Golfballs in the Yard
An Introduction to Hypothesis Tests

Randall Pruim

Calvin College

Jan 25, 2011

What

I came up with this example the old fashioned way

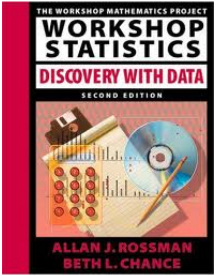
What

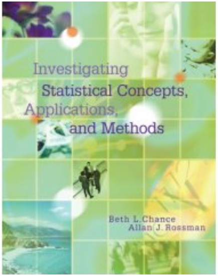
I came up with this example the old fashioned way – I stole it.

What

I came up with this example the old fashioned way – I stole it.

- I first saw this at a STATS workshop at Hope College (1999).
- The idea (and data) originated with Allan Rossman who presented this example there.
- You may know of Allan from his books, but this example is not in any of them.





The Story

Allan Rossman used to live along a golf course and collected the golf balls that landed in his yard. Most of these golf balls had a number on them.

Question: What is the distribution of these numbers?

In particular, are the numbers 1, 2, 3, and 4 equally likely?

The Story

Allan Rossman used to live along a golf course and collected the golf balls that landed in his yard. Most of these golf balls had a number on them.

Question: What is the distribution of these numbers?

In particular, are the numbers 1, 2, 3, and 4 equally likely?

Population: Golf balls driven $\sim$150 yards and sliced

The Data

Allan tallied the numbers on the first 500 golf balls that landed in his yard one summer.

1	2	3	4	other
137	138	107	104	14

Question: What is the distribution of these numbers?

In particular, are the numbers 1, 2, 3, and 4 equally likely?

The Data

Allan tallied the numbers on the first 500 golf balls that landed in his yard one summer.

1	2	3	4	other
137	138	107	104	14

Question: What is the distribution of these numbers?

In particular, are the numbers 1, 2, 3, and 4 equally likely?

Meta-Question: How do we answer this question using the data?

Step by Step

1. Tell the story; ask the questions

Step by Step

1. Tell the story; ask the questions
2. Ask students <u>what</u> they think based on the data <u>and why</u>

1	2	3	4
137	138	107	104

Step by Step

1. Tell the story; ask the questions
2. Ask students what they think based on the data and why

too many 1's and 2's $\rightarrow$

1	2	3	4
137	138	107	104

Step by Step

1. Tell the story; ask the questions
2. Ask students what they think based on the data and why

too many 1's and 2's $\rightarrow$

1	2	3	4
137	138	107	104

3. Have students suggest test statistics

Step by Step

1. Tell the story; ask the questions
2. Ask students what they think based on the data and why

too many 1's and 2's $\rightarrow$

1	2	3	4
137	138	107	104

3. Have students suggest test statistics
   - Test statistic = 1-number measure that can help us decide if our hypothesis looks good or not
   - Example: max count (138)
     - Intuition: We shouldn't get such a big number (if H_0 is true)

Step by Step

1. Tell the story; ask the questions
2. Ask students what they think based on the data and why

too many 1's and 2's →

1	2	3	4
137	138	107	104

3. Have students suggest test statistics
   - Test statistic = 1-number measure that can help us decide if our hypothesis looks good or not
   - Example: max count (138)
     - Intuition: We shouldn't get such a big number (if H_0 is true)
4. Compute test statistics (by hand) on small number of random data sets

Step by Step

1. Tell the story; ask the questions
2. Ask students what they think based on the data and why

too many 1's and 2's →

1	2	3	4
137	138	107	104

3. Have students suggest test statistics
   - Test statistic = 1-number measure that can help us decide if our hypothesis looks good or not
   - Example: max count (138)
     - Intuition: We shouldn't get such a big number (if H_0 is true)
4. Compute test statistics (by hand) on small number of random data sets
5. Compute test statistic on large number of random data sets
   - Tabulate results and display them graphically to see whether our data are "unusual"

How

I do this using R, even in classes where my students do not use R themselves.

- Allowed me to write a custom function to make trying various test statistics easy.
- Can be used for other data situations too.
- The R code for this appears in the references section of these slides and in the `fastR` package.

Could be done other ways too.

- I originally used a cgi-script written to handle only this example.
- Your favorite stat software can probably do this too.

Example – max count

Is 138 an unusually large maximum count?

```
> require(fastR)       # fastR package defines rgolfballs
> head(n=20,t(rgolfballs))     # look at first 20
```

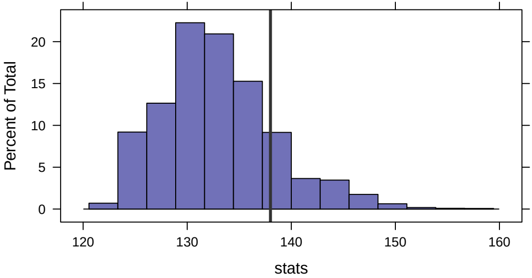
	[,1]	[,2]	[,3]	[,4]
[1,]	118	137	120	111
[2,]	125	104	137	120
[3,]	124	133	111	118
[4,]	128	125	113	120
[5,]	111	123	125	127
[6,]	125	117	127	117
[7,]	130	106	121	129
[8,]	119	120	127	120
[9,]	120	129	125	112
[10,]	114	126	115	131
[11,]	102	133	127	124
[12,]	109	140	121	116
[13,]	127	115	117	127
[14,]	120	131	108	127
[15,]	107	118	120	141
[16,]	131	116	121	118
[17,]	119	104	124	139
[18,]	115	124	115	132
[19,]	129	115	120	122
[20,]	112	136	111	127

Example – max count

```
statTally(golfballs, rgolfballs, max)    # function in fastR

Test Stat applied to sample data = 138

Of the random samples
 8101 ( 81.01 % ) had test stats < 138
 352 ( 3.52 % ) had test stats = 138
 1547 ( 15.47 % ) had test stats > 138
```

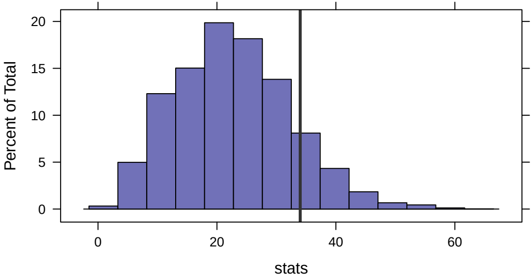


Example – range

```
statTally(golfballs, rgolfballs, function(x) {diff(range(x))})

Test Stat applied to sample data = 34

Of the random samples
 8645 ( 86.45 % ) had test stats < 34
 188 ( 1.88 % ) had test stats = 34
 1167 ( 11.67 % ) had test stats > 34
```

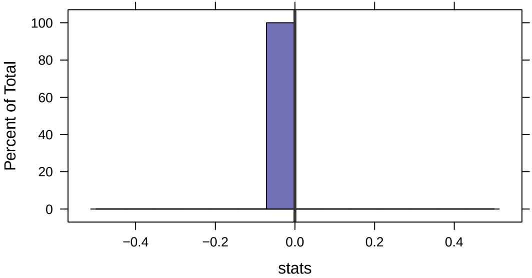


Example – World's Worst Test Statistic

```
statTally(golfballs, rgolfballs, function(x){sum(x-121.5)})

Test Stat applied to sample data = 0

Of the random samples
 0 ( 0 % ) had test stats < 0
 10000 ( 100 % ) had test stats = 0
 0 ( 0 % ) had test stats > 0
```

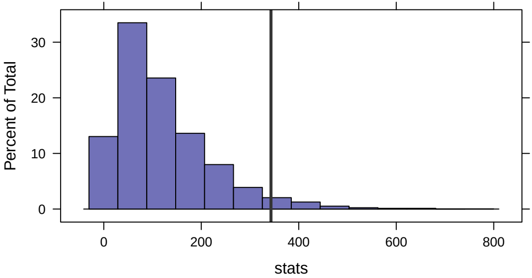


Example – variance

```
> statTally(golfballs, rgolfballs, var)

Test Stat applied to sample data = 343

Of the random samples
 9644 ( 96.44 % ) had test stats < 343
 3 ( 0.03 % ) had test stats = 343
 353 ( 3.53 % ) had test stats > 343
```



General Usage

```
statTally(sample, rdata, FUN, direction = 2, ...)
```

- `sample`: "real" data
- `rData`: matrix of simulated data (easy to do in R for many situations)
- `FUN`: a function (built-in or user defined)
  - input: data (real or simulated)
  - output: a number
- `direction`: 1 or 2
  - indicates whether samples correspond to rows (1) or columns (2)

`print( statTally(...) )` creates a histogram or stemplot

When

- Introduction to chi-squared test
  - even if I don't otherwise cover goodness of fit

- Introduction to hypothesis testing/inference
  - I have used this as my first go at inference and liked it
  - May require thinking a bit about your text, since it can be tricky to not do the book's first inference procedure first

- Introduction to empirical p-values and randomization tests
  - This could come early or late, depending on your philosophy

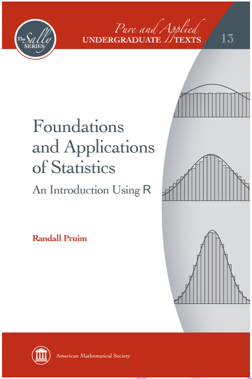
My impression is that the sooner I do this example, the better my students understand hypothesis testing and p-values.

Who

I have used this demonstration with statistics students at every level

- Intro Stats
- 200-level Stats (teachers and CS majors, primarily)
- 300-level “Math Stats” course

This example appears (twice) in my forthcoming “Math Stats” book:



Discussion Topics

Logic of a hypothesis test

1. State Hypotheses
   - Null hypothesis must provide a model (for simulations)
2. Calculate a Test Statistic
3. Determine the p-value
4. Interpret the p-value

What makes a good test statistic?

Advantages to knowing the sampling distribution

Power against particular types of alternatives

fastR

The code used in this example is in the `fastR` package available at CRAN.

- The `fastR` package is almost done; some of the documentation is still missing.
- To install this package directly within R type:

```
install.packages("fastR")
```

statTally

```
statTally <- function (
    sample, rdata, FUN,
    direction = 2,
    stemplot = dim(rdata)[direction] < 201,
    q = c(0.5, 0.9, 0.95, 0.99), ...)
{
#  all the work is in these three lines
    dstat <- FUN(sample)
    stats <- apply(rdata, direction, FUN)
    plot1 <- histogram(~stats, ...,
        panel = function(x, ...) {
            panel.histogram(x, ...)
            panel.abline(v = dstat,
                col = trellis.par.get("add.line")$col,
                lwd = 3)
        }
    )
```

statTally – continued

```
# the rest is just printing out some information
    cat("Test Stat function:\n\n")
    print(FUN)
    cat("\n")
    cat("\nTest Stat applied to sample data = ")
    cat(dstat)
    cat("\n\n")
    cat("Test Stat applied to random data:\n\n")
    print(quantile(stats, q))
    if (stemplot) {
        stem(stats)
    }
    cat("\tOf the random samples")
    cat("\n\t\t", paste(sum(stats < dstat), "(", round(100 *
        sum(stats < dstat)/length(stats), 2), "% )", "had test stats <",
        dstat))
    cat("\n\t\t", paste(sum(stats == dstat), "(", round(100 *
        sum(stats == dstat)/length(stats), 2), "% )", "had test stats =",
        dstat))
    cat("\n\t\t", paste(sum(stats > dstat), "(", round(100 *
        sum(stats > dstat)/length(stats), 2), "% )", "had test stats >",
        dstat))
    cat("\n")
    invisible(plot1)
}
```

Links

http://mosaic-web.org/

http://www.calvin.edu/~rpruim/talks/

https://r-forge.r-project.org/projects/fastr/

http://www.r-project.org/