# Comparing MCMC and VaNBayes for Heteroskedastic Models

Adira Cohen Connor McNeill Miles Woollacott

December 14, 2024

**The Honor Pledge:** *We have neither given nor received unauthorized aid on this assignment.*

Signed: Adira Cohen, Connor McNeill, Miles Woollacott

**Abstract**

Markov chain Monte Carlo (MCMC) sampling is currently one of the most popular Bayesian sampling techniques. However, MCMC can be difficult to implement for complex likelihoods. Alternative likelihood-free simulation-based approaches, such as Approximate Bayesian Computation (ABC), only work well for simple datasets. Therefore an alternative is needed which is both likelihood-free and can handle complex datasets. VaNBayes is a new approach proposed by Maceda et al. in 2024 [Mac+24] which can handle just such a situation. The algorithm works in two stages. First, many hypothetical datasets are simulated by drawing parameter values and generating corresponding datasets. A two-layer neural network (NN) is fit for each parameter to predict the parameter value using summary statistics of the data. Second, the observed data is plugged into the trained NN to output a corresponding approximate posterior distribution. To compare the performance of MCMC and VaNBayes, we use a heteroskedastic model for response $\boldsymbol{Y} \sim f(\boldsymbol{Y}|\boldsymbol{\theta})$ in which the mean is linearly related to the data through location parameter $\boldsymbol{\beta}$ and the variance is related to the data through

a multiplicative error term involving scale parameter $\boldsymbol{\gamma}$ where both $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are sparse. We generate a fixed dataset $\boldsymbol{X}$ with correlated columns. We then conduct a simulation study where for each simulation we fit MCMC with spike-and-slab priors for $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ as well as VanBayes. We compare the performance of (1) the posterior inclusion probabilities (PIPs) to measure the ability of the models to capture the sparsity of the parameters and (2) the posterior prediction distributions (PPDs) of separately generated datasets. MCMC worked very well for $\boldsymbol{\beta}$ and relatively well for $\boldsymbol{\gamma}$, while VaNBayes worked well for some elements of $\boldsymbol{\beta}$ but not others and failed for $\boldsymbol{\gamma}$. Both methods performed well on prediction, but VaNBayes had higher variance. Finally, we discuss several potential fixes for the issues we encountered with VaNBayes. We first consider choosing more meaningful summary statistics for our model because our current summary statistics do not capture the heteroskedasticity of the model; this explains VaNBayes' poor performance for $\gamma$. We also discuss changing some of the model settings, such as the size of the NN and the magnitudes of the parameters.

# 1 Introduction

In this report, we compare Markov chain Monte Carlo (MCMC) and VanBayes under the setting of a heteroskedastic model. MCMC is one of the best computational methodologies in Bayesian inference for computing posterior distributions. However, MCMC might not yield optimal results if the likelihood is complex. This means we must pivot to more flexible methodologies. Approximate Bayesian Computing (ABC) is one such popular, flexible model that is likelihood-free. However, ABC uses rejection sampling based on how close the simulated distribution is to the observed data, along with the specification of some distance metric. Therefore, this method can be extremely slow to converge for complicated datasets. We then must turn to a likelihood-free, flexible model known as VaNBayes. Recently proposed by Maceda et al. [Mac+24], VaNBayes employs neural networks and

summary statistics to better infer the posterior distribution of the parameters and posterior predictive distributions (PPDs) under less traditional models.

The structure of this report is as follows. In section 2, we discuss the statistical model of interest and the setup we will use for MCMC. In addition, we provide an introduction to VaNBayes, including a general overview and advantages/disadvantages. In section 3, we introduce and implement our simulation study used to compare MCMC to VaNBayes. In section 4, we summarize our results and provide groundwork for future studies.

# 2 Bayesian Model and Associated Methodology

## 2.1 Setup

The model we consider is of the form

$$Y_i = \beta_0 + \boldsymbol{X}_i^T \boldsymbol{\beta} + \text{expit}\left\{\gamma_0 + \boldsymbol{X}_i^T \boldsymbol{\gamma}\right\} \cdot \epsilon_i,$$

where $\text{expit}(x) = [1 + \exp(-x)]^{-1}$. Here, $\boldsymbol{X} \in \mathbb{R}^{n \times p}$, $\beta_0$ and $\gamma_0 \in \mathbb{R}$, $\boldsymbol{\beta}$ and $\boldsymbol{\gamma} \in \mathbb{R}^p$ and $\epsilon_i \overset{\text{iid}}{\sim} \mathcal{N}(0, 1)$. We will be working in a situation where both $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are sparse. In addition, $\text{Corr}(\boldsymbol{X}_{\cdot i}, \boldsymbol{X}_{\cdot j}) = \rho^{|i-j|}$, where $\rho \in (0, 1)$ is some constant. In this report, we use the values $n = 50$, $p = 10$, and $\rho = 0.5$ to generate our data. In addition, the true values of our parameters are set at $\beta_1 = \beta_2 = \beta_6 = \frac{1}{2}$ (0 otherwise), and $\gamma_2 = \gamma_3 = \gamma_{10} = 1$ (0 otherwise).

It is important to note that the error term is heteroskedastic, with the first term in the product partially depending on $\boldsymbol{X}$. This means that we have to standardize $\boldsymbol{X}$ such that the error term is not excessively large. Even when $\boldsymbol{X}$ is standardized, issues arose when we initially considered using $\exp(x)$ in the multiplicative error term, since this term became

very large quickly. We should note that JAGS was actually able to handle the exponential error term somewhat utilizing MCMC, but in order to be able to make fair comparisons between MCMC and VaNBayes, the decision was made to utilize the expit function instead which helps with stabilizing our values.

Lastly, for this report we generated 10 extra $Y$ terms for validation, denoted as $Y^*$ with corresponding $\boldsymbol{X}^*$. This will be used in the simulation study to assess model predictive potential, where we will obtain the posterior predictive distribution (PPDs) of the $Y^*$'s.

## 2.2 MCMC

For MCMC, we utilize the following priors (using $\tau$ as the precision, 1/variance):

$$\beta_j|\pi_j \sim \begin{cases} \mathcal{N}(0, \tau_\beta) & \text{with probability } \pi_j \\ 0 & \text{with probability } 1-\pi_j \end{cases}$$

$$\gamma_j|\tilde{\pi}_j \sim \begin{cases} \mathcal{N}(0, \tau_\gamma) & \text{with probability } \tilde{\pi}_j \\ 0 & \text{with probability } 1-\tilde{\pi}_j \end{cases}$$

$$\pi_j, \tilde{\pi}_j \sim \text{Beta}(0.5, 0.5)$$

$$\beta_0, \gamma_0 \sim \mathcal{N}(0, 0.1)$$

$$\tau_\beta, \tau_\gamma \sim \text{Gamma}(0.1, 0.1)$$

The priors on $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are spike-and-slab priors with a point mass at 0 and a normal distribution utilized as the spike and the slab, respectively. This allows us to look at the sparsity of these two parameters. We then look at the posterior inclusion probabilities (PIPs) for both $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ which is the proportion of times that our coefficient is non-zero. We also obtain the PPDs using JAGS in a similar setup to that used in the textbook

[RG19].

### 2.3 VaNBayes

VaNBayes is a variational Bayes method proposed by Maceda et al. [Mac+24] to address the limitations of current Bayesian sampling methods for difficult-to-fit likelihoods. This method leverages the advantages of machine learning and Bayesian inference to create a highly flexible likelihood-free sampling algorithm. The key question that VaNBayes asks is, why should we only use the information given by one dataset when we could generate as many as we need? Say we have data $\boldsymbol{Y}$ with likelihood function $f(\boldsymbol{Y}|\boldsymbol{\theta})$. VaNBayes works in two stages:

**(1) Training stage.** The first stage handles learning the relationship between the parameters and the data. First, hypothetical parameter values $\boldsymbol{\theta}$ are simulated from their prior distribution. These are then used to generate datasets $\boldsymbol{Y}$ using the likelihood $f(\boldsymbol{Y}|\boldsymbol{\theta})$ (see the red points in figure 1). Next, summary statistics $\boldsymbol{Z} = S(\boldsymbol{Y})$ are calculated for the dataset to describe its most important characteristics. For example, $\boldsymbol{Z}$ could be the regression coefficients and residuals or even the entire dataset, i.e. $\boldsymbol{Z} = \boldsymbol{Y}$. Finally, a two-layer neural network (NN) is fit for each parameter $\theta_j$ which predicts the parameter value using the summary statistics $\boldsymbol{Z}$. The NN weights $\widehat{\boldsymbol{W}}$ are saved. In this way, the NN learns the relationship $\boldsymbol{\theta}|\boldsymbol{Y}$.

**(2) Data-fitting stage.** The second stage predicts the true parameter values using the observed data. This is immediate using the trained weights $\widehat{\boldsymbol{W}}$ on summary statistics of the observed data, which outputs an estimate of the posterior distribution $\boldsymbol{\theta}|\boldsymbol{Y}_{\text{obs}}$. For example, note the distribution of $\theta$ along the the observed data $Y_{\text{obs}} = 2$ (blue) in figure 1.

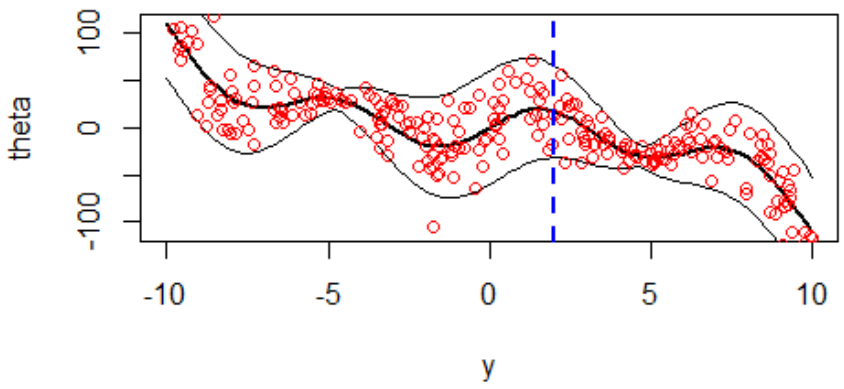


Figure 1: A toy example used to illustrate the VaNBayes approach discussed in section 2.3. Here $\boldsymbol{Y} \sim f(\boldsymbol{Y}|\theta)$ for $\theta \in \mathbb{R}$ and some difficult-to-fit density $f$. Draws were made from $\theta$ using a uniform prior on $(-10, 10)$, which were then used to generate training data $\boldsymbol{Y}$, shown in red. The mean and 2 standard deviations from the mean are given in black. The observed value of $Y_{\text{obs}} = 2$ is given in blue.

The details of the VaNBayes algorithm are given in Algorithm 1, which is based on Algorithm 1 from Maceda et al [Mac+24].

---

**Algorithm 1** VaNBayes

---

**Require:** Observed data $\boldsymbol{Y_0}$, likelihood function $f(\boldsymbol{Y}|\boldsymbol{\theta})$, training distribution $\Pi(\boldsymbol{\theta})$ and prior distribution $\pi(\boldsymbol{\theta})$. Let $\boldsymbol{Z} = S(\boldsymbol{Y})$ be summary statistics. Propose a variational posterior for $\theta$ as $p\{\theta|a(\boldsymbol{Z}; \boldsymbol{W})\}$ for a neural network $a(\boldsymbol{W}; \boldsymbol{Z})$.

1. **for** $i = 1, \ldots, N$ do
2. Generate $\boldsymbol{\theta}_i \sim \Pi(\boldsymbol{\theta})$ and then $\boldsymbol{Y}_i \sim f(\boldsymbol{Y}|\boldsymbol{\theta}_i)$
3. Compute $\boldsymbol{Z}_i = S(\boldsymbol{Y}_i)$

4. **end for**

5. Select $\widehat{\boldsymbol{W}} = \text{argmax}_{\boldsymbol{W}} \sum_{i=1}^{N} \frac{\pi(\boldsymbol{\theta}_i)}{\Pi(\boldsymbol{\theta}_i)} \log[p\{\theta_i | a(\boldsymbol{Z}_i; \boldsymbol{W})\}]$

6. Return $p\{\theta_i | a(\boldsymbol{Z}_0; \widehat{\boldsymbol{W}})\}$ to approximate the true posterior of $\theta$.

---

Convergence of the NN to weights which minimize the Kullback-Leibler divergence is theoretically guaranteed by theorem 1 below. See Maceda et al for a proof.

**Theorem 1**[1]**.** *Let $m_0(\boldsymbol{Z})$ be the marginal distribution of the summary statistic data and $p\{\boldsymbol{\gamma} | a(\boldsymbol{Z}; \boldsymbol{W})\}$ be the posterior distribution when the hyperparameters are modeled with the neural network outputs. The weights chosen by VaNBayes, $\widehat{\boldsymbol{W}}$, converge in probability to (possibly non-unique) weights that minimize the Kullback-Leibler divergence between the (true) joint posterior and $p\{\boldsymbol{\gamma} | a(\boldsymbol{Z}; \boldsymbol{W})\} m_0(\boldsymbol{Z})$ as the number of training samples, $N$, diverges.*

Theorem 1 implies that the only limitation to VaNBayes with respect to prediction accuracy is time. It is always possible to simulate more training parameters and datasets, which will in theory eventually lead to convergence of the model to the true posterior distribution. Also note that theorem 1 guarantees convergence with respect to the summary statistics, so it is critical to choose meaningful summary statistics for the question being asked.

**Advantages.** A convenient consequence of using NN is that once the NN has been trained, predictions can be made almost instantly off of new datasets. VaNBayes also provides a great deal of flexibility with hard-to-fit models, especially because it is likelihood-free. Finally, as stated above, VaNBayes should converge given enough simulated datasets.

**Disadvantages.** On the flip side, an inconvenient consequence of using NN is that NN can be slow to fit, and a new NN must be fit to each parameter. This greatly limits

[1] Theorem 2 from Maceda et al [Mac+24]

the performance in higher dimensions. Another consequence is that NN are black-box algorithms, greatly limiting interpretation.

**Cautions.** Care must be taken to prevent "blow-up" in the data generation step. In the process of drawing new parameter values from its prior distribution and generating datasets, certain unlucky draws of $\boldsymbol{\gamma}$ would give extreme outliers for $\boldsymbol{Y}$ due to the exponential variance term, which greatly limited our ability to train the NN. Therefore care must be taken when generating datasets in order for VaNBayes to fit properly. We fixed this issue by using expit instead of exp, which maps to the interval $(0, 1)$ and therefore prevents "blow-up." Another critical caution is the choice of summary statistics, which is discussed further in the conclusion.

# 3 Simulation Study

## 3.1 Setup

The goal of our simulation study is to compare the performance of MCMC and VaNBayes with the heteroskedastic data generating model. Since both $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are sparse, it makes sense to study the posterior inclusion probabilities (PIPs) - e.g., for $\boldsymbol{\beta}$, we find $P(\beta_j \neq 0|\boldsymbol{Y})$ for $j \in \{1, \ldots, p\}$ - so that we see which coefficients are included in the model. These are probabilities from the posterior distribution of our coefficients. We do the same thing for $\boldsymbol{\gamma}$. This is done for both MCMC and VaNBayes each time we run the simulation. To study how the predictions $Y^*$ are doing, we obtain the posterior median of the posterior predictive distribution (PPD) for each $k \in \{1, \ldots, n_{\text{pred}}\}$.

The workflow of the simulation study is the following:

1. Generate our covariate matrix $\boldsymbol{X}$ and the covariate matrix for the predictions $\boldsymbol{X}^*$.

These are treated as known and fixed for the simulation study.

2. Fit NN for the PIPs of $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ and the posterior quantiles (5%, median, and 95%) of the PPDs (this is done for 100,000 datasets).

3. For each simulation (this is done 100 times in our study):

   i. Generate data $Y$ from our data generating model.

   ii. Generate summary statistics $Z$ from $\boldsymbol{X}$ and $Y$. We will utilize the frequentist least squares estimator for $\boldsymbol{\beta}$, $(\boldsymbol{X}^T\boldsymbol{X})^g\boldsymbol{X}^TY$, and the standard error of the residuals as our summary statistics, SD $\left(Y - (\boldsymbol{X}^T\boldsymbol{X})^g\boldsymbol{X}^TY\right)$.

   iii. Fit the MCMC on the generated data and obtain PIPs (for $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$) and the median of the PPDs (for the predictions).

   iv. Obtain the PIPs (for $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$) and the median of the PPDs (for the predictions) using VaNBayes.

4. Plot the boxplots to determine overall distribution of the PIPs and the posterior medians of the PPDs.

### 3.2 Results

Figure 2 shows the distribution of the PIPs for $\boldsymbol{\beta}$. We see that MCMC does a really good job of predicting which $\boldsymbol{\beta}$'s are nonzero: $\beta_1, \beta_2,$ and $\beta_6$ had PIPs of around 1 nearly for each simulation. VaNBayes, on the other hand, appears to have some difficulties. We investigated that it had to do with the NN training: when the training worked properly (as it did for $\beta_5, \beta_6, \beta_7,$ and $\beta_8$) we see that VaNBayes actually did good with predicting which of the coefficients were equal to 0. However, the remaining ones are not detected either way.

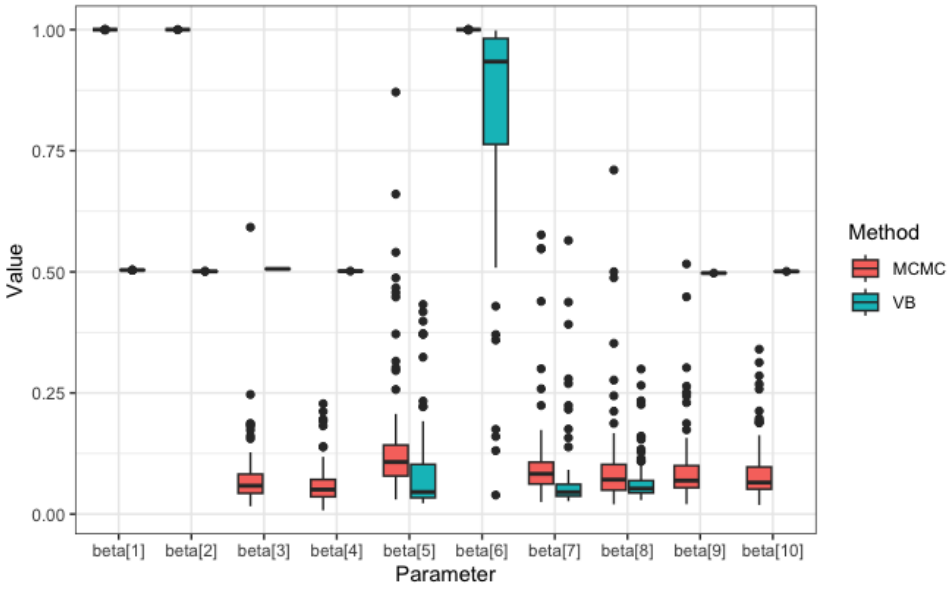


Figure 2: This figure shows the distribution of the posterior inclusion probabilities (PIPs) for $\boldsymbol{\beta}$ for MCMC (red) and VaNBayes (blue).

Figure 3 shows the distribution of the PIPs for $\boldsymbol{\gamma}$. Although the performance was not as good as for $\boldsymbol{\beta}$, MCMC was still able to notice that $\gamma_2, \gamma_3,$ and $\gamma_{10}$ were non-zero. VaNBayes was unable to determine these as the PIPs are nearly the same across all the different coefficients at around 0.5.

Figure 4 shows the distribution of the posterior medians of the PPDs for our predictions $Y^*$. We see that both MCMC and VaNBayes performed similarly to one another when it came to the accuracy of the center of the distribution. However, the distributions of the posterior medians for VaNBayes are much more spread out than those for MCMC.

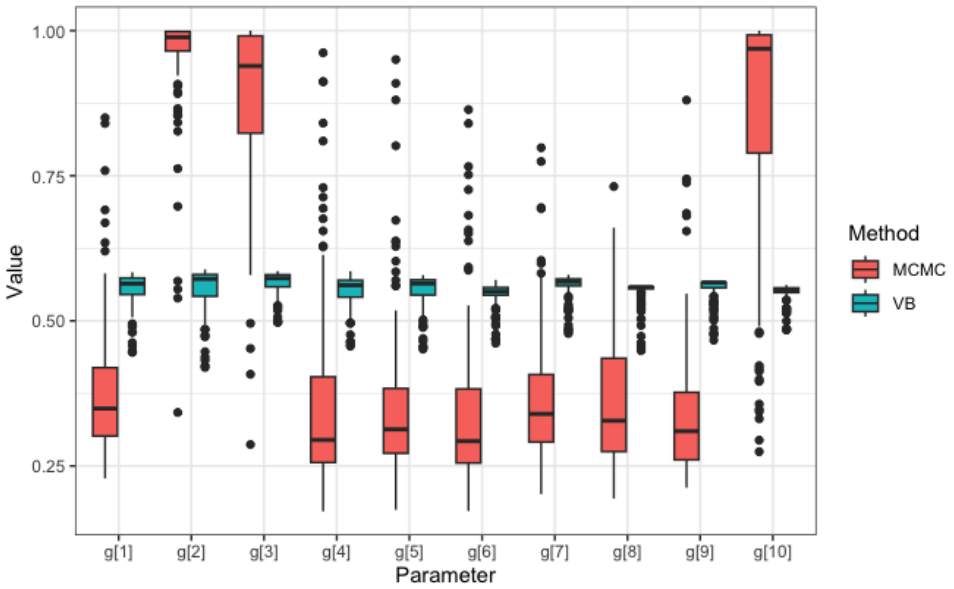


Figure 3: This figure shows the distribution of the posterior inclusion probabilities (PIPs) for $\boldsymbol{\gamma}$ for MCMC (red) and VaNBayes (blue).

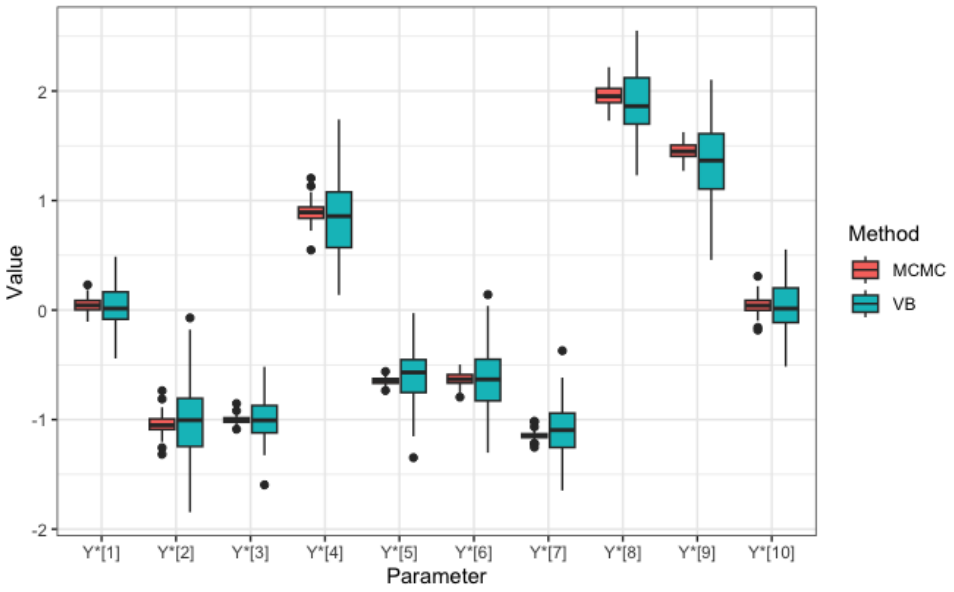


Figure 4: This figure shows the distribution of the posterior medians of the posterior predictive distributions (PPDs) for $Y^*$

# 4 Conclusion

For our toy example, MCMC performed relatively well at predicting inclusion probability for $\boldsymbol{\beta}$ and even moderately well for $\boldsymbol{\gamma}$, while VaNBayes struggled, particularly with $\boldsymbol{\gamma}$. Both models performed well with prediction, although VaNBayes had higher variance.

It is very likely that with some improvements to our approach, VaNBayes could give a much better fit for this type of model. The first avenue for improvement would be to choose more meaningful summary statistics for our model. We hypothesize that the reason VaNBayes predicts around 50% PIP for $\boldsymbol{\gamma}$ is that the summary statistics we give do not capture the heteroskedasticity, and so there is no way for the NN to learn a relationship between $\boldsymbol{\gamma}$ and $\boldsymbol{Y}$. We briefly attempted using the entire dataset $\boldsymbol{Y}$ as the summary statistics, with no improvement. However, this may also be due to the size of the NN. Perhaps a larger neural net with more informative summary statistics could yield better results. We could also try simplifying the model, such as reducing the size of $\boldsymbol{\gamma}$ (and therefore the number of columns of $\boldsymbol{X}$ that affect the variance) or increasing the magnitude of $\boldsymbol{\beta}$ to increase the signal.

# 5 References

# References

[RG19] Brian J. (Brian James) Reich and Sujit K. Ghosh 1970-. *Bayesian Statistical Methods*. Boca Raton: CRC Press, Taylor & Francis Group, 2019. ISBN: 978-0-429-20229-2.

[Mac+24] Elliot Maceda et al. *A variational neural Bayes framework for inference on intractable posterior distributions*. 2024. arXiv: 2404.10899 [stat.CO]. URL: https://arxiv.org/abs/2404.10899.

# 6 Appendix

## 6.1 JAGS Model R Code

```
# Likelihood
for (i in 1:n) {
  Y[i] ~ dnorm(mu[i], tau[i])
  mu[i] = beta0 + inprod(X[i,], beta)
  tau[i] = 1/sigma2[i]
  logit(sigma2[i]) = gamma0 + inprod(X[i,], gamma)
}
# Priors
beta0 ~ dnorm(0, 0.01)
gamma0 ~ dnorm(0, 0.01)
for (j in 1:p) {
  # Prior for Beta
  beta[j] = id_beta[j]*b[j]
  b[j] ~ dnorm(0, taub)
  id_beta[j] ~ dbern(pi[j])
  pi[j] ~ dbeta(0.5, 0.5)
  # Prior for Gamma
  gamma[j] = id_gamma[j]*g[j]
```

```
  g[j] ~ dnorm(0, taug)
  id_gamma[j] ~ dbern(pitilde[j])
  pitilde[j] ~ dbeta(0.5, 0.5)
}
taub ~ dgamma(0.1, 0.1)
taug ~ dgamma(0.1, 0.1)
# Predictions
for (i in 1:n_pred) {
  Y_pred[i] ~ dnorm(mu_pred[i], tau_pred[i])
  mu_pred[i] = beta0 + inprod(X_pred[i,], beta)
  tau_pred[i] = 1/sigma2_pred[i]
  logit(sigma2_pred[i]) = gamma0 + inprod(X_pred[i,], gamma)
}
```