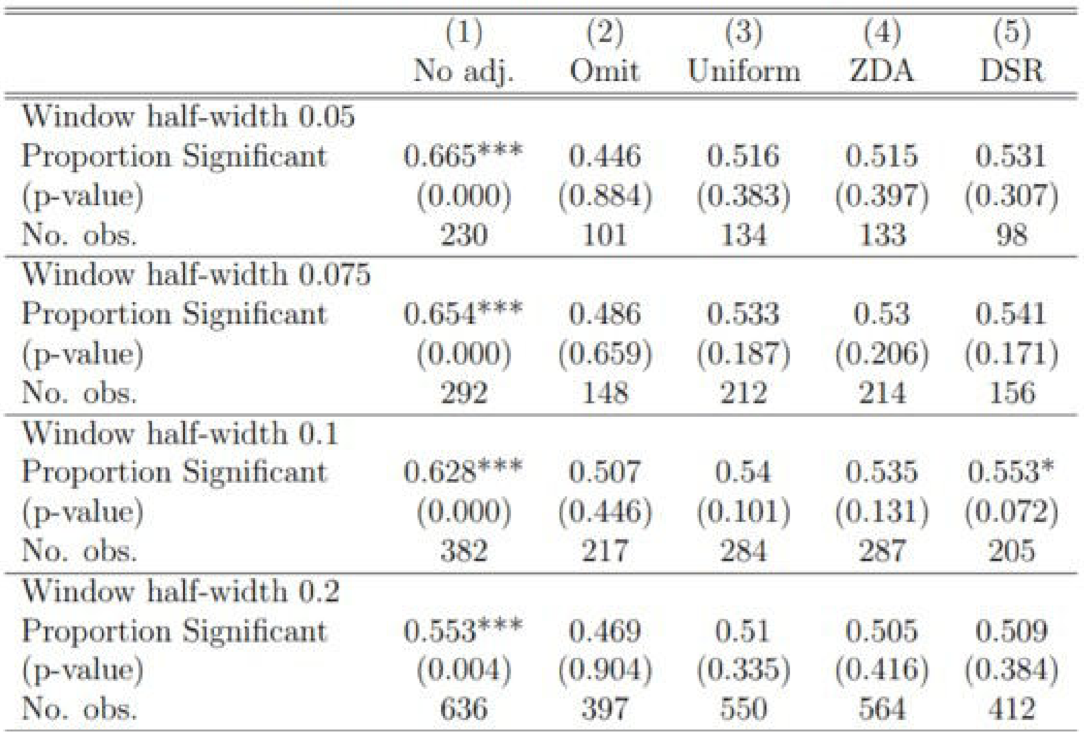
…we needed to generate several customized Latex tables, like the following excerpt:

Table 5: Randomization tests for DID subsample.



	(1) No adj.	(2) Omit	(3) Uniform	(4) ZDA	(5) DSR
Window half-width 0.05					
Proportion Significant	0.665***	0.446	0.516	0.515	0.531
(p-value)	(0.000)	(0.884)	(0.383)	(0.397)	(0.307)
No. obs.	230	101	134	133	98
Window half-width 0.075					
Proportion Significant	0.654***	0.486	0.533	0.53	0.541
(p-value)	(0.000)	(0.659)	(0.187)	(0.206)	(0.171)
No. obs.	292	148	212	214	156
Window half-width 0.1					
Proportion Significant	0.628***	0.507	0.54	0.535	0.553*
(p-value)	(0.000)	(0.446)	(0.101)	(0.131)	(0.072)
No. obs.	382	217	284	287	205
Window half-width 0.2					
Proportion Significant	0.553***	0.469	0.51	0.505	0.509
(p-value)	(0.004)	(0.904)	(0.335)	(0.416)	(0.384)
No. obs.	636	397	550	564	412

While all the data can be easily merged into a data frame in R, creating this fairly customized Latex table from a reproducible workflow seemed less straightforward. For example, building the table by pasting R strings together was inconvenient. One reason is that Latex has many backslashes \, which must be converted to double backslashes \\ inside R strings, which makes things hard to read and write.

xglue solves this problem by having templates as files that include placeholders and blocks that allow to collapse vectorized expressions or perform group_by operation on data frames to combine strings. The template for the table above can be found here.

Simpler examples that explain the usage are given in the vignette here.

Installation from R-universe

The package is hosted on R-universe. To install it run:

```
options(repos = c(
    skranz = 'https://skranz.r-universe.dev',
    CRAN = 'https://cloud.r-project.org'))
install.packages('xglue')
```

Personally, I am quite excited by the new R-universe service. It is still in its pilot phase, though. Jeroen Ooms, who kindly allowed me as a test to add non-CRAN packages, wrote in April:

> During the current pilot, R-universe mostly serves existing CRAN packages. But soon we will start letting users add other packages to their universe, such that eventually anyone can have their personal space of R packages and articles.
>
> The biggest difference with official archives like CRAN, is that in your own universe,

> there are no policies that restrict the sort of packages you may publish. You could add some experimental projects, or research compendium packages, or even packages that solely consist of a vignette containing a tutorial or writeup. As long as the package is available from git, the R-universe infrastructure will automatically build binaries and articles, every time you git-push changes.

More concretely, unlike CRAN, R-universe does not require to resolve every NOTE of `R CMD check` to build binaries. To successfully build the binaries, it just has to pass without error.

Of course, there is always a trade-off between quality tests on the one hand and speed and convenience of package development (and package updates) on the other hand. For R in general, it is absolutely great to have a time-tested, high quality archive with CRAN. (And indeed, one probably can find several things in my non-CRAN packages that would be improved if they were adapted to meet the CRAN requirements.)

But I also feel that many NOTES in `R CMD check` rather seem a matter of taste and my taste is not always the same. When I was hosting a package on CRAN, I felt that I was really reluctant to push an updated version on CRAN in fear that with a new R version CRAN policies have changed and created a bunch of new NOTES that have to be resolved before I can upload my updated version. (Indeed, for the sake of saving time, I rather don't want to update to every major new R version but skip some, as long as everything still works fine. But that means that I would not necessarily see all new NOTES on my local computer when preparing a CRAN submission.)

So I really love that with R-universe there will be a new option for package hosting that makes developers more responsible for their own quality control and automatizes the build process.