Multilevel Modeling in R (2.7)

A Brief Introduction to R, the multilevel package and the nlme package

Paul Bliese (pdbliese@gmail.com)

March 10, 2022

Table of Contents

1 Introduction

This is an introduction to how R can be used to perform multilevel analyses typical to organizational researchers. Multilevel analyses are applied to data that have some form of a nested structure. For instance, individuals may be nested within workgroups, or repeated measures may be nested within individuals, or firms may provide several years of data in what is referred to as panel data. Nested structures are often accompanied by some form of non-independence. In work settings, individuals in the same workgroup typically display some similarity with respect to performance or they provide similar responses to questions about aspects of the work environment. Likewise, in repeated measures data, individuals or firms usually display a high degree of similarity in responses over time. Non-independence may be considered either a nuisance variable or something to be substantively understood but working with nested data requires tools to deal with non-independence.

The term "multilevel analysis" is used to describe a set of analyses also referred to as random coefficient models, random effects, and mixed-effects models (see Bryk & Raudenbush, 1992; Clark & Linzer, 2014; Kreft & De leeuw, 1998; Pinheiro & Bates, 2000; Raudenbush & Bryk, 2002; Snijders & Bosker, 1999). Mixed-effects models (the term primarily used in this document) are not without limitations (e.g., Clark & Linzer, 2014), but are generally well-suited for dealing with non-independence (Bliese, Schepker, Essman & Ployhart, 2020). Prior to the widespread use of mixed-effects models, analysts used a variety of techniques to analyze data with nested structures and many of these techniques such as the econometric fixed-effect model are still widely used. In organizational research, mixed-effects models are often augmented by tools designed to quantify within-group agreement and group-mean reliability and the `multilevel` package contains many functions designed around testing within-group agreement and reliability.

This document is designed to cover a broad range of tools and approaches for analyzing multilevel data. Having worked for over two decades with both R and with multilevel data from numerous contexts, I routinely leverage different approaches and different packages depending upon the specific circumstances. Therefore, my goal in writing this document is to show how R can cover a wide range of inter-related topics related to multilevel analyses including:

- Data aggregation and merging for multilevel analyses
- Within-group agreement and reliability
- Contextual and basic econometric fixed-effect OLS models
- Covariance theorem decomposition of correlations
- Random Group Resampling
- Mixed Effects Models for nested group data
- Variants of Mixed Effects Models for Repeated Measures Data

Some of the basic analyses can conducted using R's base packages, but many of the analyses use functions in the `multilevel` package. As a broad overview, the `multilevel` package provides (a) functions for estimating within-group agreement and reliability indices, (b) functions for manipulating multilevel and longitudinal (panel) data, (c) simulations for

estimating power and generating multilevel data, and (d) miscellaneous functions for estimating reliability and performing simple calculations and data transformations. The `multilevel` package also contains several datasets to illustrate concepts.

The other library that is frequently used is the non-linear and linear mixed-effects (`nlme`) model package, (Pinheiro & Bates, 2000). The `nlme` package provides functions to estimate a variety of models for both data nested in groups and for repeated measures data collected over time (growth models). Functions in the `nlme` package have remarkable flexibility and can estimate a variety of alternative statistical models. In some cases, the `lme4` package developed by Doug Bates after the `nlme` package provides additional flexibility, so some functions from the `lme4` package are also detailed. I tend to use `lme4` when dealing with dichotomous dependent variables, or when data are partially or fully crossed, or when I want to generate an interaction plot (many more recent plotting packages were designed to work with `lme4` rather than `nlme`).

2 Reading data from files

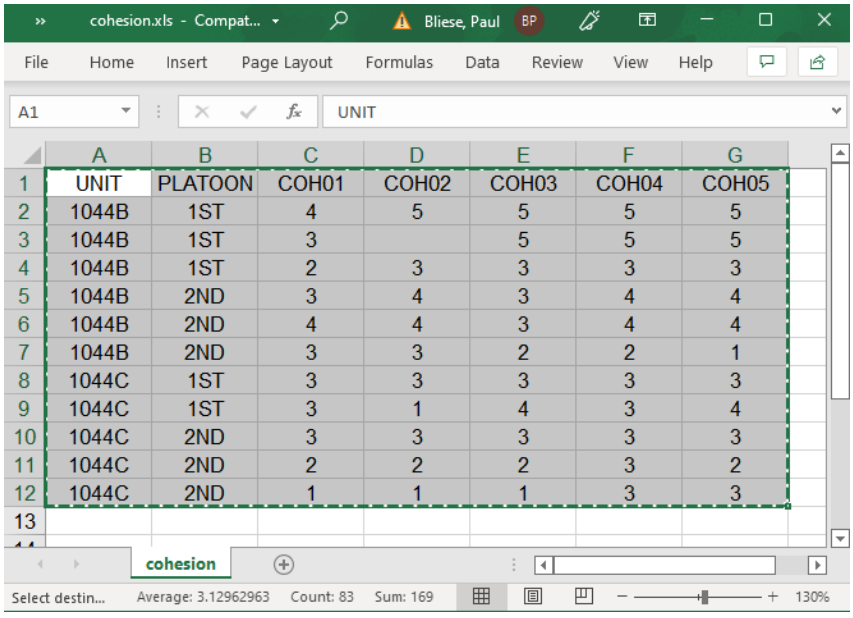
Before detailing multilevel analyses, I provide a short section on reading in data. There are numerous options for reading in data, so this section is in no way exhaustive. I provide what has been a simple and reliable way to import external files into dataframes.

In almost all cases working with research partners either in industry or academia, I have found that EXCEL files are a common platform particularly since EXCEL can read comma-delimited (csv) files. One additional advantage to EXCEL is that it is easy to quickly scan the data file for potential problems. I tend to avoid bringing in columns containing large amounts of text, and I often add an additional row under the header row with new R-friendly names (some research partners provide column headers the length of a small novel).

2.1.1 Reading data directly from EXCEL (Windows and MAC)

2.1.1.1 Windows

Consider the following data and notice how it has been highlighted and copied into the Window's clipboard (Ctrl-C):



	A	B	C	D	E	F	G
1	UNIT	PLATOON	COH01	COH02	COH03	COH04	COH05
2	1044B	1ST	4	5	5	5	5
3	1044B	1ST	3		5	5	5
4	1044B	1ST	2	3	3	3	3
5	1044B	2ND	3	4	3	4	4
6	1044B	2ND	4	4	3	4	4
7	1044B	2ND	3	3	2	2	1
8	1044C	1ST	3	3	3	3	3
9	1044C	1ST	3	1	4	3	4
10	1044C	2ND	3	3	3	3	3
11	1044C	2ND	2	2	2	3	2
12	1044C	2ND	1	1	1	3	3
13							

Once the file is in the Windows "clipboard", the following command reads the data into R:

```
> cohesion<-read.table(file="clipboard",sep="\t",header=T)
```

An even simpler variation is to use:

```
> cohesion<-read.delim(file="clipboard")
```

The `read.delim` function is variant of `read.table` that assumes the data are tab-delimited with a header. I have found that this simple approach covers about 95% of all my data entry needs to include importing either csv or EXCEL files with tens of thousands of observations.

2.1.1.2 MAC

If using a MAC, the basic ideas are the same, but the clipboard is accessed differently using pipe.

```
> cohesion<-read.delim(pipe("pbpaste"))
```

2.1.2 Reading external csv files with `file.choose` (Windows and MAC)

In cases where datasets are too large to read into EXCEL using the `file.choose()` function with `read.csv` or other `read.table` functions helps having to specify the path as in:

```
>cohesion<-read.csv(file.choose())
```

Using `file.choose()` opens the graphic user interface (gui) so one can select the file using a mouse or other device. This option is particularly useful when data are stored in complex network file structures.

2.1.3 Writing R files to EXCEL (Windows and MAC)

2.1.3.1 Windows

Because the "`clipboard`" option also works with `write.table` it is also a useful way to export the results of data analyses to EXCEL or other programs. For instance, if we create a correlation matrix from the cohesion data set, we can export this correlation table directly to EXCEL.

```
> CORMAT<-cor(cohesion[,3:7],use="pairwise.complete.obs")
> CORMAT
          COH01     COH02     COH03     COH04     COH05
COH01 1.0000000 0.7329843 0.6730782 0.4788431 0.4485426
COH02 0.7329843 1.0000000 0.5414305 0.6608190 0.3955316
COH03 0.6730782 0.5414305 1.0000000 0.7491526 0.7901837
COH04 0.4788431 0.6608190 0.7491526 1.0000000 0.9036961
COH05 0.4485426 0.3955316 0.7901837 0.9036961 1.0000000

> write.table(CORMAT,file="clipboard",sep="\t",col.names=NA)
```

Going to EXCEL and issuing the Windows "paste" command (or Ctrl-V) will insert the matrix into the EXCEL worksheet. Note the somewhat counter-intuitive use of `col.names=NA` in this example. This command does *not* mean omit the column names (achieved using `col.names=F`); instead the command puts an extra blank in the first row of the column names to line up the column names with the correct columns. Alternatively, one can use the option `row.names=F` to omit the row numbers.

Written objects may be too large for the default memory limit of the Window's clipboard. For instance, writing the full `bh1996` dataset from the `multilevel` package into the clipboard results in the following error (truncated):

```
> library(multilevel)
> data(b1996)  #Bring data from the library to the workspace
> write.table(bh1996,file="clipboard",sep="\t",col.names=NA)
Warning message:
In write.table(x, file, nrow(x),... as.integer(quote),  :
  clipboard buffer is full and output lost
```

To increase the size of the clipboard to 1.5MG (or any other arbitrary size), the `"clipboard"` option can be modified as follows: `"clipboard-1500"`. The options surrounding the use of the `clipboard` are specific to various operating systems and may change with different versions of R so it will be worth periodically referring to the help files.

2.1.3.2 MAC

If using a MAC, the "clipboard" option does not work, so the command line would be:

```
> write.table(bh1996, file=pipe("pbcopy"),sep="\t",col.names=NA)
```

Unlike Windows, the pipe option does not appear to need to be resized to accommodate large files.

2.1.4 The `foreign` package and SPSS files

The `foreign` package contains functions to import SPSS, SAS, Stata and minitab files. Help files are available for different formats. Below is a command to bring in an SPSS file as a dataframe and numbers (e.g., 4) instead of the number's value label (e.g., "agree").

```
> library(foreign)
> help(read.spss)     #look at the documentation on read.spss
> cohesion<-read.spss(file.choose(),use.value.labels=F, to.data.frame=T)
> cohesion
    UNIT PLATOON COH01 COH02 COH03 COH04 COH05
1  1044B     1ST     4     5     5     5     5
2  1044B     1ST     3    NA     5     5     5
3  1044B     1ST     2     3     3     3     3
4  1044B     2ND     3     4     3     4     4
5  1044B     2ND     4     4     3     4     4
6  1044B     2ND     3     3     2     2     1
7  1044C     1ST     3     3     3     3     3
8  1044C     1ST     3     1     4     3     4
9  1044C     2ND     3     3     3     3     3
10 1044C     2ND     2     2     2     3     2
11 1044C     2ND     1     1     1     3     3
```

2.1.5 Checking your dataframes with `str` and `summary`

With small data sets it is easy to verify that the data has been read in correctly. Often, however, one will be working with large data sets that are difficult to visual verify. Consequently, functions such as `str` (structure) and `summary` provide easy ways to examine dataframes.

```
> str(cohesion)
`data.frame':   11 obs. of  7 variables:
 $ UNIT   : Factor w/ 2 levels "1044B","1044C": 1 1 1 1 1 1 2 2 2 2 ...
 $ PLATOON: Factor w/ 2 levels "1ST","2ND": 1 1 1 2 2 2 1 1 2 2 ...
 $ COH01  : int  4 3 2 3 4 3 3 3 3 2 ...
 $ COH02  : int  5 NA 3 4 4 3 3 1 3 2 ...
 $ COH03  : int  5 5 3 3 3 2 3 4 3 2 ...
 $ COH04  : int  5 5 3 4 4 2 3 3 3 3 ...
 $ COH05  : int  5 5 3 4 4 1 3 4 3 2 ...

> summary(cohesion)
    UNIT   PLATOON     COH01           COH02          COH03
 1044B:6   1ST:5   Min.   :1.000   Min.   :1.00   Min.   :1.000
 1044C:5   2ND:6   1st Qu.:2.500   1st Qu.:2.25   1st Qu.:2.500
                   Median :3.000   Median :3.00   Median :3.000
                   Mean   :2.818   Mean   :2.90   Mean   :3.091
                   3rd Qu.:3.000   3rd Qu.:3.75   3rd Qu.:3.500
                   Max.   :4.000   Max.   :5.00   Max.   :5.000
```

```
                                    NA's   :1.00
     COH04           COH05
 Min.   :2.000   Min.   :1.000
 1st Qu.:3.000   1st Qu.:3.000
 Median :3.000   Median :3.000
 Mean   :3.455   Mean   :3.364
 3rd Qu.:4.000   3rd Qu.:4.000
 Max.   :5.000   Max.   :5.000
```

2.1.6 Loading data from packages

One of the useful attributes of R is that the data used in the examples are almost always available to the user. These data are associated with specific packages. For instance, the multilevel package uses a variety of data files to illustrate specific functions. To gain access to these data, one uses the `data` command:

```
>data(package="multilevel")
```

This command lists the data sets associated with the multilevel package, and the command

```
>data(bh1996, package="multilevel")
```

copies the `bh1996` data set to the workspace making it possible to work with the `bh1996` dataframe. If a package has been attached by the `library` function its datasets are automatically included in the search, so that if

```
>library(multilevel)
```

has been run, then

```
>data(bh1996)
```

copies the data from the package to the workspace without specifying the package.

2.2 A Brief Review of Matrix Brackets

One of the unique aspects of R is the use of matrix brackets to access various parts of a dataframe. While the bracket notation may initially appear cumbersome, mastering the use of matrix brackets provides considerable control.

The overall notation is [rows, columns]. So accessing rows 1,5,and 8 and columns 3 and 4 of the `cohesion` dataframe would be done like so:

```
> cohesion[c(1,5,8),3:4]
  COH01 COH02
1     4     5
5     4     4
8     3     1
```

Alternatively, we can specify the column names (this helps avoid picking the wrong columns).

```
> cohesion[c(1,5,8),c("COH01","COH02")]
  COH01 COH02
1     4     5
5     4     4
8     3     1
```

It is often useful to pick specific rows that meet some criteria. So, for example, we might want to pick rows that are from the 1ST PLATOON

```
> cohesion[cohesion$PLATOON=="1ST",]
   UNIT PLATOON COH01 COH02 COH03 COH04 COH05
1 1044B     1ST     4     5     5     5     5
2 1044B     1ST     3    NA     5     5     5
3 1044B     1ST     2     3     3     3     3
7 1044C     1ST     3     3     3     3     3
8 1044C     1ST     3     1     4     3     4
```

Upon inspection, we might want to further refine our choice and exclude missing values. We do this by adding another condition using `AND` operator `"&"` along with the `NOT` operator `"!".`

```
> cohesion[cohesion$PLATOON=="1ST"&!is.na(cohesion$COH02),]
   UNIT PLATOON COH01 COH02 COH03 COH04 COH05
1 1044B     1ST     4     5     5     5     5
3 1044B     1ST     2     3     3     3     3
7 1044C     1ST     3     3     3     3     3
8 1044C     1ST     3     1     4     3     4
```

These simple examples should provide an idea of how to subset large datasets when conducting analyses.

3 Multilevel Analyses

The remainder of this document illustrates how R can be used in multilevel modeling beginning with several R functions particularly useful for preparing data for subsequent analyses

3.1 Multilevel data manipulation functions

3.1.1 The `merge` Function

One of the key data manipulation tasks that must be accomplished prior to estimating several of the multilevel models (specifically contextual models and mixed-effects models) is that group-level variables must be "assigned down" to the individual. To make a dataframe containing both individual and group-level variables, one typically begins with two separate dataframes. One dataframe contains individual-level data, and the other dataframe contains group-level data. Combining these two dataframes using a group identifying variable common to both produces a single dataframe containing both individual and group data. In R, combining dataframes is accomplished using the `merge` function.

For instance, consider the `cohesion` data introduced when showing how to read data from external files. The `cohesion` data is included as a multilevel data set, so we can use the `data` function to bring it from the multilevel package to the working environment

```
>data(cohesion)
>cohesion
    UNIT PLATOON COH01 COH02 COH03 COH04 COH05
1  1044B     1ST     4     5     5     5     5
2  1044B     1ST     3    NA     5     5     5
3  1044B     1ST     2     3     3     3     3
4  1044B     2ND     3     4     3     4     4
```

```
5  1044B     2ND     4     4     3     4     4
6  1044B     2ND     3     3     2     2     1
7  1044C     1ST     3     3     3     3     3
8  1044C     1ST     3     1     4     3     4
9  1044C     2ND     3     3     3     3     3
10 1044C     2ND     2     2     2     3     2
11 1044C     2ND     1     1     1     3     3
```

Now assume that we have another dataframe with platoon sizes. We can create this dataframe as follows:

```
> group.size<-data.frame(UNIT=c("1044B","1044B","1044C","1044C"),
PLATOON=c("1ST","2ND","1ST","2ND"),PSIZE=c(3,3,2,3))
> group.size  #look at the group.size dataframe
   UNIT PLATOON PSIZE
1 1044B     1ST     3
2 1044B     2ND     3
3 1044C     1ST     2
4 1044C     2ND     3
```

To create a single file (`new.cohesion`) that contains both individual and platoon information, use the `merge` command.

```
> new.cohesion<-merge(cohesion,group.size,by=c("UNIT","PLATOON"))
> new.cohesion
     UNIT PLATOON COH01 COH02 COH03 COH04 COH05 PSIZE
1  1044B     1ST     4     5     5     5     5     3
2  1044B     1ST     3    NA     5     5     5     3
3  1044B     1ST     2     3     3     3     3     3
4  1044B     2ND     3     4     3     4     4     3
5  1044B     2ND     4     4     3     4     4     3
6  1044B     2ND     3     3     2     2     1     3
7  1044C     1ST     3     3     3     3     3     2
8  1044C     1ST     3     1     4     3     4     2
9  1044C     2ND     3     3     3     3     3     3
10 1044C     2ND     2     2     2     3     2     3
11 1044C     2ND     1     1     1     3     3     3
```

Notice that every individual now has a value for `PSIZE` – a value that reflects the number of individuals in the platoon.

In situations where there is a single unique group identifier, the `by` option can be simplified to include just one variable. For instance, if the group-level data had reflected values for each UNIT instead of PLATOON nested in unit, the by option would simply read `by="UNIT"`. In the case of PLATOON, however, there are numerous platoons with the same name (1ST, 2ND), so unique platoons need to be identified within the nesting of the larger UNIT.

3.1.2 The `aggregate` function

In many cases in multilevel analyses, we create a group-level variable by mean aggregating individual responses. Consequently, the `aggregate` function is often used in combination with the `merge` function. In our cohesion example, we can assign platoon means for the variables COH01 and COH02 back to the individuals using `aggregate` and `merge`.

The first step is to create a dataframe with group means using the `aggregate` function. The `aggregate` function has three key arguments: the first is matrix of variables to convert to group-level variables. Second is the grouping variable(s) as a list, and third is the function (`mean`, `var`, `length`, etc.) executed on the variables. To calculate the means of COH01 and COH02 (columns 3 and 4 of the cohesion dataframe) issue the command:

```
> TEMP<-aggregate(cohesion[,3:4],list(cohesion$UNIT,cohesion$PLATOON),mean)
> TEMP
  Group.1 Group.2    COH01    COH02
1   1044B     1ST 3.000000       NA
2   1044C     1ST 3.000000 2.000000
3   1044B     2ND 3.333333 3.666667
4   1044C     2ND 2.000000 2.000000
```

Notice that COH02 has an "NA" value for the mean. The NA value occurs because there was a missing value in the individual-level file. If we decide to base the group mean on the non-missing individual values from group members we can add the parameter `na.rm=T`, to designate that NA values should be removed prior to calculating the group mean.

```
> TEMP<-aggregate(cohesion[,3:4],list(cohesion$UNIT,cohesion$PLATOON),
  mean,na.rm=T)
> TEMP
  Group.1 Group.2    COH01    COH02
1   1044B     1ST 3.000000 4.000000
2   1044C     1ST 3.000000 2.000000
3   1044B     2ND 3.333333 3.666667
4   1044C     2ND 2.000000 2.000000
```

To merge the `TEMP` dataframe with the `new.cohesion` dataframe, we need to align the merge columns from both dataframes and control how the merge handles variables with the same names using the `suffixes= c("",".G")` option which leaves the variable name unchanged in the first dataframe but adds a .G suffix on the second dataframe.

```
> final.cohesion<-merge(new.cohesion,TEMP,by.x=c("UNIT","PLATOON"),
+ by.y=c("Group.1","Group.2"),suffixes=c("",".G"))
> final.cohesion
    UNIT PLATOON COH01 COH02 COH03 COH04 COH05 PSIZE  COH01.G  COH02.G
1  1044B     1ST     4     5     5     5     5     3 3.000000 4.000000
2  1044B     1ST     3    NA     5     5     5     3 3.000000 4.000000
3  1044B     1ST     2     3     3     3     3     3 3.000000 4.000000
4  1044B     2ND     3     4     3     4     4     3 3.333333 3.666667
5  1044B     2ND     4     4     3     4     4     3 3.333333 3.666667
6  1044B     2ND     3     3     2     2     1     3 3.333333 3.666667
7  1044C     1ST     3     3     3     3     3     2 3.000000 2.000000
8  1044C     1ST     3     1     4     3     4     2 3.000000 2.000000
9  1044C     2ND     3     3     3     3     3     3 2.000000 2.000000
10 1044C     2ND     2     2     2     3     2     3 2.000000 2.000000
11 1044C     2ND     1     1     1     3     3     3 2.000000 2.000000
```

The `aggregate` and `merge` functions provide tools necessary to manipulate data and prepare it for subsequent multilevel analyses. Again, note that this illustration uses a relatively complex situation where there are two levels of nesting (Platoon within Unit). In cases where

there is only one grouping variable (for example, UNIT) the commands for aggregate and merge contain the name of a single grouping variable. For instance,

```
>TEMP<-aggregate(cohesion[,3:4],list(cohesion$UNIT),mean,na.rm=T)
```

3.2 Within-Group Agreement and Reliability

The data used in this section are taken from Bliese, Halverson & Rothberg (2000) using the bhr2000 data set from the multilevel package.

```
> data(bhr2000)#imports the data into the working environment
> names(bhr2000)
 [1] "GRP"   "AF06"  "AF07"  "AP12"  "AP17"  "AP33"  "AP34"
 "AS14"  "AS15" "AS16"  "AS17"  "AS28"  "HRS"   "RELIG"
> nrow(bhr2000)
[1] 5400
```

The names function identifies 14 variables. The first one, GRP, is the group identifier. The variables in columns 2 through 12 are individual responses on 11 items that make up a leadership scale. HRS represents individuals' reports of work hours, and RELIG represents individuals' reports of the degree to which religion is a useful coping mechanism. The nrow command indicates that there are 5400 observations. To find out how many groups there are we can use the length command in conjunction with the unique command

```
> length(unique(bhr2000$GRP))
[1] 99
```

There are several functions in the multilevel library that are useful for calculating and interpreting agreement indices. These functions are rwg, rwg.j, rwg.sim, rwg.j.sim, rwg.j.lindell, awg, ad.m, ad.m.sim and rgr.agree. The rwg function calculates the James, Demaree & Wolf (1984) r_{wg} for single item measures; the rwg.j function calculates the James et al. (1984) $r_{wg(j)}$ for multi-item scales. The rwg.j.lindell function calculates $r^*_{wg(j)}$ (Lindell, & Brandt, 1997; 1999). The awg function calculates the a_{wg} agreement index proposed by Brown and Hauenstein (2005). The ad.m function calculates average deviation (AD) values for the mean or median (Burke, Finkelstein & Dusig, 1999).

A series of functions with "sim" in the name (rwg.sim, rwg.j.sim and ad.m.sim) can be used to simulate agreement values from a random null distributions to test for statistical significance agreement. The simulation functions are based on work by Dunlap, Burke and Smith-Crowe (2003); Cohen, Doveh and Eich (2001) and Cohen, Doveh and Nuham-Shani (2009). Finally, the rgr.agree function performs a Random Group Resampling (RGR) agreement test (see Bliese, et al., 2000).

In addition to the agreement measures, there are two multilevel reliability measures, ICC1 and ICC2 than can be used on ANOVA models. As Bliese (2000) and others (e.g., Kozlowski & Hattrup, 1992; Tinsley & Weiss, 1975) have noted, reliability measures such as the ICC(1) and ICC(2) are fundamentally different from agreement measures; nonetheless, they often provide complementary information to agreement measures, so this section illustrates the use of each of these functions using the dataframe bhr2000.

3.2.1 Agreement: r_{wg}, $r_{wg(j)}$, and $r^*_{wg(j)}$

Both the `rwg` and `rwg.j` functions are based upon the formulations described in James et al. (1984). he functions require three pieces of information. The first piece is the variable of interest (`x`), the second is the grouping variable (`grpid`), and third is the expected random variance (`ranvar`). The default estimate of `ranvar` is 2, which is the expected random variance based upon the rectangular distribution for a 5-point item (i.e., σ_{EU}^2) calculated using the formula ranvar=(A^2-1)/12 where A represents the number of response options associated with the scale anchors. See `help(rwg)`, James et al., (1984), or Bliese et al., (2000) for details on selecting appropriate `ranvar` values.

Below is an example using the `rwg` function to calculate agreement for the "coping using religion" item:

```
> RWG.RELIG<-rwg(bhr2000$RELIG,bhr2000$GRP,ranvar=2)
> RWG.RELIG[1:10,]  #examine first 10 rows of data
   grpid        rwg gsize
1      1 0.11046172    59
2      2 0.26363636    45
3      3 0.21818983    83
4      4 0.31923077    26
5      5 0.22064137    82
6      6 0.41875000    16
7      7 0.05882353    18
8      8 0.38333333    21
9      9 0.14838710    31
10    10 0.13865546    35
```

The function returns a dataframe with three columns. The first column contains the group names (`grpid`), the second column contains the 99 r_{wg} values – one for each group. The third column contains the group size. To calculate the mean r_{wg} value use the `summary` command:

```
> summary(RWG.RELIG)
     grpid             rwg              gsize
 1      : 1       Min.   :0.0000   Min.   :  8.00
 10     : 1       1st Qu.:0.1046   1st Qu.: 29.50
 11     : 1       Median :0.1899   Median : 45.00
 12     : 1       Mean   :0.1864   Mean   : 54.55
 13     : 1       3rd Qu.:0.2630   3rd Qu.: 72.50
 14     : 1       Max.   :0.4328   Max.   :188.00
 (Other):93
```

The `summary` command informs us that the average r_{wg} value is .186 and the range is from 0 to 0.433. By convention, values at or above 0.70 are considered good agreement, so there appears to be low agreement among individuals within the same work groups with respect to coping using religion. The `summary` command also provides information about the group sizes.

To calculate r_{wg} for work hours, the expected random variance (EV) needs to be changed from its default value of 2. Work hours was asked using an 11-point item, so EV based on the rectangular distribution (σ_{EU}^2) is 10.00 ($\sigma_{EU}^2=(11^2-1)/12$) – see the `rwg` help file for details).

```
> RWG.HRS<-rwg(bhr2000$HRS,bhr2000$GRP,ranvar=10.00)
> mean(RWG.HRS[,2])
```

```
[1] 0.7353417
```

There is apparently much higher agreement about work hours within groups than there was about using religion as a coping mechanism in this sample. By convention, this mean value would indicate agreement because r_{wg} (and $r_{wg(j)}$) values above .70 are considered to provide evidence of agreement.

The use of the `rwg.j` function is nearly identical to the use of the `rwg` function except that the first argument to `rwg.j` is a matrix instead of a vector. In the matrix, each column represents one item in the multi-item scale, and each row represents an individual response. For instance, columns 2-12 in `bhr2000` represent 11 items comprising a leadership scale. The items were assessed using 5-point response options (Strongly Disagree to Strongly Agree), so the expected random variance is (5^2-1)/12 or 2.

```
> RWGJ.LEAD<-rwg.j(bhr2000[,2:12],bhr2000$GRP,ranvar=2)
> summary(RWGJ.LEAD)
     grpid            rwg.j            gsize
 1       : 1       Min.   :0.7859   Min.   :  8.00
 10      : 1       1st Qu.:0.8708   1st Qu.: 29.50
 11      : 1       Median :0.8925   Median : 45.00
 12      : 1       Mean   :0.8876   Mean   : 54.55
 13      : 1       3rd Qu.:0.9088   3rd Qu.: 72.50
 14      : 1       Max.   :0.9440   Max.   :188.00
 (Other):93
```

Note that Lindell and colleagues (Lindell & Brandt, 1997, 1999; 2000; Lindell, Brandt & Whitney, 1999) have raised concerns about the mathematical underpinnings of the $r_{wg(j)}$ formula. Specifically, they note that this formula is based upon the Spearman-Brown reliability estimator. Generalizability theory provides a basis to believe that reliability should increase as the number of measurements increase, so the Spearman-Brown formula is defensible for measures of reliability. In contrast, there may be no theoretical grounds to believe that generalizability theory applies to measures of agreement. That is, there may be no reason to believe that agreement should increase as the number of measurements on a scale increase (but also see LeBreton, James & Lindell, 2005).

To address this potential concern with the $r_{wg(j)}$, Lindell and colleagues have proposed the $r^*_{wg(j)}$. The $r^*_{wg(j)}$ is calculated by substituting the average variance of the items in the scale into the numerator of r_{wg} formula in lieu of using the $r_{wg(j)}$ formula (r_{wg} = 1- Observed Group Variance/Expected Random Variance). Note that Lindell and colleagues also recommend against truncating the Observed Group Variance value so that it matches the Expected Random Variance value in cases where the observed variance is larger than the expected variance. Their modification results $r^*_{wg(j)}$ values being able to take on negative values. We can use the function `rwg.j.lindell` to estimate the $r^*_{wg(j)}$ values for leadership.

```
> RWGJ.LEAD.LIN<-rwg.j.lindell(bhr2000[,2:12],
bhr2000$GRP,ranvar=2)
> summary(RWGJ.LEAD.LIN)
     grpid          rwg.lindell         gsize
 1       : 1       Min.   :0.2502   Min.   :  8.00
 10      : 1       1st Qu.:0.3799   1st Qu.: 29.50
 11      : 1       Median :0.4300   Median : 45.00
 12      : 1       Mean   :0.4289   Mean   : 54.55
```

```
 13      : 1        3rd Qu.:0.4753   3rd Qu.: 72.50
 14      : 1        Max.   :0.6049   Max.   :188.00
 (Other):93
```

The average $r^*_{wg(j)}$ value of .43 is considerably lower than the average $r_{wg(j)}$ value of .89 listed earlier.

3.2.2 The a_{wg} Index

Brown and Hauenstein (2005) argue that the r_{wg} family of agreement indices have three major limitations: (1) the magnitude of the measures are dependent on sample size, (2) the scale used to assess the construct influences the magnitude of the measure, and (3) the use of the uniform null distribution is an invalid comparison upon which to base an estimate of agreement. To overcome these limitations, Brown and Hauenstein proposed the a_{wg} index as a multi-rater agreement measure analogous to Cohen's kappa. The a_{wg} index is calculated using the `awg` function.

The `awg` function has three arguments: `x`, `grpid`, and `range`. The `x` argument represents the item or scale upon which to calculate a_{wg} values. The `awg` function determines whether `x` is a vector (single item) or multiple item matrix (representing the items in a scale), and performs the a_{wg} calculation appropriate for the type of input. The second function, `grpid`, is a vector containing the group ids associated with the `x` argument. The third argument, `range`, represents the upper and lower limits of the response options. The `range` defaults to `c(1,5)` which represents a 5-point scale from (for instance) strongly disagree (1) to strongly agree (5).

The code below illustrates the use of the `awg` function for the multi-item leadership scale.

```
> AWG.LEAD<-awg(bhr2000[,2:12],bhr2000$GRP)
> summary(AWG.LEAD)
     grpid          a.wg            nitems      nraters        avg.grp.var
 1       : 1   Min.   :0.2223   Min.   :11   Min.   :  8.00   Min.   :0.2787
 10      : 1   1st Qu.:0.3654   1st Qu.:11   1st Qu.: 29.50   1st Qu.:0.4348
 11      : 1   Median :0.4193   Median :11   Median : 45.00   Median :0.5166
 12      : 1   Mean   :0.4125   Mean   :11   Mean   : 54.55   Mean   :0.5157
 13      : 1   3rd Qu.:0.4635   3rd Qu.:11   3rd Qu.: 72.50   3rd Qu.:0.5692
 14      : 1   Max.   :0.5781   Max.   :11   Max.   :188.00   Max.   :0.9144
 (Other):93
```

Notice that ratings of the a_{wg} tend to more similar in magnitude to the $r^*_{wg(j)}$ which likely reflects the facts that (a) large variances can result in negative ratings reflecting disagreement, and (b) the denominator for the measure is fundamentally based upon the idea of maximum possible variance (similarly to the $r^*_{wg(j)}$) rather than a uniform distribution.

One final note is that Brown and Hauenstein (2005) contend that the class of r_{wg} agreement indices should not be estimated in cases where group size (or number of raters) is less than the number of response options (scale anchors) associated with the items (A). In this example, A is 5 representing the scale anchors from strongly disagree to strongly agree. In contrast, however, Brown and Hauenstein (2005) state that it is appropriate to estimate a_{wg} on the number of anchors minus 1. The reason why a_{wg} can be estimated on smaller groups is that a_{wg} (unlike r_{wg}) uses a sample-based variance estimate in the denominator whereas r_{wg} uses a population-based variance estimate (recall that the formula for the rectangular variance distribution is ranvar=(A^2-1)/12 which represents a population-based value (σ_{EU}^2)). In the example there is no issue with group size given that the smallest group has eight members.

3.2.3 Significance testing using `rwg.sim` and `rwg.j.sim`

As noted in section 3.2.1, r_{wg} and $r_{wg(j)}$ values at or above .70 are conventionally considered providing evidence of within-group agreement. A series of studies by Charnes and Schriesheim (1995); Cohen, Doveh and Eick (2001); Dunlap, Burke, and Smith-Crowe (2003) and Cohen, Doveh and Nahum-Shani (2009) lay the groundwork for establishing tests of statistical significance for r_{wg} and $r_{wg(j)}$. The basic idea behind these simulations is to draw observations from a known null distribution (generally a uniform or rectangular null), and repeatedly estimate r_{wg} or $r_{wg(j)}$. Because the observations are drawn from a uniform random null, r_{wg} or $r_{wg(j)}$ estimates in the simulation should be zero. Occasionally, however, the r_{wg} or $r_{wg(j)}$ values will be larger than zero reflecting sampling variability associated with the specific attributes of the simulation. Repeatedly drawing random numbers and estimating r_{wg} and $r_{wg(j)}$ provides a way to calculate expected null values and confidence intervals.

The simulations conducted by Cohen et al., (2001) varied several factors, but the two factors found to be most important for the expected null values of the $r_{wg(j)}$ were (a) group size and (b) the number of items. Indeed, Cohen et al., (2001) found that the expected null $r_{wg(j)}$ values in the simulations differed considerably as group size and the number of items varied. These findings imply that the conventional value of .70 may be a reasonable cut-off value for significance with some configurations of group sizes and items but may not be reasonable for others. Thus, Cohen et al., (2001) recommended researchers simulate parameters based on the specific characteristics of the researchers' samples when determining whether $r_{wg(j)}$ values are significant.

In 2003, Dunlap and colleagues estimated 95% confidence intervals for the single item r_{wg} using the idea of simulating null distributions. Their work showed that the 95% confidence interval for the single item measure varied as a function of (a) group size and (b) the number of response options. In the case of 5 response options (e.g., strongly disagree, disagree, neither, agree, strongly agree), the 95% confidence interval estimate varied from 1.00 with a group of 3 to 0.12 for a group of 150. That is, one would need an r_{wg} estimate of 1.00 with groups of size three to be 95% certain the groups agreed more than chance levels, but with groups of size 150 any value equal to or greater than 0.12 would represent significant agreement.

The function `rwg.sim` provides a way to replicate the results presented by Dunlap and colleagues. For instance, to estimate the 95% confidence interval for a group of size 10 on an item with 5 response options one would provide the following parameters to the `rwg.sim` function keeping in mind that the results from a separate run will not match these results exactly because no random seed was set:

```
> RWG.OUT<-rwg.sim(gsize=10, nresp=5, nrep=10000)
> summary(RWG.OUT)
$rwg
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0000  0.0000  0.0000  0.1221  0.2167  0.8667

$gsize
[1] 10
$nresp
[1] 5
$nitems
[1] 1
$rwg.95
```

```
[1] 0.5277778
```

The results in the preceding example are based on 10,000 simulation runs. In contrast, Dunlap et al., (2003) used 100,000 simulation runs. Nonetheless, both Table 2 from Dunlap et al., (2003) and the example above suggest that 0.53 is the 95% confidence interval estimate for a group of size 10 with five response options.

Because the estimation of r_{wg} in the simulations produces a limited number of possible responses, the typical methods for establishing confidence intervals (e.g., the generic function `quantile`) cannot be used. Thus, the multilevel package provides a `quantile` method for the objects of class `agree.sim` created using `rwg.sim`. To obtain 90%, 95% and 99% confidence interval estimates (or any other values) one would issue the following command:

```
> quantile(RWG.OUT,c(.90,.95,.99))
  quantile.values confint.estimate
1            0.90        0.4222222
2            0.95        0.5277778
3            0.99        0.6666667
```

Cohen et al. (2009) expanded upon the work of Dunlap et al., (2003) and the early work by Cohen et al. (2001) by demonstrating how confidence interval estimation could be applied to multiple item scales in the case of $r_{wg(j)}$ values. The function `rwg.j.sim` is based upon the work of Cohen et al., (2009) and simulates $r_{wg(j)}$ values from a uniform null distribution for user supplied values of (a) group size, (b) number of items in the scale, and (c) number of response options on the items. Users also provide the number of simulation runs (repetitions) upon which to base the estimates. In most cases, the number of simulation runs will be 10,000 or more although the examples illustrated here will be limited to 1,000.

The final optional argument to `rwg.j.sim` is `itemcors`. If this argument is omitted, the simulated items used to comprise the scale are assumed to be independent (non-correlated). If the argument is provided, the items comprising the scale are simulated to reflect a given correlational structure. Cohen et al., (2001) showed that results based on independent (non-correlated) items were similar to results based on correlated items; nonetheless, the model with correlated items is more realistic and thereby preferable (see Cohen et al., 2009).

For an example of using `rwg.j.sim` with non-correlated items, consider a case where a researcher was estimating the expected value and confidence intervals of $r_{wg(j)}$ on a sample where group size was 15 using a 7-item scale with 5 response options for the items (A=5). The call to `rwg.j.sim` would be:

```
> RWG.J.OUT<-rwg.j.sim(gsize=15,nitems=7,nresp=5,nrep=1000)

> summary(RWG.J.OUT)
$rwg.j
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.000000 0.000000 0.009447 0.161800 0.333900 0.713700
$gsize
[1] 15
$nresp
[1] 5
$nitems
[1] 7
$rwg.j.95
```

```
[1] 0.5559117
```

In this example, the upper expected 95% confidence interval is 0.56. This is lower than 0.70, and suggests that in this case the value of 0.70 might be too stringent. Based on this simulation, one might justifiably conclude that a value of 0.56 is evidence of significant agreement ($p<.05$). Using the simulation, one can show that as group size increases and the number of items increase, the criteria for what constitutes significant agreement decreases.

To illustrate how significance testing of $r_{wg(j)}$ might be used in a realistic setting, we will examine whether group members agreed about three questions specific to mission importance in the `lq2002` data set. These data were also analyzed in Cohen et al., 2009. We begin by estimating the mean $r_{wg(j)}$ for the 49 groups in the sample and obtaining a value of .58. This value is below the .70 conventional criteria and suggests a lack of agreement.

```
> RWG.J<-rwg.j(lq2002[,c("TSIG01","TSIG02","TSIG03")],
  lq2002$COMPID,ranvar=2)
> summary(RWG.J)
     grpid        rwg.j            gsize
 10     : 1   Min.   :0.0000   Min.   :10.00
 13     : 1   1st Qu.:0.5099   1st Qu.:18.00
 14     : 1   Median :0.6066   Median :30.00
 15     : 1   Mean   :0.5847   Mean   :41.67
 16     : 1   3rd Qu.:0.7091   3rd Qu.:68.00
 17     : 1   Max.   :0.8195   Max.   :99.00
 (Other):43
```

To determine whether the value of .58 is significant, we use the `rwg.j.sim` function using item correlations and average group size (41.67 rounded to 42). In this case, notice the simulation suggests that a value of .35 is significant providing evidence of significant agreement.

```
> RWG.J.OUT<-rwg.j.sim(gsize=42,nitems=3,nresp=5,
   itemcors=cor(lq2002[,c("TSIG01","TSIG02","TSIG03")]),
   nrep=1000)
> summary(RWG.J.OUT)
$rwg.j
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
0.000000 0.000000 0.007224 0.088520 0.162500 0.548600
$gsize
[1] 42
$nresp
[1] 5
$nitems
[1] 3
$rwg.j.95
[1] 0.346875
```

3.2.4 Average Deviation (AD) Agreement using `ad.m`

Burke, Finkelstein and Dusig (1999) proposed using average deviation (AD) indices as measures of within-group agreement. Cohen et al., (2009) note that AD indices are also referred to as Mean or Median Average Deviation. AD indices are calculated by first computing the absolute deviation of each observation from the mean or median. Second, the absolute deviations

are averaged to produce a single AD estimate for each group. The formula for AD calculation on a single item is:

$AD = \Sigma|x_{ij} - X_j|/N$

where x_{ij} represents an individual observation (i) in group j; X_j represents the group mean or median, and N represents the group size. When AD is calculated on a scale, the AD formula above is estimated for each item on the scale, and each item's AD value is averaged to compute the scale AD score.

AD values are considered practically significant when the values are less than A/6 where A represents the number of response options on the item. For instance, A is 5 when items are asked on a Strongly Disagree, Disagree, Neither, Agree and Strongly Agree format so any value less than .83 (5/6) would be considered practically significant.

The function `ad.m` is used to compute the average deviation of the mean or median. The function requires the two arguments, `x` and `grpid`. The `x` argument represents the item or scale upon which to estimate the AD value. The `ad.m` function determines whether `x` is a vector (single item) or multiple item matrix (multiple items representing a scale), and performs the AD calculation appropriate for the nature of the input variable. The second function, `grpid`, is a vector containing the group ids of the `x` argument. The third argument is optional. The default value is to compute the Average Deviation of the mean. The other option is to change the `type` argument to "`median`" and compute the Average Deviation of the median.

For instance, recall that columns 2-12 in `bhr2000` represent 11 items comprising a leadership scale. The items were assessed using 5-point response options (Strongly Disagree to Strongly Agree), so the practical significance of the AD estimate is 5/6 or 0.83. The AD estimates for the first five groups and the mean of the overall sample are provided below:

```
> data(bhr2000)
> AD.VAL <- ad.m(bhr2000[, 2:12], bhr2000$GRP)
> AD.VAL[1:5,]
  grpid      AD.M gsize
1     1 0.8481366    59
2     2 0.8261279    45
3     3 0.8809829    83
4     4 0.8227542    26
5     5 0.8341355    82
> mean(AD.VAL[,2:3])
      AD.M      gsize
 0.8690723 54.5454545
```

Two of the estimates are less than 0.833 suggesting these two groups (2 and 4) agree about ratings of leadership. The overall AD estimate is 0.87, which is also higher than 0.83 and suggests a general lack of agreement.

The AD value estimated using the median instead of the mean, in contrast, suggests practically significant agreement for the sample as a whole.

```
> AD.VAL <- ad.m(bhr2000[, 2:12], bhr2000$GRP,type="median")
> mean(AD.VAL[,2:3])
      AD.M      gsize
 0.8297882 54.5454545
```

To use the ad.m function for single item variables such as the work hours (HRS) variable in the bhr2000 data simply include the HRS vector instead of a matrix as the first argument. Recall that work hours is asked on an 11-point response format scale so practical significance is 11/6 or 1.83. The average observed AD value of 1.25 suggests within-group agreement about work hours across the sample as a whole.

```
> AD.VAL.HRS <- ad.m(bhr2000$HRS, bhr2000$GRP)
> mean(AD.VAL.HRS[,2:3])
     AD.M     gsize
 1.249275 54.545455
```

3.2.5 Significance testing ad.m.sim

The function ad.m.sim is used to simulate AD values and test for significance of various combinations of group size, number of response options and number of items in multiple-item scales. The ad.m.sim function is similar to the rwg.sim and rwg.j.sim functions used to test the significance of r_{wg} and $r_{wg(j)}$; however, unlike the functions for the two forms of the r_{wg}, the ad.m.sim function works with both single items and multiple-item scales.

The ad.m.sim function is based upon the work of Cohen et al. (2009) and of Dunlap et al., (2003). The function simulates AD values from a uniform null distribution for user supplied values of (a) group size, (b) number of items in the scale, and (c) number of response options on the items. Based on Cohen et al. (2009), the final optional parameter can include a correlation matrix when simulating multiple-item scales. The user also provides the number of simulation runs (repetitions) upon which to base the estimates. Again in practice, the number of simulation runs will typically be 10,000 or more although the examples illustrated here will be limited to 1,000.

To illustrate the ad.m.sim function, consider the 11 leadership items in the bhr2000 dataframe. Recall the AD value based on the mean suggested that groups failed to agree about leadership. In contrast, the AD value based on the median suggested that groups agreed. To determine whether the overall AD value based on the mean is statistically significant, one can simulate data matching the characteristics of the bhr2000 sample:

```
> AD.SIM<-ad.m.sim(gsize=55,nresp=5,
itemcors=cor(bhr2000[,2:12]),type="mean",nrep=1000)
> summary(AD.SIM)
$ad.m
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.087   1.182   1.208   1.209   1.236   1.340

$gsize
[1] 55

$nresp
[1] 5

$nitems
[1] 11

$ad.m.05
[1] 1.138212
```

```
$pract.sig
[1] 0.8333333
```

The simulation suggests that any AD mean value less than or equal to 1.14 is statistically significant. Thus, while the AD value for the leadership items (0.87) may not meet the criteria for practical significance, it does for statistical significance. As with the r_{wg} simulation functions, the `ad.m.sim` function has a specifically associated `quantile` function to identify different cut-off points. The example below illustrates how to identify values corresponding to the .90 (.10), .95 (.05) and .99 (.01) significance levels. That is, to be 99% certain that a value was significant, it would need to be smaller than or equal to 1.114.

```
> quantile(AD.SIM,c(.10,.05,.01))
  quantile.values confint.estimate
1            0.10         1.155763
2            0.05         1.138212
3            0.01         1.114170
```

3.2.6 Agreement: Random Group Resampling

The final agreement related function in the multilevel library is `rgr.agree`. In some ways this function is similar to the `rwg.j.sim` function in that it uses repeated simulations of data to draw inferences about agreement. The difference is that the `rgr.agree` function uses the actual group data, while the `rwg.j.sim` function simulates from an expected distribution (the uniform null).

The `rgr.agree` function (a) uses Random Group Resampling to create pseudo groups and calculate pseudo group variances, (b) estimates actual group variances, and (c) performs tests of significance to determine whether actual group and pseudo group variances differ. To use `rgr.agree`, one must provide three variables. The first is a vector representing the variable upon which one wishes to estimate agreement. The second is group membership (`grpid`). The third parameter is the number of pseudo groups to generate.

The third parameter requires a little explanation, because in many cases the number of pseudo groups returned in the output will not exactly match the third parameter. For instance, in our example, we will request 1000 pseudo groups, but the output will return only 990. This is because the `rgr.agree` algorithm creates pseudo groups that are identical in size characteristics to the actual groups. In so doing, the algorithm creates sets of pseudo groups in "chunks." The size of each chunk is based upon the number of actual groups. So, if there are 99 actual groups, then the total number of pseudo groups must be evenly divisible by 99. Nine-hundred-and-ninety is evenly divisible by 99, while 1000 is not. Rather than require the user to determine what is evenly divisible by the number of groups, `rgr.agree` will do this automatically. Below is an example of using `rgr.agree` on the work hours variable.

```
> RGR.HRS<-rgr.agree(bhr2000$HRS,bhr2000$GRP,1000)
```

The first step is to create an RGR Agreement object named `RGR.HRS`. The object contains several components. In most cases, however, users will be interested in the estimated z-value indicating whether the within-group variances from the actual groups are smaller than the variances from the pseudo groups. A useful way to get this information is to use the `summary`

command. When `summary` is applied to the RGR agreement object it provides standard deviations, variance estimates, an estimate of the z-value, and upper and lower confidence intervals.

```
> summary(RGR.HRS)
$"Summary Statistics for Random and Real Groups"
  N.RanGrps Av.RanGrp.Var SD.Rangrp.Var Av.RealGrp.Var  Z-value
1       990      3.322772      0.762333       2.646583 -8.82554

$"Lower Confidence Intervals (one-tailed)"
    0.5%       1%     2.5%       5%      10%
1.648162 1.795134 1.974839 2.168830 2.407337

$"Upper Confidence Intervals (one-Tailed)"
     90%      95%    97.5%      99%    99.5%
4.251676 4.545078 4.832813 5.642410 5.845143
```

The first section of the summary provides key statistics for contrasting within-group variances from real group with within-group variances from random groups. The second and third sections provide lower and upper confidence intervals. Keep in mind that if one replicates this example one is likely to get slightly different results because no random seed was set. While the exact numbers may differ, the conclusions drawn should be the same.

The first section of the summary shows that the average within-group variance for the random groups was 3.32 with a Standard Deviation of 0.76. In contrast, the average within-group variance for the real groups was considerably smaller at 2.65. The estimated z-value suggests that, overall, the within-group variances in ratings of work hours from real groups were significantly smaller than the within-group variances from the random groups. These results suggest that group members agree about work hours. Recall that a z-value greater than or less than 1.96 signifies significance at $p<.05$, two-tailed.

The upper and lower confidence interval information allows one to estimate whether specific groups do or do not display agreement. For instance, only 5% of the pseudo groups had a variance less than 2.17. Thus, if we observed a real group with a variance smaller than 2.17, we could be 95% confident this group variance was smaller than the variances from the pseudo groups. Likewise, if we want to be 90% confident we were selecting groups showing agreement, we could identify real groups with variances less than 2.41.

To see which groups meet this criterion, use the `tapply` function in conjunction with the `sort` function. The `tapply` function partitions the first variable by levels of the second variable and performs a specified function much like the `aggregate` function (see section 3.1.2). Below we partition HRS into separate Groups (GRP) and calculate the variance for each group (`var`). Using `sort` in front of this command makes the output easier to read.

```
> sort(tapply(bhr2000$HRS,bhr2000$GRP,var))
       33        43        38        19         6        39        69        17
0.8242754 1.0697636 1.1295681 1.2783251 1.3166667 1.3620690 1.4566667 1.4630282
       20        99        98        44         4        53        61        63
1.5009740 1.5087719 1.5256410 1.5848739 1.6384615 1.6503623 1.6623656 1.7341430
```

```
       66        14        76        71        21        18        59        50
1.7354302 1.7367089 1.7466200 1.7597586 1.7808500 1.7916027 1.8112599 1.8666667
       48        60        83         8        22         2        75        11
1.8753968 1.9267300 1.9436796 1.9476190 1.9679144 2.0282828 2.1533101 2.1578947
       96        23        54        47        55        26        74        57
2.1835358 2.1864802 2.2091787 2.2165242 2.2518939 2.2579365 2.2747748 2.2808858
       45        97        64        35        32        41         1        24
2.2975687 2.3386525 2.3535762 2.3563495 2.3747899 2.4096154 2.4284044 2.4391678
       82        37        81        68        42        73        34        25
2.4429679 2.4493927 2.5014570 2.5369458 2.5796371 2.6046154 2.6476418 2.6500000
       93        62        92        12        40        88         5        29
2.6602168 2.7341080 2.7746106 2.7906404 2.7916084 2.8505650 2.8672087 2.8748616
       85        70        77        51         3        13        79        87
2.8974843 2.9938483 3.0084034 3.0333333 3.0764032 3.1643892 3.1996997 3.2664569
        7        95        78        84        46        27        36        15
3.2712418 3.2804878 3.3839038 3.3886048 3.4084211 3.4309008 3.4398064 3.4425287
       89        16        58        49         9        31        90        72
3.4444444 3.4461538 3.4949020 3.5323440 3.6258065 3.6798419 3.8352838 3.9285714
       91        80        86        10        94        28        30        56
3.9565960 3.9729730 3.9753195 4.0336134 4.0984900 4.0994152 4.6476190 4.7070707
       65        52        67
4.7537594 5.2252964 5.3168148
```

If we start counting from group 33 (the group with the lowest variance of 0.82) we find 46 groups with variances smaller than 2.41. That is, we find 46 groups that have smaller than expected variance using the 90% confidence estimate.

It may also be interesting to see what a "large" variance is when defined in terms of pseudo group variances. This information is found in the third part of the summary of the `RGR.HRS` object. A variance of 4.55 is in the upper 95% of all random group variances. Given this criterion, we have five groups that meet or exceed this standard. In an applied setting, one might be very interested in examining this apparent lack of agreement in groups 30, 56, 65, 52 and 67. That is, one might be interested in determining what drives certain groups to have very large differences in how individuals perceive work hours.

Finally, for confidence intervals not given in the summary, one can use the `quantile` function with the random variances (`RGRVARS`) in the `RGR.HRS` object. For instance to get the lower .20 confidence interval:

```
> quantile(RGR.HRS$RGRVARS, c(.20))
     20%
2.695619
```

Note that `rgr.agree` only works on vectors. Consequently, to use `rgr.agree` with the leadership scale we would need to create a leadership scale score. We can do this using the

rowMeans function. We will create a leadership scale (LEAD) and put it in the bhr2000 dataframe, so the specific command we issue is:

```
>bhr2000$LEAD<-rowMeans(bhr2000[,2:12],na.rm=TRUE)
```

Now that we have created a leadership scale score, we can perform the RGR agreement analysis on the variable.

```
> summary(rgr.agree(bhr2000$LEAD,bhr2000$GRP,1000))

$"Summary Statistics for Random and Real Groups"
  N.RanGrps Av.RanGrp.Var SD.Rangrp.Var Av.RealGrp.Var  Z-value
1       990     0.6011976     0.1317229      0.5156757 -6.46002

$"Lower Confidence Intervals (one-tailed)"
     0.5%        1%      2.5%        5%       10%
0.2701002 0.3081618 0.3605966 0.3939504 0.4432335

$"Upper Confidence Intervals (one-Tailed)"
      90%       95%     97.5%       99%     99.5%
0.7727185 0.8284755 0.8969857 0.9651415 1.0331922
```

The results indicate that the variance in actual groups about leadership ratings is significantly smaller than the variance in randomly created groups (i.e., individuals agree about leadership). For interesting cases examining situations where group members do not agree see Bliese & Halverson (1998a) and Bliese and Britt (2001).

Ongoing research continues to examine the nature of RGR based agreement indices relative to ICC(1), ICC(2) and other measures of agreement such as the r_{wg} (e.g., Lüdtke & Robitzsch, 2009). This work indicates that measures of RGR agreement are strongly related to the magnitude of the ICC values.

3.2.7 Reliability: ICC(1) and ICC(2)

Reliability indices differ from agreement indices (see Bliese, 2000; LeBreton & Senter, 2008), and the multilevel package contains the ICC1 and ICC2 functions to estimate reliability. These two functions are applied to ANOVA models and are used to estimate ICC(1) and ICC(2) as described by Bartko, (1976), James (1982), and Bliese (2000).

These two functions are applied to a one-way analysis of variance model using aov. Notice the as.factor function on GRP in the command below which designates GRP (a numeric vector) as being categorical or nominal. Once specified as categorical, R creates N-1 dummy codes in the model matrix using GRP 1 as the referent. More specifically, the contrast default in as.factor is contr.treatment which uses the first factor as the referent; however, R provides numerous options for controlling dummy and effects coding – see help(contrasts) for details. In the present example, the 99 groups result in 98 dummy-coded categories (98 df).

```
> data(bhr2000)
> hrs.mod<-aov(HRS~as.factor(GRP),data=bhr2000)
> summary(hrs.mod)
                 Df  Sum Sq Mean Sq F value    Pr(>F)
```

```
as.factor(GRP)   98  3371.4    34.4  12.498 < 2.2e-16 ***
Residuals      5301 14591.4     2.8
---
Signif. codes:  0 `***' 0.001 `**' 0.01 `*' 0.05 `.' 0.1 ` ' 1
```

The `ICC1` and `ICC2` functions are then applied to the `aov` object.

```
> ICC1(hrs.mod)
[1] 0.1741008
> ICC2(hrs.mod)
[1] 0.9199889
```

The ICC(1) value is equivalent to the ICC term referred to the mixed-effects model literature (e.g., Bryk & Raudenbush, 1992; 2002) and a value of .17 indicates that 17% of the variance in individual perceptions of work hours can be "explained" by group membership. The ICC(2) is a measure of group-mean reliability and a value of .92 indicates that groups can be reliably differentiated in terms of average work hours (see Bliese, 2000).

3.2.8 Estimate multiple ICC values: `mult.icc`

The `mult.icc` function can be used to estimate multiple ICC(1) and ICC(2) values in a given data set. Code to estimate the ICC(1) and ICC(2) values for work hours, coping with religion, and three different leadership items in the `bhr2000` data set is provided below. In the function, the first element is a subset of the dataframe with the variables of interest and the second element is the grouping variable.

```
> mult.icc(bhr2000[,c("HRS","RELIG","AF06","AF07","AP12")],bhr2000$GRP)
  Variable        ICC1      ICC2
1      HRS 0.177543969 0.9217206
2    RELIG 0.009801542 0.3506163
3     AF06 0.103492912 0.8629524
4     AF07 0.087490365 0.8394800
5     AP12 0.149052933 0.9052514
```

The results suggest that individuals use of religion as a coping mechanism had the lowest ICC(1) value (less than 1% of the variance in an individual's response can be explained by group membership). The `mult.icc` function is based upon `lme` from the `nlme` package so it returns slightly different ICC(1) and ICC(2) estimates for Work Hours (0.178 and 0.922, respectively) than estimates based on the `aov` models (0.174 and 0.920). If group sizes equal, the `lme` and `aov` approach would provide virtually identical values. In general, the preferred method with unbalanced data would be to use `lme`. One other difference (not illustrated here) is that ICC(1) values estimated in OLS can be negative, but ICC(1) values based on mixed-effects models have a lower bound of zero.

3.2.9 Comparing ICC values with a two-stage bootstrap: `boot.icc`

When examining ICC values, it can often be informative to estimate a sampling distribution to determine whether ICC values differ. For instance, the ICC(1) values for Work Hours is 0.178 (mixed-effects model), but it is not clear whether the other values which are lower significantly differ from 0.178. One way to answer the question of whether ICC values differ is to estimate a measure of variability around the point estimates. The `boot.icc` is an experimental function

that performs a two-stage bootstrap. A two-stage first samples with replacement from level-2 units (the groups) followed by sampling with replacement from individuals within the level-2 units. The function is computationally intensive, but is illustrated below both with using `lme` (the default) and `aov` (an option) as the computational algorithm underlying the ICC(1) estimate:

```
> system.time(OUT.HRS.lme<-boot.icc3(bhr2000$HRS,bhr2000$GRP,1000))
   user  system elapsed
 292.87    0.53  295.86
> quantile(OUT.HRS.lme,c(0.025,.975))
     2.5%     97.5%
0.1372000 0.2211409

> system.time(OUT.HRS.aov<-boot.icc3(bhr2000$HRS,bhr2000$GRP,1000,
  aov.est=TRUE))
   user  system elapsed
 301.93    3.35  307.89
> quantile(OUT.HRS.aov,c(0.025,.975))
     2.5%     97.5%
0.1302396 0.2160199
```

Notice that the `aov` option is slightly slower and the values are slightly smaller which is not surprising given that the `aov` estimate of the ICC(1) is smaller than the `lme` estimate. The `lme` percentile-based 95% confidence interval for the ICC(1) for work hours is [0.137, 0.221] suggesting that single point estimates of ICC(1) values outside this range would significantly differ from those associated with Work Hours. In the example using `mult.icc` everything except AP12 (I am impressed by the quality of leadership in this company) has a smaller ICC(1) value than the lower confidence interval of 0.137 for work hours. A more thorough comparison would involve estimating confidence intervals for AP12 and using both sets of confidence intervals to draw inferences (Cummings & Finch, 2005). Finally note that performing a non-parametric bootstrap of nested data is controversial because it is not clear how to best sample with replacement.

3.2.10 Visualizing an ICC(1) with `graph.ran.mean`

It is often valuable to visually examine the group-level properties of data to see the form of the group-level effects. Levin (1967) observed that high ICC(1) values can be the product of one or two highly aberrant groups rather than indicating generally shared group properties among the entire sample.

One way to examine the group-level properties of the data is to contrast the observed group means with group means that are the result of randomly assigning individuals to pseudo groups. If the actual group means and the pseudo-group means are identical, there is no evidence of group effects. If one or two groups are clearly different from the pseudo-group distribution it suggests the ICC(1) value is simply caused by a few aberrant observations. If a number of groups have higher than expected means, and a number have lower than expected means, it suggests fairly well-distributed group-level properties.

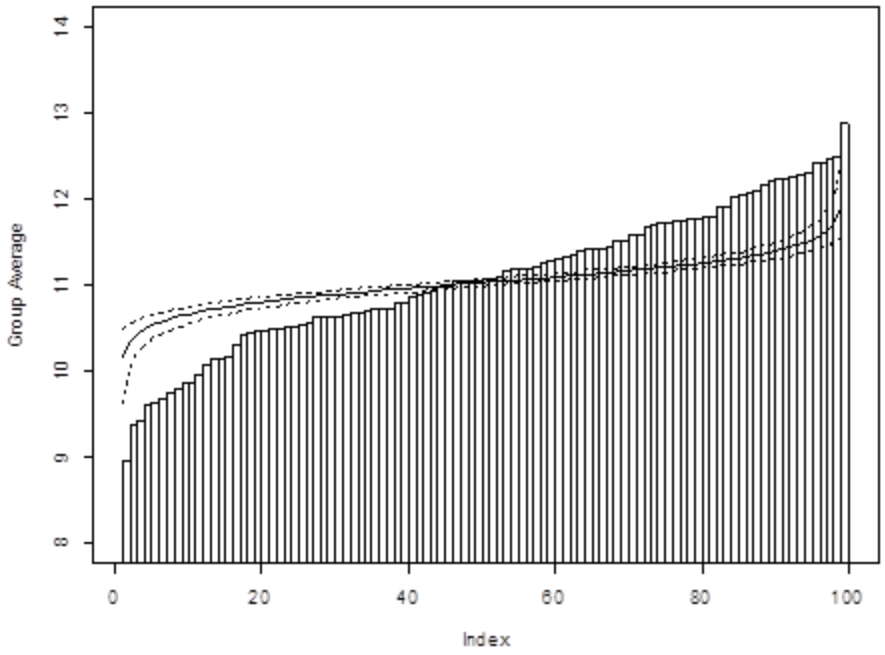
The `graph.ran.mean` function can be used to visually contrast actual group means with pseudo group means. The function requires three parameters. The first is the variable of interest. The second is the group designator, and the third is a smoothing parameter (`nreps`) determining how many sets of pseudo groups should be created to create the pseudo group curve. Low

numbers (<10) for this last parameter create a choppy line while high numbers (>25) create smooth lines. In cases where the parameter `bootci` is TRUE (see optional parameters), `nreps` should equal 1000 or more.

Three optional parameters control the y axis limits (`limits`); whether a plot is created (`graph=TRUE`) or a dataframe is returned (`graph=FALSE`); and whether bootstrap confidence intervals are estimated and plotted (`bootci=TRUE`). The default for `limits` is to use the lower 10% and upper 90% values of the raw data. The default for `graph` is to produce a plot, but returning a dataframe can be useful for exporting results for subsequent graphing in `ggplot2` or other packages. Finally, the default for `bootci` is to return a plot or a dataframe without bootstrap confidence interval estimates. In the following example, we plot the observed and pseudo group distribution of the work hours variable from the data set bhr2000.

```
> data(bhr2000)
> graph.ran.mean(bhr2000$HRS,bhr2000$GRP,nreps=1000,
limits=c(8,14),bootci=TRUE)
```

The function produces the resulting plot where the bar chart represents each groups' average rating of work hours sorted from highest to lowest, and the line represents a random distribution where 99 pseudo groups (with exact size characteristics of the actual groups) were created 1000 times and the sorted values were averaged across the 1000 iterations. The dotted lines represent the upper and lower 95% confidence interval estimates. In short, the line represents the expected distribution if there were no group-level properties associated with these data. The graph suggests fairly evenly distributed group-level properties associated with the data although two groups do stand out – one on the extreme high end and one on the extreme low end. In the end, though, the graph along with the results from the two-stage bootstrap analyses (section 3.2.11) which placed the ICC(1) estimate of 0.178 fairly close to the center of the 95% confidence interval of [0.137, 0.221] suggests that the ICC(1) values are not being driven by extreme groups (experience with other data suggests that a few extreme groups stand out in graphs and they also produce confidence intervals asymmetrical to the point estimate).



3.2.11 Simulating ICC(1) values with `sim.icc`

ICC(1) values play a key role in multilevel data; therefore, the ability to simulate ICC(1) values can be a valuable tool to help understand multilevel data and analyses. The `sim.icc` function generates data with specific ICC(1) values. Multiple vectors (items) can be generated in one of two ways: either with or without level-1 correlations. The function is used to generate a single vector (`VAR1`) below:

```
> set.seed(1535324)
> ICC.SIM<-sim.icc(gsize=10,ngrp=100,icc1=.15) #Simulate a single vector
> ICC.SIM[c(1:3,11:13),]  # Examine a few rows of simulated data
   GRP       VAR1
1    1  0.2800938
2    1 -1.4002869
3    1 -2.1422593
11   2 -1.3098119
12   2 -2.7164491
13   2 -0.3160884

> ICC1(aov(VAR1~as.factor(GRP), ICC.SIM))
[1] 0.16681
```

In the next example, four items are generated without any level-1 correlation among items. These data would represent a situation in which any observed raw correlation would be the due to the ICC(1) value. The example below uses the `waba` function discussed in section 3.4.1 to perform a variance decomposition of several raw correlations.

```
> set.seed(15324)
> ICC.SIM<-sim.icc(gsize=10,ngrp=100,icc1=.15,nitems=4)
```

```
> mult.icc(ICC.SIM[,2:5],ICC.SIM$GRP)
  Variable      ICC1      ICC2
1     VAR1 0.2035837 0.7188047
2     VAR2 0.1442111 0.6275778
3     VAR3 0.2229725 0.7415725
4     VAR4 0.1549414 0.6470794

> with(ICC.SIM,waba(VAR1,VAR2,GRP))$Cov.Theorem  #Examine CorrW
     RawCorr    EtaBx     EtaBy     CorrB    EtaWx     EtaWy       CorrW
1 0.07728039 0.530273 0.4775097 0.5939511 0.847827 0.8786265 -0.09815005

> with(ICC.SIM,waba(VAR1,VAR3,GRP))$Cov.Theorem  #Examine CorrW
    RawCorr    EtaBx     EtaBy     CorrB    EtaWx     EtaWy       CorrW
1 0.1769287 0.530273 0.5464122 0.6723887 0.847827 0.8375164 -0.02520087

> with(ICC.SIM,waba(VAR1,VAR4,GRP))$Cov.Theorem  #Examine CorrW
    RawCorr    EtaBx     EtaBy     CorrB    EtaWx     EtaWy      CorrW
1 0.1943248 0.530273 0.4874644 0.6127858 0.847827 0.8731429 0.04853107
```

Notice that the ICC(1) values for each item are variable (a function of small group sizes and a relatively small number of groups). Notice also that the `CorrW` (within-group correlation) values for three of the bivariate correlations vary around zero while `RawCorr` (the raw correlations) varies around .15 which corresponds to the simulated ICC(1) value.

As a final example, the code below incorporates a level-1 correlation of .30 among variables. Notice that the within-group correlation varies around .30 and the raw correlation increases as a function of the level-1 correlation and the ICC(1) value.

```
> set.seed(15324)
> ICC.SIM<-sim.icc(gsize=10,ngrp=100,icc1=.15,nitems=4,item.cor=.3)
> mult.icc(ICC.SIM[,2:5],ICC.SIM$GRP)
  Variable      ICC1      ICC2
1     VAR1 0.1669452 0.6671118
2     VAR2 0.1558558 0.6486689
3     VAR3 0.1381652 0.6158502
4     VAR4 0.1715351 0.6743219

> with(ICC.SIM,waba(VAR1,VAR2,GRP))$Cov.Theorem  #Examine CorrW
    RawCorr    EtaBx     EtaBy     CorrB     EtaWx     EtaWy     CorrW
1 0.3987741 0.498367 0.4883034 0.6976093 0.8669662 0.8726739 0.3026887

> with(ICC.SIM,waba(VAR1,VAR3,GRP))$Cov.Theorem  #Examine CorrW
    RawCorr    EtaBx     EtaBy     CorrB     EtaWx     EtaWy     CorrW
1 0.3746905 0.498367 0.4718088 0.7083573 0.8669662 0.8817009 0.2722794

> with(ICC.SIM,waba(VAR1,VAR4,GRP))$Cov.Theorem  #Examine CorrW
    RawCorr    EtaBx     EtaBy     CorrB     EtaWx     EtaWy     CorrW
1 0.3732463 0.498367 0.5024739 0.7104143 0.8669662 0.8645924 0.2606111
```

3.3 Regression and Contextual OLS Models

Contextual models represent a basic form a multilevel model where both the raw predictor and the group-mean of the same predictor are included in the model. For instance, regressing

Well-Being on individual work hours and group average work hours would represent a basic contextual model. A significant effect for the group-mean predictor indicates that the slope for the group-means differs from the slope for the individual-level variables and suggests a contextual effect is present (Firebaugh, 1978; Snijders & Bosker, 1999).

Prior to the introduction of multilevel mixed-effects models, OLS regression models were widely used to detect contextual effects. Firebaugh (1978) provides a good methodological discussion of these types of contextual models as does Kreft and De Leeuw (1998) and James and Williams (2000). While OLS regression has historically been used to estimate contextual regression models, the models can severely underestimate the standard error associated with the group-level effect producing tests that are too liberal. For this reason, mixed-effects models are the more appropriate way to identify contextual effects.

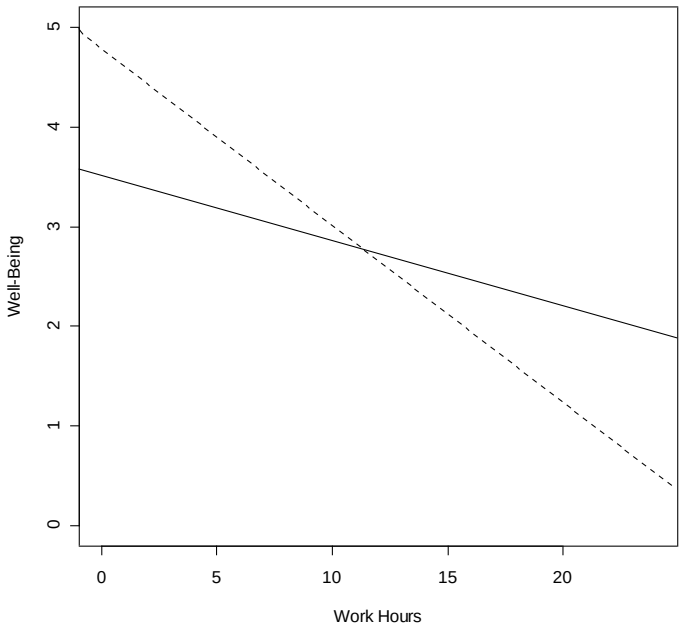
3.3.1 Contextual Effect Example

In this example, we use the `bh1996` dataframe to illustrate a contextual model involving work hours, group work hours and well-being presented in Bliese (2002). The `bh1996` dataframe has group mean variables included along with the group-mean center or demeaned variables.

```
> data(bh1996)
> names(bh1996)
 [1] "GRP"      "COHES"    "G.COHES"  "W.COHES"  "LEAD"     "G.LEAD"
 [7] "W.LEAD"   "HRS"      "G.HRS"    "W.HRS"    "WBEING"   "G.WBEING"
[13] "W.WBEING"

> tmod<-lm(WBEING~HRS+G.HRS,data=bh1996)
> round(summary(tmod)$coef,4)
            Estimate Std. Error  t value Pr(>|t|)
(Intercept)   4.7831     0.1364  35.0680        0
HRS          -0.0465     0.0049  -9.4307        0
G.HRS        -0.1308     0.0130 -10.0596        0
```

Notice that `G.HRS` is significant with a t-value of –10.06 suggesting a significant contextual effect. Later we show that this t-value is highly inflated by a standard error that is too small. Nonetheless, it is informative to plot the form of the relationship showing that the group-mean slope (the dotted line) is considerably steeper than the individual slope (the solid line). Notice the use of `!duplicated(bh1996$GRP)` to select only the first row with a specific group's group-level data effectively reducing the sample size to 99 group means:

```
> plot(bh1996$HRS,bh1996$WBEING,xlab="Work Hours",
  ylab="Well-Being",type="n") #type = n omits the 7,382 points

> abline(lm(WBEING~HRS,data=bh1996)) # plots the individual-level slope
> abline(lm(G.WBEING~G.HRS,data=bh1996[!duplicated(bh1996$GRP),]),
  lty=2) #group-level slope
```



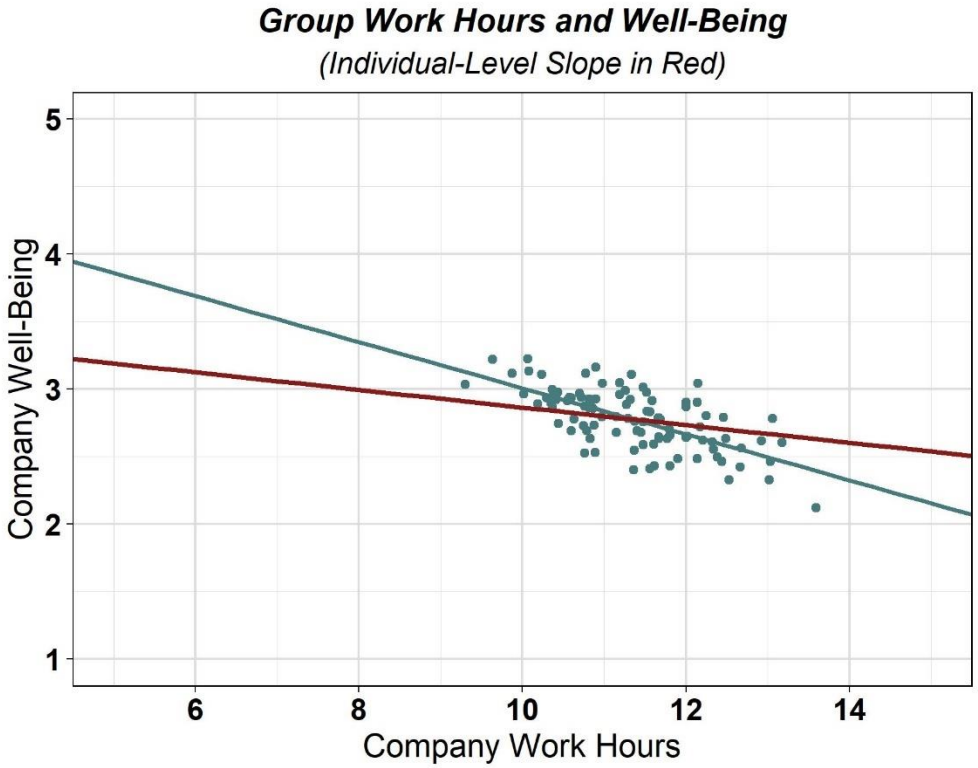
The idea that relationship strength might differ across levels is fundamental to multilevel analyses, so the basic idea of contextual regression is important. Fortunately, the problem with using OLS regression and having a standard error that is too small can be fixed in mixed-effect models (illustrated in section 4). For more details on the effects of non-independence see Bliese (2002); Bliese and Hanges (2004); Kenny and Judd, (1986) and Snijders and Bosker, (1999).

3.3.2 Contextual Effect Plot Using `ggplot2`

As an example of some of R's graphics capabilities, I reproduce the contextual effect using `ggplot2`.

```
library(ggplot2)

win.graph(height=4.75,width=6) #quartz() for MAC

data(bh1996)

bh1996.grp<-bh1996[!duplicated(bh1996$GRP),
      c("G.COHES","G.LEAD","G.HRS","G.WBEING")]

g <- ggplot(bh1996.grp, aes(x=G.HRS, y=G.WBEING))+
  labs(title = "Group Work Hours and Well-Being",
       subtitle = "(Individual-Level Slope in Red)",
       x = "Company Work Hours",
       y = "Company Well-Being")

g+coord_cartesian(xlim = c(5, 15),ylim=c(1,5))+
  geom_point(color="#477b7d")+
  geom_smooth(method="lm",fullrange=TRUE,
              se=FALSE,color="#477b7d")+
```

```
  geom_smooth(data=bh1996,aes(x=HRS,y=WBEING),
              method="lm",color="firebrick4")+
  scale_x_continuous(breaks=seq(0,24,by=2))+
  theme(
    plot.title = element_text(color="black", size=14,
                      hjust=0.5,face="bold.italic"),
    plot.subtitle = element_text(color="black", size=13,
                              hjust=0.5,face="italic"),
    axis.title.x = element_text(color="black", size=14),
    axis.title.y = element_text(color="black", size=14),
    axis.text = element_text(color="black",size=13,face="bold"),
    panel.border = element_rect(fill = NA, colour = "black",
                                size = rel(1)),
    panel.background = element_rect(fill = "transparent",
                                    colour = NA),
    panel.grid = element_line(colour = "grey87"),
    panel.grid.major = element_line(size = rel(1)),
    panel.grid.minor = element_line(size = rel(0.25)),
    axis.ticks = element_line(colour = "black",
                              size = rel(0.5))
  )
ggsave(filename = "c:\\temp\\plotgg.jpg",
       device = "jpeg")
```



3.4 Correlation Decomposition and the Covariance Theorem

OLS contextual models provide a way to determine whether regression slopes based on group means differ from regression slopes based on individual-level variables (while the OLS

contextual model for the group-mean predictor is biased by being too liberal, a null effect from the group-mean is informative). The covariance theorem provides a contextual model analog for correlations by breaking down a raw correlation into two separate components – the portion of the raw correlation attributable to within-group (individual) processes, and the portion of the correlation attributable to between-group (group-level) processes.

Robinson (1950) proposed the covariance theorem, and Dansereau and colleagues incorporated the theorem it into an analysis system they labeled WABA for Within-And-Between-Analyses (Dansereau, Alutto & Yammarino, 1984). WABA has two components: WABA I and WABA II. The first component (WABA I) uses decision tools based on eta values to inform decisions about the individual or group-level nature of the data. Eta values, however, are highly influenced by group size and unfortunately WABA I makes no group size adjustments; consequently, there is little value in using WABA I criteria unless one is working with dyads (see Bliese, 2000; Bliese & Halverson, 1998b).

Arguably a more useful way to draw inferences from eta-values is to contrast eta-values from actual groups to eta-values from pseudo groups. I illustrate this in a Random Group Resampling extension of the covariance theorem decomposition (see section 3.4.2). We begin, however, with a simple WABA analysis.

3.4.1 The `waba` and `cordif` functions

WABA II revolves around estimating the covariance theorem components, and the `waba` function in the multilevel library provides these components. The example partitions the raw correlation between work hours and well-being using the same data as used in the OLS contextual model example (section 3.3.1). The within-group correlation (`CorrW`) is group-mean centered (or demeaned) X and Y values. The group-level correlation (`CorrB`) represents the correlation between group means weighted by the size of each group.

```
> waba(bh1996$HRS,bh1996$WBEING,bh1996$GRP)
$Cov.Theorem
     RawCorr     EtaBX     EtaBY      CorrB     EtaWX     EtaWY      CorrW
1 -0.1632064 0.3787881 0.2359287 -0.7121729 0.9254834 0.9717704 -0.1107031
$n.obs
[1] 7382
$n.grps
[1] 99
```

The `waba` function returns a list with three elements. The first is the covariance theorem with all its components. The second is the number of observations, and the third is the number of groups. The latter two elements should routinely be examined because the `waba` function, by default, performs listwise deletion of missing values.

The raw correlation = (EtaBX*EtaBY*CorrB) + (EtaWX*EtaWY*CorrW) or

```
> (.379*.236*-.712)+(.925*.972*-.111)
[1] -0.1634842
```

The first set of parentheses represents the between-group component of the correlation, and the second set of parentheses represents the within-group component.

The weighted group-mean correlation of -.71 appears significantly larger than the within-group correlation of -.11. Since these two correlations are independent, we can contrast them using the `cordif` function. This function performs an r to z' transformation of the two correlations (see also the `rtoz` function) and then tests for differences between the two z' values using the formula provided in Cohen and Cohen (1983, p. 54). Four arguments are provided to `cordif`: (1) the first correlation of interest, (2) the second correlation of interest, (3) the N on which the first correlation is based, and (4) the N on which the second correlation is based. In our example, we already have the two correlations of interest (-.11 and -.71) and the relevant N values for the number of groups (99). The N for the within-group correlation is calculated as the total N minus the number of groups (see Dansereau, et al., 1984) which is 7,382 minus 99 or 7,283.

```
> cordif(rvalue1=-.1107, rvalue2=-.7122, n1=7283, n2=99)
$"z value"
[1] 7.597172
```

The z-value is larger than 1.96, so we conclude that the two correlations are significantly different for each other. This finding mirrors what we found in our contextual analysis but with an appropriate z-value.

3.4.2 Random Group Resampling of Covariance Theorem (`rgr.waba`)

As noted above, it may be interesting to see how the eta-between, eta-within, between group and within-group correlations vary as a function of the group-level properties of the data. The `rgr.waba` function provides a way to examine the group-level properties of elements of the covariance theorem. Essentially, the `rgr.waba` function allows one to answer questions such as "is the eta-between value for x larger than would be expected by chance?". The `rgr.waba` function randomly assigns individuals into pseudo groups having the exact size characteristics as the actual groups, and then calculates the covariance theorem parameters. By repeatedly assigning individuals to pseudo groups and re-estimating the covariance theorem components, one can create sampling distributions of the covariance theorem components to see if actual group results differ from pseudo group results (see Bliese & Halverson, 2002). Below I illustrate the use of `rgr.waba`. Note that this is a very computationally intensive routine, so it may take some time to complete.

```
> with(bh1996, waba(HRS,WBEING,GRP))$Cov.Theorem
     RawCorr     EtaBx     EtaBy      CorrB     EtaWx     EtaWy      CorrW
1 -0.1632064 0.3787881 0.2359287 -0.7121729 0.9254834 0.9717704 -0.1107031
> RGR.WABA<-rgr.waba(bh1996$HRS,bh1996$WBEING,bh1996$GRP,1000)
> round(summary(RGR.WABA),dig=4)
       RawCorr     EtaBx     EtaBy     CorrB     EtaWx     EtaWy     CorrW
NRep 1000.0000 1000.0000 1000.0000 1000.0000 1000.0000 1000.0000 1000.0000
Mean   -0.1632    0.1154    0.1151   -0.1614    0.9933    0.9933   -0.1632
SD      0.0000    0.0082    0.0081    0.0961    0.0010    0.0009    0.0013
```

The `summary` of the `rgr.waba` object produces a table giving the number of random repetitions, the means and the standard deviations from analysis. Notice that when there are no meaningful group differences, the between-group correlation, the raw correlation, and the within-group correlation all have the same value (with some rounding error). The raw correlation has a

standard deviation of zero because it does not change. In contrast, the between-group correlation has the highest standard deviation (.096) indicating that it varied the most across the pseudo group runs. All of covariance theorem components in the actual groups significantly vary from their counterparts in the pseudo group analysis because most actual group values are more than two standard deviations different from the pseudo group means.

To further test for significant differences, we can examine the sampling distribution of the random runs, and use the 2.5% and 97.5% sorted values to approximate 95% confidence intervals. Any values outside of this range would be considered significantly different from their pseudo group counterparts.

```
> quantile(RGR.WABA,c(.025,.975))
           EtaBx      EtaBy       CorrB     EtaWx     EtaWy      CorrW
2.5%  0.09944367 0.09916248 -0.34021577 0.9913014 0.9914049 -0.1658118
97.5% 0.13161137 0.13082964  0.03106165 0.9950432 0.9950713 -0.1607501
```

All of the covariance theorem values based on the actual groups are outside of the 95% confidence interval estimates. In other words, all the actual group results are significantly different than would be expected if individuals had been randomly assigned to groups ($p<.05$). The 99% confidence intervals draw the same conclusion at a more stringent confidence level.

```
> quantile(RGR.WABA,c(.005,.995))
           EtaBx      EtaBy      CorrB     EtaWx     EtaWy      CorrW
0.5%  0.09307571 0.09416619 -0.4065661 0.9907133 0.9908819 -0.1666120
99.5% 0.13596781 0.13473339  0.1049678 0.9956590 0.9955565 -0.1596676
```

Keep in mind that a replication is likely to differ slightly from results presented here because we did not start by setting a random seed.

3.5 Simulate Multilevel Correlations (`sim.mlcor`)

Contextual effects where relationships significantly differ across levels such as the illustration involving work hours and well-being are common. In many cases, the effects are less dramatic than having a within-group correlation of -.11 and a between-group correlation of -.71, but contextual effects exist and what drives them is relatively unexplored. One necessary, but not sufficient, condition for observing contextual effects is that both variables must have non-zero ICC(1) values (see Bliese, 1998). For this reason, researchers who are focused on modeling shared properties of constructs such as safety climate, cohesion, or team emotional cultures need to develop measures that have good ICC1 values and differentiate groups (see Bliese, Maltarich, Hendricks, Hofmann & Adler, 2019).

The `sim.mlcor` (simulate multilevel correlation) function was designed to help explore how measurement properties at different levels impact observed raw, within, and between-group correlations. We could examine, for example, how correlations would have differed if we had been able to increase the ICC(1) values or alpha values of the variables.

In the function, users provide group size, the number of groups, a between-group correlation, a within-group correlation, an ICC(1) for x, an ICC(1) for y, and alpha values for both x and y. The function returns a simulated dataset.

We can create a simulated dataset for our running example involving work hours and well-being by first obtaining the values from the actual data:

```
> data(bh1996)
> with(bh1996,waba(HRS,WBEING,GRP))
$Cov.Theorem
     RawCorr     EtaBx     EtaBy      CorrB     EtaWx     EtaWy      CorrW
1 -0.1632064 0.3787881 0.2359287 -0.7121729 0.9254834 0.9717704 -0.1107031

$n.obs
[1] 7382

$n.grps
[1] 99

> mult.icc(bh1996[,c("HRS","WBEING")],bh1996$GRP)
  Variable       ICC1      ICC2
1      HRS 0.12923699 0.9171286
2   WBEING 0.04337922 0.7717561
```

In this case, the group-level correlation of -.71 is smaller than it would have been if group means had reliabilities of 1. Instead, the ICC(2) values show that the group-mean reliability for work hours is .92 and for well-being the value is .77. We can correct the -.71 value by adjusting the incremental effect (the difference between the within-group and between-group correlation) for attenuation using ICC(2) values and adding this effect back to the within-group correlation.

```
> (-0.7121729--0.1107031)/sqrt(0.9171286*0.7717561)+-0.1107031
[1] -0.8256251
```

From this correction we can assume that if the ICC(2) values for both variables had been 1, the group-mean correlation would have been -.826. Using these data in the simulation and assuming an average group sizes of 75 (7382/99) and alpha values of 1, we obtain the following simulated dataset with results that mirror our actual data. Here I set a seed so exact results can be replicated.

```
> set.seed(578323)
> SIM.ML.COR<-sim.mlcor(gsize=75,ngrp=99,gcor=-.8256,wcor=-.1107,
+                       icc1x=0.04338,icc1y=0.12924,alphax=1,alphay=1)

> with(SIM.ML.COR,waba(X,Y,GRP))
$Cov.Theorem
     RawCorr     EtaBx     EtaBy      CorrB     EtaWx     EtaWy      CorrW
1 -0.1699012 0.2317119 0.3804799 -0.7353652 0.9727844 0.9247892 -0.1167938

$n.obs
[1] 7425

$n.grps
[1] 99

> mult.icc(SIM.ML.COR[,c("X","Y")],SIM.ML.COR$GRP)
  Variable       ICC1      ICC2
```

```
1        X 0.04142764 0.7642263
2        Y 0.13448630 0.9209720
```

To see the implications of having had a zero ICC(1) for the one of the variables, we can rerun the simulation and show that the between-group correlation no longer differs from the within or raw. This result is entirely expected because a necessary condition for contextual effects is a non-zero ICC(1) on both variables.

```
> SIM.ML.COR<-sim.mlcor(gsize=75,ngrp=99,gcor=-.8256,wcor=-.1107,
+                       icc1x=0,icc1y=0.12924,alphax=1,alphay=1)

> with(SIM.ML.COR,waba(X,Y,GRP))$Cov.Theorem
     RawCorr     EtaBx     EtaBy      CorrB     EtaWx     EtaWy      CorrW
1 -0.1304409 0.1256461 0.3688716 -0.1483209 0.9920751 0.9294803 -0.1340036

> mult.icc(SIM.ML.COR[,c("X","Y")],SIM.ML.COR$GRP)
  Variable        ICC1      ICC2
1        X 0.002640832 0.1656842
2        Y 0.125605587 0.9150646
```

To see the implications of improved the group-level measurement properties of the well-being measure to better differentiate groups, we can increase the ICC(1) for X to be .10 which produces a between-group correlation of -.76 in this particular run. The raw correlation also inherits more from the group correlation and increases to -.19.

```
> SIM.ML.COR<-sim.mlcor(gsize=75,ngrp=99,gcor=-.8256,wcor=-.1107,
+             icc1x=.10,icc1y=0.12924,alphax=1,alphay=1)

> with(SIM.ML.COR,waba(X,Y,GRP))$Cov.Theorem
     RawCorr     EtaBx     EtaBy      CorrB     EtaWx     EtaWy      CorrW
1 -0.1947374 0.3607571 0.3796512 -0.7638527 0.9326598 0.9251297 -0.1044453

> mult.icc(SIM.ML.COR[,c("X","Y")],SIM.ML.COR$GRP)
  Variable      ICC1      ICC2
1        X 0.1195602 0.9105922
2        Y 0.1338431 0.9205681
```

Finally, to illustrate one of Bliese et al.'s (2019) main points that individual reliability indices such as alpha are largely irrelevant to the magnitude of between-group correlations, we can change the alpha for both X and Y to be .70. In this case, note that the within-correlation is now -.08 and would be adjusted back to -.11 if corrected for attenuation (`-.08/sqrt(.7*.7)`)

```
> SIM.ML.COR<-sim.mlcor(gsize=75,ngrp=99,gcor=-.8256,wcor=-.1107,
+             icc1x=0.04338,icc1y=0.12924,alphax=.7,alphay=.7)
> with(SIM.ML.COR,waba(X,Y,GRP))$Cov.Theorem
     RawCorr     EtaBx     EtaBy      CorrB     EtaWx     EtaWy       CorrW
1 -0.1356601 0.2287517 0.3741004 -0.6967766 0.9734848 0.9273882 -0.08421891
> mult.icc(SIM.ML.COR[,c("X","Y")],SIM.ML.COR$GRP)
  Variable       ICC1      ICC2
1        X 0.04003354 0.7577361
2        Y 0.12957194 0.9177936
```

For detailed examinations of measurement properties, the examples presented would need to be put within a Monte Carlo function and averaged across multiple iterations, but the `sim.mlcor` function provides a way to generate multilevel correlations.

4 Mixed-Effects Models for Multilevel Data

This section illustrates the use of mixed-effects models to analyze multilevel data using the `nlme` package (Pinhiero & Bates, 2000). Most of the examples described in this section are taken from Bliese (2002) and use the Bliese and Halverson (1996) data (`bh1996`). Model notation is based on Bryk and Raudenbush's (1992) and Raudenbush and Bryk (2002).

A complete description of mixed-effects modeling is beyond the scope of this document; nonetheless, a short overview is presented to help facilitate the illustration of the methods. For more detailed discussions see Bliese, (2002); Bliese, Maltarich and Hendricks, 2018; Bryk and Raudenbush, (1992); Hofmann, (1997); Hox (2002); Kreft and De Leeuw, (1998); Pinheiro and Bates (2000); Raudenbush and Bryk (2002) and Snidjers and Bosker (1999).

One can think of mixed-effects models as ordinary regression models that have additional variance terms for handling non-independence due to group membership. The key to mixed-effects models is to understand how nesting individuals within groups can produce additional sources of variance (non-independence) in data.

The first variance term that distinguishes a mixed-effects model from a regression model is a term that reflects the degree to which groups differ in their mean values (intercepts) on the dependent variable (DV). A significant variance term (τ_{00}) indicates that groups significantly differ in terms of the DV and further suggests that it may be useful to include group-level variables as predictors. Group-level variables (or level-2 variables) differ across groups but are consistent for members within the same groups. For example, group average work hours are the same across all members of the same group and represents a level-2 variable that could potentially be used to predict group-level variance (τ_{00}) in well-being.

The second variance term that distinguishes a mixed-effects model from typical regression reflects the degree to which slopes between independent and dependent variables vary across groups (τ_{11}). Single-level regression models generally assume that the relationship between the IV and DV is constant across groups. In contrast, mixed-effects models permit testing whether the slope varies among groups. If slopes significantly vary, we can explain the variation by including a cross-level interaction using a level-2 variable such as average group work hours to explain why the slope between IV and DV in some groups is stronger than the slopes in other groups.

A third variance term is common to both mixed-effects models and regression models. This variance term, σ^2, reflects the degree to which an individual score differs from its predicted value within a specific group. σ^2 represents the within-group variance and is predicted individual-level or level-1 variables. Level-1 variables differ among members of the same group. For instance, a level-1 variable such as participant age would vary among members of the same group.

In summary, in a complete mixed-effect model analysis, one examines (1) level-1 factors related to the within-group variance σ^2; (2) group-level factors related to the between-group variation in intercepts τ_{00}; and (3) group-level factors related to within-group slope differences,

τ_{11}. The next sections re-analyze portions of the Bliese and Halverson (1996) data set to illustrate a typical sequence of steps used in multilevel modeling.

4.1 Steps in multilevel modeling

4.1.1 Step 1: Examine the ICC for the Outcome

Because multilevel modeling involves predicting variance at different levels, it is important to begin by determining the levels where significant variation exists. In the case of the two-level model (the only models considered here) we can assume there is significant variation in the within-group variance, σ^2. We do not necessarily assume there will be significant intercept variation (τ_{00}) or between-group slope variation (τ_{11}) so modeling often begins with variance decomposition of intercept variance (see Bryk & Raudenbush, 1992; Hofmann, 1997). If τ_{00} does not differ by more than chance levels, there may be little reason to use mixed-effects models as simpler OLS models will suffice (though see Bliese et al., 2018 who argue that there is virtually no downside to estimating mixed-effect models even when If τ_{00} is small or non-significant because in these cases the mixed-effect models just return the OLS estimates). Note that if slopes randomly vary (τ_{11}) even if intercepts (τ_{00}) do not, there may still be reason to estimate mixed-effects models (see Snijders & Bosker, 1999).

In Step 1, we first examine the group-level properties of the outcome variable to estimate the ICC(1) (commonly referred to simply as the ICC in mixed-effect models). Second, we determine whether the variance of the intercept (τ_{00}) is significantly larger than zero.

These two aspects of the outcome variable are examined by estimating an unconditional means or null model. An unconditional means model does not contain any predictors but includes a random intercept variance term for groups. The model estimates how much variability there is in mean Y values (i.e., how much variability there is in the intercept) relative to the total variability. In the two stage HLM notation, the model is:

$$Y_{ij} = \beta_{0j} + r_{ij}$$
$$\beta_{0j} = \gamma_{00} + u_{0j}$$

In combined form, the model is: $Y_{ij} = \gamma_{00} + u_{0j} + r_{ij}$. The null model states that the dependent variable is a function of a common intercept γ_{00}, and two error terms: the between-group error term, u_{0j}, and the within-group error term, r_{ij}. The model essentially states that any Y value can be described in terms of an overall mean plus some error associated with group membership and some individual error. A summary of the variance components of the null model provides two estimates of variance; τ_{00} associated with u_{0j} reflecting the variance in how much each groups' intercept varies from the overall intercept (γ_{00}), and σ^2 associated with r_{ij} reflecting how much each individual's score differs from the group mean. Bryk and Raudenbush (1992) note that the null model is directly equivalent to a one-way random effects ANOVA – an ANOVA model where one predicts the dependent variable as a function of group membership.

We estimate the unconditional means model and other mixed-effects models using the `lme` (for linear mixed effects) function in the `nlme` package (see Pinheiro & Bates, 2000). There are two formulas that must be specified in any `lme` call: a fixed effects formula and a random effects formula.

In the unconditional means model, the fixed portion of the model is γ_{00} (an intercept term) and the random component is $u_{0j}+r_{ij}$. The random portion of the model states that intercepts can vary among groups. We begin the analysis by attaching the `multilevel` package (which also loads the `nlme` package) and making the `bh1996` data set in the `multilevel` package available for analysis.

```
> library(multilevel)
> data(bh1996)
> Null.Model<-lme(WBEING~1,random=~1|GRP,data=bh1996,
  control=list(opt="optim"))
```

In the model, the fixed formula is `WBEING~1` indicating that the only predictor of well-being is an intercept term. The model assumes that in the absence of any predictors, the best estimate of any specific outcome value is the mean value on the outcome. The random formula is `random=~1|GRP` which specifies that the intercept can vary as a function of group membership. A random intercept model is the most basic random formula, and in many situations a random intercept model may be all that is required to adequately account for the nested nature of the grouped data. The option `control=list(opt="optim")` in the call to `lme` instructs the program to use R's general purpose optimization routine. Versions of `lme` after 2.2 default to `nlmimb` which has several advantages including better diagnostics when optimization fails. In practice, however, `nlmimb` tends to converge less often than the general purpose optimizater. Furthermore, the examples in this document were estimated under `"optim"`, so for consistency we will revert back to the original optimizer. In practice, users likely want to use the default `"nlmimb"` optimizer; however, if models fail to converge it may be useful to revert to `"optim"`.

Estimating ICC. The unconditional means model provides between-group and within-group variance estimates in the form of τ_{00} and σ^2, respectively. The formula for the ICC is $\tau_{00}/(\tau_{00} + \sigma^2)$ (see, Bryk & Raudenbush, 1992; Kreft & De Leeuw, 1998). Bliese (2000) notes that the ICC is equivalent to Bartko's ICC(1) formula (Bartko, 1976) and to Shrout and Fleiss's ICC(1,1) formula (Shrout & Fleiss, 1979). The `VarCorr` function provides estimates of variance for an `lme` object.

```
> VarCorr(Null.Model)
GRP = pdSymm(1)
            Variance   StdDev
(Intercept) 0.03580079 0.1892110
Residual    0.78949727 0.8885366
> 0.03580079/(0.03580079+0.78949727) #Calculate ICC
[1] 0.04337922
```

The estimate of τ_{00} (between-group or Intercept variance) is 0.036, and the estimate of σ^2 (within-group or residual variancel) is 0.789. The ICC estimate ($\tau_{00}/(\tau_{00} + \sigma^2)$) is .04.

To verify that the ICC results from the mixed-effects models are similar to those from an ANOVA model and the `ICC1` function (see section 0) we can perform an ANOVA analysis on the same data.

```
> tmod<-aov(WBEING~as.factor(GRP),data=bh1996)
> ICC1(tmod)
```

```
[1] 0.04336905
```

The ICC value from the mixed-effects model and the ICC(1) from the ANOVA model are similar although they will tend to differ if group sizes vary dramatically given that the ANOVA models assume equal group sizes.

Determining whether τ_{00} is significant. Returning to our original analysis involving well-being from the `bh1996` data set, we would likely be interested in knowing whether the intercept variance (i.e.,τ_{00}) estimate of 0.036 is significantly different from zero. In mixed-effects models, we perform this test by comparing –2 log likelihood values between (1) a model with a random intercept, and (2) a model without a random intercept.

A model without a random intercept can be estimated using the `gls` function in the `nlme` package. The –2 log likelihood values (i.e., Deviance) for an `lme` or `gls` object are obtained using the `logLik` function and multiplying the returned value by –2. If the –2 log likelihood value for the model with the random intercept is significantly smaller than the model without the random intercept (based on a Chi-square distribution), then we conclude that the model with the random intercept fits the data significantly "better" than does the model without the random intercept. In the R, model contrasts are conducted using the `anova` function.

```
> Null.gls<-gls(WBEING~1,data=bh1996,
  control=list(opt="optim"))

> logLik(Null.gls)*-2
`log Lik.' 19536.17 (df=2)

> logLik(Null.Model)*-2
`log Lik.' 19347.34 (df=3)

> 19536.17-19347.34
[1] 188.83

> anova(Null.gls, Null.Model)
           Model df      AIC      BIC    logLik   Test  L.Ratio p-value
Null.gls       1  2 19540.17 19553.98 -9768.084
Null.Model     2  3 19353.34 19374.06 -9673.669 1 vs 2 188.8303  <.0001
```

The –2 log likelihood value for the `gls` model without the random intercept is 19536.17. The difference of 188.8 is significant on a Chi-Squared distribution with one degree of freedom (one model estimated a variance term associated with a random intercept, the other did not, and this results in the one df difference). These results indicate significant intercept variation.

In summary, we would conclude that there is significant intercept variation in terms of general well-being scores across the 99 Army companies in our sample. We also estimate that 4% of the variation in individuals' well-being score is a function of the group to which he or she belongs. Thus, a model that allows for random variation in well-being among Army companies is a better fit than a model that does not allow for this random variation.

4.1.2 Step 2: Explain Level 1 and 2 Intercept Variance

At this point, we have two sources of variation that we can attempt to explain in subsequent modeling – within-group variation (σ^2) and between-group intercept (i.e., mean) variation (τ_{00}).

In many cases, these may be the only two sources of variation we are interested in explaining so let us begin by building a model that predicts these two sources of variation.

In our running example, we assume that individual well-being is negatively related to individual reports of work hours. At the same time, however, we assume that average work hours in an Army Company are related to the average well-being of the Company over-and-above the individual-level work hours and well-being relationship. Using Hofmann and Gavin's (1998) terminology, we are testing an incremental model where the level-2 variable predicts unique variance after controlling for level-1 variables. Our model is directly equivalent to the contextual model that we estimated in section 3.3.1 but we now use mixed-effect models rather than OLS regression.

The form of the model using Bryk and Raudenbush's (1992) notation is:

$$WBEING_{ij} = \beta_{0j} + \beta_{1j}(HRS_{ij}) + r_{ij}$$
$$\beta_{0j} = \gamma_{00} + \gamma_{01}(G.HRS_j) + u_{0j}$$
$$\beta_{1j} = \gamma_{10}$$

The first line indicates that individual well-being is a function of the groups' intercept plus a component that reflects the linear effect of individual reports of work hours plus some random error. The second line indicates that each groups' intercept (mean) is a function of some common intercept (γ_{00}) plus a component that reflects the linear effect of average group work hours plus some random between-group error. The third line states that the slope between individual work hours and well-being is fixed—it is not allowed to randomly vary across groups. Stated another way, we assume that the relationship between work hours and well-being varies by no more than chance levels among groups.

When we combine the three rows into a single equation, we get an equation that looks like a common regression equation with an extra error term (u_{0j}). This error term indicates that WBEING intercepts (i.e., means) can randomly differ across groups. The combined model is:

$$WBEING_{ij} = \gamma_{00} + \gamma_{10}(HRS_{ij}) + \gamma_{01}(G.HRS_j) + u_{0j} + r_{ij}$$

This model is specified in `lme` as:

```
> Model.1<-lme(WBEING~HRS+G.HRS,random=~1|GRP,data=bh1996,
  control=list(opt="optim"))

> summary(Model.1)
Linear mixed-effects model fit by REML
 Data: bh1996
       AIC      BIC    logLik
  19222.28 19256.81 -9606.14

Random effects:
 Formula: ~1 | GRP
        (Intercept)  Residual
StdDev:   0.1163900 0.8832353

Fixed effects: WBEING ~ HRS + G.HRS
                Value  Std.Error   DF   t-value p-value
(Intercept)  4.740829 0.21368746 7282 22.185808  <.0001
HRS         -0.046461 0.00488798 7282 -9.505056  <.0001
```

```
G.HRS       -0.126926 0.01940357   97 -6.541368  <.0001
 Correlation:
      (Intr) HRS
HRS    0.000
G.HRS -0.965 -0.252

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-3.35320562 -0.65024982  0.03760797  0.71319835  2.70917777

Number of Observations: 7382
Number of Groups: 99
```

Notice that work hours are significantly negatively related to individual well-being. Furthermore, after controlling the individual-level relationship, average work hours (G.HRS) are related to the average well-being in a group. The interpretation of this model, like the interpretation of the contextual effect model (section 3.3.1) indicates that the slope at the group-level significantly differs from the slope at the individual level. Indeed, in this example, each hour increase at the group level is associated with a -.163 (-.046+-.127) decrease in average well-being. The coefficient of -.127 reflects the degree of difference between the two slopes. Importantly, in the mixed-effect model, the t-value for G.HRS is -6.54 whereas in the OLS model the t-value was upwardly biased at -10.06.

In this basic model, we can also estimate how much of the variance was explained by these two predictors. Because individual work hours were significantly related to well-being, we expect that it will have "explained" some of the within-group variance σ^2. Similarly, since average work hours were related to the group well-being intercept we expect that it will have "explained" some of intercept variance, τ_{00}. Recall that in the null model, the variance estimate for the within-group residuals, σ^2, was 0.789; and the variance estimate for the intercept, τ_{00}, was 0.036. The `VarCorr` function on the `Model.1` object reveals that each variance component has changed slightly.

```
> VarCorr(Model.1)
GRP = pdSymm(1)
            Variance   StdDev
(Intercept) 0.01354663 0.1163900
Residual    0.78010466 0.8832353
```

Specifically, the variance estimates from the model with the two predictors are 0.780 and 0.014. That is, the variance of the within-group residuals decreased from 0.789 to 0.780 and the variance of the between-group intercepts decreased from 0.036 to 0.014. We can calculate the percent of variance explained by using the following formula:

Variance Explained = 1 – (Var with Predictor/Var without Predictor)

To follow through with our example, work hours explained 1 – (0.780/0.789) or 0.011 (1%) of the within-group variance in σ^2, and group-mean work hours explained 1 – (0.014/0.036) or 0.611 (61%) of the between-group intercept variance τ_{00}. While the logic behind variance estimates appears straightforward (at least in models without random slopes), the variance estimates should be treated with some degree of caution because they are partially dependent

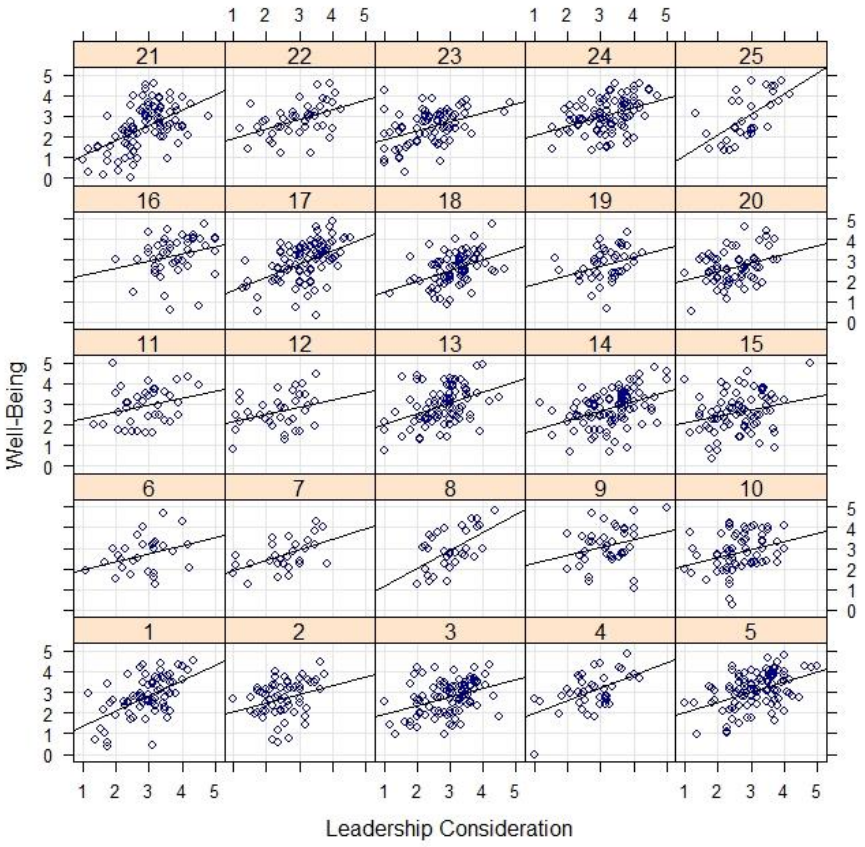
upon how one specifies the models. Interested readers are directed to Snijders and Bosker (1994; 1999) for an in-depth discussion of variance estimates.

4.1.3 Step 3: Examine and Predict Slope Variance

Let us continue our analysis by trying to explain the third source of variation, namely, variation in our slopes (τ_{11}, τ_{12}, etc.). To do this, we examine another variable from `bh1996`. This variable represents Army Company members' ratings of leadership consideration (LEAD). Generally, individual soldiers' ratings of leadership are related to well-being. In this analysis, however, we will consider the possibility that the strength of the relationship between individual ratings of leadership consideration and well-being varies among groups.

We begin by examining slope variation among the first 25 groups using `xyplot` from the `lattice` package.

```
> library(lattice)
> xyplot(WBEING~LEAD|as.factor(GRP),data=bh1996[1:1582,],
  type=c("p","g","r"),col="dark blue",col.line="black",
  xlab="Leadership Consideration",
  ylab="Well-Being")
```



From the plot of the first 25 groups in the `bh1996` data set, it seems likely that there is some slope variation. The plot, however, does not tell us whether this variation is significant. We begin our analysis of slope variability by adding leadership consideration to our model and testing whether there is significant variation in the leadership consideration and well-being slopes across groups. Our base model is:

$$
\begin{aligned}
WBEING_{ij} = \ & \beta_{0j} + \beta_{1j}(HRS_{ij}) + \beta_{2j}(LEAD_{ij}) + r_{ij} \\
& \beta_{0j} = \gamma_{00} + \gamma_{01}(G.HRS_j) + u_{0j} \\
& \beta_{1j} = \gamma_{10} \\
& \beta_{2j} = \gamma_{20}
\end{aligned}
$$

The last two lines include that neither the slope for HRS or LEAD is allowed to vary across groups. In combined form the model is:

$WBEING_{ij} = \gamma_{00} + \gamma_{10}(HRS_{ij}) + \gamma_{20}(LEAD_{ij}) + \gamma_{01}(G.HRS_j) + u_{0j} + r_{ij}$.

The model specification in `lme` is:

```
> Model.2<-lme(WBEING~HRS+LEAD+G.HRS,random=~1|GRP, data=bh1996,
+                control=list(opt="optim"))

> round(summary(Model.2)$tTable,digit=3)
             Value Std.Error   DF t-value p-value
(Intercept)  2.559     0.216 7281  11.859       0
HRS         -0.028     0.004 7281  -6.317       0
LEAD         0.496     0.013 7281  38.786       0
G.HRS       -0.079     0.019   97  -4.185       0

> VarCorr(Model.2)
GRP = pdLogChol(1)
            Variance   StdDev
(Intercept) 0.01418026 0.1190809
Residual    0.64704412 0.8043905
```

Across the sample, individuals' perceptions of leadership have a strong, positive relationship to their well-being. To determine whether the strength of this relationship differs across groups, we need to estimate a model with a random slope for LEAD. This alternative model is:

$$
\begin{aligned}
WBEING_{ij} = \ & \beta_{0j} + \beta_{1j}(HRS_{ij}) + \beta_{2j}(LEAD_{ij}) + r_{ij} \\
& \beta_{0j} = \gamma_{00} + \gamma_{01}(G.HRS_j) + u_{0j} \\
& \beta_{1j} = \gamma_{10} \\
& \beta_{2j} = \gamma_{20} + u_{2j}
\end{aligned}
$$

The last line indicates that the slope between leadership consideration and well-being for any specific group is a function of a common slope γ_{20} and a group-specific error term u_{2j}. The variance term associated with u_{2j} is τ_{12}. In this model, we have not permitted the slope between individual work hours and individual well-being to vary across groups.

In combined form the model is:

$$WBEING_{ij} = \gamma_{00} + \gamma_{10}(HRS_{ij}) + \gamma_{20}(LEAD_{ij}) + \gamma_{01}(G.HRS_j) + u_{0j} + u_{2j} * LEAD_{ij} + r_{ij}.$$

The model specification in `lme` and the relevant changes to the variance components are:

```
> Model.2a<-lme(WBEING~HRS+LEAD+G.HRS,random=~LEAD|GRP, data=bh1996,
+               control=list(opt="optim"))

> VarCorr(Model.2a)
GRP = pdLogChol(LEAD)
            Variance   StdDev    Corr
(Intercept) 0.14401197 0.3794891 (Intr)
LEAD        0.01044352 0.1021935 -0.97
Residual    0.64129330 0.8008079
```

Changing the random component to `(random=~LEAD|GRP)` produces an estimate of the slope variance, τ_{12}, (.01) and an estimate of the correlation between the intercept and slope (-.97). To test whether this model provides significantly better fit, we test the –2 log likelihood ratios between a model with and a model without a random slope for leadership consideration and well-being.

```
> anova(Model.2,Model.2a)
         Model df      AIC      BIC    logLik   Test  L.Ratio p-value
Model.2      1  6 17862.68 17904.12 -8925.341
Model.2a     2  8 17838.58 17893.83 -8911.290 1 vs 2 28.10254  <.0001
```

This comparison test is known to be conservative and we could halve the p-value (LaHuis & Ferguson, 2009), but even so the difference of 28.10 is significant on two degrees of freedom. The -2 log likelihood results indicate the model with the random effect for the leadership consideration and well-being slope provides a significantly better fit than the model without this random effect implying that the strength of the slope differs across groups.

Another way to consider the differences between the two models is to examine the empirical Bayes' estimates for each group. The values for the first five groups with the random intercept model are:

```
> coef(Model.2)[1:5,]
  (Intercept)         HRS      LEAD       G.HRS
1    2.534036 -0.02827849 0.4956385 -0.07900961
2    2.694639 -0.02827849 0.4956385 -0.07900961
3    2.458733 -0.02827849 0.4956385 -0.07900961
4    2.764899 -0.02827849 0.4956385 -0.07900961
5    2.616261 -0.02827849 0.4956385 -0.07900961
```

In this specification, group 4 has the highest level of well-being, and group 3 has the lowest, but these intercept (mean) differences are the only model parameters varying across groups. The slopes match the values from the summary of the t-table presented previously. In contrast, the empirical Bayes' estimates for model with the random slope are:

```
> coef(Model.2a)[1:5,]
  (Intercept)         HRS      LEAD       G.HRS
1    2.195403 -0.02847764 0.5715939 -0.07050472
2    2.839074 -0.02847764 0.4071772 -0.07050472
3    2.398461 -0.02847764 0.4910177 -0.07050472
4    2.846874 -0.02847764 0.4247142 -0.07050472
5    2.608235 -0.02847764 0.4679652 -0.07050472
```

In this specification, the slope indicated the strength of the relationship between individuals' perceptions of leadership consideration and their well-being also varies by group. In group 1, the relationship between the two variables is stronger (.57) than in group 2 (.41).

Given significant variation in the leadership and well-being slope, we can attempt to see what group-level properties are related to this variation. We propose that when groups are under a lot of strain from work requirements, the relationship between leadership consideration and well-being will be relatively strong. In contrast, when groups are under little strain, we expect a relatively weak relationship between leadership consideration and well-being. Our proposition represents a contextual effect in an occupational stress model (see Bliese & Jex, 2002).

Our proposition represents a cross-level interaction where the slope between individuals' perceptions of leadership consideration and their ratings of well-being varies as a function of the level-2 variable of group work demands. In mixed-effects models, we test this hypothesis by examining whether a level-2 variable explains a significant amount of the level-1 slope variation among groups. In our example, we test whether average work hours in the group "explains" group-by-group variation in the relationship between individual perceptions of leadership consideration and individual reports of well-being. In Bryk and Raudenbush's (1992) notation, the model that we are testing is:

$$
\begin{aligned}
WBEING_{ij} &= \beta_{0j} + \beta_{1j}(HRS_{ij}) + \beta_{2j}(LEAD_{ij}) + r_{ij} \\
\beta_{0j} &= \gamma_{00} + \gamma_{01}(G.HRS_j) + u_{0j} \\
\beta_{1j} &= \gamma_{10} \\
\beta_{2j} &= \gamma_{20} + \gamma_{21}(G.HRS_j) + u_{2j}
\end{aligned}
$$

In combined form the model is:

$$WBEING_{ij} = \gamma_{00} + \gamma_{10}(HRS_{ij}) + \gamma_{20}(LEAD_{ij}) + \gamma_{01}(G.HRS_j) + \gamma_{21}(LEAD_{ij} * G.HRS_j) + u_{0j} + u_{2j} * LEAD_{ij} + r_{ij}.$$

In `lme`, we specify the cross-level interaction by adding an interaction term between leadership (LEAD) and average group work hours (G.HRS). Specifically, the model is:

```
> Final.Model<-lme(WBEING~HRS+LEAD+G.HRS+LEAD:G.HRS,
random=~LEAD|GRP,data=bh1996,control=list(opt="optim"))

> round(summary(Final.Model)$tTable,dig=3)
             Value Std.Error   DF t-value p-value
(Intercept)  3.654     0.726 7280   5.032   0.000
HRS         -0.029     0.004 7280  -6.391   0.000
LEAD         0.126     0.217 7280   0.578   0.564
G.HRS       -0.175     0.064   97  -2.751   0.007
```

```
LEAD:G.HRS   0.032     0.019 7280   1.703   0.089
```

The `tTable` results from the final model indicate there is a significant cross-level interaction (the last row using a liberal p-value of less than .10). This result indicates that average work hours "explained" a significant portion of the variation in τ_{12} – the vertical cohesion and well-being slope.

4.1.4 Step 3 using the lme4 Package and Interaction Plot

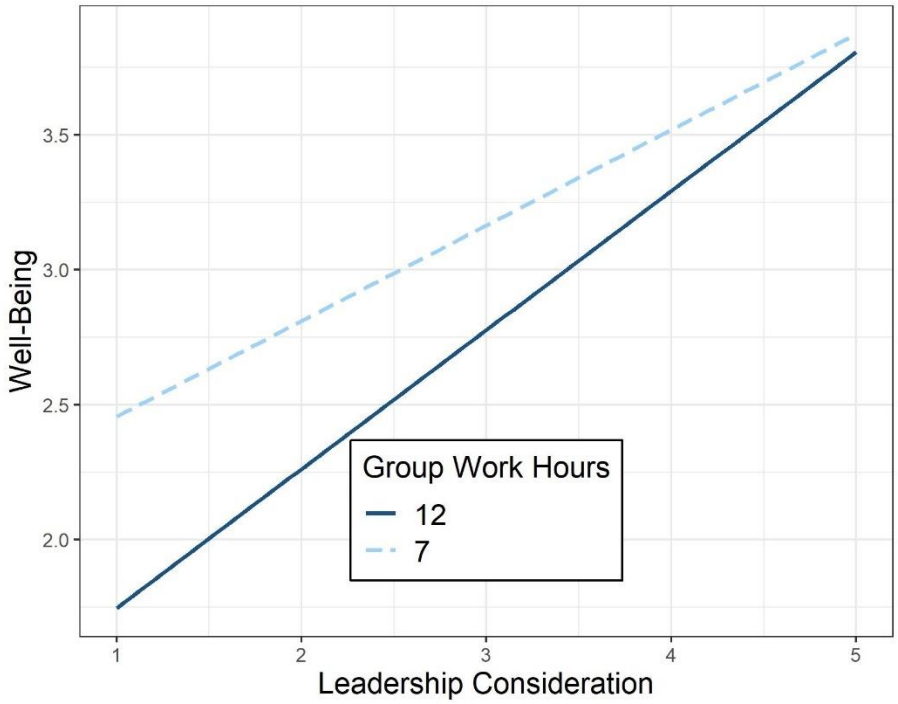
To plot the form of the interaction and make use of the graphics capabilities of `ggplot2`, we can use the lme4 package and rerun the model using `lmer`. The code also uses the lmerTest package for p-values and degrees of freedom and changes the optimizer because the default failed to converge.

```
> library(lme4)
> library(lmerTest)

> Model.2b<-lmer(WBEING~HRS+LEAD*G.HRS+(LEAD|GRP), data=bh1996,
+                control=lmerControl(optimizer = "Nelder_Mead"))

> summary(Model.2b)$coef
               Estimate  Std. Error         df   t value     Pr(>|t|)
(Intercept)  3.64325839 0.732553188   87.67621  4.973370 3.243398e-06
HRS         -0.02855876 0.004468026 7287.99657 -6.391807 1.740410e-10
LEAD         0.12894421 0.218811339   89.83115  0.589294 5.571432e-01
G.HRS       -0.17401949 0.064152902   87.45942 -2.712574 8.038535e-03
LEAD:G.HRS   0.03216543 0.019187381   89.79663  1.676384 9.714129e-02
```

With a lmer model, we can use the interactions library and the following code to plot values for group averages of 7 hours versus 12 hours of work.

```
library(interactions)
library(ggplot2)
win.graph(height=4.75,width=6) #quartz() for MAC

interact_plot(Model.2b,pred=LEAD,modx=G.HRS,
              modx.values = c(7,12),
              x.label = "Leadership Consideration",
              y.label = "Well-Being",
              legend.main="Group Work Hours")+
  theme_bw()+
  theme(legend.background=element_rect(fill="white",
                  size=0.5, linetype="solid",color ="black"),
        legend.position = c(0.5, 0.2),
        axis.title.x = element_text(color="black", size=14),
        axis.title.y = element_text(color="black", size=14),
        legend.title = element_text(color="black", size=14),
        legend.text = element_text(color="black", size=14)
  )

ggsave(filename = "c:\\temp\\plotgg.jpg",
```

```
device = "jpeg")
```

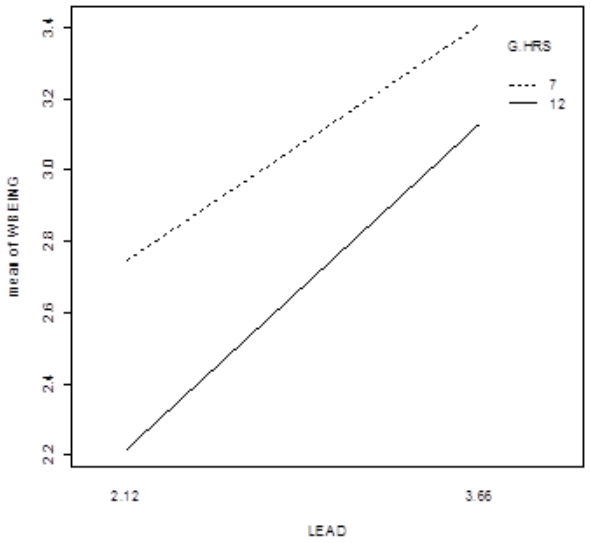


Soldiers' perceptions of leadership consideration are positively related to their well-being regardless of the group average work hours. The relationship between individuals' ratings of leadership consideration and their well-being is stronger (steeper slope) in groups with high work hours than in groups with low work hours. Another way to think about the interaction is to note that well-being really drops (in relative terms) when a soldier perceives that leadership is low in consideration and one is a member of a group with high average work hours. This pattern supports our proposition that considerate leadership is relatively more important in a high work demand context.

4.2 Plotting with `interaction.plot`

The previous example used the lme4, interactions, and ggplot2 library to make a publication quality plot. A quick alternative is to use the `interaction.plot` function illustrated below.

```
> Final.Model<-lme(WBEING~HRS+LEAD+G.HRS+LEAD:G.HRS,
  random=~LEAD|GRP,data=bh1996,control=list(opt="optim"))

> TDAT<-data.frame(HRS=c(11.2987,11.2987,11.2987,11.2987),
                   LEAD=c(2.12,2.12,3.66,3.66),
                   G.HRS=c(7, 12, 7, 12),
                   GRP=c(1,1,1,1))

> TDAT$WBEING<-predict(Final.Model,TDAT,level=1)
> with(TDAT,interaction.plot(LEAD,G.HRS,WBEING))
```



4.3 Some Notes on Centering

In multilevel modeling, centering issues is a major consideration. In our examples, we have used raw variables as predictors. In some cases, though, there may be good reasons to consider centering the level-1 variables with one of two centering options.

Level-1 variables such as leadership can be grand-mean centered or group-mean centered. Grand-mean centering is often worth considering because doing so helps reduce multicollinearity among predictors and random effect terms. In cases where interactive terms are included, grand-mean centering can be particularly helpful in reducing correlations between main-effect and interactive terms. Hofmann and Gavin (1998) and others have shown that grand-mean centered and raw variable models produce identical results for the predictors; however, grand-mean centered models may converge in situations where a model based on raw variables will not.

Grand-mean centering can be accomplished in one of two ways. The explicit way is to subtract the overall mean from the raw variable. The less obvious way is to use the `scale` function. The `scale` function is used to standardize (mean=0, sd=1) variables, but can also be used to grand-mean center if the `scale=FALSE` option is selected. Below I create grand-mean centered variables for leadership both ways.

```
> bh1996$GRAND.CENT.LEAD<-bh1996$LEAD-mean(bh1996$LEAD)
> bh1996$GRAND.CENT.LEAD<-scale(bh1996$LEAD,scale=FALSE)
```

Group-mean centering (demeaning) is another centering option with level-1 variables. In group-mean centering, each individual score is subtracted from the group mean. Review section 3.1 and the `aggregate` and `merge` functions for assigning a group-mean variable back to each individual. Once a group mean is assigned back to the individual, simply subtract the group mean from the raw score. A group-mean centered variable reflects how much an individual differs from their group average. Group-mean centering represents a different parameterization of the model than does the raw or grand-mean centered version (Hofmann & Gavin, 1998; Hox, 2002; Snijders & Bosker, 1999).

4.3.1.1 Centering and Cross-Level Interactions

There is value in using group-mean centering when testing a cross-level interaction. Bryk and Raudenbush (1992) and Hofmann and Gavin (1998) point out that group-mean centering provides the "purest" estimate of the within-group slope in these situations. That is, slope estimates based on raw variables and grand-mean centered variables can be partially influenced by between-group factors. In contrast, group-mean centered variables have had between-group effects removed. Bryk and Raudenbush (1992) show that group-level interactions can sometimes pose as cross-level interactions, so a logical strategy is to use raw or grand-mean centered variables to test for cross-level interactions but verify the final results with group-mean centered variables.

The `bh1996` dataframe has group-mean centered variables for all the predictors beginning with a "W" for "within". For comparisons, the first model uses a raw leadership variable and the second model below uses the group-mean centered leadership variable in both the fixed part of the model and in the random statement.

```
> Final.Model<-lme(WBEING~HRS+LEAD+G.HRS+LEAD:G.HRS,
+                  random=~LEAD|GRP,data=bh1996, control=list(opt="optim"))
> round(summary(Final.Model)$tTable,dig=3)
             Value Std.Error   DF t-value p-value
(Intercept)  3.654     0.726 7280   5.032   0.000
HRS         -0.029     0.004 7280  -6.391   0.000
LEAD         0.126     0.217 7280   0.578   0.564
G.HRS       -0.175     0.064   97  -2.751   0.007
LEAD:G.HRS   0.032     0.019 7280   1.703   0.089

> Final.Model.R<-lme(WBEING~HRS+W.LEAD+G.HRS+W.LEAD:G.HRS,
+              random=~W.LEAD|GRP,data=bh1996, control=list(opt="optim"))
> round(summary(Final.Model.R)$tTable,dig=3)
              Value Std.Error   DF t-value p-value
(Intercept)   4.733     0.214 7280  22.080   0.000
HRS          -0.028     0.004 7280  -6.271   0.000
W.LEAD        0.055     0.223 7280   0.249   0.804
G.HRS        -0.145     0.019   97  -7.471   0.000
W.LEAD:G.HRS  0.040     0.020 7280   2.037   0.042
```

Notice that the cross-level interaction is now significant with a t-value of 2.037 versus 1.703 in the model with raw variable. Thus, there are some minor differences between the two model specifications, but it would appear there is a significant cross-level interaction ($p<.05$) in the pure specification. For an interesting example of trying to determine whether cohesion buffering effects are cross-level or group-mean interactions see Campbell-Sills et al., (2022).

4.3.1.2 Centering and Contextual Models

Centering choice also has important implications for interpreting contextual models. When contextual models are based on raw level-1 variables, the level-2 coefficient represents the difference between the two slopes. In contrast, when the level-1 variable is group-mean centered, the level-2 coefficient captures the total effect (the level-1 slope plus any difference) and tests whether this total effect is different from zero. Below are the two models.

```
> tmod.raw<-lme(WBEING~HRS+G.HRS,random=~1|GRP,bh1996)
> round(summary(tmod.raw)$tTable, dig=3)
             Value Std.Error   DF t-value p-value
(Intercept)  4.741     0.214 7282  22.187       0
HRS         -0.046     0.005 7282  -9.505       0
G.HRS       -0.127     0.019   97  -6.542       0
>
> tmod.cent<-lme(WBEING~W.HRS+G.HRS,random=~1|GRP,bh1996)
> round(summary(tmod.cent)$tTable, dig=3)
             Value Std.Error   DF t-value p-value
(Intercept)  4.741     0.214 7282  22.187       0
W.HRS       -0.046     0.005 7282  -9.505       0
G.HRS       -0.173     0.019   97  -9.234       0
```

The first model indicates that the G.HRS slope is -0.127 stronger than the within slope of -0.046. The model represents a relative test. The second model tests whether the total between-group slope of -0.173 differs from zero. It is relatively common for researchers to make errors when interpreting these two variants of the model (see Bliese et al., 2018).

4.4 Estimating Group-Mean Reliability (ICC2) with `gmeanrel`

In mixed-effects models, it is possible to obtain an estimate of the group-mean reliability analogous to the ICC(2) (see section 3.2.7). Group mean reliability estimates are a function of the ICC and group size (see Bliese, 2000; Bryk & Raudenbush, 1992), and the `gmeanrel` function from the multilevel package calculates the ICC, the group size, and the group mean reliability for each group.

The code below illustrates the `gmeanrel` function on the `bhr2000` data set to show how the results compare to results in section 3.2.7 where the ICC(1) estimate from the ANOVA model was 0.174 and the ICC(2) estimate was 0.920.

```
> Null.Model<-lme(HRS~1,random=~1|GRP,data=bhr2000,
  control=list(opt="optim"))

> GREL.DAT<-gmeanrel(Null.Model)
> names(GREL.DAT)
[1] "ICC"     "Group"   "GrpSize" "MeanRel"

> GREL.DAT$ICC  #ICC estimate
[1] 0.177544

> GREL.DAT$MeanRel[1:20] #First 20 Reliability Estimates
 [1] 0.9272005 0.9066657 0.9471382 0.8487743 0.9465280
 [6] 0.7754791 0.7953197 0.8192754 0.8699945 0.8831157
[11] 0.8119385 0.8622636 0.9379303 0.9452644 0.9260382
[16] 0.8487743 0.9395503 0.9315061 0.8622636 0.9235985


> mean(GREL.DAT$MeanRel)
[1] 0.8955047
```

The ICC estimate is 0.178 (the same as the value produced by `mult.icc` in section 3.2.8) and slightly higher than the ANOVA based estimate of 0.174. The average group-mean

reliability from `gmeanrel` is 0.896 which is smaller (but close) to the value of 0.920 from the ANOVA model. The output also illustrates that each group receives a separate estimate of group-mean reliability. Values vary as a function of group size.

5 Growth Modeling Repeated Measures Data

Growth models are an important variation of multilevel models (see section 4). In growth models repeated observations from an individual represent the level-1 variables, and the attributes of the individual represent the level-2 variables. The fact that the level-1 variables are repeated over time poses some additional analytic issues; however, the steps used to analyze the basic growth model and the steps used to analyze a multilevel model share many key similarities.

This chapter begins by briefly reviewing some of the methodological challenges associated with growth modeling. Following this, the chapter illustrates how data must be configured to conduct growth modeling. Finally, the chapter illustrates a complete growth modeling analysis using the `nlme` package. Much of this material is adapted from Bliese and Ployhart (2002).

5.1 Methodological challenges

Since longitudinal data is collected from single entities over multiple times, it is likely that there will be a high degree of non-independence in the responses. Multiple responses from an individual will tend to be related by virtue of being provided by the same person, and this non-independence violates the statistical assumption of independence underlying many common data analytic techniques (Kenny & Judd, 1986).

Issues about non-independence are similar to those that occur when working with lower-level data nested in higher-level groups. In longitudinal designs, however, there are additional complications associated with the lower-level responses. First, it is likely that responses temporally close to each other (e.g., responses 1 and 2) will be more strongly related than responses temporally far apart (e.g., responses 1 and 4). This pattern is defined as a simplex pattern or lag 1 autocorrelation in the residuals. Second, it is likely that responses will tend to become either more variable over time or less variable over time. For instance, individuals starting jobs may initially have a low degree of variability in performance, but over time the variance in job performance may increase. In statistical terms, outcome variables in longitudinal data are likely to display heteroscedasticity. To obtain correct standard errors and to draw the correct statistical inferences, autocorrelation, and heteroscedasticity both need to be incorporated into the statistical model.

The need to test for both autocorrelation and heteroscedasticity in growth models arises because the level-1 variables (repeated measures from an individual) are ordered by time. One of the main differences between growth models and multilevel models revolves around understanding how to properly account for time in both the statistical models and in the data structures.

In R, growth modeling can be estimated using the `lme` function from the `nlme` package (Pinhiero & Bates, 2000). The `lme` function is the same function used in multilevel modeling (see section 4); however, the `nlme` package has a variety of options available for handling autocorrelation and heteroscedasticity in growth models.

Before conducting growth modeling, the data has to be set up in a way that explicitly includes time as a variable. This data transformation is referred to as changing a data set from multivariate to stacked, long, or univariate form. In the next section, we show how to create a dataframe for growth modeling.

5.2 Data Structure and the `make.univ` Function

Often data are stored in a format where each row represents observations from one individual. For instance, an individual might provide three measures of job satisfaction in a longitudinal study, and the data might be arranged in multivariate form such that column 1 is the subject number; column 2 is job satisfaction at time 1; column 3 is job satisfaction at time 2, and column 4 is job satisfaction at time 3, etc.

The `univbct` dataframe in the multilevel library allows us to illustrate a common way of storing repeated measures data. This data set contains three measures taken six-months apart on three variables – job satisfaction, commitment, and readiness. It also contains some stable individual characteristics such as respondent gender, marital status and age at the initial data collection time. These latter variables are treated as level-2 predictors in subsequent modeling.

The `univbct` dataframe is already in univariate form; however, for the purposes of illustration, we will select a subset of the entire `univbct` dataframe and transform it back into multivariate form. With this subset we will illustrate how to convert a typical multivariate dataframe back into the univariate form necessary for growth modeling.

```
> library(multilevel)
> data(univbct)
> names(univbct)
 [1] "BTN"     "COMPANY" "MARITAL" "GENDER"  "HOWLONG" "RANK"    "EDUCATE"
 [8] "AGE"     "JOBSAT1" "COMMIT1" "READY1"  "JOBSAT2" "COMMIT2" "READY2"
[15] "JOBSAT3" "COMMIT3" "READY3"  "TIME"    "JSAT"    "COMMIT"  "READY"
[22] "SUBNUM"
> nrow(univbct)
[1] 1485
> length(unique(univbct$SUBNUM))
[1] 495
```

These commands indicate there are 1485 rows in the data set representing 495 individuals so each individual provides three rows of data. To create a multivariate data set out of the `univbct` dataframe, we can select the first row for each participant in the `univbct` dataframe. In this illustration we restrict our analyses to the three job satisfaction scores and to respondent age at the initial data collection period.

```
> GROWDAT<-univbct[!duplicated(univbct$SUBNUM),c(22,8,9,12,15)]
> GROWDAT[1:3,]
  SUBNUM AGE  JOBSAT1 JOBSAT2 JOBSAT3
1      1  20 1.666667       1       3
4      2  24 3.666667       4       4
7      3  24 4.000000       4       4
```

The dataframe GROWDAT now contains data from 495 individuals. The first individual was 20 years old at the first data collection time. At time 1, the first individual's job satisfaction score was 1.67; at time 2 it was 1.0, and at time 3 it was 3.0.

Because the univbct dataframe in the multilevel package was already in univariate form, we illustrated the additional steps of converting it back to multivariate form. From a practical standpoint, though, the important issue is that the GROWDAT dataframe now represents a typical multivariate data set containing repeated measures. Specifically, the GROWDAT dataframe contains one row of information for each subject, and the repeated measures (job satisfaction) are represented by three different variables.

From a growth modeling perspective, the key problem with multivariate dataframes like GROWDAT is that they do not contain a variable that indexes time. That is, we know time is an attribute of this data because we have three different measures of job satisfaction; however, analytically we have no way of explicitly modeling time in the multivariate form of the data. Therefore, it is critical to create a new variable that explicitly indexes time which requires transforming the data to univariate or a stacked format.

The make.univ function from the multilevel package provides a simple way to perform this transformation. Two arguments are required (x and dvs), and two are optional (tname and outname). The first required argument is the dataframe in multivariate or wide format. The second required argument is a subset of the entire dataframe identifying the columns containing the repeated measures. The second required argument must be time-sorted -- column 1 must be time 1, column 2 must be time 2, and so on. The two optional arguments control the names of the two created variables: tname defaults to "TIME" and outname defaults to "MULTDV".

For instance, to convert GROWDAT into univariate form we issue the following command:

```
> UNIV.GROW<-make.univ(GROWDAT,GROWDAT[,3:5])
> UNIV.GROW[1:9,]
    SUBNUM AGE  JOBSAT1 JOBSAT2 JOBSAT3 TIME   MULTDV
1        1  20 1.666667       1       3    0 1.666667
1.1      1  20 1.666667       1       3    1 1.000000
1.2      1  20 1.666667       1       3    2 3.000000
4        2  24 3.666667       4       4    0 3.666667
4.1      2  24 3.666667       4       4    1 4.000000
4.2      2  24 3.666667       4       4    2 4.000000
7        3  24 4.000000       4       4    0 4.000000
7.1      3  24 4.000000       4       4    1 4.000000
7.2      3  24 4.000000       4       4    2 4.000000
```

Note that each individual now has three rows of data indexed by the variable "TIME". Time ranges from 0 to 2. To facilitate model interpretation, the first time is coded as 0 instead of as 1. Doing so allows one to interpret the intercept in subsequent models as the level of job satisfaction at the initial data collection time. Second, notice that the make.univ function has created a variable called "MULTDV". This variable represents the multivariate dependent variable. The variable "TIME" and the variable "MULTDV" are two of the primary variables used in growth modeling. Finally, notice that AGE, SUBNUM and the values for the three job satisfaction variables were repeated three times for each individual. By repeating the individual variables, the make.univ function has essentially created a dataframe with level-2 variables in

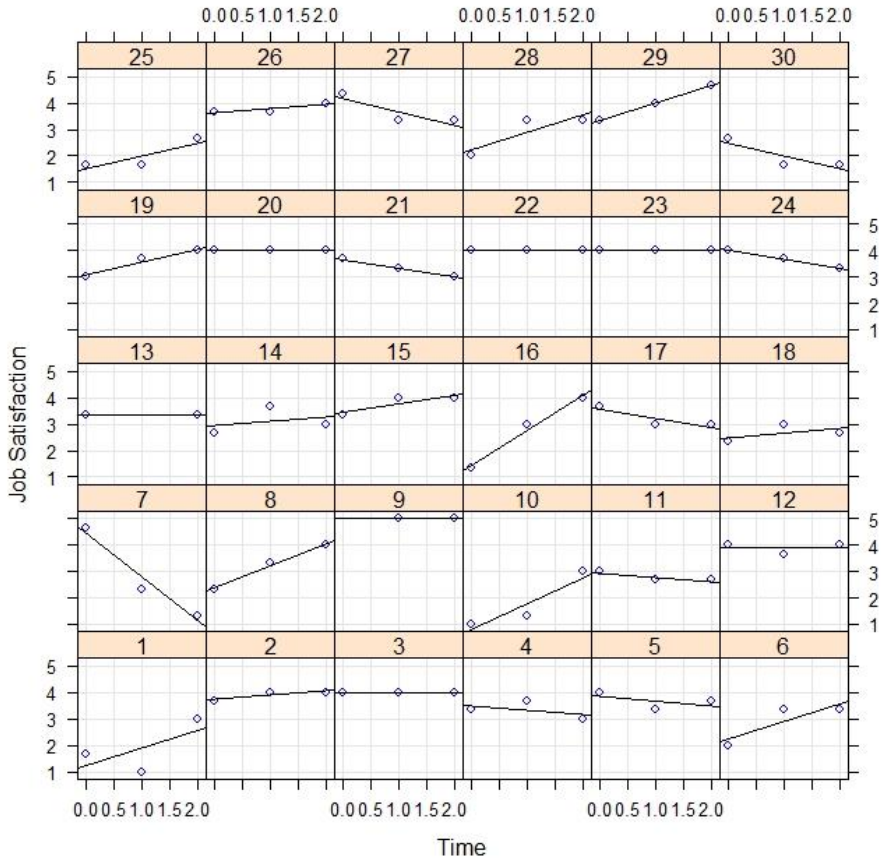
the proper format. For instance, subject age can now be used as a level-2 predictor in subsequent modeling.

In many cases, one may have only one dependent variable that needs to be converted into univariate or stacked format and therefore the `make.univ` function will suffice. If, however, it is necessary to create a univariate dataframe with multiple variables indexed by time, the `mult.make.univ` function in the `multilevel` package is available as is the `reshape` function in the base R program (see help files).

5.3 Growth Modeling Illustration

With the data in univariate form, we can begin to visually examine whether we see patterns between time and the outcome. For instance, the commands below use the `lattice` package to produce a plot of the first 30 individuals:

```
>library(lattice)
>xyplot(MULTDV~TIME|as.factor(SUBNUM),data=UNIV.GROW[1:90,],
 type=c("p","r","g"),col="blue",col.line="black",
 xlab="Time",ylab="Job Satisfaction")
```



From this plot, it appears as though there is considerable variability both in overall levels of job satisfaction and in how job satisfaction changes over time. The goal in growth modeling is to determine whether we can find consistent patterns in the relationship between time and job

satisfaction. Therefore, we are now ready to illustrate growth modeling in a step-by-step approach. In this illustration, we follow the model comparison approach outlined by Bliese and Ployhart (2002) and in also discussed in Ployhart, Holtz and Bliese (2002).

As an overview of the steps, the basic procedure is to start by examining the nature of the outcome. Much as we did in multilevel modeling, we are interested in estimating the ICC and determining whether the outcome (job satisfaction) randomly varies among individuals. Second, we are interested in examining the form of the relationship between time and the outcome. Basically, we want to know whether the outcome generally increases, decreases, or shows some other type of relationship with time. The plot of the first 30 individuals shows no clear pattern in how job satisfaction is changing over time, but the analysis might identify an overall trend among the 495 respondents. Third, we attempt to determine whether the relationship between time and the outcome is constant among individuals or whether it varies on an individual-by-individual basis. Fourth, we model in more complicated error structures such as autocorrelation, and finally we add level-2 predictors of intercept and slope variances.

5.3.1 Step 1: Examine the DV

The first step in growth modeling is to examine the properties of the dependent variable by estimating a null model and calculating the ICC.

```
> null.model<-lme(MULTDV~1,random=~1|SUBNUM,data=UNIV.GROW,
na.action=na.omit, control=list(opt="optim"))

> VarCorr(null.model)
SUBNUM = pdLogChol(1)
            Variance  StdDev
(Intercept) 0.4337729 0.6586144
Residual    0.4319055 0.6571952

> 0.4337729/(0.4337729+0.4319055)
[1] 0.5010786
```

In our example, the ICC associated with job satisfaction is .50 indicating that 50% of the variance in any individual report of job satisfaction can be explained by the properties of the individual who provided the rating. Another way to think about this is that individuals tend to remain consistent in ratings over time (a person who has high job satisfaction at one time will then to have high job satisfaction at other times). At the same time, an ICC of .50 is low enough to allow for within-person change over time. In practice, ICC values between .30 and .70 tend to be good when modeling change over time.

5.3.2 Step 2: Model Time

Step two involves modeling the fixed relationship between time and the dependent variable. In almost all cases, it is logical to begin by modeling a linear relationship and progressively add more complicated relationships such as quadratic, cubic, etc. To test whether there is a linear relationship between time and job satisfaction, we regress job satisfaction on time in a model with a random intercept.

```
> model.2<-lme(MULTDV~TIME,random=~1|SUBNUM,data=UNIV.GROW,
na.action=na.omit,control=list(opt="optim"))
> summary(model.2)$tTable
```

```
                Value  Std.Error  DF   t-value    p-value
(Intercept) 3.21886617 0.04075699 903 78.977040 0.00000000
TIME        0.05176461 0.02168024 903  2.387640 0.01716169
```

Results indicate a significant linear relationship between time and job satisfaction such that job satisfaction increases by .05 each time period. Because the first time period was coded as 0, the intercept value of 3.22 represents the average level of job satisfaction at the first time period.

More complicated time functions can be included in one of two ways – either through raising the time variable to various powers, or by converting time into power polynomials. Both techniques are illustrated.

```
> model.2b<-lme(MULTDV~TIME+I(TIME^2),random=~1|SUBNUM,
data=UNIV.GROW,na.action=na.omit,control=list(opt="optim"))

> summary(model.2b)$tTable
                  Value  Std.Error  DF    t-value   p-value
(Intercept)  3.23308157 0.04262697 902 75.8459120 0.0000000
TIME        -0.03373846 0.07816572 902 -0.4316273 0.6661154
I(TIME^2)    0.04276425 0.03756137 902  1.1385167 0.2552071

> model.2c<-lme(MULTDV~poly(TIME,2),random=~1|SUBNUM,
data=UNIV.GROW,na.action=na.omit,control=list(opt="optim"))
> summary(model.2c)$tTable
                   Value Std.Error  DF   t-value    p-value
(Intercept)    3.2704416 0.0346156 902 94.478836 0.00000000
poly(TIME, 2)1 1.5778835 0.6613714 902  2.385775 0.01724863
poly(TIME, 2)2 0.7530736 0.6614515 902  1.138517 0.25520707
```

Neither model finds evidence of a significant quadratic trend. Note that a key advantage of the power polynomials is that the linear and quadratic effects are orthogonal. Consequently, in the second model the linear effect of time is still significant even with the quadratic effect in the model so only one model needs to be estimated to identify both the linear and quadratic effects. When squaring time, it is important to run the linear model before running the model with both the linear and quadratic effect to ensure that the linear effect is identified.

5.3.3 Step 3: Model Slope Variability

A potential limitation with model 2 is that it assumes that the relationship between time and job satisfaction is constant for all individuals. Specifically, it assumes that each individual increases job satisfaction by .05 points at each time. An alternative model is one that allows slopes to vary. Given the degree of variability in the graph of the first 30 respondents, a random slope model seems like a plausible alternative. The random slope model is tested by adding the linear effect for time as a random effect. In the running example, we can `update` model.2 by adding a different random effect component and contrast model 2 and model 3.

```
> model.3<-update(model.2,random=~TIME|SUBNUM)
> anova(model.2,model.3)
        Model df      AIC      BIC    logLik   Test  L.Ratio p-value
model.2     1  4 3461.234 3482.194 -1726.617
model.3     2  6 3434.132 3465.571 -1711.066 1 vs 2 31.10262  <.0001
```

The results show that a model that allows the slope between time and job satisfaction to vary across individuals fits the data better than a model that fixes the slope to be a constant value. In cases where higher-level trends were also significant, one would also be interested in determining whether allowing the slopes of the higher-level variables to randomly vary also improved model fit. For instance, one might find that a quadratic relationship varied in strength among individuals.

5.3.4 Step 4: Modeling Error Structures

The fourth step in developing the level-1 model involves assessing the error structure of the model. It is important to scrutinize the level-1 error structure because significance tests may be affected if error structures are not properly specified. The goal of step 4 is to determine whether one's model fit improves by incorporating (a) an autoregressive structure with serial correlations and (b) heterogeneity in the error structures.

Tests for autoregressive structure (autocorrelation) are conducted by including the `correlation` option in `lme`. For instance, we can update model.3 and include lag 1 autocorrelation as follows:

```
> model.4a<-update(model.3,correlation=corAR1())
> anova(model.3,model.4a)
         Model df      AIC      BIC    logLik   Test  L.Ratio p-value
model.3      1  6 3434.132 3465.571 -1711.066
model.4a     2  7 3429.771 3466.451 -1707.886 1 vs 2 6.360465  0.0117
```

A model that allows for autocorrelation fits the data better than does a model that assumes no autocorrelation. A summary of model 4a reveals that the autocorrelation estimate is .367 (see the Phi coefficient).

```
> summary(model.4a)
Linear mixed-effects model fit by REML
 Data: UNIV.GROW
       AIC      BIC    logLik
  3429.771 3466.451 -1707.886
.....
Correlation Structure: AR(1)
 Formula: ~1 | SUBNUM
 Parameter estimate(s):
      Phi
0.3676831
```

It is important to note that the use of `correlation=corAR1()` in the default mode assumes data is structured such that time increases for each individual. Stacked data created using `make.univ` has this structure. If data are imported or otherwise manipulated so that this order is not maintained, it will be necessary either to re-order the dataframe or to specify the structure to `corAR1()` with more detail (see help files). For example, if the rows in GROW.UNIV are randomly ordered, the estimate for AR 1 changes:

```
> UNIV.GROW2<-UNIV.GROW[order(rnorm(1485)),]
> UNIV.GROW2[1:10,]
       SUBNUM AGE  JOBSAT1  JOBSAT2  JOBSAT3 TIME   MULTDV
6           2  24 3.666667 4.000000 4.000000    0 3.666667
285.2      93  20 2.333333 3.000000 3.000000    2 3.000000
```

```
339.2     109  33 3.666667 3.000000 3.333333    2 3.333333
228        74  23 5.000000       NA 5.000000    0 5.000000
894       294  37 4.000000 4.000000 4.000000    0 4.000000
1029.1    339  20 3.000000 3.333333 3.000000    1 3.333333
1416      468  20 3.333333 3.333333 3.666667    0 3.333333
696.2     228  19 4.000000 2.666667 3.333333    2 3.333333
735.1     241  25 3.666667 3.000000 3.000000    1 3.000000
51         17  20 3.666667 3.000000 3.000000    0 3.666667

> tmod<-lme(MULTDV~TIME,random=~1|TIME,na.action=na.omit,
data=UNIV.GROW2,corAR1())

> summary(tmod)
Linear mixed-effects model fit by REML
 Data: UNIV.GROW2
       AIC      BIC    logLik
  3766.914 3793.113 -1878.457
...
Correlation Structure: AR(1)
 Formula: ~1 | TIME
 Parameter estimate(s):
       Phi
0.05763463
```

In the truncated results, notice how the estimate of the phi-coefficient changed (replications will result in different estimates of the phi-coefficient because of different structures associated with the random sorting of the data). To ensure the data is in the proper structure the `order` function can be used on any dataframe to restructure by higher-level entity and time:

```
> UNIV.GROW3<-UNIV.GROW2[order(UNIV.GROW2$SUBNUM,UNIV.GROW2$TIME),]
> UNIV.GROW3[1:10,]
    SUBNUM AGE  JOBSAT1  JOBSAT2 JOBSAT3 TIME   MULTDV
3        1  20 1.666667 1.000000       3    0 1.666667
3.1      1  20 1.666667 1.000000       3    1 1.000000
3.2      1  20 1.666667 1.000000       3    2 3.000000
6        2  24 3.666667 4.000000       4    0 3.666667
6.1      2  24 3.666667 4.000000       4    1 4.000000
6.2      2  24 3.666667 4.000000       4    2 4.000000
9        3  24 4.000000 4.000000       4    0 4.000000
9.1      3  24 4.000000 4.000000       4    1 4.000000
9.2      3  24 4.000000 4.000000       4    2 4.000000
12       4  23 3.333333 3.666667       3    0 3.333333
```

Finally, we can examine the degree to which the variance of the responses changes over time using the `varExp` option (see Pinheiro & Bates, 2000 for details).

```
> model.4b<-update(model.4a,weights=varExp(form=~TIME))
> anova(model.4a,model.4b)
         Model df      AIC      BIC    logLik   Test  L.Ratio p-value
model.4a     1  7 3429.771 3466.451 -1707.886
model.4b     2  8 3428.390 3470.309 -1706.195 1 vs 2 3.381686  0.0659
```

The model that includes both autocorrelation and allows for decreases in variance fits the data marginally better (using a liberal p-value) than does the model that only includes autocorrelation.

In subsequent analyses, however, `model.4b` ran into convergence problems. Consequently, we elect to use `model.4a` as our final level-1 model.

With the completion of step 4, we have exhaustively examined the form of the level-1 relationship between time and job satisfaction. This analysis has revealed that (a) individuals vary in terms of their mean levels of job satisfaction, (b) there is a linear, but not quadratic, relationship between time and job satisfaction, (c) the strength of the linear relationships varies among individuals, and (d) there is significant autocorrelation in the data. At this point, we are ready to add level-2 variables to try and explain the random variation in intercepts (i.e., mean job satisfaction) and in the time-job satisfaction slope.

5.3.5 Step 5: Predicting Intercept Variation

Step 5 in growth modeling is to examine factors that can potentially explain intercept variation. In our case, we are interested in examining factors that explain why some individuals have high job satisfaction while other individuals have low job satisfaction. In this example, we explore the idea that age at the first data collection time is related to intercept variation.

To model this relationship, the individual-level characteristic, age, is used as a predictor of the job satisfaction intercept. The model that we will test is represented below using the Bryk and Raudenbush (1992) notation.

$$Y_{ij} = \pi_{0j} + \pi_{1j}(Time_{ij}) + r_{ij}$$
$$\pi_{0j} = \beta_{00} + \beta_{01}(Age_j) + u_{0j}$$
$$\pi_{1j} = \beta_{10} + u_{1j}$$

This equation states that respondent j's mean level of job satisfaction (π_{0j}) can be modeled as a function of two things. One is the mean level of job satisfaction (β_{00}) for all respondents. The second is a coefficient associated with the individual's age (β_{01}). Note that the error term for the intercept (u_{0j}) now represents the difference between an individuals' mean job satisfaction and the overall job satisfaction after accounting for age. In `lme` the model is specified as:

```
> model.5<-lme(MULTDV~TIME+AGE,random=~TIME|SUBNUM,
  correlation=corAR1(),na.action=na.omit,data=UNIV.GROW,
  control=list(opt="optim"))

> round(summary(model.5)$tTable,dig=3)
            Value Std.Error  DF t-value p-value
(Intercept) 2.347     0.146 897  16.086   0.000
TIME        0.053     0.024 897   2.205   0.028
AGE         0.034     0.005 486   6.241   0.000
```

Model 5 differs only from Model 4a in that Model 5 includes AGE (age at the baseline survey). Notice that AGE is positively related to levels of job satisfaction. Also notice that there are fewer degrees of freedom for AGE than for TIME since AGE is an individual (level-2) attribute. The AGE parameter indicates that a 23-year-old in the baseline survey would have average job satisfaction scores across the three times that were 0.034 higher than a 22-year-old in the baseline survey.

5.3.6 Step 6: Predicting Slope Variation

The final aspect of growth modeling involves attempting to determine attributes of individual respondents that are related to slope variability. In this section, we attempt to determine whether respondent age can explain some of the variation in the time-job satisfaction slope. The model that we test is presented below:

$$Y_{ij} = \pi_{0j} + \pi_{1j}(Time_{ij}) + r_{ij}$$
$$\pi_{0j} = \beta_{00} + \beta_{01}(Age_j) + u_{0j}$$
$$\pi_{1j} = \beta_{10} + \beta_{11}(Age_j) + u_{1j}$$

This model is similar to the model specified in step 5 except that we now test the assumption that the slope between time and job satisfaction for an individual (π_{1j}) is a function of an overall slope (β_{10}), individual age (β_{11}), and an error term (u_{1j}). In `lme`, the model is specified as:

```
> model.6<-lme(MULTDV~TIME*AGE,random=~TIME|SUBNUM,
  correlation=corAR1(),na.action=na.omit,data=UNIV.GROW,
  control=list(opt="optim"))
```

Note that the only difference between model 5 and model 6 is that we include an interaction term for TIME and AGE. A summary of model 6 reveals a significant interaction.

```
> round(summary(model.6)$tTable,dig=3)
             Value Std.Error  DF t-value p-value
(Intercept)  2.098     0.186 896  11.264   0.000
TIME         0.271     0.104 896   2.608   0.009
AGE          0.043     0.007 486   6.180   0.000
TIME:AGE    -0.008     0.004 896  -2.153   0.032
```

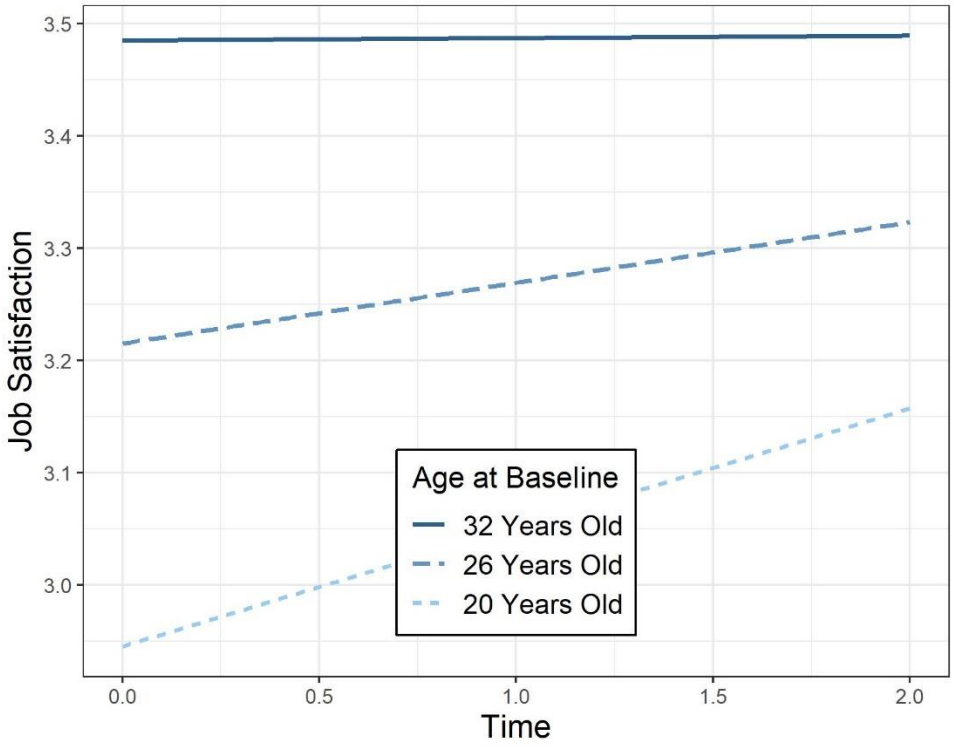
5.3.7 Plot Growth Model Using the lme4 Package and Interactions Library

To plot we first re-estimate the model in the lme4 package. The `lmer` function does not have the option to control for autocorrelation, but we can see that omitting this option does not change our substantive interpretation.

```
> library(lme4)
> library(lmerTest)

> model.6a<-lmer(MULTDV~TIME*AGE+(TIME|SUBNUM),data=UNIV.GROW)
> round(summary(model.6a)$coef,dig=3)
            Estimate Std. Error      df t value Pr(>|t|)
(Intercept)    2.078      0.186 470.301  11.176    0.000
TIME           0.273      0.104 462.965   2.630    0.009
AGE            0.044      0.007 469.523   6.276    0.000
TIME:AGE      -0.008      0.004 461.280  -2.169    0.031
```

The code below uses the `lmer` model to produce a plot using the defaults of the mean and one standard deviation above and below the mean AGE (a 32, 26 and 20 year old).

```
library(interactions)
library(ggplot2)
win.graph(height=4.75,width=6)
interact_plot(model.6a,pred=TIME,modx=AGE,
              modx.labels = c("20 Years Old","26 Years Old",
                              "32 Years Old"),
              x.label = "Time",
              y.label = "Job Satisfaction",
              legend.main="Age at Baseline")+
  theme_bw()+
  theme(legend.background=element_rect(fill="white",
           size=0.5, linetype="solid",color ="black"),
        legend.position = c(0.5, 0.2),
        axis.title.x = element_text(color="black", size=14),
        axis.title.y = element_text(color="black", size=14),
        legend.title = element_text(color="black", size=13,
                                    hjust=.5),
        legend.text = element_text(color="black", size=12)
  )
ggsave(filename = "c:\\temp\\plotgg.jpg",
       device = "jpeg")
```



Older individuals at baseline reported higher job satisfaction initially and tended to show a very slight increase over time. In contrast, younger respondents tended to report lower initial job satisfaction, but showed a more pronounced increase in job satisfaction over time.

5.4 Discontinuous Growth Models

In the previous example (section 5.3.2), two variants of time were examined (linear and quadratic). Indeed, with only three periods it is difficult to explore more than a linear and quadratic trend (through one could treat time as a categorical variable and make no assumptions about trends). In situations where numerous repeated measures are collected, however, a variety of interesting options exist for modeling time.

One particularly interesting variant is the discontinuous growth model (DGM) a model also referred to as the piecewise hierarchical linear model (Raudenbush & Bryk, 2002; Hernández-Lloreda et al., 2004) or the multiphase mixed-effects model (Cudeck & Klebe, 2002). The basic idea behind the DGM is to simultaneously use a set of two or three time-related covariates to capture a known discontinuity.

For instance, Lang and Bliese (2009) use the DGM to model the performance impact of unexpectedly changing key elements of a complex computer-based task. In the design, participants worked on the task for six trials and then on the seventh trial the task became substantially more difficult. Although there are numerous variants for modeling a discontinuity of this nature (see Bliese & Lang, 2016), the basic form can be captured by the three terms TIME, TRANS, and POST. Because these time-varying predictors represent a system of equations, TIME captures the initial linear trend; TRANS captures the immediate response to the event, and POST captures the post-transition slope change. A fourth useful variant is to include a TIME.A (for absolute) that results in expressing the TRANS and POST parameters in absolute versus relative terms.

5.4.1 Coding for DGM Simple Cases

The data set `tankdat` from Lang and Bliese (2009) was used to illustrate variants of the DGM in Bliese and Lang (2016). Below we apply a subset of the R code from Appendix B of Bliese and Lang to illustrate basic form of the DGM.

```
> data(tankdat)

> tankdat$TRANS<-ifelse(tankdat$TIME<6,0,1)
> tankdat$POST<-ifelse(tankdat$TIME>5,tankdat$TIME-6,0)
> tankdat$TIME.A<-ifelse(tankdat$TIME<5,tankdat$TIME,5)

> tankdat[1:12,c("TIME","TRANS","POST","TIME.A")]
   TIME TRANS POST TIME.A
1     0     0    0      0
2     1     0    0      1
3     2     0    0      2
4     3     0    0      3
5     4     0    0      4
6     5     0    0      5
7     6     1    0      5
8     7     1    1      5
9     8     1    2      5
10    9     1    3      5
11   10     1    4      5
12   11     1    5      5
```

TRANS represents a dummy-coded variable that is zero before the event and one after the event. POST is slightly more complex in that it begins with a zero and then begins recounting (starting with zero) after the event occurs. TIME.A begins similarly to TIME, but holds the pre-transition element (5 in this case) constant once the change has occurred.

Below the basic DGM mixed-effect model is estimated and used to illustrate the difference between TIME and TIME.A.

```
> tmod<-lme(SCORE~TIME+TRANS+POST, random=~1|ID,tankdat)
> round(summary(tmod)$tTable,dig=3)
             Value Std.Error   DF t-value p-value
(Intercept) -3.686     0.631 2021  -5.837       0
TIME         1.814     0.125 2021  14.461       0
TRANS       -4.980     0.619 2021  -8.049       0
POST        -1.220     0.177 2021  -6.880       0

> tmod.a<-lme(SCORE~TIME.A+TRANS+POST, random=~1|ID,tankdat)
> round(summary(tmod.a)$tTable,dig=3)
             Value Std.Error   DF t-value p-value
(Intercept) -3.686     0.631 2021  -5.837       0
TIME.A       1.814     0.125 2021  14.461       0
TRANS       -3.166     0.537 2021  -5.895       0
POST         0.593     0.125 2021   4.732       0
```

Notice that TIME and TIME.A have the same parameter estimate and standard errors and both indicate that the performance score increased by 1.81 each trial. In the top model (TIME), the parameter estimate for TRANS is -4.98 and the POST estimate is -1.22 (both are significant). When using TIME, both TRANS and POST represent change relative to TIME, so the decline of -4.98 assumes this time period would have increased by 1.81. Likewise, the POST slope of -1.22 indicates a slope that is 1.22 less steep than the 1.81 increase associated with TIME.

The parameters associated with TIME.A are absolute, so in the lower model the value of -3.17 represents the absolute change (relative to zero) in performance. Likewise, the now positive slope of 0.59 indicates that while the recovery slope is significantly less strong than the pre-transition slope associated with TIME, the recovery slope is still significantly positive.

The DGM model, like the growth model, can be examined in a series of steps examining person-level variability in each parameter and including predictors of this variability. Interested readers are directed to Bliese and Lang (2106) and Bliese, Kautz, and Lang (2020) for additional details. Several examples using the DGM include Kim and Ployhart, (2014); Li, Hausknecht and Dragoni (2020); Pagiavlas, et al., (2021) and Rupp et al., 2009; Stewart et al., (2017).

5.4.2 Coding for DGM Complex Cases (`dgm.code`)

In cases such as with the tank data from Lang and Bliese (2009), the coding of the time-varying parameters is simple. In many applied settings, however, the coding can be more complicated for three reasons. First the longitudinal or panel data may be unbalanced such that each higher-level entity has a different number of repeated measures. Second, the event of interest may occur at different time points for each entity. Third, entities might not have the same number of events or any events at all.

For instance, a study of the impact of employee turnover on store performance might have panel data with thousands of stores providing quarterly data for 2 years. In each quarter, turnover may or may not have occurred, so each store has a unique pattern of turnover. Attempting to code the DGM time-varying covariates on a store-by-store basis would be challenging and time consuming.

The `dgm.code` function was designed to produce a design matrix for cases where events occur on an irregular basis and/or where entities have different number of observations. Details on the using dgm.code are in the help files, but below I reproduce one example.

```
> data(tankdat)
>
> # Add a marker (1 or 0) indicating an event at random
> set.seed(343227)
> tankdat$taskchange<-rbinom(nrow(tankdat),1,prob=.1)
> tankdat[1:24,]
   ID    CONSC TIME SCORE taskchange
1   1 1.041923    0    -5          0
2   1 1.041923    1     0          0
3   1 1.041923    2    -3          0
4   1 1.041923    3    -9          0
5   1 1.041923    4    -7          0
6   1 1.041923    5    -3          0
7   1 1.041923    6    -7          0
8   1 1.041923    7    -3          0
9   1 1.041923    8   -11          0
10  1 1.041923    9    -5          0
11  1 1.041923   10    -1          1
12  1 1.041923   11    -4          0

13  2 1.426890    0     3          0
14  2 1.426890    1    17          1
15  2 1.426890    2    18          0
16  2 1.426890    3    10          0
17  2 1.426890    4    22          1
18  2 1.426890    5    14          0
19  2 1.426890    6    -3          1
20  2 1.426890    7     6          0
21  2 1.426890    8    10          0
22  2 1.426890    9    15          0
23  2 1.426890   10    14          0
24  2 1.426890   11     7          0
```

In this example, the first individual (ID=1) had a taskchange at time 10 while the second individual (ID=2) had a task change at times, 1, 4, and 6. This example illustrates several issues. First, there are clearly different patterns of events. Second, it is not clear how events to code. An additional issue is that the event may occur on the first observation in which case the TRANS and POST time-varying vectors cannot be estimated. If we attempt to create the DGM design matrix we get the following error identifying groups that start with a taskchange (truncated output):

```
> OUT<-with(tankdat,dgm.code(ID,TIME,taskchange))
```

```
[1] "The following groups start with an event"
     grp time event
97     9    0     1
169   15    0     1
193   17    0     1
241   21    0     1
337   29    0     1
373   32    0     1
385   33    0     1
Truncated…
```

To handle both the issue of multiple events and an event starting on the first occasion, the `dgm.code` function contains two control options. By setting first.obs=TRUE we can recode the first observation to zero keeping a marker for whether we made this change. By setting n.events we can limit the design matrix to code only the first few events. For instance, to code only two events and recode the first event to a zero the command would be:

```
> OUT<-with(tankdat,dgm.code(ID,TIME,taskchange,n.events=2,first.obs=TRUE))
> OUT[1:24,]
   grp time event trans1 trans2 post1 post2 time.a tot.events event.first
1    1    0     0      0      0     0     0      0          1           0
2    1    1     0      0      0     0     0      1          1           0
3    1    2     0      0      0     0     0      2          1           0
4    1    3     0      0      0     0     0      3          1           0
5    1    4     0      0      0     0     0      4          1           0
6    1    5     0      0      0     0     0      5          1           0
7    1    6     0      0      0     0     0      6          1           0
8    1    7     0      0      0     0     0      7          1           0
9    1    8     0      0      0     0     0      8          1           0
10   1    9     0      0      0     0     0      9          1           0
11   1   10     1      1      0     0     0      9          1           0
12   1   11     0      1      0     1     0      9          1           0
13   2    0     0      0      0     0     0      0          3           0
14   2    1     1      1      0     0     0      0          3           0
15   2    2     0      1      0     1     0      0          3           0
16   2    3     0      1      0     2     0      0          3           0
17   2    4     1      0      1     0     0      0          3           0
18   2    5     0      0      1     0     1      0          3           0
19   2    6     1      0      1     0     2      0          3           0
20   2    7     0      0      1     0     3      0          3           0
21   2    8     0      0      1     0     4      0          3           0
22   2    9     0      0      1     0     5      0          3           0
23   2   10     0      0      1     0     6      0          3           0
24   2   11     0      0      1     0     7      0          3           0
```

The output returns a time, time.a, trans1, trans2, post1 and post2 to model the design matrix for two events. It also records the total events for each entity (tot.events) and indicates whether the first observation was an event.

Finally, to make use of this design matrix, it would need to be merged with the original data and reordered as follows:

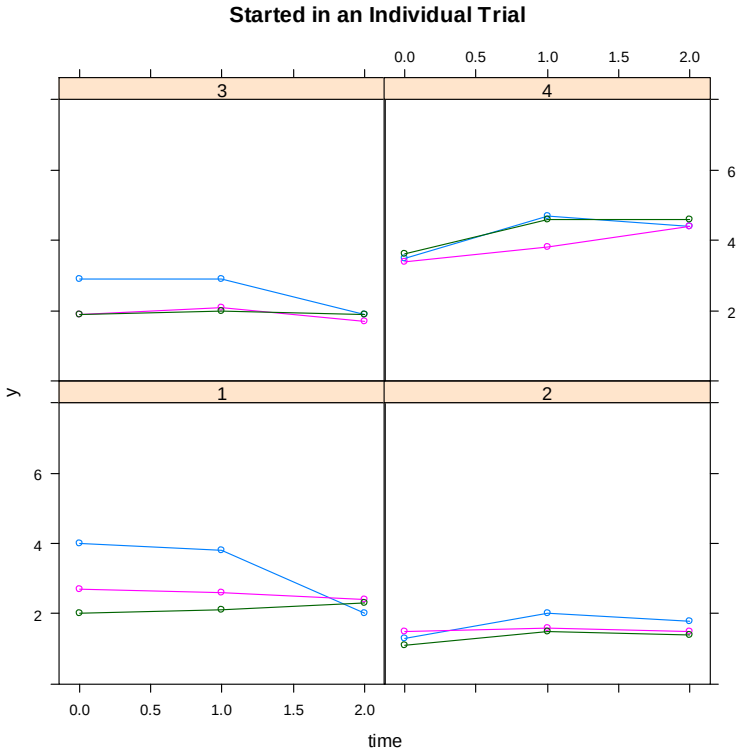
```
> tankdat.dgm<-merge(tankdat,OUT,by.x=c("ID","TIME"),by.y=c("grp","time"))
> tankdat.dgm<-tankdat.dgm[order(tankdat.dgm$ID,tankdat.dgm$TIME),]
```

5.5 Testing Emergence by Examining Error Structure

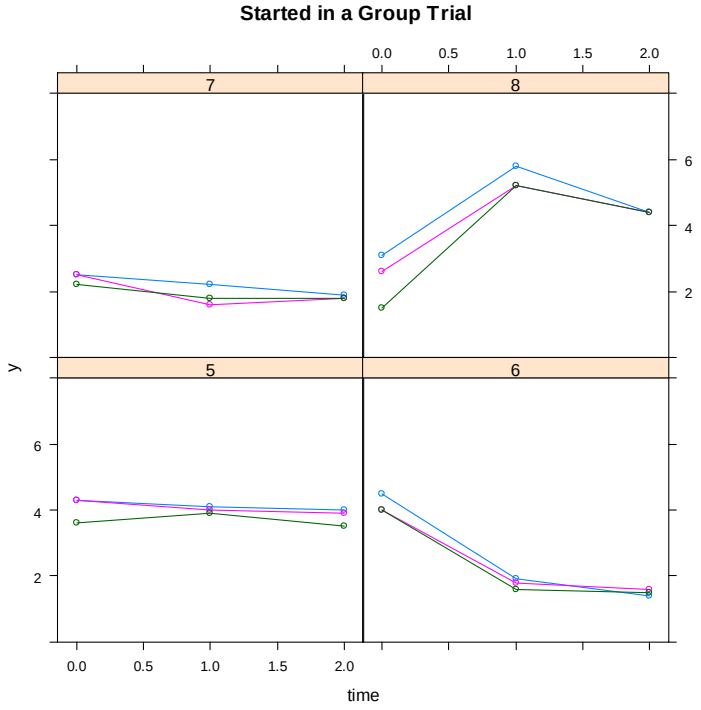
In most treatments of growth models heteroscedasticity in error structures are considered a form of model miss-specification that should be controlled (see section 5.3.4). Variants of mixed-effects models, however, have been suggested as a tool to formally test whether patterns of change in residual error variance over time have substantive meaning (Lang & Bliese, 2019; Lang et al., 2018; Lang et al., 2019).

For instance, consider the patterns displayed by participants over time in Sherif's (1935) classic experiment on group influence. In the experimental paradigm participants estimated movement of a small light (in inches) in a completely dark room. Participants either made initial estimates alone or with other group members and Sherif provided a plot of the results over three group-based trails. The data set `sherifdat` contains the values presented in Sherif's plot. The first set of figures below present the pattern for participants who began making estimates alone (and then transitioned to three trails where they made estimates with other group members). The second set of figures presents the pattern for participants who began making estimates with other group members over three trials.

```
> data(sherifdat)
> library(lattice)
> xyplot(y~time|as.factor(group),sherifdat[sherifdat$condition==1,],
  groups=person,type=c("p","l"),ylim=c(0,8),
  main="Started in an Individual Trial")
```



```
> xyplot(y~time|as.factor(group),sherifdat[sherifdat$condition==0,],
  groups=person,type=c("p","l"),ylim=c(0,8),
  main="Started in a Group Trial")
```



In both cases (either starting as an individual or starting in a group setting), the plots suggest that group members influence each other such that consensus emerges. The idea of consensus emergence appears stronger in cases where individuals started their first trial as an individual, but both conditions appear to show this effect. Lang and Bliese (2019) and Lang et al. (2018) provide details on how a three-level mixed-effect model (the census emergence model or CEM) can be estimated and how the -2log likelihood values can be contrasted to formally test whether emergence is present. Details are beyond the scope of this manual, but the basic formal test of emergence is provided below:

```
> threelevel<-lme(y ~ time,
  random = list(group=pdLogChol(~time),person=pdIdent(~1)),
  data=sherifdat,control=lmeControl(opt="optim",maxIter=3000,
  msMaxIter=3000))

> threelevelCEM<-update(threelevel,weights=varExp( form = ~ time))

> anova(threelevel,threelevelCEM)
              Model df      AIC      BIC    logLik   Test  L.Ratio p-value
threelevel        1  7 182.3422 198.0817 -84.17112
threelevelCEM     2  8 155.8097 173.7977 -69.90485 1 vs 2 28.53253  <.0001
```

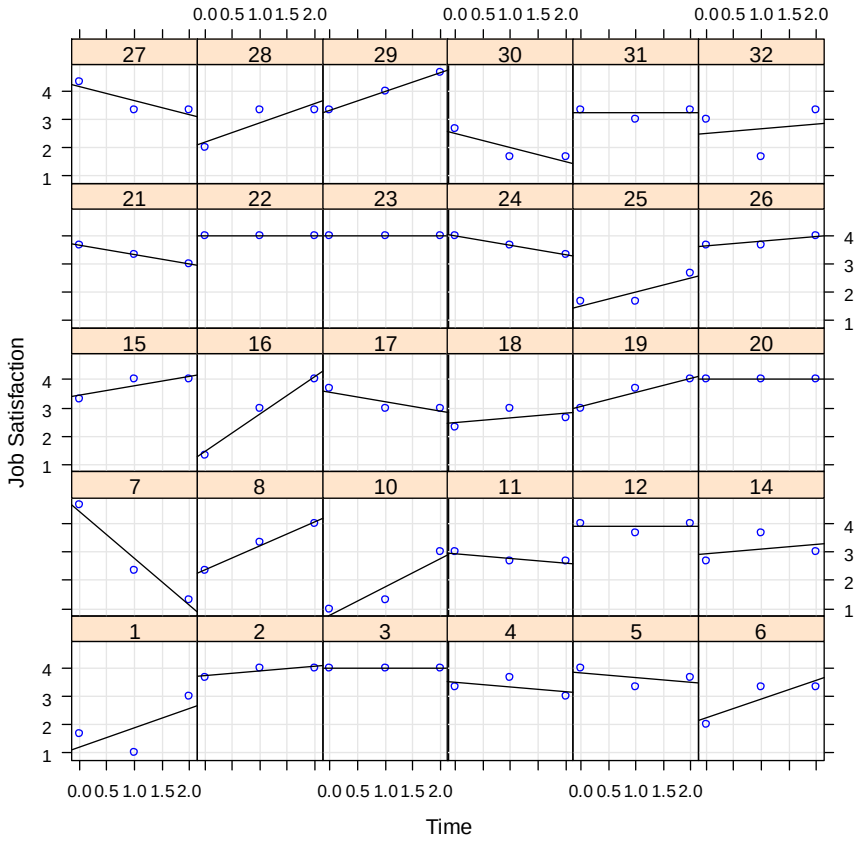
In both models, the random statement is a complex form of a three-level model that allows the slope for each group to randomly vary while fixing the time slope for individuals. A summary of the model `threelevelCEM` (not shown) provides the estimate for varExp as -1.017 indicating

an overall reduction in residual variance within groups (emergence). Including a variance term leads to a significant improvement in model fit suggesting that a significant emergence effect exists. Finally, while not demonstrated here, the models can be modified to formally test whether the emergence effect is stronger under the two conditions of starting individually or in a group.

5.6 Empirical Bayes estimates

While briefly introduced previously, one of the useful aspects of examining repeated measures in mixed-effects models is the ability to estimate predicted intercepts and slopes for individuals using (a) information about the individual along with (b) information from the rest of the sample. For instance, consider the growth modeling data presented in section 5.3. In this example, we modify the data so that only those with responses at all three times are included. We do so only to show that OLS-based estimates and empirical Bayes estimate differ even when data are complete.

```
> data(univbct)
> TEMP<-univbct[3*1:495,c(22,1:17)]  #convert to multivariate form
> TEMP<-na.exclude(TEMP[,c("SUBNUM","JOBSAT1","JOBSAT2","JOBSAT3")])
> TEMP.UNIV<-make.univ(TEMP,TEMP[,2:4],outname="JSAT")

> library(lattice)
> xyplot(JSAT~TIME|as.factor(SUBNUM),data=TEMP.UNIV[1:90,],
    type=c("p","r","g"),col="blue",col.line="black",
    xlab="Time",ylab="Job Satisfaction")
```



The figure shows large differences in intercepts and in slopes, yet each panel is estimated separately without taking into consideration any of the data from other respondents. An alternative would be to estimate a simple growth model and use data from model parameters to estimate values for each individual.

```
>tmod<-lme(JSAT~TIME,random=~TIME|SUBNUM, TEMP.UNIV,
    na.action=na.omit,control=list(opt="optim"))
```

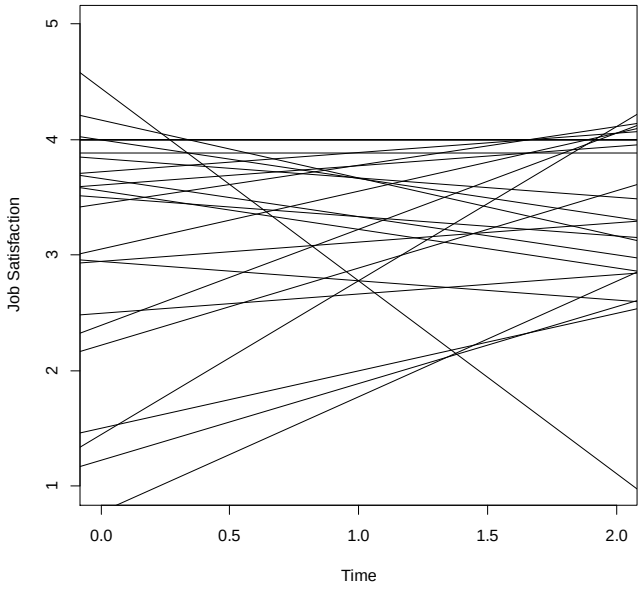
From this model, one can extract the empirical Bayes estimates for both the intercept and the slope by using the `coef` function: the first 12 values (bottom two rows) are listed.

```
> coef(tmod)[1:12,]
   (Intercept)         TIME
1     1.771548  0.358222009
2     3.701752  0.069173239
3     3.868707 -0.002492476
4     3.368637 -0.039600872
5     3.654505 -0.054411154
6     2.629151  0.313791178
7     3.537183 -0.615478500
8     2.843353  0.365710056
10    1.532927  0.496616898
11    2.892191 -0.014917079
12    3.773418  0.002444280
14    3.034727  0.103730558
```

The empirical Bayes estimates returned from `coef` correspond to what is displayed in the lattice plot. Individual 1, for instance, has a low value for satisfaction and a positive slope and individual 7 has a moderately high value and a strong negative slope.

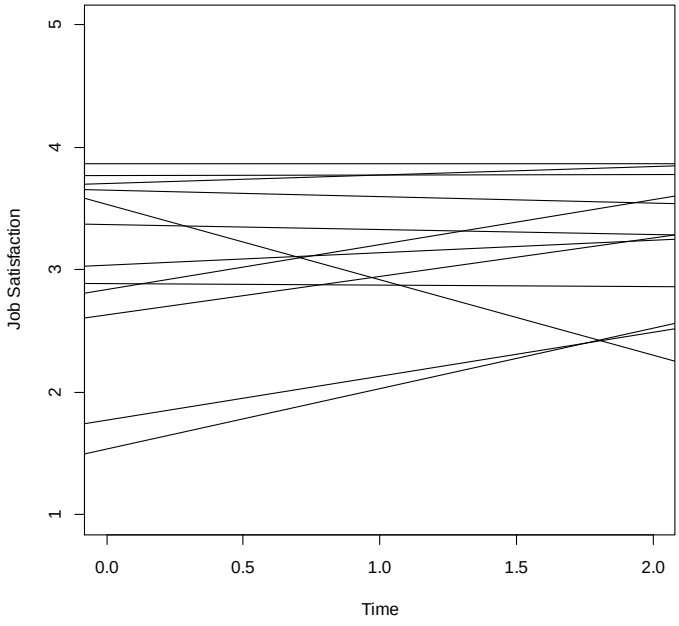
The differences can be more easily visualized by plotting all 30 individuals on a single plot. The plot represents the intercept and slope estimates from 30 separate linear regression equations.

```
>tmod3<-lmList(JSAT~TIME|SUBNUM, data=TEMP.UNIV[1:90,])

>plot(TEMP.UNIV$TIME,TEMP.UNIV$JSAT, xlab="Time",
     ylab="Job Satisfaction",type="n")

>lmplot<-function(X){
   for (i in 1:25){
   abline(X[[i]])
 }}

>lmplot(tmod3)
```



The second plot is for the same 30 individuals, but is based off of the empirical Bayes estimates.

```
>plot(TEMP.UNIV$TIME,TEMP.UNIV$JSAT, xlab="Time",
     ylab="Job Satisfaction",type="n")
>apply(coef(tmod)[1:12,],1,abline)
```



The fact that each individual's estimates are partially based on information from the rest of the sample adjusts some of the more extreme response (and explains why these are sometimes referred to as shrunken estimates). Empirical Bayes estimates may be particularly useful in situations where intercepts and slopes are used to predict other outcomes. For instance, Chen, Ployhart, Thomas, Anderson, & Bliese (2011) used empirical Bayes estimates of slope changes in job satisfaction and showed that the nature of the change (increase or decrease) was the primary predictor of turnover intentions.

It may go without saying, but one can also extract empirical Bayes estimates from non-longitudinal nested models such as those considered in section 4. In the context of non-longitudinal models, the values provides estimates of intercepts and slopes for each group adjusted for the overall intercept and slope. As a general rule, when ICC(1) values are small, the empirical Bayes estimates are more strongly adjusted to the rest of the sample (more shrinkage) than when ICC(1) values are large (see Gelman & Pardoe, 2006).

6 More on `lme4`

While the current document has focused on the `nlme` package for mixed-effects models, the `lme4` package in R provides additional flexibility in terms of specifying models. The `lme4` package is particularly valuable in dealing with (a) non-normally distributed outcomes and (b) partially crossed data structures.

6.1 Dichotomous outcomes

When the dependent variable is dichotomous or otherwise non-normally distributed, it may be useful to estimate a generalized linear mixed effects model (glmm) rather than a linear mixed effects model. Below we dichotomize WBEING and use `glmer` from the `lme4` package with a binomial link function to estimate a mixed-effects logistic regression model.

```
>library(multilevel)
>library(lme4)
>data(bh1996)
>tmod<-glmer(ifelse(WBEING>3.5,1,0)~HRS+G.HRS+(1|GRP),
       family="binomial",control=glmerControl(optimizer="bobyqa"),bh1996)

>summary(tmod)

Generalized linear mixed model fit by maximum likelihood (Laplace
  Approximation) [glmerMod]
 Family: binomial  ( logit )
Formula: ifelse(WBEING > 3.5, 1, 0) ~ HRS + G.HRS + (1 | GRP)
   Data: bh1996
Control: glmerControl(optimizer = "bobyqa")

     AIC      BIC   logLik deviance df.resid
  7572.1   7599.7  -3782.0   7564.1     7378

Scaled residuals:
    Min      1Q  Median      3Q     Max
-0.9902 -0.5559 -0.4672 -0.3587  4.6130

Random effects:
```

```
 Groups Name        Variance Std.Dev.
 GRP    (Intercept) 0.06323  0.2515
Number of obs: 7382, groups:  GRP, 99

Fixed effects:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  2.80660    0.53504   5.246 1.56e-07 ***
HRS         -0.09860    0.01465  -6.731 1.69e-11 ***
G.HRS       -0.26784    0.04923  -5.440 5.31e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Correlation of Fixed Effects:
      (Intr) HRS
HRS   -0.020
G.HRS -0.954 -0.272
```

The precision of the model in terms of log likelihood can be improved by including the `nAGQ` option with a value greater than 1 (100 in this case). Notice the slight change in log likelihood values and the minor changes in parameter estimates and standard errors between the model based on `nAGQ=1` (above) and `nAGQ=25` (below). In practice, one would likely want to change `nAGQ` values to (a) verify parameter estimates and standard errors and (b) verity that contrasts of -2log likelihood values contrasting models with `anova` are similar with higher `nAGQ` values. In my experience using values above 100 is rarely useful.

```
> tmod.r<-glmer(ifelse(WBEING>3.5,1,0)~HRS+G.HRS+(1|GRP),
  family="binomial", control=glmerControl(optimizer="bobyqa"),
  bh1996,nAGQ=25)

> logLik(tmod) # Original model with nAGQ=1
'log Lik.' -3782.036 (df=4)

> logLik(tmod.r) # Model with nAGQ = 25
'log Lik.' -3781.999 (df=4)

> summary(tmod.r)$coef
               Estimate Std. Error   z value     Pr(>|z|)
(Intercept)  2.80640657 0.53692297  5.226833 1.724383e-07
HRS         -0.09861117 0.01466700 -6.723335 1.776112e-11
G.HRS       -0.26782094 0.04939543 -5.421978 5.894300e-08
```

6.2 Crossed and partially crossed models

The second situation in which `lme4` is particularly valuable is in cases where data are partially or fully crossed. For instance, in a longitudinal study individuals might be nested within groups, but over time some individuals might switch from one group to another. If no participants switched groups, the data would be fully nested with repeated observations nested within individuals nested within groups (a three-level model). In `lme` the three-level nested model would be specified as `random= ~1|GRP/IND`. If individuals switch groups, though, the fully nested structure no longer holds. In `lme4` and the `lmer` function, however, the structure could be specified as `(1|GRP)+(1|IND)`. The `lmer` specification does not assume fully nested data and will provide variance estimates if the data are partially crossed.

6.3 Predicting values in `lme4`

As illustrated in the text, statistical models can be used to predict levels of an outcome variable given specific values of predictors. R has a number of `predict` functions linked to specific models (e.g., `lm, glm, lme, lmer, glmer`). The `predict` functions are generally consistent in terms of usage; however, there are minor differences when applied to specific models. Recall, for instance, that one must specify `level=0` to obtain overall sample based predictions when using `lme`.

In most cases in mixed-effects models, one will be interested in obtaining predictions for the overall sample rather than predictions for any specific unit; however, in the `lmer` and `glmer` functions associated with lme4, the predict command uses the option `re.form=NA` rather than `level=0` to indicate that predictions should be made based on the parameter estimates from the overall sample. An example is provided below:

```
> library(multilevel)
> library(lme4)
> data(bh1996)

> tmod<-lmer(WBEING~HRS*LEAD+(1|GRP),bh1996)

> TDAT<-data.frame(HRS=c(7,7,12,12),LEAD=c(2.12,2.12,3.66,3.66))
> predict(tmod,TDAT,re.form=NA)
       1        2        3        4
2.519160 2.519160 3.137911 3.137911
```

As another example, the code below illustrates the use of the `type="response"` option with models that have a dichotomous variable as the outcome. Notice that one can transform the prediction to a percent (-2.377 to 0.085 or 8.5%), but it is often easier to use `type="response"`.

```
> tmod<-glmer(ifelse(WBEING>3.5,1,0)~LEAD+(1|GRP),family="binomial",bh1996,
  control=glmerControl(optimizer="bobyqa"))

> TDAT<-data.frame(LEAD=c(2.12,3.66))

> predict(tmod,TDAT,re.form=NA)
         1          2
-2.3774501 -0.6565601

> exp(-2.3774501)/(1+exp(-2.3774501))
[1] 0.08490848

> predict(tmod,TDAT,re.form=NA,type="response")
         1          2
0.08490848 0.34151277
```

7 Miscellaneous Functions and Tips

The multilevel package has a number of other functions that have either been referenced in appendices of published papers, or are of basic utility to applied organizational researchers. This section briefly describes these functions. Complete help files are available in the `multilevel` package for each of the functions discussed.

7.1 Scale reliability: `cronbach` and `item.total`

Two functions that are can be particularly useful in estimating the reliability of multi-item scales are the `cronbach` and the `item.total` functions. Both functions take a single argument, a dataframe with multiple columns where each column represents one item in a multi-item scale.

7.2 Random Group Resampling for OLS Regression Models

The function `rgr.OLS` allows one to contrast a group-level hierarchical regression model with an identically specified model where group identifiers are randomly generated. This type of model was estimated in Bliese and Halverson (2002).

7.3 Estimating bias in nested regression models: `simbias`

Bliese and Hanges (2004) showed that a failure to model the nested properties of data in ordinary least squares regression could lead to a loss of power in terms of detecting effects. The article provided the `simbias` function to help estimate the degree of power loss in complex situations.

7.4 Detecting mediation effects: `sobel`

MacKinnon, Lockwood, Hoffman, West and Sheets (2002) showed that many of the mediation tests used in psychology tend to have low power. One test that had reasonable power was Sobel's (1982) indirect test for mediation. The `sobel` function provides a simple way to run Sobel's (1982) test for mediation. Details on the use of the `sobel` function are available in the help files.

8 References

Bartko, J. J. (1976). On various intraclass correlation reliability coefficients. *Psychological Bulletin, 83,* 762-765.

Becker, R. A., Chambers, J. M., & Wilks, A. R. (1988). *The New S Language*. New York: Chapman & Hall.

Bliese, P. D. (1998). Group size, ICC values, and group-level correlations: A simulation. *Organizational Research Methods, 1*, 355-373.

Bliese, P. D. (2000). Within-group agreement, non-independence, and reliability: Implications for data aggregation and Analysis. In K. J. Klein & S. W. Kozlowski (Eds.), *Multilevel*

Theory, Research, and Methods in Organizations (pp. 349-381). San Francisco, CA: Jossey-Bass, Inc.

Bliese, P. D. (2002). Multilevel random coefficient modeling in organizational research: Examples using SAS and S-PLUS. In F. Drasgow & N. Schmitt (Eds.), *Modeling in Organizational Research: Measuring and Analyzing Behavior in Organizations* (pp. 401-445). San Francisco, CA: Jossey-Bass, Inc.

Bliese, P. D., & Britt, T. W. (2001). Social support, group consensus and stressor-strain relationships: Social context matters. *Journal of Organizational Behavior, 22*, 425-436.

Bliese, P. D. & Hanges, P. J. (2004). Being both too liberal and too conservative: The perils of treating grouped data as though they were independent. *Organizational Research Methods, 7*, 400-417.

Bliese, P. D. & Halverson, R. R. (1996). Individual and nomothetic models of job stress: An examination of work hours, cohesion, and well-being. *Journal of Applied Social Psychology, 26*, 1171-1189.

Bliese, P. D., & Halverson, R. R. (1998a). Group consensus and psychological well-being: A large field study. *Journal of Applied Social Psychology, 28*, 563-580.

Bliese, P. D., & Halverson, R. R. (1998b). Group size and measures of group-level properties: An examination of eta-squared and ICC values. *Journal of Management, 24*, 157-172.

Bliese, P. D., & Halverson, R. R. (2002). Using random group resampling in multilevel research. *Leadership Quarterly, 13*, 53-68.

Bliese, P. D., & Halverson, R.R. & Rothberg, J. (2000). Using random group resampling (RGR) to estimate within-group agreement with examples using the statistical language R. *Unpublished Manuscript*.

Bliese, P. D. & Jex, S. M. (2002). Incorporating a multilevel perspective into occupational stress research: Theoretical, methodological, and practical implications. *Journal of Occupational Health Psychology, 7*, 265-276.

Bliese, P. D., & Jex S. M. (1999). Incorporating multiple levels of analysis into occupational stress research. *Work and Stress, 13*, 1-6.

Bliese, P. D., Kautz, J., & Lang, J. W. (2020). Discontinuous growth models: Illustrations, recommendations, and an R function for generating the design matrix. In Y. Griep & S. D. Hansen (Eds.), *Handbook on the Temporal Dynamics of Organizational Behavior* (pp. 319–350). Northampton, MA: Edward Elgar Publishers, Inc. DOI: https://doi.org/10.4337/9781788974387

Bliese, P. D., & Lang, J. W. B. (2016). Understanding relative and absolute change in discontinuous growth models: Coding alternatives and implications for hypothesis testing. *Organizational Research Methods, 19,* 562-592.

Bliese, P. D., Maltarich, M. A., Hendricks, J. L., Hofmann, D. A., & Adler, A. B. (2019). Improving the measurement of group-level constructs by optimizing between-group differentiation. *Journal of Applied Psychology, 104,* 293-302.

Bliese, P. D., Maltarich, M. A., & Hendricks, J. L. (2018). Back to Basics with Mixed-Effects Models: Nine Take-Away Points. *Journal of Business and Psychology, 33,* 1-23.

Bliese, P. D., & Ployhart, R. E. (2002). Growth modeling using random coefficient models: Model building, testing and illustrations. *Organizational Research Methods, 5,* 362-387.

Brown, R. D., & Hauenstein, N. M. A. (2005). Interrater agreement reconsidered: An alternative to the rwg indices. *Organizational Research Methods, 8,* 165-184.

Bryk, A. S., & Raudenbush, S. W. (1992). *Hierarchical linear models*. Newbury Park, CA: Sage.

Burke, M. J., Finkelstein, L. M., & Dusig, M. S. (1999). On average deviation indices for estimating interrater agreement. *Organizational Research Methods, 2,* 49-68.

Campbell-Sills, L., Flynn, P. J., Choi, K. W., Ng, T. H., Aliaga, P. A., Broshek, C., Jain, S., Kessler, R. C., Stein, M. B., Ursano, R. J. & Bliese, P. D., (2022). Unit cohesion during deployment and post-deployment mental health: Is cohesion an individual- or unit-level buffer for combat-exposed soldiers? *Psychological Medicine, 52,* 121-131.

Chambers, J. M. & Hastie, T. J. (1992). *Statistical Models in S*. New York: Chapman & Hall.

Chen, G., Ployhart, R. E., Thomas, H. C., Anderson, N. & Bliese, P. D. (2011). The power of momentum: A new model of dynamic relationships between job satisfaction change and turnover intentions. *Academy of Management Journal, 54,* 159-181.

Clark, T. S., & Linzer, D. A. (2015). Should I use fixed or random effects? *Political Science Research and Methods, 3,* 399-408.

Cohen, J. & Cohen, P. (1983). *Applied multiple regression/correlation analysis for the behavioral sciences* (2nd Ed.). Hillsdale, NJ: Lawrence Erlbaum Associates.

Cohen, A., Doveh, E. & Eick, U. (2001). Statistical properties of the rwg(j) index of agreement. *Psychological Methods, 6,* 297-310.

Cohen, A., Doveh, E. & Nahum-Shani, I. (2009). Testing agreement for multi-item scales with the indices rwg(j) and ADM(J). *Organizational Research Methods, 12,* 148-164.

Cudeck, R., & Klebe, K. J. (2002). Multiphase mixed-effects models for repeated measures data. *Psychological Methods, 7*, 41–63.

Cummings, G. & Finch, S. (2005). Inference by eye: Confidence intervals and how to read pictures of data. *American Psychologist, 60*, 170-180.

Dansereau, F., Alutto, J. A., & Yammarino, F. J. (1984). *Theory testing in organizational behavior: The varient approach*. Englewood Cliffs, NJ: Prentice-Hall.

Dunlap, W. P., Burke, M. J., & Smith-Crowe, K. (2003). Accurate tests of statistical significance for rwg and average deviation interrater agreement indices. *Journal of Applied Psychology, 88*, 356-362.

Firebaugh, G. (1978). A rule for inferring individual-level relationships from aggregate data. *American Sociological Review, 43*, 557-572.

Gelman, A. & Pardoe, I. (2006). Bayesian measures of explained variance and pooling in multilevel (hierarchical) models. *Technometrics, 48*, 241-251.

Hernández-Lloreta, M. V., Colmenares, F., & Martínez-Arias, R. (2004). Application of piecewise hierarchical linear growth modeling to the study of continuity in behavioral development of baboons (Papio hamadryas). *Journal of Comparative Psychology, 118,* 316–324.

Hofmann, D. A. (1997). An overview of the logic and rationale of Hierarchical Linear Models. *Journal of Management, 23*, 723-744.

Hofmann, D. A. & Gavin, M. (1998). Centering decisions in hierarchical linear models: Theoretical and methodological implications for research in organizations. *Journal of Management, 24*, 623-641.

James, L. R. (1982). Aggregation bias in estimates of perceptual agreement. *Journal of Applied Psychology, 67*, 219-229.

James, L.R., Demaree, R.G., & Wolf, G. (1984). Estimating within-group interrater reliability with and without response bias. *Journal of Applied Psychology, 69*, 85-98.

James, L. R. & Williams, L. J. (2000). The cross-level operator in regression, ANCOVA, and contextual analysis. In K. J. Klein & S. W. Kozlowski (Eds.), *Multilevel Theory, Research, and Methods in Organizations* (pp. 382-424). San Francisco, CA: Jossey-Bass, Inc.

Hox, J. J. (2002). *Multilevel analysis: Techniques and applications*. Mahwah, NJ: Lawrence Erlbaum Associates.

Klein, K. J. & Kozlowski, S. W. J, (2000). *Multilevel theory, research, and methods in organizations*. San Francisco, CA: Jossey-Bass, Inc.

Klein, K. J., Bliese, P.D., Kozlowski, S. W. J, Dansereau, F., Gavin, M. B., Griffin, M. A., Hofmann, D. A., James, L. R., Yammarino, F. J. & Bligh, M. C. (2000). Multilevel analytical techniques: Commonalities, differences, and continuing questions. In K. J. Klein & S. W. Kozlowski (Eds.), *Multilevel Theory, Research, and Methods in Organizations* (pp. 512-553). San Francisco, CA: Jossey-Bass, Inc.

Kim, Y. & Ployhart, R. E., (2014). The effects of staffing and training on firm productivity and profit growth before, during, and after the great recession. *Journal of Applied Psychology, 99*, 361-389.

Kozlowski, S. W. J., & Hattrup, K. (1992). A disagreement about within-group agreement: Disentangling issues of consistency versus consensus. *Journal of Applied Psychology, 77*, 161-167.

Kreft, I. & De Leeuw, J. (1998). *Introducing multilevel modeling*. London: Sage Publications.

LaHuis, D. M., & Ferguson, M. W. (2009). The accuracy of significance tests for slope variance components in multilevel random coefficient models. *Organizational Research Methods, 12(3)*, 418-435.

Lang, J. W. B., & Bliese, P. D., (2019). A Temporal Perspective on Emergence: Using 3-level Mixed Effects Models to Track Consensus Emergence in Groups. In S. E. Humphrey & J. M. LeBreton (Eds.), *The Handbook for Multilevel Theory, Measurement, and Analysis*. Washington, DC: American Psychological Association.

Lang, J. W. B., & Bliese, P. D. (2009). General mental ability and two types of adaptation to unforeseen change: Applying discontinuous growth models to the task-change paradigm. *Journal of Applied Psychology, 92*, 411-428.

Lang, J. W. B., Bliese, P. D., & Adler, A. B. (2019). Opening the Black Box: A Multilevel Framework for Studying Group Processes. *Advances in Methods and Practices in Psychological Science, 2*, 271-287.

Lang, J. W. B., Bliese, P. D., & de Voogt, A. (2018). Modeling Consensus Emergence in Groups Using Longitudinal Multilevel Methods. *Personnel Psychology, 71*, 255-281.

LeBreton, J. M., James, L. R. & Lindell, M. K. (2005). Recent issues regarding r_{WG}, r^*_{WG}, $r_{WG(J)}$, and $r^*_{WG(J)}$. *Organizational Research Methods, 8*, 128-138.

LeBreton, J. M., & Senter, J. L. (2008). Answers to 20 questions about interrater reliability and interrater agreement. *Organizational research methods, 11(4)*, 815-852.

Levin, J. R. (1967). Comment: Misinterpreting the significance of "explained variation." *American Psychologist, 22,* 675-676.

Li, H., Hausknecht, J. P., & Dragoni, L. (2020). Initial and Longer-Term Change in Unit-Level Turnover Following Leader Succession: Contingent Effects of Outgoing and Incoming Leader Characteristics. *Organization Science, 31(2),* 458-476.

Lindell, M. K. & Brandt, C. J. (1997). Measuring interrater agreement for ratings of a single target. *Applied Psychological Measurement, 21,* 271-278.

Lindell, M. K. & Brandt, C. J. (1999). Assessing interrater agreement on the job relevance of a test: A comparison of CVI, T, rWG(J), and r*WG(J) indexes. *Journal of Applied Psychology, 84,* 640-647.

Lindell, M. K. & Brandt, C. J. (2000). Climate quality and climate consensus as mediators of the relationship between organizational antecedents and outcomes. *Journal of Applied Psychology, 85,* 331-348.

Lindell, M. K., Brandt, C. J. & Whitney, D. J. (1999). A revised index of interrater agreement for multi-item ratings of a single target. *Applied Psychological Measurement, 23,* 127-135.

Lüdtke, O., & Robitzsch, A. (2009). Assessing within-group agreement: A critical examination of a random-group resampling approach. *Organizational Research Methods, 12,* 461-487.

MacKinnon, D. P., Lockwood, C. M., Hoffman, J. M., West, S. G., Sheets, V. (2002). A comparison of methods to test mediation and other intervening variable effects. *Psychological Methods, 7,* 83-104.

Pagiavlas, S., Kalaignanam, K., Gill, M., & Bliese, P. D. (2021). EXPRESS: Regulating Product Recall Compliance in the Digital Age: Evidence from the "Safe Cars Save Lives" Campaign. *Journal of Marketing,* DOI: 00222429211023016.

Pinheiro, J. C. & Bates, D. M. (2000). *Mixed-effects models in S and S-PLUS*. New York: Springer-Verlag.

Ployhart, R. E., Holtz, B. C. & Bliese, P. D. (2002). Longitudinal data analysis: Applications of random coefficient modeling to leadership research. *Leadership Quarterly, 13,* 455-486.

Raudenbush, S. W., & Bryk, A. S. (2002). *Hierarchical linear models: Applications and data analysis methods* (Vol. 1). Sage.

Robinson, W. S. (1950). Ecological correlations and the behavior of individuals. *American Sociological Review, 15,* 351-357.

Rupp, T. L., Wesensten, N. J., Bliese, P. D., & Balkin, T. J. (2009). Banking sleep: Realization of benefits during subsequent sleep restriction and recovery. *Sleep, 32*, 311-321.

Sherif, M. (1935). A study of some social factors in perception: Chapter 3. *Archives of Psychology, 27*, 23- 46.

Shrout, P. E., & Fleiss, J. L. (1979). Intraclass correlations: Uses in assessing rater reliability. *Psychological Bulletin, 86*, 420-428.

Singer, J. D. (1998). Using SAS PROC MIXED to fit multilevel models, hierarchical models, and individual growth models. *Journal of Educational and Behavioral Statistics, 24*, 323-355.

Snijders, T. A. B. & Bosker, R. J. (1994). Modeled variance in two-level models. *Sociological Methods and Research, 22*, 342-363.

Snijders, T. A. B. & Bosker, R. J. (1999). *Multilevel analysis: An introduction to basic and advanced multilevel modeling*. London: Sage Publications.

Sobel, M. E., (1982). Asymptotic confidence intervals for indirect effects in structural equation models. In S. Leinhardt (Ed.), *Sociological Methodology 1982* (pp. 290-312). Washington, DC: American Sociological Association.

Stewart, G. L., Astrove, S. L., Reeves, C. J. , Crawford, E. R., & Solimeo, S. L. (2017). Those with the Most Find It Hardest to Share: Exploring Leader Resistance to the Implementation of Team-based Empowerment. *Academy of Management Journal, 60*, 2266–2293.

Tinsley, H. E. A., & Weiss, D. J. (1975). Interrater reliability and agreement of subjective judgements. *Journal of Counseling Psychology, 22*, 358-376.