

A Python module to combine p values arising from discrete tests.

Gerrit Ansmann ¹

¹ Institute for Biological Physics, University of Cologne, Germany

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Vissarion Fisikopoulos](#) 

Reviewers:

- [@steppi](#)
- [@mdhaber](#)

Submitted: 27 October 2023

Published: unpublished

License

Authors of papers retain copyright
and release the work under a
Creative Commons Attribution 4.0
International License ([CC BY 4.0](#))

Summary

Most popular hypothesis tests are designed for datasets with a simple structure. For example, in a simple medical trial, this structure would be a treatment and a control group, and a test would answer whether some health score exhibits a significant difference between their members. However, many datasets are naturally segmented or benefit from being analysed in segments. Continuing our example, if we know a priori that females score higher than males on average, analysing each sex separately should allow us to see more clearly whether the treatment is effective. To draw an overall conclusion in such a case, we need to combine the p values for each sub-dataset.

Starting with Fisher's method, several ways have been proposed to combine p values ([Heard & Rubin-Delanchy, 2018](#)). Their usual implementations assume *continuous tests*, i.e., tests whose p values under the null hypothesis follow the uniform distribution on the unit interval. However, rank tests and many others are *discrete tests*, i.e., they can only yield p values from a finite selection (for a given sample size). Assuming continuous tests when combining p values from discrete tests can lead to considerably misleading outcomes ([Kincaid, 1962](#); [Mielke et al., 2004](#)), in particular when the sub-datasets are small.

We here present the Python module `combine_pvalues_discrete`, which provides a toolbox for combining p values from discrete tests. Results from individual tests are stored with the respective null distribution of p values and the combined p value is accurately estimated using a Monte Carlo simulation based on these null distributions.

Statement of need

Combining p values is a standard problem in statistical data analysis. Before the advent of modern computing, many methods have been devoted to analytically solving this problem for continuous tests ([Heard & Rubin-Delanchy, 2018](#)), and implementations of these simple methods are a staple of many statistical software suits ([Cinar & Viechtbauer, 2022](#); [Dewey, n.d.](#); [SciPy's 'Combine_pvalues', n.d.](#); [Yarkoni et al., 2022](#)). However, applying these implementations to discrete tests can result in considerable errors ([Kincaid, 1962](#); [Mielke et al., 2004](#)). Discrete tests particularly include all rank tests, which are an appropriate choice in many applications as they do not make any assumptions of normality or similar. There appears to be little awareness of this problem, and we are not aware of a software solution addressing it.

Thanks to modern computing, we can solve this problem using simple Monte Carlo simulations: For each test, we sample one p value from each of the respective discrete null distributions, apply the combining statistics to these, and repeat this until we obtain a good estimate of the null distribution of the combining statistics. Finally, we compare the combining statistics of the actual data to estimate the combined p value. The presented module `combine_pvalues_discrete` implements this approach in a fast, thorough, and tested manner:

41 It takes care of pitfalls such as correctly handling complements, sidedness, and p values from
42 a Monte Carlo estimate of the null distribution (Phipson & Smyth, 2010). Also, it handles
43 tedious and error-prone tasks such as determining the null distribution p values for a given test
44 and sample size. By storing and sampling the null distribution of p values for each sub-dataset,
45 our approach is considerably faster than using a Monte Carlo null model for the entire dataset
46 by combining samples from Monte Carlo null models of the sub-datasets.

47 Note that this module de-emphasises some typical applications of combining p values in which
48 we expect known discrete null distributions of p values to rarely occur. For example, when
49 performing a meta analysis of existing studies, the null distributions of p values are either
50 continuous, close to it, or not known. However, as a side product, our module contains
51 weighted versions of popular combining methods that may be of interest to researchers
52 combining continuous tests.

53 Acknowledgements

54 I am grateful to I. Wielert for constructive comments on earlier versions of this manuscript
55 and the documentation.

56 References

- 57 Cinar, O., & Viechtbauer, W. (2022). The 'poolr' package for combining independent and
58 dependent p values. *Journal of Statistical Software*, 101(1), 1–42. <https://doi.org/10.18637/jss.v101.i01>
59
- 60 Dewey, M. (n.d.). *Metap: Meta-analysis of significance values*. <https://cran.r-project.org/package=metap>
61
- 62 Heard, N. A., & Rubin-Delanchy, P. (2018). Choosing between methods of combining p -values.
63 *Biometrika*, 105(1), 239–246. <https://doi.org/10.1093/biomet/asx076>
- 64 Kincaid, W. M. (1962). The combination of tests based on discrete distributions. *Journal of*
65 *the American Statistical Association*, 57(297), 10–19. <https://doi.org/10.1080/01621459.1962.10482147>
66
- 67 Mielke, P. W., Johnston, J. E., & Berry, K. J. (2004). Combining probability values from
68 independent permutation tests: A discrete analog of Fisher's classical method. *Psychological*
69 *Reports*, 95(2), 449–458. <https://doi.org/10.2466/pr0.95.2.449-458>
- 70 Phipson, B., & Smyth, G. K. (2010). Permutation p -values should never be zero: Calculating
71 exact p -values when permutations are randomly drawn. *Statistical Applications in Genetics*
72 *and Molecular Biology*, 9(1), 39. <https://doi.org/10.2202/1544-6115.1585>
- 73 SciPy's 'combine_pvalues'. (n.d.). https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.combine_pvalues.html
74
- 75 Yarkoni, T., Salo, T., Peraza, J. A., & Nichols, T. E. (2022). *PyMARE: Python meta-analysis*
76 *& regression engine*. <https://doi.org/10.5281/zenodo.4266738>