Practice problems for graphics with partial answers

1. Reproduce the barley experiment plot in the Trellis plotting section of the notes. The data is in the barley data frame of the lattice package.

The help file for the barley data set (type help(barley)) shows example code at the end of it. When I ran the code, I found the variety labels overlapped each other in my default graphics window. There are a few ways to handle this problem. First, make the graphics window bigger! Second, change the layout so that there are more columns of plots. This second choice does make the plot a “little” less useful because you cannot as easily compare all locations. Below is an example of a plot with 3 rows and 2 columns of panels:

> dotplot(x = variety ~ yield | site, data = barley, groups = year,

 auto.key = TRUE, xlab = "Barley Yield (bushels/acre)", layout = c(2,3))



1. My Computational Statistics course uses graphics to better understand the results of a Monte Carlo simulation. In one class project, students evaluated the following six methods to calculate a confidence interval for a variance:
* Normal-based
* Asymptotic
* Basic bootstrap
* Percentile bootstrap
* Bootstrap corrected accelerated (BCa)
* Studentized bootstrap

While the actual methods are not important for this homework problem, next is a brief description of them. The normal-based interval is what one typically learns about in Statistical Methods in Research. Specifically, if s2 denotes the sample variance, σ2 denotes the population variance, and n denotes the sample size, the interval is



where  is the 1 – α/2 quantile from a chi-square distribution with n – 1 degrees of freedom. The asymptotic interval is



where , yi is the ith sampled value, and  is the 1 – α/2 quantile from a standard normal distribution. This interval is derived by using the central limit theorem. The remaining four methods are known as bootstrap-based methods. The confidence level was set to 95% (α = 0.05) for these intervals throughout the project.

Students simulated 500 data sets from a specified probability distribution (gamma, logistic, uniform, exponential, or normal) with a known value of σ2 which was the same for each distribution. The confidence interval methods were applied to each of the data sets and the proportion of times that the interval contained σ2 was recorded. For example, when data was simulated from a N(2.7133, 2.19562) distribution, the proportion of times that the normal-based interval contained σ2 was 477/500 = 0.954. This is known as the “estimated” confidence level. Because only 500 simulated data sets were generated, the range that we would expect all estimated confidence levels to fall within are



if the confidence interval method works as stated. Because the normal-based method with normally distributed data resulted in a value within the expected range, this indicates the interval worked well for this case. You will notice for this problem that many of the confidence intervals methods often do not work well.

The simulation results are given in the file SimResults.csv which is available from my course website. Below is a partial listing of the results:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CI** | **Coverage** | **ExpLength** | **NA** | **Distribution** | **SampleSize** |
| Normal-based | 0.794 | 16.33 | 0 | Gamma | 9 |
| Asymptotic | 0.603622 | 7.98 | 3 | Gamma | 9 |
| Basic | 0.644 | 8.28 | 0 | Gamma | 9 |
| Percentile | 0.63 | 8.28 | 0 | Gamma | 9 |
| BCa | 0.674 | 9.47 | 0 | Gamma | 9 |
| Studentized | 0.9 | 128.81 | 0 | Gamma | 9 |
| Normal-based | 0.786 | 7.99 | 0 | Gamma | 20 |
| Asymptotic | 0.766 | 7.62 | 0 | Gamma | 20 |
|  |  |  |  |  |  |
| BCa | 0.918 | 3.87 | 0 | Normal | 50 |
| Studentized | 0.934 | 4.4 | 0 | Normal | 50 |
| Normal-based | 0.954 | 2.78 | 0 | Normal | 100 |
| Asymptotic | 0.924 | 2.57 | 0 | Normal | 100 |
| Basic | 0.928 | 2.59 | 0 | Normal | 100 |
| Percentile | 0.922 | 2.59 | 0 | Normal | 100 |
| BCa | 0.946 | 2.72 | 0 | Normal | 100 |
| Studentized | 0.954 | 2.88 | 0 | Normal | 100 |

The columns represent:

* CI = Confidence interval methods
* Coverage = The proportion of times that the confidence interval contained σ2
* ExpLength = Average length of the confidence interval across all simulated data sets
* NA = Number of times out of 500 that a confidence interval could not be calculated
* Distribution = The distribution from which the data was simulated
* SampleSize = The sample size used for each of the 500 simulated data sets.

Complete the following:

* Examine the results using the graphical methods discussed in the notes. Discuss which plots are the best to use in this situation.
* Develop an overall conclusion about which method(s) are best.
* Do you think the normal-based method typically taught in STAT 801 is good to use in practice? Explain your answer.

Below is some of my code and plots:

> set1 <- read.csv("SimResults.csv")

> head(set1)

 CI Coverage ExpLength NA Distribution SampleSize

1 Normal-based 0.7940000 16.33 0 Gamma 9

2 Asymptotic 0.6036217 7.98 3 Gamma 9

3 Basic 0.6440000 8.28 0 Gamma 9

4 Percentile 0.6300000 8.28 0 Gamma 9

5 BCa 0.6740000 9.47 0 Gamma 9

6 Studentized 0.9000000 128.81 0 Gamma 9

> library(lattice)

> #Very simple plot for confidence level

> dotplot(CI ~ Coverage | Distribution, data = set1, groups = SampleSize,

 auto.key = TRUE, xlab = "Estimated true confidence level", layout = c(1,5),

 ylab = "Confidence interval method")



> #A nicer plot for confidence level

> #This is one way to obtain all of the sample sizes and put into a vector where

 the elements are characters. A more simple (but less general) way is to just

 manually enter the sample size levels as plot.levels <- c("9", "20", "50",

 "100")

> plot.levels <- levels(factor(set1$SampleSize))

> dotplot(CI ~ Coverage | Distribution, data = set1, groups = SampleSize, main =

 "Confidencel level simulation results", key = list(space = "right", points =

 list(pch = 1:4, col = c("black", "red", "blue", "darkgreen")), text = list(lab

 = plot.levels)),

 panel = function(x, y) {

 panel.grid(h = -1, v = 0, lty = "dotted", lwd = 1, col="lightgray")

 panel.abline(v = 0.95, lty = "solid", lwd = 0.5)

 panel.abline(v = c(0.925, 0.975), lty = "dotted", lwd = 0.5)

 panel.xyplot(x = x, y = y, col = c(rep("black", times = 6), rep("red",

 times = 6), rep("blue", times = 6), rep("darkgreen", times = 6)), pch

 = c(rep(1,6), rep(2,6), rep(3, 6), rep(4, 6)))

 },

 xlab = "Estimated true confidence level", layout = c(1,5), ylab = "Confidence

 interval method")



> #Expected length – I restricted the x-axis here due to some VERY large lengths

 (thus, some lengths may not be shown on the plot)

> dotplot(CI ~ ExpLength | Distribution, data = set1, groups = SampleSize, main =

 "Expected length simulation results", key = list(space = "right", points =

 list(pch = 1:4, col = c("black", "red", "blue", "darkgreen")),

 text = list(lab = plot.levels)), xlim = c(0, 20),

 panel = function(x, y) {

 panel.grid(h = -1, v = 0, lty = "dotted", lwd = 1, col="lightgray")

 panel.xyplot(x = x, y = y, col = c(rep("black", times = 6), rep("red", times

 = 6), rep("blue", times = 6), rep("darkgreen", times = 6)), pch = c(rep(1,6),

 rep(2,6), rep(3, 6), rep(4, 6)))

 },

 xlab = "Estimated expected length", layout = c(1,5), ylab = "Confidence interval

 method")



Overall, the studentized bootstrap interval appears to be the best in terms of the true confidence level, but it can be exceptionally long in length.

Many more interpretations should follow …

For some of the other plots, the “wide” format for the data is needed. Below is how the data can be transformed to this format:

> set1.wide <- reshape(data = set1, timevar = "CI", drop = c("ExpLength", "NA."),

 idvar = c("Distribution", "SampleSize"), direction = "wide", sep = ".")

> options(width = 60) #Helpful for copying output into Word

> head(set1.wide)

 Distribution SampleSize Coverage.Normal-based

1 Gamma 9 0.794

7 Gamma 20 0.786

13 Gamma 50 0.736

19 Gamma 100 0.740

25 Logistic 9 0.928

31 Logistic 20 0.880

 Coverage.Asymptotic Coverage.Basic Coverage.Percentile

1 0.6036217 0.644 0.630

7 0.7660000 0.778 0.770

13 0.8120000 0.804 0.818

19 0.8620000 0.856 0.862

25 0.6887550 0.760 0.738

31 0.8160000 0.826 0.830

 Coverage.BCa Coverage.Studentized

1 0.674 0.900

7 0.820 0.918

13 0.850 0.908

19 0.868 0.912

25 0.806 0.946

31 0.850 0.934

> #If you do not like having "Coverage." in front of the last 8 variables, here's

 one way to change it

> set1.temp <- set1

> names(set1.temp) #List of variable names

[1] "CI" "Coverage" "ExpLength"

[4] "NA" "Distribution" "SampleSize"

> names(set1.temp)[names(set1.temp) == "Coverage"] <- "C" #Within [] this provides

 a set of TRUEs and

FALSEs

> head(set1.temp)

 CI C ExpLength NA Distribution

1 Normal-based 0.7940000 16.33 0 Gamma

2 Asymptotic 0.6036217 7.98 3 Gamma

3 Basic 0.6440000 8.28 0 Gamma

4 Percentile 0.6300000 8.28 0 Gamma

5 BCa 0.6740000 9.47 0 Gamma

6 Studentized 0.9000000 128.81 0 Gamma

 SampleSize

1 9

2 9

3 9

4 9

5 9

6 9

> set1.wide2 <- reshape(data = set1.temp, timevar = "CI", drop = c("ExpLength",

 "NA."), idvar = c("Distribution", "SampleSize"), direction = "wide", sep = ".")

> head(set1.wide2)

 Distribution SampleSize C.Normal-based C.Asymptotic

1 Gamma 9 0.794 0.6036217

7 Gamma 20 0.786 0.7660000

13 Gamma 50 0.736 0.8120000

19 Gamma 100 0.740 0.8620000

25 Logistic 9 0.928 0.6887550

31 Logistic 20 0.880 0.8160000

 C.Basic C.Percentile C.BCa C.Studentized

1 0.644 0.630 0.674 0.900

7 0.778 0.770 0.820 0.918

13 0.804 0.818 0.850 0.908

19 0.856 0.862 0.868 0.912

25 0.760 0.738 0.806 0.946

31 0.826 0.830 0.850 0.934

> options(width = 80)

One of the plots that can be made with the wide-format of the data is a stars plot:

> dev.new(width = 11, height = 7) # Open a new graphics window of size 11”x7”

> stars(x = set1.wide2[ ,3:8], draw.segments = TRUE, key.loc = c(13,12))

> #I figured out the x and y coordinates below by trial and error



A problem with the interpreting the above plot is knowing how far out 0.95 would be for a ray. For example, one may think that the studentized interval does poorly for the gamma distribution. However, the reason for the small rays is that this interval has confidence levels of around 0.91, which is some of the lowest for this interval among the other distributions. Overall, I do not recommend this type of plot to interpret the simulation results.

Another type of plot that can be constructed is a parallel coordinates plot. I used a modified version of parcoord() so that all variables used the exact same y-axis and so that I could view the y-axis scale. Below is my code and output:

> parcoord2 <- function (x, col = 1, lty = 1, x.axis.names = colnames(x), ...)

 {

 matplot(1:ncol(x), t(x), type = "l", col = col, lty =

 lty, xlab = "", ylab = "", axes = FALSE, ...)

 axis(1, at = 1:ncol(x), labels = x.axis.names)

 axis(side = 2)

 for (i in 1:ncol(x)) lines(c(i, i), c(min(x), max(x)), col = "grey70")

 invisible()

 }

> # Customized line types can be specified using numbers where odd

 digit locations specify line lengths and even digit locations specify spaces.

> #For example,

 "4313" results in a line of length 4 units, then a space of 3 units, then a

 line of length 1 unit and finally a space of 3 units before the pattern begins

 again! Thus, a "dash-dot" type of line is formed.

> sample.size.line <- rep(x = c("solid", "43", "4313", "431313"), times = 5)

> parcoord2(x = set1.wide2[,c(3:8)], col = dist.color, main = "Confidence levels",

 lty = sample.size.line, lwd = 2, x.axis.names = c("Normal-based", "Asymptotic",

 "Basic", "Percentile", "BCa", "Studentized"))

> abline(h = 0.95, lwd = 5, lty = "solid")

> abline(h = 0.95 + qnorm(p = c(0.025, 0.975))\*sqrt(0.95\*0.05/500), lwd = 1, lty =

 "dotted")

> legend(locator(1), title = "Distribution legend", legend = c("Gamma", "Logistic",

 "Uniform", "Exponential", "Normal"), lty = c(1,1,1,1,1), col = c("black",

 "red", "green", "blue", "gold"), bty = "n", cex = 0.65, lwd = 2)

> legend(locator(1), title = "Sample size legend", legend = c(9, 20, 50, 100),

 lty = c("solid", "43", "4313", "431313"), col = "black", bty = "n", cex = 0.65,

 lwd = 2, seg.len = 6)



One could get by with a simpler plot, but I purposely customized it to show you what can be done.

1. Use the graphical methods discussed in the notes to examine the goblet data. This data set is in the file goblet.csv on my course website.

Please see an earlier practice problem this semester for how to read in the data and adjust the data by the x3 variable. Below are a few of the plots that I created.

#Stars plot; The "-1" index is a quick way to remove the first column here

# I used the labels argument because the goblet numbers were not included

stars(x = goblet2[,-1], draw.segments = TRUE, key.loc = c(14,10), main = "Goblet

 star plot", labels = goblet2$ID)

#Interestingly, the goblet numbers are not included when I use 2:6 as the column

# number

# stars(x = goblet2[,2:6], draw.segments = TRUE, key.loc = c(14,10), main = "Goblet

# star plot")



> library(MASS)

> parcoord(x = goblet2[,-1], main = "Goblet parallel coordinates plot")



> # A crude way to do brushing

> col.w5 <- ifelse(test = goblet2$w5 <= median(goblet2$w5), yes = "red", no =

 "blue")

> parcoord(x = goblet2[,-1], col = col.w5, main = "Goblet parallel coordinates

 plot")



These plots should be interpreted. One interesting finding here is that it appears a large connection between the base and the cup leads to larger values of w1, w2, and w4. This “trend” that we see in the data could lead to possible classifications that we could put the goblets in. We will discuss this more in later sections.

Many other types of plots can be examined here.