~~~
---
title: chapter9 - Hypothesis testing & Anova  in R
author: M Affouf
date: January 2, 2018
output:
  html_document:
    toc: true
    toc_depth: 5
    fig_width: 5
    fig_height: 5
---

##Agenda

- Testing differences in mean between two groups

- QQ plots

- Tests for 2 x 2 tables, j x k tables

- Plotting confidence intervals

Let's begin by loading the packages we'll need to get started
```{r}
library(MASS)
library(plyr)
library(ggplot2)
~~~

## Exploring the birthwt data

The first thing we'll do is apply the same pre-processing as on your
homework to get the data into a nicer form.

```{r}
# Rename the columns to have more descriptive names
colnames(birthwt) <- c("birthwt.below.2500", "mother.age",
"mother.weight",
    "race", "mother.smokes", "previous.prem.labor", "hypertension",
"uterine.irr",
    "physician.visits", "birthwt.grams")

# Transform variables to factors with descriptive levels
birthwt <- transform(birthwt,
            race = as.factor(mapvalues(race, c(1, 2, 3),
                              c("white","black", "other"))),
            mother.smokes = as.factor(mapvalues(mother.smokes,
                              c(0,1), c("no", "yes"))),
            hypertension = as.factor(mapvalues(hypertension,
                              c(0,1), c("no", "yes"))),
            uterine.irr = as.factor(mapvalues(uterine.irr,
                              c(0,1), c("no", "yes")))
            )
```
````
````

Over the past two lectures we created various tables and graphics to help us better understand the data. Our focus for today is to run hypothesis tests to assess whether the trends we observed last time are statistically significant.

### Testing differences in means

One of the most common statistical tasks is to compare an outcome between two groups. The example here looks at comparing birth weight between smoking and non-smoking mothers.

To start, it always helps to plot things

```{r, fig.align='center', fig.width = 5, fig.height = 4}
# Create boxplot showing how birthwt.grams varies between
# smoking status
qplot(x = mother.smokes, y = birthwt.grams,
      geom = "boxplot", data = birthwt,
      xlab = "Mother smokes",
      ylab = "Birthweight (grams)",
      fill = I("lightblue"))
```

This plot suggests that smoking is associated with lower birth weight.

**How can we assess whether this difference is statistically significant?
**

Let's compute a summary table

```{r}
aggregate(birthwt.grams ~ mother.smokes,
          data = birthwt,
          FUN = function(x) {c(mean = mean(x), sd = sd(x))})
```

Note the way that the `FUN` argument was supplied in this `aggregate()` call. This function is never given a name. It's an example of an **anonymous function**. In addition, the function returns a vector with *named elements*, giving the mean and standard deviation of the vector that's passed to it.

The standard deviation is good to have, but to assess statistical significance we really want to have the standard error (which the standard deviation adjusted by the group size).

```{r}
aggregate(birthwt.grams ~ mother.smokes, data = birthwt,
          FUN = function(x) {c(mean = mean(x),
                               se = sd(x) / sqrt(length(x)))})
```

#### t-test via t.test()

This difference is looking quite significant. To run a two-sample t-test, we can simple use the `t.test()` function.

~~~
```{r}
birthwt.t.test <- t.test(birthwt.grams ~ mother.smokes, data = birthwt)
birthwt.t.test
```

We see from this output that the difference is highly significant.  The
`t.test()` function also outputs a confidence interval for us.

Notice that the function returns a lot of information, and we can access
this information element by element
```{r}
names(birthwt.t.test)
birthwt.t.test$p.value   # p-value
birthwt.t.test$estimate  # group means
birthwt.t.test$conf.int  # confidence interval for difference
attr(birthwt.t.test$conf.int, "conf.level")  # confidence level
```

The ability to pull specific information from the output of the hypothesis
test allows you to report your results using inline code chunks.  That is,
you don't have to hardcode estimates, p-values, confidence intervals, etc.

```{r}
# Calculate difference in means between smoking and nonsmoking groups
birthwt.smoke.diff <- round(birthwt.t.test$estimate[1] -
birthwt.t.test$estimate[2], 1)

# Confidence level as a %
conf.level <- attr(birthwt.t.test$conf.int, "conf.level") * 100
```

**Example**:  Here's what happens when we knit the following paragraph.

```{r, eval = FALSE}
Our study finds that birth weights are on average `r birthwt.smoke.diff`g
higher in the non-smoking group compared to the smoking group (t-statistic
`r round(birthwt.t.test$statistic,2)`, p=`r round(birthwt.t.test$p.value,
3)`, `r conf.level`% CI [`r round(birthwt.t.test$conf.int,1)`]g)
```

**Output**: Our study finds that birth weights are on average `r
birthwt.smoke.diff`g higher in the non-smoking group compared to the
smoking group (t-statistic `r round(birthwt.t.test$statistic,2)`, p=`r
round(birthwt.t.test$p.value, 3)`, `r conf.level`%  CI [`r
round(birthwt.t.test$conf.int,1)`]g)

One other thing to know is that `t.test()` accepts input in multiple
forms.  I like using the formula form whenever it's available, as I find
it to be more easily interpretable.  Here's another way of specifying the
same information.

```{r}
with(birthwt, t.test(x=birthwt.grams[mother.smokes=="no"],
                     y=birthwt.grams[mother.smokes=="yes"]))
```
~~~

~~~~
Specifying `x` and `y` arguments to the `t.test` function runs a t-test to
check whether `x` and `y` have the same mean.

#### What is statistical significance testing doing?

Here's a little simulation where we have two groups, a treatment groups
and a control group.  We're going to simulate observations from both
groups.  We'll run the simulation two ways.

- First simulation (Null case): the treatment has no effect
- Second simulation (Non-null case):  the treatment on average increases
outcome

```{r}
set.seed(12345)
# Function to generate data
generateSimulationData <- function(n1, n2, mean.shift = 0) {
  y <- rnorm(n1 + n2) + c(rep(0, n1), rep(mean.shift, n2))
  groups <- c(rep("control", n1), rep("treatment", n2))
  data.frame(y = y, groups = groups)
}
```

Let's look at a single realization in the null setting.
```{r, fig.height = 5, fig.width = 7}
n1 = 30
n2 = 40
# Observation, null case
obs.data <- generateSimulationData(n1 = n1, n2 = n2)

# Box plots
qplot(x = groups, y = y, data = obs.data, geom = "boxplot")

# Density plots
qplot(fill = groups, x = y, data = obs.data, geom = "density",
      alpha = I(0.5),
      adjust = 1.5,
      xlim = c(-4, 6))

# t-test
t.test(y ~ groups, data = obs.data)
```

And here's what happens in a random realization in the non-null setting.
```{r, fig.height = 5, fig.width = 7}
# Non-null case, very strong treatment effect
# Observation, null case
obs.data <- generateSimulationData(n1 = n1, n2 = n2, mean.shift = 1.5)

# Box plots
qplot(x = groups, y = y, data = obs.data, geom = "boxplot")

# Density plots
qplot(fill = groups, x = y, data = obs.data, geom = "density",
      alpha = I(0.5),
      adjust = 1.5,
      xlim = c(-4, 6))
~~~~

~~~~
# t-test
t.test(y ~ groups, data = obs.data)
```

More interestingly, let's see what happens if we repeat our simulation
10000 times and look at the p-values.  We'll use a moderate effect of 0.5
instead of the really strong effect of 1.5 in this simulation.

```{r, cache = TRUE}
NUM_ITER <- 10000
pvals <- matrix(0, nrow = NUM_ITER, ncol = 2)
for(i in 1:NUM_ITER) {
  # Generate data
  obs.null <- generateSimulationData(n1 = n1, n2 = n2)
  obs.alt <- generateSimulationData(n1 = n1, n2 = n2, mean.shift = 0.5)

  # Record p-values
  pvals[i, 1] <- t.test(y ~ groups, data = obs.null)$p.value
  pvals[i, 2] <- t.test(y ~ groups, data = obs.alt)$p.value
}

pvals <- as.data.frame(pvals)
colnames(pvals) <- c("null", "nonnull")

# Plotting routine
qplot(x = null, data = pvals, xlab = "p-value",
      xlim = c(0, 1),
      main = "P-value when treatment has 0 effect")
qplot(x = nonnull, data = pvals, xlab = "p-value",
      xlim = c(0, 1),
      main = "P-value when treatment has MODERATE effect")
```

Let's show both histograms on the same plot.

```{r}
# Let's start by reshaping the data
# This approach isn't the best one, but it works well for this simple case
pvals.df <- data.frame(pvals = c(pvals$null, pvals$nonnull),
                       case = c(rep("null", NUM_ITER), rep("nonnull",
NUM_ITER)))

# Plot
ggplot(data = pvals.df, aes(x = pvals, fill = case)) +
  geom_histogram(alpha=0.75, position="identity") +
  xlim(0,1)

```

#### What if sample is small and data are non-Gaussian?

In your statistics classes you've been taught to approach the t-test with
caution.  If your data is highly skewed, you would need a very large
sample size for the t-statistic to actually be t-distributed.

When it doubt, you can run a non-parametric test.  Here's how we run a
~~~~

Mann-Whitney U test (aka Wilcoxon rank-sum test) using the `wilcox.test()` function.

```{r}
# Formula specification
birthwt.wilcox.test <- wilcox.test(birthwt.grams ~ mother.smokes,
data=birthwt, conf.int=TRUE)
birthwt.wilcox.test

# x,y specification
with(birthwt, wilcox.test(x=birthwt.grams[mother.smokes=="no"],
                     y=birthwt.grams[mother.smokes=="yes"]))
```

In general, hypothesis tests in R return an object of class `htest` which has similar attributes to what we saw in the t-test.

```{r}
class(birthwt.wilcox.test)
```

Here's a summary of the attributes:

name	description
statistic	the value of the test statistic with a name describing it.
parameter	the parameter(s) for the exact distribution of the test statistic.
p.value	the p-value for the test.
null.value	the location parameter mu.
alternative	a character string describing the alternative hypothesis
method	the type of test applied.
data.name	a character string giving the names of the data.
conf.int	a confidence interval for the location parameter. (Only present if argument conf.int = TRUE.)
estimate	an estimate of the location parameter. (Only present if argument conf.int = TRUE.)

### Is the data normal?

I would recommend using a non-parametric test when the data appears highly non-normal and the sample size is small. If you really want to stick to t-testing, it's good to know how to diagnose non-normality.

#### qq-plot

The simplest thing to look at is a normal qq plot of the data. This is obtained using the `qqnorm()` function.

```{r, fig.align='center', fig.width = 5, fig.height = 4}
# qq plot
with(birthwt, qqnorm(birthwt.grams[mother.smokes=="no"]))
# add reference line
with(birthwt, qqline(birthwt.grams[mother.smokes=="no"], col = "blue"))

# qq plot
```

~~~
with(birthwt, qqnorm(birthwt.grams[mother.smokes=="yes"]))
# add reference line
with(birthwt, qqline(birthwt.grams[mother.smokes=="yes"], col = "blue"))

# qq plot
x <- rnorm(115); qqnorm(x); qqline(x)
```

If the data are exactly normal, you expect the points to lie on a straight
line.  The data we have here are pretty close to lying on a line.

Here's what we would see if the data were right-skewed

```{r, fig.align='center', fig.width = 5, fig.height = 4}
set.seed(12345)
fake.data <- rexp(200)
qqnorm(fake.data)
qqline(fake.data, col = "blue") # add line
```

If you construct a qqplot and it looks like this, you should be carefully,
particularly if your sample size is small.

### Tests for 2x2 tables

Here's an example of a 2 x 2 table that we might want to run a test on.
This one looks at low birthweight broken down by mother's smoking status.
You can think of it as another approach to the t-test problem, this time
looking at indicators of low birth weight instead of the actual weights.

First, let's build our table using the `table()` function (we did this
back in Lecture 5)
```{r}
weight.smoke.tbl <- with(birthwt, table(birthwt.below.2500,
mother.smokes))
weight.smoke.tbl
```

We also previously calculated the odds ratio for this table, finding that
it was approximately 2.  This indicated that the odds of low birthweight
double when the mother smokes.

To test for significance, we just need to pass our 2 x 2 table into the
appropriate function.  Here's the result of using fisher's exact test by
calling `fisher.test`

```{r}
birthwt.fisher.test <- fisher.test(weight.smoke.tbl)
birthwt.fisher.test
attributes(birthwt.fisher.test)
```

As when using the t-test, we find that there is a significant association
between smoking an low birth weight.

You can also use the chi-squared test via the `chisq.test` function.  This
is the test that you may be more familiar with from your statistics class.
~~~

~~~
```{r}
chisq.test(weight.smoke.tbl)
```

You get essentially the same answer by running the chi-squared test, but
the output isn't as useful.  In particular, you're not getting an estimate
or confidence interval for the odds ratio.  This is why I prefer
`fisher.exact()` for testing 2 x 2 tables.

#### Tests for j x k tables

Here's a small data set on party affiliation broken down by gender.

```{r}
# Manually enter the data
politics <- as.table(rbind(c(762, 327, 468), c(484, 239, 477)))
dimnames(politics) <- list(gender = c("F", "M"),
                    party = c("Democrat","Independent", "Republican"))

politics # display the data
```

We may be interested in asking whether men and women have different party
affiliations.

The answer will be easier to guess at if we convert the rows to show
proportions instead of counts.  Here's one way of doing this.

```{r}
politics.prop <- apply(politics, 1, FUN = function(x) {x / sum(x)})
politics.prop
colSums(politics.prop) # Check that columns sum to 1
politics.prop <- t(politics.prop) # Transpose the table

# Fix dimnames
dimnames(politics.prop) <- list(gender = c("F", "M"),
                    party = c("Democrat","Independent", "Republican"))

# Output
politics.prop
```

By looking at the table we see that Female are more likely to be Democrats
and less likely to be Republicans.

We still want to know if this difference is significant.  To assess this
we can use the chi-squared test (on the counts table, not the proportions
table!).

```{r}
chisq.test(politics)
```

There isn't really a good one-number summary for general $j$ x $k$ tables
the way there is for 2 x 2 tables.  One thing that we may want to do at
this stage is to ignore the Independent category and just look at the 2 x
~~~

~~~
2 table showing the counts for the Democrat and Republican categories.

```{r}
politics.dem.rep <- politics[,c(1,3)]
politics.dem.rep

# Run Fisher's exact test
fisher.test(politics.dem.rep)
```

We see that women have significantly higher odds of being Democrat
compared to men.

### Plotting the table values with confidence

It may be useful to represent the data graphically.  Here's one way of
doing so with the `ggplot2` package.  Note that we plot the
**proportions** not the counts.

**1.** Convert the table into something ggplot2 can process by using
`melt()` from the `reshape` package.
```{r}
library(reshape)
politics.prop
politics.melt <- melt(politics.prop, id=c("gender","party"))
politics.melt
```

**2.** Create a ggplot2 object, and plot with `geom_barplot()`

```{r, fig.align='center', fig.height=4, fig.width=6}
ggplot(politics.melt, aes(x=party, y=value, fill=gender)) +
geom_bar(position="dodge", stat="identity")
```

This figure is a nice alternative to displaying a table.  One thing we
might want to add is a way of gauging the statistical significance of the
differences in height.  We'll do so by adding error bars.

#### Adding error bars to bar plots

Remember, ggplot wants everything you plot to be sitting nicely in a data
frame. Here's some code that will calculate the relevant values and do the
plotting.

**1.** Get the data into a form that's easy to work with.

```{r}
# Form into a long data frame
politics.count.melt <- melt(politics, id=c("gender", "party"))
# print
politics.count.melt

# Add a column of marginal counts
politics.count.melt <- transform(politics.count.melt, totals =
rowSums(politics)[gender])
# print
~~~

~~~~
politics.count.melt
```

**2.** Calculate confidence intervals.

To calculate confidence intervals for the proportions, we can use
`prop.test` or `binom.test`.  Essentially you call these functions the
numerator and denominator for the proportion.

We'll do this in a for loop.

```{r}
# define list of prop, and lower and upper endpoints
conf.ints <- list(prop = NULL, lower = NULL, upper = NULL)
for(i in 1:nrow(politics.count.melt)) {
  numerator <- politics.count.melt$value[i]
  denominator <- politics.count.melt$totals[i]
  prop.test.out <- prop.test(numerator, denominator)

  # Add estimate of proportion to list
  conf.ints[["prop"]][i] <- prop.test.out$estimate
  # Grab confidence interval
  interval <- prop.test.out$conf.int
  # Add estimate and endpoints to conf.ints list
  conf.ints[["lower"]][i] <- interval[1]
  conf.ints[["upper"]][i] <- interval[2]
}
conf.ints
```

**3.** Combine the confidence intervals into the data frame

```{r}
politics.toplot <- cbind(politics.count.melt, conf.ints)
politics.toplot
```

**4.** Use `ggplot()`, `geom_bar()` and `geom_errorbar()` to construct the
plots

```{r, fig.align='center'}
ggplot(politics.toplot, aes(x=party, y=prop, fill=gender)) +
  geom_bar(position="dodge", stat="identity") +
  geom_errorbar(aes(ymin=lower, ymax=upper),
                width=.2,                     # Width of the error bars
                position=position_dodge(0.9))
```



### ANOVA

You can think of ANOVA (analysis of variance) as a more general version of
the t-test, or a special case of linear regression in which all covariates
are factors.

Let's go back to our favourite birthwt data set from the MASS library.
~~~~

~~~
#### One-way ANOVA example

**Question: Is there a significant association between race and
birthweight?**

Here's a table showing the mean and standard error of birthweight by race.

```{r}
aggregate(birthwt.grams ~ race, data = birthwt, FUN = mean)
```

It looks like there's some association, but we don't yet know if it's
statistically significant.  Note that if we had just two racial categories
in our data, we could run a t-test.  Since we have more than 2, we need to
run a 1-way analysis of variance (ANOVA).

**Terminology**: a $k$-way ANOVA is used to assess whether the mean of an
outcome variable is constant across all combinations of $k$ factors.  The
most common examples are 1-way ANOVA (looking at a single factor), and 2-
way ANOVA (looking at two factors).

We'll use the `aov()` function.  For convenience, `aov()` allows you to
specify a formula.
```{r}
summary(aov(birthwt.grams ~ race, data = birthwt))
```

The p-value is significant at the 0.05 level, so the data suggests that
there is an association between birthweight and race.  In other words,
average birthweight varies across the three racial groups considered in
the data.
~~~