Automated Steady and Transient State Identification in Noisy Processes

R. Russell Rhinehart, *Fellow ISA*
professor emeritus, Oklahoma State University, rrr@okstate.edu
principal, r3eda, www.r3eda.com, russ@r3eda.com

Abstract **– Five approaches are developed, analyzed, and demonstrated for automated identification of probable steady state and probable transient state in noisy process signals. The methods are insensitive to noise variance. The simplest method (filter) and a more complicated version (array) are based on a ratio of variance measured by two methods. MACD (moving average convergence divergence), a technique commonly used in analysis of stock price trends, compares a fast and slow filter on the data and here, is normalized by a measure of data variability. It is also relatively simple. A statistical process control method appropriated from a Shewhart X-Bar & R chart is a the most computationally intensive. Finally, a four-point filtered version has similar concepts, but is much simpler to implement. A performance assessment of Type-I and Type-II errors and speed of response indicates that the X-Bar-R method is best, with the 4-point version a close second. Overall, including simplicity, the 4-point method is judged as best. This monograph will develop the equations, reveal the execution code, discuss implementation and critical values, and compare the approach to other approaches (w.r.t. computational burden, sensitivity, average run length, etc.).**

I - INTRODUCTION

General Applications

Identification of both steady state (SS) and transient state (TS) in noisy process signals is important. Steady state models are widely used in process control, on-line process analysis, and process optimization; and, since manufacturing and chemical processes are inherently nonstationary, selected model coefficient values need to be adjusted frequently to keep the models true to the process and functionally useful. Additionally, detection of SS triggers the collection of data for process fault detection, data reconciliation, neural network training, the end of an experimental trial (when you collect data and implement the next set of conditions), etc. But, either the use of SS models or their data-based adjustment should only be triggered when the process is at SS.

In contrast, transient, time-dependent, or dynamic models are also used in control, forecasting, and scheduling applications. Dynamic models have coefficients representing time-constants and delays, which should only be adjusted to fit data from transient conditions. Detection of TS triggers the collection of data for dynamic modeling. Additionally, detection of TS provides recognition of points of change, wake-up data recording, the beginning of a process response to an event, interruptions to the norm, etc.

The detection of both SS and TS can be useful in automating a sequence of experimental conditions. Often, process owners and investigators run a sequence of experiments to collect data throughout a range of operating conditions, and process operators sequence the next stage of a trial. Each sampling event is initiated when the operator observes that steady conditions are met. Then the operator implements the new set of operating conditions. Similarly, in batch operations the end of a stage is evidenced by signals reaching their equilibrium or completion values, and when operators observe that the stage is complete, they initiate the next step in the processing sequence. However, this visual method of triggering requires continual human attention, and it is subject to human error in the recognition of steady state. Features that can compromise the visual interpretation include: noisy measurements, slow process changes, multiple dynamic trends, scheduled operator duties, upcoming lunch breaks, or change-of-shift timing.

Alternately, the experimental run or batch stage can be scheduled to go to the next set of conditions at preset time intervals. Unfortunately, this method can create inefficiency if the runs are scheduled for an unnecessarily long time, or the data can be worthless if the scheduled time is insufficient for any particular set of conditions to achieve steady state. Since the time to reach steady state varies with operating conditions, it is nearly impossible to predict the necessary hold time.

If SS detection were automated, process sampling or data recording would be initiated; and after, the computer could autonomously implement the next operational stage or set of experimental conditions. But, likely on the first sampling after the new run signals go to the process, the process will not have responded, and the process output signals will remain at their prior steady state. To prevent this past SS from triggering the next trial, the computer should first seek a TS after implementing new conditions, then seek the resulting SS to trigger data collection and to implement the next run.

An automated online, real-time SS and TS identification (SSID and TSID) would be useful to trigger the next stage of an experimental plan or process phase.

The utility of SS or TS detection extends beyond chemical process management. TS could be used to detect motion and provide security warnings, etc. SS can be used to identify that the optimization in either nonlinear regression or stochastic applications have converged.

However, other applications may have characteristics that will indicate the approach proposed here is not the best. I am exploring the SSID and TSID to chemical process applications. Chemical processes are characterized by time-constants on the order of 1 second to 1 hour, they are multivariable (coupled and nonlinear), with mild and short-lived autocorrelation, variance changes with operating conditions, controllers are tuned to avoid oscillation, and flatlining measurements are not uncommon (for any number of aspects such as maintenance, sensor failure, data discretization). Additionally, control computers are inexpensive, and the operators have education typically at the associate degree level; both aspects require simplicity in algorithms. The approaches presented here might not be right if mission criticality can afford either powerful computers or highly educated operators, or if rotating machinery creates a cyclic response as a background oscillation to the steady signal.

Concepts and Issues in Detecting Steady State

If a process signal was noiseless, then SS or TS identification would be trivial. At steady state there is no change in data value. Alternately, if there is a change in data value the process is in a transient state.

However, since process variables are usually noisy, the identification needs to "see" through the noise and should announce probable SS or probable TS situations, as opposed to definitive SS or definitive TS situations. The method also needs to consider more than the most recent pair of samples to confidently make any statement.

Since the noise could be a consequence of autocorrelated trends (of infinite types), varying noise amplitude (including zero), individual spikes, non-Gaussian noise distributions, or spurious events; a useful technique also needs to be robust to such aspects.

Finally, in observing data, a process might appear to be at steady state, due to measurement discrimination intervals, when in fact it is changing but the change has not exceeded the data discretization interval. Time, as an example, continually progresses; but my digital watch only updates the screen on one-minute intervals. Time is not at steady state between the numerical display change events.

Approaches and Issues to SSID & TSID

A process might not need to be exactly "flat-lined" to be considered at SS. For practical purposes, a very small trend might have a negligible impact on SS data uses, or a small oscillation about an average value might be negligible. Accordingly, a conceptually simple approach would be to look at data values in a recent time-window and if the range between high and low is acceptably small, declare SS. This approach, however, requires the human to decide the acceptable range for each variable, and the time window; and, if the process noise level changes, the threshold should change. Although the approach is simple to understand, easy to implement, and often works acceptably, it is not a universal approach.

Another straight-forward implementation of a fully automated method for SS ID would be a statistical test of the slope of a linear trend in the time series of a moving window of data. This technique is a natural outcome of traditional statistical regression training. Here, at each sampling, use linear regression to determine the best linear trend line for the past N data points. If the process is at SS, then the trend-line slope will be, ideally, zero. However, because of the vagaries of process noise, the slope will fluctuate with near-zero values; accordingly, a non-zero value is not cause to reject the SS hypothesis. If the t-statistic for the slope (regression slope divided by standard error of the slope coefficient) exceeds the critical value, then there is sufficient evidence to confidently reject the SS hypothesis and claim it is probably in a TS. A nice feature of this approach is that the determination is independent of the noise amplitude. However, at the crest or trough of an oscillation centered in the N-data window, this slope would be nearly zero, and SS will be claimed during the TS. The approach is also somewhat of a computational burden.

Alternately, another straight-forward approach is to evaluate the average value in successive data windows. Compute the average and standard deviation of the data in successive data sets of N samples then compare the two averages with a t-test. If the process is at SS, ideally, the averages are equal, but noise will cause the sequential averages to fluctuate. If the fluctuation is excessive relative to the inherent data variability, then the t-statistic (difference in average divided by standard error of the average) will exceed the critical t-value, and the Null Hypothesis (Process is at SS) can be confidently rejected to claim it is probably in a TS. Again, however, when windows are on either side of the crest or trough in an oscillation, the averages will be similar and SS will be falsely accepted. A solution could be to use three or four data windows, each with a unique N, to prevent possible matching to a periodic oscillation.

Note, both of these methods reject the null hypothesis, which is a useful indicator that the process is confidently in a TS. But, not rejecting SS, does not permit a confident statement that the process is at SS. A legal judgment of "Not Guilty" is not the same as a declaration of "Innocent". "Not Guilty" means that there was not sufficient evidence to confidently claim "Guilty" without a doubt. Accordingly, there needs to be a dual approach that can confidently reject TS to claim probably in a SS, as well as rejecting SS to claim probable TS. This tutorial presents several dual approaches.

Further, such conventional tests have a computational burden that does not make them practicable on-line, in real-time, within most process control computers. This tutorial presents some computationally simple, as well as effective approaches.

This tutorial develops, analyzes, and demonstrates methods for probable SS and probable TS identification that are both robust to many process events and are computationally simple. The approaches have been extensively tested in lab- and pilot-scale processing units for both SS and TS ID, for the autonomous segregation of data, for model adjustment, and for the autonomous triggering of experimental plans. Several methods have also been

applied to commercial-scale multivariable processes. Beyond application to monitoring processes, the approach is regularly used to detect convergence in optimization.

Building on the discussion in Szela and Rhinehart (2003), here is a list of fifteen problems associated with automatically detecting SS and TS states in noisy processes. Any practicable method needs to address all of the issues.

1. Since processes produce noisy data, steady state identification must accommodate noisy data. When the process has settled to a final steady state, the measurements do not settle at single values. Various sources of noise cause the measurements to vary, sort of randomly. The average might settle in time, but the individual data keeps changing, fluctuating about the time-average. Statistical methods are required to identify SS or TS.
2. Statistical tests normally reject the null hypothesis. This means that normal tests would determine when there is a high probably that the process is not at steady state. This leaves the other possibilities unidentified. The other possibilities are that the process is at steady state, or that the process is still settling but not changing at a high enough rate to confidently claim "not at steady state".
3. The process noise level can change with operating conditions. This means that the methods for statistical testing must adapt to the local/temporal process noise amplitude.
4. Processes are subject to external, uncontrolled, disturbances. Effects of these on the process response, when significant, must be identified as "not at SS".
5. Sometimes spurious, one-time and extreme, perturbations misrepresent the signal, perhaps from a missed data connection or a substituted variable due to the communication system. Effects of these, must be recognized.
6. Processes are multivariable. Several variables must be monitored, and the statistical tests must accommodate the increased uncertainty.
7. Related to #6, cross correlation between variables affects the statistical level of significance – if two variables are correlated, they are not independent, and only one needs to be included in the multivariable analysis.
8. Process state measurements do not immediately respond to the initiated change. If steady state is recognized, and the next set of conditions in a trial sequence is implemented, it would not be unexpected for the immediate next sampling to find the process measurement values at the same steady state. This might cause the second-next conditions to be implemented too soon. A cyber operator, a virtual employee, an automaton must wait for the effect of the new conditions to be observed, a transient, prior to looking for the subsequent steady state.
9. The computational load that the method imposes on the computer system needs to be minimal, compatible with standard PLC-, PC-, and DCS-type products in use.
10. The noise distribution might not be Gaussian (normally) distributed. The method needs to be robust to the underlying noise distribution.
11. The noise might be autocorrelated. In autocorrelation, when one fluctuation is high (or low), what caused that event persists, influencing the subsequent data point to also be high (or low). Alternately, if there is a negative autocorrelation, a high data value could lead to a subsequent low value (and vice versa). Even if the data is Gaussian, it might not be independently distributed. Zero autocorrelation is a fundamental assumption in many methods, and the implementation needs to ensure that the method will be insensitive or immune to any autocorrelation in the data fluctuations.
12. Process signals might be noiseless during certain periods. Perhaps the reading is zero because the valve is off, or perhaps because that value is the override to prevent an execution error, or perhaps the value is repeatedly identical because the process changes are within the discrimination ability of the transmitted signal. The method needs to cope with the extreme of zero noise.

13. Patterns in the trend may confuse certain techniques and make a not-at-steady-state event appear to be at steady state. Consider an upward then downward trend centered in a window of analysis, fit to a linear trend line, and tested for a non-zero slope.
14. During a TS, the statistic needs to make large changes relative to its range of values at SS to clearly distinguish the two conditions. The statistic needs to be sensitive to the data pattern.
15. The method should be scale independent so that when an operator changes variable units or set points the method still works.

A practicable method needs to be computationally simple, robust to the vagaries of process events, easily implemented, and easily interpreted.

Applications of the R-statistic Filter Approach

The original article about the ratio of variances (R-statistic) filter technique presented in this tutorial was authored by Cao and Rhinehart (1995). Subsequently, Cao and Rhinehart (1997) and Shrowti, *et al.* (2010) presented critical values for Type-I and Type-II errors respectively.

The author has been involved in many pilot-scale and commercial scale applications using the R-statistic filter method. These include: Brown and Rhinehart (2000) that demonstrated a multivariable version of the technique on a pilot-scale distillation process. Katterhenry and Rhinehart (2001) then used it to automatically sequence experimental transitions on the distillation process, and Szela and Rhinehart (2003) demonstrated its application to automate transitions between experimental tests on a two-phase flow unit. Huang and Rhinehart (2013) report on an application in monitoring flow rate in a pilot-scale gas absorption unit, and Vennavelli and Resetarits (2013) report on its application in a commercial scale multivariable distillation unit.

Iyer and Rhinehart (2000) and Padmanabhan and Rhinehart (2005) used the approach to stop training a neural network and as the termination criterion for nonlinear regression. The author is presently using it as the convergence criterion in optimization of stochastic models. Rhinehart, *et al.* (2012), and Rhinehart (2014, 2016, and 2018) used SSID as the stopping criterion for a novel optimizer.

Personal communications from industry revealed its application to triggering reactor analysis and selecting steady and transient data for neural network modeling of a process.

Other users from the journal literature include: Bhat and Saraf (2004) who applied it to trigger data reconciliation on a commercial scale process unit. Jeison and van Lier (2006) used it to trigger on-line analysis of trans membrane pressure in a bioreactor. Salsbury and Singhal (2006) include the technique in a patented method for evaluating control system performance. Carlberg and Feord (1997) addressed the application in model adjustment of a commercial scale reactor. Mansour & Ellis (2007) included the technique within a reactor optimization. Kuehl and Horch (2005) used the technique to detect sluggish control loops. Zhang (2001) used the technique to segregate data sections for neural network models. And, Ye, *et al.* (2009) used it to trigger safety assessment in multimode processes.

Null Hypothesis, Statistics, Critical Values, and the Claim

There is a common structure in statistical analysis. A refresher of the elements will aid understanding of the methods and issues related to evaluating the method. First, is the Null Hypothesis, the statement of the assumed state. Usually it is that there is zero difference, that two treatments, or periods are equal. Here the Null Hypothesis is that the process is at steady state, that there is no difference in level (average value) in sequential data.

Next, in structuring a statistical test is the choice of a statistic that would have an expected value if the Null Hypothesis is true, and a distinguishable different value if it were not true. For example, if a process is at steady

state then the slope of the trend will have a value of zero. In this case, the statistic is the slope; and the expected value, if the Null Hypothesis is true, is zero. But, there can be a wide variety of statistics. Another could be the average value of the data in sequential windows. The literature to be reviewed will present statistics based on polynomial and wavelet representations of the time series, of ratios of variances, of differences between filtered variables, of patterns of runs in residuals, and others.

Whatever the statistic, it will not have a fixed (uniform or constant) value at steady state, because the process data is noisy, and the sample size is finite. Accordingly, even when the Null Hypothesis is true, the value of the statistic will fluctuate from sample-to-sample reflecting the vagaries of Nature. The statistic will have a distribution of values, a range of values bounding the ideal expected value, when the process is at SS.

If the value of the statistic has a small deviation from the expected value, a deviation that is within the normally expected range when at SS, then there is not sufficient evidence to claim the Null Hypothesis is not true, to reject the Null Hypothesis. However, if the process is not at SS, then the distribution of the statistic will have values form a different range. When the value of the statistic has an unusually large deviation, out from the normal range, then the likely explanation is that the Null Hypothesis is not true. Then the Null Hypothesis can be rejected.

The larger is the deviation from the expected value, the larger is the confidence that the Null Hypothesis can be rejected. Usually, statistical tests relating to an economic analysis are based on a 95% confidence. This means that when the Null hypothesis is actually true, the Null Hypothesis will be falsely rejected 5% of the time. However, for Statistical Process Control, the 99.9% level of confidence is used to temper the undesired effects of false alarms, rejecting the Null hypothesis when it is true. And, where safety is the issue, it is not uncommon to use the 99.99% confidence. The critical value represents the extreme value of the statistic that would have a 100%-95%=5% chance (or 100%-99.99%=0.01% chance) of happening if the Null Hypothesis were true. The level of significance, α, is the probability (on a 0 to 1 basis, not a 0 to 100% basis) of the extreme value. α=(100-confidence)/100.

If the value of the statistic exceeds the critical value, then the statistical tradition is to claim “Reject the Null Hypothesis”, otherwise the claim is “Cannot Reject the Null Hypothesis”, which is commonly stated as “Accept the Null Hypothesis”. In this SSID case, if the statistic value exceeds the critical value then the claim would be “Transient State” otherwise accept “Steady State”. However, in this tradition “Accepting the Null Hypothesis” when it cannot be rejected does not mean there is confidence that it is true. It really means “not enough evidence to confidently reject”. In the legal parallel, when there is not enough evidence to confidently convict, the jury says “not guilty”. This does not mean “innocent”. As another parallel, in play reviews in US football, the field referee’s call is “confirmed” if the review of video tapes confidently reveal it was the correct call, but if the video can neither support nor deny the field call, the referee’s call “stands”. “Stands” does not mean it was confidently correct.

Additionally, however, if the process is truly at SS, there is a chance that a rogue confluence of random events in the data could create a statistic value that rejects the SS hypothesis. The probability is α, the Type-I error, the probability of rejecting the Null Hypothesis when it is true. Alternately, if the process is in a slight transient state, nearly, but not quite at SS, then there is a chance that a statistic will have a numerical value within the normal range, and the SS hypothesis would not be rejected. This is a Type-II error, accepting the null hypothesis when it is false. To solve this, we will have an alternate threshold on the other side of the distribution. But in either case there remains a possibility that the T-I or T-II error could be made. Accordingly, the claim should not be a definite statement such as “The process is at SS”. The claims should be “The probable condition is SS” or “The probable condition is TS.” And, two statistic thresholds are called for not just one.

Approaches

Both the technical literature and on-line discussion groups reveal the many approaches that individuals are using for steady and transient state detection. Many methods have been devised. Here are some from the journal papers.

Vasyutynskyy (2005) sought to monitor control loops in building automation, and needed to identify settling time. The method was to split a data window in half, calculate the mean and variance in each half, then compute the statistic based on the ratio of the differences in averages scaled by the standard deviations. The process is claimed to be at SS when the ratio was equal to unity. By scaling the difference signal by the standard deviation the statistic is normalized, the signal is scaled by the noise.

Schladt and Hu (2007) applied SS ID to trigger data reconciliation using soft sensors (nonlinear steady state models). They cite the use of t-tests, Wald-Wolfowitz runs test, the Mann-Whitney test, the Mann-Kendall test, and the methods of Cao and Rhinehart (1995). They chose to use the t-test of difference between means of sequential data windows. Steady State is indicated when the statistic value is nearly zero. This method scales the difference in level by the noise standard deviation.

Jiang, *et al.* (2003) proposed a wavelet based method of multi-time-scale process trend analysis. The process trends are modeled with wavelets, and a wavelet transform modulus index which ranges between 0 and 1 is used to trigger SS. Steady State is indicated when the statistic value is nearly zero. This method does not scale the signal by the noise level.

Roux, *et al.* (2008) compare the technical performance of a Savitzky-Golay filter (polynomial interpolation), Cao and Rhinehart (1995), a non-parametric reverse arrangements test, and a von Neumann Rank test on both simulated and real plant data. In the polynomial interpolation method, the data in a time series window of length 'n' is fit with a polynomial of order less than 'n', and the derivative of the polynomial at the window center is the statistic. Steady State is indicated when the statistic value is nearly zero. This method does not scale the signal by the noise level.

Castiglioni, *et al.* (2004) applied steady state analysis to beat-by-beat values of blood pressure and heart rate in investigating incremental exercise tests. They used a runs test to compare the runs (a series of measurements on the same side of the mean). The statistic is the number of runs in a time window. They empirically derived critical values for the statistic from historical data. When the statistic is lower than the critical value they reject the SS hypothesis. This method does not scale the signal by its variability, and assumes all individuals have the same variation at SS.

Kim, *et al.* (2008) applied SS detection to support fault detection analysis of a residential air conditioner. They calculated the data variance over a moving window. If at SS the variance of a noiseless signal should be zero. They reject the SS Hypothesis if the variance is large. This method does not scale the signal by the noise level.

Yao, *et al.* (2009) used a Mahalanobis distance as the statistic for multi-variable batch analysis. If the statistic is large reject SS. The statistic is scaled by the data variability.

Barolo, *et al.* (2003) adjusted the Murphree Tray Efficiency of SS distillation models with data from a process, and accepted SS when the overall material and energy balances closed. The statistic would be the deviation in the balances, and should have a value of zero. They selected a threshold of the deviation value as a critical value. Their approach does not scale the balance closure by its variability.

Flehmig and Marquardt (2006) propose a polynomial approach to identify trends in multivariable processes. If the measured trend error in the recent data window is small the process is at SS. The statistic is scaled by the data variability.

Crowe, *et al.* (1955) propose an approach that calculates the variance on a data set by two approaches – the mean-square deviation from the average and the mean-square deviation between successive data. The ratio of variances is an F-like statistic, and assuming no autocorrelation in the data, it has an expected value of unity when the process is at SS. This approach scales the statistic by the inherent data variability.

There are alternate methods for steady state identification, and the research and practice communities reveal separate methods. The published literature substantially contains techniques being investigated by the research and academic communities. To assess state of the art in the practice community, I have reached out to industry groups in two occasions. First was an "Ask the Experts" column in "CONTROL for the process industries" magazine, and second was a discussion within the LinkedIn Group discussion for the Automation & Control Engineering Group.

Loar (1994) described the use of a moving average chart and the application of statistical process control to detect a violation of the 3-sigma limits to trigger the not-at-SS recognition. This method scales the threshold by the data variability.

Alekman (1994) reported the use of the t-statistic to test for the difference between data means in successive windows. This scales the statistic by the variance, but as mentioned earlier, can claim SS if an oscillation is centered between the two windows.

Jubien and Bihary (1994) calculate the standard deviation of data in a recent window and compare that value to a threshold expected value from a nominal SS period. If the standard deviation excessively exceeds the expected threshold then the process is not at SS. The authors appropriately note that success with this method relies on the ability to determine which are the essential variables that need to be observed, the window length, and the threshold standard deviation for each variable which would need to change with its inherent variability.

Svensson (2012) acknowledged the use of two methods. One, similar to that of Jubian and Bihary is to compare the standard deviation in a moving window to a standard value. The other is to calculate the residual from steady-state mass and energy balances, and flag not-at-SS when the residual is large. Neither accounts for changes in the data inherent variability.

Nachtwey (2012) suggested either a Savitsky-Golay smoother/filter to find the mid-window derivative value of a polynomial; or an Alpha-Beta-Gamma filter, similar to, but simpler than a Kalman filter, to estimate the rate of change of a variable. Then compare the derivatives against limits. He reported on the application of the ABG filter approach determine if an actuator/motor is stopped. In either case the threshold rate needs to be determined and scaled by data inherent variability.

Stanley (2012) also suggested the Savitsky-Golay filter, and yet another simpler approach labeled MACD (for Moving Average Convergence Divergence): Use two first-order filters with differing time-constants to filter the data. During a transient the filtered values approach the data at different rates, but at SS they will both agree on the average value. So, at SS the difference between the filtered values is ideally zero, but it will have some fluctuations about zero due to data noise. When the deviation is large reject the SS hypothesis. The deviation should be scaled by the data inherent variability. The MACD analysis is commonly used in automating detection of trends in stock prices.

In the on-line discussions, Stanley (2012) also identified associated issues with an application of any method: A) Anti-aliasing to remove the influence of high frequency noise. B) Accepting low amplitude oscillations (perhaps resulting from a control system) that are inconsequential when averaged out. C) Appropriate choice of the variables to be observed. D) Falsely claiming a process is at SS when the sensor failed and the reported measurement is at SS. E) Falsely claiming the process is not at SS when the change in measurement was the result of a sensor recalibration or fault. These points reveal the need for an educated interpreter of the identifier.

He offered a solution to the false indications related to a sensor failure. Observe redundant variables, and in a voting scheme report SS or TS when a large subset are in agreement. Then, if the process is at SS one failing sensor would not lead to a not at SS claim.

An observation I make is that the methods used by practitioners are often based on standard, well-known concepts (differences between averages, SPC techniques) or are often computationally simple (differences

between two first-order filters). This observation supports the author's bias that for something to gain industrial acceptance, it needs to be simple in both concept and implementation.

A second observation is that many of these approaches have a significant computational burden with the storage and processing of the data, making their application within conventional data acquisition and control hardware problematic. For a technique to be practicable, it must be both simple and effective.

Third, many of the approaches reported in the literature are not scaled by the inherent variability of the data. Here are a few examples of how process data changes variability: Fluctuations in flow rate data depend on the flow-induced turbulence, the intensity of which changes with flow rate. Variability in pH data depends on the titration curve which defines the sensitivity of pH to acid-base mixing imperfections. It changes with compositon. Variability in pressure drop data in 2-phase situations (boiling, distillation, absorption/stripping, pipelines) depends on the flow rates of the two components. Although using threshold values for a statistic is an effective approach, in the author's process experience, the noise amplitude on process data often changes with operating conditions, and the threshold needs to change with data variability.

Fourth, common issue of all of these approaches is that each rejects the Null Hypothesis. But, not rejecting SS does not necessarily imply that a process is at SS. A "Not Guilty" verdict does not mean the accused is innocent, it simply means that there was not enough evidence to confidently claim "Guilty".

Finally, If one method always worked perfectly, then there would not be the plethora of methods. I am not aware of a perfect approach.

The probable SS and probable TS identification of the filter approach (Cao and Rhinehart, 1995) is executed using an easily implemented statistical method, which is scaled by inherent variability of the data, and has defined critical values for both probable SS and probable TS. It was extended to multivariable process and demonstrated on lab-scale and pilot-scale processes to automatically trigger an experimental sequence. It was also demonstrated as a convergence criterion in nonlinear regression optimization. It does not require specific tuning for each application – the coefficient values are universal. It is relatively insensitive to the distribution of the noise. Those are good points. But, it fails when there is a cyclic background trend, or on noiseless periods, and can miss small changes relative to the noise amplitude. Nothing seems perfect. But often the simplicity and effectiveness of the filter method fit the application need.

II – METHOD – FILTER

This section explains one method, the Filter version of the R-statistic approach. It also uses that method to discuss issues related to data discretization, autocorrelation, T-I and T-II errors, and the like that are relevant to all techniques.

Conceptual model

Begin with this conceptual model of the phenomena: The true process variable (PV) is at a constant value (at SS) and fluctuations on the measurement and signal transmission process create uncorrelated "noise", independently distributed fluctuations on the measurement. Such random measurement perturbations could be attributed to mechanical vibration, stray electromagnetic interference in signal transmission, thermal electronic noise, flow turbulence, etc. Alternately, the SS-PV-with-noise concept includes a situation in which the true PV is averaging at a constant value, and internal spatial nonuniformity (resulting from nonideal fluid mixing or multiphase mixtures in a boiling situation), or an internal distribution of properties (crystal size, molecular weight) create temporal changes to the local measurement.

If the noise distribution and variance would be uniform in time, then statistics would call this time series as stationary. However, for a process, the true value, nominal value, or average may be constant in time, but the noise distribution may change. So, SS does not necessarily mean stationary in a statistical sense of the term.

The first Null Hypothesis of this analysis is the conventional Null Hypothesis that the process is at steady state, Ho: SS. For the approach recommended here, the statistic, a ratio of variances, the R-statistic, will be developed. Due to the vagaries of noise, the R-statistic will have a distribution of values at SS. As long as the R-statistic value is within the normal range, the null hypothesis cannot be rejected. When the R-statistic has an extreme value, then the null hypothesis can be rejected with a certain level of confidence, and probable TS claimed.

By contrast, there is no single conceptual model of a transient state. A transient condition could be due to a ramp change in the true value, or an oscillation, or a first-order transient to a new value, or a step change, etc. Each is a unique type of transient. Further, each single transient event type has unique characteristics such as ramp rate, cycle amplitude and frequency, and time-constant and magnitude of change. Further, a transient could be comprised of any combination or sequence of the not-at-SS events. Since there is no unique model for TS, there can be no Null Hypothesis, or corresponding unique statistic that can be used to reject the TS hypothesis and claim probable SS. Accordingly, an alternate approach needs to be used to claim probable SS.

The alternate approach used here is to take a transient condition which is barely detectable or decidedly inconsequential (per human judgment) and set the probably SS threshold for the R-statistic as an improbable low value, but not so low as to be improbably encountered when the process is truly at SS.

Method Equations

This method uses the R-statistic, a ratio of two variances, as measured on the same set of data by two methods – deviations from the average, and successive data deviations. It was first published in 1995 by Cao and Rhinehart. I discovered in retrospect, that a similar ratio of variances method was presented by von Neumann, et al. (1941) and Crowe, et al. (1955). But, in this filter approach, variances are incrementally updated in a computationally more efficient manner.

Figure (1) illustrates the concept (Huang, 2013). The dotted line represents the true trend of a process. The value starts at 15, ramps to a value of 10 at a time of 50, and then holds steady. The diamond markers about that trend represent the measured data. The dashed line, the true trend, is unknowable, only the measurements can be known and they are infected with noise-like fluctuations, masking the truth.

The method first calculates a filtered value of the process measurements, indicated by the curved line that lags behind the data. Then the variance in the data is measured by two methods. The deviation indicated by d_2 in the upper left of the figure is the difference between measurement and the filtered trend. The deviation indicated by d_1 in the lower right is the difference between sequential data measurements.

If the process is at SS, as illustrated in the 80 to 100 time period, X_f is almost the middle of the data. Then a process variance, σ^2, estimated by d_2 will ideally be equal to the value of σ^2 as estimated by d_1. Then the ratio of the variances, $r = \frac{\sigma^2{}_{d_2}}{\sigma^2{}_{d_1}}$ will be approximately equal to unity, $r = \frac{\sigma^2{}_{d_2}}{\sigma^2{}_{d_1}} \cong 1$. Alternately, if the process is in a TS, such as in the 20-40 time period on the figure, then X_f is not the middle of data, the filtered value lags behind, and the variance as measured by d_2 will be much larger than the variance as estimated by d_1, $\sigma^2{}_{d_2} \gg \sigma^2{}_{d_1}$, and ratio will be much greater than unity, $r = \frac{\sigma^2{}_{d_2}}{\sigma^2{}_{d_1}} >> 1$.

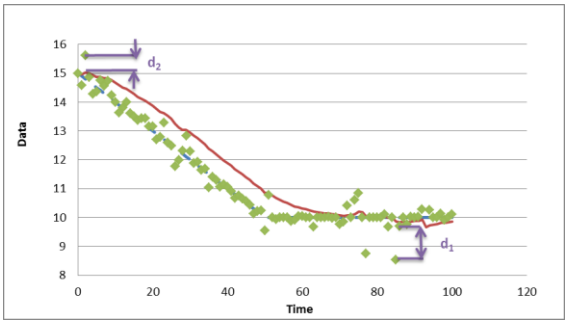

Figure 1: Illustration actual process (dashed line), noisy measurements (diamond markers), filtered data (solid line), and deviations

To minimize computational burden, in this method a filtered value (not an average) provides an estimate of the data mean:

$$X_{f,i} = \lambda_1 X_i + (1-\lambda_1) X_{f,i-1} \qquad (1)$$

X = the process variable
X_f = Filtered value of X
λ_1 = Filter factor
i = Time sampling index

The first method to obtain a measure of the variance uses an exponentially weighted moving "variance" (another first-order filter) based on the difference between the data and the filtered value, representing the average:

$$\upsilon^2_{f,i} = \lambda_2 (X_i - X_{f,i-1})^2 + (1-\lambda_2)\upsilon^2_{f,i-1} \qquad (2)$$

$\upsilon^2_{f,i}$ = Filtered value of a measure of variance based on differences between data and filtered values
$\upsilon^2_{f,i-1}$ = Previous filtered value

Equation (2) is a measure of the variance to be used in the numerator or the ratio statistic. The previous value of the filtered measurement is used instead of the most recently updated value to prevent autocorrelation from biasing the variance estimate, $v^2_{f,i}$, keeping the equation for the ratio relatively simple. Equation (2) does not provide the variance, even at SS, because using the filtered X_f, rather than the true average, adds a bit of variability to the difference $(X_i - X_{f,i-1})$.

The second method to obtain a measure of variance is an exponentially weighted moving "variance" (another filter) based on sequential data differences:

$$\delta^2_{f,i} = \lambda_3 (X_i - X_{i-1})^2 + (1-\lambda_3)\delta^2_{f,i-1} \qquad (3)$$

$\delta^2_{f,i}$ = Filtered value of a measure of variance
$\delta^2_{f,i-1}$ = Previous filtered value

This will be the denominator measure of the variance. It, also, is not the variance, but is effectively twice the variance when the process is at SS.

The ratio of variances, the R-statistic, may now be computed by the following simple equation:

$$R = \frac{(2-\lambda_1) v^2_{f,i}}{\delta^2_{f,i}} \qquad (4)$$

The calculated value is to be compared to its critical values to determine SS or TS. Neither Equations (2) nor (3) compute the variance. They compute a measure of the variance. Accordingly, the $(2-\lambda_1)$ coefficient in Equation (4) is required to scale the ratio, to represent the classic variance ratio. Complete executable code, including

initializations, is presented in VBA in the software on the site www.r3eda.com. The essential assignment statements for Equations (1) to (4) are:

```
nu2f = l2 * (measurement - xf) ^ 2 + cl2 * nu2f

xf = l1 * measurement + cl1 * xf

delta2f = l3 * (measurement - measurement_old) ^ 2 +  cl3 * delta2f

measurement_old = measurement

R_Filter = (2 - l1) * nu2f / delta2f
```

The coefficients l1, l2, and l3 represent the filter lambda values, and the coefficients cl1, cl2, and cl3 represent the complementary values.

The five computational lines of code of this method require direct, no-logic, low storage, and low computational operation calculations. In total there are four variables and seven coefficients to be stored, ten multiplication or divisions, five additions, and two logical comparisons per observed variable.

I initialize filtered values with 0, which is more convenient than initializing them with a more representative value from recent past data. This leads to an initial not-at-SS value of the R-statistic, but after about 35 samples the initial wrong values are incrementally updated with representative values.

Being a ratio of variances the statistic is scaled by the inherent noise level in the data. It is also independent of the dimensions chosen for the variable.

If the process is at steady state, then the R-statistic will have a distribution of values near unity. Alternately, if the process is not at steady state, then the filtered value will lag behind the data, making the numerator term larger than the denominator, and the ratio will be larger than unity.

Critical values for the R-statistic, based on the process being at steady state with independent and identically distributed variation (white noise), were also developed by Cao and Rhinehart (1997), who suggest that filter values of λ_1=0.2 and λ_2=λ_3=0.1 produce the best balance of Type I and Type II errors. An R-critical value is selected and determined by the level of significance, α, alternately the confidence level, [100(1-α)], that we want to achieve. The null hypothesis is that the process is at steady state. If the computed R-statistic is greater than R-critical, then we are 100(1-α) percent confident that the process is not at steady state. Consequently, a value of R less than or equal to R-critical means the process may be at steady state. Often we assign values of either "0" or "1" to a variable, SS, which represents the state of the process. If R-calculated > R-critical1 ~2.5, "reject" steady state with 100(1-α) confidence, assign SS=0. Alternately, if R-calculated< R-critical2 ~.9, "accept" that the process may be at steady-state, and assign SS=1. If in-between values happen for R-calculated, hold the prior 0 or 1 (reject or accept) state, because there is no confidence in changing the most recent declaration.

The method presumes no autocorrelation in the time series of the process measurement data at SS. This is ensured by selection of the sampling time interval, which may be longer than the control interval. Running in real time, the identifier does not have to sample at the same rate as the controller.

The filter factors in Equations (1), (2) and (3) can be related to the number of data (the length of the time window) effectively influencing the average or variance calculation. Ideally the effective number of data in the window N=1/λ. If λ=0.1 then effectively the method is observing $N = 10$ most recent data points. However, based on a first-order decay, long past data retain some influence, and roughly, the number of data effectively influencing the window of observation is about 3.5/λ. If λ=0.1 then effectively the method is observing $N \approx 35$ data points. However, not all data points have equal weighting. The filter is a first-order decay, an exponentially declining weighting N past data. Truly, the long past data retain some influence, but the collective fractional influence is 1-(1- λ)^N. With λ=0.1 and N=35 the old data only have a 0.025 fractional influence.

Larger λ values mean that fewer data are involved in the analysis, which has a benefit of reducing the time for the identifier to catch up to a process change, reducing the average run length (ARL). But, larger λ values has an undesired impact of increasing the variability on the statistic, confounding interpretation. The reverse is true: Lower λ values undesirably increase the ARL to detection, but increase precision (minimizing statistical errors).

Originally, Cao and Rhinehart recommended values were λ_1=0.2, λ_2= λ_3=0.1, effectively meaning that the most recent 35 data points are used to calculate the R-statistic value. Since then, I usually recommend λ_1= λ_2= λ_3=0.1 for convenience and effectiveness However, sometimes I use λ_1= λ_2= λ_3=0.05 to have less uncertainty. These are not a critical choice.

Type-I Error

If the null hypothesis is that a process is at SS, then a Type-I error is a claim of not-at-SS when the process is actually at SS. The concept is best understood by considering the distribution of the R-statistic of a SS process. The statistical vagaries in the data create a distribution of the R-statistic values. Figure 2 represents the statistical distribution of the R-statistic values at steady state. Note that the distribution is not the symmetric, bell-shaped, normal distribution. It is a Chi-Square type of distribution, skewed, and with no values below zero.

The R-statistic will have some variability because of the random fluctuations in the sequential measured data. If the value of R is less than the upper 95% confidence value, the process may be at steady state, but if it is beyond (larger than) the 95% confidence value then it is likely that the process is not at steady state. If the process is at steady state, there is a small, α=5%, chance that R > $R_{0.95}$. The level of significance, α, is the probability of making a Type-I error, of rejecting the SS hypothesis when it is true.

Five-percent is the standard level of significance for economic decisions. However, if α=5%, and the process is at SS, the identifier will make a false claim 5% of the time, about every 20th sample, which would render it nearly useless for automation. Accordingly, I use an α=0.5%, $R_{0.995}$, threshold to trigger a TS claim.

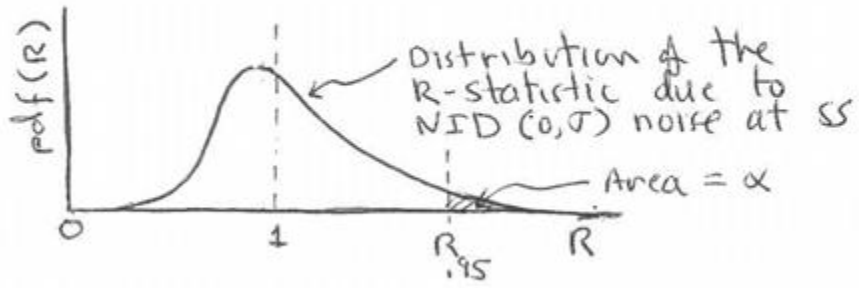

Figure 2: R-statistic distribution at steady state

Figure 3 includes the distribution of the R-statistic for a process that is not at steady state, one that is in a transient state, with its distribution of R-statistic values greater that unity. For a process that is not at steady state, there is a high chance that R> $R_{0.95}$. As illustrated in by the shaded area in Figure 3 it is about a 70% chance.

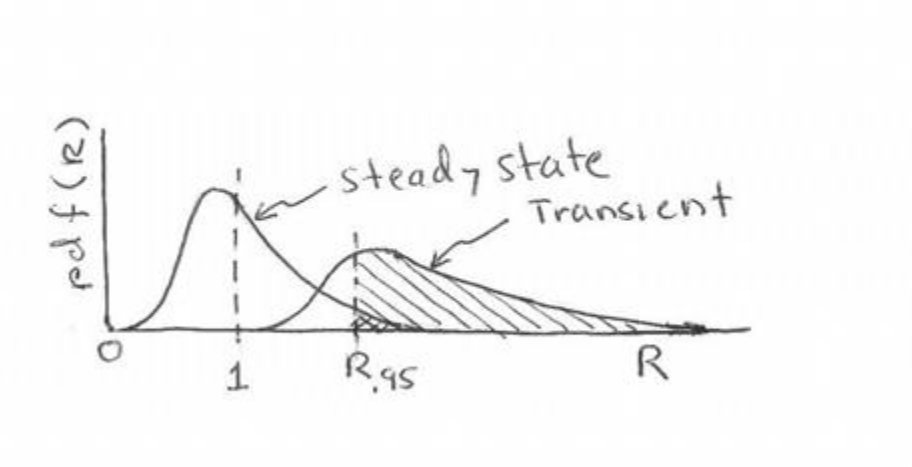

Figure 3: High chance of not being at steady state

So, if R> $R_{0.95}$ the likely explanation is that the process is in a transient state. As illustrated by the shaded areas to the right of the $R_{0.95}$ value, the probability that an excessive R-value could come from the SS or the TS distribution, the odds are about 70% to 5% or 15:1 for being in the transient state. Claim TS.

There are several ways to reduce the probability of a T-I error. An obvious one is to choose a smaller value for alpha, and in statistical process control (SPC) the value of α=.27% is normally accepted. But too small a level of significance means that a not-at-SS event might be missed – a T-II error.

Type-II Error

If the null hypothesis is that the process is at SS, then a Type-II error is claiming the process is at SS when it is actually in a TS. If R< $R_{0.95}$ this does not indicate the process is at steady state. Figure 4 provides an illustration using the same two SS and TS distributions. As illustrated, the likelihood of R< $R_{0.95}$ if at steady state is 95%, and if not at steady state it is 60%. Here the odds are about 1.6:1 that the steady state conclusion is true.

The 1.6:1 odds are not very definitive. So, to be confident that the process is at steady state, consider the left side of the distributions, and an alternate T-I error. My appreciation to Dennis Williams, Consulting Engineer, LyondellBasell, Houston, TX, for correcting my nomenclature in this section.

Alternate Type-I Error

For a given transient state, $R_{\beta,TS}$ is the lower β critical value and for a given steady state, $R_{1-\alpha,SS}$ is the 1-α upper critical value. If R > $R_{1-\alpha,SS}$, then it will reject SS (accept TS). And if R < $R_{1-\alpha,SS}$, it will reject TS (accept SS).

If the process is at a transient state then β is the probability of R< $R_{\beta,TS}$. Figure 4 shows that the TS process has only about a 1% chance that R< $R_{\beta,TS}$. However, if the process is at steady state then as illustrated in Figure 4 there is a 25% likelihood that R< $R_{\beta,TS}$. So, if R< $R_{1-\alpha,SS}$ the odds are about 25:1 that process is at steady state. Claim SS.

However, if $R_{\beta,TS} \le R \le R_{1-\alpha,SS}$ then there is a high likelihood of the process being either at steady or transient state. There is no adequate justification to make either claim. So, retain the last claim.

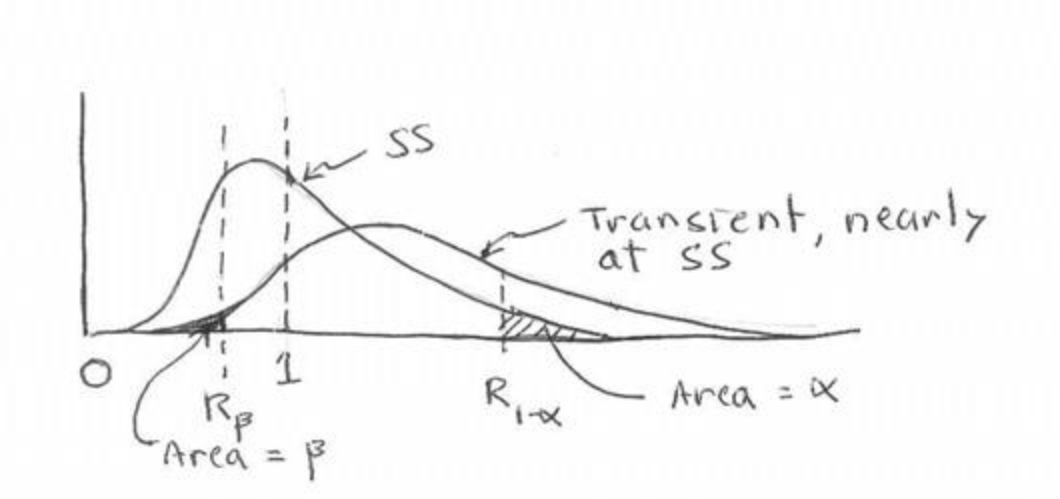

Figure 4: Critical value for steady state identification

Both T-I and Alternate T-I errors are important. A Type-I error is the trigger of a "not at steady state" claim when the process is at steady state. An Alternate Type-I error is the trigger of an "at steady state" claim when the process is in a transient state. In any statistical test, the user needs to choose the level of significance, α, the probability of a T-I error, and β, the probability of an Alternate T-I error. Once decided, the $R_{1-\alpha,SS}$ critical value can be obtained from Cao and Rhinehart (1997), and the $R_{\beta,TS}$ critical value from Shrowti, *et al.* (2010).

However, it is more convenient and less dependent on idealizations to visually select data from periods that represent a transient or steady period, and to find the R-critical values that make the algorithm agree with the user interpretation. My experience recommends $R_{1-\alpha,SS}$=3 to 4 and $R_{\beta,TS}$=0.85 to 0.9, chosen by visual inspection as definitive demarcations for a transient and steady process.

Illustration

Figure 5 illustrates the method. The process variable, PV, is connected to the left hand vertical axis (log10-scale) and is graphed with respect to sample interval. Initially it is at a steady state with a value of about 5. At a sample number 200, the PV begins a first-order rise to a value of about 36. At sample number 700, the PV makes a step rise to a value of about 40. The ratio-statistic is attached to the same left-hand axis and shows an initial kick to a high value as variables are initialized, then relaxes to a value that wanders about the unity SS value. When the PV changes at sample 200, the R-statistic value jumps up to values ranging between 4 and 11, which relaxes back to the unity value as the trend hits a steady value at a time of 500. Then when the small PV step occurs at sample 700, the R-value jumps up to about 4, then decays back to its nominal unity range. The SS value is connected to the right-hand vertical axis and has values of either 0 to 1 that change when the R-value exceeds the two limits, of $R_{\beta,TS}$ and $R_{1-\alpha,SS}$.

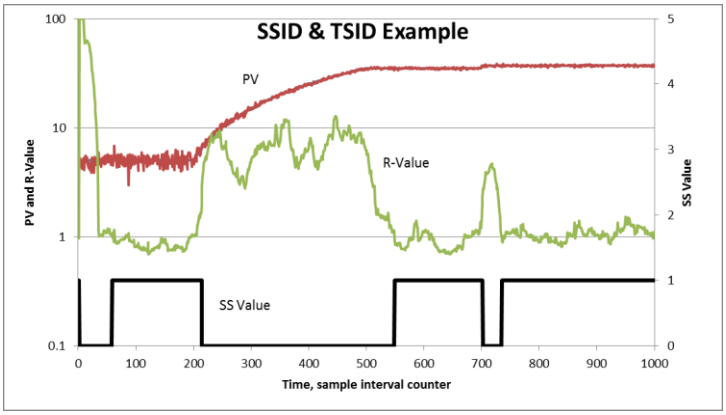

Figure 5: Example

III - Discussion of Important Attributes

Distribution Separation

A good statistic will provide a large separation of the SS and TS distributions relative to the range of the distribution. The distribution shift is the signal that results of the process shift from SS to TS, and the width of the distribution is the uncertainty, or noise associated with the statistic. A good method to detect SS and TS will have a large signal-to-noise aspect, a large shift in the not-at-SS distribution relative to the width of the distribution.

Figure 5 reveals the issue. In the several SS periods (sample 100-200, 600-700, and 800-1000) the R-statistic has values in the 0.8 to 2 value. In the TS instances (sample 200-500, and 700) the R-statistic has values in the 5-10 range, which are definitely different from the values in the SS periods. There is good separation in the two ranges of values.

In this ratio-of-variance method, the shift in the statistic is proportional to the square of the deviation. By contrast, in a method that observes the difference between two means, the shift in the statistic is linearly proportional to the deviation.

Lower filter lambda values, for instance $\lambda = 0.05$, make the variation in the R-statistic smaller, meaning that the separation between SS and TS R-values is more definitive. However, larger lambda values, for instance $\lambda = 0.2$, make time-to-detection, average run length (ARL) faster. Again, refer to Figure 5. Note that after the step change in the PV at sample 700, the PV holds at its new value, but the method did not return to a SS claim until about sample 740.

Average Run Length

Another aspect to be considered as a quality performance metric of a statistic is average run length (ARL), the number of samples after an event occurs to be confident in declaring a change. In the moving window methods with N data being considered, the last not-at-SS data event must be out of the window for the analysis of the window to be able to declare "at SS". This would appear to be N data, or the average run length of ARL=N. However, when the process is at SS, the statistic will not always be at the extreme value. There is a probability, β, that it is beyond the extreme value. When at SS, if there is no autocorrelation in the R-statistic, the number of data required to randomly have a value that is less than the β-probable extreme is 1/β. Then the average run length would be the number of samples to clear the window plus the expected number to likely create an extreme value. So, simplistically, ARL = N + 1/β. See the Section "Multivariable ARL" for a more rigorous derivation and formula.

The filter factors in Equations (1-3) can be related to the number of data (the length of the time window) in the average or variance calculation. Roughly, the number of data with a residual influence on the statistic is about 3/λ to 5/λ, depending on your choice of what constitutes an inconsequential residual influence of past data. To determine an average run length, first expand the filter mechanism to reveal the exponentially weighted moving average form:

$$X_{f_i} = \lambda X_i + (1-\lambda)X_{f_{i-1}}$$
$$= \lambda X_i + (1-\lambda)\left[\lambda X_{f_{i-1}} + (1-\lambda)X_{f_{i-2}}\right]$$
$$= \lambda X_i + (1-\lambda)\{\lambda X_{f_{i-1}} + (1-\lambda)[\lambda X_{f_{i-2}} + (1-\lambda)X_{f_{i-3}}]\}$$
$$= \lambda X_i + (1-\lambda)\lambda X_{f_{i-1}} + (1-\lambda)^2\lambda X_{i-2} + \cdots + (1-\lambda)^N\lambda X_{i-N} + (1-\lambda)^{N+1}\lambda X_{i-(N+1)}$$

Now it is possible to determine the value of N that makes the persisting influence of the old $X_{i-(N+1)}$ trivial, for the event to clear from the statistic. If at SS for N samplings then $X_i \cong X_{i-1} \cong X_{i-2} \cdots$ and $X_{f_i} \sim X_i$. Consider the value of N that makes

$$(1-\lambda)^{N+1}X_{f_{old}} \ll X_{SS}$$

As an estimate assume " ≪ " means 2%, and perhaps $X_{f_{old}} \sim 5X_{SS}$. If $\lambda = 0.1$, then the condition is

$$(1-0.1)^{N+1}(5X_{SS}) = 0.02X_{SS}$$

Rearranging to solve for N,

$$N = \frac{\mathrm{Ln}(\frac{0.02}{5})}{\mathrm{Ln}(1-0.1)} - 1 = \frac{\mathrm{Ln}(0.004)}{\mathrm{Ln}(0.9)} - 1 \approx 50.$$

Assuming "<<" means 5%, then $N = \frac{\mathrm{Ln}(\frac{0.05}{5})}{\mathrm{Ln}(1-0.1)} - 1 \approx 43$. Assuming "<<" means 5% and $X_{f_{old}} \sim 3X_{SS}$, $N = \frac{\mathrm{Ln}(\frac{0.05}{3})}{\mathrm{Ln}(1-0.1)} - 1 \approx 38$.

So, N for the influence to be gone, to permit X_f, ν_f^2 , and δ_f^2 to relax from an upset to be close to their SS values is between 38 and 50 depending on magnitude of event and what decay fraction makes it inconsequential.

Recommended values by Cao and Rhinehart (1995) are λ_1= 0.2 and λ_2= λ_3=0.1, effectively meaning that the most recent 38 to 50 data points are used to calculate the R-statistic value. Effectively, N≈4.5/λ. However, alternate lambda values have been recommended to ensure greater confidence in the SS claim, or faster identification.

The ARL for a transition from SS to TS is just a few samples, which is not a problem. However, the transition from TS to SS requires the filter statistics to relax, which takes time and results in an ARL ≈ 4.5/λ + .7/β.

Balancing ARL, α, and β

Larger λ values mean that fewer data are involved in the analysis, which has a benefit of reducing the time for the identifier to catch up to a process change (average run length, ARL) but has a undesired impact of increasing the variability on the statistic, broadening the distribution and confounding interpretation. Lower λ values undesirably increase the average run length to detection, but increase precision (minimizing Type-I and Alternate Type-I statistical errors) by reducing the variability of the distributions increasing the signal-to-noise separation of a TS to SS situation.

This work recommends filter values of λ_1= λ_2= λ_3=0.1, balancing precision with ARL and convenience of use. At these choices, the author observes that ARL for automatic detection roughly corresponds to the time an operator might confidently declare that the process has achieved a SS.

However, rather than choose critical values for $R_{1-\alpha,SS}$ and $R_{\beta,TS}$ from ideal theoretical distributions and Equation (4), it is more convenient and less dependent on idealizations to visually select data from periods that represent a transient or steady period, and to find the R-critical values that make the algorithm agree with the user interpretation. This work uses $R_{1-\alpha,SS}$= 3 to 4 and $R_{\beta,TS}$=0.9 to 1, chosen by visual inspection as definitive demarcations for a transient and steady process.

Distribution Robustness

Cao and Rhinehart (1995, 1997) and Shrowti, *et al.* (2010) investigated the robustness of the R-statistic distributions to a variety of noise distributions in the process variable. They found that the R-statistic distributions and the critical values are relatively insensitive to the noise distribution. This finding is not unexpected recognizing the well-known robustness of the F- and Chi-Square statistics (also based on a ratio of variances) for distributions with modest deviations from normal.

IV – Implementation Issues

This section discusses issues that are common to any SS&TSID technique.

Autocorrelation

The basis for the R-statistic method presumes that there is no autocorrelation in the time-series process data when at SS. Autocorrelation means that if a measurement is high (or low) the subsequent measurement will be related to it. For example, if a real process event causes a temperature measurement to be a bit high, and the event has persistence, then the next measurement will also be influenced by the persisting event and will also be a bit high. Autocorrelation could be related to control action, thermal inertia, data filters in sensors, signal discretization, etc. Autocorrelation would make the denominator estimate of the variance smaller, which would tend to make all R-statistic distributions shift to the right, requiring a reinterpretation of critical values for each process variable.

As common in Statistical Process Control techniques, it is more convenient to choose a sampling interval that eliminates autocorrelation, than to model and compensate for autocorrelation in the test statistic. This can be done if the source of the autocorrelation is from random perturbations. Although they have some persistence, the influence fades, and new random perturbation values cause measurement fluctuation. This short-lived persistence from random perturbations is different from the regular cycling from a feedback mechanism, or the flat-line constant value that arises when a signal is having a coarse discretization.

A plot of the current process measurement versus the previous sampling of the process measurement over a sufficiently long period of time (equaling several time-constants) at steady state is required to establish the presence/absence of autocorrelation. To detect autocorrelation, visually choose a segment of data that is at steady state, and plot the PV value w.r.t. its prior value.

Figure 6 plots data with a lag of 1 sample (a measurement w.r.t. the prior measurement) and shows autocorrelation. [In statistics jargon, lag means delay. In system dynamics and control jargon, lag means a time-constant. Here, I'm using the statistics jargon.] Figure 7 plots the same data but with a lag of 5 samples, and shows zero autocorrelation.

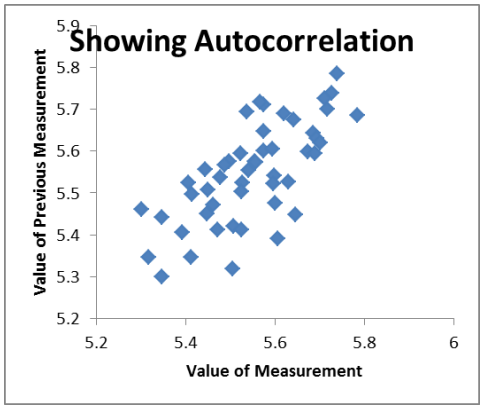

Figure 6: Data showing autocorrelation

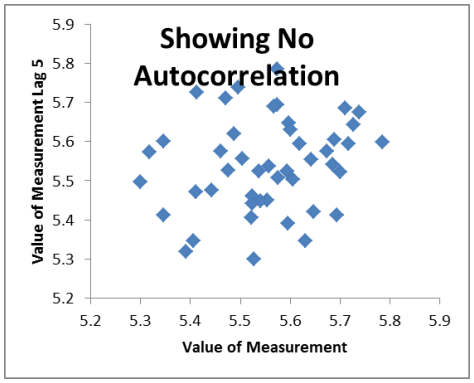

Figure 7: Data showing no autocorrelation when the interval is 5 samplings

What this means is that the SS or TS monitor should be only looking at every 5th data value from the process data stream. If sampling for data acquisition and control is once per second, then only run the SSID on a 5 second interval.

Signal Discretization

There is a smallest increment that can be displayed or reported. In a digital image it is pixel size. On a digital clock it is a 1-second interval, and on a stop watch perhaps a 1/100th of a second. On a digital speedometer display, 1 mph, and on a digital temperature display 1 F or 0.5 C. As time, speed, temperature, etc. progresses, the displayed value remains unchanged until it crosses over the discretization threshold to the next interval. If a data transmission system is a 10-bit system, then the binary number can have 2^{10} = 1024 values. If the calibration uses the upper 80 % of values (paralleling a standard 4-20 mA or 3-15 psig analog signal), then the PV range is matched with about 820 binary counts. There are only 820 possible values, with a discretization interval of about 0.12% of full scale.

Discretization is visible as a signal that repeatedly jumps between only a few values.

Because many process measurements are nonlinear, such as orifice flow measurements based on orifice pressure drop, the discretization may not be visible in one data range, but clearly distinguished in another.

Discretization confounds an estimate of data variance, or noise amplitude. If, for instance a signal was at SS, with a value of 53.5 and a noise sigma of 0.1, then the noisy measurement will be within 53.5 +/- 3 sigma or about 53.2

to 53.8. But, if the digital discrimination interval was 1, then all measurements would hold at the 53 value, and an estimate of the noise amplitude would, erroneously, be zero, not 0.1.

Aberrational Autocorrelation

If the autocorrelation in the PV is due to sustained oscillations from a feedback mechanism, then this method of delaying the sample will only work if the delay (lag) matches the cycle time of the oscillation. Additionally, if the autocorrelation is due to signal discretization that reports quantized values (the PV holds one value until the signal has changed enough to jump to a new level, and the quantized values are repeated, in between values are not permissible), then spacing out sampling does not eliminate autocorrelation.

Here are seven options that I've explored:

1. Re-tune the controller to stop the oscillations. If the process is nonlinear of makes large throughput changes, a controller that is properly tuned in one operating region, might be too aggressive in another. Perhaps "gain schedule" the controller coefficient values with operational region.
2. Re-calibrate the instruments or install alternate A/D devices to avoid the flatlining due to signal discretization.

My favored approaches are 1 and 2 – eliminate the problem. Alternately mask it:

3. Add simulated noise to a process variable to mask autocorrelation that cannot be removed by extending signal sampling interval. Either uniform or Gaussian distributed perturbations have worked well for both cyclic autocorrelation (such as that from a feedback loop) and in the case of discrimination error (where a variable periodically flat lines, because the instrument system discretizes the signal). The issues with this approach are that the addition of noise to the signal can make the method miss smallish changes in signal level that it would otherwise find. Too great a noise amplitude, and it will think the process is always at SS. So, you want to add just enough noise amplitude to mask the autocorrelation, but not so much that it hides changes. Second, you need to determine the amplitude of the noise for each application. I am not sure that such testing and determination of a right noise value, and the addition of the noise algorithm, is any better than the following ideas
4. Modifying the R-statistic thresholds to match the data. Choose periods that you consider to be steady, and look at the range of the R-statistic values in those periods. Use the distribution to select values for the Type-1 and Alternate Type-1 triggers. I prefer using common thresholds for all applications. This approach requires unique values for each process variable.
5. Only use the numerator variance measure – the squared deviation of the signal from the filtered value. Do not normalize it by the square of the sequential deviations. If the process is in a transient, the value is high. If at SS, it is low. So, observe the numerator variance in time, and choose threshold values that you observe definitively reveal SS and TS. This simplifies the algorithm by removing a line of code. This approach was my original method of automating the identification of SS from the mid 1980's. It evolved to the ratio method, which was analyzed by my PhD candidate, Songling Cao, in the mid 1990's. The ratio approach is more universal because the variables are scaled. If only testing the numerator variance, each PV needs to have a threshold set by the user, and if the process variance changes with operating conditions, then thresholds are required for each region.
6. Alternately, use one of the simpler approaches to look at the range of data in a window, and if larger than normal claim TS, and if within a normal range accept SS. But this also requires the user to set individual thresholds for each PV, and possibly in each operating range.

Alternately use an alternate technique:

7. Apply alternate techniques that are not confounded by such autocorrelation. My favorite of several is the 4Points method described below. But it is a bit more complicated than the R-statistic filter method.

Multivariable Extension

In a multivariable analysis, there is a greater chance that data vagaries will cause a false TS claim in any one monitored variable. Also, there could be an excessive wait for all PV monitors to return to SS.

One approach is to change the threshold to be more cautious about rejecting SS and more accepting in accepting SS. This analysis is predicated on independence of all PVs.

In expanding the technique for a multivariable analysis, we choose to claim that a process is not at steady state if any process variable is not at steady state, and that the PV might be at steady state if all variables might be. This can be easily computed with a single statistic:

$$SS_{process} = \prod_{j=1}^{N} SS_j \qquad (5)$$

N = Total number of variables in process

j = variable index

SS_j = 1 if the variable is at steady state and SS_j =0 if the individual process variable is in a transient state

$SS_{process}$ = 1 if all variables are at steady state and $SS_{process}$ =0 if any one variable is in a transient state

Assign a value of either 0 or 1 to a dichotomous variable, SSID(i), if the i^{th} process variable is at probable transient or probable steady conditions. At incidences of intermediate values of the R-statistic, the dichotomous SSID(i) variable retains its latest definitive value. Therefore, an $SS_{process}$ value of 1 means that the last definitive state of each variable was steady state – there has not been a definite transient since the last definite steady state. Similarly, an $SS_{process}$ value of 0 means that the last definitive state of at least one variable was transient state – there has not been a definite steady state since the last definite transient.

If the process is at steady state and variations and trends on each variable are independent of other variables then:

$$P(SS_{process}=1) = \prod_{i=1}^{N} P(SS_i=1) = \prod_{i=1}^{N}(1-\alpha_i) \qquad (6)$$

$$(1-\alpha_{process}) = \prod_{i=1}^{N}(1-\alpha_i) \qquad (7)$$

For example, if we want $\alpha_{process}$ = .05, then to determine the required level of significance for the SSID test on each variable.

$$\alpha_i = 1 - \sqrt[N]{(1-\alpha_{process})} \qquad (8)$$

Where

N = number of variables observed in the analysis,

$\alpha_{process}$ = desired overall level of significance for the process statement

α_i = required level of significance for each individual test

If, for example, there are N=10 variables with independent behaviors, and the desired level probability of a Type-I error is 0.001 (99.9% confidence), then from Equation (8) the level of significance for each individual test would need to be about 0.0001 (99.99% confidence).

Again, in a multivariable analysis, there is a greater chance that data vagaries will cause a false TS claim, and there could be an excessive wait for all PV monitors to return to SS. Further, the signals will probably not be independent. If an event makes one PV change or appear to change, it likely affects several PVs. So, as an

alternate solution, the implementer might want to temper the monitors: Claim TS only if 10% or more of the PVs are at TS, and claim SS if 90% or more are at SS.

Cross Correlation

In addition to the R-statistic requirement that there is no autocorrelation between data points (during steady state), Equation (8) also requires that there is no cross correlation between variables (at steady state). When the process is at steady state, this means that the noise on one variable is not correlated to the noise on another. Variables that are being observed should not have correlated noise at steady state conditions. The engineer should find the minimum set of independent process variables which would present a comprehensive view of the process state and which have uncorrelated noise or responses.

For example, consider a tank that has inlet flow, outlet flow, and level measurements. Only two of the three are necessary for observation. The third would be correlated to the other two.

As another example, consider that temperature is measured on both Tray 7 and Tray 8 in a distillation column. If some event changes the composition on Tray 7 it will very soon similarly affect Tray 8, so the temperatures will be cross-correlated. Only one of the two temperatures is needed for observation of that section of the column.

Selection of Variables

A process will likely have many variables that can be observed (for instance tray temperatures on many trays in a distillation column), but if one is in a transient state, whatever causes it will also be affecting other measurements, so not all variables need to be observed. The user should consider which subset of all possible variables are the minimum number that are key to detecting wither TS or SS, and only set up the procedure on that minimal set. This minimizes computational burden, and as well, reduces the number of false alarms generated by normal vagaries in the data. The selection needs user understanding of the process for recognition of redundancy in the variables. The user should also select variables that have the least autocorrelation. For instance if there are two temperature measurements that could be used, and one is in a thermowell and the other not, the thermowell will create autocorrelation. Use the other. Also select variables that show the largest change-to-noise ratio during a transient.

Some variables respond rapidly, such as flow rates, liquid inventory, or pressure. Other variables respond much more slowly, such as composition or temperature in a distillation column. (But, in a heat exchanger, temperature could respond rapidly, and in a reactor composition could change rapidly.) Some variables may be grouped into a subcategory for rapidly responding phenomena such as “hydraulics”. And other variables might be grouped in to a subcategory such as “composition”. This would reveal that one part of the process has settled, a comforting thing to know; but, we need to wait a bit for the residual changes in compositions to come to SS.

Noiseless Data

As a final aspect of the method development, recognize that this statistical method only accommodates noisy data. If the data are noiseless and at steady state, the data series will have exactly the same values, which would drive the denominator measure of variance to zero, eventually leading to an execution error in Equation (4). But, the numerator variance also approaches zero. When either is very small, digital truncation could lead to a large R-statistic, and a not-at-SS claim, even though the PV is at SS. This truncation effect could be device specific.

If a PV approaches a noiseless SS, then a deterministic method is easier to implement and understand – if the ABS(PV-PVprior)<threshold then at SS. One might choose the threshold to be 1/100th of the PV range used to display the PV.

A noiseless signal could arise from several situations: the flow rate is off, the device is broken, the signal is being overridden, the signal is not changing as much as the discrimination error and the reported value is constant, the process is not being used, etc. I'll elaborate on the periodic flat-lining due to digital discrimination. This has happened with on-line analyzers and with temperature sensors. In some operating regions the sample-to-sample fluctuation exceeds the digital discrimination and the measurement seems normally noisy (but it actually just hops between about 5 or so values). But, in other cases the process variation is less than the digital threshold, and the signal flat lines.

To avoid this condition, one could add a normally and independently distributed noise signal of zero mean and small variance NID(0,σ) to the original data. Alternately, one could limit the minimum measure of the estimate of the data standard deviation. In either case, a guide would be to choose a noise sigma, or limit the sigma to 1/100th of the range used for graphical display.

If adding artificial noise, I recommend the Box-Muller method for adding Gaussian distributed independent noise to a simulated signal.

$$x_{measured,i} = x_{true,i} + w_i \tag{9}$$

$$w_i = \sigma\left(\sqrt{-2\ln(r_{1,i})}\sin(2\pi r_{2,i})\right) \tag{10}$$

Here w_i is a normally and independently distributed "white noise" perturbation of zero mean and standard deviation σ variation that is added to the true value to mimic the measurement. The variables $r_{1,i}$ and $r_{2,i}$ represent independent random numbers uniformly distributed in the interval above 0 and up to 1, $0 < r \leq 1$.

A simpler noise-adding approach is to add uniformly distributed, not Gaussian, perturbations. Here $w_i = 2a(r_i - 0.5)$. Where a is the desired amplitude of the perturbation range.

In some control computers, there is no random number generator function. We've had to use our own pseudo-random number generation code as a subroutine!

Bhat and Saraf (2004) also added artificial noise, but to mask autocorrelation.

In any of these cases, the noise amplitude (or sigma) needs to be appropriate to the signal. It should not be too large to override detecting changes when the signal is making real changes, but large enough to mask the flat-line periods. I think a knowledgeable implementer can select an appropriate value. But, this means an additional variable needs to be selected.

However, it might be best to reject such a signal and observe another that represents the same category of process behavior, and does not have noiseless periods.

Andy Robinson, Principal, Phase 2 Automation. Raleigh, NC, suggested to me the simple solution of a minimal threshold on the denominator variance. (Andy wishes to share credit with his colleague David Goodman.) If the flat-line periods (truly a noiseless SS, or changes masked by signal discrimination) would want the denominator variance to approach zero, this would reset it at the threshold. The threshold value would need to be scaled appropriate to the process. The same variation of 1-inch would have a magnitude of 1.6E-5 if the units changed to miles. But I think a process implementer would know what an appropriate minimum value would be for the process. Square that value to represent variance. I think that the PV range represented by the thickness of the line on a plot that is being used to present the data is a good value. If the line is appropriate to the visual monitoring, then this represents the visual discrimination ability. So, I recommend 1/100th of the full scale for display. I like the simplicity and effectiveness of this approach. It only has an impact on the calculation when

the signal is noiseless and the denominator value approaches zero. However, it adds a tuning factor that the user must choose, and which is application specific.

V - ALTERNATE R-STATISTIC STRUCTURE - ARRAY

The rational for the three first-order filters (for average and variances) is to reduce computational burden. But, they have the disadvantage of permitting a past event to linger with an exponentially weighted diminishing factor. Further, the filter structure is not as comfortable for many who are familiar with conventional sums in calculating variances. Further, the window of fixed length N is easier to grasp than the exponential weighted infinite window, or the interpretation of the three lambda values. Alternately, in using a window of N data, the event is totally out of the window and out of the analysis after N samples, which provides a clearer user understanding of data window length. Finally, the use of a single coefficient, N, is more convenient than specifying three separate λ values.

Returning to the base concept with the R-statistic as a ratio of variances, first calculate the average

$$\bar{X} = \frac{1}{N}\sum_{i=1}^{N} X_i \qquad (11)$$

Where i is the sample counter starting at the last sample and is counting back in time N data points in the window. Then the conventional variance can be expanded:

$$\sigma_1^2 = \frac{1}{N-1}\sum_{i=1}^{N}(X_i - \bar{X})^2 = \frac{1}{N-1}\left[\left(\sum_{i=1}^{N} X_i^{\,2}\right) - N\bar{X}^2\right] \qquad (12)$$

And substituting for the average

$$\sigma_1^2 = \frac{1}{N-1}\left[\left(\sum_{i=1}^{N} X_i^{\,2}\right) - \frac{1}{N}\left(\sum_{i=1}^{N} X_i\right)^2\right] \qquad (13)$$

Assuming no autocorrelation (independent deviations) and at SS, the variance can also be estimated as the differences between successive data.

$$\sigma_2{}^2 = \frac{1}{2(N-2)}\sum_{i=1}^{N-1}(X_i - X_{i-1})^2 \qquad (14)$$

There are N-1 terms in a set of N data because there are N-1 differences in the data window, and then N-2 in the divisor

Then the ratio of variances is:

$$R = \frac{\sigma_1{}^2}{\sigma_2{}^2} = 2\frac{(N-2)\left[\sum_{i=1}^{N} X_i{}^2 - \left(\sum_{i=1}^{N} X_i\right)^2/N\right]}{(N-1)\sum_{i=1}^{N-1}(X_i - X_{i-1})^2} = 2\frac{(N-2)}{(N-1)}\frac{[Sum1 - Sum2^2/N]}{Sum3} \qquad (15)$$

This is essentially the von Neumann (1941) approach, but what appears to be a computationally expensive on-line calculation can be substantially simplified by using an array structure to the window of data and a pointer that indicates the array element to be replaced with the most recent measurement. With this concept, the sums are incrementally updated at each sampling.

The method is: First increment the counter to the next data storage location. Then decrement the sums with the data point to be removed. Then replace the data array element with the new value. Then increment the sums. Then recalculate the R-statistic. In comparison to the filter method, this requires storage of the N data, the N squared data differences, and the three sums. The pointer updating adds a bit of extra computation. Our investigations indicate that the array approach takes about twice the execution time as the filter approach. Also,

since it stores 2N data values, it has about that much more RAM requirement. The increased computer burden over the filter method may not be excessive in today's on-line computing devices.

This approach calculates a true data variance from the average in the data window of N samples, and has the additional benefit that the sensitivity of the R-statistic to changes is greater than with the filter method (about twice as sensitive from experience). The array approach can respond to step changes in the PV of about 1.5σ. Accordingly, this defined window, array approach, is somewhat better in detecting small deviations from SS.

Further, the value of window length, N, explicitly defines the number of data after an event clears to fully pass through the data window and influence on the R-value. By contrast the filer method exponentially weights the past data and the persisting influence ranges between about 4/λ to 5/λ.

Additionally, the array method seems less conflicted by autocorrelation and discretization flat spots in the time series of data.

However, the array method is computationally more burdensome than the filter method. Nominally, the filter method needs to store and manipulate 7 values at each sampling. With λ=0.1, this characterizes a window of about 45 samples. For the same characterization, the array method needs to store and manipulate over 180 variables at each sampling.

However, there is only one adjustable variable representing the data window, N. By contrast, the filter approach has three lambda values providing some flexibility in optimizing performance, as done by Bhat and Saraf (2004), but perhaps tripling the user confusion.

Today, my preference is to use the filter method for its computational simplicity. But, if data vagaries (changes in autocorrelation, flat spots, coarse discretization, small shifts relative to noise) make it dysfunctional, then the array method is a bit better.

VI – X-BAR & R SPC APPROACH

Seeking a solution to flat-lining of signals or autocorrelation (perhaps due to controller cycling) led me to a classic Statistical Process Control (SPC) technique, the Shewhart X-Bar & R Chart, which seems to work well. Segregate the sequential data into sets, sub-windows, each with n=5 (or so) data in each set. If there are N sub-windows then there are n*N data in the SPC analysis. Typically, SPC uses 100 data in a chart, but I've been pleased with a total of 50 (n=10 and N=5, or n=5 and N=10, which does not seem critical). Calculate the average, X-Bar, and range, R, for each sub-window as it moves one sample: Place data in a matrix, bumping out the oldest X-Bar and R values. Then calculate the average X-Bar value, X-Bar-Bar, and the average Range, R-Bar. For computational simplicity, first decrement the sums for X-Bar-Bar and R-Bar, then update the matrix, then increment the sums. Use Hartley's coefficient to convert R-Bar to an estimate of the standard deviation. This assumes that data are independently distributed. Nominally, if any subwindow X-Bar is more than 3 sigma from X-Bar-Bar, then claim TS. If every X-Bar is less than 2 sigma from X-Bar-Bar, then the process is possibly at SS, claim at SS.

Alternately, one could calculate the standard deviation of each sub sample, then pool them to estimate the population sigma (assuming at SS in the data window. Hartley's conversion of R-bar to sigma is a classic SPC approach.

Note also, here R represents the range of data within each sub sample, it is not the R-statistic, the ratio of variances.

This approach is a bit more complicated than the filter method, but not more complicated than the array method.

As I implemented this approach, it collects n data in a sub-sample, then does the calculation to see if the new or any past sub-sample averages are out of the 3 sigma range or if all are within the 2 sigma range. So, it only makes a decision after n data. By contrast, the filter and array method looks at the R-statistic after each sampling. So, if

the choice is to have 50 data in a window, the N=10 sub sets with n=5 data each will respond faster than the N=5, n=10 choice.

Also by contrast, data in each sub-window could be updated at each sampling, rather than waiting for a collection of n data to do the test.

I've run this on diverse simulated data sequences, and sets of industrial data, and it seems to work well in spite of autocorrelation, and flat-lining. And, it works well with a variety of reasonable n-data in a sub-window, and number of sub-windows in the analysis.

VII – MACD Approach

This is a dual first-order filter method termed MACD (Moving Average Convergence Divergence). It was developed by Gerald Appel, 1979, and suggested to me by Gregory Stanley (Greg Stanley and Associates). Use two first-order filters with differing time-constants to filter the most recent data. During a transient the filtered values approach the changing data at different rates, but at SS they will both agree on the average value (within normal variability). So, at SS the difference between the filtered values is ideally zero, but it will have some fluctuations about zero due to data noise. The Null Hypothesis is that the time series is at SS, and the difference between the two filtered values is zero (but with its normal fluctuation about zero). When the deviation is large reject the SS hypothesis. The deviation should be scaled by the data inherent variability.

Commonly used in stock trading analysis, MACD is the difference between EWMA12(closing price) and EWMA26(closing price), and a third signal EWMA9(MACD) is graphed with MACD. EWMA means exponentially weighted moving average, a conventional first-order filter. The numbers represent the day-duration of the filters, N, and the filter factor is calculated as 2/(N+1). (Why not just 1/N, I wonder?) The 26, 12, 9 seems to have evolved from practice as most revealing meaningful trends. If MACD is "+" the stock price is increasing relative to recent trend, if "-" it is decreasing. If the EWMA9(MACD) signal crosses the MACD value then a change is happening. The MACD difference is not normalized by the inherent signal variability.

For SSID and TSID: How to choose the two filter time-constants? How to scale the difference by noise? There is autocorrelation in the two filters, both are influenced by the same data, so the normal independence of a difference in a t-test is not true. And what to use for dof in a critical value. It seems that we need our own experience to answer the questions.

I chose to scale the two filter differences by the standard deviation of the noise as measured by the square root of the variance calculated by filtering the square of sequential data differences. This is a t-statistic approach, but not normalized by the number of data, N. As a guide to choosing the filter values, with too large a lambda the variation of the statistic is too great to be able to confidently see small changes, but too small a filter value lengthens the delay to identify change. In seeking to optimize performance, I choose the filter value for the fast lag to be 0.2 and the filter values for the slow lag and variance estimates to be 0.05 each. Monte Carlo simulations to get the statistic distribution, and from it the threshold to confidently reject SS. An alternate Type-I threshold is needed to confidently declare SS. I find that upper and lower critical values of 1 and 0.02 are nearly best for performance.

A key advantage of this approach is its computational and conceptual simplicity.

VIII – 4-Point Approach

This is another relatively simple approach, and has roots in the SPC approach described above. Here, four points in the moving window are looked at, and a filtered value is calculated for each. The first filtered value represents the average of the most recent data value (the first in the window), the second is that of a data point that is (1-gamma)*N samples old, the third is gamma*N samples old, and the fourth is that of data that is N samples old

(the last in the moving window). Gamma is the golden ratio, $\gamma = \frac{\sqrt{5}-1}{2} = 0.618\ldots$ The variance is calculated by the sequential data differences from the leading data. The t-statistic is composed of the maximum difference between filtered values scaled by the filtered sigma.

This seems very simple and very effective.

A classic approach is to segregate the data into the older and recent halves of the window, then do a t-test on the differences of averages in the window. But, in an oscillation, when the split is at the peak or valley, the averages will be the same and SS will be erroneously declared.

A solution could be to divide the entire data window into three sub windows, and the t-test is on the two windows of maximum average difference. Perhaps equal partitioning of the number of data in each sub window will be fine, but maybe unequal spacing, with the longest the most past (perhaps to best establish a historical base line) and the shortest the most recent (perhaps to make the faster recognition). Unequal spacing might be best to not let symmetry in a data pattern confound the test. Or perhaps, the middle window should be the longest, making the difference between 1st and 3rd windows largest in a ramp or first-order change.

I chose 4 points and the golden ratio spacing to eliminate missing a periodic signal.

If multiple windows and a classic t-test some questions still remain: What to use for the pooled sigma or N in the t-test? One could use a filtered data-to-data difference as the simplest to implement, but this would be confounded by autocorrelation in the noise. With autocorrelation, sigma would be small, making normal average differences seem large. One could estimate sigma by the classic deviation from average; but in a ramp, sigma would be large. Filter on all data is simplest.

What to use for DoF in the t-statistic if the sub-windows with greatest difference change? My solution would be to just use the maximum differences scaled by sigma estimate (eliminating the degree of freedom scaling) and let data at SS define the two thresholds.

If using averages in windows, the code needs to transfer data from window #1 to #2 to #2 to #4. Alternately, it could have 1 array with N data and a pointer for each sub window to ID the in and out data. Then the multi-window code and calculation would be similar to the array method of the R-statistic.

But, here 4 points in the past N data are used to represent 4 windows, and the "averages" of the windows are calculated by the simpler first-order filters. The 4 points are the most recent and Nth past data and two intermediate points spaced by the golden ratio. (This is the same as the spacing in the Golden Section optimization approach.) I think that it is unlikely that a periodic oscillation would match that pattern and choose that spacing for that reason.

Based on experience with the other techniques, I chose 0.1 for each data filter lambda and 0.05 for the variance filter. Large filter values create variation that can mask smallish changes, and small filter values can lead to a delay in detecting a TS or a return to a SS.

In a t-test the denominator is a pooled sigma, from variance calculated by the number of items in the two averages in the numerator. Here, with filter factors of 0.1 for the data averages the effective N for averaging is 10=1/0.1. However, the sigma is calculated with a filter value of 0.05, making N effectively 20. It is simpler to not normalize the t-statistic by the dof.

In my investigation of diverse confounding effects, the 4Points approach has been best overall. But when the noise is uncorrelated the filter method is effective and remains the simplest.

IX - TESTING SIMULATOR & METHOD EVALUATION

Some will want to create their own simulations for testing the concept. This would include noise generation, adding autocorrelation and discretization, additive background changes, showing PDF and CDF distributions, determining critical values from CDF, structuring tests to reveal TS ID after SS, and SS after TS, exploring the effects of the two R-critical values on alpha and beta (the probabilities of a T-I or T-II error) and the delay in detecting transitions to SS or TS, and exploring the impact of N and lambda values.

I have an Excel VBA file that simulates several data patterns, and can run the several SSID techniques on the data. The user can choose the data pattern – steady, step, ramp, first order, third order, and oscillation. The user can choose the magnitude and rates of signal changes, amplitude of the noise, method for generating noise, noise distribution, signal discretization, and noise autocorrelation. The user can choose the R-Statistic Filter, R-Statistic Array, X-Bar-R, MACD, or 4-Point method for SSID and TSID. The user can choose sampling interval, or additive noise to mask autocorrelation. The program provides several performance statistics – the number of times the method claims TS when the truth is SS, number of times the method claims SS when the truth is TS, the delay in claiming TS after an event starts, and the delay in claiming SS after an event ends.

The user can independently change the magnitude of the change in a signal (such as the step size or ramp rate) and noise amplitude (standard deviation). However, since all SS&TSID methods are scaled by the noise amplitude, only one of the two need to be changed. Doubling the step increase and doubling the noise amplitude has the same signal to noise ratio as tripling or halving both. So, the Excel Sheet only has an input for the noise sigma. However, you can change the code to make any signal magnitude or rate that you wish.

Sheet1 of the file "SSID 2019 Explore XbarR Array FIlter MACD & 4Points" has the user input cells highlighted in yellow. Procedure run buttons are in gray. Contact me with my company email address, russ@r3eda.com, if you would like the Excel VBA file. The code is open.

Generating Data

Use Cell(3,18) to choose the method for generating the random numbers. "rnd()" is the built in VBA uniform random number generator. This is often fully functional, but it is a single precision generator and in simulations that have millions of calls, it seems to express some repeat or missing values. "uprn" is my code for a classic and common generator. It seems to be better statistically, but only meaningful after millions of realizations, and not as fast as the "rnd()" function.

Use Cell(4,18) to choose the trial. Enter the integer (0, 1, 2, … 6) associated with the list in Column 6 and 7, Rows 1 to 7. If you wish to change the amplitude of the change, you'll need to edit the VBA code. The NID acronym means normally and independently distributed. The UID acronym means uniformly and independently distributed. I use the Box-Muller method to calculate the NID values.

Enter the noise standard deviation in Cell(5,18). It must be >=0. If you choose trial #3, uniform distribution, the range of the uniform distribution is calculated to provide the standard deviation you specify.

The autocorrelation lambda value in Cell(6,18) is the first-order filter factor that blends prior and independently new noise values. The factor must be between 0 and 1. Effectively the lambda value is $1 - e^{-\Delta t/\tau}$, where Δt is the sampling interval, and τ is the time-constant.

The lag lambda value in Cell(7,18) is the filter factor that blends prior and independently new signal values. The factor must be between 0 and 1. Again, effectively the lambda value is $1 - e^{-\Delta t/\tau}$.

The choice of discretization interval in Cell(8,18) truncates sampled values to that interval. The entry must be =>0.

To generate data with that characteristic click on the "Generate Data" button. Time and process variable (PV) data value will be written in Columns 12 and 13, and the truth about SS or TS in Column 16 (a 0 if TS, a 1 if SS).

Observing Autocorrelation

After generating data, select a data period that visually appears at steady state. Enter start and end number of the data sequence in cells(10,18) and (11,18), and a sampling interval in Cell(12,18). If the sample interval is 1, each data is sampled. If 2, every other is sampled. If 5, every fifth is sampled. Most techniques are confounded by autocorrelation. Click on the Autocorrelation button and choose a sampling interval so that the Autocorrelation Chart appears as a shotgun pattern. This will be the sampling interval for the SSID tests.

Running a SS&TSID Algorithm

The yellow highlighted cells in the top of Column 22 are where you input coefficient and trigger values for the 5 SSID algorithms. Run buttons are aligned with their respective input fields. Click on a button of your choice, and the algorithm will display the statistic in Column 14 and the claim about SS or TS in column 15 (a 0 if TS, a 1 if SS). The blue trace in the strip charts is the data, the red is the statistic, and the green is the SS or TS claim.

Once data has been generated you can click on each of the SSID buttons to see how that algorithm analyzes the same data set.

Optimizing Method Coefficient Values

Each technique has choices for coefficient values, for instance of filter relaxation rates or window size, and of the two thresholds. I did a cursory optimization of these values to make the method perform best on the evaluation criteria and the 32 trials I chose for testing. Like a recipe for cooking, the optimization was discretized to convenient values (such as filter lambda choices of 0.05, 0.1 and 0.2) as a recipe is also discretized to convenient values, (2 tsp or 1 heaping tsp, rather than 1.68335 tsp).

One method might be better for one particular type of situation if tuned differently, but the choice of coefficient values and thresholds was an approximate best for overall performance.

Evaluating SS&TSID Methods

Unfortunately, the data generated is just one realization of the choices that provide a stochastic generation of data. Statistical vagaries in the particular time series generation may cause some algorithms to make a false claim. On one particular realization, one SSID algorithm might be better than another, but only after many realizations can one confidently say that a particular algorithm is statistically better. I find that after about 200 realizations one can confidently assess performance. Don't panic. I automated the multiple trials.

A good algorithm will not claim TS when the data is SS, or SS when the data is TS, and will be fast to identify TS when an event starts, and fast to claim SS when an event ends. So, I have the subroutine provide a count of undesirable events – claim TS when SS, claim SS when TS, delay to ID TS, delay to ID SS.

The button "Evaluate A Specified Trial" does this. You input the data generation features in Column 18, and the SSID algorithm coefficient and trigger values in Column 22. Then press the "Evaluate A Specified Trial" button. For 200 times, a new data realization is generated, each algorithm is tested on the data, and the results are displayed in Column 5. Cell(2,5) displays the trial number. Below that is the average of the several evaluation statistics for each method. After each of the 200 data generation realizations, the screen will briefly pause to reveal the 4-Points analysis. You can certainly edit the VBA subroutine "Evaluate" to change the number of realizations, to pause after each SSID algorithm, or to not pause and run faster.

During the trials you will see the cumulative average of the statistics (count of undesirables) in Column 5, fluctuate, and begin to settle on characteristic values for that method on that data feature. After 200 trials, the fluctuations become small enough that a definitive rating can be claimed.

However, this only assesses the SSID methods on one particular type of data feature, such as a step of 2 with a sigma of 0.5 and no autocorrelation. In practice, the algorithm might encounter steps and ramps and oscillation and autocorrelation, and discretization. So a comprehensive test should include 200 trials on all of those situations. The button "Automatically Evaluate All Trials from Sheet2" does this.

Open Sheet2. In Rows 2-5 Columns 3-35 enter your choices for the 32 trials. Then return to Sheet1 and press the button "Automatically Evaluate All Trials from Sheet2". The subroutine "Main" will sequence "Evaluate" through all 32 trials and move the results to Sheet2. Then it will provide overall assessment of the method performance.

Rows 7, 13, 19, 25, and 31 reveal the total count of undesirables. I have each undesirable equally weighted. Certainly you could provide weighting that makes one more important than another. The green highlighted cells reveal the method with the least count of undesirables (best) and the red cells reveal the worst. Columns 37 and 38 provide a count of the times that a particular method was best or worst. But a method might be marginally better than another and the best may be effectively a tie. So, in the rows under the count of best, Columns 37 and 38 provide a count of the times that the method was significantly better or worse, meaning that its total count differed from the next best (or worst) by at least 5. I think the data in Rows 8, 14, 20, 26, and 32 are a more representative assessment of desirability. Column 39 is the difference in best and worst counts, and Column 40 is the relative rating form overall worst (0) to overall best (1).

Rows 44-48 rank the method on each trial. A 5 is best and a 1 is worst. The problem with a rank is that a 235.532 out ranks a 235.533, even if the values are statistically indistinguishable. I think a relative rating is a better way of assessing performance; so, Rows 50-54 provide the relative rating of the 5 SSID methods for each trial. Then Column 37 sums the relative ratings, and presents an overall relative ranking on a 0 (worst) to 1 (best) basis. Note, a 0 rating does not mean it does not work. The filter approach using the R-statistic works, and in my experience works acceptably. It came in, in the performance race, but it was last place of these 5 techniques. In the final race of the Olympics one runner comes in last, but that does not mean it is not a fantastic athlete.

In my assessment of technical performance of the five SSID techniques on a wide variety of test conditions, the X-Bar-R approach is best, and a second-best at a rating of 0.60 to 0.90 depending on the discretization was the 4-Points approach. Third best was MACD with a relative rating of 0.50 to 0.80.

But, simplicity of the algorithms in conceptual understanding, number of user-defined values, and computational burden is also important. I think that the R-Statistic Filter version and MACD are tied as the simplest, and that the 4-Points approach is a close second. I think that the X-Bar-R approach is the most complicated.

Performance also includes a balance of being best or second best, but not worst or second worst. In this sense The X-Bar-R and 4-Points are tied as best and significantly above MACD, which is significantly above Array and Filter.

Balancing simplicity with performance, today, I like the 4-Points approach best for the variety of features in industrial process data and SS & TS ID. However, I still believe the filter method is a strong candidate for convergence criterion (just seeking SS, and uncorrelated data). It is effective and very simple.

In general here are some preliminary observations from exploring noise level, autocorrelation, with some discretization on several transient responses. The evaluation includes number of T-I and T-II errors and delay to recognize TS and SS. The

- Overall all 4Points is best, and Array is second best.
- 4Points and Array are tied for not worst.
- Filter is worst, but the tied for simplest, and fully adequate to recognize SS when uncorrelated noise.

- When oscillation, XBarR was best.
- For autocorrelation 4Points was best.
- For simplicity, MACD and Filter tie as best, with 4-Points a close second

X - ANALYSIS OF THE TECHNIQUE - FILTER

Those interested in mathematical analysis may enjoy this section.

Cao and Rhinehart (1995) provide proofs related to the statistical expectations used in this monograph. If the noise is uncorrelated the expectations (long term average values) of the variables are:

$$E(x_f) = \mu$$

$$E(\upsilon^2_{f,i}) = \frac{2}{2-\lambda_1}\sigma^2$$

$$E(\delta^2_{f,i}) = 2\sigma^2$$

Multivariable ARL

Consider the following situation: There are M independent variables being observed, the common window length is N, and each variable was in a transient state, but simultaneously transitioned to a steady state. SS for the process will be claimed when each identifier is unity, which means that each R-statistic value has had a value below the beta-critical value. Assume a common beta value for all variables, and that each PV is independent. Assume that there is an inconsequential chance that any of the variables will exceed the alpha critical value during the SS period.

There is little chance of claiming SS until N samples after the TS event passes out of the window. Once at SS, the probability that any one identifier switches to SS in a window of k samples after N is

$$P(1 in\, k) = 1-(1-\beta)^k$$

Assuming independence of the R-values (neither autocorrelation of cross correlation), the probability that all M variable identifiers switch in k samples after N is

$$P(1 and\, 2\, and \ldots and\, M\ are\, SS\, in\, k) = \left[1-(1-\beta)^k\right]^M$$ The ARL is then the N plus the k-value that provides a 50% probability of each ID being at SS.

$$ARL = N + k_{50\%} = N + \frac{\ln(1-.5^{1/M})}{\ln(1-\beta)}$$

For the SISO case of M=1 and small beta,

$$ARL \cong N + \frac{\ln(1-.5)}{-\beta/(1-\beta)} \cong N + \frac{0.693147\ldots}{\beta}$$

The independence of R-values is the least defensible of the assumptions in that analysis. Data does reveal autocorrelation, which makes sense, if a confluence of data makes the R-value either high or low, it persists in the window. Further, if some event influences one measurement, it may likely influence another measurement chosen by the user for process monitoring. However, it appears to reveal ARL relations that are consistent with experience.

Power

For small step changes, the R-statistic value might not exceed the critical value. Experience with λ=0.1 shows that the process variable must make a step change of about 3σ (three times the noise standard deviation) or a ramp change of about 3σ within the effective window length (3.5/λ) to confidently trigger detection of a transient. Smaller λ values improve the detectability of small PV changes w.r.t. noise, but this increases ARL.

Filter Relaxation Rates

Equations (1), (2), and (3) are ARMA(1,1) formulations of first-order differential equations, in which

$\lambda_i = 1 - e^{-\Delta t/\tau_i}$

Where i=1, 2, or 3.

The differential equations are

$$\tau_1 \frac{dx_f}{dt} + x_f = x \quad , \quad x_f(t=0) = x_{f,0}$$

$$\tau_2 \frac{dy}{dt} + y = (x - x_f)^2 \quad , \quad y(t=0) = y_0$$

$$\tau_3 \frac{dz}{dt} + z = (\Delta x)^2 \quad , \quad z(t=0) = z_0$$

Where the symbols y and z represent υ^2 and δ^2 from Equations (2) and (3).

Nominally, if at steady state, the values of x_f, y, and z are

$$x_{f,0} = \bar{x} \quad , \quad y_0 = \frac{2}{2-\lambda_1}\sigma^2 \quad , \quad z_0 = 2\sigma^2$$

For simple changes in either average or variance, the differential equations can be solved analytically for a nominal transition of the filtered values from initial to final values, and from this the r(t) transition determined.

Asymptotic Value if Constant Input

If the value of x "freezes" and remains noiselessly constant, x=c, and $\Delta x = 0$, then solving the ODE for the filtered PV value:

$x_f(t) = c - (c - x_{f,0})e^{-t/\tau_1}$

Then numerator variance equation becomes

$$\tau_2 \frac{dy}{dt} + y = (c - x_{f,0})^2 e^{-2t/\tau_1} \quad , \quad y(t=0) = y_0$$

And, if $\tau_2 \neq \tau_1/2$ the solution is

$y(t) = y(t) = ae^{-t/\tau_2} + be^{-2t/\tau_1}$

The solution to the denominator equation is

$z(t) = de^{-t/\tau_3}$

Where a, b, and d are coefficients dependent on initial conditions, the steady x-value, and time-constants.

Then the variance ratio is

$$r(t) = \left(\alpha e^{-t/\tau_2} + \beta e^{-2t/\tau_1}\right)/e^{-t/\tau_3}$$

Where α and β are positive constants. Rearranged

$$r(t) = \left(\alpha e^{-t\left(\frac{1}{\tau_2}-\frac{1}{\tau_3}\right)} + \beta e^{-t\left(\frac{2}{\tau_1}-\frac{1}{\tau_3}\right)}\right)$$

Under this fault condition of a "frozen" signal value, we would desire the R-value to relax to zero (an extreme unexpected value) as an indication of a fault. The two effective time-constants in the above equation are

$$\tau_A = \frac{\tau_3 - \tau_2}{\tau_3 \tau_2} \quad , \quad \tau_B = \frac{2\tau_3 - \tau_1}{\tau_3 \tau_1}$$

And r(t) will relax to zero at a rate determined by the larger of the two

$$\tau_{effective} = \max\{\tau_A, \tau_B\}$$

If the frozen value, x=c, is the initial steady state average, $x_{f,0}$, then the coefficient beta is zero, and effectively only τ_A is relevant.

For r(t) to relax to zero, both τ_A and τ_B must be positive. Which means that $\tau_2 < \tau_3$ and $\tau_1 < 2\tau_3$, which translates to conditions on the lambda values in Equations (1), (2), and (3).

$$\lambda_2 > \lambda_3 \quad , \quad \lambda_1 > 1 - \sqrt{1 - \lambda_3}$$

Values of $\lambda_3 = 0.05$, $\lambda_1 = \lambda_2 = 0.1$ meet those conditions, and are an alternate recommended set.

XI - Combining SS&TS Identification for Experimental Sequence Automation

An important complication occurs when automating a process using this steady state identification method. When the conditions of a process are changed, the measured process variables will usually not change instantaneously. They may remain at steady state for a short time before changes become measurable. To prevent immediate sequencing through a new set of conditions, the automation algorithm should not trigger the next set of conditions until the system passes through a time where it is probably not at steady state. After this has been detected, the algorithm will allow the next set of conditions to begin after the process returns to steady state.

In addition, a time limit for any one run in the experimental procedure can be included in the virtual employee, the cyber operator, in order to identify any occurrence of either 1) a change was made and not detected or 2) the change made the process so unstable that steady state could not be obtained.

Figure 7 (Katterhenry and Rhinehart, 2001; Rhinehart, 2002; Szela and Rhinehart, 2003) illustrates the logic used for the automation method for an experimental sequence. Process data is measured and tested for steady state. If the algorithm determines the process is probably not at steady state, then the value of variable TRANSIENT is set to 1, followed by a check to determine whether or not the time limit has been exceeded. If not, the next sampling is observed and new data is analyzed. If the time limit has been exceeded, the next run is then implemented, the time recorded (the POC variable), and the next sampling is observed and the new data is analyzed. This event will repeat until the algorithm determines the process is probably at steady state. When this occurs, the value for the variable TRANSIENT is tested. If the value is 1, this means that the process was previously in a probably transient condition and is now probably at steady state. Then the system sets the value of TRANSIENT to 0, triggers the next run, and records the time as the point of change. If TRANSIENT is 0, then a check is made to determine whether or not the process has exceeded the time limit. If the time limit has not been exceeded, no trigger occurs and the next process data is tested. If the time limit has been exceeded, the next run is initiated and the time is recorded as the point of change. This logic requires that the process go through a time interval of probably not at steady state before another run will be triggered upon a probably steady state condition.

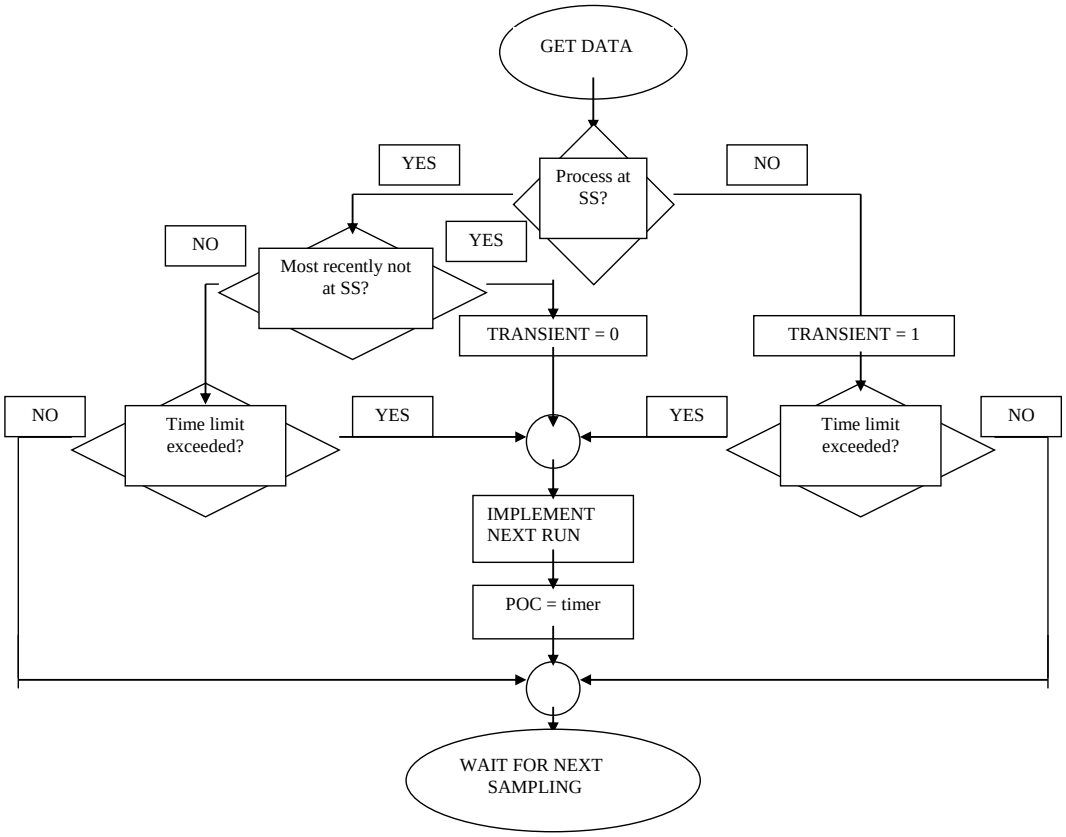

Figure 8. Flow chart of logic for the virtual employee, used for automating a sequence of experimental runs based on process steady state identification.

XII - USING SSID AS REGRESSION CONVERGENCE CRITERION

Nonlinear, least-squares optimization is commonly used to determine model parameter values that best fit empirical data by minimizing the sum of squared deviations (SSD) of data-to-model. Nonlinear optimization proceeds in successive iterations as the search progressively seeks the optimum values for model coefficients. As the optimum is approached, the optimization procedure needs a criterion to stop the iterations. However, the current stop-optimization criteria of thresholds on either the objective function, changes in objective function, change in decision variable, or number of iterations require *a priori* knowledge of the appropriate values. They are scale dependent, application dependent, starting-point dependent, and optimization algorithm dependent; right choices require human supervision.

Iyer and Rhinehart (2000), Padmanabhan and Rhinehart (2005), Ratakonda, *et al.* (2012), and Rhinehart, *et al.* (2012) report on the use of this filter method of SSID to stop optimizer iterations when there is no statistical evidence of SSD improvement relative to the variation in the data.

An observer of an optimization procedure for empirical data will note that the sum of squared deviations (SSD) between the data and the model drops to an asymptotic minimum with progressive optimization iterations. The novelty is to calculate the sum of squared deviations (SSD) of a random subset of data (a unique, randomly selected subset at each iteration). The random subset SSD (RS SSD) will appear as a noisy signal relaxing to its noisy "steady-state" value as iterations progress. At "steady-state", when convergence is achieved, and there is no further improvement in the SSD, the RS SSD value is an independently distributed variable. When RS SSD is confidently at steady-state, optimization should be stopped. Since the test looks at signal-to-noise ratio, it is scale independent and "right" for any particular application.

A root-mean-square, rms, value could be used as an alternate to SSD.

Although the stopping criterion is based on the random and independent subset of data, the optimization procedure still uses the entire data set to direct changes in the Decision Variable (model parameter) values.

Figure 9 illustrates the procedure using rms as the objective. The dashed line represents the rms value of the least squares objective function as it approaches its ultimate minimum with optimizer iterations. The dots represent the rms value of randomly selected fraction of data (15 out of 30 data sets). And the thicker curved line that approaches the dashed line, and lags behind the data, represents the filtered random-subset rms value. The SSID procedure identified SS at iteration number 118, and stopped the optimization procedure. The graph indicates that in iterations beyond 100, the dashed line is effectively unchanging w.r.t. the variability in the data. The variability indicates both data variability and the process-model mismatch. In either case, the optimizer was not making substantive improvement in the model.

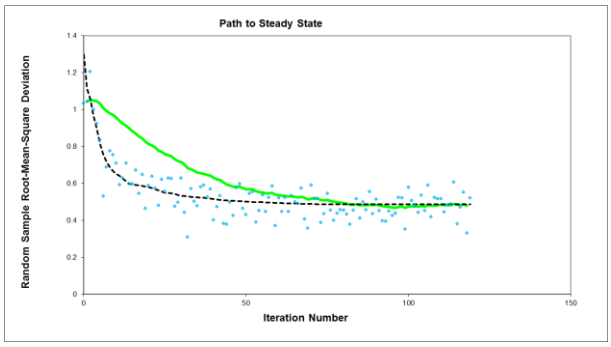

Figure 9: SSID application to nonlinear regression.

XIII - USING SSID AS STOCHASTIC OPTIMIZATION CONVERGENCE CRITERION

Monte Carlo type simulations are often used to model the outcome of events with probabilistic influences such as rolling the dice, device reliability, molecular behavior, or human interactions. The models support the design of best response rules or system features. The simulations return a stochastic value, a value that changes with each replication, with each duplication of nominally identical inputs. The distribution of outcomes represents what one might expect in the real world, but this means that each simulation provides a different answer. One could average many simulations, but the Central Limit Theorem indicates that the variability can be attenuated by $\sqrt{N}$, not eliminated.

Figure 10 represents a search for coefficients in the best lead-lag rules to guide the motion of a defensive robot soccer player attempting to intercept the ball-carrying opponent. The game pieces start in randomized places and

the evasive moves of the opponent are also randomized. The surface appears as a curved valley with steep walls and an irregular floor. This is one realization of the surface.

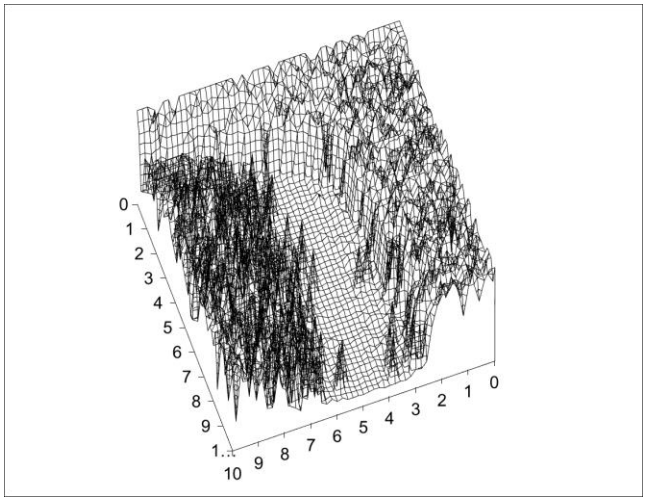

Figure 10: A stochastic optimization problem

At any generation, the surface retains the same general shape, but the irregularities shift. The numbered axes represent scaled value of the lead and lag coefficients. The vertical axis represents the objective function value, a combination of fraction of defensive failures (goals scored) and playing-time to intercept the ball. The objective is to minimize the OF value, to find the lowest place in the valley floor. Unfortunately, at each sampling the floor has a different value, as if you were standing on a trampoline bed and kids were jumping on it all around you or trying to measure the lake height when it is continually perturbed by waves and ripples. Every re-sampling of the exact same spot yields a different value.

To seek the best set of decision variables (DV) (design parameters of hardware, procedures, or rules), optimizers use the objective function (OF) value as a search guide, and often also as a stopping criterion. However, the stochastic nature of the OF value is a confounding aspect that will guide the solution toward an improbable, fortuitous best value. One solution (Rhinehart, 2014) is to use a multiplayer approach, and reevaluate the apparently best solution and assign a worst-of-N realizations as the OF value.

Any number of stopping criteria can be used, but the common ones related to the OF value, or to the change in OF value, or to the change in DV values are scale dependent, and require *a priori* knowledge. However, SSID can also be applied to determine convergence by observing the worst OF value at each iteration. During the phase of the search in which the optimizer is making improvements, the noisy OF value will progressively change. However, at convergence, the OF value will average at the same value, but sequential trial solutions will return a noisy response. The iteration sequence value of the best OF has autocorrelation, the worst player OF value has none.

XIV – IMPLEMENTATION

Each variable has to have a SSID value of 1 to confidently state the process is at SS. If any variable has a zero value, then it has not confidently been at SS since it was in a TS, and at least one part of the process would not be at SS.

This filter method looks at each variable independently, and uses the same core algorithm to determine the SSID value. This is like a common PID function would be sequentially called to calculate many output values.

The method only uses the current and most recent past data, but this is on the sampling interval that eliminates autocorrelation, which might be a lower frequency than the DCS/PLC/PC uses for monitoring or control.

Lambda values that we've found to best balance Type-I and Type-II errors, the average run length to recognize SS, and user convenience are .1, .1, and .1.

For nonlinear regression optimizer convergence, there is one variable to observe and it sequential values are uncorrelated. The method can be directly applied and the ideal critical values used. However, for process analysis, user involvement is usually required to set it up. First, the user needs to choose what variables to monitor. These should be minimum set that indicates the important features. If too many variables are selected, there will always be an aberration in one of them, and the identifier will never find all of them at SS. For a minimum set, for example: If there are three temperature sensors on near-by trays, or three flow meters in a line, they all should be indicating the same thing. Only one is needed. If there are in and out flow measurements on a tank then tank level is not needed (if the in and out flows are constant, the level must be constant).

As a part of this, the user should realize that hydraulics (flow rates, and levels) come to SS faster than gas pressure in large volumes, faster than temperature or composition. So, the user might want to separate the process attributes into a set of hydraulic variables, another set of inventory (T, P, x) variables, and monitor the hydraulic state of the process and the inventory state separately. My thanks to Anand Vennavelli, Research Engineer, Fractionation Research, Inc., Stillwater, OK, for this insight

Second, the user needs to choose a sampling interval that eliminates autocorrelation in the steady state data. This could be automated, but for me it is easiest to eye-ball select a SS period then progressively shift the data until there is no eye-ball apparent autocorrelation in the data. One certainly could run each SSID algorithm on a time scale that is right for each variable. But for simplicity, in multivariable applications, I've run all on the same sampling interval, the longest in the set of variables we were observing.

Best results are with an unfiltered or not averaged, raw data. Filtering or averaging creates autocorrelation. This means the SSID must run at lower frequency, which means it takes longer to report on an event. The display for the operators, the historian, and the value for the controller could still be filtered or averaged.

Ideally, the trigger values to confidently state TS and SS are about 3 and 1. However, I've found that there is usually some autocorrelation remaining. Further, the greater the number of variables the higher the upper r-critical must be to prevent Type-I error. Ideally, Equation (8) would indicate the proper alpha value, but this is also predicated on no cross correlation between observed variables. Accordingly, and third in the human choices, I've found it more convenient and effective to increase/decrease the trigger values a bit to make the SS/TS selection match human judgment.

When the SSID is initiated we do not know whether the process is at SS or TS. So, we initialize the SS values with an in-between value, perhaps 0.5. If there are 3 variables, and each is initialized with 0.5, the product is 0.5^3=0.125. Subsequently, when all variables are confidently at SS, the product is 1, or when any one is confidently in a TS the product is 0. So, there could be three possible display values – 0, 1, or an in-between value. If product = 1 then report "Confidently at SS". If product = 0 then report "Confidently in (or after) a recent TS". If product is between 0 and 1 then report "Initializing".

XV – Balance in Choosing Coefficients

Here is a collection of notes and observations:

- In the filter or array method, increasing Trigger1 (the probably not at SS threshold, so call it TS threshold) reduces false TS claims when at SS, but it will cause the identifier to miss small transients, giving a false SS when at a small TS. One must choose T1 to balance false TS and missed TS claims.

- In the filter or array method, decreasing Trigger2 (the probably at SS threshold, so call it SS threshold) reduces false SS claims when in a small TS, but delays the claim of SS after the process is truly settled after a TS. One must choose T2 to balance speed of response to surety of the SS claim.
- Increasing the number of data in the window (or equivalently decreasing lambda) reduces the variability of the statistic and permits less extreme T1 and T2 trigger values (or reduces the number of false claims), but it requires a longer SS period to make the at-SS claim. A longer window of data delays the recognition of SS.
- Increasing sample interval (time between samples) reduces (or eliminates) autocorrelation, which makes the data match the assumed noise attributes (independent and random) of the methods, making the methods truer to the data, resulting in fewer false claims, but it delays the response.
- Increasing the number of variables observed increases the probability of a false TS claim.
- If the threshold is set to the 3 sigma value (the 99.73 % level of confidence), and the data are ideally noisy, then there is a 0.27% chance of a false TS claim when actually at SS. If a PV is sampled once per second, and observed that frequently, then there are 60*60*24 samples per day, leading to a probable 60*60*24*0.0027=233 false TS claims per day. If SSID-TSID observation is reduced to once per minute, then there are only 4 false TS claims per day, but the identification will be delayed. The observation frequency needs to balance the effect of false claims and requirements for immediacy of response.
- If one variable is being observed, and it has an alpha level of significance, then if two redundant variables are being observed and both are at SS, the probability of a false TS claim is $\alpha_2 = 1 - (1-\alpha)^2$. If n redundant variables, then $\alpha_n = 1 - (1-\alpha)^n$. The more variables being observed, the higher the probability of a false TS claim. So, minimize the number of variables. Be sure that the subset of variables observes all aspects of a process, but do not use redundant variables. Choose the subset of variables to be observed, from the redundant variables that are 1) most nearly ideal NID(0,sigma) when at SS, 2) make the largest changes in a transient, and 3) are least subject to spurious events.
- In the X-Bar-R method, the data is divided into N subwindows of n data each. The average range of the n data sets is used to calculate the sigma of the data, and scaled by $\sqrt{n}$ provides the sigma of the X-Bar. But, the method claims TS if any of the X-Bars are beyond the critical number of sigmas. This is an OR conjunction, so the probability of one being rogue increases with the number of subwindows, N. So, for instance, if a 95% confidence in a TS is desired, then alpha = 0.05 for the test, and if there are N subwindows, then alpha for an individual must be $\alpha_N = 1 - (1-\alpha)^N$, or $\alpha = 1 - \sqrt[N]{(1-\alpha_N)}$. So if a 2-sigma certainty is desired, then for N=10 windows, $\alpha = 0.005$, which is the ~2.8 sigma level on each sub-window test.
- How many data should be in a subwindow and how many windows? Classic SPC uses 100 data in a window, so if there n=10 in a subwindow, then there should be N=10 subwindows. But still, how to divide the data? 2 and 50, or 50 and 2, or 25 and 4? If too few data in a subwindow, then the X-Bar is biased by the autocorrelation. So, the more the better. But, if no autocorrelation, large n-data is not needed to get an unbiased estimate of X-Bar. The larger the n (data in a sub window) the longer is the interval between the next SS or TS assessment, but too few data in each subwindow increases the uncertainty on the R-bar value, reducing sensitivity. I've been pleased with 50 total data, and 5 or 10 in a window. But what might be universally best?
- A solution to detecting small transients when doing experimental runs, is to reinitialize the SSID filter values, array, or past X-Bar & R values with zero whenever new experimental conditions are implemented. Often an experimental design does not randomly jump to new conditions, but incrementally adjusts conditions (to maximize safety, reduce upsets, accelerate time to next SS, permit experimenter direction, to come close to but avoid limiting issues like flooding). However, small incremental changes, might not make enough of a change in the signal to noise value to trigger TS. If you know that new conditions were sent to the process, then there is no need to wait 'til the TS

detection happens. You know it did, so reinitialize the memory to produce the TS signal and then just wait for the SS decision.

- These balances depend on the application. If they lead to human action (investigation, correction), then the cost of false alarms may be greater than the penalty for delayed recognition or missed minor events. Here a 4-sigma threshold may be appropriate. However, if the action is automated (control, turning data or video recording on or off) or if "better be safe than sorry" is the philosophy, then excessive operation may not be an issue, and a 2-sigma threshold and observation of redundant variables and higher frequency sampling may be fully acceptable. The right choices for levels of significance, sampling interval, and which variables are to be observed requires domain and context knowledge more so than statistical knowledge.
- My new favorite approach to dealing with signals that flat-line is to place a threshold on the numerator sigma estimate (in the filter or the array method), or on the R-bar value in the SPC method. My prior approach was to add noise to the measurement. In any case one has to choose the noise amplitude or the limit on the variance or r-bar value. I would offer that if one looks at the PV range one the plots that humans use to observe the data, then if the change in the signal is less than $1/1000^{th}$ of that entire range, it is at SS. So perhaps artificial noise could have an amplitude of $1/100^{th}$ of the range, or the thresholds could have a similar value.

XVI - CONCLUSION

This tutorial presents and analyzes several methods to automate detection of probable steady state and probable transient state conditions. The methods are scale independent, extend to multivariable systems, and are independent of noise level of distribution. The user needs to choose a sampling interval to eliminate autocorrelation.

The author has been a part of application of the filter method on a variety of pilot-scale and full-scale process and optimization applications.

An array version of the filter method and a Shewhart X-Bar and R approach are also described. These are more complicated computationally, but are less sensitive to autocorrelation and seem to be better in identifying small shifts.

The 4-Points parallel to an X-Bar-R analysis seems to be the best balance of simplicity and several performance metrics on a wide range of SS and TS events.

For regression and optimization, however, the noise is uncorrelated and only the approach to SS needs to be detected. Here the filter approach is fully adequate and simple to implement and well used by the author. However, I suspect that the sigma-scaled MACD approach would be equivalent; but, I have not tested it.

In a multivariable process the user needs to select variables to be monitored, and criteria should be that observed variables need to be the minimum number that observes all aspects of the process. In a multivariable analysis, there is a greater chance that data vagaries will cause a false TS claim. Also, an excessive wait for all PV monitors to return to SS. So, the implementer might want to temper the monitors: Claim TS only if 10% of the PVs are at TS, and claim SS if 90% or more are at SS.

XVII - ACKNOWLEDGMENT

This work was partly sponsored from an endowment from the Amoco Foundation; and largely dependent on the energy, creativity, and investigations of the authors whose names appear in the reference list.

XVIII – REFERENCES

[1] Albers, J. E., Hydrocarbon Processing, March, 1994, pp. 65-.

[2] Alekman, S. L., "Significance tests can determine steady-state with confidence", CONTROL for the Process Industries, Putman Publications, Chicago, IL, Vol. 3, No. 11, November 1994, pp. 62 & 64.

[3] Barolo, M., G. B. Guarise, C. Marchi, S. A. Rienzi, and A. Trotta, "Rapid and reliable tuning of dynamic distillation models on experimental data", Chemical Engineering Communications, Vol. 190, No. 12, 2003, pp. 1583-1600.

[4] Bhat, S. A., and D. N. Saraf, "Steady-state identification, gross error detection, and data reconciliation for industrial process units", Industrial and Engineering Chemistry Research, Vol. 43, No. 15, 2004, pp. 4323-4336.

[5] Brown, P. R.; and R. Russell Rhinehart, "Demonstration of a Method for Automated Steady-State Identification in Multivariable Systems", Hydrocarbon Processing, Vol. 79, No. 9, 2000, pp. 79-83.

[6] Cao, S., and R. R. Rhinehart, "An Efficient Method for On-Line Identification of Steady-State," Journal of Process Control, Vol. 5, No 6, 1995, pp. 363-374.

[7] Cao, S., and R. R. Rhinehart, "Critical Values for a Steady-State Identifier," Journal of Process Control, Vol. 7, No. 2, 1997, pp. 149-152.

[8] Carlberg, P. J., and D. M. Feord, "On-line model based optimization and control of a reactor system with heterogeneous catalyst", Proceedings of the American Control Conference, Albuquerque, NM, June 1997, pp. 2021-2022.

[9] Castiglioni, P., G. Merati, and M. Di Rienzo, "Identification of steady states and quantification of transition periods from beat-by-beat cardiovascular time series: Application to incremental exercise test", Computers in Cardiology, Vol. 31, 2004, pp. 269-272.

[10] Crowe, E. L., F. A. Davis, and M. W. Maxfield, Statistics Manual, Dover Publications, New York, NY, 1955.

[11] Flehmig, F., and W. Marquardt, "Detection of multivariable trends in measured process quantities", Journal of Process Control, Vol. 16, 2006, pp. 947-957.

[12] Forbes, J. F., and T. E. Marlin, Industrial Engineering and Chemistry Research, Vol. 33, 1994, pp. 1919-.

[13] Fruehauf, P. S., and D. P. Mahoney, "Distillation column control design using steady state models: Usefulness and limitations", ISA Transactions, Vol. 32, 1993, pp. 157-175.

[14] Huang, T, "Steady State and Transient State Identification in an Industrial Process", MS Thesis, Oklahoma State University, 2013

[15] Huang, T., and R. R. Rhinehart, "Steady State and Transient State Identification for Flow Rate on a Pilot-Scale Absorption Column", Proceedings of the 2013 American Control Conference, Washington DC, June, 2013

[16] Iyer, M. S., and R. R. Rhinehart, "A Novel Method to Stop Neural Network Training," Proceedings of the 2000 American Control Conference, June 28-30, 2000, Chicago, IL, Paper WM17-3, pp. 929-933

[17] Jeison, D., and J. B. van Lier, "On-line cake layer management by trans membrane pressure steady state assessment in anaerobic membrane bioreactors", Biochemical Engineering Journal, Vol. 29, 2006, pp. 204-209.

[18] Jiang, T., B. Chen, X. He, and Paul Stuart, "Application of steady-state detection method based on wavelet transform", Computers and Chemical Engineering, Vol. 27, 2003, pp. 569-578.

[19] Jubien, G., and G. Bihary, "Variation in standard deviation is best measure of steady state", CONTROL for the Process Industries, Putman Publications, Chicago, IL, Vol. 3, No. 11, November 1994, pp. 64.

[20] Katterhenry, P. R., and R. R. Rhinehart, "Use a Virtual Employee to Trigger a Sequence of Conditions", CONTROL for the Process Industries, Vol. XIV, No. 10, 2001, pp. 53-55.

[21] Keeler, J. D., E. Hartman, and S. Piche, "Process modeling and optimization using focused attention neural networks", ISA Transactions, Vol. 37, No. 1, 1998, pp. 41-52.

[22] Keller, J. Y., M. Darouach, and G. Krzakala, Computers and Chemical Engineering, Vol. 18, No. 10, 1994, pp. 1001-.

[23] Kim, M., S. H. Yoon, P. A. Domanski, W. V. Payne, "Design of a steady-state detector for fault detection and diagnosis of a residential air conditioner", International Journal of Refrigeration, Vol. 31, 2008, pp. 790-799.

[24] Kuehl, P., and A. Horch, "Detection of sluggish control loops – experiences and improvements", Control Engineering Practice, Vol. 13, 2005, pp. 1019-1025

[25] Le Roux, G. A. C., B. F. Santoro, F. F. Sotelo, M. Teissier, and X. Joulia, "Improving steady-state identification", Proceedings of the 18th European Symposium on Computer Aided Process Engineering – ESCAPE 18, B. Braunschweig and X. Joulia Editors, Elsevier B.V./Ltd., London, 2008

[26] Lin. T. D. V., Hydrocarbon Processing, April 1993, pp.107-.

[27] Loar, J., "Use SPC to compare process variables with control limits", CONTROL for the Process Industries, Putman Publications, Chicago, IL, Vol. 3, No. 11, November 1994, pp. 62.

[28] Mansour, M. and J. E. Ellis, "Methodology of on-line optimization applied to a chemical reactor", Applied mathematical Modeling, 2007, doi:10.1016/j.apm.2006.11.014

[29] Nachtwey, P., in LinkedIn Automation & Control Engineering discussion group, topic "Automated Steady State Identification" http://www.linkedin.com/groupAnswers?viewQuestionAndAnswers=&discussionID=144128653&gid=1967039&commentID=94720027&trk=view_disc&ut=3L2w_8e797U5o1, 2012

[30] Ochiai, S., "Calculating process control parameters from steady state operating data", ISA Transactions, Vol. 36, No. 4, 1997, pp. 313-320.

[31] Prata, D. M., M. Schwaab, E. L. Lima, and J. C. Pinto, "nonlinear dynamic data reconciliation and parameter estimation through particle swarm optimization: Application for an industrial polypropylene reactor", Chemical Engineering Science, Vol. 64, 2009, pp. 3953-3967.

[32] Shrowti, N., K. Vilankar, and R. R. Rhinehart, "Type-II Critical Values for a Steady-State Identifier", Journal of Process Control, Vol. 20, No. 7, pp. 885-890, 2010.

[33] Padmanabhan, V., and R. R. Rhinehart, "A Novel Termination Criterion for Optimization", Proceedings of the 2005 American Control Conference, paper ThA18.3, Portland, OR, June 8-10, 2005, pp. 2281-2286.

[34] Ratakonda, S.; U. M. Manimegalai-Sridhar; R. R. Rhinehart; and S. V. Madihally, "Assessing Viscoelastic Properties of Chitosan Scaffolds and Validating with Cyclical Tests", Acta Biomaterialia, Vol. 8, No. 4, April, pp 1566-1575, 2012.

[35] Rhinehart, R. R., "Automated steady-state identification: experimental demonstrations", Journal of Process Analytical Chemistry, Vol. 7, No. 2, 2002, pp. 1-4.

[36] Rhinehart, R. R., M. Su, and U. Manimegalai-Sridhar, "Leapfrogging and Synoptic Leapfrogging: a new optimization approach", Computers and Chemical Engineering, Vol. 40, 11 May 2012, pp. 67-81.

[37] Rhinehart, R. R., Nonlinear Regression Modeling for Engineering Applications: Modeling, Model Validation, and Enabling Design of Experiments, Wiley, New York, NY, September, 2016. ISBN 9781118597965. 361 pages with companion web site www.r3eda.com.

[38] Rhinehart, R. R., Engineering Optimization: Applications, Methods, and Analysis, 2018, John Wiley & Sons, ISBN-13: 978-1118936337, ISBN-10:1118936337, 776 pages with companion web site www.r3eda.com.

[39] Rhinehart, R. R., "Convergence Criterion in Optimization of Stochastic Processes", Computers & Chemical Engineering, Vol. 68, 4 Sept 2014, pp 1-6

[40] Salsbury, T. I., and A. Singhal, "Method of and apparatus for evaluating the performance of a control system, US Patent 7,024,336, April 4, 2006

[41] Schladt, M., and B. Hu, "Soft sensors on nonlinear steady-state data reconciliation in the process industry", Chemical Engineering and Processing, Vol. 46, 2007, pp. 1107-1115

[42] Stanley, G., in LinkedIn Automation & Control Engineering discussion group, topic "Automated Steady State Identification" http://www.linkedin.com/groupAnswers?viewQuestionAndAnswers=&discussionID=144128653&gid=1967039&commentID=94720027&trk=view_disc&ut=3L2w_8e797U5o1, 2012

[43] Subawalla, H. A. J. Zehnder, M. Turcotte, and M. J. Bonin, "Multivariable optimization of a multiproduct continuous chemicals facility", ISA Transactions, Vol. 43, 2004, pp. 153-168.

[44] Svensson, C., in LinkedIn Automation & Control Engineering discussion group, topic "Automated Steady State

Identification" http://www.linkedin.com/groupAnswers?viewQuestionAndAnswers=&discussionID=144128653&gid=1967039&commentID=94720027&trk=view_disc&ut=3L2w_8e797U5o1, 2012

[45] Szela, J. T., and R. R. Rhinehart, "A Virtual Employee to Trigger Experimental Conditions", Journal of Process Analytical Chemistry, Vol. 8, No. 1, 2003.

[46] Vasyutynskyy, V., "Passive monitoring of control loops in building automation", Technical Report, Technische Universitat Dresden, Department of Computer Science, 2005

[47] Vennavelli, A. N., and M. R. Resetarits, "Demonstration of the SS and TS Identifier at the Fractionation Research Inc. (FRI) Distillation Unit", proceedings of the 2013 American Control Conference, paper TuC19.2, pages 4425-4429, Washington DC, June, 2013

[48] von Neumann, J., R. Kent, H Bellison, and B Hart, "The mean square successive difference", The Annals of Mathematical Statistics, 1941, 12, 153-162.

[49] von Neumann, J, "Distribution of the ratio of the mean square successive difference to the variance", The Annals of Mathematical Statistics, 1941, 12, 367-395.

[50]Yao, Y., C. Zhao, and F. Gao, "Batch-to-Batch steady state identification based on variable correlation and Mahalanobis distance", Industrial Engineering Chemistry Research, Vol. 48, No. 24, 2009, pp. 11060-11070

[51] Ye, L., Y. Liu, Z. Fei, and J. Liang, "Online probabilistic assessment of operating performance based on safety and optimality indices for multimode industrial processes", Industrial Engineering Chemistry Research, Vol. 48, No. 24, 2009, pp. 10912-10923.

[52] Zhang, J., "Developing robust neural network models by using both dynamic and static process operating data", Industrial Engineering Chemistry Research, Vol. 40, No. 1, 2001, pp. 234-241.

XIX. BIO

Russ Rhinehart is an emeritus professor of the School of Chemical Engineering at Oklahoma State University, where he served as program Head for 13 years. He was in industry for the first third of his career. He authored several books (https://www.amazon.com/author/rrussellrhinehart), supports a web site (www.r3eda.com), and has a regular "Develop Your Potential" series in CONTROL magazine, all with the hope to enable self-learners on topics related to engineering analysis. His degrees are from the U of Maryland and North Carolina State U. He is a fellow of ISA (the International Society of Automation), inductee to the Control Process Automation Hall of Fame, and Past President of the American Automatic Control Council (www.a2c2.org). He offers short courses and seminars on topics related to modeling, control, and optimization.

XX. APPENDICES

Code for X-Bar-R Algorithm

```
'
'
Sub SSID_XBarR() 'Classic SPC X-Bar and R technique
   'nominally should have 100 data in a whole window, but 50 seems to work well
   'this is dimensioned to permit segregating data into up to 20 subwindows,
   '   but 10 of 5 data each seems to work well
   'if this flatlines then accept SS

Dim xbar(20), xrange(20)
```

```
Cells(1, 15) = "X-Bar-R"
Cells(1, 14) = "max deviation"

upper = Cells(8, 22)
lower = Cells(9, 22)
stepsize = Int(Cells(12, 18))
If stepsize < 1 Then stepsize = 1

nsubwindows = Cells(10, 22)      'number of subwindows
ndata = Cells(11, 22)            'number of data elements in a subwindow
Hartley = 1.1274 * Log(ndata) + 0.483   'RRR model for Hartley's coefficient.
'Hartley's coefficient to convert R-bar into sigma - See Oakland page 232
sqrtnH = Sqr(ndata) * Hartley

noiseamplitude = Cells(1, 22)  'artificially added noise as a portion of PV range (uniform distribution)
Randomize          'randomize generation of masking perturbation
uprn_m = 2# ^ 32
uprn_x = Timer * (uprn_m / (24# * 60# * 60#)) 'initial seed randomized by the computer clock
   'and scaled for its possible range

For kkk = 1 To nsubwindows            'initialize subwindow values
   xbar(kkk) = 0
   xrange(kkk) = 0
Next kkk                          'subwindow counter
sumxbar = 0
sumxrange = 0
SS = 0.5    'indeterminate

kk = 1                      'pointer for subwindow
For i = 2 To 1201 Step ndata * stepsize  'windows of n sequential data values times stepsize
   xsum = 0
   For k = 1 To ndata * stepsize Step stepsize      'data in subwindow separated by stepsize
      x = Cells(i + k - 1, 13) + noiseamplitude * (2 * uprn - 1)   'get data
      If k = 1 Then             'find max and min in sub window
         xmax = x
         xmin = x
      Else
         If x > xmax Then xmax = x
         If x < xmin Then xmin = x
      End If
      xsum = xsum + x           'sum to calculate average in sub window
   Next k

   sumxbar = sumxbar - xbar(kk)       'decrement prior window sums
   sumxrange = sumxrange - xrange(kk)
   xbar(kk) = xsum / ndata        'calc and assign average in a n-data subwindow
   xrange(kk) = xmax - xmin       'calc and assign range in a n-data subwindow
   sumxbar = sumxbar + xbar(kk)       'increment to create new window sums
   sumxrange = sumxrange + xrange(kk)
```

```
      kk = kk + 1                  'increment subwindow pointer
      If kk > nsubwindows Then kk = 1
      xbarbar = sumxbar / nsubwindows 'grand average
      rbar = sumxrange / nsubwindows  'average range
      If rbar < 0.1 Then rbar = 0.1   'limit on average range

      TSindicator = 0       'initialize not in a TS, accept SS
      SSindicator = 1       'initialize can't reject SS, accept SS
      maxdeviation = 0      'here it is the deviation in x-variable terms
      For kkk = 1 To nsubwindows
         subdev = Abs(xbar(kkk) - xbarbar)
         If subdev > maxdeviation Then maxdeviation = subdev
         If subdev > upper * rbar / sqrtnH Then TSindicator = 1 'if any one is excessive, then 99% confident TS
         If subdev > lower * rbar / sqrtnH Then SSindicator = 0 'if any one is excessive, then 95% confident not SS
      Next kkk
      maxdeviation = maxdeviation * sqrtnH / rbar  'this converts it to the number of sigmas
      If maxdeviation > 5 Then maxdeviation = 5   'display convenience

      If TSindicator = 1 Then SS = 0  'confident in not at SS
      If SSindicator = 1 Then SS = 1  'confident in at SS

      For kkkk = 1 To ndata * stepsize              'display analysis
            Cells(i + kkkk - 1, 14) = maxdeviation
            Cells(i + kkkk - 1, 15) = SS
      Next kkkk
   Next i

End Sub
```

Code for MACD Algorithm

```
'
'
Sub SSID_MACD()

   Cells(1, 15) = "MACD"
   Cells(1, 14) = "t-statistic"

   upper = Cells(12, 22)    't-statistic trigger for TS
   lower = Cells(13, 22)    't-statistic trigger for SS
   stepsize = Int(Cells(12, 18))
   If stepsize < 1 Then stepsize = 1

   l1 = Cells(14, 22)        'filter lambda values, fast EWMA
   l2 = Cells(15, 22)        'slow EWMA
   l3 = Cells(16, 22)        'variance EWMV
   cl1 = 1 - l1
   cl2 = 1 - l2
   cl3 = 1 - l3
```

```
   vf = 0
   x1f = 0
   x2f = 0
   xold = 0
   SS = 0.5          'indeterminate
   noiseamplitude = Cells(1, 22)  'artificially added noise as a portion of PV range (uniform distribution)
   Randomize         'randomize generation of masking perturbation
   uprn_m = 2# ^ 32
   uprn_x = Timer * (uprn_m / (24# * 60# * 60#)) 'initial seed randomized by the computer clock
      'and scaled for its possible range

   For i = 2 To 1201 Step stepsize       'sigma-scaled MACD Method
      x = Cells(i, 13) + noiseamplitude * (2 * uprn - 1)
      x1f = l1 * x + cl1 * x1f    'fast change
      vf = l3 * 0.5 * (x - xold) ^ 2 + cl3 * vf
      If vf < 0.01 Then vf = 0.01
      xold = x
      t = Abs(x1f - x2f) / Sqr(vf)
      If t > 5 Then                'no sense in making it relax from absurd values
         t = 5
         vf = Abs(x1f - x2f) / t
      End If
      For ii = i To i + stepsize
         Cells(ii, 14) = t
      Next ii
      x2f = l2 * x + cl2 * x2f    'slow change - placed here to prevent autocorrelation in X1f and x1f
      If t <= lower Then SS = 1    'probable SS
      If t > upper Then SS = 0    'probable TS
      For ii = i To i + stepsize
         Cells(ii, 15) = SS
      Next ii
   Next i

End Sub
```

Code for Filter Algorithm

```
'
'
Sub SSID_Filter()      'Cao-Rhinehart Filter Method

   Cells(1, 15) = "Filter"
   Cells(1, 14) = "r-statistic"

   upper = Cells(2, 22)    'r-statistic trigger for TS
   lower = Cells(3, 22)    'r-statistic trigger for SS
   stepsize = Int(Cells(12, 18))
   If stepsize < 1 Then stepsize = 1
```

```
   l1 = Cells(4, 22)          'filter lambda values
   l2 = Cells(4, 22)
   l3 = Cells(4, 22)
   cl1 = 1 - l1
   cl2 = 1 - l2
   cl3 = 1 - l3

   n2f = 0           'initial values
   d2f = 0
   xf = 0
   xold = 0
   SS = 0.5          'indeterminate
   noiseamplitude = Cells(1, 22)  'artificially added noise as a portion of PV range (uniform distribution)
   Randomize         'randomize generation of masking perturbation
   uprn_m = 2# ^ 32
   uprn_x = Timer * (uprn_m / (24# * 60# * 60#)) 'initial seed randomized by the computer clock
      'and scaled for its possible range

   For i = 2 To 1201 Step stepsize       'Cao and Rhinehart Method
      x = Cells(i, 13) + noiseamplitude * (2 * uprn - 1)
      n2f = l2 * (xf - x) ^ 2 + cl2 * n2f
      xf = l1 * x + cl1 * xf
      d2f = l3 * (x - xold) ^ 2 + cl3 * d2f
      If d2f < 0.01 Then d2f = 0.01      'Andrew Price Robinson (Andy) idea, Phase2Automation, December 2018
      xold = x
      If d2f > 0 Then
         r = (2 - l1) * n2f / d2f
         If r > 5 Then              'no sense in making it relax from absurd values
            r = 5
            n2f = d2f * r / (2 - l1)
         End If
         For ii = i To i + stepsize
            Cells(ii, 14) = r
         Next ii
      End If
      If (2 - l1) * n2f <= lower * d2f Then SS = 1   'probable SS
      If (2 - l1) * n2f > upper * d2f Then SS = 0    'probable TS
      For ii = i To i + stepsize
         Cells(ii, 15) = SS
      Next ii
   Next i

End Sub
```

Code for Array Algorithm

```
'
'
```

```
Sub SSID_Array()       'Rhinehart-Gore Array Method

   Dim y(100)             'up to N=100 values in window  (Certainly could make it more)
   Dim dy2(100)

   Cells(1, 15) = "Array"
   Cells(1, 14) = "r-statistic"

   For j = 1 To 100     'initialize array - up to 50 data
      y(j) = 0
      dy2(j) = 0
   Next j

   upper = Cells(5, 22)
   lower = Cells(6, 22)
   stepsize = Int(Cells(12, 18))
   If stepsize < 1 Then stepsize = 1

   'initialize
   N = Cells(7, 22)    'number of data in the window, fewer means greater variability, more is longer to recover SS
   sum1 = 0
   sum2 = 0
   sum3 = 0
   yold = 0
   jn = 1      'numerator data element
   jd = 1      'denominator data element
   SS = 0.5     'indeterminate
   noiseamplitude = Cells(1, 22)  'artificially added noise as a portion of PV range (uniform distribution)
   Randomize          'randomize generation of masking perturbation
   uprn_m = 2# ^ 32
   uprn_x = Timer * (uprn_m / (24# * 60# * 60#)) 'initial seed randomized by the computer clock
      'and scaled for its possible range

   For i = 2 To 1201 Step stepsize
      jn = jn + 1
      If jn > N Then jn = 1
      jd = jd + 1
      If jd > N - 1 Then jd = 1
      sum1 = sum1 - y(jn) ^ 2
      sum2 = sum2 - y(jn)
      sum3 = sum3 - dy2(jd)
      y(jn) = Cells(i, 13) + noiseamplitude * (2 * uprn - 1)
      dy2(jd) = (y(jn) - yold) ^ 2
      yold = y(jn)
      sum1 = sum1 + y(jn) ^ 2
      sum2 = sum2 + y(jn)
      sum3 = sum3 + dy2(jd)
      If sum3 < 0.01 Then sum3 = 0.01
      r = 2 * (N - 2) * (sum1 - sum2 ^ 2 / N) / ((N - 1) * sum3)
```

```
      If r > 5 Then r = 5
      For ii = i To i + stepsize
         Cells(ii, 14) = r
      Next ii
      If 2 * ((N - 1) / N) * (sum1 - (sum2 ^ 2) / N) <= lower * sum3 Then SS = 1 'probable SS
      If 2 * ((N - 1) / N) * (sum1 - (sum2 ^ 2) / N) > upper * sum3 Then SS = 0   'probable TS
      For ii = i To i + stepsize
         Cells(ii, 15) = SS
      Next ii
   Next i

End Sub
```

Code for 4-Points Algorithm

```
'
'
Sub SSID_4Points()       'Rhinehart t-statistic on 3 windows Method
   '  Here Four points in the moving window are looked at and a filtered value is calculated
   '     for each.  The first filtered value represents the average of the most recent data
   '     value, the second is that of a data point that is (1-gamma)*N samples old,
   '     the second is gamma*N samples old, and the last is
   '     that of data that is N samples old.  The variance is calculated by the sequential
   '     data differences from the leading data.  The t-statistic is composed of the maximum
   '     difference between filtered values scaled by the filtered sigma.
   '  This seems very simple and very effective

   Dim y(100)            'up to N=100 values in window  (Certainly could make it more)

   Cells(1, 15) = "4Points"
   Cells(1, 14) = "t-statistic"

   For j = 1 To 100     'initialize array - up to 100 data
      y(j) = 0
   Next j

   upper = Cells(17, 22)
   lower = Cells(18, 22)
   stepsize = Int(Cells(12, 18))
   If stepsize < 1 Then stepsize = 1

   'initialize
   N = Cells(19, 22)    'number of data in the window, fewer means less detectability of slow trends, more is longer
to recover SS
   yold = 0                'for sigma updating
   datavariance = 0          'initialization of variance estimate
   g = (Sqr(5) - 1) / 2
   iput = 1    'write data number for placing new data in array, and also the read number for most recent
   iread4 = iput + 1   'Start number for oldest data set
```

```
iread3 = Round((1 - g) * N, 0)     'Start number for third data set
iread2 = Round(g * N, 0)      'Start number for second data set
y1filt = 0
y2filt = 0
y3filt = 0
y4filt = 0

SS = 0.5     'indeterminate
noiseamplitude = Cells(1, 22)  'artificially added noise as a portion of PV range (uniform distribution)
Randomize          'randomize generation of masking perturbation
uprn_m = 2# ^ 32
uprn_x = Timer * (uprn_m / (24# * 60# * 60#)) 'initial seed randomized by the computer clock
   'and scaled for its possible range

For i = 2 To 1201 Step stepsize
   y(iput) = Cells(i, 13) + noiseamplitude * (2 * uprn - 1)
   datavariance = (0.05) * 0.5 * (y(iput) - yold) ^ 2 + (0.95) * datavariance
   yold = y(iput)
   y1filt = (0.1) * y(iput) + (0.9) * y1filt
   y2filt = (0.1) * y(iread2) + (0.9) * y2filt
   y3filt = (0.1) * y(iread3) + (0.9) * y3filt
   y4filt = (0.1) * y(iread4) + (0.9) * y4filt

   iput = iput + 1           'increment pointers
   If iput > N Then iput = 1
   iread2 = iread2 + 1
   If iread2 > N Then iread2 = 1
   iread3 = iread3 + 1
   If iread3 > N Then iread3 = 1
   iread4 = iread4 + 1
   If iread4 > N Then iread4 = 1

   maxfilt = y1filt    'find max and min
   minfilt = y1filt
   If y2filt > maxfilt Then maxfilt = y2filt
   If y2filt < minfilt Then minfilt = y2filt
   If y3filt > maxfilt Then maxfilt = y3filt
   If y3filt < minfilt Then minfilt = y3filt
   If y4filt > maxfilt Then maxfilt = y4filt
   If y4filt < minfilt Then minfilt = y4filt

   If datavariance < 0.01 Then datavariance = 0.01
   t = (maxfilt - minfilt) / Sqr(datavariance)
   If t > 5 Then t = 5
   For ii = i To i + stepsize
      Cells(ii, 14) = t
   Next ii

   If t <= lower Then SS = 1 'probable SS
```

```
      If t > upper Then SS = 0    'probable TS
      For ii = i To i + stepsize
         Cells(ii, 15) = SS
      Next ii
   Next i

End Sub
```