Improving the Development of the I-Chart For Use in Biopharmaceutical Manufacturing Operations

Cleveland State University

McNair Scholar's Program Summer Research Institute Paper

Kimberly Schveder

August 25, 2017

Abstract

The Shewhart control charts are statistical tools used by pharmaceutical companies, as well as chemical and other batch manufacturers, to help detect errors in the manufacturing process and ensure control of product quality. One particular type of control chart is the I-chart. The average run length (ARL) statistic of the I-chart is a key measure indicative of the chart's reliability for identifying atypical process output. The ARL statistic can easily be simulated when the manufacturing process is normally distributed with known population parameters. This paper investigates the impact on the ARL statistic when the I-chart is based on mean and standard deviation estimates obtained from small sample sizes of 50 batches or less. The methodology of Quesenberry (1993) is employed to ascertain the impact of small sample estimation on I-chart performance and provide recommendations for how I-charts should be constructed to account for the uncertainty of using a small number of batches to construct them. In order to do this, two values used in the construction of the width of the confidence limits were examined. The limits are determined by $\pm$ k x standard deviation. This study varied the values of k and used two different methods to estimate the standard deviation. For samples of size n = 30, 40, and 50, the value of k = 2.75 and the Levey-Jennings method of estimating the standard deviation provide I-chart limits that are better than the industry's current standard. It is too soon to draw any conclusions for smaller sample sizes.

Keywords: Shewhart, Control Charts, I-charts, Levey-Jennings, Moving Range Method

Contents

Improving the Development of the I-Chart

For Use in Biopharmaceutical Manufacturing Operations

Introduction

Once upon a time, a grandmother had 20 grandchildren and wanted to make each child a batch of cookies that were equal in size, texture, and had the same amount of chocolate chips, so that the children would not fight over the best cookies. So, after making each batch, the grandma sampled only one cookie (to save the rest for her grandchildren) to see if its batch had the same type of cookie as the previous batch(es) of cookies. Many pharmaceutical companies are under similar situations as the grandma, her batches of cookies, and her 20 grandchildren.

Pharmaceutical companies (and other manufacturing companies) want to make several batches of medicine and other chemicals to distribute and sell while following the FDA's rules and regulations on manufacturing such chemicals. According to Peng, Lionberger, Viehmann, Iver, & Yu (2015),

> Control charts are … useful tools to monitor the routine commercial production for identifying continual improvement opportunities during product life cycle. When the control chart detects the presence of a special cause, continual improvement strategy can be initiated to correct and prevent potential failures so that the process remains in control (p. 70).

It is clear that control charts are very useful tools when monitoring process control and for reporting to the FDA regarding product quality. Specifically, according to Gardner, Shewhart control charts primarily are used in pharmaceutical manufacturing plants to monitor a process for quality and process control. They are also often reported to the FDA as a way to demonstrate product quality (2015). These Shewhart control charts include the I-chart.

The I-chart is a Shewhart Control Chart for individual measurements. Each sample taken from each batch consists of a single item or measurement. Because only an individual item or measurement is taken from each batch, the sample size referred to in this paper is the number of product batches that have been produced by the process. If there are n batches, the sample size is n, in other words.

There is an example of an I-Chart in Figure 1. To construct the center line and control chart boundaries, a fixed number of batches used, with one measurement per batch. This is referred to as Phase 1. It uses the estimate for μ (the true mean), $\overline{x}$, as the center line, where $\bar{x}$ is the mean of all of the sample values (i.e. a given number of batches, where the observation of size one is taken from each batch). The Upper Control Limit (UCL) and Lower Control Limit (LCL) are calculated and placed onto the chart (respectfully above and below the centerline) so that they can be used to detect any outliers, values that are outside of the control chart limits. Then, this is the control chart bounds that are used for the rest of the production (called Phase II). When outliers are detected, the system will likely be out of control.

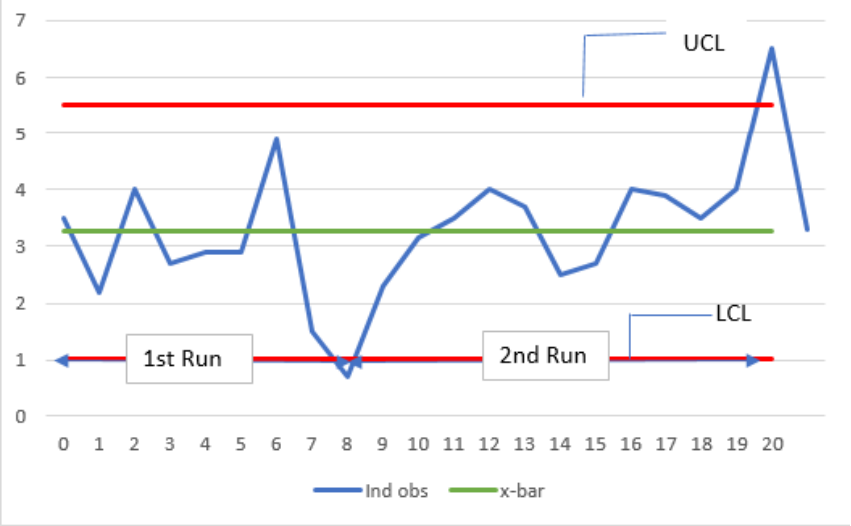


Figure 1: An example of an I-Chart (a.k.a an Individual Control Chart).

In practice, particularly in biopharmaceutical manufacturing, the I-chart limits are calculated as $\bar{x} \pm 3$ sd, where sd, the process standard deviation, is estimated via the moving range method (see section 2.1). In other words, this case is the conventional way of calculating the control limits for the I-chart and is compared to the nine other charts (see sections 4 and 5). The overall intent in these experiments were to establish a range of expected batch results so that, if a result is observed outside of the established control limits, a manufacturer may investigate this signal of poor batch quality and potentially prevent the subject batch from being released to the market.

However, there is still a chance that the outliers detected via the above method are false signals - that is, that the outliers indicate that the manufacturing process is out of control in some way, when in reality the process is in control (also known as a Type I error) - especially when the control limits are based on statistical analysis of a small number of product batches. Are there adjustments that we need to make based on the initial number of batches used to create these limits? Might there be a better approach for estimating the standard deviation and/or adjusting

the width of the control limit range based on the initial sample size (number of batches)? Quesenberry (1993) had the very same questions. He wanted to know "how large the sample size n must be for the estimated control limits to perform essentially like the 'known' limits" (p. 244). Thus, he performed a simulation study to find out.

The I-chart will be discussed in great detail here. The impact of sampling error on the performance of the Shewhart process control chart for individual measurements when monitoring a stable process will be found. The two methods for estimating the standard deviation for the control limits of the I-chart will be evaluated for their potential to achieve optimal performance given the size of the sample used to determine such limits. This will involve finding the ideal width for control limits to be calculated (given the sample size) to approximate the performance of the standard chart with known population parameters. Furthermore, the method of calculating the sample standard deviation so that it affects the performance of the control chart as compared to the standard chart will be examined.

Since there is an interest in improving the performance of the I-chart during the Phase I calculations based on a small number of batches (either n=10,20,30,40,50), there are two parameters that were investigated. First, I will examine the different confidence widths to understand their impact. So, instead of using 3 as in the conventional manner, using multipliers of k = 2.5, 2.75, 3.0, 3.5, 4.0. Second, the method of calculating the sample standard deviation will be examined using both the conventional moving range method and the Levey-Jennings method. To determine the performance of the chart, the ARL will be calculated for Phase II simulations.

Background

I-Chart Theory

Quesenberry (1993) studied the "Characterization of Run Length Distribution for Individual Measurements Charts." He addressed and answered the questions stated in the Introduction section. Quesenberry specifically used the Moving Range (MR) method to run his simulations. According to Quesenberry (1993), "if the process mean μ and standard deviation σ were known, the control limits for a classical Shewhart control chart for μ would be given by:"

$$UCL = \mu + 3\sigma$$

and

$$LCL = \mu - 3\sigma$$

(p. 242). In reality, μ and σ are unknown, so estimations must be calculated instead.

The first calculation needed to proceed with Quesenberry's method is the MR. According to Montgomery (2009), MR is calculated as

$$MR_i = |x_i - x_{i-1}|, \quad i = 2, 3, \ldots, m.$$

(p. 260). In other words, the MR is a measurement of the difference between successive measurements. There are m-1 MR_i values. We must average these values to get an estimate of $\overline{MR}$:

$$\overline{MR} = \sum \frac{MRi}{m-1}$$

The $\overline{MR}$ is used to calculate the UCL and LCL of the individual chart.

Because $\bar{x}$ is an estimate of μ and $\hat{\sigma} = \overline{MR}/d_2 = \overline{MR}/1.128 = 0.8865\overline{MR}$ is an estimate of σ. Lastly, the UCL and LCL can be found by the following (Notice that these equations come from substituting $\bar{x}$ in for μ and $\hat{\sigma}$ in for σ.):

$$UCL = \bar{x} + 3(.8865\overline{MR}) = \bar{x} + 2.6595\overline{MR}$$

and

$$LCL = = \bar{x} - 3(.8865\overline{MR}) = \bar{x} - 2.6595\overline{MR}$$

(Quesenberry, 1993, p. 244). The d_2 constant that is used above to calculate the $\hat{\sigma}$, according to Milivojevich (2014), came from a statistical process control tables. It is used to estimate the underlying population standard deviation. The d_2 constant is dependent on the number of observations used to calculate the moving range, which in this case is two; there is one observation per batch, and to calculate the MR, you need observations from two consecutive batches (Milivojevich 2014).

There is an alternative method to the MR Method, one that was not used by Quesenberry. This is called the Levey-Jennings (LJ) Method, also known as a Standard (SD) method to calculate control limits by using a standard method of calculating the σ estimate (i.e. not using a moving range and d_2 estimate). This standard method is completed by using the sample standard deviation, $s = \sqrt{\sum \frac{(x-\bar{x})^2}{n-1}}$, where x is each scored value of an observation. According to Karkalousos & Evangelopoulos (2011), the LJ method plotting begins

> from the distribution curve of the control limits (μ±3s). The first step is to rotate clockwise the distribution curve of the control limits by 90^o. The second step it to draw seven lines which start from the points μ+3s, μ+2s, μ+s, μ, μ-s, μ-2s and μ-3s. These

seven lines form the Levey-Jennings chart. For every different control level, a different Levey-Jennings chart is being plotted.

(pp. 341-42). As seen in Figure 2, each of the different control levels indicate a different point in the control process. The upper and lower control limits remain the μ ± 3s.

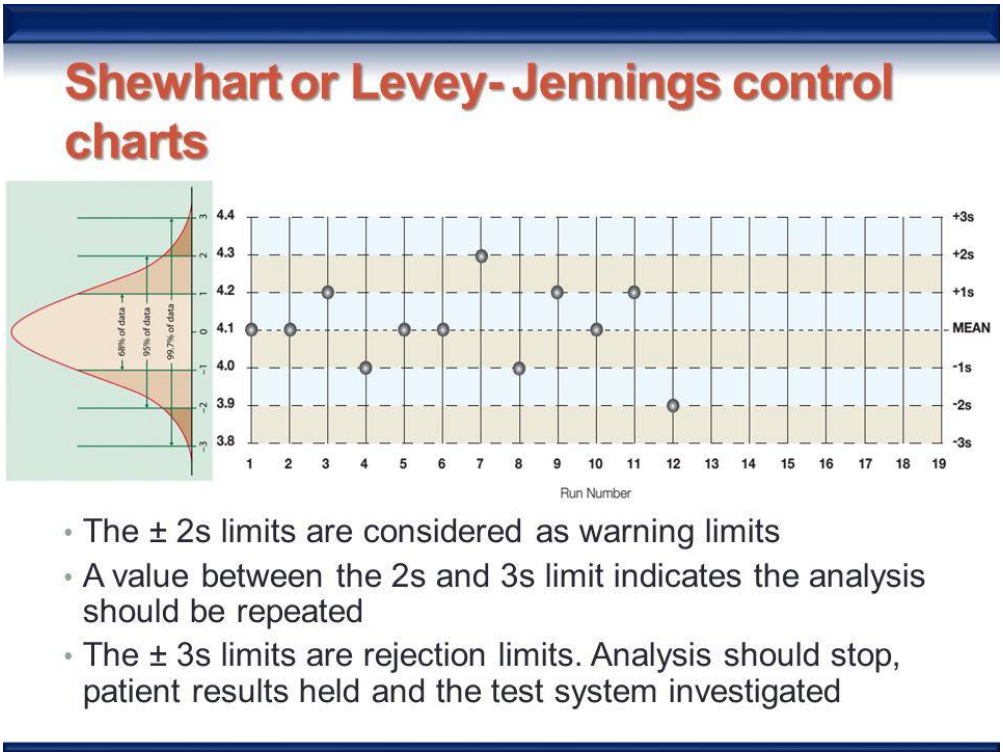


Figure 2: Graphing and Interpreting a Levey-Jennings Control Chart. Perry. J. (2016). Shewhart or Levey- Jennings control charts. Licensed by slideplayer.com.

In other words, the Levey-Jennings method calculates an estimate of standard deviation in a standard way, and gets different values of the UCL and the LCL (we are only concerned with the two lines considered the rejection limits), and thus different values of the ARL, than from the MR method. Both the MR method and LJ (SD) method were used on each value of k and each batch size n, and then the results from each were compared with each other to detect performance variations. The MR method estimates the standard deviation by the calculation of the moving range (Montgomery, 2009, pp. 266-267). Both the MR and LJ methods were tested

under the same experimental cases to see which method works better with which experimental cases (see sections 4 and 5).

Many terms above can be defined and clarified with an example and a story. Suppose that a grandma wants to make 20 batches of chocolate chip cookies with the right texture and consistency (soft and chewy) for her grandchildren. Beforehand, the grandma tested out her recipe several times and made several batches of cookies for herself and for the parents of her grandchildren (who do not mind as much about the quality of their cookies), and then hence established her UCL and LCL. After baking her Phase I and establishing the control limits, she then hired a cooking assistant to help her cook the 20 batches for her 20 grandchildren. In Phase II, grandma used the UCL and LCL to test to see if the cookies in the new batches are in bound. As seen in Figure 1, the scale of softness might be on a scale from 0 to about 7, with 0 being the hardest and 7 being the softest and the most undercooked chocolate chip cookie. The grandma doesn't want the cookies to be dark, crisp, and crunchy on one hand, or too gooey on the other, respectively.

These two types of cookies are our two control limits; she should strive to bake the cookies in the very middle of these two extremes. The two red solid lines, one the highest and the other the lowest, are the Upper Control Limit (UCL) and the Lower Control Limit (LCL), respectively. Each batch's individual softness will be plotted around the average level of softness the grandmother strives for. The blue line connects the values of the observations of size one from each batch together. The centerline, $\bar{x}$, is the green, solid line, and is the mean of these observations.

Once Phase II starts, a run consists of the number of plotted values until and including the first value that is out of bounds. The process is restarted, and then a second run is started. The

Average Run Length (ARL) is the mean of all of the run lengths. In Figure 1, the first run has a run length of 8, the second RL is 12. For Figure 1, the ARL = (8+12)/2 = 10. When the observation falls out of bounds, the grandma should check and fix her process to avoid over-baking or under-baking the cookies, perhaps by adjusting cooking temperature or by adding more or less ingredient(s). Her cooking assistant carries out her orders, as she demands. Then, the grandma's baking process will be back on track to making the ideal cookies.

The results from Quesenberry's (1993) experiments "show that the estimated control limits perform essentially like the limiting case only for a very large sample size n of 2000 or so … There is a clear dependence of the events that consist of comparing the values X_i with the estimated control limits, even for n of 500 and 1000" (pp. 245-246). (The "limiting case" that were mentioned above is known as the "ideal case", or the yellow control lines in each of the Panels (also called cases) in section 4). He found that more false alarms should be expected after short runs, and, between these false alarms, a few more very long runs occur. Hence, he concludes that, to avoid such dependence from estimating the control limits, the sample size n should be at least 300 units.

However, in the context of pharmaceutical manufacturing, Quesenberry's recommendations may be unrealistic. Due to lengthy production processes (spanning weeks in many cases for a single product batch) it would take years to achieve the sample sizes Quesenberry recommends. Furthermore, the FDA requires control limits to be in place in order for manufacturers to release product to the market, so companies have a vested interest in implementing limits as quickly as possible. Lastly, according to Montgomery (2009), each item in a batch may be identical to each other, but are different from those in other batches, "only

because of laboratory or analysis error, as in many chemical processes" (p. 259). Thus, the company may want to see if there are any between-group differences.

Limitations and Potential Problems with the I-Chart

There are several limitations and potential problems with the I-Chart. The most relevant limitations and potential problems of the I-Chart are those regarding false alarms. According to Montgomery, the individual control chart's ability to detect small shifts is not very good (2009). Say a shift of one standard deviation in a manufacturing process occurs. In order to detect the shift, it would take about 44 samples in the out-of-control state (Montgomery 2009). The resulting economic consequences could be serious. Thus, individual control charts may be limited in their usefulness during Phase II (see section 3.2) of process monitoring.

Furthermore, although some have thought about using control limits narrower than the standard 3σ on the I-Chart in order to make its ability to detect small process shifts better, such a suggestion is dangerous (Montgomery 2009). Narrower limits will, in fact, reduce the number and size of the individual run lengths greatly and hence affect the size of the ARL by reducing it. Therefore, the number of false alarms increases, and the charts become much less reliable in quality and process control.

The occurrence of false alarms increases when control limits are based on a small sample of data. This is because the probability that any point exceeds the control limits increases as the sample size decreases. The ARL is the reciprocal of this probability by definition (Montgomery 2009), and so the ARL also decreases.

Methods of Research

In this paper, Quesenberry's approach is adapted to study the effects of sampling error when constructing the I-chart using small numbers of batches (e.g. 50 or less). Additionally, different values of k standard deviations are used to adjust the width of the control limit range. The objective here is to assess whether a wider control range helps offset the effect of sampling error. All simulations were completed in RStudio for Windows, version 1.0.143.

Experimental Settings

In this simulation, there were five levels of the value k, the value that was used to calculate the control limits in Phase 1 of the simulation: k = 2.5, 2.75, 3.0, 3.5, 4.0. Quesenberry's (1993) values of k were mimicked here because he chose his in increments of 0.5. After the original values of k were complete, a change was noticed in Panels A, B, E, and F, in section 4, where the values of k = 2.5 and k = 3.0, so a value of k = 2.75 was tested, for every n, so that the midpoint results could be observed.

The n observations in each batch (i.e. the sample size) were set as either n = 10, n = 20, n = 30, n = 40, or n = 50. I-Charts were constructed by both the MR method and the LJ or SD, method. There are $5 \times 5 \times 2 = 50$ total experimental settings to look at in this simulation: 25 for the LJ (SD) method and 25 for the MR method**.**

Specification of Estimate Precision

Quesenberry (1993, p. 244) states that the number of simulations run for each sample size were such as to achieve a "sufficiently small standard error" for the ARL estimate. However, Quesenberry's definition of "sufficiently small" is not specified and therefore unknown. For the current study the following methodology was used to achieve a "sufficiently small" standard

error. Assume that N is the number of run lengths collected and the ratio of the standard error (se) to the mean is called "Ratio." Refer to Figure 3 to follow along with the following procedure:

1. Simulations for a Phase 1 data set, with a given sample size n, value of k, and method (LJ (SD) or MD method) were run.
2. When N = 10,000, point 1 (Pt1) was set equal to (N, Ratio).
3. When N = 20,000, point 2 (Pt2) was set equal to (N, Ratio).
4. The difference D(ratio) was computed as follows: D(Ratio) = Pt2(Ratio) – Pt1(Ratio). Here, a vertical distance was calculated.
5. If D(Ratio) was found to be less than .005, the precision threshold is equal to Pt1(Ratio) and the procedure stops.
6. If D(Ratio) was found to be greater than .005, Pt1 is set equal to Pt2 and another 10,000 simulations are generated per step 1.
7. When N = Pt2(N) + 10000, N and its corresponding ratio were stored in Pt2.
8. The procedure goes back to step 4.

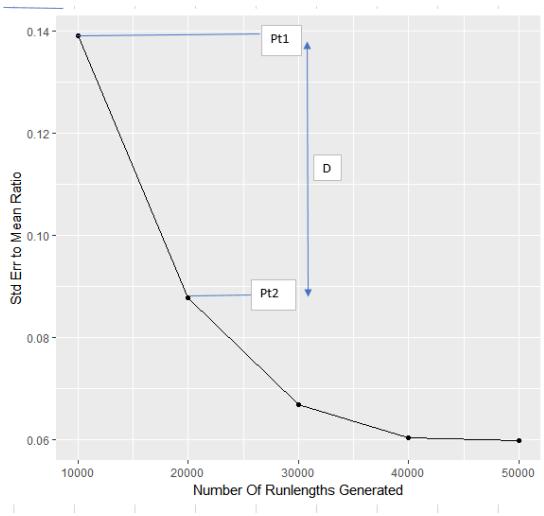


Figure 3: Visualizing the Determination of the Precision Threshold. The change in N from Pt1 to Pt2 is by 10,000 run lengths. k=2.5 and the sample size is 50.

So, in summary, instead of stopping at a fixed value for all simulation cases like Quesenberry did, the loop continued until the change in the standard error to mean ratio drops below .005 (measured every 10,000 generated run lengths). Once that change drops below .005, the loop stops and takes the previous value of the change in the standard error to the mean. For each given n and k, the above procedure was repeated four times (see Appendix section 6), and then the smallest value generated was taken as the precision value for each of the 50 simulations. Altogether, we repeated this procedure 50×4=200 times.

A few remarks are needed to explain the above procedure. This process is guaranteed to stop because there is an asymptote to zero of the se to mean ratio, as the number of run lengths approaches infinity, as seen in Figure 3. The value 0.005 was chosen as the value for the D(Ratio) in the above algorithm because it is close to zero. This cutoff value helped to simulate and choose a sufficiently small standard error, according to Quesenberry, in a limited time frame.

Execution of Data Simulations

Monte Carlo simulations in the study have been performed in the same manner as Quesenberry as follows, for a given sample size n and value of k:

LOOP (repeated 10,000 times):

1. A "Phase 1" dataset with a single variable containing n observations was randomly generated, using the standard normal probability distribution. So, in other words, a variable x has n observations taken from a probability distribution where 99.7% of the values are between -3 and 3.

2. From the randomly generated data in (1), the sample mean $\bar{x}$ and sample standard deviations *s* were calculated.
3. The I-chart control limits ($\bar{x} \pm k s$) were calculated based on the "Phase 1" data using the parameter estimates from (2).
4. A “Phase 2” dataset with a single variable was generated by repeatedly producing random observations from the standard normal probability distribution until a generated observation fell outside of the calculated control limit range from (3).
5. The number of observations *m* produced in the “Phase 2” dataset in (4) was recorded as a run length *L*.

END LOOP

6. The mean and standard deviation of all of the values *L* was calculated to give us the ARL for the simulation.
7. The standard error of the set of observations *L* from the previous step was calculated, given the standard deviation and the size of the set of observations *L*.
8. If se/ARL is below the specified precision threshold (see section 3.2), the process is stopped. If not, start again at step 1. The summary statistics calculated here varied every time.

Results

All of the following tables of results are formatted in the following way:

For each set of simulation conditions (sample size, method of estimating standard deviation, and k), 10,000 simulations provided 10,000 run lengths. From those simulations, a mean and standard deviation were calculated. They are reported as ARL and standard deviation

in Table 1. Additionally, the standard error was calculated for each set of simulation conditions. Quesenberry (1993) formatted his tables in the same way as the ones above. The ARL along with its standard error and standard deviation of the run lengths were included for each of the 50 cases because "considering just the run length distribution mean and not its standard deviation [and standard error] can be misleading when judging a control procedure" (Quesenberry, 1993, p. 242).

Table 1A: The MR-Simulation Method Results

MR	n = 10	20	30	40	50
k=2.5	2780.5 (127.4, 29535.7)	487.6 (20.5, 7239.9)	216.5 (6.7, 1880.1)	151.1 (3.8, 487.3)	126.6 (2.4, 418.4)
2.75	6574.1 (231.5, 46747.7)	1729.1 (76.4, 18139.8)	745.4 (31.2, 7313.4)	451.7 (11.6, 3160.4)	349.6 (8, 1478.8)
3.0	13470.3 (416.1, 72908.8)	5167.5 (182.1, 37371.2)	2464.2 (99.4, 19728.2)	1455.8 (53.3, 9513.7)	1033.5 (33, 6105.3)
3.5	31938.5 (806.4, 112135.4)	24358.5 (531.9, 92565.3)	17451.1 (410.4, 71413)	13743.0 (341.4, 60099.4)	10530.9 (274.4, 47050.4)
4.0	53202.7 (1045, 147109.4)	56515.5 (1021.9, 144783.3)	57362.5 (997.9,142730.9)	54163.9 (933.9, 133429.5)	54088.7 (930.9, 130481.7)

Table 1B: The Levey-Jennings Method Results

LJ (SD)	n = 10	20	30	40	50
k= 2.5	514.6 (27.8, 8810.2)	141.08 (3.1, 431.8)	116(1.95, 295)	104.3 (1.3, 199)	97.2 (1.2, 166)
2.75	1810.3 (85.4, 19729.8)	446.7 (12.1, 3198.4)	297.4 (6.6, 992.8)	246.6 (4.1, 612.9)	226.9 (3,448.4)
3.0	4770.3 (170.5, 36199)	1515.7 (55.6, 10979.3)	846 (23, 4258)	647 (12.4, 2263.9)	587.9 (12.4, 1773.6)
3.5	21300.6 (497, 86391.9)	3201.7 (330.8,58997.8)	8752.7 (229.5, 39617.6)	6512.5 (160, 29157.4)	5164.17 (118, 18657.4)
4.0	48192.1 (938.6, 135004.8)	52327.7 (957.6, 134353.6)	46898.8 (835.8, 118276)	44016.6 (782.8, 110950.4)	38808.6 (676.1, 97042.5)

It might be easier to understand the results of the table using graphs. Figure 4 shows panel E of Figure 5. This graph is presenting the information in Table 1A, row k=3.0. The

header of the graph displays the value of k and the method of estimating the standard deviation. There are six lines displayed, one for each sample size and a control line. Each line is showing the distribution of run length for 10,000 simulations by percentile. For example, the identified point represents the 80th percentile of run length for sample size of 10 (which is approximately 2,000). This means that 80% of the run lengths were less than 2,000 and 20% were longer than 2,000.

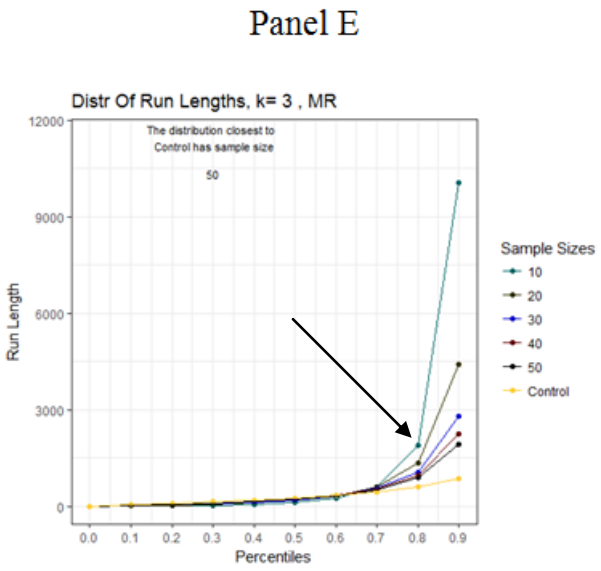
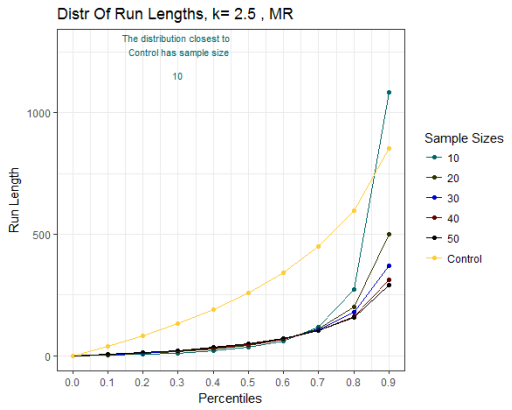


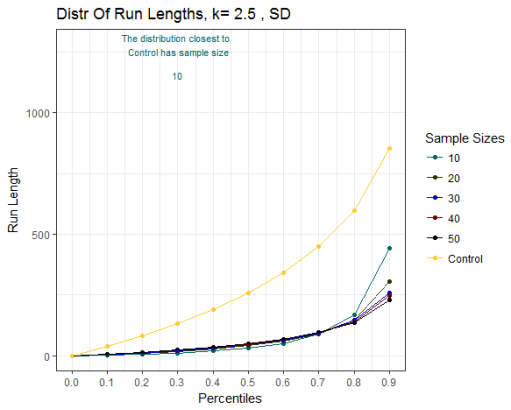
Figure 4. The reference panel of Figure 5.

In general, a process with a shorter run length is more likely to produce false signals. A process with a long run length misses signals. The control lines (the yellow lines) on each graph is the ideal plot. It is based on the theoretical distribution of run lengths and is the same in each of the ten panels. It appears to change but that is due to the vertical axis scaling that changes from graph to graph. In each panel, the sample size closest to the control line is noted. When plots fall below the control plot, there are too many false signals, and when the other plots fall above the control plot, there are not enough signals.

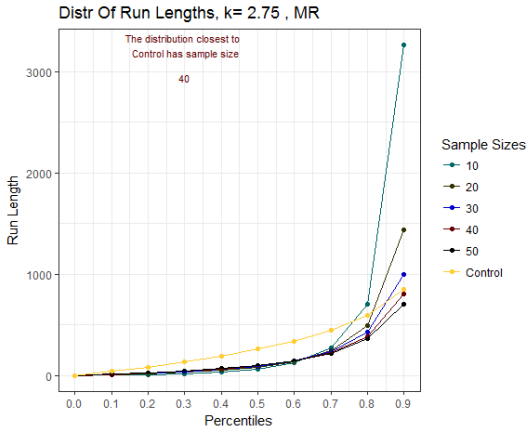
Panel A



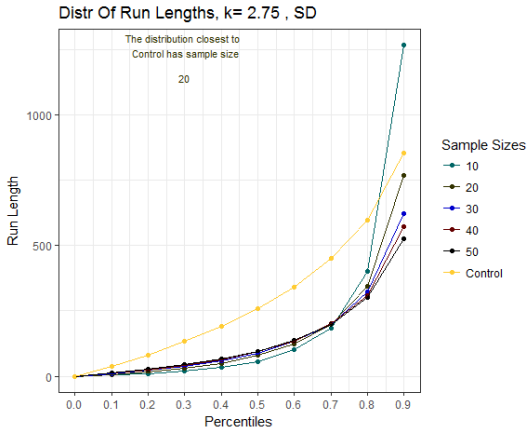
Panel B



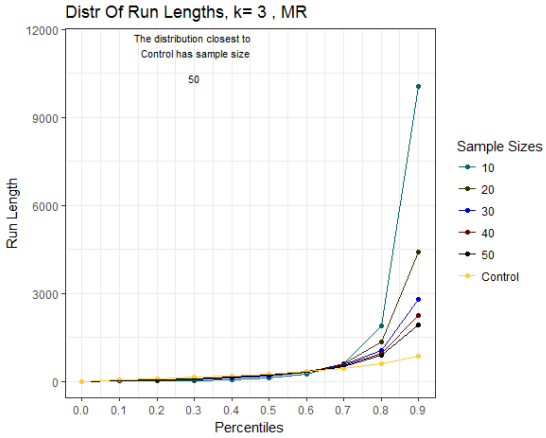
Panel C



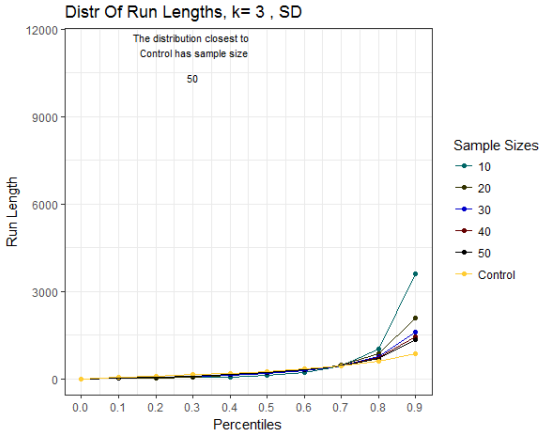
Panel D



Panel E



Panel F



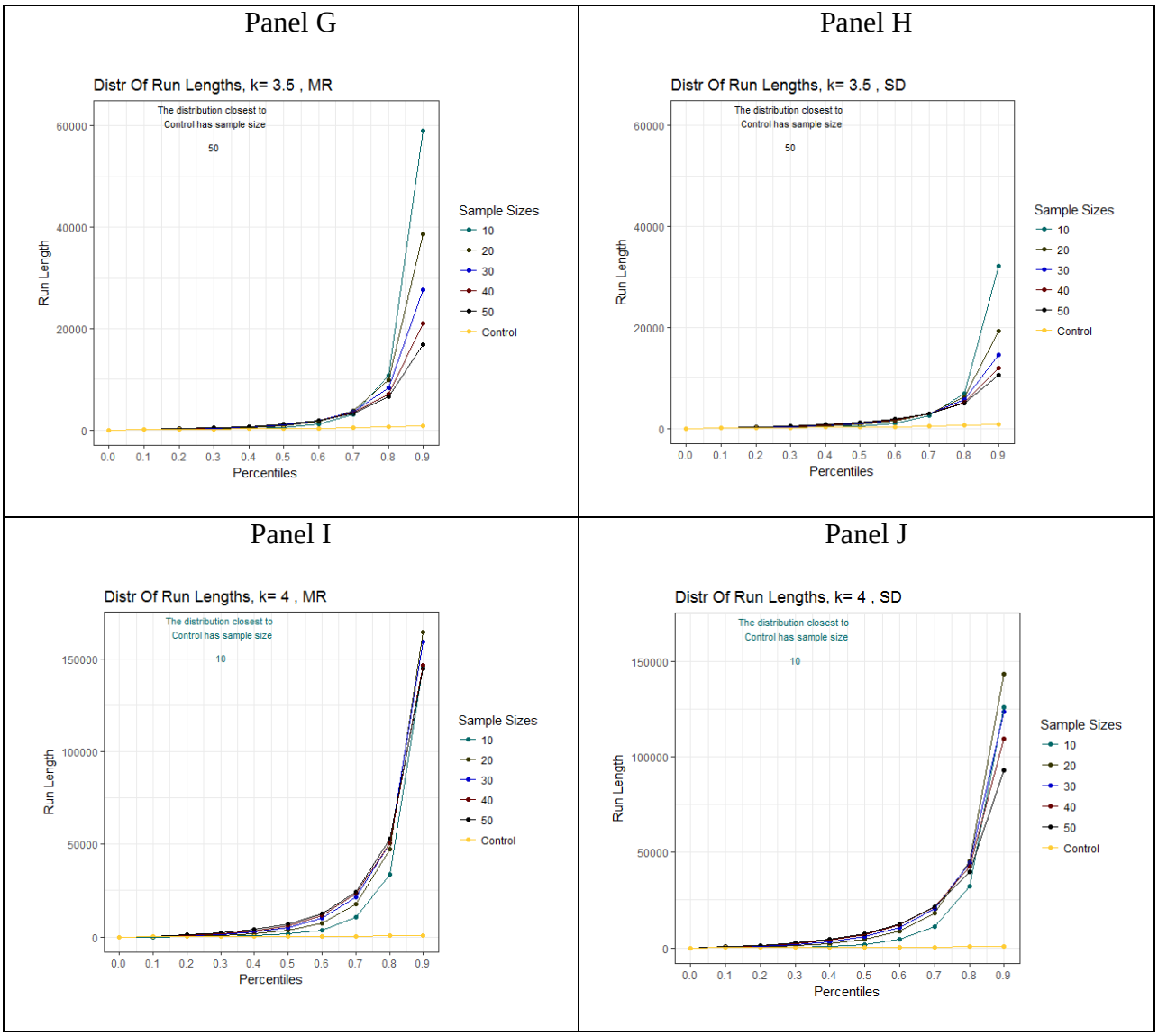


Figure 5. Distribution of Run Lengths Charts. Panel E is the reference panel.

Panel E represents the predominant current industrial practice, as mentioned in the introduction. It has a value of k = 3 and applies the MR method to calculate the control limits. This Panel is the reference point for understanding how to improve the I-Chart for the pharmaceutical manufacturing processes. The methods used to estimate the standard deviation are the Moving Range in the left panels and the corresponding Levey-Jennings method in the right panels. Each row of panels represent a different value of k, starting at the top with k = 2 and

ending at the bottom with k = 4. Now, the different parameters can be compared to determine which method works best for each of the sample sizes.

The distributions line for all sample sizes when k = 2.5 (Panels A and B) were too far below the yellow control line for sample size. Although sample size of 10 is considered the line closest to the control, that is largely due to the point match at 100%. Overall, there are too many signals being detected, indicating a higher probability for false alarms.

The two cases for k = 2.75 in Panels C and D were simulated and added after observing the differences between the cases where k = 2.5 and k = 3.0. The distributions in Panel C were found to be a better fit over the control line for most of the distributions when compared to Panel D; there are just enough signals being indicated in Panel C. However, Panel D had distributions that were still far underneath the yellow control line.

On the contrary, the cases in Panels E and F, where k = 3.0, were close to and just above the yellow control line. Notice that the Panel F distributions fall closer to the control line than those in Panel E.

Lastly, the cases in Panels G and H, where k = 3.5, and in Panels I and J, where k = 4.0, indicate that the distributions are too far above the control line. The distributions in these four cases are out of control. There are not as many signals detected as the ideal case because the control limits are too far away from the mean.

The percentile can be a measurement of the certainty that the I-chart will work. Hence, we want distributions to be as close to the control line in each case and the percentile to be as large as possible, to maximize such certainty. In panels A through H, the distribution lines are nearly coincident up until the 70^{th} percentile, where there is a 70% chance the I-Chart will work.

Before the 70th percentile, sample size does not appear to have an effect. This deviating point on the percentile axis is the place where the ARL varies with sample size and therefore the amount of signals detected varies with the sample size. On the contrary, in Panels I through J all of the distributions stay fairly close to each other along every value of the percentiles demonstrating that sample size is not a factor is using a k = 4. However, since the distributions are all above the control line, they also miss signals.

Notice that the five distributions in Panels A, B, C, D, E, F, G, and H (other than the control) all tend to deviate from each other around the 70th percentile in each of the Panels; the distributions start to go in different directions when all of them get to approximately the 0.7 mark on the Percentile axis of each Panel, in other words. This deviating point on the percentile axis is the place where the amount of signals detected by the control charts starts varies with the sample size. On the contrary, in Panels I and J all of the distributions stay fairly close to each other along every value of the percentiles. To compare the five distributions in each Panel, the higher the last point (in the 90th percentile or set of points at and after the 70th percentile) on the curve, the larger the positive skewness of the distribution. In the skewed distributions the mean is affected, which is the ARL. When the ARL is large, fewer signals are detected.

For example, n = 20 and n = 50 in Panel A, (when k = 2.5 and MR method is used) the n = 20 case has larger 80th percentile in run length L than n = 50. This means that the batch size n = 20 produces fewer signals than the n = 50 distribution. However, more signals are still being produced than the yellow control line for both n = 20 and n = 50.

At the 90th percentile, if the n = 10 and n = 50 distributions in Panels E and F are compared to each other, the n = 10 distribution has a longer L than the n = 50 distribution. Hence, the n = 10 distribution produces fewer signals than the n = 50 distributions even though

both distributions produce fewer signals than the control. Lastly, notice that in Panels C and D, the n = 10 distributions is above the n = 50 distribution, so therefore the former's L is longer and produces fewer signals than the latter. Also, notice that in both of these cases, the n = 10 distribution is above the control and the n = 50 is below the control, so the former produces fewer signals than the control and the latter produces more signals than the control.

Conclusions and Further Research

There are plenty of observations that will be useful to pharmaceutical manufacturers' operations improvement. In general, it may be ideal to have run lengths greater than that of the known case (the yellow control line above), which means that the points settle above the gold curve in panels A through J. However, at the same time, the run lengths that are too much greater than the known case indicate limits that are too wide to sufficiently control process output for meeting quality requirements. As the 90^{th} percentile of the observed run lengths for the known case is about 700 observations per batch (see panel A), this means that our optimal arrangement should produce run lengths of at most 1,000 observations at the 90^{th} percentile.

The default settings that pharmaceutical manufacturers use currently are shown in panel E, the chart that uses the MR method and k = 3.0. For the sample sizes shown it can be seen that for at least 15% of run lengths, the run length is 1,500 observations or more. In contrast, Panel F shows that only 10% of run lengths produce run lengths of this magnitude. Therefore, the Levey-Jennings methodology may be deemed to yield better chart performance than the current practice based on the MR approach, particularly for sample sizes of 30 or more.

However, for smaller sample sizes of 20 or less there is no ideal solution; instead, one is faced with choosing to operate with one of two risks. To eliminate the risk of producing limits

that are too wide, the Levey-Jennings methodology using k=2.7 (when n=10) or k=2.8 (when n=20) is recommended (see Panel D, 90th percentile values). But, as seen in Panel D, the majority of run lengths for all sample sizes are much less than those for the known case, meaning many more signals will be produced under this scenario. To minimize the number of false signals, the Levey-Jennings methodology using k=3.0 is recommended (Panel F). This poses some risk of producing limits that are too wide for the process, particularly for sample size n=10 (see 90th percentile), but there is no better alternative.

There might be some problems with these results when applied in pharmaceutical manufacturing. Recall that the Phase I and Phase II data for the simulation took random samples from the standard normal distribution. The assumption of process normality may not always be the case in drug manufacturing facilities; the actual probability distribution of the process may be skewed, or it may be a mixed distribution. Therefore, in further research, the goal will be to determine limits for non-binomial probability distributions, by modifying the formulas and code used to simulate the Phase I and Phase II datasets in this paper.

Appendix

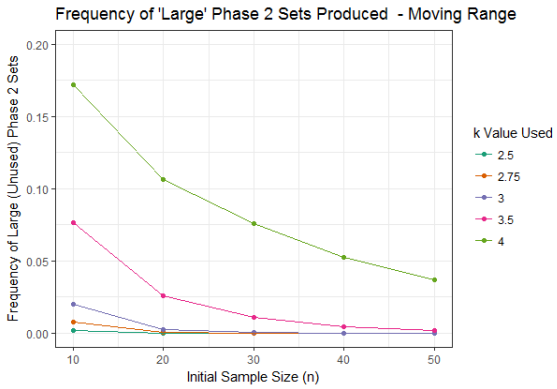
Written by Matt Perkins, the head programmer in this research project.

During the initial process of running simulations, a couple of issues were faced which required adjustments to the R code that was in use at that time. The first issue was that the code was generating phase two sets which contained over 20 million elements, greatly increasing the run time of a given simulation. The other issue was that the simulations were designed to run until a user-specified standard error to mean ratio was obtained, but there was uncertainty in how this

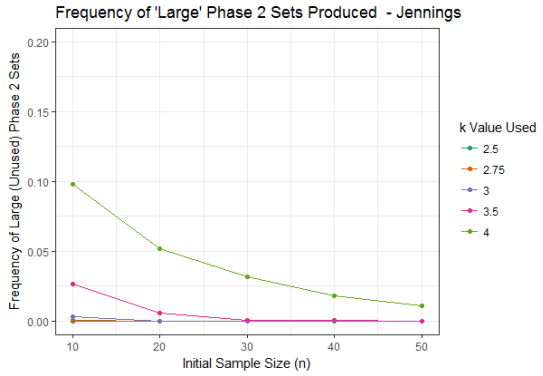
user-specified value should be determined. Changes in the original methodology were made to correct these issues.

The code was first edited to terminate the process of generating any phase two set which exceeded a length of one million elements. If the size of any phase II set reached one million elements, the simulation would abandon this phase II set and begin generating a new phase one and phase II set. It was tabulated how many times this happened in relation to the total number of phase II sets whose generation was attempted during a simulation. These tabulations can be seen in the plots below.

Plot 1: This plot shows the frequency of simulations abandoned during a given Moving Range simulation. Each coordinate for a given line corresponds to the initial sample size of the phase one set generated. Each line corresponds to the value of k that was used to compute the control limits for a given simulation.



Plot 2: This plot shows the frequency of simulations abandoned during a given Levey Jennings simulation. Each coordinate for a given line corresponds to the initial sample size of the phase one set generated. Each line corresponds to the value of k that was used to compute the control limits for a given simulation.



Once enough code was developed to begin running simulations, the next issue faced was when to stop the simulations and tabulate the results. It was planned that this would take place when the standard error to ARL ratio fell below a specified value, but it had not yet been determined what this value should be. The distribution of the standard error to ARL ratio was analyzed over the number of phase two sets that were generated throughout the simulation. Figure 2 of this can be found on page 15. In Figure 3, it can be seen that as the number of phase two sets generated increases, the standard error to ARL ratio approaches a horizontal asymptote. An algorithm for estimating the value of this asymptote was generated and can be found on page 15.

The algorithm was applied for each sample size and k value up to four times. For higher values of k and smaller sample sizes (ex. {k,n}={4,10}), the algorithm was only applied twice due to those simulations requiring excessive computing time. For each value of k and sample size, the smallest of the four algorithm results was taken to be the final value used to terminate a given simulation. Tables 3, 4, and 5 containing the values can be found below.

Table 3: The Levey-Jennings Method Results

	2.5	2.75	3	3.5	4
10	0.051	0.08	0.058	0.037	0.025
	0.065	0.061	0.089	0.037	0.025
	0.058	0.078	0.063	0.036	0.025
	0.065	0.1	0.062	0.037	0.026
20	0.021	0.032	0.054	0.073	0.031
	0.024	0.028	0.05	0.046	0.031
	0.017	0.025	0.045	0.047	0.03
	0.019	0.027	0.033	0.044	0.031
30	0.015	0.017	0.028	0.04	0.034
	0.017	0.017	0.024	0.036	0.03
	0.014	0.021	0.023	0.043	0.033
	0.015	0.017	0.022	0.068	0.033
40	0.012	0.013	0.018	0.028	0.032
	0.012	0.014	0.017	0.029	0.029
	0.016	0.014	0.017	0.029	0.033
	0.012	0.014	0.017	0.035	0.036
50	0.015	0.017	0.014	0.024	0.03
	0.015	0.017	0.014	0.021	0.029
	0.015	0.012	0.013	0.02	0.03
	0.014	0.013	0.014	0.025	0.033

Table 4: The Moving-Range Method Results

	2.5	2.75	3	3.5	4
10	0.086	0.052	0.041	0.031	0.022
	0.075	0.055	0.046	0.031	0.027
	0.072	0.054	0.047	0.032	0.027
	0.069	0.059	0.045	0.031	0.027
20	0.055	0.069	0.064	0.039	0.025
	0.051	0.081	0.072	0.037	0.025
	0.046	0.156	0.13	0.04	0.025
	0.046	0.078	0.066	0.038	0.031
30	0.028	0.056	0.067	0.048	0.028
	0.031	0.034	0.087	0.043	0.029
	0.021	0.038	0.066	0.044	0.028
	0.021	0.037	0.047	0.047	0.028
40	0.023	0.035	0.04	0.071	0.033
	0.018	0.024	0.035	0.043	0.032
	0.016	0.023	0.057	0.043	0.033
	0.017	0.065	0.044	0.049	0.032
50	0.014	0.024	0.022	0.046	0.033
	0.013	0.017	0.023	0.057	0.033
	0.02	0.016	0.028	0.06	0.033
	0.018	0.018	0.02	0.086	0.034

Table 5: The Control Results

10	20	30	40	50
0.0101	0.01023	0.01005	0.01017	0.01014
0.0102	0.01002	0.00998	0.01029	0.00993
0.0101	0.01006	0.01012	0.01018	0.01019
0.0101	0.01012	0.01032	0.01001	0.01019

References

Gardner, Jeff. (2015). How Type I Error Impacts Quality System Effectiveness. *Pharmaceutical Engineering*. *35*(1), 1-3.

Karkalousos, P. & Evangelopoulos, A. (2011). Quality Control in Clinical Laboratories. In O. Ivanov (Ed.), *Applications and Experiences of Quality Control* (pp. 331-360). Retrieved from http://www.intechopen.com/books/applications-and-experiences-of-quality-control/quality-control-inclinical-laboratories

Milivojevich, A. (2014, August 11). Range Statistics and d2 Constant – How to Calculate Standard Deviation. Retrieved from https://andrewmilivojevich.com/range-statistics/

Montgomery, D. C. (2009). Control Charts of Variables. In J. Welter & S. Dumas & L. Sapira (Eds.), *Introduction to Statistical Quality Control* (pp. 226-275), (Vol 1, 6th ed.). Jefferson City, MO; John Wiley & Sons, Inc.

Peng, D. Y. & Lionberger, R. & Viehmann, A. & Iyer, K. & Yu, L. X. (2015). Using Control Charts to Evaluate Pharmaceutical Manufacturing Process Variability. *Pharmaceutical Engineering*; *35*(3), 59, 61-64, 67, 70-71, 74.

Perry, J. (2016). *Shewhart or levey-Jennings control chart* [infographic]. Retrieved from http://slideplayer.com/slide/7828843/25/images/15/Shewhart+or+Levey-+Jennings+control+charts.jpg

Quesenberry, C. P. (1993). The Effect of Sample Size on Estimated Limits for $\bar{X}$ and X Control Charts. *Journal of Quality Technology*, *25*(4); 237-247

Acknowledgements

Thank you Jeff Gardner, the President of DataPharm and principle consultant, for helping me through this research project and giving me advice and criticism when needed. I would like to also thank my coworker Matt Perkins and colleague at DataPharm for helping to run all of the code and for helping me out when I got stuck on this research paper. I have great thanks for Dr. Linda Quinn's help and support with this McNair Research project, as well as the McNair Scholars Program staff at Cleveland State University for giving me the support and resources that I needed to complete this research project.

Reflections on this Project

By doing this McNair research project this summer, I learned quite a lot about researching, applying and going to graduate school, about what a job as a statistician is really like, and about my own areas that I succeed at and that I can improve on. Researching takes plenty of patience and persistence, as well as good organization and communication skills. Although I did get frustrated in the middle of the research process, especially when developing the code and when running the simulations, I learned that as long as you keep trying and waiting, everything will eventually get done. These lessons I learned will carry on with me into my senior project and graduate school research projects.

As I was proving to myself that I can complete scientific research, I also found out this summer that this research will help me greatly in graduate school and beyond. This research project and the projects to come will help me demonstrate to graduate schools and PhD programs, and future employers, that I am capable of researching and organizing the process of research. This summer, along with conducting research, I am very thankful to have shadowed

two professional statisticians: Jeff Gardner and Dr. Linda Quinn. Jeff is a private statistical consultant for mainly pharmaceutical companies, and Dr. Linda Quinn does statistical consulting and teaches statistics to CSU undergraduates and graduates. I learned that the work of statisticians, in general, can be quite stressful, but it is very rewarding and offers many other job benefits.

I learned this summer, what steps I need to take to become a statistician. I came into this summer after taking two basic statistics classes at CSU. Now, as I return in the fall as a senior, I'll take many more applied mathematics, statistics, and programming classes. I'll do this same when I work on my first master's degree (after graduating) and eventually in my PhD program someday. These classes will give me the programming, technical writing, analytical, and problem solving skills that I will need to become a successful statistician someday.

R code from this Project

Experimental Simulation Code

```
genControlLimits <- function(n1, varcount=1, whichMethod, k)  {
 # this function:
 # - randomly generates a "phase one" data set of one variable having n observations
 # - calculates the upper and lower control limits for each phase 1 data set using specified method (MR or SD)
 # - returns I-chart control limits
 phase1 <- matrix(rnorm(n1*varcount, 0, 1), nrow = n1, ncol = varcount)    #generate the phase1 data
 mean_var <- apply(phase1, 2, mean)    #calculate the mean and std deviation of the phase1 data set
 std_var <- apply(phase1, 2, sd)
 if(whichMethod=="SD") {    #jennings method
   Upper_SPC_Limit <- mean_var + k * std_var   #calculate control limits using std dev method
   Lower_SPC_Limit <- mean_var - k * std_var   #control limits are +/- 3
 }
 if(whichMethod=="MR") {    #moving range method
   moving_range <- apply(phase1, 2, function(z) abs(diff(z)))
   MR_mean <- apply(moving_range, 2, mean)
```

```
   Upper_SPC_Limit <- mean_var + k * 0.8865 * MR_mean    #calculate control limits using moving range method
   Lower_SPC_Limit <- mean_var - k * 0.8865 * MR_mean    # control limits are +/-
  }
  if(whichMethod=="CONTROL") {    #use control SPC limits
   Upper_SPC_Limit <- 3   #calculate control limits using std dev method
   Lower_SPC_Limit <- -3   #control limits are +/- 3
  }
  return(c(Upper_SPC_Limit,Lower_SPC_Limit))  #return the control limits in a vector
}
##################################################################################################################
#########
genPhase2Set <- function(upperCL, lowerCL) {
  #This function
  # - takes in the upper and lower I-chart control limits
  # - randomly generates a "phase two" data set until a generated value falls outside the control limit range
  # - returns the run length. If the "phase two"  set reaches a run length of 1,000,000 , it will stop generating the set
  #   and return the size of 1,000,000
  exc <- 0     #flag signaling when a value falls outside the control limit range
  runLength1 <- 0    #variable holding our run length of the "phase two" set
  while(exc==0) {   #while flag is set to 0, randomly generate elements of the set
    phase2 <- (matrix(rnorm(1, 0, 1), nrow = 1, ncol = 1))
    if(phase2 > upperCL || phase2 < lowerCL) {
      exc <- 1    #if random value falls outside of control limit range, set flag to 1
    }
    if (exc == 0)   #if flag is not set, increment number of run lengths
    {
   runLength1 <- runLength1 + 1
      if(runLength1==1000000)   #break out of the loop if more than 1,000,000 points are generated in a set to reduce run time
        return(runLength1)
    }
  }
  return(runLength1)
}
##################################################################################################################
#########
getChangeInPrecision <- function(ratioOfStdErrorToMean,lastError,numRunLengths,lastNumRunLengths) {
  #this function can be used to track the change in the standard error to mean ratio
  precisionDecider1<-c()
  precisionDecider2<-c()
```

```
 if(numRunLengths==10000) {
  precisionDecider1<-c(numRunLengths,stdErrorToMeanRatio)
  return(100)
 }
 if(numRunLengths>10000) {
  precisionDecider1<-c(lastNumRunLengths,lastError)
  precisionDecider2<-c(numRunLengths,stdErrorToMeanRatio)
  precisionDecider<-precisionDecider2-precisionDecider1
  return(precisionDecider[2])
 }
}
##############################################################################################
quesenberry <- function (n1, k, varcount=1, whichMethod,precision=.05) {
 #this function:
 # - takes in a desired phase 1 sample size (n=10,20,30,40,50), k value (k=3,3.5,4,4.5), variable count (for later use), either
 #   "SD" or "MR" for the desired method of calculating control limits, and a threshold to specify desired level of precision
 # - uses the sample size, k value, variable count, and method of choice to call the genControlLimits function to generate a phase 1 set
 #   and calculate a set of control limits
 # - uses the output of genControlLimits to call the genPhase2Set; generates a phase 2 set until an element of the set falls outside
 #   of the control limits or the size of the set reaches 1,000,000
 # - stores the size of the generated set into a vector if the size is less than 1,000,000; if the size is 1,000,000 , throw out that set.
 #   tabulate how many times this happens so that the percentage of data not used can be calculated
 # - after the size of a set is stored, calculate desired statistics, including mean and standard error, compare the ratio of standard error
 #   to mean with the specified threshold. if the ratio is less than the threshold, return the calculated statistics in a data frame. otherwise,
 #   repeat the process
 i <- 1   #counter
 runLength <- c()
 stdErrorToMeanRatio <- 100
 bigCounter<-0          #count number of times a large sample is generated
 repeat{  #loop will generate phase 2 run lengths until the specified precision value is met
  controlLimits <- genControlLimits(n1,varcount,whichMethod,k)   #generate a "phase one" data set and extract calculated control limits
  tempRunLength<-genPhase2Set(controlLimits[1],controlLimits[2])   #generates a phase 2 data set and records its
                                                # run length in a temp variable
  if(tempRunLength==1000000)              #if the run length reached a million points,
   bigCounter<-bigCounter+1              #do not store information in run length array
                                #increment our large data set counter
  else                                  #if phase 2 data set size did not reach 1 million
   runLength <- c(runLength,tempRunLength) #store information with our other run lengths
```

```
  if(length(runLength)!=0)                #if we have collected the length of a phase 2 set
  {                                  # calculate and update running statistics
    error <- sd(runLength)/sqrt(length(runLength))
    stdErrorToMeanRatio <- error/mean(runLength)
  }
  if(length(runLength)>2 && stdErrorToMeanRatio<precision)        #once the ratio of standard error to mean falls below the specified threshold
    break;                           # terminate the loop and return desired statistics
  i <- i+1   #increment simulation counter and repeat
 }  #end simulation
 ###Print final information
 print(c("for {n,k} =",n1,k))
 print(c("threshold of precision specified = ", precision))
 if(whichMethod=="MR")
  print(c("method of calculation = moving range method"))
 if(whichMethod=="SD")
  print(c("method of calculation = jennings method"))
 print("-------------------------------------")
 print(c("average run length = ", mean(runLength)))
 print(c("standard deviation = ",sd(runLength)))
 print(c("standard error = ", error))
 print(c("standard error / mean = ", stdErrorToMeanRatio))
 print("-------------------------------------")
 print(c("large simulations (%) = ",bigCounter/i*100))
 print(c("maximum run length = ", max(runLength)))
 print("-------------------------------------")
 print("#############################################################################")
 #store information about run lengths in a data frame so that information can be extracted
 finalInfo <- data.frame(n1,k,whichMethod,precision,mean(runLength),sd(runLength),
                    error,stdErrorToMeanRatio,bigCounter/i*100,max(runLength))
 colnames(finalInfo)<-c("n","k", "method", "specified threshold", "ARL","standard deviation",
                   "standard error", "standard error/mean", "large simulations (%)", "max run length")
 return(list(runLength,finalInfo))   #the distrubution can be analyzed
}
```

Precision Simulation Code

```
genControlLimits <- function(n1, varcount=1, whichMethod, k)  {
```

```
 # this function:
 # - randomly generates a "phase one" data set of one variable having n observations
 # - calculates the upper and lower control limits for each phase 1 data set using specified method (MR or SD)
 # - returns I-chart control limits
 phase1 <- matrix(rnorm(n1*varcount, 1, 1), nrow = n1, ncol = varcount)    #generate the phase1 data
 mean_var <- apply(phase1, 2, mean)    #calculate the mean and std deviation of the phase1 data set
 std_var <- apply(phase1, 2, sd)
 if(whichMethod=="SD") {    #jennings method
   Upper_SPC_Limit <- mean_var + k * std_var   #calculate control limits using std dev method
   Lower_SPC_Limit <- mean_var - k * std_var
 }
 if(whichMethod=="MR") {    #moving range method
   moving_range <- apply(phase1, 2, function(z) abs(diff(z)))
   MR_mean <- apply(moving_range, 2, mean)
   Upper_SPC_Limit <- mean_var + k * 0.8865 * MR_mean    #calculate control limits using moving range method
   Lower_SPC_Limit <- mean_var - k * 0.8865 * MR_mean
 }
 return(c(Upper_SPC_Limit,Lower_SPC_Limit))  #return the control limits in a vector
}
##########################################################################################################################
#########
genPhase2Set <- function(upperCL, lowerCL) {
 #This function
 # - takes in the upper and lower I-chart control limits
 # - randomly generates a "phase two" data set until a generated value falls outside the control limit range
 # - returns the run length. If the "phase two"  set reaches a run length of 1,000,000 , it will stop generating the set
 #   and return the size of 1,000,000
 exc <- 0     #flag signaling when a value falls outside the control limit range
 runLength1 <- 0    #variable holding our run length of the "phase two" set
 while(exc==0) {   #while flag is set to 0, randomly generate elements of the set
   phase2 <- (matrix(rnorm(1, 0, 1), nrow = 1, ncol = 1))

   if(phase2 > upperCL || phase2 < lowerCL) {
     exc <- 1    #if random value falls outside of control limit range, set flag to 1
   }
   if (exc == 0)   #if flag is not set, increment number of run lengths
   {
  runLength1 <- runLength1 + 1
```

```
    if(runLength1==1000000)   #break out of the loop if more than 1,000,000 points are generated in a set to reduce run time
      return(runLength1)
   }
  }
 return(runLength1)
}
#########################################################################################################################
#########
getChangeInPrecision <- function(stdErrorToMeanRatio,lastError,numRunLengths,lastNumRunLengths) {
 #this function can be used to track the change in the standard error to mean ratio
 precisionDecider1<-c()
 precisionDecider2<-c()
 if(numRunLengths==10000)
  return(100)
 if(numRunLengths>10000) {
  precisionDecider1<-c(lastNumRunLengths,lastError)
  precisionDecider2<-c(numRunLengths,stdErrorToMeanRatio)
  precisionDecider<-precisionDecider2-precisionDecider1
  return(precisionDecider[2])
 }
}
#########################################################################################
quesenberryPrec <- function (n1, k, varcount, whichMethod,precision=.05) {
 #this function:
 # - takes in a desired phase 1 sample size (n=10,20,30,40,50), k value (k=3,3.5,4,4.5), variable count (for later use), either
 #   "SD" or "MR" for the desired method of calculating control limits, and a threshold to specify desired level of precision
 # - uses the sample size, k value, variable count, and method of choice to call the genControlLimits function to generate a phase 1 set
 #   and calculate a set of control limits
 # - uses the output of genControlLimits to call the genPhase2Set; generates a phase 2 set until an element of the set falls outside
 #   of the control limits or the size of the set reaches 1,000,000
 # - stores the size of the generated set into a vector if the size is less than 1,000,000; if the size is 1,000,000 , throw out that set.
 #   tabulate how many times this happens so that the percentage of data not used can be calculated
 # - after the size of a set is stored, calculate desired statistics, including mean and standard error, compare the ratio of standard error
 #   to mean with the specified threshold. if the ratio is less than the threshold, return the calculated statistics in a data frame. otherwise,
 #   repeat the process
 i <- 1   #counter
 runLength <- c()
 stdErrorToMeanRatio <- 100
```

```
 bigCounter<-0          #count number of times a large sample is generated
 stdErrorToMeanVector <- c()
 precPlotVectorX<-c()
 precPlotVectorY<-c()
 repeat{  #loop will generate phase 2 run lengths until the specified precision value is met
  controlLimits <- genControlLimits(n1,varcount,whichMethod,k)   #generate a "phase one" data set and extract calculated control limits
  tempRunLength<-genPhase2Set(controlLimits[1],controlLimits[2])   #generates a phase 2 data set and records its
                                                # run length in a temp variable
  if(tempRunLength==1000000)               #if the run length reached a million points,
   bigCounter<-bigCounter+1               #do not store information in run length array
                                  #increment our large data set counter
  else                                    #if phase 2 data set size did not reach 1 million
   runLength <- c(runLength,tempRunLength) #store information with our other run lengths
  if(length(runLength)>1)                 #if we have collected the length of a phase 2 set
  {                                       # calculate and update running statistics
   error <- sd(runLength)/sqrt(length(runLength))
   stdErrorToMeanRatio <- error/mean(runLength)
   stdErrorToMeanVector<- c(stdErrorToMeanVector,stdErrorToMeanRatio)
  }
  #if(length(runLength)>2 && stdErrorToMeanRatio<precision)        #once the ratio of standard error to mean falls below the specified
threshold
   #break;                                 # terminate the loop and return desired statistics
  #this block determines when the change in the standard error to mean ratio between two points (x,stderr1/mean1,x+10000,stderr2/mean2)
  #drops below .005 and returns (stderr1/mean1)
  if(length(runLength)>999 && length(runLength)%%10000==0)
  {
   if(length(runLength)==10000)
   {
    tempNewError<-stdErrorToMeanRatio
    tempNewRunLength<-length(runLength)
   }
   precPlotVectorX<-c(precPlotVectorX,length(runLength))
   precPlotVectorY<-c(precPlotVectorY,stdErrorToMeanRatio)
   changeInPrecision<-abs(getChangeInPrecision(stdErrorToMeanRatio,tempNewError,length(runLength),tempNewRunLength))
   if(changeInPrecision<.005) {   #break out of loop when change in the precision value drops below .005
    precisionCounter<-precisionCounter+1
   }
   if(changeInPrecision>.005)
```

```
      precisionCounter<-0
    if(precisionCounter==1){
      print("DONE - The change in precision has dropped below .005")
      print("################################################################################")
      print(c("Final Precision Value to be Used = ",tempNewError))
      break
    }
    tempNewError<-stdErrorToMeanRatio
    tempNewRunLength<-length(runLength)
  }
  i <- i+1   #increment simulation counter and repeat
 }  #end simulation
 ###Print final information
 print(c("for {n,k} =",n1,k))
 print(c("threshold of precision specified = ", precision))
 if(whichMethod=="MR")
  print(c("method of calculation = moving range method"))
 if(whichMethod=="SD")
  print(c("method of calculation = jennings method"))
 print("-------------------------------------")
 print(c("average run length = ", mean(runLength)))
 print(c("standard deviation = ",sd(runLength)))
 print(c("standard error = ", error))
 print(c("standard error / mean = ", stdErrorToMeanRatio))
 print("-------------------------------------")
 print(c("large simulations (%) = ",bigCounter/i*100))
 print(c("maximum run length = ", max(runLength)))
 print("-------------------------------------")
 print("################################################################################")
 #store information about run lengths in a data frame so that information can be extracted
 finalInfo <- data.frame(n1,k,precision,mean(runLength),sd(runLength),
                         error,stdErrorToMeanRatio,bigCounter/i*100,max(runLength))
 colnames(finalInfo)<-c("n","k","specified threshold", "ARL","standard deviation",
                        "standard error", "standard error/mean", "large simulations (%)", "max run length"
 return(tempNewError) #return the data frame with our information
}
```