¦ 2014 ■ vol. 10 ■ no. 1

DOI: 10.20982/tqmp.10.1.p056

Representing Error bars in within-subject designs in typical software packages

Fearghal O'Brien [a], Denis Cousineau [b]

[a] School of psychology, Trinity College Dublin, Ireland

[b] École de psychologie, Université d'Ottawa, Canada

Abstract ■ The addition of error bars in graphs can greatly assist readers in determining where differences lie between groups or conditions. The use of error bars to represent the standard error of the mean or confidence intervals is relatively straightforward for between-subject (independent group) designs. However, confusion still abounds as to how best to graph these same measures for within-subject (repeated-measures) designs. This paper serves both to consolidate previous recommendations regarding error bars for within-subject studies and to provide clear instructions as to how to implement these recommendations using typical statistical packages.

Keywords ■ Error bars, within-subject designs; SPSS; R; Mathematica

obrienfk@tcd.ie

Introduction

The use of error bars to represent standard errors or confidence intervals is becoming increasingly popular in psychological publications, and indeed publications in many other disciplines. Many journal editors now insist that all graphs, where appropriate, should include error bars (e.g. Journal of Biological Chemistry). The APA task force on statistical inference states that "In all figures, include graphical representations of interval estimates whenever possible" (Wilkinson and the task force on statistical inference, 1999, p. 601). The graphical representation of data has always played a vital role in the exploration and communication of results. Typical graphs (such as bar and line graphs) represent the mean of a number of different groups and/or conditions. Error bars help to communicate the size of the difference between groups and to indicate instances where statistically significant differences exist. The discussion below will be on the use of error bars to represent standard errors of the mean (simply referred to as standard errors, SE, henceforth) though all of the points made also apply to the representation of confidence intervals. Loftus and Masson (1994; also see Baguley, 2012) provide an excellent introduction to the issue of error bars in a within-subject design and so only a brief description will be given below.

For between-subject designs, the inclusion of error bars is a relatively simple affair. To take a hypothetical example, imagine we measured the reaction times of 10 participants responding (e.g. pressing a button) to one-syllable words and 10 participants responding to two-syllable words. Figure 1 (left) displays the average RT of each of the 20 participants from this hypothetical study. The left panel displays each participant's average reaction time and a line connecting the two groups. While it is reasonably clear that the means are not equal, the extent to which they differ is difficult to assess. Error bars, representing either the standard error of the mean or confidence intervals, can aid in this interpretation.

If we wish to represent the standard error using error bars, then all that is needed is the pooled standard deviation of the groups, s_p, so that standard error (SE) is given by the equation

$$SE = \sqrt{\frac{2s_p^2}{n}}$$

where s_p^2 represents the pooled variance of the groups (the average variance of the groups weighted by the degrees of freedom) and n is the number of participants in each group (here assumed equal for simplicity). Figure 1 (right) show the mean of each group with error bars representing the standard error. A rule of thumb for error bars representing standard errors (if the groups have at least 10 participants in each) is that if there is a gap of more than one error bar (e.g. one SE) between the groups, then there is a significant difference at the $p < .05$ level (Cumming & Finch,

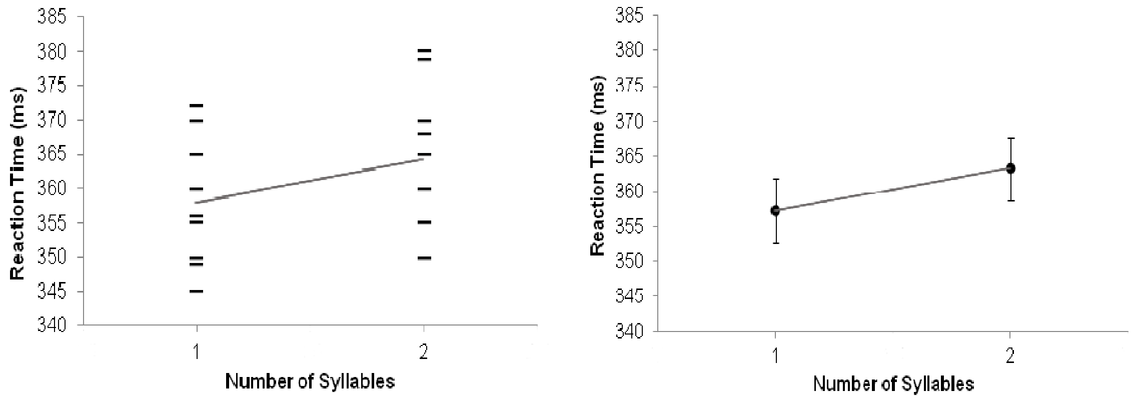

Figure 1 ■ Reaction time to one and two syllables words in a between-subject design. The graph on the left shows the individual scores for all 20 participants while the graph on the right displays the mean of the two groups and error bars representing the SE.

2005).[1] In Figure 1(right), there is no space between the error bars suggesting the groups are not significantly different. This observation is confirmed by formal statistical tests, $t(18) = 1.3$, $p = .21$.

Imagine that the same study was carried out in a within-subject design with a total of ten participants reacting to both one- and two-syllable words. Fig. 2 shows the hypothetical results which are identical to those from the previous example. The difference in this case is that each participant has a set of two responses and these are connected by grey lines in Figure 2 (left). What should be clear is that for nearly all the participants, their reaction times are larger for the two-syllable condition. A repeated-measures t-test reveals that the conditions do differ significantly, $t(9) = 4.8$, $p < .01$. However, when it comes to graphically representing the information below with means and error bars, it is not clear what standard error the bars should represent. If they represent the pooled standard error across the two conditions, then the error bars will be exactly as they are in Figure 1(right) above. This would lead to a situation where the graph is suggesting non-significance while the t test finds that the conditions are statistically different.

Loftus and Masson (1994) explain precisely why this problem arises. Within-subject statistical comparisons are based on the idea of controlling for the variance between individuals and focusing only on the difference between their scores in each condition. If error bars are based on the pooled standard error within the groups, then they will not be controlling for the variance between the participants. Loftus and Masson (1994) propose an alternative to calculate a more appropriate standard error by carrying out a repeated-measures ANOVA and using the formula

$$SE = \sqrt{\frac{MS_e}{n}}$$

where MS_e is the mean square error of the model and n in the number of participants.

Figure 2 (right) shows the two conditions with these adjusted error bars which now, correctly, indicate a significant difference between the groups. However, as Cousineau (2005) points out, this method is not without its problems. Firstly, it slightly defeats the purpose of visually exploring the data if one has to carry out the entire analysis in order to create the graph. Furthermore, relying on the MS_e of the overall repeated-measures ANOVA means that the error bars will be the same for all conditions. This assumes that there is no interaction between participant and condition which is not always the case. Cousineau (2005) proposes a simple solution in which the data is

[1] Briefly, a t-test on two groups is given by the t statistic, $t = \left|\overline{X}_1 - \overline{X}_2\right| / (\sqrt{2} s_p / \sqrt{n})$, in which the numerator is the separation between the two groups' mean, and s_p is the pooled standard deviation. If the groups are separated by a distance totalizing 3 SE, then, $\left|\overline{X}_1 - \overline{X}_2\right| / (\sqrt{2} s_p / \sqrt{n})$ reduces to $(3SE)/(\sqrt{2}SE) = 3/\sqrt{2} \approx 2.12$. On the other hand, the critical value at 5% for a t-test with 9 degrees of freedom is 2.09. Hence, with three standard errors between the means, the t statistic is just above signification.

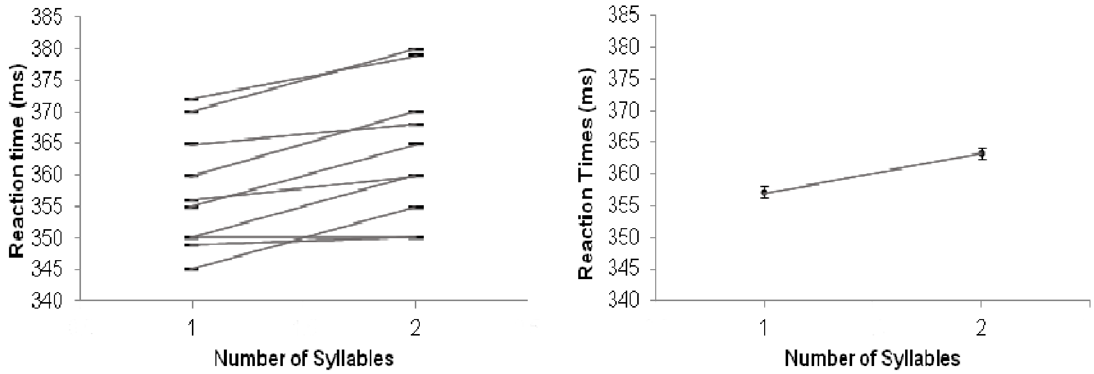

Figure 2 ■ Reaction time to one and two-syllables words in a within-subject design. The graph on the left shows two pairs of scores for all 10 participants while the graph on the right displays the mean of the two conditions. The error bars represent the SE following Cousineau-Morey corrections.

standardized across participants [2]and the standard error is taken from this standardized data. The specific method of standardizing the data is mathematically trivial and is discussed in greater detail in Cousineau's original paper which also provides syntax for SPSS to standardize the data. However, Morey (2008) demonstrated mathematically that standard errors of standardized data are smaller than the true population standard errors. This bias is most important in 2-condition designs and decreases as the number of conditions increases. Morey proposes a simple solution to the problem and that is to adjust the standard errors of normalized data using the correction factor (cf):

$$cf = \sqrt{\frac{\prod_{i=1}^{p} M_i}{(\prod_{i=1}^{p} M_i) - 1}}$$

where p is the number of fixed factors and M_i is the number of levels of the ith within-subject factor. In other words, $\prod_{i=1}^{p} M_i$ is the total number of measures per subject. For example, in a one-factor ($p = 1$), within-subject design with two levels, the correction factor is

$$cf = \sqrt{\frac{2}{2-1}} = \sqrt{2}$$

Hence, the error bars should be increased by 41% ($\sqrt{2} \approx 1.41$) in a 2-measure design. The correction factor becomes smaller as the number of conditions increases. If there were 2 fixed factors, each with 2 levels, then the correction factor would be

$$cf = \sqrt{\frac{2 \times 2}{2 \times 2 - 1}} = \frac{2}{\sqrt{3}} \approx 1.15$$

While Cousineau (2005) provides a simple SPSS syntax to implement the error bars, there are a number of problems with the approach. The most important is that the original syntax does not incorporate the Morey correction which unfortunately must be calculated each time a standard error is required. A second problem with Cousineau's (2005) original syntax is that it requires the SPSS dataset to be organized so that only one variable codes all of the scores across all of the conditions, another variable identifies participants and yet another variable identifies the condition (long-format data). This is an unusual layout for an SPSS dataset where data is generally stored so that each condition occupies its own column (wide-format data). While a piece of syntax can transform a long-format to wide-format, this does require a certain degree of competency with SPSS syntax (see for example Giguère and Lacroix, 2005).

This leads onto the final point. Even if Morey's correction wasn't necessary and even if our file was already set up in the way Cousineau's syntax requires, researchers would still need to use the basic syntax of SPSS. Command line languages (such as R) are much more powerful and flexible than their Graphical User

[2] Volker and Loftus, 2012, called it a method of normalization, but standardization seems more appropriate and this is the expression used herein.

Interface (GUI) counterparts. However, the main advantage of GUI approaches is that they are (relatively) easy for users to interact with. The present paper will describe a package for SPSS (called a "bundle" in this software) which, when installed, will allow users to create normalized data with the Morey adjustments without using any syntax.

Incorporating the correction factor in the data

One way to incorporate the correction factor is to compute the standard errors using standardized data and multiply these standard errors by the *cf.* These corrected standard errors can then be used to plot appropriate error bars for a within-subject design. While this approach is not particularly complex, it is difficult to automatize. Here, we present precise details as to how to achieve this standardization and how this can be done in SPSS (other statistical packages are discussed briefly in Appendix A). Cousineau and O'Brien (2014) give a more complete description of each step in the process. In the present paper, we shall simply say that the data is first centered using the following equation

$$Y_{si} = X_{si} - \overline{X}_{s\cdot} + \overline{X}_{\cdot\cdot} \qquad (1)$$

where Y_{sj} is the transformed score for subject s in condition j, X_{sj} is the original score of the s^{th} participant in the j^{th} condition, $\bar{X}_{s\cdot}$ is the mean for the participant across the conditions, and $\bar{X}_{\cdot\cdot}$ is the overall mean. The correction factor is then incorporated using this equation

$$W_{si} = \sqrt{\frac{J}{J-1}} \times (Y_{si} - \overline{Y}_{\cdot i}) + \overline{Y}_{\cdot i}$$

where $\bar{Y}_{\cdot j}$ is the mean for the condition across participants and $\sqrt{J/(J-1)}$ is the correction factor described in the introduction (with $J = \prod_i^P M_i$, described above). The mean of the W values is the same as the mean of the original data X, but the spread has been modified twice. A typical plot (e.g. a line or bar chart) of means of W containing error bars will in fact show the within-subject error bars of the data X.

We provide in Appendix A sample codes in three common programming language, Matlab, R and Mathematica that will standardize (Equation 1) and correct variance (Equation 2). For convenience, we present in Appendix B a short program that can be cut and pasted into a syntax window of SPSS that will automatize everything. While the code in Appendix B is sufficient for computing the standardized data described in this section in SPSS, in the next section we present an extension command for SPSS that is much cleaner and more transparent than the code in Appendix B.

WSplot bundle for SPSS

For users of SPSS version 18 and above, we provide a new command that performs all the operations described in the previous section. To install the bundle the first time, refer to Appendix C. When installed, a new option will be available in the Graphs section of the toolbar called "Within-subject error bar plot"

The new SPSS command is called WSPLOT and has the following syntax:

```
WSPLOT
   /WSVARIABLES within-subject variable …
[  /BSVARIABLES between-subject variable … ]
[  /INTERVALS    [ CI  {  (95** )  } ]      ]
                       {  (alpha)  }
                 [ SE  {  (  1** )  } ]
                       {  (  mult  )  }
[  /METHOD  { SINGLE **    } { REGULAR ** }
            { DIFFERENCE   } { TWOTIER    } ]
[  /PRINT   [ SE ] [ DEBUG ] ].
```

** Default if the subcommand is omitted.
[ ] Brackets denote optional sub-commands
{ } Braces denote alternatives.

For example, using the sample data in the previous section, we would obtain the plot with error bars reflecting within-subject variability with the following syntax command:

```
WSPLOT
   /WSVARIABLES OneSyllable TwoSyllable.
```

In this example, there is no between-subject grouping variable. A dummy group is created, called EBTempGroup, containing only one group.

A graphical interface in SPSS

However, users do not need to enter any syntax if they do not wish to. When the bundle is installed, a new menu entry in the Graph menu called *Within-subject error bar plot* (see Figure 3) allows access to the dialog box shown in Figure 5. Using this graphical interface, a user can drag and drop within-subject variables and

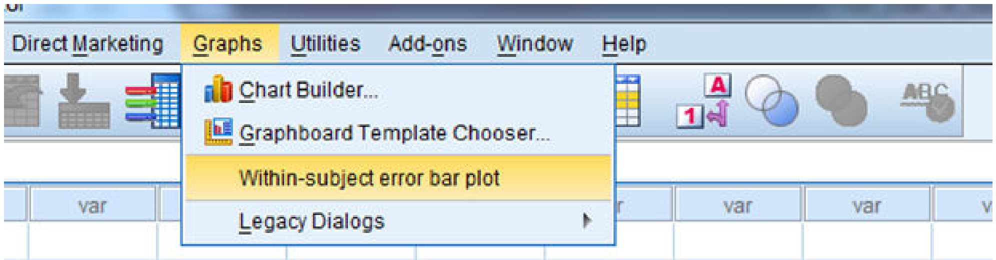

Figure 3 ▪ The new menu entry for accessing the WSPLOT extension command in SPSS

optionally set between-subject variables and other optional settings.

With the checkbox "Print standard errors", the exact value of the within-subject standard errors will be listed in a descriptive statistics output box.

As can be seen in Figure 4, there is a between-subject variable box. This allows the user to separate data into groups before computing errors bars for the within-subject variables in each group. It is important to note that in such an instance, the error bars produced are appropriate only for examining within-subject differences. The error bars are *not* appropriate for examining differences between groups on any of the separate within-subject variables. In instances where a user wishes to examine both between-subject and within-subject differences the plotting method "Two-tier" should be used. With the "Regular" option, bars are produced only for within-subject comparison. With the "Two-tiered" option, two sets of bars are produced on the same plot: one for within-subject comparisons and one for the between-subject comparison (see Baguley, 2012, for a more detailed description of two-tier error bars). This option should be left at "Regular" in instances where there is no between-subject variable being used.

Users are also presented with two "Standard error methods": "Single" and "Difference". The option "Difference" should be used whenever pairwise comparisons are examined. When this is the case, the error bars must be increased by a factor of $2/\sqrt{2}$ (see Baguley, 2012 or Volker and Masson, 2012, for the argument).

Finally, the input box "Multiplier" is used to define the width of the error bars. By default, confidence intervals show 95% confidence intervals, but with a multiplier set to 99, a 99% confidence interval is requested.

Conclusion

There is no perfect error bar. All error bars assume something from the data. For example, the between-subject confidence interval assumes the normality of the data. Yet, data may be non-normal (e.g. skewed data). In this last case, asymmetrical error bars might be preferable, an error bar which is longer in the top direction if the data is positively skewed. On the other end, if the data is symmetrical but has a strong, non-normal kurtosis, how do we represent this in an error bar? Not to mention that kurtosis might be different on the left and the right tail of the distribution… It is clear that there is a limit to the information that can be contained in an error bar. As soon as more refined representations are desired, the error bar should be supplanted by other techniques (see, for example, Marmolejo-Ramos & Matsunaga, 2009).

Error bars, as Baguley reminds us "are intended chiefly as an aid to the exploration and interpretation of data. Thus, they may *complement* formal inference, but are not intended to mimic null-hypothesis tests." (2012, p. 173, our emphasis). As such, using complex error bars may be just doing the opposite of what they are meant to achieve, i.e. give the reader a general intuition of the *pattern* of results. Error bars are in no way a substitute for formal statistical tests and Equations 1 and 2 were not derived with this objective in mind. However, in many instances, error bars give greater depth to a plot allowing readers to see not only if there are differences between groups or conditions, but whether these differences are significant ones. It is hoped that the information in this paper will assist those who wish to accurately graph differences for within-subject designs.

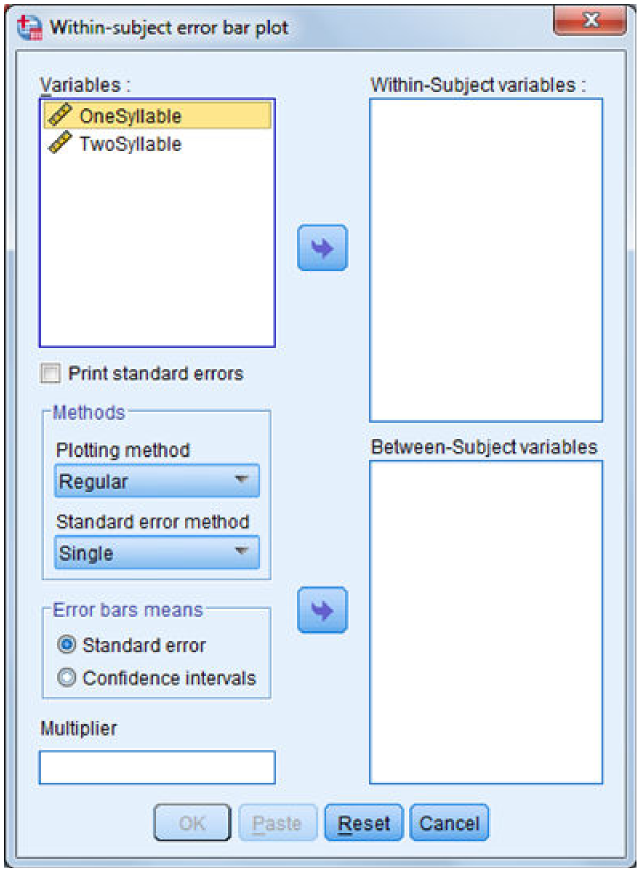

Figure 4 ■ the Graphical user interface for the command WSPLOT

References

Baguley, T. (2012). Calculating and graphing within-subject confidence intervals for ANOVA. *Behavior Research Methods*, *44*, 158-175.

Cousineau, D. (2005). Confidence intervals in within-subjects designs: A simpler solution to Loftus and Masson's method. *Tutorials in Quantitative Methods for Psychology, 1*(1), 42-45.

Cumming, G., & Finch, S. (2005). Inference by eye - Confidence intervals and how to read pictures of data. *American Psychologist, 60*(2), 170-180. doi: 10.1037/0003-066x.60.2.170

Journal of Biological Chemistry. (n.d.). Retrieved September 30, from http://www.jbc.org/site/misc/edpolicy.xhtml

Lacroix, G. L., & Giguère, G. (2006). Formatting data files for repeated-measures analyses in SPSS: Using the Aggregate and Restructure procedures. *Tutorials in Quantitative Methods for Psychology*, *2*, 20-25.

Loftus, G. R., & Masson, M. E. J. (1994). Using confidence-intervals in within subject designs. *Psychonomic Bulletin & Review, 1*(4), 476-490. doi: 10.3758/bf03210951

Morey, R. D. (2008). Confidence intervals from normalised data: A correction to Cousinea (2005). *Tutorials in Quantitative Methods for Psychology, 4*(2), 61-64.

Volker, H. F., & Loftus, G. R. (2012). Standard errors and confidence intervals in within-subjects designs: Generalizing Loftus and Masson (1994) and avoiding the biases of alternative accounts. *Psychonomic Bulletin & Review*, *19*, 395-404.

Wilkinson, L., & the Task Force on Statistical Inference (1999). Statistical methods in psychology journals: Guidelines and explanations. *American Psychologist*, *54*, 594-604.

Appendices follows.

Appendix A: Equations 1 and 2 in three common software programming language

Equations 1 and 2 can easily be programmed in many software, and here we provide code for three commonly used ones, Matlab, R and Mathematica. In Matlab, the code, assuming that J is already computed, is

```
Y = X - repmat(mean(X'),size(X,2),1)' + mean(mean(X))
Z = ( sqrt(J/(J-1)) .* ( Y' - repmat(mean(Y), size(Y,1),1)')'
```

In R, the equivalent code is

```
Y <- X - rowMeans(X) + mean(X)
Z <- t( sqrt(J/(J-1)) * ( t(Y) - colMeans(Y) ) + colMeans(Y)
```

Finally, in Mathematica, the code is

```
Y = X - Mean[X^T] + Mean[Mean[X]]
Z = ( Sqrt[J/(J-1)] ( Y^T - Mean[Y] ) + Mean[Y] )^T
```

Baguley (2012) provided a complete set of instruction in R to obtain a within-subject confidence interval mean plot. Using R, the equivalent command would then be:

```
plot.wsci(data, difference = FALSE)
```

Here, it is assumed that data contains only the two within-subject variables. If data contains extra information, e.g. subject number, in R you can select the desired columns, with, e.g. data[2:4] which will extract columns two to four inclusively.

Appendix B: A short SPSS program that will compute within-subject error bars.

This program requires the Python plug-in for SPSS (see Appendix C for installing this plug-in). This code has only been tested with Python 2.6 and 2.7 (the latest version of Python 2.0 as of the publication of this paper). It has not been tested with python 3.0. The following code can be pasted in a syntax window. Only three lines must be modified; it will also plot the mean plot with the correct error bars. If you are using the bundle described in the text, you do not need to cut and past the following.

The three lines that must be modified are the following

```
variables = ["Likely", "Compared", "Global"]
grouping_variables = ["Q13_2"]
Graph = "SE"
```

On the first line, you must list all the within-subject variables, each within double-quote. On the second line, you must list all the between-subject variables. Finally, on the last line, you choose SE or CI depending on whether you want the plot to show standard errors or confidence intervals. If you are copying and pasting the syntax, then start from just after this sentence:

```
SPLIT FILE OFF.
BEGIN PROGRAM PYTHON.
import spss

#################################################################################
#################################################################################
##  The syntax below is intended to help with the plotting of error bars for within-subject
##  and mixed measures designs (i.e. those designs that contain both within-subject and
##  between-subject designs).
##
##  Firstly, enter your within-subject variable names into the square brackets below beside
##  "variables". Make sure that the variable names are exactly as they appear in
```

```
##  SPSS and that each variable is contained within quotation marks. In the square brackets,
##  each variable should be separately by a comma.
##
##  Next, you need to enter in any between-subject variables into the square brackets
##  associated with "grouping_variables". If there is no grouping variable, then just leave
##  an empty space between the square brackets i.e. [ ]
##  If there is only one between-subject variable, then just have this in between the square
##  brackets and have no commas i.e. ["Independent1"]
##  If you have 2 or three between-subject variables, enter them in as so ["Independent1",
##  "Independent2"] or ["Independent1", "Independent2", "Independent3"].
##  The graphs plotted below can only deal with 3 independent group variables at most.
##
##  The last thing that you need to do is specify whether you what the error bars on the
##  SPSS graphs to represent confidence intervals or standard errors.  Beside where it says
##  "Graph" below, enter "CI" (including the quotation marks) or "SE"
##  to plot confident intervals or standard errors respectively.
##
##  When you run this syntax (using Run > All in the top toolbars) there will be a new
##  variable in SPSS for each of your within-subject measures variables.  These new
##  variables are not intended for any statistical analysis. They are purely for
##  the plotting of error bars.
##
##  It's worth noting that the generation of error bars may be unreliable if there is
##  missing data in the dataset. It may be best to filter out any variables which do not
##  have a data point at all levels of the IVs since a repeated measures ANOVA,
##  or a mixed methods one, will exclude these cases from the analysis anyway.
##
##  If the syntax does not work, it may be for the following reasons:
##    1) You don't have the python pluggin installed for SPSS
##    2) You haven't used the correct formatting for the variables and grouping_variables
##  lists  -Google "python lists" if you want more information on the proper way that these
##  lists should be typed.
##    3) You're running the syntax on a file that you've already run it on. Sometimes SPSS
##  kicks up a fuss about creating that already exist. If you've ran the syntax once already
##  and want to run it again (maybe with a different configuration of of variables),
##  try deleting the variables created the first time you ran the code.
##    4) Your variable names are not exactly what they are in the SPSS file -remember that
##  case matters -an uppercase C is not the same as a lowercase c.
##    5) You've accidently changed something in the file other than what's recommended.
#####################################################################################
#####################################################################################
variables =[]
grouping_variables =[]
Graph = "SE"

#####################################################################################
##  Nothing needs to be edited from here on.
#####################################################################################
if len(grouping_variables) == 0:
    grouping_variables.append("EBTempGroup")
```

```
    spss.Submit("""COMPUTE EBTempGroup=1.
EXECUTE.""")

cf = len(variables)/(len(variables)-1.0)

titleString = "".join(variables)
commaString = ", ".join(variables)
GroupingVariablesString = " ".join(grouping_variables)
NewVariables = [x + "_errorbar" for x in variables]
DescriptivesString = " ".join(NewVariables)

spss.Submit("""COMPUTE %s.subj = MEAN(%s).

Aggregate outfile=* mode=addvariables
/break=%s
/%s.group = mean(%s.subj).""" % (titleString, commaString, GroupingVariablesString,
titleString, titleString))

for x in variables:
    spss.Submit("""COMPUTE %s_ebtemp=(%s - %s.subj + %s.group).
AGGREGATE outfile=* mode=addvariables
/break=%s
/%s_ebtempmean = mean(%s_ebtemp).
COMPUTE %s_errorbar = ((%s_ebtemp - %s_ebtempmean) * sqrt(%f)) + %s_ebtempmean.
EXECUTE.""" %(x, x, titleString, titleString, GroupingVariablesString, x, x, x, x, x, cf,x))

#####################################################################################
##  The syntax below is for plotting graphs in SPSS.
#####################################################################################
if Graph == "SE" or "Se" or "sE" or "se":
    graphingString = "MEANSE(%s, 1) " * len(variables) % tuple(variables)
else:
    graphingString = "MEANCI(%s, 95) " * len(variables) % tuple(variables)

if Graph == "SE" or "Se" or "sE" or "se":
    Label = "Error bars shows +/- 1 SE"
else:
    Label = "Bâtons de variation : 95pc CI"

if len(grouping_variables) == 1:
    spss.Submit("""GGRAPH
  /GRAPHDATASET NAME="graphdataset"
   VARIABLES=%s %s
   MISSING=LISTWISE REPORTMISSING=YES
   TRANSFORM=VARSTOCASES(SUMMARY="#SUMMARY" INDEX="#INDEX" LOW="#LOW" HIGH="#HIGH")
  /GRAPHSPEC SOURCE=INLINE.
BEGIN GPL
  SOURCE: s=userSource(id("graphdataset"))
  DATA: %s=col(source(s), name("%s"), unit.category())
  DATA: SUMMARY=col(source(s), name("#SUMMARY"))
```

```
  DATA: INDEX=col(source(s), name("#INDEX"), unit.category())
  DATA: LOW=col(source(s), name("#LOW"))
  DATA: HIGH=col(source(s), name("#HIGH"))
  COORD: rect(dim(1,2), cluster(3,0))
  GUIDE: axis(dim(3), label("%s"))
  GUIDE: axis(dim(2), label("Moyenne"))
  GUIDE: legend(aesthetic(aesthetic.color.interior), label(""))
  GUIDE: text.footnote(label("%s"))
  SCALE: linear(dim(2), include(0))
  SCALE: cat(aesthetic(aesthetic.color.interior), include("0", "1"))
  SCALE: cat(dim(1), include("0", "1"))
  ELEMENT: interval(position(INDEX*SUMMARY*%s), color.interior(INDEX),
    shape.interior(shape.square))
  ELEMENT: interval(position(region.spread.range(INDEX*(LOW+HIGH)*%s)),
    shape.interior(shape.ibeam))
END GPL.""" % (grouping_variables[0], graphingString, grouping_variables[0],
grouping_variables[0], grouping_variables[0], Label, grouping_variables[0],
grouping_variables[0]))

elif len(grouping_variables) == 2:
    spss.Submit("""GGRAPH
  /GRAPHDATASET NAME="graphdataset" VARIABLES=%s %s %s MISSING=LISTWISE REPORTMISSING=NO
    TRANSFORM=VARSTOCASES(SUMMARY="#SUMMARY" INDEX="#INDEX" LOW="#LOW" HIGH="#HIGH")
  /GRAPHSPEC SOURCE=INLINE.
BEGIN GPL
  SOURCE: s=userSource(id("graphdataset"))
  DATA: %s=col(source(s), name("%s"), unit.category())
  DATA: SUMMARY=col(source(s), name("#SUMMARY"))
  DATA: INDEX=col(source(s), name("#INDEX"), unit.category())
  DATA: %s=col(source(s), name("%s"), unit.category())
  DATA: LOW=col(source(s), name("#LOW"))
  DATA: HIGH=col(source(s), name("#HIGH"))
  COORD: rect(dim(1,2), cluster(3,0), wrap())
  GUIDE: axis(dim(3), label("%s"))
  GUIDE: axis(dim(2), label("Moyenne"))
  GUIDE: axis(dim(4), label("%s"), opposite())
  GUIDE: legend(aesthetic(aesthetic.color.interior), label(""))
  GUIDE: text.footnote(label("%s"))
  SCALE: linear(dim(2), include(0))
  SCALE: cat(aesthetic(aesthetic.color.interior), include("0", "1"))
  SCALE: cat(dim(1), include("0", "1"))
  ELEMENT: interval(position(INDEX*SUMMARY*%s*%s), color.interior(INDEX),
    shape.interior(shape.square))
  ELEMENT: interval(position(region.spread.range(INDEX*(LOW+HIGH)*%s*%s)),
    shape.interior(shape.ibeam))
END GPL.""" % (grouping_variables[0], graphingString, grouping_variables[1],
grouping_variables[0], grouping_variables[0], grouping_variables[1], grouping_variables[1],
grouping_variables[0], grouping_variables[1], Label, grouping_variables[0],
grouping_variables[1], grouping_variables[0], grouping_variables[1]))
```

```
elif len(grouping_variables) == 3:
    spss.Submit("""GGRAPH
  /GRAPHDATASET NAME="graphdataset"
    VARIABLES=%s %s %s %s
    MISSING=LISTWISE REPORTMISSING=NO
    TRANSFORM=VARSTOCASES(SUMMARY="#SUMMARY" INDEX="#INDEX" LOW="#LOW" HIGH="#HIGH")
  /GRAPHSPEC SOURCE=INLINE.
BEGIN GPL
  SOURCE: s=userSource(id("graphdataset"))
  DATA: %s=col(source(s), name("%s"), unit.category())
  DATA: SUMMARY=col(source(s), name("#SUMMARY"))
  DATA: INDEX=col(source(s), name("#INDEX"), unit.category())
  DATA: %s=col(source(s), name("%s"), unit.category())
  DATA: %s=col(source(s), name("%s"), unit.category())
  DATA: LOW=col(source(s), name("#LOW"))
  DATA: HIGH=col(source(s), name("#HIGH"))
  COORD: rect(dim(1,2), cluster(3,0), wrap())
  GUIDE: axis(dim(3), label("%s"))
  GUIDE: axis(dim(2), label("Moyenne"))
  GUIDE: axis(dim(4), label("%s"), opposite())
  GUIDE: axis(dim(5), label("%s"), opposite())
  GUIDE: legend(aesthetic(aesthetic.color.interior), label(""))
  GUIDE: text.footnote(label("%s"))
  SCALE: linear(dim(2), include(0))
  SCALE: cat(aesthetic(aesthetic.color.interior), include("0", "1"))
  SCALE: cat(dim(1), include("0", "1"))
  ELEMENT: interval(position(INDEX*SUMMARY*%s*%s*%s), color.interior(INDEX),
    shape.interior(shape.square))
  ELEMENT: interval(position(region.spread.range(INDEX*(LOW+HIGH)*%s*%s*%s)),
    shape.interior(shape.ibeam))
END GPL.""" % (grouping_variables[0], graphingString, grouping_variables[1],
grouping_variables[2], grouping_variables[0], grouping_variables[0], grouping_variables[1],
grouping_variables[1], grouping_variables[2], grouping_variables[2], grouping_variables[0],
grouping_variables[1], grouping_variables[2], Label, grouping_variables[0],
grouping_variables[1], grouping_variables[2], grouping_variables[0], grouping_variables[1],
grouping_variables[2]))

#####################################################################################
##  The commands below are just to clean up all the variables that were only  used for the
##  calculation of the error bar variables.
#####################################################################################
for x in variables:
    spss.Submit("""DELETE VARIABLES %s_ebtemp %s_ebtempmean.
EXECUTE.""" % (x, x))

spss.Submit("""DELETE VARIABLES %s.subj %s.group.
EXECUTE.""" % (titleString, titleString))

if grouping_variables[0] == "EBTempGroup":
    spss.Submit("""DELETE VARIABLES EBTempGroup.
```

```
EXECUTE.""")

##########################################################################################
##  The syntax below produced descriptive statistics based on the new variables.
##########################################################################################
if grouping_variables[0] != "EBTempGroup":
    spss.Submit("""SPLIT FILE LAYERED BY %s.""" % (GroupingVariablesString))
spss.Submit("""DESCRIPTIVES VARIABLES=%s
  /STATISTICS=MEAN SEMEAN.""" % (DescriptivesString))

END PROGRAM.
SPLIT FILE OFF.
```

Appendix C: installing the extension command for SPSS

You need to follow two steps in order to be able to use the Within-Subject Error Bars extension command to SPSS. The steps given are for Windows as this is the only OS that the package works with at present. If you intend to just copy and paste the syntax from Appendix B, then you need only install the Python plug-in described below.

Python

You must install the python plug-in for windows. Follow this url to download the official SPSS plug-in:

`https://www.ibm.com/developerworks/mydeveloperworks/wikis/home/wiki/We70df3195ec8_4f95_9773_42e448fa9029/page/Downloads%20for%20IBM%C2%AE%20SPSS%C2%AE%20Statistics?lang=en`

This plug-in will also install python 2.7 which is required for the function to work. Just install all of the above in the same way you would any other program.

Extension

You must then download and install the extension "WSPLOT.SPE" from the journal's website. To install the extension, launch SPSS and go to the menu UTILITIES/EXTENSION BUNDLES/INSTALL EXTENSION BUNDLE... Locate the place on your computer where WSPLOT.SPE was downloaded to and choose this file. If you receive the error "Cannot install extension to any of the specified locations" then you need to make a small security adjustment.

1. Go to your main Program Files folder and right-click the main SPSS folder. It will be called something like SPSSStatisticXX or PASWStatistics (where XX is the version number) and could be located in a folder called IBM.
2. Choose "Properties" and go to the security tab.
3. Click the "Edit" button and under the "Group or user names" choose "Users..."
4. For "Full control", tick the box under "Allow" –all other Allow boxes will automatically tick which is fine.
5. Click "Ok" and "Ok" again.

The extension should now install and you will need to restart SPSS before you can use the new command and/or dialog box. If you have any trouble with this installation, please contact the correspondence author.

Citation

O'Brien, F., & Cousineau, D. (2014). Representing Error bars in within-subject designs in typical software packages. *The Quantitative Methods for Psychology, 10* (1), 56-67.

Received: 04/08/12