

Help on Mike Misc package and related project stuff.

## List of Goodies

- I. Bootstrapping functions
- II. Robust/alternative functions
- III. Other

### General instructions for using:

Specific functions are italicized, example code in red. For group comparisons I provide an example using the Guyer dataset in the car library. For regressions, they take on the same form as you've done so a generic example is given.

To use either the Mike misc or Wilcox functions, right click and save the files to your computer, then in R File/Source R code and select those files.

The bootstrap ones I suspect may give some issues just getting your data the way it needs for inputting (particularly with Cohen's d) but do know that after doing one or two you will get the hang of it for future use. Feel free to ask for assistance.

Regarding robust functions for regression, the main things to examine are the coefficients and the residual standard error compared to the regular regression output. Also note that the `bootregcoeff` function can be modified if you'd like to bootrify your robustified coefficientness. Send me an email if interested.

I. **Bootstrapping functions.** All the bootstrapping functions are meant to be applied as a 'statistic' using the `boot` function from the `boot` library.

A. *bootd*: A bootstrapped Cohen's d.

Requires:

a dataset with just the DV and the grouping variable e.g.

```
mydat=data.frame(Depvar,groupvar)
```

The `boot` package. If you already have it installed this package, sourcing the Mike misc package I think will load it automatically.

Used for: bootstrapping the Cohen's d interval when assumptions don't hold

Example usage with the Guyer dataset from car library

```
attach(Guyer)
myboot=boot(Guyer[,c("cooperation","condition")],statistic=bootd,R=1000,strata=
Guyer[, "condition"])
myboot      #output
mean(myboot$t) #how does your bootstrapped mean d compare with the original?
plot(myboot)  #histogram and QQ plot of the bootstrapped ds. The statistic is
labeled 't*'
boot.ci(myboot,type="bca") #same old bootstrap CI we've discussed but 'bias-
corrected'
```

Explanation of 1<sup>st</sup> line. Form is `boot(data, statistic, R, strata=groupvar)`. For the data I told it I wanted the Guyer data but just those two variables in the form of `(Depvar,groupvar)`. The

statistic is the `bootd` function in the package, the `R` is the number of resamples (i.e. bootstrapped samples), and `strata` is used because we are dealing with multiple samples.

- B. *bootcor*: a bootstrapped correlation coefficient. Note that a simple correlation does not qualify as one of your required analyses.

```
#Generic example
mydata=data.frame(var1,var2)
myboot=boot(mydata,bootcor,R=1000)
myboot
mean(myboot$t)
plot(myboot)
boot.ci(myboot, type="bca")
```

- C. *bootregcoeff*: bootstrapped regression coefficients. This will have output for each coefficient. Tips data example is given so you know what to expect.

```
attach(tips)
myboot=boot(tips, bootregcoeff, R=1000, formula=TIP~TOTBILL)
myboot
plot(myboot, index=1) # intercept
plot(myboot, index=2) # predictor coefficient

boot.ci(myboot, type="bca", index=1) # CI limit for the intercept
boot.ci(myboot, type="bca", index=2) # CI for the predictor coefficient
```

- D. *bootRsq*: bootstrapped R-squared

```
attach(tips)
myboot=boot(tips, bootRsq, R=1000, formula=TIP~TOTBILL)
myboot
plot(myboot)
boot.ci(myboot, type="bca")
```

## II. Robust/alternative approaches

### A. Robust t-test stuff

- i. *yuenbt*: robust t-test based on trimmed means and winsorized variance.  
Requires: Wilcox package (must be “sourced”). The Rallfun files are downloadable from the class datacode page. Just download and from the R menu File/Source R code.  
Used for: independent samples t-test with heterogeneity of variance and/or outliers

```
attach(Guyer)
yuenbt(Guyer$cooperation[condition=="A"], Guyer$cooperation[condition=="P"],
side=T, alpha=.05, nboot=599)
```

Returns a CI for the trimmed mean difference, the t-statistic, and the p-value if you put `side=T`.

- ii. *robustd*: A robust d  
Requires:  
winvar function which is in the Mikemisc package already (taken from the Wilcox library)  
Used for: a Cohen's d in cases of problematic assumptions, in particular, heterogeneity of variance and outliers. Uses trimmed means and winsorized variance.

## Example usage with the Guyer dataset from car library

```
attach(Guyer)
robustd(Guyer$cooperation[condition=="A"], Guyer$cooperation[condition=="P"])
```

### B. Robust Regression stuff

#### i. *lmRob*: robust regression

Requires: library(MASS)

Used for: regression in cases of heteroscedasticity and/or outliers

Stuff you won't recognize: a test of bias for the estimates.

```
mymodel=lmRob(DV~Pred,data=mydata)
plot(mymodel) #You have several options, hit escape at the console when done.
summary(mymodel)
```

#### ii. *lmrob*

Requires: library(robustbase)

Used for: regression in cases of heteroscedasticity and/or outliers

Stuff you won't recognize: part of the output are 'algorithmic parameters' for which you do not need to worry about. However do note the residual weights and see how some cases will get full weight  $\approx 1$  while others might get very low. A full outlier would get near zero weight and it would notify you of any that qualify as such.

```
mymodel=lmrob(DV~Pred,data=mydata)
plot(mymodel) #again several options. Clicking on the graphs cycles through them.
summary(mymodel)
```

#### iii. *rlm*

Requires: library(robust)

```
mymodel=rlm(DV~Pred,data=mydata)
plot(mymodel) #same stuff as 'Basic diagnostic plots' that you did with Rcmdr
summary(mymodel)
```

### C. Bayesian regression: library(MCMCpack)

#### i. MCMCregress

```
mymodel = MCMCregress(DV~Pred, data=mydata)
plot(mymodel) #plots the distribution of the parameters fitted to the data
summary(mymodel)
```

Explanation (what little I can give). MCMC stands for Markov Chain Monte Carlo (also Mike Clark's Methodological Chicanery) which involves simulation from a distribution of some kind (e.g. normal, aka "Gaussian", for the coefficients). It is in the spirit of bootstrapping but think of it as sampling with respect to a known or assumed distribution rather than our data, and the focus is on letting the parameters vary while holding the data constant. However, in the summary you will find things you are familiar with.

The means refer to the average parameter value (i.e. the average coefficient) and the sd are the standard errors for them (divide one by the other for your t-statistic if desired). The sigma2 is your mean square error that you would see in an ANOVA table, the square root of which is the standard error of estimate (or residual standard error). I'm not sure what the naive/time series well enough so don't bother with

that. If I understand Rich correctly we want them pretty low for this type of data.

The quantiles at 2.5% and 97.5% for the coefficients provide 95% *credible* intervals, not confidence intervals. That is *the* interval you would expect the population parameters to 'fall into' 95% of the time (again, in the Bayesian approach the model, i.e. the parameters are random and assumed to vary, rather than be fixed in the population). Since this is a naïve approach, i.e. we are going into it 'uninformed' and without prior information, it won't differ much from the regular confidence interval, but it will be wider.

### III. Other stuff

#### A. Inferential confidence intervals

##### i. *inf.conf.int*

Example with Guyer data set in the car library

```
inf.conf.int(Guyer$cooperation[condition=="A"], Guyer$cooperation[condition=="P"],  
conf.level=.95, paired=F)
```

Explanation of output. Provides Tryon's inferential confidence intervals for each group with a statement on whether they overlap or not. For independent samples (paired=F), it will also provide an automatic equivalence test comparing the Range based on the group differences to an equivalence range based on a mean difference of Cohen's  $d = .20$  and the variance exhibited in the data (it takes the lower mean, finds the mean  $.20$  std dev higher and creates a range from that). Since both ranges start with the same mean for the lesser group, they will have the same lower limit. A statement of equivalence is given if the data's difference range is less than the equivalence range.

If dealing with paired samples, the inferential confidence intervals are calculated. However, an equivalence test is not given because Cohen's  $d$  may be calculated in more than one way in that situation. I may implement an option in the future for doing so, but one of the ways you could calculate Cohen's  $d$  is to treat them as if they were independent, in which case just put paired = F to get the equivalence test.

#### B. Vector difference norm.

##### i. *vectdiff*

Requires: equal N samples, no missing values.

Used for: paired t-test situation

```
vectdiff(sample1, sample2)
```

Explanation of output. See the additional notes on comparing two groups on the class website. Will provide the vector difference norm, which is a single value of little interpretability. However, if you then take the first group and create a second group by multiplying it by some amount (e.g. 1.1 would produce a new group of scores 10% greater, multiplying by .75 would produce a new group of scores 75% less), you will get a number output as well. Through a bit of trial and error you'll eventually settle on one that is close to the original, and now you will get a sense of how much the scores differ/change on average. Especially useful for the paired t-test scenario.

```
vectdiff(sample1, sample1*.75)
```