The Journal of Open Source Software

meow: An R package for conducting simulations of computer adaptive testing (CAT) algorithms

Klint Kanopka [1¶] **and Sophia Deng** [1]

1 New York University, USA ¶ Corresponding author

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

Computer adaptive testing (CAT) is an approach where the responses an examinee have given to questions influence the next questions they will be presented with. This style of testing allows for increased measurement precision even though individuals respond to fewer items, a benefit for test administrators and test takers alike. Advances in CAT are done at the algorithmic level; new methods are developed to better select next items, update person and item parameters, control how often items are exposed to respondents, and decide when to stop administering new questions. Determining the potential consequences of these new algorithms requires developers to run comparison simulations that demand porting the methods of other researchers into ad hoc frameworks that are often designed for single simulation studies. With `meow`, users can modularize different components of the CAT administration process and conduct standardized, comparable, and reproducible simulation studies.

Statement of need

`meow` is a package written in R to facilitate psychometric research in the computer adaptive testing (CAT) space that focuses on the development of new CAT algorithms. Software to simulate CAT data already exists, the most popular of which are `mirtCAT` (Chalmers, 2016) and `catR`(Magis & Barrada, 2017). The key to understanding the purpose of these packages (and the gap that `meow` fills) is that these packages primarily exist to simulate the administration of CATs and compare test-designs. As such, these packages come with a selection of in-built item response theory models for ability estimation, item selection methods, exposure controls, and stopping rules. The issue, however, is that if a researcher is developing any *new* parameter update algorithms, item selection algorithms, exposure controls, or stopping rules, this requires a largely from-scratch implementation to conduct simulation studies that compare their methods to existing methods. As CAT research increasingly augments traditional psychometrics with increasingly computational approaches (Liu et al., 2024), the need for algorithmic comparison studies only increases. Through `meow`, users can easily implement new CAT algorithms that can be quickly integrated into existing simulation studies and shared with other researchers doing CAT development work.

Features

To avoid frequent reinvention of the wheel, `meow` modularizes CAT simulations by dividing the CAT into three distinct parts: (1) data generation, (2) parameter updates, and (3) item selection (including exposure control and a stopping rule). Each of these modules is supported by a central framework that provides a consistent API to facilitate simple and reproducible simulation studies. While `meow` contains a selection of built-in data generating processes (DGPs), parameter update methods, and item selection algorithms, the advance is a documented and flexible platform in which users can implement their own versions of any of these pieces and swap them in and out of simulation studies to provide directly comparable and reproducible results.

Simulations in `meow` are built around a single call to the function `meow()`. As arguments, this takes an item selection function, a parameter update function, and a data loader function. These three functions dictate how the simulation will be carried out, and `meow` comes bundled with implementations of a selection of common parameter update and item selection algorithms, including example implementations of recent published methods. Users also have the ability to supply initial values for internal parameters and set random seeds for each individual component of the simulation, allowing for customizability, reproducability, and comparability between runs. Each simulation outputs consistently formatted estimates and bias for each internal parameter at each iteration of the simulation, allowing users to easily parse and recycle visualization code using commonly available `R` tools.

The real value of `meow` is to the developer of new algorithms, however. Using a common API, users can implement their own methodological developments quickly and easily in `meow`, and then submit pull requests to have their work merged back into the package for other researchers to use in their own simulation studies.

Ongoing Reseach

Acknowledgements

We specifically thank Kylie Gorney for valuable feedback on the implementation and usability of this software.

References

Chalmers, R. P. (2016). Generating adaptive and non-adaptive test interfaces for multidimensional item response theory applications. *Journal of Statistical Software*, *71*, 1–38.

Liu, Q., Zhuang, Y., Bi, H., Huang, Z., Huang, W., Li, J., Yu, J., Liu, Z., Hu, Z., Hong, Y., & others. (2024). Survey of computerized adaptive testing: A machine learning perspective. *arXiv Preprint arXiv:2404.00712*.

Magis, D., & Barrada, J. R. (2017). Computerized adaptive testing with r: Recent updates of the package catR. *Journal of Statistical Software*, *76*, 1–19.