Fixing posteriors

August Arnstad

24.11.2023

Contents

GENERAL SETUP

First, we set up the necessary libraries and configure the environment for our analysis. This includes loading essential packages and setting options for chunk output and plot dimensions."

```
## Loading required package: usethis
```

INSTALLING THE PACKAGE

This section ensures the devtools package is installed, which is required for installing packages from GitHub. We then install the BayesianImportance package directly from GitHub using devtools::install_github(). In the package under the Hello.R file, all functions are defined with corresponding documentation.

```
# If not already installed, install the 'devtools' package
if (!require(devtools)) install.packages("devtools")

# Install BayesianImportance
devtools::install_github("AugustArnstad/BayesianImportance")
```

SIMULATE DATA

In this part, we simulate data to demonstrate the functionality of the BayesianImportance package. We generate random variables with different correlation structures, random effects, and an error term. The data is then structured into data frames for further analysis. If you have a suitable dataset you can use this instead.

```
library(BayesianImportance)
library(INLA)
library(mnormt)
library(ggplot2)
library(reshape2)
library(RColorBrewer)
set.seed(1234)

n <- 10000
```

```r
nclass_gamma <- 200
nclass_eta <- 100

mu <- c(1, 2, 3)

sigma1 <- matrix(c(1, 0, 0, 0, 1, 0, 0, 0, 1), 3, 3)
sigma2 <- matrix(c(1, 0.1, 0.1, 0.1, 1, 0.1, 0.1, 0.1, 1), 3, 3)
sigma3 <- matrix(c(1, 0.5, 0.5, 0.5, 1, 0.5, 0.5, 0.5, 1), 3, 3)
sigma4 <- matrix(c(1, 0.9, 0.9, 0.9, 1, 0.9, 0.9, 0.9, 1), 3, 3)

# Sample a standardized correlated design matrix
X1 <- rmnorm(n, mu, sigma1)
X2 <- rmnorm(n, mu, sigma2)
X3 <- rmnorm(n, mu, sigma3)
X4 <- rmnorm(n, mu, sigma4)

# Add random effects
gamma <- rep(rnorm(nclass_gamma, 0, sqrt(1)), each = n/nclass_gamma)
# eta <- rep(rnorm(nclass_eta, 0, sqrt(1)), each=n/nclass_eta)
epsilon = rnorm(n, mean = 0, sd = sqrt(1))

# Define some formula
Y1 <- 1 + 1 * X1[, 1] + sqrt(2) * X1[, 2] + sqrt(3) * X1[, 3] + gamma +
    epsilon  # + eta
Y2 <- 1 + 1 * X2[, 1] + sqrt(2) * X2[, 2] + sqrt(3) * X2[, 3] + gamma +
    epsilon  # + eta
Y3 <- 1 + 1 * X3[, 1] + sqrt(2) * X3[, 2] + sqrt(3) * X3[, 3] + gamma +
    epsilon  # + eta
Y4 <- 1 + 1 * X4[, 1] + sqrt(2) * X4[, 2] + sqrt(3) * X4[, 3] + gamma +
    epsilon  # + eta

# Collect as a dataframe
data_bayes1 = data.frame(cbind(Y1, X = X1))
data_bayes1 = data.frame(cbind(data_bayes1, gamma = gamma))
# data_bayes1 = data.frame(cbind(data_bayes1, eta=eta)) data_bayes1

data_bayes2 = data.frame(cbind(Y2, X = X2))
data_bayes2 = data.frame(cbind(data_bayes2, gamma = gamma))
# data_bayes2 = data.frame(cbind(data_bayes, eta=eta)) data_bayes2

data_bayes3 = data.frame(cbind(Y3, X = X3))
data_bayes3 = data.frame(cbind(data_bayes3, gamma = gamma))
# data_bayes = data.frame(cbind(data_bayes, eta=eta)) data_bayes3

data_bayes4 = data.frame(cbind(Y4, X = X4))
data_bayes4 = data.frame(cbind(data_bayes4, gamma = gamma))
# data_bayes = data.frame(cbind(data_bayes, eta=eta)) data_bayes4
```

USAGE

Here we demonstrate the usage of the BayesianImportance package. We fit Bayesian models and sample posterior distributions for different simulated datasets using functions from the package. Some EDA is required, for example obtaining the number of classes in total across all random effects, which is used as

input.

```
set.seed(1234)
model1 <- run_bayesian_imp(Y1 ~ V2 + V3 + V4 + (1 | gamma), data = data_bayes1)
posteriors1 <- sample_posteriors(Y1 ~ V2 + V3 + V4 + (1 | gamma), data = data_bayes1,
    5000, n, n_classes = 200)


model2 <- run_bayesian_imp(Y2 ~ V2 + V3 + V4 + (1 | gamma), data = data_bayes2)
posteriors2 <- sample_posteriors(Y2 ~ V2 + V3 + V4 + (1 | gamma), data = data_bayes2,
    5000, n, n_classes = 200)


model3 <- run_bayesian_imp(Y3 ~ V2 + V3 + V4 + (1 | gamma), data = data_bayes3)
posteriors3 <- sample_posteriors(Y3 ~ V2 + V3 + V4 + (1 | gamma), data = data_bayes3,
    5000, n, n_classes = 200)


model4 <- run_bayesian_imp(Y4 ~ V2 + V3 + V4 + (1 | gamma), data = data_bayes4)
posteriors4 <- sample_posteriors(Y4 ~ V2 + V3 + V4 + (1 | gamma), data = data_bayes4,
    5000, n, n_classes = 200)
```

PLOTTING

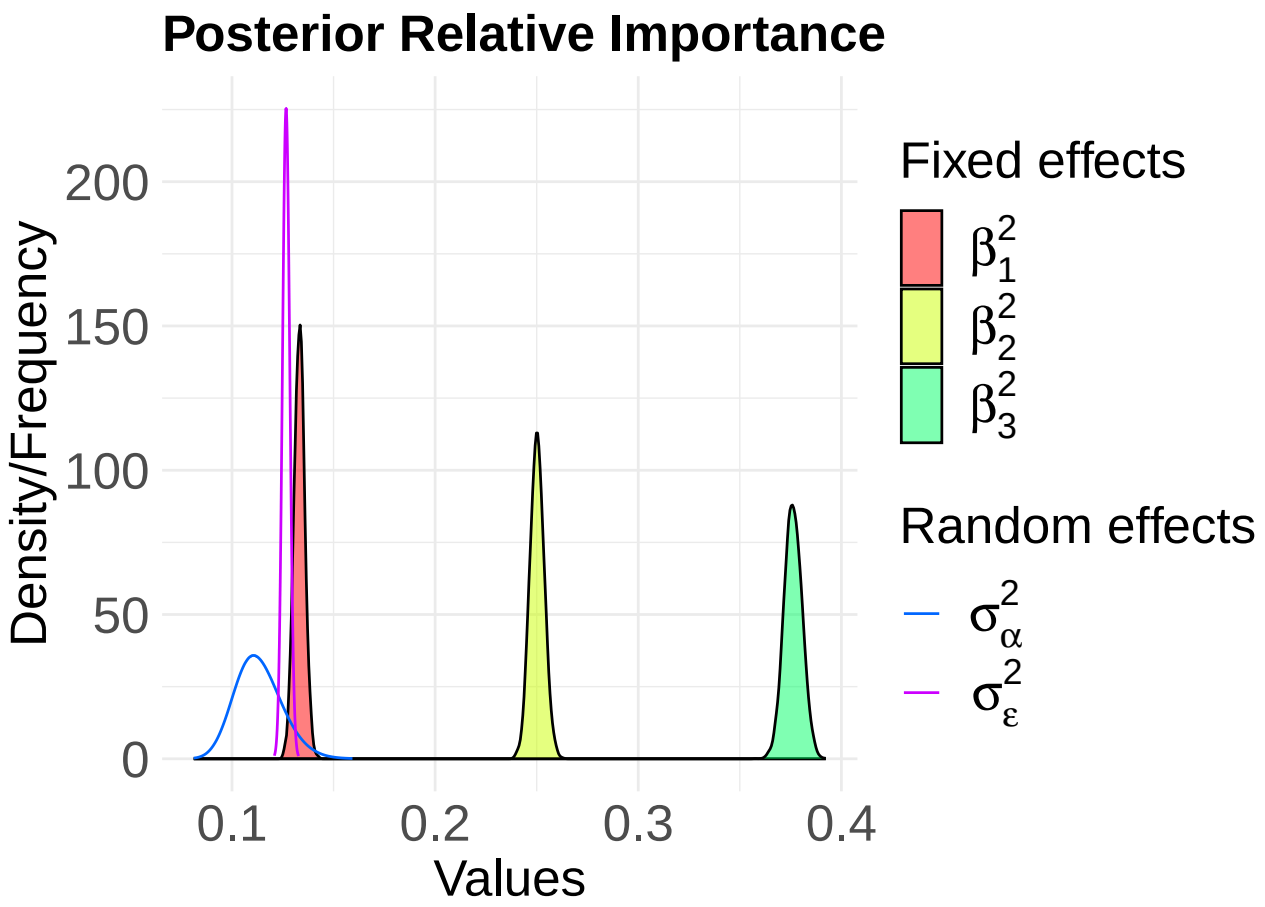
This code block defines a function plot_posterior that visualizes posterior distributions and relative importance from the sampled posteriors. The function creates density plots and overlays line plots for variance marginals. This can be modified for the specific problem at hand, and is not included in the package since it is a problem specific function.

```
plot_posterior <- function(betas, importances, marginals, importance = FALSE) {

    if (importance) {
        mat_long <- melt(as.data.frame(importances))
    } else {
        mat_long <- melt(as.data.frame(betas))
    }
    names(mat_long) <- c("Variable", "Value")

    df_marginals <- do.call(rbind, marginals)
    df_marginals$Effect <- as.factor(df_marginals$Effect)

    if (!importance) {
        plot_title = paste("Posterior Distributions")
    } else {
        plot_title = paste("Posterior Relative Importance")
    }

    random_effect_labels <- c(expression(sigma[alpha]^2), expression(sigma[epsilon]^2))
    fixed_effect_labels <- if (importance) {
        c(expression(beta[1]^2), expression(beta[2]^2), expression(beta[3]^2))
    } else {
        c(expression(beta[1]), expression(beta[2]), expression(beta[3]))
    }
    colors = rainbow(5)
```

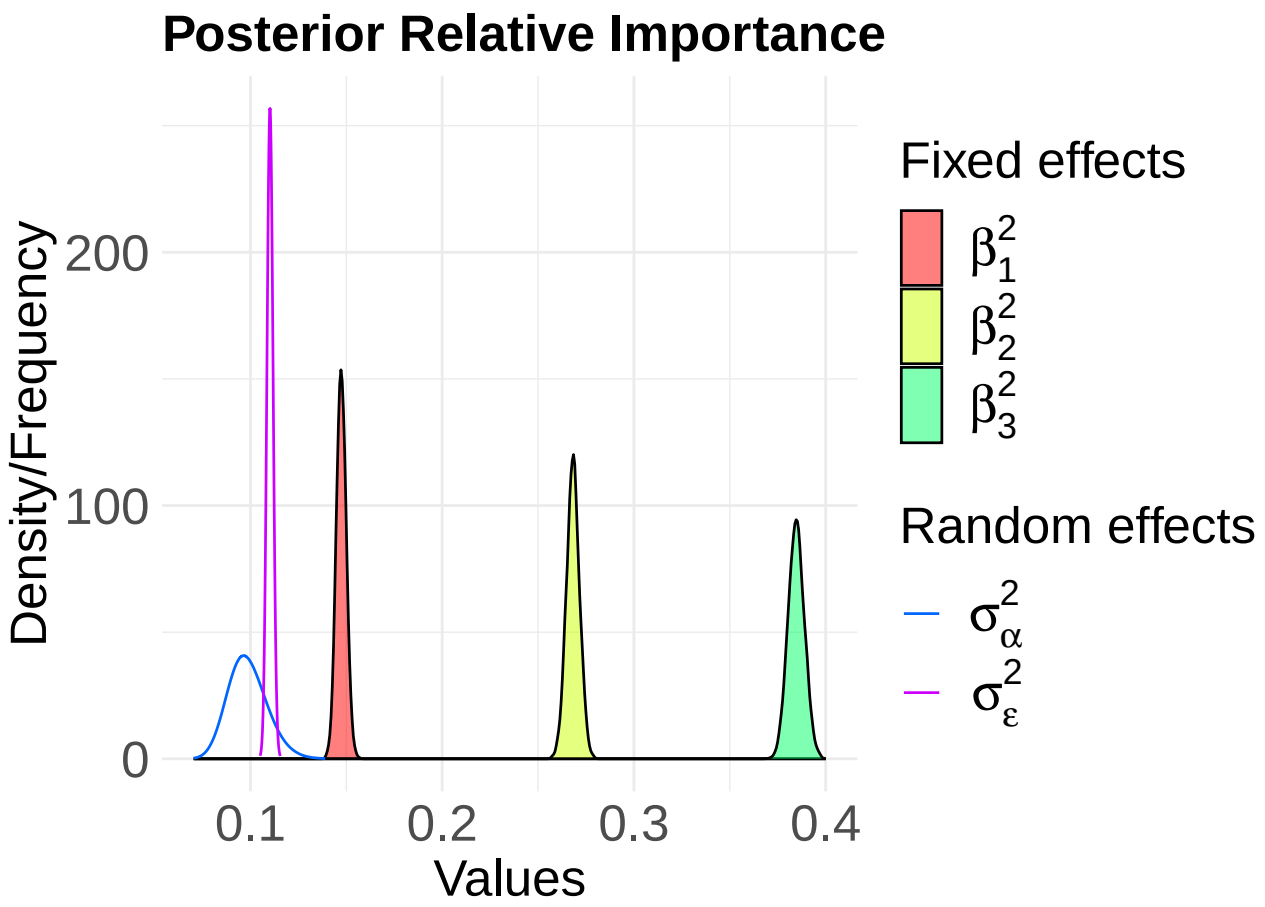
```
    p <- ggplot() + geom_density(data = mat_long, aes(x = Value, fill = as.factor(Variable)),
        alpha = 0.5) + geom_line(data = df_marginals, aes(x = x, y = y,
        color = Effect)) + labs(title = plot_title, x = "Values", y = "Density/Frequency") +
        theme_minimal() + theme(legend.position = "right", legend.text = element_text(size = 20),
        legend.title = element_text(size = 20), plot.title = element_text(size = 20,
            face = "bold"), axis.title.x = element_text(size = 20), axis.title.y = element_text(size = 2
        strip.text = element_text(size = 20), axis.text.x = element_text(size = 20),
        axis.text.y = element_text(size = 20)) + scale_fill_manual(name = "Fixed effects",
        values = colors[1:3], labels = fixed_effect_labels) + scale_color_manual(name = "Random effects
        values = colors[4:5], labels = random_effect_labels)
    return(p)
}

plot1 <- plot_posterior(posteriors1$beta, posteriors1$importance, posteriors1$marginals,
    importance = TRUE)

plot2 <- plot_posterior(posteriors2$beta, posteriors2$importance, posteriors2$marginals,
    importance = TRUE)

plot3 <- plot_posterior(posteriors3$beta, posteriors3$importance, posteriors3$marginals,
    importance = TRUE)

plot4 <- plot_posterior(posteriors4$beta, posteriors4$importance, posteriors4$marginals,
    importance = TRUE)
```

These plots are generated for each model's posteriors. We create and store the plots in variables plot1, plot2, plot3, and plot4 for the respective models.

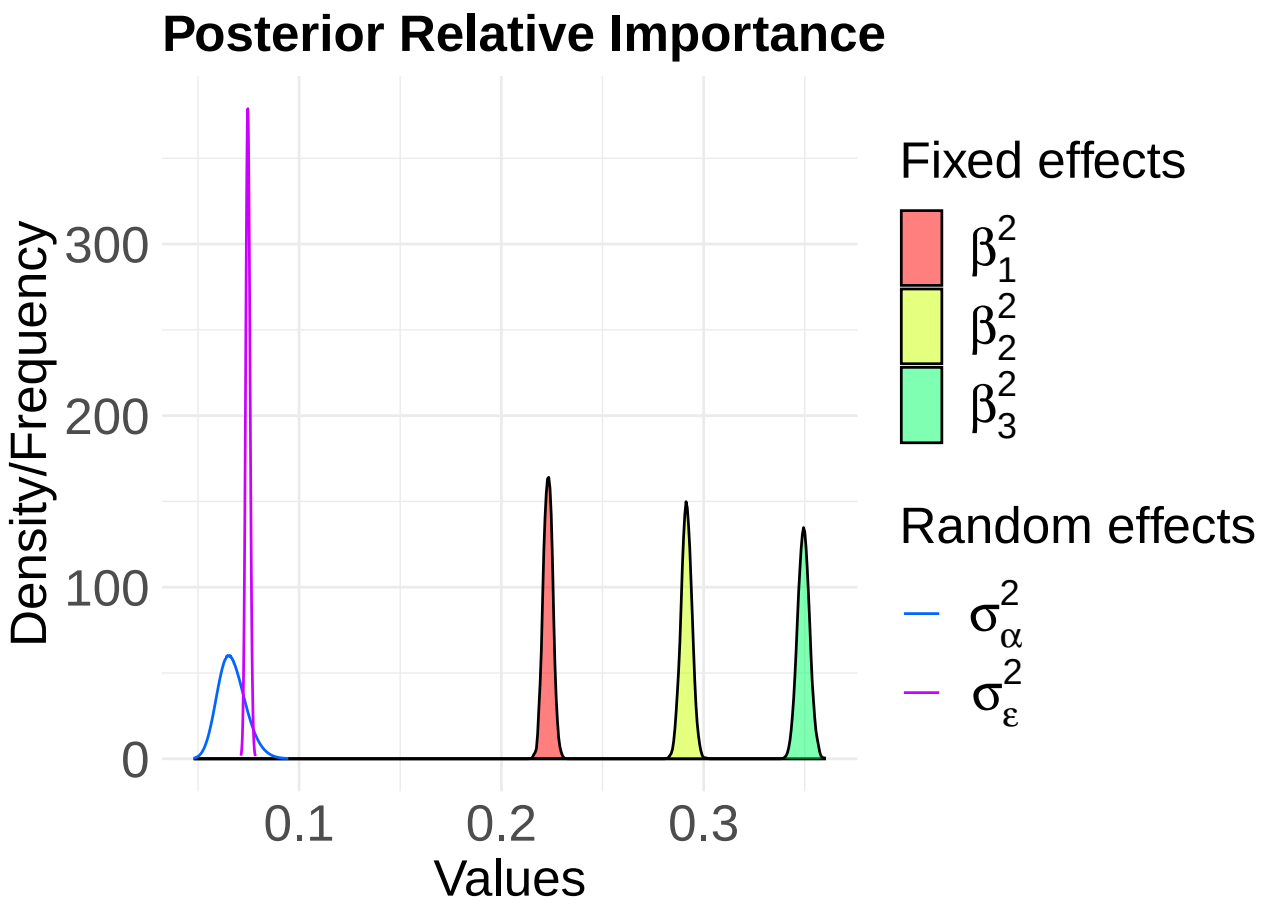
```
plot1
```


Posterior Relative Importance
Density/Frequency
200
150
100
50
0
0.1
0.2
0.3
0.4
Values
Fixed effects
β_1^2
β_2^2
β_3^2
Random effects
σ_α^2
σ_ε^2

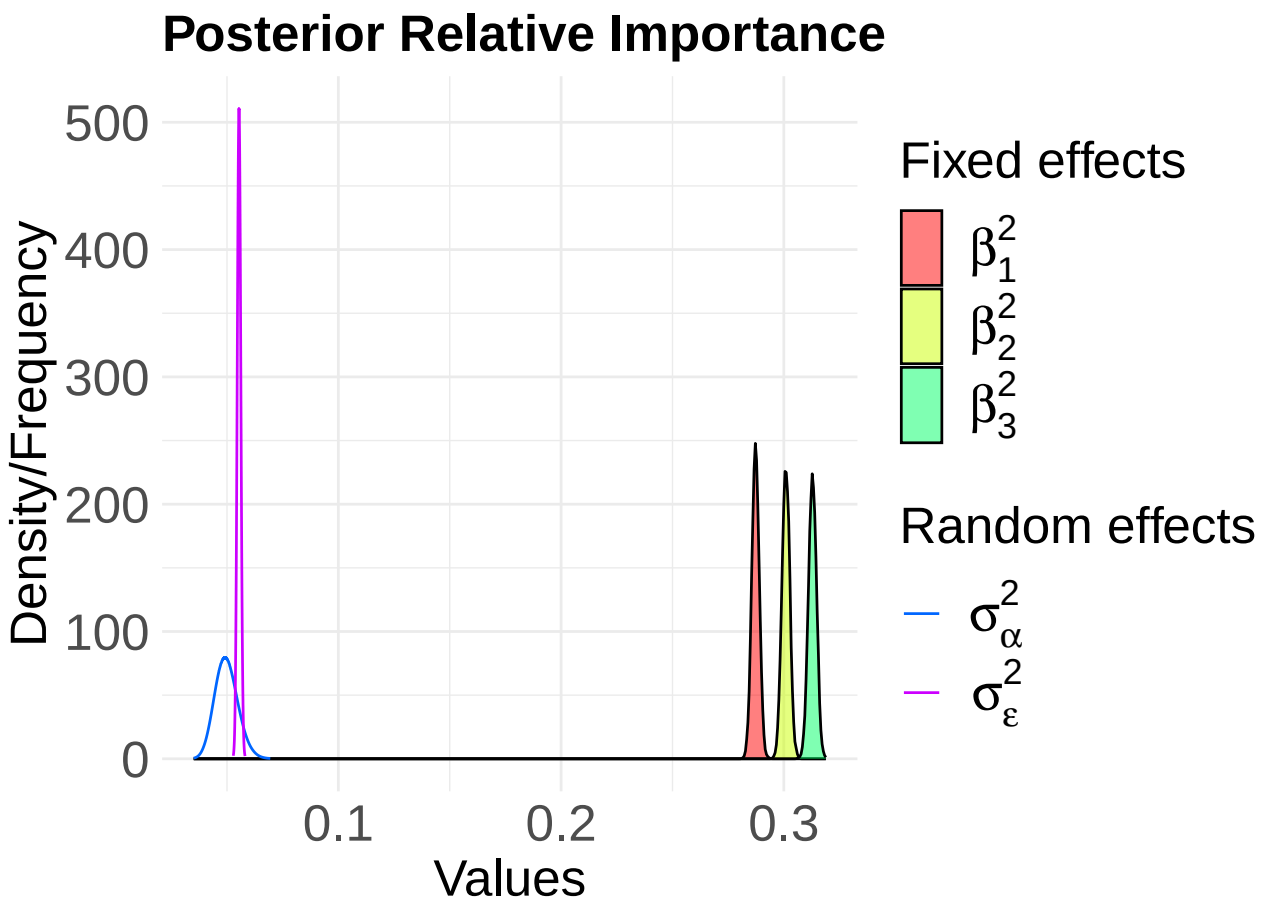
```
plot2
```


Posterior Relative Importance
Density/Frequency
200
100
0
0.1
0.2
0.3
0.4
Values
Fixed effects
β_1^2
β_2^2
β_3^2
Random effects
σ_α^2
σ_ε^2

```
plot3
```


Posterior Relative Importance
Density/Frequency
300
200
100
0
0.1
0.2
0.3
Values
Fixed effects
β_1^2
β_2^2
β_3^2
Random effects
σ_α^2
σ_ε^2

```
plot4
```



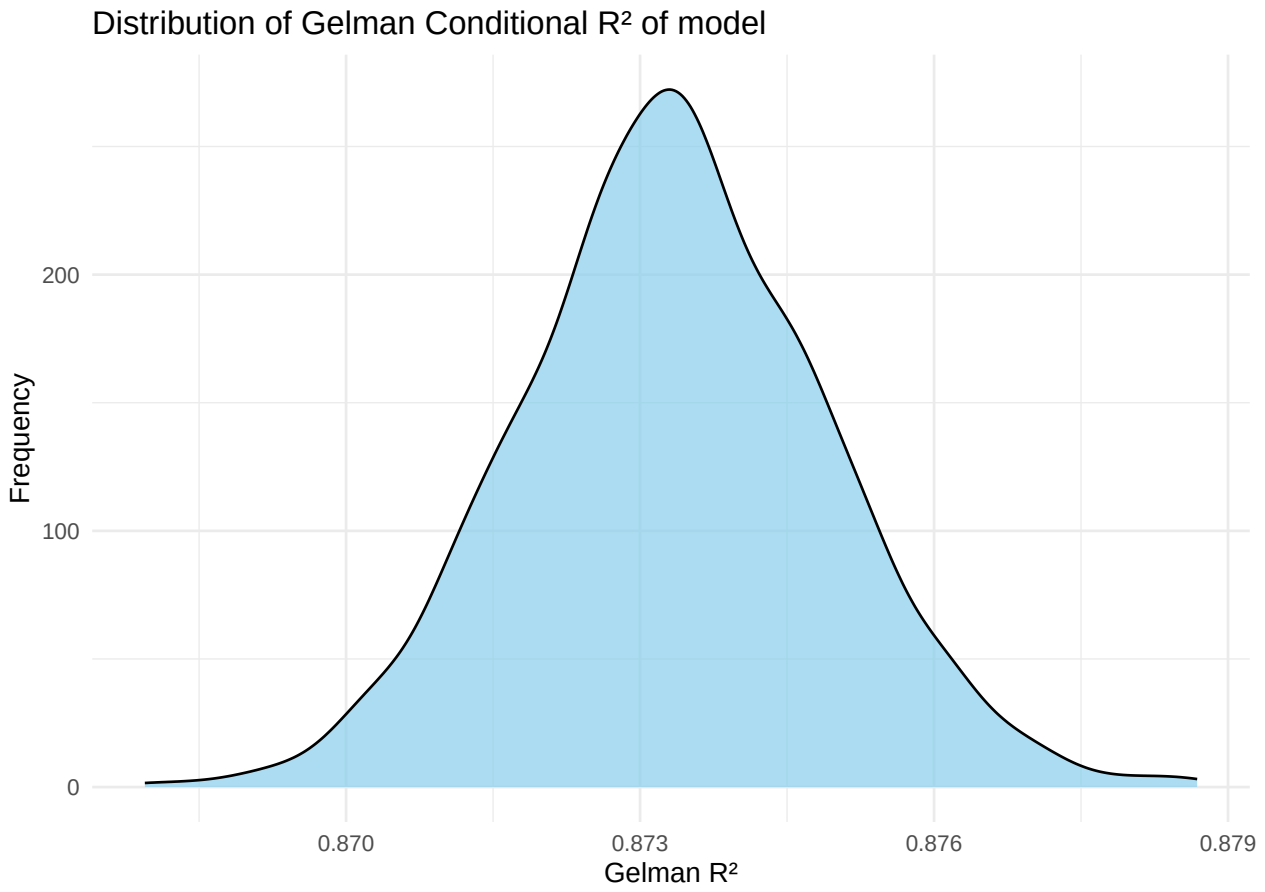
GELMAN R2

Here, we compute and visualize the Gelman conditional R^2 metrics for each model. This metric gives insight into the proportion of variance explained by the fixed effects in the models.

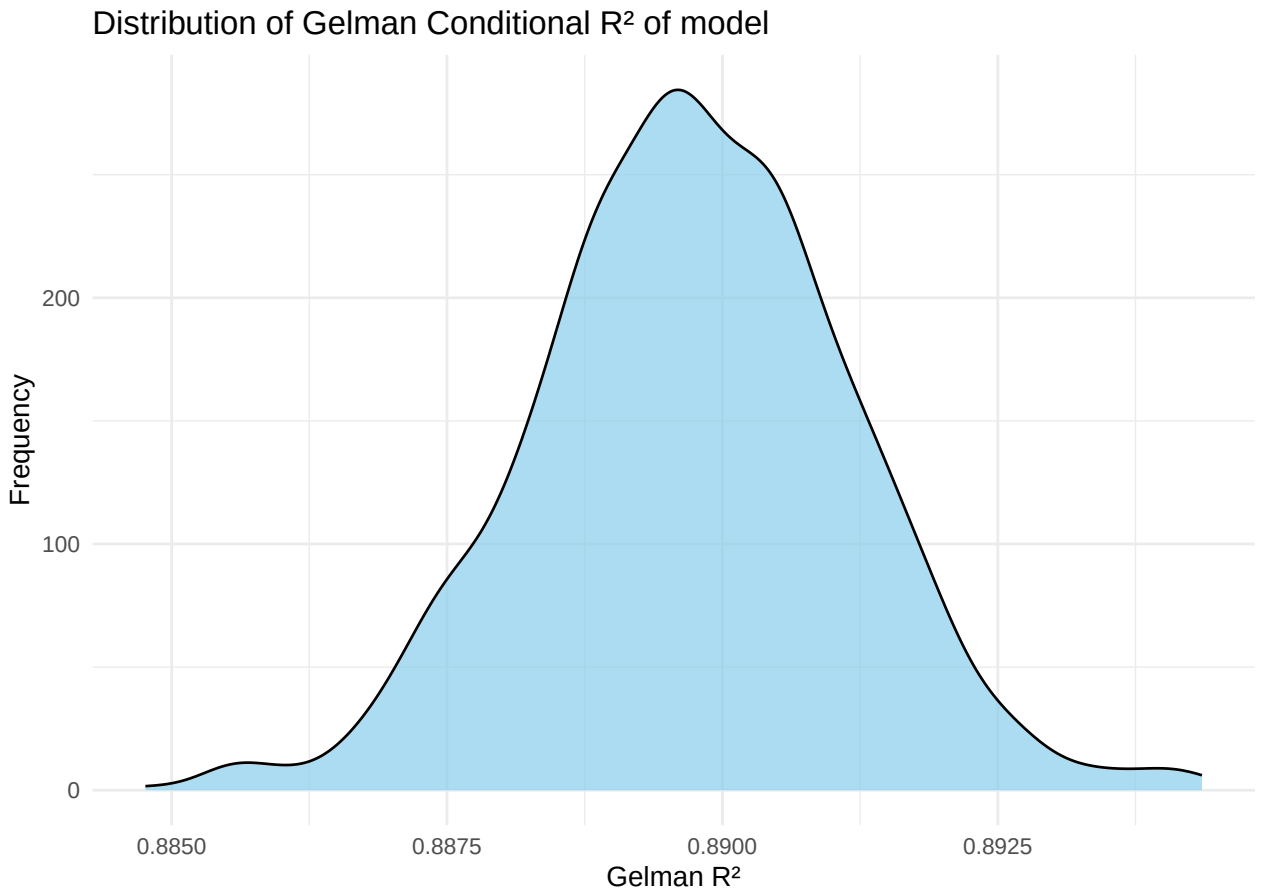
```
set.seed(1234)
R2_1 <- BayesianImportance::gelman_r2_metrics(model1, plot = TRUE)

R2_2 <- BayesianImportance::gelman_r2_metrics(model2, plot = TRUE)

R2_3 <- BayesianImportance::gelman_r2_metrics(model3, plot = TRUE)

R2_4 <- BayesianImportance::gelman_r2_metrics(model4, plot = TRUE)
```

This section displays the plots for the Gelman R^2 metrics computed in the previous step. Each plot corresponds to one of the models and illustrates the distribution of the conditional R^2 values.

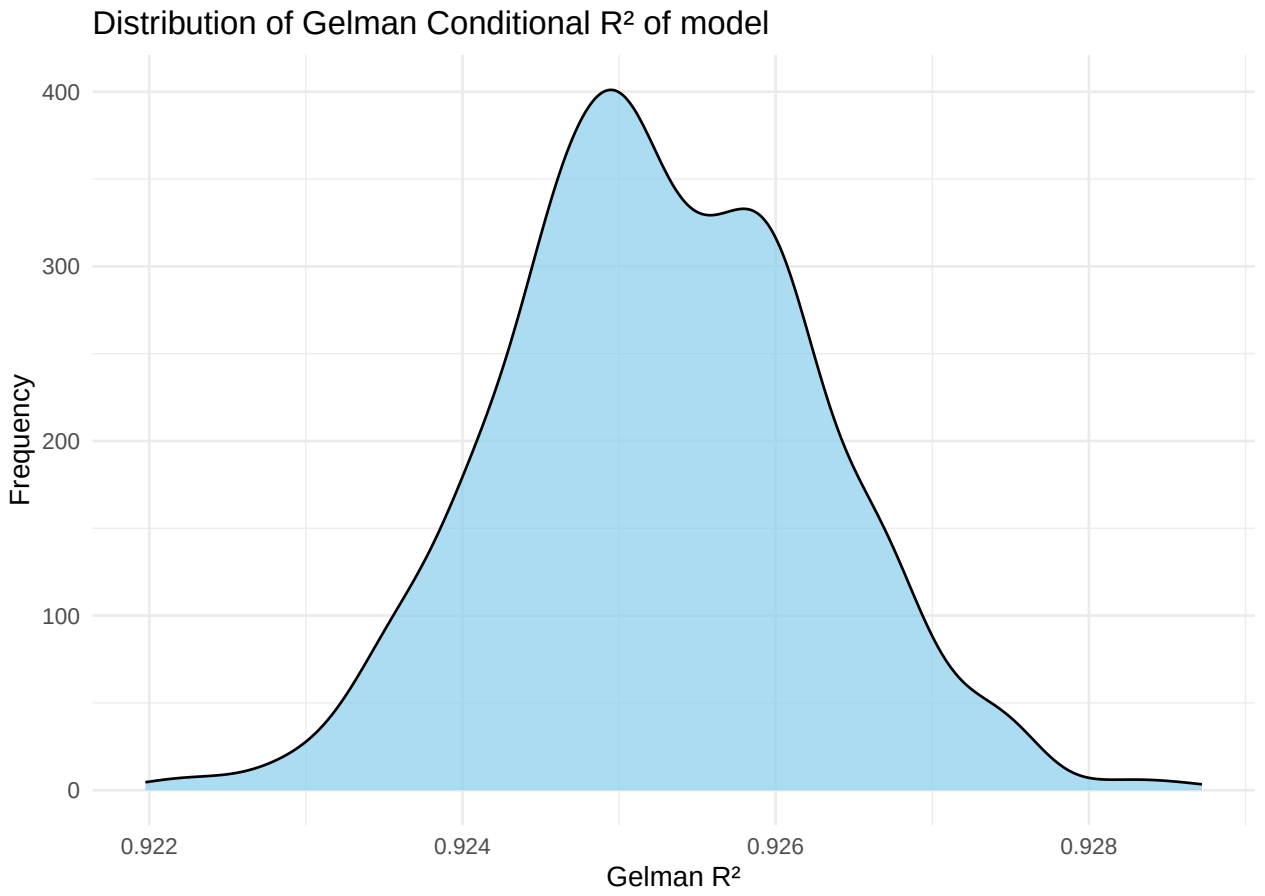
```
R2_1$plot
```


Distribution of Gelman Conditional R² of model
Frequency
200
100
0
0.870
0.873
0.876
0.879
Gelman R²

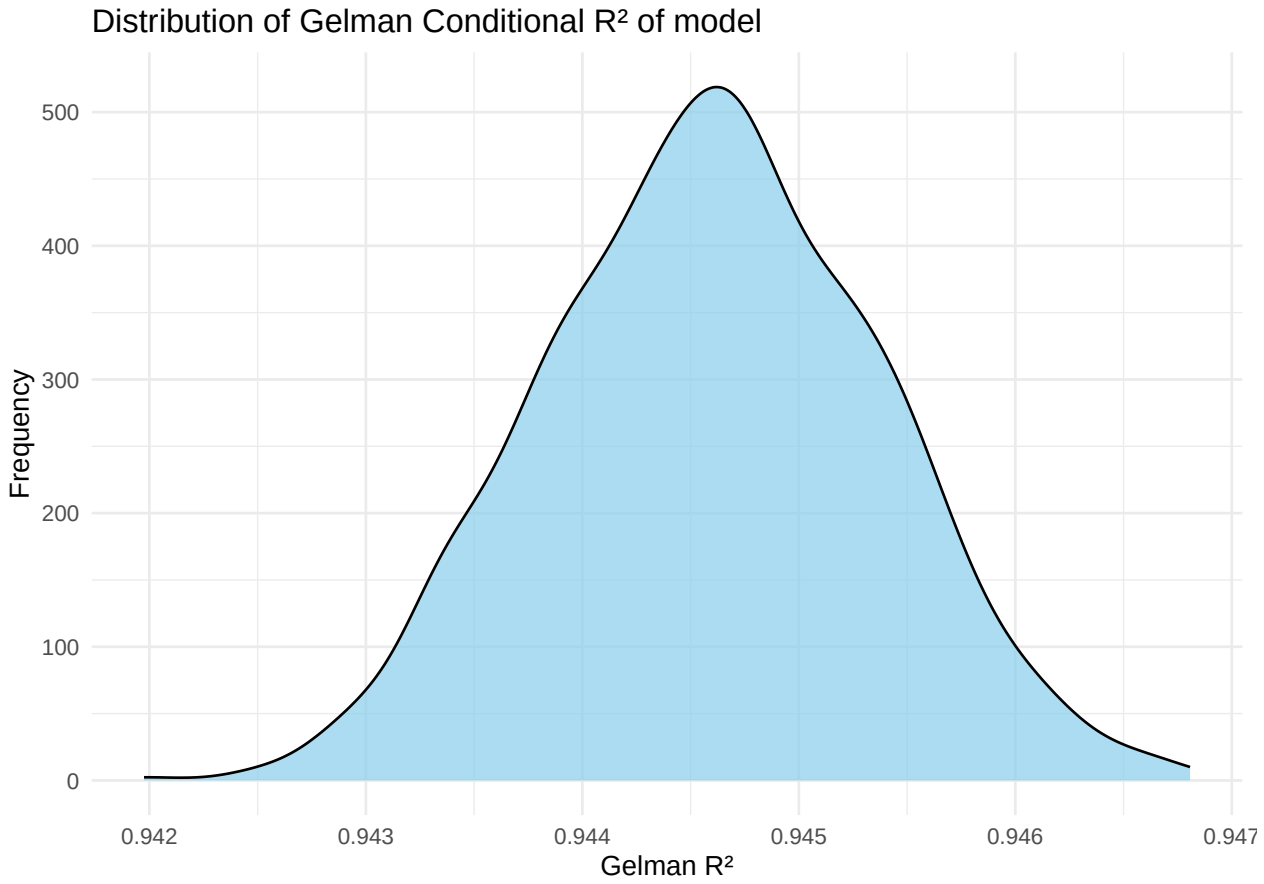
```
R2_2$plot
```


Distribution of Gelman Conditional R² of model
Frequency
200
100
0
0.8850
0.8875
0.8900
0.8925
Gelman R²

```
R2_3$plot
```


Distribution of Gelman Conditional R² of model
400
300
200
100
0
Frequency
0.922
0.924
0.926
0.928
Gelman R²

```
R2_4$plot
```


Distribution of Gelman Conditional R² of model
Frequency
500
400
300
200
100
0
0.942
0.943
0.944
0.945
0.946
0.947
Gelman R²