Software Profiling of Rogue Events in High-Volume Gauging

SOFTWARE PROFILING OF ROGUE EVENTS IN HIGH-VOLUME GAUGING

BY

THOMAS P.K. BERING, M.Eng., (Electrical Engineering)

McMaster University, Hamilton, ON

A THESIS

SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING

AND THE SCHOOL OF GRADUATE STUDIES

OF MCMASTER UNIVERSITY

IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Doctor of Philosophy (2013) McMaster University

(Mechanical Engineering) Hamilton, Ontario, Canada

TITLE:	Software Profiling of Rogue Events in High-Volume Gauging
AUTHOR:	Thomas P.K. Bering B.Eng., (Computer Engineering) McMaster University, Hamilton, ON M.Eng., (Electrical Engineering) McMaster University, Hamilton, ON
SUPERVISOR:	Dr. S.C. Veldhuis
NUMBER OF PAGES:	xxviii, 250

This thesis is dedicated to McMaster University which has schooled me off and on for almost 25 years.

Abstract

Customers are placing ever increasing demands on automotive part manufacturers for high quality parts at low cost. Increasingly, the demand is for zero defects or defect rates in the less than one part per billion. This creates a significant challenge for manufacturers as to how to achieve these low defect levels economically while producing large volumes of parts. Importantly, the presence of infrequent process and measurement (gauge) events can adversely affect product quality.

This thesis uses a statistical mixture model that allows one to assume a main production process that occurs most of the time, and secondary rogue events that occur infrequently. Often the rogue events correspond to necessary operator activity, like equipment repairs and tooling replacement. The mixture model predicts that some gauge observations will be influenced by combinations of these rogue events. Certain production applications, like those involving feedback or high-reliability gauging, are heavily influenced by rogue events and combinations of rogue events. A special runtime software profiler was created to collect information about rogue events, and statistical techniques (rogue event analysis) were used to estimate the waste generated by these rogue events.

The value of these techniques was successfully demonstrated in three different industrial automotive part production applications. Two of these systems involve an automated feedback application with Computer Numerically Controlled (CNC) machining centers and Coordinate Measuring Machine (CMM) gauges. The third application involves a high-reliability inspection system that used optical, camera-based, machine-vision technology. The original system accepted reject parts at a rate of 98.7 part per million (ppm), despite multiple levels of redundancy. The final system showed no outgoing defects on a 1 million part factory data sample, and a 100 million part simulated data sample. It is expected that the final system reliability will meet the 0.001 ppm specification, which represents a huge improvement.

Acknowledgements

My appreciation to Amy for her inspiration and support.

Thanks to the supervisory committee members for their support. Dr. Stephen Veldhuis for his unfaltering optimism, encouragement, and support. Dr. Gary Bone for the most detailed, thorough and meticulous proofreading that I have ever encountered. Dr. John MacGregor for the suggestion of multi-variate analysis using Principal Components Analysis (PCA), which eventually led to the discovery of the unusual mathematical effects involving Q-charting that are discussed in this thesis. Dr. Mark Lawford for the software engineering perspective. Dr. Peter Macdonald for his perspective on the statistics, and his insights in how to present a complex topic in a concise and understandable manner.

Additionally, thanks go to the industrial partners. Particularly, Mr. Gary Burkhalter provided the acoustic data that was used to develop much of the theory. Mr. Mick Janke and Mr. David Ellison gave access to their CNC manufacturing equipment and production cells. Thanks to Mr. Peter Detmers and Mr. David Palmer for inspiring the development of the CNC to CMM closed loop manufacturing software, and providing access to measuring equipment.

Thanks to Mr. Michael Parchimowicz for reviewing this work and recommending a visit to the Department of Mathematics and Statistics at McMaster University. Thanks to James Nixon for his involvement in the Fanuc and Siemens CNC control customization work.

Notation and Nomenclature

Notation

§ identifies a section number in the thesis.

(x.y) identifies equation y from Chapter x.

Nomenclature

Variable	Definition
0	The number 0.
$\mathbf{0}$	The zero matrix. A matrix containing zeros.
α	Probability of a rogue process event. See §4.1.1.
β	Probability of a rogue gauge event. See §4.1.1.
γ	See §4.3.3.
ε	Probability of a rogue event. Often assumed to be small or zero. See §4.2.5, §4.2.8 and §4.3.5.5.
$\mathbf{\Lambda}$	See §4.4.1.2. For $\mathbf{\Lambda}_A$ see §4.4.2.
$\Phi(x)$	The distribution function of the standard normal distribution. See §3.1.6.

Variable	Definition
$\phi(x)$	The density function of the standard normal distribution. See §3.1.7.
ρ	See §4.3.3.
$\mathbf{\Sigma}$	The population variance matrix. Subscripts describe which variance matrix.
σ	The population standard deviation. Subscripts identify which population standard deviation.
σ^2	The population variance. Subscripts identify which variance.
θ	Probability of a rogue event. See §4.2.5.
ϑ	The confidence limit in §4.4.6.2 is $1 - \vartheta$.
ξ	See §4.4.5.
ω_i	The number of parts quarantined for an event that lasts i update samples.
ω_{FN}	The total number of parts quarantined per manufacturing period (day).
A	The action of the controller. A represents Accept for an Accept/Reject decision or Chase for a Chase/Wait decision.
A'	The complement of A. A' represents Reject for an Accept/Reject decision or Wait for a Chase/Wait decision.
$\mathrm{Cov}(X, Y)$	The covariance of X and Y.
C_{PK}	The process capability index. See §3.3.
c_i	The scrap / rework rate corresponding to ω_i.
D	The delay in the feedback loop. May be expressed in update-samples, samples or parts. See §4.2.1.

Variable	Definition
D_{CNC}	Delay between first finish cut to part exiting CNC. For a double spindle lathe, typically $D_{CNC} \approx 2T_{CNC}$
E	An event. Subscripts indicate which event.
$\mathrm{E}(X)$	See §3.1.3
f_A	Rate of events per period (day). See §8.4.
G	See §4.4.3.
$g(\boldsymbol{Y})$, $g_Q(\boldsymbol{Y})$	See §4.4.1.2.
H	See §4.4.3.
I	The identity matrix. A square matrix containing ones on the diagonal and zeros elsewhere.
k_{FS}	The in-spec (good) sample rate to fine sample rate ratio.
k_{US}	Update sample rate to general sample rate integer. See (6.1).
μ	A population mean. Subscripts identify which population mean.
$\mathcal{N}(\mu, \sigma^2)$	is a normal distribution with mean μ and variance σ^2.
N	The number of elements in the *X*, *W* and *Y* vectors. See §4.4
n	The number of elements in the eigenvalue matrix. See §4.4.1.2.
n_{CMM}	The number of Coordinate Measuring Machines (CMM) in the cell.
n_{CNC}	The number of CNC machines in cell.
n_{CS}	The sampling rate for control purposes in parts per sample. In application, this will be n_{GS} or n_{FS}, depending on what the queuing algorithm selects.

Variable	Definition
n_{FS}	Fast (or fine) sample rate in parts.
n_{GS}	The in-spec (good) sample rate in parts. This is the sample rate in normal production.
n_Q	Number of quarantine queues.
N_P	See §4.4.6.2.
N_S	Number of parts in a sample.
N_{SG}	Number of parts in a subgroup. 5 parts per subgroup is a typical subgroup size.
n_U	Update sample rate in parts rounded up to the next n_{CS},n_{FS}, or $n_G S$-th increment as applicable. For example, if the process is capable of updating in 4 or 5 parts, and $n_{CS} = n_{GS} = n_{FS} = 3$ then $n_U = 6$.
$\mathrm{O}(n)$	Big-O notation. See §3.1.10.
P	A probability. For P_A, P_C and P_E see §4.3 and §4.3.3.
$\mathcal{P}(\mu, \sigma^2)$	The process distribution with mean μ and variance σ^2. Subscripts indicate which distribution. See §4.1.1.
P_E^*	Upper bound for P_E. See §4.3.
Q	The Q-chart variable. See §4.4.1.2.
$\mathcal{R}(\mu, \sigma^2)$	The rogue distribution with mean μ and variance σ^2. Subscripts indicate which distribution. See §4.1.1.
$\boldsymbol{T}$	An intermediate vector. See (4.161).
T_{CMM}	CMM cycle time (inspection time).
T_{CNC}	Cycle Time of CNC
T_{CONV}	Conveyor Transport Time

Variable	Definition
T_{WAIT}	Wait time for CMM.
U_{CC}	Required measurements for common cause variation.
U_{CMM}	Capacity of CMM measuring process (in parts).
U_{CNC}	Capacity of CNC machining process (in parts).
U_{SC}	Required measurements for special cause variation, including measurements already scheduled for common cause variation.
$U_{SCEXTRA}$	Required measurements for special cause variation, excluding measurements already scheduled for common cause variation.
$\mathbf{V}$	See §4.4.1.2. For $\mathbf{V}_A$ see §4.4.2.
$\mathrm{Var}(X)$	The variance of X. See §3.1.3.
W	The random variable representing the gauge error. See §4.1.1.
$\boldsymbol{W}$	The random vector representing the gauge error. See §4.4.
X	The random variable representing the process error. See §4.1.1.
$\boldsymbol{X}$	The random vector representing the process error. See §4.4.
Y	The random variable representing the observation, *i.e.* the observed error. See §4.1.1.
$\boldsymbol{Y}$	The random vector representing the observation, *i.e.* the observed error. See §4.4.

Definitions and Abbreviations

Bound - As used in this thesis, it means to establish an upper or lower bound on a variable. Optionally, the bound can be reduced until the economic effects of the variable are no longer significant.

CMM - *Coordinate Measuring Machine* A machine that automatically measures parts under computer control.

CNC - *Computer Numerically Controlled* A designation for an automatic machining center that indicates it executes programs under computer control, as opposed to under manual control, or under the control of a set of prerecorded commands on paper tape (NC).

Common Cause Variation - In SPC, the variation that is naturally present in the production process is common cause variation. Also, see special cause variation.

C_{PK} - *Process capability index*. Properly, C_{PK}. See §3.3.

distribution function - *Cumulative Distribution Function* - See §3.1.6.

density function - *Probability Density Function* - See §3.1.7.

GR&R - *Gauge Repeatability and Reproducibility* - A method for analyzing how accurately a gauge's measurements of the same parts repeat on the same gauge (repeatability) and other gauges (reproducibility). The frequently used test specified in Measurement Systems Analysis by AIAG (2003) specifies 10 parts tested by 3 different operators with 3 observations per operator.

IID - *independent and identically distributed* - All the distributions under consideration are the same, and their values are statistically independent from each other (see independent).

independent - See §3.1.1 and §3.1.8.

in-spec - The part either meets or is within the customer specifications, within the expected accuracy of the gauge as defined by the GR&R.

M-SPC - *Multivariate Statistical Process Control.* In M-SPC, multiple random variables are monitored on a single set of control charts. For systems involving large numbers of variables, this significantly reduces the total number of control charts.

Nominal - The center value from the part print. Tolerance limits are often defined relative to the nominal. For example in the dimension 1.25 ± 0.25 mm. 1.25 mm is the nominal.

on-line - An operation that takes place inside the automated production process, *i.e.* on the line.

off-line - An operation that takes place outside of the automated production process, *i.e.* off the line.

Outgoing Defects - are parts that are accepted by the gauge, even though the end-user and/or subsequent inspection will likely reject the part as defective.

out-of-spec - The part does not meet the customer specifications. See in-spec.

Part Print - The engineering specification document that specifies the part dimensions.

PPAP - *Production Part Approval Process* - All production parts must be checked for suitability in the customer's application for both manufacturing and product design reasons. The intention of the PPAP process is to verify a supplier's production methods, ensuring the production process is capable of meeting the customer's needs. Undocumented changes on the part of a supplier have occasionally caused unforeseen adverse impacts on the customer's manufacturing operations or final product. The point of the PPAP process is to approve a known manufacturing process in such a manner that all the parts produced by the known process are acceptable to the customer. If the supplier continues to follow the approved process, then all parts from the approved process will continue to remain acceptable to the customer, subject to the expected risks listed on the relevant FMEA and control plan documents.

ppb - *parts per billion* - A short form for 10^{-9}.

ppm - *parts per million* - A short form for 10^{-6}.

Random Variable - See §3.1.2.

Random Vector - See §3.1.4.

Repeatability - measures the influence of random effects on gauge observations, given a gauge measuring the same part with the same operator (measurement conditions). Repeatability is usually specified as a tolerance zone, or a percentage of the part tolerance zone. It is determined with a GR&R.

Reproducibility - measures the influence of different operators (measurement conditions) on the observations of a gauge. Typically, manual gauges are heavily influenced by the human operator that uses them. Alternatively, the GR&R procedure can also be used to test reproducibility between two automatic gauges, or between an automatic and a manual gauge, or under different measurement conditions. Reproducibility is usually specified as a tolerance zone, or a percentage of the part tolerance zone. It is determined with a GR&R.

Rogue Event - An event outside of the main process and/or measurement operation.

Rogue Distribution - The distribution associated with a particular class of rogue events.

SPC - *Statistical Process Control* A method used to optimize industrial processes where out-of-control conditions are detected on control charts, investigations are made, and the processes are adjusted accordingly.

Special Cause Variation - In SPC, the variation to be eliminated is special cause variation. Also, see common-cause variation. Rogue variation can be either special or common cause variation, depending on the process designer's current goals.

Wear Offset - the CNC machines use cutting tools that experience wear. The wear offsets are used to compensate for this wear.

Declaration of Academic Achievement

Thesis

This thesis represents the cumulation of a great deal of work on a variety of industrial applications, and the theoretical frameworks that explain the problems encountered. In terms of the theoretical framework, Dr. MacGregor suggested using Principle Components Analysis (PCA). I later adapted this into the vision of using multivariate analysis to detect stable distributions of good production by monitoring computer programs via data logging and runtime profiling through Q-charting.

Early versions of this thesis used an additive model, which I created, to explain the probability distributions encountered. Dr. Macdonald suggested the basic mixture model, which simplified the mathematics significantly from the earlier additive model. With guidance from Dr. Macdonald, I developed the remaining theory and simulations presented in Chapter 4 based on this model.

I wrote this thesis. The committee members, Dr. Bone, Dr. Lawford, Dr. Macdonald, and Dr. Veldhuis, helped in the review and editing of this thesis. They provided a great deal of advice on sections and material to improve, include, expand upon, simplify and eliminate.

Some of the material in this thesis has appeared previously in presentations, posters and papers that I also created. Certain key graphs, notably Figures 5.1 through 5.4, were first shown in presentations and posters beginning in August 2005. Also, Figures 9.1 through 9.4 or variations thereof were first shown in a similar time frame. Elements of Chapters 5 through 9 were published in Bering and Veldhuis (2010a), Bering and Veldhuis (2010b), and Bering and Veldhuis (2010c). I wrote these papers; Dr. Veldhuis supervised and edited them.

Industrial Applications

This thesis was inspired from real industrial applications. These applications were constructed and implemented at third party sites using third party contractors. Dr. Veldhuis was invaluable in providing the original motivation and ongoing encouragement and support for these projects. The research, software, and engineering components of the applications were written and developed by me.

In terms of the CNC feedback application, I wrote the feedback software. This is a very large set of programs, and consists of many individual programs and libraries, not all of which are used in any single cell. I wrote 238,297 lines of code in the primary code base. Using a 3,902 line configuration file that I wrote, and a code generators that I wrote, another 551,339 lines of code were automatically generated. More than 26,798 lines of code come from third-party external libraries. In total, I wrote or automatically generated 793,538 lines of code, not including the third-party libraries.

An industrial cell consists of many programmable devices, in addition to the programs running on the feedback computer listed above. Modifications to the cell PLC (Programmable Logic Controller) software related to feedback were overseen by me, and I wrote the specification and some of the PLC code for feedback communication. The bulk of the PLC programming was done by third party contractors. James Nixon and I wrote the CNC (Computer Numeric Control) code related to feedback. The remainder of the CNC code was provided by the relevant automotive parts supplier. The CMM (Coordinate Measuring Machine) code was written by the vendor, with the exception of the code relating to feedback which was written either by me or in discussion with me.

I wrote the run-time profiler and the data visualization tools to visualize the results. These utilities include 34,736 lines of code, not including third party libraries or libraries included in the code totals previously mentioned.

In terms of the spring inspections system described in Chapter 9, many systems were purchased from third parties. One of these systems underwent significant modification and improvement by me. The modified system captured the spring images that provided the raw data for the analysis presented in Chapter 9.

Given the captured spring images, it was possible to implement new and improved versions of the same algorithms that were supplied by the original third-party developers. The first two algorithms in Chapter 9 were new versions of the existing algorithms, entirely written and developed by me. The third algorithm, the pattern matching algorithm, was entirely created by me and was fundamentally different from the previous solutions. The resulting effort consisted of represents 21,618 lines of code, written by me, not including third-party libraries or code listed in the previous totals. As all of the original solutions were reimplemented, everything shown in Chapter 9 comes from simulations and software written by me.

The example in §4.4.4, is a simplification of a model that I originally developed for an industrial acoustic measurement application. It is one of several additional applications that were used in the test and development of the technologies presented in this thesis. Many of these applications are beyond the scope of this thesis. The work presented here was selected because of both academic importance and industrial interest.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Overview

Canadian automotive parts suppliers spent $8.9 billion dollars on capital equipment in the decade from 2000 to 2009 (DesRosiers, 2011). Increasingly, these capital projects feature ever more complex mechanical, electronic and software systems, with software being used in new and innovative ways. Mechanical systems can fail in many different ways, creating difficult to predict interactions between the mechanical and software systems. In software projects, validating and debugging software represents 50-75% of the total software cost (Hailpern and Santhanam, 2002). For the projects of interest in this thesis, the performance of the gauging and control software has a large impact on the economic performance of the production line, and hence the return on capital investment. Immense pressure exists to ensure that new capital investments are brought into production smoothly, with minimal software or part quality issues. In particular, unanticipated quality issues can result in expensive product recalls, which can make it uneconomical to invest in new manufacturing equipment.

In many cases, the part volumes associated with each individual part production process have increased. The result is that even unlikely (one in a million) failure modes occur frequently in the context of producing hundreds of millions or billions of parts. It is impossible to understand all possible failure mechanisms, and their interactions, in advance. At low defect rates, complex interactions within the manufacturing process chain can have unexpected adverse results of considerable economic significance.

Current approaches to this are defined by initiatives such as Six Sigma (Tennant, 2001), quality standards (ISO, 2003), and requirements such as 1 ppb (part per billion) or zero defect specifications. Significant progress has been made in controlling effects that can be made or predicted around the normal distribution. These include both Statistical Process Control (SPC) approaches and deploying automated process control (APC) approaches. Many systems benefit from the use of SPC, APC, and Six Sigma techniques, and many of these techniques have been extended to work with many different distributions. However, the fundamental problem with any technique that assumes a known distribution to make predictions about gauge performance is that if the actual gauge observations do not follow the assumed distribution, then the predictions can be wrong.

One of the remaining problems which is addressed in this thesis involves identifying defects that are infrequent and thus hard to detect. For example, the production volumes involved in the automotive industry are very large, and each car contains many parts (on the order of 10,000). To increase the number of defect-free cars, assembly plants are requiring part manufacturers to have very small defect rates. Achieving these low rates requires the highest performance levels from the manufacturing processes, gauges and control systems deployed on the production line.

Historically, automotive parts were manufactured with dedicated, single function pieces of automation, and sophisticated electronic control systems

did not exist. Since the advent of inexpensive computers, modern manufacturing features highly sophisticated pieces of programmable electronic equipment. This includes Computer Numeric Control (CNC) systems, and Programmable Logic Control (PLC) systems. These systems can implement sophisticated control algorithms. Simply deploying an automatic control system does not always result in an inexpensive, low defect rate, high performance industrial process. Instead, it is necessary to deploy a control system with appropriately selected algorithms onto appropriately designed production and measurement (gauging) systems. While all production processes, algorithms, and gauges are different, the automotive parts supplier industry designs the systems to meet a common set of quality standards included in ISO TS16949 (ISO, 2003). Thus, a wide variety of algorithms and systems are treated in a common way. For the industrial applications described in this thesis, the common methodologies described in existing literature were ineffective in reducing costs and reducing defect rates. As a result, this thesis presents new methods, which are applicable to many algorithms and systems used in high-volume manufacturing.

This thesis describes two new techniques that were developed to assist engineers in their efforts to make economically sound improvements in production processes. The techniques are

Rogue Event Analysis - Chapter 4 presents the statistical theory used for rogue event analysis. The assumption is that manufacturing processes have multiple modes of operation, which are modeled as mixture distributions. A main operating mode exists in which the majority of parts are produced. With modern SPC techniques, this mode is often highly capable (accurate and repeatable), and has a correspondingly high process capability index C_{PK}. Unfortunately, malfunctions in the production line generate rogue events. These rogue events may or may not require automatic or human operator intervention to correct. In some

systems, at infrequent but measurable rates, the automated systems can be confused by interactions of multiple rogue events, and ship reject product to a customer. This results in complaints and outgoing defects. Rogue event analysis is about understanding the impact of these rogue events on production processes.

Runtime Profiling with SPC - Infrequent events are most easily detected with ample data from real production systems, however this is rarely available during system development and initial deployment. The runtime profiler is aimed at getting meaningful data from systems in full production. Importantly, the runtime profiler developed in this thesis was designed to monitor program behaviour and compare it against control limits developed using SPC techniques. For high-reliability applications, it is designed to quarantine any parts that are not within the application-specific control limits. Information gathered on-line is logged to disk and can be reviewed off-line. Program changes and system improvements can then be made based on well considered off-line analysis.

If the above two techniques are used together, they explain the effects observed in practical automated inspection (gauging) systems, and manufacturing cells.

An important aspect of this thesis was to develop models that explained the statistical effects observed in the cell. Probability calculations can be performed when the distributions of the underlying random variables are known. In principle, any distribution can be assumed, however in practice the normal distribution was often assumed. However, modern run-time profiling has the capability of gathering large quantities of information from the programs being monitored. This includes program flow behaviour, and the values of any process related input and output values gathered during

the execution of the program. From the production line in Chapter 9, million part data samples can be gathered within two weeks. With these large data samples, it is no longer necessary to assume the normal distribution, as the underlying distributions can be directly measured to accuracy levels proportionate to the sample size.

In order to better explain the events observed, in Chapter 4, a mixture model is proposed in this thesis to be used instead of the more conventional normal distribution model. Some applications require predictions about infrequent events to accuracy levels beyond what can be achieved with realistic sample sizes, even if every part produced is monitored by the run-time profiler. These include applications with zero defect / 1 ppb outgoing defect specifications.

This thesis proposes statistical models that can be used to estimate both frequent and infrequent events. With accurate models, it is possible to select and optimize algorithms to improve the economic returns on production processes. Specifically, for any given set of inputs, decision theory can be used to make the best decision out of a finite number of alternatives. For continuous systems, an optimal output can be found. However, this process only works if the underlying statistical model is accurate.

Without an accurate model, it is difficult to make accurate predictions about the performance of industrial systems. Thus, this thesis is about developing a mixture model, and using it to make predictions and explain the observed effects encountered in manufacturing cells. Use of these models permitted the measurement (gauging), material transport and feedback control processes to be improved. This resulted in improved economic returns and reduced waste in the example applications.

1.2 Premise of Operation

Modern automotive part production processes are designed to be repeatable

so that parts can be produced with consistent characteristics (dimensions). Practical processes cannot be made to repeat perfectly. In manufacturing, any losses associated with the inability of the production process to repeat perfectly, are described as “waste”. Waste may include quarantined parts, rework, scrap parts and downtime.

Once the process becomes sufficiently capable, then the process should be sufficiently accurate and repeatable that the waste generated should be minimal. However, many applications generate more waste than the process capability measurements would predict. In industrial applications, infrequent events can affect the production process and cause additional waste. In this thesis, the infrequent events that fundamentally alter the nature of the production process will be called rogue events. *The challenge addressed in this thesis is to find and reduce the amount of waste generated by these rogue events.*

Often, the mechanical, electrical, chemical, thermal, and/or optical systems involved in the production process are assumed to be predictable. However, if these systems are measured with sufficiently accurate measuring devices, variability in the measurements can be detected. If these random effects correspond to variation in the parts being manufactured, they are described as “Process Error”. If the random effects correspond to errors in the measurement (gauge) process, they are described as “Gauge Error”. When a gauge takes a measurement, the resulting “observation” is the sum of the process error and the gauge error.

In most modern production processes, computer software reads the input signals from the gauge; the results are displayed; and decisions are made on how to control the production process. The effect of rogue events on process and gauge error can be unpredictable. Thus the decision making process, whether implemented automatically under computer software control, or manually under operator control, can never be made completely perfect. Some waste in the production process will always be generated.

Many computerized gauges are capable of gathering large numbers of inputs (observations). The interpretation of these observations, when the physical system is considered, is often difficult. This necessitated the development of the mixture model to explain the effects observed.

Modern manufacturing systems often run 24 hours per day and 7 days per week (24/7). Additionally, computers gather information very quickly, and can make decisions far too fast for people to review and validate. This creates significant challenges when trying to determine the root causes of waste, and how to minimize it. One solution is to record large volumes of data automatically from the production process, and then analyze the data off-line.

In computers, tools that automatically gather production data are referred to as “profilers”, and tools that automatically gather production data on deployed (running) systems are referred to as “runtime profilers”. Historically, it was believed that the computer costs associated with profiling were too expensive to be used on production systems, as such runtime profiling was never used. Older profiling approaches primarily gathered timing data from the programs. However, the newer approach used in this thesis annotated the source code to collect almost all critical program flow and process-related input and output data in the program. With the price of modern computers, computer costs are no longer an issue for many production processes. It is now possible to capture large volumes of information in real-time without incurring significant costs.

The automatic profiling / data logging approach is very effective at gathering information on rogue events. However, the data log can contain so much information that no one has the time to review it. As such, easily detected rogue events may be missed amongst all of the information. Interactions amongst infrequent events can be extremely hard to find inside logging data, as the predictive capacity of any logging set is limited by the size

of the sample analyzed. For high-reliability applications, a person may not be able to analyze enough observations to make accurate predictions about infrequent events. Thus, the automatically gathered profiling data must be analyzed automatically with suitable statistical techniques. This motivated the development of the mixture models presented in Chapter 4.

1.3 Statistical Methods

Statistical methods assume the presence of random variables. The behaviour of these random variables is often described by the mean, variance, and distribution of the variable. A typical distribution is the normal (Gaussian) distribution.

The conventional approach is to estimate the mean and variance based on small samples of data, and extrapolate the nature of the distribution by assuming the normal distribution applies. Often, the sampling is done with outlier rejection, which helps to minimize the influence of infrequent outliers on the estimates of the mean and variance. If the majority of the waste in the process is dominated by processes that can be predicted through this extrapolation process, then the normal distribution model works well.

The issue encountered in the industrial applications was that the infrequent rogue events were causing the majority of the waste in the production process, and the conventional approach did not predict these infrequent events accurately. As such, this thesis uses a mixture model that assumes a main production process when no rogue events are present, and secondary rogue processes when the rogue events are present.

The presence of rogue events can significantly affect the accuracy and performance of the software measuring parts and controlling feedback cell operations. For instance, if one type of rogue event happens one percent of

the time, and one rogue event results in ten parts being quarantined, then that single type of rogue event results in ten percent of the total production being quarantined. Manufacturers, partially understanding this effect, will only operate cell designs where the impact of rogue events is limited.

Sometimes, two different types of rogue events can occur simultaneously. This is referred to as a combined rogue event. The odds of this occurring can be quite small, often less than a few hundred parts per million. Occasionally, it can be difficult for algorithms to distinguish between combined rogue events and main production, because both may exist in the pass-zone of the gauge. The odds of a combined rogue event occurring inside the pass-zone of the gauge is quite small, however it can be a measurable and costly effect when the customers specification is 1 ppb or zero defects.

In this thesis, an obviously reject part being accepted as good by an automated gauge is referred to as an "outgoing defect". Outgoing defects cause significant issues in automated high-reliability gauging applications. In these applications, if an end-user finds obviously reject (defective) parts inside a part shipment (or their car), then expensive production, safety and/or product recall and quarantine procedures will be implemented such that the quality problem is contained and people are not placed in danger unnecessarily. *A key achievement in this thesis is that it makes it possible to predict the rate of combined rogue events and estimate the rate of outgoing defects.*

Within reasonable economic limits, manufacturers want profitable manufacturing cells that produce no more waste than necessary. This will only occur if the manufacturing cell deals with rogue events effectively, and not by shipping obviously defective production to the customer (end-user). If an end-user finds an outgoing defect inside shipped production, then the resulting product recalls will threaten the long-term profitability of the manufacturing operation.

1.4 Application of Statistical Methods

As it is now possible to estimate of the rate of outgoing defects resulting from a given choice of software algorithms, appropriate inspection algorithms can be selected for the industrial applications described in this thesis. In a high-reliability application, selecting an appropriate inspection algorithm can dramatically reduce the number of outgoing defects and the costs associated with them. The theory for this is developed in Chapter 4, and the industrial application shown in Chapter 9.

The usefulness of the rogue events approach is not restricted to high-reliability inspection systems. In a feedback application, it is vital to be able to distinguish between process error and gauge error. Importantly, all instances of rogue process error need to be responded to quickly, as much waste can result.

The insight gained in the feedback application was that it was necessary to select appropriate algorithms at two different points in the cell design. As shown in Chapter 4, delays in the feedback loop can create misleading feedback information. As such, any rogue events that create delay in the feedback loop will adversely impact feedback loop performance. Thus, the feedback algorithm needs to be able to respond quickly to rogue process error, in order to keep the process operating within specifications. As such, if the rates of the two competing sources of rogue events cannot be kept within acceptable limits, excessive waste will result. This effect was demonstrated both theoretically, and in the industrial applications that are described next.

1.4.1 Industrial Importance

The first two industrial applications relate to CNC machining cells with Coordinate Measuring Machine (CMM) Feedback. These were large complex

cells, which were attempted at four different plants, with the promise of significantly reducing waste and increasing worker productivity. The first industrial application examines the feedback loop, and details how rogue events can be the dominant source of special cause variation and waste inside the cell.

The second industrial application examines the material transfer system inside the same class of cells, and details how the material transfer system must be adapted to deal with rogue events such that the resulting waste is reduced. Four companies attempted the creation of these cells, and only two succeeded. Those that succeeded employed the techniques developed and discussed in this thesis.

The third industrial application was an attribute inspection system. The challenge was that certain gauging algorithms were highly capable, and could easily pass the standard methods of estimating their reliability as detailed in Measurement Systems Analysis (AIAG, 2003). However, when rogue events were present, these methods generated misleading results. In practice, this meant that it was very hard to predict which algorithms were effective on-site, and the customer was unable to determine if gauge malfunctions were causing excessive outgoing defects. This problem was solved through the use of the new runtime profiler described in Chapter 5, and by designing a gauge algorithm that made it very unlikely that multiple rogue events could cancel in an undetectable or difficult to detect manner.

The industrial applications show how important it is to use collected information to improve processes in a manner that reduces waste.

1.5 Thesis Layout

This thesis describes technology that continues to be used in real world

industrial applications. As such, the thesis layout was chosen such that the theory is first developed, and the later chapters describe the industrial applications themselves.

Chapter 2 presents a detailed literature review.

Chapter 3 presents a short background on the key theoretical concepts used in this thesis.

Chapter 4 presents the statistical theory used to describe rogue event analysis.

Chapter 5 describes the runtime profiler that was used in the development of this thesis.

Chapter 6 provides an introduction to the industrial applications. It outlines pertinent background information required to understand the industrial applications presented in Chapters 7 through 9.

Chapter 7 describes the first industrial application, a CNC manufacturing cell with CMM feedback. The goal of Chapter 7 is to present the theory and experimental results relating to the effects of rogue events on the feedback loop, and the selection of feedback algorithms that reduce these negative effects.

Chapter 8 describes the second industrial application, and focuses on the conveyor system using in the manufacturing cell. The feedback algorithms from Chapter 8 determine the frequency at which adverse rogue events occur, and the conveyor system algorithms determine the severity of each event in terms of waste generated per event. As such, significant gains can be made by improving the material transport systems and the algorithms that control them.

Chapter 9 describes the third industrial application, and focuses on a high-reliability inspection system. This industrial application was the best demonstration of the profiling technologies developed, because the frequency of the adverse events was sufficiently small that automated technologies were required.

Chapter 10 presents the conclusions.

Chapter 2

Literature Survey

2.1 Why aim for zero defects?

Companies did not always aim for zero defects. Historically, companies aimed for minimum economic losses. This has been termed the economic conformance level (ECL) model, which is detailed in Daniel and Reitsperger (1991, p. 603), and Juran (1974, pp. 5-12). The ECL calculation assumes that the defect rate is constant (does not change with time), and thus it underestimates the impact of future defect reduction initiatives.

Many modern quality improvement books recommend zero defects as a long-term goal, as product quality can be made to improve with time. This includes the landmark books by Shewhart (1931), Deming (1986), Juran (1974), Ishikawa (1976), and Crosby (1979). These books have resulted in SPC, TQM method, lean manufacturing, and later Six Sigma. In particular, as explained in Design for Six Sigma by Yang and EI-Haik (2008, pp. 12-14), Juran consolidated the works from Deming and Ishikawa into the Total Quality Management (TQM) method. These methods have been applied with success in the American automotive industry (Lightstone *et al.*, 2005), the Japanese automotive industry (Daniel and Reitsperger, 1991), French

automotive industry (Soderquist and Motwani, 1999), and Korean automotive industry (Park *et al.*, 2001). Industrial standards like ISO 9000:2000, QS 9000, and ISO/TS 16949 now encapsulate TQM principles, as discussed in Kartha (2004).

A common benchmark is the Process Capability Index C_{PK}. The relationship between C_{PK} and the expected defect rate is shown in Table 2.1. This table assumes normally distributed process error. Two important effects are worth noting. Firstly, a part dimension that achieves a $C_{PK} = 2$ should have a defect rate of no more than 2 parts per billion (ppb). Secondly, if the effect of the defect involves an interference fit between two mating surfaces, then two $C_{PK} = 1.33$ processes will fail to assemble (interfere) with a probability rate of less than 8 ppb. As an example of an interference / assembly problem, consider the case of a bolt being inserted into a washer. If a bolt and a washer are drawn from a random sample of production parts, then the interference will only occur if the bolt is too large to fit inside the washer. Many practical assembly problems match are correspond to this basic problem.

Table 2.1: Relationship between C_{PK} and expected defect rates

C_{PK}	Sigma	Pr(In-Spec)	Pr(Out-of-spec)	Pr(Interference)
2.0	6σ	99.999 9998%	2.0 ppb	$< 1{\cdot}10^{-8}$ ppb
1.66	5σ	99.999 943%	573.3 ppb	0.0008 ppb
1.5	4.5σ	99.999 32%	6.8 ppm	0.098 ppb
1.33	4σ	99.993 7%	63.3 ppm	7.7 ppb
1.0	3σ	99.73%	0.27%	11.0 ppm

Notes:

1. Interference calculation assumes two mating parts with equal tolerance bands and standard deviations.
2. Parts per billion = ppb.
3. Parts per million = ppm.

The domestic automotive industry has achieved 108 quality defects per 100 vehicles sold, according to the initial quality survey by Power and Associates (2010). If it is assumed that the average car has on the order of 10,000 parts/vehicle sold, with 100 dimensions per part, this means the average rate of outgoing defects per dimension per part is on the order of 1.08

ppm. What the automotive industry would prefer, is an average rate of outgoing defects per dimension per part of 1 ppb, which would translate into 0.1 quality defects per 100 vehicles sold, or almost "zero defects." For the average automotive parts supplier, with production runs less than 100 million parts, a 1 ppb specification translates into "zero defects" as well. The above analysis is approximate. However, the approximations do not change the reality that it is economically desirable to reduce the outgoing defect rates at the supplier level from rates measured in parts per million to rates on the order of one part per billion or less. *The challenge facing the automotive industry is to make the zero defects / 2 ppb promise of* $C_{PK} = 2$ *production a reality.*

2.1.1 Costs of Defects

Gitlow (1990, p. 13) presents two different views of losses due to departures from nominal. In the historical view, any part made that was measured above the upper specification limit, or below the lower specification limit, created the economic loss. This model does not fairly represent the true costs of defects. Instead, Gitlow (1990, p. 14) points out that the economic costs of quality are related to the deviation from nominal. Significant quality issues are likely to incur significantly higher costs than marginal quality issues.

In all three of the example applications, the cell operators were able to notice the failures of the initially deployed software algorithms. Additionally, the JD Power IQS initial quality survey (Power and Associates, 2010) only detects quality problems noticeable to the customer (end-user). As such, it is reasonable to conclude that many of the quality complaints relate to nonborderline outgoing defects. Unfortunately, with modern demands on productivity, it is no longer cost-effective to deploy as many operators and support personnel on production lines. As such, it is necessary to use automation, which may or may not detect quality issues in the same way as people.

A second important concept relating to the cost of defects is encapsulated in the book title "Quality is Free" from Crosby (1979). Continuous improvement does not mean just minimizing the defects as seen by the customer. It means both the supplier and customer working together to reduce waste everywhere in the system. The practical implication of this is that it is not sufficient for a supplier to adopt a high-cost method of production (like scrapping many parts), in an effort to minimize overall production costs. The goal is to eliminate waste throughout the entire supply chain.

2.1.2 Statistical Process Control (SPC)

SPC was originally proposed in Shewhart (1931). Over time it grew to include many concepts, as more encompassing quality improvement efforts like TQM and Six Sigma were developed. For the purposes of this thesis, the SPC philosophy includes

1. Control charting the process being monitored,
2. Detecting potential out-of-control conditions (alarms),
3. Investigating the alarms, and
4. Improving the system to remove the root cause by engaging in continuous improvement.

All of the above SPC steps can be implemented manually, with no computer assistance, by trained factory floor personnel. The advantage to implementing SPC in a manual production environment is that trained factory floor personnel are often needed to accomplish Steps 3 and 4.

Modern production is often highly automated. Steps 1 and 2 of the SPC philosophy lend themselves readily to automated systems. With modern computers and gauges, it is easy to implement automatic data gathering, charting, detection of out-of-control conditions, and alarm generation. It is also possible to quarantine (for later investigation) any suspect production. However, Shewhart developed the original SPC charting methodology before the advent of modern computers in a production line setting. As such, operators had a reasonable chance of implementing steps 3 and 4. With modern automated equipment, and the continuous reductions in shop floor personnel, the challenge is to accomplish steps 3 and 4 in an automated environment.

Control charting remains an area of active research interest. Modern applications of control charts, including the X Bar R chart, are covered in many texts including Ryan (2000) and Vardeman and Jobe (1999). Kourti (2005) points out the significance of multi-variate methods (M-SPC). New control charts continue to be developed, including the concept of the generalized zero-inflated Poisson chart, as discussed in Chen (2004) and Chen *et al.* (2008). Control charts have been used to monitor systems with rare defects, and Xie *et al.* (1995) describes a control chart with that specific goal. Additionally, control charts have been applied to many diverse areas, including software as described by Florac *et al.* (2000).

The repeatability and reproducibility of gauges are monitored through the GR&R procedure specified in the Measurement Systems Analysis (AIAG, 2003). Gauge outputs have been control charted before because, for most industrial processes, the data being control charted are from gauges. Normally, this control charting is used to monitor the production process; however, control charts can also be used to monitor the performance of the gauge, as will be discussed later in this thesis. The idea of control charting gauge data is mentioned in Wheeler (2006) and control charts are recommended for situations where the gauge performance is either borderline or

incapable. Wheeler (2006) does not specify exactly how the control charting is to be done. From the nature of the Wheeler's paper, it would likely be done by measuring the same part 3 or more times, potentially on a regular and on-going basis, and then X Bar R charting the results. Any alarms in the range chart would indicate a problem with the gauge.

To the best of the author's abilities to ascertain, no theoretical foundation for control charting gauge inputs and intermediate gauge data has been published in the literature. In a complex modern gauge, vast amounts of input and intermediate data exists. However, in the author's experience, many people simply assume it is technically impossible to collect significant amounts of data from a modern complex high-speed production gauge. Chapter 5, describes the new run-time profiler that can collect the required data in real-time. Given the new run-time profiler, Chapter 4 presents the relevant theory as to why the input data should be monitored.

The problems solved in this thesis are not the only aspects of SPC and TQM which have not been thoroughly theoretically examined. Sitkin *et al.* (1994) notes

> "the potential contributions of TQM could be lost if its theoretical underpinnings and boundary conditions are not critically assessed. ... The overall patterns that emerge from our analysis suggest that TQM is not a panacea that can be unthinkingly used, but that it must be implemented with a clear sense of the degree to which the context is characterized by uncertainty, nonroutineness, and/or instability."

TQM was developed as a series of managerial philosophies, and SPC Control Charting was developed as a series of techniques. Unfortunately, it is very difficult to develop theoretical models that bound the applicability of these techniques. Furthermore, the work of Sitkin *et al.* (1994) did not resolve all

of the issues surrounding the applicability of TQM. This thesis will address an industrially significant shortcoming of TQM and SPC. It will develop a model of how events associated with frequently detected alarms on control charts can interact to produce less frequent rogue events that do not trigger an alarm.

2.1.3 Solving the Quality Challenges

The automotive parts supply base needs to improve its outgoing part quality while reducing part costs. Even though SPC was originally developed as a manual technique, implementing SPC has yielded many benefits in modern production processes. Some parts of SPC are readily automated, but it is difficult to automate investigations and continuous improvement activities. This thesis details a new type of runtime profiler for use in these investigations, so that they can be performed off-line. As such, the investigations do not require staff to monitor and diagnose software performance at full line-speed. Additionally, the theoretical models will be used to interpret the output, and shed light on why continuous improvement is required.

The focus of this thesis will be on nonborderline effects. A significant push has taken place inside the automotive industry to improve C_{PK} and GR&R scores. This has caused many implementation specific challenges. With work, these challenges are often overcome. Many industrial applications have achieved the required C_{PK} and GR&R scores. Yet these same applications fail to achieve the expected "Quality is Free" (Crosby, 1979) supplier and customer cost reductions. In the industrial applications described in this thesis, it was necessary to implement the new methods described in Chapters 7 to 9 to get the desired quality results.

Finally, this thesis will look at automated tools, notably the new runtime profiler, that are intended to make it simpler to perform continuous

improvement on automated production systems. Profiling is a significant area of software engineering, and it will be discussed in the next section.

2.2 Profiling

Software engineers use profilers to collect diagnostic information from executing programs. Usually, profilers are only used in test builds for various software development activities including software optimization. Runtime profilers are the specific class of profilers used to collect diagnostic information on released software.

Profilers can be used to observe running programs and collect data, which can then be used to detect anomalies. Frankl and Weiss (1993), Hutchins *et al.* (1994), Frankl and Weiss (1991), Frankl and Weyuker (1993b), and Frankl and Weyuker (1993a) examine various software testing methods and techniques. Dickinson *et al.* (2001) lists methods of classifying different techniques, so profilers can be compared. Many methods are available, and are discussed in the literature. In this research, the Branch Count Sequence (BCS) technique from Harrold *et al.* (1998) was selected for use with the runtime profiler because it could be implemented in a deterministic manner with fixed memory requirements. Specifically, it was desired to have fixed memory and execution time (for the profiler), even if lengthy loops were encountered in the inspection algorithms.

Another useful feature of profilers is the ability to detect changes in program execution that lead to the accept gauge condition, without following the typical accept execution path. This may point to either software or hardware failure. Pavlopoulou and Young (1999) and Elbaum and Hardojo (2004) suggest that embedding a known good profile into a program can be used to detect anomalies in running programs. This capability was implemented.

The number of code executions of each code block was computed on a per-part basis. If the number of executions was outside of control limits, then the run-time profiler would automatically ensure the resulting suspect parts were quarantined. Additionally, the run-time profiler also monitored the Q-score function as described in Chapter 4, and would quarantine parts if the Q-score was outside of control limits.

Of the existing profiling approaches in literature, the BCS technique from Harrold *et al.* (1998) was used in this thesis, as it was considered to be more effective than the technique from Liblit *et al.* (2005). As such, the BCS technique was implemented in this thesis.

Runtime profiling is usually challenged by execution speed. The closest similar work found in the literature survey is from Liblit *et al.* (2003) and Liblit *et al.* (2005). Liblit uses the predicates equal ($=$), less than ($<$), less than or not equal ($\leq$), greater than ($>$), greater than or not equal ($\geq$), and not equal ($\neq$). These predicates are used to determine if any unusual numbers are present in the program. By randomly selecting a few predicates, a few pairs of variables in the program, and/or a few comparisons of variables to constants, then it is possible to detect infrequent events in very large numbers of program executions while only doing a few comparisons on any given execution. This approach is not that useful for high-volume manufacturing, as it does not guarantee that the measurements from any given part will be properly tested against all of the relevant control limits.

The manufacturing problem requires a more focused approach that ensures gauge observations are sufficiently similar such that reliable algorithm operation can occur. Appropriate checks to control limits need to be performed each and every time a part is measured. This is a new issue for both profiling and runtime profiling, and has not been done in previous efforts. Chapter 4 outlines the statistical theory of how algorithms can be influenced by gauge error.

2.2.1 Limitations of Existing Profiling Techniques

Consider a case where a part has a dimension X. The goal is to determine if the magnitude of X is less than or equal to U_X, *i.e.* $|X| \leq U_X$. A gauge measures the part, and makes an observation labeled Y.

What is the optimal algorithm for determining if $|X| \leq U_X$?

Option A: Test if $|Y| \leq U_X$.

Option B: Test if $|Y| \leq U_T$, for some to be determined U_T.

Option C: Assume $|X| \leq U_X$ always, and not monitor the process.

In many manufacturing applications, Option C will work a large percentage of the time, because engineers design the processes such that the probability $\Pr(|X| \leq U_X) \approx 1$. Options A and B may also be appropriate, depending on the undetermined statistical relationship between X and Y.

It is not obvious that any conventional profiling technique works well in this application, because this problem is not about conventional software bugs. This is a statistical data analysis problem that requires empirical information to solve.

The insight that motivated the development of this thesis was that run-time profiling tools could be used for more than just software problems. They could be used to solve important industrial problems, including the problems solved in Chapters 6.3 through 9. One of these applications was a feedback system. Thus, the background associated with a feedback system is discussed next.

2.3 Manufacturing with Feedback

The industrial applications that will be presented in Chapters 6, 7, and 8 involve metal cutting cells, and solving problems relating to measuring and compensating for errors caused by thermal variation and tool wear. This has been extensively covered in the metalworking literature. The existing literature does not predict which cell/part combinations yield the highest economic returns.

For instance, Gibson and Hoang (1994) points out that three different error compensation strategies exist: in-process, in-cycle, and post-process. However, each of those three strategies require the automotive parts manufacturer to make different capital investments. In-process gauging uses processes similar to Shawky *et al.* (1998), where parts are measured on-line during machining. In-cycle gauging stops the machining process while the measurement is being made, as in the approach from Liu (1999). Trips to the manufacturing plant revealed that both in-process and in-cycle gauging would significantly increase the total cycle time of the CNC machines. Increased cycle times mean more CNC machines are needed for the same output, so it is not commonly used in high-volume applications. Alternatively, post-process gauging measures parts after they have been manufactured and allowed to cool. This makes for accurate measurements, but creates time delays. The time delays cause large amounts of waste if the machining process was done incorrectly. The economic choice is whether it is preferable to tolerate increased off-line waste, or to purchase additional equipment. This is a critical decision for the auto-parts manufacturer, as production is routed to the lowest cost supplier.

A significant number of papers use Neural Networks and/or Expert Systems to help in the diagnosis and feedback of manufacturing data within an automated cell. Bagshaw and Newman (1999) and Bagshaw and Newman (2002) examine feedback within a small manufacturing cell. Similarly,

Gibson and Hoang (1994) applies SPC techniques into an Automatic SPC (ASPC) based control system using an expert system. Hamrol (2000) used neural networks and SPC techniques, again in the context of machining. A review paper on applying Neural Networks to SPC was completed by Zorriassatine and Tannock (1998). The issue with expert system and neural network approaches is that they fail to develop a clear linkage between the inputs the expert systems have, and the relevant background that influences the waste levels in the cell.

Companies have also applied manual SPC based approaches, including those discussed in texts like Vardeman and Jobe (1999). Also, it is clear from visits to manufacturing companies, that many companies have internal procedures, often part-specific, describing the recommended way to measure part dimensions and adjust tool wear accordingly. However, when operator tool wear actions were observed, the operators were not consistently following established procedures. This raised the question as to why the operators thought that what they were doing was better or easier.

Automotive parts manufacturers make large capital investments in manufacturing cells. To justify this investment, they need to be able to determine, in advance, if the chosen cell configuration is appropriate, and to be able to improve the process once production starts. Chapter 7 details different algorithms tested with the machining process, and the selection process used to improve the machining process. The insights gained show that only certain part/process combinations work well in cells with CNC manufacturing and CMM feedback. Research on the manufacturing cells showed that in addition to tuning the feedback algorithm, it was also necessary to consider the impact the material transport algorithms had on productivity and quality.

2.4 Material Transport Algorithms

Flexible material transport systems represent a relatively new area of research. In comparison to metal cutting, which dates back to the industrial revolution, the modern factory robot is a relatively recent invention. Scheduling general flexible manufacturing systems (FMS) was reviewed by Buzacott and Yao (1986). Brandin (1996) demonstrated it was possible to do on-line supervisory control of a manufacturing cell. Supervisory control focuses on cell and factory scale issues like machine scheduling and availability, and assumes all control actions are governed by high-level predictable routing specifications. This research focuses on the less predictable low-level process issues, including the waste implications of feedback loops. Brandin (1996) omits any reference to scrap, rework, inspection and operator involvement in cell operation.

Mears *et al.* (2009) analyzed the integration of a CMM into the CNC machine tool, and some of the resulting feedback and control system issues. Bering *et al.* (2002) described generalized quality information feedback. Bagshaw and Newman (1999) and Gibson and Hoang (1994) explored Statistical Process Control, SPC, with CNC feedback. Later, Bering and Veldhuis (2010c) detailed the practical issues involved with CMM feedback systems and the appropriate feedback algorithms for CNC machining cells experiencing special cause variation.

None of the scheduling or feedback control papers deal with the practical issues experienced in trying to optimize the capital equipment decisions involved in making the automated cell introduced in Chapter 7. As addressed in Bering and Veldhuis (2010b), the CNC to CMM feedback process interacts with the material transport system, in a manner unique to this cell configuration.

The problems encountered with the CNC manufacturing with CMM feedback cells inspired the development of the runtime profiler. It made a convincing case for the need to have a general purpose tool to profile and tune production gauge and control systems. The resulting runtime profiler was tested on a spring orientation problem, which is described next.

2.5 The Spring Orientation Problem

The spring orientation detection application was implemented through a machine-vision system. It is an attribute gauging problem, where conventional GR&R analysis does not yield useful results. Even though GR&R analysis fails, acceptance sampling theory continues to apply, and the limitations in acceptance sampling are discussed in §2.5.1. Acceptance sampling theory defines the limits of what can be accomplished by sampling.

Obtaining correct spring pitch is crucial in certain applications. Inserting the springs into the engine in the correct orientation affects engine performance, emissions, and fuel economy.

A fundamental issue present in this gauging application, and many others, is that it can be difficult to create provable engineering theory to relate the gauge results to customer requirements. The spring orientation application is one of those applications. Too many failure modes exist, and it quickly becomes difficult to show which ones may cause outgoing defects. When profiling data is analyzed, it quickly becomes obvious that not all these failure modes occur, and of the ones that do occur, many cannot interact in a manner likely to cause a gauge to accept a defective part and cause a problem with outgoing defects. The spring orientation problem was finally understood and solved by using the rogue events analysis and the runtime profiler as presented in this thesis.

2.5.1 Attributes and Attribute Acceptance Sampling

An attribute is a characteristic of the part being measured, and has the property that it is either present or it is not. An example of this in the spring inspection problem, is that the spring is either correctly oriented or it is not. The mechanical system prevents any intermediate orientations from reaching the end-user.

Traditionally, attribute acceptance sampling is used to tell if a "lot" of product uniformly has the same attribute. The size of each "lot" is application-specific, and for the purposes of this section it will be assumed to have N parts. An attribute acceptance sampling process works by gathering testing a sample of n parts, where $1 \leq n \leq N$. For $n < N$, there are many different strategies for selecting the optimal set of parts to sample. However, assuming an ideal attribute gauge, for any sampling plan with $n < N$, there exists the possibility of that defects will exist in the $N - n$ parts that have not been inspected. Formally, the average number of outgoing defects can be calculated, and this number is referred to as Average Outgoing Quality (AOQ).

To achieve an AOQ result meaningful to the 1 ppb accuracy level, either n must be larger than 1 billion $n \geq 10^9$ for ideal gauges. In practice, this implies extremely large lot sizes on the order $N > n \geq 10^9$. In many industrial applications, it is not feasible to store, batch, and gauge parts in lots of over one billion parts.

For critical applications where very low levels of outgoing defects must be achieved and the lot size $N \ll 10^9$, attribute acceptance sampling theory indicates that $n = N$, and every part should be inspected by the gauge. Formally, this results in an AOQ of zero, implying no outgoing defects will occur (assuming an ideal gauge).

Attribute acceptance sampling commonly uses two definitions for error:

1. Producer's Risk (Type I Error) is the likelihood that an attribute acceptance sampling test will fail, even though customer's quality expectations are met. Assume the sample size is much smaller than the lot size, *i.e.*$n \ll N$. If a large number of reject parts are present in the measured sample of size n, and a small number of defective parts are present in the unmeasured bacth of size $N - n$, then the producer risks scrapping all N parts even though a large precentage of the unmeasured parts $N - n$ might be good.

2. Consumer's Risk (Type II Error) is the likelihood that an attribute acceptance test will pass, even though the customer's quality expectations are not met. If a small number of reject parts (or no rejects) are present in the measured sample of size n, and a significant number of defective parts are present in the unmeasured bacth of size $N - n$, then the acceptance sampling test will pass, even though significant numbers of defective parts are present in the batch. This will trigger considerable costs for the consumer.

With $n = N$ and ideal gauges, the producer's risk and the consumer's risk should both be zero. However, real production lines do not have ideal gauges.

Attribute acceptance tests are commonly used in a production environment as part of line startup. Many of the industrial plants used a $n = N = 10,000$ part attribute sample test to evaluate whether a new production line and gauging system was suitable for production. A new production line is run until the production gauge accepts 10,000 parts. All of these parts are then tested using secondary gauges in the quality department, that can be assumed to be ideal.

The advantage of the 10,000 part commissioning test is that if any part passes the production gauge, and is rejected on the secondary (ideal) gauges, then it can be safely assumed that the production gauge is not suitable for production. However, the converse is not true.

Specifically, for any n, where $n < N$ and $n \ll 10^9$, if all parts pass the secondary gauge (no rejects), then it does not imply the production line/gauge combination is reliable to a 10^{-9} or 1 ppb reliability specification. One of the key accomplishments of this thesis is to provide some guidance in how to determine which production line/gauge combinations are reliable to 1 ppb, and which are not. However, other methods of assessing gauge performance also exist.

2.5.2 Variables and Variable Acceptance Sampling

It is possible that a part has a variable characteristic, like part height. Formally, these variable characteristics are the random variables in the manner described in Chapter 3. It is possible to determine some representative characteristics of these variables, like the population mean and variance, assume a statistical distribution, like the normal distribution, and make predictions about the properties of these variables. Variable acceptance sampling tests can have been developed around these assumptions.

Vardeman and Jobe (1999, p. 473) points out the limitation of variable acceptance sampling. Specifically, unless the process distribution actually follows the normal distribution, the resulting predictions are often useless for predicting the shape of the tails of the distribution. Thus it is necessary to monitor the system to verify that the expected results continue to apply, and this was one of the motivations behind developing the run-time profiler that is capable of collecting information on the distributions of random variables.

Obviously, the performance of any acceptance sampling plan will be limited by the accuracy of the gauge doing the inspection. Formally, the GR&R procedure described in the next section is often used to estimate the accuracy of the gauge.

2.5.3 GR&R

When a gauge measures a part, measurement error will be present. Assume each part has a single fixed value of process error, an unknown amount of measurement error, and a resulting gauge reading (observation). It is possible to make certain assumptions, and to identify and determine the influence of random measurement error and operators on the observations. For certain classes of gauges, operator technique affects measurement error. Formally, the GR&R methods from MSA (AIAG, 2003) can be used to identify the measurement error and the influence of operators on gauge readings (observations).

Wall (1996) and Wall (1997) uses a similar set of assumptions to estimate the Probability of Detection (POD) and the Probability of False Indication (PFI). These quantities can be related in concept to Type I (Producer's Risk) and Type II (Consumer's Risk) from acceptance sampling, however POD and PFI are fundamentally different because Wall is attempting to measure the performance of a gauge and in acceptance sampling one is measuring a lot of parts.

Wall's work is particularly important when considering the case of a borderline attribute gauge. For instance, consider the case of an attribute gauge where the measurement error is sufficiently large, that in frequently occurring cases a region exists where an ambiguity is present between whether the attribute is present (accept) or not (reject). For such a gauge, it can be difficult to choose an appropriate pass-zone where the POD and PFI have suitable values.

The manufacturing plants visited in the development of this thesis were keenly aware of gauging problems relating to thresholds, POD and PFI. However, in the case of the spring inspection problem, these issues did not apply. There was no obvious overlapping border-zone that was causing the problems.

Binary attribute gauges have been examined previously, including in van Wieringena and de Mastb (2008), Danila *et al.* (2012b), Danila *et al.* (2012a) and Danila (2012). Binary attribute gauges measure a two-state (binary) attribute and output a binary (accept/reject) result. However, the attribute gauges of interest in this thesis are not true binary attribute gauges. Industrial high-reliability gauges may measure a binary (present/not-present) attribute and then output an accept/reject result. However, for complex gauges, the accept/reject result does not correlate to the present/not-present attribute. Specifically, high-reliability gauges are invariably configured that if any significant automation or measurement failure occurs, then out of an abundance of caution, the part being measured will almost always be rejected even if it is good (not defective). Which automation and measurement failures are considered significant is usually a matter of interpretation (and potential ambiguity).

In theory, given sufficient information, it is possible to develop a sufficiently sophisticated model of a complex system to eliminate any ambiguity and to determine if the system will be reliable. However, as a practical matter, it is difficult to get this information. As such, the applications discussed in Chapters 6 through 9 were developed by collecting profile data from deployed manufacturing cells first, and then the models presented in Chapter 4 were developed. This represents a fundamentally different way of developing and commissioning a new production gauge.

2.6 Summary

Zero defects is an important quality goal for the automotive industry. Considering the large number of parts in a vehicle, small numbers of defects at the part supplier level can result in large numbers of defects inside the finished automobile. Significant nonborderline defects should cost more than

borderline defects. These defects are particularly difficult to explain with methodologies that assume the normal distribution applies, because manufacturing processes with a $C_{PK} \geq 2$ should generate very few defects.

Statistical process control techniques, including control charting, are in widespread use inside the automotive industry. New control charts are being developed for new applications of SPC. Effective use of control charting requires operators to investigate problems and make improvements such that the problem will not repeat. This process is referred to as continuous improvement. The number of operators in manufacturing has decreased as machine speeds and productivity has improved. This has made it more difficult to do investigations and implement continuous improvements.

Profiling is a well-developed area of Software Engineering. Profiling can be computationally expensive, hence the common practice was to do profiling during initial software development, and then remove the profiling code before the program is shipped. This practice continues to this day. However, it is no longer as expensive as it once was to instrument production systems. As such, runtime profiling is about instrumenting production systems, capturing data from them, storing and transmitting the data for later analysis, and then analyzing the data to some useful economic purpose. In the case of this thesis, no one has used SPC with runtime profiling to debug industrial gauges in the manner suggested here.

The industrial applications are from researched areas of engineering. A key challenge for CNC production cells using CMMs to acquire measurements for feedback is that prior work was unable to explain why some systems worked in some applications, and many did not work in all applications. Guidance was needed on how to determine which feedback system produced superior economic results across many different part and cell configurations. Also, the CNC production cells with CMM feedback have their own challenges, specific to the application. This is particularly influenced by

the need to involve the operator inside the feedback loop. A synergy is required such that the feedback loop appropriately corrects the process when necessary; the operator performs the other critical production tasks that the feedback system cannot perform; and the synergies of both systems are utilized to improve the economic results of the cell.

The vision system used in the spring orientation example is a high-reliability attribute gauge, and this presents difficult challenges. The solutions to this problem will be explored in Chapter 4.

Chapter 3

Background Theory

3.1 Statistical Theory

3.1.1 Events

Probability can be expressed in terms of events. In this thesis, an event will be labeled E_A, where E denotes an "Event", and the A subscript is used to denote the specific event being analyzed. The probability of an event occurring is related to the probability of the event not occurring

$$\Pr(E_A) = 1 - \Pr(E'_A) \tag{3.1}$$

where E'_A denotes event E_A not occurring. The symbol for either event E_A or the event E_B or both occurring is $E_A \cup E_B$. The symbol for both event E_A and event E_B occurring simultaneously is $E_A \cap E_B$. If it is known that event E_A has occurred, the conditional probability that E_B occurs given E_A can be defined as

$$\Pr(E_B|E_A) = \frac{\Pr(E_B \cap E_A)}{\Pr(E_A)} \tag{3.2}$$

as specified in Helstrom (1984, p. 22). Event E_A is independent from event E_B if and only if

$$\Pr(E_A) = \Pr(E_A|E_B) = \Pr(E_A|E'_B) \tag{3.3}$$

where $\Pr(E_A|E_B)$ represents the conditional probability that event E_A occurs given that E_B is also occurring.

3.1.2 Random Variables

In probability, random variables are functions that map sets of numbered events onto the real number line, such that the probability of occurrence of different sets of events can be determined. Beaumont provides the definition

> "A random variable is a function on a sample space with two properties:
>
> 1. the values it assumes are real numbers; and
> 2. for every real number x, the probability that the value of the function is less than or equal to x can be calculated."
>
> (Beaumont, 2005, p. 81)

Random variables are also referred to as stochastic variables in some texts. In this thesis, random variables are denoted by uppercase Q, S, T, W, X, Y, and Z. The probability notation

$$\Pr(X = x) \tag{3.4}$$

refers to the probability that the random variable X, when observed, has outcome (number) x.

It is important not to confuse the definition of a random variable with that of a variable used in computers or conventional algebra, as they are fundamentally different concepts. In computers, the variable x represents a known or determinable value. For example, a computer program may make the assignment $x = 2$. On the other hand, the random (stochastic) variable X is a function that is the set of all possible numbers that could occur. For instance, if a six-sided die was tossed, then X could be any of $1...6$ with $\Pr(X = 2) = \frac{1}{6}$.

Often it is helpful to know the probability that a random variable is in a range. In this thesis, limits will be denoted with a capital U. For example

$$\Pr(U_1 \leq X \leq U_2) \tag{3.5}$$

represents the probability that the random variable X is between or equal to the limits U_1 and U_2. For example, if a 6-sided die was thrown, then the probability that any of the numbers between 3 and 5 occur, inclusive, is $\Pr(3 \leq X \leq 5) = \frac{1}{2}$.

At many points in the thesis, the limits will be assumed symmetric and denoted U_T, where $U_T = U_2 = -U_1$. The resulting probability will be shown as

$$\Pr(|X| \leq U_T) = \Pr(-U_T \leq X \leq U_T) \tag{3.6}$$

3.1.3 Expected Value

From (Helstrom, 1984, p. 96), the expected value of a random variable is defined as

$$\mathrm{E}(X) = \int_{-\infty}^{\infty} x f_X(x) dx \tag{3.7}$$

where $f_X(x)$ is the density function as described in §3.1.7. The population mean μ_X is the expected value of the random variable X, *i.e.*

$$\mu_X = \mathrm{E}(X) \tag{3.8}$$

The population variance σ_X^2 of X is given by

$$\sigma_X^2 = \mathrm{Var}(X) = \mathrm{E}\left\{[X - \mathrm{E}(X)]^2\right\} \tag{3.9}$$

The quantity σ_X is known as the population standard deviation, and is the square root of the population variance σ_X^2. The covariance of X and Y is given by

$$\mathrm{Cov}(X, Y) = \mathrm{E}\left\{[X - \mathrm{E}(X)][Y - \mathrm{E}(Y)]\right\} \tag{3.10}$$

The covariance and variance are related by the property that $\mathrm{Var}(X) = \mathrm{Cov}(X, X)$.

3.1.4 Random Vectors

It is possible to define random vectors $\boldsymbol{X}$ and conventional vectors x. Vectors and matrices will be shown in bold face. The expected value of a vector is a vector containing the expected values of each component of the vector. Similarly, the expected value of a matrix is a matrix consisting of the expected values of each entry of the matrix. For this thesis, it is assumed that vectors expressed in matrix notation are column matrices.

Using matrix notation, we may define the vector mean and Variance-Covariance matrix $\boldsymbol{\Sigma}$ as

$$\boldsymbol{\mu}_X = \mathrm{E}(\boldsymbol{X}) \tag{3.11}$$

$$\boldsymbol{\Sigma} = \mathrm{E}\left\{[\boldsymbol{X} - \mathrm{E}(\boldsymbol{X})][\boldsymbol{X} - \mathrm{E}(\boldsymbol{X})]^T\right\} \tag{3.12}$$

The $\boldsymbol{\Sigma}$ matrix has variance terms on the diagonal, and covariance elsewhere. If $\boldsymbol{X} = [X_1 ... X_N]^T$, then the corresponding $\boldsymbol{\Sigma}$ matrix is

$$\boldsymbol{\Sigma} = \begin{bmatrix} \mathrm{Var}(X_1) & \mathrm{Cov}(X_1, X_2) & \cdots & \mathrm{Cov}(X_1, X_{n-1}) & \mathrm{Cov}(X_1, X_n) \\ \mathrm{Cov}(X_2, X_1) & \mathrm{Var}(X_2) & \cdots & \mathrm{Cov}(X_2, X_{n-1}) & \mathrm{Cov}(X_2, X_n) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathrm{Cov}(X_{n-1}, X_1) & \mathrm{Cov}(X_{n-1}, X_2) & \cdots & \mathrm{Var}(X_{n-1}) & \mathrm{Cov}(X_{n-1}, X_n) \\ \mathrm{Cov}(X_n, X_1) & \mathrm{Cov}(X_n, X_2) & \cdots & \mathrm{Cov}(X_n, X_{n-1}) & \mathrm{Var}(X_n) \end{bmatrix} \tag{3.13}$$

The $\boldsymbol{\Sigma}$ matrix has additional special properties, including the fact that it is a symmetric matrix, and a matrix $\boldsymbol{\sigma}$ exists such that $\boldsymbol{\sigma^2} = \boldsymbol{\Sigma}$ applies.

3.1.5 Distributions

Many random variables are assumed to be from standard distributions. The following states

$$X \overset{D}{=} \mathcal{N}(\mu, \sigma^2) \tag{3.14}$$

that the random variable is from the normal distribution with mean μ and variance σ^2.

3.1.6 Distribution Function

The cumulative probability function (distribution function) of a normal distribution with mean μ and variance σ^2 is given by

$$\begin{aligned} F(x; \mu, \sigma^2) &= \Pr(X \leq x) \quad \text{where } X \overset{D}{=} \mathcal{N}(\mu, \sigma^2) && (3.15) \\ &= \Phi\left(\frac{x - \mu}{\sigma}\right) && (3.16) \\ &= \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{(t-\mu)^2}{2\sigma^2}} \, \mathrm{d}t && (3.17) \end{aligned}$$

where

F represents the distribution function of the distribution being analyzed.

μ represents the mean of the distribution being analyzed.

σ^2 represents the variance of the distribution being analyzed, and

Φ represents the distribution function of the standard normal distribution, which is a normal distribution with a mean of zero and a variance of one.

For continuous random variables, the derivative of the distribution function exists.

3.1.7 Density Function

The derivative of the distribution function is the probability density function, hereinafter known as the density function. The density function of a normal distribution with mean μ and variance σ^2 is given by

$$f(x;\mu,\sigma^2) = \frac{dF(x;\mu,\sigma^2)}{dx} \tag{3.18}$$

$$= \frac{1}{\sigma}\phi\left(\frac{x-\mu}{\sigma}\right) \tag{3.19}$$

$$= \frac{1}{\sigma\sqrt{2\pi}}e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{3.20}$$

where

f represents the density function of the distribution being analyzed.

F represents the distribution function of the distribution being analyzed.

μ represents the mean of the distribution being analyzed.

σ^2 represents the variance of the distribution being analyzed, and

ϕ represents the density function of the standard normal distribution, which is a normal distribution with a mean of zero and a variance of one.

For continuous random variables, the derivative of the distribution function exists.

The normal distribution is symmetric around the mean μ. This property is often used to calculate the size of the tails of a distribution, using the formula

$$\Pr\left(x \leq |X - \mu|\right) = 2\Phi\left(\frac{-x}{\sigma}\right) \tag{3.21}$$

where $x \geq 0$ and $X \sim \mathcal{N}(\mu, \sigma^2)$. (3.21) was used to calculate the values presented in the Pr(Out-of-Spec) column in Table 2.1.

3.1.8 Independence of Random Variables

Random variables can be independent. Specifically, the random variable Y is independent of X if and only if

$$\Pr(Y \leq y) = \Pr(Y \leq y | X = x) \quad \text{for all } x, y \tag{3.22}$$

A necessary condition for independence is that the variables are not correlated, *i.e.* $\mathrm{Cov}(X, Y) = 0$. Normal distributions have the special property that two normal random variables will be independent if and only if $\mathrm{Cov}(X, Y) = 0$.

Statistical independence is critical to reliable systems, as it affects system reliability. Consider two events E_A and E_B with two corresponding random variables X_A and X_B. If E_A and E_B are independent, the probability that both occur simultaneously is

$$\Pr(E_A \cap E_B) = \Pr(E_A) \Pr(E_B) \tag{3.23}$$

The practical consequence of this result is that it is extremely unlikely that two rare events E_A and E_B occur simultaneously. If E_A and E_B are not independent, the probability that both occur simultaneously is

$$\Pr(E_A \cap E_B) = \Pr(E_A)\Pr(E_B|E_A) = \Pr(E_A|E_B)\Pr(E_B) \tag{3.24}$$

where $1 \geq \Pr(E_B|E_A) \geq 0$ and $1 \geq \Pr(E_A|E_B) \geq 0$. When $\Pr(E_B|E_A)$ and/or $\Pr(E_B|E_A)$ are unknown, then the probability of both E_A and E_B occurring simultaneously is bounded solely by the less frequently occurring event. Specifically

$$\Pr(E_A \cap E_B) \leq \min\left(\Pr(E_A), \Pr(E_B)\right) \tag{3.25}$$

where $\min$ represents the minimum of $\Pr(E_A)$ and $\Pr(E_B)$.

Proving independence is crucial for analysis of high-reliability systems. From a practical perspective this is done by analyzing the covariance of the underlying random variables X_A and X_B, and making the argument that they should be independent based on knowledge of the system and the relevant distributions.

3.1.9 Mixture Distributions

In mixture distributions, the random variable X may take on outcomes from two different distributions. The notation used in this thesis to define a random variable from a mixture distribution is

$$X \overset{D}{=} (1-\alpha)\mathcal{P}_1(\mu_1, \sigma_1^2) + \alpha\mathcal{P}_2(\mu_2, \sigma_2^2) \tag{3.26}$$

where

X is the random variable.

$\mathcal{P}_1$ represents the first distribution from which X may take on values.

$\mathcal{P}_2$ represents the second distribution from which X may take on values.

α represents the probability that X will take on values from the second distribution.

$1-\alpha$ represents the probability that X will take on values from the first distribution.

μ_1 represents the mean of the first distribution.

μ_2 represents the mean of the second distribution.

σ_1^2 represents the variance of the first distribution.

σ_2^2 represents the variance of the second distribution.

When X takes on a value from $\mathcal{P}_1$ the associated event will be denoted E_1. Similarly, when X takes on a value from $\mathcal{P}_2$ the associated event will be denoted E_2.

When mixture distributions with sufficiently small α are present, many sampling methods often return estimates of the mean μ_1 and the variance σ_1, and yield no useful information about the mean μ_2 and the variance σ_2. Importantly, if the α is sufficiently small, a sample of ten parts might not contain any parts corresponding to the second mixture. Alternatively, if outlier rejection is used, then a sample from the second distribution may be dismissed as an outlier and ignored. A ten part sample is one of the suggested sample sizes for the GR&R procedure from AIAG (2003).

This thesis is based on sampling manufacturing data using automated techniques, including runtime profiling, that have the capability of recording every measurement (observation) from automated gauges. As such, the data being analyzed was unaffected by the problems caused by gathering small samples and extrapolating.

3.1.10 Definition of Bachmann-Landau Notation

It is occasionally helpful to describe the asymptotic properties and convergence properties of functions in a compact manner. In this thesis, Bachmann-Landau "Big O" notation will be used to do this. Knuth (1976, p. 19) defines:

> "$\mathrm{O}[f(n)]$ denotes the set of all $g(n)$ such that there exists positive constants C and n_0 with $|g(n)| < Cf(n)$ for all $n > n_0$.
>
> $\Omega[f(n)]$ denotes the set of all $g(n)$ such that there exists positive constants C and n_0 with $g(n) > Cf(n)$ for all $n > n_0$."

The practical significance of the above functions is that they allow the straightforward comparison of functions as n becomes large. If $g(n) = \mathrm{O}[f(n)]$, then an upper bound on the behaviour of $g(n)$ has been established. As Knuth (1976) notes, using $=$ for set notation is misleading, however this is the standard notation widely used in literature.

Example #1: If the execution time of an algorithm is $\mathrm{O}(1)$, it means that the algorithm will execute in a finite amount of time.

Example #2: The sample mean converges on the population mean in $\mathrm{O}\left(\frac{1}{\sqrt{n}}\right)$ where n is the size of the sample used to compute the sample mean. This means that in the worst case, if we want to double the accuracy of the estimate (halve the worst-case error), then it is necessary to quadruple the sample size.

3.2 Minimization in the Presence of Information with Limited Accuracy

In the development of this thesis, some confusion occurred between the practice of using a derivative to find a minimum, and the needs of business people

who would settle for a sufficiently small result near a minimum. From Stewart (2003, p. 279), an absolute minimum exists at c whenever $f(x) \geq f(c)$ for all $x \in D$, where D is the domain of f. From Stewart (2003, p. 280), a local minimum exists at c whenever $f(x) \geq f(c)$, and for all x near c. The local and global minimums may occur at any critical point, which is where the derivative is zero or does not exist, Stewart (2003, p. 283). Additionally, in terms of numerical analysis, it is common to approximate a minimum value with a sufficiently accurate estimate. This is explained in Press *et al.* (1992, p. 398).

An issue with random variables is that the sample mean converges on the population mean as $\mathrm{O}\left(\frac{1}{\sqrt{n}}\right)$, where n is the sample size. See (Papoulis, 1965, pp. 259-260) for a derivation. With limited accuracy, approximations are necessary.

Regions near a minimum can be found with methods that do not require the computation of a derivative. Based on the previously mentioned definitions outlined in Stewart (2003, pp. 110,279-283), four techniques of minimization can be outlined:

1. **Reduce until a constraint is reached.** Often physical systems have fundamental constraints. When these constraints are present, it is sometimes possible to estimate minimum values of functions based on the physical properties of the system. As these physical properties are often much less influenced by statistical effects, the accuracy of the resulting conclusions is much improved.

2. **Mathematical methods not involving explicit derivatives.** Sometimes, it is possible to determine a minimum analytically without the computation of a derivative.

3. **Selection.** The minimum of a function, $f(c)$, in a domain $c \in \{1...n\}$, can be found by computing every $f(c)$ in the domain and selecting the lowest. When it is expensive to evaluate $f(c)$ directly, simulations and/or numerical methods can be used instead.

4. **Bounding.** If an upper bound can be found, and the upper bound reduced until the problem has no remaining economic, numerical and/or statistical significance, then a point has been found that sufficiently approximates a minimum for most practical purposes. Formally, bounding does not minimize a function, or even prove a minimum exists. It simply removes the influence of a parameter on the final result. Often, for the industrial applications in this thesis, this result will be an approximation of a theoretical limit, and truncated for numerical, cost or practical reasons. However, bounding does work in the limit, as shown by Theorem 2 from Stewart (2003, p. 110).

When the people from the industrial plants visited were discussing "finding the minimum", they were typically referring to using one of the above four techniques.

3.3 Process Capability Index

The process capability index C_{PK} provides an easy way to compare the relative capabilities of manufacturing processes. Importantly, it assumes the processes are normally distributed, and the average and standard deviation used in the C_{PK} calculation are usually estimated from a small sample of parts measured during the course of a manufacturing run. Rewritten in the notation used in this thesis, Montgomery (2008, p. 355) defines C_{PK} as

$$C_{PK} = \frac{\min\left(U_2 - \mu, \mu - U_1\right)}{3\sigma} \tag{3.27}$$

where

μ is the population mean (or an estimate of the population mean).

σ is the population standard deviation (or an estimate of the population standard deviation).

U_1 is the lower tolerance limit as defined on the part-print.

U_2 is the upper tolerance limit as defined on the part-print.

Some authors use the notation $\hat{C}_{PK}$ to denote a C_{PK} computed from a sample.

The practical significance of the C_{PK} is that a process with a $C_{PK} \geq 2$ should produce less than 2 ppb of reject parts, assuming the process is normally distributed. In Chapter 4 and in particular in §4.3.2.1, it will be assumed that the process follows a mixture model, and as such, it is not normally distributed. If deviations from the normal distribution model occur sufficiently often, then their presence can be detected via control charting.

3.4 Control Charting

Control charting is the major Statistical Process Control (SPC) technique in use in many industrial settings. The philosophy is that by quickly detecting changes in the process mean and/or process variation, it is possible to have the operators perform investigations and appropriately correct the manufacturing process to minimize overall variation.

The most widely used control chart is the Shewhart X Bar R control chart, which is described in Shewhart (1931) and Shewhart (1939). Since that time, many other control charts have been developed, and new control charts continue to be developed for special applications, as noted by Chen *et al.* (2008).

Many control charting techniques, including the Shewhart X Bar R control chart, employ the concept of subgrouping. Samples are taken from N_{SG} consecutive parts, and these are individually labelled $x_{SG,1}, ...x_{SG,N_{SG}}$ and collectively labeled $\mathbf{x}_{SG}$ when expressed as an N_{SG} element vector.

Control charts were developed before the development of shop floor computers. Typically, one subgroup might consist of 5 consecutive parts in 100. This allowed both the subgrouping and control charting to be implemented manually. The operator would compute

$$\bar{x}_{SG} = \text{average}(x_{SG,1}, ...x_{SG,N_{SG}}) \tag{3.28}$$

$$r_{SG} = \text{range}(x_{SG,1}, ...x_{SG,N_{SG}}) \tag{3.29}$$

and graph the results as shown in Figure 3.1. Points inside the range given by

$$U_{X1} \leq \bar{x}_{SG} \leq U_{X2} \tag{3.30}$$

$$U_{R1} \leq r_{SG} \leq U_{R2} \tag{3.31}$$

are labeled “in-control”. Points outside the above limits are labeled “out-of-control”, and represent an “alarm”. Note that the control limits are set based on the statistical capability of the process, usually at the $\mu \pm 3\sigma$ marks. As such, “in-control” does not imply “in-spec” (within specification), and “out-of-control” does not imply “out-of-spec”. Instead, a well-designed process will have a cushion, between the control limits and the specification limits, and the size of the cushion is related to the achievable process C_{PK}.

Modern methods use computers to automatically plot the control charts, and sometimes computers are capable of interpreting the charts and automatically taking certain corrective actions, as proposed in Bering *et al.*

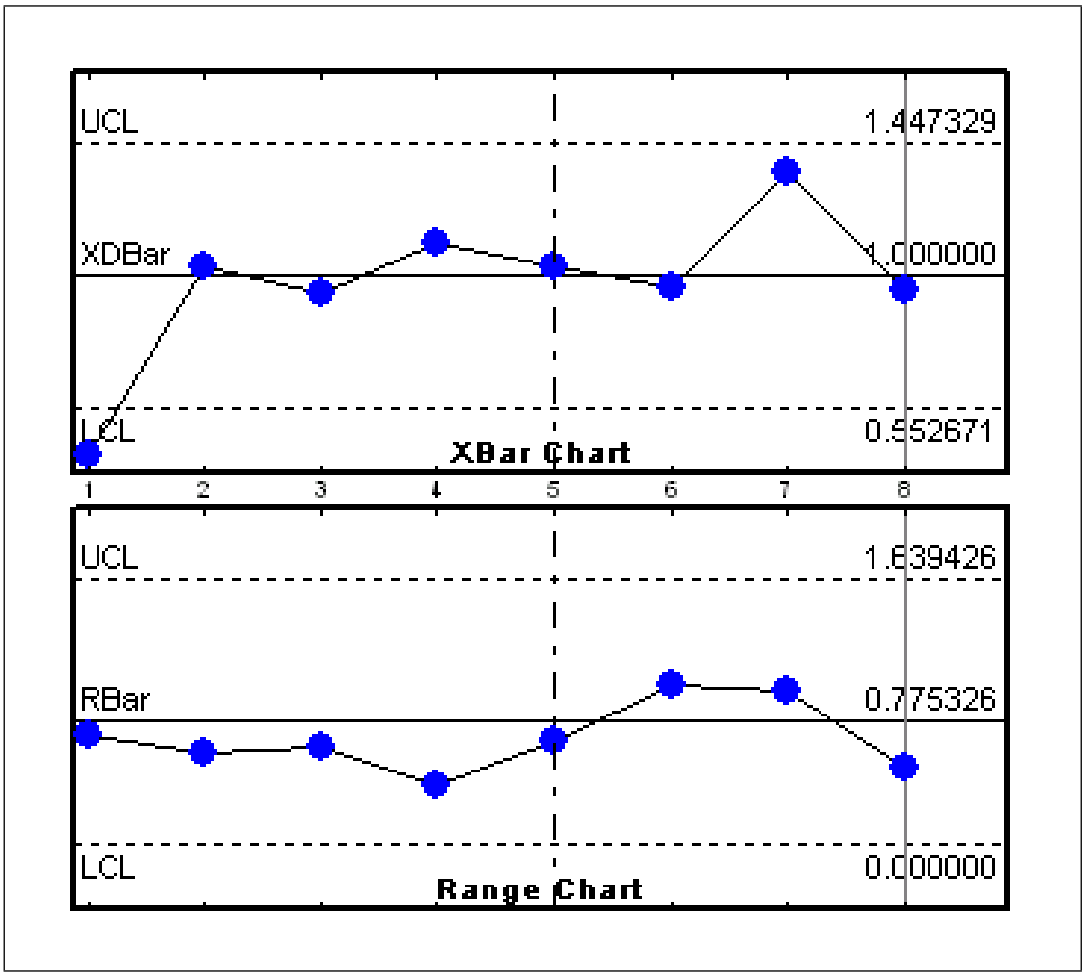

Figure 3.1: Screen Capture of X Bar R Chart from MeasurLink (Mitutoyo)

(2002). The practical problem with control charts is that they were originally developed as a manual method. It was possible to depend on the operators to perform certain functions, and these functions have no equivalent in automatic systems. These functions include the following scenarios

1. The operators were responsible for identifying when incorrect gauge readings were taking place.
2. The operators were responsible for investigating and then performing a correction. These investigations were important because alarms could have had multiple causes, and the same corrective actions were not always appropriate.
3. Continuous improvement. The control chart was intended to be used as part of broader quality improvement initiative, in which systems and processes were improved to reduce variation.

This thesis proposes several fundamental changes to resolve these challenges.

First, automatic systems have the capability of analyzing vast amounts of data, unlike human operators. As such, it is desirable to redefine the subgroup such that it is the gauge inputs, and not infrequent collections of part dimensions (gauge outputs).

Secondly, it is helpful to use automated tools when performing continuous improvement initiatives. Previously, manual investigations were needed. The approach suggested in this thesis is to use computer-based tools, like the proposed runtime profiler.

Thirdly, a common complaint is that the excessive use of control charting creates false alarms. Specifically, if the probability of a false alarm occurring on a single control chart is $\alpha = 0.01$ and $N = 400$ control charts are used, and assuming every chart tracks an independent variable, then false alarms will occur very frequently $N\alpha = 4$ per subgroup. This thesis uses existing control charting methodologies. However, it does so in a very different and unique way. Sometimes, Boole's inequality and the Bonferroni correction are used to compensate systems when large numbers of control charts are used. However, this thesis will use the multivariate control chart variable Q. See §4.4.1.2 for details.

This thesis embraces the opportunities provided by the modern production and gauge technologies, to overcome problems in automated cells.

3.5 Summary

Probability notation uses random variables to represent sets of possible outcomes. These random variables often come from standard distributions including the normal distribution.

Mixture distributions represent distributions in which two or more individual distributions are combined together. Sampling techniques may not be effective at detecting the existence of both of the constituent distributions when the sample size is limited and the probability of occurrence of one of the distributions is sufficiently small.

When minimization is discussed in this thesis, it means selecting amongst the available options and driving the process variables, to a region sufficiently close to a minimum that it meets business objectives. This is in keeping with existing definitions of minimum used in numerical methods.

A spectrum of technologies including mixture variables, multivariate analysis, and runtime profiling, will be used to solve the challenges presented in modern manufacturing applications.

Chapter 4

Theory

4.1 Increase Production, Reduce Waste

In order to meet the customer's quality expectations, the production machinery has to make decisions. This chapter examines the impact of binary decisions based on measured data (observations) assuming mixture models for both the process error and the gauge error.

Two major types of applications are examined. The first application involves feedback. The goal is to maximize the number of acceptable parts produced on the production line by deciding whether feedback should be performed (chase), or feedback should be delayed (wait). The second application is an accept / reject gauge. Based on an observation, the gauge must decide whether the part is acceptable to the customer (accept), or should be quarantined and rejected as waste, rework and/or scrap (reject). In a production cell, both of these abstract decisions may be implemented on the same gauge.

The conventional solution is to assume

$$Y = X + W \tag{4.1}$$

$$W \stackrel{D}{=} \mathcal{P}_3(\mu_3, \sigma_3^2) \tag{4.2}$$

where $\mathcal{P}_3(\mu_3, \sigma_3^2)$ is a suitable measurement distribution with mean μ_3 and variance σ_3^2. The MSA manual from AIAG (2003) specifies several methods including a GR&R method and a method to measure the bias μ_3 in the gauge reading. An important underlying assumption in the AIAG/MSA methodology is that $\mathcal{P}_3$ has suitably small tails, such as when it can be assumed to be normally distributed, *i.e.* $\mathcal{P}_3(\mu_3, \sigma_3^2)$ is $\mathcal{N}(\mu_3, \sigma_3^2)$. When it can be assumed that $\mu_3 \approx 0$, the standard deviation is suitably small σ_3, and the tails are sufficiently small, then

$$Y \approx X \tag{4.3}$$

because $W \approx 0$. Specifically, if the total measurement repeatability and reproducibility (R&R) as measured by the GR&R is less than 10%, then the tolerance zone is greater than $51.5\sigma_3$, following the GR&R methodology from the Measurement Systems Analysis (AIAG, 2003). If the process goal is $C_{PK} \approx 2$, then σ_1 will be the dominant standard deviation for C_{PK} as σ_3 will only have a small impact on the calculation. Many applications have larger repeatability and reproducibility numbers, and many applications have smaller C_{PK} numbers. However, the point of the MSA manual and SPC procedures is that only suitably good gauges are used in production. Using gauges with excessively large R&R numbers makes it impossible to achieve a $C_{PK} \geq 2$.

The problem with this procedure is that it is only valid if the gauge continues to work in production as flawlessly as it did during a brief preproduction test. Additional, in-process gauge checks are also performed, however these checks are not completely effective at preventing all measurement problems. In practical systems, measurement related problems (gauge rogue events) occur frequently enough to have a measurable impact on cell performance. Thus, the impact of rogue events on the measuring process must be modeled.

4.1.1 Rogue Distributions

The ideas presented in this thesis came from the observation of many practical manufacturing cells that were not getting the returns expected from the previous justification. Sophisticated data logging (profiling) tools were used that were capable of recording every gauge observation Y and the resulting actions A or A'. With these automated tools, it was no longer necessary to use small sample methods and extrapolate using the normal distribution. Additionally, the equipment was highly programmable, so it was relatively easy to switch control algorithms. However, the automotive suppliers only wanted changes that would result in increased production and reduced waste. This was a challenge, as the observed behaviour of the production cells did not match the existing mathematical models. Thus, new models were needed.

One of the key observations from the data analysis was that the tails on the distributions were significantly heavier than what was predicted from using a normal model. These heavy tails were attributable to events happening either in the measurement process (rogue gauge events) or in the part production process (rogue process events). The presence of heavy tails resulted in a new mixture model being developed to describe the part production and measurement process

$$X \overset{D}{=} (1-\alpha)\,\mathcal{P}_1\left(0,\sigma_1^2\right) + \alpha\mathcal{R}_2\left(\mu_2,\sigma_2^2\right) \tag{4.4}$$

$$W \overset{D}{=} (1-\beta)\,\mathcal{P}_3\left(0,\sigma_3^2\right) + \beta\mathcal{R}_4\left(\mu_4,\sigma_4^2\right) \tag{4.5}$$

$$Y = X + W \tag{4.6}$$

where

X is the process error.

W is the gauge error.

Y is the observation, which reflects the total error (both process and measurement) as measured by the gauge.

$\mathcal{P}_1$ is the process distribution which occurs with event E_1, probability $1-\alpha$ and density function $f_1(x)$.

$\mathcal{R}_2$ is the rogue process distribution which occurs with event E_2, probability α and density function $f_2(x)$.

$\mathcal{P}_3$ is the measurement distribution which occurs with event E_3, probability $1-\beta$ and density function $f_3(w)$.

$\mathcal{R}_4$ is the rogue measurement distribution which occurs with event E_4, probability β and density function $f_4(w)$.

μ_2, μ_4 are the means of the rogue process and rogue measurement distributions, respectively.

σ_i^2, $i \in \{1...4\}$ are the variances of the process, rogue process, measurement, and rogue measurement distributions, respectively.

In a production application, a gauge observes Y and inferences about X and W are made indirectly.

The mixture distribution for Y is given by

$$\begin{aligned} Y \overset{D}{=} \; & (1-\alpha)(1-\beta)\mathcal{P}_{13}(0, \sigma_1^2 + \sigma_3^2) + \\ & \alpha(1-\beta)\mathcal{R}_{23}(\mu_2, \sigma_2^2 + \sigma_3^2) + \\ & (1-\alpha)\beta\mathcal{R}_{14}(\mu_4, \sigma_1^2 + \sigma_4^2) + \\ & \alpha\beta\mathcal{R}_{24}(\mu_2 + \mu_4, \sigma_2^2 + \sigma_4^2) \end{aligned} \tag{4.7}$$

where

$\mathcal{P}_{13}$ is the main (rogue-free) distribution, and has density function $f_{13}(y)$ and occurs when the combined event $E_{13} = E_1 \cap E_3$ occurs.

$\mathcal{R}_{23}$ is the process rogue distribution, and has density function $f_{23}(y)$ and occurs when the combined event $E_{23} = E_2 \cap E_3$ occurs.

$\mathcal{R}_{14}$ is the measurement rogue distribution, and has density function $f_{14}(y)$ and occurs when the combined event $E_{14} = E_1 \cap E_4$ occurs.

$\mathcal{R}_{24}$ is the combined rogue distribution, and has density function $f_{24}(y)$ and occurs when the combined event $E_{24} = E_2 \cap E_4$ occurs.

Figure 4.1 shows the density function of the observations Y with no rogue distributions, and with a rogue process distribution. It was generated with $\alpha = 0$ (top) and $\alpha = \frac{1}{40}$ (bottom). Additionally, it was assumed that the relevant process and rogue distributions were normally distributed; and $\beta = 0$, $\mu_1 = \frac{1}{6}$, $\sigma_1 = \frac{1}{6}$, $\sigma_2 = 20$, $\sigma_3 = 2$. These values are similar to the values from the industrial application that are covered in Chapters 7 and 8.

Conventional sampling measurements with outlier rejection can be misleading. With no expectation of rogue distributions, the top half of Figure 4.1 shows a normal distribution. The expected probability of a reject part from this normal distribution is $\Pr(|Y| > U_T) \approx 8 \cdot 10^{-6}$, where $U_T = 10$. If the actual distribution contains rogue parts, as in the lower half of Figure 4.1, with the same tolerance band the expected probability of a reject part increases to $\Pr(|Y| > U_T) \approx 0.015$.

Being out by a few orders of magnitude on the predicted numbers of rejects is a significant problem in itself. However, many cells quarantine more than one part for every defect found. This is further explained in Chapter 6.

Given the significance of the rogue distributions, the question was asked: Why had these rogue distributions not been noticed before?

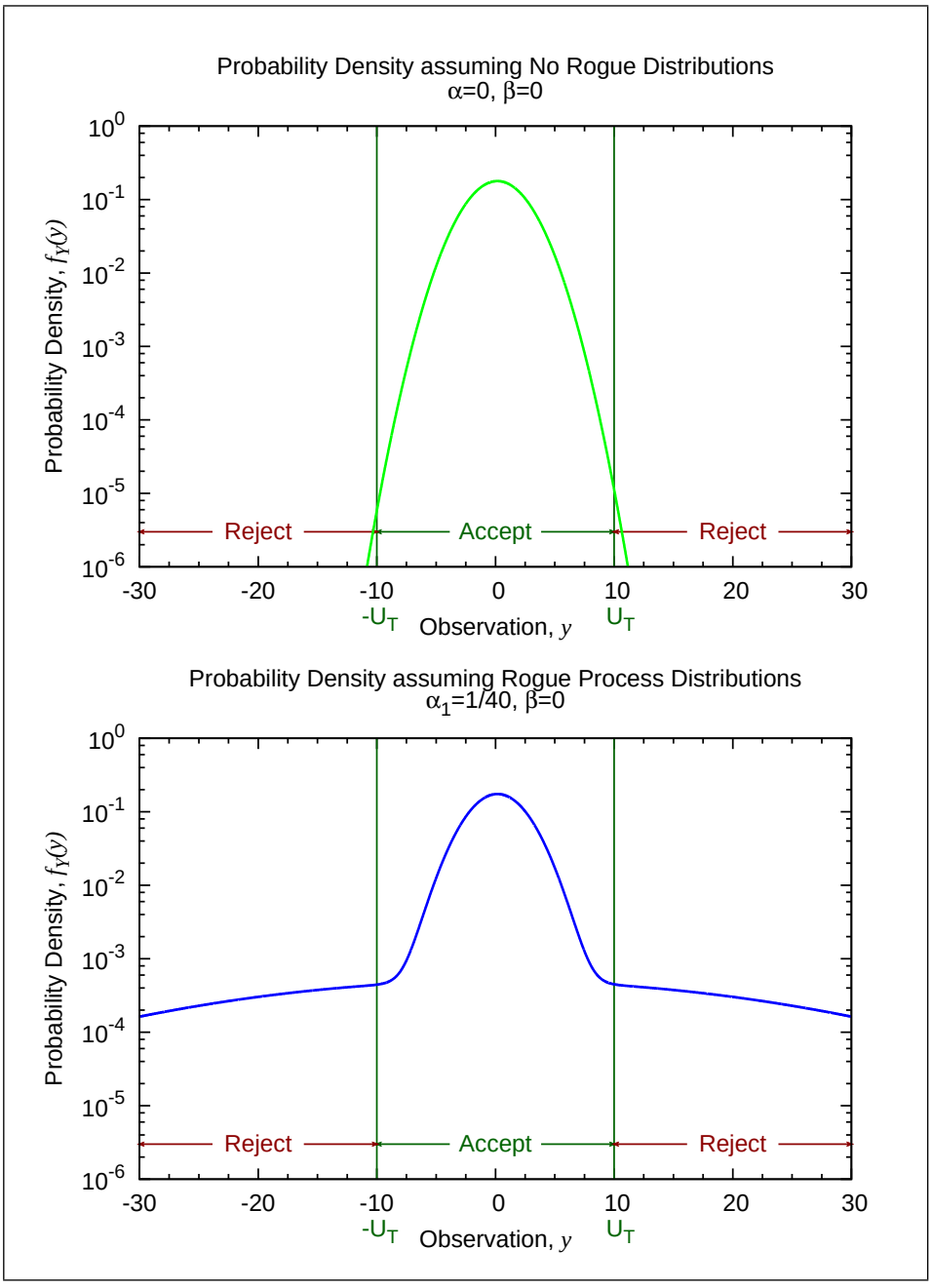

Figure 4.1: Graph of Observations Y for the case where no rogue distributions are present and the case where rogue process distributions are present. This graph assumes normal distributions and mixtures of normal distributions. This graph uses a logarithmic scale on the ordinate axis, see Figure 4.2 for a linear scale.

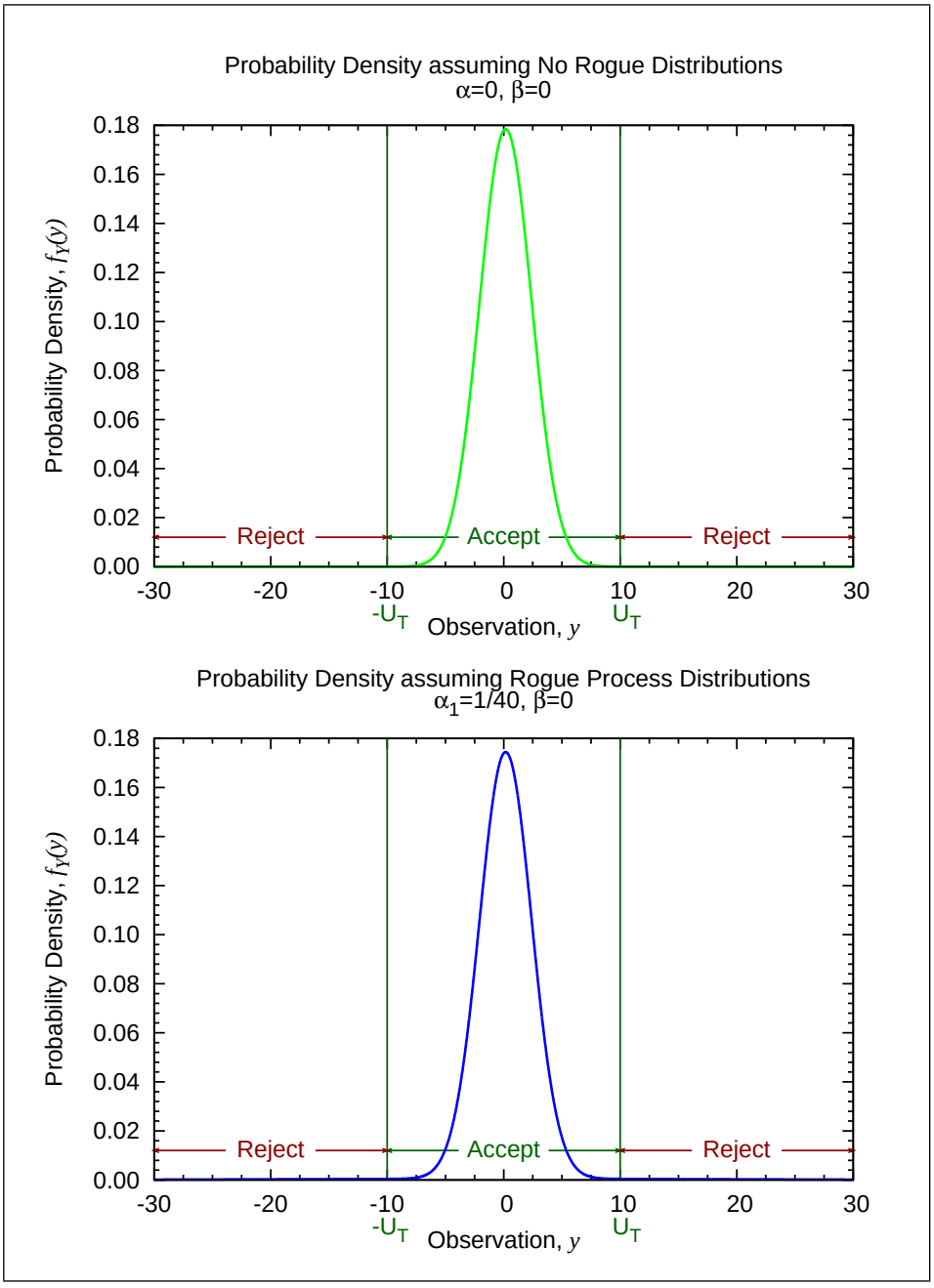

Figure 4.2: Graph of Observations Y for the case where no rogue distributions are present and the case where rogue process distributions are present. This graph assumes normal distributions and mixtures of normal distributions. This graph uses a linear scale on the ordinate axis, see Figure 4.1 for a logarithmic scale.

Firstly, sampling with outlier rejection tends to eliminate the influence of the rogue distributions, because the rogue distributions are discarded as outliers. Secondly, Figure 4.2 shows the density function of the observations Y with and without rogue distributions on a linear scale. Visually, the obvious rogue distributions from the logarithmically scaled graph in Figure 4.1 disappear when the linear scaling in Figure 4.2 is used. Lastly, in the industrial plants visited in the development of this thesis, the accounting departments had noticed the discrepancy between the engineering estimates and collected history. As such, previous production history was used to predict waste in a new manufacturing cell, and not engineering assumptions based on assuming a simple normal distribution. Unfortunately, previous experience does not consistently apply when new kinds of automated cells are being developed. Rogue distributions are only noticeable and quantifiable when large sample sizes, no outlier rejection, and nonlinear graph scaling are used.

The new runtime profiler described in Chapter 5 made it inexpensive to gather the large numbers of gauge data points required to understand the rogue distributions. Additionally, it was able to monitor the responses of the software algorithms to these rogue distributions. This represented the new source of information that motivated the development of this thesis.

Capturing the responses of the software algorithms to the rogue distributions was important. Ordinarily, having a rogue process event rate of $\Pr(|Y| > U_T) \approx 0.015$ would not result in large numbers of parts being quarantined as potential scrap. Initially no one on the industrial projects either knew about or appreciated the significance of the rogue events. As such, the software inside the cell was not designed to handle them. The initial software algorithms chosen to control the cell caused additional rogue events and responded poorly to the rogue events that existed. This effect is discussed in the next section in the context of a feedback loop.

4.2 Effects of Feedback

4.2.1 Feedback Loop Analysis

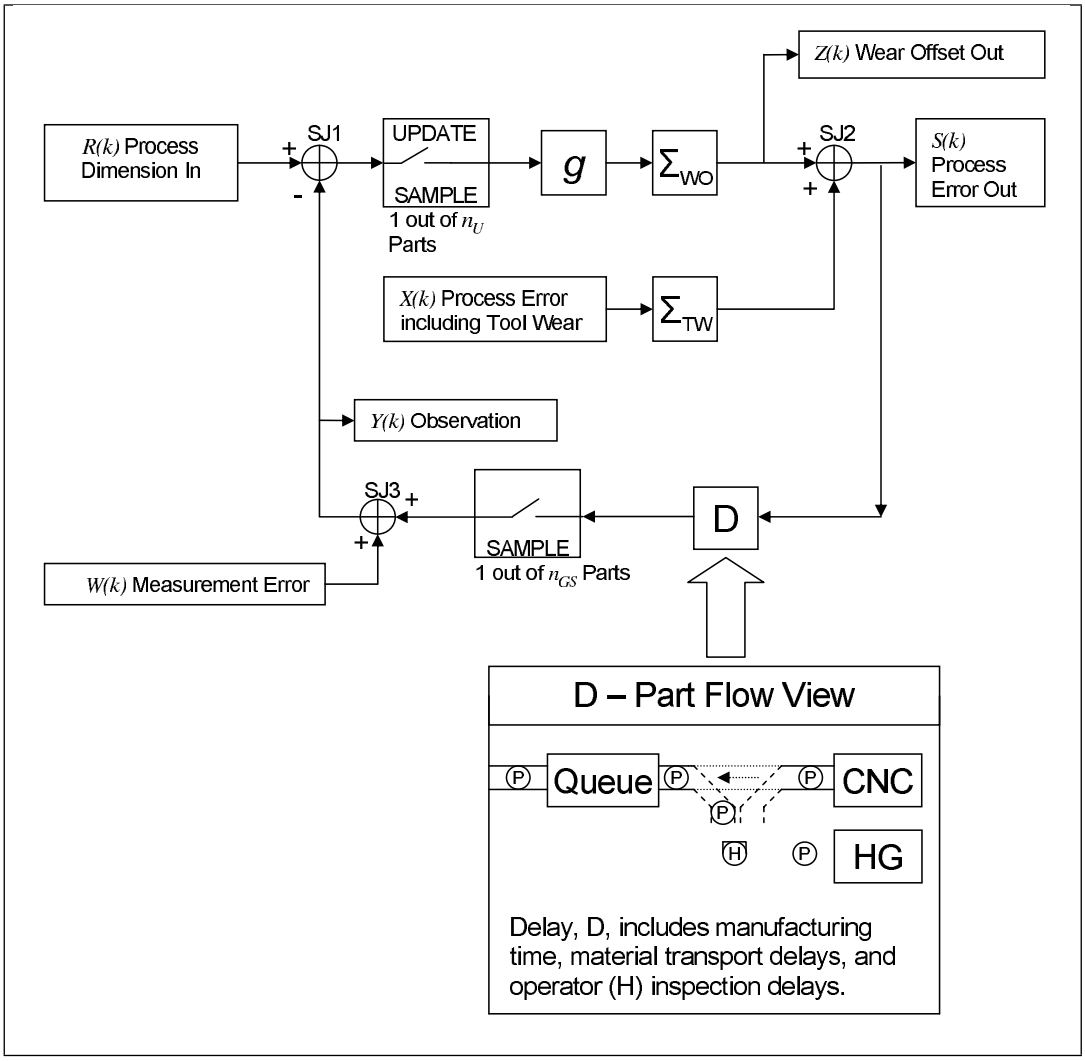

Figure 4.3: Block diagram of the feedback loop from the industrial application.

The block diagram of the feedback loop from the industrial application is shown in Figure 4.3. The index $k \geq 0$ represents the number of update samples since system startup at $k = 0$. Proceeding around the feedback loop

starting in the top left corner:

$R(k)$ is the input signal to the feedback loop. The feedback analysis assumes $R(k) = 0$, as one of the goals of this feedback loop is to maintain either small or zero error in response to the main distribution and as many rogue distributions as possible.

Summing Junction SJ1 The first summing junction subtracts the observation $Y(k)$ from $R(k)$ and forwards the difference to the update sampling block.

Update Sample ensures that every n_U parts the difference is fed to the gain block.

Gain Block, g has a gain of g. As explained in §4.2.2-§4.2.5, the recommended gain value is slightly less than one.

Sum Block Σ_{WO} the sum block sums the total of the updates from the beginning of the analysis to the current sample.

$Z(k)$ is the current output of the sum block Σ_{WO}.

Summing Junction SJ2 adds the output of Σ_{TW} to $Z(k)$. Σ_{TW} comes from $X(k)$ which is described next.

$X(k)$ represents the tool wear and process noise in the process. Mostly, tool wear is a slow incremental process in which the tool abrades as it is worn down by the metal cutting process. However, tool wear is not only a single sided process. Material can be deposited on the tool in the form of a built up edge. Also, the tool can be replaced, which leads to sudden changes that will be modeled as rogue events. When the tool is replaced, the operator is instructed to aim for zero error, however the replacement has a large standard deviation.

Sum Block Σ_{TW} represents the aggregation of the tool wear. This block is present because it models the constantly increasing process of tool wear, which continues until the tool is replaced.

Process (Part) Error $S(k)$ is the sum of the aggregate tool wear and $Z(k)$. This is the dimension that the end-user will observe.

D is a variable length delay and reordering process. Ideally, $D = 1$ update-sample. However, in practice human intervention and a complex material handling system is present. Chapter 8 describes the material handling algorithms.

Sample every 1 in n_{GS} parts are sampled for inspection. As such, n_U must be a multiple of n_{GS}. This block is used to considerable advantage in Chapters 7 and 8.

Summing Junction SJ3 when a part is inspected, gauge error is present. This block adds the gauge error $W(k)$ to the process error $S(k - D)$.

$W(k)$ represents the gauge error in the inspection process.

$Y(k)$ is the observation. This is the as measured sum $W(k)+S(k-D)$, where the D is present because of delay term D in the loop.

Summing Junction SJ1 the loop completes at the first summing junction where $R(k) - Y(k)$ is fed to the update sample block.

Conventionally, every feedback loop has samples numbered from 0 to k. The process error $X(k)$ of any given sample $k \geq 0$ is modeled with the following mixture distribution

$$X(k) = (1 - n_U\alpha)\mathcal{P}_1(n_U\mu_1, n_U\sigma_1^2) + \alpha\mathcal{R}_2(0, \sigma_2^2) \tag{4.8}$$

where

n_U is the update sample rate.

α is the average rate at which process rogue events occur per part. The feedback loop updates every n_U parts, as such the average rate of rogue events per update is $n_U\alpha$.

μ_1 is the long-term average rate of tool wear per part.

σ_1^2 is the variance in the tool wear per part. The variance per update-sample is a function of n_U because over short periods of time the rate of tool wear is not constant. Specifically, under certain process conditions, including when the cutting tool is new, wear occurs faster than average.

σ_2^2 is the variance of the rogue process distribution. It is assumed that the primary cause of rogue process events is tool replacement. The operator aims for a zero average mean in his tool replacement activities; however, the standard deviation is large $\sigma_2 >> \sigma_1$.

$\mathcal{P}_1$ is the process distribution.

$\mathcal{R}_2$ is the rogue process distribution.

4.2.2 Analysis Approach

From a practical point of view, this system works best when the gain is $g \approx 1$ and a fixed loop delay of $D = 1$. §4.2.3 shows that the optimal gain value is $g = 1$ for the specific case where the loop delay is $D = 1$. Certain standard gauge inspection procedures exist have the effect of creating a $D = 2$ system with a fixed $D = 2$ delay for an extended period of time. It is vital that the feedback loop remain stable for the specific cases when an operator attempts these procedures. As explained in §4.2.5, loop stability can still be maintained if the gain $g < 1$. If a gain g is chosen in the range $g = 0.8$ to $g = 0.95$, then the feedback response closely approximates the

response of a linear system with a fixed $D = 1$ delay and $g \approx 1$, while remaining stable when $D = 2$.

For $g \approx 1$, §4.2.5 shows that for fixed $D \geq 3$, the system is unstable. This result creates a challenge, as D is a random number based on operator intervention, and it could be any value. Thus the effect of the random delay is to make the system nonlinear, and potentially stable or unstable, based on the impact of operator intervention on D.

The practical solution to this problem is to model the industrial system as a linear feedback loop where $g \approx 1$ and $D = 1$, and to describe the situations where the model does not apply as rogue events. This is the approach adopted in §4.2.7, and it works for these three reasons:

1. The operators are scored on the scrap generated in the manufacturing cell. As such, the operators have an incentive to try to operate the cell such that $\Pr(D = 1)$ is high, and a minimum amount of scrap is generated.

2. Normally, only one or two parts are removed from the production line for manual inspection. This means $D \in \{1, 2\}$.

3. The line is not permitted to operate indefinitely with an unstable feedback loop. Whenever excessive rejects occur, the production line is automatically stopped.

The line is set up such that the operators rapidly learn how to operate with feedback in place, and to minimize the number of events where $D > 1$.

Some consideration was made as to whether this system is stationary and/or wide sense stationary. [1] The operator interventions are strongly correlated to quality problems, as often when quality problems are occurring,

[1] A stationary process has a joint probability distribution that is unaffected by time (and formally space).

the operator will be attempting to fix a problem on the production line. Often, there is some intelligence in these actions, even if the operator makes mistakes in an effort to test a theory. For instance, if an operator performs an action that does not have the intended effect, then they often undo the action and then try something else. Assuming a stationary process would imply that the operators next action would be independent of the operators previous actions, and no intelligent thought was occurring. As such, it is difficult to justify the stationary assumption, which is helpful in showing some stronger results regarding loop stability.

§4.2.7 uses the approach where the quality problems are modeled as specific types of rogue events. When these rogue events are not occurring, the system can be assumed to be stationary and to have a fixed $D = 1$ loop delay. This is the common case that is analyzed in the next section.

4.2.3 Linear Analysis with Fixed One-Sample Delay

In terms of update-samples, with $D = 1$ update-sample the difference equations for this feedback loop are

$$Z(k) = C_1 - g \sum_{i=0}^{k} Y(i) \tag{4.9}$$

$$S(k) = C_2 + Z(k) + \sum_{i=0}^{k} X(i) \tag{4.10}$$

$$Y(k) = W(k) + S(k-1) \quad \text{for } k > 0 \tag{4.11}$$

$$Y(0) = W(0) + C_1 + C_2 \tag{4.12}$$

where

g is the gain of the gain-block.

k designates the k-th update-sample since system start.

C_1 and C_2 are random variables that reflect the initial state of the system.

$S(k)$, $R(k)$, $W(k)$, $X(k)$, $Y(k)$, $Z(k)$ are the random variables as described previously.

These simplify to

$$Y(k) = W(k) + \sum_{i=0}^{k-1} X(i) - g \sum_{i=0}^{k-1} Y(i) + C_1 + C_2 \tag{4.13}$$

Subtracting $Y(k-1)$ from $Y(k)$ yields

$$Y(k) - Y(k-1) = W(k) - W(k-1) + X(k-1) - gY(k-1) \tag{4.14}$$

Rearranging yields

$$Y(k) = W(k) - W(k-1) + X(k-1) + (1-g)Y(k-1) \tag{4.15}$$

Expanding recursively results in

$$\begin{aligned} Y(k) &= W(k) - g\sum_{i=1}^{k-1}(1-g)^{k-1-i}W(i) - (1-g)^k W(0) + \\ &\quad \sum_{i=0}^{k-1}(1-g)^{k-1-i}X(i) + (1-g)^k Y(0) \end{aligned} \tag{4.16}$$

As $k \to \infty$, it can be seen that for g in the range of $2 > g > 0$ and nonzero $Y(0)$, the output $Y(k)$ remains finite. This can be seen by proving bounded input bounded output (BIBO) stability.

For any g in the range of $2 > g > 0$

$$\sum_{i=0}^{\infty}(1-g)^i = \frac{1}{g} \tag{4.17}$$

As such, for any set of inputs where $|X(k)| < M$ and $|W(k)| < M$ for all $k \geq 0$, and initial starting value $Y(0) < M$, the output will be in the range

$$|Y(k)| \leq 4M + \frac{M}{g} \tag{4.18}$$

for all $k > 0$, $k \to \infty$. Thus for any given set of inputs, and gain g in the range $2 > g > 0$, the output is bounded and the system is BIBO stable. As all inputs must be real numbers and therefore must be finite, the output must also be finite.

In the industrial application, the special case of $g = 1$ is particularly useful. When $g = 1$

$$Y(k) = W(k) - W(k-1) + X(k-1) \tag{4.19}$$

and when $g \approx 1$

$$Y(k) \approx W(k) - W(k-1) + X(k-1) \tag{4.20}$$

within a sufficiently small error band that it does not appear to significantly affect the industrial application.

A reviewer of this thesis noticed that certain distributions, like the normal distributions, have tails that go to infinity. As such, it is theoretically possible that an unlikely event could occur where the input is very large, and the BIBO limit from (4.18) would be effectively useless. If $g = 1$, then (4.19) applies. If a suitably small M and a suitably small $\varepsilon \geq 0$ can be found such that

$$Pr(|W(k)| > M \ \cup \ |W(k-1)| > M \ \cup \ |X(k-1)| > M) < \varepsilon \tag{4.21}$$

then

$$Pr(|Y(k)| > 3M) < \varepsilon \tag{4.22}$$

For a sufficiently small ε, a suitable BIBO bound M can be found for practical purposes. The captured data from the industrial projects showed that extreme outputs were not being accurately modeled by the normal distribution, because of the presence of rogue events. Thus, it is recommended that when estimating appropriate values for M and ε that the appropriate application specific mixture model be used. In the case of this feedback system, this model is shown in (4.66).

In terms of rogue events, (4.19) expresses $Y(k)$ in terms of the measurement error from two consecutive observations, and the process error from one observation. Thus, the impact of any given rogue process or measurement event is contained to the manufacturing parts affected by the two consecutive observations. From an industrial point of view, this is preferable to (4.16), where a single rogue event could affect many observations as the power series decays. The best loop performance will occur when $g = 1$, because the number of parts that need to be quarantined for each rogue event is small. It is also possible to show that $g = 1$ results in the minimum variance in the observations, as is shown in the next section.

4.2.3.1 The Effect of the Gain on the Variance of the Observations

Equation (4.15) states

$$Y(k) = W(k) - W(k-1) + X(k-1) - (1-g)Y(k-1) \tag{4.23}$$

A challenge in computing the variance of $Y(k)$ is that the above definition is recursive, and as such $Y(k-1)$ is a function of $W(k-1)$.

In order to compute the variance of $Y(k)$ without involving recursion, a new independent variable ΔW will be defined. The mean and second moments of ΔW will be computed, and then the mean and second moments of $Y(k)$ will be computed. Solving the resulting equations will result in an expression for the variance of $Y(k)$.

Firstly, to simplify notation, define

$$X = X(k-1) \tag{4.24}$$
$$W = W(k) \tag{4.25}$$
$$W_P = W(k-1) \tag{4.26}$$
$$Y = Y(k) \tag{4.27}$$
$$Y_P = Y(k-1) \tag{4.28}$$

Assume, for the purposes of this section, that the input error sources X, W, and W_P are stationary and independent. Assume Y and hence Y_P are stationary. Assume X and Y_P are independent.

Define the change ΔW in the measurement error as

$$\Delta W = W - W_P \tag{4.29}$$

These definitions simplify (4.23) to

$$Y = \Delta W + X - (1-g)Y_P \tag{4.30}$$

Assume W is stationary, and that W and W_P are independent. As such, $E(W) = E(W_P)$. The mean $E(\Delta W)$ is

$$\mathrm{E}(\Delta W) = \mathrm{E}(W - W_P) \tag{4.31}$$
$$= \mathrm{E}(W) - \mathrm{E}(W_P) \tag{4.32}$$
$$= 0 \tag{4.33}$$

Similarly, the second moment $\mathrm{E}(\Delta W^2)$ is

$$\mathrm{E}(\Delta W^2) = \mathrm{E}\left[(W - W_P)^2\right] \tag{4.34}$$
$$= \mathrm{E}\left(W^2\right) - 2\mathrm{E}\left(W\,W_P\right) + \mathrm{E}\left(W_P^2\right) \tag{4.35}$$

Successive samples of W are stationary and independent. As such

$$\mathrm{E}(W^2) = \mathrm{E}(W_P^2) \tag{4.36}$$

and

$$\begin{aligned} \mathrm{E}(W\,W_P) &= \mathrm{E}(W)E(W_P) && (4.37)\\ &= \mathrm{E}^2(W) && (4.38) \end{aligned}$$

Substituting into (4.35) yields

$$\begin{aligned} \mathrm{E}(\Delta W^2) &= 2\mathrm{E}(W^2) - 2\mathrm{E}^2(W) && (4.39)\\ &= 2\mathrm{Var}(W) && (4.40) \end{aligned}$$

Substituting (4.30) into the expression for the mean $E(Y)$ yields

$$\begin{aligned} \mathrm{E}(Y) &= \mathrm{E}[\Delta W + X + (1-g)Y_P] && (4.41)\\ &= \mathrm{E}(\Delta W) + \mathrm{E}(X) + (1-g)\mathrm{E}(Y_P) && (4.42) \end{aligned}$$

Y is stationary, and as such $E(Y) = E(Y_P)$. Substituting (4.33) yields

$$\mathrm{E}(Y) = \mathrm{E}(X) + (1-g)\mathrm{E}(Y) \tag{4.43}$$

Collecting like terms and simplifying

$$\mathrm{E}(Y) = \frac{\mathrm{E}(X)}{g} \tag{4.44}$$

Substituting (4.30) into the expression for the second moment $\mathrm{E}(Y^2)$

yields

$$\begin{aligned} \mathrm{E}(Y^2) &= \mathrm{E}\left\{[\Delta W + X + (1-g)Y_P]^2\right\} && (4.45) \\ &= \mathrm{E}(\Delta W^2) + \mathrm{E}(X^2) + (1-g)^2\mathrm{E}(Y_P^2) + 2\mathrm{E}(\Delta W\, X) + \\ &\quad 2(1-g)\mathrm{E}(\Delta W\, Y_P) + 2(1-g)\mathrm{E}(X\, Y_P) && (4.46) \end{aligned}$$

The random variables ΔW and X are independent. As such

$$\begin{aligned} \mathrm{E}\left(\Delta W\, X\right) &= \mathrm{E}(\Delta W)\mathrm{E}(X) && (4.47) \\ &= 0 && (4.48) \end{aligned}$$

The random variables ΔW and Y_P are also independent. As such

$$\begin{aligned} \mathrm{E}\left(\Delta W\, Y_P\right) &= \mathrm{E}(\Delta W)\mathrm{E}(Y_P) && (4.49) \\ &= 0 && (4.50) \end{aligned}$$

The random variables X and Y_P are independent. Thus

$$\mathrm{E}(X\, Y_P) = \mathrm{E}(X)\mathrm{E}(Y_P) \quad (4.51)$$

Y is stationary, and as such $E(Y_P) = E(Y)$. Substituting (4.44) yields

$$\mathrm{E}(X\, Y_P) = \frac{\mathrm{E}^2(X)}{g} \quad (4.52)$$

Substituting (4.40), (4.48), (4.50), (4.52) and $E(Y^2) = E(Y_P^2)$ into (4.46) yields

$$\mathrm{E}(Y^2) = 2\mathrm{Var}(W) + \mathrm{E}(X^2) + (1-g)^2\mathrm{E}(Y^2) + 2(1-g)\frac{\mathrm{E}^2(X)}{g} \quad (4.53)$$

Collecting like terms

$$\mathrm{E}(Y^2) = \frac{2\mathrm{Var}(W) + \mathrm{E}(X^2)}{g(2-g)} + \frac{2(1-g)\mathrm{E}^2(X)}{g^2(2-g)} \quad (4.54)$$

The variance $\mathrm{Var}(Y)$ is given by

$$\begin{aligned}
\mathrm{Var}(Y) &= \mathrm{E}(Y^2) - \mathrm{E}^2(Y) && (4.55)\\
&= \frac{2\mathrm{Var}(W) + \mathrm{E}(X^2)}{g(2-g)} + \frac{2(1-g)\mathrm{E}^2(X)}{g^2(2-g)} - \frac{\mathrm{E}^2(X)}{g^2} && (4.56)\\
&= \frac{2\mathrm{Var}(W) + \mathrm{E}(X^2) - \mathrm{E}^2(X)}{g(2-g)} && (4.57)\\
&= \frac{2\mathrm{Var}(W) + \mathrm{Var}(X)}{g(2-g)} && (4.58)
\end{aligned}$$

Thus, minimum variance is attained when $g = 1$. When $g = 1$, (4.58) becomes

$$\mathrm{Var}(Y) = 2\mathrm{Var}(W) + \mathrm{Var}(X) \tag{4.59}$$

To a certain extent, this result is expected as (4.19) is somewhat similar to the equation for a deadbeat controller, and Ordys *et al.* (2007, p. 85) mentions that a minimum variance controller is the stochastic equivalent of a deadbeat controller.

Both Ordys *et al.* (2007, p. 85) and Sinha (1988, p. 278) note that deadbeat and minimum variance controllers are highly sensitive to the accuracy of the pole and zero cancellations. The new and interesting result for this application is that the quadratic $g(2-g)$ from (4.58) does not vary significantly in the region where $g \approx 1$. Thus, in the case of this particular system, for this particular plant with a fixed $D = 1$ delay, having $g \approx 1$ yields results that are sufficiently close to the optimum for practical purposes.

4.2.4 Effect of the Minimum Variance Assumption on the Stochastic Model

Historically, a great deal of effort has been expended on deploying feedback systems where part history is indicative of next part performance, and almost every modern production processes now includes one or more feedback

controllers. The concept of a deadbeat controller, or a minimum variance controller, is that all error originating from part history will be compensated for on the next feedback adjustment for this system, and for more sophisticated systems, in the minimum number of feedback adjustments.

Formally, the system from Figure 4.3 is not a deadbeat / minimum variance controller, because the output $S(k)$ can have an uncompensated mean error. Specifically, if the mean of $W(k)$ drifts, the mean of $S(k)$ will drift also. However, from a practical point of view, it is easy to detect and compensate for any drift in $S(k)$ through periodic SPC inspections of the outgoing product. These are performed as a matter of routine procedure. Thus, for modeling purposes, this system shares almost all of the important characteristics of a deadbeat / minimum variance controller.

In the industrial applications, it was noticed that doing time-series correlations over the part history was no longer a useful activity, because so many feedback loops had already been deployed on the production line. When these feedback loops yield a sufficiently good approximation of a minimum variance controller, then it can be assumed that the observation is a random variable, uninfluenced by previous part history.

Minimum variance controllers are known for their sensitivity to having an incorrect model. A rogue event, like an incorrect measurement or a problem in the part production process, will break the minimum variance model. Unfortunately, the initial occurrence of many of these rogue events cannot be predicted by analyzing previous history. They correspond to abrupt process changes like operator interventions or cutting tools failing catastrophically and then being replaced. Thus, room for optimization exists in tuning the systems to behave appropriately in response to rogue events.

4.2.4.1 Optimizing Minimum Variance Control

A minimum variance or deadbeat controller gives optimal process response

if the observation $Y(k)$ is accurate. Rogue events affect the accuracy of $Y(k)$. Consider four cases, each relating to one of the four main combined events E_{13} through E_{24}.

E_{13} corresponds to the main distribution. From (4.59), the main distribution is relatively sensitive to the standard deviation of the measurement distribution. As explained in Chapter 7, optimizing the feedback response to the main distribution involves reducing $Var(Y)$ by reducing $Var(W)$ through averaging several consecutive observations and reducing the number of loop-updates per observation.

E_{23} corresponds to the process rogue distribution. Generally, the goal of the controller is to track process changes. In the application from Chapter 7, several causes for sudden process changes exist, and it is worthwhile to track significant process changes.

E_{14} corresponds to the measurement rogue distribution. Assuming the standard deviation σ_4 of the rogue measurement distribution is large relative to the standard deviation of the main process distribution σ_1, the safest response is usually to ignore the measurement, and not do a loop update.

E_{24} is the combined rogue distribution. Combined rogue distributions are significant in that their presence indicates a loop update needs to be performed, even though the value of the loop update may not be obvious (because of the large measurement error.) This will inevitably mean that another (valid) observation must be acquired before the process can be properly corrected. In practice, this means either a missed loop update, or an incorrect loop update.

Combined rogue distributions are relatively rare, are often difficult to distinguish from the other events (E_{13}, E_{23}, or E_{14}), and inevitably, the feedback loop will need a minimum of another observation before the process can be properly corrected. As such, if the system can tolerate running in an unanticipated state for the duration of one loop iteration (or one observation), then it may be desirable to proceed as if one of the other events occurred. The feedback loop will automatically correct itself when the first valid observation arrives.

It is commercially useful to be able to distinguish between E_{13}, E_{23}, and E_{14}, for the case of the feedback loop. Given a single observation, and a model of the rogue distributions, it is possible to compute the conditional probabilities $Pr(E_{13}|Y(k)$, $Pr(E_{23}|Y(k)$, and $Pr(E_{14}|Y(k)$. Assume the most likely event for any given value of $Y(k)$ is E_{MAX}.

For a feedback loop, the consequences of misidentification are usually constant - the loop will correct itself on the next iteration. Assuming the consequences for a misidentification are constant, for any given value of $Y(k)$, an on-line control system can assume event E_{MAX} is occurring, and proceed accordingly.

With more observations, it is possible to better identify if a particular type of rogue variation is occurring. However, in a feedback loop application, additional observations often result in costly delays. In particular, in certain feedback applications, delay may be as costly as an incorrect adjustment. Thus, it is better to take a chance that may result in successfully correcting the process, than to consistently wait for better data (and put up with the resulting costs.)

4.2.4.2 Determining Conditional Probabilities

With automated data logging, it is possible to capture the values $Y(k)$ and

$Z(k)$ for use in off-line data analysis. The effect of tool wear $X(k)$ is cumulative, as is shown in Figure 4.3, where $X(k)$ is summed in Sum Block Σ_{TW}. Thus, assuming one sample delay, $Y(k) + Z(k-1)$ yields an estimate of the cumulative tool wear.

When the main event E_{13} occurs the sum $Y(k)+Z(k-1)$ is approximately constant, and influenced only by the normal tool wear and measurement error.

If rapid tool wear occurs, or a tool is replaced, then the sum $Y(k)+Z(k-1)$ will suddenly change, and this will be repeated across many parts. This is a clear indication that the rogue event E_{23} occurred, at the point where this sudden change was first noticed.

If the sum $Y(k) + Z(k-1)$ will suddenly changes, and then returns to an approximation of its previous value on the next part, then a rogue measurement event E_{14} occurred. From (4.20), an error in $W(k)$ will affect two consecutive observations, in opposite directions, from the $+W(k)$ and $-W(k-1)$ terms. Thus, two consecutive aberrant observations in $Y(k)$ is also a clear indicator of a rogue measurement event E_{14}.

If the sum $Y(k) + Z(k-1)$ suddenly changes and then settles on a new value that is different than both the next $Y(k)+Z(k-1)$ sum and the previous $Y(k-1) + Z(k)$ sum, then this is a clear indication that either a combined rogue event E_{24} occurred or the process is experiencing two rogue process events E_{23} in a row.

A computer can examine the collected data from the cell and quickly determine how often E_{13}, E_{23}, and E_{14} occur, and this information is sufficient estimate $Pr(E_{13}|Y(k)$, $Pr(E_{23}|Y(k)$, and $Pr(E_{14}|Y(k)$. The system can then be optimized as described previously in Section 4.2.4.1.

It is possible to identify more events in the off-line information than the controller can effectively respond to. For instance, it is possible to identify

instances where either the combined rogue event E_{24} occurred or two consecutive rogue process events E_{23} occurred. However, unless these events are tied with with sufficient frequency at specific values of $Y(k)$, then they do not affect the determination of the most frequently occurring event, and hence the on-line controller response.

If the captured data shows patterns of rogue measurement events, particularly if they correlate to unusual system timings, then it indicates the delay D in the cell has changed. This led to the discovery that it is important to ensure the model includes events for cases where $D = 2$, as discussed in the next section.

4.2.5 z-Transform Analysis

The feedback loop must be stable for the case where a fixed delay $D = 2$ applies. As such, the z-transform of this system will be computed to determine loop stability when D is any fixed number, $D \geq 1$. This thesis uses capital letters for random variables including S, X, W and Y. As such, the z-transform of the corresponding variables is denoted with the calligraphic font $\mathcal{S}$, $\mathcal{X}$, $\mathcal{W}$ and $\mathcal{Y}$, respectively.

For $D \geq 1$, the z-transform equations are

$$\mathcal{S} = \frac{-gz\mathcal{Y}}{z-1} + \frac{z\mathcal{X}}{z-1} \tag{4.60}$$

$$\mathcal{Y} = \mathcal{W} + z^{-D}\mathcal{S} \tag{4.61}$$

where

g is the z-transform of a g gain block.

$\mathcal{S}$ is the z-transform of S.

$\mathcal{W}$ is the transform of W.

$\mathcal{X}$ is the transform of X.

$\mathcal{Y}$ is the transform of Y.

z is the z-transform variable.

z^{-D} is the z-transform of a D-delay block.

$\frac{z}{z-1}$ is the z-transform of a sum block.

This system of equations simplifies to

$$\mathcal{Y} = \mathcal{W} - \frac{gz^{1-D}\mathcal{Y}}{z-1} + \frac{z^{1-D}\mathcal{X}}{z-1} \tag{4.62}$$

$$\mathcal{Y} = \frac{(z-1)\mathcal{W} + z^{1-D}\mathcal{X}}{z-1+gz^{1-D}} \tag{4.63}$$

For a fixed delay of $D = 1$, the system is asymptotically stable in the range $2 > g > 0$, which matches the result from (4.16). For a fixed delay of $D = 2$, the system is asymptotically stable in the range $1 > g > 0$. For fixed delays $D \geq 3$, the system is unstable with $g \approx 1$, and small g values are needed maintain system stability.

4.2.6 Mixture Model for Random Delays

The random variable D can be changed both by direct human intervention in the cell, and by the automated material handling algorithms. Chapter 8 examines several different material handling algorithms, and these algorithms also impact the distribution of D.

Sometimes, when a small change in D occurs, the effect is to change a $D = 1$ system into a $D = 2$ system. Loop stability can be maintained in a $D = 2$ system by using a g value slightly less than one. The practical reason for maintaining a $g < 1$, for instance $g = 0.9$ is that if something is going wrong with the feedback loop and $D = 2$, then stability will be maintained. For one of the industrial applications, this was potentially an issue during cell startup, which occurred approximately once per week.

When periods of excess delay happen, it shows up as misleading data in the Y observations, because the measured part does not reflect the results of the latest loop update. This can lead to complex models for W. In the later chapters, W will be estimated from the captured data. For the purposes of this chapter, it is assumed that $g \approx 1$, and

$$\begin{aligned} W(k) \quad \stackrel{D}{=} \quad & (1 - 2\beta_1 - 2\beta_2 - \theta - \varepsilon)\mathcal{P}_3(0, \sigma_3^2) + \theta\mathcal{R}_4(\mu_4, \sigma_4^2) + \\ & \beta_1\mathcal{R}_5(\mu_5, \sigma_5^2) + \beta_1\mathcal{R}_5(-\mu_5, \sigma_5^2) + \\ & \beta_2\mathcal{R}_6(\mu_6, \sigma_6^2) + \beta_2\mathcal{R}_6(-\mu_6, \sigma_6^2) + \\ & \varepsilon\mathcal{R}_7(\mu_7, \sigma_7^2) \end{aligned} \tag{4.64}$$

where

$W(k)$ is the random variable that is used to model the gauge error present in the feedback loop.

$\mathcal{P}_3$ is the measurement distribution.

$\mathcal{R}_5$ is the rogue measurement distribution representing gauge error due to the wrong part being measured once.

$\mathcal{R}_6$ is the rogue measurement distribution representing gauge error due to the wrong part being measured twice in a row.

$\mathcal{R}_7$ is the rogue measurement distribution representing a serious gauge error due to the fault in the probing system. It is important that the feedback system not respond to these observations, as machine damage could result. This distribution is covered in §4.2.8.

$\mathcal{R}_4$ is the rogue measurement distribution corresponding to all other sources of error.

$2\beta_1$ is the probability that the feedback loop measures one incorrect reject part,

$2\beta_2$ is the probability that the feedback loop measures two incorrect reject parts consecutively,

ε is used to model an important, but never recorded, failure condition that can cause severe machine damage. Much of the following analysis assumes $\varepsilon = 0$. The $\varepsilon \neq 0$ case is discussed in §4.2.8.

θ is the probability that any other rogue gauge event occurs.

μ_4, μ_5, μ_6, and μ_7 are the means of the corresponding rogue measurement distributions.

σ_3^2 is the variance of the gauge error.

σ_4^2, σ_5^2, σ_6^2, and σ_7^2 are the means of the corresponding rogue measurement distributions.

Two β_1 and β_2 constants were used because the line is not allowed to become unstable indefinitely. Both of the industrial applications stopped the line if they were unable to correct an out-of-specification condition within two loop updates. If feedback results in an improperly adjusted process due to rogue gauge error, then one additional loop update exists to correct the impact of the gauge error. If the process is not returned to specification promptly, then equipment damage may result. As such, if the process continues to generate

reject parts for longer than two loop updates, then the line will be stopped and operator intervention will occur.

4.2.7 Probability that waste occurs in a system with feedback

From (4.19)

$$Y(k) = X(k) + W(k) - W(k-1) \tag{4.65}$$

where each of the associated $Y(k)$, $W(k)$ and $X(k)$ terms are from the associated distributions.

Assuming only one rogue event per observation can occur, Y has the distribution

$$\begin{aligned} Y \quad \stackrel{D}{=} \quad & (1 - n_U\alpha - 4\beta_1 - 4\beta_2 - 2\theta - 2\varepsilon)\mathcal{P}_{13}(n_U\mu_1, n_U\sigma_1^2 + 2\sigma_3^2) + \\ & n_U\alpha\mathcal{R}_{23}(0, \sigma_2^2 + 2\sigma_3^2) + \\ & \theta\mathcal{R}_{14}(\mu_{14}, \sigma_{14}^2) + \\ & \theta\mathcal{R}_{14}(-\mu_{14}, \sigma_{14}^2) + \\ & 2\beta_1\mathcal{R}_{15}(n_U\mu_1 + \mu_5, n_U\sigma_1^2 + \sigma_3^2 + \sigma_5^2) + \\ & 2\beta_1\mathcal{R}_{15}(n_U\mu_1 - \mu_5, n_U\sigma_1^2 + \sigma_3^2 + \sigma_5^2) + \\ & 2\beta_2\mathcal{R}_{16}(n_U\mu_1 + \mu_6, n_U\sigma_1^2 + \sigma_3^2 + \sigma_6^2) + \\ & 2\beta_2\mathcal{R}_{16}(n_U\mu_1 - \mu_6, n_U\sigma_1^2 + \sigma_3^2 + \sigma_6^2) + \\ & \varepsilon\mathcal{R}_{17}(n_U\mu_1 + \mu_7, n_U\sigma_1^2 + \sigma_3^2 + \sigma_7^2) + \\ & \varepsilon\mathcal{R}_{17}(n_U\mu_1 - \mu_7, n_U\sigma_1^2 + \sigma_3^2 + \sigma_7^2) \end{aligned} \tag{4.66}$$

where

$\mathcal{P}_{13}$ is the main distribution.

$\mathcal{R}_{23}$ is the rogue process distribution,

$\mathcal{R}_{15}$ is the rogue measurement distribution due to one reject part.

$\mathcal{R}_{16}$ is the rogue measurement distribution due to two consecutive reject parts.

$\mathcal{R}_{17}$ is the rogue measurement distribution due to serious gauge failure.

$\mathcal{R}_{14}$ is the rogue measurement distribution that models all rogue events interactions not covered above.

In application, $\mathcal{R}_{14}$ corresponds to a series of rogue events with small means and variances that can be recognized by their collective impact, but are not as easy to individually isolate as the other rogue events. Practical experience was that it was often worth responding to these rogue events with feedback. Tool wear is most consistent when the cutting depth is constant, and maintaining a constant cutting depth requires constant loop updates. Effectively, the standard deviation of the next observation increases with the number of parts since the last adjustment n_U. Maintaining a small n_U improves the accuracy of the next observation, even if the current observation has marginal predictive accuracy.

Figure 4.4 shows the density function of the observations Y with (bottom) and without (top) measurement rogue distributions. The top graph has $\beta_1 = \beta_2 = 0$. The lower graph has $\beta_1 = \alpha/8$ and $\beta_2 = \beta_1/2$. Both graphs assume normal distributions, have $n_U = 6$, $U_T = 10$, $\alpha = \frac{1}{40}$, $\theta = \varepsilon = 0$, $\mu_1 = \frac{1}{6}$, $\mu_5 = U_T$, $\mu_6 = 2U_T$, $\sigma_1 = 1$, $\sigma_2 = 20$, $\sigma_3 = 2$, and $\sigma_5 = \sigma_6 = 20$. These values were chosen to be similar to industrial applications.

Figure 4.4 shows graphs with heavier tails than shown in Figure 4.1. These heavier tails result in $\Pr(|Y| > U_T) \approx 0.10$ (top) and $\Pr(|Y| > U_T) \approx 0.18$ (bottom). Reject rates over 0.1 are often unacceptable. Additionally, some cells quarantine more than one part for every rogue event where $|Y| > U_T$ is detected. This would result in the majority of production being quarantined as potential waste.

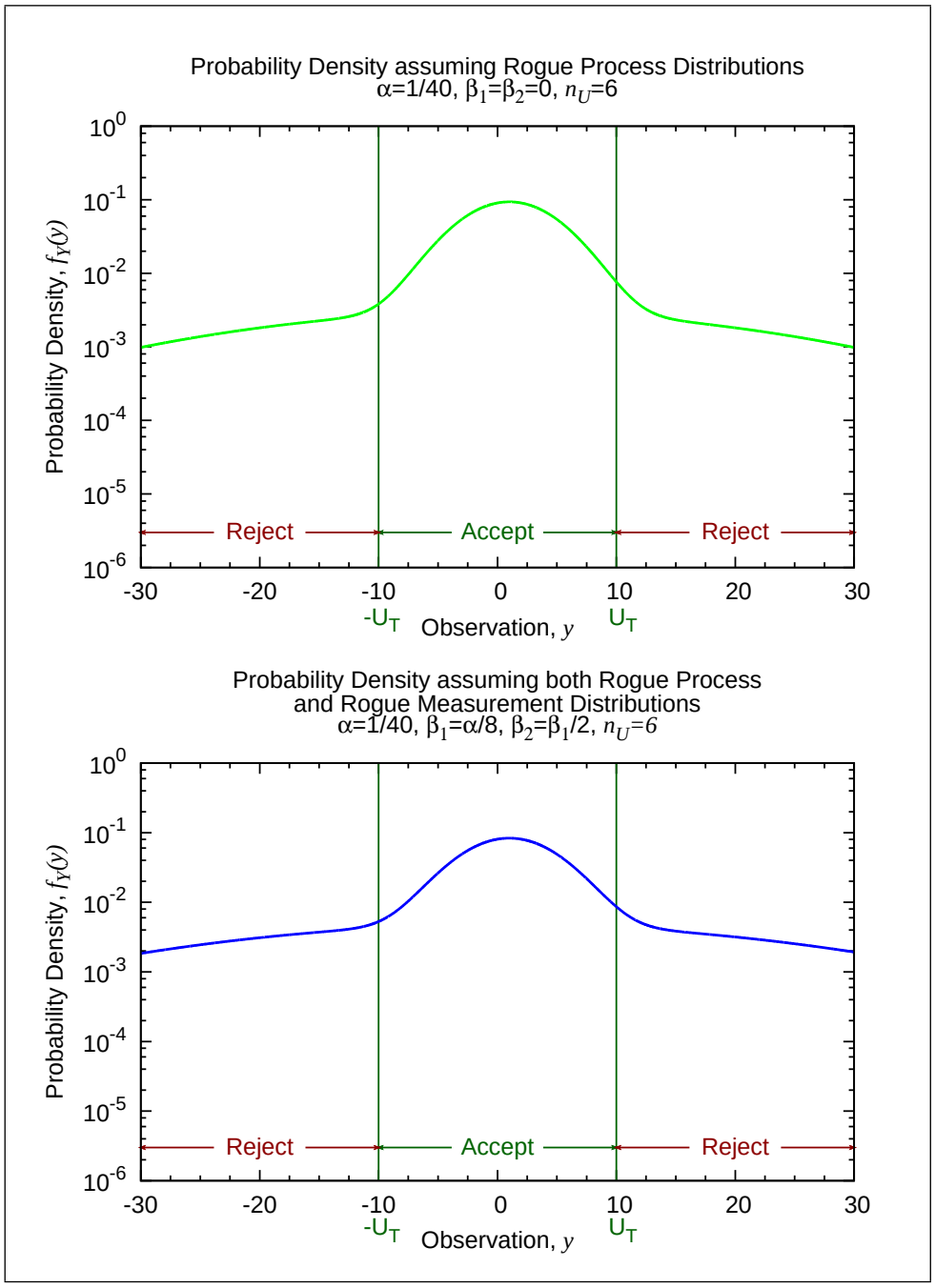

Figure 4.4: Graph of Observations Y for the case where rogue process distributions are present, and the case where both rogue process and rogue measurement distributions are present. This graph uses a logarithmic scale, and considers the case where only one in n_U parts is observed.

The performance of the cell with feedback can be unacceptably poor, and this was observed on the production line in some of the initial deployments. This problem was solved using the algorithms described later in the thesis. The key improvements were:

1. Measuring more parts than the minimum required for loop updates, and using the data to determine if a rogue event was occurring. As suggested in §4.2.4.1, different feedback control strategies can be selected based on the type of rogue event occurring, Chapter 7 shows the resulting feedback algorithms.

2. Material transport systems can increase D, and some of the algorithms that control automatic material transport systems can result in significant sporadic increases in D. The best algorithms shown in Chapter 8 result in consistently small values of D. This makes it possible to achieve small β_1, β_2 and n_U simultaneously.

3. Additional samples from processes experiencing rogue variation allow the exact timing of the rogue variation to be more easily identified. This results in fewer parts being quarantined and significantly reduced waste.

4. If the feedback algorithms work correctly, then there is less need for human intervention in the delay loop, again reducing the variation in D and hence β_1 and β_2.

The θ term is always present because it models all other variation. However, if the large contributors α, β_1 and β_2 are kept in check, then θ is of acceptable magnitude. This leaves ε which is discussed next.

4.2.8 Detecting serious gauge error

Earlier, it was assumed that $\varepsilon = 0$. In the actual cell, very large gauge errors can be generated when the measuring probe on the CMM slows down quickly and misinterprets the slow down as the detection of the part's surface. It is vital that these large gauge errors never be fed back. Assuming $\varepsilon = 0$ and applying (4.66), it is possible to estimate the maximum value of $|Y|$ to any required significance level.

It is common practice for the measuring probe to be slowed down a specific distance from the surface of the part, to allow time for the measuring probe on the CMM to settle. This slow down distance is fully software programmable. As the slow down distance can be set to be sufficiently large, μ_7 and σ_7^2 are under software control.

Additionally, the feedback algorithms can be set such that feedback is never performed when Y is outside of a certain range. Specifically, assume feedback may (or may not) be performed when $|Y|$ is in the range $|Y| \leq U_2$ for some limit U_2, and that feedback is never performed when $|Y| > U_2$. Assuming $|\mu_6| \geq |\mu_{14}|$, $|\mu_6| \geq |\mu_5|$, $\sigma_6^2 \geq \sigma_{14}^2$, $\sigma_6^2 \geq \sigma_5^2$ and $\sigma_6^2 \geq \sigma_3^2$, and if

$$U_2 < |\mu_7| - n_U|\mu_1| - |\mu_6| - k_S\sqrt{n_U\sigma_1^2 + \sigma_6^2 + \sigma_7^2} \tag{4.67}$$

for suitably large k_S, then the likelihood of the feedback system responding to an observation with a very large gauge error can be made arbitrarily small.

In this case, choosing $k_S > 8$ may be advisable because the potential machine damages are very large. A system reliability significantly on the order of $6 \cdot 10^{-16}$ is useful as it offers less risk than a 10^{-9} specification. Specifically, a million dollars of damage at a 10^{-9} probability level represents 0.1 cents per part. A million dollars of damage at $6 \cdot 10^{-16}$ probability represents a negligible cost. As k_S can be set to any value by changing the slow down

distance, using a larger k_S is preferable. The effectiveness of the selected k_S value can be verified through profiling of the captured data.

In the actual system, huge concerns were expressed over the issue of feedback responding to serious gauge error. However, the captured data showed the problem to be a nonissue. Every instance of probe failure was caught inside the measurement program, as described above, before any feedback could occur.

4.2.9 Impact of Rogue Events on Feedback

The crucial concept with feedback as applied to the industrial applications is to realize that the rogue events exist. The impact of these rogue events can be measured with data collection and runtime profiling, even if conventional sampling with outlier rejection may fail to detect them. Certain classes of algorithms, particularly feedback algorithms, can be affected by rogue process and gauge events. Additionally, when feedback is present, material handling systems can cause delays, which can cause additional rogue gauge events.

Once it is realized that significant rogue distributions exist, then it is possible to start selecting and optimizing algorithms in the manner discussed in Chapters 7 and 8. Feedback algorithms are not the only class of algorithms affected by rogue events. In the development of this thesis, it was realized that high-reliability gauging is also affected by rogue events.

4.3 High-Reliability Gauging

The previous section looked at the univariate case where the consequences of misinterpreting observations were relatively small, and the performance

of the algorithms involved could be observed by inspecting the impact of decision making on the performance of the feedback loop. High-reliability gauging poses a different set of challenges. To minimize costs, it is desirable for the gauge to have a large probability of acceptance, P_A where

$$P_A = \Pr(|Y| \leq U_T) \tag{4.68}$$

Subject to the constraint that the outgoing defect rate P_E should be within customer limits P_E^*

$$P_E \leq P_E^* \tag{4.69}$$

where

Y is the observation - the total error observed by the gauge including both process (part) error and gauge error.

U_T is the upper tolerance limit.

P_A is the probability of acceptance, the rate at which the gauge accepts parts.

P_E is the outgoing defect rate, the probability that the end-user may find a defective part sufficiently outside of specification to cause a customer issues.

P_E^* is the upper limit on the outgoing defect rate and is a customer specification.

Many customers accept an outgoing defect rate of $P_E^* = 10^{-9} = 1ppb$, as sufficiently close to zero defects that any difference will not be noticed in a production lifetime of 100 million parts or less.

In Chapter 9, three different algorithms are examined. All of these algorithms were initially sold as being reliable. However, as discussed in Chapter 9, when the performance of the system was measured

1. The first algorithm has a $P_A = 0.9768$ and measured $P_E = 7.1 \cdot 10^{-3}$.
2. The second algorithm has $P_A = 0.9715$ and measured $P_E = 1.5 \cdot 10^{-3}$.
3. The third algorithm has $P_A = 0.9665$ and measured $P_E = 0$ on a 1 million part sample.

The problem being examined in this section is how to develop an estimate of P_E of sufficient accuracy to determine if the algorithm meets the $P_E^* \geq P_E$ specification.

It is fairly obvious from the testing that the first and second algorithms, by themselves, do not meet specification. As such, the customer deployed the first and second algorithms with up to 6 levels of redundancy to reach the P_E^* specification. In theory, redundancy should significantly increase system reliability, as every additional set of independent observations should reduce the P_E of the overall system exponentially. However, if one rogue event can affect all of the observations, then the observations will not be independent, and the system will not be truly redundant. When the captured profile data was analyzed, it became obvious that independence could not be assumed. This motivated the development of a theory that clearly explained the issues in each system.

Based on the runtime profiling information collected on a production sample of over one million parts, the third algorithm was new and was developed as part of this thesis research. However, it was never used in production, because no theory existed that showed it would work. Subsequently, after losing their most profitable customer, the plant was closed. This new theory is needed because of its industrial importance to the Canadian automotive supply base.

4.3.1 True-Attribute Gauge Definition

Many types of gauges exist, and a big focus of gauge design is the measurement and elimination of small errors from the gauging process. For example, see AIAG (2003). The industrial problems being examined in this thesis could only be explained through large deviations in the gauge reading. Particularly perplexing problems were being observed in a class of gauges that in this thesis are referred to as "true-attribute gauges."

The concept behind an attribute gauge is that an attribute is either present or it is not present. Often the gauge observations are in the form of random variables, and it is up to the gauge to determine if the observations imply that the attribute is present. Some systems are designed such that these decisions may be straightforward, and thus a high-reliability gauge can be built. The rogue mixture model was developed to explain and predict outgoing defects in these gauges.

This thesis defines a "true-attribute gauge" as a gauge that meets the following "true-attribute gauge conditions"

$$\Pr(|X| > U_X | E_{13}) \ll P_E^* \tag{4.70}$$

$$\Pr(|Y| \leq U_T | E_{23}) \ll P_E^* \tag{4.71}$$

$$\Pr(|Y| \leq U_T | E_{14}) \ll P_E^* \tag{4.72}$$

$$\tag{4.73}$$

where

X is the inherent error in the process (part) error, and is particular to the part being measured.

U_X is the upper limit on this process error.

Y is the observation - the total error observed by the gauge including both process (part) error and gauge error.

U_T is the upper tolerance limit.

E_{13} is the main event that corresponds to the main distribution.

E_{23} is the rogue process event that corresponds to a rogue process (part) error. This should always be detectable by the gauge.

E_{14} is the rogue gauge event that corresponds to a rogue gauge error (a problem with the reading.)

E_{24} (not shown) is the combined rogue event where both a rogue gauge event and a rogue process event occur simultaneously (in the same observation).

A true-attribute gauge has the property that

$$\Pr(|X| > U_X \cap |Y| \leq U_T \mid E_{13} \cup E_{23} \cup E_{14}) \ll P_E^* \tag{4.74}$$

Within the accuracy specification P_E^*, the gauge should never accept a reject part provided no more than one rogue event occurs. The challenge with true-attribute gauges is to estimate and deliver gauges where $P_E^* > P_E \approx \alpha\beta\lambda_{24}$.

In the industrial plants visited in the development of this thesis, many people were familiar with ineffective gauges. The two causes were

1. The borderline part problem. A part with $|X|$ sufficiently close to U_T may test as either accept A or reject A'. The methods described in the MSA manual by AIAG (2003) are effective in predicting these problems.

2. The rogue event problem. The methods in the MSA manual are ineffective at predicting or detecting this problem, as the methods in the MSA manual are dependent on extrapolations from small samples and the ability to find borderline parts.

Both problems can exist inside the same gauge. However, a true-attribute gauge is only affected by the rogue event problem, as no borderline parts exist. When working with applications with stringent safety, performance, and/or economic targets, it is critical to have methods to properly estimate gauge reliability.

The theory in the following sections applies to all gauges that have observations that can be described with the mixture models presented. True-attribute gauges are the preferred example, because of their industrial importance and the difficulties involved in predicting their performance with standard SPC approaches. Hence, the remainder of this chapter is new work focusing on rogue events as applied to true-attribute gauges.

4.3.2 Basic Analysis of Simple Univariate Gauge

4.3.2.1 Definitions

A univariate gauge makes one observation per part being measured. Assume

$$X \overset{D}{=} (1-\alpha)\,\mathcal{P}_1\left(0,\sigma_1^2\right) + \alpha\mathcal{R}_2\left(\mu_2,\sigma_2^2\right) \tag{4.75}$$

$$W \overset{D}{=} (1-\beta)\,\mathcal{P}_3\left(0,\sigma_3^2\right) + \beta\mathcal{R}_4\left(\mu_4,\sigma_4^2\right) \tag{4.76}$$

$$Y = X + W \tag{4.77}$$

where

X is the process error.

W is the gauge error.

Y is the observation, which reflects the total error (both process and measurement) as measured by the gauge.

$\mathcal{P}_1$ is the process distribution which occurs with event E_1, probability $1-\alpha$ and density function $f_1(x)$.

$\mathcal{R}_2$ is the rogue process distribution which occurs with event E_2, probability α and density function $f_2(x)$.

$\mathcal{P}_3$ is the measurement distribution which occurs with event E_3, probability $1-\beta$ and density function $f_3(w)$.

$\mathcal{R}_4$ is the rogue measurement distribution which occurs with event E_4, probability β and density function $f_4(w)$.

μ_2, μ_4 are the means of the rogue process and rogue measurement distributions, respectively.

σ_i^2, $i \in \{1...4\}$ are the variances of the process, rogue process, measurement, and rogue measurement distributions, respectively.

In application, a gauge observes Y and inferences about X and W must be made indirectly.

The mixture distribution for Y is given by

$$\begin{aligned} Y \quad \stackrel{D}{=} \quad & (1-\alpha)(1-\beta)\mathcal{P}_{13}(0, \sigma_1^2 + \sigma_3^2) + \\ & \alpha(1-\beta)\mathcal{R}_{23}(\mu_2, \sigma_2^2 + \sigma_3^2) + \\ & (1-\alpha)\beta\mathcal{R}_{14}(\mu_4, \sigma_1^2 + \sigma_4^2) + \\ & \alpha\beta\mathcal{R}_{24}(\mu_2 + \mu_4, \sigma_2^2 + \sigma_4^2) \end{aligned} \tag{4.78}$$

where

$\mathcal{P}_{13}$ is the main (rogue-free) distribution, and has density function $f_{13}(y)$ and occurs when the combined event $E_{13} = E_1 \cap E_3$ occurs.

$\mathcal{R}_{23}$ is the process rogue distribution, and has density function $f_{23}(y)$ and occurs when the combined event $E_{23} = E_2 \cap E_3$ occurs.

$\mathcal{R}_{14}$ is the measurement rogue distribution, and has density function $f_{14}(y)$ and occurs when the combined event $E_{14} = E_1 \cap E_4$ occurs.

$\mathcal{R}_{24}$ is the combined rogue distribution, and has density function $f_{24}(y)$ and occurs when the combined event $E_{24} = E_2 \cap E_4$ occurs.

It is well known that with feedback loops, human intervention, and mechanical failures, rogue events can occur in clusters. The goal of high-reliability gauging is to identify the first rogue event in any given cluster. After this first rogue event has been identified, production can be quarantined and the effects of the cluster contained. §6.3 provides an overview of this activity in an industrial application.

As clusters of related events can be quarantined and eliminated from consideration, assume X and W are independent. Assume the goal is to detect the first rogue event or combined rogue event in any given cluster, and as such (4.78) applies.

The next section gives a simple example showing how to apply (4.75) through (4.78).

4.3.3 An Initial Example

Assume the main and rogue distributions are normally distributed, and P_E is given by

$$P_E = \Pr(|X| > U_X \cap |Y| \leq U_T) \tag{4.79}$$

where U_X is a customer-specific upper limit on the process error.

Define γ_{ij} as

$$\begin{aligned}
\gamma_{ij} &= \gamma_{ij}(U_T) && (4.80)\\
&= \Pr(|X| > U_X \cap |Y| \leq U_T | E_{ij}) && (4.81)\\
&= \int_{U_X}^{\infty} f_i(x) \int_{-U_T}^{U_T} f_j(y-x) dy dx + \int_{-\infty}^{-U_X} f_i(x) \int_{-U_T}^{U_T} f_j(y-x) dy dx && (4.82)\\
&= \int_{U_X}^{\infty} f_i(x) \int_{-U_T-x}^{U_T-x} f_j(w) dw dx + \int_{-\infty}^{-U_X} f_i(x) \int_{-U_T-x}^{U_T-x} f_j(w) dw dx && (4.83)
\end{aligned}$$

where $f_1(x)$, $f_2(x)$, $f_3(w)$, and $f_4(w)$ are the density functions of the process $\mathcal{P}_1$, rogue process $\mathcal{R}_2$, measurement $\mathcal{P}_3$, and rogue measurement $\mathcal{R}_4$ distributions, respectively. As this is a mixture variable problem, the general P_E can be decomposed into

$$P_E = (1-\alpha)(1-\beta)\gamma_{13} + \alpha(1-\beta)\gamma_{23} + (1-\alpha)\beta\gamma_{14} + \alpha\beta\gamma_{24} \tag{4.84}$$

If the main and rogue distributions are normally distributed

$$\begin{aligned}
\gamma_{ij}(U_T) &= \int_{U_X}^{\infty} \frac{1}{\sigma_i}\phi\left(\frac{x-\mu_i}{\sigma_i}\right) \int_{-U_T-x}^{U_T-x} \frac{1}{\sigma_j}\phi\left(\frac{w-\mu_j}{\sigma_j}\right) dw dx + \\
&\quad \int_{-\infty}^{-U_X} \frac{1}{\sigma_i}\phi\left(\frac{x-\mu_i}{\sigma_i}\right) \int_{-U_T-x}^{U_T-x} \frac{1}{\sigma_j}\phi\left(\frac{w-\mu_j}{\sigma_j}\right) dw dx && (4.85)\\
\gamma_{ij}(U_T) &= \int_{U_X}^{\infty} \frac{1}{\sigma_i}\phi\left(\frac{x-\mu_i}{\sigma_i}\right) \left[\Phi\left(\frac{U_T-\mu_j-x}{\sigma_j}\right) - \Phi\left(\frac{-U_T-\mu_j-x}{\sigma_j}\right)\right] dx + \\
&\quad \int_{-\infty}^{-U_X} \frac{1}{\sigma_i}\phi\left(\frac{x-\mu_i}{\sigma_i}\right) \left[\Phi\left(\frac{U_T-\mu_j-x}{\sigma_j}\right) - \Phi\left(\frac{-U_T-\mu_j-x}{\sigma_j}\right)\right] dx && (4.86)
\end{aligned}$$

where

$\phi(x)$ is the density function of the standard normal distribution.

$\Phi(x)$ is the distribution function of the standard normal distribution.

$\mu_1 = \mu_3 = 0$ are the means of the process and measurement distributions, respectively, and are assumed to be zero.

μ_2, μ_4, σ_i, $i \in \{1...4\}$ are the means and variances as defined previously.

The means of the process $\mu_1 = 0$ and measurement $\mu_3 = 0$ distributions are assumed to be zero. The remaining standard deviations σ_i and means μ_i are from the relevant process, rogue process, measurement, and rogue measurement distributions.

Similarly, define ρ_{ij} as

$$\begin{aligned} \rho_{ij} &= \rho_{ij}(U_T) && (4.87)\\ &= \Pr(|Y| \leq U_T | E_{ij}) && (4.88)\\ &= \int_{-U_T}^{U_T} f_{ij}(y)dy && (4.89) \end{aligned}$$

where $f_{13}(y)$, $f_{23}(y)$, $f_{14}(y)$, and $f_{24}(y)$ are the density functions of the main $\mathcal{P}_{13}$, process rogue $\mathcal{R}_{23}$, measurement rogue $\mathcal{R}_{14}$, and combined rogue $\mathcal{R}_{24}$ distributions, respectively. The conditional probability ρ_{ij} establishes an upper bound on γ_{ij} via

$$\begin{aligned} \Pr(|Y| \leq U_T | E_{ij}) &\geq \Pr(|X| > U_X \cap |Y| \leq U_T | E_{ij}) && (4.90)\\ \rho_{ij} &\geq \gamma_{ij}(U_T) && (4.91) \end{aligned}$$

Later in the chapter, ρ_{24} is shown to be small in certain multivariate applications. When this applies, (4.91) establishes a useful upper bound on γ_{24}.

The general P_A can be decomposed into

$$P_A = (1-\alpha)(1-\beta)\rho_{13} + \alpha(1-\beta)\rho_{23} + (1-\alpha)\beta\rho_{14} + \alpha\beta\rho_{24} \quad (4.92)$$

If the main and rogue distributions are normally distributed

$$\rho_{ij}(U_T) = \Phi\left(\frac{U_T - \mu_{ij}}{\sigma_{ij}}\right) - \Phi\left(\frac{-U_T - \mu_{ij}}{\sigma_{ij}}\right) \tag{4.93}$$

where $\mu_{13} = 0$ and the remaining means and standard deviations are from the corresponding distributions $\mathcal{P}_{13}$ through $\mathcal{R}_{24}$.

For economic reasons, most production gauges are designed such that α and β are small and $\rho_{13} \approx 1$. This means $P_A \approx 1$. The point of having an industrial "capable process" is that $P_A \approx 1$ and economic returns are maximized.

Define the conditional probability P_C as

$$\begin{aligned} P_C &= \Pr\left(|X| > U_X \mid |Y| \leq U_T\right) && (4.94)\\ &= \frac{\Pr(|X| > U_X \cap |Y| \leq U_T)}{\Pr(|Y| \leq U_T)} && (4.95)\\ &= \frac{P_E}{P_A} && (4.96) \end{aligned}$$

A graph of the conditional probability is shown in Figure 4.5, with $\alpha = \beta = 0.02$, $\sigma_1 = \frac{1}{6}$, $\mu_1 = \mu_2 = \mu_3 = \mu_4 = 0$, $\sigma_2 = 5$, $\sigma_3 = \frac{\sigma_1}{3}$, $\sigma_4 = 10$, and $U_X = 1.3$.

Focusing on just the main distribution, with $U_T = 1$, the process from Figure 4.5 is a $C_{PK} = 2$ process and should be capable. However, because of the presence of the rogue distributions, no value of U_T leads to a $P_C < 10^{-9}$, even if U_T is made very small.

It is often difficult to estimate the process error limit U_X, which represents the point at which an out-of-spec part will cause an outgoing defect. If the MSA manual from AIAG (2003) is examined, an implicit assumption is that if parts from an incapable $C_{PK} = 0$ process are sorted by a gauge with a GR&R score between 10% and 30%, then a number of parts must exist in the resulting distribution out to tolerance limits of about $U_X = 1.3U_T$. As such, it is possible to assume $U_X \geq 1.3U_T$. Many people react with some concern

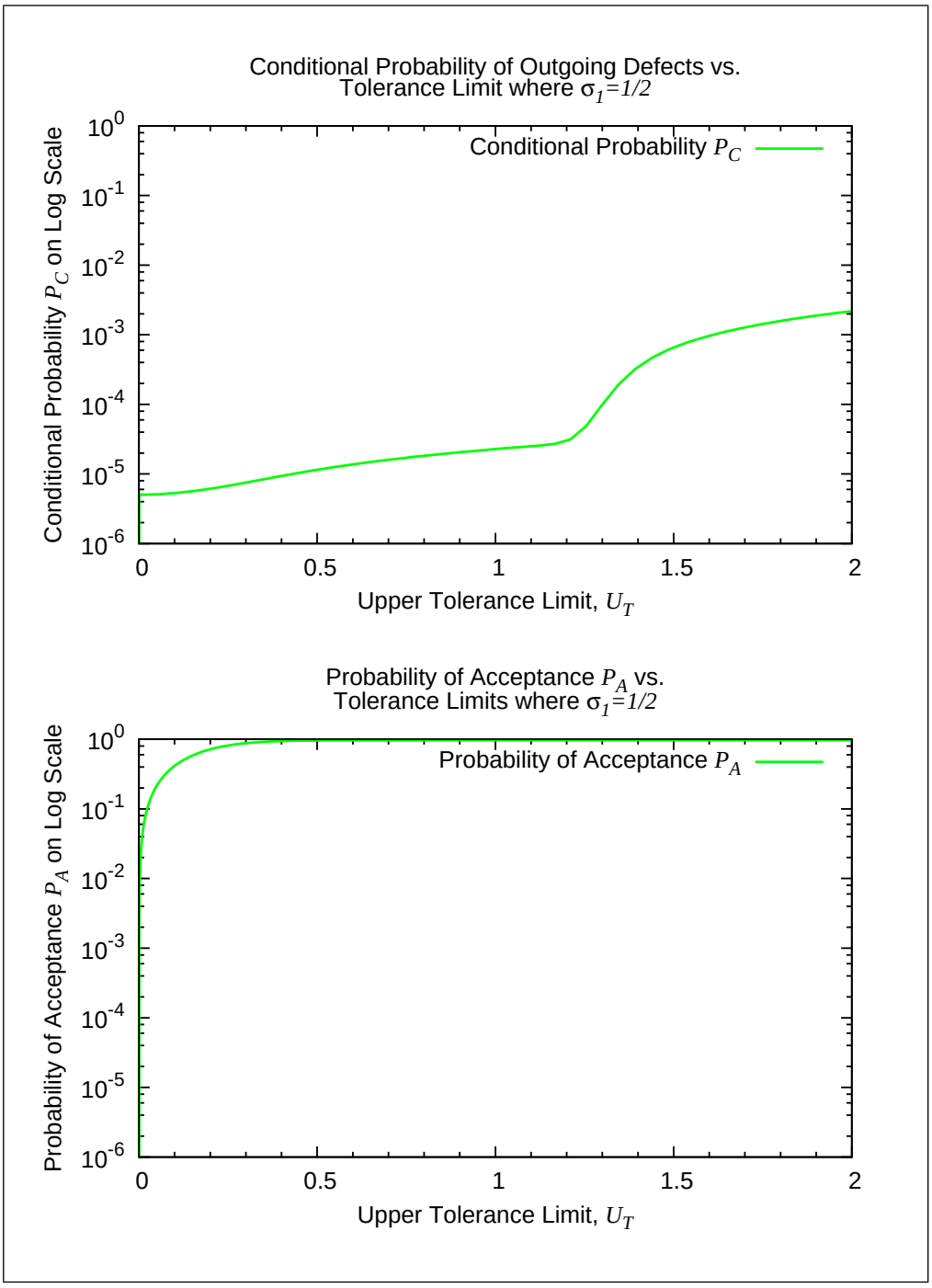

Figure 4.5: Log Scale Graph of the Conditional Probability $P_C = \Pr(|X| > U_X \mid |Y| \leq U_T)$ and the Probability of Acceptance $P_A = \Pr(|Y| \leq U_T)$

as to whether this is a correct assumption, as depending on the application, it may mean that U_T must be reduced and in some systems it is not feasible to reduce U_T. Importantly, reducing U_T on an incapable $C_{PK} = 0$ process will often result in unacceptably large amounts of waste. However, for this example, some value of $U_X > U_T$ must be assumed. Later, methods for true-attribute gauges are presented that remove the need to assume a large U_X when computing the outgoing defect rate.

It is possible to calculate P_C in the limit

$$L = \lim_{U_T \to 0^+} P_C \tag{4.97}$$

$$= \lim_{U_T \to 0^+} \frac{P_E}{P_A} \tag{4.98}$$

$$L = \lim_{U_T \to 0^+} \frac{(1-\alpha)(1-\beta)\gamma_{13}(U_T) + \alpha(1-\beta)\gamma_{23}(U_T) + (1-\alpha)\beta\gamma_{14}(U_T) + \alpha\beta\gamma_{24}(U_T)}{(1-\alpha)(1-\beta)\rho_{13}(U_T) + \alpha(1-\beta)\rho_{23}(U_T) + (1-\alpha)\beta\rho_{14}(U_T) + \alpha\beta\rho_{24}(U_T)} \tag{4.99}$$

In order to employ l'Hopital's rule, it is necessary to find the derivative of $\gamma_{ij}(U_T)$ and $\rho_{ij}(U_T)$ with respect to U_T.

$$\dot{\gamma}_{ij}(0) = \left.\frac{d\gamma_{ij}}{dU_T}\right|_{U_T=0} \tag{4.100}$$

$$= 2\int_{U_X}^{\infty} f_i(x) f_j(-x) dx + 2\int_{-\infty}^{-U_X} f_i(x) f_j(-x) dx \tag{4.101}$$

Assuming the main and rogue distributions are normally distributed

$$\dot{\gamma}_{ij}(0) = 2c_{ij}\left[\Phi\left(\frac{-U_X + u_{ij}}{s_{ij}}\right) + \Phi\left(\frac{-U_X - u_{ij}}{s_{ij}}\right)\right] \tag{4.102}$$

The constants c_{ij}, u_{ij}, and $s_{i,j}$ are

$$c_{ij} = \frac{e^{-\frac{\mu_i+\mu_j}{2\left(\sigma_i^2+\sigma_j^2\right)}}}{\sqrt{2\pi\left(\sigma_i^2+\sigma_j^2\right)}} \tag{4.103}$$

$$s_{ij} = \frac{\sigma_i\sigma_j}{\sqrt{\sigma_i^2+\sigma_j^2}} \tag{4.104}$$

$$u_{ij} = \frac{\sigma_j^2\mu_i - \sigma_i^2\mu_j}{\sigma_i^2+\sigma_j^2} \tag{4.105}$$

Computing the derivative of ρ_{ij}

$$\dot{\rho}_{ij}(0) = \left.\frac{d\rho_{ij}}{dU_T}\right|_{U_T=0} \tag{4.106}$$

$$= \frac{2}{\sigma_{ij}}\phi\left(\frac{-\mu_{ij}}{\sigma_{ij}}\right) \tag{4.107}$$

As the derivatives are constant

$$L = \frac{(1-\alpha)(1-\beta)\dot{\gamma}_{13}(0) + \alpha(1-\beta)\dot{\gamma}_{23}(0) + (1-\alpha)\beta\dot{\gamma}_{14}(0) + \alpha\beta\dot{\gamma}_{24}(0)}{(1-\alpha)(1-\beta)\dot{\rho}_{13}(0) + \alpha(1-\beta)\dot{\rho}_{23}(0) + (1-\alpha)\beta\dot{\rho}_{14}(0) + \alpha\beta\dot{\rho}_{24}(0)} \tag{4.108}$$

For the constants used in the Figure 4.5, this limit evaluates to $5.0423 \cdot 10^{-6}$ which corresponds to the simulations.

This limit expression can be further simplified. In most gauges, the probabilities P_A and P_E are driven by only one of the constituent four terms. In the case of the true-attribute gauge

$$P_A \approx (1-\alpha)(1-\beta)\rho_{13} \tag{4.109}$$

$$P_E \approx \alpha\beta\gamma_{24} \tag{4.110}$$

The main distribution is usually the primary distribution in the pass-zone, and usually the combined rogue distribution causes the majority of outgoing

defects. This reduces the limit to

$$L \approx \frac{\alpha\beta\dot{\gamma}_{24}(0)}{(1-\alpha)(1-\beta)\dot{\rho}_{13}(0)} \tag{4.111}$$

For the constants used in the Figure 4.5, this limit evaluates to $5.0478 \cdot 10^{-6}$, which agrees with the previous result within 0.11%.

In the author's experience, many customers expect that reducing U_T will reduce P_C to tolerable levels. However, the supplier experience was that reducing U_T leads to significantly increased costs, and often the customer complaints continued.

What this limit and Figure 4.5 shows is that for processes experiencing rogue effects, reducing U_T does not yield the required reductions in P_C. The P_C curve has a flat bottom that may be several orders of magnitude higher than P_E^*. As such, reducing U_T does not provide solution to the problem of ensuring the gauge is sufficiently reliable, and alternative methods are needed.

The focus of the remainder of this chapter will be on methods that achieve both a large P_A and $P_E^* > P_E$. As such, the remainder of this chapter assumes

$$P_A \approx 1 \tag{4.112}$$

and

$$P_C = \frac{P_E}{P_A} \approx P_E \tag{4.113}$$

These assumptions are needed, because minimizing U_T and P_A is not a viable solution for the industrial problems of interest.

4.3.4 Estimating how frequently rogue events occur

From a practical point of view, most production lines are busy, noisy, and somewhat dangerous environments, poorly suited to thoughtful and detailed mathematical analysis. As such, it is necessary to have a simple method to estimate α and β. From (4.110), estimates of α, β and γ_{24} are needed to estimate P_E.

To find α and β, first note that

$$\alpha = \Pr(E_2) \tag{4.114}$$

$$\beta = \Pr(E_4) \tag{4.115}$$

as α and β are simply the probabilities of occurrence of rogue process events E_2 and rogue measurement events E_4. Define α' and β' as

$$\alpha' = \Pr(|Y| > U_T \cap E_2) \quad \Rightarrow \quad \alpha' = \alpha \, \Pr(|Y| > U_T \,|\, E_2) \tag{4.116}$$

$$\beta' = \Pr(|Y| > U_T \cap E_4) \quad \Rightarrow \quad \beta' = \beta \, \Pr(|Y| > U_T \,|\, E_4) \tag{4.117}$$

where

α' is the rate of significant rogue process events.[2]

β' the rate of significant rogue measurement events.[3]

U_T is the tolerance limit below which the gauge accepts parts, and above which the gauge rejects parts. For the purposes of this section, it is assumed that the gauge is testing to the part-print tolerance limit U_T.

[2]The prime notation is used because it will be assumed that the conditional probability is $\Pr(|Y| > U_T \,|\, E_2) \approx 1$. As such, $\alpha' \approx \alpha$. Equation (4.116) is not a differential equation.

[3]Similarly, (4.117) is not a differential equation.

For sufficiently large values of σ_2, $\Pr(|Y| > U_T \,|\, E_2) \approx 1$ and $\alpha' \approx \alpha$. Similarly, for sufficiently large values of σ_4, $\beta' \approx \beta$. For many industrial applications, these assumptions will apply with sufficient accuracy to get a meaningful order-of-magnitude estimate of P_E. This is sufficient for computing the sample sizes required for determining the exact value of P_E.

If better estimates of P_E are needed, U_T could be adjusted to an appropriate control limit so these conditions are met. Alternatively, this procedure details a method of estimating the conditional conditional density functions $f_{23}(y)$ and $f_{14}(y)$. Remembering that combined rogue distributions are often infrequent events, α can be estimated by computing an estimate of $\Pr(|Y| > U_T \,|\, E_2)$ and then using (4.116). Similarly, β can be estimated with (4.117). For large values of P_E, the recommended procedure is simply to find approximate values of α and β, and then confirm the estimate by designing an appropriate sample study.

In order to estimate P_E, it is also helpful to have an estimate of γ_{24}. This technique also helps in estimating γ_{24}.

4.3.4.1 Step 1: Initial Sort

Many gauges sort production into two categories - Accept A where $|Y| \leq U_T$ and Reject A' where $|Y| > U_T$. The percentage of parts rejected is often easily calculated, and most automatic gauges compute it as a running total. The percentage of parts rejected is an estimate of the probability of rejection P_R which is related to α' and β' by

$$P_R = \alpha' + \beta' - \alpha'\beta'(1 - \gamma_{24}) \tag{4.118}$$

$$P_R \approx \alpha' + \beta' \tag{4.119}$$

The first step to finding α' and β' is to sort a batch of parts into a reject bin and an accept bin, identify the bins, and record $\alpha' + \beta'$.

The second step is to find β' by performing a second sort on the parts in the reject bin, and noting the gauge readings. The percentage of accept parts on the second sort allows the estimation of gauge errors versus total errors, via the following probability estimate

$$\Pr(|Y| \leq U_T| \text{ once rejected parts}) \approx \Pr(E_4|E_2 \cup E_4) \tag{4.120}$$

$$\Pr(E_4|E_2 \cup E_4) = \frac{\beta'}{\alpha' + \beta' - \alpha'\beta'} \tag{4.121}$$

$$\Pr(E_4|E_2 \cup E_4) \approx \frac{\beta'}{\alpha' + \beta'} \tag{4.122}$$

which implies

$$\Pr(|Y| \leq U_T| \text{ once rejected parts}) \approx \frac{\beta'}{\alpha' + \beta'} \tag{4.123}$$

Assuming a true-attribute gauge, specifically condition (4.71), the sample average provides an estimate of μ_2. Also, the sample histogram yields an approximation of $f_{23}(y)$. For $\sigma_2^2 >> \sigma_3^2$, the variance of the reject parts from the second sort provides an estimate of σ_2^2, and $f_{23}(y)$ approximates $f_2(x)$.

4.3.4.2 Step 2: Second Sort

Careful consideration must be given to whether to do the second sort automatically or manually with a different gauge. The above equation makes the assumption that gauge error in the second test is independent of gauge error in the first test. In many production applications, the number of reject parts is small, and the costs of a second manual test are lower than the costs of removing the automatic gauge from production. Also, it is unlikely that a secondary manual gauge will have the same failure modes as a primary automatic gauge. Thus, with a manual measurement, the gauge error on the first sort operation will be independent of the gauge error on the second sort operation.

Alternatively, the second sort can be done automatically with the same automatic gauge. This creates the possibility that a group of parts exists such that the two gauge errors may not be independent. In some applications, this possibility is small, and the second sort can be done automatically. When this is done, it is possible to get an idea of the shape of the $\mathcal{R}_4$ distribution. Assuming a true-attribute gauge, specifically condition (4.72), the sample average and histogram of the accept parts from a second automatic sort provides an estimate of μ_4 and $f_{14}(y)$. For $\sigma_4^2 >> \sigma_1^2$, the variance of the reject parts from the second sort provides an estimate of σ_4^2, and $f_{14}(y)$ approximates $f_4(w)$.

4.3.4.3 Step 3: Estimating the distributions

The sample variance and histogram of the accept population from the first sort can be used to estimate $\sigma_1^2 + \sigma_3^2$ and the density function $f_{13}(y)$. Further, given estimates of α' and β', the probabilities α and β can be estimated based on estimates of the relevant distributions gathered with runtime profiling. Additionally, as estimates of $f_2(x)$ and $f_4(w)$ have been found, then $f_{24}(y)$ and γ_{24} can be estimated.

A sample study of appropriate size can then be conducted on the accept population of the first sort to confirm the prediction from the analysis. Alternatively, the data from the first sort can be automatically captured, logged, and profiled. If a sufficiently large amount of captured data is available, a secondary algorithm can be used to confirm the accuracy (or inaccuracy) of the original algorithm. This later approach resulted in the captured data set that is analyzed in Chapter 9.

4.3.5 A More Complex Univariate Gauge

In the initial example from §4.3.3, it was impossible to meet the $P_E^* > P_E$

specification with reasonable $P_A \approx 1$. This example represents an attempt to overcome this problem. It is from a practical gauge that is discussed in Chapter 9. The outgoing defects encountered with this gauge were poorly understood by the company that operated it. Gauge performance is explained here in terms of the rogue mixture model.

In the spring gauging application that is discussed in Chapter 9, the critical value for determining spring orientation on a variable rate spring is given by the ratio Y_R of 4 readings $\{Y_1 ... Y_4\}$

$$Y_R = \frac{Y_1 Y_2}{Y_3 Y_4} \tag{4.124}$$

In a correctly-oriented variable rate spring, the spacing between the coils on the lower half of the spring Y_3 and Y_4 should be significantly less than the spacing between the coils on the upper half of the spring Y_1 and Y_2. As such, regardless of the exact details of the spring in question, a spring with $y_R > 1$ is correctly oriented, and a spring with $y_R < 1$ is incorrectly oriented. As discussed in Chapter 9, other process controls on the line were in place, and the orientation gauge only had to detect correct orientation.

The log of each $Y_i, i \in \{1, ...2\}$ is given by

$$\ln Y_i \overset{D}{=} \begin{cases} \mathcal{N}(\mu_4, \sigma_4^2) & \text{with events} \quad E_\alpha \cap E_i \\ \mathcal{N}(\mu_3, \sigma_3^2) & \text{with events} \quad E'_\alpha \cap E_i \\ \mathcal{N}(\mu_2, \sigma_2^2) & \text{with events} \quad E_\alpha \cap E'_i \\ \mathcal{N}(\mu_1, \sigma_1^2) & \text{with events} \quad E'_\alpha \cap E'_i \end{cases} \tag{4.125}$$

and the log of each $Y_i, i \in \{3, ...4\}$ is given by

$$\ln Y_i \overset{D}{=} \begin{cases} \mathcal{N}(\mu_3, \sigma_3^2) & \text{with events} \quad E_\alpha \cap E_i \\ \mathcal{N}(\mu_4, \sigma_4^2) & \text{with events} \quad E'_\alpha \cap E_i \\ \mathcal{N}(\mu_1, \sigma_1^2) & \text{with events} \quad E_\alpha \cap E'_i \\ \mathcal{N}(\mu_2, \sigma_2^2) & \text{with events} \quad E'_\alpha \cap E'_i \end{cases} \tag{4.126}$$

Event E_α corresponds to process error and occurs with probability α. Independent events E_i, $i \in \{1...4\}$ correspond to gauge error and occur with probability β. This means there are 32 different combinations of process and gauge error.

Taking the logarithm of both sides of (4.124), the following expansion results

$$\begin{aligned}
\ln Y_R \overset{D}{=} & (1-\alpha)(1-\beta)^4\mathcal{N}(2\mu_1-2\mu_2,2\sigma_1^2+2\sigma_2^2)+\\
& 2(1-\alpha)(1-\beta)^3\beta\mathcal{N}(\mu_1-2\mu_2+\mu_3,\sigma_1^2+2\sigma_2^2+\sigma_3^2)+\\
& 2(1-\alpha)(1-\beta)^3\beta\mathcal{N}(2\mu_1-\mu_2-\mu_4,2\sigma_1^2+\sigma_2^2+\sigma_4^2)+\\
& 4(1-\alpha)(1-\beta)^2\beta^2\mathcal{N}(\mu_1-\mu_2+\mu_3-\mu_4,\sigma_1^2+\sigma_2^2+\sigma_3^2+\sigma_4^2)+\\
& (1-\alpha)(1-\beta)^2\beta^2\mathcal{N}(-2\mu_2+2\mu_3,2\sigma_2^2+2\sigma_3^2)+\\
& (1-\alpha)(1-\beta)^2\beta^2\mathcal{N}(2\mu_1-2\mu_4,2\sigma_1^2+2\sigma_4^2)+\\
& 2(1-\alpha)(1-\beta)\beta^3\mathcal{N}(\mu_1+\mu_3-2\mu_4,\sigma_1^2+\sigma_3^2+2\sigma_4^2)+\\
& 2(1-\alpha)(1-\beta)\beta^3\mathcal{N}(-\mu_2+2\mu_3-\mu_4,\sigma_2^2+2\sigma_3^2+\sigma_4^2)+\\
& (1-\alpha)\beta^4\mathcal{N}(2\mu_3-2\mu_4,2\sigma_3^2+2\sigma_4^2)+\\
& \alpha(1-\beta)^4\mathcal{N}(-2\mu_1+2\mu_2,2\sigma_1^2+2\sigma_2^2)+\\
& 2\alpha(1-\beta)^3\beta\mathcal{N}(-\mu_1+2\mu_2-\mu_3,\sigma_1^2+2\sigma_2^2+\sigma_3^2)+\\
& 2\alpha(1-\beta)^3\beta\mathcal{N}(-2\mu_1+b+\mu_4,2\sigma_1^2+\sigma_2^2+\sigma_4^2)+\\
& 4\alpha(1-\beta)^2\beta^2\mathcal{N}(-\mu_1+\mu_2-\mu_3+\mu_4,\sigma_1^2+\sigma_2^2+\sigma_3^2+\sigma_4^2)+\\
& \alpha(1-\beta)^2\beta^2\mathcal{N}(2\mu_2-2\mu_3,2\sigma_2^2+2\sigma_3^2)+\\
& \alpha(1-\beta)^2\beta^2\mathcal{N}(-2\mu_1+2\mu_4,2\sigma_1^2+2\sigma_4^2)+\\
& 2\alpha(1-\beta)\beta^3\mathcal{N}(b-2\mu_3+\mu_4,\sigma_2^2+2\sigma_3^2+\sigma_4^2)+\\
& 2\alpha(1-\beta)\beta^3\mathcal{N}(-\mu_1-\mu_3+2\mu_4,\sigma_1^2+\sigma_3^2+2\sigma_4^2)+\\
& \alpha\beta^4\mathcal{N}(-2\mu_3+2\mu_4,2\sigma_3^2+2\sigma_4^2)
\end{aligned} \tag{4.127}$$

This can be expressed in the notation used in this thesis by writing

$$\begin{aligned}
\ln Y_R \overset{D}{=} & (1-\alpha)(1-\beta)^4\mathcal{P}_{13}+\alpha(1-\beta)^4\mathcal{R}_{23}+\\
& (1-\alpha)\left[1-(1-\beta)^4\right]\mathcal{R}_{14}+\alpha\left[1-(1-\beta)^4\right]\mathcal{R}_{24}
\end{aligned} \tag{4.128}$$

where the main distribution $\mathcal{P}_{13}$ is given by

$$\mathcal{P}_{13} \overset{D}{=} \mathcal{N}(2\mu_1 - 2\mu_2, 2\sigma_1^2 + 2\sigma_2^2) \tag{4.129}$$

The rogue part distribution $\mathcal{R}_{24}$ is given by

$$\mathcal{R}_{23} \overset{D}{=} \mathcal{N}(-2\mu_1 + 2\mu_2, 2\sigma_1^2 + 2\sigma_2^2) \tag{4.130}$$

The rogue measurement distribution $\mathcal{P}_{14}$ is given by

$$\begin{aligned}
\mathcal{R}_{14} \overset{D}{=} & \frac{2(1-\beta)^3}{[1-(1-\beta)^4]}\mathcal{N}(\mu_1 - 2\mu_2 + \mu_3, \sigma_1^2 + 2\sigma_2^2 + \sigma_3^2) + \\
& \frac{2(1-\beta)^3\beta}{[1-(1-\beta)^4]}\mathcal{N}(2\mu_1 - \mu_2 - \mu_4, 2\sigma_1^2 + \sigma_2^2 + \sigma_4^2) + \\
& \frac{4(1-\beta)^2\beta^2}{[1-(1-\beta)^4]}\mathcal{N}(\mu_1 - \mu_2 + \mu_3 - \mu_4, \sigma_1^2 + \sigma_2^2 + \sigma_3^2 + \sigma_4^2) + \\
& \frac{(1-\beta)^2\beta^2}{[1-(1-\beta)^4]}\mathcal{N}(-2\mu_2 + 2\mu_3, 2\sigma_2^2 + 2\sigma_3^2) + \\
& \frac{(1-\beta)^2\beta^2}{[1-(1-\beta)^4]}\mathcal{N}(2\mu_1 - 2\mu_4, 2\sigma_1^2 + 2\sigma_4^2) + \\
& \frac{2(1-\beta)\beta^3}{[1-(1-\beta)^4]}\mathcal{N}(\mu_1 + \mu_3 - 2\mu_4, \sigma_1^2 + \sigma_3^2 + 2\sigma_4^2) + \\
& \frac{2(1-\beta)\beta^3}{[1-(1-\beta)^4]}\mathcal{N}(-\mu_2 + 2\mu_3 - \mu_4, \sigma_2^2 + 2\sigma_3^2 + \sigma_4^2) + \\
& \frac{\beta^4}{[1-(1-\beta)^4]}\mathcal{N}(2\mu_3 - 2\mu_4, 2\sigma_3^2 + 2\sigma_4^2)
\end{aligned} \tag{4.131}$$

and the combined rogue distribution $\mathcal{P}_{24}$ is given by

$$\begin{aligned}\mathcal{R}_{24}\overset{D}{=}&\frac{2(1-\beta)^3\beta}{[1-(1-\beta)^4]}\mathcal{N}(-\mu_1+2\mu_2-\mu_3,\sigma_1^2+2\sigma_2^2+\sigma_3^2)+\\&\frac{2(1-\beta)^3\beta}{[1-(1-\beta)^4]}\mathcal{N}(-2\mu_1+b+\mu_4,2\sigma_1^2+\sigma_2^2+\sigma_4^2)+\\&\frac{4(1-\beta)^2\beta^2}{[1-(1-\beta)^4]}\mathcal{N}(-\mu_1+\mu_2-\mu_3+\mu_4,\sigma_1^2+\sigma_2^2+\sigma_3^2+\sigma_4^2)+\\&\frac{(1-\beta)^2\beta^2}{[1-(1-\beta)^4]}\mathcal{N}(2\mu_2-2\mu_3,2\sigma_2^2+2\sigma_3^2)+\\&\frac{(1-\beta)^2\beta^2}{[1-(1-\beta)^4]}\mathcal{N}(-2\mu_1+2\mu_4,2\sigma_1^2+2\sigma_4^2)+\\&\frac{2(1-\beta)\beta^3}{[1-(1-\beta)^4]}\mathcal{N}(b-2\mu_3+\mu_4,\sigma_2^2+2\sigma_3^2+\sigma_4^2)+\\&\frac{2(1-\beta)\beta^3}{[1-(1-\beta)^4]}\mathcal{N}(-\mu_1-\mu_3+2\mu_4,\sigma_1^2+\sigma_3^2+2\sigma_4^2)+\\&\frac{\beta^4}{[1-(1-\beta)^4]}\mathcal{N}(-2\mu_3+2\mu_4,2\sigma_3^2+2\sigma_4^2)\end{aligned}\tag{4.132}$$

The issue is that the combined rogue distribution contains many terms, and if any of these terms correspond to a population of parts inside the pass-zone of the gauge, then P_E may be significant.

A simulation of the parts and gauge was constructed, and the following approximation was developed. The main distribution was

$$\mathcal{P}_{13}\overset{D}{=}\mathcal{N}(1.751,0.116^2)\tag{4.133}$$

The rogue part distribution was

$$\mathcal{R}_{23}\overset{D}{=}\mathcal{N}(-1.751,0.116^2)\tag{4.134}$$

The rogue measurement distribution was

$$\begin{aligned}\mathcal{R}_{14}\overset{D}{=}&\frac{2(1-\beta)^3\beta}{[1-(1-\beta)^4]}\mathcal{N}(3.930,0.084^2)+\\&\frac{2(1-\beta)^3\beta}{[1-(1-\beta)^4]}\mathcal{N}(-1.434,0.116^2)+\\&\frac{4(1-\beta)^2\beta^2}{[1-(1-\beta)^4]}\mathcal{N}(0.745,0.084^2)+\\&\frac{(1-\beta)^2\beta^2}{[1-(1-\beta)^4]}\mathcal{N}(6.109,0.030^2)+\\&\frac{(1-\beta)^2\beta^2}{[1-(1-\beta)^4]}\mathcal{N}(-4.618,0.116^2)+\\&\frac{2(1-\beta)\beta^3}{[1-(1-\beta)^4]}\mathcal{N}(2.924,0.030^2)+\\&\frac{2(1-\beta)\beta^3}{[1-(1-\beta)^4]}\mathcal{N}(-2.439,0.085^2)+\\&\frac{\beta^4}{[1-(1-\beta)^4]}\mathcal{N}(-0.260,0.030^2)\end{aligned} \tag{4.135}$$

The combined rogue distribution was

$$\begin{aligned}\mathcal{R}_{24}\overset{D}{=}&\frac{2(1-\beta)^3\beta}{[1-(1-\beta)^4]}\mathcal{N}(1.434,0.116^2)+\\&\frac{2(1-\beta)^3\beta}{[1-(1-\beta)^4]}\mathcal{N}(-3.930,0.0845^2)+\\&\frac{4(1-\beta)^2\beta^2}{[1-(1-\beta)^4]}\mathcal{N}(-0.745,0.084^2)+\\&\frac{(1-\beta)^2\beta^2}{[1-(1-\beta)^4]}\mathcal{N}(4.618,0.116^2)+\\&\frac{(1-\beta)^2\beta^2}{[1-(1-\beta)^4]}\mathcal{N}(-6.109,0.030^2)+\\&\frac{2(1-\beta)\beta^3}{[1-(1-\beta)^4]}\mathcal{N}(-2.924,0.030^2)+\\&\frac{2(1-\beta)\beta^3}{[1-(1-\beta)^4]}\mathcal{N}(2.439,0.085^2)+\\&\frac{\beta^4\mathcal{N}(0.260,0.030^2)}{[1-(1-\beta)^4]}\end{aligned} \tag{4.136}$$

Thus, if only the main distribution and the rogue process distribution are considered, then $Y_R > 1$ represents an appropriate pass-zone. However, in this case, it is known that combined rogue distributions are also present. As such, an improved pass-zone that results in a sufficiently large P_A must be computed.

4.3.5.1 Determining an Appropriate Pass-Zone

From a practical point of view, it is desirable to have P_A as large as possible. The conventional solution is to choose a desired C_{PK} value and design the production process accordingly. Often $C_{PK} = 2$ is chosen, and a normally distributed process is assumed. In this case, this implies $\alpha = \beta = 0$. From (3.27), these assumptions imply $U_T = 6\sigma_{13}$. Additionally, from Table 2.1, a normally distributed $C_{PK} = 2$ process results in $P_A \approx 0.999\,999\,998$. Assuming $\alpha = \beta = 0$, (4.109) implies $P_A = \rho_{13}$, and hence $\rho_{13} \approx 0.999\,999\,998$, where ρ_{13} is the probability the gauge will accept a part from the main distribution.

Later in the thesis, we will be using a different pass-zone with a different definition of U_T. In order to keep everything comparable throughout the thesis, it is desirable to choose a fixed value of ρ_{13}, because the definition of ρ_{13} is unaffected by α, β or the definition of U_T. Some calculations used in later sections fail to converge if $\rho_{13} \approx 0.999\,999\,998$ is assumed, for reasons of numerical precision. As such, a smaller value of ρ_{13} needed to be chosen, and $\rho_{13} = 0.9999$ was found to be the largest value of ρ_{13} that worked in all of the relevant calculations.

As such, for numerical purposes, the following sections assume a constant $\rho_{13} = 0.9999$, and compute the pass-zone accordingly. From (4.96) and (4.109), if $\rho_{13} = 0.9999$, then

$$P_A = (1-\alpha)(1-\beta)\rho_{13} \qquad \Longrightarrow \quad P_A \approx 1 \tag{4.137}$$

$$P_C = \frac{P_E}{P_A} \qquad \Longrightarrow \quad P_C \approx P_E \tag{4.138}$$

Assuming $\rho_{13} = 0.9999$ has the following advantages

1. For many processes $\rho_{13} = 0.9999$ yields a sufficiently good approximation for a low scrap/waste process.

2. Later in this chapter, methods are presented that can reduce $\rho_{24} > \lambda_{24}$ to very small numbers relative to P_E^*. A "small" $\rho_{13} = 0.9999$ ensures numeric tractability. Making ρ_{13} very large and ρ_{24} very small causes numeric, computational and simulation issues. For instance, some practical systems may desire $\rho_{13} \geq 1 - 10^{-9}$ and $\rho_{24} \leq 10^{-9}$, and this level of numeric accuracy is computationally challenging.

3. Assuming a constant ρ_{13} standardizes the U_T calculations. This makes the results for P_E directly comparable between univariate and multivariate gauges.

4. The factories studied in this thesis expressed rejects relative to the amount initially produced P_E, not the amount finally shipped P_C. Assuming $\rho_{13} = 0.9999$ and $P_C \approx P_E$ was acceptable.

For the application in this example, the pass-zone is $U_1 \leq \ln Y_R \leq U_2$, where $U_1 = 1.320$ and $U_2 = 2.182$ are the application-specific upper and lower limits, respectively. This pass-zone is shown in Figure 4.6, and corresponds to the dark green lines on either side of the word "Accept". Everything outside of the accept zone is reject. In Figure 4.6, $\alpha = \beta = 0.02$, and the distributions are from equations (4.133) through (4.136).

4.3.5.2 Simulation Results

The problem with this gauge can be seen in the distribution of $\ln Y_R$ developed from simulation, as shown in Figure 4.6. The upper half of Figure 4.6 shows the main distribution (where no rogue events occur), the rogue process distributions, the rogue measurement distributions, and the combined

rogue distributions. Importantly, one of the rogue process distributions is almost completely obscured by the much larger main (rogue-free) distribution. For the gauge to be economically useful, it must accept a large percentage of the parts in the main distribution.

Just because the gauge accepts the part, does not mean the part will be acceptable to the end-user. In particular, it is possible the that manufacturing process made a defective part that just happened to measure within specification due to a simultaneous error in the measurement system. A defective part that is incorrectly accepted by the gauge is an outgoing defect. Outgoing defects happen frequently if a combined rogue distribution occurs in the same region as the main distribution. Figure 4.6 shows that both the main distribution and a combined rogue distribution occur in the same region, which makes it impossible to select a pass-zone that has a high probability ρ_{13} of passing acceptable parts, while simultaneously rejecting the combined rogue parts.

The lower half of Figure 4.6 shows the main distribution (where no rogue events occur) against the total rogue distribution (where any E_α or E_i occurs).

The fundamental problem with this gauge algorithm is that selecting any pass-zone that admits the main distribution, will also inevitably admit the combined rogue distribution that occurs in the same region. As such, this gauge algorithm is ineffective for detecting combined rogue events. This application was eventually solved using the multivariate methods described in §4.4.

4.3.5.3 Reasons for In-plant Observations

Two key issues confused the in-plant analysis of the gauge. Firstly, if the density $f(Y_R)$ is presented on a linear scale, as shown in Figure 4.7, the

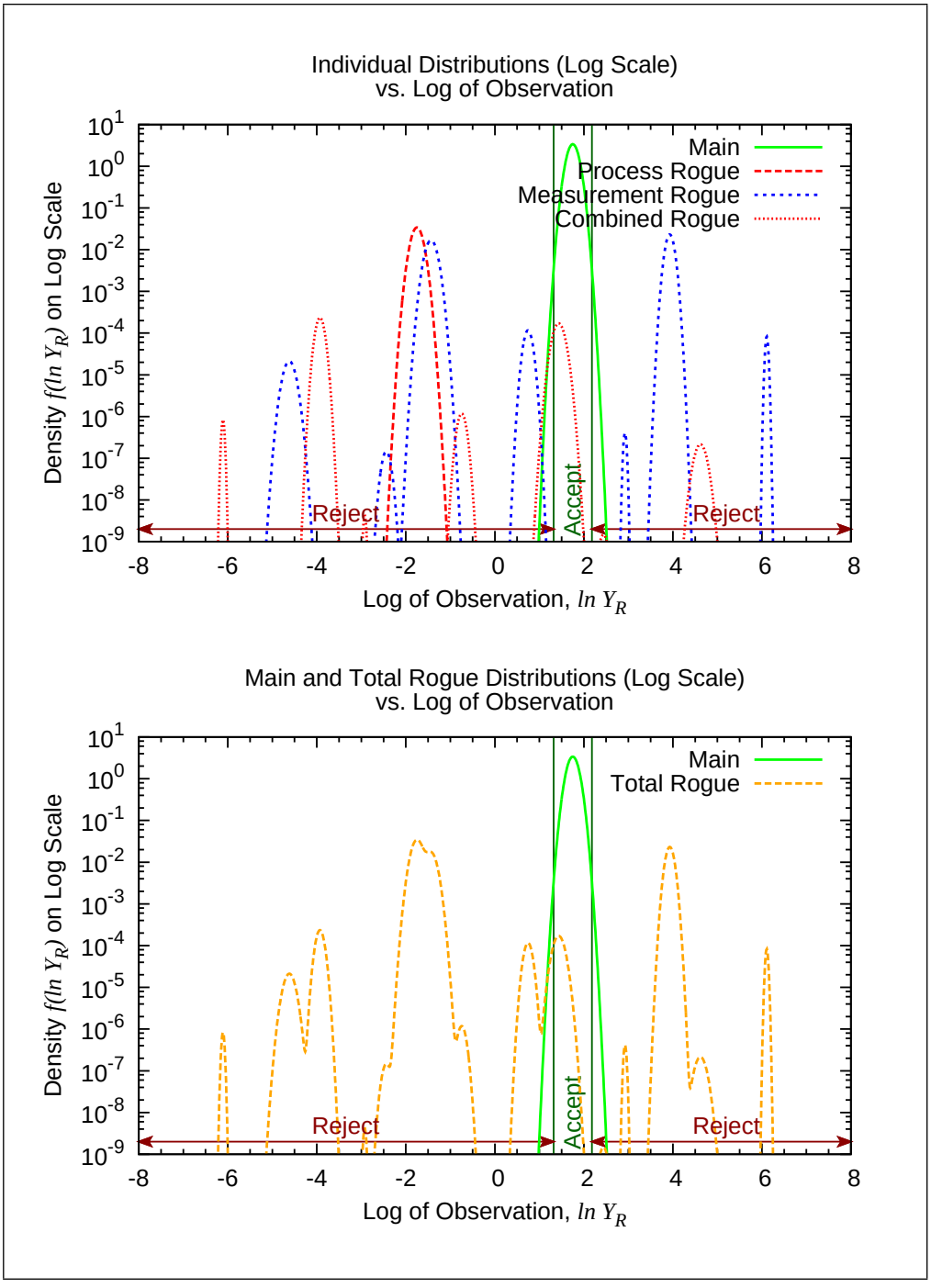

Figure 4.6: Graph of the distributions from (4.128) and (4.133) through (4.136). This graph shows $\ln Y_R$ on the abscissa and uses a logarithmic scale on the ordinate axis. See Figure 4.7 for linear scale.

gauge appears very effective. No troublesome rogue distributions are visible, as they do not register with linear scaling. The rogue distributions are only detectable if log scaling is used, and if a sufficiently large number of data points exist for the log scaling to be meaningful. Small sample sizes, such as the 10 part sample used in typical GR&R tests, are ineffective at detecting the existence of infrequently occurring rogue distributions.

The second issue that confused the in-plant analysis of the gauge is the presence of the measurement rogue distributions near the main distribution. This can give the false impression that all rogue events near the pass-zone are measurement rogue events. Specifically, if simple normal distributions are assumed, and the method from Wall (1997) is applied incorrectly, then the expected result of tightening the pass-zone is increased aberrant rejects and significantly decreased outgoing defects. However, this does not work in this application, because of the presence of the combined rogue distribution in the pass-zone.

In gauges where the rogue mixture model applies, it is necessary to find the rogue measurement distributions and the rogue process distributions. The rogue distributions from all combinations of measurement E_i and process E_α events must be estimated. Any rogue distribution involving a process rogue event E_α must be outside the pass-zone of the gauge, such that $P_E < P_E^*$. Ideally, to eliminate any confusion whatsoever, all of the rogue distributions should be outside the pass-zone of the gauge.

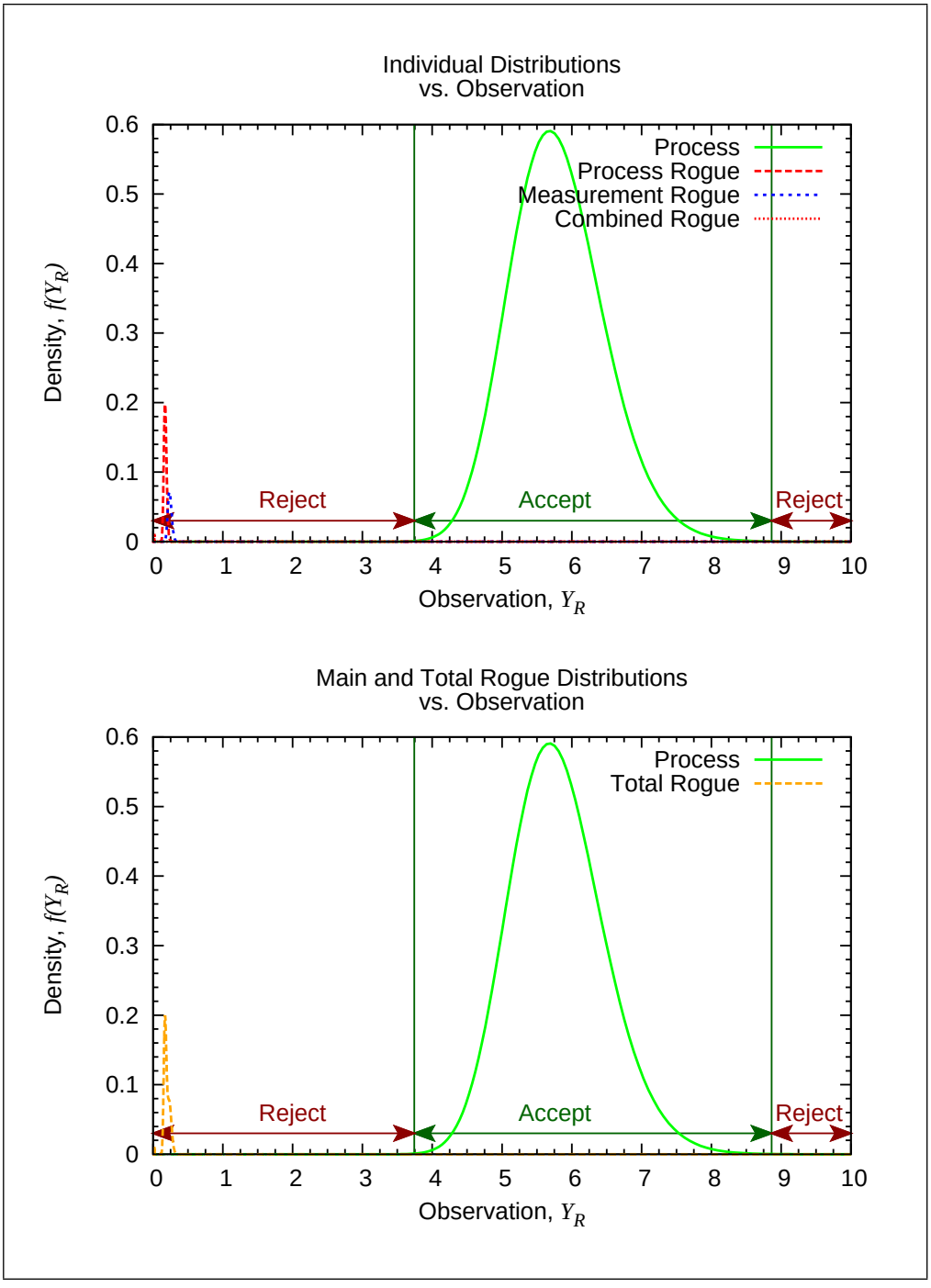

Figure 4.7: Graph of Distributions from (4.128) and (4.133) through (4.136). This graph shows Y_R on the abscissa and uses a linear scale on the ordinate axis. See Figure 4.6 for a logarithmic scale.

4.3.5.4 General Simplified Model

Consider the case where we define a Y_D of the form

$$\begin{aligned} Y_D \overset{D}{=} & (1-\alpha-\beta-\varepsilon+\alpha\beta)\mathcal{P}_{13}(\mu_{13},\sigma_{13}^2) + \\ & \alpha(1-\beta)\mathcal{R}_{23}(\mu_{23},\sigma_{23}^2) + \\ & (1-\alpha)\beta\mathcal{R}_{14}(\mu_{14},\sigma_{14}^2) + \\ & \alpha\beta\mathcal{R}_{24}(\mu_{24},\sigma_{24}^2) + \\ & \varepsilon\mathcal{R}_x \end{aligned} \tag{4.139}$$

where

$\mathcal{P}_{13}(\mu_{13},\sigma_{13}^2)$ is the main (rogue-free) distribution, with mean μ_{13} and variance σ_{13}^2.

$\mathcal{R}_{23}(\mu_{23},\sigma_{23}^2)$ is the process rogue distribution, with mean μ_{13} and variance σ_{13}^2.

$\mathcal{R}_{14}(\mu_{14},\sigma_{14}^2)$ is the measurement rogue distribution, with mean μ_{13} and variance σ_{13}^2.

$\mathcal{R}_{24}(\mu_{24},\sigma_{24}^2)$ is the combined rogue distribution, with mean μ_{13} and variance σ_{13}^2.

$\mathcal{R}_\varepsilon$ represents any portions of Y_D that occur outside of the pass-zone, and are not explained by any of the above remaining terms.

α is the probability of a rogue process event E_2.

β is the probability of a rogue process event E_4.

ε is the probability that an event happens outside the pass-zone, and is not explained by any of the above terms.

For any event E_{ij}, and suitably chosen $\mathcal{P}_{13}$ through $\mathcal{R}_{24}$, a Y_D can be found such that

$$\Pr(U_1 \leq Y_D \leq U_2 \cap E_{ij}) \approx \Pr(U_1 \leq \ln Y_R \leq U_2 \cap E_{ij}) = \rho_{ij} \tag{4.140}$$

For small α and β

$$P_A \approx (1-\alpha)(1-\beta)\rho_{13} \tag{4.141}$$

and

$$P_E \approx \alpha\beta\rho_{24} \tag{4.142}$$

To the degree the approximation holds, the simplified form from (4.139) can be used to determine if $P_E \leq P_E^*$.

This model is interesting, because P_A and P_E can be calculated from Y_D, and Y_D can be a simple approximation to a more complex distribution.

4.3.5.5 Applying the General Simplified Model

Reviewing Figure 4.6, three numerically significant normal distributions are present in the pass-zone. These normal distributions correspond to the dominant main, measurement rogue, and combined rogue distributions in the pass-zone. In the case of this example, no significant process rogue distribution exists in the pass-zone. Referring to equations (4.133) through (4.136), a suitable approximation for $\ln Y_R$ in the pass-zone is

$$\begin{aligned} Y_D \overset{D}{=} \; & (1-\alpha)(1-\beta)^4\mathcal{N}(1.751, 0.116^2) + \\ & 4\beta^2(1-\alpha)(1-\beta)^2\mathcal{N}(0.745, 0.084^2) + \\ & 2\alpha\beta(1-\beta)^3\mathcal{N}(1.434, 0.116^2) + \\ & \varepsilon\mathcal{R}_\varepsilon \end{aligned} \tag{4.143}$$

As $\mathcal{R}_\varepsilon$ parameterize the portions of Y_D outside of the pass-zone, it is not necessary to compute $\mathcal{R}_\varepsilon$ to find P_A or P_E.

At this point, ρ_{13} can be calculated from the distribution function of the relevant normal distribution.

$$\rho_{13} \approx 1 - \Phi\left(\frac{U_2 - 1.751}{0.116}\right) + \Phi\left(\frac{1.751 - U_1}{0.116}\right) \tag{4.144}$$

From (4.143), it can be seen that the appropriate approximation for P_A is

$$P_A \approx (1-\alpha)(1-\beta)^4\rho_{13} \tag{4.145}$$

Similarly, ρ_{24} is given by

$$\rho_{24} \approx \Phi\left(\frac{U_2 - 1.434}{0.116}\right) - \Phi\left(\frac{1.434 - U_1}{0.116}\right) \tag{4.146}$$

Assuming $U_1 = 1.320$, $U_2 = 2.182$ yields

$$\rho_{24} \approx 0.838 \tag{4.147}$$

The appropriate approximation for P_E is

$$P_E \approx 2\alpha\beta(1-\beta)^3\rho_{24} \tag{4.148}$$

$$\approx 2\alpha\beta\rho_{24} \tag{4.149}$$

Assuming, as before, $\alpha = \beta = 0.02$, and evaluating numerically, it is obvious that $P_E \gg P_E^*$ and this application does not meet specification when $P_E^* = 10^{-9}$. Another example of these calculations is given in §9.6.2.

4.3.5.6 Industrial Use

This simplified model can be used to practical effect. Given a specification of $P_E^* = 10^{-9}$ and engineering knowledge of the rogue measurement and rogue process events present in the manufacturing process, it is possible to find Y_D and hence if $P_E \gg P_E^*$. For instance, as discussed in §4.3.4, α, β and

estimates of the rogue measurement and rogue process distributions can be found by observing the manufacturing process. From this, the nature of the combined rogue distribution can be inferred, and hence Y_D. Given Y_D it is possible to find P_E.

In terms of calculations that can be performed on a noisy production line, sometimes it is very straightforward to show that P_E is unacceptably large. Assume line speeds of 100 to 5000 parts/hour. If α, β and estimates of the rogue measurement and rogue process distributions can be found in a few hours, then α and β must be large. For instance, if at least 10 rogue gauge events and at least 10 rogue process events in 10,000 parts were detected, then $\alpha \geq 0.001$ and $\beta \geq 0.001$. If it can be assumed that λ_{24} is also large, for instance $\lambda_{24} > 0.1$, then (4.110) yields $P_E \geq 10^{-7}$ which implies $P_E \gg 10^{-9}$.

It is more challenging to show $P_E \leq P_E^*$, because of the sample sizes required. Having automated data gathering tools, like a runtime profiler, makes it easier to gather large samples. Given sufficient data, it is possible to find the dominant main, rogue, and combined rogue distributions in the pass-zone, and thus a suitable Y_D approximation can be found. From this, P_E can be found.

In the industrial high-reliability application from Chapter 9, the simple model of $P_E = \alpha\beta\gamma_{24}$ worked, and the approximation $Y_D \approx Y$ could be used. To reduce the gauge outgoing defect rate, it was necessary to reduce γ_{24} by finding algorithms that did not have large combined rogue populations in the pass-zone.

4.3.6 Univariate Gauging Performance

One of the shop floor observations that inspired this thesis was that if a univariate gauge has rogue distributions that might cause outgoing defects, it is just a matter of time before they are detected. This section provided the theory to explain that observation when it applies.

From Figure 4.5, P_C versus U_T curve has a flat bottom. For some gauges, even in the limit $P_C > P_E^*$. Thus, for some problems, no significant advantage exists in reducing the U_T and tolerating a small P_A.

In the case of true-attribute gauges, simplified equations to estimate (4.109) and (4.110) in terms of α, β, ρ_{13} and γ_{24} were presented. A method was presented to find α and β in a practical gauge. (4.91) provides an upper bound on γ_{24}, as $\rho_{24} \geq \gamma_{24}$. Given, α, β, and either γ_{24} or a bound on γ_{24}, (4.110) provides an estimate of P_E, the probability of outgoing defects.

In view of the limit L from (4.99), many univariate gauges are inadequate for high-reliability gauging. However, useful applications for univariate gauging exist. As covered in Chapters 6 through 8, if the feedback loop is made fairly reliable and the customers' reliability expectations are modest, then α and β may be made sufficiently small that the problems posed by excessive λ_{24} are not significant. However, this is not truly a high-reliability gauge, as a measurable number of outgoing defects are present. For systems with outgoing defect rate limits of $P_E^* \leq 1$ ppb, it is recommended that the multivariate methods discussed in the next section be used.

4.4 Analysis of a Multivariate Gauge

4.4.1 Definitions

In a multivariate analysis, the random variables X, W, and Y, are converted into their vector forms and expressed as the column matrices $\boldsymbol{X}$, $\boldsymbol{W}$, and $\boldsymbol{Y}$, respectively. This changes the basic definitions to

$$\boldsymbol{X} \overset{D}{=} (1-\alpha)\,\mathcal{P}_1\left(\mathbf{0}, \boldsymbol{\Sigma}_1\right) + \alpha \mathcal{R}_2\left(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2\right) \tag{4.150}$$

$$\boldsymbol{W} \overset{D}{=} (1-\beta)\,\mathcal{P}_3\left(\mathbf{0}, \boldsymbol{\Sigma}_3\right) + \beta \mathcal{R}_4\left(\boldsymbol{\mu}_4, \boldsymbol{\Sigma}_4\right) \tag{4.151}$$

$$\boldsymbol{Y} = \boldsymbol{X} + \boldsymbol{W} \tag{4.152}$$

where

$\boldsymbol{X}$ is an N-element vector random variable representing the process error.

$\boldsymbol{W}$ is an N-element vector random variable representing the gauge error.

$\boldsymbol{Y}$ is an N-element vector random variable representing the gauge error representing the observation.

$\mathcal{P}_1$ is the process distribution which occurs with event E_1, and probability $1-\alpha$.

$\mathcal{R}_2$ is the rogue process distribution which occurs with event E_2, and probability α.

$\mathcal{P}_3$ is the measurement distribution which occurs with event E_3, and probability $1-\beta$.

$\mathcal{R}_4$ is the rogue measurement distribution which occurs with event E_4, and probability β.

$\boldsymbol{\mu}_2$ and $\boldsymbol{\mu}_4$ are the means of the corresponding rogue distributions. They are N-element vectors.

$\boldsymbol{\Sigma}_1$, $\boldsymbol{\Sigma}_2$, $\boldsymbol{\Sigma}_3$, and $\boldsymbol{\Sigma}_4$ are $N \times N$ variance matrices for the corresponding distributions.

In application, a gauge observes $\boldsymbol{Y}$ and inferences about $\boldsymbol{X}$ and $\boldsymbol{W}$ can be made only indirectly.

The mixture distribution for $\boldsymbol{Y}$ is given by

$$\begin{aligned} \boldsymbol{Y} \overset{D}{=} \; & (1-\alpha)(1-\beta)\mathcal{P}_{13}(\mathbf{0}, \boldsymbol{\Sigma}_{13}) + \\ & \alpha(1-\beta)\mathcal{R}_{23}(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_{23}) + \\ & (1-\alpha)\beta\mathcal{R}_{14}(\boldsymbol{\mu}_4, \boldsymbol{\Sigma}_{14}) + \\ & \alpha\beta\mathcal{R}_{24}(\boldsymbol{\mu}_{24}, \boldsymbol{\Sigma}_{24}) \end{aligned} \tag{4.153}$$

where

$\mathcal{P}_{13}$ is the main distribution which occurs when the combined event $E_{13} = E_1 \cap E_3$ occurs.

$\mathcal{R}_{23}$ is the rogue process distribution which occurs when the combined event $E_{23} = E_2 \cap E_3$ occurs.

$\mathcal{R}_{14}$ is the rogue main distribution which occurs when the combined event $E_{14} = E_1 \cap E_4$ occurs.

$\mathcal{R}_{24}$ is the combined rogue distribution which occurs when the combined event $E_{24} = E_2 \cap E_4$ occurs.

The mean $\boldsymbol{\mu}_{24} = \boldsymbol{\mu}_2 + \boldsymbol{\mu}_4$. The variance matrices $\boldsymbol{\Sigma}_{ij}$ are given by

$$\boldsymbol{\Sigma}_{ij} = \boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j \tag{4.154}$$

for $i \in \{1, 2\}$ and $j \in \{3, 4\}$.

4.4.1.1 Problem Definition

As in the univariate case, the goal is to achieve a large probability of acceptance P_A while maintaining a low outgoing defect rate P_E. In the multivariate case, this can be achieved by ensuring sufficient observations are present with the correct eigenvalue properties such that the probability of a combined rogue event occurring inside the pass-zone of the gauge becomes insignificant (much less than 1 ppb). Define

$$P_A = \Pr(g(\boldsymbol{Y}) \leq U_T) \tag{4.155}$$

With the constraint

$$P_E^* \geq P_E \tag{4.156}$$

where

P_E is the outgoing defect rate.

P_E^* is the upper limit on the outgoing defect rate and is a customer specification.

$g(\boldsymbol{Y})$ is a score function that is defined in the next section.

As in the univariate case, it is assumed that

$$P_E \approx \alpha\beta\gamma_{24} \tag{4.157}$$

The multivariate γ_{ij} and ρ_{ij} are given by

$$\gamma_{ij} = \Pr(g_X(\boldsymbol{X}) > U_X \cap g(\boldsymbol{Y}) \leq U_T | E_{ij}) \tag{4.158}$$

$$\rho_{ij} = \Pr(g(\boldsymbol{Y}) \leq U_T | E_{ij}) \tag{4.159}$$

$$\rho_{ij} \geq \gamma_{ij} \tag{4.160}$$

where $g_X(\boldsymbol{X})$ is the score function for the random vector $\boldsymbol{X}$.

The probability ρ_{24} has the property that it is often easier to compute than γ_{24}. As such, in applications where ρ_{24} can be made very small, it is simpler to set the upper bound ρ_{24} sufficiently small rather than to compute γ_{24}. An important advantage of minimizing ρ_{24} is that it is unnecessary to assume a specific U_X value as was done in §4.3.3. Instead, the new assumption is that any part involving a combined rogue event should be rejected. If all parts affected by a combined rogue event are assumed to be rejects, then $\gamma_{24} = \rho_{24}$ for practical purposes.

4.4.1.2 Score Functions, Q Score

In the multivariate case, the input value is a vector, and this must be converted to a scalar to be compared against the tolerance limit. The Q score

used here is a quadratic form. As such, it has special statistical properties that make the following analysis much more straightforward. The Q score has the property that it is the optimal pass-zone for admitting the main distribution.

This thesis makes the assumption that $g(\boldsymbol{Y}) = g_Q(\boldsymbol{Y})$, with the $Q = g_Q(\boldsymbol{Y})$ score function given by

$$Q = g_Q(\boldsymbol{Y}) = ||\boldsymbol{T}||^2 \quad \text{and} \quad \boldsymbol{T} = \boldsymbol{\Lambda}^{-1}\mathbf{V}^T\boldsymbol{Y} \tag{4.161}$$

where

$\boldsymbol{T}$ is the $\boldsymbol{T}$-vector, an intermediate vector used in the computation of Q.[4]

$\boldsymbol{\Lambda}$ is an $n \times n$ eigenvalue matrix, with n-eigenvalues on it's main diagonal.

$\mathbf{V}$ is an $N \times n$ eigenvector matrix consisting of n-eigenvectors.

The $\boldsymbol{\Lambda}$ and $\mathbf{V}$ can be found by decomposing the variance $\boldsymbol{\Sigma}_{13}$ of the main distribution as

$$\boldsymbol{\Sigma}_{13} = \mathbf{V}\boldsymbol{\Lambda}^2\mathbf{V}^T + \mathbf{E} \tag{4.162}$$

In practice, the eigenvector matrix $\mathbf{V}$ and eigenvalue matrix $\boldsymbol{\Lambda}$ are found through the SVD decomposition of $\boldsymbol{\Sigma}_{13}$. Any error present in the SVD decomposition is represented by the zero or positive semi-definite error matrix $\mathbf{E}$. This error is zero when $\operatorname{rank}(\boldsymbol{\Sigma}_{13}) = n$.

The error $\boldsymbol{\epsilon}$ in reducing $\boldsymbol{Y}$ to n-eigenvalues via $\boldsymbol{T}$ can be computed from

$$\boldsymbol{\epsilon} = \boldsymbol{Y} - \mathbf{V}\boldsymbol{\Lambda}\boldsymbol{T} \tag{4.163}$$

[4]If a Principle Components Analysis (PCA) was being performed, the $\boldsymbol{T}$-vector would contain the principle components. See Jolliffe (2002, p. 2) and Esbensen *et al.* (2002, p. 34).

When $||\boldsymbol{\epsilon}|| = 0$ the approximation is exact, and $Y = \mathbf{V}\boldsymbol{\Lambda}T$. $||\boldsymbol{\epsilon}||$ is monitored as a large $||\boldsymbol{\epsilon}||$ is a clear indicator that a rogue effect is present in the system. However, by itself, $||\boldsymbol{\epsilon}||$ is not a reliable indicator of rogue events to a P_E^* reliability level. Hence, the $Q = g_Q(Y)$ was used.

Assuming only nonzero eigenvalues are included in $\boldsymbol{\Lambda}^2$, then for any valid variance matrix $\boldsymbol{\Sigma}_{13}$

$\boldsymbol{\Lambda}^2$ is positive definite,

$\boldsymbol{\Sigma}_{13}$ is positive semi-definite,

$\mathbf{V}$ is an orthonormal basis with the property that $\mathbf{V}\mathbf{V}^T = \mathbf{I}$.

Additionally, the matrix $\boldsymbol{\Lambda}^k$ can be found for any integer k from

$$\boldsymbol{\Lambda}^k = \begin{bmatrix} \lambda_1^k & 0 & 0 & \cdots & 0 \\ 0 & \lambda_2^k & 0 & \cdots & 0 \\ 0 & 0 & \lambda_3^k & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \lambda_n^k \end{bmatrix}_{n \times n} \tag{4.164}$$

where $\lambda_1, ...\lambda_n$ are the n-eigenvalues. Note that $\boldsymbol{\Lambda}\boldsymbol{\Lambda}^{-1} = I$, and thus $\boldsymbol{\Lambda}^{-1}$ and $\boldsymbol{\Lambda}^{-2}$ are the inverses of $\boldsymbol{\Lambda}$ and $\boldsymbol{\Lambda}^2$ respectively.

From the above, another equivalent definition for $g_Q(Y)$ can be found from

$$g_Q(Y) = ||\boldsymbol{T}||^2 \tag{4.165}$$

$$g_Q(Y) = \boldsymbol{T}^T\boldsymbol{T} \tag{4.166}$$

$$g_Q(Y) = \boldsymbol{Y}^T\mathbf{V}\boldsymbol{\Lambda}^{-1}\boldsymbol{\Lambda}^{-1}\mathbf{V}^T\boldsymbol{Y} \tag{4.167}$$

$$\tag{4.168}$$

Define the pseudo-inverse $\boldsymbol{\Sigma}_{13}^+$ as

$$\boldsymbol{\Sigma}_{13}^+ = \mathbf{V}\boldsymbol{\Lambda}^{-2}\mathbf{V}^T \tag{4.169}$$

This yields

$$g_Q(\boldsymbol{Y}) = \boldsymbol{Y}^T \boldsymbol{\Sigma}_{13}^{+} \boldsymbol{Y} \tag{4.170}$$

The origins of the above definitions are not unique to this thesis. (Ryan, 2000, p. 272) defined the multivariate Q-score as

$$Q = (\boldsymbol{X} - \boldsymbol{\mu}_X)^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{X} - \boldsymbol{\mu}_X) \tag{4.171}$$

In this thesis, the mean of the main distribution is assumed to be zero, making $\boldsymbol{\mu}_X = 0$. This makes the $g_X(\boldsymbol{X})$-score function

$$g_X(\boldsymbol{X}) = \boldsymbol{X}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{X} \tag{4.172}$$

which means that the key difference between (4.170) and Ryan's form is the use of the pseudo-inverse to deal with the case where $\boldsymbol{\Sigma}^{-1}$ is a singular noninvertible positive semi-definite variance matrix.

The automotive industry expects the suppliers to hold a consistent process via SPC monitoring of the gauge observations to ensure a stable and consistent process. In this case, the goal is to ship only products from the main distribution $\mathcal{P}_{13}(\boldsymbol{0}, \boldsymbol{\Sigma}_{13})$ to the customer. As such, the relevant $g_Y(\boldsymbol{Y})$ is

$$g(\boldsymbol{Y}) = g_Q(\boldsymbol{Y}) = \boldsymbol{Y}^T \boldsymbol{\Sigma}_{13}^{+} \boldsymbol{Y} \tag{4.173}$$

From a practical point of view, the matrix computation in (4.173) is inefficient, hence the mathematically equivalent method given in (4.161) is used.

4.4.1.3 The χ^2 Distribution

The $g_Q(Y)$ function is a quadratic form. To calculate probabilities based on this quadratic form, it is necessary to have the relevant distribution function. When the following assumptions apply, the χ^2 distribution can be used

for this purpose.

Let Z be a vector of normal variates with means and variances given by

$$\begin{aligned} \boldsymbol{Z} &= \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) && (4.174)\\ \boldsymbol{Z} &= \begin{bmatrix} Z_1 & \cdots & Z_n \end{bmatrix}^T && (4.175)\\ \boldsymbol{\mu} &= \begin{bmatrix} \mu_1 & \cdots & \mu_n \end{bmatrix}^T && (4.176)\\ \boldsymbol{\Sigma} &= \begin{bmatrix} \sigma_1^2 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2^2 & 0 & \cdots & 0 \\ 0 & 0 & \sigma_3^2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \sigma_n^2 \end{bmatrix} && (4.177) \end{aligned}$$

where $\sigma_i^2 > 0$ for all $i \in \{1...n\}$.

The central χ^2 distribution is given by

$$\sum_{i=1}^{n} \frac{(Z_i - \mu_i)^2}{\sigma_i^2} \overset{D}{=} \chi^2(0) \tag{4.178}$$

or equivalently

$$(\boldsymbol{Z} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\boldsymbol{Z} - \boldsymbol{\mu}) \overset{D}{=} \chi^2(n, 0) \tag{4.179}$$

The noncentral chi-square distribution with parameter $k^2 \geq 0$ is given by

$$\sum_{i=1}^{n} \frac{Z_i^2}{\sigma_i^2} \overset{D}{=} \chi^2(n, k^2) \tag{4.180}$$

or equivalently

$$\boldsymbol{Z}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{Z} \overset{D}{=} \chi^2(n, k^2) \tag{4.181}$$

where k^2 is given by

$$k^2 = \boldsymbol{\mu}^T \boldsymbol{\Sigma}^{-1} \boldsymbol{\mu} = \sum_{i=1}^{n} \frac{Z_i^2}{\sigma_i^2} \tag{4.182}$$

The distribution function of the χ^2 distribution can be used for probability estimation as follows

$$\Pr(\boldsymbol{Z}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{Z} < U) = F_{\chi^2}(U; n, k^2) \tag{4.183}$$

where U is any upper limit and $F_{\chi^2}(U; n, k^2)$ is the distribution function of the χ^2 distribution in n degrees of freedom and with the noncentrality parameter k^2 from (4.182).

4.4.1.4 Initial Estimates

If $\boldsymbol{\Sigma}_{ij} = \sigma_{ij}^2\boldsymbol{\Sigma}_{13}$ for some constant σ_{ij}^2, then the noncentral χ^2 distribution can be used to estimate the likelihood of a rogue event existing inside the $g_Q(\boldsymbol{Y}) \leq U_T$ pass-zone.

For any event E_{ij}, the noncentrality constant k_{ij}^2 can be computed from

$$\begin{aligned} k_{ij}^2 &= ||\mathrm{E}(\boldsymbol{T}|E_{ij})||^2 && (4.184)\\ &= ||\boldsymbol{\Lambda}^{-1}\mathbf{V}^T\boldsymbol{\mu}_{ij}||^2 && (4.185)\end{aligned}$$

where

k_{ij} is the noncentrality constant assuming event E_{ij} is present.

$\boldsymbol{\Lambda}$ is from (4.161).

$\mathbf{V}$ is from (4.161).

$\boldsymbol{\mu}_{ij}$ is the expected value of $\mathrm{E}(Y|E_{ij})$. In this case, $\boldsymbol{\mu}_{13} = \mathbf{0}$, $\boldsymbol{\mu}_{23} = \boldsymbol{\mu}_2$, $\boldsymbol{\mu}_{14} = \boldsymbol{\mu}_4$, $\boldsymbol{\mu}_{24} = \boldsymbol{\mu}_2 + \boldsymbol{\mu}_4$. See (4.153).

The relevant ρ_{ij} can be found from

$$\begin{aligned} \rho_{ij} &= \Pr(||\boldsymbol{T}||^2 < U_T|E_{ij}) && (4.186)\\ \rho_{ij} &= F_{\chi^2}\left(\frac{U_T}{\sigma_{ij}^2}; n, \frac{k_{24}^2}{\sigma_{ij}^2}\right) && (4.187)\end{aligned}$$

where

$F_{\chi^2}(U; n, k^2)$ is the distribution function of the χ^2 distribution.

σ_{ij} is the constant such that $\mathbf{\Sigma}_{ij} = \sigma_{ij}^2 \mathbf{\Sigma}_{13}$. By definition, $\sigma_{13}^2 = 1$.

ρ_{ij} is the conditional probability of finding elements from the associated distribution in the pass-zone. Importantly, ρ_{13} is the probability of finding observations from the main distribution in the pass-zone given that no rogue events are occurring. Ideally, ρ_{13} should be large, *i.e.* $\rho_{13} \approx 1$. The probability ρ_{24} is the probability of finding combined rogue events in the pass-zone, given that a combined rogue event is occurring. To meet specification, $\alpha\beta\rho_{24} < P_E^*$. As such, ρ_{24} should be very small.

U_T sets the size of the pass-zone.

n is the number of independent observations. $\text{rank}(\mathbf{\Sigma}_{13}) = n$.

Assuming the χ^2 distribution applies is often useful for estimating the impact of n on the probabilities involved in advance of equipment construction. When little is known about $\mathbf{\Sigma}_{24}$, then some assumptions must be made. When the runtime profiler was initially being tested, assuming the χ^2 distribution applied provided a useful tool for the qualitative selection of different algorithms in different industrial applications.

In at least some industrial applications, quantitative predictions based on the χ^2 distribution should be treated with caution. For $n > 2$, no cases were found where the assumption $\mathbf{\Sigma}_{24} \approx \sigma_{24}^2 \mathbf{\Sigma}_{13}$ applies. Hence, assuming the χ^2 distribution applies may not yield accurate numerical predictions. The χ^2 distribution is a good first start, when no better information is available. However, when better data becomes available, better methods should be used, as described in the next section.

4.4.2 Numerical Methods for General Variance Matrices

In most practical gauges, the combined rogue variance Σ_{24} is unrelated to Σ_{13}. This often improves rogue event detection, and is an effect that can be exploited to significant benefit. The focus of this section is finding ρ_{24} by reduction to a standard computational form.

Define the m eigenvectors $\mathbf{V}_A$ and eigenvalues $\mathbf{\Psi}$ of the variance of $\boldsymbol{T}$ when E_{24} occurs as

$$\mathrm{Var}(\boldsymbol{T}\,|E_{24}) = \mathbf{V}_A\mathbf{\Psi}^2\mathbf{V}_A^T = \mathbf{\Lambda}^{-1}\mathbf{V}^T\mathbf{\Sigma}_{24}\mathbf{V}\mathbf{\Lambda}^{-1} \tag{4.188}$$

where the number of eigenvalues $m \leq n$, and $\mathbf{V}_A\mathbf{V}_A^T = \mathbf{I}$.

Define the random variable $\boldsymbol{Z}$ as

$$\boldsymbol{Z} = \mathbf{\Psi}^{-1}\mathbf{V}_A^T\boldsymbol{T} - \mathbf{c} \tag{4.189}$$

and the constant $\mathbf{c}$ as

$$\mathbf{c} = \mathbf{\Psi}^{-1}\mathbf{V}_A^T\mathbf{\Lambda}^{-1}\mathbf{V}^T\boldsymbol{\mu}_{24} \tag{4.190}$$

This results in

$$\boldsymbol{T} = \mathbf{V}_A\mathbf{\Psi}(\boldsymbol{Z} + \mathbf{c}) \tag{4.191}$$

and the mean and variance of $\boldsymbol{Z}$ being

$$\mathrm{E}(\boldsymbol{Z}) = \mathbf{0} \tag{4.192}$$

$$\mathrm{Var}(\boldsymbol{Z}) = \mathbf{I} \tag{4.193}$$

From (4.161)

$$\begin{aligned} Q &= g_Q(\boldsymbol{Y}) && (4.194)\\ &= ||\boldsymbol{T}||^2 && (4.195)\\ &= \boldsymbol{T}^T\boldsymbol{T} && (4.196)\\ &= (\boldsymbol{Z}+\mathbf{c})^T\boldsymbol{\Psi}\mathbf{V}_A^T\mathbf{V}_A\boldsymbol{\Psi}(\boldsymbol{Z}+\mathbf{c}) && (4.197)\\ &= (\boldsymbol{Z}+\mathbf{c})^T\boldsymbol{\Psi}^2(\boldsymbol{Z}+\mathbf{c}) && (4.198) \end{aligned}$$

(4.198) is in quadratic form.

Algorithm AS 106 from Sheil and O'Muircheartaigh (1977) can be used to find

$$F(U;\mathbf{c},\boldsymbol{\Psi},\boldsymbol{\mu},\boldsymbol{\Sigma}) = \Pr((\boldsymbol{Z}+\mathbf{c})^T\boldsymbol{\Psi}^2(\boldsymbol{Z}+\mathbf{c}) < U) \quad \text{where } \boldsymbol{Z}\overset{D}{=}\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma}) \quad (4.199)$$

for fixed vector c, symmetric semi-positive definite matrix $\boldsymbol{\Psi}$, fixed mean $\boldsymbol{\mu}$, variance $\boldsymbol{\Sigma}$, and upper limit U. Alternatively, algorithm AS 155 from Davies (1980) presents a method to solve an equivalent equation.

In this specific case

$$\begin{aligned} \rho_{24} &= \Pr((\boldsymbol{Z}+\mathbf{c})^T\boldsymbol{\Psi}^2(\boldsymbol{Z}+\mathbf{c}) < U_T) && (4.200)\\ \rho_{24} &= F(U_T;\mathbf{c},\boldsymbol{\Psi},\mathbf{0},\mathbf{I}) && (4.201) \end{aligned}$$

Algorithm AS 106 from Sheil and O'Muircheartaigh (1977) starts with an eigenvalue decomposition to reduce (4.199) to (4.201). Thus, the eigenvalue decomposition from (4.188) is needed.

This method can be used to calculate ρ_{14} and ρ_{23} in addition to ρ_{24} by simply replacing the relevant constants. This can be particularly helpful when newly built gauges are not properly handling rogue process and/or rogue measurement distributions.

4.4.3 Estimating the General Matrices

To make useful probability predictions with the above formulation, it is necessary to estimate $\mathbf{\Sigma}_{13}$ through $\mathbf{\Sigma}_{24}$. In many applications, a sample variance matrix can be built based on the observations from the accepted parts of an existing gauge. The sample variance matrix of the accepted parts is an approximation of $\mathbf{\Sigma}_{13}$. Limitations including the lack of available samples make it difficult to find $\mathbf{\Sigma}_{14}$, $\mathbf{\Sigma}_{23}$ and $\mathbf{\Sigma}_{24}$ in the same manner. Thus, a model-based approach that generates positive semi-definite variance matrices is needed.

The following approach worked well for the industrial applications used in this thesis. Define

$$Y = \begin{cases} \mathbf{G}_{13}\mathbf{H}_{13}\boldsymbol{S} & \text{when } E_{13} \text{ occurs} \\ \mathbf{G}_{23}\mathbf{H}_{23}\boldsymbol{S} + \boldsymbol{\mu}_{23} & \text{when } E_{23} \text{ occurs} \\ \mathbf{G}_{14}\mathbf{H}_{14}\boldsymbol{S} + \boldsymbol{\mu}_{14} & \text{when } E_{14} \text{ occurs} \\ \mathbf{G}_{24}\mathbf{H}_{24}\boldsymbol{S} + \boldsymbol{\mu}_{24} & \text{when } E_{24} \text{ occurs} \end{cases} \tag{4.202}$$

where

S is a standard normal random vector of length n, *i.e.* $\boldsymbol{S} \overset{D}{=} \mathcal{N}(\mathbf{0}, \mathbf{I})$.

$\mathbf{G}_{13}$ is a nonzero $N \times n$ matrix of rank n,

$\mathbf{G}_{14}$, $\mathbf{G}_{23}$, and $\mathbf{G}_{24}$ are nonzero $N \times n$ matrices.

$\mathbf{H}_{13}$, $\mathbf{H}_{14}$, $\mathbf{H}_{23}$, and $\mathbf{H}_{24}$ are nonsingular $n \times n$ matrices.

This results in

$$\mathbf{K}_{13} = \mathrm{E}(\boldsymbol{T}|E_{13}) = \mathbf{0} \tag{4.203}$$

$$\mathbf{K}_{ij} = \mathrm{E}(\boldsymbol{T}|E_{ij}) = \mathbf{\Psi}^{-1}\mathbf{V}^T\boldsymbol{\mu}_{ij} \quad \text{for } ij \in 14, 23, 24 \tag{4.204}$$

The variance of Y conditional on event E_{ij} is given by

$$\mathbf{\Sigma}_{13} = \mathrm{Var}(\boldsymbol{Y}|E_{ij}) = \mathbf{G}_{ij}\mathbf{H}_{ij}\mathbf{H}_{ij}^T\mathbf{G}_{ij}^T \tag{4.205}$$

and the variance of T conditional on event E_{ij} is given by

$$\begin{aligned} \mathrm{Var}(\boldsymbol{T}|E_{ij}) &= \mathbf{\Lambda}^{-1}\mathbf{V}^T\mathbf{\Sigma}_{ij}\mathbf{V}\mathbf{\Lambda}^{-1} && (4.206)\\ &= \mathbf{\Lambda}^{-1}\mathbf{V}^T\mathbf{G}_{ij}\mathbf{H}_{ij}\mathbf{H}_{ij}^T\mathbf{G}_{ij}^T\mathbf{V}\mathbf{\Lambda}^{-1} && (4.207) \end{aligned}$$

For an accurate model, choose G_{ij} and H_{ij} such that the resulting Y has characteristics that match the physical system being analyzed. For instance, many systems have a known frequency response curve. As such, G can be a frequency transform matrix, and H can be the scaled values the frequencies are expected to take. The frequency transform approach is shown in the next example in §4.4.4.

Alternatively, if a discrete time linear response function $r_{ij}(k)$ is known, then G_{ij} can be of the form

$$\mathbf{G}_{ij} = \begin{bmatrix} r(0) & r(1) & r(2) & \cdots & r(k) \\ 0 & r(0) & r(1) & \cdots & r((k-1)) \\ 0 & 0 & r(0) & \cdots & r(1) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & r(0) \end{bmatrix} \quad \text{where } r(kT) = r_{ij}(kT) \tag{4.208}$$

For each event E_{13} through E_{24}, different response functions can be used to build unique $G_{ij}(k)$ and H_{ij} matrices. If necessary, every line of G_{ij} can have its own unique response function.

The GH matrix approach can be extended to deal with applications where it is necessary to model process and measurement variation separately. Separate GH matrices can be used to define X and W, and these can be combined together to find Y. This technique is shown in §4.4.4.

4.4.4 Industrial Multivariate Example

To verify the computations through simulation, and to provide a good example, the following model was developed

$$\begin{aligned}
\boldsymbol{X} &\overset{D}{=} (1-\alpha)\mathcal{N}(\mathbf{0},\boldsymbol{\Sigma}_1) + \alpha\mathcal{N}(\boldsymbol{\mu}_2,\boldsymbol{\Sigma}_2) && (4.209)\\
\boldsymbol{W} &\overset{D}{=} (1-\beta)\mathcal{N}(\mathbf{0},\boldsymbol{\Sigma}_3) + \beta\mathcal{N}(\boldsymbol{\mu}_4,\boldsymbol{\Sigma}_4) && (4.210)\\
\boldsymbol{Y} &= \boldsymbol{X}+\boldsymbol{W} && (4.211)\\
\boldsymbol{Y} &\overset{D}{=} (1-\alpha)(1-\beta)\mathcal{N}(\mathbf{0},\boldsymbol{\Sigma}_{13}) + \alpha(1-\beta)\mathcal{N}(\boldsymbol{\mu}_2,\boldsymbol{\Sigma}_{23}) + \\
&\quad (1-\alpha)\beta\mathcal{N}(\boldsymbol{\mu}_4,\boldsymbol{\Sigma}_{14}) + \alpha\beta\mathcal{N}(\boldsymbol{\mu}_{24},\boldsymbol{\Sigma}_{24}) && (4.212)
\end{aligned}$$

For given constants a and σ_4, the means and variances can be computed from

$$\begin{aligned}
\boldsymbol{\mu}_2 &= \begin{bmatrix} a & \cdots & a \end{bmatrix}^T && (4.213)\\
\boldsymbol{\mu}_4 &= \begin{bmatrix} -Na & 0 & \cdots & 0 \end{bmatrix}^T && (4.214)\\
\boldsymbol{\mu}_{24} &= \boldsymbol{\mu}_2 + \boldsymbol{\mu}_4 && (4.215)\\
\boldsymbol{\Sigma}_1 &= \mathbf{G}\mathbf{H}_1\mathbf{H}_1\mathbf{G}^T && (4.216)\\
\boldsymbol{\Sigma}_2 &= \mathbf{G}\mathbf{H}_2\mathbf{H}_2\mathbf{G}^T && (4.217)\\
\boldsymbol{\Sigma}_3 &= \sigma_3^2\mathbf{I} && (4.218)\\
\boldsymbol{\Sigma}_4 &= \sigma_4^2\mathbf{I} && (4.219)\\
\boldsymbol{\Sigma}_{13} &= \boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_3 && (4.220)\\
\boldsymbol{\Sigma}_{14} &= \boldsymbol{\Sigma}_1 + \boldsymbol{\Sigma}_4 && (4.221)\\
\boldsymbol{\Sigma}_{23} &= \boldsymbol{\Sigma}_2 + \boldsymbol{\Sigma}_3 && (4.222)\\
\boldsymbol{\Sigma}_{24} &= \boldsymbol{\Sigma}_2 + \boldsymbol{\Sigma}_4 && (4.223)
\end{aligned}$$

where a is a constant. The constants $\sigma_3 = \sigma_4 = 1/6$ for this simulation.

For this example, the G matrix is based on the Discrete Cosine Transform (DCT), which is a frequency based metric, and the H_i matrices assume mass

damping

$$\mathbf{G} = \begin{bmatrix} g_{0,0} & \cdots & g_{0,N} \\ \vdots & \ddots & \vdots \\ g_{N,0} & \cdots & g_{N,N} \end{bmatrix} \tag{4.224}$$

$$g_{i,j} = \cos\left[\frac{\pi i}{N}\left(j+\frac{1}{2}\right)\right] \tag{4.225}$$

$$\mathbf{H_i} = \begin{bmatrix} h_{i,0} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & h_{i,N} \end{bmatrix} \tag{4.226}$$

$$h_{i,j} = \frac{f_i^2}{f_i^2 + j^2} \tag{4.227}$$

Constants f_1 and f_2 represent the corner frequencies of the transfer function, and they are $f_1 = 4$ and $f_2 = 0.4$ for this simulation. Assume $\alpha = 0.1$, $\beta = 0.01$, and $P_E^* = 10^{-9}$. This means a $\rho_{24} < 10^{-6}$ is desirable.

First, consider the case of a univariate gauge, by setting $g(\boldsymbol{Y})$ to the average function

$$g_A(\boldsymbol{Y}) = \frac{Y_1 + ... + Y_N}{N} \tag{4.228}$$

Assume a pass-zone of $|g_A(\boldsymbol{Y})| < U_A$, $U_A = \dfrac{a}{2}$. With this pass-zone, the mean of the combined rogue distribution $\boldsymbol{\mu}_{24}$ is inside the pass-zone, as $g(\boldsymbol{\mu}_{24}) = \mathbf{0}$. When the mean of the distribution is inside the pass-zone, the standard deviation σ_4 must be very large for the gauge to meet specification. For instance $\sigma_4 > 8 \cdot 10^5 U_A$ results in $\rho_{24} < 10^{-6}$. It is often difficult to construct gauges where σ_4 is this large, because of nonlinearities and saturation effects which tend to limit the maximum observable standard deviation. As such, it is rarely feasible to overcome rogue effects in univariate gauges by increasing the standard variation of the rogue events.

Now, consider the case of a multivariate gauge, with $g(\boldsymbol{Y}) = g_Q(\boldsymbol{Y})$. As per (4.161)

$$g_Q(\boldsymbol{Y}) = ||\boldsymbol{T}||^2 \quad \text{and} \quad \boldsymbol{T} = \boldsymbol{\Lambda}^{-1}\mathbf{V}^T\boldsymbol{Y} \tag{4.229}$$

where $\mathbf{\Lambda}$ and $\mathbf{V}$ are matrices containing the first n eigenvalues and eigenvectors of $\mathbf{\Sigma}_{13}$. Set the pass-zone to be $g_Q(\boldsymbol{Y}) < U_T$, where U_T meets the criteria

$$\rho_{13} = 0.9999 = F_{\chi^2}(U_T; n, 0) \tag{4.230}$$

The value of $\rho_{13} = 0.9999$ was selected to yield a large P_A, while still being numerically tractable. The resulting ρ_{23} and ρ_{24} conditional probabilities for given values of the constant a are shown in Figure 4.8.

It can be seen from the upper and lower halves of Figure 4.8 that large values of a result in very small ρ_{23} and ρ_{24} values. This is of practical significance. Many attribute univariate gauges exist where the main distribution and rogue distribution are separated by a values on the order of 6σ to 12σ, because that is the design recommendation from Six Sigma and good GR&R practice. However, if rogue distributions are present in these gauges, they may fail to meet their 1 ppb design goals. These gauges can be fixed by implementing a multivariate pass-zone.

It is not necessary to evaluate a large number of eigenvalues to achieve the effect shown in the top half of Figure 4.8. The lower half presents the graphs with only three active eigenvalues, *i.e.* when $n = m = 3$, in an application defined to have $N = 8$ eigenvalues. When dealing with variance matrices acquired from sampling processes, it may take very large numbers of samples for the smaller eigenvalues to converge and become meaningful. Gauges can be made to work using only the top few eigenvalues. For example, the third algorithm that is discussed in Chapter 9 uses the top 11 eigenvalues and eigenvectors to describe a 500 data point sample.

When dealing with a true-attribute problem like this, the multivariate pass-zone is surprisingly noncritical. A wide gap exists between the regions where the rogue distributions occur and the region where the main distribution occurs. Any suitable approximation of the $g_Q(\boldsymbol{Y}) \leq U_T$ ellipse works, because there is no gray zone caused by rogue effects.

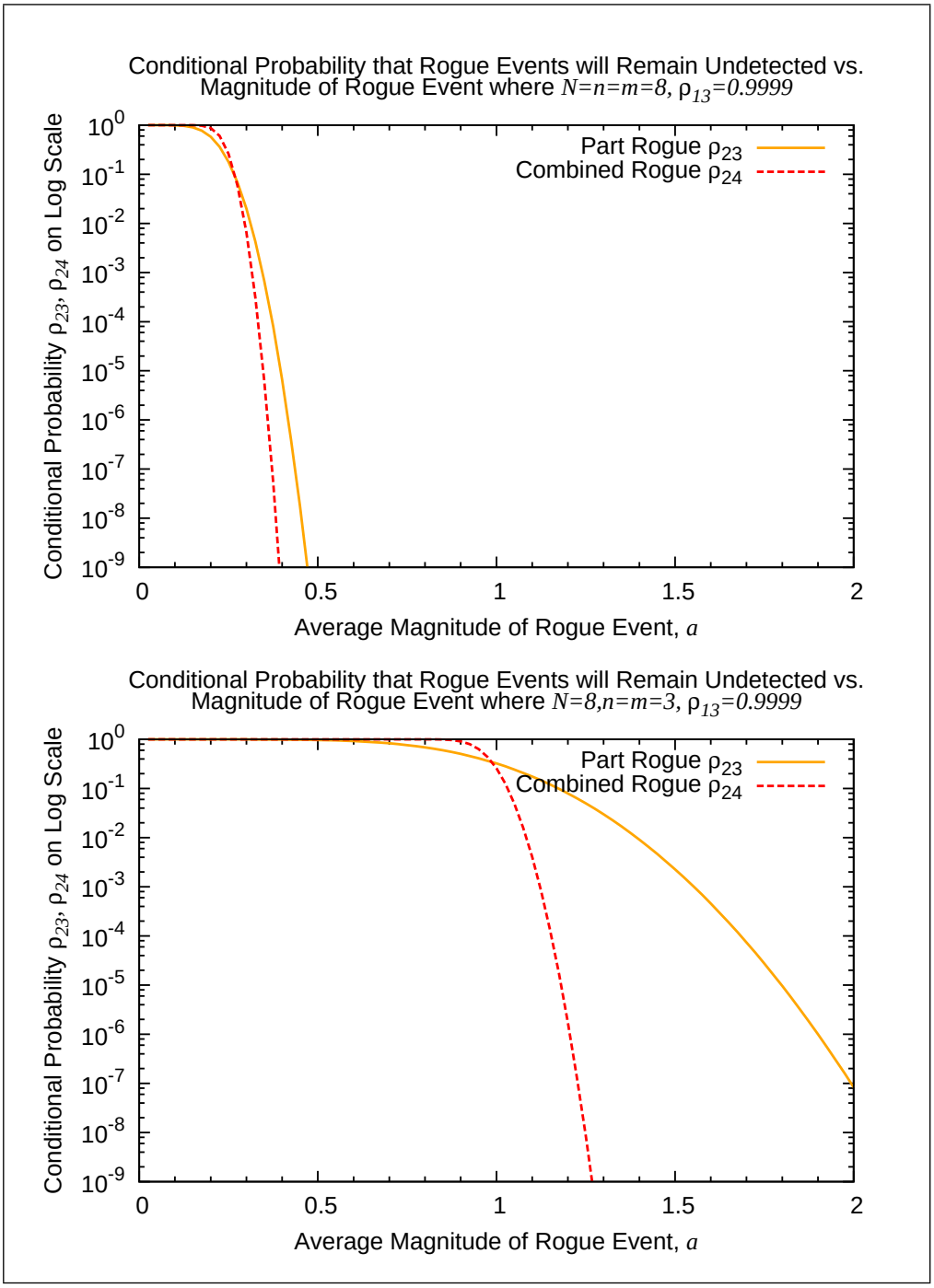

Figure 4.8: Conditional Probabilities in Sample Univariate Gauge versus Rogue Magnitude a

4.4.5 Finding an upper bound on ρ_{24} with a minimum $||Y_{24}||$

In the feedback application in §4.2.7, the random variable Y from (4.66) had a near-symmetric distribution, implying the mean of Y was near zero. A key feature of feedback systems is that they generate results with small or zero average error. This is a problem for high-reliability univariate gauging, because it is easier to detect errors with large means and small standard deviations.

Consider for the purposes of illustration, the special case where $||\boldsymbol{Y}|| > M^2$ for any given constant M^2. For sufficiently large M^2, $g_Q(\boldsymbol{Y}) > U_T$ in all cases, and $P_E = 0$. This special case allows for the average value of $\boldsymbol{Y}$ to be zero, while having nonzero Y. This special case is only useful for the purposes of illustration, because most real applications have gauge error. As such no constant exists where $||\boldsymbol{Y}|| > M^2$. Also, if $g_Q(\boldsymbol{Y}) > U_T$ in all cases, then both $P_A = 0$ and $P_E = 0$.

A more realistic practical model is as follows. Suppose either $\boldsymbol{\mu}_2$ or $\boldsymbol{\mu}_4$ or both are random variables. Define the distribution of the sum $\boldsymbol{\mu}_2 + \boldsymbol{\mu}_4$ as

$$\boldsymbol{\mu}_2 + \boldsymbol{\mu}_4 \overset{D}{=} \mathcal{R}_5(\boldsymbol{\mu}_5, \boldsymbol{\Sigma}_5) \tag{4.231}$$

where the distribution $\mathcal{R}_5$, mean $\boldsymbol{\mu}_5$ and variance $\boldsymbol{\Sigma}_5$ are the distributions of the sum $\boldsymbol{\mu}_2 + \boldsymbol{\mu}_4$. Define $\boldsymbol{K}_{24}$ as

$$\boldsymbol{K}_{24} = \boldsymbol{\Lambda}^{-1}\mathbf{V}^T(\boldsymbol{\mu}_2 + \boldsymbol{\mu}_4) \tag{4.232}$$

where $\boldsymbol{\Lambda}$ and $\mathbf{V}$ are the eigenvalues and eigenvectors from (4.162). Assume $\boldsymbol{K}_{24}$ meets the condition that

$$||\boldsymbol{K}_{24}||^2 \geq M^2 \tag{4.233}$$

for some constant M^2. This condition limits the values that the sum $\boldsymbol{\mu}_2 + \boldsymbol{\mu}_4$ can take. In particular, the sum $\boldsymbol{\mu}_2 + \boldsymbol{\mu}_4$ is not normally distributed.

The remaining random variables are defined in the usual way

$$\boldsymbol{X} \overset{D}{=} (1-\alpha)\,\mathcal{P}_1\left(\mathbf{0}, \boldsymbol{\Sigma}_1\right) + \alpha \mathcal{R}_2\left(\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2\right) \tag{4.234}$$

$$\boldsymbol{W} \overset{D}{=} (1-\beta)\,\mathcal{P}_3\left(\mathbf{0}, \boldsymbol{\Sigma}_3\right) + \beta \mathcal{R}_4\left(\boldsymbol{\mu}_4, \boldsymbol{\Sigma}_4\right) \tag{4.235}$$

$$\boldsymbol{Y} = \boldsymbol{X} + \boldsymbol{W} \tag{4.236}$$

Find the maximum ξ where

$$\xi = F(U_T; \boldsymbol{\Psi}^{-1}\mathbf{a}, \boldsymbol{\Psi}, \mathbf{0}, \mathbf{I}) \tag{4.237}$$

for any point a in the domain of the random variable K_{24}. $\boldsymbol{\Psi}$ is the eigenvalue from (4.188), and can be found from the decomposition of $\boldsymbol{\Sigma}_{24}$. The density function F is Sheil's algorithm AS 106 from (4.199).

From (4.200)

$$\rho_{24} = \Pr\left[(\boldsymbol{Z}^T + \mathbf{c}^T)\boldsymbol{\Psi}^2(\boldsymbol{Z} + \boldsymbol{\Psi}^{-1}) < U_T\right] \tag{4.238}$$

As ξ is a maximum

$$\xi > \rho_{24} \tag{4.239}$$

At first, it might not be obvious what industrial applications have this distribution. However, one can look for observations influenced by these effects

1. Feedback systems result in small average errors, but in only a finite number of directions (basis vectors). Mathematically, $\boldsymbol{\Sigma}_2$ has a finite set of basis vectors. For instance, feedback may only correct the average of all the inputs, yielding $\operatorname{rank}(\boldsymbol{\Sigma}_2) = 1$, and the lone basis vector $\frac{(1, \ldots 1)}{N}$. For any given $\boldsymbol{\Sigma}_2$, the basis vectors can be found from eigen decomposition.

2. The rogue measurements also occur in particular directions, and these directions are often different than the directions in which feedback occurs. Mathematically, the basis vectors of Σ_4 are different than the basis vectors of Σ_2. For instance, assume that measurement rogue events affect only one input, and all inputs are equivalent. Modeling all gauge error as variation error on the first input (since all inputs are equivalent) results in $\text{rank}(\Sigma_4) = 1$ and the lone basis vector is $(1, 0, ...0)$.

3. The rogue process errors and the rogue gauge errors of interest are large.

When Σ_2 and Σ_4 have different basis vectors, it can become impossible for sufficiently large rogue events to interact in the $g_Q(Y) \leq U_T$ pass-zone.

Many applications cannot achieve the above 3 conditions precisely, because even when rogue events are present, additional random effects are also present. Hence, the definition of the above system of equations in terms of the sum $\boldsymbol{\mu}_2 + \boldsymbol{\mu}_4$ instead of a simpler restraint on Y.

4.4.5.1 Industrial Application

The third algorithm from Chapter 9 meets the criteria described in the previous section. The rogue gauge error had a distinctive triangle shape, and each triangle was known to have a large magnitude. The rogue process error also had a distinctive shape, which could not be described by any simple combination of triangles, and had a different large magnitude. Through simulation, the point a at which the maximum ξ occurred was found. This resulted in $\xi = 4.5 \cdot 10^{-125}$ which means $\rho_{24} < \xi$ was sufficiently small, and the $P_E^* = 10^{-9} > P_E$ specification was met, regardless of α and β. Assuming $\alpha = 0.03$ and $\beta = 0.02$ results in $P_E < 2.7 \cdot 10^{-128}$. This result was verified a number of different ways, including by a 100 million part simulation that showed no outgoing defects, and on-line via a one million part sample run.

In application, the estimate of ξ can be checked by observing captured data on the combined rogue distribution, and confirming that no "near misses" exist that would question the validity of the assumed constraint from (4.231).

When the computed ρ_{24} on the known rogue distributions is extremely small, as in this case, it is necessary to consider the accuracy to which a prediction can be made.

4.4.6 Confidence Limits

The previous section described how to reduce outgoing defects to, for most practical purposes, immeasurably small levels. Unfortunately, this does not automatically mean the gauge is reliable to 10^{-9} reliability levels. Specifically, it is possible for the gauge to experience a rogue distribution that was not part of the model. If the dominant combined rogue distribution in the pass-zone is not in the model, then the resulting reliability prediction will be wrong.

Two methods exist for quantifying reliability in this situation. For each of these methods, the case of a rogue process event occurring for the first time is equivalent to the case of a rogue gauge event occurring for the first time. As such, it is assumed a rogue gauge event occurs for the first time, and derivations proceed accordingly.

4.4.6.1 Conditional Probability Method

Consider the case where a significant rogue gauge event E_4 occurs for the first time. This may result in

1. A rogue gauge event E_{14} occurring in isolation.

2. A detected combined rogue event where E_{24} occurs, and the observation $g_Q(\boldsymbol{Y}) > U_T$.

3. An outgoing defect event E_E, where the combined rogue event occurs E_{24} and the observation is in the pass-zone $g_Q(\boldsymbol{Y}) \leq U_T$.

From (4.157), the conditional probability of a combined rogue event causing an outgoing defect E_E is given by

$$\Pr(E_E|E_4) = \frac{P_E}{\beta} = \alpha\gamma_{24} \tag{4.240}$$

where

α is the probability of a rogue process event,

β is the probability of a rogue gauge event,

γ_{24} is the conditional probability that a combined rogue event causes an outgoing defect, and $\gamma_{24} \leq 1$.

This can be compared against the conditional probability that an outgoing defect will not occur E'_E.

$$\Pr(E'_E|E_4) = 1 - \alpha\gamma_{24} \tag{4.241}$$

These probabilities represent a substantial improvement over current practice, which will invariably result in the customer receiving reject production. The industrial applications have small values of $\alpha\gamma_{24}$. As such, if the runtime profiler is being used, the probability of detection of the rogue event is almost one. The runtime profiler will capture the rogue gauge event as an event where $g_Q(\boldsymbol{Y}) > U_T$. As this triggers a reject condition in the current implementation, the part will be quarantined and the customer will be safe. Additionally, if the rejects are analyzed, and the significance of the

new rogue gauge event appreciated, the gauge can be checked to ensure it is effective against this new type of reject. The customer is better protected against receiving reject parts with runtime profiling, than without.

If the case where a significant rogue process event E_2 occurs for the first time is considered, then the conditional probability of a combined rogue event causing an outgoing defect E_E is given by

$$\Pr(E_E|E_2) = \beta\gamma_{24} \tag{4.242}$$

For small $\beta\gamma_{24}$ the customer is protected in the same way as for the rogue gauge event.

This section examined the conditional probability that the next part measured will be an outgoing defect caused by a rogue gauge event or rogue process event of a type not previously observed. The next section assumes no clustering and computes the probability estimate (not the conditional probability estimate).

4.4.6.2 Confidence Limits and Rogue Events not Previously Detected

Assume no rogue gauge events E_4 were detected in N_P parts, and no clustering occurs. In this case, a binomial proportion confidence limit applies. Thus, the method from Clopper and Pearson (1934) is used to compute the upper limit of the 1-sided $1-\vartheta$ confidence interval given no events have occurred in N_P Bernoulli trials. For a confidence interval of $1-\vartheta$, the upper confidence limit on β is given by

$$\beta \leq 1 - \left(\frac{\vartheta}{2}\right)^{\frac{1}{N_P}} \tag{4.243}$$

$$\tag{4.244}$$

For $N_P \gg -\ln\frac{\vartheta}{2}$, the following approximation can be computed using the Taylor series expansion of the exponential function

$$1-\left(\frac{\vartheta}{2}\right)^{\frac{1}{N_P}} \approx \frac{-\ln\frac{\vartheta}{2}}{N_P} \tag{4.245}$$

$$\tag{4.246}$$

Assuming the above approximation and $\vartheta = 0.05$ for a 95% confidence interval

$$\beta < \frac{3.7}{N_P} \tag{4.247}$$

It is a given that $\gamma_{24} \leq 1$, as all probabilities must be less than or equal to one. (4.157) yields

$$P_E < \frac{3.7\alpha}{N_P} \tag{4.248}$$

with a 95% confidence where α is the known rate of rogue process events.

The practical interpretation of this is that once

$$N_P > \frac{3.7\alpha}{P_E^*} \tag{4.249}$$

then the gauge can be assumed to meet the $P_E < P_E^*$ reliability specification for rogue gauge events. Similarly, considering the possibility of unknown process events, once

$$N_P > \frac{3.7\beta}{P_E^*} \tag{4.250}$$

then the gauge can be assumed to meet the $P_E < P_E^*$ reliability specification for rogue process events. If both the above criteria are met, the gauge can be assumed to meet the $P_E < P_E^*$ reliability specification.

The limitation of the above formulas lies in the assumptions. They assume no clustering, suitably chosen α and β, and 95% confidence limits. From an industrial point of view, machinery can wear, and this can cause

clusters of rogue events. As such, it is likely necessary to continue runtime profiling throughout the life cycle of the application.

4.4.6.3 Industrial Application - Confidence Limits

Using the method from §4.4.6 and (4.248), assuming an $\alpha = 0.03$ and $N_P = 10^6$ observations, then $P_E < 1.1 \cdot 10^{-7}$. Thus the prediction is that the third algorithm from Chapter 9 will encounter problems from unexpected rogue distributions $P_E < 1.1 \cdot 10^{-7}$ long before it encounters a combined rogue event with $\xi = 4.5 \cdot 10^{-125}$, $\alpha = 0.03$, $\beta = 0.02$, and a predicted $P_E = 2.7 \cdot 10^{-128}$.

This is the typical result. If a reliable gauging algorithm is chosen, the confidence limit becomes the practical upper bound on gauge performance.

Review of the million observations gathered revealed that the third algorithm from Chapter 9 found a new and unexpected type of rogue process event (part defect) inside its one million part sample run. This new type of rogue process event was of significant concern, because if the rogue event was undetected it would cause an outgoing defect and significant customer issues. However, this rogue event did not affect the reliability estimation results, as the gauge was sufficiently effective at detecting this new defect mode. It was an example of a case where the gauge algorithm "got lucky", and was effective against problems that were not supposed to happen.

4.5 Discussion - Rogue Event Analysis

4.5.1 Rational

This thesis was inspired by industrial applications. Engineers were good at being able to define the main production processes and at making sure the

main distribution was acceptable from a quality perspective. Similarly, the software programmers were good at making sure the accept path of the software was reliable. However, both groups were poor at accurately predicting the specific rates of rogue events present in an unfamiliar production process, and the predictions grew worse the more unlikely the event was. Thus, combined rogue events that had never been detected before were difficult to predict accurately.

It is vital to have accurate predictions of rogue events for the appropriate control algorithms to be selected for industrial applications. The runtime profiler, that is covered in the next chapter, is very effective at capturing execution information from programs. The $g_Q(Y)$ score function allows the profiler to acquire information automatically about rogue events outside the pass-zone of the gauge. Given this information, a statistical method is needed to predict concentrations of rogue events inside the pass-zone. This is the problem solved in this chapter, and the results are summarized here.

4.5.2 Rogue Event Analysis

The mixture models developed in this chapter show that it is possible to extrapolate from rogue distributions that are detected outside of the pass-zone to the combined rogue distributions that occur inside the pass-zone. As shown in §4.3.5, practical applications can have many rogue distributions outside the pass-zone. However, as discussed in §4.3.5.5 this can still be reduced to a simple approximation of what is happening inside the pass-zone.

Not all the interesting activity takes place inside the pass-zone. For instance, in the feedback application from §4.2.7, the majority of waste is being generated outside the pass-zone. This waste is straightforward to profile. Chapters 6.3 to 8 will discuss tweaking algorithms to reduce waste in the feedback cell.

Not all rogue event problems can be solved with software. The impact of rogue events should be considered before major capital investments are done in new automated cells. Once the investment is made, then the goal is to achieve a successful startup in the shortest amount of time possible. It is expensive to staff new installations with skilled on-site personnel, simply to deal with the contingency that an unexpected bug may happen. Nevertheless, this is frequently done.

With the analysis presented in this chapter and captured data from the runtime profiler, it is now possible to make reasonably accurate predictions about the impact of rogue events on different software algorithms. This makes off-line simulations possible, and software can be optimized much more quickly with less reliance on on-site coverage.

4.5.3 Technique

When implementing Rogue Event Analysis in new applications, start with identifying the main distribution. With either univariate or multivariate methods, use the method from §4.3.5.1 to determine an appropriate pass-zone. Everything inside that pass-zone is expected based on assuming the normal distribution and an appropriate ρ_{13}.

Anecdotal experiences with the automotive part suppliers suggested that customer complaints usually focus on easily identifiable problems. For a capable process, these problems will first be detected as rogue measurement and rogue process distributions occurring outside the pass-zone. There are many times fewer rogue parts in the rogue distributions than in the main distribution, assuming $\alpha + \beta < 0.05$ which often occurs. When sufficient profiling information is available, this means it is much easier to determine

the underlying causes of the rogue measurement and rogue process distributions, and to estimate their frequencies, means, variances and distributions, appropriately. Significant rogue defects often occur for obvious reasons, whereas the main distribution is often affected by the accumulation of many individual small process error sources that are difficult to identify individually.

Once the nature of the rogue measurement and rogue process distributions is determined, then the combined rogue distributions can be estimated. The dominant distributions inside the pass-zone can be estimated using the method from §4.3.5.4. While many rogue events may be present, most problems come down to a single combined rogue distribution in the pass-zone. An example of this is given in §4.3.5.5.

Once the rogue distributions are understood sufficiently well, they can often be controlled. The goal is to select optimal algorithms and parameters, such that the combined rogue distributions are outside the pass-zone. When the combined rogue distributions are outside the pass-zone, a gauge can have a large pass-zone with no outgoing defects. This is the ideal characteristic for a high-reliability production gauge.

Some attention must be given to clusters of rogue events. A cluster frequently occurs when machinery is broken or a common cause event is happening in the cell. This breaks the independence between X and W, and means that outgoing defects are no longer a function of $\alpha\beta\gamma_{24}$ and are instead a simple function of α and a malfunction-specific γ. As such, the primary function of the reliability formulas presented is to find the first instance of any specific defect cluster. Clusters often occur when automated feedback systems are present in the cell.

As discussed in §4.2, feedback systems can cause problems because of their tendency to create distributions with zero means. This is good in the case of the main distribution, but can make the rogue distributions more

difficult to detect. §4.4.5 and §4.4.5.1 suggest a practical way around this problem. Alternatively, if a rogue gauge event has a sufficiently large standard deviation, it is always detectable given a sufficiently large number of observations.

4.5.4 What to do about rogue events in the pass-zone?

With modern programmable machinery, many rogue event distributions can be modified in software. However, some gauge applications are irreparable. It is not advisable to reduce tolerance limits and accept high levels of waste in situations such as the situation modeled in §4.3.3. Even in the limit, reducing tolerances does not prevent reject product from reaching the customers in some situations.

Feedback can cause situations like the one from §4.3.3, because many feedback systems tend to ensure that the mean process error and/or the mean gauge error are close to zero. Thus, the impact of the feedback system on rogue events should be considered.

Many gauges already gather large numbers of inputs and consolidate the results into a single output variable. Camera-based machine-vision systems, as in the one in Chapter 9, are excellent applications for multivariate analysis. Other gauge types include coordinate measuring machines (CMM) and acoustic gauges. With falling technology prices, it is much cheaper to deploy a multivariate technique than has historically been the case.

4.6 Conclusion

The impact of rogue events on a production process can be modeled using mixture distributions for the process error and the gauge error. The presence of rogue distributions significantly affects the performance of the manufacturing cell, whether measured as waste in the feedback application, or

as outgoing defects in the high-reliability gauging application. It is recommended that rogue events be analyzed as soon as relevant data is available. The runtime profiler, described in the next chapter, proved effective at gathering information on rogue events.

Chapter 5

Runtime Profiler

5.1 Introduction

Chapters 6 through 8 detail a feedback application that was challenging to debug. It was used to build the list of requirements for the runtime profiling approach that was first tested in the industrial application from Chapter 9. The initial requirement list was

1. $\mathrm{O}(1)$ execution times of the profiler.[1] It is desirable to be able to profile any execution of the program, no matter how complicated the looping behaviour inside the program. A profiler with an $\mathrm{O}(1)$ execution time can be guaranteed to complete within a fixed amount of executive time. As such, a sufficiently powerful computer will be able to complete all profiling activities within the time it takes for the mechanical system to deliver a new part to the gauge. By contrast, an $\mathrm{O}(n)$ profiler may be able to execute in less time on average, however it offers no guarantees on execution time if the application being profiled does an inordinate amount of looping. In this application, $\mathrm{O}(1)$ execution times

[1] Big O notation was explained in §3.1.10.

were achieved by implementing Branch Count Sequence (BCS) technique from Harrold *et al.* (1998). This counted the number of passes through any given code block. The counts were kept in a master list, and this master list was saved once for each part delivered to the gauge for inspection.

2. All inputs, outputs, and critical program variables needed to be recorded. This was implemented by saving key structures to disk once for each part observed.

3. Data Compression. It was known from the feedback application that the data logging files would be huge. Therefore, data compression was needed so large runs of part data could be stored cost effectively (on a single hard drive).

In application, the primary performance limit of the profiler would be the long-term data storage capability of the disk drive. 100 GB of disk storage is spread over 10 million parts, then each data record must be approximately 10 kB (or less). While technological advancements since the initial application have eased this problem, the disk limit continues to be the primary issue in recording large amounts of information. Data compression reduces the size of the records from items 1 and 2, and this makes runtime profiling the complete BCS results feasible.

The BCS checks were effective at detecting coding errors along infrequently executed code paths that corresponded to reject conditions of the gauge. In particular, when designing a complex inspection system, it is common to make many assumptions about the incoming data (observations). If any of these assumptions are violated, it is common to reject the part out of an abundance of caution. This can result in a large number of unnecessary aberrant rejects, when it is discovered that acceptable parts are being rejected because actual production does not follow initial assumptions. When

the runtime profiler was used, it was possible to reduce the rate of aberrant rejects because statistical Q-scores could be cross-referenced against program decisions.

Despite the BCS algorithms success at reducing aberrant rejects, *the surprising result was that BCS code checks were ineffective at detecting outgoing defects in the industrial application* from Chapter 9. This was demonstrated through the off-line use of redundant algorithms that used the recorded input data to verify the output data. The gauging professionals that wrote the inspection algorithms were very careful to ensure that only values of Y inside the $|Y| < U_T$ pass-zone would lead to an accept. This lead to the discovery of the rogue distributions and the combined rogue distributions discussed in Chapter 4. Combined rogue distributions can have observations inside $|Y| \leq U_T$ pass-zone, even though $|X| > U_X$.

It is not obvious if combined rogue distributions are programming errors in the conventional sense. In particular, the software was coded to the programmers understanding of the problem. If the programmer is making assumptions about the incoming data, and not rejecting parts when the assumptions are violated, then outgoing defects may result. The experienced gauge designer knows that software cannot validate every possible assumption. For instance, combined rogue distributions have the interesting property that they cannot be easily detected in software, but different algorithms will experience more outgoing defects than others. In particular, as discussed in §4.3, multivariate gauge algorithms perform better than univariate gauge algorithms. Additionally, as Chapter 9 demonstrates, using multivariate gauge algorithms can solve the combined rogue distribution problem. As this thesis demonstrates, software profiling tools helpfully collect the data required to understand these effects.

5.2 Runtime Profiler with SPC

The vision system from Chapter 9 was capable of gathering large amounts of information per part. The captured data that related to an application-specific graph was control charted using a multivariate Q-chart. This lead to the discovery that a multivariate Q-chart is effective at detecting combined rogue distributions.

Thus the final version of the runtime profiler implemented the following four features

1. BCS profiling with $\mathrm{O}(1)$ execution time. This was effective in detecting many coding errors in the real-time gauge, feedback, display, and control algorithms, but not the problems that led to outgoing defects in this application.

2. Multivariate Q-charting using the $g_Q(Y)$ score function, as discussed in §4.4.1.2.

3. Inputs, outputs, and critical program variables are captured. This captured data can be used to test new and/or different algorithms to double-check the results and to select the best available algorithm for any given application.

4. Data Compression, to reduce the size of the output image on disk.

A key insight gained from implementing the system was that improving the algorithm can reduce total rejects, thus covering the profiling and hardware costs.

5.2.1 Execution Time

It was possible to implement all four of these features with reasonable execution times. The system specification was one part inspected every 500 ms. The original system inspected one part in 250 (ms). With the profiling library, the system inspected a part in 10 ms or less, and the majority of the time was spent compressing the outgoing log files and writing them to disk. The system without the profiling library could inspect a part in 1 (ms) or less. These numbers were achieved both with a Pentium 4-M single core portable PC and an AMD Athlon X2/4200 dual core PC with a single-threaded program.

5.2.2 Tracing a Specific Anomaly

Automated systems often run seven days a week and twenty four hours per day. The equivalent shift-coverage requires many people and significant man-hours to find and remove infrequently occurring software bugs. This represents a significant cost in commissioning new systems. The runtime profiler solves this problem, by allowing debugging information to be captured automatically and then analyzed later.

In Chapter 4 it became obvious that conventional graphs had issues when dealing with infrequent events. Importantly, if a logarithmic-scale was not used on the ordinate axis, then infrequent events could be obscured by the process of graphing. Additionally, it was also necessary to display many variables on the same graph when dealing with problems that could have a Y-vector with 500 elements. Lastly, it was also necessary to be able to display data from 10,000 to 1,000,000 parts on the same graph, and as such it was impossible to dedicate either a unique label or a unique pixel to each part. This meant a new approach to visualizing the data was needed, as conventional graphs would be inadequate for the application.

The new visualization approach is shown by the annotated screen captures given in Figures 5.1 through 5.4. Each screen capture shows one of the four key steps of the debugging process using the new run-time profiler that was constructed in the development of this thesis. These four key steps are described in the next four sections.

5.2.2.1 Step 1: Review the Score Function

As shown in Figure 5.1, the overall Q-Score is reviewed using the Red, Yellow, and Green intensity-based colour scheme. The intensity of the colours is logarithmically weighted, such that any occurrence of any part at that particular value stands out. The logarithmic weighting is used because, as explained in Chapter 4, linear scaling tends to obscure infrequent events. The extremely bright regions indicate that large percentages of the population occur with that particular value. The red zones correspond to reject (misoriented) product, the green zones correspond to correctly oriented product, and yellow zones correspond to cases where both occur.

In the Figure 5.1, three errant observations show as yellow lines in the reject zone. At these Q-scores, these errant observations correpond to extreme outliers, and represent potential outgoing defects. Assuming normally distributed data, the probability of these outliers occurring by random chance is less than 10^{-30}. Given that three outliers occurred, it can be safely assumed that the cause was not random chance. Therefore, further investigation is needed.

In the screen capture shown in Figure 5.1, blue is the background colour. Dark grey indicates a pass-zone. White marks values from the part under current investigation.

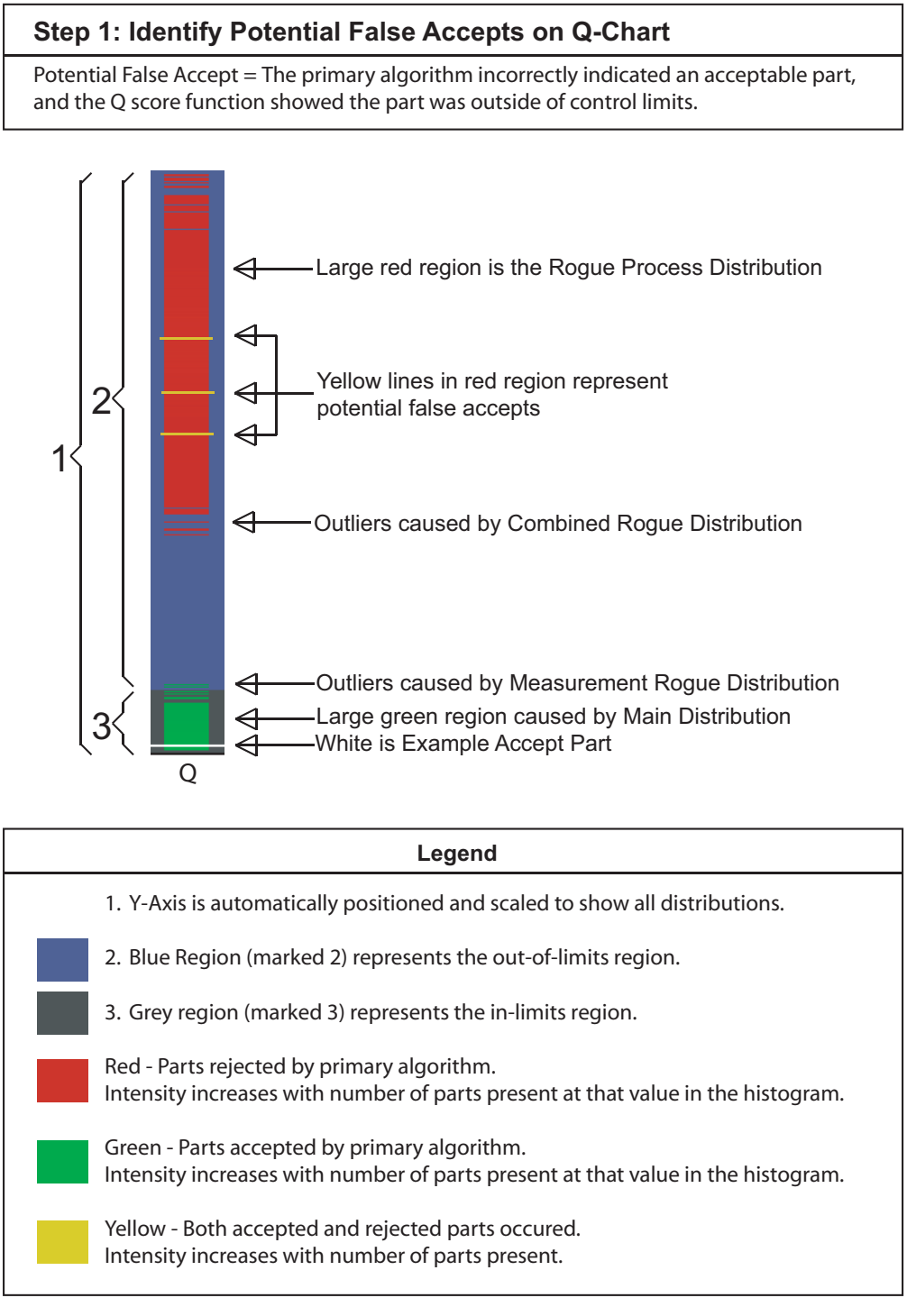

Figure 5.1: Debugging Using Runtime Profiler Step 1. This screen capture reflects the Q-Score.

5.2.2.2 Step 2: Check the *T*-vector

Drilling down, the T-vector is shown in Figure 5.2. This graph shows the individual T-values. Of particular importance is T_4, the fourth element of the $\boldsymbol{T} = (T_1, ...T_{11})$. It is the primary orientation detecting bar, and it clearly shows that the springs of concern are misoriented.

5.2.2.3 Step 3: Check the Observations

The screen to review the raw observations is shown in Figure 5.3. Reviewing the raw pitch-profile chart confirms that the springs are incorrectly oriented, with three yellow traces clearly in the red region. In this case, reject input data leading to an accept result is a key indication of a problem (bug).

5.2.2.4 Step 4: Check the Program Trace

The fourth and final step in the analysis is to review the actual BCS program trace, which is shown in Figure 5.4. This screen capture has particularly complex scaling, in order to cope with the vast number of possible statements that could be executed in the program, and the vast number of times each statement could be executed. Each column in the abscissa represents a group of statements in the source code, arranged in order of execution. Groups of statements that neither execute nor are intended to execute are omitted from the graph. Groups of non-executing statements can be readily determined by examining previous programs in an automated way. Omitting these extraneous statements simplifies the graph enormously.

In Figure 5.4, the ordinate is nonlinearly scaled, such that any value inside the dark grey (center) zone represents a valid frequency of execution. The lower dark blue zone represents a block of statements that executed too few times. The upper dark blue zone represents a block of statements that executed too many times.

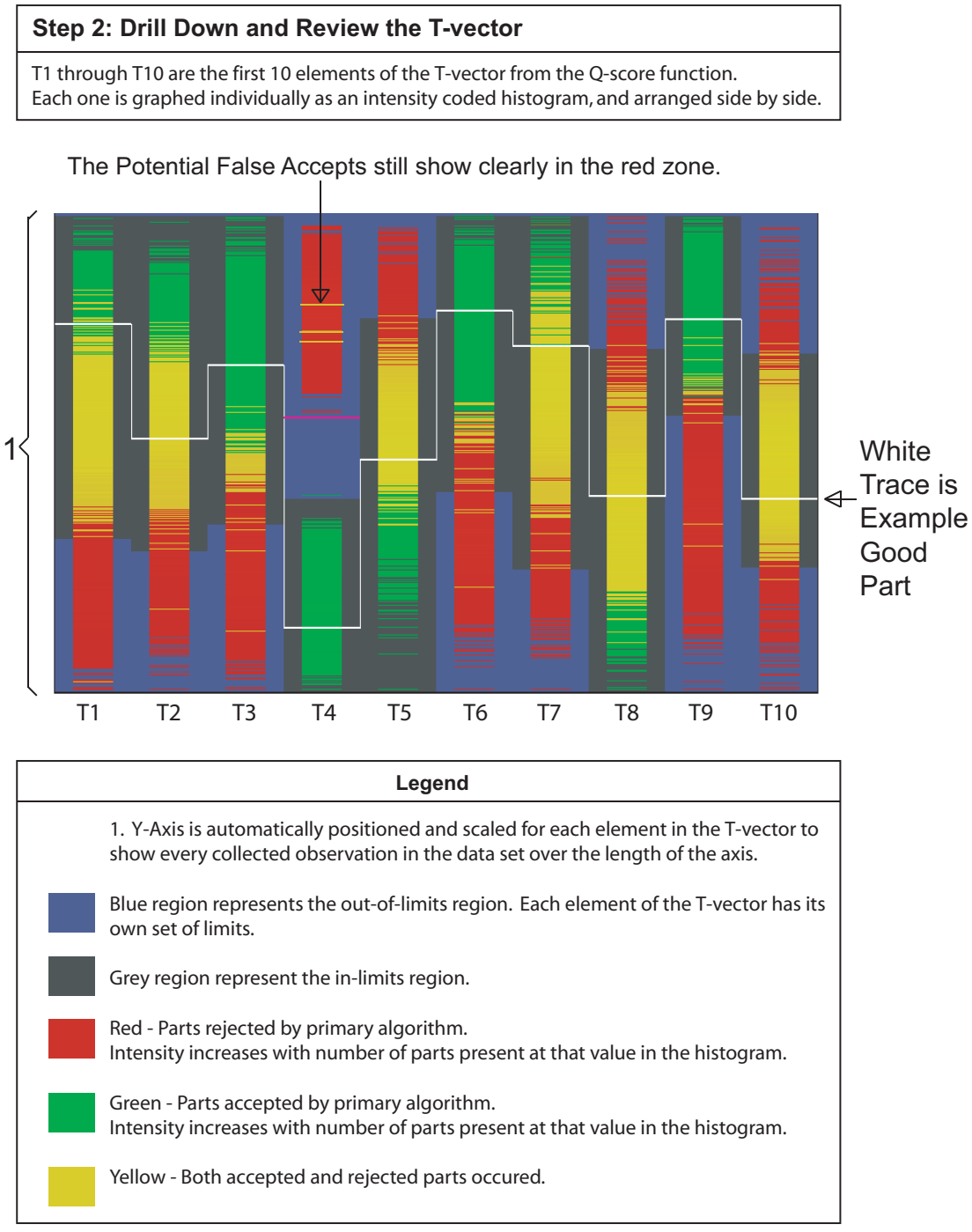

Figure 5.2: Debugging Using Runtime Profiler Step 2. This screen capture reflects the elements of the T-vector.

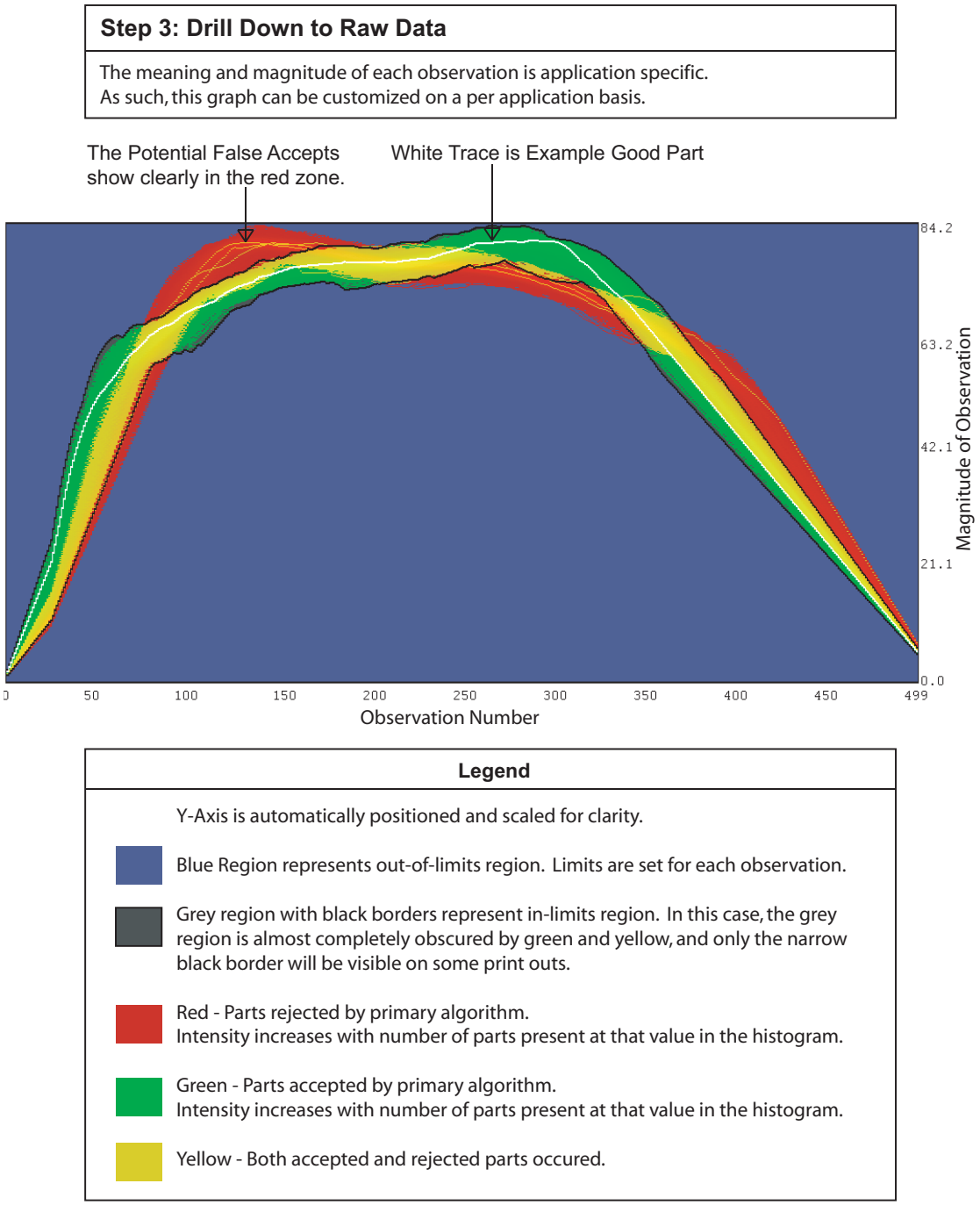

Figure 5.3: Debugging Using Runtime Profiler Step 3. This screen capture shows the elements of the observation Y-vector.

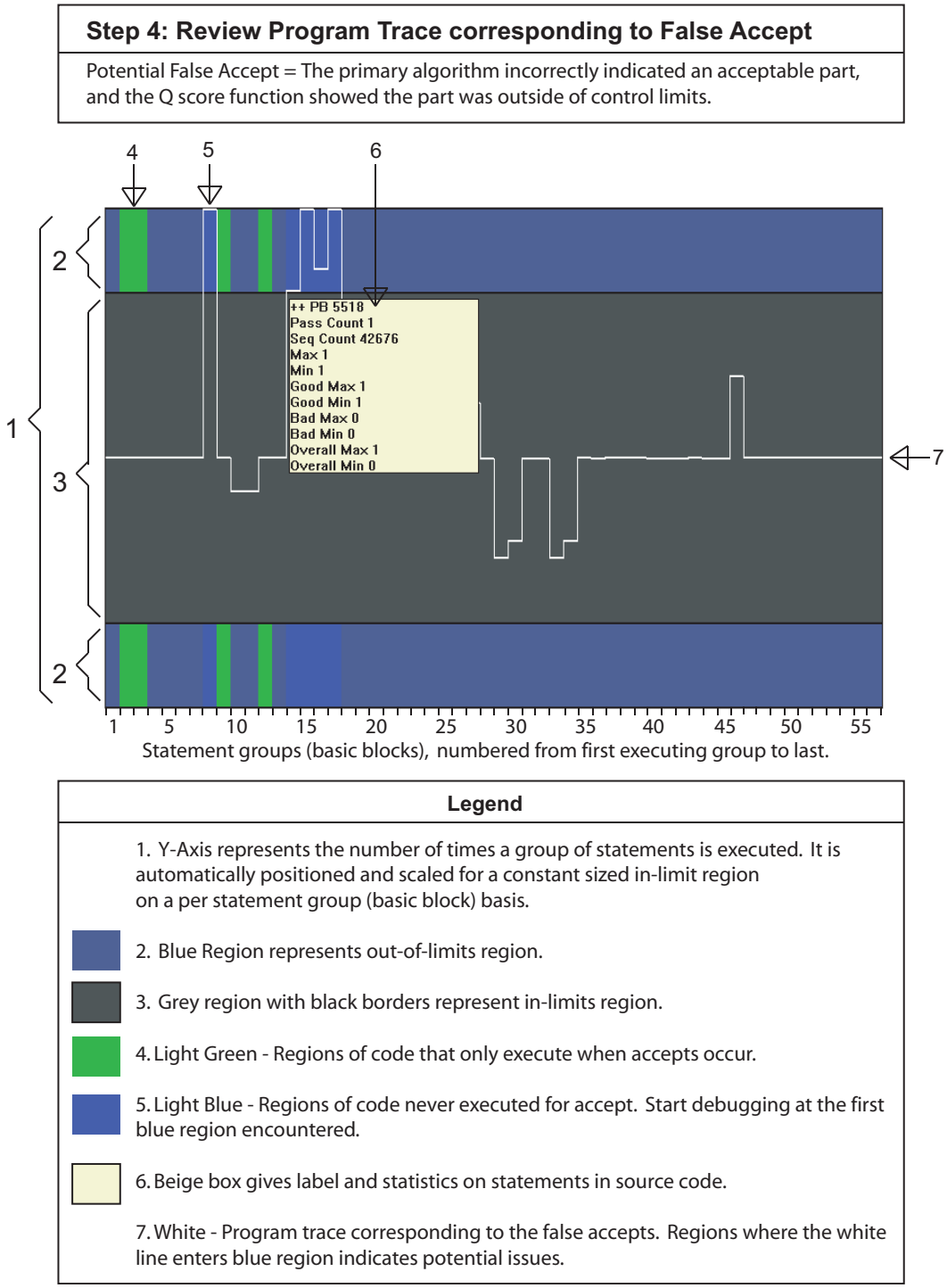

Figure 5.4: Debugging Using Runtime Profiler Step 4. This screen capture displays the BCS trace data.

By comparing the rates of occurrence of each zone against the preset limits for each group of statements (basic block), the incorrectly executing statements can be quickly identified. The information shown in the beige box in Figure 5.4 uniquely identifies each group of statements in the source code. Tracing a malfunctioning group of statements will often show the error. Alternatively, the inspection program can be rerun in a debug mode with the captured data, allowing a full program debug.

5.2.2.5 Insights Gained from the New Runtime Profiler

Significant economic gains were made in the Spring Orientation algorithm by tracing cases with the yellow lines in the green zone. These correspond to aberrant rejects: cases where the Q-score function from §4.4.1.2 indicated the part was correctly oriented and the pattern algorithm disagreed. A great many aberrant reject bugs were located in this manner. Reducing the aberrant rejects in this manner resulted in the trace shown in Figure 5.1, which shows a green pass-zone without any aberrant rejects - an excellent result.

Problems relating to outgoing defects were not detected as changes in the BCS program trace. In particular, issues relating to combined rogue distributions in the pass-zone typically are not detectable by BCS methods, and the methods from Chapter 4 must be used.

Unclear pass-zones often show up as yellow overlap regions between the green and red zones. For an example of this, see the T_1, T_2, and T_3 columns from Figure 5.2. Individually, these do not have sufficient statistical capability for orientation detection. However, collectively, especially when T_4 is present in conjunction with T_2 and T_5, the required statistical capability level is present.

Using the new runtime profiler resulted in much quicker detection and problem resolution of program anomalies, especially when the anomalies are occurring infrequently in high-speed systems.

5.3 Conclusions

The major discovery in this thesis was that the outgoing defects were not being caused by conventional coding “bugs”, but were actually being caused by combined rogue distributions. Instances where this problem occurs can be spotted by using a runtime profiler of the type described that implements the relevant score functions. This can be implemented feasibly, in industrially acceptable execution times. The cost of the extra hardware and software is easily offset by reduced commissioning times and reduced outgoing defects.

Chapter 6

Introduction to Industrial Applications

6.1 Overview

The fundamental point of this thesis is that rogue events can dominate the operational costs of practical industrial processes and, by reducing the impact of the rogue events, costs can be significantly reduced. As this work was done with real industrial processes, it was not possible to demonstrate all of this technology inside one practical industrial manufacturing system.

The approach taken in this thesis is to use three different industrial applications:

Chapter 7 focuses on the feedback loop in a CNC machining cell with CMM feedback. This is an industrially significant system and the effects observed were not covered in previous literature.

Chapter 8 focuses on the material transport algorithms in the CNC machining cell with feedback. The goal is to minimizing delay and maximizing the rate at which feedback loop updates can be performed, given limited measurement capacity. This maximizes feedback performance and reduces waste. Additionally, optimizing the material transport algorithms proved vital to cell success.

Chapter 9 focuses on a high-reliability attribute gauging application. High-reliability systems pose a different set of challenges as feedback systems, as shown in this application.

Chapters 7 and 8 focus on a particular class of manufacturing cell, and the background to this cell will be introduced in this chapter. To a large extent, much of this introductory material will be relevant to Chapter 9. Chapter 9 will cover the implementation specific details for the high-reliability attribute gauging application separately.

The development of a new class of Computer Numerically Controlled (CNC) manufacturing cell with Coordinate Measuring Machine (CMM) feedback was motivated by similar requirements from a number of automotive part suppliers. Automotive part suppliers are under tremendous pressure to reduce costs to remain competitive with both local and overseas competition. As such, four different plants independently made efforts to implement CNC manufacturing cells with CMM feedback. These cells, if they worked, had decisive cost advantages, that would keep the plants competitive. These manufacturing cells encountered significant problems. In fact, only two of the four plants succeeded in implementing CNC manufacturing with CMM feedback. Implementing the concepts and lessons discussed in Chapters 7 and 8 proved crucial for cell success, and only the cells that implemented these concepts succeeded.

6.2 Manufacturing Cell with Feedback

Figure 6.1 shows the cell structure under analysis. Each CNC dual spindle lathe has two live turrets capable of both milling and turning. The dual spindle feature allows a single machine to produce a finished part that requires machining on all surfaces.

In production, it is customary to sequentially label major part processing operations OP 10, OP 20, etc. Minor part processing operations are labeled by increasing one counts. Thus. the minor operations from OP10 are labeled OP 11, OP 12, and so on, up to OP 14. Major parallel operations inside the same machine are designated with a five code. For instance, a parallel operation at OP 10 would be labeled OP 15, and the corresponding minor parallel operations are designated OP 16 through OP 19. When two or more different types of parts are manufactured inside the cell, each part process has its own OP 10, OP 15 production process. The cell is responsible for making sure the parts from each manufacturing process are separately tracked, identified, packed, and shipped.

In this case, the dual spindle lathe holds two parts, one in each spindle, with the main spindle performing OP 10 on the first part while the subspindle performs OP 15 on the second part. The result is that the two-stage machining process can produce a new manufactured part every cycle. With every CNC in the cell in operation, n_{CNC} complete parts are produced every T_{CYC} seconds. The parts are complete when they leave the CNC machines, and require no further manufacturing operations. This makes the cell flexible, as no requirement exists for all the CNC machines to produce identical parts.

The cell is able to implement feedback in three different ways:

Automatic: referring to Figure 6.1, in automatic mode the parts are delivered to the CMM through the material transport system, and offsets

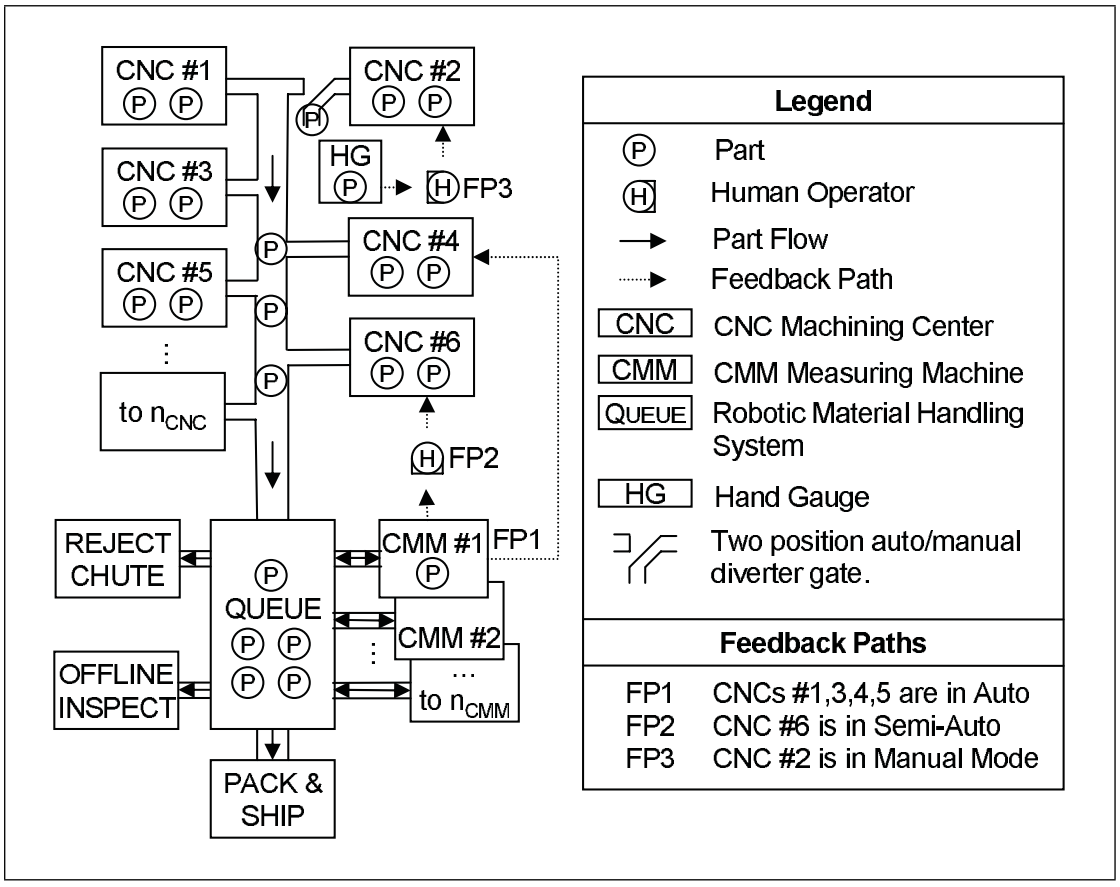

Figure 6.1: Cell Layout Diagram

are electronically sent to the CNC machines along feedback path FP1 under computer control.

Semiautomatic: the parts are delivered to the automatic CMM, where the operator reads them, and manually adjusts the CNC wear offsets following feedback path FP2.

Manual: the operator H manually grabs the parts, and measures them with the hand gauge HG. If adjustments are required, they are manually input to the CNC following feedback path FP3. The operator may place the part back on the production line, where it will follow the automatic process. When the operator places manually measured product on the automated production line, the feedback control system is responsible for making sure that an automatic update will not duplicate a manual update already performed. Manual measurements with hand gauges are less accurate than the automatic measurements from the CMM, and are not reported to the computer system as observations. As such, the manual measurements introduce many issues with gauge repeatability and reproducibility, and process accuracy. However, manual measurements have the advantage that they allow the operator to setup new tooling much quicker than is possible with the automatic process.

The cell operator is able to make feedback adjustments in the cell at any time, for any reason, and is able to access fresh production from the CNC at any time. This capability is very important to cell operation.

From a feedback perspective, the auto/manual capability has the disadvantage that operator involvement affects feedback loop stability and performance. Manual involvement creates the possibility that the feedback loop will become sufficiently inaccurate that large volumes of scrap can be created. §4.2.5 discussed the influence of operator intervention and conveyor system delays on the stability of the feedback loop.

Nevertheless, field trials indicated that the cells only worked if they allowed the operator easy access to finished parts. The CNC machining process is complex and fraught with numerous error sources that can result in out of tolerance parts being produced. These error sources include tool wear, thermal error, workpiece clamping, vibration, and workpiece and material issues. It is extremely important to allow the operator to correct these failures in an appropriate manner as quickly as possible. In practice, this means that methods must be provided to allow for quick and easy transitions between automatic and manual mode and back again, such that as much time as possible can be spent in automatic mode.

With many parts being produced simultaneously, the route of any individual part through the cell can easily become obscured. Figure 6.2 is a part routing diagram. The part routing diagram shows the part-path corresponding to a single part. Parts are produced simultaneously on all the CNCs in the cell. As such, many parts are progressing through Figure 6.2 simultaneously and in parallel.

Following Figure 6.2 the parts are made in the CNC in operations OP10 and OP15. In automatic mode, parts needing to be observed (measured) are sent to the CMM through path AB. After the part is observed by the CMM, the part can be packed and shipped to the customer (Route CE), additional inspections may be required (Route CE), or the part can be quarantined for rework and scrap (Route CF). Automatic feedback may or may not be performed, depending if the observation supports a feedback adjustment, and if the cell feedback mode (Automatic, Semiautomatic, or Manual) allows it. The decision process in the CMM gauge is shown in Figure 6.3.

From Figure 6.3, the key decisions are

1. *Should the part be accepted?* If the part is accepted, it will be packed and shipped to the customer.

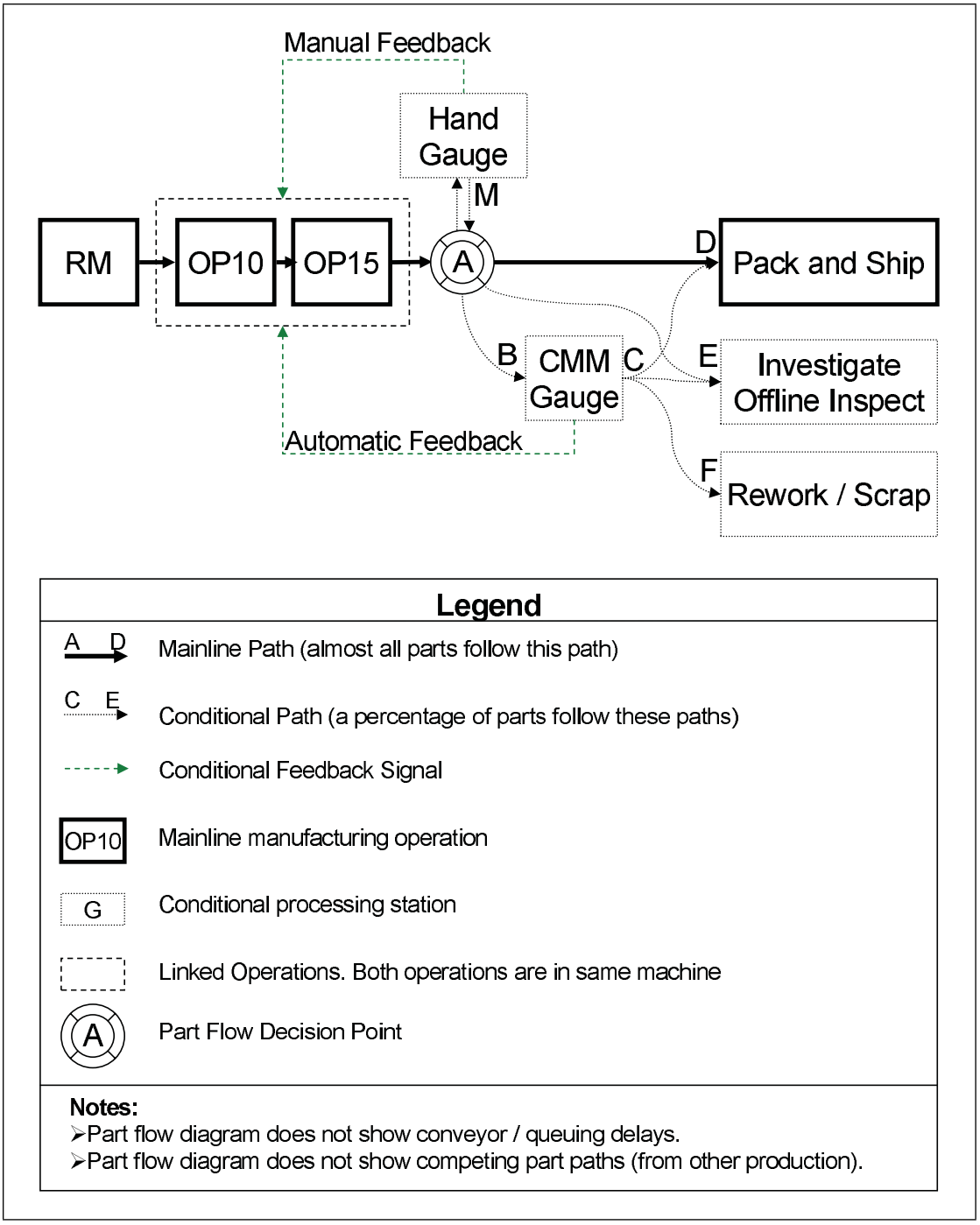

Figure 6.2: Part Flows in a Simple Production Cell

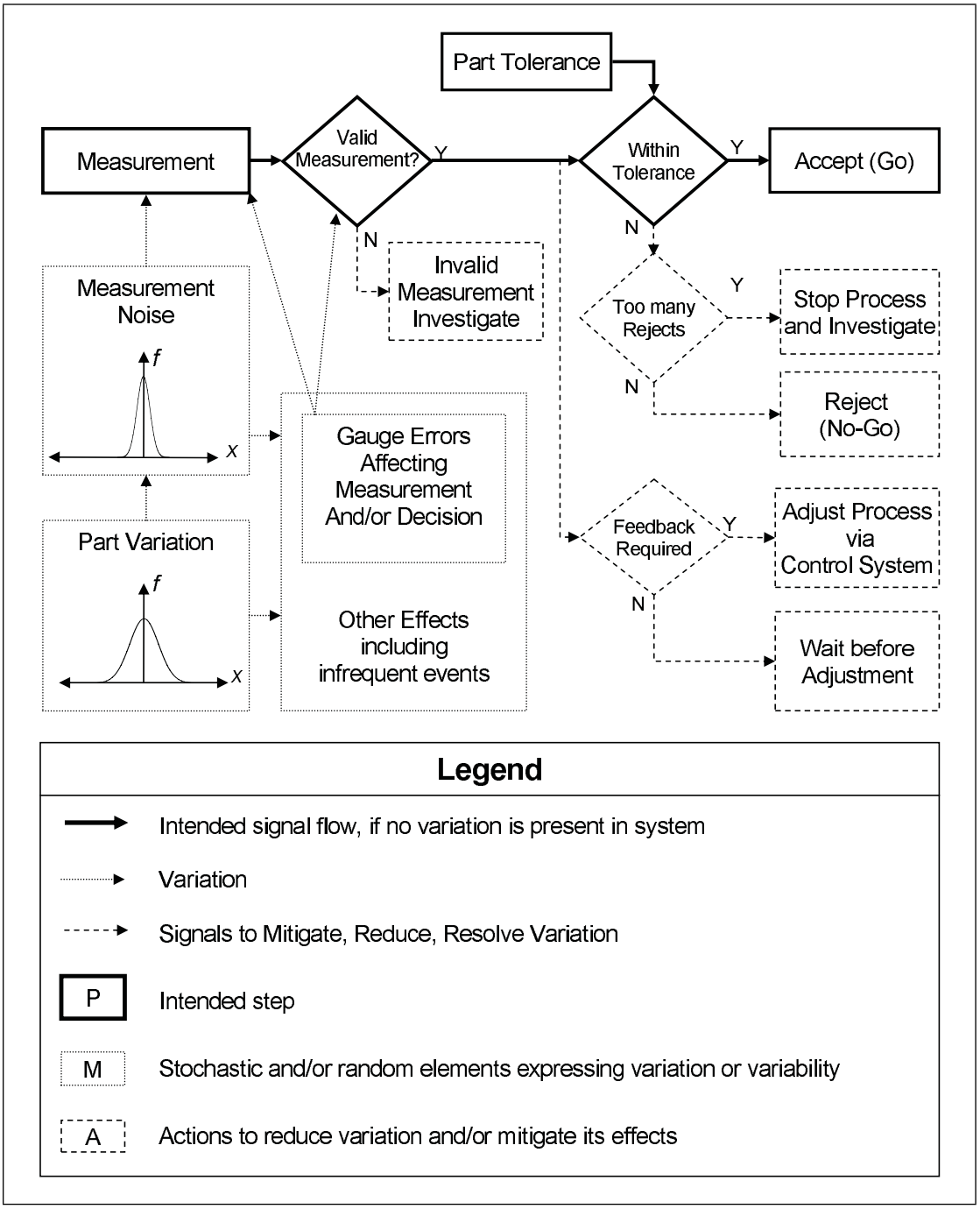

Figure 6.3: Decision Process in the Production Gauge

2. *Are too many rejects present?* If a limited number of parts are rejected, the parts will be quarantined for rework and/or scrap. However, an excessively large number of rejects, the relevant CNC machine will be stopped and the situation investigated by the operator. The point at which the CNC is stopped is application dependent, but the CNC is usually stopped after either three to five consecutive reject parts are observed. Stopping the machine after three to five rejects parts permits one or two feedback operations to occur to correct the process, while preventing the accumulation of large quantities of reject parts.

3. *Is feedback appropriate?* It is not appropriate to feedback every observation. Doing so can actually increase the standard deviation of the manufacturing process and reduce the process capability index C_{PK}. Additionally, the CNC machine can potentially be damaged if large and inappropriate adjustments are made, if feedback is attempted when the machine should be stopped, or if the automatic system unexpectedly overrides an operator setup adjustment.

The accuracy of these decisions is affected by the relative magnitudes of the process error and gauge error and in the observations.

From a capital cost point of view, it is desirable to purchase the minimum amount of production gauging capacity for correct cell operation, subject to the constraints that the cell must operate economically and the customer should always receive in-spec product, *i.e.* product that is within part-print specifications. To protect the customer against receiving out-of-spec (outside of specification) product when insufficient production gauging (CMM) capacity is available, it is recommended that the following rules be adopted:

Rule 1: All parts observed as accept, can be packed as in-spec parts (Route CD).

Rule 2: All parts observed as reject, are rejected. Repeated rejects are investigated. (Routes CE/CF).

Rule 3: All parts manufactured between two parts with in-spec observations can be accepted as in-spec (Route AD) if no process changes are pending, in-progress, or have recently occurred (within the last 3 samples).

Rule 4: All remaining parts are quarantined for on-line (Route AB) or off-line measurement (Route AE). Measurements on these parts occur to late for feedback, and as such are not counted as observations for feedback.

If available CMM capacity is present in the cell, it is desirable to minimize the amount of off-line measurement. However, diverting CMM capacity to measure quarantine parts affects the stability and performance of the feedback algorithms.

6.3 Implementing Feedback in the Manufacturing Cells

Figure 6.4 shows how the process reacts to special cause variation. Specifically, the top trace shows the parts produced by the CNC. The lower trace represents the parts for observation by the CMM.

The feedback controller can only adjust the CNC process after a CMM observation. Thus it cannot react to sudden special cause variation, until Sample 10 is observed.

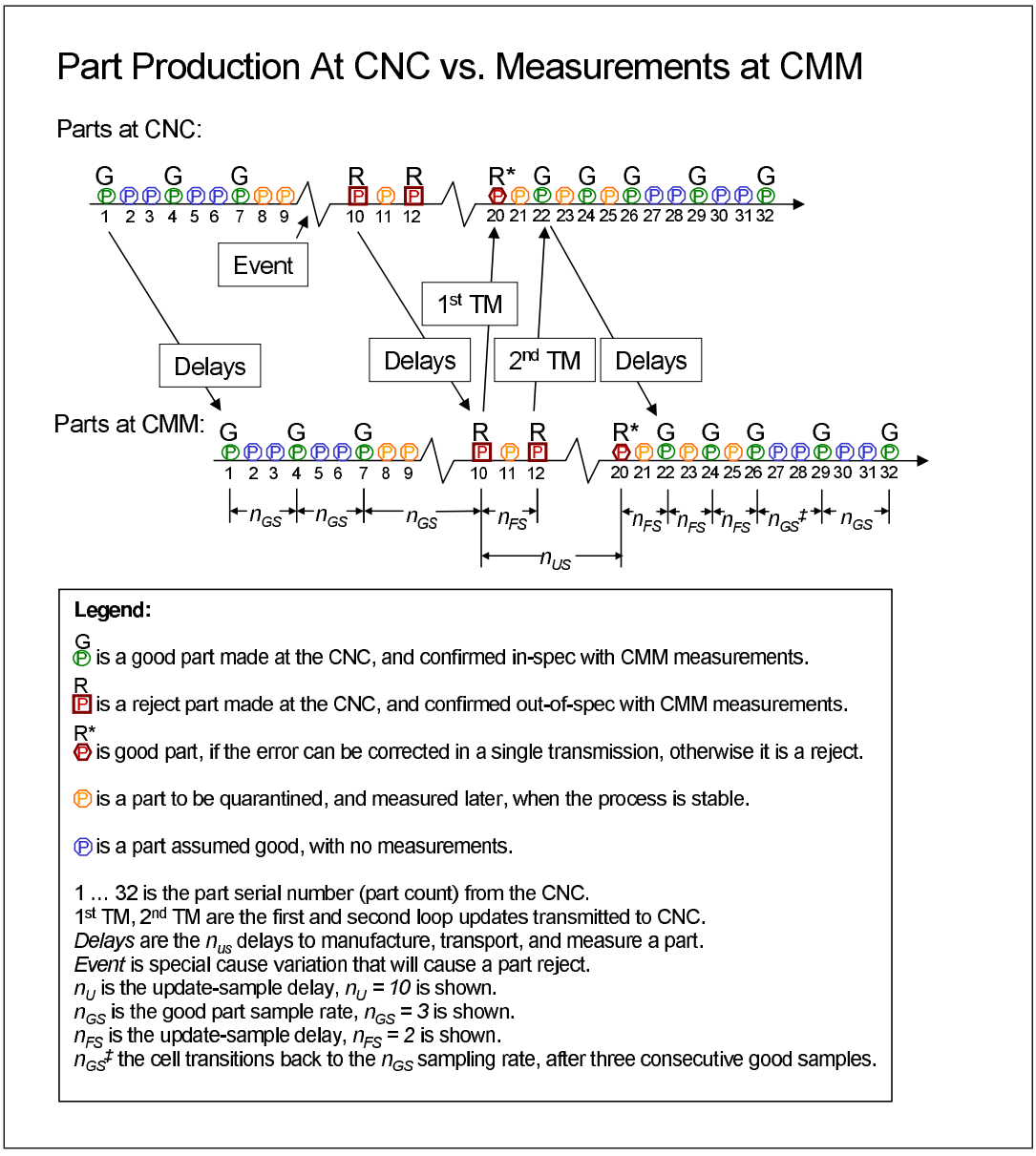

Figure 6.4: Process reacting to a rogue event

In certain circumstances, the variation will not be detected until Sample 12 is observed. In this application, the CNC machining process has two steps with the main spindle performing OP10 while the secondary spindle performs OP15. If the event occurred, as shown, immediately before Part 10 was manufactured, it is possible Part 10 already had some of its OP10 features complete while OP15 is being performed. As the OP10 features were complete before the event happened, these preexisting features will be within specification, "in-spec". As such, Part 11 will be the first part to show all the changes caused by the event, and Part 12 the first observed part reflecting all the changes in the event. The probability of this split update event occurring is $\frac{1}{n_{GS}}$, and it causes a delay of n_{FS}.

The constants n_{GS} and n_{FS} are expressed in terms of parts. From a shop floor perspective, measuring constants in terms of parts has the advantage the sampling rates can simply be determined by counting how many parts go down the line, for each part observed.

The CNC is sent offset data based on the latest sub-spindle machined surfaces after part 10, and the main spindle surfaces after part 12. The parts are machined on the main spindle first, and then machined on the sub-spindle, which creates a one-part timing delay. This delay will either not be detected when the intervening part is not sampled, or it will be detected as a one sample period delay, in this case 2 parts, when the intervening part is sampled. The update-sample delay in the cell, n_U, is most easily determined by counting the physical number of parts in the cell at any stage of the production process. Alternatively, the update-sample delay, n_U, can be analytically computed from

$$n_U = k_{US} n_{FS} \tag{6.1}$$

where k_{US} is the smallest integer such that

$$n_U T_{CNC} > D_{CNC} + T_{CONV} + T_{WAIT} + T_{CMM} + T_{OP} \tag{6.2}$$

where

D_{CNC} is the delay between the first finishing cut on the CNC, and the time the part exits the CNC. Typically, $D_{CNC} \approx 2T_{CNC}$. Built-in exit conveyors on the CNC will increase D_{CNC}. Careful CNC program design can reduce D_{CNC} by postponing finishing cuts to later points in the machining cycle.

T_{CONV} is the total conveyor, material transport, part cleaning, cool-down delay.

T_{WAIT} is the wait time for the CMM.

T_{CMM} is the observation (measurement) time in the CMM.

T_{OP} includes the time for the operator to improve and/or modify the tool offset, and the time required to transmit it to the CNC. In automatic mode, the updates are fully automatic and almost instantaneous, resulting in $T_{OP} \approx 0$.

Eventually, a within specification part will arrive at the CMM. In this example, this will be either 20 or 22. An additional 3 in-spec parts must be observed to ensure the process is stable, before the normal sampling rate n_{GS} occurs. This means the total number of quarantined parts for a single rogue event will be n_{GS} for the initial detection, plus n_U to process the feedback, plus a conditional delay of n_{FS} if a split update occurs. The probability of a split update occurring is $\frac{1}{n_{GS}}$. Additionally, two further n_{FS} sample periods are needed for the process to stabilize. This yields a total CMM load from

the event in Figure 6.4 of

$$\omega_1 = n_{GS} + n_U + \frac{n_{FS}}{n_{GS}} + 2n_{FS} \tag{6.3}$$

$$= n_{GS} + n_U + k_{FS} + 2n_{FS} \tag{6.4}$$

Not every part observed by the CMM will be scrap. For the specific case shown in Figure 6.4, scrap on average will be

$$c_1 \approx \frac{n_{GS} - 1}{2} + n_U + k_{FS} \tag{6.5}$$

with all quantities expressed in parts. The average number of parts scrapped per part produced can be estimated by multiplying c_1 by the probability of producing scrap.

Occasionally, due to rogue events involving special cause variation, the feedback loop will not be able to correctly adjust the process. In this case, the controller is permitted a second try.

$$\omega_2 = \omega_1 + n_U \tag{6.6}$$

Essentially, the second feedback loop correction cycle increases the weight by n_U.

Sometimes a third try is also needed. A practical danger exists in operating the CNC machines for an excessively long time without correct tool offsets. As such, even if this third update is correct, most cells are configured such that the CNC machine will be stopped, and the further investigation by a cell operator requested. This results in

$$\omega_3 \geq \omega_2 + n_U \tag{6.7}$$

The greater than or equal is present, because the operator may not restart the machine immediately, and downtime will likely be involved. In one of

the plant cells, it was observed on multiple occasions that a single instance of ω_3 downtime could account for 25% to 50% of the daily scrap. As downtime can upset the thermal characteristics of the machining process, and cause additional scrap, strong financial incentives exist to avoid unplanned downtime and the associated waste.

6.4 Summary

This chapter defined the cell layout under investigation, and the operation of the gauges that observe the parts. It described the key formulas that will be used to relate the weights of individual rogue events to total cell performance. The next two chapters will describe the effects of key algorithms on the cell, starting with the feedback algorithms, and moving to the material transport algorithms. In the process, the physical cell characteristics will be related to the overall waste presented in the cell, in an overall cost minimization and cost reduction framework.

Chapter 7

Manufacturing with Feedback

7.1 Application Overview

This chapter focuses on the CNC manufacturing with CMM feedback application, and the importance of using improved algorithms for effective feedback. With the presence of manual intervention, sampling, breakdowns, delays, and many other effects, a cell may not be completely controllable through automatic means. These effects can cause the feedback loop to become unstable and/or perform poorly. To compensate, the feedback algorithm must update the tool wear offsets only at appropriate times and by appropriate amounts.

From a practical perspective, the primary goal of this section is to find, from the available possibilities, the feedback algorithm that will produce the least waste in the production cell. The academic literature, as described in §2.3, suggested many different possible feedback strategies. Additionally, from visits to many different plants, it was observed that many different feedback strategies were being used. Also, when a single cell was running in semiautomatic mode, and the operators in the production plants were observed, the operators did not consistently follow any single feedback strategy. The challenge posed by these cells is that different types of rogue events

require different responses, and a single response to all types of variation is ineffective.

7.2 Introduction to Feedback Loop

The feedback loop is the same feedback loop from §4.2.1. The block diagram is repeated in Figure 7.1 for convenience.

Two sampling switches exist in this feedback loop. The "UPDATE SAMPLE" switch closes the feedback loop. As $Z(k)$ is located after the update-sample switch, the wear offset output will always be denoted $Z(k)$, where k represents the k-th operation of the update-sample switch.

For many of the algorithms that will be discussed later in this chapter, it is only necessary to consider the operation of the update-sample switch. For these systems, the observation will be denoted $Y(k)$, and it will be assumed that this observation corresponds to the k-th update-sample.

It is possible to improve control system performance by collecting more observations, and then only occasionally doing feedback loop updates (update-samples). This is done by using the second "SAMPLE" switch to gather observations, and then later deciding if the observations will lead to an update-sample. For systems that gather more observations than are used for loop updates, $Y(j)$ will denote the j-th observation. It is assumed that the control system software is capable of keeping track of the current values of j and k, such that the j-th observation $Y(j)$ is used for the corresponding k-th wear offset update $Z(k)$. This bookkeeping is easily implemented with modern computers.

7.2.1 Delay in the Physical Cell

Both of the successful industrial cells were implemented such that the typical delay D encountered in the cell was between 4 and 5 parts in normal

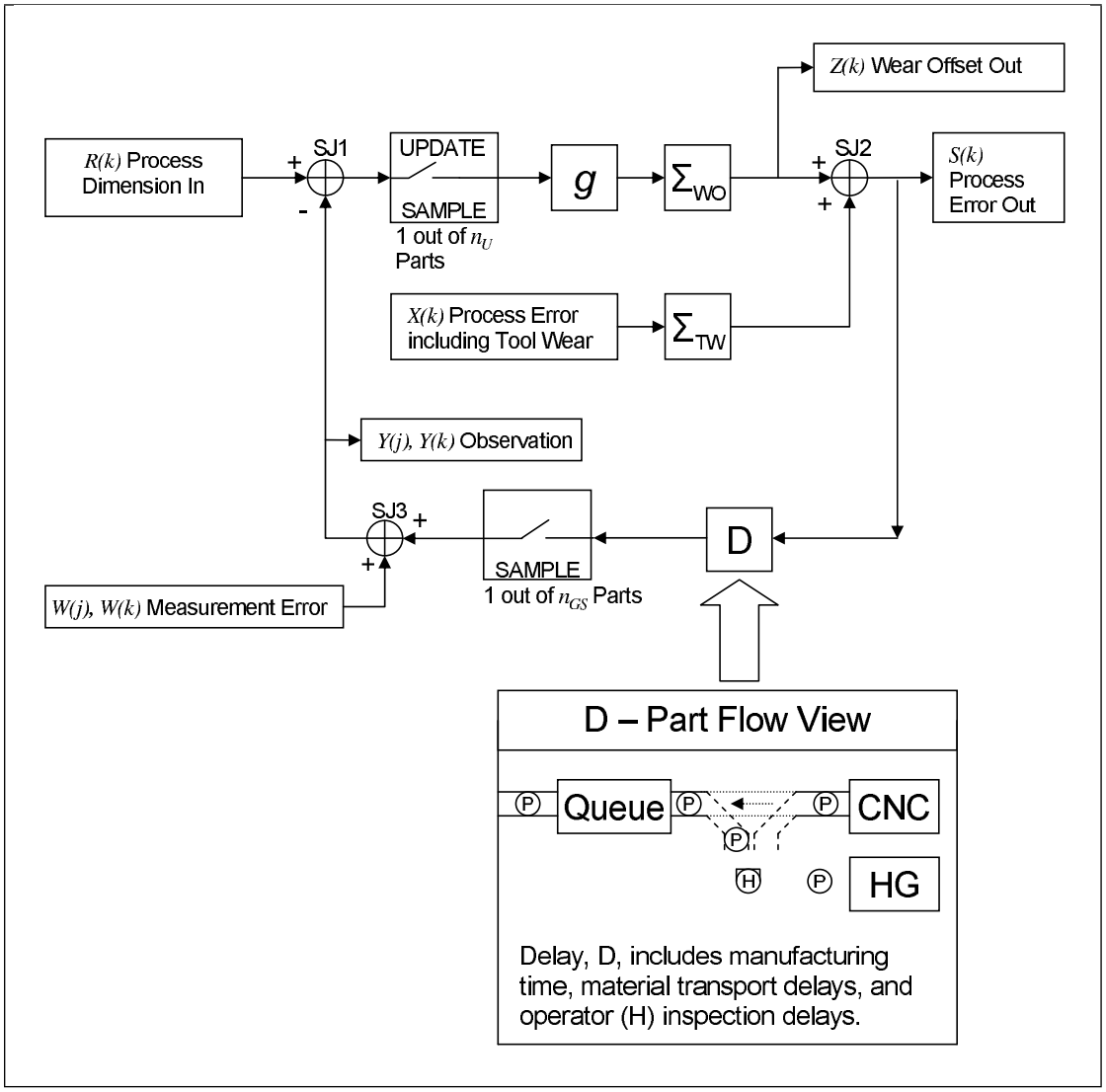

Figure 7.1: Feedback Loop for Sample Update

operation. The industrial cell from which the data was captured had a fixed $n_{GS} = 3$ parts per sample. This means that if the update sample rate $n_U = 2n_{GS}$, then $D = 1$ update-sample for the vast majority of the update-samples. More specifically

$$n_U = 2n_{GS} \qquad \Longrightarrow \qquad \Pr(D = 1) \approx 1 \tag{7.1}$$

Similarly if the update sample rate $n_U = n_{GS}$, then $D = 2$ update-samples for the vast majority of the update-samples.

$$n_U = n_{GS} \qquad \Longrightarrow \qquad \Pr(D = 2) \approx 1 \tag{7.2}$$

Assuming D appropriately, the off-line method §4.2.4.2 can be used to determine the conditional probabilities of different events occurring based on a single observation $Y(j)$.

7.2.2 Probability Estimation From Captured Data

The feedback loop is restricted to two actions: CHASE (A) and WAIT (A'). With CHASE, the update sample loop is iterated. To maintain $D = 1$ update-sample, the loop update frequency should be limited. With WAIT, the update sample loop is not iterated. Many consecutive WAIT actions can be performed. However, eventually feedback will be needed to keep the process in control.

By only performing CHASE when it is most advantageous, the probability of variations in D affecting the feedback loop is minimized, and effective control action results. Additionally, if CHASE actions are timed to coincide with rogue process events, then waste is minimized.

With collected data, it is possible to estimate the conditional probabilities of the CHASE / WAIT decision meeting with success. Unfortunately, multiple

rogue events can occur in a row, and as such neither the CHASE nor the WAIT decision will bring the process back into specification. Sometimes, two consecutive CHASE decisions are required. This is referred to as a TWO-STEP CHASE. Also, in some circumstances, the outcome of the CHASE and WAIT decision is the same within expected gauge error. This is an INCONCLUSIVE result. Including, CHASE, TWO-STEP CHASE, WAIT, INCONCLUSIVE, 4 different probabilities were computed for any given observation $Y(j)$.

CHASE E_A and TWO-STEP CHASE E_B: The optimal action is CHASE if the CHASE action A yields $|Y(j+D)| < U_T$ and waiting would not obtain a better result. A TWO-STEP CHASE occurs if it requires two consecutive CHASE actions for $|Y(j+D)| < U_T$, and waiting would not yield better results. Sometimes, the tool wears quickly, and two consecutive iterations are required to keep the process tolerance, as the extent of the rapid tool wear problem is not obvious on the first iteration.

WAIT E_C: The optimal action is to not update the output.

INCONCLUSIVE E_D: The optimal action by the feedback system is not obvious. For instance, both CHASE and WAIT operations may result in the same value of $|Y(j+D)|$ within gauge error. Alternatively, it may not be possible for the feedback controller to achieve $|Y(j+D)| < U_T$ as a malfunction is occurring.

Using a sample of data, it is possible to graph the optimal outcome for any given value of y. The resulting histogram is shown in the top half of Figure 7.2. It can be seen that significant tails exist, and that these tails are bigger than that predicted from the normal distribution. These large tails represent rogue events.

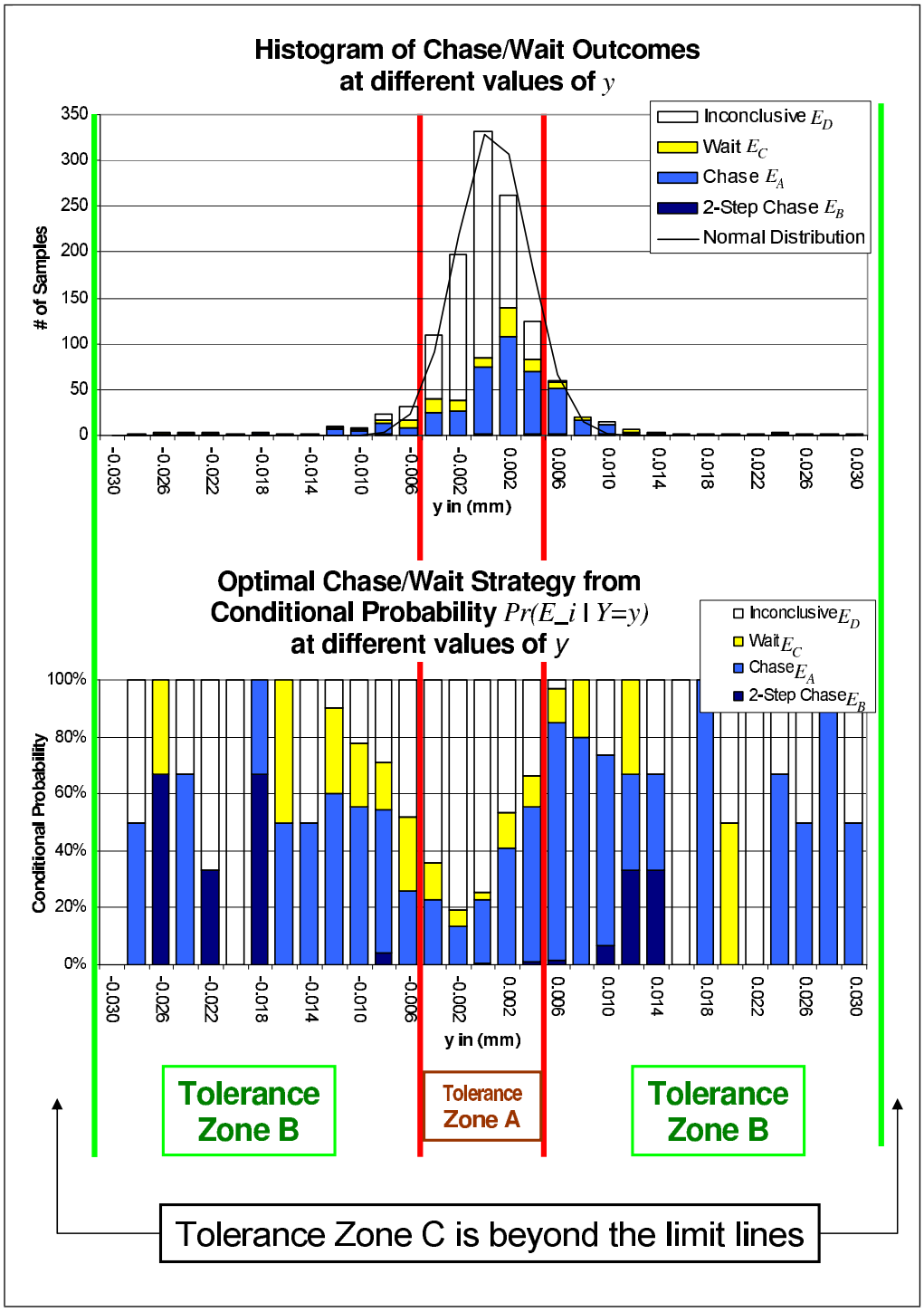

Figure 7.2: Zone A, B, C - shown against the captured histograms from the industrial plant.

The conditional probability $\Pr(E_i|Y(j) = y)$ is shown in the lower half of Figure 7.2 for various values of y. It should be noted that at the tales shown in the upper half of Figure 7.2, only one to three data points are present, thus the graph shown in the lower half of Figure 7.2 becomes very discrete in the tales with many bars being exactly 1/3, 1/2, or full height. Nevertheless, it is possible to discern large zones labeled A, B and C on the graph. In the center Zone A a large concentration of inconclusive readings E_D exist. Given the limit on update-samples, it is better to CHASE infrequently in this region, and WAIT the rest of the time. Outside the center region, in Zone B, the problem is more biased to CHASE decisions.

In the end, the decision making process was divided into three zones as shown in Figure 7.2.

Zone A is a frequently occurring but largely indeterminate process zone. Occasional updates in this zone keep the process well centered. Excessive updates cause issues with feedback loop stability.

Zone B is the region in which feedback should occur. Many rogue process events are present in this region, and rogue process events require a feedback response.

Zone C is a region in which feedback should never occur, because it can damage the machine. Note that no samples exist in this region, so it is not included on the graph. This corresponds to the ε case addressed in §4.2.8.

7.3 Wear Update Algorithms

Simulation results will be presented on seven different wear update algorithms. When dealing with infrequent events, simulations have a significant

advantage over field data, in that all of the algorithms see exactly the same sequence of events. This greatly improves the accuracy and validity of the comparison, as it controls for infrequent statistical effects.

Five of the simulated wear update algorithms (#1 to #5) were causal algorithms, and did not rely on future predictions. All of the causal algorithms were field tested.

The remaining two algorithms (#6C and #7C) are noncausal, and are designed to serve as benchmarks for comparison purposes. The advantage of the noncausal algorithms is that they can be used to estimate the maximum possible performance achievable inside this system. The following subsections describe each algorithm individually. The results of testing these algorithms are presented in §7.4.

7.3.1 Algorithm #1: SPC

This algorithm was the first to be developed and deployed. A key aspect of SPC is averaging, and this algorithm simply takes the average of the last 3 observations, and immediately updates the system with the result. This algorithm was implemented as a simplification of the work from Bering *et al.* (2002). For modeling purposes, averaging represents a simple effective algorithm that makes decisions on three data samples instead of one.

As implemented, this algorithm is

$$Z(k) = Z(k-1) - \frac{Y(k) + Y(k-1) + Y(k-2)}{3} \tag{7.3}$$

where

k is the current update-sample count.

$Y(k)$ is the observation corresponding to the current update-sample.

$Y(k-1)$ is the observation from the previous update-sample,

$Y(k-2)$ is the observation from 2 update-samples prior.

$Z(k)$ is the wear offset corresponding to the current update-sample.

$Z(k-1)$ is the wear offset corresponding to the previous update-sample.

This algorithm updates every cycle, and as such, $n_{GS} = n_U$ for the simulation. Every cycle updates are required in this application, because of the presence of rogue events.

The earlier work from Bering *et al.* (2002) has an algorithm to avoid doing updates every iteration, and this was intentionally not implemented in this simulation. The earlier work did not contemplate a situation in which rogue events happened as often as they did in the industrial cells examined in this thesis. When these rogue events were encountered in the industrial plants, it became obvious that new algorithms were required and hence this the work presented in this thesis was developed.

7.3.2 Algorithm #2: Feedback

This algorithm was developed and deployed next. It makes the assumption that for a sufficiently small gain g, the feedback loop should be stable. As such, this feedback algorithm feeds back the observation as a correction to the CNC immediately.

The feedback algorithm was implemented in two different forms. The first form triggered a feedback loop update k for every observation j, resulting in $n_{GS} = n_U$. The wear offset value $Z(k)$ was computed from

$$Z(k) = Z(k-1) - gY(j) \quad \text{where} \quad j = k \tag{7.4}$$

where

g is the gain of the feedback loop.

j is the current sample count.

k is the current update-sample count.

$Y(j)$ is the observation corresponding to the current measurement sample.

$Z(k)$ is the wear offset corresponding to the current update-sample.

$Z(k-1)$ is the wear offset corresponding to the previous update-sample.

With $n_{GS} = n_U$, the vast majority of update-samples would have $D = 2$ update-samples. As such, gain values of $g = 0.33$ and $g = 0.5$ were used, with $g = 0.5$ representing the gain value that should generate the minimum variance.

The second feedback algorithm triggered a loop update for every second observation. As such, $n_{GS} = 2n_U$, and the vast majority of update-samples would have $D = 1$ update-sample. The wear offset value $Z(k)$ was computed from

$$Z(k) = Z(k-1) - gY(j) \quad \text{where} \quad j = 2k \tag{7.5}$$

With $j = 2k$, the odd numbered samples $j = \{1, 3, ...\}$ are ignored. This algorithm was modeled with $g = 1$ which is the theoretical optimum as shown in §4.2.3.1.

The feedback algorithm with $g = 1$ is better in simulation than in reality. Importantly, the data set used in this simulation was based on a day of normal cell operation, and as such, does not include any instances of the operator consistently causing a larger than expected loop delay. Nevertheless, aggressive and precisely-timed updates are very effective at controlling the process. The limitation is in achieving the precise timing in a practical cell, when operator effects are present.

7.3.3 Algorithm #3: Fixed Increment

If the observation is outside the control limits ($\pm 3\mu m$), the fixed increment algorithm applies a correction in H-sized increments, where H=$4\mu m$. These values were chosen because for this industrial data set, $2H$ should be on the order of a frequently occurring Zone B adjustment in Figure 7.2. Assuming the operators use the convention that action only occurs when the control limits are exceeded, then the control limits should be one unit of positioning resolution ($1\mu m$) inside $\pm H$. The algorithm matches the instructions given to the cell operators for use in semiautomatic mode. Semiautomatic operation was described in §6.2.

This algorithm is

$$Z(k) = Z(k-1) - \begin{cases} H & \text{when } Y(k) \geq H \\ -H & \text{when } Y(k) \leq -H \\ 0 & \text{otherwise and no update occurs.} \end{cases} \tag{7.6}$$

and $n_{GS} = n_U$. The vast majority of update-samples have $D = 2$ update-samples, assuming the operator is attentive.

From a practical point of view, it must be remembered that this algorithm is used in semiautomatic mode, and is implemented by having the operator press the "+" and "-" wear offset buttons on the CNC machines. This difference in detail does not affect this simulation.

7.3.4 Algorithm #4: Two-Stage

If the observation is outside the control limits at ($+5\mu m$ to $-4\mu m$), the feedback algorithm is used with $g = 1$, $n_U = 2n_{GS}$. Otherwise, this algorithm attempts to look for a trend in the last three observations. If they are all consistently above or below the mean, then the average of the three observations is used to update the feedback loop. This algorithm implements

feedback in the mathematically ideal Zone B, while limiting the rate of feedback updates in Zone A. The control limits are chosen to correspond to those from Figure 7.2 for Zone A, and were optimized to the nearest micrometer based on simulation results in an effort to minimize scrap.

As tool wear generates a distribution with a nonzero mean, some experimentation was done in the manufacturing cell as to whether the average of the three observations or the last of the three observations should be used for feedback. No consistent advantage was found. As such, the average is used in the simulations.

When an update occurs, this algorithm sets

$$Z(k) = Z(k-1) - g \begin{cases} Y(j) & \text{when } Y(j) > 5\mu m \\ Y(j) & \text{when } Y(j) < -4\mu m \\ \frac{Y(j)+Y(j-1)+Y(j-2)}{3} & \text{otherwise.} \end{cases} \tag{7.7}$$

where

g is the gain. This algorithm was simulated with $g = 1$.

j is the current sample count.

k is the current update-sample count.

$Y(j)$ is the observation corresponding to the current measurement sample.[1]

$Y(j-1)$ is the observation corresponding to the previous measurement sample.

[1]Referring to §7.2 and Figure 7.1, it is possible to gather more observations (samples) than to perform feedback loop updates (update-samples). From §4.2.3.1, the minimum variance response is attained when D is one update-sample and $g \approx 1$. In Zone B, it is desirable to respond to a rogue process event as quickly as possible. Thus in Zone B, assuming another update is not already in progress, the current observation $Y(j)$ is used to update $Z(k)$ immediately via the update-sample switch. Otherwise in Zone A, where rogue events are unlikely, several observations $Y(j)$, $Y(j-1)$, $Y(j-2)$ can be gathered before the update-sample switch closes.

$Y(j-2)$ is the observation corresponding to the measurement sample from 2 samples prior.

$Z(k)$ is the wear offset corresponding to the current update-sample.

$Z(k-1)$ is the wear offset corresponding to the previous update-sample.

Per §7.2, it is assumed that the j-th observation $Y(j)$ is used for the corresponding k-th wear offset update $Z(k)$.

When $+5\mu m \geq Y(j) \geq -4\mu m$, an update will only occur if

$$\frac{g\,|Y(j)+Y(j-1)+Y(j-2)|}{3} \geq 1\mu m \tag{7.8}$$

and if no other update has occurred as a result of the $j-1$ and $j-2$ samples.

When $Y(j) > +5\mu m$ or $Y(j) < -4\mu m$, an update will only occur if no update occurred on the previous $j-1$ sample. This means that $n_U \geq 2n_{GS}$. As a result, from §7.2.1 and (7.1), the vast majority of loop updates will have $D = 1$ update-sample.

7.3.5 Algorithm #5: Split-Sample

In this system, when rapid process change has been encountered, it may be necessary to issue two feedback loop updates in rapid succession. If an update is already in process, and the delay in the system is uncertain, then it may be difficult to know which parts have been machined with the revised offsets. As such, the next observation may be from a part made with either the old or the new wear offsets. As a result, the optimal response is not obvious.

These ambiguously-timed observations can still be used for feedback, without compromising loop stability, if the next offset is reduced by the worst case change in the offset table. The resulting optimization is a *split-sample* update. Split-sample updates do not affect loop-stability, because the magnitude of the update has been reduced appropriately. The resulting feedback loop is shown in Figure 7.2.

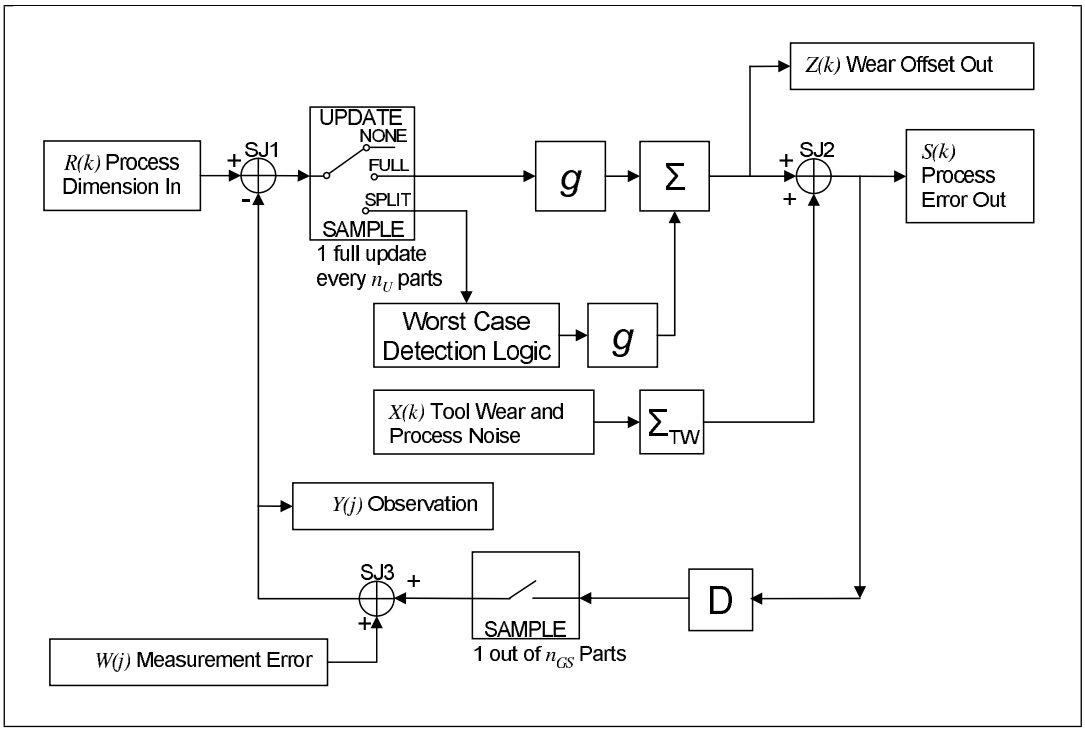

Figure 7.3: Feedback Loop for Split Sample Update

The resulting split-sample algorithm is identical to the two-stage algorithm, except for the addition of the split-sample update option. When a split-sample update is needed, *i.e.* when the $j-1$ observation triggered an update and the $j-2$ observation did not trigger an update, then

$$Z(k) = Z(k-1) - g \begin{cases} Y(j) - |Z(k-1) - Z(k-2)| & \text{when } Y(j) > 5\mu m \\ Y(j) + |Z(k-1) - Z(k-2)| & \text{when } Y(j) < -4\mu m \\ 0 & \text{otherwise} \end{cases} \quad (7.9)$$

where

g is the gain for this simulation.

j is the current sample count.

k is the current update-sample count.

$Y(j)$ refers to the observation corresponding to the current sample.

$Z(k)$ refers to the wear offset corresponding to the current update-sample.

$Z(k-1)$ refers to the wear offset corresponding to the previous update-sample.

$Z(k-2)$ refers to the wear offset corresponding to the update-sample from 2 update-samples prior.

Per §7.2, it is assumed that the j-th observation $Y(j)$ is used for the corresponding k-th wear offset update $Z(k)$.

When the result of the above calculation yields $Z(k) = Z(k-1)$, the update is automatically suppressed and no update-sample event will occur. Gains of $g = 1$ and $g = 0.9$ were used in this simulation. When the $j-1$ observation did not trigger an update, this algorithm uses (7.7) and (7.8) and proceeds in the same manner as algorithm #4: two-stage.

It is important to avoid two or more consecutive split-sample updates. Specifically, if many rogue events occur in a cluster, it is often advantageous to wait for the system to stabilize. As such, the algorithm will only perform a split-sample updates when the $j-1$ observation triggered an update and the $j-2$ observation did not trigger an update.

It takes a great deal of book keeping to implement algorithm #5, and to a lesser extent with algorithm #4. As such, it would be difficult to get an operator to consistently perform the necessary calculations to implement this algorithm. However, this algorithm can be easily implemented with modern computers.

7.3.6 Reference Algorithm #6C: Expected

The expected value of the system performance was calculated for reference. It was computed from the expected process scrap rate assuming a normal process distribution using the average, μ_{CC}, and standard deviation, σ_{CC}, of the common-cause variation. It was included to show the expected scrap rates from a model that does not include the effects of rogue events.

The calculation is

$$\Pr(\text{Scrap}) = 1 - \Phi\left(\frac{U_{TOL} - \mu_{CC}}{\sigma_{CC}}\right) + \Phi\left(\frac{\mu_{CC} - L_{TOL}}{\sigma_{CC}}\right) \tag{7.10}$$

where U_{TOL} and L_{TOL} are the upper and lower tolerance limits respectively.

It should be noted that this estimate does not properly model any rogue effects, including the effects related to operator intervention, rapid cutting tool wear, and cutting tool replacement. As such, when the manufacturers used this estimate, they were very disappointed by system performance.

7.3.7 Reference Algorithm #7C: Foreknowledge

The optimal CHASE/WAIT decision was applied at each step in the simulation, based on which decision yielded the optimal future results. This value was calculated to serve as the theoretical upper limit on control system performance when rogue events are modeled.

The calculation is

$$Z(k) = Z(k-1) - \begin{cases} Y(k) & \text{when} \quad |Y_A(k+2)| < |Y_{A'}(k+2)| \\ 0 & \text{when} \quad |Y_A(k+2)| \geq |Y_{A'}(k+2)| \end{cases} \tag{7.11}$$

where

k is the update-sample count, and an update occurs for every sample.

$Y_A(k+2)$ is the projected value of Y assuming the CHASE action is taken and $Z(k) = Z(k-1) - Y(k)$.

$Y_{A'}(k+2)$ is the projected value of Y assuming the WAIT action is taken and $Z(k) = Z(k-1)$.

As this algorithm uses future results, it cannot be implemented in a physical control system. This algorithm updates on every iteration, as such $n_U = n_{GS}$.

7.4 Simulation Results

The industrial partners were strongly committed to reducing scrap, and would only run the best available algorithms. As such, the effectiveness of the different algorithms was simulated based on a factory acquired data set. Using factory acquired data means that the simulation results fully reflect the impact of operator interaction and all of the complexities of the machining process. Additionally, the simulations can be made extremely accurate, to the point that they exactly match algorithm behaviour in the field.

The results shown in Table 7.1 represent the limit of what can be achieved with these algorithms in the simulated cell. Nevertheless, the simulations do have their limitations. For instance, two of the simulations were run with $g = 1$. In practice, some special cases exist that make it advisable to set $g < 1$. Those special cases were not in this data set, as this data set came from mature production with experienced operators that knew how to maximize the probability that $D = 1$ update-sample with $n_U = 2n_{GS}$. Overall, the results trend closely with the results that were attained in the industrial cell, and the factories involved were more than satisfied with the results.

Table 7.1: Simulated Performance of Different Feedback Algorithms Based on Captured Data

Process Parameters		
# of Samples, Parts	1238 Samples	3714 Parts
Average μ_{CC}	0.0007 mm	
Standard Deviation σ_{CC}	0.0029 mm	
Tolerance Bands	**Narrow**	**Wide**
U_{TOL}	0.009 mm	0.018 mm
L_{TOL}	-0.007 mm	-0.014 mm
Statistical Tolerance Band	$5.6\sigma_{CC}$	$11.2\sigma_{CC}$
Algorithm Results (% Waste)		
Reference Algorithms		
#6C Expected	0.54%	0.1 ppm
#7C Foreknowledge	13.86%	5.72%
Algorithms		
#1 SPC	61.40%	31.86%
#2 Feedback: $g = 0.5$, $n_U = n_{GS}$	68.35%	32.24%
#2 Feedback: $g = 0.33$, $n_U = n_{GS}$	36.17%	15.28%
#2 Feedback: $g = 1$, $n_U = 2n_{GS}$	23.54%	8.83%
#3 Fixed Increment	29.08%	8.61%
#4 Two-Stage: $g = 1$	24.99%	8.00%
#5 Split-Sample: $g = 1$	23.11%	7.25%
#5 Split-Sample: $g = 0.9$	21.00%	7.73%

The tolerance ranges were chosen based on the predictions of the common-cause normal distribution. The narrow tolerance range was chosen to be close to the tails of the predicted normal distribution from Figure 7.2. The wide tolerance range was chosen to be roughly twice as big as the narrow tolerance range, and corresponds to a $C_{PK} = 1.6$ process. When dealing with single sided monotonic tool wear, it is advantageous to run the process slightly off of center. The narrow tolerance band is based on a customer specification, and is centered for optimal production performance.

In terms of widely-used process standards, the wide tolerance range meets the Motorola Six Sigma definition where a 6σ band is divided into two sections: the first is a 1.5σ band to allow for variation in the mean, and the second is a 4.5σ band to allow for variation from the mean. This represents

a $C_{PK} = 1.5$ process.

Often, samples affected by rogue events are eliminated from a short-run data set because one errant data point can significantly skew a data set. Additionally, operators are expected to quarantine the affected parts that might be affected by rogue events. However, in a fully automated cell, no operator is present. This sampling bias, coupled assuming the normal distribution applies and no rogue events occur, can significantly underestimate process waste. This can be seen in the simulation results for Reference Algorithm #6C. For the narrow tolerance band, common cause variation predicts 0.54% waste versus the best achievable waste from the split-sample algorithm of 23.11%. Using a wider tolerance band, shows a more dramatic underestimation effect: 0.1 ppm versus 7.25%. The prediction errors occur because the rogue events are not modeled.

Some caution must be exercised when using the algorithm #2 feedback with $g = 1$ and $n_U = 2n_{GS}$. The data set in this simulation does not contain some of the infrequent cases that may be encountered on cell startup, with inexperienced operators, and/or with special quality procedures. The cell controller should work in all of these cases, in addition to the usual $D = 1$ update-samples case. Advantages exist in maintaining a lower loop update rate and a $g < 1$ to deal with the cases where $D = 2$ update-samples. Algorithm #5 has a lower average loop update rate, and with $g = 0.9$ it also yields better results than algorithm #2.

The results showed that algorithm #5 split-sample was most effective. The split-sample algorithm offered the fastest response to rogue process error. Additionally, operating with $g = 0.9$ yielded reasonable results for the response of the split-sample algorithm. This is an important result if the controller is to remain stable in corner cases where $D = 2$ update-samples for extended periods of time.

Interestingly, with the narrow tolerance band, the algorithm #5 split-sample with $g = 0.9$ outperforms the same algorithm with $g = 1$. Referring

back to Figure 7.2, it can be seen that the CHASE/WAIT decision choice is not reliable. If $g = 0.9$, g is sufficiently large that when rogue process variation occurs, the process is rapidly brought back into specification. However, it is also likely measurement variation caused by either random delay or measuring error, will yield an incorrect feedback response. For narrow tolerance bands, operating with the split-sample update with $g = 0.9$ allows the controller to quickly recover from an incorrect adjustment in a sufficiently large number of corner cases as to make an appreciable difference on the total score for this data set. This effect does not always result in an improvement, as can be seen by comparing the split-sample algorithm results with $g = 1$ and $g = 0.9$ for the wide and the narrow tolerance bands. However, the split-sample algorithm with $g = 0.9$ performs better than any of the other algorithms when complex corner cases are considered. Thus, the split-sample algorithm #5 with $g = 0.9$ proved to be the robust choice in the industrial applications.

For ease of operator training at wide tolerance levels, a simple fixed increment algorithm yields competitive results. Flexibility in control algorithms and cell configurations (allowing manual modes) is critical for cell success. It is recommended that captured profiling data be used to select the optimal algorithm in industrial applications based on the actual process conditions encountered.

7.5 Estimating Performance in Advance of Cell Construction

The best information comes from profiling a running process in a manufacturing cell, because this data is highly accurate and reflects the full manufacturing process with all of its complexities. This data source is not available when the initial cost estimates of production are being compiled. Often,

small prototype runs are performed. If sufficient data is available, then a full probability distribution can be computed, as was done in §7.2.2. However, the prototype runs are often too short for this.

Prototype runs are often of sufficient length to allow the estimation of α. Specifically, it is possible to capture how often the operator must intervene in the process to keep part production within the applicable control or tolerance limits. This can be a reasonable approximation of α. If the chosen feedback algorithm and update sample rates make sense for the expected α, with an allowance for β, then a reasonable likelihood of cell success can be expected. If on the other hand, the expected α is very large, then it is not advisable to construct an expensive automated cell for the application. Once the cell is deployed, captured data can be used to develop simulations to permit more accurate analysis of system performance.

7.6 Summary

The split-sample update algorithm yielded the best quality results. This was born out in the two successful plant deployments. The split-sample update algorithm has the advantage that it has multiple modes of operation, and selects the appropriate mode for different process conditions.

From the analysis presented, the reason why plant operators do not follow a simple single control algorithm, is that a single simple algorithm does not yield optimal results across all process conditions. The SPC approach works well when the input data is dominated by noise that can be assumed to be normally distributed. A strong feedback approach works well when the cutting tool is wearing rapidly. The control approach changes according to the type nature of the process error, rogue process error, gauge error and rogue gauge error present in the manufacturing process. When the unexpected happens and the above approaches fail, operator intervention is required.

This chapter focused on the control system aspects of ensuring loop stability. It presented the split-sample update algorithm which yielded the best results in the manufacturing cells. The next chapter will focus on the key performance limitations set by the material transport system algorithms and design. Based on this information, the system can be further optimized.

Chapter 8

Manufacturing with Feedback: Impact of Queuing / Material Transport Algorithms

8.1 Application Overview

As discussed in §4.2.5, variations in the delay inside the feedback loop could affect cell performance. In the industrial application, this delay is influenced by physical factors, including cell layout, the relative production and measurement capacities of the machines in the cell, and the performance of the robotic material transport system. This chapter is about the algorithmic improvements that can be made to the material transport system, and the physical constraints of the cell.

Initially, the automotive suppliers purchased material transport systems that featured long and variable delays. When the collected data were reviewed, they strongly suggested correlations between problems in the material transport system and problems in the production process. This resulted in the research presented in this chapter.

Ultimately, the sampling rates and delays present in the cell will be limited by characteristics of the cells, including the characteristics of the material transport system and the number and performance of the CNC machines and CMM gauges in the cell. The next section presents a calculation on the minimum CMM capacity required for feedback loop operation. Assuming that a minimum CMM capacity is present, §8.3 addresses the problem of how to minimize queue delays by improved software queue control algorithms.

8.2 Capacity Limits

From a practical point of view, there is an advantage in minimizing the capital costs associated with the cell by minimizing measurement capacity. This section aims to derive a formula that captures this constraint.

A certain amount of inspection capacity needs to be dedicated to providing the feedback system with ongoing samples such that tool wear can be corrected for. This capacity will be influenced by the production capacity of the CNCs, U_{CNC} and the sampling rate n_{GS}. For this analysis, it will be assumed that the peak production rate from the CNCs will be equal to the total possible cell production per period, *i.e.* $U_{CNC} = f_{PER}$.

Some material transport algorithms increase the sampling rate when the process goes out-of-spec. This increased sampling rate is denoted by n_{FS} where $n_{GS} \geq n_{FS} \geq 1$.

Let f_A be the rate at which the process jumps out-of-specification. The minimum CMM capacity needed to monitor the process occurs when $f_A = 0$, and is

$$U_{CC} = \frac{U_{CNC}}{n_{GS}} \tag{8.1}$$

The additional CMM capacity needed to support the faster sampling due to f_A is

$$U_{FS} \approx f_A(n_{GS} - 1)\left[1 + k_{FS}(k_{US} + 2)\right] \tag{8.2}$$

where

$$k_{FS} = \frac{n_{FS}}{n_{GS}} \tag{8.3}$$

$$k_{US} = \frac{n_U}{n_{GS}} \tag{8.4}$$

U_{CC} represents the measurement capacity needed for the cell to make feedback adjustments at the normal rate, and U_{FS} represents the measurement capacity needed for additional sampling. For the cell to keep up with production, the available CMM capacity must exceed cell requirements, hence

$$U_{CMM} \geq U_{CC} + U_{FS} \tag{8.5}$$

If this inequality is violated, even for a short period of time, the feedback loop will be starved of updates. To prevent instability due to insufficient updates, one or more of the CNC machines should be shut down or switched to manual mode.

The waste produced by the feedback system in the cell is given by ω_{FN}. As $\omega_1 > c_1$, some of the parts in ω_{FN} will be within specification, and will not require rework. In order to find out which parts need rework and which parts are within specification, all of the parts need to be measured. These measurements can be taken either inside the production cell (on-line), or outside the production cell (off-line), however on-line measurements are often cheaper. Thus, if the cell has additional available measurement capacity, *i.e.* $U_{CMM} > U_{CC}$, it is worthwhile to measure ω_{FN} parts on-line, to see which parts require rework and which do not.

Some of the ω_{FN} parts will have already been measured by the CMM because the CMM automatically observes one part in every n_{GS} as part of

its normal sampling strategy. As such, the additional CMM load created by these extra measurements, $U_{SCEXTRA}$, will be given by

$$U_{SCEXTRA} \approx \omega_{FN}(1 - 1/n_{GS}) \tag{8.6}$$

The total capacity required for CMM measurements inside the cell is

$$U_{TOTAL} = U_{CC} + U_{SCEXTRA} \tag{8.7}$$

It is desirable to construct the CNC machining cell such that

$$U_{CNC} \gg U_{CMM} > U_{TOTAL} > 0 \tag{8.8}$$

This ensures sufficient measurement capacity to measure parts during adverse process conditions. Additionally, capital costs can be reduced when $U_{CNC} \gg U_{CMM}$.

8.3 Queuing Algorithms

The choice of material transport (queuing) algorithm affects n_{FS}, n_{GS}, n_U, and hence ω_1. Often, queuing algorithms are implemented in software. In this case, the queuing algorithm is implemented as physical pieces of material transport equipment. As such, an incorrect choice of equipment can result in sudden cell feedback loop failures and/or uneconomic production. The following four queuing algorithms are popular choices.

8.3.1 Algorithm #1: Oldest First

The oldest first algorithm is a First-In First-Out (FIFO) design. Conceptually, and often in implementation, each CNC has a conveyor. At the CNC

parts are loaded onto the conveyor, and the parts are unloaded at the other end of the conveyor, near the CMM.

Standard practice is to not purchase any more inspection capacity than necessary, as this minimizes the capital cost of the cell. As such, assume the limiting case where the CMM is just keeping up with production from the CNCs. The CMM will be measuring parts from one machine, and an additional $n_{CNC} - 1$ parts will be waiting to be measured. The average CMM cycle time will be T_{CMM}/n_{CMM} which results in T_{WAIT} being

$$T_{WAIT} = \frac{(n_{CNC} - 1)T_{CMM}}{n_{CMM}} \tag{8.9}$$

This is the result from Bering and Veldhuis (2010b). This algorithm sets $n_{FS} = n_{GS}$, $k_{FS} = 1$.

Regardless of its popularity in implementation, this algorithm is deeply flawed. By selecting the first, and hence oldest part on the conveyor, this algorithm selects the part with the longest wait time for the feedback loop updates. Additionally, this algorithm has the major flaw that if $U_{TOTAL} > U_{CMM}$, even for short periods of time, then parts backup on the conveyors and feedback updates slow or stop.

The feedback loop is sensitive to short conveyor stoppages. As such, stoppages on the order of worst-case conveyor transit times can create additional scrap, rework, and quarantine parts. If the CMM attempts to inspect all of the parts immediately, in an effort to catch up, then the feedback loop will be left in an unstable state and the process will not be brought back under control. Parts will continue to back up on the conveyors until the CNCs are switched to manual mode or the automation is stopped. This algorithm has been found to cause the cell to suddenly stop working for no significant reason.

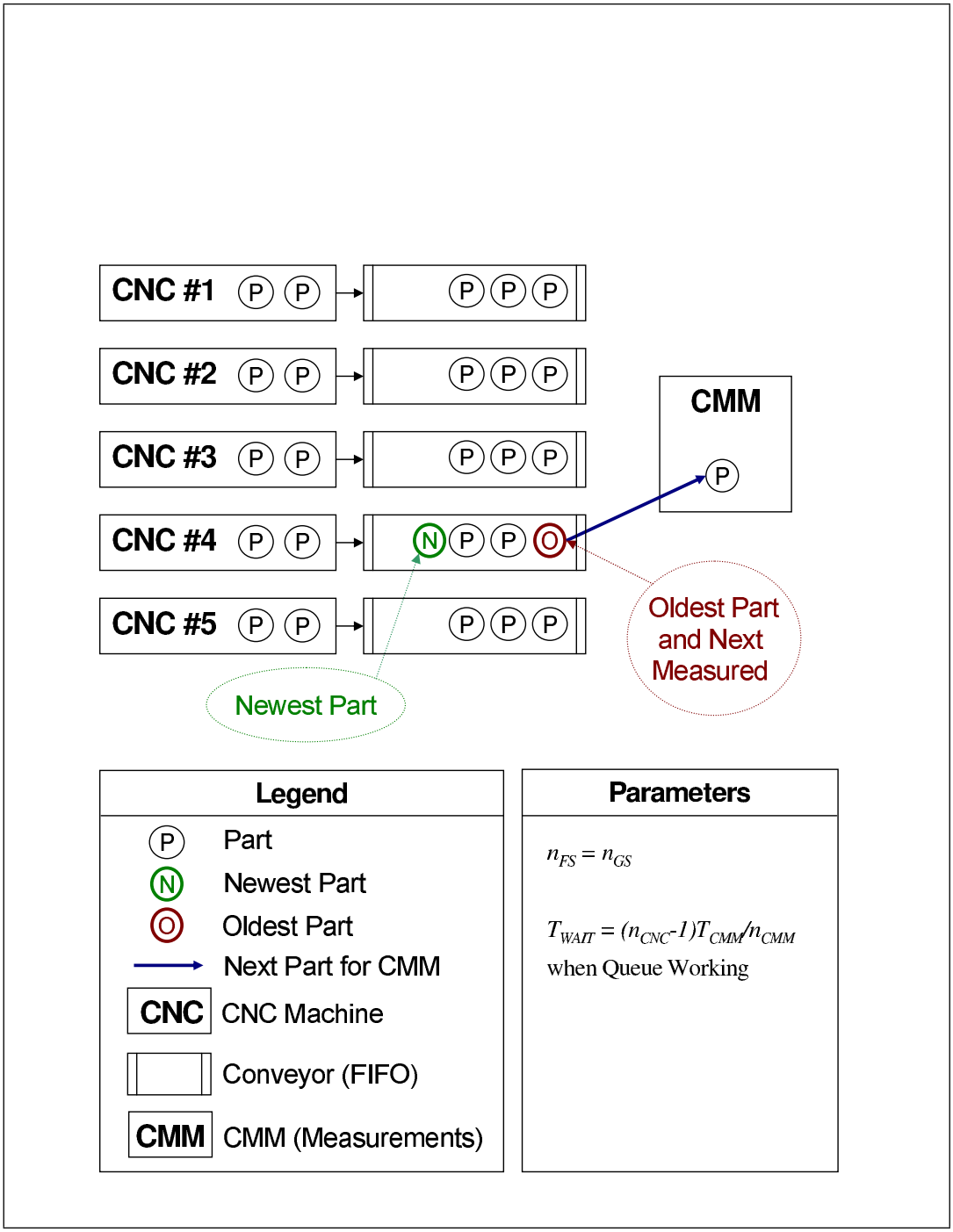

Figure 8.1: Queuing Algorithm #1: Oldest First

8.3.2 Algorithm #2: Every n_{GS}-th Part

This algorithm observes every n_{GS}-th part from every CNC. This is a very easy to understand strategy, and is compatible with discrete time control system analysis techniques that assume a fixed sampling frequency. It has the advantage that it reserves some CMM capacity for the measurement of quarantined parts.

For easy analysis, the CMM wait time and the queue wait was modeled as a MM/1 Markov Queue Process, with $\lambda = U_{CNC}/n_{GS}$, and $\mu = U_{CMM}$, yielding

$$T_{CMM} + T_{WAIT} = (\mu - \lambda)^{-1} \quad \text{for } (\mu - \lambda)^{-1} > T_{CMM} \tag{8.10}$$

If the cell is very lightly loaded, the above equation may result in $T_{WAIT} < 0$, which is impossible. In such a case, it may be necessary to assume $T_{WAIT} = 0$. This condition tends to occur only in models, as most manufacturing cells are designed for heavy resource utilization to maximize capital cost recovery.

This algorithm sets $n_{GS} = n_{FS}$.

8.3.3 Algorithm #3: Newest Part from the queue containing the oldest part

This algorithm uses a LIFO (Last In, First Out) implemented (in concept) as n_{CNC} bidirectional conveyors. The CMM observes the newest (most recent) available part from the conveyor containing the oldest (least recent) part. This minimizes $n_{GS} = n_{FS}$ and T_{WAIT}, where

$$T_{WAIT} = \frac{T_{CMM}}{2n_{CMM}} \tag{8.11}$$

On average, the newest part only has to wait for the existing CMM cycle to complete.

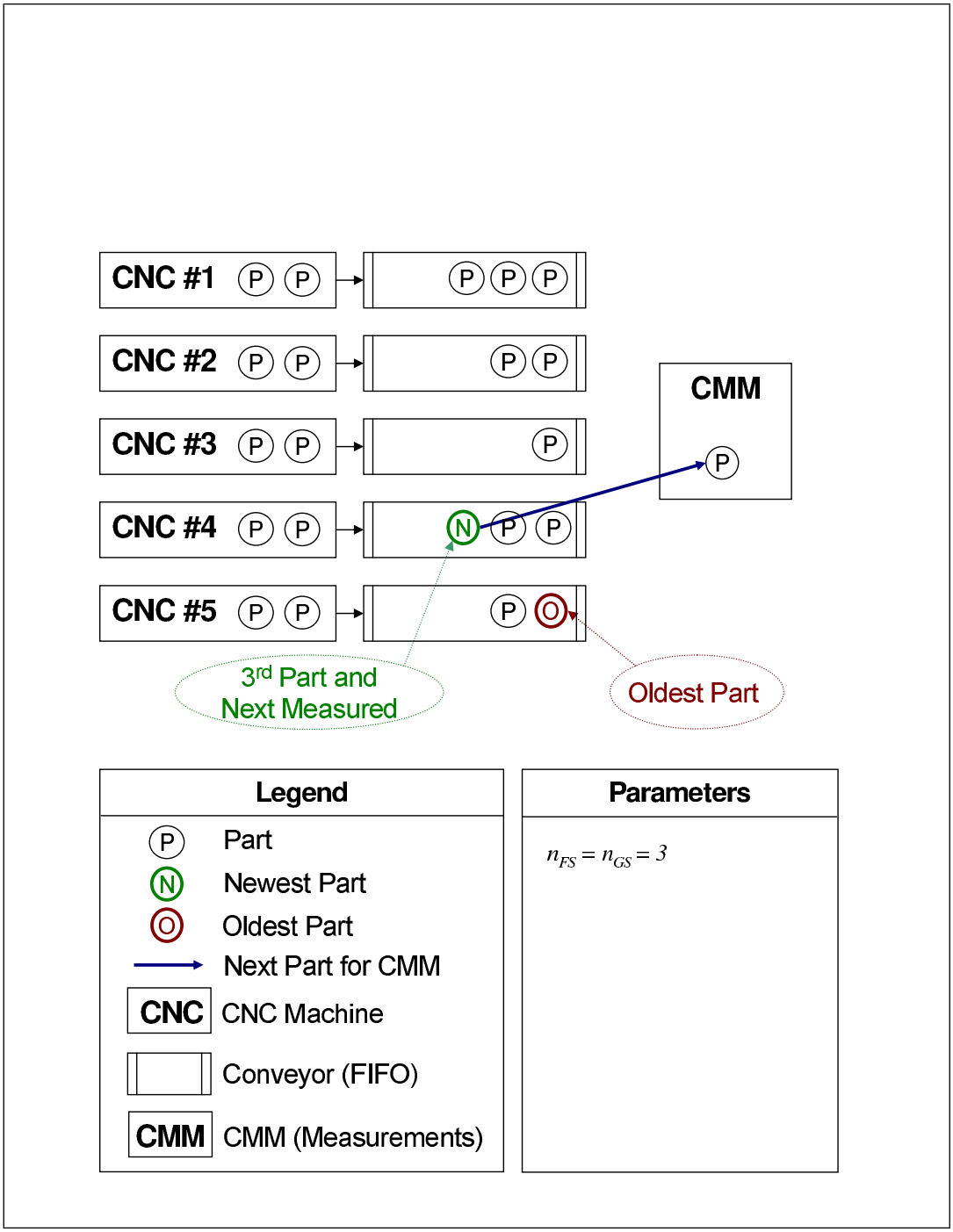

Figure 8.2: Queuing Algorithm #2: Every n_{GS}-th Part

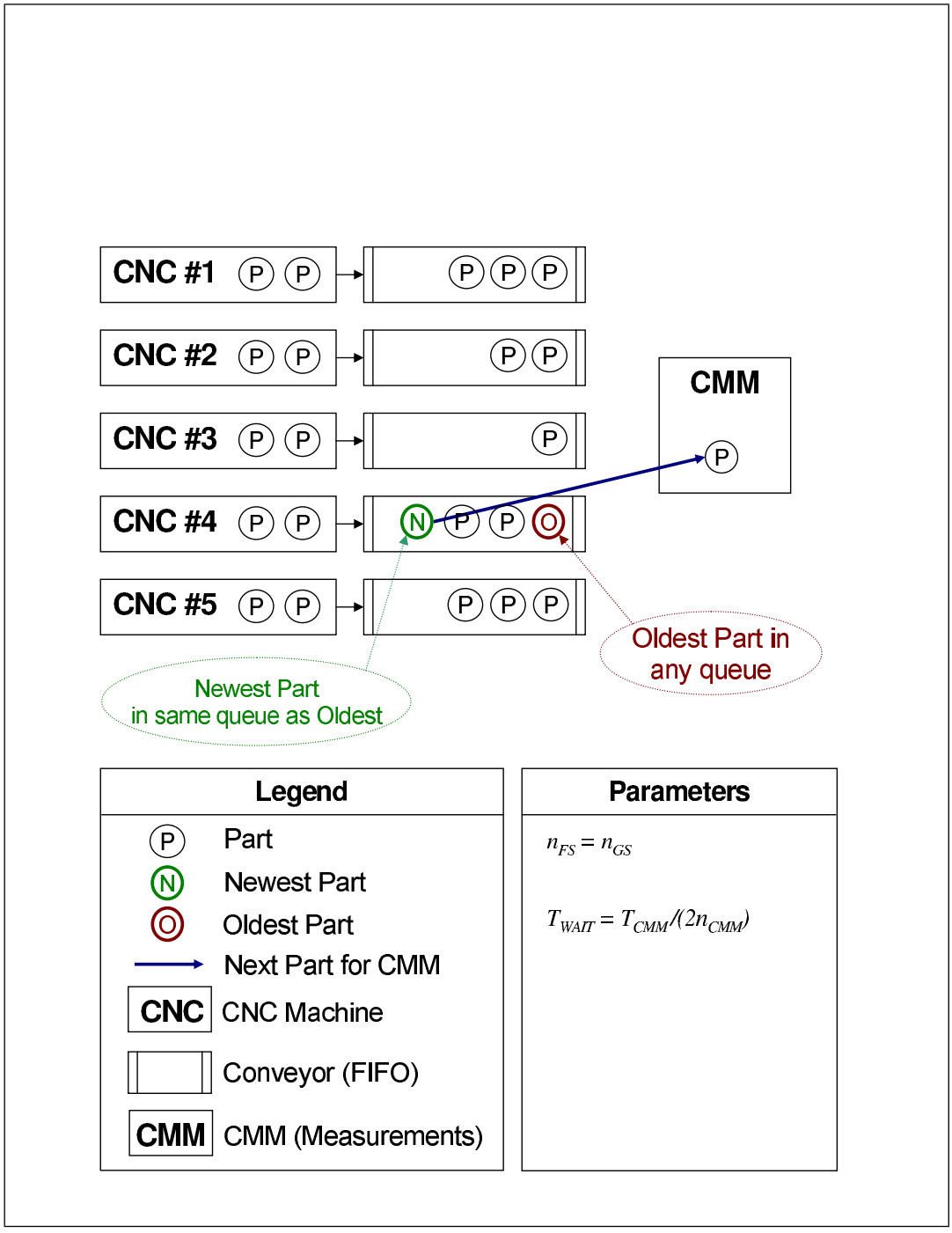

Figure 8.3: Queuing Algorithm #3: Newest part from queue containing the oldest part

This algorithm is very fair, and prioritizes all production equally. From a wait time perspective, this algorithm can only be improved upon if it is possible to accurately predict future events inside the cell. To a certain extent it is possible to predict future events in the cell if the CNCs can be assumed to have a constant cycle time. However, this assumption did not apply in practice. The cycle times on the CNC machines varied, because the cycle time on the bar feeders varied, operator intervention occurred, and the material transport system serviced multiple CNC macines. In the case of the production cells, the cycle time inconsistencies were sufficiently large that it created an unstable and difficult to predict arrival distribution at the CMM. For a system with an unpredictable arrival distribution, this algorithm yields the minimum average T_{WAIT}.

For algorithms 3 and 4, the special cause quarantine parts are best handled by creating an extra number of quarantine queues, n_Q. Every time a part is measured from a quarantine queue, a timer is set to the current time. The CMM measures the newest part from either the CNC queues or the quarantine queue containing the oldest part. When the quarantine queues are empty, this algorithm simply dedicates all available measuring capacity to keeping up with production. This results in n_{GS} simply being the ratio between the production speed and the measuring speed. As such

$$n_{GS} = n_{FS} = \frac{U_{CNC}}{U_{CMM}} \tag{8.12}$$

When all of the quarantine queues contain parts, they receive equal treatment as the CNC machines. As such

$$n_{GS} = n_{FS} = \frac{(n_{CNC} + n_Q)U_{CNC}}{n_{CNC}U_{CMM}} \tag{8.13}$$

This result is shown in Bering and Veldhuis (2010b).

8.3.4 Algorithm #4: Newest Part with Penalties

This algorithm is very similar to #3, with the change that if a CNC is consistently producing acceptable parts, the oldest part on the associated conveyor has its age reduced by a penalty. For example, with a 30 second penalty, the machines producing consistently acceptable parts will get measured on average once every 150 seconds, whereas the machines that have recently produced reject product will get measured every 120 seconds, or 30 seconds more often on average. The precise cycle times will vary depending on how many machines have parts available to be measured, and how many machines are producing reject parts. However, in the event that many parts are waiting to be measured, no machine will on-average be penalized for more than the magnitude of the penalty.

The effect of the timing penalty is to make $n_{FS} < n_{GS}$. By judiciously selecting penalties, n_{FS} can be set to 1 or 2. This permits tighter sampling on the CNCs experiencing marginal process conditions. In the event that all of the machines are running the same, this algorithm remains fair. All machines will get the same inspection rate.

Some care should be exercised in limiting the size of the penalty. If n_{FS} is made too small relative to n_{GS}, then the in-spec processes will be starved of inspections which will eventually cause them to run out-of-control. This effect may result in worse overall process performance.

8.3.5 Special Case Algorithms

§4.2 discussed the optimum performance of the feedback loop which occurs with $g \approx 1$ and $D = 1$ update-sample. Feedback loop performance improves if n_U, the update-sample rate, is made as small as possible, within the practical limitations of the cell design while maintaining a high probability that $D = 1$

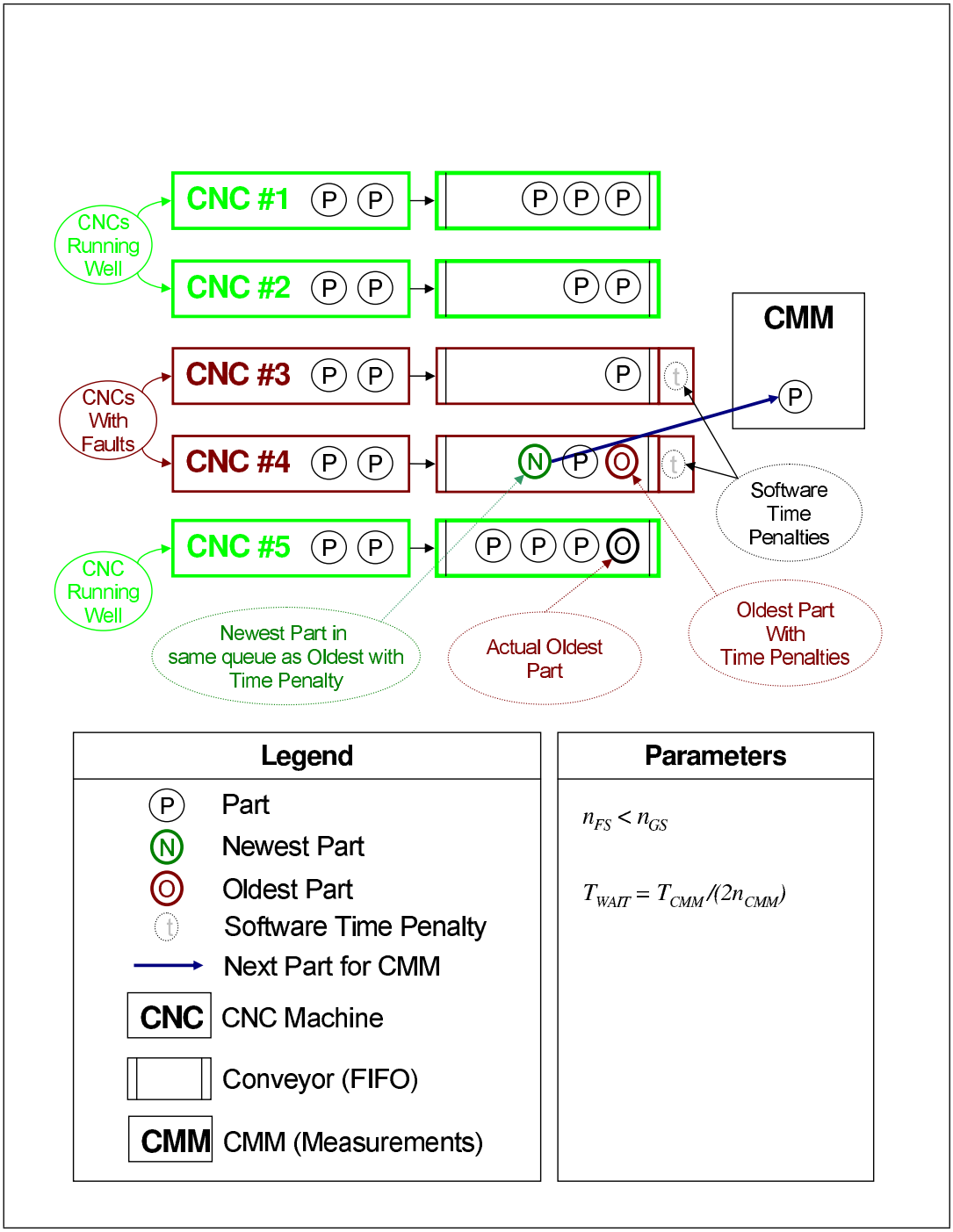

Figure 8.4: Queuing Algorithm #4: Newest Part with Penalties

update-sample. It is possible to construct a cell with D sufficiently small that these conditions are met with $n_U = 1$, 2 or 3 parts. This is accomplished by reducing the measurement time T_{CMM}, the waiting for measurement time T_{WAIT}, and all the material transport, conveyor, cooling and cleaning delays, T_{CONV}. Additionally, the delay in the CNC itself, D_{CNC}, can be reduced by moving all finished machining operations to the end of the cycle on the second (sub) spindle. The extent to which this can be actually achieved is limited by both the machining process, the cell and CNC machine design. This is shown in Figure 8.5.

In manual mode, many operators assume that they are operating the machine with $n_U \geq D$, $n_U = 2$ parts, and $n_{GS} = 1$ part. When this occurs, the second part from the machine, "the second off", reflects updated offsets. After careful study of the CNCs and operator actions in manual mode, it was observed that instances where $D = 3$ parts could occur in special conditions, and these circumstances occurred with the production parts. This obviously violates the $n_U \geq D$ assumption when $n_U = 2$ parts.

This situation occurs when the CNC exit conveyor adds a sufficiently large delay to the CNC cycle, and makes $D_{CNC} > 2T_{CNC}$ with some part programs. This situation confused the operators, and explained some of the observed behaviour. When the operators became confused, they increased n_U by waiting for the CNC to make an additional 3 to 5 parts before measuring and adjusting again.

8.3.6 Queuing Algorithm Summary

The key to minimizing delay in the feedback loop is to minimize the length of time before the newest part is observed. Minimizing this time requires minimizing part delivery times, which are limited by the cost and performance of the material transport system and the time required to cool and clean

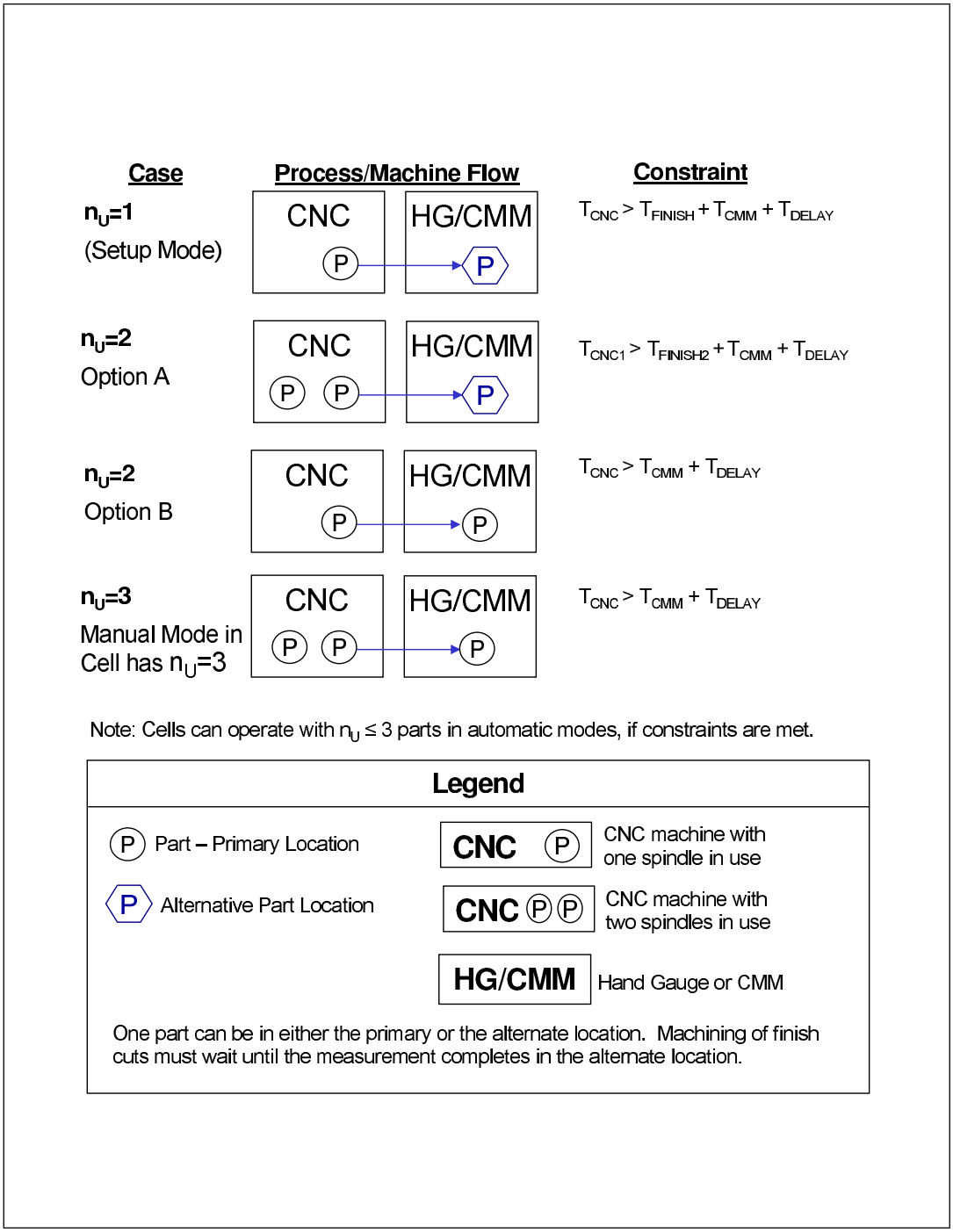

Figure 8.5: Special Case Algorithms

the parts. Additionally, assuming these systems are implemented with programmable robotic systems, the chosen material transport algorithm also has a big impact on cell performance.

8.4 Simulation of the Material Transport Algorithms

In the manufacturing cell with feedback, the control loop performance is affected by delays present in the conveyor system. These delays are affected by the total CMM load. As such, the conveyor system was simulated at two different rogue event rates. The first, $f_A = 1$ incident/day, corresponds to minimal special cause variation and infrequent rogue events. This gives an approximation of the delays experienced in the cell when the process is running very well (normally). The design goal of the cell was for the cell to operate with a bent-bar condition, which caused vibration and rapid tool wear. A bent bar would result in rogue events occurring at a rate of $f_A = 115$ incidents/day. As such, the second simulation was run at $f_A = 115$ incidents/day, and it gives an approximation of the delays experienced in the cell when the process is running at the limits of process capability. The results are shown in Table 8.1.

The Oldest First algorithm quits operating shortly after the rate of special cause variation exceeds 50 incidents/day. At $f_A = 50$ incidents/day, the oldest first algorithm was running with 8.1% scrap and 17.4% waste. This corresponds to the point at which the short term requirements for CMM capacity exceed CMM availability, the conveyors start backing up, and timely feedback updates cease.

The Oldest First algorithm caused many practical difficulties inside the cells. This algorithm was implemented with conveyors which are extremely

Table 8.1: Simulated Performance of Material Transport Algorithms

Algorithm	**Waste**[1] ω_1	**Reject**[1] c_1	**Off-line**[2] % of Total	**Reject**[3] % of Total
$f_A = 1$ *incidents / day (0.022%)*				
#1: Oldest First	12.9	6.5	-	0.15%
#2: Every n_{GS}-th Part	14.3	6.3	-	0.14%
#3: Newest Part	11.2	4.8	-	0.11%
#4: Penalties	8.3	4.3	-	0.10%
Special Case: $n_U = 3$	7.0	4.0	-	0.09%
Special Case: $n_U = 2$	6.0	3.0	-	0.07%
Special Case: $n_U = 1$	5.0	1.0	-	0.02%
$f_A =$ *1 part in 39 = 115 incidents / day (2.55%)*				
#1: Oldest First		*- Inoperative -*		
#2: Every n_{GS}-th Part	14.3	6.3 (4.9)[4]	35%	16.1%
#3: Newest Part	12.3	5.0 (5.1)[4]	33%	12.7%
#4: Penalties	9.1	4.5 (4.7)[4]	20%	11.4%
Special Case: $n_U = 3$	7.0	4.0	-	10.2%
Special Case: $n_U = 2$	6.0	3.0	-	7.7%
Special Case: $n_U = 1$	5.0	1.0	-	2.55%

Notes:

1. ω_1 and c_1 are expressed as affected parts per incident.

2. Off-line % Total is percentage parts quarantined for off-line measurement, as a percentage of total production.

3. Reject % Total is percentage parts that were made outside of part-print dimensions.

4. Numbers in parentheses are from the cell simulation based on actual production data.

popular pieces of equipment. Many automation companies deploy conveyors by default. Three of the four cells initially attempted deployed this algorithm. None of the attempts met with success. The two successful cells used Algorithms 2 and 3. As such, it is recommended that the conventional oldest first conveyor approach not be used in production cells when the part tolerances are such that the presence of the conveyors affects feedback loop performance.

Algorithm #4 performs almost as well as the special case where every part is observed and $n_U = 3$. It should be noted that the Special Case where $n_U = 1$ has an unfair advantage over the other machining processes. Specifically, $n_U = 1$ is only achievable when finishing operations take place on a

single spindle, which often means no split-offset updates occur. The general process case assumes finishing cuts on both spindles, with the resultant possibility of split-offset updates. As split offset updates were known to occur in both successful production cells, this is the reason why $c_1 = 2$ for $n_U = 1$, and $c_1 = 4$ for $n_U = 2$.

The design goal of the cell was for the cell to operate with $f_A = 115$ incidents/day of special cause variation, and this goal was met. The scrap results are within the allowable range for short-term continued operation. The penalty algorithm generates approximately 11% scrap, and this is competitive with the special case $n_U = 3$ algorithm at 10%. If high-levels of special cause variation were going to be a long-term issue for the cell, then optimizing the cell design and manufacturing process such that the $n_U = 1$ special case assumptions applied would be advantageous.

The simulated performance estimates matched the observed conditions from the production cells. When production conditions were the worst, 2 lab-grade CMMs were employed to keep up with the quarantine part flow. As the process improved, f_A was reduced, and the load on the CMMs dramatically reduced. A simulation of the live CNC system was performed, and the results are shown in parentheses in Table 8.1. Improving cell timing will reduce c_1, and the resulting cell scrap. If special cause variation is synchronous to a detectable event, this can be exploited. With 1 part in 39 variation, $n = 3$, the every n-th part algorithm can synchronize to the bar feed. This is a good argument for its deployment in the field, and why the simulation number for Algorithm #2 is lower than Algorithm #3. Otherwise, for the general case, Algorithm #4 is the best.

8.4.1 Additional Field Observations

Algorithms 1 to 3 were implemented in various production cells. Real cells

matched the ω_i predictions closely and smaller ω_i consistently resulted in improved cell performance, reduced scrap and better control action. A key subtlety, however, is that in practice n_{GS}, n_U, and n_{FS} are always integers. This means when $n_U = 2.6$, sometimes n_U will be 2, sometimes 3, and occasionally n_U may be 4. It proved vital to ensure that the PLC logic caused the automation to move parts in a manner that minimizes n_U and ω_i, and to ensure that the automation was executing the PLC commands in the manner expected and modeled. In the field, it was worthwhile to analyze all rogue events in an effort to reduce waste wherever possible. Reductions in n_U and ω_i through tweaking of automation and minimization of delay times translated into improved process results.

Reducing certain cell parameters, like n_U, n_{GS} and n_{FS} can have multiple beneficial impacts. For instance, a reduction in n_U reduces c_i and ω_i, which directly reduces scrap. However, reducing n_U also improves control action, and this also reduces scrap. Improved control action may mean that the finishing cutting tool insert may undergo reduced tool wear because, when it is properly adjusted, there is less chance of vibration and excessively deep cuts. Reduced tool wear can translate into reduced operator intervention, which can reduce f_A too. As such, there were strong synergies in the cell, compounding the impact of every improvement. Thus, reducing the severity, ω_i, and frequency, f_A, of the rogue events, through any applicable means, increased cell performance.

8.5 Summary

Waste reduction initiatives are ultimately limited by key cell performance characteristics. Importantly, the cell minimum CMM inspection capacity required by the cell can be found from (8.5). It is important to select a queuing algorithm which minimizes both the average and maximum wait

times before parts are observed. For the conveyor algorithm, the wait time T_{WAIT} can become very large, causing feedback loop instability.

When feedback loop stability is maintained by setting $D = 1$ update-sample, as discussed in §4.2, then reducing n_{GS}, n_U and n_{FS} results in improved feedback performance. Scrap is impacted directly, through (6.5), and because faster sampling generally results in faster feedback response. Fast feedback response is most effective when correcting for the rapid process changes that correspond to rogue process events. However, practical cells have limitations on how many samples they can gather cost effectively. Algorithm #4, Newest Part with Penalties, performed the best, because it attempts to balance data collection when rogue process events are present, while ensuring every machine in the cell is consistently monitored. Given sufficient inspection capacity and a suitably designed cell, subject to constraints, the special case optimizations deliver the minimum achievable process waste.

The delays present in the feedback loop, including T_{WAIT}, are critical to feedback loop performance. As such, when new cells are being constructed, it is recommended that conveyor delays and cell timing data be captured and profiled.

Chapter 9

Spring Orientation

9.1 Application Overview

The first two industrial applications deal with the control system and the material transport system in a production cell. As such, it is appropriate that the third industrial application focuses on the third major element of a production cell: the gauging system. This gauging application is a high-reliability application where certain classes of gauging error (outgoing defects) have large economic costs associated with them. This application provides the best illustration of the debugging capabilities of the runtime profiler, because unlike the Manufacturing with Feedback example, line speeds and production volumes were far too large for a person to cope with. As such, automated tools were required.

The spring-orientation application is an example of what this thesis terms a "true-attribute" gauge. The AIAG GR&R (AIAG, 2003) standards specify procedures for attribute gauges. These depend on assuming a known underlying repeatable continuous characteristic exists that can be linked to gauge error. In effect, measurement inaccuracy is due to the existence of a "gray" zone in which the underlying characteristic can be misinterpreted.

The true-attribute gauges examined here did not have a single underlying repeatable characteristic. They monitored many characteristics, and if the part was presented in different orientations, the observations were not repeatable. This creates a challenge in estimating the performance of these gauges.

The goal of this chapter is to show how rogue events analysis and runtime profiling can be employed in this application to examine the effects of rogue events on algorithms and identify algorithms that produce fewer outgoing defects.

9.2 Spring Orientation System

The previous sections dealt with applications that responded well to semi-automated data analysis, and the results could be understood in the context of simple graphs. Many industrial applications yield large volumes of data, and as such, some method of automated processing is required. As such, this section describes algorithms from an industrial application involving large amounts of data and parts being processed too quickly for human analysis. The profiler used to test the individual algorithms will be described in the following section.

The customer required an industrial noncontact machine-vision and gauging system to inspect flexible parts. The challenge was to select an algorithm that could reliably detect if the parts were correctly oriented (right side up) for the next phase in the production sequence. Figure 9.1 shows the correct and incorrect orientations. The parts had only a few stable orientations, and mechanical safeguards prevented all but 2 orientations (right side up and upside down) from being processed by the next machine. The absence of intermediate orientations and the clustering of outgoing defects meant that many of the marginal detection issues dealt with by

Wall (1997) and the GR&R procedures (AIAG, 2003) were not obviously applicable or were ineffective.

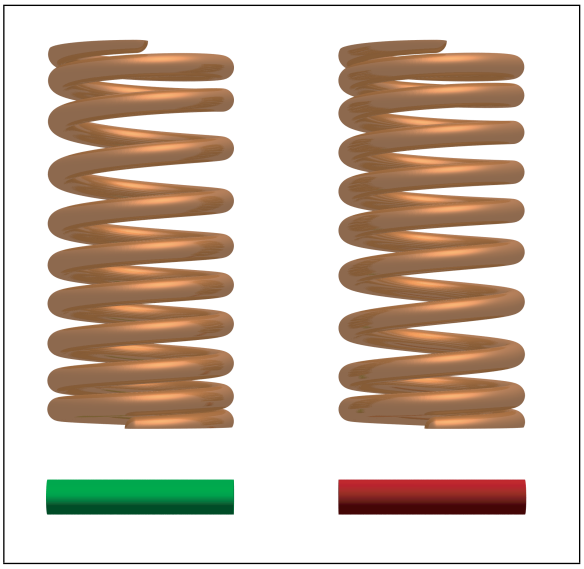

Figure 9.1: Correct (Left/Green) and Incorrect (Right/Red) Spring Orientations

Unfortunately, detecting spring orientation is a challenging operation, and presents a number of practical problems. As might be gathered from Figure 9.1, the difference between the orientations is subtle. People can spot the difference, but double human inspection has a detectable outgoing defect rate in multimillion part samples. Another issue is that springs are flexible. They readily bounce and deform. Occasionally the vision system inspects springs in flight as they bounce along the conveyor. Together these results indicate that it may not be possible to accurately detect spring orientation, even though spring manufacturing experience indicated otherwise.

Together all these factors make it difficult to determine which algorithms, if any, will meet the customer's requirements. This led to the purchase of many different systems in an attempt to find one that worked. All of the

purchased systems failed to meet reliability expectations, and no one understood why. It was extremely difficult to detect any outgoing defects whatsoever, and this left open the question of whether the cause of the defects was software, hardware or operator related. Additionally, as no one was able to differentiate between a reliable and an unreliable system, it was difficult to even specify the purchase of new equipment. This system was of both academic and practical interest.

9.3 Methodology

In production, multiple inspection systems are used in sequence to monitor all of the critical characteristics for the parts. This particular gauge checks if the orientation is correct, and if it is, then it outputs an accept signal. All other failures, including any measurement failures, results in a reject signal.

Measuring one lot of parts in the gauge results in the following two groups of parts

Accepts or Total Accepts - the number of parts accepted by the gauge.

Rejects or Total Rejects - the number of parts rejected by the gauge.

These two groups can be further separated. The Accepts can be divided into

True Accepts - the number of defect-free parts in the accepts. For any reliable gauging algorithm, this number is almost the same as Total Accepts.

Outgoing Defects - the number of defective parts in the accepts. This should be zero.

Similarly, the Rejects can be divided into

Process Rejects - these rejects correspond to defective production.

Measurement Rejects - these rejects correspond to good production that was rejected because of a rogue measurement event.

Technical issues exist both with classifying the accepts and rejects, and the next sections detail the issues and the experimental controls to solve them.

9.3.1 Finding All of the Outgoing Defects

The production cell was tested with a variety of techniques, before the run-time profiler and the related tools were developed. The following techniques were attempted with no success:

1. Manual Secondary Inspection - These parts can be inspected by hand, visually, and their orientation determined. However, when people inspect massive numbers of similar parts, then they start assuming all parts are the same. At the peak of teh effort, every part was being checked by hand by four different operators, and outgoing defects were being found in the hand inspected batches.

2. Automated Secondary Inspection with unreliable algorithms - More testers were purchased to reinspect the same part multiple times. However, this approach failed because if the part was inspected once by an algorithm that did not work well, then repeated observations by the same algorithm risks generating identical results, even if independent observations are gathered on the same part.

Aside from being ineffective, using secondary inspection testing has another issue. It can only prove the existence of outgoing defects, and it does not clearly indicate their underlying causes. For instance, in a production cell, an outgoing defect could be caused by a software problem, an electrical problem, a mechanical problem, or an operator problem. By themselves, secondary inspections cannot confirm the existence of a software problem, when other explanations are possible.

These issues motivated the development of the runtime profiler, which could capture extensive information on program execution. From the captured data, two new and reliable algorithms were developed for spring orientation testing. These permitted the generation of a database of confirmed good production, process rejects, and measurement rejects. Operation of the new and reliable algorithms was also checked by generating a database of one hundred million samples of known orientations, to ensure that no new defect modes would be found. Additionally, the new and reliable algorithms were deliberately designed to be diverse. For instance, one algorithm (not described in this thesis) used out-of-focus image regions to determine spring orientation, while a complementary algorithm (described later in this chapter), used the in-focus image regions to determine spring orientation. All of these checks gave us some confidence that we had reliable algorithms to determine spring orientation. At this point the theory presented in Chapter 4 was developed to help us understand the differences between the reliable and unreliable algorithms.

9.3.2 Process, Measurement, and Combined Rogue Defects

Following the method from Chapter 4, it is desirable to ensure that the probability of the combined rogue distribution occurring in the pass-zone of the score function is minimal. Effectively, this results in four groups of

practically acquired images

1. Obvious Process Rejects - These are the springs that are clearly upside down.
2. Obvious Measurement Rejects - These are springs that are clearly measurement errors, and all algorithms agree on this.
3. Marginal Measurement Rejects - Based on the same image data, some algorithms accept these parts, and some algorithms reject these parts.
4. Combined Process and Measurement Rejects - A collection of image exists, largely caused by feeder problems, and in these images, it is impossible to tell what the correct spring orientation is. For almost all of these parts, more than one spring is in the field of view, making it reliably identify the features on the spring of interest.

Different measuring techniques require different areas of the part to be in-focus. One of the issues with categories 3 and 4, is that some algorithms can actually identify spring orientation. However, from a reliability point of view, it would be a poor idea to push the limits of the algorithms capability this hard, because it might cause unexpected outgoing defects.

To simplify the accounting, the above four categories were simply reduced to total rejects and measurement rejects. Total rejects represents the sum of all four of the above classifications, and represents the cost to the supplier of all reject (waste) production. Measurement rejects represents the sum of items 2 through 4. Measurement rejects are similar to producer's risk (Type I Error), in that they represent the savings available if the equipment could be made perfectly reliable. In particular, If all of the feeder, material transport, image acquisition and natural bounce problems associated with these parts could be eliminated, then in theory no measurement rejects would exist.

In application, some of the measurement rejects are unavoidable as they are caused by the nature of the feeding process, and the bouncy nature of the flexible parts (springs) being inspected. However, the run-time profiler also detected the presence of an additional group of measurement rejects where the rogue measurements were being caused by software issues (not mechanical issues.) Once the underlying causes of this group of software related rejects was understood, then software improvements permitted resulted in fewer measurement rejects. The cost savings in waste reduction was large enough cost savings to cover the cost of the additional profiling hardware.

9.3.3 Test Design

The end result of the new approach was that we had captured over one million parts, and had a database that indicated with reasonable certainty which parts were correctly oriented, which parts were incorrectly oriented, and which parts were known to have measurement issues. Given this information, multiple algorithms could be tested. The next section describes the three most commercially and theoretically significant algorithms that were analyzed.

9.4 Vision System Algorithms

Three possible algorithms are considered in this thesis. The first two were considered the most “provable”, in that it was possible to make strong software engineering arguments of algorithmic correctness. As such, they were deployed onto a production line. These three algorithms were selected because two of them were actually deployed in production, and the third was the recommended replacement.

9.4.1 Algorithm #1: Intensity

The intensity algorithm looked at the average intensity of the image over a specific area, and compared it to a preset value. If the average intensity was below the preset, the part was correctly oriented. The strength of this algorithm is that it depends on a simple, easy to observe, inherent property of the part. This creates a robust argument for algorithmic correctness.

The weakness of the intensity algorithm was the field testing showed outgoing defects. Figure 9.2 shows how this might happen when clutter is present in the image. The fundamental problem with using engineering principles to demonstrate an algorithm's effectiveness is that the approach fails if the engineer fails to foresee all the rogue events the system will experience.

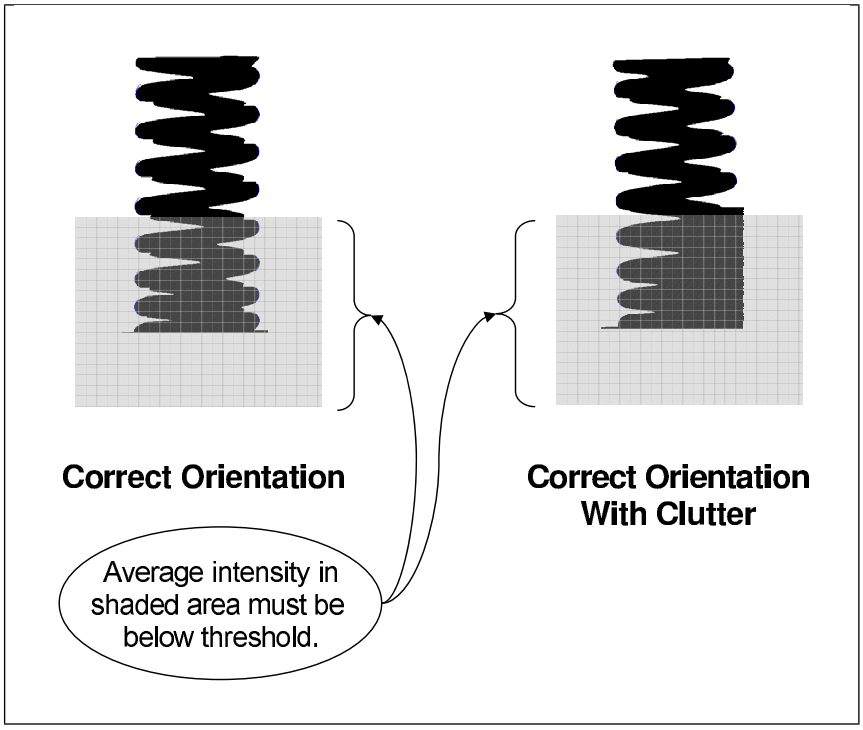

Figure 9.2: Vision System Algorithm #1: Intensity

The intensity algorithm inspired the model in §4.3.3.

9.4.2 Algorithm #2: Ratio

As shown in Figure 9.3, the ratio algorithm observed the distance between two features, and divided it by the distance between two other features. If the ratio was greater than the preset, then the part was correctly oriented. The strength of this algorithm was that it was based on a known mechanical property of the part.

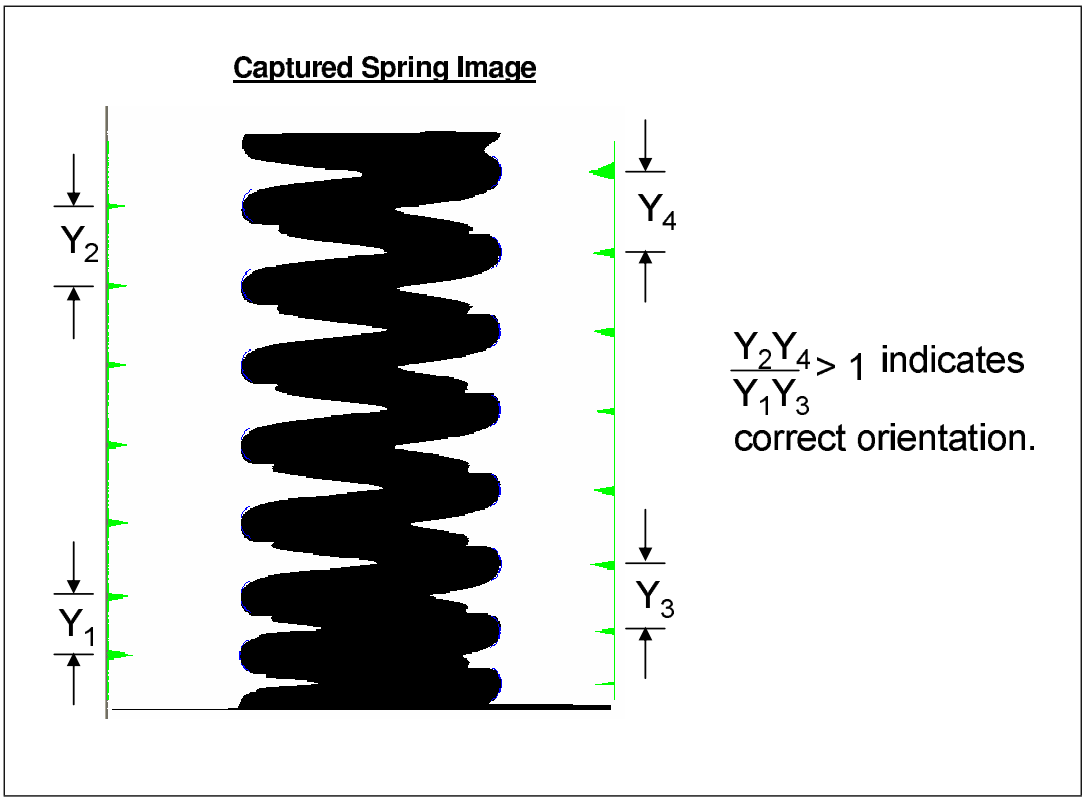

Figure 9.3: Vision System Algorithm #2: Ratio

The weakness of the ratio algorithm was that failures in the feature recognition algorithm could theoretically cause failures in the ratio algorithm, which would result in outgoing defects. Developing a robust feature recognition algorithm was a challenge in this application, as none of the tested algorithms was robust against all issues encountered in production.

As such, error margins were included inside the ratio algorithm in an effort to ensure that it would never generate an outgoing defect.

The ratio algorithm is the algorithm modeled in §4.3.5.

9.4.3 Algorithm #3: Pattern

The pattern algorithm observed the position of all of the in-focus features, normalized the positions to a preset length, and checked if they fit a preset pattern. The strength of this algorithm was that it forced recognition of all the features. As such, a failure in the feature recognition algorithm should not cause an outgoing defect.

In the spring industry, specialized machines are used to measure the pitch profile of certain classes of springs. These machines work by determining the position of the coiled and unground spring at many different points along its length. The pattern algorithm implemented a similar approach. From one camera view, it measured the height of each spring coil present in the spring image, shown as Y_p in Figure 9.4. This was graphed against the coil spacing (difference between two consecutive heights), shown as ΔY. The resulting interpolated $\Delta Y(Y_p)$ values become the multivariate Y observation used in the multivariate analysis in §4.4. The resulting interpolated pitch-profile graph is shown as a green line in Figure 9.4. The red zones are boundaries which the green line is not supposed to cross. Crossing a boundary is detected as an incorrectly oriented spring.

A key challenge facing the pattern algorithm was that no one knew if it was possible to construct accurate pitch-profile graphs from a single camera view of a ground and heat treated spring. The other pitch profile systems used many camera views and unground springs, which could be measured much more accurately. Additionally, the univariate models suggested that pitch-profile decisions might be marginal. Multivariate (M-SPC, PCA) testing was not done until the development of the runtime profiler. The results

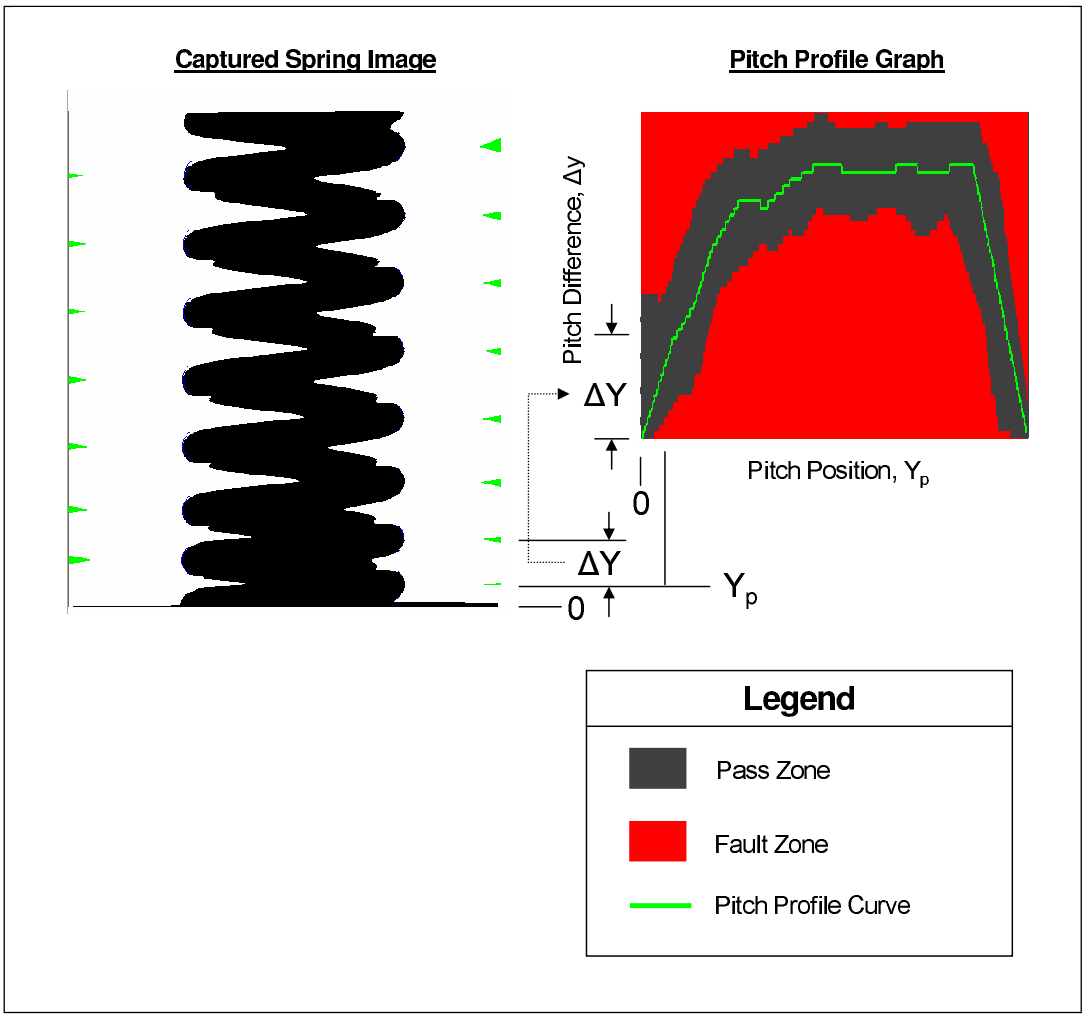

Figure 9.4: Vision System Algorithm #3: Pattern

of this analysis resulted in the model provided in §4.4.5.1.

The runtime profiler provided many insights into the performance of this algorithm. For instance, the BCS algorithm was very effective at detecting unusual code paths that led to unnecessary rejects. Nevertheless, the possibility existed that other sources of rogue variation might be present, and the runtime profiler was used to look for these. It was estimated that the reductions in unnecessary rejects would provide sufficient financial returns to cover the cost of the profiling.

9.5 Simulation Results

The data presented in Table 9.1 are from a million-part data sample gained by profiling the Algorithm #2 Ratio, with production parts.

Table 9.1: Results from 1 034 788 Part Sample

Algorithm	**Accepts**	**Total Rejects**	**Measurement Rejects**	**Outgoing Defects**
Currently in-use production process[*1]*:*				
2. Ratio	93.42%	6.58%	1.92%	98.7 ppm
Revised Algorithms (after profiling):				
1. Intensity	97.68%	2.31%	0.16%	0.71%
2. Ratio	97.15%	2.84%	0.19%	0.15%
3. Pattern	96.65%	3.35%	0.47%	0 ppm
Manual (Human) Cross-Check:				
Visual Inspection	96.71%	3.29%	0.41%	17 ppm

Notes:

*1. The Ratio algorithm was deployed on-line with two levels of redundancy. The results above are from both levels combined together.

The primary insight from Table 9.1 was that the pattern algorithm was a much more accurate algorithm both in terms of outgoing defects and measurement rejects. Additionally, algorithm #1 (Intensity) and algorithm #2 (Ratio) had detectable issues with rogue and combined rogue distributions.

As a practical matter, human inspection is never perfectly reliable. In production, there were 17 parts (17 ppm) where the human cross-check passed defective product. Subsequent review of all the unusual classifications showed that the pattern algorithm was accurate (and that people make mistakes).

Based on the human cross-check results, 2.88% of the population corresponded to obvious process rejects, and 0.41% of the population where obvious measurement errors. Specifically, rogue measurement events happened with sufficient frequency that 0.41% should not have been accepted because the measurements were sufficiently corrupted that the results would be suspect. Thus, the pattern algorithm rejected all of these suspect parts. However, the Ratio and Intensity algorithms accepted a portion of these parts, which is less than ideal.

9.6 System Reliability Estimation

9.6.1 Algorithm #1: Pattern

The failure in the algorithm #1 is obvious. Clutter will cause rogue gauge error, and it will make the test ineffective. This could be easily demonstrated in application.

9.6.2 Algorithm #2: Ratio

The model presented in §4.3.5 was designed to show the active statistical effects present in algorithm #2. As such, in order to estimate P_E, the α and β values need to be updated to match the empirical values. From Table 9.1,

assume

$$\begin{aligned} \alpha &= 0.0284 + 0.0015 && (9.1)\\ &= 0.0299 && (9.2)\\ \beta &= 0.0019 + 0.0015 && (9.3)\\ &= 0.0034 && (9.4) \end{aligned}$$

From §4.3.5

$$\begin{aligned} U_1 &= 1.320 && (9.5)\\ U_2 &= 2.182 && (9.6) \end{aligned}$$

The combined rogue distribution in the pass-zone has mean $\mu = 1.434$ and variance $\sigma^2 = 0.116$. Using the approximation that the dominant combined rogue distribution $Y_D \approx \ln Y_R$ from (4.146) yields

$$\begin{aligned} \rho_{24} &= \Phi\left(\frac{U_2 - \mu}{\sigma}\right) - \Phi\left(\frac{U_1 - \mu}{\sigma}\right) && (9.7)\\ &\approx 0.838 && (9.8) \end{aligned}$$

From (4.148), the resulting P_E is

$$\begin{aligned} P_E &= 2\alpha\beta(1-\beta)^3\rho_{24} && (9.9)\\ &\approx 1.69 \cdot 10^{-4} && (9.10)\\ &\approx 169 \text{ ppm} && (9.11) \end{aligned}$$

The previous sections used ω_i and c_i to reflect the number of parts quarantined and scrapped (reworked) in each quality incident. In this application, defects clustered in groups from 7 to 10 related parts, where a typical value of $c = 9$ can be assumed. The actual value from the simulation was $P_E = 1500$ ppm. Assuming a cluster size of $c = 9$ parts, $\frac{P_E^*}{c} = 167$ ppm, which matches $P_E \approx 169$ ppm.

It should be noted that accurate predictions like this are uncommon in practical systems. The goal is to determine if the algorithm is capable of detecting the first instance of any defect cluster, and this algorithm is definitely not capable of this. Additionally, probability predictions in real world systems have many practical issues, including the following

1. Often X and W are not perfectly independent. From the data, rogue gauge error occurs with an observed probability of $\frac{0.015}{0.0284+0.0015} = 0.0502$ when rogue process error is present, and it occurs with an observed probability of 0.0019 when rogue process error is not present. Thus, independence is not present in the raw data.

2. Issues with independence are often caused by clustering. Clustering effects usually indicate that an underlying physical cause is affecting the assumptions in the statistical model. It is important to track operator activity as it relates to clusters of events. Often the operator is either fixing a problem (which caused the cluster) or the operator made a mistake (which caused the cluster.) In this application, when the captured data was reviewed it was obvious that clusters of failures were occurring.

3. The statistical model and simulations are often simplifications of the true complexity of the real world manufacturing process.

As such, from a practical point of view and for small P_E, achieving $P_E \approx P_E^*$ within two orders of magnitude is a great result, and a far better result than any other technique that was attempted on this particular production line.

It might not be obvious why a result accurate within two orders of magnitude would be a great result. Runtime profiling and rogue event analysis revealed

➤ all the outgoing defects in this example;

- ➤ all of the results in Table 9.1 and in this section;
- ➤ the existence of clusters;
- ➤ that X and W were not perfectly independent; and,
- ➤ that this algorithm was incapable of finding the first instance of a potential outgoing defect.

From a practical point of view, finding the first instance of any defect cluster is vital, because it represents the start of the quarantine zone. Without an accurate quarantine zone, it is impossible to do the necessary checks to find the size of the cluster, and hence c_i. Without an accurate c_i, it is impossible to calculate the defect rate. In short, without the collected information, it is not known if the outgoing defects exist, how large the clusters are, or if the gauge can accurately detect the beginning of a cluster of outgoing defects. *Getting an accurate prediction with collected information is far easier with the tools developed in this thesis than without them.*

It is also possible to compare the predictions from the model to the initially captured information from the production line, which used a different implementation of the same ratio algorithm. Factoring in clustering, the empirical result was $\frac{P_E^*}{9} = \frac{98.7 \cdot 10^{-6}}{9} = 10.9$ ppm. Using $\alpha = 0.0658 - 0.0192 = 0.0466$ and $\beta = 0.0192$ results in $P_E = 1.414 \cdot 10^{-3} = 1414$ ppm. However, this algorithm was marketed as having three independent levels of redundancy. The following P_E predictions are for the combined reliability of a system with n-independent levels of redundancy, where individually each level had an outgoing defect rate of 1414 ppm.

- ➤ One level of redundancy implies $P_E = 1414$ ppm.
- ➤ Two levels of redundancy implies $P_E^2 = 2.0$ ppm.
- ➤ Three levels of redundancy implies $P_E^3 = 0.0028$ ppm.

➤ The empirical result was 10.9 ppm.

The issue with redundancy is that many systems are not truly redundant, as the "redundant" observations have hidden dependencies. A review of the application revealed that the third level of redundancy was ineffective, and the second level would not be as effective as the first, hence the above results. The practical system behaved better than a singly redundant system and modestly worse than a doubly redundant system.

An important observation from the runtime profiling data was that redundancy does not yield the anticipated returns, as the redundant observations had hidden dependencies. Thus, neither the claimed "three levels of redundancy" nor the deployed "six levels of redundancy" actually worked. If the input data has hidden dependencies, the recommended solution is to use a multivariate method.

9.6.3 Algorithm #3: Pattern

The third algorithm, pattern matching, used the runtime profiler to double check its results. The runtime profiler implemented the $g_Q(Y)$ score function from (4.161) that double-checked that the captured observations were within control limits. The pattern algorithm implemented a pass-zone that is thought to be an effective approximation of this pass-zone.

A great deal of computational effort was invested in attempting to determine if either the pattern algorithm or equivalently the $g_Q(Y)$ multivariate pass-zone were ineffective. Test runs of up to 100 million randomly generated test parts were commissioned. These test runs showed no outgoing defects either. From §4.4.5.1, the predicted P_E was extremely low $P_E = 2.7 \cdot 10^{-128}$. There is no way to confirm these estimates experimentally. It must be assumed to be correct within the confidence limits given in §4.4.6.3 given no detected failures on any of the test runs.

An important industrial result was that the runtime profiler / pattern algorithm found an unexpected rogue defect, a constant-pitch spring. Springs with a constant pitch would cause problems if they were shipped to the customer. Both the profiler and the pattern algorithm detected the constant pitch spring. Given the presence of this unusual defect, it is important to remember that §4.4.6.3 predicted that previously unknown rogue events could be a more significant source of issues than the expected combined rogue events for this system. The straight spring was not an outgoing defect in this particular case, but it easily could have been. Periodic review of the captured profiling data is needed to detect these unusual line events.

9.7 Summary

Runtime profiling was of significant benefit in this application. This application had caused the industrial partners much grief and expense. Highly reliable systems with no obvious gray zones render many techniques including GR&R, variable acceptance sampling, and large batch attribute acceptance sampling ineffective when attempting to isolate infrequent failure modes. Algorithm #2, Ratio, was sufficiently effective that no one ever demonstrated a problem with the gauge software until profiling data were examined by secondary algorithms.

The vision system also suffered from the issue that line speeds were sufficiently fast that most conventional debugging techniques would have been ineffective, even if sampling methods detected a problem in a timely manner. Firstly, a large volume of state information must be captured to understand the program execution sequence. Capturing a defective product does not explain how the algorithm failed. Secondly, the profiling data revealed the algorithm was failing while executing the intended “accept” path. The algorithms were being affected by combinations of rogue events, and

combined rogue events were generating the outgoing defects. As such, the Rogue Event Analysis methodology is required to understand which algorithms are vulnerable to interactions of rogue events, and which algorithms are not vulnerable.

The result of this analysis indicates that the pattern algorithm is highly effective at detecting correct spring orientation, and has a less than 1 ppb probability of outgoing defects due to incorrect orientation with a 95% confidence interval at 111 ppb. As the sample size becomes larger, the confidence interval will fall. As such, it is recommended that runtime profiling be continued on an ongoing basis.

Chapter 10

Conclusion

Statistical analysis of rogue events provides insights into how infrequent events impact production cells in economically significant ways. A new runtime profiler was demonstrated that could detect these effects, and a statistical model was developed to aid in understanding the significance of rogue events in relation to zero defect requirements. It provided methods to understand which algorithms will be effective inside systems affected by random rogue events, and the probability equations that can be used to improve them.

Three industrial applications were used to highlight the value of these contributions, and all of the applications are of theoretical and significant economic significance. The first industrial application involved CMM to CNC feedback algorithms, and selecting the most appropriate algorithm for the application. It solves the problem of how to keep the feedback loop stable in the presence of variable queue timing. This was published in Bering and Veldhuis (2010c).

The second industrial application examined how the material transport system affected overall cell productivity, and the CMM to CNC feedback loop. Timely feedback loop updates translate directly into improved product

quality. Additionally, if some CMM capacity is reserved to measure quarantined parts, then the cost of rework outside the cell is minimized. As shown in Chapter 8, selecting of the optimal material transport algorithm balances these two priorities. The results were published in Bering and Veldhuis (2010b).

Combined rogue events can affect the performance of a gauging system. For a high-reliability true-attribute gauge, it is recommended that a multivariate score function be used to define the pass-zone as described in Chapter 4. An industrial high-reliability gauging application was examined in Chapter 9. It was shown that of several possible algorithms, only one had the desired statistical properties that would minimize the costs of outgoing defects. Additionally, the benefits gained from reduced waste in the gauging application more than offset the additional cost of the runtime profiling used to collect the data. The runtime profiling methodology and application results were published in Bering and Veldhuis (2010a).

When deployed in manufacturing, this work has the potential to change the economics of new machine, cell, and gauge startups significantly, and to create a competitive advantage for the Canadian manufacturing sector.

References

AIAG (2003). *Measurement Systems Analysis*. Automotive Industry Action Group (AIAG), Cincinnati, OH.

Bagshaw, R. and Newman, S. (1999). Quality information feedback within a contemporary metalworking smaller manufacturing enterprise. In *Proceedings of the Institution of Mechanical Engineers*, volume 213 Part B No. 5, pages 533–538. Sage Publications.

Bagshaw, R. W. and Newman, S. T. (2002). Manufacturing data analysis of machine tool errors within a contemporary small manufacturing enterprise. *International Journal of Machine Tools and Manufacture*, **42**(9), 1065 – 1080.

Beaumont, G. (2005). *Probability and Random Variables: Mathematical and Computational Methods, Software Development, Applications*. Mathematics, statistics, operations research. Horwood Pub., Chichester, West Sussex.

Bering, T. and Veldhuis, S. (2010a). A quality framework to check the applicability of engineering and statistical assumptions for automated gauges. In *Proceedings of the 2010 IEEE Conference on Automation Science and Engineering (CASE)*, pages 319–325. IEEE.

Bering, T. and Veldhuis, S. C. (2010b). Effects of cell layout and cell control strategies on multi-machine manufacturing cells with cmm feedback. In *Proceedings of the 20th International Flexible Automation and Intelligent Manufacturing Conference, FAIM 2010*, pages 709–718. University of California.

Bering, T. and Veldhuis, S. C. (2010c). Performance of wear compensation strategies in multi-machine cells with cmm feedback. *Transactions of the North American Manufacturing Research Institute of the SME*, **38**(1).

Bering, T., Veldhuis, S., Bryson, G., and Detmers, P. (2002). Closed loop feedback control of production part errors. In *American Society of Mechanical Engineers (ASME) International Mechanical Engineering Congress and Exposition (IMECE-2002)*, number IMECE2002-39088, pages 249–257. ASME.

Brandin, B. A. (1996). The real-time supervisory control of a manufacturing cell. *IEEE Robotics and Automation*, **12**(1), 1–14.

Buzacott, J. A. and Yao, D. D. (1986). Flexible manufacturing systems: A review of analytical models. *Management Science*, **32**(7), 890–905.

Chen, N., Zhou, S., Chang, T.-S., and Huang, H. (2008). Attribute control charts using generalized zero-inflated poisson distribution. *Quality and Reliability Engineering International*, **24**(7), 793–806.

Chen, Y.-K. (2004). Economic design of control charts for non-normal data using variable sampling policy. *International Journal of Production Economics*, **92**(1), 61 – 74.

Clopper, C. and Pearson, E. (1934). The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, **26**(4), 404–413.

Crosby, P. B. (1979). *Quality is Free*. McGraw Hill, New York, NY.

Daniel, S. J. and Reitsperger, W. D. (1991). Linking quality strategy with management control systems: Empirical evidence from japanese industry. *Accounting, Organizations and Society*, **16**(7), 601 – 618.

Danila, O. (2012). *Assessing Binary Measurement Systems*. Ph.D. thesis, University of Waterloo, Waterloo, ON.

Danila, O., Steiner, S., and MacKay, R. (2012a). Assessing a binary measurement system with varying misclassification rates using a latent class random effects model. *Journal of Quality Technology*, **44**(3), 179–191.

Danila, O., Steiner, S., and MacKay, R. (2012b). Assessing a binary measurement system with varying misclassification rates when a gold standard is available. *Technometrics*. Accepted for publication Sept. 2012.

Davies, R. B. (1980). The distribution of linear combinations of χ^2 random variables. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, **29**(3), 323–333.

Deming, W. E. (1986). *Out of the crisis*. Massachusetts Institute of Technology, Center for Advanced Engineering Study, Cambridge University Press, Cambridge, MA.

DesRosiers, D. (2011). Year in review - number 43 - capital expenditures - assembly vs parts vs truck body. In *DesRosiers Automotive Reports - Email Report*, Richmond Hill, ON. DesRosiers Automotive Consultants Inc.

Dickinson, W., Leon, D., and Fodgurski, A. (2001). Finding failures by cluster analysis of execution profiles. In *Software Engineering, 2001. ICSE 2001. Proceedings of the 23rd International Conference on*, pages 339–348. IEEE.

Elbaum, S. and Hardojo, M. (2004). An empirical study of profiling strategies for released software and their impact on testing activities. *SIGSOFT Software Engineering Notes*, **29**(4), 65–75.

Esbensen, K. H., Guyot, D., Westad, F., and Houmller, L. P. (2002). *Multivariate Data Analysis - in Practice: An Introduction to Multivariate Data Analysis and Experimental Design*. Camo Software Inc., Woodbridge, NJ, 5th edition.

Florac, W., Carleton, A., and Barnard, J. (2000). Statistical process control: analyzing space shuttle onboard software process. *Software, IEEE*, **17**(4), 97 –106.

Frankl, P. and Weiss, S. (1993). An experimental comparison of the effectiveness of branch testing and data flow testing. *Software Engineering, IEEE Transactions on*, **19**(8), 774–787.

Frankl, P. and Weyuker, E. (1993a). A formal analysis of the fault-detecting ability of testing methods. *Software Engineering, IEEE Transactions on*, **19**(3), 202–213.

Frankl, P. and Weyuker, E. (1993b). Provable improvements on branch testing. *Software Engineering, IEEE Transactions on*, **19**(10), 962–975.

Frankl, P. G. and Weiss, S. N. (1991). An experimental comparison of the effectiveness of the all-uses and all-edges adequacy criteria. In *TAV4: Proceedings of the symposium on Testing, analysis, and verification*, pages 154–164, New York, NY. ACM.

Gibson, P. and Hoang, K. (1994). Automatic statistical process control of a cnc turning centre using tool offsets and tool change. *International Journal of Advanced Manufacturing Technology*, **9**(3), 147–155.

Gitlow, H. S. (1990). *Planning for Quality, Productivity, and Competitive Position*. Dow Jones-Irwin, Homewood, IL.

Hailpern, B. and Santhanam, P. (2002). Software debugging, testing, and verification. *IBM Systems Journal*, **41**(1), 4–12.

Hamrol, A. (2000). Process diagnostics as a means of improving the efficiency of quality control. *Production Planning and Control: The Management of Operations*, **11**(8), 797–805.

Harrold, M. J., Rothermel, G., Wu, R., and Yi, L. (1998). An empirical investigation of program spectra. In *PASTE '98: Proceedings of the 1998 ACM SIGPLAN-SIGSOFT workshop on Program analysis for software tools and engineering*, pages 83–90, New York, NY. ACM.

Helstrom, C. W. (1984). *Probability and Stochastic Processes for Engineers*. Macmillan Publishing Company, New York, NY.

Hutchins, M., Foster, H., Goradia, T., and Ostrand, T. (1994). Experiments of the effectiveness of dataflow- and controlflow-based test adequacy criteria. In *ICSE '94: Proceedings of the 16th international conference on Software engineering*, pages 191–200, Los Alamitos, CA. IEEE Computer Society Press.

Ishikawa, K. (1976). *Guide to quality control*. Asian Productivity Organization, Tokyo.

ISO (2003). *ISO/TS 16949:2002*. Automotive Industry Action Group (AIAG), and International Standards Organization (ISO), Cincinnati, OH.

Jolliffe, I. (2002). *Principal Component Analysis*. Springer-Verlag New York, Inc., New York, NY, 2nd edition.

Juran, J. M. (1974). *Quality control handbook*. McGraw Hill, New York, NY, 3rd edition.

Kartha, C. (2004). A comparison of iso 9000:2000 quality system standards, qs9000, iso/ts 16949 and baldrige criteria. *The TQM Magazine*, **16**, 331 – 340.

Knuth, D. E. (1976). Big omicron and big omega and big theta. *SIGACT News*, **8**, 18–24.

Kourti, T. (2005). Application of latent variable methods to process control and multivariate statistical process control in industry. *International Journal of Adaptive Control and Signal Processing*, **19**(4), 213–246.

Liblit, B., Aiken, A., Zheng, A. X., and Jordan, M. I. (2003). Bug isolation via remote program sampling. *SIGPLAN Notices*, **38**(5), 141–154.

Liblit, B., Naik, M., Zheng, A. X., Aiken, A., and Jordan, M. I. (2005). Scalable statistical bug isolation. In *PLDI '05: Proceedings of the 2005 ACM SIGPLAN conference on Programming language design and implementation*, pages 15–26, New York, NY. ACM.

Lightstone, M., Dillingham, D. R., and Carpenter, D. D. (2005). The evolution of quality in the automotive industry. *Automotive Excellence*, pages 6 – 12.

Liu, Z.-Q. (1999). Repetitive measurement and compensation to improve workpiece machining accuracy. *The International Journal of Advanced Manufacturing Technology*, **15**, 85–89. 10.1007/s001700050043.

Mears, L., Roth, J., Djurdjanovic, D., Yang, X., and Kurfess, T. (2009). Quality inspections of machining operations: Cmm integration to machine tool. *Journal of Manufacturing Science and Engineering*, **131**(5), 1–12.

Montgomery, D. C. (2008). *Introduction to Statistical Quality Control*. John Wiley and Sons, Inc., New York, NY, 6th edition.

Ordys, A., Uduehi, D., and Johnson, M., editors (2007). *Process Control and Performance Assessment*. Springer-Verlag London Ltd., London, England, 4th edition.

Papoulis, A. (1965). *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill Book Company, New York, NY.

Park, S., Hartley, J. L., and Wilson, D. (2001). Quality management practices and their relationship to buyer's supplier ratings: a study in the korean automotive industry. *Journal of Operations Management*, **19**(6), 695 – 712.

Pavlopoulou, C. and Young, M. (1999). Residual test coverage monitoring. In *Software Engineering, 1999. Proceedings of the 1999 International Conference on*, pages 277–284. IEEE Computer Society Press.

Power, J. and Associates (2010). J.d. power and associates reports: Domestic brands surpass imports in initial quality for the first time in iqs history. *J.D. Power and Associates Automotive Reports - Press Release*.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (1992). *Numerical recipes in C: the art of scientific computing*. Cambridge University Press, Cambridge, MA, 2nd edition.

Ryan, T. P. (2000). *Statistical Methods for Quality Improvement*. John Wiley and Sons, Inc., New York, NY.

Shawky, A., Rosenburger, T., and Elbestawi, M. (1998). In-process monitoring and control of thickness error in machining hollow shafts. *Precision Machining*, **8**, 301–322.

Sheil, J. and O'Muircheartaigh, I. (1977). The distribution of non-negative quadratic forms in normal variables. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, **26**(1), 92–98.

Shewhart, W. A. (1931). *Economic control of quality of manufactured product*. D. Van Nostrand Company, New York, NY.

Shewhart, W. A. (1939). *Statistical method from the viewpoint of quality control*. The Graduate School, the Department of Agriculture, Washington.

Sinha, N. (1988). *Control Systems*. Holt, Rinehart and Winston, Inc., New York, NY.

Sitkin, S. B., Sutcliffe, K. M., and Schroeder, R. G. (1994). Distinguishing control from learning in total quality management: A contingency perspective. *The Academy of Management Review*, **19**(3), pp. 537–564.

Soderquist, K. and Motwani, J. (1999). Quality issues in lean production implementation: A case study of a french automotive supplier. *Total Quality Management*, **10**(8), 1107–1122.

Stewart, J. (2003). *Calculus: Early Transcendentals*. Brooks/Cole - Thomson Learning, Belmont, CA, 5e edition.

Tennant, G. (2001). *Six Sigma: SPC and TQM in Manufacturing and Services*. Gower Publishing Ltd., Burlington, VT.

van Wieringena, W. N. and de Mastb, J. (2008). Measurement system analysis for binary data. *Technometrics*, **50**(4), 468–478.

Vardeman, S. B. and Jobe, J. M. (1999). *Statistical Quality Assurance Methods for Engineers*. John Wiley and Sons, Inc., New York, NY.

Wall, M. (1996). Modelling of inspection reliability. In *Inspection Reliability: State-of-the-Art (Digest No. 1996/178), IEE Colloquium on*, pages 7/1–7/20.

Wall, M. (1997). Modelling of inspection reliability. *Engineering Science and Education Journal*, **6**(2), 63–72.

Wheeler, D. J. (2006). An honest gauge r&r study. In *2006 ASQ/ASA Fall Technical Conference*, pages 1–19. Reprint from SPC Press.

Xie, W., Xie, M., and Goh, T. N. (1995). Control charts for processes subject to random shocks. *Quality and Reliability Engineering International*, **11**(5), 355–360.

Yang, K. and EI-Haik, B. (2008). *Design for Six Sigma: A Roadmap for Product Development*. McGraw Hill, New York, NY.

Zorriassatine, F. and Tannock, J. D. T. (1998). A review of neural networks for statistical process control. *Journal of Intelligent Manufacturing*, **9**, 209–224.