

Bayesian Ridge Regression - Second Report

Dominik Strache, Nicolai Bäuerle & Joel Beck

Introduction

Our first report was focused on the underlying mathematical model of Bayesian Ridge Regression and provided a brief introduction to the `asp21bridge` package. Since then, the central `mcmc_ridge()` function, which implements the Markov Chain Monte Carlo sampler and whose name has been adapted to meet existing naming conventions of other groups, has been slightly extended as well as fully unit tested and now shows a stable and reliable performance.

This report investigates the effect of manipulating some of the sampling parameters in a controlled environment, such that changes in the estimation outcome can be directly linked to respective changes in the model inputs. After an introductory exploration phase, we decided for a subset of all possible model variations that indicated the greatest potential for interesting and relevant findings.

As a result, the simulation studies of this report will be conducted on the following components:

- The **data** input (sections 1 and 2), which is captured by the function argument `m` or alternatively the combination of `X`, `Z` and `y`. Here, the sampler's *robustness* is tested across various scenarios.
- The **sample size** `n` (section 3) and the **number of simulations** `num_sim` (section 4). Here, we are looking for a possible stabilization process with increasing either of these two input parameters hinting at *asymptotic/convergence* properties.
- The **hyperparameters** `a_tau`, `b_tau`, `a_xi` and `b_xi` (section 5) of the Inverse Gamma prior distribution of the variance parameters τ^2 and ξ^2 , as specified in the first report. The effect of hyperparameters in a hierarchical Bayesian model can be difficult to predict based on pure logical reasoning. Therefore simulations are a useful tool to either confirm prior assumptions or discover unexpected behaviour.

Since the resulting simulation studies serve different purposes (e.g. diagnostic vs. explorative), they demand for different approaches in the simulation settings, the implementation as well as the analysis and presentation of the results. For that reason, we decided against forcing all of the following sections into one common rigid framework. Instead, each section individually motivates, explains and interprets the methods chosen for its particular use case.

In order to keep the analysis compact and succinct, there will be almost no code included. It is worth noting though that the `R Markdown` document generating this report as well as all `R Scripts` used for the simulations are contained in the `simulation-studies` folder of the `asp21bridge` package. Thus, each figure as well as all numerical results are fully reproducible and can be repeated and extended by the reader.

1 Correlated Predictor Variables

Up to this point, we have often illustrated the usage and results of the `mcmc_ridge()` sampler with simulated data from the built-in `toy_data` set. There, each regressor variable is independently sampled from a normal distribution and the outcome variable is simulated based on a correctly specified location-scale regression model $y_i \sim \mathcal{N}(\mathbf{x}_i^T \boldsymbol{\beta}, \exp(\mathbf{z}_i^T \boldsymbol{\gamma})^2)$. All these conditions lead to an excellent performance of the `mcmc_ridge()` sampler, but might arguably not represent the most challenging task.

Sections 1 and 2 analyze the sampler’s performance on simulated data, which might be closer to data found in the real world. First, we will induce correlation among the predictor variables, whereas in the following section the distributional assumptions are considerably changed. Further, the `mcmc_ridge()` performance is compared to the Maximum Likelihood based `lmls()` estimates and the Markov Chain Monte Carlo `mcmc()` sampler without penalty from the `lmls` package.

1.1 Simulation Setting

- The design matrix $\mathbf{X} = (\mathbf{x}_1 \quad \mathbf{x}_2 \quad \mathbf{x}_3)$ is simulated from a three dimensional normal distribution $\mathcal{N}_3(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with mean vector $\boldsymbol{\mu} = (-5 \quad 2 \quad 0)^T$ and covariance matrix $\begin{pmatrix} 1 & \rho & \rho \\ \rho & 3 & \rho \\ \rho & \rho & 5 \end{pmatrix}$. Hence, the dependence among the regressors is fully determined by the parameter ρ .
- The design matrix $\mathbf{Z} = (\mathbf{z}_1 \quad \mathbf{z}_2)$ consists of linear combinations of the regressors \mathbf{x}_1 up to \mathbf{x}_3 , more specifically $\mathbf{z}_1 = 0.8 \cdot \mathbf{x}_1 + 0.2 \cdot \mathbf{x}_2$ and $\mathbf{z}_2 = \mathbf{x}_2 - 0.5 \cdot \mathbf{x}_3$.
- In both design matrices intercept columns are added for estimation purposes. The true coefficient vectors are given by $\boldsymbol{\beta} = (\beta_0 \quad \beta_1 \quad \beta_2 \quad \beta_3)^T = (0 \quad 3 \quad -1 \quad 1)^T$ and $\boldsymbol{\gamma} = (\gamma_0 \quad \gamma_1 \quad \gamma_2)^T = (0 \quad 2 \quad 0)^T$.
- Three different values were chosen for $\rho \in \{0, -0.5, 0.9\}$ to compare the ‘nice’ case of uncorrelated predictors with the performance for negative and positive dependence. For each covariance structure the three models `mcmc_ridge()`, `mcmc()` and `lmls()` were fitted, where each Posterior Mean estimate from both of the Markov Chain Monte Carlo samplers is based on 1000 samples.
- Moreover, we compared the performance of the usual `mcmc_ridge()` implementation, which draws $\boldsymbol{\beta}$ from the closed form full conditional (multivariate normal) distribution, with an alternative sampling process that uses a Metropolis-Hastings approach for both, the location parameter $\boldsymbol{\beta}$ as well as the scale parameter $\boldsymbol{\gamma}$. The latter is initiated by the `mcmc_ridge()` argument `mh_location = TRUE`. The variance of the corresponding proposal distribution is set to a carefully chosen default value, but can be manually changed by means of the `prop_var_loc` argument.

1.2 Simulation Results

Figure 1 displays the posterior mean estimates for both MCMC samplers and the Maximum Likelihood estimates for the `lmls()` function of one complete simulation cycle. For a better visual comparison the true values for each coefficient are indicated by grey circles, whereas the acceptance rate(s) of the Metropolis-Hastings algorithm used in the sampling process are provided in grey boxes.

The scaling of the x - axis is dominated by one outlier in the lower panel for each correlation structure. While the Metropolis-Hastings approach for $\boldsymbol{\beta}$ performs moderately well for most of the coefficients, it massively overestimates the intercept β_0 .

This observation can be made across many different data sets: In some special cases the performance is close to (but never better) than sampling directly and independently from a multivariate normal distribution. However, most of the time the performance is significantly worse and the samples show (obviously) much larger correlation requiring a higher number of simulations for stable estimation. For that reason, we limit the Metropolis-Hastings sampling process for $\boldsymbol{\beta}$ to this one illustration and will focus on the classical `mcmc_ridge()` implementation in the remaining parts of the report.

The upper panel in Figure 1 indicates a very good performance by all three estimation procedures in consideration. Further, all acceptance probabilities are in a reasonable range supporting a fast convergence of all Markov Chains.

It is important to remember, that each point in the plot only represents exactly one measurement. In order to make any conclusions about bias and variance of the different estimation models, the above procedure is repeated 50 times. The black points in Figure 2 represent the mean of these 50 Posterior Mean estimates. Since we cannot rely on distributional theory for the standard errors, the variability of the estimates is

Model Performance for different Predictor Correlation Structures

True coefficient values are indicated by grey circles

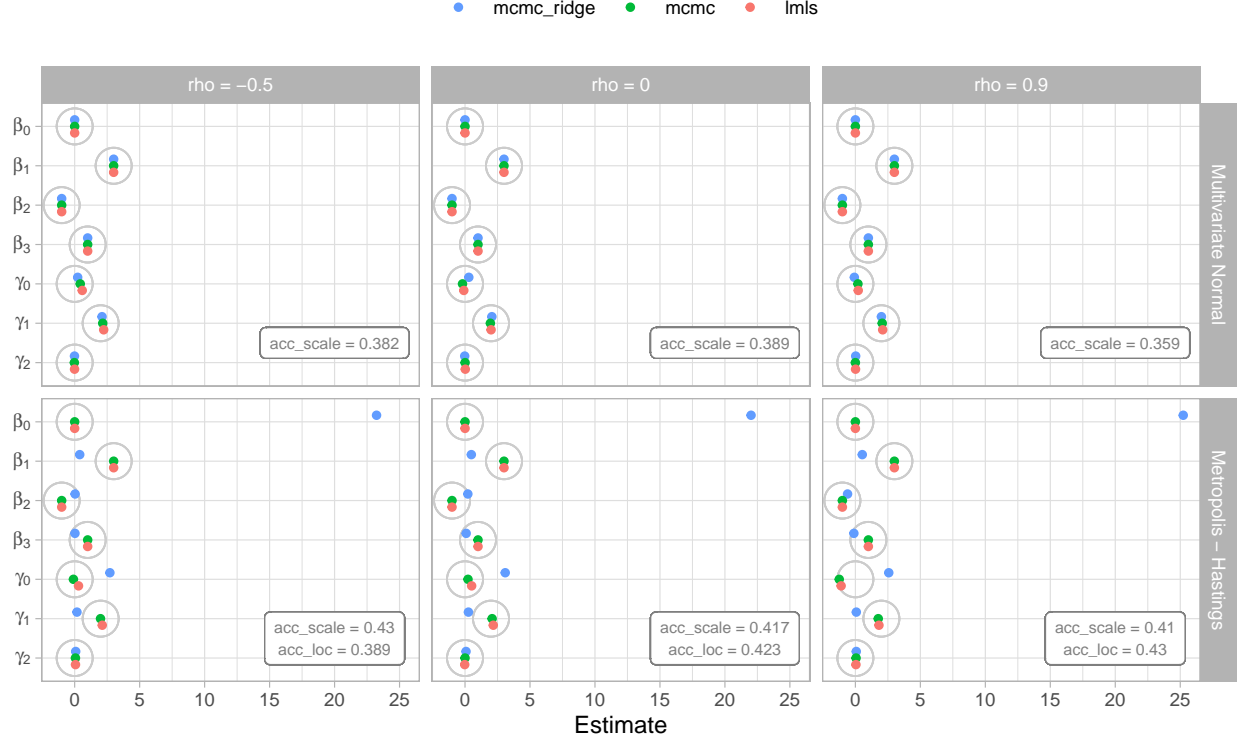


Figure 1: Comparison of Correlation Structures - One Simulation Cycle

displayed by nonparametric ‘confidence’ intervals, which are simply given by the range from the empirical 0.05 quantile to the 0.95 quantile of the 50 estimated values.

Further investigations have shown that the `mcmc_ridge()`, the `mcmc()` and the `lmls()` functions perform very similar for each correlation structure. For that reason only the results of the `mcmc_ridge()` sampler are shown in Figure 2.

There are three conclusions from this first simulation study:

1. The correlation structure does not have a significant impact of the sampling results. The three plot facets look almost identical.
2. The `mcmc_ridge()` sampler (as well as the `mcmc()` and `lmls()` functions) are very robust towards correlated data and perform extremely well. In particular, all three approaches (visually and numerically) provide close to unbiased estimates.
3. The variability among the β estimates is almost nonexistent, such that results from a single simulation cycle are already reliable and representative. While the estimates for the γ vector are still correct on average, the variability across different simulations is significant (particularly for γ_0). Thus, averaging the results from multiple repetitions of the sampling process is advisable.

2 Challenging the Model Assumptions

This simulation study is structured in a very similar way to the study considered in section 1. Instead of varying the correlation structure among the regressors in the underlying data set, both the regressors and the

Empirical 90% Confidence Intervals for Posterior Mean Estimates

True coefficient values are marked by grey circles

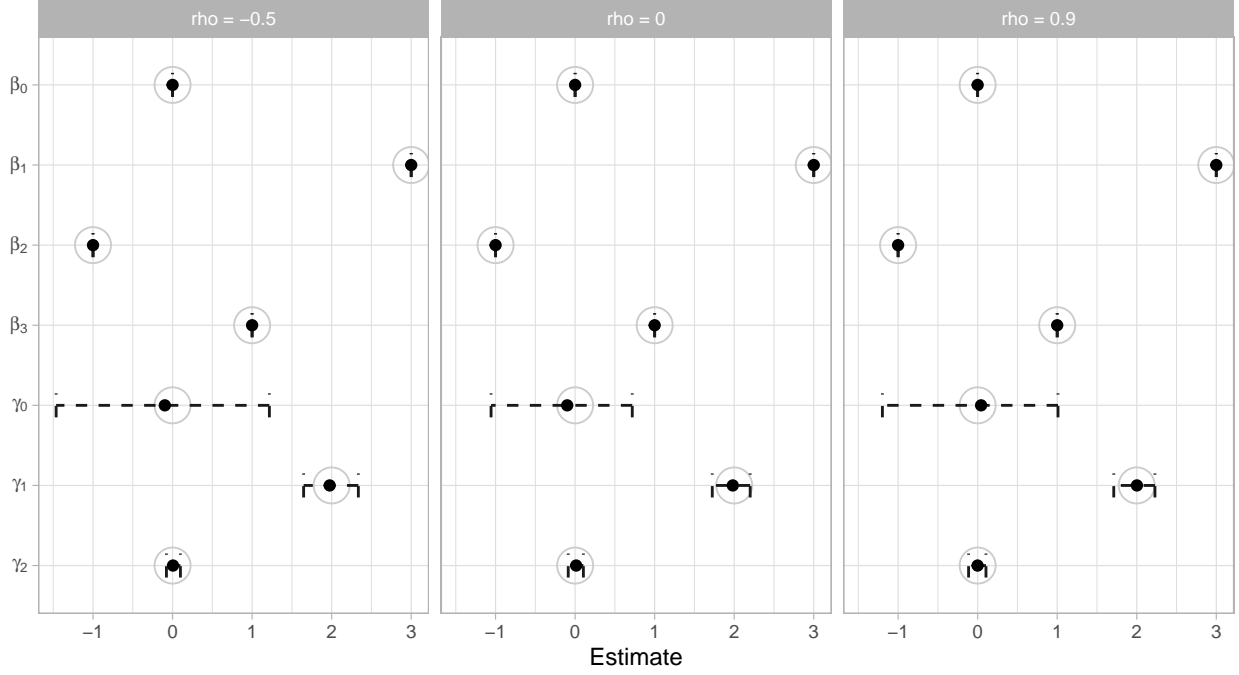


Figure 2: Comparison of Correlation Structures - 50 Simulation Cycles

outcome variable y are sampled from distributions that are more challenging for estimation than the normal distribution.

2.1 Simulation Setting

- The design matrix $\mathbf{X} = (\mathbf{1}_n \quad \mathbf{x}_1 \quad \mathbf{x}_2 \quad \mathbf{x}_3 \quad \mathbf{x}_4)$ contains four independently sampled regressor variables plus one intercept column:
 - $\mathbf{x}_1 \stackrel{iid}{\sim} \mathcal{N}(5, 16)$,
 - $\mathbf{x}_2 \stackrel{iid}{\sim} \text{Exp}(5)$,
 - $\mathbf{x}_3 \stackrel{iid}{\sim} \mathcal{U}([-2, 12])$,
 - $\mathbf{x}_4 \stackrel{iid}{\sim} \text{Ber}(0.3)$.
- The design matrix $\mathbf{Z} = (\mathbf{1}_n \quad \mathbf{x}_1 \quad \mathbf{x}_2 \quad \mathbf{z}_3)$ contains the additional regressor variable $\mathbf{z}_3 \stackrel{iid}{\sim} t_{10}$, which is independently sampled from all other columns.
- The true coefficient vectors are given by $\boldsymbol{\beta} = (\beta_0 \quad \beta_1 \quad \beta_2 \quad \beta_3 \quad \beta_4)^T = (0 \quad -3 \quad -1 \quad -1 \quad 2)^T$ and $\boldsymbol{\gamma} = (\gamma_0 \quad \gamma_1 \quad \gamma_2 \quad \gamma_3)^T = (0 \quad 1 \quad 2 \quad 3)^T$.
- Three different specifications for the outcome distribution were chosen:
 - $y_i \sim \mathcal{N}(\mu, \sigma^2)$,
 - $y_i \sim \mu + \left(\sigma \cdot \sqrt{\frac{3}{5}}\right) T$, where $T \sim t_5$,
 - $y_i \sim \mu + \sigma \cdot U$, where $U \sim \mathcal{U}([0, 1])$.

In order to isolate the impact of the different shapes of the three probability distributions, the mean $\mu = \mathbf{x}_i^T \boldsymbol{\beta}$ and the variance $\sigma^2 = \exp(\mathbf{z}_i^T \boldsymbol{\gamma})^2$ are held constant across the models.

Posterior Means / MLE for (misspecified) Regression Models

True coefficient values are marked by grey circles

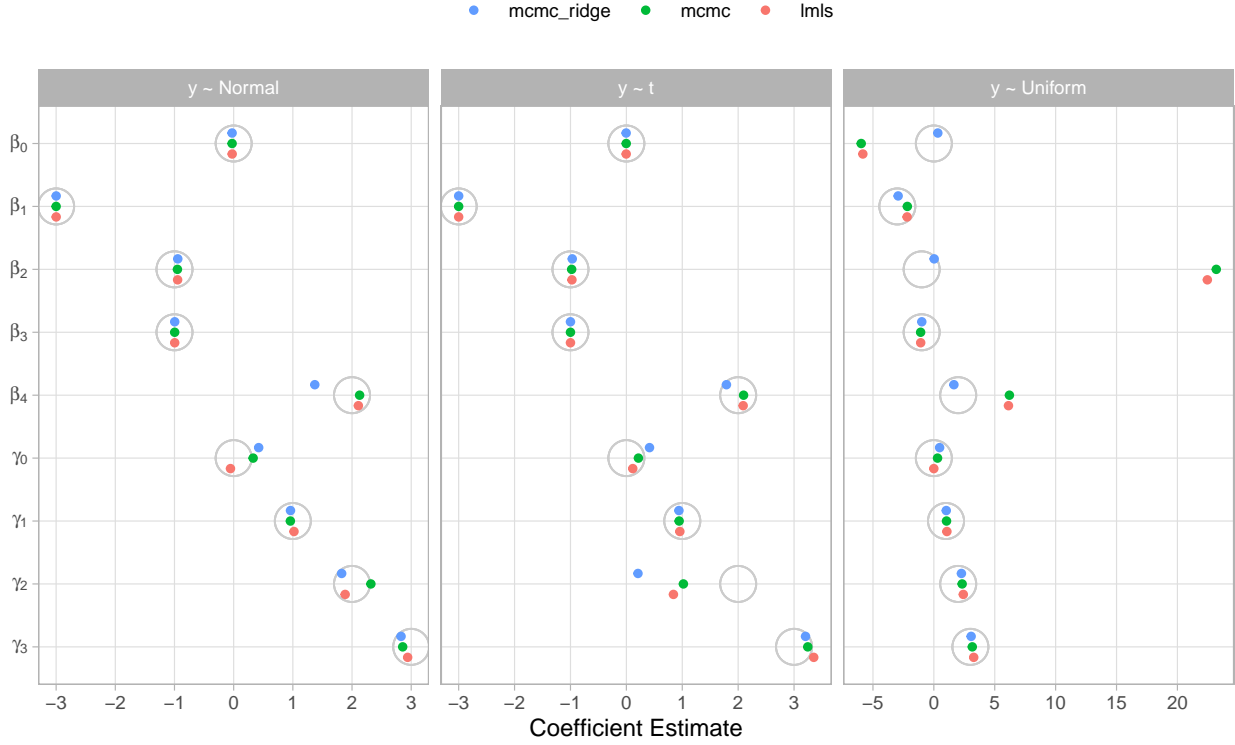


Figure 3: Comparison of Outcome Distributions - One Simulation Cycle

Note that the `lmls()`, `mcmc()` and `mcmc_ridge()` models are built upon the assumption $y_i \sim \mathcal{N}(\mu, \sigma^2)$. Hence, we expect all three estimation procedures to perform well under the first outcome specification, which they were designed for. The remaining two cases analyze the performance in presence of a mild (t distribution) and a moderately strong (uniform distribution) violation of this model assumption.

2.2 Simulation Results

Just as in section 1 the results of one complete simulation cycle (each of the $3 \cdot 3 = 9$ models was fitted once / each data point represents one estimate) are displayed in Figure 3. Note that the second facet is labeled by $y \sim t$, although it is formally sampled from an affine transformation of a t -distributed random variable, which does not follow an exact t distribution.

The differences within each facet as well as between the facets are significant. All three models seem to estimate the β vector well, when there are no or only mild violations of the normal assumption for y . If y is sampled from a uniform distribution, there are major differences for β_0 , β_2 and β_4 (notice the extended x -scale in the third facet). Interestingly, the γ vector is estimated very well in the latter case with more deviations in the setting, where y is based on the t distribution.

To gain insights beyond this single simulation cycle, which could well be disturbed by random noise, we repeat the sampling process 50 times. The resulting means as well as empirical confidence intervals (analogous to section 1) are plotted in Figure 4.

This plot (literally) paints a drastically different picture, emphasizing the necessity of repeating experiments multiple times whenever possible. Across all 50 simulations the deviation of the estimates for β_2 (corresponding to the regressor variable from the exponential distribution) is huge for all three distributional specifications

Empirical 90% Confidence Intervals for Posterior Mean Estimates

True coefficient values are marked by grey circles

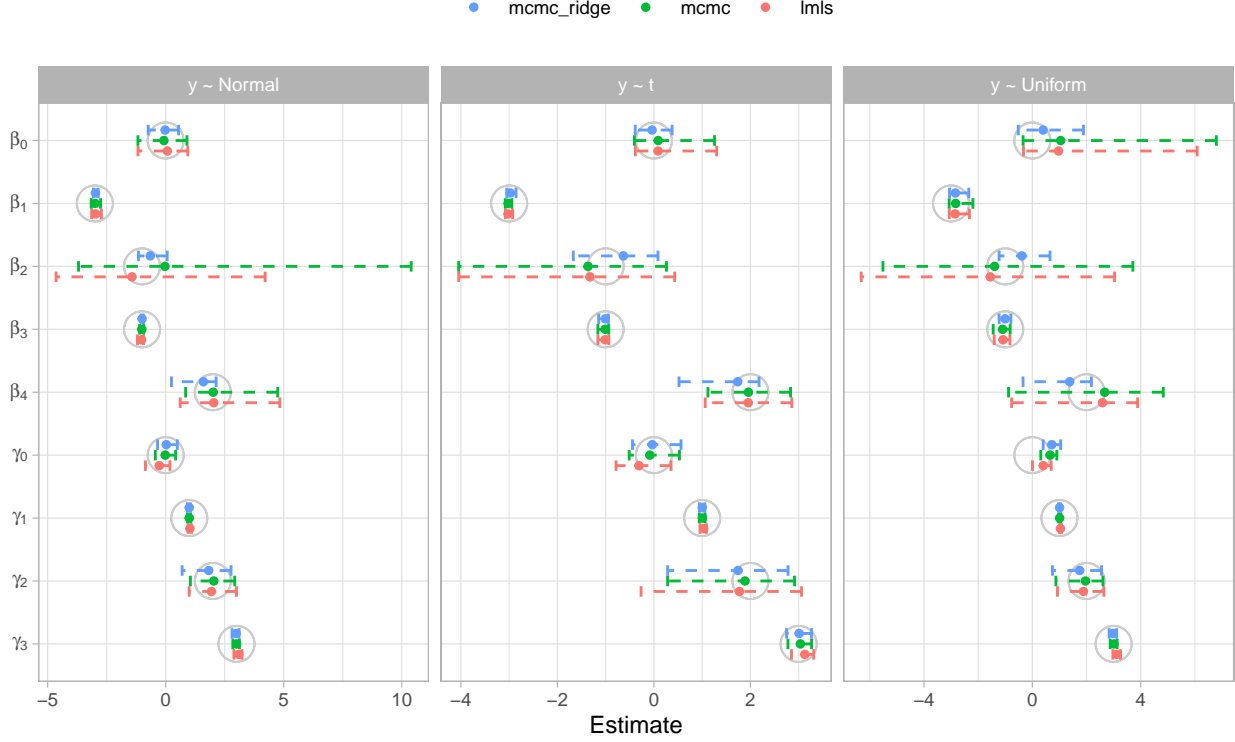


Figure 4: Comparison of Outcome Distributions - 50 Simulation Cycles

of y .

The small bias induced by all three models is negligible compared to the wide confidence intervals, which is particularly interesting, when y stems from a normal distribution. In this case all models should perform well, however the `lmls()` and the `mcmc()` Posterior Mean / Maximum Likelihood estimates vary wildly across the simulation cycles. A similar effect can be observed for β_0 in case of the uniform distribution. Here, all models overestimate the true value on average, while the `mcmc_ridge()` function again shows the smallest variability. In contrast, the estimates for γ_0 in the right facet are fairly stable across simulation cycles, but also consistently wrong at the same time.

Estimates of the bias and the standard error of the Posterior Mean / Maximum Likelihood estimates can be more distinctly compared by their numerical values provided in Tables 1 and 2. In order to emphasize the interesting/differing entries, both tables only include a subset of the estimated coefficients.

Considering the bias estimates in Table 1 first, there are no obvious patterns that would suggest the superiority of one model. Further, none of the three models tend to only over- or underestimate the true coefficient values. The most interesting entries are the bias estimates for β_0 and γ_0 in the uniform case, where all three models agree to significantly overestimate. However, the intercept coefficients are often of minor interest.

The standard error estimates displayed in Table 2 clearly indicate the worst performance of the `lmls()` and the `mcmc()` model for β_2 . In almost all cases (and sometimes very significantly), the `mcmc_ridge()` sampler has the smallest standard error. This finding nicely confirms the underlying mathematical theory: The present prior specifications in the Bayesian setting, which induces the equivalent form of a frequentist Ridge penalty, can lead to biased estimation.

However, this loss in accuracy can be (as it is in this case) dominated by the gain in precision by the shrinkage

Table 1: Bias of Coefficient Estimates

	Normal			t			Uniform		
	lmls	mcmc	mcmc_ridge	lmls	mcmc	mcmc_ridge	lmls	mcmc	mcmc_ridge
β_0	0.07	-0.07	-0.02	0.09	0.09	-0.04	0.97	1.05	0.40
β_2	-0.43	0.97	0.35	-0.32	-0.37	0.37	-0.55	-0.39	0.61
β_4	0.04	0.01	-0.40	-0.05	-0.04	-0.26	0.59	0.67	-0.62
γ_0	-0.27	-0.02	0.03	-0.31	-0.08	-0.03	0.40	0.66	0.72
γ_2	-0.06	0.04	-0.18	-0.23	-0.11	-0.26	-0.11	-0.04	-0.25

Table 2: Standard Errors of Coefficient Estimates

	Normal			t			Uniform		
	lmls	mcmc	mcmc_ridge	lmls	mcmc	mcmc_ridge	lmls	mcmc	mcmc_ridge
β_0	0.83	0.69	0.47	0.62	0.62	0.49	2.15	2.37	1.47
β_2	7.11	4.41	0.40	2.54	2.58	0.54	6.48	6.48	0.67
β_4	1.32	1.33	0.68	0.57	0.56	0.53	4.10	4.30	0.81
γ_0	0.33	0.28	0.27	0.35	0.33	0.34	0.22	0.19	0.21
γ_2	0.78	0.65	0.62	1.01	0.93	0.89	0.55	0.54	0.56

effect of the penalty. Note that (except for γ_0 in the most right facet in Figure 4) the `mcmc_ridge()` sampler slightly *overestimates* coefficients with true *negative* values and *underestimates* those with true *positive* values. This again is caused by the Ridge penalty leading to estimated coefficients close to zero.

In summary, the following conclusions can be drawn:

1. All three models are affected by changes in the regressor and/or outcome distributions. In the former case the regressor variables sampled from the Exponential and the Bernoulli distributions were the greatest challenge, in the latter case the outcome variable from the Uniform distribution. This is generally not surprising, since these distributions deviate most from the nicely behaved normally distributed case.
2. As expected, the `lmls()` function is affected strongly by violating the model assumptions, since its estimation process is based on the normal (log) Likelihood. Surprisingly, the `mcmc()` sampler without penalty often did not perform much better.
3. While the `mcmc_ridge()` function does not excel at estimation accuracy, it does lead to the most stable estimation with smallest standard errors in the vast majority of cases. As emphasized above, this behaviour nicely agrees with the mathematical theory of Ridge penalization.
4. Had we not conducted repeated experiments, our conclusions would have been quite different. Simulation results are therefore always worth repeating many times to consolidate the correct interpretations.

2.3 Technical Aspects

As outlined in the previous paragraph, a total of $50 \cdot 3 \cdot 3 = 450$ models were fitted to analyze the performance differences. In order to speed up the involved computations of this specific and some of the other simulation studies in this report, we used the *parallel computing* capabilities of R.

There are many options from various packages to choose from. We decided to use the `furrr` package, which is built on top of the `future` package specialized on parallel processing. As the name suggests, `furrr` provides a convenient way to use many functions from the popular `purrr` package, while using multiple cores at the same time. This *functional programming* based approach (similar to the `apply()` family in ‘base R’) is

particularly well suited for simulation studies and provides some structural as well as minor performance advantages compared to the classical `for`-loop approach.

The following (slightly modified) code snippet provides a brief insight into the implementation:

```
plan(multisession, workers = 8)

full_results <- tibble(id = 1:50) %>%
  mutate(samples = future_map(
    .x = id,
    .f = ~ show_results(n = 50, num_sim = 1000),
    .options = furrr_options(seed = 1)
  ))
```

The `plan()` function borrowed from the `future` package initializes the parallel computing process and determines the number of cores/workers available for computation. The `show_results()` helper function fits all three models `mcmc_ridge()`, `mcmc()` and `lmls()` for each outcome distribution in a single simulation cycle.

This entire procedure is repeated 50 times in parallel using the `future_map()` function from the `furrr` package, where the results of all 450 models are saved in a well organized structure inside of a list column. This new column of the data frame contains complete information about all simulations, such that any required element for the further analysis can be easily extracted and post processed.

Finally, the `.options()` argument allows the specification of a random seed. Random number generation in the context of parallel computing is slightly more involved compared to the sequential approach. This additional complexity is automatically handled by the `future_map()` function, such that all results are sampled in a statistically valid and fully reproducible manner.

3 Sample Size

This simulation study analyzes the effect of the sample size n on the means of the posterior distribution for the coefficients of β and γ . There are two main goals of this simulation study: On the one hand, we want to investigate whether the posterior means of large samples are closer to the true values than the posterior means of small samples. On the other hand, we want to analyze whether the `mcmc_ridge()` penalty affects the location of the posterior means.

3.1 Simulation Setting

- The design matrix $\mathbf{X} = (\mathbf{1}_n \quad \mathbf{x}_1 \quad \mathbf{x}_2)$ contains two independently sampled regressor variables plus one intercept column:
 - $\mathbf{x}_1 \stackrel{iid}{\sim} \mathcal{N}(1, 1)$,
 - $\mathbf{x}_2 \stackrel{iid}{\sim} \mathcal{N}(2, 1)$.
- The design matrix $\mathbf{Z} = (\mathbf{1}_n \quad \mathbf{z}_1 \quad \mathbf{z}_2)$ is structured in the same way with the regressor variables:
 - $\mathbf{z}_1 \stackrel{iid}{\sim} \mathcal{N}(1, 1)$,
 - $\mathbf{z}_2 \stackrel{iid}{\sim} \mathcal{N}(2, 1)$.
- The true coefficient vectors are given by $\beta = (\beta_0 \quad \beta_1 \quad \beta_2)^T = (1 \quad -1 \quad 4)^T$ and $\gamma = (\gamma_0 \quad \gamma_1 \quad \gamma_2)^T = (0 \quad -0.5 \quad 1)^T$.
- The posterior means are analyzed with respect to 6 different sample sizes: $n \in \{0, 50, 100, 200, 300, 500\}$.
- In the next step, the outcome vector $y \in \mathbb{R}^n$ is simulated and passed to the `mcmc_ridge()` function with `nsim = 500` simulations.

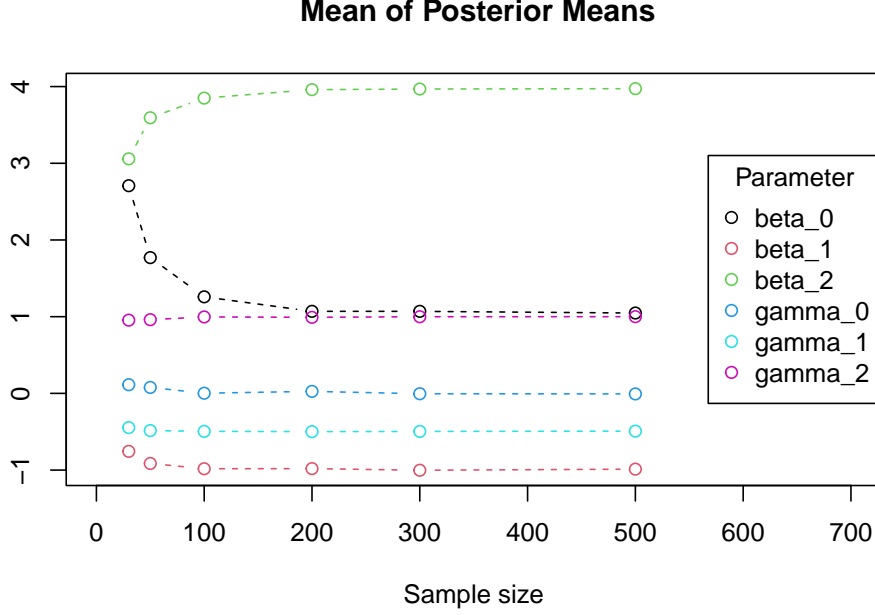


Figure 5: Mean value of 100 Posterior Mean Estimates

- To make the results more stable, the above procedure is repeated 100 times. For each coefficient, the mean value of the Posterior Mean estimates of each coefficient is calculated as well as the Mean Absolute Error (*MAE*) with respect to the true values of β and γ .

3.2 Simulation Results

The means of the Posterior Mean estimates are displayed in Figure 5. For larger sample sizes ($n \geq 200$) none of the six parameters are extremely biased.

Moreover, for $n = 30$, β_0 and β_2 are significantly biased, which might be caused by the high `mcmc_ridge()` penalty for $\beta_2 = 4$. The significant bias of β_0 might be explained by a counteract of the β_2 bias.

After getting an impression about empirical biases of the coefficients, we now focus on the variability of the posterior means of the coefficients, which are measured by the *MAE* based on the results of the 100 repetitions. Figure 6 points out that the posterior means of β_0 have significantly larger errors than the posterior means of β_2 for $n = 30$. However, this might also be explained by the fact that for $n = 30$, β_0 has a greater empirical bias than β_2 as could be observed in Figure 5.

In addition, for increasing sample sizes, the *MAE* of the Posterior Means tend to zero for all coefficients except β_0 . Nevertheless, also the errors of β_0 seem to become smaller with increasing sample size.

4 Number of Simulations

Another important model input is `nsim`, the number of Markov Chain Monte Carlo simulations within the `mcmc_ridge()` function.

As explained in our first report, we chose the proposal density for the Metropolis-Hastings algorithm of γ in order to achieve acceptance rates between 35% und 50%. A higher number of `mcmc_ridge()` simulations increases the absolute number of accepted proposals and might deliver more precise approximations of the true posterior distributions, but also increases the computational cost such that there is a trade-off between precision and speed.

This simulation study analyzes the effect of the `nsim` parameter on the posterior distribution.

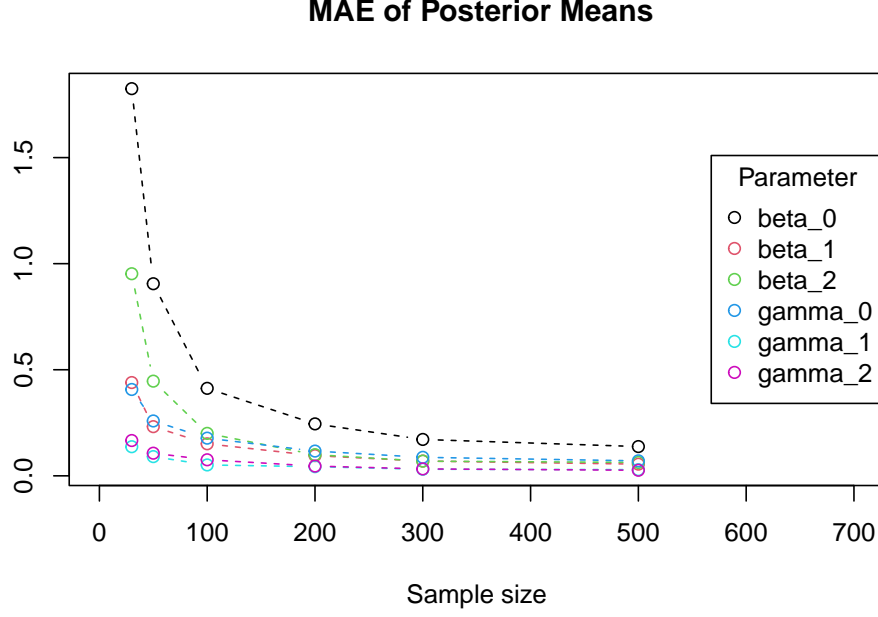


Figure 6: Mean Absolute Error Estimates

4.1 Simulation Setting

- The design matrix $\mathbf{X} = (\mathbf{1}_n \quad \mathbf{x}_1 \quad \mathbf{x}_2)$ contains two independently sampled regressor variables plus one intercept column. The sample size n is chosen to be 100.
 - $\mathbf{x}_1 \stackrel{iid}{\sim} \mathcal{N}(1, 1)$.
 - $\mathbf{x}_2 \stackrel{iid}{\sim} \mathcal{N}(2, 1)$.
- The design matrix $\mathbf{Z} = (\mathbf{1}_n \quad \mathbf{z}_1 \quad \mathbf{z}_2)$ similarly contains two independently sampled regressor variables plus one intercept column. The sample size n is chosen to be 100 as well.
 - $\mathbf{z}_1 \stackrel{iid}{\sim} \mathcal{N}(1, 1)$.
 - $\mathbf{z}_2 \stackrel{iid}{\sim} \mathcal{N}(2, 1)$.
- The true coefficient vectors are given by $\boldsymbol{\beta} = (\beta_0 \quad \beta_1 \quad \beta_2)^T = (1 \quad -1 \quad 4)^T$ and $\boldsymbol{\gamma} = (\gamma_0 \quad \gamma_1 \quad \gamma_2)^T = (0 \quad -0.5 \quad 0.5)^T$.
- In the next step, the outcome vector $y \in \mathbb{R}^n$ is simulated according to the correct normality assumption and passed to the `mcmc_ridge()` function.
- The posterior means are analyzed with respect to 4 different values for $\text{nsim} \in \{100, 300, 500, 1000\}$.
- Similar as in the simulation study concerning the sample size, the above procedure is repeated 100 times to make the results more stable. For each coefficient, the mean value of the 100 Posterior Means is calculated as well as the mean absolute error (*MAE*) with respect to the true values of $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$. In addition, the variances *within* the Markov Chains are analyzed.

4.2 Simulation Results

Similar as in the sample size simulation study, Figure 7 contains the mean values of the Posterior Mean estimates for the different `nsim` specifications. The result is quite clear: In the range between `nsim` = 100 and `nsim` = 1000, the `nsim` parameter has no significant impact on the mean of the posterior means.

Furthermore, for the chosen sample size $n = 100$, none of the 6 coefficients has a large bias. However, the

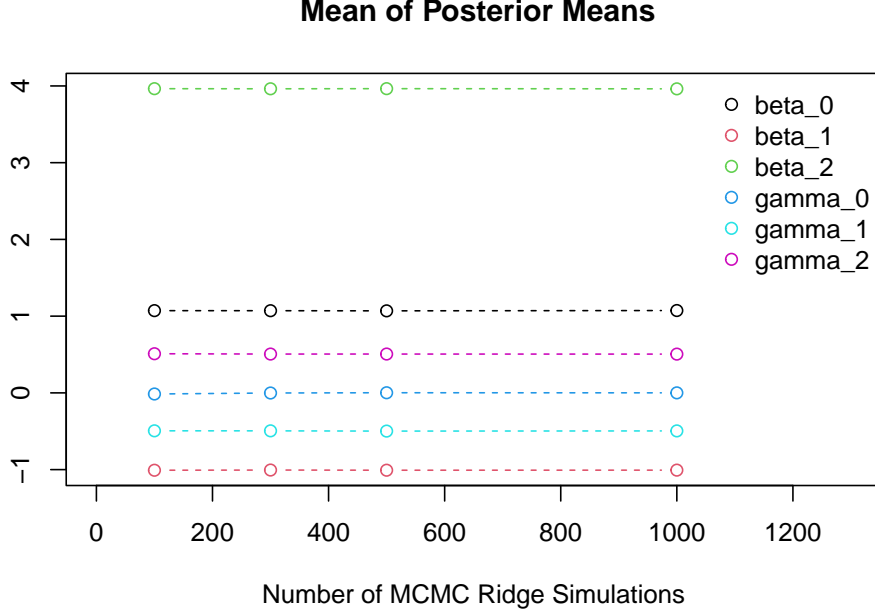


Figure 7: Mean value of 100 Posterior Mean Estimates

estimates of β_0 are all close to 1.07, whereas the estimates of β_2 are all close to 3.96. Thus, we obtain small biases for β_0 and β_2 , which again might be a result of the high `mcmc_ridge()` penalty for $\beta_2 = 4$ (similar as discussed in section 3).

We also analyzed the *MAE* for different values of `nsim`. Figure 8 provides interesting results for the errors of the β and γ coefficients: On the one hand, `nsim` seems not to have any impact on the errors of the β coefficients. But on the other hand, an increase of `nsim` leads to more precise γ estimates and apparently lower *MAE* values for all γ coefficients.

In the last step, we analyzed whether the value of `nsim` has any impact on the posterior variance within the Markov Chains. The results are displayed in Figure 9. Again, it can be observed that the `nsim` parameter has no impact on the variances within the β samples of the Markov chain, whereas the variance within the γ samples increases with increasing value of `nsim`.

This result is not really surprising, since the Markov Chain typically jumps to points which are quite close to the current location. As a consequence, if the number of jumps increases, also the variance of the samples might increase.

In combination with the lower *MAE* for the γ parameters, one can conclude that the approximated posterior distributions of the γ coefficients might be squeezed for small values of `nsim` and converge to the true posterior distributions if the number of simulations increases.

5 Hyperparameters

In the past, we have been sampling data with the `mcmc_ridge()` function without having a closer look on the effect of the hyperparameters and model inputs `a_tau`, `b_tau`, `a_xi` and `b_xi`. However, they affect the full conditionals of τ^2 and ξ^2 , which follow an Inverse Gamma distribution, as stated in the mathematical model in the first report and repeated here for convenience:

$$\begin{aligned}\tau^2 \mid \cdot &\sim IG(a_\tau + \frac{K}{2}, b_\tau + \frac{1}{2}\tilde{\beta}^T \tilde{\beta}), \\ \xi^2 \mid \cdot &\sim IG(a_\xi + \frac{J}{2}, b_\xi + \frac{1}{2}\tilde{\gamma}^T \tilde{\gamma}).\end{aligned}$$

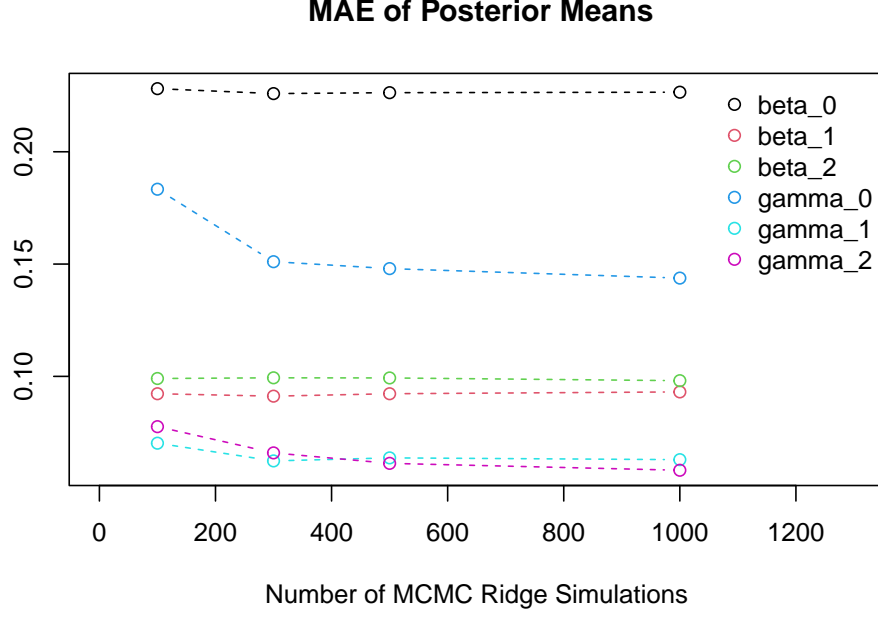


Figure 8: Mean Absolute Error Estimates

In a next step, the mean and variance of the full conditional multivariate normal distribution of β are assumed to be positively affected by τ^2 :

$$\Sigma_\beta = \left(\mathbf{W}^T \mathbf{W} + \frac{1}{\tau^2} \begin{pmatrix} 0 & 0 \\ 0 & \mathbf{I}_K \end{pmatrix} \right)^{-1} \quad \text{and} \quad \mu_\beta = \Sigma_\beta \mathbf{W}^T \mathbf{u},$$

with

$$\mathbf{w}_i := \frac{\mathbf{x}_i}{\exp(\mathbf{z}_i^T \boldsymbol{\gamma})} \in \mathbb{R}^{K+1}, \quad \mathbf{W} := \begin{pmatrix} \mathbf{w}_1^T \\ \vdots \\ \mathbf{w}_n^T \end{pmatrix} \in \mathbb{R}^{n \times (K+1)}$$

and

$$u_i := \frac{y_i}{\exp(\mathbf{z}_i^T \boldsymbol{\gamma})} \in \mathbb{R}, \quad \mathbf{u} := \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} \in \mathbb{R}^n,$$

while ξ^2 directly affects the full conditional of $\boldsymbol{\gamma}$, which is sampled by the Metropolis Hastings algorithm:

$$\begin{aligned} f(\boldsymbol{\gamma} \mid \cdot) &\propto \exp \left(-\frac{1}{2} \cdot \left[\frac{1}{\xi^2} \tilde{\boldsymbol{\gamma}}^T \tilde{\boldsymbol{\gamma}} + \sum_{i=1}^n \left(\frac{1}{\exp(\mathbf{z}_i^T \boldsymbol{\gamma})^2} (y_i - \mathbf{x}_i^T \boldsymbol{\beta})^2 + 2 \cdot \mathbf{z}_i^T \boldsymbol{\gamma} \right) \right] \right) \\ &= \exp \left(-\frac{1}{2} \cdot \left[\frac{1}{\xi^2} \tilde{\boldsymbol{\gamma}}^T \tilde{\boldsymbol{\gamma}} + 2 \cdot \mathbf{1}_n^T \mathbf{Z} \boldsymbol{\gamma} + \sum_{i=1}^n \left(\frac{y_i - \mathbf{x}_i^T \boldsymbol{\beta}}{\exp(\mathbf{z}_i^T \boldsymbol{\gamma})} \right)^2 \right] \right). \end{aligned}$$

Thus, data generated with different hyperparameter specifications will inevitable impact all estimated coefficients.

5.1 Simulation Setting

- The design matrix $\mathbf{X} = (\mathbf{x}_1 \quad \mathbf{x}_2)$ is simulated from a two dimensional normal distribution $\mathcal{N}_2(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ with mean vector $\boldsymbol{\mu} = (1 \quad 2)^T$ and identity covariance matrix $\boldsymbol{\Sigma} = \mathbf{I}_2$. The same holds true for the design matrix $\mathbf{Z} = (\mathbf{z}_1 \quad \mathbf{z}_2)$ with mean vector $\boldsymbol{\mu} = (5 \quad 3)^T$ and identity covariance matrix.

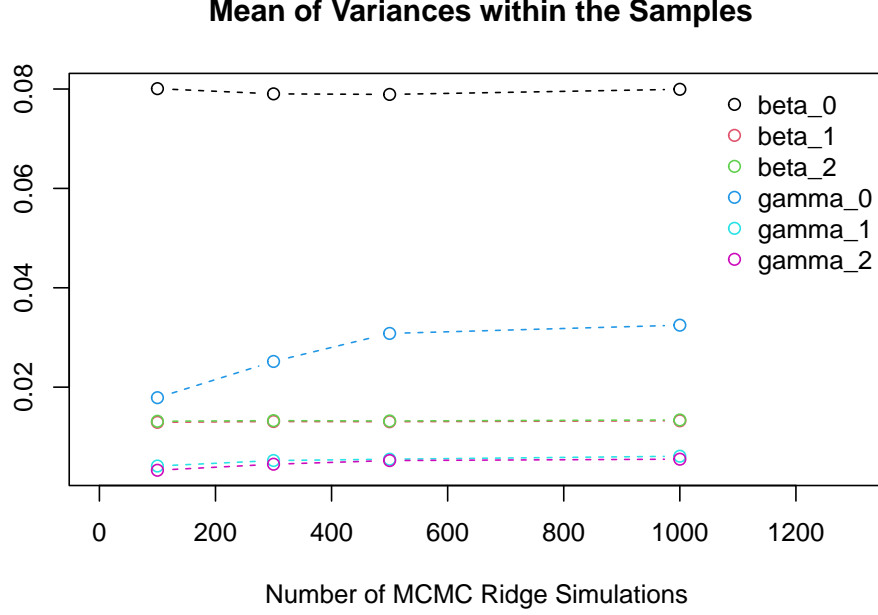


Figure 9: Mean value of 100 Posterior Variance Estimates

- In both design matrices intercept columns are added for estimation purposes. The true coefficient vectors are given by $\beta = (\beta_0 \ \beta_1 \ \beta_2)^T = (0 \ -1 \ 4)^T$ and $\gamma = (\gamma_0 \ \gamma_1 \ \gamma_2)^T = (0 \ -2 \ 1)^T$.
- For sampling the location parameter, the full conditional multivariate normal distribution of β is chosen, i.e. `mcmc_ridge(..., mh_location = FALSE)` is used. Therefore, the location estimate is directly affected by the hyperparameters.
- For simulating the influence of the hyperparameters, nine different values are chosen: $a_\tau, b_\tau, a_\xi, b_{xi} \in \{-1, 0, 0.5, 1, 2, 10, 50, 100, 200\}$. Since for statistical properties like the mean of an Inverse Gamma distribution $\frac{b}{a-1}$ the condition $a > 1$ is required, particular attention is given to larger values. However, it is an aim to inspect the performance of the sampler for smaller hyperparameter values than 1 as well.

5.2 Simulation Results

The first two plots of Figures 10 and 11 display the absolute deviations of the Posterior Mean estimates from the true parameters with the stated different values for a_τ and b_τ . For each estimate, the Posterior Mean averages over 1000 simulations of the `mcmc_ridge()` sampler. Note, that location and scale parameters are plotted separately, according to the relationship mentioned above. For a better overview, the dotted line displays the linear trend of all estimate deviations.

The x - axis is transformed by a pseudo logarithm in order to clearly visualize the deviations in the range of -1 to 10 , which would not be possible on original scales. Since -1 and 0 are also part of the hyperparameter values, the `pseudo_log_trans()` function of the `scales` package is applied, log-transforming positive values only.

It can be observed, that the intercept estimates in each plot show the largest deviations from their true value. In Figure 10, however, the overall deviations of β estimates from their corresponding true value are small in absolute value. In contrast, deviations of the γ estimates in Figure 11 are fairly significant, especially for γ_0 .

The functional chain that applies to the estimates of β can be described by the effect of the mean of the inverse gamma distribution on τ^2 : A larger value for b_τ leads to larger values of τ^2 , which are again affecting the full posterior parameters of β and, thus, potentially increase the absolute deviation of the corresponding estimates from their true values. a_τ causes the opposite effect. This numerically observable effect, however, is hid by the overall small deviation in the first two plots of Figure 10.

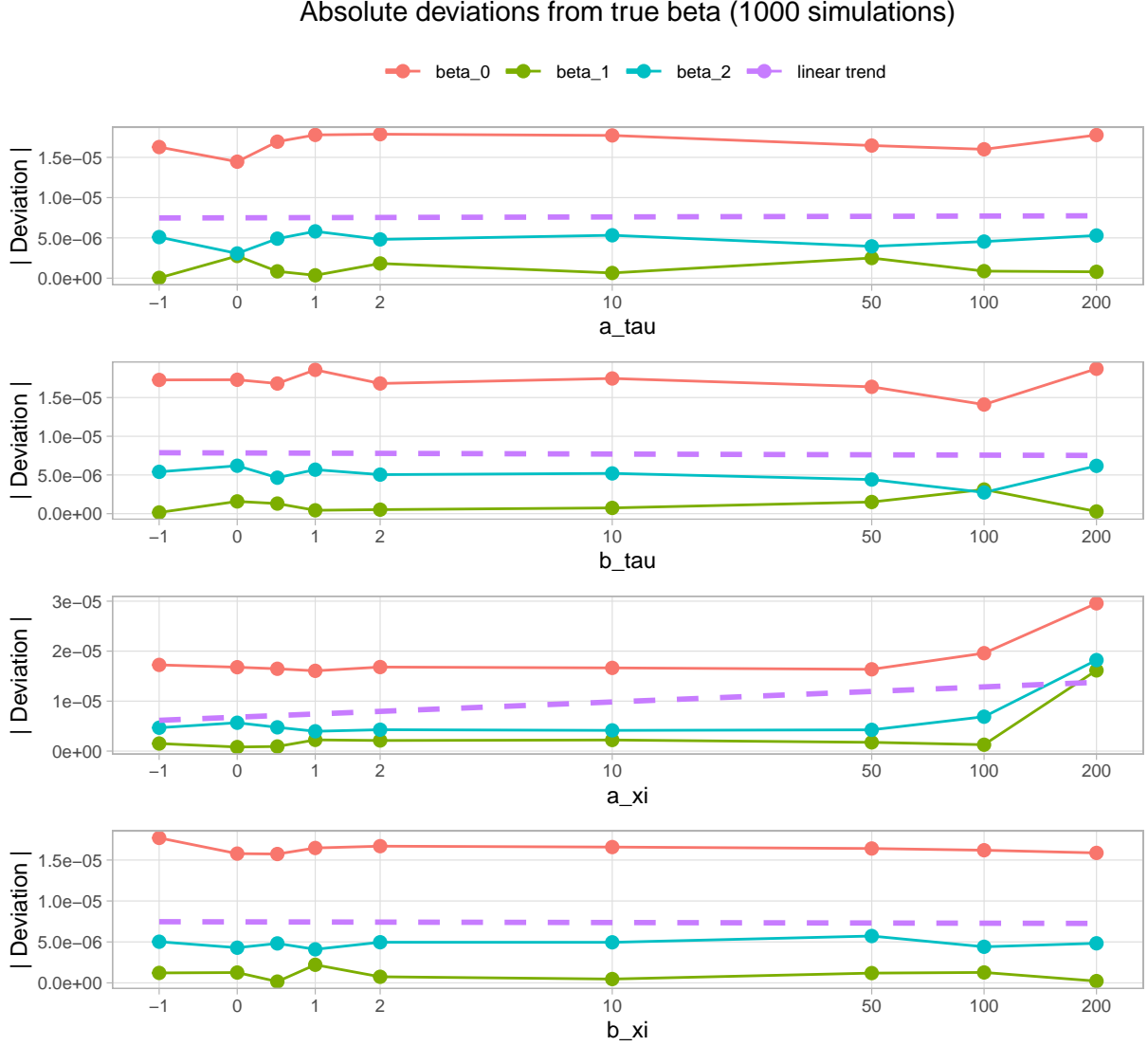


Figure 10: Comparison of the absolute deviations of beta parameters

It is remarkable, that the deviation of β estimates is smallest when $a_\tau, b_\tau \in \{50, 100\}$. For values of $a_\tau \leq 1$, one obtains wider variances of absolute deviations, since the Posterior Mean requires values larger than one.

In the upper two plots of Figure 11, there is no clear impact of τ^2 and its parameters. Rooted in no direct effect of τ^2 on γ according to our underlying mathematical model, one observes cross-effects through the sampling procedure of the `mcmc_ridge()` sampler, where the full posterior $f(\gamma \mid \cdot)$ depends on β . Anyway, our sampler produces the lowest deviation of γ estimates for $a_\tau, b_\tau \in \{0.5, 200\}$, where 0.5 is chosen by coincidence here, since wide variations for $a_\tau \leq 1$ of absolute deviations are observable again.

The lower two plots of Figures 10 and 11 are constructed analogously, but showing the impact of a_ξ and b_ξ on the location and scale parameters respectively. Again, the overall absolute deviations for the β estimates from their true values are small, whereas the deviations for the γ estimates are considerably larger. Once again, the intercept estimates display the largest deviations from their true value.

Arguing with the mean of the Inverse Gamma distribution of ξ^2 in a similar way, one obtains larger mean values for b_ξ , while a_ξ lowers them. The impact of ξ^2 on γ is assumed to decrease $f(\gamma \mid \cdot)$ according to our underlying theoretical model. This effect is indicated by the linear trend lines in the second half of Figure 11.

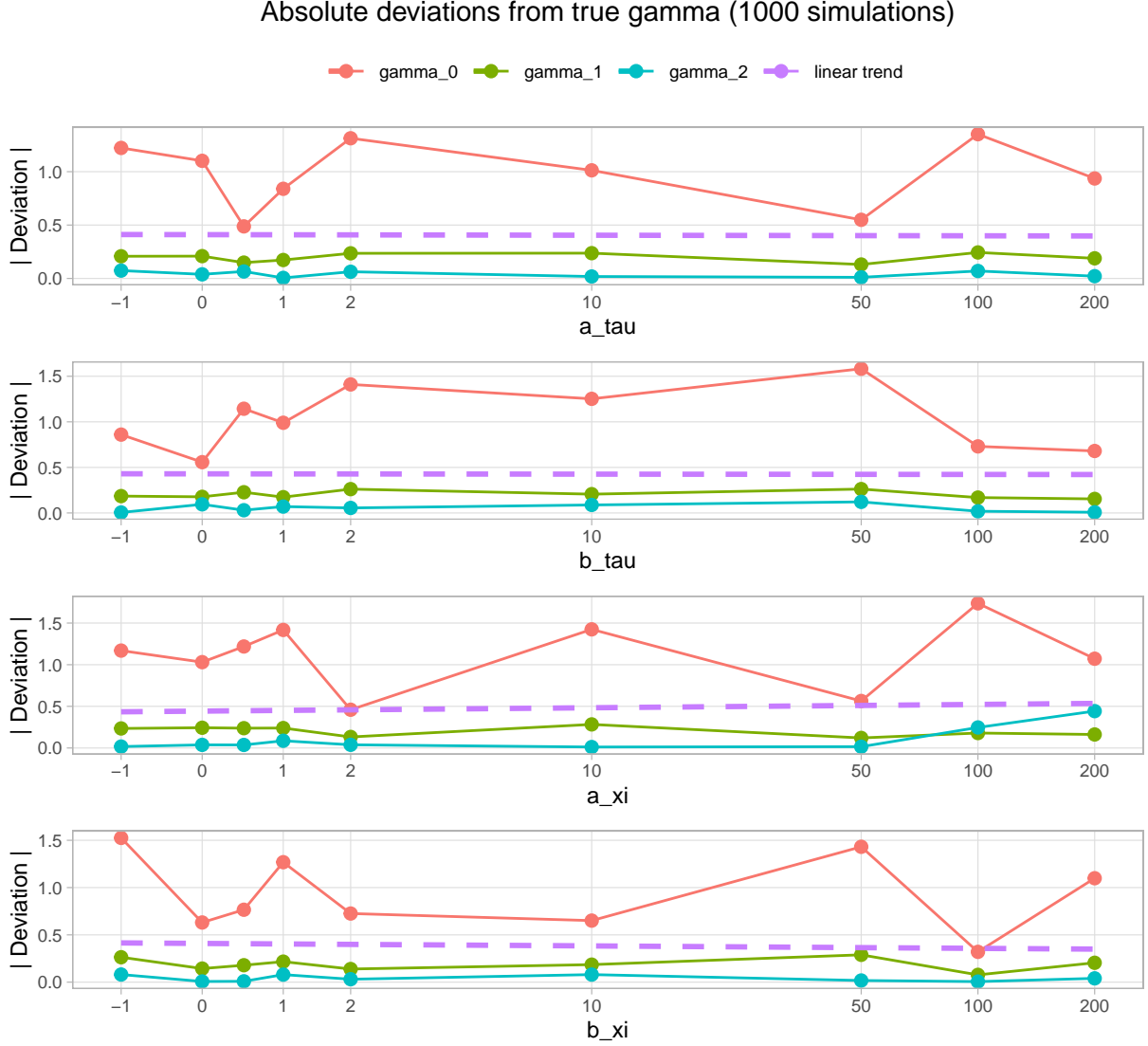


Figure 11: Comparison of the absolute deviations of gamma parameters

In general, one obtains smaller deviations for larger values of b_ξ and lower ones of a_ξ , where especially lots of randomness occurs in the deviations of γ_0 . Therefore, the impact of a_ξ and b_ξ on the scale intercepts is overshadowed by the randomness induced by the Metropolis Hastings algorithm. The same wide variations exclusively for $a_\xi \leq 1$ cannot be obtained in the same manner as for a_τ .

The sampler exhibits the best results for the *scale* estimates for $a_\xi = 2$ and $b_\xi = 100$. Though, due to the wide overall variation, these results must be taken with care.

The effect of a_ξ and b_ξ on β can be explained through the cross-effects of the matrix \mathbf{W} and the vector \mathbf{u} introduced at the beginning of this section, both containing γ . These diminish with increasing values of the γ entries.

The matrix \mathbf{W} affects the variance of the full conditional distribution of β negatively, while the mean is positively affected. Hence, larger values of a_ξ cause higher Posterior Means of the location parameters. The positive linear trend in the second half of Figure 10 for values of a_ξ is particularly interesting. For values of b_ξ , the trend comes off inferior. The wider variations of deviations for $a_\xi \leq 1$ is again not observable here. Nonetheless, the randomness observable for scale estimates does not show up for location estimates anymore.

The smallest deviations of the *location* estimates can be detected for $a_\xi = 1$ and $b_\xi = 200$.

Shortly noted, the acceptance rates of the Metropolis Hastings algorithm for sampling γ are always between 0.31 and 0.53. For the value range of a_τ , b_τ and a_ξ , no distinct pattern is observable in this regard. With growing values of b_ξ , however, acceptance rates are more likely to grow. Since the acceptance rates are in reasonable ranges enabling statistically valid estimation, these results are not further investigated here.

Next Steps

In the process of writing the first and second report, the main work of developing both the mathematical foundation as well as the code base of the `asp21bridge` package has been done. Thus, the following weeks will focus on filling remaining gaps and working out details that were left open due to time constraints. This includes taking a closer look at code efficiency and possible code refactoring for the sake of modularity.

Further, we will reach out to other groups with similar topics (the Bayesian Lasso Regression group in particular) in order to initiate a collaboration. In addition to exploring similarities and differences between our own sampler and the procedures from the `lmls` package, this would allow for an interesting comparison between different penalization approaches.

The final step consists of combining all pieces of the project into a single structured and consistent final report. Aside of the work shown in the first two reports, some new elements like a brief discussion of design and implementation choices during the development stage will be added with the aim of providing the reader a solid understanding of the underlying ideas as well as the practical usage of the package.