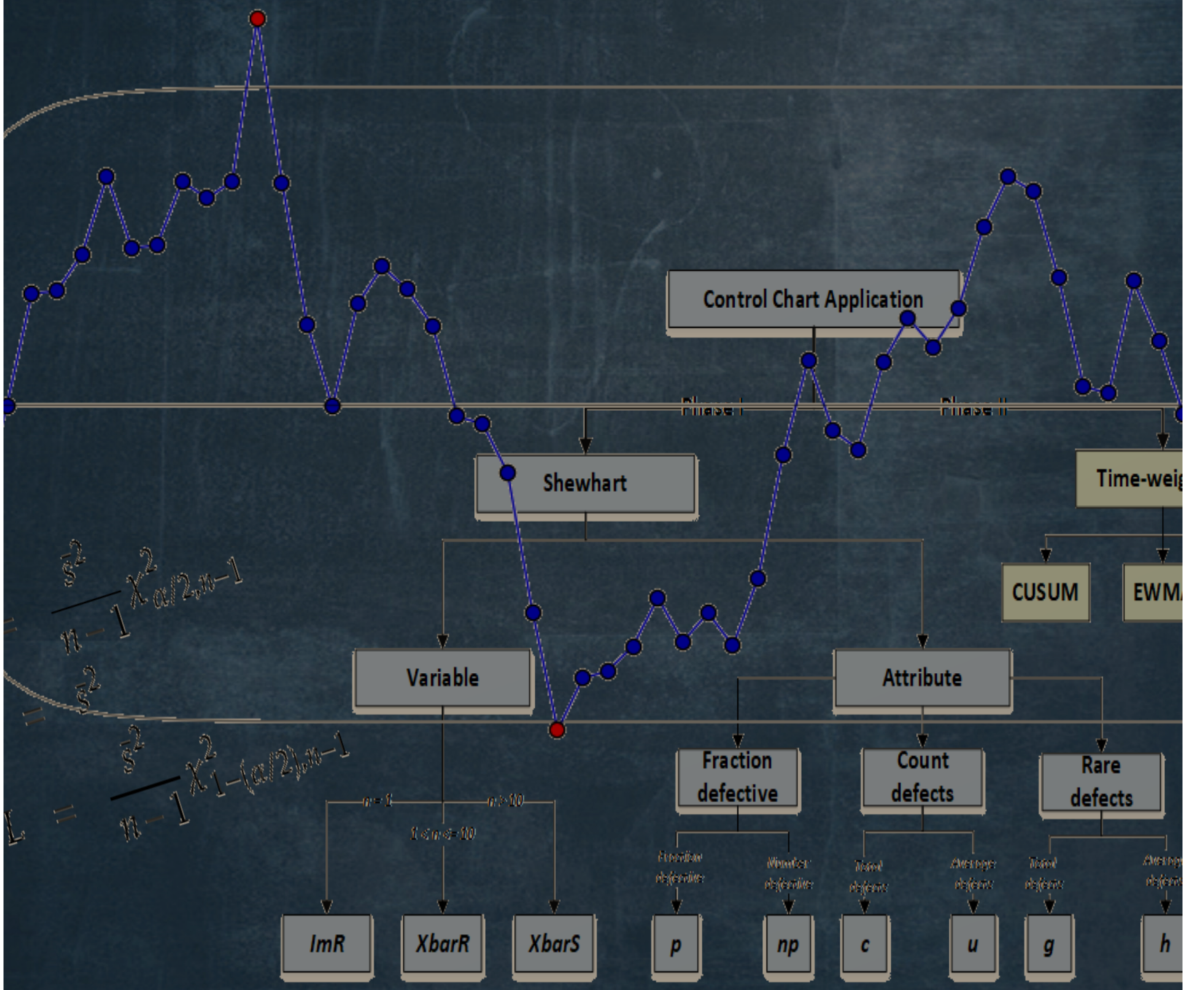
Applications of Control Charts for Quality Improvement in Health Care



Jerome Niyirora

ISBN **978-1-64176-080-5**
Edition 0.0

Cover design by Jerome Niyirora

About the Textbook

This textbook introduces students to the application methods of control charts to improve quality in health care. The textbook is written to be accessible to any student in the areas of health information management, health care informatics, and health care industrial engineering. Having a basic background in statistics would be beneficial, but such training is not a prerequisite to understanding how to apply the techniques discussed here. Several How-To sections are included to demonstrate the implementation of the given control charts using software such as Minitab and Excel. Additionally, samples of a Python code are included and can directly be accessed in a Jupyter Notebook at `https://github.com/JeromeNN`.

The textbook starts with **Chapter 1**, which contains introductory concepts of quality improvement using control charts. **Chapter 2** reviews the Shewhart model and the related charts. **Chapter 3** shows how to apply time-weighted control charts to detect small shifts in the process. **Chapter 4** appraises various techniques for adjusting control charts to account for factors such as autocorrelation and patient risk. **Chapter 5** considers other quality improvement techniques that relate to control charts. The Appendix presents factors for constructing variable control charts.

This textbook is not meant to be a comprehensive manuscript regarding quality improvement in health care. Instructors and students can supplement the chapter reading with additional resources, such as those referenced in the bibliography section of this text. The reader is also encouraged to consult a complementary textbook by the same author, titled **Basic Tools for Quality Improvement in Health Care Informatics**.

About the Author

Jerome Niyirora, Ph.D. is an Assistant Professor in the Health Information Management (HIM) Program, which is part of the College of Health Sciences at SUNY Polytechnic Institute (SUNY Poly). Jerome graduated from SUNYIT (now SUNY Poly) with a dual bachelor's degree in HIM and Health Services Management. Subsequently, he earned a Master of Science in Health Services Administration. Jerome also holds a Master of Science in Industrial Engineering from SUNY University at Buffalo, and a Ph.D. degree in Systems Science from SUNY Binghamton University. Jerome is a Registered Health Information Administrator (RHIA). His research focuses on the integration of system science theory, calculus of variations, and machine learning techniques into the improvement and optimal design of healthcare processes.

To
Missy
Jonathan and Aaron

Contents

CHAPTER 1

Basics of quality improvement

In this chapter, we introduce key concepts of quality improvement using control charts. Our discussion includes background information about control charts, the definition of the word "quality", and the frameworks of quality improvement. We also review basic notions of statistics that are required to apply control charts to improve quality in health care.

Key concepts and tools: *Quality; Quality improvement; Variability; Control charts; DMAIC[1] ; PDCA[2] ; Shewhart model; Hypothesis testing; Central limit theorem; Out-of-control behaviors; Special cause variation; Common cause variation; OCAP[3]*

Major objectives
After studying this chapter, you will be able to:

1. *Define critical concepts of quality improvement and control charts*
2. *Understand statistical measures of variability*
3. *Explain the basic notion of the central limit theorem*
4. *Compute and interpret standard sample statistics*
5. *Understand the construct of the Shewhart control chart model*
6. *Differentiate between special and common cause variation*
7. *Distinguish different types of control charts*
8. *Develop implementation strategies of control charts*
9. *Propose a quality improvement policy using control charts*
10. *Create an out-of-control action plan (OCAP)*

[1] **DMAIC**: Define, Measure, Analyze, Improve, and Control

[2] **PDCA**: Plan, Do, Check, and Act

[3] **OCAP**: Out-of-control action plan

1.1 Introduction

This book discusses applications of control charts to improve quality in health care. Control charts are essential tools of SPC[4] that we utilize to monitor and improve quality.

1.1.1 What is quality?

The word **quality** is a broad term that means different things depending on the context. For example, regarding a **product**, quality is often characterized using the following dimensions [29, 44]:

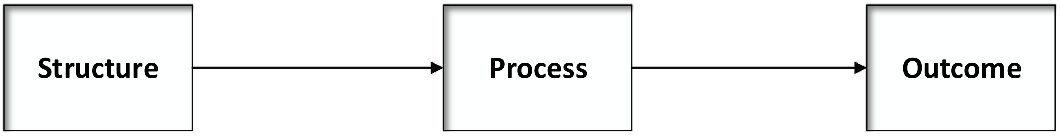
1. **Performance**: Will the product do the intended job?
2. **Reliability**: How often will the product fail?
3. **Durability**: How long will the product last?
4. **Serviceability**: How easily is the product fixed?
5. **Aesthetics**: What is the visual appeal of the product?
6. **Features**: What more can the product do?
7. **Perceived quality**: What is the reputation of the product or the company?
8. **Conformance to standards**: Is the product made as the designer intended?

A “quality product” is expected to be highly rated in all of the above dimensions. The dimensions of a “quality service” are not as well defined. But, the following features are often utilized to measure quality in general services [51]:

1. **Tangibles**: How is the physical appearance of the structure used to deliver a service such as equipment, building, personnel, and communication materials?
2. **Reliability**: Is the service as advertised, and is it accurate?
3. **Responsiveness**: Is the service responsive to the immediate needs of the customers?
4. **Assurance**: Is the service courteous, and do employees convey a sense of trust and confidence?
5. **Empathy**: Does the service exhibit care about the individual needs of the customer?

[4] **SPC:** Statistical Process Control

In health care, we can apply these general dimensions of service quality to evaluate administrative services such as patient registration, billing, bed management, and others. But, we cannot easily apply these dimensions to measure the quality of medical care, especially the **reliability** dimension, since, in most cases, we (patients) lack the technical knowledge to determine the accuracy of the care that we receive. This phenomenon is commonly known as **information asymmetry**.[5] Even for experts, the task of measuring the quality of health care is a daunting one given several factors to consider. One of the commonly accepted methods for measuring health care quality is the Donabedian's **Structure-Process-Outcome** model depicted in Figure 1.1.



exa

Figure 1.1: Donabedian's Structure-Process-Outcome model for measuring health care quality

At the **structure** level of Donabedian's paradigm, we measure the efficacy of resources and administrative processes used to provide medical care. At the **process** level, we evaluate the degree to which the applied medical standards are state-of-the-art. At the **outcome** level, we measure the result of care, such as the patient's health status and satisfaction [23]. Specific quality measures that relate to different levels of the Donabedian's model can be found at the websites of the Joint Commission [18] and the Centers for Medicare and Medicaid Services (CMS) [17].

1.1.2 Quality and variability

We can conceptualize quality as the **state of meeting the desired standard**, within some acceptable tolerance. Accordingly, we apply the following definition to most cases of quality improvement that we consider in this book:

Definition 1.1 (Quality)
Quality inversely relates to **variability** around the standard [44].

One implication of Definition 1.1 is that too much variability is associated with **low quality**, whereas little variability relates to **high quality**. Another implication is that before we can improve quality, we must know or be able to estimate the desired standard. In other words, we need to understand what the **customer** expects or is willing to pay for. In medical care, we must stay informed about the latest evidence-based standards of delivering care. Additionally, we must be able to measure variability around these standards. Then, to improve quality Definition 1.2:

[5] **Information asymmetry in health care** is the phenomenon where patients know less about the medical care they buy than health care providers who provide it [28, 6, 7, 53].

Definition 1.2 (Quality improvement)
Quality improvement is the **reduction of variability** around the desired standard [44].

1.1.3 Variability in the process

Several methods exist to help assess variability in the process, including various statistical measures of variance and graphical tools. Among the most common graphical techniques are control charts and histograms. We will return to statistical measures and control charts later. Here, we briefly review how histograms work.

Constructing a histogram

A histogram is a type of bar chart that we use to asses variability in processes that generate continuous data. To construct this chart, we first group data into **bins**, also called **intervals** or **classes**. We must choose an appropriate binning strategy for a histogram to be informative. Given the lack of a universal agreement about how to structure the bins, one common rule of thumb to select the number of bins k such that:

$$k = \left\lceil \sqrt{n} \right\rceil \tag{1.1}$$

where n is the number of observations in the data and the ceiling symbols $\lceil . \rceil$ signify *rounding up*. Another formula that we can use follows from Sturges' rule and looks like this [44]:

$$k = 1 + \log_2 n \tag{1.2}$$

where $\log$ is a mathematical symbol for logarithm. After deciding on the number of bins, we calculate the width of each bin h like this:

$$h = \frac{\max(x) - \min(x)}{k} \tag{1.3}$$

where x represents the sample data. Subsequently, we determine the range of the bins as follows:

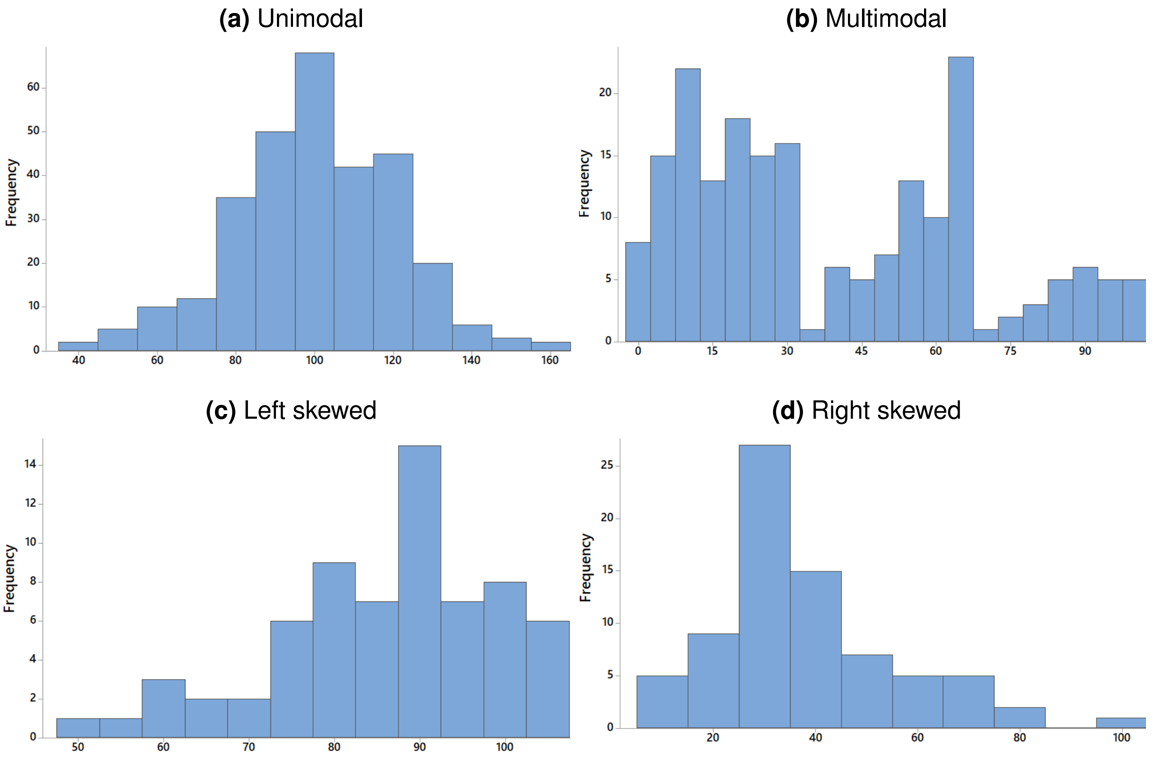
$$bin_1 = [\min(x),\ \min(x) + h + 1) \tag{1.4}$$
$$bin_2 = [\min(x) + h + 1,\ min(x) + 2h + 1) \tag{1.5}$$
$$...$$
$$bin_k = [\min(x) + (k-1)h + 1,\ \min(x) + kh + 1) \tag{1.6}$$

Finally, we throw each observation into the bin with a suitable range, and construct a histogram by graphing a bar chart in ascending order of the bin index (e.g., $bin_1, \ldots, bin_k$).

Interpreting a histogram

By visually examining the shape of a histogram, we can make inferences about the process variability regarding the skewness, kurtosis, modality, and the likely probability distribution. We can also learn about other characteristics of the process, such as the central tendency (e.g., mean or median) and outliers. We show examples of typical histograms in Figure 1.2.

Figure 1.2: Examples of histograms


(a) Unimodal

(b) Multimodal

(c) Left skewed

(d) Right skewed

Figure 1.2a shows a bell-shaped unimodal (one peak) histogram suggesting a normal distribution with the mean around 100. Figure 1.2b shows a multimodal histogram that suggests a mixture of probability distributions. Figure 1.2c shows a histogram with a left-skewed distribution. Figure 1.2d displays a histogram with a right-skewed distribution and a possible outlier in bin 100. Histograms with fewer outliers will generally have a low kurtosis measure. The latter quantity describes the degree of the *peakedness* and *flatness* (e.g., heavy-tail or light-tail) of the given distribution as compared to the normal distribution [22].

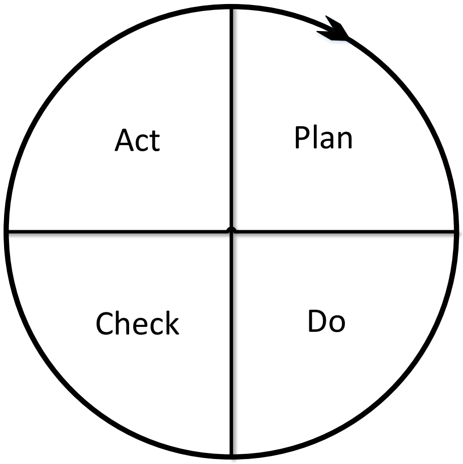
1.1.4 Frameworks of quality improvement

The most commonly used models of quality improvement are the **Plan, Do, Check, Act (PDCA)** cycle, and the **Define, Measure, Analyze, Improve, Control (DMAIC)** framework.

1.1.5 PDCA

Figure 1.3 portrays a graphical representation of the PDCA cycle, also called the **Shewhart cycle** or **PDSA** (where S stands for Study).

Figure 1.3: The PDCA cycle

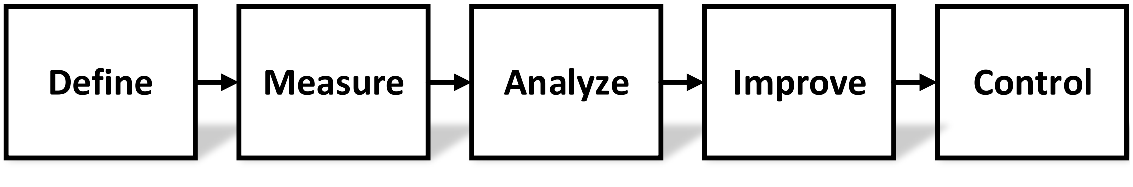


The arrow on the circle signifies the concept of “continuous” quality improvement. In the **Plan** phase of this cycle, we propose changes or experiments to run. In the **Do** phase, we carry out our plan, usually on a small scale. In the **Check** phase, we analyze the results from the Do phase. In the **Act** phase, we fully implement changes or abandon them. The cycle is iterated as many times as necessary to improve quality [44].

1.1.6 DMAIC

Figure 1.4 shows the basic phases of the DMAIC framework.

Figure 1.4: Phases of the DMAIC framework



The definition of each one of these phases follows [9, 30, 44].

Define: In this phase, we define opportunities for improvement. Additionally, we create flow charts, a project charter,[6] and a SIPOC[7] diagram.

Measure: In this phase, we collect data on key process input variables (KPIV) and key process output variables (KPOV). We attempt to establish baseline measures for WIP,[8] VOC,[9] PCE,[10] cycle time, process completion rate, sigma level, and financial metrics. At the end of the project, we will refer back to these initial measures to assess how much improvement we have made.

Analyze: In this phase, we try to make sense of the data we collected in the Measure phase by analyzing root cause, correlation, risk, stability, capability, and hypothesis tests. One of the tools that we may apply here is the cause-and-effect diagram to evaluate the causes of the quality problem. Another essential tool that we may deploy is a preliminary control chart to assess the stability of the process.

Improve: This phase is probably the most difficult since we have to turn our analysis into actions to improve the process. Creative thinking is imperative. To generate ideas for improvement, we may have to hold brainstorming sessions, not only with current process stakeholders but also with experts and the crowd [42]. One particularly useful statistical tool that we can employ here is DOE[11] [44]. For problems related to efficiency and cycle times, we can employ **Lean** techniques to eliminate non-value-added activities from the process. We continue to apply these improvement tools until control charts signal that the process is stable.

Control: In this phase, we implement tools to monitor the process, such as control charts, run charts, and dashboards. To the extent possible, we should automate this phase. Otherwise, we must develop a sampling plan to audit and monitor the new process. Additionally, we must create an out-of-control plan (OCAP) to prescribe actions to be taken when out-of-control behaviors arise in the improved process.

1.1.7 Technical aspects of quality improvement

We have previously mentioned that we cannot improve quality if we don't know the standard. If the standard is not given, we need to know how to approximate it from the process samples. Additionally, we must be able to measure variability if we are to reduce it to improve quality. All these activities require technical skills that, without them, quality improvement efforts may fail. Indeed, it is believed that the movement of total quality improvement (TQM) in the 1980s was not overly successful, partly because of the lack

[6] **Project charter**: a short document with clear statements about the goal, the business case, the opportunity, the project scope, plan, and team for quality improvement [44]

[7] **SIPOC:** Suppliers, Inputs, Process, Outputs, and Customers

[8] **WIP:** Work-in-Process (e.g., work yet to be finished)

[9] **VOC:** Voice of the Customer *(e.g., customer feedback)*

[10] **PCE:** Process Cycle Efficiency

[11] **DOE**: Design of Experiments

of emphasis on statistical aspects of quality improvement [44]. The strategy that tends to emphasize technical skills more is **Lean Six-Sigma**. The Lean part of this strategy focuses on making processes efficient. The **Six-Sigma** part utilizes statistical techniques to reduce variability in the process to reach a performance state of no more than 3.4 defective parts per million (PPM) opportunities (DPMO) in the process [30]. **Statistical process control (SPC)** is another strategy that emphasizes the reduction of variation to achieve the stability and capability of the process. The seven primary tools of SPC, which are also encountered in the Lean Six-Sigma strategy, are [44]:

1. **Histograms or stem-and-leaf plots** to visually assess the distribution of the process given the central tendency and variance. These tools may also help evaluate the process capability to meet the given quality specifications.

2. **Checksheets** to help collect and tabulate the process data by frequency, time, location, and cause.

3. **Pareto charts** to identify the "vital few" factors causing the majority of quality problems. A common analytical method applied here is the 80/20 rule that implies that 20% of the factors cause 80% of the problems.

4. **Cause-and-effect diagrams** to help analyze the root causes of the given effect. The causes are often categorized by factors such as materials, machines, measurement, people, methods, and policy.

5. **Defect concentration diagrams** to model the spatial distribution of defects on a particular product.

6. **Scatter diagrams** to characterize the relationship between two variables in the process. We often seek to conclude positive correlation, negative correlation, or no correlation between the variables under study.

7. **Control charts** to monitor the process behaviors over time or by the sample number. Control charts help detect and reduce special cause variation in the process.

In this book, we focus on the applications of **control charts** to improve quality in health care. In the next section, we review the basic statistical concepts that we will need to put these charts into practice.

1.2 Statistical background

In this section, we introduce basic statistical concepts that are requisite for the implementation of control charts. Some of the topics we highlight include process sampling, common probability distributions, point estimation, confidence intervals, hypothesis testing, correlation analysis, and regression.

1.2.1 Process sampling

Sample

In most quality improvement projects, it is not practical to observe and record everything that is happening in the process at all times. Even if such efforts were practical, they would likely be prohibitively expensive and time-consuming. A simplified approach is to use portions of the process to **infer** the performance of the entire process.

> **Definition 1.3 (Sample)**
> A **sample**, also referred to as **subgroup**, is a smaller set of measurements that we take from a larger set of the **population** measurements.

Terminology

We use the word **population** to suggest the entire process and the term **statistic** to indicate a statistical measure of the process, such as the **mean** (or the average) and the **variance**. The words **infer** and **inference** imply that we are making conclusions about the population using sample statistics. We will frequently use the expression **sampling distribution** to refer to the probability distribution of any sample statistic, most often the sample mean. We use letter n to denote the **sample size** (e.g., number of features) and letter m to symbolize the total number of samples that we call **sample number** (e.g., number of training examples).

Sampling techniques

Several sampling techniques apply to quality improvement, such as **random sampling**, where each item in the process has an equal chance of being selected. In **systematic sampling**, we set conditions for including an item in the sample (e.g., selecting every item in a prescribed position). In **stratified sampling**, we create distinct strata (e.g., groups) and choose an item from each stratum randomly or systematically. In **cluster sampling**, we group the population into naturally occurring clusters (e.g., geographical regions) and then sample from each cluster randomly or systematically. Both the stratified and cluster sampling techniques tend to reduce bias by representing all relevant segments of the population [39].

Sampling musts

We must choose the proper **sample size** and the **frequency of sampling** to allow for more accurate inferential conclusions about our process. This requirement is particularly important for successful deployments of control charts to monitor the stability of the process. One rule of thumb is to take at least 25 samples before drawing any conclusions about the stability of a process [44]. In general, the larger the sample size, the more likely we are to detect unstable instances or out-of-control behaviors in the process. But, the practicality of large samples is not always feasible. Instead, we try to **take smaller samples**, but **more frequently** [44].

It is also imperative that when sampling, we ensure **rational subgroups** to promote the independence of the samples. By this, we imply that we should seek to maximize the difference between the samples while minimizing the difference within each sample. A natural approach to obtaining rational subgroups is sampling from **units produced during the same time interval** (e.g., same day). Another approach is **sampling only from units produced since the last subgroup** [44]. Without rational subgroups, erroneous conclusions about the stability of the process are likely, especially when samples were initially assumed to be independent.

1.2.2 Sample statistics

Typical sample statistics include the **mean** $\bar{x}$, the **variance** s^2, and the **standard deviation** $s = \sqrt{s^2}$. Given a sample of size n, say $(x_1, x_2, \ldots, x_n)$, we calculate the statistics $\bar{x}$, s^2, and s, as follows:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{1.7}$$

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2 \tag{1.8}$$

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{1.9}$$

We commonly refer to $\sum_{i=1}^{n}(x_i - \bar{x})^2$ as the **sum of squares (SS)** and $\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$ as the **mean square (MS)**. Here, $n-1$ signifies the **degrees of freedom** or simply the number of independent values in a statistic [38]. We use the sample mean to estimate the average behavior of the process. Both the variance and standard deviation are nonnegative measures of variability in the process. The bigger these quantities, the more variable the process and thus the inferior the quality. Unlike the variance which expresses squared variation measures, the standard deviation quantifies the process variability in the original units.

How-To 1.1 (Sample statistics in Excel 2013)

1. To calculate the sample mean, use the $AVERAGE()$ function.
2. To calculate the sample variance, use the $VAR.S()$ function.
3. To calculate the sample standard deviation, use the $STDEV.S()$ function.

You can also calculate sample statistics via the *Data Analysis add-in > Descriptive Statistics > select your input range > check the option of Statistics summary > OK*.

If the Data Analysis add-in is not loaded in your Excel spreadsheet, you can add it by clicking on *File >Options > Add-ins > Analysis ToolPak > Manage: Excel Add-in > Check the box of Analysis ToolPak >OK*.

How-To 1.2 (Sample statistics in Minitab 18)
To calculate the sample mean, variance, and standard deviation in Minitab 18, click on *Stat > Basic Statistics > Display Descriptive Statistic > load your data into Variables >OK*.

How-To 1.3 (Python 3.6)

Script 1.1: A script for running descriptive statistics in Python 3.6

```
#Python 3.6 has several modules that one could use to compute sample
    statistics.
#One of these modules is Pandas that allows for the calculation of
    descriptive statistics as follows

#import pandas
from pandas import *
#import your sample data from Excel. The column name in Excel is
    Defects. The Sheet name is Data.
data = read_excel(r'C:\..\How-To1.3.xlsx','Data')
data = data.Defects
#print the sample mean, variance, and standard deviation,
    respectively
print (data.mean(), data.var(), data.std())
```

1.2.3 Probability distributions

Several probability distributions apply for quality improvement in health care.

Definition 1.4 (Probability distribution)
A **probability distribution** is a mathematical function that relates each value of a random variable to a real number between 0 and 1 to express the relative chances of that value occurring.

In Box 1.1, we grouped the most common probability distributions into classes of **discrete** and **continuous**. We apply discrete distributions to model processes that generate random attribute data (e.g., number of data entry errors), whereas we use continuous distributions to model processes that generate random variable data (e.g., duration of system downtime).

Box 1.1 Common Probability Distributions

Type	Name	Probability Distribution	Mean $\mu =$	Variance $\sigma^2 =$
Discrete	*Bernoulli*	$p(x) = \begin{cases} p & x = 1 \\ 1-p & x = 0 \end{cases}$	p	$p(1-p)$
	Binomial	$p(x) = \binom{n}{x} p^x (1-p)^{n-x}$ $x = 0, 1, \ldots, n$	np	$np(1-p)$
	Geometric	$p(x) = (1-p)^{x-1} p$ $x = 1, 2, \ldots$	$\frac{1}{p}$	$\frac{1-p}{p^2}$
	Poisson	$p(x) = \frac{e^{-\lambda}\lambda^x}{x!}$ $x = 0, 1, \ldots; \lambda > 0$	λ	λ
Continuous	*Normal*	$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$ $-\infty < x < \infty$ $-\infty < \mu < \infty, \sigma > 0$	μ	σ^2
	Exponential	$f(x) = \lambda e^{-\lambda x}$ $x \geq 0; \lambda \geq 0$	$\frac{1}{\lambda}$	$\frac{1}{\lambda^2}$

It follows that for discrete distributions, $0 \leq p(x) \leq 1$ and $\sum_x p(x) = 1$. For continuous random variables, $0 \leq p(a \leq x \leq b) = \int_a^b f(x)dx \leq 1$ and $\int_{-\infty}^{\infty} f(x)dx = 1$.

1.2.4 Normal distribution

During quality improvement projects, we tend to assume that our samples were drawn from a **normally distributed** process. This common assumption is the result of the central limit theorem (CLT).

CLT

Definition 1.5 (CLT and the sample mean)
Regardless of the population probability distribution, the sampling distribution of the sample mean $\bar{x}$ with size n, drawn from a population with mean μ and variance σ^2,

is about normally distributed and can be modeled as follows [44].

$$\bar{x} \sim Normal\left(\mu, \frac{\sigma^2}{n}\right) \tag{1.10}$$

It follows that as $n \to \infty$,

$$Z = \frac{\bar{x} - \mu}{\sigma/\sqrt{n}} \sim Normal(0,1) \tag{1.11}$$

We commonly refer to the Z statistic in Definition 1.5 as the **Z-score**. The numerator of this statistic, $\bar{x} - \mu$, is known as the **sampling error** whereas its denominator, $\sigma/\sqrt{n}$, is referred to as the **standard error** [38]. By mapping Z into a probability space, as $\Phi(Z)$, we obtain the probability p of a given standard value being less than Z. To find the probability of a standard value being greater than Z, we calculate $1 - \Phi(Z)$. It follows that $Z = \Phi^{-1}(p)$.

How-To 1.4 **($\Phi(Z)$ in Minitab 18)**
Store your Z-score value in a given column then click on *Calc* > *Probability Distributions* > *Normal* > *Input column* >*OK*.

How-To 1.5 **($\Phi(Z)$ in Excel 2013)**
To calculate $\Phi(Z)$ in Excel 2013, use this cumulative distribution function: $= NORM.S.DIST(Z,1)$.

How-To 1.6 **(Python 3.6)**

Script 1.2: A script for computing the inverse of Z-score in Python 3.6

```
#import norm from Scipy
from scipy.stats import norm
Z = x#some given constant x
#print the standard inverse of Z,  which is a probability.
print norm.cdf(Z)
```

Sampling distributions related to Normal

Typical sampling distributions that relate to Normal are **Chi-square** (χ^2), **t-distribution**, and **F-distribution**. We characterize these distributions as follows:

1. Given independent and normally distributed sample data points $x_1, x_2, \ldots, x_n$, where σ is known, $y = x_1^2 + x_2^2 + \cdots + x_n^2$ is distributed according to χ^2 with $n-1$ degrees of

freedom (χ^2_{n-1}). We can express y as follows [44]:

$$\begin{aligned} y &= \frac{1}{\sigma^2}\sum_{i=1}^{n}(x_i - \bar{x})^2 && (1.12) \\ &\equiv \frac{(n-1)s^2}{\sigma^2} \sim \chi^2_{n-1} && (1.13) \end{aligned}$$

2. If σ is unknown, if follows that:

$$\frac{\bar{x}-\mu}{s/\sqrt{n}} \sim N(0,1)\Bigg/\sqrt{\frac{\chi^2_{n-1}}{n-1}} \qquad (1.14)$$

   We refer to Equation 1.14 as the ***t-distribution***, or **Student's t-distribution**, with $n-1$ degrees of freedom [44]. We typically use this distribution to compare the means between two small samples, where we don't know the population standard deviation.

3. The ratio between two independent Chi-square random variables w and y with respective degrees of freedom u and v, is distributed according to ***F-distribution***, as follows [44]:

$$F_{u,v} = \frac{w/u}{y/v} \qquad (1.15)$$

   We often use the F-distribution in the analysis of variance (ANOVA) to test the equality of the means from independent samples drawn from normally distributed populations with equal variances.

1.2.5 Probability plots

A probability plot is a visual statistical tool that we can employ to validate our assumptions regarding the probability distribution of our process. We can approximate a probability plot by fitting a line to data (e,g., data that fall between the 25th and the 75th percentiles) [44]. If the hypothesized probability distribution were reasonable, points would fall along the fitted line. We will often use this technique to verify the normality assumption.

How-To 1.7 (Python 3.6)

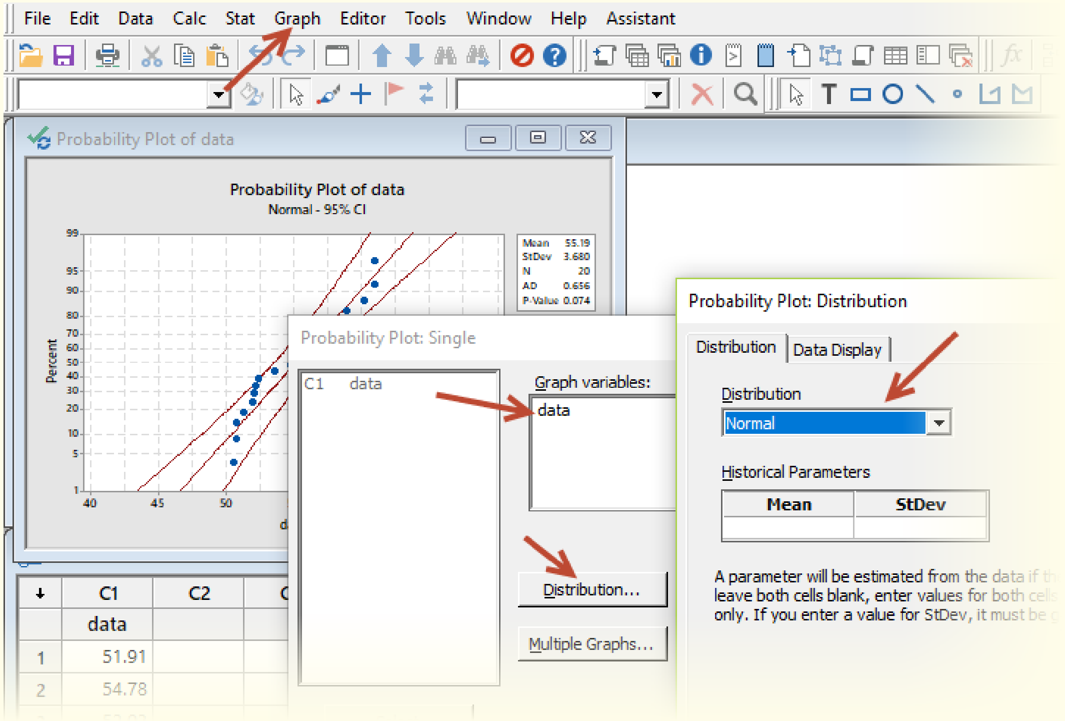
Script 1.3: A script for graphing a normal probability plot in Python 3.6

```
#import stats from scipy
from scipy import stats
import matplotlib.pyplot as plt #a plotting module
import seaborn as sns # an optional module to prettify the plot
fig = plt.figure()
ax = fig.add_subplot(111)
```

```
x = [...] #x is an array of your data
stats.probplot(x, plot=ax)
sns.despine(offset=10, trim=False)
plt.show()
```

How-To 1.8 (Normal probability plot in Minitab 18)
Click on *Graph* > *Probability Plot* > *Single* >*Select your data* > *OK*. To change the type of distribution, click on the *Distribution* tab. See the snapshot in Figure 1.5.

Figure 1.5: Options for a normal probability plot in Minitab 18



Probability Plot of data
Normal - 95% CI

Mean 55.19
StDev 3.680
N 20
AD 0.656
P-Value 0.074

↓	C1	C2
	data	
1	51.91	
2	54.78	

C1 data

Graph variables: data

Distribution: Normal

Mean	StDev

A parameter will be estimated from the data if th
leave both cells blank, enter values for both cells
only. If you enter a value for StDev, it must be g

1.2.6 Point estimators

In this subsection, we review the point estimation methods for the population mean and standard deviation parameters.

Definition 1.6 A **point estimator** is a statistic that we calculate from the process samples to estimate the population parameter [44].

The sample mean $\bar{x}$ and variance s^2 are **unbiased point estimators** of the corresponding population mean μ and variance σ^2 since

$$E(\bar{x}) = \mu \tag{1.16}$$

$$E(s^2) = \sigma^2 \tag{1.17}$$

Here, E symbolizes expectation. But, the sample standard deviation is a **biased point estimator** of the population standard deviation σ, since

$$E(s) \neq \sigma \tag{1.18}$$

Continuous process

For continuous processes with the sample size $n > 10$, we can approximate the unbiased estimator of σ this way:

$$\sigma \approx \frac{\bar{s}}{c_4} \tag{1.19}$$

where $\bar{s}$ is the mean of the sample standard deviations and c_4 is a constant that varies with n, as indicated in Appendix Table 12. When the sample size $n \leq 10$, we approximate the unbiased estimator of σ as follows:

$$\sigma \approx \frac{\bar{R}}{d_2} \tag{1.20}$$

where $\bar{R}$ is the mean of sample ranges and d_2 is a constant that also varies with n, as indicated in Appendix Table 12 [44].

Discrete processes

For processes that generate attribute data, we can approximate the mean μ like this:

$$\mu \approx \bar{p} \tag{1.21}$$

where $\bar{p}$ is the mean of the corresponding Bernoulli distribution. We apply the following approximation for the standard deviation σ:

$$\sigma \approx \hat{\sigma} = \sqrt{\frac{\bar{p}(1-\bar{p})}{n}} \tag{1.22}$$

where n denotes sample size as before.

1.2.7 Confidence interval

In the previous subsection, we discussed point estimators. In this subsection, we extend this concept into the related topic of the **confidence interval**.

> **Definition 1.7** A **confidence interval** is a range constructed from sample statistics to estimate a probabilistic interval that covers a given population parameter [38]

We create a confidence interval by setting lower and upper confidence limits of a parameter of interest. Here, we only review how to generate the confidence intervals of the population mean μ and variance σ^2 when the population is normally distributed.

Given a lower confidence limit A and an upper confidence limit B, we denote the probability of the population mean μ falling between A and B as follows:

$$P\{A \leq \mu \leq B\} = 1 - \alpha \tag{1.23}$$

where α is the significance level or the error that we are willing to commit. Using α, we calculate the percentage of our **confidence level** this way: 100(1-α)%. Equation 1.23 reflects a **two-sided** confidence interval. A **one-sided** confidence interval looks like this:

$$P\{A \leq \mu\} = 1 - \alpha \text{ or } P\{\mu \leq B\} = 1 - \alpha \tag{1.24}$$

The confidence interval of the population mean μ

Assuming normally distributed population, we construct a confidence interval of μ by considering the following two cases:

Case 1: a confidence interval of μ when σ is known

From the notion of CLT in Definition 1.5, we use the standardized value Z to construct the confidence interval of μ this way:

$$\mu = \bar{x} \pm Z_{\alpha/2} \frac{\sigma}{\sqrt{n}} \tag{1.25}$$

where $\alpha/2$ signifies a two-sided confidence interval. In a one-sided confidence interval, we replace $\alpha/2$ with α. We typically construct this confidence interval when the sample size n is large enough to allow for good point estimation of the population standard deviation (e.g., $n \geq 30$ [38]).

Case 2: a confidence interval of μ when σ is unknown

In this case, we use the t-distribution to determine the confidence interval of μ. Using Equation 1.14, we proceed as follows:

$$\mu = \bar{x} \pm t_{\alpha/2,n-1} \frac{s}{\sqrt{n}} \tag{1.26}$$

where s is the sample standard deviation, and $n-1$ represents the degrees of freedom. In a one-sided confidence interval, we replace $\alpha/2$ with α. We typically employ this confidence interval technique for smaller sampler sizes (e.g., $n < 30$ [38])

The confidence interval of the population variance σ^2

When the population is normally distributed, and we know neither the mean μ nor the variance σ^2, we use sample statistics and the Chi-square distribution χ^2 to create a two-sided confidence interval for σ^2, as follows [44]:

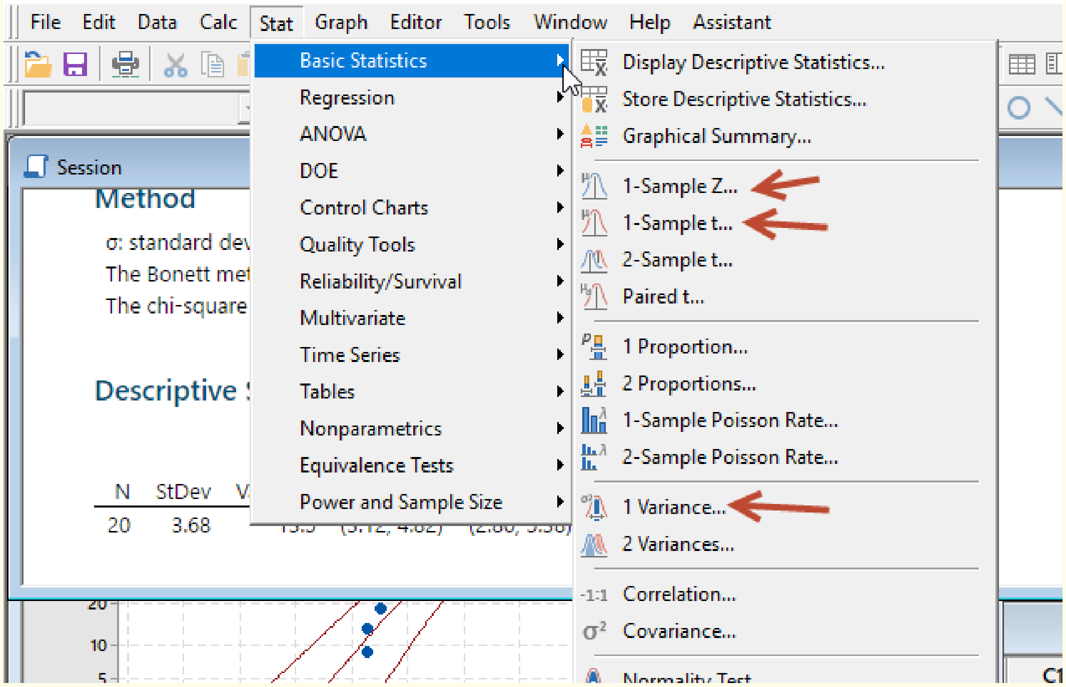
$$\frac{(n-1)s^2}{\chi^2_{\alpha/2,n-1}} \leq \sigma^2 \leq \frac{(n-1)s^2}{\chi^2_{1-\alpha/2,n-1}} \tag{1.27}$$

Here, s^2 is the sample variance, and $n-1$ represents the degrees of freedom. We determine a one-sided confidence interval this way:

$$\sigma^2 \leq \frac{(n-1)s^2}{\chi^2_{1-\alpha,n-1}} \tag{1.28}$$

How-To 1.9 (Confidence intervals of μ and σ^2 in Minitab 18)
Click on *Stats > Basic Statistics*. For the confidence interval of μ when σ is known, choose *1-Sample Z*. For the confidence interval of μ when σ is unknown, choose *1-Sample t*. For the confidence interval of σ^2, choose *1 Variance*. **Note:** Minitab may give you the confidence interval of the standard deviation, instead of that of the variance. Simply square the given limits to obtain the confidence interval of the variance. See the snapshot in Figure 1.6.

Figure 1.6: Options for the confidence intervals of μ and σ^2 in Minitab 18



How-To 1.10 (Confidence intervals of μ and σ^2 in Excel 2013)
There is no special option in Excel to calculate confidence intervals, but we can use the built-in functions to estimate the interval of interest. For example, we estimate the confidence interval for the population mean like this:

1. When σ is known, we use the function $= -NORM.S.INV(\alpha/2)$ to determine $Z_{\alpha/2}$. For example, when $\alpha = 0.05$, we find $Z_{0.05/2} = Z_{0.025}$ by $= -NORM.S.INV(0.025) = 1.96$.

2. When σ is unknown, we use the function $= -T.INV(\alpha/2, n-1)$ to obtain the critical value of the t-distribution. For example, when $\alpha = 0.05$, and $n = 10$, we find the critical value by $= -T.INV(\alpha/2, n-1) = -T.INV(0.025, 9) = 2.262$.

We can use Excel to construct the confidence interval for the population variance σ^2 by first determining the pertaining values of the Chi-square distribution. For example, we determine $\chi^2_{1-\alpha/2,n-1}$ as $= CHISQ.INV.RT(1-\alpha/2, n-1)$. To obtain $\chi^2_{\alpha/2,n-1}$, we use this function: $= CHISQ.INV.RT(\alpha/2, n-1)$.

How-To 1.11 (Python 3.6)

Script 1.4: A script for calculating confidence intervals in Python 3.6

```
#modules to import
from scipy.stats import norm
from scipy.stats import t
from scipy.stats import chi2
from numpy import *
from pandas import *
#confidence interval of mu when sigma is known
data = read_excel(your directory) #import data from Excel
data = data.Defects# The column name in Excel is Defects.
xbar = data.mean()
sigma = ...#given
alpha = ...#given (e.g., alpha = 0.05)
Z = norm.ppf(1-alpha/2.)
N = len(data)#sample number
A = xbar -Z*sigma/sqrt(N)#lower confidence limit
B = xbar +Z*sigma/sqrt(N)#upper confidence limit
print (A, B)
```

1.2.8 Hypothesis testing

We conduct a hypothesis test to evaluate the likelihood of our belief about the value of the population parameter. During hypothesis testing, we make two statements. Our first statement that we call the **null hypothesis** or H_0 expresses our belief that the population parameter equals a particular value or falls within a given range. Our second statement that we call an **alternative hypothesis** or H_1, opposes the first statement. We **reject** H_0, when a derived **test statistic** is greater than a given **critical value** or when the ***p-value*** $< \alpha$. Otherwise, we **fail to reject** H_0. As before, α is the significance level, or simply the error that we are willing to commit.

Hypothesis test for the process mean

We start by assuming that $\mu = \mu_0$, where μ_0 is the conjectured value. We formulate a **two-tailed** or two-sided hypothesis test as follows:

$$H_0 : \mu = \mu_0 \tag{1.29}$$

$$H_1 : \mu \neq \mu_0 \tag{1.30}$$

Alternatively, we can create a **one-tailed** or one-sided test this way:

$$H_0 : \mu = \mu_0 \tag{1.31}$$

$$H_1 : \mu < \mu_0 \tag{1.32}$$

or as:

$$H_0 : \mu = \mu_0 \tag{1.33}$$

$$H_1 : \mu > \mu_0 \tag{1.34}$$

Next, we find and compare the **test statistic** Z_0 to the corresponding **critical value** Z_α. In a normally distributed process where σ is **known**, we refer to our hypothesis test as the **Z-test**. We obtain the test statistic $\boldsymbol{Z_0}$ as follows:

$$Z_0 = \frac{\bar{x} - \mu}{\sigma/\sqrt{n}} \tag{1.35}$$

The critical value is given by $Z_{\alpha/2}$ in a two-sided test or Z_α in a one-sided test. We reject H_0 when $|Z_0| > Z_{\alpha/2}$ in a two-sided test or when $|Z_0| > Z_\alpha$ in a one-sided test. We use $|.|$ symbols to denote the absolute value function. When the population standard deviation σ is **unknown**, we conduct a **t-test**, instead of a Z-test, and obtain the test statistic $\boldsymbol{t_0}$ this way:

$$t_0 = \frac{\bar{x} - \mu}{s/\sqrt{n}} \tag{1.36}$$

Given the degrees of freedom $n-1$, the critical value is obtained by $t_{\alpha/2,n-1}$ in a two-sided test or $t_{\alpha,n-1}$ in a one-sided test. In a two-sided test, we reject H_0 when $|t_0| > t_{\alpha/2,n-1}$. In a one-sided test, we reject H_0 when $|t_0| > t_{\alpha,n-1}$.

Hypothesis test for the process variance

We structure a hypothesis test for the process variance like this:

$$H_0 : \sigma^2 = \sigma_0^2 \tag{1.37}$$

$$H_1 : \sigma^2 \neq \sigma_0^2 \tag{1.38}$$

where σ_0^2 is the variance that we have conjectured. Alternatively, we could set $H_1 : \sigma^2 > \sigma_0^2$ or $H_1 : \sigma^2 < \sigma_0^2$. Assuming normal distribution and unknown population mean and variance, we use the Chi-square distribution to obtain the test statistic, $\boldsymbol{\chi_0^2}$, as follows:

$$\chi_0^2 = \frac{(n-1)s^2}{\sigma_0^2} \tag{1.39}$$

We recall that s^2 is the sample variance, and $n-1$ represents the degrees of freedom. The two-sided critical value is given by $\chi^2_{\alpha/2,n-1}$. We reject the null hypothesis when $\chi_0^2 > \chi^2_{\alpha/2,n-1}$ or $\chi_0^2 < \chi^2_{1-\alpha/2,n-1}$. The critical value in a one-sided test is $\chi^2_{\alpha,n-1}$. For $H_1 : \sigma^2 > \sigma_0^2$, we reject the null hypothesis when $\chi_0^2 > \chi^2_{\alpha,n-1}$. For $H_1 : \sigma^2 < \sigma_0^2$, we reject the null hypothesis when $\chi_0^2 < \chi^2_{1-\alpha,n-1}$ [44].

1.2.9 P-value

We have briefly introduced p-value before as a probability measure of a hypothesis test. We formally define this quantity next.

Definition 1.8 (P-value)
$\boldsymbol{p\text{-}value}$ is the probability that one would observe a test statistic that was greater or equal to the given critical value, if the null hypothesis were true [38]

We recall that the critical value is a statistic that we construct using the confidence level α. When $\boldsymbol{p\text{-}value < \alpha}$, we reject the null hypothesis. Most statistical packages, such as Minitab, provide us with p-values automatically when we run hypothesis tests. In a normal distribution, we calculate p-values as follows [44]:

1. $H_1 : \mu \neq \mu_0$, p-value = $2(1 - \Phi(|Z_0|))$
2. $H_1 : \mu > \mu_0$, p-value = $1 - \Phi(Z_0)$
3. $H_1 : \mu < \mu_0$, p-value = $\Phi(Z_0)$

Z_0 is the test statistic that we obtain per Equation 1.35.

1.2.10 The type of error

While testing our hypothesis, we are likely to make two types of errors:

Type I error (α): we reject H_0 when H_0 is true.

Type II error (β): we fail to reject H_0 when H_0 is false.

Operationally speaking, the α error is equivalent to erroneously rejecting the quality of a good product or service. In control chart applications, this type of error is also referred to as **false alarm** and signifies the chances of observing erroneous out-of-control signals. The β error is equivalent to mistakenly delivering a bad product or service to the customer [44]. We choose α to indicate the error probability that we are willing to tolerate. When our process is normally distributed, we typically set $\alpha = 0.05$ and determine β this way [44]:

$$\beta = \Phi\left(Z_{\alpha/2} - \frac{\delta\sqrt{n}}{\sigma}\right) - \Phi\left(-Z_{\alpha/2} - \frac{\delta\sqrt{n}}{\sigma}\right) \tag{1.40}$$

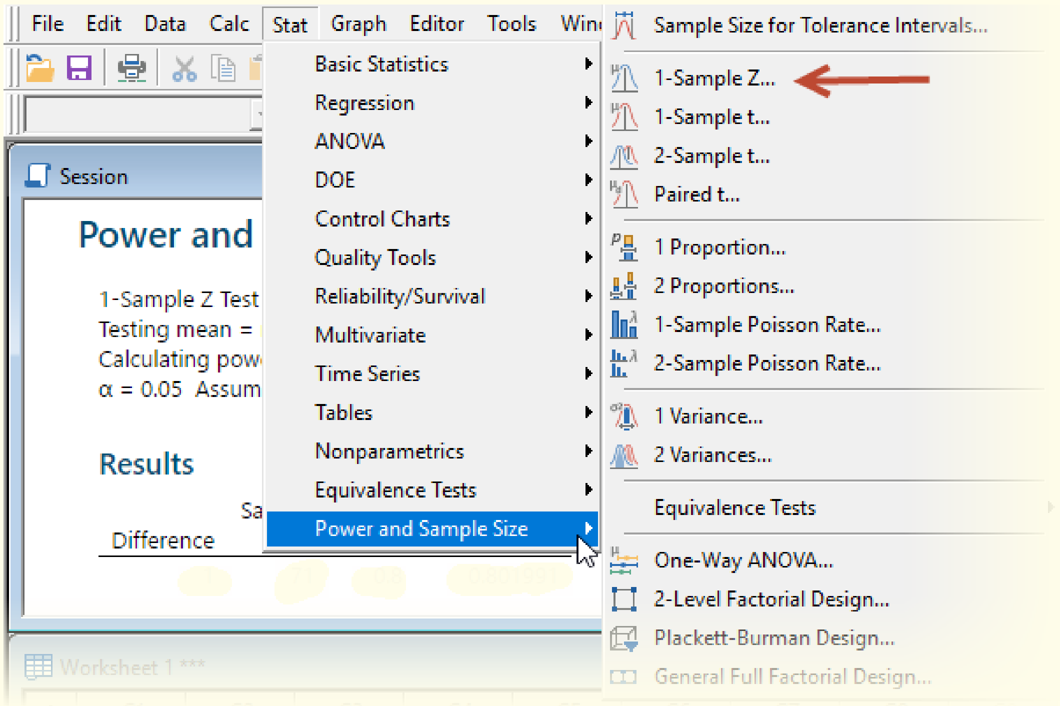
Here, δ is the difference between the true process parameter and the hypothesized parameter. For example in the context of the mean, $\delta = \mu_1 - \mu_0$, where μ_1 is the true mean and μ_0 is the hypothesized mean. From the β error, we obtain the **power of our statistical test** as like this:

$$Power = 1 - \beta \tag{1.41}$$

This power test expresses the probability of rejecting the null hypothesis correctly. To increase the power test, we increase the sample size n, which in turn decreases β per Equation 1.40. The **operating-characteristic (OC)** curves can help us choose the proper sample size to satisfy a particular value of β [44]. Minitab has an option for obtaining the appropriate sample size n when the desired power test is provided. Minitab instructions for the power of 1-sample Z test are shown in 1.12.

How-To 1.12 (Power for 1-sample Z test in Minitab 18)
Click on *Stat > Power and Sample Size > select the test of interest*. For example, to determine the sample size in 1-Sample Z Test, leave the sample size field blank, and specify the difference that you want to detect δ. You also need to specify the population standard deviation and the desired power value. See the snapshot in Figure 1.7.

Figure 1.7: Power and Sample Size for 1-Sample Z Test in Minitab 18



How-To 1.13 (Python 3.6)

Script 1.5: A script for the power of 1-sample t test in Python 3.6

```
#import tt_solve_power from statsmodels to run power test one sample
    t test
from statsmodels.stats.power import tt_solve_power
#https://www.statsmodels.org/stable/generated/statsmodels.stats.power
    .tt_ind_solve_power.html
delta = ...#absolute difference given by mu1 - mu0
sigma = ... #standard deviation
effect_size = delta/sigma
alpha = ...#given (e.g., 0.05)
#solve for the sample size (nobs), given the target power
power =...#given (e.g., 0.8)
sample_size = tt_solve_power(effect_size=effect_size, nobs= None,
    alpha=alpha, power=power, alternative='two-sided')
print (sample_size)

#solve for the power, given the sample size
sample_size = ...#given
sample_power = tt_solve_power(effect_size=effect_size, nobs=
    sample_size, alpha=alpha, power=None, alternative='two-sided')
print (sample_power)
```

1.2.11 Comparing the means of two independent samples

We can use hypothesis testing to check whether the means of two **independent** samples were drawn from the same population or two similarly distributed populations. To test whether the population means, μ_1 and μ_2, are equal, we obtain two samples of sizes n_1 and n_2 and means $\bar{x}_1$ and $\bar{x}_2$, respectively. We formulate the null hypothesis like this:

$$H_0 : \mu_1 = \mu_2 \tag{1.42}$$

We frame alternative hypotheses as $H_1 : \mu_1 \neq \mu_2$, $H_1 : \mu_1 > \mu_2$, or $H_1 : \mu_1 < \mu_2$. Assuming a normally distributed population with the known variance σ^2, we apply the Z-test and obtain the **test statistic** as follows:

$$Z_0 = \frac{\bar{x}_1 - \bar{x}_2}{\sigma\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}} \tag{1.43}$$

If the variance is unknown, we apply the t-test and obtain the **test statistic** like this:

$$t_0 = \frac{\bar{x}_1 - \bar{x}_2}{s_p\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}} \tag{1.44}$$

Here, s_p is the square root of the pooled variance s_p^2 that is computed this way:

$$s_p^2 = \frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2} \tag{1.45}$$

where $n_1 + n_2 - 2$ represents the degrees of freedom from two samples [38]. We obtain critical values and conclude about the null hypothesis as before.

1.2.12 Comparing the means of more than two independent samples

Under the assumptions of the normal distribution, independence, and equal variances, we can test the following hypothesis [38]:

$$H_0 : \mu_1 = \mu_2 = \cdots = \mu_m \tag{1.46}$$

$$H_1 : \text{the means are not all equal} \tag{1.47}$$

Here, m is the index of the samples. Using the Analysis of Variance (ANOVA) test, we obtain the following **test statistic**:

$$F_0 = \frac{MS_{treatments}}{MS_{error}} \tag{1.48}$$

where $treatments$ symbolizes independent samples, $MS_{treatments}$ is the mean square of treatments, and MS_{error} is the mean square of the error. Our **critical value** originates from the F-distribution and looks like this:

$$F_{\alpha,m-1,m(n-1)} \tag{1.49}$$

If $F_0 > F_{\alpha,m-1,m(n-1)}$, we reject the null hypothesis. We will return to the topic of ANOVA in Section 5.3.3.

1.2.13 Comparing the means of two dependent samples

When two samples are **dependent**, we conduct hypothesis testing using the **paired t-test.** Our **test statistic** looks like this:

$$t_0 = \frac{\bar{d}}{s_d/\sqrt{n}} \tag{1.50}$$

where $\bar{d}$, the mean of the difference between two samples (each with size n), is given by:

$$\bar{d} = \frac{1}{n}\sum_{j=1}^{n} d_j \tag{1.51}$$

From Equation 1.50, s_d is the standard deviation of the differences, which is obtained by taking the square root of the variance s_d^2, given by:

$$s_d^2 = \frac{1}{n-1}\sum_{j=1}^{n}(d_j - \bar{d})^2 \tag{1.52}$$

As before, the critical value is given by $t_{\alpha/2,n-1}$ in a two-sided test or $t_{\alpha,n-1}$ in a one-sided test. In a two-sided test, we reject H_0 when $|t_0| > t_{\alpha/2,n-1}$. In the one-sided test, we reject H_0 when $|t_0| > t_{\alpha,n-1}$.

1.2.14 Comparing the means of more than two dependent samples

Standard statistical software such as Minitab have options to compare the means of more than two dependent samples, such as when one repeats measures on single or double factors. See Minitab instructions in How-To 1.16 about how to analyze a design of a single-factor with repeated measures.

How-To 1.14 (Z-test and t-test in Excel 2013)
You can run a t-test in Excel by using this function: $= T.TEST(sample_1, sample_2, tails, type)$. In the *tails* option, you specify whether the test is one or two-tailed. In the *type* option, you specify whether you are running a paired t-test (1), a t-test with equal variances (2), or a t-test with unequal variances (3). You can also run a t-test via the **Data Analysis add-in**. For that, open the *Data Analysis* add-in, as shown in the snapshot portrayed in Figure 1.8 and select the appropriate test (see the snapshot in Figure 1.9).

Figure 1.8: Excel 2013 Data Analysis add-in option

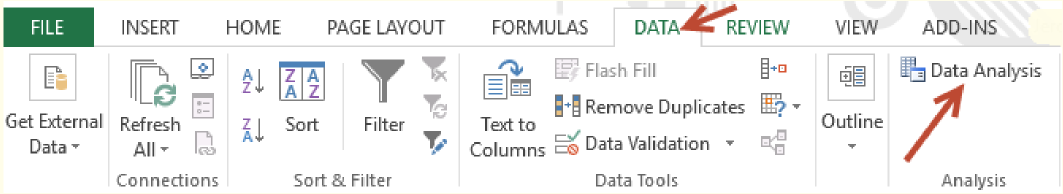
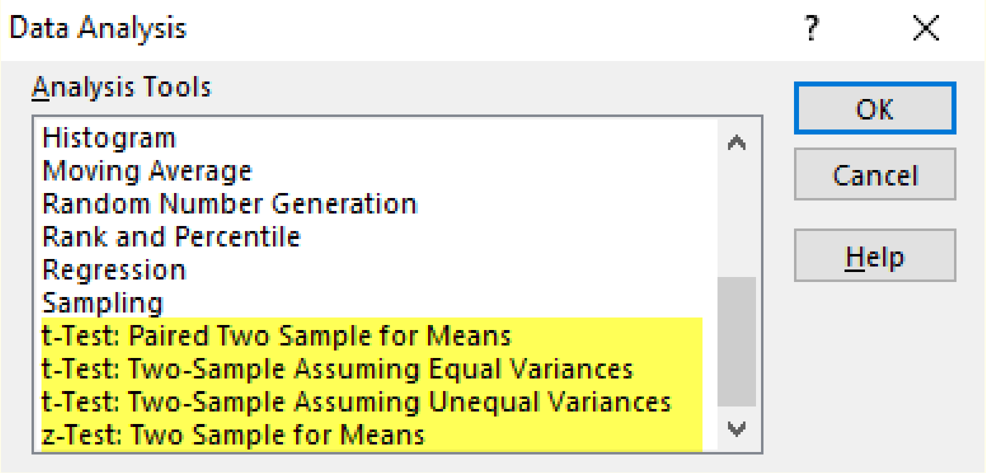


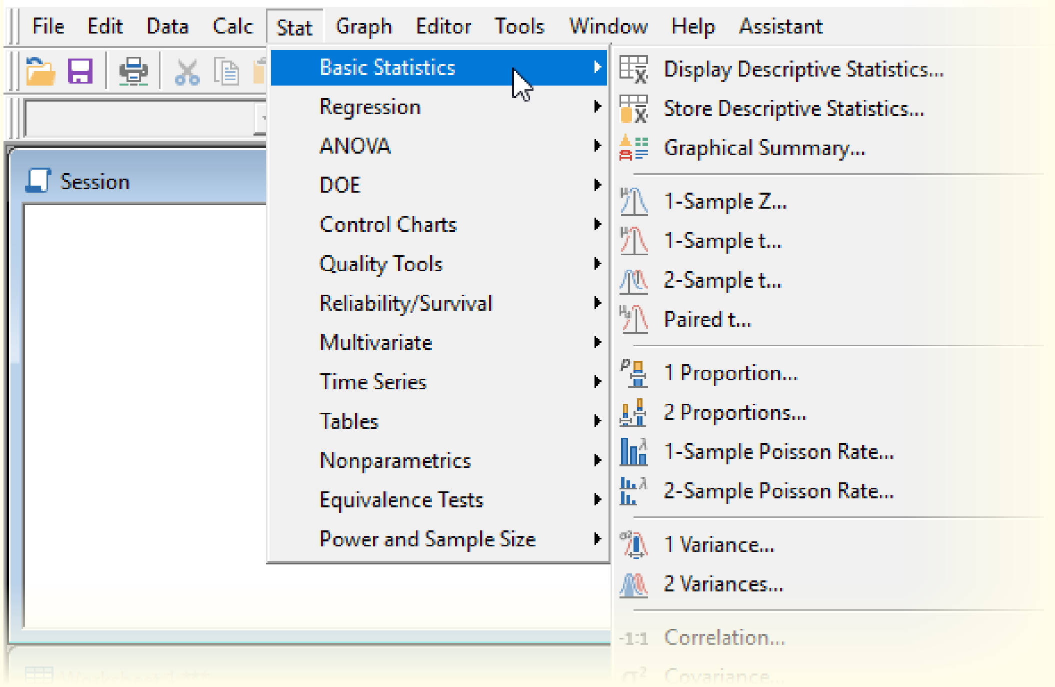
Figure 1.9: Z-test and t-test via Data Analysis add-in option in Excel 2013


Data Analysis
?
Analysis Tools
Histogram
Moving Average
Random Number Generation
Rank and Percentile
Regression
Sampling
t-Test: Paired Two Sample for Means
t-Test: Two-Sample Assuming Equal Variances
t-Test: Two-Sample Assuming Unequal Variances
z-Test: Two Sample for Means
OK
Cancel
Help

If the Data Analysis add-in is not loaded, you can add it by clicking on *File >Options > Add-ins > Analysis ToolPak > Manage: Excel Add-in > Check the box of Analysis ToolPak >OK*.

How-To 1.15 (Z-test and t-test in Minitab 18)
To conduct a Z-test or a t-test in Minitab 18, click on *Stat>Basic Statistics* and select the appropriate test. See the snapshot portrayed in Figure 1.10.

Figure 1.10: Z-test and t-test options in Minitab 18



How-To 1.16 (Single-factor with repeated measures in Minitab 18)
To conduct an analysis of a single-factor design with repeated measures in Minitab 18, click on *Stat* > *ANOVA* > *General Linear Model* > *Fit General Linear Model* > *Select Factors* > *Select Responses* > *Click on Random/Nest >Factor Type >Random >OK*.

How-To 1.17 (Python 3.6)

Script 1.6: A script for running Z-test and t-test in Python 3.6

```
#import modules
from scipy import stats
from statsmodels.stats import weightstats
#https://www.statsmodels.org/stable/generated/statsmodels.stats.
    weightstats.ztest.html
#https://www.statsmodels.org/stable/generated/statsmodels.stats.
    weightstats.ttest_ind.html
#t-test returns (t-statistic, two-tailed p-value)
x1 = [...] #array of the first dataset
x2 = [...] #array of the second dataset
```

```
equal_var = ...#True if equal variances, otherwise False
stats.ttest_ind(x1,x2, equal_var = equal_var)
#Z-test returns (Z-statistic, two-tailed p-value)
weightstats.ztest(x1, x2, alternative= 'two-sided', usevar='pooled',
    ddof=1.0)
```

1.2.15 Comparing variances of two independent samples

We hypothesize that two samples (sample 1 and sample 2), with variances s_1^2 and s_2^2, were drawn from two independent populations with equal variances, σ_1^2 and σ_2^2. We create a hypothesis test to validate our conjecture as follows:

$$H_0 : \sigma_1^2 = \sigma_2^2 \tag{1.53}$$

$$H_1 : \sigma_1^2 \neq \sigma_2^2 \tag{1.54}$$

Using F-distribution, we obtain the **test statistic** this way:

$$F_0 = \frac{s_1^2}{s_2^2} \tag{1.55}$$

The lower **critical value** is given by $F_{1-\alpha/2,n_1-1,n_2-1}$ whereas the upper critical value is given by $F_{\alpha/2,n_1-1,n_2-1}$. Here, $n_1 - 1$ and $n_2 - 1$ are degrees of freedom of sample 1 and sample 2, respectively. If $F_0 > F_{\alpha/2,n_1-1,n_2-1}$ or $F_0 < F_{1-\alpha/2,n_1-1,n_2-1}$ we reject the null hypothesis. To perform a one-sided test, we formulate H_1 and F_0 statistics this way:

$$H_1 : \sigma_1^2 > \sigma_2^2, \qquad F_0 = s_1^2/s_2^2 \tag{1.56}$$

$$H_1 : \sigma_1^2 < \sigma_2^2, \qquad F_0 = s_2^2/s_1^2 \tag{1.57}$$

For the case of $H_1 : \sigma_1^2 > \sigma_2^2$, we reject the null hypothesis when $F_0 > F_{\alpha,n_2-1,n_1-1}$. For the case of $H_1 : \sigma_1^2 < \sigma_2^2$, we reject the null hypothesis when $F_0 < F_{\alpha,n_1-1,n_2-1}$.

How-To 1.18 (F-test in Excel 2013)
To calculate the critical value of $F_{\alpha,\, n_1-1,\, n_2-1}$ in Excel 2013, use this function $= F.INV.RT(\alpha,\ n_1 - 1,\ n_2 - 1)$. You can estimate the p-value by $= F.DIST.RT(F_0,\ n_1 - 1,\ n_2 - 1)$. You can also conduct the F-test using the **Data Analysis add-in** under the Data tab. Choose the option of *F-test Two-Sample for Variances*.

How-To 1.19 (F-test in Minitab 18)
Click on *Stat > Basic Statistics > 2 Variances*. Review the snapshot in Figure 1.6. **Note** that Minitab runs the test on equal standard deviations, but this will not alter the conclusion about the null hypothesis.

How-To 1.20 (Python 3.6)

Script 1.7: A script for running F-test in Python 3.6

```
#import f from scipy.stats
from scipy.stats import f
alpha = ...#given  e.g, 0.05
var1 = ...#variance of first variable
var2 = ... #variance of second variable
df1 = ...#degrees of freedom first variable
df2 = ...#degrees of freedom second variable
F0 = var1 /var2 #F-statistic
pvalue = 1-f.cdf(F0, df1, df2) #if p-value < alpha, reject null
    hypothesis
print (pvalue)
```

1.2.16 Correlation analysis

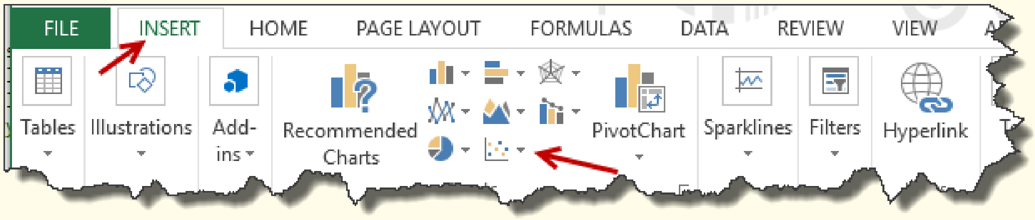
We conduct a correlation analysis to evaluate the relationship between two variables. One way to conduct such analysis is via scatter plots to examine how variables vary together visually. Another way is to apply using Pearson and Spearman correlation coefficients. Another similar technique that we do not cover in this section is the Kendall rank correlation coefficient [2].

Scatter diagrams

A scatter diagram is one of the major tools of statistical process control (SPC) that we use in the *Analyze* phase of DMAIC to evaluate the correlation between two variables. To create a scatter diagram, we start by organizing the data in the table format by storing each variable in a separate column. Each row constitutes an ordered pair of observations. To create a scatter plot, we simply graph a 2-dimensional chart of the variables in the Cartesian coordinates system.

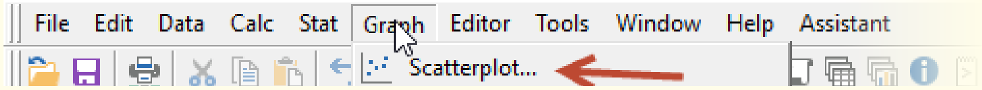
How-To 1.21 (Creating a scatter plot in Excel 2013)
To create a scatter diagram in Excel 2013, click on *INSERT* > *Scatter Charts*. See the snapshot in Figure 1.11.

Figure 1.11: Options for a scatter plot in Excel 2013



How-To 1.22 (Creating a scatter plot in Minitab 18)
To create a scatter diagram in Minitab 18, click on *Graph* >*Scatterplot* . See the snapshot in Figure 1.12).

Figure 1.12: Options for a scatter plot in Minitab 18



Interpreting a scatter diagram

We characterize the relationship in a scatter plot as follows:

1. **Positive correlation** (or direct relationship) when two variables increase together.

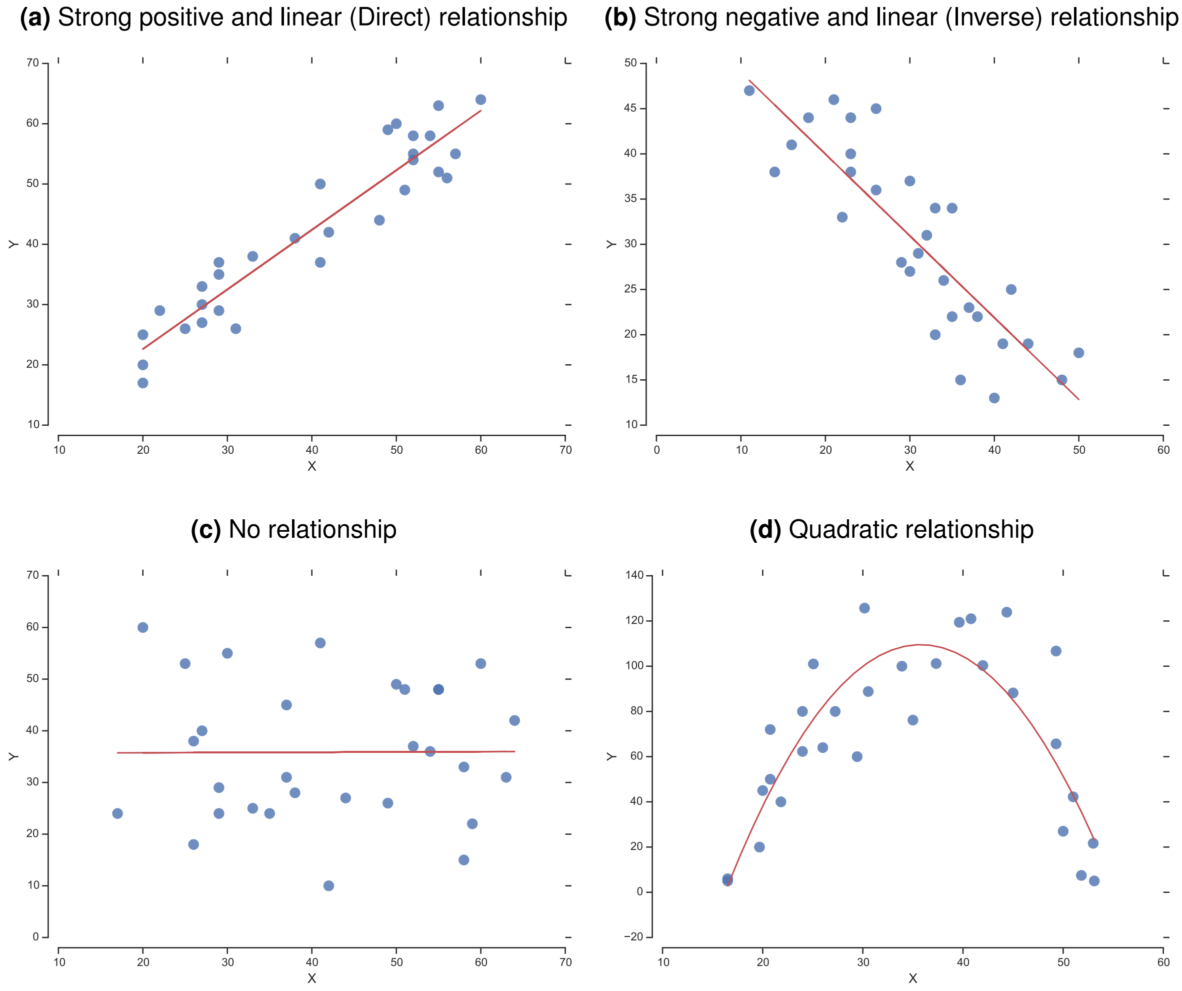
2. **Negative correlation** (or inverse relationship) when one variable increases as the other one decreases.

3. **No correlation** (or no relationship) when two variables do not vary together.

We can further differentiate the strength of the relationship by using terms such as **strong, moderate,** or **weak**. For example, we can describe two variables as being *strongly and positively correlated*. It is also common to fit a line on the scatter data to evaluate the **linearity** or **non-linearity** of the relationship. Figure 1.13 portrays examples of scatter plots. In Subfigure 1.13a, a strong positive and linear relationship exists between variables X and Y. This association implies that as X increases, Y tends to increase too. In Subfigure 1.13b, we observe a strong negative and linear relationship between X and Y, implying that as X increases, Y tends to decrease. In Subfigure 1.13c, no obvious relationship is discernible, meaning that X and Y are likely independent variables. Subfigure 1.13a shows a non-linear relationship between X and Y, which, after fitting a line, seems quadratic.

Remark: *Scatter diagrams do not suggest that one variable causes another, rather imply that a* ***potential relationship*** *exists. Other statistical tools, such as the design of experiments (DOE), are best suited for entertaining the* ***causality*** *issue [44].*

Figure 1.13: Examples of scatter diagrams



(a) Strong positive and linear (Direct) relationship

(b) Strong negative and linear (Inverse) relationship

(c) No relationship

(d) Quadratic relationship

Pearson correlation coefficient

The Pearson correlation coefficient is a parametric technique that we use to quantify the degree of the relationship between two continuous variables. We calculate this coefficient by taking the ratio between the corresponding covariance and standard deviations statistics. To demonstrate, let's assume that we have two samples X and Y, each with size n, and means of $\overline{X}$ and $\overline{Y}$, respectively. The covariance of X and Y, $cov(X,Y)$ is given by:

$$cov(X,Y) = \frac{\sum(X-\overline{X})(Y-\overline{Y})}{n-1} \tag{1.58}$$

where $n-1$ represents the degrees of freedom. We obtain the Pearson correlation coefficient $\rho(X,Y)$, sometimes denoted as r, this way:

$$\rho(X,Y) \equiv r = \frac{cov(X,Y)}{s_X s_Y} \tag{1.59}$$

Here, s_X is the standard deviation of variable X and s_Y is the standard deviation of variable Y. It follows that [39]:

$$-1 \leq r \leq 1 \tag{1.60}$$

The more r is close to -1, the more X and Y are **negatively correlated.** The more r is close to 1, the more X and Y are **positively correlated**. When r is around 0, the two variables have little to no correlation.

Spearman coefficient

The Spearman correlation coefficient is a non-parametric technique that uses ranks instead of the actual values of the variables under study. Given two variables X and Y, we replace a given value of x in X by its rank $R(x)$ and a given value of y in Y by its rank $R(y)$. We denote the Spearman coefficient as r_S and compute it this way:

$$r_S = \frac{cov(R(X), R(Y))}{s_{R(X)} s_{R(Y)}} \tag{1.61}$$

where $-1 \leq r_S \leq 1$. In the simplified case of no ties in ranks, we compute r_S as follows:

$$r_S = 1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n(n^2-1)} \tag{1.62}$$

where $d_i = R(x_i) - R(y_i)$ [63]. We typically use the Spearman coefficient to assess the degree of a monotonic relationship between two variables. Just like in Pearson's case, the more r_S is close to -1, the more X and Y are **negatively correlated**. The more r_S is close to 1, the more X and Y are **positively correlated**. When r_S is around 0, the two variables are likely not to be correlated.

How-To 1.23 (Covariance and Correlation in Excel 2013) To calculate $cov(X,Y)$ in Excel, use this function $= COVARIANCE.S(X,Y)$. To calculate $\rho(X,Y)$, use this function $= CORREL(X,Y)$.

How-To 1.24 (Covariance and Correlation in Minitab 18) In Minitab, you can calculate the covariance and correlation between two variables by clicking on *Stat > Basic Statistics > chose Correlation... or Covariance... > select your data > OK.*

How-To 1.25 (Python 3.6)

Script 1.8: A script for calculating the covariance and correlation in Python 3.6

```
#import numpy
import numpy as np
x1 = [...] # first variable
x2 = [...] # second variable
```

```
# print the covariance
print(np.cov(x1, x2)[0][1])
# print the correlation of coefficient
print(np.corrcoef(x1, x2)[0][1])
```

1.2.17 Regression

We use regression techniques to create a mathematical model of the relationship between dependent and independent variables. Here, we only consider a **linear regression** model with one dependent variable.

One dependent variable and one independent variable

Let's assume that we have two variables X and Y, where Y is the dependent variable and X the independent variable. The corresponding linear regression model follows.

$$Y = \beta_0 + \beta_1 X + \epsilon \tag{1.63}$$

Both β_0 and β_1 are regression coefficients. The ϵ term denotes the regression error that we refer to as residuals. We estimate β_1 using $\hat{\beta}_1$, as follows:

$$\hat{\beta}_1 = r\frac{s_Y}{s_X} \equiv \frac{cov(X,Y)}{var(X)} \tag{1.64}$$

where $var(X)$ is the variance of X and r is the Pearson correlation coefficient between X and Y. We calculate $\hat{\beta}_0$ to approximate β_0 this way:

$$\hat{\beta}_0 = \overline{Y} - \beta_1\overline{X} \tag{1.65}$$

where $\overline{Y}$ is the mean of Y and $\overline{X}$ is the mean of X [39]. We can predict the values of Y using $\hat{Y}$, as follows:

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X \tag{1.66}$$

The coefficient $\hat{\beta}_1$ represents the **expected change in $\hat{Y}$ per unit change in** X. We can think of $\hat{\beta}_0$ as the intercept of the line $\hat{Y}$. We obtain the error term this way:

$$\epsilon = Y - \hat{Y} \tag{1.67}$$

The sum of squares of the error SS_{error} is given by ϵ^2 as follows:

$$\begin{aligned} \epsilon^2 &= (Y - \hat{Y})^2 && (1.68) \\ &\equiv SS_{error} && (1.69) \end{aligned}$$

We assume ϵ to be a normally distributed random vector with mean zero and variance σ^2. The unbiased estimator of σ^2 is given by $\hat{\sigma}^2$ and is calculated as follows:

$$\hat{\sigma}^2 = \frac{SS_{error}}{n-p} \tag{1.70}$$

where $n-p$ represents the degrees of freedom. Here, n is the size of Y and p is the number of independent variables k plus one for the intercept. That is, $p = k+1$ [44].

One dependent variable and multiple independent variables

When we have more than one independent variable, say k, we create a multiple linear regression model as follows:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k + \epsilon \tag{1.71}$$

where $X_1,\ X_2, \ldots, X_k$ are independent variables, $\beta_0,\ \beta_1, \ldots, \beta_k$ are regression coefficients, and ϵ is the error term. From the least-squares method, we obtain the β coefficients as follows:

$$\hat{\beta} = (X'X)^{-1}X'Y \tag{1.72}$$

where $'$ symbol indicates transpose. Here, X is a matrix of size $n \times p$, Y is a vector of size $n \times 1$, and β is a vector of size $p \times 1$. Our $\hat{Y}$ model looks as follows:

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X_1 + \hat{\beta}_2 X_2 + \cdots + \hat{\beta}_k X_k \tag{1.73}$$

The coefficient $\hat{\beta}_j$ (for $j : 0, 1, \ldots, k$) represents the **expected change in $\hat{Y}$ per unit change in X_j while holding all other independent variables constant**. If all independent variables were zero, $\hat{Y}$ would equal β_0. As before, $\epsilon = Y - \hat{Y}$ and $SS_{error} = \epsilon^2$

1.2.18 Goodness-of-fit

The techniques that we utilize to assess the goodness-of-fit of a linear regression model are the coefficient of determination, residual plots, and hypothesis testing.

Coefficient of determination

We denote the coefficient of determination using R^2 to represent the percent of the variation in Y that is explained by the variation in independent variables. We recall that:

$$SS_{error} = (Y - \hat{Y})^2 \tag{1.74}$$

Additionally, it follows that [38]:

$$\begin{aligned} SS_{regression} &= (\hat{Y} - \overline{Y})^2 && (1.75) \\ SS_{total} &= (Y - \overline{Y})^2 && (1.76) \end{aligned}$$

We calculate R^2 as follows:

$$R^2 = \frac{SS_{regression}}{SS_{total}} \equiv 1 - \frac{SS_{error}}{SS_{total}} \tag{1.77}$$

where $0 \le R^2 \le 1$. The higher the value of R^2, the better the goodness-of-fit of our model.

Remark: *If we added more variables, necessary or unnecessary, to our regression model,* SS_{error}

would decrease per Equation 1.70, which would in turn increase R^2 per Equation 1.77. Accordingly, we must be careful when interpreting R^2. We recommend using $R^2_{adjusted}$, instead of R^2, to determine the goodness-of-fit of a regression model. We find $R^2_{adjusted}$ this way:

$$R^2_{adjusted} = 1 - \frac{SS_{error}/(n-p)}{SS_{total}/(n-1)} \tag{1.78}$$

where $n-1$ is the degrees of freedom of Y and $n-p$ is the degrees of freedom of the residuals [44].

Residual plots

Besides R^2 and $R^2_{adjusted}$, we can also use residual plots to assess the goodness-of-fit of our linear regression model. If a model is a good fit, we expect the residuals to be normally distributed with mean zero and a constant variance. So, the normal plot of the residuals should indicate that the points fall alongside the fitted line. Furthermore, we expect to see random behavior around a horizontal line when we plot ϵ versus $\hat{Y}$. Any non-random behavior indicates poor fit, which suggests non-linear behaviors [39]. Statistical packages such as Minitab also include p-values to facilitate the interpretation of the residual plots.

Hypothesis testing of regression models

In addition to the coefficient of determination and residual plots, we can also employ hypothesis testing to assess the goodness-of-fit of our regression model. In this test, we conjecture that the partial regression coefficients of the model are zero, as follows:

$$H_0 \ : \ \beta_1 = \beta_2 = \cdots = \beta_k = 0 \tag{1.79}$$

$$H_1 \ : \ \textit{At least one } \beta_j \neq 0, \textit{ for j:1,2,...,k} \tag{1.80}$$

Using the analysis of variance (ANOVA) technique, we obtain the **test statistic** as follows:

$$F_0 = \frac{MS_{regression}}{MS_{error}} \tag{1.81}$$

The **critical value** is given by:

$$F_{\alpha,\,k,\,n-k-1} \tag{1.82}$$

where α is the significance level, n is the size of Y, and k is the number of independent variables. A good-fit model will have at least one non-zero coefficient. That is, the test statistic will be greater than the critical value or the p-value less α. We could also test the hypothesis that the coefficient β_j (for $j : 1, \ldots, k$) is zero using the t-test, as follows [38]:

$$H_0 : \beta_j = 0 \tag{1.83}$$

$$H_1 : \beta_j \neq 0 \tag{1.84}$$

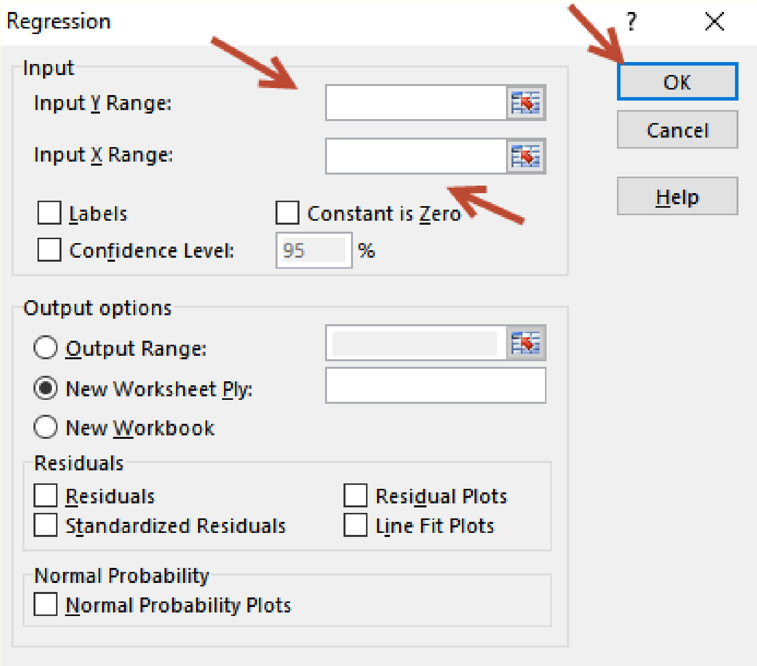
We obtain the test statistic this way [38]:

$$t = \frac{\beta_j - 0}{S_{\beta_j}} \tag{1.85}$$

where the $zero$ in the Equation 1.85 comes from the null hypothesis and S_{β_i} is the standard error of the coefficient β_j. The degrees of freedom for this test are given by $n - p$, where n is the number of observations and $p = k + 1$, as before. We obtain the p-value for this test, as illustrated in How-To 1.27.

How-To 1.26 (Linear Regression in Excel 2013)
Open the Data Analysis add-in > *Regression* > *Input Y Range for dependent variable* > *Input X Range for independent variables* > Check *Labels* if you want to include the column headings > Check the boxes of *Residuals plots* to check for the model goodness-of-fit. See the snapshot in Figure 1.14.

Figure 1.14: Regression using the Data Analysis add-in, Excel 2013

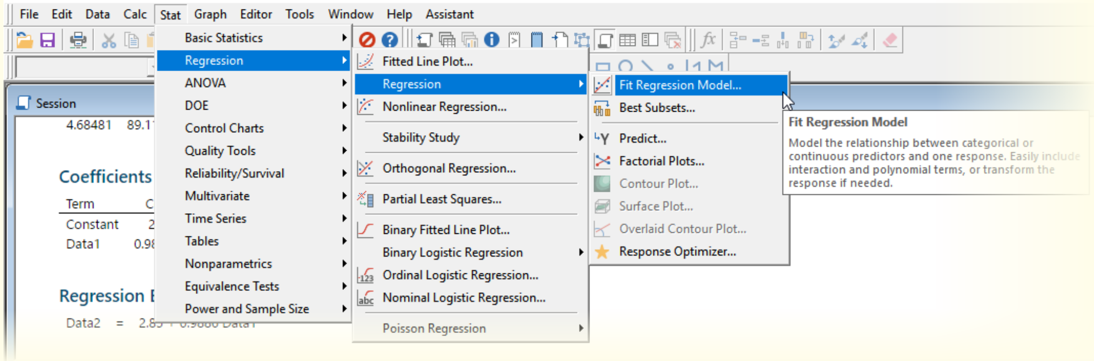


How-To 1.27 (The p-value of the t-test in Excel 2013)
To obtain the p-value of the t-test in Excel 2013, use this equation: $= T.DIST.2T(ABS(t), n - p)$

How-To 1.28 (Linear Regression in Minitab 18)
Click on *Stat* > *Regression* > *Regression* > *Fit Regression Model...*On the screen that pops up, *input your Y data in the Responses and your X data in the Continuous predictors* > On the same screen, click on *Graphs* > *Check the boxes for Normal probability plot of residuals* and *Residuals versus fits* >*OK* > *OK*. See the snapshot

in Figure 1.15.

Figure 1.15: Options for fitting a regression model, Minitab 18



Fit Regression Model

Model the relationship between categorical or continuous predictors and one response. Easily include interaction and polynomial terms, or transform the response if needed.

How-To 1.29 **(Python 3.6)**

Script 1.9: A script for linear regression in Python 3.6

```
# import modules
import statsmodels.api as sm
X = [[...],[...],..., [...]] # create a matrix of independent
    variables
y = [...]# create an array of the dependent variable
X = sm.add_constant(X)# Add the intercept
model = sm.OLS(y, X).fit()# fit an ordinary regression model
model.summary()# The summary includes several goodness-of-fit
    measures
```

1.3 Control charts

In this section, we introduce two major types of control charts, namely **Shewhart** and **time-weighted**.

1.3.1 Shewhart control charts

The model

Walter A. Shewhart is credited for introducing the concept of control charts in the 1920s while working at Bell Labs [44]. As summarized in Box 1.2, the Shewhart's model of a control chart has three main components: an upper control limit (UCL), a centerline (CL), and a lower control limit (LCL).

Box 1.2 Shewhart's model of control charts

The construct of a Shewhart control chart is as follows:

$$UCL = \mu_w + L\sigma_w \tag{1.86}$$
$$CL = \mu_w \tag{1.87}$$
$$LCL = \mu_w - L\sigma_w \tag{1.88}$$

where w is the statistic of interest. Here, μ_w and σ_w symbolize the mean and standard deviation of w, respectively. Letter L denotes the distance measured in standard deviation units from CL to LCL or UCL [44].

Control charts built following Box 1.2 are referred to as **Shewhart control charts**.

The types of Shewhart control charts

The two common categories of Shewhart's control charts are **variable** and **attribute**. We use **variable** charts when a process generates continuous independent data. Typical variable charts include ImR, $XbarR$, and $XbarS$. We employ ImR charts when the sample size $n = 1$. We apply $XbarR$ charts when $1 < n \leq 10$. When $n > 10$, we use $XbarS$ charts. The normal distribution is typically assumed in all variable charts. We apply **attribute** charts when a process generates independent discrete data. Typical attribute charts include p, np, c, u, g, and h. The p chart relates to the Bernoulli random variable. The np chart relates to the binomial random variable. The c and u assume the Poisson random variable. The g and h charts presume the geometric random variable.

How do Shewhart control charts work?

An analogy that is often used to describe the mechanics of a Shewhart control chart is that of *hypothesis testing*. For demonstration purposes, let's consider the following hypothesis of a normally distributed process:

$$H_0 : \mu = \mu_0 \tag{1.89}$$
$$H_1 : \mu \neq \mu_0 \tag{1.90}$$

Here, μ is the process mean and μ_0 is the hypothesized mean that we estimated from the process samples. The standard deviation σ is given, otherwise, we are able to obtain its unbiased estimator from the process samples. If $\bar{x}_i$ is the mean of sample i of this process, the following test statistic L applies to the hypothesis under consideration:

$$L = \frac{\bar{x}_i - \mu_0}{\sigma/\sqrt{n}} \tag{1.91}$$

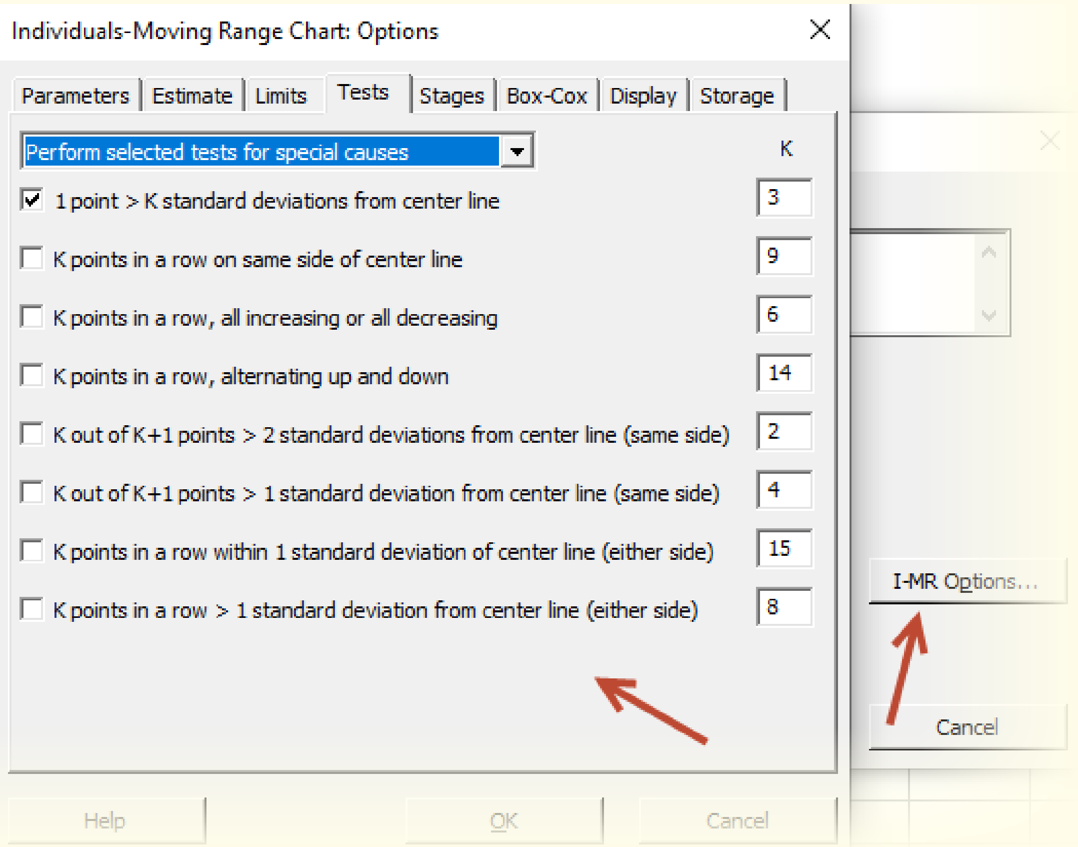
where n is the sample size. We obtain the confidence interval of the process mean μ in terms of $\bar{x}_i$ as follows:

$$\mu_0 - L\frac{\sigma}{\sqrt{n}} \leq \bar{x}_i \leq \mu_0 + L\frac{\sigma}{\sqrt{n}} \tag{1.92}$$

We can think of the lower limit of the confidence interval in Equation 1.92 as LCL, the upper limit as UCL, and μ_0 as CL. If $\bar{x}_i$ falls in this interval, then $\bar{x}_i \approx \mu_0$ and we conclude that sample i is in control. If $\bar{x}_i \neq \mu_0$, $\bar{x}_i$ will fall outside of the confidence interval, which will lead us to reject H_0. Accordingly, we will conclude that the process is **out-of-control** or has **special cause variation** at that particular sample number. Failure to reject H_0 at any sample number, is equivalent to saying that the process is in **statistical control**, or simply that the process only exhibits **common cause** variation. By common cause variation, we imply that the process **behaves randomly** within the control limits. If the process does not behave randomly within the control limits, out-of-control behaviors are likely due to the **small shifts** in the process. One way to detect such shifts is employing specialized rubrics referred to as **sensitizing rules**, as exemplified in How-To 1.30. It is important to emphasize that these rules are not universal. Another way to detect small shifts in the process is by utilizing **time-weighted** control charts.

How-To 1.30 (Sensitizing rules in Minitab 18)
For each control chart, Minitab 18 displays applicable sensitizing rules. To access these rules, click on *Chart options* > *Tests*. See Figure 1.16 for an example of sensitizing rules of an ImR control chart.

Figure 1.16: Sensitizing rules of an ImR control chart in Minitab 18



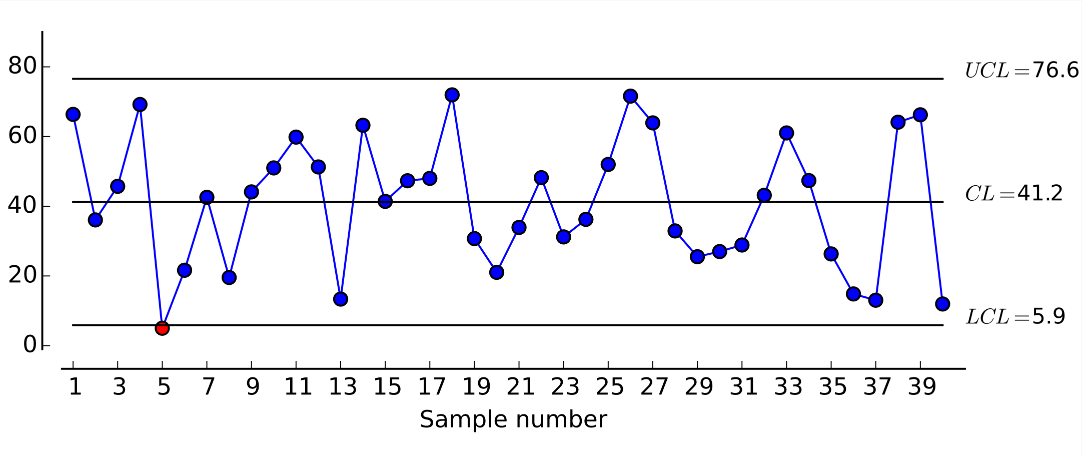
Perform selected tests for special causes

	K
☑ 1 point > K standard deviations from center line	3
☐ K points in a row on same side of center line	9
☐ K points in a row, all increasing or all decreasing	6
☐ K points in a row, alternating up and down	14
☐ K out of K+1 points > 2 standard deviations from center line (same side)	2
☐ K out of K+1 points > 1 standard deviation from center line (same side)	4
☐ K points in a row within 1 standard deviation of center line (either side)	15
☐ K points in a row > 1 standard deviation from center line (either side)	8

Example 1.1 (Special cause variation)

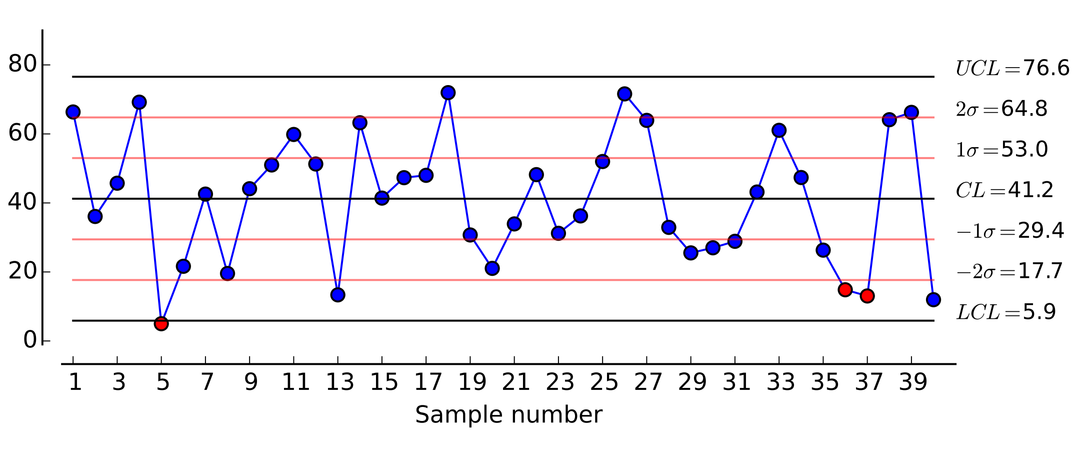
Figure 1.17 shows an example of a control chart with out-of-control behaviors since at sample #5, a point fell outside of LCL.

Figure 1.17: A control chart with one type of special cause variation



After applying the sensitizing rules, Figure 1.18 further indicates that this process has special cause variation at samples 35 and 36, given that 2 out of 3 consecutive points are between the second and third standard deviation limits on the same side.

Figure 1.18: A control chart with two types of special cause variation



1.3.2 Time-weighted control charts

Besides Shewhart control charts, other types of control charts have been suggested, mainly **time-weighted** control charts. Time-weighted control charts involve the *weighing* of samples over *time*. This structure allows for the incorporation of past information into the current performance measure, which facilitates the detection of small shifts in the process over time. Like Shewhart charts, time-weighted control charts also have UCL, CL, and LCL, but the formulas are different. Still, the interpretation of out-of-control behavior is comparable. That is, if a point falls outside of the control limits, special cause variation has occurred. We typically do not apply sensitizing rules to time-weighted controls. The three common types of time-weighted control charts that we consider in this book are the Cumulative Sum **(CUSUM)**, Exponentially Weighted Moving Average **(EWMA)**, and a regular Moving Average **(MA)**.

1.3.3 Implementing control charts

Control charts are effective at improving quality if they are implemented correctly. For example, before implementing control charts, we must decide on how big the sample sizes must be and how frequently sampling should be done. Besides, we must understand how to choose distance L and know how to detect out-of-control behaviors.

Sampling

Adequate **sample sizes** and the **frequency** of sampling are required for a successful application of control charts. In general, we need at least 25 samples to construct a control chart. The larger the sample sizes, the easier it is to detect small shifts in the process. If obtaining large samples is not feasible, we try to take small samples but dot it more frequently. In this effort, we must ensure **rational subgroups** to maximize the between sample differences while minimizing the within-sample differences. Doing so will promote the independence of the samples, an assumption that we often make when creating Shewhart control charts. An appropriate partition of the **time-ordered** samples is critical in minimizing the dependence of the subgroups [44]. In general, sampling only from units produced since the last subgroup, or from units produced under similar conditions, will help ensure rational subgroups [44].

Choosing L

As indicated earlier, letter L in a Shewhart control chart denotes the distance, in standard deviation units, from the centerline to the control limit. It is customary to choose $L = 3$, but depending on the application, this setting may be different. For example, in the type of control charts known as *risk-adjusted*, the limits are typically set to $L < 3$ [65]. Why should we care about where we set the control limits? We should care because different settings of L may lead to different conclusions about the following hypothesis:

$$H_0 : \text{process in control} \tag{1.93}$$

$$H_1 : \text{process not in control} \tag{1.94}$$

As it can be observed in the confidence interval in Equation 1.92, if L is too big, meaning too far from CL, $\bar{x}_i$ is more likely to fall within limits. This outcome increases the probability β for **Type II error**, meaning that we are likely to fail to reject H_0 when it is false. In contrast, if L is too small, $\bar{x}_i$ is more likely to fall outside of the limits, which increases the probability α of **Type I error** where we reject H_0 when it is true. We can think of type I error as a **false alarm** in a stable process [44, 65]. Both type I and II errors affect how we measure the performance of control charts. For example, using the probability of type I error α, we can construct the control chart performance measure referred to as **average run length (ARL)**, as follows:

$$ARL_\alpha = \frac{1}{\alpha} \tag{1.95}$$

The ARL_α measure also known as *in-control ARL* or ARL_0, allows us to determine the expected number of samples until a false alarm occurs in a stable process. In a normally distributed process with $L = 3$, the probability of a point falling either outside of LCL or UCL is given by $1 - 0.9973 = 0.0027$. This implies that $ARL_\alpha = 1/0.0027 \approx 370$. Hence, we can expect a false alarm at every 370^{th} sample, if the process has not shifted. If the process mean has shifted, we gauge the process performance using the ARL_β measure that we obtain like this:

$$ARL_\beta = \frac{1}{1-\beta} \tag{1.96}$$

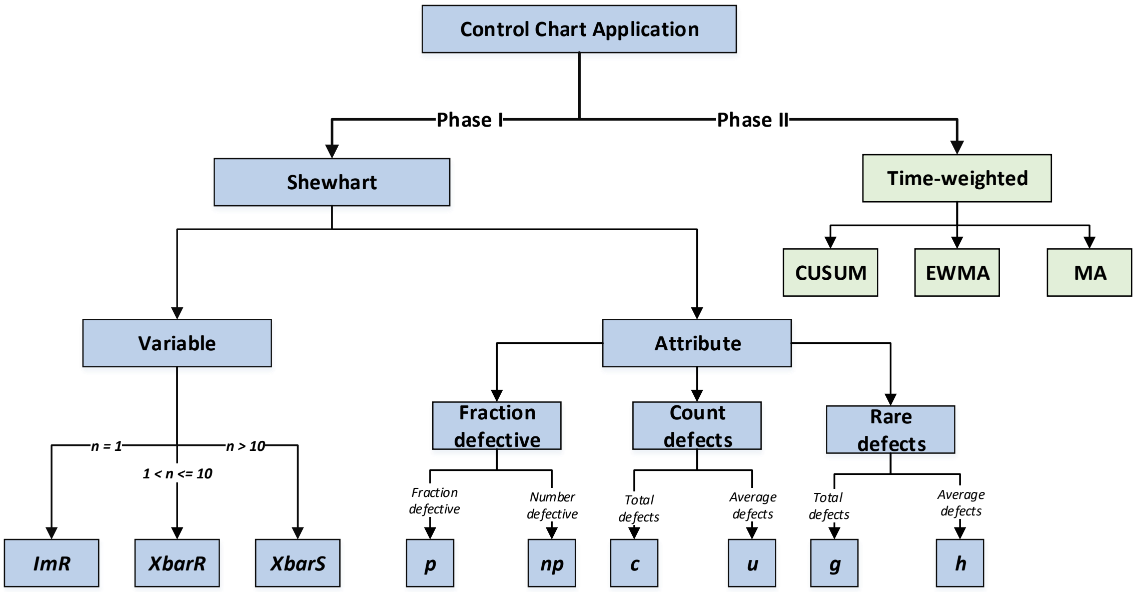
As previously noted, β is the probability of type II error and $1 - \beta$ is the power test that expresses the probability of correctly rejecting H_0. The ARL_β measure in Equation 1.96

is also referred to as *out-of-control ARL* or ARL_1. This statistic expresses the expected number of samples to detect a mean shift of $k\sigma$ in the process. In practice, we want to be able to detect a shift in the process as soon as it happens. We can accomplish this by increasing the sample size n since that decreases β [44].

Phases of control charts

There are two major phases of control chart application: **phase I** and **phase II**. In **phase I**, our main goal is to stabilize the process using Shewhart control charts. We accomplish this goal by detecting and removing **large shifts**, or **sustained shifts**, out of the process. In general, **phase I** coincides with the *Improve* period of DMAIC when we attempt to remove all assignable causes and modify control charts repetitively until the process is stable. Every time we remove an assignable cause, variability decreases and the process improves. In **phase II**, the process is stable. Hence, we are not likely to detect large shifts. The interest now is to detect and remove **small shifts**. For that, time-weighted control charts are recommended [44]. As we have discussed earlier, one could still apply Shewhart control charts with sensitizing rules to detect small shifts, but this is likely to complicate performance measures such as ARL. Besides, it is much easier to interpret time-weighted charts over sensitizing rules since we only pay attention to points that fall outside of the limits. In typical quality improvement projects, **phase II** coincides with the *Control* period of DMAIC. Figure 1.19 displays a basic decision tree of control charts by the phase of implementation.

Figure 1.19: A basic decision tree of control charts by the phase of implementation



Adjusting control charts

When applying control charts, we often have to make adjustments to alleviate misleading results. For example, we often assume independent samples when creating control charts, but this is not always true. When the process data are **autocorrelated**, we need to make adjustments to remove autocorrelation. One of the techniques to do this is fitting a time series model to sample data and then generate **residuals** that can be monitored using traditional charts. Another adjustment that we make when sample data are dependent concerns processes with correlated variables. In such instances, we have to monitor the variables jointly, instead of individually, which requires the creation of the type of control charts referred to as **multivariate**.

Additional adjustments to traditional control charts include the weighing of defects and risk factors. Indeed, if we are monitoring a unit with several defects that occur independently and these defects don't have the same importance, it may be convenient to create one control chart of the weighted defects instead of control charts of individual defects. In general, do to this, we need to have some **demerit system** that may vary by the process. Regarding risk, this is a phenomenon that occurs more often in health care than in the manufacturing sector. It's been observed that patient risk-factors tend to affect the outcomes of care [65]. So, if one is monitoring the quality of care, it is essential to account for this risk to obtain more accurate quality measures of the process [65].

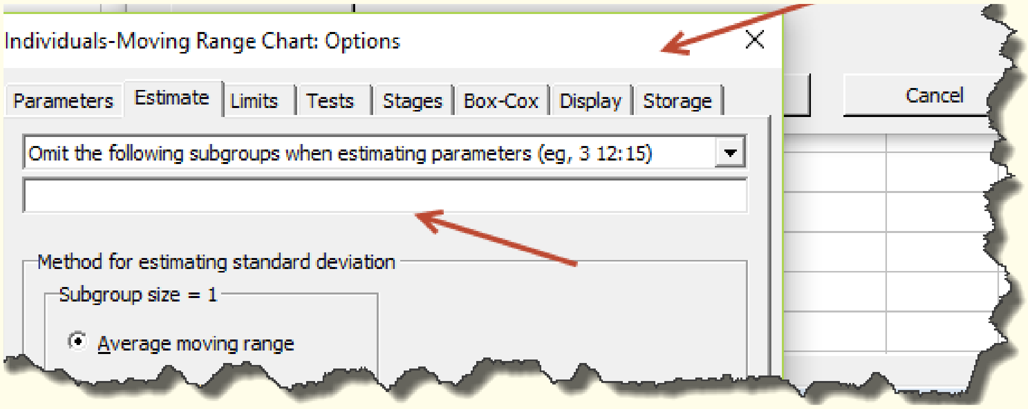
1.3.4 How to improve a process using control charts?

A process with out-of-control behaviors is **unstable**. To stabilize the process, we must attempt to assign a reason to the special cause and try to rectify it. When the cause is **assignable** and rectifiable, we remove the out-of-control sample from the process and recalculate the centerline and control limits. **By removing assignable causes, variability decreases, which in turn improves the process** [44]. A process with no special cause variation needs no improvement. But, one could still decide to **reduce common cause variation**, which amounts to creating a new process [44].

Remark: *Control charts only tell us about the samples that are out-of-control but say nothing about the actual reasons for out-of-control behaviors. Assigning causes and fixing particular sources of the special cause variation requires good knowledge of the process and a well-tuned understanding of how to apply various quality improvement tools such as Lean and designed experiments.*

How-To 1.31 (Omitting samples in Minitab 18)
To remove an assignable cause, click on *Chart Options tab of a control chart* > *Estimate* > *Enter the sample number(s)* to be omitted. See the snapshot in Figure 1.20.

Figure 1.20: Omitting samples with assignable causes in Minitab 18



Omit the following subgroups when estimating parameters (eg, 3 12:15)

Method for estimating standard deviation

Subgroup size = 1

● Average moving range

Note: After omitting a sample number, Minitab 18 calculates a new centerline and also adjusts control limits, but, depending on setting in Minitab, the omitted sample may still be visible on the chart.

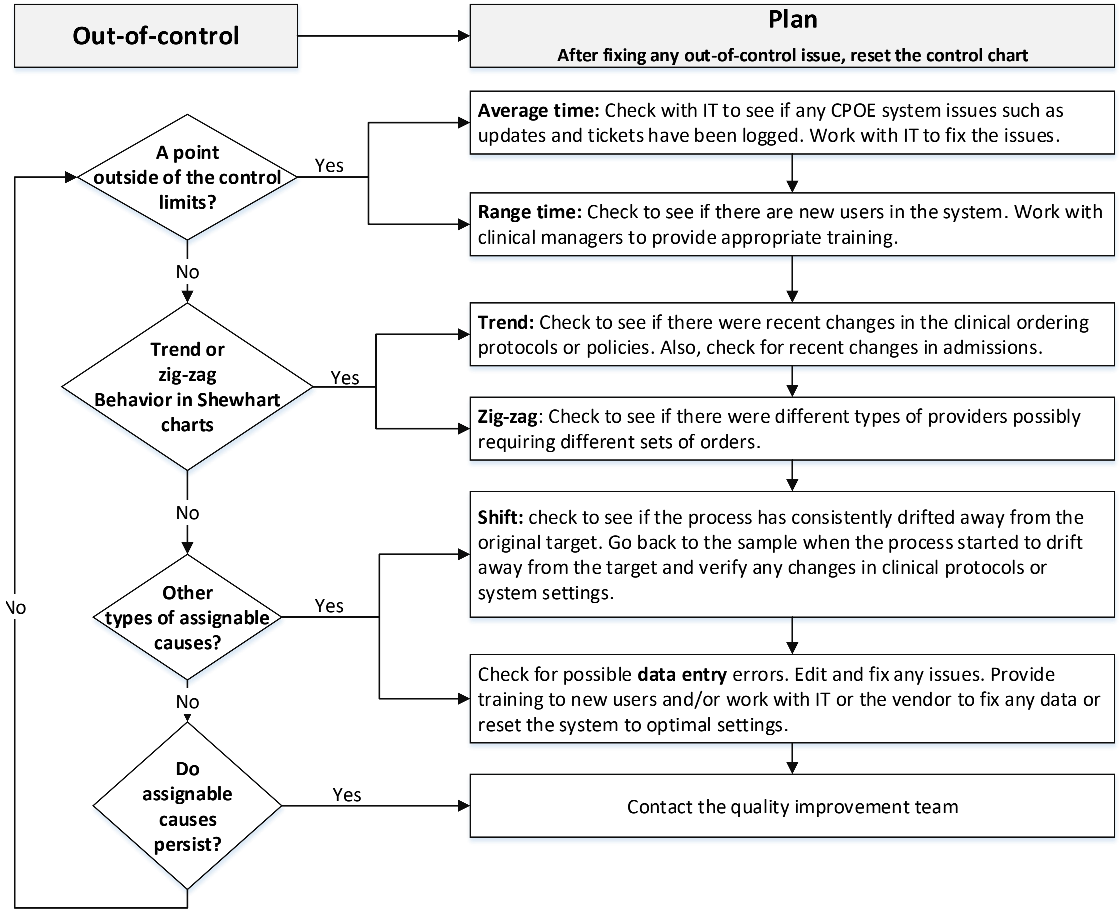
Example 1.2 (Assignable special cause variation)
A medical coding manager at Metropolis Hospital monitors the daily processing time of inpatient records. The control chart indicates that the sample from last Monday was out-of-control. After brainstorming with coders and the IT staff, the manager realized that the reason for out-of-control behaviors was encoder updates that overwrote some coding data on that day. Since the manager knows the reason for the out-of-control behaviors (that is, the manager assigned the cause), the sample from last Monday will be omitted, and the centerline and control limits recalculated. By removing the out-of-control sample, variability will decrease in the process control chart.

Example 1.3 (Common cause variation)
The office manager at Metropolis Radiology Center uses control charts to monitor the average coding time for outpatient records. No special variation causes have been noticed, but still, the manager would like to reduce common cause variation by creating a new process based on computer-assisted coding (CAC) technology. The manager believes that the new process will minimize variability by improving the consistency of coding. The manager will need new control charts to monitor the new process.

1.3.5 An out-of-control action plan (OCAP)

During the *Control* period of DMAIC, we implement phase II of the control chart application to monitor small shifts in the process. During this period, we also create an out-of-control plan, so the owner of the process knows what to do when new out-of-control behaviors are detected. We typically organize an OCAP document as a flow chart with two primary components: **checkpoints**, which are decision nodes for detecting special cause variation, and **terminators**, which are plausible corrective actions [44]. We illustrate with a generic example in Figure 1.21.

Figure 1.21: A generic OCAP document about a particular Computerized provider order entry ()CPOE) process


Out-of-control
Plan
After fixing any out-of-control issue, reset the control chart
A point outside of the control limits?
Yes
No
Average time: Check with IT to see if any CPOE system issues such as updates and tickets have been logged. Work with IT to fix the issues.
Range time: Check to see if there are new users in the system. Work with clinical managers to provide appropriate training.
Trend or zig-zag Behavior in Shewhart charts
Yes
No
Trend: Check to see if there were recent changes in the clinical ordering protocols or policies. Also, check for recent changes in admissions.
Zig-zag: Check to see if there were different types of providers possibly requiring different sets of orders.
Other types of assignable causes?
Yes
No
Shift: check to see if the process has consistently drifted away from the original target. Go back to the sample when the process started to drift away from the target and verify any changes in clinical protocols or system settings.
Check for possible data entry errors. Edit and fix any issues. Provide training to new users and/or work with IT or the vendor to fix any data or reset the system to optimal settings.
Do assignable causes persist?
Yes
No
Contact the quality improvement team

1.4 Check Your Understanding

1. What happens to quality when variation increases?

2. The term *subgroup* has the same meaning as

   (a) sample

   (b) number

   (c) population

   (d) variance

3. The manager of Metropolis Clinic has received several patient complaints about excessive waiting times. To investigate the matter, the manager performed a retrospective audit of 5 random visits each day for 25 days. From this scenario, the sample size is:

   (a) 1

   (b) 5

   (c) 12

   (d) 60

4. Which of the following is an example of a **statistic** measure of a process:

   (a) a sample mean

   (b) a sample size

   (c) a sample number

   (d) a random sample

5. In ——— sampling, each item in the process has an equal chance of being selected:

   (a) random

   (b) systematic

   (c) stratified

   (d) cluster

6. We use this formula $\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$ to calculate the sample:

   (a) mean

   (b) variance

   (c) standard deviation

   (d) median

7. Your population has a mean $\mu = 50$ and $\sigma = 10$. If you took a sample of size 7 and observed a mean of 55, what is your sampling error?

   (a) 5
   (b) 7
   (c) 10
   (d) 40

8. Your population has a mean $\mu = 50$ and $\sigma = 10$. If you took a sample of size 7 and observed a mean of 55, what is your standard error?

   (a) 3.8
   (b) 7.0
   (c) 11.9
   (d) 17.6

9. The ratio between the two sample variances is likely to follow which distribution?

   (a) normal distribution
   (b) t-distribution
   (c) F-distribution
   (d) Chi-square distribution

10. When sampling your process, you should seek to —— within subgroup differences

    (a) maximize
    (b) minimize

11. Which of the following sample statistic is a biased estimator of the related population parameter?

    (a) Mean
    (b) Median
    (c) Variance
    (d) Standard deviation

12. Which of the following distribution has the mean and variance parameters that are equal?

    (a) Geometric
    (b) Poisson
    (c) Normal
    (d) Bernoulli

13. Which statistic do you need to create a confidence interval for your population mean?

    (a) Standard error

    (b) Sampling error

    (c) Regression error

    (d) Statistic error

14. A false alarm in the process is equivalent to a:

    (a) type I error.

    (b) type II error.

15. When comparing the means in paired observations, we typically apply which statistical test?

    (a) t-test

    (b) Z-test

    (c) F-test

    (d) Chi-square-test

16. Our null hypothesis says that $H_0 : \sigma_1^2 = \sigma_2^2$. What statistic test are we likely to run?

    (a) F-test

    (b) t-Test

    (c) Z-test

    (d) M-test

17. When performing an ANOVA test, as the degrees of freedom in the error decrease, we are likely to conclude that:

    (a) There is no difference in the population means

    (b) There is a difference in the population means

18. Why are we likely to assume normal distribution during quality improvement projects?

19. What graphical technique can we use to verify the normality assumption?

20. Why do we have to adjust the coefficient of determination?

21. Given $\alpha = 0.05$, and $n = 10$, what is the t-statistic for constructing a confidence interval?

22. Given individual observations, which one of the following control charts are we likely to use to monitor variability?

(a) $\overline{MR}$

(b) $\overline{R}$

(c) $\overline{S}$

(d) S^2

23. If we want to monitor the average coding time per record, what type of control charts are we likely to use?

(a) Variable control charts

(b) Attribute control charts

24. From the operating curves, we observe that as the sample size increases

(a) α decreases.

(b) β decreases.

(c) α increases.

(d) β increases.

25. Removing ——— results in the creation of a new process.

(a) special cause variation from an existing process

(b) common cause variation from an existing process

26. When there are at least eight consecutive points on the same side of the centerline of a control chart, we likely have ——

(a) a shift in the process.

(b) zig-zag behaviors in the process.

(c) trend behaviors in the process.

(d) oscillation behaviors in the process.

27. ARL_α in a stable and normally distributed process with L = 2 is about:

(a) 3

(b) 10

(c) 20

(d) 370

28. The g chart is related to which random variable?

(a) Geometric

(b) Poisson

(c) Binomial

(d) Normal

29. How are we likely to monitor a process with variable data when the sample size is $n = 20$?

    (a) Using ImR charts

    (b) Using $XbarR$ charts

    (c) Using $XbarS$ charts

    (d) Using p and np charts

30. The $Xbar$ chart is related to which random variable?

    (a) Geometric

    (b) Poisson

    (c) Binomial

    (d) Normal

31. H_0 : *a process in control*
    H_1 : *a process not in control*
    You noticed that a point fell outside of the control limits and concluded that the special cause variation existed. Your conclusion was equivalent to:

    (a) rejecting the null hypothesis.

    (b) failing to reject the null hypothesis.

32. Sensitizing rules are mainly applied to detect ——

    (a) large shifts in the process.

    (b) small shifts in the process.

33. The individual credited with introducing control charts is ——

    (a) Deming.

    (b) Juran.

    (c) Ishikawa.

    (d) Shewhart.

34. ARL_β is also referred to as ——

    (a) in-control ARL.

    (b) out-of-control ARL.

35. How can you state the null hypothesis about the coefficients of a multiple regression model?

36. Given one dependent variable Y and one independent variable X, how would you obtain the intercept of the corresponding linear regression model?

37. Given the residual plot from a regression model, how would you determine the goodness-of-fit?

38. Consider the following multiple linear regression model $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k + \epsilon$. How can you interpret β_2 concerning Y?

39. You have created a regression model from your sample data. How can you use the total sum of squares and the error sum of squares to obtain the coefficient of determination?

40. From fitting a multiple regression model in Excel, with 20 observations, you obtained the results in the following table:

	Coefficients	*Standard Error*	*t Stat*
Intercept	89.75368	7.970321	11.26099
X1	1.7707	0.318356	5.562016
X2	1.669831	0.556528	3.000446

   (a) Write down the regression equation for this case

   (b) What can you conclude about the significance of the coefficient of your regression model? Justify your answer using p-values

CHAPTER 2

Shewhart Control Charts

In this chapter, we widen our discussion of Shewhart control charts. We focus on variable and attribute charts. We demonstrate how to create and implement these charts using Excel, Python, and Minitab software. We include several examples as well as a necessary review of statistical concepts that concern particular control charts under discussion.

Key concepts and tools: *Control charts; Variable charts; Attribute charts; Out-of-control behaviors; Assignable causes; Stabilize the process;*

Major objectives
After studying this chapter, you will be able to:

1. *Define key concepts of Shewhart control charts*
2. *Distinguish different types of Shewhart control charts*
3. *Explain the statistical formulation of Shewhart control charts*
4. *Recognize out-of-control behaviors in Shewhart control charts*
5. *Use Shewhart control charts to improve a process*
6. *Develop an implementation strategy of Shewhart control charts*
7. *Survey data transformation techniques when monitoring rare events*
8. *Create variable control charts using Excel, Python, and Minitab*
9. *Create attribute control charts using Excel, Python, and Minitab*
10. *Justify the role of Shewhart control charts in improving quality in health care*

2.1 Introduction

In this chapter, we expand on the discussion of Shewhart control charts that we introduced in Chapter 1. We recall that we implement these charts during phase I of chart application to detect and remove **large** or **sustained shifts** out of the process. When a process generates continuous independent data, we apply **variable** control charts. Examples of continuous data in health care include measurements of waiting time, patient's height and weight, infusion rate, service charges, and case mix index. The types of variable charts that we consider here include ImR, $XbarR$, and $XbarS$. We employ ImR charts when the sample size $n = 1$. We apply $XbarR$ charts when $1 < n \leq 10$. When $n > 10$, we use $XbarS$ charts. The normal distribution is typically assumed in all variable charts.

Besides variable charts, we also consider **attribute** charts that we apply when a process generates independent discrete data of defects. The types of attribute charts that we discuss include p, np, c, u, g, and h. When a sample unit has **one or more defects**, we can classify it as being either **defective** or **non-defective**. The p chart, which relates to the Bernoulli random variable, can be used to monitor the fraction of defective units. The np chart is derived from the binomial random variable and is applied to monitor the expected number of defective units. The c and u charts relate to the Poisson random variable, and we can employ them to monitor the rate of defects. The g and h charts presume the geometric random variable and are applicable for monitoring rare events. To monitor the time between rare events, we apply traditional variable charts.

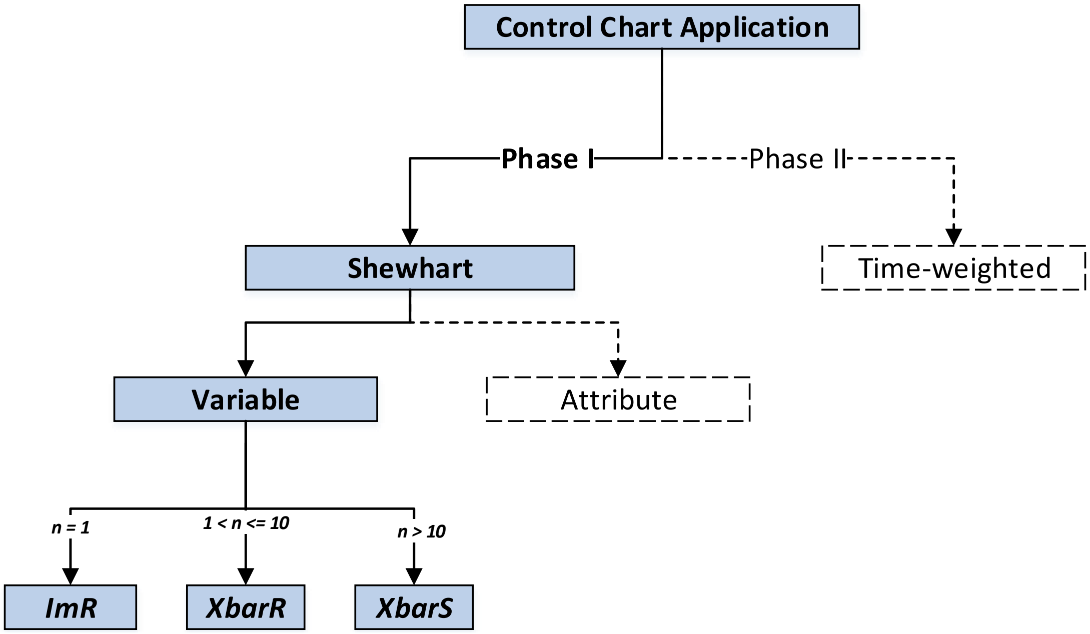
Throughout this chapter, we include several How-To clauses to demonstrate how to implement Shewhart control charts using Excel, Python, and Minitab software. We also include a basic review of the statistical concepts behind each chart we that discuss.

2.2 Variable control charts

Figure 2.1 maps the types of variable control charts that we discuss in this chapter, namely ImR, $XbarR$, and $XbarS$ charts. The following scenarios exemplify health care processes that can be monitored using variable charts.

Scenario 1: When patients visit the emergency department (ED), they are assigned an emergency severity index (ESI), and the most seriously ill patients (ESI-level 1) are prioritized. The ED manager would like to use variable control charts to monitor the time it takes for ESI-level 1 patients to be seen by the physician. Currently, there are no electronic means of capturing this amount of time. As a result, the manager has decided to randomly sample 5 ESI-level 1 patients and monitor their waiting time using ImR charts. For more discussion about using variable control charts to monitor ED waiting times, see Ross (2013) [53].

Figure 2.1: A basic map of variable control charts



Scenario 2: The CIO[1] of Metropolis Hospital has received several complaints about delays in the queries for diagnostic images from the PACS[2] system. The CIO understands that the efficiency of the retrieval of patient information is critical for the timeliness of medical care delivery. Depending on the type of information, several databases may have to be queried, and data may have to be synthesized and displayed in a tailored format for the user [57]. The CIO has put together a team of experts to help improve the PACS query process. To start, the team has collected 25 samples of size $n = 15$ and established preliminary $XbarS$ control charts.

Scenario 3: The director of Health Informatics at Metropolis Hospital manages an involved department that encompasses health information management, data analytics, and quality reporting. To keep track of the department's expenditures, the director has decided to use variable control charts to monitor biweekly expenses. The budget target set by the finance department will be used to set up an appropriate variable control chart. See Carey and Lloyd (1995)[10] to read more about a case study that discusses the use of variable control charts to monitor financial metrics.

Scenario 4: You are the manager of the revenue cycle at your hospital, and your CFO[3] has asked you to help decrease the average collection period (ACP)[4]. You understand that some of the processes related to ACP are outside of your direct control,

[1] **CIO:** Chief Information Officer

[2] **PACS:** Picture Archiving and Communication System

[3] **CFO**: Chief Financial Officer

[4] **ACP:** In healthcare finance, ACP is also known as days-in-patient account receivable [28]

but you do have a contribution to make such as reducing the number of days to mail patient co-payment invoices. To monitor your process, you have decided to use $XbarR$ control charts. A case study discussing the use of variable control charts to monitor days to mail patient invoices can be found in Carey and Lloyd (1995)[10].

2.2.1 Statistics for variable charts

To construct variable charts, we must know or be able to estimate the process mean and standard deviation. Unless **standard values** are provided, we always estimate preliminary variable control charts using **at least 25 samples**. We assume that our samples are independent and identically distributed following the normal random variable. We calculate the mean $\bar{x}_i$ of sample i as follows:

$$\bar{x}_i = \frac{1}{n}\sum_{j=1}^{n} x_{ij} \tag{2.1}$$

for $i : 1, \ldots, m$, where m is the total number of samples. We obtain the overall mean $\bar{\bar{x}}$ this way:

$$\bar{\bar{x}} = \frac{1}{m}\sum_{i=1}^{m} \bar{x}_i \tag{2.2}$$

When the sample size is variable, we obtain $\bar{\bar{x}}$ like this:

$$\bar{\bar{x}} = \frac{\sum_{i=1}^{m} n_i \bar{x}_i}{\sum_{i=1}^{m} n_i} \tag{2.3}$$

We recall from Chapter 1 that the sample mean $\bar{\bar{x}}$ is the unbiased estimator of the process mean μ. We calculate the standard deviation of sample i as follows:

$$s_i = \sqrt{\frac{\sum_{j=1}^{n}(x_j - \bar{x}_i)^2}{n-1}} \tag{2.4}$$

where $n-1$ signifies the degrees of freedom. Given m number of samples, we obtain the average of standard deviations $\bar{s}$ this way:

$$\bar{s} = \frac{1}{m}\sum_{i=1}^{m} s_i \tag{2.5}$$

When the sample size is variable, we obtain $\bar{s}$ as follows [44]:

$$\bar{s} = \sqrt{\frac{\sum_{i=1}^{m}(n_i - 1)s_i^2}{\sum_{i=1}^{m}(n_i - m)}} \tag{2.6}$$

where s_i^2 is the variance of sample i [44]. As indicated in Chapter 1, when the sample size $n > 10$, we use $\bar{s}$ to derive the unbiased estimator of the process standard deviation σ, as follows:

$$\sigma \approx \hat{\sigma} = \frac{\bar{s}}{c_4} \tag{2.7}$$

Parameter c_4 depends on the sample size n in Appendix Table 12. When the sample size $1 \leq n \leq 10$, we use the range method to estimate variability. We obtain the range R_i of sample x_i, for $i : 1,\ldots,m$, where m is the sample number, as follows:

$$R_i = \max(x_{ij}) - \min(x_{ij}) \tag{2.8}$$

for $j : 1,\ldots,n$. We use the $\min()$ function to find the minimum value, and $\max()$ function to find the maximum value. Given m number of samples, we derive $\overline{R}$, the average of the ranges, this way:

$$\overline{R} = \frac{1}{m}\sum_{i=1}^{m} R_i \tag{2.9}$$

We use the $\overline{R}$ statistic to estimate the process standard deviation as follows:

$$\sigma \approx \hat{\sigma} = \frac{\bar{R}}{d_2} \tag{2.10}$$

where the parameter d_2 is obtained from Appendix Table 12 by the constant sample size n. In the special case of individual observations, we calculate the *moving range (MR)* over the span of two observations this way:

$$MR_i = |x_i - x_{i-1}| \tag{2.11}$$

for $i : 2,\ldots,m$. Here, the |.| symbol signifies the absolute value, x_i is the current observation, and x_{i-1} is the previous observation. The average of moving ranges, $\overline{MR}$, in m samples is obtained as follows:

$$\overline{MR} = \frac{1}{m-1}\sum_{i=2}^{m} MR_i \tag{2.12}$$

We approximate σ like this:

$$\sigma \approx \hat{\sigma} = \frac{\bar{M}R}{d_2} \tag{2.13}$$

From Appendix Table 12, when $n = 2$, $d_2 = 1.128$.

2.2.2 *ImR* control charts

We use ImR control charts, also denoted as I-MR, to monitor a process when the sample size $n = 1$. The part of I monitors individual observations, whereas the part of MR monitors variability using moving ranges.

Formulation

Box 2.1 summarizes the formulas for ImR control charts.

Box 2.1 Formulas for ImR control charts when $L = 3$

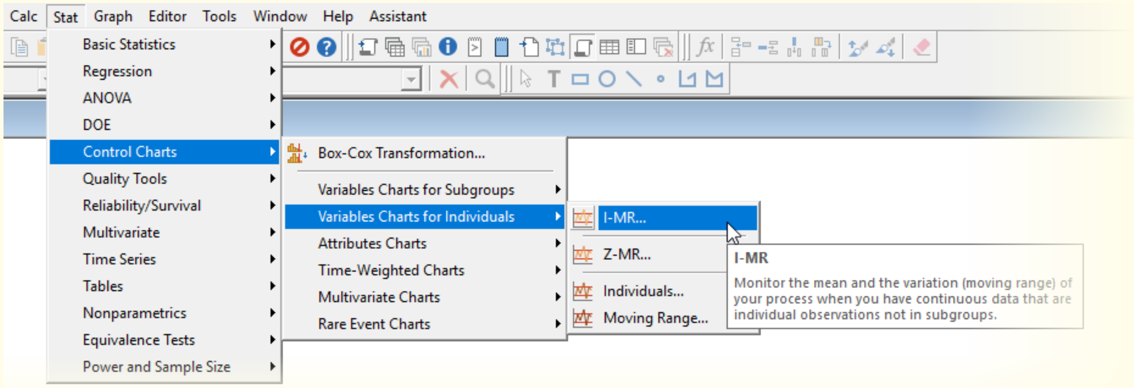
The formulas for ImR charts are as follows:

	I **chart**	MR **chart**
UCL	$\bar{x} + 3\frac{\overline{MR}}{d_2}$	$D_4\overline{MR}$
CL	$\bar{x}$	$\overline{MR}$
LCL	$\bar{x} - 3\frac{\overline{MR}}{d_2}$	$D_3\overline{MR}$

Here, $\bar{x}$ is the overall mean of observations. We obtain parameters d_2, D_3, and D_4 in Appendix Table 12 when $n = 2$. Since we can't have negative observations, when $LCL < 0$, we set $LCL = 0$.

How-To 2.1 (I-MR charts in Minitab 18)
To construct ImR charts in Minitab 18, click on *Stat* > *Control Charts* > *Variable Control Charts for Individuals* > *I-MR* > *Select your data* > *OK*. See the snapshot in Figure 2.2.

Figure 2.2: Options for I-MR control charts in Minitab 18



I-MR

Monitor the mean and the variation (moving range) of your process when you have continuous data that are individual observations not in subgroups.

How-To 2.2 (Python 3.6)

Script 2.1: A script for creating I-MR charts in Python 3.6

```
#VARIABLE CHARTS- IMR charts
#import modules
from pandas import*
from pylab import*
from numpy import*
import seaborn as sns

data = read_excel('your directory')
xr = data.xt
xbar = mean(xr)
d2 = 1.128
D3 = 0.
D4 = 3.267
mr = [abs(xr[i]-xr[i-1]) for i in range(1, len(xr))]
mrbar = mean(mr)
UCLa = [xbar + 3.*(mrbar/d2)]*len(xr)
LCLa = [ max(0,xbar - 3.*(mrbar/d2))]*len(xr)
CLa = [xbar]*len(xr)

markers = []
colors = []
for i in range (len(data)):
    x1 = data.ix[i]['xt']
    if x1 > UCLa[0]:
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
#Plotting
fig=figure()
ax1 =  fig.add_subplot(111)
t = arange(len(data))
ax1.plot(UCLa, 'k-', alpha = 0.5)
ax1.plot(LCLa, 'k-',alpha = 0.5)
ax1.plot(CLa, 'k-',alpha = 0.5)
ax1.plot(xr,'b-',zorder=1, alpha = 1.)
for x,y,c,m in zip(t, xr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
xlim(xmin = -0.3)
xlim(-0.3, t[-1]+1)
sns.color_palette("Blues")
sns.despine(offset=10, top = True, trim=True)
sns.axes_style({'xtick.right': False})
#label y-axis and x-axis
ax1.set_xlabel('Sample number')
ax1.set_ylabel('Individual Observations')
ax1.yaxis.set_ticks_position('left') #remove yticks from right up
ax1.xaxis.set_ticks_position('bottom') #remove yticks from right up
```

```
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL=$'+str(round(UCLa[0],2)), xy = (xlim()[1], list(
    UCLa)[-1]), xytext = (xlim()[1],list(UCLa)[-1]),fontsize = 11)
ax1.annotate ('$\overline{X}=$'+str(round(CLa[0],2)), xy = (xlim()
    [1], list(CLa)[-1]), xytext = (xlim()[1],list(CLa)[-1]),fontsize
    = 11)
ax1.annotate ('$LCL=$'+str(round(LCLa[0],2)), xy = (xlim()[1], list(
    LCLa)[-1]), xytext = (xlim()[1],list(LCLa)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(data), step = 2), arange(1, len(data)+1, step = 2))
show()

#plot MR chart
UCLa = [mrbar*D4]*len(mr)
LCLa = [max(0,mrbar*D3)]*len(mr)
CLa = [mrbar ]*len(mr)

markers = []
colors = []

for i in range (len(mr)):
    x1 = mr[i]
    x2 = UCLa[i]
    x3 = LCLa[i]
    if (x1 > x2 or  x1<x3) :
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
#Plotting Xbar
fig=figure()
ax1 =  fig.add_subplot(111)
t = arange(len(data))
ax1.plot(UCLa, 'k-', alpha = 0.5)
ax1.plot(LCLa, 'k-',alpha = 0.5)
ax1.plot(CLa, 'k-',alpha = 0.5)
ax1.plot(mr,'b-',zorder=1)
for x,y,c,m in zip(t, mr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
ylim(-0.5, 40)
xlim(-0.3, t[-1])
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel('Sample number')
ax1.set_ylabel('Moving Range')
ax1.yaxis.set_ticks_position('left') #remove yticks from right up
ax1.xaxis.set_ticks_position('bottom') #remove yticks from right up
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL=$'+str(round(UCLa[-1],2)), xy = (xlim()[1], list(
    UCLa)[-1]), xytext = (xlim()[1],list(UCLa)[-1]),fontsize = 11)
ax1.annotate (r'$\overline{MR}=$'+str(round(CLa[0],2)), xy = (xlim()
    [1], list(CLa)[-1]), xytext = (xlim()[1],list(CLa)[-1]),fontsize
    = 11)
ax1.annotate ('$LCL=$'+str(round(LCLa[-1],2)), xy = (xlim()[1], list(
    LCLa)[-1]), xytext = (xlim()[1],list(LCLa)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(data), step = 2), arange(2, len(data)+2, step = 2))
show()
```

How-To 2.3 (I-MR charts in Excel 2013)
Excel does not have a built-in option to create I-MR charts, but we can manually program the formulas from Box 2.1 as we demonstrate in Example 2.1.

Example 2.1 (I-MR charts)
The lab manager at Metropolis Hospital is interested in monitoring the turn around time (TAT) of "stat" potassium orders for each patient. A shorter TAT is preferred since potassium levels are required before patients could be taken off the heart/lung bypass machine in the cardiac unit [49]. To set up preliminary ImR control charts, the manager collected 30 random samples, as portrayed in Table 2.1.

Table 2.1: TAT, in minutes (X), of "stat" potassium orders at Metropolis Hospital

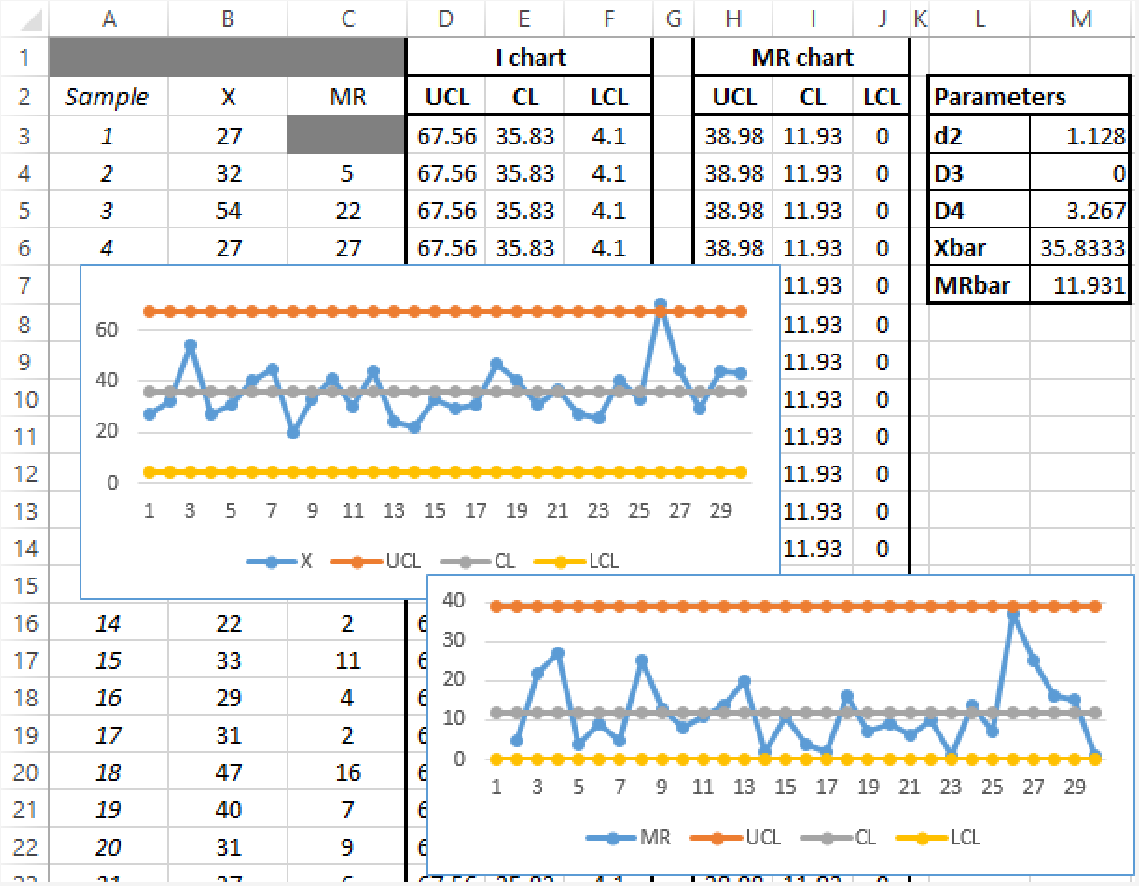
Sample#	X	*Sample#*	X
1	27	*16*	29
2	32	*17*	31
3	54	*18*	47
4	27	*19*	40
5	31	*20*	31
6	40	*21*	37
7	45	*22*	27
8	20	*23*	26
9	33	*24*	40
10	41	*25*	33
11	30	*26*	70
12	44	*27*	45
13	24	*28*	29
14	22	*29*	44
15	33	*30*	43

To help the manager construct the appropriate control chart, we start by consulting Appendix Table 12 when $n = 2$ and obtain $d_2 = 1.128$, $D_3 = 0$, and $D_4 = 3.267$. Next, we set up our spreadsheet as portrayed in Figure 2.3. To create the I-MR charts, we follow these steps:

Step 1: From the setup in Figure 2.3, we have column B containing individual observations that we monitor in the I chart. We also use this column to determine the moving ranges (MR).

Step 2: We obtain the MR values per Equations 2.11 and 2.12. For example, we calculated the MR value in cell C4 using $= ABS(B3-B4)$. We dragged down this formula to populate the rest of the values. Furthermore, we obtained $\overline{MR}$ value in cell M7 using $= AVERAGE(C4:C32)$.

Figure 2.3: A setup of I-MR control charts in Excel based on the data in Table 2.1



	A	B	C	D	E	F	G	H	I	J	K	L	M
1					I chart				MR chart				
2	Sample	X	MR	UCL	CL	LCL		UCL	CL	LCL		Parameters	
3	1	27		67.56	35.83	4.1		38.98	11.93	0		d2	1.128
4	2	32	5	67.56	35.83	4.1		38.98	11.93	0		D3	0
5	3	54	22	67.56	35.83	4.1		38.98	11.93	0		D4	3.267
6	4	27	27	67.56	35.83	4.1		38.98	11.93	0		Xbar	35.8333
7									11.93	0		MRbar	11.931
8									11.93	0			
9									11.93	0			
10									11.93	0			
11									11.93	0			
12									11.93	0			
13									11.93	0			
14									11.93	0			
15													
16	14	22	2	6									
17	15	33	11	6									
18	16	29	4	6									
19	17	31	2	6									
20	18	47	16	6									
21	19	40	7	6									
22	20	31	9	6									

Step 3: In this step, we calculate the $\overline{X}$ value. We obtained the value in cell M6 using $= AVERAGE(B3:B32)$.

Step 4: To construct the control limits, we apply the formulas in Box 2.1. For example, we obtained the UCL value in cell $D3$ by $= ROUND(\$M\$6+3*(\$M\$7/\$M\$3),2)$ and obtained the LCL value in cell $J3$ by $= \$M\$4*\$M\$7$. The CL value in cell E3 follows from $= ROUND(\$M\$6,2)$. We use the $ROUND(x,2)$ function to round x to two decimal places. The \$ sign in Excel formulas allows us to drag down the formula and easily populate the rest of the values while keeping cells with the \$ sign constant.

Step 5: In this final step, we insert line charts of columns D-F to create the I chart and line charts of columns H-J to create the MR chart.

Figure 2.4 portrays I-MR charts that we created by following the instructions in How-To 2.1. In Figure 2.5, we show similar charts that we reproduced in Python 3.6 using the script in How-To 2.2. From any of these sets of I-MR charts, we appreciate that the I chart exhibits out-of-control behavior at sample 26. The MR chart does not show any special cause variation, but it is also clear that sample 26 is too close to UCL. To improve this process, the manager needs to investigate sample 26 and remove any sources of variability. Let's assume that the manager was able to assign the reason for the out-of-control behaviors at sample 26. Then, the manager can omit that sample and recalculate the centerline and control limits, as portrayed in Figure 2.5. To omit a sample number using Minitab 18, review the instructions in How-To 1.31.

Figure 2.4: I-MR control charts created in Minitab using the data in Table 2.1

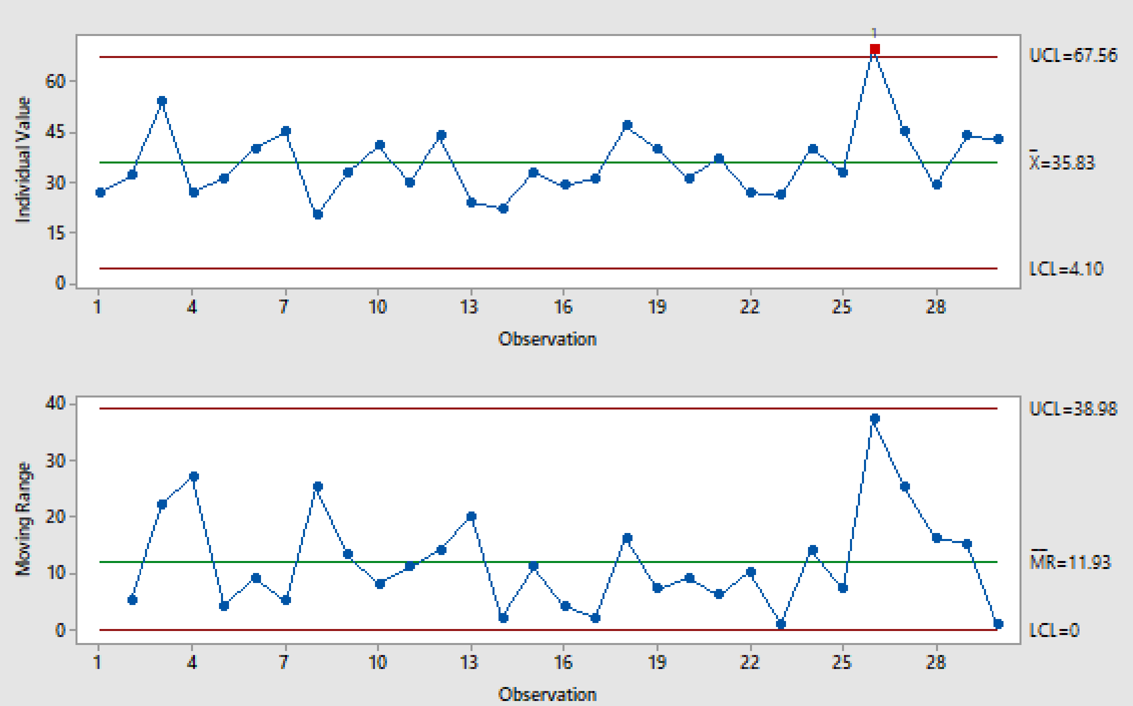
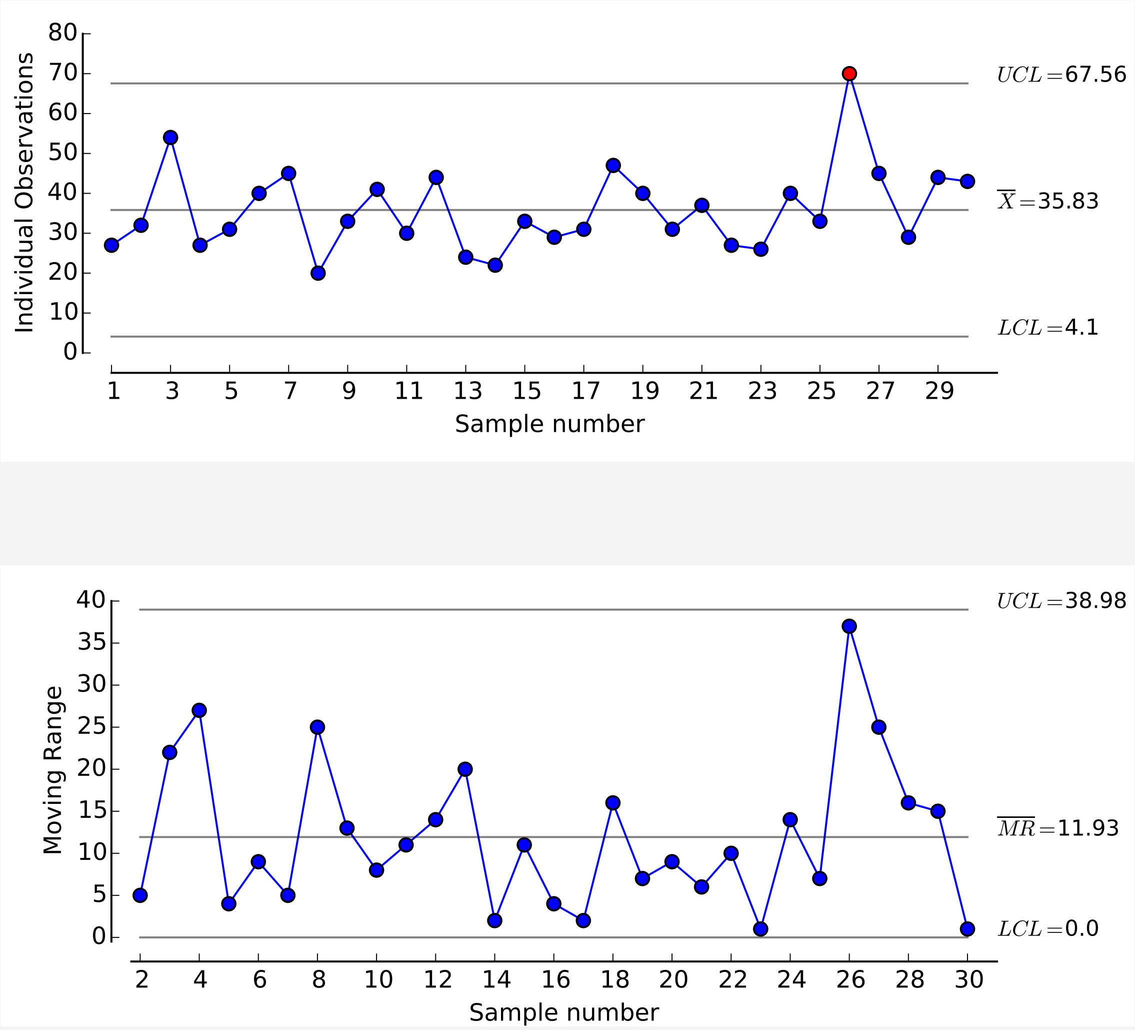


Figure 2.5: I-MR control charts produced in Python using the data in Table 2.1

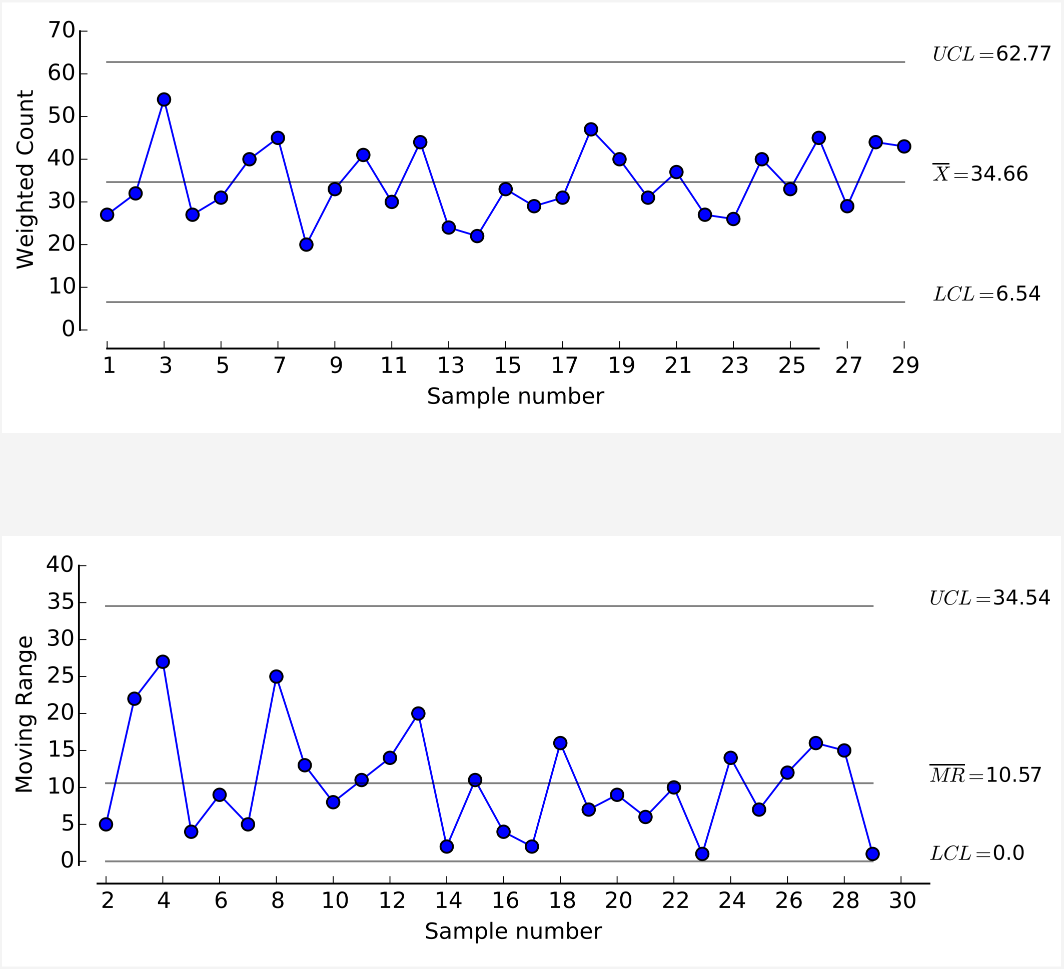


(a) I-$Chart$

(b) MR-$Chart$

Figure 2.5: Revised I-MR control charts after omitting sample 26 of the data in Table 2.1

(a) *I-Chart revised*



(b) *MR-Chart revised*

By examining the revised control charts in Figure 2.5, we notice that variability in the process has decreased, given that the control limits moved closer to the centerline. For example, in the I-$chart$, UCL decreased from 67.56 to 62.77 and LCL moved up from 4.1 to 6.54. In the MR-$chart$, UCL decreased from 38.98 to 34.54, and LCL remained at zero. In both cases, the mean behaviors also decreased. Since the new control charts are stable, the manager can use them to monitor the future process.

2.2.3 *XbarR* control charts

The $XbarR$ charts, also noted as $Xbar$-R or XmR, help us monitor the process mean behaviors using the $Xbar$ chart and the variability using the R chart. We typically employ

$Xbar$-R charts when the sample size $1 < n \leq 10$.

Formulation

Box 2.2 summarizes the formulas for these charts.

Box 2.2 The formulas for $XbarR$ control charts when $L = 3$

When μ **and** σ **standards are given**, we apply these formulas:

	$Xbar$ **chart**	R **chart**
UCL	$\mu + A\sigma$	$D_2\sigma$
CL	μ	$d_2\sigma$
LCL	$\mu - A\sigma$	$D_1\sigma$

When μ **and** σ **standards are not given**, we apply the following formulas:

	$Xbar$ **chart**	R **chart**
UCL	$\bar{\bar{x}} + A_2\overline{R}$	$D_4\overline{R}$
CL	$\bar{\bar{x}}$	$\overline{R}$
LCL	$\bar{\bar{x}} - A_2\overline{R}$	$D_3\overline{R}$

In the $Xbar$ chart, we monitor $\bar{x}_i$, the average mean in each sample i, for $i : 1, \ldots, m$, where m is the total number of samples. In the R chart, we monitor R_i, the range in each sample i. We obtain the values of A, A_2, D_1, d_2, D_3, and D_4 from Appendix Table 12 by the sample size n.

How-To 2.4 (Python 3.6)

Script 2.2: A script for creating $Xbar$-R charts in Python 3.6

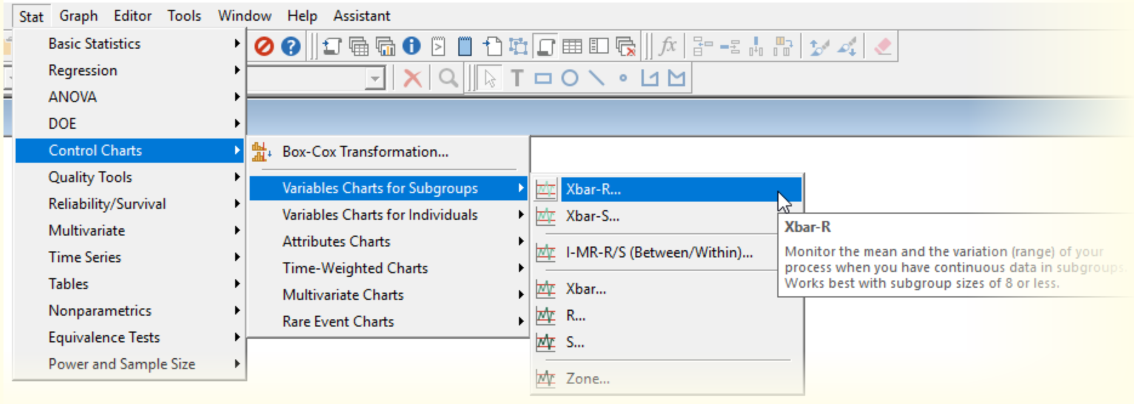
```
#VARIABLE CHARTS- XbarR charts
#Import modules
from pandas import*
from pylab import*
from numpy import*
import seaborn as sns
#import data from an excel spreadsheet
data = read_excel('your directory')
#if column names in Excel, select the column of interest (e.g., data
    = data['column name']
xr =  [mean(data.loc[i]) for i in range(len(data))]
t = arange(len(data))
rr =  [max(data.loc[i])-min(data.loc[i])*1. for i in range(len(data))
    ]
```

```
xbar = mean(xr)
rbar = mean(rr)
#Xbar chart
#parameters from Appendix Table 1. Below is an example of values when
     n = 5
A2 = 0.577
D3 = 0.
D4 = 2.114
#control limits
UCL = [xbar + A2*rbar]*len(xr)
LCL = [max(0,xbar - A2*rbar)]*len(xr)
CL = [xbar]*len(xr)
#mark red a point that falls outside of the control limits. Otherwise
    , mark the point blue.
markers = []
colors = []
for i in range (len(xr)):
    x1 = xr[i]
    x2 = UCL[i]
    x3 = LCL[i]
    if (x1 > x2 or  x1<x3) :
        markers.append(’o’)
        colors.append(’r’)
    else:
        markers.append(’o’)
        colors.append(’b’)
#plotting Xbar
fig=figure()
ax1 =  fig.add_subplot(111)
ax1.plot(UCL, ’k-’, alpha = 0.5)
ax1.plot(LCL, ’k-’,alpha = 0.5)
ax1.plot(CL, ’k-’,alpha = 0.5)
ax1.plot(xr,’b-’,zorder=1)
for x,y,c,m in zip(t, xr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel(’Sample number’)
ax1.set_ylabel(r’$\overline{X}$’)
#annotate the values of UCL, LCL, and CL
ax1.annotate (’$UCL=$’+str(round(UCL[-1],2)), xy = (xlim()[1], list(
    UCL)[-1]), xytext = (xlim()[1],list(UCLa)[-1]),fontsize = 11)
ax1.annotate (r’$\overline{\overline{X}}=$’+str(round(CL[0],2)), xy =
     (xlim()[1], list(CL)[-1]), xytext = (xlim()[1],list(CL)[-1]),
    fontsize = 11)
ax1.annotate (’$LCL=$’+str(round(LCL[-1],2)), xy = (xlim()[1], list(
    LCLa)[-1]), xytext = (xlim()[1],list(LCLa)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(data), step = 2), arange(1, len(data)+1, step = 2))
show()
#########################
#R chart
```

```
#control limits
UCL = [rbar*D4]*len(xr)
LCL = [max(0,rbar*D3)]*len(xr)
CL = [rbar]*len(xr)
#mark red a point that falls outside of the control limits. Otherwise
    , mark the point blue
markers = []
colors = []
for i in range (len(xr)):
    x1 = rr[i]
    x2 = UCL[i]
    x3 = LCL[i]
    if (x1 > x2 or  x1<x3) :
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
#plotting R-chart
fig=figure()
ax1 =  fig.add_subplot(111)
t = arange(len(data))
ax1.plot(UCL, 'k-', alpha = 0.5)
ax1.plot(LCL, 'k-',alpha = 0.5)
ax1.plot(CL, 'k-',alpha = 0.5)
ax1.plot(rr,'b-',zorder=1)
for x,y,c,m in zip(t, rr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel('Sample number')
ax1.set_ylabel('R')
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL=$'+str(round(UCL[-1],2)), xy = (xlim()[1], list(
    UCL)[-1]), xytext = (xlim()[1],list(UCL)[-1]),fontsize = 11)
ax1.annotate (r'$\overline{R}=$'+str(round(CL[0],2)), xy = (xlim()
    [1], list(CL)[-1]), xytext = (xlim()[1],list(CL)[-1]),fontsize =
    11)
ax1.annotate ('$LCL=$'+str(round(LCL[-1],2)), xy = (xlim()[1], list(
    LCL)[-1]), xytext = (xlim()[1],list(LCL)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(data), step = 2), arange(1, len(data)+1, step = 2))
show()
```

How-To 2.5 (*Xbar*-*R* **charts in Minitab 18**) To construct *Xbar*-*R* charts in Minitab 18, click on *Stat* > *Control Charts* > *Variable Control Charts for Subgroups* > *Xbar-R* > *In the drop-down menu, select* ***Observations for a subgroup are in one row of columns*** >*Select your data* > *OK.* See the snapshot in Figure 2.5.

Figure 2.5: Options for $Xbar$-R control charts in Minitab 18



Monitor the mean and the variation (range) of your process when you have continuous data in subgroups. Works best with subgroup sizes of 8 or less.

How-To 2.6 (**$Xbar$-R charts in Excel 2013**)
Excel does not have a built-in option to create $Xbar$-R charts, but we can manually program the formulas in Box 2.2 as we demonstrate in Example 2.2.

Example 2.2 (XbarR charts)
The coding manager at Metropolis Hospital is interested in setting up control charts to monitor the processing time of inpatient records. The manager defined processing time as the total time it takes, in minutes, to code and abstract a record. To get started, the manager looked up the standards about the mean and standard deviation of inpatient processing times but found none. Subsequently, the manager decided to estimate the process parameters using sample data. Table 2.2 shows the data that the manager sampled from 30 random days in 2018. On each day, the manager sampled 5 records.

Table 2.2: Samples of processing times of inpatient records at Metropolis Hospital

Sample #	x_1	x_2	x_3	x_4	x_5
1	28	32	34	32	33
2	31	27	31	28	30
3	15	31	20	34	15
4	33	32	31	31	30
5	26	27	34	29	34
6	34	27	32	30	31
7	25	34	33	33	25
8	32	32	35	26	32
9	28	33	35	29	30
10	25	32	29	28	50
11	33	32	30	35	35
12	34	29	21	32	26
13	27	30	31	31	35
14	27	28	28	34	34
15	29	33	33	25	32
16	27	26	27	25	35
17	29	33	32	30	35
18	29	27	29	25	34
19	33	27	32	33	44
20	32	27	28	29	28
21	35	32	34	32	27
22	28	33	28	40	30
23	34	32	32	30	32
24	35	25	25	28	34
25	28	25	26	31	27
26	27	33	33	34	34
27	33	31	29	27	10
28	32	31	35	31	29
29	30	27	35	33	28
30	32	25	33	25	29

Next, the manager looked up the values of parameters A_2, D_3, and D_4 in Appendix Table 12, when $n = 5$, and obtained 0.577, 0, 2.114, respectively. We help the manager create $Xbar$-R charts in Excel by following these steps:

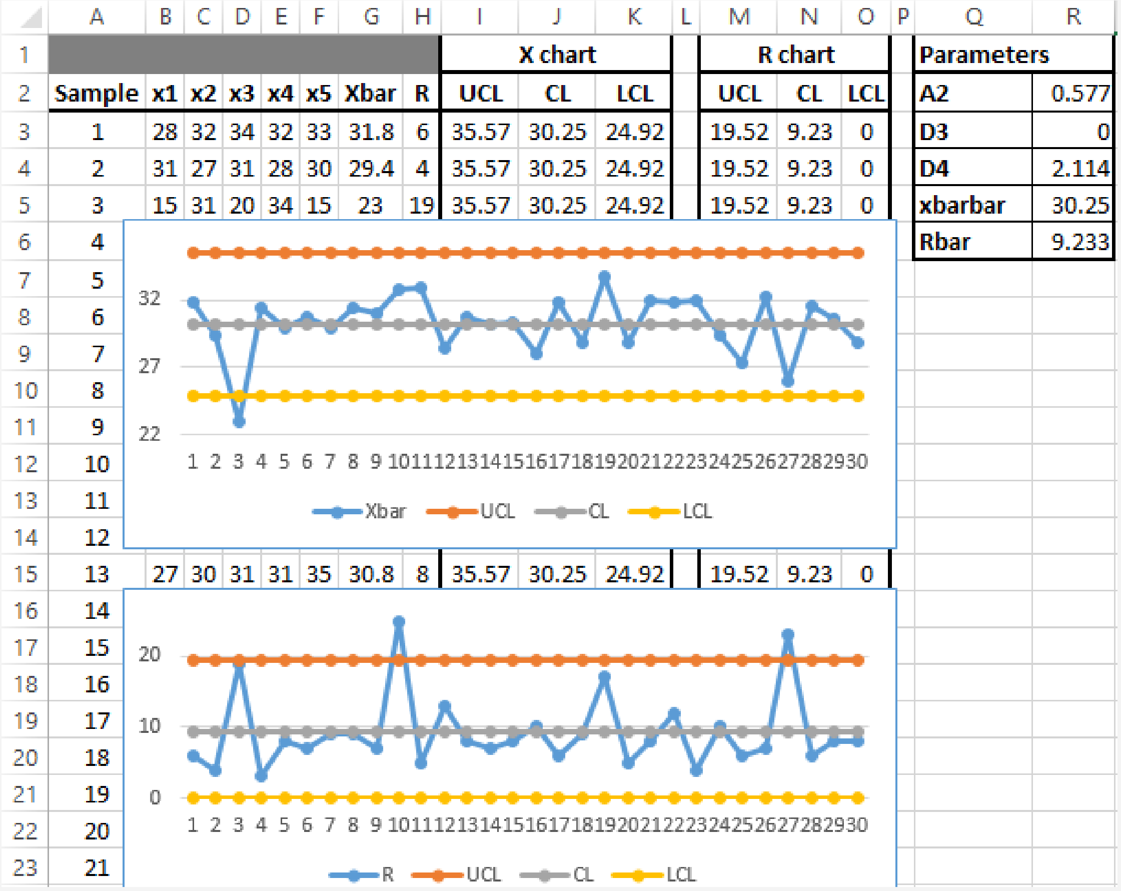
Step 1: First, we set up our spreadsheet as shown in Figure 2.6, and calculate the mean statistics per Equations 2.1 and 2.2. For example, the $Xbar$ value in cell G3 was calculated using $= AVERAGE(B3 : F3)$. This formula can be dragged down to populate the rest of the values. The value of $Xbarbar$ ($\bar{\bar{x}}$) in cell R5 was calculated using $= AVERAGE(G3 : G32)$.

Step 2: We obtain the range per Equations 2.8 and 2.9. For example, the R value in cell H3 was obtained by $MAX(B3 : F3) - MIN(B3 : F3)$. We dragged down this formula to populate the rest of the values. The $Rbar$ value in cell R6 was obtained using $= AVERAGE(H3 : H32)$.

Step 3: To construct the control limits, we applied the formulas from Box 2.2, when no standards were given. For example, we obtained the UCL value in cell I3 using $= ROUND(\$R\$5 + \$R\$2 * \$R\$6, 2)$, and obtained the LCL value in cell K3 using $= ROUND(\$R\$5 - \$R\$2 * \$R\$6, 2)$. The center limits are based on the values of $Xbarbar$ and $Rbar$. We dragged down these formulas to populate the rest of the values.

Step 4: In this final step, we insert the line charts of columns $I - K$ to construct the $Xbar$ chart, and columns $M - O$ to construct the R chart.

Figure 2.6: A setup of $Xbar$-R control charts in Excel based on the data in Table 2.2

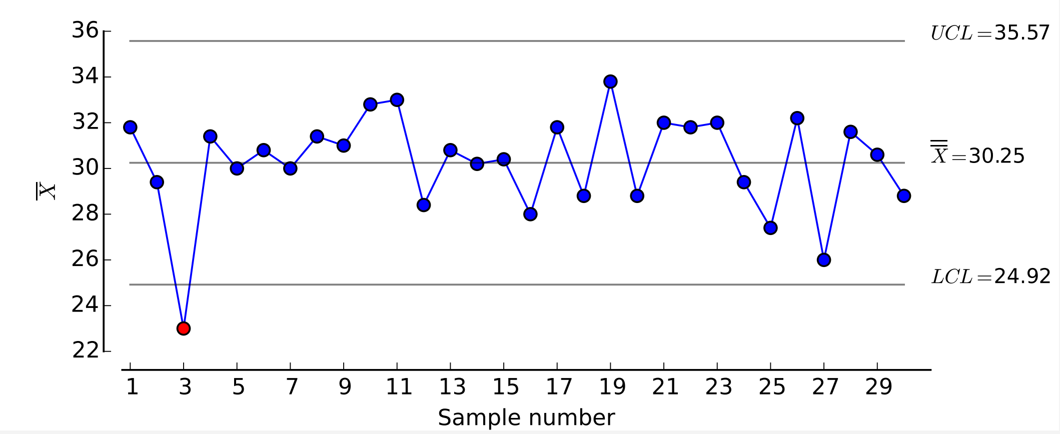


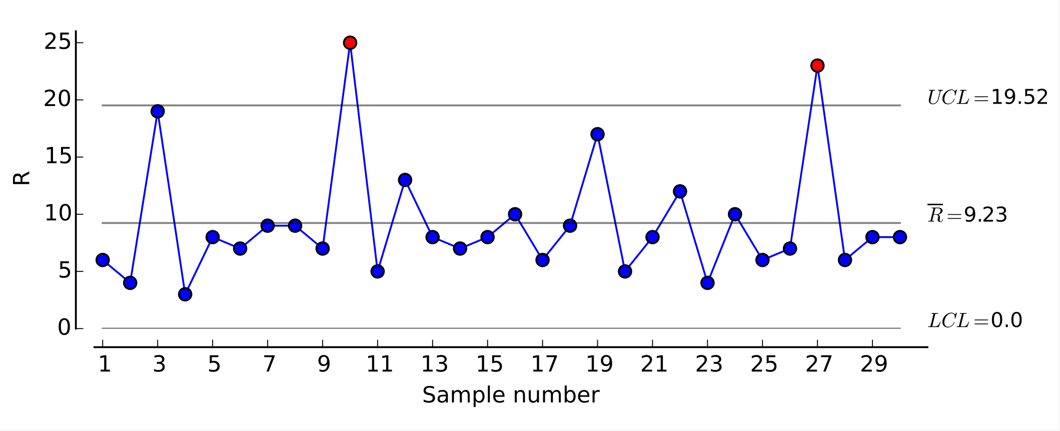
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1									X chart				R chart				Parameters	
2	Sample	x1	x2	x3	x4	x5	Xbar	R	UCL	CL	LCL		UCL	CL	LCL		A2	0.577
3	1	28	32	34	32	33	31.8	6	35.57	30.25	24.92		19.52	9.23	0		D3	0
4	2	31	27	31	28	30	29.4	4	35.57	30.25	24.92		19.52	9.23	0		D4	2.114
5	3	15	31	20	34	15	23	19	35.57	30.25	24.92		19.52	9.23	0		xbarbar	30.25
6	4																Rbar	9.233
7	5																	
8	6																	
9	7																	
10	8																	
11	9																	
12	10																	
13	11																	
14	12																	
15	13	27	30	31	31	35	30.8	8	35.57	30.25	24.92		19.52	9.23	0			
16	14																	
17	15																	
18	16																	
19	17																	
20	18																	
21	19																	
22	20																	
23	21																	

We reproduced the same charts in Python by running the script in How-To 2.4. The resulting charts are displayed in Figure 2.7.

Figure 2.7: $Xbar$-R control charts produced in Python using the data in Table 2.2

(a) $Xbar$ chart

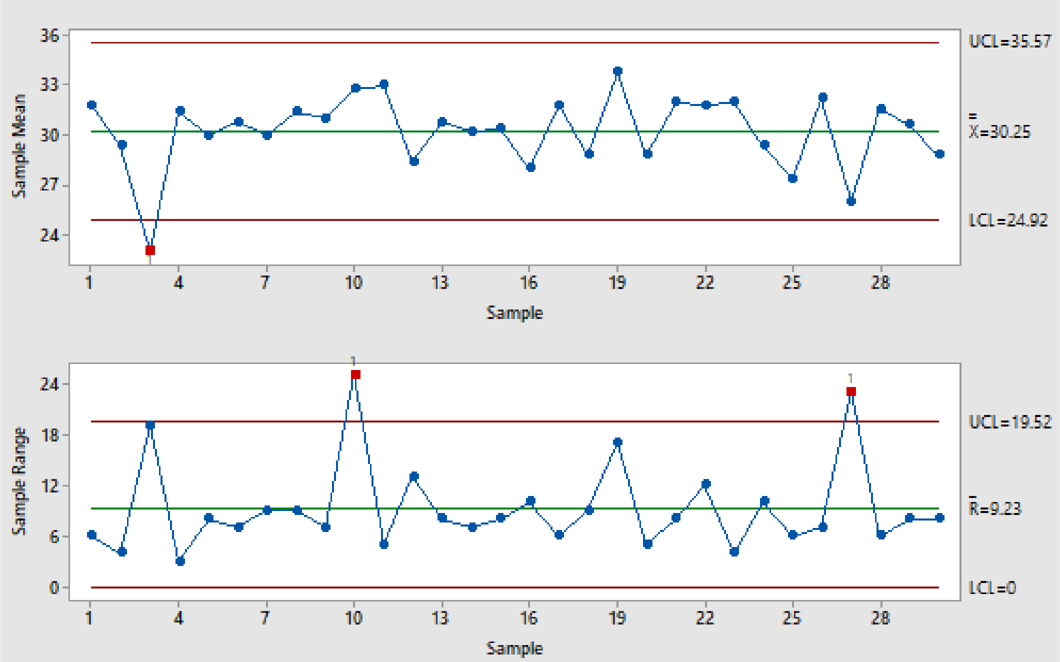




(b) R chart

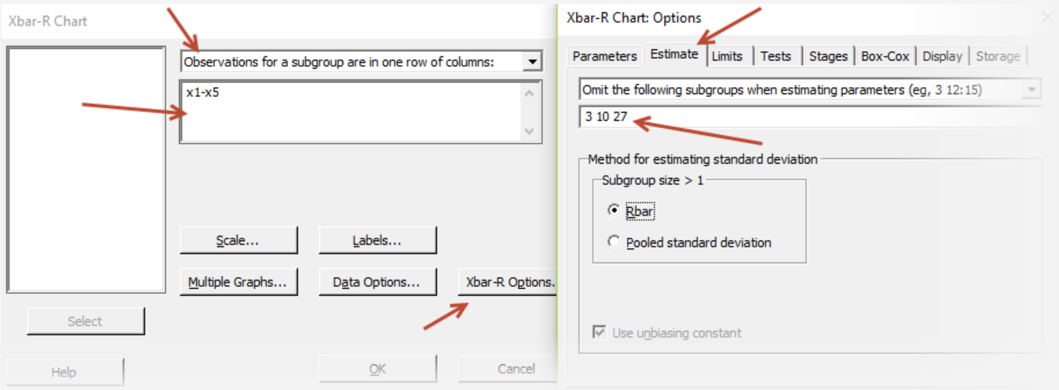
We also recreated similar charts in Minitab by following the instructions in How-To 2.5. The charts that we obtained are illustrated in Figure 2.7.

Figure 2.7: $Xbar$-R control charts in Minitab 18 from the data in Table 2.2



By examining any of these sets of $Xbar$-R control charts, we conclude that the process is unstable in both the $Xbar$ and R charts. The manager needs to investigate samples 3, 10, and 27 and remove any sources of variations. Once the manager has found and fixed all issues, these samples should be omitted and control charts revised. If new out-of-control behaviors arise, the manager should repeat the same process until the process is stable. For demonstration purposes, we used Minitab to omit samples 3, 10, and 27, as shown in Figure 2.8.

Figure 2.8: Omitting out-of-control samples from the $Xbar$-R chart in Figure 2.7



Xbar-R Chart

Observations for a subgroup are in one row of columns:

x1-x5

Xbar-R Chart: Options

Omit the following subgroups when estimating parameters (eg, 3 12:15)

3 10 27

Method for estimating standard deviation

Subgroup size > 1

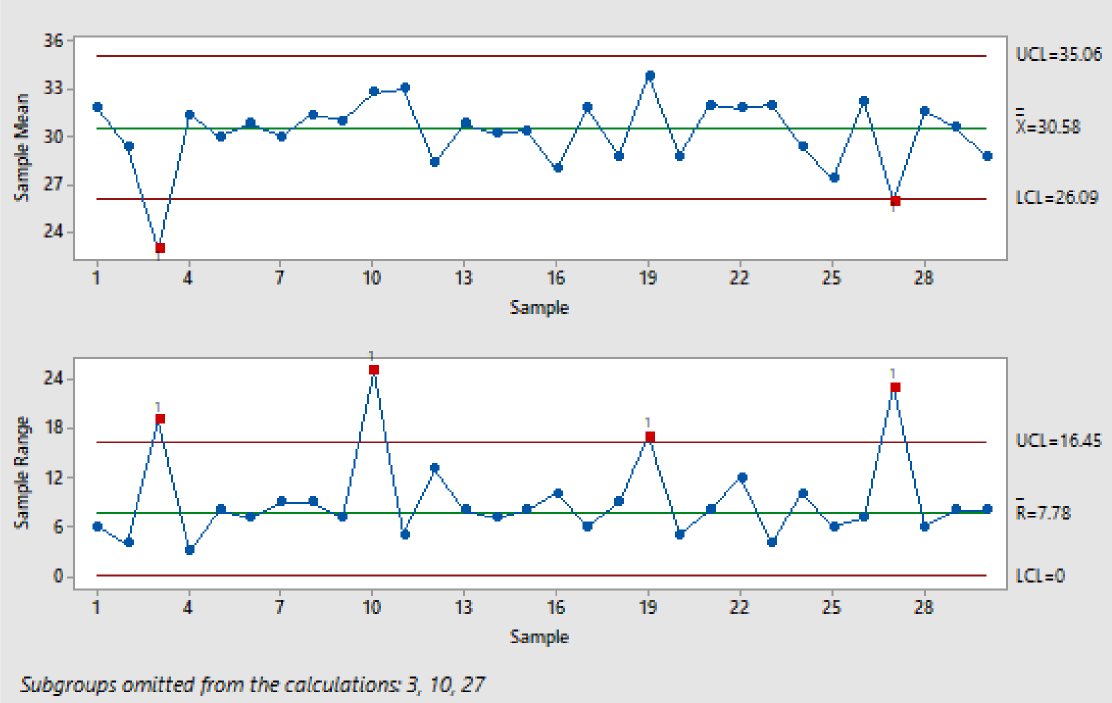
● Rbar

○ Pooled standard deviation

☑ Use unbiasing constant

The revised charts are portrayed in Figure 2.9. The default settings in Minitab preserve the omitted points on the charts, but the centerline and control limits are adjusted.

Figure 2.9: Revised $Xbar$-R control charts in Minitab 18, after omitting samples 3, 10, and 27



Subgroups omitted from the calculations: 3, 10, 27

2.2.4 *XbarS* control charts

We use $XbarS$ charts, also noted as $Xbar$-S, to monitor the mean behavior using the $Xbar$ chart and the variability using the S chart. We typically utilize these charts when the sample size $n > 10$.

Formulation

Box 2.3 summarizes the formulas for $Xbar$-S charts.

Box 2.3 The formulas for $XbarS$ control charts when $L = 3$

When μ **and** σ **standards are given**, we apply these formulas:

	$Xbar$ **chart**	S **chart**
UCL	$\mu + A\sigma$	$B_6\sigma$
CL	μ	$c_4\sigma$
LCL	$\mu - A\sigma$	$B_5\sigma$

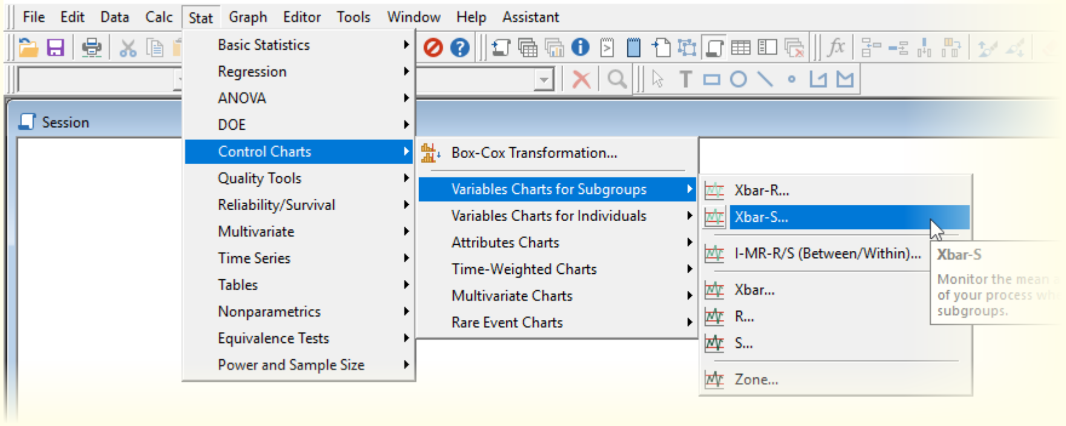
When μ **and** σ **standards are not given**, we implement the following formulas:

	$Xbar$ **chart**	S **chart**
UCL	$\bar{\bar{x}} + A_3\bar{s}$	$B_4\bar{s}$
CL	$\bar{\bar{x}}$	$\bar{s}$
LCL	$\bar{\bar{x}} - A_3\bar{s}$	$B_3\bar{s}$

In the $Xbar$ chart, we monitor $\bar{x}_i$, the average mean in each sample i, for $i : 1, \ldots, m$, where m is the total number of samples. In the S chart, we monitor s_i, the standard deviation in each sample i. We obtain the values of A, A_3, B_3, B_4, B_5, B_6, and c_4 from Appendix Table 12 by the constant sample size n.

How-To 2.7 (XbarS charts in Minitab 18) To construct $XbarS$ charts in Minitab 18, click on *Stat* > *Control Charts* > *Variable Control Charts for Subgroups* > *Xbar-S* > *In the drop-down menu, select* ***Observations for a subgroup are in one row of columns*** >*Select your data* >. See the snapshot in Figure 2.10.

Figure 2.10: Options for $Xbar$-S control charts in Minitab 18



How-To 2.8 (Python 3.6)

Script 2.3: A script for creating $Xbar$-S charts in Python 3.6

```
#VARIABLE CHARTS- XbarS charts
#Import modules
from pandas import*
from pylab import*
from numpy import*
import seaborn as sns
#import data from an excel spreadsheet
data = read_excel('your directory')
#if column names in Excel, select the column of interest (e.g., data
    = data['column name']
xr =  [mean(data.loc[i]) for i in range(len(data))]
sd =  [std(data.loc[i], ddof = 1)*1. for i in range(len(data))]
xbar = mean(xr)
sdbar = mean(sd)
t = arange(len(data))
#parameters fro Appendix Table 1. Next is an example of values when n
     = 13
A3 = 0.850
B3 = 0.382
B4 = 1.618
#Xbar chart
#control limits
UCL = [xbar + A3*sdbar]*len(xr)
LCL = [max(0,xbar - A3*sdbar)]*len(xr)
CL = [xbar]*len(xr)
#mark red a point that falls outside of the control limits. Otherwise
    , mark the point blue.
markers = []
colors = []
for i in range (len(xr)):
    x1 = xr[i]
    x2 = UCL[i]
    x3 = LCL[i]
    if (x1 > x2 or  x1<x3) :
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
#plotting Xbar
fig=figure()
ax1 =  fig.add_subplot(111)
ax1.plot(UCL, 'k-', alpha = 0.5)
ax1.plot(LCL, 'k-',alpha = 0.5)
ax1.plot(CL, 'k-',alpha = 0.5)
ax1.plot(xr,'b-',zorder=1)
for x,y,c,m in zip(t, xr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel('Sample number')
ax1.set_ylabel(r'$\overline{X}$')
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL=$'+str(round(UCL[-1],2)), xy = (xlim()[1], list(
    UCL)[-1]), xytext = (xlim()[1],list(UCL)[-1]),fontsize = 11)
ax1.annotate (r'$\overline{\overline{X}}=$'+str(round(CL[0],2)), xy =
     (xlim()[1], list(CL)[-1]), xytext = (xlim()[1],list(CL)[-1]),
    fontsize = 11)
ax1.annotate ('$LCL=$'+str(round(LCL[-1],2)), xy = (xlim()[1], list(
    LCL)[-1]), xytext = (xlim()[1],list(LCL)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(data), step = 2), arange(1, len(data)+1, step = 2))
show()
#########################
#S chart
#control limits
UCL = [sdbar*B4]*len(xr)
LCL = [max(0,sdbar*B3)]*len(xr)
CL = [sdbar]*len(xr)
#mark red a point that falls outside of the control limits. Otherwise
    , mark the point blue
markers = []
colors = []
for i in range (len(xr)):
    x1 = sd[i]
    x2 = UCL[i]
    x3 = LCL[i]
    if (x1 > x2 or  x1<x3) :
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
#plotting S chart
fig=figure()
ax1 =  fig.add_subplot(111)
t = arange(len(data))
ax1.plot(UCL, 'k-', alpha = 0.5)
ax1.plot(LCL, 'k-',alpha = 0.5)
ax1.plot(CL, 'k-',alpha = 0.5)
ax1.plot(sd,'b-',zorder=1)
for x,y,c,m in zip(t, sd, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#pabel y-axis and x-axis
ax1.set_xlabel('Sample number')
ax1.set_ylabel(r'$\overline{S}$')
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL=$'+str(round(UCL[-1],2)), xy = (xlim()[1], list(
    UCL)[-1]), xytext = (xlim()[1],list(UCL)[-1]),fontsize = 11)
```

```
ax1.annotate (r'$\overline{S}=$'+str(round(CL[0],2)), xy = (xlim()
    [1], list(CL)[-1]), xytext = (xlim()[1],list(CL)[-1]),fontsize =
    11)
ax1.annotate ('$LCL=$'+str(round(LCL[-1],2)), xy = (xlim()[1], list(
    LCL)[-1]), xytext = (xlim()[1],list(LCL)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(data), step = 2), arange(1, len(data)+1, step = 2))
show()
```

How-To 2.9 ($Xbar$-S charts in Excel 2013)
Excel does not have a built-in option to create $Xbar$-S charts, but we can manually program the formulas in Box 2.3 as we demonstrate in Example 2.3.

Example 2.3 (XbarS charts)
You are an intern at Metropolis Hospital, and you are working on a project to stabilize the process of complete blood count (CBC) orders in the ED. The goal is to reduce turn-around-time (TAT). Since the process on the weekends tends to be different than the process on the weekdays, you have decided to start with orders placed on the weekdays. Table 2.3 shows the random samples that you have collected so far. Given that the sample size $n > 10$, you have decided to use the $Xbar$-S control charts to monitor the process. To get started, you looked up parameters A_3, B_3, and B_4 in Appendix Table 12 when $n = 13$ and obtained 0.850, 0.382, 1.618, respectively.
To create $Xbar$-S control charts from this dataset, first set up your spreadsheet in Excel, as shown in Figure 2.11, then proceed with these steps:

Step 1: Calculate the mean in each sample per Equation 2.1. For example, calculate the $Xbar$ in cell O3 using $= AVERAGE(B3 : N3)$. You can drag down this formula to produce the rest of the $Xbar$ values. Additionally, calculate the $Xbarbar$ in cell Z5, per Equation 2.2, using $= AVERAGE(O3 : O32)$.

Step 2: Next, calculate the sample standard deviation per Equation 2.4. For example, the S value in cell P3 was obtained using $= STDEV.S(B3 : N3)$. You can drag down this formula to populate the rest of the S values. To obtain the $Sbar$, in cell R6, use $= AVERAGE(P3 : P32)$.

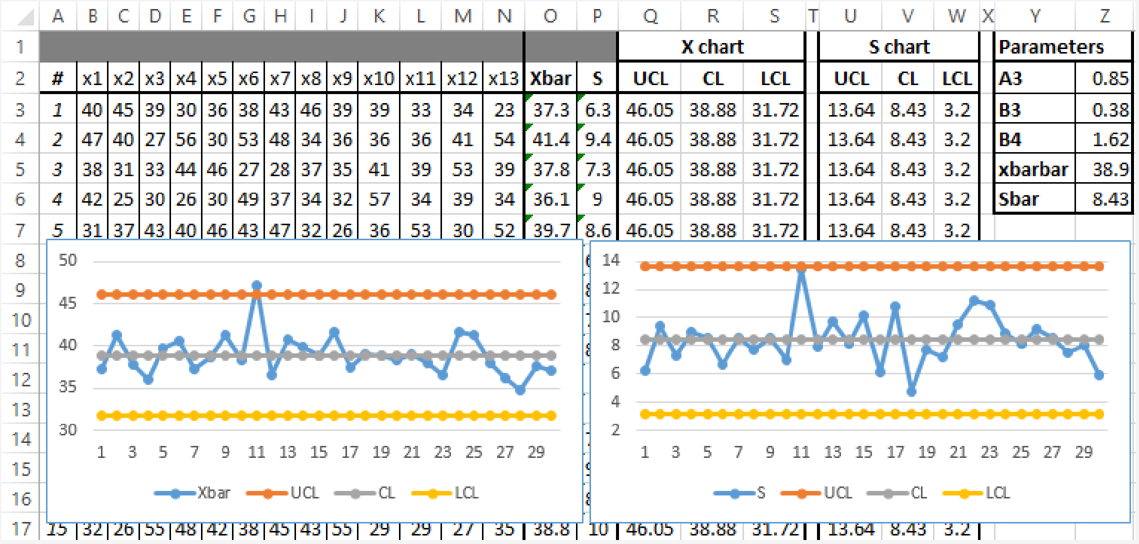
Step 3: To construct control limits, apply the formulas in Box 2.3 when no standards are given. For example, the UCL value in cell U3 was obtained using $= ROUND(\$Z\$4 * \$Z\$6, 2)$, and the LCL value in cell W3 was obtained by $= ROUND(\$Z\$3 * \$Z\$6, 2)$. The CL value in cell V3 was determined as $= \$Z\$6$.

Step 4: In this final step, insert the line charts of columns O and Q-S to create the $Xbar$ chart. Also, insert lines charts of columns P and U-W to create the S chart.

Table 2.3: TAT of CBC orders in the ED, weekdays, Metropolis Hospital, 2018

Sample#	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}	x_{11}	x_{12}	x_{13}
1	40	45	39	30	36	38	43	46	39	39	33	34	23
2	47	40	27	56	30	53	48	34	36	36	36	41	54
3	38	31	33	44	46	27	28	37	35	41	39	53	39
4	42	25	30	26	30	49	37	34	32	57	34	39	34
5	31	37	43	40	46	43	47	32	26	36	53	30	52
6	39	47	45	46	32	43	36	52	29	34	38	42	45
7	30	35	44	42	56	27	38	41	36	44	38	24	30
8	27	34	29	46	39	38	35	45	43	47	35	53	32
9	40	47	59	45	40	38	37	33	35	25	49	49	39
10	49	46	33	28	34	41	29	41	44	47	37	37	31
11	24	43	42	41	38	50	39	60	58	48	80	44	45
12	31	33	48	37	42	28	41	42	45	26	30	26	46
13	34	41	24	42	41	50	39	36	57	26	54	46	39
14	27	44	51	35	41	58	35	33	32	39	40	41	43
15	32	26	55	48	42	38	45	43	55	29	29	27	35
16	39	36	50	46	43	39	34	54	42	42	38	46	33
17	31	33	57	25	28	27	25	49	45	33	39	44	51
18	33	41	34	41	49	40	37	43	40	37	31	42	39
19	46	40	45	33	29	36	41	58	37	29	35	39	37
20	38	52	44	51	42	33	29	34	39	32	32	35	36
21	30	23	48	59	37	41	52	38	34	34	39	39	33
22	29	26	27	54	47	29	39	39	21	36	42	56	48
23	34	30	55	40	36	40	47	30	23	26	29	27	57
24	56	31	45	30	45	44	41	27	45	55	35	43	45
25	47	39	50	39	30	33	35	37	59	35	42	50	40
26	31	50	31	52	42	49	35	29	26	49	33	33	33
27	44	57	38	46	27	36	33	28	32	34	28	38	30
28	28	31	44	44	25	32	36	36	40	40	20	34	43
29	45	36	29	42	47	34	34	40	42	51	22	30	36
30	40	31	29	41	40	31	38	37	30	41	33	42	49

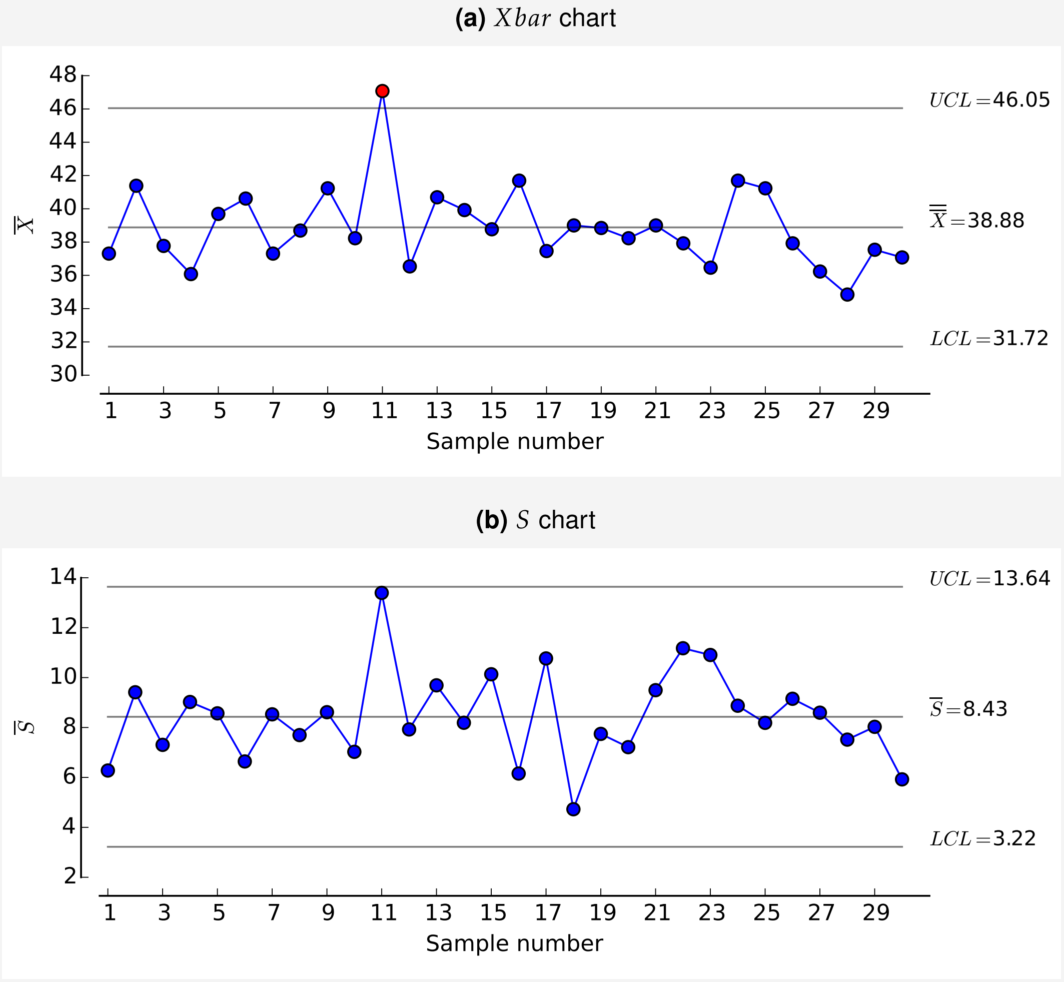
Figure 2.11: An Excel setup for $Xbar$-S control charts using the data in Table 2.2



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1																		X chart				S chart			Parameters	
2	#	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	Xbar	S	UCL	CL	LCL		UCL	CL	LCL		A3	0.85
3	*1*	40	45	39	30	36	38	43	46	39	39	33	34	23	37.3	6.3	46.05	38.88	31.72		13.64	8.43	3.2		B3	0.38
4	*2*	47	40	27	56	30	53	48	34	36	36	36	41	54	41.4	9.4	46.05	38.88	31.72		13.64	8.43	3.2		B4	1.62
5	*3*	38	31	33	44	46	27	28	37	35	41	39	53	39	37.8	7.3	46.05	38.88	31.72		13.64	8.43	3.2		xbarbar	38.9
6	*4*	42	25	30	26	30	49	37	34	32	57	34	39	34	36.1	9	46.05	38.88	31.72		13.64	8.43	3.2		Sbar	8.43
7	*5*	31	37	43	40	46	43	47	32	26	36	53	30	52	39.7	8.6	46.05	38.88	31.72		13.64	8.43	3.2			
8																										
9																										
10																										
11																										
12																										
13																										
14																										
15																										
16																										
17	15	32	26	55	48	42	38	45	43	55	29	29	27	35	38.8	10	46.05	38.88	31.72		13.64	8.43	3.2			

To create the same charts in Python, run the script in How-To 2.8. Your charts may look as illustrated in Figure 2.12.

Figure 2.12: $XbarS$ control charts produced in Python using the data in Table 2.3

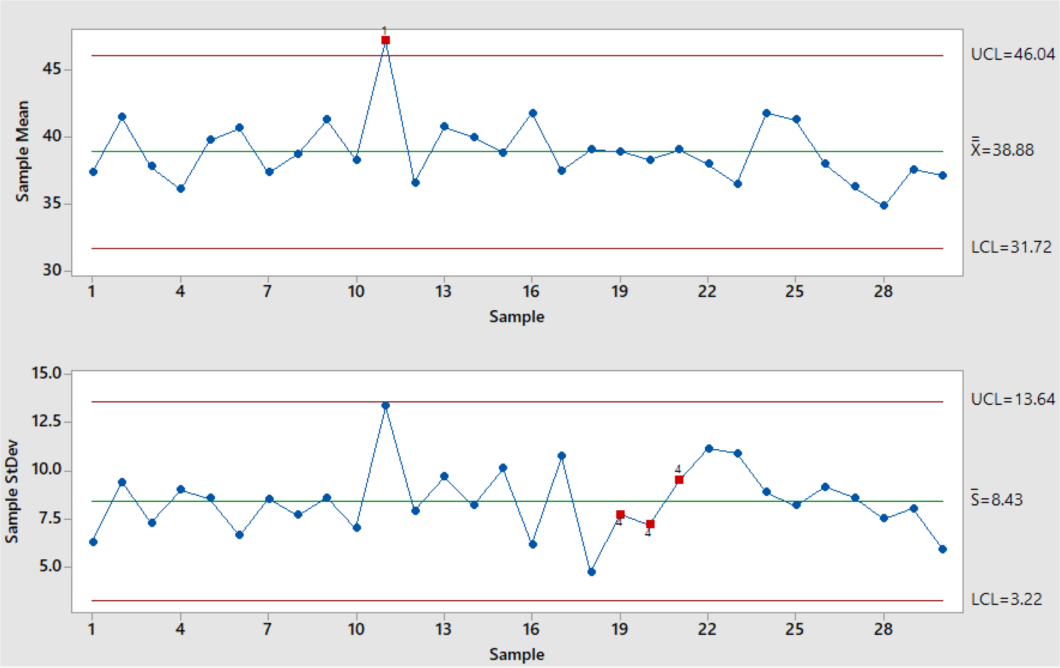


(a) $Xbar$ chart

(b) S chart

From these charts, we appreciate that sample 11 is out-of-control in the $Xbar$ chart. The S chart shows no points outside of the limits, but some unusual alternating behaviors are noticeable. After applying sensitizing rules in Minitab 18, it becomes apparent that the S chart fails test 4 at points 19, 20, 21 (see Figure 2.13). Test 4 in Minitab 18 indicates that **14 points in a row are alternating up and down**. To run sensitizing rules for this case, click on *Chart Options* > *Tests* and select all the sensitizing rules of interest (Review the instructions in How-To 1.30).

Figure 2.13: $Xbar$-S control charts produced in Minitab 18 using the data in Table 2.2



Remark (s^2 control chart): *Besides the unbiased estimation methods of the standard deviation, we can also monitor the process variability using the variance. To accomplish that, we use the χ^2 distribution and set up control limits as follows:*

$$UCL = \frac{\bar{s}^2}{n-1}\chi^2_{\alpha/2,n-1} \tag{2.14}$$

$$CL = \bar{s}^2 \tag{2.15}$$

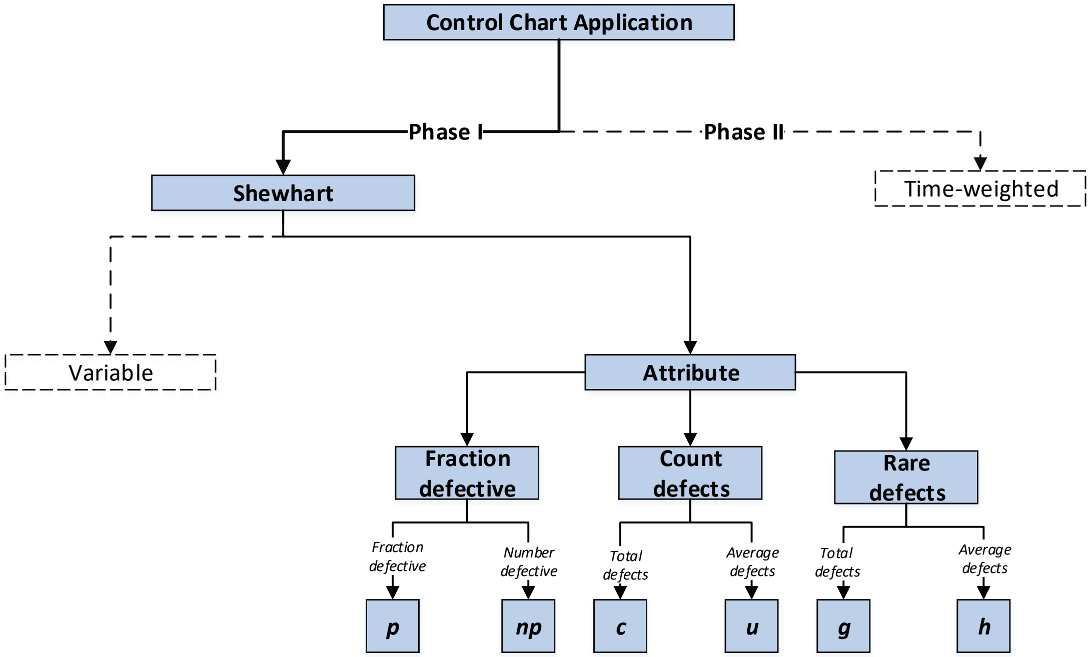
$$UCL = \frac{\bar{s}^2}{n-1}\chi^2_{1-(\alpha/2),n-1} \tag{2.16}$$

where $\bar{s}^2$ is the mean of all sample variances and χ^2 has the degree of freedom of $n-1$ at the significance level α. If the standard deviation is given, we replace $\bar{s}^2$ with σ^2. This type of control chart is known as ***s^2 control chart*** *[44].*

2.3 Attribute control charts

We recall that we employ attribute charts to monitor processes that generate independent discrete data. The attribute charts that we consider here are p, np, c, u, g, and h. If the interest is to monitor the **fraction of defective units**, we apply the $\boldsymbol{p}$ chart. If the interest is to monitor the **number of defective units**, we use the $\boldsymbol{np}$ charts. The p and np charts portray the same information, but some users tend to prefer the np charts since they portray whole units instead of fractions. If the interest is to monitor the **number of defects** per sample, we use c charts. We apply u charts to monitor the **average number of defects** per unit. Additionally, u charts allow us to monitor weighted defects using a **demerit system**. For processes with **rare defects**, we use g charts to monitor the **number of events** between successive defects. The h charts permit us to monitor the **average number of events** between successive defects. Instead of monitoring rare events, we could instead monitor the time between defects. To do that, we may have to transform the data into normal distribution using methods such as **Nelson's transformation**. Then, we can apply traditional variable charts to monitor the process. A map summarizing attribute charts is illustrated in Figure 2.14.

Figure 2.14: A basic map of attribute charts



As compared to variable charts, attribute charts tend to be more prevalent in health care given that most of the data generated there are discrete [44]. The following scenarios describe possible health care processes that could be monitored using attribute charts.

Scenario 1: The manager of Labor and Delivery at Metropolis Hospital is interested in using p and np charts to monitor the rate of cesarean-section (c-section) at the hospital. Each month, a retrospective review will be conducted, and the rate of c-section will be determined by dividing the total number of c-section by the total number of deliveries. A detailed scenario discussing the application of p charts to monitor the rate of primary c-section is presented in Carey and Lloyd (1995) [10].

Scenario 2: The manager of a local surgery center will use p charts to monitor the adverse outcomes of thyroid surgery. More specifically, the manager will track the fraction of thyroid surgical patients who experience a recurrence of either laryngeal nerve palsy or hypocalcemia. Each month the manager will perform an audit to determine the rate of interest. Additionally, the manager will adjust for patient risk factors and create a separate risk-adjusted p chart. See Ross (2013) [53] for more discussion about the use of p charts to monitor the outcomes of thyroid surgery.

Scenario 3: The use of restraints and seclusions on patients is a practice that is discouraged due to the potential of psychological and physical traumas [27, 55]. The chairperson of the quality assurance committee of the psychiatric unit at Metropolis Hospital would like to use c charts to monitor the daily use rate of restraints and seclusions. The chairperson will also use u charts to monitor the use rate per patient. A case study discussing the application of control charts to monitor restraints in psychiatric units is presented in Carey and Lloyd (1995) [10].

Scenario 4: On rare occasions, the use of information technology (IT) in clinical settings has led to serious patient harm. Some of the incidents that have been reported include patient harm from wrong medications owing to the computerized provider order entry (CPOE) system that failed to display alerts about drug interactions [15, 35]. In one serious incident, a hospital's picture archiving and communication system (PACS) showed a wrong x-ray image, which resulted in patient death [35]. The CIO of Metropolis Hospital is working with the Chief Medical Officer (CMO) to establish a system for tracking IT technical issues that result in severe patient harm or near misses. Given the rarity of these events, g and h charts will be used for monitoring the process.

Scenario 5: Document Imaging Technology allows for patient paper records to be scanned and converted into digital images. The typical process of scanning patient records includes the steps of removing staples, repairing torn papers, and the verification of barcodes on each page to ensure the correct indexing of documents into the patient's electronic health record (EHR) [25, 48]. While errors due to barcoding technology are infrequent (about 3 transactions in a million [48]), they do occasionally occur nonetheless, and one could potentially end up with a patient record with information from another patient. The manager of health informatics at Metropolis Hospital would like to use Nelson's transformation and ImR charts to monitor the number of days between barcode errors.

2.3.1 p and np charts

The statistics for a p chart

A p chart is based on the Bernoulli random variable given by:

$$p(x) = \begin{cases} p & x = 1 \\ 1 - p & x = 0 \end{cases} \tag{2.17}$$

where p is the probability of an event under study such as that of a unit being defective. If a randomly selected unit is defective, $x = 1$, otherwise $x = 0$. The mean μ and the standard deviation σ of the **Bernoulli random variable** are given by:

$$\mu = p \tag{2.18}$$

$$\sigma = \sqrt{p(1-p)} \tag{2.19}$$

Since we generally don't know p, we estimate it from the process samples via the method of the maximum likelihood estimator(MLE). Let's suppose that we took m number of samples. Each sample i has a constant size n and D_i is the number of defective units in this sample. The MLE estimator of p is given by $\bar{p}$ that we calculate as follows:

$$\bar{p} = \frac{\sum_{i=1}^{m} D_i}{mn} \tag{2.20}$$

When the sample size is variable, we obtain $\bar{p}$ like this:

$$\bar{p} = \frac{\sum_{i=1}^{m} D_i}{\sum_{i=1}^{m} n_i} \tag{2.21}$$

The standard deviation of the $\bar{p}$ statistic follows from the central limit theorem and is given by:

$$\sigma_{\bar{p}} = \sqrt{\frac{\bar{p}(1-\bar{p})}{n}} \tag{2.22}$$

The statistics for an np chart

An np chart is based on the binomial random variable given by:

$$P\{D = x\} = \binom{n}{x} p^x (1-p)^{n-x} \qquad x = 0, 1, 2, \ldots, n \tag{2.23}$$

where D symbolizes the number of defective units. Here, $P(D = x)$ characterizes the probability of observing x number of defective units in a sample of size n. The mean and standard deviation of the binomial random variable are given by:

$$\mu = np \tag{2.24}$$

$$\sigma = \sqrt{np(1-p)} \tag{2.25}$$

where p, if not given, is estimated by $\bar{p}$ per Equation 2.20 or 2.21.

Formulation

Box 2.4 summarizes the formulas for creating p and np charts.

Box 2.4 The formulas for p and np control charts when $L = 3$

When the **standard** of p is given, we apply these formulas:

	p **chart**	np **chart**
UCL	$p + 3\sqrt{\frac{p(1-p)}{n}}$	$np + 3\sqrt{np(1-p)}$
CL	p	np
LCL	$p - 3\sqrt{\frac{p(1-p)}{n}}$	$np - 3\sqrt{np(1-p)}$

When **no standard** of p is given, we apply the following formulas:

	p **chart**	np **chart**
UCL	$\bar{p} + 3\sqrt{\frac{\bar{p}(1-\bar{p})}{n}}$	$n\bar{p} + 3\sqrt{n\bar{p}(1-\bar{p})}$
CL	$\bar{p}$	$n\bar{p}$
LCL	$\bar{p} - 3\sqrt{\frac{\bar{p}(1-\bar{p})}{n}}$	$n\bar{p} - 3\sqrt{n\bar{p}(1-\bar{p})}$

In the np chart we monitor the number of defective units D_i, for $i : 1, \ldots, m$, where m is the total number of samples. In the p chart, we monitor the fraction of defective units $\hat{p}_i = D_i/n$ when n is constant. When n is variable, we replace n by n_i, which creates variable control limits. Alternatively, we could monitor the process using the standardized values given by:

$$Z_i = \frac{\hat{p}_i - p}{\sqrt{\frac{p(1-p)}{n_i}}} \qquad i = 1, 2, \ldots, m \tag{2.26}$$

with $UCL = 3$, $CL = 0$, and $LCL = -3$. If not given, p is estimated with $\bar{p}$ [44].

Remark (LCL $>$ 0): *It is common practice to set the LCL value x to $\max(0, x)$ since negative defective units don't make sense. This approach generally works well, but it does not allow for the capture information about when the process is doing well (e.g., small numbers of defective units) since all we will observe is LCL = 0. To remedy this problem, we choose the sample size n so that*

LCL is always positive. To do that, we proceed as follows:

$$\bar{p} - L\sqrt{\frac{\bar{p}(1-\bar{p})}{n}} = 0 \tag{2.27}$$

$$-L\sqrt{\frac{\bar{p}(1-\bar{p})}{n}} = -\bar{p} \tag{2.28}$$

$$\left(-L\sqrt{\frac{\bar{p}(1-\bar{p})}{n}}\right)^2 = (-\bar{p})^2 \tag{2.29}$$

$$\frac{L^2\bar{p}(1-\bar{p})}{n} = \bar{p}^2 \tag{2.30}$$

$$\frac{L^2\bar{p}(1-\bar{p})}{\bar{p}^2} = n \tag{2.31}$$

$$\frac{L^2(1-\bar{p})}{\bar{p}} = n \tag{2.32}$$

So, to ensure that $LCL > 0$, we must choose n such that:

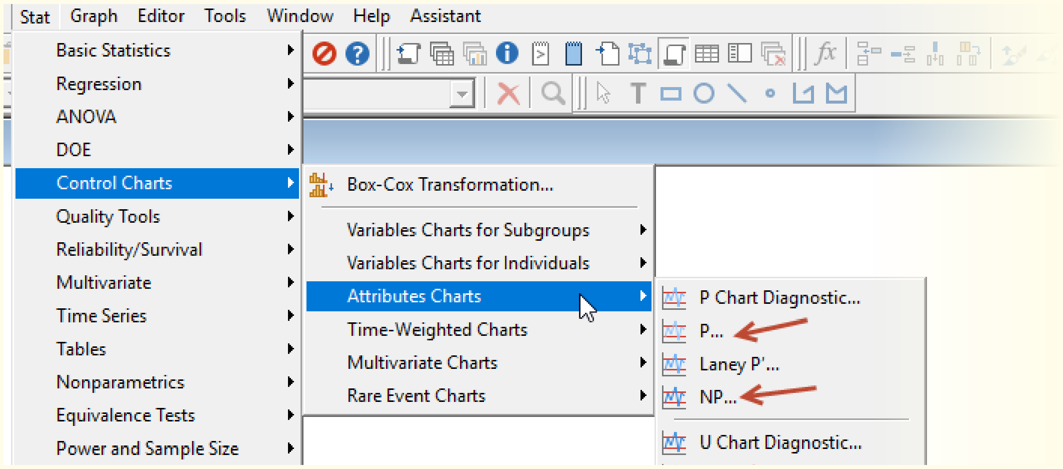
$$n \geq \left\lceil \frac{L^2(1-\bar{p})}{\bar{p}} \right\rceil \tag{2.33}$$

where $\lceil . \rceil$ is a ceiling function. In this book, the default value of L is 3.

How-To 2.10 (p and np charts in Minitab 18)

To construct p and np charts in Minitab 18, click on *Stat > Control Charts > Attribute Charts > select the appropriate chart > Select your data > Input the sample size or the column of sample sizes>OK*. See the snapshot in Figure 2.15.

Figure 2.15: Options for *p* and *np* control charts in Minitab 18



How-To 2.11 (Python 3.6)

Script 2.4: A script for creating p and Z charts in Python 3.6

```
#ATTRIBUTE CHARTS- P and Z charts
#Import modules
from pandas import*
from pylab import*
from numpy import*
import seaborn as sns
#import data from an excel spreadsheet
#Di is the column of defects and Ni is the column of sample sizes
data = read_excel('your directory')
xr = [data.Di[i]*1./data.Ni[i] for i in xrange(len(data))]#phat
t = arange(len(xr))
pbar = 1.*data.Di.sum()/ data.Ni.sum()
#P chart
#control limits
UCL = [pbar + 3.*sqrt(pbar*(1.-pbar)/data.Ni[i])  for i in xrange(len
    (xr))]
LCL = [max(0.,pbar - 3.*sqrt(pbar*(1.-pbar)/data.Ni[i]))  for i in
    xrange(len(xr))]
CL = [pbar]*(len(xr))
#mark red a point that falls outside of the control limits. Otherwise
    , mark the point blue
markers = []
colors = []
for i in xrange (len(xr)):
    x1 = xr[i]
    if x1 > UCL[i]:
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
#plotting the P chart
fig=figure()
ax1 =  fig.add_subplot(111)
ax1.step(t,UCL, 'k-', alpha = 0.5, where = 'mid')
ax1.step(t, LCL, 'k-',alpha = 0.5,  where = 'mid')
ax1.plot(CL, 'k-',alpha = 0.5)
ax1.plot(xr,'b-',zorder=1)
for x,y,c,m in zip(t, xr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
ylim(ymin = -0.01)
xlim(-0.5, t[-1]+1)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel('Sample number')
ax1.set_ylabel('Fraction nonconforming')
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL=$'+str(round(UCL[0],3)), xy = (xlim()[1], list(
    UCL)[-1]), xytext = (xlim()[1],list(UCL)[-1]),fontsize = 11)
ax1.annotate ('$\overline{P}=$'+str(round(CL[0],3)), xy = (xlim()[1],
     list(CL)[-1]), xytext = (xlim()[1],list(CL)[-1]),fontsize = 11)
ax1.annotate ('$LCL=$'+str(round(LCL[0],3)), xy = (xlim()[1], list(
    LCL)[-1]), xytext = (xlim()[1],list(LCL)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(data), step = 2), arange(1, len(data)+1, step = 2))
show()
#####################################
#Z chart
data['xt'] = data.Di/data.Ni
pbar = 1.*data.Di.sum()/ data.Ni.sum()
xr = [(data.xt[i] - pbar)/sqrt(pbar*(1.-pbar)/data.Ni[i]) for i in
    xrange(len(data))]
t = arange(len(xr))
#control limits
UCL = [3.  for i in xrange(len(xr))]
LCL = [-3 for i in xrange(len(xr))]
CL = [0]*(len(xr))
#mark red a point that falls outside of the control limits. Otherwise
    , mark the point blue
markers = []
colors = []
for i in xrange (len(xr)):
    x1 = xr[i]
    if x1 > UCL[i]:
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
#plotting Z chart
fig=figure()
ax1 =  fig.add_subplot(111)
ax1.step(t,UCL, 'k-', alpha = 0.5, where = 'mid')
ax1.step(t, LCL, 'k-',alpha = 0.5,  where = 'mid')
ax1.plot(CL, 'k-',alpha = 0.5)
ax1.plot(xr,'b-',zorder=1)
for x,y,c,m in zip(t, xr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
ylim(-3.5, 3.5)
xlim(-0.5, t[-1]+1)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel('Sample number')
ax1.set_ylabel('$Z_i$')
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL=$'+str(round(UCL[0],3)), xy = (xlim()[1], list(
    UCL)[-1]), xytext = (xlim()[1],list(UCL)[-1]),fontsize = 11)
ax1.annotate (''+str(round(CL[0],1)), xy = (xlim()[1], list(CL)[-1]),
     xytext = (xlim()[1],list(CL)[-1]),fontsize = 11)
ax1.annotate ('$LCL=$'+str(round(LCL[0],3)), xy = (xlim()[1], list(
    LCL)[-1]), xytext = (xlim()[1],list(LCL)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(data), step = 2), arange(1, len(data)+1, step = 2))
show()
```

How-To 2.12 (p and np charts in Excel 2013)
Excel does not have a built-in option to create p and np charts, but we can manually program the formulas in Box 2.4 as we demonstrate in Example 2.4.

Example 2.4 (p and np charts)
The HIM[a] manager at Metropolis Hospital sits on the revenue cycle steering committee. At the last week's meeting, it was decided to start monitoring billing denials by the department. The HIM manager was charged with monitoring denials due to medical coding errors. Table 2.4 presents monthly denials for the last 30 months. The column of $Denials$ represents the variable sample size of all denials, and the $Coding$ column contains the number of denials due to medical coding errors.

Table 2.4: Denials related to medical coding errors at Metropolis Hospital

Sample#	Denials	Coding
1	36	4
2	39	2
3	22	7
4	44	5
5	22	5
6	45	2
7	33	6
8	21	6
9	37	7
10	40	3
11	41	2
12	27	7
13	34	8
14	23	4
15	27	7
16	38	5
17	20	5
18	49	4
19	39	6
20	46	6
21	45	5
22	37	5
23	42	2
24	45	7
25	30	4
26	38	2
27	24	9
28	27	3
29	32	2
30	48	6

The manager has decided to apply the p and np charts to monitor the process. Using Python, the manager programmed the formulas in Box 2.4, when p is not given, and produced the charts in Figures 2.16 - 2.18.

Figure 2.16: A p chart based on the data in Table 2.4

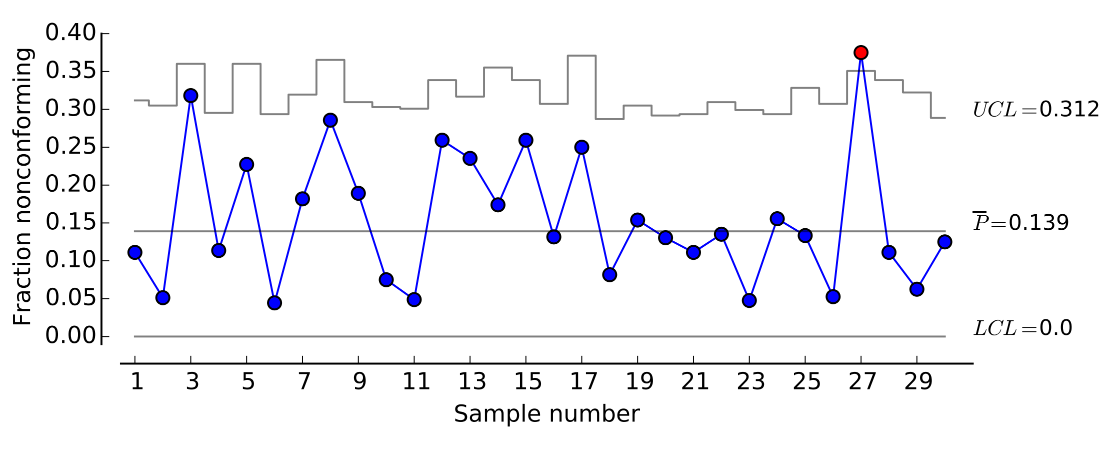


Figure 2.17: An np chart based on the data in Table 2.4

.

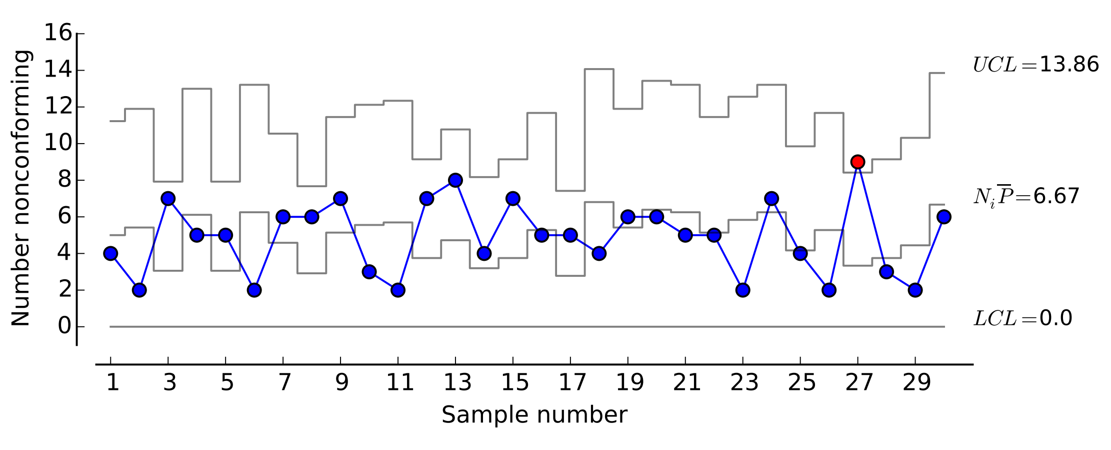
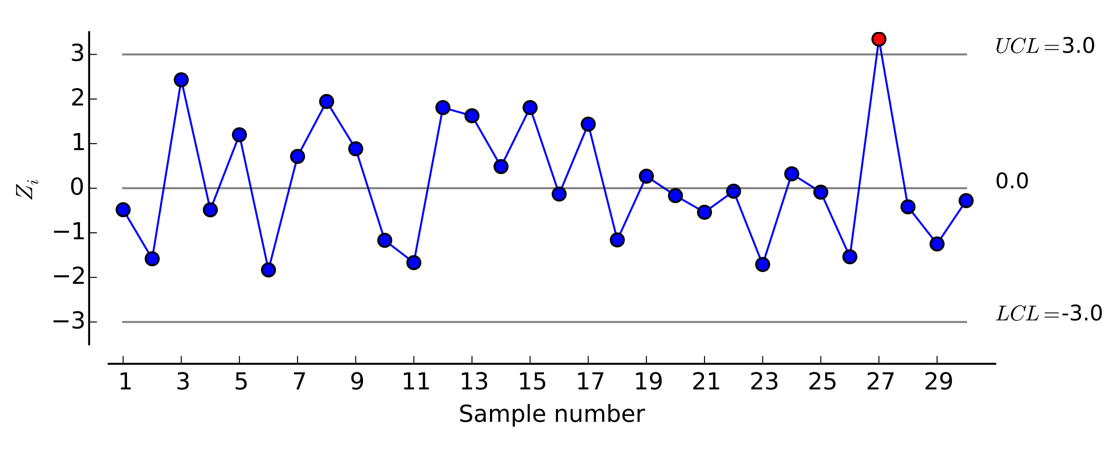


Figure 2.18: A Z chart based on the data in Table 2.4

.

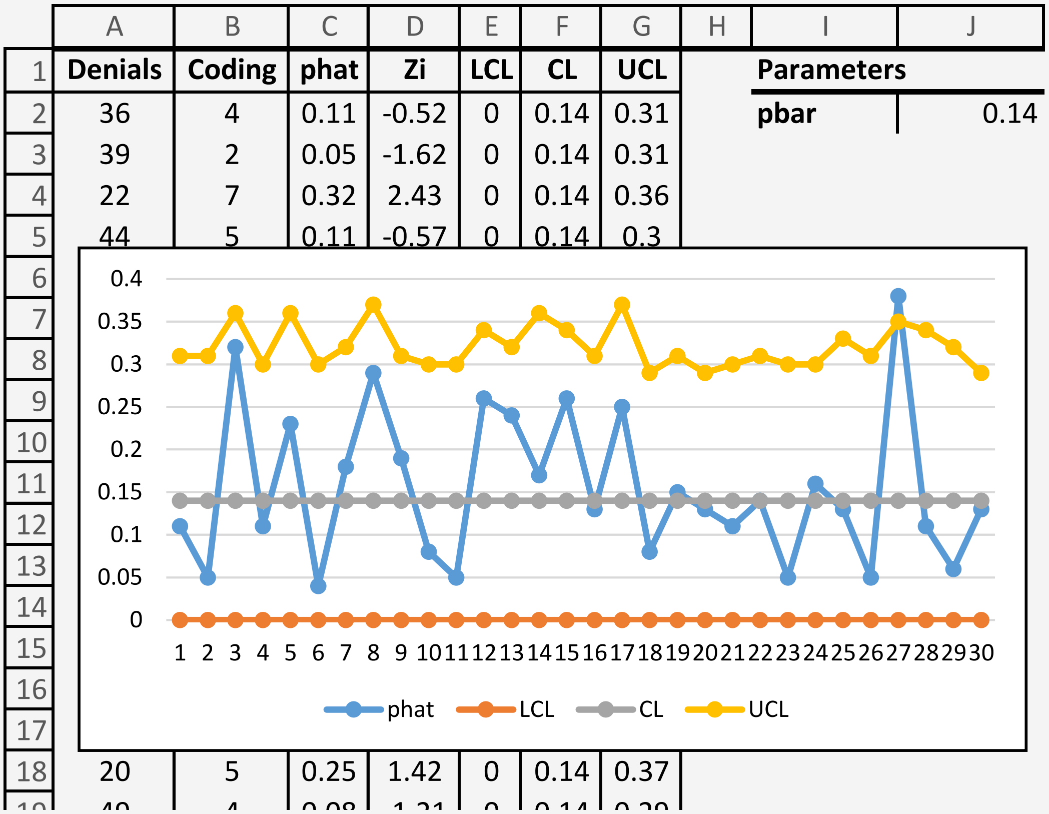


All charts indicate out-of-control behaviors in sample 27. To improve the process, the manager needs to find and fix all the causes of special variation in this sample. Subsequently, sample 27 can be omitted, and a new control chart created to continue to monitor the process. This procedure should be iterated until no more out-of-control behaviors exist in the process.

Next, we demonstrate how you can reproduce the p and Z charts in Excel. The replication of the np chart is left for an exercise.

1. First set up your spreadsheet, as illustrated in Figure 2.19.

Figure 2.19: A setup of Excel to create the p and Z charts based on the data in Table 2.4



	A	B	C	D	E	F	G	H	I	J
1	Denials	Coding	phat	Zi	LCL	CL	UCL		Parameters	
2	36	4	0.11	-0.52	0	0.14	0.31		pbar	0.14
3	39	2	0.05	-1.62	0	0.14	0.31			
4	22	7	0.32	2.43	0	0.14	0.36			
5	44	5	0.11	-0.57	0	0.14	0.3			
6										
7										
8										
9										
10										
11										
12										
13										
14										
15										
16										
17										
18	20	5	0.25	1.42	0	0.14	0.37			

2. Determine the $\bar{p}$ value in cell $J2$ as indicated in Equation 2.21. Proceed this way: $= ROUND(SUM(B2:B31)/SUM(A2:A31), 2)$.

3. Calculate the $\hat{p}_i$ statistic for each month i using this ratio: $Coding/Denials$. For example, the value in cell $C2$ was obtained by $= ROUND(B2/A2, 2)$. You can drag down this formula to populate the rest of the values.

4. Create control limits when no standards are given, as indicated in Box 2.4. For example, the value of LCL in cell $E2$ was calculated using $= MAX(0, \$J\$2 -$

$3 * SQRT(\$J\$2 * (1 - \$J\$2)/A2))$. The value of CL in cell $F2$ was obtained by $= \$J\$2$. The value of UCL in cell $G2$ was determined using $= ROUND(\$J\$2 + 3 * SQRT(\$J\$2 * (1 - \$J\$2)/A2), 2)$.

5. To create the p chart, insert line charts for columns C, E, F, and G.
6. To determine the Z_i statistic for each month i, encode Equation 2.26. For example, the value of Z_i in cell $D2$ was obtained by $= ROUND((C2 - \$J\$2)/SQRT(\$J\$2 * (1 - \$J\$2)/A2), 2)$. Drag down this formula to populate the rest of the values.

To reproduce the same charts in Minitab, follow the instructions in How-To 2.10.

[a] **HIM:** Health Information Management

2.3.2 c and u charts

The statistics for a c chart

We use a c chart to monitor the variable count of defects C in the process. The probability of observing $C = x$ count of defects obeys the Poisson random variable and is expressed like this:

$$p(C = x) = \frac{e^{-c} c^x}{x!} \qquad x = 0, 1, 2, \ldots \tag{2.34}$$

where c, the rate of defects, if not given, can be approximated using the MLE method as follows:

$$c \approx \bar{c} = \frac{\sum_{i=1}^{m} C_i}{m} \tag{2.35}$$

where m is the total number of samples. We recall that the mean and variance of a Poisson random variable are equal, meaning that, for our process, we have:

$$\mu = c \tag{2.36}$$

$$\sigma = \sqrt{c} \tag{2.37}$$

The statistics for a u chart

We use a u chart to monitor the average count of defects per unit. A u chart also obeys the Poisson random variable with a rate $\bar{u}$ that we estimate this way:

$$\bar{u} = \frac{\bar{c}}{n} \tag{2.38}$$

where $\bar{c}$ is obtained per Equation 2.35 and n is the sample size. From the central limit theorem, we approximate the standard deviation of our process this way:

$$\sigma_{\bar{u}} = \sqrt{\frac{\bar{u}}{n}} \equiv \frac{1}{n}\sqrt{\bar{c}} \tag{2.39}$$

Formulation with unweighted defects

Box 2.5 presents the formulas to construct the c and u charts when defects are not weighted. In other words, we assume that defects as have equal severity.

Box 2.5 The formulas for the c and u control charts when $L = 3$ and defects are unweighted

When the **standard rate c** is given, we apply the formulas below:

	***c* chart**	***u* chart**
UCL	$c + 3\sqrt{c}$	$u + 3\sqrt{\frac{u}{n}}$
CL	c	c
LCL	$c - 3\sqrt{c}$	$u - 3\sqrt{\frac{u}{n}}$

where $u = c/n$. When the **standard rate c** is not given, we employ the following formulas:

	***c* chart**	***u* chart**
UCL	$\bar{c} + 3\sqrt{\bar{c}}$	$\bar{u} + 3\sqrt{\frac{\bar{u}}{n}}$
CL	$\bar{c}$	$\bar{u}$
LCL	$\bar{c} - 3\sqrt{\bar{c}}$	$\bar{u} - 3\sqrt{\frac{\bar{u}}{n}}$

In the c chart, we monitor the count of defects C_i, for $i : 1, \ldots, m$, where m is the total number of samples. In the u chart, we monitor the average count of defects given by $u_i = C_i/n$. When the sample size is variable, we replace n by n_i, which creates variable control limits. Alternatively, we can monitor the process using standardized values given by:

$$Z_i = \frac{u_i - \bar{u}}{\sqrt{\frac{\bar{u}}{n_i}}} \tag{2.40}$$

We set $UCL = 3$, $CL = 0$, and $LCL = -3$ [44].

Remark (LCL>0): *Like in the p and np charts, it is common practice to set the LCL value x to* $\max(0, x)$ *since negative defects can't occur. To be able to capture all instances of low defects in* *the process, we choose n so $LCL > 0$ as follows:*

$$\begin{aligned} \bar{u} - L\sqrt{\frac{\bar{u}}{n}} &= 0 && (2.41) \\ -L\sqrt{\frac{\bar{u}}{n}} &= -\bar{u} && (2.42) \\ \left(-L\sqrt{\frac{\bar{u}}{n}}\right)^2 &= (-\bar{u})^2 && (2.43) \\ \frac{L^2\bar{u}}{n} &= \bar{u}^2 && (2.44) \\ \frac{L^2}{\bar{u}} &= n && (2.45) \end{aligned}$$

So, to assure $LCL > 0$, we must choose n that satisfies this inequality:

$$n \geq \left\lceil \frac{L^2}{\bar{u}} \right\rceil \qquad (2.46)$$

where $\lceil . \rceil$, as before, is the ceiling function for rounding up.

Formulation with weighted defects

When defects differ in severity, we group them into classes and assign each class a weight according to some **demerit system**. Box 2.6 presents a demerit system of defects commonly used in the manufacturing sector [44]. This weighing system can also be adopted for typical health care processes.

Box 2.6 A demerit system of defects

Each independent defect is assigned to one of the following classes [44]:

Class A: a very serious defect with a weight of **100**

Class B: a serious defect with a weight of **50**

Class C: a moderately serious defect with a weight of **10**

Class D: a minor defect with a weight of **1**

To monitor a process with weighted defects, we use u charts. The corresponding formulas are summarized in Box 2.6.

Box 2.7 A demerit-based u chart when $L = 3$

We construct a demerit-based u chart as follows [44]:

$$UCL = \bar{u} + 3\hat{\sigma}_u \quad (2.47)$$

$$CL = \bar{u} \quad (2.48)$$

$$LCL = \bar{u} - 3\hat{\sigma}_u \quad (2.49)$$

where

$$\bar{u} = 100\bar{u}_A + 50\bar{u}_B + 10\bar{u}_C + 1\bar{u}_D \quad (2.50)$$

and

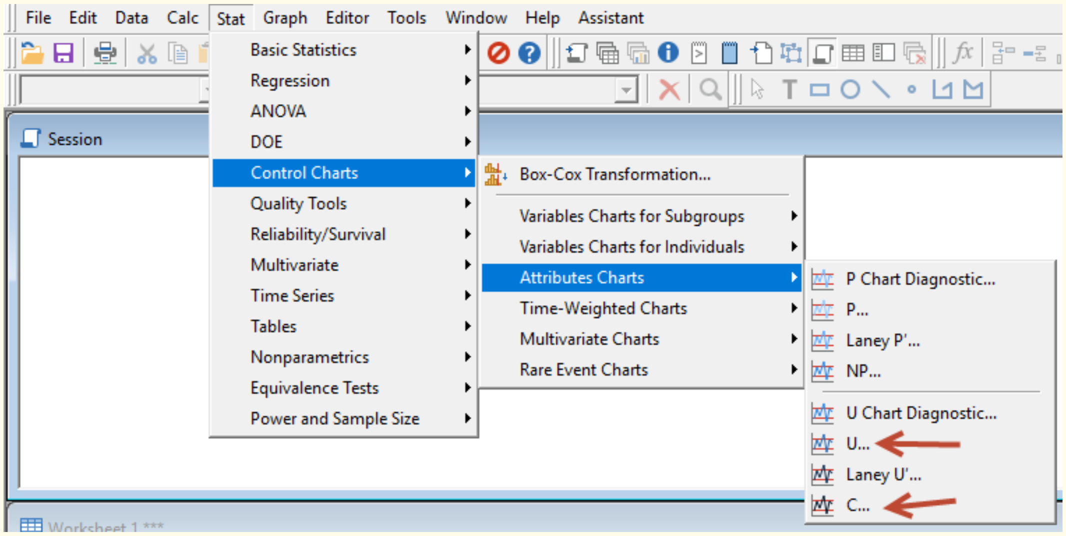
$$\hat{\sigma}_u = \sqrt{\frac{100^2\bar{u}_A + 50^2\bar{u}_B + 10^2\bar{u}_C + 1\bar{u}_D}{n}} \quad (2.51)$$

Here, $\bar{u}_A$, $\bar{u}_B$, $\bar{u}_C$, and $\bar{u}_D$ are the averages of defects in classes A, B, C, and D, respectively. The weighted statistic to be monitored uD_i, for each sample i, is given by:

$$uD_i = 100u_{iA} + 50u_{iB} + 10u_{iC} + 1u_{iD} \qquad i = 1, 2, \ldots, m \quad (2.52)$$

How-To 2.13 (**c and u charts in Minitab 18**)
To create the c and u charts in Minitab, Click on *Stat > Control Charts > Attribute Charts > Select the appropriate chart*. **Note:** there is no option for creating a demerit-based u chart. We will show how to construct this chart using Excel.

Figure 2.20: Options for creating *c* and *u* control charts in Minitab 18



How-To 2.14 (Python 3.6)

Script 2.5: A script for creating c and u charts in Python 3.6

```
#ATTRIBUTE CHARTS C and U
#import modules
from pandas import*
from pylab import*
from numpy import*
import seaborn as sns
#Import data from an excel spreadsheet
#Ci is the column of defects and Ni is the column of sample sizes
data = read_excel('your directory')
#C-CHART
xr = data.Ci
cbar = mean(data.Ci)
t = arange(len(xr))
#control limits
UCL = [cbar + 3.*sqrt(cbar)]*len(xr)
LCL = [cbar - 3.*sqrt(cbar)]*len(xr)
CL = [cbar]*len(xr)
#mark red a point that falls outside of the control limits. Otherwise
    , mark the point blue
markers = []
colors = []
for i in range (len(xr)):
    x1 = xr[i]
    if (x1 > UCL[i] or x1<LCL[i]):
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
#Plotting
fig=figure()
ax1 =  fig.add_subplot(111)
ax1.step(t,UCL, 'k-', alpha = 0.5, where = 'mid')
ax1.step(t, LCL, 'k-',alpha = 0.5,  where = 'mid')
ax1.step(t,CL, 'k-',alpha = 1, where = 'mid')
ax1.plot(xr,'b-',zorder=1)
for x,y,c,m in zip(t, xr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel('Sample number')
ax1.set_ylabel('Number of nonconformities')
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL=$'+str(round(UCL[-1],2)), xy = (xlim()[1], list(
    UCL)[-1]), xytext = (xlim()[1],list(UCL)[-1]),fontsize = 11)
ax1.annotate ('$CL=$'+str(round(CL[-1],2)), xy = (xlim()[1], list(CL)
    [-1]), xytext = (xlim()[1],list(CL)[-1]),fontsize = 11)
ax1.annotate ('$LCL=$'+str(round(LCL[-1],2)), xy = (xlim()[1], list(
    LCL)[-1]), xytext = (xlim()[1],list(LCL)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(data), step = 2), arange(1, len(data)+1, step = 2))
show()
##############################
#U-CHART
xr = data.Ci/data.Ni
t = arange(len(xr))
n = mean(data.Ni)
ubar = cbar/n
#control limits
UCL = [ubar + 3.*sqrt(ubar/n)]*len(xr)
LCL = [ubar - 3.*sqrt(ubar/n)]*len(xr)
CL = [ubar]*len(xr)
#mark red a point that falls outside of the control limits. Otherwise
    , mark the point blue
markers = []
colors = []
for i in range (len(xr)):
    x1 = xr[i]
    if (x1 > UCL[i] or x1<LCL[i]):
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
#plotting U chart
fig=figure()
ax1 =  fig.add_subplot(111)
ax1.step(t,UCL, 'k-', alpha = 1, where = 'mid')
ax1.step(t, LCL, 'k-',alpha = 1,  where = 'mid')
ax1.step(t,UCL, 'k-', alpha = 0.5, where = 'mid')
ax1.step(t, LCL, 'k-',alpha = 0.5,  where = 'mid')
ax1.step(t,CL, 'k-',alpha = 0.5, where = 'mid')
ax1.plot(xr,'b-',zorder=1)
for x,y,c,m in zip(t, xr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel('Sample number')
ax1.set_ylabel('u')
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL=$'+str(round(UCL[-1],2)), xy = (xlim()[1], list(
    UCL)[-1]), xytext = (xlim()[1],list(UCL)[-1]),fontsize = 11)
ax1.annotate ('$\overline{C}=$'+str(round(CL[-1],2)), xy = (xlim()
    [1], list(CL)[-1]), xytext = (xlim()[1],list(CL)[-1]),fontsize =
    11)
ax1.annotate ('$LCL=$'+str(round(LCL[-1],2)), xy = (xlim()[1], list(
    LCL)[-1]), xytext = (xlim()[1],list(LCL)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(data), step = 2), arange(1, len(data)+1, step = 2))
show()
```

How-To 2.15 (c and u charts in Excel 2013)
Excel does not have a built-in option to create c and u charts, but we can manually program the formulas in Box 2.5 as we demonstrate in Example 2.5.

Table 2.5: Weekly samples and the changes made to the codes of the CAC system by the class

Week	n	A	B	C	D
1	25	14	4	3	29
2	25	5	4	19	12
3	25	10	1	8	13
4	25	14	4	10	30
5	25	1	0	1	15
6	25	4	2	2	21
7	25	9	1	10	14
8	25	2	2	16	29
9	25	8	3	14	13
10	25	11	1	10	25
11	25	12	3	12	26
12	25	10	5	2	12
13	25	1	4	1	24
14	25	14	0	17	11
15	25	8	1	16	25
16	25	8	5	12	13
17	25	9	5	3	20
18	25	14	5	13	19
19	25	6	1	9	28
20	25	2	5	5	26
21	25	6	3	7	28
22	25	11	1	4	15
23	25	5	2	16	20
24	25	8	0	9	16
25	25	10	4	9	10
26	25	15	4	8	10
27	25	14	3	2	28
28	25	6	3	6	13
29	25	4	1	16	28
30	25	4	5	1	15
31	25	6	1	5	22
32	25	8	3	11	19
33	25	13	2	11	16
34	25	8	4	17	14
35	25	4	3	9	18
36	25	2	1	11	11
37	25	1	1	16	16
38	25	14	2	8	23
39	25	15	2	3	26
40	25	2	0	7	29

Example 2.5 (c and u charts)
Metropolis Hospital has just implemented an EHR system, which came with a module on computer-assisted coding (CAC). Like many CAC technologies on the market, the one soon to be used at Metropolis Hospital also employs the elements of convolution networks and natural language processing (NLP) to translate electronic clinical notes into medical classification and terminology codes such as SNOMED-CT[a], ICD-10-CM/PCS[b], and CPT[c]. These codes can then be used for many purposes, including health information exchange (HIE), reimbursement, quality reporting, and operational management. The hospital's CEO is excited about the prospects of this new technology to reduce coding costs and boost productivity. But, the HIM manager has cautioned that all existing CAC systems still require trained medical coders to audit the codes [24]. Accordingly, the manager has decided to track the performance of the CAC system by taking 25 random samples each week and recording the changes that coders make, including adding, remov-

ing, or modifying codes. Furthermore, the manager created a demerit system to weigh the significance of the changes. Table 2.5 presents random samples that the manager has collected. The demerit system utilized has four classes (A, B, C, and D) as described next:

Class A: The change of the principal diagnosis or principal procedure code (weight of **100**)

Class B: The change of a secondary diagnosis or a secondary procedure code (weight of **50**)

Class C: The change of a code classified as a factor influencing health status and contact with health services (weight of **10**)

Class D: The change of an external cause code (weight of **1**)

Using the data in Table 2.5, the manager used Python to program the c chart, unweighted u chart, and weighted u chart as portrayed in Figures 2.21 - 2.23, respectively.

Figure 2.21: A c chart created using Python based on the data in Table 2.5

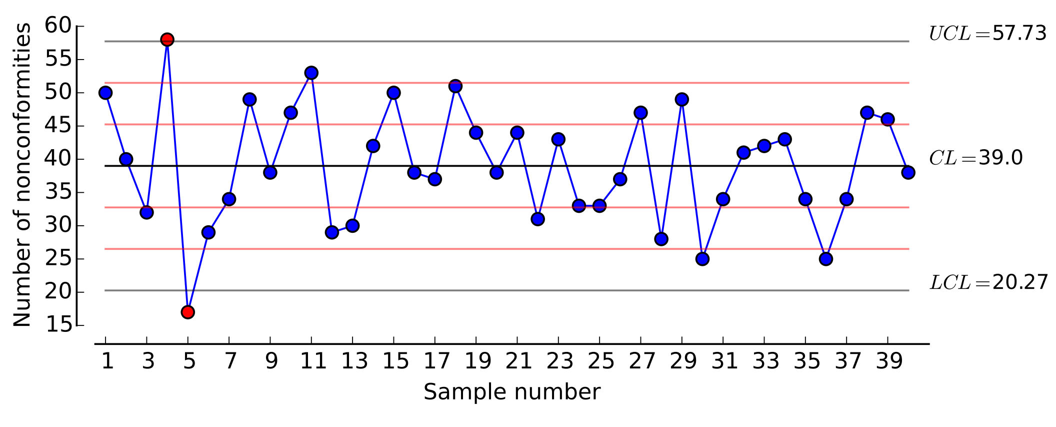


Figure 2.22: An unweighted u chart created using Python based on the data in Table 2.5

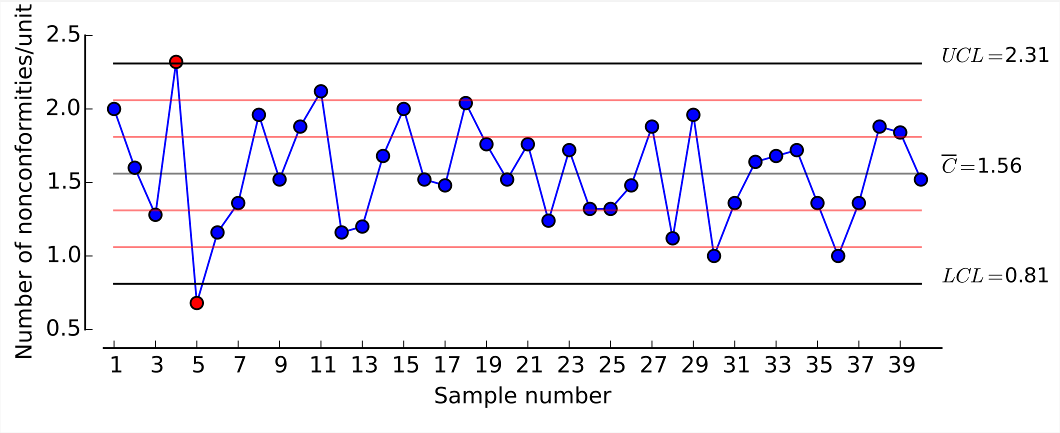
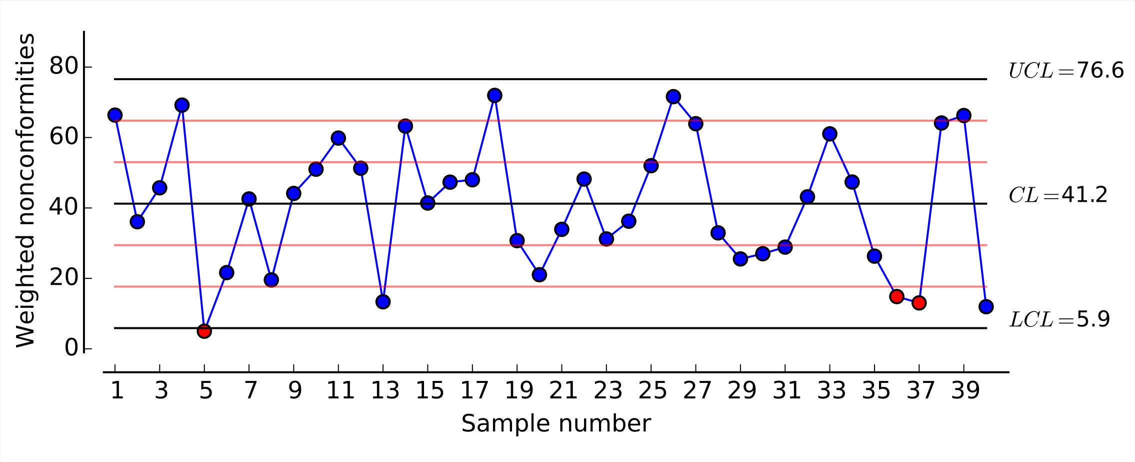


Figure 2.23: A weighted u chart created using Python based on the data in Table 2.5

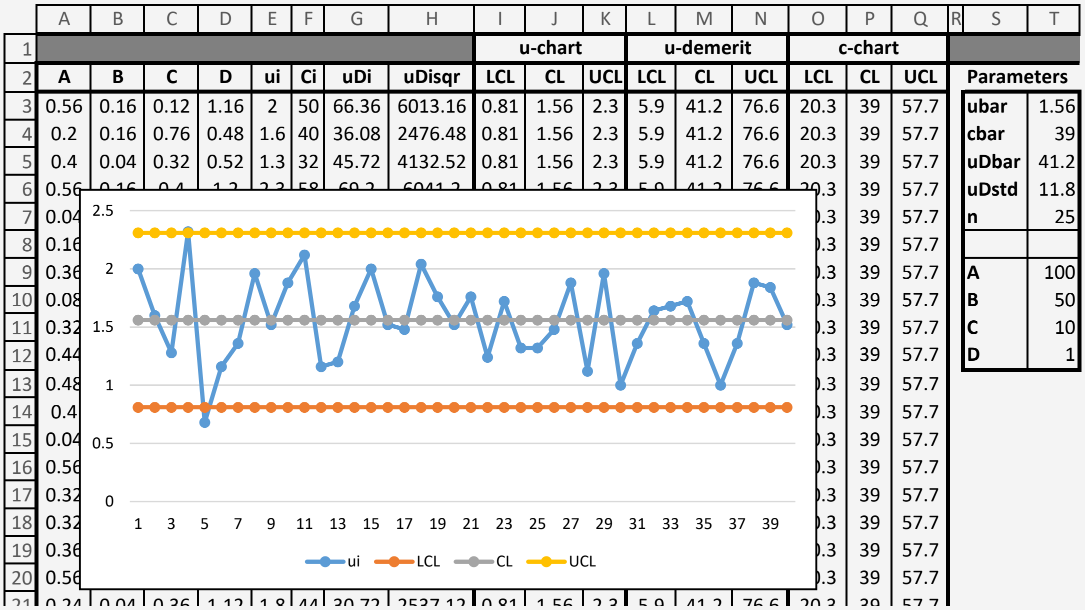
.



Both the c and unweighted u charts indicate that samples 4 and 5 are out-of-control. After weighing the defects with the given demerit-system, the u chart shows that only sample 5 falls outside of the control limits. Also, samples 36 and 37 violated a sensitizing rule that forbids two out of three consecutive points from falling between the second and third limits on the same side. To improve the process, the manager will have to find and remove all assignable causes from the process.

To reproduce the same charts in Excel, set up your spreadsheet, as illustrated in Figure 2.24.

Figure 2.24: A setup of Excel to create the c and u control charts based on the data in Table 2.5



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1									u-chart			u-demerit			c-chart					
2	A	B	C	D	ui	Ci	uDi	uDisqr	LCL	CL	UCL	LCL	CL	UCL	LCL	CL	UCL		Parameters	
3	0.56	0.16	0.12	1.16	2	50	66.36	6013.16	0.81	1.56	2.3	5.9	41.2	76.6	20.3	39	57.7		ubar	1.56
4	0.2	0.16	0.76	0.48	1.6	40	36.08	2476.48	0.81	1.56	2.3	5.9	41.2	76.6	20.3	39	57.7		cbar	39
5	0.4	0.04	0.32	0.52	1.3	32	45.72	4132.52	0.81	1.56	2.3	5.9	41.2	76.6	20.3	39	57.7		uDbar	41.2
6	0.56	0.16	0.4	1.2	2.3	58	69.2	6041.2	0.81	1.56	2.3	5.9	41.2	76.6	20.3	39	57.7		uDstd	11.8
7	0.04														.3	39	57.7		n	25
8	0.16														.3	39	57.7			
9	0.36														.3	39	57.7		A	100
10	0.08														.3	39	57.7		B	50
11	0.32														.3	39	57.7		C	10
12	0.44														.3	39	57.7		D	1
13	0.48														.3	39	57.7			
14	0.4														.3	39	57.7			
15	0.04														.3	39	57.7			
16	0.56														.3	39	57.7			
17	0.32														.3	39	57.7			
18	0.32														.3	39	57.7			
19	0.36														.3	39	57.7			
20	0.56														.3	39	57.7			
21	0.24	0.04	0.36	1.12	1.8	44	30.72	2537.12	0.81	1.56	2.3	5.9	41.2	76.6	20.3	39	57.7			

The explanation of how we set up the spreadsheet in Figure 2.24 follows.

1. Columns A through D in Figure 2.24 reflect the data from Table 2.5 after dividing each row by 25. For example, the values of row 3, columns A-D, were obtained as: 14/25 = 0.56, 4/25 = 0.16, 3/25 = 0.12, and 29/25 = 1.16, respectively.

2. Column E contains the u_i statistic to be monitored in an unweighted u chart. For example, the value in cell $E3$ was obtained by $= SUM(A3 : D3)$. You can drag down this formula to populate the rest of the values.

3. Column F contains the C_i statistic to be monitored in a c chart. For example, the value in cell $F3$ was obtained by $= E3 * \$T\$7$, which is equivalent to summing all values of the first row in Table 2.5 (14 + 4 + 3+ 29 = 50). You can drag down the formula in F3 to populate the rest of the C_i values.

4. Column G contains the uD_i statistic to be monitored in a demerit-based u chart. Equation 2.52 is used to calculate each value. For example, the value in cell $G3$ was obtained using $= A3 * \$T\$9 + B3 * \$T\$10 + C3 * \$T\$11 + D3 * \$T\$12$.

5. Column H contains values that will be used to determine the standard deviation to create the control limits for the demerit-based u chart. Each value was obtained using the radicand in Equation 2.51. For example, the value in cell $H3$ was obtained like this: $= A3 * \$T\$9^{\wedge 2} + B3 * \$T\$10^{\wedge 2} + C3 * \$T\$11^{\wedge 2} + D3 * \$T\$12^{\wedge 2}$. You can drag down this formula to populate the rest of the values. To include an exponent in Excel, use the caret sign ^.

6. The details for calculating the parameters for creating the control limits follow.

   (a) The $\bar{u}$ value in cell $T3$ was obtained this way: $= AVERAGE(E3 : E42)$.

   (b) The $\bar{c}$ value in cell $T4$ was calculated this way: $= AVERAGE(F3 : F42)$.

   (c) The $\bar{u}D$ value in cell $T5$ was obtained this way: $= AVERAGE(G3 : G42)$.

   (d) The $\hat{\sigma}_u$ value in cell $T6$ was determined by $= SQRT(AVERAGE(H3 : H42)/T7)$.

   (e) The sample size n in cell $T7$ comes from column **n** in Table 2.5.

   (f) The demerit weights in cells $T9$ - $T12$ come from Box 2.6.

7. To determine the control limits, per formulas in Boxes 2.5 and 2.7, the first row was computed as follows:

   (a) The value of LCL in cell $I3$ was obtained as follows: $= \$T\$3 - 3 * SQRT(\$T\$3/\$T\$7)$.

   (b) The value of CL in cell $J3$ was calculated like this: $= \$T\$3$.

   (c) The value of UCL in cell $K3$ was given by: $= \$T\$3 + 3 * SQRT(\$T\$3/\$T\$7)$.

   (d) The value of LCL in cell $L3$ was computed this way: $= \$T\$5 - 3 * \$T\$6$.

   (e) The value of CL in cell $M3$ was obtained like this: $= \$T\$5$.

   (f) The value of UCL in cell $N3$ was obtained this way: $= \$T\$5 + 3 * \$T\$6$.

   (g) The value of LCL in cell $O3$ was calculated this way: $= \$T\$4 - 3 * SQRT(\$T\$4)$.

   (h) The value of CL in cell $P3$ was obtained this way: $= \$T\$4$.

   (i) The value of UCL in cell $Q3$ was calculated this way: $= \$T\$4 + 3 * SQRT(\$T\$4)$.

8. To create the control charts of interest proceed by:

   (a) Inserting line charts of columns E and I-K to create an unweighted u chart.

   (b) Inserting line charts of columns F and O-Q to create a c chart.

   (c) Inserting line charts of columns G and L-N to create a demerit-based u chart.

You can recreate the c and unweighted u charts in Minitab by following the instructions in How-To 2.13.

[a] **SNOMED-CT:** Systematized Nomenclature of Medicine – Clinical Terms

[b] **ICD-10-CM/PCS:** The International Classification of Diseases, Tenth Revision, Clinical Modification/Procedure Coding System

[c] **CPT:** Current Procedural Terminology

2.3.3 g and h charts

The statistics for a g chart

We use g charts to model the **total number of events** between rare incidents. It is assumed that these events are independent and identically distributed according to the geometric distribution modeled as follows [34]:

$$p(x) = p(1-p)^{x-a} \qquad x = a, a+1, a+2, \ldots \tag{2.53}$$

where a is the minimum number of events, and p is the probability of an event. Unless p is given, we can estimate it using $\bar{p}$ like this [44]:

$$\bar{p} = \frac{1}{\bar{x} - a + 1} \tag{2.54}$$

Here, $\bar{x}$ is the average number of events. The mean μ_T and standard deviation σ_T for the process of the total number of events are given by [44]:

$$\mu_T = n\left(\frac{1-p}{p} + a\right) \tag{2.55}$$

$$\sigma_T = \frac{n(1-p)}{p^2} \tag{2.56}$$

where n is the sample size.

The statistics for a h chart

We use h charts to model the **average number of events** between rare incidents. The occurrence of these events is also assumed to follow the geometric distribution shown Equation 2.53. The mean $\mu_{\bar{x}}$ and standard deviation $\sigma_{\bar{x}}$ in this case, are given by:

$$\mu_{\bar{x}} = \frac{1-p}{p} + a \tag{2.57}$$

$$\sigma_{\bar{x}} = \frac{(1-p)}{np^2} \tag{2.58}$$

If not given, we estimate p as indicated in Equation 2.54.

Formulation

Box 2.8 summarizes the formulas for the g and h charts.

The time between rare events

Instead of monitoring the number of events between successive rare incidents, we could instead try to monitor the time between them. If defects occur according to a Poisson distribution, the time between events can be modeled using an exponential distribution.

Box 2.8 The formulas for the g and h control charts when $L = 3$

When the **standard p is given**, we construct the g and h charts as follows:

	g chart	**h chart**
UCL	$n\left(\frac{1-p}{p}+a\right)+3\sqrt{\frac{n(1-p)}{p^2}}$	$\frac{1-p}{p}+a+3\sqrt{\frac{1-p}{np^2}}$
CL	$n\left(\frac{1-p}{p}+a\right)$	$\frac{1-p}{p}+a$
LCL	$n\left(\frac{1-p}{p}+a\right)-3\sqrt{\frac{n(1-p)}{p^2}}$	$\frac{1-p}{p}+a-3\sqrt{\frac{1-p}{np^2}}$

When the **standard p is not given**, we construct the g and h charts this way:

	g chart	**h chart**
UCL	$n\left(\frac{1-\bar{p}}{\bar{p}}+a\right)+3\sqrt{\frac{n(1-\bar{p})}{\bar{p}^2}}$	$\frac{1-\bar{p}}{\bar{p}}+a+3\sqrt{\frac{1-\bar{p}}{n\bar{p}^2}}$
CL	$n\left(\frac{1-\bar{p}}{\bar{p}}+a\right)$	$\frac{1-\bar{p}}{\bar{p}}+a$
LCL	$n\left(\frac{1-\bar{p}}{\bar{p}}+a\right)-3\sqrt{\frac{n(1-\bar{p})}{\bar{p}^2}}$	$\frac{1-\bar{p}}{\bar{p}}+a-3\sqrt{\frac{1-\bar{p}}{n\bar{p}^2}}$

In the g chart, we monitor the total number of events between successive rare incidents. In the h chart, we monitor the corresponding average number of events.

But, the exponential distribution is not typically used to construct control charts due to the skewness of this distribution [44]. To get around this issue, we transform the exponential data into a Weibull distribution to allow for the approximation of the normal distribution. The following **Nelson's transformation** is usually used for this purpose [44, 65].

$$\hat{x} = x^{0.2777} \tag{2.59}$$

Here, x is the original data, and $\hat{x}$ is the transformed data. Subsequently, we can apply traditional variable control charts using the transformed data [65]. We should note that, besides Nelson's method in Equation 2.59, other data transformation techniques exist, such as the approach used to create T charts in Minitab 18 [66].

How-To 2.16 (Python 3.6)

Script 2.6: A script for creating a g chart in Python 3.6

```
#ATTRIBUTE CHARTS G chart
#import modules
from pandas import*
from pylab import*
from numpy import*
import seaborn as sns
#Import data from an excel spreadsheet
#Bi is the column for the number of events between successive rare
    events
data = read_excel('your directory')
#G chart
xr = data.Bi
t = arange(len(xr))
n = 1.
a = 1.
xbarbar = xr.mean() - a +1.
p = 1./xbarbar
L = 3.
std = sqrt(n*(1.-p)/(p**2))
#control limits
CL = [n*(((1.-p)/p)+a)]*len(xr)
UCL = [CL[0] + L*std ]*len(xr)
LCL = [max(0,CL[0] - L*std)]*len(xr)
#mark red a point that falls outside of the control limits. Otherwise
    , mark the point blue
markers = []
colors = []
for i in xrange (len(xr)):
    x1 = xr[i]
    if (x1 > UCL[i] or x1<LCL[i]):
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
#plotting the G chart
fig=figure()
ax1 =  fig.add_subplot(111)
ax1.step(t,CL, 'k-',alpha = 1, where = 'mid')
ax1.step(t,UCL, 'k-', alpha = 1, where = 'mid')
ax1.step(t, LCL, 'k-',alpha = 1,  where = 'mid')
ax1.plot(xr,'b-',zorder=1)
for x,y,c,m in zip(t, xr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel('Sample number')
ax1.set_ylabel('Number of events')
```

```
#annotate the values of UCL, LCL, and CL
ax1.annotate (’$UCL=$’+str(round(UCL[-1],2)), xy = (xlim()[1], list(
    UCL)[-1]), xytext = (xlim()[1],list(UCL)[-1]),fontsize = 11)
ax1.annotate (’$CL=$’+str(round(CL[-1],2)), xy = (xlim()[1], list(CL)
    [-1]), xytext = (xlim()[1],list(CL)[-1]),fontsize = 11)
ax1.annotate (’$LCL=$’+str(round(LCL[-1],2)), xy = (xlim()[1], list(
    LCL)[-1]), xytext = (xlim()[1],list(LCL)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(data), step = 2), arange(1, len(data)+1, step = 2))
show()
```

How-To 2.17 (Python 3.6)

Script 2.7: A script for Nelson’s transformation in Python 3.6

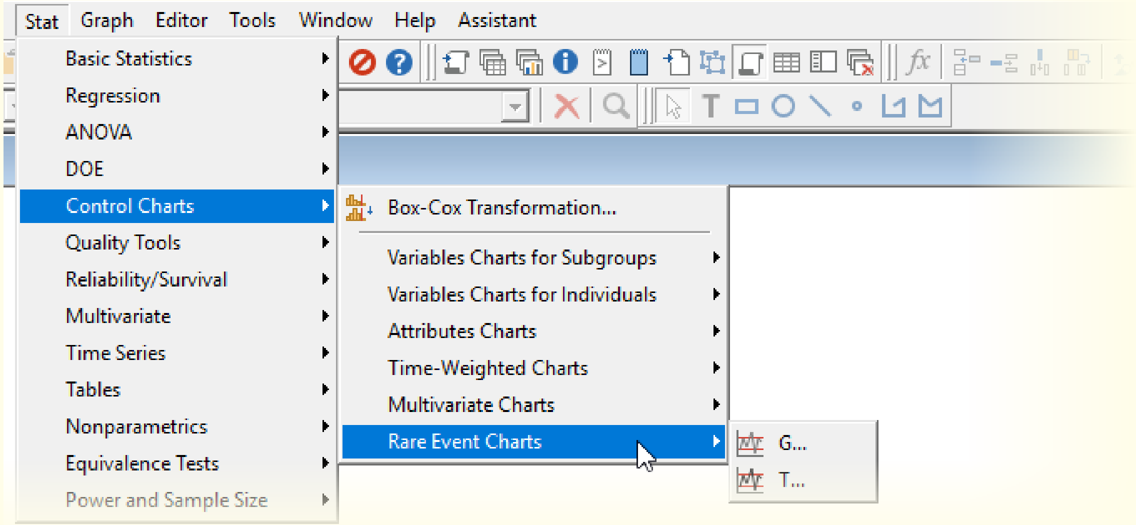
```
#ATTRIBUTE CHARTS to VARIABLE CHARTS via Nelson’s transformation
#Import modules
from pandas import*
from pylab import*
from numpy import*
import seaborn as sns
#import data from an excel spreadsheet. Days is the column with time
    data
data = read_excel(’your directory’)
#Nelson’s transformation
data[’xt’] = data[’Days’].apply(lambda x: x**0.2777)
#initialize parameters for IMR charts
xr = data.xt
xbar = mean(xr)
d2 = 1.128
D3 = 0.
D4 = 3.267
mr = [abs(xr[i]-xr[i-1]) for i in range(1, len(xr))]
mrbar = mean(mr)
t = arange(len(data))
#I chart
#control limits
UCL = [xbar + 3.*(mrbar/d2)]*len(xr)
LCL = [ max(0,xbar - 3.*(mrbar/d2))]*len(xr)
CL = [xbar]*len(xr)
#mark red a point that falls outside of the control limits. Otherwise
    , mark the point blue
markers = []
colors = []
for i in range (len(data)):
    x1 = data.ix[i][’xt’]
    if x1 > UCL[0]:
        markers.append(’o’)
        colors.append(’r’)
    else:
        markers.append(’o’)
        colors.append(’b’)
```

```
#plotting I chart
fig=figure()
ax1 =  fig.add_subplot(111)
ax1.plot(UCL, 'k-', alpha = 0.5)
ax1.plot(LCL, 'k-',alpha = 0.5)
ax1.plot(CL, 'k-',alpha = 0.5)
ax1.plot(xr,'b-',zorder=1)
for x,y,c,m in zip(t, xr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel('Sample number')
ax1.set_ylabel('Individual value')
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL=$'+str(round(UCL[0],2)), xy = (xlim()[1], list(
   UCL)[-1]), xytext = (xlim()[1],list(UCL)[-1]),fontsize = 11)
ax1.annotate ('$\overline{X}=$'+str(round(CL[0],2)), xy = (xlim()[1],
    list(CL)[-1]), xytext = (xlim()[1],list(CL)[-1]),fontsize = 11)
ax1.annotate ('$LCL=$'+str(round(LCL[0],2)), xy = (xlim()[1], list(
   LCL)[-1]), xytext = (xlim()[1],list(LCL)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(data), step = 3), arange(1, len(data)+1, step = 3))
show()
####################
#MR chart
#control limits
UCL = [mrbar*D4]*len(mr)
LCL = [max(0,mrbar*D3)]*len(mr)
CL = [mrbar ]*len(mr)
#mark red a point that falls outside of the control limits. Otherwise
   , mark the point blue
markers = []
colors = []
for i in range (len(mr)):
    x1 = mr[i]
    x2 = UCL[i]
    x3 = LCL[i]
    if (x1 > x2 or  x1<x3) :
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
#plotting MR chart
fig=figure()
ax1 =  fig.add_subplot(111)
t = arange(len(data))
ax1.plot(UCL, 'k-', alpha = 0.5)
ax1.plot(LCL, 'k-',alpha = 0.5)
ax1.plot(CL, 'k-',alpha = 0.5)
ax1.plot(mr,'b-',zorder=1)
for x,y,c,m in zip(t, mr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
ylim(ymin = -0.1)
xlim(-0.3, t[-1]+1)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel('Sample number')
ax1.set_ylabel('Moving Range')
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL=$'+str(round(UCL[-1],2)), xy = (xlim()[1], list(
    UCL)[-1]), xytext = (xlim()[1],list(UCL)[-1]),fontsize = 11)
ax1.annotate (r'$\overline{MR}=$'+str(round(CL[0],2)), xy = (xlim()
    [1], list(CL)[-1]), xytext = (xlim()[1],list(CL)[-1]),fontsize =
    11)
ax1.annotate ('$LCL=$'+str(round(LCL[-1],2)), xy = (xlim()[1], list(
    LCL)[-1]), xytext = (xlim()[1],list(LCL)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(data), step = 3), arange(1, len(data)+1, step = 3))
show()
```

How-To 2.18 (Rare events in Minitab 18)

To construct rare events charts in Minitab 18, click on *Stat* > *Control Charts* > *Rare Event Charts* > *select the appropriate chart*. Minitab 18 only has the choices for the G and T charts. See the snapshot in Figure 2.25.

Figure 2.25: Options for the G and T charts in Minitab 18



How-To 2.19 (g and h charts in Excel 2013)

Excel does not have a built-in option to create g and h charts, but we can manually program the formulas in Box 2.8 as we demonstrate in Example 2.6.

Example 2.6 (g chart)
During a regular meeting of the patient safety committee at Metropolis Hospital, a discussion came up about how to track and monitor hospital-acquired conditions (HACs). The HIM manager who sits on this committee suggested that in addition to the data from the incident reporting system, ICD-10-CM/PCS[a] and POA[b] indicators could also be used to track HACs. The committee approved this suggestion and recommended that ICD-10-CM/PCS codes and POA indicators be used to track the following HACs events as defined by CMS [26]:

1. Foreign objects retained after surgery
2. Air embolism
3. Stage III and IV pressure ulcers
4. Falls and trauma
5. Manifestations of glycemic control
6. Catheter-associated urinary tract infection
7. Vascular catheter-associated infection
8. Post-surgical wound infection
9. Deep vein thrombosis
10. Iatrogenic pneumothorax with venous catheterization.

Given the rarity of these HACs at Metropolis Hospital, the committee decided to apply the g chart to monitor the number of discharges between successive incidents. The historical data collected so far is shown in Table 2.6.

Table 2.6: The number of discharges between HACs

HAC#	**Discharges**	***HAC#***	**Discharges**
	Between		Between
1	20	*14*	210
2	16	*15*	81
3	120	*16*	15
4	41	*17*	45
5	380	*18*	17
6	66	*19*	174
7	21	*20*	29
8	249	*21*	312
9	34	*22*	66
10	24	*23*	24
11	100	*24*	35
12	50	*25*	142
13	31	*26*	27

Python was applied to create the g chart per Box 2.8 using $a = 1$ and $n = 1$. The resulting chart is portrayed in Figure 2.26.

Figure 2.26: A g chart created using Python based on the data in Table 2.6

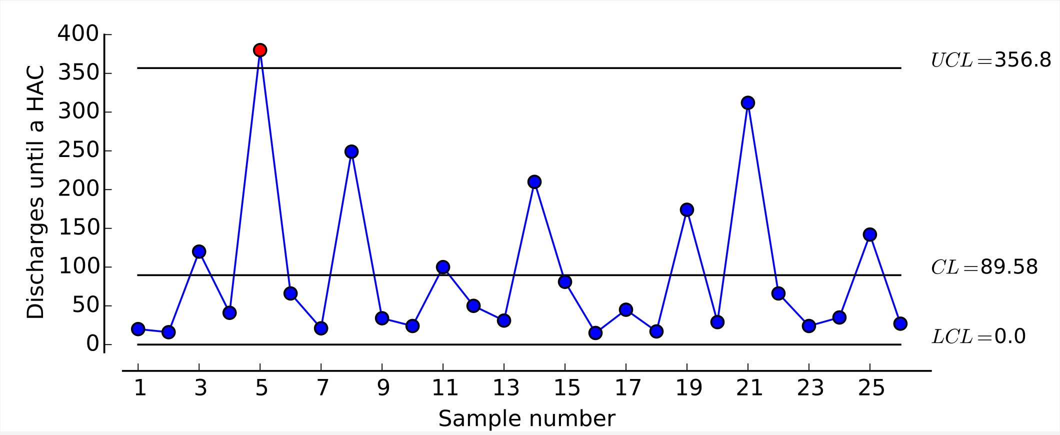
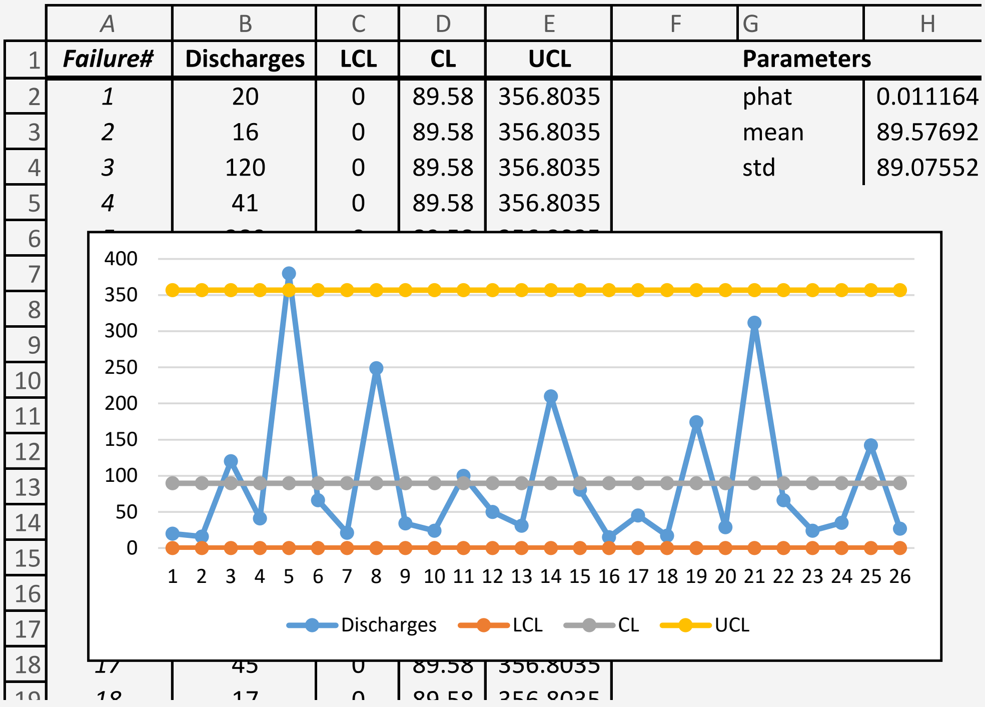


Figure 2.26 shows that sample 5 is out-of-control. Unless this was a reporting error, this represents a process that was doing well since there was an unusually high number of discharges between HAC#4 and HAC#6. The manager should investigate this point and learn about what happened during that period and then recalculate the control charts. To reproduce the same chart in Excel, first set up your spreadsheet, as shown in Figure 2.27 and then follow subsequent instructions.

Figure 2.27: A setup of a g control chart in Excel based on the data in Table 2.6



	A	B	C	D	E	F	G	H
1	***Failure#***	**Discharges**	**LCL**	**CL**	**UCL**		**Parameters**	
2	*1*	20	0	89.58	356.8035		phat	0.011164
3	*2*	16	0	89.58	356.8035		mean	89.57692
4	*3*	120	0	89.58	356.8035		std	89.07552
5	*4*	41	0	89.58	356.8035			
6								
7								
8								
9								
10								
11								
12								
13								
14								
15								
16								
17								
18	*17*	45	0	89.58	356.8035			
19	*18*	17	0	89.58	356.8035			

1. First, determine the parameters of your g chart:
   (a) The phat ($\hat{p}$) in cell $H2$ follows from Equation 2.54 when $a = 1$. This quantity was obtained using $= 1/H3$.
   (b) The mean value in cell $H3$ was calculated as $= AVERAGE(B2 : B27)$.
   (c) Per Box 2.8, the value of the standard deviation in cell $H4$ was obtained this way $= SQRT((1 - H2)/H2^{\wedge 2}$.
2. Next, create the control limits. For example,
   (a) The LCL value in cell $C2$ was obtained this way: $= MAX(0, 1/\$H\$2 - 3 * \$H\$4)$. This formula can be dragged down to populate the rest of the values.
   (b) The CL value in cell $D2$ was obtained like this: $= \$H\$3$. This formula can also be dragged down to populate the rest of the values.
   (c) The UCL value in cell $E2$ was obtained this way: $= MAX(0, 1/\$H\$2 + 3 * \$H\$4)$. Again, this formula can be dragged down to populate the rest of the values.
3. Finally, create the g chart by inserting line charts of columns B-E.

[a] **ICD-10-CM/PCS:** International Classification of Diseases-10-Clinical Modification/Procedure Coding System

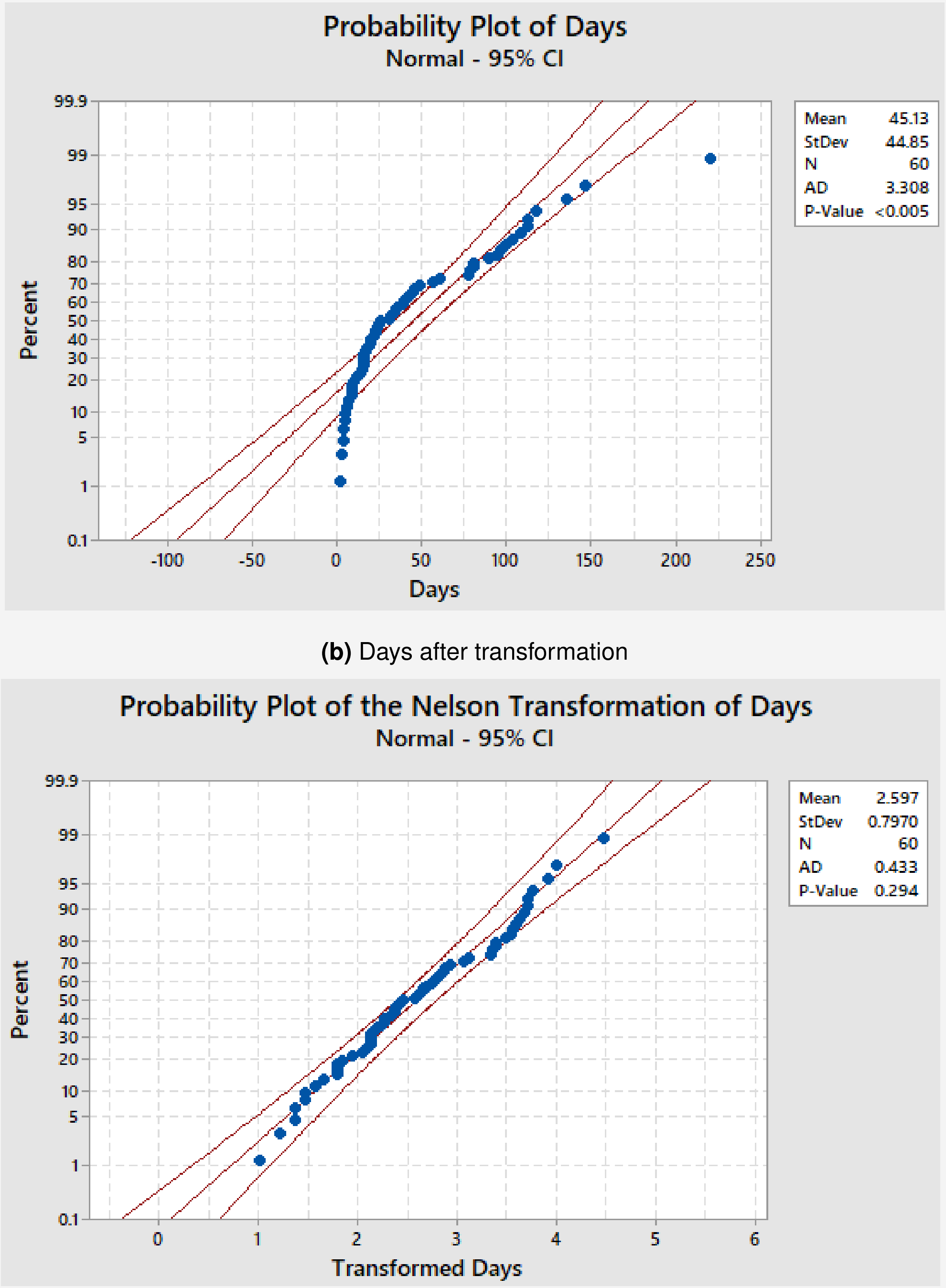
[b] **POA:** Present on Admission

Table 2.7: Number of days until the next HAC event

HAC#	**Days** between	**Days transformed**	*HAC #*	**Days** Between	**Days Transformed**
1	56	3.058242	31	33	2.640531
2	21	2.329058	32	43	2.841935
3	19	2.265217	33	78	3.353009
4	48	2.930088	34	8	1.781509
5	8	1.781509	35	15	2.121292
6	117	3.752625	36	4	1.469576
7	4	1.469576	37	17	2.19632
8	15	2.121292	38	80	3.376666
9	140	3.944389	39	5	1.563522
10	1	1.000000	40	146	3.990624
11	11	1.946233	41	3	1.35674
12	99	3.582514	42	31	2.595082
13	112	3.707386	43	34	2.662513
14	8	1.781509	44	60	3.117401
15	16	2.159654	45	15	2.121292
16	22	2.359341	46	3	1.35674
17	45	2.878042	47	15	2.121292
18	112	3.707386	48	108	3.670133
19	24	2.417045	49	41	2.804594
20	23	2.388646	50	80	3.376666
21	13	2.038647	51	25	2.444601
22	89	3.478128	52	103	3.622137
23	19	2.265217	53	39	2.765914
24	96	3.552031	54	77	3.341015
25	14	2.081037	55	6	1.644722
26	18	2.23146	56	44	2.860137
27	35	2.684032	57	135	3.904754
28	22	2.359341	58	2	1.212261
29	38	2.746034	59	94	3.531324
30	30	2.571559	60	9	1.840743

Example 2.7 (Data transformation)
Following from Example 2.6, the safety committee also decided to monitor the number of days between successive HACs. Table 2.6 shows the transformed number of days to be monitored using I-MR charts. Before applying these charts, the committee wanted to make sure that the normality assumption, which is required for variable charts, was satisfied. Accordingly, probability plots were created in Minitab, as illustrated in Figure 2.28.

Figure 2.28: Probability plots of days based on the data in Table 2.7

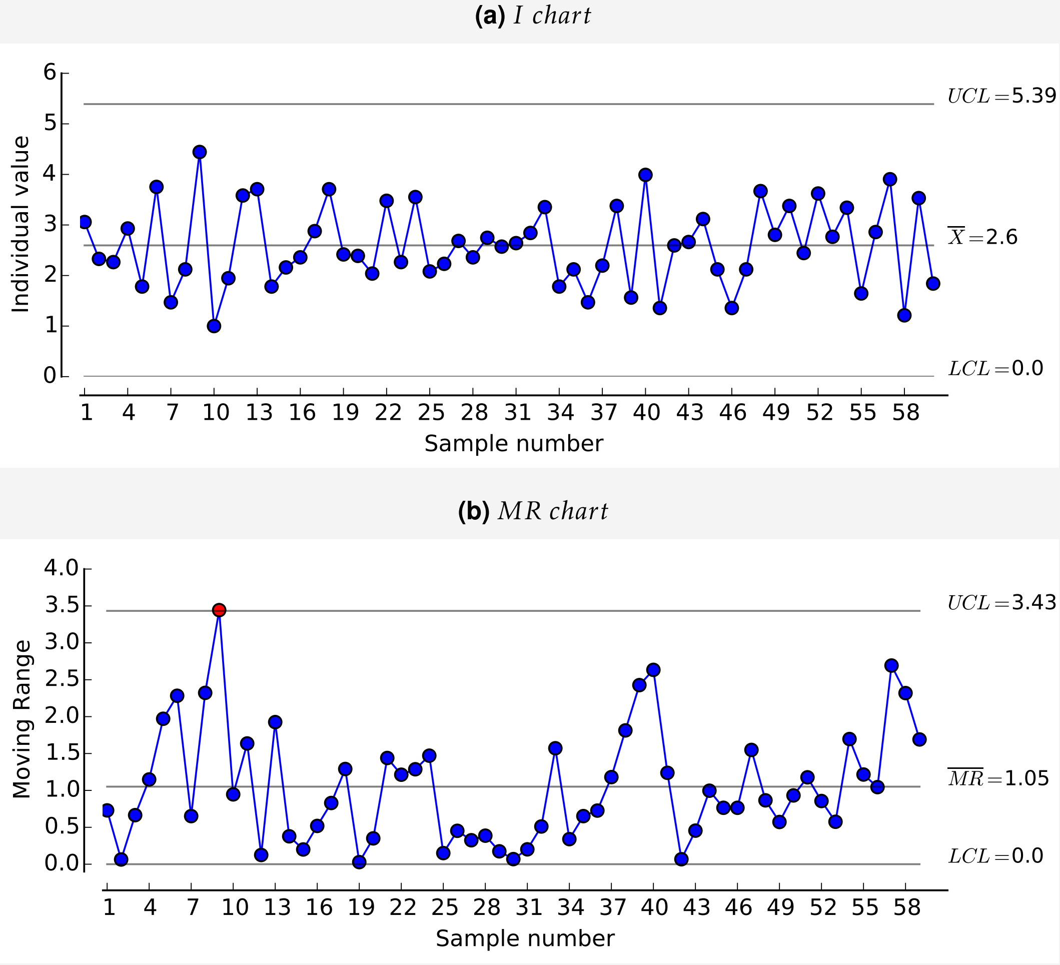


(a) Days before transformation

(b) Days after transformation

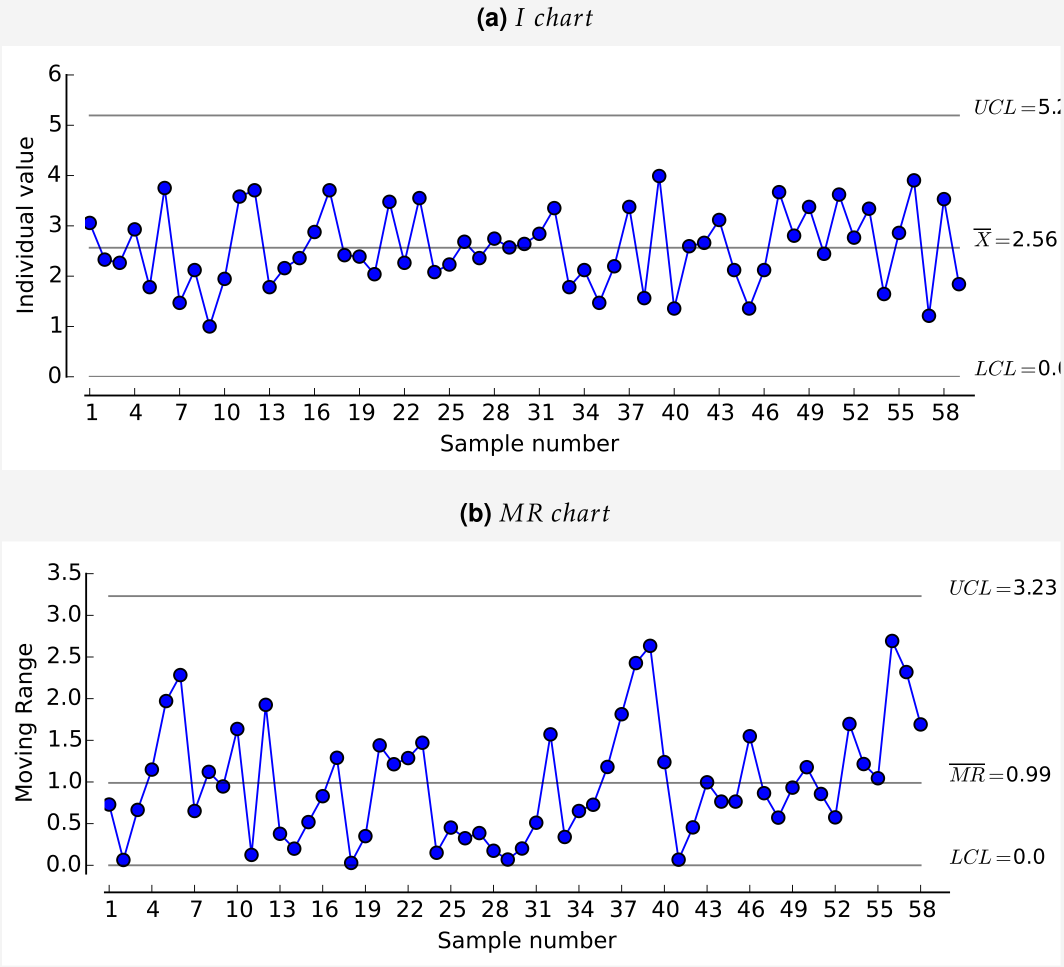
The plot in Subfigure 2.28b indicates that the transformed days are about normally distributed since $p\text{-}value > 0.05$. This result suggests the failure to reject the hypothesis that the transformed days are normally distributed. Subsequently, the committee created $I\text{-}MR$ charts per Box 2.1. The resulting charts are portrayed in Figure 2.29.

Figure 2.29: I-MR control charts based on the transformed number of days in Table 2.7



Subfigure 2.29a indicates a stable process, but Subfigure 2.29b shows out-of-control behaviors in the moving ranges at sample number 9. The committee investigated this sample and assigned it to a data entry error. As a result, sample 9 was omitted from the process, and control charts were recreated to monitor the future process. The new charts are presented in Figure 2.30.

Figure 2.30: I-MR control charts, based on the transformed number of days in Table 2.7, after removing sample 9



(a) *I chart*

(b) *MR chart*

2.4 EXERCISES

1. Using Excel, reproduce the $XbarS$ chart in Example 2.3.

2. The chief financial officer at Metropolis Hospital would like to start monitoring the costs for total hip replacements. Table 2.8 presents samples of 40 individual patients who recently had hips replaced at this facility. Using Excel, create appropriate control charts, and stabilize the process. What can you conclude? ***Hint:*** *to stabilize the process, if any point falls outside of the control limits, remove it and create new control charts. Repeat this procedure until the process is stable.*

Table 2.8: Costs X in thousand, for total hip replacements at Metropolis Hospital, 2018

Patient#	**X**	***Patient#***	**X**
1	34	*21*	33
2	41	*22*	35
3	40	*23*	28
4	36	*24*	39
5	37	*25*	37
6	35	*26*	36
7	42	*27*	36
8	37	*28*	48
9	31	*29*	39
10	42	*30*	33
11	44	*31*	27
12	35	*32*	28
13	37	*33*	25
14	26	*34*	34
15	27	*35*	42
16	49	*36*	31
17	29	*37*	47
18	30	*38*	29
19	40	*39*	38
20	39	*40*	37

3. At Metropolis Hospital, the CIO keeps track of the queue time of IT tickets. Table 2.9 portrays the samples that the manager has collected. Use this data to create appropriate control charts and then stabilize the process. What can you conclude? If the goal is to have an average queue time of 70 minutes, how is the IT department doing on this goal?

Table 2.9: Queue time in minutes, clinical IT tickets at Metropolis Hospital, 2018

Sample#	**x1**	**x2**	**x3**	**x4**	**x5**	**x6**	**x7**
1	110	45	84	86	58	75	38
2	80	70	110	148	32	84	42
3	22	111	37	100	109	62	63
4	82	67	55	30	129	63	57
5	74	40	82	59	82	43	84
6	38	40	139	100	59	44	85
7	80	123	52	80	48	45	96
8	76	97	54	64	65	75	110
9	64	47	66	135	65	79	91
10	80	35	47	59	50	119	44
11	65	76	118	64	38	49	66
12	60	67	41	86	73	128	88
13	105	49	87	68	105	63	42
14	104	138	50	67	102	54	122
15	51	54	58	81	79	143	61
16	47	60	90	61	66	58	145
17	61	94	104	60	83	50	96
18	32	79	79	99	61	70	84
19	32	101	111	85	48	103	40
20	55	23	97	123	58	47	54
21	46	32	47	55	88	103	80
22	52	130	63	32	50	141	105
23	108	66	104	101	103	100	95
24	90	38	64	64	107	44	84
25	102	48	51	93	55	53	58
26	33	64	57	110	41	114	67
27	37	81	84	90	69	113	52
28	99	96	80	112	84	131	32
29	94	89	114	80	41	96	83
30	78	44	118	58	75	107	63
31	74	123	61	95	109	68	114
32	102	45	63	49	127	78	70
33	79	130	122	51	32	61	62
34	102	31	68	97	68	109	99
35	122	62	56	87	38	70	77
36	81	104	85	52	57	40	28
37	79	43	86	72	57	110	50
38	49	95	71	132	90	112	82
39	200	46	99	150	90	103	80
40	83	72	46	80	44	87	36
41	123	44	108	73	56	51	86
42	96	114	106	33	56	90	101
43	37	53	44	46	50	70	82
44	71	82	112	85	102	43	119
45	126	37	40	31	47	100	120
46	104	87	113	79	59	79	59
47	89	95	54	64	61	102	66
48	61	44	50	59	33	63	73
49	43	53	66	94	95	82	80
50	45	65	56	84	62	24	107

4. The Chief Medical Officer (CMO) at Metropolis Hospital keeps track of the time to extubation of ICU patients. Table 2.10 presents monthly sample statistics of this time. The sample size is $n = 13$, $\mu = 6$, $\sigma = 3$. Create appropriate control charts, **with** and **without** these standards.

   (a) What can you conclude about the stability of the process?

   (b) Any ideas about how to improve the stability of the process?

Table 2.10: Monthly sample statistics of time to extubation, Metropolis Hospital, 2015-2018

Week #	$\bar{x}$	$\bar{s}$
1	5.2	2.4
2	6.3	4.5
3	6.5	3.9
4	7.5	2.9
5	6.9	3.1
6	5.7	3.3
7	7.3	2.1
8	6.8	3.9
9	5	2.3
10	6.8	2.5
11	5.9	3.7
12	8.3	3.5
13	5.1	4.2
14	5.5	3.7
15	5.6	3.2
16	6.3	3.8
17	6.9	3.4
18	7.2	5.8
19	6.5	2.4
20	6.1	3.1
21	5.6	2.7
22	5.3	2.7
23	7.9	7.9
24	5.8	3.6
25	3.4	2
26	6.7	4.6
27	7.5	2.7
28	6.2	2.3
29	5	3.4
30	6.1	1.9

5. The manager of the emergency department (ED) at Metropolis Hospital would like to monitor and control the number of patients leaving without being seen (LWBS). Table 2.11 presents samples collected on weekdays. Implement appropriate control charts for the rate and the number of LWBS. If any point falls outside of the limits, remove it, and recreate the charts. Repeat this procedure until the process is stable.

   (a) What can you conclude?

   (b) How would this process perform against the LWBS target of 2.5%?

Table 2.11: Weekday samples of LWBS in the ED at Metropolis Hospital, 2018

Sample#	Visits	LWBS
1	221	7
2	217	10
3	201	9
4	200	6
5	247	4
6	231	7
7	243	5
8	240	7
9	246	7
10	225	4
11	247	9
12	202	4
13	233	7
14	216	8
15	215	5
16	224	4
17	220	6
18	201	4
19	198	8
20	225	8
21	250	7
22	245	9
23	225	9
24	225	8
25	209	7
26	237	8
27	181	7
28	196	6
29	183	6
30	233	9
31	182	5
32	245	3
33	197	8
34	239	7
35	204	6
36	213	4
37	180	5
38	208	11
39	231	7
40	195	6
41	201	7
42	230	4
43	203	7
44	236	8
45	191	6
46	230	9
47	234	8
48	229	3
49	231	9
50	224	3
51	226	4
52	201	3
53	249	7
54	225	8
55	181	8
56	216	7
57	220	6
58	249	16
59	216	4
60	184	6

6. The CMO at Metropolis Hospital wants to monitor postoperative sternal wound infections for patients who had open-heart surgeries. Given the rarity of such events, the manager will not monitor the number of infections, but rather the number of surgeries between successive infections. Sample data are presented in Table 2.12. Create an appropriate control chart for this process. What can you conclude?

Table 2.12: Number of surgeries between post-operative sternal wound infections at Metropolis Hospital, 2018

Infection#	#s of surgeries	Infection#	#s of surgeries
1	27	17	45
2	38	18	7
3	7	19	8
4	9	20	6
5	9	21	5
6	13	22	12
7	7	23	6
8	14	24	26
9	72	25	4
10	4	26	61
11	7	27	14
12	4	28	6
13	3	29	13
14	30	30	19
15	7	31	51
16	35	32	5

Table 2.13: Weekly samples of billing errors at Metropolis Hospital, 2018

Week	n	A	B	C	D	*Week*	n	A	B	C	D
1	33	12	5	10	16	*21*	33	10	1	18	13
2	33	10	3	16	11	*22*	33	14	2	2	12
3	33	14	4	10	28	*23*	33	3	3	4	17
4	33	7	5	9	14	*24*	33	10	3	4	22
5	33	5	3	10	11	*25*	33	6	2	8	27
6	33	11	2	15	11	*26*	33	15	0	0	15
7	33	4	0	1	10	*27*	33	5	5	1	21
8	33	7	0	19	19	*28*	33	14	2	10	29
9	33	1	4	9	10	*29*	33	8	3	9	23
10	33	9	5	4	23	*30*	33	15	1	19	18
11	33	10	2	11	28	*31*	33	14	2	18	27
12	33	14	4	7	11	*32*	33	10	0	4	23
13	33	13	0	14	16	*33*	33	2	4	3	10
14	33	12	3	0	10	*34*	33	9	3	6	22
15	33	12	3	12	22	*35*	33	12	5	11	30
16	33	13	3	20	30	*36*	33	12	3	15	21
17	33	15	0	13	26	*37*	33	11	0	9	13
18	33	14	4	3	30	*38*	33	12	1	14	12
19	33	11	1	15	14	*39*	33	4	5	5	11
20	33	5	1	17	16	*40*	33	13	5	17	16

7. Consider the data in Table 2.13 showing medical billing errors by the demerit class defined and weighted per Box 2.6.

(a) Create a c chart for this process

(b) Create a Z chart for this process

(c) Create a demerit-based u chart for this process.

(d) Compare and contrast the results from your charts. What can you conclude?

8. The Chief Nursing Officer (CNO) at Metropolis Hospital has just asked you to help create an appropriate control chart for monitoring urinary tract infection (UTI) at the hospital. The CNO has indicated that the numbers of UTIs are few and far between. Accordingly, you suggested monitoring the number of days between UTIs instead of tracking the count of UTIs. The CNO handed you the data presented in Table 2.14.

   (a) Create two applicable control charts and choose the best chart.

   (b) Create probability plots to support the choice of your control chart.

   (c) What can you conclude?

Table 2.14: Days between UTIs at Metropolis Hospital, 2014 - 2018

UTI#	**Days**	***UTI#***	**Days**
	Between		Between
1	61	*17*	25
2	46	*18*	17
3	46	*19*	15
4	102	*20*	35
5	38	*21*	14
6	11	*22*	20
7	22	*23*	24
8	91	*24*	62
9	18	*25*	31
10	31	*26*	60
11	77	*27*	18
12	17	*28*	43
13	94	*29*	30
14	2	*30*	16
15	31	*31*	43
16	66	*32*	29

9. Table 2.15 presents frequencies of monthly overrides of critical alerts in the CPOE system at Metropolis Hospital. Use this data to create an appropriate control chart.

   (a) Using probability plots, justify the choice of the control chart that you applied.

   (b) What can you conclude about the stability fo this process?

Table 2.15: Critical alert overrides in the CPOE system at Metropolis Hospital, 2014-2018

Month	#Overrides	Month	#Overrides
1	42	21	40
2	46	22	47
3	17	23	47
4	25	24	9
5	41	25	44
6	38	26	24
7	23	27	8
8	55	28	23
9	28	29	36
10	12	30	55
11	14	31	63
12	26	32	60
13	28	33	53
14	32	34	26
15	11	35	26
16	53	36	13
17	20	37	11
18	5	38	8
19	8	39	29
20	15	40	75

10. Recreate the np chart in Example 2.4. What should the sample size be to ensure that $LCL > 0$?

CHAPTER 3

Time-Weighted Control Charts

Summary

In this chapter, we review how to apply time-weighted control charts to detect small shifts in the process. Here, we only consider cumulative sum (CUSUM), exponentially weighted moving average (EWMA), and moving average (MA) charts. We provide statistical formulas of these charts and demonstrate how to implement them using Excel, Minitab, and Python software.

Key concepts and tools

Time-weighted charts; CUSUM; EWMA; MA; Out-of-control behaviors; Assignable causes; Special cause variation; Common cause variation; Phase II; Proportional–integral–derivative (PID); Process regulation

Major objectives

After studying this chapter, you will be able to:

1. *Define key concepts and tools of time-weighted control charts*
2. *Recognize the need for time-weighted charts*
3. *Reiterate different phases of chart application*
4. *Compare and contrast Shewhart and time-weighted control charts*
5. *Design and apply CUSUM control charts*
6. *Design and apply EWMA control charts*
7. *Design and apply MA control charts*
8. *Evaluate the stability of a process using time-weighted control charts*
9. *Implement time-weighted control charts using Excel, Python, and Minitab*
10. *Regulate a process using an EWMA model*

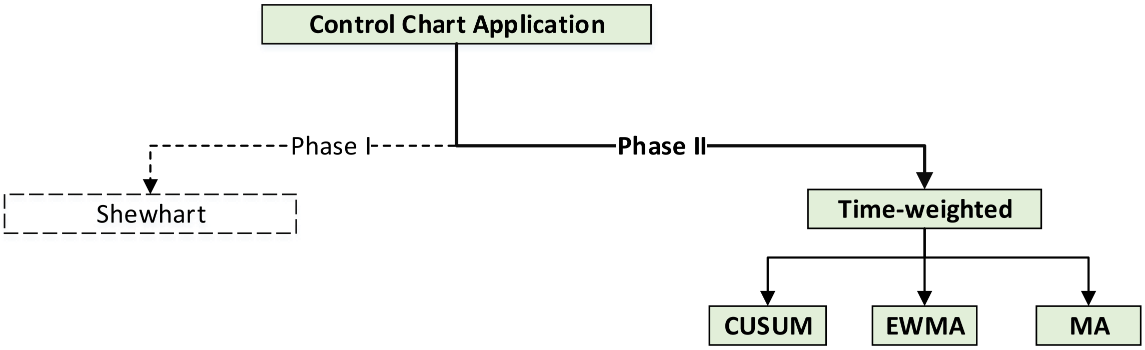
3.1 Introduction

In this chapter, we discuss the time-weighted control charts. We typically implement these charts during **phase II** of the control chart application to detect **small shifts** in the process. We recall that we apply Shewhart charts in phase I to detect big shifts and stabilize the process.

As the name implies, time-weighted charts involve *weighing* data over *time*. In this case, we weigh each sample given the time when that sample was collected. This procedure permits us to incorporate past information into the current process performance measure, which facilitates the detection of small shifts in the process over time. Here, we only consider the three most common types of time-weighted charts, namely **CUSUM, EWMA,** and **MA**. These charts are applicable to both variable and attribute processes when samples are independent. When samples are dependent, we can, for instance, apply EWMA techniques for autocorrelated data [44]. Besides, we can also employ EWMA concepts to **regulate a process**, as we demonstrate later.

Figure 3.1 depicts a basic decision tree for selecting time-weighted charts by the phase of control chart application.

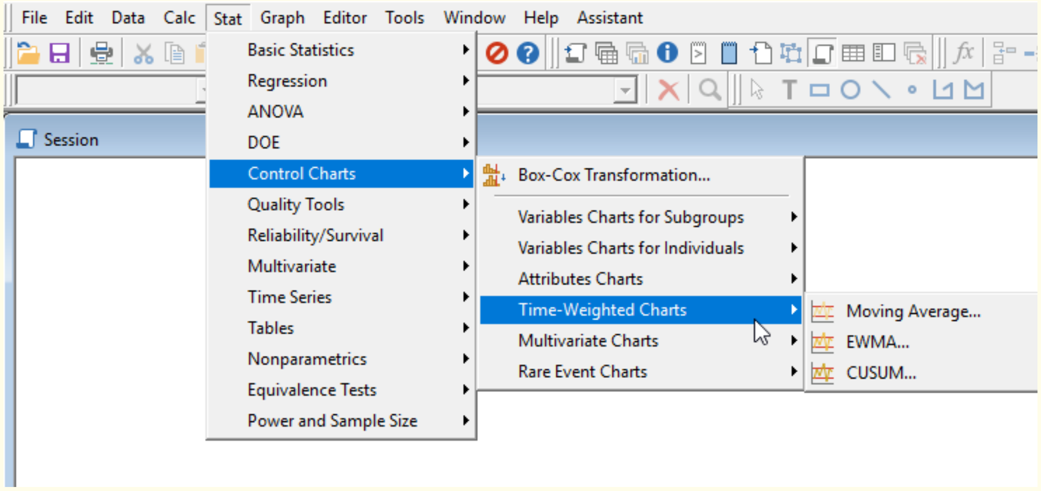
Figure 3.1: Selecting a time-weighted chart by the phase of control chart application



Like in Shewhart charts, we detect out-of-control behaviors in time-weighted charts by finding points that fall outside of the control limits. But, unlike in traditional Shewhart charts, special cause variation in a time-weighted chart is not confined to the sample that falls outside of the limits since this event is due to the cumulative effect of multiple samples. So, when investigating the source of special cause variation, we have to **begin from the sample when the process first started to drift away from the target**. After removing all assignable causes, we reset the control charts to initial conditions and resume monitoring the process. We typically do not apply sensitizing rules with time-weighted charts, but we could still look for obvious abnormalities, such as points that hover too close to the control limits. In How-To 3.1, we present general instructions for accessing time-weighted charts options in Minitab 18. When discussing each chart, we will present specific instructions as well as a Python script to create the chart of interest. We will also show how to set up time-weighted charts using Microsoft Excel.

How-To 3.1 (Time-weighted charts in Minitab 18)
Click on Stat > Control Charts > Time-Weighted Charts > Select the chart of interest. See the snapshot in Figure 3.2.

Figure 3.2: A menu of time-weighted charts in Minitab 18



Before diving into specifics of time-weighted charts, we first present possible scenarios where these charts may be applicable in health care processes.

Scenario 1: Last year, General Hospital implemented a CPOE[1] system. Over the past few months, the CIO of this hospital has worked hard to stabilize the order entry process using p-charts. The CPOE system is now compliant with the Centers for Medicare and Medicaid Services (CMS) stage 3 meaningful use, given that more than 60% of orders for medication, laboratory, and diagnostic imaging are entered by licensed professionals using this system [16]. The CIO's next plan is to apply CUSUM charts to attempt to detect small shifts in the new process.

Scenario 2: CMS mandates the reporting of several quality measures, including the time to administer antibiotics to pneumonia patients and the time to give aspirin to heart attack patients [12]. The quality manager at Metropolis Hospital, who is in charge of data reporting, used ImR charts to stabilize the process of the timely reporting of all required quality measures. Next, the manager will implement CUSUM charts to monitor small shifts in the new process.

Scenario 3: The manager of Metropolis Medical Practice monitors the productivity of physicians. One measure of productivity that the manager uses is the relative value unit (RVU). Given that the process has been stable for the past several months, the manager has decided to apply MA charts to monitor small shifts in the future

[1] **CPOE:** computerized provider order entry

process. For further discussion about the use of RVUs to monitor productivity in medical practices, see Stewart (2002) [60] and Thor et al. (2007) [61].

Scenario 4: Metropolis Hospital has recently implemented an EHR system with a medium for documenting clinical problem lists using SNOMED-CT[2]. Currently, the hospital is using this option to record the emergency department (ED) chief complaints. But, providers still have an option to use free-text. The goal of the hospital is to minimize the utilization of free-text to promote data integrity and consistency. Accordingly, the ED manager has proposed using c-charts to monitor the daily number of times the free-text alert is overwritten. After stabilizing the process, the manager intends to implement EWMA charts for phase II of the control chart application. To read more about the use of control charts to monitor the documentation of chief complaints in the ED, see Aronsky et al. (2001) [4].

3.2 CUSUM charts

We apply CUSUM charts to monitor cumulative deviations C_i from the target μ_0. We determine C_i this way:

$$\begin{aligned} C_i &= \sum_{j=1}^{i}(x_j - \mu_0) && (3.1) \\ &\equiv x_i - \mu_0 + C_{i-1} && (3.2) \end{aligned}$$

where x_i is the statistic of interest in sample i, for $i : 1, \ldots, m$, and m is the number of samples. It is common practice to apply CUSUM charts to monitor individual observations when the sample size $n = 1$. When $n > 1$, we monitor $\bar{x}_i$, the mean of sample i. We assume that all samples are independent. If the process is in control, C_i will fluctuate around zero. If C_i continuously drifts away from the target until it exceeds the limits, the special cause variation likely has occurred.

Formulation

Box 3.1 summarizes the formulas for a type of CUSUM charts known as **Tabular** or **Algorithmic**. **V-mask** is another common type of CUSUM charts, but we do not consider it here [44].

[2] **SNOMED-CT:** Systematized Nomenclature of Medicine -Clinical Terms

Box 3.1 Tabular CUSUM charts for mean behaviors

We set the control limits of a tabular CUSUM chart to $\pm H$, where $H = h\sigma$ and h is a constant set between 4 and 5 to ensure good in-control ARL performance. As before, σ denotes the process standard deviation, and we can estimate it from the process samples. When $h = 4.77$, $ARL \approx 370$, which corresponds to the performance of an in-control 3σ Shewhart control chart. In CUSUM charts, we monitor two statistics given by C_i^+ and C_i^-. The formulas of these quantities are as follows:

$$C_i^+ = \max\left[0, x_i - \mu_0 - k + C_{i-1}^+\right] \tag{3.3}$$

$$C_i^- = \min\left[0, x_i - \mu_0 + k + C_{i-1}^-\right] \tag{3.4}$$

where $C_0^+ = C_0^- = 0$ and k is the slack value generally set to half the shift that we want to detect. So, if we wanted to detect a shift of one standard deviation, we would set $k = 1/2$. In practice, it is common to standardize CUSUM charts by replacing $x_i - \mu_0$ in Equations 3.3 and 3.4 with y_i given by [44]:

$$y_i = \frac{x_i - \mu_0}{\sigma} \tag{3.5}$$

To monitor variability, we utilize a scale CUSUM chart formulated in Box 3.2.

Box 3.2 A scale CUSUM chart for variability

To monitor variability using CUSUM charts, we track the statistics of S_i^+ and S_i^- given by:

$$S_i^+ = \max\left[0, v_i - k + C_{i-1}^+\right] \tag{3.6}$$

$$S_i^- = \min\left[0, v_i + k + C_{i-1}^-\right] \tag{3.7}$$

Here, $S_0^+ = S_0^- = 0$ and

$$v_i = \frac{\sqrt{|y_i| - 0.822}}{0.349} \tag{3.8}$$

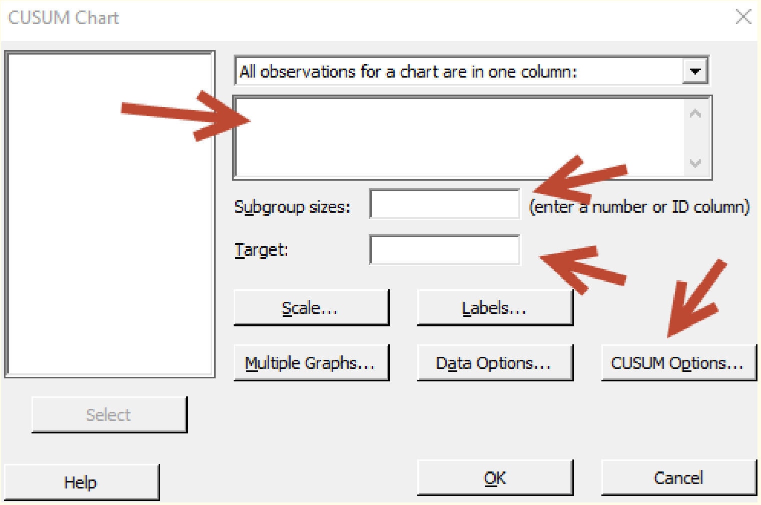
where y_i is as defined in Equation 3.5 [32]. We set the control limits and the slack value k as previously described in Box 3.1 [44].

How-To 3.2 (Tabular CUSUM options in Minitab 18)

1. Click on CUSUM as portrayed in How-To 3.1
2. *Input the subgroup size > input the target > click on CUSUM Options* as

shown in Figure 3.3.

Figure 3.3: CUSUM chart options in Minitab 18, the main screen



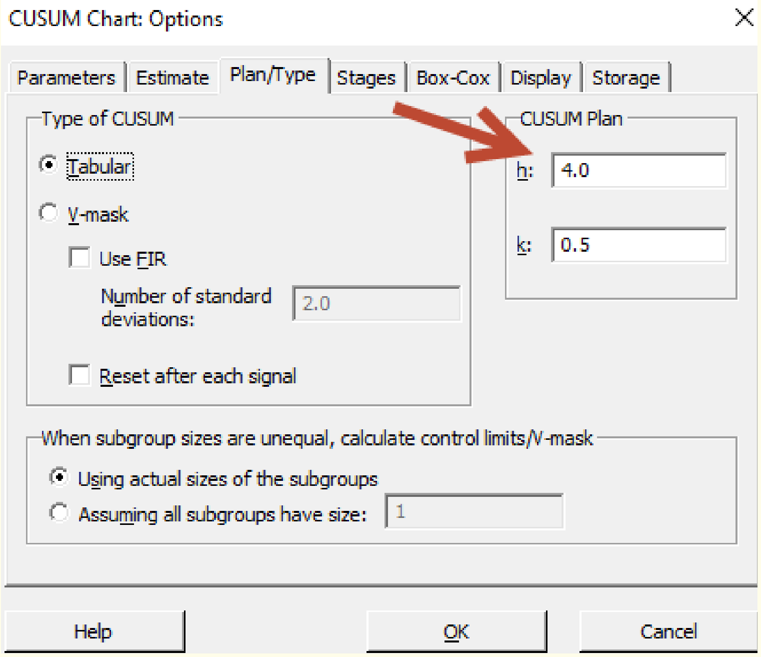
All observations for a chart are in one column:

Subgroup sizes: (enter a number or ID column)

Target:

3. *Click on tab Plan/Type > change the* h *and* k *values* as necessary. Make sure the Tabular type is selected. See the snapshot in Figure 3.4.

Figure 3.4: CUSUM chart options in Minitab 18, Plan/Type tab



Type of CUSUM

● Tabular

○ V-mask

□ Use FIR

Number of standard deviations: 2.0

□ Reset after each signal

CUSUM Plan

h: 4.0

k: 0.5

When subgroup sizes are unequal, calculate control limits/V-mask

● Using actual sizes of the subgroups

○ Assuming all subgroups have size: 1

4. Click $OK > OK$ to create your chart.

How-To 3.3 (Python 3.6)

Script 3.1: A script for creating a tabular CUSUM chart using Python 3.6

```
#TIME WEIGHTED CHARTS CUSUM
from pandas import*
from pylab import*
from numpy import*
import seaborn as sns
data = read_excel(’your directory’)
#parameters
L = 3.
lmda = 0.2
h= 5.
d2 = 1.128
xr = data.xt
mrbar = mean([abs(data.xt[i] -data.xt[i-1]) for i in range(1,len(data
    .xt))])
sigma = mrbar/d2
K = sigma*0.5
target = 10.# for python2
xbar = target #mean(xr)
#set up control limits
c1 =[0.]
c2 = [0.]
for i in range(len(xr)):
    c1i = max(0, xr[i] - (xbar + K) + c1[i])
    c2i = min(0,  xr[i]-xbar + K +c2[i])
    c1.append(c1i)
    c2.append(c2i)
UCL = [h*sigma]*len(xr)
LCL = [-h*sigma]*len(xr)
CL = [0]*len(xr)
#mark red a point that falls outside of the control limits. Otherwise
    , mark the point blue
markers = []
colors = []
markers1 = []
colors1 = []
for i in range (len(data)):
    x1 = c1[i+1]
    if x1 > UCL[i]:
        markers.append(’o’)
        colors.append(’r’)
    else:
        markers.append(’o’)
        colors.append(’b’)
for i in range (len(data)):
    x2 = c2[i+1]
    if  x2 <LCL[i]:
        markers1.append(’D’)
        colors1.append(’r’)
    else:
        markers1.append('D')
        colors1.append('g')
#Plotting CUSUM chart
fig=figure()
ax1 =  fig.add_subplot(111)
ax1.plot(UCL, 'k-', alpha = 0.5)
ax1.plot(LCL, 'k-',alpha = 0.5)
ax1.plot(CL, 'k-',alpha = 0.5)
ax1.plot(c1[1:],'b-',zorder=1)
ax1.plot(c2[1:],'g-',zorder=1)
for x,y,c,m in zip(t, c1[1:], colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
for x,y,c,m in zip(t, c2[1:], colors1, markers1):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel('Sample number')
ax1.set_ylabel('CUSUM')
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL=$'+str(round(UCL[0],2)), xy = (xlim()[1], list(
   UCL)[-1]), xytext = (xlim()[1],list(UCL)[-1]),fontsize = 11)
ax1.annotate (''+str(round(abs(CL[0]),2)), xy = (xlim()[1], list(CL)
   [-1]), xytext = (xlim()[1],list(CL)[-1]),fontsize = 11)
ax1.annotate ('$LCL=$'+str(round(LCL[0],2)), xy = (xlim()[1], list(
   LCL)[-1]), xytext = (xlim()[1],list(LCL)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(xr), step = 2), arange(1, len(xr)+1, step = 2))
show()
```

How-To 3.4 (A tabular CUSUM chart in Excel 2013)
Excel does not have a built-in option for creating a tabular CUSUM chart, but we can manually program the formulas in Box 3.1 as we demonstrate in Example 3.1.

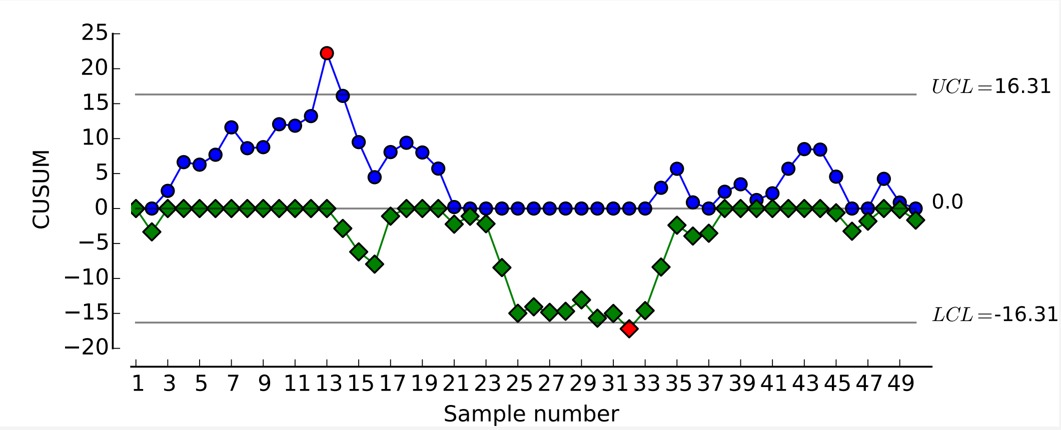
Example 3.1 (A CUSUM chart for the process in scenario 1)
Let us reconsider scenario 1 about General Hospital and a CPOE process. The CIO has transitioned to phase II, and Table 3.1 presents data for the most recent entry times for radiology orders.

Table 3.1: Entry times in minutes (x_t) for radiology orders at General Hospital

Sample#	x_t	*Sample#*	x_t
1	9.63	*26*	9.29
2	5.03	*27*	7.58
3	14.16	*28*	8.52
4	15.74	*29*	10
5	11.27	*30*	5.75
6	13.05	*31*	9.05
7	15.55	*32*	6.17
8	8.65	*33*	10.97
9	11.78	*34*	14.6
10	14.91	*35*	14.35
11	11.42	*36*	6.82
12	13.01	*37*	8.77
13	20.63	*38*	14.03
14	5.52	*39*	12.7
15	5.02	*40*	9.38
16	6.61	*41*	12.59
17	15.23	*42*	15.15
18	12.96	*43*	14.44
19	10.24	*44*	11.56
20	9.33	*45*	7.76
21	6.13	*46*	5.73
22	9.5	*47*	9.8
23	7.3	*48*	15.88
24	2.1	*49*	8.24
25	1.83	*50*	6.84

To help the CIO create a CUSUM chart for phase II, we applied the Python script in How-To 3.3 and generated the chart portrayed in Figure 3.5.

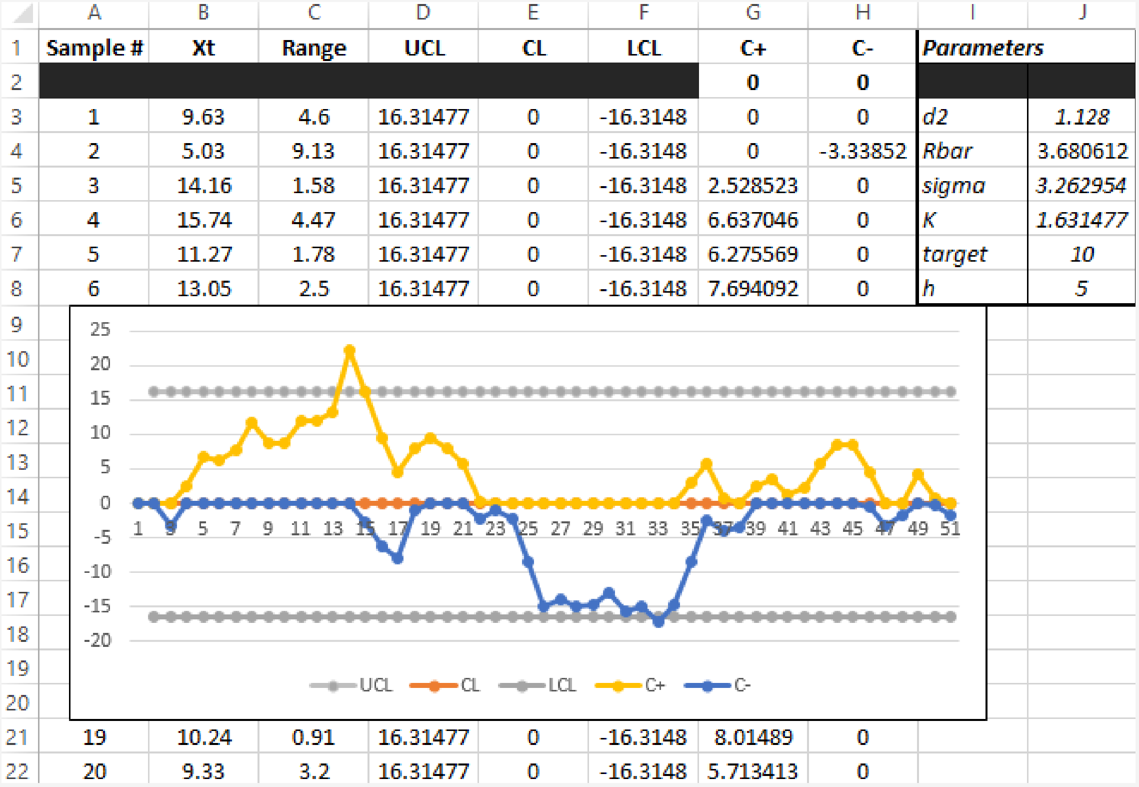
Figure 3.5: A CUSUM chart created using Python 3.6 based on the data in Table 3.1



The CUSUM chart in Figure 3.5 indicates that samples 13 and 32 fall outside of the control limits. To investigate the causes of these behaviors, the CIO needs to begin from the sample where the cumulative sum started to drift away from 0. For the case of sample 13, the CIO should start an investigation from sample 3. For the case of sample 32, the manager should go back to sample 22 to find the reason for out-of-control behaviors. After removing all assignable causes, the CIO should reset the chart to zero and resume monitoring the process. In practice, out-of-control behaviors should be investigated as soon as they occur, and the charts reset thereafter.

To reproduce the CUSUM chart in Figure 3.5 using Excel, we first set up our spreadsheet, as shown in Figure 3.6. In our setup, we left row 2 blank, except for cells G2 and H2, to initialize $C_0^+ = C_0^- = 0$ per Box 3.1.

Figure 3.6: A setup of Excel to create a CUSUM chart based on the data in Table 3.1



	A	B	C	D	E	F	G	H	I	J
1	Sample #	Xt	Range	UCL	CL	LCL	C+	C-	*Parameters*	
2							0	0		
3	1	9.63	4.6	16.31477	0	-16.3148	0	0	*d2*	*1.128*
4	2	5.03	9.13	16.31477	0	-16.3148	0	-3.33852	*Rbar*	3.680612
5	3	14.16	1.58	16.31477	0	-16.3148	2.528523	0	*sigma*	*3.262954*
6	4	15.74	4.47	16.31477	0	-16.3148	6.637046	0	*K*	*1.631477*
7	5	11.27	1.78	16.31477	0	-16.3148	6.275569	0	*target*	*10*
8	6	13.05	2.5	16.31477	0	-16.3148	7.694092	0	*h*	*5*
9										
10										
11										
12										
13										
14										
15										
16										
17										
18										
19										
20										
21	19	10.24	0.91	16.31477	0	-16.3148	8.01489	0		
22	20	9.33	3.2	16.31477	0	-16.3148	5.713413	0		

The explanation of our spreadsheet setup follows.

1. The target value in this case is $\mu_0 = 10$, as indicated in cell $J7$.

2. To estimate the standard deviation σ, we used the $\bar{R}/d_2$ technique, where $\bar{R}$ is the average of moving ranges and $d_2 = 1.128$ per Appendix Table 12 when $n = 2$. For example, we obtained the moving range value in cell $C3$

by: $= ABS(B3 - B4)$, where $ABS()$ is the absolute value function in Excel. Subsequently, we dragged down the formula in $C3$ until cell $C51$ to populate the rest of the values. After that, we obtained $Rbar$, shown in cell $J4$, by: $= AVERAGE(C3 : C51)$. Our estimate of σ is indicated in cell $J5$ and was obtained by $= J4/J3$.

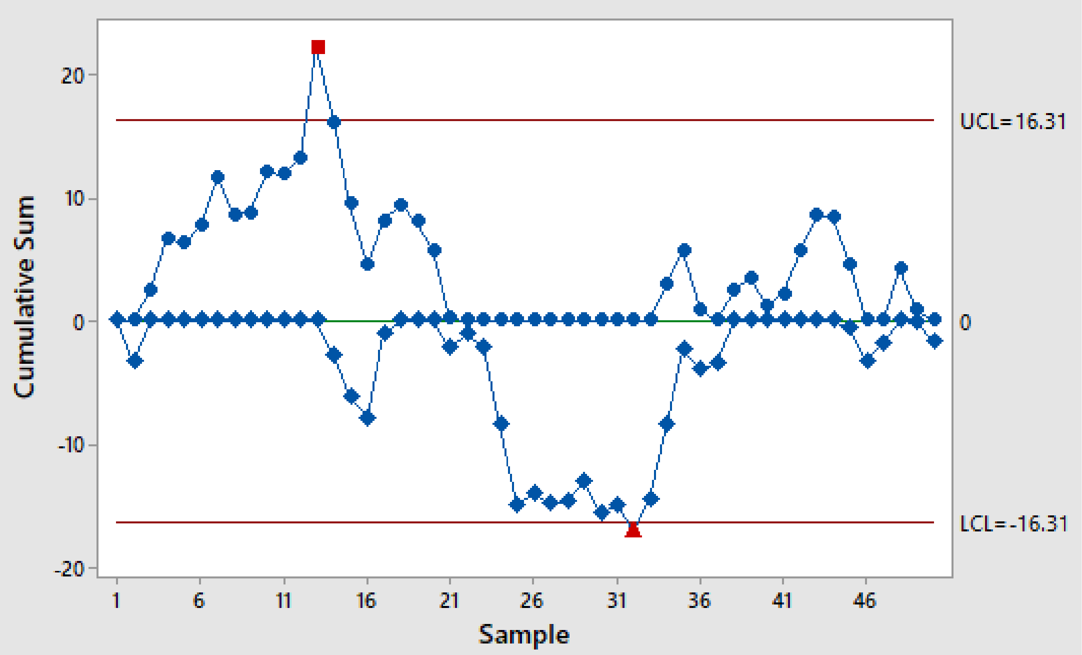
3. We calculated the C_i^+ and C_i^- statistics per Box 3.1. For example, we obtained the value in cell $G3$ by $= MAX(0,(B3-\$J\$7-\$J\$6+G2))$ and the value in cell $H3$ by $= MIN(0, B3 - \$J\$7 + \$J\$6 + H2)$. We dragged down these formulas until row 52 to populate the rest of the values.

4. For control limits, we set CL to 0 and obtained the values of UCL and LCL in cells $D3$ and $F3$ by $= \$J\$8*\$J\$5$ and $= \$J\$8*\$J\$5$, respectively. We dragged down these formulas until row 52.

5. Finally, we created the CUSUM chart of interest by inserting the line charts of columns D, E, F, G, and H.

To reproduce the CUSUM chart in Minitab 18, we set the subgroup size to 1, the target to 10, h to 5, and k to 0.5. The resulting CUSUM chart is portrayed in Figure 3.7. The CUSUM statistics and interpretation are as before.

Figure 3.7: A CUSUM chart created using Minitab 18 based on the data in Table 3.1



3.3 EWMA charts

We create EWMA charts by weighting all process observations or sample statistics of interest. The weights we apply decrease geometrically, thus making the most recent observations weigh more than the earlier ones. Like in CUSUM charts, we typically use EWMA charts to monitor individual observations when the sample size $n = 1$. When $n > 1$, we monitor $\bar{x}_i$, the mean of sample i, for $i : 1, \ldots, m$, where m is the total number of samples.

Formulation

We summarize the formulas of EWMA charts in Box 3.3.

Box 3.3 EWMA charts

In EWMA charts, we monitor the statistic z_i, given by:

$$z_i = \lambda x_i + (1 - \lambda) z_{i-1} \tag{3.9}$$

where $z_0 = \mu_0$ and x_i is the observation from sample i. Here, λ is the discount factor commonly set between 0.2 and 0.4. The smaller λ, the more sensitivity to small shifts in the process. When $\lambda = 1$, the resulting EWMA chart is equivalent to its Shewhart counterpart (e.g., ImR chart). Given sample i, we determine the centerline and limits as follows:

$$UCL_i = \mu_0 + L\sigma\sqrt{\frac{\lambda}{2-\lambda}[1-(1-\lambda)^{2i}]} \tag{3.10}$$

$$CL_i = \mu_0 \tag{3.11}$$

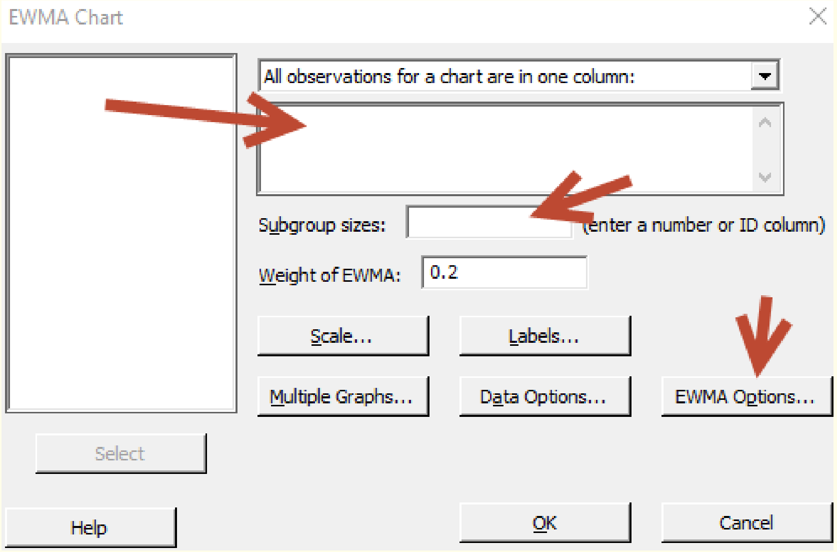
$$LCL_i = \mu_0 - L\sigma\sqrt{\frac{\lambda}{2-\lambda}[1-(1-\lambda)^{2i}]} \tag{3.12}$$

where μ_0 is a constant target. We typically set $L = 3$ and estimate σ using $\overline{MR}/d_2$ for individual observations [45].

How-To 3.5 (EWMA in Minitab 18)

1. Click on EWMA as pictured in How-To 3.1
2. *Input your data > select the subgroup size > click on EWMA options*. Change the default value of λ (Weight of EWMA) as necessary. See the snapshot in Figure 3.8.

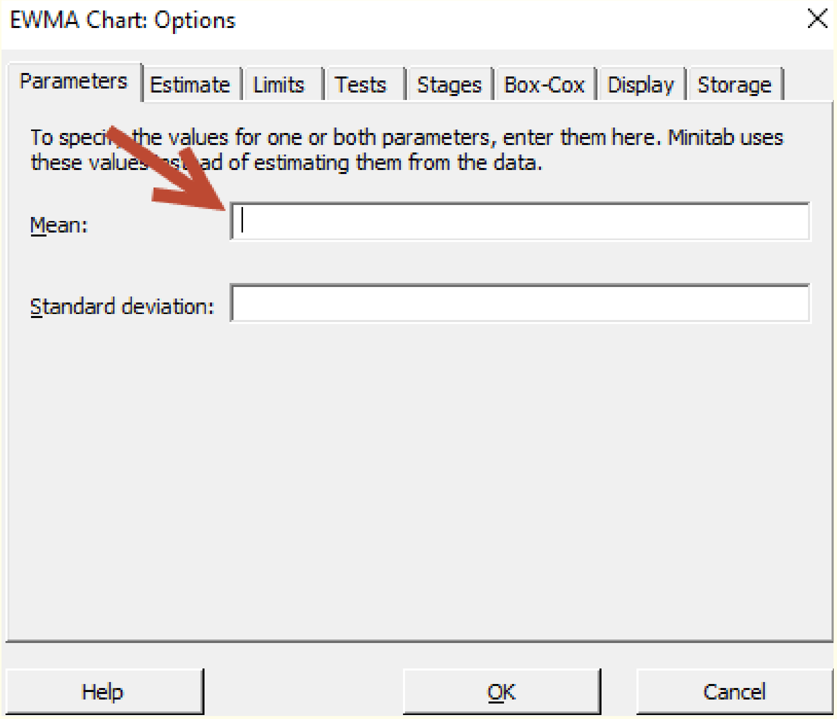
Figure 3.8: EWMA chart options in Minitab 18, the main screen



Weight of EWMA: 0.2

3. *Click on the Parameters tab> input your mean target*, as shown in Figure 3.9

Figure 3.9: EWMA chart in Minitab 18, the Parameters tab



To specify the values for one or both parameters, enter them here. Minitab uses these values instead of estimating them from the data.

How-To 3.6 (Python 3.6)

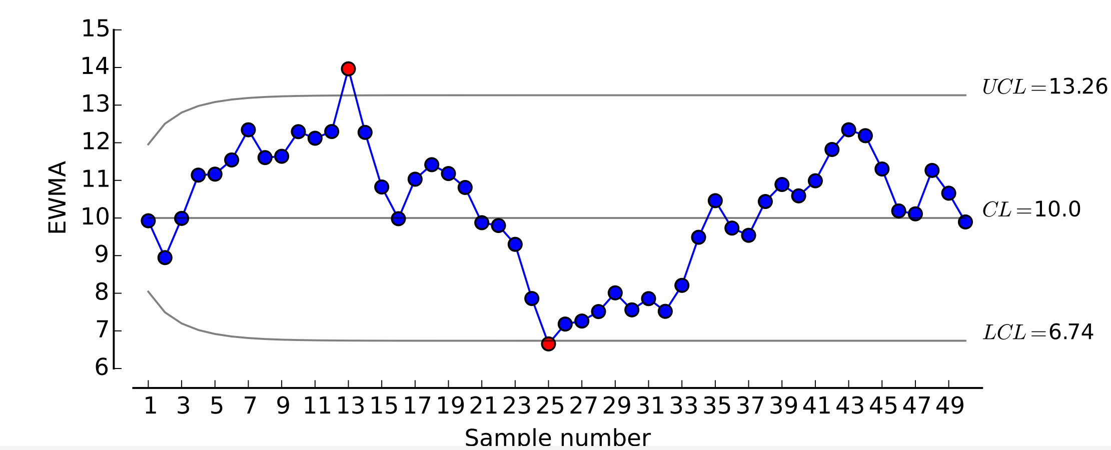
Script 3.2: A script for creating an EWMA chart using Python 3.6

```
##TIME WEIGHTED CHARTS EWMA chart
#Import modules
from pandas import*
from pylab import*
from numpy import*
import seaborn as sns
#import data from an excel spreadsheet
data = read_excel('your directory')
#parameters
L = 3.
lmda = 0.2
d2 = 1.128
mrbar = mean([abs(data.xt[i] -data.xt[i-1]) for i in range(1,len(data
    .xt))])
target = 10.
xbar = target
sigma = mrbar/d2
term1 = lmda/(2.-lmda)
term2 = (1.-lmda)
xr = []
xr0 = data.xt
#set up control charts
UCL = [xbar + L*sigma*sqrt(term1*(1. - term2**(2*i))) for i in range
    (1, len(xr0)+1)]
LCL = [xbar - L*sigma*sqrt(term1*(1. - term2**(2*i))) for i in range
    (1, len(xr0)+1)]
CL = [xbar]*len(xr0)
zv = [xbar]
for i in range(len(xr0)):
    z = lmda*xr0[i] + (1.-lmda)*(zv[i])
    zv.append(z)
    xr.append(z)
#mark red a point that falls outside of the control limits. Otherwise
    , mark the point blue
markers = []
colors = []
for i in range (len(xr0)):
    x1 = xr[i]
    x2 = UCL[i]
    x3 = LCL[i]
    if (x1 > x2 or  x1<x3) :
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
#Plotting EWMA chart
fig=figure()
ax1 =  fig.add_subplot(111)
t = arange(len(data))
ax1.plot(UCL, 'k-', alpha = 0.5)
ax1.plot(LCL, 'k-',alpha = 0.5)
ax1.plot(CL, 'k-',alpha = 0.5)
```

```
ax1.plot(xr,'b-',zorder=1)
for x,y,c,m in zip(t, xr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel('Sample number')
ax1.set_ylabel('EWMA')
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL=$'+str(round(UCL[-1],2)), xy = (xlim()[1], list(
    UCL)[-1]), xytext = (xlim()[1],list(UCL)[-1]),fontsize = 11)
ax1.annotate ('$CL=$'+str(round(CL[0],2)), xy = (xlim()[1], list(CL)
    [-1]), xytext = (xlim()[1],list(CL)[-1]),fontsize = 11)
ax1.annotate ('$LCL=$'+str(round(LCL[-1],2)), xy = (xlim()[1], list(
    LCL)[-1]), xytext = (xlim()[1],list(LCL)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(xr), step = 2), arange(1, len(xr)+1, step = 2))
show()
```

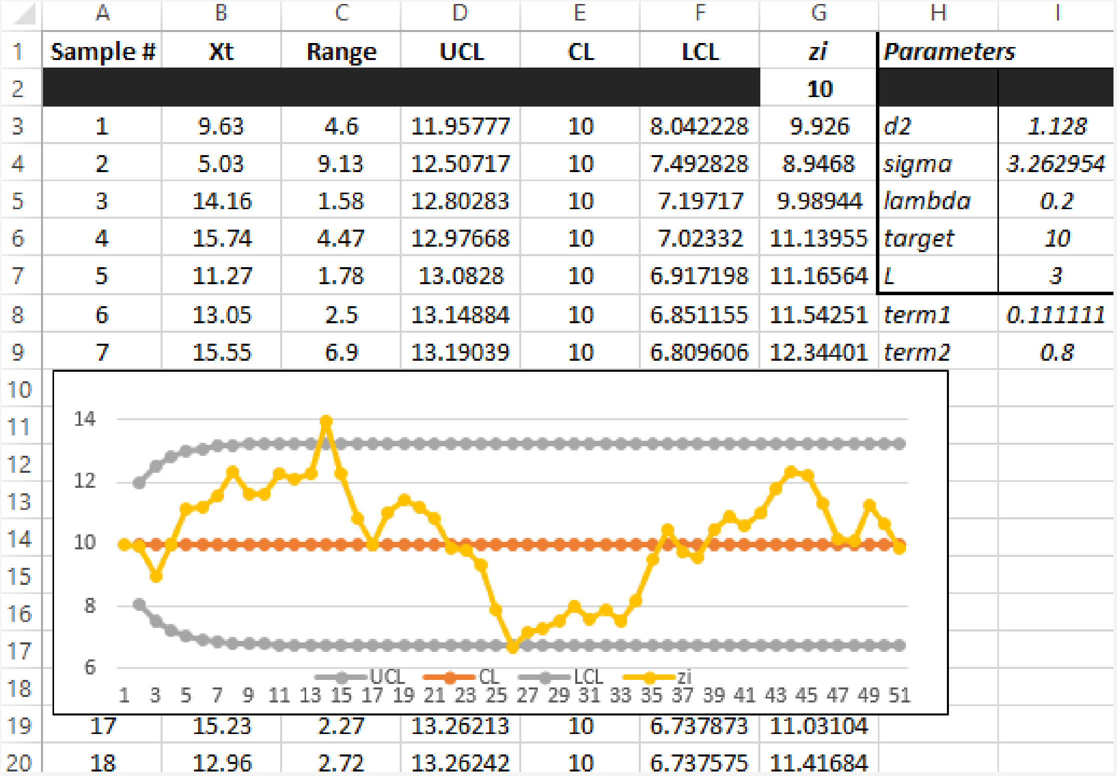
Example 3.2 (An EWMA chart for the process in scenario 1)
In this example, we once again revisit scenario 1 about General Hospital and the CPOE process. We use the data in Table 3.1 and apply the Python script in How-To 3.6 to generate the EWMA chart portrayed in Figure 3.10.

Figure 3.10: An EWMA chart created in Python 3.6 based on the data in Table 3.1



To this EWMA chart in Excel, we begin by setting up our spreadsheet, as illustrated in Figure 3.11.

Figure 3.11: A setup of Excel to create an EWMA chart based on the data in Table 3.1



	A	B	C	D	E	F	G	H	I
1	Sample #	Xt	Range	UCL	CL	LCL	zi	*Parameters*	
2							10		
3	1	9.63	4.6	11.95777	10	8.042228	9.926	*d2*	*1.128*
4	2	5.03	9.13	12.50717	10	7.492828	8.9468	*sigma*	*3.262954*
5	3	14.16	1.58	12.80283	10	7.19717	9.98944	*lambda*	*0.2*
6	4	15.74	4.47	12.97668	10	7.02332	11.13955	*target*	*10*
7	5	11.27	1.78	13.0828	10	6.917198	11.16564	*L*	*3*
8	6	13.05	2.5	13.14884	10	6.851155	11.54251	*term1*	*0.111111*
9	7	15.55	6.9	13.19039	10	6.809606	12.34401	*term2*	*0.8*
19	17	15.23	2.27	13.26213	10	6.737873	11.03104		
20	18	12.96	2.72	13.26242	10	6.737575	11.41684		

In our Excel setup, we left the second row blank to initialize $z_0 = \mu_0$. We implemented the following steps to recreate the EWMA chart of interest.

1. We set the λ parameter in cell $I5$ to 0.2, the target parameter in cell $I6$ to 10, and the L parameter in cell $I7$ to 3.

2. We estimated σ as in Example 3.1.

3. We set up $term1$ in cell $I8$ as $= I5/(2 - I5)$. This term is equivalent to $\left(\frac{\lambda}{2-\lambda}\right)$.

4. We set up $term2$ in cell $I9$ as $= 1 - I5$. This term is equivalent to $(1 - \lambda)$.

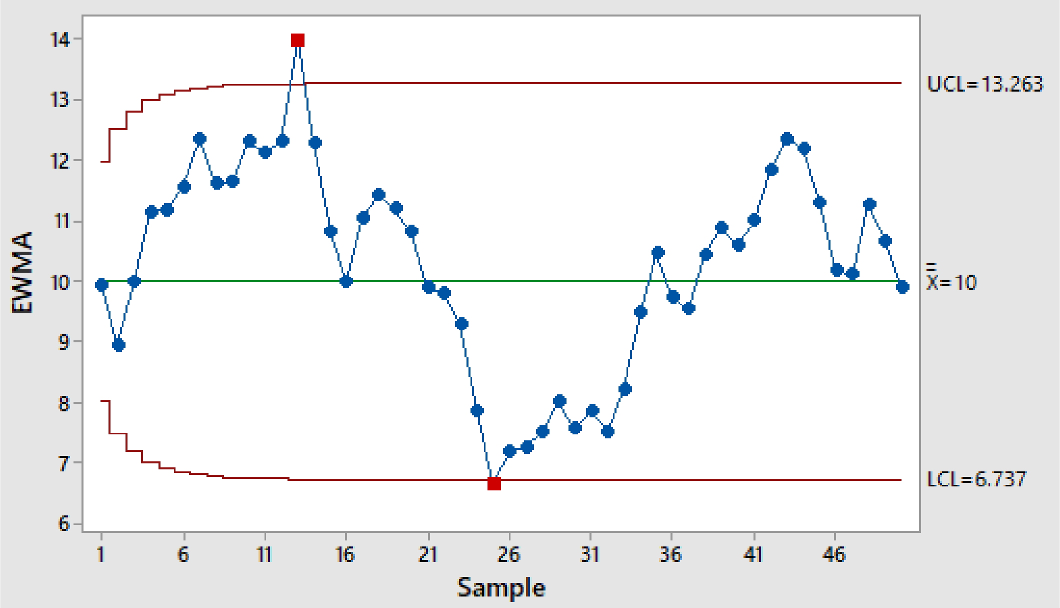
5. We computed the z_i statistic per Box 3.3. For example, we determined the z_i value in $G3$ by $= \$I\$5 * B3 + (1 - \$I\$5) * G2$. We dragged down this formula to populate the rest of the values.

6. By this step, we can set up the control limits. For example, we obtained the values of UCL and LCL in cells $D3$ and $F3$ using $= \$I\$6 + \$I\$7 * \$I\$4 * SQRT(\$I\$8 * (1 - \$I\$9(2 * A3)))$ and $= \$I\$6 - \$I\$7 * \$I\$4 * SQRT(\$I\$8 * (1 - \$I\$9(2 * A3)))$, respectively. We dragged down these formulas to populate the rest of the values. The CL value is set to the target of 10.

7. We plotted the EWMA control chart by inserting the line charts of columns D through G.

To reproduce the same chart in Minitab, we set the subgroup size of 1 and the mean parameter of 10. The resulting chart is portrayed in Figure 3.12.

Figure 3.12: An EWMA chart produced in Minitab based on the data in Table 3.1



Like in the CUSUM chart, the EWMA chart shows two areas of out-of-control behaviors. But, unlike in the CUSUM chart, the EWMA chart indicates that the second out-of-control sample is 25, not 32. Still, to assign causes of the special cause variation, we must go back to where the process started to drift away from the target. As before, the CIO needs to find and remove all assignable causes. Subsequently, the CIO will have to reset the EWMA control chart before resuming to monitor the process.

3.3.1 Process regulation using EWMA

So far, we have assumed that we can control a process by finding and removing special cause variation manually. This approach works well in discrete and stable cases where out-of-control behaviors occur occasionally. In continuous processes where disturbances could be frequent, regulating a process manually would be a tedious task. But, if we could somehow find a variable to compensate for these disturbances, we might be able to regulate the process automatically. In health care, most of the processes we work with are discrete, and we can often control them manually. Although, a few processes are candidates for automatic control such as drug infusions [13, 41] and fast-paced processes like queues in the emergency department [46, 47].

An integral control model

Process regulation takes many forms, including fuzzy [37] and Proportional-Integral-Derivative (PID) techniques [13, 41]. Here, we briefly discuss the **integral** part of PID [8, 44]. We begin by imagining y to be the process output variable that we are trying to regulate using the input variable x called the **set point**. Our goal is to adjust variable x at time t now so y remains close to the process target T at time $t+1$. If N is the amount of expected disturbance or drift, then at time $t+1$ the following mathematical relationship holds [44]:

$$y_{t+1} - T = N_{t+1} \tag{3.13}$$

We assume that we can use the EWMA function with a discount factor λ to predict N_{t+1} using $\hat{N}_{t+1}$, as follows:

$$\begin{aligned}
\hat{N}_{t+1} &= \lambda N_t + (1-\lambda)\hat{N}_t && (3.14)\\
&= \lambda N_t + \hat{N}_t - \lambda \hat{N}_t && (3.15)\\
&= \hat{N}_t + \lambda(N_t - \hat{N}_t) && (3.16)\\
&= \hat{N}_t + \lambda e_t && (3.17)
\end{aligned}$$

where $\hat{N}_t$ is the drift that we predicted at $t-1$, but the actual drift turned out to be N_t, and e_t is given by:

$$\begin{aligned}
e_t &= N_t - \hat{N}_t && (3.18)\\
&\equiv y_t - T && (3.19)
\end{aligned}$$

Our goal is to cancel out the predicted drift $\hat{N}_{t+1}$ as follows:

$$g x_t = -\hat{N}_{t+1} \tag{3.20}$$

where g is a **process gain** that expresses how much $\hat{N}_{t+1}$ changes for each unit change in the input x_t. It follows that [44]:

$$x_t - x_{t-1} = -\frac{\lambda}{g} e_t \equiv -\frac{\lambda}{g}(y_t - T) \tag{3.21}$$

The cumulative adjustments that we made, up to time t, are given by the following **integral** controller:

$$x_t = -\frac{\lambda}{g}\sum_{i=1}^{t} e_i \equiv -\frac{\lambda}{g}\sum_{i=1}^{t}(y_i - T) \tag{3.22}$$

Example 3.3 (Integral Control)
Table 3.2 contains samples of a clinical process y_t taken every 10 minutes. The target for this process is $T = 100$ and the gain is given by $g = 2$. The discount factor is set to $\lambda = 0.4$.

Table 3.2: Samples of a clinical process taken every 10 minutes

t	y_t	t	y_t	t	y_t	t	y_t
1	96	26	113	51	124	76	155
2	111	27	96	52	122	77	140
3	93	28	92	53	110	78	138
4	110	29	114	54	122	79	130
5	116	30	126	55	125	80	141
6	90	31	97	56	132	81	145
7	116	32	114	57	140	82	180
8	93	33	123	58	132	83	148
9	116	34	123	59	136	84	165
10	107	35	123	60	139	85	163
11	82	36	110	61	143	86	174
12	111	37	143	62	139	87	143
13	87	38	121	63	113	88	174
14	110	39	118	64	110	89	149
15	85	40	145	65	135	90	177
16	98	41	150	66	115	91	162
17	94	42	139	67	133	92	158
18	119	43	117	68	134	93	158
19	92	44	116	69	154	94	155
20	128	45	129	70	144	95	179
21	104	46	145	71	133	96	146
22	126	47	117	72	140	97	166
23	125	48	120	73	142	98	155
24	119	49	125	74	144	99	175
25	124	50	148	75	151	100	159

A line chart of y_t over time t is portrayed in Figure 3.17. From this chart, it is clear that the process quickly drifts away from the target over time. After implementing an automatic regulator, we were able to control the process, as shown in Figure 3.14.

Figure 3.13: Unregulated process y_t with a target $T = 100$

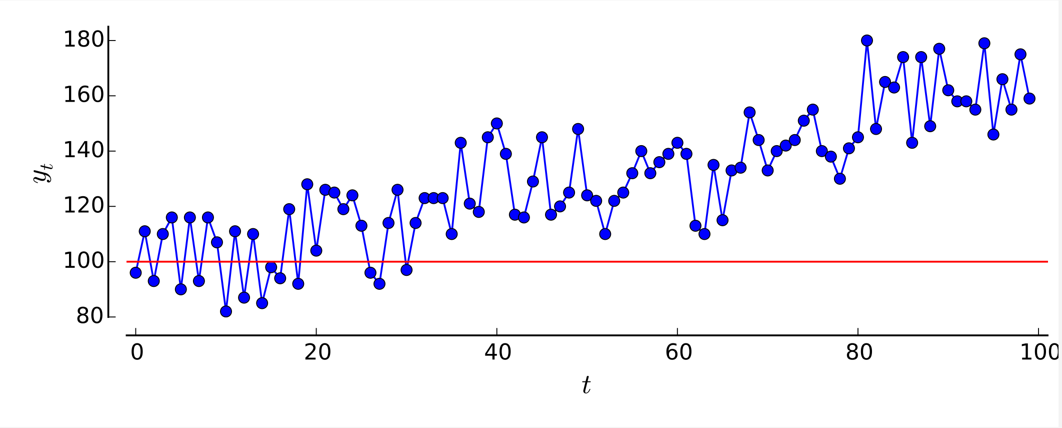
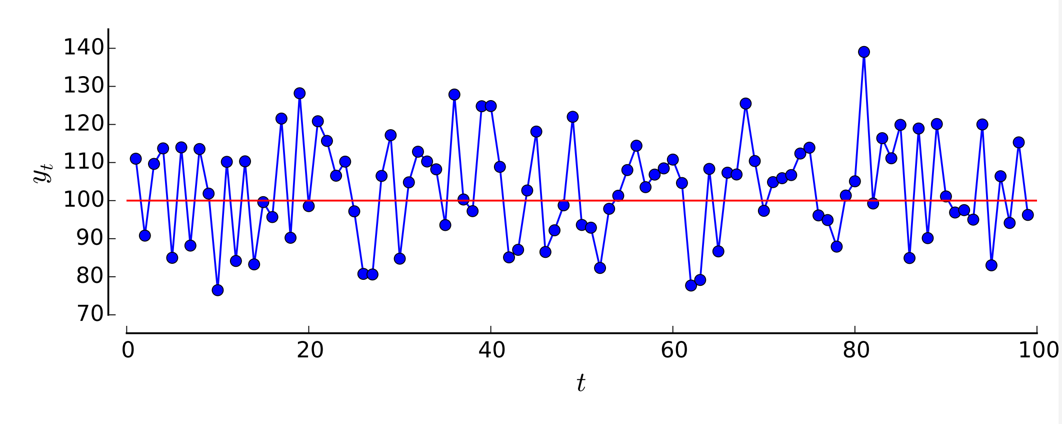
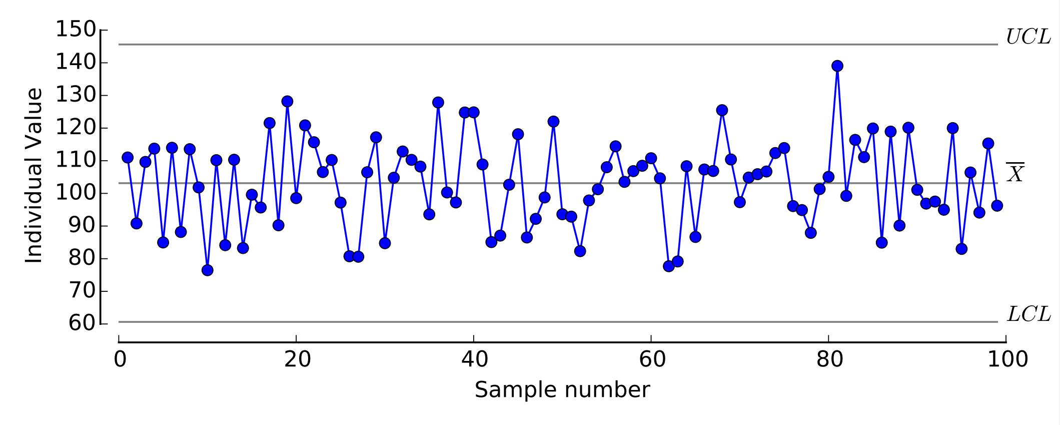


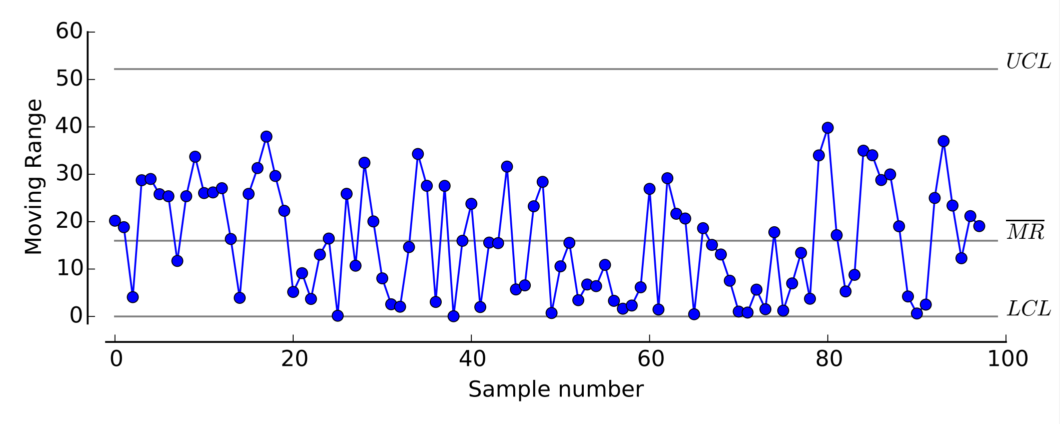
Figure 3.14: Regulated process y_t with a target $T = 100$



We created I-MR control charts of the regulated process as illustrated in Figure 3.15. From these charts, we appreciate that the process is statistically stable.

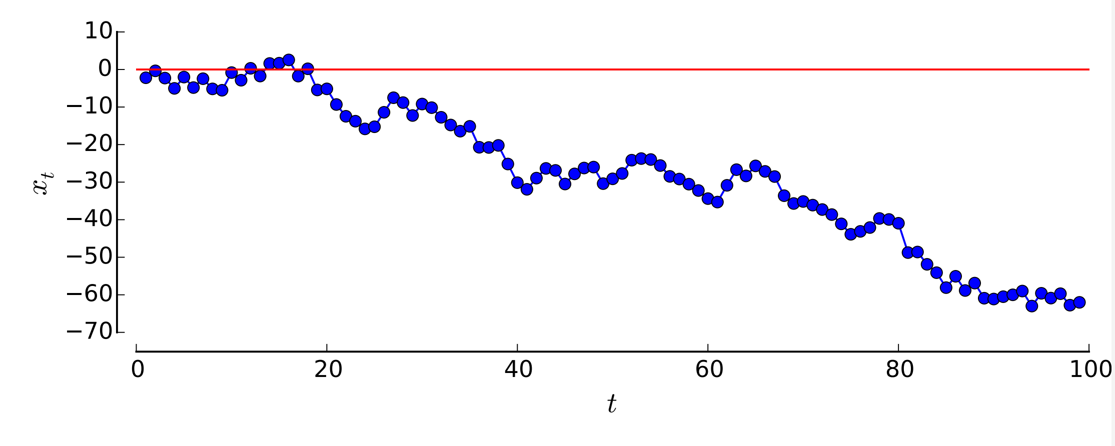
Figure 3.15: The control charts of the regulated process $\hat{y}_t$


(a) *I-chart*


(b) *MR-chart*

The dynamics of the integral controller x_t are plotted in Figure 3.16.

Figure 3.16: The dynamics of the integral controller x_t



From the dynamics of x_t in Figure 3.16, it is clear that regulating the process, manually, is tedious. But, it is still possible to design a manual controller using an **adjustment chart** created using the following equation:

$$x_t - x_{t-1} = -\frac{0.4}{2}(y_t - 100) = -\frac{1}{5}(y_t - 100) \tag{3.23}$$

From Equation 3.23, it follows that the operator would change x_t by one unit per 5 units deviations from the target. The change would be in the direction that brings y_t closer to the target T.

We used Excel to create the **integral** controller of the process y_t. To start, we set up our spreadsheet, as indicated in Figure 3.17. A detailed explanation follows.

1. Initial columns are A, B, G, and H. Column A shows the increments of time t. Column B stores the values of the unregulated process denoted as y_t. We store our parameters in columns G and H.

2. The derived columns are D, E, and F. Column D contains the running differences of the setpoints, $x_t - x_{t-1}$. Column E contains the cumulative values of the setpoint x_t. Column F shows the regulated values denoted as $\hat{y}_t$

3. We initialize the value of the regulated process in cell $F3$ as follows: $\hat{y}_2 = y_2 = B3$.

4. At $t = 1$, $x_1 = 0$.

5. At $t = 2$, $x_2 = x_2 - x_1$ since $x_1 = 0$. Per Equation 3.21, $x_t - x_{t-1} = -\frac{\lambda}{g}(y_t - T)$. So, at $t = 2$, $x_t - x_{t-1} = -\frac{\lambda}{g}(y_t - T) = -\frac{0.4}{2}(111 - 100) = -\frac{1}{5}(111 - 100) = -2.2$. In our Excel spreadsheet, this is programmed in cell $D3$ as: $= -(\$H\$1/\$H\$2) * (F3 - \$H\$3)$.

6. The value of $\hat{y}_3$ in cell $F4$ was obtained using $\hat{y}_t = y_t + x_t$, which in our spreadsheet corresponds to $= B4 + E3$.

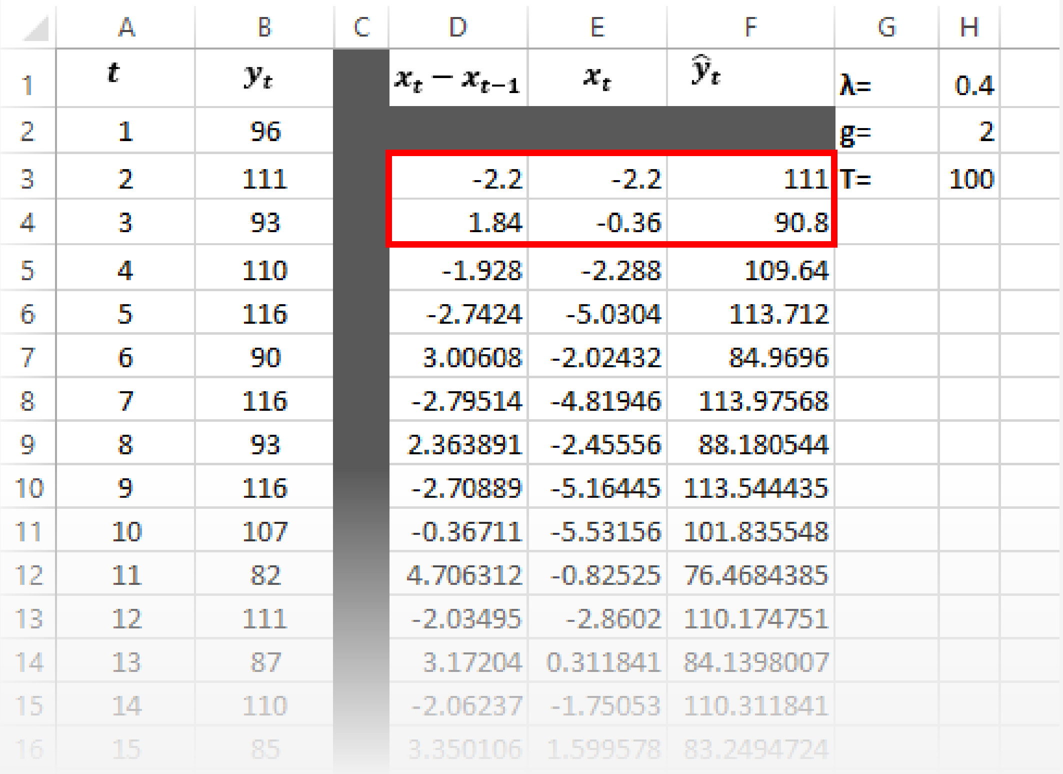
7. The next value of the setpoint x_3, in cell $E4$, was found by $= D3 + E2$.

8. The value of $x_t - x_{t-1}$ in cell $D4$ was programmed as follows: $= -(\$H\$1/\$H\$2) * (F4 - \$H\$3)$.

9. Now we drag down all formulas in cells $D4$ to $F4$ to populate the rest of the values.

10. Finally, we insert the line chart of column F to recreate the chart shown in Figure 3.14.

Figure 3.17: The setup of Excel for regulating the process y_t



	A	B	C	D	E	F	G	H
1	t	y_t		$x_t - x_{t-1}$	x_t	$\hat{y}_t$	λ=	0.4
2	1	96					g=	2
3	2	111		-2.2	-2.2	111	T=	100
4	3	93		1.84	-0.36	90.8		
5	4	110		-1.928	-2.288	109.64		
6	5	116		-2.7424	-5.0304	113.712		
7	6	90		3.00608	-2.02432	84.9696		
8	7	116		-2.79514	-4.81946	113.97568		
9	8	93		2.363891	-2.45556	88.180544		
10	9	116		-2.70889	-5.16445	113.544435		
11	10	107		-0.36711	-5.53156	101.835548		
12	11	82		4.706312	-0.82525	76.4684385		
13	12	111		-2.03495	-2.8602	110.174751		
14	13	87		3.17204	0.311841	84.1398007		
15	14	110		-2.06237	-1.75053	110.311841		
16	15	85		3.350106	1.599578	83.2494724		

To recreate the chart in Figure 3.16, we would insert a line chart of column E of the spreadsheet pictured in Figure 3.17.

3.4 MA charts

The MA chart is the last kind of time-weighted control charts that we consider here. To set up this chart, we first need to define the moving average span of w. Subsequently, we determine the statistic to be monitored, M_i, and then set up appropriate control limits as highlighted in Box 3.4.

Box 3.4 MA charts

We set up the MA centerline and control limits as follows:

$$UCL = \mu_0 + L\frac{\sigma}{\sqrt{w}} \tag{3.24}$$

$$CL = \mu_0 \tag{3.25}$$

$$UCL = \mu_0 - L\frac{\sigma}{\sqrt{w}} \tag{3.26}$$

where μ_0 is the target, σ is the process standard deviation, w is the span of the moving averages, and L is typically set to 3. When the sample number i is less than w, we set up the control limits using this interval:

$$\mu_0 \pm L\frac{\sigma}{\sqrt{i}} \tag{3.27}$$

We monitor the statistic M_i given by:

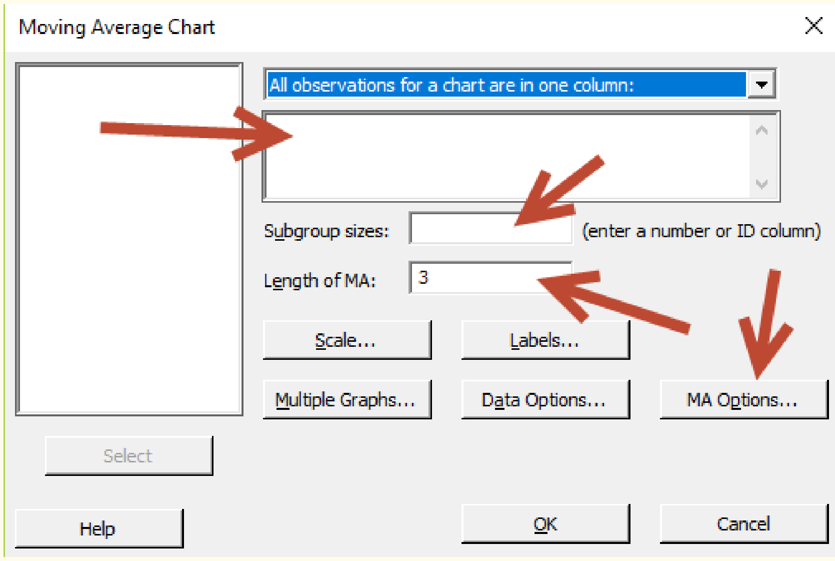
$$M_i = \frac{x_i + x_{i-1} + \cdots + x_{i-w+1}}{w} \tag{3.28}$$

The wider the span w, the more sensitive to small shifts the process is [44].

How-To 3.7 (MA charts in Minitab 18)

1. Click on Moving Average as shown in How-To 3.1
2. *Input your data > select the subgroup size > change the length of MA as necessary > click on MA options*. See the snapshot in Figure 3.18.

Figure 3.18: The MA chart options in Minitab 18, the main screen



Moving Average Chart

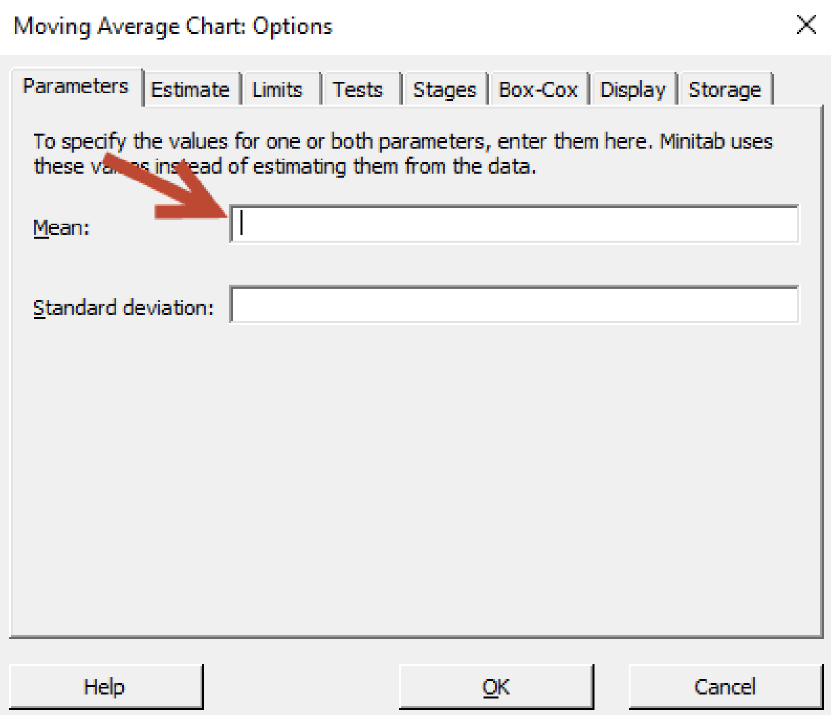
All observations for a chart are in one column:

Subgroup sizes: (enter a number or ID column)

Length of MA: 3

3. After clicking on the MA options, access the Parameters tab, and input your mean target. See the snapshot in Figure 3.19

Figure 3.19: The MA chart options in Minitab 18, Parameters tab



Moving Average Chart: Options

To specify the values for one or both parameters, enter them here. Minitab uses these values instead of estimating them from the data.

Mean:

Standard deviation:

How-To 3.8 (Python 3.6)

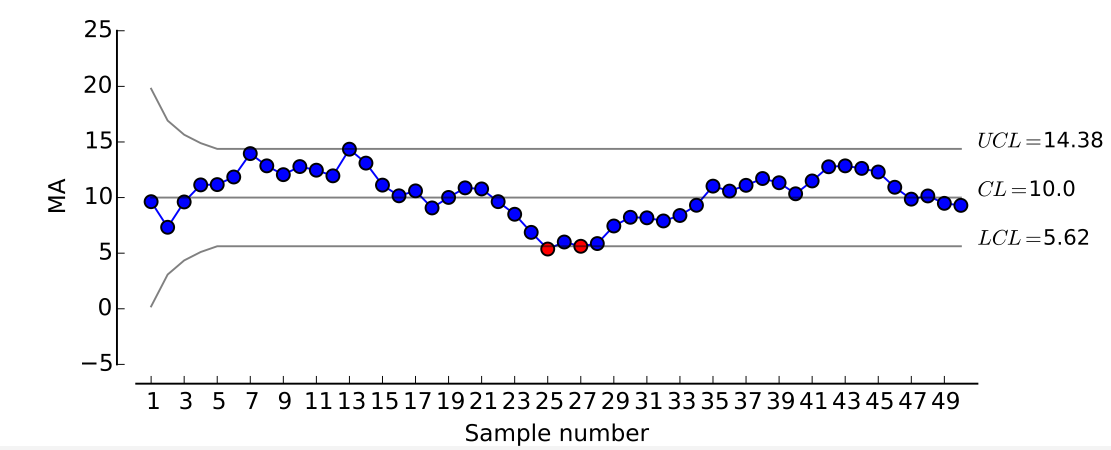
Script 3.3: A script for creating an MA chart using Python 3.6

```
##TIME WEIGHTED CHARTS MA chart
#Import modules
from pandas import*
from pylab import*
from numpy import*
import seaborn as sns
#import data from an excel spreadsheet
data = read_excel('your directory')
#parameters
xr = []
w0 = 5#morving range
L = 3.
d2 = 1.128
w = [min(i, w0) for i in range (1,len(data) +1)]
mrbar = mean([abs(data.xt[i] -data.xt[i-1]) for i in range(1,len(data
    .xt))])
sigma = mrbar/d2
data = list(data.xt)
target = 10.
xbar = target
for i in range (len(w)):
    #j is the beginning and w[i] is the end of the sliding scale
    j = 0
    if i < w[i]:#beginning of the moving average
        j = 0
    else:
        j = i-w[i]+1
    m =data[j:j+w[i]]
    n = len(m)
    Mi =  sum(m)/n
    xr.append(Mi)

UCLa = [xbar + L*sigma/sqrt(w[i]) for i in range(len(w))]
LCLa = [xbar - L*sigma/sqrt(w[i]) for i in range(len(w))]
CLa = [xbar]*len(xr0)

markers = []
colors = []

for i in range (len(xr)):
    x1 = xr[i]
    x2 = UCLa[i]
    x3 = LCLa[i]
    if (x1 > x2 or  x1<x3) :
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
```

```
#Plotting
fig=figure()
ax1 =  fig.add_subplot(111)
t = arange(len(data))
ax1.plot(UCLa, 'k-', alpha = 0.5)
ax1.plot(LCLa, 'k-',alpha = 0.5)
ax1.plot(CLa, 'k-',alpha = 0.5)
ax1.plot(xr,'b-',zorder=1)
for x,y,c,m in zip(t, xr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
#ylim(0, 20)
xlim(-0.9, t[-1]+1)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
ax1.set_xlabel('Sample number')
ax1.set_ylabel('MA')

ax1.annotate ('$UCL=$'+str(round(UCLa[-1],2)), xy = (xlim()[1], list(
    UCLa)[-1]), xytext = (xlim()[1],list(UCLa)[-1]),fontsize = 11)
ax1.annotate ('$CL=$'+str(round(CLa[0],2)), xy = (xlim()[1], list(CLa
    )[-1]), xytext = (xlim()[1],list(CLa)[-1]),fontsize = 11)
ax1.annotate ('$LCL=$'+str(round(LCLa[-1],2)), xy = (xlim()[1], list(
    LCLa)[-1]), xytext = (xlim()[1],list(LCLa)[-1]),fontsize = 11)

xticks(arange(len(xr), step = 2), arange(1, len(xr)+1, step = 2))
#ax1.legend(fancybox=True,fontsize='medium',markerscale=0.8,
    labelspacing=0.1).draggable()

ax1.yaxis.set_ticks_position('left') #remove yticks from right up
ax1.xaxis.set_ticks_position('bottom') #remove yticks from right up
show()
#print LCLa
```

Example 3.4 (An EWMA chart for the process in scenario 1)
In this last example, we once again revisit scenario 1 about General Hospital and the CPOE process and attempt to create a corresponding MA chart. We applied the Python script in How-To 3.8 to generate the MA chart in Figure 3.20.

Figure 3.20: MA chart produced in Python based on the data in Table 3.1



To reproduce the same chart in Excel, we first set up our spreadsheet, as shown in Figure 3.21. In our setup, we left rows 2 to 5 blank to initialize the span of the moving averages. Then, we proceed as follows:

1. We set $w = 5$ in cell $I8$, the $target = 10$ in cell $I9$, and $L = 3$ in cell $I10$.

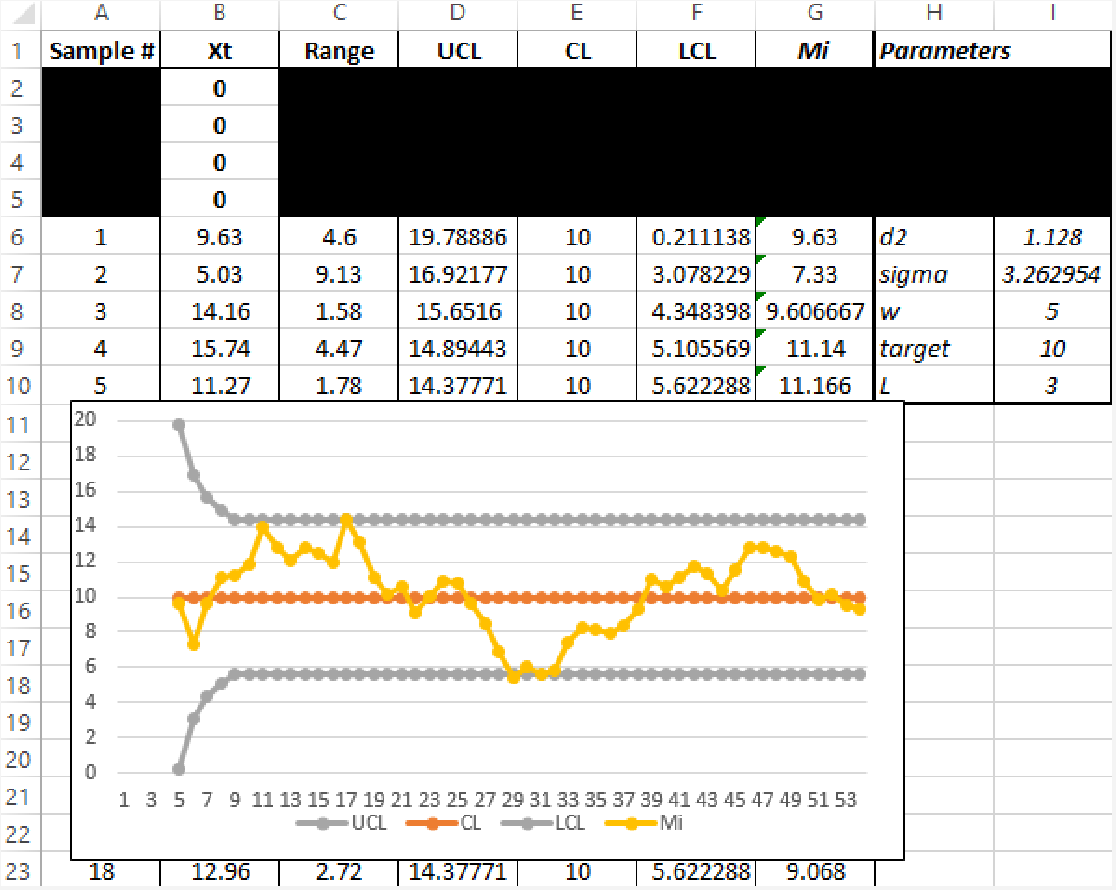
2. We estimated σ as in Example 3.1.

3. We set up the formula for the moving average statistic M_i in cell $G6$, as follows: $= SUM(B2 : B6)/MIN(A6, \$I\$8)$. We obtained the rest of the values by dragging down this formula.

4. Next, we programmed the formulas for the control limits. For example, the values of UCL and LCL in cells D6 and F6 were respectively computed as $= \$I\$9 + \$I\$10 * \$I\$7/SQRT(MIN(A6, \$I\$8))$ and $= \$I\$9 - \$I\$10 * \$I\$7/SQRT(MIN(A6, \$I\$8))$. We set CL to equal the target. We dragged down all these formulas to obtain the rest of the values.

5. We created the MA chart by inserting the lines charts of columns C-G.

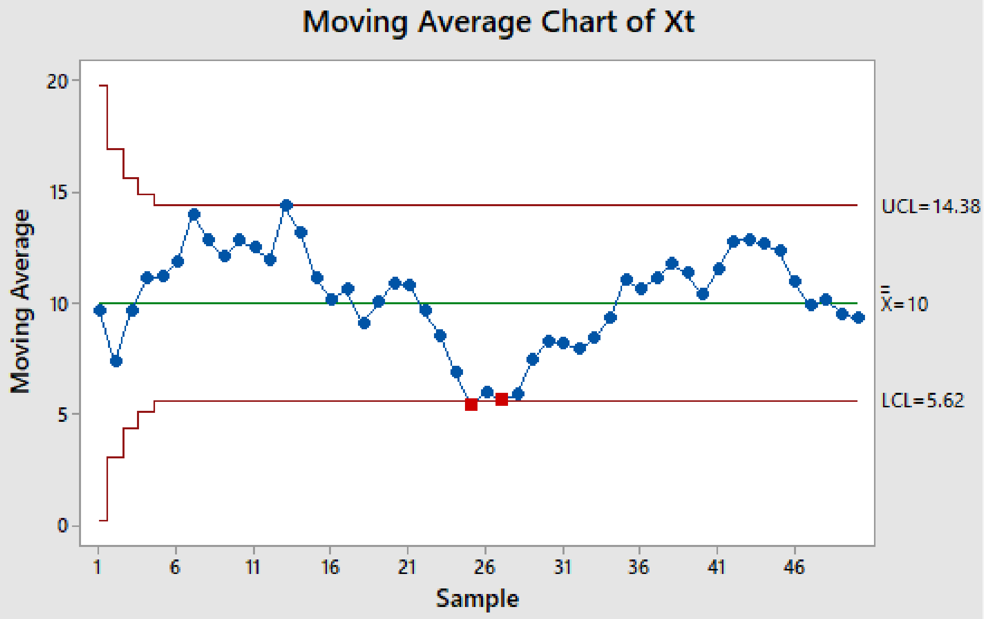
Figure 3.21: A setup of Excel to create an MA chart based on the data in Table 3.1



	A	B	C	D	E	F	G	H	I
1	**Sample #**	**Xt**	**Range**	**UCL**	**CL**	**LCL**	***Mi***	***Parameters***	
2		**0**							
3		**0**							
4		**0**							
5		**0**							
6	1	9.63	4.6	19.78886	10	0.211138	9.63	*d2*	*1.128*
7	2	5.03	9.13	16.92177	10	3.078229	7.33	*sigma*	*3.262954*
8	3	14.16	1.58	15.6516	10	4.348398	9.606667	*w*	*5*
9	4	15.74	4.47	14.89443	10	5.105569	11.14	*target*	*10*
10	5	11.27	1.78	14.37771	10	5.622288	11.166	*L*	*3*
11									
12									
13									
14									
15									
16									
17									
18									
19									
20									
21									
22									
23	18	12.96	2.72	14.37771	10	5.622288	9.068		

We reproduced a similar MA chart in Minitab 18 by following instructions in How-To 3.7. The resulting chart is portrayed in Figure 3.22.

Figure 3.22: MA chart in Minitab 18 based on the data in Table 3.1



The MA control chart shows that the process is out-of-control at samples 25 and 27. These results are slightly different than the behaviors that we observed in the CUSUM and EWMA charts. In the EWMA chart, we found that samples 13 and 25 were out-of-control. In the CUSUM charts, we observed that samples 13 and 32 were out-of-control. Still, a closer look at the MA chart reveals that sample 13 is also almost out-of-control. We also see that sample 22 coincides with the previously observed start of the LCL drift in both the CUSUM and EWMA charts.

Remark: *In general, there is no major difference in the performance of the time-weighted control charts that we discussed in this chapter. But, in practice, the CUSUM and EWMA charts are more widely used [44]*

3.5 EXERCISES

1. Follow Examples 3.1, 3.2, and 3.4 and reproduce the given CUSUM, EWMA, and MA control charts. Program the formulas in Python or Excel and verify your results in Minitab.

2. The chief of orthopedic surgery at Metropolis Hospital monitors the proper femorotibial alignment for patients who are post-total knee arthroplasty (TKA). Studies have shown that bone malalignments post-TKA can lead to long-term poor functional outcomes, including increased wear and early implant failures. The proper alignment is assessed by measuring the medial angle between the axes of the femur and the tibia [14, 65]. The data in Table 3.3 shows angle measurements of 60 post-TKA patients at this facility. The target is 180%.

   (a) Using Excel, Python, and Minitab, create a CUSUM chart for individual angle measurements in Table 3.3

   (b) Use the same dataset to create ImR charts.

   (c) Compare and contrast your results in (a) and (b). What actions, if any, should be taken to stabilize the process?

3. Table 3.4 presents data about the *time to administration of pain medication to ED patients with broken bones* at Metropolis Hospital. Tracking and reporting this information is required by CMS as one of its mandated quality measures about timely and effective care [17].

   (a) Using Excel, Python, and Minitab, create EWMA and MA control charts related to individual angle measurements in Table 3.4

   (b) Use the same dataset to create ImR charts.

   (c) Compare and contrast your results. What can you conclude?

4. Use the data in Table 3.5 to set up an integral controller for the y_t nursing process. Assume $T = 100$, $\lambda = 0.2$, and $g = 2$.

   (a) Create ImR charts of the uncontrolled process

   (b) Create ImR charts of the controlled process.

   (c) Compare and contrast your results. What can you conclude?

   (d) How would you set up an adjustment chart to allow the nurse to control the process manually?

Table 3.3: Femorotibial angle measurements of post-TKA patients at Metropolis Hospital

Patient#	Angle	Patient#	Angle
1	193.5	*31*	169.9
2	170.5	*32*	183.3
3	176	*33*	182.3
4	171.4	*34*	197.8
5	172.5	*35*	198.8
6	198.3	*36*	171
7	184.3	*37*	175.7
8	174	*38*	179.1
9	170	*39*	177.5
10	160.4	*40*	179.9
11	165	*41*	169.4
12	160.7	*42*	161.4
13	171.4	*43*	179.2
14	168.1	*44*	197.8
15	170.5	*45*	174.6
16	188	*46*	184.7
17	163.4	*47*	167.9
18	180.4	*48*	177.4
19	179.4	*49*	171.6
20	190.8	*50*	196.1
21	199.5	*51*	175.3
22	198.9	*52*	200.9
23	200.4	*53*	169.9
24	189.3	*54*	189.5
25	181.2	*55*	176
26	176.3	*56*	191.5
27	195.8	*57*	170.1
28	198.2	*58*	189.6
29	175.2	*59*	182
30	175.5	*60*	200.4

Table 3.4: Time, in minutes, to the administration of pain medication to ED patients with broken bones at Metropolis Hospital

Patient#	**Time**	***Patient#***	**Time**
1	42	*21*	40
2	46	*22*	47
3	17	*23*	47
4	25	*24*	9
5	41	*25*	44
6	38	*26*	24
7	23	*27*	8
8	55	*28*	23
9	28	*29*	36
10	12	*30*	55
11	14	*31*	63
12	26	*32*	60
13	28	*33*	53
14	32	*34*	26
15	11	*35*	26
16	53	*36*	13
17	20	*37*	11
18	5	*38*	8
19	8	*39*	29
20	15	*40*	75

Table 3.5: The dynamics of the y_t nursing process

t	y_t	t	y_t	t	y_t	t	y_t
1	95	14	101	27	95	40	111
2	110	15	99	28	97	41	145
3	97	16	109	29	130	42	126
4	99	17	96	30	97	43	10
5	111	18	110	31	97	44	119
6	109	19	120	32	127	45	139
7	101	20	124	33	120	46	121
8	103	21	114	34	146	47	137
9	114	22	110	35	114	48	134
10	100	23	97	36	121	49	136
11	200	24	97	37	145	50	111
12	91	25	90	38	115		
13	118	26	113	39	133		

CHAPTER 4

Adjusted control charts

Summary

In this chapter, we study how to adjust control charts for risk and autocorrelation. We also learn how to control multivariate processes. As in previous chapters, we emphasize practical applications with examples and How-To clauses to demonstrate how to encode relevant formulas using Excel and Python software. We also show how to apply the Minitab statistical package.

Key concepts and tools

Control charts; Risk-adjusted; P chart; CUSUM[1] *chart; Hotelling T^2; Autocorrelation; Multivariate normal; Covariance matrix; Time series; Residuals; Lag; ACF*[2]*; ARIMA*[3]

Major objectives

After studying this chapter, you will be able to:

1. *Define key concepts and tools for adjusted control charts*
2. *Learn how to recognize risk in health care processes*
3. *Explain the concept of autocorrelation*
4. *Recognize the need for multivariate control adjustment*
5. *Design and construct risk-adjusted control charts*
6. *Design and construct control charts for autocorrelated data*
7. *Design and construct multivariate control charts*
8. *Implement adjusted control charts using Excel, Python, and Minitab*
9. *Appraise the stability of the process in adjusted control charts*
10. *Survey implementation strategies for adjusted control charts*

[1] **CUSUM**: Cumulative sum

[2] **ACF**: Auto-correlation function

[3] **ARIMA**: Autoregressive integrated moving average

4.1 Introduction

In this chapter, we learn how to adjust control charts when some of our assumptions in the previous chapters no longer hold. For example, the charts that we have discussed so far relate to uncorrelated single variables. Additionally, we have been implicitly assuming the uniformity of inputs, and thus eliminated the need for conditioning quality measures on anything else but the process. These assumptions are occasionally violated in some of the processes that we encounter in health care, an incidence that requires us to make adjustments to the traditional control charts. The first type of adjustment that we consider relates to **risk**. We create **risk-adjusted charts** to alleviate bias in quality measures due to the patient's health risk factors. The second type of adjustment we discuss concerns **autocorrelation**. When our samples are correlated over time, deploying traditional control charts may lead to erroneous conclusions. We will demonstrate how to apply **time series** techniques to allow us to monitor the process using the model **residuals**. The last type of adjustment that we make to traditional control charts pertains to processes where variables of interest are jointly distributed. To monitor such processes, we employ **multivariate control charts**. One kind of these charts that we review here is the Hotelling T^2 control chart.

4.2 Risk-adjusted control charts

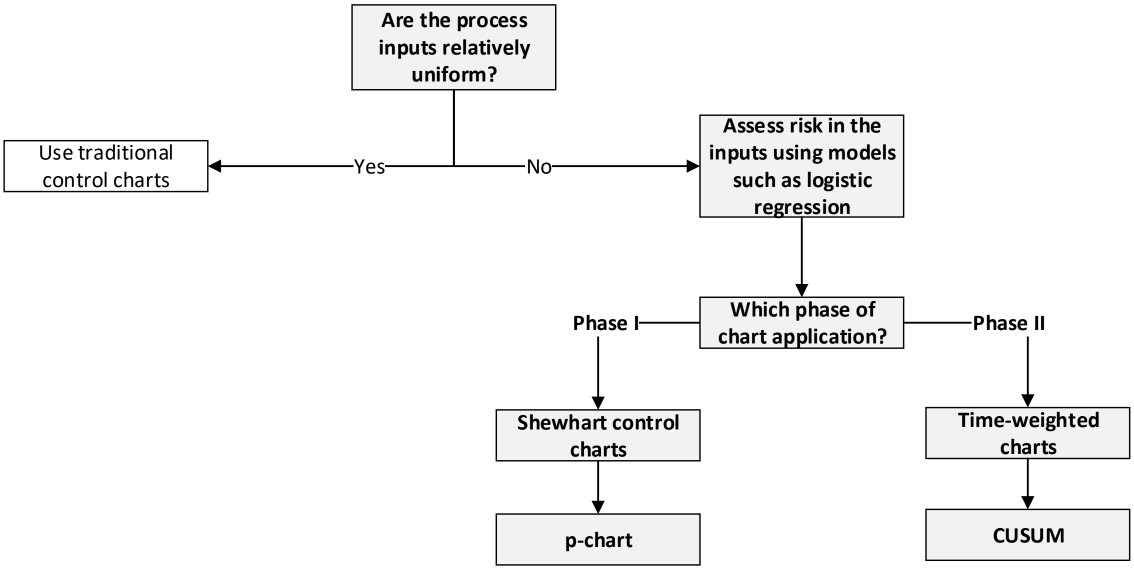
Risk-adjusted control charts may be applied to any process where the quality of outputs is affected by uneven inputs. In the manufacturing sector, risk-adjustment may not be necessary since raw materials tend to be approximately uniform. But, when monitoring processes related to patient care, there is likely to be a need to adjust traditional control charts to account for the **risk** inherent in the patient's **case-mix**[4] or **risk factors** [3, 65]. It is imperative to have an accurate measure of this risk for the adjusted control charts to be of any value. Statistical techniques such as logistic regression are often utilized for this purpose [65]. To review how to fit a logistic regression model to data, see Sharma (1995) [56].

As depicted in Figure 4.1, in this section, we only discuss the most common risk-adjusted control charts, namely p and $CUSUM$ charts. Possible scenarios where these charts may be applied are discussed next.

Scenario 1: A quality manager at Metropolis Hospital is interested in monitoring mortality rates in an intensive care unit (ICU). The manager understands that the patients' risk factors tend to bias mortality rates. For example, patients with acute renal failure often have a higher risk of mortality than patients without this condition. Also, the risk of dying tends to increase with age [21]. Accordingly, the manager has decided to apply the risk-adjusted charts to monitor the process. See Cook et al. (2003) [19] for an actual case study where risk-adjusted control charts were employed to monitor ICU mortality rates.

[4] **Case-mix**: a term broadly used to describe the variation in the patient's quality outcomes due to intrinsic factors such as comorbidity, sex, age, and health status [19, 31]

Figure 4.1: A basic decision tree for risk-adjusted charts


Are the process inputs relatively uniform?
Yes
No
Use traditional control charts
Assess risk in the inputs using models such as logistic regression
Which phase of chart application?
Phase I
Phase II
Shewhart control charts
Time-weighted charts
p-chart
CUSUM

Scenario 2: The administrator of a skilled nursing facility decided to apply risk-adjusted control charts to monitor patient fall rates. The manager is aware that the risk of falling tends to increase with age [54]. The manager also had noticed that dementia patients fell more often as compared to the residents who do not suffer from this condition [62]. For further discussion about the application of risk-adjusted charts to monitor patient falls, see Alemi et al. (2001) [3].

Scenario 3: The chief medical officer at Children's Hospital wants to use risk-adjusted CUSUM control charts to monitor outcomes of pediatric cardiac surgeries. To measure risk, the manager decided to apply the standard Parsonnet score, where a high score indicates an increased risk of dying. See Steiner et al. (2000)[59] for more discussion about the use of risk-adjusted CUSUM charts to monitor outcomes of cardiac surgeries.

4.2.1 Risk-adjusted p-charts

Similar to traditional p-charts, risk-adjusted p-charts are generally deployed in *phase I of chart application* to monitor processes that generate binomial data. We recall that p-charts obey the Bernoulli distribution as follows:

$$p(x) = \begin{cases} p & x = 1 \\ 1 - p & x = 0 \end{cases} \tag{4.1}$$

As before, p signifies the probability of an event x. When $x = 1$, an event occurred. Otherwise, an event did not occur. In both the traditional and risk-adjusted p-charts, we monitor p_i, the probability of an event in sample i, that we obtain as follows:

$$p_i = \frac{1}{n_i} \sum_{j=1}^{n_i} x_{ij} \tag{4.2}$$

Here, n_i is the size of sample i. We can think of x_{ij} as the outcome of patient j in sample i. The only difference between the traditional and risk-adjusted p-charts involves control limits. In traditional p-charts, we construct the control limits using parameter $\bar{p}$ that we obtain like this:

$$\bar{p} = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n_i} x_{ij}}{\sum_{i=1}^{m} n_i} \tag{4.3}$$

As before, m is the total number of samples. In risk-adjusted p-charts, we construct the control limits using the parameter $\hat{p}_{ij}$, the probabilistic risk of a particular outcome for patient j in sample i. We obtain the expected risk in sample i as follows:

$$\frac{1}{n_i} \sum_{j=1}^{n_i} \hat{p}_{ij} \tag{4.4}$$

To estimate $\hat{p}_{ij}$, we typically apply logistic regression on outcomes of care given the patient's risk factors [65]. In some groupers of medical codes such as APR-DRG[5] in 3M [5], the risk of mortality is automatically generated based on the severity of illness reflected from the coded medical diagnoses and other medical information. Box 4.1 summarizes the typical formulation of risk-adjusted p-charts.

Box 4.1 Risk-adjusted p-charts

For each sample i, the upper control limit (UCL_i), the lower control limit (LCL_i), and the centerline (CL_i) are constructed as follows [19, 65]:

$$UCL_i = \frac{1}{n_i} \left[\sum_{j=1}^{n_i} \hat{p}_{ij} + L \sqrt{\sum_{j=1}^{n_i} \hat{p}_{ij}(1-\hat{p}_{ij})} \right] \tag{4.5}$$

$$CL_i = \frac{1}{n_i} \sum_{j=1}^{n_i} \hat{p}_{ij} \tag{4.6}$$

$$LCL_i = \max\left(0, \frac{1}{n_i} \left[\sum_{j=1}^{n_i} \hat{p}_{ij} - L \sqrt{\sum_{j=1}^{n_i} \hat{p}_{ij}(1-\hat{p}_{ij})} \right]\right) \tag{4.7}$$

where $\hat{p}_{ij}$ is the predicted probabilistic risk of patient j in sample i. As in traditional p-charts, L is the distance from the control limit to the centerline. For mortality risk-adjustment, it is common to set $L = 2$ [65].

[5] **APR-DRG:** All Patient Refined Diagnosis Related Groups

How-To 4.1 (Risk-adjusted p-charts in Minitab 18)
Minitab 18 does not have an option for creating risk-adjusted p-charts

How-To 4.2 (Risk-adjusted p-charts in Excel 2013)
Excel 2013 does not have a built-in option for creating risk-adjusted p-charts, but it is possible to manually program the formulas in Box 4.1 as we demonstrate in Example 4.1.

How-To 4.3 (Python 3.6)

Script 4.1: A script for creating a risk-adjusted p-chart using Python 3.6. language

```
#RISK-ADJUSTED P-CHART
#import modules
from pandas import*
from pylab import*
from numpy import*
import seaborn as sns
data = read_excel('your directory')
#initialize parametes
L = 2.
#find unique days
days = data.Day.unique()
#determine control limits, centerline, and the statistic to be
   monitored.
UCL = []
CL = []
LCL = []
Day = []
P = []
for i in days:
    dayi = data[data.Day==i]
    ni = len(dayi)
    sumphat1 = dayi.Probability.sum()/ni
    sumphat2 = sqrt(sum(dayi.Probability*(1.-dayi.Probability)))/ni
    UCLi = sumphat1 + L*sumphat2
    CLi =  sumphat1
    LCLi = max(0,sumphat1 - L*sqrt(sumphat2))
    Pi = float(dayi.Died.sum())/ni
    #append results
    UCL.append(UCLi )
    CL.append(CLi)
    LCL.append(LCLi)
    P.append(Pi)
    Day.append(i)
xr = P
#mark red the point that falls outside of the control limits.
   Otherwise, mark the point blue.
markers = []
colors = []
for i in range (len(xr)):
    x1 = xr[i]
    if (x1 > UCL[i] or x1 < LCL[i]):
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
#plot the adjusted p-chart using the step function for limits
fig=figure()
ax1 =  fig.add_subplot(111)
t = arange(len(P))
ax1.step(t, UCL, 'k-', alpha = 0.5, where = 'mid')
ax1.step(t, LCL, 'k-',alpha = 0.5,  where = 'mid')
ax1.step(t, CL, 'k-',alpha = 0.5, where = 'mid')
ax1.plot(xr,'b-',zorder=1)
for x,y,c,m in zip(t, xr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel('Sample number')
ax1.set_ylabel('Fraction nonconforming')
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL=$'+str(round(UCL[-1],2)), xy = (xlim()[1], list(
   UCL)[-1]), xytext = (xlim()[1],list(UCL)[-1]),fontsize = 11)
ax1.annotate ('$\overline{P}=$'+str(round(CL[-1],2)), xy = (xlim()
   [1], list(CL)[-1]), xytext = (xlim()[1],list(CL)[-1]),fontsize =
   11)
ax1.annotate ('$LCL=$'+str(round(LCL[-1],2)), xy = (xlim()[1], list(
   LCL)[-1]), xytext = (xlim()[1],list(LCL)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(UCL), step = 2), arange(1, len(UCL)+1, step = 2))
show()
```

Example 4.1 (ICU mortality rates)

Let us revisit scenario 1, where the quality manager at Metropolis Hospital wants to use risk-adjusted charts to monitor mortality rates in ICU. On each day i, the manager records the outcome of the discharged patient j as $x_{ij} = 1$, if the patient died, and $x_{ij} = 0$, if the patient survived. The risk of mortality of each patient $\hat{p}_{ij}$ was estimated using logistic regression. So far, the manager has collected data for 5 days, as displayed in Table 4.1. Using the data from Table 4.1, we determine the daily statistic p_i per Equation 4.2. Our results are as follows:

$$p_1 = 1/5;\ p_2 = 0/4;\ p_3 = 1/6;\ p_4 = 1/3;\ p_5 = 2/9$$

We use Equations 4.5 - 4.7 to calculate UCL_i, C_i, and LCL_i for each day i. For example, on day 1 with $L = 2$, we obtained the following results:

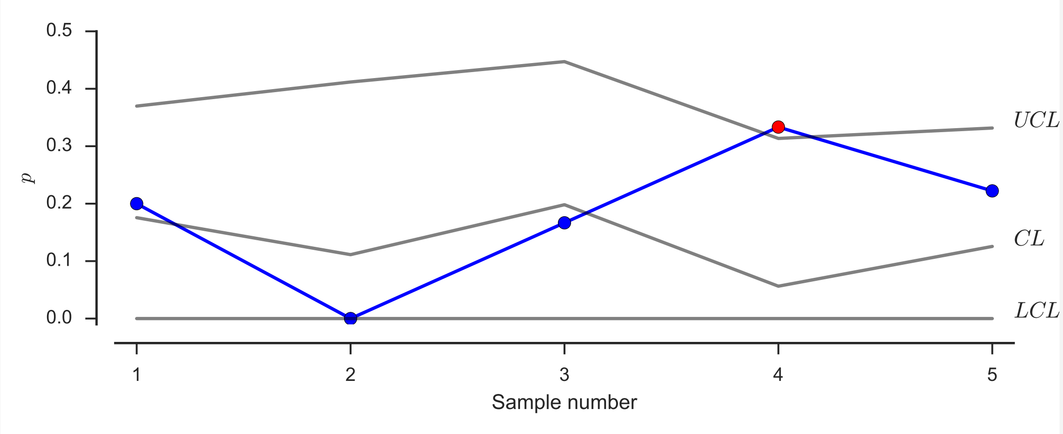
$$\begin{aligned} UCL_1 &= \frac{1}{5}\left[0.878 + 2\sqrt{0.236}\right] = 0.370 \\ CL_1 &= \frac{1}{5}(0.020 + 0.008 + 0.040 + 0.010 + 0.800) = \frac{0.878}{5} = 0.176 \\ LCL_1 &= \max\left(0, \frac{1}{5}\left[0.878 - 2\sqrt{0.236}\right]\right) = \max(0, -0.019) = 0 \end{aligned}$$

Table 4.1: Daily tracking of ICU deaths, Metropolis Hospital, 2018

Day_i	$Patient_j$	$\hat{p}_{ij}$	x_{ij}
1	*1*	0.020	0
1	*2*	0.008	0
1	*3*	0.040	0
1	*4*	0.010	0
1	*5*	0.800	1
2	*1*	0.062	0
2	*2*	0.260	0
2	*3*	0.110	0
2	*4*	0.013	0
3	*1*	0.040	0
3	*2*	0.280	0
3	*3*	0.050	0
3	*4*	0.738	1
3	*5*	0.030	0
3	*6*	0.051	0
4	*1*	0.001	0
4	*2*	0.029	0
4	*3*	0.139	1
5	*1*	0.060	0
5	*2*	0.140	0
5	*3*	0.010	0
5	*4*	0.150	1
5	*5*	0.040	0
5	*6*	0.100	0
5	*7*	0.010	0
5	*8*	0.210	0
5	*9*	0.410	1

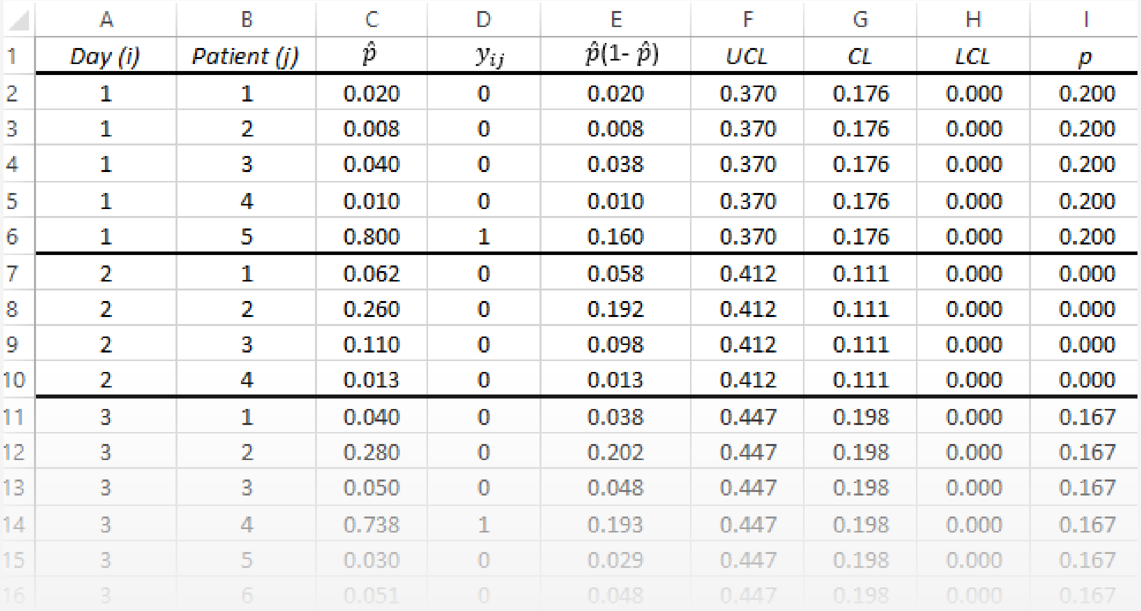
After running the Python script in How-To 4.3, we produced the risk-adjusted p-chart illustrated in Figure 4.2.

Figure 4.2: A risk-adjusted p-chart based on the data in Table 4.1



To reproduce a similar chart in Excel, we begin by setting up our spreadsheet, as illustrated in Figure 4.3.

Figure 4.3: A setup of an Excel spreadsheet to construct a risk-adjusted p-chart based on the data in Table 4.1



	A	B	C	D	E	F	G	H	I
1	*Day (i)*	*Patient (j)*	$\hat{p}$	y_{ij}	$\hat{p}$(1- $\hat{p}$)	*UCL*	*CL*	*LCL*	*p*
2	1	1	0.020	0	0.020	0.370	0.176	0.000	0.200
3	1	2	0.008	0	0.008	0.370	0.176	0.000	0.200
4	1	3	0.040	0	0.038	0.370	0.176	0.000	0.200
5	1	4	0.010	0	0.010	0.370	0.176	0.000	0.200
6	1	5	0.800	1	0.160	0.370	0.176	0.000	0.200
7	2	1	0.062	0	0.058	0.412	0.111	0.000	0.000
8	2	2	0.260	0	0.192	0.412	0.111	0.000	0.000
9	2	3	0.110	0	0.098	0.412	0.111	0.000	0.000
10	2	4	0.013	0	0.013	0.412	0.111	0.000	0.000
11	3	1	0.040	0	0.038	0.447	0.198	0.000	0.167
12	3	2	0.280	0	0.202	0.447	0.198	0.000	0.167
13	3	3	0.050	0	0.048	0.447	0.198	0.000	0.167
14	3	4	0.738	1	0.193	0.447	0.198	0.000	0.167
15	3	5	0.030	0	0.029	0.447	0.198	0.000	0.167
16	3	6	0.051	0	0.048	0.447	0.198	0.000	0.167

In our spreadsheet setup, we computed the value in cell $F2$ this way:

$$= (1/COUNT(A2:A6))*(SUM(C2:C6)+2*(SQRT(SUM(E2:E6)))) \quad (4.8)$$

Here, we used the $COUNT()$ function to determine the number of discharged patients on each day. We used the $SUM()$ function to aggregate the number of deaths. We applied the $SQRT()$ function to calculate the square root of a given quantity. We present the summary of statistics for constructing a risk-adjusted p-chart in Table 4.2.

Table 4.2: Statistics for a risk-adjusted p-chart based on the data in Table 4.1

Day	UCL	CL	LCL	**p**
1	0.37	0.18	0	0.20
2	0.41	0.11	0	0.00
3	0.45	0.20	0	0.17
4	0.31	0.06	0	0.33
5	0.33	0.13	0	0.22

Subsequently, we can insert line charts of columns *UCL, CL, LCL,* and p in Table 4.2 to generate a control chart similar to the one shown in Figure 4.2.

From the chart in Figure 4.2, we notice that the process was out-of-control on day 4 since a point fell outside of UCL_4. One possible interpretation of this result is that the mortality rate on day 4 was statistically high given the expected risk on that day. The manager should go back on this day and investigate what happened and correct any assignable causes.

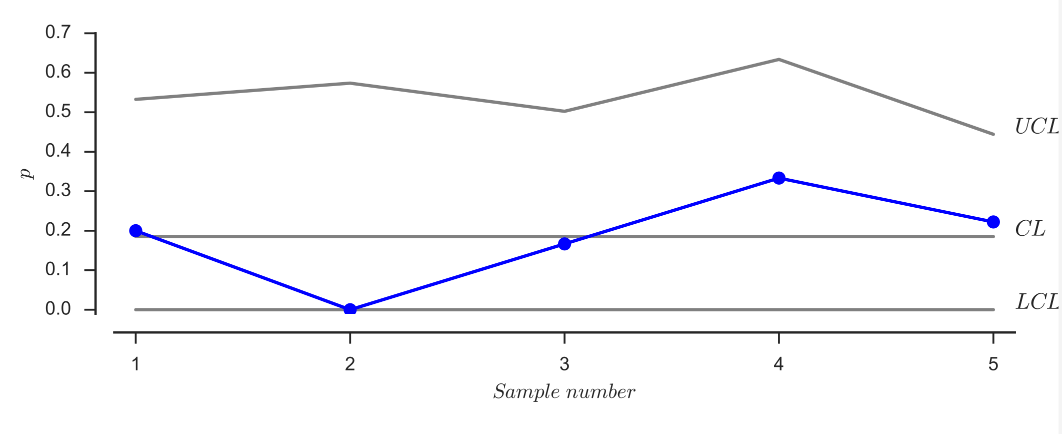
For comparison purposes, let us create a traditional p-chart using the same dataset in Table 4.1. We start by summarizing the statistics to be plotted in Table 4.3.

Table 4.3: Statistics for the traditional p-chart based on the data in Table 4.1

Day	**UCL**	**CL**	**LCL**	**p**
1	0.53	0.19	0	0.20
2	0.57	0.19	0	0.00
3	0.50	0.19	0	0.17
4	0.63	0.19	0	0.33
5	0.44	0.19	0	0.22

We calculated the quantities in Table 4.3 by following the formulas in Box 2.4 with $L = 2$. We produced the p-chart presented in Figure 4.4.

Figure 4.4: A traditional p-chart based on the data in Table 4.3



From the chart in Figure 4.4, we notice no out-of-control behaviors. In other words, the traditional p-chart failed to capture the low risk and high mortality rate on day 4. We should note that the number of samples is too small here to make any conclusive inferences. In general, the sample number should be at least 25 before inferring about the stability of the process, unless the standard p was given.

4.2.2 Risk-adjusted CUSUM charts

We use risk-adjusted CUSUM charts to detect small shifts in the processes that have been adjusted for risk. Given an indicator process that obeys a Bernoulli distribution, the outcome of patient j can be classified as $x_j = 1$ if an event occurred, and $x_j = 0$, if no event happened. In our discussion, we confine ourselves to the event of morality and assume individual observations where the sample size $n = 1$. To create a risk-adjusted CUSUM chart for mortality, we must be able to calculate the weight W_j as follows:

$$W_j = x_j \cdot \log(OR) - log(1 - \hat{p}_{j0} + OR \cdot \hat{p}_{j0}) \tag{4.9}$$

where OR symbolizes the odds ratio of mortality given by:

$$OR = \frac{\hat{p}_{j1}(1 - \hat{p}_{j0})}{\hat{p}_{j0}(1 - \hat{p}_{j1})} \tag{4.10}$$

Here, $\hat{p}_{j0}$ is the probability that patient j dies under the null hypothesis $H_0 : OR = 1$, whereas $\hat{p}_{j1}$ is the probability that patient j dies under the alternative hypothesis $H_1 : OR \neq 1$ [65]. In general, we use logistic regression to approximate $\hat{p}_{j0}$, and we postulate the value of OR. For example, if we assume that $OR = 2$ and a point falls outside of the upper control limit, we can say that the odds of mortality have doubled. Likewise, if we assume $OR = 0.5$ and the CUSUM statistic falls outside of the lower control limit, we can conclude that the odds of mortality have been halved [19].

Box 4.2 summarizes the statistics that we monitor in risk-adjusted CUSUM charts. We obtain control limits like in traditional CUSUM charts by setting the limits to $\pm H$, where $H = h\sigma$ and h ranges between 4 or 5. If not given, we estimate σ from the process (review Box 3.1)

Box 4.2 Risk-adjusted CUSUM charts

For each patient j, we compute and monitor the following statistics:

$$C_j^+ = \max(0, C_{j-1} + W_j), \text{ if OR} \geq 1 \tag{4.11}$$

$$C_j^- = \min(0, C_{j-1} - W_j), \text{ if OR} < 1 \tag{4.12}$$

We determine W_j per Equation 4.9. C_j^- and C_j^+ are cumulative sums of W_j and $C_0^+ = C_0^- = 0$. We apply C_j^- to monitor the odds of the decrease in mortality, whereas C_j^+ helps us monitor the odds of the increase in mortality [19]. After the CUSUM chart signals out-of-control behaviors, we address any assignable causes and reset the chart to $C_0^+ = C_0^- = 0$ [19].

How-To 4.4 (Risk-adjusted CUSUM charts in Minitab 18)
Minitab 18 does not have an option for creating risk-adjusted CUSUM charts

How-To 4.5 (Risk-adjusted CUSUM charts in Excel 2013)
Excel 2013 does not have a built-in option for creating risk-adjusted CUSUM charts, but it is possible to manually program the formulas in Box 4.2 as we demonstrate in Example 4.2.

How-To 4.6 (Python 3.6)

Script 4.2: A script for creating a risk-adjusted CUSUM chart using Python 3.6

```
#RISK-ADJUSTED CUSUM
#import modules
from pandas import*
from pylab import*
from numpy import*
import seaborn as sns
#import data from an Excel spreadsheet. The column of "Probability"
   represents the risk, and the column of "Died" represents the
   outcome.
data = read_excel('your directory')
#initialize parameters
OR1 = 2.
OR2 = 0.5
h1 = 4.5
#estimate the variance
N = len(data)+1
ssquare = sum(data.Probability*(1.-data.Probability))
sigma =sqrt(ssquare)/N
H = h1*sigma
#create control limits
UCL = [H]*N
CL = [0.]*N
LCL = [-H]*N
#compute C_^+ and C^- statistics to be monitored
Wa = [0]
Wb = [0]
for i in range(len(data)):
    xj = data.Died.iloc[i]
    pj = data.Probability.iloc[i]
    w1 = xj*log(OR1) -log(1.-pj + OR1*pj)
    w2 = xj*log(OR2) -log(1.-pj + OR2*pj)
    Wa.append(w1)
    Wb.append(-w2)
#cumsum is a cumulative function in python
cum1 = cumsum(Wa)
cum2 = cumsum(Wb)
cplus = [max(i,0) for i in cum1]
cminus =  [min(i,0) for i in cum2]
c1 = cplus
c2 = cminus
#mark red the point that falls outside of the control limits.
   Otherwise, mark the point blue.
markers = []
```

```
colors = []
markers1 = []
colors1 = []
for i in range (N):
    x1 = c1[i]
    if x1 > UCL[i]:
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
for i in range (N):
    x2 = c2[i]
    if  x2 <LCL[i]:
        markers1.append('o')
        colors1.append('r')
    else:
        markers1.append('o')
        colors1.append('g')
#plotting the adjusted CUSUM chart
fig=figure()
ax1 =  fig.add_subplot(111)
t = arange(len(CL))
ax1.plot(UCL, 'k-', alpha = 0.5)
ax1.plot(LCL, 'k-',alpha = 0.5)
ax1.plot(CL, 'k-',alpha = 0.5)
ax1.plot(c1,'b-',zorder=1)
ax1.plot(c2,'g-',zorder=1)
for x,y,c,m in zip(t, c1, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
for x,y,c,m in zip(t, c2, colors1, markers1):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel('Patient number')
ax1.set_ylabel('Cumulative risk')
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL=$'+str(round(UCL[0],2)), xy = (xlim()[1], list(
   UCL)[-1]), xytext = (xlim()[1],list(UCL)[-1]),fontsize = 11)
ax1.annotate ('$CL=$'+str(round(abs(CL[0]),2)), xy = (xlim()[1], list
   (CL)[-1]), xytext = (xlim()[1],list(CL)[-1]),fontsize = 11)
ax1.annotate ('$LCL=$'+str(round(LCL[0],2)), xy = (xlim()[1], list(
   LCL)[-1]), xytext = (xlim()[1],list(LCL)[-1]),fontsize = 11)
#set xticks to start from one since Python starts counting from zero
xticks(arange(N, step = 1), arange(1, N+1, step = 1))
show()
```

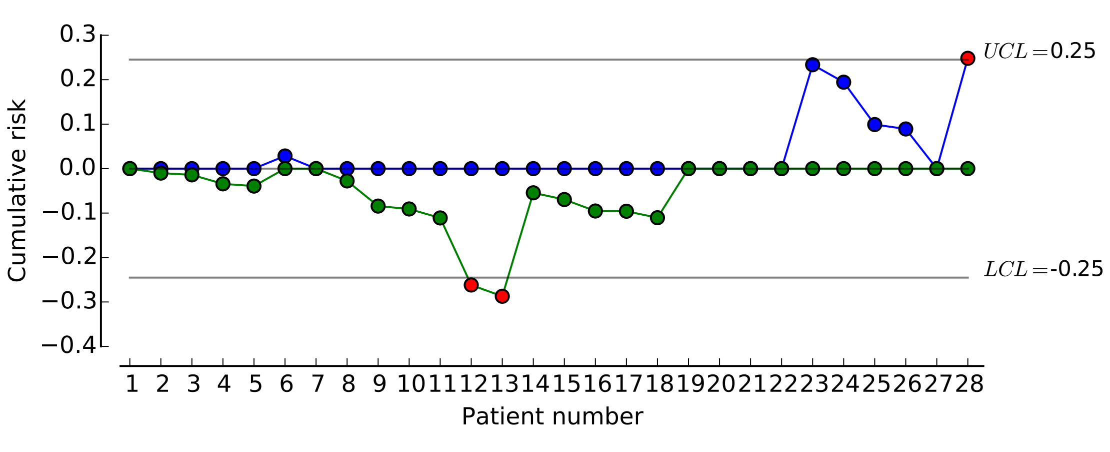
Example 4.2 (A Risk-adjusted CUSUM chart)
In this example, we use the data from Example 4.1 and attempt to create a related risk-adjusted CUSUM chart for individual patients. We set $h = 4.5$ and approximate

σ as follows:

$$\hat{\sigma} = \frac{1}{n}\sqrt{\sum_{j=1}^{n}\hat{p}_j(1-\hat{p}_j)} \tag{4.13}$$

where n is the total number of patients and $\hat{p}_j$ is the predicted mortality risk of patient j. From Table 4.1, we obtained $\hat{\sigma} = 0.054$, meaning that $H = 0.054(4.5) = \pm 0.2452$. Next, we calculate W_j for the upper CUSUM chart with $OR = 2$. For example, for patient 1, $W_1 = 0\log(2) - \log(1 - 0.020 + 2(0.020)) = -0.020$, and for patient 5, $W_5 = 1\log(2) - \log(1 - 0.8 + 2(0.8)) = 0.105$. We implemented the same procedure to determine the lower CUSUM where $OR = 0.5$. We applied the Python script in How-To 4.6 to produce the chart illustrated in Figure 4.5.

Figure 4.5: A risk-adjusted CUSUM chart for individual patients in Table 4.1.



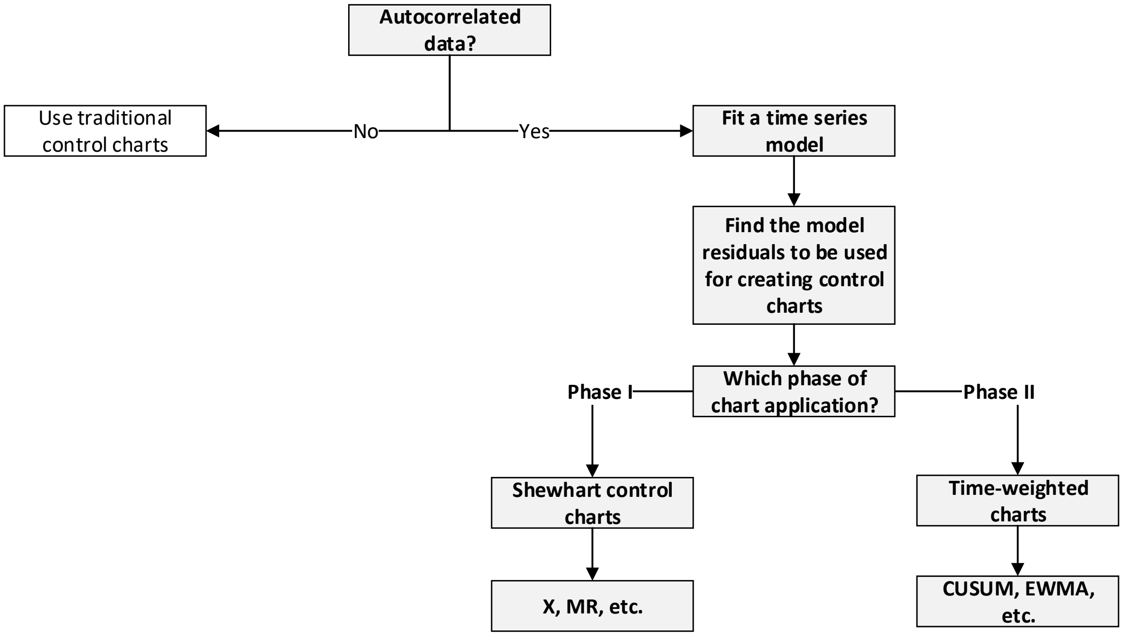
From the CUSUM chart in Figure 4.5, we can conclude that from patient number 7 up until patient number 12, the risk of mortality was halved. We also notice that from patient number 27 to patient number 28, the risk of mortality doubled. The manager should attempt to find the reasons for these out-of-control behaviors by going back to the sample number where the process started to drift away from zero. Next, the manager should reset the charts to $C_0^+ = C_0^- = 0$ before resuming the monitoring procedure.

4.3 Control charts for autocorrelated data

Both the traditional Shewhart and time-weighted charts assume independence of samples. When this assumption no longer holds, and **autocorrelation** exists, we utilize control charts for autocorrelated data to monitor the process. One approach for creating these charts involves fitting an appropriate **time series** model to the data and then applying traditional charts to monitor the process using the model **residuals** [44]. Figure 4.6

presents a basic decision tree for selecting an appropriate control chart for autocorrelated data.

Figure 4.6: A basic decision tree for control charts of autocorrelated data



Autocorrelation tends to occur in processes that generate continuous data such as in chemical plants [44]. In health care, most of the processes we deal with for quality improvement are discrete. But, autocorrelation can still manifest as we entertain in the following scenarios.

Scenario 1: A lab manager at General Hospital has observed that current measurements of creatinine assays could be predicted from previous measurements over time. Therefore, for monitoring and controlling the process, the manager has decided to adopt control charts for autocorrelated data. See Winkel et al. (2007) [65] for a discussion about a scenario of autocorrelation in lab assays.

Scenario 2: A lab manager at Metropolis Hospital has been getting several complaints from emergency department physicians about delays in the turn-around-time (TAT) of CBC [6] orders. Knowing that delayed lab results are often associated with increased patient length of stay [58], the manager took these concerns very seriously and started monitoring TAT of CBC orders using ImR charts. The initial assessment of these charts revealed a stationary mean but with obvious trends in the data. Before proceeding any further, the manager decided to test for autocorrelation in the samples. Autocorrelation was detected, and subsequently, the manager decided to monitor the process using charts for autocorrelated data.

[6] **CBC:** Complete Blood Count

4.3.1 How to measure autocorrelation?

We measure **autocorrelation** using a function of the correlation between successive samples over time. To motivate our discussion, let's assume that over time t we have collected data x_t about a given process. After plotting x_t against t, we observed trend behaviors, and thus suspected that autocorrelation existed. To validate this suspicion, we can, for instance, calculate the Pearson's correlation coefficient in the data given one-time **lag**. That is, we determine the relationship between x_t and x_{t-1}. We conclude that autocorrelation exists when the Pearson coefficient is significant. Given k lags, we write a general **autocorrelation function (ACF)**, ρ_k, as follows [45]:

$$\rho_k = \frac{Cov(x_t, x_{t-k})}{\sigma_t \sigma_{t-k}} \tag{4.14}$$

where $Cov()$ symbolizes the covariance function and σ denotes the standard deviation. We recall that a Pearson correlation coefficient obeys this inequality: $-1 < \rho_k < 1$. If the process is **stationary**, to imply that the mean and the variance are relatively constant, the denominator in Equation 4.14 will simply be the variance of x_t, given by $\sigma_t \sigma_t = \sigma_t^2$. If σ_t^2 is not known, we estimate it using s_t^2, the sample variance of x_t. Accordingly, we can approximate ACF this way:

$$\rho_k \approx \hat{\rho}_k = \frac{Cov(x_t, x_{t-k})}{s_t^2} \tag{4.15}$$

We conclude that $\hat{\rho}_k$ is statistically significant when its value exceeds two standard deviations [44]. For example, for the lag $k = 1$, autocorrelation exists when:

$$|\hat{\rho}_1| > \frac{2}{\sqrt{n}} \tag{4.16}$$

where n is the number of observations in the series and $1/\sqrt{n}$ is the approximate standard deviation of $\hat{\rho}_1$ [8, 65]. Let's define K as the number of lags needed to conclude autocorrelation. It follows that $K \leq \frac{n}{4}$ is big enough to detect any significant autocorrelation in the data [44].

4.3.2 Time series model and residuals

When ACF is not statistically significant, the following **Shewhart model** applies [44]:

$$x_t = \mu + e_t, \qquad t = 1, 2, \ldots \tag{4.17}$$

where e_t is the residuals term that is normally distributed with mean 0 and a constant standard deviation σ. When ACF is statistically significant, the model in Equation 4.17 no longer applies, unless autocorrelation was removed. We can remove autocorrelation by **sampling less frequently** since, as the time lag increases, this decreases autocorrelation. While this approach may work, a lot of process data are discarded, which may impose a longer period to detect out-of-control behaviors [44]. An alternative way is to

fit an autoregressive integrated moving average (ARIMA) model to data and then use the model residuals e_t to monitor the process using traditional control charts [44]. A typical ARIMA model is denoted as $ARIMA(p,d,q)$, where p characterizes the degrees of the autoregressive (AR) term, d represents the degrees of the differencing term, and q denotes the degrees of the moving average term. A term for the seasonal effects can also be added as appropriate [45]. For simplicity, we only consider $ARIMA(1,0,0)$ where all other arguments are zero except the AR term. This model is equivalent to $AR(1)$ and can be written this way:

$$x_t = \mu + \phi x_{t-1} + e_t \tag{4.18}$$

where μ is the stationary mean, x_t is the observation at time t, x_{t-1} is the previous observation at time $t-1$, and ϕ is a constant coefficient of autocorrelation that is between -1 and 1, and is obtained from fitting the model. In words, this model says that μ is the intercept and x_t depends on x_{t-1} at rate ϕ. We calculate, e_t, the residuals term, like this:

$$e_t = x_t - \hat{x}_t \tag{4.19}$$

where $\hat{x}_t$ is the fitted value of x_t that we obtain as follows:

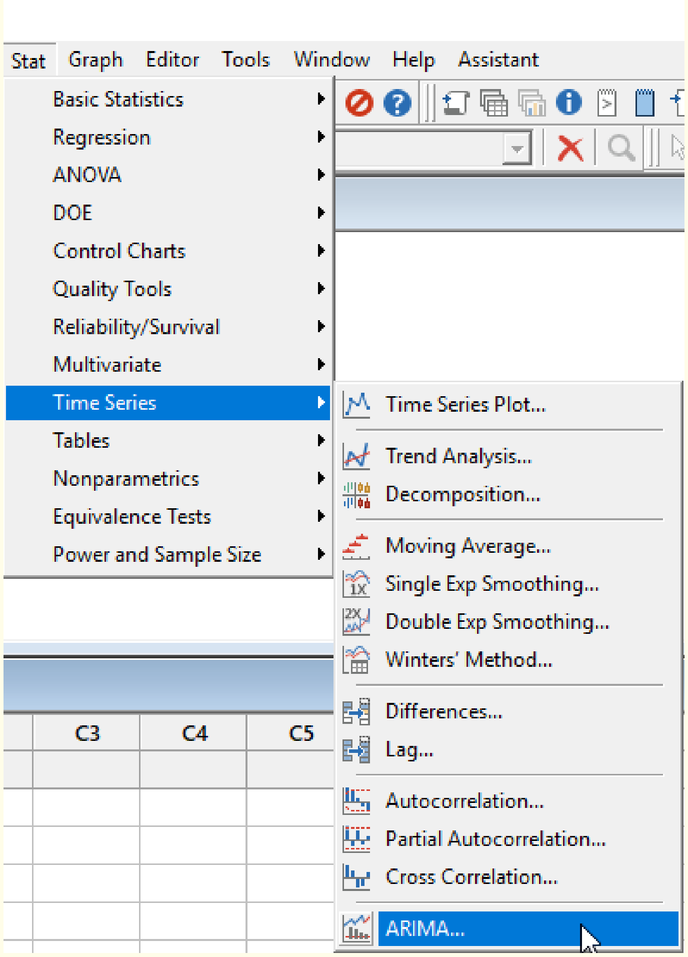
$$\hat{x}_t = \mu + \phi x_{t-1} \tag{4.20}$$

To fit an $ARIMA(1,0,0)$ model using Minitab, see How-To 4.7. To fit this model, using the Statsmodels module in Python, see How-To 4.8. It is also possible to use Excel to approximate a simple ARIMA model like $ARIMA(1,0,0)$. For that, we use the regression option as expressed in How-To 4.9.

How-To 4.7 (ACF and ARIMA (1,0,0) in Minitab 18)

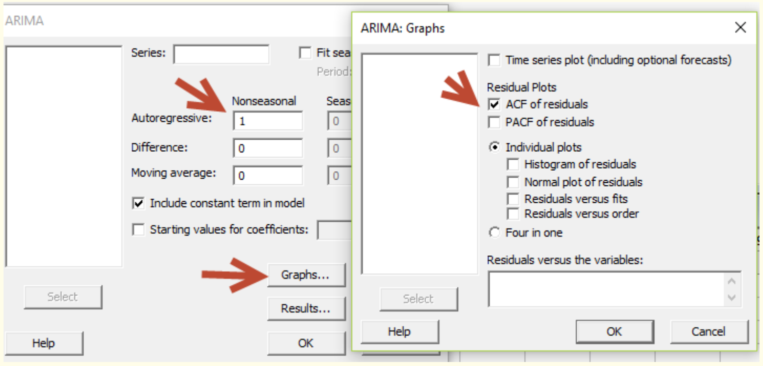
- To generate ACF in Minitab, click on *Stat > Time Series > Autocorrelation*. To fit an $ARIMA(1,0,0)$ model, click on *Stat > Time Series > ARIMA*. See the snapshot in Figure 4.7.

Figure 4.7: ARIMA, main screen, Minitab 18



- To view ACF for residuals, click on *Graphs > check ACF of residuals*. See the next snapshot

Figure 4.8: ARIMA, Autoregressive and Graphs screen, Minitab 18


ARIMA

Series:

□ Fit sea

Period:

	Nonseasonal	Seas
Autoregressive:	1	0
Difference:	0	0
Moving average:	0	0

☑ Include constant term in model

□ Starting values for coefficients:

ARIMA: Graphs

□ Time series plot (including optional forecasts)

Residual Plots

☑ ACF of residuals

□ PACF of residuals

● Individual plots

□ Histogram of residuals

□ Normal plot of residuals

□ Residuals versus fits

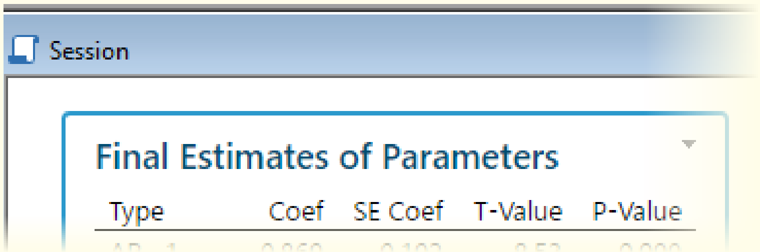
□ Residuals versus order

○ Four in one

Residuals versus the variables:

- Your ARIMA model coefficients will be displayed in a table titled **Final Estimates of Parameters**, as illustrated in the next snapshot.

Figure 4.9: ARIMA, Parameter Estimates, Minitab 18



Final Estimates of Parameters

Type	Coef	SE Coef	T-Value	P-Value

How-To 4.8 **(Python 3.6)**

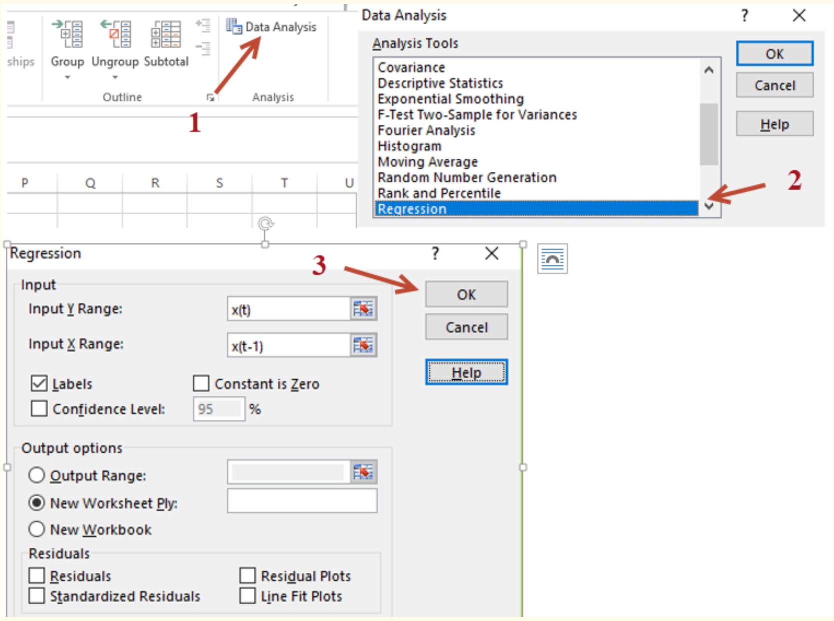
Script 4.3: A script for fitting an ARIMA model in Python 3.6

```
#ARIMA
#import modules
from pandas import *
from pylab import *
import seaborn as sns
import statsmodels.api as sm
from statsmodels.tsa.arima_model import ARIMA
from pandas.plotting import autocorrelation_plot
#import data from an Excel spreadsheet where xt is the column containing individual observations
data = read_excel('your directory')
series = data.xt
dates = sm.tsa.datetools.dates_from_range('1980m1', length=len(series))
y = series
arma_mod = sm.tsa.ARIMA(y, order=(1,0,0))
arma_res = arma_mod.fit()
print(arma_res.summary())
autocorrelation_plot(series)
pyplot.show()

#plot main results
fig = plt.figure()
ax1 = fig.add_subplot(111)
fig = sm.graphics.tsa.plot_acf(series.values.squeeze(), lags=25, ax=ax1)
sns.despine(offset=10, trim = False)
#set xticks to start from one since Python starts counting from zero
xticks(arange(len(series), step = 1), arange(0, len(series), step = 1))
#label y-axis and x-axis
ax1.set_xlabel('$Lag$ $k$')
ax1.set_ylabel(r'$\hat{\rho}_k$', size = 14)
show()
```

How-To 4.9 (Regression in Excel 2013)

- To fit a linear regression model using the Data Analysis add-in, click on *Data Analysis > Regression > Upload x_t data for InputY Range and x_{t-1} data for Input X Range > OK*. See the snapshot illustrated in Figure 4.10. If the Data Analysis add-in is not loaded, you can add it by clicking on *File >Options > Add-ins > Analysis ToolPak > Manage: Excel Add-in > Check the box of Analysis ToolPak >OK*.

Figure 4.10: Regression options in the Data Analysis add-in, Excel 2013



Input Y Range: x(t)

Input X Range: x(t-1)

☑ Labels ☐ Constant is Zero

☐ Confidence Level: 95 %

○ Output Range:

● New Worksheet Ply:

○ New Workbook

☐ Residuals ☐ Residual Plots

☐ Standardized Residuals ☐ Line Fit Plots

- To create a scatter plot between x_t and x_{t-1}, *select your data >Click on th* ***Insert*** *tab > Click on the* ***Scatter*** *plot in the charts menu*. To fit a linear function to the scatter plot, right-click on any point in the scatter plot and then click on **Add Trendline...**. To display the fitted linear function, select **Display Equation on chart**. You can also display the R^2 measure.

Example 4.3 (ARIMA(1,0,0))
The lab manager at General Hospital is interested in monitoring the process of assays for substance X. The current standard is 175 mml/l. Table 4.4 presents assay data for the past 26 days..

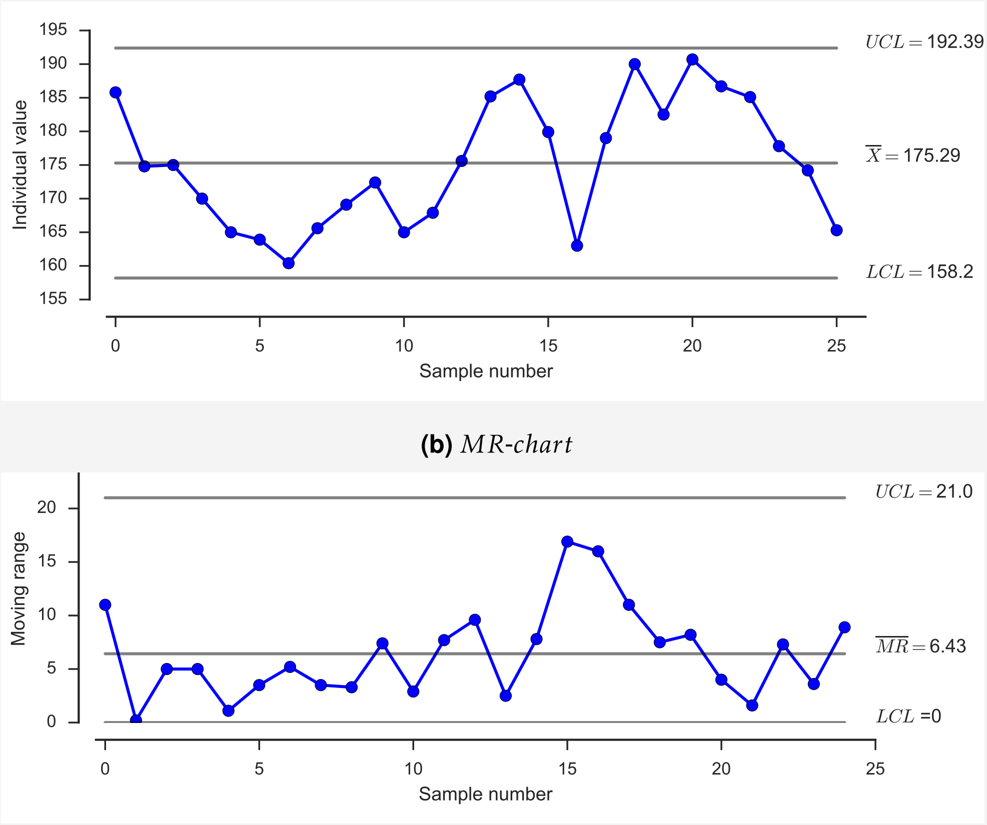
Table 4.4: Assay data in mml/l for substance X

t	x_t	t	x_t
1	185.8	*14*	185.2
2	174.8	*15*	187.7
3	175.0	*16*	179.9
4	170.0	*17*	163.0
5	165.0	*18*	179.0
6	163.9	*19*	190.0
7	160.4	*20*	182.5
8	165.6	*21*	190.7
9	169.1	*22*	186.7
10	172.4	*23*	185.1
11	165.0	*24*	177.8
12	167.9	*25*	174.2
13	175.6	*26*	165.3

Subsequently, the manager created ImR control charts as displayed in Figure 4.11.

Figure 4.11: Control chart for individual and moving ranges of the assay data in Table 4.4

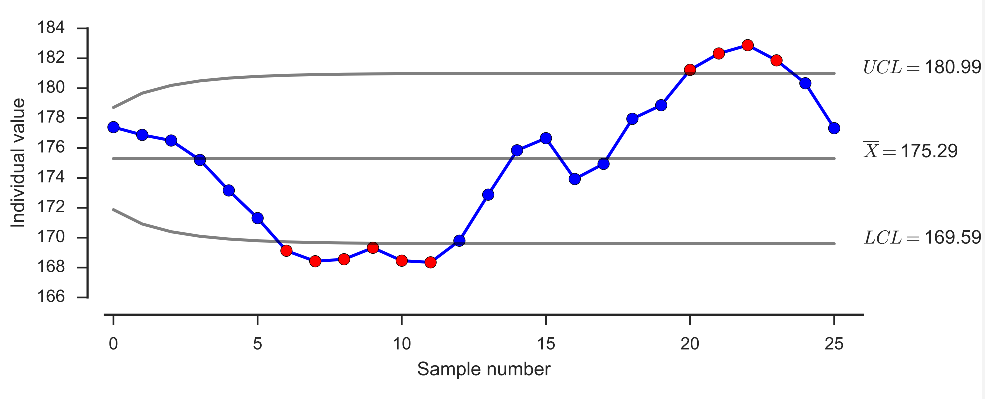
(a) I*-chart*



(b) MR*-chart*

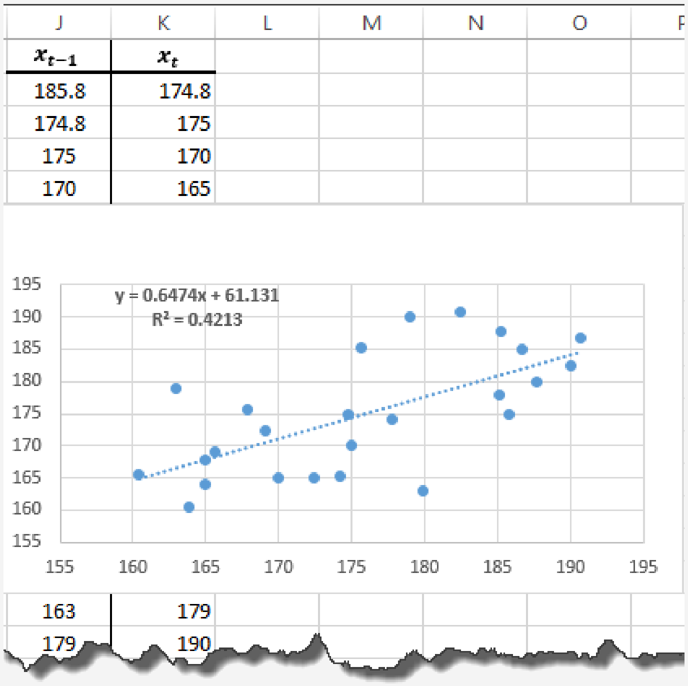
The manager did not detect any out-of-control behaviors in the ImR charts and was satisfied that the mean of individual observations was close to the target. But, the seemingly non-random behaviors in the $I\text{-}chart$ made the manager suspicious of small shifts or autocorrelation in the process. To validate this suspicion, the manager proceeded to set $\lambda = 0.2$ and calculate $\hat{\sigma} = \frac{\overline{MR}}{d_2}$ to create an EWMA chart shown in Figure 4.12.

Figure 4.12: EWMA control chart based on the data in Table 4.4



As the manager had suspected, the EWMA chart in Figure 4.12 showed several out-of-control points, likely due to small shifts in the process, but trends in the data persisted. To test for autocorrelation, the manager created a scatter plot between x_t and x_{t-1} as displayed in Figure 4.13.

Figure 4.13: A scatter plot of x_t and x_{t-1} in Excel based on the data in Table 4.4

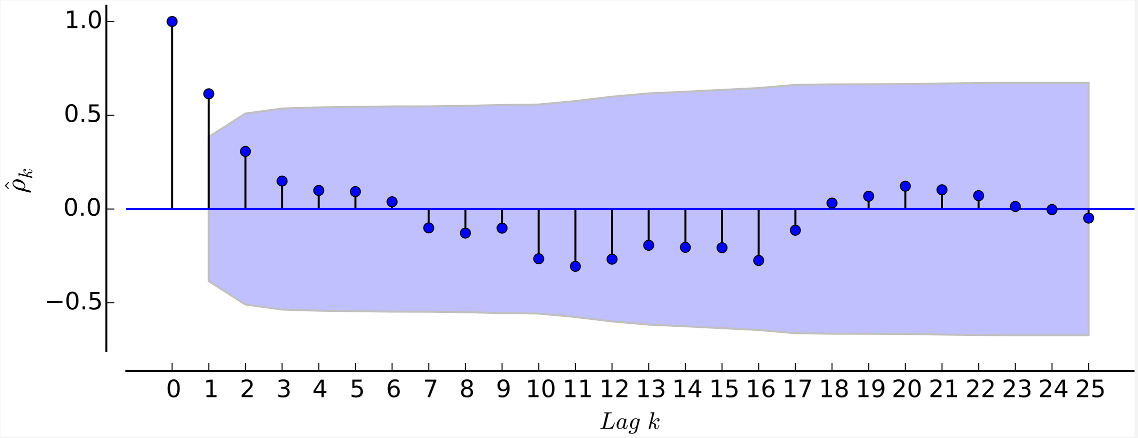


J	K	L	M	N	O	P
x_{t-1}	x_t					
185.8	174.8					
174.8	175					
175	170					
170	165					
163	179					
179	190					

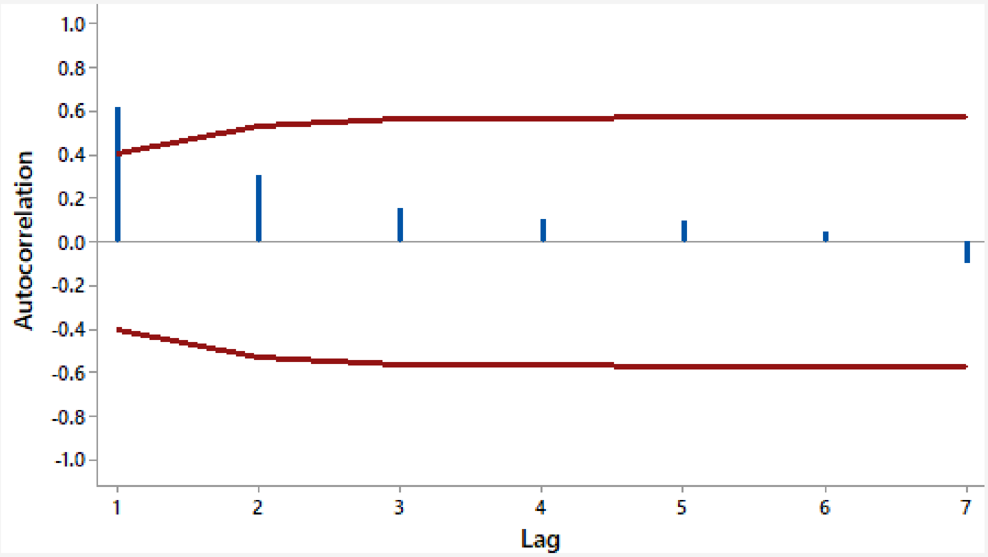
The scatter plot in Figure 4.13 shows a positive correlation between x_t and x_{t-1}. From this result, the manager decided to plot ACF in both Python and Minitab, as shown in Figure 4.14.

Figure 4.14: ACF of the data in Table 4.4

(a) ACF in Python



(b) ACF in Minitab 18



From the ACF plots in Figures 4.14, we notice that $\hat{\rho}_1$ falls outside of the two standard deviation limit, which makes it statistically significant at the 5% significance level. The result of $\hat{\rho}_0 = 1$ in Subfigure 4.14a is expected since the data is being covaried on itself (e.g., $Cov(x_t, x_t)$). No other lags of ACF are significant. Accordingly, the manager decided to fit $AR(1)$ model in Minitab as displayed in Table 4.5.

Table 4.5: Final Estimates of Parameters of $AR(1)$ in Minitab 18 based on the data in Table 4.4

Type	Coef	SE Coef	T-Value	P-Value
AR 1	0.68	0.156	4.36	0
Constant	56.06	1.41	39.7	0
Mean	175.33	4.42		

The first p-value in Table 4.5 indicates that the coefficient of the $AR(1)$ model is statistically significant. The actual $AR(1)$ model can be written as follows:

$$x_t = 56.06 + 0.68x_{t-1} + e_t \tag{4.21}$$

From this model, the manager obtained the residuals by $e_t = x_t - (56.06 + 0.68x_{t-1})$, as displayed in Table 4.6.

Table 4.6: Residuals (e_t) based on the $AR(1)$ model in Equation 4.21

t	x_t	$\hat{x}_t$	e_t
1	185.8		
2	174.8	182.404	-7.604
3	175	174.924	0.076
4	170	175.06	-5.06
5	165	171.66	-6.66
6	163.9	168.26	-4.36
7	160.4	167.512	-7.112
8	165.6	165.132	0.468
9	169.1	168.668	0.432
10	172.4	171.048	1.352
11	165	173.292	-8.292
12	167.9	168.26	-0.36
13	175.6	170.232	5.368
14	185.2	175.468	9.732
15	187.7	181.996	5.704
16	179.9	183.696	-3.796
17	163	178.392	-15.392
18	179	166.9	12.1
19	190	177.78	12.22
20	182.5	185.26	-2.76
21	190.7	180.16	10.54
22	186.7	185.736	0.964
23	185.1	183.016	2.084
24	177.8	181.928	-4.128
25	174.2	176.964	-2.764
26	165.3	174.516	-9.216

The ACF of the residuals shows no further autocorrelation, as Figure 4.15 indicates. Accordingly, the manager has decided to monitor the process using the EWMA chart on the residuals of the $AR(1)$ model. The resulting chart is illustrated in Figure 4.16. This new chart indicates that the process is stable.

Figure 4.15: ACF of Residuals in Minitab 18

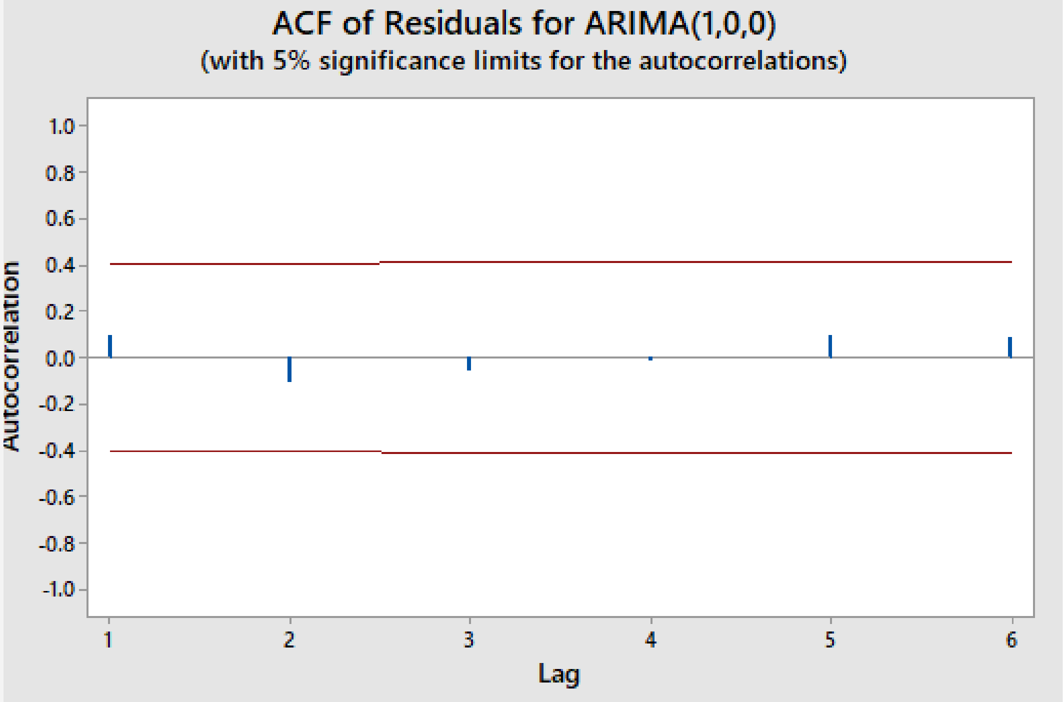
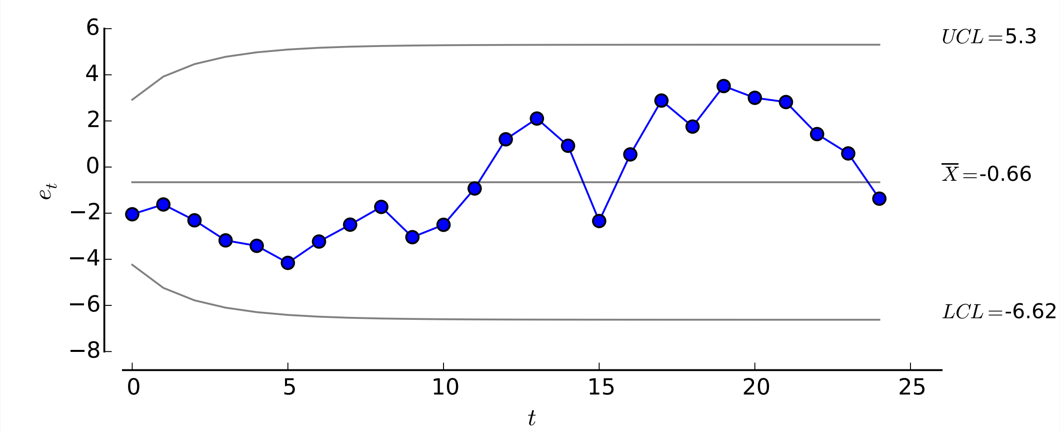


Figure 4.16: EWMA control chart based on the residuals in Table 4.6



Following instructions in How-To 4.9, we approximate the $AR(1)$ model in Excel using the regression option. We use x_{t-1} for Input X Range and x_t for Input Y Range. The summary of Excel outputs is presented in Table 4.7.

Table 4.7: A summary of the regression output for the process x_t, Excel 2013

Regression Statistics	
Multiple R	0.649112
R Square	0.421346
Adjusted R Square	0.396187
Standard Error	7.199552
Observations	25

ANOVA

	df	*SS*	*MS*	*F*	*Significance F*
Regression	1	868.0786	868.0786	16.74743	0.00044715
Residual	23	1192.172	51.83356		
Total	24	2060.25			

	Coefficients	*Standard Error*	*t Stat*	*P-value*	*Lower 95%*	*Upper 95%*	*Lower 95.0%*	*Upper 95.0%*
Intercept	**61.13115**	27.83072	2.196535	0.038398	3.558911799	118.7034	3.558912	118.7034
AR(1)	**0.647388**	0.158194	4.092362	0.000447	0.320138207	0.974637	0.320138	0.974637

From Table 4.7, we notice that the coefficients of this model are slightly different, but this does not affect the significance of the model since the p-value is still less than 0.05. Additionally, comparing the model coefficient to two standard deviations, given by $2/\sqrt{25} = 0.4$, we observe that $0.65 > 0.4$. Hence per Equation 4.16, we can still conclude that the $AR(1)$ model in Excel is statistically significant. To run the ARIMA (1,0,0) using Python, use the script in How-To 4.8.

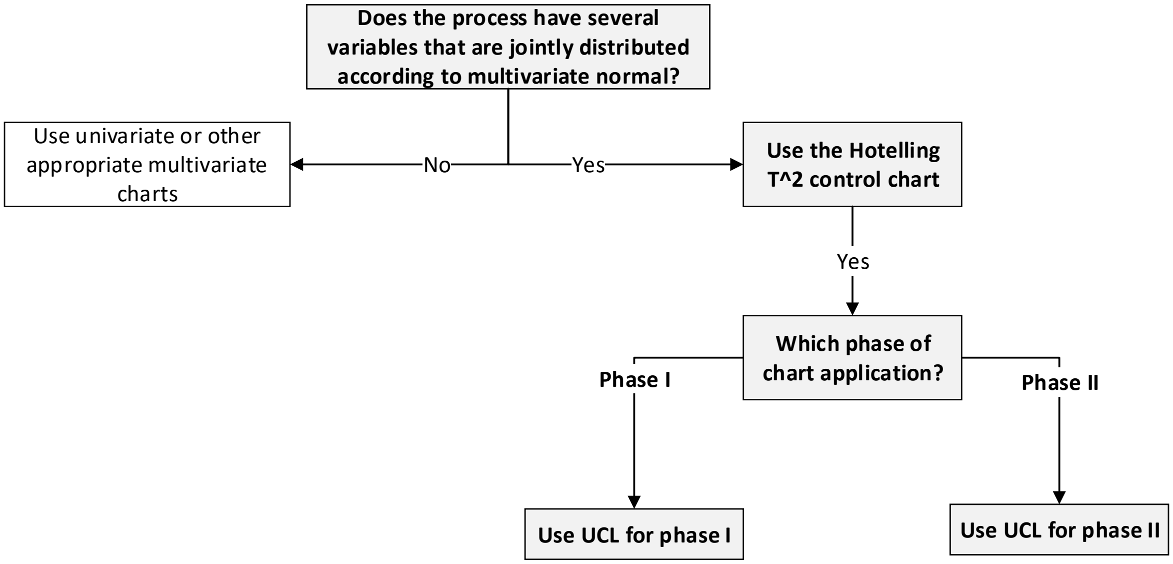
4.4 Multivariate control charts

In this section, we discuss how to monitor a process with dependent variables using **multivariate** control charts. In the previous section, we discussed how dependence in the process samples could lead to incorrect control results if one failed to account for autocorrelation. The same outcome is likely if a multivariate process has dependent variables, and one monitors them separately, instead jointly. In general, paired variables based on the same process are likely to be dependent. But, if these variables are found to be independent, they can be monitored disjointedly.

Several techniques exist for monitoring multivariate processes. Examples include multivariate EWMA charts to detect small shifts in the process [44], and multivariate attribute control charts to monitor processes with dependent binomial variables [40]. Here, we focus on one type of multivariate charts known as **Hotelling T^2**. This chart is the multivariate version of the $X\text{-}bar$ Shewhart control chart. We apply the T^2 chart when variables are jointly distributed according to a **multivariate normal** random variable [44].

Figure 4.17 shows a basic decision tree for deciding whether the T^2 chart is appropriate.

Figure 4.17: When is the T^2 chart appropriate?


Does the process have several variables that are jointly distributed according to multivariate normal?
No
Yes
Use univariate or other appropriate multivariate charts
Use the Hotelling T^2 control chart
Yes
Which phase of chart application?
Phase I
Phase II
Use UCL for phase I
Use UCL for phase II

Next, we present possible scenarios of health care processes where multivariate process control charts may be applicable.

Scenario 1: A quality manager at Metropolis Imaging Center is interested in monitoring the amount of radiation that patients are exposed to during cardiac angiogram procedures. The variables of interest are the dose area product (measured in mGy-cm squared)[7], fluoroscopy time (measured in minutes), and the number of digital images (measured using the count of frames captured). Since these variables are related according to a multivariate normal distribution, the manager has decided to use the T^2 chart to monitor this process. See Waterhouse et al. (2010)[64] for further discussion about the use of multivariate control charts to monitor radiation in patients.

Scenario 2: An office manager at a community clinic is trying to monitor the health status of patients with chronic respiratory conditions. From consulting with clinicians, the manager decided to control three output variables: partial pressure oxygen (PaO2), partial pressure carbon dioxide (PaCO2), and the body mass index (BMI). Since these variables are believed to be correlated and jointly distributed according to a multivariate normal random variable, the manager decided to apply the T^2 chart to monitor this process. See Correia (2011) [20] for further discussion about the use of multivariate control charts to monitor the health status of patients with chronic respiratory conditions.

Scenario 3: A clinical data manager at General Hospital is concerned about data entry errors and would like to set up control charts to monitor the accuracy, completeness, and consistency of the data captured at the point-of-care. Since these variables are

[7] mGy-cm: milligray- centimeter

likely dependent, the manager is looking into applying multivariate control charts for attribute data, as suggested in Lu (1998) [40]. For more discussion about the use of control charts to monitor data quality, see Jones et al. (2014) [33].

4.4.1 How to set up the T^2 chart?

Monitoring mean behaviors

Let's assume that a process has a $p-$component vector of x variables that are jointly distributed according to the multivariate normal denoted as follows:

$$f(x) = \frac{1}{(2\pi)^{p/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu)} \tag{4.22}$$

where the $'$ symbol signifies transpose, $|\Sigma|$ denotes the determinant of the $p \times p$ covariance matrix Σ, and μ is a $p \times 1$ mean vector. When μ and Σ are given, we monitor:

$$\chi_0^2 = n(\bar{x} - \mu)\Sigma^{-1}(\bar{x} - \mu) \tag{4.23}$$

where $\bar{x}$ is a $p \times 1$ mean vector, and n is the constant sample size. The statistic χ_0^2 is distributed according to Chi-square (χ^2). The corresponding LCL is always zero, and we obtain UCL as follows:

$$UCL = \chi_{\alpha,p}^2 \tag{4.24}$$

Here, α is the upper percentage of the χ^2 distribution, and p denotes the degrees of freedom [44].

When μ and Σ are not known, we can estimate them from the process using $\bar{\bar{x}}$ and S, where $\bar{\bar{x}}$ is a p-component vector of sample means and S is the corresponding covariance matrix that is determined from averages of sample variances and covariances [44]. In this case, the T^2 statistic that we monitor is given by:

$$T^2 = n(\bar{x} - \bar{\bar{x}})'S^{-1}(\bar{x} - \bar{\bar{x}}) \tag{4.25}$$

The formulation in Equation 4.25 is referred to as **Hotelling**. This chart only has one limit, the UCL, and no centerline. But, one can always use the median of the data for the centerline, as it is done in Minitab 18. Box 4.3 summarizes techniques for calculating the UCL of the Hotelling T^2 chart when $n > 1$. Box 4.4 presents similar calculations when $n = 1$.

Box 4.3 Hotelling T^2 control chart when $n > 1$

When $n > 1$, we obtain the UCL of the T^2 chart as follows:

$$UCL_{PhaseI} = \frac{p(m-1)(n-1)}{mn-m-p+1} F_{\alpha,p,mn-m-p+1} \quad (4.26)$$

$$UCL_{PhaseII} = \frac{p(m+1)(n-1)}{mn-m-p+1} F_{\alpha,p,mn-m-p+1} \quad (4.27)$$

where α is the upper percentage of F distribution, m is the number of subgroups (each with size $n > 1$), and p is the number of variables. For a large m (e.g., $m > 100$), we set up the UCL per Equation 4.24 [44].

Box 4.4 Hotelling T^2 control chart when $n = 1$

When $n = 1$, we obtain the UCL of the T^2 chart as follows:

$$UCL_{PhaseI} = \frac{p(m-1)}{m-p} F_{\alpha,p,m-p} \quad (4.28)$$

$$UCL_{PhaseII} = \frac{(m-1)^2}{m} \beta_{\alpha,p/2,(m-p-1)/2} \quad (4.29)$$

where α is the upper percentage of F and β distributions, m is the number of subgroups (each with size $n = 1$), and p is the number of variables. For a large m (e.g., $m > 100$) we set up the UCL per Equation 4.24 [44].

Monitoring variability

Monitoring variability in multivariate processes requires monitoring the determinant of the covariance matrix in each sample. To do that, we apply the **generalized variance (GV)** control chart that we construct per Box 4.5 [44].

Box 4.5 GV control chart

Given sample i and covariance matrix S_i, the statistic to be monitored is:

$$|S_i| \tag{4.30}$$

where the |.| symbols here indicate determinant, not an absolute value. The control limits are obtained by:

$$\begin{aligned} UCL &= \frac{|S|}{b1}\left(b_1 + 3\sqrt{b2}\right) && (4.31) \\ CL &= |S| && (4.32) \\ LCL &= \frac{|S|}{b1}\left(b_1 - 3\sqrt{b2}\right) && (4.33) \end{aligned}$$

where S is the overall covariance matrix and

$$\begin{aligned} b_1 &= \frac{1}{(n-1)^p}\prod_{i=1}^{p}(n-i) && (4.34) \\ b_2 &= \frac{1}{(n-1)^{2p}}\prod_{i=1}^{p}(n-i)\left[\prod_{j=1}^{p}(n-j+2)-\prod_{j=1}^{p}(n-j)\right] && (4.35) \end{aligned}$$

4.4.2 How does the T^2 chart work?

Setup

Unlike in univariate control charts where single variable calculus is applicable, much of the mathematical machinery behind multivariate control relates to linear algebra. This circumstance makes the computation of the control statistics a bit different than in the univariate case. We demonstrate with a bivariate case of individual observations where the related variables are x_1 and x_2. We obtain T^2 as follows:

$$T^2 = n(x-\bar{x})'S^{-1}(x-\bar{x}) \tag{4.36}$$

From Equation 4.36, x is a vector of x_1 and x_2 given by $x' = [x_1, x_2]$, and $\bar{x}$ is a mean vector of $\bar{x}_1$ and $\bar{x}_2$ given by $\bar{x}' = [\bar{x}_1, \bar{x}_2]$. Let s_1^2 be the variance of x_1, s_2^2 the variance of x_2, and $s_{21} = s_{12}$ the covariance of x_1 and x_2. We obtain the covariance matrix S this way:

$$S = \begin{bmatrix} s_1^2 & s_{12} \\ s_{12} & s_2^2 \end{bmatrix} \tag{4.37}$$

We obtain the inverse of S, denoted as S^{-1}, like this [36]:

$$S^{-1} = \begin{bmatrix} s_1^2 & s_{12} \\ s_{12} & s_2^2 \end{bmatrix}^{-1} = \frac{1}{|S|}\begin{bmatrix} s_2^2 & -s_{12} \\ -s_{12} & s_1^2 \end{bmatrix} = \frac{1}{s_1^2 s_2^2 - s_{12}^2}\begin{bmatrix} s_2^2 & -s_{12} \\ -s_{12} & s_1^2 \end{bmatrix} \tag{4.38}$$

Subsequently, we can compute the T^2 statistic this way:

$$\begin{aligned} T^2 &= n[x_1 - \bar{x}_1, x_2 - \bar{x}_2]\begin{bmatrix} s_1^2 & s_{12} \\ s_{12} & s_2^2 \end{bmatrix}^{-1}\begin{bmatrix} x_1 - \bar{x}_1 \\ x_2 - \bar{x}_2 \end{bmatrix} && (4.39) \\ &= \frac{n}{s_1^2 s_2^2 - s_{12}^2}[x_1 - \bar{x}_1, x_2 - \bar{x}_2]\begin{bmatrix} s_2^2 & -s_{12} \\ -s_{12} & s_1^2 \end{bmatrix}\begin{bmatrix} x_1 - \bar{x}_1 \\ x_2 - \bar{x}_2 \end{bmatrix} && (4.40) \\ &= \frac{n}{s_1^2 s_2^2 - s_{12}^2}\left[s_2^2(x_1 - \bar{x}_1)^2 + s_1^2(x_2 - \bar{x}_2)^2 - 2s_{12}(x_1 - \bar{x}_1)(x_2 - \bar{x}_2)\right] && (4.41) \end{aligned}$$

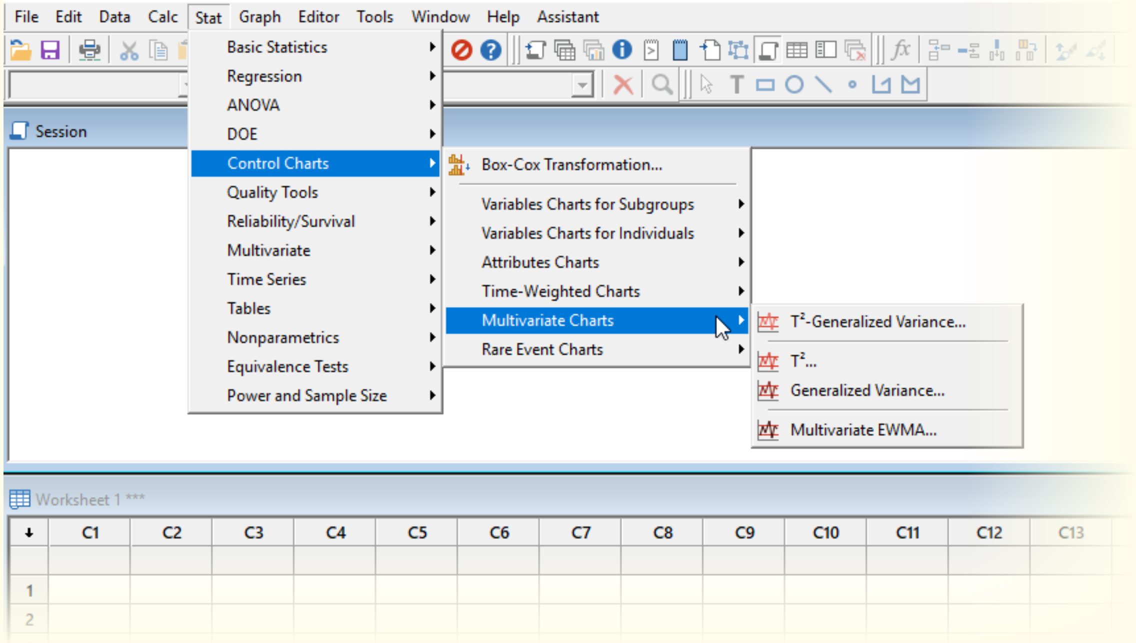
To compute the T_i^2 statistic for sample i, we substitute x_1 and x_2 with their respective numeric values. All other parameters remain constant.

Interpretation

The detection of out-of-control behaviors in multivariate control charts is as before. That is, if the test statistic falls outside of the UCL, the process has special cause variation. But, unlike in the univariate case, it is challenging to pinpoint assignable causes in multivariate control charts due to the interdependence of the variables. The GV chart adds one more layer of complexity since the determinant of the covariance in different samples may be the same even though the variances in respective variables are quite different. Hence, it is recommended to monitor the variance of each variable, in addition to monitoring the joint generalized variance [44].

How-To 4.10 (Multivariate charts in Minitab 18)
Click on *Stat* > *Multivariate Charts*. See the snapshot in Figure 4.18.

Figure 4.18: Multivariate control charts in Minitab 18



How-To 4.11 **(Python 3.6)**

Script 4.4: A script for creating a multivariate control chart using Python 3.6

```
#MUTLIVARIATE CONTROL CHART
#import modules
import scipy.stats
from pandas import *
from pylab import *
import operator
import seaborn as sns
from functools import reduce
data = read_excel('your directory')
#iniliaze parameters and statistics for the control T^2 chart
u1 = data.mean()['Coding']
u2 = data.mean()['Abstracting']
s1 = data.mean()['VarC']
s2 = data.mean()['VarA']
s12 = data.mean()['CovCA']
n = 10 #given sample size in Example 4.4. Change this parameter as
    necessary
for i, j in zip(data.Coding, data.Abstracting):
    T2 = (n/((s1*s2) -s12**2))*(s2*(i-u1)**2 + s1*(j - u2)**2 - 2*s12
        *(i - u1)*(j-u2))
df_cov = DataFrame(array([[s1, s12],[s12,s2]]))
df_inv = array(DataFrame(np.linalg.inv(df_cov.values), df_cov.columns
    , df_cov.index))
df_det = np.linalg.det(df_cov.values)
p = len(df_cov)
```

```
m = len(data)*1.
a = p*(m-1)*(n-1.)
b = m*n-m-p+1.
d = a/float(b)
v1 = p
v2 = b
alpha = 0.001
F = scipy.stats.f.ppf(q=1-alpha, dfn=v1, dfd=v2)
#calculate control limits
UCLm = [d*F]*len(data)
LCLm = [0]*len(data)
#mark red the point that falls outside of the UCL. Otherwise, mark
    the point blue.
mv = []
markers = []
colors = []
for i in range (len(data)):
    x1 = data.ix[i]['Coding']
    x2 = data.ix[i]['Abstracting']
    row = array([x1-u1,x2-u2])
    dot1 = dot(df_inv,row)
    dot2 = dot(row.T,dot1)
    mv.append(dot2*n)
    if dot2*n > UCLm[0]:
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
#Plotting the T^2 chart
fig=figure()
ax1 =  fig.add_subplot(111)
t = arange(len(data))
ax1.plot(UCLm, 'k-', alpha = 0.5)
ax1.plot(LCLm, 'k-',alpha = 0.5)
ax1.plot(mv,'b-',zorder=1)
for x,y,c,m in zip(t, mv, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
ax1.set_xlabel('Sample number')
ax1.set_ylabel('$T^2$')
ax1.annotate ('$UCL$', xy = (xlim()[1], list(UCLm)[-1]), xytext = (
    xlim()[1],list(UCLm)[-1]))
ax1.annotate ('$LCL$', xy = (xlim()[1], list(LCLm)[-1]), xytext = (
    xlim()[1],list(LCLm)[-1]))
#set xticks to start from one since Python starts counting from zero
N = len(UCLm)
xticks(arange(N, step = 1), arange(1, N+1, step = 1))
show()
#
    xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

```
# GV chart for variability
prodlist1 = []
prodlist2 = []
for i in range(1, p+1):
    prodlist1.append(n-i)
for i in range(1, p+1):
    prodlist2.append(n-i + 2.)
prod1 = reduce(operator.mul, prodlist1, 1)*1.
prod2 = reduce(operator.mul, prodlist2, 1)*1.
b1 = (1./((n-1.)**p))*prod1
b2 = ((1./((n-1.)**(2*p)))*prod1)*(prod2 - prod1)
UCLgv = [(df_det/b1)*(b1+3.*sqrt(b2))]*len(data)
LCLgv = [max(0,(df_det/b1)*(b1-3.*sqrt(b2)))]*len(data)
CLgv = [df_det]*len(data)
#mark red the point that falls outside of the UCL. Otherwise, mark
    the point blue.
gv = []
markers = []
colors = []
for i in range(len(data)):
    s1i = float(data.ix[i]['VarC'])
    s2i = float(data.ix[i]['VarA'])
    s12i = float(data.ix[i]['CovCA'])
    Si = DataFrame(array([[s1i, s12i],[s12i,s2i]]))
    Ai = Si
    DAi = np.linalg.det(Ai)
    gv.append(DAi)
    if (DAi > UCLgv[0] or  DAi < LCLgv[0]):
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
#plot the GV chart
fig=figure()
ax1 =  fig.add_subplot(111)
t = arange(len(data))
ax1.plot(UCLgv, 'k-', alpha = 0.5)
ax1.plot(LCLgv, 'k-',alpha = 0.5)
ax1.plot(CLgv, 'k-',alpha = 0.5)
ax1.plot(gv,'b-',zorder=1)
for x,y,c,m in zip(t, gv, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
ax1.set_xlabel('Sample number ($i$)')
ax1.set_ylabel('$|S_i|$')
ax1.annotate ('$UCL$', xy = (xlim()[1], list(UCLgv)[-1]), xytext = (
    xlim()[1],list(UCLgv)[-1]))
ax1.annotate ('$CL$', xy = (xlim()[1], list(CLgv)[-1]), xytext = (
    xlim()[1],list(CLgv)[-1]))
ax1.annotate ('$LCL$', xy = (xlim()[1], list(LCLgv)[-1]), xytext = (
    xlim()[1],list(LCLgv)[-1]))
```

```
#set xticks to start from one since Python starts counting from zero
N = len(UCLgv)
xticks(arange(N, step = 1), arange(1, N+1, step = 1))
show()
```

Example 4.4 (Multivariate control in medical coding)
A medical coding process involves two major sub-processes: the transformation of medical diagnoses and surgeries into codes, which is referred to as **coding**, and the reporting of administrative data (e.g., patient discharge disposition), which is known to as **abstracting** [11].
The medical coding manager at Metropolis Hospital would like to monitor the daily averages of the time it takes to code and abstract an inpatient record. Given that these two sub-processes are likely correlated, the manager has decided to use the T^2 chart to monitor the process. It is assumed that the performance of all coders is statistically the same. Additionally, it is assumed that patient records are statistically independent. To proceed, the manager sampled 10 inpatient records daily and determined the average, variance, and covariance statistics for coding (variable x_1) and abstracting (variable x_2). The data that the manager has collected so far are displayed in Table 4.8. Here, $\bar{x}_1$ and s_1^2 represent the average and variance of the coding time, and $\bar{x}_2$ and s_2^2 represent the average and variance of the abstracting time. The daily covariance is denoted by s_{12}.

Table 4.8: Daily statistics for coding (x_1) and abstracting (x_2) at Metropolis Hospital

Sample#	$\bar{x}_1$	$\bar{x}_2$	s_1^2	s_2^2	s_{12}
1	17	5	30	17	15
2	13	5	22	19	18
3	14	2	21	9	8
4	16	3	30	17	15
5	19	9	31	11	10
6	14	4	20	20	19
7	13	3	16	19	15
8	20	5	31	12	11
9	15	5	16	18	14
10	16	5	20	6	4
11	13	4	25	14	13
12	13	5	10	7	5
13	18	6	28	15	14
14	19	4	24	12	11
15	18	7	10	12	8
16	14	5	15	8	6
17	19	7	10	7	5
18	14	4	19	15	14
19	12	5	31	11	10
20	20	4	32	12	11
21	14	5	24	19	18
22	13	4	21	7	5
23	14	4	21	8	6
24	17	4	15	7	6
25	14	4	28	7	6
26	17	5	13	20	11
27	20	6	30	16	14
28	13	3	31	8	7
29	14	5	28	17	15
30	13	3	11	8	7

To create the T^2 chart, the manager applied the Python script in How-To 4.11 and obtained the following statistics:

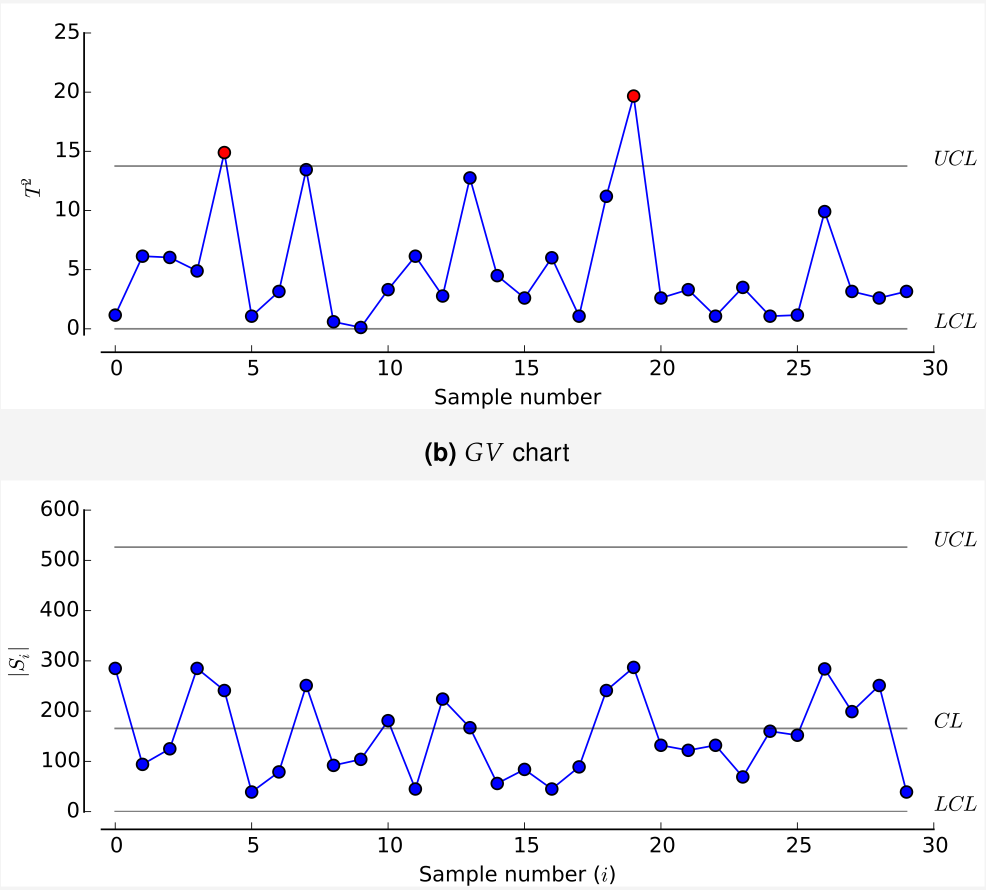
$$\bar{\bar{x}} = \begin{bmatrix} \bar{\bar{x}}_1 \\ \bar{\bar{x}}_2 \end{bmatrix} = \begin{bmatrix} 16 \\ 5 \end{bmatrix} \quad S = \begin{bmatrix} \bar{s}_1^2 & \bar{s}_{12} \\ \bar{s}_{12} & \bar{s}_2^2 \end{bmatrix} = \begin{bmatrix} 22.1 & 10.7 \\ 10.7 & 12.6 \end{bmatrix} \tag{4.42}$$

Since the process is still in phase I, the manager established UCL_{phaseI} as follows:

$$\begin{aligned} UCL &= \frac{p(m-1)(n-1)}{mn-m-p+1} F_{\alpha,p,mn-m-p+1} && (4.43) \\ &= \frac{2(30-1)(2-1)}{30(2)-30-2+1} F_{0.001,2,269} && (4.44) \\ &= \frac{522}{269}(7.088) && (4.45) \\ &= 13.75 && (4.46) \end{aligned}$$

where *n = 10, p = 2, m = 30*, and $\alpha = 0.001$. The value of 7.088 was determined using the Scipy module in Python, but an approximate value can be found in any F table with $\alpha = 0.001$, $v_1 = 2$, and $v_2 = 269$. The resulting T^2 chart is portrayed in Figure 4.19a. The corresponding GV chart is illustrated in Figure 4.19b.

Figure 4.19: T^2 and GV charts based on the data in Table 4.8



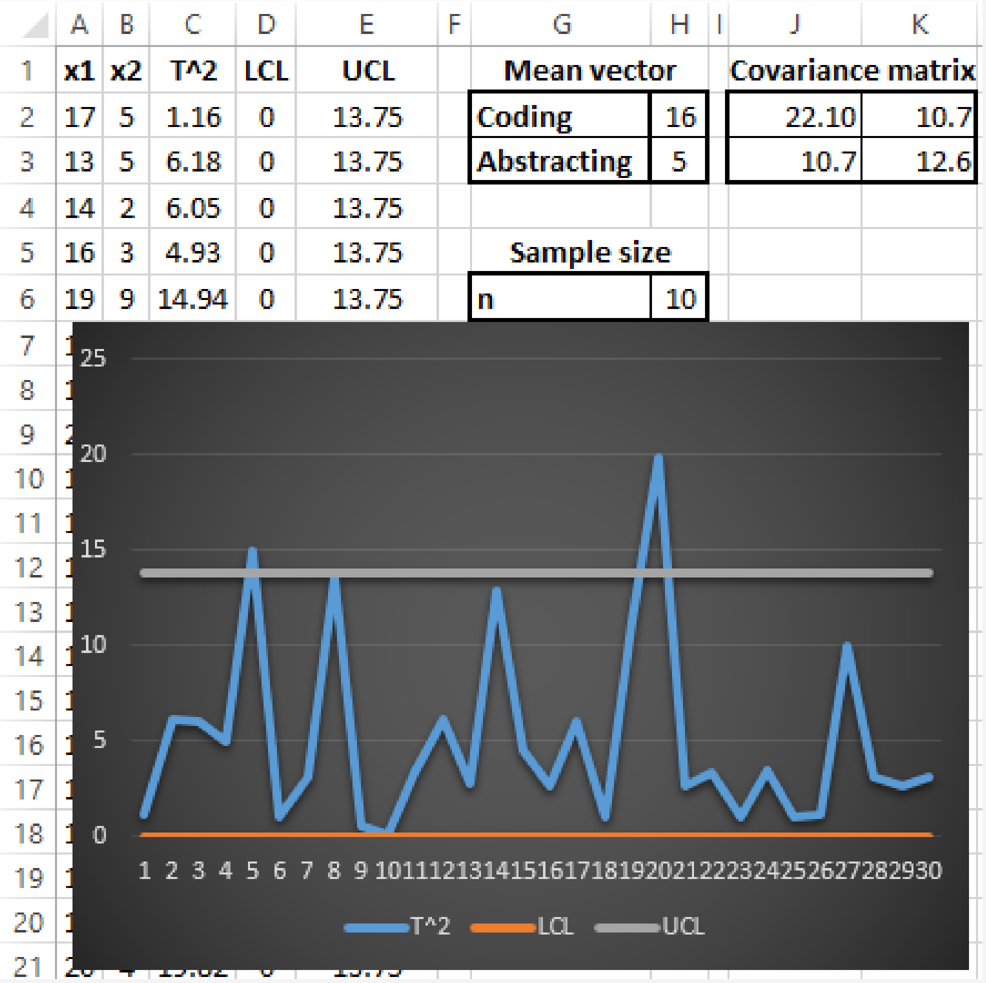
(a) T^2 chart

(b) GV chart

The dynamics of the chart in Figure 4.19a reveal two out-of-control points. The manager is going to investigate the cause of this special cause variation. After fixing any assignable issues, the manager will omit these points and create new control limits to monitor the future process. The GV chart in Figure 4.19b is stable.

We now use Excel to reproduce the T^2 chart in Figure 4.19a. To do that, we begin by setting up our Excel spreadsheet, as shown in Figure 4.20.

Figure 4.20: A setup of Excel to reproduce the T^2 chart based on the data in Table 4.8



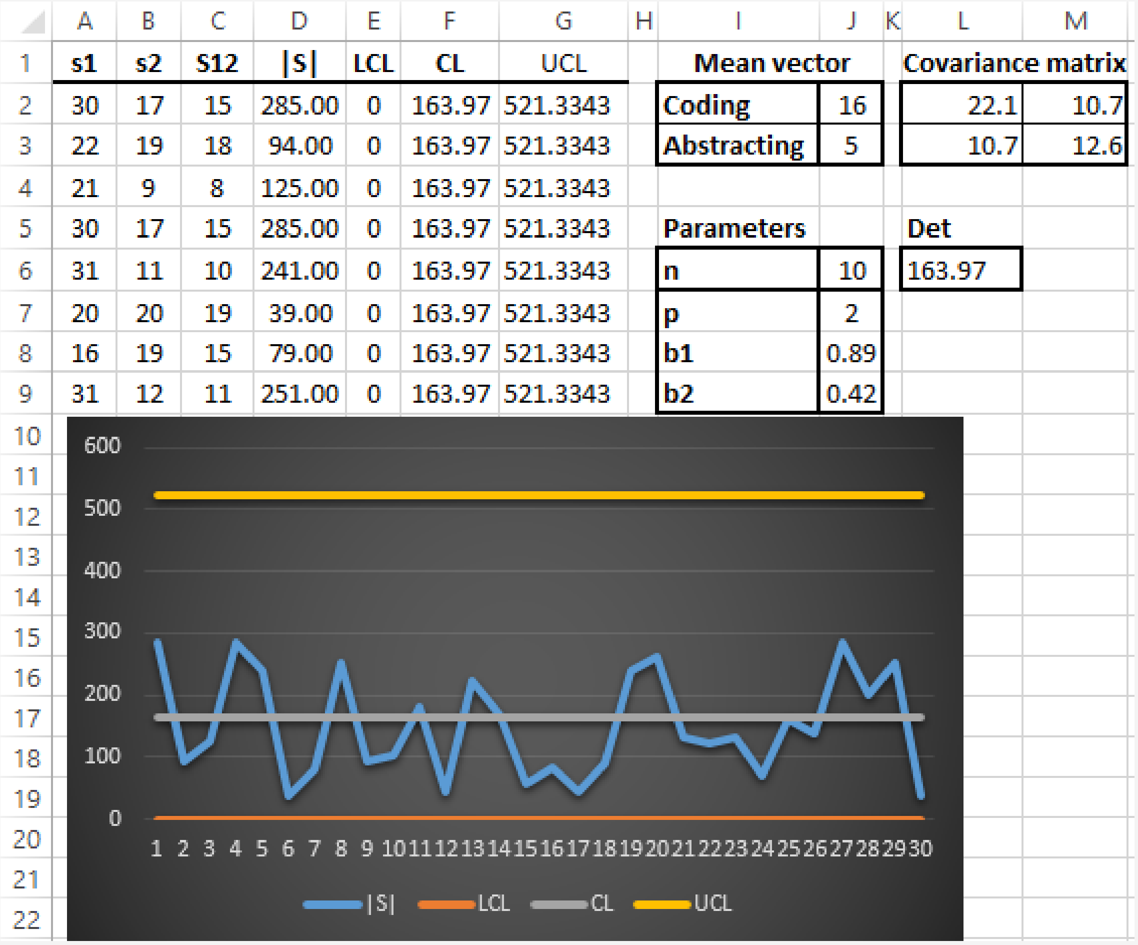
	A	B	C	D	E	F	G	H	I	J	K
1	x1	x2	T^2	LCL	UCL		Mean vector			Covariance matrix	
2	17	5	1.16	0	13.75		Coding	16		22.10	10.7
3	13	5	6.18	0	13.75		Abstracting	5		10.7	12.6
4	14	2	6.05	0	13.75						
5	16	3	4.93	0	13.75		Sample size				
6	19	9	14.94	0	13.75		n	10			

We calculated the T^2 statistic for each day. For example the T^2 value in cell C2 was obtained by: =(H6/(J2*K3-J3*K2))*((K3*(A2-H2)*(A2-H2))+(J2*(B2-H3)*(B2-H3))-2*(K2*(A2-H2)*(B2-H3))). We determined the diagonal elements of the covariance matrix in Figure 4.20 by taking the averages of s_1 and s_2 to obtain 22.10 and 12.6, respectively. The off-diagonal element is the average of the s_{12} column, which equals 10.7. The mean vector is composed of averages of x_1 and x_2. We programmed the UCL formula for phase I per Box 4.3. As before, *n = 10, p*

= 2, *m* = 30 and the F value at $\alpha = 0.001$ is 7.088.

To reproduce the GV chart in Excel, we start by setting up our spreadsheet as pictured in Figure 4.21.

Figure 4.21: A setup of Excel to reproduce the GV chart based on the data in Table 4.8



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	s1	s2	S12	|S|	LCL	CL	UCL		Mean vector			Covariance matrix	
2	30	17	15	285.00	0	163.97	521.3343		Coding	16		22.1	10.7
3	22	19	18	94.00	0	163.97	521.3343		Abstracting	5		10.7	12.6
4	21	9	8	125.00	0	163.97	521.3343						
5	30	17	15	285.00	0	163.97	521.3343		Parameters			Det	
6	31	11	10	241.00	0	163.97	521.3343		n	10		163.97	
7	20	20	19	39.00	0	163.97	521.3343		p	2			
8	16	19	15	79.00	0	163.97	521.3343		b1	0.89			
9	31	12	11	251.00	0	163.97	521.3343		b2	0.42			

We calculated the GV statistics per Box 4.5. For example, we determined row 2 in Figure 4.21 as follows:

$$UCL = ((L2*M3-L3*M2)/J8)*(J8+3*SQRT(J9)) \quad (4.47)$$

$$CL = L2*M3-L3*M2 \quad (4.48)$$

$$|S| = A2*B2-C2*C2 \quad (4.49)$$

We set LCL to zero since it was originally negative.

4.5 EXERCISES

1. Use Excel to recreate both the risk-adjusted and traditional p charts in Example 4.1.

2. Use Excel to recreate the risk-adjusted CUSUM in Example 4.2.

3. The administrator of Metropolis Skilled Nursing Facility is interested in using risk-adjusted control charts to monitor the rate of resident falls. To estimate the risk, the administrator will use the Morse Fall Scale [52] of each resident. Table 4.9 presents the data that the manager has collected so far. The $Census$ column indicates the total number of residents on any given day. The $Falls$ column shows the daily number of residents who fell at least once. The $Risk$ column contains the averages of the daily probabilistic measures of the Morse Fall Scale, and the $StDev$ column represents the corresponding standard deviation.

   (a) Estimate the process standard deviation

   (b) Create a risk-adjusted p-chart with $L = 2$

   (c) Create a traditional p-chart with $L = 2$. Explain the difference in your p charts.

   (d) Create a risk-adjusted CUSUM chart of the expected daily falls. Use $h = 4.5$ and $OR = 2$ for upper CUSUM, and $OR = 0.5$ for lower CUSUM. What can you conclude?

4. Using Minitab, Excel, and Statsmodels in Python, recreate the results in Example 4.3.

5. The data in Table 4.10 represents queue times, in minutes, of IT tickets related to a clinical decision support system (CDSS) at Metropolis Hospital. The CIO of this hospital has set a goal of 30 minutes, on average, to respond to users' questions.

   (a) Using Excel or Minitab, create ImR and $EWMA$ control charts of this process. What can you conclude?

   (b) Using Python Statsmodels or Minitab, fit AR(1) and AR(2) models. What can you conclude from your results?

   (c) Create the final control chart to monitor the process. Discuss your results.

6. Consider the sample data of a bivariate process in Table 4.11, where the $\bar{x}_1$ column represents the mean behavior of variable x_1 and the $\bar{x}_2$ column represents the mean behavior of variable x_2. From an in-control process with $m = 50$ samples and the constant sample size of $n = 10$, we are given the following statistics:

$$\bar{\bar{x}} = \begin{bmatrix} 15 \\ 5 \end{bmatrix} \quad S = \begin{bmatrix} 21 & 9 \\ 9 & 15 \end{bmatrix} \tag{4.50}$$

   Use Excel or Python to create a phase II T^2 and GV charts. Assume $\alpha = 0.001$. What can you conclude?

Table 4.9: Falls at Metropolis Skilled Nursing Facility

Day	*Census*	*Falls*	*Risk*	*StDev*
1	83	5	0.48	0.38
2	75	9	0.14	0.11
3	93	6	0.26	0.2
4	76	9	0.23	0.18
5	72	12	0.12	0.1
6	88	4	0.06	0.05
7	99	6	0.24	0.19
8	91	6	0.35	0.28
9	92	7	0.26	0.2
10	85	9	0.45	0.36
11	87	4	0.05	0.04
12	90	20	1	0.8
13	73	4	0.6	0.48
14	83	7	0.86	0.68
15	83	9	0.72	0.58
16	85	4	0.06	0.05
17	77	4	0.2	0.16
18	76	6	0.53	0.42
19	77	3	0.28	0.22
20	77	6	0.45	0.36
21	75	8	0.2	0.16
22	99	10	0.4	0.32
23	93	7	0.14	0.11
24	78	8	0.27	0.22
25	95	10	0.3	0.24
26	94	6	0.28	0.22
27	76	10	0.1	0.08
28	86	15	0.1	0.08
29	95	10	0.8	0.64
30	92	3	0.55	0.44
31	99	8	0.85	0.68
32	79	3	0.85	0.68
33	94	7	0.08	0.06
34	91	9	0.4	0.32
35	98	6	0.21	0.17
36	98	8	0.2	0.16
37	85	9	0.28	0.22
38	73	3	0.18	0.14
39	87	9	0.23	0.18
40	88	8	0.85	0.68
41	82	6	0.26	0.21
42	94	7	0.7	0.56
43	98	5	0.28	0.22
44	80	7	0.23	0.18
45	80	9	0.8	0.64
46	97	6	0.44	0.35
47	73	6	0.8	0.64
48	75	9	0.44	0.35
49	92	3	0.11	0.08
50	100	5	0.06	0.05

Table 4.10: Queue times of IT tickets for CDSS-related questions

t	x_t	t	x_t	t	x_t	t	x_t
1	35.9	*19*	30.7	*37*	34.1	*55*	33.6
2	33	*20*	30.3	*38*	28.9	*56*	31.2
3	33.5	*21*	29.5	*39*	27.6	*57*	31.3
4	29.1	*22*	27.8	*40*	24.5	*58*	27.1
5	26.8	*23*	30.8	*41*	29.2	*59*	31
6	25.7	*24*	35.3	*42*	31.8	*60*	31
7	24.3	*25*	29.1	*43*	29.4	*61*	34.6
8	22.9	*26*	30.2	*44*	29.3	*62*	33.8
9	27.3	*27*	31.3	*45*	31.3	*63*	31.7
10	26.9	*28*	27.3	*46*	29.3	*64*	31.6
11	22.9	*29*	27.9	*47*	26.1	*65*	30.2
12	22.1	*30*	33.7	*48*	29.6	*66*	30.5
13	25.6	*31*	37.5	*49*	30.3	*67*	28.9
14	22.5	*32*	37.8	*50*	31.5	*68*	28.6
15	25.7	*33*	35.8	*51*	31.1	*69*	25.7
16	29.5	*34*	34.8	*52*	34.1	*70*	23.7
17	33.2	*35*	34.1	*53*	35.9		
18	32.5	*36*	34	*54*	32.3		

Table 4.11: Sample statistics of a bivariate medical process at Metropolis Hospital

Sample#	$\bar{x}_1$	$\bar{x}_2$	*Sample#*	$\bar{x}_1$	$\bar{x}_2$
1	20	6	*16*	15	5
2	19	7	*17*	14	8
3	20	5	*18*	15	5
4	16	8	*19*	18	7
5	14	6	*20*	14	5
6	14	6	*21*	20	6
7	20	4	*22*	19	8
8	20	6	*23*	15	6
9	18	6	*24*	13	4
10	17	6	*25*	13	4
11	13	5	*26*	20	7
12	15	3	*27*	13	5
13	14	3	*28*	14	5
14	14	5	*29*	17	7
15	20	6	*30*	17	3

7. Consider individual clinical measurements in Table 4.12, where $m = 45$ and $n = 1$. Additionally, we are given these statistics:

$$\bar{x} = \begin{bmatrix} 82.5 \\ 39.1 \end{bmatrix} \quad S = \begin{bmatrix} 14 & 13 \\ 13 & 25 \end{bmatrix} \tag{4.51}$$

Use Excel or Python to create phase II T^2 and GV charts. Assume $\alpha = 0.001$. What can you conclude?

Table 4.12: Individual clinical measurements

Sample#	x_1	x_2	*Sample#*	x_1	x_2
1	82.2	45.1	*11*	82.6	48.3
2	81.1	48.55	*12*	74.7	38.35
3	89.2	44.6	*13*	73.4	43.7
4	89.9	53.95	*14*	75.7	45.85
5	80.5	41.25	*15*	81.5	46.75
6	78.4	43.2	*16*	84.9	45.45
7	76.8	40.4	*17*	85.5	43.75
8	74.4	38.2	*18*	74.8	41.4
9	79.7	43.85	*19*	89.1	49.55
10	85	45.5	*20*	71.8	43.9

CHAPTER 5

Tools related to control charts

In this last chapter, we examine quality improvement tools that are related to control charts. Among the topics we discuss are capability analysis and run charts. We also consider benchmarking methods such as funnel charts and analysis of means (ANOM). We demonstrate how to create and implement these concepts using Excel, Python, and Minitab software.

Key concepts and tools: *Capability analysis; Run charts; Benchmarking quality; Analysis of means (ANOM); Analysis of variance (ANOVA); League tables; Funnel charts*

Major objectives
After studying this chapter, you will be able to:

1. *Define key concepts and tools related to control charts*
2. *Recognize the need for capability analysis*
3. *Differentiate between ANOVA and ANOM techniques*
4. *Compare and contrast run charts to control charts*
5. *Implement capability analysis ratios using Excel and Minitab*
6. *Explain the notion of benchmarking quality*
7. *Create funnel charts using Excel and Python*
8. *Conduct ANOM using Minitab*
9. *Develop run charts using Excel, Python, and Minitab*
10. *Interpret the significance of the results from the tools discussed in this chapter*

5.1 Introduction

In this last chapter, we review a few of the methods related to control charts. We begin with a discussion about capability analysis to help answer the question of whether our process is capable of meeting the customer's specification limits. Next, we consider benchmarking techniques that we can employ to compare quality outcomes among several health care providers. One of these techniques that we discuss is funnel charts that we create using a quality target and control limits. Another benchmarking technique that we consider is ANOM. We can use this tool to compare mean statistics in a process using a centerline and decision limits. The last topic we consider relates to run charts. We create such charts using only a centerline and then attempt to deduce out-of-control behaviors using run statistics and visual inspections of non-random behaviors. We typically implement the methods that we discuss here during the *Analyze* and *Improve* phases of DMAIC[1].

5.2 Process capability analysis

During quality improvement efforts, we conduct **capability analysis** to help answer the question of whether our process is capable of meeting the customer's specification limits or standards. For example, in health care, we want to know whether our process can meet various guidelines from regulating and accrediting agencies. An important pre-requisite to capability analysis is the **stability** of the process.

Definition 5.1 **Capability analysis** is a technique that involves measuring the uniformity of the process, given the specification limits and variability in the process [44].

Several techniques for conducting capability analysis exist. Here, we emphasize the use of graphical techniques such as histograms and process capability ratios (PCRs) like C_p.

5.2.1 The C_p ratio

Given the customer's upper specification limit (USL) and lower specification limit (LSL), we utilize the C_p ratio to measure the **potential** capability of a process. We formulate this ratio as follows:

$$C_p = \frac{USL - LSL}{6\sigma} \tag{5.1}$$

where σ represents the process variability in standard deviation units. We obtain 6σ from the sum of 3σ below the mean μ and 3σ above the mean. We recall that 99.73% of the normally distributed process falls within $\mu \pm 3\sigma$. Since μ is not part of the formula in

[1] **DMAIC**: Define, Measure, Analyze, Improve, and Control

Equation 5.1, we cannot conclude anything about the process mean using the C_p ratio. To make such a conclusion, we apply other PCRs such as C_{pu}, C_{pl}, and C_{pk}.

5.2.2 The C_{pu}, C_{pl}, and C_{pk} ratios

Like the C_p ratio, the C_{pu}, C_{pl}, and C_{pk} ratios also allow us to measure the capability of the process. But unlike the C_p ratio, these new ratios account for the process mean μ.

1. The C_{pu} ratio, formulated as

$$C_{pu} = \frac{USL - \mu}{3\sigma}, \tag{5.2}$$

   measures the process capability of satisfying the USL given the mean μ and 3σ.

2. The C_{pl} ratio, formulated as

$$C_{pl} = \frac{\mu - LSL}{3\sigma}, \tag{5.3}$$

   measures the process capability of meeting the LSL given the mean μ and 3σ.

3. The C_{pk} ratio, computed as

$$C_{pk} = \min\left(C_{pu}, C_{pl}\right), \tag{5.4}$$

   measures the **actual** capability of the process. Here, $\min()$ symbolizes the minimum function.

When $C_{pk} = C_p$, we deduce that the process mean μ is about centered between USL and LCL. When $C_{pk} < C_p$, the process is off-center. For cases of $C_{pk} < 0$, the process is completely outside of the specification limits.

5.2.3 Implementing PCRs

Before we can implement PCRs, we must be given the process standard deviation σ or be able to estimate this statistic from the process sample. We also may need to know the process mean μ. We can estimate both of these parameters, as previously discussed in Chapter 1.

Interpretation

In general, any **PCR greater than 1** implies that the process is capable of meeting the given standard. Otherwise, to improve the capability measure of interest, we must decrease the variability in the process given by σ. For an existing process, it is recommended that a PCR measure be at least 1.33 in a two-sided measure or 1.25 in a one-sided measure. For a new process, the recommended PCR values are 1.50 for a two-sided measure and 1.45 for a one-sided measure. In a two-sided, centered, and normally distributed process, a C_p of 0.25 implies 453,255 ppm[2], a C_p of 1 implies 2,700 ppm, and a C_p of 2 signifies about 0.0018 ppm [44]. Another practical interpretation of the C_p ratio follows from:

$$P = \left(\frac{1}{C_p}\right) \tag{5.5}$$

where P is the percentage of the specification bandwidth being used by the process.

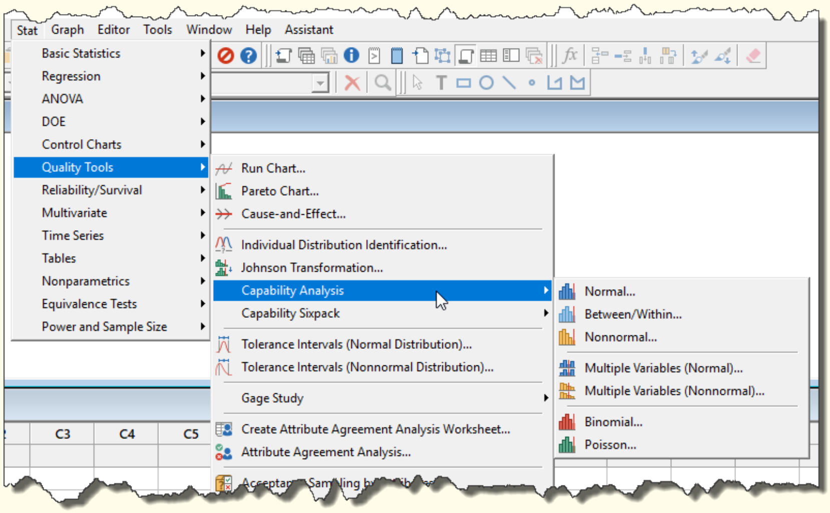
1. If $P \leq 1$: the process is capable of meeting the specification limits.
2. If $P > 1$: the process is using more than the allowed bandwidth and is therefore not capable of meeting the specification limits.

Besides measuring the process capability, the C_p ratio can also be utilized to evaluate the acceptability of the $gauge$ instrument [44].

Remark: *When a process is* ***not*** *in control, performance indices like P_p, P_{pu}, P_{pl}, and P_{pk} are sometimes used to measure the capability. These indices are essentially similar to the PCRs discussed earlier except that σ is replaced with the process standard deviation s. These ratios should be applied with caution, or not be used at all, due to possible misleading conclusions that may result when the process is not stable [44].*

How-To 5.1 (PCRs in Minitab 18) Click on *Stat* > *Quality Tools* > *Capability Analysis* > *Normal* > *upload your data*. See the snapshot in Figure 5.2. You can change the distribution as necessary.

Figure 5.1: Capability analysis options in Minitab 18

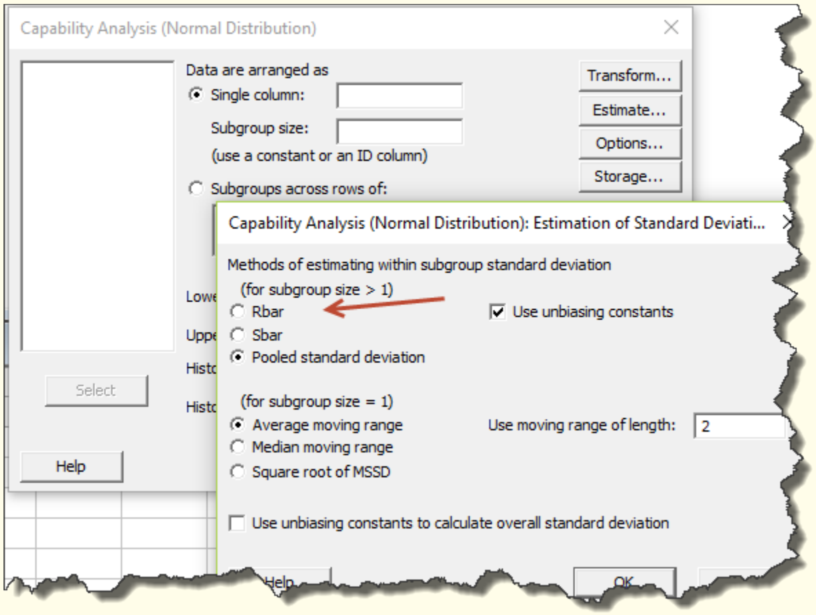


For additional graphical results, click on *Stat* > *Quality Tools* > *Capability Sixpack*

[2] **ppm:** defective parts per million

> *select the appropriate distribution* > *upload your data* > *OK*. To change the σ estimation method, click on *Estimate* > Select the preferred technique (see the snapshot in Figure 5.2)

Figure 5.2: σ estimation options in Minitab 18



Capability Analysis (Normal Distribution)

Data are arranged as

● Single column:

Subgroup size:

(use a constant or an ID column)

○ Subgroups across rows of:

Capability Analysis (Normal Distribution): Estimation of Standard Deviati...

Methods of estimating within subgroup standard deviation

(for subgroup size > 1)

○ Rbar

○ Sbar

● Pooled standard deviation

☑ Use unbiasing constants

(for subgroup size = 1)

● Average moving range

○ Median moving range

○ Square root of MSSD

Use moving range of length: 2

□ Use unbiasing constants to calculate overall standard deviation

Table 5.1: Processing times of inpatient records at Metro City Hospital in 2018

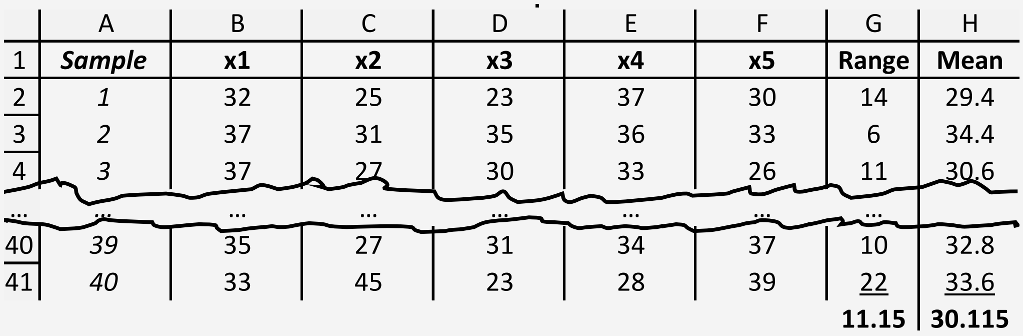
Sample	x_1	x_2	x_3	x_4	x_5	*Sample*	x_1	x_2	x_3	x_4	x_5
1	32	25	23	37	30	*21*	27	23	33	24	32
2	37	31	35	36	33	*22*	31	33	37	30	32
3	37	27	30	33	26	*23*	30	36	31	26	25
4	36	32	27	21	31	*24*	36	28	30	26	30
5	34	40	40	26	30	*25*	23	22	28	17	32
6	32	33	38	30	29	*26*	30	26	37	32	23
7	20	35	34	28	22	*27*	32	30	31	28	34
8	35	21	35	33	25	*28*	29	36	30	25	27
9	36	29	41	27	26	*29*	27	32	27	32	30
10	33	27	35	31	20	*30*	29	24	23	26	29
11	30	29	33	24	31	*31*	28	30	25	27	31
12	31	27	27	32	35	*32*	31	31	28	30	31
13	26	24	38	32	25	*33*	24	26	37	30	35
14	37	26	21	41	31	*34*	26	31	30	25	28
15	33	24	31	33	29	*35*	28	26	25	38	26
16	26	30	35	32	22	*36*	30	29	30	30	28
17	29	37	34	34	24	*37*	32	33	28	25	35
18	33	34	25	33	31	*38*	21	38	32	30	30
19	26	30	29	26	41	*39*	35	27	31	34	37
20	33	24	33	32	35	*40*	33	45	23	28	39

Example 5.1 (Capability analysis)

For the past 40 days, the medical coding manager at Metro City Hospital has been sampling five inpatient records daily $(x_1,\ldots,x_5)$ to track the processing times in minutes (mins). The manager would like to test the process capability to meet the rate of USL = 45 minutes and LSL = 15 minutes. The process is statistically stable. Table 5.1 shows the data collected so far.

To help the manager analyze the capability of this process, we start by estimating parameters σ and μ. We approximate μ using $\bar{\bar{x}}$, which is the average of all sample averages. To approximate σ, we utilize Equation 1.20 since $n \le 10$. Using Excel, we obtain the range by $MAX(sample) - MIN(sample)$. We calculate the average of each sample using the $AVERAGE(sample)$ function. We illustrate the setup of our spreadsheet in Figure 5.3.

Figure 5.3: A spreadsheet setup for calculating the ranges and averages of the data in Table 5.1



	A	B	C	D	E	F	G	H
1	***Sample***	**x1**	**x2**	**x3**	**x4**	**x5**	**Range**	**Mean**
2	*1*	32	25	23	37	30	14	29.4
3	*2*	37	31	35	36	33	6	34.4
4	*3*	37	27	30	33	26	11	30.6
...	...	...	...	...	...	...	...	
40	*39*	35	27	31	34	37	10	32.8
41	*40*	33	45	23	28	39	22	33.6
							11.15	**30.115**

About Figure 5.3, we obtained the range of sample 1 by $= MAX(B2:F2) - MIN(B2:F2) = 14$. We obtained the average of sample 1 by $= AVERAGE(B2:F2) = 29.4$. We applied the same formulas to other samples. In the end, we calculated $\bar{R}$ using $= AVERAGE(G2:G41) = 11.15$ and $\bar{\bar{x}}$ using $= AVERAGE(H2:H41) = 30.115$. To approximate σ, we proceeded as follows:

$$\sigma \approx \hat{\sigma} = \frac{\bar{R}}{d_2} = \frac{11.15}{2.326} = 4.80 \tag{5.6}$$

We obtained the value of $d_2 = 2.326$ from Appendix Table 12 when $n = 5$. We proceeded to calculate the PCRs of interest as follows:

$$C_p = \frac{USL - LSL}{6\sigma} = \frac{45-15}{6(4.80)} = 1.043 \tag{5.7}$$

$$C_{pu} = \frac{USL - \mu}{3\sigma} = \frac{45-30.115}{3(4.80)} = 1.035 \tag{5.8}$$

$$C_{pl} = \frac{\mu - LSL}{3\sigma} = \frac{30.115-15}{3(4.80)} = 1.051 \tag{5.9}$$

$$C_{pk} = \min\left(C_{pu}, C_{pl}\right) = \min(1.035, 1.051) = 1.035 \tag{5.10}$$

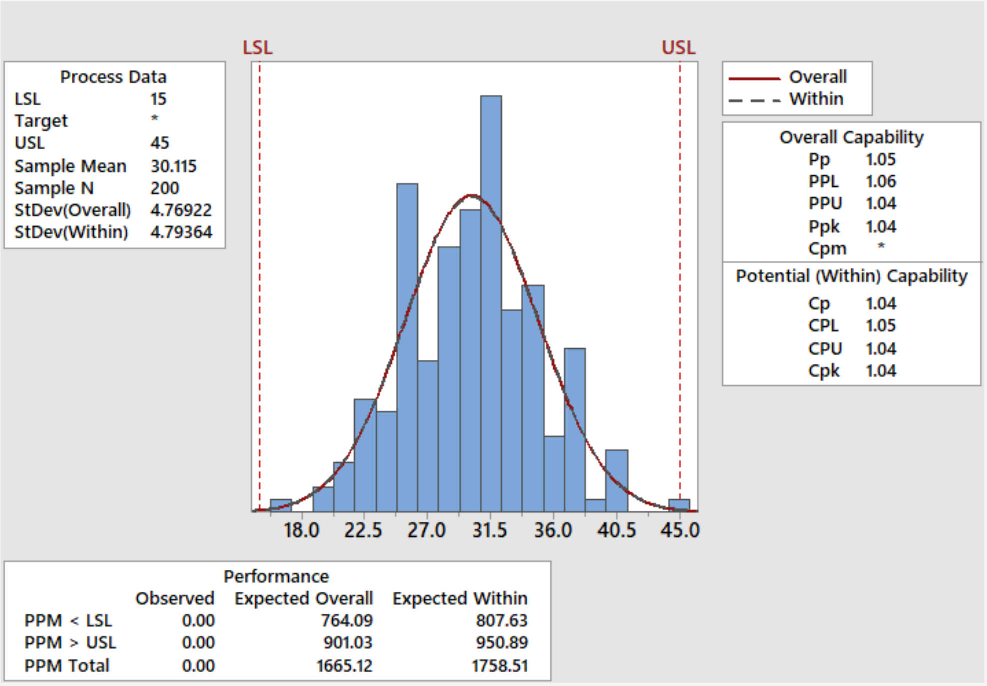
Given that all ratios are greater than 1, the process is capable. Furthermore, we can deduce that the process mean is about centered between USL and LSL, since $C_p \approx C_{pk}$. We estimate the amount of the bandwidth that the process is utilizing this way:

$$P = \left(\frac{1}{C_p}\right)100 = \left(\frac{1}{1.043}\right)100 = 96\%. \tag{5.11}$$

The 96% utilization rate of the allowed bandwidth is not desirable since it leaves little to no room for error. To improve the capability, the process variability must be decreased. Figure 5.4 portrays a Minitab capability report. The **histogram** in this figure confirms that the process is about centered, but is using almost all of the allowed bandwidth between LSL and USL. The **Capability Plot** substantiates this claim. The intervals in this plot portray different estimations of the spread in the process. The **within** interval follows from the unbiased estimation of σ using sample data.

Figure 5.5 depicts additional Minitab results related to the sixpack report. The **Xbar and R** charts in Figure 5.5 indicate that the process is stable since points move randomly around the centerline, and no point falls outside of the control limits. The **Last 25 Subgroups** chart validates the conclusion about the random behavior of the sample mean since no clear pattern is discernible. But, we do notice that sample 40 seems to show more variability than other points. The **Normal Probability Plot** portrays points falling alongside the fitted line, an indication that the process is about normally distributed.

Figure 5.4: Process capability report in Minitab 18 based on the data in Table 5.1



Process Data	
LSL	15
Target	*
USL	45
Sample Mean	30.115
Sample N	200
StDev(Overall)	4.76922
StDev(Within)	4.79364

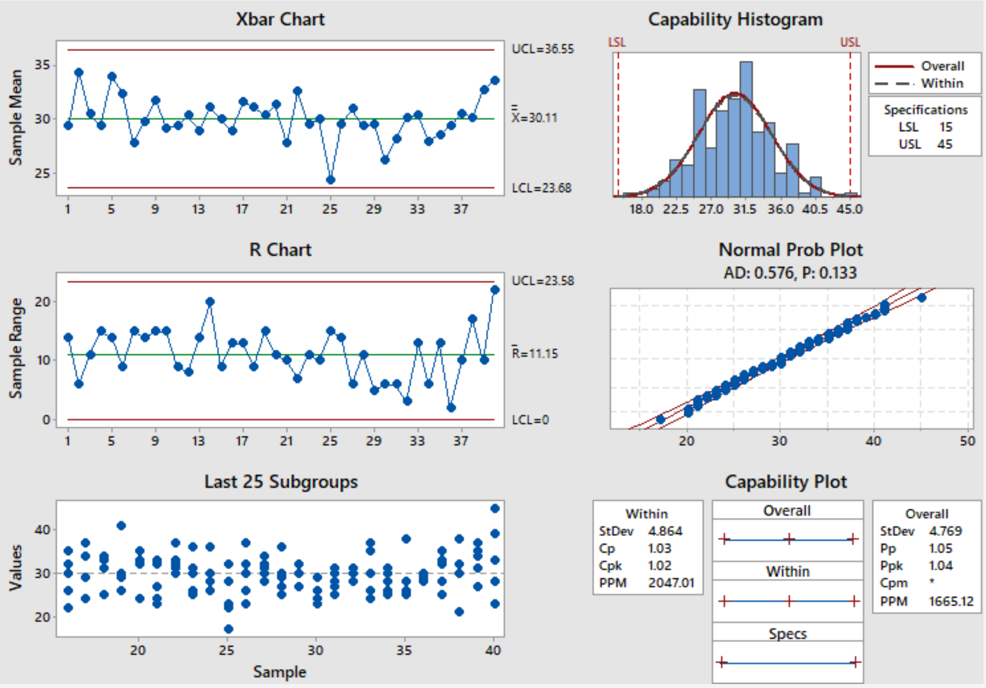
Overall Capability	
Pp	1.05
PPL	1.06
PPU	1.04
Ppk	1.04
Cpm	*

Potential (Within) Capability	
Cp	1.04
CPL	1.05
CPU	1.04
Cpk	1.04

Performance

	Observed	Expected Overall	Expected Within
PPM < LSL	0.00	764.09	807.63
PPM > USL	0.00	901.03	950.89
PPM Total	0.00	1665.12	1758.51

Figure 5.5: Process capability sixpack report in Minitab 18 based on the data in Table 5.1



5.3 Benchmarking quality

In this section, we discuss how to benchmark quality from multiple health care organizations using tools that utilize the concept of control charts. Specifically, we consider **funnel charts** and **analysis of means (ANOM)**. We briefly also review the concepts of the analysis of variance (ANOVA) to demonstrate the difference between this test and the ANOM tool.

5.3.1 Funnel charts

Like control charts, funnel charts are constructed using control limits and a centerline. A typical funnel chart tends to look like a p chart and is structured as follows [53]:

$$UCL = p + L\sqrt{\frac{p(1-p)}{n_i}} \tag{5.12}$$

$$CL = p \tag{5.13}$$

$$LCL = p - L\sqrt{\frac{p(1-p)}{n_i}} \tag{5.14}$$

Here, p is the target, L is the number of standard deviations away from the centerline, commonly chosen to create a 95% or 99.8% confidence interval [65]. The notation of n_i signifies the sample size of organization i, for $i : 1, 2, \ldots, k$, where k is the total number of organizations being compared. The acronyms *UCL, CL,* and *LCL*, stand for *upper control limit*, *centerline*, and *lower control limit* respectively. If a point falls outside of a particular limit, we conclude that the deviation of the corresponding sample number, away from the target or the grand mean, is statistically significant. The only difference between a funnel chart and a p chart is that in the former, we have to sort samples in ascending order, a practice that we do not adhere to in the latter. Review Chapter 2 about how to create p charts.

5.3.2 ANOM

ANOM, not to be confused with ANOVA (see Subsection 5.3.3), is a statistical technique that we can employ to benchmark quality from several health care organizations. The main difference between ANOVA and ANOM is that the former compares the equality of means, whereas the latter compares each mean to the overall mean. Just like in funnel charts, the formulas for ANOM charts often look like those of a p-chart [53]. In Minitab 18, the formulas for the binomial ANOM chart look like this [1]:

$$UDL = p + L\sqrt{\frac{p(1-p)}{n}} * \sqrt{\frac{k-1}{k}} \quad (5.15)$$

$$LDL = p - L\sqrt{\frac{p(1-p)}{n}} * \sqrt{\frac{k-1}{k}} \quad (5.16)$$

Here, UDL stands for *upper decision limit* and LDL means *lower decision limit*. The notation of n symbolizes a constant sample size, and k represents the number of organizations in the study. As before, L represents the number of standard deviations away from the centerline p. Minitab 18 determines L as follows:

$$L = \Phi^{-1}(1 - \alpha/(2 * k)) \quad (5.17)$$

where $*$ symbol characterizes multiplication and $\Phi(.)$ is the cumulative density function of a standard normal. As before, α is the given significance level. As in funnel charts, if a point falls outside of a particular limit, we conclude that the corresponding sample is statistically different from the target or the grand mean.

How-To 5.2 (Python 3.6)

Script 5.1: A script for creating funnel charts using Python 3.6

```
#FUNNEL CHARTS
#import modules
from pandas import*
from pylab import*
from numpy import*
import seaborn as sns
```

```
#import data from Excel where columns are named: ID, Discharges, and
   Readmits. Discharges are considered the sample size.
data = read_excel(your directory)
#Initialize parameters
data['xt'] = data.Readmits/data.Discharges
data = data.sort_values(by='Discharges')
data = data.reset_index()
nbar = mean(data.Discharges)
xr = data.xt
CL = [pbar]*(len(xr))
t = arange(len(xr) )
#L = 3
pbar = 1.*data.Readmits.sum()/ data.Discharges.sum()
UCL2 = [pbar + 3.*sqrt(pbar*(1.-pbar)/data.Discharges[i])  for i in
   range(len(xr))]
LCL2 =  [pbar - 3.*sqrt(pbar*(1.-pbar)/data.Discharges[i])  for i in
   range(len(xr))]
#L =2
UCL1 = [pbar + 2.*sqrt(pbar*(1.-pbar)/data.Discharges[i])  for i in
   range(len(xr))]
LCL1 =  [pbar - 2.*sqrt(pbar*(1.-pbar)/data.Discharges[i])  for i in
   range(len(xr))]
#mark red the point that falls outside of the outer control limits.
   Otherwise, mark the point blue.
#mark black the point that falls outside of the inner control limits.
    Otherwise, mark the point blue.
markers = []
colors = []
for i in range (len(xr)):
    x1 = xr[i]
    if (x1 > UCL2[i] or  x1 < LCL2[i]):
        markers.append('o')
        colors.append('r')
    else:
        markers.append('o')
        colors.append('b')
colors = array(colors)
for i in range (len(xr)):
    x1 = xr[i]
    if (x1 > UCL1[i] and x1 < UCL2[i]):
        colors[i]='k'
    if (x1 < LCL1[i] and x1 > LCL2[i]):
        colors[i]='k'
#Plotting the funnel chart
fig=figure()
ax1 =  fig.add_subplot(111)
ax1.plot(t, UCL2, 'k-', alpha = 0.5)
ax1.plot(t, LCL2, 'k-',alpha = 0.5)
ax1.plot(t, UCL1, 'k-', alpha = 0.5)
ax1.plot(t, LCL1, 'k-',alpha = 0.5)
ax1.plot(CL, 'k-',alpha = 0.5)
ax1.plot(xr,'b-',zorder=1)
for x,y,c,m in zip(t, xr, colors, markers):
    ax1.scatter(x,y,c=c, marker=m, s =45, alpha = 1.,zorder=2)
xlim(-0.5, t[-1]+1)
sns.color_palette("Blues")
sns.despine(offset=10, trim=False)
#label y-axis and x-axis
ax1.set_xlabel('Hospital ID sorted by Sample Size')
ax1.set_ylabel('Fraction nonconforming')
#annotate the values of UCL, LCL, and CL
ax1.annotate ('$UCL2=$'+str(round(UCL2[0],3)), xy = (xlim()[1], list(
    UCL2)[-1]), xytext = (xlim()[1],list(UCL2)[-1]),fontsize = 11)
ax1.annotate ('$\overline{P}=$'+str(round(CL[0],3)), xy = (xlim()[1],
     list(CL)[-1]), xytext = (xlim()[1],list(CL)[-1]),fontsize = 11)
ax1.annotate ('$LCL2=$'+str(round(LCL2[0],3)), xy = (xlim()[1], list(
    LCL2)[-1]), xytext = (xlim()[1],list(LCL2)[-1]),fontsize = 11)
ax1.annotate ('$UCL1=$'+str(round(UCL1[0],3)), xy = (xlim()[1], list(
    UCL1)[-1]), xytext = (xlim()[1],list(UCL1)[-1]),fontsize = 11)
ax1.annotate ('$LCL1=$'+str(round(LCL1[0],3)), xy = (xlim()[1], list(
    LCL1)[-1]), xytext = (xlim()[1],list(LCL1)[-1]),fontsize = 11)
#set xticks to reflect hospital IDs sorted by sample size
xticks(arange(len(data), step = 1), data.ID)
show()
```

How-To 5.3 (Funnel charts in Excel 2013)
Excel does not have built-in options for creating either ANOM or funnel charts, but we can manually program the respective formulas as we demonstrate how to create funnel charts in Example 5.2.

Example 5.2 (A funnel chart in Excel 2013 and Python 3.6)
The league table given in Table 5.2 depicts the ranking of 20 hospitals in Central City, regarding the rates of pneumonia patients readmitted within 30 days of discharge. The lower the rate, the better. The average readmission rate is 13.7%. Hospitals that are above this average are marked in red. Table 5.3 demonstrates the setup of an Excel spreadsheet for creating a funnel chart based on the data in Table 5.2. In this setup, ID denotes the *hospital ID* column, n_i symbolizes the column of *Pneumonia Discharges* or sample size, x_i represents the column of *Readmit with 30 days*, and x_i/n_i represents to the $Rate$ column. We calculated the average rate using $p = \sum_{i=1}^{k} x_i/n_i$, where k is the total number of hospitals. Before plotting the funnel chart, we sorted Table 5.3 by the sample size n_i. We applied Equations 5.12 - 5.14 to create inner control limits (LCL_1 and UCL_1 using $L = 2$) and outer control limits (LCL_2 and UCL_2 using $L = 3$). For example, we obtained the value of LCL_1 in cell $E2$ using $= I2 - 2 * SQRT(I2 * (1 - I2)/B2)$ and the value of UCL_2 in cell $H2$ using $= I2 + 3 * SQRT(I2 * (1 - I2)/B2)$. From Table 5.3, we can create a funnel chart by inserting line charts of columns *D* to *I*. The resulting chart would look like Figure 5.6 that we plotted using the Python script in How-To 5.2. From this chart, we see that hospitals with IDs 3, 8, and 9 fail $UCL2$. Also, we see that hospitals with IDs 7, 13, 17, and 18 fail $LCL1$.

Figure 5.6: A funnel chart based on the data in Table 5.2

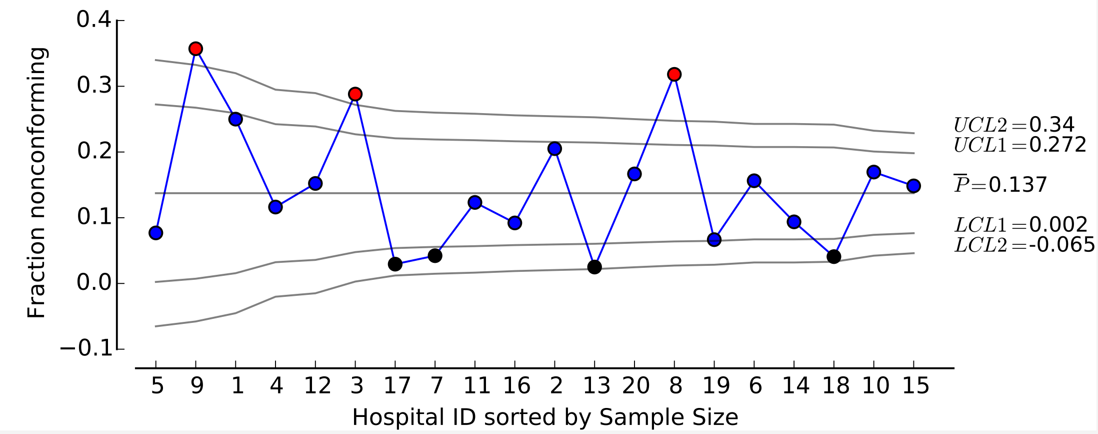


Table 5.2: Pneumonia patients readmitted within 30 days of discharge in Central City, 2018

Hospital ID	Pneumonia Discharges	Readmit with 30 days	Rate	Rank
13	80	2	2.5%	1
17	68	2	2.9%	2
18	98	4	4.1%	3
7	71	3	4.2%	4
19	90	6	6.7%	5
5	26	2	7.7%	6
16	76	7	9.2%	7
14	96	9	9.4%	8
4	43	5	11.6%	9
11	73	9	12.3%	10
15	128	19	14.8%	11
12	46	7	15.2%	12
6	96	15	15.6%	13
20	84	14	16.7%	14
10	118	20	16.9%	15
2	78	16	20.5%	16
1	32	8	25.0%	18
3	59	17	28.8%	17
8	88	28	31.8%	19
9	28	10	35.7%	20

Table 5.3: A setup of an Excel spreadsheet for creating a funnel chart based on the data in Table 5.2

	A	B	C	D	E	F	G	H	I
1	ID	n_i	x_i	x_i/n_i	LCL_1	UCL_1	LCL_2	UCL_2	p
2	5	26	2	7.7%	0.2%	27.2%	-6.5%	34.0%	13.7%
3	9	28	10	35.7%	0.7%	26.7%	-5.8%	33.2%	13.7%
4	1	32	8	25.0%	1.6%	25.9%	-4.5%	32.0%	13.7%
5	4	43	5	11.6%	3.2%	24.2%	-2.0%	29.5%	13.7%
6	12	46	7	15.2%	3.6%	23.9%	-1.5%	29.0%	13.7%
7	3	59	17	28.8%	4.8%	22.7%	0.3%	27.2%	13.7%
8	17	68	2	2.9%	5.4%	22.1%	1.2%	26.3%	13.7%
9	7	71	3	4.2%	5.6%	21.9%	1.5%	26.0%	13.7%
10	11	73	9	12.3%	5.7%	21.8%	1.6%	25.8%	13.7%
11	16	76	7	9.2%	5.8%	21.6%	1.9%	25.6%	13.7%
12	2	78	16	20.5%	5.9%	21.5%	2.0%	25.4%	13.7%
13	13	80	2	2.5%	6.0%	21.4%	2.2%	25.3%	13.7%
14	20	84	14	16.7%	6.2%	21.2%	2.5%	25.0%	13.7%
15	8	88	28	31.8%	6.4%	21.1%	2.7%	24.7%	13.7%
16	19	90	6	6.7%	6.5%	21.0%	2.8%	24.6%	13.7%
17	14	96	9	9.4%	6.7%	20.8%	3.2%	24.3%	13.7%
18	6	96	15	15.6%	6.7%	20.8%	3.2%	24.3%	13.7%
19	18	98	4	4.1%	6.8%	20.7%	3.3%	24.2%	13.7%
20	10	118	20	16.9%	7.4%	20.1%	4.2%	23.2%	13.7%
21	15	128	19	14.8%	7.6%	19.8%	4.6%	22.9%	13.7%

In contrast to the hospital ranking in Table 5.2 where ten hospitals may be considered deficient since they are above the average, the funnel chart indicates that only a few hospitals fail the control limits at 99.73% confidence interval. In typical applications, the negative part of the funnel limits is set to zero; we plotted it here only to demonstrate the funnel shape from which the name *funnel chart* is derived.

5.3.3 ANOVA

ANOVA is a powerful statistical test that we use to examine the equality of the means in more than two independent samples. We assume that the samples are normally distributed with equal variances. The two types of ANOVA test that we consider here are **one-way ANOVA** and **two-way ANOVA**. In both cases, we work with the sum of squares

(SS) to formulate the statistical tests of interest.

One-way ANOVA

In a one-way ANOVA test, we obtain the total sum of squares (SS_{total}) in the data as follows:

$$SS_{total} = SS_{treatments} + SS_{error} \tag{5.18}$$

The word $treatments$ is used to denote independent samples. The $SS_{treatments}$ term represents the **between** samples variation, whereas the SS_{error} term symbolizes the **within** samples variation. Let's assume that we have a number of samples, each with size n. We determine the corresponding sum of squares as follows:

$$SS_{total} = \sum_{i=1}^{a}\sum_{j=1}^{n}(y_{ij} - \bar{y}_{..})^2 \tag{5.19}$$

$$SS_{treatments} = n\sum_{i=1}^{a}(\bar{y}_{i.} - \bar{y}_{..})^2 \tag{5.20}$$

$$SS_{error} = \sum_{i=1}^{a}\sum_{j=1}^{n}(y_{ij} - \bar{y}_{j.})^2 \tag{5.21}$$

where y_{ij} is the entry in row i and column j, $\bar{y}_{..}$ is the overall mean of the data, $\bar{y}_{i.}$ is the mean of row i, and $\bar{y}_{j.}$ is the mean of column j. The degrees of freedom (df) are given by:

$$df_{total} = an - 1 \tag{5.22}$$

$$df_{treatments} = a - 1 \tag{5.23}$$

$$df_{error} = a(n-1) \tag{5.24}$$

It follows that:

$$df_{total} = df_{treatments} + df_{error} \tag{5.25}$$

Using the sum of squares and the degrees of freedom, we obtain the mean squares (MS) this way:

$$MS_{treatements} = \frac{SS_{treatments}}{df_{treatments}} \tag{5.26}$$

$$MS_{error} = \frac{SS_{error}}{df_{error}} \tag{5.27}$$

The $MS_{treatements}$ term estimates the variance in treatments. The MS_{error} term is the unbiased estimator of the variance of the model residuals in Equation 5.18 [44]. We use the ratio between $MS_{treatements}$ and MS_{error} to obtain the following **F-test statistic**:

$$F_0 = \frac{MS_{treatements}}{MS_{error}} \tag{5.28}$$

A one-tailed **critical value** is given by:

$$F_{\alpha,\, df_{treatments},\, df_{error}} \equiv F_{\alpha,\, a-1,\, a(n-1)} \tag{5.29}$$

At the significance level α, we reject the null hypothesis when $F_0 > F_{\alpha,\, a-1,\, a(n-1)}$. Our conclusion implies that at least one mean is different. The ANOVA test does not identify the different mean in question.

Two-way ANOVA

In a two-way ANOVA test, we add a blocking variable $SS_{blocking}$ and SS_{total} becomes:

$$SS_{total} = SS_{treatments} + SS_{blocking} + SS_{error} \tag{5.30}$$

The $SS_{blocking}$ term represents different blocks or conditions under which the sample data were collected. If data were arranged in a table format, we could view **treatments** as independent rows and **blocks** as independent columns. Given a number of treatments and b number of blocks, we formulate the sum of squares of a two-way ANOVA this way:

$$\begin{aligned}
SS_{total} &= \sum_{i=1}^{a}\sum_{j=1}^{b}(y_{ij} - \bar{y}_{..})^2 && (5.31)\\
SS_{treatments} &= b\sum_{i=1}^{a}(\bar{y}_{i.} - \bar{y}_{..})^2 && (5.32)\\
SS_{blocking} &= a\sum_{i=1}^{b}(\bar{y}_{.j} - \bar{y}_{..})^2 && (5.33)\\
SS_{error} &= \sum_{i=1}^{a}\sum_{j=1}^{b}(y_{ij} - \bar{y}_{i.} - \bar{y}_{.j} + \bar{y}_{..})^2 && (5.34)\\
&\equiv SS_{total} - SS_{treatments} - SS_{blocking} && (5.35)
\end{aligned}$$

As earlier, we can view y_{ij} as the entry in row i and column j, $\bar{y}_{..}$ is the overall mean of the data, $\bar{y}_{i.}$ is the mean of row i, and $\bar{y}_{j.}$ is the mean of column or block j. To obtain the degrees of freedom (df), we proceed as follows:

$$\begin{aligned}
df_{total} &= ab - 1 && (5.36)\\
df_{treatments} &= (a-1) && (5.37)\\
df_{blocking} &= (b-1) && (5.38)\\
df_{error} &= (a-1)(b-1) && (5.39)
\end{aligned}$$

where

$$df_{total} = df_{treatments} + df_{blocking} + df_{error} \tag{5.40}$$

We summarize the two-way mean squares and the corresponding F_0 statistics in Table 5.4.

Table 5.4: Mean Squares and the F_0 statistics of a two-way ANOVA test

Source of variation	**Mean Square**	F_0
Treatments	$MS_{treatements} = \frac{SS_{treatments}}{df_{treatments}}$	$\frac{MS_{treatments}}{MS_{error}}$
Blocks	$MS_{blocking} = \frac{SS_{blocking}}{df_{blocking}}$	$\frac{MS_{blocking}}{MS_{error}}$
Error	$MS_{error} = \frac{SS_{error}}{df_{error}}$	

Just like in a one-way ANOVA test, we draw conclusions about our hypothesis by comparing the F_0 statistic to the *critical value*. We could also use p-values to interpret the significance of the test.

In some applications such as in designed experiments and gauge analysis, we can further decompose SS_{total} to include the term representing the **interaction** between *treatments* and the *blocking* variables. With this new component, SS_{total} becomes:

$$SS_{total} = SS_{treatments} + SS_{blocking} + SS_{interaction} + SS_{error} \tag{5.41}$$

From the model in Equation 5.41, the degrees of freedom of the interaction term are given by $(a-1)(b-1)$. We obtain the test statistic by taking the ratio between $MS_{interaction}$ and MS_{error}. The conclusion about the null hypothesis would be as before.

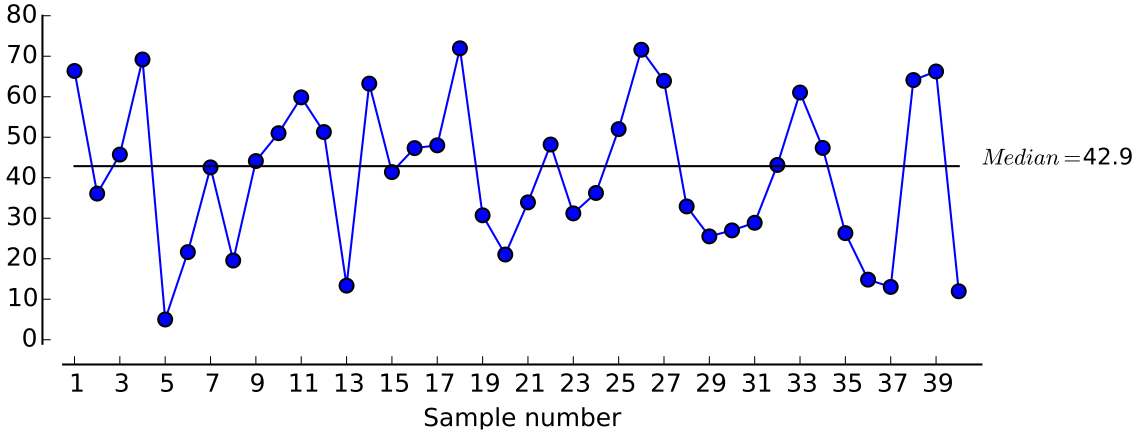
5.4 Run charts

5.4.1 A general concept

Run charts are like control charts but without limits. Moreover, the centerline in the run charts is typically the median of the data, not the mean. While, whenever possible, we should utilize control charts to monitor processes, using run charts has its advantages such as the simpler setup. Also, in run charts, we do not have to assume any particular probability distribution, meaning that we could use these tools to monitor any process. The major downside of run charts is that the interpretation requires a bit more effort since there is no easy way to tell that a particular sample is out-of-control. Figure 5.7 portrays an example of a run chart that relates to the control chart in Figure 1.18.

From the control chart in Figure 1.18 , we have previously concluded that there were two areas of special cause variation due to a point falling outside of LCL and a violation of one of the sensitizing rules since 2 out of 3 consecutive points were between the second and third standard deviation limits on the same side. Now the question at hand is that since there are no limits in the run chart in Figure 5.7, how do we go about detecting out-of-control behaviors?

Figure 5.7: An example of a run chart based on the chart in Figure 1.17



5.4.2 Interpreting run charts

We detect out-of-control behaviors in run charts by finding instances of **non-randomness** in the process. We utilize two techniques to accomplish this task: **visual examination** and statistical **run tests**.

Visual examination

We can use a visual examination to detect special cause variation in run charts by applying the following rules [50]:

1. Seven or more points on the same side of the median signify a likely **shift** in the process.

2. Seven or more points that are either increasing or decreasing indicate a **trend** in the process.

We can also visually detect other likely special causes, such as a **cluster** of points and **zig-zag** or **oscillation** behaviors. To test the significance of these out-of-control behaviors, we apply statistical run tests.

Run tests

We conduct statistical run tests to determine the significance of non-randomness in the process. Our null hypothesis is that the sequence of the process measurements is random. Run tests work for both the **metric** and **non-metric** data. For our purposes, a sequence of groups of data points that are above (**run-ups** +) or below (**run-downs** -) the centerline constitute non-metric data. In contrast, a sequence of successive numerical differences in the process constitutes metric data. We define the term **run** as follows:

Definition 5.2 (Run) A run is a sequence of identical signs (+ or −) until a different sign occurs or the series terminates [63].

To construct a run test, we assume that there are R total number of runs in the data, given by:

$$R = y_1 + y_2 \tag{5.42}$$

where y_1 is the number of run-ups and y_2 is the number of run-downs. We use n_1 to symbolize the total count of points in the set y_1 and n_2 to denote the total count of points in the set y_2. We calculate $p(y_1, y_2)$, the probability of obtaining exactly y_1 runs and y_2 runs this way [65]:

$$p(y_1, y_2) = \frac{\binom{n_1-1}{y_1-1}\binom{n_2-1}{y_2-1}}{\binom{n_1+n_2}{n_1}} \tag{5.43}$$

We obtain the probability of R being less or equal to some value k, that is $p(R \leq k)$ by adding together the probabilities of different combinations of y_1 and y_2 that produce $y_1 + y_2 \leq k$. We use $p(R \leq k)$ to establish the confidence interval for R. If R falls outside of our interval, we reject the null hypothesis, which implies that out-of-control behaviors exist. For small sample sizes of n_1 and n_2, we can establish the confidence interval of R by consulting reference tables of run statistics that are often found in standard statistical textbooks (e.g., see Wackerly et al. (2007)[63]). As n_1 and n_2 get larger (e.g., when both n_1 and n_2 are greater than 10), we assume a normal distribution and establish the confidence interval as follows [63]:

$$E(R) - |Z|\sqrt{V(R)} \leq R \leq E(R) + |Z|\sqrt{V(R)} \tag{5.44}$$

where Z is a standard value given by:

$$Z = \frac{R - E(R)}{\sqrt{V(R)}} \tag{5.45}$$

Here, $E(R)$ symbolizes the expectation of runs, and $V(R)$ represents the variance of runs. When the process generates non-metric data, we obtain $E(R)$ and $V(R)$ this way [65]:

$$E(R) = \frac{2n_1n_2}{n} + 1 \tag{5.46}$$

$$V(R) = \frac{2n_1n_2(2n_1n_2 - n)}{(n-1)n^2} \tag{5.47}$$

where n is the total number of observations given. For metric data, we approximate $E(R)$ and $V(R)$ as follows [65]:

$$E(R) \approx \frac{2n - 1}{3} \tag{5.48}$$

$$V(R) \approx \frac{16n - 29}{90} \tag{5.49}$$

Interpretation

When $R < E(R) - |Z|\sqrt{V(R)}$, we have **too few runs**, whereas $R > E(R) + |Z|\sqrt{V(R)}$ implies that we have **too many runs**. In general, having too many or too few runs is indicative of non-randomness in the process data [63]. In non-metric data, having **too few runs** signals the tendency of the data to **cluster** above or below the centerline. The result of **too many runs** implies **zig-zag** behaviors, which may mean a **mixture** of two processes. It follows that [43]:

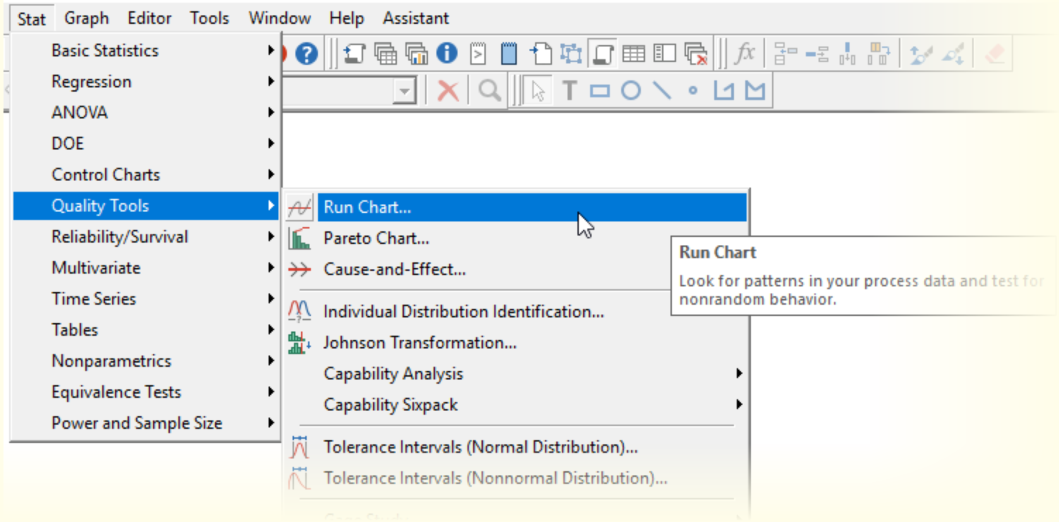
$$\textit{(p-value for mixtures) + (p-value for clustering) = 1} \tag{5.50}$$

For metric data, having **too few runs** implies the tendency of the data to form **trends** above or below the centerline. Having **too many runs** suggests that **oscillation** behaviors exist in the process. It is given that [43]:

$$\textit{(p-value for trends) + (p-value for oscillation) = 1} \tag{5.51}$$

How-To 5.4 (Run charts in Minitab 18)
Click on *Stat* > *Quality Tools*> *Run Chart* > *check the single column radio option* > *select your data* > *Input the subgroup size* > *Plot subgroup medians* > *OK.*

Figure 5.8: The options for run charts in Minitab 18



How-To 5.5 (Python 3.6)

Script 5.2: A script for calculating run statistics in Python 3.6

```
#RUN Statistics
#Import modules
from pandas import *
import numpy as np
```

```
from scipy.stats import norm
#import data using Pandas. The spreadsheet has a column named "Run"
    to represent the data to be monitored.
data = read_excel(your directory)
#initialize parameters
lenData = len(data.Run)
alpha = 0.05#significance level
Z = norm.ppf(alpha)
#count runs from running differences
signs = [0]
for i in range(1, lenData ):
    s = np.sign(data.Run[i]-data.Run[i-1])
    if s>0:
        signs.append(1)
    else:
         signs.append(-1)
runs = array([1. for i in range(1, lenData ) if signs[i] !=signs[i
   -1]])
rundown = array([1. for i in signs if i==-1])
runup= array([1. for i in signs if i==1])
R = runs.sum()
R1 = rundown.sum()
R2 = runup.sum()
n1 = len(rundown)
n2 = len(runup)
n = lenData
ER = (2.*n - 1.)/3.
VR = (16.*n - 29.)/90.
lower = ER-Z*sqrt(VR)
upper = ER+Z*sqrt(VR)
Z0 = (R - ER)/sqrt(VR)
p_value_trend= norm.cdf(Z0)
p_value_oscillation = 1.- p_value_trend
print ('------------------------------------------------------')
#print run statistics
if p_value_trend < alpha:
    if R<lower:
        print ('Reject the null hypothesis for trends. There are too
           few runs.')
        print ('The p-value for trends is %s'%round(p_value_trend,3))
        print ('The p-value for oscillation is %s'%round(1.-
           p_value_trend,3))
        print ('The confidence interval is supposed to be '+str(round
           (min(lower, upper),2))+ ' <= R <= '+str(round(max(lower,
           upper),2))+', but  R = ' +str(round(R,2))+'.')
    else:
        print ('Reject the null hypothesis for trends. There are too
           many runs.')
        print ('The p-value for trends is %s'%round(p_value_trend,3))
        print ('The p-value for oscillation is %s'%round(1.-
           p_value_trend, 3))
        print ('The confidence interval is supposed to be '+str(round
           (min(lower, upper),2))+ ' <= R <= '+str(round(max(lower,
           upper),2))+', but R = ' +str(round(R,2))+'.')
else:
    print ('Failed to reject the null hypothesis for trends')
    if p_value_oscillation <alpha:
        print ('Reject the null hypothesis for oscillation')
    print ('The p-value for trends is %s'%round(p_value_trend,3))
    print ('The p-value for oscillation is %s'%round(1.-
       p_value_trend,3))
    print ('The confidence interval is '+str(round(min(lower, upper)
       ,2))+ ' <= R <= '+str(round(max(lower, upper),2))+' and R = '
        +str(round(R,2))+'.')
#
    ****************************************************************************

#count runs above and below the median statistic
median = np.median(data.Run)
signs = [0]
for i in range(0, lenData ):
    s = np.sign(data.Run[i]-median)
    if s>0:
        signs.append(1)
    else:
         signs.append(-1)
runs = 0
for i,j in enumerate(signs):
    try:
        if signs[i+1]!=signs[i]:
            runs+=1
    except:
        pass
rundown = array([1. for i in signs if i==-1])
runup= array([1. for i in signs if i==1])
R = runs
R1 = rundown.sum()
R2 = runup.sum()
n1 = len(rundown)
n2 = len(runup)
n = lenData
ER = (2.*n1*n2/n) + 1.
VR = 2.*n1*n2*(2.*n1*n2 - n)/((n-1.)*(n**2))
lower = ER-Z*sqrt(VR)
upper = ER+Z*sqrt(VR)
Z0 = (R - ER)/sqrt(VR)
p_value_cluster = norm.cdf(Z0)
p_value_mixture = 1.- p_value_cluster
#print run statistics
print ('--------------------------------------------------------')
if p_value_cluster < alpha:
    if R<lower:
        print ('Reject the null hypothesis for clustering. There are
           too few runs.')
        print ('The p-value for clustering is %s'%round(
           p_value_cluster,3))
        print ('The p-value for mixtures is %s'%round(1.-
            p_value_cluster,3))
        print ('The confidence interval is supposed to be '+str(round
            (min(lower, upper),2))+' <= R <= '+str(round(max(lower,
            upper),2))+', but R = ' +str(round(R,2))+'.')
    else:
        print ('Reject the null hypothesis for clustering. There are
            too many runs.')
        print ('The p-value for clustering is %s'%round(
            p_value_cluster,3))
        print ('The p-value for mixtures is %s'%round(1.-
            p_value_cluster,3))
        print ('The confidence interval is supposed to be '+str(round
            (min(lower, upper),2))+' <= R <= '+str(round(max(lower,
            upper),2))+', but R = ' +str(round(R,2))+'.')
else:
    print ('Failed to reject the null hypothesis for clustering')
    if p_value_mixture <alpha:
        print ('Reject the null hypothesis for mixtures')
    print ('The p-value for clustering is %s'%round(p_value_cluster
       ,3))
    print ('The p-value for mixtures is %s'%round(1.-p_value_cluster
       ,3))
    print ('The confidence interval is '+str(round(min(lower, upper)
       ,2))+ ' <= R <= '+str(round(max(lower, upper),2))+' and R = '
        +str(round(R,2))+'.')
print ('-------------------------------------------------------')
```

How-To 5.6 (Run charts in Excel 2013)
Excel does not have a built-in option to create run charts, but we can manually program the given formulas as we demonstrate in Examples 5.3 - 5.5.

Example 5.3 (Binary process data)
Metropolis Hospital monitors patient satisfaction. On the survey questionnaire, patients are asked whether they are satisfied (Y) or not satisfied (N) with their hospital stay. The following is the sequence of the responses from the most recent ten discharges:

$$Y,Y,N,Y,N,N,Y,Y,N,Y \tag{5.52}$$

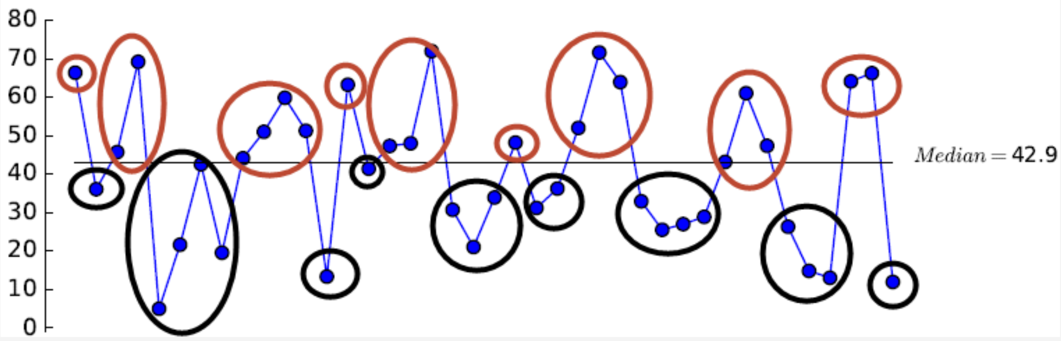
The officials at Metropolis Hospital have asked us to help answer the question of whether these responses are random. To answer this question, we begin by enumerating the runs in the data as follows:

$$\underbrace{Y,Y}_{R1},\overbrace{N}^{R2},\underbrace{Y}_{R3},\overbrace{N,N}^{R4},\underbrace{Y,Y}_{R5},\overbrace{N}^{R6},\underbrace{Y}_{R7} \tag{5.53}$$

Expression 5.53 indicates that we have 7 runs meaning that $R = 7$. We represent run-downs ($R1, R3, R5$, and $R7$) with y_1. We notice that the number of data points in y_1 corresponds to $n_1 = 6$. We represent run-ups $(R2, R4$, and $R6)$ with y_2 where $n_2 = 4$. By consulting reference tables of runs, when $n_1 = 6$ and $n_2 = 4$, we establish the confidence interval of $p(R \leq 2 = 0.010)$ and $p(R > 8) = 1 - p(R \leq 8) = 0.024$. Given that $R = 7$ falls within this confidence interval, **we fail to reject the null hypothesis** that the sequence of patient responses is random.

Example 5.4 (Non-metric data from run charts)
We revisit the run chart depicted in Figure 5.7 and attempt to identify out-of-control behaviors in this chart. We start by counting the number of runs above and below the median, as we illustrate in Figure 5.9.

Figure 5.9: Counting the number of runs in the chart portrayed in Figure 5.7



From Figure 5.9, we identify 9 run-ups with 20 data points. We represent this information with $y_1 = 9$ and $n_1 = 20$. Coincidentally, the information for run-downs is identical as follows: $y_2 = 9$ and $n_2 = 20$. The total number of runs is given by $R = y_1 + y_2 = 9 + 9 = 18$. We use Equations 5.45 - 5.47 to construct the following confidence interval:

$$E(R) = 21 \tag{5.54}$$
$$V(R) = 9.74 \tag{5.55}$$
$$|Z| = 1.96 \tag{5.56}$$
$$15 \leq R \leq 27 \tag{5.57}$$

We set the significance level to $\alpha = 0.05$. Given that $R = 18$ falls within the resulting confidence interval, **we fail to reject the null hypothesis** that this sequence is random. We compute the corresponding *p-value for clustering* as follows:

$$p\text{-}value \equiv \Theta^{-1}(Z_0) = 0.168 \tag{5.58}$$

where Z_0 is the test statistic obtained this way:

$$Z_0 = \frac{18 - 21}{\sqrt{9.74}} = -0.96108 \tag{5.59}$$

Since $p\text{-}value > 0.05$, we reject the null hypothesis that the process does not have clusters. The *p-value for mixtures* is given by $1 - 0.168 = 0.832$, which leads us to conclude that there is no statistically significant process mixture in this example.

Example 5.5 (Metric data from run charts)
Let us revisit Example 5.4, but this time investigate the randomness in the sequence of successive numerical differences using the data in Table 5.5.

Table 5.5: Data related to the run chart in Figure 5.9

Sample	Data	*Sample*	Data
1	66.36	*21*	33.92
2	36.08	*22*	48.2
3	45.72	*23*	31.2
4	69.2	*24*	36.24
5	5.0	*25*	52.0
6	21.64	*26*	71.6
7	42.56	*27*	63.92
8	19.56	*28*	32.92
9	44.12	*29*	25.52
10	51.0	*30*	27.0
11	59.84	*31*	28.88
12	51.28	*32*	43.16
13	13.36	*33*	61.04
14	63.24	*34*	47.36
15	41.4	*35*	26.32
16	47.32	*36*	14.84
17	48.0	*37*	13.04
18	71.96	*38*	64.12
19	30.72	*39*	66.24
20	21.04	*40*	11.96

We begin by determining the signs of the running differences, as illustrated in Table 5.6. This table has two derived columns. We use the column of $SIGN$ to track the signs of the differences and the column of $COUNT\ SIGN$ to identify runs. For example, we determined the sign in cell B3 using the $SIGN()$ function in Excel as follows: $SIGN(36.08 - 66.36) = -$. Likewise, we obtained the sign in cell B4 this way: $SIGN(45.72 - 36.08) = +$. To automate this procedure, we encoded the following formula in cell B3:

$$= IF(SIGN(A3 - A2) = 1, ''+'', ''-'') \tag{5.60}$$

Next, we dragged down this formula to populate the rest of the signs. In the $COUNT\ SIGN$ column, we encoded 1 if a run existed per Definition 5.2. For example, the value of 1 in cell C3 was obtained using $IF(B4 = B5, 0, 1)$. Dragging down this formula allowed us to populate the rest of the values. The sum of the $COUNT\ SIGN$ column gave us the total number of runs R. To find the number in the run-ups (+) or n_1, we used this formula:

$$= COUNTIF(B3 : B41, ''+'') \tag{5.61}$$

Correspondingly, we found the number of the run-downs(-) or n_2 like this:

$$= COUNTIF(B3 : B41, ''-'') \tag{5.62}$$

In the end, we determined that $R = 19$, $n_1 = 22$, and $n_2 = 17$.

Table 5.6: A spreadsheet setup for creating run tests of the data in Table 5.5

	A	B	C
1	**Data**	**SIGN**	**COUNT SIGN**
2	66.36		
3	36.08	-	1
4	45.72	+	0
5	69.2	+	1
6	5.0	-	1
7	21.64	+	0
8	42.56	+	1
9	19.56	-	1
10	44.12	+	0
11	51.0	+	1
...	...	...	...

Finally, we applied the formulas in Equations 5.45, 5.48, and 5.49 to find:

$$E(R) = 26.3 \tag{5.63}$$

$$V(R) = 6.78 \tag{5.64}$$

$$|Z| = 1.96 \tag{5.65}$$

$$21.2 \le R \le 31.4 \tag{5.66}$$

Since $R = 19$ falls outside of the confidence interval, **we reject the null hypothesis** at the significance level of 0.05. Additionally, since $R < E(R) - |Z|\sqrt{V(R)}$, that is $19 < 21.2$, we conclude that a **trend** is likely in the process. To estimate the p-$value$ for the trend, we proceed this way:

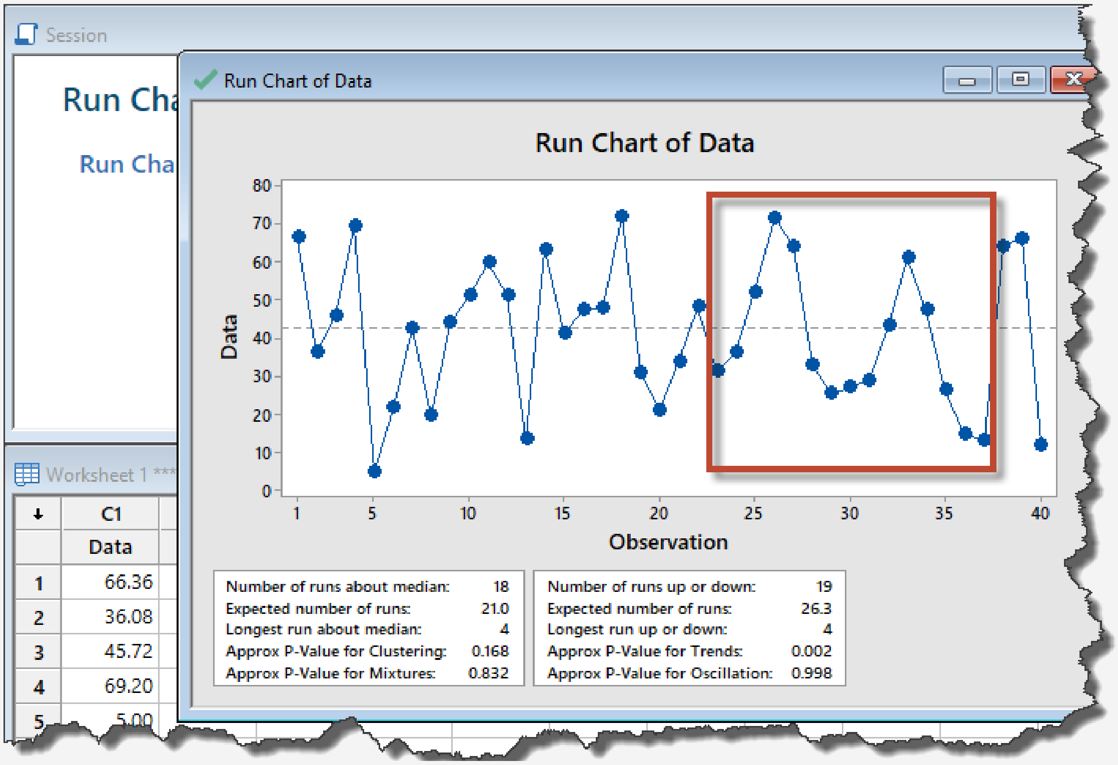
$$Z_0 = \frac{19 - 26.3}{\sqrt{6.78}} = -2.814 \tag{5.67}$$

We obtain the probability of interest this way:

$$p\text{-}value \equiv \Theta^{-1}(Z_0) = 0.002 \tag{5.68}$$

Since p-$value < 0.05$, we reject the null hypothesis that there is no trend in the data. We obtain the p-$value$ for oscillation by $1 - 0.002 = 0.998$. Since p-$value$ > 0.05, we conclude that no statistically significant oscillation exists in this process. By following instructions in How-To 5.4, we reproduced similar results in Minitab, as portrayed in Figure 5.10. We used a subgroup of size 1.

Figure 5.10: The run chart in Minitab based on the data in Table 5.5



↓	C1
	Data
1	66.36
2	36.08
3	45.72
4	69.20
5	5.00

```
Number of runs about median:      18
Expected number of runs:        21.0
Longest run about median:          4
Approx P-Value for Clustering: 0.168
Approx P-Value for Mixtures:   0.832
```

```
Number of runs up or down:         19
Expected number of runs:         26.3
Longest run up or down:             4
Approx P-Value for Trends:      0.002
Approx P-Value for Oscillation: 0.998
```

The boxed area in Figure 5.10 shows sequences that exhibit trend-like behaviors. As noted earlier, unlike control charts, run charts don't allow us to pinpoint special cause variation to a particular sample number.

5.5 EXERCISES

1. At the confidence interval of 99.73%, what is the LDL of an ANOM chart in Minitab with p = 0.10, the constant sample size of 100, and 10 health care organizations in the study?

2. From an audit of 30 orders for the diagnostic test Z, you found the average entry time in the CPOE system to be 7 minutes with $\bar{R} = 3$. Given LSL = 5 minutes and USL = 10 minutes:

   (a) What is the capability of this process?

   (b) What can you conclude about the central tendency of this process?

3. After further analysis, it was determined that hospitals with the IDs of 3, 8, and 9 were not in the same statistical state as the rest of the hospitals. Remove these hospitals from Table 5.2. Next,

   (a) Create a new funnel chart using Excel or Python

   (b) Approximate your results using the ANOM chart in Minitab.

   (c) Estimate the C_{pu} ratio of the process given $USL = 15\%$.

   (d) What can you conclude?

4. From 25 samples, each with size 10, you estimated that your hospital's EHR average downtime was 30 minutes with the standard deviation of 2 minutes. Your process is assumed stable and normally distributed. Given $LSL = 10$ minutes and $USL = 45$ minutes:

   (a) What is the **potential** capability of the system?

   (b) What is the **actual** capability of the system?

   (c) What is the percentage of the allowed bandwidth is the process using?

   (d) What is the probability of observing a downtime period longer than 45 minutes?

5. Using Excel or Python, reproduce the run statistics in Examples 5.4 and 5.5.

6. The Chief Nursing Officer (CNO) at Metropolis Hospital has just asked you to help create an appropriate monitoring mechanism for urinary tract infections (UTIs) at the hospital. The CNO knows that UTIs are rare at this hospital and, therefore would like you to monitor the number of days between events. The CNO handed you the data presented in Table 5.7.

   (a) Create an appropriate control chart

   (b) Create a corresponding run chart

   (c) Compare and contrast your results. What can you conclude?

Table 5.7: Days between UTI infections at Metropolis Hospital, 2014 - 2018

UTI#	**Days** Between	*UTI#*	**Days** Between
1	61	*17*	25
2	46	*18*	17
3	46	*19*	15
4	102	*20*	35
5	38	*21*	14
6	11	*22*	20
7	22	*23*	24
8	91	*24*	62
9	18	*25*	31
10	31	*26*	60
11	77	*27*	18
12	17	*28*	43
13	94	*29*	30
14	2	*30*	16
15	31	*31*	43
16	66	*32*	29

7. The Chief Financial Officer (CFO) at Central City Hospital would like to start monitoring the costs for total hip replacements. Table 5.8 presents samples of 40 individual patients. Using Excel, create appropriate control and run charts for the CFO. What can you conclude from both charts?

8. At Gotham Hospital, the CIO keeps track of the queue time of IT tickets. Table 5.9 portrays the samples that the manager has collected. Use this data to create an appropriate run and control chart.

   (a) What new insights did the run chart reveal as compared to control charts?

   (b) Is the process capable of meeting USL = 180 minutes? Show your work and explain your results.

9. The Chief Medical Officer (CMO) at Metropolis Hospital keeps track of the time to extubation of ICU patients. Table 5.10 presents monthly sample statistics of this process. The sample size is $n = 13$, $\mu = 6$, and $\sigma = 3$. Create appropriate control charts, **with** and **without** the standards.

   (a) What can you conclude about the stability of the process?

   (b) What can you infer about the capability of the process given specification limits set to $\mu \pm L\sigma$ where $L = 2.878$?

   (c) Any ideas about how to improve the stability and capability of the process?

Table 5.8: Costs in thousand for total hip replacements at Central City Hospital, 2018

Sample#	**X**	***Sample#***	**X**
1	34	*21*	33
2	41	*22*	35
3	40	*23*	28
4	36	*24*	39
5	37	*25*	37
6	35	*26*	36
7	42	*27*	36
8	37	*28*	48
9	31	*29*	39
10	42	*30*	33
11	44	*31*	27
12	35	*32*	28
13	37	*33*	25
14	26	*34*	34
15	27	*35*	42
16	49	*36*	31
17	29	*37*	47
18	30	*38*	29
19	40	*39*	38
20	39	*40*	37

10. Table 5.11 presents the frequency of the monthly overrides of critical alerts in the CPOE system at Metro City Hospital.

    (a) Using Excel or Python, create the corresponding run chart. What out-of-control behaviors can you visually detect?

    (b) Use Excel or Python to calculate run statistics and any appropriate confidence intervals with $\alpha = 0.05$. What can you conclude about the hypothesis of the process being stable? Explain.

    (c) Create an appropriate control chart.

    (d) Compare and contrast the results from your control chart to those from the run chart. What can you conclude?

Table 5.9: Queue time in minutes, clinical IT tickets at Gotham Hospital, 2018

Sample#	**x1**	**x2**	**x3**	**x4**	**x5**	**x6**	**x7**
1	110	45	84	86	58	75	38
2	80	70	110	148	32	84	42
3	22	111	37	100	109	62	63
4	82	67	55	30	129	63	57
5	74	40	82	59	82	43	84
6	38	40	139	100	59	44	85
7	80	123	52	80	48	45	96
8	76	97	54	64	65	75	110
9	64	47	66	135	65	79	91
10	80	35	47	59	50	119	44
11	65	76	118	64	38	49	66
12	60	67	41	86	73	128	88
13	105	49	87	68	105	63	42
14	104	138	50	67	102	54	122
15	51	54	58	81	79	143	61
16	47	60	90	61	66	58	145
17	61	94	104	60	83	50	96
18	32	79	79	99	61	70	84
19	32	101	111	85	48	103	40
20	55	23	97	123	58	47	54
21	46	32	47	55	88	103	80
22	52	130	63	32	50	141	105
23	108	66	104	101	103	100	95
24	90	38	64	64	107	44	84
25	102	48	51	93	55	53	58
26	33	64	57	110	41	114	67
27	37	81	84	90	69	113	52
28	99	96	80	112	84	131	32
29	94	89	114	80	41	96	83
30	78	44	118	58	75	107	63
31	74	123	61	95	109	68	114
32	102	45	63	49	127	78	70
33	79	130	122	51	32	61	62
34	102	31	68	97	68	109	99
35	122	62	56	87	38	70	77
36	81	104	85	52	57	40	28
37	79	43	86	72	57	110	50
38	49	95	71	132	90	112	82
39	200	46	99	150	90	103	80
40	83	72	46	80	44	87	36
41	123	44	108	73	56	51	86
42	96	114	106	33	56	90	101
43	37	53	44	46	50	70	82
44	71	82	112	85	102	43	119
45	126	37	40	31	47	100	120
46	104	87	113	79	59	79	59
47	89	95	54	64	61	102	66
48	61	44	50	59	33	63	73
49	43	53	66	94	95	82	80
50	45	65	56	84	62	24	107

Table 5.10: Monthly sample statistics of the time to extubation process, Metropolis Hospital, 2015-2018

Week #	$\bar{x}$	$\bar{s}$
1	5.2	2.4
2	6.3	4.5
3	6.5	3.9
4	7.5	2.9
5	6.9	3.1
6	5.7	3.3
7	7.3	2.1
8	6.8	3.9
9	5	2.3
10	6.8	2.5
11	5.9	3.7
12	8.3	3.5
13	5.1	4.2
14	5.5	3.7
15	5.6	3.2
16	6.3	3.8
17	6.9	3.4
18	7.2	5.8
19	6.5	2.4
20	6.1	3.1
21	5.6	2.7
22	5.3	2.7
23	7.9	7.9
24	5.8	3.6
25	3.4	2
26	6.7	4.6
27	7.5	2.7
28	6.2	2.3
29	5	3.4
30	6.1	1.9

Table 5.11: Critical alert overrides in the CPOE system at Metro City Hospital, 2014-2018

Month	#Overrides
1	42
2	46
3	17
4	25
5	41
6	38
7	23
8	55
9	28
10	12
11	14
12	26
13	28
14	32
15	11
16	53
17	20
18	5
19	8
20	15
21	40
22	47
23	47
24	9
25	44
26	24
27	8
28	23
29	36
30	55
31	63
32	60
33	53
34	26
35	26
36	13
37	11
38	8
39	29
40	75

Python 3.6 Scripts

.1 Appendix

Table 12: Factors for Constructing Variable Control Charts

	Chart for Averages			Chart for standard deviations					Chart for Ranges						
	Factors forControl Limits			Factors for:					Factors for CL		Factors for Control Limits				
				CL	Control Limits										
n	A	A_2	A_3	c_4	B_3	B_4	B_5	B_6	d_2	$1/d_2$	d_3	D_1	D_2	D_3	D_4
2	2.121	1.880	2.659	0.7979	0	3.267	0	2.606	1.128	0.8862	0.852	0	3.686	0	3.266
3	1.732	1.023	1.954	0.8862	0	2.568	0	2.276	1.693	0.5908	0.888	0	4.357	0	2.574
4	1.500	0.729	1.628	0.9213	0	2.266	0	2.088	2.059	0.4857	0.879	0	4.697	0	2.281
5	1.342	0.577	1.427	0.9400	0	2.089	0	1.964	2.326	0.4299	0.864	0	4.918	0	2.114
6	1.225	0.483	1.287	0.9515	0.030	1.970	0.029	1.874	2.534	0.3946	0.848	0	5.078	0	2.003
7	1.134	0.419	1.182	0.9594	0.118	1.882	0.113	1.806	2.704	0.3698	0.833	0.206	5.203	0.076	1.924
8	1.061	0.373	1.099	0.9650	0.185	1.815	0.179	1.751	2.847	0.3512	0.819	0.389	5.306	0.137	1.863
9	1.000	0.337	1.032	0.9693	0.239	1.761	0.232	1.707	2.970	0.3367	0.807	0.548	5.392	0.184	1.816
10	0.949	0.308	0.975	0.9727	0.284	1.716	0.276	1.669	3.078	0.3249	0.797	0.688	5.467	0.223	1.777
11	0.905	0.285	0.927	0.9754	0.321	1.679	0.313	1.637	3.173	0.3152	0.787	0.813	5.533	0.256	1.744
12	0.866	0.266	0.886	0.9776	0.354	1.646	0.346	1.610	3.258	0.3069	0.778	0.924	5.593	0.284	1.716
13	0.832	0.249	0.850	0.9794	0.382	1.618	0.374	1.585	3.336	0.2998	0.770	1.026	5.646	0.307	1.693
14	0.802	0.235	0.817	0.9810	0.406	1.594	0.399	1.563	3.407	0.2935	0.763	1.119	5.695	0.328	1.672
15	0.775	0.223	0.789	0.9823	0.428	1.572	0.421	1.544	3.472	0.2880	0.756	1.204	5.739	0.347	1.653
16	0.750	0.212	0.763	0.9835	0.448	1.552	0.440	1.526	3.532	0.2831	0.750	1.283	5.781	0.363	1.637
17	0.728	0.203	0.739	0.9845	0.466	1.534	0.458	1.511	3.588	0.2787	0.744	1.357	5.819	0.378	1.622
18	0.707	0.194	0.718	0.9854	0.482	1.518	0.475	1.496	3.640	0.2747	0.738	1.425	5.855	0.392	1.608
19	0.688	0.187	0.698	0.9862	0.497	1.503	0.490	1.483	3.689	0.2711	0.733	1.490	5.888	0.404	1.596
20	0.671	0.180	0.680	0.9869	0.510	1.490	0.504	1.470	3.735	0.2677	0.728	1.550	5.920	0.415	1.585
21	0.655	0.173	0.663	0.9876	0.523	1.477	0.516	1.459	3.778	0.2647	0.724	1.607	5.950	0.425	1.575
22	0.640	0.167	0.647	0.9882	0.534	1.466	0.528	1.448	3.819	0.2618	0.719	1.661	5.978	0.435	1.565
23	0.626	0.162	0.633	0.9887	0.545	1.455	0.539	1.438	3.858	0.2592	0.715	1.712	6.004	0.444	1.556
24	0.612	0.157	0.619	0.9892	0.555	1.445	0.549	1.429	3.895	0.2567	0.712	1.761	6.030	0.452	1.548
25	0.600	0.153	0.606	0.9896	0.565	1.435	0.559	1.420	3.931	0.2544	0.708	1.807	6.055	0.460	1.540

For $n > 25$:

$$\begin{aligned}
A &= \frac{3}{\sqrt{n}} \\
c_4 &\approx \frac{4(n-1)}{4n-3} \\
A_3 &= \frac{A}{c_4} \\
B_3 &= 1 - \frac{3}{c4\sqrt{2(n-1)}} \\
B_4 &= 1 + \frac{3}{c4\sqrt{2(n-1)}} \\
B_5 &= c_4 - \frac{3}{2(n-1)} \\
B_6 &= c_4 + \frac{3}{2(n-1)}
\end{aligned}$$

For $n \leq 25$ see Table 12.

Bibliography

[1] Minitab 18. Methods and formulas for binomial data in analysis of means. https://support.minitab.com/en-us/minitab/18/help-and-how-to/modeling-statistics/anova/how-to/analysis-of-means/methods-and-formulas/binomial-data/, 2018. [Online; accessed 4-November-2018].

[2] Hervé Abdi. The kendall rank correlation coefficient. *Encyclopedia of Measurement and Statistics. Sage, Thousand Oaks, CA*, pages 508–510, 2007.

[3] Farrokh Alemi and Douglas W Oliver. Tutorial on risk-adjusted p-charts. *Quality Management in Healthcare*, 10(1):1–9, 2001.

[4] Dominik Aronsky, Diane Kendall, Kathleen Merkley, Brent C James, and Peter J Haug. A comprehensive set of coded chief complaints for the emergency department. *Academic Emergency Medicine*, 8(10):980–989, 2001.

[5] Richard Averill, NORBERT Goldfield, JACK Hughes, JOHN MULDOON, J Gay, and ELIZABETH Mc CULLOUGH. What are apr-drgs? an introduction to severity of illness and risk of mortality adjustment methodology, 2003.

[6] Åke Blomqvist. The doctor as double agent: Information asymmetry, health insurance, and medical care. *Journal of Health Economics*, 10(4):411–432, 1991.

[7] Åke Blomqvist and Pierre Thomas Léger. Information asymmetry, insurance, and the decision to hospitalize. *Journal of health economics*, 24(4):775–793, 2005.

[8] George EP Box, Gwilym M Jenkins, and Gregory C Reinsel. *Time series analysis: forecasting and control*. Prentice Hall, 1994.

[9] Forrest W Breyfogle III. *Implementing six sigma: smarter solutions using statistical methods*. John Wiley & Sons, 2003.

[10] Raymond G Carey and Robert C Lloyd. *Measuring quality improvement in healthcare: a guide to statistical process control applications*. ASQ Quality Press, 1995.

[11] Anne B Casto and Elizabeth Forrestal. *Principles of healthcare reimbursement*. Citeseer, 2013.

[12] Mark R Chassin, Jerod M Loeb, Stephen P Schmaltz, and Robert M Wachter. Accountability measures—using measurement to promote quality improvement. `https://www.mnhospitals.org/Portals/0/Documents/ptsafety/Chassin.pdf`, 2010.

[13] Frederick Chee, Tyrone L Fernando, Andrey V Savkin, and Vernon Van Heeden. Expert pid control system for blood glucose control in critically ill patients. *IEEE Transactions on Information Technology in Biomedicine*, 7(4):419–425, 2003.

[14] Jeffrey J Cherian, Bhaveen H Kapadia, Samik Banerjee, Julio J Jauregui, Kimona Issa, and Michael A Mont. Mechanical, anatomical, and kinematic axis in tka: concepts and practical applications. *Current reviews in musculoskeletal medicine*, 7(2):89–95, 2014.

[15] Ka-Chun Cheung, Willem van der Veen, Marcel L Bouvy, Michel Wensing, Patricia MLA van den Bemt, and Peter AGM de Smet. Classification of medication incidents associated with information technology. *Journal of the American Medical Informatics Association*, 21(e1):e63–e70, 2013.

[16] CMS. Eligible hospital and critical access hospital medicaid ehr incentive program stage 3 objectives and measures objective 4 of 8. `https://www.cms.gov/Regulations-and-Guidance/Legislation/EHRIncentivePrograms/Downloads/MedicaidEHStage3_Obj4.pdf`, 2017. [Online; accessed 6-June-2019].

[17] CMS. Quality measures. `https://www.cms.gov/Medicare/Quality-Initiatives-Patient-Assessment-Instruments/QualityMeasures/index.html`, 2018. [Online; accessed 4-November-2018].

[18] Joint Commision. Measures. `https://www.jointcommission.org/core_measure_sets.aspx`, 2018. [Online; accessed 4-November-2018].

[19] David A Cook, Stefan H Steiner, Richard J Cook, Vern T Farewell, and Anthony P Morton. Monitoring the evolutionary process of quality: risk-adjusted charting to track outcomes in intensive care. *Critical care medicine*, 31(6):1676–1682, 2003.

[20] Florbela Correia, Rui Nêveda, and Pedro Oliveira. Chronic respiratory patient control by multivariate charts. *International journal of health care quality assurance*, 24(8):621–643, 2011.

[21] Arnaldo De Mendonça, J-L Vincent, PM Suter, Rui Moreno, NM Dearden, Massimo Antonelli, Jukka Takala, Charles Sprung, and Francis Cantraine. Acute renal failure

in the icu: risk factors and outcome evaluated by the sofa score. *Intensive care medicine*, 26(7):915–921, 2000.

[22] Lawrence T DeCarlo. On the meaning and use of kurtosis. *Psychological methods*, 2(3):292, 1997.

[23] Avedis Donabedian. The quality of care: how can it be assessed? *Jama*, 260(12):1743–1748, 1988.

[24] Michelle Dougherty, Sandra Seabold, and Susan E White. Study reveals hard facts on cac. *Journal of AHIMA*, 84(7):54–56, 2013.

[25] Rose T Dunn. Benchmarking imaging: Making every image count in scanning programs. *Journal of AHIMA*, 78(6):42–46, 2007.

[26] Centers for Medicare and Medicaid Services (CMS). Hospital-acquired conditions. `https://www.cms.gov/Medicare/Medicare-Fee-for-Service-Payment/HospitalAcqCond/Hospital-Acquired_Conditions.html`, 2018. [Online; accessed 3-November-2018].

[27] B Christopher Frueh, Rebecca G Knapp, Karen J Cusack, Anouk L Grubaugh, Julie A Sauvageot, Victoria C Cousins, Eunsil Yim, Cynthia S Robins, Jeannine Monnier, and Thomas G Hiers. Special section on seclusion and restraint: Patients' reports of traumatic or harmful experiences within the psychiatric setting. *Psychiatric Services*, 56(9):1123–1133, 2005.

[28] C.L Gapenski. *Fundamentals of Healthcare Finance, Second Edition*. Health Administration Press, 2012.

[29] David Garvin. Competing on the eight dimensions of quality. *Harvard Business Review*, pages 101–109, 1987.

[30] Michael L George. *Lean six sigma for service*. McGraw-Hill New York, NY, 2003.

[31] Sheldon Greenfield, Sherrie H Kaplan, Richard Kahn, John Ninomiya, and John L Griffith. Profiling care provided by different groups of physicians: effects of patient case-mix (bias) and physician-level clustering on quality assessment results. *Annals of Internal Medicine*, 136(2):111–121, 2002.

[32] Douglas M Hawkins. A cusum for a scale parameter. *Journal of Quality Technology*, 13(4):228–231, 1981.

[33] L Allison Jones-Farmer, Jeremy D Ezell, and Benjamin T Hazen. Applying control chart methods to enhance data quality. *Technometrics*, 56(1):29–41, 2014.

[34] Frank C Kaminsky, James C Benneyan, Robert D Davis, and Richard J Burke. Statistical control charts based on a geometric distribution. *Journal of Quality Technology*, 24(2):63–69, 1992.

[35] Mi Ok Kim, Enrico Coiera, and Farah Magrabi. Problems with health information technology and their effects on care delivery and patient outcomes: a systematic review. *Journal of the American Medical Informatics Association*, 24(2):246–250, 2017.

[36] David C Lay. *Linear Algebra and its Applications, 3rd updated Edition*. Addison Wesley, 2005.

[37] Harold W Lewis. *The foundations of fuzzy control*, volume 10. Springer Science & Business Media, 1997.

[38] Douglas Lind, William Marchal, and Samuel Wathen. *Basic Statistics for Business and Economics with Student CD*. McGraw-Hill/Irwin, 2007.

[39] Douglas A Lind, William G Marchal, and M Wathen. *Statisticl Techniques in Business & Economics*. USA, McGraw-Hill/Irwin, 2007.

[40] XS Lu. Control chart for multivariate attribute processes. *International Journal of Production Research*, 36(12):3477–3489, 1998.

[41] Gianni Marchetti, Massimiliano Barolo, Lois Jovanovic, Howard Zisser, and Dale E Seborg. An improved pid switching control strategy for type 1 diabetes. *ieee transactions on biomedical engineering*, 55(3):857–865, 2008.

[42] Andrew McAfee and Erik Brynjolfsson. *Machine, platform, crowd: Harnessing our digital future*. WW Norton & Company, 2017.

[43] Minitab. Methods and formulas for run chart. `https://support.minitab.com/en-us/minitab/18/help-and-how-to/quality-and-process-improvement/quality-tools/how-to/run-chart/methods-and-formulas/methods-and-formulas/`, 2018. [Online; accessed 4-November-2018].

[44] Douglas C Montgomery. *Introduction to statistical quality control*. John Wiley & Sons, 2007.

[45] Douglas C Montgomery, Cheryl L Jennings, and Murat Kulahci. *Introduction to time series analysis and forecasting*. John Wiley & Sons, 2015.

[46] Jerome Niyirora and Jamol Pender. Optimal staffing in nonstationary service centers with constraints. *Naval Research Logistics (NRL)*, 63(8):615–630, 2016.

[47] Jerome Niyirora and Jun Zhuang. Fluid approximations and control of queues in emergency departments. *European Journal of Operational Research*, 261(3):1110–1124, 2017.

[48] Pamela K Oachs and Amy L Watters. *Health information management: Concepts, principles, and practice*. AHIMA, 2016.

[49] Gregory S Ogrinc. *Fundamentals of Health Care Improvement: A Guide to Improving Your Patient's Care*. Joint Commission Resources, 2012.

[50] Gregory S Ogrinc et al. *Fundamentals of Health Care Improvement: A Guide to Improving Your Patient's Care*. Joint Commission Resources, 2012.

[51] Ananthanarayanan Parasuraman, Valarie A Zeithaml, and Leonard L Berry. Servqual: A multiple-item scale for measuring consumer perc. *Journal of retailing*, 64(1):12, 1988.

[52] Karen L Perell, Audrey Nelson, Ronald L Goldman, Stephen L Luther, Nicole Prieto-Lewis, and Laurence Z Rubenstein. Fall risk assessment measures: an analytic review. *The Journals of Gerontology Series A: Biological Sciences and Medical Sciences*, 56(12):M761–M766, 2001.

[53] Thomas K Ross. *Health Care Quality Management: Tools and Applications*. John Wiley & Sons, 2014.

[54] Laurence Z Rubenstein. Falls in older people: epidemiology, risk factors and strategies for prevention. *Age and ageing*, 35(suppl_2):ii37–ii41, 2006.

[55] Justin Newton Scanlan. Interventions to reduce the use of seclusion and restraint in inpatient psychiatric settings: what we know so far a review of the literature. *International Journal of Social Psychiatry*, 56(4):412–423, 2010.

[56] Subhash Sharma. *Applied multivariate techniques*. John Wiley & Sons, Inc., 1995.

[57] Michael Shepherd. Challenges in health informatics. In *2007 40th Annual Hawaii International Conference on System Sciences (HICSS'07)*, pages 135–135. IEEE, 2007.

[58] Adam J Singer, Peter Viccellio, Henry C Thode Jr, Jay L Bock, and Mark C Henry. Introduction of a stat laboratory reduces emergency department length of stay. *Academic Emergency Medicine*, 15(4):324–328, 2008.

[59] Stefan H Steiner, Richard J Cook, Vern T Farewell, and Tom Treasure. Monitoring surgical performance using risk-adjusted cumulative sum charts. *Biostatistics*, 1(4):441–452, 2000.

[60] Louis J Stewart, David Greisler, and Kenneth J Feldman. Measuring primary care practice performance within an integrated delivery system: a case study. *Journal of Healthcare Management*, 47(4):250–262, 2002.

[61] Johan Thor, Jonas Lundberg, Jakob Ask, Jesper Olsson, Cheryl Carli, Karin Pukk Härenstam, and Mats Brommels. Application of statistical process control in healthcare improvement: systematic review. *BMJ Quality & Safety*, 16(5):387–399, 2007.

[62] Carol Van Doorn, Ann L Gruber-Baldini, Sheryl Zimmerman, J Richard Hebel, Cynthia L Port, Mona Baumgarten, Charlene C Quinn, George Taler, Conrad May, Jay Magaziner, et al. Dementia as a risk factor for falls and fall injuries among nursing home residents. *Journal of the American Geriatrics Society*, 51(9):1213–1218, 2003.

[63] Dennis Wackerly, William Mendenhall, and Richard Scheaffer. *Mathematical statistics with applications*. Nelson Education, 2007.

[64] Mary Waterhouse, Ian Smith, Hassan Assareh, and Kerrie Mengersen. Implementation of multivariate control charts in a clinical setting. *International Journal for Quality in Health Care*, 22(5):408–414, 2010.

[65] Per Winkel and Nien Fan Zhang. *Statistical development of quality in medicine*, volume 62. John Wiley & Sons, 2007.

[66] Terry Zimmer. The t chart in minitab statistical software. `http://www.minitab.com/en-us/Published-Articles/The-T-Chart-in-Minitab-Statistical-Software/`, 2018. [Online; accessed 4-November-2018].

Index