R code and real-data illustrations for "Analyzing data from single-case alternating treatments designs"

Rumen Manolov and Patrick Onghena

Contents

1 Initial information about this document

In the present document we offer some R code we created flor analyzing alternating treatments designs (ATDs) data and represinting visually the data being analyzed. The document accompanies a manuscript on the topic and serves as its online supplemental material. R (R Core Team, 2013) is a free software, available at `https://cran.r-project.org/`. We indicate how the code can be used (i.e., what part of the code has to be modified so that users can analyze their own data) and we also offer the default output of the code.

Nevertheless, we do not claim that this is the only code available for ATD data. Specifically, the SCDA plug-in for R (Bulté & Onghena, 2012, 2013; `https://cran.r-project.org/web/packages/RcmdrPlugin.SCDA/index.html`) offers the possibility to represent the data graphically and also the apply randomization tests. Moreover, in the CD accompanying the book by Edgington and Onghena (2007) there are randomization tools relevant for ATDs. NAP and Tau-U can implemented using the R code developed by Kevin Tarlow and available at (Brossart, Vannest, Davis, & Patience, 2014; `https://dl.dropboxusercontent.com/u/2842869/Tau_U.R`) or via the website `singlecaseresearch.org`.

Regarding the software implementations of extensions of piecewise regression, several commercial options exist (e.g., HLM, Stata, SAS) and SAS code has been provided in different articles for multiple baseline designs (e.g., Baek & Ferron, 2013; Moeyaert, Ferron, et al., 2014; Moeyaert, Ugille, et al., 2014), or using R (Manolov, Moeyaert, & Evans, 2015) but none is focused on ATD data. The same is the case for the standardized mean difference by Hedges, Pustejovsky, Shadish (2012, 2013), which is applicable to single-case experimental designs different from ATDs. The the *scdhlm* package for R implementing these d-statistics is available at `https://github.com/jepusto/scdhlm`. Finally, free software is also available for simulation modeling analysis (Borckardt Nash, 2014; `http://clinicalresearcher.org/software.htm`), but this analytical technique has not been discussed in the context of ATDs either.

Baek, E. K., & Ferron, J. M. (2013). Multilevel models for multiple-baseline data: Modeling across-participant variation in autocorrelation and residual variance. *Behavior Research Methods*, *45*, 65-74.

Borckardt, J., Nash, M. (2014). Simulation modelling analysis for small sets of single-subject data collected over time. *Neuropsychological Rehabilitation*, *24*, 492-506.

Brossart, D. F., Vannest, K., Davis, J., & Patience, M. (2014). Incorporating nonoverlap indices with visual analysis for quantifying intervention effectiveness in single-case experimental designs. *Neuropsychological Rehabilitation*, *24*, 464-491.

Bulté, I., & Onghena, P. (2012). When the truth hits you between the eyes: A software tool for the visual analysis of single-case experimental data. *Methodology*, *8*, 104-114.

Bulté, I., & Onghena, P. (2013). The Single-Case Data Analysis package: Analysing single-case experiments with R software. *Journal of Modern Applied Statistical Methods*, *12*, 450-478.

Edgington, E. S., & Onghena, P. (2007). *Randomization tests* (4th ed.). London: Chapman Hall/CRC.

Hedges, L. V., Pustejovsky, J. E., Shadish, W. R. (2012). A standardized mean difference effect size for single case designs. *Research Synthesis Methods*, *3*, 224-239.

Hedges, L. V., Pustejovsky, J. E., Shadish, W. R. (2013). A standardized mean difference effect size for multiple baseline designs across individuals. *Research Synthesis Methods*, *4*, 324-341.

Manolov, R., Moeyaert, M., & Evans, J. J. (2015, April). *Single-case data analysis: Software resources for applied researchers*. Unpublished software tutorial. doi: DOI: 10.13140/RG.2.1.3981.7766. Retrieved November 2, 2015 from `https://www.researchgate.net/publication/275517964_Single-case_data_analysis_Software_resources_for_applied_researchers`

Moeyaert, M., Ugille, M., Ferron, J., Beretvas, S. N., & Van Den Noortgate, W. (2014). The influence of the design matrix on treatment effect estimates in the quantitative analyses of single-case experimental designs research. *Behavior Modification*, *38*, 665-704.

Moeyaert, M., Ferron, J., Beretvas, S., & Van Den Noortgate, W. (2014). From a single-level analysis to a multilevel analysis of since-case experimental designs. *Journal of School Psychology*, *52*, 191-211.

R Core Team (2013). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. `http://www.R-project.org/`.

Further information on single-case data software can be consulted from the following sources:

Chen, L.-T., Peng, C.-Y. J., & Chen, M.-E. (2015). Computing tools for implementing standards for single-case designs. *Behavior Modification*, *39*, 835-69.

Manolov, R., & Moeyaert, M. (2016, August 22). How can single-case data be analyzed? Software resources, tutorial, and reflections on analysis. *Behavior Modification*. Advance online publication. doi: 10.1177/0145445516664307

The second part of the document offers the numerical and graphical results of applying the analytical techniques to real data obtained using alternating treatments designs. The data sets chosen are all part of the review presented in the main text of the manuscript. Moreover, the data sets are represented on Figure 1 of the manuscript. The idea was to progress from simpler data patterns in the first illustrations to more complex data patterns in the last illustrations, in order to explore how well the different models fit the data and how much do the quantifications differ.

The references of the articles from which the data sets were extracted are:

Andersen, M. N., Daly, III, A. J., & Young, N. D. (2013). Examination of a one-trial brief experimental analysis to identify reading fluency interventions. *Psychology in the Schools*, *50*, 403−414.

Bryant, B. R., Ok, M., Kang, E. Y., Kim, M. K., Lang, R., Bryant, D. P., & Pfannestiel, K. (2015). Performance of fourth-grade students with learning disabilities on multiplication facts comparing teacher-mediated and technology-mediated interventions: A preliminary investigation. *Journal of Behavioral Education*, *24*, 255−272.

Coleman, M. B., Cherry, R. A., Moore, T. C., Park, Y., & Cihak, D. F. (2015). Teaching sight words to elementary students with intellectual disability and autism: A comparison of teacher-directed versus computer-assisted simultaneous prompting. *Intellectual and Developmental Disabilities*, *53*, 196−210.

Eilers, H. J., & Hayes, S. C. (2015). Exposure and response prevention therapy with cognitive defusion exercises to reduce repetitive and restrictive behaviors displayed by children with autism spectrum disorder. *Research in Autism Spectrum Disorders*, *19*, 18−31.

Sil, S., Dahlquist, L. M., & Burns, A. J. (2013). Case study: videogame distraction reduces behavioral distress in a preschool-aged child undergoing repeated burn dressing changes: a single-subject design. *Journal of Pediatric Psychology*, *38*, 330−341.

Yakubova, G., & Bouck, E. C. (2014). Not all created equally: Exploring calculator use by students with mild intellectual disability. *Education and Training in Autism and Developmental Disabilities*, *49*, 111−126.

2 New proposals

2.1 Average difference between successive observations: ADISO

2.1.1 Performing all comparisons in the same order

First, note that the code is created for performing comparisons in the AB order, whereas A and B can be the labels specified for any condition. Second, the values obtained are specified, separated by commas, after **score <- c(**. The labels for the conditions are provided in quotation marks ("") and separated by commas after **condi<- c(**.In order for the code to work without further modifications, one of the conditions should be called "A" and the other one "B". It is necessary also to specify the aim after **aim** <- using the words (within quotation marks) **"reduce"** or **"increase"** target behavior. The remaining part of the code is copied and pasted in the R console, which leads to obtaining the results.

```
# Input data: values in score <- c() separated by commas
# Specify conditions using the labels A and B
# Specify aim
```

```
score <- c(26.7,33.1,16.6,26.9,20.1,13.2,16.7,3.1,6.6,0,36.4,13.1,9.8,9.8,0,0)
condi<- c("A","A","B","B","A","A","B","B","A","B","A","B","B","A","B","B")
aim <- "reduce"
```

```
# Copy and paste the rest of the code in the R console
```

```
# Objects needed for the calculations
changes<- 0
for (i in 1:(length(score)-1))
  if (condi[i]!=condi[i+1]) changes <- changes + 1
groups<- changes + 1
count_nonA<- 0
Aends<- rep(0,groups/2)
for (i in 2:length(score))
  if ((condi[i]!="A") && (condi[i]!=condi[i-1]))
  {
    count_nonA<- count_nonA + 1
    Aends[count_nonA] <- i-1
  }
count_nonB<- 0
Bends <- rep(0,groups/2)
Bends[groups/2] <- length(score)
for (i in 2:length(score))
  if ((condi[i]!="B") && (condi[i]!=condi[i-1]))
  {
    count_nonB<- count_nonB + 1
    Bends[count_nonB] <- i-1
  }
Astarts<- rep(0,groups/2)
Astarts[1] <- 1
for (i in 2:(groups/2))
  Astarts[i] <- Bends[i-1] + 1
Bstarts<- rep(0,groups/2)
for (i in 1:(groups/2))
  Bstarts[i] <- Aends[i] + 1
mean_dif<- rep(0,groups/2)
weights<- rep(0,groups/2)
B_greater<- 0
A_greater<- 0
for (i in 1:(groups/2))
{
```

```
  mean_dif[i] <- mean(score[Astarts[i]:Aends[i]])-mean(score[Bstarts[i]:Bends[i]])
  weights[i] <- Aends[i]-Astarts[i]+1 + Bends[i]-Bstarts[i]+1
  if ((aim=="reduce") && (mean_dif[i] > 0)) B_greater<- B_greater + 1
  if((aim=="increase") && (mean_dif[i] < 0)) B_greater<- B_greater + 1
  if ((aim=="reduce") && (mean_dif[i] < 0))A_greater<- A_greater + 1
  if ((aim=="increase") && (mean_dif[i] > 0)) A_greater<- A_greater + 1
}
num<- 0
den<- 0
for (i in 1:(groups/2))
{
  num<- num + mean_dif[i]*weights[i]
  den<- den + weights[i]
}
average<- num/den
A_superiority<- A_greater/(groups/2)
B_superiority<- B_greater/(groups/2)
print("Differences"); print(mean_dif)
print("Weights per difference"); print(weights)
paste("ADISO = ", round(average,2))
paste("Percentage of comparisons with A superiority = ", round(A_superiority*100,2))
paste("Percentage of comparisons with B superiority = ", round(B_superiority*100,2))
```

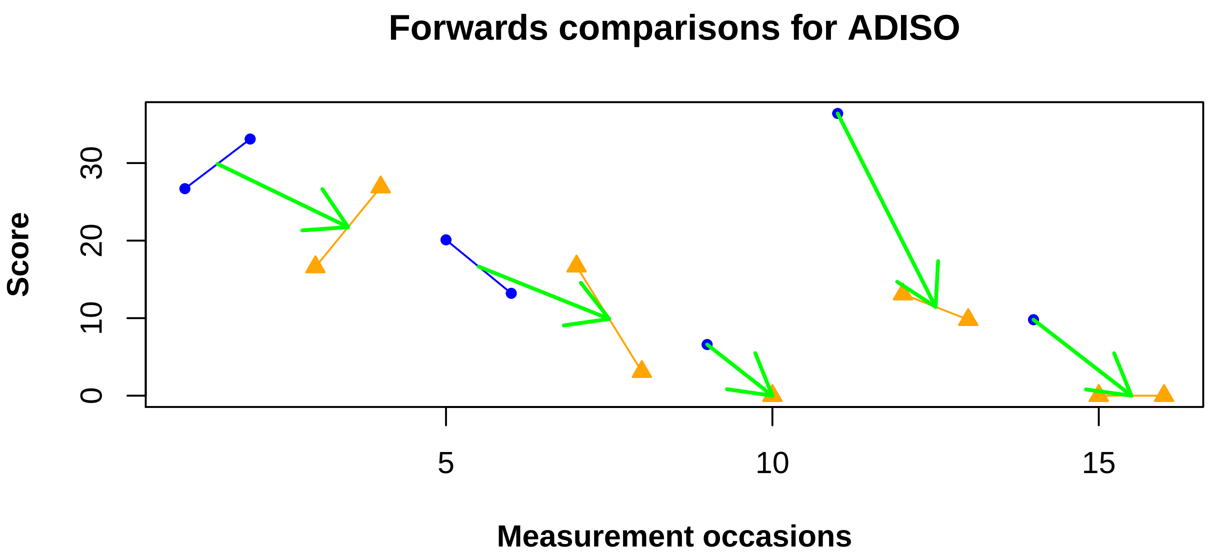
Code ends here. Default output of the code below:

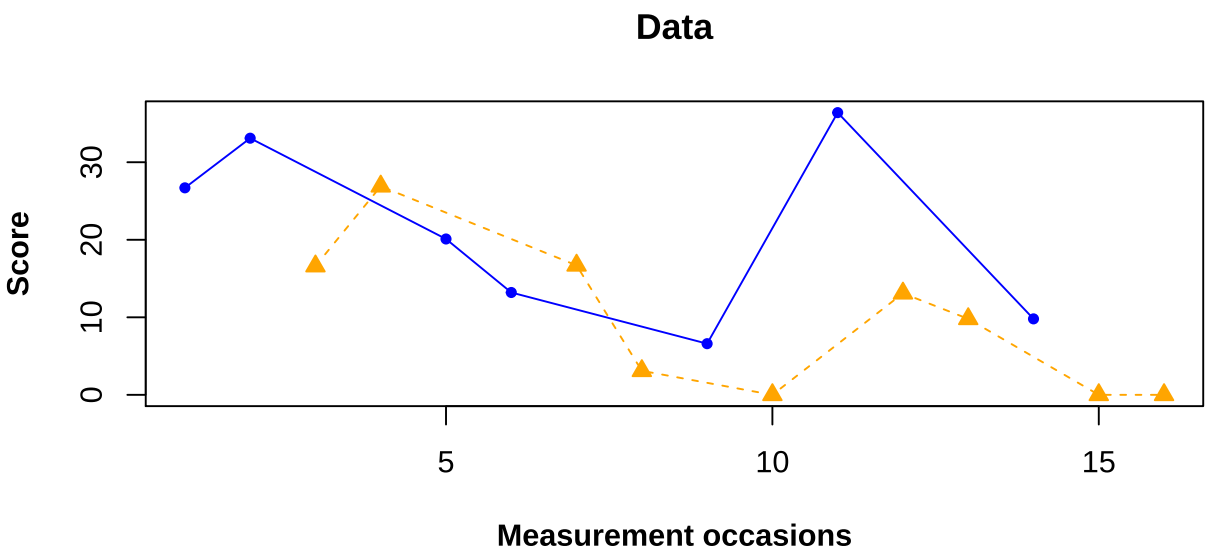
```
## [1] "Differences"
## [1]  8.15  6.75  6.60 24.95  9.80
## [1] "Weights per difference"
## [1] 4 4 2 3 3
## [1] "ADISO =  11.07"
## [1] "Percentage of comparisons with A superiority =  0"
## [1] "Percentage of comparisons with B superiority =  100"
```

The following code allows obtaining the graphical representation of what is being compared

```
indep<- 1:length(score)
par(mfrow=c(2,1))
plot(indep,score, xlim=c(indep[1],indep[length(indep)]),
     ylim=c(min(score),max(score)), xlab="Measurement occasions",
     ylab="Score", font.lab=2, pch=" ")
points(indep[condi=="B"], score[condi=="B"], pch=24,
       bg="orange", col="orange")
points(indep[condi=="A"], score[condi=="A"], pch=20,
       bg="blue", col="blue")

for (i in 1: (groups/2))
{  if (Astarts[i] != Aends[i])
    lines(c(Astarts[i],Aends[i]),c(score[Astarts[i]],
                                   score[Aends[i]]),col="blue");
   if (Bstarts[i] != Bends[i])
     lines(c(Bstarts[i],Bends[i]),c(score[Bstarts[i]],
                                    score[Bends[i]]),col="orange")
}

for (i in 1: (groups/2))
{
  if ( (aim == "reduce") &&
         (mean(score[Astarts[i]:Aends[i]]) >= mean(score[Bstarts[i]:Bends[i]])) )
    arrows(mean(c(Astarts[i],Aends[i])),mean(score[Astarts[i]:Aends[i]]),
           mean(c(Bstarts[i],Bends[i])),mean(score[Bstarts[i]:Bends[i]]), col="green", lwd=2);

  if ( (aim == "reduce") &&
         (mean(score[Astarts[i]:Aends[i]]) < mean(score[Bstarts[i]:Bends[i]])) )
    arrows(mean(c(Astarts[i],Aends[i])),mean(score[Astarts[i]:Aends[i]]),
           mean(c(Bstarts[i],Bends[i])),mean(score[Bstarts[i]:Bends[i]]), col="red", lwd=2);

  if ( (aim == "increase") &&
         (mean(score[Astarts[i]:Aends[i]]) >= mean(score[Bstarts[i]:Bends[i]])) )
    arrows(mean(c(Astarts[i],Aends[i])),mean(score[Astarts[i]:Aends[i]]),
           mean(c(Bstarts[i],Bends[i])),mean(score[Bstarts[i]:Bends[i]]), col="red", lwd=2);

  if ( (aim == "increase") &&
         (mean(score[Astarts[i]:Aends[i]]) < mean(score[Bstarts[i]:Bends[i]])) )
    arrows(mean(c(Astarts[i],Aends[i])),mean(score[Astarts[i]:Aends[i]]),
           mean(c(Bstarts[i],Bends[i])),mean(score[Bstarts[i]:Bends[i]]), col="green", lwd=2);

}
title(main="Forwards comparisons for ADISO")

plot(indep,score, xlim=c(indep[1],indep[length(indep)]),
     ylim=c(min(score),max(score)), xlab="Measurement occasions",
     ylab="Score", font.lab=2, pch=" ")
points(indep[condi=="B"], score[condi=="B"], pch=24, bg="orange", col="orange")
points(indep[condi=="A"], score[condi=="A"], pch=20, bg="blue", col="blue")
lines(indep[condi=="B"],score[condi=="B"],lty="dashed",col="orange")
lines(indep[condi=="A"],score[condi=="A"],lty="solid",col="blue")
title(main="Data")
```

The default graphical output is:





2.1.2 Performing comparisons in a user-defined order

This code refers to the same procedure, but instead of carrying out the comparisons in the same order (AB, regardless of which condition is labelled "A" and which "B") it allows comparing conditions according to segmentation points defined by the user. In that sense, it is necessary to mark after which measurement occasion should a segmentation take place. In the current example, the segmentation point 3 means that the first three values will be compared, with the software comparing the A and B conditions' measurements included in this segment. Segmentation point 5 means that what is compared are the A and B conditions' values recorded up to measurement occasion 5, but excluding the first three measurement occasions, which were already part of the previous comparison.

```
# Input data: values in score <- c() separated by commas
# Specify conditions using the labels A and B
# Specify the measurement occasions after which the segmentation is performed
# Specify aim
```

```r
score <- c(26.7,33.1,16.6,26.9,20.1,13.2,16.7,3.1,6.6,0,36.4,13.1,9.8,9.8,0,0)
condi<- c("A","A","B","B","A","A","B","B","A","B","A","B","B","A","B","B")
cutsentry <- c(3,5,8,10,13,16)
aim<- "reduce"
```

```
# Copy and paste the rest of the code in the R console
```

```r
# Objects needed for the calculations
ends <- c(0,cutsentry)
starts <- (ends[1:length(cutsentry)]+1)
difs <- rep(0,length(cutsentry))
weights <- rep(0,length(cutsentry))


for (i in 1:length(cutsentry))
{
  difs[i] <- mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE) -
             mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE);
  weights[i] <- length(c(score[starts[i]:ends[i+1]]))
}

num<- 0
den<- 0
B_greater<- 0
A_greater<- 0
for (i in 1:length(cutsentry))
{
  num<- num + difs[i]*weights[i]
  den<- den + weights[i]
  if ((aim=="reduce") && (difs[i] > 0)) B_greater<- B_greater + 1
  if((aim=="increase") && (difs[i] < 0)) B_greater<- B_greater + 1
  if ((aim=="reduce") && (difs[i] < 0))A_greater<- A_greater + 1
  if ((aim=="increase") && (difs[i] > 0)) A_greater<- A_greater + 1
}
average<- num/den
A_superiority<- A_greater/length(cutsentry)
B_superiority<- B_greater/length(cutsentry)

print("Differences"); print(difs)
print("Weights per difference"); print(weights)
paste("ADISO = ", round(average,2))
paste("Percentage of comparisons with A superiority = ", round(A_superiority*100,2))
paste("Percentage of comparisons with B superiority = ", round(B_superiority*100,2))
```

Code ends here. Default output of the code below:

```
## [1] "Differences"
## [1] 13.30 -6.80  3.30  6.60 24.95  9.80
## [1] "Weights per difference"
## [1] 3 2 3 2 3 3
## [1] "ADISO =  9.6"
## [1] "Percentage of comparisons with A superiority =  16.67"
## [1] "Percentage of comparisons with B superiority =  83.33"
```

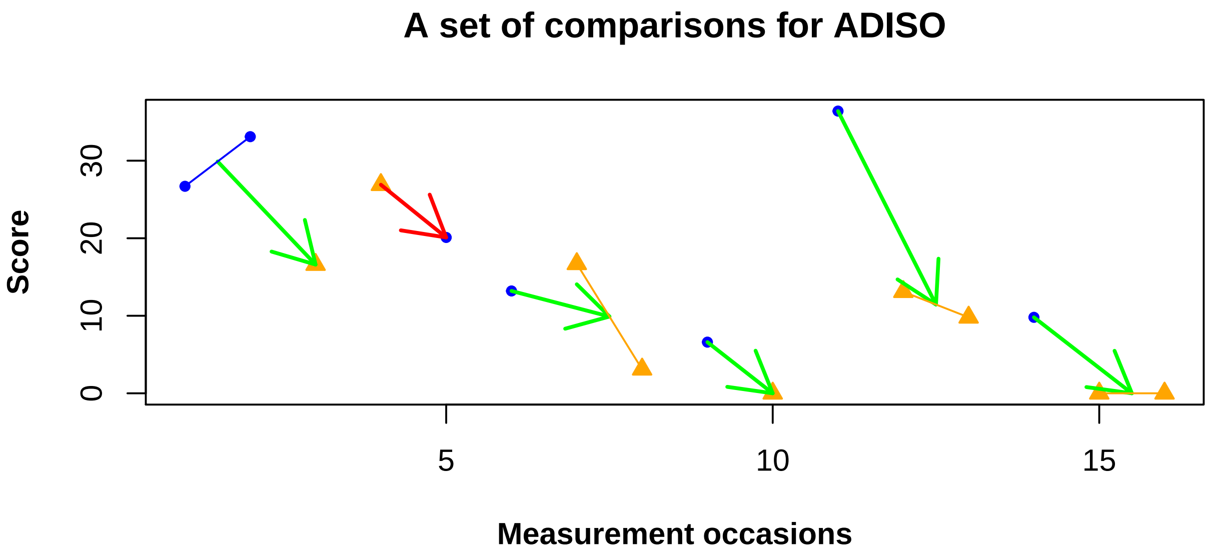
The following code allows obtaining the graphical representation of what is being compared

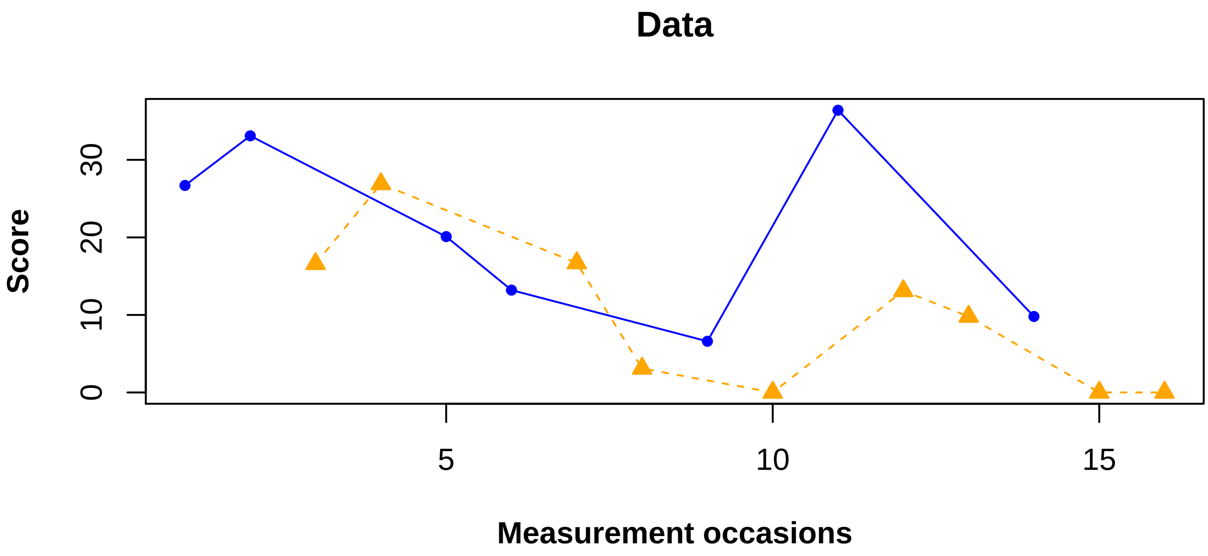
```
par(mfrow=c(2,1))
indep <- 1:length(score)
plot(indep,score, xlim=c(indep[1],indep[length(indep)]),
     ylim=c(min(score),max(score)), xlab="Measurement occasions",
     ylab="Score", font.lab=2, pch=" ")
points(indep[condi=="B"], score[condi=="B"], pch=24, bg="orange", col="orange")
points(indep[condi=="A"], score[condi=="A"], pch=20, bg="blue", col="blue")


for (i in 1: (length(cutsentry)))
{
  if ( (aim == "reduce") &&
          (mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE) >=
            mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE))  &&
          (mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]) <
            mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"])) )
    arrows(mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE),
           mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE),
           col="green", lwd=2);


  if ( (aim == "reduce") &&
          (mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE) >=
            mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE))  &&
          (mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]) >
            mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"])) )
    arrows(mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE),
           mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE),
           col="green", lwd=2);


  if ( (aim == "reduce") &&
          (mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE) <
            mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE))  &&
          (mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]) <
            mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"])) )
    arrows(mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE),
           mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE),
           col="red", lwd=2);


  if ( (aim == "reduce") &&
          (mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE) <
            mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE))  &&
          (mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]) >
            mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"])) )
    arrows(mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE),
           mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE),
           col="red", lwd=2);
```

```
  if ( (aim == "increase") &&
          (mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE) >=
            mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE))   &&
          (mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]) <
            mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"])) )
    arrows(mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE),
           mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE),
           col="red", lwd=2);


  if ( (aim == "increase") &&
          (mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE) >=
            mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE))   &&
          (mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]) >
            mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"])) )
    arrows(mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE),
           mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE),
           col="red", lwd=2);


  if ( (aim == "increase") &&
          (mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE) <
            mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE))   &&
          (mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]) <
            mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"])) )
    arrows(mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE),
           mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE),
           col="green", lwd=2);


  if ( (aim == "increase") &&
          (mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE) <
            mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE))   &&
          (mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]) >
            mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"])) )
    arrows(mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"],na.rm=TRUE),
           mean(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]),
           mean(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"],na.rm=TRUE),
           col="green", lwd=2);
}


for (i in 1: length(cutsentry))
{  if (length(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"]) == 2)
  lines(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="A"],
        c(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="A"]),col="blue");
  if (length(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"]) == 2)
    lines(c(starts[i]:ends[i+1])[condi[starts[i]:ends[i+1]]=="B"],
          c(score[starts[i]:ends[i+1]][condi[starts[i]:ends[i+1]]=="B"]),col="orange")
}

title(main="A set of comparisons for ADISO")
```

```
plot(indep,score, xlim=c(indep[1],indep[length(indep)]),
     ylim=c(min(score),max(score)), xlab="Measurement occasions",
     ylab="Score", font.lab=2, pch=" ")
points(indep[condi=="B"], score[condi=="B"], pch=24, bg="orange", col="orange")
points(indep[condi=="A"], score[condi=="A"], pch=20, bg="blue", col="blue")
lines(indep[condi=="B"],score[condi=="B"],lty="dashed",col="orange")
lines(indep[condi=="A"],score[condi=="A"],lty="solid",col="blue")
title(main="Data")
```

The default graphical output is:





2.2 A comparison of actual and linearly interpolated values: ALIV

The values obtained are specified, separated by commas, after **score <- c(**. The labels for the conditions are provided in quotation marks ("") and separated by commas after **condi<- c(**. In order for the code to work without further modifications, one of the conditions should be called "A" and the other one "B". The aim of the study is specified after **aim <-** via either **"reduce"** or **"increase"**.

```
# Input data: values in score <- c() separated by commas
# Specify conditions using the labels A and B
# Specify aim
```

```
score <- c(26.7,33.1,16.6,26.9,20.1,13.2,16.7,3.1,6.6,0,36.4,13.1,9.8,9.8,0,0)
condi<- c("A","A","B","B","A","A","B","B","A","B","A","B","B","A","B","B")
aim <- "reduce" # Alternatively aim <- "increase"
```

```
# Copy and paste the rest of the code in the R console
```

```
indep<- 1:length(score)
time_B<- indep[condi=="B"]
score_B<- score[condi=="B"]
time_A<- indep[condi=="A"]
score_A<- score[condi=="A"]

# Create necessary objects
interp.A.temp <- rep(0,length(score))
interp.B.temp <- rep(0,length(score))
interp.A.temp[time_A] <- score[time_A]
interp.B.temp[time_B] <- score[time_B]
# Linear interpolation for condition A
for (i in 1:(length(time_A)-1))
  if (time_A[i] != time_A[i+1] - 1)
  {
    if ((time_A[i] == time_A[i+1] - 2))
        interp.A.temp[time_A[i]+1] <- interp.A.temp[time_A[i]] +
      (interp.A.temp[time_A[i+1]]-interp.A.temp[time_A[i]])/2
    if ((time_A[i] == time_A[i+1] - 3))
      {
        interp.A.temp[time_A[i]+1] <- interp.A.temp[time_A[i]] +
          (interp.A.temp[time_A[i+1]]-interp.A.temp[time_A[i]])/3;
        interp.A.temp[time_A[i]+2] <- interp.A.temp[time_A[i]] +
          2*((interp.A.temp[time_A[i+1]]-interp.A.temp[time_A[i]])/3)
      }
  }

# Linear interpolation for condition B
for (i in 1:(length(time_B)-1))
  if (time_B[i] != time_B[i+1] - 1)
  {
    if ((time_B[i] == time_B[i+1] - 2))
      interp.B.temp[time_B[i]+1] <- interp.B.temp[time_B[i]] +
      (interp.B.temp[time_B[i+1]]-interp.B.temp[time_B[i]])/2
    if ((time_B[i] == time_B[i+1] - 3))
    {
      interp.B.temp[time_B[i]+1] <- interp.B.temp[time_B[i]] +
        (interp.B.temp[time_B[i+1]]-interp.B.temp[time_B[i]])/3;
      interp.B.temp[time_B[i]+2] <- interp.B.temp[time_B[i]] +
        2*((interp.B.temp[time_B[i+1]]-interp.B.temp[time_B[i]])/3)
    }
```

```
  }

# Exclude initial and final measurement occasions
start <- max(time_A[1],time_B[1])
end <- min(time_A[length(time_A)],time_B[length(time_B)])
interp.A <- interp.A.temp[start:end]
interp.B <- interp.B.temp[start:end]
# Compare all values for the same measurement occasions
diff.interp <- interp.B-interp.A
print("ALIV Differences"); print(round(diff.interp),digits=3)

## [1] "ALIV Differences"
##  [1] -12   2   3   7   6  -6  -5 -22 -30 -14  -9  -5

paste("ALIV Mean difference = ", mean(diff.interp))

## [1] "ALIV Mean difference =  -7"
```

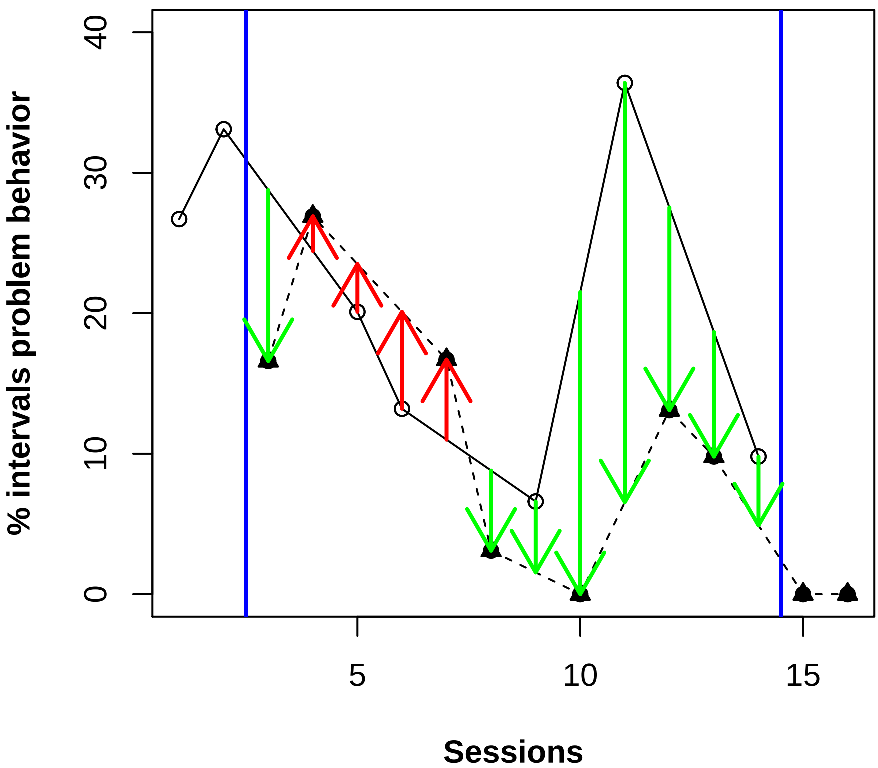
Code ends here. Default output of the code below:

```
## [1] "ALIV Differences"
##  [1] -12   2   3   7   6  -6  -5 -22 -30 -14  -9  -5
## [1] "ALIV Mean difference =  -7"
```

The following code allows obtaining the graphical representation of what is being compared

```
indep<- 1:length(score)
plot(indep,score, xlim=c(indep[1],indep[length(indep)]),ylim=c(0,40),
     xlab="Sessions", ylab="% intervals problem behavior", font.lab=2)
points(indep[condi=="B"], score[condi=="B"], pch=24, bg="black")
points(indep[condi=="A"], score[condi=="A"], bg="black")
lines(indep[condi=="B"],score[condi=="B"],lty="dashed")
lines(indep[condi=="A"],score[condi=="A"],lty="solid")
abline(v=max(time_A[1],time_B[1])-0.5,col="blue",lwd=2)
abline(v=min(time_A[length(time_A)],time_B[length(time_B)])+0.5,
       col="blue",lwd=2)

sharedtime <- start:end
for (i in 1:length(sharedtime))
{
  if ( (aim == "reduce") && (interp.A[i] >= interp.B[i]) )
 arrows(sharedtime[i],interp.A[i],sharedtime[i],interp.B[i],
        col="green", lwd=2)
  if ( (aim == "reduce") && (interp.A[i] < interp.B[i]) )
arrows(sharedtime[i],interp.A[i],sharedtime[i],interp.B[i],
       col="red", lwd=2)
  if ( (aim == "increase") && (interp.A[i] >= interp.B[i]) )
arrows(sharedtime[i],interp.A[i],sharedtime[i],interp.B[i],
       col="red", lwd=2)
  if ( (aim == "increase") && (interp.A[i] < interp.B[i]) )
arrows(sharedtime[i],interp.A[i],sharedtime[i],interp.B[i],
       col="green", lwd=2)
}
```

The default graphical output is:



2.3 Randomization test using ALIV as test statistic

The values obtained are specified, separated by commas, after **score <- c(**. The labels for the conditions are provided in quotation marks ("") and separated by commas after **condi<- c(**. In order for the code to work without further modifications, one of the conditions should be called "A" and the other one "B". The aim of the study is specified after **aim <-** via either **"reduce"** or **"increase"**.

```
# First: copy and paste the ALIV function
```

```
aliv <- function(score, condi, aim) {

  indep<- 1:length(score)
  time_B<- indep[condi=="B"]
  score_B<- score[condi=="B"]
  time_A<- indep[condi=="A"]
  score_A<- score[condi=="A"]

  # Create necessary objects
  interp.A.temp <- rep(0,length(score))
  interp.B.temp <- rep(0,length(score))
  interp.A.temp[time_A] <- score[time_A]
  interp.B.temp[time_B] <- score[time_B]

  # Linear interpolation for condition A
  for (i in 1:(length(time_A)-1))
    if (time_A[i] != time_A[i+1] - 1)
    {
      if ((time_A[i] == time_A[i+1] - 2))
        interp.A.temp[time_A[i]+1] <- interp.A.temp[time_A[i]] +
        (interp.A.temp[time_A[i+1]]-interp.A.temp[time_A[i]])/2
      if ((time_A[i] == time_A[i+1] - 3))
      {
        interp.A.temp[time_A[i]+1] <- interp.A.temp[time_A[i]] +
          (interp.A.temp[time_A[i+1]]-interp.A.temp[time_A[i]])/3;
        interp.A.temp[time_A[i]+2] <- interp.A.temp[time_A[i]] +
          2*((interp.A.temp[time_A[i+1]]-interp.A.temp[time_A[i]])/3)
      }
      if ((time_A[i] == time_A[i+1] - 4))
      {
        interp.A.temp[time_A[i]+1] <- interp.A.temp[time_A[i]] +
          (interp.A.temp[time_A[i+1]]-interp.A.temp[time_A[i]])/4;
        interp.A.temp[time_A[i]+2] <- interp.A.temp[time_A[i]] +
          2*((interp.A.temp[time_A[i+1]]-interp.A.temp[time_A[i]])/4);
        interp.A.temp[time_A[i]+3] <- interp.A.temp[time_A[i]] +
          3*((interp.A.temp[time_A[i+1]]-interp.A.temp[time_A[i]])/4)
      }
      if ((time_A[i] == time_A[i+1] - 5))
      {
        interp.A.temp[time_A[i]+1] <- interp.A.temp[time_A[i]] +
          (interp.A.temp[time_A[i+1]]-interp.A.temp[time_A[i]])/5;
        interp.A.temp[time_A[i]+2] <- interp.A.temp[time_A[i]] +
          2*((interp.A.temp[time_A[i+1]]-interp.A.temp[time_A[i]])/5);
        interp.A.temp[time_A[i]+3] <- interp.A.temp[time_A[i]] +
          3*((interp.A.temp[time_A[i+1]]-interp.A.temp[time_A[i]])/5);
        interp.A.temp[time_A[i]+4] <- interp.A.temp[time_A[i]] +
          4*((interp.A.temp[time_A[i+1]]-interp.A.temp[time_A[i]])/5)
      }
    }
```

```
  # Linear interpolation for condition B
  for (i in 1:(length(time_B)-1))
    if (time_B[i] != time_B[i+1] - 1)
    {
      if ((time_B[i] == time_B[i+1] - 2))
        interp.B.temp[time_B[i]+1] <- interp.B.temp[time_B[i]] +
        (interp.B.temp[time_B[i+1]]-interp.B.temp[time_B[i]])/2
      if ((time_B[i] == time_B[i+1] - 3))
      {
        interp.B.temp[time_B[i]+1] <- interp.B.temp[time_B[i]] +
          (interp.B.temp[time_B[i+1]]-interp.B.temp[time_B[i]])/3;
        interp.B.temp[time_B[i]+2] <- interp.B.temp[time_B[i]] +
          2*((interp.B.temp[time_B[i+1]]-interp.B.temp[time_B[i]])/3)
      }
      if ((time_B[i] == time_B[i+1] - 4))
      {
        interp.B.temp[time_B[i]+1] <- interp.B.temp[time_B[i]] +
          (interp.B.temp[time_B[i+1]]-interp.B.temp[time_B[i]])/4;
        interp.B.temp[time_B[i]+2] <- interp.B.temp[time_B[i]] +
          2*((interp.B.temp[time_B[i+1]]-interp.B.temp[time_B[i]])/4);
        interp.B.temp[time_B[i]+3] <- interp.B.temp[time_B[i]] +
          3*((interp.B.temp[time_B[i+1]]-interp.B.temp[time_B[i]])/4)
      }
    }
  # Exclude initial and final measurement occasions
  start <- max(time_A[1],time_B[1])
  end <- min(time_A[length(time_A)],time_B[length(time_B)])
  interp.A <- interp.A.temp[start:end]
  interp.B <- interp.B.temp[start:end]
  # Compare all values for the same measurement occasions
  if (aim =="increase") diff.interp <- interp.B-interp.A
  if (aim =="reduce") diff.interp <- interp.A-interp.B
  return(mean(diff.interp))
}
```

```
# Second:
# Input data: values in score <- c() separated by commas
# Specify conditions using the labels A and B
# Specify aim
```

```
score <- c(26.7,33.1,16.6,26.9,20.1,13.2,16.7,3.1,6.6,0,36.4,13.1,9.8,9.8,0,0)
condi <- c("A","A","B","B","A","A","B","B","A","B","A","B","B","A","B","B")
aim <- "reduce" # Alternatively aim <- "increase"
```

```
# Third:
# Locate a previously created file including all possible randomizations
# For situations in which n_A = n_B the SCDA plug-in (Bulté \& Onghena, 2013) can be used
# Otherwise the executable file from the CD accompanying the book by Edgington \& Onghena (2007) can be
```

```
sequ <- read.table(file.choose(),header=FALSE,sep=",")
```

```
# Fourth:
# Copy and paste the rest of the code in the R console
```

```
actualALIV <- aliv(score,condi,aim)

if (aim=="reduce") actualMD <- mean(score[condi=="A"])-mean(score[condi=="B"]) else
                   actualMD <- mean(score[condi=="B"])-mean(score[condi=="B"])

md <- rep(0,dim(sequ)[1])
alivs <- rep(0,dim(sequ)[1])

for (i in 1:dim(sequ)[1])
{
  if (aim=="reduce") md[i] <- mean(score[sequ[i,]=="A"])-mean(score[sequ[i,]=="B"])
  if (aim=="increase")  md[i] <- mean(score[sequ[i,]=="B"])-mean(score[sequ[i,]=="A"])
  alivs[i] <- aliv(score,sequ[i,],aim)
}

count_md <- 0
count_alivs <- 0
for (i in 1:dim(sequ)[1])
{
  if (actualMD <= md[i]) count_md <- count_md +1
  if (actualALIV <= alivs[i]) count_alivs <- count_alivs +1
}

p_md <- count_md/dim(sequ)[1]
p_aliv <- count_alivs/dim(sequ)[1]

paste("Mean difference = ", round(actualMD,2))
paste("p value for the Mean difference = ", round(p_md,4))

paste("ALIV = ", round(actualALIV,2))
paste("p value for ALIV = ", round(p_aliv,4))
```

```
# Example of 50 possible randomizations for the Eilers and Hayes (2015) data
```

```
##    V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 V16
## 1   A  A  B  A  A  B  B  A  B   B   A   B   B   A   B   B
## 2   A  A  B  A  B  A  B  A  B   B   A   B   B   A   B   B
## 3   A  A  B  A  B  A  B  B  A   B   A   B   B   A   B   B
## 4   A  A  B  A  B  A  B  B  A   B   B   A   B   A   B   B
## 5   A  A  B  A  B  A  B  B  A   B   B   A   B   B   A   B
## 6   A  A  B  A  B  B  A  A  B   B   A   B   B   A   B   B
## 7   A  A  B  A  B  B  A  B  A   B   A   B   B   A   B   B
## 8   A  A  B  A  B  B  A  B  A   B   B   A   B   A   B   B
## 9   A  A  B  A  B  B  A  B  A   B   B   A   B   B   A   B
## 10  A  A  B  A  B  B  A  B  B   A   A   B   B   A   B   B
## 11  A  A  B  A  B  B  A  B  B   A   B   A   B   A   B   B
## 12  A  A  B  A  B  B  A  B  B   A   B   A   B   B   A   B
## 13  A  A  B  A  B  B  A  B  B   A   B   B   A   A   B   B
## 14  A  A  B  A  B  B  A  B  B   A   B   B   A   B   A   B
## 15  A  A  B  A  B  B  A  B  B   A   B   B   A   B   B   A
## 16  A  A  B  B  A  A  B  A  B   B   A   B   B   A   B   B
## 17  A  A  B  B  A  A  B  B  A   B   A   B   B   A   B   B
## 18  A  A  B  B  A  A  B  B  A   B   B   A   B   A   B   B
## 19  A  A  B  B  A  A  B  B  A   B   B   A   B   B   A   B
## 20  A  A  B  B  A  B  A  A  B   B   A   B   B   A   B   B
## 21  A  A  B  B  A  B  A  B  A   B   A   B   B   A   B   B
## 22  A  A  B  B  A  B  A  B  A   B   B   A   B   A   B   B
## 23  A  A  B  B  A  B  A  B  A   B   B   A   B   B   A   B
## 24  A  A  B  B  A  B  A  B  B   A   A   B   B   A   B   B
## 25  A  A  B  B  A  B  A  B  B   A   B   A   B   A   B   B
## 26  A  A  B  B  A  B  A  B  B   A   B   A   B   B   A   B
## 27  A  A  B  B  A  B  A  B  B   A   B   B   A   A   B   B
## 28  A  A  B  B  A  B  A  B  B   A   B   B   A   B   A   B
## 29  A  A  B  B  A  B  A  B  B   A   B   B   A   B   B   A
## 30  A  A  B  B  A  B  B  A  A   B   A   B   B   A   B   B
## 31  A  A  B  B  A  B  B  A  A   B   B   A   B   A   B   B
## 32  A  A  B  B  A  B  B  A  A   B   B   A   B   B   A   B
## 33  A  A  B  B  A  B  B  A  B   A   A   B   B   A   B   B
## 34  A  A  B  B  A  B  B  A  B   A   B   A   B   A   B   B
## 35  A  A  B  B  A  B  B  A  B   A   B   A   B   B   A   B
## 36  A  A  B  B  A  B  B  A  B   A   B   B   A   A   B   B
## 37  A  A  B  B  A  B  B  A  B   A   B   B   A   B   A   B
## 38  A  A  B  B  A  B  B  A  B   A   B   B   A   B   B   A
## 39  A  A  B  B  A  B  B  A  B   B   A   A   B   A   B   B
## 40  A  A  B  B  A  B  B  A  B   B   A   A   B   B   A   B
## 41  A  A  B  B  A  B  B  A  B   B   A   B   A   A   B   B
## 42  A  A  B  B  A  B  B  A  B   B   A   B   A   B   A   B
## 43  A  A  B  B  A  B  B  A  B   B   A   B   A   B   B   A
## 44  A  A  B  B  A  B  B  A  B   B   A   B   B   A   A   B
## 45  A  A  B  B  A  B  B  A  B   B   A   B   B   A   B   A
## 46  A  B  A  A  B  A  B  A  B   B   A   B   B   A   B   B
## 47  A  B  A  A  B  A  B  B  A   B   A   B   B   A   B   B
## 48  A  B  A  A  B  A  B  B  A   B   B   A   B   A   B   B
## 49  A  B  A  A  B  A  B  B  A   B   B   A   B   B   A   B
## 50  A  B  A  A  B  B  A  A  B   B   A   B   B   A   B   B
```

Code ends here. Default output of the code below:

```
## [1] "Mean difference =  11.27"
## [1] "p value for the Mean difference =  0.0143"
## [1] "ALIV =  7"
## [1] "p value for ALIV =  0.0994"
```

3 Previously available analytical options

3.1 Percentage of nonoverlapping data by Wolery et al.

The values obtained are specified, separated by commas, after **score <- c(**. The labels for the conditions are provided in quotation marks ("") and separated by commas after **condi<- c(**.In order for the code to work without further modifications, one of the conditions should be called "A" and the other one "B". The remaining part of the code is copied and pasted in the R console, which leads to obtaining the results. The aim of the study is specified after **aim** <- via either **"reduce"** or **"increase"**.

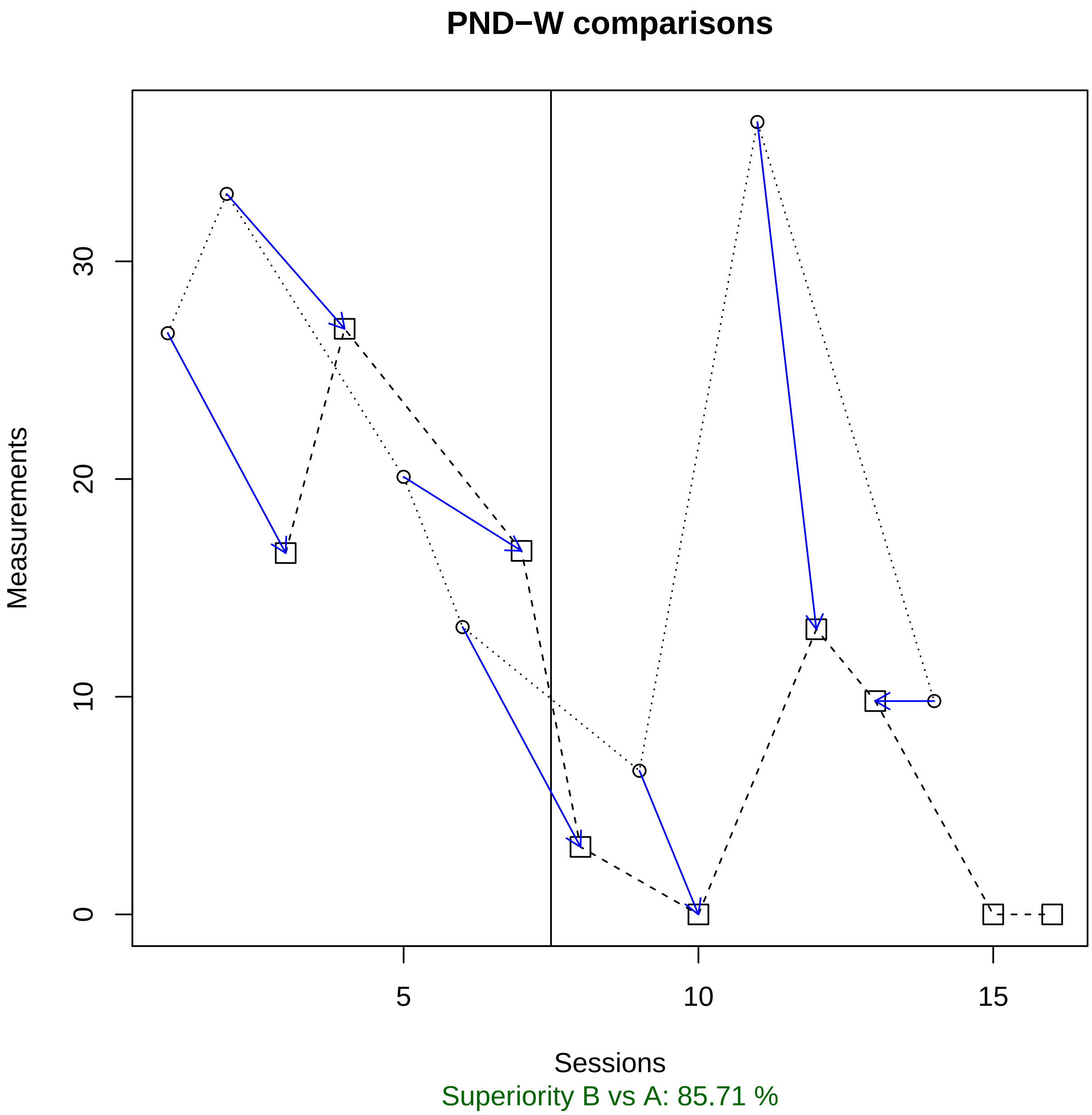
```
# Input data: values in score <- c() separated by commas
# Specify conditions using the labels A and B
# Specify aim
```

```
score <- c(26.7,33.1,16.6,26.9,20.1,13.2,16.7,3.1,6.6,0,36.4,13.1,9.8,9.8,0,0)
condi<- c("A","A","B","B","A","A","B","B","A","B","A","B","B","A","B","B")
aim <- "reduce" # Alternatively: "increase"
```

```
# Copy and paste the rest of the code in the R console
```

```
Adata <- score[condi=="A"]
Bdata <- score[condi=="B"]
indep <- 1:length(score)
comparisons <- min(length(Adata),length(Bdata))

A.super <- 0
B.super <- 0


for (i in 1:comparisons)
{
  if ((aim=="reduce") && (Bdata[i] < Adata[i])) B.super <- B.super + 1
  if ((aim=="reduce") && (Bdata[i] > Adata[i])) A.super <- A.super + 1
  if ((aim=="increase") && (Bdata[i] > Adata[i])) B.super <- B.super + 1
  if ((aim=="increase") && (Bdata[i] < Adata[i])) A.super <- A.super + 1

}


plot(indep,score, xlim=c(indep[1],indep[length(indep)]),ylim=c(min(score),max(score)),
     xlab="Sessions", ylab="Measurements", pch="")
abline (v=(length(Adata)+0.5))
points(indep[condi=="A"], Adata, bg="black")
points(indep[condi=="B"], Bdata, pch=0, bg="black",cex=1.6)
lines(indep[condi=="A"], Adata,lty="dotted")
lines(indep[condi=="B"], Bdata,lty="dashed")
title (main="PND-W comparisons")
for (i in 1:comparisons)
  arrows(indep[condi=="A"][i], Adata[i],
         indep[condi=="B"][i],Bdata[i],col="blue",length=0.10)

title(sub=list(paste("Superiority B vs A:",
                     round((B.super/comparisons)*100,2),"%"),col="darkgreen"))
```

Code ends here. Default output of the code below:



3.2 Nonoverlap all pairs

The values obtained are specified, separated by commas, after **score** <**- c(**. The labels for the conditions are provided in quotation marks ("") and separated by commas after **condi**<**- c(**.In order for the code to work without further modifications, one of the conditions should be called "A" and the other one "B". The remaining part of the code is copied and pasted in the R console, which leads to obtaining the results. The aim of the study is specified after **aim** <**-** via either **"reduce"** or **"increase"**.

```
# Input data: values in score <- c() separated by commas
# Specify conditions using the labels A and B
# Specify aim
```

```
score <- c(26.7,33.1,16.6,26.9,20.1,13.2,16.7,3.1,6.6,0,36.4,13.1,9.8,9.8,0,0)
condi<- c("A","A","B","B","A","A","B","B","A","B","A","B","B","A","B","B")
aim <- "reduce" # Alternatively: "increase"
```

```
# Copy and paste the rest of the code in the R console
```

```
control<- score[condi=="A"]
intervention<- score[condi=="B"]
probsup<-function (x1,x2,aim){
n1 <- sum(!is.na(x1))
n2 <- sum(!is.na(x2))
if (aim=="reduce") mayor<- sum(unlist(lapply(x1,x2,FUN='>')),na.rm=TRUE)
if (aim=="increase") mayor<- sum(unlist(lapply(x1,x2,FUN='<')),na.rm=TRUE)
igual<- 0.5*sum(unlist(lapply(x1,x2,FUN='==')),na.rm=TRUE)
prob_sup <- (mayor+igual)/(n1*n2)
cat('NAP= ', prob_sup,'\n')
}
probsup(control,intervention,aim)
```

Code ends here. Default output of the code below:

```
## NAP=  0.77
```

3.3 Monotonic trends using Tau-U

The values obtained are specified, separated by commas, after **score <- c(**. The labels for the conditions are provided in quotation marks ("") and separated by commas after **condi<- c(**.In order for the code to work without further modifications, one of the conditions should be called "A" and the other one "B". The remaining part of the code is copied and pasted in the R console, which leads to obtaining the results. The aim of the study is specified after **aim** <- via either **"reduce"** or **"increase"**.

```
# Input data: values in score <- c() separated by commas
# Specify conditions using the labels A and B
```

```
score <- c(26.7,33.1,16.6,26.9,20.1,13.2,16.7,3.1,6.6,0,36.4,13.1,9.8,9.8,0,0)
condi<- c("A","A","B","B","A","A","B","B","A","B","A","B","B","A","B","B")
aim <- "reduce" # Alternatively: "increase"
```

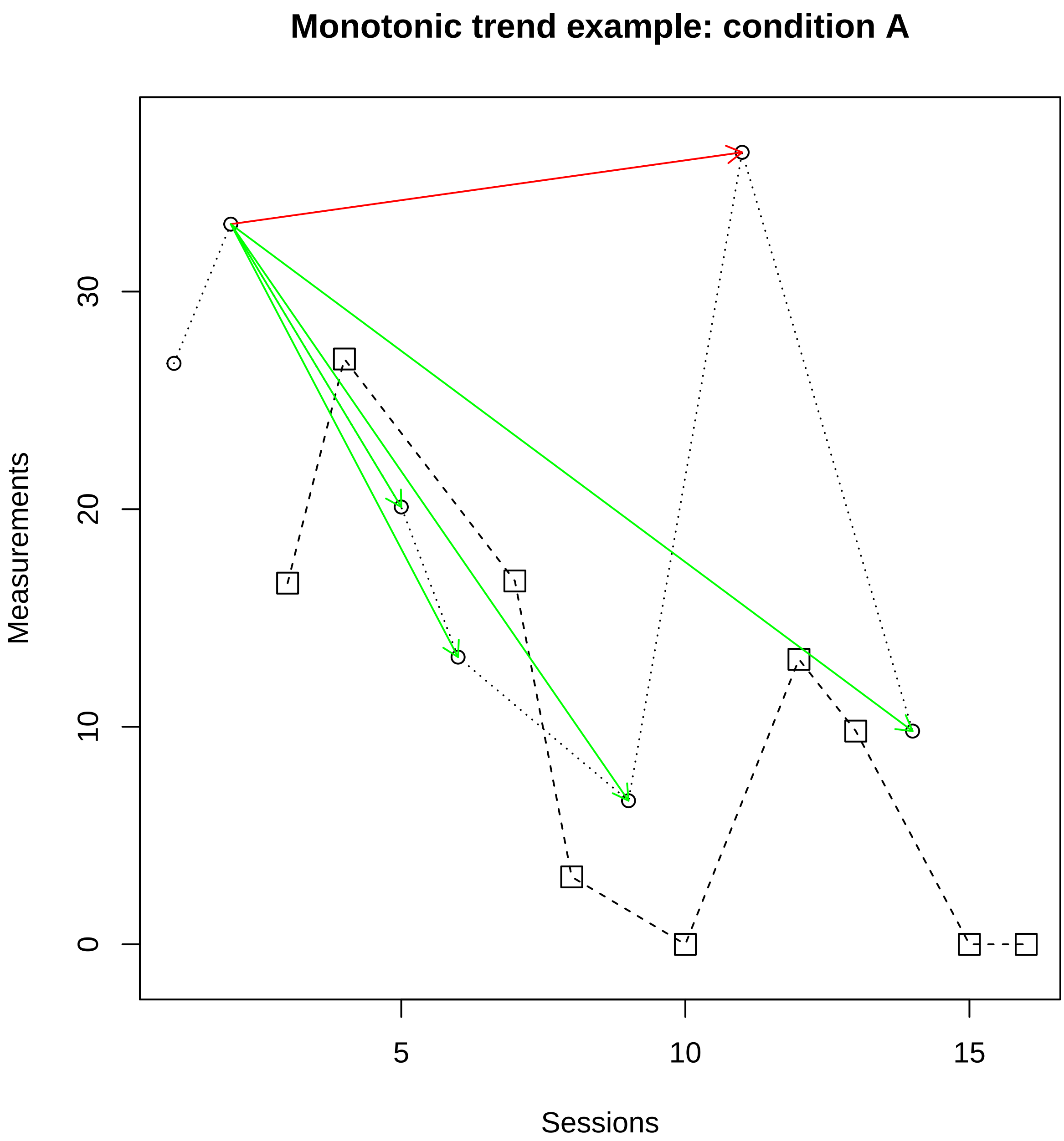
```
# Copy and paste the rest of the code in the R console
```

```
score_c <- score[condi=="A"]
length_c <- length(score[condi=="A"])
count_up <- 0
count_down <- 0
counter <- 0
for (i in length_c:1)
  for (j in 1:(length_c-1))
  if (i > j)
  {
    counter <- counter + 1
    if (score_c[i] > score_c[i-j]) count_up <- count_up + 1
    if (score_c[i] < score_c[i-j]) count_down <- count_down + 1
  }

print("Condition A")
if (count_up > count_down)
  paste("Positive monotonic trend:", round((count_up-count_down)*100/counter,2), "% increase")
if (count_up < count_down)
  paste("Negative monotonic trend:", round((count_down-count_up)*100/counter,2), "% decrease")

score_c <- score[condi=="B"]
length_c <- length(score[condi=="B"])
count_up <- 0
count_down <- 0
counter <- 0
for (i in length_c:1)
  for (j in 1:(length_c-1))
  if (i > j)
  {
    counter <- counter + 1
    if (score_c[i] > score_c[i-j]) count_up <- count_up + 1
    if (score_c[i] < score_c[i-j]) count_down <- count_down + 1
  }

print("Condition B")
if (count_up > count_down)
  paste("Positive monotonic trend:", round((count_up-count_down)*100/counter,2), "% increase")
if (count_up < count_down)
  paste("Negative monotonic trend:", round((count_down-count_up)*100/counter,2), "% decrease")
```

```r
indep <- 1:length(score)
plot(indep,score, xlim=c(indep[1],indep[length(indep)]),ylim=c(min(score)-1,max(score)+1),
 xlab="Sessions", ylab="Measurements", pch="")
points(indep[condi=="A"], score[condi=="A"], bg="black")
points(indep[condi=="B"], score[condi=="B"], pch=0, bg="black",cex=1.6)
lines(indep[condi=="A"],score[condi=="A"],lty="dotted")
lines(indep[condi=="B"],score[condi=="B"],lty="dashed")
title (main="Monotonic trend example: condition A")

Atimes <- 1:length(score[condi=="A"])
where1 <- floor(quantile(Atimes, probs = 0.25, names = FALSE))

for (j in 1:length(score[condi=="A"]))
    if (j > where1)
    {
      if ((score[condi=="A"][where1] > score[condi=="A"][j]) && (aim=="increase"))
       arrows(indep[condi=="A"][where1],score[condi=="A"][where1],
              indep[condi=="A"][j],score[condi=="A"][j],
              col="red",length=0.1)
      if ((score[condi=="A"][where1] < score[condi=="A"][j]) && (aim=="increase"))
       arrows(indep[condi=="A"][where1],score[condi=="A"][where1],
              indep[condi=="A"][j],score[condi=="A"][j],
              col="green",length=0.1)
      if ((score[condi=="A"][where1] < score[condi=="A"][j]) && (aim=="reduce"))
       arrows(indep[condi=="A"][where1],score[condi=="A"][where1],
              indep[condi=="A"][j],score[condi=="A"][j],
              col="red",length=0.1)
      if ((score[condi=="A"][where1] > score[condi=="A"][j]) && (aim=="reduce"))
       arrows(indep[condi=="A"][where1],score[condi=="A"][where1],
              indep[condi=="A"][j],score[condi=="A"][j],
              col="green",length=0.1)
      if (score[condi=="A"][where1] == score[condi=="A"][j])
       arrows(indep[condi=="A"][where1],score[condi=="A"][where1],
              indep[condi=="A"][j],score[condi=="A"][j],
             col="blue",length=0.1)
     }
```

Code ends here. Default output of the code below:

```
## [1] "Condition A"
## [1] "Negative monotonic trend: 33.33 % decrease"
## [1] "Condition B"
## [1] "Negative monotonic trend: 58.33 % decrease"
```



3.4 Compare means

The values obtained are specified, separated by commas, after **score** <- **c(**. The labels for the conditions are provided in quotation marks ("") and separated by commas after **condi**<- **c(**.In order for the code to work without further modifications, one of the conditions should be called "A" and the other one "B". The remaining part of the code is copied and pasted in the R console, which leads to obtaining the results.

```
# Input data: values in score <- c() separated by commas
# Specify conditions using the labels A and B
```

```
score <- c(26.7,33.1,16.6,26.9,20.1,13.2,16.7,3.1,6.6,0,36.4,13.1,9.8,9.8,0,0)
condi<- c("A","A","B","B","A","A","B","B","A","B","A","B","B","A","B","B")
```

```
# Copy and paste the rest of the code in the R console
```

```
paste("Average for control condition = ", round(mean(score[condi=="A"]),2))
paste("Average for intervention condition = ", round(mean(score[condi=="B"]),2))
paste("Difference in means = ", round(mean(score[condi=="A"])- mean(score[condi=="B"]),2))
```

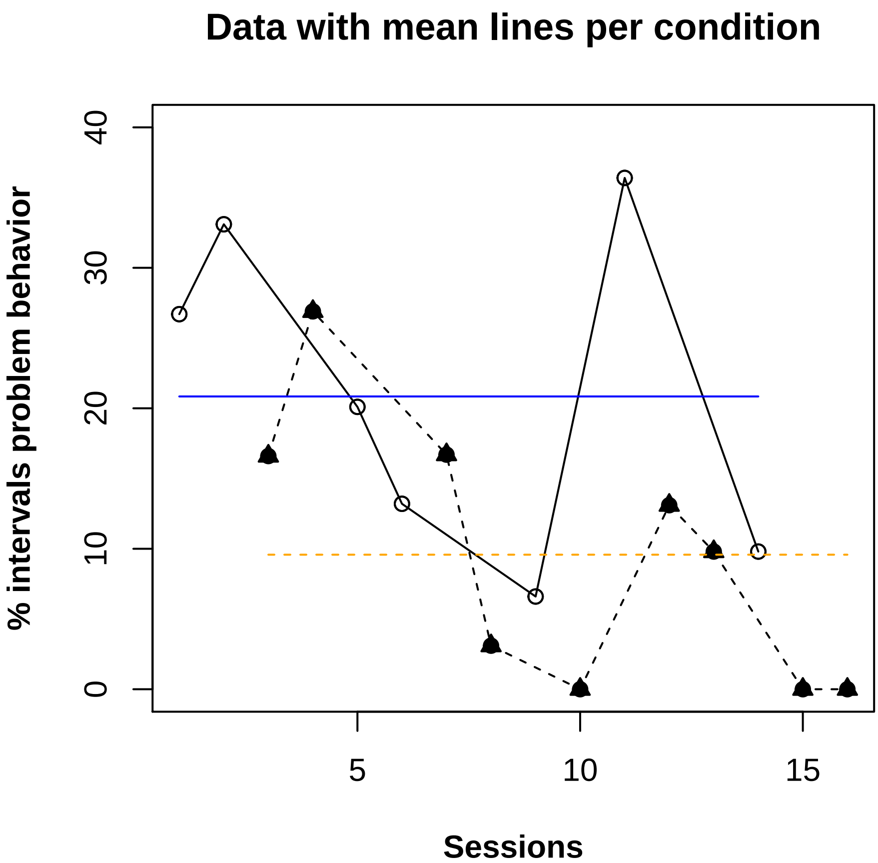
Code ends here. Default output of the code below:

```
## [1] "Average for control condition =  20.84"
## [1] "Average for intervention condition =  9.58"
## [1] "Difference in means =  11.27"
```

The following code allows obtaining the graphical representation of what is being compared

```
# Objects needed for the calculations
indep<- 1:length(score)
plot(indep,score, xlim=c(indep[1],indep[length(indep)]),
     ylim=c(0,40), xlab="Sessions",
     ylab="% intervals problem behavior", font.lab=2)
points(indep[condi=="B"], score[condi=="B"], pch=24, bg="black")
points(indep[condi=="A"], score[condi=="A"], bg="black")
lines(indep[condi=="B"],score[condi=="B"],lty="dashed")
lines(indep[condi=="A"],score[condi=="A"],lty="solid")
title(main="Data with mean lines per condition")
time_B<- indep[condi=="B"]
time_A<- indep[condi=="A"]
score_A<- score[condi=="A"]
score_B<- score[condi=="B"]
mean_A<- rep(mean(score[condi=="A"]),length(score_A))
mean_B<- rep(mean(score[condi=="B"]),length(score_B))
lines(time_B,mean_B,lty="dashed",col="orange")
lines(time_A,mean_A,col="blue")
```

The default graphical output is:



3.5 Carry out two separate linear regressions

The values obtained are specified, separated by commas, after **score <- c(**. The labels for the conditions are provided in quotation marks ("") and separated by commas after **condi<- c(**.In order for the code to work without further modifications, one of the conditions should be called "A" and the other one "B". The remaining part of the code is copied and pasted in the R console, which leads to obtaining the results.

```
# Input data: values in score <- c() separated by commas
# Specify conditions using the labels A and B
```

```
score <- c(26.7,33.1,16.6,26.9,20.1,13.2,16.7,3.1,6.6,0,36.4,13.1,9.8,9.8,0,0)
condi<- c("A","A","B","B","A","A","B","B","A","B","A","B","B","A","B","B")
```

```
# Copy and paste the rest of the code in the R console
```

```
indep<- 1:length(score)
time_B<- indep[condi=="B"]
score_B<- score[condi=="B"]
reg_B<- lm(score_B ~ time_B)
time_A<- indep[condi=="A"]
score_A<- score[condi=="A"]
reg_A<- lm(score_A ~ time_A)
b <- reg_B$coefficients[1] + reg_B$coefficients[2]*indep
a <- reg_A$coefficients[1] + reg_A$coefficients[2]*indep
start<- max(time_A[1],time_B[1])
end<- min(time_A[length(time_A)],time_B[length(time_B)])
d <- a[start:end]-b[start:end]
paste("R-squared for A = ", round(summary(reg_A)$r.squared,2))
paste("R-squared for B = ", round(summary(reg_B)$r.squared,2))
paste("Average difference between predicted values = ", round(mean(d),2))
```

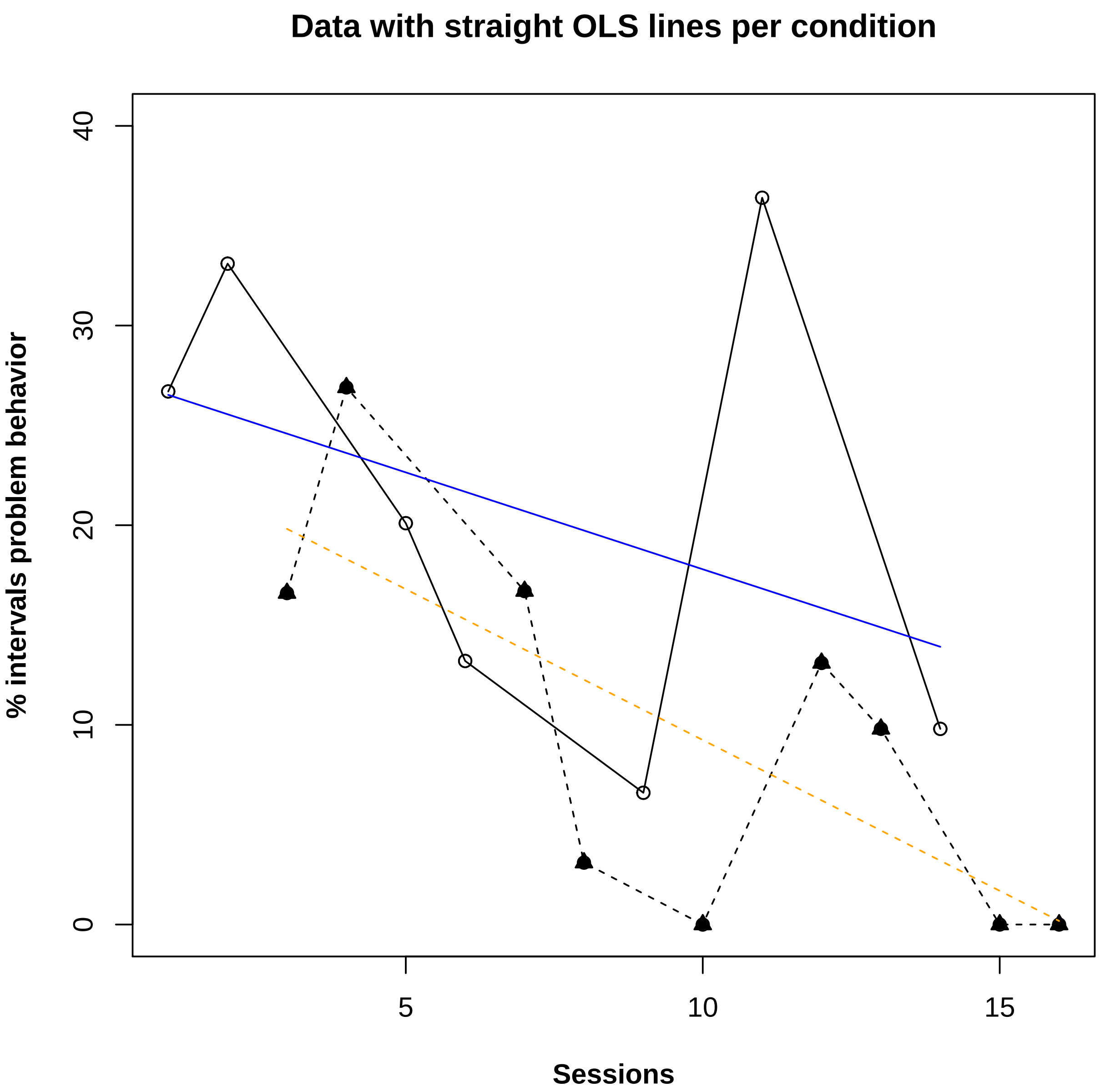
Code ends here. Default output of the code below:

```
## [1] "R-squared for A =  0.16"
## [1] "R-squared for B =  0.54"
## [1] "Average difference between predicted values =  7.74"
```

The following code allows obtaining the graphical representation of what is being compared

```
indep<- 1:length(score)
plot(indep,score, xlim=c(indep[1],indep[length(indep)]),
     ylim=c(0,40), xlab="Sessions",
     ylab="% intervals problem behavior", font.lab=2)
points(indep[condi=="B"], score[condi=="B"], pch=24, bg="black")
points(indep[condi=="A"], score[condi=="A"], bg="black")
lines(indep[condi=="B"],score[condi=="B"],lty="dashed")
lines(indep[condi=="A"],score[condi=="A"],lty="solid")
title(main="Data with straight OLS lines per condition")
lines(time_B,reg_B$fitted,lty="dashed",col="orange")
lines(time_A,reg_A$fitted,col="blue")
```

The default graphical output is:



3.6 Carry out two separate quadratic regressions

The values obtained are specified, separated by commas, after **score <- c(**. The labels for the conditions are provided in quotation marks ("") and separated by commas after **condi<- c(**.In order for the code to work without further modifications, one of the conditions should be called "A" and the other one "B". The remaining part of the code is copied and pasted in the R console, which leads to obtaining the results.

```
# Input data: values in score <- c() separated by commas
# Specify conditions using the labels A and B
```

```
score <- c(26.7,33.1,16.6,26.9,20.1,13.2,16.7,3.1,6.6,0,36.4,13.1,9.8,9.8,0,0)
condi<- c("A","A","B","B","A","A","B","B","A","B","A","B","B","A","B","B")
```

```
# Copy and paste the rest of the code in the R console
```

```
indep<- 1:length(score)
indep_sq<- indep*indep
time_B<- indep[condi=="B"]
time_Bsq<- time_B * time_B
score_B<- score[condi=="B"]
reg_q_B<- lm(score_B ~ time_B + time_Bsq)
time_A<- indep[condi=="A"]
time_Asq<- time_A * time_A
score_A<- score[condi=="A"]
reg_q_A<- lm(score_A ~ time_A + time_Asq)
b <- reg_q_B$coefficients[1] + reg_q_B$coefficients[2]*indep +
     reg_q_B$coefficients[3]*indep_sq
a <- reg_q_A$coefficients[1] + reg_q_A$coefficients[2]*indep +
     reg_q_A$coefficients[3]*indep_sq
start<- max(time_A[1],time_B[1])
end<- min(time_A[length(time_A)],time_B[length(time_B)])
d <- a[start:end]-b[start:end]
paste("R-squared for A = ", round(summary(reg_q_A)$r.squared,2))
paste("R-squared for B = ", round(summary(reg_q_B)$r.squared,2))
paste("Average difference between predicted values = ", round(mean(d),2))
```

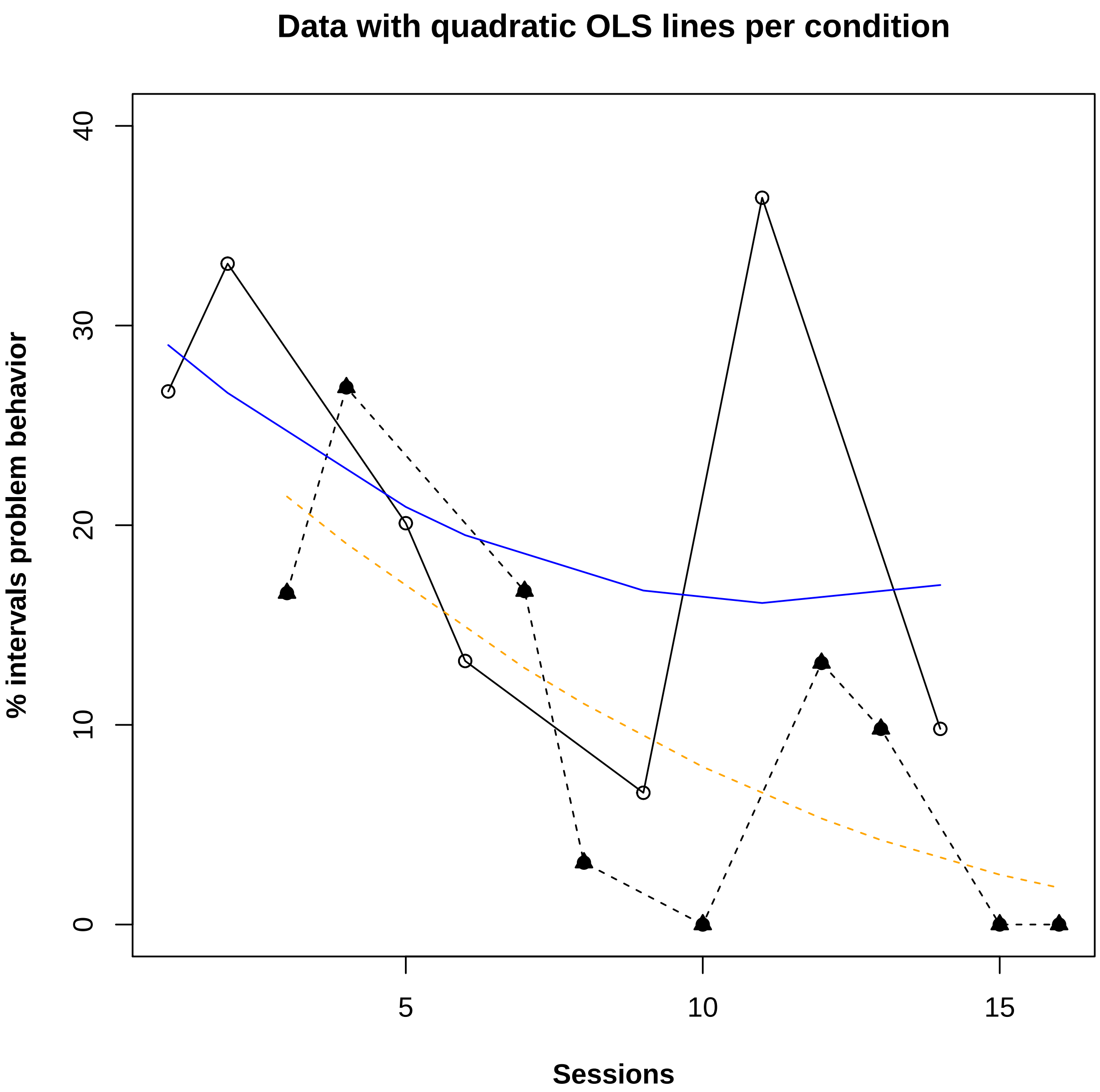
Code ends here. Default output of the code below:

```
## [1] "R-squared for A =  0.19"
## [1] "R-squared for B =  0.55"
## [1] "Average difference between predicted values =  7.44"
```

The following code allows obtaining the graphical representation of what is being compared

```
indep<- 1:length(score)
plot(indep,score, xlim=c(indep[1],indep[length(indep)]),
     ylim=c(0,40), xlab="Sessions",
     ylab="% intervals problem behavior", font.lab=2)
points(indep[condi=="B"], score[condi=="B"], pch=24, bg="black")
points(indep[condi=="A"], score[condi=="A"], bg="black")
lines(indep[condi=="B"],score[condi=="B"],lty="dashed")
lines(indep[condi=="A"],score[condi=="A"],lty="solid")
title(main="Data with quadratic OLS lines per condition")
lines(time_B,reg_q_B$fitted,lty="dashed",col="orange")
lines(time_A,reg_q_A$fitted,col="blue")
```

The default graphical output is:



3.7 Carry out two local regression analyses (LOESS)

The values obtained are specified, separated by commas, after **score <- c(**. The labels for the conditions are provided in quotation marks ("") and separated by commas after **condi<- c(**. In order for the code to work without further modifications, one of the conditions should be called "A" and the other one "B".

```
# Input data: values in score <- c() separated by commas
# Specify conditions using the labels A and B
```

```
score <- c(26.7,33.1,16.6,26.9,20.1,13.2,16.7,3.1,6.6,0,36.4,13.1,9.8,9.8,0,0)
condi<- c("A","A","B","B","A","A","B","B","A","B","A","B","B","A","B","B")
```

The fraction of data (smoothing parameter) to be used can be modified after **fraction1 <-** and **score <-**, with the defaults used being **0.6** and **0.8**. The remaining part of the code is copied and pasted in the R console, which leads to obtaining the results. But they can be changed in the following lines:

```
fraction1 <- 0.60
fraction2 <- 0.80
```

```
# Copy and paste the rest of the code in the R console
```

```
indep<- 1:length(score)
time_B<- indep[condi=="B"]
score_B<- score[condi=="B"]
time_A<- indep[condi=="A"]
score_A<- score[condi=="A"]
reg_B1 <- loess(score_B~time_B,span=fraction1,degree=1)
reg_B2 <- loess(score_B~time_B,span=fraction2,degree=1)
reg_A1 <- loess(score_A~time_A,span=fraction1,degree=1)
reg_A2 <- loess(score_A~time_A,span=fraction2,degree=1)
reg_B1.q <- loess(score_B~time_B,span=fraction1,degree=2)
reg_B2.q <- loess(score_B~time_B,span=fraction2,degree=2)
reg_A1.q <- loess(score_A~time_A,span=fraction1,degree=2)
reg_A2.q <- loess(score_A~time_A,span=fraction2,degree=2)
ss.total<- sum(scale(score_A, center=TRUE, scale=FALSE)^2)
ss.resid<- sum(residuals(reg_A1)^2)
paste("Pseudo R-squared for A = ", round((1-ss.resid/ss.total),2))
ss.total<- sum(scale(score_B, center=TRUE, scale=FALSE)^2)
ss.resid<- sum(residuals(reg_B1)^2)
paste("Pseudo R-squared for B = ", round((1-ss.resid/ss.total),2))
```

We here use the results for linear regressions using 60% of the data (reg_{A1} and reg_{B1}), but it is possible to change that by using the predicted values from a different regression model, by modifying the objects reg_{A1} and reg_{B1} in $allpredicted_A$ and $allpredicted_B$.

```
allpredicted_A<- 1:length(score)
for (i in 1:length(score))
allpredicted_A[i] <- predict(reg_A1, i)
allpredicted_B<- 1:length(score)
for (i in 1:length(score))
allpredicted_B[i] <- predict(reg_B1, i)
start<- 1
end<- length(score)
for (i in 1:length(score))
{
if ( (!is.na(allpredicted_A[i])) &&
```

```
       (!is.na(allpredicted_B[i])) && (start==1) ) start <- i
if ( (!is.na(allpredicted_A[i])) &&
       (!is.na(allpredicted_B[i])) ) end <- i
}
d_AB<- allpredicted_A[start:end]-allpredicted_B[start:end]
paste("Average difference between predicted values = ", round(mean(d_AB),2))
```

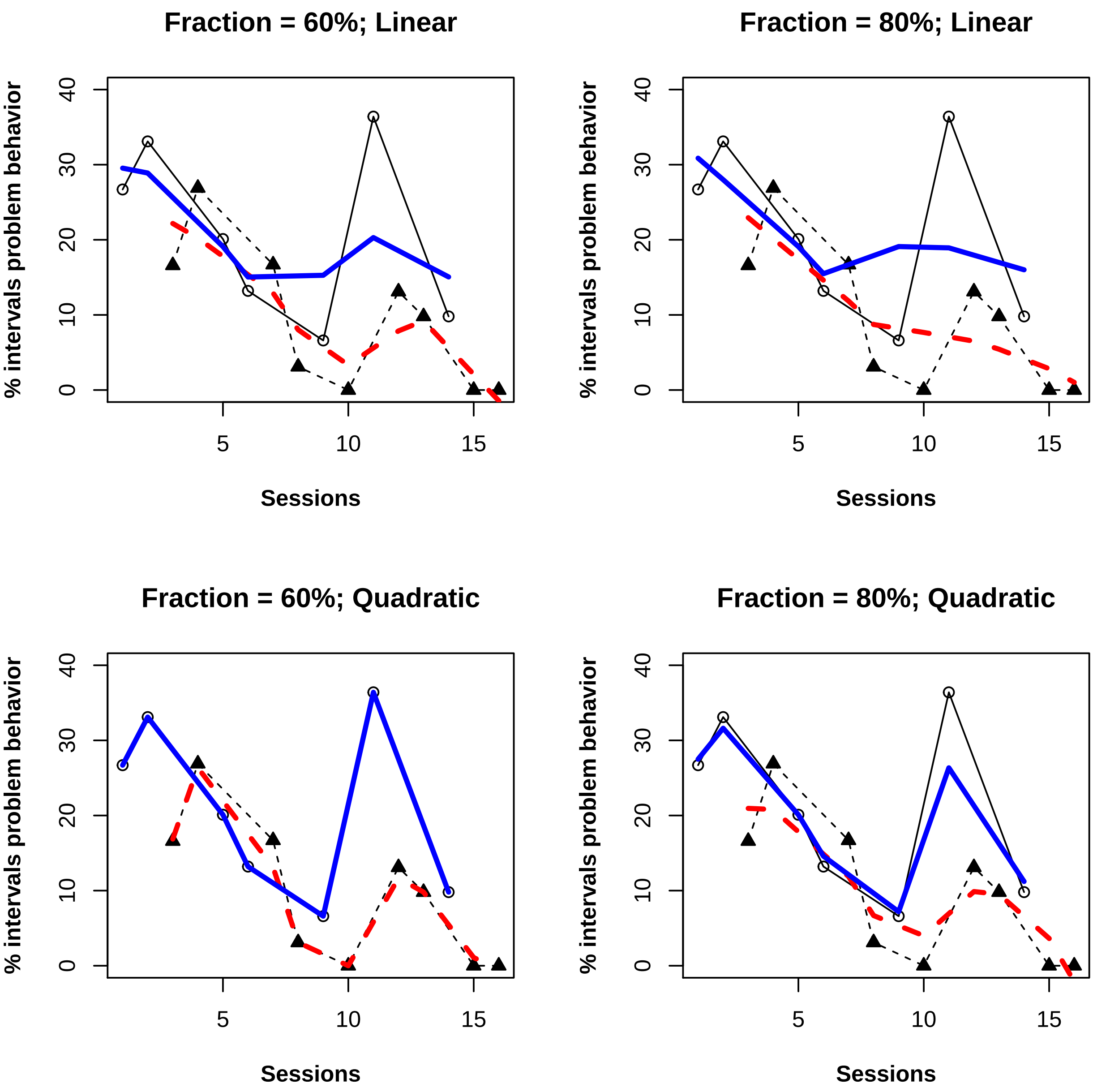
Code ends here. Default output of the code below:

```
## [1] "Pseudo R-squared for A =  0.52"
## [1] "Pseudo R-squared for B =  0.78"
## [1] "Average difference between predicted values =  6.76"
```

The following code allows obtaining the graphical representation of what is being compared

```
par(mfrow=c(2,2))
plot(indep,score, xlim=c(indep[1],indep[length(indep)]),
     ylim=c(0,40), xlab="Sessions", ylab="% intervals problem behavior",
     font.lab=2, pch= " ")
points(indep[condi=="B"], score[condi=="B"], pch=24, bg="black")
points(indep[condi=="A"], score[condi=="A"], bg="black")
lines(indep[condi=="B"],score[condi=="B"],lty="dashed")
lines(indep[condi=="A"],score[condi=="A"],lty="solid")
title (main="Fraction = 60%; Linear")
j <- order(time_B)
lines(time_B[j],reg_B1$fitted[j],col="red",lty="dashed",lwd=3)
j <- order(time_A)
lines(time_A[j],reg_A1$fitted[j],col="blue",lwd=3)

plot(indep,score, xlim=c(indep[1],indep[length(indep)]),
     ylim=c(0,40), xlab="Sessions", ylab="% intervals problem behavior",
     font.lab=2, pch= " ")
points(indep[condi=="B"], score[condi=="B"], pch=24, bg="black")
points(indep[condi=="A"], score[condi=="A"], bg="black")
lines(indep[condi=="B"],score[condi=="B"],lty="dashed")
lines(indep[condi=="A"],score[condi=="A"],lty="solid")
title (main="Fraction = 80%; Linear")
j <- order(time_B)
lines(time_B[j],reg_B2$fitted[j],col="red",lty="dashed",lwd=3)
j <- order(time_A)
lines(time_A[j],reg_A2$fitted[j],col="blue",lwd=3)

plot(indep,score, xlim=c(indep[1],indep[length(indep)]),
     ylim=c(0,40), xlab="Sessions", ylab="% intervals problem behavior",
     font.lab=2, pch= " ")
points(indep[condi=="B"], score[condi=="B"], pch=24, bg="black")
points(indep[condi=="A"], score[condi=="A"], bg="black")
lines(indep[condi=="B"],score[condi=="B"],lty="dashed")
lines(indep[condi=="A"],score[condi=="A"],lty="solid")
title (main="Fraction = 60%; Quadratic")
j <- order(time_B)
lines(time_B[j],reg_B1.q$fitted[j],col="red",lty="dashed",lwd=3)
j <- order(time_A)
lines(time_A[j],reg_A1.q$fitted[j],col="blue",lwd=3)

plot(indep,score, xlim=c(indep[1],indep[length(indep)]),
     ylim=c(0,40), xlab="Sessions", ylab="% intervals problem behavior",
     font.lab=2, pch= " ")
points(indep[condi=="B"], score[condi=="B"], pch=24, bg="black")
points(indep[condi=="A"], score[condi=="A"], bg="black")
lines(indep[condi=="B"],score[condi=="B"],lty="dashed")
lines(indep[condi=="A"],score[condi=="A"],lty="solid")
title (main="Fraction = 80%; Quadratic")
j <- order(time_B)
lines(time_B[j],reg_B2.q$fitted[j],col="red",lty="dashed",lwd=3)
j <- order(time_A)
lines(time_A[j],reg_A2.q$fitted[j],col="blue",lwd=3)
```

The default graphical output is:



3.8 Piecewise regression

To use this code, it is necessary to first create a data matrix such as the one shown next. The data are then located via the first line of code. In case the same names for the columns are used as shown here, the code only needs to be copied and pasted without further modifications.

Data matrix for immediate effect: difference between the initial predicted values for the two conditions.

Time	Score	condi_B	condi_A	Time1	Time2	condiA_T2	condiB_T2
1	26.7	0	1	0	-2	0	0
2	33.1	0	1	1	-1	1	0
3	16.6	1	0	2	0	0	0
4	26.9	1	0	3	1	0	1
5	20.1	0	1	4	2	4	0
6	13.2	0	1	5	3	5	0
7	16.7	1	0	6	4	0	4
8	3.1	1	0	7	5	0	5
9	6.6	0	1	8	6	8	0
10	0	1	0	9	7	0	7
11	36.4	0	1	10	8	10	0
12	13.1	1	0	11	9	0	9
13	9.8	1	0	12	10	0	10
14	9.8	0	1	13	11	13	0
15	0	1	0	14	12	0	12
16	0	1	0	15	13	0	13

Data matrix for the difference in conditions (between predicted values) at the last measurement occasion.

Time	Score	condi_B	condi_A	Time2	condiA_T2	condiB_T2
1	26.7	0	1	-15	-15	0
2	33.1	0	1	-14	-14	0
3	16.6	1	0	-13	0	-13
4	26.9	1	0	-12	0	-12
5	20.1	0	1	-11	-11	0
6	13.2	0	1	-10	-10	0
7	16.7	1	0	-9	0	-9
8	3.1	1	0	-8	0	-8
9	6.6	0	1	-7	-7	0
10	0	1	0	-6	0	-6
11	36.4	0	1	-5	-5	0
12	13.1	1	0	-4	0	-4
13	9.8	1	0	-3	0	-3
14	9.8	0	1	-2	-2	0
15	0	1	0	-1	0	-1
16	0	1	0	0	0	0

```
# The already existing data matrix (e.g., a tab-delimited .txt) is located
```

```
Piecewise <- read.table(file.choose(),header=TRUE)
```

```
# Copy and paste the rest of the code in the R console
```

```
reg<- lm(Score ~ condi_B + condiA_T2 + condiB_T2, Piecewise)
summary(reg)

res<-  sqrt(sum((residuals(reg)^2))/df.residual(reg))
print("Square root of the mean square error"); print(res)
```

Code ends here. Default output of the code below:

Comparing predicted values for the last measurement occasion:

```
##
## Call:
## lm(formula = Score ~ condi_B + condiA_T2 + condiB_T2, data = Piecewise)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -12.164  -5.202  -0.934   5.537  19.577
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  11.9712     8.0581   1.486   0.1632
## condi_B     -11.7924     9.6750  -1.219   0.2463
## condiA_T2    -0.9703     0.7945  -1.221   0.2454
## condiB_T2    -1.5106     0.7045  -2.144   0.0532 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.227 on 12 degrees of freedom
## Multiple R-squared:  0.4991,Adjusted R-squared:  0.3739
## F-statistic: 3.986 on 3 and 12 DF,  p-value: 0.03493
```

```
## [1] "Square root of the mean square error"
## [1] 9.226883
```

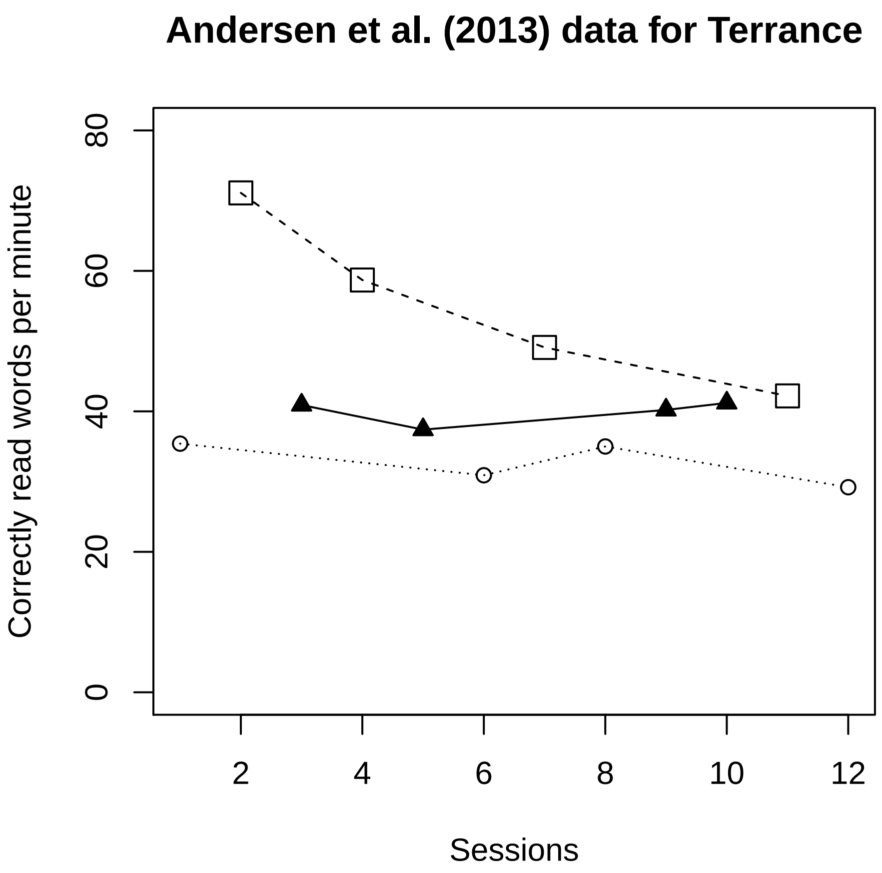
4 Illustrations with real data

4.1 Andersen et al. (2013) data

Illustration 1 contains the data gathered by Andersen et al.(2013) for Terrance [their figure 1, middle right panel].

Andersen, M. N., Daly, III, A. J., & Young, N. D. (2013). Examination of a one-trial brief experimental analysis to identify reading fluency interventions. *Psychology in the Schools*, *50*, 403–414.

```
score <- c(35.4,71.1,40.9,58.7,37.4,30.9,49.1,35,40.2,41.2,42.2,29.2)
condi <- c(1,3,2,3,2,1,3,1,2,2,3,1)
```

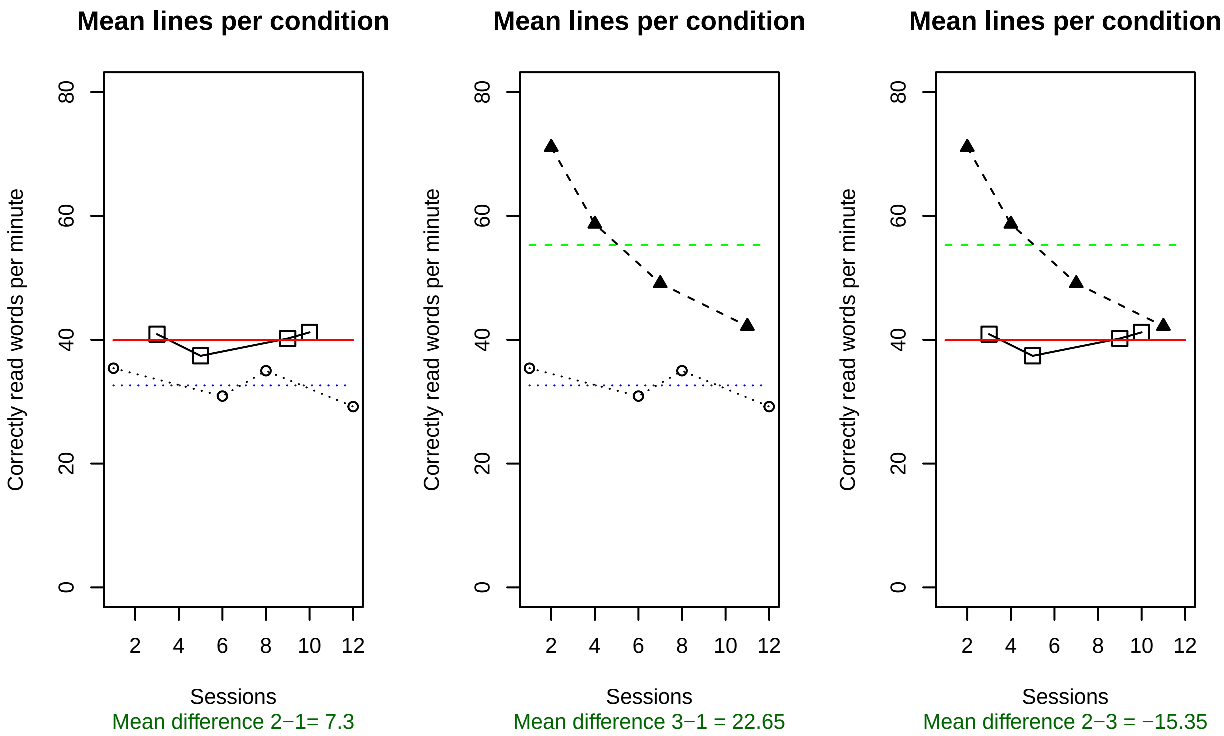


A summary of the results follows on the next page.

Conditions: 1 - ○; 2 - ▲; 3 - □

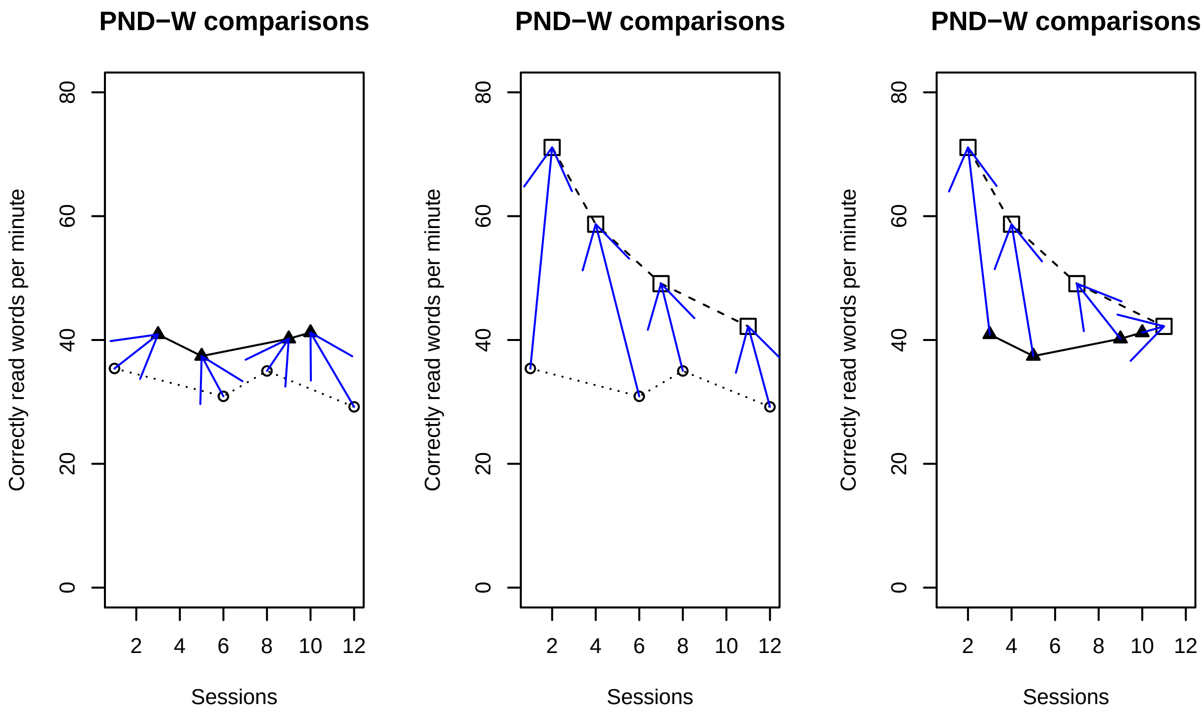
Analytical technique	Quantifications
Mean difference	▲−○= 7.30 □−○ = 22.65 □−▲ = 15.35
Percentage of nonoverlapping data according to the proposal by Wolery, Gast, et al. (2010).	▲ 100.00% superior to ○ □ 100.00% superior to ○ □ 100.00% superior to ▲
Nonovelap of all pairs	▲−○ = 100% □−○ = 100% □−▲ = 100%
Tau-U: only the direction and amount of monotonic trend is represented here, not including overlap	○ monotonic trend = 66.67% decrease ▲ monotonic trend = 33.33% increase □ monotonic trend = 100% decrease
Piecewise regression, comparing at the last intervention measurement occasion	▲−○ last point: 6.65 □−○ last point: 10.67 □−▲last point: −4.02 ○ slope = −0.47 ▲ slope = −3.08 □ slope = 0.17
Average difference between the values fitted via linear regression fitted lines	▲−○ = 7.14 □−○ = 20.99 □−▲ = 13.85
Average difference between the values fitted via quadratic regression fitted lines	▲−○ = 5.93 □−○ = 19.63 □−▲= 12.64
Average difference between the values fitted via local regression (LOESS) with fraction α = 0.60 and straight lines fitted	▲−○ = 6.58 □−○ = 20.30 □−▲ = 12.78
ALIV: A comparison involving actual and linearly interpolated values	▲−○ = 6.58 □−○ = 20.30 □−▲ = 12.78
ADISO: Weighted average difference between successive observations ADISO-O: percentage of ordinal superiority	▲−○ = 7.30 (100% superior) □−○ = 22.65 (100% superior) □−▲ = 15.35 (100% superior) ▲ 100% superior to ○ □ 100% superior to ○ □ 100% superior to ▲

4.1.1 Mean difference



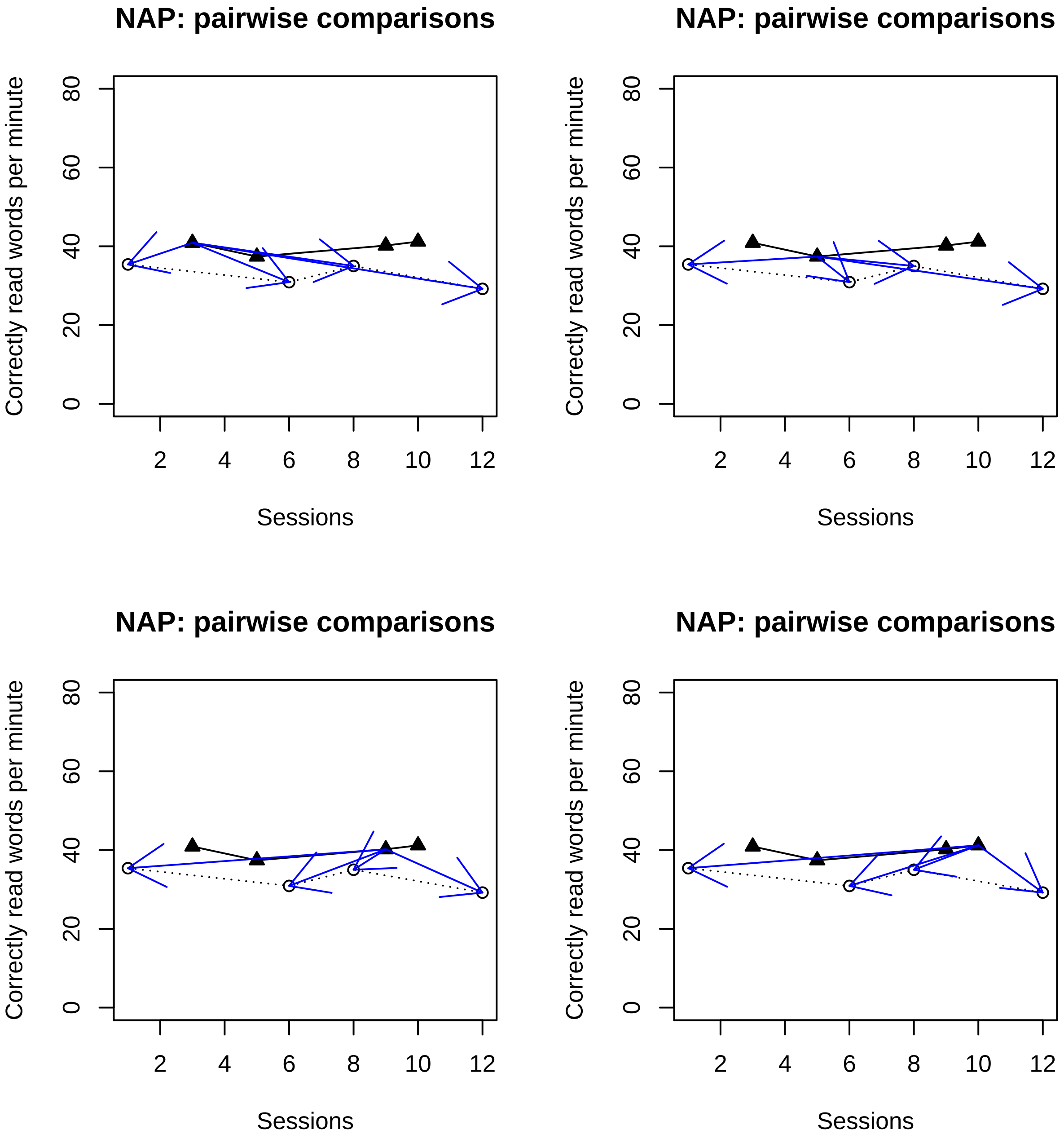
4.1.2 Percentage of nonoverlapping data: Wolery et al. version

```
##      Comparison PND-W
## [1,] "2 > 1"    "100 %"
## [2,] "3 > 1"    "100 %"
## [3,] "2 > 3"    "0 %"
```



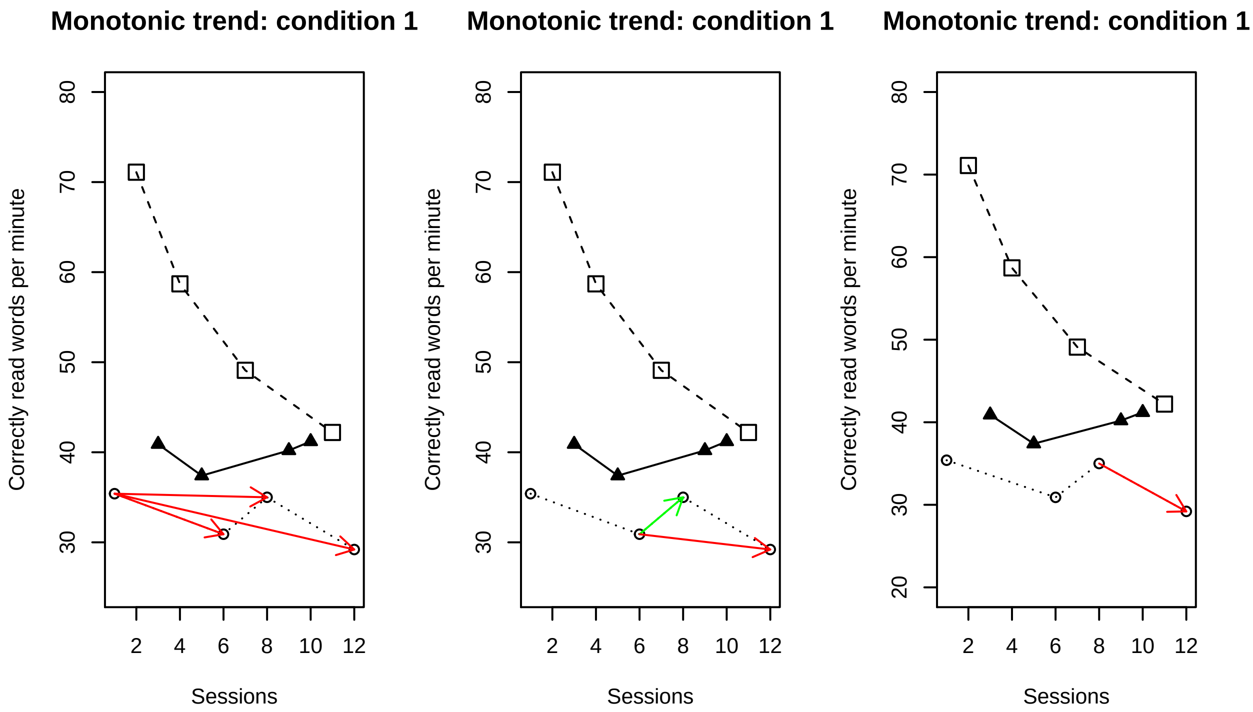
4.1.3 Nonoverlap of all pairs

```
##       Comparison NAP
## [1,] "2 > 1"    "1"
## [2,] "3 > 1"    "1"
## [3,] "2 > 3"    "1"
```



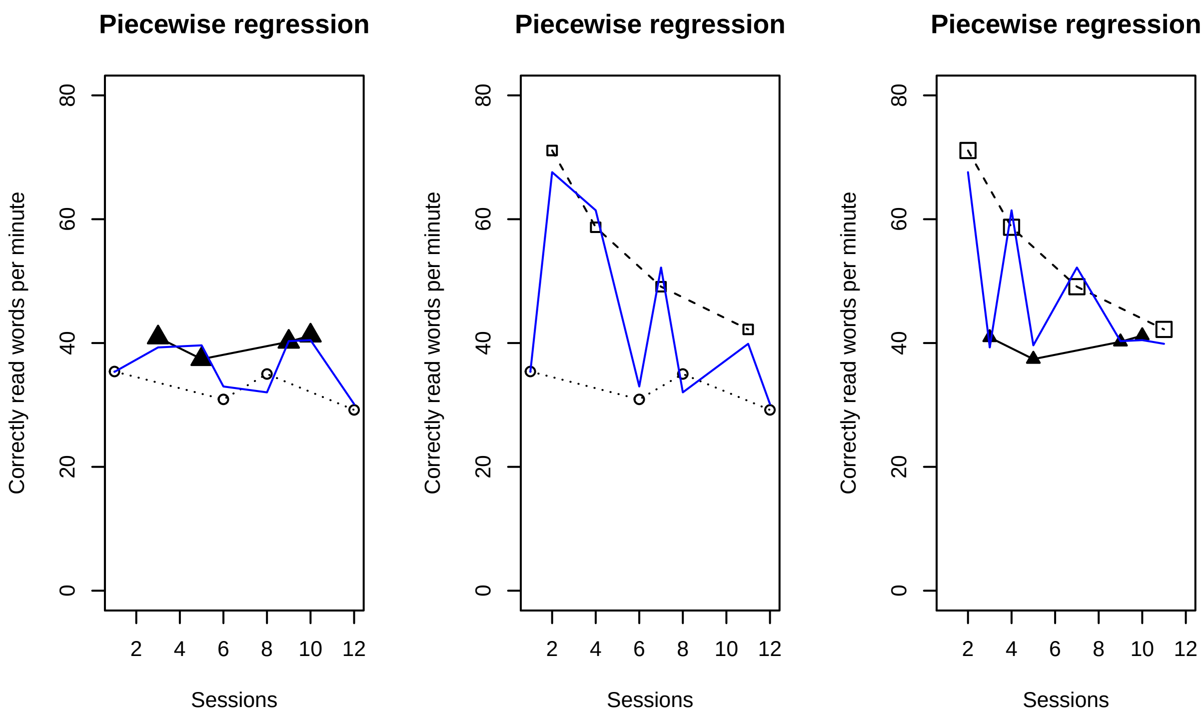
4.1.4 Monotonic trends estimated as in Tau-U

```
## [1] "Condition 1"
## [1] "Negative monotonic trend: 66.67 % decrease"
## [1] "Condition 2"
## [1] "Positive monotonic trend: 33.33 % increase"
## [1] "Condition 3"
## [1] "Negative monotonic trend: 100 % decrease"
```



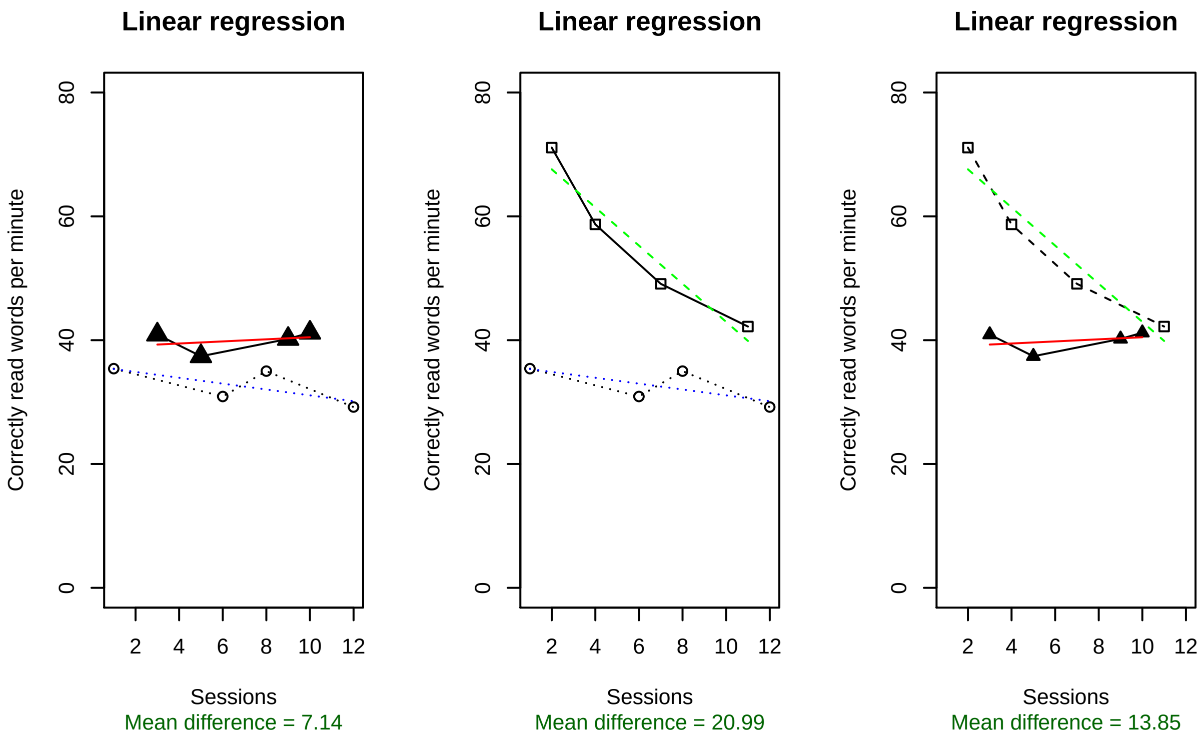
4.1.5 Piecewise regression, comparing at the last measurement occasion

```
## [1] "Comparison 2 vs. 1"
##      Time Score condi_B condi_A Time2 condiA_T2 condiB_T2
## [1,]    1  35.4       1       0   -11       -11         0
## [2,]    3  40.9       0       1    -9         0        -9
## [3,]    5  37.4       0       1    -7         0        -7
## [4,]    6  30.9       1       0    -6        -6         0
## [5,]    8  35.0       1       0    -4        -4         0
## [6,]    9  40.2       0       1    -3         0        -3
## [7,]   10  41.2       0       1    -2         0        -2
## [8,]   12  29.2       1       0     0         0         0
## [1] "  "
## [1] "Comparison 3 vs. 1"
##      Time Score condi_B condi_A Time2 condiA_T2 condiB_T2
## [1,]    1  35.4       1       0   -11       -11         0
## [2,]    2  71.1       0       1   -10         0       -10
## [3,]    4  58.7       0       1    -8         0        -8
## [4,]    6  30.9       1       0    -6        -6         0
## [5,]    7  49.1       0       1    -5         0        -5
## [6,]    8  35.0       1       0    -4        -4         0
## [7,]   11  42.2       0       1    -1         0        -1
## [8,]   12  29.2       1       0     0         0         0
## [1] "  "
## [1] "Comparison 2 vs. 3"
##      Time Score condi_B condi_A Time2 condiA_T2 condiB_T2
## [1,]    2  71.1       0       1   -10         0       -10
## [2,]    3  40.9       1       0    -9        -9         0
## [3,]    4  58.7       0       1    -8         0        -8
## [4,]    5  37.4       1       0    -7        -7         0
## [5,]    7  49.1       0       1    -5         0        -5
## [6,]    9  40.2       1       0    -3        -3         0
## [7,]   10  41.2       1       0    -2        -2         0
## [8,]   11  42.2       0       1    -1         0        -1
```

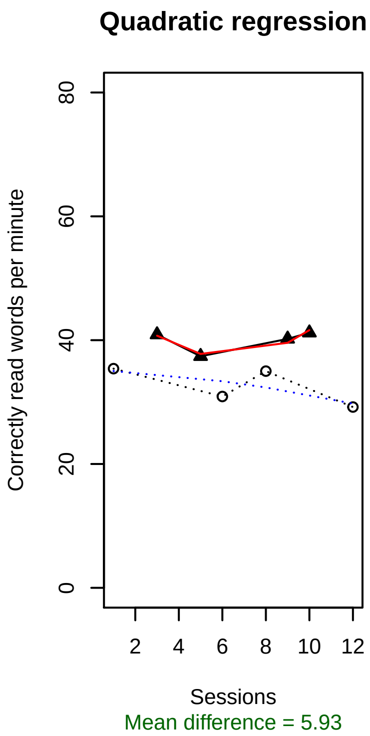


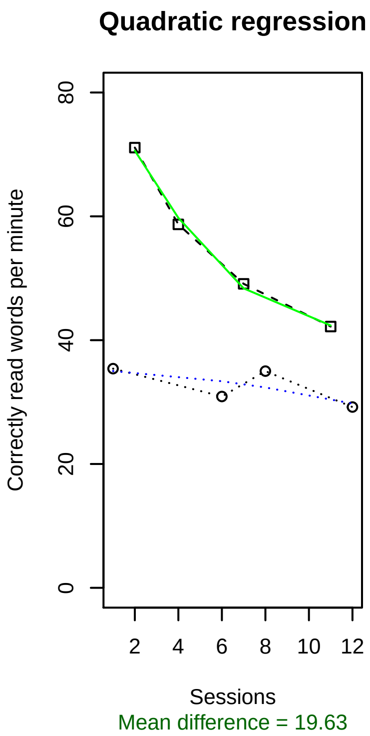
```
##      Conditions Difference at last point
## [1,] "2-1"      "-10.67"
## [2,] "3-1"      "-6.65"
## [3,] "2-3"      "4.02"
##      Condition Trend
## [1,]         1 -0.47
## [2,]         2  0.17
## [3,]         3 -3.08
```

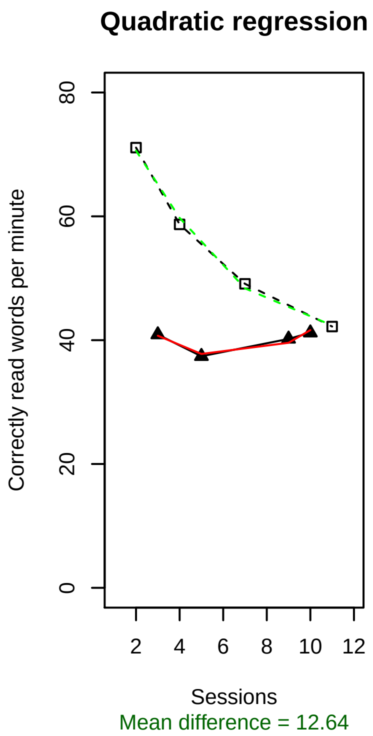
4.1.6 Linear regression: mean difference between fitted values



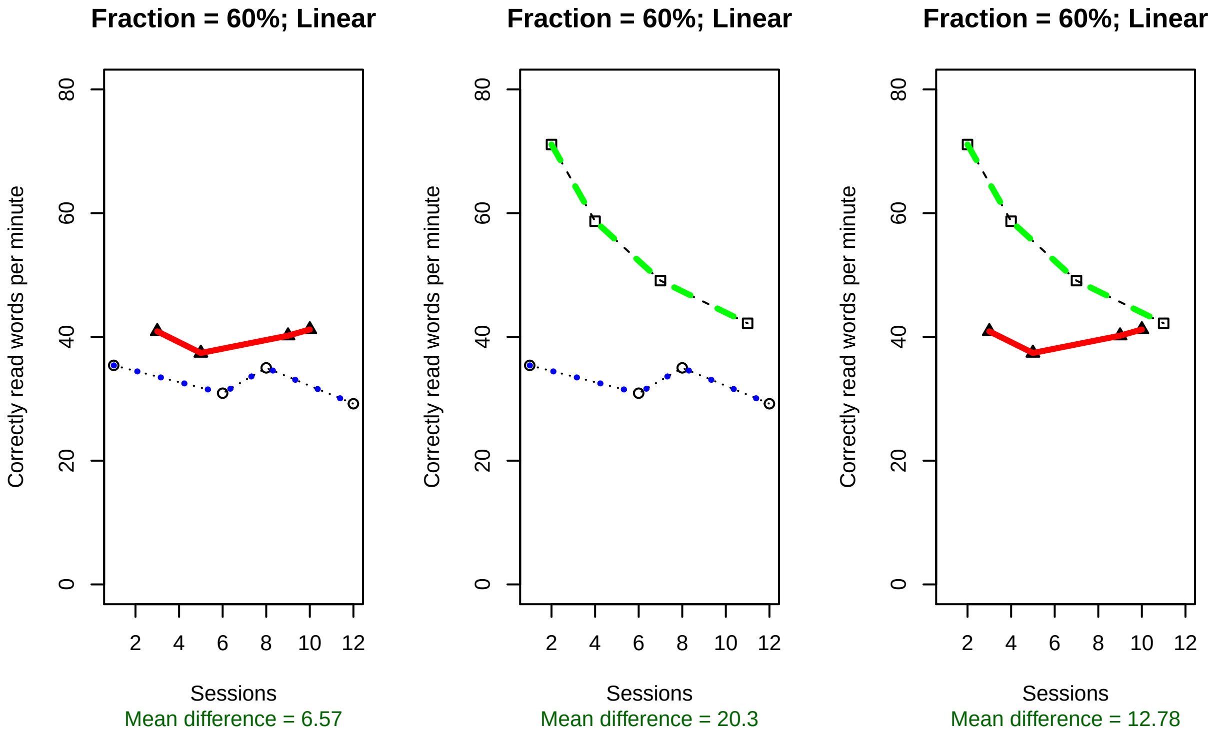
4.1.7 Quadratic regression: mean difference between fitted values



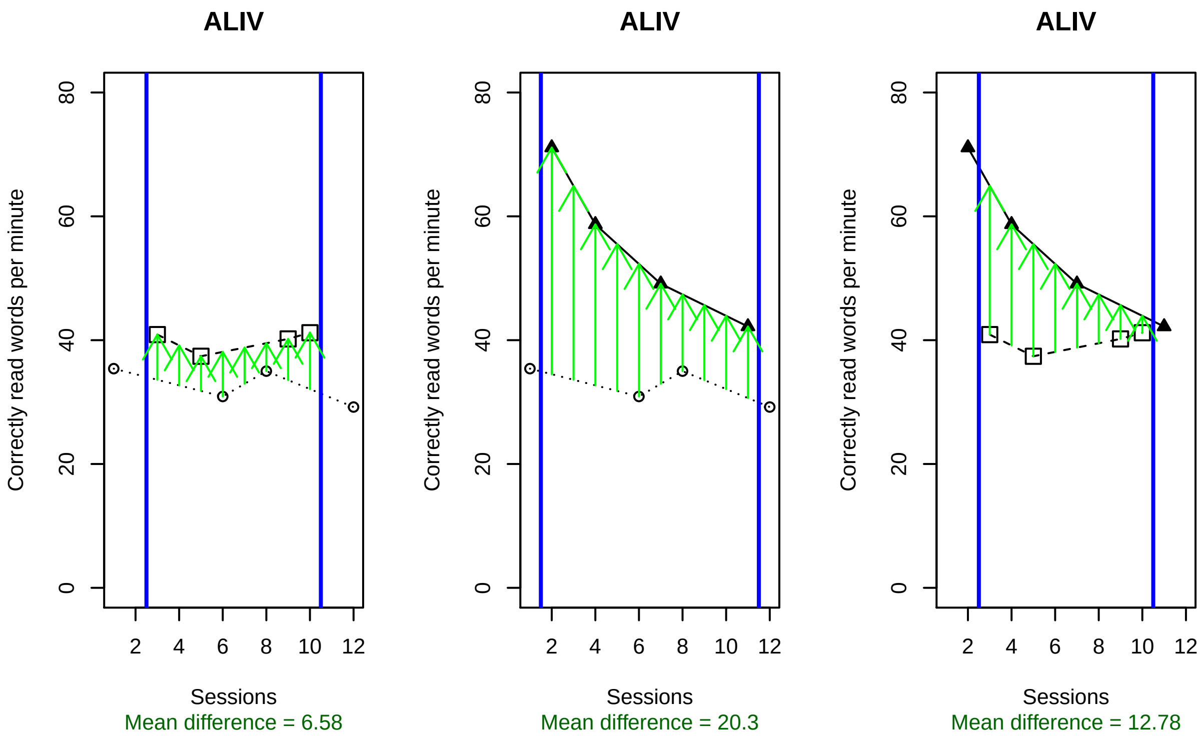




4.1.8 Local regression (LOESS): mean difference between fitted values

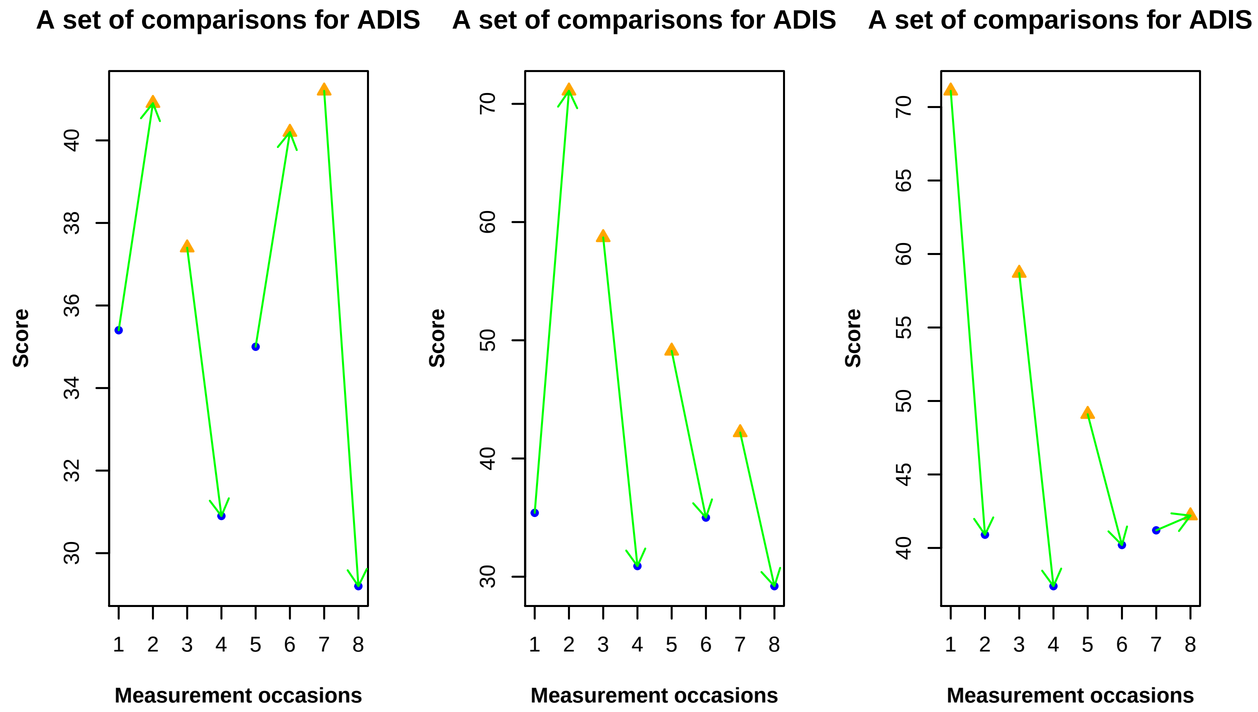


4.1.9 Actual and linearly interpolated values: ALIV



4.1.10 Weighted average difference between successive observations: ADISO

```
## Error in `[<-`(`*tmp*`, 2, 4, value = "1 superior:  0 %.  3 superior:  100 %"):  subscript
out of bounds
## Error in `[<-`(`*tmp*`, 3, 4, value = "2 superior:  0 %.  3 superior:  100 %"):  subscript
out of bounds
```



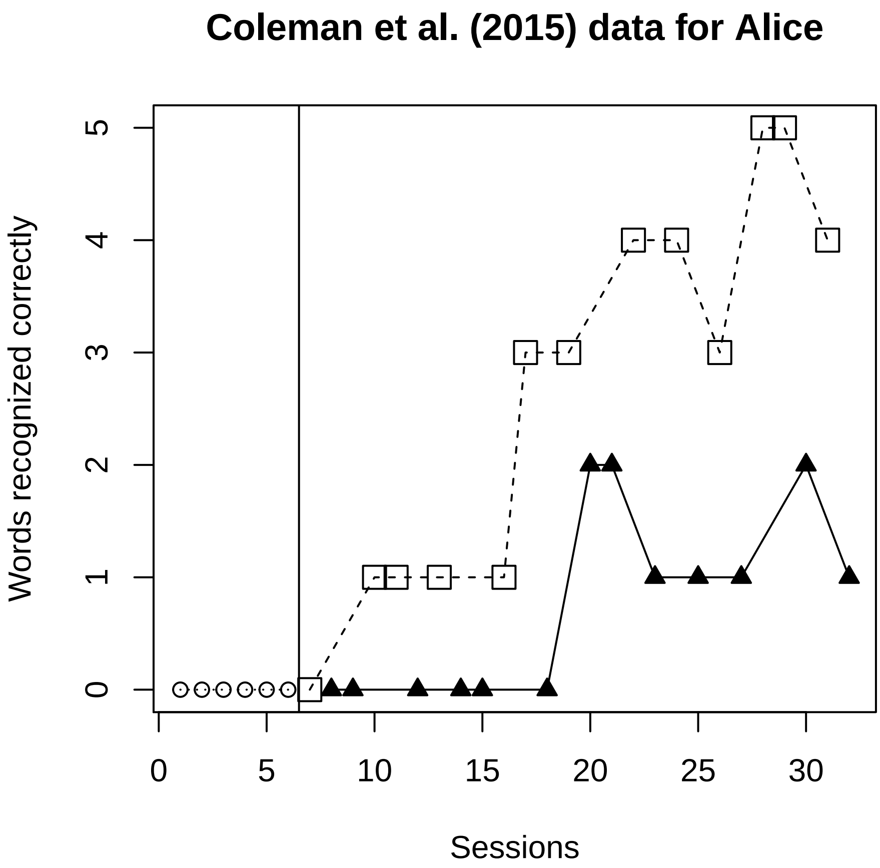

```
##      Comparison ADISO    Superiority
## [1,] "1-2"      "-7.3"   "1 superior:  0 %. 2 superior:  100 %"
## [2,] "1-3"      "-22.65" "-22.65"
## [3,] "2-3"      "-15.35" "-15.35"
```

4.2 Coleman et al. (2015) data

Illustration 2 contains the data gathered by Coleman et al.(2015) data for Alice [their figure 2].

Coleman, M. B., Cherry, R. A., Moore, T. C., Park, Y., & Cihak, D. F. (2015). Teaching sight words to elementary students with intellectual disability and autism: A comparison of teacher-directed versus computer-assisted simultaneous prompting. *Intellectual and Developmental Disabilities*, *53*, 196−210.

```
score <- c(0,0,0,0,0,0,0,0,0,1,1,0,1,0,0,1,3,0,3,2,2,4,1,4,1,3,1,5,5,2,4,1)
condi <- c(1,1,1,1,1,1,3,2,2,3,3,2,3,2,2,3,3,2,3,2,2,3,2,3,2,3,2,3,3,2,3,2)
```

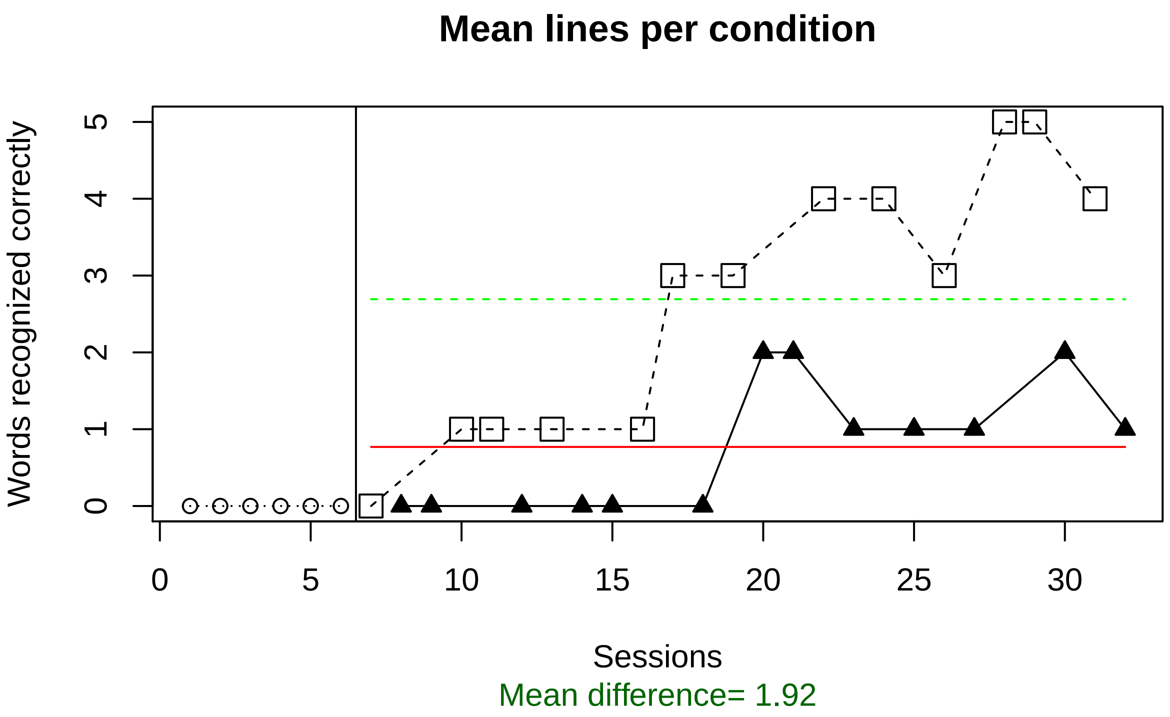


A summary of the results follows on the next page.

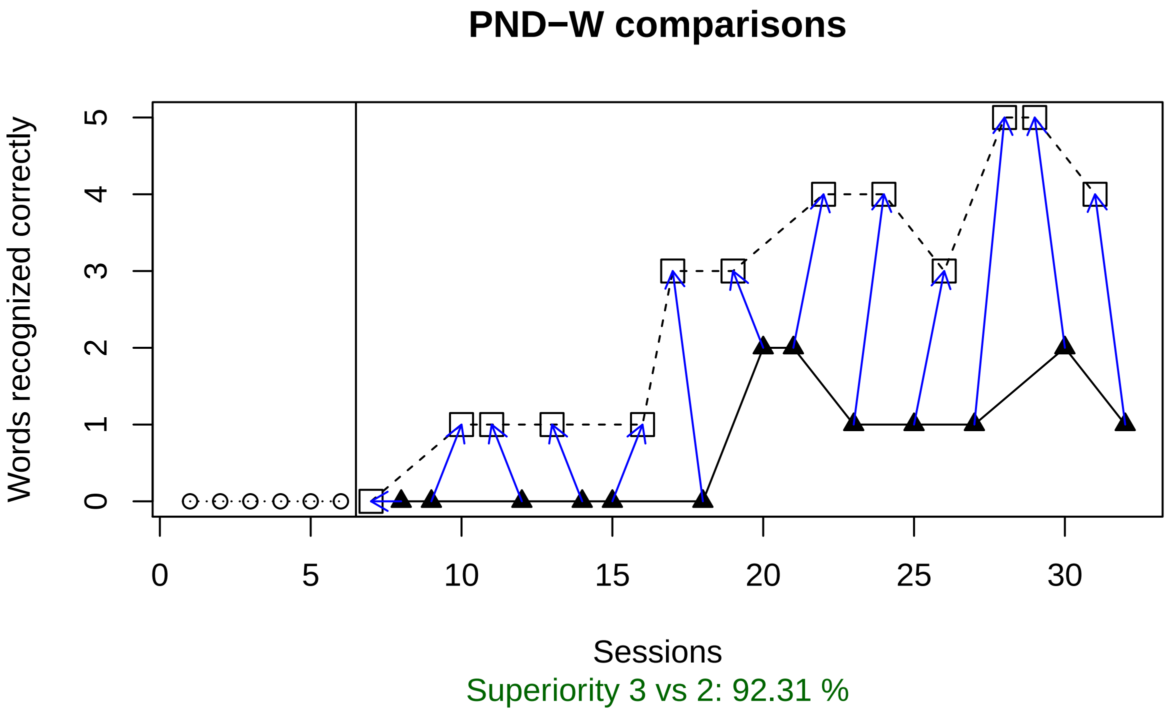
Conditions: 1 - ○ [baseline not considered]; 2 - ▲; 3 - □

Analytical technique	**Quantifications**
Mean difference	□−▲ = 1.92
Percentage of nonoverlapping data according to the proposal by Wolery, Gast, et al. (2010).	□ 92.31% superior to ▲
Nonovelap of all pairs	□−▲ = 82.00%
Tau-U: only the direction and amount of monotonic trend is represented here, not including overlap	▲ monotonic trend = 46.15% increase □ monotonic trend = 73.08% increase
Piecewise regression, comparing at the last intervention measurement occasion	□−▲last point = 3.49 ▲ slope = 0.07 □ slope = 0.20
Average difference between the values fitted via linear regression fitted lines	□−▲ = 1.93
Average difference between the values fitted via quadratic regression fitted lines	□−▲= 1.95
Average difference between the values fitted via LOESS (fraction α = 0.60; straight lines fitted)	□−▲ = 1.97
ALIV: A comparison involving actual and linearly interpolated values	□−▲ = 1.96
ADISO: Weighted average difference between successive observations ADISO-O: percentage of ordinal superiority	□−▲ = 1.77 □ 90.00% superior to ▲

4.2.1 Mean difference

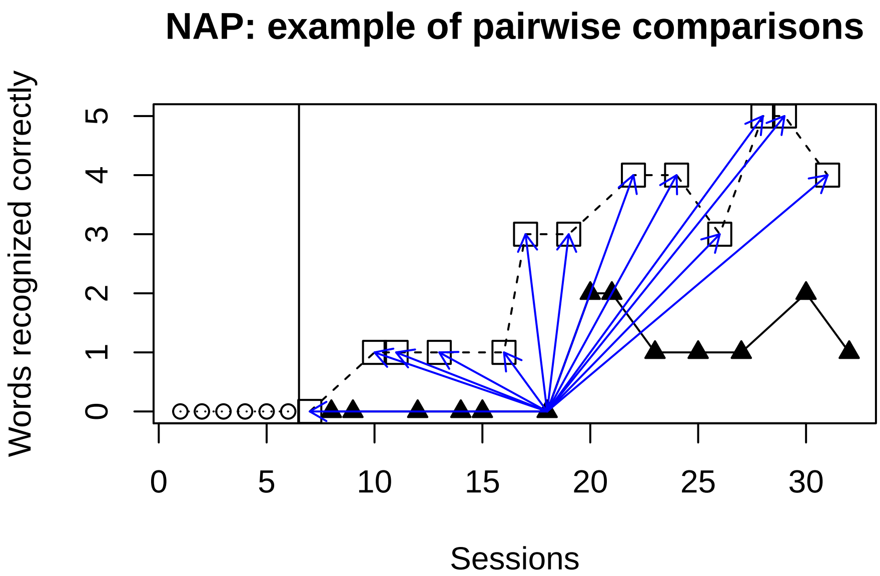


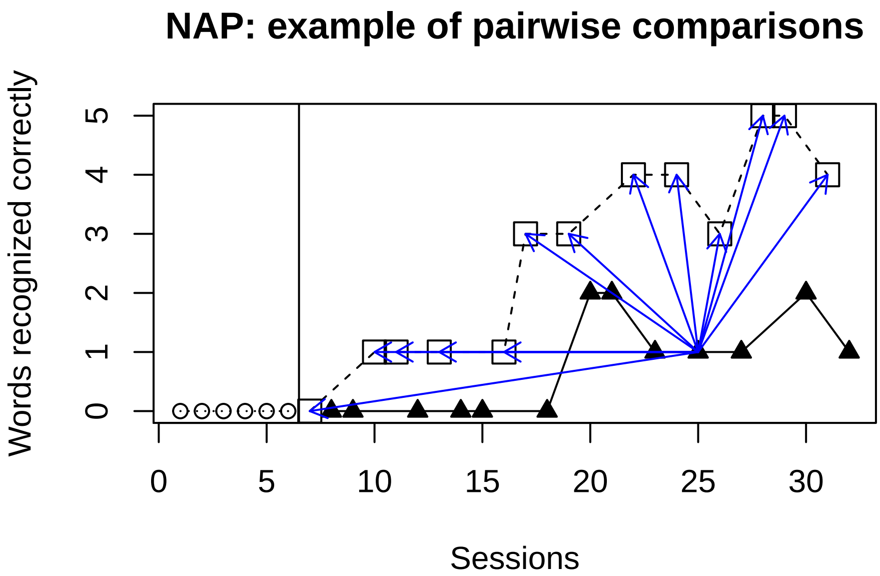
4.2.2 Percentage of nonoverlapping data: Wolery et al. version



4.2.3 Nonoverlap of all pairs

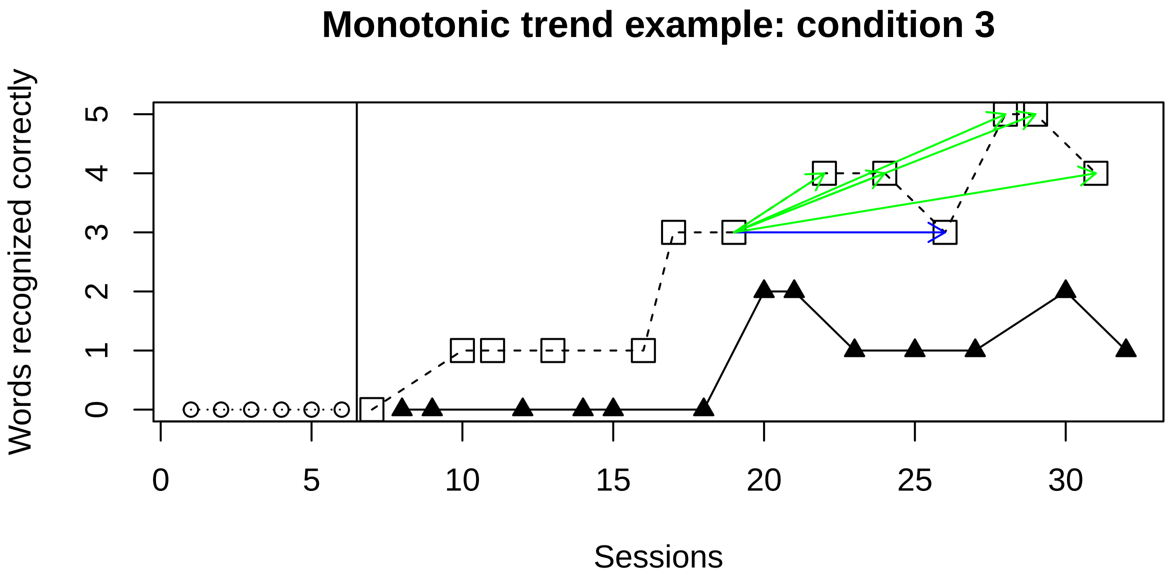
```
## [1] "NAP =  0.82"
```

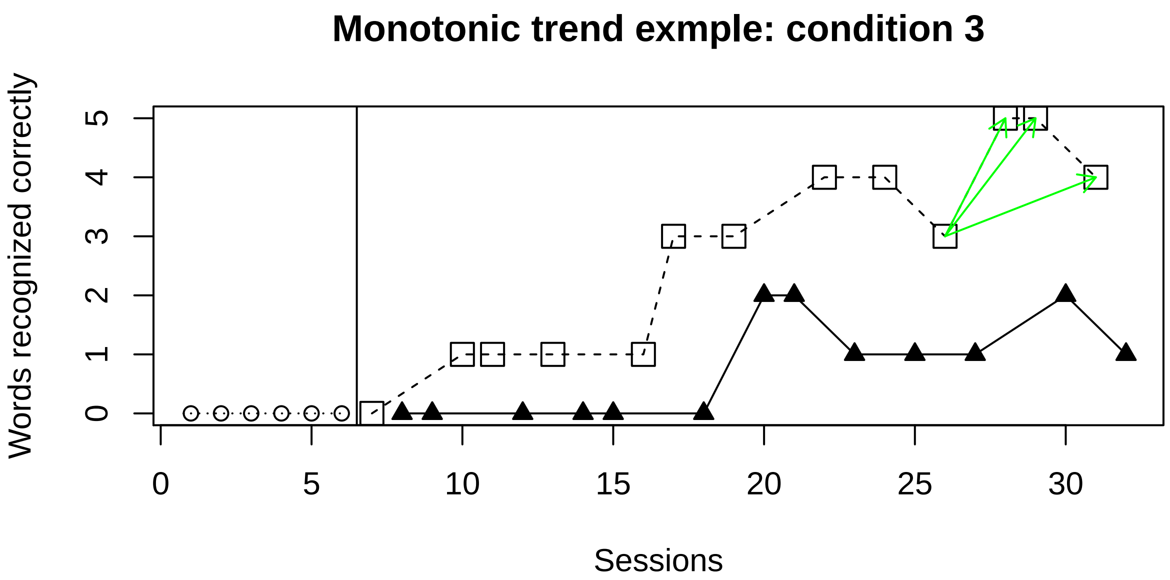




4.2.4 Monotonic trends estimated as in Tau-U

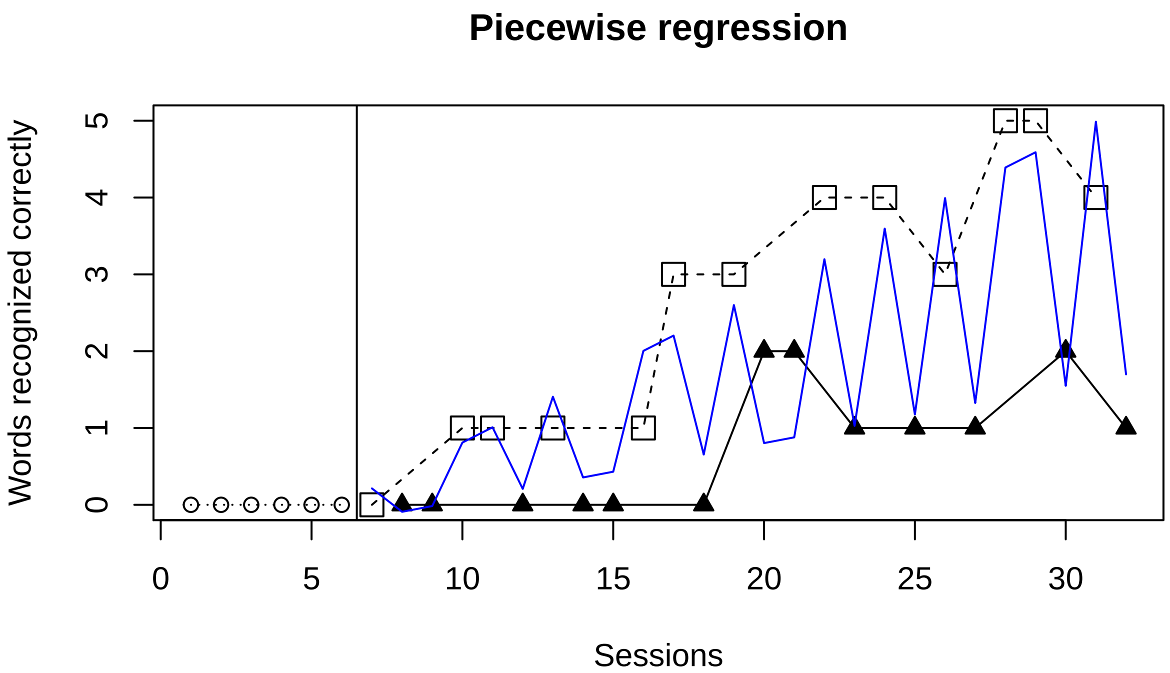
```
## [1] "Condition 2"
## [1] "Positive monotonic trend: 46.15 % increase"
## [1] "Condition 3"
## [1] "Positive monotonic trend: 73.08 % increase"
```





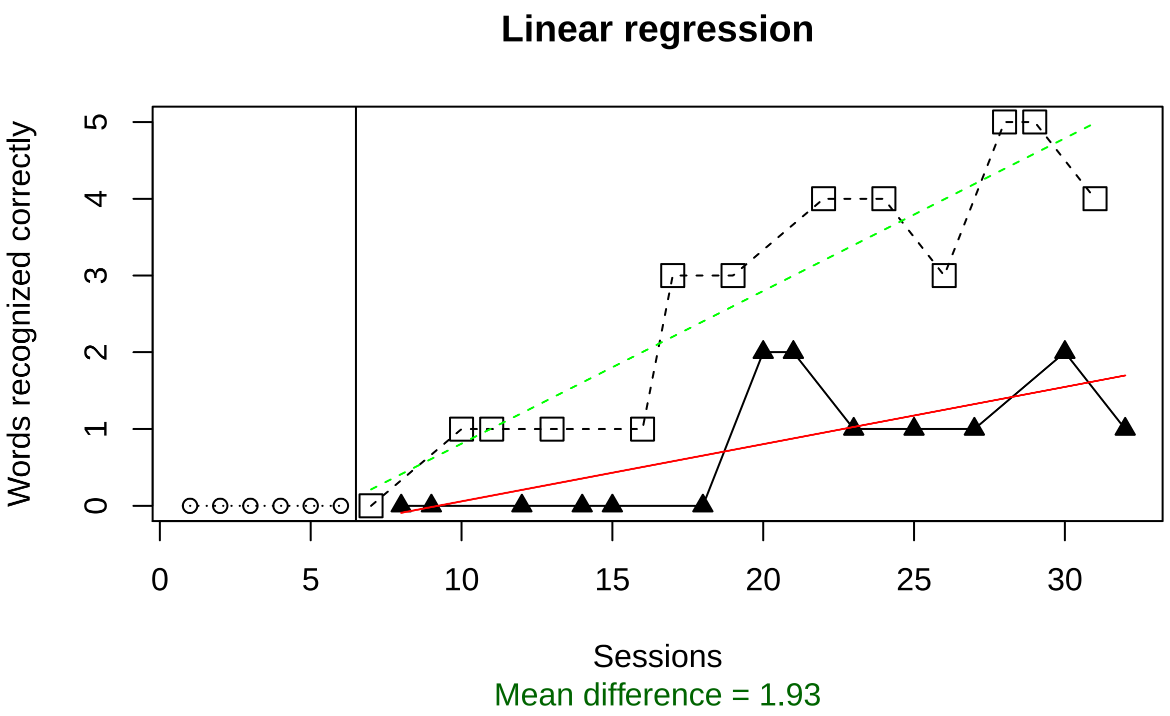
4.2.5 Piecewise regression, comparing at the last measurement occasion

```
## [1] "Comparison 3 vs. 2"
##        Time Score condi_B condi_A Time2 condiA_T2 condiB_T2
##   [1,]    7     0       0       1   -25         0       -25
##   [2,]    8     0       1       0   -24       -24         0
##   [3,]    9     0       1       0   -23       -23         0
##   [4,]   10     1       0       1   -22         0       -22
##   [5,]   11     1       0       1   -21         0       -21
##   [6,]   12     0       1       0   -20       -20         0
##   [7,]   13     1       0       1   -19         0       -19
##   [8,]   14     0       1       0   -18       -18         0
##   [9,]   15     0       1       0   -17       -17         0
## [10,]   16     1       0       1   -16         0       -16
## [11,]   17     3       0       1   -15         0       -15
## [12,]   18     0       1       0   -14       -14         0
## [13,]   19     3       0       1   -13         0       -13
## [14,]   20     2       1       0   -12       -12         0
## [15,]   21     2       1       0   -11       -11         0
## [16,]   22     4       0       1   -10         0       -10
## [17,]   23     1       1       0    -9        -9         0
## [18,]   24     4       0       1    -8         0        -8
## [19,]   25     1       1       0    -7        -7         0
## [20,]   26     3       0       1    -6         0        -6
## [21,]   27     1       1       0    -5        -5         0
## [22,]   28     5       0       1    -4         0        -4
## [23,]   29     5       0       1    -3         0        -3
## [24,]   30     2       1       0    -2        -2         0
## [25,]   31     4       0       1    -1         0        -1
## [26,]   32     1       1       0     0         0         0
## [1] "  "
```

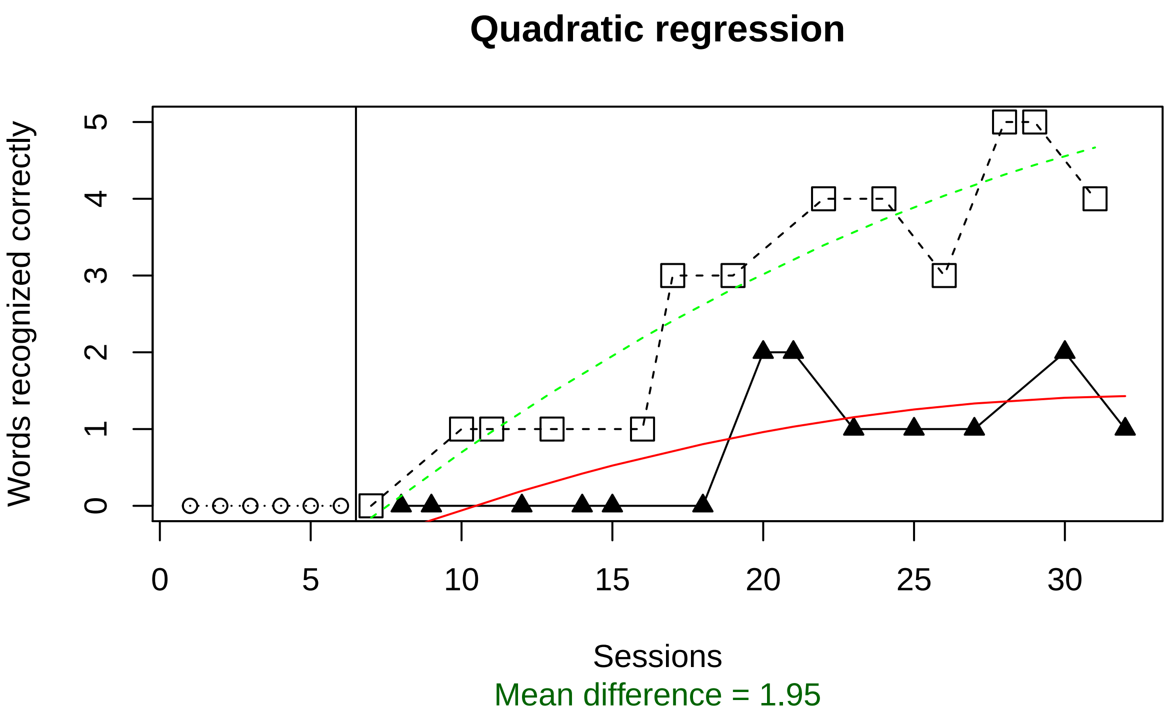



```
## [1] "Condition 2 trend = 0.07"
## [1] "Condition 3 trend = 0.2"
## [1] "Last occasion difference 2-3 = -3.49"
```

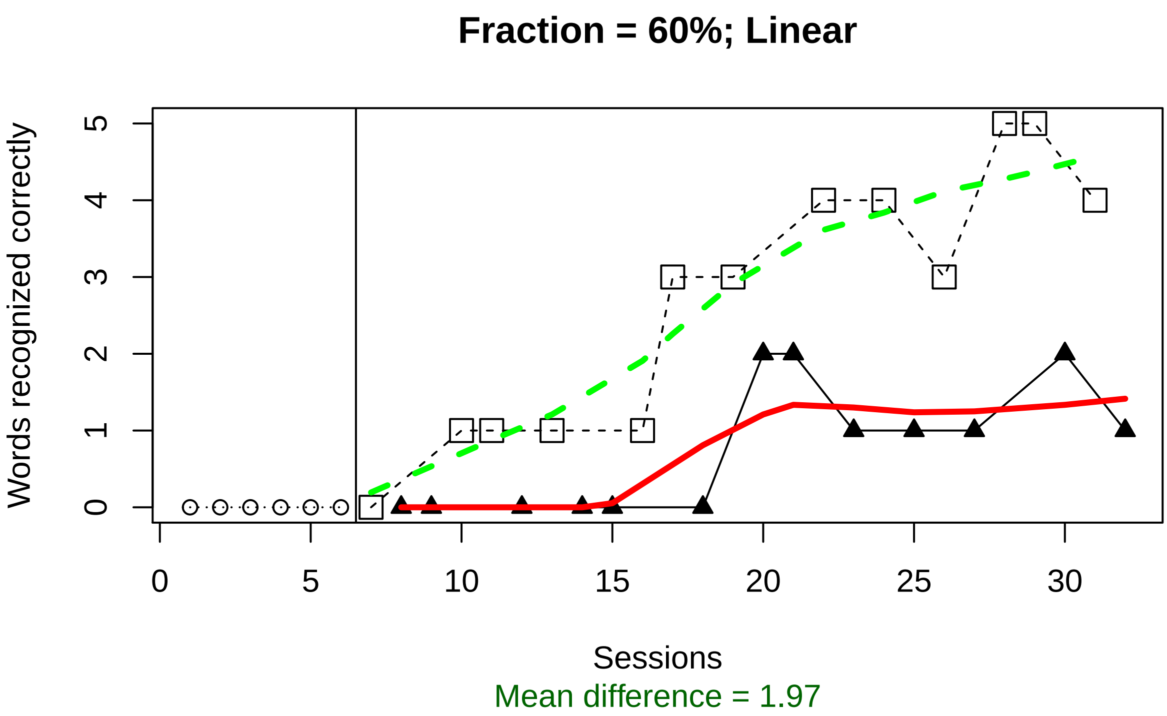
4.2.6 Linear regression: mean difference between fitted values



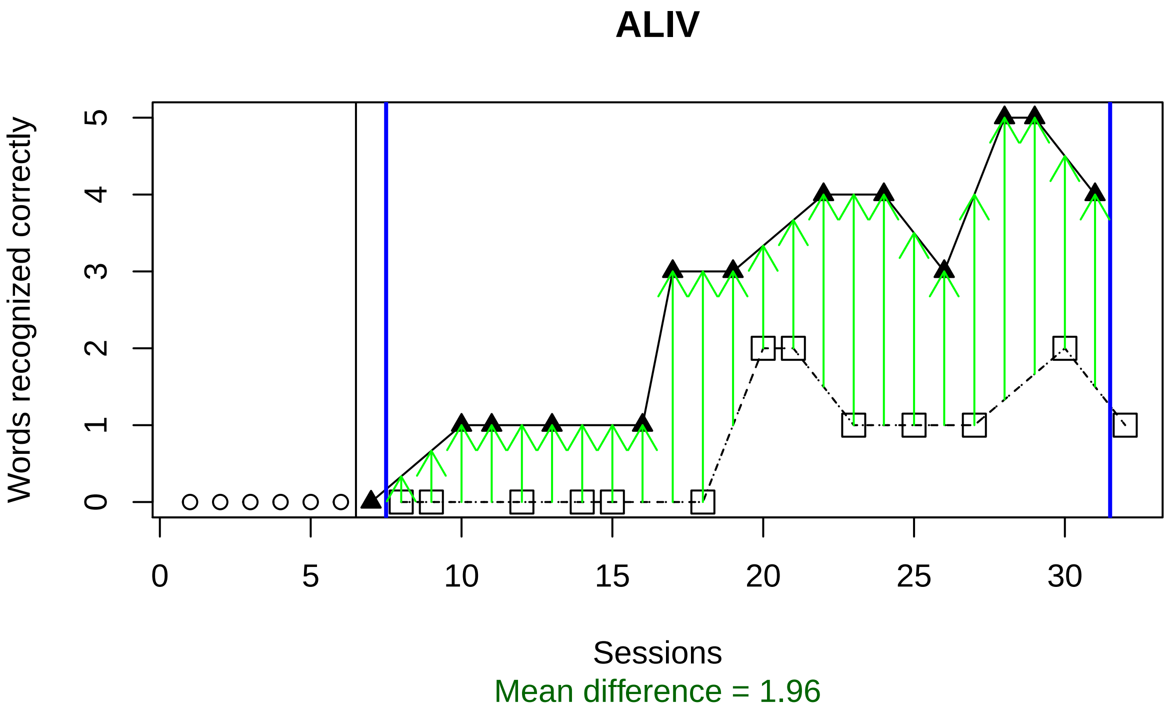
4.2.7 Quadratic regression: mean difference between fitted values



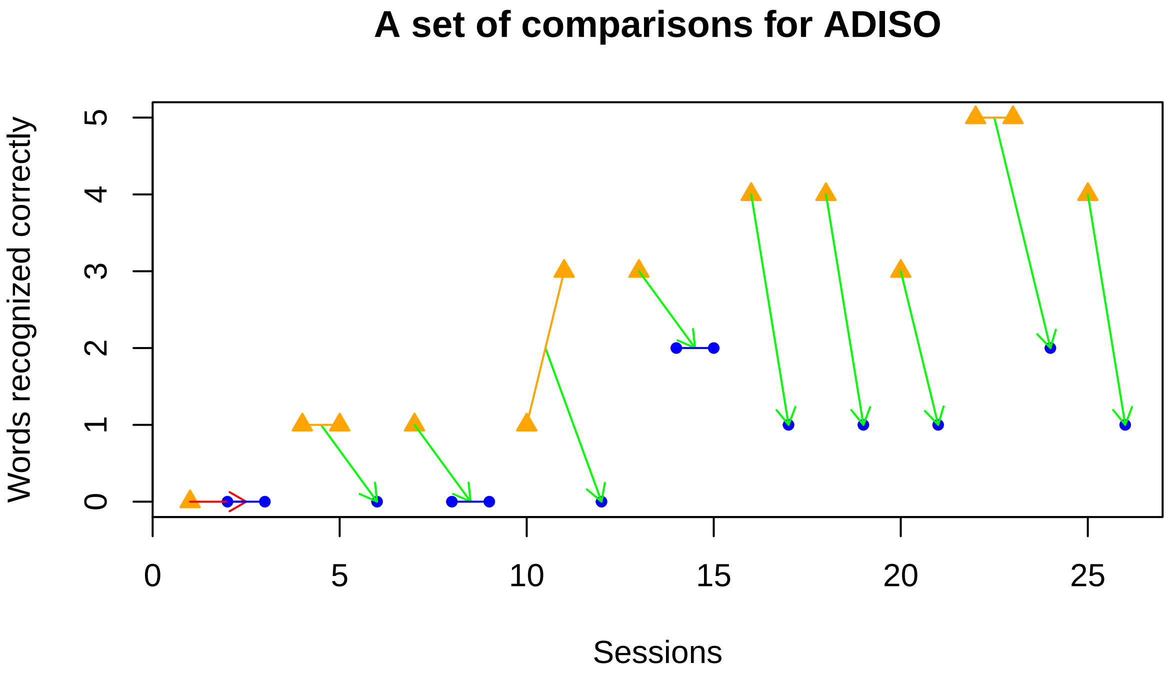
4.2.8 Local regression (LOESS): mean difference between fitted values



4.2.9 Actual and linearly interpolated values: ALIV



4.2.10 Weighted average difference between successive observations: ADISO



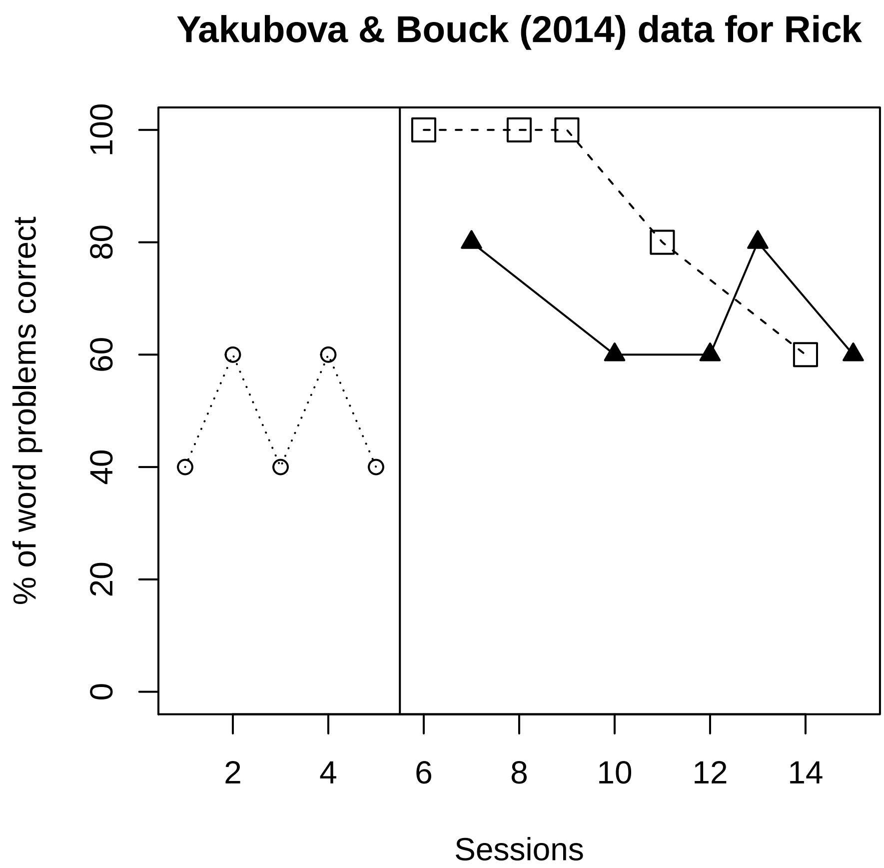

```
##       Comparison ADISO   Superiority
## [1,] "2-3"      "-1.77" "1 superior:  0 %. 2 superior:  90 %"
```

4.3 Yakubova and Bouck (2014) data

Illustration 3 contains the data gathered by Yakubova and Bouck (2014) for Rick [their figure 1; lower panel].

Yakubova, G., & Bouck, E. C. (2014). Not all created equally: Exploring calculator use by students with mild intellectual disability. *Education and Training in Autism and Developmental Disabilities*, *49*, 111−126.

```
score <- c(40,60,40,60,40, 100,80,100,100,60,80,60,80,60,60)
condi <- c(1,1,1,1,1,3,2,3,3,2,3,2,2,3,2)
```

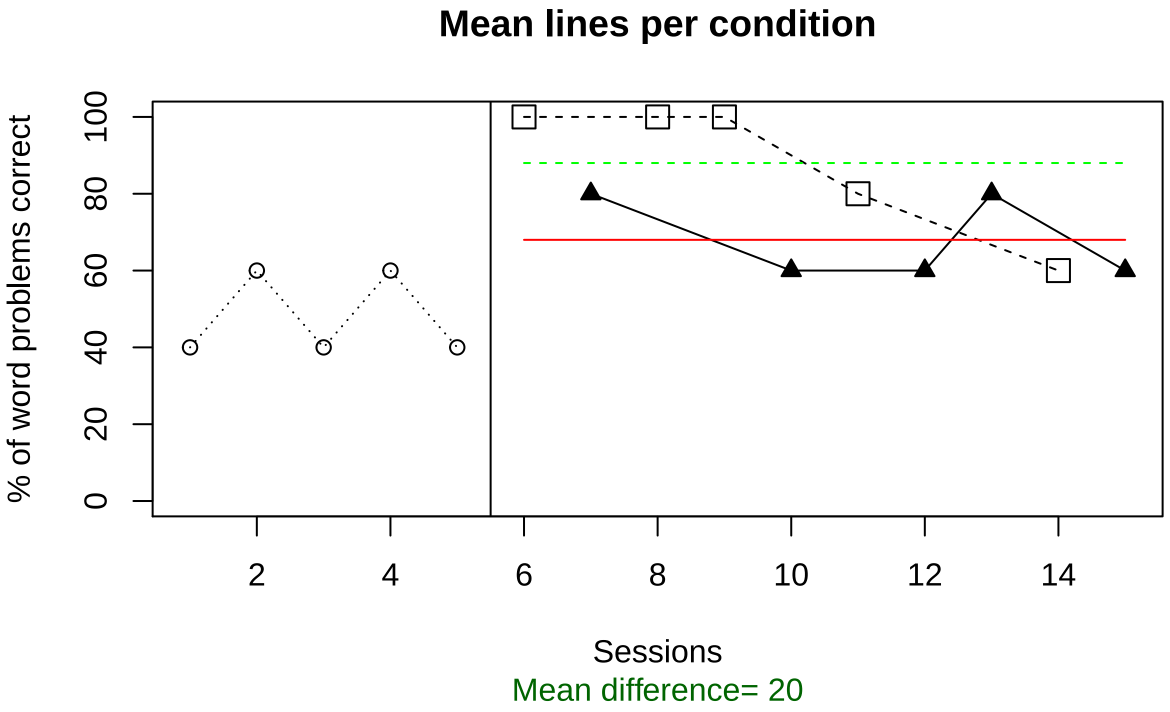


A summary of the results follows on the next page.

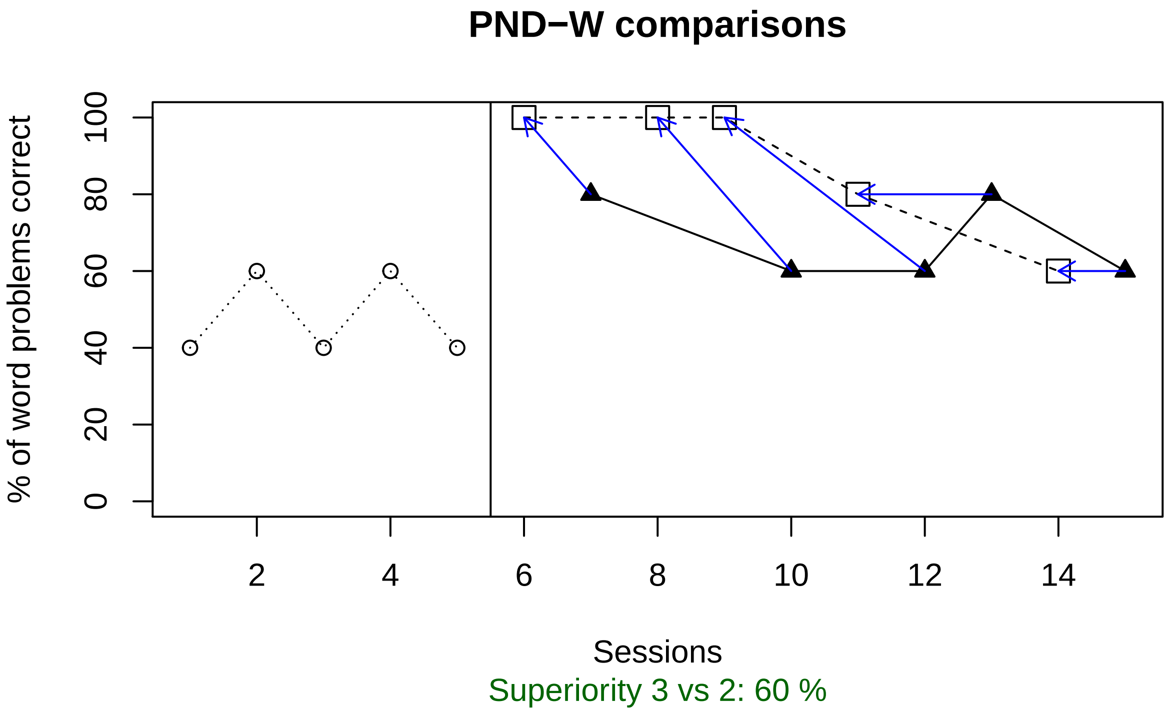
Conditions: 1 - ○ [baseline not considered]; 2 - ▲; 3 - □

Analytical technique	**Quantifications**
Mean difference	□−▲ = 20
Percentage of nonoverlapping data according to the proposal by Wolery, Gast, et al. (2010).	□ 66.00% superior to ▲
Nonovelap of all pairs	□−▲ = 82.00%
Tau-U: only the direction and amount of monotonic trend is represented here, not including overlap	▲ monotonic trend = 20% decrease □ monotonic trend = 70% decrease
Piecewise regression, comparing at the last intervention measurement occasion	□−▲last point = −4.19 ▲ slope = −1.51 □ slope = −5.48
Average difference between the values fitted via linear regression fitted lines	□−▲ = 13.71
Average difference between the values fitted via quadratic regression fitted lines	□−▲= 16.08
Average difference between the values fitted via LOESS (fraction α = 0.60; straight lines fitted)	□−▲ = 15.00
ALIV: A comparison involving actual and linearly interpolated values	□−▲ = 15.00
ADISO: Weighted average difference between successive observations ADISO-O: percentage of ordinal superiority	□−▲ = 19.00 □ 75.00% superior to ▲

4.3.1 Mean difference

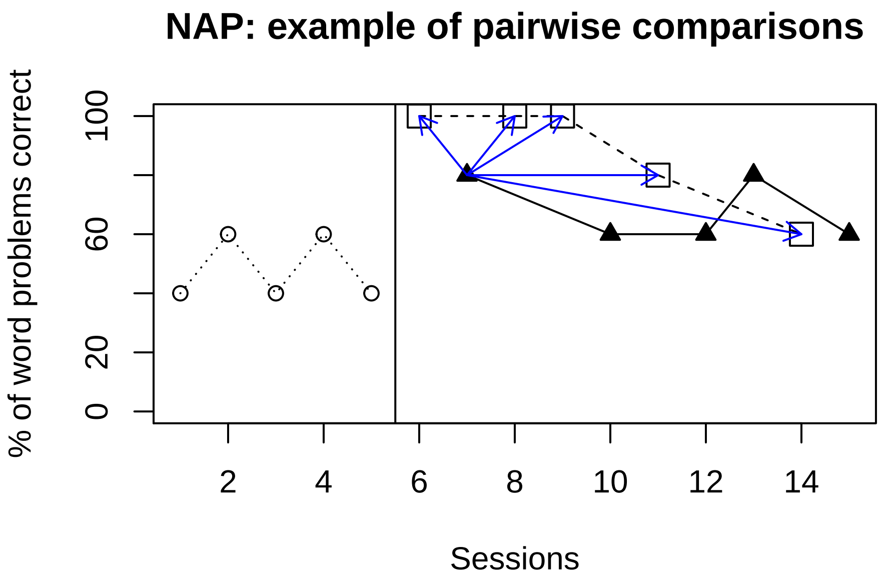


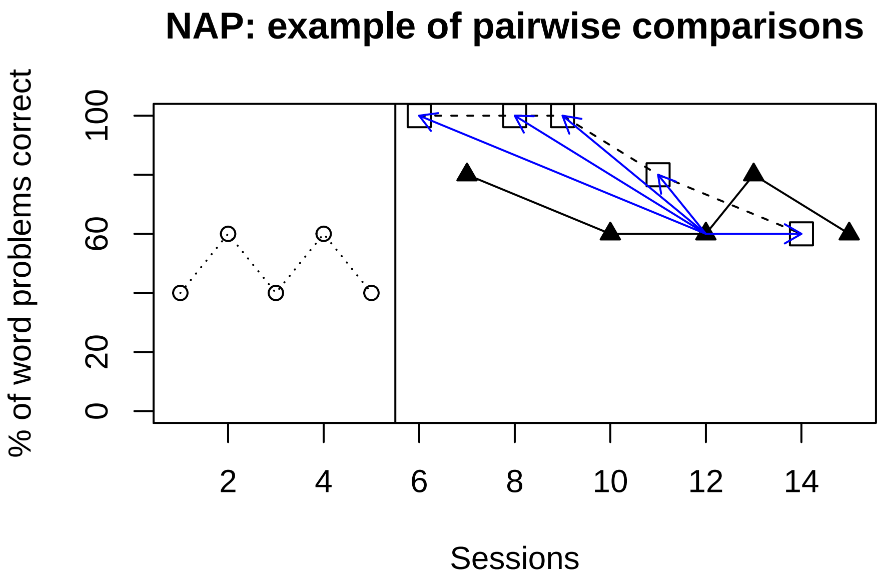
4.3.2 Percentage of nonoverlapping data: Wolery et al. version



4.3.3 Nonoverlap of all pairs

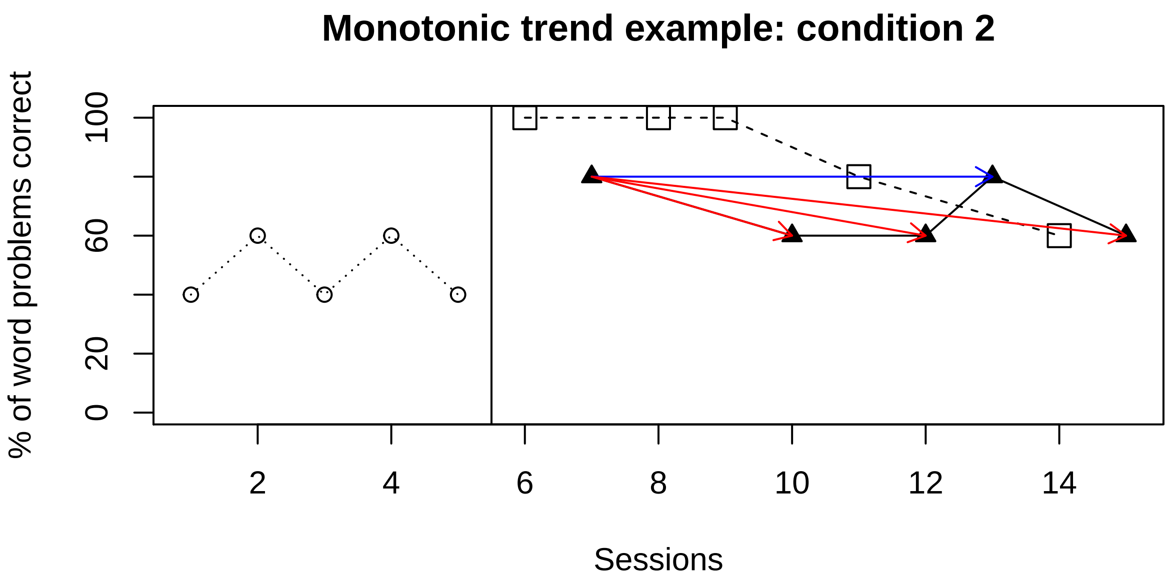
```
## [1] "NAP =  0.82"
```

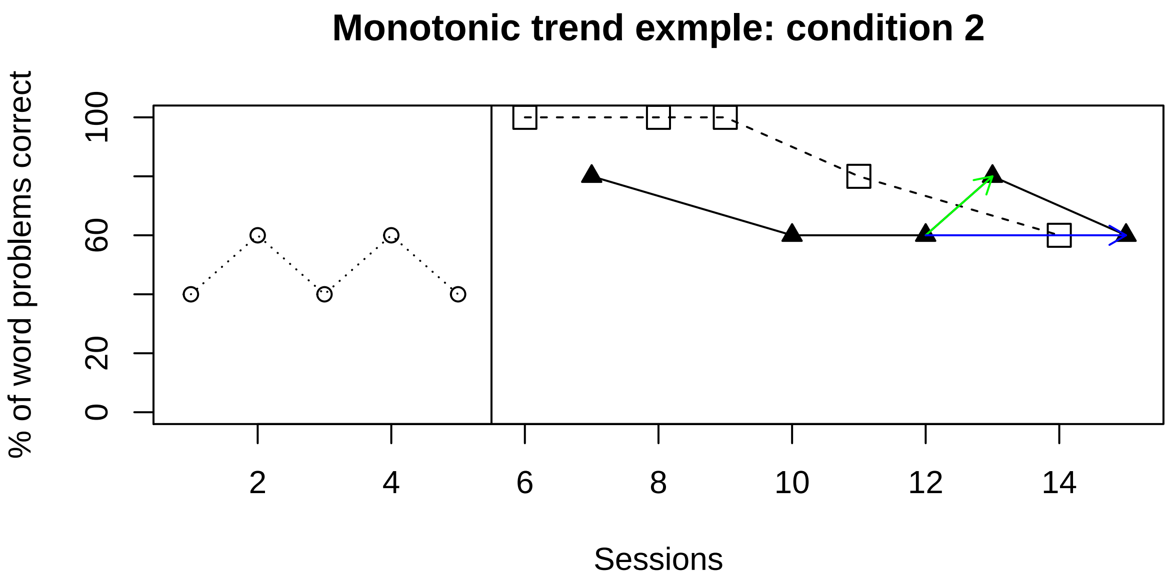




4.3.4 Monotonic trends estimated as in Tau-U

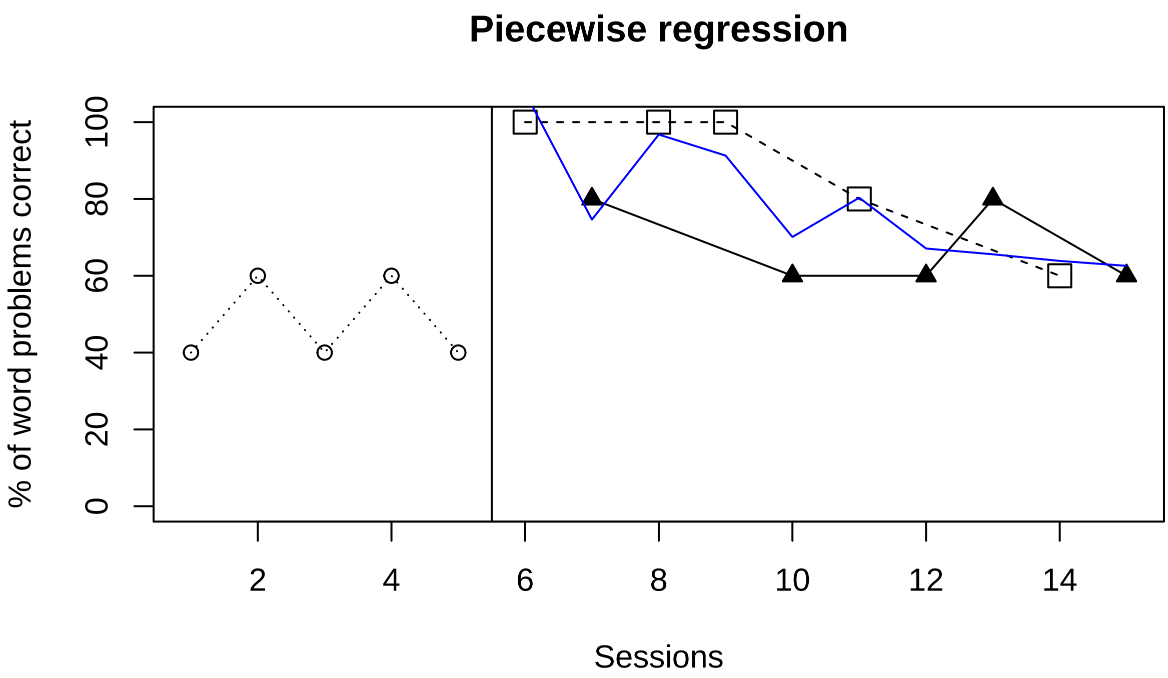
```
## [1] "Condition 2"
## [1] "Negative monotonic trend: 20 % decrease"
## [1] "Condition 3"
## [1] "Negative monotonic trend: 70 % decrease"
```





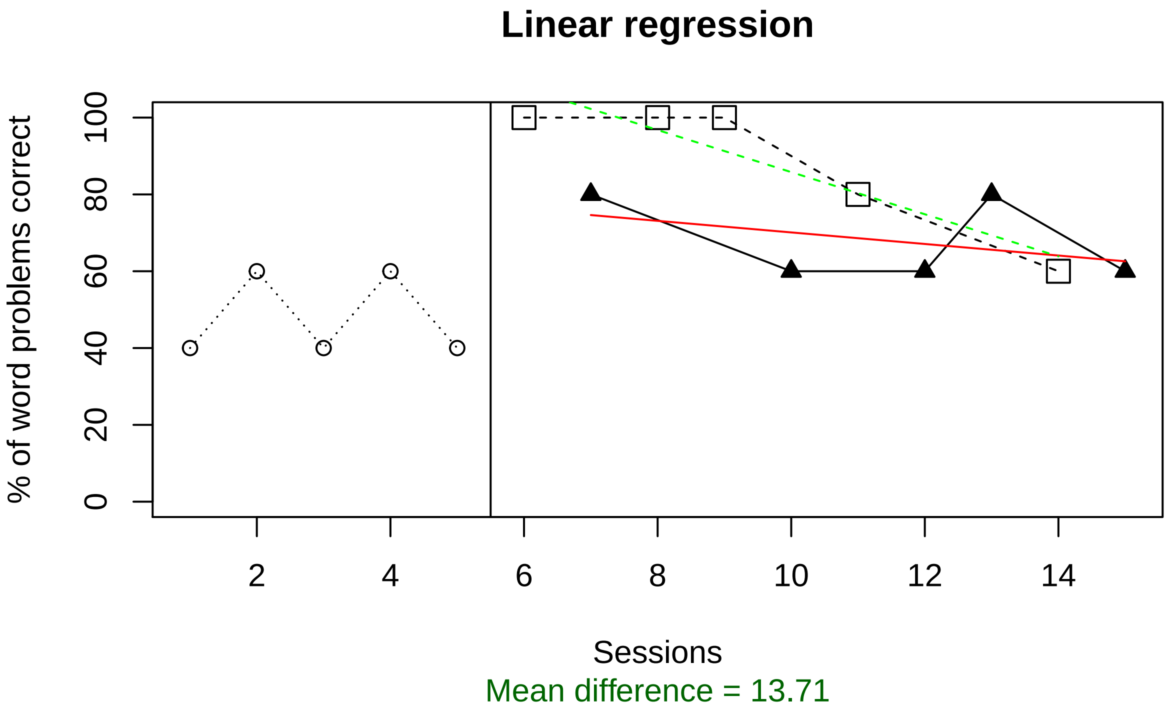
4.3.5 Piecewise regression, comparing at the last measurement occasion

```
## [1] "Comparison 3 vs. 2"
##       Time Score condi_B condi_A Time2 condiA_T2 condiB_T2
##  [1,]    6   100       0       1    -9         0        -9
##  [2,]    7    80       1       0    -8        -8         0
##  [3,]    8   100       0       1    -7         0        -7
##  [4,]    9   100       0       1    -6         0        -6
##  [5,]   10    60       1       0    -5        -5         0
##  [6,]   11    80       0       1    -4         0        -4
##  [7,]   12    60       1       0    -3        -3         0
##  [8,]   13    80       1       0    -2        -2         0
##  [9,]   14    60       0       1    -1         0        -1
## [10,]   15    60       1       0     0         0         0
## [1] "  "
```

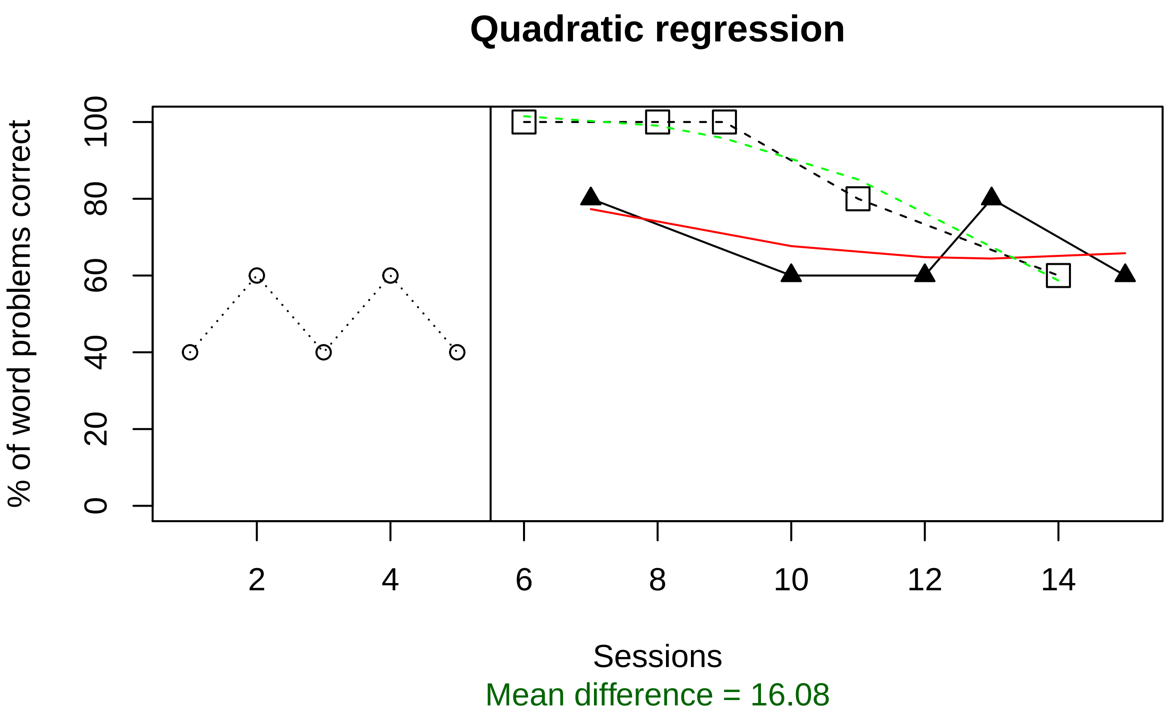



```
## [1] "Condition 2 trend = -1.51"
## [1] "Condition 3 trend = -5.48"
## [1] "Last occasion difference 2-3 = 4.19"
```

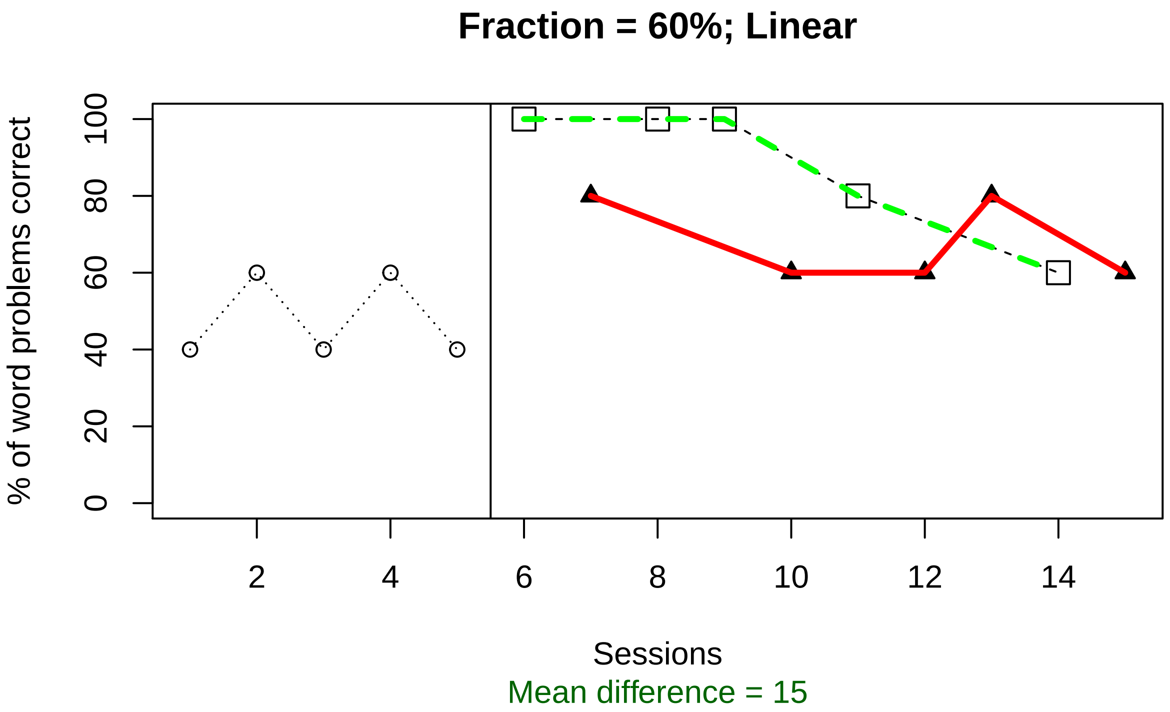
4.3.6 Linear regression: mean difference between fitted values



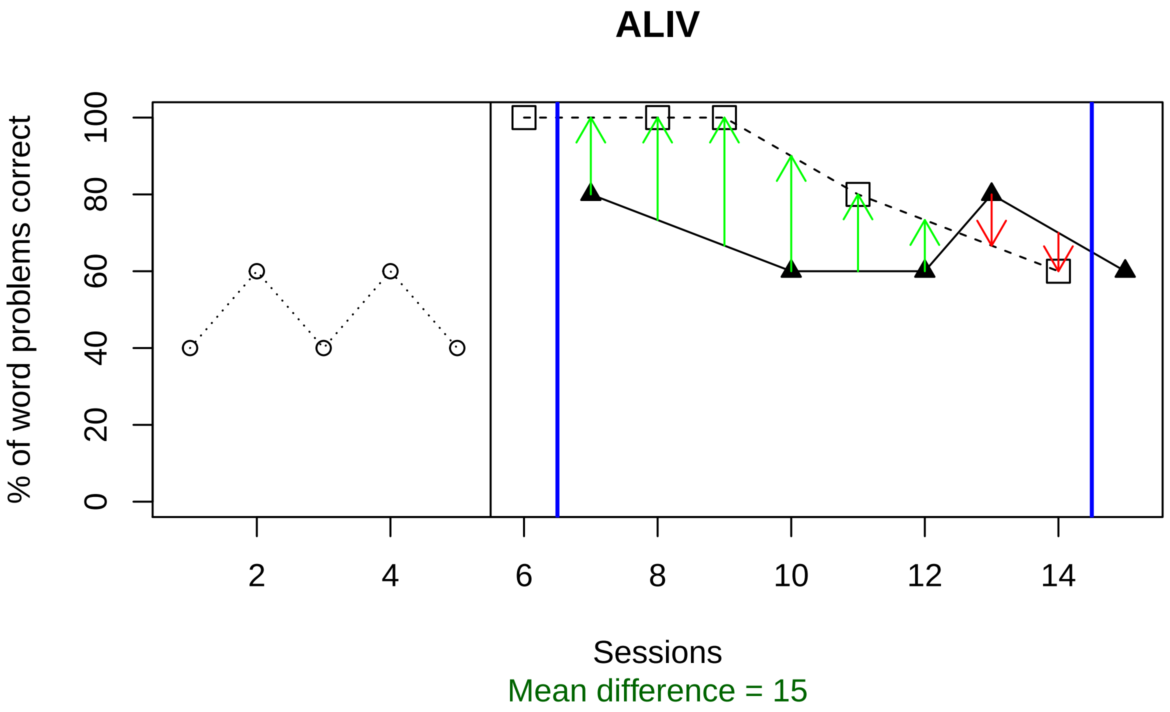
4.3.7 Quadratic regression: mean difference between fitted values



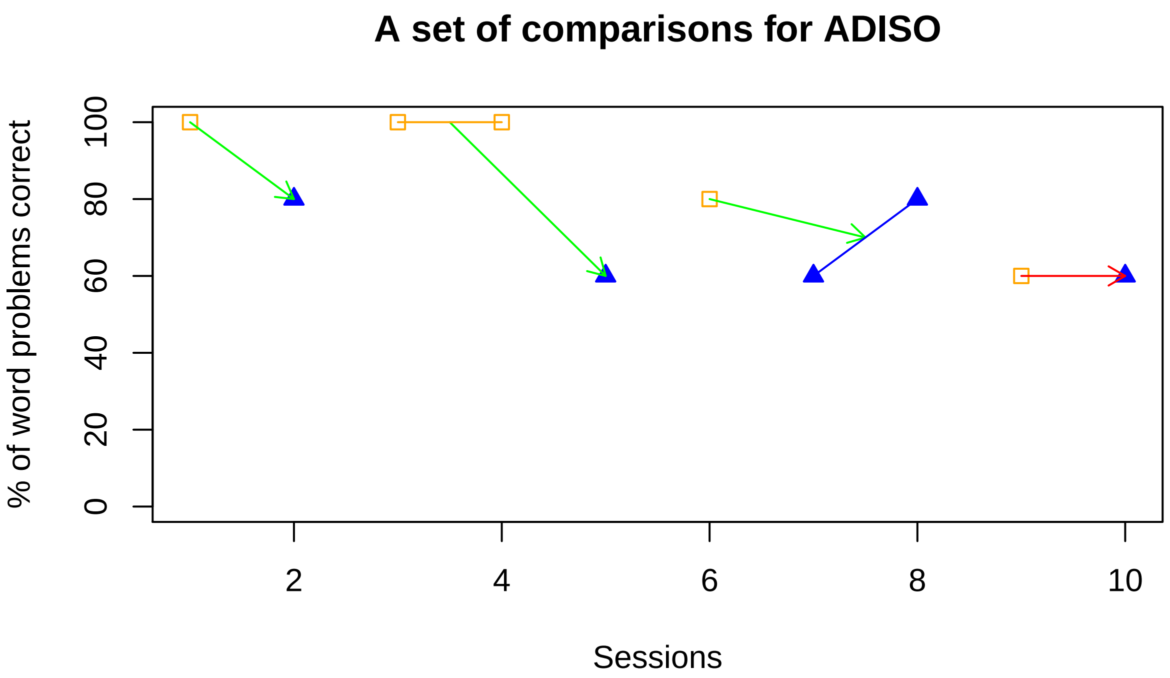
4.3.8 Local regression (LOESS): mean difference between fitted values



4.3.9 Actual and linearly interpolated values: ALIV



4.3.10 Weighted average difference between successive observations: ADISO



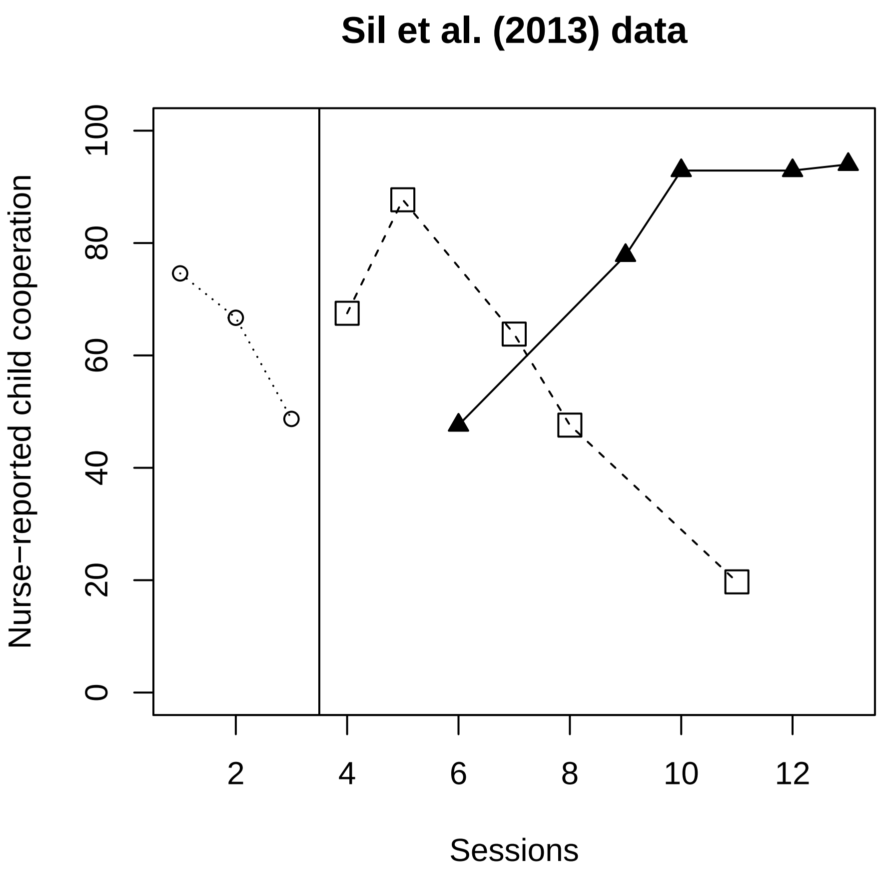

```
##      Comparison ADISO Superiority
## [1,] "2-3"      "-19" "1 superior:  0 %. 2 superior:  75 %"
```

4.4 Sil et al. (2013) data

Illustration 4 contains the data gathered by Sil et al. (2013) for child cooperation as reported by the nurses [their figure 3; lower panel].

Sil, S., Dahlquist, L. M., & Burns, A. J. (2013). Case study: videogame distraction reduces behavioral distress in a preschool-aged child undergoing repeated burn dressing changes: a single-subject design. *Journal of Pediatric Psychology*, *38*, 330−341.

```
score <- c(74.6,66.7,48.7,67.5,87.7,47.6,63.8,47.6,77.8,92.9,19.7,92.9,94)
condi <- c(1,1,1,3,3,2,3,3,2,2,3,2,2)
```

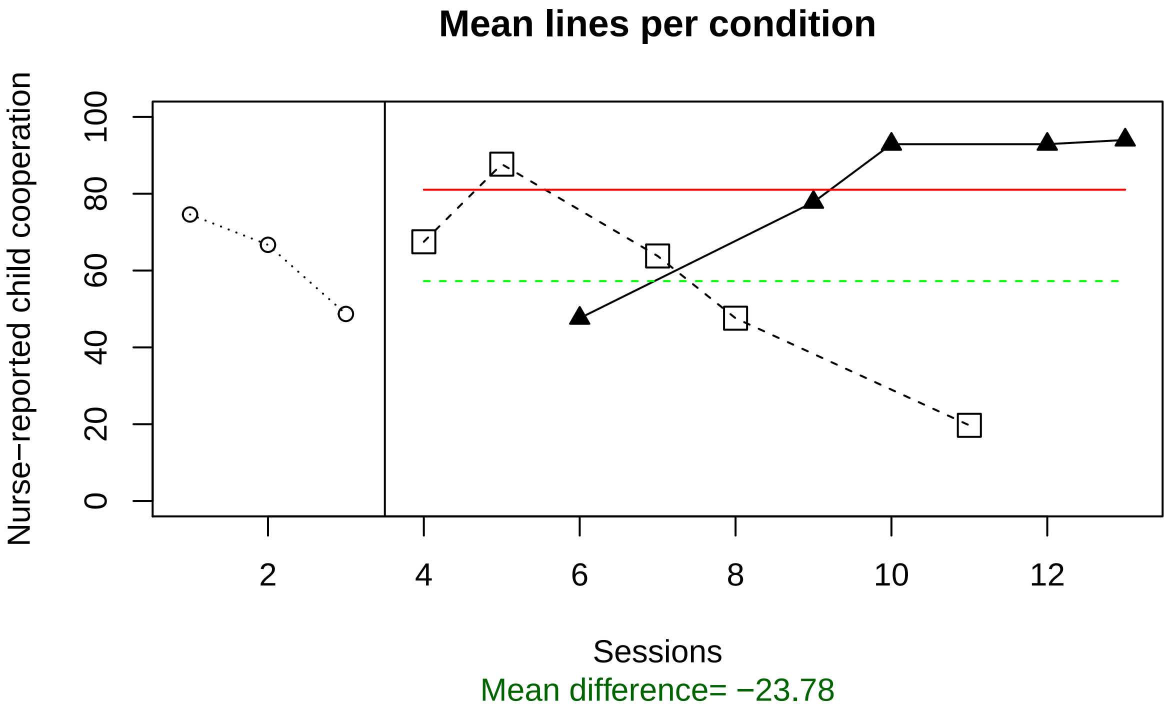


A summary of the results follows on the next page.

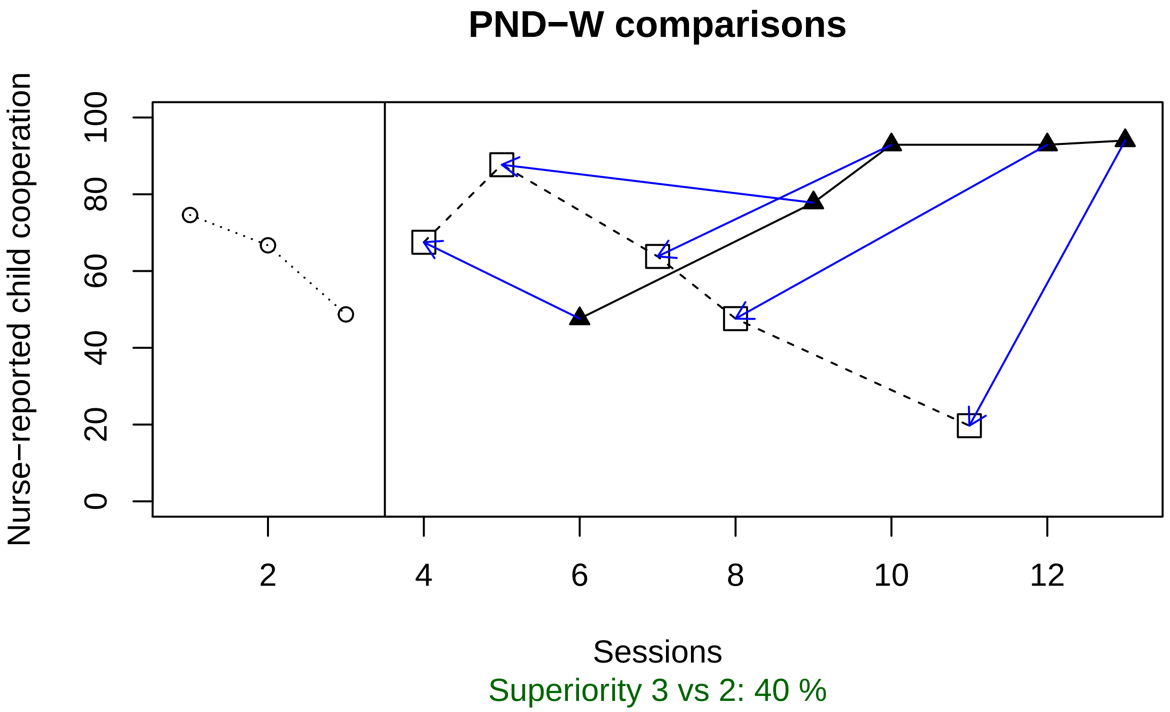
Conditions: 1 - ○ [baseline not considered]; 2 - ▲; 3 - □

Analytical technique	Quantifications
Mean difference	□−▲ = −23.78
Percentage of nonoverlapping data according to the proposal by Wolery, Gast, et al. (2010).	□ 40% superior to ▲
Nonovelap of all pairs	□−▲ = 18.00%
Tau-U: only the direction and amount of monotonic trend is represented here, not including overlap	▲ monotonic trend =90.00 % increase □ monotonic trend = 80.00% decrease
Piecewise regression, comparing at the last intervention measurement occasion	□−▲last point = −94.04 ▲ slope = 6.65 □ slope = −8.38
Average difference between the values fitted via linear regression fitted lines	□−▲ = −26.38
Average difference between the values fitted via quadratic regression fitted lines	□−▲= −26.85
Average difference between the values fitted via LOESS (fraction α = 0.60; straight lines fitted)	□−▲ = −27.08
ALIV: A comparison involving actual and linearly interpolated values	□−▲ = −27.08
ADISO: Weighted average difference between successive observations ADISO-O: percentage of ordinal superiority	□−▲ = −24.98 □ 33% superior to ▲

4.4.1 Mean difference

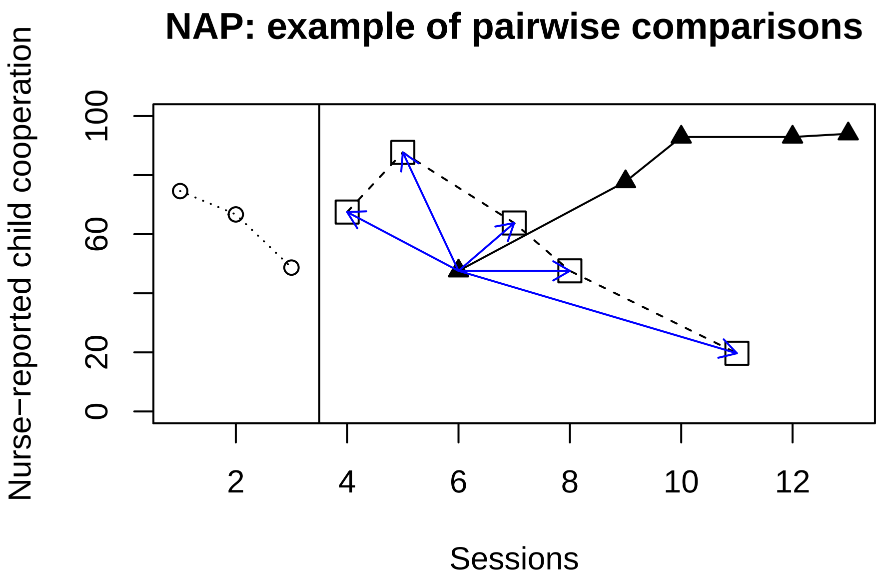


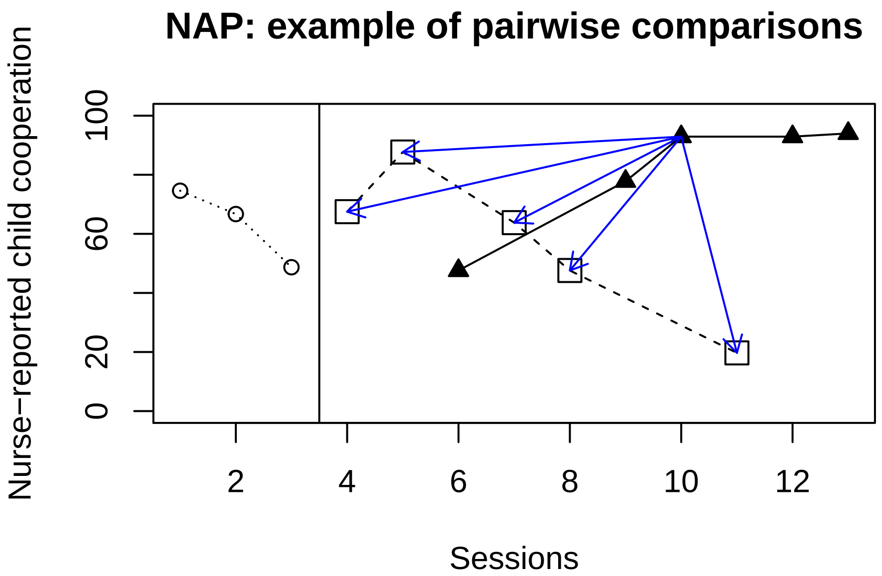
4.4.2 Percentage of nonoverlapping data: Wolery et al. version



4.4.3 Nonoverlap of all pairs

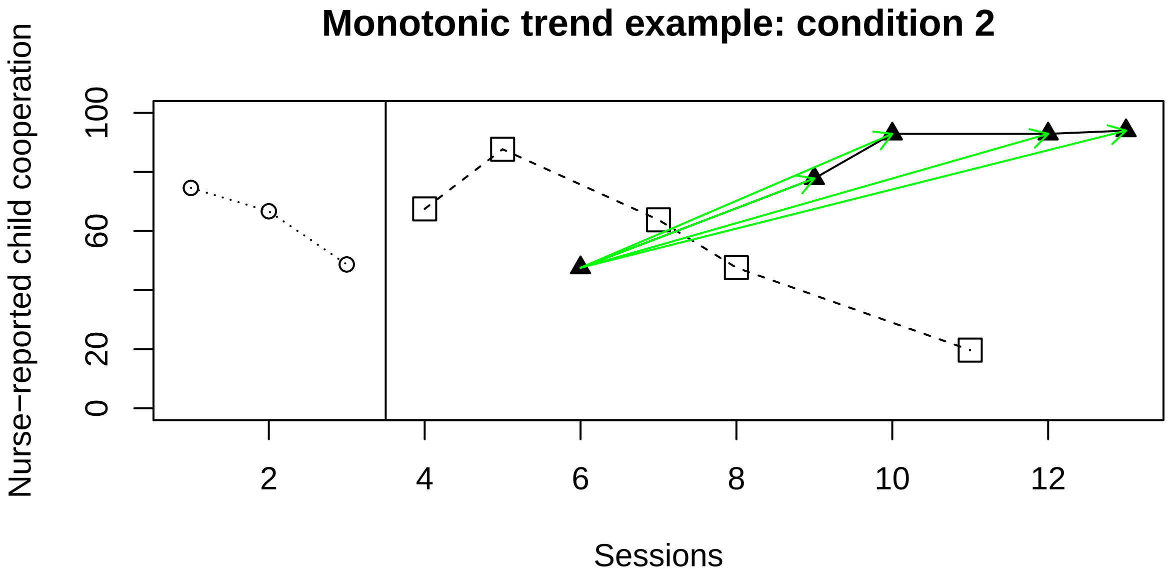
```
## [1] "NAP =  0.18"
```

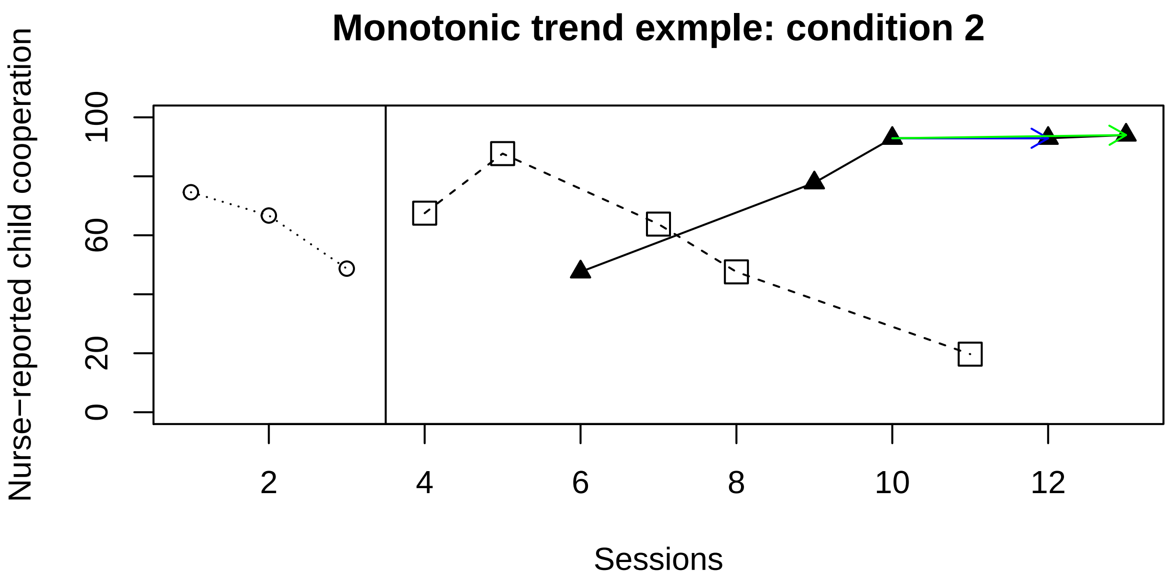




4.4.4 Monotonic trends estimated as in Tau-U

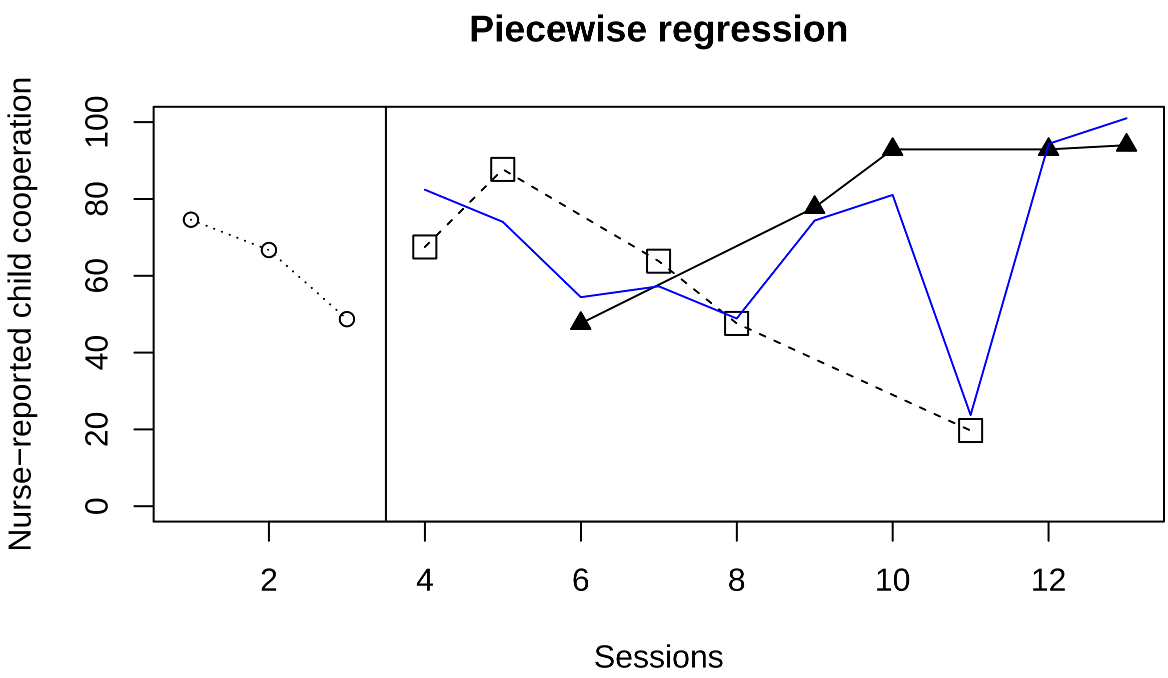
```
## [1] "Condition 2"
## [1] "Positive monotonic trend: 90 % increase"
## [1] "Condition 3"
## [1] "Negative monotonic trend: 80 % decrease"
```





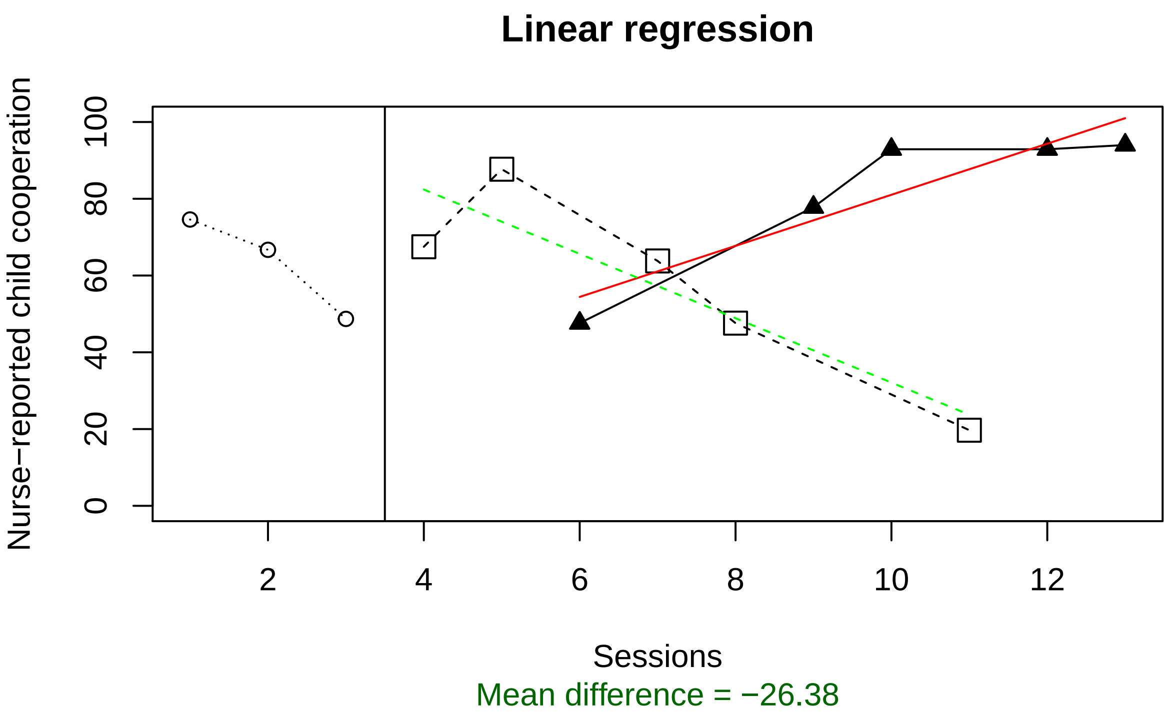
4.4.5 Piecewise regression, comparing at the last measurement occasion

```
## [1] "Comparison 3 vs. 2"
##       Time Score condi_B condi_A Time2 condiA_T2 condiB_T2
##  [1,]    4  67.5       0       1    -9         0        -9
##  [2,]    5  87.7       0       1    -8         0        -8
##  [3,]    6  47.6       1       0    -7        -7         0
##  [4,]    7  63.8       0       1    -6         0        -6
##  [5,]    8  47.6       0       1    -5         0        -5
##  [6,]    9  77.8       1       0    -4        -4         0
##  [7,]   10  92.9       1       0    -3        -3         0
##  [8,]   11  19.7       0       1    -2         0        -2
##  [9,]   12  92.9       1       0    -1        -1         0
## [10,]   13  94.0       1       0     0         0         0
## [1] "  "
```

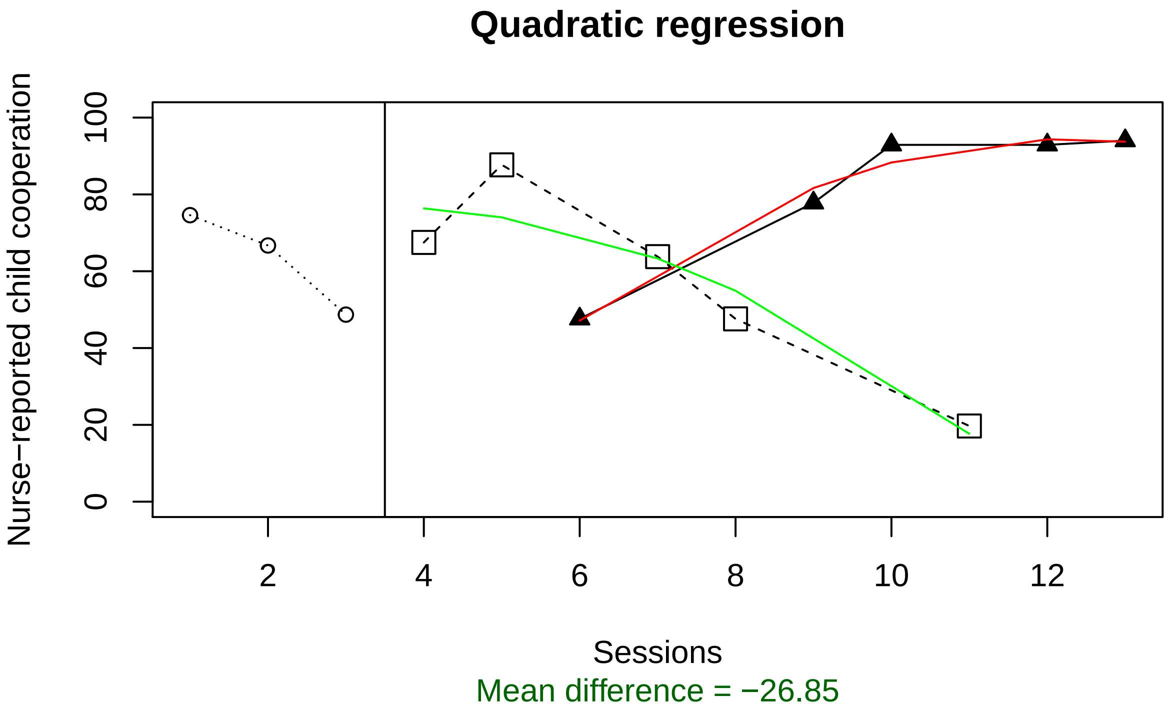



```
## [1] "Condition 2 trend = 6.65"
## [1] "Condition 3 trend = -8.38"
## [1] "Last occasion difference 2-3 = 94.04"
```

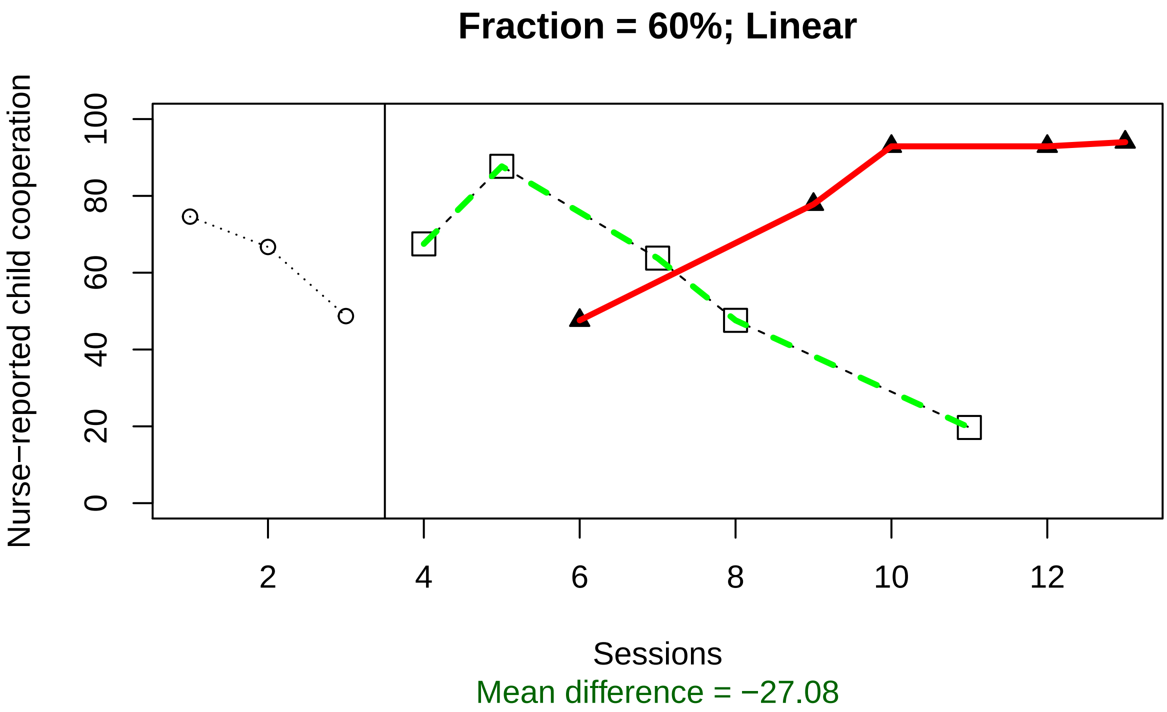
4.4.6 Linear regression: mean difference between fitted values



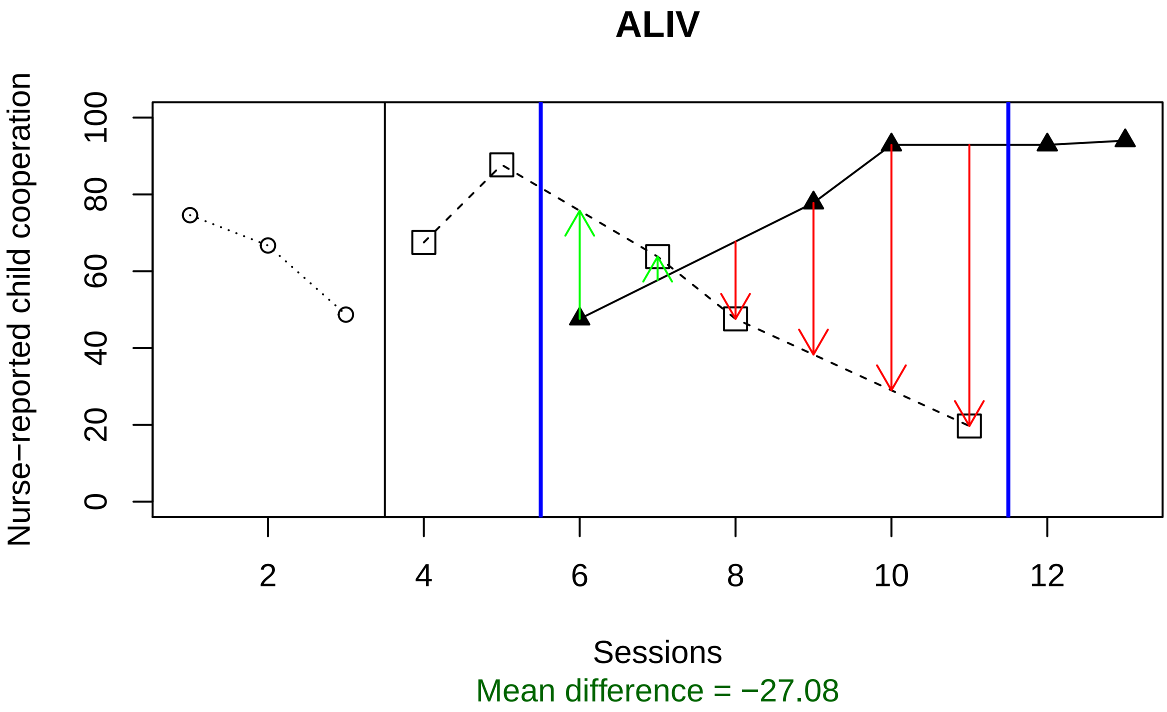
4.4.7 Quadratic regression: mean difference between fitted values



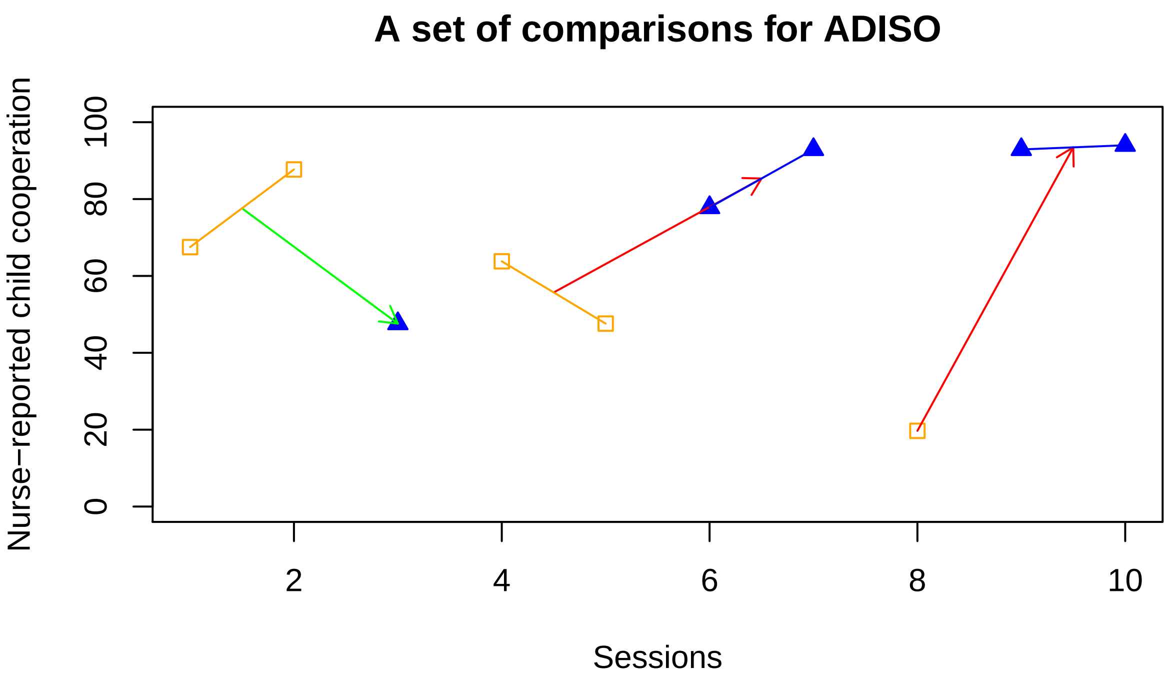
4.4.8 Local regression (LOESS): mean difference between fitted values



4.4.9 Actual and linearly interpolated values: ALIV



4.4.10 Weighted average difference between successive observations: ADISO



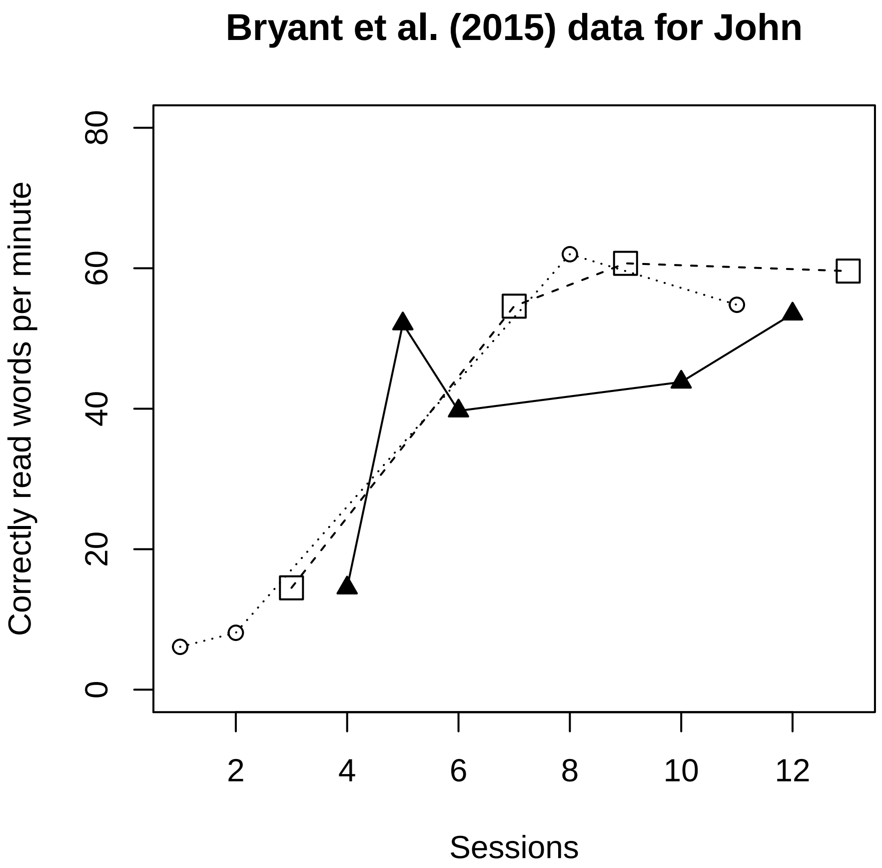

```
##       Comparison ADISO   Superiority
## [1,] "2-3"      "24.98" "1 superior:  66.67 %. 2 superior:  33.33 %"
```

4.5 Braynt et al. (2015) data

Illustration 5 contains the data gathered by Bryant et al. (2015) for John [their figure 1; middle panel].

Bryant, B. R., Ok, M., Kang, E. Y., Kim, M. K., Lang, R., Bryant, D. P., & Pfannestiel, K. (2015). Performance of fourth-grade students with learning disabilities on multiplication facts comparing teacher-mediated and technology-mediated interventions: A preliminary investigation. *Journal of Behavioral Education*, *24*, 255–272.

```
score <- c(6.1,8.1,14.5,14.5,52.1,39.7,54.6,62,60.7,43.8,54.8,53.5,59.6)
condi <- c(1,1,3,2,2,2,3,1,3,2,1,2,3)
```

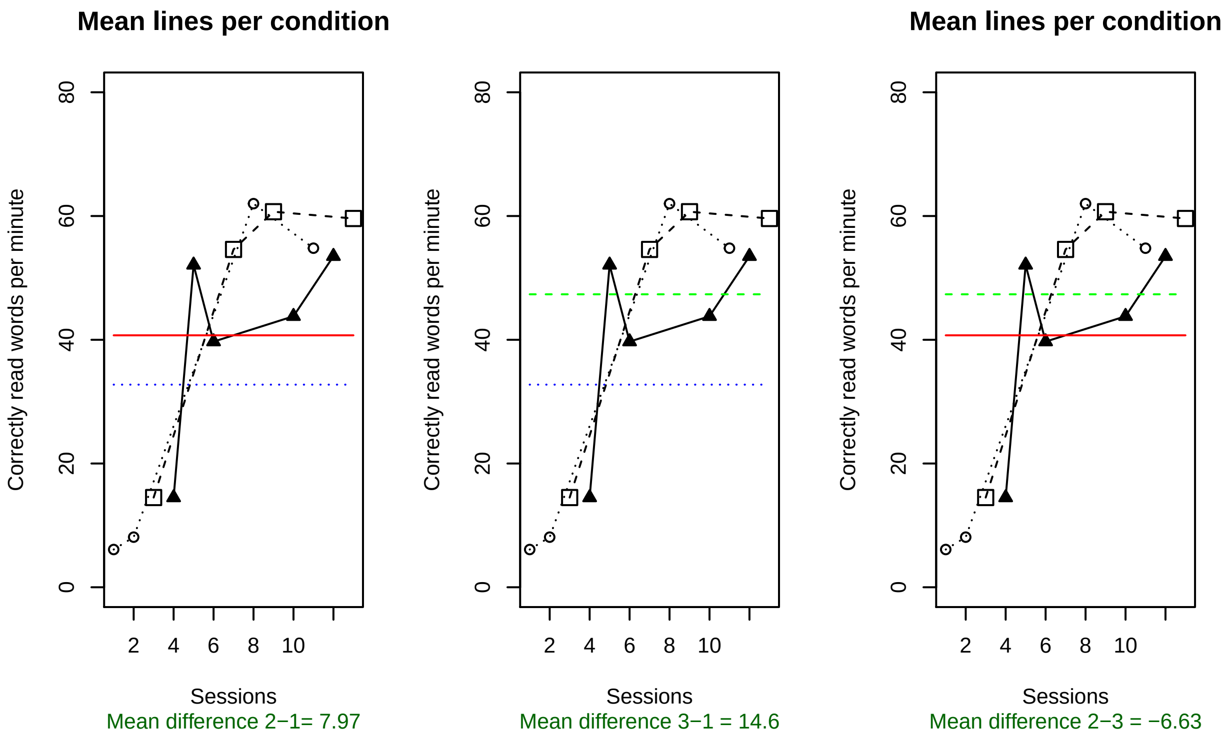


A summary of the results follows on the next page.

Conditions: 1 - ○; 2 - ▲; 3 - □

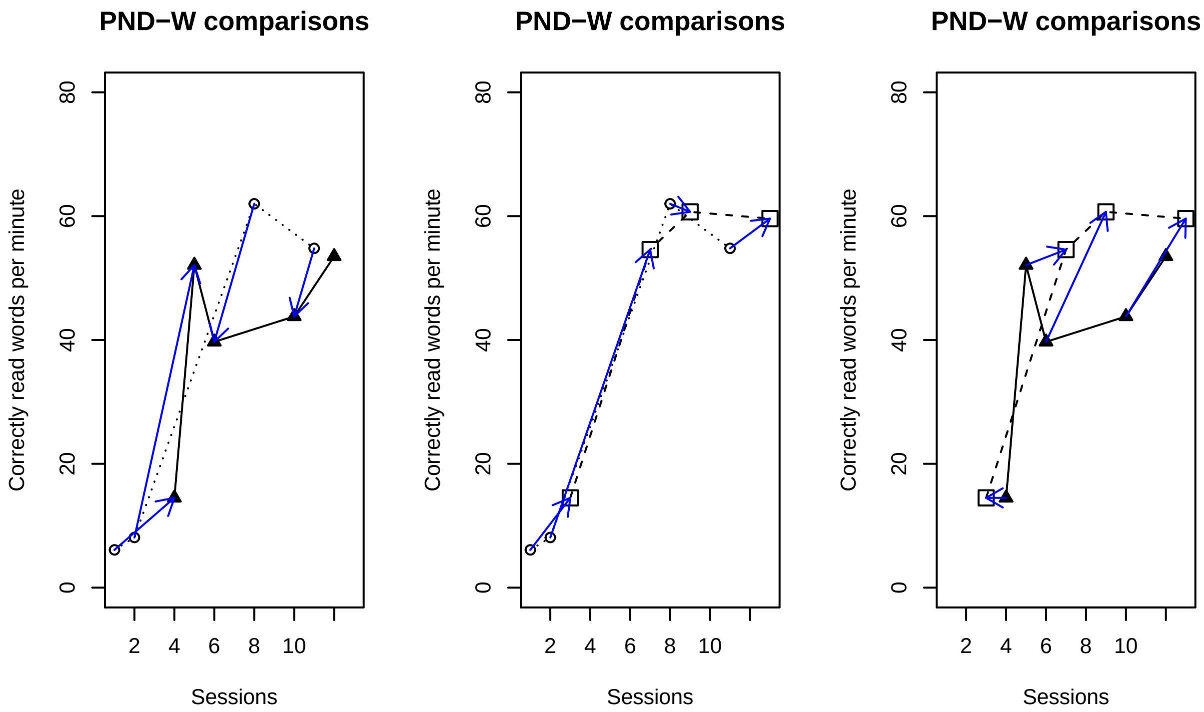
Analytical technique	**Quantifications**
Mean difference	▲−○= 7.97 □−○ =14.6 □−▲ = 6.63
Percentage of nonoverlapping data according to the proposal by Wolery, Gast, et al. (2010).	▲ 50.00% superior to ○ □ 75.00% superior to ○ □ 25.00% superior to ▲
Nonovelap of all pairs	▲−○ = 50.00% □−○ = 62.50% □−▲ = 77.50%
Tau-U: only the direction and amount of monotonic trend is represented here, not including overlap	○ monotonic trend = 66.67% increase ▲ monotonic trend = 60.00% increase □ monotonic trend = 66.67% decrease
Piecewise regression, comparing at the last intervention measurement occasion	▲−○ last point: 20.14 □−○ last point: 6.67 □−▲last point: −13.47 ○ slope = 5.81 ▲ slope = 2.76 □ slope = 4.45
Average difference between the values fitted via linear regression fitted lines	▲−○ = −3.37 □−○ = 1.44 □−▲ = 4.98
Average difference between the values fitted via quadratic regression fitted lines	▲−○ =−8.80 □−○ =−2.71 □−▲= 7.99
Average difference between the values fitted via LOESS (fraction α = 0.60; straight lines fitted)	▲−○ = −9.09 □−○ = 0.33 □−▲ = 8.84
ALIV: A comparison involving actual and linearly interpolated values	▲−○ = −8.47 □−○ = 0.31 □−▲ = 8.84
ADISO: Weighted average difference between successive observations ADISO-O: percentage of ordinal superiority	▲−○ = 11.41 □−○ = 14.60 □−▲ = 2.16 ▲ 33.33% superior to ○ □ 66.67% superior to ○ □ 75.00% superior to ▲

4.5.1 Mean difference



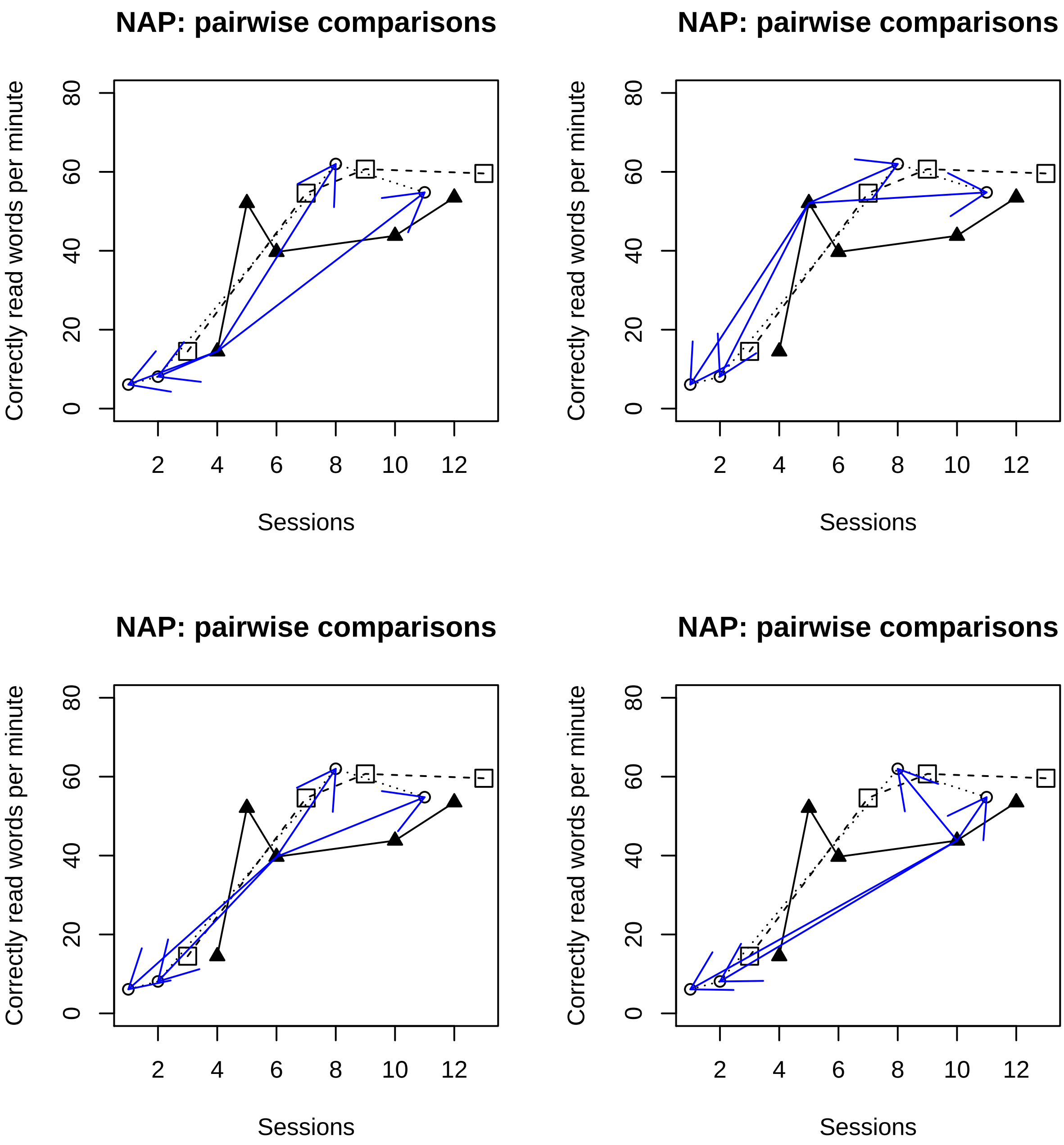
4.5.2 Percentage of nonoverlapping data: Wolery et al. version

```
##      Comparison PND-W
## [1,] "2 > 1"    "50 %"
## [2,] "3 > 1"    "75 %"
## [3,] "3 > 2"    "25 %"
```



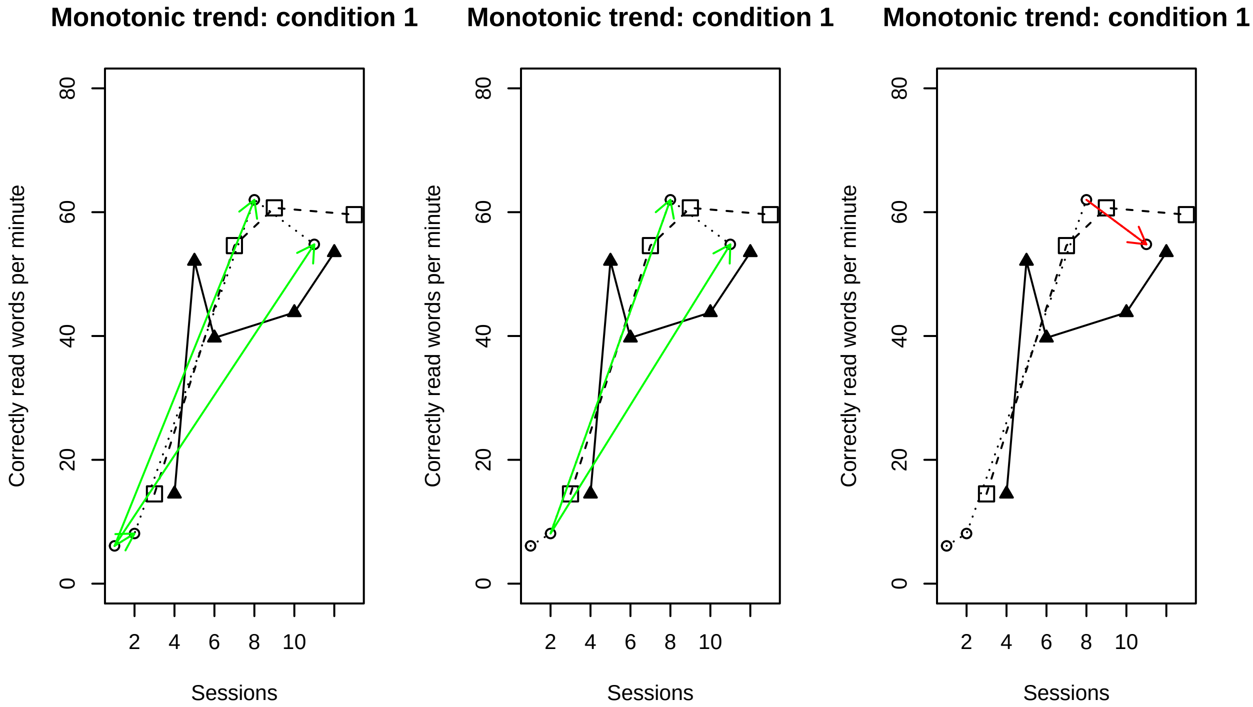
4.5.3 Nonoverlap of all pairs

```
##      Comparison NAP
## [1,] "2 > 1"    "0.5"
## [2,] "3 > 1"    "0.625"
## [3,] "3 > 2"    "0.775"
```



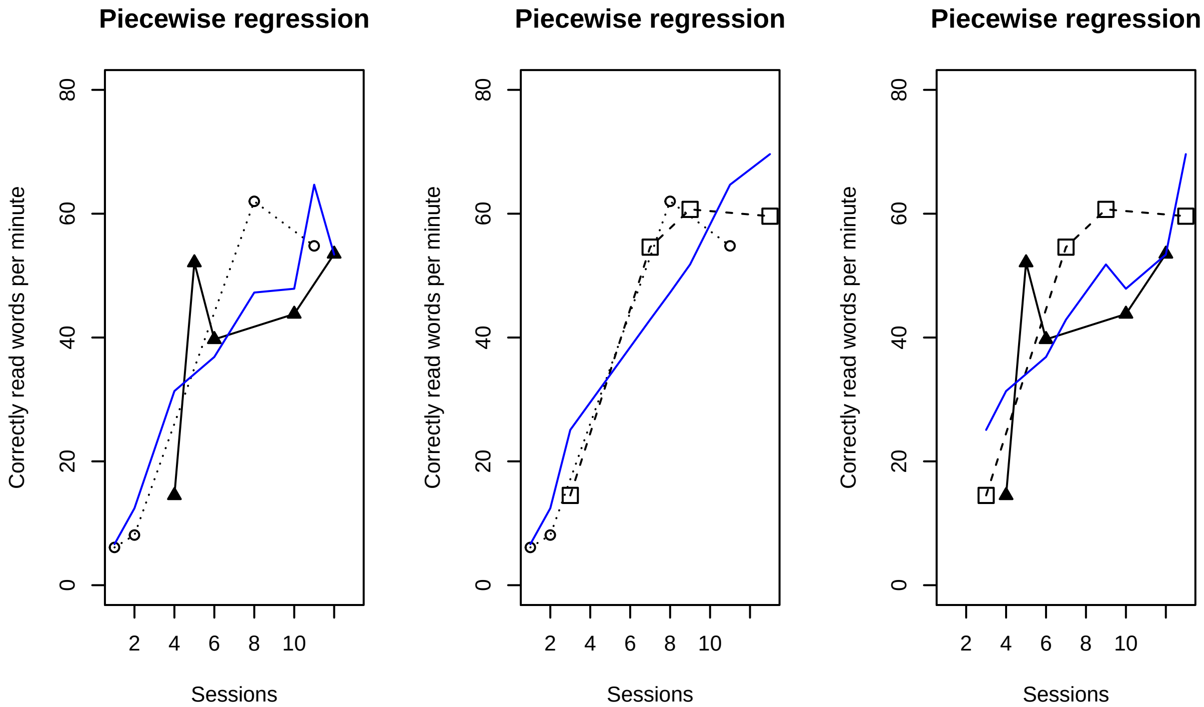
4.5.4 Monotonic trends estimated as in Tau-U

```
## [1] "Condition 1"
## [1] "Positive monotonic trend: 66.67 % increase"
## [1] "Condition 2"
## [1] "Positive monotonic trend: 60 % increase"
## [1] "Condition 3"
## [1] "Positive monotonic trend: 66.67 % increase"
```



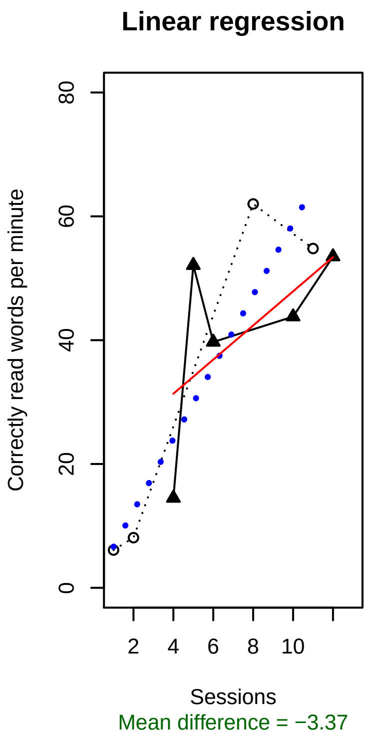
4.5.5 Piecewise regression, comparing at the last measurement occasion

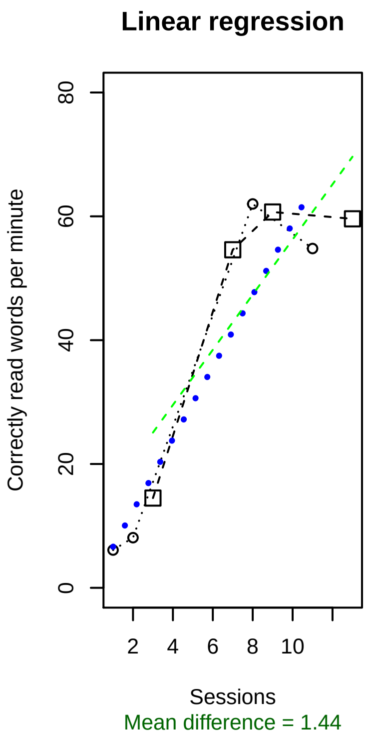
```
## [1] "Comparison 2 vs. 1"
##        Time Score condi_B condi_A Time2 condiA_T2 condiB_T2
##   [1,]    1   6.1       1       0   -12       -12         0
##   [2,]    2   8.1       1       0   -11       -11         0
##   [3,]    4  14.5       0       1    -9         0        -9
##   [4,]    5  52.1       0       1    -8         0        -8
##   [5,]    6  39.7       0       1    -7         0        -7
##   [6,]    8  62.0       1       0    -5        -5         0
##   [7,]   10  43.8       0       1    -3         0        -3
##   [8,]   11  54.8       1       0    -2        -2         0
##   [9,]   12  53.5       0       1    -1         0        -1
## [1] "  "
## [1] "Comparison 3 vs. 1"
##       Time Score condi_B condi_A Time2 condiA_T2 condiB_T2
## [1,]    1   6.1       1       0   -12       -12         0
## [2,]    2   8.1       1       0   -11       -11         0
## [3,]    3  14.5       0       1   -10         0       -10
## [4,]    7  54.6       0       1    -6         0        -6
## [5,]    8  62.0       1       0    -5        -5         0
## [6,]    9  60.7       0       1    -4         0        -4
## [7,]   11  54.8       1       0    -2        -2         0
## [8,]   13  59.6       0       1     0         0         0
## [1] "  "
## [1] "Comparison 2 vs. 3"
##        Time Score condi_B condi_A Time2 condiA_T2 condiB_T2
##   [1,]    3  14.5       0       1   -10         0       -10
##   [2,]    4  14.5       1       0    -9        -9         0
##   [3,]    5  52.1       1       0    -8        -8         0
##   [4,]    6  39.7       1       0    -7        -7         0
##   [5,]    7  54.6       0       1    -6         0        -6
##   [6,]    9  60.7       0       1    -4         0        -4
##   [7,]   10  43.8       1       0    -3        -3         0
##   [8,]   12  53.5       1       0    -1        -1         0
##   [9,]   13  59.6       0       1     0         0         0
```

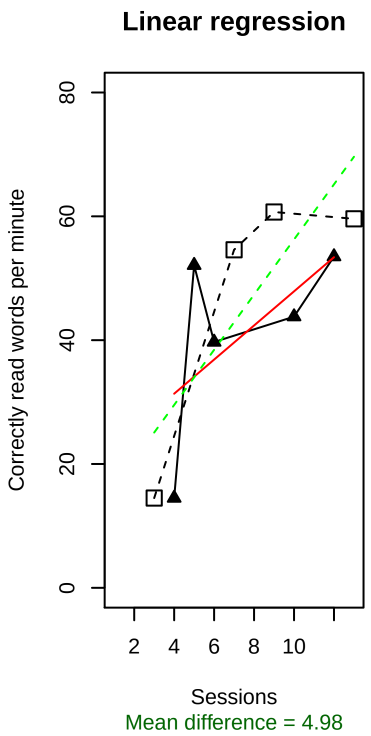


```
##      Conditions Difference at last point
## [1,] "2-1"      "20.14"
## [2,] "3-1"      "6.67"
## [3,] "3-2"      "-13.47"
##      Condition Trend
## [1,]         1  5.81
## [2,]         2  2.76
## [3,]         3  4.45
```

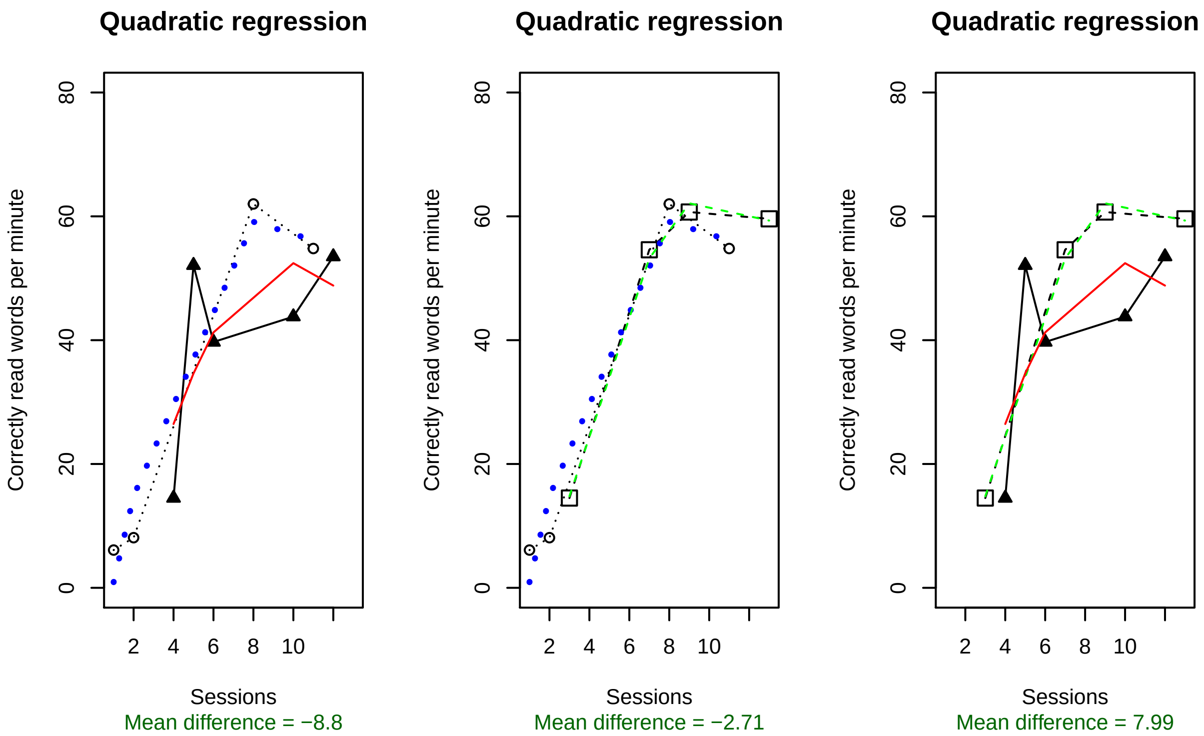
4.5.6 Linear regression: mean difference between fitted values



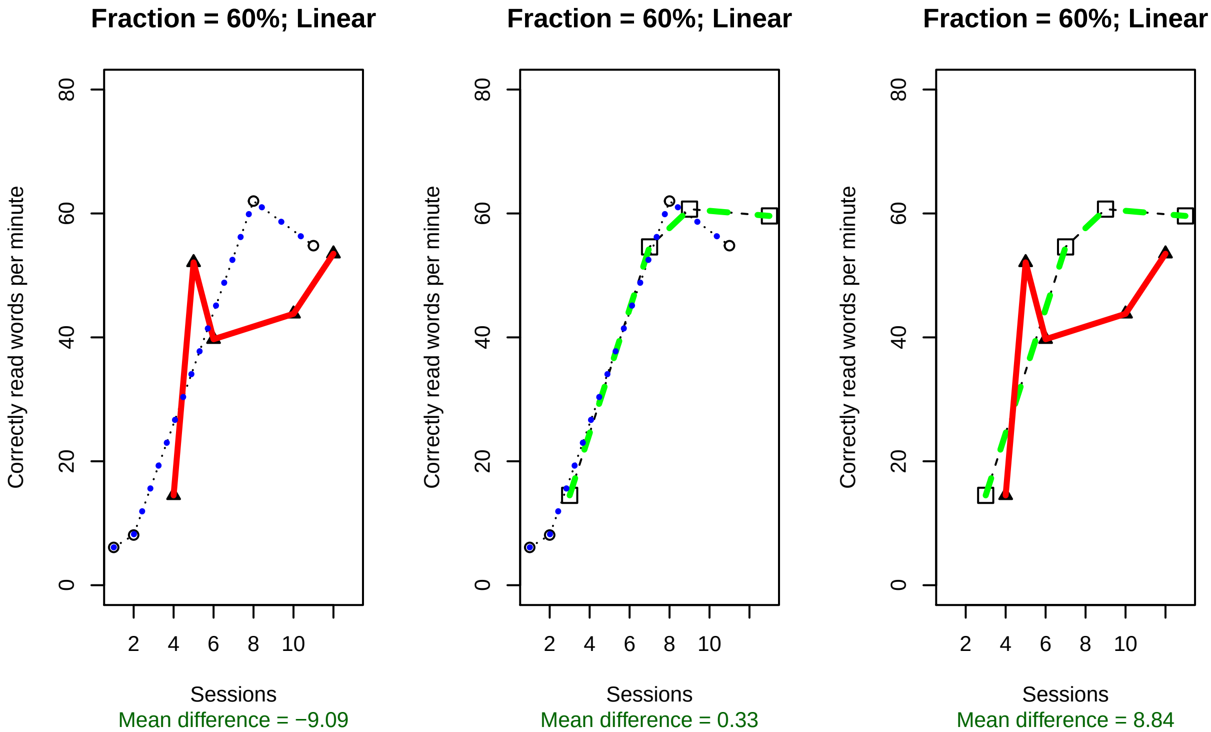




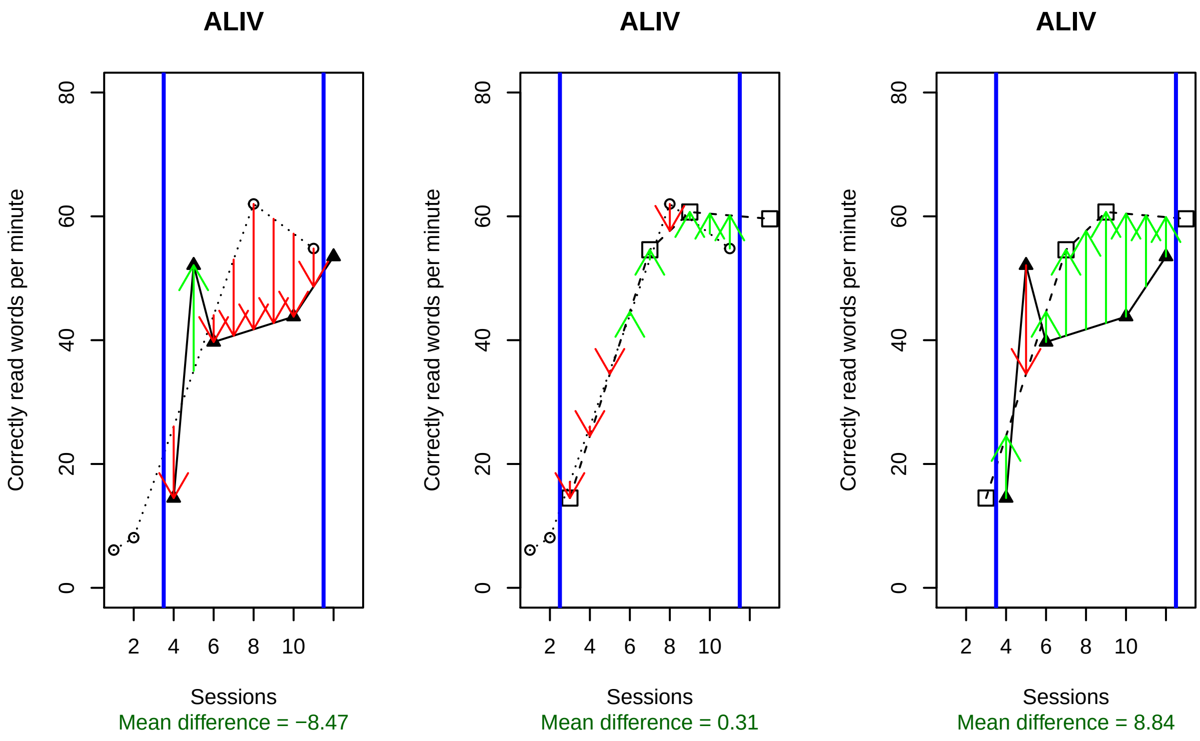
4.5.7 Quadratic regression: mean difference between fitted values



4.5.8 Local regression (LOESS): mean difference between fitted values

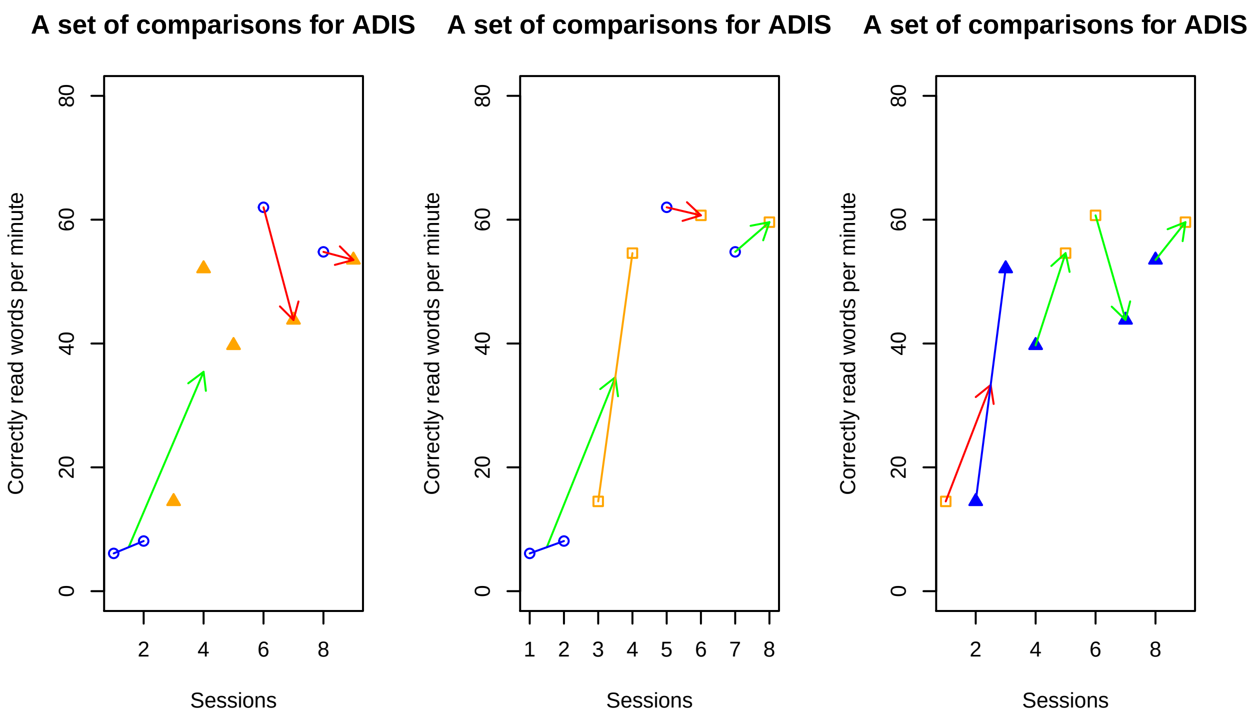


4.5.9 Actual and linearly interpolated values: ALIV



4.5.10 Weighted average difference between successive observations: ADISO

```
## Error in `[<-`(`*tmp*`, 2, 4, value = "1 superior:  33.33 %.  3 superior:  66.67 %"):  subscript
out of bounds
## Error in `[<-`(`*tmp*`, 3, 4, value = "2 superior:  25 %.  3 superior:  75 %"):  subscript
out of bounds
```



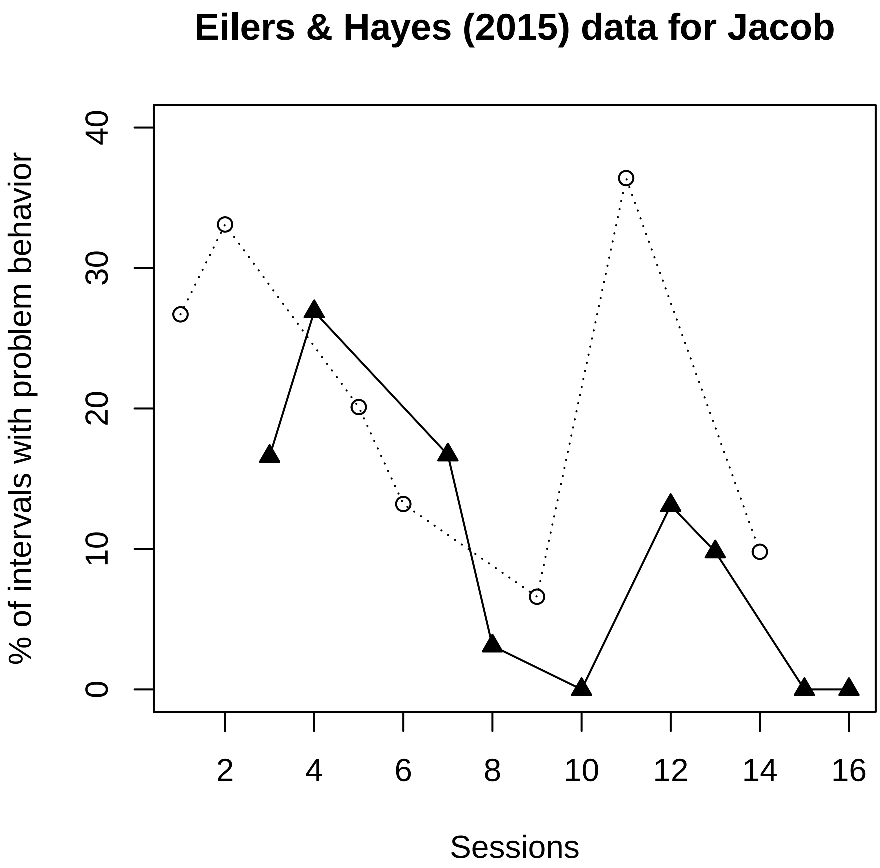

```
##       Comparison ADISO    Superiority
## [1,] "1-2"      "-11.41" "1 superior:  66.67 %. 2 superior:  33.33 %"
## [2,] "1-3"      "-10.32" "-14.6"
## [3,] "2-3"      "-4.78"  "-2.16"
```

4.6 Eilers and Hayes (2015) data

Illustration 6 contains the data gathered by Eilers and Hayes (2015) for Jacob [their figure 2; upper panel].

Eilers, H. J., & Hayes, S. C. (2015). Exposure and response prevention therapy with cognitive defusion exercises to reduce repetitive and restrictive behaviors displayed by children with autism spectrum disorder. *Research in Autism Spectrum Disorders*, *19*, 18–31.

```
score <- c(26.7,33.1,16.6,26.9,20.1,13.2,16.7,3.1,6.6,0,36.4,13.1,9.8,9.8,0,0)
condi <- c(3,3,2,2,3,3,2,2,3,2,3,2,2,3,2,2)
```

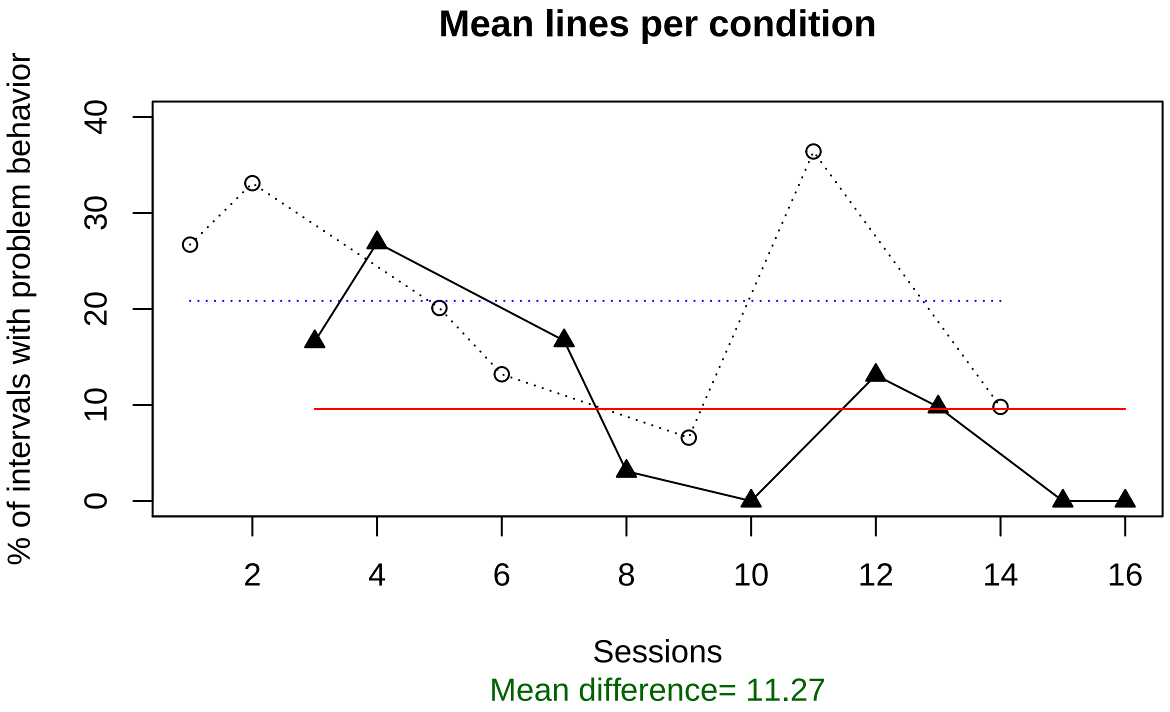


A summary of the results follows on the next page.

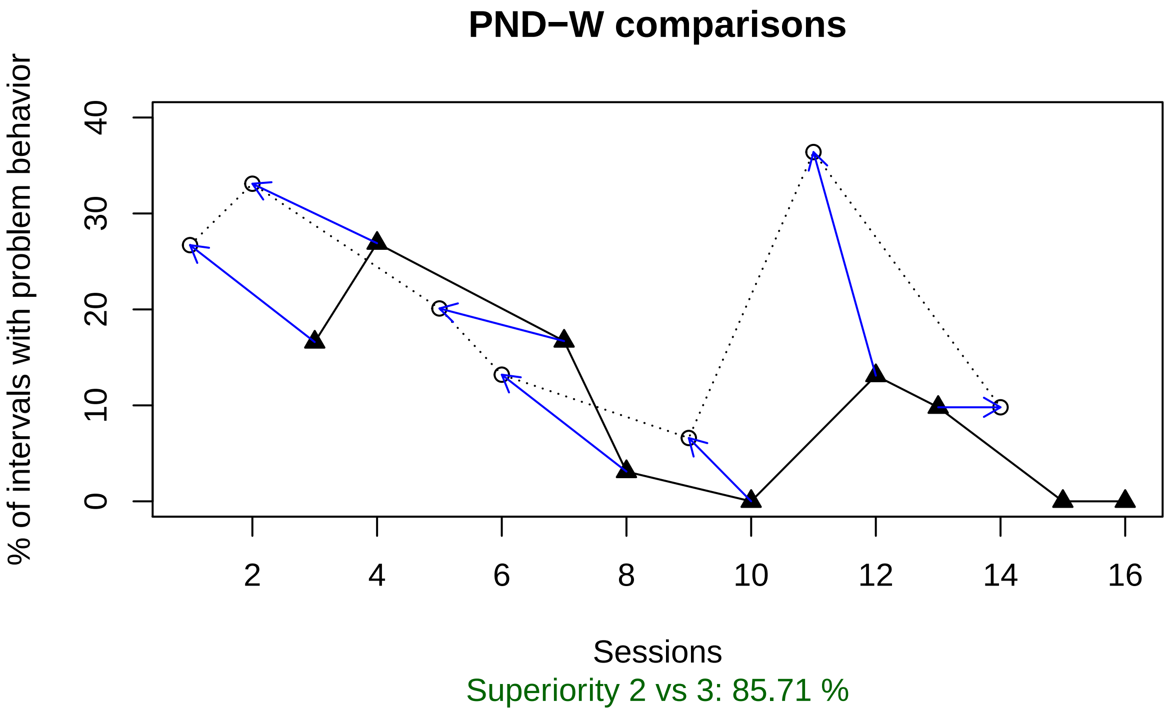
Conditions: 2 - ▲; 3 - ○

Analytical technique	Quantifications
Mean difference	▲−○ = −11.27
Percentage of nonoverlapping data according to the proposal by Wolery, Gast, et al. (2010).	▲85.71 % superior (i.e., lower value) to ○
Nonovelap of all pairs	▲−○ = 77.00%
Tau-U: only the direction and amount of monotonic trend is represented here, not including overlap	○ monotonic trend =33.33% crease ▲ monotonic trend =58.33% crease
Piecewise regression, comparing at the last intervention measurement occasion	▲−○last point = −11.79 ○ slope = −0.97 ▲ slope = −1.51
Average difference between the values fitted via linear regression fitted lines	▲−○ = −7.74
Average difference between the values fitted via quadratic regression fitted lines	▲−○= −7.44
Average difference between the values fitted via LOESS (fraction α = 0.60; straight lines fitted)	▲−○ = −6.76
ALIV: A comparison involving actual and linearly interpolated values	▲−○ = −7.00
ADISO: Weighted average difference between successive observations ADISO-O: percentage of ordinal superiority	▲−○ = −9.60 ▲ 83.33% superior to○

4.6.1 Mean difference

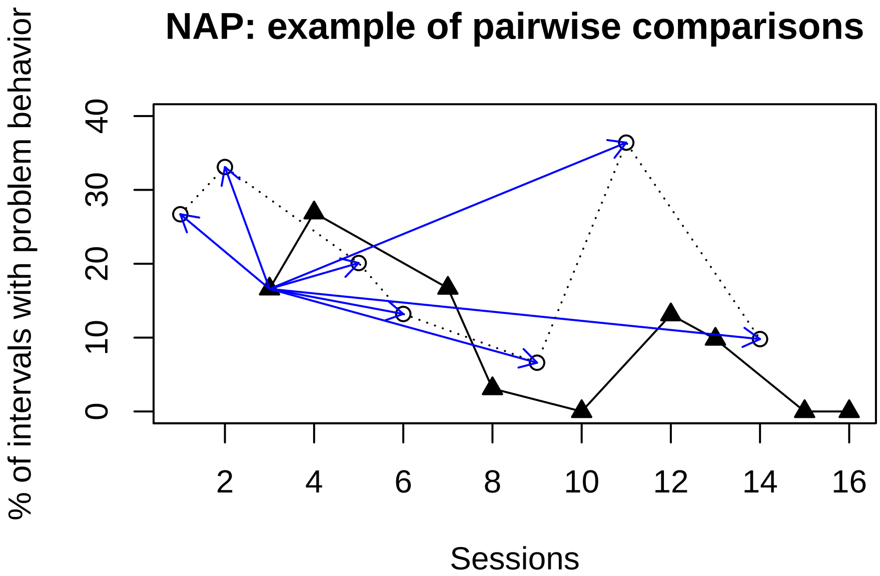


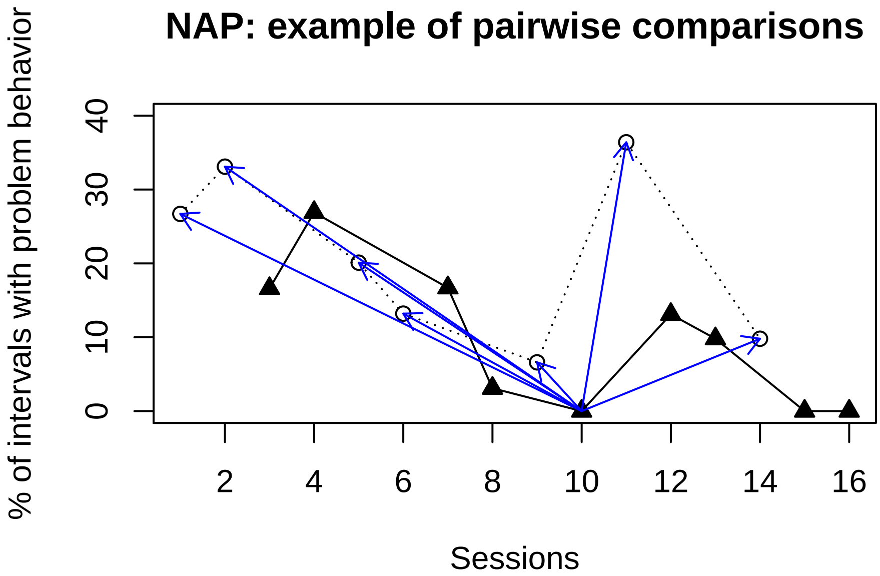
4.6.2 Percentage of nonoverlapping data: Wolery et al. version



4.6.3 Nonoverlap of all pairs

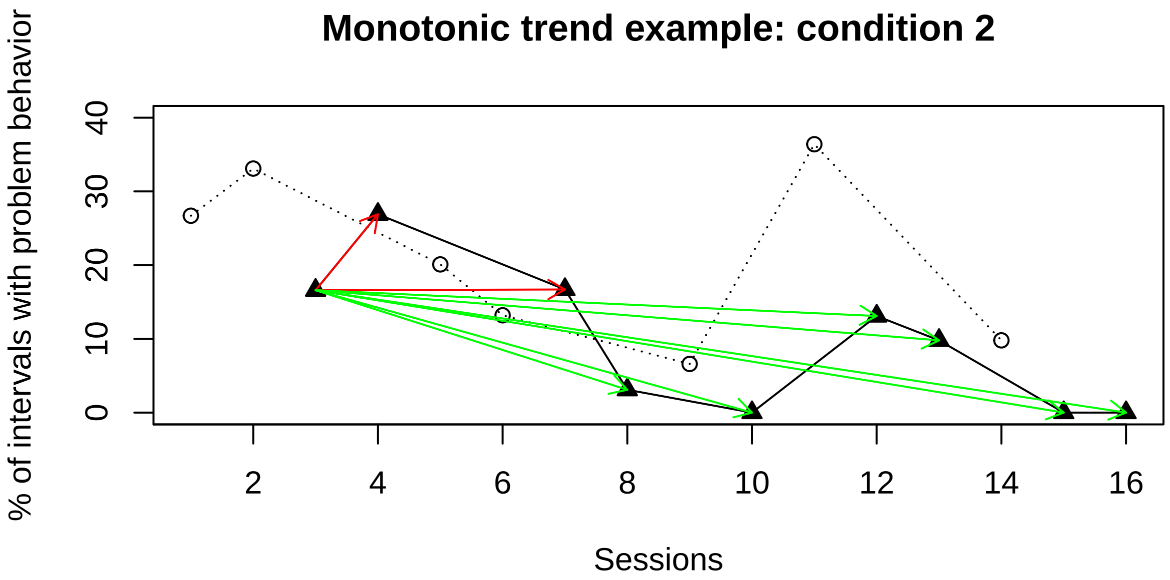
```
## [1] "NAP =  0.77"
```

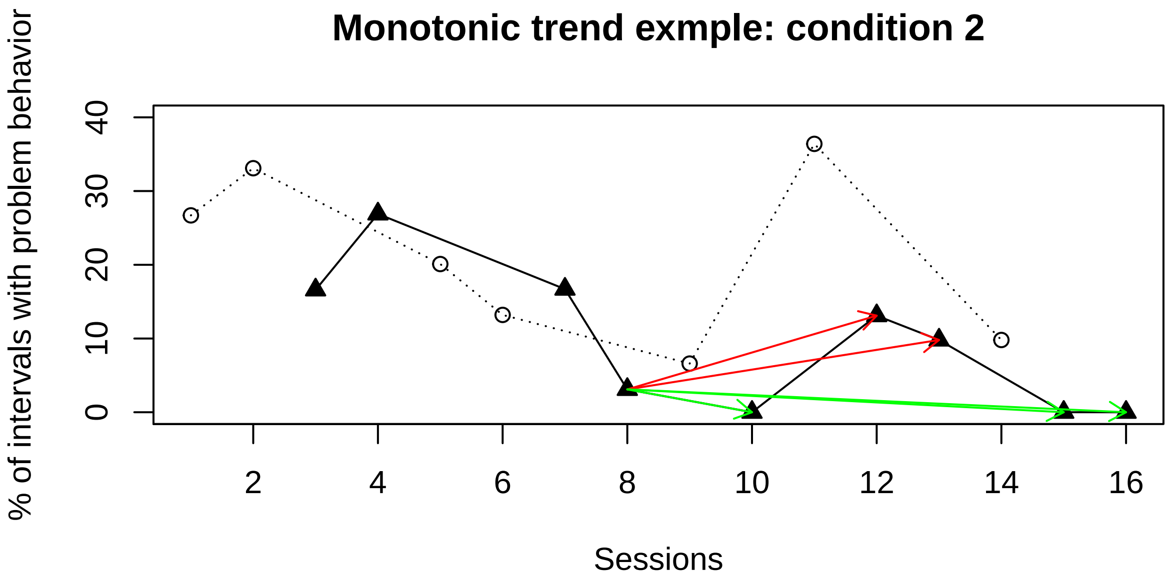




4.6.4 Monotonic trends estimated as in Tau-U

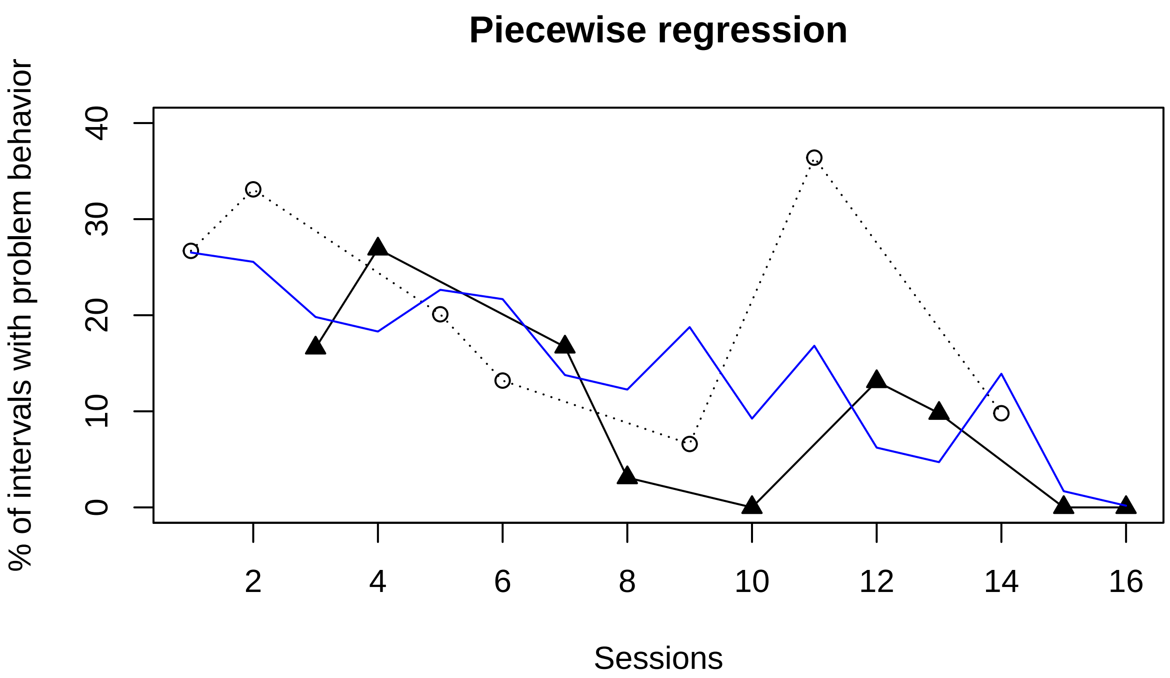
```
## [1] "Condition 2"
## [1] "Negative monotonic trend: 58.33 % decrease"
## [1] "Condition 3"
## [1] "Negative monotonic trend: 33.33 % decrease"
```





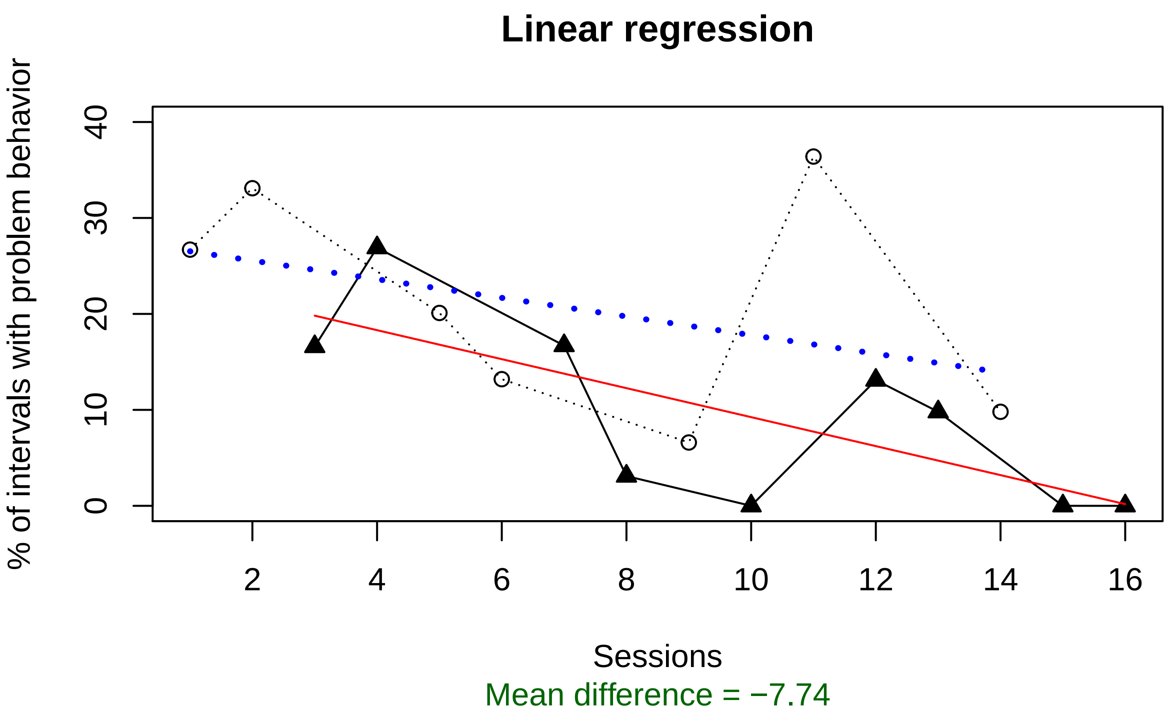
4.6.5 Piecewise regression, comparing at the last measurement occasion

```
## [1] "Comparison 3 vs. 2"
##       Time Score condi_B condi_A Time2 condiA_T2 condiB_T2
##  [1,]    1  26.7       0       1   -15         0       -15
##  [2,]    2  33.1       0       1   -14         0       -14
##  [3,]    3  16.6       1       0   -13       -13         0
##  [4,]    4  26.9       1       0   -12       -12         0
##  [5,]    5  20.1       0       1   -11         0       -11
##  [6,]    6  13.2       0       1   -10         0       -10
##  [7,]    7  16.7       1       0    -9        -9         0
##  [8,]    8   3.1       1       0    -8        -8         0
##  [9,]    9   6.6       0       1    -7         0        -7
## [10,]   10   0.0       1       0    -6        -6         0
## [11,]   11  36.4       0       1    -5         0        -5
## [12,]   12  13.1       1       0    -4        -4         0
## [13,]   13   9.8       1       0    -3        -3         0
## [14,]   14   9.8       0       1    -2         0        -2
## [15,]   15   0.0       1       0    -1        -1         0
## [16,]   16   0.0       1       0     0         0         0
## [1] "  "
```

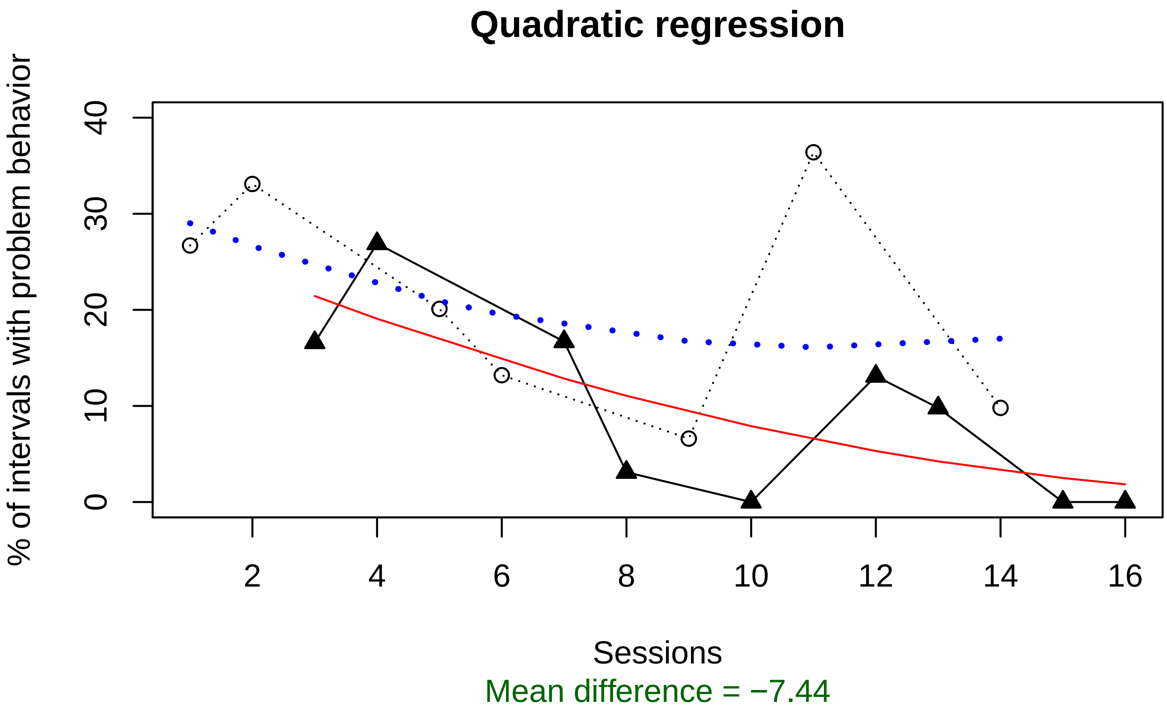



```
## [1] "Condition 2 trend = -1.51"
## [1] "Condition 3 trend = -0.97"
## [1] "Last occasion difference 2-3 = -11.79"
```

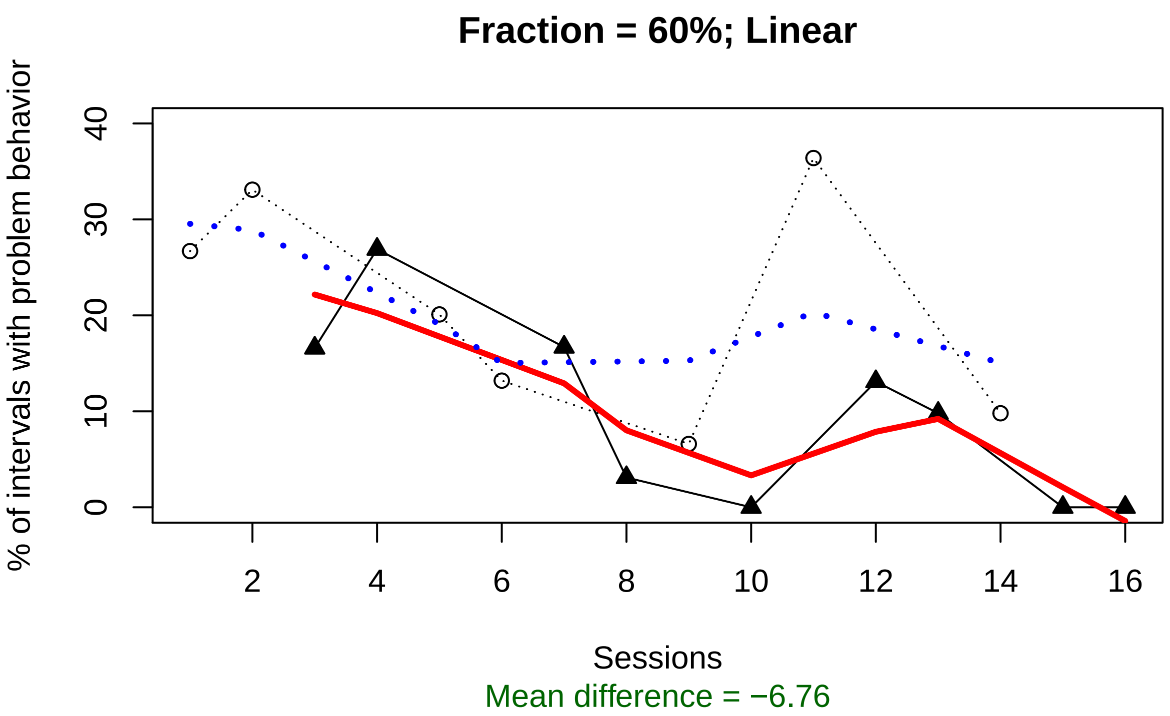
4.6.6 Linear regression: mean difference between fitted values



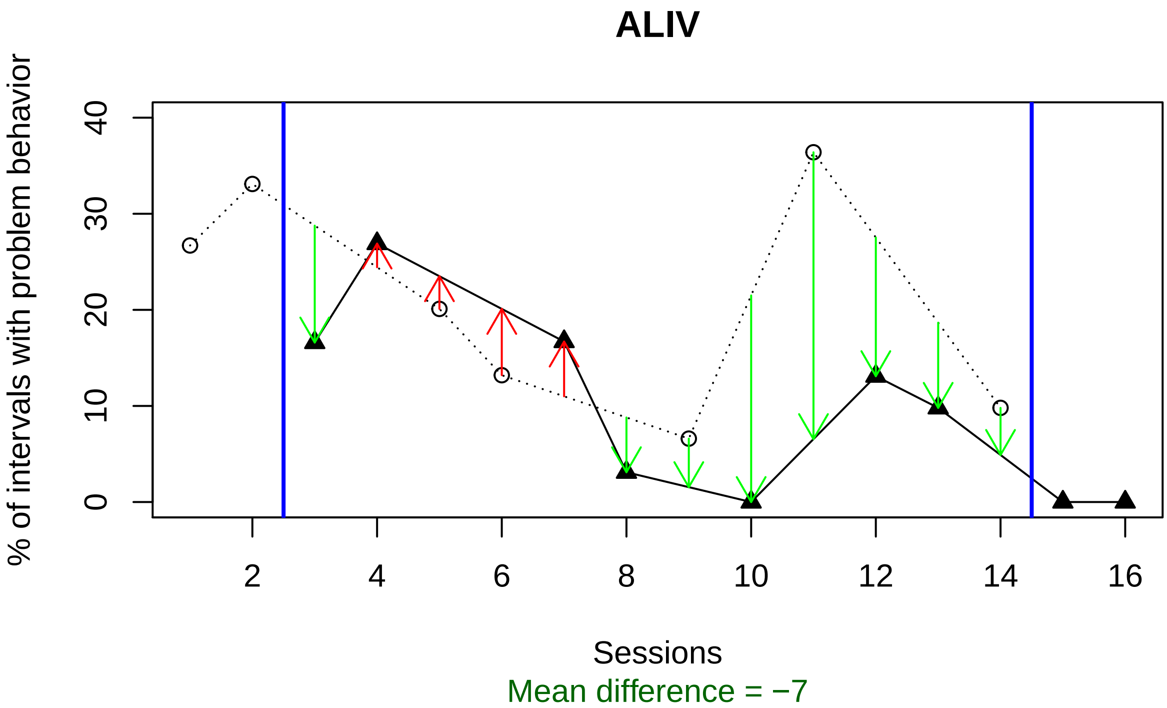
4.6.7 Quadratic regression: mean difference between fitted values



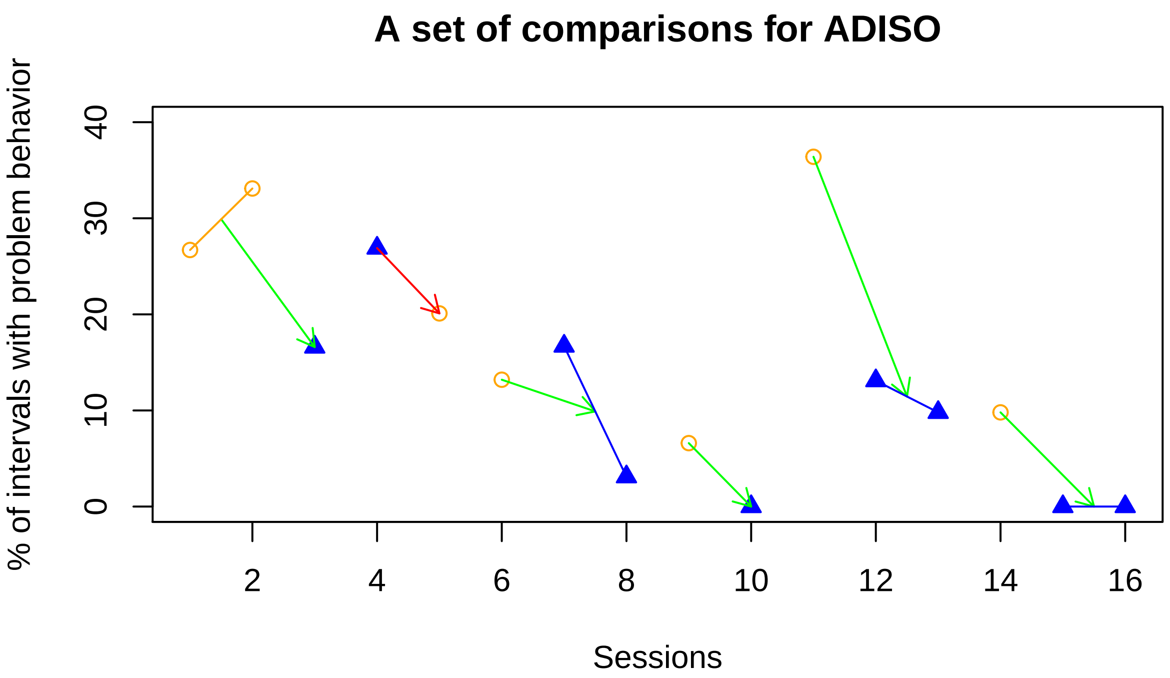
4.6.8 Local regression (LOESS): mean difference between fitted values



4.6.9 Actual and linearly interpolated values: ALIV



4.6.10 Weighted average difference between successive observations: ADISO




```
##       Comparison ADISO  Superiority
## [1,] "2-3"      "-9.6" "1 superior:  16.67 %. 2 superior:  83.33 %"
```