ELECTRONOTES
1016 Hanshaw Road
Ithaca, NY 14850

APPLICATION NOTE NO. 408

April 5, 2014

EXAMPLES OF INVERTING OF SMOOTHERS

INTRODUCTION

In the previous application note, AN-407 [1] we discussed how we might approach an "Un-Filtering" (deconvolution, equalization) problem relating to an often used Moving Average (MA) or Running Mean. In addition to the fairly direct attack of AN-407, a somewhat older app note, AN-366 [2] that used a least-squares approach was discussed as well, where the least-square FIR fit was found numerically.

While the approaches (and complication of noise, etc.) in AN-407 were typical, here the least-squares approach will be examined for two other cases: trapezoidal smoothing [3] and Savitzky-Golay [4]. So we will begin with a slightly more general approach to direct computation, and then see how least-squares works out.

DIRECT COMPUTATION

The direct computation simply asserts that H(z)/H(z)≡1. There is, however, a subtle difference when it comes to the actual implementation of this idea. The identity equation could be decomposed in many different ways [many - in the case where H(z) itself is factored]. We immediately think of H(z)•[1/H(z)] and [1/H(z)]•H(z)] as important cases, where we imply that the actual implementation occurs in a certain order. Of particular importance here is the fact that zeros and poles of H(z) interchange in 1/H(z). Since poles (and not zeros) are restricted to the interior of the unit circle, for stability, in using 1/H(z) as well as H(z) we seem to restrict the zeros to be inside the unit circle as well. [Famously, in traditional "frequency sampling implementation", we assert that if unit circle zeros are done first, we can place poles there after and weight the various paths for a desired response.]

Since here we are illustrating with FIR "smoothers" such as MA and Savitzky-Golay (SG) we, choose to consider H(z) to have a form:

$$H(z) = h_0 + h_1 z^{-1} + h_2 z^{-2} + \ldots + h_N z^{-N} \tag{1}$$

This of course gives:

$$\frac{H(z)}{H(z)} = \frac{h_0 + h_1 z^{-1} + h_2 z^{-2} + \ldots + h_N z^{-N}}{h_0 + h_1 z^{-1} + h_2 z^{-2} + \ldots + h_N z^{-N}} \tag{2}$$

which in turn makes the inverse 1/H(z) to be:

$$\frac{1}{H(z)} = \frac{1}{h_0 + h_1 z^{-1} + h_2 z^{-2} + \ldots + h_N z^{-N}} \tag{3}$$

and this is not acceptable for and IIR structure (meaning Fig. 1) unless $h_0 = 1$. This is easily fixed by multiplying top and bottom by $1/h_0$:

$$\frac{1}{H(z)} = \frac{1/h_0}{1 + (h_1/h_0) z^{-1} + (h_2/h_0) z^{-2} + \ldots + (h_N/h_0) z^{-N}} \tag{4}$$

which has the same poles.

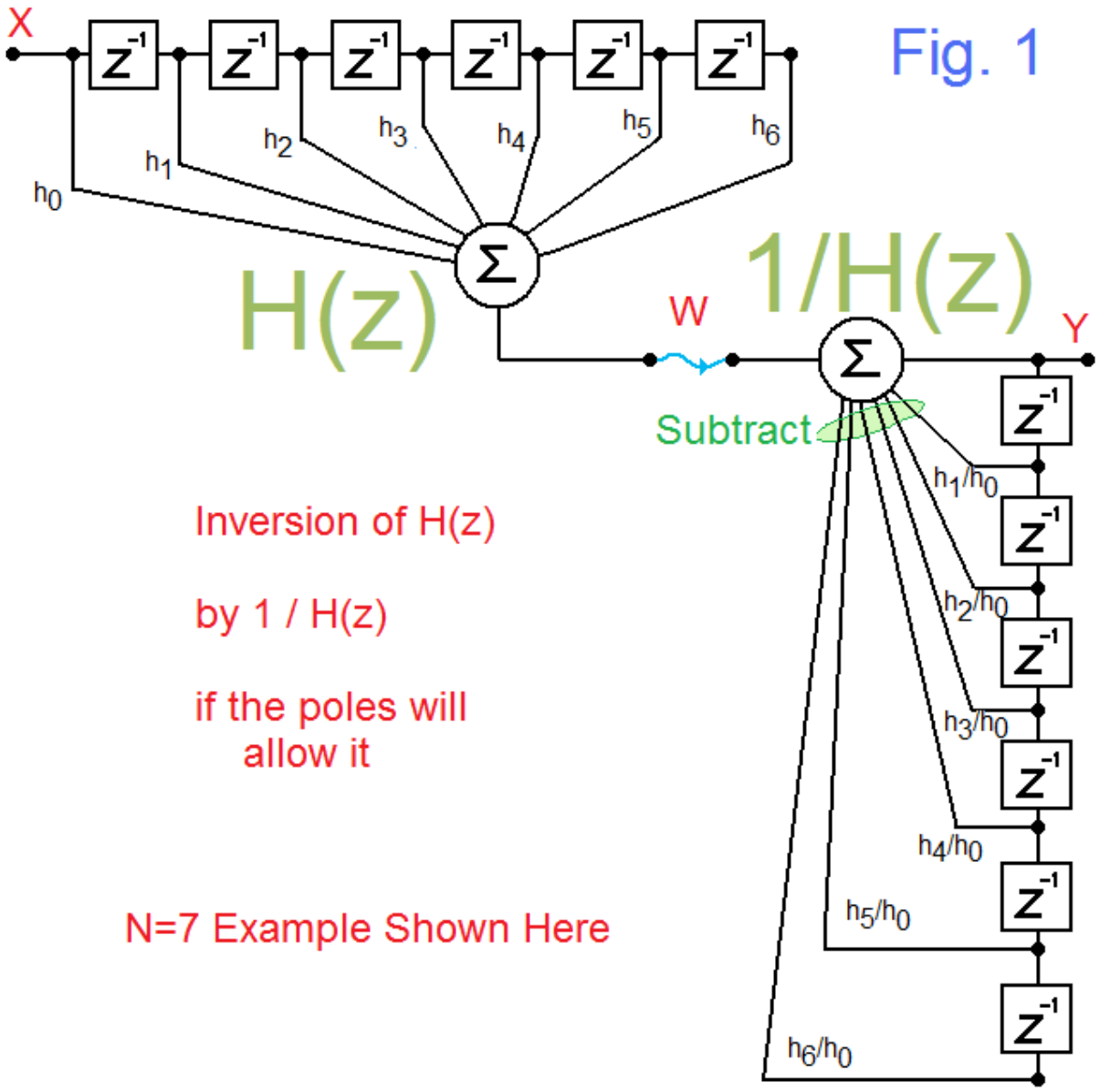


Fig. 1

As stated [1, 2] there are possible problems with this direct approach relating to possible instabilities and possible noise or both. Any H(z) that has zeros on the unit circle (e.g., MA), would lead to problematic, conditionally stable unit circle poles. Worse, zeros of H(z) outside the unit circle (as SG and other linear-phase designs will have) will not work with this IIR inversion, and we must use the approximate FIR inversion [2].

A more immediate problem but one for which simple solutions are at hand is that it is not easy to write expressions for the impulse response of 1/H(z) in a closed from. This would be the problem of calculating the inverse z-Transform of equation (4). On the other hand, we can easily use a Matlab function such as ***hiir =filter(N,D,[1 zeros(1,99)])*** where N is just the numerator coefficients (which is just a single $1/h_0$ in this case) and D is the denominator coefficients (h/h_0) and [1 0 0 …..0] is a stand-in for a single impulse. (No Matlab., etc.? It's trivial to write your own code.) To evaluate results, we just try different values and see if the response dies off fast enough. Or, it may be the case that the response does not die off, or blows up, in which case we hope our finding agrees with any determination of pole positions we have found.

EXAMPLE 1 - MOVING AVERAGE

Here we will look at the case of a length-10 MA to see if it can be inverted with an IIR approach (as just above) and/or a FIR least-square approach [2]. Further we will choose a noise-free test signal for this, and for that which follows later in this note, consisting of a length-20 rectangle and then two cycles of 15 samples each of a sinewave (Fig. 2a).

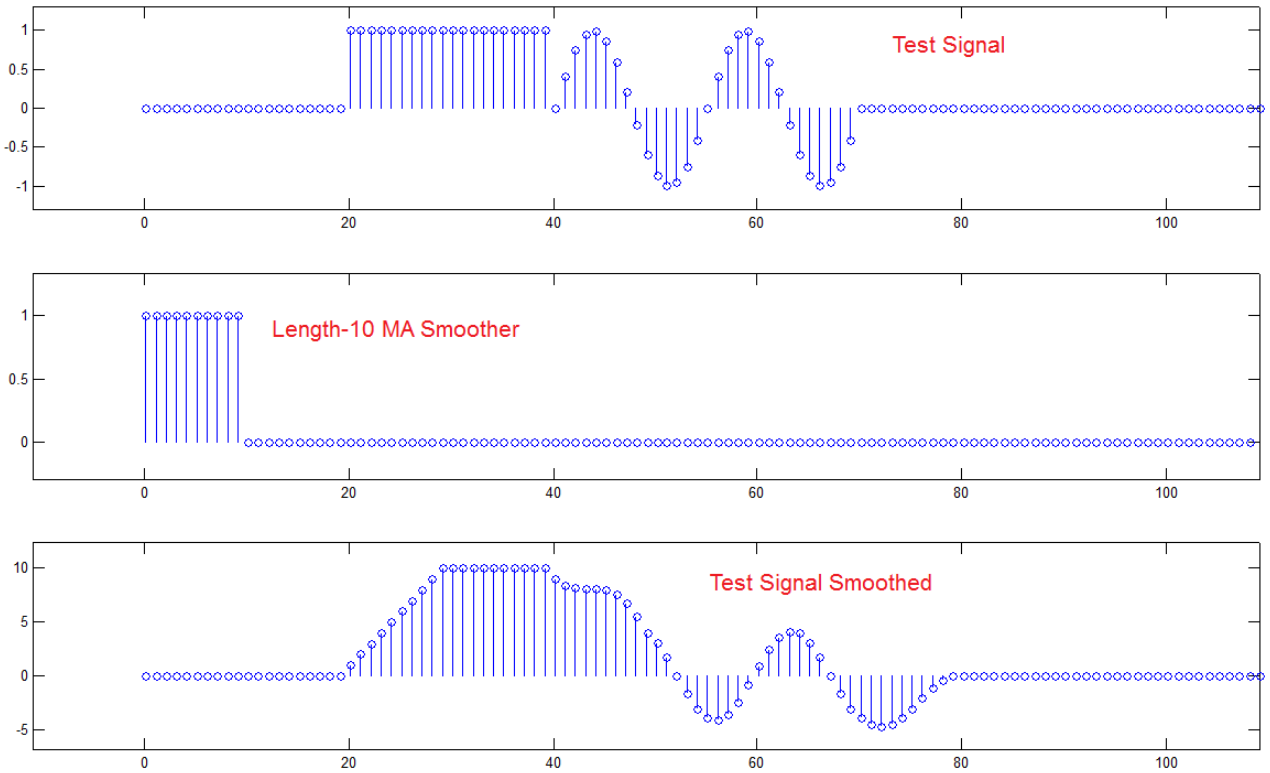


Fig. 2a

As we see clearly in Fig. 2a, the original test signal is smoothed (smeared if you prefer here) by the length-10 MA (middle of Fig. 2a) producing the result at the bottom of Fig. 2a. Note, as two examples of what happened, the gradual ramp-up of the rectangle, and the extended width of the last sinewave lobe. Our goal is to now reverse this filtering.

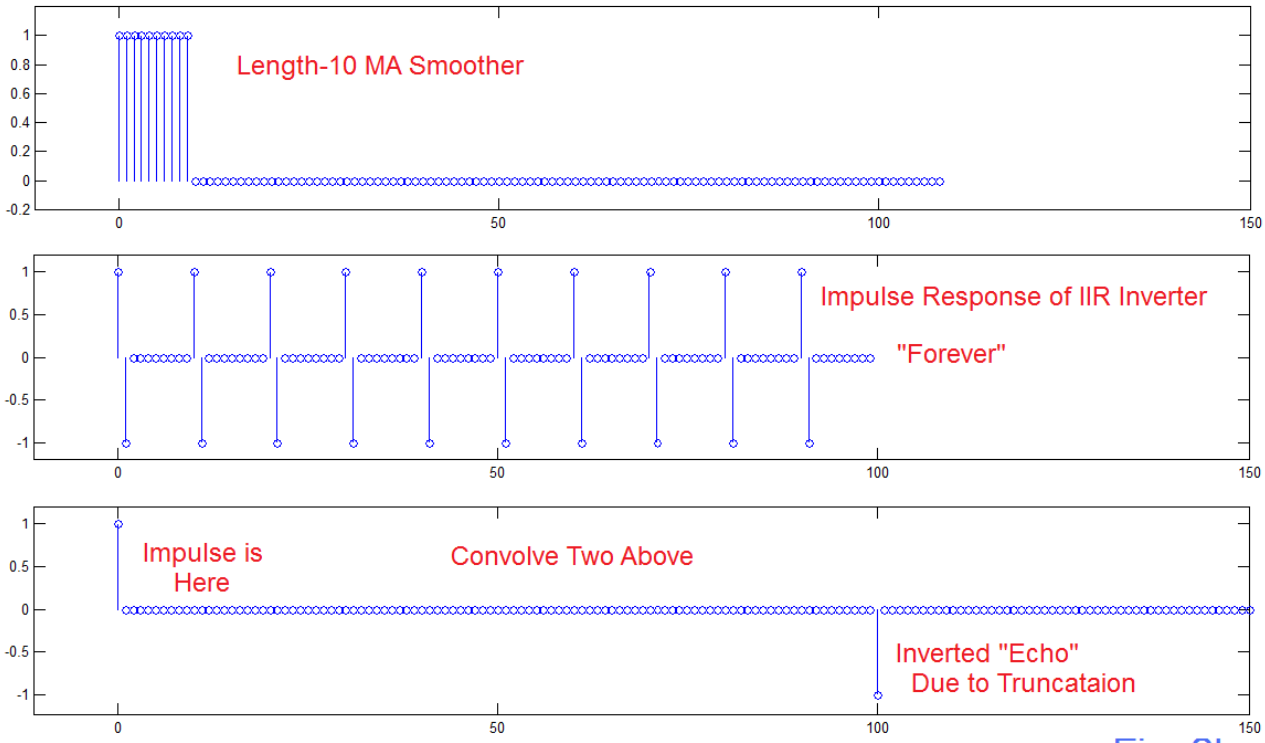


Fig. 2b

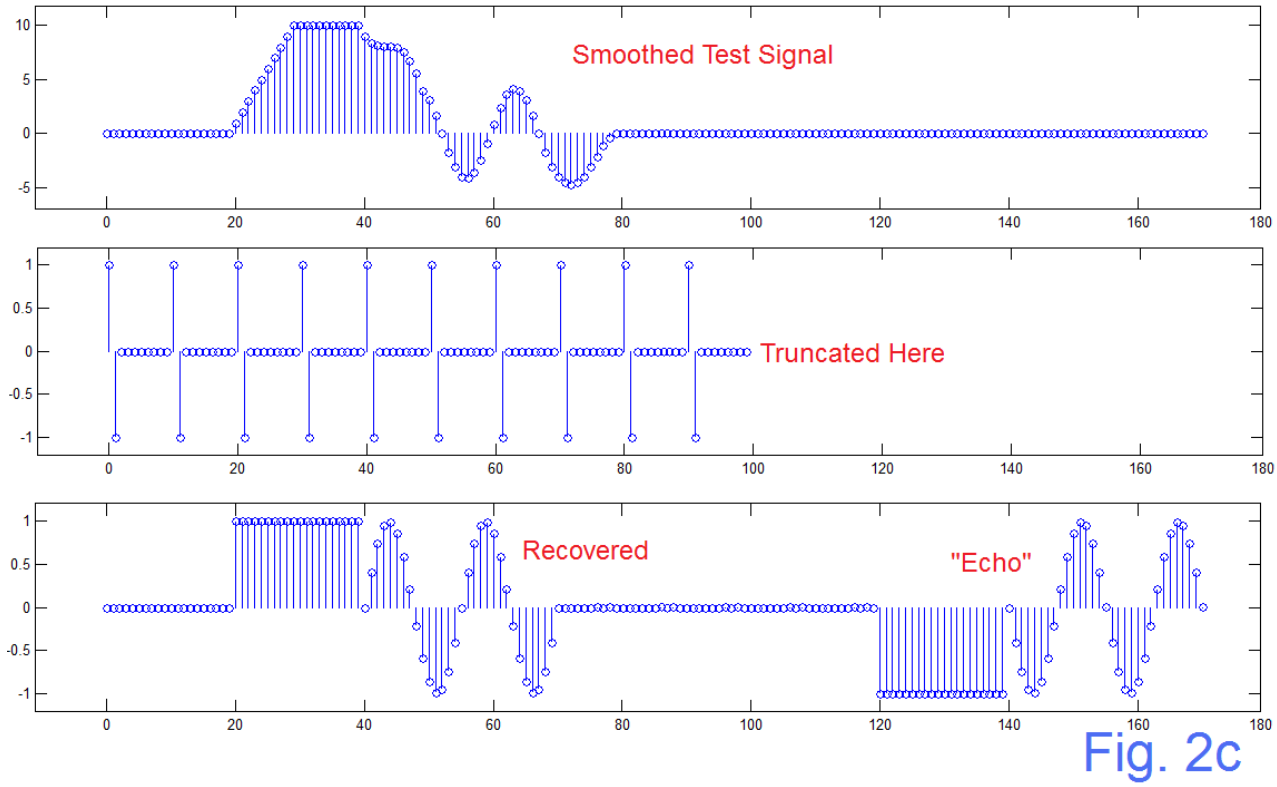


Fig. 2c

Equations (1) and (4) with all h(n)=1 apply here, and this is the problem of the previous app note [1]. In Fig. 2b, we show the calculated IIR impulse response. Actually, we show the first 100 values of the impulse response, as it goes on forever without decay. Convolving the FIR smoother with the IIR inverter should give us an impulse at n=0, and it does (bottom line of Fig. 2b, at extreme left). We also note the "echo" response that was due to the truncation of the IIR inverter response. One goal is to make sure the IIR response is long enough (exceeding the combined length of the smoothing convolution by a comfortable margin). That is, it should exceed the length of the test signal plus the length of the smoother minus 1; and then add some extra safety space.

Fig. 2c shows the resulting recovery. The top panel of Fig. 2c is the smoothed input, the same as the bottom panel of Fig. 2a. The middle panel of Fig. 2c is the truncated IIR response, a truncated form of the "forever" suggested in Fig. 2b (middle panel). Convolving these two gives us, perhaps surprisingly the bottom panel of Fig. 2c, which is a recovery of the original input as seen on the left side with an inverted echo on the right side. This is quite tricky to see and to believe. To see this "working" we have provided in the **Appendix** a block-by-block, shift-by-shift version of a simpler case.

It is tempting to try to write out a recipe for doing this inversion successfully. In fact, we have probably done more than enough to illustrate how to do this. Yet when it gets right down to attacking real data, there is a lot going on. So the recommendation would be to set up the programming for your particular problem, test it, and then don't believe anything that does not look reasonable until such time as you retry the same code on some well defined data which you can easily monitor, step-by-step.

A perhaps safer approach is to use the least-squares FIR equalizer method [2] where we do not have to deal with the IIR aspects of the problem. This we expect to be an approximation, and the result has only z-plane zeros to deal with (and often, a mess of them – see Fig. 2f). In this case, we have the same length-10 smoother, and we seek (our choice) a length 100 FIR equalizer. The code for the least-squares approach was presented in [2] and an updated version is at the end of this note.

Fig. 2d shows the result of this computation, which has a resemblance to the IIR impulse response as we might expect. The FIR response is non causal – indeed it is linear phase which we often desire anyway. Thus it has a delay, which is often not an issue as smoothed data is not always (or usually) running time, or even has time as an independent variable. Note the particular symmetry, and the expected tapering. [Recall from [1] the artificial tapering of the IIR response by moving poles slightly inside the unit circle.] The convolution of the original smoother with the FIR equalizer should give us an impulse. We have a length 100 equalizer, which we see tapers but evidently has non-zero residuals. In consequence, the recovered impulse (bottom panel of Fig. 2d) is imperfect. Note the delay that was expected.

The recovery with the FIR equalizer is seen in Fig. 2e, and is relatively good. Fig. 2g shows the frequency responses of the smoother, of the equalizer, and their product. Note the "lack of interest" of the FIR equalizer in compensating for the smoother nulls! In fact, it puts additional nulls there – hence the rounded bottoms of the nulls in the product. The product (green) is nonetheless showing an interest in approximating 1.

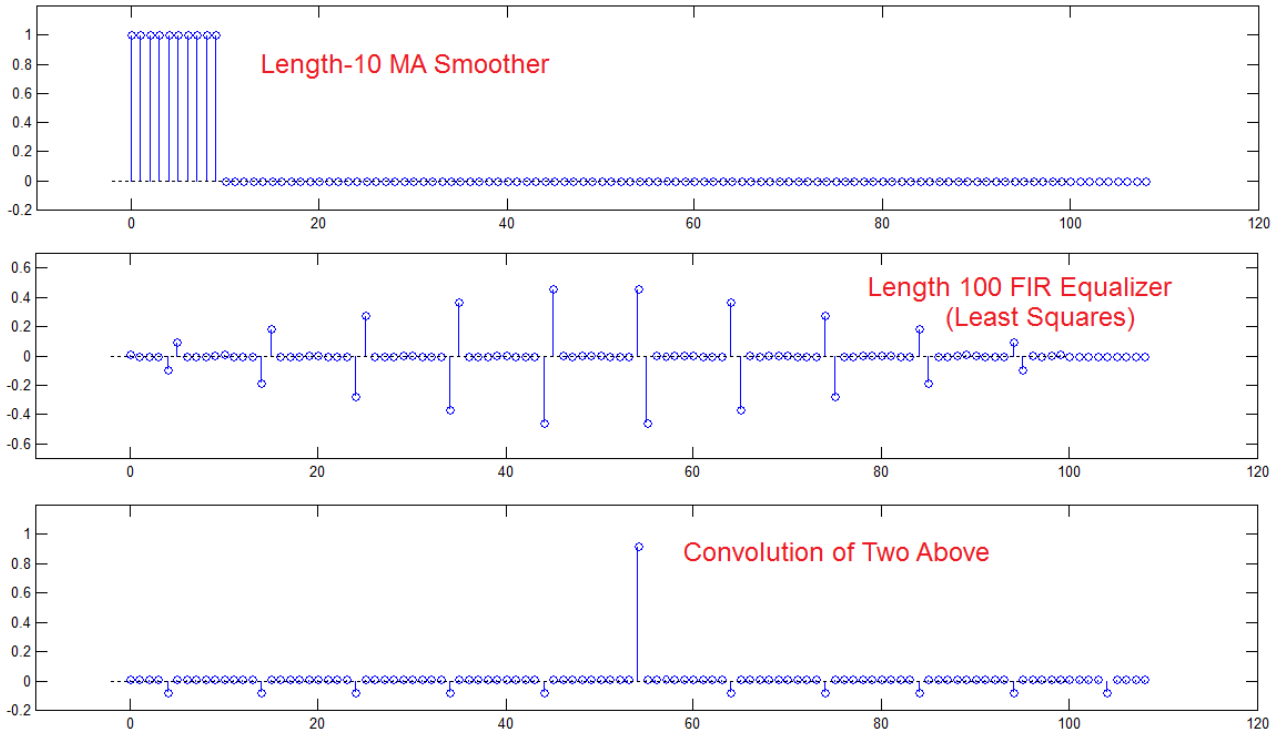

Length-10 MA Smoother
Length 100 FIR Equalizer
(Least Squares)
Convolution of Two Above

Fig. 2d

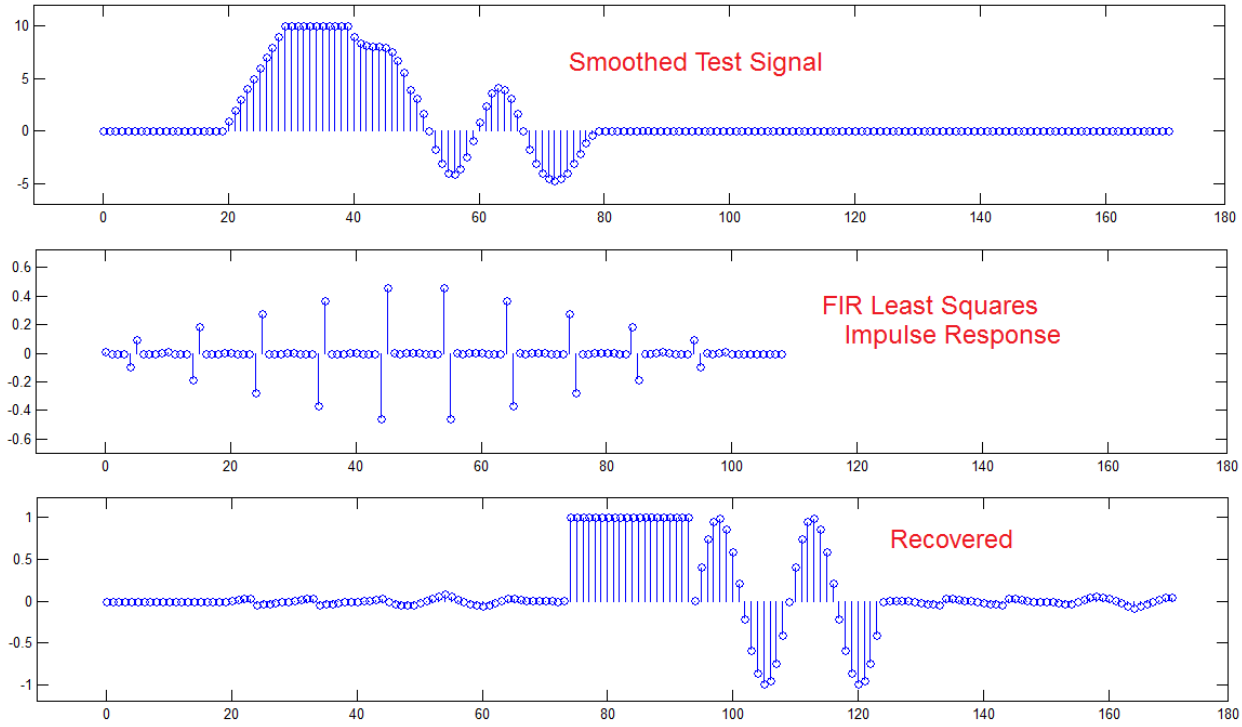

Smoothed Test Signal
FIR Least Squares
Impulse Response
Recovered

Fig. 2e

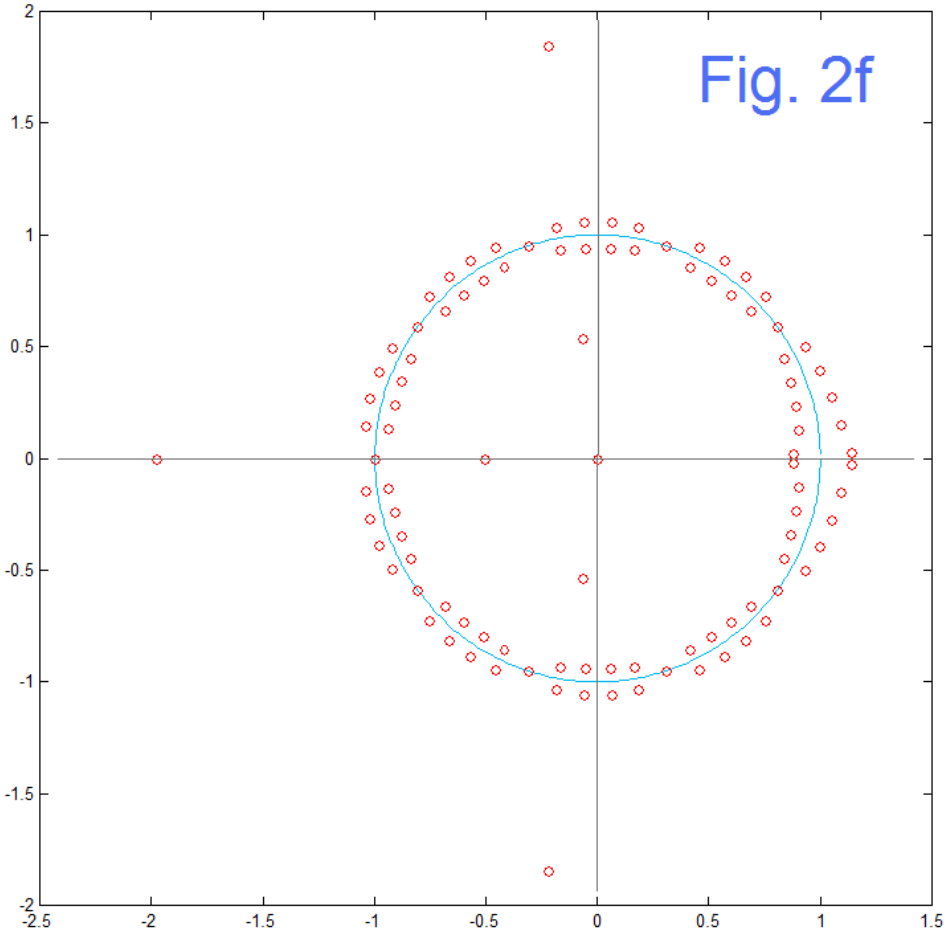


Fig. 2f
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2.5
-2
-1.5
-1
-0.5
0
0.5
1
1.5

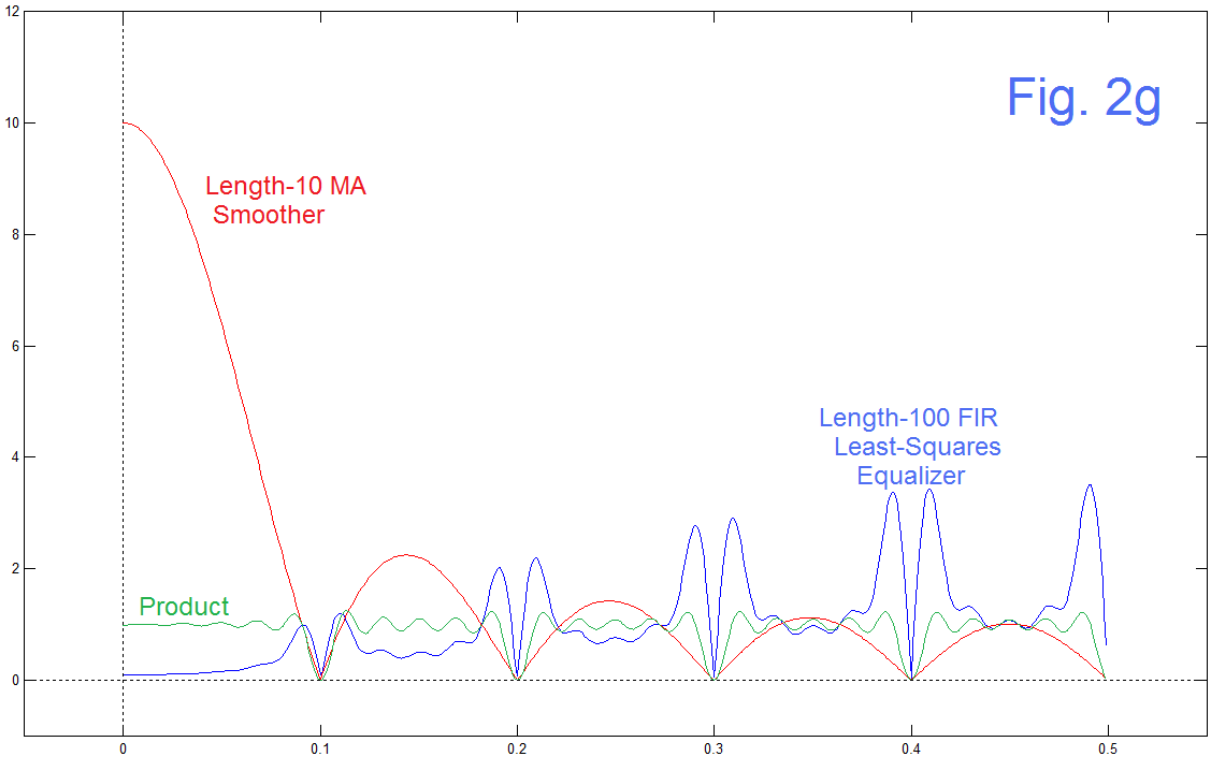


Fig. 2g
Length-10 MA
Smoother
Length-100 FIR
Least-Squares
Equalizer
Product
12
10
8
6
4
2
0
0
0.1
0.2
0.3
0.4
0.5

NOISE

Here we have found that the ubiquitous rectangular MA smoother has yielded not only to the least-squares approach in providing a stable FIR inverter, but even an IIR inverter seems possible in the absence of noise. We need to be clear about noise. A signal may well be noisy in the sense that obtaining it involves measurement errors and or roundoff errors. It may also be the case that the signal "looks like noise" in that it looks like a lot of random stuff and that if there are certain trends and components, they are small compared to the noise. Things as crude as MA smoothers are often used with such noise-like signals. This is not the "noise" that we worry about with messing up reconstruction.

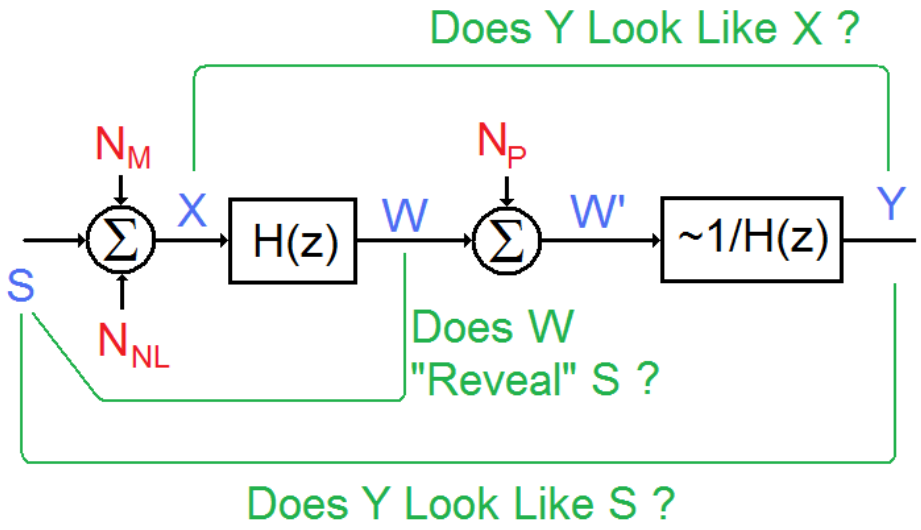


Fig. A

Fig. A shows a larger picture where we suggest that there are three types of "noise" here. We have in mind that there is a "true" signal S that is noiseless. The signal data X we have on record includes S, but there may be substantial measurement errors (N_M) and in addition, the data itself may be "Noise-Like" (N_{NL}) due to confounding influences and possibly chaotic behavior, etc. In such cases, we may be employing a smoother H(z) in an attempt to make the hidden S, in some sense, more apparent (Does W "Reveal" S ?) or does H(z) reduce N_M and N_{NL}? At this point, any inversion by 1/H(z) is not even involved.

The complications with 1/H(z) may be an ongoing problem, such as we have suggested would be the case where H(z) is not something we have intentionally put in, but something like a communications channel. We need 1/H(z) to fix undesired distortions in the channel. The channel may well involve extra random noise, a processing noise, N_P. In the case of intentional smoothing, N_P may be such things as data recording noise (like roundoff - a table keeps number to only one decimal place) or recovery from a plotted graph. The question here is (Does Y Look Like X ?). It depends on how bad N_P is, and on whether or not 1/H(z) is satisfactory. Perhaps we just want X back to try a different smoothing. The third question (Does Y Look Like S ?) is not a primary question here.

TRAPEZOIDAL

The apparent success with the rectangular MA might well encourage us. In particular, as we have discussed in the section above on NOISE, we may have no N_P of any real consequence, and just want X back for another try – perhaps a different smoothing. In another app note [3] we suggested a ploy of increasing the length of the MA by 1 and making the end taps 1/2 instead of 1. The idea was primarily to place the output at the center in the case of what started as even length where there would have been a 1/2 sample offset. For example, our length-10 MA with impulse response [1 1 1 1 1 1 1 1 1 1] would be modified to length-11 as [1/2 1 1 1 1 1 1 1 1 1 1/2] which is trapezoidal. We saw [3] several advantages to this. It might seem a minor change.

In fact, there is a major complication here. We can still do a least-square inversion quite satisfactorily, but the IIR inversion does not work. It can be seen that the length-11 trapezoid is the convolution of a length-10 rectangle with a length-2 sequence [1/2 1/2]. This adds an extra zero (that we expected to obtain), and puts it at z=-1, which was already occupied by an original zero from the length-10. Hence we end up with a second-order zero at z=-1, which actually makes a better FIR low-pass. However, in the IIR inversion, we get a second-order pole on the unit circle which does blow up. [Recall that a unit-circle pole is generally "conditionally stable", neither causing a blow-up or a decay. A second order pole blows up.] This is the simple reason the IIR inverter does not work.

In making graphs, we could run the same Matlab program that produced the series of Figures 2a through 2g. Here we will do this run, and for the trapezoid, number them with a 3 instead of a 2. (Shortly we will use SG and use a numbering starting with 4.) However, while we will use the same letter for corresponding graphs, we do not need all the graphs. Only about half of them are included.

The smoothing by the trapezoid is very similar to Fig. 2a, but when we compute the impulse response of the IIR inverter, it blows up, as seen in Fig. 3b. We could perhaps work around this for a range of input samples, but numerical problems soon come in.

Fig. 3d shows the case where the least-squares FIR inverse is computed, and we see a reasonably-well recovered impulse in the bottom panel of Fig. 3d. The actual structure of the impulse response is curious (steps of 5 here!) and would require somewhat more study to understand. It is not always similar, depending on the number of ones between the 1/2 ends, particularly as they change from an even to an odd number. For odd numbers of ones (like the 9 here), we seem to have steps. For even numbers of ones, half the steps are more or less gone. Curious.

The key result though has to be whether or not the inverter does a reasonable job of undoing the smoothing, and this is seen in Fig. 3e, where there is a credible result.

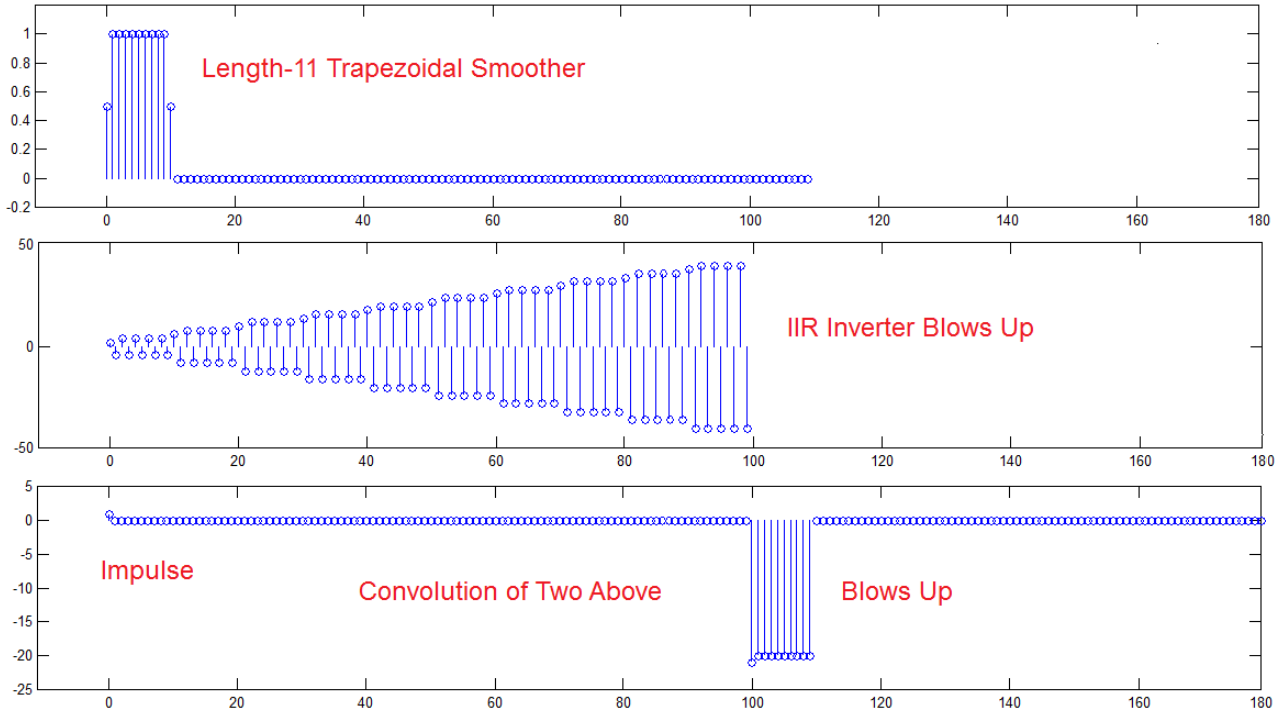

Length-11 Trapezoidal Smoother
IIR Inverter Blows Up
Impulse
Convolution of Two Above
Blows Up

Fig. 3b

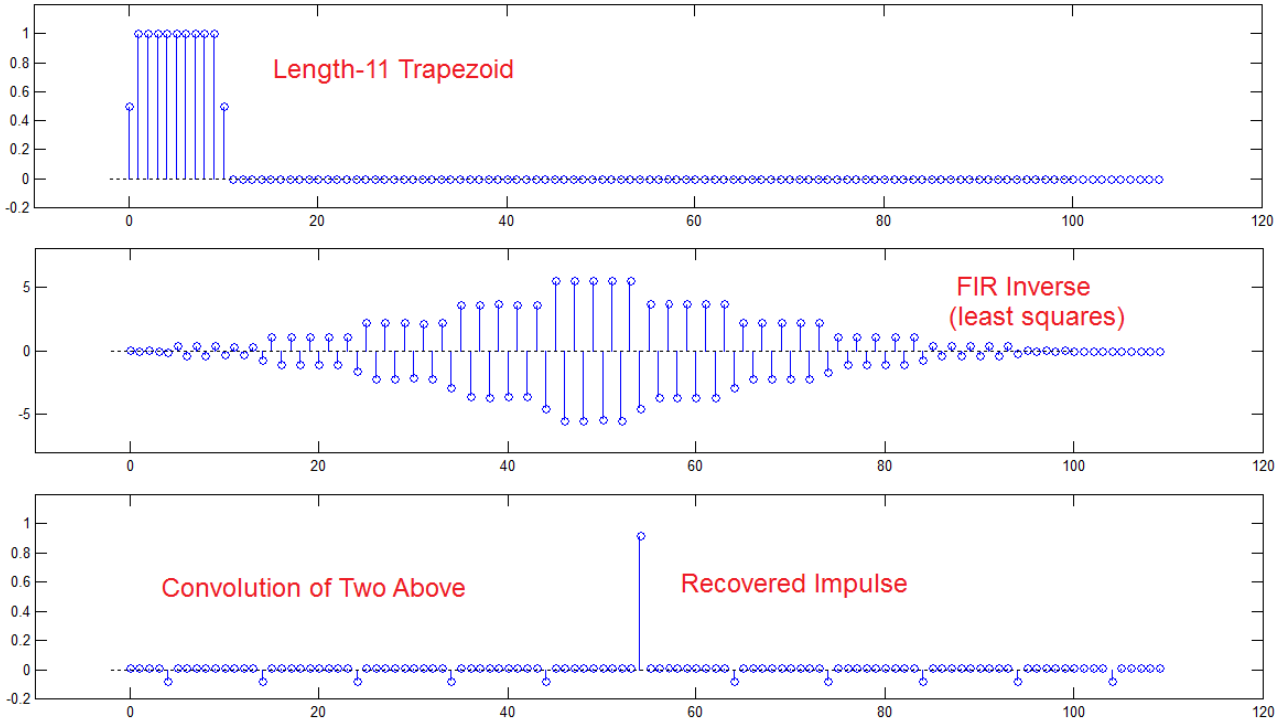

Length-11 Trapezoid
FIR Inverse
(least squares)
Convolution of Two Above
Recovered Impulse

Fig. 3d

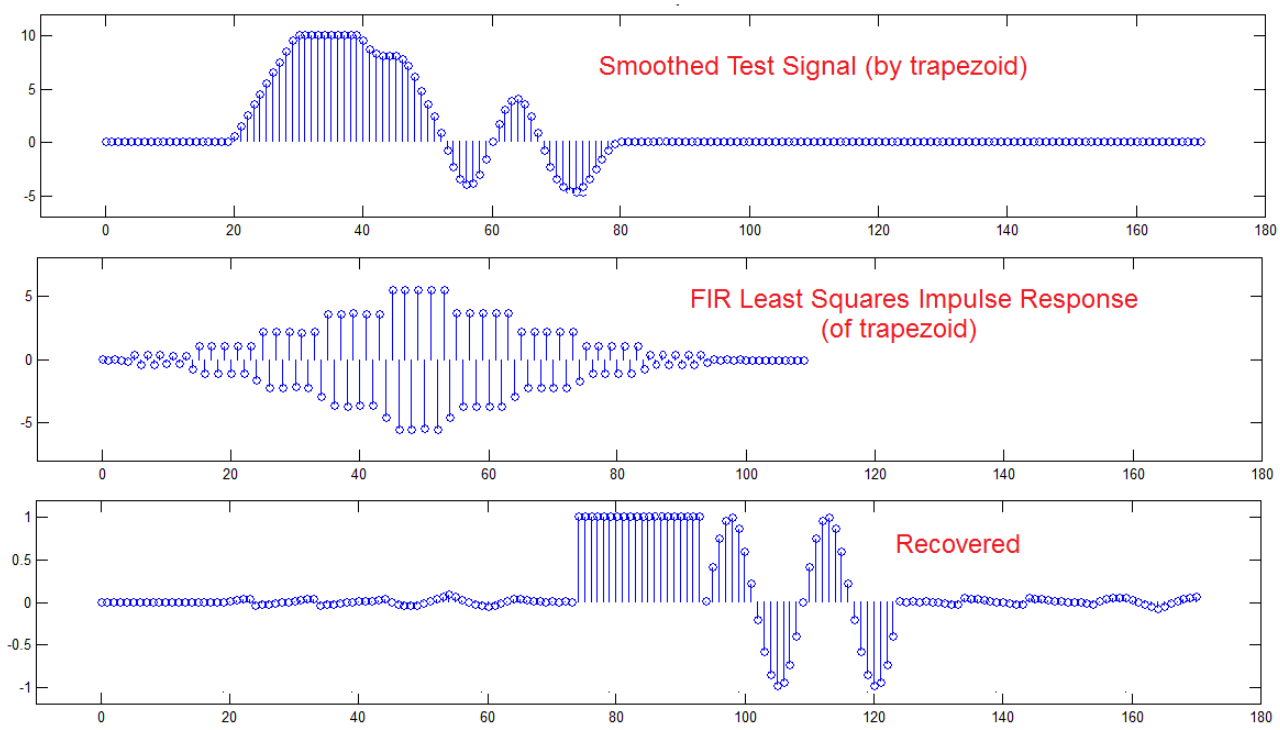
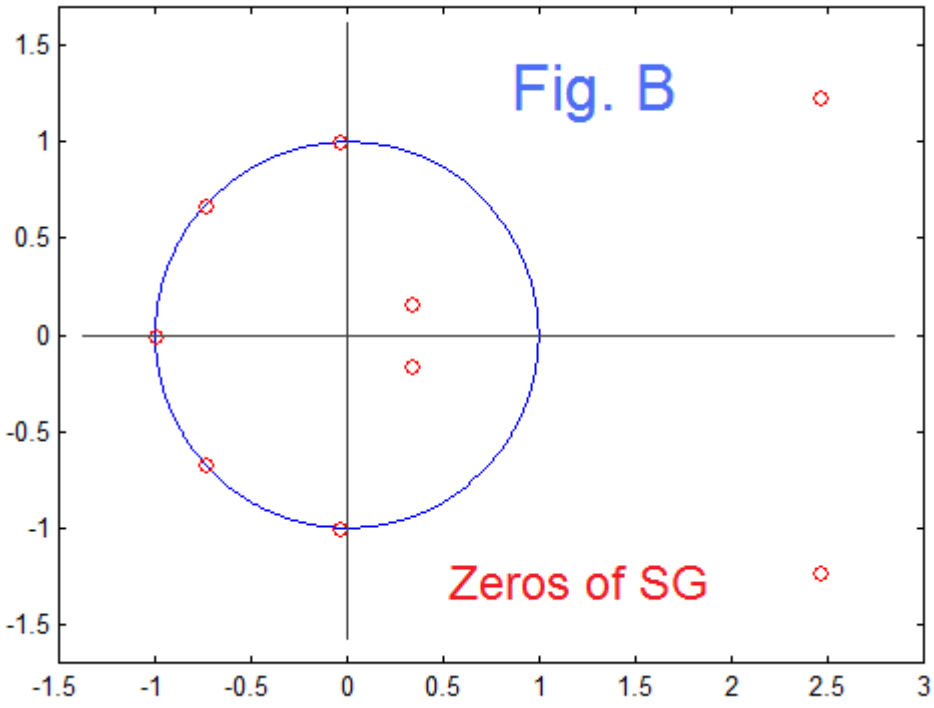


Fig. 3e

SAVITZKY-GOLAY

Our third example of a FIR smoother will be the Savitzky-Golay [4]. Once again we can just plug in the impulse response and out will come the graphs a to g. These will have the number 4, and we will only include a few of them here. It is easily seen that the direct IIR inversion will have problems even worse than the trapezoid: the SG is linear phase with some zeros inside the unit circle, demanding zeros outside at reciprocal positions,



and thus the IIR inverse will include poles outside the unit circle (Fig. B). The examination begins with the SG program ***hsg=sg(5,10)*** from [4].

hsg = [0.0391 -0.1172 0.0117 0.2148 0.3516 0.3516 0.2148 0.0117 -0.1172 0.0391] (5)

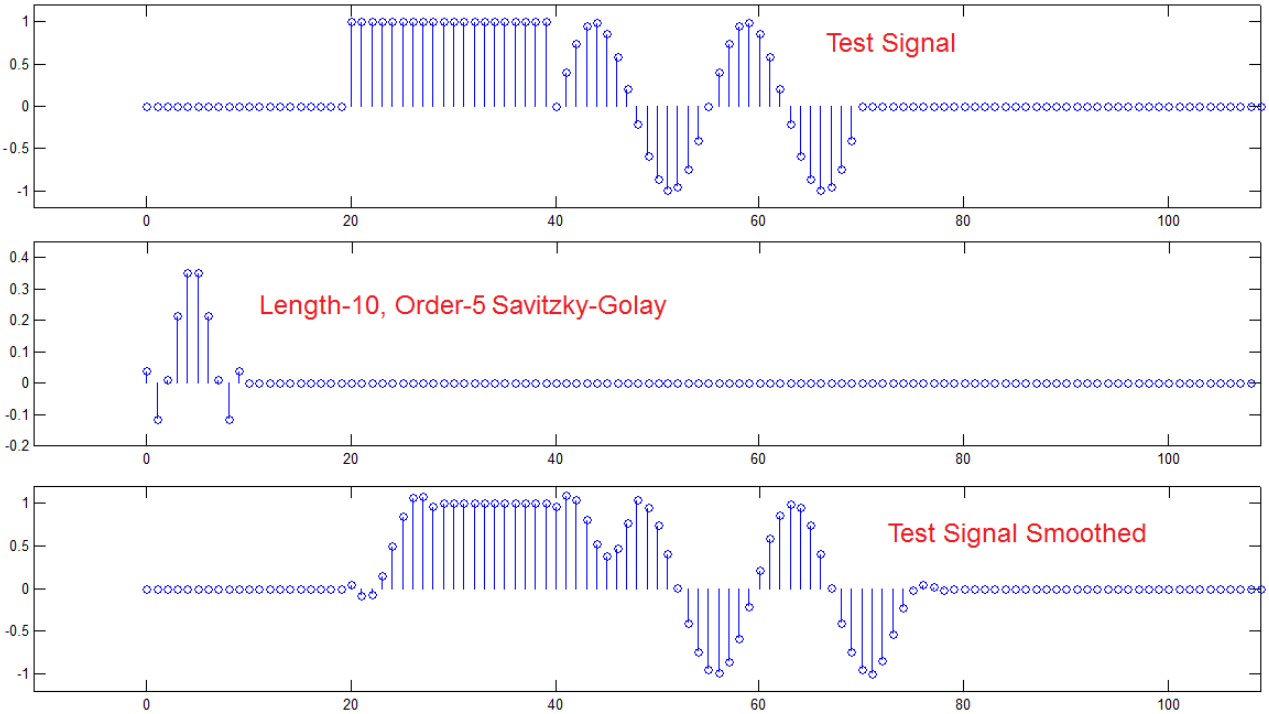

Test Signal
Length-10, Order-5 Savitzky-Golay
Test Signal Smoothed

Fig. 4a

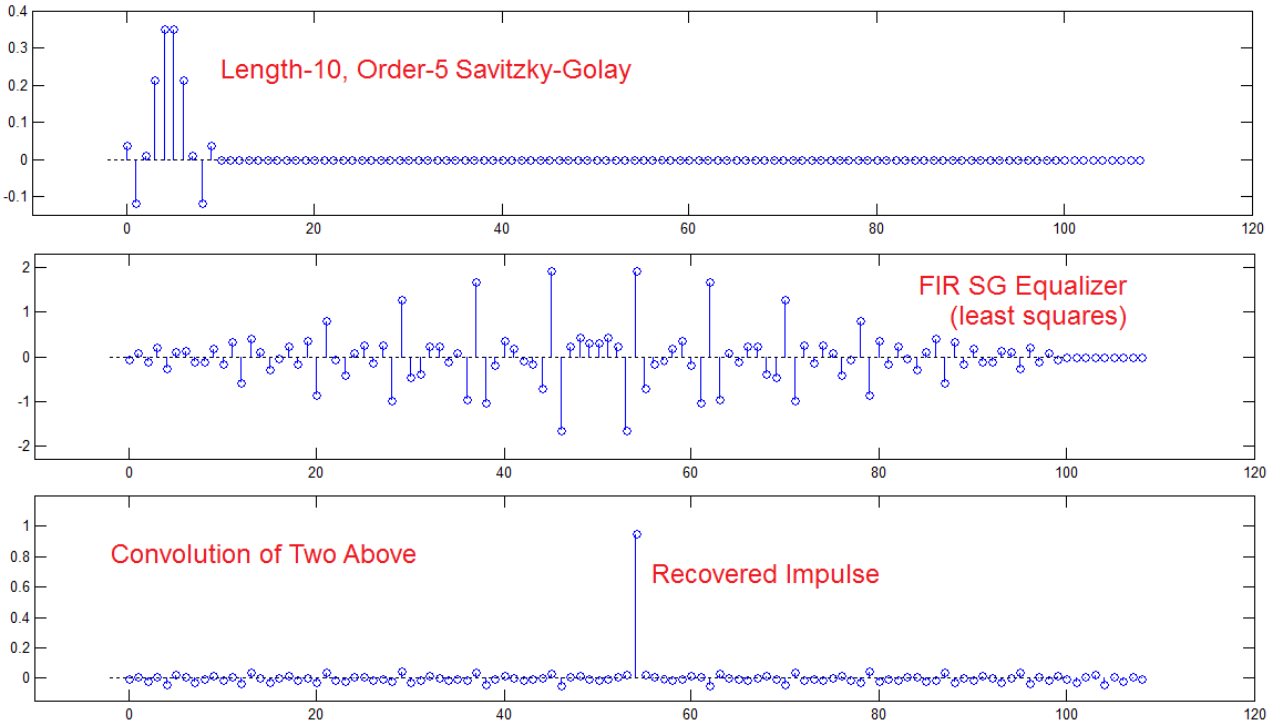

Length-10, Order-5 Savitzky-Golay
FIR SG Equalizer
(least squares)
Convolution of Two Above
Recovered Impulse

Fig. 4d

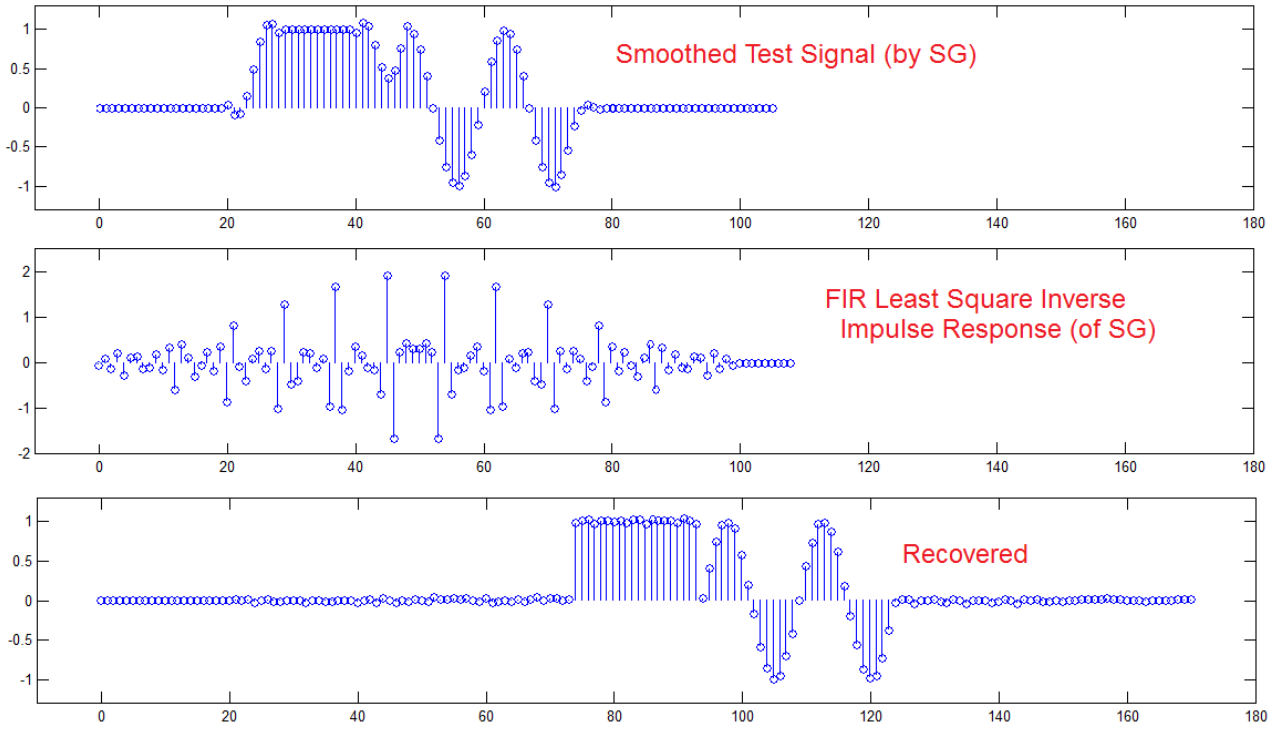

Smoothed Test Signal (by SG)
FIR Least Square Inverse
Impulse Response (of SG)
Recovered

Fig. 4e

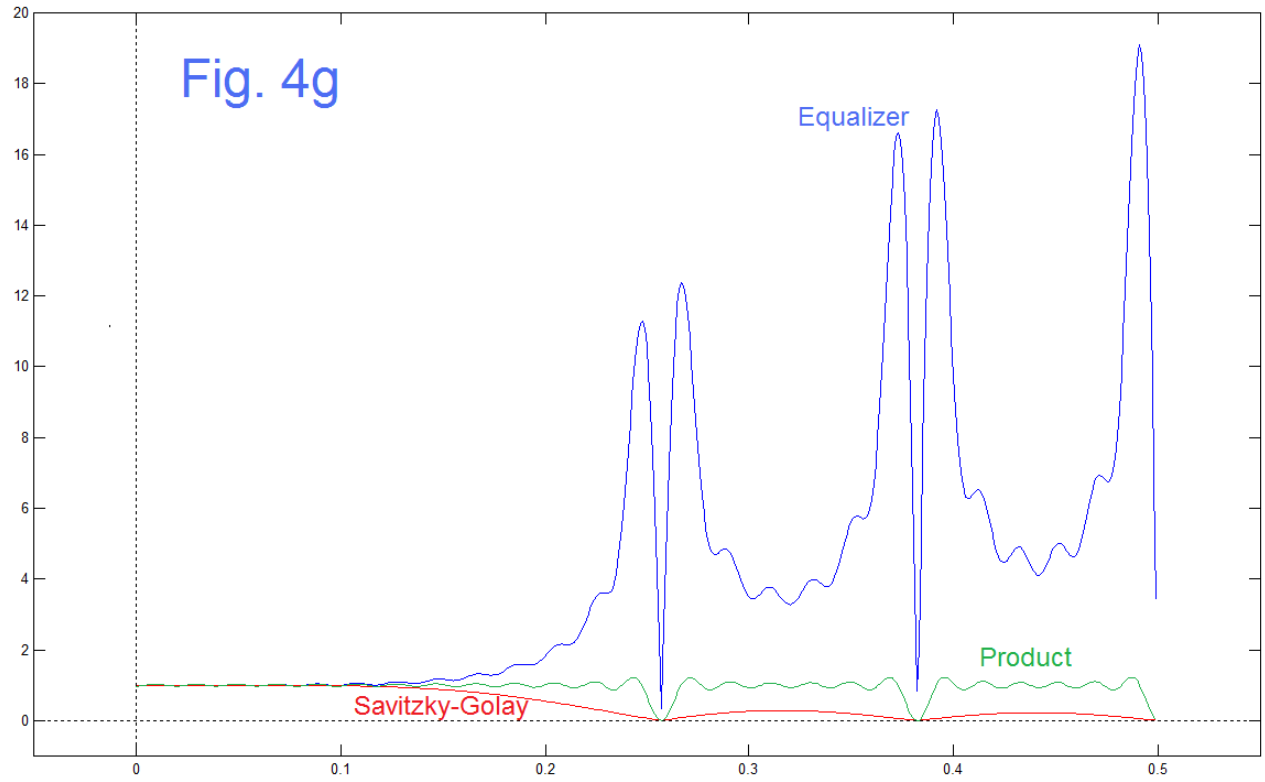


Fig. 4g
Equalizer
Product
Savitzky-Golay

In Fig. 4a, we have shown the shape of the smoothed test signal corresponding to the FIR smoother of equation (5). Note that we see less tapering of the edges of the rectangle, corresponding to the higher frequencies allowed by the SG (see Fig. 4g, red curve, or [4]). At the same time, there is far less elongation of the final lobe of the second sinusoidal cycle (Fig. 4a as compared to Fig. 2a).

As suggested, the IIR inversion fails. In fact, the calculation shows a blow-up at the far end amounting to 10^{44}! We do not explore this further – we expected poles outside the unit circle to blow up (Fig. B shows the zeros, and the poles are in the same locations). Fig. 4d shows however that the FIR least-squares methods still works just fine, with the impulse response in the middle panel and the recovered impulse in the bottom panel.

Continuing, we see that the FIR equalizer works quite well in recovering the original test signal (bottom panel of Fig. 4e) where we see some "jitter" but note that the overshoot on the rectangular corners is removed.

REFERENCES

All here written by B. Hutchins for Electronotes

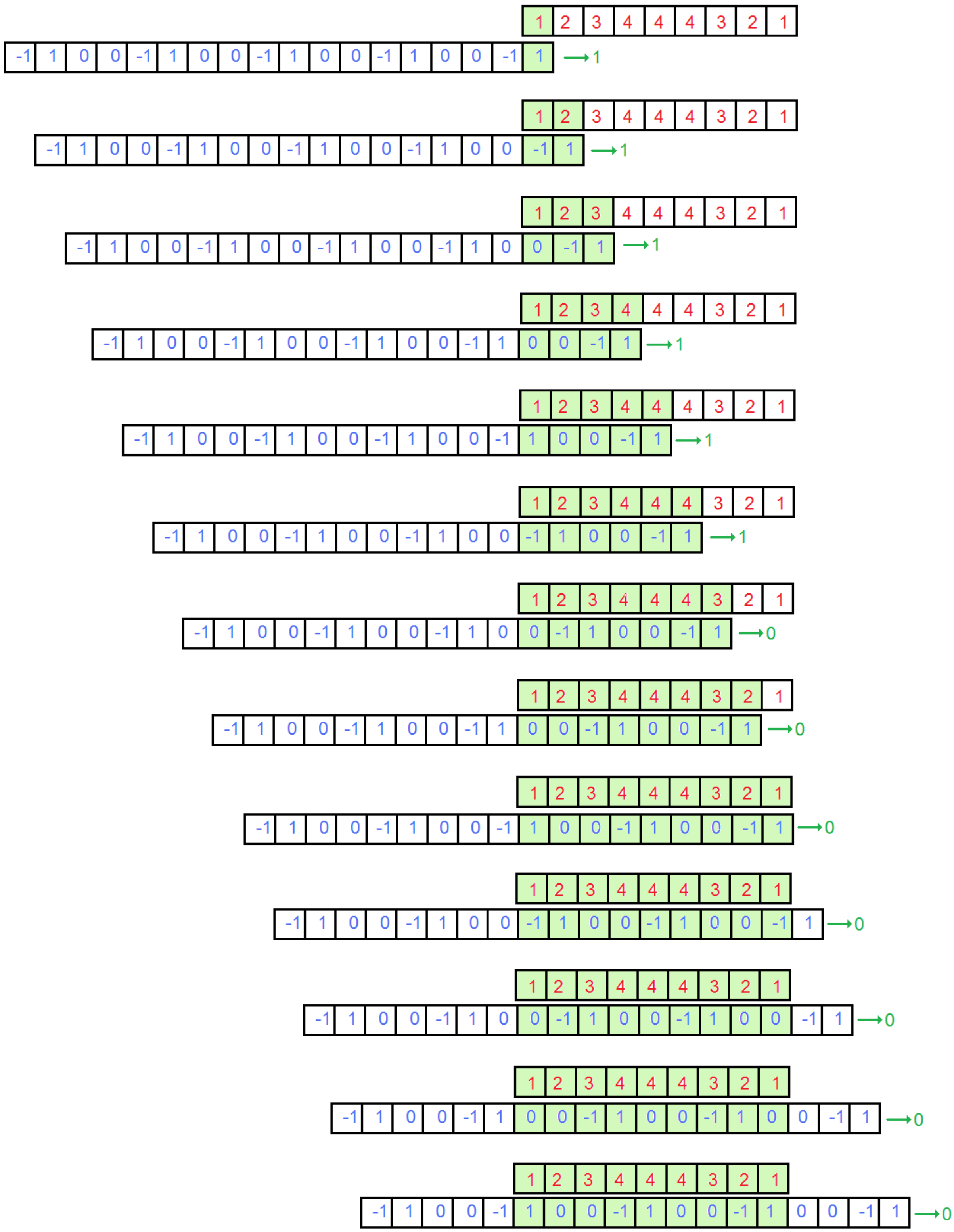
[1] "Inverting a Moving Average Operation" Electronotes Application Note AN-407, March 30, 2014
http://electronotes.netfirms.com/AN407.pdf

[2] "Unfiltering – Equalization", Electronotes Application Note AN-366, May 2006
http://electronotes.netfirms.com/AN366.pdf

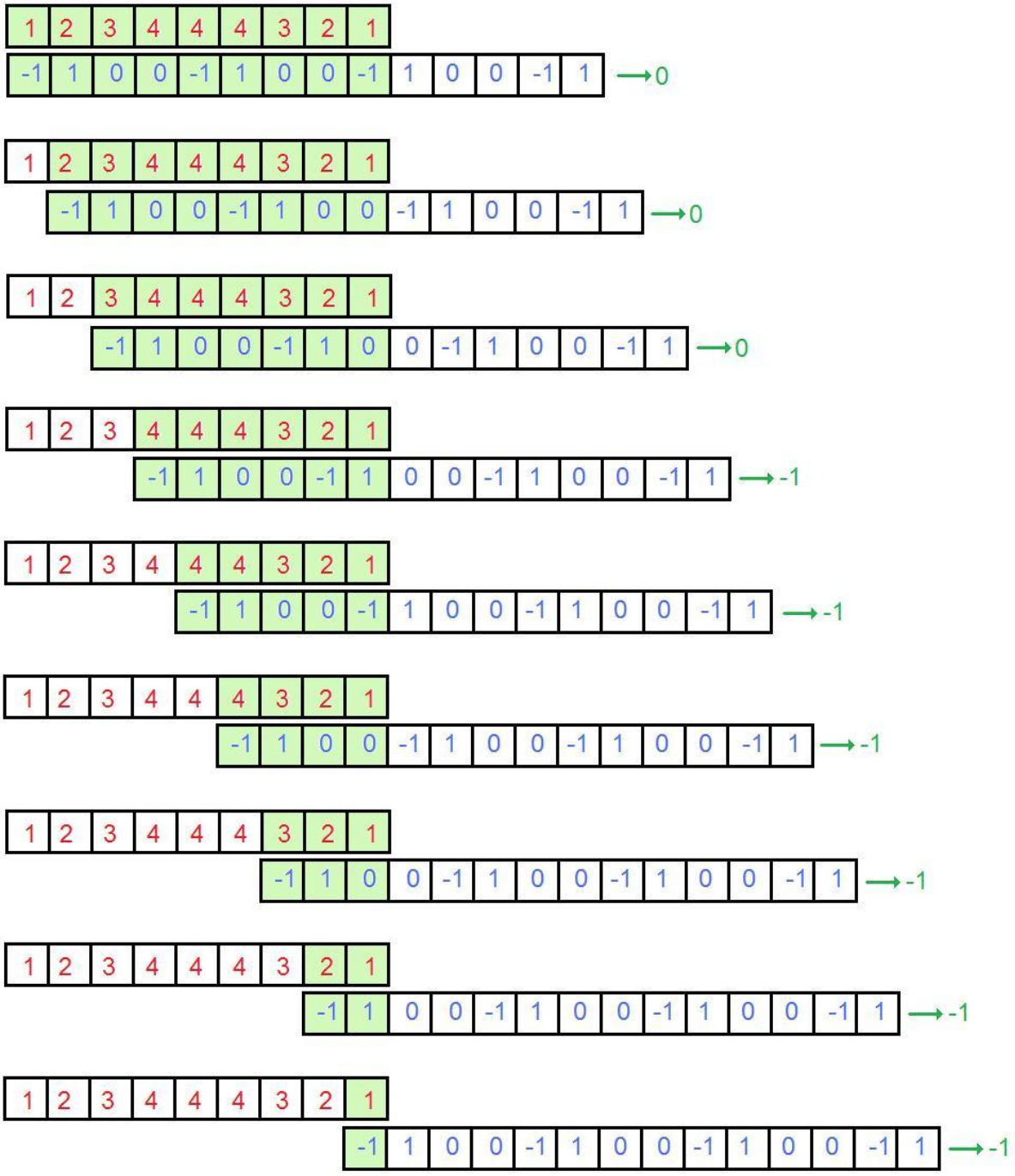
[3] "Yearly Moving Averages as FIR Filters", Electronotes Application Note AN-401, Dec 22, 2013
http://electronotes.netfirms.com/AN401.pdf

[4] "Savitzky-Golay Smoothing", Electronotes Application Note AN-404, Feb 13, 2014
http://electronotes.netfirms.com/AN404.pdf

APPENDIX: Here we show in great detail to use of the inverting impulse response (length-4 MA) to deconvolve the tapered step back into and original signal that was just six ones in a row (green numbers → x).



The portion below, which eventually follows the beginning on the page above, shows how an “inverted echo” results from a truncation (end) of the equalizer impulse response as it moves out.



MATLAB CODE FOR LEAST-SQUARES INVERTER

```
function [h1,h2,h3]=lsi(h1,N)
% function lsi(h1,N)
%   h1 = FIR to invert
%   N = length of inverting FIR
%   h2 = inverting FIR (to calculate)

m=convmtx(h1,N)';
sm=size(m);
r=zeros(1,sm(1));
lr=length(r);
if mod(lr,2)==1;
   r((lr+1)/2) = 1
end
if mod(lr,2)==0;
   r(lr/2)=1
end
% least squares inversion
h2=pinv(m)*r' ;
% test - does this look like r
h3=conv(h2,h1) ;
h2=h2' ;
h3=h3' ;

figure(1)
Lh3=length(h3);
h1=[h1 zeros(1,(Lh3-length(h1)))]
h2=[h2 zeros(1,(Lh3-length(h2)))]
% PLOT h1
subplot(311)
stem([0:Lh3-1],h1)
hold on
plot([-2 100],[0 0],':k')
title('Smoothing Filter  h1')
hold off
if min(h1)==0
   m1=-0.2*max(h1);
   else
   m1=1.2*min(h1)
end
axis([-1 Lh3+1 m1 1.2*max(h1)])
```

```
%
% PLOT h2
subplot(312)
stem([0:Lh3-1],h2)
hold on
plot([-2 100],[0 0],':k')
title('FIR Inverter (Least Sq.) h2')
hold off
axis([-1 Lh3+1 1.3*min(h2) 1.3*max(h2)])

%
% PLOT h3
subplot(313)
stem([0:Lh3-1],h3)
hold on
plot([-2 100],[0 0],':k')
title('Convolved h1 h2')
hold off
axis([-1 Lh3+1 1.5*min(h3) 1.3*max(h3)])
figure(1)

figure(2)
c=exp(-j*2*pi*[0:359]/360);
plot(c,'c')
hold on
plot(roots(h2),'or')
hold off
axis equal
figure(1)
```