Tooth Growth Analysis

Michael Szczepaniak

May 31, 2017 (latest revision)

Synopsis

This report explores the relationship between the length of teeth in 10 guinea pigs to three Vitimin C dosages and two delivery methods using the R ToothGrowth data set. The data provided evidence that an increased mean tooth length was observed with increased dosage with both delivery methods under a Type I error rate $\alpha = 0.05$. The data also provided evidence that an increased mean tooth length was observed with the orange juice deliver method for the 0.5 and 1.0 mg doses under a Type I error rate $\alpha = 0.05$.

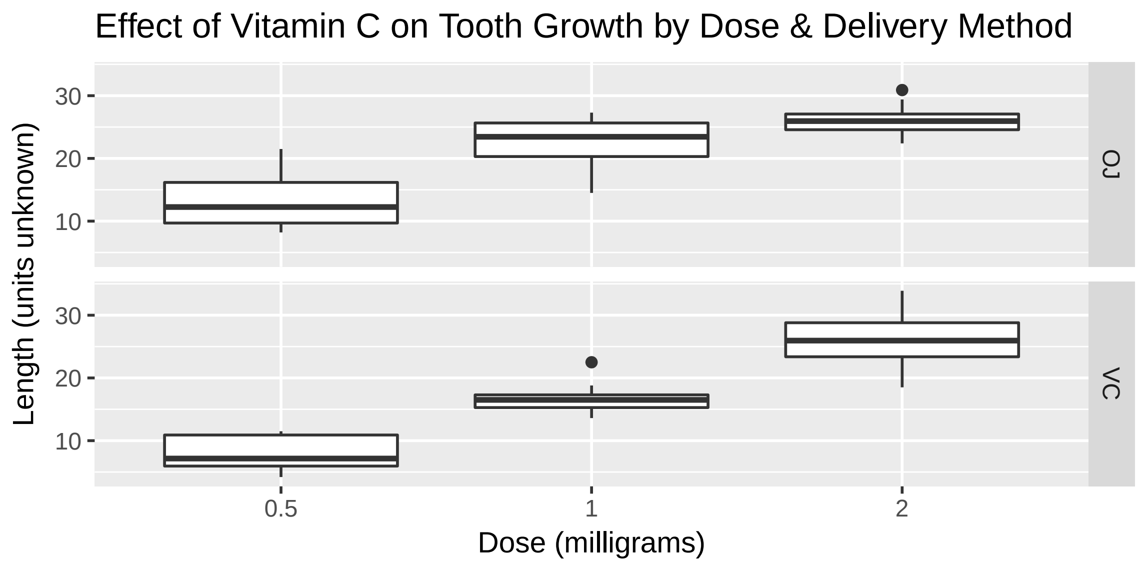
Loading Data & Exploratory Data Analysis

Note: All code used for the report are contained the **Code Used for the Analysis** section.

The *loadToothData* function was used to load the data. The data was then passed to the function *getToothSummary* to create the following summary table. The **Group** variable concatinates the **Delivery** and **Dose** variables.

Group	Mean.Length	Std.Dev.Length
OJ0.5	13.23	4.46
OJ1.0	22.70	3.91
OJ2.0	26.06	2.66
VC0.5	7.98	2.75
VC1.0	16.77	2.52
VC2.0	26.14	4.80

The *loadToothData* function was then passed to the *exploreBox* function to create the boxplot below.



From this initial look at the data, three questions emerge to guide further exploration:

1. Does increased vitamin C dosage lead to an increase or decrease in tooth length?
2. Does vitamin C delivery method lead to an increase or decrease in tooth length?
3. Are the effects of dosage and delivery method on tooth length independent or coupled?

The plot above suggests a possible relationship between increasing dosage and tooth length for both delivery methods. Dosage and delivery method were then explored by hypothesis testing. The independence of the impact of dosage and delivery as stated in the third question above was not explored in this report.

Analysis of the Data

Because the data does not contained ids for each animal, a paired analysis is not possible without assuming the data was ordered in a particular way. For this reason, the analysis will focus on group comparisons. The data contains 6 groups each with 10 measurments and are defined in the summary table below.

To answer the first two questions, the null hypothesis was defined as $H_0 : \mu_A = \mu_B$ (Mean A = Mean B in the testing table) and the alternative hypothesis as $H_a : \mu_B > \mu_A$ (Mean B > Mean A in the testing table) where A and B are the groups defined in the testing table (created by the function *createTestingTable*) shown below.

Test	t1	t2	t3	t4	t5	t6	t7
Addresses	Question 1.	Question 1.	Question 1.	Question 1.	Question 2.	Question 2.	Question 2.
Group A	OJ0.5	OJ1.0	VC0.5	VC1.0	VC0.5	VC1.0	VC2.0
Group B	OJ1.0	OJ2.0	VC1.0	VC2.0	OJ0.5	OJ1.0	OJ2.0
Mean A	13.23	22.7	7.98	16.77	7.98	16.77	26.14
Mean B	22.7	26.06	16.77	26.14	13.23	22.7	26.06

Hypothesis Testing

The function **getTestResults** ran the seven hypothesis tests and generated the result shown below.

```
##   test.num      P_value reject_FPR reject_FWER reject_FDR
## 1        1 8.784919e-05       TRUE        TRUE       TRUE
## 2        2 3.919514e-02       TRUE       FALSE       TRUE
## 3        3 6.811018e-07       TRUE        TRUE       TRUE
## 4        4 9.155603e-05       TRUE        TRUE       TRUE
## 5        5 6.358607e-03       TRUE        TRUE       TRUE
## 6        6 1.038376e-03       TRUE        TRUE       TRUE
## 7        7 9.638516e-01      FALSE       FALSE      FALSE
```

The first column is the test number. Test 1 corresponds to **t1**, 2 to **t2**, etc. The second column is the P-value for the test. The last three columns specify whether the null hypothesis (H_0) should be reject based on the type of multiple testing correction applied.

The **reject__FPR** column makes no multiple testing corrections and only controls the False Positive Rate (FPR). The last two columns of the above results apply multiple testing corrections to two groups. The groupings were defined by which question the test was attempting to address. The first group was constructed from the results of tests 1-4. The second group was constructed from the results of tests 5-7.

The **reject__FWER** reports the results for controlling the Family-wise Error Rate (FWER) by applying a Bonferroni correction to tests 1-4 (m = 4), and to tests 5-7 (m = 3). The **reject__FDR** reports the results for controlling the False Detection Rate (FDR) by applying a Benjamini Hochberg (BH) correction to the two groups previously described.

Assumptions & Conclusions

Assumptions

1. Variation between the populations tested were not equal.
2. Distribution of group means are not skewed (relatively symmetric) and mound-shaped.
3. The guineau pigs were IID normal in their selection.
4. The effects of dosage and delivery method on tooth length are independent.

Conclusions

1. With no multiple testing correction, controling only the FPR, we would reject H_0 in favor of H_a for all tests except **t7**.
2. Controlling FDR using a BH correction yielded the same result as only controlling FPR.
3. If we control FWER using a Bonferroni correction, we would reject H_0 in favor of H_a for all tests except **t7 and t2**.
4. Based on 1. and 2., the evidence of the data supports the hypothesis that an increased mean tooth length will be observed with increased dosage with both delivery methods under a Type I error rate α of 0.05. If we accept the more conservative Bonferroni correction, we would accept H_0 for the 1.0 to 2.0 doses for the orange juice delivery test (test 2).
5. Also based on 1., 2. and 3., the evidence of the data supports the hypothesis that an increased mean tooth length will be observed using the orange juice deliver method for the 0.5 and 1.0 mg dose levels but not for the 2.0 dose level under a Type I error rate α of 0.05

Code Used for the Analysis

```r
## Load the R ToothGrowth dataset and returns an updated data frame
loadToothData <- function() {
    suppressMessages(suppressWarnings(require(dplyr)))
    library(datasets)
    library(dplyr)
    data("ToothGrowth")
    # rename variables/columns to make them clearer
    tooth.data <- rename(ToothGrowth, Length=len, Delivery=supp, Dose=dose)
    # add group id
    tooth.data <- mutate(tooth.data,
                         Group=factor(paste0(Delivery, sprintf("%.1f", Dose))))
    tooth.data <- mutate(tooth.data,  Dose = factor(Dose))
    return(tooth.data)
}

## Summarizes the tooth data by group
getToothSummary <- function(tooth.data = loadToothData()) {
    suppressMessages(suppressWarnings(library(dplyr)))
    suppressMessages(suppressWarnings(library(knitr)))
    library(dplyr)
    tooth.summ <- group_by(tooth.data, Group)
    tooth.table <- summarize(tooth.summ, Mean.Length = mean(Length),
                             Std.Dev.Length = sd(Length))
    library(knitr)
    kable(tooth.table, digits=2)
}

## Creates the first explorator box plox of tooth length by dose and delivery
exploreBox <- function(tooth.data = loadToothData()) {
    library(ggplot2)
    # exploratory panel plot
    p <- ggplot(tooth.data, aes(Dose, Length))
    p <- p + geom_boxplot()
    p <- p + facet_grid(Delivery ~ .)
    p <- p + xlab("Dose (milligrams)") + ylab("Length (units unknown)")
    p <- p + ggtitle("Effect of Vitamin C on Tooth Growth by Dose & Delivery Method")
    print(p)
}

## Creates the table describing the statistical tests to be run on the data
## including the pairings.
createTestingTable <- function(tooth.data = loadToothData()) {
    suppressMessages(suppressWarnings(library(dplyr)))
    suppressMessages(suppressWarnings(library(knitr)))
    library(dplyr)
    tooth.summ <- group_by(tooth.data, Group)
    tooth.table <- summarize(tooth.summ, Mean.Length = mean(Length),
                             Std.Dev.Length = sd(Length))
    df <- data.frame(Test=c("Addresses", "Group A", "Group B", "Mean A", "Mean B"),
                     t1=c("Question 1.", "OJ0.5", "OJ1.0",
                          as.character(tooth.table[1,2]),
                          as.character(tooth.table[2,2])),
                     t2=c("Question 1.", "OJ1.0", "OJ2.0",
                          as.character(tooth.table[2,2]),
                          as.character(tooth.table[3,2])),
                     t3=c("Question 1.", "VC0.5", "VC1.0",
                          as.character(tooth.table[4,2]),
                          as.character(tooth.table[5,2])),
                     t4=c("Question 1.", "VC1.0", "VC2.0",
                          as.character(tooth.table[5,2]),
                          as.character(tooth.table[6,2])),
                     t5=c("Question 2.", "VC0.5", "OJ0.5",
                          as.character(tooth.table[4,2]),
                          as.character(tooth.table[1,2])),
                     t6=c("Question 2.", "VC1.0", "OJ1.0",
                          as.character(tooth.table[5,2]),
                          as.character(tooth.table[2,2])),
                     t7=c("Question 2.", "VC2.0", "OJ2.0",
                          as.character(tooth.table[6,2]),
                          as.character(tooth.table[3,2])))
    library(knitr)
    kable(df, digits=2)
}

## Evaluates t-test results for Dose and Delivery under unequal group variances
getPvalue <- function(test.data, testDose = TRUE) {
    pval <- NULL
    if(testDose) {
        pval <- t.test(Length ~ Dose, paired = FALSE, var.equal = FALSE,
                       data = test.data)$p.value
    } else {
        pval <- t.test(Length ~ Group, paired = FALSE, var.equal = FALSE,
                       data = test.data)$p.value
    }

    return(pval)
}

## Returns a data frame with the p_vals vector passed in as the p_values
## column and the rejectH0 column of boolean values designating whether the
## P-value was rejected (TRUE) or not (FALSE) under a Type I error rate
## alpha.
rejectBonferroni <- function(p_vals, alpha) {
    alpha.fwer <- alpha / length(p_vals)
    reject <- p_vals <= alpha.fwer
    result <- data.frame(p_values = p_vals, rejectH0 = reject)
    return(result)
}

## Ranks the p_values passed in and does a Benjamini Hochberg (BH) correction.
## A data frame with the following 4 columns is returned:
##
## test.id - test identifier, used tell which P-value corresponded to what test
## p.vals - passed in p_values in their original order
```

```r
## alpha.bh.adj - BH adjusted alpha used to reject or accept H0
## rejectH0 - boolean values designating whether the P-value was rejected
##            (TRUE) or not (FALSE) under Type I error rate alpha.
rejectBenjHoch <- function(p_values, alpha) {
    suppressMessages(suppressWarnings(library(dplyr)))
    library(dplyr)
    result <- data.frame(test.id = 1:length(p_values), p.vals = p_values,
                         alpha.bh.adj = alpha)
    result <- mutate(result, rejectH0 = (p_values < alpha))
    result <- arrange(result, p.vals)
    m <- length(p_values)
    result$alpha.bh.adj <- alpha * (1:m) / m
    result$rejectH0 = (result$p.vals <= result$alpha.bh.adj)
    result$rank <- 1:m
    result <- arrange(result, test.id)  # put back in original order

    return(result)
}

## Returns the results of the 7 t-tests and evaluates the results against
## FPR (no correction), FWER (Bonferroni), and FDR (Benjamini Hochberg)
getTestResults <- function(tooth.data = loadToothData(), alpha = 0.05) {
    t.pvals <- vector(mode = "numeric", length = 7)
    t.logic <- vector(mode = "logical", length = 7)
    # test dosages
    binary.data <- filter(tooth.data, Group == "OJ0.5" | Group == "OJ1.0")
    t.pvals[1] <- getPvalue(binary.data)
    binary.data <- filter(tooth.data, Group == "OJ1.0" | Group == "OJ2.0")
    t.pvals[2] <- getPvalue(binary.data)
    binary.data <- filter(tooth.data, Group == "VC0.5" | Group == "VC1.0")
    t.pvals[3] <- getPvalue(binary.data)
    binary.data <- filter(tooth.data, Group == "VC1.0" | Group == "VC2.0")
    t.pvals[4] <- getPvalue(binary.data)
    # test delivery
    binary.data <- filter(tooth.data, Group == "VC0.5" | Group == "OJ0.5")
    t.pvals[5] <- getPvalue(binary.data, FALSE)
    binary.data <- filter(tooth.data, Group == "VC1.0" | Group == "OJ1.0")
    t.pvals[6] <- getPvalue(binary.data, FALSE)
    binary.data <- filter(tooth.data, Group == "VC2.0" | Group == "OJ2.0")
    t.pvals[7] <- getPvalue(binary.data, FALSE)
    # rejection tests
    t.fpr <- (t.pvals < alpha)
    t.fwer.q1 <- rejectBonferroni(t.pvals[1:4], alpha)$rejectH0
    t.fwer.q2 <- rejectBonferroni(t.pvals[5:7], alpha)$rejectH0
    t.fwer <- c(t.fwer.q1, t.fwer.q2)
    t.fdr.q1 <- rejectBenjHoch(t.pvals[1:4], alpha)$rejectH0
    t.fdr.q2 <- rejectBenjHoch(t.pvals[5:7], alpha)$rejectH0
    t.fdr <- c(t.fdr.q1, t.fdr.q2)
    t.test.num <- 1:7

    return(data.frame(test.num = 1:7, P_value = t.pvals,reject_FPR = t.fpr,
                      reject_FWER = t.fwer, reject_FDR = t.fdr))
}
```