A Model of Noisy Software Engineering Data* (Status Report)

Roseanne Tesoriero and Marvin Zelkowitz
Department of Computer Science and
Institute for Advanced Computer Studies
University of Maryland
College Park, Maryland USA
+1-301-405-2690
{roseanne, mvz}@cs.umd.edu

ABSTRACT

Software development data is highly variable, which often results in underlying trends being hidden. In order to address this problem, a method of data analysis, adapted from the financial community, is presented that permits the shape of the curve of some activity to be reduced to a few line segments, called the characteristic curve. This process is used on sample data from the NASA/GSFC Software Engineering Laboratory and has shown to be a reasonable method to understand the underlying process being plotted.

KEYWORDS

Curve fitting, Exponential moving average, Financial data

1 INTRODUCTION

In this report we will look at an analysis technique, adapted from the financial world, that yields reasonable results for noisy project data. We first describe our environment and the type of data we desire to understand. then we describe the underlying model used to analyze the data.

1.1 Software Engineering Data

As part of our research with the Software Engineering Laboratory (SEL), which has been studying software development practices in the flight dynamics area since 1976 [2], we have been enhancing the Software Management Environment (SME) into a Web-based data visualization tool called WebME. WebME (and its SME predecessor) retrieves data from the SEL database and provides the user the ability to plot various attributes (e.g., staff hours, errors, lines of code) and to compare a given project's attributes with the same attributes from other projects in the database.

*This research supported in part by NASA grant NCC5170 to the University of Maryland.

Lines of code produced, staff hours, error reports filed, change reports filed, modules created, and modules changed are all reported weekly by project personnel or are extracted automatically from the source code libraries. Once in the data base, we can plot these as raw data weekly scatter plots or as various growth functions to show how the collected statistics change over time.

The original SME was driven by parameters entered by the project manager. Certain dates (e.g., end of design, start of acceptance testing) were considered critical points in determining important milestones for a project. However, such dates often appear arbitrary in hindsight as a project seems to take on a life of its own. Can one develop these critical dates by analyzing the data itself and determining the underlying relationships among the attributes without resorting to manager-supplied data?

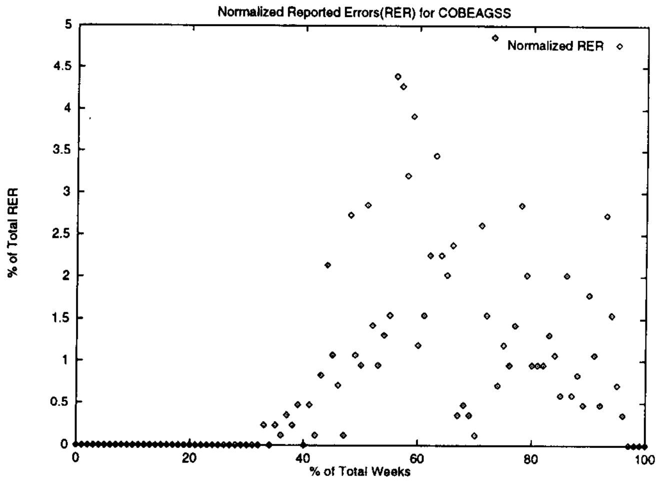

Figure 1: Scatter plot data of reported errors by week

Figure 1 represents the weekly number of error reports filed for a certain NASA project.[1] This data is quite

[1] All data presented here is normalized from 0% to 100%. That allows us to compare multiple projects on the same graph. The time duration for the projects considered here range from 100 to 120 weeks – about 2 years.

noisy and it is hard to see any trend or underlying model in the data. Is there any underlying process that determines how many errors are found each week? Can we make any reasonable models of this process?

1.2 Financial models

One way to partition the data is based on using trend changes as a signal for process changes. In the financial markets, the price of a stock or commodity is highly variable. An investor's objective is to buy at a minimum price and sell at a maximum price. However, because prices fluctuate frequently, an investor would not want to trade at every trend change in the market.

The problem of trend detection for financial data turns out to be similar to our problem. We have highly variable data and we want to detect major trend changes while ignoring minor fluctuations. Techniques used to detect trend changes with financial data should be applicable to our domain.

In particular, financial markets look at long term versus short term trends. Moving averages have long been used in this domain, where an N-day moving average is the average value of some feature over the past N days. If the long term average (i.e., using a large value of N) is greater than the short term average (i.e.. using a small value of N), then a stock has a decreasing trend in value; otherwise it is increasing. Such trends eliminate the daily fluctuations inherent in this form of data. If the trend moves from negative to positive, then its price has presumably reached its minimum and should be bought. If the trend moves from positive to negative, then it has peaked and should be sold since waiting will only decrease its price.

The *Moving Average Convergence/Divergence* (MACD) trading system [1] [4] determines when the long term changes in a stock's value differs from the short term changes, which signals a decision to buy or sell the stock. When the trend crosses the *signal line* (i.e., the moving average of the long term average less the short term average) in a positive direction, the price is about to rise and a stock should be bought; if it crosses the signal line in the negative direction, a sell is indicated.

2 EQUATION MODELING

Based on the MACD examples, we have developed a three step process for analyzing each data attribute. Given the raw scatter plot data for some attribute (such as given in Figure 1), we want to reduce it to several linear segments that best represent the governing processes during the period represented by each segment. We will call this the *characteristic curve* and our initial goal is to find the end points for each such linear segment, which we call the *pivot points* to this curve. Once we do that, we can apply more traditional curve fitting techniques to each segment in order to develop underlying models of each process.

The three steps we have developed are:

1. Use smoothing techniques to provide a rough envelope that represents the approximate behavior of the data. This process is not sufficient by itself. For example, the data of Figure 1 results in a smoothed curve (Figure 2) which still has 12 local maxima when using an 8 point moving average.

2. Determine which of the extreme points represent a significant event for these processes. Other local maxima (or minima) are assumed to be minor perturbations in the data and are to be ignored. We call these significant trend changes *pivot points.*

3. Connect the set of pivot points into a segmented line. This represents the *characteristic curve* for the original raw data.

2.1 Modeling Algorithm

We outline the algorithm in the following sections:

Data Smoothing. In order to remove day to day variability in the value of a stock, N-day moving averages are used. Often a short range moving average (e.g., 30 days) is compared with a longer range moving average (e.g., 150 days) in order to compare local changes to a stock's price compared with the longer range trend. The crossover points between the short and long term moving averages signal trend reversals.

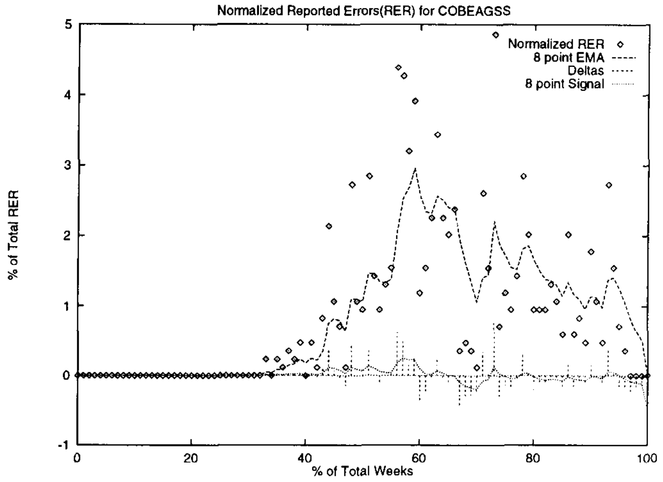

Figure 2: Smoothed data using moving averages

This *simple moving average*, however, has a weakness. If a critical point is reached (e.g., the value reaches a maximum), the damping effects of the earlier points in the average delay the signaling of this phenomenon.

That is, the moving average will continue to rise for several days after the peak is reached since all points are weighted equally in computing the average. In order to enhance the perception of such directional changes, the *exponential moving average* (EMA) is used for the MACD trading system described earlier. Rather than being the simple average of the last N points, the exponential moving average is given by the equation:

$$EMA_i = (1 - \frac{2}{N+1}) * EMA_{i-1} + \frac{2}{N+1} * v_i$$

where:

EMA_i is the exponential moving average at time i

v_i is the new data value at time i

and $\frac{2}{N+1}$ is the smoothing constant where N is the number of points in the average.

For $N = 9$, $\frac{2}{N+1}$ has a value of .2 meaning each new point has about twice the "impact" (20% instead of 11%) that a simple moving average would have. Each successively older point has less of an effect on the total average. and the result is a moving average more sensitive to leading edge changes.

The higher curve in Figure 2 shows the effects of the EMA on the error data of Figure 1. From this EMA of the scatter plot data, we want to extract only those maxima and minima that represent significant changes in the underlying process.

Computation of derivatives. If we could simply take the derivative of this curve, we could solve for the derivative being zero in order to find the local maxima and minima. However, the actual (smoothed) data does not permit such computations. We can use the EMA to help again for this process. Between any two points we can compute the instantaneous derivative $\delta_i = \frac{\Delta v_i}{\Delta t}$. If we compute this for each time period t, and take the EMA for these delta values, we get what is called in the financial community the ***signal line*** (Figure 2). Where the signal line crosses the X-axis represents a zero EMA, or in other words, the average δ_i in the interval is 0, which represents an extreme value for the curve.[2] In our example, this signal line crosses the X-axis 7 times. Each of these represents a critical point in the original data.

What does this signal line represent? It is the average slope of the instantaneous derivatives for the past N points. If the signal line is 0, it means that the average delta between successive points is 0 and we have a local maximum or minimum. We simply have to go back over the last N points to determine which value of time t_i represents that extreme value. We call such values *pivot points.*

[2]In the original MACD development, the signal line was the EMA of the difference between the long term and short term EMA. Here we are only concerned with the slope of the δ_i curve.

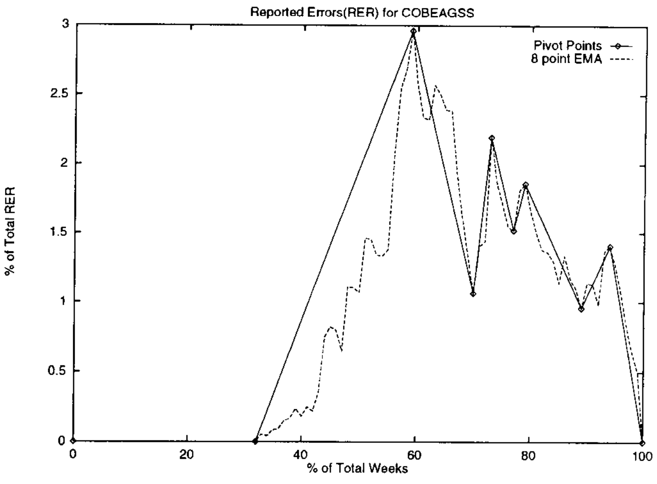

Figure 3: Characteristic Curve for original data

Computation of Characteristic Curve. Once we have identified the pivot points, we connect each segment with a straight line (Figure 3). This segmented line describes the general shape of the curve we are interested in. We have been able to eliminate minor hills and valleys from the curve and have left only the major features of the original data.

2.2 Applicability to NASA data

We applied the analysis presented here to multiple projects from our SEL database, and we were able to generate characteristic curves containing 5 to 9 pivot points for all data attributes.

In order to compare different projects, in Figure 4 we show three different error characteristic curves from different projects. All three, however, represent similar applications developed using similar development processes. What is interesting is that two of these projects, COBEAGSS and GOESAGSS, have characteristic curves that appear quite similar (e.g., both have an initial peak at time 60%, a value close to 3% of total errors, and a slow decay toward 1% as time approaches 100%), while the third curve differs significantly from this behavior.

One interest to us was the first dip noticed between 60% and 80% of project completion in the error data of Figure 3. Looking through old records (from 1988) we discovered that the minima pivot point at 70% occurred just before the start of acceptance testing for the project. This was also the time when the contracting organization that built the software moved into a new building. In the other cases studied, however, the same drop before acceptance testing was found. The

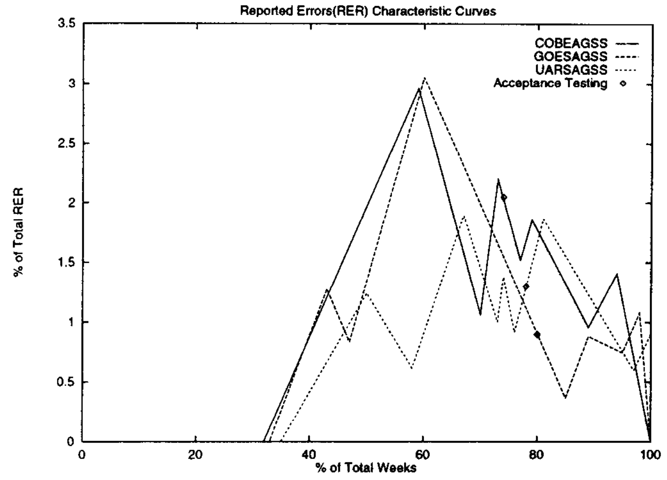

Figure 4: Multiple characteristic curves for 3 projects

characteristic curves have identified a potential area for improvement.

These examples reveal characteristics of the error data that were hidden by the raw scatter plots of error rates. In addition to uncovering areas for improvement, given a baseline characteristic curve, the impact of process changes can be visualized with our technique. For example, if the goal of a process change was to eliminate the discovered dip in reported errors between integration and acceptance testing, by computing the characteristic curve of the new projects and comparing them to the existing baseline curve, we could see the impact of the process change on our goal.

3 CONCLUSIONS

In this report we have addressed the problem of developing a characteristic curve for noisy data based upon a concept we have called the pivot point of a curve. We have adapted the exponential moving average, used previously within the financial community, to our software engineering development data. We have developed a simple algorithm that reduces such noisy data into 5 to 9 line segments which show the general shape of the measured data.

This work leads to several immediate follow-on studies:

1. In our data (Figure 3) it seems apparent that the first segment should be concave upwards to obtain a better fit. We are looking at using standard regression tests for these intra-pivot point segments.

2. We need to understand the impact of each pivot point. We believe we can identify the start of acceptance testing by the shape of the characteristic curve. What about other milestones?

3. Can we catalog a project by the shape of the characteristic curve? The COBEAGSS and GOESAGSS error curves both reach an early peak and then slowly decay while the UARSAGSS curve reaches a relatively constant steady-state error rate during much of development. Can we use cluster analysis to clump together similarly shaped characteristic curves?

We believe the most important aspect of this work is showing that raw software development data is noisy and that proper smoothing and analysis techniques can be used to extract valuable trend information. Figure 5 shows a structure to the data that is simply not apparent when viewing the original scatter plot.

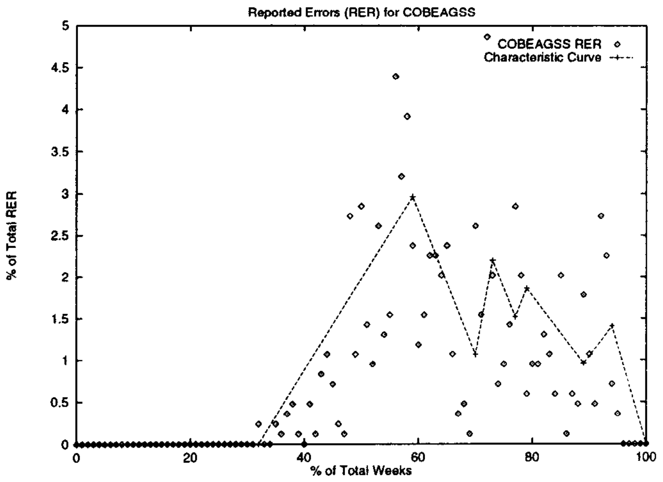

Figure 5: Raw data with its characteristic curve

REFERENCES

[1] Appel G. and W. F. Hitschler, *Stock market trading systems.* Dow Jones-Irwin, Homewood, Illinois, 1980.

[2] Basili V., M. Zelkowitz, F. McGarry, J. Page, S. Waligora, and R. Pajerski, SEL's software process-improvement program, *IEEE Software* 12, 6 (1995) 83-87.

[3] Pring M. J., *Technical Analysis Explained.* McGraw Hill, 1991.

[4] Seykota E., MACD: Sweet anticipation?, *Futures* 20, 4 (March, 1991) 36+.

[5] Zelkowitz M. V., Advances in software engineering resource estimation, *Advances in Computer Programming Management* 1, Heyden Publishing Co., 1980, 206–225.