pref_voting: The Preferential Voting Tools package for Python

Wesley H. Holliday [1*] **and Eric Pacuit** [2*]

1 University of California, Berkeley **2** University of Maryland ***** These authors contributed equally.

DOI: 10.21105/joss.07020

Software

- Review
- Repository
- Archive

Editor: Britta Westner

Reviewers:

- @dmnapolitano
- @pkrafft

Submitted: 24 April 2024
Published: 21 January 2025

Summary

Preferential Voting Tools (`pref_voting`) is a Python package designed for research in voting theory (Brams & Fishburn, 2002; Dummett, 1984; Pacuit, 2019; Tideman, 2006; Zwicker, 2016), a subfield of social choice theory (Arrow, 1963; Fishburn, 1973; Kelly, 1988; Sen, 2017), and for practical applications of the theory. The basic problem of voting theory concerns how to combine "inputs" from multiple individual voters into a single social "output". For example, a common type of input from each voter is a *ranking* of some set of candidates, while a common type of social output is the selection of a *winning candidate* (or perhaps a set of candidates tied for winning). A *voting method* is then a function that takes in a ranking from each voter and outputs a winning candidate (or set of tied candidates). Other functions may instead output a social ranking of the candidates (Arrow, 1963) or a probability distribution over the candidates (Brandt, 2017), and other input types are also possible, such as sets of approved candidates (Brams & Fishburn, 2007), or assignments of grades to candidates (Balinski & Laraki, 2010), or real-valued functions on the set of candidates (d'Aspremont & Gevers, 2002; Sen, 2017). Faced with a function of any of these types, voting theorists study the function from several perspectives, including the general principles or "axioms" it satisfies (Felsenthal, 2012; Nurmi, 1987, 1999), its susceptibility to manipulation by strategic voters (Taylor, 2005), its statistical behavior according to probability models for generating voter inputs (Green-Armytage et al., 2016; Merrill, 1988), the complexity of the function and related computational problems (e.g., the problem of determining if the function can be manipulated in a given election) (Faliszewski et al., 2009), and more. These studies are greatly facilitated by the implementation of algorithms for computing the relevant functions and checking their properties, which are provided in `pref_voting`.

Statement of need

Research in the burgeoning field of *computational social choice* (COMSOC) (Aziz et al., 2019; Brandt et al., 2016; Geist & Peters, 2017) often applies computer-assisted techniques to the study of voting methods and other collective decision procedures. The aim of `pref_voting` is to contribute to a comprehensive set of tools for such research. Other packages in this area include `abcvoting` (Lackner et al., 2023), which focuses on approval-based committee voting, `preflibtools` (Mattei & Walsh, 2013), which contains tools for working with preference data from PrefLib.org, and `prefsampling` (Boehmer et al., 2024), which implements probability models for generating voter rankings. The `pref_voting` package provides functionality not available in these previous packages, while also interfacing with `preflibtools` and `prefsampling`. Like `pref_voting`, the `VoteKit` (MGGG Redistricting Lab, 2024) and `VoteLib` (Šimbera, 2021) packages provide implementations of a number of voting methods; and like `prefsampling`, `VoteKit` provides tools for generating elections. However, neither package includes all the voting methods and functionality in `pref_voting`, as described below. The `pref_voting` package has already been used in COMSOC research (Holliday et al., Forthcoming; Hornischer & Terzopoulou, 2024) and in online COMSOC tools (Peters, 2024). The package can also be used by election administrators to determine election outcomes, as it is used in the Stable Voting website.

Functionality

Elections

The `pref_voting` package includes classes for the most important representations of elections, or types of `edata`, used in voting theory:

- `Profile`: each voter linearly orders the candidates;
- `ProfileWithTies`: each voter ranks the candidates, allowing ties and omissions of candidates;
- `GradeProfile`: each voter assigns grades from some finite list of grades to selected candidates (with approval ballots as a special case);
- `UtilityProfile`: each voter assigns a real number to each candidate;
- `SpatialProfile`: each voter and each candidate is placed in a multi-dimensional space;
- `MajorityGraph`: an edge from candidate A to candidate B represents that more voters rank A above B than vice versa;
- `MarginGraph`: a weighted version of a `MajorityGraph`, where the weight on an edge represents the margin of victory (or other measure of strength of majority preference).

The package also includes methods for transforming one type of representation into another, e.g., turning a `SpatialProfile` into a `UtilityProfile` given a choice of how spatial positions of voters and candidates determine voter utility functions (Merrill & Grofman, 1999), or turning a `MarginGraph` into a minimal `Profile` that induces that `MarginGraph` by solving an associated linear program, and so on. Other methods are included for standard voting-theoretic tests and operations, e.g., testing for the existence of Condorcet winners/losers, removing candidates, and so on. Methods are also included to import from and export to the PrefLib preference data format, the ABIF format (Lanphier, 2024), and other data formats.

Generating elections

For sampling profiles according to standard probability models, `pref_voting` interfaces with the `prefsampling` package. In addition, `pref_voting` contains functions for sampling other types of `edata` listed above, as well as functions for enumerating such objects up to certain equivalence relations.

Collective decision procedures

Several classes of collective decision procedures are built into `pref_voting`:

- `VotingMethod`: given edata, outputs a list of candidates, representing tied winners;
- `ProbVotingMethod`: given edata, outputs a dictionary whose keys are candidates and whose values are probabilities;
- `SocialWelfareFunction`: given `edata`, outputs a ranking of the candidates.

Dozens of such functions are implemented in `pref_voting` and organized into standard groups identified in voting theory, e.g., positional scoring rules, iterative methods, margin-based methods (weighted tournament solutions), and cardinal methods.

Axioms

The `pref_voting` package also contains an `Axiom` class for functions that check whether a collective decision procedure satisfies a given axiom with respect to some edata. Each axiom comes with a `has_violation` method that checks whether there is at least one violation of the axiom by the procedure for the given `edata`, as well as a `find_all_violations` method that enumerates all such violations together with relevant data. Axioms are divided into several well-known groups from voting theory, e.g., dominance axioms, monotonicity axioms, variable voter axioms, and variable candidate axioms.

Analysis

Finally, `pref_voting` comes with functions that facilitate the analysis of collective decision procedures, such as producing data on the frequency of axiom violations in elections generated using one of the available probability models.

Acknowledgements

We thank Jobst Heitzig and Dominik Peters for helpful contributions, Zoi Terzopoulou for helpful feature requests, and all three for helpful feedback on this paper.

References

Arrow, K. J. (1963). *Social choice and individual values* (2nd ed.). John Wiley & Sons, Inc. https://doi.org/10.12987/9780300186987

Aziz, H., Brandt, F., Elkind, E., & Skowron, P. (2019). Computational social choice: The first ten years and beyond. In B. Steffen & G. Woeginger (Eds.), *Computing and software science* (Vol. 10000). Springer. https://doi.org/10.1007/978-3-319-91908-9_4

Balinski, M., & Laraki, R. (2010). *Majority judgement: Measuring, ranking and electing*. MIT Press. https://doi.org/10.7551/mitpress/9780262015134.003.0001

Boehmer, N., Faliszewski, P., Janeczko, Ł., Kaczmarczyk, A., Lisowski, G., Pierczyński, G., Rey, S., Stolicki, D., Szufa, S., & Wąs, T. (2024). Guide to numerical experiments on elections in computational social choice. In K. Larson (Ed.), *Proceedings of the thirty-third international joint conference on artificial intelligence, IJCAI-24* (pp. 7962–7970). International Joint Conferences on Artificial Intelligence Organization. https://doi.org/10.24963/ijcai.2024/881

Brams, S. J., & Fishburn, P. C. (2002). Voting procedures. In K. J. Arrow, A. K. Sen, & K. Suzumura (Eds.), *Handbook of social choice and welfare* (Vol. 1, pp. 173–236). North-Holland. https://doi.org/10.1016/S1574-0110(02)80008-X

Brams, S. J., & Fishburn, P. C. (2007). *Approval voting* (2nd ed.). Springer. https://doi.org/10.1007/978-0-387-49896-6

Brandt, F. (2017). Rolling the dice: Recent results in probabilistic social choice. In U. Endriss (Ed.), *Trends in computational social choice* (pp. 3–26). AI Access. https://archive.illc.uva.nl/COST-IC1205/BookDocs/Chapters/TrendsCOMSOC-01.pdf

Brandt, F., Conitzer, V., Endriss, U., Lang, J., & Procaccia, A. D. (Eds.). (2016). *Handbook of computational social choice*. Cambridge University Press. https://doi.org/10.1017/CBO9781107446984

d'Aspremont, C., & Gevers, L. (2002). Social welfare functionals and interpersonal comparability. In K. J. Arrow, A. K. Sen, & K. Suzumura (Eds.), *Handbook of social choice and welfare* (Vol. 1, pp. 459–541). North-Holland. https://doi.org/10.1016/S1574-0110(02)80014-5

Dummett, M. (1984). *Voting procedures*. Clarendon Press.

Faliszewski, P., Hemaspaandra, E., Hemaspaandra, L., & Rothe, J. (2009). A richer understanding of the complexity of election systems. In S. Ravi & S. Shukla (Eds.), *Fundamental* *problems in computing: Essays in honor of Professor Daniel J. Rosenkrantz*. Springer-Verlag. https://doi.org/10.1007/978-1-4020-9688-4_14

Felsenthal, D. S. (2012). Review of paradoxes afflicting procedures for electing a single candidate. *Electoral Systems: Paradoxes, Assumptions and Procedures*, 19–92. https://doi.org/10.1007/978-3-642-20441-8_3

Fishburn, P. C. (1973). *The theory of social choice*. Princeton University Press. https://doi.org/10.1515/9781400868339

Geist, C., & Peters, D. (2017). Computer-aided methods for social choice theory. In U. Endriss (Ed.), *Trends in computational social choice* (pp. 249–267). AI Access. https://archive.illc.uva.nl/COST-IC1205/BookDocs/Chapters/TrendsCOMSOC-13.pdf

Green-Armytage, J., Tideman, T. N., & Cosman, R. (2016). Statistical evaluation of voting rules. *Social Choice and Welfare*, *46*, 183–212. https://doi.org/10.1007/s00355-015-0909-0

Holliday, W. H., Kristoffersen, A., & Pacuit, E. (ForthcomingForthcoming). Learning to manipulate under limited information. *Proceedings of the 39th Annual AAAI Conference on Artificial Intelligence (AAAI-25)*. https://arxiv.org/abs/2401.16412

Hornischer, L., & Terzopoulou, Z. (2024). *Learning how to vote with principles: Axiomatic insights into the collective decisions of neural networks*. https://arxiv.org/abs/2410.16170

Kelly, J. S. (1988). *Social choice theory: An introduction*. Springer. https://doi.org/10.1007/978-3-662-09925-4

Lackner, M., Regner, P., & Krenn, B. (2023). abcvoting: A python package for approval-based multi-winner voting rules. *Journal of Open Source Software*, *8*(81), 4880. https://doi.org/10.21105/joss.04880

Lanphier, R. (2024). Abif. In *GitHub repository*. GitHub. https://github.com/electorama/abif

Mattei, N., & Walsh, T. (2013). PrefLib: A library of preference data. *Proceedings of Third International Conference on Algorithmic Decision Theory (ADT 2013)*, 259–270. https://doi.org/10.1007/978-3-642-41575-3_20

Merrill, S. (1988). *Making multicandidate elections more democratic*. Princeton University Press. https://doi.org/10.1515/9781400859504

Merrill, S., & Grofman, B. (1999). *A unified theory of voting: Directional and proximity spatial models*. Cambridge University Press. https://doi.org/10.1017/CBO9780511605864

MGGG Redistricting Lab. (2024). VoteKit. In *GitHub repository*. GitHub. https://github.com/mggg/VoteKit

Nurmi, H. (1987). *Comparing voting systems*. D. Reidel. https://doi.org/10.1007/978-94-009-3985-1

Nurmi, H. (1999). *Voting paradoxes and how to deal with them*. Springer. https://doi.org/10.1007/978-3-662-03782-9

Pacuit, E. (2019). Voting methods. In E. N. Zalta (Ed.), *The Stanford Encyclopedia of Philosophy* (Fall 2019). https://plato.stanford.edu/archives/fall2019/entries/voting-methods/; Metaphysics Research Lab, Stanford University.

Peters, D. (2024). *Pref.tools: Tools that are useful for analyzing preferences*. https://pref.tools

Sen, A. (2017). *Collective choice and social welfare: An expanded edition*. Harvard University Press. https://doi.org/10.4159/9780674974616

Šimbera, J. (2021). Votelib: Evaluation of voting systems in python. In *GitHub repository*. GitHub. https://github.com/simberaj/votelib

Taylor, A. D. (2005). *Social choice and the mathematics of manipulation*. Cambridge University Press. https://doi.org/10.1017/cbo9780511614316

Tideman, N. (2006). *Collective decisions and voting: The potential for public choice*. Routledge. https://doi.org/10.4324/9781315259963

Zwicker, W. S. (2016). Introduction to the theory of voting. In F. Brandt, V. Conitzer, U. Endriss, J. Lang, & A. D. Procaccia (Eds.), *Handbook of computational social choice* (pp. 23–56). Cambridge University Press. https://doi.org/10.1017/cbo9781107446984.003