Infer: An R Package for Tidyverse-Friendly Statistical Inference

Simon P. Couch  
Johns Hopkins University

Andrew P. Bray  
University of California - Berkeley

Chester Ismay  
Flatiron School

Evgeni Chasnovski

B. Baumer  
Smith College, bbaumer@smith.edu

See next page for additional authors

Follow this and additional works at: https://scholarworks.smith.edu/sds_facpubs

Part of the Data Science Commons, Other Computer Sciences Commons, and the Statistics and Probability Commons

Recommended Citation

Couch, Simon P.; Bray, Andrew P.; Ismay, Chester; Chasnovski, Evgeni; Baumer, B.; and Cetinkaya-Rundel, Mine, "Infer: An R Package for Tidyverse-Friendly Statistical Inference" (2021). Statistical and Data Sciences: Faculty Publications, Smith College, Northampton, MA.  
https://scholarworks.smith.edu/sds_facpubs/45

This Article has been accepted for inclusion in Statistical and Data Sciences: Faculty Publications by an authorized administrator of Smith ScholarWorks. For more information, please contact scholarworks@smith.edu
infer: An R package for tidyverse-friendly statistical inference

Simon P. Couch¹,², Andrew P. Bray³, Chester Ismay⁴, Evgeni Chasnovski⁵, Benjamin S. Baumer⁶, and Mine Çetinkaya-Rundel²,⁷

¹ Johns Hopkins, Department of Biostatistics ² RStudio ³ UC Berkeley, Department of Statistics and Reed College Mathematics Department (on leave) ⁴ Flatiron School ⁵ No Affiliation ⁶ Smith College, Program in Statistical & Data Sciences ⁷ Duke University, Department of Statistical Science

Summary

infer implements an expressive grammar to perform statistical inference that adheres to the tidyverse design framework (Wickham et al., 2019). Rather than providing methods for specific statistical tests, this package consolidates the principles that are shared among common hypothesis tests and confidence intervals into a set of four main verbs (functions), supplemented with many utilities to visualize and extract value from their outputs.

Statement of Need

Packages implementing methods for basic statistical inference in R are highly variable in their interfaces. The structure of inputted data, argument names, expected argument types, argument orders, output types, and spelling cases varies widely both within and among packages. This diversity in approaches obscures the intuition shared among common inferential procedures, makes details of usage difficult to remember, and prevents an expressive and idiomatic coding style.

infer is an R package for randomization-based hypothesis testing, naturalizing an intuitive understanding of statistical inference via a unified and expressive grammar. Four functions provide functionality encompassing a large swath of basic frequentist statistical inference, abstracting away details of specific tests and shifting the focus of the analyst to the observed data and the processes that generated it. Such a grammar lends itself to applications in teaching, data pedagogy research, applied scientific research, and advanced predictive modeling. For one, the principled approach of the infer package has made it an especially good fit for teaching introductory statistics and data science (Baumer et al., 2020; Çetinkaya-Rundel & Ellison, 2021; Ismay & Kim, 2019) and research in data pedagogy (Fergusson & Pfannkuch, 2021; Loy, 2021). Further, the package has already seen usage in a number of published scientific applications (Ask et al., 2021; Fallon & Hinds, 2021; McLean et al., 2021). Finally, the package integrates with the greater tidymodels collection of packages, a burgeoning software ecosystem for tidyverse-aligned predictive modeling used across many modern research and industrial applications (Kuhn & Wickham, 2020). To date, the package has been downloaded more than 400,000 times.
Underlying Principles

Regardless of the hypothesis test in question, an analyst asks the same kind of question when conducting statistical inference: is the effect/difference in the observed data real, or due to random chance? To answer this question, the analyst begins by assuming that the effect in the observed data was simply due to random chance, and calls this assumption the null hypothesis. (In reality, they might not believe in the null hypothesis at all—the null hypothesis is in opposition to the alternate hypothesis, which supposes that the effect present in the observed data is actually due to the fact that “something is going on.”) The analyst then calculates a test statistic from the data that describes the observed effect. They can use this test statistic to calculate a p-value via juxtaposition with a null distribution, giving the probability that the observed data could come about if the null hypothesis were true. If this probability is below some pre-defined significance level $\alpha$, then the analyst can reject the null hypothesis.

The workflow of this package is designed around this idea. Starting out with some dataset,

- `specify()` allows the analyst to specify the variable, or relationship between variables, that they are interested in.
- `hypothesize()` allows the analyst to declare the null hypothesis.
- `generate()` allows the analyst to generate data reflecting the null hypothesis or using the bootstrap.
- `calculate()` allows the analyst to calculate summary statistics, either from
  - the observed data, to form the observed test statistic.
  - data `generate()`d to reflect the null hypothesis, to form a randomization-based null distribution of test statistics.

As such, the ultimate output of an infer pipeline using these four functions is generally an observed statistic or null distribution of test statistics. These four functions are thus supplemented with several utilities to visualize and extract value from their outputs.

- `visualize()` plots the null distribution of test statistics.
  - `shade_p_value()` situates the observed statistic in the null distribution, shading the region as or more extreme.
- `get_p_value()` calculates a p-value via the juxtaposition of the test statistic and the null distribution.

The workflow outlined above can also be used for constructing confidence intervals via bootstrapping with the omission of the `hypothesize()` step in the pipeline. The resulting bootstrap distribution can then be visualized with `visualize()`, the confidence interval region can be situated in the bootstrap distribution with `shade_confidence_interval()`, and the bounds of the confidence interval can be calculated with `get_confidence_interval()`.

Beyond this, the `infer` package offers:

- methods for inference using theory-based distributions
- shorthand wrappers for common statistical tests using tidy data
- model-fitting workflows to accommodate multiple explanatory variables

Comparison to Other Packages

Several software packages on the Comprehensive R Archive Network share functionality with infer (R Core Team, 2021). broom and parameters convert model objects to unified output formats, though they do not provide methods for fitting models, describing null distributions, performing bootstrapping, or calculating summary statistics from tabular data (Lüdecke et al., 2020; Robinson et al., 2021). statsExpressions, and adjacent packages in the easystats ecosystem, implement wrappers with consistent interfaces for theory-based hypothesis tests (Patil, 2021). Similarly, mosaic is a package used to teach statistics by unifying summary statistics, visualization, and modeling with a consistent API built around R’s formula interface. The mosaic package also includes functionality to conduct randomization-based inference (Pruim et al., 2017). At a higher level, though, the structure of each of these packages is defined by model types and statistics, where each model type or statistic has its own associated function and/or object class. In contrast, infer is structured around four functions, situating statistics and model types within a more abstracted grammar.¹

Acknowledgements

We acknowledge contributions from Albert Y. Kim, Jo Hardin, Jay Lee, Amelia McNamara, Nick Solomon, and Richie Cotton.

References


¹This grammar follows from Allen Downey’s “there is only one test” framework (Downey, 2016).


