R at a Glance:
Useful functions and syntax for Psych 252

Getting started
Install library: install.packages('mylibrary')
Load library: library(mylibrary)
Useful libraries to start with:

<table>
<thead>
<tr>
<th>Load data</th>
<th>Clean up data</th>
<th>Statistics</th>
<th>Share results</th>
</tr>
</thead>
<tbody>
<tr>
<td>xlsx</td>
<td>plyr</td>
<td>car</td>
<td>ggplot2</td>
</tr>
<tr>
<td>R.Matlab</td>
<td>dplyr</td>
<td>nlm</td>
<td>rmarkdown</td>
</tr>
<tr>
<td>jsonlite</td>
<td>tidyr</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Set working directory: setwd '~/path/to/my/data'
Get current working directory: getwd()

Search for a function in the R documentation:
By part of its name: ??Fun
By exact name: ?functionName

Data frames
Combine elements into a vector: c(1,2,3,4)
Create number sequence: seq(start, end)
Combine vectors by rows: rbind(vectr1, vectr2)
Combine vectors by columns: cbind(vectr1, vectr2)
Create data frame from vectors: data.frame(tag = value)
Load an existing data set: read.csv('mydata.csv')
Load data: read.xlsx('mydata.xls')

Exploring datasets
View topmost rows: head(data)
View structure of data: str(data)
View summary of data: summary(data)

Data types
(In the examples below, fill in datatype with the type of data you want, such as factor, character, numeric, or logical.)
Converting to other data types: as.dataype(vectr)
Check data type of a vector: is.datatype(vectr)
View/set levels of a factor: levels(data$columnName)

Selecting and extracting data
View all column names: names(data)
Get column by name: data$columnName
Get i-th row: data[i, ]
Get j-th column: data[, j]
Get element at row i, column j: data[i, j]
Select rows using logical: data[data$group == 'A', ]

Notes:
[1] Make sure you are in the correct working directory!
[2] summary returns different results depending on the object you want to summarize. Try it out with data frames, models, statistical tests, etc., to see what information it gives you.
[3] Don't forget the comma! This will return all rows that match the condition in the brackets. You can use this syntax to subset your data frame by any criterion, such as subjects that are above a certain age, all measurements taken in a drug trial before treatment, etc.

Basic math & statistics

<table>
<thead>
<tr>
<th>Name</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Variable</td>
<td>x &lt;- 10</td>
</tr>
<tr>
<td>Assignment</td>
<td>x = 10</td>
</tr>
<tr>
<td>Addition</td>
<td>x + y</td>
</tr>
<tr>
<td>Subtraction</td>
<td>x - y</td>
</tr>
<tr>
<td>Multiplication</td>
<td>x * y</td>
</tr>
<tr>
<td>Exponent</td>
<td>x ** y</td>
</tr>
<tr>
<td>Modulus</td>
<td>x %% y</td>
</tr>
<tr>
<td>Less than</td>
<td>x &lt; y</td>
</tr>
<tr>
<td>Less or equal to</td>
<td>x &lt;= y</td>
</tr>
<tr>
<td>Greater than</td>
<td>x &gt; y</td>
</tr>
<tr>
<td>Greater or equal to</td>
<td>x &gt;= y</td>
</tr>
<tr>
<td>Equal to</td>
<td>x == y</td>
</tr>
<tr>
<td>Not equal to</td>
<td>x != y</td>
</tr>
<tr>
<td>NOT x</td>
<td>!x</td>
</tr>
<tr>
<td>x OR y</td>
<td>x</td>
</tr>
<tr>
<td>x AND y</td>
<td>x &amp; y</td>
</tr>
<tr>
<td>x IN y</td>
<td>x %in% y</td>
</tr>
<tr>
<td>Exponential</td>
<td>exp(x)</td>
</tr>
<tr>
<td>Logarithm</td>
<td>log(x)</td>
</tr>
<tr>
<td>Square root</td>
<td>sqrt(x)</td>
</tr>
<tr>
<td>Round</td>
<td>round(x)</td>
</tr>
<tr>
<td>Absolute value</td>
<td>abs(x)</td>
</tr>
<tr>
<td>Sum</td>
<td>sum(vectr)</td>
</tr>
<tr>
<td>Scale &amp; center</td>
<td>scale(vectr)</td>
</tr>
<tr>
<td>Length of vector</td>
<td>Length(vectr)</td>
</tr>
<tr>
<td>Maximum</td>
<td>max(vectr)</td>
</tr>
<tr>
<td>Minimum</td>
<td>min(vectr)</td>
</tr>
<tr>
<td>Mean</td>
<td>mean(vectr)</td>
</tr>
<tr>
<td>Median</td>
<td>median(vectr)</td>
</tr>
<tr>
<td>Std. dev.</td>
<td>sd(vectr)</td>
</tr>
<tr>
<td>Variance</td>
<td>var(vectr)</td>
</tr>
<tr>
<td>Correlation</td>
<td>cor(vectr1, vectr2)</td>
</tr>
<tr>
<td>Covariance</td>
<td>cov(vectr1, vectr2)</td>
</tr>
<tr>
<td>T-Test</td>
<td>t.test(y ~ x ,data)</td>
</tr>
<tr>
<td>Chi-squared test</td>
<td>chisq.test(table)</td>
</tr>
<tr>
<td>ANOVA</td>
<td>aov(y ~ x, data = d)</td>
</tr>
<tr>
<td></td>
<td>lm(y ~ x, data)</td>
</tr>
</tbody>
</table>
More statistics

Sampling from distributions
For the commands below, use `norm` to sample from the normal distribution with mean 0 and s.d. 1, or substitute `norm` with the name of another distribution.

View all available distributions
Get probability of quantile \( x^{[1]} \)
Get quantile with probability \( p^{[1]} \)
Get \( n \) samples from distribution

[1] By default, `pnorm(x)` will return \( P(X \leq x) \), the probability of drawing values that are less than or equal to \( x \), and `qnorm(p)` will return some \( x \) that satisfies \( P(X \leq x) = p \). To instead compute \( P(X > x) \), use `pnorm(x, lower.tail = FALSE)`.

Modeling datasets
Simple linear model
Logistic regression
Mixed-effects model
Summary of model
Compare model fits

Plotting with ggplot2

Making any plot with ggplot follows the same basic steps:
1. Choosing a `dataset` to plot
2. Using `geoms` to specify what kinds of marks (such as lines, dots, or bars) will appear on the plot
3. Using aesthetic mappings to specify how different properties of the dataset will appear on the plot. The most basic of these is choosing which variables will appear on the x and y axis.
4. Changing the look of the plot with custom settings.

Basic syntax
Use + to add elements, layers, and custom options.

```r
ggplot(data, aes(x=IV, y=DV, color=cond)) +
  geom_point() +
  geom_smooth(method = ‘lm’) +
  xlab(‘Time’) +
  ylab(‘Score’)"
```

[1] `color` color-codes lines and points according to the factor of your choice (here, ‘cond’). `fill` color-codes bars in bar graphs.
[2] Each geom has custom options available that can be specified as arguments to the geom function. Check the documentation!

Geoms
In the examples below:
```r
myplot <- ggplot(data, aes(x = IV, y = DV))

# Histogram
ggplot(data, aes(x=age)) +
  geom_histogram(binwidth=5)

# Bar graph
myplot + geom_bar(stat=‘identity’)

# Scatter plot
myplot + geom_jitter()

# Line graph
myplot + geom_line()

# Error bars
myplot + geom_errorbar(aes(ymin = lower, ymax = upper), width = 0.1)
```

Customization
Adding a title
Label x-axis
Label y-axis
Faceting
```r
ggtitle(‘My Plot’) +
xlab(‘Condition’) +
ylab(‘Response’)"
```
```r
facet_grid(gender ~ .) +
facets( ~ gender)
```
```r
facet_grid(age ~ gender) +
facet_wrap(~ gender)
```

Programming basics

Comments
```r
# this is a comment! the computer will ignore it, but the humans reading your code will appreciate it.
```

Defining functions
```r
myfunction <- function(input1, input2, …) {
  statements
  return(output)
}
```

Control statements
```r
if (chk1) {
  # run this if chk1 is true
}
else (chk2) {
  # run this if chk2, but not chk1, is true
} else {
  # if all else fails, run this
}
```

For loops:
```r
for (item in sequence) {
  # carry out these same instructions for each item in the vector sequence
}
```

Useful commands
```r
Combine strings of text
Attach’ data frame to environment
‘Detach’ data frame
Evaluate expression using contents of data frame
```
```r
paste(str1, str2)
attach(mydata)
detach(mydata)
with(mydata, expr, …)
```