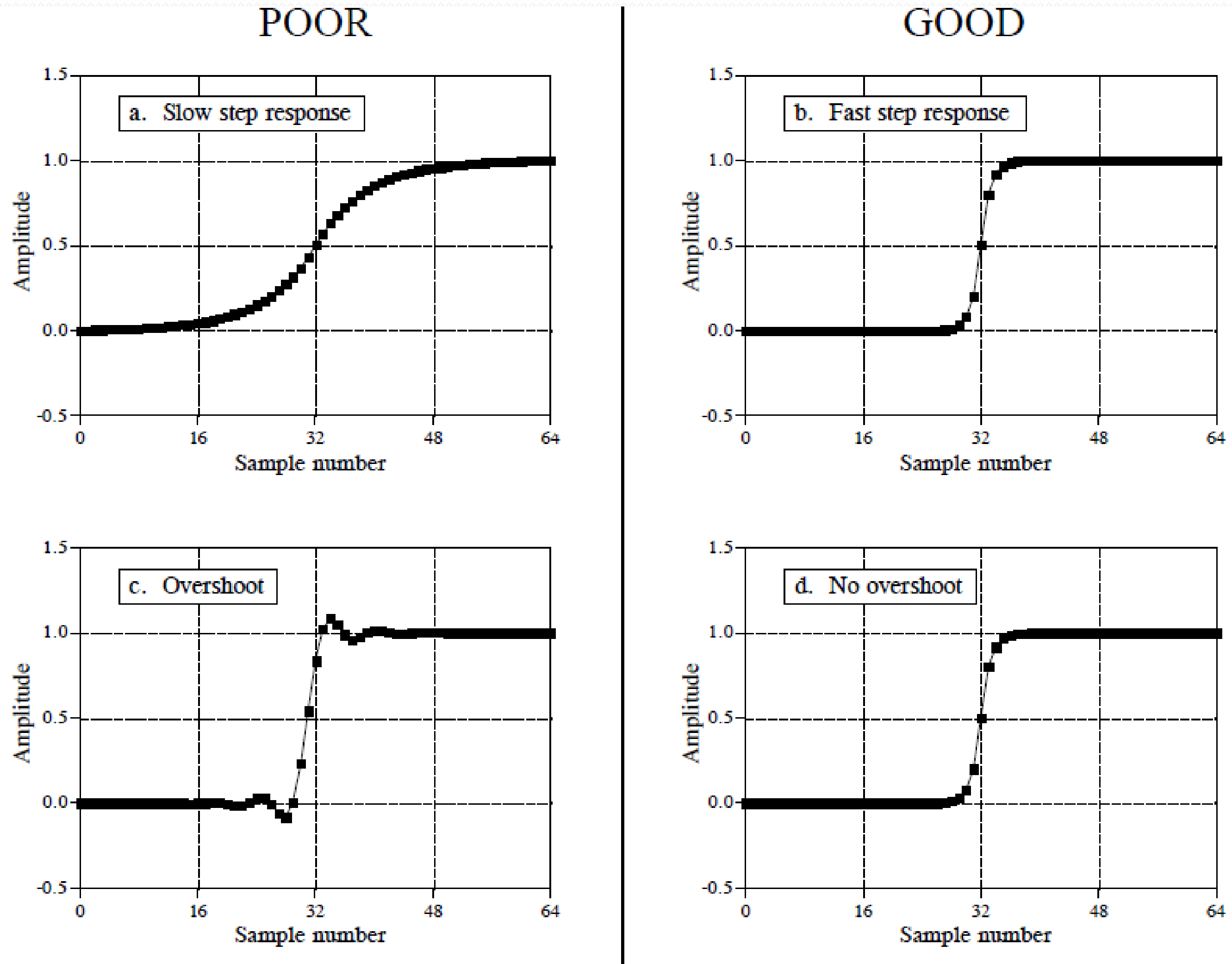
- Finite Impulse Response (FIR)
  - Impulse Response
    - Filter Kernel
  - Step Response
  - Frequency Response

$$y[n] = \sum_{i=0}^{M} a_{-i} x[n-i]$$

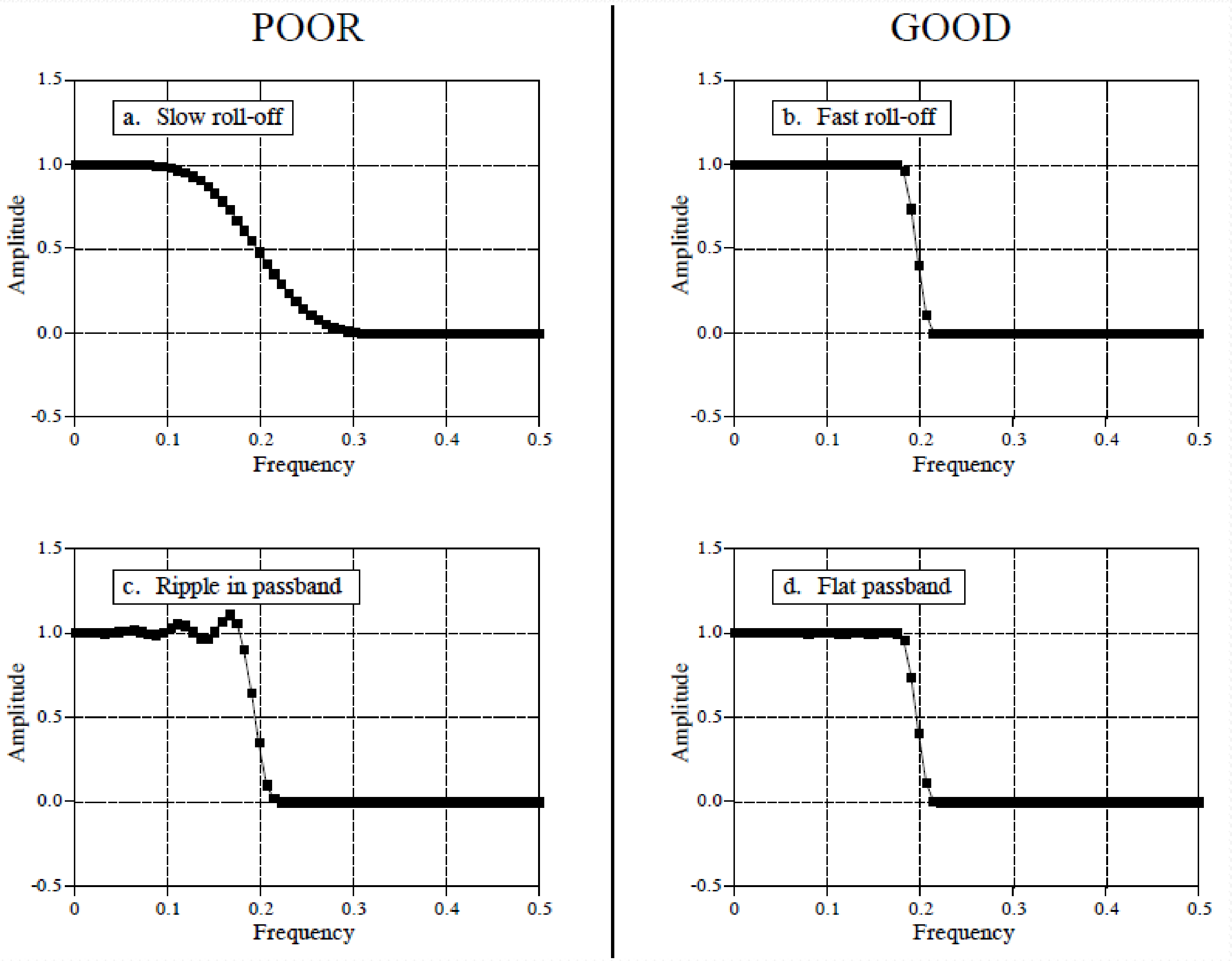
- Infinite Impulse Response (IIR)
  - Recursion Coefficient

$$y[n] = \sum_{j=0}^{M_1} a_{-j} x[n-j] + \sum_{i=0}^{M_2} b_{-i} y[n-i]$$

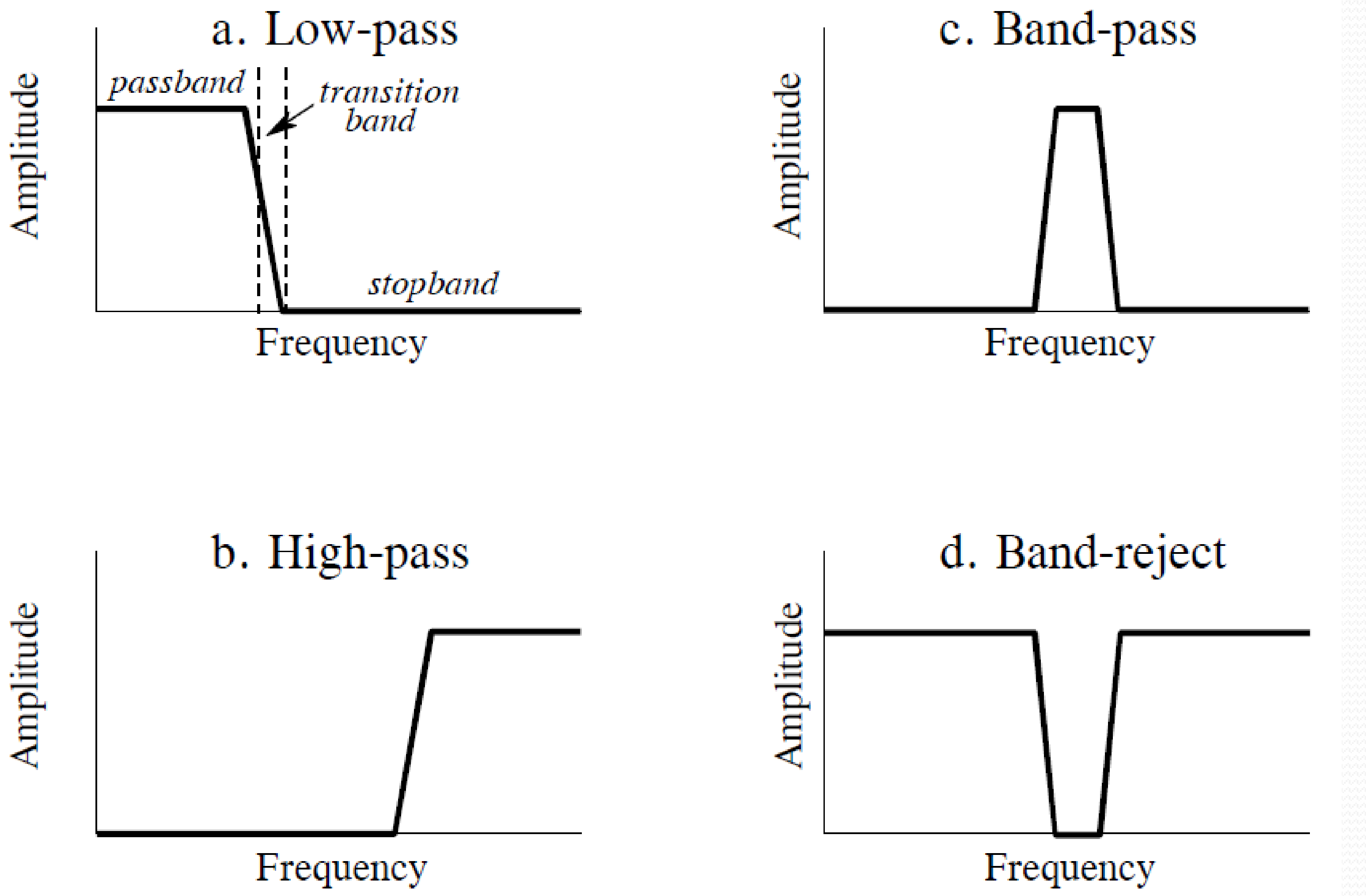
REMINDER 2: Time Domain Parameters



REMINDER 3: Frequency Domain Parameters

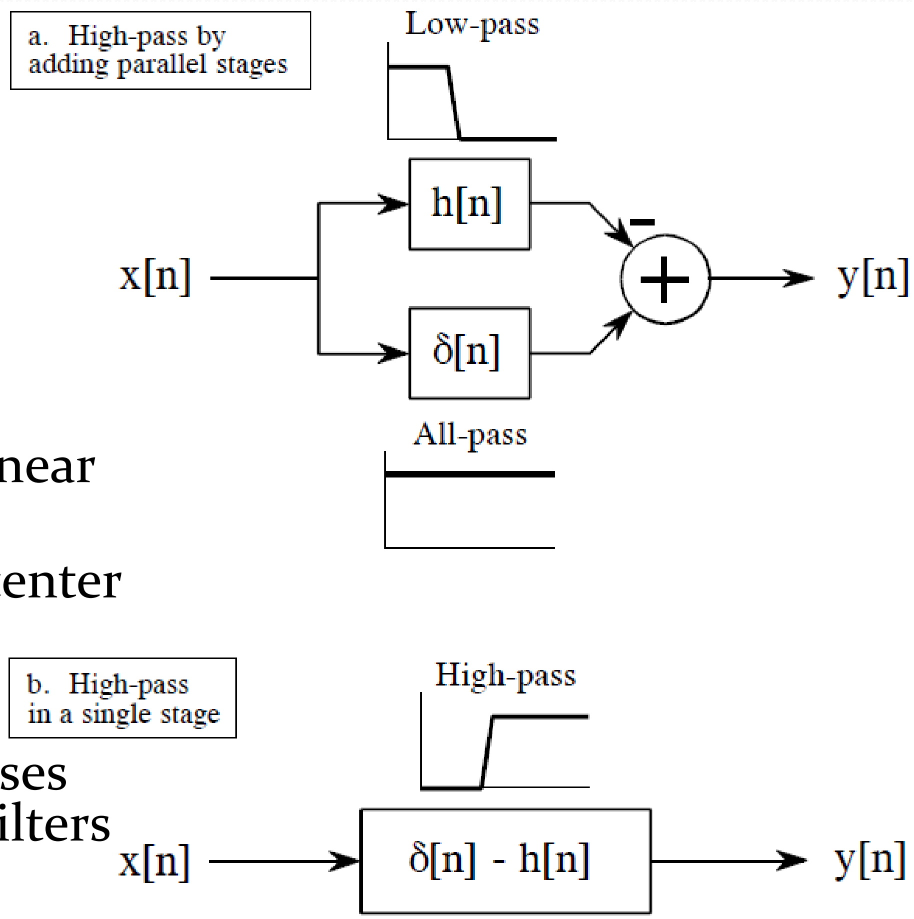


REMINDER 4: Most Common Filter Types



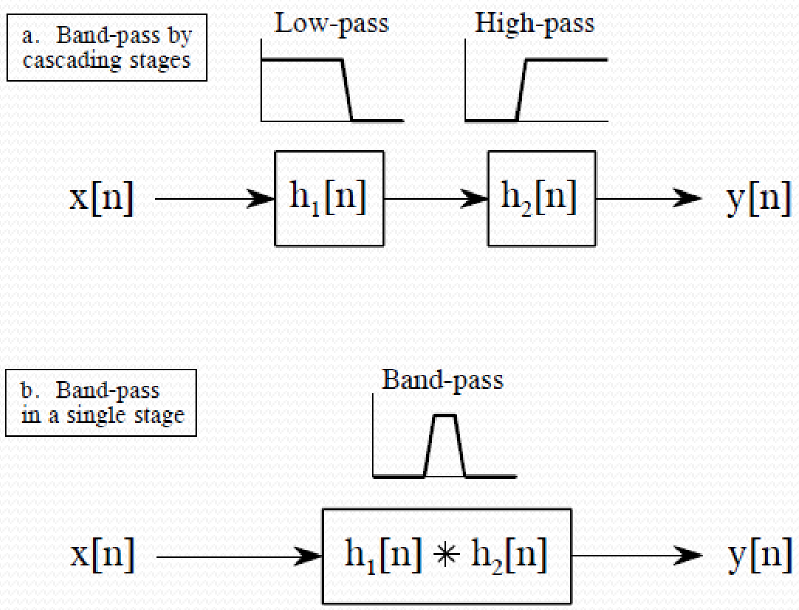
REMINDER 5: Converting Low Pass to High Pass

- Spectral Inversion
- Steps Done to IR:
  - Change sign of each sample
  - Add one to the center sample
- Conditions:
  - Original Filter is symmetric (Linear Phase)
  - The one must be added in the center
- Reason for Conditions:
  - To make the low frequency phases from the low pass and all pass filters the same



REMINDER 6: Converting Low Pass to High Pass

- Spectral Reversal
- Steps Done to IR:
  - Change sign of every other sample
- Why is it working?:
  - Multiplication with a sign of frequency of 0.5



Filter Equation

$$y[j] = \frac{1}{M} \sum_{i=n_0}^{n_0+M-1} x[j+i]$$

n_o	Meaning
>0	Noncausal. Output at time j depends on input AFTER j only with skip
0	Noncausal. Output at time j depends on input AFTER j only
-(M-1)/2	Noncausal. Output at time j depends on M/2 points BEFORE j and M/2 points after it
-(M-1)	Causal. Output at time j depends on M points BEFORE and at j
<-(M-1)	Causal with delay

Numeric Example

- x[n]=[1,2,3,4,5,4,3,2,1] , M=3, n_0=-1

$$y[j]=\frac{1}{3}\sum_{i=-1}^{1}x[j+i]$$

$$y[0]=\frac{0+1+2}{3}=1 \qquad y[1]=\frac{1+2+3}{3}=2$$

$$y[2]=\frac{2+3+4}{3}=3 \qquad y[3]=\frac{3+4+5}{3}=4$$

$$y[4]=\frac{4+5+4}{3}=4\frac{1}{3} \qquad y[5]=\frac{3+4+5}{3}=4$$

- y[n]=[1,2,3,4,4.33,4,3,2,1]

Calculating Moving Average

```
100 'MOVING AVERAGE FILTER
110 'This program filters 5000 samples with a 101 point moving
120 'average filter, resulting in 4900 samples of filtered data.
130 '
140 DIM X[4999]                      'X[ ] holds the input signal
150 DIM Y[4999]                      'Y[ ] holds the output signal
160 '
170 GOSUB XXXX                       'Mythical subroutine to load X[ ]
180 '
190 FOR I% = 50 TO 4949              'Loop for each point in the output signal
200   Y[I%] = 0                      'Zero, so it can be used as an accumulator
210   FOR J% = -50 TO 50             'Calculate the summation
220     Y[I%] = Y[I%] + X(I%+J%]
230   NEXT J%
240   Y[I%] = Y[I%]/101              'Complete the average by dividing
250 NEXT I%
260 '
270 END
```

Using Convolution

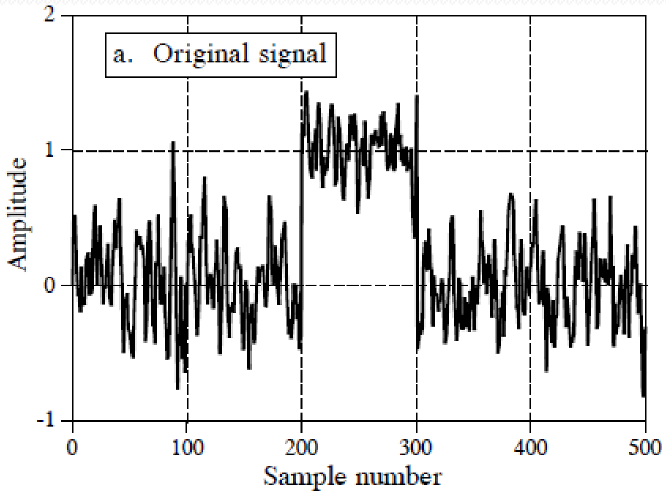
$$y[j]=\frac{1}{M}\sum_{i=n_0}^{n_0+M-1}x[j+i]$$

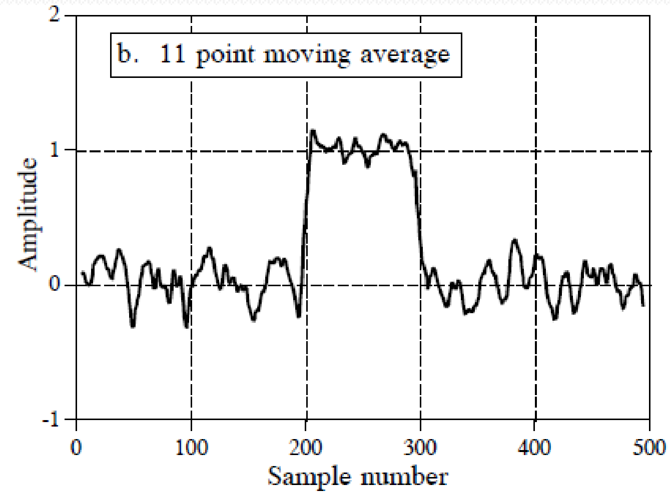
$$y[j]=\sum_{i=n_0}^{n_0+M-1}x[j-i]h[i]=\sum_{i=-n_0}^{-n_0+M-1}x[j+i]h[i]$$

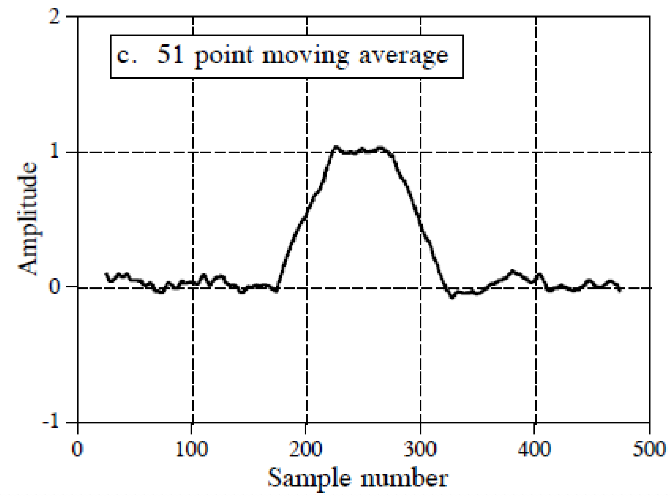
$$h[j]=\frac{1}{M}$$

n_o	Range of j
>0	-M+1-n_o:-n_o
0	-M+1:0
-(M-1)/2	-M/2:(M-1)/2
-(M-1)	0:M-1
<-(M-1)	n_o:n_o+M-1

What is it useful for?

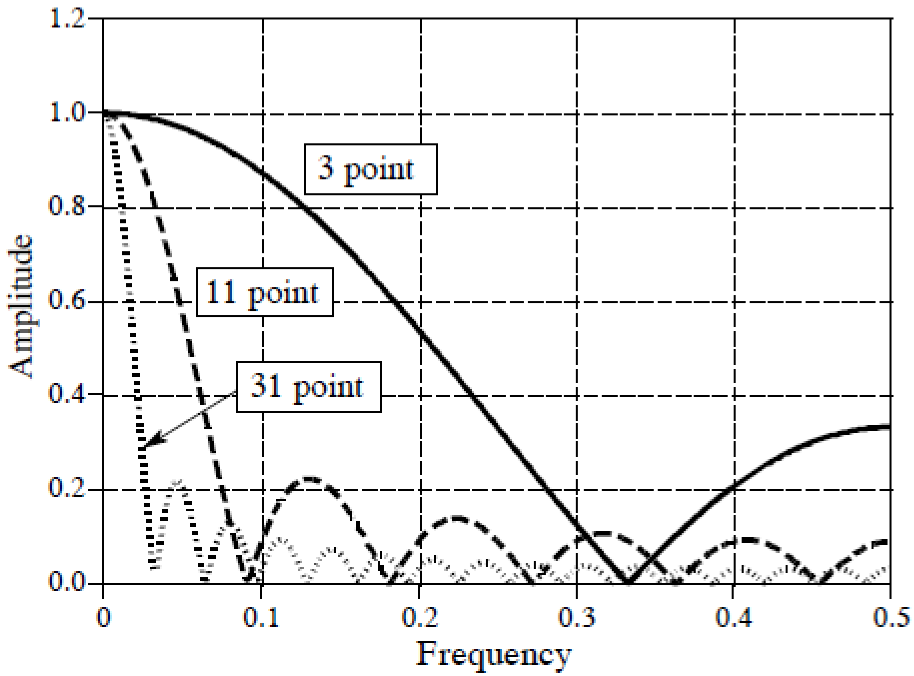






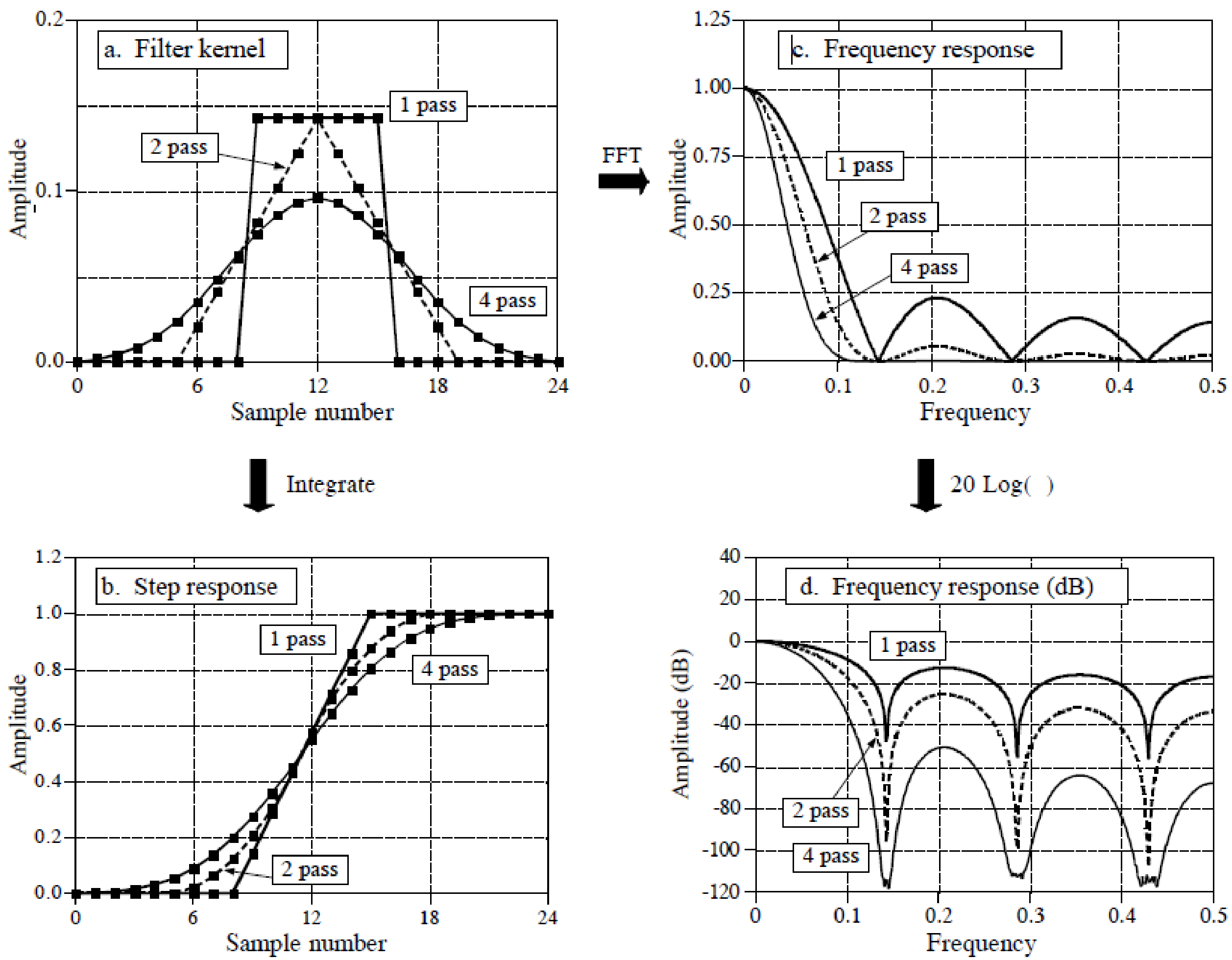
OPTIMAL for white noise reduction
while keeping the sharpest step response

Frequency Response



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

Moving Average Relatives



Moving average vs. its relatives

- Frequency domain
  - Moving average filters have worst stopband attenuation.
- Time Domain
  - Moving average give equal weight to all samples. Others taper near edges.
  - Moving average has sharpest step response but least smooth one.
- Speed
  - Moving average is the fastest and using recursive implementation it is the FASTEST digital filter at all.

Recursive implementation

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

where: $p = (M-1)/2$

$q = p + 1$

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

where: $p = (M-1)/2$

$q = p + 1$

Recursive Implementation

```
100 'MOVING AVERAGE FILTER IMPLEMENTED BY RECURSION
110 'This program filters 5000 samples with a 101 point moving
120 'average filter, resulting in 4900 samples of filtered data.
130 'A double precision accumulator is used to prevent round-off drift.
140 '
150 DIM X[4999]                    'X[ ] holds the input signal
160 DIM Y[4999]                    'Y[ ] holds the output signal
170 DEFDBL  ACC                    'Define the variable ACC to be double precision
180 '
190 GOSUB XXXX                     'Mythical subroutine to load X[ ]
200 '
210 ACC = 0                        'Find Y[50] by averaging points X[0] to X[100]
220 FOR I% = 0 TO 100
230   ACC = ACC + X[I%]
240 NEXT I%
250 Y[50] = ACC/101
260 '                              'Recursive moving average filter (Eq. 15-3)
270 FOR I% = 51 TO 4949
280   ACC = ACC + X[I%+50] - X[I%-51]
290   Y[I%] = ACC/101
300 NEXT I%
310 '
320 END
```