Statistics for Engineering, 4C3/6C3

Assignment 4

Kevin Dunn, kevin.dunn@mcmaster.ca **Due date: 27 February 2013**

Note: Assignment objectives

- Demonstrate ability to perform phase 1 and phase 2 for monitoring charts
- Understand and interpret Cp and Cpk values
- Build and use least squares models in R

Question 1 [12]

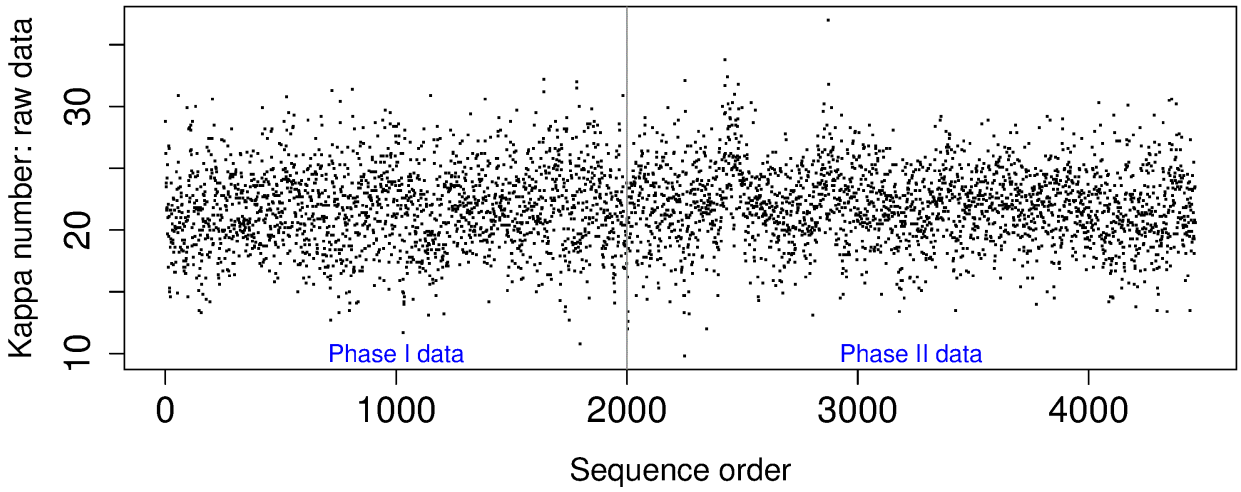
The Kappa number is a widely used measurement in the pulp and paper industry. It can be measured on-line, and indicates the severity of chemical treatment that must be applied to a wood pulp to obtain a given level of whiteness (i.e. the pulp's bleachability). Data on the website contain the Kappa values from a pulp mill. Use the first 2000 data points to construct a Shewhart monitoring chart for the Kappa number. In general, you may use any subgroup size you like, but please use $n = 5$ for ease of grading. Then use the remaining data as your phase 2 (testing) data.

Does the chart perform as expected, i.e. does it capture what you might consider to be unusual operation at the appropriate type-I error rate?

Solution

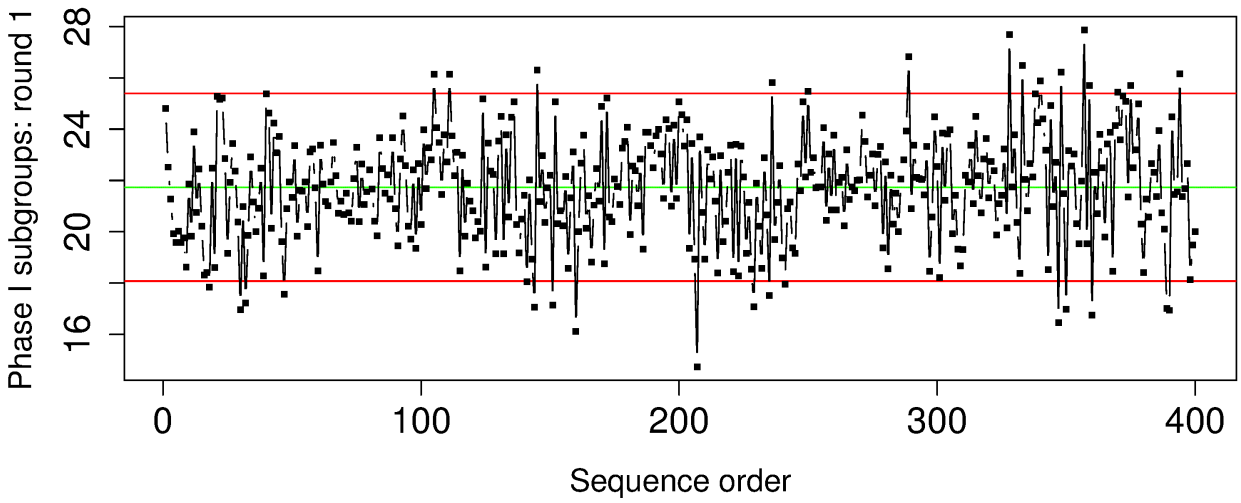
The intention of this question is for you to experience the process of iteratively calculating limits from phase 1 data and applying them to phase 2 data.

The raw data for the entire data set looks as follows. There are already regions in the phase 2 data that we expect to not be from normal operation (around 2500 and 2900)

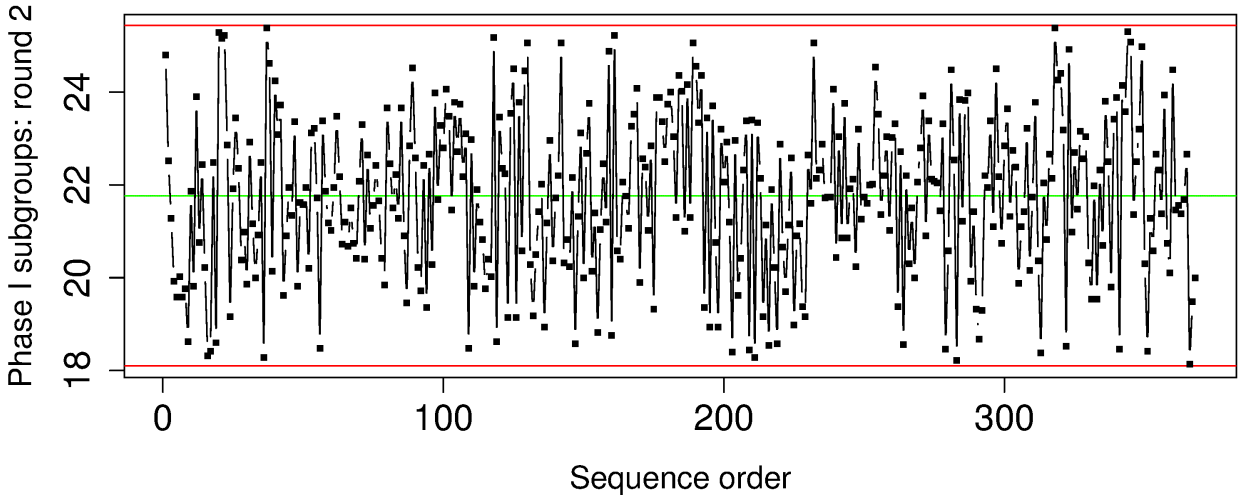


I used subgroups of size 5 for the figures in this answer, however, the code below is very general, and you can regenerate the plots if you chose a different subgroup size. Just change one of the lines near the top.

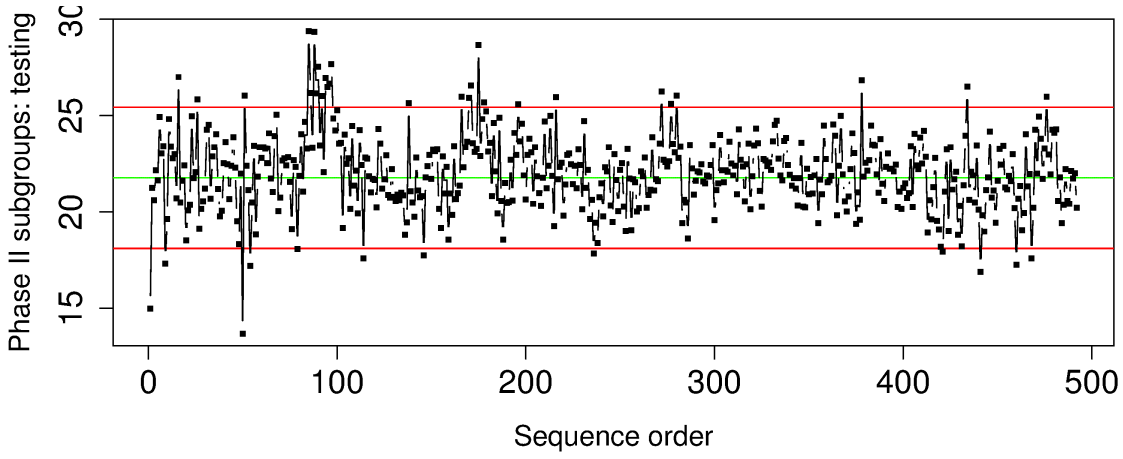
The upper and lower control limits are calculated, and with a subgroup size of $n = 5$, there are 400 subgroups and the limits are: UCL = 18.07, target = 21.73, and UCL = 25.39. This is illustrated on the phase 1 data here:



Next we remove the subgroups which lie outside the limits. Please try using the R code to see how to do it automatically. The new limits, after removing the subgroups beyond the limits from the first round are: LCL = 18.1, target = 21.76 and UCL = 25.43. They barely changed. But the updated plot with subgroups removed is now shown below. There is no need to perform another round of pruning. Only if you used a subgroup size of 4 would you need to do a third round. You could also have just shifted the limits to a different level, for example, to ± 4 standard deviations. We can do this if we have enough process knowledge to understand the implication of it, in terms of profit.



Now apply these control limits to the phase 2 data. The plot is shown below:



The limits identify 2 prolonged periods of unusual operation at sequence point 80 and 180. If we apply the Western Electric rules, we see a third unusual region around sequence step 280. A few other alarms are scattered in the phase 2 data. About 8% of the subgroups lie outside these control limits, so these phase 2 data are definitely not from in-control operation; which we expected from the raw data plot at the start of this question.

The code for all the calculation steps is provided here:

```
kappa <- read.csv('http://datasets.connectmv.com/file/kappa-number.csv')
summary(kappa)
attach(kappa)    # gives access to the variable "Kappa"

N.all <- length(Kappa)
N.phase1 <- 2000
N.phase2 <- N.all - N.phase1
N.subgroup <- 5

phase1.start <- 1
phase1.end <- floor(N.phase1/N.subgroup)
phase2.start <- phase1.end + 1
phase2.end <- floor(N.all/N.subgroup)

# Plot all the data
bitmap('Kappa-raw-data.png', type="png256", width=10, height=4, res=300, pointsize=14)
par(mar=c(4.2, 4.2, 0.5, 0.5))  # (bottom, left, top, right); defaults are par(mar=c(5, 4, 4, 2) + 0
par(cex.lab=1.3, cex.main=1.5, cex.sub=1.5, cex.axis=1.5)
plot(Kappa, type="p", pch=".", cex=2, main="", ylab="Kappa number: raw data",
     xlab="Sequence order")
abline(v=N.phase1, col="gray50")
text(N.phase1/2, 10, "Phase I data", col="blue")
text(N.phase1 + N.phase2/2, 10, "Phase II data", col="blue")
dev.off()

# We won't check the phase I raw data for outliers; we will use the phase I
# subgroups to check for outliers.

# Create the subgroups on ALL the raw data.  Form a matrix with `N.subgroup` rows
# placing the vector of data down each row, then going across to form the columns.

# Calculate the mean and standard deviation within each subgroup (columns of the matrix)
reshaped.data <- matrix(Kappa, N.subgroup, N.all/N.subgroup)
subgroup.x.bar <- apply(reshaped.data, 2, mean)
subgroup.S <- apply(reshaped.data, 2, sd)

phase1.xbar <- subgroup.x.bar[phase1.start:phase1.end]
phase1.S <- subgroup.S[phase1.start:phase1.end]
phase2.xbar <- subgroup.x.bar[phase2.start:phase2.end]
phase2.S <- subgroup.S[phase2.start:phase2.end]

# We are going to repeatedly have to calculate the phase 1 limits.  Create a function.
shewhart_limits <- function(xbar, S, subgroup.size, N.stdev=3){
    # Give the xbar and S vector containing the subgroup means and standard deviations.
    # Also give the subgroup size used.  Returns the lower and upper control limits
    # for the Shewhart chart (UCL and LCL) which are N.stdev away from the target.

    x.double.bar <- mean(xbar)
    s.bar <- mean(S)
    an = c(NA, 0.793, 0.886, 0.921, 0.940, 0.952, 0.959, 0.965)
    LCL <- x.double.bar - 3*s.bar/an[subgroup.size]/sqrt(subgroup.size)
    UCL <- x.double.bar + 3*s.bar/an[subgroup.size]/sqrt(subgroup.size)
    c(LCL, UCL)
```

```
return(list(LCL, x.double.bar, UCL))
}
limits <- shewhart_limits(phase1.xbar, phase1.S, N.subgroup)
LCL <- limits[1]
xdb <- limits[2]
UCL <- limits[3]
c(LCL, xdb, UCL)

# Any points outside these limits?  Yup, quite a few.
bitmap('Kappa-phaseI-first-round.png', type="png256", width=10, height=4, res=300, pointsize=14)
par(mar=c(4.2, 4.2, 0.5, 0.5))  # (bottom, left, top, right); defaults are par(mar=c(5, 4, 4, 2) + 0
par(cex.lab=1.3, cex.main=1.5, cex.sub=1.5, cex.axis=1.5)
plot(phase1.xbar, type="b", pch=".", cex=5, main="", ylab="Phase I subgroups: round 1",
     xlab="Sequence order")
abline(h=UCL, col="red")
abline(h=LCL, col="red")
abline(h=xdb, col="green")
lines(phase1.xbar, type="b", pch=".", cex=5)
dev.off()

# Find the indices of the point outside the limits.  You could use the identify function,
# or you can find them programatically, using boolean (logical) vectors.  Take a look
# at what the variables "outside" and "inside" look like to understand what they do.
outside <- (phase1.xbar > UCL) + (phase1.xbar < LCL)
outside <- as.logical(outside)
inside <- !outside

# Now use only the data inside the existing limits to recalculate the phase I limits.
phase1.xbar <- phase1.xbar[inside]
phase1.S <- phase1.S[inside]
limits <- shewhart_limits(phase1.xbar, phase1.S, N.subgroup)
LCL <- limits[1]
xdb <- limits[2]
UCL <- limits[3]
c(LCL, xdb, UCL)

# Replot the data: everything is inside the limits this time
bitmap('Kappa-phaseI-second-round.png', type="png256", width=10, height=4, res=300, pointsize=14)
par(mar=c(4.2, 4.2, 0.5, 0.5))  # (bottom, left, top, right); defaults are par(mar=c(5, 4, 4, 2) + 0
par(cex.lab=1.3, cex.main=1.5, cex.sub=1.5, cex.axis=1.5)
plot(phase1.xbar, type="b", pch=".", cex=5, main="", ylab="Phase I subgroups: round 2",
     xlab="Sequence order")
abline(h=UCL, col="red")
abline(h=LCL, col="red")
abline(h=xdb, col="green")
lines(phase1.xbar, type="b", pch=".", cex=5)
dev.off()

outside <- (phase1.xbar > UCL) + (phase1.xbar < LCL)
sum(outside)  # yay, it is zero!

# Using subgroups of size 4 or smaller will require an additional
# round of pruning subgroups
outside <- as.logical(outside)
inside <- !outside
phase1.xbar <- phase1.xbar[inside]
phase1.S <- phase1.S[inside]
limits <- shewhart_limits(phase1.xbar, phase1.S, N.subgroup)
LCL <- limits[1]
xdb <- limits[2]
```

```
UCL <- limits[3]
c(LCL, xdb, UCL)

# Replot the data: everything is inside the limits after the second
# or third round of pruning.
bitmap('Kappa-phaseI-third-round.png', type="png256", width=10, height=4, res=300, pointsize=14)
par(mar=c(4.2, 4.2, 0.5, 0.5))  # (bottom, left, top, right); defaults are par(mar=c(5, 4, 4, 2) + 0
par(cex.lab=1.3, cex.main=1.5, cex.sub=1.5, cex.axis=1.5)
plot(phase1.xbar, type="b", pch=".", cex=5, main="", ylab="Phase I subgroups: round 3",
    xlab="Sequence order")
abline(h=UCL, col="red")
abline(h=LCL, col="red")
abline(h=xdb, col="green")
lines(phase1.xbar, type="b", pch=".", cex=5)
dev.off()

outside <- (phase1.xbar > UCL) + (phase1.xbar < LCL)
sum(outside)  # yeah, it is zero!

# Now test the Shewhart limits on the phase II data
bitmap('Kappa-phaseII-testing.png', type="png256", width=10, height=4, res=300, pointsize=14)
par(mar=c(4.2, 4.2, 0.5, 0.5))  # (bottom, left, top, right); defaults are par(mar=c(5, 4, 4, 2) + 0
par(cex.lab=1.3, cex.main=1.5, cex.sub=1.5, cex.axis=1.5)
plot(phase2.xbar, type="b", pch=".", cex=5, main="", ylab="Phase II subgroups: testing",
     xlab="Sequence order")
abline(h=UCL, col="red")
abline(h=LCL, col="red")
abline(h=xdb, col="green")
lines(phase2.xbar, type="b", pch=".", cex=5)
dev.off()
outside.phase2 <- (phase2.xbar > UCL) + (phase2.xbar < LCL)
alpha <- sum(outside.phase2) / length(phase2.xbar)
alpha

# Alpha = 7.9% for this phase 2 data, much higher than the 0.27% expected for
# 3 sigma limits to be expected, because there are process problems on at
# least 3 occasions in this phase 2 data.
```

Question 2 [5]

The following are real data of particle size from the most recent 20 shipments of a speciality powder, taken from the supplier's certificates of analysis:

50.9, 52.9, 51.6, 50.8, 54.6, 52.9, 53.1, 48.4, 51.6, 53.1, 53.8, 52.4, 53.1, 50.8,54.6, 52.9, 50.0, 53.8, 54.6, 52.2

Calculate the supplier's capability, given their lower specification limit of $45\,\mu$m and their upper limit at $59\,\mu$m . Clearly state all assumptions you make during the calculations (median = $52.9\,\mu$m, sd = $1.64\,\mu$m)

How would you interpret this supplier's capability? Would you purchase your raw material from them if you were making a safety-critical product from this supplier (e.g. air-bags or medication or desiccants for a breaking mask)?

Solution

Since we are operating closer to one specification limit than the other, we need to use the Cpk value, and not the Cp value. We also need to make same assumptions:

- The process is operating stably.
- Assumes the particle size data are normally distributed, which was confirmed in a qq-plot, using this code:

```
data <- c(50.9, 52.9, 51.6, 50.8, 54.6, 52.9, 53.1, 48.4, 51.6, 53.1,
          53.8, 52.4, 53.1, 50.8,54.6, 52.9, 50.0, 53.8, 54.6, 52.2)
library(car)
qqPlot(data)
```

We can take the median as the process target, and the standard deviation used for σ, in the absence of better information (you could also use the mean, if you prefer). So with that:

$$\text{Cpk} = \frac{\text{USL} - \text{Target}}{3\sigma} = \frac{59 - 52.9}{(3)(1.64)} = 1.24$$

This Cpk is OK, but not sufficient for companies making safety-critical products, where a Cpk exceeding 1.6 is recommended.

Question 3 [7]

Plastic sheets are manufactured on your blown film line. The Cp value is 1.7. You sell the plastic sheets to your customers with specification of $2\,\text{mm} \pm 0.4\,\text{mm}$.

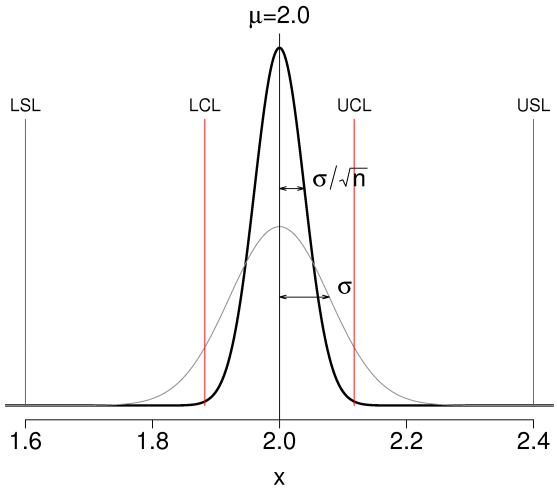
1. List three important assumptions you must make to interpret the Cp value.
2. What is the theoretical process standard deviation, σ?
3. What would be the Shewhart chart limits for this system using subgroups of size $n = 4$?

See the last slide in the course slides to see a visual illustration of this question.

Solution

1. The notes show that Cp values require us to assume that (a) the process values follow a normal distribution, the process was centered when the data were collected, and (c) that the process was stable (use a monitoring chart to verify this last assumption).
2. The range from the lower to the upper specification limit is 0.8 mm, which spans 6 standard deviations. Given the Cp value of 1.7, the process standard deviation must have been $\sigma = \frac{0.8}{1.7 \times 6} = 0.0784$ mm.
3. This time we have the process standard deviation, so there is no need to estimate it from historical phase 1 data (remember the assumption that Cp and Cpk value are calculated from stable process operation?). The Shewhart control limits would be: $\overline{\overline{x}} \pm 3 \times \frac{\sigma}{\sqrt{n}} = 2 \pm 3 \times 0.0784/2$. The LCL = 1.88 mm and the UCL = 2.12 mm.

An illustration is shown here with the USL, LSL, LCL and UCL, and target values on a normal distribution curve. However, for illustration, I have added to the diagram the distribution for the Shewhart chart (thicker line) and distribution for the raw process data (thinner line).



The R code used to generate this figure:

```
n = 4                    # subgroup size
Cp = 1.7
x.range = 0.8
x.sd = x.range/Cp/6      # process std.dev.
x.bar = 2                # process target
LSL = x.bar - 0.4
USL = x.bar + 0.4
LCL = x.bar - 3 * x.sd / sqrt(n)
UCL = x.bar + 3 * x.sd / sqrt(n)

# A vector of 500 equally spaced points, with 10% offset on either side
x <- seq(LSL-0.1*x.range, USL+0.1*x.range, x.range/500)
px <- dnorm(x, mean=x.bar, sd=x.sd)
p.shewhart <- dnorm(x, mean=x.bar, sd=x.sd/sqrt(n))

plot(x, p.shewhart, type="l", xlab=(expression(""*x)), ylab="", frame.plot=FALSE,
                    main=(expression(""*mu*"=2.0")), xlim=c(LSL, USL), cex.lab=1.8,
                    cex.main=1.8, lwd=3, cex.sub=1.8, cex.axis=1.8, yaxt="n")
lines(x, px, type="l", col="gray50")
abline(v=x.bar)

upper.limit = max(p.shewhart)*0.8
segments(x0=LSL,y0=0, x1=LSL, y1=upper.limit, col="gray30")
segments(x0=USL,y0=0, x1=USL, y1=upper.limit, col="gray30")
segments(x0=LCL,y0=0, x1=LCL, y1=upper.limit, col="red")
segments(x0=UCL,y0=0, x1=UCL, y1=upper.limit, col="red")
text(LSL, upper.limit, "LSL", cex=1.3, pos=3)
text(USL, upper.limit, "USL", cex=1.3, pos=3)
text(LCL, upper.limit, "LCL", cex=1.3, pos=3)
text(UCL, upper.limit, "UCL", cex=1.3, pos=3)

# Sigma for the process
y <- dnorm(x.bar+x.sd, mean=x.bar, sd=x.sd)
arrows(x0=x.bar, y0=y, x1=x.bar+x.sd, y1=y, code=3, angle=15, length=0.1)
text(x.bar + x.sd, y+0.2, (expression(""*sigma)), cex=1.8, pos=4)

# Sigma for the Shewhart chart
y <- dnorm(x.bar+x.sd/sqrt(n), mean=x.bar, sd=x.sd/sqrt(n))
arrows(x0=x.bar, y0=y, x1=x.bar+x.sd/sqrt(n), y1=y, code=3, angle=15, length=0.1)
text(x.bar+x.sd/sqrt(n), y+0.2, (expression(""*sigma/sqrt(n))), cex=1.8, pos=4)
```

Question 4 [10 = 3 + 2 + 3 + 1 + 1]

Parts 1 and 2 from a previous midterm, 2010

The production of low density polyethylene is carried out in long, thin pipes at high temperature and pressure (1.5 kilometres long, 50mm in diameter, 500 K, 2500 atmospheres). One quality measurement of the LDPE is its melt index. Laboratory measurements of the melt index can take between 2 to 4 hours. Being able to predict this melt index, in real time, allows for faster adjustment to process upsets, reducing the product's variability. There are many variables that are predictive of the melt index, but in this example we only use a temperature measurement that is measured along the reactor's length.

These are the data of temperature (K) and melt index (units of melt index are "grams per 10 minutes").

Temperature = T [Kelvin]	441	453	461	470	478	481	483	485	499	500	506	516
Melt index = m [g per 10 mins]	9.3	6.6	6.6	7.0	6.1	3.5	2.2	3.6	2.9	3.6	4.2	3.5

The following calculations have already been performed:

- Number of samples, $n = 12$
- Average temperature = $\overline{T} = 481$ K
- Average melt index, $\overline{m} = 4.925$ g per 10 minutes.
- The summed product, $\sum_i \left(T_i - \overline{T}\right)\left(m_i - \overline{m}\right) = -422.1$
- The sum of squares, $\sum_i \left(T_i - \overline{T}\right)^2 = 5469.0$

1. Use this information to build a predictive linear model for melt index from the reactor temperature.
2. What is the model's standard error and how do you interpret it in the context of this model? You might find the following software software output helpful, but it is not required to answer the question.

```
Call:
lm(formula = Melt.Index ~ Temperature)

Residuals:
    Min      1Q  Median      3Q     Max
-2.5771 -0.7372  0.1300  1.2035  1.2811

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) --------    8.60936   4.885 0.000637
Temperature --------    0.01788  -4.317 0.001519

Residual standard error: 1.322 on 10 degrees of freedom
Multiple R-squared: 0.6508,     Adjusted R-squared: 0.6159
F-statistic: 18.64 on 1 and 10 DF,  p-value: 0.001519
```

3. Calculate the ANOVA table entries:
   - RegSS
   - RSS
   - TSS
4. Prove to yourself that $R^2 = \dfrac{\text{RegSS}}{\text{TSS}}$. Ensure that the value agrees with the `R` output above. Also ensure you can reproduce the R output.
5. Prove to yourself that you get the identical R^2 value *from only using the raw data* by calculating $R^2 = [r(x, y)]^2$.

Solution

1. The simplest linear predictive model possible is $m = \beta_0 + \beta_1 T + \varepsilon$, predicting the melt index from temperature. Once we find estimates for these coefficients we write: $m = b_0 + b_1 T + e$. And one way to calculate these coefficients is by least squares. In the class notes we showed that for a variable x used to predict a variable y that:

$$b_0 = \overline{\mathrm{y}} - b_1 \overline{\mathrm{x}}$$
$$b_1 = \frac{\sum_i \left(x_i - \overline{\mathrm{x}}\right)\left(y_i - \overline{\mathrm{y}}\right)}{\sum_i \left(x_i - \overline{\mathrm{x}}\right)^2}$$

Using the pre-calculated values, and that in our case $T = x$, and that $m = y$

$$b_1 = \frac{-422.1}{5469.0} = -0.0772 \frac{\text{g per 10 minutes}}{K}$$
$$b_0 = 4.925 + 0.0772 \times 481 = 42.0 \text{g per 10 minutes}$$

A predictive model of melt flow is: $\hat{m} = 42.0 - 0.0772 \times T$

1. The standard error, S_E can be read directly from the software output as 1.322 g per 10 minutes. If you like, you could also have calculated it by hand, using the above predictive model, calculating residuals ($e_i = m_i - \hat{m}_i$), from which the standard error is $\sqrt{\frac{\sum_i^n e_i^2}{n-k}}$, where $n = 12$ and $k = 2$ (there are 2 parameters in the model). However I recommend you always use the software output and avoid these tedious hand calculations.

The interpretation of the standard error for this model is that the approximate prediction error of melt index has a standard deviation of 1.322 grams per 10 minutes (if the residuals are normally distributed).

END