Clay S. Turner

Slope Filtering: An FIR Approach to Linear Regression

"DSP Tips and Tricks" introduces practical design and implementation signal processing algorithms that you may wish to incorporate into your designs. We welcome readers to submit their contributions to the Associate Editors Rick Lyons (R.Lyons@ieee.org) or Britt Rorabaugh (dspboss@aol.com).

While developing a communications receiver, the author encountered a problem that was efficiently solved using a method called *slope filtering*, which is a novel application of the standard equations of linear regression. This article introduces the slope filtering process and its effective use in applications such as communications receiver carrier recovery, signal rate of change estimation, and signal transition and transition-polarity detection.

MOTIVATION FOR SLOPE FILTERING

The slope filtering technique was developed while attempting to design a carrier recovery process for a communications system that used quadrature (I/Q) modulation. The communications receiver had the typical problem that its local oscillator was not frequency locked to that of the received signal's original transmitter oscillator. Since both oscillators (transmitter and receiver) have nearly constant frequencies over short time intervals, when the instantaneous phase of the received signal was compared to the phase of its corresponding ideal equivalent (known from the protocol), the resulting phase offset error function was approximately linear with respect to time. To properly demodulate the received signal the receiver must estimate, and then eliminate, this phase error between the ideal and received signals.

Because the receiver must operate with low signal-to-noise ratio (SNR) input signals, the linear phase error function's excessive noise suggested that a statistical method was needed for the receiver to estimate the phase offset error. It was in solving this problem that the author developed the slope filtering scheme to manipulate classic linear regression equations into a form designed for efficient computation. We now look at how this manipulation is carried out and then return to the communications receiver example.

LINEAR REGRESSION

The time-domain slope filtering process was developed by first borrowing the procedure of linear regression from the field of statistics [1], [2]. If we have a set of N ordered pairs $(x_0, y_0), (x_1, y_1), \ldots, (x_{N-1}, y_{N-1})$ and we fit, via a least squares method, a straight line through the data then the regression line is $\hat{y} = \alpha + \beta\hat{x}$, where

$$\alpha = \frac{\left(\sum_k y_k\right)\left(\sum_k x_k^2\right) - \left(\sum_k x_k\right)\left(\sum_k x_k y_k\right)}{N\left(\sum_k x_k^2\right) - \left(\sum_k x_k\right)^2} \tag{1}$$

and

$$\beta = \frac{N\left(\sum_k x_k y_k\right) - \left(\sum_k x_k\right)\left(\sum_k y_k\right)}{N\left(\sum_k x_k^2\right) - \left(\sum_k x_k\right)^2} \tag{2}$$

where $k = 0, 1, \ldots, N-1$. The slope of the best-fit line to an N-length segment of data is given by β. It is important to recall that (1) and (2) are valid for arbitrary sets of x_k values and there is no assumption of equal spacing for the x_k values.

As it is written, (2) doesn't appear amenable for efficient implementation in a signal processing application. But if we look at it from a DSP viewpoint and apply some adroit logic and manipulations, we can make (2) acceptable for such use. Since programmable DSP chips themselves are designed to efficiently calculate vector dot products, we seek to manipulate (2) into such a form. This means we need to somehow decouple the y_k samples from the x_k samples, except for a final dot product type of formulation.

So starting with (2), we first change the index counter for y_k from k to i and then change the product of sums in (2) to a double summation. These changes result in

$$\beta = \frac{N\left(\sum_i x_i y_i\right) - \sum_i \left(\sum_k x_k\right) y_i}{N\left(\sum_k x_k^2\right) - \left(\sum_k x_k\right)^2} \tag{3}$$

where index i, just like k, counts from zero to $N-1$. Next we factor out the y_k samples giving us

Digital Object Identifier 10.1109/MSP.2008.929816

$$\beta = \sum_i \left(\frac{N \cdot x_i - \sum_k x_k}{N\left(\sum_k x_k^2\right) - \left(\sum_k x_k\right)^2} \right) y_i. \tag{4}$$

Based on (4), we have our desired expression

$$\beta = \sum_i \beta_i y_i \tag{5}$$

where β_i is defined as

$$\beta_i = \frac{N \cdot x_i - \sum_k x_k}{N\left(\sum_k x_k^2\right) - \left(\sum_k x_k\right)^2}. \tag{6}$$

We have converted (2) into the vector dot product in (5) where the x_k and y_k samples are decoupled from each other. Fortunately if all of the x_i in (6) are known a priori, then the β_i coefficients may be precomputed and stored in memory. So, if we have a block of data (N samples), we just use our prestored length-N β_i coefficients in a tapped-delay line filter structure implementing (5) to find the slope (rate of change) of the data. This is the process we call slope filtering.

APPLICATION: RECEIVER CARRIER RECOVERY

Now that we see how to transform the regression formula into a useful (for signal processing tasks) form, we return to its application to the communications receiver problem. Because the receiver uses an independent frequency reference for heterodyning, the baseband signal will have a residual frequency and phase offset when compared to the ideal. We need to estimate these errors to be able to correct them.

To find the error function both the received and ideal signals are represented in complex polar form and then the differences in the arguments of their corresponding samples become the samples of the error function. For relatively short time duration signals, the error function is well approximated as a linear function where its slope represents the frequency offset and the intercept represents the phase offset between the transmitter's and receiver's oscillators.

By applying (5) to the error function, the approximate rate of change of phase is found. And of course the rate of phase change is equal to the frequency; thus we can estimate the frequency offset of the receiver. The regression calculation may be used to find the frequency offset by applying it to the phase error function. Unwrapping the error function's phase may be needed to preserve the linearity of the error function but, since its underlying structure is linear, unwrapping is quite easy. If (1) is also manipulated into a dot product formulation, and it is applied to the error function, the initial (at the start of the data segment under analysis) phase error is found. A digital quadrature oscillator [3], initialized with the negatives of the frequency and phase offsets, can be used to correct the received signal's frequency and phase errors by simple modulation.

APPLICATION: SIGNAL RATE OF CHANGE ESTIMATION

REAL-TIME RATE OF CHANGE ESTIMATION

For applications where continual signal rate of change estimations are needed, as opposed to the single dot product calculation of (5), when the x_i samples are equally spaced and ordered from low to high we can write $x_k = x_0 + k$ where $k = 0, 1, 2, \ldots, N-1$. Hence we are looking at a set of N consecutive equispaced x_k samples. Substituting our expression for x_k into (6) results in

$$\beta_i = \frac{N(x_0 + i) - \sum_k (x_0 + k)}{N\left(\sum_k (x_0 + k)^2\right) - \left(\sum_k (x_0 + k)\right)^2} \tag{7}$$

which after significant algebraic simplification becomes

$$\beta_i = \frac{12 \cdot i - 6(N-1)}{N(N^2 - 1)} \tag{8}$$

giving us the β_i coefficients for use in a N-stage tapped-delay line finite-impulse response (FIR) structure to compute our desired rate of change β in (5).

If the delay line structure used to implement (5) is designed for filter convolutions instead of correlations, then the β_i coefficients need to be reversed in order. Since the β_i coefficients have negative symmetry, a simple negation will be sufficient as shown in Figure 1.

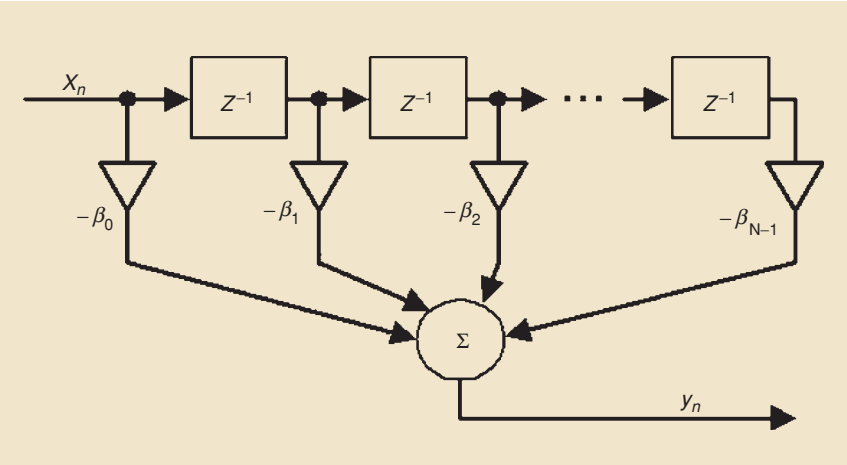
The units for β in (5) are the units of y per sample interval. So, for example, if y is in volts and we are sampling at a 1 kHz rate, then the units of β are volts per millisecond.

PROPERTIES OF RATE CHANGE OF CHANGE ESTIMATION

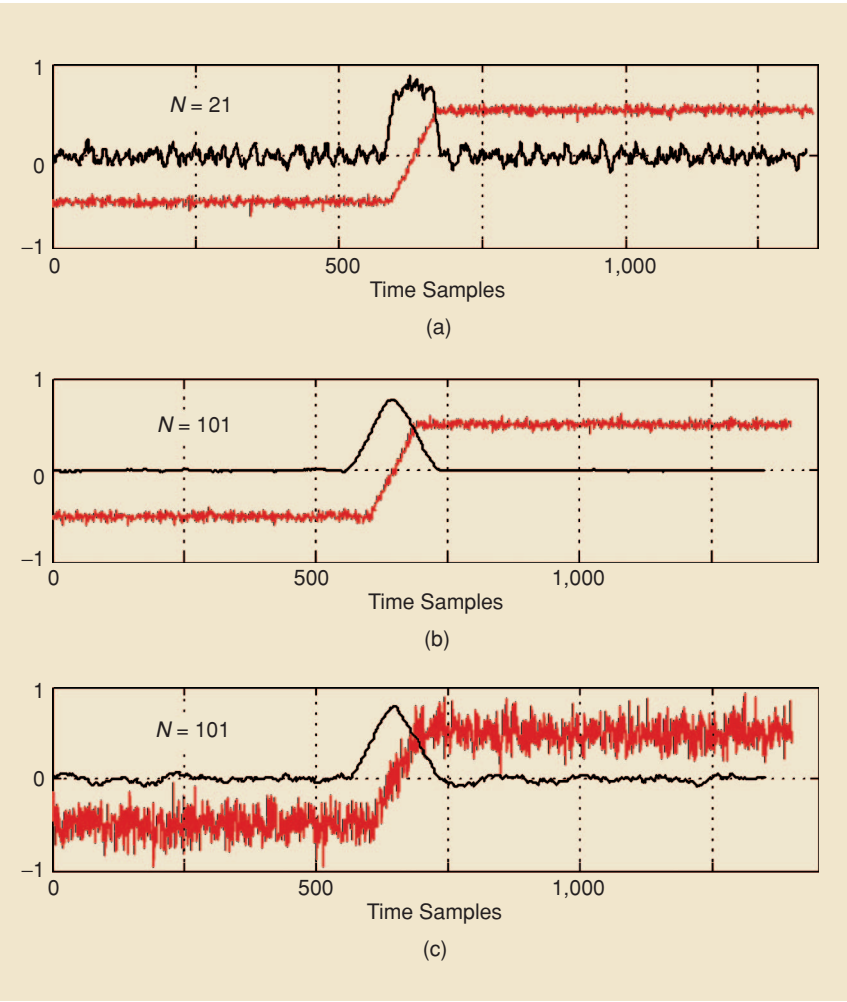
Equation (8) is remarkable for several reasons.

- It represents the coefficients of an odd symmetric linear-phase FIR filter, which means the delay is a constant of $(N-1)/2$ samples. In addition, half of the multiplications in (5) may be eliminated by using a "folded FIR" filter structure.
- Its result is origin independent! The β_i coefficients don't change as we traverse the data as evidenced by the fact the starting value x_0 is not a part of (8). So in terms of implementing (5) via a filter, this means the coefficients are time-invariant.
- The β_i coefficients define a linear ramp. This means we are in effect correlating our signal with a ramp to find the slope. This may be viewed as optimal in the sense of matched filtering allowing the user to easily pick an optimal N. That is, if our transitions are expected to span M samples, then we make the β_i coefficients M samples in length. And this brings us to the important observation that transitions usually don't exist in isolation, they are preceded and followed by generally level signals. This means that the span of the β_i coefficients may be longer than just the transition width. In fact this *super spanning* can increase the SNR of our slope filter's output! Of course there are practical limits to

the amount of super spanning, but having a span 20% or 50% greater than the transition width is certainly acceptable. In the case of pulse detection with matched filtering in mind, the number of β_i coefficients may be set to the number of samples comprising the pulse. But if super-spanning is appropriate, a greater number of coefficients may yield the highest SNR.



[FIG1] **Real-time slope filtering structure.**



[FIG2] **Time-domain slope filtering examples: (a) span $N = 21$, (b) $N = 101$, and (c) $N = 101$ with a very noisy input signal.**

APPLICATION: SIGNAL TRANSITION DETECTION

Another useful application of time-domain slope filtering is signal transition detection. Figure 2 illustrates this idea where a series of noisy input signals with transitions are the red traces, and the outputs of the slope filter (delay compensated on the graphs) are shown as the black traces. In that figure, the length of the input signal's transition is 80 samples. Examples of super-spanning are shown in Figure 2(b) and (c).

Why does time-domain slope filtering work so well when compared to ideal differentiators? By inspecting the coefficients for each of these cases, we find that the regression method places the strongest weights at the ends of the data span and conventional differentiators weigh heaviest near the middle. Thus from a "torque" calculation point of view, conventional differentiators rely strongly on fewer terms (all near the middle) and therefore the result is more susceptible to noise. One can also note that the difference is that in time domain slope filtering a first-order polynomial is fitted to the data, whereas with a traditional differentiator a low-order trigonometric polynomial is fitted to the data.

Even if a differentiator is designed with the Parks-McClellan (PM) algorithm to closely match the performance shown in Figure 2(a), it will generally not fair well when the noise levels are increased. That is because a trigonometric polynomial fit admits polynomial terms higher than first order. To see this, recall that the PM algorithm fits a weighted sum of sinusoids to the differentiator's desired frequency response. And, when transformed back to the time domain, its impulse response will contain polynomial terms higher than the first power.

APPLICATION: SIGNAL TRANSITION-POLARITY DETECTION

Our noise-tolerant time-domain slope filtering scheme also comes in handy in applications where one just wishes to know if a signal is increasing or decreasing, for example when detecting the polarity of signal transitions. In this case, we can multiply (8) by any positive number, since we only need the sign (positive or negative) of (5). For a start, just remove (multiply it out) the denominator of (8). This results in

$$\hat{\beta}_i = 12i - 6(N-1) \tag{9}$$

and also divide by 12 to yield

$$\tilde{\beta}_i = i - \frac{1}{2}(N-1). \tag{10}$$

Now (5) becomes (for polarity detection only)

$$\text{Polarity}_{\text{odd } N} = \text{sgn}\left[\sum_{i=0}^{N-1}\left(i - \frac{N-1}{2}\right)\cdot y_i\right]. \tag{11}$$

For example with $N = 7$, we have

$$\text{Polarity}_{N=7} = \text{sgn}[-3y_0 - 2y_1 - y_2 + y_4 + 2y_5 + 3y_6]. \tag{12}$$

And for even N, we can rescale (10) by doubling all its coefficients so they are integers. For even N, this formulation results in

$$\text{Polarity}_{\text{even } N} = \text{sgn}\left[\sum_{i=0}^{N-1}(2i - (N-1))\cdot y_i\right]. \tag{13}$$

For example when $N = 6$ we have

$$\text{Polarity}_{N=6} = \text{sgn}[-5y_0 - 3y_1 - y_2 + y_3 + 3y_4 + 5y_5]. \tag{14}$$

The polarity detection algorithms in (11) and (13) are expressed in forms amenable to fast calculation.

CONCLUSIONS

We presented a novel way of manipulating the standard equations of regression analysis into a form that is highly compatible with the methods of digital signal processing and, from them, derived a computationally efficient slope-filtering differentiator, as defined in (5) and (6), that is superior in its noise tolerance relative to PM designed differentiators. We extended the algorithm with a focus on both transition detection and transition-polarity detection in (11) and (13).

AUTHOR

Clay S. Turner (Clay.Turner@ PaceOMatic.Com) is senior technical director for Pace-O-Matic, Inc. He has over 30 years experience in digital signal processing, mathematical and embedded programming. When not busy, he likes to photograph, scuba, backpack, cycle, and go horseback riding with his wife.

REFERENCES

[1] M. Spiegel, *Theory and Problems of Statistics* (Shaum's Outline Series). New York: McGraw-Hill, 1961, p. 220.

[2] P. Meyer, *Introductory Probability and Statistical Applications*. Reading, MA: Addison-Wesley, 1965, pp. 137–139.

[3] C. Turner, "Recursive discrete-time sinusoidal oscillators," *IEEE Signal Processing Mag.*, vol. 20, no. 3, pp. 103–111, May 2003. SP

---

from THE EDITOR continued from page 4

for ensuring the high quality of the articles published in *SPM*. We are indebted to Professor Ray Liu, in his role of VP-Publications, and Mercy Kowalczyk, SPS executive director and *SPM* associate editor, for their valuable advice. Eric Zavesky of Columbia University has provided critical assistance in maintaining the Web sites and review database. Finally, the senior managing editor of *SPM*, Geri Krolin-Taylor of IEEE, together with her editing staff, once again demonstrated the highest level of professionalism, creativity, and work ethic. They are indispensable assets for any editor running a dynamic publication like *SPM*.

With this issue, the magazine will be transitioning to a new team of capable hands. Li Deng, current area editor for feature articles, will be at the helm serving as the new editor-in-chief. He will be joined by Min Wu (incumbent area editor for newsletter) and new blood, Antonio Ortega, leading the feature articles section, Dan Shonfeld leading the special issues section, and Ghassan AlRegib leading the columns/forum section. These individuals are respected pioneers and active volunteers in the Society. I have no doubt that, with their collective vision and leadership, *SPM* will continue its successful momentum and move to an even higher stature. I now look forward to joining all of you, the readers, in not only enjoying the articles but also more importantly writing articles and providing feedback for *SPM*. SP