Baseline asymmetry, Tau projection, B-field estimation and automatic half-cycle rejections

Technical report THEM2001-02 (version 1.00)
(Follows preliminary reports 2001-02A and 2001-02B)
April-May 2001

Daniel Lemire, Ph.D.
http://www.ondelette.com
lemire@ondelette.com
(514) 524-3517
Report for **THEM Geophysics Inc.**

Keywords: Tau domain, splines, wavelets, baseline

Please forward comments and criticism to Daniel Lemire at lemire@ondelette.com.

Introduction

This report assumes that the reader is familiar with **THEM Geophysics Inc**. THEM system is an innovative EM system appropriate for rough terrain mineral exploration. It was developed over the last 4-5 years and is still considered a prototype.

One of the unresolved issue has been the noise level. It is only one of several issues of course, but signal-to-noise ration seems to be a convenient way to measure the quality of an EM system and is of significant commercial interest. Unfortunately, there isn't a standardized measure of the noise level because processing software and hardware vary greatly from one system to another.

At this point in time, it seems that processing the signal itself (and not subproducts such as channels) might be a priority because there are qualified consultants able to prepare the various subproducts in a satisfying manner for the industry. Two problems have been identified over time : is baseline correction is sufficient and how to remove the high frequency noise in the signal?

Data under consideration

We will study the file bob3.dat provided by THEM Geophysics Inc. The file is 217 Mbytes in size and dated from April 21st 1999. X and Z coils are available but only Z coil will be considered here. This analysis might not apply to all collected data by THEM EM system and the reader should be aware that there might be components in the signal which are specific to this particular file. It is however believed to be a representative signal set by **THEM Geophysics Inc.** It is assumed that the emitter was set at 30 Hz so that we have 60 half-cycles (or *frames*) per second and 512 samples per half-cycles (for a sampling rate of 30kHz) over both the X and Z coils. For this data, the scaling factor for conversion in PPM that 1 million divided by 8 times the standard deviation which is around 30 which means that the smallest non-zero value measured by the system is 30 PPM. Notice that the amplitude of the signal is estimated using the standard deviation and through a simple max - min formula which is more sensitive to changes in the morphology of the signal and noise during the on-time.

Definition of the problem

Current EM processing software extract all of their information from the off-time of the signal, that is, the delay during which the emitter is turned off and the signal is decaying (more or less according to an exponential curve). It is also where the apparent signal-to-noise ratio is lower (it can be as low as 1:25).

Baseline correction is achieved through the SIMn algorithm using $n = 3$ and a late window of 48 samples (see [2]). Whenever stacking is not explicitly specified we stacked each two subsequent half-cycles (simply averaging one with the other after inverting the signs). This type of data will be referred to as "raw" since no denoising was used on it.

Our goal is to increase substantially the signal-to-noise ratio and make the data more accurate. The current implementation of THEM software achieve noise reduction through stacking[1] over 6 frames or more and using an experimental wavelet denoising scheme referred to as *DeSpike* (to remove noise due to sudden and brief atmospheric discharges). This wavelet denoising scheme won't be discussed here even though it is important to handle atmospheric discharges in any future processing software.

Summary

1 Stacking means that we average the responses over *n* half-cycles (switching signs) and then use this averaged response instead of the x half-cycles (in effect, downsampling the signal by a factor of *n*). We can also replace the average with the median.

Stacking half-cycles simply doesn't achieve the noise reduction we need. It is quite possible that the signal is polluted by hardware deficiencies such as bad shielding.

A new software is completed. It has quite a number of new features including Tau projection, experimental B-field estimation and automatic half-cycle rejections. It is worth noting however that the author was unable improve in any significant manner the baseline correction (see history below). It is believed that half-cycle asymmetries and other defects are making a more accurate baseline estimation difficult. Further research is needed.

Software summary

The library can now read directly into a THEM DAT file. A very simple yet efficient high-level API was developped (classes THEMFile and THEMFilter) to read and process THEM DAT Files. See software documentation for further information (file mhelp.chm).

Baseline correction (SIMn)

We briefly recall our baseline correction algorithm. For each half-cyle, we solve for a polynomial

$$p(x) = \sum_{j=0}^{n} a_j x^j$$

while we note the signal samples

$$\{s_k\}_{k=0}^{k_{\max}}$$

while y_0 defines the continuity condition ($p(0) = y_0$) for the baseline (relative to the previous half-cycle). We choose a value k1 between 0 and kmax (usually $k_1 = k_{\max} - 48$ when $k_{\max} = 511$) which defines the *late half-cycle width*. We then solve the Lagrange problem defined by

$$L(a_0,\dots,a_n,\lambda_0,\lambda_1) = \sum_{k=k_1}^{k_{\max}} (p(k) - s_k)^2 - \lambda_0 l_0 - \lambda_1 l_1$$

where

$$l_0(a_0,\dots,a_n) = p(0) - y_0$$

and

$$l_1(a_0,\dots,a_n) = \sum_{k=0}^{k_{\max}} p(k) - \sum_{k=0}^{k_{\max}} s_k .$$

This polynomial then defines the baseline and we can substract it from the signal to get the corrected signal. We can then use the value of the polynomial at k_{ma} as y_0 for the next half-cycle and so on. Only the first y_0 must be guessed.

Are pulses symmetrical? Is the baseline flawed?

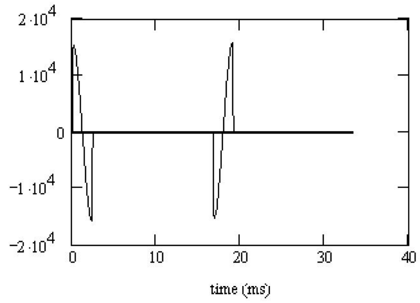
It will be verified that odd and even pulses are asymmetrical in the processed data. Amplitude and morphology are slightly different. In what follows, no stacking was used (baseline correction was applied : SIMn algorithm with parameters $n = 3$ and $k_2 = 48$).

While this analysis was done only on a sample of data and the exact nature of the asymmetry may vary from flight to fight, we can still conclude that unlikely what was previously assumed (by the SIMn algorithm in particular) even and odd half-cycles are not symmetrical.

A. Morphological comparison

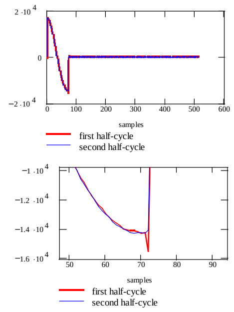
The morphology of a full cycle should look like what is presented in Fig. 1. While the scale may vary, the first half of a cycle should match the second half with signs reversed. We will note the first half of the frame F_1 and the second half F_2 with the understanding that $F_1(i)$ refers to the ith sample of frame F_1 (in our test data, each frame has 512 samples). We therefore have that $F_1(i) \approx -F_s(i)$. Of course, there is a small time differential between the two half-cycles but at 30 Hz, the motion over the ground is unlikely to exceed 1 m which shouldn't change much the geophysical response given that the emitter itself is about 10 m in diameter.

Figure 1. General morphology of a full cycle (simulated data)



We are therefore free to assume that $F_1(i) + F_s(i) \approx 0$ for any given cycle and that the remaining is noise-related or perhaps, software-related.

Figure 2. Comparing F1 and F2 visually, we can see some asymmetries by simply plotting the two signals together. A small peak can sometimes be seen at the end of the on-time for only the odd half-cycles.



Clearly, we don't have symmetry. It is clear that the peak we can see in half the half-cycles is not baseline dependent and is a system artifact. However, we still need to verify that the baseline correction satisfy some basic symmetry.

B. Invariance of the SIMn algorithm under sign change and translation

Let's note the raw signal $S(i)$ as a function of sample i. We will note T_{512} the translation of the signal by 512 samples (one half-cycle) so that $T_{512}(S)(i) = S(i + 512)$. Finally, let's note B the baseline correction so that $B(S)(i)$ gives you the baseline corrected signal. It can be shown that the SIMn algorithm is invariant under sign change : changing the sign of the data, computing the baseline and then changing the signs again amount to a normal baseline correction so that $-B(S)(i) = B(-S)(i)$ for every i.

In order to show that SIMn doesn't introduce any asymmetry, we need to show that $B(T_{512}(S))\ (i) = T_{512}(S)(i)$ for every i. It is best checked numerically... What we get considering the first frames of the Bob2 signal is that while the first frame we compare is different, the others are virtually identical. Therefore symmetry is preserved by the SIMs algorithm.

Table 1. Root Mean Square values of the error in selected frames. Values are in PPM so they have been scaled by $10^6/(\max S - \min S)$.

	$B(T_{512}(S))\ (i) - T_{512}(S)(i)$
R.M.S. of error in first frame	480 PPM
R.M.S. of error second frame	7 PPM
R.M.S. of error in all other frames	0 PPM

C. Is SIMn stable?

We need to verify that the baseline correction algorithm (SIMn) is numerically stable. Since the baseline is computed independently from half-cycle to half-cycle, requiring only continuity of the baseline [2], it is enough to show that the error made when requiring continuity with an anaccurate previous frame will, at most, propage with a lower amplitude to following half-cycles. Because the problem is linear, we can simply solve the Lagrange system for a null signal and $y_0 = \varepsilon$ (continuity condition) and show that the resulting polynomial is smaller than ε at $k_{max} + 1$ (see description of the algorithm above). This will imply exponential decay of the error through half-cycles.

We have to solve

$$2\sum_{k=k_1}^{k_{max}} k^i p(k) - \lambda_0 \delta_{0,i} - \lambda_1 \sum_{k=0}^{k_{max}} k^i = 0 \quad \text{where} \quad \delta_{i,0} = \begin{cases} 1 \text{ if } i = 0 \\ 0 \text{ otherwise} \end{cases}, \ i = 0,\dots,n;$$

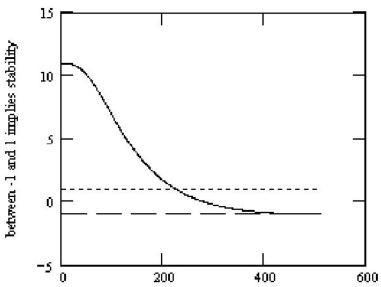
$$p(0) = y_0$$

and

$$\sum_{k=0}^{k_{max}} p(k) = 0.$$

For $n = 1$, we have $p(x) = \varepsilon - 2\varepsilon x / (k_{max} + 1)$ which implies $p(k_{max} + 1) = \varepsilon$ and so the algorithm isn't stable for $n = 1$ (but we never use $n = 1$ in practice). For n = 2, the mathematics get more tedious, but we can show that SIMn is unstable (as in $n = 1$). For $n = 3$ and $k_{max} = 511$, we have stability for $k_1 > 225$ as indicate in Fig. 3. Optimal stability ($p(k_{max} + 1) = 0$) is reached around $k_1 = 280$. At recommended settings ($k_1 = k_{max} - 48$), the algorithm is stable.

Figure 3. Stability analysis for SIM3. Notice that the algorithm is stable for $k_1 > 225$.



D. Does symmetry matter?

If instead of having $F_1(i) \approx -F_s(i)$ we have $F_1(i) \approx -F_s(i)\, a(i)$ where $a(i)$ is not constant over i, then stacking over a cycle will give us $F_1(i)$ - $F_s(i) \approx (1 + a(i))\, F_s(i)$. Since the factor $(1 + a(i))$ can be assumed to have no geophysical meaning, we are actually loosing or corrupting information. We can then ask whether or not $a(i)$ is significantly different from a constant. It should be noted that as long as $a(i)$ is more or less constant, it won't be a problem, so that, for example, the mean of $a(i)$ is irrelevant. Please note that $a(i)$ is not necessarily purely system dependent function and we expect a(i) to vary over time during a single flight.

One way to estimate $a(i)$ is to first estimate F_1 and then F_2 through stacking or averaging over a large number of cycles (we chose cycles 252 to 277 in the data file) see Fig. 4. We also scale the values in PPM (using a factor of $10^6/(\max F_k - \min F_k)$ where k = 1 or 2 respectively). We got a standard deviation of about 30 for $a(i)$ with a mean value of 0.127 : we would want a standard deviation close to 0 and might have expected a mean value near 1).

Table 2. F_1 and F_2 as computed by stacking using cycles 252 to 277 (using a conversion factor of 28 PPM/system unit)

	F1 (PPM)	***F2 (PPM)***	***F1/F2 = a***
Mean	-,065	-,005	,127
Standard deviation	1,28E+005	1,33E+005	29,8
Standard deviation of the second half	292	271	22,5

Figure 4. $a(i)$ defined by $F_1(i) \approx -F_s(i)\, a(i)$. Noticed that $a(i)$ is not a constant which shows asymetry. The data was smoothed using a moving median filter with a window of 21 samples.

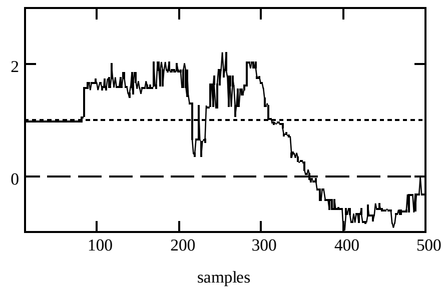
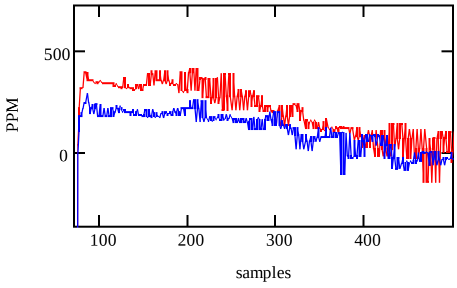


Figure 5. F_1 (on top) and F_2 (below) both converted in PPM (a factor of 28 was used for both F1 and F2 for convertion to PPM).



D. Conclusion regarding the baseline correction algorithm

From A, B and C above, we can deduce quite convincingly that the baseline algorithm cannot introduce any significant asymmetry in the signal such as asymmetrical odd and even pulses. This asymmetry is caused by the system itself and it is therefore recommended that stacking be done over full cycles when processing the data.

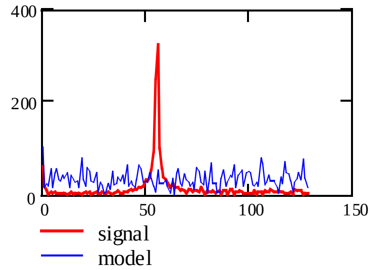
Signal Modeling

We use a model because it is the only approach that allows us to know exactly what the real signal is. In experimental data, one can only guess usually using a model, what the real signal is. Using a model from the start and polluting us allows us to evaluate scientifically our algorithms. It should be noted that we assume from this point forward that the baseline has been properly corrected. Notice in particular that this model doesn't take into account asymmetry between odd and even half-cycles.

A. White noise and stacking

We used white noise for our model, as shown in Fig. 6, this is clearly a rough approximation since the actual signal seems to exhibit colored noise. However, since the main purpose of our modeling is to test the baseline algorithm and similar low frequency issues, the model is still adequate.

Figure 6. White noise from our model compared with actual noise during the off-time. The actual signal shows a clear peak indicative of colored noise white white noise show a mostly flat spectrum. Colored noise doesn't diminish significantly with stacking and usually indicates a shielding problem.



B. Baseline correction

Applying the baseline correction algorithm (SIMn for n = 3) on the synthesized data allow us to check that while the algorithm isn't perfect, it doesn't introduce a systematic biais in the data (see Fig. 7 and 8).

Figure 7. Synthetic signal (on top) corrected with our SIMn algorithm (see below). Since the baseline deviation was simulated with a rapidely oscillating sine wave, we show that some of the time (see first half-cycle), the baseline correction isn't perfect. It is expected since we use polynomial (splines) as a modeling tool and a sine wave cannot always be perfectly approximated by polynomials.

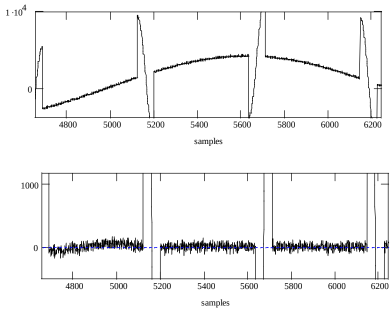
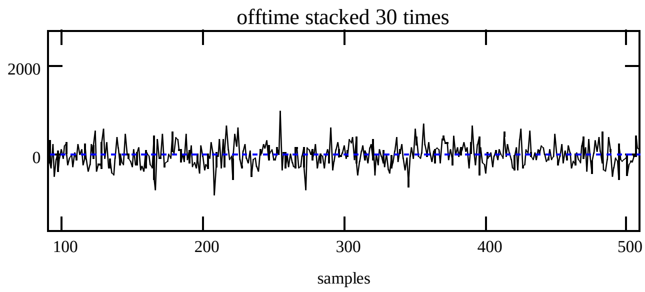


Figure 8. Stacking the corrected off-time signals show that the SIMn algorithm for the chosen parameters doesn't introduce a systematic bias. Overall, the off-time, once stacked is relatively flat as it should.



Tau projection

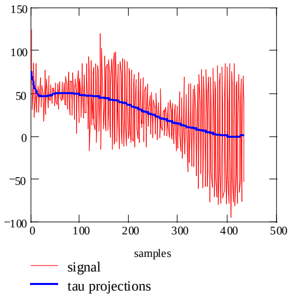
A. Algorithm

The algorithm is relatively simple. First we identify the onset of the off-time. This can be done by taking the derivative of the signal and assuming the the beginning of the off-time is defined as being a few samples after the largest local maximum of the derivative. We then project the off-time on the tau space defined as the space generated by basis functions $e^{tau_i x}$ where $tau_i = i \times 0.03 \times 1000 / (60 \times 512)$ for i = 0, ..., 18 (default values). Mathematically, the projection is simply the solution of the problem $\sum_x (e^{-Tau_j x} \sum_{k=1}^{n} A_k e^{-Tau_k x}) = \sum_x e^{-Tau_j x} s_i$ The projection is done using the GMRes algorithm which is known to be a very reliable algorithm for nearly singular problems such as this one (the basis $e^{tau_i x}$ is nearly linearly independent) by finding the A_k such that $\| \sum_x (e^{-Tau_j x} (s_i - \sum_{k=1}^{n} A_k e^{-Tau_k x})) \| \leq \epsilon$. It should be noted that such a projection isn't numerically stable, that is, if the signal is itself a sum of exponential

,we have no guarantee to be able to recover within some ε this sum. Indeed, we are inverting a matrix whose smallest eigenvalue is very small, of the order of 10^{-13} for the chosen tau_i values. In particular, this means that tau coefficients (A_k) are not meaningful.

As shown in Fig. 9 however, the projected data can be seen to approximate very closely the original signal without any of the high frequency noise.

Figure 9. Tau projection applied to F_1



B. Without tau projection (not recommended)

In order to make results easier to compare and understand, we start by showing results we get without tau projection (but the baseline was corrected, there was the usually stacking over 6 half-cycles and some wavelet shrinkage was also applied). The data of the Bob2.dat was used.

Figure 10. Typical off-time response without tau projection. Please note that raw data is usually presented on a larger scale (notice the values on the y axis).

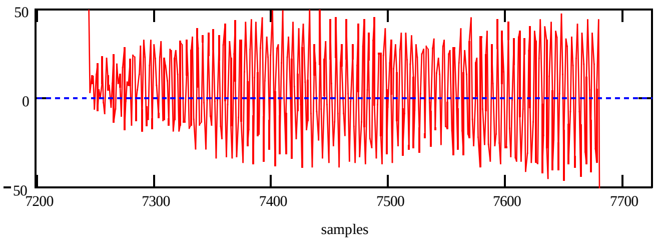
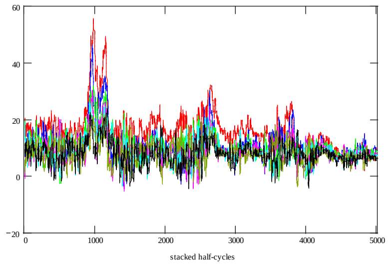


Figure 11. Channels without tau projection. As in other channels in this document, a narrow low-pass filter was used: running average over 10 samples. A more aggressive low-pass filtering would normally be recommended, but this figure should serve as a basis for comparison with what follows.



C. Tau projection (recommended)

While we can still see some residual lowpass problems in the off-time response, the tau projection works well and allows us to get clean channels with little effort. The only drawback seems to be that such an algorithm requires a fair amount of CPU cycles. From a research point of view, the tau projection allows us to see easily what is going on in the signal itself without having to worry about excessive high frequency noise. It should provide us with a very good foundation for future work. At the processing level, it is hard to imagine a better way to clean the signal.

Figure 12. Typical off-time responses after applying the tau projection (and stacking over 6 half-cycles). The signal is always rounded off to the nearest integer so that apparent small discontinuities in the signal should be ignored.

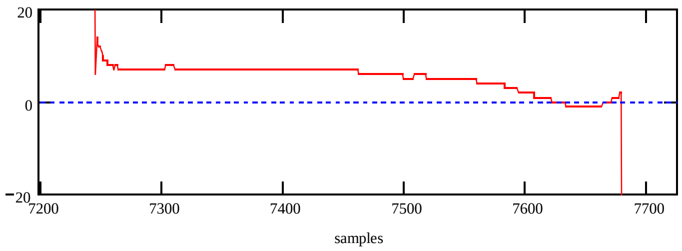
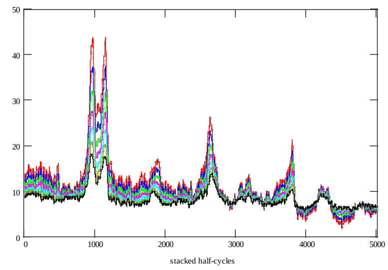


Figure 13. Some channels from the Tau projection (a running average with a window of 10 stacked half-cycles or 60 raw half-cycles is also applied).



B-field estimation

Still very much experimental, we achieve what seems to be an accurate estimation of the B-field: looking at the channels show that the anomalies are still visible (compare Fig. 13 with Fig. 15). Unfortunately, the results are not entirely convincing when looking at the channels only (anomalies are difficult to see), but the typical off-time response (see Fig. 14) is certainly very nice.

Figure 14. Typical off-time response after processing (stacked over 6 half-cycles). Notice how monotonic the response is and how little noise is present. The amplitude of the off-time response is higher in the B-field estimation.

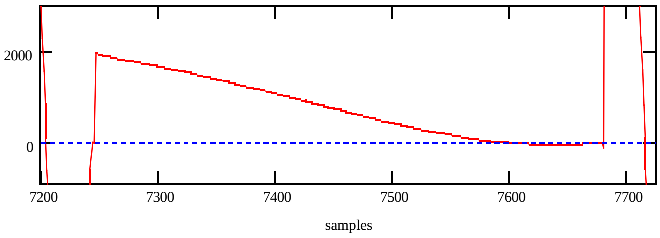
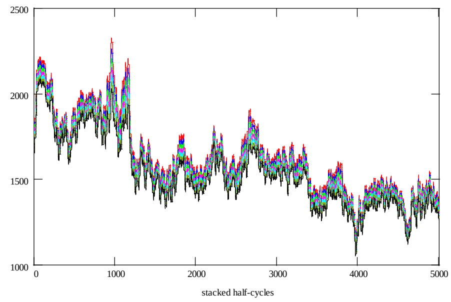


Figure 15. Some early channels for B-field estimation of stacked half-cycles (each channel amounts to 2 samples of stacked half-cycles). Interpretation seems difficult at this point. Further research is needed.



Software revisions

April 19th 2001
Some support for more recent THEM DAT files has been added to the DSP library.

March 8th 2001

Some prototypical B-field is not in place and tested (class IntegratedTauProjection). While it clearly works and provide stunning results, interpretation of the channels doesn't seem obvious at this point.

It should be noted that IntegratedTauProjection, just like TauProjection are expensive procedures and while the algorithms are robust and fast, it just takes a long time to do all the computations. Memory usage should be extremely low (please use the THEMFilter and THEM File classes as examples!).

March 5th 2001

Added ExtendedSIMn which derived from SIMn. Should provide a better baseline eventually? Worked during simulations, but doesn't work with real data most certainly because of the half-cycles asymetries which weren't taken into account in the modelling.

February 15 th 2001

I've added the GMRES algorithm and a few matrices and vectors to the lot. This allows us to proceed with the TauProjection class which should prove interesting. I hope to finish the BiSIMn algorithm which would correct the baseline over a full cycle (I'm hoping the result will be a notch above what we have right now).

I'm also improving the stacking by using the median to stack (instead of the average). This provide automatic rejection of bad half-cycles.

January 18th 2001

Corrected a stupid bug in the computation of the median (see SpecialMath class).

January 17th 2001

Added a Downsampling class and also modified slightly the SIMn class.

We can now import the data directly through DAT files. The library can now serve as an extractor!

January 12th 2001

- Checked the Wavelet Shrinkage for bugs.
- Changed all the memmove to memcpy (where do these memmove come from?)

January 11th 2001

- Converted all of the headers to the new namespace headers.
- Reading and writing binary files has been optimised (wewere actually reading and writing one sample at a time which is pretty stupid).

January 10th 2001

Changed memory allocation to handle failures more efficiently. (Replaced new by new(std::nothrow) wherever it was applicable and caught exceptions elsewhere.)

January 9th 2001

Changed name of precompiled headers class from StdAfx to PrecompHeader.

References

[1] Daniel Lemire, *Off-time denoising : is stacking the solution?*, Technical Report THEM2001-01, Montreal, January 12th 2000.
[2] Daniel Lemire, *Rapport sur un nouvel algorithme de correction de la ligne de base pour THEM Geophysics Splines, interpolation des moyennes et moindres carrés (SIM),* Technical Report, Montreal, June 12th 1999.

Daniel Lemire is a R&D consultant with a Ph.D.
in Engineering mathematics. He lives in Montreal
and can be found on the web at www.ondelette.com.