USING STATISTICAL PROCESS CONTROL TO MONITOR INVENTORY ACCURACY

by

KYLE HUSCHKA

A THESIS

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Industrial and Manufacturing Systems Engineering
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2009

Approved by:

Major Professor
John R. English

Abstract

Inventory accuracy is critical for almost all industrial environments such as distribution, warehousing, and retail. It is quite common for companies with exceptional inventory accuracy to use a technique called cycle counting. For many organizations, the time and resources to complete cycle counting are limited or not available. In this work, we promote statistical process control (SPC) to monitor inventory accuracy. Specifically, we model the complex underlying environments with mixture distributions to demonstrate sampling from a mixed but stationary process. For our particular application, we concern ourselves with data that result from inventory adjustments at the stock keeping unit (SKU) level when a given SKU is found to be inaccurate. We provide estimates of both the Type I and Type II errors when a classic *c* chart is used. In these estimations, we use both analytical as well as simulation results, and the findings demonstrate the environments that might be conducive for SPC approaches.

Table of Contents

List of Figures

List of Tables

Acknowledgements

First and foremost, this thesis could not have been written without my advisor Dean John English, who not only served as my supervisor, but helped guide me through my last two years of college. I have learned a lot throughout my time working with Dean English both scholastically and otherwise. It has been an honor and privilege to work beside him, and the experience is something I will remember forever.

I would also like to thank a few people who helped contribute to the content of this thesis. John Miller worked as honors student under Dean English at the University of Arkansas and contributed a great deal in formulating the problem for this thesis. John was also very accommodating by having phone conferences to discuss the work he had done, and all of his help was greatly appreciated. Dr. Todd Easton provided support in the programming areas of this research. Not only was his knowledge very useful in formulating these programs, but Dr. Easton also provided guidance as my advisor throughout my entire college career.

Finally, I would like to thank the remaining members of my committee, Dr. John Boyer and Dr. Shing Chang. Both have served on my committee and been willing to meet several times throughout my Master's studies, and provided valuable feedback from these meetings. All of their input has been very valuable to this thesis.

CHAPTER 1 - Introduction and Objectives

1.1 Background

1.1.1 Inventory Management

Inventory accuracy is a critical concern for almost all industrial environments. Most large and successful retail and warehousing environments desire to maintain accurate perpetual inventory. To achieve accurate inventory, the recorded inventory for stock keeping units (SKUs) must match the actual on-hand inventory. If inventory accuracy is low, there are many problems that can arise. First, there are several non-value added activities that must take place to rectify the discrepancies of the inventory. These tasks include correcting an inventory data base, hurried replacement of product, and any other time taken by employees to correct the inaccurate inventory. Although these costs are reason enough to validate the importance of accurate inventory, perhaps the most significant setback of inaccurate inventory is the loss of customer good will. This occurs if there is insufficient inventory to satisfy the customers demand. Alternatively, time can also be wasted on unnecessary replenishment of inventory.

Cycle counting is a proven method to monitor and check inventory accuracy on a continual basis. Today, cycle counting is being used to replace annual physical inventory checks which require the entire factory or warehouse to be shut down to count all SKUs at one time. Basically, cycle counting is a process in which SKUs that are kept in inventory are counted periodically to examine their accuracy. If discrepancies are found, the reason for the discrepancy is examined, and the inventory records are updated, or inventory adjustments are made. Most methods of cycle counting require counting all SKUs over a period of time to ensure that the actual inventory matches the inventory that is recorded in the entire company's inventory data base. Brooks and Wilson (2004) stated that "through the proper use of cycle counting, inventory record accuracy above 95% can be consistently maintained".

However, the feasibility of cycle counting becomes controversial when considering a very large retail or warehousing environment that contains thousands of SKUs. With such a large environment, the labor requirements to implement cycle counting could become excessive. For this reason, large retail and warehouse environments pose a problem for the implementation of cycle counting.

It has been determined that inventory accuracy is important to companies and that having accurate inventory creates a significant amount of value. However, it has also been established that obtaining this accurate inventory could cost a large amount money and labor. A new, cost effective method for monitoring inventory accuracy for large retail and warehouse environments needs to be developed.

Statistical Process Control (SPC) is an effective statistical approach used to monitor processes and improve quality through variance reduction. Basically, SPC uses sampling to monitor the performance of a process. An advantage of SPC is that it requires reduced resource expenses because it depends on random samples and not 100 percent inspection. It is obvious that the aforementioned problem of finding a cost effective method for monitoring inventory accuracy for large operating units would be a suitable application of sampling approaches. In this case, the inventory accuracy of a warehouse would be measured without 100 percent inspection. There are two specific tools that would be suitable for this environment.

First would be the use of a p chart, which is used to monitor the percent nonconforming of a process. For a p chart, a random sample of n is selected, and each item is examined to determine if it meets the quality requirements. The observed number of defective items is divided by the sample size to provide an estimate of the percent nonconforming, or p. The observed p*'s* are plotted in relationship to time, and their variability is assessed. If non-random behavior is observed, process adjustments are pursued to rectify the problem. A p chart could be used for monitoring inventory accuracy where you would examine the percent of SKUs for which the physical inventory exactly matches the recorded inventory for a sample size n within an operating unit.

The second option is the use of a c chart, which monitors the number nonconformities in a process. For a c chart, an inspection unit of size n is sampled, and the observed number of nonconformities is plotted in relationship to time. Similar to the p chart, the observed c*'s* are analyzed for non random behavior. A c chart could be used for monitoring inventory accuracy to examine the number of inventory adjustments for a given unit, where an inventory adjustment is represented by the number of units an observed SKU is above or below the recorded inventory. The use of a c chart will be the focus of this research.

Historically, SPC and cycle counting are domains that have not been considered to be complementary of one another. If these two methodologies can be incorporated, a cost effective method to accurately monitor inventory for large operating units can be realized. The goal of this research is to show that SPC can be applied to environments needing to monitor and/or improve inventory accuracy. It is the thesis of this work to present the c chart as an efficient alternative to cycle counting for monitoring inventory accuracy.

1.2 Objective

This research focuses on examining a population, which is based off a large retail or warehousing environment, that consists of thousands of SKUs and each SKU can have a different Bernoulli rate, which violates the assumption that all samples come from the same stationary process. It is the premise of this effort that the populations can be modeled by the underlying sub populations that follow a mixture distribution. The objective of this research is to develop and analyze a SPC approach for cycle counting by examining the use of the *c* chart for populations that follow a mixture distribution. The experimental units measured in this *c* chart are the number of inventory adjustments, or the number of units above or below the recorded inventory level for a specific SKU. The number of adjustments made to the inventory records is being measured because inventory adjustment accuracy is often the data that organizations track. This model will be used to inspect the type I and type II error rates of the *c* chart with such populations.

1.2.1 Tasks

The research tasks are summarized as follows:

Task 1: Analytical Model

Sub task 1.1: Develop an analytical model that is applicable for modeling inventory adjustments when the population is a mixture of multiple sub populations of known Poisson distributions.

Sub task 1.2: Use this analytical model to determine type I and type II error rates for the *c* chart based on a useful set of experimental conditions. Furthermore, the analytical results are limited to practically feasible problems; therefore we explore sample sizes of no greater than 5.

Sub Task 1.3: Present observations and conclusions of analytical model.

Task 2: Simulation Model

Sub task 2.1: Develop a simulation model that enables the examination of a mixture distribution on *c* charts for practical sized problems that involve sample sizes much larger than 5.

Sub task 2.2: Use the simulation model to evaluate type I error rates for different values of the parameters of λ and n for the *c* chart.

Sub task 2.3: Present observations and conclusions of simulation model

Task 3: Review summary of the findings of the research and any extensions of the research

CHAPTER 2 - Literature Review

2.1 Introduction

The purpose of this research is to illustrate how an understanding of statistical process control allows for improvements in the area of cycle counting. The first section introduces cycle counting, and the various methods of cycle counting being employed today. The second section gives an overview of statistical process control and more specifically the tools that are necessary for this research.

2.2 Overview of Cycle Counting

A basic definition of cycle counting is counting inventories on a regular basis throughout the year. The use of cycle counting has become increasingly popular and has begun replacing annual physical inventories. Performing an annual inventory requires shutting down the entire factory, store, or warehouse one or more times a year, and conducting wall-to-wall physical inventory. Cycle counting is a method that can replace these annual inventory checks with a process that involves counting a small portion of inventory on an incremental basis with the objective of inspecting each SKU within inventory during a fixed period.

The primary idea of examining portions of inventory with cycle counting involves sampling. In our example sampling is a technique in which certain SKUs within the inventory are selected, and the accuracy of those SKUs is measured. Once this has been done it can be inferred that the accuracy is a characteristic of the entire population. The ability to make this inference, as opposed to surveying the entire population, allows cycle counting to save both time and money when implemented. Tersine (2004) indicates that the primary goals of cycle counting are to:

- Identify the causes of the errors
- Correct the conditions causing the errors
- Maintain a high level of inventory record accuracy
- Provide a correct statement of assets

Also, Brooks and Wilson (2005) state that with the proper use of cycle counting, inventory record accuracy above 95% can be consistently maintained. This section provides an overview of cycle counting and its application and now shifts to look at different methods of cycle counting.

2.2.1 Geographic Method

The first method of cycle counting discussed is the Geographic method. This method is described by Schreibfeder (2005) as starting at one end of the warehouse and counting a certain number of products each day until you reach the other end of the building. This method is the most basic method of cycle counting and results in counting all of your items an equal number of times per year. It is also recommended by Schreibfeder (2005) that if you implement this method that you should try to count each item at least four times per year.

2.2.2 Random Sample Cycle Counting

Another method of cycle counting is called random sample cycle counting. This method is described by Brooks and Wilson (2005) as a technique that selects its sample from the population of SKUs at random, thereby giving every member of the population an equal opportunity to be a member of the counting sample. There are two methods of random cycle counting, one called constant population counting and another called diminishing population counting.

The constant population technique is also known as counting with replacement. In this method, the part numbers that are chosen to be counted on one day will be put back in the population and have a possibility of being counted again on the next day.

Alternatively, the diminishing population technique is similar to counting without replacement. Counting without replacement means that the parts that are selected are not returned to population until all of the other parts have been sampled.

Important characteristics of these two methods are mentioned by Brooks and Wilson (2005). First, constant population cycle counting contains the possibility that some parts may be repeatedly sampled while others may never be sampled. Diminishing population cycle counting assures that all parts are counted, but the regularity with which each part is counted is uneven as the counter cycles through the entire population of parts.

2.2.3 ABC Method

The next method discussed is ABC cycle counting or the Ranking Method. This method is based on the Pareto Principle that there are many contributors to a result, but a minority of the contributors account for a majority of the result. In this method class A items are the most important items, followed by class B items and class C items respectively. This method places emphasis on parts that:

- Account for a majority of the money spent on inventory

- Account for a majority of the volume of product flow

- Have history of a poor accuracy level

Rossetti et al. (2001) state that the ranking method used by each company can be tailored to the specific priorities of the organization. After determining what the company would like to place emphasis on, they determine a sampling plan. For example, if the company wanted to place emphasis on items that account for a majority of money spent on inventory, they would sample roughly the top 20% of items based on money spent at a higher rate than the other items that cost less.

Rossetti et al. (2001) warn that ABC cycle counting has a disadvantage in that the category classifications are primarily based on financial considerations. When considering delaying production or shipments, items in the C category can be just as important as class A items. Thus, it is important to consider lead-time, usage, and bill of material (BOM) level when employing ABC cycle counting. This idea of multiple decision criteria is put into motion as Plaff (1999) illustrates the notion of using multiple criteria to determine the classes for ABC cycle counting. Bergman (1988) presents a multiple criteria weighting system in which each SKU is ranked according to common usage across the BOM, lead-time, method of issue, and number of issues. Finally, Flores and Whybark (1987) extend the ABC analysis based on usage and dollar value to include non-cost factors such as certainty of supply, impact of a stock out, and rate of obsolescence.

2.2.4 Process Control Cycle Counting

The final method of cycle counting is a controversial method called Process Control Cycle Counting that was introduced by Brooks and Wilson (2005). This method has two requirements for its implementation to be feasible. First, inventory records must have piece count by multiple location capability, and second, an inventory record listing of all quantities in all locations must be available to the cycle counter. Once these two requirements have been met the method can be applied. The first step is to assign counters to a specific area. Once this has been done, counters verify every part in every location, but they only count parts that are easy to count. For parts that are difficult to count, the part identification, location, and order of magnitude are checked. If the order of magnitude is way off (e.g., there are 1,000 pieces but inventory says there should only be 50), then the part has to actually be counted.

As mentioned earlier this method is seen as very controversial and has several advantages and disadvantages. The disadvantages include determining when parts are considered "easy-to-count." Many people fear that cycle counters say that all parts are difficult to count so that they do not have to count any parts. According to Brooks and Wilson (2005), the main advantage of Process Control Cycle Counting is that you can count 10 to 20 times the part numbers in a given time without greater cost.

2.2.5 Summary of Cycle Counting

The most obvious reason for implementing cycle counting is measuring inventory accuracy. Despite this well known idea, Brooks and Wilson (2005) propose that the highest value of cycle counting may be its ability to identify errors and the causes of these errors. If you were to perform annual inventory checks, then you will find errors, but would have no idea whether errors occurred a week ago or eleven months ago. The ability to identify causes of discrepancies in inventory is beneficial because these causes can be addressed and fixed so that they do not occur again. Wilson (1995) and Stahl (1998) portray cycle counting as a quality assurance process emphasizing the correcting of errors, which is the type of cycle counting this research wishes to improve upon.

2.3 Statistical Process Control

The use and understanding of statistical process control (SPC) tools is inherent for this work. Montgomery (2006) states that "the field of statistical quality control can be broadly defined as those statistical and engineering methods that are used in measuring, monitoring, controlling, and improving quality". SPC is basically the use of statistical techniques to measure and analyze the variation in processes, and improve the quality of the process through variance reduction. The four basic steps of SPC are measure the process, eliminate undesirable variation from the process, monitor the process, and continuously improve the process. There are seven common tools that are used in SPC: Check Sheets, Pareto Charts, Flow Charts, Cause and Effect Diagrams, Histograms, Scatter Diagrams, and Control Charts. The fundamental and most commonly used tool of SPC is the control chart, and it is the major tool that is tested in this paper.

The control chart was first proposed by Shewhart (1926, 1927) and plots the measurements of a quality characteristic in samples taken from a process versus time or the sample number. Miller (2003) states that all processes are subject to degrees of variation, which explains the basic principal of why the control chart is a useful monitoring technique for almost all processes. The chart has a center line (CL) and upper and lower control limits (UCL and LCL). The center line represents the mean of the process characteristic when in control. The LCL and UCL indicate a range in which practically all of the data points need to be within in order for the quality characteristics to be considered in control. The control limits of a given statistic, say *Y*, are determined as follows:

$$\mathrm{UCL}y = E[Y] + \ L\sqrt{Var[Y]} \qquad (1)$$

$$LCLy = E[Y] - \ L\sqrt{Var[Y]} \qquad (2)$$

In these two equations *E[Y]* is the expected value of the random variable *Y*, *Var[Y]* is the variance of the random variable *Y*, and *L* is the number of standard deviations away from the mean of the underlying distribution.

2.3.1 Variable Control Charts

There are two types of data that can be monitored by control charts: variable and attribute. Variable control charts are used when the quality characteristic can be measured and expressed in numbers (Ex: length, weight, volume, etc). The most commonly used variable control charts are $\overline{X}$, *R*, and *S* charts. The $\overline{X}$ chart is used to monitor the mean of a variable, while the *R* chart is used to monitor the variation of a variable. In cases when a large sample size is used (e.g. n>10), an *S* chart should be used to monitor the variation (Montgomery, 1991). The $\overline{X}$ and *R* charts are efficient at detecting relatively large shifts in processes, but are less proficient at detecting small shifts in processes.

2.3.2 Attribute Control Charts

The second type of data that can be monitored by control charts is attribute data. Attribute control charts are used for product characteristics that can be evaluated with a discrete response (Ex: pass/fail, yes/no, good/bad, number defective). Some commonly used attribute charts are *np*, *p*, *u*, and *c* charts. *P* charts are used to monitor the fraction or percentage of non-conforming relative to a random sample. This chart can be used to measure how many inaccurate inventories exist out of the entire sample. The *p* chart is based on the binomial distribution. Sometimes it is more appealing to deal with the total number of non-conforming rather than the fraction of non-conforming, and this leads to the introduction of the *np* chart. The *np* chart is used to monitor the number of times a condition occurs relative to a constant sample size, n. The c chart is used to monitor the number of non-conformities per inspection unit, where an inspection unit could be one or more SKUs that are inspected for inventory adjustments. The c chart is based on the Poisson distribution. The final chart is the u chart, which is used to examine the average number of non-conformities per inspection unit. For the cycle counting domain that we are examining there are at least two useful charts: p charts for percent accuracy and c charts for monitoring the number of inventory adjustments. This research will focus on attribute charts and specifically their application to monitoring inventory accuracy.

There have been many attempts to find substitutes for the traditional control charts for attributes that can be found in the literature. Quesenberry (1991) examines the use of a "Q chart" for both a binomial and Poisson parameters. The use of the "g chart" and the "h chart" to monitor the total number of defects and average number of defects based on the geometric distribution was also examined by Burke (1992). Chang (2008) considers a Shewhart charting scheme for simultaneous monitoring of multiple measurement gauges used for measuring a product or process attribute. A summary of all of these efforts to find substitutes for traditional control charts can be found in Woodall (1997) and also in Ryan and Schwertman (1997). However, as mentioned earlier for the cycle counting domain that we are examining the two most useful charts will be the p chart and c chart.

2.3.3 Other Applicable SPC and Modeling Approaches

This section of the paper discusses a few SPC or modeling approaches that are applicable to the problem domain being dealt with in this paper. The first such approach is known as short run SPC. Short run SPC is used for any type of system that is characterized by short production runs. For the $\overline{X}$ and R charts, the deviation from nominal should be used instead of the measured variable on the control chart, and this is sometime referred to as the DNOM control chart.

Montgomery (2005) states that two important points should be made relative to the DNOM approach:

1. An assumption is that the process standard deviation is approximately the same for all parts.

2. Procedure works best when the sample size is the same for all part numbers.

Montgomery (2005) also mentions that a variety of other approaches can be applied to the short-run production environment. For example, the CUSUM and EWMA charts have potential applications to short production runs, because they have shorter average run length performance than Shewhart-type charts. Also, Hillier (1969) examines how the number of subgroups used in calculating the trial control limits for Shewhart charts impacts the false alarm rate. In particular, he presented a table of factors to use in setting limits for $\bar{X}$ and R charts based on a small number of subgroups for the case of n = 5.

Another SPC topic that is applicable to this problem is a multiple stream process (MSP). This is defined as a process with data at a point in time consisting of measurements from several individual sources or streams. The most common procedure for monitoring a MSP is the group control chart (GCC), which was introduced by Boyd (1950). Also, Nelson (1986) determines the one sided in control average run length for a process that has s streams.

A method of modeling that is similar to this problem is compound Poisson distributions. As stated in Feller (1970), when considering the equation $S_n = X_1 + \dots + X_N$, when N has a Poisson distribution, and the X_i have a common distribution, then S_n follows a compound distribution. This shows that the compound Poisson distribution has a property that is similar to when the random variables are summed are in our problem, which are explained in detail in the next chapter of the thesis.

This section shows that there are several SPC and modeling approaches that have been developed and are currently practiced that are relevant to the topic of this paper. Namely, short run SPC, group control charts for multiple stream processes, and compound Poisson distributions have been discussed. It is important to note that these approaches have some similarities to the SPC approach being examined in this paper. Specifically, all of these approaches involve combining data from different processes into one control chart. However, it is also important to consider that the approach for this paper deals with a very different problem than the two methods discussed in this section.

2.4 Type I and Type II Errors for Control Charts

The control limits on control charts are typically drawn at 3 standard deviations away from the mean of the underlying distribution that is descriptive of the sample population. Whenever a data point falls outside of these limits it indicates an out of control condition. There are two types of errors associated with control charts: type I and type II. Type I errors (*p'*) are known as a "false positive" and occur when we reject the null hypothesis (H_o) but it is actually true. Plainly speaking, this is when a difference is observed but in truth there is none. In the application of control charts, a type I error is made when the operator concludes the process is out of control when in fact it is in control. Type II errors are known as a "false negative" and occur when we fail to reject H_o but the alternative is actually true. In other words this is the failure to observe a difference when there actually is one. In the application of control charts, a type II error is made when the operator concludes that the process is in control when it is in fact out of control.

The average run length (ARL) is the average number of points that is needed to detect an out of control condition. For control charts the ARL can be calculated using the following equation:

$$ARL = \frac{1}{p'} \tag{3}$$

In this equation, p' is the probability that a point is plotted beyond the control limits. The in-control ARL for a process is simply the inverse of the probability of a type I error. Similarly, the out of control ARL is the inverse of 1-P(Type II error). This out of control ARL is the average number of points needed to detect a process shift when one has occurred. The general formula for computing the probability of a type I and type II error can be seen below:

$$p' = 1 - (P[X_1 \leq UCL \vdots \text{ Ho is True}] - P[X_1 \leq LCL \vdots \text{ Ho is True}]) \tag{4}$$

$$\beta = (P[X_1 \leq UCL \vdots \text{ Ho is not True}] - P[X_1 \leq LCL \vdots \text{ Ho is not True}]) \tag{5}$$

To summarize, large in-control ARLs and small out of control ARLs are desired. ARL calculations can become quite complicated as documented in the literature. For example, Champ and Woodall (1987) model control charts using a Markov chain approach to evaluate the ARL when supplementary run rules are applied. Davis and Woodall (2002) present a model of the synthetic control chart and use it to evaluate the zero-state and steady-state ARLs.

2.4.1 Variable Control Chart Robustness

This section examines the aforementioned variable control charts in the literature, and particularly how ARL has been used to evaluate the performance of each one of the different types of variable control charts mentioned.

2.4.1.1 $\bar{X}$ Chart

The $\bar{X}$ chart is used to monitor the means of successive samples. Based off equation (4) the type I and type II error rates for the $\bar{X}$ chart can be computed using the following calculation:

$$p' = 1 - \left(P\left[X_1 \leq \mu + 3\left(\frac{\sigma}{n}\right)\right] - P\left[X_1 \leq \mu - 3\left(\frac{\sigma}{n}\right)\right]\right) \tag{6}$$

Where μ can be estimated by $\hat{\mu} = \bar{\bar{X}}$ and σ can be estimated by $\hat{\sigma} = \frac{\bar{R}}{d_2}$ where $\bar{R}$ is the average range of the sample and d_2 is a value that depends on the subgroup size and can be looked up in a statistical table. Also, X_1 represents a random sample taken from the specified population.

Crowder (1987) and Lucas and Saccucci (1990) look at using the EWMA control chart instead of the $\bar{X}$ chart, as well as, how to evaluate different methods of computing the ARL properties of EWMA control charts. Burroughs (2003) studies the effect of using run rules on $\bar{X}$ charts and determines that they improve the sensitivity of the charts, but cause the chart performance to differ more from the known parameter. As mentioned earlier Champ and Woodall (1987) use Markov chains to compute the ARLs for the $\bar{X}$ chart, but their methods are general and can also be applied to other control charts. Finally, Costa (1999a) and Costa (1990b) extend the study of the $\bar{X}$ chart with variable parameters to include variable action limits.

2.4.1.2 R and S Chart

As mentioned earlier, R and S charts are used to monitor the variation of the variable being observed. Chen (1998) studies the effects of parameter estimation on the R and S charts. Specifically, he looks at the run lengths when the standard deviation of each chart is estimated. Branching off a previous article, Costa (1999a) shows how joint $\bar{X}$ and R charts with varying sample sizes and variable intervals improves the control chart performance in terms of the speed with which shifts in the process mean and/or variance are detected. Coates (1998) examines the effect that rounding numbers has on the control limit factors and type I risks.

2.4.2 Attribute Control Chart Robustness

This section provides a review of the most common attribute control charts and discusses recent literature dealing with modifications to these charts. As mentioned earlier, Woodall (1997) provides an excellent review of attribute control charts and provides bibliographies of over 250 papers on this application.

2.4.2.1 P Chart and NP Chart

The *p* chart monitors the fraction of nonconforming products, while the *np* chart monitors the number of times nonconformity occurs. Using equation (4), the resulting error rates (type I or type II) of the *p* chart can be determined as follows:

$$p' = 1 - \left(P\left[X_1 \leq \bar{p} + 3\sqrt{\frac{\bar{p}(1-\bar{p})}{n}} \right] - P\left[X_1 \leq \bar{p} - 3\sqrt{\frac{\bar{p}(1-\bar{p})}{n}} \right] \right) \tag{7}$$

In this equation $\bar{p}$ can be estimated by $\hat{p} = \bar{p}$. Once again, X_1 represents a random sample taken from the population being examined.

The plotting of *p* and *np* charts by a computer program was first designed by Larson (1969). There have been a large number of papers on using the cumulative sum (CUSUM) methods to detect shifts in the fraction of nonconforming. For a given in-control ARL, the CUSUM methods have lower ARLs than the corresponding Shewhart-type charts for small to moderate shifts in the parameter. Specifically, Gan (1993) proposes a CUSUM chart that is intended for detecting upward shifts in the fraction nonconforming and also provides a way to determine the optimal parameters for the model considering a fixed in-control ARL.

More recently Luceno and Puig-Pey (2000) design an algorithm to compute the ARL probability distributions for CUSUM charts, and eventually Luceno and Puig-Pey (2002) develop this algorithm into a fast and accurate computer algorithm. Alternatively, Gan (1990) studies the exponentially weighted moving average (EWMA) control chart for Binomial data when a small shift is of interest. Additionally, Khoo (2004) develops a moving average control chart for fraction of nonconforming that is more sensitive than the p chart at detecting out of control signals. Chen (1998) uses arcsine and integral transformations to adjust the control limits and improve the performance of the *p* chart. Ryan and Schwertman (1999) examine the use of dual np charts to detect shifts in the control limits. This method would be particularly useful when there is a restriction on the maximum sample size. Ryan and Schwertman (1999) suggest using one chart to provide an early warning of quality deterioration, and the other chart to have approximate normal theory properties. Finally, Nelson (1997) looks at using supplementary runs tests for np control charts that improve the detection of a reduction in proportion non-conforming.

2.4.2.2 C Chart and U Chart

The c chart measures the number of non-conformities per inspection unit, while the *u* chart is used to examine the rate of non-conformities, or the number of non-conformities divided by the number of units inspected. Once again, using equation (4), the type I and type II error rates for the *c* chart can be computed as follows:

$$p' = 1 - \left(P\left[X_1 \leq \bar{c} + 3\sqrt{\bar{c}}\right] - P\left[X_1 \leq \bar{c} - 3\sqrt{\bar{c}}\right]\right) \tag{8}$$

Where $\bar{c}$ can be estimated by $\hat{\lambda} = \bar{c}$, and X_1 represents a random variable from the population being monitored.

McCue (1976) looks at using the power and probability of a type I error using only an upper control limit chart for the number of defects which uses *a*-sigma limits, where the multiplier "a" may be different from 3. The application of the CUSUM method to detect shifts in the number of nonconforming has also been examined. White (1997) suggests procedures for constructing a Poisson CUSUM chart that is designed to distinguish between an "acceptable" mean rate and a "detectable" mean rate. Additionally, Vardeman and Ray (1985) present tables of ARLs for an exponential case and comment on the application of exponential CUSUM charts for controlling the intensity of a Poisson process.

EWMA charts have also been used to detect shifts in the number of nonconforming. Specifically, Jones et al. (2001) derive the run length distribution of the EWMA control chart with estimated parameters. Borror (1998) creates plots of the ARL for the Poisson EWMA chart that can be used to determine the control chart design parameters needed to realize the desired in-control ARLs. Gan (1990) proposes three modified EWMA charts for Poisson data. Xie and Goh (1993) employ a zero-inflated Poisson (ZIP) distribution to monitor the number of defects or nonconformities. This method is used to monitor data when there are a large number of zeros in the sample, and performs significantly better than the traditional *c* chart in this environment.

Jackson (1972) shows that the Poisson distribution is the appropriate model to use in a large number of instances, but it will not fit all instances. He goes on to show that the inappropriate use of the Poisson distribution may result in incorrect inferences regarding proportions of the population within certain limits. The particular instances examined are when defects occur in clusters and when defects are the result of two underlying causes. There are many more examples that can be found throughout the literature. This research proposes the use of the *c* chart to improve inventory accuracy for a large retail or warehouse environment.

This literature review gives an overview of the robustness of both variable and attribute control charts. It is obvious from this review that there has been extensive research on modifications to both types of charts in an attempt to either improve their performance or extend the possible environments for which these charts can be used. From this information it can be concluded that the focus of this research to examine the use of a *c* chart on a mixture distribution has not been investigated.

CHAPTER 3 - Complete Enumeration

3.1 Introduction

This section of the thesis provides the baseline for results by determining the pdf for several examples using complete enumeration of all possible events. Specifically, we determine the pdf for a population of five sub-populations (*m*=5), and sample sizes of two and three (*n*=2 and *n*=3). We then determine the expected value and variance for the populations. With these values the control limits are determined. Finally, the type I error rates for the example problems are obtained using complete enumeration. Solving these examples using complete enumeration creates a foundation of this work that supports extensions of the results by providing a confident basis for verification.

After obtaining the outcomes of examples for sample sizes of *n*=2 and *n*=3, the problem is looked at using a very limited, but balanced set of scenarios to provide an assessment of type I and type II errors. This set of scenarios serves as sensitivity analysis on the results by examining different problem structures and determining if desirable type I error rates are still acquired, which provides an assessment of the power of the approach. These scenarios are discussed in greater detail within the next few sections.

3.2 Method

The method used to solve the aforementioned examples of our problem is described in this section. This includes a numerical example that shows how the results are calculated. The notation used in our model to solve these problems is shown below:

$\propto_i$ = proportion of the population represented by pdf i

λ_i = Poisson arrival rate of the ith population

X_i = random variable representing the number of inventory adjustments from sample population

Y= random variable that consists of the sum of the values for X_i

Z= random variable that results from a mixture of m subpopulations

n = number of SKU's sampled to assess inventory adjustments for the population

m = number of subpopulations

Aside from the notation, it is also important to note that the number of inventory adjustments in a subpopulation is representative of the number of adjustments that have to be made to each specific SKU. For example, if inventory is checked and records show that there should be 55 units in stock and there are only 50 units, an inventory adjustment of 5 units would be required. It is pertinent to mention that the sum of random variables for our problem follows this expression:

$$Y = X_1 + X_2 + .. + X_n \tag{9}$$

Where once again n represents the number of SKUs being sampled, and in our particular example n=2. Also, Z is a variable that is observed from a population that results from a mixture of m subpopulations. The resulting pdf for Z can be represented as follows:

$$\mathrm{f}_z = \propto_1 \ \mathrm{f}_{z1} + \propto_2 \ \mathrm{f}_{z2} + \cdots + \propto_m \ \mathrm{f}_{zm}$$

Where Z_i is the variable of sub population i. This mixture population closely resembles mixture distributions found in popular probability texts (e.g. Feller (1970)).

3.2.1 Constructing an Example

The first step to solving our example problem is to determine the expected value and variance of a mixture distribution example. The expected value and variance are determined by using basic expected value operators and is shown in Miller (2008). For our application, we will be using a mixture of m Poisson distributions, and the resulting expected value, E(Z) and variance, Var(Z) are as follows:

$$E(Z) = \sum_{i=1}^{m} \propto_i \; E(Z_i) \tag{10}$$

$$Var(Z) = \sum_{i=1}^{m} \propto_i \; [(\lambda_i)^2 + \lambda_i] - (\sum_{i=1}^{m} \propto_i \lambda_i)^2 \tag{11}$$

Where $\propto_i$ is the proportion of the population represented by the pdf i, and λ_i is the Poisson arrival rate of the ith population.

The next step is to determine what the control limits of our problem will be. This is done using the following equations:

$$\text{UCL} = nE[Z] + \; 3\sqrt{nVar[Z]} \tag{12}$$

$$\text{LCL} = nE[Z] - \; 3\sqrt{nVar[Z]} \tag{13}$$

In these equations n is the number of SKUs sampled, and X_i is a random variable that represents the number of inventory adjustments in the sample population.

Now that these equations have been introduced, an example with m = 5 subpopulations and n = 2 SKUs sampled is presented. The problem has the following parameters:

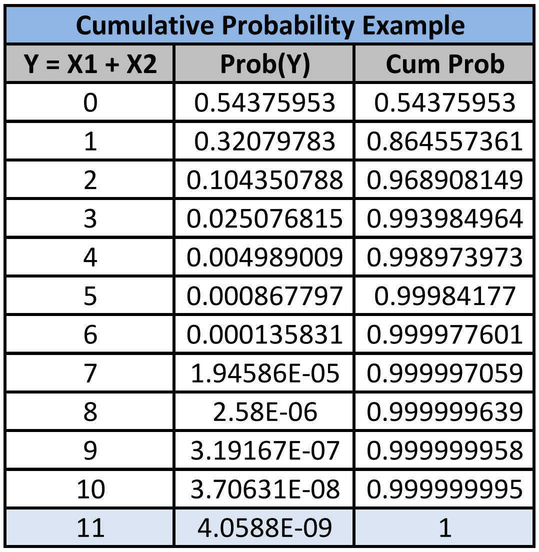
λ_i = 0.25, 0.50, 0.75, 0.20, 0.30

$\propto_i$ = 0.15, 0.20, 0.05, 0.40, 0.20

Once these parameters are presented, E(Z) is computed according to (10), and it is determined to be 0.315. Similarly, the Var(Z) is computed according to (11), and it is 0.337. Next, an UCL = 4.74 and LCL = -2.22 are computed using (12) and (13), respectively.

Now that the control limits have been established, complete enumeration is used to exhaust all possible numerical combinations of the number of inventory adjustments for the two SKUs being sampled. For example, the only combination for Y=0 is X_1=0 and X_2=0. Therefore, the probability of *Y*=0 is simply the Pr(X_1=0)*Pr(X_2=0). This process is carried out all the way to *Y*=11, which consists of 12 different combinations of X_1 and X_2. After enumerating *Y*=11, the cumulative probability sums to 1 using nine significant digits, and it can be concluded that practically all possible combinations of X_1 and X_2 have been exhausted. Table 3.1 presents the pdf for this scenario.

Table 3.1 Cumulative Probability for Example Problem



Cumulative Probability Example		
Y = X1 + X2	Prob(Y)	Cum Prob
0	0.54375953	0.54375953
1	0.32079783	0.864557361
2	0.104350788	0.968908149
3	0.025076815	0.993984964
4	0.004989009	0.998973973
5	0.000867797	0.99984177
6	0.000135831	0.999977601
7	1.94586E-05	0.999997059
8	2.58E-06	0.999999639
9	3.19167E-07	0.999999958
10	3.70631E-08	0.999999995
11	4.0588E-09	1

The final step is to determine the type I error rate for this example. The type I error rate calculation for this example is done using a variation of (4) that can be seen below:

$$p' = 1 - (P[X_1 + X_2 \leq UCL] - P[X_1 + X_2 < LCL]) \qquad (14)$$

In this equation, it can be seen that the negative value of the LCL in our example pushes the second part of the equation to equal 0. Our problem has an UCL = 4.74, and to find the type I error, all the possible combinations of X_1 + X_2 that are less than or equal to 4 must be summed and subtracted from 1. This is relatively easy considering that all of the possible combinations have already been enumerated. Using this information the type I error rate for our example turns out to be 0.001026. This example shows how to calculate the probability of a type I error for a fundamental example of our problem by complete enumeration. The next section provides a type I error study of a set of scenarios using complete enumeration.

3.3 Experimental Results

As mentioned above, this section provides the type I errors for a balanced set of experimental conditions across values of λ_i and $\propto_i$ for a given m and n. These different structures are examined in the next section.

3.3.1 Problem Structures

The various problem structures presented are for examples consisting of n=2 and n=3 SKUs being sampled. These problem structures are designed to give a broad representation of possible designs for real world mixture distributions. The modifications made to each of these models consist of the following alterations to $\propto$ and λ:

$\propto$: Skewed Left (SL), Skewed Right (SR), and Fixed (F)

λ: Equal, Minimum–Maximum, Maximum–Minimum, Weighted Left (WL), Weighted Right (WR)

Examples of each one of these modifications can be seen below:

$\propto$ - SL: 0.025, 0.025, 0.025, 0.025, 0.9

$\propto$ - SR: 0.9, 0.025, 0.025, 0.025, 0.025

$\propto$ - F: 0.2, 0.2, 0.2, 0.2, 0.2

λ - Equal: 0.5, 0.5, 0.5, 0.5, 0.5

λ - MinMax: 0.1, 0.2, 0.3, 0.4, 0.5

λ - MaxMin: 0.5, 0.4, 0.3, 0.2, 0.1

λ - WL: 0.9, 0.1, 0.1, 0.1, 0.1

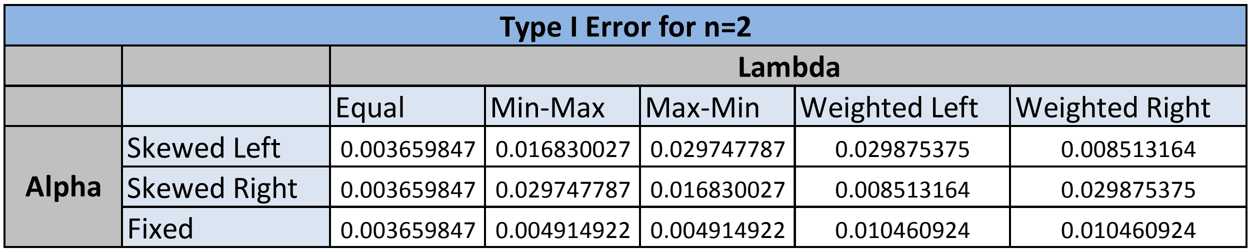
λ - WR: 0.1, 0.1, 0.1, 0.1, 0.9

Considering all of these alterations, there is a total of 15 different models to be examined for each value of *n*. The next section discusses the results that are found for these models. The type I error for each of these models are calculated using the same logic presented for the example problem in section 3.2.1.

3.4 Results

This section provides the results for all of the combinations of problem structures discussed in the previous section. First, the pdf's for the problem structures are defined and can be seen in Appendix A. Using these pdf's the type I error rates for *n*=2 can be summarized in table 3.2.

Table 3.2 Type I error for n=2



Type I Error for n=2						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.003659847	0.016830027	0.029747787	0.029875375	0.008513164
	Skewed Right	0.003659847	0.029747787	0.016830027	0.008513164	0.029875375
	Fixed	0.003659847	0.004914922	0.004914922	0.010460924	0.010460924

This table shows that the combinations have type I errors that are very small, but they exceed the traditional view of .0027 under the classic assumption of normality. They compare favorably if the probability of a type I error is designed to be less than 5%. There are some differences in the type I errors, but it is more important to note that all of the values are small enough to consider the model to be effective in terms of type I error for all of the different problem structures. There are also several patterns evident in this chart. First, is when the values of λ are equal, the type I error for all values of $\propto$ are the same. This logically makes sense because the E(X) and Var(X) will be the same for all three structures. Next, when the values of ∝ are fixed, the type I error for the Min-Max and Max-Min structures are the same because the calculations involved are the same, the numbers are just multiplied in reverse order. The same can be said for fixed ∝ values for the weighted left and weighted right structures. The final patterns in the chart are the diagonal values of the Min-Max and Max-Min, and Weighted Left and Weighted Right. These diagonal values are equal for both skewed left and skewed right ∝ values. Once again, this is because the numbers being multiplied are essentially the same as those being multiplied in reverse order. These patterns are useful to note, but the most important result of the graph is that all of these type I error values are significantly small, showing that the model is effective for a broad spectrum of problem structures. Similarly, the PDF's for *n*=3 are defined in Appendix B. The type I error rates for *n*=3 are be summarized in table 3.3.

Table 3.3 Type I error for n=3

Type I Error for n=3						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.004455981	0.003748813	0.008903452	0.01098578	0.005020188
	Skewed Right	0.004455981	0.008903452	0.003748813	0.005020188	0.01098578
	Fixed	0.004455981	0.016825077	0.016825077	0.024336803	0.024336803

This table shows that the type I errors for *n*=3 follow a very similar trend to the *n*=2 values. All of the patterns for the *n*=2 values are also evident in this chart. The most important conclusion that can be drawn is once again, that all of these type I errors are small enough to demonstrate that the model is effective for all of the different problem structures.

3.5 Conclusion

There are several conclusions that are drawn after completely enumerating the mixture distribution problem. The first is that solving example problems with complete enumeration establishes a foundation for further analysis. Specifically, the sample sizes of 4 or 5 become quite detailed; therefore, in the next chapter we promote and analyze an approach using conditional probabilities as shown in Miller (2008). The results of this chapter provide an excellent basis for verification for chapter 4. Clearly, extending this work to more practical sizes (e.g., *n*= 25, 50, and 100), and this approach is not tractable; therefore, as will be seen in chapter 5, we will use these results to validate larger scale simulations. Also, the results in this chapter provide a wide variety of mixture distribution structures. Inspecting these different structures shows that there are several patterns or similarities that can be identified between different structures. Finally, perhaps the most important aspect of this section is that the type I errors for all of the mixture distribution structures are small enough to prove that the method being used if effective for all of these structures.

CHAPTER 4 - Conditional Probabilities

4.1 Introduction

This chapter of the thesis examines using conditional probabilities to calculate both the type I and type II error rates of mixture distribution problems. The use of conditional probabilities greatly simplifies the calculations required to determine these errors, making its application to our problem advantageous. This approach follows the suggestion we have discussed using complete enumeration to calculate the type I errors. Up to this point it is evident that this method is not suitable for practical purposes, and merely serves as a foundation for the rest of this thesis. Applying conditional probabilities to solve type I and type II errors for mixture distributions serves as a practical method to computing these errors.

4.2 Method

This section describes the methods for which conditional probabilities are calculated. The derivation of equations for the type I error values for *n*=2, 3, 4, and 5 are shown as derived in Miller (2008). First, when 2 random samples of SKUs are collected, the total number of inventory adjustments for the sample is simply the sum of X_1 and X_2 (or *n*=2, using our notation). The resulting control limits are shown in equation (15)

$$\text{Control Limits} = 2E[\mathrm{Z}] \pm 3\sqrt{2Var[\mathrm{Z}]} \qquad (15)$$

Using the UCL for a sample of 2,

$$P[X_1 + X_2 \leq UCL]$$

$$= \sum_{X_1=0}^{UCL} P[X_1] \times P[X_2 \leq (UCL - X_1)] \tag{16}$$

Using the LCL for a sample of 2,

$$P[X_1 + X_2 < LCL]$$

$$= \sum_{X_1=0}^{LCL} P[X_1] \times P[X_2 < (LCL - X_1)] \tag{17}$$

The probability of a type I error when *n*=2 is shown in equation (18)

$$1 - p' = P[X_1 + X_2 \leq UCL] \text{ - } P[X_1 + X_2 < LCL]$$

$$p' = 1 - (P[X_1 + X_2 \leq UCL] \text{ - } P[X_1 + X_2 < LCL]) \tag{18}$$

For *n*=3, the control limits are shown in equation (19)

$$\text{Control Limits} = 3E[\mathrm{Z}] \pm 3\sqrt{3Var[\mathrm{Z}]} \tag{19}$$

Using the UCL for a sample of 3,

$$P[X_1 + X_2 + X_3 \leq UCL]$$

$$= \sum_{X_1=0}^{UCL} P[X_1] P[X_2 + X_3 \leq (UCL - X_1)]$$

$$= \sum_{X_1=0}^{UCL} P[X_1] \left[\sum_{X_2=0}^{UCL-X_1} P[X_2] P[X_3 \leq (UCL - X_1 - X_2)]\right] \tag{20}$$

Using the LCL for a sample of 3,

$$P[X_1 + X_2 + X_3 < LCL]$$

$$= \sum_{X_1=0}^{LCL} P[X_1] P[X_2 + X_3 < (LCL - X_1)]$$

$$= \sum_{X_1=0}^{LCL} P[X_1] \left[\sum_{X_2=0}^{LCL-X_1} P[X_2] P[X_3 < (LCL - X_1 - X_2)]\right] \tag{21}$$

Therefore, using the probability of being within the range of the control limits, the type I error rate, p' is defined in equation (22)

$$1 - p\text{'} = P[X_1 + X_2 + X_3 \leq UCL] \text{ - } P[X_1 + X_2 + X_3 < LCL]$$

$$p\text{'} = 1 - (P\left[X_1 + X_2 + X_3 \leq UCL\right] \text{ - } P[X_1 + X_2 + X_3 < LCL]) \tag{22}$$

For *n*=4, the control limits are shown in equation (23)

$$\text{Control Limits} = 4E[\mathrm{Z}] \pm 3\sqrt{4Var[\mathrm{Z}]} \tag{23}$$

Using the UCL for a sample of 4,

$$P[X_1 + X_2 + X_3 + \ X_4 \leq UCL]$$

$$= \sum_{X_1=0}^{UCL} P[X_1] \times P[X_2 + X_3 + \ X_4 \leq (UCL - X_1)]$$

$$= \sum_{X_1=0}^{UCL} P[X_1] \times \left[\sum_{X_2=0}^{UCL-X_1} P[X_2] \times P[X_3 + \ X_4 \leq (UCL - X_1 - X_2)]\right]$$

$$= \sum_{X_1=0}^{UCL} P[X_1] \times \left[\sum_{X_2=0}^{UCL-X_1} P[X_2] \times \left[\sum_{X_3=0}^{UCL-X_1-X_2} P[X_3] \times P[\ X_4 \leq (UCL - X_1 - X_2 - X_3)]\right]\right] \tag{24}$$

Using the LCL for a sample of 4,

$$P[X_1 + X_2 + X_3 + \ X_4 \leq LCL]$$

$$= \sum_{X_1=0}^{LCL} P[X_1] \times P[X_2 + X_3 + \ X_4 \leq (LCL - X_1)]$$

$$= \sum_{X_1=0}^{LCL} P[X_1] \times \left[\sum_{X_2=0}^{LCL-X_1} P[X_2] \times P[X_3 + \ X_4 \leq (LCL - X_1 - X_2)]\right]$$

$$= \sum_{X_1=0}^{LCL} P[X_1] \times \left[\sum_{X_2=0}^{LCL-X_1} P[X_2] \times \left[\sum_{X_3=0}^{LCL-X_1-X_2} P[X_3] \times P[\ X_4 \leq (LCL - X_1 - X_2 - X_3)]\right]\right] \tag{25}$$

Therefore, using the probability of being within the range of the control limits, the type I error rate, p', is defined in equation (26)

$$1 - p\text{'} = P[X_1 + X_2 + X_3 + \ X_4 \leq UCL] \text{ - } P[X_1 + X_2 + X_3 + \ X_4 < LCL]$$

$$p' = 1 - (P\left[X_1 + X_2 + X_3 + X_4 \leq UCL\right] \text{ - } P[X_1 + X_2 + X_3 + X_4 < LCL]) \tag{26}$$

For *n*=5, the control limits are shown in equation (23)

$$\text{Control Limits} = 5E[\text{Z}] \pm 3\sqrt{5Var[\text{Z}]} \tag{27}$$

Using the UCL for a sample of 5,

$$P[X_1 + X_2 + X_3 + X_4 + X_5 \leq UCL]$$

$$= \sum_{X_1=0}^{UCL} P[X_1] \times P[X_2 + X_3 + X_4 + X_5 \leq (UCL - X_1)]$$

$$= \sum_{X_1=0}^{UCL} P[X_1] \times \left[\sum_{X_2=0}^{UCL-X_1} P[X_2] \times P[X_3 + X_4 + X_5 \leq (UCL - X_1 - X_2)]\right]$$

$$= \sum_{X_1=0}^{UCL} P[X_1] \times \left[\sum_{X_2=0}^{UCL-X_1} P[X_2] \times \left[\sum_{X_3=0}^{UCL-X_1-X_2} P[X_3] \times P[\, X_4 + X_5 \leq (UCL - X_1 - X_2 - X_3)]\right]\right]$$

$$\sum_{X_1=0}^{UCL} P[X_1] \times \left[\sum_{X_2=0}^{UCL-X_1} P[X_2] \times \left[\sum_{X_3=0}^{UCL-X_1-X_2} P\left[X_3\right] \times \left[\sum_{X_4=0}^{UCL-X_1-X_2-X_3} P[\, X_4] \times P[X_{5]} \leq (UCL - X_1 - X_2 - X_3 - X_4)\right]\right]\right]\right] \tag{28}$$

Using the LCL for a sample of 5,

$$P[X_1 + X_2 + X_3 + X_4 + X_5 \leq LCL]$$

$$= \sum_{X_1=0}^{LCL} P[X_1] \times P[X_2 + X_3 + X_4 + X_5 \leq (LCL - X_1)]$$

$$= \sum_{X_1=0}^{LCL} P[X_1] \times \left[\sum_{X_2=0}^{LCL-X_1} P[X_2] \times P[X_3 + X_4 + X_5 \leq (LCL - X_1 - X_2)]\right]$$

$$= \sum_{X_1=0}^{LCL} P[X_1] \times \left[\sum_{X_2=0}^{LCL-X_1} P[X_2] \times \left[\sum_{X_3=0}^{LCL-X_1-X_2} P[X_3] \times P[\, X_4 + X_5 \leq (LCL - X_1 - X_2 - X_3)]\right]\right]$$

$$= \sum_{X_1=0}^{LCL} P[X_1] \times \left[\sum_{X_2=0}^{LCL-X_1} P[X_2] \times \left[\sum_{X_3=0}^{LCL-X_1-X_2} P[X_3] \times \left[\sum_{X_4=0}^{LCL-X_1-X_2-X_3} P[\, X_4] \times P[X_{5]} \leq (LCL - X_1 - X_2 - X_3 - X_4)\right]\right]\right] \tag{29}$$

Therefore, using the probability of being within the range of the control limits, the type I error rate, p', is defined in equation (30)

$$1 - p' = P[X_1 + X_2 + X_3 + X_4 + X_5 \leq UCL] - P[X_1 + X_2 + X_3 + X_4 + X_5 < LCL]$$

$$p' = 1 - (P[X_1 + X_2 + X_3 + X_4 + X_5 \leq UCL] - P[X_1 + X_2 + X_3 + X_4 + X_5 < LCL]) \quad (30)$$

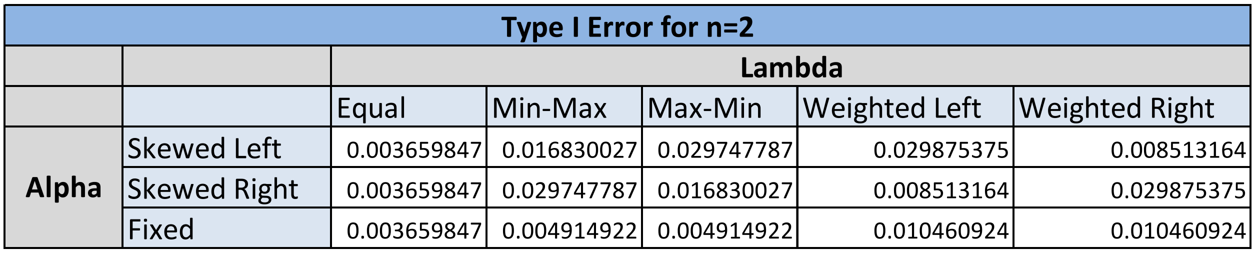
We have shown a probabilistic approach to determine the type I errors for sample sizes of 2, 3, 4, and 5. The next step is to develop a method that calculates these errors. To achieve this, a model is created in C++. This model has the ability to calculate the type one error for different values of $\propto$, λ, m, and n using conditional probabilities. To make changes to the values of $\propto$ and λ in the model, the user modifies input values. To alter the number of subpopulations, *m*, additional constants for $\propto$ and λ with input values for each must be created. Finally, to change the value of the number of SKUs sampled, *n*, the constant called *dim* in the code must be altered to represent the desired number of SKUs to be sampled. The code for this C++ program can be found in Appendix A. Using this code the results for the type I error for problems containing *n*=2, 3, 4, and 5 are obtained.

4.3 Results

4.3.1 Type I Errors

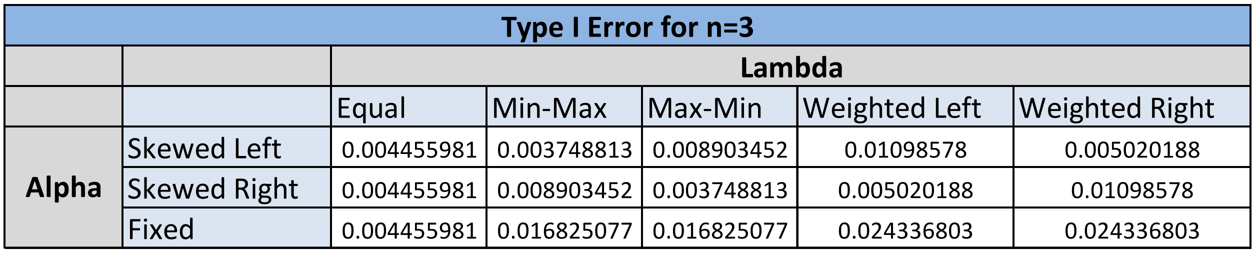
The first step in analyzing the results from this program is to compare the results for *n*=2 and *n*=3 to the results that were obtained in the complete enumeration section of the thesis. As mentioned earlier, the complete enumeration results serve as a foundation to which we compare the results obtained from our program to confirm their accuracy. To do this, the program is run for all of the models created in the complete enumeration section. Specifically, we compare the table of the type I errors obtained from the program to the table of type I errors obtained from the complete enumeration. The type I errors for all of the structures examined are nearly identical for both *n*=2 and *n*=3. The errors for *n*=2 and *n*=3 calculated using conditional probabilities are shown in tables 4.1 and 4.2 respectively.

Table 4.1 Conditional Probability Type I Errors for n=2



Type I Error for n=2						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.003659847	0.016830027	0.029747787	0.029875375	0.008513164
	Skewed Right	0.003659847	0.029747787	0.016830027	0.008513164	0.029875375
	Fixed	0.003659847	0.004914922	0.004914922	0.010460924	0.010460924

Table 4.2 Conditional Probability Type I Errors for n=3

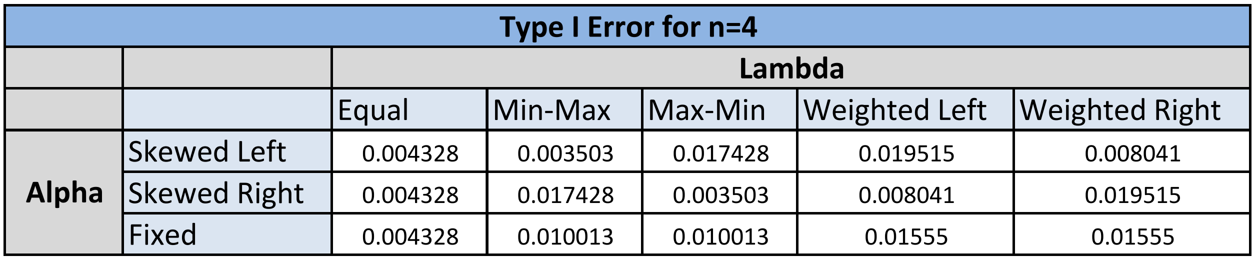


Type I Error for n=3						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.004455981	0.003748813	0.008903452	0.01098578	0.005020188
	Skewed Right	0.004455981	0.008903452	0.003748813	0.005020188	0.01098578
	Fixed	0.004455981	0.016825077	0.016825077	0.024336803	0.024336803

These tables show that the only differences between these errors and the completely enumerated results are slight rounding errors of decimals that are less than a thousandth off. These similarities allow us to reach the conclusion that our model accurately calculates the type I errors of mixture distributions.

Now that the validity of the model created has been confirmed, the next step is to increase the value of *n* to try and reach a more practical level. For this thesis, the value of *n* is increased to 4 and 5. This demonstrates that the model works for increasing the value of *n*, and that the desired value for each case in the real world could be obtained. The type I error rates for *n*=4 are summarized in table 4.3.

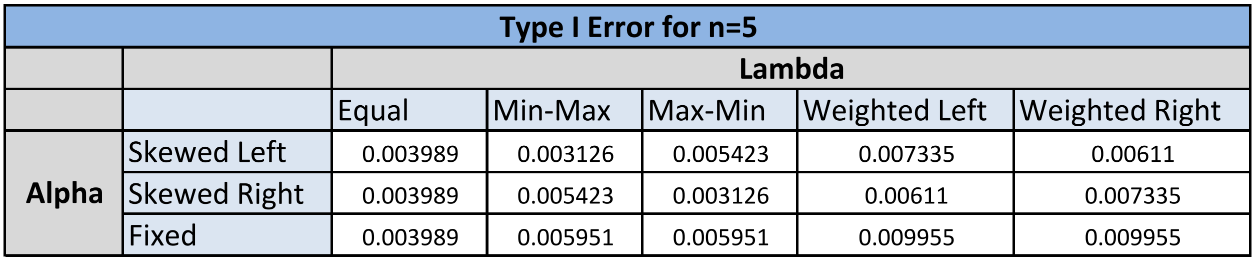
Table 4.3 Conditional Probability Type I Errors for n=4



Type I Error for n=4						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.004328	0.003503	0.017428	0.019515	0.008041
	Skewed Right	0.004328	0.017428	0.003503	0.008041	0.019515
	Fixed	0.004328	0.010013	0.010013	0.01555	0.01555

This chart shows the type I errors for all the problem structures examined in this paper. It is also evident that the patterns that were illustrated in the charts for *n*=2 and *n*=3 are still present. The type I error rates for *n*=5 are summarized in table 4.4.

Table 4.4 Conditional Probability Type I Errors for n=5



Type I Error for n=5						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.003989	0.003126	0.005423	0.007335	0.00611
	Skewed Right	0.003989	0.005423	0.003126	0.00611	0.007335
	Fixed	0.003989	0.005951	0.005951	0.009955	0.009955

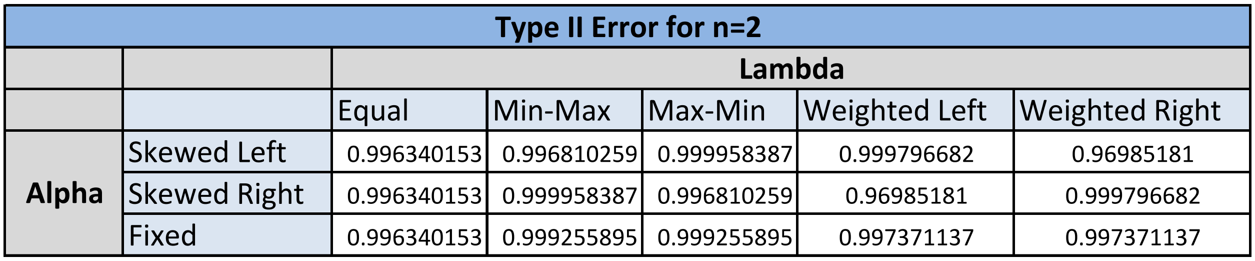
Once again, this shows the type I errors for the entire range of problem structures, and the patterns within the chart are still noticeable. The results for *n*=4 and *n*=5 show that our method provides small values for the type I error rates. Furthermore, these values illustrate a tendency to decrease as the value of n increases. This is significant, because in practice, the method of using a mixture distribution to monitor inventory accuracy would likely involve using higher values of *n*. So, this section shows that by using conditional probabilities it can be concluded that our method will not only obtain significantly low type I error rates for smaller values of *n*, but will also achieve improved results as the value of *n* increases.

4.3.2 Type II Errors

Up to this point in the paper, the type II error rates have not been examined yet. For this section we will look at the type II error rates of small shifts for *n* = 2, 3, 4, and 5. This section also examines the type II error rates of large shifts for *n*=2, 3, 4, and 5. This is done by assuming that the underlying distribution is going to be the model of fixed $\propto$ values and equal λ values (assumed "in-control" condition), and the type II errors are determined by overlaying each scenario on the assumed "in-control" condition. Making this assumption provides a basis to compute the type II error rates. This does not provide a complete examination of all of the type II error rates, but it presents an idea of how these models perform in terms of type II errors.

Since the distribution consists of fixed $\propto$ values and equal λ values, the type II error rates can be computed by the probability that the distribution actually follows the specific problem structure considered. First, we examine the type II errors for small shifts that consist of all the problem structures and *n* =2. These type II errors for *n*=2 are summarized in Table 4.5.

Table 4.5 Type II Errors for n=2



Type II Error for n=2						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.996340153	0.996810259	0.999958387	0.999796682	0.96985181
	Skewed Right	0.996340153	0.999958387	0.996810259	0.96985181	0.999796682
	Fixed	0.996340153	0.999255895	0.999255895	0.997371137	0.997371137

As seen in this table, the type II errors are all relatively large. This is actually not a complete surprise, as the type I errors for this model are very small, and small type I errors can often correspond with larger type II errors when the shift is not very significant. This shows that additional numerical work could be done to create more realistic type I errors so that the type II errors do not become too excessive. In essence this relaxes or widens the control limits in an attempt to balance the type I and type II errors. Another way to look at these type II errors is that they show how important it is to accurately estimate the values of $\propto$ and λ. These type II errors demonstrate that if there is small shift in the λ values, it is going to very difficult to detect that the wrong parameters are being used.

Now that the type II errors for *n*=2 have been examined, the results for *n*=3, 4, and 5 will be presented for smaller shifts. The type II errors for *n*=3, 4, and 5 respectively are summarized in the following tables:

Table 4.6 Type II Errors for n=3

Type II Error for n=3						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.995544019	0.996251187	0.999982229	0.999905216	0.955404222
	Skewed Right	0.995544019	0.999982229	0.996251187	0.955404222	0.999905216
	Fixed	0.995544019	0.999365281	0.999365281	0.997905047	0.997905047

Table 4.7 Type II Errors for n=4

Type II Error for n=4						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.995672	0.996497	0.999999	0.999999	0.944491
	Skewed Right	0.995672	0.999999	0.996497	0.944491	0.999999
	Fixed	0.995672	0.999657	0.999657	0.999999	0.999999

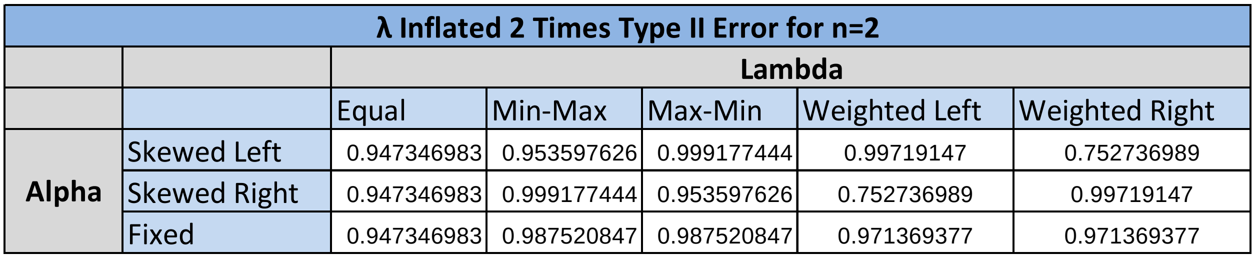
Table 4.8 Type II Errors for n=5

Type II Error for n=5						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.996011	0.996874	0.999999	0.999999	0.936976
	Skewed Right	0.996011	0.999999	0.996874	0.936976	0.999999
	Fixed	0.996011	0.999831	0.999831	0.999999	0.999999

These tables show that the performance is similar to the results for *n*=2. This gives us an idea of how our model performs in terms of type II errors. Our method does not thoroughly evaluate all possible type II errors, but the next step is to take a look at the type II errors when a large shift is present.

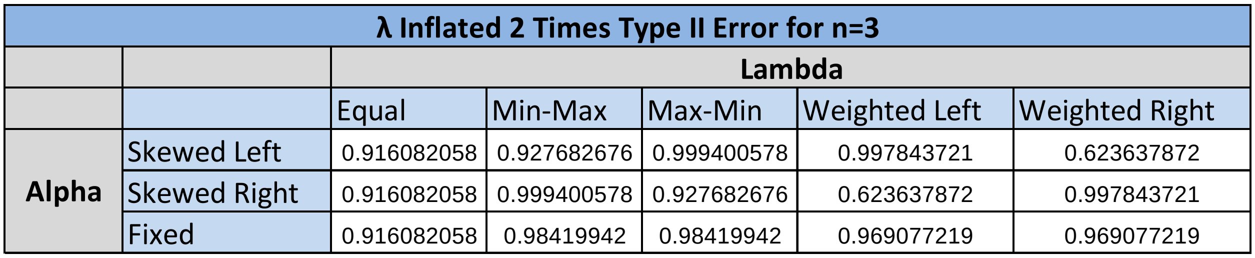
To represent a large shift, the values for λ in each of our problem structures will be multiplied by 2 and 3. This means that if the problem structure is being multiplied by 2, and has original λ values of .1, .2, .3, .4, and .5, then the values would change to .2, .4, .6, .8, 1. These scenarios are important to look at because large retails stores would want to be able to detect if they were making 2 and 3 times the number of inventory adjustments they assumed. For these large shifts, we will once again assume that the underlying distribution is going to be the model of fixed ∝ values and equal λ values, but this time the actual distributions are assumed to be the inflated values of 2 and 3 times the original λ values for each structure. The following tables present the type II errors for *n*=2, 3, 4, and 5 and inflated values of 2 times the original λ values.

Table 4.9 λ Inflated 2 Times Type II Error for n=2



λ Inflated 2 Times Type II Error for n=2						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.947346983	0.953597626	0.999177444	0.99719147	0.752736989
	Skewed Right	0.947346983	0.999177444	0.953597626	0.752736989	0.99719147
	Fixed	0.947346983	0.987520847	0.987520847	0.971369377	0.971369377

Table 4.10 λ Inflated 2 Times Type II Error for n=3



λ Inflated 2 Times Type II Error for n=3						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.916082058	0.927682676	0.999400578	0.997843721	0.623637872
	Skewed Right	0.916082058	0.999400578	0.927682676	0.623637872	0.997843721
	Fixed	0.916082058	0.98419942	0.98419942	0.969077219	0.969077219

Table 4.11 λ Inflated 2 Times Type II Error for n=4

λ Inflated 2 Times Type II Error for n=4						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.889556	0.906492	0.999708	0.998591	0.518460
	Skewed Right	0.889556	0.999708	0.906492	0.518460	0.998591
	Fixed	0.889556	0.982983	0.982983	0.968077	0.968077

Table 4.12 λ Inflated 2 Times Type II Error for n=5

λ Inflated 2 Times Type II Error for n=5						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.867077	0.888769	0.999879	0.999107	0.432785
	Skewed Right	0.867077	0.999879	0.888769	0.432785	0.999107
	Fixed	0.867077	0.982721	0.982721	0.967134	0.967134

The results from these charts show that the type II errors for larger shifts are much more reasonable than the type II errors for smaller shifts. It is also important to note that the type II errors decrease siginificantly when the value of *n* increases. This is significant because once our problem reaches more practical values of *n* (25, 50, and 100), the type II errors should only continue to decrease. The next set of tables shows the type II errors when the values of λ are multiplied by 3, to represent an even larger shift. The results for *n*= 2, 3, 4, and 5 and inflated to 3 times the λ values can be seen below.

Table 4.13 λ Inflated 3 Times Type II Error for n=2

λ Inflated 3 Times Type II Error for n=2						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.815263245	0.835176042	0.996033985	0.990227394	0.459125164
	Skewed Right	0.815263245	0.996033985	0.835176042	0.459125164	0.990227394
	Fixed	0.815263245	0.948812464	0.948812464	0.915564198	0.915564198

Table 4.14 λ Inflated 3 Times Type II Error for n=3

λ Inflated 3 Times Type II Error for n=3						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.702930435	0.737237134	0.996249783	0.990395103	0.277976337
	Skewed Right	0.702930435	0.996249783	0.737237134	0.277976337	0.990395103
	Fixed	0.702930435	0.926219341	0.926219341	0.899825346	0.899825346

Table 4.15 λ Inflated 3 Times Type II Error for n=4

λ Inflated 3 Times Type II Error for n=4						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.606680	0.652683	0.996957	0.991967	0.168131
	Skewed Right	0.606680	0.996957	0.652683	0.168131	0.991967
	Fixed	0.606680	0.908847	0.908847	0.889443	0.889443

Table 4.16 λ Inflated 3 Times Type II Error for n=5

λ Inflated 3 Times Type II Error for n=5						
		Lambda				
		Equal	Min-Max	Max-Min	Weighted Left	Weighted Right
Alpha	Skewed Left	0.525048	0.582937	0.997601	0.993082	0.101876
	Skewed Right	0.525048	0.997601	0.582937	0.101876	0.993082
	Fixed	0.525048	0.894962	0.894962	0.881996	0.881996

Once again, these tables confirm that our model is even more effective at detecting type II errors when the shift is even larger, in this case when the actual λ values are 3 times more than they were originally assumed, opposed to 2 times in the previous graphs. These graphs also still show the trend of the type II errors decreasing as the sample size *n* increases.

4.4 Conclusion

There are several conclusions that can be reached after looking at the type II analysis done in this section. In terms of detecting small shifts, additional work could be conducted to create more realistic type I errors so that the type II errors do not stay too extreme. Furthermore, smaller shifts could be considered. The large type II error values also show how precise that estimates for $\propto$ and λ must be. If these estimates are off, it should be very difficult to determine if the model estimates chosen are correct or not. Additionally, this section shows that our model is much more effective at detecting large shifts in the actual number of inventory adjustments. This is significant because retail stores will initially be most concerned with knowing if their number of inventory adjustments is actually at a higher level than they assumed.

CHAPTER 5 - Simulation

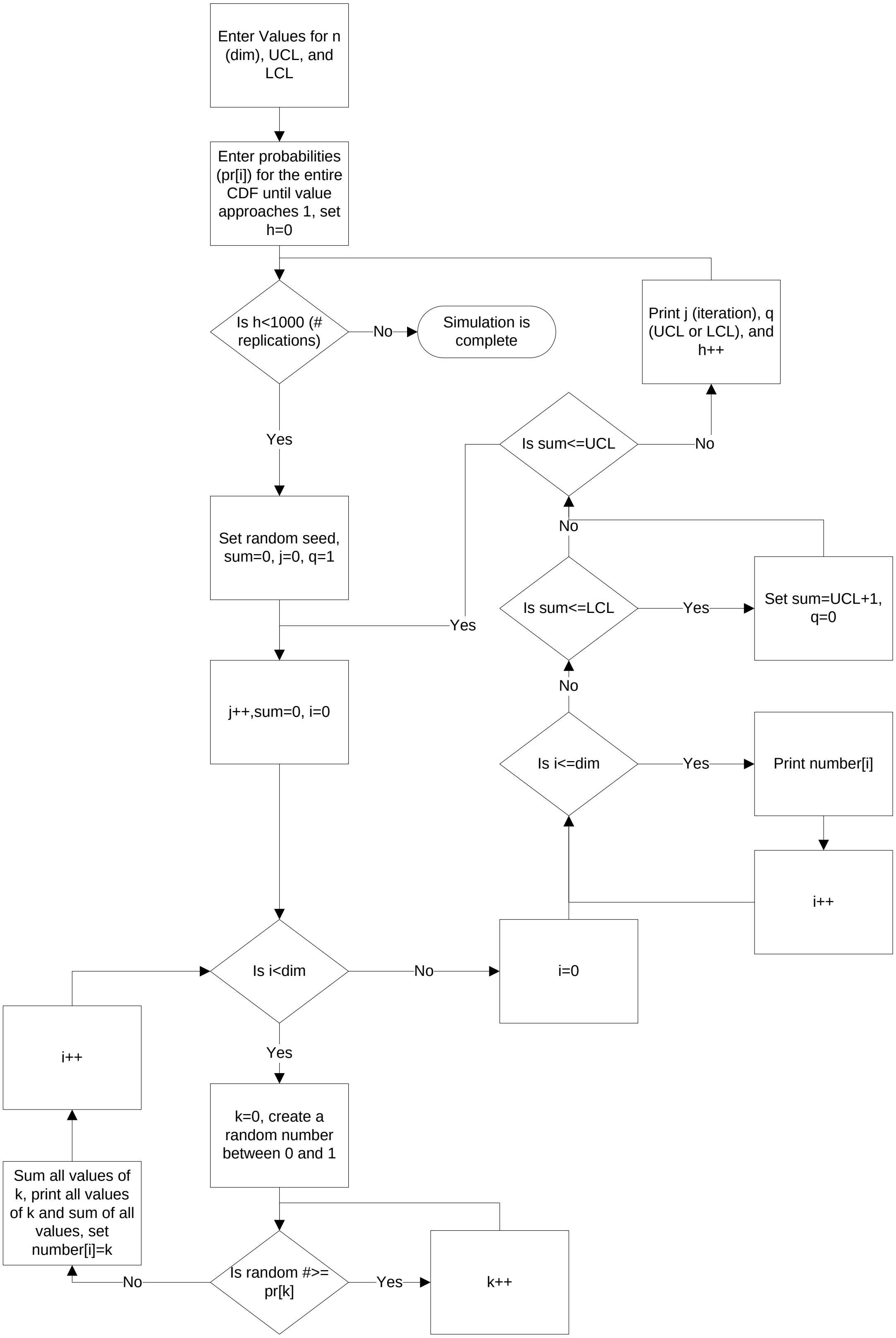
5.1 Introduction

As mentioned earlier, this chapter of the thesis extends our work to more practical sizes of *n*= 25, 50, and 100. Through computer simulation using C++ programming, we construct a simple program that determines the approximate type I error rate for a set of problems by estimating the ARLs. Up to this chapter, we have examined using complete enumeration and conditional probabilities to calculate type I and type II errors. Simulation allows us to estimate type I error rates for more practical size problems. Clearly, an infinite number of scenarios could be evaluated, but in this chapter we simply want to present the methodology.

5.2 Method

This section describes the methods used in our simulation to determine the ARL of our problem. Figure 1 presents a flow chart to help understand the methods used in the simulation.

Figure 1: Flow chart for Simulation


Enter Values for n (dim), UCL, and LCL
Enter probabilities (pr[i]) for the entire CDF until value approaches 1, set h=0
Is h<1000 (# replications)
No
Simulation is complete
Yes
Set random seed, sum=0, j=0, q=1
j++,sum=0, i=0
Is i<dim
No
i=0
Yes
k=0, create a random number between 0 and 1
Is random #>= pr[k]
Yes
k++
No
Sum all values of k, print all values of k and sum of all values, set number[i]=k
i++
Is i<=dim
Yes
Print number[i]
i++
No
Is sum<=LCL
Yes
Set sum=UCL+1, q=0
No
Is sum<=UCL
Yes
No
Print j (iteration), q (UCL or LCL), and h++

From this flow chart it can be seen that the first step enters given values for n (*dim*), UCL, LCL, and the probabilities (*pr[i]*) for the entire CDF until the value sums to 1. Since the simulation is run for three cases, the value of n (25, 50, and 100) has to be changed for each run, and the UCL and LCL must be changed as well. Next, a decision on whether the variable *h* is less than 1,000 is made to represent the completion of the 10 sets of 100 that are run for the simulation. Obviously, if *h* is greater than 1,000 then the simulation is complete, and if not a random seed is assigned, *sum* is set to 0, j is set to 0, and q is set to 1. The variable *sum* will be used to sum all of the inventory adjustments so that it can be determined if they are within the control limits. The variable *j* is used to count the iterations, and shows how many iterations it takes for a number to be above the UCL or below the LCL (ARL), while the variable *q* is used to show whether the point that plots out of control is above the UCL or below the LCL (*q*=1 is above UCL and *q*=0 is below the LCL). Next, the variable *j* is incremented by 1 and the variables *i* and *sum* are set to 0.

The next decision is to determine if *i* is less than the value entered for *dim*. If it is, a variable *k* is set to 0 and then a random number between 0 and 1 is created. This ensures that if *dim*=25, then there will be 25 random numbers created. It is then determined if the random number is greater than or equal to *pr[k],* which are entered in the first step. If the random number is greater than the *pr[k],* then *k* is incremented by 1 to determine what number of inventory adjustments corresponds to the random number. If the random number is not greater than *pr[k],* then the value of *k* is recorded and this is the number of inventory adjustments associated with that random number. The value of *k* is recorded and sum is set to equal all values of *k* that are eventually determined by this decision block. Also the value *number[i]* is set to equal *k* so that these numbers can be printed out later. After these values are printed, the value of *i* is incremented, and this loop continues until *i* is greater than *dim*. Next, *i* is reset to a value of 0 because it is used again as a counter.

The next decision is whether *i* is less than or equal to dimension. If it is, then *number[i]* is printed, and *i* is incremented until *i* is less than *dim*. These numbers are printed solely for the purpose of examining all of the output of the program to ensure that it is working properly.

Now that all values of *k* are stored in the variable *sum*, the next decision is to determine if the value for *sum* is less than or equal to LCL. If the value is not less than or equal to the LCL the decision block of whether the value for *sum* is less than or equal to the UCL is approached. If *sum* is less than or equal to the LCL, then *sum* is set to UCL+1 and sent to the next decision block. Also, the value of *q* is set to zero to indicate that the control limits were violated on the lower bound. When *sum* is not less than or equal to the UCL, the values for *j* (iteration) and *q* (UCL or LCL) are printed and the value of *h* is incremented. The final option is if *sum* is not less than LCL and is less than UCL, the value of *j* is incremented and the entire process is repeated until a value for *sum* is outside of the control limits.

As mentioned before this process is repeated until 1,000 replications of the variable *h* are completed so that a good estimate of the ARL can be made. This ARL estimate can then be used to determine and estimate of the type I error as described in equation (3). An example of the C++ code used in this simulation can be seen in Appendix B.

5.3 Results

The first step in analyzing the results for this simulation is to compare the results obtained from the complete enumeration section to the results of the simulation. This was done by examining the problem with fixed $\propto$ values of .2 and equal λ values of .5. These parameters were entered into the simulation to compare the results. After 10 sets of 100 replications, the simulation gave an ARL of 273.98. A 95% confidence interval for this estimate has an upper interval of 289.13 and a lower interval of 258.83. The type I error from the complete enumeration of this example corresponds to an ARL of 273.24, which clearly falls between the 95% confidence interval of the ARL for our simulation. The similarities between these results allow us to conclude that our simulation is able to accurately estimate the ARL for the mixture distributions being examined.

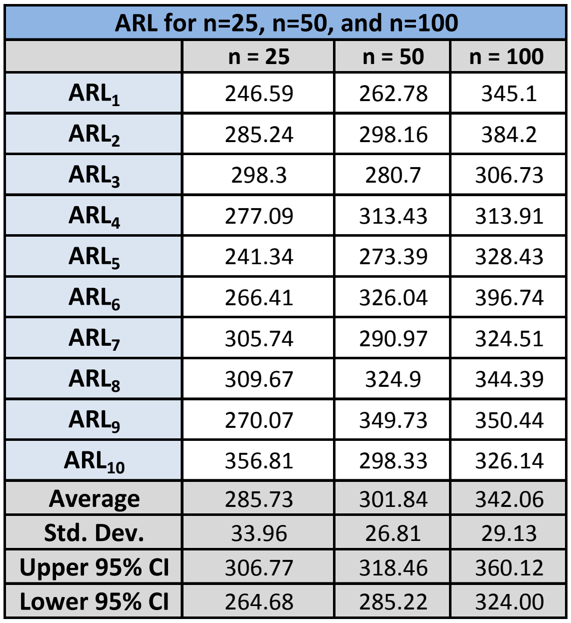
Now that the accuracy of our simulation is confirmed, this section shows the results for more practical sized problems of *n* = 25, 50, and 100. The type I errors are calculated using the following random problem structure for each sample size:

λ_i = 2, 3, 4, 5, 6

$\propto_i$ = 0.30, 0.30, 0.20, 0.10, 0.10

The results from the simulation can be seen in the following table.

Table 5.1 ARL for n=25, n=50, and n=100



ARL for n=25, n=50, and n=100			
	n = 25	n = 50	n = 100
ARL_1	246.59	262.78	345.1
ARL_2	285.24	298.16	384.2
ARL_3	298.3	280.7	306.73
ARL_4	277.09	313.43	313.91
ARL_5	241.34	273.39	328.43
ARL_6	266.41	326.04	396.74
ARL_7	305.74	290.97	324.51
ARL_8	309.67	324.9	344.39
ARL_9	270.07	349.73	350.44
ARL_{10}	356.81	298.33	326.14
Average	285.73	301.84	342.06
Std. Dev.	33.96	26.81	29.13
Upper 95% CI	306.77	318.46	360.12
Lower 95% CI	264.68	285.22	324.00

First, the simulation is run for *n* = 25 and as can be seen from the table, an ARL of 313.67 is obtained, which corresponds to a type I error of 0.003188. For *n* =50 the ARL is 267.53 or a type I error of 0.003742, while *n* = 100 corresponds to an ARL of 342.06, or type I error of 0.002923. The standard deviation and 95% confidence intervals for each scenario is shown as well. The type I error estimates of this simulation are even smaller than the type I errors of the initial problems examined in the previous section. It is evident that these type I errors are very desirable, and confirm that our method is also applicable to more practical size problems.

5.4 Conclusion

This section has shown that our method of computing the type I error for a mixture distribution is applicable to problems that are much more realistic than our initial examples. It is unlikely that large retail stores would sample *n*=2, 3, 4, and 5 SKUs when they have a store full of thousands of SKUs. This section shows that when *n*=25, 50, and 100 our method is still able to obtain very low type I errors. It is much more realistic to assume that a retail store would sample 25 SKUs, which is why this section is so important to the research. The results from this section solidify this work as an application that should be applied to an environment such as a large retail or warehousing store.

CHAPTER 6 - Future Work and Applications

6.1 Future Work

This thesis has progressed through reviewing literature on cycle counting and SPC, completely enumerating examples of our problem, using conditional probabilities to extend the computing capabilities, and finally creating a simulation to apply our method to more practical problems. We have shown that this method is effective in monitoring inventory accuracy and detecting changes. However, after completing this work, there are several areas of additional research identified that could further advance the research to become more applicable and effective for large retailers.

The first area involves the balancing of type I and type II errors for our model. Throughout this paper, the type I errors have proved to be very small as they were based on 3σ limits for the control charts, however in Chapter 4 a concise examination of type II errors for our model shows that their values are extremely large for small shifts. To make this method effective in practice, fixed type I errors (e.g. 5%) could be required so the type II errors are based upon fixed type I errors. As a result, balance in the performance of both type I and type II errors could be realized, and the ability to detect smaller shifts would be increased.

Aside from the actual performance of our model, it would also be valuable to extend the simulation to collect data in a more balanced and exhaustive manner. One possibility to achieve this is to find an industrial business to partner with on the research and validate experimental boundaries of the factors. Locating a business that would be able to provide data and real world examples would be very advantageous to tailoring this research to practical examples. Having this partner would not only allow for the simulation to be extended to match reality, but would open several opportunities including what values of $\propto$ and λ should be used and how difficult it is for a business to obtain these values.

Another area that could be examined is how estimation errors for the parameters of $\propto$ and λ should be handled. Undoubtedly, there will be errors and how these areas are handled is very important to the success of this research. Currently, this model assumes a stationary process. Altering this assumption to assume that the process is non stationary could be a method to control these estimation errors.

The final research area that could be developed is using this method in conjunction with a minimum cost model. A model could be developed that examines the costs a retail environment realizes due to inaccurate inventory, the advantages of obtaining completely accurate inventory, and the cost of sampling. If this cost model is able to show a cost savings by implementing this research, it would create interest from industrial businesses and greatly advance the possibility of having this method implemented by a company.

6.2 Applications

This section will briefly touch on the applications of how this method is implemented, and specifically how the parameters for the models should be developed. This research does not delve into determining optimal parameters, but will touch briefly on how the values of $\propto$ and $\boldsymbol{\lambda}$ should be chosen. It is our recommendation that the values of λ be determined by historical data. For example, a company could look at the past 50 inventory adjustments for a certain SKU, and the average of these adjustments should be used as the initial value for λ**.** The values for $\propto$ have also not been examined for optimal parameters, but these values should be experimented with and will vary for each application. So, to determine values for $\propto$ and λ, it is recommended that historical data is used, and to allow these values to be altered for each specific situation.

References

Borror, C.M., Rigdon, S.E., and Champ, C.W. (1992). "An Exponential Weighted Moving Average Control Chart for Poisson Data" Proceedings for the 23rd Annual Modeling and Simulation Conference, Pittsburg, PA. pp. 1775-1782.

Borror, C.M., Champ, C.W., and Rigdon, S.E. (1998). "Poisson EWMA Control Charts," *Journal of Quality Technology,* Vol. 30, No. 4, pp. 352-361.

Boyd, D.F. (1950). "Applying the Group Control Chart for $\bar{X}$ and *R,*" *Industrial Quality Control,* Vol. 6.

Brooks, R.B. and Wilson, L.W. (2005). *Inventory Record Accuracy – Unleashing the Power of Cycle Counting,* John-Wiley & Sons.

Burroughs, T.E., Rigdon, S.E., and Champ, C.W. (1993). "An Analysis of Shewhart Charts with Run Rules When No Standards Are Given," *Proceedings of the Quality and Productivity Section of the American Statistical Association,* August 8-12, San Francisco, CA, pp. 16-19.

Champ, C.W., and Woodall, W.H. (1987). "Exact Results for Shewhart Control Charts with Supplementary Runs Rules," *Technometrics,* Vol. 29, No. 4, pp. 393-399.

Chang, T. (2008). "Simultaneous Monitoring of Bias, Linearity, and Precision of Multiple Measurement Gauges," *Journal of Quality Technology,* Vol. 40, No. 3, pp. 268-281.

Chen, G. (1998). "An Improved *P* Chart Through Simple Adjustments," *Journal of Quality Technology,* Vol. 30, No.2, pp. 142-151.

Cheng, G. (1998). "The Run Length Distribution of *R*, *S*, and S^2 Control Charts when σ Is Estimated," *Canadian Journal of Statistics,* 26, pp. 311-322.

Coates, E., Okell, E., and Tricker, A. (1998). "The Effect on the *R* Chart of Precision of Measurement," *Journal of Quality Technology,* Vol. 30, No. 3, pp. 232-239.

Costa, A.B. (1999a). "Joint *X-Bar* and *R* Charts with Variable Sample Sizes and Sampling Intervals," *Journal of Quality Technology,* Vol. 31, No. 4, pp. 387-397.

Costa, A.B. (1999b). "*X-Bar* Charts with Variable Parameters," *Journal of Quality Technology,* Vol. 31, No. 4, pp. 408-416.

Crowder, S.V. (1987). "A Simple Method for Studying Run-Length Distributions of Exponentially Weighted Moving Average Charts," *Technometrics,* Vol. 29, No. 4, pp. 401-407.

Davis, R. B., and Woodall, W. H. (2002). "Evaluating and Improving the Synthetic Control Chart," *Journal of Quality Technology,* Vol. 34, No. 2, pp. 200-208.

Feller, W. (1970). *An Introduction to Probability Theory and Its Applications,* 3rd, rev. edn, Wiley, New York. Original edition:1950.

Gan, F. (1990). "Monitoring Observations Generated from a Binomial Distribution Using Modified Exponentially Weighted Moving Average Control Chart," *Journal of Statistical Computation and Simulation.* pp. 45-60.

Gan, F. (1990). "Monitoring Poisson Observations Using Modified Exponentially Weighted Moving Average Control Charts," *Communication in Statistics,* Vol. 19(1), pp. 103-124.

Gan, F. (1993). "An Optimal Design of CUSUM Control Charts for Binomial Counts," *Journal of Applied Statistics,* Vol. 20(4), pp. 172-183.

Goe, T.N., and Xie, M. (1993). "SPC of Near Zero-Defect Process Subject to Random Shock," *Quality and Reliability International,* Vol. 9, pp. 89-93.

Hillier, F.S. (1969). “$\overline{\text{X}}$ and R Chart Control Limits Based on a Small Number of Subgroups,” *Journal of Quality Technology,* Vol. 1.

Jackson, J.E-. (1972). “All Count Distributions are not Alike,” *Journal of Quality Technology,* Vol. 4, pp. 86-92.

Jones, A., Champ, C.W., and Rigdon, S.E. (2001). “The Performance of Exponentially Weighted Moving Average Charts With Estimated Parameters,” *Technometrics,* Vol. 33, pp. 156.

Khoo, M.C. (2004). “A Moving Average Control Chart for Monitoring the Fraction Non-Conforming,” *Quality and Reliability International,* Vol. 20, pp. 617-635.

Larson, K.E. (1969). “Plotting *p* and *np* Charts,” *Journal of Quality Technology,* Vol. 1, pp. 217-220.

Lucas, J.M., and Saccucci, M.S. (1990). “Exponentially Weighted Moving Average Control Schemes: Properties and Enhancements,” *Technometrics,* Vol. 32, No. 1, pp. 1-12.

Luceno, P., and Puig-Pey, J. (2000). “Evaluation of the Run-Length Probability Distribution for CUSUM Charts,” *Technometrics,* Vol. 32, pp. 411.

Luceno, P. and Puig-Pey, J. (2002). “Computing Run Length Probability Distribution for CUSUM Charts,” *Technometrics,* Vol. 34, No. 2, pp. 209-215.

McCue, B.C. (1976). “The Probability of a Type I Error and the Power to Detect a Single Slippage for the Upper Limit Control Chart for Number of Defects,” *Journal of Quality Control,* Vol. 8, pp. 133-139.

Miller, E.J. (2003). “Decipher Control Charts: A Practical Example of Common Cause and Special Cause Variation Aids in the Understanding of Control Charts,” *Quality,* Vol. 42.8, p22(1).

Miller, J.M. (2008). “A SPC Approach to Monitoring Inventory Adjustments for Large Retail and Warehouse Environments,” undergraduate honors thesis, University of Arkansas.

Montgomery, D.C. (1991). *Introduction to Statistical Quality Control,* 2nd ed., John Wiley & Sons, New York.

Nelson, L.S. (1986). “Control Chart for Multiple Stream Processes,” *Journal of Quality Technology,* Vol. 18.

Nelson, L.S. (1997). “An Early-Warning Test for Use with the Shewhart *p* Control Chart,” *Journal of Quality Control,* Vol. 15, pp. 68-71.

Quesenberry, C.P. (1991). “SPC *Q* Charts for a Binomial Parameter: Short or Long Runs,” *Journal of Quality Technology,* Vol. 23, pp. 239-246.

Quesenberry, C.P. (1991). “SPC *Q* Charts for a Poisson Parameter: Short or Long Runs,” *Journal of Quality Technology,* Vol. 23, pp. 296-303.

Rossetti, M.D., Collins, T., and Kurgund, R. (2001) “Inventory Cycle Counting – A Review,” *The proceedings of the 2001 Industrial Engineering Research Conference,* J. Rajgopal, K. Needy, Dallas, Texas.

Ryan, T.P. and Schwertman, N.C. (1997). “Optimal Limits for Attributes Control Charts,” *Journal of Quality Technology,* Vol. 29, No. 1, pp. 86-98.

Ryan, T.P. and Schwertman, N.C. (1999). “Using Dual *NP* Charts to Detect Changes,” *Quality and Reliability International,* Vol. 15, pp. 317-320.

Schreibfeder, Jon. “Cycle Counting Can Eliminate Your Annual Physical Inventory!” *EIM.* 13 Oct. 2008 <http://www.effectiveinventory.com/article9.html>.

Shewhart, W.A. (1926). “Quality Control Charts,” *Bell Systems Technical Journal,* pp. 593-603.

Shewhart, W.A. (1927). “Quality Control Charts,” *Bell Systems Technical Journal,* pp. 722-735.

Stahl, R.A. (1998). “Cycle Counting: A Quality Assurance Process,” *Hospital Material Management Quarterly,* Vol. 20, No. 2, pp. 22-28.

Tersine, R. (1994). *Principles of Inventory and Materials Management,* 4th ed., Prentice Hall, Englewood Cliffs, NJ.

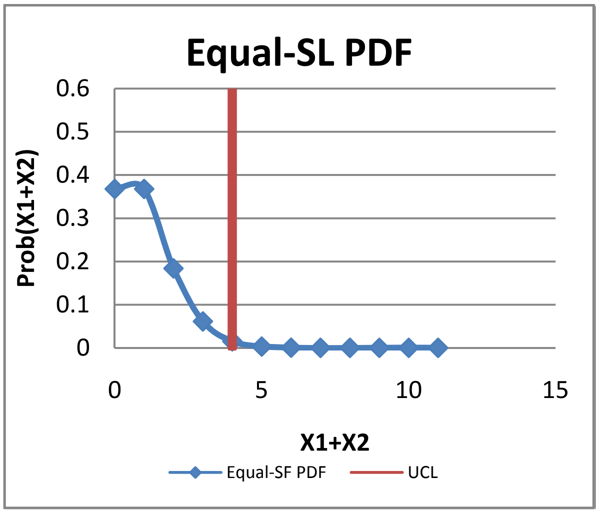
Vardeman, S., and Ray, D. (1985). “Run Lengths for CUSUM Schemes When Observations Are Exponentially Distributed,” *Technometrics,* Vol. 27, No. 2, pp. 145-150.

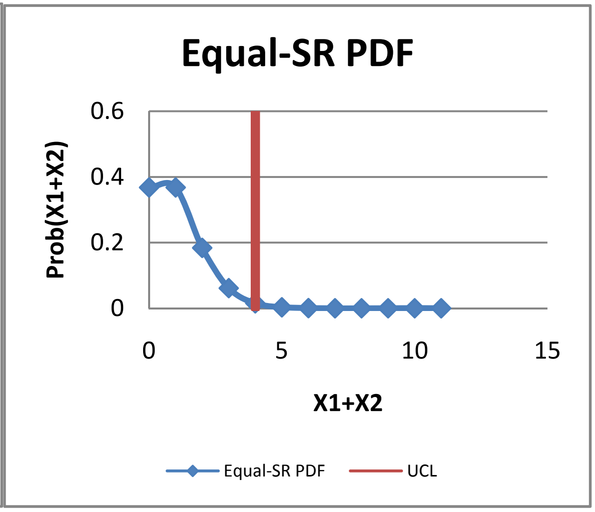
White, C.H., Keats, J.B., and Stanley, J. “Poisson CUSUM Versus C Chart for Defect Data,” *Quality Engineering,* Vol. 9(4), pp. 673-679.

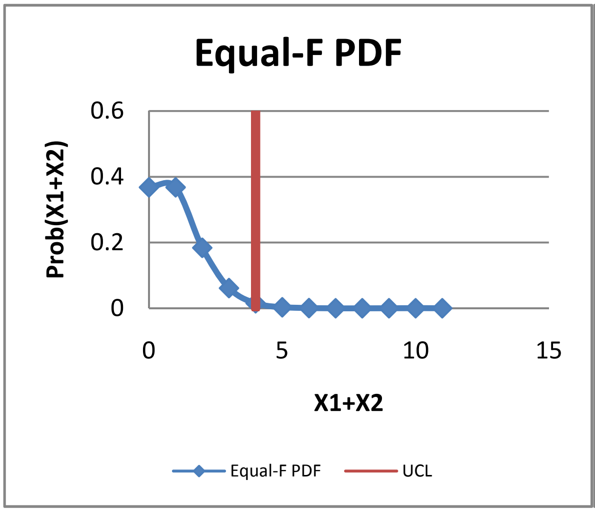
Wilson, J. (1995). “Quality Control Methods in Cycle Counting for Record Accuracy Management,” *International Journal of Operations & Production Management,* Vol. 15, No. 7, pp. 27-39.

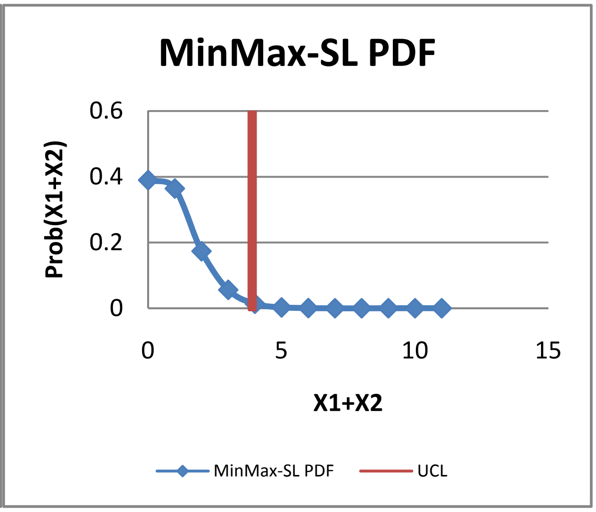
Woodall, W.H. (1997). “Control Charts Based on Attribute Data: Bibliography and Review,” *Journal of Quality Technology,* Vol. 29, No. 2, pp. 172-183.

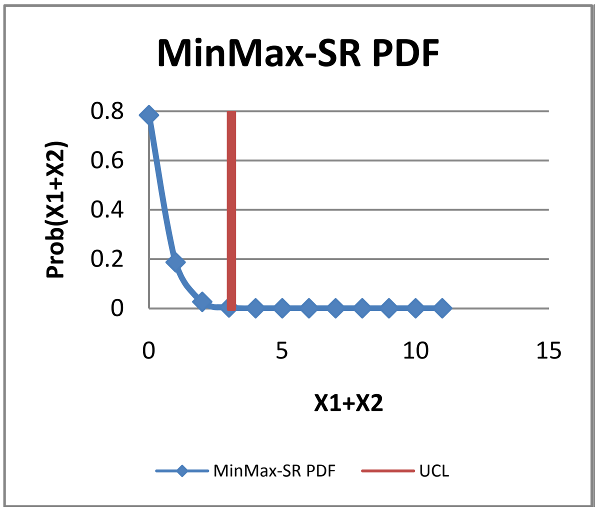
Appendix A – PDF's for n=2 Problem Structures

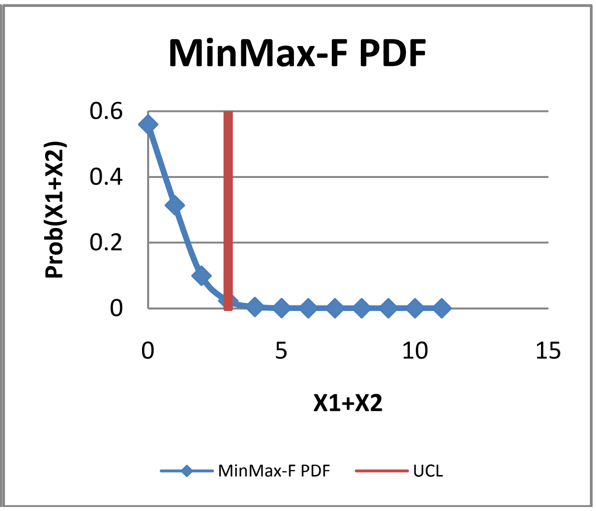

Equal-SL PDF

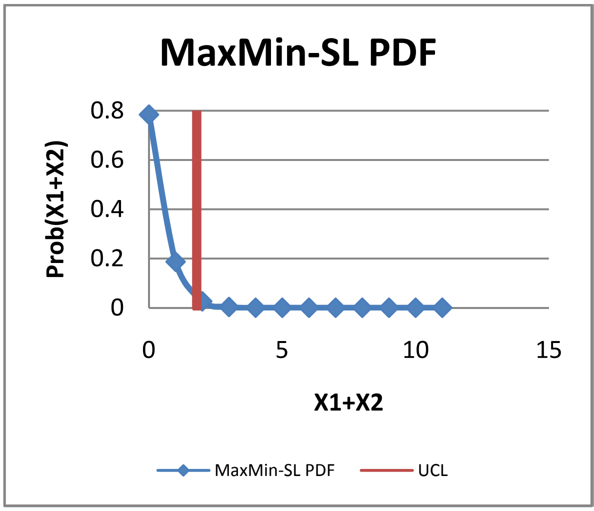

Equal-SR PDF

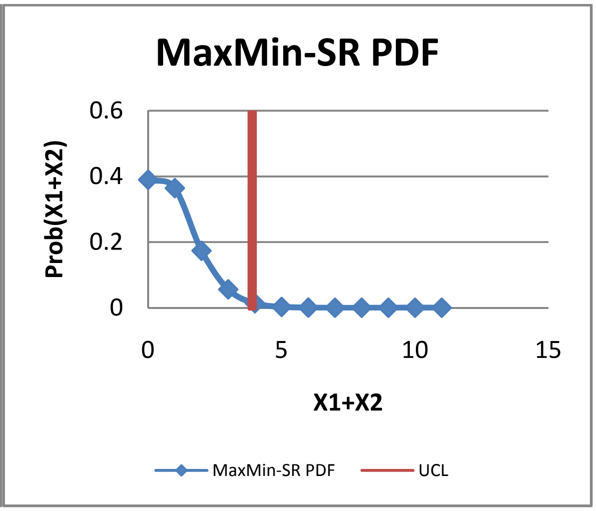

Equal-F PDF

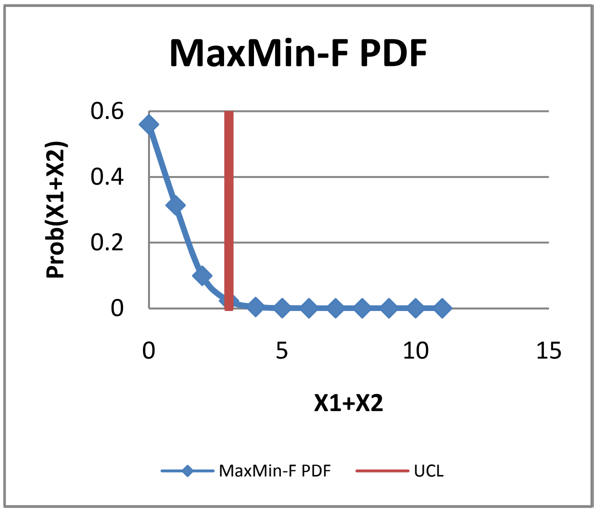

MinMax-SL PDF

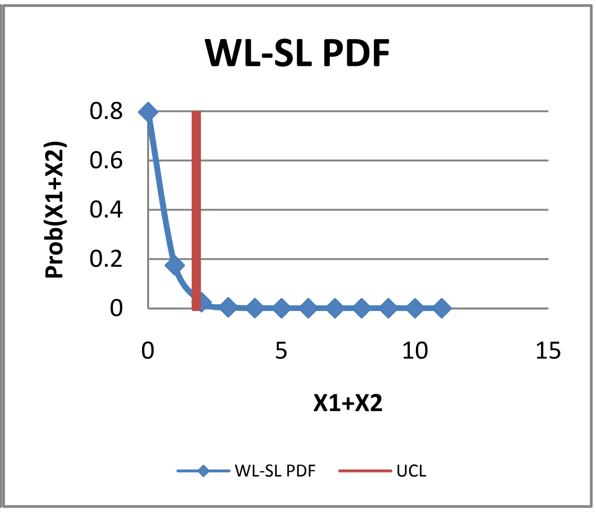

MinMax-SR PDF

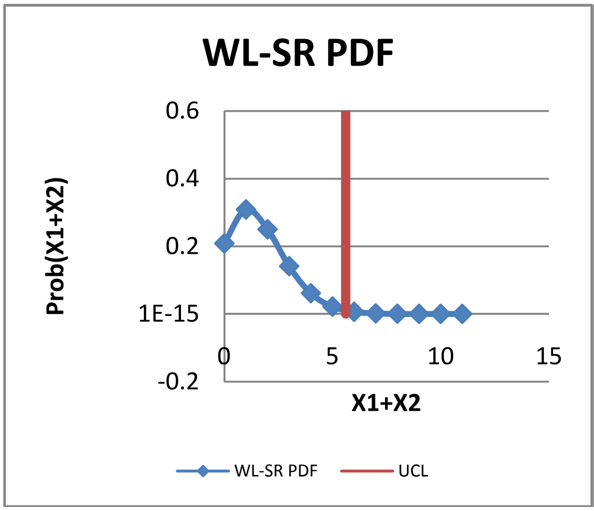

MinMax-F PDF

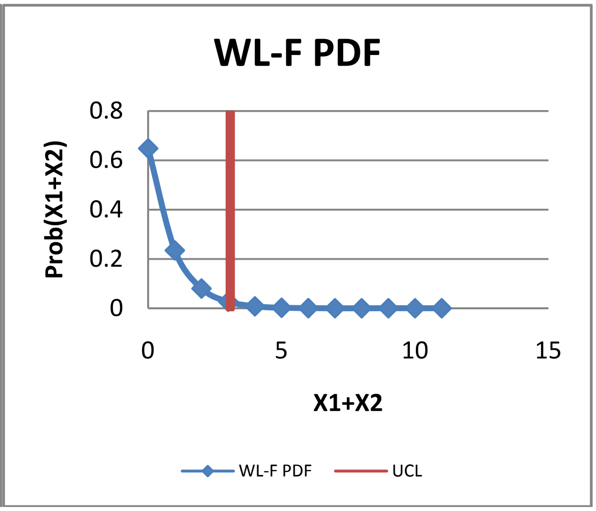

MaxMin-SL PDF
Prob(X1+X2)
0.8
0.6
0.4
0.2
0
0
5
10
15
X1+X2
MaxMin-SL PDF
UCL

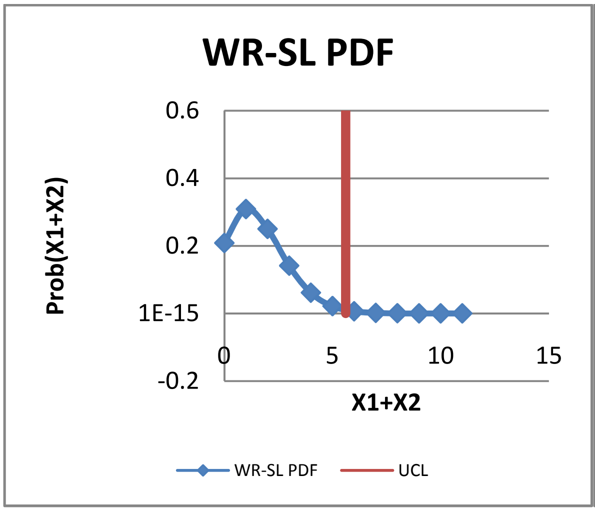

MaxMin-SR PDF
Prob(X1+X2)
0.6
0.4
0.2
0
0
5
10
15
X1+X2
MaxMin-SR PDF
UCL

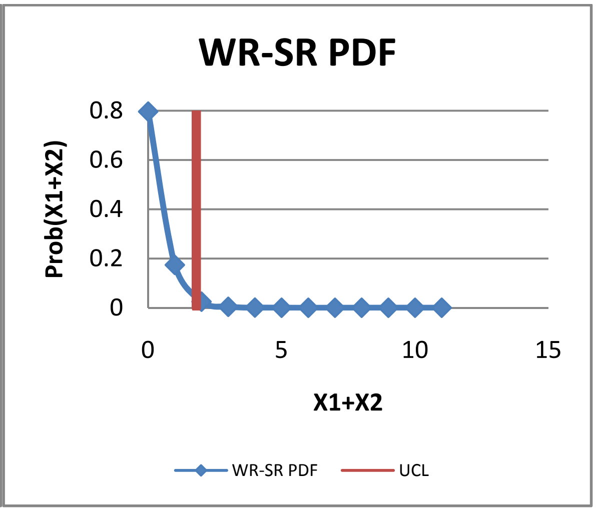

MaxMin-F PDF
Prob(X1+X2)
0.6
0.4
0.2
0
0
5
10
15
X1+X2
MaxMin-F PDF
UCL

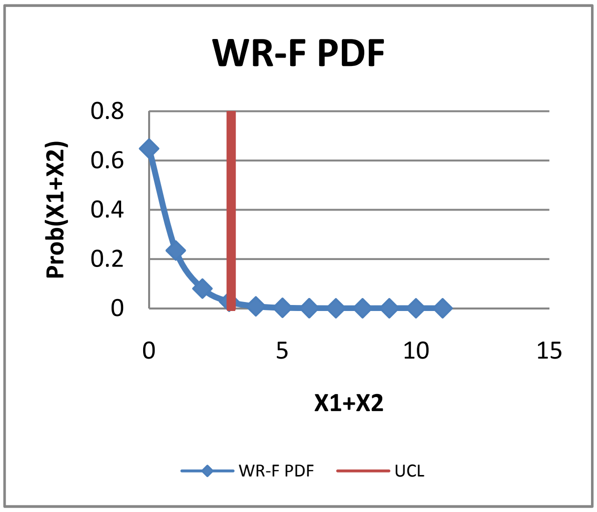

WL-SL PDF
Prob(X1+X2)
0.8
0.6
0.4
0.2
0
0
5
10
15
X1+X2
WL-SL PDF
UCL


WL-SR PDF
Prob(X1+X2)
0.6
0.4
0.2
1E-15
-0.2
0
5
10
15
X1+X2
WL-SR PDF
UCL

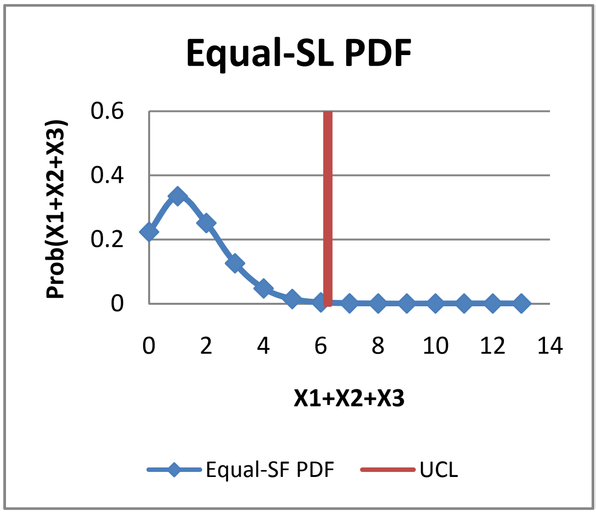

WL-F PDF
Prob(X1+X2)
0.8
0.6
0.4
0.2
0
0
5
10
15
X1+X2
WL-F PDF
UCL

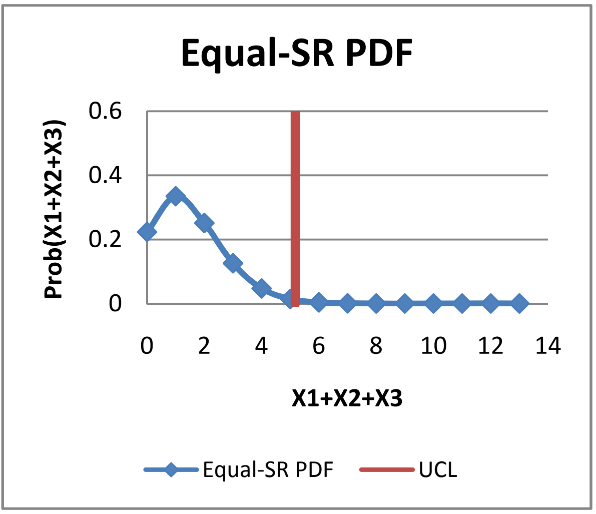

WR-SL PDF
Prob(X1+X2)
0.6
0.4
0.2
1E-15
-0.2
0
5
10
15
X1+X2
WR-SL PDF
UCL

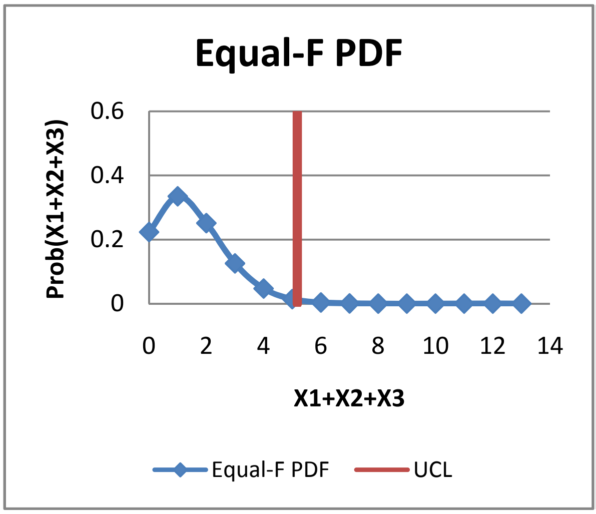

WR-SR PDF
Prob(X1+X2)
0.8
0.6
0.4
0.2
0
0
5
10
15
X1+X2
WR-SR PDF
UCL

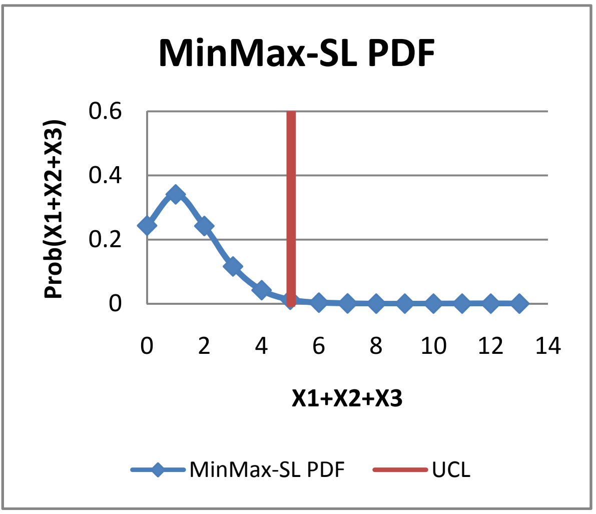

WR-F PDF
Prob(X1+X2)
0.8
0.6
0.4
0.2
0
0
5
10
15
X1+X2
WR-F PDF
UCL

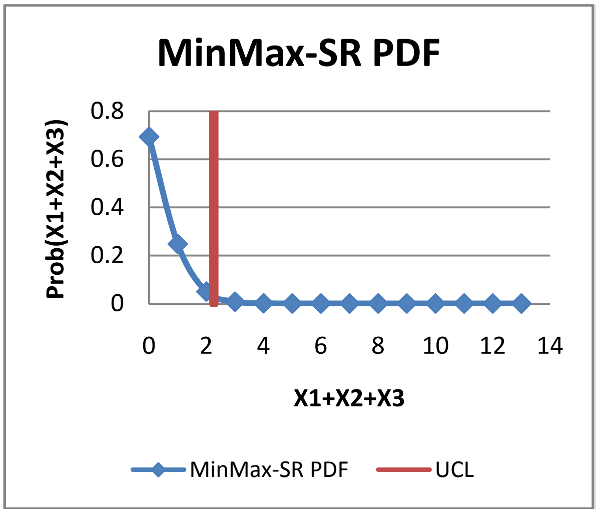
Appendix B – PDF's for n=3 Problem Structures

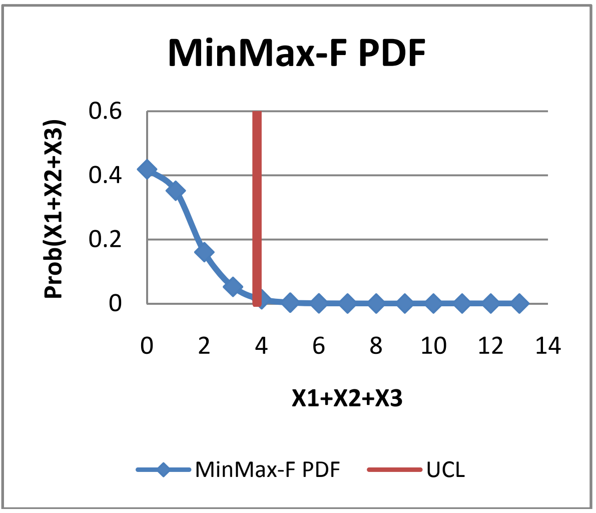


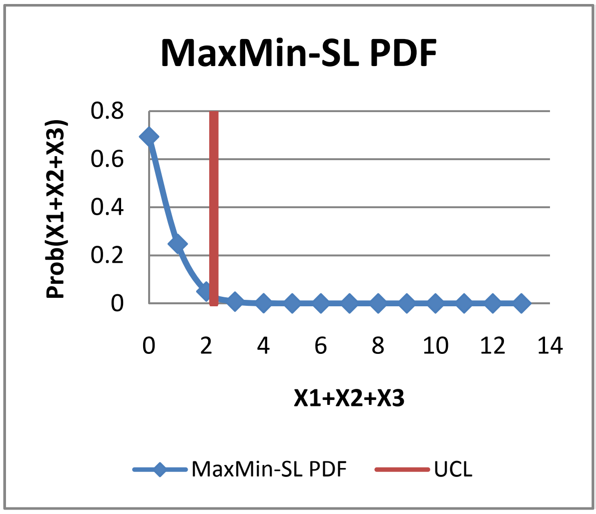


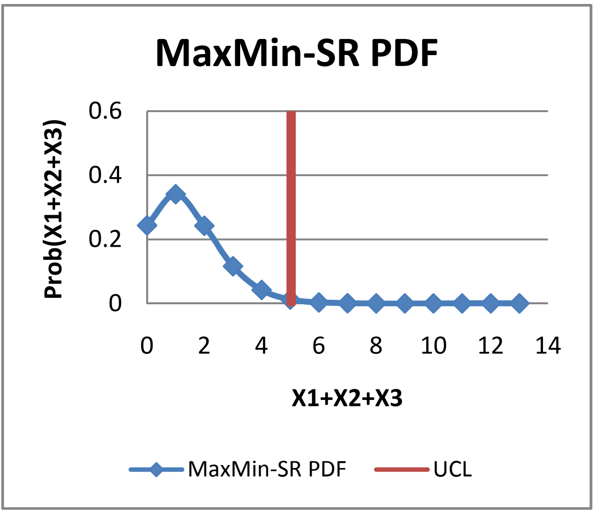


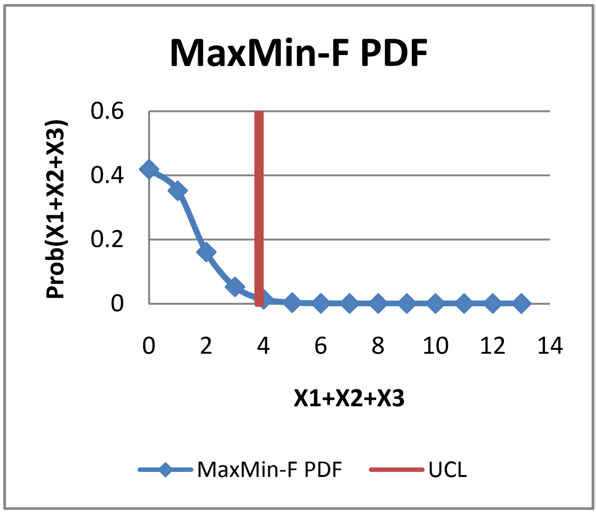


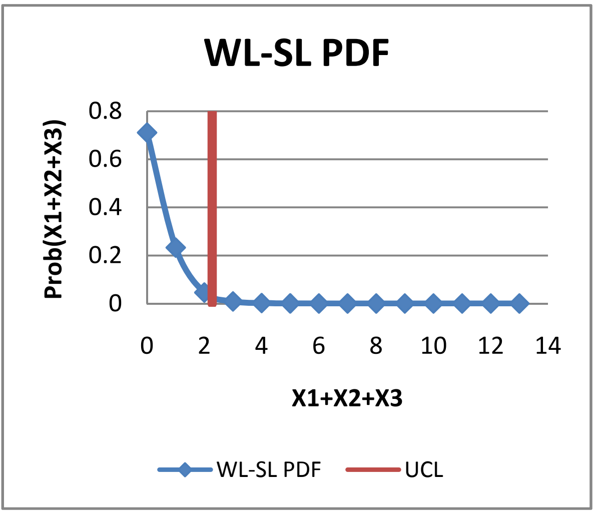


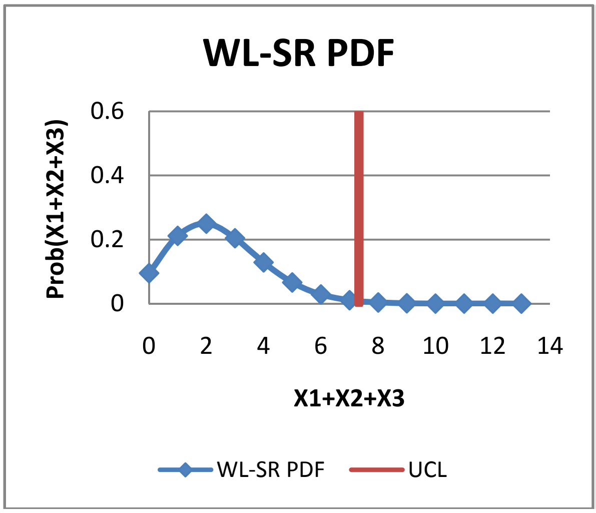


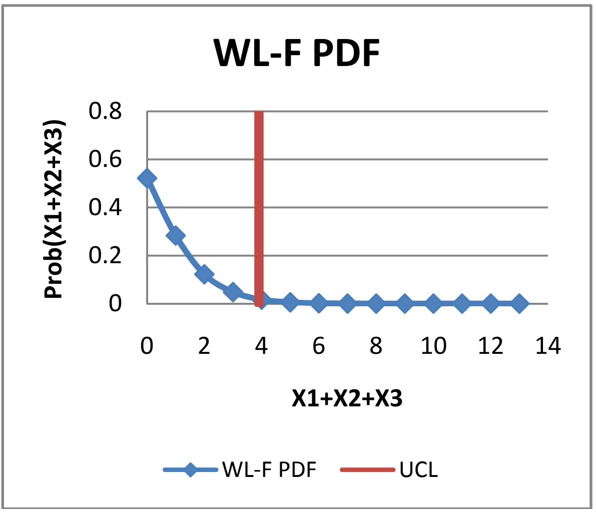

MaxMin-SL PDF

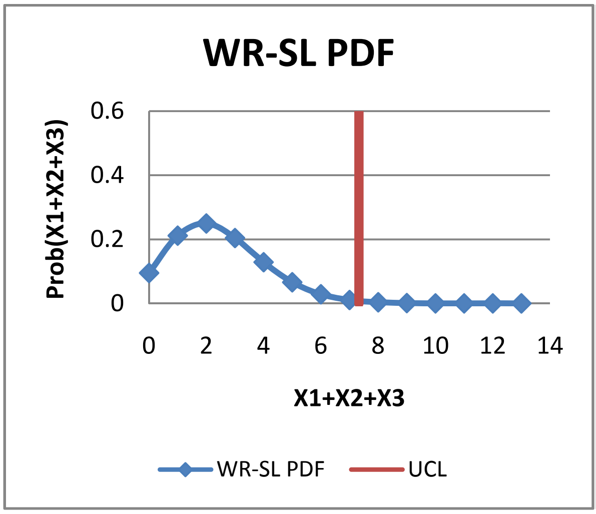

MaxMin-SR PDF

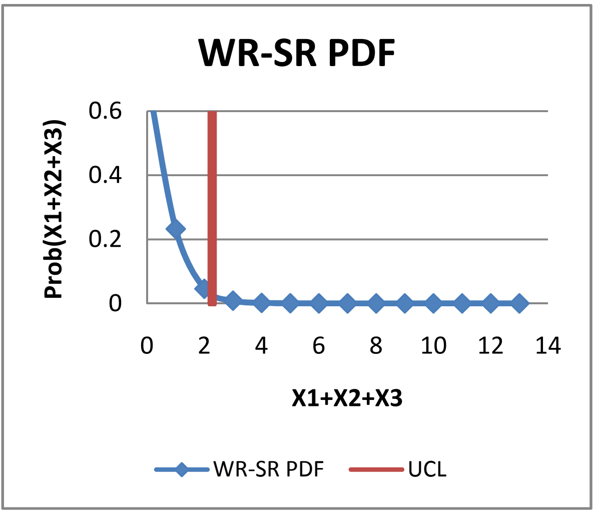

MaxMin-F PDF

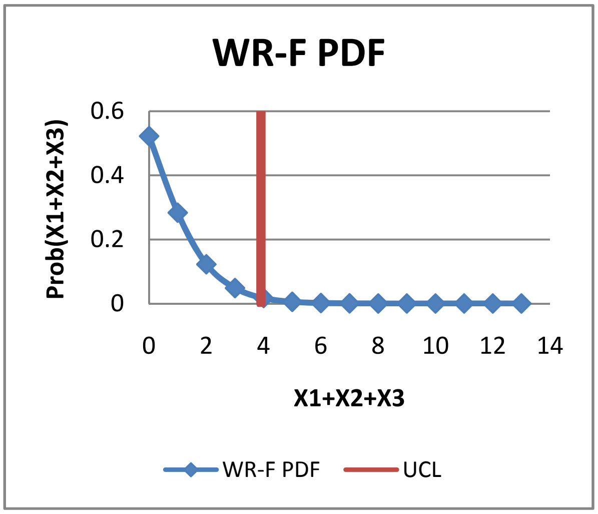

WL-SL PDF


WL-SR PDF


WL-F PDF


WR-SL PDF
Prob(X1+X2+X3)
0.6
0.4
0.2
0
0
2
4
6
8
10
12
14
X1+X2+X3
WR-SL PDF
UCL


WR-SR PDF
Prob(X1+X2+X3)
0.6
0.4
0.2
0
0
2
4
6
8
10
12
14
X1+X2+X3
WR-SR PDF
UCL


WR-F PDF
Prob(X1+X2+X3)
0.6
0.4
0.2
0
0
2
4
6
8
10
12
14
X1+X2+X3
WR-F PDF
UCL

Appendix C - C++ Program for Conditional Probabilities

```
#include "stdafx.h"
#include <time.h>
#include <ctype.h>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include "math.h"
#include "defines.h"

#define MAXLAMBDA 11
#define MAXALPHA  11
#define MAXDIM  11
#define MAXUCL  200


double factorial (int);
int _tmain(int argc, _TCHAR* argv[])
{
    int i,j,k;
    double sum, alpha[MAXDIM], lambda[MAXDIM];
      int ucl, dim;
      double prob[MAXUCL], old[MAXUCL], probb[MAXUCL];

      dim=4;
      ucl=10;
      alpha[0]=.9;
      alpha[1]=.025;
      alpha[2]=.025;
      alpha[3]=.025;
      alpha[4]=.025;
      lambda[0]=.2;
      lambda[1]=.2;
      lambda[2]=.2;
      lambda[3]=.2;
      lambda[4]=.2;

      for (i=0; i<=ucl; i++) {
            prob[i]=0;
      }
      for (i=0; i<=ucl; i++) {
            for (j=0; j<=ucl; j++) {
                  prob[i]+=alpha[j]*(pow(lambda[j],i)*pow(2.718,((-
                  1)*lambda[j])) /factorial(i));
            }
      }
      for (i=0; i<=ucl; i++) {
            old[i]=prob[i];
            printf(" value = %d prob %f old %f\n", i, prob[i], old[i]);

      }
      for (i=1; i<=dim; i++) {
            printf("dim = %d\n", i+1);
            for (j=0; j<=ucl; j++) {
                  sum =0;
                  for (k=0; j-k>=0; k++) {
                        sum+=prob[j-k]*old[k];
                  }
                  probb[j] = sum;
            }
            for (j=0; j<=ucl; j++) {
                  prob[j]=probb[j];
                  printf (" the sum of xs equals %d  has prob %f\n", j,
                  prob[j]);
            }
      }
}
double factorial (int i) {
      double fact;

      printf("%d ",i);
      if (i<=1) return (1);

      fact=1;
      while (i!=1) {
            fact=fact*(double) i;
            i--;
      }
      printf(" factorial %f \n", fact);
      return((double)fact);
}
```

Appendix D – C++ Program for Simulating Practical Sized Problems

```
#include "stdafx.h"
#include <time.h>
#include <ctype.h>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include "math.h"
#include "defines.h"

#define MAXLAMBDA 11
#define MAXALPHA  11
#define MAXDIM  100
#define MAXUCL  200


double factorial (int);
int _tmain(int argc, _TCHAR* argv[])
{
    int sum, q,i,j,k,x,h, number[MAXDIM];
    double  pr[MAXDIM];
      int ucl, dim, lcl;
      double random;
      FILE *fp;

      fp = fopen("output.csv", "w");

      dim=15;
      ucl=77;
      lcl=24;

      pr[0]=0.060122;
      pr[1]=0.20564;
      pr[2]=0.396243;
      pr[3]=0.579624;
      pr[4]=0.727105;
      pr[5]=0.833046;
      pr[6]=0.903301;
      pr[7]=0.946934;
      pr[8]=0.97243;
      pr[9]=0.986454;
      pr[10]=0.99371;
      pr[11]=0.997241;
      pr[12]=0.998856;
      pr[13]=0.999551;
      pr[14]=0.999833;
      pr[15]=0.999941;
      pr[16]=0.99998;
      pr[17]=0.999994;
      pr[18]=0.999998;
      pr[19]=0.999999;
      pr[20]=1;
      for(h=0; h<100; h++) {

      srand(1350+1000*h);

      sum=0;
      j=0;
      q=1;
      while (sum<=ucl){
            j++;
            sum=0;
            for(i=0; i<dim; i++) {
                  x=rand();
                  random=(double) x/RAND_MAX;
                  k=0;
                  while (random>=pr[k]) {
                        k++;
                  }
                  sum+=k;
                  printf("k=%d sum %d\n", k, sum);
                  number[i]=k;
            }
            for(i=0; i<dim; i++) {
                  printf (" %d ", number[i]);
            }
            printf("\n");

            if(sum<=lcl){
                  sum=ucl+1;
                  q=0;
            }
      }
      printf("iteration %d\n", j);
      fprintf(fp,"%d,%d\n", j, q);
      }

      fclose(fp);


}
```