The Journal of Open Source Software

Pymer4: Connecting R and Python for Linear Mixed Modeling

Eshin Jolly[1]

1 Dartmouth College

DOI: 10.21105/joss.00862

Software

- Review
- Repository
- Archive

Submitted: 28 July 2018
Published: 26 November 2018

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

Summary

In the social sciences the prevalence of clustered, correlated, and/or repeatedly measured data is very common. Modeling such data have classically involved the use of techniques such as the repeated measured analysis of variance (ANOVA) with correction factors applied to account for violations of model assumptions. More recently multi-level or hierarchical linear models have been brought to bear on modeling these kind of data given their flexibility. One of the most popular packages used for this purpose is the powerful `lme4` package in the R statistical programming language (Bates, Mächler, Bolker, & Walker, 2015). While this package is highly favored by individuals that utilize R, to date there exist few comparable packages in the scientific Python community. While some solutions do exist they are often more complicated to use and sometimes less flexible (e.g. `statsmodels` (Seabold & Perktold, 2010) or require users to adopt a different statistical framework (e.g. Bayesian modeling, `bambi` (Yarkoni & Westfall, 2016)). This leaves Python users in want for a tool that: a) is highly compatible with existing scientific python tools (e.g. pandas, `numpy`, `matplotlib`, `seaborn` (McKinney, 2012)), b) has an API that is easy to use but not unfamiliar to those who use to `lme4` , c) offers additional functionality that anticipates users' needs when analyzing real data (e.g. significance testing, simulating data, post-hoc analyses, etc).

For these reasons `pymer4` was developed to offer users a way to work with multilevel models within the scientific Python ecosystem. To accomplish this `pymer4` leverages the existing prowess of `lme4`, through the use of the `rpy2` (Gautier, 2008) cross-langauge compatibility library. This offers analysts a tool at the level of abstraction they expect, without needing to worry about writing code specific to interfacing between languages (i.e. custom `rpy2` scripts, functions, etc). Beyond making `lme4` functionality available in python, `pymer4` makes working with multilevel models feel native to working in scientific Python and offers several key features users are most likely to utilize. This includes tight integration with `pandas` and `numpy` for data representation and manipulation, as well as `matplotlib` (Hunter, 2007) and `seaborn` (Waskom et al., 2014) for data visualization. A key contribution of `pymer4` is its role in providing a unified interface to tasks that are spread over several *different* R-packages. For example, p-value computation lacking in `lme4` is instead provided using the `lmerTest` (Kuznetsova, Brockhoff, & Christensen, 2017) library, while post-hoc subgroup comparisons are provided using the `lsmeans` (Lenth & others, 2016) package. This obviates the need for users to move between libraries, remember idiosyncratic commands, and allows them to instead focus on the analysis at hand.

Additionally, `pymer4` offers several extra features implemented at a high-level of abstraction that would otherwise require users to write custom functions and scripts. These include: simulating various types of data, visualizing model estimates, changing statistical inference procedures (e.g. parametric to non-parametric), automatically storing model fits and residuals, and estimating ordinary-least-squares regression models with various types of robust standard error estimators (Huber & others, 1967). Overall, the key contribution of `pymer4` is the development of a consistent and intuitive API by which users can accomplish analytic tasks in a simple and style familiar to users of the scientific Python. We hope this reduces the development "switch-cost" many users experience when moving between tools to accomplish a single goal.

Examples

Documentation for `pymer4` includes numerous example analyses that demonstrate various functionality. Below we demonstrate how easily `pymer4` integrates into a scientific Python analysis workflow:

```
# imports
import os
import pandas as pd
import seaborn as sns
from pymer4.models import Lmer
from pymer4.utils import get_resource_path

# Load included example data with pandas
df = pd.read_csv(os.path.join(get_resource_path(),'sample_data.csv'))

# Fit a multi-level model with 1 categorical predictor using dummy-codes with '1.0
model = Lmer('DV ~ IV3 + (IV3|Group)',data=df)
model.fit(factors={'IV3':['1.0','0.5','1.5']})

# Perform post-hoc comparisons on the fitted model
model.post_hoc(marginal_vars='IV3')

# Plot model coefficients
model.plot_summary()

# Visualize residuals with seaborn; residuals are conveniently stored in the model
sns.regplot(x= 'IV2',
            y= 'residuals',
            data= model.data,
            fit_reg= False
            )
```

Acknowledgements

The author thanks Luke J. Chang, Jin Hyun Cheong, and Pin-hao Chen for testing and feedback. The author also thanks Jeroen Van Baar for assistance in testing installations on different platforms.

References

Bates, D., Mächler, M., Bolker, B., & Walker, S. (2015). Fitting linear mixed-effects models using lme4. *Journal of Statistical Software*, *67*(1), 1–48. doi:10.18637/jss.v067.i01

Gautier, L. (2008). Rpy2: A simple and efficient access to r from python. *URL http://rpy. sourceforge. net/rpy2. html*.

Huber, P. J., & others. (1967). The behavior of maximum likelihood estimates under nonstandard conditions. In *Proceedings of the fifth berkeley symposium on mathematical statistics and probability* (Vol. 1, pp. 221–233). University of California Press. doi:https://projecteuclid.org/euclid.bsmsp/1200512988

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in science & engineering*, *9*(3), 90–95. doi:10.1109/MCSE.2007.55

Kuznetsova, A., Brockhoff, P. B., & Christensen, R. H. B. (2017). LmerTest package: Tests in linear mixed effects models. *Journal of Statistical Software*, *82*(13). doi:http://dx.doi.org/10.18637/jss.v082.i13

Lenth, R. V., & others. (2016). Least-squares means: The r package lsmeans. *Journal of statistical software*, *69*(1), 1–33. doi:http://dx.doi.org/10.18637/jss.v069.i01

McKinney, W. (2012). *Python for data analysis: Data wrangling with pandas, numpy, and ipython.* " O'Reilly Media, Inc.".

Seabold, S., & Perktold, J. (2010). Statsmodels: Econometric and statistical modeling with python. In *Proceedings of the 9th python in science conference* (Vol. 57, p. 61). SciPy society Austin. doi:https://doi.org/10.5281/zenodo.275519

Waskom, M., Botvinnik, O., Hobson, P., Warmenhoven, J., Cole, J., Halchenko, Y., Vanderplas, J., et al. (2014). Seaborn: Statistical data visualization. *URL: https://seaborn. pydata. org/(visited on 2017-05-15)*. doi:https://doi.org/10.5281/zenodo.1313201

Yarkoni, T., & Westfall, J. (2016). Bambi: A simple interface for fitting bayesian mixed effects models. doi:https://dx.doi.org/10.31219/osf.io/rv7sn