**bwsample: Processing Best-Worst Scaling data**

**Ulf A. Hamster**

1 Berlin-Brandenburgische Akademie der Wissenschaften, Berlin, Germany

**Summary**

*bwsample* is a Python package that provides algorithms to sample best-worst scaling sets (BWS sets), extract and count pairwise comparisons from user-evaluated BWS sets, and compute rankings and scores.

**Statement of need**

We are using the *bwsample* package as part of an *Active Learning* experiment in which linguistics experts and lay people (crowdsourcing) judge sentences examples with the Best-Worst Scaling (BWS) method (Fig. 1). BWS is “… the cognitive process by which respondents repeatedly choose the two objects in varying sets of three or more objects that they feel exhibit the largest perceptual difference on an underlying continuum of interest” ([Finn & Louviere, 1992, p. 13](https://doi.org/10.21105/joss.03324)). In our context, BWS is primarily used as a means of data collection that is more economically efficient than using pairwise comparison user interfaces ([Hamster, 2021](https://doi.org/10.21105/joss.03324)).

---

**Figure 1:** Using *bwsample* (bws) in an Active Learning experiment.
Software Features

The sampling algorithms ensure overlapping BWS sets, and are deployed in the REST API for a Web App. Overlaps are required for counting pairs by logical inference (Hamster, 2021). A possible question is how many items have to be shown twice a) initially, and b) after the pair frequency database grew over time to gather reasonable amounts of counting or resp. frequency data? The implemented counting algorithms can distinguish between 3 types of directly extract pairs and 7 types of logically inferred pairs. This creates an opportunity for further analysis, e.g., to detect inconsistent evaluations (Hamster, 2021), or to assign weights to different types of pairs. To compute rank items from pairwise comparison data, five algorithms are available: a) Eigenvector estimation of the reciprocal pairwise comparison matrix as scores (Saaty, 2003), b) MLE estimation of the Bradley-Terry-Luce probability model (Hunter, 2004, pp. 386–387), c) Simple ratios for each pair and sum the ratios for each item, d) Chi-Squared based p-value for each pair and sum 1 minus p-values for each item, and e) Estimation of the transition probability that the next element is better. All ranking algorithms are implemented based on sparse matrix operations.

Acknowledgements

This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - 433249742.

Software Dependencies

bwsample is written in Python 3.6+ (Van Rossum & Drake, 2009) and uses the following software packages:

- NumPy, https://github.com/numpy/numpy (Harris et al., 2020)
- scikit-learn, https://github.com/scikit-learn/scikit-learn (Pedregosa et al., 2011)

References


Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau,


