The IDA Mark 7 Data Acquisition System Filter Problem

Peter Davis, Jonathan Berger, and David Horwitt

Cecil H. and Ida M. Green Institute of Geophysics and Planetary Physics

Scripps Institution of Oceanography Reference Series 01-10

June 2001
1. Summary

This report describes a problem in the IDA MK 7 data acquisition system (DAS) that affects all recorded and archived data originating from the broadband seismometers. The stations and date of installation of the MK 7 systems are given in Table 1. The affected channels have SEED convention names of $bh^*$, $lh^*$, and $vh^*$. Other channels follow a different processing path and are not affected.

A series of tests conducted on a MK 7 DAS in the laboratory led to the following conclusions:

- There is definitely a bug in the Digital Signal Processor (DSP) code that performs filtering and decimation of the digitized signal stream. The output of the defective code $Y(t) = F(X(t)) + E(t)$, where $F(X(t))$ is the correctly filtered output of the raw digitized stream $X(t)$, and $E(t)$ is an error term.
- The error term arises from an incorrect handling of round-off error in the convolution of the filter coefficients and the data.
- The error term in the output of the first filtering/decimation, the $bh^*$ streams, is approximately equal to $1.6 \times 10^{-3}$ times the lower 16 bits of the 2’s complement representation of the data. Thus, the error is limited in amplitude (104 counts) and produces its largest effect when the input data values cross a $N \times 2^{16}$ boundary where $N$ is an integer.
- The error increases with each subsequent filtering/decimation so the $vh^*$ streams are most affected.
- In the spectral domain, the greatest effect of the error appears in the long-period spectrum (greater than 10 seconds).
- The DSP code is simple to fix.
- Most of the information can be recovered by reprocessing the broadband data (20 sps) and then applying correct filtering/decimation routines to produce the long- and very long-period data channels.

In this report we will give some background information on the discovery of the problem, our diagnosis, and the characterization of the effects of the error on the data recorded and archived.
The IDA MK 7 Filter Problem

2. The Problem

In a MK 7 DAS, the broadband seismic channels, that is the output of the STS-1 or KS54000 seismometers, are digitized by HP 24-bit ADCs at 100 samples per second (sps). These channels (eh*) are not normally recorded. The data are passed to a digital signal processor (DSP) which implements a cascade of digital finite impulse response (FIR) filters to create streams with sample rates of 20 sps, 5 sps, 1 sps, 0.2 sps, and 0.1 sps. The cascades are made up of three basic filters- a decimate by 5, decimate by 4 and decimate by 2. So the recorded and archived data streams come from the following cascades:

<table>
<thead>
<tr>
<th>Stream Name</th>
<th>Data Rate</th>
<th>Cascade</th>
</tr>
</thead>
<tbody>
<tr>
<td>bh*</td>
<td>20 sps</td>
<td>5</td>
</tr>
<tr>
<td>lh*</td>
<td>1 sps</td>
<td>5,4,5</td>
</tr>
<tr>
<td>vh*</td>
<td>0.1 sps</td>
<td>5,4,5,5,2</td>
</tr>
</tbody>
</table>

In December, 2000 the new IDA data acquisition system (The IDA MK 8 Data Acquisition System, Berger et al., SIO Reference Series 00-19) was deployed at the IDA station PFO recording the STS-1 seismometer in parallel with the standard MK 7 system. Shortly after, it was observed (Goran Ekstrom and Meredith Nettles, priv.com.,2001) that the data recorded by the two systems gave significantly different long-period noise (around 100 seconds) for some periods of time and the same for other periods. (See Figure 1.) They further pointed out that there seemed to have been an increase in long-period noise at a number of IRIS/IDA stations beginning at the times that the MK 7 data logger was deployed at those stations.

Ekstrom & Nettles also noted that if the recorded bh* channels (20 sps) were filtered and decimated in the lab, the resulting lh* channels (1 sps) showed lower long-period noise compared with those produced by the MK 7 system. These observations suggested that the problem might be centered on the software internal to the MK 7 system that filters the data. What was very puzzling was that the increased noise at problematic stations was not constant
Figure 1. Comparison of 100 second noise at PFO. Background noise from the $lhz$ channel in the 100 second band recorded during March, 2001, on dual systems operating at Pinyon Flat with input from the same STS1 sensor. The signal power in narrow frequency bands is calculated hourly from the rms amplitude of the $lhz$ channel, and converted to a decibel deviation from Peterson's (1993) new low noise model. The top of the plot is $lhz$ from the MK7 recording GSN PFO data; the middle band, $lhz$ from the MK8 system - XPF; and the narrow, bottom band, the GSN network average for the period. This sample indicates the MK7 recorded data was noisier than MK 8 for part, but not all of the period. Plot provided courtesy of Goran Ekstrom and Meredith Nettles, Harvard University.
The IDA MK 7 Filter Problem

but highly variable in time. Previously Rudolf Widmer-Schnidrig had expressed concerns about some vh* recordings of IRIS/IDA data (Widmer-Schnidrig, priv. com., 1999). At the time, we found no clear pattern to the problematic observations.

4. Problem Diagnosis

After confirming the observations of Ekstrom and Nettles, we re-examined the PFO data for the time period of Figure 1 by reproducing the lh\(z\) channel using double precision floating point (64 bit) routines on the SUN computers at the IDA DCC. When this decimated stream is compared with the lh\(z\) stream produced by the DAS, the source of the noise becomes clearer. Figure 2 shows a plot of the March PFO lh\(z\) data during the same period as in Figure 1, March, 2001. When the stream produced by the MK 7 DAS (lh\(z\)) is compared to a stream produced in the lab on a SUN computer by decimating the higher rate DAS bh\(z\) stream (here ll\(z\)), the two are found to be discrepant by up to approximately 210 counts.

The residual difference between the two time series varies throughout the month. When the bh\(z\) input is uniformly greater than zero at the beginning of the month, the residual is constant and near zero. When the bh\(z\) input oscillates though zero, the residual varies between 0 and –210 counts. This behavior continues from 2001,062-12:00 to 2001,074-16:00 and again from 2001,082-23:00 to the end of the month. Lastly, when bh\(z\) is uniformly less than zero, the residual is nearly constant at –210 counts. When one compares Figures 1 and 2, it is clear the periods of noisy behavior correspond to when the input is oscillating through zero and the DAS is not filtering the data properly. For comparison, Figure 3 shows the same month of data recorded at the IDA station WRAB during which only large earthquakes causes the data to cross one of the 2\(^{16}\) boundaries.

To illustrate the effect more clearly, a MK 7 system was set up in the laboratory with a large, long-period triangle waveform as input. Figure 4 shows the results. The error signals, rsb and rsb, can be seen to decrease as the input data approached zero value for the positive side and then jump to a maximum value as the input data transit 0 values. It is clear from this that the error is proportional to the value of the lower 16-bits of the 2’s complement representation of the input data value.

3. The Simulator.

The DSP, an Analog Devices ADSP-2100-JG, is a 16-bit fixed-point microprocessor which performs the digital filtering and decimation. As the filters require a sum-of-products computation using operands that are greater than 16 bits in magnitude, the DSP implements a sum-of-products calculation using coefficients and data that are both represented in double precision. On the ADSP-2100, this is accomplished through the use of the mixed-mode multiply instructions. (Using the ADSP-2100 Family, Analog Devices, [http://www.analog.com/publications/documentation/Using_ADSP-2100_Vol1/books.html](http://www.analog.com/publications/documentation/Using_ADSP-2100_Vol1/books.html))

The subroutine that realizes the sum-of-products operation used in computing the filters

---

1 Spectra of the lh\(z\) channel were computed for II stations with the MK7 data logger during late 2000 and compared with identical data segments derived on a SUN from the bh\(z\) channel. These plots are available on the IDA web page, [http://quakeinfo.ucsd.edu/idaweb/](http://quakeinfo.ucsd.edu/idaweb/), under What’s New.
Figure 2. Comparison of recorded and derived $lhz$ at PFO during March, 2001. The top trace, labeled $lhz$, is the 1 Hz stream recorded by the MK7 DAS. The middle trace, labeled $llz$, is the 1 Hz stream produced by decimating the DAS $bhz$ stream on a SUN computer using double precision floating point arithmetic. The bottom trace, labeled $res$, is the first difference between trace $lhz$ and trace $llz$. The residual ranges between 0 and $-210$ counts and varies in character throughout the month.
Figure 3. Comparison of recorded and derived \(lhz\) at WRAB during March, 2001. The top trace, labeled \(lhz\), is the 1 Hz stream recorded by the MK7 DAS. The middle trace, labeled \(llz\), is the 1 Hz stream produced by decimating the DAS \(bhz\) stream on a SUN computer using double precision floating point arithmetic. The bottom trace, labeled \(res\), is the first difference between trace \(lhz\) and trace \(llz\). The input stream is uniformly less than zero except during occasional earthquakes. The overall background noise of the station is low, and the residual oscillates about a value of \(-160\) counts with bounds of zero and \(-210\) counts.
Figure 4. Laboratory Tests. A triangular waveform was digitied by a laboratory MK 7. The $bhz$ stream is the 20 sps data as recorded. The $rsb$ stream is the difference between the $bhz$ and the 20 sps data produced by filtering the raw 100 sps data on a SUN computer. The $rsl$ stream is the difference between the $lhz$ and the Sun-derived 1 sps data. The residuals from the 20 Hz channels ($rsb$) are bounded by 0 and –104 counts and jump discontinuously when the input transitions through zero or –2 exp 16.
The IDA MK 7 Filter Problem

works as follows. (See also Section 5.) First, the sum of the products of the low halves of the coefficients and the high halves of the data values is computed; this sum is accumulated with the sum of the products of the high halves of the coefficients and the low halves of the data values. This sum is then shifted right 16 bits and then accumulated with the sum of the products of the high halves of the coefficients and the high halves of the data values. A conditional saturation is then performed on the final 32-bit result before storage to data memory. Note that because the result is only the most significant 32 bits, the products of the low-order coefficients and the low-order data affect only the least significant bit of the result and are therefore not computed. This scheme works as long as the coefficients and a data a properly normalized to the most significant part of the double-precision word.

As a final step in the diagnosis of the problem, the portion of the DAS software that performs filtering and decimation was translated from assembler to C to create a simulator capable of being run on platforms other than the DAS. By working with this simulator, we were able to determine that the problem was caused by a loss of precision resulting from the way the filter coefficients and data were being normalized in the digital signal processor’s (DSP’s) registers. This error was two-fold. First, the original assembler assumed the filter coefficients terms were in the range (+/- 1.0), which they are, but they in fact sum to 1.0 and individually are much smaller than 1.0. Second, the code did not fully exploit the fact that the output of the HP digitizers is only 24 bits while the space available for computation was 32 bits.

This code, named 2100sim.c, is given in Appendix 2 of this report and is available on the IDA WWW page (http://quakeinfo.ucsd.edu/idaweb/, under What’s New). The code has a flag to allow the user to observe how the DAS output can be improved by handling the coefficients and data properly.

We include here three examples, Figures 5 — 7, illustrating how well the simulator imitates the behavior of the MK7 DAS. The basis of these figures is the recording of the 23 June 2001 Arequipa earthquake (Origin Time=20:33:14, lat=16.14˚, long=73.31W, mag=8.1). The PFO DAS was configured temporarily to record the unfiltered 100 Hz data as channel ehz so that a correctly decimated version of bhz could be derived for reference. This data set represents a good test of the software because there are many transitions through zero and 2\(^{16}\).

There are three traces in each figure. The top trace (bhz, lhz, and vhz in Figs 5-7, respectively) shows the channel as recorded by the DAS. The middle trace, labeled rBADx, represents the difference between the top trace and the 2100sim.c code’s simulation of that channel produced by decimating the appropriate higher rate channel (bh from eh, lh from bh, vh from lh). A low average count on this trace confirms the software’s ability to reproduce the aberrant behavior of the DAS. The bottom trace, labeled rOKx, represents the difference between a stream produced from ehz using double precision arithmetic on a SUN workstation and a stream produced by the 2100sim.c software with the ‘repair data’ flag set. A low average count on this trace suggests the effectiveness that a minor change in the DSP software should have on improving data quality.

4. Seismological Effects

Figure 8 shows an example of a small local earthquake(M_L=3.0, Lat=33.95˚, Long=-117.13˚, h=18.8 km., dist=73 km.). The series labeled res is the difference between the
recorded $bhz$ and that derived from the unfiltered $ehz$ channel. This plot illustrates the nature of the time-domain error signal. The large amplitude transients occur when the input values transit 0. The maximum amplitude of the error signal is bounded by 104 counts which in this case represents about 1 % of the input signal.

Figures 9 through 12 illustrate the effects of the MK 7 filter problem on the analysis of the normal modes from a typical large earthquake. In this case the records analyzed (Gabi Laske, priv. com, 2001) were recorded at the IDA station PFO from the Arequipa earthquake of 23 June 2001. Records were obtained both from the MK 7 DAS and from the MK 8 DAS recording in parallel.

Figure 13 and 14 illustrate the effects in the spectral domain for a period of normal background noise. The time period analyzed is shown on the top panel of Figure 9. It is a period of normal background noise where the input data values are varying about 0-value.

To try to give some idea of the effect the MK 7 filter problem has on the recorded noise levels at all affected IRIS/IDA stations, we performed an analysis of the network data for the last half of the year 2000. For this time period we derived an $lhz$ channel in the lab from the $bhz$ channel recorded by the DAS (the data that is distributed to the community) and compared that with the DAS-recorded $lhz$. The results are given in Appendix 2. From these plots it is apparent that not all stations exhibit excessive noise in the long-period band for this period. Of course for a given station-epoch the excess long-period noise is simply a function of where the average values the data streams lie with respect to the critical $2^{16}$ boundaries.
The IDA MK 7 Filter Problem

Figure 5. Software Simulator. The top trace shows the bhz channel as recorded by the DAS. The middle trace, labeled rBADb, represents the difference between the DAS’s version of bhz and the 2100sim.c code’s simulation of that channel produced by decimating the ehz channel. The bottom trace, labeled rOKb, represents the difference between a stream produced from ehz using double precision arithmetic on a SUN workstation and a stream produced by the 2100sim.c software with the ‘repair data’ flag set. The lower the amplitude of the middle trace, the better the software simulates the DAS’s precision loss. Likewise, the lower the amplitude of the bottom trace, the more effective the correction to the software will be.
Figure 6. Software Simulator. As in Figure 5 but for lhz channel.

Filter: None, Amp: Auto

BRTT dbpick: USE plot_lh.ps pdavis Tue Jun 26 14:05:34 2001
Figure 7. Software Simulator. As in Figure 5 but for vhz channel.
Figure 8. Small Earthquake. The top trace shows the $bhz$ channel as recorded by the MK 7. The bottom trace represents the difference between the MK 7 $bhz$ recording and the 20 Hz stream produced by decimating the raw 100 Hz stream on a SUN using double precision floating point arithmetic.
The IDA MK 7 Filter Problem

Figure 9. Arequipa Earthquake. Fourier amplitude spectra in the normal mode band for the 70 hour period beginning 33 minutes before the origin time of the earthquake. The spectrum labeled $vhz$ comes from the 0.1 sps channel as recorded by the MK 7 DAS. The spectrum labeled $vbz$ is a 0.1 sps channel derived in the lab from the recorded $bhz$. The spectrum labeled $vlz$ is a 0.1 sps channel derived in the lab from the unfiltered $ehz$. The bottom spectrum labeled $vhz$ comes from the 0.1 sps channel as recorded by the MK 8 DAS.
**Figure 10. Arequipa Earthquake.** Fourier amplitude spectra in the low end of the normal mode band for the 70 hour period beginning 33 minutes before the origin time of the earthquake. The spectrum labeled $vhz$ comes from the 0.1 sps channel as recorded by the MK 7 DAS. The spectrum labeled $vbz$ is a 0.1 sps channel derived in the lab from the recorded $bhz$. The spectrum labeled $vlz$ is a 0.1 sps channel derived in the lab from the unfiltered $ehz$. The bottom spectrum labeled $vhz$ comes from the 0.1 sps channel as recorded by the MK 8 DAS.
Figure 11. Arequipa Earthquake. Fourier amplitude spectra in the normal mode band for the 25 hour period beginning 33 minutes before the origin time of the earthquake. The spectrum labeled $v_{hz}$ comes from the 0.1 sps channel as recorded by the MK 7 DAS. The spectrum labeled $v_{bz}$ is a 0.1 sps channel derived in the lab from the recorded $b_{hz}$. The spectrum labeled $v_{lz}$ is a 0.1 sps channel derived in the lab from the unfiltered $e_{hz}$. The bottom spectrum labeled $v_{hz}$ comes from the 0.1 sps channel as recorded by the MK 8 DAS.
**Figure 12. Arequipa Earthquake.** Fourier amplitude spectra in the normal mode band for the 50 hour period beginning 13 hours and 27 minutes after the origin time of the earthquake. The spectrum labeled $vhz$ comes from the 0.1 sps channel as recorded by the MK 7 DAS. The spectrum labeled $vbz$ is a 0.1 sps channel derived in the lab from the recorded $bhz$. The spectrum labeled $vlz$ is a 0.1 sps channel derived in the lab from the unfiltered $ehz$. The bottom spectrum labeled $vhz$ comes from the 0.1 sps channel as recorded by the MK 8 DAS.
The IDA MK 7 Filter Problem

Figure 13. Long-Period Spectral Effects. Data from PFO for period 2001:173-02:00 through 04:00. During this period the data values were varying around 0. The red spectrum is the $lhz$ channel as recorded. The green spectrum is the 1 sps data SUN-derived from the $bhz$ channel. The blue spectrum is 1 sps data SUN-derived from the unfiltered 100 sps data.
In order to better understand this problem’s effect on the entire recording range of the MK7 DAS, a numerical experiment was performed using the commercial computational package MATLAB(R). The steps in this experiment were as follows:

1. A time series of length $2^{20}$ points was computed using the built-in function GWN. GWN produces a time series containing Gaussian white noise of a specified power. This series was used to represent the unfiltered output of the H-P digitizer board.

2. The DSP simulation software 2100sim (described above) was applied to the series in (1) with the ‘b’ flag set in order to mimic the behavior of the DAS with the bug still in the DSP software. This produced a time series equivalent to a BH (20 Hz) stream with the defect still present.

3. The simulation software was applied to the series in (1), this time with the ‘r’ flag set, to produce a second BH stream, this one without defect.

4. The series in (2) and (3) were fed to the function SPECTRUM, which computed the transfer function between the two using Welch's method. The output of this step was a matrix containing information on input signal strength, relative transfer function, and coherence for a single series of fixed power.
5. Steps 1-4 were repeated for varying power settings of the function GWN. In this way an ensemble of transfer functions was computed for a wide range of input signal amplitudes.

6. The signal power was renormalized to units appropriate for comparison with a physical system by applying the system response for the MK7 DAS recording an STS1 seismometer. Each measurement was also renormalized to a bandwidth identical to Figure 1.2 of the document "The Design Goals for a New Global Seismographic Network" (IRIS, 1985) that has circulated among the IRIS community for some time.

7. The ensemble from (6) was then divided into bins of width 0.2 log(frequency) and height 0.5 log(ground acceleration). Within each bin, the variation of transfer function amplitude and phase was computed. It is this measure of variation that is contoured in Figures 15 and 16.

The background of these figures is rendered similar to that of Figure 1.2 of the IRIS document for ease of comparison. Estimates were made at the time of the amplitudes of waves one might expect to record at a canonical distance of 30 degrees from an earthquake epicenter. These amplitude ranges for various earthquake magnitudes are shown as solid line segments. The dashed lines labeled NHNM and NLNM are the noise models of Peterson.

As these contour plots indicate, the greatest discrepancy between a series actually produced by the MK7 DSP filters and a series properly filtered is at very low amplitudes just above the Peterson New Low Noise Model at long periods (10 - 1000 seconds) and at small amplitudes above 5 Hz. The latter will not come into play very much for quakes at distances 30 degrees and above because of the low pass properties of the Earth but may have some effect on small local and regional events nearby. As is shown below, a data repair algorithm can reduce the effects of the software error even above 5 Hz.
Figure 15. Amplitude Variations. The amplitude variations of an ensemble of transfer functions computed for a wide range of input signal amplitudes. See text for explanation.
Figure 16. Phase Variations. The phase variations of an ensemble of transfer functions computed for a wide range of input signal amplitudes. See text for explanation.
5. A Reprocessing Scheme

From the analysis in the previous sections of this report it is clear that all $bh^*$, $lh^*$, and $vh^*$ channels are contaminated. The $lh^*$ and $vh^*$ channels can be generated in the lab from the corresponding $bh^*$ channels. The $bh^*$ channels, however cannot be re-generated from the $eh^*$ channels as the latter were not recorded. The challenge then is to re-process the $bh^*$ channels to remove an estimate of the error.

The subroutine that realizes the sum-of-products operation used in computing the filters works as follows. Let $d$ represent the unfiltered (and unrecorded) data. Then the output of the filtering/decimation routine can be written as:

First, the sum of the products of the low halves of the coefficients and the high halves of

$$X(n) = \sum_{k=n-i}^{n+i} d(k) \cdot c(k)$$

The data and coefficients are split into two parts, the high 16-bits part and the low 16-bit part.

$$d(k) = d_h + d_i; \quad c(k) = c_h + c_i$$

the data values is computed; this sum is accumulated with the sum of the products of the high halves of the coefficients and the low halves of the data values. This sum is then shifted right 16 bits and accumulated with the sum of the products of the high halves of the coefficients and the high halves of the data values. Note that because the result is only the most significant 32 bits, the products of the low-order coefficients and the low-order data, the last term in the equation below, affect only the least significant bit of the result and are therefore not computed. This scheme works as long as the coefficients and the data a properly normalized to the most significant part of the double-precision word.

The crux of the defect in the original coding of the DSP filtering algorithm was in the normalization of the data and the coefficients. This error was two-fold. First, the original assembler assumed the filter coefficients terms were in the range (+/- 1.0), which they are, but they in fact sum to 1.0 and individually are much smaller than 1.0. Second, the code did not fully exploit the fact that the output of the HP digitizers is only 24 bits while the space available for computation was 32 bits.

The algorithm does not calculate the last term in the equation above so the output of the filter/decimation routine can be written as the sum of the recorded signal plus an error term.

$$X(n) = X_p(n) + X_e(n)$$
where the error term (that is the part not calculated in the filter/decimation routine) is given by:

\[ X_E(n) = \sum_{k-n}^{n+i} d_i(k) \ast c_i(k) \]

We have devised a method to estimate this error term by the following simple technique. The MATLAB instructions are given below:

```matlab
% b are the recorded bh* data (20 sps)
ib=interp(b,5) +2^23+52;
remib=rem(ib,65536);
error=resample(remib,20,100,cl);
estb=b+error;
% estb are the “repaired bh* data.
```

First the \( bh^* \) data are interpolated to 5 times the recorded sample rate to estimate the \( eh^* \) data. The first constant, \( 2^{23} \), is added to the interpolated series to insure the series has only positive values. A second constant, \( 52 \), is added for the following reason.

The faulty MK 7 filtering/decimation routine results in an offset between the recorded \( bh^* \) data and the unfiltered \( eh^* \) data. This is illustrated in the Figure 17 where one can clearly see the effect when \( eh_z \) transits 0-value.

From the equation above for \( X_E \), we can see that if the input to the filter/decimation routine were a constant 0-value, the output would be 0. But if the input were a constant value of –1, the output would be in error by \( 2^{16} \) (the low 16-bits of –1 in 2s-complement representation) times the sum of the filter coefficients low 16-bit parts = 104. So an equal offset must be added to the interpolated series before forming the low part of the word.

Finally the estimated error signal is formed by the convolution and decimation of the low part of the interpolated offset data and the low part of the filter coefficients.
Figure 17. The top panel shows a section of unfiltered $ehz$ (100 sps) data. The bottom panel show the difference between the recorded $bhz$ and Sun-derived version. When $ehz$ crosses 0-value, $bhz$ has a value of -52 which is $1/2 \times 2^{16}$ times the sum of the low part of the filter coefficients.

To illustrate the efficacy of this technique we apply it to several examples of seismic signals and noise from PFO with the MK 7 DAS configured to record unfiltered $ehz$ as well as the normal complement of streams.

In the first example we applied the technique to a section of data that includes the seismogram of the recent Arequipa earthquake. Figure 18 shows the results. Note that the rms of the original error signal was 23.3 counts compared with the rms of 0.8 counts in the error signal remaining after applying this technique.

In the second example we apply the technique to a section of background noise during a period when the data are averaging about 0-value. Figure 19 top half shows the time series of $bhz$ with the spectra in the lower half. This figure may be compared with Figure 13.

In the third example (Figure 20) we apply the technique to the small earthquake previously discussed and shown in Figure 8. In this case the technique does not perform very well, reducing the residual only from 11.6 to 4.8 counts. The reason can be seen by examination Figure 18. The top three panels show the seismogram, the residual of the recorded $bhz$
Figure 18. The Arequipa earthquake seismogram recorded at PFO. Start time is 2001:175-20:43:40 for 2000 seconds. The top panel is the $bhz$ data as recorded. The middle panel is the difference between the SUN-derived $bhz$ and the recorded $bhz$. The bottom panel is the difference between the SUN-derived $bhz$ and the estimated $bhz$ using the technique described herein.

and error estimate. The bottom three panels show the expanded first arrival section. The $ehz$ trace is the unfiltered data, interpolated $bhz$ channel - our approximation of $ehz$, and the lower trace the error series. Clearly the high frequencies in the original signal (>6 Hz) that cause the values to transit the critical 0-value are absent in our approximation. They have been filtered out. It may be that a more sophisticated interpolation or de-convolution scheme might improve the performance but that has not been attempted.
Figure 19. Ambient noise at PFO. Start time is 2001:173-02:00:08 for 2 hours. The red spectrum is the $bhz$ as recorded. The blue spectrum is the SUN-derived $bhz$ and the green spectrum is the spectrum of the estimated $bhz$ using the technique described herein. See also Figure 13.
The IDA MK 7 Filter Problem

Figure 20. The small earthquake discussed in section 4 (see Figure 8). The top three panels show the seismogram, the residual of the recorded $bhz$ and error estimate. The bottom three panels show the expanded first arrival section. The $ehz$ trace is the unfiltered data, interpolated $bhz$ channel - our approximation of $ehz$, and the lower trace the error series.
APPENDIX 1. 2100SIM.C

/*
* simulate double-precision convolving in ADSP2100
* Input is on stdin, output is on stdout
* -r flag indicates apply correction to arithmetic
* -b flag appropriate to convert 100 Hz to 20 Hz
* -l flag appropriate to convert 20 Hz to 1 Hz, step 1
* -k flag appropriate to convert 20 Hz to 1 Hz, step 2
* -v flag appropriate to convert 1 Hz to 0.1 Hz, step 1
* -w flag appropriate to convert 1 Hz to 0.1 Hz, step 2
* Examples:
* 2100sim -b < ehz.w > bhz.w
* 2100sim -l < bhz.w | 2100sim -k > lhz.w
* 2100sim -v < lhz.w | 2100sim -w > vhz.w
*
* original by David Horwitt, 6/2001
* modified by Pete Davis, 6/2001
*/

#include <stdio.h>
#include <stdlib.h>


#define N 99

double coeffs1[] = {
-3.3760433455320e-07 , -8.62870365783046e-07,  
-1.54925510359428e-06 , -2.04006210064518e-06,  
-1.60327181294713e-06 , 8.76374543121259e-07,  
-6.73020258859275e-06 , 1.70897692521521e-05,  
-3.2228883708637e-05 , 5.07337972758030e-05,  
6.87041320226640e-05 , 7.92876817655227e-05,  
7.29081221264359e-05 , 3.85255553007711e-05,  
-3.38987447479268e-05 , -1.49823725293308e-04,  
-3.05914785855410e-04 , -4.86083794611545e-04,
};
The IDA MK 7 Filter Problem

-6.5898755959722e-04, -7.78326765065234e-04,
-7.87101686367347e-04, -6.26427121752141e-04,
-2.48596537979401e-04, 3.67087312213652e-04,
1.19835604038013e-03, 2.16751266371809e-03,
3.13647324365399e-03, 3.91288800347265e-03,
1.26971307223184e-03, 3.97828826866033e-03,
2.85238865895774e-03, 7.97673128916730e-04,
-2.14136298845586e-03, -5.74628869339185e-03,
-9.61061101854342e-03, -1.31538249613502e-02,
-1.56670324577331e-02, -1.63886975573323e-02,
-1.46032478728347e-02, -9.75057296908953e-03,
-1.53045915138463e-03, 1.00149447145010e-02,
2.44562160337606e-02, 4.09803949487304e-02,
5.84476702187432e-02, 7.54974186772003e-02,
9.06920433466751e-02, 1.02680221247804e-01,
1.10357262245546e-01, 1.13000042789150e-01,
1.10357262245546e-01, 1.02680221247804e-01,
9.06920433466751e-02, 7.54974186772003e-02,
5.84476702187432e-02, 4.09803949487304e-02,
2.44562160337606e-02, 1.00149447145010e-02,
-1.53045915138463e-03, -9.75057296908953e-03,
-1.46032478728347e-02, -1.63886975573323e-02,
-1.56670324577331e-02, -1.31538249613502e-02,
-9.61061101854342e-03, -5.74628869339185e-03,
-2.14136298845586e-03, 7.97673128916730e-04,
2.85238865895774e-03, 3.97828826866033e-03,
2.85238865895774e-03, 3.91288800347265e-03,
1.19835604038013e-03, 3.67087312213652e-04,
-2.48596537979401e-04, -6.26427121752141e-04,
-7.87101686367347e-04, -7.78326765065234e-04,
-6.58987559597222e-04, -4.86083794611545e-04,
-3.05914785855410e-04, -1.49823725293308e-04,
-3.38987447479268e-05, 3.85255553007711e-05,
7.29081221264359e-05, 7.92876817655227e-05,
6.87041320226640e-05, 5.07337972758030e-05,
3.22288833708637e-05, 1.70897692521521e-05,
6.7302025889275e-06, 8.763745431259e-07,
-1.60327181294713e-06, -2.04006210064518e-06,
The IDA MK 7 Filter Problem

-1.54925510359428e-06, -8.62870365783046e-07, -3.37604433455320e-07
};

/*
 * coefficients for decimation by 4
 */

double coeffs2[] = {
-1.11358802210833e-08, -3.78394868622766e-07, -1.34125322382762e-06, -2.90200053199354e-06,
-4.41871855638605e-06, -4.33851392376124e-06, -3.7857364266237e-07, 9.55231844926978e-06,
2.56732422422379e-05, 4.43664095302544e-05, 5.6949909328311e-05, 5.05028851990804e-05,
1.17698201993374e-05, -6.58971724463716e-05, -1.74288753959477e-04, -2.84269417837086e-04,
-3.4629902261837e-04, -3.0113748903074e-04, 7.29381834945973e-04, 1.17239583853185e-03,
1.40248505797650e-03, 1.2449828710360e-03, 5.10478430542183e-04, -7.20216929992503e-04,
-2.24521847603683e-03, -3.63940045096584e-03, -4.35222414109690e-03, -3.8609124261652e-03,
-1.86972140267345e-03, 1.50361659319673e-03, 5.62254939768756e-03, 9.37321090537620e-03,
1.13814796963813e-02, 1.03845481436407e-02, 5.67801426718469e-03, -2.48443686096107e-03,
-1.26751452741356e-02, -2.23683881505476e-02, -2.83481967797188e-02, -2.73955346771786e-02,
-1.71113746411353e-02, 3.32994457023328e-03, 3.27105769849895e-02, 6.77429988093373e-02,
1.03543552498421e-01, 1.34531042601875e-01, 1.55557099885596e-01, 1.62999851141968e-01,
1.55557099885596e-01, 1.34531042601875e-01, 1.03543552498421e-01, 6.77429988093373e-02,
3.27105769849895e-02, 3.32994457023328e-03, -1.71113746411353e-02, -2.73955346771786e-02,
-2.83481967797188e-02, -2.23683881505476e-02, -1.26751452741356e-02, -2.48443686096107e-03,
The IDA MK 7 Filter Problem

5.67801426718469e-03, 1.03845481436407e-02,
1.13814796963813e-02, 9.37321090537620e-03,
5.62254939768756e-03, 1.50361659319673e-03,
-1.86972140267345e-03, -3.86099124261652e-03,
-4.3522414109690e-03, -3.63940045096854e-03,
-2.4521847603683e-03, -7.20216929992503e-04,
5.10478430542183e-04, 1.22449828710360e-03,
1.40248505797650e-03, 1.17239583853185e-03,
7.29381834945973e-04, 2.6116672950156e-04,
-1.01270331271485e-04, -3.01137489083074e-04,
-3.46299902261837e-04, -2.84269417837086e-04,
-1.74288753959477e-04, -6.58971724463716e-05,
1.17698201993374e-05, 5.05028851990804e-05,
5.69499093928311e-05, 4.43664095302544e-05,
2.56732422422379e-05, 9.55231844926978e-06,
-3.78577364266237e-07, -4.33851392376124e-06,
-4.41871855638605e-06, -2.90200053199354e-06,
-1.34125322382762e-06, -3.78394868622766e-07,
-1.1135802210833e-08
};

/*
 * coefficients for decimation by 2
 */

double coeffs3[] = {
-4.37277543655446e-07, -4.82801027074436e-06,
-2.65789291225044e-05, -9.47667419127147e-05,
-2.36451055257277e-04, -4.0538349303201e-04,
-3.72431439911244e-04, 2.93047105382995e-04,
1.9367829415069e-03, 4.16007818890067e-03,
5.1900765175525e-03, 2.17544214795206e-03,
-6.85295453791835e-03, -1.98534426519137e-02,
-2.86946125362195e-02, -2.09204607745684e-02,
1.37748558164710e-02, 7.5302808508624e-02,
1.48903546801816e-01, 2.09347513604984e-01,
2.32756403074331e-01, 2.09347513604984e-01,
1.48903546801816e-01, 7.5302808508624e-02,
1.37748558164710e-02, -2.09204607745684e-02,
-2.86946125362195e-02, -1.98534426519137e-02,

32 SIO Reference Series 01-10
}

union ls {
    long l;
    unsigned short s[2];
};

#endif

/*****************************************************************
* tofixed(): convert floating point coefficients to fixed point
* (1:31 format).
*****************************************************************/

static void tofixed(double in[], unsigned short out[], int n) {
    int i;
    double f;
    union ls u;

    for (i=0; i < n; i++) {
        f = in[i] * 0x7fffffff;
        u.l = (long)f;
        out[(i*2)+LSW] = u.s[LSW];
        out[(i*2)+MSW] = u.s[MSW];
    }
/* tofixed() */
/*****************************************************************
* dp_convolve(): perform double precision convolve as DSP2100 does
* it.
* Return value is filtered value
*******************************************************************/

long dp_convolve(unsigned short coeffs[], unsigned short delay[],
                 int index, int n, double renorm) {
    int i, ix;
    long y, mr;
    /*
    * this is used to simulate a 40 bit MAC accumulator
    */
    double yf;

    yf = 0.0;
    /*
    * no point performing coefficient low word * data high word
    * since we don’t have the resolution to save it
    */

    /*
    * coefficient low word * data high word
    */
    ix = index;
    for (i = 0; i < n; i++) {
        mr = (coeffs[(i*2)+LSW] * ((short)delay[(ix*2)+MSW])); /* signed */
        /*
        * the MR register is shifted left 1 bit before
        * summing (ADSP-2100 multiplier feature when in
        * fractional mode)
        */
        mr <<= 1;
        yf += mr;
        if (++ix >= n)
            ix = 0;
    };
/* coefficient high word * data low word */
ix = index;
for (i=0; i < n; i++) {
    mr = (((short)coeffs[(i*2)+MSW]) * /* signed */
          delay[(ix*2)+LSW]);
    /*
     * MR shift
     */
    mr <<= 1;
    yf += mr;
    if (++ix >= n)
        ix = 0;
};

/*
 * shift out lowest 16 bits which are unresolvable in
 * our output
 */
yf /= 65536;

/*
 * coefficient high word * data high word */
ix = index;
for (i=0; i < n; i++) {
    mr = (((short)coeffs[(i*2)+MSW]) * /* signed */
          ((short)delay[(ix*2)+MSW])); /* signed */
    /*
     * MR shift before summing
     */
    mr <<= 1;
    yf += mr;
    if (++ix >= n)
        ix = 0;
};

yf /= renorm;
The IDA MK 7 Filter Problem

```c
y = (long)yf;
return(y);

} /* dp_convolve() */

/**************************************************************/
*/ getval(): input next value from stdin. 
*/ Return value is >0 if value read, else EOF 
 **************************************************************/ static int getval(long *val)
{
    int n;

    n = fread(val,sizeof(long),1,stdin);
    return((n <= 0) ? EOF : n);

} /* getval() */

int main(int argc, char *argv[])
{
    int i,n,decimate,iin,iout;
    int nfilt, ndecimation, norm = 1;
    int fixflag = 0;
    int datanorm = 64;
    int ndelay;
    long y;
    double local_coef[N];
    double renorm;
    union ls u;
    unsigned short cfixed[N*2];
    static unsigned short delay[N*2] = { 0 };

    while ((n = getopt(argc,argv,“blkrvw”)) != -1) {
        switch (n) {
            case ‘b’:
                ndecimation = 5;
                ndelay = 2;
                nfilt = 99;
                for (i=0; i<nfilt; i++) local_coef[i] = coeffs1[i];
```
```
```c
break;
case 'l':
    ndecimation = 4;
    ndelay = 4;
    nfilt = 99;
    for (i=0; i<nfilt; i++) local_coef[i] = coeffs2[i];
    break;
case 'k':
    ndecimation = 5;
    ndelay = 3;
    nfilt = 99;
    for (i=0; i<nfilt; i++) local_coef[i] = coeffs1[i];
    break;
case 'r':
    fixflag = 1;
    break;
case 'v':
    ndecimation = 5;
    ndelay = 2;
    nfilt = 99;
    for (i=0; i<nfilt; i++) local_coef[i] = coeffs1[i];
    break;
case 'w':
    ndecimation = 2;
    ndelay = 0;
    nfilt = 41;
    for (i=0; i<nfilt; i++) local_coef[i] = coeffs3[i];
    break;
default: fprintf(stderr,usage);
    exit(1);
};

/*
 * if fixing precision problem, renorm the coefs:
 */
if ( ndecimation == 2 && fixflag ) norm = 1;
if ( ndecimation == 4 && fixflag ) norm = 2;
if ( ndecimation == 5 && fixflag ) norm = 4;
```
for (i=0; i<nfilt; i++) local_coef[i] *= (double)norm;

/*
 * convert double precision coefficients to
 * 1:31 fixed point
 */
tofixed(local_coef, cfixed, nfilt);

/*
 * this simulator does not treat timing. delays arbitrarily compensate.
 */
for (i=0; i<ndelay; i++) getval(&u.l);

/*
 * read in data and pass through convolver
 */
iin = iout = decimate = n = 0;
renorm = (fixflag) ? (double)norm * (double)datanorm : (double)1.0;
while (getval(&u.l) != EOF) {
    if ( fixflag ) u.l *= datanorm;
    delay[(iin*2)+LSW] = u.s[LSW];
    delay[(iin*2)+MSW] = u.s[MSW];
    iin++;
    /*
     * ensure enough data before commencing
     */
    if (++n < nfilt )
        continue;

    /*
     * generate filtered value
     */
    y = dp_convolve(cfixed,delay,iout,nfilt,renorm);

    /*
     * fix indexing for circular buffers
     */
    if (++iout == nfilt)
        iout = 0;
if (iin >= nfilt)
    iin = 0;

/*
 * simple decimator
 */
if (++decimate == ndecimation) {
    (void)fwrite(&y,sizeof(long),1,stdout);
    decimate = 0;
}
exit(0);
} /* main() */

/*** 2100sim.c ***/

/*** 2100sim.c ***/
APPENDIX 2. IDA Station Noise

To try to give some idea of the effect the MK 7 filter problem has on the recorded noise levels at the IRIS/IDA stations we performed an analysis of the network data for the last half of the year 2000. For this time period we derived the $lh_z$ channel from the $bh_z$ channel in the lab and compare that with the recorded $lh_z$.

Following the procedures of Astiz used in the FDSN Station Book (http://www.fdsn.org/FDSNstation.htm), the seismic noise levels determined from the robust PSE analysis are shown in decibels (db) with respect to acceleration ($m^{**2}/s**4)/Hz$. In these plots, the horizontal axes indicates the period in seconds. The blue curve shows the spectra of the recorded $lh_z$ channel and the red curve shows the spectra of the derived $lh_z$ channel.

The station codes are given in the upper-left corner. The lower and upper dash curves are the low (LNM) and high noise models (HNM) of Peterson (1993). These curves are shown for reference.
The IDA MK 7 Filter Problem

Seismic Noise Level

AAK II

ABKT II

Period (sec)

PSD relative to 1 (m/s²)**2/Hz (dB)

Seismic Noise Level

SUN derived

MK7

Period (sec)

PSD relative to 1 (m/s²)**2/Hz (dB)
The IDA MK 7 Filter Problem

Seismic Noise Level

ARU II

PSD relative to 1 (m/s^2)^2/Hz (dB)

Period (sec)

Seismic Noise Level

ASCN II

PSD relative to 1 (m/s^2)^2/Hz (dB)

Period (sec)
Seismic Noise Level

BFO II

Seismic Noise Level

BORG II

PSD relative to 1 (m/s^2)^2/Hz (dB)

Period (sec)
The IDA MK 7 Filter Problem

Seismic Noise Level

BRVK II

PSD relative to 1 (m/s^2)**2/Hz (dB)

Seismic Noise Level

CMLA II

CMLA II

PSD relative to 1 (m/s^2)**2/Hz (dB)
Seismic Noise Level

COCO II

PSD relative to 1 (m/s²)²/Hz (dB)

Period (sec)

Seismic Noise Level

EFI II

PSD relative to 1 (m/s²)²/Hz (dB)

Period (sec)
The IDA MK 7 Filter Problem

Seismic Noise Level

ERM II

Seismic Noise Level

ESK II

PSD relative to 1 (m/s^2)**2/Hz (dB)
The IDA MK 7 Filter Problem

Seismic Noise Level

FFC II

Seismic Noise Level

HOPE II

PSD relative to 1 (m/s^2)^2/Hz (dB)

Period (sec)
Seismic Noise Level

KIV II

Seismic Noise Level

KURK II

PSD relative to 1 (m/s²)**2/Hz (dB)

Period (sec)
The IDA MK 7 Filter Problem

Seismic Noise Level

KWAJ II

LVZ II

Seismic Noise Level

PSD relative to 1 (m/s^2)^2/Hz (dB)

Period (sec)

SUN derived

MK7

LVZ II

SUN derived

MK7
The IDA MK 7 Filter Problem

Seismic Noise Level

MBAR II

Seismic Noise Level

MSEY II

PSD relative to 1 (m/s^2)^2/Hz (dB)

Period (sec)

SUN derived

MK7
The IDA MK 7 Filter Problem

Seismic Noise Level

NNA II

Seismic Noise Level

OBN II

Period (sec)

PSD relative to 1 (m/s²)**2/Hz (dB)

SUN derived

MK7
The IDA MK 7 Filter Problem

Seismic Noise Level

PALK II

Seismic Noise Level

PFO II
The IDA MK 7 Filter Problem

Seismic Noise Level

SUR II

TLY II

PSD relative to 1 (m/s^2)^2/Hz (dB)

Period (sec)
The IDA MK 7 Filter Problem

Seismic Noise Level

WRAB II

SUN derived
MK7

PSD relative to 1 (m/s^2)^2/Hz (dB)

Period (sec)

Seismic Noise Level

SACV II

SUN derived
MK7

PSD relative to 1 (m/s^2)^2/Hz (dB)

Period (sec)