Introduction to the New Statistics - R Workbook

Authors here

16 May 2016

Contents

1 Overview

An overview of the workbook to go here.

Info on how to install R functions etc.

Load the `itns` library. This will make all the data sets used in the book accessible.

```
library(itns)
```

2 R Basics

Some basic info for R beginners to go here.

eg how to install R, and sources of help.

This section can be skipped by anyone who already knows how to use R.

2.1 Useful Websites for Learning R

A list of useful websites, online tutorials etc for learning R to go here.

R-Studio Resources for Learning R (with links to interactive tutorials for beginners)

R Bloggers - How to Learn R

Quick-R

Cookbook for R

R Quick Reference Card

List a few particularly useful packages

dplyr

https://www3.nd.edu/~steve/computing_with_data/24_dplyr/dplyr.html

rehape and reshape2

etc

3 Picturing and Describing Data

Materials relevant to ITNS Ch 3 here.

3.1 Descriptive Statistics

List of inbuilt R functions to compute basic summary statistics.

Measure	R function
Mean	`mean()`
Median	`median()`
Minimum	`min()`
Maximum	`max()`
Range	`range()`
Interquartile Range	`IQR()`
Variance	`var()`
Standard Deviation	`sd()`
Percentiles	`quantile()`

3.2 Examples

Using the pen laptop data.

Here are the first few cases:

```r
head(pen_laptop1)
```

```
##   group transcription
## 1   Pen          12.1
## 2   Pen           6.5
## 3   Pen           8.1
## 4   Pen           7.6
## 5   Pen          12.2
## 6   Pen          10.8
```

Compute summary statistics:

```r
# Minimum
min(pen_laptop1$transcription)
```

```
## [1] 1
```

```r
# Maximum
max(pen_laptop1$transcription)
```

```
## [1] 34.7
```

```r
# Mean
mean(pen_laptop1$transcription)
```

```
## [1] 11.53385
```

```r
# Median
median(pen_laptop1$transcription)
```

```
## [1] 10.7
```

```r
# 95th percentile
quantile(pen_laptop1$transcription, probs = .95)
```

```
##   95%
## 21.34
```

```r
# From the 5th to 95th percentiles, in steps of 5
quantile(pen_laptop1$transcription, probs = seq(from = .05, to = .95, by = .05))
```

```
##    5%   10%   15%   20%   25%   30%   35%   40%   45%   50%   55%   60%
##  2.16  3.38  5.10  6.24  8.00  8.50  8.70  9.16  9.68 10.70 11.22 12.04
##   65%   70%   75%   80%   85%   90%   95%
## 12.64 13.20 15.20 17.06 17.82 18.92 21.34
```

```r
# Variance
var(pen_laptop1$transcription)
```

```
## [1] 44.7654
```

```r
# Standard deviation
sd(pen_laptop1$transcription)
```

```
## [1] 6.690695
```

```r
# Range
range(pen_laptop1$transcription)
```

```
## [1]  1.0 34.7
```

```r
# Interquartile range
IQR(pen_laptop1$transcription)
```

```
## [1] 7.2
```

3.3 Summary function

```r
summary(pen_laptop1$transcription)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00    8.00   10.70   11.53   15.20   34.70
```

3.4 Statitics by group

Using inbuilt `by()` function

```r
by(pen_laptop1$transcription, pen_laptop1$group, summary)
```

```
## pen_laptop1$group: Laptop
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.20    9.45   12.80   14.52   17.85   34.70
## --------------------------------------------------------
## pen_laptop1$group: Pen
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   5.200   8.600   8.812  11.280  20.100
```

Using the more powerful and flexible `dplyr` package

```r
# Load the dplyr package
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
group_by(pen_laptop1, group) %>%
summarize(avg = mean(transcription))
```

```
## Source: local data frame [2 x 2]
##
##    group       avg
##   (fctr)     (dbl)
## 1 Laptop 14.519355
## 2    Pen  8.811765
```

```r
# Going further - find the mean and standard deviation for each group
group_by(pen_laptop1, group) %>%
summarize(avg = mean(transcription),
          sd = sd(transcription)
          )
```

```
## Source: local data frame [2 x 3]
##
##    group       avg       sd
##   (fctr)     (dbl)    (dbl)
## 1 Laptop 14.519355 7.285576
## 2    Pen  8.811765 4.749339
```

For a two-way design, blame1 dataset

There are two independent variables - *Socioeconomic status (ses)* (high or low) and *race* (black or white).

```
blame1 %>%
  group_by(race, ses) %>%
  summarize(avg = mean(blame), std.dev = sd(blame))
```

```
## Source: local data frame [4 x 4]
## Groups: race [?]
##
##     race    ses      avg  std.dev
##   (fctr) (fctr)    (dbl)    (dbl)
## 1  black   high 2.570423 1.842544
## 2  black    low 2.966292 1.824389
## 3  white    low 2.800000 1.734270
## 4  White   high 3.181818 1.837054
```

3.5 Apply to multiple variables at a time

`summary()` function does this by default.

Example: college_survey1 data

```
summary(college_survey1)
```

```
##        id              gender        age            school_year
##  Min.   :21363617   Female:175   Min.   :18.00   First-year:43
##  1st Qu.:21442046   Male  : 68   1st Qu.:19.00   Sophomore :64
##  Median :21466849                Median :20.00   Junior    :53
##  Mean   :21456592                Mean   :21.79   Senior    :57
##  3rd Qu.:21470366                3rd Qu.:22.00   Post-bac  :26
##  Max.   :21516638                Max.   :59.00
##                                  NA's   :4
##  transfer        student_athlete student_athlete_code   wealth_sr
##  No  :209   In season     : 12    No  :212            Min.   :1.0
##  Yes : 33   non-athlete   :212    Yes : 30            1st Qu.:2.0
##  NA's:  1   Out of season: 18    NA's:  1            Median :3.0
##             NA's          :  1                        Mean   :2.9
##                                                       3rd Qu.:3.0
##                                                       Max.   :5.0
##                                                       NA's   :3
##       gpa             act        subjective_well_being positive_affect
##  Min.   :0.800   Min.   :15.00   Min.   :1.000         Min.   :1.200
##  1st Qu.:3.000   1st Qu.:21.00   1st Qu.:4.125         1st Qu.:3.000
##  Median :3.400   Median :24.00   Median :5.000         Median :3.400
```

```
##  Mean   :3.343   Mean   :24.18   Mean   :4.941         Mean   :3.433
##  3rd Qu.:3.800   3rd Qu.:27.00   3rd Qu.:6.000         3rd Qu.:3.900
##  Max.   :4.000   Max.   :36.00   Max.   :7.000         Max.   :5.000
##  NA's   :6       NA's   :28                            NA's   :8
##  negative_affect relationship_confidence    exercise
##  Min.   :1.000   Min.   :1.000           Min.   :   0.00
##  1st Qu.:1.700   1st Qu.:3.105           1st Qu.:   2.75
##  Median :2.200   Median :3.770           Median :  22.00
##  Mean   :2.322   Mean   :3.664           Mean   :  54.78
##  3rd Qu.:2.800   3rd Qu.:4.290           3rd Qu.:  61.00
##  Max.   :4.600   Max.   :5.000           Max.   :1810.00
##  NA's   :8       NA's   :11              NA's   :23
##  academic_motivation_intrinsic academic_motivation_extrinsic
##  Min.   :1.500                 Min.   :2.500
##  1st Qu.:4.000                 1st Qu.:5.170
##  Median :4.830                 Median :6.000
##  Mean   :4.862                 Mean   :5.764
##  3rd Qu.:5.830                 3rd Qu.:6.500
##  Max.   :7.000                 Max.   :7.000
##  NA's   :20                    NA's   :20
##  academic_motivation_amotivation intelligence_value  raven_score
##  Min.   :1.000                   Min.   :2.330      Min.   : 0.00
##  1st Qu.:1.000                   1st Qu.:3.330      1st Qu.:25.00
##  Median :1.000                   Median :3.670      Median :37.50
##  Mean   :1.971                   Mean   :3.619      Mean   :37.71
##  3rd Qu.:2.500                   3rd Qu.:4.000      3rd Qu.:50.00
##  Max.   :7.000                   Max.   :5.000      Max.   :87.50
##  NA's   :20                      NA's   :20         NA's   :26
```

Using dplyr package.

Split by gender, then find mean for three variables (subjective well being, positive affect, negative affect)

```
college_survey1 %>%
  group_by(gender) %>%
  summarise_each(funs(mean(., na.rm = TRUE)), subjective_well_being, positive_affect, negative_affect)
```

```
## Source: local data frame [2 x 4]
##
##   gender subjective_well_being positive_affect negative_affect
##   (fctr)                 (dbl)           (dbl)           (dbl)
## 1 Female              4.903314           3.346        2.374176
## 2   Male              5.038088           3.662        2.185692
```

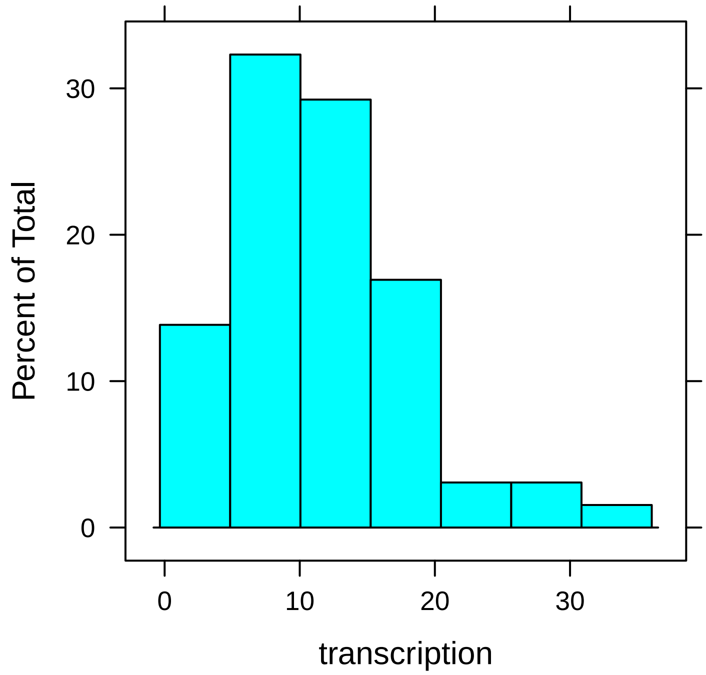
See also plyr, apply functions.

useful Links Quick R Guide to Basic Statistics in R

3.6 Histograms

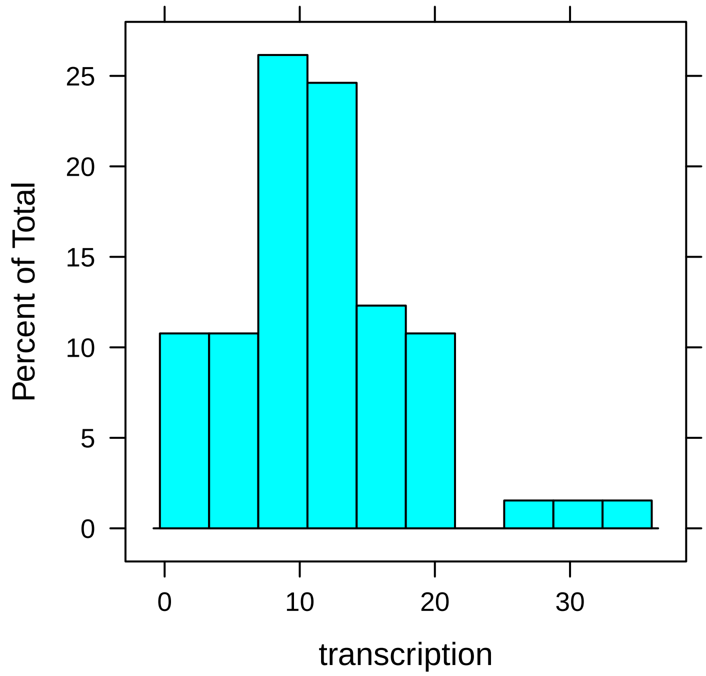
3.6.1 Single Sample

```r
library(lattice)
histogram(~transcription, data = pen_laptop1)
```



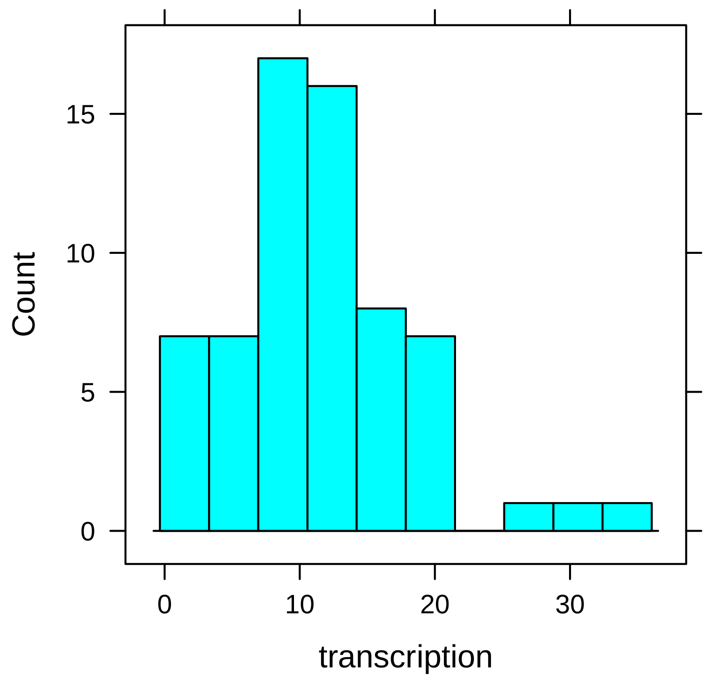
Alter the number of bins

```r
histogram(~transcription, data = pen_laptop1, nint = 10)
```



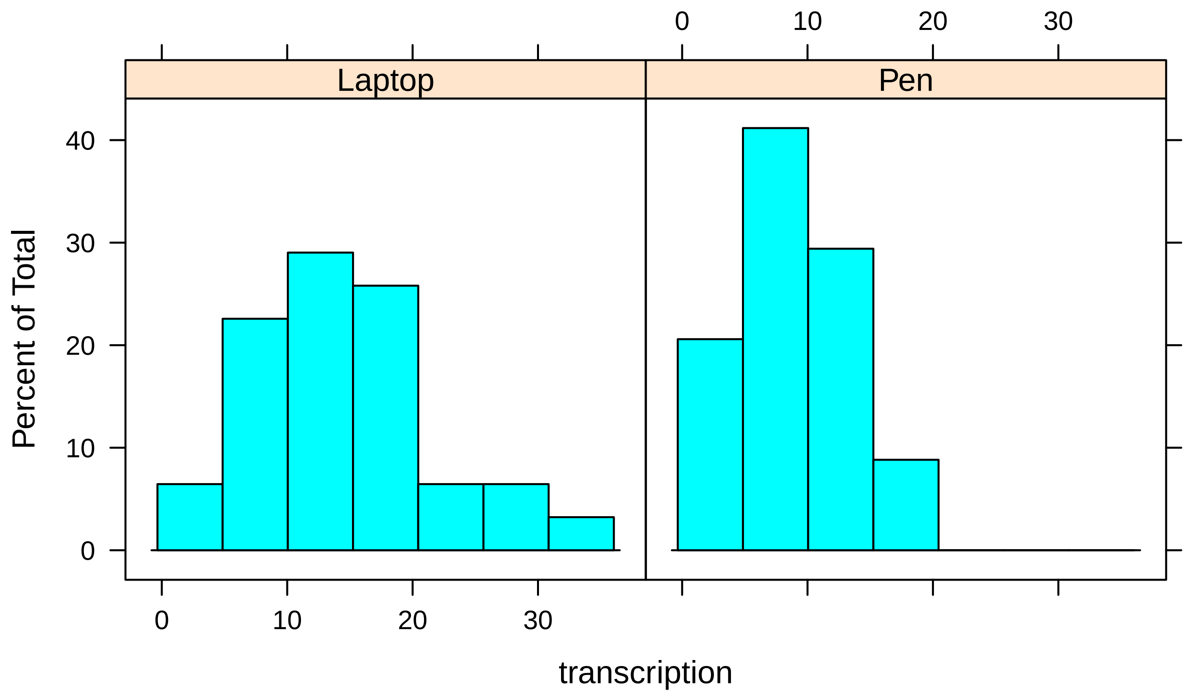
Display count rather than count on the Y-axis

```
histogram(~transcription, data = pen_laptop1, nint = 10, type = "count")
```



3.6.2 Multiple Samples

```r
histogram(~transcription | group, data = pen_laptop1)
```



3.6.3 Links

Lattice Histogram Vignette

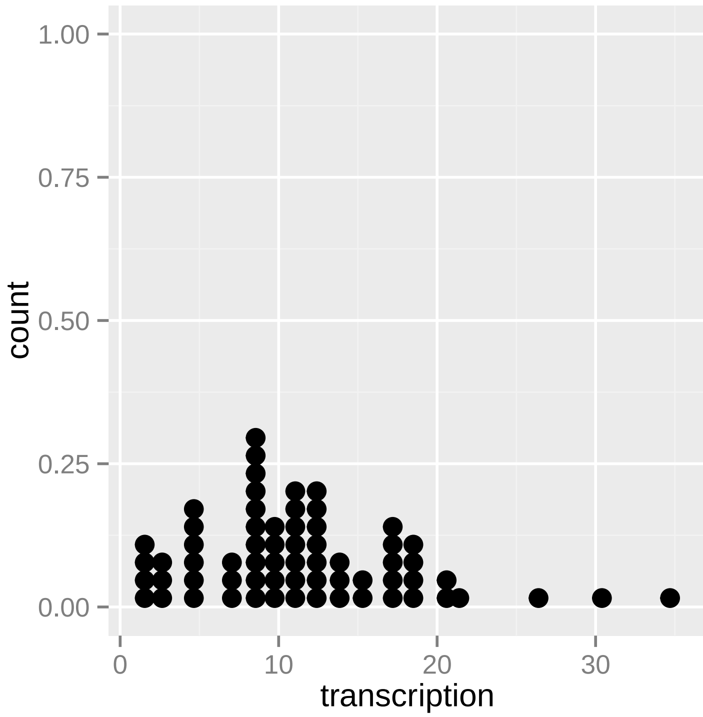
3.7 Stacked Dot Plots

3.7.1 Single group

A basic stacked dotplot generated using the `ggplot2` package.

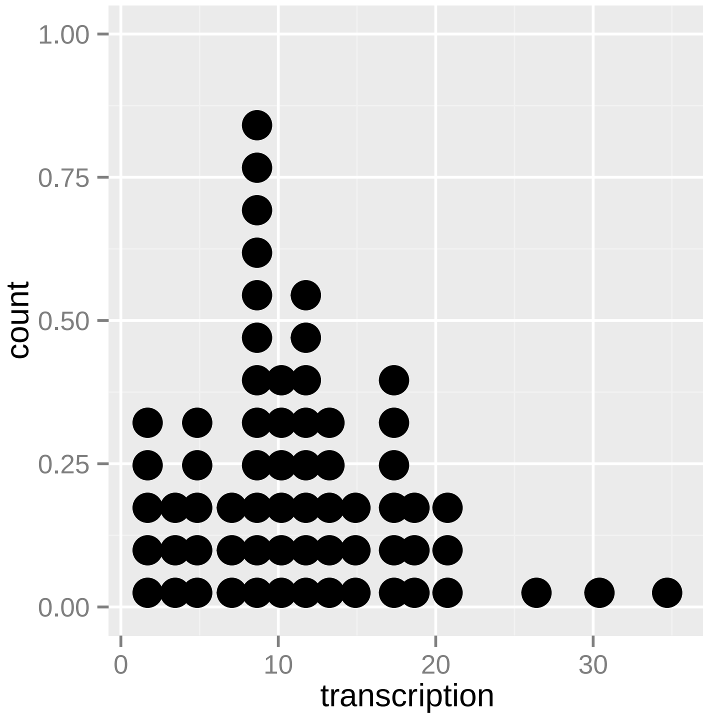
```r
library(ggplot2)
p <- ggplot(pen_laptop1, aes(x = transcription)) +
  geom_dotplot()
p
```

```
## stat_bindot: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```



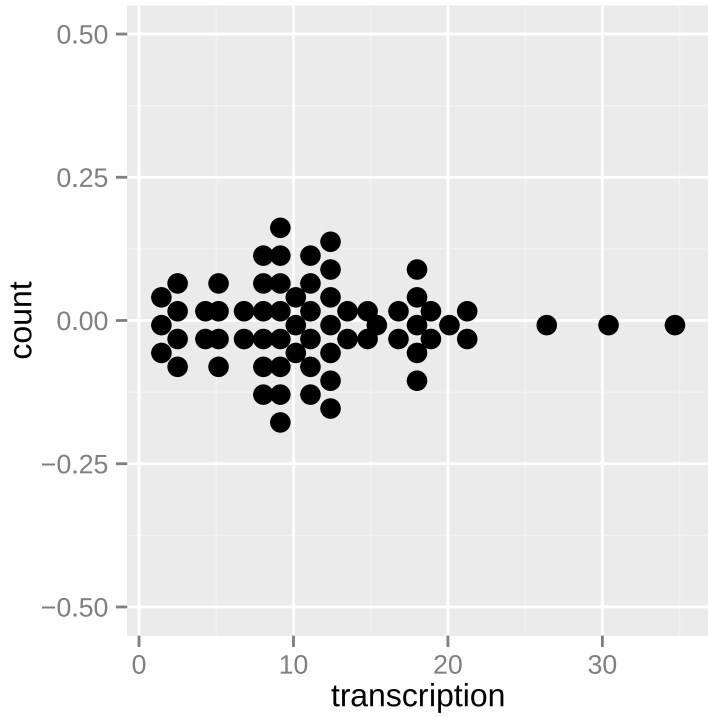
Change the bindwith, dotsize, and stack ratio

```
p <- ggplot(pen_laptop1, aes(x = transcription)) +
  geom_dotplot(binwidth = 1.5, dotsize = 1.2, stackratio = 1.5)
p
```



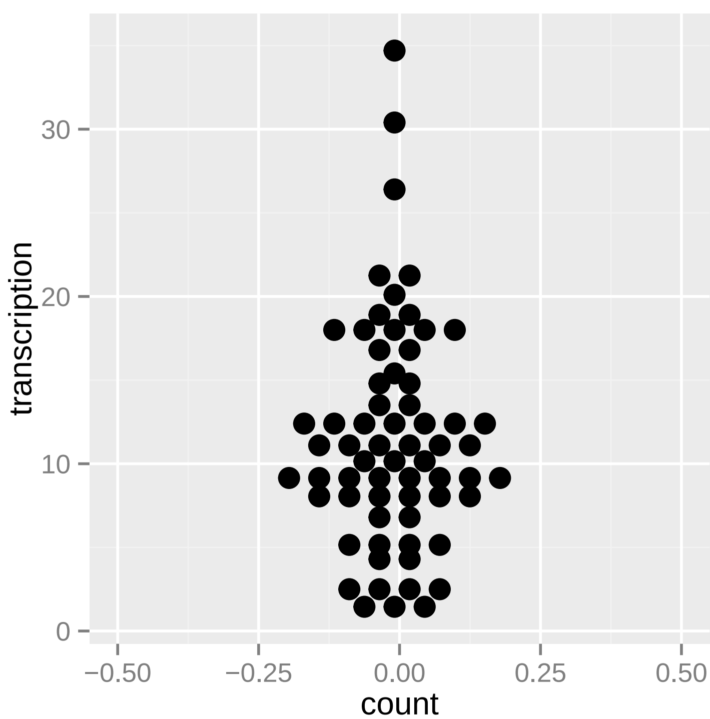
Center the dots

```
p <- ggplot(pen_laptop1, aes(x = transcription)) +
  geom_dotplot(binwidth = 1, dotsize = 1.2, stackratio = 1.5, stackdir = "center")
p
```



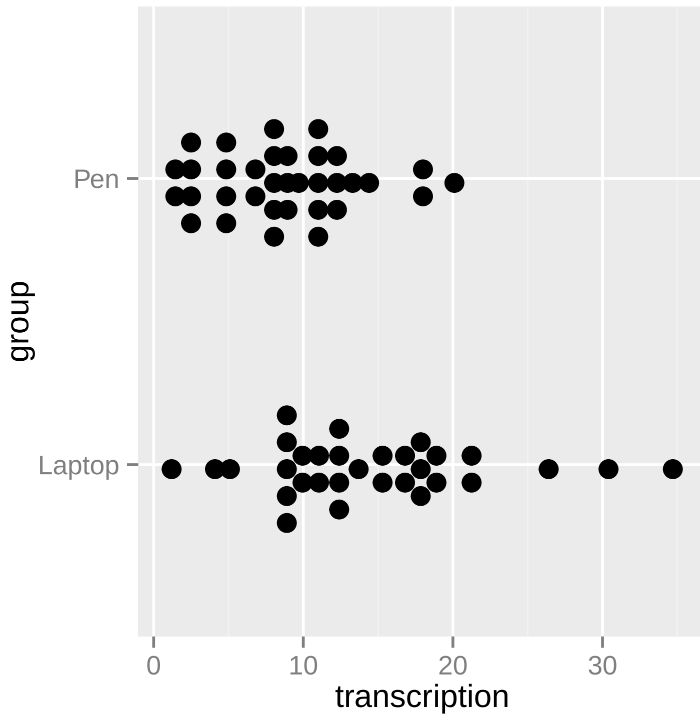
Rotate

```
p + coord_flip()
```



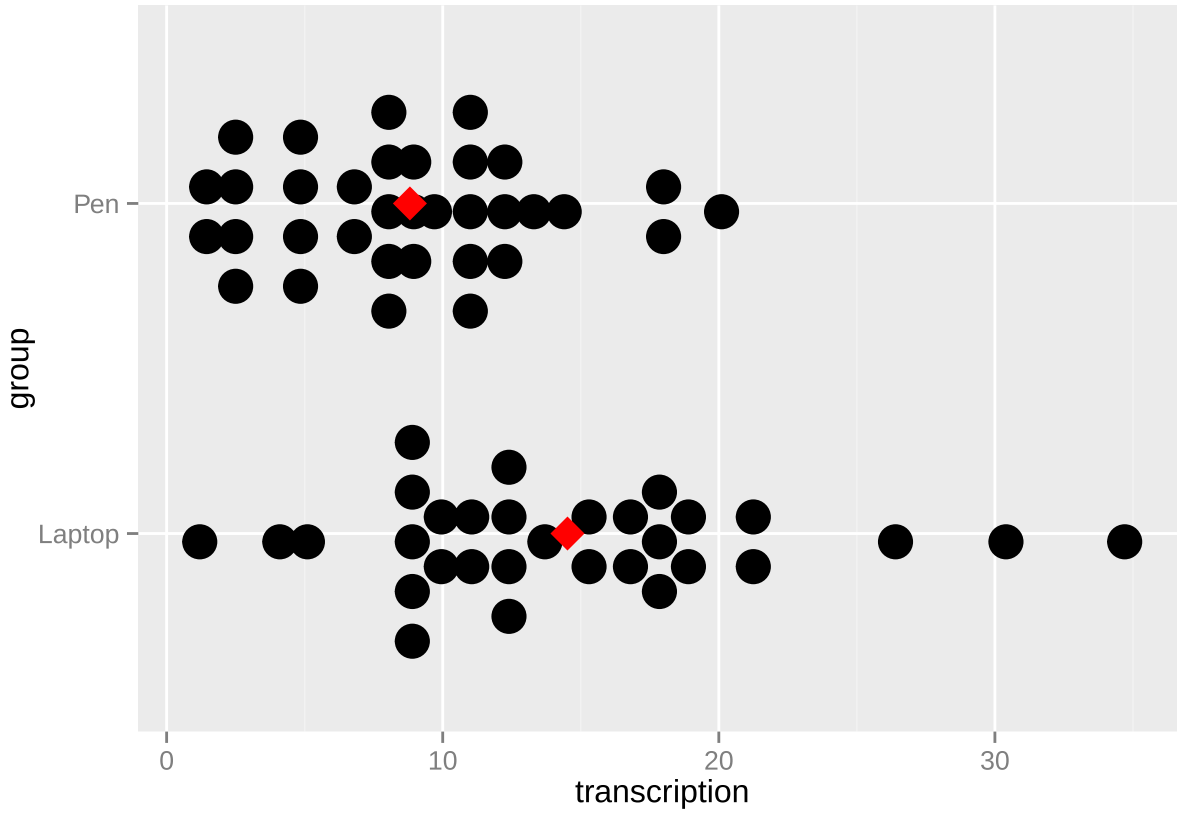
3.8 Multiple groups

```
p <- ggplot(pen_laptop1, aes(x = group, y = transcription)) +
      geom_dotplot(binaxis = "y", binwidth = 1, dotsize = 1.2, stackratio = 1.5, stackdir = "center")
p + coord_flip()
```



Add the mean of each group (as a red diamond)

```
p + stat_summary(fun.y = mean, geom = "point", shape = 18,
                 size = 6, color = "red") +
  coord_flip()
```



3.9 Clicable links to other sources of information

- ggplot2 dotplot tuturial on the Statistical tools for high-throughput data analysis website.
- Examples of how to generate dotplots using other R packages.

3.10 Other Links

- Examples of Lattice Graphics

To add: Add lattice book and vignettes

ggplot2 ggplot2 book ggplot2 website r bloggers ggplot2 cookbook

4 Z-score and T-scores

4.1 Compute z-scores

4.1.1 `scale()`

The inbuilt `scale()` function converts raw scores to Z-scores. The sample mean is subtracted from each score, and the resulting values are divided by the sample standard deviation.

```r
# For the pen_laptop dataset, convert all raw transcription scores to Z-scores
scale(pen_laptop1$transcription)
```

```
##              [,1]
##  [1,]  0.08461809
##  [2,] -0.75236522
##  [3,] -0.51322713
##  [4,] -0.58795778
##  [5,]  0.09956422
##  [6,] -0.10968160
##  [7,] -1.57440239
##  [8,] -1.29042591
##  [9,]  0.42837909
## [10,] -0.46838874
## [11,]  0.92160140
## [12,]  1.28030853
## [13,] -1.40999496
## [14,] -0.06484321
## [15,] -0.04989708
## [16,] -0.12462773
## [17,] -1.43988722
## [18,] -0.94666491
## [19,] -0.27408904
## [20,] -0.94666491
## [21,] -1.36515657
## [22,] -0.66268843
## [23,] -0.42355034
## [24,] -0.52817326
## [25,] -0.03495095
## [26,] -0.45344261
## [27,] -0.36376582
## [28,] -1.05128783
## [29,] -0.34881969
## [30,]  0.26397166
## [31,]  1.01127818
## [32,] -1.30537204
## [33,] -0.96161104
## [34,]  0.12945648
## [35,]  0.32375618
## [36,]  1.42976984
## [37,]  0.54794814
## [38,]  2.81975997
## [39,]  0.18924101
## [40,] -0.28903517
## [41,] -0.33387356
## [42,]  0.92160140
## [43,]  0.57784040
## [44,] -0.42355034
## [45,]  0.18924101
## [46,] -0.13957386
## [47,] -0.96161104
## [48,]  0.77214010
## [49,]  0.92160140
```

```
## [50,] -0.42355034
## [51,]  2.22191475
## [52,]  0.96643979
## [53,]  1.11590110
## [54,]  0.80203236
## [55,]  1.08600884
## [56,] -0.45344261
## [57,] -1.54451013
## [58,] -0.00505869
## [59,]  1.47460823
## [60,] -0.18441226
## [61,] -0.37871195
## [62,]  0.18924101
## [63,]  0.06967196
## [64,]  3.46244359
## [65,] -1.11107235
## attr(,"scaled:center")
## [1] 11.53385
## attr(,"scaled:scale")
## [1] 6.690695
```

4.2 Find a tail probability from a raw score

Use IQ example from Ch 4. IQ tests have a population mean of 100 and SD of 15. Therefore a person with a score of 115 would have a z score of 1, and score higher than about 84% of the population.

Use the inbuilt `pnorm()` function to find that probability.

```r
# Expressed on a 0 - 1 scale
pnorm(q = 115, mean = 100, sd = 15)
```

```
## [1] 0.8413447
```

```r
# Expressed on 0 - 100 scale
pnorm(q = 115, mean = 100, sd = 15)* 100
```

```
## [1] 84.13447
```

4.3 Find a tail probabilities from z-scores

```r
# Z-score of 1
pnorm(q = 1)* 100
```

```
## [1] 84.13447
```

```r
# Z-scores of -1.96 and 1.96
pnorm(q = c(-1.96, 1.96))* 100
```

```
## [1]  2.49979 97.50021
```

There is no need to specify the mean and standard deviation as the defaults are 0 and 1.

4.4 Find Z-scores given tail probabilities

```
# Find Z-scores corresponding to area under the curve

qnorm(p = .025) # lower tail
```

```
## [1] -1.959964
```

```
qnorm(p = .025, lower.tail = FALSE) # upper tail
```

```
## [1] 1.959964
```

4.5 Find raw scores given tail probabilities

```
# Find raw scores corresponding to area under the curve

qnorm(p = .025, mean = 100, sd = 15) # lower tail
```

```
## [1] 70.60054
```

```
qnorm(p = .025, mean = 100, sd = 15, lower.tail = FALSE) # upper tail
```

```
## [1] 129.3995
```

4.6 Find tail areas for chosen t

e.g., t-value of 2.145 and df = 14

```
pt(q = 2.145, df = 14) # upper tail
```

```
## [1] 0.9750099
```

```
pt(q = -2.145, df = 14) # lower tail
```

```
## [1] 0.02499008
```

```
pt(q = 2.145, df = 14, lower.tail = FALSE) # another to find lower tail probability
```

```
## [1] 0.02499008
```

4.7 Find critical t-values given df and tail area

```r
qt(p = .025, df = 14) # lower tail critical value
```

```
## [1] -2.144787
```

```r
qt(p = .025, df = 14, lower.tail = FALSE) # upper tail critical value
```

```
## [1] 2.144787
```

```r
qt(p = .975, df = 14) # alternte way to find upper tail critical t value
```

```
## [1] 2.144787
```

4.8 Calculate p-values given z-scores or t-scores

Tutorial here

5 Effect sizes and Confidence Intervals for Means and Mean Differences

5.1 Confidence Interval for the Mean of a Single Sample

Using t-distribution. There is no inbuilt function for Z as far as I know.

By default a 95% confidence interval for the mean is returned, as well as the estimated mean, t-value, degrees of freedom and p-valu.

```r
t.test(pen_laptop1$transcription)
```

```
##
##  One Sample t-test
##
## data:  pen_laptop1$transcription
## t = 13.898, df = 64, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##   9.875973 13.191719
## sample estimates:
## mean of x
##  11.53385
```

To use a different confidence level, use the `conf.level` argument. For example, to compute a 99% confidence interval you would use

```r
t.test(pen_laptop1$transcription, conf.level = .99)
```

```
##
##  One Sample t-test
##
## data:  pen_laptop1$transcription
## t = 13.898, df = 64, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 99 percent confidence interval:
##   9.330639 13.737053
## sample estimates:
## mean of x
##  11.53385
```

5.2 Two Independent Groups

5.2.1 Mean Difference and Confidence Interval

5.2.1.1 Assuming unequal variances

This is the default in R.

```r
t.test(transcription ~ group, data = pen_laptop1)
```

```
##
##  Welch Two Sample t-test
##
## data:  transcription by group
## t = 3.7031, df = 50.816, p-value = 0.0005254
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  2.612991 8.802189
## sample estimates:
## mean in group Laptop    mean in group Pen
##            14.519355             8.811765
```

5.2.1.2 Assuming equal variances

```r
t.test(transcription ~ group, data = pen_laptop1, var.equal = TRUE)
```

```
##
##  Two Sample t-test
##
## data:  transcription by group
## t = 3.7738, df = 63, p-value = 0.0003579
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  2.685265 8.729915
## sample estimates:
## mean in group Laptop    mean in group Pen
##            14.519355             8.811765
```

5.2.2 Standaridzed Mean Difference

Use the `MBESS` package.

Load the package, and then extract the data to be used in the analysis.

```
library(dplyr)
library(MBESS)

# Extract transcription scores for the 'Laptop' group
x <- pen_laptop1 %>%  filter (group == "Laptop")

# Extract transcription scores for the 'Pen' group
y <- pen_laptop1 %>% filter (group == "Pen")

# Find the sample size for each group, excluding any missing data
nx <- na.omit(length(x$transcription))
ny <- na.omit(length(y$transcription))

## Find the standardized mean difference (biased)
d <- smd(x$transcription, y$transcription)
d
```

```
## [1] 0.9371681
```

```
## Find unbiased standardized mean difference
du <- smd(x$transcription, y$transcription, Unbiased = TRUE)
du
```

```
## [1] 0.9259595
```

```
# Compute a confidence interval
ci.smd (smd = d, n.1 = nx, n.2 = ny)
```

```
## $Lower.Conf.Limit.smd
## [1] 0.4204238
##
## $smd
## [1] 0.9371681
##
## $Upper.Conf.Limit.smd
## [1] 1.447208
```

The `smd` and 'ci.smd‘ functions used the pooled standard deviation to compute **d**.

To use the standard deviation of one of the groups instead (e.g., when the homogeneity of variance assumption is not met), use the alternative functions `smd.c` and 'ci.smd.c‘

For example, to use the laptop group standard deviation as the standardizer:

```
# Biased d, using the Laptop SD as the standardizer
d <- smd.c(Group.T = y$transcription, # Pen raw data
      Group.C = x$transcription # Laptop raw data
      )
d
```

```
## [1] -0.7834096
```

```r
# Confidence Interval
ci.smd.c (smd.c = d, n.C = nx, n.E = ny)
```

```
## $Lower.Conf.Limit.smd.c
## [1] -1.303023
##
## $smd.c
## [1] -0.7834096
##
## $Upper.Conf.Limit.smd.c
## [1] -0.2525728
```

5.3 Trimmed mean difference

Use WRS2 package.

5.4 Two Dependent Groups

5.4.1 Unstandardized Mean Change

Use thomason1 dataset.

```r
t.test(thomason1$pre, thomason1$post, paired = TRUE)
```

```
##
##  Paired t-test
##
## data:  thomason1$pre and thomason1$post
## t = -3.8555, df = 11, p-value = 0.002674
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.618115 -0.715218
## sample estimates:
## mean of the differences
##               -1.666667
```

5.4.2 Standardized Mean Change

There is no MBESS function for this.

Need to write a function or show calculations.

5.5 One and two-way designs

One-way independent groups

one way dependent groups

Two-way independent groups

Between by within

Contrasts

6 Correlation

6.1 Figures

scatterplot and other plots in R

corplot package

6.1.1 Correlations and Confidence interval

```
cor(thomason1)
```

```
##            pre      post
## pre  1.0000000 0.8923908
## post 0.8923908 1.0000000
```

```
cor.test(thomason1$pre, thomason1$post)
```

```
##
##  Pearson's product-moment correlation
##
## data:  thomason1$pre and thomason1$post
## t = 6.2535, df = 10, p-value = 9.462e-05
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.6528351 0.9696774
## sample estimates:
##       cor
## 0.8923908
```

6.1.2 Difference in correlations

here. Have to look up.

7 Regression

Using home_prices data.

```
mod <- lm(price ~ size, data = home_prices)
mod
```

```
##
## Call:
## lm(formula = price ~ size, data = home_prices)
##
## Coefficients:
## (Intercept)         size
##      -66024         2713
```

```r
summary(mod)
```

```
##
## Call:
## lm(formula = price ~ size, data = home_prices)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -627083  -90490  -34039   49459  955266
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -66024.3    23426.5  -2.818  0.00515 **
## size          2713.4      125.4  21.646  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 179100 on 298 degrees of freedom
## Multiple R-squared:  0.6112, Adjusted R-squared:  0.6099
## F-statistic: 468.6 on 1 and 298 DF,  p-value: < 2.2e-16
```

```r
confint(mod)
```

```
##                   2.5 %     97.5 %
## (Intercept) -112126.598 -19921.980
## size           2466.688   2960.059
```

```r
anova(mod)
```

```
## Analysis of Variance Table
##
## Response: price
##            Df     Sum Sq    Mean Sq F value    Pr(>F)
## size        1 1.5036e+13 1.5036e+13  468.56 < 2.2e-16 ***
## Residuals 298 9.5626e+12 3.2089e+10
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Use car package when type III sums of squares are needed.

```r
library(car)
mod2 <- lm(price ~ size + location, data = home_prices)
anova(mod2) # default R method
```

```
## Analysis of Variance Table
##
## Response: price
##            Df     Sum Sq    Mean Sq  F value    Pr(>F)
## size        1 1.5036e+13 1.5036e+13 1020.909 < 2.2e-16 ***
## location   26 5.5567e+12 2.1372e+11   14.511 < 2.2e-16 ***
## Residuals 272 4.0059e+12 1.4728e+10
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
Anova(mod2, type = "III") # type II SOS, same as SPSS
```

```
## Anova Table (Type III tests)
##
## Response: price
##                 Sum Sq  Df F value    Pr(>F)
## (Intercept) 1.3598e+11   1   9.233  0.002608 **
## size        9.5428e+12   1 647.954 < 2.2e-16 ***
## location    5.5567e+12  26  14.511 < 2.2e-16 ***
## Residuals   4.0059e+12 272
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Diagnostic plots.

Confidence intervals for the Mean of Y, at every X.

8 Frequencies, Proportions, Risks

PropCIs package does most of what we want

chisquare

9 Precision and Planning

MBESS has functions for this. Functions start with ss prefix. eg ss.aipe.smd

10 To do

10.1 Cat's Eye Picture

10.2 MOE for means ?