Re-Evaluating Machine Learning for MRP Given the Comparable Performance of (Deep) Hierarchical Models*

Max Goplerud†

Abstract

Multilevel Regression and Post-Stratification (MRP) is a popular use of hierarchical models in political science. Multiple papers have suggested that relying on machine learning methods can give substantially better performance than traditional approaches using hierarchical models. However, these comparisons are often unfair to traditional techniques as they omit possibly important interactions or non-linear effects. I show that complex ("deep") hierarchical models that include interactions can nearly match or out-perform state-of-the-art machine learning methods. Combining multiple models into an ensemble can improve performance, although deep hierarchical models are themselves given considerable weight in these ensembles. The main limitation to using deep hierarchical models is speed; this paper derives new techniques to further accelerate estimation using variational approximations and provides accompanying software that uses weakly informative priors and can estimate non-linear effects using splines. This allows flexible and complex hierarchical models to be fit as quickly as many comparable machine learning techniques.

Words: 3,992

Key Words: multilevel regression and post-stratification (MRP), machine learning, hierarchical models, variational inference

*I thank Michael Auslen, James Bisbee, Danny Choi, Jeff Gill, Kosuke Imai, Gary King, Shiro Kuriwaki, Dustin Tingley, and participants at PolMeth 2021 for comments on this draft. This research was supported in part by the University of Pittsburgh Center for Research Computing through the resources provided. Open-source software implement the methods in this paper is available at `https://github.com/mgoplerud/vglmer`. A demonstration of how to use the accompanying software is available in Appendix F. All remaining errors are my own.

†Assistant Professor, Department of Political Science, University of Pittsburgh. 4600 Wesley W. Posvar Hall, Pittsburgh, PA 15260. Email: mgoplerud@pitt.edu. URL: `https://mgoplerud.com`

The growing popularity of machine learning continues to revolutionize parts of political science by allowing easy estimation of flexible and powerful models. One increasingly popular application is using machine learning when performing "multilevel regression and post-stratification" (MRP) to extrapolate nationally representative surveys to smaller geographic units such as states (Park, Gelman and Bafumi 2004; Lax and Phillips 2009). MRP is a two-step process that begins by fitting a predictive model to the survey using demographic and state-level information. Next, opinion estimates for the states are obtained by a weighted average of the predicted values for various demographic groups inside of that state using their known distribution. While the performance of MRP depends on *both* steps, multiple papers have found that using machine learning for the predictive model out-performs traditional methods ("multilevel regression") by considerable margins (e.g. Goplerud et al. 2018; Ornstein 2020; Bisbee 2019; Broniecki, Leemann and Wüest 2021). A plausible justification is that the linear, additive, nature of traditional models is insufficient to capture the complex relationship between the covariates and the outcome.

The reliance on *simple* hierarchical models, however, unnecessarily limits their usefulness. Unlike some machine learning methods that can automatically estimate interactions (or non-linear effects of continuous predictors), hierarchical models can only estimate interactions that the researcher has explicitly included. This is both a strength and a weakness. While hierarchical models are highly modular and allow the researcher to explicitly incorporate domain-specific knowledge as to important predictors or interactions, there is a risk of mis-specification—and thus worse performance—if important interactions are omitted. Thus, a "fair" test of MRP's performace must examine a model that explicitly includes many possibly relevant interactions. Ghitza and Gelman (2013) illustrate this by adding a broad array of interactions and uncovering considerably more subtle results than traditional methods could identify. Following their usage, I refer to complex hierarchical models that explicitly include interactions or non-linear effects as "deep MRP".[1]

[1]This method is distinct from "deep learning" (e.g. involving neural networks).

Thus, despite the understandable enthusiasm for applying machine learning to MRP, it simply unknown in a systematic way whether machine learning out-performs deep MRP. The main reason for this gap in the literature is a practical one. Traditional methods for estimating deep MRP may need nearly twenty random effects to capture the underlying heterogeneity and thus are usually very slow to estimate. Given that one might wish to fit these models repeatedly (e.g. comparing different specifications), this has quite reasonably caused researchers to "rule out" deep MRP.

Fortunately, recent work has shown that deep MRP can be estimated very quickly using variational inference while producing very similar point estimates to traditional methods (Goplerud 2021). However, that paper only tested those algorithms on the single dataset from Ghitza and Gelman (2013) and did not compare against machine learning techniques. My initial systematic tests found that those algorithms performed unfavorably against machine learning. This paper provides two improvements to existing variational methods that result in competitive performance: First, Goplerud (2021) relied on an improperly calibrated prior that often resulted in too little regularization. Second, those algorithms cannot capture non-linear effects of continuous covariates (e.g., presidential vote share).

Those concerns are addressed by, first, extending the variational algorithms to include a weakly informative prior (Huang and Wand 2013) that can more appropriately regularize random effects and, second, allowing the use of penalized splines for continuous predictors. After implementing a number of novel computational techniques to accelerate estimation, the accompanying open-source software can fit highly flexible deep MRP in minutes—rather than the hours possibly needed for traditional approaches.

This paper illustrates the importance of deep MRP by reanalyzing two papers that suggest machine learning methods clearly out-perform MRP (Ornstein 2020; Bisbee 2019). It demonstrates two important stylized facts: Deep multilevel models (i) are given considerable weight in an ensemble of machine learning methods and (ii) are competitive with BART in terms of performance. While recent work reports that BART performs noticeably better

than traditional MRP (Bisbee 2019), I demonstrate that this is not the case. I show that, especially at moderate sample sizes, BART usually only slightly out-performs even traditional MRP. Thus, while machine learning methods that combine many methods together in an ensemble can improve performance, (deep) MRP should continue to be used as a highly competitive single method or in any ensemble approach.

1 Fitting Deep MRP Fast

The key limitation in fitting MRP with interactions is the speed of estimation. Earlier research has shown that fitting a single deep MRP model can take multiple hours (e.g. Goplerud et al. 2018; Goplerud 2021). This is because of the presence of high-dimensional integrals that traditional methods either numerically approximate or address using Bayesian methods.

Variational inference provides a different approach for fast estimation; the goal is to find the best approximating distribution to the posterior given some simplifying assumption—usually that blocks of parameters are independent (Grimmer 2010). However, the accuracy of this approximation can depend heavily on the specific problem, and thus needs extensive testing to ensure its reliability. Goplerud (2021) derived a new general algorithm for binomial hierarchical models and conducted extensive explorations of its performance on the single dataset considered in Ghitza and Gelman (2013). Those algorithms fit an extremely complex hierarchical model in around one minute—versus hour(s) for existing approaches. It demonstrated excellent performance by recovering posterior means on coefficients and predictions that closely aligned with the gold standard approach of Bayesian estimation.[2] Appendix A provides a full exposition of the variational algorithm.

To illustrate the extensions in this paper, I focus on a simplified MRP model: Equation 1 shows a hierarchical model without fixed effects and with a random intercept for state and a

[2]As with most variational methods, it under-estimates posterior uncertainty; Goplerud (2021) provides a post-estimation adjustment to mitigate some of this problem.

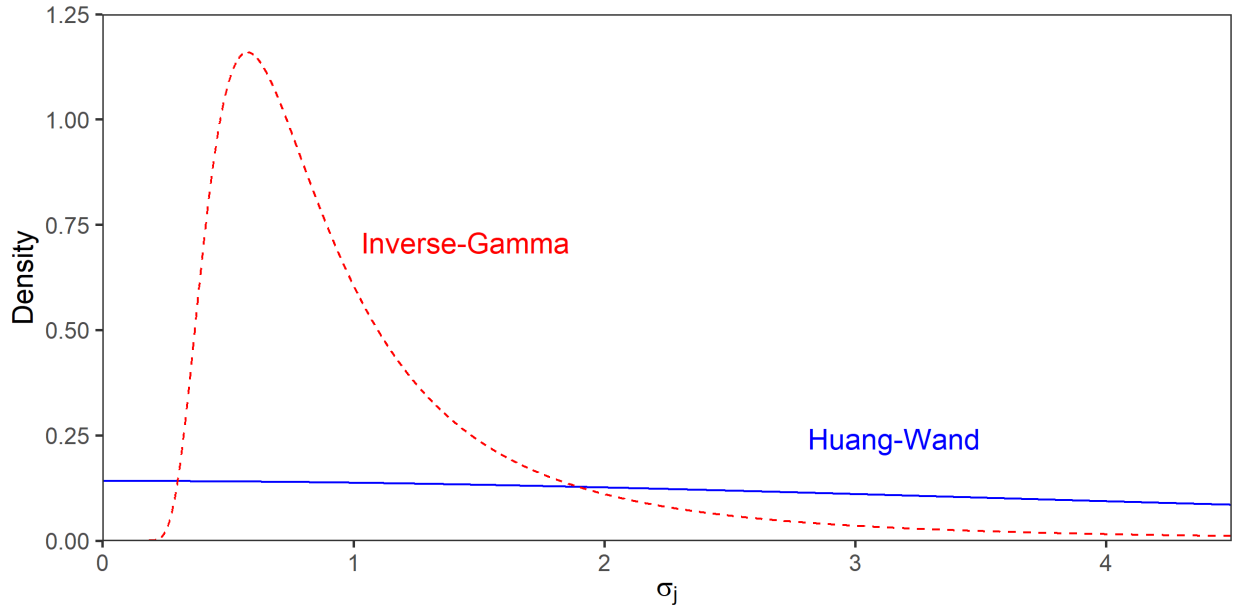
random intercept for race, where y_i is the (binary) response for observation i. The notation follows Gelman and Hill (2006) where $\alpha^{\text{state}}_{g[i]}$ select the random effect for state g of which observation i is a member. Appendix A shows the generalization to random slopes, arbitrary numbers of random effects, and fixed effects.

$$
\begin{aligned}
&y_i \sim \text{Bern}(p_i); \quad p_i = \frac{\exp(\psi_i)}{1+\exp(\psi_i)}; \quad \psi_i = \beta_0 + \alpha^{\text{state}}_{g[i]} + \alpha^{\text{race}}_{g'[i]}; \\
&\alpha^{\text{state}}_{g} \sim N(0, \sigma^2_{\text{state}}); \quad \alpha^{\text{race}}_{g'} \sim N(0, \sigma^2_{\text{race}}); \quad p(\beta_0) \propto 1; \\
&\sigma^2_j \sim p_0(\sigma^2_j) \quad \text{for} \quad j \in \{\text{state}, \text{race}\}
\end{aligned}
\tag{1}
$$

The choice of prior on the variance of the random effect $p_0(\sigma^2_j)$ is a difficult task. Some inferential techniques assume a flat prior; a risk of this strategy is that the approximation degenerates and estimates σ^2_j equal to zero. This sets all random effects estimates equal to zero, and is rather common for the Laplace approximation in `glmer` (Chung et al. 2015). A proper prior prevents this problem and thus is preferable. An Inverse-Gamma prior is a popular choice, but it is difficult to calibrate the strength correctly (Gelman 2006). Figure 1 illustrates this by showing the prior used in Goplerud (2021) ("Inverse-Gamma") against the Huang and Wand (2013) prior employed in this paper. The latter prior implies the popular half-t prior on the standard deviation σ_j (Gelman 2006), and it can be generalized to multi-dimensional random effects. In that case, it imposes half-t priors on each marginal standard deviation while maintaining (if desired) a uniform prior on the correlations. Appendix A provides details.

Figure 1 shows that Goplerud (2021)'s prior puts effectively no mass on small values of σ_j (e.g., $P(\sigma_j \leq 0.25) \approx 0.0003$). Thus, in the event where the true value is small (i.e. the random effect is mostly irrelevant), the prior results in too large estimates of σ_j thereby under-regularizing ("under-pooling") the coefficients which likely results in poorer performance. By contrast, the Huang-Wand prior puts non-trivial weight on very small σ_j

Figure 1: Comparing Prior Density for Random Effect Standard Deviation σ_j



Note: The dashed line shows the prior density on σ_j given an Inverse-Gamma prior on σ_j^2 with $\alpha_0 = 1$ and $\beta_0 = 1/2$. The solid line shows a Huang-Wand prior on σ_j^2 with $\nu = 2$ and $A = 5$ (i.e. half-t on σ_j).

and thus allows for strong regularization when appropriate. Appendix A provides a stylized example of this phenomenon.

Unfortunately, Appendix A.5 illustrates that naively incorporating the Huang-Wand prior dramatically increases estimation time. While it does increase the time per iteration, the major problem is that estimation requires five-to-ten times more iterations to converge. Thus, a key contribution of this paper is to accelerate variational algorithms when this more appropriate prior is employed. Appendix A provides a full explanation of the techniques employed: (i) quadratic interpolation and (ii) a novel application of parameter expansion.

The model in Equation 1 can be extended by adding many interactions between geographic and demographic factors ("deep MRP"; Ghitza and Gelman 2013). However, Broniecki, Leemann and Wüest (2021) note that additional state-level predictors (e.g. unemployment rate) may also provide considerable benefits. Unlike hierarchical models, many machine learning methods can automatically estimate non-linear effects or interactions between these continuous predictors whereas they must be specified explicitly for MRP.

I address that scenario by allowing estimation of non-linear effects using splines as in a generalized additive model. Estimation is straightforward by representing splines as additional hierarchical terms; Appendix A provides details.

Appendix B provides simulations to illustrate the importance of using hierarchical models that include interactions or non-linear effects. It shows that ignoring important interactions or non-linearities hurts the performance of hierarchical models vis-à-vis alternative models, especially as the sample size increases. But, after those terms are included, hierarchical models perform well even against machine learning methods.

2 Comparing Methods for Fitting MRP

To compare deep hierarchical models against machine learning systematically, I use Buttice and Highton (2013)'s popular dataset for validating new methods for MRP (e.g, Ornstein 2020; Bisbee 2019; Broniecki, Leemann and Wüest 2021). It consists of eighty-nine policy questions that are collected from multiple years of the National Annenberg Election Studies (2000, 2004, 2008) and the Cooperative Congressional Election Studies (2006 and 2008). The benefit of these large samples is that it is possible to use the entire dataset to get a "ground truth" by taking the observed average in each state while drawing a smaller sub-sample (e.g. 1,500 respondents) to mimic the conditions under which a researcher would need to apply MRP to obtain reliable state-level estimates.

Existing comparisons, however, only rely on a simple hierarchical model outlined below (Equation 2), following the original specification in Buttice and Highton (2013). The model includes random effects for age, education (`educ`), gender-race combination (`gXr`), state, and region. The state-level continuous predictors `pvote` (state-level Republican presidential two-party vote share) and `relig` (share of population identifying as Evangelical Protestant or Mormon) are indexed with $g[i]$ as they are constant within a state.

$$\Pr(y_i = 1) = \text{logit}^{-1}\left(\begin{array}{l}\beta_0 + \beta_{\text{pvote}} \cdot \texttt{pvote}_{g[i]} + \beta_{\text{relig}} \cdot \texttt{relig}_{g[i]} + \\ \alpha^{\text{age}}_{g[i]} + \alpha^{\text{educ}}_{g[i]} + \alpha^{\text{gXr}}_{g[i]} + \alpha^{\text{state}}_{g[i]} + \alpha^{\text{region}}_{g[i]}\end{array}\right) \tag{2}$$

$$\alpha^j_g \sim N(0, \sigma^2_j) \quad \text{for all } j \text{ and } g$$

This model includes no interactions between variables or non-linear effects on continuous predictors, and thus is likely insufficiently rich to capture the true underlying relationship. It is reasonable to suspect that a "properly specified" MRP model should include at least some interactions to be competitive with methods that can automatically learn interactions or non-linearities.

I consider three expansions of this model's hierarchical component. First, I consider a deep MRP where all two-way interactions between demographics and geography are included, as well as a triple interaction between the three demographic variables. Second, I add splines to capture possible non-linear effects in the state-level continuous variables. Table 1 summarizes the specifications; Appendix F provides a demonstration of how to fit these models in the accompanying software.[3]

Table 1: Deep MRP Specifications

Model	`pvote` and `relig`	Demographics and State
Simple	Linear	Additive
Deep	Linear	Interacted
Splines	Splines	Additive
Combined	Splines	Interacted

Note: The second column indicates how these two variables are included. It is either "Linear" (Equation 2) or "spline" where a spline is used to allow for non-linear effects for each variable. The third column indicates how the random effects on age, education, gender × race, state, and region are included. Additive refers to five random effects added together (Equation 2). "Interacted" refers to including the interactions noted in the main text alongside the additive terms.

It is important to stress that this paper tracks the existing analyses comparing machine learning and MRP as closely as possible. There are thus other specifications that likely

[3]All methods use a Huang-Wand prior for each random effect; hyper-parameters are identical to Figure 1.

improve upon Table 1, although I show that adding this set of interactions enables MRP to perform competitively against state-of-the-art machine learning techniques.

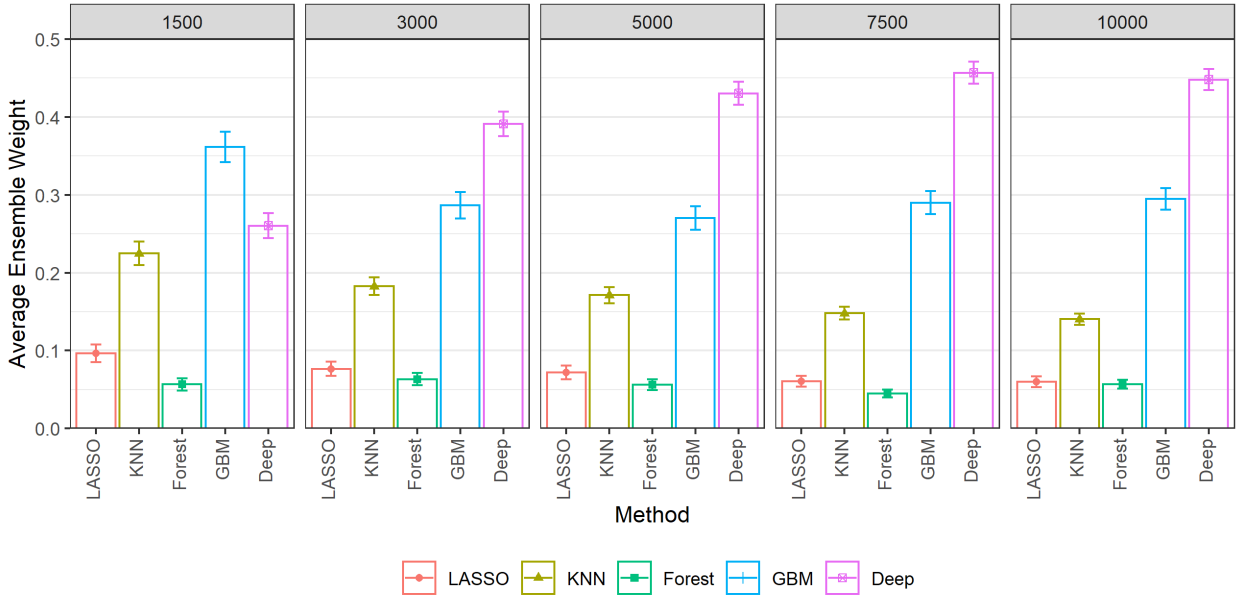
3 Deep MRP in an Ensemble

The first comparison explores whether deep MRP adds much benefit when used alongside a suite of machine learning methods. I begin by using a technique known as "stacking" that takes the predictions of many different methods and combines them into a single prediction known as an ensemble. Ornstein (2020) applied this method to MRP and found considerable gains over traditional methods. The method performs K-fold cross-validation to get an out-of-sample prediction for each observation in the survey using each constituent model. The out-of-sample predictions are combined to see which weighted average (convex combination) best predicts the outcome, and then this is post-stratified. It is often the case that the ensemble out-performs any single method (Ornstein 2020; Broniecki, Leemann and Wüest 2021), although this is not guaranteed and can be empirically assessed by, for example, using a held-out dataset.

A useful property of ensembles is the ability to compare the weights given to the constituent models. The weights reflect both the performance of the method and its "distinctiveness" from the other methods in the ensemble. Using each survey in Buttice and Highton (2013), I drew ten different samples of varying sizes and estimated an ensemble using five-fold cross-validation with the models in Ornstein (2020) where I swapped the traditional ("Simple") MRP model with the Deep MRP model from Table 1. Figure 2 summarizes the weights given to each model, averaging across the surveys and simulations.

The results provide clear support for the importance of deep MRP in an ensemble: It is the highest weighted method when the sample size is over 1,500 and is given over 40% of the total weight when the sample size is 5,000 or higher. The performance of deep MRP is corroborated by the fact that, of the methods in the ensemble, it has the lowest cross-

Figure 2: Weights Given to Models in Ensemble



Note: This figure shows the ensemble weights averaged across all surveys and ten simulations per survey. Each panel reports the sample size of the survey. 95% confidence intervals are shown. The first four methods are from Ornstein (2020): LASSO, k-Nearest Neighbors (KNN), Random Forest (Forest), Gradient Boosting Machine (GBM). The final method is "Deep" MRP from Table 1.

validated error on the survey data when $N > 1500$. In terms of computational time, fitting this deep model on the full survey takes around one minute for the largest sample size of 10,000 observations. Thus, deep MRP can be added to an ensemble with limited cost.

Appendix D provides additional analyses. First, it compares a larger ensemble that includes the four variational MRP methods from Table 1.[4] It corroborates the above figure; the MRP models, collectively, are given around 40-50% of the weight. It also shows an expected trade-off between traditional and deep MRP where traditional (non-interactive) methods are given decreasing weight as the sample size increases. This suggests the ensemble upweights more complex methods as the amount of data increases. The spline-based methods receive relatively low weight, but this may be due to the limited variation in the continuous variables that are measured at the state-level.[5] Second, it confirms that, in terms of raw

[4]It also examines the "Deep" hierarchical model with an Inverse-Gamma/Wishart prior.

[5]Appendix B provides a simulated example where the splines are critically important to strong performance.

performance, a well-designed ensemble usually beats any single constituent method. The five-model ensemble beats all of its constituent methods by more than 5% on at least one sample size considered.

4 MRP and BART

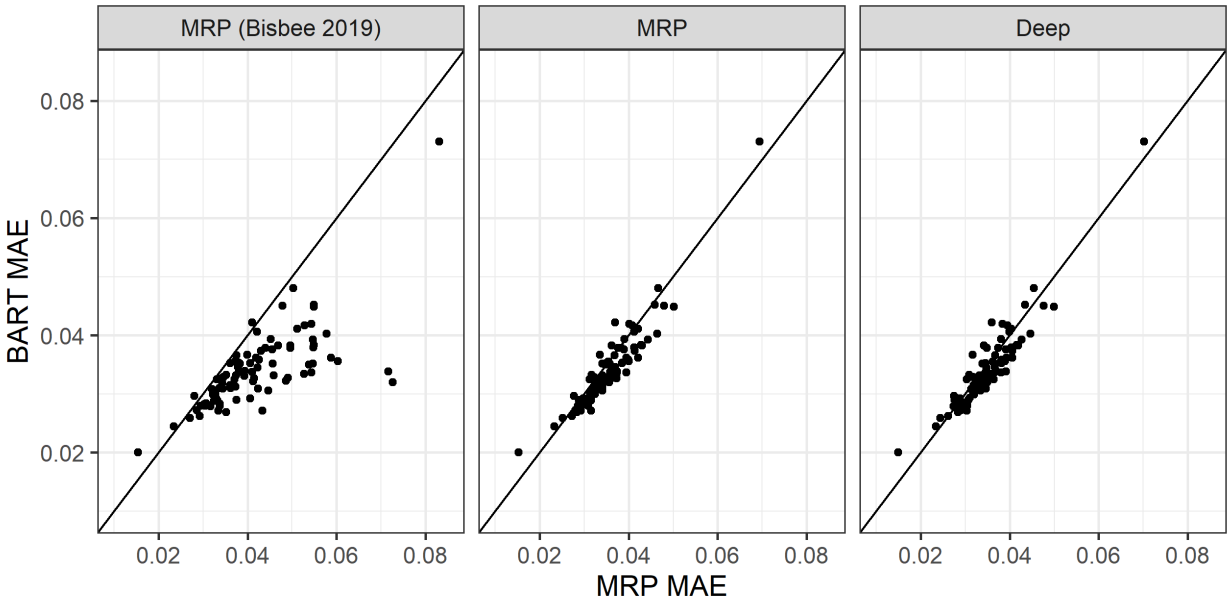
One limitation of ensembles is appropriately quantifying uncertainty of the post-stratified estimates. This is challenging because it may be difficult to quantify the uncertainty of the estimates from the individual machine learning methods used in the ensemble.[6] It also requires careful work to interpret the effects of the included variables. Thus, researchers often seek to rely on a single model that can incorporate uncertainty and remains highly flexible. To that end, Bayesian Additive Regression Trees (BART; Chipman, George and McCulloch 2010) are an attractive choice. They implement the popular "random forest" method in a Bayesian framework that allows for quantification of uncertainty. Bisbee (2019) applies BART to MRP and reports that it substantially out-performs (traditional) MRP. The magnitude of the improvement is large (e.g. around 20-30% decrease in mean absolute error). This motivates an initial question: Does BART improve upon deep MRP?

After some preliminary exploration, I discovered an error in Bisbee (2019)'s replication archive. Appendix E describes it in detail; in brief, it arbitrarily injected random noise into the MRP estimates at the prediction stage. When this is corrected, traditional MRP's performance increases markedly and is only slightly beaten by BART.

Following the main analysis in Bisbee (2019), Figure 3 shows the predictive accuracy on the surveys in Buttice and Highton (2013) for a sample with 1,500 observations. Appendix E replicates other analyses in Bisbee (2019). For simplicity, I show only three methods: the traditional ("Simple") MRP following Bisbee (2019)'s provided code, a corrected traditional MRP, and Deep MRP estimated using variational inference (see Table 1).

[6]Broniecki, Leemann and Wüest (2021) suggest bootstrapping. They show promising results, but this can be computationally expensive and thus sometimes a single method is desirable.

Figure 3: Visualizing Performance: MRP versus BART



Note: Each plot compares the mean absolute error (MAE) of BART and MRP, averaged across two-hundred simulations per survey. Points below the 45-degree line indicate that MRP performs worse. The three methods shown are the traditional MRP and prediction method in Bisbee (2019) ("MRP (Bisbee 2019)"), a MRP with the same specifications but a corrected prediction ("MRP"), and a deep MRP fit using variational inference ("Deep").

Fixing the error shows a noticeably different story; rather than being clearly beaten by BART, traditional MRP looks visually similar to BART in terms of its error across surveys. Table 2 provides a more concise quantitative summary. It shows the percentage gap in mean absolute error versus BART averaged across the eighty-nine surveys: $(\text{MAE}_k - \text{MAE}_{\text{BART}})/\text{MAE}_{\text{BART}} \cdot 100$ where MAE_{k} indicates the mean absolute error of the model k averaged across two-hundred simulations. A positive number indicates that BART out-performs the other method. This measure is *relative* as BART and MRP both decrease the observed mean absolute error as sample size increases.

Table 2 shows that BART does outperform traditional MRP, but it does so by quite small margins (1%-4%) and its relative advantage declines as sample size increases. Deep MRP performs slightly better versus BART; for modest sample sizes (3,000-6,000), it actually slightly out-performs BART although it does slightly worse at small and very large sample

Table 2: Relative Mean Absolute Error versus BART

	Sample Size					
Method	1500	3000	4500	6000	7500	10000
MRP (Bisbee 2019)	22.56	26.30	30.99	35.16	38.72	44.25
MRP	4.54	1.66	1.26	1.04	1.03	1.08
Deep	2.62	-1.04	-1.01	-0.61	0.10	1.11

Note: This table reports percentage gap in mean absolute error between BART and the alternative methods; positive numbers indicate that BART out-performs its competitor. Figure 3 defines the abbreviations.

sizes. In terms of performance, across all sample sizes, Deep MRP out-performs BART between 45-55% of the time and thus they can be considered to reach an effective "draw" in terms of performance. The table also suggests a small-but-systematic improvement of deep MRP over traditional MRP. Comparing the traditional ("Simple") MRP against deep MRP shows that, except for the largest sample sizes, traditional MRP performs around 1-3% worse than deep MRP in terms of mean absolute error and is beaten around 60-70% of the time.

5 Conclusion

This paper has shown that with recent advances in variational inference and novel technical extensions, it is possible to rapidly estimate deep MRP; Appendix C shows that the deep MRP can be often estimated much more quickly than machine learning methods that require tuning of external hyper-parameters and as quickly as BART without tuning. This allows researchers to seamlessly use deep MRP when repeated estimation is required such as in cross-validation or creating an ensemble.[7]

To understand the relationship between deep MRP and machine learning, I re-examined data from Buttice and Highton (2013) and conducted additional purely synthetic simulations (Appendix B). The results demonstrated clearly that deep MRP is highly competitive in performance—effectively tying the state-of-the-art BART method. Compared to tradi-

[7]If quantification of uncertainty is required, the results in this paper also support estimating a deep hierarchical model using a fully Bayesian approach as it will likely share the strong performance of the variational method.

tional MRP, adding interactions and estimating a deep model results in a small, but systematic and non-trivial, gain in performance at most observed sample sizes. There are also non-performance reasons to prefer hierarchical models including their familiarity to many researchers and the ease of extending them to account for question-specific substantive considerations. Thus, if a single method is desired, (deep) MRP provides a competitive and attractive option compared to commonly used machine learning techniques.

This paper also corroborated the growing consensus that combining high-performing methods can out-perform any single method. Deep MRP also performs very well when used in this context; it often has the strongest out-of-sample predictive accuracy on the survey itself and thus is given high weight in an ensemble. Thus, while there is an important role for machine learning in model-based post-stratification to effectively combine methods, this paper suggests a continued important role for (deep) hierarchical models.

References

Bisbee, James. 2019. "BARP: Improving Mister P Using Bayesian Additive Regression Trees." *American Political Science Review* 113(4):1060–1065.

Broniecki, Philipp, Lucas Leemann and Reto Wüest. 2021. "Improved Multilevel Regression with Post-Stratification Through Machine Learning (autoMrP)." *The Journal of Politics. Advance Access* .

Buttice, Matthew K. and Benjamin Highton. 2013. "How Does Multilevel Regression and Poststratification Perform with Conventional National Surveys?" *Political Analysis* 21(4):449–467.

Chipman, Hugh A, Edward I George and Robert E McCulloch. 2010. "BART: Bayesian additive regression trees." *The Annals of Applied Statistics* 4(1):266–298.

Chung, Yeojin, Andrew Gelman, Sophia Rabe-Hesketh, Jingchen Liu and Vincent Dorie.

2015. "Weakly Informative Prior for Point Estimation of Covariance Matrices in Hierarchical Models." *Journal of Educational and Behavioral Statistics* 40(2):136–157.

Gelman, Andrew. 2006. "Prior Distributions for Variance Parameters in Hierarchical Models." *Bayesian Analysis* 1(3):515–533.

Gelman, Andrew and Jennifer Hill. 2006. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.

Ghitza, Yair and Andrew Gelman. 2013. "Deep Interactions with MRP: Election Turnout and Voting Patterns Among Small Electoral Subgroups." *American Journal of Political Science* 57(3):762–776.

Goplerud, Max. 2021. "Fast and Accurate Estimation of Non-Nested Binomial Hierarchical Models Using Variational Inference." *Bayesian Analysis* Forthcoming.

Goplerud, Max, Shiro Kuriwaki, Marc Ratkovic and Dustin Tingley. 2018. "Sparse Multilevel Regression (and Poststratification [sMRP])." *Unpublished manuscript* .

Grimmer, Justin. 2010. "An Introduction to Bayesian Inference via Variational Approximations." *Political Analysis* 19(1):32–47.

Huang, Alan and Matt P. Wand. 2013. "Simple Marginally Noninformative Prior Distributions for Covariance Matrices." *Bayesian Analysis* 8(2):439–452.

Lax, Jeffrey R. and Justin H. Phillips. 2009. "How Should We Estimate Public Opinion in The States?" *American Journal of Political Science* 53(1):107–121.

Ornstein, Joseph T. 2020. "Stacked Regression and Poststratification." *Political Analysis* 28(2):293–301.

Park, David K., Andrew Gelman and Joseph Bafumi. 2004. "Bayesian Multilevel Estimation with Poststratification: State-Level Estimates from National Polls." *Political Analysis* 12(4):375–385.

Supporting Information for "Re-Evaluating Machine Learning for MRP Given the Comparable Performance of (Deep) Hierarchical Models"

A Models and Inference

This appendix provides information on the model specification noted in the main text as well as brief derivations of the new technical contributions. The main paper presents an example with an intercept and two random effects. It is often convenient to represent the model in a "general design" notation (Zhao et al. 2006). Using the results in Goplerud (2021), I present a model with an arbitrary number of random effects using both the general design (Equation A.1) and Gelman and Hill (2006) notation (Equation A.2). In both cases, I use conditionally conjugate Inverse-Wishart priors on the variance components $\boldsymbol{\Sigma}_j$, a flat prior on the fixed effects $\boldsymbol{\beta}$, and the conventional normal prior on the random effects. $\boldsymbol{z}^b_{i,j}$ represents the covariates for individual i for random effect j, e.g. $\boldsymbol{z}^b_{i,j} = 1$ for a random intercept. Each random effect $j \in \{1, \cdots, J\}$ has g_j levels (e.g. 50 states) and a dimensionality of d_j (e.g. $d_j = 1$ for a random intercept). Thus, $\boldsymbol{\Sigma}_j$ is a $p \times p$ symmetric matrix. Each observation i is Binomial with n_i trials and y_i successes; in this paper, $n_i = 1$ as the outcome is binary.

$$\begin{aligned} &y_i|\boldsymbol{\beta}, \boldsymbol{\alpha} \sim \text{Binom}(n_i, p_i), \quad p_i = \frac{\exp(\psi_i)}{1+\exp(\psi_i)}, \quad \psi_i = \boldsymbol{x}_i^T\boldsymbol{\beta} + \boldsymbol{z}_i^T\boldsymbol{\alpha} \\ &\boldsymbol{\alpha}_j|\boldsymbol{\Sigma}_j \sim N\left(\boldsymbol{0}, \boldsymbol{I}_{g_j} \otimes \boldsymbol{\Sigma}_j\right), \quad \boldsymbol{\Sigma}_j \sim \text{IW}(\nu_j, \boldsymbol{\Phi}_j), \quad p(\boldsymbol{\beta}) \propto 1 \\ &\boldsymbol{z}_{i,j} = \boldsymbol{m}_{i,j} \otimes \boldsymbol{z}^b_{i,j}, \quad \boldsymbol{\alpha}^T = [\boldsymbol{\alpha}_1^T, \cdots, \boldsymbol{\alpha}_J^T], \quad \boldsymbol{z}_i^T = [\boldsymbol{z}_{i,1}^T, \cdots, \boldsymbol{z}_{i,J}^T] \end{aligned} \tag{A.1}$$

$$\begin{aligned} y_i|\boldsymbol{\beta}, \{\{\boldsymbol{\alpha}_{j,g}\}_{g=1}^{g_j}\}_{j=1}^J \sim \text{Binom}(n_i, p_i), \quad p_i = \frac{\exp(\psi_i)}{1+\exp(\psi_i)}, \quad \psi_i = \boldsymbol{x}_i^T\boldsymbol{\beta} + \sum_{j=1}^J \left[\boldsymbol{z}^b_{i,j}\right]^T \boldsymbol{\alpha}_{j,g[i]} \\ \boldsymbol{\alpha}_{j,g}|\boldsymbol{\Sigma}_j \sim N(\boldsymbol{0}_{d_j}, \boldsymbol{\Sigma}_j), \quad \boldsymbol{\Sigma}_j \sim \text{IW}(\nu_j, \boldsymbol{\Phi}_j) \quad \forall (j,g), \quad p(\boldsymbol{\beta}) \propto 1 \end{aligned} \tag{A.2}$$

As posed, this problem is difficult to estimate even with variational techniques. A key move in Goplerud (2021) is to use Polya-Gamma data augmentation (Polson, Scott and Windle 2013) to augment the posterior with one Polya-Gamma variable ω_i for each observation (diagonally stacked into $\boldsymbol{\Omega}$). Polson, Scott and Windle (2013) note the following result for Polya-Gamma variables where a Polya-Gamma variable ω_i has two parameters b, c and the identity below holds for any positive a and b.

$$\frac{\exp(\psi)^a}{[1+\exp(\psi)]^b} = 2^{-b}\int \exp(s\psi - \psi^2/2\omega) f_{PG}(\omega|b,0)d\omega, \quad s = a - b/2 \tag{A.3a}$$

$$\omega \sim PG(b,c) := \omega = \frac{1}{2\pi^2}\sum_{k=1}^{\infty}\frac{Z_k}{(k-1/2)^2 + c^2/(4\pi^2)}, \quad Z_k \overset{i.i.d.}{\sim} \text{Gamma}(b,1) \tag{A.3b}$$

After augmenting the likelihood for each observation with its corresponding Polya-Gamma variable, the relevant portion of the augmented posterior is shown below.

$$p(\boldsymbol{y}, \boldsymbol{\Omega}|\boldsymbol{\alpha}, \boldsymbol{\beta}) \propto \exp\left(\boldsymbol{s}^T[\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{\alpha}] - \frac{1}{2}\left[\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{\alpha}\right]^T \boldsymbol{\Omega}\left[\boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}\boldsymbol{\alpha}\right]\right) \prod_{i=1}^{N} f_{PG}(\omega_i|n_i, 0) \tag{A.4}$$

This is amenable to a Gibbs Sampler for all coefficients, including the Polya-Gamma random variables. Goplerud (2021) notes that this also means it is amenable to closed form mean-field variational inference with no further assumptions nor integration. This result holds for any number and configuration of random effects, slopes, etc. I repeat the relevant result and algorithm (Result 1 and Algorithm 1) below. Please see that paper for full details and extensive simulations on this algorithm.

Result A.1 (Existence of CAVI (Scheme I from Goplerud 2021)). *Consider the factorization assumption:*

Scheme I: "Strong Factorization" — $\mathcal{X}_1 = q(\boldsymbol{\beta}) \prod_{j=1}^{J} q(\boldsymbol{\alpha}_j) q(\boldsymbol{\Sigma}) q(\boldsymbol{\Omega})$

For the model in Equation A.1 and for each choice of $\mathcal{X}_k$ above, each step of the CAVI algorithm can be implemented exactly in closed form, with no additional assumptions. For each $\mathcal{X}_k$, the optimal approximation for $q(\boldsymbol{\beta}, \boldsymbol{\alpha})$ is multivariate normal, $q(\boldsymbol{\Sigma})$ is the product of J independent Inverse-Wishart densities, and $q(\boldsymbol{\Omega})$ is the product of N independent Polya-Gammas.

This paper uses the following algorithm (Algorithm 1 from Goplerud 2021) for inference on the above model. Note that it immediately allows any of the traditional deep MRP models (i.e. adding many additional random intercepts and/or slopes) to be fit easily.

The remainder of this section of the Appendix outlines the various technical extensions to Goplerud (2021) in this paper: the Huang-Wand prior, SQUAREM, Parameter-Expansion, and, finally, splines.

A.1 Huang-Wand Prior

The choice of prior for a hierarchical model is a non-trivial task. Existing non-Bayesian software typically employs a flat improper prior (e.g. the Laplace approximation in `lme4`; Bates et al. 2015), although this has been shown to have some theoretical and empirical problems (e.g. Gelman 2006; Chung et al. 2015). A popular choice in the case of a single variance parameter is the half-t prior popularized by Gelman (2006) that has a tractable form. Huang and Wand (2013) generalize this by allowing it govern multivariate variance parameters (e.g. for a random slope and intercept). Their prior is shown below in both its mixture and marginal formulations (p. 441) where $\boldsymbol{\Sigma}$ is a positive definite matrix of dimensionality $p \times p$.

Algorithm A.1 CAVI from Goplerud (2021)

Set Priors of Inverse-Wishart: $\{\nu_j, \mathbf{\Phi}_j\}_{j=1}^J$; **Set Number of Iterations**: T
Initialize Variational Parameters: $\{\tilde{b}_i, \tilde{c}_i\}_{i=1}^N$ (for Polya-Gamma); $\tilde{\boldsymbol{\mu}}_\beta, \tilde{\mathbf{\Lambda}}_\beta, \tilde{\boldsymbol{\mu}}_\alpha, \tilde{\mathbf{\Lambda}}_\alpha$ (for $\boldsymbol{\beta}, \boldsymbol{\alpha}$); $\{\tilde{\nu}_j, \tilde{\mathbf{\Phi}}_j\}_{j=1}^J$ (for $\mathbf{\Sigma}_j$)
For t in $1, \cdots, T$

1. Update Polya-Gammas - $q\left(\{\omega_i\}_{i=1}^N\right)$: $\tilde{b}_i = n_i, \quad \tilde{c}_i = \sqrt{E_{q(\boldsymbol{\alpha},\boldsymbol{\beta})}\left[(\boldsymbol{x}_i^T\boldsymbol{\beta} + \boldsymbol{z}_i^T\boldsymbol{\alpha})^2\right]}$
2. Update $q(\boldsymbol{\beta}) \sim N(\tilde{\boldsymbol{\mu}}_\beta, \tilde{\mathbf{\Lambda}}_\beta)$:

$$\tilde{\mathbf{\Lambda}}_\beta = \left(\sum_{i=1}^N E_{q(\omega_i)}[\omega_i]\boldsymbol{x}_i\boldsymbol{x}_i^T\right)^{-1}, \quad \tilde{\boldsymbol{\mu}}_\beta = \tilde{\mathbf{\Lambda}}_\beta \boldsymbol{X}^T\left(\sum_{i=1}^N \left(y_i - \frac{n_i}{2}\right) - E_{q(\omega_i)}[\omega_i]\cdot \boldsymbol{z}_i^T E_{q(\boldsymbol{\alpha})}[\boldsymbol{\alpha}]\right)$$

3. Update $q\left(\boldsymbol{\alpha}_j\right) \sim N(\tilde{\boldsymbol{\mu}}_{\alpha,j}, \tilde{\mathbf{\Lambda}}_{j,\alpha})$, where $\boldsymbol{T}_j$ stacks the block diagonal expectation of the precision on the random effects ($\mathbf{\Sigma}_j^{-1}$):

$$\tilde{\mathbf{\Lambda}}_{\alpha,j} = \left(\boldsymbol{T}_j + \sum_{i=1}^N E_{q(\omega_i)}[\omega_i]\boldsymbol{z}_{i,j}\boldsymbol{z}_{i,j}^T\right)^{-1}, \quad \boldsymbol{T}_j = E_{q(\mathbf{\Sigma}_j)}\left[\boldsymbol{I}_{g_j} \otimes \mathbf{\Sigma}_j^{-1}\right]$$

$$\tilde{\boldsymbol{\mu}}_{\alpha,j} = \tilde{\mathbf{\Lambda}}_{\alpha,j}\boldsymbol{Z}_j^T\left[\sum_{i=1}^N \left(y_i - \frac{n_i}{2}\right) - E_{q(\omega_i)}[\omega_i]\cdot\left(\boldsymbol{x}_i^T E_{q(\boldsymbol{\beta})}[\boldsymbol{\beta}] + \sum_{\ell:\{1,\cdots,J\}\setminus j} \boldsymbol{z}_{i,\ell}^T E_{q(\boldsymbol{\alpha}_\ell)}[\boldsymbol{\alpha}_\ell]\right)\right]$$

4. Update $q\left(\{\mathbf{\Sigma}_j\}_{j=1}^J\right)$: $\tilde{\nu}_j = \nu_j + g_j, \quad \tilde{\mathbf{\Phi}}_j = \mathbf{\Phi}_j + \sum_{g=1}^{g_j} E_{q(\boldsymbol{\alpha}_{j,g})}\left[\boldsymbol{\alpha}_{j,g}\boldsymbol{\alpha}_{j,g}^T\right]$
5. Check for convergence, evaluate ELBO (see Goplerud 2021).

end For

$$\begin{aligned} &\mathbf{\Sigma} \sim \text{InverseWishart}\left(\nu + p - 1, \quad 2\nu \cdot \text{diag}(1/a_1, \cdots, 1/a_p)\right); \\ &a_k \sim \text{InverseGamma}(1/2, 1/A_k^2) \end{aligned} \tag{A.5a}$$

$$p(\mathbf{\Sigma}) \propto |\mathbf{\Sigma}|^{-(\nu+2p)/2} \cdot \prod_{k=1}^p \left[\nu(\mathbf{\Sigma}^{-1})_{kk} + 1/A_k^2\right]^{-(\nu+p)/2} \tag{A.5b}$$

They note some desirable properties; first, the marginal distributions on the standard deviations of $\mathbf{\Sigma}$ have the half-t formulation proposed by Gelman (2006). Further, if $\nu = 2$, then the prior implies a uniform distribution on the correlations between any two components of the prior. Second, for larger A_k, the prior can be made increasingly less informative while still remaining proper. Thus, it provides a way to stabilize the model slightly while not heavily distorting the posterior inferences.

In terms of variational inference, Huang and Wand (2013) show it can be easily put into a coordinate ascent variational framework. Specifically, if one modifies the factorization assumption in Result A.1 above to assume independence between $\mathbf{\Sigma}$ and $\{a_k\}$, the resulting posterior has an Inverse-Wishart distribution on $\mathbf{\Sigma}$ and independent Inverse-Gamma distri-

butions on $\{a_k\}$ because of the conditional conjugacy. Result A.2 presents the updates for an algorithm with this prior.

Result A.2 (CAVI with Huang-Wand Prior)**.** *Consider the model in Equation A.1 where the prior on $\boldsymbol{\Sigma}_j$ is replaced with a Huang-Wand prior (Equation A.5) with hyper-parameters ν_j and $A_{j,k}$. The augmented posterior can be expressed as: $p\left(\boldsymbol{\beta}, \boldsymbol{\alpha}, \{\omega_i\}_{i=1}^N, \{\boldsymbol{\Sigma}_j, \{a_{j,k}\}_{k=1}^{d_j}\}_{j=1}^J | \boldsymbol{y}\right)$. Consider the following proposed factorization:*

$$\mathcal{X}_{\text{HW}} = q(\boldsymbol{\beta}) \prod_{j=1}^{J} \left[\left[q(\boldsymbol{\alpha}_j)) q(\{a_{j,k}\}_{k=1}^{d_j}) \right] q(\boldsymbol{\Sigma}_j) \right] q(\boldsymbol{\Omega})$$

Each step of CAVI can be implemented in closed form with no additional assumptions. The approximating distributions on $\boldsymbol{\beta}$, $\boldsymbol{\alpha}$, $\boldsymbol{\Omega}$, $\boldsymbol{\Sigma}_j$ have the same form as in Result A.1. The factorization on $q(\{a_{j,k}\})$ is the product of independent Inverse-Gamma densities.

The algorithm can be estimated as follows: Replace Step 4 in Algorithm A.1 as follows:

- *Update $q(\{\boldsymbol{\Sigma}_j\}_{j=1}^J)$:* $\tilde{\nu}_j = \nu_j + d_j - 1 + g_j$; $\tilde{\boldsymbol{\Phi}}_j = 2\nu_j \cdot \text{diag}\left(E_{q(a_{j,k})}[1/a_{j,k}]\right) + \sum_{g=1}^{g_j} E_{q(\boldsymbol{\alpha}_{j,g})}[\boldsymbol{\alpha}_{j,g}\boldsymbol{\alpha}_{j,g}^T]$
- *Update $q(\{a_{j,k}\})$:* $q(a_{j,k}) \sim \text{InverseGamma}(\tilde{a}_{j,k}, \tilde{b}_{j,k})$

$$\tilde{a}_{j,k} = \frac{\nu_j + d_j}{2} \quad \text{and} \quad \tilde{b}_{j,k} = 1/A_{j,k}^2 + \nu_j \left[E_{q(\boldsymbol{\Sigma}_j)}[\boldsymbol{\Sigma}_j^{-1}]\right]_{k,k}$$

Given the tight coupling of $\boldsymbol{\Sigma}_j$ and $a_{j,k}$ and the relatively cheap cost of the updates, the accompanying software performs this update a handful of times at each iteration, i.e. approximating optimizing both simultaneously. In all applications in this paper, I set $\nu_j = 2$ and $A_{j,k} = 5$.

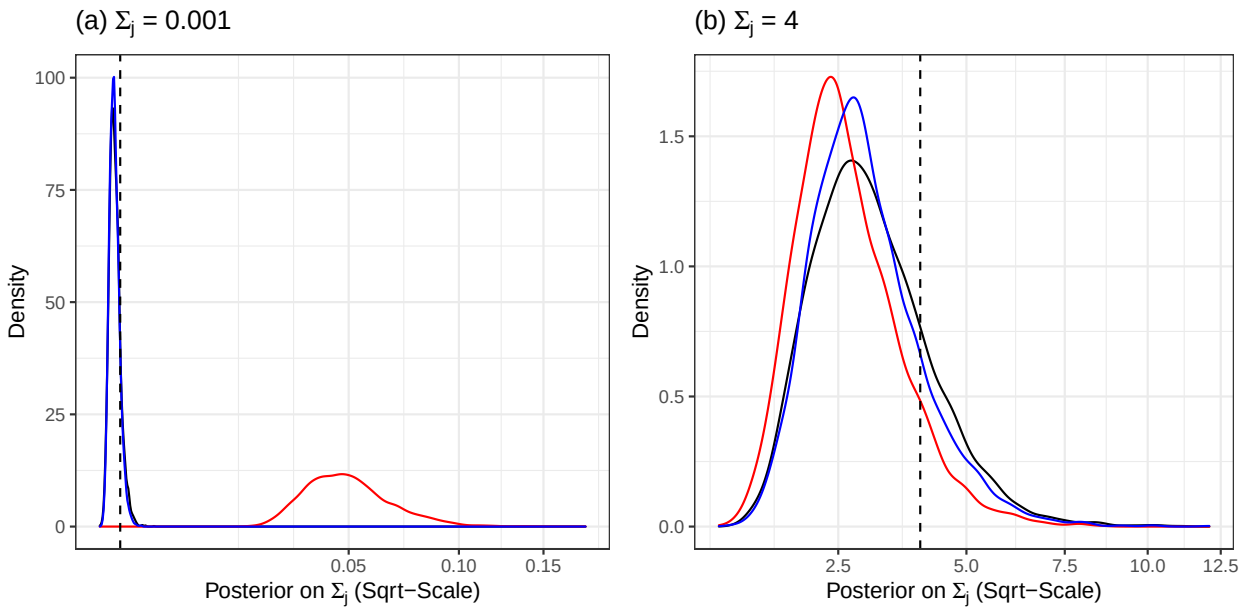
To illustrate the importance of this prior and the reasonableness of the variational approximation, consider the following stylized example: Conditional on the random effects $\boldsymbol{\alpha}_{j,g}$, does the Huang-Wand prior, its variational approximation, or the default Inverse-Wishart in Goplerud (2021) (i.e. $\nu = d + 1, \boldsymbol{\Phi} = \boldsymbol{I}$ or $a_0 = 1$, $b_0 = 1/2$ for the Inverse-Gamma) capture the true value well? To test this, I considered a scenario where $\boldsymbol{\Sigma}_j = 0.001$ (i.e. very small) and $\boldsymbol{\Sigma}_j = 4$ (i.e. large). I drew twenty samples from the corresponding normal, i.e. $g_j = 20$, and estimated the posterior or variational approximation. Figure A.1 shows the results.

The mis-calibration of the Inverse-Wishart prior is obvious when $\boldsymbol{\Sigma}_j = 0.001$, the resulting posterior is far too large—implying considerable under-shrinkage/regularization on the random effect estimates. This corroborates the concern in the main text that, especially for irrelevant random effects, the Inverse-Wishart prior will perform poorly. By contrast, the Huang-Wand prior is reasonably well-calibrated in both scenarios. The variational approximation is also rather good; it can correctly shrink irrelevant random effects by setting $\boldsymbol{\Sigma}_j$ to a small value, while closely approximating the true value for large ones.

A.2 SQUAREM

The models used in this paper employ a simple acceleration technique that maintains the monotonic convergence. Originally developed for Expectation-Maximization algorithms,

Figure A.1: Comparison of Posterior Estimates



The red line shows the posterior estimates with an Inverse-Gamma(1, 0.5) prior. The black line shows the posterior from the Huang-Wand prior; the blue line shows the variational approximation.

SQUAREM (Varadhan and Roland 2008) is a squared iterative method that proceeds as follows in the application to variational inference. Noting that one can group all of the variational parameters into a block called $\boldsymbol{\theta}$, the SQUAREM algorithm can be adapted from Table 1 in Varadhan and Roland (2008) as shown below.

Algorithm A.2 SQUAREM (from Table 1 of Varadhan and Roland 2008)

1. Begin with initial parameters $\boldsymbol{\theta}^{(0)}$
2. Perform one step of CAVI (i.e. one step from Algorithm A.1) to get $\boldsymbol{\theta}^{(1)}$.
3. Perform a second step of CAVI (i.e. one step from Algorithm A.1 using $\boldsymbol{\theta}^{(1)}$ as initial values) to get $\boldsymbol{\theta}^{(2)}$.
4. Define the following quantities: $\boldsymbol{r} = \boldsymbol{\theta}^{(1)} - \boldsymbol{\theta}^{(0)}; \boldsymbol{v} = (\boldsymbol{\theta}^{(2)} - \boldsymbol{\theta}^{(1)}) - \boldsymbol{r}$
5. Calculate the step-length for SQUAREM α using the following formula:

$$\alpha = \min\left(-||\boldsymbol{r}||_2/||\boldsymbol{v}||_2, -1\right)$$

6. Propose a new $\boldsymbol{\theta}^*$ such that $\boldsymbol{\theta}^* = \boldsymbol{\theta}^0 - 2\alpha\boldsymbol{r} + \alpha^2\boldsymbol{v}$
7. Evaluate whether the new $\boldsymbol{\theta}^*$ increases the objective (ELBO). If it does, set $\boldsymbol{\theta}^*$ as the new parameter estimates. If not, propose a new $\alpha \leftarrow (\alpha - 1)/2$ and try again. Note that if $\alpha = -1$, then $\boldsymbol{\theta}^* = \boldsymbol{\theta}^{(2)}$.

The above algorithm has many desirable properties; first, the final backtracking step (Step 7) ensures that it cannot decrease the objective and thus maintains monotonic convergence of the variational algorithm. Second, it is rather inexpensive to compute—it only requires

the evaluation of the objective function at the proposed parameter vector $\boldsymbol{\theta}^*$ after performing three steps of the algorithm. Thus, the implementation of SQUAREM attempts to accelerate the model every three steps. In practice, it usually succeeds after no more than a few backtracking steps.

One additional modification is required for the above application: Given that many of the variational parameters are bounded (e.g. positive or symmetric matrices), I transform all parameters to live on an unbounded scale before applying SQUAREM. This ensures that all proposed $\boldsymbol{\theta}^*$ are valid regardless of the choice of α. For positive-constrained parameters, I take the log; for matrices, I take either the Cholesky decomposition or the LU decomposition and perform element-by-element SQUAREM where elements that are constrained to be positive are logged.

A.3 Parameter-Expansion

Another way to improve the speed of estimation is parameter expansion. It proceeds as follows: Note that Equation A.1 assumes a mean-zero prior on the random effect $\boldsymbol{\alpha}_{j,g}$: $\boldsymbol{\alpha}_{j,g} \sim N(\mathbf{0}, \boldsymbol{\Sigma}_j)$. Goplerud (2021) provides the following definition of a parameter expansion below where the random effects are linearly transformed, and all other parameters adjusted, such that the log-posterior remains unchanged.

Definition A.1 (Expansions for Hierarchical Models)**.** *Define a set of expansion parameters $\boldsymbol{\xi}$ that consists, for each j, of a mean shift $\boldsymbol{\mu}_j \in \mathbb{R}^{d_j}$ and a scale shift $\boldsymbol{R}_j \in \mathbb{R}^{d_j \times d_j}$ such that $\boldsymbol{R}_j$ is invertible. I use superscript ^{X} to denote the "expanded" parameters. The mapping between $\boldsymbol{\theta}^X$ and $\boldsymbol{\theta}$ for a fixed $\boldsymbol{\xi}$ is denoted as $t_{\boldsymbol{\xi}}(\boldsymbol{\theta}^X)$ and listed below. $\boldsymbol{M}_j$ is a $p \times d_j$ matrix such that $[\boldsymbol{M}_j]_{a,b} = 1$ if the covariate corresponding to $[\boldsymbol{z}_{i,j}]_b$ is the same as the covariate for $[\boldsymbol{x}_i]_a$. All other elements of $\boldsymbol{M}_j$ are zero. For simplicity, assume that each element of $\boldsymbol{z}_i$ corresponds to some variable in $\boldsymbol{x}_i$, i.e. that each column of $\boldsymbol{M}_j$ has exactly one non-zero element.*

$$\left[\boldsymbol{\beta}, \boldsymbol{\alpha}, \{\boldsymbol{\Sigma}_j\}_{j=1}^J, \boldsymbol{\Omega}\right] = t_{\boldsymbol{\xi}}([\boldsymbol{\beta}^X, \boldsymbol{\alpha}^X, \{\boldsymbol{\Sigma}_j^X\}_{j=1}^J, \boldsymbol{\Omega}^X]) = \begin{cases} \boldsymbol{\beta} = \boldsymbol{\beta}^X + \sum_{j=1}^J \boldsymbol{M}_j \boldsymbol{R}_j \boldsymbol{\mu}_j \\ \boldsymbol{\alpha}_{j,g} = \boldsymbol{R}_j \left(\boldsymbol{\alpha}_{j,g}^X - \boldsymbol{\mu}_j\right) \\ \boldsymbol{\Sigma}_j = \boldsymbol{R}_j \boldsymbol{\Sigma}_j^X \boldsymbol{R}_j^T \\ \boldsymbol{\Omega} = \boldsymbol{\Omega}^X \end{cases}$$

The augmented model is listed below for an important special case treated in detail ("Mean Expansion") in the empirical analysis. The full expansion ("Translation Expansion") is also listed.

- *Mean Expansion: Assume all $\boldsymbol{R}_j = \boldsymbol{I}_{d_j}$.*

$$\ln p(y_i | \omega_i, \boldsymbol{\beta}^X, \boldsymbol{\alpha}^X) \propto \boldsymbol{s}^T[\boldsymbol{X}\boldsymbol{\beta}^X + \boldsymbol{Z}\boldsymbol{\alpha}^X] - 1/2[\boldsymbol{X}\boldsymbol{\beta}^X + \boldsymbol{Z}\boldsymbol{\alpha}^X]^T \boldsymbol{\Omega} [\boldsymbol{X}\boldsymbol{\beta}^X + \boldsymbol{Z}\boldsymbol{\alpha}^X]$$
$$p(\boldsymbol{\beta}^X) \propto 1, \quad \boldsymbol{\alpha}_{j,g}^X | \boldsymbol{\Sigma}_j^X, \sim N\left(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j^X\right), \quad p(\boldsymbol{\Sigma}_j^X) \sim IW(\nu_j, \boldsymbol{\Phi}_j)$$

- *Translation Expansion:*

$$\ln p(y_i|\omega_i, \boldsymbol{\beta}^X, \boldsymbol{\alpha}^X) \propto \boldsymbol{s}^T[\boldsymbol{X}\boldsymbol{\beta}^X + \boldsymbol{Z}\boldsymbol{R}\boldsymbol{\alpha}^X] - 1/2[\boldsymbol{X}\boldsymbol{\beta}^X + \boldsymbol{Z}\boldsymbol{R}\boldsymbol{\alpha}^X]^T\boldsymbol{\Omega}[\boldsymbol{X}\boldsymbol{\beta}^X + \boldsymbol{Z}\boldsymbol{R}\boldsymbol{\alpha}^X]$$
$$\boldsymbol{R} = \text{blockdiag}\left(\{\boldsymbol{I}_{g_j} \otimes \boldsymbol{R}_j\}_{j=1}^J\right), \quad p(\boldsymbol{\beta}^X) \propto 1, \quad \boldsymbol{\alpha}_{j,g}^X|\boldsymbol{\Sigma}_j^X \sim N\left(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j^X\right)$$
$$p(\boldsymbol{\Sigma}_j^X) \sim IW(\nu_j, \boldsymbol{R}_j^{-1}\boldsymbol{\Phi}_j\boldsymbol{R}_j^{-T})$$

With this definition, Goplerud (2021) applied a procedure known as PX-VB (Parameter-Expanded Variational Bayes; Jaakkola and Qi 2007) to improve convergence. Their restatement of the result from Jaakkola and Qi (2007) is provided below:

Lemma A.1 (Parameter Expanded Variational Bayes - Jaakkola and Qi 2007)**.** *Given some factorization assumption $\mathcal{X}$, the following procedure converges no slower than the associated CAVI algorithm and maintains a monotonic improvement of the* ELBO*.*

1. *Perform one step of CAVI (e.g. Algorithm A.1, Steps 1-4) giving $q(\boldsymbol{\theta})$ and $\text{ELBO}_{q(\boldsymbol{\theta})}$.*
2. *Noting $q(\boldsymbol{\theta}) \sim^d q(\boldsymbol{\theta}^X)$ when $\boldsymbol{\xi} = \boldsymbol{\xi}_{\text{Null}}$ and thus $\text{ELBO}_{q(\boldsymbol{\theta})}^{X-\boldsymbol{\xi}_{\text{Null}}} = \text{ELBO}_{q(\boldsymbol{\theta})}$, maximize the $\text{ELBO}_{q(\boldsymbol{\theta})}^{X-\boldsymbol{\xi}}$ over $\boldsymbol{\xi}$.*

$$\hat{\boldsymbol{\xi}} = \arg\max_{\boldsymbol{\xi}} \text{ELBO}_{q(\boldsymbol{\theta})}^{X-\boldsymbol{\xi}} = \arg\max_{\boldsymbol{\xi}} E_{q(\boldsymbol{\theta})}[\ln p^X(\boldsymbol{y}, \boldsymbol{\theta}|\boldsymbol{\xi})] - E_{q(\boldsymbol{\theta})}[\ln q(\boldsymbol{\theta})]$$

   Note that $\text{ELBO}_{q(\boldsymbol{\theta})}^{X-\hat{\boldsymbol{\xi}}} \geq \text{ELBO}_{q(\boldsymbol{\theta})}$.

3. *Apply the reduction function to recover a distribution on the original, non-expanded space. Equivalently, transform $q(\boldsymbol{\theta})$ by applying a change-of-variables using $t_{\hat{\boldsymbol{\xi}}}(\boldsymbol{\theta})$.*

$$q'(\boldsymbol{\theta}) = \int t_{\hat{\boldsymbol{\xi}}}(\boldsymbol{\theta}) q(\boldsymbol{\theta}) d\boldsymbol{\theta}$$

   Note that $\text{ELBO}_{q'(\boldsymbol{\theta})} = \text{ELBO}_{q(\boldsymbol{\theta})}^{X-\hat{\boldsymbol{\xi}}}$ and $\text{ELBO}_{q'(\boldsymbol{\theta})} \geq \text{ELBO}_{q(\boldsymbol{\theta})}$.

Using this result, Goplerud (2021) assumes that there is only a mean-expansion, i.e. $\boldsymbol{\alpha}_{j,g}^X$ has some non-zero mean, and derives a closed-form update for the parameter expansion step. This involves re-centered each random effect to be mean-zero and adjusting the corresponding fixed effects to keep the expected linear predictor constant.

This paper extends this work further by analyzing a PX-VB method for the translation expansion where $\boldsymbol{\alpha}_{j,g}^X$ is not only given a non-zero mean but is multiplied by some matrix $\boldsymbol{R}_j$. Assume that the random effects have been adjusted to be mean zero and thus the only expansion term is $\boldsymbol{R}_J$. Examining the objective (ELBO) and collecting terms—given the factorization (Scheme I from Goplerud (2021)) assumed in this paper–results in the following objective. I use $\boldsymbol{\rho}$ to denote the stacked vectorized $\boldsymbol{R}_j$ matrices: $\boldsymbol{\rho}^T = \left[\text{vec}(\boldsymbol{R}_1)^T, \cdots \text{vec}(\boldsymbol{R}_J)^T\right]$. To build a more tractable algorithm, I also update the mean parameter of the variational distribution on $\boldsymbol{\beta}$ ($\tilde{\boldsymbol{\mu}}_\beta$) simultaneously to the expansion parameters.

$$
\begin{aligned}
\text{ELBO}_{q(\boldsymbol{\theta})}^{X-\boldsymbol{\xi}} \propto \quad & \boldsymbol{s}^T\left(\boldsymbol{X}\tilde{\boldsymbol{\mu}}_\beta + \boldsymbol{B}\boldsymbol{\rho}\right) - \frac{1}{2}\left(\boldsymbol{X}\tilde{\boldsymbol{\mu}}_\beta + \boldsymbol{B}\boldsymbol{\rho}\right)^T E_{q(\boldsymbol{\Omega})}[\boldsymbol{\Omega}]\left(\boldsymbol{X}\tilde{\boldsymbol{\mu}}_\beta + \boldsymbol{B}\boldsymbol{\rho}\right) + \\
& - \frac{1}{2}\boldsymbol{\rho}^T\left(\sum_{i=1}^{N} E_{q(\omega_i)}[\omega_i]\boldsymbol{K}_i\boldsymbol{K}_i^T\right)\boldsymbol{\rho} + \\
& \sum_{j=1}^{J} -\nu_j \cdot \ln|\boldsymbol{R}_j| - \frac{1}{2}\text{tr}\left(\boldsymbol{R}_j^{-1}\boldsymbol{\Phi}_j\left[\boldsymbol{R}_j^{-1}\right]^T E_{q(\boldsymbol{\Sigma}_j)}[\boldsymbol{\Sigma}_j^{-1}]\right) \\
& \text{where} \quad \boldsymbol{b}_{ij} = E_{q(\boldsymbol{\alpha}_{j,g[i]})}[\boldsymbol{\alpha}_{j,g[i]}] \otimes \left[\boldsymbol{z}_{i,j}^b\right]^T \quad \boldsymbol{b}_i^T = [\boldsymbol{b}_{i1}^T, \cdots, \boldsymbol{b}_{iJ}^T] \quad \boldsymbol{B} = \begin{bmatrix} \boldsymbol{b}_1^T \\ \cdots \\ \boldsymbol{b}_N^T \end{bmatrix} \\
& \boldsymbol{L}_{ij}^T\boldsymbol{L}_{ij} = \text{Var}(\boldsymbol{\alpha}_{j,g[i]}); \quad \boldsymbol{k}_{ij} = \left(\boldsymbol{L}_{ij} \otimes [\boldsymbol{z}_{ij}^b]^T\right) \quad \boldsymbol{K}_i^T = [\boldsymbol{k}_{i1}^T, \cdots, \boldsymbol{k}_{iJ}^T]
\end{aligned}
\tag{A.6}
$$

Note that the third line ($-\nu_j \ln|\boldsymbol{R}_j| \cdots$) represents the contribution of the prior; if this did not exist, then there is a least-squared closed-form update for the expansion parameters $\boldsymbol{\rho}$. However, given the final line, for most choices of prior, this is intractable. Thus, I rely on a "one-step-late" idea for parameter expansion; Van Dyk and Tang (2003) apply a similar logic in the parameter-expanded EM case. The idea is to take the gradient of the above but evaluate the gradient with respect to the intractable prior term as its *null* value, i.e. $\boldsymbol{R}_j = 1$. By setting the modified gradient equal to zero, one obtains the following update for $(\boldsymbol{\rho}, \tilde{\boldsymbol{\mu}}_\beta)$ where $\mathbf{0}_p$ is a vector of zeros with the dimensionality of $\boldsymbol{\beta}$.

$$
\begin{aligned}
\hat{\boldsymbol{\rho}}, \hat{\tilde{\boldsymbol{\mu}}}_\beta \triangleq \quad & \mathbf{0} = [\boldsymbol{X}\boldsymbol{B}]^T\boldsymbol{s} - \\
& \left[[\boldsymbol{X}\boldsymbol{B}]^T E_{q(\boldsymbol{\Omega})}[\boldsymbol{\Omega}][\boldsymbol{X}\boldsymbol{B}] + \begin{pmatrix} \mathbf{0}_{p\times p} & \mathbf{0}_{p\times\sum_j d_j^2} \\ \mathbf{0}_{\sum_j d_j^2 \times p} & \sum_{i=1}^N E_{q(\omega_i)}[\omega_i]\boldsymbol{K}_i\boldsymbol{K}_i^T \end{pmatrix}\right]\begin{pmatrix} \tilde{\boldsymbol{\mu}}_\beta \\ \boldsymbol{\rho} \end{pmatrix} + \\
& \left[\mathbf{0}_p^T, \quad \text{vec}\left(-\nu_j\boldsymbol{I} + E_{q(\boldsymbol{\Sigma}_j)}[\boldsymbol{\Sigma}_j^{-1}]\boldsymbol{\Phi}_j\right)^T\right]^T
\end{aligned}
\tag{A.7}
$$

However, because of the one-step-late approximation, it is not guaranteed that these proposed updates $(\hat{\boldsymbol{\rho}}, \hat{\tilde{\boldsymbol{\mu}}}_\beta)$ will improve the objective. Thus, to ensure monotonic convergence, if it does not increase the objective, the software performs a few steps of numerical optimization (e.g. L-BFGS-B). Given availability of an analytic gradient and the modest size of the problem (the size of $\tilde{\boldsymbol{\mu}}_j$ plus $\sum_j d_j^2$), this is fairly inexpensive. In the case of a Huang-Wand prior, better performance is obtained by profiling out the $\tilde{b}_{j,k}$ parameters and performing parameter expansion on this objective.

A.4 Splines

Consider the case of a single spline on a covariate x_i such as presidential two-party vote share. I follow Ruppert, Wand and Carroll (2003)'s presentation of splines as a hierarchical

model as shown below. I present the simplest case of truncated linear functions as deviations from a globally linear trend; other extensions (e.g. to penalized B-splines; Eilers and Marx 1996) are straightforward.

The accompanying software follows the common strategy of $\max(N_x/4, 35)$ knots where N_x is the number of unique values of $\{x_i\}$ and the knots placed at equally spaced quantiles of the distribution of $\{x_i\}$ (e.g., Ruppert, Wand and Carroll 2003); user-defined choices are possible.

$$
\begin{aligned}
&y_i \sim \text{Bern}(p_i); \quad p_i = \frac{\exp(\psi_i)}{1+\exp(\psi_i)}; \quad \psi_i = \beta_0 + \beta_1 x_i + \sum_{k=1}^{K} \gamma_k \left(x_i - \kappa_k\right)^+; \\
&\gamma_k \sim N(0, \sigma_\gamma^2); \quad p(\beta_0, \beta_1) \propto 1; \quad \sigma_\gamma^2 \sim p_0(\sigma_\gamma^2); \quad (x_i - \kappa_k)^+ = \begin{cases} 0 & \text{if} \quad x_i < \kappa_k \\ x_i - \kappa_k & \text{if} \quad x_i \geq \kappa_k \end{cases}
\end{aligned}
\tag{A.8}
$$

Note that as $\sigma_\gamma^2 \to 0$, i.e. the estimated random effect variance declines, the model collapses to one with a simple linear effect on the predictor x_i. As σ_γ^2 becomes large, the estimated effect becomes increasingly "wiggly". As in a normal random effect model, the data (and prior) thus determines the smoothness of the effect on x_i. Given that this model has one additional hierarchical term, it fits into the general design framework noted above (Equation A.1). If a spline on a second variable w_i were desired, it would be governed by a different variance parameter (e.g., $\sigma_{\gamma'}^2$) and enter as a second additional hierarchical term.

One important extension is to allow "factor-by-curve" splines, i.e. interactions between a spline and a categorical factor (Ruppert, Wand and Carroll 2003). This would allow, for example, the effect of presidential vote share to vary by education in a smooth fashion. In the most common formulation (again see Ruppert, Wand and Carroll 2003), the linear "fixed" component of the spline is not regularized and thus a baseline category is omitted. As that may over-fit, I present a slightly modified version where (i) the linear component is regularized and (ii) the smooth components share a variance component. The linear predictor for the binomial model is shown below. I focus on a single continuous covariate x_i and some factor j that has g_j levels where $z_i \in \{1, \cdots, g_j\}$ denotes the value for observation i.

$$
\begin{aligned}
&\psi_i = \beta_0 + \beta_1 x_i + \sum_{k=1}^{K} \left[\gamma_{k,\text{Global}} (x_i - \kappa_k)^+\right] + \sum_{g=1}^{G_j} I(z_i = g) \left[(\alpha_{0,g} + \alpha_{1,g} x_i) + \sum_{k=1}^{K} \gamma_{k,g} \left(x_i - \kappa_k\right)^+\right]; \\
&\gamma_k \sim N(0, \sigma_{\gamma,\text{Global}}^2); \quad [\alpha_{0,g}, \alpha_{1,g}]^T \sim N(\mathbf{0}, \mathbf{\Sigma}_\alpha); \quad \gamma_{k,g} \sim N(0, \sigma_\gamma^2) \\
&p(\beta_0, \beta_1) \propto 1; \quad \sigma_{\gamma,\text{Global}}^2 \sim p_0(\sigma_{\gamma,\text{Global}}^2); \quad \mathbf{\Sigma}_\alpha \sim p_0(\mathbf{\Sigma}_\alpha); \quad \sigma_\gamma^2 \sim p_0(\sigma_\gamma^2)
\end{aligned}
\tag{A.9}
$$

The model consists, therefore, of (a) a linear effect on x_i, (b) deviations from this linear effect by group g as a random slope/intercept combination, (c) a smooth effect on x_i, and (d) deviations from this smooth effect by group g. This formulation scales nicely to the case where the same continuous covariate (e.g. two-party vote share) has factor-by-curve for

multiple factors (e.g. by education and ethnicity). The implementation is shown below.

- Add a random slope for the continuous variable x_i to each of the grouping factors, e.g. `(1 + x | g) + (1 + x | g2)`, etc.
- Add a global spline term corresponding to $\{\gamma_{k,\text{Global}}\}_{k=1}^{K}$. This adds one additional random intercept.
- Add derivations from the global spline term for each group for each of the interacting factors. This adds an additional random intercept for each grouping factor.

Thus, it is clear that this can be fit into the standard architecture of Goplerud (2021) for variational inference in logistic binomial models. The results are similar to existing work on variational inference for splines. In this paper, I assume independence between each of the spline-based random effects. I save for future explorations more detailed factorization schemes that, for example, allow dependencies between the deviations of the spline terms.

A.5 Importance of Acceleration Techniques

This section briefly illustrates the key need for acceleration techniques when the Huang-Wand (i.e. half-t) prior is employed. The main issue when using this prior is that the model may require many more iterations to reach convergence. While there is some additional cost to estimating the Huang-Wand prior, the major concern is that estimation with this prior can often require hundreds of additional iterations to converge.

Thus, techniques that reduce the number of iterations considerably can be highly desirable—even if they increase the cost per iteration. As the discussion above notes, the cost of using SQUAREM is dominated by a handful of extra evaluations of the ELBO and a small number of matrix decompositions to extrapolate parameters on an unbounded scale; this is relatively inexpensive overall. The parameter expansion can be somewhat more expensive; however, note that it involves solving a least squares system that is only a function of the size of the fixed effects and the *dimension* d_j of the random effects. As this is relatively small (as only g_j, i.e. the number of levels, is often large), it can be done relatively inexpensively.

Table A.1 illustrates this quantitatively; first, the left panel shows the number of iterations required for convergence across the simulations used in Appendix B. We see that the number of iterations is considerably larger—often by a multiple of 5-to-10—when comparing Inverse-Wishart ("IW") versus the Huang-Wand ("HW") prior. This has a corresponding effect on run-time (the right panel); while the Inverse-Wishart model can be estimated quickly in around 15-20 seconds, the Huang-Wand prior takes considerably longer (250-300 seconds). Yet, after applying acceleration techniques, the number of iterations can be reduced considerably—actually smaller than the original (non-accelerated) Inverse-Wishart model—with a run time that is comparable to the original choice of prior.

B Simulations for Deep Hierarchical Models

To illustrate the importance of deep MRP, this section provides some simulations on synthetic data; they are designed to show the impact of ignoring important interactions or non-linear

Table A.1: Comparing the Impact of Acceleration Techniques

(a) Number of Iterations

N	IW	HW	Accelerated HW
250	228	1464.68	21.68
500	262.08	1652.74	23.11
1000	183.06	1392.23	22.22
2000	106.85	1133.96	23.11

(b) Estimation Time (seconds)

N	IW	HW	Accelerated HW
250	17.27	266.13	7.79
500	22.72	319.8	8.69
1000	19.26	294.73	8.94
2000	14.71	276.27	10.57

Note: "IW" stands for Inverse-Wishart. "HW" stands for Huang-Wand. "Accelerated HW" uses the two acceleration techniques discussed in the main text. See Appendix B for more details.

effects for the "prediction" part of MRP (i.e. predictive accuracy on held-out data similar to the original survey). Equation A.10 shows the data generating process; each model includes three continuous covariates with (possibly) non-linear functional forms[1] as well as two random effects (one with 5 levels and one with 50 levels) and their interaction. All observations are assigned to random effect groups fully at random.

$$x_{i,0} \sim \text{Unif}(0,1); \quad x_{i,1} \sim N(0,1); \quad \Pr(x_{i,2} = k) = 0.01 \quad \forall k \in \{0, \cdots, 1\} \tag{A.10a}$$

$$\begin{aligned} f_0(x_0) &= \frac{1}{3} \cdot (\exp(2x_0) - 3.75887)\,; \quad f_1(x_1) = \frac{1}{2}x_1; \\ f_2(x_2) &= 1/4 \cdot (0.2 \cdot {x_2}^{11} \cdot (10 \cdot (1 - x_2))^6 + 10 \cdot (10 \cdot x_2)^3 \cdot (1 - x_2)^{10}) - 1 \end{aligned} \tag{A.10b}$$

$$\alpha_g \sim N(0,1); \quad \alpha_{g'} \sim N(0,1) \tag{A.10c}$$

$$\alpha_{g,g'} = 0 \cdot \gamma_{g,g'} + \nu_{g,g'}(1 - \gamma_{g,g'}); \quad \gamma_{g,g'} \sim \text{Bern}(p_{\text{inter}}); \quad \nu_{g,g'} \sim N(0,4) \tag{A.10d}$$

$$\psi_i = f_0(x_{i,0}) + f_1(x_{i,1}) + f_2(x_{i,2}) + \alpha_{g[i]} + \alpha_{g'[i]} + \alpha_{g[i],g'[i]} \tag{A.10e}$$

$$y_i \sim \text{Bern}(p_i); \quad p_i = \frac{\exp(\psi_i)}{1 + \exp(\psi_i)} \tag{A.10f}$$

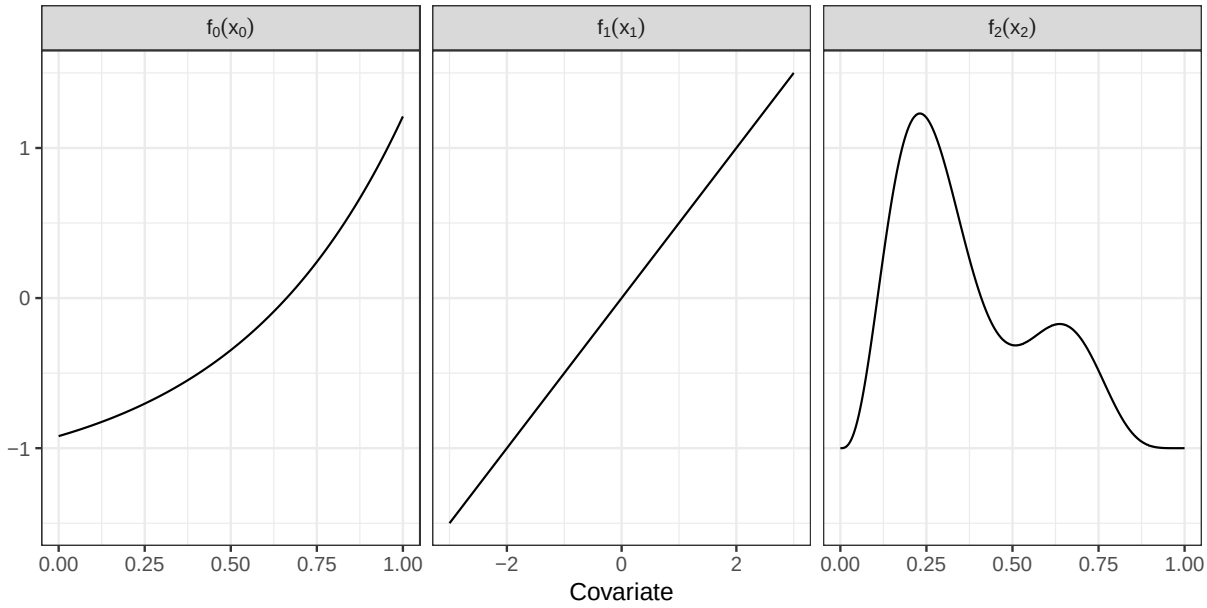
The effects of three continuous predictors, f_0, f_1, f_2, are shown below across the range of plausible values.

The contribution of the interactive random effect, indexed by $\alpha_{g,g'}$ for level g of the first factor and level g' of the second factor, is designed to be large to highlight the important of missing a possibly crucial interactive effect. The one difference from a typical simulation environment is that there is some probability p_{inter} that the random effect $\alpha_{g,g'}$ is zero. This allows to capture scenarios where the interaction is critically important (p_{inter} is small) or whether it is more irrelevant (p_{inter} is large) as only a small proportion of observations have some non-zero effect. In my analysis, I explore two relatively extreme cases where $p_{\text{inter}} = 0.25$ (most interactions matter) and $p_{\text{inter}} = 0.99$ where most interactions are irrelevant.

The prior expectation is that models that are not able to estimate interactions (e.g. simple MRP) should do considerably worse for $p_{\text{inter}} = 0.25$ whereas ignoring the interactions should impose little cost in terms of predictive performance for $p_{\text{inter}} = 0.99$.

[1] These are adapted from `mgcv`'s documentation `gamSim`.

Figure A.2: Effects of Continuous Predictors



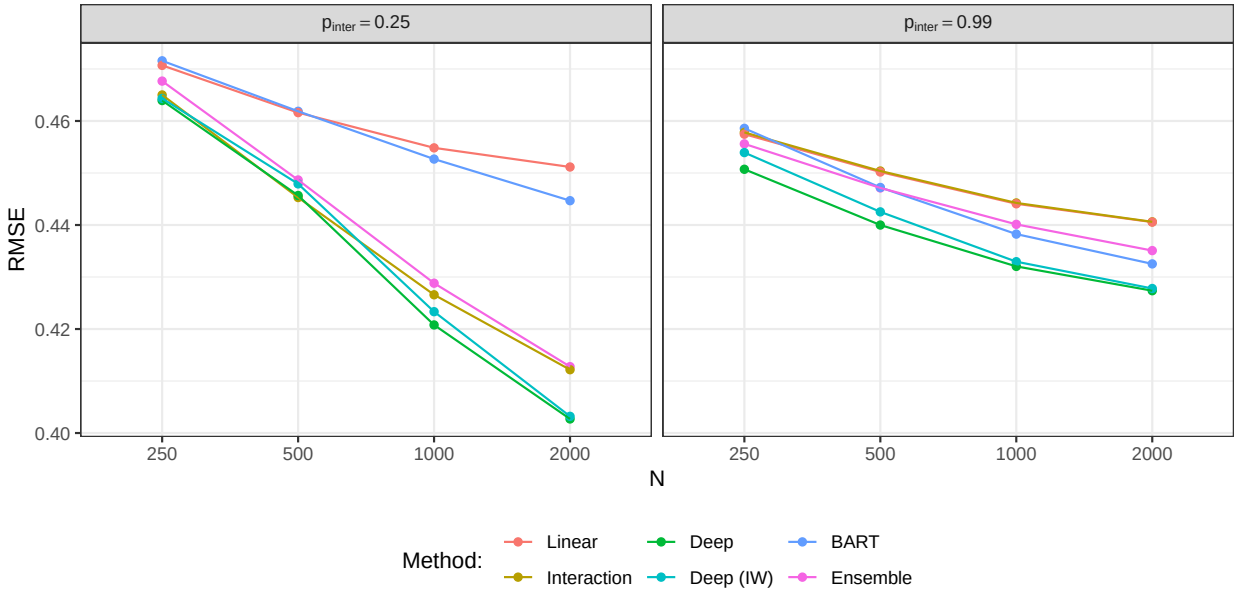
This simulation, therefore, is designed to capture the features that deep hierarchical models are designed to address and allows for a comparison where simple hierarchical models are expected to do poorly. It also allows us to compare this against popular alternative machine learning approaches to see how they compare.

I generate datasets of size $N \in \{250, 500, 100, 2000\}$ and a corresponding test dataset of an identical size. The key quantity the simulations consider is the out-of-sample predictive accuracy on the test data. I consider six models, outlined below.

- Linear: This model includes only the two primary random effects, i.e. $\psi_i = \beta_{\text{Int}} + \beta_0 x_{i,0} + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \alpha_{g[i]} + \alpha_{g'[i]}$. It is fit using `lme4`.
- Interaction: This model adds the interactive random effect, i.e. $\alpha_{g[i],g'[i]}$ to the above model. It is fit using `lme4`.
- Deep: This model adds splines for each continuous predictor to the model. It is fit using the variational approach.
- Deep (IW): This model uses an Inverse-Wishart prior instead of the Huang-Wand prior.
- BART: This model is fit using `bartMachine` on the default settings.
- Ensemble: This includes an ensemble of a random forest, BART, and LASSO (with interactions). It **does not** include any hierarchical models.

Figure A.3 presents the root mean-square error (RMSE) on the out of sample predictions across each level of sparsity and sample size. The results are averaged across 1,000 simulations.

Figure A.3: Performance of Models on Simulated Data



Consider first the panel where interactions are highly important $p_{\text{inter}} = 0.25$. In this case, a simple hierarchical model that includes neither interactions nor splines does very poorly compared to other models. Adding the interactions improves performance considerably, and a model that includes both splines and interactions performs the best. Note that, in this case, BART performs about the same as the simple hierarchical model for small sample sizes, but its advantage grows as it is given more data.

Consider now a case where the interactions are mostly irrelevant ($p_{\text{inter}} = 0.99$). In this case, simple hierarchical models and the hierarchical model with interactions do about the same as there is little cost to ignoring the (rare) interactions. By contrast, BART does noticeably better in this setting as it is able to capture the non-linearity of the continuous predictor effectively. At the smallest sample size, however, all three methods perform about the same. It is worth noting that the hierarchical model with interactions and splines again does the best insofar as it can effectively capture the smooth non-linear functional forms using splines even at small sample sizes. Even though the model includes interactions that are irrelevant, this does not materially harm its performance as they are likely aggressively regularized. Indeed, experimentation with a clearly "overfit" hierarchical model (i.e. allowing the splines to vary across the categorical factors—a feature that is *not* present in the data generating process) does about the same suggesting that, in many cases, the regularization in hierarchical models is reasonably robust against preventing severe overfitting even when the model is simpler than the exact one specified.

Finally, we note that the use of the Inverse-Wishart prior ("Deep (IW)") in Goplerud (2021) results in slightly worse performance across both simulation settings and most sample sizes. This agrees qualitatively with the results in Appendix D that shows the "Deep (IW)" model performs worse for small sample sizes in both performance on held-out data and error for MRP.

I also conducted an analysis of ensembles using this synthetic formulation. I find that of an ensemble that includes LASSO, random forest, BART, and the hierarchical methods, the same story found in the main paper holds—the deep methods are given increasing weight as N increases and are the dominant method for large sample sizes. Yet, as Figure A.3 shows, an ensemble that **excludes** hierarchical models performs considerably worse than the deep hierarchical model on its own.

C Estimation Time

This section provides a direct comparison of the estimation times of the algorithms considered across the various simulations in this paper. Table A.2 reports the estimation time for three settings (purely synthetic simulations [Appendix B], ensemble [Section 3], and BART [Section 4]). It shows that, in all cases, deep MRP estimated with variational inference is highly competitive in terms of time to estimation. All of the models are estimated using a single core and 8 GB of memory. All times are the average time in seconds, averaged across all models estimated for a particular sample size.

Across a variety of sample sizes and different model complexities, we see that deep MRP (i.e. including many interactions) is highly competitive in terms of estimation time with other ML methods. When comparing against BART (Panel C), we see that even the most complex variational model is within 15 seconds of BART on average. For the largest data sizes (e.g. 10,000), we see that the variational methods are actually faster than BART.

Look beyond BART, the ensemble analysis (Panel B) shows that the variational methods are considerably faster than other standard machine learning techniques (e.g. random forests, K-nearest neighbors, etc). The reason for this speed gain is that those models require tuning of the hyper-parameters (Ornstein 2020) such as the number of variables included in each tree for a random forest. By contrast, hierarchical models do not require explicit tuning of the amount of regularization as it is estimated internally in the model. This provides considerable savings in terms of cost of estimation.

In the synthetic examples, I also compare the variational model against the same model (i.e. three splines and interaction random effects) estimated using `mgcv` (via `gamm4`). It is considerably faster, especially as the sample size grows.

D Additional Empirical Results: Ensemble

This section outlines additional empirical results for the ensemble analysis in Section 3. The only substantial modification from Ornstein (2020) is adjusting the replication archive where a linear hierarchical model is used in the ensemble instead of a logistic hierarchical model. The Supplemental Information on the Dataverse contains a replication of the synthetic simulations in Ornstein (2020); **To Reviewers: This file should be attached as "Dataverse_Supplement.PDF".**.

I briefly describe how the ensemble was created using the standard "stacking" or "SuperLearning" procedure—see Ornstein (2020) for a detailed discussion. I split the data into $K = 5$ folds. For each fold k, the model is estimated using the other folds and an out-of-sample prediction is obtained for each observation in fold k. Thus, out-of-sample predictions

Table A.2: Timing of Methods

(a) Synthetic

N	Simple	Inter	Deep	BART	GAMM4
250	0.25	0.58	7.79	2.32	10.38
500	0.29	0.84	8.69	2.75	20.18
1000	0.45	1.32	8.94	5.1	36.72
2000	0.77	2.15	10.57	10.25	62.67

(b) Ensemble (Section 3 in Main Text)

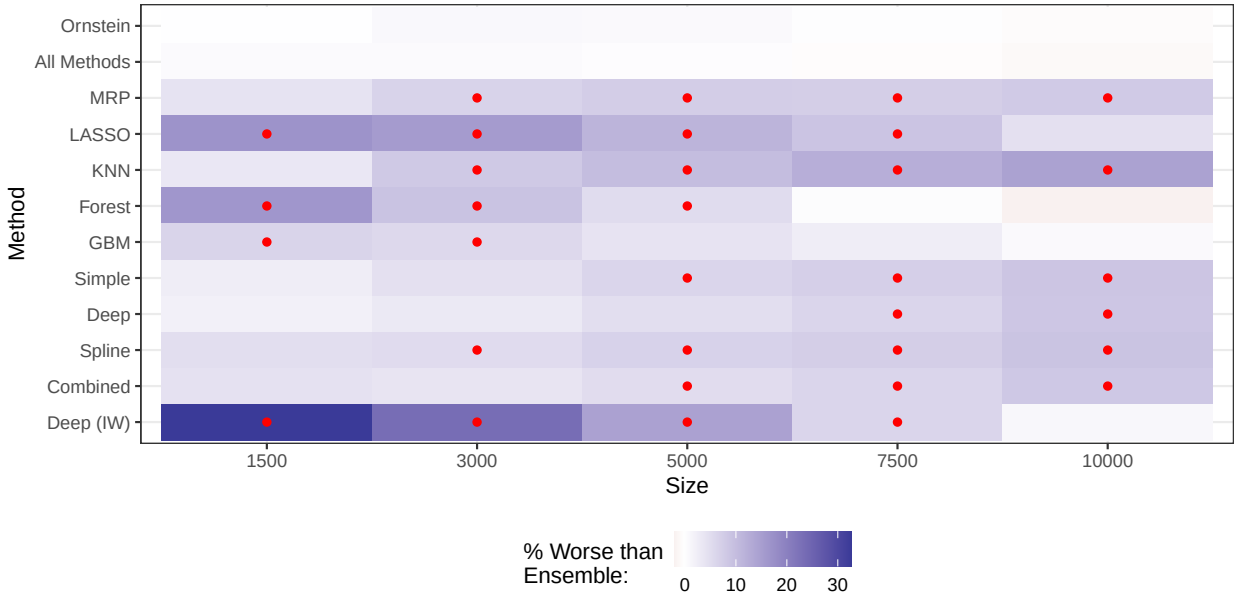
N	Methods in Ornstein (2020)					Variational Methods for MRP				
	MRP	LASSO	KNN	Forest	GBM	Simple	Deep	Spline	Combined	Deep (IW)
1500	1.84	1.03	17.31	24.05	69.49	4.04	18.21	8.38	26.17	8.05
3000	2.91	1.33	42.25	73.77	147.26	3.65	18.38	8.02	26.15	7.75
5000	4.18	1.74	85.75	131.87	266.43	3.67	20.36	8.33	28.47	7.98
7500	7.35	2.39	140.49	204.9	422.49	4.25	25.12	10.1	35.02	9.59
10000	10.88	3.42	212.74	280.51	605.98	4.8	29.04	11.7	40.39	11.21

(c) BART (Section 4 in Main Text)

N	Linear	Inter	Spline	Combined	BART
1500	4.47	20.41	9.12	28.14	14.69
3000	4.39	23.15	9.61	31.6	21.66
4500	4.64	27.81	10.59	36.85	30.48
6000	4.83	31.06	11.4	40.69	40.57
7500	4.98	34.03	12.03	44.67	49
10000	5.12	38.49	12.87	49.29	63.16

Note: All times are in seconds and averaged across all surveys and/or simulations. In Panel (a), "Simple" and "Inter" refer to models estimated using `lme4` with no interactions or interactions, respectively. "VI" refers to a model fit using the variational methods in this paper that include the random effect for interaction as well as splines for each continuous covariate. "GAMM4" represents the model with interaction random effects and spline fit with `gamm4`. In Panel (b), the methods correspond to the four variational models estimated in the main text; see Table 1 in the main document for details. In Panel (c), the methods correspond to the models described in the main text (Section 3) and below (Appendix D).

Figure A.4: Relative Mean Absolute Error of Ensembles and Constituent Methods



Note: This plot reports the percentage point gap in mean absolute error between an ensemble of five methods and alternative specifications; positive numbers in blue indicate the ensemble out-performs its competitor. Figure 2 describes the abbreviations. Red dots indicate that the ensemble in the main text out-performs the comptitor by at least 5%.

are constructed for the entire training dataset and each model. Those out of sample predictions are then used to construct the ensemble weights. The constituent models are then fit on the entire training dataset and a weighted average of their predictions is constructed. This is then post-stratified.

The power of the ensemble versus its constituent methods is illustrated in Figure A.4. It shows the percentage gap in mean absolute error (MAE) versus the ensemble of five methods reported in Figure 2. For model k, the table reports $(\text{MAE}_k - \text{MAE}_{\text{Ens}})/\text{MAE}_{\text{Ens}} \cdot 100$ where MAE_{Ens} reports the MAE for the five-model ensemble in the main text. A positive number (blue) indicate the model performs worse than the ensemble; a negative indicates that it beats the ensemble.

I also show the performance of the original ensemble reported in Ornstein ("Ornstein") and an ensemble that includes ten methods (the five in Ornstein; five variational MRP specifications - "All Methods"). Recall that "MRP" refers to a simple MRP fit with the Laplace approximation and flat prior using `glmer`.

A key implication of Figure A.4 shows that individual models usually perform noticeably worse than the ensemble in almost all circumstances and by often non-trivial margins (e.g. more than 5%). Put another way, every constituent method performs more than 5% worse than the ensemble at some sample size examined.

I next show the ensemble weights that come from the 'All Models" ensemble that puts together many different MRP specifications into a single ensemble. The six MRP models (MRP with Laplace approximation and flat prior ["MRP"] and the five variational models)

are given, collectively, around 40-50% of the weight.

Table A.3: Weights Given to Models in Ensemble

Sample Size	Methods in Ornstein (2020)					Variational Methods for MRP				
	MRP	LASSO	KNN	Forest	GBM	Simple	Deep	Spline	Comb.	Deep (IW)
1500	0.119	0.071	0.207	0.042	0.285	0.107	0.061	0.040	0.014	0.053
3000	0.125	0.049	0.176	0.050	0.220	0.139	0.095	0.056	0.029	0.060
5000	0.094	0.047	0.168	0.045	0.210	0.140	0.114	0.073	0.032	0.078
7500	0.065	0.039	0.147	0.032	0.239	0.111	0.167	0.061	0.050	0.088
10000	0.050	0.041	0.138	0.038	0.247	0.102	0.167	0.057	0.060	0.098

Note: This table shows the ensemble weights averaged across all simulations. The first five columns include a hierarchical model fit with a flat prior and Laplace approximation (MRP), LASSO, k-Nearest Neighbors (KNN), a random forest (Forest), and a gradient boosting machine (GBM). The next four columns report hierarchical models estimated with variational inference and a Huang-Wand prior. The final column ("Deep [IW]") reports the results of the deep model using the Inverse-Wishart prior discussed in the main text.

The table also shows an expected but important trade-off between simple and deep MRP; as sample size increases, the deep MRP models (e.g. "Deep" and "Combined") are given increasing weight whereas the traditional MRP models (e.g. "MRP" and "Simple") are given decreasing weight. This is reasonable as the ensemble upweights more complex methods as the amount of data increases. The relatively weak performance of the spline-based methods (e.g. "Spline" and "Combined") may be due to the limited variation in the continuous variables as they are measured at the state-level.

E Additional Empirical Results: BART

This section outlines additional empirical results for the ensemble analysis in Section 4. I begin by explaining the error in the replication data for Bisbee (2019) and then provide additional analyses re-examining the main analyses in the paper.

E.1 Explanation of the Prediction Error

I began by examining the replication code on the paper's Dataverse.[2] The problem arises in the predict stage on the lines colored in red and blue.

```
mrp.form <- as.formula("y ~ pvote + religcon + (1|age) + (1|educ) + (1|gXr)
   + (1|stateid) + (1|region)")

...

temp<-catch.warning(glmer(mrp.form,
```

[2]See doi:10.7910/DVN/LMW871, Version 1.1

```
family=binomial(link="logit"),
data=sample.data))
model<-temp$value
mrp.warn<-ifelse(is.null(temp$warning)==T, 0, 1)

ranvars <- names(ranef(model))
full.ranefs <- rep(list(NA),length(ranvars))

i = 1
for(rv in ranvars) {
full.ranefs[[i]]<- data.frame(effect = rep(NA,length(unique(census.data[,rv
   ])))))}
rownames(full.ranefs[[i]]) <- as.character(c(unique(census.data[,rv])))
for(j in as.character(unique(census.data[,rv])))\{
full.ranefs[[i]][j,1] <- ranef(model)[rv][[1]][j,1]}
}
full.ranefs[[i]][,1][is.na(full.ranefs[[i]][,1])] <- 0
i = i+1
}

names(full.ranefs) <- ranvars

if(length(fixef(model)) > 1) {
fevars <- names(fixef(model))[2:length(fixef(model))]
feres <- apply(sapply(fevars, function(fe) fixef(model)[fe]*census.data[,fe
   ]),1,sum)
} else {
feres <- 0
}

# Create a prediction for each cell in Census data
res <- 0
for(rv in ranvars) {
temp <- full.ranefs[[rv]][census.data[,rv],1]
res <- res+temp
}
mrp.p <- invlogit(fixef(model)["(Intercept)"]
+ res
+ feres)
```

The problem is that when `unique` is called in R, it does not sort the values. Thus, `unique(census.data[,rv])` depends on the order of the rows in the data. For example, if the first values of some column x were, "1, 1, 3, 2, 4, 5,...", then unique would return "1, 3, 2, 4, 5". This creates a discrepancy between the code in red and blue when the predictions are assigned to post-stratification cells.

This causes a critical problem: Arbitrarily reshuffling the row order on the test data will give different predictions. The effect is to randomly inject noise by sometimes arbitrarily giving certain post-stratification cells the wrong random effect depending solely on the order of the rows in the test data (census data). A safe method for predicting would be to use the inbuilt `predict` function from the `lme4` package. In my replication analysis, I call this second function on the identical fitted model from `lme4`.

```
mrp.p.correct <- predict(model, newdata = census.data, allow.new.levels =
   TRUE, type = 'response')
```

I have also found that modifying the red line to include a sort appears to solve the problem, although I have not tested this extensively.

```
rownames(full.ranefs[[i]]) <- as.character(
c(sort(unique(census.data[,rv])))
)
```

Additional discussion of the implication of this prediction error is found on the replication archive for this paper. It also demonstrates that, when corrected, the performance of MRP improves with sample size across all surveys, as expected. **To Reviewers: This file should be attached as "Dataverse_Supplement.PDF".**

E.2 Additional Results for BART

In this section, I replicate other results in Bisbee (2019): (i) presenting fuller results on the mean absolute error, (ii) comparing the correlation between BART and MRP, (iii) showing the sensitivity to omitting state-level predictors, and (iv) sensitivity to sample size.

First, I replicate Table 2 to include all four variational methods. As above, it shows that the spline-based methods perform slightly worse than those without splines. There is also some interesting comparisons between the two simple formulations fit with either the Laplace approximation and flat prior ("MRP") and with the Huang-Wand prior and variational inference ("Simple"). For small sample sizes, the variational method does better by about 1.5% percentage points, although it does slightly worse as sample size becomes very large. This suggests an more nuanced role for the prior and its interaction with sample size that future work into MRP should explore.

Following Bisbee (2019), I show the relationship between the correlation of each method's estimates against the truth. As before, the corrected simple MRP and deep MRP show quite similar correlations to BART.

Table A.5 show the percentage gap in correlation versus BART averaged across the 89 surveys and six sample sizes: $(\rho_k - \rho_{\text{BART}})/\rho_{\text{BART}} \cdot 100$ where ρ_{BART} indicates the correlation of the estimates from the BART model and the true state-leveel values, averaged across 200 simulations. A **negative** number indicates that BART out-performs the other method. As expected from the main text, BART slightly out-performs the MRP models, although at the smallest sample size of 1,500, deep MRP has a slight edge. In general, however, the differences are rather small. For most sample sizes, the difference in correlation between BART and MRP is between about 0-4%.

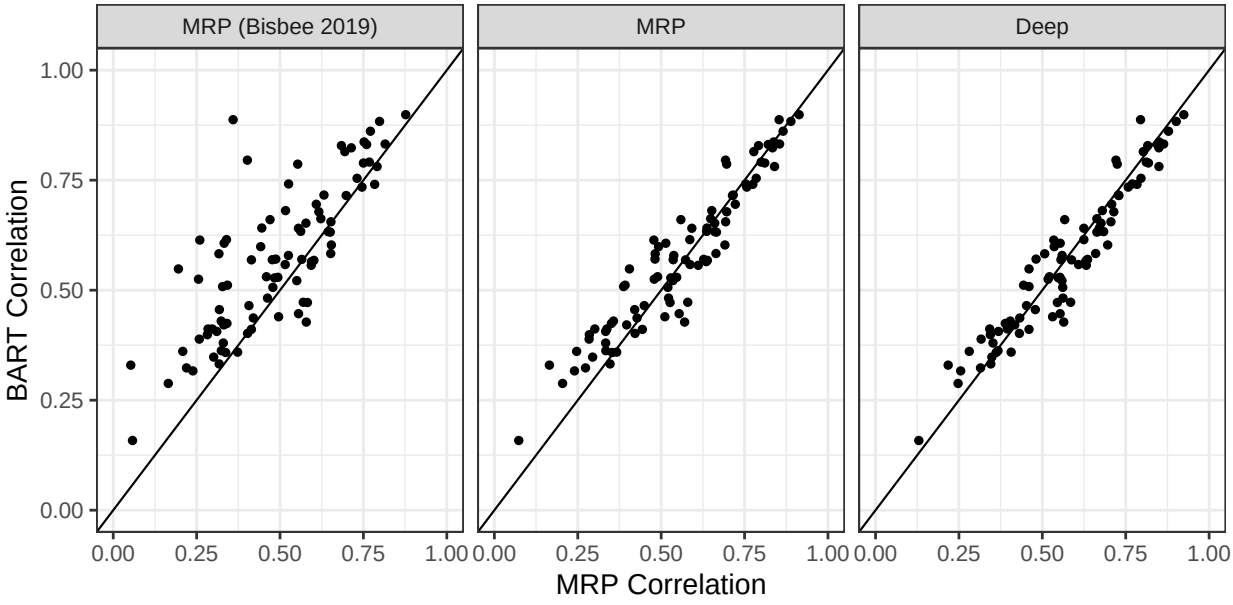
The next result in Bisbee (2019) is about the robustness of BART to mis-specification. The test is as follows: Remove the two contexual predictors (presidential vote share and

Table A.4: Relative Mean Absolute Error versus BART

	Sample Size					
Method	1500	3000	4500	6000	7500	10000
MRP (Bisbee 2019)	22.56	26.30	30.99	35.16	38.72	44.25
MRP	4.54	1.66	1.26	1.04	1.03	1.08
Simple	3.10	0.13	0.25	0.58	1.10	1.84
Deep	2.62	-1.04	-1.01	-0.61	0.10	1.11
Spline	5.59	1.51	1.13	1.41	2.00	2.78
Combined	5.06	0.28	-0.28	-0.03	0.61	1.49

Note: This table reports percentage gap in mean absolute error between BART and the alternative methods; positive numbers indicate that BART out-performs its competitor. Figures 2 and 3 describe the methods.

Figure A.5: Visualizing Performance (Correlation): MRP versus BART



Note: Each plot compares the correlation from an MRP method (on the horizontal axis) to the MAE from BART (on the vertical axis). The 45-degree line is shown. Points above the line indicate that MRP performs worse. The three methods shown are the simple MRP with flat prior, Laplace approximation, and prediction method in Bisbee (2019) ("MRP (Bisbee 2019)"), the simple MRP with the same specifications but a correct prediction ("MRP"), and a deep MRP fit with a Huang-Wand prior and variational inference ("Deep").

religiosity) and see how the mean absolute error changes (increases) and correlation changes (decreases). The claim presented in the paper is that there are relatively minor changes to BART in this mis-specified model whereas it dramatically impacts the performance of MRP. Figure A.6 shows this is not the case. Both the corrected simple MRP and deep MRP lie near the 45-degree line and seem to have similar performance to BART (i.e. similar expected degregdations in performance).

An additional claim about BART is that it can "do more with less data" (Bisbee, 2019,

Table A.5: Relative Correlation versus BART

Method	1500	3000	4500	6000	7500	10000
MRP (Bisbee 2019)	-15.32	-16.97	-17.57	-17.83	-17.58	-17.49
MRP	-4.48	-4.24	-4.27	-4.12	-3.86	-3.63
Simple	-1.09	-2.14	-3.17	-3.67	-3.97	-4.28
Deep	0.38	-0.49	-1.69	-2.30	-2.89	-3.43
Spline	-3.29	-3.41	-3.99	-4.35	-4.61	-4.88
Combined	-1.61	-1.60	-2.28	-2.75	-3.23	-3.66

Note: This table reports the table reports the relative gap (percentage points) in correlation against the truth between BART and the alternative methods named in each row: $(\rho_k - \rho_{\text{BART}})/\rho_{\text{BART}} \cdot 100$. Negative numbers indicate that BART out-performs its competitor. The method names are as described in Figures 2 and 3.

p. 1063). This is tested by examining how the mean absolute error at the state-level changes with the number of observations for that state. Bisbee (2019) does this by comparing the regression coefficients on (scaled) number of observations and absolute error between BART and MRP.

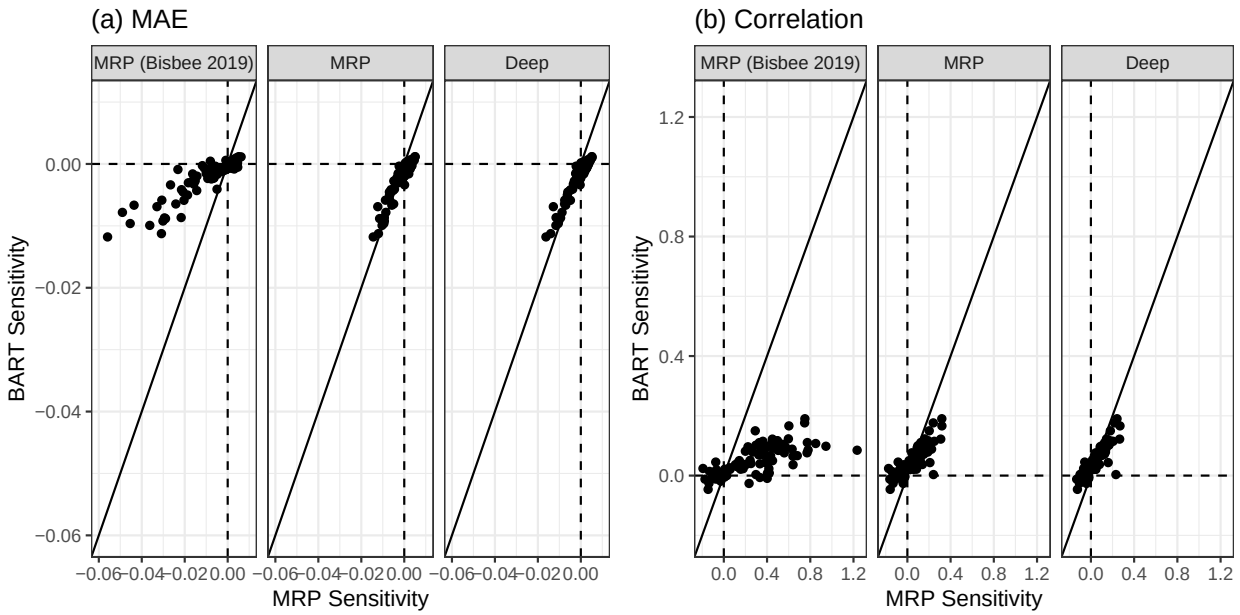
While a reasonable strategy, it has the limitation of conflating two points: As the above results suggest, (traditional) MRP often has a slight disadvantage in performance versus BART. The separate method-by-method regression has the disadvantage of comparing the *slope* with respect to the number of samples but ignoring the *intercept*. Indeed, one might expect that since MRP performs less well at baseline, it might be reasonably expected to have a steeper slope. This is not necessarily evidence that it performs worse with less data, but rather that it can "catch up" to BART quickly as the state-level sample size increases. To illustrate this, Figure A.7 proceeds as follows: for each simulation-state estimate, it takes the number of observations in each state and uses this to predict the state absolute error. It uses a smooth curve to plot the relationship by the four methods under consideration in this Appendix.[3] I present results for 1,500 for simplicity. The two dashed lines indicate the 25-75th percentile range of observed state-level sample sizes.

It shows limited evidence in favor of BART. While BART does perform slightly better for the smallest sample sizes, when we are in the range of most observations (around 10-39 observations per state), the differences are very minor in the curves. An interesting point to note is also that the curves seem to decline more quickly (i.e. improve faster as sample size for a state increases) for the MRP models than with BART. In total, however, the differences between BART and all correctly specified MRP models are rather small in substantive magnitude.

Overall, therefore, this suggests that while BART has a slight edge under certain circumstances (e.g. for very small numbers of observations per state), there is limited evidence of it being systematically better than MRP. It does not appear to be substantially more robust to mis-specification nor have materially better performance in the simulations considered in Bisbee (2019).

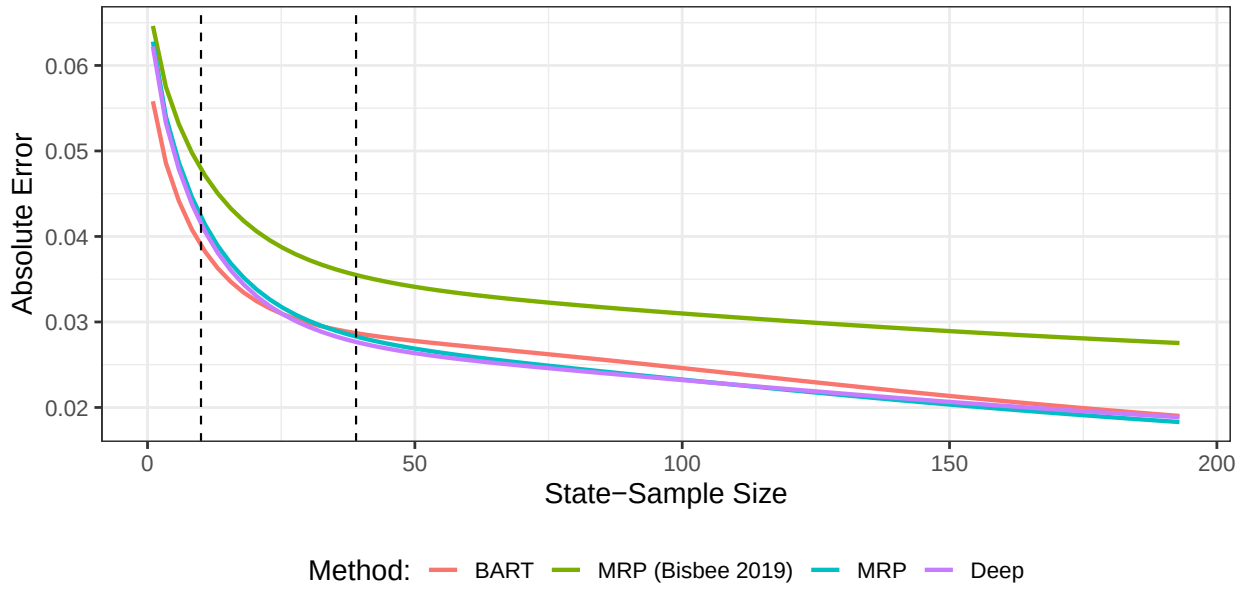
[3]Specifically, given the skewed distrubtion of the sample size per state, I use `geom_smooth(..., method = 'gam', formula = y ~ s(log(x), k =5))`.

Figure A.6: Sensitivity to Mis-Specification



Note: This figure reports the change in MAE or correlation between the main specification and one that removes the state-level continuous predictors. Negative values for MAE indicates that doing so hurts performance; positive values for correlation indicates that doing so hurts performance.

Figure A.7: State-Level Sample Size and Absolute Error



Note: This figure plots a smooth regression between the number of state-level observations and the observed absolute error in the state prediction. The curve for each method is shown in the corresponding color.

F Software

This appendix provides a brief demonstration of how to use the accompanying software available at [REDACTED]; it can be installed directly into R using, e.g., `devtools::install_github()`. Broadly speaking, the package has been designed to be as similar as possible to `lme4` in terms of how random effects are specified. A similar syntax to `mgcv` is used for estimating splines. The default options use a Huang-Wand prior as well as both acceleration techniques; these can be modified as shown below.

For simplicity, I rely on a dataset from the `pscl` package on the votes of members of Congress for the Iraq War. The first model illustrates the flexibility of the accompanying software from this paper (`vglmer`) as it includes multiple random effects, random intercpets and random slopes, as well as splines. The second example illustrates how default options could be modified if desired.

```
# Fitting Ensembles
library(SuperLearner)
# Fitting variational hierarchical models ("variational + glmer")
library(vglmer)

# Load data from pscl on the Iraq War Vote
iraq_data <- pscl::iraqVote
iraq_data$region <- state.region[match(iraq_data$state.abb, state.abb)]
head(iraq_data)
#>   y state.abb             name  rep state.name gorevote region
#> 1 1        AL  SESSIONS (R AL) TRUE    Alabama    41.59  South
#> 2 1        AL    SHELBY (R AL) TRUE    Alabama    41.59  South

# "v_s(gorevote)" estimates a non-linear effect using penalized splines.
fit_vglmer <- vglmer(y ~ v_s(gorevote) + (1 | region) +
(1 + gorevote | rep), data = iraq_data, family = 'binomial')

fit_vglmer <- vglmer(y ~ v_s(gorevote) + (1 | region) +
(1 + gorevote | rep), data = iraq_data,
# Manually specify  a Huang-Wand prior
control = vglmer_control(prior_variance = 'hw'),
family = 'binomial')
```

From this model, output can be extracted in a highly similar way to `lme4`. Predictions can also be done in a straightforward fashion.

```
# Terms can be extracted in a similar fashion to lme4
ranef(fit_vglmer)
#> $region
#>              id (Intercept)
#> 1     Northeast   0.2501737
```

```
#> 2         South   0.2967595
#> 3 North Central  -0.3242042
#> 4          West  -0.2227290
#>
.... [Other Random Effects Not Shown for Space]

fixef(fit_vglmer)
#> (Intercept)    gorevote
#>   8.7964438  -0.1423059
vcov(fit_vglmer)
#>             (Intercept)      gorevote
#> (Intercept)  2.18534247 -0.0449548442
#> gorevote    -0.04495484  0.0009501862

# Predict and turn into probability scale
plogis(
predict(fit_vglmer, newdata = iraq_data, allow_missing_levels = TRUE)
)
```

Finally, a key implication of the paper is that hierarchical models perform well in ensembles. The accompanying package extends the popular `SuperLearner` package in R to accomodate hierarchical models that can take a formula as an argument. Thus, one can easily specify an ensemble that includes both simple and deep MRP, and use a data-driven procedure to select the optimal combination.

```
# The formula must be added manually like any other tuning parameter for
# SuperLearner (e.g. using "create.Learner" or manually as below)
SL_v1 <- function(...){
SL.vglmer(formula = y ~ v_s(gorevote) + (1 | region) +
(1 + gorevote | rep), ...)
}

fit_SL <- SuperLearner(Y = iraq_data$y,
X = iraq_data[,c('gorevote', 'region', 'rep')],
family = binomial(), verbose = TRUE,
SL.library = c('SL.ranger', 'SL.glmnet', 'SL.bartMachine', 'SL_v1'))
```

References

Bates, Douglas, Martin Mächler, Ben Bolker and Steve Walker. 2015. "Fitting Linear Mixed-Effects Models Using lme4." *Journal of Statistical Software* 67(1):1–48.

Bisbee, James. 2019. "BARP: Improving Mister P Using Bayesian Additive Regression Trees." *American Political Science Review* 113(4):1060–1065.

Chung, Yeojin, Andrew Gelman, Sophia Rabe-Hesketh, Jingchen Liu and Vincent Dorie. 2015. "Weakly Informative Prior for Point Estimation of Covariance Matrices in Hierarchical Models." *Journal of Educational and Behavioral Statistics* 40(2):136–157.

Eilers, Paul HC and Brian D Marx. 1996. "Flexible smoothing with B-splines and penalties." *Statistical science* 11(2):89–121.

Gelman, Andrew. 2006. "Prior Distributions for Variance Parameters in Hierarchical Models." *Bayesian Analysis* 1(3):515–533.

Gelman, Andrew and Jennifer Hill. 2006. *Data Analysis Using Regression and Multilevel/Hierarchical Models*. Cambridge University Press.

Goplerud, Max. 2021. "Fast and Accurate Estimation of Non-Nested Binomial Hierarchical Models Using Variational Inference." *Bayesian Analysis* Forthcoming.

Huang, Alan and Matt P. Wand. 2013. "Simple Marginally Noninformative Prior Distributions for Covariance Matrices." *Bayesian Analysis* 8(2):439–452.

Jaakkola, Tommi S. and Yuan Qi. 2007. Parameter Expanded Variational Bayesian Methods. In *Neural Information Processing Systems 2007*.

Ornstein, Joseph T. 2020. "Stacked Regression and Poststratification." *Political Analysis* 28(2):293–301.

Polson, Nicholas G., James G. Scott and Jesse Windle. 2013. "Bayesian Inference for Logistic Models Using Pólya–Gamma Latent Variables." *Journal of the American Statistical Association* 108(504):1339–1349.

Ruppert, David, Matt P Wand and Raymond J Carroll. 2003. *Semiparametric regression*. Cambridge University Press.

Van Dyk, David A. and Ruoxi Tang. 2003. "The One-Step-Late PXEM Algorithm." *Statistics and Computing* 13(2):137–152.

Varadhan, Ravi and Christophe Roland. 2008. "Simple and globally convergent methods for accelerating the convergence of any EM algorithm." *Scandinavian Journal of Statistics* 35(2):335–353.

Zhao, Yihua, John Staudenmayer, Brent A. Coull and Matt P. Wand. 2006. "General Design Bayesian Generalized Linear Mixed Models." *Statistical Science* 21(1):35–51.