ABSTRACT

ECONOMIC DESIGN OF A SINGLE CUSUM CHART WITH COMBINED ONE-SIDED MONITORING OF PROCESS MEAN

by

Matthew Bergenn

April, 2019

Chair: Dr. Chris Carolan

Major Department: Mathematics

A single, two-sided CUSUM chart utilizing continuously variable sampling intervals and continuously variable sample sizes monitors a process mean and is optimized through an economic design metric. The combined CUSUM statistic is capable of detecting positive and negative shifts simultaneously in one chart, which relies on consecutive indications of either an increase or decrease in mean. A family of polynomial shapes define the rate at which the minimum sample size/maximum sampling interval sweeps to the maximum sample size/minimum sampling interval as the combined CUSUM statistic approaches the boundary. All possible transition probabilities are derived and nine parameters are optimized by minimizing a long-run hourly cost function using 16 different scenarios, varying costs and times spent in/out of control.

ECONOMIC DESIGN OF A SINGLE CUSUM CHART WITH COMBINED ONE-SIDED MONITORING OF PROCESS MEAN

A Thesis

Presented to

The Faculty of the Department of Mathematics

East Carolina University

In Partial Fulfillment

of the Requirements for the Degree

Master of Arts in Mathematics

by

Matthew Bergenn

April, 2019

ECONOMIC DESIGN OF A SINGLE CUSUM CHART WITH COMBINED ONE-SIDED MONITORING OF PROCESS MEAN

by

Matthew Bergenn

APPROVED BY:

DIRECTOR OF THESIS: Dr. Chris Carolan

COMMITTEE MEMBER: Dr. Peng Xiao

COMMITTEE MEMBER: Dr. Jungmin Choi

COMMITTEE MEMBER: Dr. John Kros

COMMITTEE MEMBER: Dr. Chris Jantzen

CHAIR OF THE DEPARTMENT OF MATHEMATICS: Dr. Johannes Hattingh

DEAN OF THE GRADUATE SCHOOL: Dr. Paul Gemperline

ACKNOWLEDGEMENTS

I would first like to thank my advisor, Dr. Chris Carolan, for years of thoughtful instruction which prepared me well for research and the writing of this thesis. His sharing of ideas and his time devotion are gifts for which my gratitude could never be fully expressed. I would also like to thank my committee members for their insightful advice and time devotion. Specifically, I would like to thank Dr. Kros for his help in revising, in addition to his significant contribution to the ideas presented in this work. I am also grateful to Drs. Choi, Jantzen, and Xiao for the many ways they positively impacted my education at ECU. I would also like to thank my fiance, Erica Clark, for her unwavering support and patience. Finally, I would like to thank my family for always encouraging me.

TABLE OF CONTENTS

1 Introduction 1

2 Literature review 2

3 A combined CUSUM scheme 3

3.1 The combined CUSUM statistic 3

3.2 Discretizing the CCSUM for use of a Markov chain 4

3.3 Continuously variable sampling intervals and sample sizes 6

3.4 Economic design of the CCSUM model 9

3.5 Utilizing Markov chain properies 10

4 Numerical Analysis 12

4.1 Optimization method and procedures 12

4.2 Optimizing parameters for sample size and sampling interval 13

4.3 Results and comparison of $\overline{X}$ and CCSUM chart designs 15

4.4 Incorporating statistical design into an economic design model 20

5 Conclusions 23

References 24

A Derivation of transition probabilities 25

B R code 30

Chapter 1: Introduction

Cumulative sum (CUSUM) control charts have been noted for their usefulness in quality control since their development in the 1950's by Page[3]. Typical CUSUM statistic calculations involve computing a standardized sample average, comparing its extremity to a reference parameter, k, and adding it to the previous CUSUM statistic. Subsequent samples of size n, taken at some sampling interval, h, may eventually cause the statistic to cross the boundary, b, at which point the process may be stopped while the search for an assignable cause ensues. A variety of methods can be used to determine the most effective values for these parameters, k, b, n, and h.

One such method is through statistical design[9], where certain constraints are applied for the purpose of satisfying some desirable statistical properties. For example, it may be advantageous to limit the rate of false signals or the time the process runs out-of-control. This can be achieved by an increase in sampling, which will naturally increase the cost of process monitoring. Note that an increase in sampling can come in the form of larger sample sizes, shorter sampling intervals, or a combination of both.

An alternative method to statistical design is economic design[2], which determines parameter values by balancing sampling costs, false signals, and out-of-control time. A cost function incorporating these metrics is minimized, revealing the most efficient parameter settings. This paper utilizes the economic design method to find the optimal parameter settings. However, it should be noted that incorporating statistical design into an economic design model would not be difficult, and would produce the most cost efficient method of obtaining the desired statistical properties.

Chapter 2: Literature review

Previous research utilizing CUSUM techniques to monitor process control often delivered superior performance measures over other control charts, especially at detecting small to moderate shifts in mean[4]. However, this increase in effectiveness was offset by implementation difficulties due to the nature of their complex designs. Reynolds and Stoumbos[7] proposed a method of statistical process control which consisted of two, one-sided CUSUM charts monitoring both positive and negative shifts in mean in addition to one CUSUM chart measuring deviation in variance. Such schemes have proved to be very efficient, however, they are often difficult to put into practice and develop. This is due to the number of chart parameters necessary in addition to the amount of simulation required to assess their statistical properties.

Attempts to simplify CUSUM chart process monitoring have been explored, either through the simplification of the CUSUM statistic or by some other method of parameter reduction. Wu et al. investigated the combining of mean and variance into a single statistic[11] and also by reducing the the two-chart mean monitoring into a single chart that measures mean deviation by its absolute value or distance from its target[10]. In both cases, two-sided mean detection is possible. However, CUSUM statistics that incorporate squaring or taking the absolute value of mean deviation do not keep track of the sign of the shift, positive or negative. In these schemes, extreme sample statistics consecutively landing on opposite sides of the target, however unlikely, are summed as deviations without regard to their sign and therefore, information is lost as well as some level of sensitivity in shift detection[10].

Chapter 3: A combined CUSUM scheme

3.1 The combined CUSUM statistic

This paper proposes the monitoring of process mean by a CUSUM chart which merges two, one-sided CUSUMs into a single chart. The process is assumed to begin in-control and produce units that are normally distributed with mean, μ, and standard deviation, σ. The time until shifting out-of-control is distributed exponentially with rate, λ. Furthermore, an out-of-control process is considered as producing units with mean, $\mu \pm \Delta\sigma$, and standard deviation, σ. At the initiation of the monitoring process, the value of the combined CUSUM (CCSUM) statistic is zero, i.e., $C_{t=0} = 0$. Once the first sample is taken, the current CCSUM statistic, C_t is calculated by,

$$C_t = \begin{cases} \max\left\{0\,,\ C_{t-1} + s \cdot \text{Trunc}\left(\dfrac{Z_t - k}{s}\right)\right\}, & \text{if } C_{t-1} > 0 \text{ and } Z_t > -k \\[2em] \min\left\{0\,,\ C_{t-1} + s \cdot \text{Trunc}\left(\dfrac{Z_t + k}{s}\right)\right\}, & \text{if } C_{t-1} < 0 \text{ and } Z_t < k \\[2em] \text{sign}(Z_t) \cdot \max\left\{0\,,\ s \cdot \text{Trunc}\left(\dfrac{|Z_t| - k}{s}\right)\right\}, & \text{otherwise} \end{cases}$$

where C_{t-1} is the previous CCSUM statistic, Z_t is the current standardized sample average given by $Z_t = \left(\overline{X_t} - \mu\right) / \left(\sigma/\sqrt{n_t}\right)$, k is a reference parameter, and s is the step size by which the CCSUM statistic is allowed to increment.

The CCSUM statistic is calculated according to three cases. The first of which is used when the previous CCSUM statistic is positive, $C_{t-1} > 0$, and the current standardized sample average is greater than the lower reference parameter, $Z_t > -k$. These first case conditions can be summarized in the following statement; if there have been recent indications of an increase in mean, the CCSUM statistic prioritizes the detection of a positive shift. Detecting a positive shift continues as the main focus until significant evidence points to the contrary, namely, obtaining a $Z_t \leq -k$. When $C_{t-1} > 0$ and $Z_t > -k$, the CCSUM statistic will either progress toward the boundary ($Z_t \geq k+s$), stagnate ($k-s < Z_t < k+s$), or regress toward zero ($Z_t \leq k-s$). Note that the CCSUM statistic can regress by at most $2k$ standard deviations, or more specifically, the largest multiple of s less than this value.

The second case of the CCSUM statistic is handled similarly to the first, but rather serves to prioritize the detection of a negative shift in mean. The third and final case of the CCSUM statistic is used when the previous CCSUM statistic is zero, $C_{t-1} = 0$, or when neither of the conditions are met for the first two cases. The latter happens during the monitoring of one side of the chart, positive or negative, and a significant standardized sample average of opposite sign is obtained, at which point the sign of the most recent sample average becomes the primary focus of mean shift detection.

In this paper, the CCSUM statistic is designed for its use in conjunction with a Markov chain approach in determining charting parameters. The following section introduces this approach to prepare the reader for later references to Markov chain properties and the optimization algorithm.

3.2 Discretizing the CCSUM for use of a Markov chain

The resulting CCSUM statistic is forced into a discrete value by use of the truncate function so that the procedure can be modeled by a Markov chain. The matrix of transition probabilities (P matrix) is indexed by discrete states, 0 to b, in increments of s, where the boundary, b, is a multiple of s. An arbitrary state of the P matrix,

i steps of s from the zero state, is labeled $\pm i \cdot s$, coupled with a control status. Because the CCSUM statistic monitors mean deviation on only one side of its target at a time, each non-zero state in the P matrix represents both directions of shift, thus minimizing the size of the P matrix. The options for control status include in-control, I, and not in-control, N. However, at the moment a process goes out-of-control, a non-zero CCSUM statistic can find itself on either the same or opposite side of the mean shift, thus creating the need for two different out-of-control statuses, N_C and N_D. N_C specifies a CCSUM statistic on the same side (concordant) of the shift, while N_D indicates the CCSUM statistic is on the opposite side (discordant) of the shift. To illustrate the necessity of these additional statuses, consider an example where the current CCSUM statistic is $2 \cdot s$. If the process is out-of-control and the shift is positive, the current state is $(2 \cdot s, N_C)$. Alternatively, if the process is out-of-control and the shift is negative, the current state is $(2 \cdot s, N_D)$.

Define $r = b/s$ and $m = 3r$ so that the P matrix has order $m + 1$. The ordering of all possible states and their respective indices in the P matrix are provided below.

$$
\begin{aligned}
1 &\equiv (0, I) \\
2 &\equiv (\pm 1 \cdot s, I) \\
3 &\equiv (\pm 2 \cdot s, I) \\
\vdots & \qquad \vdots \\
r &\equiv (\pm(b - s), I) \\
r + 1 &\equiv \text{False signal} \\
r + 2 &\equiv (0, N) \\
r + 3 &\equiv (\pm 1 \cdot s, N_C) \\
r + 4 &\equiv (\pm 2 \cdot s, N_C)
\end{aligned}
$$

$$
\begin{aligned}
&\vdots \qquad\qquad \vdots \\
2r+1 &\equiv (\pm(b-s), N_C) \\
2r+2 &\equiv (\pm 1 \cdot s, N_D) \\
2r+3 &\equiv (\pm 2 \cdot s, N_D) \\
&\vdots \qquad\qquad \vdots \\
3r &\equiv (\pm(b-s), N_D) \\
3r+1 &\equiv \text{True signal}
\end{aligned}
$$

Indexing the states is necessary, since expected values will be derived later using properties of Markov chains. The next section discusses how sample sizes and sampling intervals are determined, and is more intuitively discussed in terms of the absolute value of the current CCSUM statistic.

3.3 Continuously variable sampling intervals and sample sizes

Variable sampling intervals (VSI) have been shown to produce more efficient models of process control monitoring as opposed to fixed intervals. Reynolds et al.[6] investigated VSI in conjunction with CUSUM charts showing an improvement in efficiency in its application to process control. Rendtel[5] later contributes the idea of varying both variable sampling intervals and sampling sizes (VSSI) in the context of CUSUM schemes. The logic behind variable sampling intervals and sample sizes is that when a process is in-control and the CCSUM statistic is close to its target, it is more cost efficient to have larger sampling intervals and smaller sample sizes. Conversely, if it is suspected that the process is out-of-control and the CCSUM statistic is wandering toward its alarm boundary, it is desirable to take larger samples, more quickly. Much of what has been written on this topic includes a multi-stage procedure, such as a two-stage or a three-stage procedure, in accomplishing VSSI[5]. In a two-stage VSSI scheme, for example, a warning zone is designated as an intermediate detection area prior to reaching the boundary, such that a CUSUM statistic entering this region will cause a transition from a more conservative setting for sample size and sampling interval, $(n_1\,,\,h_1)$, to a setting with increased sampling, $(n_2\,,\,h_2)$.

A unique sampling interval and sample size could be individually optimized for each state in the proposed CCSUM scheme. However, this would create the need for an unwieldy array of parameters, and lead to an extremely time-consuming optimization procedure. Carolan et al.[1] proposed continuously variable sampling intervals in conjunction with X-bar ($\overline{X}$) control charts by calculating the probability of obtaining a subsequent statistic closer to the boundary and multiplying by a parameter representing the maximum sampling interval. The maximum of the resulting value and the minimum sampling interval is taken as the sampling interval associated with the current state. This idea allowed for the sampling interval to sweep continuously from an optimal maximum value to a minimum based on its proximity to the boundary.

This paper introduces a variation on this idea in that the rate at which the maximum sampling interval (or minimum sample size) decreases (or increases) with respect to the proportion of states traversed, of all states prior to reaching the boundary, is defined by a family of polynomial shapes indexed by 3 parameters (minimum, maximum, shape). The formulas are given by,

$$n_{i\cdot s} = n_{min} + n_{range}\left(\frac{i\cdot s}{b-s}\right)^{\alpha_n}$$

and

$$h_{i\cdot s} = h_{min} + h_{range}\left[1-\left(\frac{i\cdot s}{b-s}\right)\right]^{\alpha_h}$$

for sample size and sampling interval, respectively. A unique sampling interval and sample size (before necessarily rounding) are assigned to each state, by row, in the P matrix as a result. The variety of polynomial shapes attainable range from linear to concave up or down by letting the parameter, α, take any value in the domain, $0 < \alpha < \infty$. In addition to a shape parameter, a minimum and maximum sample size/sampling interval are required for parameterization. Therefore, the defining parameters are (n_{min} , n_{max} , α_n) and (h_{min} , h_{max} , α_h), for sample size and sampling interval calculation, respectively. The functions governing sample sizes and sampling intervals are plotted in Figure 3.1. These graphs aim to show the variety of shapes possible, and it should be noted that rounding sample size output values to the nearest integer creates a step function that will retain the general shape of the polynomial.

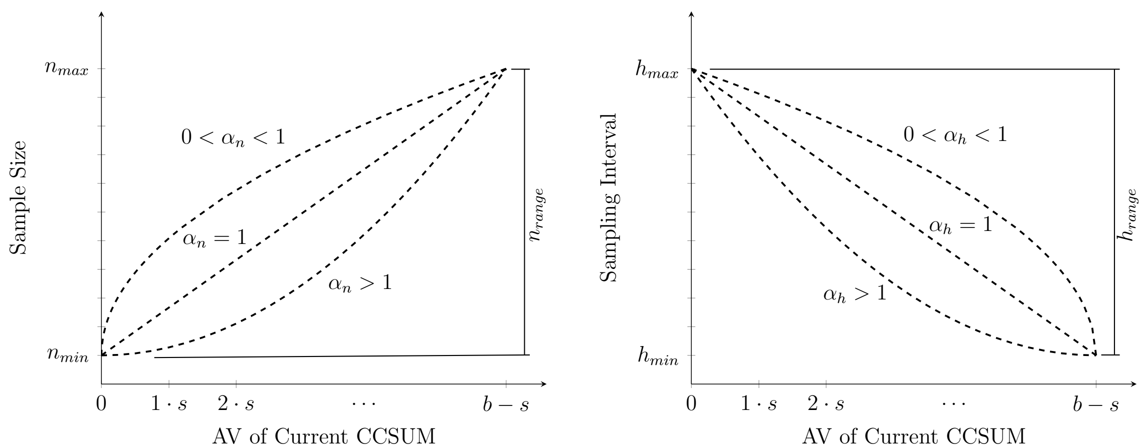

Figure 3.1

A variety of methods were investigated in order to attain the optimal shape for these curves. During the optimization process, it was discovered that changing the shape of the function that defines the rate of increase or decrease has a large impact on the long-run hourly cost of production. This is especially true concerning the function that governs the sampling interval, $h_{i \cdot s}$. For instance, both shape parameters were initially commissioned to compound the proportion of the states traversed with respect to how far they were from the zero state. However, it was later determined that allowing the shape parameter, α_h, to compound the proportion of remaining states dramatically improved the results. This is due to the inherent properties of the shapes produced when very large or very small values for α_h are found to be optimal. While the optimal function governing sample sizes is relatively linear in nature, i.e. $\alpha_n \approx 1$, the optimal function defining the sampling interval sweeps drastically towards its minimum upon leaving state zero. This requirement is more efficiently met in the case where the proportion of remaining states is compounded by α_h.

3.4 Economic design of the CCSUM model

Economic design employs a long-run hourly cost (LRHC) function to determine the optimal parameters for each scenario of cost/time settings. The optimal parameters are those which minimize LRHC, the expected cost of one cycle divided by the expected time to complete one cycle. A cycle begins at the start of a process and ends when an assignable cause is repaired. The LRHC is calculated by,

$$LRHC = \frac{c_1\, E[N] \;+\; c_2\, E[OOCT] \;+\; c_3\, E[F] \;+\; c_4}{E[PT] \;+\; t_1\, E[F] \;+\; t_2}$$

where the constants and expected values are defined below.

c_1	the sampling cost per unit
c_2	the out-of-control cost per hour
c_3	the repair cost per signal (false)
c_4	the repair cost per signal (true)
t_1	the time spent per signal (false)
t_2	the time spent per signal (true)

$E[PT]$	the expected number of hours spent in production
$E[OOCT]$	the expected number of hours spent out-of-control
$E[N]$	the expected number of units sampled per cycle
$E[F]$	the expected number of false signals

The four terms in the numerator of the LRHC function represent the total costs per cycle in regards to sampling, running out-of-control, false signals, and repairing a true signal. The three terms that define the denominator represent the total time spent in production per cycle, the total time caused by false signals per cycle, and the total time spent repairing an assignable cause. The expected values in the LRHC function are calculated using the properties of Markov chains while the remaining variables are necessarily predefined.

3.5 Utilizing Markov chain properies

The P matrix of order $m+1$, as previously defined, must first be reduced to a matrix containing only transient states, denoted by P_T. When a true signal is received, production ceases, and the search for an assignable cause ensues. Therefore, the only absorbing state in the P matrix lies in the true signal state, whereby the corresponding row and column is removed to find P_T, which has order m. Calculate $S = [I - P_T]^{-1} = \{s_{ij}\}$, of which the first row contains the expected number of visits to each state throughout one cycle of the process. Thus, the expected values from the LRHC formula are provided below.

$$E[OOCT] = E[PT] - \frac{1}{\lambda}$$

where

$$E[PT] = \sum_{j=1}^{r} s_{1j} \cdot h_{(j-1)\cdot s} \; + \sum_{j=r+2}^{2r+1} s_{1j} \cdot h_{(j-r-2)\cdot s} \; + \sum_{j=2r+2}^{3r} s_{1j} \cdot h_{(j-2r-1)\cdot s}$$

The values for $E[N]$ and $E[F]$ are derived as follows.

$$E[N] = \sum_{j=1}^{r} s_{1j} \cdot n_{(j-1)\cdot s} \; + \sum_{j=r+2}^{2r+1} s_{1j} \cdot n_{(j-r-2)\cdot s} \; + \sum_{j=2r+2}^{3r} s_{1j} \cdot n_{(j-2r-1)\cdot s}$$

and

$$E[F] = s_{1\,,\,r+1}$$

The next section explains the process of parameter optimization as well as the resulting benefits of the proposed CCSUM scheme. In direct comparison to previous research, a detailed analysis is presented and the incorporation of statistical design is discussed.

Chapter 4: Numerical Analysis

Comparisons are made between the results of this paper to the previous work of Carolan et al.[1] where $\overline{X}$ control charts with continuously variable sampling intervals and two stage sample sizes are employed. Tabular results are notated with the subscripts, $\overline{X}$, where findings from Carolan et al.[1] are listed, and CCSUM, when referring to the discoveries of this paper. The in-text reference to the work of Carolan et al.[1] will be referred to as the $\overline{X}$ chart model.

4.1 Optimization method and procedures

Eight constants are used in conjunction with the LRHC formula in determining the settings of the 9 optimal parameters in each of the 16 scenarios (examples) via the P matrix. Since a solution for optimizing these 9 parameters is not possible through analysis, a grid search method is used for the task. The grid search was performed using the R software environment, and an example of the R code used for this paper is provided in the appendix. The fixed settings per each scenario are provided in Table I, where each constant is defined as stated in the previous chapter.

It was discovered during optimization that reducing the value of certain parameters, in their respective domains, led to increased efficiency in terms of LRHC. For example, the step size, s, continued to be more optimal for decreasing positive values approaching zero. This is expected since smaller values of s lead to a more continuous state space for the Markov chain, which translates to a more sensitive detection of mean shift. Due to limitations in computer processing memory, the value of s has a necessary lower limit. For this paper, the step size is defined to be $s = 0.01$, for all scenarios. The minimum sampling interval, h_{min}, is another parameter that op-

Table I. Process parameters (constants)								
Fixed settings per example								
Ex.	c_1	c_2	c_3	c_4	t_1	t_2	λ	Δ
1	2	500	1500	1000	2	1	0.01	0.5
2	5	500	1500	1000	2	1	0.01	0.5
3	2	1500	1500	1000	2	1	0.01	0.5
4	5	1500	1500	1000	2	1	0.01	0.5
5	2	500	3000	1000	5	1	0.01	0.5
6	5	500	3000	1000	5	1	0.01	0.5
7	2	500	3000	1000	5	1	0.01	0.5
8	5	1500	3000	1000	5	1	0.01	0.5
9	2	500	1500	1000	2	1	0.01	1
10	5	500	1500	1000	2	1	0.01	1
11	2	1500	1500	1000	2	1	0.01	1
12	5	1500	1500	1000	2	1	0.01	1
13	2	500	3000	1000	5	1	0.01	1
14	5	500	3000	1000	5	1	0.01	1
15	2	500	3000	1000	5	1	0.01	1
16	5	1500	3000	1000	5	1	0.01	1

timizes at its lowest allowable setting. This should also come as no surprise, since smaller values of h_{min} mean that once the process drifts significantly away from its target, it is more cost efficient to sample as quickly as possible to either drive the CCSUM statistic back to zero or further advance it across the boundary. However, there is a minimum feasible sampling interval for every manufacturing process, to be determined by the scientist. In this study, sampling can occur once every 3 minutes at a minimum, i.e., $h_{min} = 0.05$ for all scenarios.

4.2 Optimizing parameters for sample size and sampling interval

After completing the optimization procedure, trends were apparent regarding the 6 parameters allocated to govern sample sizes and sampling intervals. For instance, while the minimum and maximum sample size, n_{min} and n_{max}, can be quite varied across all scenarios, the value for α_n, the shape parameter, remained relatively consistent. The average value for α_n among all scenarios is 1.80; this means that, on average, the polynomial shape describing the rate of transition from n_{min} to n_{max} is approximately quadratic. Conclusions can be drawn from the values of α_h received through optimization as well. Although, the range of these values is much larger, there is not much variation in terms of the shapes of the sampling interval functions overall. Therefore, it may be informative to analyze one scenario in particular, for the purpose of gaining a better general idea of what is optimal for any given scenario.

The following two figures display the optimal settings for sample sizes and sampling intervals from scenario 4. Figure 4.1 displays the graph of the sample size function. The function is designed to output rounded values during optimization, hence, the step function graph. The sweep from $n_{min} = 10$ to $n_{max} = 21$ could even

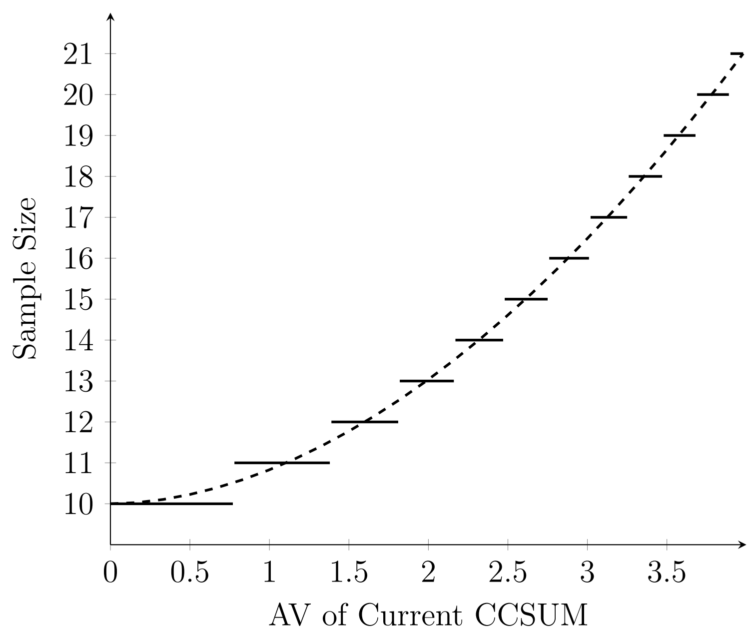

Figure 4.1

be described as relatively linear in shape and is indicative of the general shape of $n_{i \cdot s}$ for all scenarios. Figure 4.2 presents the graph of the sampling interval function resulting from scenario 4. This illustrates the aforementioned drastic sweep to h_{min} that all scenarios require in minimizing LRHC. This optimal shape for sampling interval can be summarized in the following statement; as the CCSUM statistic begins to drift away from state zero, it is economically more efficient to sample more quickly right away than it is to immediately begin taking larger sample sizes.

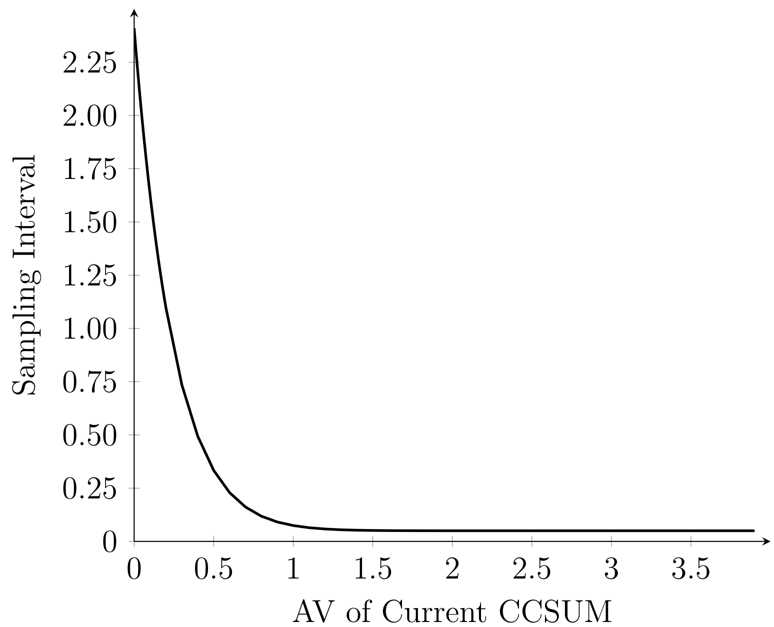

Figure 4.2

4.3 Results and comparison of $\overline{X}$ and CCSUM chart designs

Since the settings per each scenario used in this paper are identical to the $\overline{X}$ chart model, the effectiveness of the CCSUM chart design and procedures can be measured by direct comparison. Table II displays the final optimized parameter settings for the variable components of the CCSUM chart design, in addition to the LRHC results from each of the two proposed designs and subsequent savings. The savings are

Table II. Process parameters (variables), results, and comparison

Optimized settings per example										Results/comparison		
Ex.	b	s	k	h_{min}	h_{max}	α_h	n_{min}	n_{max}	α_n	$\mathrm{LRHC}_{\mathrm{CCSUM}}$	$\mathrm{LRHC}_{\overline{X}}$	% Savings
1	4.02	0.01	0.94	0.05	3.13	20.68	13	21	1.59	37.96	40.28	5.76%
2	3.63	0.01	0.88	0.05	4.86	15.37	12	16	1.68	53.70	57.13	6.00%
3	3.95	0.01	0.96	0.05	1.86	20.39	14	31	2.03	59.25	62.91	5.82%
4	3.99	0.01	0.84	0.05	2.41	15.78	10	21	1.87	86.52	92.19	6.15%
5	4.97	0.01	0.93	0.05	3.15	24.70	13	25	1.64	38.39	40.58	5.40%
6	4.55	0.01	0.90	0.05	4.52	22.56	11	18	1.23	54.63	57.89	5.63%
7	4.75	0.01	0.98	0.05	1.85	27.01	14	37	1.75	59.94	63.22	5.19%
8	4.48	0.01	0.93	0.05	2.84	22.01	13	25	1.77	87.81	93.19	5.77%
9	3.96	0.01	1.11	0.05	1.99	26.07	5	10	1.99	24.48	25.41	3.66%
10	4.04	0.01	0.99	0.05	2.82	21.09	4	7	1.90	32.35	34.28	5.63%
11	3.79	0.01	1.15	0.05	1.13	30.89	5	17	2.04	35.76	37.50	4.64%
12	3.53	0.01	1.11	0.05	1.81	23.39	5	10	2.22	49.89	52.71	5.35%
13	4.66	0.01	1.13	0.05	1.98	33.75	5	12	1.69	24.64	25.76	4.35%
14	4.32	0.01	1.10	0.05	3.19	27.45	5	8	1.73	32.93	34.55	4.69%
15	4.50	0.01	1.16	0.05	1.12	37.93	5	22	1.90	36.07	37.69	4.30%
16	4.60	0.01	1.06	0.05	1.56	32.97	4	13	1.73	49.95	53.11	5.95%

presented as a percent decrease of the CCSUM LRHC from the $\overline{X}$ LRHC as given by the formula below.

$$\% \text{ Savings} = \frac{\text{LRHC}_{\overline{X}} - \text{LRHC}_{\text{CCSUM}}}{\text{LRHC}_{\overline{X}}} \cdot 100\%$$

The results shown in Table II clearly point to an improvement of design in the CCSUM model over the $\overline{X}$ model. Savings were especially robust in the scenarios where a smaller shift required detecting. This is expected due to the information retaining properties of CUSUM charts in general.

A detailed cost breakdown is provided in Table III, where individual cost and time values per scenario are presented from both the $\overline{X}$ chart and CCSUM chart models. Included in this table are the first three terms of the LRHC function's numerator along with two time metrics, all of which are defined in the previous chapter. However, the concept of lag time has yet to be discussed or defined. Recall the assumption that the time until the process shifts out-of-control is distributed exponentially with rate, λ. For all scenarios, this rate is set to be $\lambda = 0.01$ (see Table I), which implies that the expected time until the process shifts out-of-control is 100 hours. For the purpose of this paper, lag time is defined as the amount of elapsed time from when the process shifts out-of-control until any indication is given that a problem exists, or more specifically, until the next sample is taken. Therefore, lag time is calculated by,

$$\text{Lag} = \sum_{1}^{r} s_{1j} \cdot h_{(j-1)\cdot s} - \frac{1}{\lambda}$$

where $\{s_{ij}\} = [I - P_T]^{-1}$ as defined in the last chapter. Note that the product being summed is the amount of time spent in each in-control state, where s_{1j} is the expected number of visits to each state, $h_{j\cdot s}$ is the sampling interval associated with each state, and r is the number of in-control states excluding the false signal state.

The CCSUM chart model boasts superior cost savings to the $\overline{X}$ chart model, in addition to lowering both the expected time spent out of control and the expected number of false signals in all 16 scenarios. In 6 scenarios, the $\overline{X}$ chart model has a shorter lag time than the CCSUM model, as highlighted in Table III. This simply means that in the $\overline{X}$ chart model, a sample is taken sooner after a shift occurs. However, in each of these 6 scenarios, the CCSUM model detects the shift earlier than the $\overline{X}$ model due to spending less time out-of-control. While lag times do not have much bearing on the overall efficiency throughout a cycle, they do provide some insight on the detection properties of different control chart schemes.

Table III. Cost breakdown

Example	N cost	OOC cost	F cost	$E[OOCT]$	Lag	$E[F]$
$1_{\overline{X}}$	1542	1606	50	3.211	1.764	0.035
1_{CCSUM}	1421	1497	30	2.994	1.471	0.020
\$ savings	121	109	20	= 250/cycle		
$2_{\overline{X}}$	2367	2547	159	5.094	2.735	0.106
2_{CCSUM}	2254	2360	65	4.721	2.259	0.044
\$ savings	113	187	94	= 394/cycle		
$3_{\overline{X}}$	2613	2778	87	1.852	1.002	0.058
3_{CCSUM}	2431	2609	49	1.740	0.871	0.033
\$ savings	182	169	38	= 389/cycle		
$4_{\overline{X}}$	4062	4315	228	2.876	1.461	0.152
4_{CCSUM}	3826	4052	100	2.701	1.096	0.067
\$ savings	236	263	128	= 627/cycle		
$5_{\overline{X}}$	1567	1618	48	3.245	1.764	0.016
5_{CCSUM}	1453	1513	28	3.027	1.475	0.009
\$ savings	114	105	20	= 239/cycle		
$6_{\overline{X}}$	2462	2572	123	5.144	2.862	0.041
6_{CCSUM}	2336	2385	58	4.770	2.117	0.019
\$ savings	126	187	65	= 378/cycle		

Table III. Cost breakdown (continued)						
Example	N cost	OOC cost	F cost	$E[OOCT]$	Lag	$E[F]$
$7_{\overline{X}}$	2647	2806	76	1.871	1.002	0.025
7_{CCSUM}	2470	2648	43	1.765	0.871	0.014
$ savings	177	158	33	= 368/cycle		
$8_{\overline{X}}$	4169	4362	181	2.908	1.528	0.060
8_{CCSUM}	3942	4092	77	2.728	1.328	0.026
$ savings	227	270	104	= 601/cycle		
$9_{\overline{X}}$	760	829	20	1.658	0.872	0.013
9_{CCSUM}	721	777	12	1.554	0.956	0.008
$ savings	39	52	8	= 99/cycle		
$10_{\overline{X}}$	1223	1289	40	2.578	1.125	0.027
10_{CCSUM}	1129	1195	21	2.390	1.334	0.014
$ savings	94	94	19	= 207/cycle		
$11_{\overline{X}}$	1308	1486	32	0.991	0.588	0.022
11_{CCSUM}	1238	1385	22	0.923	0.544	0.015
$ savings	70	101	10	= 181/cycle		
$12_{\overline{X}}$	2090	2255	62	1.504	0.645	0.041
12_{CCSUM}	1962	2114	34	1.409	0.869	0.023
$ savings	128	141	28	= 297/cycle		
$13_{\overline{X}}$	772	850	25	1.699	0.712	0.008
13_{CCSUM}	732	786	10	1.572	0.955	0.003
$ savings	40	64	15	= 119/cycle		
$14_{\overline{X}}$	1235	1300	47	2.600	1.125	0.016
14_{CCSUM}	1170	1218	18	2.437	1.538	0.006
$ savings	65	82	29	= 176/cycle		
$15_{\overline{X}}$	1322	1494	30	0.996	0.585	0.010
15_{CCSUM}	1260	1397	19	0.932	0.540	0.006
$ savings	62	97	11	= 170/cycle		
$16_{\overline{X}}$	2090	2289	73	1.526	0.648	0.024
16_{CCSUM}	1965	2120	31	1.413	0.745	0.010
$ savings	125	169	42	= 336/cycle		

4.4 Incorporating statistical design into an economic design model

In this paper, the charting parameters are optimized by strict economic design, which yields the setting with the lowest LRHC. Statistical design chooses the appropriate settings according to some desired statistical properties, such as low average run length or low false signal rate. Previous work by Saniga[8] explores metrics which incorporate both economic and statistical design and strike a balance between the two design methods. While process monitoring is an expense to the manufacturer and quality control is of utmost importance to most, it could be said that economic design is an ideal starting ground for the incorporation of statistical design. Strict economic design establishes the baseline of process monitoring costs; incorporating statistical design forces the manager to decide how much, in cost, he or she is willing to give up in order to gain a desired statistical property or properties.

Similar to the interplay between type I and type II error probabilities in hypothesis testing, the expected number of false signals in a cycle, $E[F]$, and the mean time the process runs out of control in a cycle, $E[OOCT]$ are inversely related. Increasing the boundary, b, the reference parameter, k, and/or the sampling intervals, $h_{i \cdot s}$, all result in lowering $E[F]$ at the expense of raising $E[OOCT]$. Conversely, lowering these settings will lower $E[OOCT]$ at the expense of raising $E[F]$.

Interestingly, increasing the sample sizes, $n_{i \cdot s}$, only serves to decrease $E[OOCT]$, but at no expense to $E[F]$ which remains stationary. The counterbalance here is increased sampling costs. Once one recognizes this fact, it becomes clear that some economic consideration is appropriate. So, the balance between $E[F]$ and $E[OOCT]$ also should involve a consideration of cost.

Economic design, through the LRHC function, does exactly this. It seeks to keep both $E[F]$ and $E[OOCT]$ low, with cost efficiency in mind. Incorporating statistical design into economic design, called economic statistical design[8], simply selects the most cost effective setting from among all statistically viable settings. For example, if the design was required to have the properties $E[F] \leq 0.05$ and $E[OOCT] \leq 1.5$ hours, then the economic statistical design procedure would select the most efficient design among all designs with the required statistical properties.

Consider an example of incorporating statistical design in scenario 4. The expected time spent out-of-control per cycle, as given in Table III, is $E[OOCT] = 2.701$ hours, as determined by strict economic design. If a manager were to demand that $E[OOCT] \leq 2$ hours, he or she could expect an increase in sampling costs. Using economic statistical design, the optimal settings can be derived to establish the most cost effective way to limit the mean out-of-control time to no more than 2 hours per cycle. Table IV displays the optimal settings and the cost breakdown, illustrating the results of this example. In this table, example 4 refers to the settings derived from strict economic design and example 4* refers to the settings using economic statistical design.

Table IV. Incorporating statistical design using scenario 4

Optimized settings per example										Results
Example	b	s	k	h_{min}	h_{max}	α_h	n_{min}	n_{max}	α_n	LRHC
4	3.99	0.01	0.84	0.05	2.41	15.78	10	21	1.87	86.52
4*	4.05	0.01	0.86	0.05	1.83	15.70	11	19	1.94	90.14

Cost breakdown per example						
Example	N cost	OOC cost	F cost	$E[OOCT]$	Lag	$E[F]$
4	3826	4052	100	2.701	1.096	0.067
4*	5192	3000	99	2.000	0.832	0.066
\$ savings	-1366	1052	1	$= -313$/cycle		

Notice that there are adjustments made in all parameters not held constant. However, the most significant difference in parameter optimization is in the shortening of the maximum sampling interval, a reduction of almost 25%. A smaller range of sample sizes was found to be optimal, and yet, in both cases they are centered around 15 to 16 units. This narrowed range of sample size values points to the de-emphasis of its contribution to lowering out-of-control time. In other words, the CCSUM statistic will gain more information from sampling more quickly rather than sampling a larger number of units at a time. Shorter sampling intervals allow the CCSUM statistic to quickly reach the boundary once a process has shifted out-of-control.

The cost breakdown in Table IV shows the monetary trade-off when requiring $E[OOCT] \leq 2$. In finding the most cost efficient setting for achieving this statistical property, the boundary and reference values were increased, thus lowering the expected number of false signals and associated costs. A reduction in lag time should be expected, since reducing h_{min} means less time elapses between sampling. More specifically, less time will elapse between the time the process goes out of control until another sample is taken. Clearly, an increase in sampling costs and a decrease in out-of-control costs will result. However, it is interesting to find out that a manager who wishes to implement this design, under scenario 4 settings, should expect to spend an extra \$3.62 per hour, compared to the hourly cost determined by strict economic design.

Chapter 5: Conclusions

The topic of process control has been widely discussed and researched, where CUSUM schemes lead the way in the detection of small to moderate shifts in mean. Some research[10][11] tackle the implementation difficulties due to overly complex schemes where a high level of efficiency are shown. However, in the calculation of the CUSUM statistic, information regarding the sign of the shift is lost due to the method of simplification, and thus some sensitivity of mean detection is compromised. This paper proposes a CUSUM statistic calculation that differentiates between positive and negative trends in shift detection.

In addition to the novelty of the statistic, this paper provides a method of assigning a sample size and unique sampling interval to each CCSUM statistic, where each depends on only 3 parameters (minimum, maximum, shape). Variable sampling intervals and sample sizes have long been employed and shown to improve the results of process control[5]. However, the extent to which variability is explored has not been fully investigated, most likely due to concerns with an excess in parameterization.

Through an economic design metric, the improved efficiency of the CCSUM scheme is unveiled through direct comparison with the previous work of Carolan et al.[1]. The process monitoring method proposed in this paper shows considerable improvement in all categories, exhibited by the Tables II and III in the previous chapter. Specifically, the improvement in LRHC shown in Table II is made possible by the individual cost savings presented in Table III. To summarize, an overall increase in efficiency can be attributed to a reduction in sampling, a decrease in time spent while out-of-control, and a lower false signal rate, all of which are important to managers.

REFERENCES

[1] Christopher A. Carolan, John F. Kros, and Said E. Said. Economic design of xbar control charts with continuously variable sampling intervals. *Quality and Reliability Engineering International*, 26(3):235–245, 2010.

[2] DC Montgomery. The economic design of control charts: a review and literature survey. *Journal of Quality Technology*, 12(2):75–87, 1980.

[3] ES Page. Continuous inspection schemes. *Biometrika*, 41(1):100–115, 1954.

[4] ES Page. Cumulative sum schemes using gauging. *Technometrics*, 4(1):97–109, 1962.

[5] U Rendtel. Cusum-schemes with variable sampling intervals and sample sizes. *Statistical Papers*, 31(1):103, 1990.

[6] MR Reynolds Jr, RW Amin, and JC Arnold. Cusum charts with variable sampling intervals. *Technometrics*, 32(4):371–396, 1990.

[7] MR Reynolds Jr and ZG Stoumbos. Should observations be grouped for effective process monitoring? *Journal of Quality Technology*, 36(4):343–366, 2004.

[8] EM Saniga. Economic statistical control-chart designs with an application to xbar and r charts. *Technometrics*, 31(3):313–320, 1989.

[9] WH Woodall. The statistical design of quality control charts. *Journal of the Royal Statistical Society. Series D (The Statistician)*, 34(2):155–160, 1985.

[10] Z Wu, M Yang, MBC Khoo, and F Yu. Optimization designs and performance comparison of two cusum schemes for monitoring process shifts in mean and variance. *European Journal of Operational Research*, 205(1):136–150, 2010.

[11] Z Wu, S Zhang, and PH Wang. A cusum scheme with variable sample sizes and sampling intervals for monitoring the process mean and variance. *Qual. Reliab. Engng. Int.*, 23(2):157–170, 2007.

Appendix A: Derivation of transition probabilities

All non-zero probability statements are given below except for $P_{False\,(0,I)} = 1$ and $P_{True\,True} = 1$. A CCSUM statistic reaching the boundary while the process is in-control produces a false signal, and the CCSUM statistic returns to 0. If a CCSUM statistic reaches the boundary and the process is out-of-control, a true signal is produced, production ceases, and an assignable cause is identified and corrected. Hence, the true signal serves as an absorbing state in the P matrix.

When the process is in-control, the units produced are assumed to be normally distributed with mean, μ, and standard deviation, σ. The transition probabilities to in-control states are written in terms of a standard normal random variable, $Z \sim N(0,1)$. When the process is not in-control, the assumption is that the mean has shifted Δ standard deviations while the standard deviation remains unaffected. Therefore, a process that is not in-control produces units with mean, $\mu \pm \Delta\sigma$, and standard deviation, σ. Consider the case where a positive shift in mean occurs, where units now come from the distribution, $N(\mu + \Delta\sigma, \sigma)$. Therefore,

$$Z_t = \frac{\overline{X}_t - \mu}{\sigma/\sqrt{n}} = \frac{\overline{X}_t - (\mu + \Delta\sigma) + \Delta\sigma}{\sigma/\sqrt{n}} = \frac{\overline{X}_t - (\mu + \Delta\sigma)}{\sigma/\sqrt{n}} + \Delta\sqrt{n}\,.$$

This implies that the new standardized random variable is distributed, $N(\Delta\sqrt{n}, 1)$, which can be expressed by the shifted standard normal random variable, $Z + \Delta\sqrt{n}$. The negative case is handled similarly and results in the shifted standard normal random variable, $Z - \Delta\sqrt{n}$. Therefore, the transition probabilities to or from out-of-control states are expressed accordingly, i.e., $Z \pm \Delta\sqrt{n_{i\cdot s}}$. In the following probability statements, the notations, Z^{Δ^+} and Z^{Δ^-}, are used for $Z + \Delta\sqrt{n_{i\cdot s}}$ and $Z - \Delta\sqrt{n_{i\cdot s}}$, respectively.

There are 26 unique cases that define all possible transition probabilities. Recall that states in the P matrix are ordered pairs of the form, (CCSUM statistic, control status). The probability statements below are organized by their row in the P matrix, i.e., the state it is transitioning from. Determining transition probabilities from one state to another requires addressing the control status transition and conditionally addressing the CCSUM statistic transition. If a process is out-of-control, it is assumed that it will remain out-of-control until a true signal is received. Therefore, out-of-control states only transition to other out-of-control states. In this paper, the time, T, until a process shifts out-of-control is assumed to be exponentially distributed with rate, $\lambda = 0.01$. The sampling interval associated with the absolute value of the current CCSUM statistic, $| \pm i \cdot s|$, ranging from 0 to $b - s$, is notated as $h_{i \cdot s} = \{h_{max}\,,\ h_{1 \cdot s}\,,\ h_{2 \cdot s}\,,\ \ldots\,,\ h_{b-2 \cdot s}\,,\ h_{min}\}$. Likewise, each state has a corresponding sample size given by $n_{i \cdot s} = \{n_{min}\,,\ n_{1 \cdot s}\,,\ n_{2 \cdot s}\,,\ \ldots\,,\ n_{b-2 \cdot s}\,,\ n_{max}\}$.

Transition from state $x = (0, I)$:

- To state $y = (0, I)$ with probability,

$P_{xy} = P(-k - s < Z < k + s) \cdot P(T < h_{max})$

- To state $y = (\pm j \cdot s, I)$ with probability,

$P_{xy} = P\big(k + j \cdot s < |Z| < k + (j + 1) \cdot s\big) \cdot P(T < h_{max})$

- To state $y = False$ with probability,

$P_{xy} = P(|Z| > k + b) \cdot P(T < h_{max})$

- To state $y = (0, N)$ with probability,

$P_{xy} = P(-k - s < Z + \Delta\sqrt{n_{min}} < k + s) \cdot P(T > h_{max})$

- To state $y = (\pm j \cdot s, N_C)$ with probability,

$P_{xy} = P(k + j \cdot s < Z + \Delta\sqrt{n_{min}} < k + (j + 1) \cdot s) \cdot P(T > h_{max})$

- To state $y = (\pm j \cdot s, N_D)$ with probability,

$P_{xy} = P(k + j \cdot s < Z - \Delta\sqrt{n_{min}} < k + (j + 1) \cdot s) \cdot P(T > h_{max})$

- To state $y = True$ with probability,

$P_{xy} = P(|Z + \Delta\sqrt{n_{min}}| > k + b) \cdot P(T > h_{max})$

Transition from state $x = (\pm i \cdot s, I)$:

- To state $y = (0, I)$ with probability,

$P_{xy} = P(-k - s < Z < \max\{-k,\, k - i \cdot s\}) \cdot P(T < h_{i \cdot s})$

- To state $y = (\pm j \cdot s, I)$ with probability, $P_{xy} =$

$$\begin{cases} \big[P(\max\{-k,\, k-(i-j+1)\cdot s\} < Z < \max\{-k,\, k-(i-j)\cdot s\}) \\ \quad + P(-k-(j+1)\cdot s < Z < -k - j\cdot s)\big] \cdot P(T < h_{i\cdot s}) & \text{if } i > j \\ \big[P(k + (j-i)\cdot s < Z < k + (j-i+1)\cdot s) \\ \quad + P(-k-(j+1)\cdot s < Z < -k - j\cdot s)\big] \cdot P(T < h_{i\cdot s}) & \text{if } i < j \\ \big[P(\max\{-k,\, k-s\} < Z < k+s) \\ \quad + P(-k-(j+1)\cdot s < Z < -k - j\cdot s)\big] \cdot P(T < h_{i\cdot s}) & \text{if } i = j \end{cases}$$

- To state $y = False$ with probability,

$P_{xy} = \big[P(Z > k + b - i \cdot s) + P(Z < -k - b)\big] \cdot P(T < h_{i \cdot s})$

- To state $y = (0, N)$ with probability,

$$P_{xy} = \big[\tfrac{1}{2}P(-k - s < Z^{\Delta^+} < \max\{-k,\, k - i \cdot s\}) \\ \quad + \tfrac{1}{2}P(\min\{k, -k + i \cdot s\} < Z^{\Delta^+} < k + s)\big] \cdot P(T > h_{i \cdot s})$$

- To state $y = (\pm j \cdot s, N_C)$ with probability, $P_{xy} =$

$$\begin{cases} \big[\tfrac{1}{2}P(\max\{-k,\, k-(i-j+1)\cdot s\} < Z^{\Delta^+} < \max\{-k,\, k-(i-j)\cdot s\}) \\ \quad + \tfrac{1}{2}P(-k-(j+1)\cdot s < Z^{\Delta^-} < -k - j\cdot s)\big] \cdot P(T > h_{i\cdot s}) & \text{if } i > j \\ \big[\tfrac{1}{2}P(k + (j-i)\cdot s < Z^{\Delta^+} < k + (j-i+1)\cdot s) \\ \quad + \tfrac{1}{2}P(-k-(j+1)\cdot s < Z^{\Delta^-} < -k - j\cdot s)\big] \cdot P(T > h_{i\cdot s}) & \text{if } i < j \\ \big[\tfrac{1}{2}P(\max\{-k,\, k-s\} < Z^{\Delta^+} < k+s) \\ \quad + \tfrac{1}{2}P(-k-(j+1)\cdot s < Z^{\Delta^-} < -k - j\cdot s)\big] \cdot P(T > h_{i\cdot s}) & \text{if } i = j \end{cases}$$

• To state $y = (\pm j \cdot s, N_D)$ with probability, $P_{xy} =$

$$\begin{cases} \big[\frac{1}{2}P(\min\{k, -k+(i-j)\cdot s\} < Z^{\Delta^+} < \min\{k, -k+(i-j+1)\cdot s\}) & \\ \quad + \frac{1}{2}P(k + j\cdot s < Z^{\Delta^-} < k+(j+1)\cdot s)\big] \cdot P(T > h_{i\cdot s}) & \text{if } i > j \\ \big[\frac{1}{2}P(-k-(j-i+1)\cdot s < Z^{\Delta^+} < -k-(j-i)\cdot s) & \\ \quad + \frac{1}{2}P(k + j\cdot s < Z^{\Delta^-} < k+(j+1)\cdot s)\big] \cdot P(T > h_{i\cdot s}) & \text{if } i < j \\ \big[\frac{1}{2}P(-k-s < Z^{\Delta^+} < \min\{k, -k+s\}) & \\ \quad + \frac{1}{2}P(k + j\cdot s < Z^{\Delta^-} < k+(j+1)\cdot s)\big] \cdot P(T > h_{i\cdot s}) & \text{if } i = j \end{cases}$$

• To state $y = True$ with probability,

$$P_{xy} = \Big(\tfrac{1}{2}\big[P(Z^{\Delta^+} > k+b-i\cdot s) + P(Z^{\Delta^+} > -k-b)\big]$$
$$+\tfrac{1}{2}\big[P(Z^{\Delta^-} > k+b-i\cdot s) + P(Z^{\Delta^-} > -k-b)\big]\Big) \cdot P(T > h_{i\cdot s})$$

Transition from state $x = (0, N)$:

• To state $y = (0, N)$ with probability,

$$P_{xy} = P(-k-s < Z + \Delta\sqrt{n_{min}} < k+s)$$

• To state $y = (\pm j \cdot s, N_C)$ with probability,

$$P_{xy} = P(k + j\cdot s < Z + \Delta\sqrt{n_{min}} < k+(j+1)\cdot s)$$

• To state $y = (\pm j \cdot s, N_D)$ with probability,

$$P_{xy} = P(k + j\cdot s < Z - \Delta\sqrt{n_{min}} < k+(j+1)\cdot s)$$

• To state $y = True$ with probability,

$$P_{xy} = P(|Z + \Delta\sqrt{n_{min}}| > k+b)$$

Transition from state $x = (\pm i, N_C)$:

• To state $y = (0, N)$ with probability,

$$P_{xy} = P(-k-s < Z^{\Delta^+} < \max\{-k,\ k-i\cdot s\})$$

• To state $y = (\pm j \cdot s, N_C)$ with probability, $P_{xy} =$

$$\begin{cases} P(\max\{-k,\, k-(i-j+1)\cdot s\} < Z^{\Delta^+} < \max\{-k,\, k-(i-j)\cdot s\}) & \text{if } i > j \\ P(k+(j-i)\cdot s < Z^{\Delta^+} < k+(j-i+1)\cdot s) & \text{if } i < j \\ P(\max\{-k,\, k-s\} < Z^{\Delta^+} < k+s) & \text{if } i = j \end{cases}$$

• To state $y = (\pm j \cdot s, N_D)$ with probability,

$P_{xy} = P(-k-(j+1)\cdot s < Z^{\Delta^+} < -k-j\cdot s)$

• To state $y = True$ with probability,

$P_{xy} = P(Z^{\Delta^+} > k+b-i\cdot s) + P(Z^{\Delta^+} < -k-b)$

Transition from state $x = (\pm i, N_D)$:

• To state $y = (0, N)$ with probability,

$P_{xy} = P(-k-s < Z^{\Delta^-} < \max\{-k,\, k-i\cdot s\})$

• To state $y = (\pm j \cdot s, N_C)$ with probability,

$P_{xy} = P(-k-(j+1)\cdot s < Z^{\Delta^-} < -k-j\cdot s)$

• To state $y = (\pm j \cdot s, N_D)$ with probability, $P_{xy} =$

$$\begin{cases} P(\max\{-k,\, k-(i-j+1)\cdot s\} < Z^{\Delta^-} < \max\{-k,\, k-(i-j)\cdot s\}) & \text{if } i > j \\ P(k+(j-i)\cdot s < Z^{\Delta^-} < k+(j-i+1)\cdot s) & \text{if } i < j \\ P(\max\{-k,\, k-s\} < Z^{\Delta^-} < k+s) & \text{if } i = j \end{cases}$$

• To state $y = True$ with probability,

$P_{xy} = P(Z^{\Delta^-} > k+b-i\cdot s) + P(Z^{\Delta^-} < -k-b)$

Appendix B: R code

```
#Long Run Hourly Cost Function
LRHC <- function(b,s,k,hmin,hmax,halpha,nmin,nmax,nalpha,lam,del,
   ftime,ttime,ncost,ooccost,fcost,tcost){

r <- round(b/s)                                  #Forces search algorithm
   values to fit discrete model
b <- s*r
s <- b/r
m <- 3*r                                         #Order of the S-matrix
P <- matrix(0,m+1,m+1)                           #Zero matrix order m+1
states <- seq(0,b-s,by=s)                        #Creates a vector of size
    b/s, incremented by s, ranging from zero to b-s
                                                 #Paired with a control
                                                    status of I,N,NC, or
                                                    ND to identify a
                                                    specific state in the
                                                    P-matrix

#Continuously Variable Sampling Interval
hrange <- hmax-hmin
h <- hmin+hrange*(1-states/(b-s))^halpha         #Creates a vector
    of decreasing sampling intervals

#Continuously Variable Sample Size
nrange <- nmax-nmin
```

```
n <- nmin+nrange*(states/(b-s))^nalpha            #Creates a vector
    of increasing sample sizes
n <- round(n)

#The P-Matrix
#In the commentary below, i and j is used in lieu of the complete
    notation, +/-i*s and +/-j*s.

#Row 1 - (0,I) to ?
P[1,1] <- 2*(pnorm(k+s)-.5)*exp(-hmax*lam) #(0,I) to (0,I)

P[1,2:r] <- 2*(pnorm(k+states[2:r]+s)-pnorm(k+states[2:r]))*exp(-
    hmax*lam) #(0,I) to (j,I)

P[1,r+1] <- 2*(1-pnorm(k+b))*exp(-hmax*lam) #(0,I) to False

P[1,r+2] <- (pnorm(k+s,mean=del*sqrt(nmin))-pnorm(-k-s,mean=del*
    sqrt(nmin)))*(1-exp(-hmax*lam)) #(0,I) to (0,N)

P[1,(r+3):(2*r+1)] <- (pnorm(k+states[2:r]+s,mean=del*sqrt(nmin))
    -pnorm(k+states[2:r],mean=del*sqrt(nmin)))*(1-exp(-hmax*lam))
    #(0,I) to (j,NC)

P[1,(2*r+2):(3*r)] <- (pnorm(k+states[2:r]+s,mean=-del*sqrt(nmin)
    )-pnorm(k+states[2:r],mean=-del*sqrt(nmin)))*(1-exp(-hmax*lam)
    ) #(0,I) to (j,ND)
```

```
P[1,(3*r+1)] <- (1-pnorm(k+b,mean=del*sqrt(nmin))+pnorm(-k-b,mean
    =del*sqrt(nmin)))*(1-exp(-hmax*lam)) #(0,I) to TRUE

#Rows 2:r - (i,I) to ?
P[2:r,1] <- (pnorm(pmax(-k,k-states[2:r]))-pnorm(-k-s))*exp(-h[2:
    r]*lam) #(i,I) to (0,I)

for(i in 2:r){
for(j in 2:r){
        if(i>j){
                P[i,j] <- (pnorm(max(-k,k-(i-j)*s))-pnorm(max(-k,
                    k-(i-j+1)*s))+pnorm(-k-(j-1)*s)-pnorm(-k-j*s))
                    *exp(-h[i]*lam)
        }
        if(i<j){
                P[i,j] <- (pnorm(k+(j-i+1)*s)-pnorm(k+(j-i)*s)+
                    pnorm(-k-(j-1)*s)-pnorm(-k-j*s))*exp(-h[i]*lam
                    )
        }
        if(i==j){
                P[i,j] <- (pnorm(k+s)-pnorm(max(-k,k-s))+pnorm(-k
                    -(i-1)*s)-pnorm(-k-i*s))*exp(-h[i]*lam)
        }
}
} #(i,I) to (j,I)

P[2:r,r+1] <- (1-pnorm(k+b-states[2:r])+pnorm(-k-b))*exp(-h[2:r]*
```

```
    lam) #(i,I) to False

P[2:r,r+2] <- (.5*(pnorm(pmax(-k,k-states[2:r]),mean=del*sqrt(n
    [2:r]))-pnorm(-k-s,mean=del*sqrt(n[2:r])))+.5*(pnorm(k+s,mean=
    del*sqrt(n[2:r]))-pnorm(pmin(k,-k+states[2:r]),mean=del*sqrt(n
    [2:r]))))*(1-exp(-h[2:r]*lam)) #(i,I) to (0,N)

for(i in 2:r){
for(j in 2:r){
        if(i>j){
                P[i,r+1+j] <- (.5*(pnorm(max(-k,k-(i-j)*s),mean=
                    del*sqrt(n[i]))-pnorm(max(-k,k-(i-j+1)*s),mean
                    =del*sqrt(n[i])))+.5*(pnorm(-k-(j-1)*s,mean=-
                    del*sqrt(n[i]))-pnorm(-k-j*s,mean=-del*sqrt(n[
                    i]))))*(1-exp(-h[i]*lam))
        }
        if(i<j){
                P[i,r+1+j] <- (.5*(pnorm(k+(j-i+1)*s,mean=del*
                    sqrt(n[i]))-pnorm(k+(j-i)*s,mean=del*sqrt(n[i
                    ])))+.5*(pnorm(-k-(j-1)*s,mean=-del*sqrt(n[i])
                    )-pnorm(-k-j*s,mean=-del*sqrt(n[i]))))*(1-exp
                    (-h[i]*lam))
        }
        if(i==j){
                P[i,r+1+j] <- (.5*(pnorm(k+s,mean=del*sqrt(n[i]))
                    -pnorm(max(-k,k-s),mean=del*sqrt(n[i])))+.5*(
                    pnorm(-k-(i-1)*s,mean=-del*sqrt(n[i]))-pnorm(-
```

```
                    k-i*s,mean=-del*sqrt(n[i]))))*(1-exp(-h[i]*lam
                    ))
            }
    }
    } #(i,I) to (j,NC)

    for(i in 2:r){
    for(j in 2:r){
            if(i>j){
                    P[i,(2*r+j)] <- (.5*(pnorm(max(-k,k-(i-j)*s),mean
                      =-del*sqrt(n[i]))-pnorm(max(-k,k-(i-j+1)*s),
                      mean=-del*sqrt(n[i])))+.5*(pnorm(-k-(j-1)*s,
                      mean=del*sqrt(n[i]))-pnorm(-k-j*s,mean=del*
                      sqrt(n[i]))))*(1-exp(-h[i]*lam))
            }
            if(i<j){
                    P[i,(2*r+j)] <- (.5*(pnorm(k+(j-i+1)*s,mean=-del*
                      sqrt(n[i]))-pnorm(k+(j-i)*s,mean=-del*sqrt(n[i
                      ])))+.5*(pnorm(-k-(j-1)*s,mean=del*sqrt(n[i]))
                      -pnorm(-k-j*s,mean=del*sqrt(n[i]))))*(1-exp(-h
                      [i]*lam))
            }
            if(i==j){
                    P[i,(2*r+j)] <- (.5*(pnorm(k+s,mean=-del*sqrt(n[i
                      ]))-pnorm(max(-k,k-s),mean=-del*sqrt(n[i])))
                      +.5*(pnorm(-k-(i-1)*s,mean=del*sqrt(n[i]))-
                      pnorm(-k-i*s,mean=del*sqrt(n[i]))))*(1-exp(-h[
```

```
                    i]*lam))
          }
}
} #(i,I) to (j,ND)

P[2:r,(3*r+1)] <- (0.5*(1-pnorm(k+b-states[2:r],mean=del*sqrt(n
    [2:r]))+pnorm(-k-b,mean=del*sqrt(n[2:r])))+0.5*(1-pnorm(k+b-
    states[2:r],mean=-del*sqrt(n[2:r]))+pnorm(-k-b,mean=-del*sqrt(
    n[2:r]))))*(1-exp(-h[2:r]*lam)) #(i,I) to True

#Row r+1 - False Signal
P[r+1,1] <- 1 #False to (0,I)

#Row r+2 - (0,N) to ?
P[r+2,r+2] <- pnorm(k+s,mean=del*sqrt(nmin))-pnorm(-k-s,mean=del*
    sqrt(nmin)) #(0,N) to (0,N)

P[r+2,(r+3):(2*r+1)] <- pnorm(k+states[2:r]+s,mean=del*sqrt(nmin)
    )-pnorm(k+states[2:r],mean=del*sqrt(nmin)) #(0,N) to (j,NC)

P[r+2,(2*r+2):(3*r)] <- pnorm(k+states[2:r]+s,mean=-del*sqrt(nmin
    ))-pnorm(k+states[2:r],mean=-del*sqrt(nmin)) #(0,N) to (j,ND)

P[r+2,(3*r+1)] <- 1-pnorm(k+b,mean=del*sqrt(nmin))+pnorm(-k-b,
    mean=del*sqrt(nmin)) #(0,N) to True

#Rows (r+3):(2*r+1) - (i,NC) to ?
```

```
P[(r+3):(2*r+1),r+2] <-  pnorm(pmax(-k,k-states[2:r]),mean=del*
    sqrt(n[2:r]))-pnorm(-k-s,mean=del*sqrt(n[2:r])) #(i,NC) to (0,
    N)

for(i in 2:r){
for(j in 2:r){
        if(i>j){
                P[r+1+i,r+1+j] <- (pnorm(max(-k,k-(i-j)*s),mean=
                    del*sqrt(n[i]))-pnorm(max(-k,k-(i-j+1)*s),mean
                    =del*sqrt(n[i])))
        }
        if(i<j){
                P[r+1+i,r+1+j] <- (pnorm(k+(j-i+1)*s,mean=del*
                    sqrt(n[i]))-pnorm(k+(j-i)*s,mean=del*sqrt(n[i
                    ])))
        }
        if(i==j){
                P[r+1+i,r+1+j] <- (pnorm(k+s,mean=del*sqrt(n[i]))
                    -pnorm(max(-k,k-s),mean=del*sqrt(n[i])))
        }
}
} #(i,NC) to (j,NC)

for(i in 2:r){
for(j in 2:r){
        P[r+1+i,(2*r+j)] <- pnorm(-k-(j-1)*s,mean=del*sqrt(n[i]))
            -pnorm(-k-j*s,mean=del*sqrt(n[i]))
```

```
}
} #(i,NC) to (j,ND)

P[(r+3):(2*r+1),(3*r+1)] <- (1-pnorm(k+b-states[2:r],mean=del*
    sqrt(n[2:r])))+pnorm(-k-b,mean=del*sqrt(n[2:r])) #(i,NC) to
    True

#Rows (2*r+2):(3*r) - (i,ND) to ?
P[(2*r+2):(3*r),r+2] <-  pnorm(pmax(-k,k-states[2:r]),mean=-del*
    sqrt(n[2:r]))-pnorm(-k-s,mean=-del*sqrt(n[2:r])) #(i,ND) to
    (0,N)

for(i in 2:r){
for(j in 2:r){
        P[(2*r+i),r+1+j] <- pnorm(-k-(j-1)*s,mean=-del*sqrt(n[i])
            )-pnorm(-k-j*s,mean=-del*sqrt(n[i]))
}
} #(i,ND) to (j,NC)

for(i in 2:r){
for(j in 2:r){
        if(i>j){
                P[(2*r+i),(2*r+j)] <- (pnorm(max(-k,k-(i-j)*s),
                    mean=-del*sqrt(n[i]))-pnorm(max(-k,k-(i-j+1)*s
                    ),mean=-del*sqrt(n[i])))
        }
        if(i<j){
```

```
            P[(2*r+i),(2*r+j)] <- (pnorm(k+(j-i+1)*s,mean=-
                del*sqrt(n[i]))-pnorm(k+(j-i)*s,mean=-del*sqrt
                (n[i])))
        }
        if(i==j){
            P[(2*r+i),(2*r+j)] <- (pnorm(k+s,mean=-del*sqrt(n
                [i]))-pnorm(max(-k,k-s),mean=-del*sqrt(n[i])))
        }
}
} #(i,ND) to (j,ND)

P[(2*r+2):(3*r),(3*r+1)] <- (1-pnorm(k+b-states[2:r],mean=-del*
    sqrt(n[2:r])))+pnorm(-k-b,mean=-del*sqrt(n[2:r])) #(i,ND) to
    True

#Row 3*r+1 - True Signal
P[(3*r+1),(3*r+1)] <- 1 #True to True

PT <- P[1:m,1:m]          #Matrix of transient states

I <- diag(1,m)            #Identity matrix

#S-Matrix
T <- solve(I-PT)          #Inverse matrix of (I-PT)

numvisit <- T[1,1:m]      #First row of S-Matrix
                          #Expected number of visits to each state
```

```
times <- c(h,ftime,h,h[2:r])        #Time per visit for each state
                                    #h,tfalse represent the in-
                                       control states
                                    #h,h[2:r] represent the out-of-
                                       control states

samples <- c(n,0,n,n[2:r])          #Sample size for each state
                                    #n,0 represent the in-control
                                       states
                                    #n,n[2:r] represent the out-of-
                                       control states

meanvisit <- c(times*numvisit,ttime)      #Long-run expected time
   spent in each state

totalncost <- c(samples*numvisit*ncost) #Long-run expected
   sampling cost for each state

cycletime <- sum(meanvisit)               #Time to complete one
   cycle

totooctime <- sum(meanvisit[(r+2):m])+sum(meanvisit[1:r])-(1/lam)
         #Total expected time spent out-of-control

totooccost <- totooctime*ooccost          #Total expected cost when
    process is out-of-control
```

```
totfcost <- meanvisit[r+1]*fcost            #Expected cost of false
    signal

cyclecost <- sum(totalncost,totooccost,totfcost,tcost)   #Total
    expected cost of one cycle

lrhcost <- cyclecost/cycletime  #Long-run hourly cost

return(lrhcost)
}
```