One factor Repeated Measures ANOVA with R

A template document for traditional and current approaches.

Bruce Dudek

2023-04-25

Contents

Preface

This document can be a standalone "how-to" document for R users. However, it is primarily intended for students in the APSY510/511 statistics sequence at the University at Albany. It is a fairly thorough treatment of graphical and inferential evaluation of one-factor designs. It presumes prior background coverage of the repeated measures ANOVA logic from standard textbooks such as Howell (2013), Keppel (2004), or Maxwell, Delaney and Kelley (2017). The analyses are intended to parallel and exhaust the methods already covered with SPSS, and to extend them to many additional topics.

This book/monograph uses the **bookdown** package (Xie, 2020a) for R (R Core Team, 2020), which was built on top of **rmarkdown** (Allaire et al., 2020) and **knitr** (Xie, 2015). RStudio (RStudio Team, 2015) was used for all writing and R programming.

Chapter 1

Background and R Setup

The goal of this document is provision of a template for using R to evaluate data from a 1-factor repeated measures design that is often called a within-subjects problem. Rather than providing one data point for a DV measurement as was the case for the "between-groups" design, each case provides more than one measurement since they are measured "repeatedly".

The standard R axiom that there are always multiple ways of performing any task is never more accurate than with the ANOVA models. Beginning with graphical depiction and extending to standard NHST inferences, contrast analysis and post hoc tests, and evaluation of assumptions, etc., we can add to that list major divisions in approaches to repeated measures analysis, and this document could become very very long.

This document

- Is intended for use by APSY511 course at the University at Albany, but can be more broadly used by data analysts.
- Is a fairly full one-factor repeated measures anova exposition for a five category design.
- Implements graphical summaries and numerical descriptions in an EDA section.
- Approaches ANOVA as linear modeling and is supplemented with analytical contrasts, and multiple comparison tests.
- Includes a section on the Multivariate approach to the repeated measures problem.
- Provides templates for both the traditional Univariate/GLM approach as well as linear mixed models approaches.
- Includes graphical and inferential evaluation of assumptions.
- Provides brief illustrations of Bayes Factor, resampling, and robust methods, as well as a non-parametric approach.

One primary philosophy drives much of this document: repeated measures ANOVA is not dead. It can be very useful in experimental design situations where there are no missing data. Otherwise, linear mixed effects models have an advantage. The traditional methods are much criticized on the basis of flawed error terms when non-sphericity is present. Adjustment method such as Greenhouse-Geisser and Huynh-feldt are looked at with disdain in some quarters. However, an overarching perspective on ANOVA can argue that omnibus effects are the least interesting parts of an analysis. Follow up analyses employing contrasts (and in factorial designs, contrasts on main effects, interactions and simple main effects) are valuable tools. With the implementation of specific error terms the tests of those contrasts are not subject to the non-sphericity consequences (see the contrasts sections below for citations). Since the GG and HF methods seem to be looked down upon by mixed effects modelers, it becomes a non-issue if the focus is on contrast analysis, perhaps instantiated with orthogonal sets. This recommendation is also informed by an understanding that mixed effects modeling of contrasts is a somewhat fuzzy area where clear additional understanding is required. That said, the document also contains some rudimentary linear mixed modeling approaches.

The document is always under development.

One of the primary goals is to reproduce all the work we have accomplished with the SPSS GLM, and MANOVA procedures (and then some).

Several R packages are required:

```
#if (!requireNamespace("BiocManager", quietly = TRUE))
#    install.packages("BiocManager")
#BiocManager::install("Biobase", version = "3.8")
```

```
knitr::opts_chunk$set(echo = TRUE, warning=FALSE)
# load packages and import data
library(afex)
library(BayesFactor)
library(car)
library(emmeans)
library(ez)
library(foreign)
library(ggplot2)
library(ggthemes)
library(granova)
library(gt)
library(kableExtra)
library(knitr)
library(lme4)
library(multcomp)
library(nlme)
library(nortest)
```

```
library(permuco)
library(phia)
library(plyr)
library(psych)
library(rmarkdown)
library(Rmisc)
library(sciplot)
library(sjstats)
library(tidyr)
library(WRS2)
```

Package citations for packages loaded here (in the above order): **afex** (Singmann et al., 2020), **BayesFactor** (Morey and Rouder, 2018),
car (Fox et al., 2020), **emmeans** (Lenth, 2020), **ez** (Lawrence, 2016), **ggplot2** (Wickham et al., 2020), **ggthemes** (Pruzek and Helmreich, 2014), **granova** (Pruzek and Helmreich, 2014), **gt** (Iannone et al., 2019), **kableExtra** (Zhu, 2019), **knitr** (Xie, 2020b), **lme4** (Bates et al., 2019), **multcomp** (Hothorn et al., 2020), **nlme** (Pinheiro et al., 2020), **nortest** (Gross and Ligges, 2015), **permuco** (Frossard and Renaud, 2019), **phia** (De Rosario-Martinez, 2015), **psych** (Revelle, 2020), **rmarkdown**(Allaire et al., 2020), **Rmisc**(Hope, 2013), **sciplot** (Morales et al., 2020), **sjstats** (Lüdecke, 2020), **tidyr**(Wickham and Henry, 2020), **WRS2** (Mair and Wilcox, 2020)

1.1 A note on R functions and usage style

With the large set of packages needed for a suite of functions used in this document, it can often be confusing to the novice R user to sort out which package a particular function comes from. I have used two strategies to aid in avoidance of this confusion. Often, in introductory text in a section, I refer to a "`function` from a **package** (e.g., `bar` from **foo**). In other places I use the double colon convention to call a function from its package. For example"`psych::describe`" calls the `describe` function from the **psych** package. In cases where neither method of specification exists, that would typically mean that the function is in the base set of R packages that are installed on initial setup. For example, `aov` is in the base system **stats** package and is used without specifying that.

1.2 Resources

The following list will provide a good start for those needing a broader background in ANOVA techniques and more detailed sources for the primary packages employed in this document.

In addition, the following internet resources can be helpful.

- Salvatore S. Mangiafico's R Companion: https://rcompanion.org/

handbook/I_09.html

- http://dwoll.de/rexrepos/posts/anovaRBp.html
- http://www.jason-french.com/tutorials/repeatedmeasures.html
- https://www.datanovia.com/en/lessons/repeated-measures-anova-in-r/#one-way-repeated-measures-anova

Chapter 2

Import data set and do Exploratory Data Analysis

Many applications of repeated measures designs involve simply tracking participant across time and measuring the DV at fixed time points. It is also possible to employ such a design when the IV is a manipulated variable. The levels of the repeated factor thus represent different "treatment" conditions. The primary example data set used here is of this latter type. In such experiments, the order of treatments is often randomized across participants/subjects. The data set used here is a textbook example, taken from the Keppel textbook (Keppel and Wickens, 2004, exercise #1, Ch. 16, pp 366-367).

The study outlined in the exercise presumed to evaluate an appetitive behavior among rats, tongue protrusion (called DV in the data set). Tongue protrusions are also used in social situations, presumably as part of a chemical senses system for evaluating airborne molecules that may carry social significance. Accordingly, a study was designed where rats were exposed to bedding types. Two control types of conditions were employed, clean bedding and bedding from the rat's own home cage. Three other conditions were bedding from other species, iguana, whiptail lizard, and kangaroo rat. The IV (called "type" in the data frame) thus had five levels, with each of ten subjects being measured under each level, making the study a simple one-factor repeated measures design.

Other data sets are used for explicit purposes in other sections of this document.

2.1 Data Import

Many of the methods for repeated measures analysis require the "long" format data file. This is available as a .csv file and is imported with the code found below. In a separate document/tutorial, the conversion from wide to long format

(or vice versa) is illustrated. There are several ways of doing this conversion in R, but at the time of this writing, tidyverse tools are the most convenient. The `pivot_longer` and `pivot_wider` functions in the **tidyr** package are useful ways to execute the conversion, but other methods are also illustrated in the separate tutorial document.

This code imports the primary data file, and the code chunk also designates the "snum" case number variable as a factor since that is required in most of the analyses to follow in later chapters. It is imported as a numeric variable. In addition, the ordering of the "type" levels is changed from the default alphabetical to match prior graphical work with the data set and prior analyses and analyses using contrasts in SPSS.

```
# read data set - long form exported from SPSS
rpt1.df <- read.csv("data/1facrpt_long.csv", stringsAsFactors=TRUE)
# change the snum variable to a factor variable (was numeric)
rpt1.df$snum <- as.factor(rpt1.df$snum)
# change the order of the factor levels of type to match the
# original order and match our prior SPSS work,
# including setting  up contrasts
rpt1.df$type <- ordered(rpt1.df$type,
                        levels=c("clean","homecage","iguana",
                                 "whiptail", "krat"))
# look at a few lines from the data frame
headTail(rpt1.df)
```

```
##     snum     type  DV
## 1      1    clean  24
## 2      1 homecage  15
## 3      1   iguana  41
## 4      1 whiptail  30
## ... <NA>     <NA> ...
## 47    10 homecage   7
## 48    10   iguana   4
## 49    10 whiptail   7
## 50    10     krat  23
```

2.2 Numerical Exploratory Data Analysis

Using `describeBy` from the **psych** package, we can now examine a few descriptive statistics for the five conditions. Note that only a subset of the statistics produced by `describeBy` are requested, and the resultant table is more nicely formatted using `gt`. The summaries are also split into two tables to control the width of the tables.

```
d1 <- describeBy(rpt1.df$DV, group=rpt1.df$type,
                 type=2, mat=T)[,c(2,4:9)]
```

```
gt(d1)
```

group1	n	mean	sd	median	trimmed	mad
clean	10	6.8	7.177124	6.5	5.500	5.1891
homecage	10	6.0	5.676462	5.5	5.625	6.6717
iguana	10	7.3	12.445883	3.0	4.000	4.4478
whiptail	10	9.4	8.796464	6.5	8.000	7.4130
krat	10	23.1	18.174769	16.5	22.125	10.3782

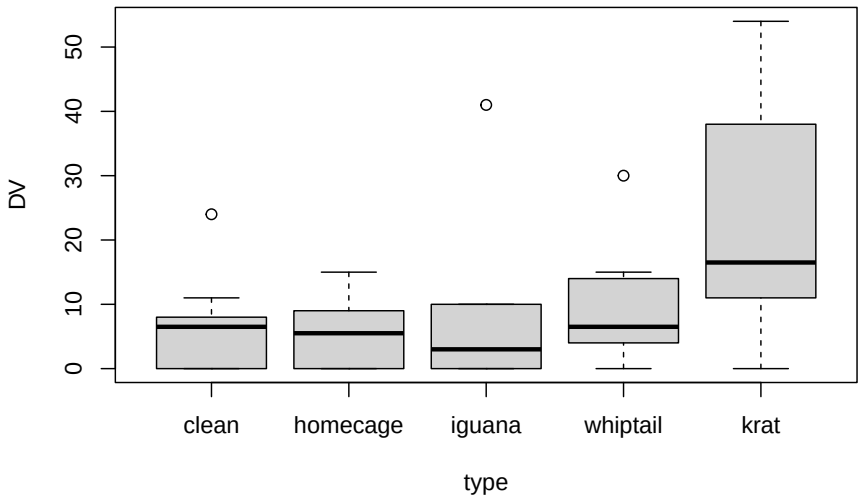
```
d2 <- describeBy(rpt1.df$DV, group=rpt1.df$type,
                 type=2, mat=T)[,c(2,10:15)]
gt(d2)
```

group1	min	max	range	skew	kurtosis	se
clean	0	24	24	1.5779391	3.4183872	2.269606
homecage	0	15	15	0.6150626	-0.7291575	1.795055
iguana	0	41	41	2.6438283	7.4942010	3.935734
whiptail	0	30	30	1.4989085	2.7856163	2.781686
krat	0	54	54	0.7791971	-0.6475045	5.747367

2.3 Graphical EDA

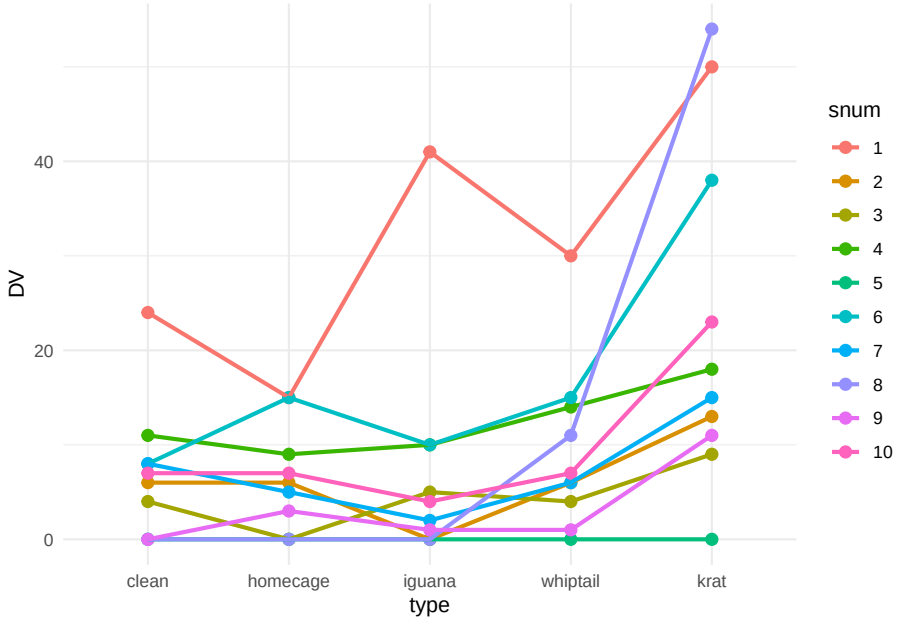
Simple boxplots are readily obtained wtih this long-format data set. The `DV~type` model specification is possble because of this long-format data structure, even though the "type" variable is not variable based on different groups of participants.

```
boxplot(DV~type, data=rpt1.df)
```



One EDA plot that is often used for repeated measures designs is called a profile plot. It is a simple line graph where each case is represented by one line. This type of plot is argued to enable a visual comparision of the patterns across conditions and the assesment of how consistently the cases show the overall pattern. The negative aspect of the plot is that for a study like this where the IV is categorical, the shape of the lines is actually meaningless. If the repeated measure factor were Time, then such a plot would have greater value. Recall that the placement of the categories along the X axis is actually arbitrary here, so the profiles have no intrinsic meaning - they just permit a comparison of cases to one another. Also, apologies for the non-colorblind friendly palette of colors used for the lines…….. (see a later section of this chapter for a method of control of color palettes in `ggplot`).

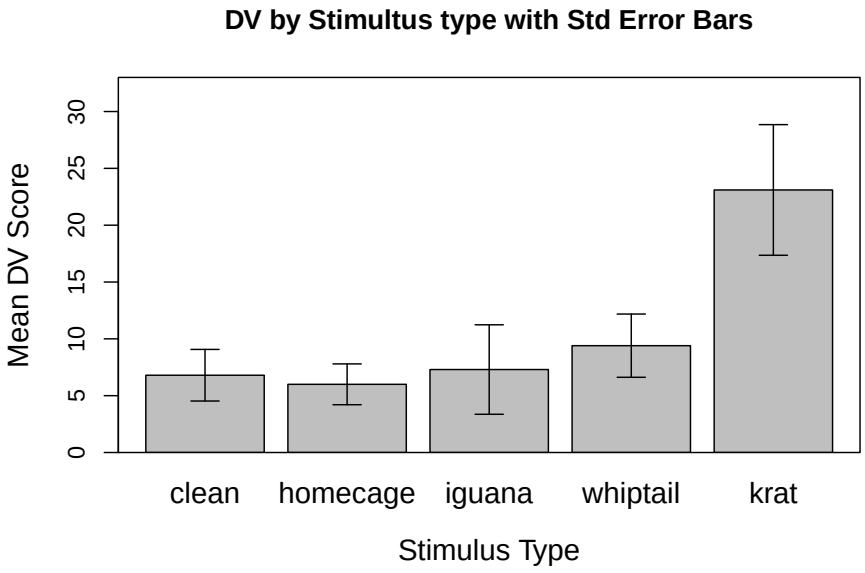
```
#library(ggplot2)
ggplot(rpt1.df, aes(type, DV, colour=snum)) +
  geom_point(size = 2.5) +
  geom_line(aes(group = snum), linewidth = 1) +
  theme_minimal()
```



When the repeated factor is categorical rather than quantitative (like time), it is almost expected that the data be summarized graphically with bar graphs plotting the means with error bars (of some sort) added. This plot, also called a dynamite plot, has received considerable criticism as we have discussed. The `bargraph.CI` function can readily draw such a graph, as easily as we saw it done for between-groups designs - this is enabled because of the long-format structure of the data file.

Note that those std errors of the mean are based on between subject variation and have no specific usage in regard to the omnibus inferential tests performed in chapter 3 and later.

```
#require(sciplot)
bargraph.CI(rpt1.df$type,rpt1.df$DV,lc=TRUE, uc=TRUE,legend=T,
            cex.leg=1,bty="n",col="gray75",
            ylim=c(0,33),
            xlab="Stimulus Type",
            ylab="Mean DV Score",main="DV by Stimultus type with Std Error Bars",
            cex.names=1.25,cex.lab=1.25)
box()
```



2.4 Graphical EDA with ggplot2: boxplots and bar graphs with error bars

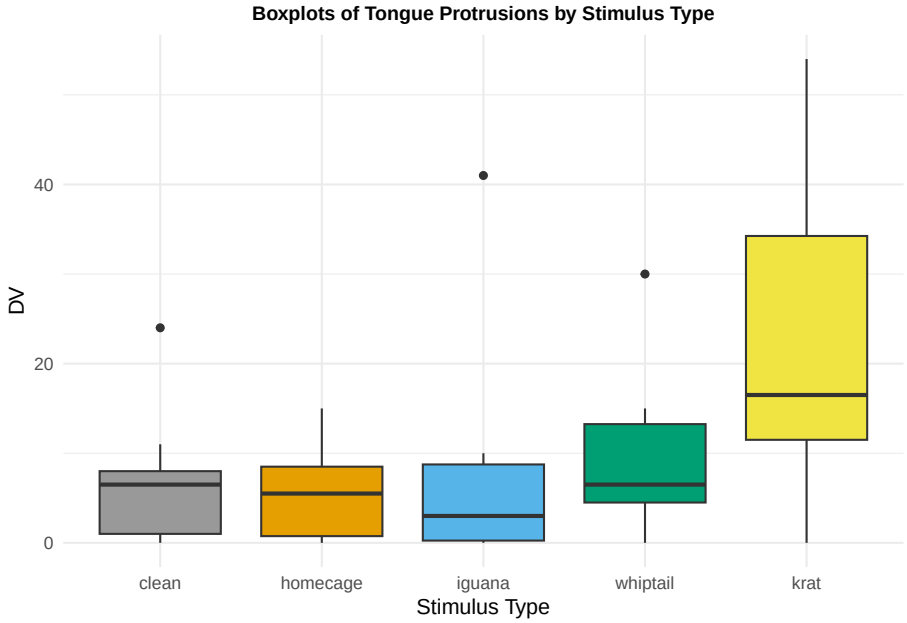
Alternative approaches to boxplots and bar graphs (with error bars) are available with the **ggplot2** package. This section also includes demonstration of a violin plot that includes the mean plus error bars. An important section subsequent to this one explores alternative definitions of standard errors for within-subjects (repeated measures) factors.

First I show a `ggplot` version of the boxplot to reinforce the idea that the **ggplot2** package is a useful tool. The core of the plot is drawn with the first two lines of the `ggplot` function code, and the remainder of the lines of code control stylistic attributes of the plot. A vector of colorblind-friendly color codes is established first and then used for the "fills". Other commented lines of code show other ways of controlling the color scheme. The reader should note that since the categories are labeled, the use of color here is superfluous and probably should be avoided. It provided an opportunity to illustrate the use of a colorblind friendly palette. See, for example:

- http://mkweb.bcgsc.ca/colorblind/
- http://www.cookbook-r.com/Graphs/Colors_(ggplot2)/#a-colorblind-friendly-palette

```
# first, establish a colorblind friendly palette
cbPalette <- c("#999999", "#E69F00", "#56B4E9", "#009E73",
```

```
               "#F0E442", "#0072B2", "#D55E00", "#CC79A7")
p2<-ggplot(data = rpt1.df, aes(x = type, y = DV, fill=type)) +
  geom_boxplot() +xlab("Stimulus Type") + ylab("DV") +
  scale_fill_manual(values=cbPalette) +
  #scale_fill_brewer(palette="Paired") +
  #scale_colour_grey() + scale_fill_grey() +
  ggtitle("Boxplots of Tongue Protrusions by Stimulus Type") +
  guides(fill=FALSE) + # removes legend
  theme_minimal() +
  theme(plot.title = element_text(size=10,
                                  face = "bold", hjust = .5))
p2
```



The traditional bar graph to depict means with standard error bars (or CI's) can also be achieved with `ggplot`. In order to create this graph, the summary statistics have to be extracted from the data set. Means, standard errors, and CI's are obtained from long-format data frames with the `summarySE` function from the **Rmisc** package. The error bars are then plotted one "se" or one "ci" up and down from the mean. The variable in the summarySE-produced data frame called "DV" is actually the man of the condition. The variable called "ci" is the one-sided distance from the mean to an upper or lower confidence limit (95% here).

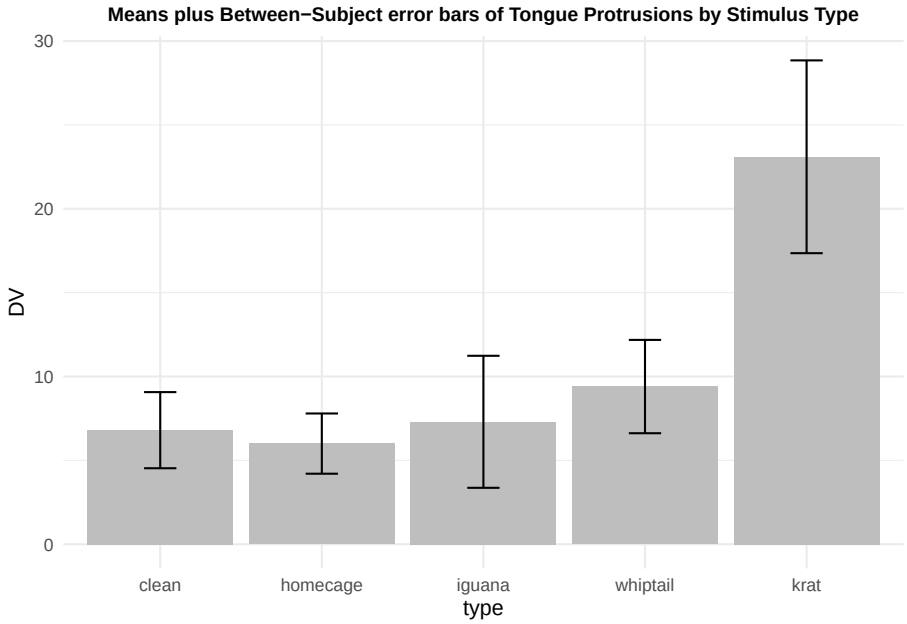
```
rpt1summary.b <- Rmisc::summarySE(rpt1.df,
                                  measurevar="DV",
```

```
                                    groupvars="type",
                                    conf.interval=.95)
rpt1summary.b
```

```
##       type  N   DV        sd       se        ci
## 1    clean 10  6.8  7.177124 2.269606  5.134205
## 2 homecage 10  6.0  5.676462 1.795055  4.060696
## 3   iguana 10  7.3 12.445883 3.935734  8.903248
## 4 whiptail 10  9.4  8.796464 2.781686  6.292611
## 5     krat 10 23.1 18.174769 5.747367 13.001446
```

The values from this data frame are then used to draw the graph:

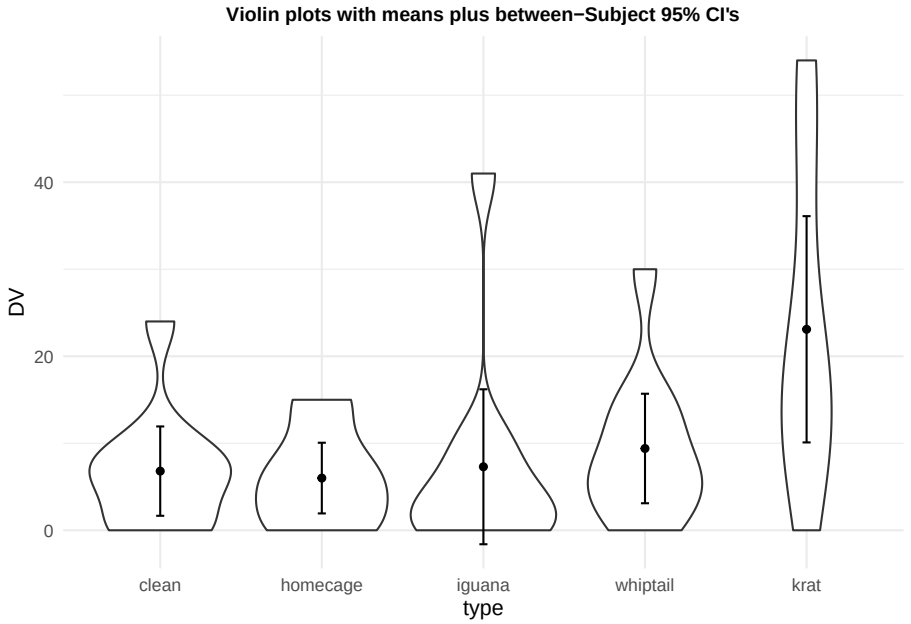
```
p3 <- ggplot(rpt1summary.b, aes(x=type, y=DV, fill=type)) +
  geom_bar(position=position_dodge(), stat="identity", fill="gray") +
  geom_errorbar(aes(ymin=DV-se, ymax=DV+se),
                width=.2,                    # Width of the error bars
                position=position_dodge(.9)) +
  ggtitle("Means plus Between-Subject error bars of Tongue Protrusions by Stimulus Type") +
  guides(fill=FALSE) + # removes legend
  theme_minimal() +
  theme(plot.title = element_text(size=10,
                                    face = "bold", hjust = .5))
p3
```



Another possibility is to draw violin plots for each condition and to also display

the means and standard errors (or confidence intervals as I did here). Notice that with the floor of zero, one of the CI's actually extends to a non-sensical value - a frequent occurrence when the distribution is skewed with a mean close to zero. This will be a more helpful plot when sample sizes are larger and the densities are not dominated by small numbers of points in regions of the Y axis DV scale.

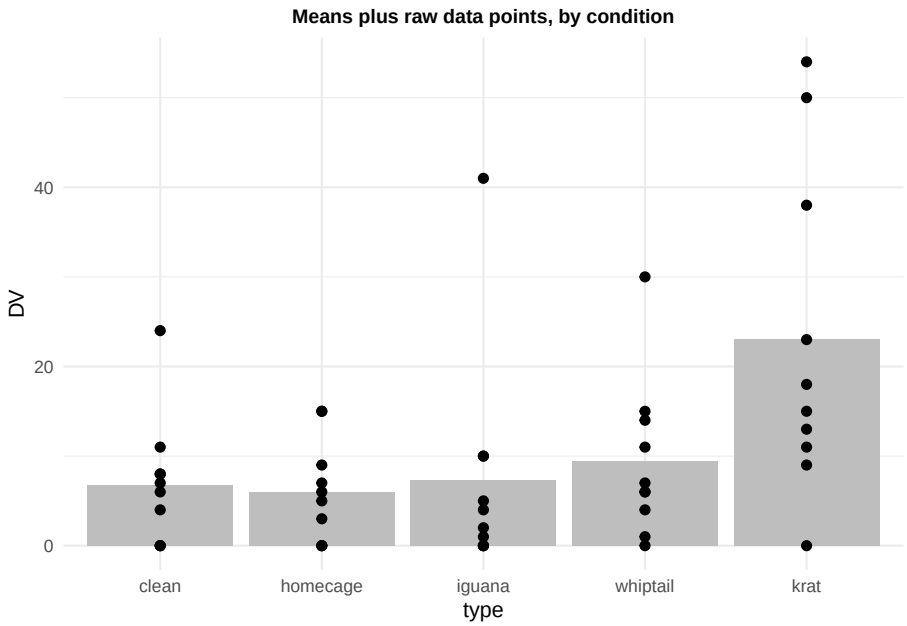
```
p4 <- ggplot(rpt1.df, aes(x=type, y=DV)) +
  geom_violin() +
  geom_point(aes(y=DV),data=rpt1summary.b, color="black") +
  geom_errorbar(aes(y=DV, ymin=DV-ci, ymax=DV+ci),
                color="black", width=.05, data=rpt1summary.b)+
  ggtitle("Violin plots with means plus between-Subject 95% CI's") +
  guides(fill=FALSE) + # removes legend
  theme_minimal() +
  theme(plot.title = element_text(size=10,
                                  face = "bold", hjust = .5))
p4
```



Showing the raw data is always a good thing. The profile plot above shows this, but here we see how to create a traditional bar chart of the means with raw data points added.

```
p4b <- ggplot(rpt1summary.b, aes(x=type, y=DV)) +
  geom_bar(position=position_dodge(), stat="identity", fill="gray") +
  geom_point(aes(y=DV, x=type), data=rpt1.df,color="black", size=2) +
  ggtitle("Means plus raw data points, by condition") +
```

```
  guides(fill=FALSE) + # removes legend
  theme_minimal() +
  theme(plot.title = element_text(size=10,
                                  face = "bold", hjust = .5))
p4b
```



2.5 What are the proper error bars for a plot of a within-subjects design?

There is a literature concerning the correct choice for error bars to display in the kinds of plots shown above with repeated measures designs. The issues are twofold and relate to the origin of the reason for displaying error bars. First, what is the purpose of displaying error bars? Sometimes they are employed simply to show an index of dispersion with a group, or within a condition as is the case for a repeated measures factor level. Error bars that are standard deviations can accomplish this, but showing the raw data as we saw above is also a good choice. At other times error bars are employed to provide a method of "inference by eye" (Cumming and Finch, 2005) by visualization of the degree of overlap of error bars, confidence intervals and means.

In the plots drawn above, depicting means plus/minus error bars or confidence intervals, the "error" is derived from the "subject" variation at each level of the repeated factor (each condition). However, this is not the error that is related to any of the inferences done regarding variation in means in such designs. In

fact, the whole purpose of the repeated measures design is to treat it as a randomized block design where the main effect of "subject", or case, is pulled out of the analytical system and the remaining error that is used is the condition by subject interaction (the "Axs" term). So, the standard errors used above are not helpful because they come from a mix of the main effect of subject and the condition by subject interaction term - think of subject variation at a level of the repeated factor as a simple main effect. The sum of the variation of that set of simple main effects would comprise a pooling of the main effect of subject and the interaction term. If a descriptive measure of dispersion is desired, then simple standard deviations might be displayed or better yet, the raw data points. But if a connection to inferential analyses performed on the data set is the goal, then a different kind of error bar should be used - one that relates to the Axs interaction term.

A literature dating back to Tukey has argued for the ability to do "inference by eye" in order to decide whether two levels of a factor are "significantly" different. Cumming and Finch (2005) gave specific guidance about the deviation between means of two conditions relative to std errors or CI's in order to "see" that two means would be "significantly" different. For standard errors plotted on means in bar graphs, the two means would have to be approximately 3 std errors apart to reach significance at the nominal .05 level. This is the Cumming and Finch "Rule 7). CI overlap would be approximately 50% for the same conclusion. But this presumes that the"correct" std error is used (and employed in the construction of a confidence interval).

2.5.1 The method of Loftus and Masson (1994)

Several decades ago, I began using what I called "generalized standard errors" for plotting purposes in depiction of means from ANOVA designs (e.g., (Phillips and Dudek, 1989)). The idea was to generate a standard error of the mean using the appropriate error term (actually the square root of that MS) and divide by the square root of the appropriate N. This would give a standard error depiction that reflected the error that was actually used in the inferences done with ANVOA methods, rather than the collection of individual group or condition standard errors such as those depicted above. Over a decade later, Loftus and Masson (1994) published an article essentially arguing the same thing. The article laid out the idea that for repeated measures designs, the ANOVA error term (for the omnibus F test) in a 1-factor design is the condition by subject interaction term (Axs in the randomized block notation). The square root of this MS, divided by the square root of N would be a more appropriate standard error to use for error bars in bar graphs of condition means - and each condition would have this same standard error placed on the point/bar on the plot.

It is fairly simple to construct such a standard error once the ANOVA is complete. This basic one factor repeated measures ANOVA is explored in detail in the next chapter, but the analysis is accomplished here in order to find the MS(Axs) term.

```
contrasts(rpt1.df$type) <- contr.sum
fitplot <- aov(DV ~ type + Error(snum/type),data=rpt1.df)
summary(fitplot)
```

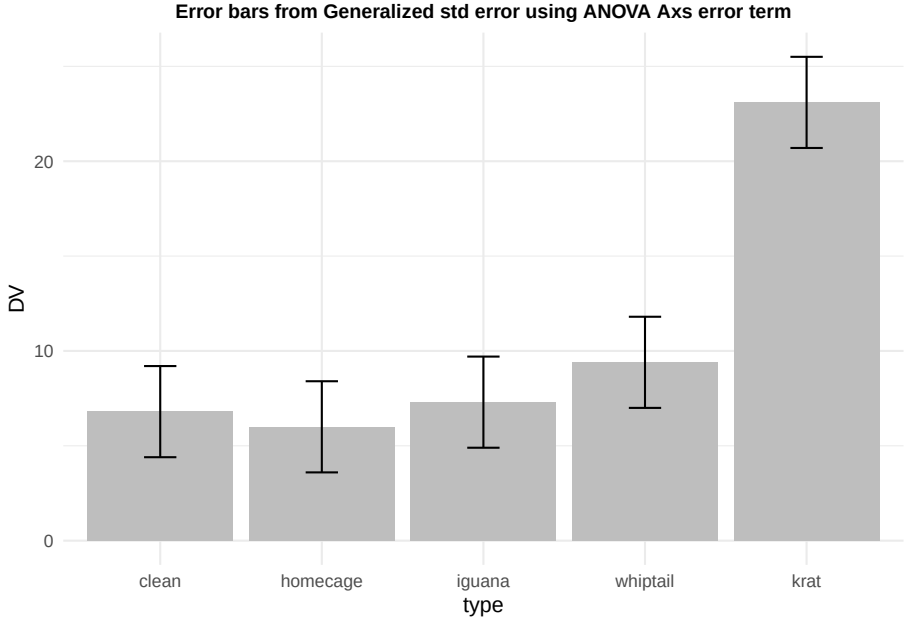
```
##
## Error: snum
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals  9   3740   415.5
##
## Error: snum:type
##           Df Sum Sq Mean Sq F value   Pr(>F)
## type       4   2042   510.4   8.845 4.42e-05 ***
## Residuals 36   2077    57.7
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The standard error can be manually created from the MS error term (Axs) found in the summary table.

```
wstderr <- (57.7^.5)/(10^.5)
```

We can use the same summary data frame as above in order to extract the means from the data set, and employ the manually created standard error value to draw error bars (+/- 1 std error) on the bars. The same std error is used for each condition in this plot.

```
p5 <- ggplot(rpt1summary.b, aes(x=type, y=DV, fill=type)) +
  geom_bar(position=position_dodge(), stat="identity", fill="gray") +
  geom_errorbar(aes(ymin=DV-wstderr, ymax=DV+wstderr),
                width=.2,                    # Width of the error bars
                position=position_dodge(.9)) +
  ggtitle("Error bars from Generalized std error using ANOVA Axs error term") +
  guides(fill=FALSE) + # removes legend
  theme_minimal() +
  theme(plot.title = element_text(size=10,
                                  face = "bold", hjust = .5))
p5
```



This generalized standard error has the advantage of being the actually error variance that is used in the analysis, albeit for the omnibus F test. It is, however, burdened by the same sphericity assumption that we will see can handicap its use in that omnibus ANOVA - covered in later chapters.

2.5.2 Alternatives to the Loftus and Masson approach

The Loftus and Masson approach received some criticism (Bakeman and McArthur, 1996; Morrison and Weaver, 1995) as being too difficult to execute and perhaps not exactly what was needed (I have found that criticism to be shallow). An alternative was proposed by Cousineau (2005). This idea was a method to remove the between subject variation from the error term with a "normalization" procedure. The approach centered the data for each subject/case around its own aggregate mean. Thus with this method, each case would have the same mean, although still varying across the repeated measure factor. The remaining variation of data points at each level of the repeated factor would reflect the components of the Axs interaction. Morey(2008) pointed out a flaw in the "normalization" method and presented a corrected version. This corrected std error or CI is provided by the `summarySEwithin` function in the **Rmisc** package and that is what is implemented here, first with the creation of the summary data frame.

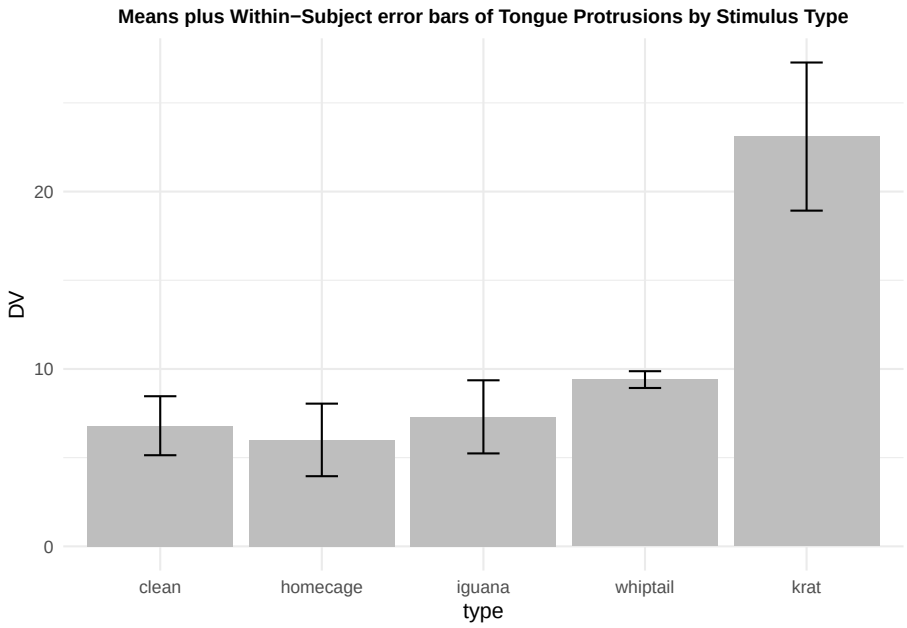
```
rpt1summary.w <- Rmisc::summarySEwithin(rpt1.df,
                                        measurevar="DV",
                                        withinvars="type",
                                        idvar="snum")
```

```
rpt1summary.w
```

```
##       type  N   DV        sd        se       ci
## 1    clean 10  6.8  5.258010 1.6627287 3.761354
## 2 homecage 10  6.0  6.467182 2.0451026 4.626343
## 3   iguana 10  7.3  6.518734 2.0614046 4.663221
## 4 whiptail 10  9.4  1.495177 0.4728166 1.069586
## 5     krat 10 23.1 13.202883 4.1751181 9.444773
```

Now the graph can be redrawn using these "normalized" standard errors or "normalized" confidenc intervals. This plot uses the standard errors:

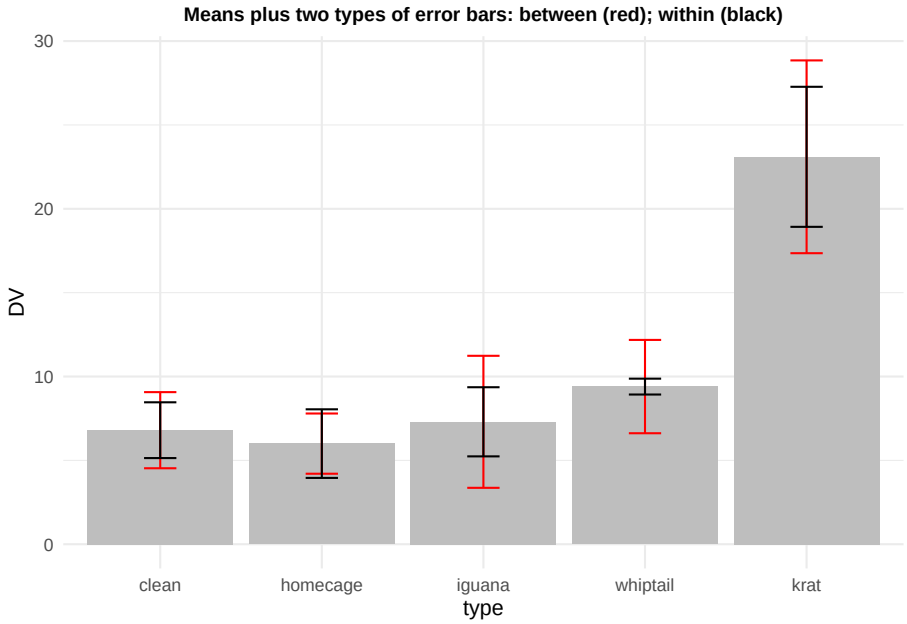
```
p6 <- ggplot(rpt1summary.w, aes(x=type, y=DV, fill=type)) +
  geom_bar(position=position_dodge(), stat="identity", fill="gray") +
  geom_errorbar(aes(ymin=DV-se, ymax=DV+se),
                width=.2,                    # Width of the error bars
                position=position_dodge(.9)) +
  ggtitle("Means plus Within-Subject error bars of Tongue Protrusions by Stimulus Type") +
  guides(fill=FALSE) + # removes legend
  theme_minimal() +
  theme(plot.title = element_text(size=10,
                                  face = "bold", hjust = .5))
p6
```



In order to compare, we can add both the between subjects error bar that was used initially and this "normalized" approach on the same plot. It is clear that

the corrected standard error will, appropriately, be smaller.

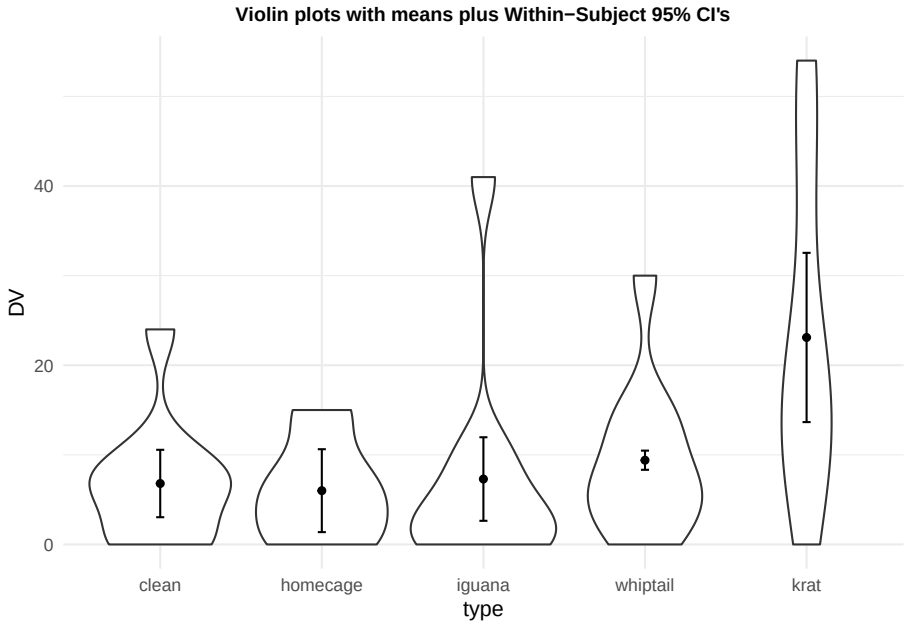
```
p7 <- ggplot(rpt1summary.w, aes(x=type, y=DV, fill=type)) +
  geom_bar(position=position_dodge(), stat="identity", fill="gray") +
  geom_errorbar(aes(ymin=DV-se, ymax=DV+se), data=rpt1summary.b,
                width=.2,                    # Width of the error bars
                position=position_dodge(.9),
                colour="red") +
  geom_errorbar(aes(ymin=DV-se, ymax=DV+se),
                width=.2,                    # Width of the error bars
                position=position_dodge(.9)) +

  ggtitle("Means plus two types of error bars: between (red); within (black)") +
  guides(fill=FALSE) + # removes legend
  theme_minimal() +
  theme(plot.title = element_text(size=10,
                                  face = "bold", hjust = .5))
p7
```



We can also redraw the violin plot redone with CI's based on the within-subject std error:

```
p8 <- ggplot(rpt1.df, aes(x=type, y=DV)) +
  geom_violin() +
  geom_point(aes(y=DV),data=rpt1summary.w, color="black") +
  geom_errorbar(aes(y=DV, ymin=DV-ci, ymax=DV+ci),
```

```
                color="black", width=.05, data=rpt1summary.w)+
  ggtitle("Violin plots with means plus Within-Subject 95% CI's") +
  guides(fill=FALSE) + # removes legend
  theme_minimal() +
  theme(plot.title = element_text(size=10,
                                  face = "bold", hjust = .5))
p8
```



The Cousineau/Morey method seems to be an improvement on the traditional "between subjects" standard error initially outlined above, but it too can be criticized. Those individual standard errors are not used anywhere in the analysis, so it is not clear how meaningful they are. If they are to assist in "inference by eye", then they should relate to comparisons between means of different conditions. They are not directly related to inferences about such comparisons.

2.5.3 Franz and Loftus 2012 approach for pairwise comparisons

Franz and Loftus (2012) argued that "normalization method leads to biased results and is uninformative with regard to circularity." They made the simple point that if the desire is to visually compare pairs of means from different conditions, then the error bars should reflect those pairwise comparisions. And, comparison of pairs of repeated measure levels is tantamount to performing a dependent samples t-test. So their direct solution is to depict the mean differences between pairs and use the standard error that is created to perform each of the

different paired t-tests. In the next section, we will couch this perspective as a narrowly defined se of analytical contrasts, but for now, it makes considerable sense.

In order to examine the pairwise comparisons, the first step (as in the dependent samples t-test) is to create difference scores for each subject/case for each pair of levels. In the example we are working with here, with five levels of the repeated measure factor, this produces ten pairs and thus ten new variables. Rather than labor to obtain those paired difference variables in R with the long-format data set, I created the ten new variables beginning with a wide format version of the data file, in Excel, and then saved it to a .csv file.

```
pairdiffs <- read.csv("data/1facrpt_with_diffs.csv")
pairdiffs
```

```
##    snum diff2.1 diff3.1 diff4.1 diff5.1 diff3.2 diff4.2 diff5.2 diff4.3 diff5.3
## 1     1      -9       17        6       26       26       15       35      -11        9
## 2     2       0       -6        0        7       -6        0        7        6       13
## 3     3      -4        1        0        5        5        4        9       -1        4
## 4     4      -2       -1        3        7        1        5        9        4        8
## 5     5       0        0        0        0        0        0        0        0        0
## 6     6       7        2        7       30       -5        0       23        5       28
## 7     7      -3       -6       -2        7       -3        1       10        4       13
## 8     8       0        0       11       54        0       11       54       11       54
## 9     9       3        1        1       11       -2       -2        8        0       10
## 10   10       0       -3        0       16       -3        0       16        3       19
##    diff5.4
## 1       20
## 2        7
## 3        5
## 4        4
## 5        0
## 6       23
## 7        9
## 8       43
## 9       10
## 10      16
```

But in order to use ggplot and the summary functions to draw the plot, we need to convert this wide data frame to a long version, using tools from **tidyr**. The `pivot_longer` function makes this simple. The 10x10 data matrix is now a 100 x 1 data matrix and the data frame has id variables for subject number and pair identity.

```
pairs_long <-
  tidyr::pivot_longer(data=pairdiffs,
                      cols=2:11,
                      names_to="pair",
```

```
                      values_to="DV")
pairs_long$pair <- as.factor(pairs_long$pair)
pairs_long
```

```
## # A tibble: 100 x 3
##     snum pair       DV
##    <int> <fct>   <int>
##  1     1 diff2.1    -9
##  2     1 diff3.1    17
##  3     1 diff4.1     6
##  4     1 diff5.1    26
##  5     1 diff3.2    26
##  6     1 diff4.2    15
##  7     1 diff5.2    35
##  8     1 diff4.3   -11
##  9     1 diff5.3     9
## 10     1 diff5.4    20
## # ... with 90 more rows
```

At this point, we need to compute the traditional standard error for each of the ten variables since that is the standard error used in the paired t-tests. The `summarySE` function from **Rmisc** accomplishes this as we saw with the "between subjects" standard errors computed several sections above. Note that division of the mean (called DV in this data frame) by the "se" yields the t value if that particular pair were examined with a dependent samples t-test (see a later chapter).

```
# treat as BG for different variables (ten of them)
pairs_summary <- Rmisc::summarySE(pairs_long,
                                  measurevar="DV",
                                  groupvars="pair")

pairs_summary
```

```
##       pair  N   DV        sd       se        ci
## 1  diff2.1 10 -0.8  4.237400 1.339983  3.031253
## 2  diff3.1 10  0.5  6.450667 2.039880  4.614530
## 3  diff3.2 10  1.3  9.238206 2.921377  6.608614
## 4  diff4.1 10  2.6  4.115013 1.301281  2.943703
## 5  diff4.2 10  3.4  5.541761 1.752459  3.964337
## 6  diff4.3 10  2.1  5.782156 1.828478  4.136306
## 7  diff5.1 10 16.3 16.275749 5.146844 11.642969
## 8  diff5.2 10 17.1 16.251154 5.139066 11.625375
## 9  diff5.3 10 15.8 15.504838 4.903060 11.091493
## 10 diff5.4 10 13.7 12.596737 3.983438  9.011163
```

Now we can draw a plot that produces the ten means of the paired difference

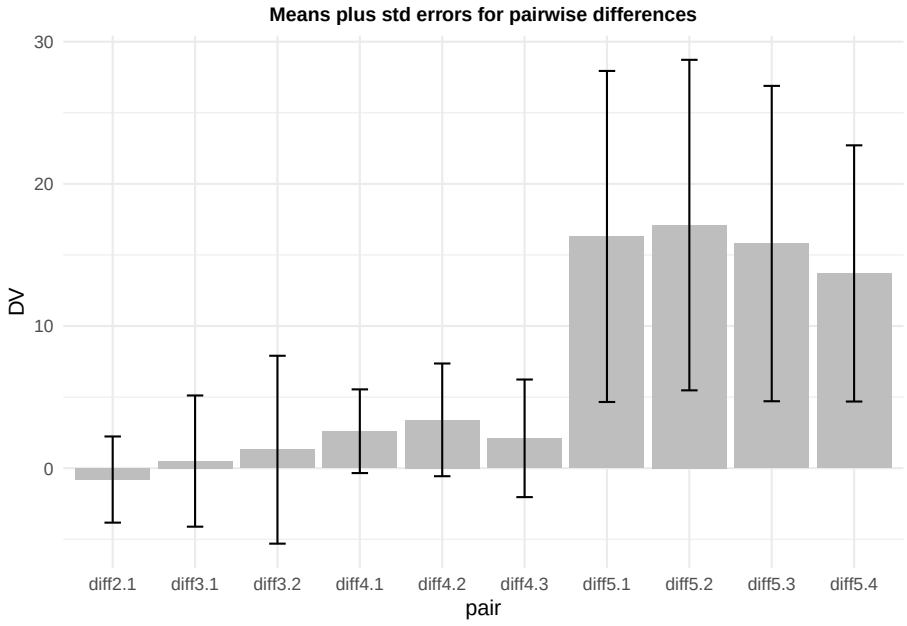
score variables. The t-test of each of these paired differences is compared to a null hypothesis value of zero, so our "inference by eye" approach would ask whether the CI derived from these standard errors overlaps the null value of zero. This plot depicts the means with 95% CI's. We conclude that condition 5 (kangaroo rat) is significantly different from each of the other four conditions.

The main value of taking this paired approach is that (1) the errors associated with each pair are specific to that comparison and (2) not burdened by the sphericity assumption.

A downside of this method is that as implemented, it does not take into account the accumulated error problem of the multiple simultaneous tests, most of which are probably not a priori hypotheses. Fortunately, the CI could be adjusted with one of the many bonferroni/holm/FDR methods available.

With this kind of plot, the visual purpose is to compare each mean to the null hypothesis value of zero, rather than to each other.

```
p9 <- ggplot(pairs_summary, aes(x=pair, y=DV, fill=pair)) +
  geom_bar(position=position_dodge(), stat="identity", fill="gray") +
  geom_errorbar(aes(ymin=DV-ci, ymax=DV+ci), data=pairs_summary,
                width=.2,                    # Width of the error bars
                position=position_dodge(.9)) +
  ggtitle("Means plus std errors for pairwise differences") +
  guides(fill=FALSE) + # removes legend
  theme_minimal() +
  theme(plot.title = element_text(size=10,
                                  face = "bold", hjust = .5))
p9
```



2.6 Conclusions on error bars and plotting graphs and specific errors for analytical/orthogonal contrasts

It strikes me that all of this discussion is obviated if the goal of the analysis is hypothesis driven inference. The method of choice for evaluating specific hypotheses is the use of analytical contrasts. These contrasts are conceptual extensions of the paired t-test approach, but are more elegant. They will be considered in a later chapter of this tutorial, and plots analogous to the paired difference plot just shown will be created then. That approach can also be couched in terms of orthogonal sets which may be desireable when concerns about alpha rate inflation exist.

2.7 Conclusions from EDA

With the EDA, it becomes clear that the rate of tongue protrusion among the ten rats was fairly consistently low, except for the "krat" condition where it was higher and more dispersed. Inferential methods for evaluating this pattern are found in the next several chapters.

Chapter 3

Traditional Approaches to One Factor Repeated Measures Designs

The primary goal of this chapter is the elaboration of the traditional "Univariate" approach to the 1-factor Repeated Measures design, evaluation of it's sphericity assumption. These methods have been the topic of considerable discussion in light of the development of alternatives such as linear mixed effects modeling. Extreme perspectives even argue that the SS partitioning approach of the traditional method is "dead" (McCulloch, 2005). We emphasize that well executed traditional methods, especially those centered on contrast based hypothesis testing, are sound. Later sections in this and other chapters delve into alternative methods. A secondary goal of the chapter is provision of code to implement the multivariate approach to evaluation of the omnibus null hypothesis that all condition means are equal. The multivariate approach is not burdened with the sphericity assumption that has brought the univariate approach under strong criticism. The discussion of which of these to use had been centered around relative power given varying sample sizes and degree of non-sphericity. That conversation has largely been replaced by strong recommendations to use the linear mixed models approach that is outlined in a later chapter. An additional major goal of this document is to explore methods for examining analytical/orthogonal contrasts and pairwise comparison post hoc tests and the stage will be set to produce those analyses in the next chapter.

3.1 The traditional "univariate" GLM approach to the repeated measures problem and the Multivariate approach

The initial sections here, on the univariate approach, emphasize the implementation of the repeated measures design as the theoretical sibling of randomized block designs. It approaches the question in the style of the long litany of experimental design textbooks such as those by Winer, Kirk, Myers, Keppel, and Hays, as well as the current comprehensive textbook by Maxwell, Delaney and Kelley (Howell, 2013; Hays, 1994; Keppel and Wickens, 2004; Kirk, 2013; Myers et al., 2010; Winer et al., 1991). In this variance partitioning approach, three sources of variance are identified, owing to the factorial arrangement of the two factors (IV and "subjects/case":

1. "treatment", the repeated measures factor, called "factor A" in the standard textbook approach.
2. "subject" or case (the "blocking" factor)
3. The Axs interaction term which will serve as the error term for factor A in this approach.

The theory of the F test from the traditional/univariate approach to omnibus F test for this this one factor repeated measures design is summarized in this table:

Test of Omnibus Effect of Factor A:

Source	df	EMS	Error Term
s	s-1	$\sigma_E^2 + a\sigma_s^2$	
A	a-1	$\sigma_E^2 + s\theta_A^2 + \sigma_{Axs}^2$	MS_{Axs}
A x s	(a-1)(s-1)	$\sigma_E^2 + \sigma_{Axs}^2$	

The Axs interaction is usually listed in software output from R as a "residual". It's characterization here as an interaction stems from the perspective that a 1 factor repeated measures design is a special case of a randomized block design where the treatment and subject factors are marginal effects (main effects) and the interaction term is possible because each subject is tested under all treatment conditions. In many repeated measures studies, the repeated factor (IV) is characterized simply as a time factor where the same DV is measured at varying time points. In our first example data set, outlined in the previous chapter, the IV was actually a collection of five treatment conditions and each participant was measured under each condition.

The multivariate evaluation approaches the test of the omnibus null hypothesis in a different way, and it is also illustrated in two of the following sections.

The major challenge in replicating the information set that we learned to derive from the SPSS MANOVA or GLM procedure is that there is not simply one

function or a small set of functions that can give us everything. That list includes:

In this chapter,

1. The omnibus F test of the repeated factor - in some software, this is called the "averaged F test"
2. Evaluation of the sphericity assumption
3. GG and HF correction factors and adjusted p values
4. Implementation the multivariate approach to evaluation of the omnibus null hypothesis

In later chapters we will attempt to add:

5. Easy implementation of orthogonal/analytical contrasts
6. Error terms specific to the contrasts
7. Post hoc evaluation of pairwise comparisons among levels

We will piece these analyses together with several different approaches, and will find that finding correct error terms for contrasts (specific error terms that permit avoidance of the sphericity assumption) is not easily accomplished.

The multiple different ways of accomplishing the omnibus ANOVA found in this chapter can be a bit bewildering. The first method uses the standard/basic approach that employs base system functions. The most efficient approach is probably Method V, using the **afex** package. The other methods provide alternative strategies that can be helpful in some situations.

3.1.1 Method I: Use of the `aov` function

Structuring the data set in the "long" format as was done in Chapter 2 permits the standard `aov` function to partition the variation into the two marginal (main) effects and the interaction term. This requires only a slight change in the code structure compared to the use of `aov` in "between-subjects" designs. The "Error" argument tells `aov` that the subject factor is repeated across levels of type. The notation appears to be similar to what we have learned as "nesting". However, subject is not nested within type as the code might imply with the use of the forward slash, even though some in the R world speak of it that way. It is better to read the notation as implying that all levels of type are found under each subject. It strikes me as an unwieldy way of specifying that the Axs interaction term is the Error term to test the main effect of type. But the whole perspective regarding 1-factor repeated measures designs in R is not conceptually aligned with the randomized block idea - I would wager that most users and many programmers are not aware of the analogy.

It is important to understand that use of the `aov` function this way has a requirement that the "case" variable (snum here) is a factor. We already specified that factor characteristic in the earlier chapter where the primary data set was imported in chapter 2.

In addition, leaving the contrast set for the "type" factor at the default indicator/dummy coding is not a good idea for repeated measures designs. `contr.sum` produces deviation/effect coding and either that or orthgonal contrast coding should be employed for repeated measures analyses with the `aov` and other R methods.

Finally, we need to use the `summary` function on the `aov` object, instead of the `anova` function, in order to produce the traditional SS/df/MS/F/pvalue ANOVA table.

```
# perform the 1-factor repeated measures anova
# the notation for specification of the error is not intrinsically
# obvious.  some reading in R and S model specification is required.
# always best to use "sum to zero" contrasts for ANOVA
contrasts(rpt1.df$type) <- contr.sum
fit.1 <- aov(DV~type + Error(snum/type), data=rpt1.df)
summary(fit.1)
```

```
##
## Error: snum
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals  9   3740   415.5
##
## Error: snum:type
##           Df Sum Sq Mean Sq F value   Pr(>F)
## type       4   2042   510.4   8.845 4.42e-05 ***
## Residuals 36   2077    57.7
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Checking the df for the terms summarized is a doublecheck. We can verify the parallel to the two main effects and the interaction term of the randomized blocks perspective is correct. N was ten, so 9 df for the main effect of snum is correct. Note that `summary` labels this term "Residuals". The "type" IV had five levels, so we expected the 4 df for that main effect. And the Axs interaction term should, and does have 9x4=36 df; the interaction term is specified with the "snum:type" notation standard for interactions in `aov` objects. Once again, `summary` on this `aov` object labels this term "Residuals", a duplicative use of the label but we recognize its Axs identity by understanding the origina of the 36 df. The F and p values match our work with SPSS for this same data set.

One way to quickly obtain additional information is with use of the `model.tables` function. I obtained the five condition means and the set of deviation effects for each level of type (deviation effects as would be found using effect coding)

```
#report the means of each level
print(model.tables(fit.1,"means"),digits=3)
```

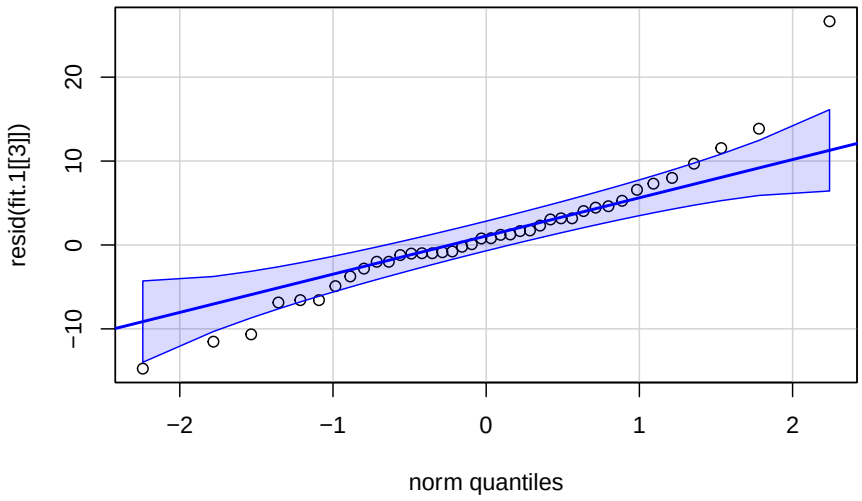
```
## Tables of means
## Grand mean
##
## 10.52
##
##  type
## type
##    clean homecage   iguana whiptail     krat
##      6.8      6.0      7.3      9.4     23.1
```

```
# report deviation effects for each level
print(model.tables(fit.1,"effects", n=TRUE),digits=3)
```

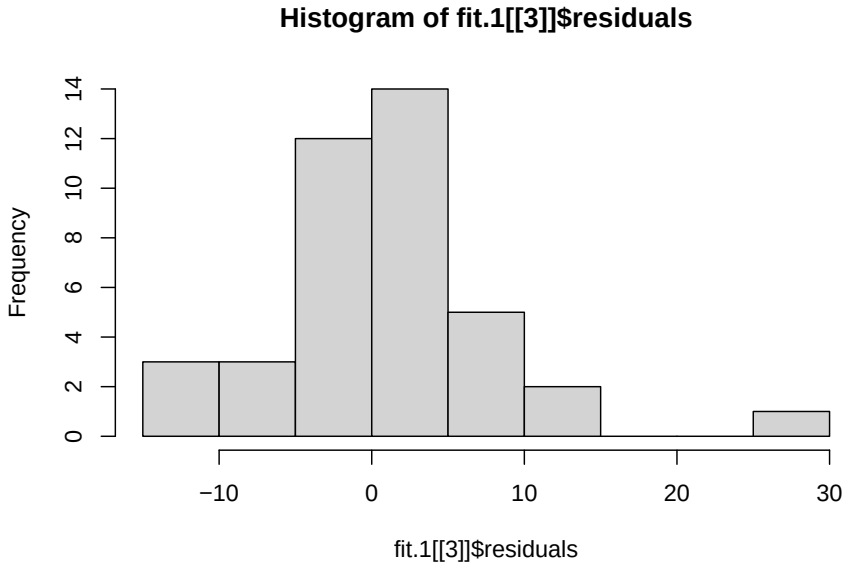
```
## Tables of effects
##
##  type
## type
##    clean homecage   iguana whiptail     krat
##    -3.72    -4.52    -3.22    -1.12    12.58
```

We can evaluate the normality assumption for the residuals with a qqplot. Finding those residuals requires looking at the structure of the fit.1 object and locating the residuals for the Axs interaction. It is in the third section of the list of three components in the fit object (called snum:type here if the 'str(fit.1) function is executed). One major outlier is present and there is a hint of positive skew to the distribution but on balance, most of the data points seem to indicate a reasonable fit to the normality assumption. Note that there are only 40 residuals here, but there are 50 data points. The discrepancy is that the original five variables are transformed to contrast vectors and only four vectors are needed for the five-level factor. Thus four new transformed variables and n=10 yields 40 residuals.

```
#str(fit.1)
qqPlot(resid(fit.1[[3]]),id=FALSE)
```




```
hist(fit.1[[3]]$residuals)
```



3.1.2 Method II, a multivariate linear model

Method II uses the multivariate linear model. It implements both the multivariate test of the omnibus null hypothesis as well as the averaged F test approach with GG and HF corrections. It also permits the user to obtain the Mauchly Sphericity test and GG/HF corrections.

This approach requires the data frame to be "wide", AND, only contain the variables that are the levels of the repeated factor. No additional variables such as a subject code can exist in the matrix of variables that is created from the data frame.

This appproach is modeled after an article by Dalgaard:

http://www.r-project.org/doc/Rnews/Rnews_2007-2.pdf

This approach requires the "wide" version of the data file, and that file is available as the .csv file loaded here. The functions to be used require that the data be in matrix form, not as a data frame. So, that conversion is also accomplished here, following data import.

```
# read data file
rpt1w.df <- read.table("data/1facrpt_wide1.csv",header=T,sep=",")
# change the data from a data frame to a matrix,
# leaving out the snum variable.
# this method requires that the matrix contains
# only the data values
rpt1w.mat <- as.matrix(rpt1w.df)[,2:6]
# view the data
rpt1w.mat
```

```
##       clean homecage iguana whiptail krat
##  [1,]    24       15     41       30   50
##  [2,]     6        6      0        6   13
##  [3,]     4        0      5        4    9
##  [4,]    11        9     10       14   18
##  [5,]     0        0      0        0    0
##  [6,]     8       15     10       15   38
##  [7,]     8        5      2        6   15
##  [8,]     0        0      0       11   54
##  [9,]     0        3      1        1   11
## [10,]     7        7      4        7   23
```

The multivariate linear model is based on use of the standard `lm` function. The model specification argument uses the "~1" syntax to indicate that all variable in the matrix are to be included. Then the `estVar` function can be used to visualize the variance-covariance matrix of the five variables.

```
# first fit the five variate model to get the var/cov matrix
mlmfit.1 <- lm(rpt1w.mat~1)
```

```
estVar(mlmfit.1)
```

```
##             clean homecage    iguana  whiptail      krat
## clean    51.51111 32.88889  82.40000  55.97778  58.46667
## homecage 32.88889 32.22222  50.88889  39.44444  49.22222
## iguana   82.40000 50.88889 154.90000  99.42222 122.41111
## whiptail 55.97778 39.44444  99.42222  77.37778 124.51111
## krat     58.46667 49.22222 122.41111 124.51111 330.32222
```

```
# doublecheck with the `cov` function
#cov(rpt1w.mat)
```

Next, an intercept only model is fit by removing the variates from the full model using the `update` function.

```
# now fit an intercept only model and use the
# anova function in the next section to compare models,
mlmfit.0 <- update(mlmfit.1, ~0)
#estVar(mlmfit.0)
```

A test based on multivariate normal theory evaluates an hypothesis that all variates (repeated measures levels) have equal means is possible with this syntax. A model comparison approach is used by passing the two models created above to the `anova` function. The Pillai test statistic, approximate F value, df and p value match what we produced in SPSS. This test does not have the sphericity assumption. The multivariate test is evaluated as the degree of improvement in fiit (or reduction in residuals) in the full model (mlmfit.1) relative to the intercept-only model (mlmfit.0).

```
anova(mlmfit.1, mlmfit.0, X=~1)
```

```
## Analysis of Variance Table
##
## Model 1: rpt1w.mat ~ 1
## Model 2: rpt1w.mat ~ 1 - 1
##
## Contrasts orthogonal to
## ~1
##
##   Res.Df Df Gen.var.  Pillai approx F num Df den Df Pr(>F)
## 1      9      9.6149
## 2     10  1  11.2467 0.64954   2.7801      4      6 0.1269
```

The Mauchly test can be performed on the mlmfit.1 object. The W test statistic and p value match our SPSS MANOVA results.

```
mauchly.test(mlmfit.1, X=~1)
```

```
##
```

```
##  Mauchly's test of sphericity
##  Contrasts orthogonal to
##  ~1
##
##
## data:  SSD matrix from lm(formula = rpt1w.mat ~ 1)
## W = 0.0038542, p-value = 7.391e-06
```

The univariate averaged F test can also be produced by adding the "test" argument. This produces the same F value as we derived with the standard ANOVA approach in SPSS MANOVA, the approach that assumes sphericity. Note that the GG and HF epsilons are produced and that the adjustment produces the same p values as we obtained in MANOVA.

```
anova(mlmfit.1, mlmfit.0, X=~1, test="Spherical")
```

```
## Analysis of Variance Table
##
## Model 1: rpt1w.mat ~ 1
## Model 2: rpt1w.mat ~ 1 - 1
##
## Contrasts orthogonal to
## ~1
##
## Greenhouse-Geisser epsilon: 0.3793
## Huynh-Feldt epsilon:        0.4401
##
##   Res.Df Df Gen.var.      F num Df den Df     Pr(>F)    G-G Pr   H-F Pr
## 1      9      9.6149
## 2     10  1  11.2467 8.8447      4     36 4.4236e-05 0.0055009 0.003391
```

While Method II gives the multivariate test and provides another way to obtain the traditional univariate F test along with GG and HF corrections (as well as the Mauchly test) it has a downside. The model specification formulae are difficult arguments for the relative novice at R programming, especially the ~1 and ~0 types of structures. There is also not a simple way to extract tests of contrasts with this approach, that I have found. Method III uses a similar underlying logic, but is simpler to implement.

3.1.3 Method III, also a Multivariate Linear Model but using `Anova` from car

Method III is similar to Method II, but uses an `Anova` approach in John Fox's car package that permits the Mauchly test and the GG and HF corrections as well as permitting Type III sums of squares (which lm and anova does not. However, Type III SS is not relevant here - since there are no missing data points, the data set is balanced. But this approach can be useful for advanced

designs that contain both repeated measures factors and between-groups factors as well as unequal N.

```
# use the same "wide" data frame as from Method II
# change the data from a data frame to a matrix
# and leave out the snum variable
# this method requires that the matrix only contains the data values
rpt1w.mat <- as.matrix(rpt1w.df[,-1])
# view the data
rpt1w.mat
```

```
##       clean homecage iguana whiptail krat
##  [1,]    24       15     41       30   50
##  [2,]     6        6      0        6   13
##  [3,]     4        0      5        4    9
##  [4,]    11        9     10       14   18
##  [5,]     0        0      0        0    0
##  [6,]     8       15     10       15   38
##  [7,]     8        5      2        6   15
##  [8,]     0        0      0       11   54
##  [9,]     0        3      1        1   11
## [10,]     7        7      4        7   23
```

```
# start by defining the multivariate linear model
# this code returns the level means.
mlmfit.2 <- lm(rpt1w.mat ~1)
mlmfit.2
```

```
##
## Call:
## lm(formula = rpt1w.mat ~ 1)
##
## Coefficients:
##             clean  homecage  iguana  whiptail  krat
## (Intercept)   6.8    6.0       7.3     9.4      23.1
```

The completion of the analysis requires the creation of the repeated measures factor. The data set, in the wide format, is simply a collection of five variables. The "type" variable used in Method I does not exist. So we need to create an object that is that five-level factor. The factor is also "ordered" to keep the levels in the same order as was done above for graphing purposes, but also to have the upcoming contrast analysis match what we did in SPSS MANOVA. The creation of this "stimulus" factor (really the same thing as "type" in the long format data set) is analogous to using the "wsfactors" subcommand in SPSS MANOVA and GLM. My choice of the word "stimulus" is shorthand for "stimulus type" and this reinforces the identity to the "type" variable as used in Method I.

```
# now define a variable that gives the design of the study,
# the levels of the repeated factor, and order them the
# same way that we did in the original SPSS analyses
stimulus <- factor(c("clean", "homecage", "iguana" ,
                     "whiptail", "krat"))
stimulus  <- ordered(stimulus,
                     levels=c("clean","homecage","iguana",
                              "whiptail", "krat"))
stimulus
```

```
## [1] clean    homecage iguana   whiptail krat
## Levels: clean < homecage < iguana < whiptail < krat
```

The implementation of the analysis with this approach focuses on use of the `Anova` function from the **car** package. Two crucial arguments are required to specify the repeated measures nature of the analysis of the design. These are the "idata" and "idesign" arguments. `Anova` uses the multivariate object defined above and then applies the idesign specification. Applying `summary` to that `Anova` object produces the multivariate and univariate tests of the omnibus hypothesis. It also provides the Mauchly test of sphericity, the GG and HF epsilons, and corrected univariate test p-values, all matching the Method II and SPSS results. The terms "idesign", "idata", and "icontrasts" derive from the phrase "Intra-subject" which implies repeated measures.

When using the idesign argument, `Anova` transforms the levels of the repeated measures factors into orthogonal contrasts. Look carefully at the product of the `summary` function here and see if you can guess what this default set of contrasts is. Hint: it is the default because the most common type of repeated measure factor is a collection of measurements at various time points, and is thus a quantitative IV.

There may be something that I am missing about using this method, but I can't sort out a direct way of testing the contrasts. This is in spite of the fact that their SS are reported. We will return to this with a manual method in a later section of the next chapter.

```
#require(car)
mlmfit2.anova <- Anova(mlmfit.2,
                       idata=data.frame(stimulus),
                       idesign=~stimulus, type="III")
# with the summary function, when multivariate=TRUE,
# and univariate=TRUE,
# both multivariate and averaged F tests of the
# repeated effect are printed.
summary(mlmfit2.anova, multivariate=T, univariate=T)
```

```
##
## Type III Repeated Measures MANOVA Tests:
```

```
##
## ----------------------------------------
##
## Term: (Intercept)
##
##  Response transformation matrix:
##          (Intercept)
## clean              1
## homecage           1
## iguana             1
## whiptail           1
## krat               1
##
## Sum of squares and products for the hypothesis:
##             (Intercept)
## (Intercept)     27667.6
##
## Multivariate Tests: (Intercept)
##                  Df test stat approx F num Df den Df    Pr(>F)
## Pillai            1 0.5967217  13.3171      1      9 0.0053237 **
## Wilks             1 0.4032783  13.3171      1      9 0.0053237 **
## Hotelling-Lawley  1 1.4796774  13.3171      1      9 0.0053237 **
## Roy               1 1.4796774  13.3171      1      9 0.0053237 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## ----------------------------------------
##
## Term: stimulus
##
##  Response transformation matrix:
##          stimulus.L stimulus.Q    stimulus.C stimulus^4
## clean    -0.6324555  0.5345225 -3.162278e-01  0.1195229
## homecage -0.3162278 -0.2672612  6.324555e-01 -0.4780914
## iguana    0.0000000 -0.5345225 -4.095972e-16  0.7171372
## whiptail  0.3162278 -0.2672612 -6.324555e-01 -0.4780914
## krat      0.6324555  0.5345225  3.162278e-01  0.1195229
##
## Sum of squares and products for the hypothesis:
##            stimulus.L stimulus.Q stimulus.C stimulus^4
## stimulus.L  1296.0000   906.6815  342.00000  164.64132
## stimulus.Q   906.6815   634.3143  239.26317  115.18306
## stimulus.C   342.0000   239.2632   90.25000   43.44702
## stimulus^4   164.6413   115.1831   43.44702   20.91571
##
## Multivariate Tests: stimulus
```

```
##                   Df test stat approx F num Df den Df  Pr(>F)
## Pillai             1 0.6495405 2.780095      4      6 0.12692
## Wilks              1 0.3504595 2.780095      4      6 0.12692
## Hotelling-Lawley   1 1.8533965 2.780095      4      6 0.12692
## Roy                1 1.8533965 2.780095      4      6 0.12692
##
## Univariate Type III Repeated-Measures ANOVA Assuming Sphericity
##
##             Sum Sq num Df Error SS den Df F value    Pr(>F)
## (Intercept) 5533.5      1   3739.7      9 13.3171  0.005324 **
## stimulus    2041.5      4   2077.3     36  8.8447 4.424e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Mauchly Tests for Sphericity
##
##          Test statistic    p-value
## stimulus      0.0038542 7.3907e-06
##
##
## Greenhouse-Geisser and Huynh-Feldt Corrections
##  for Departure from Sphericity
##
##          GG eps Pr(>F[GG])
## stimulus 0.3793   0.005501 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##             HF eps  Pr(>F[HF])
## stimulus 0.4400688 0.003391046
```

The multivariate tests match what was seen in Method II above, as does the Univariate F test which also matches the F test seen with Method I. The "Sum of squares and products for the hypothesis" matrix provides a starting point for evaluating contrasts and this will be developed in the next chapter.

3.1.4 Method IV: Implement the univariate analysis with `ezanova` from the ez package

The `ezanova` function is a wrapper for the `aov` function and may be simpler to use than the `aov` function for some designs. It has the added value of performing the Mauchly sphericity test and providing GG and HF epsilons and corrected p values as well as an effect size indicator. The structure of the code requires an argument for the data frame (needs the long format data frame), the dependent variable, the ID factor for the case variable (subject, or snum in our data set),

and the repeated measures factor (the within argument).

This might be the simplest of all approaches to obtain the omnibus F test plus the sphericity test and associated corrections. The main arguments that create the repeated measures analysis are "dv" which is the dependent variable name, "wid" which is the subject variable ID code (and the period is a odd but necessary part of the code) and "within" which is the repeated measure IV.

```
#library(ez)
fit1.ez <- ezANOVA(data=rpt1.df, detailed=T, return_aov=T,
                   dv=DV,
                   wid=.(snum),
                   within=.(type))
fit1.ez
```

```
## $ANOVA
##        Effect DFn DFd     SSn     SSd         F            p p<.05       ges
## 1 (Intercept)   1   9 5533.52 3739.68 13.317097 5.323652e-03     * 0.4875125
## 2        type   4  36 2041.48 2077.32  8.844723 4.423582e-05     * 0.2597805
##
## $`Mauchly's Test for Sphericity`
##   Effect           W            p p<.05
## 2   type 0.003854226 7.390741e-06     *
##
## $`Sphericity Corrections`
##   Effect       GGe       p[GG] p[GG]<.05       HFe       p[HF] p[HF]<.05
## 2   type 0.3792955 0.005500897         * 0.4400688 0.003391046         *
##
## $aov
##
## Call:
## aov(formula = formula(aov_formula), data = data)
##
## Grand Mean: 10.52
##
## Stratum 1: snum
##
## Terms:
##                 Residuals
## Sum of Squares    3739.68
## Deg. of Freedom         9
##
## Residual standard error: 20.38431
##
## Stratum 2: snum:type
##
## Terms:
```

```
##                     type Residuals
## Sum of Squares  2041.48   2077.32
## Deg. of Freedom       4        36
##
## Residual standard error: 7.596271
## Estimated effects may be unbalanced
```

3.1.5 Method V: Implement the univariate analysis with the afex package

The **afex** package permits specification of repeated measures models using styles from **ez**, `aov`, `lm`, `car::Anova` and other modeling functions. Here, the code reflects the basic `aov` syntax for the model specification and uses `Anova` from **car** for calculations.

This initial approach requests the base univariate/averaged F approach, first setting the contrasts to the "sum to zero" contrast set that should be done for ANOVA designs, especially repeated measures - orthogonal contrasts are also "sum to zero" contrasts and will be examined in a later chapter. Here, the `contr.sum` specification sets the contrasts to "effect" coding.

```
contrasts(rpt1.df$type) <- contr.sum
fit1.afex <- aov_car(DV~type + Error(snum/type), data=rpt1.df)
summary(fit1.afex)
```

```
##
## Univariate Type III Repeated-Measures ANOVA Assuming Sphericity
##
##             Sum Sq num Df Error SS den Df F value    Pr(>F)
## (Intercept) 5533.5      1   3739.7      9 13.3171  0.005324 **
## type        2041.5      4   2077.3     36  8.8447 4.424e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Mauchly Tests for Sphericity
##
##      Test statistic    p-value
## type      0.0038542 7.3907e-06
##
##
## Greenhouse-Geisser and Huynh-Feldt Corrections
##  for Departure from Sphericity
##
##      GG eps Pr(>F[GG])
## type 0.3793   0.005501 **
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##         HF eps  Pr(>F[HF])
## type 0.4400688 0.003391046
```

The inital approach does not produce an effect size estimate. One way to obtain the "generalized effect size" statistic is to simply ask to see the afex fit object, but as a stand alone method this may not be desireable since it only reports the GG corrected p value.

```
fit1.afex
```

```
## Anova Table (Type 3 tests)
##
## Response: DV
##   Effect          df    MSE       F  ges p.value
## 1   type 1.52, 13.65 152.13 8.84 ** .260    .006
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '+' 0.1 ' ' 1
##
## Sphericity correction method: GG
```

A more full way of taking control is to use the "anova_table" argument in the original `aov_car` specification. The correction argument in the "anova_table" list permits "none", "GG" or "HF" for sphericity corrections. Simply asking for a printout of the fit2.afex object also yields the corrected p value (or uncorrected as is requested here) and an effect size statistic. The options for the "es" specification are "pes" for partial eta squared, and "ges" for the generalized effect size. See the help pages for the **afex** functions for references regarding the generalized effect size statistic if that has not yet been covered in class.

```
# make sure "sum to zero" contrasts are specified - done above
fit2.afex <- aov_car(DV~type + Error(snum/type), data=rpt1.df,
                     anova_table = list(correction = "none", es="pes"))
fit2.afex
```

```
## Anova Table (Type 3 tests)
##
## Response: DV
##   Effect    df   MSE        F  pes p.value
## 1   type 4, 36 57.70 8.84 *** .496   <.001
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '+' 0.1 ' ' 1
```

An example of using one of the correction methods and a choice of the "ges" effect size is next:

```
# make sure "sum to zero" contrasts are specified - done above
fit3.afex <- aov_car(DV~type + Error(snum/type), data=rpt1.df,
```

```
                    anova_table = list(correction = "HF", es="ges"))
fit3.afex
```

```
## Anova Table (Type 3 tests)
##
## Response: DV
##   Effect          df    MSE       F  ges p.value
## 1   type 1.76, 15.84 131.12 8.84 ** .260    .003
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '+' 0.1 ' ' 1
##
## Sphericity correction method: HF
```

Neither of these previous two code chunks provide the sphericity test. That can be done by applying the `summary` function to the `aov_car` objects. For example…..:

```
summary(fit3.afex)
```

```
##
## Univariate Type III Repeated-Measures ANOVA Assuming Sphericity
##
##             Sum Sq num Df Error SS den Df F value    Pr(>F)
## (Intercept) 5533.5      1   3739.7      9 13.3171  0.005324 **
## type        2041.5      4   2077.3     36  8.8447 4.424e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Mauchly Tests for Sphericity
##
##      Test statistic    p-value
## type      0.0038542 7.3907e-06
##
##
## Greenhouse-Geisser and Huynh-Feldt Corrections
##  for Departure from Sphericity
##
##      GG eps Pr(>F[GG])
## type 0.3793   0.005501 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##         HF eps  Pr(>F[HF])
## type 0.4400688 0.003391046
```

Finally, we can examine use of the other **afex** functions that provide the same

general approach, but uses the style of model specification from the 'ez' package. First, the "ez" style:

```
# make sure "sum to zero" contrasts are specified - done above
fit4.afex <- aov_ez(data=rpt1.df, detailed=T, return_aov=T,
                    dv="DV",
                    id="snum",
                    within="type",
                      anova_table = list(correction = "HF", es="ges"))
fit4.afex
```

```
## Anova Table (Type 3 tests)
##
## Response: DV
##   Effect          df    MSE       F  ges p.value
## 1   type 1.76, 15.84 131.12 8.84 ** .260    .003
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '+' 0.1 ' ' 1
##
## Sphericity correction method: HF
```

```
summary(fit4.afex)
```

```
##
## Univariate Type III Repeated-Measures ANOVA Assuming Sphericity
##
##             Sum Sq num Df Error SS den Df F value    Pr(>F)
## (Intercept) 5533.5      1   3739.7      9 13.3171  0.005324 **
## type        2041.5      4   2077.3     36  8.8447 4.424e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Mauchly Tests for Sphericity
##
##      Test statistic    p-value
## type      0.0038542 7.3907e-06
##
##
## Greenhouse-Geisser and Huynh-Feldt Corrections
##  for Departure from Sphericity
##
##      GG eps Pr(>F[GG])
## type 0.3793   0.005501 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
##         HF eps  Pr(>F[HF])
## type 0.4400688 0.003391046
```

And next, the "lme" style:

```
# make sure "sum to zero" contrasts are specified - done above
fit5.afex <- aov_4(DV ~ type  + (type|snum), data=rpt1.df,
                   anova_table = list(correction = "none", es="ges"))
fit5.afex
```

```
## Anova Table (Type 3 tests)
##
## Response: DV
##   Effect    df   MSE        F  ges p.value
## 1   type 4, 36 57.70 8.84 *** .260   <.001
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '+' 0.1 ' ' 1
```

```
summary(fit5.afex)
```

```
##
## Univariate Type III Repeated-Measures ANOVA Assuming Sphericity
##
##             Sum Sq num Df Error SS den Df F value    Pr(>F)
## (Intercept) 5533.5      1   3739.7      9 13.3171  0.005324 **
## type        2041.5      4   2077.3     36  8.8447 4.424e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Mauchly Tests for Sphericity
##
##      Test statistic    p-value
## type      0.0038542 7.3907e-06
##
##
## Greenhouse-Geisser and Huynh-Feldt Corrections
##  for Departure from Sphericity
##
##      GG eps Pr(>F[GG])
## type 0.3793   0.005501 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##         HF eps  Pr(>F[HF])
## type 0.4400688 0.003391046
```

This implementation with **afex** is probably the simplest way to accomplish the

primary requirements of finding the omnibus F test, evaluating the sphericity assumption, correcting the p value for any sphericity violations, and providing an effect size estimate. The approach taken for fit2.afex, along with use of the summary function on that object strikes me as the most efficient way of doing the traditional omnibus ANOVA. Objects from these **afex** functions also provide a direct way to interface with the **emmeans** approach to contrasts and post hoc testing found in a later chapter.

3.2 Commentary on the Univariate/Multivariate methods

The user has two kinds of decisions to make. The first is whether the multivariate GLM approach is desired, or the univariate one (the averaged F test). Method II and Method III produce both types of test and the choice depends, in part on examination of the sphericity assumption. The real question is whether to use the multivariate test or the GG/HF corrected univariate approach. Typically researchers have not frequently used the multivariate approach because of its relative low power for the typical small sample sizes of many repeated measures experiments. A literature on the relative merits of the two approaches can be found in your textbooks and the Dudek stat toolkit bibliography.

Of all the methods, the simplest to implement might be method V, using **afex**. This provides the sphericity test and GG/HF corrections to the univariate p value for the omnibus test. If the user does not prefer the corrected univariate test in the face of non-sphericity, then Method I, II or Method III can be employed once the sphericity assumption is examined (from Method II or III).

Chapter 4

Analytical Contrasts

Once first decision is made regarding univariate (the so-called averaged F test) vs multivariate tests, the user must still decide how to proceed in following up these omnibus tests with pairwise comparisons or analytical/orthogonal contrasts. It can be argued that the omnibus F tests in ANOVA are largely superfluous to the goals of researchers. Studies are designed with specific a priori hypotheses that should/could be specified as contrasts, including potential multi-group comparisons. This approach is viewed as more elegant and flexible that a shotgun approach of doing all pairwise comparisons among levels. Although assessment of pairwise comparisons among levels of a repeated measures factor are covered in this document, the analytical (and perhaps orthogonal) contrasts approach is considered the method of choice.

Until recently, and unfortunately, most of the built-in methods I have found employ the omnibus Axs error term in creating standard errors or F tests for such comparisons. This error term will almost always be flawed because of the likely lack of sphericity that is common in repeated measures studies. This is what we saw above in the attempt to use the "split" argument in the summary of the `aov` fit in the first part of this chapter.

The theory that outlines the rationale for specific error terms is summarized in the following figure which envisions partitioning the omnibus effect into contrasts and the Axs interaction into interaction contrasts which are the specific error terms for their respective contrast:

Full Table with Tests of Contrasts on Factor A:

Source		df	EMS	Exact Error Term
s		s-1	$\sigma_E^2 + a\sigma_s^2$	-
A		a-1	$\sigma_E^2 + s\theta_A^2 + \sigma_{Axs}^2$	MS_{Axs}
	A_{C1}	1	$\sigma_E^2 + s\theta_{Ac1}^2 + \sigma_{Ac1\,x\,s}^2$	$MS_{Ac1\,x\,s}$
	.	.	.	.
	.	.	.	.
	.	.	.	.
	A_{Cj}	1	$\sigma_E^2 + s\theta_{Acj}^2 + \sigma_{Acj\,x\,s}^2$	$MS_{Acj\,x\,s}$
A x s		(a-1)(s-1)	$\sigma_E^2 + \sigma_{Axs}^2$	
	$A_{C1\,x\,s}$	(1)(s-1)	$\sigma_E^2 + \sigma_{Ac1\,x\,s}^2$	
	.	.	.	
	.	.	.	
	.	.	.	
	$A_{Cj\,x\,s}$	(1)(s-1)	$\sigma_E^2 + \sigma_{Acj\,x\,s}^2$	

Note that when the homogeneity of covariance/sphericity assumption is satisfied, then the different $A_{Cj\,x\,s}$ sources (the specific error terms) are all numerically equivalent.

These specific error terms are not flawed in the presence of non-sphericity. They can also be viewed as "tailored" to the contrast in question and may be generally preferred. In the rare circumstance where sphericity can be confidently assumed, then use of the omnibus error term would provide more power (larger df). Otherwise, specific error terms should be seen as the preferred approach.

We will see how use of the **emmeans** enables hypothesis tests with specific error terms, but only for some ANOVA fit objects. This is outlined in the section on **afex** and **emmeans** usage toward the end of this chapter.

It is possible to obtain specific error terms are with the manual approaches outlined below or perhaps in following up linear mixed effects modeling (next chapter). The manual approaches are not terribly burdensome for a simple one factor repeated measures design, but will become very much more challenging for larger designs - e.g., two repeated factors or mixed designs with grouping factors and repeated measures factors.

The conclusion I still hold is that the SPSS MANOVA procedure is efficient and helpful in pursuing these followup contrast types of questions. The SPSS GLM procedure can also be used, but its syntax (with L matrix specification) is obnoxiously obtuse. The default setting of specific error terms is of considerable value in the MANOVA procedure. It is not clear to me why this has not found its way into the R ecosystem more prominently. It is perhaps because the standard data format for repeated measures is the long format data structure. Specific

error terms are more easily generated out of wide format data structures, for repeated measures. Perhaps there is still something I haven't yet learned about the suite of R functions available. Or see the use of the `glht` function usage in the last sections in the next chapter here.

4.1 Partitioning the omnibus effect into orthogonal contrasts

The value of full rank models that employ orthogonal contrast sets is clear for completely randomized designs (between-groups designs). For repeated measures factors partitioning into orthogonal sets is useful for establishing the Omnibus F test of the factor as well. Non-orthogonal contrasts with alpha rate adjustments can also be useful and attention that that approach is possible with the `emmeans` function later in this chapter and the `glht` function in the next chapter.

4.2 Contrasts with the `aov` model object

Rather than use the default indicator coding or the "contr.sum" coding that was used initially, we can change the contrasts for the repeated measure factor using the same approach considered for between subjects factors in completely randomized designs. The set chosen here would likely contain several a priori hypotheses about this design and all four contrasts are rational.

```
contrasts.type <- matrix(c(4, -1, -1, -1, -1,
                           0,  3, -1, -1, -1,
                           0,  0, -1, -1,  2,
                           0,  0, -1,  1,  0),
                           ncol=4)
contrasts(rpt1.df$type) <- contrasts.type
contrasts(rpt1.df$type)
```

```
##           [,1] [,2] [,3] [,4]
## clean        4    0    0    0
## homecage    -1    3    0    0
## iguana      -1   -1   -1   -1
## whiptail    -1   -1   -1    1
## krat        -1   -1    2    0
```

Once this set is in place, we can use familiar methods for finding tests of these contrasts. The same `aov` model fit that comprised Method I above is implemented again here. In a second application of the `summary` function to the `aov` model object, we can use the "split" argument to obtain evaluation of these four orthogonal contrasts.

From examination of the four F values and their component numerator and

denominator MS, we can see that these four tests all use the OMNIBUS error term and are not GG/HF adjusted. Specific error terms are, unfortunately, not possible using the `aov` function for the model fit. `The summary.lm` function also will not work with a repeated measures object fit from `aov`. Other methods for obtaining evaluation of contrasts are found below.

```
fit.1 <- aov(DV~type + Error(snum/type), data=rpt1.df)
summary(fit.1)
```

```
##
## Error: snum
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals  9   3740   415.5
##
## Error: snum:type
##           Df Sum Sq Mean Sq F value   Pr(>F)
## type       4   2042   510.4   8.845 4.42e-05 ***
## Residuals 36   2077    57.7
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(fit.1, split=list(type=list(ac1=1, ac2=2, ac3=3, ac4=4)))
```

```
##
## Error: snum
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals  9   3740   415.5
##
## Error: snum:type
##             Df Sum Sq Mean Sq F value   Pr(>F)
## type         4 2041.5   510.4   8.845 4.42e-05 ***
##   type: ac1  1  173.0   173.0   2.998   0.0919 .
##   type: ac2  1  396.0   396.0   6.863   0.0128 *
##   type: ac3  1 1450.4  1450.4  25.136 1.44e-05 ***
##   type: ac4  1   22.1    22.1   0.382   0.5404
## Residuals   36 2077.3    57.7
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

4.3 Contrasts with the Method III "mlmfit" object

We saw in the earlier chapter that contrast SS could be obtained from the "mlmfit" approach. Since orthogonal polynomial trend analysis, as done in the initial Method III analysis, is not appropriate for our "stimulus/type" factor, next we can add in the ability to specify our own contrasts. This is the same orthogonal

set used just above. The name of the factor was changed to "stimulus" for this approach so as to not confuse it with the `aov` approach where we used the variable name "type" in the long format data frame. Then we will rerun the `Anova` function from car, using these contrasts.

```
contrasts.stimulus <- matrix(c(4, -1, -1, -1, -1,
                               0,  3, -1, -1, -1,
                               0,  0, -1, -1,  2,
                               0,  0, -1,  1,  0),
                             ncol=4)
contrasts(stimulus) <- contrasts.stimulus
contrasts(stimulus)
```

```
##          [,1] [,2] [,3] [,4]
## clean       4    0    0    0
## homecage   -1    3    0    0
## iguana     -1   -1   -1   -1
## whiptail   -1   -1   -1    1
## krat       -1   -1    2    0
```

```
icontr <- contrasts(stimulus)
```

First we will refit the same "mlmfit" model as from Method III in the earlier chapter.

```
# start by defining the multivariate linear model
# this code returns the level means.
mlmfit.2 <- lm(rpt1w.mat ~1)
mlmfit.2
```

```
##
## Call:
## lm(formula = rpt1w.mat ~ 1)
##
## Coefficients:
##             clean  homecage  iguana  whiptail  krat
## (Intercept)   6.8    6.0       7.3     9.4      23.1
```

Next we redo the application of the `Anova` function to that model fit with an additional specification. The new "icontr" object is specified with one additional argument in `Anova`, called "icontrasts".

Also notice:

1. Hypothesis SS and for each contrast are produced, but only found in the SSP matrix and t/F tests not performed. We will have to compute the F's manually after computing the respective MS.
2. The error SS are computed for the contrasts but not printed by use of the `summary` function. We will have to manually extract them.

3. The hypothesis SS and error SS are not corrected for the size of the coefficients chosen. I.e., the coefficients are not orthonormalized in this illustration, so the SS won't match your SPSS-generated SS unless you divide each by the respective sum of the squared contrast coefficients.

4. Despite using the contrasts that we specified, `Anova` does not produce tests of those individual contrasts with the `summary` function. I have not found a direct way to obtain those tests. In the next sections, I show ways to obtain them more "manually"

```
mlmfit3.anova <- Anova(mlmfit.2,
                       idata=data.frame(stimulus),
                       idesign=~stimulus,
                       icontrasts=icontr,type="III")
#str(mlmfit3.anova)
summary(mlmfit3.anova)
```

```
##
## Type III Repeated Measures MANOVA Tests:
##
## ----------------------------------------
##
## Term: (Intercept)
##
##  Response transformation matrix:
##          (Intercept)
## clean              1
## homecage           1
## iguana             1
## whiptail           1
## krat               1
##
## Sum of squares and products for the hypothesis:
##             (Intercept)
## (Intercept)     27667.6
##
## Multivariate Tests: (Intercept)
##                  Df test stat approx F num Df den Df    Pr(>F)
## Pillai            1 0.5967217  13.3171      1      9 0.0053237 **
## Wilks             1 0.4032783  13.3171      1      9 0.0053237 **
## Hotelling-Lawley  1 1.4796774  13.3171      1      9 0.0053237 **
## Roy               1 1.4796774  13.3171      1      9 0.0053237 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## ----------------------------------------
##
```

```
## Term: stimulus
##
##  Response transformation matrix:
##          stimulus1 stimulus2 stimulus3 stimulus4
## clean            4         0         0         0
## homecage        -1         3         0         0
## iguana          -1        -1        -1        -1
## whiptail        -1        -1        -1         1
## krat            -1        -1         2         0
##
## Sum of squares and products for the hypothesis:
##           stimulus1 stimulus2 stimulus3 stimulus4
## stimulus1    3459.6    4054.8   -5487.0    -390.6
## stimulus2    4054.8    4752.4   -6431.0    -457.8
## stimulus3   -5487.0   -6431.0    8702.5     619.5
## stimulus4    -390.6    -457.8     619.5      44.1
##
## Multivariate Tests: stimulus
##                  Df test stat approx F num Df den Df  Pr(>F)
## Pillai            1 0.6495405 2.780095      4      6 0.12692
## Wilks             1 0.3504595 2.780095      4      6 0.12692
## Hotelling-Lawley  1 1.8533965 2.780095      4      6 0.12692
## Roy               1 1.8533965 2.780095      4      6 0.12692
##
## Univariate Type III Repeated-Measures ANOVA Assuming Sphericity
##
##             Sum Sq num Df Error SS den Df F value    Pr(>F)
## (Intercept) 5533.5      1   3739.7      9 13.3171  0.005324 **
## stimulus    2041.5      4   2077.3     36  8.8447 4.424e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Mauchly Tests for Sphericity
##
##          Test statistic    p-value
## stimulus      0.0038542 7.3907e-06
##
##
## Greenhouse-Geisser and Huynh-Feldt Corrections
##  for Departure from Sphericity
##
##          GG eps Pr(>F[GG])
## stimulus 0.3793   0.005501 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
##             HF eps  Pr(>F[HF])
## stimulus 0.4400688 0.003391046
```

The omnibus univariate and multivariate tests are identical the the initial Method III illustration above. Unfortunately, there does not appear to be a way of requesting that `Anova` test these newly created contrasts.

The output just above did give the SS for each of the contrasts, seen on the leading diagonal of the SSP (hypothesis matrix). The `Anova` function did create the error SS as well, but they were not printed with the results. In order to find the error SS for each contrast (the Acontrast x s) terms for specific error terms, we need to examine the structure of the mlmfit3.anova object. It is called the SSPE matrix. Note that in addition to the SShypothesis and SSerror values, we can look for the error df as well (it is called error.df)

```
str(mlmfit3.anova)
```

```
## List of 14
##  $ SSP       :List of 2
##   ..$ (Intercept): num [1, 1] 27668
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr "(Intercept)"
##   .. .. ..$ : chr "(Intercept)"
##   ..$ stimulus   : num [1:4, 1:4] 3460 4055 -5487 -391 4055 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:4] "stimulus1" "stimulus2" "stimulus3" "stimulus4"
##   .. .. ..$ : chr [1:4] "stimulus1" "stimulus2" "stimulus3" "stimulus4"
##  $ SSPE      :List of 2
##   ..$ (Intercept): num [1, 1] 18698
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr "(Intercept)"
##   .. .. ..$ : chr "(Intercept)"
##   ..$ stimulus   : num [1:4, 1:4] 4976 4381 -5133 -169 4381 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:4] "stimulus1" "stimulus2" "stimulus3" "stimulus4"
##   .. .. ..$ : chr [1:4] "stimulus1" "stimulus2" "stimulus3" "stimulus4"
##  $ P         :List of 2
##   ..$ (Intercept): num [1:5, 1] 1 1 1 1 1
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:5] "clean" "homecage" "iguana" "whiptail" ...
##   .. .. ..$ : chr "(Intercept)"
##   ..$ stimulus   : num [1:5, 1:4] 4 -1 -1 -1 -1 0 3 -1 -1 -1 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:5] "clean" "homecage" "iguana" "whiptail" ...
##   .. .. ..$ : chr [1:4] "stimulus1" "stimulus2" "stimulus3" "stimulus4"
##  $ df        : Named num [1:2] 1 1
```

```
##   ..- attr(*, "names")= chr [1:2] "(Intercept)" "stimulus"
##  $ error.df  : int 9
##  $ terms     : chr [1:2] "(Intercept)" "stimulus"
##  $ repeated  : logi TRUE
##  $ type      : chr "III"
##  $ test      : chr "Pillai"
##  $ idata     :'data.frame':  5 obs. of  1 variable:
##   ..$ stimulus: Ord.factor w/ 5 levels "clean"<"homecage"<..: 1 2 3 4 5
##   .. ..- attr(*, "contrasts")= num [1:5, 1:4] 4 -1 -1 -1 -1 0 3 -1 -1 -1 ...
##   .. .. ..- attr(*, "dimnames")=List of 2
##   .. .. .. ..$ : chr [1:5] "clean" "homecage" "iguana" "whiptail" ...
##   .. .. .. ..$ : NULL
##  $ idesign   :Class 'formula'  language ~stimulus
##   .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
##  $ icontrasts: num [1:5, 1:4] 4 -1 -1 -1 -1 0 3 -1 -1 -1 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : chr [1:5] "clean" "homecage" "iguana" "whiptail" ...
##   .. ..$ : NULL
##  $ imatrix   : NULL
##  $ singular  : Named logi [1:2] FALSE FALSE
##   ..- attr(*, "names")= chr [1:2] "(Intercept)" "stimulus"
##  - attr(*, "class")= chr "Anova.mlm"
```

This SSP matrix duplicates the information that was seen in the `summary` output and the SS for the contrasts are on the leading diagonal. They are numerically larger than those from the `summary` output above, but by the expected factor of the sum of the squared coefficients for each contrast - they are not orthonormalized.

```
# view the contrast hypothesis SS_CP matrix
mlmfit3.anova$SSP$stimulus
```

```
##           stimulus1 stimulus2 stimulus3 stimulus4
## stimulus1    3459.6    4054.8   -5487.0    -390.6
## stimulus2    4054.8    4752.4   -6431.0    -457.8
## stimulus3   -5487.0   -6431.0    8702.5     619.5
## stimulus4    -390.6    -457.8     619.5      44.1
```

Similarly, the SS for the specific errors are found on the leading diagonal of this SSPE matrix - again, not orthonormalized contrasts.

```
# view the contrast error SS_CP matrix
mlmfit3.anova$SSPE$stimulus
```

```
##           stimulus1 stimulus2 stimulus3 stimulus4
## stimulus1    4976.4    4381.2   -5133.0    -169.4
## stimulus2    4381.2    6371.6   -4069.0     369.8
## stimulus3   -5133.0   -4069.0    6882.5     735.5
```

```
## stimulus4    -169.4     369.8     735.5     300.9
```

In order to do the F tests, we need to:

1. Extract the relevant SS from the leading diagonals of the matrices examined above.
2. Create the MS error for each contrast by dividing by the correct df for the Axs term (1x(s-1))
3. Realize that each contrast is a 1df source, so each hypothesis SS is already a MS
4. Divide the MSeffect values by the MSerror values to produce the F values
5. find p values for each F

It made the naming easier to "attach" the mlmfit3.anova object. First, obtain the SSeffect values (equivalent to the MSeffect values since they are 1 df terms:

```
attach(mlmfit3.anova)
#show effect/cov values - the SS of the contrasts are on the diagonal
#SSP$stimulus
effect <- diag(SSP$stimulus)
#note that the diagonal from SSP matrix is already the MS since df = 1
effect
```

```
## stimulus1 stimulus2 stimulus3 stimulus4
##    3459.6    4752.4    8702.5      44.1
```

Then obtain the MSerror values:

```
#show error/cov values - the specific error SS are on the diagonal
#SSPE$stimulus
x1 <- diag(SSPE$stimulus)
x1
```

```
## stimulus1 stimulus2 stimulus3 stimulus4
##    4976.4    6371.6    6882.5     300.9
```

```
error <- x1/error.df
```

Now compute the F values and obtain p values for each contrast. Note that they match our SPSS work.

```
# now compute F's; F values are on the diagonal
Fvalues <- effect/error
Fvalues
```

```
## stimulus1 stimulus2 stimulus3 stimulus4
##  6.256812  6.712851 11.379949  1.319043
```

```
# obtain the p values for the four F tests (all have 1,9 df)
pf(Fvalues,1,9,lower.tail=F)
```

```
##   stimulus1   stimulus2   stimulus3   stimulus4
```

```
## 0.033786236 0.029169567 0.008212304 0.280372227
```

Once again, note that the MS values for the numerator and denominator are neither one corrected for the size of the coefficients chosen. But since neither are corrected, their ratio still produces the expected F value. We could have done more work to divide each SS first by the sum of the squared coefficients for each contrast, but the same F values would have been produced.

There is a more efficient way of doing this by writing a function and then passing the `Anova` object to it. F values and p values are returned.

```
model <- mlmfit3.anova
rptcontr <- function(model){
  SSCPeffect <- as.matrix(model$SSP$stimulus,ncol=4)
  effect <- diag(SSCPeffect)
  #effect
  #return(effect)
  SSCPerror <- as.matrix(model$SSPE$stimulus,ncol=4)
  error <- diag(SSCPerror)
  dferror <- mlmfit3.anova$error.df
  mserror <- error/dferror
  Fval <- round(effect/mserror,3)
  pval <- round(pf(Fval, 1,dferror, lower.tail=F),4)
  vals <- as.data.frame(cbind(Fval, pval, dferror))
  vals
  return(gt(vals,rownames_to_stub=T))
}
rptcontr(model=mlmfit3.anova)
```

	Fval	pval	dferror
stimulus1	6.257	0.0338	9
stimulus2	6.713	0.0292	9
stimulus3	11.380	0.0082	9
stimulus4	1.319	0.2804	9

4.4 Recall how to "manually" implement contrasts for repeated factors.

The method for obtaining contrasts just reviewed is very laborious and does not easily extend to multiple factor repeated measures designs. At this point, it would be useful to recall another class demonstration of how to conceptualize contrasts for repeated measures.

Since each case provides a data point for each level of the repeated measure factor, we can create contrast vectors manually as new variables - essentially

transformations. I have used the `mutate` function from **dplyr** to do this. For a review of `mutate`, see the separate document on subsetting and variable creation.

```
rpt1w.df <- mutate(rpt1w.df,
            ac1=(4*clean+(-1)*homecage+(-1)*iguana+(-1)*whiptail+(-1)*krat))
rpt1w.df <- mutate(rpt1w.df,
            ac2=(0*clean+(3)*homecage+(-1)*iguana+(-1)*whiptail+(-1)*krat))
rpt1w.df <- mutate(rpt1w.df,
            ac3=(0*clean+(0)*homecage+(-1)*iguana+(-1)*whiptail+(2)*krat))
rpt1w.df <- mutate(rpt1w.df,
            ac4=(0*clean +(0)*homecage+(-1)*iguana+(1)*whiptail+(0)*krat))
gt(rpt1w.df)
```

snum	clean	homecage	iguana	whiptail	krat	ac1	ac2	ac3	ac4
1	24	15	41	30	50	-40	-76	29	-11
2	6	6	0	6	13	-1	-1	20	6
3	4	0	5	4	9	-2	-18	9	-1
4	11	9	10	14	18	-7	-15	12	4
5	0	0	0	0	0	0	0	0	0
6	8	15	10	15	38	-46	-18	51	5
7	8	5	2	6	15	4	-8	22	4
8	0	0	0	11	54	-65	-65	97	11
9	0	3	1	1	11	-16	-4	20	0
10	7	7	4	7	23	-13	-13	35	3

Once created, we can test a hypothesis that each contrast has a value of zero (the null). Each contrast is now a single vector of ten values, and the sd of each vector is the square root of the specific error term MS values calculated above. So, when we perform a one sample t-test with a null value of zero, the t's that are produced are the square roots of the F values computed above. One might compare the mean of the first contrast vector to the value we examined when we manually did this exercise in spreadsheet form at the point in time that the repeated measures theory was first presented in class.

Also note that these F's (and the squares of the t's below) do not match the F tests of these same contrasts that were obtained in the first section of this chapter where the "split" argument was used in the `summary` function. This is because the "split" approach uses the omnibus error term (MS Axs) for all of the contrasts, but the approaches here utilize error that is specific to each contrast. The latter would always be preferred when sphericity is not present in the data system (almost always).

```
t.test(rpt1w.df$ac1)
```

```
##
##  One Sample t-test
```

```
##
## data:  rpt1w.df$ac1
## t = -2.5014, df = 9, p-value = 0.03379
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -35.421285  -1.778715
## sample estimates:
## mean of x
##     -18.6
```

```
t.test(rpt1w.df$ac2)
```

```
##
##  One Sample t-test
##
## data:  rpt1w.df$ac2
## t = -2.5909, df = 9, p-value = 0.02917
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -40.833812  -2.766188
## sample estimates:
## mean of x
##     -21.8
```

```
t.test(rpt1w.df$ac3)
```

```
##
##  One Sample t-test
##
## data:  rpt1w.df$ac3
## t = 3.3734, df = 9, p-value = 0.008212
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##   9.717798 49.282202
## sample estimates:
## mean of x
##      29.5
```

```
t.test(rpt1w.df$ac4)
```

```
##
##  One Sample t-test
##
## data:  rpt1w.df$ac4
## t = 1.1485, df = 9, p-value = 0.2804
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -2.036306  6.236306
```

```
## sample estimates:
## mean of x
##       2.1
```

This manual approach works well, is fairly efficient, and is important because of its use of the error terms specific to the contrast rather than the omnibus Axs error term which is burdened with the sphericity assumption. There is a literature that argues for the use of specific error terms to avoid this non-sphericity problem. Using the omnibus Axs error term may inflate type I error rates more for contrasts than for the omnibus F test in repeated measures designs when the sphericity assumption is violated (see section in the tookit bibliography, in particular a paper by Boik (Boik, 1981).

Also see chapter nine for a more efficient way of producing these specific error term approaches.

4.4.1 Create a plot of these contrasts and their specific errors

Modeling on the approach shown in chapter 2 for visualizing the pairwise comparisons, we plot the four orthogonal contrast means and standard errors for the contrasts just examined above with the one-sample t-test procedure. First we need to extract the variables from the wide format data frame and restructure the data to a long format. Then the `summarySE` function can be used to find the means, SE's and CI's.

```
aclong <-
  tidyr::pivot_longer(data=rpt1w.df[, c(1,7:10)],
                      cols=2:5,
                      names_to="contrast",
                      values_to="contrastvalue")
aclong
```

```
## # A tibble: 40 x 3
##     snum contrast contrastvalue
##    <int> <chr>            <dbl>
##  1     1 ac1                -40
##  2     1 ac2                -76
##  3     1 ac3                 29
##  4     1 ac4                -11
##  5     2 ac1                 -1
##  6     2 ac2                 -1
##  7     2 ac3                 20
##  8     2 ac4                  6
##  9     3 ac1                 -2
## 10     3 ac2                -18
## # ... with 30 more rows
```

```
summary.contrasts <- Rmisc::summarySE(aclong,
                                      measurevar="contrastvalue",
                                      groupvars="contrast")
summary.contrasts
```

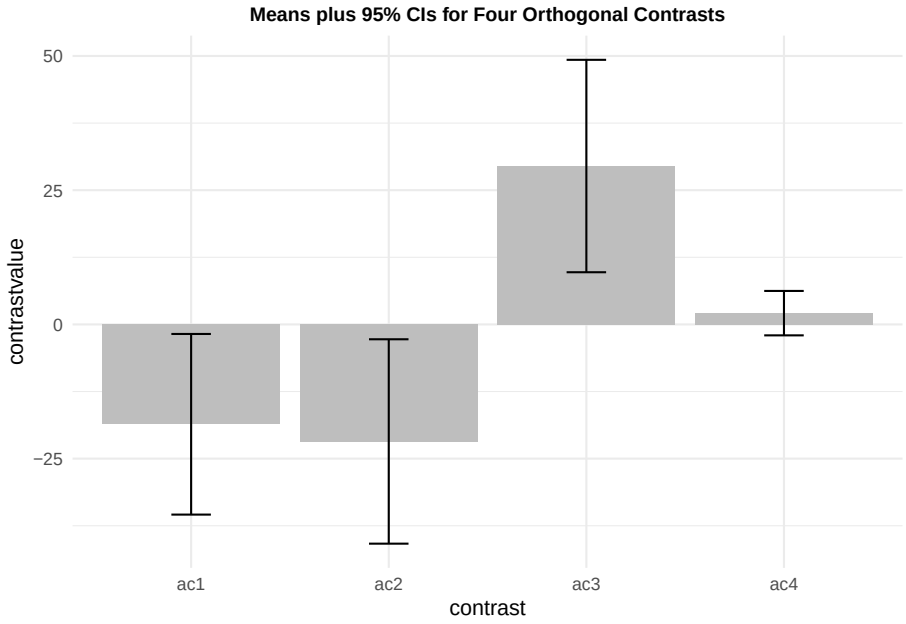
```
##   contrast  N contrastvalue        sd       se        ci
## 1      ac1 10         -18.6 23.514535 7.435949 16.821285
## 2      ac2 10         -21.8 26.607434 8.414009 19.033812
## 3      ac3 10          29.5 27.653611 8.744840 19.782202
## 4      ac4 10           2.1  5.782156 1.828478  4.136306
```

Much like the plot of paired differences in chapter 2, the visual task here is to compare each mean value to zero. Confidence intervals are overlaid and we see all but the last contrast have CI's that do not overlap zero, and that is what is expected since the first three were significant when tested above with the one-sample t-tests. The important point here is that the standard error used in calculating the CI is an error specific to the particular contrast.

Also note that contrast 4 is a pairwise comparison of the same pair of levels examined in chapter 2 as pair 4 and 3 (comparing whiptail lizard to iguana). The mean value, the CI, and the plot are the same here as for that pair seen in chapter 2.

This graph, plus the graph from chapter 2 of means plus raw data points would be a valueable way to present data for this design.

```
p10 <- ggplot(summary.contrasts, aes(x=contrast, y=contrastvalue, fill=contrast)) +
  geom_bar(position=position_dodge(), stat="identity", fill="gray") +
  geom_errorbar(aes(ymin=contrastvalue-ci, ymax=contrastvalue+ci), data=summary.contrasts,
                width=.2,                    # Width of the error bars
                position=position_dodge(.9)) +
  ggtitle("Means plus 95% CIs for Four Orthogonal Contrasts") +
  guides(fill=FALSE) + # removes legend
  theme_minimal() +
  theme(plot.title = element_text(size=10,
                                  face = "bold", hjust = .5))
p10
```



4.5 Working with afex and emmeans for contrasts and pairwise comparison follow ups

The **emmeans** package provides strong facilities for performing follow up analyses such as pairwise multiple comparisons and contrasts, in conjunction with an **afex** model. This section briefly demonstrates both of those. In a recent release of **emmeans**, the tests of contrasts and pairwise comparisons in this repeated measures design began to employ specific error terms instead of the flawed omnibus error term previously used. This may have resulted from a change in the `aov_car` function in **afex** to use a "multivariate/wide data" approach. For this 1 factor repeated measures design, `emmeans` can now be recommended, but only in conjunction wth an **afex** fit object. `emmeans` will not produce specific error terms when applied to `aov` models.

The `emmeans` function can utilize anova objects from a variety of sources. It works well with objects from `afex` so that is the approach taken here. The first line of code here simply extracts the information from the `afex` object to perform pairwise tests of the means of the five conditions. Since this would likely be a post hoc approach, the "adjust" argument in the "pairs" function permits access to the large family of correction types for error rate inflation. I illustrated the use with the Tukey method, but holm, by, bonf, fdr, and others are available. The df for the error in each of these tests is listed as 9 and the SE vary from test to test. This implies that the specific errors are used for all of the pairwise tests and that error is based on the potentially flawed MS Axs

error term is avoided. The second pairs function here produces the same tests, but without p value adjustments.

```
fit1afex.emm <- emmeans(fit1.afex, "type", data=rpt1.df)
pairs(fit1afex.emm, adjust="tukey")
```

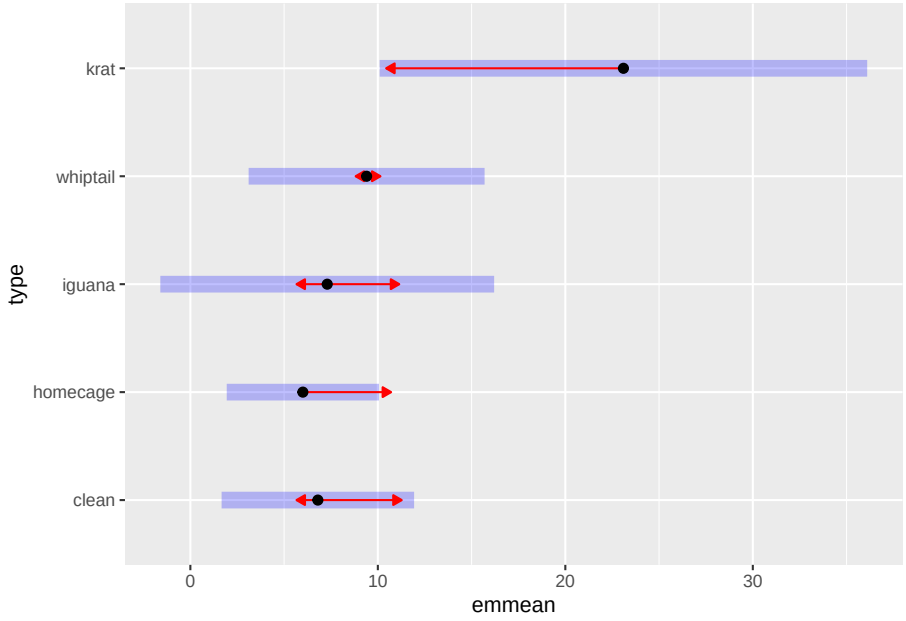
```
##  contrast            estimate   SE df t.ratio p.value
##  clean - homecage         0.8 1.34  9   0.597  0.9721
##  clean - iguana          -0.5 2.04  9  -0.245  0.9990
##  clean - whiptail        -2.6 1.30  9  -1.998  0.3385
##  clean - krat           -16.3 5.15  9  -3.167  0.0666
##  homecage - iguana       -1.3 2.92  9  -0.445  0.9905
##  homecage - whiptail     -3.4 1.75  9  -1.940  0.3634
##  homecage - krat        -17.1 5.14  9  -3.327  0.0526
##  iguana - whiptail       -2.1 1.83  9  -1.148  0.7785
##  iguana - krat          -15.8 4.90  9  -3.222  0.0614
##  whiptail - krat        -13.7 3.98  9  -3.439  0.0447
##
## P value adjustment: tukey method for comparing a family of 5 estimates
```

```
pairs(fit1afex.emm, adjust="none")
```

```
##  contrast            estimate   SE df t.ratio p.value
##  clean - homecage         0.8 1.34  9   0.597  0.5652
##  clean - iguana          -0.5 2.04  9  -0.245  0.8119
##  clean - whiptail        -2.6 1.30  9  -1.998  0.0768
##  clean - krat           -16.3 5.15  9  -3.167  0.0114
##  homecage - iguana       -1.3 2.92  9  -0.445  0.6668
##  homecage - whiptail     -3.4 1.75  9  -1.940  0.0843
##  homecage - krat        -17.1 5.14  9  -3.327  0.0088
##  iguana - whiptail       -2.1 1.83  9  -1.148  0.2804
##  iguana - krat          -15.8 4.90  9  -3.222  0.0104
##  whiptail - krat        -13.7 3.98  9  -3.439  0.0074
```

A plotting capabilty of the `emmeans` grid is possible, including visualization of CI's and significant differences. The user should read the help on the `emmeans` plotting functions to understand what the symbols mean.

```
plot(fit1afex.emm, comparisons = TRUE)
```



The implementation of a pairwise comparison method that uses a specific error term, in this 1 factor repeated measure design, could easily be done as a dependent samples "t-test". For example, the comparison of the iguana and whiptail conditions is illustrated. Note that the mean difference is the same as the pairwise comparison just above from `emmeans`, as are the t value and p values (with the table where adjust was set to "none"). Therefore, this "manual" approach produces the same results as the approach taken above for contrast4 - it is the same hypothesis test (comparing iguana and whiptail conditions). By using the dependent samples t-test for all possible pairs in a post hoc type of approach, the p values could be submitted to the `p.adjust` function for error rate inflation correction.

```
t.test(rpt1w.df$iguana, rpt1w.df$whiptail, paired=TRUE)
```

```
##
##  Paired t-test
##
## data:  rpt1w.df$iguana and rpt1w.df$whiptail
## t = -1.1485, df = 9, p-value = 0.2804
## alternative hypothesis: true mean difference is not equal to 0
## 95 percent confidence interval:
##  -6.236306  2.036306
## sample estimates:
## mean difference
##            -2.1
```

Linear Combinations (contrasts) can also be evaluated in the **emmeans** approach using the `contrast` function to produce the matrix of coefficients, and the `test` function to test them. Fortunately, this approach now employs specific error terms that match how we derived them above both from the mlmfit object and where we manually calculated them.

```
lincombs <- contrast(fit1afex.emm,
               list(type1=c(4, -1,-1,-1,-1),
                    type2=c(0,  3, -1, -1, -1),
                    type3=c(0,  0, -1, -1,  2),
                    type4=c(0,  0, -1,  1,  0)
                    ))
test(lincombs, adjust="none")
```

```
##  contrast estimate   SE df t.ratio p.value
##  type1       -18.6 7.44  9  -2.501  0.0338
##  type2       -21.8 8.41  9  -2.591  0.0292
##  type3        29.5 8.74  9   3.373  0.0082
##  type4         2.1 1.83  9   1.148  0.2804
```

A plot and table of confidence intervals for the contrasts can also be obtained, even employing p value adjustments as illustrated here with the bonferroni-sidak adjustment. As was the case when applied to the **afex** model object, specific error terms are employed. In addition, p value adjustments for the multiple testing problem, are simple to implement with the `confint` function that produces adjusted confidence intervals for each contrast.

```
confint(lincombs, adjust="sidak")
```

```
##  contrast estimate   SE df lower.CL upper.CL
##  type1       -18.6 7.44  9   -41.64     4.44
##  type2       -21.8 8.41  9   -47.88     4.28
##  type3        29.5 8.74  9     2.40    56.60
##  type4         2.1 1.83  9    -3.57     7.77
##
## Confidence level used: 0.95
## Conf-level adjustment: sidak method for 4 estimates
```

It is also possible to use the `aov_ez` flavor of the afex ANOVA functions. This approach also provides specific error terms in the construction of the t-tests for the custom set of contrasts.

First the base/omnibus ANOVA is performed:

```
contrasts(rpt1.df$type) <- contr.sum
fit2afex.ez <- aov_ez(id="snum",
                      dv="DV",
                      data=rpt1.df,
                      within="type")
```

```
summary(fit2afex.ez)
```

```
##
## Univariate Type III Repeated-Measures ANOVA Assuming Sphericity
##
##             Sum Sq num Df Error SS den Df F value    Pr(>F)
## (Intercept) 5533.5      1   3739.7      9 13.3171  0.005324 **
## type        2041.5      4   2077.3     36  8.8447 4.424e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Mauchly Tests for Sphericity
##
##      Test statistic    p-value
## type      0.0038542 7.3907e-06
##
##
## Greenhouse-Geisser and Huynh-Feldt Corrections
##  for Departure from Sphericity
##
##      GG eps Pr(>F[GG])
## type 0.3793   0.005501 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##         HF eps  Pr(>F[HF])
## type 0.4400688 0.003391046
```

The `emmeans` contrast approach provides the same specific error terms as when using the `aov_car` flavor of the **afex** approach.

```
fit2.ez.emm <- emmeans(fit2afex.ez, "type", data=rpt1.df)
lincombsez <- contrast(fit2.ez.emm,
               list(type1=c(4, -1,-1,-1,-1),
                    type2=c(0,  3, -1, -1, -1),
                    type3=c(0,  0, -1, -1,  2),
                    type4=c(0,  0, -1,  1,  0)
                    ))
test(lincombsez, adjust="none")
```

```
##  contrast estimate   SE df t.ratio p.value
##  type1       -18.6 7.44  9  -2.501  0.0338
##  type2       -21.8 8.41  9  -2.591  0.0292
##  type3        29.5 8.74  9   3.373  0.0082
##  type4         2.1 1.83  9   1.148  0.2804
```

The ability to have **emmeans** perform the correct contrast tests in this one factor repeated measures ANOVA is welcomed. Its extension to factorial designs is uncertain.

4.5.1 Use of emmeans on aov fit objects.

The **emmeans** capabilities extend to many model objects beyond afex fits. Unfortunately, when the contrast approach developed above is applied to `aov` fit objects, the error terms are not specific. But this could be a useful approach when the sphericity assumption is confidently met. In fact, when sphericity is perfect, then all specific error terms will have the exact same numeric value, but df will be smaller than for the omnibus Axs error term - thus less power.

Here, the original `aov` fit is reproduced and then the **emmeans** approach for contrasts is applied.

```
contrasts(rpt1.df$type) <- contr.sum
fit.1 <- aov(DV~type + Error(snum/type), data=rpt1.df)
summary(fit.1)
```

```
##
## Error: snum
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals  9   3740   415.5
##
## Error: snum:type
##           Df Sum Sq Mean Sq F value   Pr(>F)
## type       4   2042   510.4   8.845 4.42e-05 ***
## Residuals 36   2077    57.7
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Now the custom contrasts approach; notice the df for the errors are all 36, reflecting use of the omnibus AxS error source rather than specific error terms. The pattern of significant effects is different than with specific error terms. We conclude that the way `emmeans` and `contrast` operate on fit objects differs for `aov` and `afex` fits. Use of `afex` fits is preferred.

```
fit1.aov.emm <- emmeans(fit.1, "type", data=rpt1.df)
```

```
## Note: re-fitting model with sum-to-zero contrasts
```

```
lincombsaov <- contrast(fit1.aov.emm,
               list(type1=c(4, -1,-1,-1,-1),
                    type2=c(0,  3, -1, -1, -1),
                    type3=c(0,  0, -1, -1,  2),
                    type4=c(0,  0, -1,  1,  0)
                    ))
test(lincombsaov, adjust="none")
```

```
##  contrast estimate    SE df t.ratio p.value
##  type1       -18.6 10.74 36  -1.731  0.0919
##  type2       -21.8  8.32 36  -2.620  0.0128
##  type3        29.5  5.88 36   5.014  <.0001
##  type4         2.1  3.40 36   0.618  0.5404
```

4.6 Commentary on Contrast Analysis with the Univariate/Multivariate methods

Even if the first decision is made regarding univariate vs multivariate tests, the user must still decide how to proceed in following up these omnibus tests with pairwise comparisons or analytical/orthogonal contrasts. Unfortunately, the most direct method I have found ("split" argument with an `aov` object' employs the omnibus error term in creating standard errors or F tests for such comparisons. The only ways to obtain specific error terms are with the manual approaches outlined above or the use of **emmeans** on an afex model fit. The manual approaches are not terribly burdensome for a simple 1 factor repeated measures design, but will become very much more challenging for larger designs - e.g., two repeated factors or mixed designs with grouping factors and repeated measures factors.

There may be another way of approaching contrasts using the Linear Mixed Models methodology seen in the next chapter. That alternative may provide specific error terms as well.

The conclusion I still hold is that the SPSS MANOVA procedure is efficient and helpful in pursuing these followup contrast types of questions. Its default setting of specific error terms is of considerable value in the MANOVA procedure. It is not clear to me why this has not comprehensively found its way into the R ecosystem such that it is simple to implement. (perhaps there is still something I haven't yet learned about the suite of R functions employed here, especially perhaps `emmeans`). In attempts to apply these same strategies to factorial repeated measures designs, the types of error terms used are not so clearly specific. That work is in other tutorial documents.

Chapter 5

Linear Mixed Models

As an alternative to the traditional methods found in Chapter 3, this chapter briefly introduces Linear Mixed Effects Modeling. Although at this point in the course we have not covered any of the theory of LMM, we can examine the basics of implementation for this simple one-factor repeated measures design. LMM is a class of techniques that handle nested/hierarchical designs, and longitudinal growth curve modeling of which repeated measures designs can be seen as a subset. This suite of methods is designed to handle random factors such as "subjects" in a repeated measures design. The major advantages of LMM approaches to repeated measures are:

- Can handle missing data. The traditional approaches usually employ casewise deletion if any data points are missing for a case.
- Can handle repeated factors such as time where not all participants are measured at exactly the same time points or that vary in total number of time points measured.
- Can specify alternative covariance strutures to the compound symmetry and sphericity patterns that the traditional methods assume.

Of these, the only one that is important for the one-factor design that is illustrated here is the last point.

The illustrations here use two different packages, **nlme** and **lme4**. Detailed training in these methods is advised before routine usage.

5.1 Basic LMM Analysis using `lme`

Here, I illustrate the `lme` function from **nlme**. It requires the same long-format data frame as the `aov` function we used for the traditional analysis. That data

set is imported again here, along with the changes of "snum" to a factor and reordering the "type" variable levels.

```
# need "long" form of the data set as in Method I of chapter 3
rpt1.df <- read.csv("data/1facrpt_long.csv")
# change the snum variable to a factor variable (was numeric)
rpt1.df$snum <- as.factor(rpt1.df$snum)
rpt1.df$type <- ordered(rpt1.df$type,
                        levels=c("clean","homecage","iguana",
                                 "whiptail", "krat"))
# look at beginning and ending few lines of the  data frame
gt(headTail(rpt1.df))
```

snum	type	DV
1	clean	24
1	homecage	15
1	iguana	41
1	whiptail	30
NA	NA	...
10	homecage	7
10	iguana	4
10	whiptail	7
10	krat	23

The syntax of the `lme` function is somewhat similar to the `aov`/`lm` syntax. The most important specification is the way that the random factor is specified. The ~1|snum/type specification is not intuitively obvious. Understanding it requires a background in the theory of LMM and a careful reading of the R help page on the function, neither of which will be reviewed here.

Initially, we fit a model that contains all of the specifics required for the overall/omnibus model evaluation. Note that we examine the model with the standard `anova` and `summary` functions that we have become familiar with in other ANOVA/regression analyses.

In interpreting this initial output, several things become apparent:

1. The `anova` output yields the same F and p value as the traditional Univariate approach. This is because the traditional GLM approach can be viewed as a subset of types of LMM analyses and the default covariance structure is the one of compound symmetry, and that would be a problematic assumption here too.
2. `summary` produces information criteria (AIC/BIC) as well as info on the overall intercept (related to the overall mean) and a term that indicates that the intercept is permitted to vary across subjects (related to the Axs interaction).

3. Information on coefficients reflects the fact that individual contrast vectors were created and by default, they are those of trend analysis. This is because the defaults presume that the repeated measure is a time-related factor and that trend would thus be desireable. Since this is not the case for our illustration the succeeding code chunks changes the contrasts and we re-run the analysis.

```
#require(nlme)
# first, default, LMM model assumes sphericity and reproduces  the
# omnibus F test produced as the "average F" in the SPSS MANOVA approach
fit1.lme <- lme(DV ~ type, random = ~1|snum/type, data=rpt1.df)
anova(fit1.lme)
```

```
##             numDF denDF   F-value p-value
## (Intercept)     1    36 13.317097   8e-04
## type            4    36  8.844723  <.0001
```

```
summary(fit1.lme)
```

```
## Linear mixed-effects model fit by REML
##   Data: rpt1.df
##        AIC      BIC   logLik
##   357.0839 371.5372 -170.542
##
## Random effects:
##  Formula: ~1 | snum
##         (Intercept)
## StdDev:    8.459511
##
##  Formula: ~1 | type %in% snum
##         (Intercept) Residual
## StdDev:    6.930132 3.110725
##
## Fixed effects:  DV ~ type
##                 Value Std.Error DF  t-value p-value
## (Intercept) 10.520000  2.882776 36 3.649260  0.0008
## type.L      11.384200  2.402152 36 4.739167  0.0000
## type.Q       7.964385  2.402152 36 3.315521  0.0021
## type.C       3.004164  2.402152 36 1.250614  0.2191
## type^4       1.446227  2.402152 36 0.602055  0.5509
##  Correlation:
##        (Intr) type.L type.Q type.C
## type.L 0
## type.Q 0      0
## type.C 0      0      0
## type^4 0      0      0      0
##
```

```
## Standardized Within-Group Residuals:
##          Min           Q1          Med           Q3          Max
## -0.756930607 -0.157209242 -0.003407451  0.137647579  1.550657810
##
## Number of Observations: 50
## Number of Groups:
##           snum type %in% snum
##             10             50
```

In this next section, we recreate the same set of orthogonal contrasts used for analyses in the traditional approaches found in chapter 3.

```
# now create contrasts for the type factor
contrasts.type <- matrix(c(4, -1, -1, -1, -1,
                           0,  3, -1, -1, -1,
                           0,  0, -1, -1,  2,
                           0,  0, -1,  1,  0),
                         ncol=4)
contrasts(rpt1.df$type) <- contrasts.type
contrasts(rpt1.df$type)
```

```
##          [,1] [,2] [,3] [,4]
## clean       4    0    0    0
## homecage   -1    3    0    0
## iguana     -1   -1   -1   -1
## whiptail   -1   -1   -1    1
## krat       -1   -1    2    0
```

Fitting the model again now employs those orthogonal contrasts. Note that the t values in the summary table below for these contrasts are not the same as those we found in chapter 4 using the "manual" method or from our SPSS work. They do match the t and p values found in using **emmeans** in chapter 4. The "estimates" shown in the table are correct, reflecting the fact that the contrast coefficients have been orthonormalized. For example, multiple the first "estimate" (for type1) by 20 (the sum of the squared coefficients) and the value is -18.6, the mean of the manually created vector for contrast 1 that was done in chapter 3. The std errors and the df for the error reveal why there is a mismatch. In this analysis that is essentially a reproduction of the GLM approach, the 36 df for the error reveals the fact that errors are not specific to the contrast - the omnibus Axs MS is used in the error. This recreates the same problem with the sphericity assumption that the GLM approach has if the omnibus Axs error term is used for contrasts.

```
fit2.lme <- lme(DV ~ type, random = ~1|snum/type,   data=rpt1.df)
anova(fit2.lme)
```

```
##             numDF denDF   F-value p-value
## (Intercept)     1    36 13.317097   8e-04
```

```
## type            4    36  8.844723  <.0001
```

```
summary(fit2.lme)
```

```
## Linear mixed-effects model fit by REML
##   Data: rpt1.df
##        AIC      BIC    logLik
##   365.0495 379.5028 -174.5247
##
## Random effects:
##  Formula: ~1 | snum
##         (Intercept)
## StdDev:    8.459511
##
##  Formula: ~1 | type %in% snum
##         (Intercept) Residual
## StdDev:    6.930132 3.110725
##
## Fixed effects:  DV ~ type
##                 Value Std.Error DF   t-value p-value
## (Intercept) 10.520000 2.8827764 36  3.649260  0.0008
## type1       -0.930000 0.5371375 36 -1.731400  0.0919
## type2       -1.816667 0.6934415 36 -2.619784  0.0128
## type3        4.916667 0.9806744 36  5.013557  0.0000
## type4        1.050000 1.6985778 36  0.618164  0.5404
##  Correlation:
##       (Intr) type1 type2 type3
## type1 0
## type2 0      0
## type3 0      0     0
## type4 0      0     0     0
##
## Standardized Within-Group Residuals:
##          Min           Q1          Med           Q3          Max
## -0.756930607 -0.157209242 -0.003407451  0.137647579  1.550657810
##
## Number of Observations: 50
## Number of Groups:
##           snum type %in% snum
##             10             50
```

The **phia** package has a function called `testInteractions` that can work on a `lme` model object. This function is built to handle factorial designs, and thus the name `testInteractions`. But it can handle main effect contrasts which is what we have in this one-factor design (that can be conceived as a two factor randomized blocks design). In testing it for the first contrast, the "estimate" is the expected -18.6 value seen before for this contrast (the "psi" value for the

linear combination). I have not yet sorted out how the chi-square test statistic is created, but obviously this function approaches the question in a different way than the standard methods.

```
# phia may be able to work on these lme objects to obtain contrasts
#modmeans <- interactionMeans(fit2.lme)
#modmeans
#interactionMeans(fit2.lme, factors="type" )
# define first contrast on the rptd factor
ac1 <- list(type=c(4,-1,-1,-1,-1))
testInteractions(fit2.lme, custom=ac1,adjustment="none")
```

```
## Chisq Test:
## P-value adjustment method: none
##       Value Df  Chisq Pr(>Chisq)
## type1 -18.6  1 2.9977    0.08338 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

At this point, we have not done a full modeling process using LMM. We will need to implement expansion of the analysis to alternative covariance strutures and comparisons of different models. We also need to revisit the question of contrasts. Before we expand the approach, lets examine how a second LMM function handles the design.

5.2 Basic LMM Analysis using `lmer`

The **lme4** package and its `lmer` function are used heavily in some cognitive psychology and linguistics research areas. It also uses the same long-format data frame that we have been using above. The rudimentary approach shown here recaptures the same omnibus F value that was found with the traditional univariate approach (averaged F test). More detailed modeling is in the following section.

```
#require(lme4)
# using the same contrast set as above and from our SPSS MANOVA example
# we can specify our contrasts
#attach(rpt1.df)
contrasts.type <- matrix(c(4, -1, -1, -1, -1,
                           0,  3, -1, -1, -1,
                           0,  0, -1, -1,  2,
                           0,  0, -1,  1,  0),
                         ncol=4)
contrasts(rpt1.df$type) <- contrasts.type
contrasts(rpt1.df$type)
```

```
##          [,1] [,2] [,3] [,4]
```

```
## clean       4    0    0    0
## homecage   -1    3    0    0
## iguana     -1   -1   -1   -1
## whiptail   -1   -1   -1    1
## krat       -1   -1    2    0
```

```
fit1.lmer <- lmer(DV ~ type + (1|snum), data=rpt1.df)
anova(fit1.lmer)
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##      Sum Sq Mean Sq NumDF DenDF F value    Pr(>F)
## type 2041.5  510.37     4    36  8.8447 4.424e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(fit1.lmer)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: DV ~ type + (1 | snum)
##    Data: rpt1.df
##
## REML criterion at convergence: 349
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -1.8484 -0.3839 -0.0083  0.3361  3.7866
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  snum     (Intercept) 71.56    8.460
##  Residual             57.70    7.596
## Number of obs: 50, groups:  snum, 10
##
## Fixed effects:
##             Estimate Std. Error      df t value Pr(>|t|)
## (Intercept)  10.5200     2.8828  9.0000   3.649  0.00532 **
## type1        -0.9300     0.5371 36.0000  -1.731  0.09194 .
## type2        -1.8167     0.6934 36.0000  -2.620  0.01280 *
## type3         4.9167     0.9807 36.0000   5.014 1.44e-05 ***
## type4         1.0500     1.6986 36.0000   0.618  0.54036
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##       (Intr) type1 type2 type3
## type1 0.000
```

```
## type2 0.000  0.000
## type3 0.000  0.000 0.000
## type4 0.000  0.000 0.000 0.000
```

The results from the `lmer` analysis are identical to those from `lme`. But neither of these approaches utilizes the full set of modeling strategies that typical LMM methods use. The next sections begin to do that.

Our conclusion at this point is that we have put in place LMM code that can recreate the GLM outcome for this simple one-factor repeated measures design, including tests of contrasts that use the potentially flawed omnibus residual term (Axs).

5.3 A modeling approach

The typical approach to LMM methods is to compare models. We have seen the basics of this idea in an earlier "modeling" document. Here, for LMM models, there are several possible comparisons. The most rudimentary is to compare the fully fit model that includes the "type" IV with an intercept-only model. Those two models are created here, using the `lme` function.

```
# Intercept-only Model
basefit1.lme <- lme(DV ~ 1, random = ~1 | snum/type, data=rpt1.df,
                    method = "ML")
# Augmented Model
typemodel1.lme <- lme(DV ~ type, random = ~1 | snum/type,
                      data=rpt1.df, method = "ML")
#summary(basefit1.lme)
```

Now we can use the `anova` function to compare the two. The full model, called typemodel1.lme, has lower AIC/BIC indices and the likelihood ratio test is significant, indicating that the full model is a better fit.

Note that this involves evaluation of only the omnibus effect, and the sphericity assumption is in place for typemodel1.lme, so it may not be the best model available. See the following section for an approach that changes the covariance matrix specification.

```
anova(basefit1.lme, typemodel1.lme)
```

```
##                Model df      AIC      BIC    logLik   Test  L.Ratio p-value
## basefit1.lme       1  4 394.5131 402.1612 -193.2565
## typemodel1.lme     2  8 375.1337 390.4299 -179.5669 1 vs 2 27.37933  <.0001
```

```
#summary(typemodel1.lme)
```

5.4 Alternative covariance structures

One of the major advantages of LMM approaches is their ability to handle specifications of alternative covariance matrix structures. Choices are seen in the help for `lme` and its "correlation" argument. That will be implemented below, but first we will repeat what was done above with the first lme illustration. Our chosen orthogonal contrasts from above are in place.

```
fit2.lme <- lme(DV ~ type, random = ~1|snum/type,   data=rpt1.df)
anova(fit2.lme)
```

```
##             numDF denDF   F-value p-value
## (Intercept)     1    36 13.317097   8e-04
## type            4    36  8.844723  <.0001
```

```
#summary(fit2.lme)
```

The "correlation" argument permits specification of the covariance matrix structure. For IVs such as a time factor, such covariance structures might reasonably be autoregressive types, but for our categorical/manipulated IV (type) it is not clear what the structure might be. First, the compound symmetry/sphericity structure is specified. This reproduces the results of initial default analysis where the correlation argument was left out.

```
fit3.lme <- lme(DV ~ type, random = ~1|snum,
                correlation=corCompSymm(form=~1|snum),
                #correlation=corSymm(form=~1|snum), # specifies "unstructured"
                method = "ML", data=rpt1.df)
anova(fit3.lme)
```

```
##             numDF denDF   F-value p-value
## (Intercept)     1    36 13.317097   8e-04
## type            4    36  8.844723  <.0001
```

```
#summary(fit3.lme)
```

Now we can compare a model with a different covariance structure to this compound symmetry model. Three models are created here for purposes of comparison. The first is the intercept-only model that does not require a "correlation" argument since there is no IV specified - this repeats the baseline model illustrated above. Second is the full model with the compound symmetry specification. The first two recreate what was done above, just with different object names specified. The third is the full model, but with an "unstructured" covariance matrix specified.

```
# Intercept-only Model
basefit3.lme <- lme(DV ~ 1, random = ~1 | snum/type,
                    method = "ML", data=rpt1.df)
# Augmented Model with Compound symmetry cov matrix
```

```
typemodel3a.lme <- lme(DV ~ type, random = ~1|snum,
                correlation=corCompSymm(form=~1|snum),
                #correlation=corSymm(form=~1|snum),
                method = "ML", data=rpt1.df)
# Augmented Model with unstructured cov matrix
typemodel3b.lme <- lme(DV ~ type, random = ~1|snum,
                #correlation=corCompSymm(form=~1|snum),
                correlation=corSymm(form=~1|snum),
                method = "ML", data=rpt1.df)
```

Now we can use the `anova` function to compare models. First is the repetition of the intercept-only model and the full model with compound symmetry specified. This recreates the comparison made above where the conclusion was that the full model was a better fit.

```
anova(basefit3.lme, typemodel3a.lme)
```

```
##                 Model df      AIC      BIC    logLik   Test  L.Ratio p-value
## basefit3.lme        1  4 394.5131 402.1612 -193.2565
## typemodel3a.lme     2  8 375.1337 390.4299 -179.5669 1 vs 2 27.37933  <.0001
```

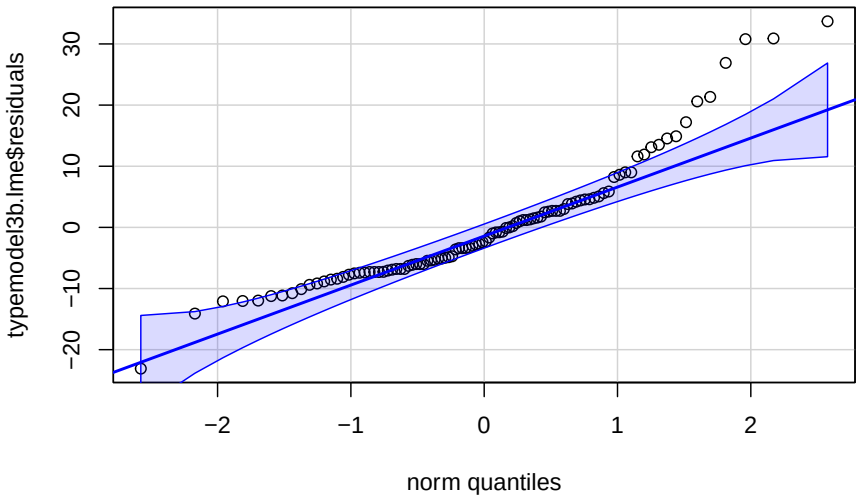
The second comparison is between the two full models, the one with the compound symmetry specification (3a) and the one with the unstructured covariance matrix specification (3b). The unstructured matrix version is a better fit as evaluated by the AIC/BIC criteria and the likelihood ratio test is significant, providing another indicator of better fit of the unstructured matrix model.

```
anova(typemodel3a.lme,typemodel3b.lme)
```

```
##                 Model df      AIC      BIC    logLik   Test  L.Ratio p-value
## typemodel3a.lme     1  8 375.1337 390.4299 -179.5669
## typemodel3b.lme     2 17 343.0687 375.5731 -154.5343 1 vs 2 50.06506  <.0001
```

The residuals are more directly accessible with an `lme` object that they were for the `aov` object in chapter 3. We can examine the residuals from this `lme` fit with a qq plot and/or histogram and we find some evidence of non-normality with a positive skewness being present. So even though the modeling suggested the unstructured covariance matrix model was the "best", it may have some issues with the normality assumption. We could also pass those residuals to a normality test such as the Anderson Darling test as we have reviewed in earlier analyses. We might also plot the residuals against the yhats to evaluate homoscedasticity (it is problematic with this data set because of the truncated distributions of the five variables at zero.)

```
#str(typemodel3b.lme)
qqPlot(typemodel3b.lme$residuals,id=FALSE)
```




```
#hist(typemodel3b.lme$residuals)
#library(nortest)
#ad.test(typemodel3b.lme$residuals)
#plot(typemodel3b.lme$fitted,typemodel3b.lme$residuals)
```

5.4.1 Contrast evaluation from the alternative covariance structure model

We can return to the use of the `testInteractions` function and evaluate the same first orthogonal contrast employed above. But this time we will utilized the model that employed the unstructured covariance matrix specification. We see that the same -18.6 "psi" value is obtained again, but the test statistic value and the p value differ from the one seen above with the compound symmetry specification. This indicates a different set of residuals used in the `lme` model here. More exploration of how this test is constructed is warranted before usage can be recommended.

```
# phia may be able to work on these lme objects to obtain contrasts
#modmeans <- interactionMeans(fit2.lme)
#modmeans
#interactionMeans(fit2.lme, factors="type" )
# define first contrast on the rptd factor
ac1 <- list(type=c(4,-1,-1,-1,-1))
testInteractions(typemodel3b.lme, custom=ac1,adjustment="none")
```

```
## Chisq Test:
## P-value adjustment method: none
##       Value Df  Chisq Pr(>Chisq)
## type1 -18.6  1 7.6477   0.005684 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

5.5 Post hoc pairwise comparisons and planned/orthogonal contrasts

Although we have explored some attempts to evaluate contrasts in the sections above, using `lme` models, we broaden the perspective here.

If the analyst wants to perform post hoc pairwise comparison tests, it is also possible to pass the LMM object to the `glht` function from the **multcomp** package. This method employs a strategy not covered in class and is one that produces an approximate standard normal Z test statistic. By evaluating type-model3b.lme, these tests employ standard errors based on the residual from that analysis where the covariance matrix was specified. The error rate inflation problem is addressed by using the Tukey correction method. Since the std errors vary from comparison to comparison, it is clear that one single error term is not employed, thus the errors are specific to the comparison.

```
library(multcomp)
multcomps <- glht(typemodel3b.lme, linfct = mcp(type = "Tukey"))
summary(multcomps)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: lme.formula(fixed = DV ~ type, data = rpt1.df, random = ~1 |
##     snum, correlation = corSymm(form = ~1 | snum), method = "ML")
##
## Linear Hypotheses:
##                            Estimate Std. Error z value Pr(>|z|)
## homecage - clean == 0      -0.800      1.226  -0.653  0.93692
## iguana - clean == 0         0.500      1.906   0.262  0.99775
## whiptail - clean == 0       2.600      1.029   2.527  0.05893 .
## krat - clean == 0          16.300      4.734   3.443  0.00348 **
## iguana - homecage == 0      1.300      2.661   0.489  0.97676
## whiptail - homecage == 0    3.400      1.455   2.337  0.09376 .
## krat - homecage == 0       17.100      4.777   3.580  0.00234 **
## whiptail - iguana == 0      2.100      1.755   1.196  0.64896
```

```
## krat - iguana == 0          15.800      4.290   3.683  0.00142 **
## krat - whiptail == 0        13.700      3.962   3.458  0.00320 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

The `ghlt` function is also capable of evaluating a set of user-specified contrasts of any kind. Here, I created a matrix of our standard set of orthogonal contrasts employed elsewhere in this document and passed them to the `glht` function based on the unstructured covariance matrix model. Note that the errors differ for the four contrasts and are specific. This may be the best alternative for evaluating these contrasts with this data set, starting with the best fit LMM model, although an **emmeans** approach is found below. Note that the "estimates" here are the correct values for the means of the four chosen contrasts that we have examined previously. The reason that these effects are tested using a Z test statistic, rather than a t, is not apparent - requires beter understanding of `ghlt`.

```
contr <- rbind("clean_vs_all" = c(4,-1,-1,-1,-1),
               "hc_vs_three" = c(0,3,-1,-1,-1),
               "krat_vs_lizards" = c(0, 0,-1, -1, 2),
               "iguana_vs_whiptail" = c(0,0,1,-1,0))
multcontrasts.lmm <- glht(typemodel3b.lme, linfct = mcp(type = contr))
summary(multcontrasts.lmm)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: User-defined Contrasts
##
##
## Fit: lme.formula(fixed = DV ~ type, data = rpt1.df, random = ~1 |
##     snum, correlation = corSymm(form = ~1 | snum), method = "ML")
##
## Linear Hypotheses:
##                           Estimate Std. Error z value Pr(>|z|)
## clean_vs_all == 0          -18.600      6.726  -2.765   0.0181 *
## hc_vs_three == 0           -21.800      7.672  -2.841   0.0144 *
## krat_vs_lizards == 0        29.500      8.069   3.656   <0.001 ***
## iguana_vs_whiptail == 0    -2.100      1.755  -1.196   0.5049
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

5.5.1 Use of emmeans on the lme model

It is worth noting that **emmeans** can also perform its many capabilities on a `lme` model object. Here, we find that the t-tests are based on 36 df, implying use of some omnibus residual term. The t values do not match the univariate approach in chap 4 when **emmeans** was employed for the `aov` model object and they do not match those found from the initial `lme` approach, so the altered covariance stucture object produced different residuals. It is not clear how the concept of a "specific" error term fits into the linear mixed effects modeling framework.

```
fit3b.lme.emm <- emmeans(typemodel3b.lme, "type", data=rpt1.df)
lincombslme <- contrast(fit3b.lme.emm,
               list(type1=c(4, -1,-1,-1,-1),
                    type2=c(0,  3, -1, -1, -1),
                    type3=c(0,  0, -1, -1,  2),
                    type4=c(0,  0, -1,  1,  0)
                    ))
test(lincombslme, adjust="none")
```

```
##  contrast estimate   SE df t.ratio p.value
##  type1       -18.6 7.09 36  -2.624  0.0127
##  type2       -21.8 8.09 36  -2.696  0.0106
##  type3        29.5 8.51 36   3.468  0.0014
##  type4         2.1 1.85 36   1.135  0.2639
##
## Degrees-of-freedom method: containment
```

Chapter 6

Bayesian Approaches

With the many and varied styles of Bayesian Analysis, it is difficult to find succinct advice on the relative merits of the various approaches. But the availability of BayesFactor-based algorithms and their ease of use has made them a go-to approach. This section will not attempt to provide a thorough overview of implementation of Bayesian inference for the repeated measure design. Instead, it will demonstrate the utility of the **BayesFactor** package. Those interested in Bayesian inference should consult the work of John Kruschke to see alternative approaches.

6.1 Bayes Factor analysis

The approach used in the **BayesFactor** package is somewhat analogous to the way we specified LMM models. The `anovaBF` function simply requires specification of a standard model, including the case variable. The key is the specification of that case variable (snum) as a random factor with the "whichRandom" argument. The default prior for the effect with **BayesFactor** functions is Cauchy with a scale parameter $r = \sqrt{2}/2$.

Note that the `anovaBF` function uses the long-format data frame that was initially introduced in chapter 2. The large BF (555.149) indicates substantial evidence in support of the alternative hypothesis relative to a model that excludes the "type" factor.

```
mod1.bf = anovaBF(DV ~type + snum, data = rpt1.df,
             whichRandom="snum")
mod1.bf
```

```
## Bayes factor analysis
## --------------
## [1] type + snum : 564.1317 ±1.5%
```

```
##
## Against denominator:
##   DV ~ snum
## ---
## Bayes factor type: BFlinearModel, JZS
```

In this simple model there is only one model comparison of interest, the full model with "type" vs an intercept only model. That is the comparison reported by the code above, so no further model comparison strategies are required.

The first of two remaining issues could be addressed in future revisions of this chapter. That first issue is the logic for choice of priors in repeated measures designs - we have used the commonly employed default method in `anovaBF`. The second is a return to the question of how to do contrast analysis or post hoc pairwise tests. An indirect way of doing contrasts from a Bayesian perspective is addressed next.

6.2 Contrasts with BayesFactor methods?

One way of approaching contrasts with the BF method is to return to the manually created contrast variables initially examined in chapter 3. I will repeat the creation of those objects here using the wide format data frame.

```
rpt1w.df <- mutate(rpt1w.df,
            ac1=(4*clean+(-1)*homecage+(-1)*iguana+(-1)*whiptail+(-1)*krat))
rpt1w.df <- mutate(rpt1w.df,
            ac2=(0*clean+(3)*homecage+(-1)*iguana+(-1)*whiptail+(-1)*krat))
rpt1w.df <- mutate(rpt1w.df,
            ac3=(0*clean+(0)*homecage+(-1)*iguana+(-1)*whiptail+(2)*krat))
rpt1w.df <- mutate(rpt1w.df,
            ac4=(0*clean+(0)*homecage+(-1)*iguana+(1)*whiptail+(0)*krat))
gt(rpt1w.df)
```

snum	clean	homecage	iguana	whiptail	krat	ac1	ac2	ac3	ac4
1	24	15	41	30	50	-40	-76	29	-11
2	6	6	0	6	13	-1	-1	20	6
3	4	0	5	4	9	-2	-18	9	-1
4	11	9	10	14	18	-7	-15	12	4
5	0	0	0	0	0	0	0	0	0
6	8	15	10	15	38	-46	-18	51	5
7	8	5	2	6	15	4	-8	22	4
8	0	0	0	11	54	-65	-65	97	11
9	0	3	1	1	11	-16	-4	20	0
10	7	7	4	7	23	-13	-13	35	3

Originally we did four separate one-sample t-tests (null was that each linear combination contrast had a mean of zero). We can do that again, but using BF methods for the 1-sample t-test analog. I illustrate here by examining the third contrast which compares the kangaroo rat condition to the average of the two lizard conditions. The alternative hypothesis that the effect size is not zero is favored moderately.

```
ttestBF(rpt1w.df$ac3, mu=0)
```

```
## Bayes factor analysis
## --------------
## [1] Alt., r=0.707 : 7.261164 ±0%
##
## Against denominator:
##   Null, mu = 0
## ---
## Bayes factor type: BFoneSample, JZS
```

Chapter 7

Robust and Resampling Methods

A few illustrations of robust methods, permutation testing, and bootstrapping are illustrated here. Some authors treat non-parametric tests as members of the class of robust methods. We have already reviewed that topic in a previous chapter.

7.1 Robust Tests

Wilcox has provided a wealth of robust methods for various inferential tests and designs with the **WRS2** package (Mair and Wilcox, 2020). The 1-factor repeated measures design is evaluated with the `rmanova` function. This function requires the long format data and we will use the initial one first used in this document in chapter 2. The robust method evaluates trimmed means and the "tr" argument specifies the degree of trimming. When tr=0, `rmanova` replicates the traditional ANOVA and produces the familiar F value from those earlier analyses. I illustrate here with "tr=.2", but the small number of data points per condition (ten) may mean that this level of trimming is too large. Note that the F statistic from the robust/trimmed approach is smaller than with "tr=0", but the test is still significant even with reduced/adjusted df.

```
library(WRS2)
#rmanova(rpt1.df$DV, groups=rpt1.df$type, blocks=rpt1.df$snum, tr=0)
rmanova(rpt1.df$DV, groups=rpt1.df$type, blocks=rpt1.df$snum, tr=.2)
```

```
## Call:
## rmanova(y = rpt1.df$DV, groups = rpt1.df$type, blocks = rpt1.df$snum,
##     tr = 0.2)
##
```

```
## Test statistic: F = 5.6244
## Degrees of freedom 1: 1.32
## Degrees of freedom 2: 6.61
## p-value: 0.04576
```

Also from **WRS2**, the `rmmcp` function implements tests of pairwise condition comparisons. Wilcox indicates that the method derives critical p-values using the Rom (1990) method where the alpha level is smaller for comparisons with smaller observed p-values. Rom's method is a modification of the Hochberg approach to p value adjustment for multiple comparisons. Note that with ten comparisons and the alpha rate correction, power is not high and only one of the comparisons is significant when tr=.2 (none when tr=0). See Wilcox' writings (Wilcox, 2017) for further explanation. The details are not in the help page for `rmmcp`. Unlike what Wilcox text says, the current version of `rmmcp` from **WRS2** function does not have an argument to change to the method of Hochberg or other methods of adjusting critical p values.

```
allpairs1 <- rmmcp(rpt1.df$DV, groups=rpt1.df$type, blocks=rpt1.df$snum, tr=.2)
#str(allpairs1)
allpairs1
```

```
## Call:
## rmmcp(y = rpt1.df$DV, groups = rpt1.df$type, blocks = rpt1.df$snum,
##     tr = 0.2)
##
##                          psihat  ci.lower ci.upper p.value  p.crit   sig
## clean vs. homecage      0.83333  -2.81237  4.47903 0.32500 0.02500 FALSE
## clean vs. iguana        0.33333  -4.13565  4.80232 0.73634 0.05000 FALSE
## clean vs. whiptail     -1.66667  -8.66274  5.32941 0.30702 0.01690 FALSE
## clean vs. krat        -12.33333 -34.36130  9.69464 0.04421 0.00730 FALSE
## homecage vs. iguana     1.16667  -3.49487  5.82820 0.28580 0.01270 FALSE
## homecage vs. whiptail  -1.66667  -7.71383  4.38050 0.24540 0.01020 FALSE
## homecage vs. krat     -12.50000 -29.68804  4.68804 0.01782 0.00568 FALSE
## iguana vs. whiptail    -2.66667  -8.50075  3.16742 0.08093 0.00851 FALSE
## iguana vs. krat       -12.00000 -24.04772  0.04772 0.00508 0.00511  TRUE
## whiptail vs. krat     -11.16667 -27.77203  5.43870 0.02373 0.00639 FALSE
```

One could use the `p.adjust` function to apply other p value adjustment methods. First, we can extract the list of ten p values from the "allpairs1" object.

```
pvals1 <- allpairs1$comp[1:10,6]
pvals1
```

```
##  [1] 0.324998613 0.736344309 0.307018998 0.044212265 0.285798317 0.245404979
##  [7] 0.017822300 0.080927194 0.005084467 0.023730192
```

Then we can apply the `p.adjust` function to that vector. Results not shown to save space.

```
p.adjust(pvals1, method="none")
p.adjust(pvals1, method="bonferroni")
p.adjust(pvals1, method="holm")
p.adjust(pvals1, method="hoch")
p.adjust(pvals1, method="hommel")
p.adjust(pvals1, method="BH")
p.adjust(pvals1, method="BY")
p.adjust(pvals1, method="fdr")
```

The raw, unadjusted, p values find four of the ten pairs to differ at the nominal .05 alpha level. When using any of the `p.adjust` methods, no pairs are found to differ. The Rom method has slightly more power and found that one of the pairs, iguana vs krat, differs.

Looking back at all ten pairs of difference, it is interesting to note that the iguana vs krat comparison is not the pair with the largest mean difference (the psihat value). This seeming discrepancy comes about because each test is done as a paired groups test (dependent samples test) and the standard error of the difference can vary between pairs as well as the mean difference. This assertion can be checked by rerunning the `rmmcp` function without any trimming and then comparing the results from one pair to a dependent samples t-test for the same pair.

```
rmmcp(rpt1.df$DV, groups=rpt1.df$type, blocks=rpt1.df$snum, tr=0)
```

```
## Call:
## rmmcp(y = rpt1.df$DV, groups = rpt1.df$type, blocks = rpt1.df$snum,
##     tr = 0)
##
##                        psihat  ci.lower ci.upper p.value  p.crit   sig
## clean vs. homecage        0.8  -4.14409  5.74409 0.56521 0.01690 FALSE
## clean vs. iguana         -0.5  -8.02647  7.02647 0.81187 0.05000 FALSE
## clean vs. whiptail       -2.6  -7.40129  2.20129 0.07680 0.00851 FALSE
## clean vs. krat          -16.3 -35.29012  2.69012 0.01142 0.00730 FALSE
## homecage vs. iguana      -1.3 -12.07890  9.47890 0.66683 0.02500 FALSE
## homecage vs. whiptail    -3.4  -9.86598  3.06598 0.08429 0.01020 FALSE
## homecage vs. krat       -17.1 -36.06142  1.86142 0.00883 0.00568 FALSE
## iguana vs. whiptail      -2.1  -8.84647  4.64647 0.28037 0.01270 FALSE
## iguana vs. krat         -15.8 -33.89064  2.29064 0.01045 0.00639 FALSE
## whiptail vs. krat       -13.7 -28.39754  0.99754 0.00740 0.00511 FALSE
```

```
t.test(rpt1w.df$iguana, rpt1w.df$krat, paired=T)
```

```
##
##  Paired t-test
##
## data:  rpt1w.df$iguana and rpt1w.df$krat
```

```
## t = -3.2225, df = 9, p-value = 0.01045
## alternative hypothesis: true mean difference is not equal to 0
## 95 percent confidence interval:
##  -26.891493  -4.708507
## sample estimates:
## mean difference
##           -15.8
```

7.2 Resampling Methods

Both permutation tests and bootstrapping of repeated measures designs would have to focus on resampling the data points within each case/subject/participant. Resampling of cases as a whole would not make sense since that variation is what is "controlled for" by doing the repeated measures analysis. I have not found many illustrations of these methods for repeated measures designs.

7.2.1 Howell's permutation approach

David Howell has shared R code for manually performing a permutation test with the 1-factor repeated measures design. I have adapted it here for our five-category design.

https://www.uvm.edu/~statdhtx/StatPages/Randomization%20Tests/RepeatedMeasuresAnovaR.html

The approach finds a way to reshuffle the five DV values for each case, randomly and within each case. The primary output is a frequency histogram of all of the F values produced by this procedure - 1000 of them since I specifed the number of resamplings as that value. Note that our observed F value for the data set is actually larger than any of the 1000 permuted samples and thus the empirical p-value is zero, an outcome that is rather extreme.

The function requires the long-format data frame which is imported here once again and given a different name so as to not confuse it with any of the earlier data frames in this document.

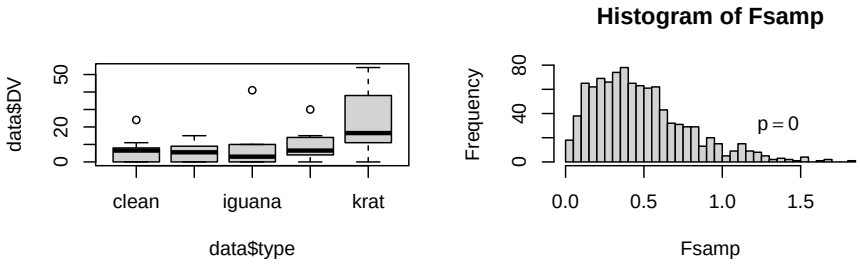
A note of caution about this method: I have carefully examined Howell's code and found that it does accomplish the permutations as intended. However, for our data set, the empirical distribution of F values has a large and suprising majority of values well less than 1.0. This strikes me as an unexpected outcome. Perhaps the non-sphericity in our data set is contributing to the extreme position of our observed F value (8.847) relative to the empirical sampling distribution based on permutations. This requires more exploration.

```
data <- read.csv("data/1facrpt_long.csv")
# make sure that the case variable is a factor
data$snum <- factor(data$snum)
```

```
# change the order of the factor levels of type
# to match the original order
# and match our prior SPSS work, including setting
# up contrasts
data$type <- ordered(data$type,
                     levels=c("clean","homecage","iguana",
                              "whiptail", "krat"))
# specify sum to zero contrasts for the type factor
options(contrasts=c("contr.sum","contr.poly"))
aovbase <- aov(DV~type + Error(snum/type), data = data)
#str(summary(aovbase))
obtF <- summary(aovbase)$"Error: snum:type"[[1]][[4]][1]
#obtF
par( mfrow = c(2,2))
boxplot(data$DV~data$type)
nreps <- 1000
counter <- 0
Fsamp <- numeric(nreps)
levels <- c(1:5)
permTypes <-NULL
orderedType <- data[order(data$type),]
set.seed(12357)
for (i in 1:nreps) {
for (j in 1:10) {
  permTypes <- c(permTypes, sample(levels,5,replace = FALSE))
  }
orderedType$permTypes <- permTypes
sampAOV <- aov(DV~factor(permTypes)+Error(factor(snum)),
               data = orderedType)
Fsamp[i] <- summary(sampAOV)$"Error: Within"[[1]][[4]][1]
if (Fsamp[i] > obtF) counter = counter + 1
permTypes <- NULL
}

p <- counter/nreps
cat("The probability of sampled F greater than obtained F is = ", p, '\n')
```

```
## The probability of sampled F greater than obtained F is =  0
```

```
hist(Fsamp, breaks = 50, bty = "n")
 legend(1,50,bquote(paste(p  == .(p))), bty = "n" )
```



7.3 Bootstrappping

I don't see much usage of bootstrapping with repeated measures designs, or with more complex factorial designs either. But it can be done although with more complex designs, multiple approaches have to be considered. With our simple 1-factor repeated measures design, the bootstrapping is more straight forward, requiring resampling within each case, across the five conditions. One very easy implementation uses a function provided in the **WRS2** package. The `rmanovab` function implements bootstrapping along with robust estimation using trimmed means. The first illustration here sets the trimming to zero, thus matching the core non-robust approach.

The `rmanovab` function can use the same long-format data frame that has been the commonly used one in this document. Here, I use the long-format data frame object created just above for the permutation illustration. The function requires two initial arguments specifying the DV and the IV. The number of bootstrap samples is set to 1000 here (probably overkill) and the trimming to zero. The analysis reports the original F value and compares it to the 95th percentile empirical F distribution cutoff. In this illustration our omnibus F exceeds that critical value, so the null is rejected.

Note that in this illustration, trimming is probably not a good idea with only n=10. Doing trimming results in a substantial power loss here. The function help does not make it clear exactly how the bootstrapping is done, and I have some curiosity about whether bootstrapping whole cases has been performed.

If that is the case, then it is not clear that it is the best approach. It requires a bit more exploration of Wilcox' algorithms before I can be fully confident in what the results mean.

```
library(WRS2)
set.seed(12345)
rmanovab(data$DV, data$type, data$snum, nboot = 1000, tr=0)
```

```
## Call:
## rmanovab(y = data$DV, groups = data$type, blocks = data$snum,
##     tr = 0, nboot = 1000)
##
## Test statistic: 8.8447
## Critical value: 6.9944
## Significant:  TRUE
```

The `pairdepb` function is another **WRS2** function that does bootstrapping. It produces pairwise comparison tests for all possible pairs. From the function help, it is not clear exactly how the bootstrapping is done and it is surprising that all of the CV are identical and none significant. I need to work through Wilcox' textbook a bit more on this function before I can recommend it.

```
## post hoc
set.seed(12354)
pairdepb(data$DV, data$type, data$snum, nboot = 1000, tr=0)
```

```
## Call:
## pairdepb(y = data$DV, groups = data$type, blocks = data$snum,
##     tr = 0, nboot = 1000)
##
##                        psihat  ci.lower ci.upper     test    crit   sig
## clean vs. homecage       0.8  -6.05004  7.65004  0.45083 6.17647 FALSE
## clean vs. iguana        -0.5 -11.20390 10.20390  1.05789 6.17647 FALSE
## clean vs. whiptail      -2.6 -16.06131 10.86131 -1.14708 6.17647 FALSE
## clean vs. krat         -16.3 -55.36506 22.76506 -2.23985 6.17647 FALSE
## homecage vs. iguana     -1.3 -11.53631  8.93631  0.80452 6.17647 FALSE
## homecage vs. whiptail   -3.4 -14.69917  7.89917 -1.63989 6.17647 FALSE
## homecage vs. krat      -17.1 -54.42551 20.22551 -2.42698 6.17647 FALSE
## iguana vs. whiptail     -2.1 -12.16089  7.96089 -2.66027 6.17647 FALSE
## iguana vs. krat        -15.8 -51.53824 19.93824 -2.76520 6.17647 FALSE
## whiptail vs. krat      -13.7 -43.26975 15.86975 -2.43691 6.17647 FALSE
```

Chapter 8

Nonparametric Approaches

Non-parametric methods are often employed when variables are not measured with interval-level measurement or when the normality assumptions cannot be satisfied. They typically transform the data to ranks prior to analysis. A good source to review these methods is the Hollander, et al, textbook (Hollander et al., 2013).

There is one easily implemented non-parametric test for the 1-factor repeated measures omnibus test. It is the Friedman's Rank Sum test and is implemented in the base R installation with the `friedman.test` function.

The function requires the data in wide format but also in matrix form, where the only variables are the repeated measure factor levels. The matrix is arranged so that the levels are columns. We already used this wide format matrix for multivariate linear modeling in chapter 2, so that matrix is still available.

```
kable(rpt1w.mat)
```

clean	homecage	iguana	whiptail	krat
24	15	41	30	50
6	6	0	6	13
4	0	5	4	9
11	9	10	14	18
0	0	0	0	0
8	15	10	15	38
8	5	2	6	15
0	0	0	11	54
0	3	1	1	11
7	7	4	7	23

Execution of the function is accomplished by passing only one argument, the name of the data matrix. The test statistic is a chi-squared variable and it is

significant for this data set. Degrees of freed are found as the number of levels of the repeated factor minus one.

```
friedman.test(rpt1w.mat)
```

```
##
##  Friedman rank sum test
##
## data:  rpt1w.mat
## Friedman chi-squared = 22.061, df = 4, p-value = 0.0001949
```

Chapter 9

Trend Analysis and A Pre-Post Design

This chapter utilizes two different data sets than the core one used in prior chapter. There are three goals. First is to review and extend some of the core analyses found in prior chapters. The second is to implement orthogonal trend analysis in a design where the repeated factor is a quantitative variable (time). The third is to approach a commonly used design, the pre-post design and emphasize the use of analytical and orthogonal contrasts to approach the primary questions posed by such a design. These analyses will confront the issue of appropriate choice of error terms for contrasts as seen in earlier chapters with recommended approaches.

9.1 Trend Analysis

This illustration uses a data set from the Maxwell, Delaney and Kelley (2017) textbook on experimental design and analysis. It is a hypothetical data set where a set of twelve children had been measured at each of four ages. The dependent variable is an age-normed score of a general cogntive test, the "McCarthy Scale of Children's Ability". As a one factor repeated measure design, the traditional method views the study as a factorial of the two variables, Age and Subject. The four ages are 30, 36, 42 and 48 months and thus represent points on a continuum. This means that the IV levels can be viewed as a representing points on a quantitative scale, and trend analysis is appropriate. An apriori hypothesis might predict that children's cognitive abilities are growing rapidly and largely linearly at these ages. But the shape of the curve might have a bend, reflecting a quadratic component. This design can also be viewed through the lens of a longitudinal growth curve design where linear mixed modeling would be appropriate. But since there are no missing data and all subjects

were measured at the same time, we will illustrate the traditional "univariate ANOVA" approach here.

9.1.1 Import the data and perform EDA.

The data file are imported as a long-format .csv file where the McCarthy Scale score is called DV and the IV is called age. There is also a "subject" number variable.

```
mdklong <- read.csv("data/maxwell_tab11_5long.csv", stringsAsFactors=TRUE)
headTail(mdklong)
```

```
##     subject               age  DV
## 1         1     thirty_months 108
## 2         1  thirtysix_months  96
## 3         1   fortytwo_months 110
## 4         1 fortyeight_months 122
## ...     ...               <NA> ...
## 45       12     thirty_months 113
## 46       12  thirtysix_months 117
## 47       12   fortytwo_months 132
## 48       12 fortyeight_months 130
```

Three preparatory steps are needed. First, we convert the numeric type variable, subject, to a factor. Then the levels of the IV (age) are placed in the correct order since as character variable, the default would be to order them alphabetically. Finally, a new numeric variable is created, reflecting the actual months of age of the measurement - four levels, equally spaced. The last step is required for drawing line graphs depicting the age growth curve, even though using age as a factor is how the analytical methods proceed.

```
mdklong$subject <- as.factor(mdklong$subject)
mdklong$age <- ordered(mdklong$age,
                       levels=c("thirty_months", "thirtysix_months",
                                "fortytwo_months", "fortyeight_months"))
mdklong$agenumeric <- dplyr::recode(mdklong$age,
                                    "thirty_months" = 30,
                                    "thirtysix_months" = 36,
                                    "fortytwo_months"  = 42,
                                    "fortyeight_months" = 48)
headTail(mdklong)
```

```
##     subject               age  DV agenumeric
## 1         1     thirty_months 108         30
## 2         1  thirtysix_months  96         36
## 3         1   fortytwo_months 110         42
## 4         1 fortyeight_months 122         48
## ...    <NA>               <NA> ...        ...
```

```
## 45       12     thirty_months 113         30
## 46       12  thirtysix_months 117         36
## 47       12   fortytwo_months 132         42
## 48       12 fortyeight_months 130         48
```

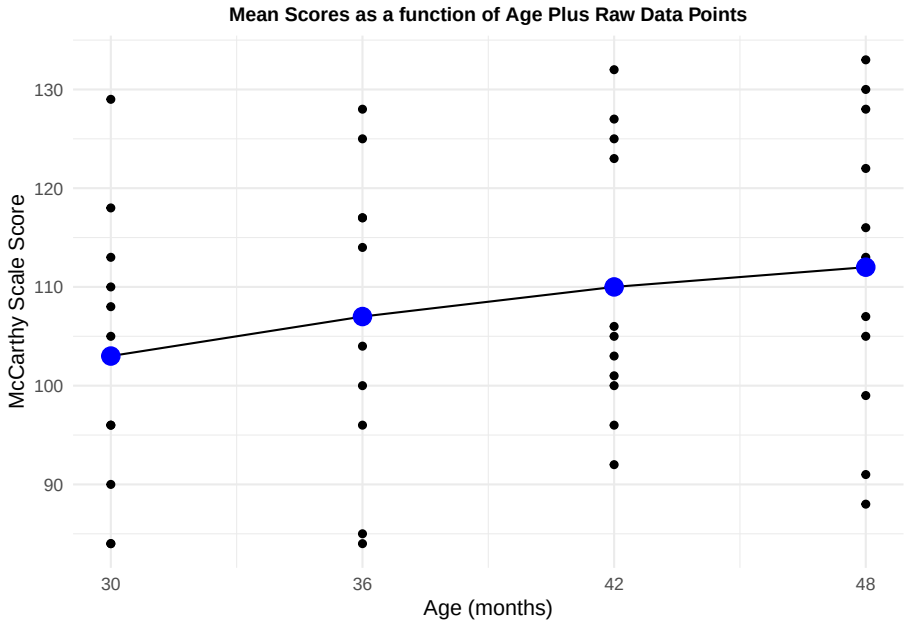
In preparation for using **ggplot2** for graphing, we derive some summary statistics u sing the `Rmisc::summarySE` function. The mean of each category is under the column labeled "DV".

```
summary.mdk.b <- Rmisc::summarySE(mdklong,
                                  measurevar="DV",
                                  groupvars="agenumeric",
                                  conf.interval=.95)
summary.mdk.b
```

```
##   agenumeric  N  DV       sd       se       ci
## 1         30 12 103 13.71131 3.958114 8.711750
## 2         36 12 107 14.16141 4.088046 8.997729
## 3         42 12 110 13.34166 3.851407 8.476889
## 4         48 12 112 14.76482 4.262237 9.381121
```

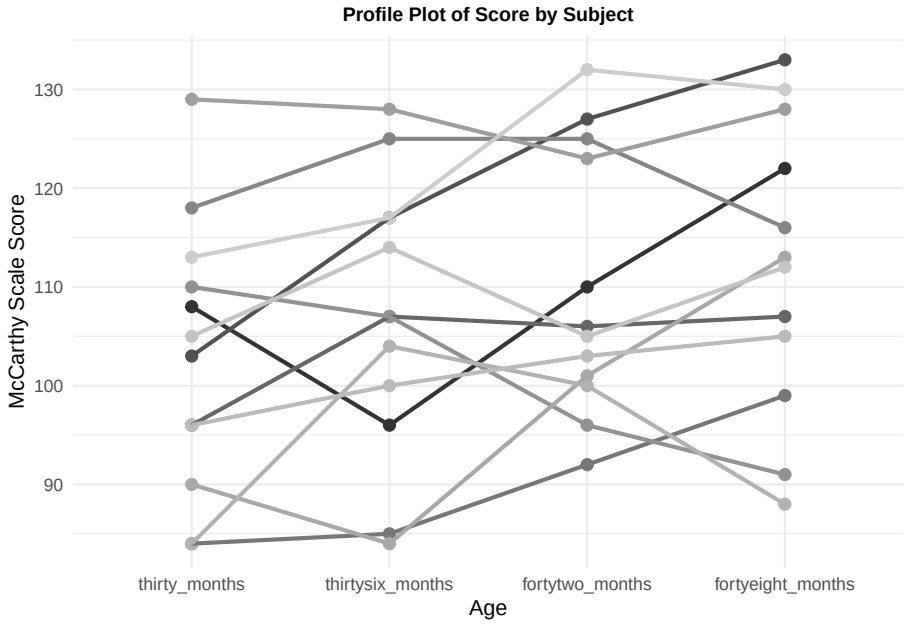
An initial graph plots the means of the four ages and also shows the raw data points. The means do depict a largely linear increase in scores, but analysis of that linearity will evaluate whether it is large, relative to sampling noise, when the inferential tests are done.

```
p1 <- ggplot(data=summary.mdk.b, aes(x=agenumeric, y=DV)) +
  geom_line()+
  geom_point(data=mdklong, aes(x=agenumeric, y=DV)) +
  geom_point(color="blue", size=4) +
  scale_x_continuous(breaks=seq(30,50,6))+
  xlab("Age (months)") +
  ylab("McCarthy Scale Score")+
  ggtitle("Mean Scores as a function of Age Plus Raw Data Points")+
  theme_minimal() +
  theme(plot.title = element_text(size=10,
                                  face = "bold", hjust = .5))
p1
```



Although the first graph showed the raw data points, it does not permit visualization of the pattern of change across age for each subject. We naturally wonder how consistently subjects show this linear increasing trend that is seen with the means. A profile plot provides this information. Although most subjects tend to increase scores across age, not all do, and there is some inconsistency in the pattern of increase - different shapes. This latter inconsistency in linearity is why a specific error term to test each trend component is useful/important as we will see below.

```
p2 <- ggplot(mdklong, aes(age, DV, colour=subject)) +
  geom_point(size = 2.5) +
  geom_line(aes(group = subject), size = 1) +
  xlab("Age") +
  ylab("McCarthy Scale Score") +
  scale_colour_grey() +
  ggtitle("Profile Plot of Score by Subject") +
  theme_minimal() +
  theme(legend.position = "none") + # removes legend
  theme(plot.title = element_text(size=10,
                                  face = "bold", hjust = .5))
p2
```



9.1.2 The omnibus analysis for the McCarthy Scale by Age dataset

The omnibus ANOVA is most efficiently accomplished with the `aov_car` function from the **afex** package. This code chunk requests that analysis plus the Greenhouse-Geisser correction for non-sphericity. First, it is useful to recall that setting the contrasts for the IV to "sum to zero" contrasts (effect coding) is desireable for these repeated measure designs, although later we will change the contrasts to orthogonal polynomial (trend) contrasts which are also a type of "sum to zero" contrast.

Use of the `summary` function on the `aov_car` object produces the ominbus F test (uncorrected), which is significant here. The Mauchley sphericity test is also provided and it is significant with an epsilon value a good bit below 1.0, so we are concerned about use of the omnibus Axs error term in this analysis. The corrected F tests using both the GG and Huynh-Feldt corrections are not significant. This is not necessarily a problem since the primary "a priori" hypothesis was for a linear increase and the test of that is more interesting than the test of the omnibus null hypothesis.

```
contrasts(mdklong$age) <- contr.sum
fit1.mdk <- aov_car(DV ~ age + Error(subject/age), data=mdklong,
                    anova_table = list(correction = "GG", es="ges"))
summary(fit1.mdk)
```

```
##
```

```
## Univariate Type III Repeated-Measures ANOVA Assuming Sphericity
##
##             Sum Sq num Df Error SS den Df  F value    Pr(>F)
## (Intercept) 559872      1     6624     11 929.7391 5.586e-12 ***
## age            552      3     2006     33   3.0269   0.04322 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Mauchly Tests for Sphericity
##
##     Test statistic  p-value
## age        0.24265 0.017718
##
##
## Greenhouse-Geisser and Huynh-Feldt Corrections
##  for Departure from Sphericity
##
##      GG eps Pr(>F[GG])
## age 0.60954    0.07479 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##        HF eps Pr(>F[HF])
## age 0.7248502 0.06353773
```

A quick way of obtaining a corrected F test plus the generalized effect size statistics is just to ask for the contents of the fit1.mdk object.

```
fit1.mdk
```

```
## Anova Table (Type 3 tests)
##
## Response: DV
##   Effect          df   MSE      F  ges p.value
## 1    age 1.83, 20.11 99.73 3.03 + .060    .075
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '+' 0.1 ' ' 1
##
## Sphericity correction method: GG
```

9.1.3 Implementing Orthogonal Polynomial Trend Analysis

Trend analysis is begun by specifying the polynomial contrast set for the IV factor. I included the exact values of the IV levels (with the "scores" argument), although it was not necessary here because the default is to assume equal spacing

of the IV levels, which we have. Notice the fractional values of the contrast coefficients. These have been orthonormalized. Note that the IV that is used is the age factor which does not have numeric values. The exact numeric values of the age levels are indicated with the "scores" argument.

```
contrasts(mdklong$age) <- contr.poly(4,scores=(c(30,36,42,48)))
contrasts(mdklong$age)
```

```
##                            .L   .Q         .C
## thirty_months     -0.6708204  0.5 -0.2236068
## thirtysix_months  -0.2236068 -0.5  0.6708204
## fortytwo_months    0.2236068 -0.5 -0.6708204
## fortyeight_months  0.6708204  0.5  0.2236068
```

We can check to see that the contrasts are orthonormalized with simple functions since the contrasts are a matrix. First we square the values and the sum the columns of the matrix to see that they do, in fact, sum to 1.0.

```
colSums(contrasts(mdklong$age)^2)
```

```
## .L .Q .C
##  1  1  1
```

If the IV levels had been unequally spaced, the contrast function could handle that by specifying the values of the levels with the "scores" argument. For example, if the levels were 30, 36, 45, and 60 months, the code would be this:

```
# code chunk not run
#contrasts(mdklong$age) <- contr.poly(4,scores=(c(30,36,45,60)))
#contrasts(mdklong$age)
```

One way of testing each of the contrasts is to use the "split" argument on the summary function where the ANOVA object is produced by the `aov` function. Note that the "residuals" term is the MS Axs residual from the omnibus ANOVA. This is the flawed error term that should not be used if there is non-sphericity. Unfortunately the first summary table here does not show that MS Axs term and it is easy to assume that the error term labled "Residuals" is what was used in the denominator of the F values. Some quick arithmetic can verify that (e.g., 540.0/602.2 does not equal the F value of 8.883 for the linear term). Use of the `summary` function without the "split" argument in the succeeding code chunk reveals that the MS Axs value of 60.79, with its 33 df, is the denominator of all three of the contrast F values in this "split" table. Since this data set appeared to have some clear degree of non-sphericity, the omnibus error term should be avoided in favor of specific error terms for each contrast.

```
fit2.mdk <- aov(DV ~ age + Error(subject/age), data=mdklong)
summary(fit2.mdk, split=list(age=list(linear=1, quadratic=2, cubic3=3)))
```

```
##
```

```
## Error: subject
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals 11   6624   602.2
##
## Error: subject:age
##                   Df Sum Sq Mean Sq F value  Pr(>F)
## age                3    552   184.0   3.027 0.04322 *
##   age: linear      1    540   540.0   8.883 0.00537 **
##   age: quadratic   1     12    12.0   0.197 0.65972
##   age: cubic3      1      0     0.0   0.000 1.00000
## Residuals         33   2006    60.8
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(fit2.mdk)
```

```
##
## Error: subject
##           Df Sum Sq Mean Sq F value Pr(>F)
## Residuals 11   6624   602.2
##
## Error: subject:age
##           Df Sum Sq Mean Sq F value Pr(>F)
## age        3    552  184.00   3.027 0.0432 *
## Residuals 33   2006   60.79
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

9.1.4 Specific error terms for the Trend contrasts

Earlier in this document, in chapters 3 and 4, we saw the difficulty of finding a way to obtain specific error terms. for example Alinear x subject. Since we need them for this trend analysis, we can take the approach of creating new variables for each trend component by applying the trend coefficients to the data for each subject, individually. This method creates three new variables (linear, quadratic, and cubic) and values of them for each subject, which are then each submitted to a one-sample t-test against a null hypothesis of zero. A plot can also be drawn.

The approach uses a wide form of the data set and then converts the DV data to a matrix so that matrix multiplication methods can be applied.

```
mdkwide <- read.csv("data/maxwell_tab11_5wide.csv")
headTail(mdkwide)
```

```
##     subject thirty_months thirtysix_months fortytwo_months fortyeight_months
## 1         1           108               96             110               122
## 2         2           103              117             127               133
```

```
## 3         3            96              107             106             107
## 4         4            84               85              92              99
## ...     ...           ...              ...             ...             ...
## 9         9            84              104             100              88
## 10       10            96              100             103             105
## 11       11           105              114             105             112
## 12       12           113              117             132             130
```

```
mwide <- as.matrix(mdkwide[, 2:5]) # only uses the DV variables, leaving subject out
headTail(mwide)
```

```
##     thirty_months thirtysix_months fortytwo_months fortyeight_months
## 1             108              96             110              122
## 2             103             117             127              133
## 3              96             107             106              107
## 4              84              85              92               99
## ...           ...             ...             ...              ...
## 9              84             104             100               88
## 10             96             100             103              105
## 11            105             114             105              112
## 12            113             117             132              130
```

Now the data set is a 12x4 matrix. The contrasts matrix used above is a 4x3 matrix. If we post-multitply the data matrix by the coefficients matrix, the product will contain three new variables that are the creation of the trend component value for each subject. To reiterate, the contrast coefficients are applied to data points from each case individually, rather than to condition means. The inferential result will be the same, but this is a direct method of obtaining error terms specific to each contrast.

```
trendmat <- mwide%*%contrasts(mdklong$age)
trendmat
```

```
##               .L  .Q            .C
##  [1,]  12.521981  12 -6.260990e+00
##  [2,]  22.360680  -4  0.000000e+00
##  [3,]   7.155418  -5  3.130495e+00
##  [4,]  11.627553   3 -1.341641e+00
##  [5,]  -1.341641  -8 -4.472136e-01
##  [6,] -15.205262  -1  3.130495e+00
##  [7,]  -1.788854   3  3.130495e+00
##  [8,]  19.230185   9 -6.260990e+00
##  [9,]   1.788854 -16  3.577709e+00
## [10,]   6.708204  -1 -3.552714e-15
## [11,]   2.683282  -1  7.602631e+00
## [12,]  14.758049  -3 -6.260990e+00
```

Next we return the trend contrast matrix to the form a data frame and change

the column/variable names to make the labeling more clear.

```
trendcontrasts <- as.data.frame(trendmat)
colnames(trendcontrasts) <- c("linear", "quadratic", "cubic")
headTail(trendcontrasts)
```

```
##     linear quadratic cubic
## 1    12.52        12 -6.26
## 2    22.36        -4     0
## 3     7.16        -5  3.13
## 4    11.63         3 -1.34
## ...    ...       ...   ...
## 9     1.79       -16  3.58
## 10    6.71        -1     0
## 11    2.68        -1   7.6
## 12   14.76        -3 -6.26
```

At this point, we can test each variable with a one-sample t-test. The squares of these t values would be identical to F tests if the contrasts were done with software that provides the hypothesis and specific error terms more directly (e.g., SPSS GLM or MANOVA). The "error term" here with these t-tests is specific because it is the variation of the contrast across the 12 subjects, thus it is a contrast x subject interaction term with the proper 11 df. It is just couched as a standard error of the mean in the production of the t value.

It is interesting that by using the specific error term approach, one of the contrasts is significant, even though the corrected omnibus F test for the age factor was not. This can easily happen with contrast partitioning when one contrast absorbs most of the effect of the condition into its single df rather than having it "diluted" by the 3 df omnibus term. And, since the error is "specific" it needs no correction for non-sphericity.

```
t.test(trendcontrasts$linear,mu=0)
```

```
##
##  One Sample t-test
##
## data:  trendcontrasts$linear
## t = 2.2414, df = 11, p-value = 0.04659
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##   0.1208294 13.2955785
## sample estimates:
## mean of x
##  6.708204
```

```
t.test(trendcontrasts$quadratic,mu=0)
```

```
##
```

```
##  One Sample t-test
##
## data:  trendcontrasts$quadratic
## t = -0.46749, df = 11, p-value = 0.6493
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -5.708132  3.708132
## sample estimates:
## mean of x
##        -1
```

```
t.test(trendcontrasts$cubic,mu=0)
```

```
##
##  One Sample t-test
##
## data:  trendcontrasts$cubic
## t = -2.7544e-15, df = 11, p-value = 1
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -2.838875  2.838875
## sample estimates:
##     mean of x
## -3.552714e-15
```

This approach is a bit simpler than the "mlmfit" approach in chapter 4. However, expanding the logic to larger designs with more than one repeated factor, or including between-group factors will not be as simple and direct.

9.1.5 Visualization of the trend contrasts

The trend pattern can be seen from the line graph created in the initial section of this chapter. However, we can also draw bar graphs that depict the mean of each of the new trend contrast variables that were created in the analyses just performed in the preceding sections.

Following the same strategy as used in chapter 4, we first need to put the trend contrasts data set into a long format using `pivot_longer`. The levels of the contrast variable are also ordered so that they are the expected linear-quadratic-cubic order.

```
trendlong <-
  tidyr::pivot_longer(data=trendcontrasts,
                      cols=1:3,
                      names_to="contrast",
                      values_to="contrastvalue")
trendlong$contrast <- ordered(trendlong$contrast,
                       levels=c("linear", "quadratic", "cubic"))
```

```
trendlong
```

```
## # A tibble: 36 x 2
##    contrast  contrastvalue
##    <ord>             <dbl>
##  1 linear            12.5
##  2 quadratic         12.0
##  3 cubic             -6.26
##  4 linear            22.4
##  5 quadratic         -4.00
##  6 cubic              0
##  7 linear             7.16
##  8 quadratic         -5.00
##  9 cubic              3.13
## 10 linear            11.6
## # ... with 26 more rows
```

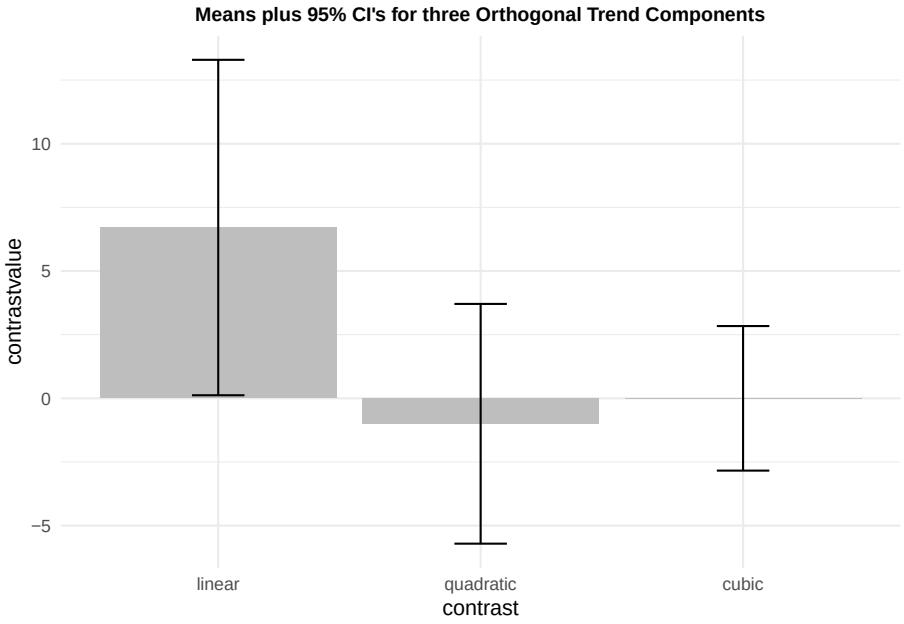
The `summarySE` function provides the summary data to be used in the ggplot graph. The "contrastvalue" column is the set of means of the three contrasts.

```
summary.trend <- Rmisc::summarySE(trendlong,
                                  measurevar="contrastvalue",
                                  groupvars="contrast")
summary.trend
```

```
##    contrast  N contrastvalue        sd       se       ci
## 1    linear 12  6.708204e+00 10.367782 2.992921 6.587375
## 2 quadratic 12 -1.000000e+00  7.410067 2.139102 4.708132
## 3     cubic 12 -3.552714e-15  4.468069 1.289820 2.838875
```

Now we can draw the bar graph. This plot uses CI's, but standard errors of the mean could also be used in the "geom_errorbar" argument. The use of the 95% CI does give the visual consistency with the outcome of the one sample t-test on this contrast since the CI does not overlap zero for the linear contrast, but the CIs for the non-significant quadratic anad cubic components do overlap zero.

```
ptrend <- ggplot(summary.trend, aes(x=contrast, y=contrastvalue, fill=contrast)) +
  geom_bar(position=position_dodge(), stat="identity", fill="gray") +
  geom_errorbar(aes(ymin=contrastvalue-ci, ymax=contrastvalue+ci), data=summary.trend,
                width=.2,                    # Width of the error bars
                position=position_dodge(.9)) +
  ggtitle("Means plus 95% CI's for three Orthogonal Trend Components") +
  guides(fill=FALSE) + # removes legend
  theme_minimal() +
  theme(plot.title = element_text(size=10,
                                  face = "bold", hjust = .5))
ptrend
```



9.1.6 Conclusion on Univariate approach to trend contrasts and specific error terms

Specific error terms will typically be important to use but this direct computational method is not the most efficient. It will be more challenging to use this direct method for more complex designs - more repeated factors or mixed designs with between-groups factors. It appears that there is no other way in R to obtain these specific error term based tests.

We can also address conclusions on this analysis. The linear trend component was the only one that was significant, using these standard NHST methods. However, the size of the effect is not large. The mean is about .65 standard deviations away from the null value of zero (see the summary.trend object above). Although viewed as a effect size statistic ("d"), this value would be labeled moderate or large. But if we think back to the original profile plot for this data set, the pattern of increases was not clearly consistent for the twelve subjects and some of them showed no increase across age. Only two of them showed the clear linear pattern of increases from each measurement to the next. So what are we to conclude? At this point all we can really say is that there is a modest increase in scores across ages, one that was significant with NHST methods for the linear trend component. But are we convinced? Perhaps not. It might be nice to take a Bayesian perspective on the strength of the evidence. The next section take a quick approach to that.

9.1.7 Bayes Factor evaluation for trend.

Once construed as the single trend contrast variable in the previous section, the linear trend variable was analyzed as a one-sample location test, the traditional t-test. There is a very quick way to obtain a Bayes Factor for such one-sample "t-test" applications. The **BayesFactor** package has a function that can analyze this one sample design and easily provide the Bayes Factor, for assessment of strength of the evidence for the alternative hypothesis. In its default configuration the `ttestBF` function can provide the Bayes Factor, extracted from the object with the `extractBF` function.

```
#library(BayesFactor)
BF10.mdktrendlinear <- extractBF(ttestBF(trendcontrasts$linear), onlybf = TRUE)
BF10.mdktrendlinear
```

```
## [1] 1.750326
```

This is a small BF10 value, indicating only weak evidence in support of the alternative hypothesis (a linear trend). It contemplates the two tailed model, but that is ok here since we would typically be interested in a two tailed test with the traditional methods.

So we would probably not argue strenuously for a conclusion that McCarthy scores increase linearly across time.

9.1.8 Linear Mixed Effects Modeling and Trend

A common alternative to the traditional univariate ANOVA performed above is a linear mixed effects analysis of repeated measures data. Setting this analysis up follows the approach oulined in chapter 5, and we will use the `lme` function from the **nlme** package.

First, we can look at the covariance and correlation matrices for the four levels of the repeated measure factor, using the wide format data frame. This is to provide assistance in choosing the type of covariance structure to model with the `lme` analysis. For time varying measurements, it is expected that the covariance stucture follows a first order autoregressive pattern. That is, measurements closer in time to one another are more strongly covarying. That is somewhat the case for this data set, but the pattern is not perfect.

```
cov(mdkwide[,2:5])
```

```
##                   thirty_months thirtysix_months fortytwo_months
## thirty_months          188.0000        154.36364        127.3636
## thirtysix_months       154.3636        200.54545        143.6364
## fortytwo_months        127.3636        143.63636        178.0000
## fortyeight_months      121.1818         97.45455        168.0909
##                   fortyeight_months
## thirty_months             121.18182
## thirtysix_months           97.45455
## fortytwo_months           168.09091
## fortyeight_months         218.00000
```

```
cor(mdkwide[,2:5])
```

```
##                   thirty_months thirtysix_months fortytwo_months
## thirty_months         1.0000000        0.7949863       0.6962361
## thirtysix_months      0.7949863        1.0000000       0.7602352
## fortytwo_months       0.6962361        0.7602352       1.0000000
## fortyeight_months     0.5985912        0.4660875       0.8533083
##                   fortyeight_months
## thirty_months             0.5985912
## thirtysix_months          0.4660875
## fortytwo_months           0.8533083
## fortyeight_months         1.0000000
```

The first and second models are designed to repeat the univariate ANOVA type of approach. The first model is an intercept only model and the second is a model that contains the age factor. But the second model assumes compound symmetry (which is spherical), and we know this is not a supported assumption (Mauchly test above and pattern of covariances). Nonetheless, this should duplicate the univariate omnibus F test.

```
# intercept only model
mdk1.lme <- lme(DV ~ 1, random = ~1 | subject/age,
                    method = "ML", data=mdklong)
# Augmented Model with Compound symmetry cov matrix
mdk1a.lme <- lme(DV ~ age, random = ~1|subject,
                       correlation=corCompSymm(form=~1|subject),
                       #correlation=corSymm(form=~1|snum),
                       method = "ML", data=mdklong)
```

An ANOVA on the "mdk1a.lme" model verifies that the same F/p values are obtained as for the traditional method, and the comparison of the full model with the intercept only model verifies that the age model is a better fit, although not convincingly since the BIC value is larger for the age model (AIC is smaller).

```
anova(mdk1a.lme)
```

```
##             numDF denDF  F-value p-value
## (Intercept)     1    33 929.7391  <.0001
## age             3    33   3.0269  0.0432
```

```
anova(mdk1.lme, mdk1a.lme)
```

```
##           Model df      AIC      BIC    logLik   Test  L.Ratio p-value
## mdk1.lme      1  4 373.4653 380.9501 -182.7327
## mdk1a.lme     2  7 370.7143 383.8127 -178.3572 1 vs 2 8.750988  0.0328
```

The next model changes the covariance structure to AR1. When compared to the compound symmetry model, both AIC and BIC are smaller, so AR1 is preferred.

```
# Augmented Model with AR1 cov matrix
mdk1c.lme <- lme(DV ~ age, random = ~1|subject,
                 #correlation=corCompSymm(form=~1|subject),
                 correlation=corAR1(form=~1|subject),
                 method = "ML", data=mdklong)
anova(mdk1a.lme, mdk1c.lme)
```

```
##           Model df      AIC      BIC    logLik
## mdk1a.lme     1  7 370.7143 383.8127 -178.3572
## mdk1c.lme     2  7 362.3200 375.4184 -174.1600
```

Somewhat surprisingly, the anova on the AR1 model does not return a significant effect of age. This reinforces the concerns we developed with the univariate approach about the strength of the age effect.

```
anova(mdk1c.lme)
```

```
##             numDF denDF  F-value p-value
## (Intercept)     1    33 904.3556  <.0001
## age             3    33   1.9392  0.1424
```

Nonetheless, since at least the linear trend component was likely to be an a priori hypothesis, we can attempt to evaluate the trend components with the `glht` function. The trend contrasts were created wtih the whole number pattern of coefficients that is appropriate for equally spaced factor levels.

As was the case in chapter 5, we note that the tests provided by `glht`are z tests and thus approximations, as are the p values. Here we find some support for a linear trend.

```
trendcontrasts2<- rbind("linear" = c(-3,-1,1,3),
                "quadratic" = c(1,-1,-1,1),
                "cubic" = c(1,-3,3,-1))
trendcontrasts.lmm <- glht(mdk1c.lme, linfct = mcp(age = trendcontrasts2))
```

```
summary(trendcontrasts.lmm)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: User-defined Contrasts
##
##
## Fit: lme.formula(fixed = DV ~ age, data = mdklong, random = ~1 | subject,
##     correlation = corAR1(form = ~1 | subject), method = "ML")
##
## Linear Hypotheses:
##                   Estimate Std. Error z value Pr(>|z|)
## linear == 0     3.000e+01  1.236e+01   2.428   0.0448 *
## quadratic == 0 -2.000e+00  3.595e+00  -0.556   0.9241
## cubic == 0     -1.498e-14  6.356e+00   0.000   1.0000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

9.1.9 General conclusion regarding trend for the Age factor.

The inconsistent pattern of evidence in support of a linear increase in scores across age is not an unusual or surprising outcome. The real answer here is that the linear increase is inconsistent across this set of 12 subjects. A larger sample size would likely have led to a more clear conclusion. This is an interesting message, that even with an N of 12 in a repeated measures design, evidence for weak or modest effects is difficult. The reader should once again refer to the profile plot to see why this is a reasonable outcome and conclusion. The subjects are simply not consistently showing any specific shape of change across time, and some are not even increasing. The aggregate is largely linear and enough so to produce significance with a traditional NHST test.

One caution is not to place too much emphasis on what this example says for the illustration regarding McCarthy scores. The data set is a hypothetical data set and we saw some patterns in it that made it clear that it was a textbook example where the data were likely created by the author rather than real data. For example it is curious that the mean value for the quadratic trend variable was exactly 1.0, and that the covariance matrix was not more clearly AR1. The primary message here is in the code sequence, not the outcome for this particular hypothesized study.

9.2 A Pre-Post design

Another primary use of repeated measures designs is in the so-called Pre-post design. One or more baseline measurements are made of the outcome variable and a treatment is then applied and perhaps continued. Later measures after the treatment (post) are then taken. The design could be as simple as one "Pre" measurement and one "Post" measurement and that repeated measures design analysis via traditional NHST would be the same as a dependent samples t-test. But additional pre-treatment or post-treament measurements might also occur.

An illustration of this design is provided by the Howell (2013) textbook with the data in table 14.3. A study by Blanchard and colleagues ("Temperature biofeedback in the treatment of migraine headache". Archives of General Psychiatry, 35:581-588, 1978) evaluated the impact of migraine headache occurrence in patients prior to or after the administration of a treatment regimen involving temperature biofeedback. There were four baseline weeks prior to treatment and six weeks after treatment. Howell presents data for the last two baseline weeks and the final three weeks after treatment began, thus the repeated measure factor has five levels. Howell did not provide the original data, but rather, simulated the outcome reported in the Blanchard paper. In research studies such as this, there might be expected to be a control group that was similarly measured, but did not receive treatment. That would comprise a so-called mixed design, with the treatment group factor being a between-groups factor and the repeated testing the repeated measure factor. In this analysis in the Howell text, and here, only the treatment group is shown since the Howell chapter and this chapter are on one-factor repeated measure designs.

This illustration is included here because we will see how the design provides a nice example of why the omnibus F test is often not very interesting. Instead, the primary rationale for the study and the hypotheses of interest, are better couched as contrasts.

9.2.1 Import the data and perform EDA

The data set imported here is not the exact data set that the Howell textbook provides. I have altered a few pieces of the data to better simulate what might be reasonably expected covariances among the pairs of the five levels of the repeated measure factor. The dependent variable is an index of headache severity comprised from a combination frequency and duration records. The initial data set is in a .csv file.

```
howell14.3wide <- read.csv("data/howell_tab14-3wideb.csv")
knitr::kable(howell14.3wide)
```

Subject	baseline1	baseline2	training1	training2	training3
1	21	22	8	6	5
2	20	19	10	4	3
3	17	15	8	4	4
4	25	30	13	12	10
5	30	27	13	8	5
6	19	27	8	7	3
7	26	16	7	2	3
8	17	18	8	1	2
9	26	24	14	8	6

Much of the work with the data set here will require a long format version of the data set, so `pivot_longer` is used to convert from wide to long. In addition the Subject variable and the time variable (week of measurement) are specified as factors, rather than numeric variables. And then the order of the time variable is specified in the sequential order that it existed in during the study. Often this last step is necessary because R orders factors alphabetically by default. But in our example the specific reordering was unnecessary because the original variable names (which became the factor levels in the long format data frame) were already properly ordered. So this last specification is placed here as a reminder if this code is used as a template for other situations

```
howell14.3long <-
  tidyr::pivot_longer(data=howell14.3wide,
                      cols=2:6,
                      names_to="time",
                      values_to="DV")
howell14.3long$Subject <- as.factor(howell14.3long$Subject)
howell14.3long$time <- as.factor(howell14.3long$time)
howell14.3long$time <- ordered(howell14.3long$time,
           levels=c("baseline1", "baseline2", "training1", "training2", "training3"))
```

Examination of a few lines of the the long format data frame shows the new structure:

```
psych::headTail(howell14.3long)
```

```
##   Subject      time  DV
## 1       1 baseline1  21
## 2       1 baseline2  22
## 3       1 training1   8
## 4       1 training2   6
## 5    <NA>      <NA> ...
## 6       9 baseline2  24
## 7       9 training1  14
## 8       9 training2   8
## 9       9 training3   6
```

One additional configuration of the data frame is needed. In order to plot the data with an X axis that reflects the successive weeks of the study, we need a numeric variable to code for week. Since the current time variable is a factor, its use on the x axis of a plot will not produce the proper spacing of measurements at weeks 1, 2 (baseline), and 6, 7, 8, (training weeks). So, a numeric time variable is needed:

```
howell14.3long$timenumeric <- dplyr::recode(
  howell14.3long$time,
  "baseline1" = 1,
  "baseline2" = 2,
  "training1"  = 6,
  "training2" = 7,
  "training3" = 8
)
psych::headTail(howell14.3long)
```

```
##   Subject      time  DV timenumeric
## 1       1 baseline1  21           1
## 2       1 baseline2  22           2
## 3       1 training1   8           6
## 4       1 training2   6           7
## 5    <NA>      <NA> ...         ...
## 6       9 baseline2  24           2
## 7       9 training1  14           6
## 8       9 training2   8           7
## 9       9 training3   6           8
```

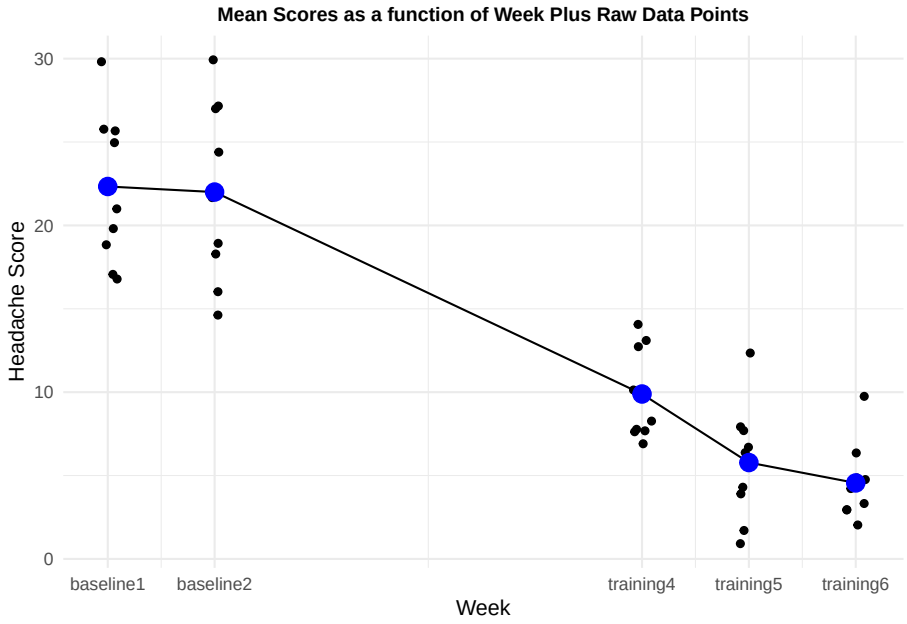
Descriptive statistics can quickly be obtained with the `summarySE` function in the **Rmisc** package. The "ci" variable found in the data framed produced by `summarySE` is the distance up and down from the mean to produce the full CI (95% here). Work in chapter 2 of this document might suggest that a different computation of standard error should be done,reflecting the standard error and the ci for purposes of plotting the repeated measures design, but the ensuing graphs do not plot error bars, so it is unnecessary. `summarySE` is used here in order to obtain the means associated with the numerically- coded "week" variable.

```
summary.h143 <- Rmisc::summarySE(howell14.3long,
                                 measurevar="DV",
                                 groupvars="timenumeric",
                                 conf.interval=.95)
knitr::kable(summary.h143)
```

timenumeric	N	DV	sd	se	ci
1	9	22.333333	4.582576	1.5275252	3.522480
2	9	22.000000	5.338539	1.7795130	4.103564
6	9	9.888889	2.713137	0.9043789	2.085501
7	9	5.777778	3.419714	1.1399047	2.628625
8	9	4.555556	2.403701	0.8012336	1.847648

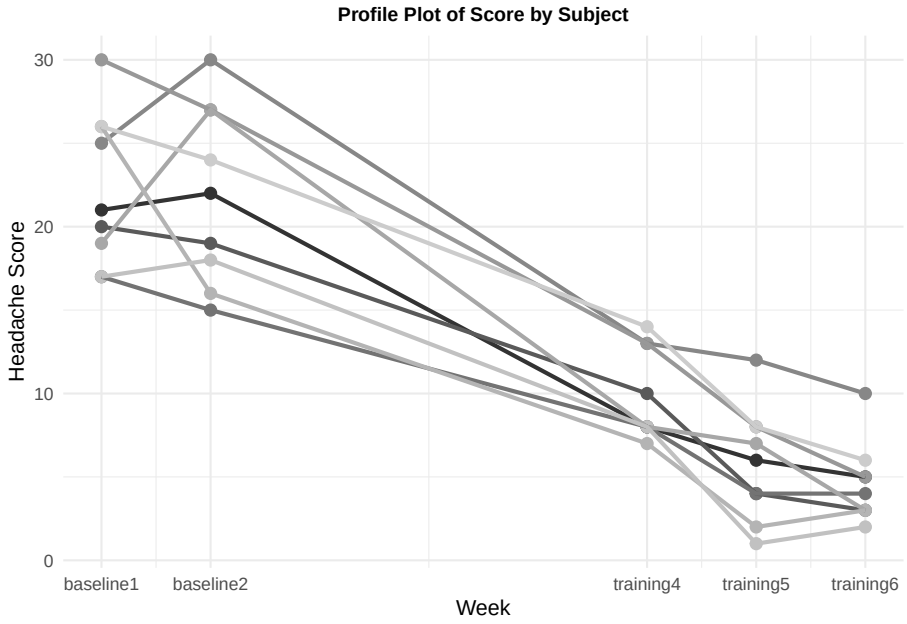
The first plot is a line graph of the week by week means of the DV. Raw data points are also plotted to provide a sense of dispersion. The ggplot axis coordinates are determined with the numerically coded time variable in the initial aesthetic and that is found in use of the summarized data frame. But the raw data points come from the full long-formate data frame, which also has the numerically coded time variable. The line geom uses the summary data frame and the y axis specification from the summary data frame is the means. So be careful in reading the code. In the first geom (jitter), y=DV refers to individual data points, but in the line geom and the second point geom, y=DV refers to the means from the summary data frame. The x axis labels are changed from the original numeric codes of "timenumeric" to the labels that describe the exact week/condition and this is accomplished with the "scale_x_continuous" phrasing. Other attributes of the graph are created with more self explanatory options. Finally, notice that the plot of the raw data points uses `geom_jitter` instead of `geom_point` in order to counter the fact that multiple data points are drawn at the same position, obscuring the individual points if one uses `geom_point`.

```
pp1 <- ggplot(data=summary.h143, aes(x=timenumeric, y=DV)) +
  geom_jitter(data=howell14.3long, aes(x=timenumeric, y=DV),
              width=.1) +
  geom_line(data=summary.h143, aes(x=timenumeric, y=DV)) +
  geom_point(data=summary.h143,aes(x=timenumeric, y=DV),color="blue", size=4) +
  ylab("Headache Score") +
  xlab("Week") +
  ggtitle("Mean Scores as a function of Week Plus Raw Data Points") +
  scale_x_continuous(breaks=c(1,2,6,7,8),
                     labels = c("baseline1","baseline2", "training4", "training5", "training
  theme_minimal() +
  theme(plot.title = element_text(size=10,
                                  face = "bold", hjust = .5))
pp1
```



A profile plot provides information on how the individual cases respond across the measurement times. There is considerable consistency across subjects in that they all report diiminished headache severity by the fourth week of training, relative to their own baseline values.

```
p2 <- ggplot(howell14.3long, aes(timenumeric, DV, colour=Subject)) +
  geom_point(size = 2.5) +
  geom_line(aes(group = Subject), size = 1) +
  scale_x_continuous(breaks=c(1,2,6,7,8),
                     labels = c("baseline1","baseline2", "training4", "training5", "training
  xlab("Week") +
  ylab("Headache Score") +
  scale_colour_grey() +
  ggtitle("Profile Plot of Score by Subject") +
  theme_minimal() +
  theme(legend.position = "none") + # removes legend
  theme(plot.title = element_text(size=10,
                                  face = "bold", hjust = .5))
p2
```



9.2.2 The omnibus analysis: Why?

In this section, the omnibus ANOVA is performed, and the 4 df effect of time was significant. But this is a fairly uninteresting outcome. There are just too many different patterns of outcome that would give rise to a significant F test with five conditions. The study design clearly indicated a few likely a priori contrasts that will be evaluated below. One can justify an assertion that the omnibus F test here is neither required nor useful - should we proceed directly to assessment of the a priori contrasts?

Perhaps not. There is some value in examining the degree to which the sphericity assumption is violated. If it appears to not be violated, then one might argue that use of the omnibus A x s error term would be appropriate for the contrasts instead of the specific error term approach advocated in earlier sections and that is performed here. If sphericity holds, then the most efficient method of performing contrast analysis would be with the functions from the **emmeans** package as seen in chapter 4. But for this data set, we can see here that the GG and HF episilons are both well below 1.0, even though they are also above the lower bound value of .25 for this five condition design. So the information produced by the **afex** package approach is not completely useless.

```
contrasts(howell14.3long$time) <- contr.sum
fit1.h143 <- afex::aov_car(DV ~ time + Error(Subject/time), data=howell14.3long,
                    anova_table = list(correction = "GG", es="ges"))
summary(fit1.h143)
```

```
##
## Univariate Type III Repeated-Measures ANOVA Assuming Sphericity
##
##             Sum Sq num Df Error SS den Df F value    Pr(>F)
## (Intercept) 7501.4      1   395.64      8  151.68 1.758e-06 ***
## time        2711.0      4   199.02     32  108.97 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## Mauchly Tests for Sphericity
##
##      Test statistic  p-value
## time       0.062965 0.044182
##
##
## Greenhouse-Geisser and Huynh-Feldt Corrections
##  for Departure from Sphericity
##
##       GG eps Pr(>F[GG])
## time 0.56457  4.233e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##        HF eps   Pr(>F[HF])
## time 0.797857 6.836594e-15
```

9.2.3 Building contrasts

With the need for specific error terms established, this section creates those contrasts individually for each subject/case rather than using an approach that applies the trend coefficients to the condition means. This is the same approach seen in chapter 4 and for trend analysis in the initial part of this chapter.

The coefficient set creates four orthogonal contrasts that might each be argued to be a priori contrasts. The first evaluates whether the average of the three training week conditions differ from the average of the two baseline conditions, and this may be the primary a priori hypothesis around which the study was designed. The second contrast hypotheses that scores at training week four may not have reached to lowest headache severity levels that the last two weeks did and so compares that week four to the average of weeks five and six. Comparison of weeks five and six in the third contrast evaluates the hypothesis that no further drop occurs after week 5. Finally the fourh contrast which compares the two baseline conditions evaluates whether baseline scores are stable in these last two measurements of pre-training.

The contrasts are created as a matrix of coefficients. where each contrast will be

a column in the matrix that is produced (but note that the vectors are in rows of the code that creates them). The exact values of the coefficients were chosen so that the scale reflects the questions posed by each of the above hypotheses will be accurately reflected in the values of the means of the new contrast vectors. For example, the mean of the first contrast would be the exact difference in the average of the first two measurements minus the average of the last three measurements. Using weights of the first contrast as, for example, 3 ,3, -2, -2, -2, would pose the same contrast question be the mean of the vector would be larger - a scaling artifact.

Inferences would not depend on the exact values of the coefficients, but only the signs/patterns - and also recall that any set of contrast weights has to sum to zero.

```
contrasts.time <- matrix(c(1/2, 1/2, -1/3, -1/3, -1/3,
                           0,    0,    1, -1/2, -1/2,
                           0,    0,    0,    1,   -1,
                           1,   -1,    0,    0,    0), ncol=4)
```

If we wanted to orthonormalize the coefficients, this would be done with a function from the **far** package. However this is not executed here and is only included for completeness.

```
contrasts.time <- far::orthonormalization(contrasts.time)
```

Now we assign that coefficient matrix to the time variable in the long format data frame.

```
contrasts(howell14.3long$time) <- contrasts.time
contrasts(howell14.3long$time)
```

```
##                 [,1] [,2] [,3] [,4]
## baseline1  0.5000000  0.0    0    1
## baseline2  0.5000000  0.0    0   -1
## training1 -0.3333333  1.0    0    0
## training2 -0.3333333 -0.5    1    0
## training3 -0.3333333 -0.5   -1    0
```

We can double check the orthogonality of the set by creating a correlation matrix for all pairs of the four codign vectors and with the `describe` function, we see that the coefficient values all sum to zero, as they should because their means are zero.

```
psych::describe(contrasts(howell14.3long$time))
```

```
##    vars n mean   sd median trimmed  mad   min max range skew kurtosis   se
## X1    1 5    0 0.46  -0.33       0 0.00 -0.33 0.5  0.83 0.29    -2.25 0.20
## X2    2 5    0 0.61   0.00       0 0.74 -0.50 1.0  1.50 0.65    -1.40 0.27
## X3    3 5    0 0.71   0.00       0 0.00 -1.00 1.0  2.00 0.00    -1.40 0.32
## X4    4 5    0 0.71   0.00       0 0.00 -1.00 1.0  2.00 0.00    -1.40 0.32
```

```
cor(contrasts(howell14.3long$time))
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    0    0    0
## [2,]    0    1    0    0
## [3,]    0    0    1    0
## [4,]    0    0    0    1
```

Next, we use apply the coefficients to the data for each case. This is done efficiently as a matrix operation. The wide format data frame is used, and converted to a matrix with the `as.matrix` function. This wide format data matrix is a 9x5 matrix. The contrast matrix, seen above, is a 5x4 matrix, thus the matrices are conformable and the produce is the 9x4 matrix desired. Each contrast is represented by nine values for the nine subjects. The four new variables are formatted into a data frame, and the columns are named to reflect the four orthogonal contrasts.

```
orthcontrasts <- as.matrix(howell14.3wide[,2:6])%*%contrasts(howell14.3long$time)
orthcontrasts <- as.data.frame(orthcontrasts)
colnames(orthcontrasts) <- c("orth1", "orth2", "orth3", "orth4")
```

```
headTail(orthcontrasts)
```

```
##     orth1 orth2 orth3 orth4
## 1   15.17   2.5     1    -1
## 2   13.83   6.5     1     1
## 3   10.67     4     0     2
## 4   15.83     2     2    -5
## ...   ...   ...   ...   ...
## 6      17     3     4    -8
## 7      17   4.5    -1    10
## 8   13.83   6.5    -1    -1
## 9   15.67     7     2     2
```

Prior to analysis, (but post hoc since we also examined the data with graphs/means), we might entertain one more contrast of interest. It might be valuable to compare the final baseline measurement to the first training period measurement (training week 4). If this were indeed a contrast chosen after looking at the data we would classify it as a post hoc contrast.

It can be created as a five element vector.

```
postcontr1 <- as.matrix(c(0, 1, -1, 0, 0),nrow=5)
postcontr1
```

```
##      [,1]
## [1,]    0
## [2,]    1
## [3,]   -1
```

```
## [4,]    0
## [5,]    0
```

Next, we use the familiar matrix multiplication operation to create a new contrast vector with a value for each case from the wide format data frame and we call this data vector "posthoc1". This time the wide data frame is converted to a matrix within the same line of code as the matrix multiplication is accomplished.

```
posthoc1 <- as.matrix(howell14.3wide[,2:6])%*%postcontr1
posthoc1
```

```
##       [,1]
##  [1,]   14
##  [2,]    9
##  [3,]    7
##  [4,]   17
##  [5,]   14
##  [6,]   19
##  [7,]    9
##  [8,]   10
##  [9,]   10
```

Now we can bind the original matrix of the orthogonal contrast vectors with this posthoc vector, producing a 9 x 5 matrix: Nine cases and five variables - four planned orthogonal and one post hoc contrast

```
allcontrasts <- cbind(orthcontrasts,posthoc1)
knitr::kable(allcontrasts)
```

orth1	orth2	orth3	orth4	posthoc1
15.16667	2.5	1	-1	14
13.83333	6.5	1	1	9
10.66667	4.0	0	2	7
15.83333	2.0	2	-5	17
19.83333	6.5	3	3	14
17.00000	3.0	4	-8	19
17.00000	4.5	-1	10	9
13.83333	6.5	-1	-1	10
15.66667	7.0	2	2	10

###Evaluating the contrasts for the pre-post design.

Following the approach outlined above in chapter 4 and in the trend analysis section of this chapter, we submit each of these new variables to a one-sample t-test with a null hypothesis that the mean is zero. The results of these t-tests would be identical to the F tests done a more traditional way that students would have been familiar with from using the SPSS MANOVA procedure - the squares of these t's would be those F values.

There is a question of alpha rate adjustment for multiple comparisons since we have done five tests. It can be argued that the four planned orthogonal contrasts can be tested at a nominal, non-adjusted, alpha level (.05), but the post hoc test should have some adjustment applied. Since five tests were done, a bonferroni adjustment would set the alph level at .01. This is why I chose the CI for the post hoc contrast as a 99% CI and lef the orthogonal ones at their default 95% values.

```
t1 <- t.test(allcontrasts$orth1, mu=0, alternative="two.sided")
t2 <- t.test(allcontrasts$orth2, mu=0, alternative="two.sided")
t3 <- t.test(allcontrasts$orth3, mu=0, alternative="two.sided")
t4 <- t.test(allcontrasts$orth4, mu=0, alternative="two.sided")
t5 <- t.test(allcontrasts$posthoc1, mu=0, alternative="two.sided", conf.level = .99)
t1
```

```
##
##  One Sample t-test
##
## data:  allcontrasts$orth1
## t = 18.083, df = 8, p-value = 8.979e-08
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  13.45877 17.39308
## sample estimates:
## mean of x
##  15.42593
```

```
t2
```

```
##
##  One Sample t-test
##
## data:  allcontrasts$orth2
## t = 7.2488, df = 8, p-value = 8.815e-05
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  3.219984 6.224461
## sample estimates:
## mean of x
##  4.722222
```

```
t3
```

```
##
##  One Sample t-test
##
## data:  allcontrasts$orth3
## t = 2.1368, df = 8, p-value = 0.0651
```

```
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -0.09676476  2.54120920
## sample estimates:
## mean of x
##  1.222222
```

```
t4
```

```
##
##  One Sample t-test
##
## data:  allcontrasts$orth4
## t = 0.19612, df = 8, p-value = 0.8494
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -3.586120  4.252787
## sample estimates:
## mean of x
## 0.3333333
```

```
t5
```

```
##
##  One Sample t-test
##
## data:  allcontrasts$posthoc1
## t = 8.9147, df = 8, p-value = 1.987e-05
## alternative hypothesis: true mean is not equal to 0
## 99 percent confidence interval:
##   7.552624 16.669598
## sample estimates:
## mean of x
##  12.11111
```

An alternative way to do the multiple comparison adjustment is to submit the set of p values to the `p.adjust` procedure and that is done here with the Holm method, just illustrate the alternative possibilities for adjusted p values. It was argued that the adjusted p value would only be needed for the final contrast (the post hoc one), but all are included in case the adjustment is desired for all of them and to provide the right number of tests for the Holm method to operate on.

The final contrast is significant even with this Holm adjustment.

```
p.adjust(c(t1$p.value,t1$p.value,t2$p.value,
           t3$p.value,t4$p.value,t5$p.value), method="holm")
```

```
## [1] 5.387577e-07 5.387577e-07 2.644635e-04 1.301942e-01 8.494094e-01
```

```
## [6] 7.949708e-05
```

9.2.4 Visualization of the contrasts

Even though the comparisons of the five contrasts could be visualized from the original line graph of this data set (and perhaps the profile plot), it can be useful to plot the means of the five contrasts as a bar graph and add confidence intervals based on the specific errors embodied in the variances of those five variables.

First we need to convert the wide format data matrix of the five contrasts to a long format.

```
contrastslong <-
  tidyr::pivot_longer(data=allcontrasts,
                      cols=1:5,
                      names_to="contrast",
                      values_to="contrastvalue")
contrastslong$contrast <- ordered(contrastslong$contrast,
                                  levels=c("orth1", "orth2", "orth3",
                                           "orth4", "posthoc1"))
contrastslong
```

```
## # A tibble: 45 x 2
##    contrast contrastvalue
##    <ord>            <dbl>
##  1 orth1             15.2
##  2 orth2              2.5
##  3 orth3              1
##  4 orth4             -1
##  5 posthoc1          14
##  6 orth1             13.8
##  7 orth2              6.5
##  8 orth3              1
##  9 orth4              1
## 10 posthoc1           9
## # ... with 35 more rows
```

Next, we once again use `summarySE` to extract the relevant summary information. Note that I requested a 99% confidence interval even though the 95% interval is probably appropriate for the four orthogonal contrasts. I have not sorted out how to use ggplot to put a different CI on different bars of the same graph, so I went with 99% for all here.

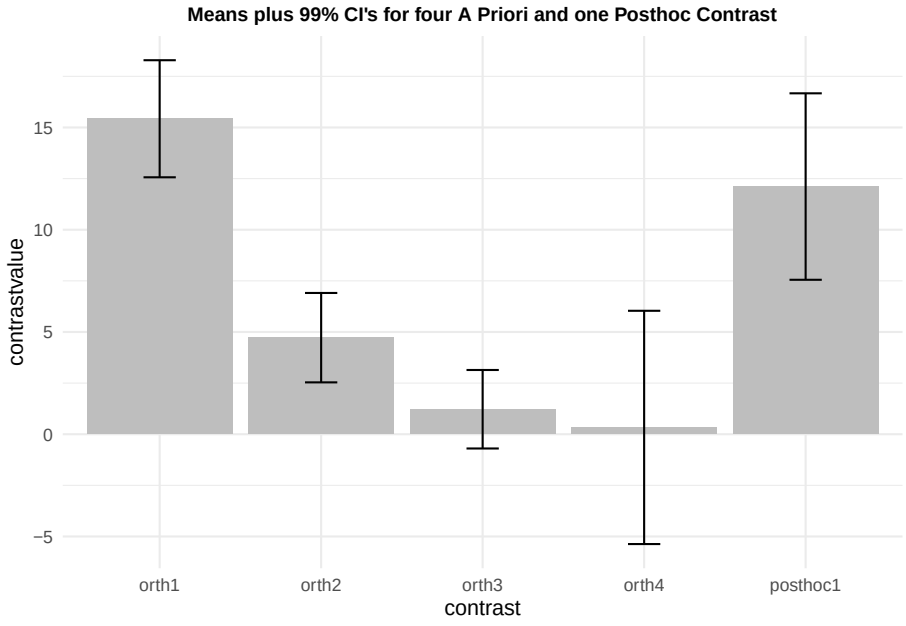
```
summary.allcontr <- Rmisc::summarySE(contrastslong,
                                     measurevar="contrastvalue",
                                     groupvars="contrast",
                                     conf.interval=.99)
```

```
summary.allcontr
```

```
##   contrast N contrastvalue       sd        se       ci
## 1    orth1 9    15.4259259 2.559176 0.8530587 2.862342
## 2    orth2 9     4.7222222 1.954340 0.6514466 2.185856
## 3    orth3 9     1.2222222 1.715938 0.5719795 1.919213
## 4    orth4 9     0.3333333 5.099020 1.6996732 5.703062
## 5 posthoc1 9    12.1111111 4.075673 1.3585577 4.558487
```

Now `ggplot` is used for the typical bar graph with confidence intervals overlaid as "error bars".

```
pall <- ggplot(summary.allcontr, aes(x=contrast, y=contrastvalue, fill=contrast)) +
  geom_bar(position=position_dodge(), stat="identity", fill="gray") +
  geom_errorbar(aes(ymin=contrastvalue-ci, ymax=contrastvalue+ci), data=summary.allcontr,
                width=.2,                    # Width of the error bars
                position=position_dodge(.9)) +
  ggtitle("Means plus 99% CI's for four A Priori and one Posthoc Contrast") +
  guides(fill=FALSE) + # removes legend
  theme_minimal() +
  theme(plot.title = element_text(size=10,
                                  face = "bold", hjust = .5))
pall
```



9.2.5 Bayes Factor values for the pre-post contrasts

The **BayesFactor** package provides a function (`ttestBF`) for doing a style of Bayesian analysis that creates Bayes Factors for each test that is the one sample t-test. Then we can use `extractBF`, from the same package, to find the BF10 value. BF10 is interpreted as the relative strength of evidence for support of the alternative hypothesis and I used it for each of the contrasts that were significant as a traditional t-test. All three had extremely large BF10 values, indicating strong support for the alternative hypothesis against the null.

```
BF10.orth1 <- extractBF(ttestBF(allcontrasts$orth1), onlybf = TRUE)
```

```
## t is large; approximation invoked.
```

```
BF10.orth1
```

```
## [1] 116788.3
```

```
BF10.orth2 <- extractBF(ttestBF(allcontrasts$orth2), onlybf = TRUE)
BF10.orth2
```

```
## [1] 312.9519
```

```
BF10.posthoc1 <- extractBF(ttestBF(allcontrasts$posthoc1), onlybf = TRUE)
BF10.posthoc1
```

```
## [1] 1108.65
```

The contrasts that were not significant in the traditional t-tests are now characterized with BF01 levels that indicate relative support for the null hypothesis. BF01 is found as the reciprocal of BF10. The third orthogonal contrast led to an inconclusive outcome. Nether BF01, nor BF10 were very far from zero, indicating a situation where evidence for or against the null is weak. The fourth orthogonal contrast produced a modest sized BF01, indicated weak to moderate suppport for the null

```
BF01.orth3 <- 1/extractBF(ttestBF(allcontrasts$orth3), onlybf = TRUE)
BF01.orth3
```

```
## [1] 0.6670013
```

```
BF01.orth4 <- 1/extractBF(ttestBF(allcontrasts$orth4), onlybf = TRUE)
BF01.orth4
```

```
## [1] 3.058797
```

In this section, I am implying that supplementing the traditional NHST method with the Bayes Factor score results in a more full characterization of the data and does not require choosing sides in the Frequentist/Bayesian debate. Both sets of information are helpful in describing the results of the study.

Chapter 10

Reproducibility and history

Version 1.4 April 25, 2023 Added several changes to contrast section incorporating updated capability of the emmeans package. Added much explanatory text. Clarified wording in many sections. Corrected typos Added references

Version 1.3 Jan 18, 2021 Added several methods of drawing graphs with error bars in chapter 2 and 4. Added a chapter on trend analysis and on Pre-post designs. Refined wording in several sections.

Version 1.2 Dec 31, 2020 Separated section on contrast analysis to a separate chapter. Refined the section on using **afex** in chapter 3 Edited style/grammar in several sections.

Version 1.1 Nov, 2020 Edited style and grammar

Version 1.0 August, 2020

```
sessionInfo()
```

```
## R version 4.2.2 (2022-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19044)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.utf8
## [2] LC_CTYPE=English_United States.utf8
## [3] LC_MONETARY=English_United States.utf8
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.utf8
##
## attached base packages:
```

```
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] WRS2_1.1-4             tidyr_1.2.1            sjstats_0.18.2
##  [4] sciplot_1.2-0          Rmisc_1.5.1            lattice_0.20-45
##  [7] rmarkdown_2.19         psych_2.2.9            plyr_1.8.8
## [10] phia_0.2-1             permuco_1.1.2          nortest_1.0-4
## [13] nlme_3.1-161           multcomp_1.4-20        TH.data_1.1-1
## [16] MASS_7.3-58.1          survival_3.5-0         mvtnorm_1.1-3
## [19] knitr_1.41             kableExtra_1.3.4       gt_0.8.0
## [22] granova_2.1            ggthemes_4.2.4         ggplot2_3.4.0
## [25] foreign_0.8-84         ez_4.4-0               emmeans_1.8.3
## [28] car_3.1-1              carData_3.0-5          BayesFactor_0.9.12-4.4
## [31] coda_0.19-4            afex_1.2-1             lme4_1.1-31
## [34] Matrix_1.5-3
##
## loaded via a namespace (and not attached):
##  [1] insight_0.18.8      webshot_0.5.4       httr_1.4.4
##  [4] numDeriv_2016.8-1.1 backports_1.4.1     tools_4.2.2
##  [7] sjlabelled_1.2.0    utf8_1.2.2          R6_2.5.1
## [10] DBI_1.1.3           mgcv_1.8-41         colorspace_2.0-3
## [13] permute_0.9-7       withr_2.5.0         tidyselect_1.2.0
## [16] mnormt_2.1.1        compiler_4.2.2      performance_0.10.2
## [19] cli_3.6.0           rvest_1.0.3         xml2_1.3.3
## [22] sandwich_3.0-2      labeling_0.4.2      bookdown_0.31
## [25] bayestestR_0.13.0   scales_1.2.1        mc2d_0.1-22
## [28] pbapply_1.7-0       systemfonts_1.0.4   stringr_1.5.0
## [31] digest_0.6.31       minqa_1.2.5         svglite_2.1.1
## [34] pkgconfig_2.0.3     htmltools_0.5.4     highr_0.10
## [37] fastmap_1.1.0       rlang_1.0.6         rstudioapi_0.14
## [40] farver_2.1.1        generics_0.1.3      zoo_1.8-11
## [43] dplyr_1.0.10        magrittr_2.0.3      Rcpp_1.0.9
## [46] munsell_0.5.0       fansi_1.0.3         abind_1.4-5
## [49] lifecycle_1.0.3     stringi_1.7.12      yaml_2.3.6
## [52] grid_4.2.2          parallel_4.2.2      sjmisc_2.8.9
## [55] splines_4.2.2       pillar_1.8.1        boot_1.3-28.1
## [58] estimability_1.4.1  reshape2_1.4.4      codetools_0.2-18
## [61] glue_1.6.2          evaluate_0.19       modelr_0.1.10
## [64] vctrs_0.5.1         nloptr_2.0.3        MatrixModels_0.5-1
## [67] gtable_0.3.1        purrr_1.0.1         reshape_0.8.9
## [70] assertthat_0.2.1    datawizard_0.6.5    xfun_0.36
## [73] xtable_1.8-4        broom_1.0.2         viridisLite_0.4.1
## [76] tibble_3.1.8        lmerTest_3.1-3      ellipsis_0.3.2
```

Bibliography

Allaire, J., Xie, Y., McPherson, J., Luraschi, J., Ushey, K., Atkins, A., Wickham, H., Cheng, J., Chang, W., and Iannone, R. (2020). *rmarkdown: Dynamic Documents for R*. R package version 2.1.

Bakeman, R. and McArthur, D. (1996). Picturing repeated measures: Comments on loftus, morrison, and others. *Behavior Research Methods, Instruments, and Computers*, 28(4):584–589.

Bates, D., Maechler, M., Bolker, B., and Walker, S. (2019). *lme4: Linear Mixed-Effects Models using 'Eigen' and S4*. R package version 1.1-21.

Boik, R. J. (1981). A priori tests in repeated measures designs: Effects of nonsphericity. *Psychometrika*, 46(3):241–255.

Cousineau, D. (2005). Confidence intervals in within-subject designs: A simpler solution to loftus and masson's method. *Tutorials in quantitative methods for psychology*, 1(1):42–45.

Cumming, G. and Finch, S. (2005). Inference by eye: confidence intervals and how to read pictures of data. *Am Psychol*, 60(2):170–80.

De Rosario-Martinez, H. (2015). *phia: Post-Hoc Interaction Analysis*. R package version 0.2-1.

Fox, J., Weisberg, S., and Price, B. (2020). *car: Companion to Applied Regression*. R package version 3.0-7.

Franz, V. H. and Loftus, G. R. (2012). Standard errors and confidence intervals in within-subjects designs: Generalizing loftus and masson (1994) and avoiding the biases of alternative accounts. *Psychonomic Bulletin and Review*, 19(3):395–404.

Frossard, J. and Renaud, O. (2019). *permuco: Permutation Tests for Regression, (Repeated Measures) ANOVA/ANCOVA and Comparison of Signals*. R package version 1.1.0.

Gross, J. and Ligges, U. (2015). *nortest: Tests for Normality*. R package version 1.0-4.

Hays, W. L. (1994). *Statistics*. Harcourt College Publishers, Fort Worth, 5th edition.

Hollander, M., Wolfe, D. A., and Chicken, E. (2013). *Nonparametric statistical methods*. John Wiley and Sons, Inc., Hoboken, New Jersey, 3rd edition.

Hope, R. M. (2013). *Rmisc: Ryan Miscellaneous*. R package version 1.5.

Hothorn, T., Bretz, F., and Westfall, P. (2020). *multcomp: Simultaneous Inference in General Parametric Models*. R package version 1.4-12.

Howell, D. C. (2013). *Statistical methods for psychology*. Wadsworth Cengage Learning, Belmont, CA, 8th edition.

Iannone, R., Cheng, J., and Schloerke, B. (2019). *gt: Easily Create Presentation-Ready Display Tables*. R package version 0.1.0.

Keppel, G. and Wickens, T. D. (2004). *Design and analysis : a researcher's handbook*. Pearson Prentice Hall, Upper Saddle River, N.J, 4th edition.

Kirk, R. E. (2013). *Experimental design : procedures for the behavioral sciences*. Sage Publications, Thousand Oaks, 4th edition.

Lawrence, M. A. (2016). *ez: Easy Analysis and Visualization of Factorial Experiments*. R package version 4.4-0.

Lenth, R. (2020). *emmeans: Estimated Marginal Means, aka Least-Squares Means*. R package version 1.4.5.

Loftus, G. R. and Masson, M. E. J. (1994). Using confidence intervals in within-subject designs. *Psychonomic Bulletin and Review*, 1(4):476–490.

Lüdecke, D. (2020). *sjstats: Statistical Functions for Regression Models (Version 0.17.9)*.

Mair, P. and Wilcox, R. (2020). Robust statistical methods in r using the wrs2 package. *Behavior Research Methods*, 52:464–488.

Maxwell, S. E., Delaney, H. D., and Kelley, K. (2017). *Designing experiments and analyzing data : a model comparison perspective*. Routledge, New York, NY, third edition / edition.

McCulloch, C. E. (2005). Repeated measures anova, rip? *Chance*, 18(3):29–33.

Morales, M., with code developed by the R Development Core Team, with general advice from the R-help listserv community, and especially Duncan Murdoch. (2020). *sciplot: Scientific Graphing Functions for Factorial Designs*. R package version 1.2-0.

Morey, R. D. (2008). Confidence intervals from normalized data: A correction to cousineau (2005). *reason*, 4(2):61–64.

Morey, R. D. and Rouder, J. N. (2018). *BayesFactor: Computation of Bayes Factors for Common Designs*. R package version 0.9.12-4.2.

Morrison, G. R. and Weaver, B. (1995). Exactly how many p values is a picture worth? a commentary on loftus's plot-plus-error-bar approach. *Behavior Research Methods, Instruments, and Computers*, 27(1):52–56.

Myers, J. L., Well, A., and Lorch, R. F. (2010). *Research design and statistical analysis*. Routledge, New York, 3rd edition.

Phillips, T. J. and Dudek, B. C. (1989). Modification of ethanol effects by bicuculline: genotype-dependent responses and inheritance. *Psychopharmacology*, 98(4):549–555.

Pinheiro, J., Bates, D., and R-core (2020). *nlme: Linear and Nonlinear Mixed Effects Models*. R package version 3.1-145.

Pruzek, R. M. and Helmreich, J. E. (2014). *granova: Graphical Analysis of Variance*. R package version 2.1.

R Core Team (2020). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Revelle, W. (2020). *psych: Procedures for Psychological, Psychometric, and Personality Research*. R package version 1.9.12.31.

Rom, D. M. (1990). A sequentially rejective test procedure based on a modified bonferroni inequality. *Biometrika*, 77(3):663–665.

RStudio Team (2015). *RStudio: Integrated Development Environment for R*. RStudio, Inc., Boston, MA.

Singmann, H., Bolker, B., Westfall, J., Aust, F., and Ben-Shachar, M. S. (2020). *afex: Analysis of Factorial Experiments*. R package version 0.27-2.

Wickham, H., Chang, W., Henry, L., Pedersen, T. L., Takahashi, K., Wilke, C., Woo, K., Yutani, H., and Dunnington, D. (2020). *ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics*. R package version 3.3.0.

Wickham, H. and Henry, L. (2020). *tidyr: Tidy Messy Data*. R package version 1.1.0.

Wilcox, R. R. (2017). *Introduction to robust estimation and hypothesis testing*. Elsevier, Waltham, MA, 4th edition. edition.

Winer, B. J., Brown, D. R., and Michels, K. M. (1991). *Statistical principles in experimental design*. McGraw-Hill series in psychology. McGraw-Hill, New York, 3rd edition.

Xie, Y. (2015). *Dynamic Documents with R and knitr*. Chapman and Hall/CRC, Boca Raton, Florida, 2nd edition. ISBN 978-1498716963.

Xie, Y. (2020a). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.18.

Xie, Y. (2020b). *knitr: A General-Purpose Package for Dynamic Report Generation in R*. R package version 1.28.

Zhu, H. (2019). *kableExtra: Construct Complex Table with 'kable' and Pipe Syntax*. R package version 1.1.0.