# A small R library which computes precision of descriptive statistics from measurement precision: A companion to "How many decimals?"

Denis Cousineau

Université d'Ottawa

Denis.Cousineau@uottawa.ca

## Loading the library

The exact computations to round descriptive statistics are found in the main text under Table 1. The equations are all simple; however, there are a lot of formulas to remember. To assist in rounding descriptive statistics, I have designed a small R library called MeasurementPrecision (no space and two capital letters) which resides on GitHub. To upload it, first install the devtools library from CRAN (Wickham, Hester & Chang, 2019). Then prior to the first use, issue the commands

devtools::install\_github("dcousin3/MeasurementPrecision")

library(MeasurementPrecision)

On subsequent sessions, you can only use

library(MeasurementPrecision)

## A basic use

Let's assume the following two sets of observations

sample1 <- c(83, 58, 79, 50, 49, 53, 62, 79, 66)  
sample2 <- c(71, 62, 83, 93, 56, 82, 66, 69, 82, 86, 74, 61, 59, 101, 94, 86, 75)

To get a rounded descriptive statistic, use a command named roundMP.*statistic*. For example, to round the mean of the first sample, use:

roundMP.mean(fromData = sample1, deltax=0.5)

where deltax, a mandatory argument, is the precision of the instrument. The command returns a one-line data frame with four columns:

# machine.precision extrinsic systematic non.systematic  
# 64.33333 64 64.3 64.3

where machine.precision is the unrounded result, extrinsic is the extrinsic precision-based rounding; systematic is the result assuming systematic measurement error and non.systematic is the result assuming non-systematic measurement error.

In any of the commands, you can use fromData if you want to specify raw data or fromStatistics to provide already-calculated descriptive statistics (provide them with all the precision you can). For example,

roundMP.mean(  
 fromStatistics = list(mean = 64.333333, sd = 13.20982, n = 9),   
 deltax = 0.5  
)

returns the same results as above. If you issue this command with an empty list of statistics, an error message will let you know which statistics are required.

## Getting rounded statistics beyond the mean

You can also round the standard deviation (sd), the standard error of the mean (semean) and the confidence interval of the mean (cimean):

roundMP.sd(fromData =sample1, deltax = 0.5)  
roundMP.semean(fromData =sample1, deltax = 0.5)  
roundMP.cimean(fromData =sample1, deltax = 0.5)

In roundMP.cimean, add gamma = for a different coverage. For example, gamma = 0.80 will round a 80% confidence interval of the mean.

A one-sample *t*-test requires the null hypothesis for the mean, provided with mu0, for example:

roundMP.t.test(fromData = sample2, mu0 = 65, deltax = 0.5)

where 65 kg is the average planetary body weight for humans.

For statistics on two independent samples, you can use

roundMP.meandiff(fromData = list(sample2, sample1), deltax = 0.5)  
roundMP.sdpool(fromData = list(sample2, sample1), deltax = 0.5)  
roundMP.cohen.d(fromData = list(sample2, sample1), deltax = 0.5)

The argument fromData accepts vectors, matrices, data frames or a list of vectors (as illustrated here).

The non-systematic estimates are by default obtained from the simplifying assumptions described in Appendix B of the main paper. To use the full expression (non-parametric solution), add assumptions=FALSE to any of the commands, for example

roundMP.t.test(fromData = list(sample2, sample1),   
 deltax=0.5, assumptions = FALSE  
)

Generally, there is not much differences whether the simplified or the full expression are used.

## Arguments fromData, fromStatistics and fromObject

Regarding t.test, it is also possible to get a rounded result from a t.test object directly using the argument fromObject instead (instead of fromData or fromStatistics). The input has to contain a t-test, not a Welch test (so use var.equal = TRUE for two-samples):

res <- t.test(sample1, sample2, var.equal = TRUE)  
roundMP.t.test(fromObject = res, deltax = 0.5)

Note that the library MeasurementPrecision must be declared prior to use the t.test function as it is redefined by MeasurementPrecision.

## Detailed output

Finally, to obtain more details on the computations, and see the exact precision, you can add the option verbose=TRUE to any command. For example,

roundMP.mean(fromData = list(sample1), deltax = 0.5, verbose = TRUE)

returns

----------------------------------------------------------------------  
mean of input is: 64.33333  
delta\_x of instrument is: 0.5  
EXTRINSINC PRECISION: (result based on standard error of the mean)  
 - precision for mean is: 4.403282  
 - rounded mean of input is: 64  
SYSTEMATIC ERROR INTRINSINC PRECISION: (result assumption-free)   
 - precision for mean is: 0.50005  
 - rounded mean of input is: 64.3  
NON-SYSTEMATIC ERROR INTRINSINC PRECISION: (result assumption-free)   
 - precision for mean is: 0.1666833  
 - rounded mean of input is: 64.3  
----------------------------------------------------------------------  
 machine.precision extrinsic systematic non.systematic  
1 64.33333 64 64.3 64.3

The last two lines are identical to the solution provided earlier, but detailed information returns the precision for each scenario, whether a simplifying assumption was used, and the resulting rounded result.