Bambi: A simple interface for fitting Bayesian mixed effects models

Tal Yarkoni and Jacob Westfall

University of Texas at Austin

Tal Yarkoni is a research assistant professor at the University of Texas at Austin. His research focuses on developing new methods for the large-scale acquisition, management, and synthesis of psychology and neuroimaging data.

Jacob Westfall is a postdoctoral researcher at the University of Texas at Austin. His research is in the areas of social and quantitative psychology, focusing in particular on mixed-effects models, power analysis and optimal experimental design, and psychometrics.

Bambi: A simple interface for fitting Bayesian mixed effects models

Abstract

The popularity of Bayesian statistical methods has increased dramatically among psychologists in recent years; however, many common types of mixed effects models remain difficult to fit in a Bayesian setting. Here we introduce an open-source Python package named *Bambi* (BAyesian Model Building Interface) that is built on top of the powerful PyMC3 probabilistic programming framework and makes it easy to specify complex generalized linear mixed effects models using a formula notation similar to those found in packages like lme4 and nlme. We demonstrate Bambi's versatility and ease-of-use by applying it to three social and personality psychology datasets that span a range of common statistical models--including multiple regression, logistic regression, and mixed-effects modeling with crossed random effects. We conclude with a discussion of the strengths and weaknesses of a Bayesian approach in the context of common statistical practice in psychology.

Keywords (from the SPPS list):
Hierarchical Linear Modeling/Multilevel Modeling, Quantitative Models, Advanced quantitative methods

Introduction

Psychologists are increasingly relying on Bayesian statistical methods. Van de Schoot et al. (2016) recently documented that between 1990 and 2015, the relative proportion of papers in psychology journals using or discussing Bayesian methods increased roughly fourfold. The most impressive growth was actually not in technical or theoretical papers on Bayesian methodology, but rather in empirical papers applying Bayesian regression models to real datasets, which papers have experienced explosive growth since around 2010. In other words, Bayesian methods are penetrating the mainstream of psychological statistics and data analysis.

The recent rise of Bayesian methods within psychology is occurring in the face of two major barriers that have historically limited the widespread adoption of Bayesian statistics. First, many Bayesian models are computationally difficult to implement. Historically, fitting Bayesian models required either considerable mathematical work (to derive analytical solutions to difficult statistical problems) or enormous computing resources (to obtain approximate solutions by probabilistically sampling from the full parameter space). Although many large Bayesian models remain computationally intractable even given modern computing resources, thanks to recent hardware and software advances, most models of the kind psychologists routinely work with can now be efficiently fit within a Bayesian paradigm on an average modern laptop using entirely free software. Thus, the computational barrier to widespread adoption of Bayesian statistics is rapidly disappearing.

A second challenge, however, persists: Bayesian model-fitting packages remain much less accessible than their frequentist counterparts. Whereas a number of maximum likelihood-based packages allow compact, easy specification of complex mixed models (e.g., the use of formula notation in R packages like lme4 and nlme), we know of no Bayesian packages that support rapid specification of mixed models of arbitrary complexity (though excellent tools exist for common use cases such as t-tests and ANOVAs; cf Dropmann, Verhagen, & Wagenmakers, 2015; R. D. Morey, Rouder, & Jamil, 2014). The development of a simpler, more intuitive interface would, in our view, go a long way towards promoting the widespread use of Bayesian methods within psychology and other sciences.

Here we introduce a new open-source Python package designed to make it considerably easier for researchers to fit mixed models using a Bayesian approach. Bambi (BAyesian Model Building Interface) can be used for fitting generalized linear mixed models (Bolker et al., 2009; Stroup, 2012), a very general class of models that includes most of the models commonly used in psychology: from linear regression and ANOVA, to logistic and poisson regression, to multilevel/hierarchical or crossed random effects models, to even more specialized models such as those found in signal detection theory (Egan, 1975; Swets, 2014) and the social relations model (Kenny & La Voie, 1984). Bambi is built on the PyMC3 Python package (Patil, Huard, & Fonnesbeck, 2010; Salvatier, Wiecki, & Fonnesbeck, 2016), which in turn is built on the Theano deep learning package (Bastien et al., 2012; Bergstra et al., 2010), and implements the state-of-the-art No U-Turn MCMC sampler (Hoffman & Gelman, 2014). Importantly, Bambi affords both ease-of-use and considerable power: beginning users can specify GLMMs in much

the same way as such models are specified in packages like lme4, whereas advanced users can directly access all of the internal objects exposed by PyMC3 and Theano, allowing virtually limitless flexibility.

In the next section, we provide a high-level technical overview of the Bambi package and the model-fitting functionality it supports. We then illustrate its use via a series of increasingly complex applications to real datasets from social and personality psychology, beginning with a straightforward multiple regression model, and culminating with a linear mixed model that involves custom priors. Importantly, all analyses are supported by extensive documentation in the form of interactive notebooks available at the project repository on GitHub (http://github.com/bambinos/bambi), enabling readers to re-run, modify, and otherwise experiment with the models described here (and many others) on their own machines. We conclude with a brief discussion of what the dramatic growth and accessibility of Bayesian statistical methods might mean for psychology in the coming years.

Package overview

Bambi is written in the Python programming language. Python is a general-purpose, dynamic, high-level programming language with wide adoption in scientific computing and data science. The availability of excellent numerical computing libraries in Python enables us to focus our development efforts on building a high-level, user-friendly interface to Bayesian models. Bambi deliberately maintains a clear separation of concerns between model specification and model fitting. When users create a new model with Bambi, an abstract representation of the model is generated that is completely independent of the specific numerical package(s) ultimately used to compile and/or fit the model. In principle, our hope is that this architecture will eventually allow Bambi users to switch seamlessly between different computational back-ends (e.g., PyMC3, Stan, BUGS, etc.) without having to modify any of their models or code.

In practice, Bambi currently relies exclusively on PyMC3 as its back-end. PyMC3 is a probabilistic programming framework written in Python that supports many state-of-the-art Bayesian methods (Salvatier et al., 2016). PyMC3 in turn relies on the Theano deep learning library (Bergstra et al., 2010), which provides highly optimized numerical routines that run on both the CPU and the GPU. One benefit of relying on this existing software stack is that models created using Bambi automatically benefit from an enormous amount of optimization and a very active developer community. As new functionality is added to packages like PyMC3 and Theano (e.g., "blackbox" variational inference in PyMC3; (Kucukelbir, Tran, Ranganath, Gelman, & Blei, 2016; Ranganath, Gerrish, & Blei, 2014)), Bambi can easily leverage this functionality, with virtually no extra development effort or user code required. Another benefit is that advanced users can seamlessly extend their models beyond what the Bambi interface supports by directly accessing and modifying the PyMC3 models that Bambi maintains internally.

The source code for Bambi is released under the permissive MIT open-source license, and is hosted and maintained on GitHub (http://github.com/bambinos/bambi). Development of Bambi

closely follows standard best practices in open source software development, including fully automated continuous integration testing, ubiquitous version control, and comprehensive documentation.

Installation

Installing Bambi requires a Python interpreter (version 2.7+ or 3.1+). The easiest way to obtain a working Python environment is by installing the free Anaconda Python Distribution (https://www.continuum.io/downloads). Assuming Python is installed on a user's system, Bambi (and all of its dependencies) can be installed from the command line in a single line:

```
pip install bambi
```

Power users and developers are encouraged to use alternative installation procedures, which are described in detail in the README found on the Bambi GitHub repository. The README also provides more detail regarding installation of optional dependencies and covers troubleshooting of common problems.

Usage

To illustrate the use of Bambi, we re-analyze a number of existing datasets drawn from contexts that will be familiar to many social and personality psychologists. We begin with a simple multiple regression example and gradually scale up to a relatively complex generalized linear mixed-effect model. For the sake of brevity, we provide only partial code listings here; fullly executable versions of all examples are available from the Bambi GitHub repository (http://github.com/bambinos/bambi) as interactive Jupyter notebooks, enabling users to easily reproduce, modify, and extend all analyses we describe. The notebooks also contain much more extensive documentation and explanation.

Multiple regression

We begin with an example from personality psychology. The data that we consider come from the Eugene-Springfield community sample (Goldberg, 1999), a longitudinal study of hundreds of adults who completed dozens of different self-report and behavioral measures over the course of 15 years. Among the behavioral measures is an index of illegal drug use (the "drugs" outcome from the Behavioral Report Inventory; for details, see Grucza & Goldberg, 2007). We wish to know: which personality traits are associated with higher or lower levels of drug use? In particular, how do participants' standings on the "Big Five" personality dimensions (Openness to experience, Conscientiousness, Extraversion, Agreeableness, Neuroticism), as measured by the NEO-PI-R, predict drug use?

We assume the dataset is loaded in an array-like object named `data` (note that the `#` symbol at the beginning of a line indicates a comment, not syntax to be interpreted). Then it is simple to specify a multiple regression model using a familiar formula-like interface:

```
# Load the bambi library
from bambi import Model

# Set up the model
model = Model(data)
model.add_formula('drugs ~ o + c + e + a + n')
```

This fully specifies the model. We could now fit the model as-is, although we have not specified prior distributions for any of the parameters. If no priors are given explicitly by the user, then bambi chooses smart default priors for all parameters of the model based on plausible implied partial correlations between the outcome and the predictors (technical details are given in an Appendix). We can inspect the priors using the command:

```
model.plot(kind='priors')
```

This command results in Figure 1. Notice that the standard deviations of the priors for the slopes are quite small, with most of the probability mass being between -.04 and .04. This is due to the relative scales of the outcome and the predictors: the outcome, drugs, is a mean score that ranges from 1 to about 4, while the predictors are all sum scores that range from about 20 to 180. So a one-unit change in any of the predictors -- which is a trivial increase on the scale of the predictors -- is likely to lead to a very small absolute change in the outcome. These priors are actually quite wide (or “weakly informative”) on the partial correlation scale.

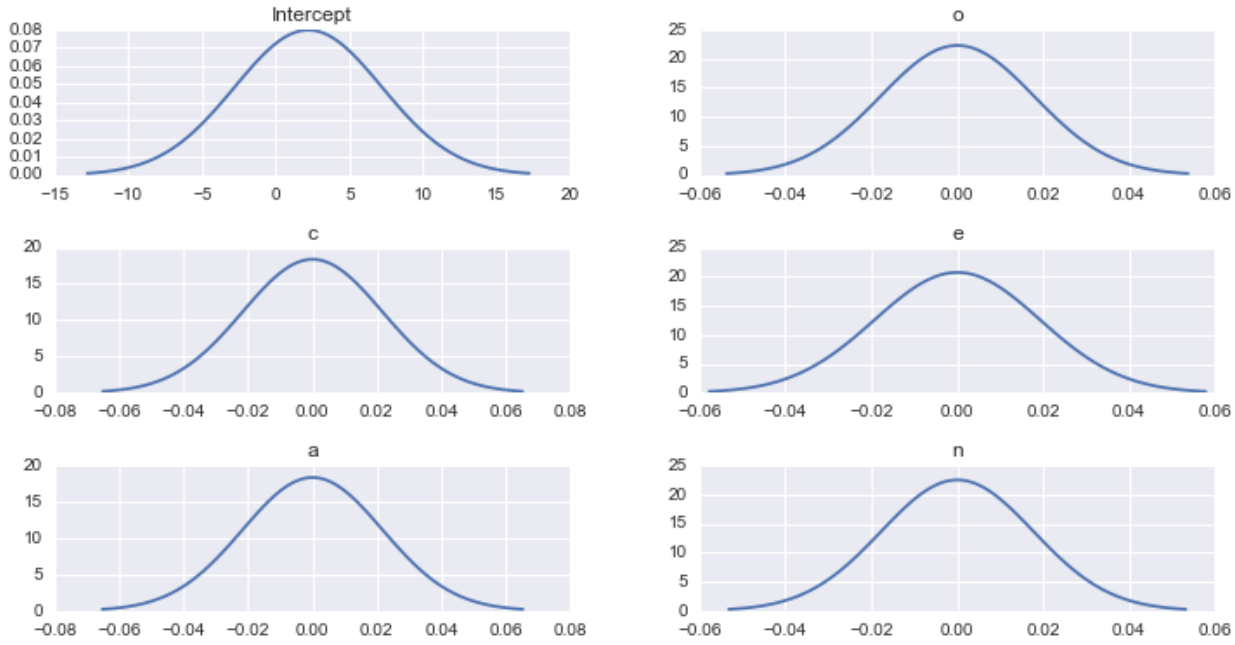

Figure 1. Default prior distributions for all of the regression coefficients. If the user does not explicitly state the priors to be used for the model parameters, bambi will choose smart default priors based on the values

of the regression coefficients that would imply plausible partial correlations between the outcome and the predictors (see Appendix).

To begin the actual model fitting process, we can now write:

```
fitted = model.fit(samples=2000, njobs=2)
```

In non-Bayesian models, which are typically estimated in a maximum likelihood setting, one obtains parameter estimates using either an exact mathematical expression or by running an optimization algorithm that searches through the parameter space for the “best” parameter estimates, iterating until reaching some well-defined convergence criteria. In Bayesian models, one obtains the parameter estimates by drawing samples of plausible parameter values from the joint posterior distribution of all the parameters until one has obtained a sufficiently large and well-behaved sample of plausible parameter values (Andrieu, de Freitas, Doucet, & Jordan, 2003). The code above says to run two parallel Markov Chain Monte Carlo (MCMC) chains, each one drawing 2000 samples from the joint posterior distribution of all the parameters.

We can now display the results using the command:

```
fitted.plot(burn_in=50)
```

Setting the `burn_in` parameter to 50 instructs bambi to discard the first 50 samples of both chains. This is standard practice in MCMC, as the samples tend to be unstable in this early “burn-in” period (Raftery & Lewis, 1996). This command results in Figure 2. The left panels of Figure 2 show the marginal posterior distributions for all of the model’s parameters, which summarize the most plausible values of the regression coefficients, given the data we have now observed. The lines and annotations indicate the means of the posteriors, which are reasonable point estimates of the parameters, although there is not necessarily any need to reduce these distributions to single point estimates. These posterior density plots show two overlaid distributions because we ran two MCMC chains. The right panels of Figure 2 show the sampling paths of the two MCMC chains as they wander through the parameter space. These “trace plots” are useful for diagnostic purposes (Raftery & Lewis, 1996).

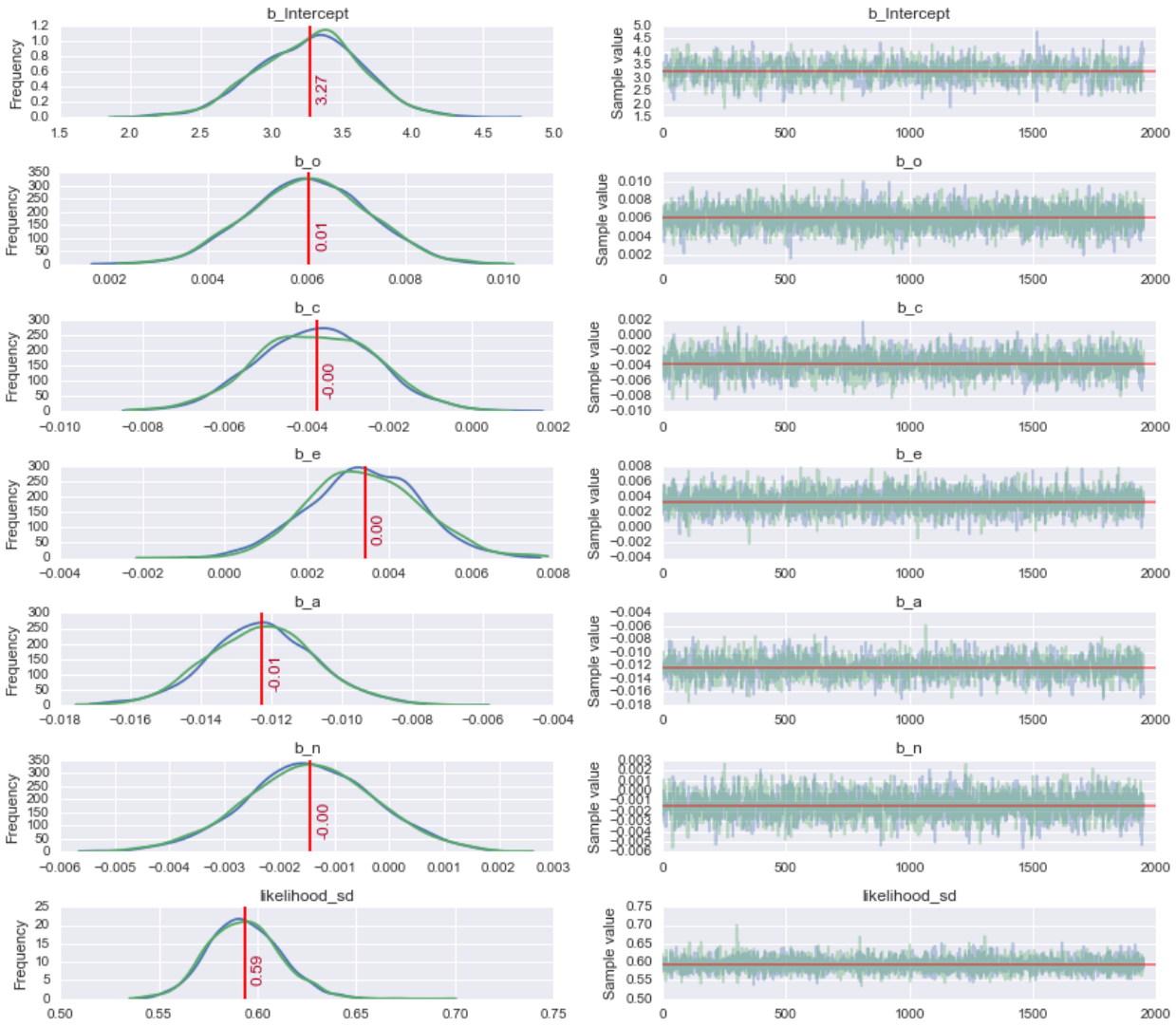

Figure 2. Smoothed histograms (left) and sample traces (right) of parameter estimates for all model terms. The parameters representing regression coefficients have a 'b_' prepended to the front of the variable name. The parameter labeled 'likelihood_sd' is the residual standard error, usually denoted σ.

The model results suggest that 4 of the 5 personality dimensions (all but Neuroticism) have at least some non-trivial association with drug use. We may further be interested in asking: which of these personality dimensions *matter more* for the prediction of drug use? There are many possible ways to think about what it means for one predictor to be relatively "more important" than another predictor (Hunsley & Meyer, 2003; Westfall & Yarkoni, 2016), but one conceptually straightforward way to approach the issue is to compare the partial correlations of each predictor with the outcome, controlling for the set of all other predictors. These comparisons are somewhat challenging using traditional frequentist methods, perhaps requiring a bootstrapping approach, but they can be formulated very naturally in the Bayesian framework. We can simply apply the relevant transformation (transforming the regression coefficients into partial correlation coefficients; see the Appendix) to all the posterior samples to obtain the joint posterior

distribution on the partial correlation or partial η^2 scales. This is what we have done in Figure 3. These posteriors allow us to draw intuitively compelling conclusions such as: according to the model, the probability that Openness to experience is a stronger predictor than Extraversion is about 94%. The probability that Agreeableness is the strongest of the five predictors is about 99%.

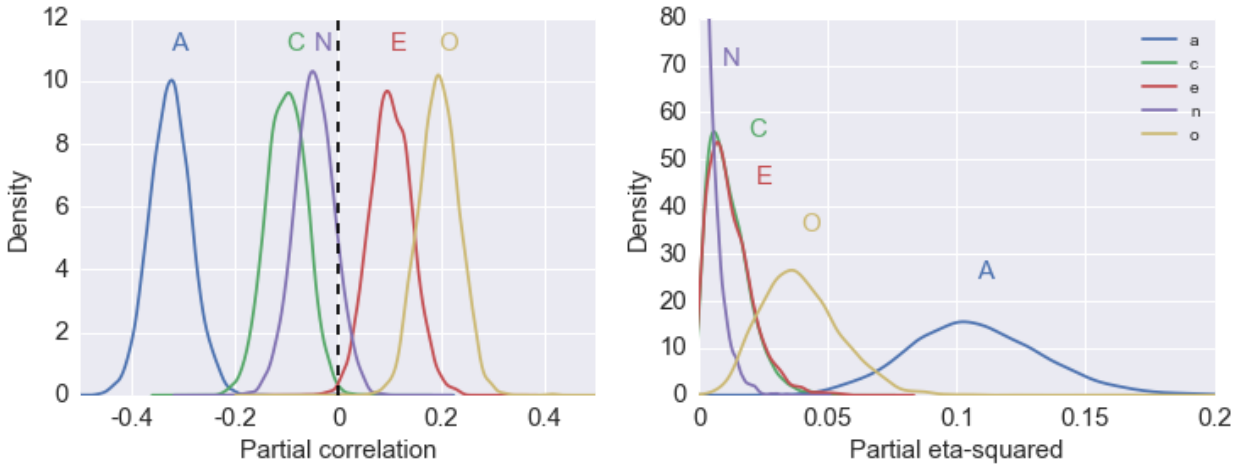

Figure 3. Posterior distributions of the relationships between the Big Five predictors and drug use on the partial correlation (left) and partial η^2 (right) scales.

Logistic regression

Our next example involves an analysis of the American National Election Studies (ANES) data. The ANES is a nationally representative, cross-sectional survey used extensively in political science. We obtained the most recent dataset, the 2016 pilot study, consisting of responses from 1200 voting-age U.S. citizens, from http://electionstudies.org. From this full dataset we extracted the subset of 421 respondents who had observed values on the following variables[1]:

- `vote`: "If the 2016 presidential election were between Hillary Clinton for the Democrats and Donald Trump for the Republicans, would you vote for Hillary Clinton, Donald Trump, someone else, or probably not vote?"
- `party_id`: "Generally speaking, do you usually think of yourself as a Republican, a Democrat, an independent, or what?"
- `age`: Computed from the respondent's birth year.

Clearly, respondents who self-identify as Democrats will be more likely to say they would vote for Clinton over Trump, respondents who self-identify as Republicans will tend to vote for Trump over Clinton, and Independent respondents will likely be somewhere between these two. What

[1] The high rate of missingness is due to the fact that different subsets of respondents were randomly selected by the ANES survey designers to view questions about different pairs of presidential primary candidates. The missing respondents are thus missing completely at random (MCAR).

we are interested in is the relationship between respondent age and intentions to vote for Clinton vs. Trump, and in particular how age may interact with party identification in predicting voting intentions. For brevity of presentation, we focus on voting intentions toward Clinton vs. Trump, not toward third-party candidates.

As before, we assume that the dataset is loaded in an array-like object named `data`. Then we can specify and fit the logistic regression model using the following commands:

```
clinton_model = Model(data)
clinton_fitted = clinton_model.fit('vote ~ party_id + party_id:age',
      family='binomial', link='logit', samples=2000, njobs=2)
```

We name the model `clinton_model` because the outcome variable, `vote`, is coded such that values of 1 indicate voting for Clinton while values of 0 indicate voting for Trump. Notice that in this case we specify and fit the model in a single line of code. As before, we instruct bambi to run two parallel MCMC chains, each drawing 2000 samples from the joint posterior. The only additional things we had to do to fit this model, compared to the multiple regression case, were to specify `family='binomial'` and `link='logit'`.

The key results from the model are given in Figure 4. The left panel shows the marginal posterior distributions of the slopes for the Age predictor in each party affiliation group. These posteriors show that, among Democrats, there is not much association between age and voting intentions. However, among both Republicans and Independents, there is a distinct tendency for older respondents to be less likely to indicate intent to vote for Clinton. The probabilities that the Age slopes for the Republicans and Independents are lower than the Age slopes for Democrats are both about 99%, indicating an age-by-party identification interaction. The right panel of Figure 4 is interesting; it is a “spaghetti plot” showing plausible logistic regression curves for each party identification category, given the data we have now observed. These are obtained by taking each individual posterior sample of parameters drawn by the MCMC chains and plotting the logistic regression curve implied by those sampled parameters. The spaghetti plot clearly shows the model predictions as well as the uncertainty around those predictions.

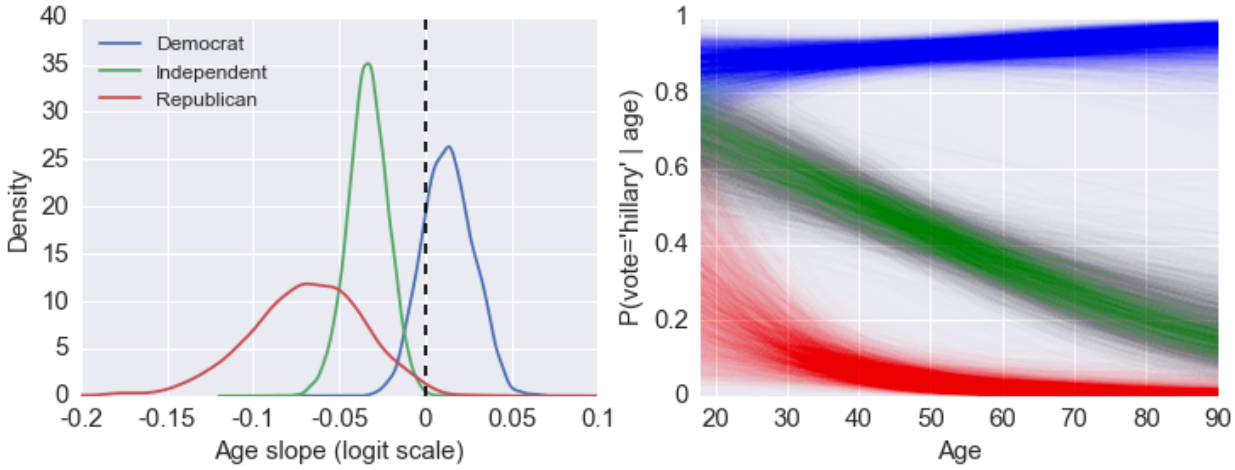

Figure 4. Left: Posterior distributions of the Age slopes for each party identification category, on the logit scale. Right: Spaghetti plot showing the model predictions and associated uncertainty.

Linear mixed models (LMMs)

Bambi makes it easy to fit mixed effects models with both fixed and random effects. To illustrate, we conduct a Bayesian reanalysis of the data reported in a recent registered replication report (RRR; van Allen Z. M. Vandekerckhove M. Wainwright B. Wayand J. F. Zeelenberg R. Zetzer E. E. Zwaan R. A., 2016) of a seminal study by Strack, Martin, & Stepper (1988). The original Strack et al. study tested the "facial feedback hypothesis", which holds that emotional responses are, in part, driven by facial expressions. Strack and colleagues reported in their Study 1 that participants rated cartoons as funnier when holding a pen between their teeth (unknowingly inducing a smile) than when holding a pen between their lips (unknowingly inducing a pout). The article has been cited over 1,400 times, and has been influential in popularizing the view that emotional experience not only shapes, but can also be shaped by, emotional expression. Yet in a 2016 Registered Replication Report led by Wagenmakers and colleagues at 17 independent sites, spanning over 2,500 participants, no evidence in support of the original effect could be found. We reanalyze and extend the Wagenmakers et al. analysis using a Bayesian LMM.

Here we fit a model conceptually similar to the one reported in Wagenmakers et al. (2016). Whereas Wagenmakers and colleagues fit a random-effects meta-analysis to the point estimates obtained from the 17 studies, we instead fit a multilevel linear mixed model containing the following terms: (1) the fixed effect of experimental condition ("smile" vs. "pout"), which is is the primary variable of interest; (2) random intercepts for the 17 studies; (3) random condition slopes for the 17 different study sites; (4) random intercepts for all subjects; (5) random intercepts for the 4 stimuli used at all sites[2]; and (6) fixed covariates for age and gender (since they are included in the dataset, and could conceivably account for variance in the outcome).

[2] We note that the “maximal model” for this dataset, in the sense of (Barr, Levy, Scheepers, & Tily, 2013), would also include random condition slopes across stimuli, which we have opted not to include in this purely didactic example.

Our model departs from the meta-analytic approach used by Wagenmakers et al. in that the latter allows for study-specific subject and error variances (though in practice, such differences are unlikely to impact the fixed estimate of the experimental condition effect). On the other hand, our approach properly accounts for the fact that the same stimuli were presented in all 17 studies. By explicitly modeling stimulus as a random factor, we ensure that our inferences can be generalized over a population of stimuli *like* the ones Wagenmakers et al. used, rather than applying only to the exact 4 Far Side cartoons that were selected (Judd, Westfall, & Kenny, 2012; Westfall, Judd, & Kenny, 2015).

We create a Bambi model, fit it, and store the results in a new object, much like we've already seen:

```
      # Initialize and specify the model without fitting it right
away
      model = Model(data)
      model.add_formula('value ~ 0 + condition + age + gender',
      random=[‘condition|study’, '1|uid', ‘1|stimulus’])

      # Set a custom prior on random factor variances--just for
      illustration
      random_sd = Prior('Uniform', lower=0, upper=10)
      random_prior = Prior('Normal', mu=0, sd=random_sd)
      model.set_priors(random=random_prior)

      # Fit the model, drawing 5,000 MCMC samples
      results = model.fit(samples=5000)
```

Notice that we've elected to set a custom prior here; specifically, we indicate that we'd like the variances of all random effects to be modeled using a uniform prior in the range of 0 - 10. This non-default prior actually has no discernible impact in this particular case (because the dataset is relatively large); we explicitly set it here only to illustrate the flexibility that Bambi provides. As the package documentation explains, one can easily specify a completely different prior for each model term, and any one of the many pre-existing distributions implemented in PyMC3 can be assigned.

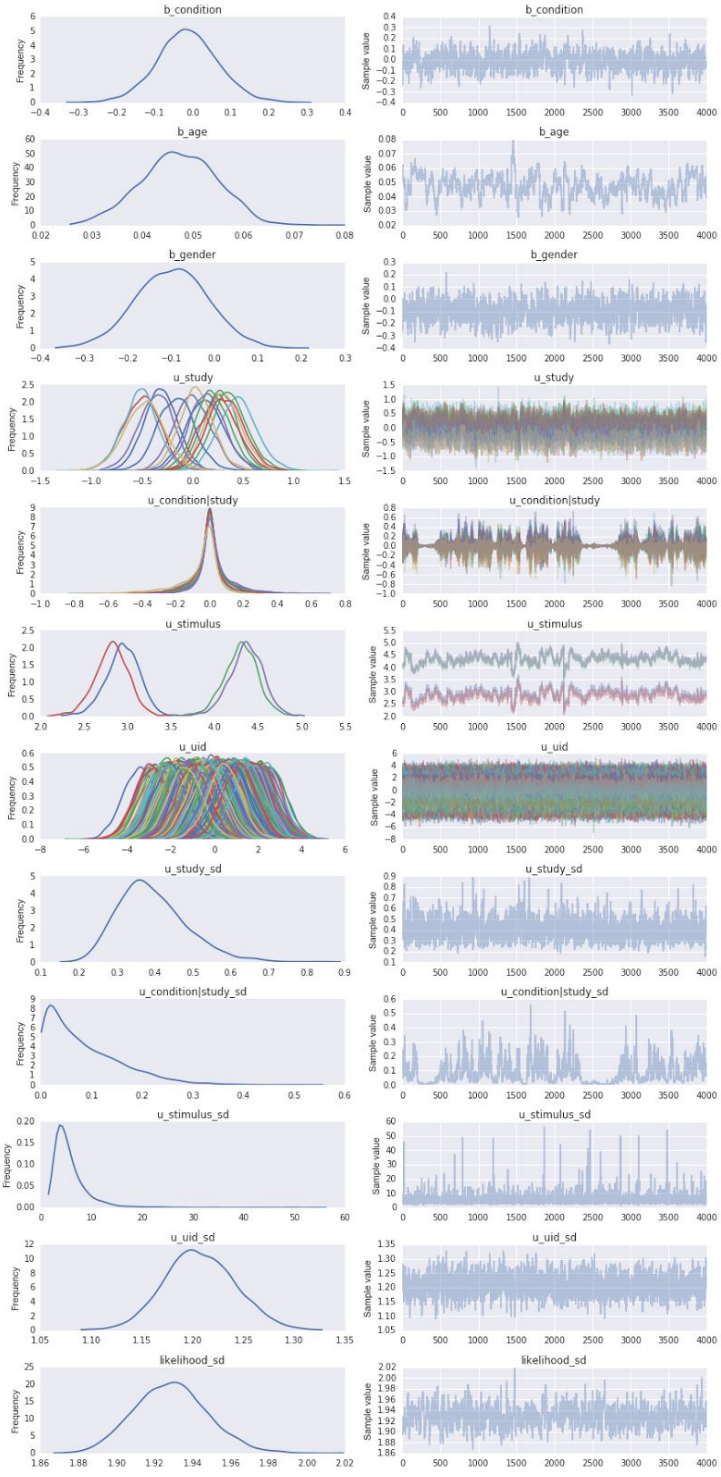

Figure 5. Smoothed histograms (left) and sample traces (right) of parameter estimates for all model terms. Fixed effects and random effects are denoted by the prefixes 'b_' and 'u_', respectively; terms with the suffix '_sd' are standard deviations of random effect terms (e.g., u_uid_sd is the SD of the means of the individual subject intercepts). The parameter labeled 'likelihood_sd' is the residual standard error, usually denoted σ.

Inspection of the results (Fig. 5) reveals essentially no effect of the experimental manipulation, consistent with the findings reported in Wagenmakers et al. (2016). Moreover, as reported in Wagenmakers et al., the variation across sites is surprisingly small in terms of both the random intercepts (*u_study_id*) and the random slopes *(u_condition|study_id*). One implication of this is that the constitution of the sample, the language of instruction, or any of the hundreds of other potential between-site differences, appear to make much less of a difference to participants' comic ratings than one might have intuitively supposed.

Interestingly, our LMM also highlights an additional point of interest not discernible from the results reported by Wagenmakers and colleagues: the stimulus-level variance (*u_stimulus_sd*) is very large compared to the other factors. This is problematic, because it suggests that any effects one identifies using a conventional analysis that fails to model stimulus effects could potentially be driven by idiosyncratic differences in the selected comics. Note that this is a problem that affects both the RRR and the original Strack study equally. In other words, regardless of whether the RRR would have obtained a statistically significant replication of the Strack study given different stimuli, if the effect is strongly driven by idiosyncratic properties of the specific stimuli used in the experiment--which is not unlikely, given that the results are based on just four stimuli drawn from a stimulus population that is likely quite heterogeneous--then there would have been little basis for drawing strong conclusions from that result in the first place. Either way, the moral of the story is that any factor that can be viewed as a sample from some population that one intends to generalize one's conclusions over (such as the population of funny comics) should be explicitly included as a random factor in one's model (Judd et al., 2012; Westfall et al., 2015). Bambi makes it very easy to fit such models within a Bayesian framework.

Discussion

We have introduced a high-level model-building interface that combines the ease of use of traditional mixed model packages like lme4 with the flexibility and power of the state-of-the-art PyMC3 probabilistic programming framework. The sample applications presented here illustrate how Bambi makes it possible to fit sophisticated Bayesian generalized linear mixed models (GLMMs) with little programming knowledge, using a formula-based syntax that will already be familiar to many users.

The Bayesian approach to fitting GLMMs is attractive for several reasons. Practically speaking, the ability to inject outside information into the analysis in a principled way, in the form of the prior distributions of the parameters, can have a beneficial regularizing effect on parameter estimates that are computationally difficult to pin down. For example, the variances of random effect terms can often be quite hard to estimate precisely, especially for small datasets and unbalanced designs. In the traditional maximum likelihood setting, these difficulties often manifest in point estimates of the variances and covariances that defy common sense, or in outright failure of the model to successfully converge on a satisfactory solution. Setting an informative Bayesian prior on these variance estimates can help bring the model estimates back

down to earth, resulting in a convergent fitted model that is much more believable (Chung, Gelman, Rabe-Hesketh, Liu, & Dorie, 2015). As a second example, it is well known that categorical outcome models such as logistic regression can suffer problems due to quasi-complete separation of the outcome with respect to the predictors included in the model, which has the effect of driving the parameter estimates toward unrealistic values approaching ∞ or $-\infty$. These problems are further exacerbated in mixed-effects logistic regression, where separation in some of the individual clusters can easily distort the overall fixed parameter estimates, even though there is no separation at the level of the whole dataset. Again, informative priors can help to rein in these diverging parameter estimates (Gelman, Andrew, Aleks, Pittau, & Yu-Sung, 2008).

For users with more programming experience, the probabilistic programming paradigm that PyMC3 supports may also confer additional benefits. A key benefit of this approach is that researchers can theoretically fit any model they can write down; no analytical derivation of a solution is required, as the highly optimized Theano numerical library efficiently computes gradients via automatic differentiation. Because Bambi is simply a high-level interface to PyMC3 models, in practice, researchers can use Bambi to very quickly specify a model, and subsequently elaborate on or extend the model in native Python code. Subject to computational limitations (see below), users can in principle fit any model that allows variables to be modeled as probability distributions--including arbitrary non-linear transformations. From the standpoint of psychological science, this opens the door to a unified analytical framework for modeling a diverse array of social, cognitive, or biological process models (e.g., diffusion models of reaction time, signal detection models, neuronal spiking models, etc.).

The present implementation of Bambi has a number of limitations worth noting--most of which reflect either inherent limitations of the MCMC approach or current feature limitations in the PyMC3/Theano stack Bambi relies on. First, MCMC sampling is a computationally intensive process. While the highly optimized numerical packages that Bambi is built on make it possible to fit a very wide range of models on standard hardware, very large models--e.g., those involving hundreds of thousands of observations and/or thousands of parameters--remain impractical to fit using an MCMC-based approach. Fortunately, new methods that efficiently approximate the optimal Bayesian solution are continuously being implemented in PyMC3, and hence are easy to support in Bambi (e.g., Bambi users can already take advantage of PyMC3's support for automatic differentiation variational inference; Kucukelbir et al., 2016).

Second, PyMC3--and hence, Bambi--currently has limited support for Bayesian model comparison. While some metrics are already available (e.g., the DIC and WAIC), there is no support for the Bayes factor (BF) that has been vocally championed by Bayesian psychologists in recent years ((Richard D. Morey, Jan-Willem, & Rouder, 2016; Rouder, Morey, Verhagen, Province, & Wagenmakers, 2016); but see also (Kruschke, 2011; Kruschke & Liddell, n.d.)). Unfortunately, this limitation reflects the inherent difficulty of computing BFs, which require integration over the full parameter space, and hence are typically only available for analytically tractable models (e.g., t-tests or ANOVAs with highly constrained prior distributions).

Importantly, the open-source nature of Bambi--and in particular, its reliance on numerical packages that already have well-established userbases and developer communities--means that the available functionality will continue to grow rapidly. Our hope is that many researchers steeped in frequentist analysis methods will find the Bambi model-fitting interface sufficiently intuitive and familiar to warrant adopting a Bayesian approach for at least some classes of common analysis problems.

References

Andrieu, C., de Freitas, N., Doucet, A., & Jordan, M. I. (2003). An Introduction to MCMC for Machine Learning. *Machine Learning*, *50*(1-2), 5–43.

Barr, D. J., Levy, R., Scheepers, C., & Tily, H. J. (2013). Random effects structure for confirmatory hypothesis testing: Keep it maximal. *Journal of Memory and Language*, *68*(3). http://doi.org/10.1016/j.jml.2012.11.001

Bastien, F., Lamblin, P., Pascanu, R., Bergstra, J., Goodfellow, I., Bergeron, A., … Bengio, Y. (2012, November 23). *Theano: new features and speed improvements*. *arXiv [cs.SC]*. Retrieved from http://arxiv.org/abs/1211.5590

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., … Bengio, Y. (2010). Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for scientific computing conference (SciPy)* (Vol. 4, p. 3). Austin, TX.

Bolker, B. M., Brooks, M. E., Clark, C. J., Geange, S. W., Poulsen, J. R., Stevens, M. H. H., & White, J.-S. S. (2009). Generalized linear mixed models: a practical guide for ecology and evolution. *Trends in Ecology & Evolution*, *24*(3), 127–135.

Chung, Y., Gelman, A., Rabe-Hesketh, S., Liu, J., & Dorie, V. (2015). Weakly Informative Prior for Point Estimation of Covariance Matrices in Hierarchical Models. *Journal of Educational and Behavioral Statistics: A Quarterly Publication Sponsored by the American Educational Research Association and the American Statistical Association*, *40*(2), 136–157.

de Schoot R. Winter S. Ryan O. Zondervan-Zwijnenburg M. & Depaoli S. Van, V. (2016). A Systematic Review of Bayesian Papers in Psychology: The Last 25 Years. *Psychological Methods*.

Dropmann, D., Verhagen, A. J., & Wagenmakers, E. J. (2015). JASP (Version 0.7)[Computer

software].

Egan, J. P. (1975). *Signal detection theory and {ROC} analysis* (p. Series in Cognition and Perception). Academic Press.

Gelman, A., Andrew, G., Aleks, J., Pittau, M. G., & Yu-Sung, S. (2008). A weakly informative default prior distribution for logistic and other regression models. *The Annals of Applied Statistics*, *2*(4), 1360–1383.

Goldberg, L. R. (1999). A broad-bandwidth, public domain, personality inventory measuring the lower-level facets of several five-factor models. *Personality Psychology in Europe*. Retrieved from http://projects.ori.org/lrg/PDFs_papers/A%20broad-bandwidth%20inventory.pdf

Grucza, R. A., & Goldberg, L. R. (2007). The comparative validity of 11 modern personality inventories: predictions of behavioral acts, informant reports, and clinical indicators. *Journal of Personality Assessment*, *89*(2), 167–187.

Hoffman, M. D., & Gelman, A. (2014). The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research: JMLR*, *15*, 30.

Hunsley, J., & Meyer, G. J. (2003). The incremental validity of psychological testing and assessment: conceptual, methodological, and statistical issues. *Psychological Assessment*, *15*(4), 446–455.

Judd, C. M., Westfall, J., & Kenny, D. A. (2012). Treating stimuli as a random factor in social psychology: a new and comprehensive solution to a pervasive but largely ignored problem. *Journal of Personality and Social Psychology*, *103*(1), 54–69.

Kenny, D. A., & La Voie, L. (1984). The Social Relations Model. In Leonard Berkowitz (Ed.), *Advances in Experimental Social Psychology* (Vol. Volume 18, pp. 141–182). Academic Press.

Kruschke, J. K. (2011). Bayesian Assessment of Null Values Via Parameter Estimation and Model Comparison. *Perspectives on Psychological Science: A Journal of the Association for Psychological Science*, *6*(3), 299–312.

Kruschke, J. K., & Liddell, T. M. (n.d.). The Bayesian New Statistics: Two Historical Trends Converge. *SSRN Electronic Journal*. http://doi.org/10.2139/ssrn.2606016

Kucukelbir, A., Tran, D., Ranganath, R., Gelman, A., & Blei, D. M. (2016, March 2). *Automatic Differentiation Variational Inference*. *arXiv [stat.ML]*. Retrieved from http://arxiv.org/abs/1603.00788

Morey, R. D., Jan-Willem, R., & Rouder, J. N. (2016). The philosophy of Bayes factors and the quantification of statistical evidence. *Journal of Mathematical Psychology*, *72*, 6–18.

Morey, R. D., Rouder, J. N., & Jamil, T. (2014). BayesFactor: Computation of Bayes factors for common designs. *R Package Version 0.9*.

Patil, A., Huard, D., & Fonnesbeck, C. J. (2010). PyMC: Bayesian Stochastic Modelling in Python. *Journal of Statistical Software*, *35*(4), 1–81.

Raftery, A. E., & Lewis, S. M. (1996). Implementing mcmc. *Markov Chain Monte Carlo in Practice*. Retrieved from https://books.google.com/books?hl=en&lr=&id=TRXrMWY_i2IC&oi=fnd&pg=PA115&dq=raftery%2Blewis&ots=7hYmxkLupp&sig=mGIhQdbzbghLIWmgBlmI1W_LAlQ

Ranganath, R., Gerrish, S., & Blei, D. M. (2014). Black Box Variational Inference. *AISTATS*. Retrieved from http://www.jmlr.org/proceedings/papers/v33/ranganath14.pdf

Rouder, J. N., Morey, R. D., Verhagen, J., Province, J. M., & Wagenmakers, E.-J. (2016). Is There a Free Lunch in Inference? *Topics in Cognitive Science*, *8*(3), 520–547.

Salvatier, J., Wiecki, T. V., & Fonnesbeck, C. (2016). Probabilistic programming in Python using PyMC3. *PeerJ Computer Science*, *2*, e55.

Strack, F., Martin, L. L., & Stepper, S. (1988). Inhibiting and facilitating conditions of the human smile: a nonobtrusive test of the facial feedback hypothesis. *Journal of Personality and Social Psychology*, *54*(5), 768–777.

Stroup, W. W. (2012). *Generalized Linear Mixed Models: Modern Concepts, Methods and Applications*. Taylor & Francis.

Swets, J. A. (2014). *Signal Detection Theory and ROC Analysis in Psychology and Diagnostics: Collected Papers*. Taylor & Francis.

van Allen Z. M. Vandekerckhove M. Wainwright B. Wayand J. F. Zeelenberg R. Zetzer E. E. Zwaan R. A., W. E.-J. B. T. D. L. G. Q. F. A. A. A. R. B. A. D. N. A. E. S. B. S. D. B.-H. E.-M. B. L. C. C. T. L. C.-J. R. J. C. C. A. C. N. S. C. K. T. C. A. C. L. D. J. M. D. K. F. A. H. F. F. H. U. H. K. J. J. J. L. H. K. O. K. C. K. S. L. P. L. J. D. L. S. L. J. L. D. N. C. N. O. S. Ö. A. A. P.-U. A. P. P. B. P. C. R. S. R. T.-A. R. R. I. S. M. S.-S. N. B. S. K. S. J. A. S. T. G. T. J. M., Jr. (2016). Registered Replication Report: Strack, Martin, & Stepper (1988). *Perspectives on Psychological Science: A Journal of the Association for Psychological Science*.

Westfall, J., Judd, C. M., & Kenny, D. A. (2015). Replicating studies in which samples of participants respond to samples of stimuli. *Perspectives on Psychological Science: A Journal of the Association for Psychological Science*, *10*(3), 390–399.

Westfall, J., & Yarkoni, T. (2016). Statistically Controlling for Confounding Constructs Is Harder than You Think. *PloS One*, *11*(3), e0152719.

Appendix: Statistical details about the default priors

Consider a regression equation:

$$Y = \beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p + e.$$

Our goal is to devise a set of default priors for the regression coefficients that--in the absence of any additional information supplied by the user--will still allow one to obtain reasonable parameter estimates *in general*. One obviously unsatisfactory solution would be to just use, say, standard Normal distributions for all the priors, i.e.:

$$\beta_j \sim N(0, 1).$$

However, this ignores the fact that the observed variables may all be on wildly different scales. So for some predictors the prior may be extremely narrow or "informative," shrinking estimates strongly toward 0, while for other predictors the prior may be extremely wide or "vague," leaving the estimates essentially unchanged.

One remedy to this differential shrinkage issue could be to set the standard deviation of the prior to a very large value, so that the prior is likely to be wide relative to almost any predictor variables. For example,

$$\beta_j \sim N(0, 10^{10}).$$

This is better, although in principle it suffers from the same problem--that is, a user could still conceivably use variables for which even this prior is implausibly narrow (although in practice it would be unlikely). A different but related worry is that different scalings of the variables in the same dataset--for example, due to changing the units of measurement--will lead to the priors having different levels of informativeness. This seems undesirable because scaling and shifting the variables has no meaningful consequence for traditional test statistics and standardized effect sizes (with some obvious exceptions pertaining to intercepts and models with interaction terms).

The approach we take for the default priors in bambi is to set the prior indirectly by defining the prior on the corresponding partial correlation, and then seeing what scale this implies for the prior on the raw regression coefficient scale.

One can transform the multiple regression coefficient for the predictor X_j into its corresponding partial correlation (i.e., the partial correlation between the outcome and the predictor, controlling for all the other predictors) using the identity:

$$\rho^p_{YX_j} = \beta_j \sqrt{\frac{(1 - R^2_{X_j X_{-j}})\text{var}(X_j)}{(1 - R^2_{YX_{-j}})\text{var}(Y)}},$$

where β_j is the slope for X_j, $R^2_{X_j X_{-j}}$ is the R^2 from a regression of X_j on all the other predictors (ignoring Y), and $R^2_{YX_{-j}}$ is the R^2 from the regression of Y on all the predictors *other than* X_j.

Now we define some prior distribution for $\rho^p_{YX_j}$ that has mean zero and standard deviation σ_ρ. This implies that

$$\begin{aligned}\text{var}(\beta_j) &= \text{var}(\rho^p_{YX_j} \sqrt{\frac{(1 - R^2_{YX_{-j}})\text{var}(Y)}{(1 - R^2_{X_j})\text{var}(X_j)}}) \\ &= \frac{(1 - R^2_{YX_{-j}})\text{var}(Y)}{(1 - R^2_{X_j})\text{var}(X_j)} \sigma^2_\rho\end{aligned}.$$

One can tune the width or informativeness of this prior by setting different values of σ_ρ, corresponding to different standard deviations of the distribution of plausible partial correlations. Our default prior is a Normal distribution with zero mean and standard deviation following this scheme, with $\sigma_\rho = \sqrt{1/3} \approx .577$, which is the standard deviation of a flat prior in the interval [-1,1]. We allow users to specify their priors in terms of σ_ρ, or in terms of four labels: “narrow” meaning $\sigma_\rho = .2$, “medium” meaning $\sigma_\rho = .4$, “wide” meaning $\sigma_\rho = \sqrt{1/3}$ (i.e., the default), or “superwide” meaning $\sigma_\rho = .8$. (Note that the maximum possible standard deviation of a distribution of partial correlations is 1, which would be a distribution with half of the values at $\rho^p_{YX_j} = -1$ and the other half at $\rho^p_{YX_j} = 1$.) Viewed from this partial correlation perspective, it seems hard to theoretically justify anything wider than our “wide” default, since this would correspond to something that is wider than a flat prior on the partial correlation scale (although we note that, purely practically speaking, there are often no discernible problems in using such a wider prior).

The default prior for the intercept β_0 must follow a different scheme, since partial correlations with the constant term are undefined. We first note that in ordinary least squares (OLS) regression $\beta_0 = \bar{Y} - \beta_1 \bar{X}_1 - \beta_2 \bar{X}_2 - \ldots$. So we can set the mean of the prior on β_0 to

$$\text{E}[\beta_0] = \bar{Y} - \text{E}[\beta_1]\bar{X}_1 - \text{E}[\beta_2]\bar{X}_2 - \ldots.$$

In practice, the priors on the slopes will typically be set to have zero mean, so the mean of the prior on β_0 will typically reduce to $\bar{Y}$.

Now for the variance of β_0 we have (assuming independence of the slope priors):

$$\text{var}(\beta_0) = \bar{X}_1^2 \text{var}(\beta_1) + \bar{X}_2^2 \text{var}(\beta_2) + \ldots.$$

In other words, once we have defined the priors on the slopes, we can combine this with the means of the predictors to find the implied variance of β_0. Our default prior for the intercept is a Normal distribution following this scheme, and it assumes that all of the slopes were set to have "wide" priors (as defined above), regardless of what the user actually selected for the width of the slope priors.