Bootstrapped Welch *t*-test

E-mail distributed on 07-10-2024

Dear all,

A widely held misconception about non-parametric tests is that they do not assume homogeneity of variances. In fact, common non-parametric tests such as permutation tests and rank tests also make this assumption, and are therefore not appropriate for dealing with unequal variances, a problem more generally known as heteroscedasticity (e.g., in groups, or in model residuals). Tests that do correct for heteroscedasticity, typically do so by introducing further parameters, so should themselves not be considered as "non-parametric".

Consider the problem of comparing two independent means with the classical *t*-test. Its heteroscedastic version is known as the Welch *t*-test (see my earlier newsletter for an extensive discussion). Intuitively, one would assume that one can construct a permutation Welch *t*-test simply by permuting the data and extracting the Welch *t*-value instead of the original *t*-value, as a basis to calculate an empirical *p*-value. However, this approach would fail due to the fact that permuting data removes not only mean differences between the groups, but also variance differences. That is, the permutations will not preserve the heteroscedasticity of the original data, and we would end up basically with a conventional permutation *t*-test.

One solution is to use a resampling method that does preserve heteroscedasticity, such as the bootstrap. The bootstrap achieves this because it keeps the group labels and outcome values together when resampling, whereas permutations only resample group labels or outcome values, without regard to the other. It would be logical therefore to consider a bootstrapped Welch *t*-test. In R, such a test could be constructed by manually resampling the data in a for-loop. However, calculating the correct confidence intervals requires mathematical adjustments which are slightly more complex. For this reason, it is better to rely on the functions `boot` and `boot.ci` from the `boot` package. The attached script does just that.

Bootwelch function

To define the `bootwelch` function in your R session, one either loads the script manually, or from its file location with the source function. Once defined, the function is then run simply as follows:

```
bootwelch(y=..., g=..., nsim=999)
```

Where `y` is the outcome data, `g` is a grouping factor linked to the outcome values, and `nsim` sets the number of bootstrap resamples (defaults to 999[1]). For a toy example, the output looks as follows:

```
set.seed(124)
group1 <- rnorm(100, mean=0, sd=2)
group2 <- rnorm(100, mean=2, sd=6)
test <- data.frame(group=rep(0:1,each=100), y=c(group1,group2))
bootwelch(y=test$y, g=test$group, nsim=999)

> ------------------------------------------
> Bootstrapped Welch t-test (999 resamples)
> ------------------------------------------
>
> Welch t: -2.455, BCA 95%CI: [-4.467, -0.434]
>
> Minimal bootstrapped Welch DF: 82.97 (mean: 119.89)
> Minimal theoretical p-value: 0.0162
>
> Adjusted Cohen's d: 0.35
```

At the top, we have conventional bootstrap output, with the original Welch *t*-value and the accelerated bias-corrected (ABC) bootstrap 95% confidence interval.[2] Normally, one checks whether this interval includes 0 to reject the null hypothesis. Unlike a permutation test, which outputs solely a *p*-value, a bootstrap test typically outputs a confidence interval. For the toy data, the interval does not contain 0, therefore we would conclude that the group means are significantly different.

The function prints an additional result using a different approach to the ABC interval. Alongside bootstrapping the Welch *t*-value for every resample, the function also bootstraps the adjusted Welch degrees of freedom. The minimal adjusted DF is then used to calculate a "theoretical" minimal bootstrapped *p*-value for the original observed Welch *t*-value, as a kind of worst-case scenario (maximal heteroscedasticity for these data). For the toy data, the minimal DF is as low as 82.97, when the total original sample size was 200! **Be cautioned** that this output is experimental, however, since I cannot vouch that boostrapping DFs is defensible theoretically, and moreover, this approach is once again parametric in assuming the referenced *t*-distribution.

Finally, the function also returns a Cohen's *d* as a standardized effect, adjusted for heteroscedasticity, as discussed in the newsletter on this subject.

Is it useful?

While you are welcome to use the attached script, the practical utility of this test may be somewhat limited. My experience in general has been that non-parametric tests rarely deviate substantially from their parametric counterparts, especially permutation tests and bootstrap tests. For bootstrapping, its advantage lies primarily in producing confidence intervals for parameters and statistics that otherwise have no (straightforward) parametric counterpart. Rank tests sometimes differ from non-rank tests because outliers are affecting the data.

[1] Should be larger for very large samples
[2] Sometimes also called BCA interval

For the bootstrapped Welch *t*-test, the paradox is that it may be redundant in both large and small samples. That is, for large samples, it will probably produce results that are almost identical to the parametric Welch *t*-test. For small samples, the bootstrapped Welch *t*-test **should be avoided altogether**! This is because bootstrapping increases small-sample bias, and one risks therefore producing a more biased test. The primary scenario where this test will be useful is in data affected both by heteroscedasticity and severe non-normality. In every other scenario, the parametric Welch *t*-test should be preferred. Having the bootstrapped version will serve mainly as a backup for critical reviewers in papers.

Finally, note that the `bootwelch` function was written for quick implementation. It could be extended to the general Welch test for more than 2 groups, and to multiple regression models with heteroscedastic residuals. In fact, bootstrapping under heteroscedasticity is an active area of statistical research. Unfortunately, implementations in R currently have drawbacks, with package `lmboot` not producing bootstrap confidence intervals, and package `fwildclusterboot` defunct since April this year. If you require something more specific, general, or modified, I recommend to contact me directly.

--

Ben Meuleman, Ph.D.
Statistician
Swiss Center for Affective Sciences
University of Geneva | Campus Biotech
Chemin des Mines 9 | CH-1202 Genève
ben.meuleman@unige.ch | +41 (0)22 379 09 79