Principles of statistical inference project - Part 2

Carmel Gafa

June 22, 2025

1 Question 1

Find and download a dataset from the internet that includes at least three quantitative variables (more is better). Once you have selected a dataset, send the link via email to Dr. Monique Borg Inguanez (monique.inguanez@um.edu.mt) for approval, ensuring that each student works with a unique dataset. Using the approved dataset, fit a Bayesian multiple linear regression model ($p < 2$) using JAGS. Your submission should clearly include the following:

The selected dataset is the Concrete Compressive Strength dataset, which was downloaded from the University of California, Irvine. The response variable of the dataset is the Concrete compressive strength as a function of eight predictors, including ingredients and ageing time.

The following table outlines the variables of the dataset, which had no missing values:

Variable Name	**Role**	**Type**	**Units**
Cement	Feature	Continuous	kg/m^3
Blast Furnace Slag	Feature	Integer	kg/m^3
Fly `ash`	Feature	Continuous	kg/m^3
Water	Feature	Continuous	kg/m^3
Superplasticizer	Feature	Continuous	kg/m^3
Coarse Aggregate	Feature	Continuous	kg/m^3
Fine Aggregate	Feature	Continuous	kg/m^3
Age	Feature	Integer	day
Concrete Compressive Strength	Target	Continuous	MPa

Table 1: Variables in the Concrete Compressive Strength dataset

A first test on the dataset involved a pairwise and visual inspection of the relationships to determine the correlation of the predictors with the response and to start forming some ideas about multicollinearity. These results are shown below in Figure 1 and Figure 2.

	cement	slag	ash	water
cement	1.00000000	-0.27521591	-0.397467341	-0.08158675
slag	-0.27521591	1.00000000	-0.323579901	0.10725203
ash	-0.39746734	-0.32357990	1.000000000	-0.25698402
water	-0.08158675	0.10725203	-0.256984023	1.00000000
superplasticizer	0.09238617	0.04327042	0.377503146	-0.65753291
coarseagg	-0.10934899	-0.28399861	-0.009960828	-0.18229360
fineagg	-0.22271785	-0.28160267	0.079108491	-0.45066117
age	0.08194602	-0.04424602	-0.154370516	0.27761822
strength	0.49783192	0.13482926	-0.105754916	-0.28963338

	superplasticizer	coarseagg	fineagg	age
cement	0.09238617	-0.109348994	-0.22271785	0.08194602
slag	0.04327042	-0.283998612	-0.28160267	-0.04424602
ash	0.37750315	-0.009960828	0.07910849	-0.15437052
water	-0.65753291	-0.182293602	-0.45066117	0.27761822
superplasticizer	1.00000000	-0.265999148	0.22269123	-0.19270003
coarseagg	-0.26599915	1.000000000	-0.17848096	-0.00301588
fineagg	0.22269123	-0.178480957	1.00000000	-0.15609470
age	-0.19270003	-0.003015880	-0.15609470	1.00000000
strength	0.36607883	-0.164934614	-0.16724125	0.32887300

	strength
cement	0.4978319
slag	0.1348293
ash	-0.1057549
water	-0.2896334
superplasticizer	0.3660788
coarseagg	-0.1649346
fineagg	-0.1672412
age	0.3288730
strength	1.0000000

Figure 1: Correlation matrix of all variables in the Concrete Compressive Strength dataset.

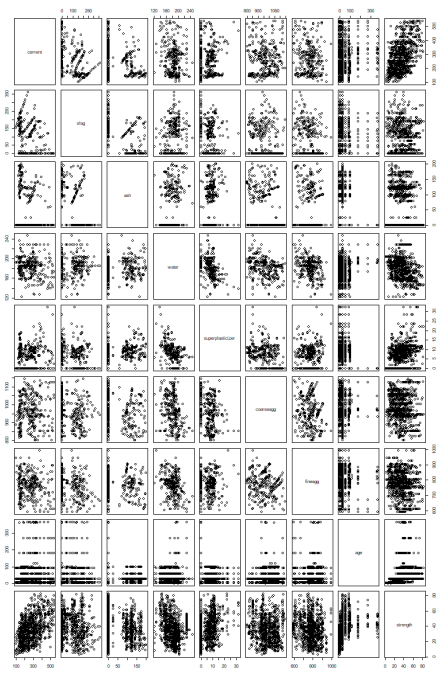


Figure 2: Pairwise scatterplot matrix of all variables in the Concrete Compressive Strength dataset.

From this result, the best correlation with strength is achieved with `cement`, followed by a moderate correlation with `superplasticiser` (+0.37) and `age` (+0.33). Water has a moderate negative correlation (-0.29). `Slag` and `ash`

```
Loading required package: carData
        cement           slag            ash          water
      7.488944       7.276963       6.170634       7.003957
superplasticizer     coarseagg       fineagg            age
      2.963776       5.074617       7.005081       1.118367
```

Figure 3: Variance Inflation Factor values for all predictors in the Concrete Compressive Strength dataset.

have a weak correlation. We can also see a strong negative correlation between `superplasticiser` and `water` (-0.66) and `water` and `fine aggregate` (-0.45).

Variance inflation factor analysis was then conducted to investigate multicollinearity, with the results of the analysis presented in Figure 3.

The analysis shows that cement, `Slag`, `ash`, and `water` exhibit high collinearity (in the region of 7), while `superplasticiser` (2.96) and `age` (1.12) have moderate values.

Following this investigation, cement, `superplasticiser`, and `age` were selected as predictors.

The following code snippet illustrates the data preparation. Prior to this step, the datasets's column names were shortened so they could be manipulated easily.

```
script_dir <- getwd()
concrete_cleansed_path <- file.path(
script_dir, "project_2_code/concrete_cleansed.csv")

concrete_cleansed <- read.csv(concrete_cleansed_path)

jags_data <- list(
x_cement = concrete_cleansed$cement,
x_superplasticizer = concrete_cleansed$superplasticizer,
x_age = concrete_cleansed$age,
y_strength = concrete_cleansed$strength,
n = nrow(concrete_cleansed)
)
```

1.1 A full specification of the Bayesian model:

- **The likelihood function**
- **The prior distributions selected for the parameters**

The response variable in this excercise is `strength`, representing the concrete compressive strength in MPa. As we have seen in the previous section, the predictors included in the model are `cement`, `superplasticizer`, and `age`, all selected based on correlation and multicollinearity analysis.

The model assumes the following structure:

$$y_i \sim \mathcal{N}(\mu_i, \tau^{-1}) \quad \text{for } i = 1, \ldots, n$$
$$\mu_i = \beta_0 + \beta_1 \cdot x_{\text{cement},i} + \beta_2 \cdot x_{\text{superplasticizer},i} + \beta_3 \cdot x_{\text{age},i}$$

The prior distributions for the parameters are defined as follows:

$$\begin{aligned}
\beta_0 &\sim \mathcal{N}(0, 100) \\
\beta_1 &\sim \mathcal{N}(0, 100) \\
\beta_2 &\sim \mathcal{N}(0, 100) \\
\beta_3 &\sim \mathcal{N}(0, 100) \\
\tau &\sim \text{Gamma}(0.01, 0.01)
\end{aligned}$$

The variance of the likelihood is modeled via the precision parameter τ, where;

$$\tau = \frac{1}{\sigma^2}$$

The model was defined using the following statement:

```
model_description <- "
model {
        for (i in 1:n) {
                y_strength[i] ~ dnorm(mu[i], tau)
                mu[i] <- beta0 + (beta1 * x_cement[i]) + (beta2 *
                    x_superplasticizer[i]) + (beta3 * x_age[i])
        }
        beta0 ~ dnorm(0, 0.01)
        beta1 ~ dnorm(0, 0.01)
        beta2 ~ dnorm(0, 0.01)
        beta3 ~ dnorm(0, 0.01)
        tau ~ dgamma(0.01, 0.01)
} "
```

1.2 A justification for the chosen priors, including any assumptions or reasoning behind them.

The priors in this model are weakly informative; that is, they do not strongly influence the results, but they give only a little information about what values we expect for the parameters. The coefficients for β, for example, were given normal priors centred at 0 with a standard deviation of 10, which allows for a large range of possible values. These priors are wide enough to let the data have the biggest impact on the results, but they also help the model stay stable and avoid extreme or unrealistic values during sampling.

- **Regression Coefficients (β_0, β_1, β_2, β_3):** Each coefficient was assigned a normal prior $\mathcal{N}(0, 100)$, corresponding to a mean of 0 and a large variance of 100, reflecting a belief that the effect of each predictor is centred around

zero but allows for a wide range of plausible values. As discussed, these priors are considered weakly informative because they do not strongly constrain the parameter estimates but prevent extreme values that could destabilize the sampling process.

- **Precision Parameter (τ):** A Gamma$(0.01, 0.01)$ prior was used for the precision of the normal likelihood, which is a standard weak prior for precision in Bayesian models. It gives a very wide range of possible values for the standard deviation, which means we do not assume much about how spread out the data is, reflecting that we have little prior knowledge about the variation in the response.

The priors were selected to reflect minimal assumptions about the parameter values in the absence of strong domain knowledge, which is appropriate given the exploratory nature of this analysis.

1.3 The selected burn-in period for your MCMC chains, including a detailed explanation of how this was chosen.

Trace plots are the first tool that is used to determine a suitable burn-in period. In this analysis, we examine chain mixing, which the overlap of the chain samples can visually assess, and compactness within the chain samples. Additionally, chain variations should be consistent around a central value. Trace plots, however, are subjective in interpretation and can indicate early convergence. Another tool that is then used is the Gelman-Rubin convergence diagnostic, which utilises both between-chain and within-chain variance to assess convergence.

Figure 4 shows the trace plot for the parameter β_0, that is, the intercept after a burn-in of just 200 samples. The trance can visually satisfy the criteria for convergence.

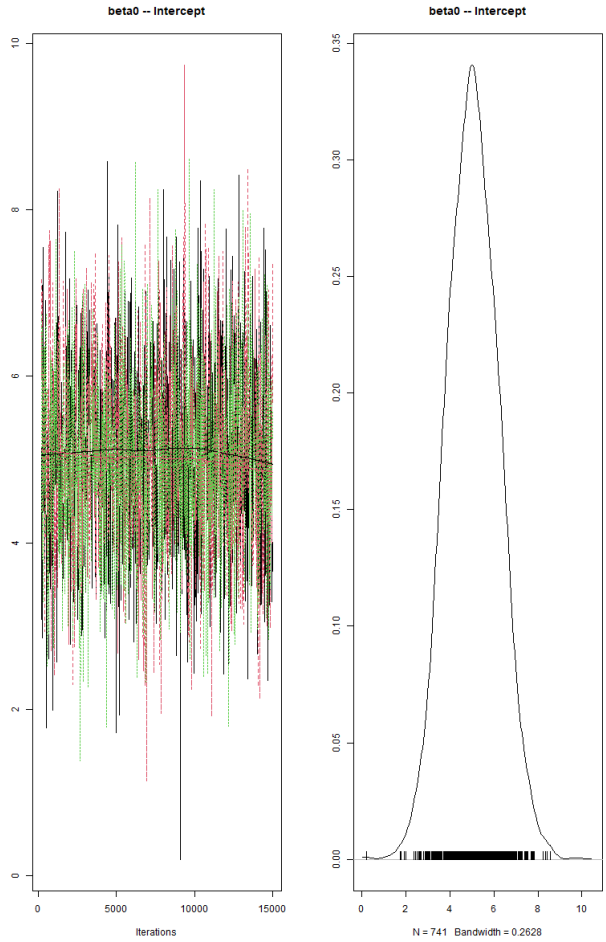


Figure 4: Trace and density plot for β_0 (Intercept) across 3 chains after iteration 200.

The Gelman-Rubin diagnostic was also examined. Figure 5 shows that the values for all parameters converge to 1.00 after iteration 6000, confirming chain convergence and the initial iterations were likely influenced by starting values.

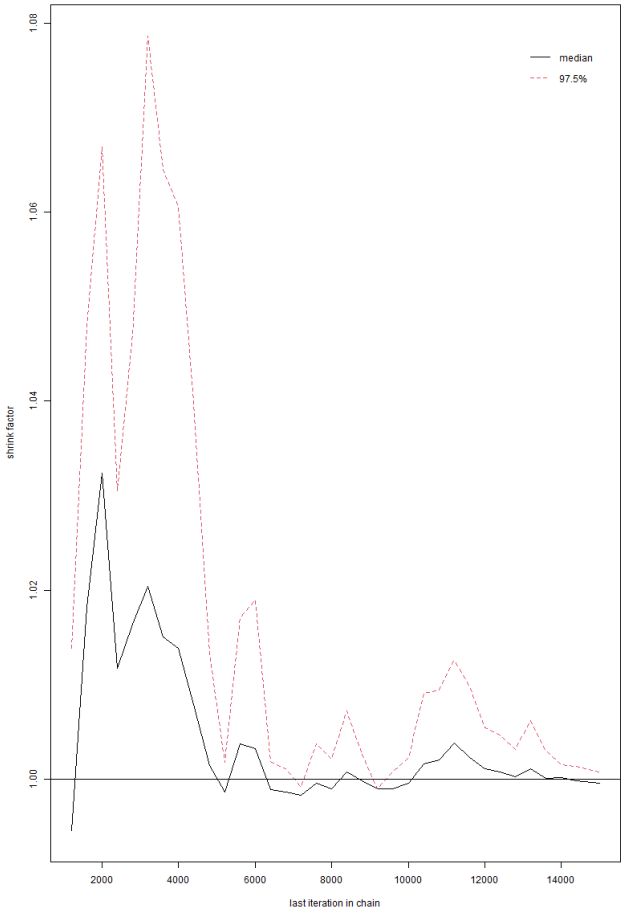


Figure 5: Gelman–Rubin diagnostic for β_0. Convergence is reached after approximately 6000 iterations

Based on these observations, a burn-in of 6000 iterations was selected to ensure that only samples drawn from the stationary portion of the chains were retained for posterior inference.

The following snippet shows the execution of the model and creation of the plot used in this analysis

```
n_chains <- 3
n_burnin <- 200
n_samples <- 15000
parameters_to_monitor <- c("beta0", "beta1", "beta2", "beta3",
    "tau")

initial_values <- list(
```

```
list(beta0 = -10, beta1 = 0.5, beta2 = 0.1, beta3 = 0.05, tau =
    0.5),
list(beta0 = 0,   beta1 = 1,   beta2 = 0.3, beta3 = 0.1,  tau = 1),
list(beta0 = 10,  beta1 = 2,   beta2 = 0.5, beta3 = 0.2,  tau = 2)
)

model <- jags.model(textConnection(model_description),
data = jags_data,
inits = initial_values,
n.chains = n_chains)

post <- coda.samples(model = model,
variable.names = parameters_to_monitor,
n.iter = n_samples,
thin = 20)

post_burned <- window(post, start = n_burnin)
print(summary(post_burned))
plot(post_burned[, "beta0"], main="beta0 -- Intercept")
```

1.4 A presentation and interpretation of:

- **Convergence diagnostics (e.g., trace plots, Gelman-Rubin statistics)**
- **Accuracy diagnostics (e.g., effective sample size, autocorrelation)**

The Figures below illustrate convergence diagnostics for all monitored parameters in the model. Trace plots are shown after discarding an initial burn-in of 200 samples, highlighting chain behaviour and visual mixing. Additionally, Gelman-Rubin diagnostic plots are presented both after a burn-in of 200 samples and after a more conservative burn-in of 6000 samples. These plots allow for a comparative evaluation of convergence, confirming that the longer burn-in more effectively removed transient behaviour and yielded PSRF values stabilising around 1.00 across all parameters.

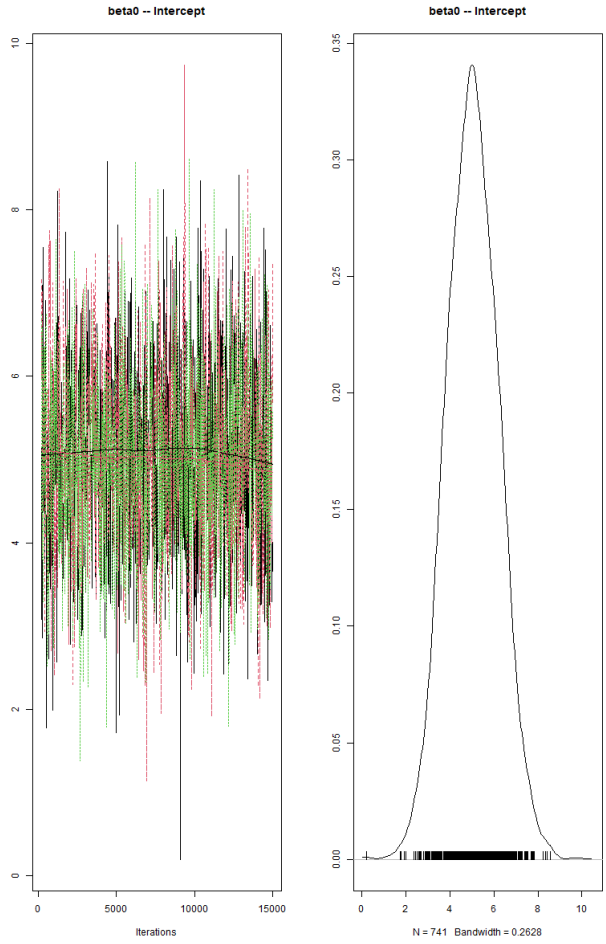


Figure 6: Trace and posterior density plots for the parameter β_0 (Intercept). The trace plot shows samples from three chains with good mixing and no signs of divergence or drift. All chains fluctuate around a common central value and remain tightly overlapped throughout the sampling period, a sign of stable convergence. The density plot shows a unimodal posterior distribution, peaking near 5.0 and spanning a broad range from approximately 2 to 8. The shape of the curve is slightly asymmetric near the peak, but remains smooth overall. The rug plot confirms dense and well-distributed sampling, supporting a reliable estimate of the intercept parameter. This indicates that the model has effectively identified a stable posterior distribution for β_0.

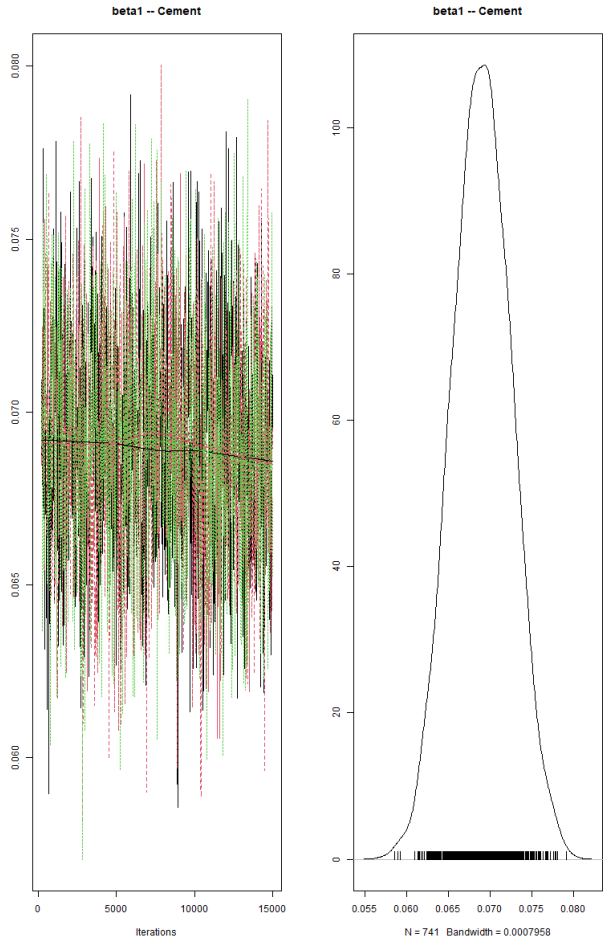


Figure 7: Trace and posterior density plots for the parameter β_1 (Cement). The trace plot on the left displays samples from three Markov Chain Monte Carlo (MCMC) chains. These chains show good mixing, with overlapping fluctuations around a stable mean after a burn-in period of 200 iterations. There is no visible drift or separation between the chains, and each chain explores the same region of the parameter space, indicating that convergence for β_1 has been achieved. The density plot on the right illustrates a smooth, unimodal posterior distribution, with the majority of the probability mass concentrated between approximately 0.060 and 0.078, peaking near 0.069. This suggests a strong positive effect of cement on concrete compressive strength.

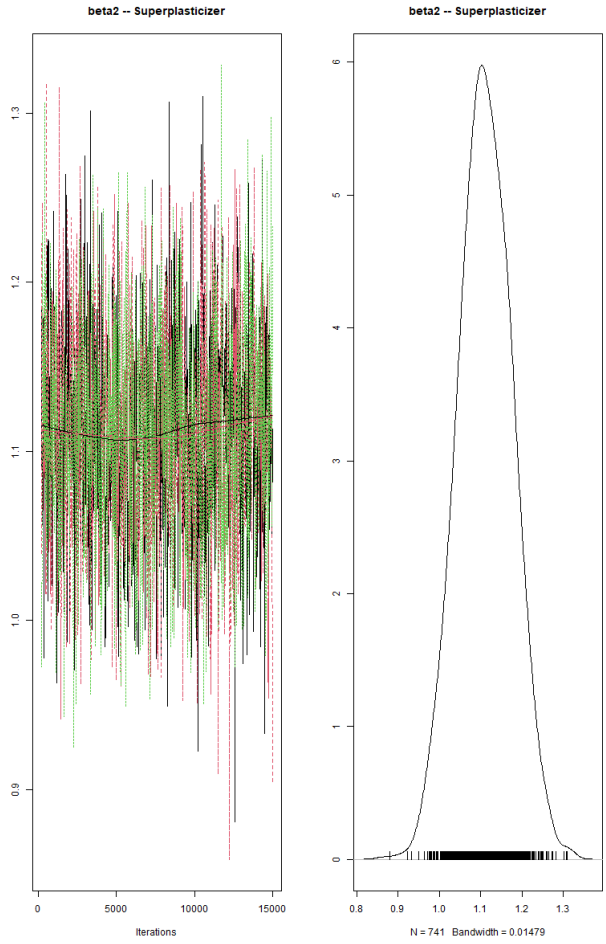


Figure 8: Trace and posterior density plots for the parameter β_2 (Superplasticizer). The trace plot shows three well-mixed chains with no indications of divergence, upward drift, or chain separation. The chains overlap consistently and fluctuate around a common mean, showing convergence and stationarity. The posterior density plot is unimodal, with the distribution between approximately 0.95 and 1.25, and a peak near 1.11. The curve is slightly asymmetric, with a marginally longer right tail, indicating slight skewness in the posterior. The rug plot confirms that sampling was dense and thorough across the high-probability region. These results indicate a stable and significant positive effect of superplasticizer on concrete compressive strength.

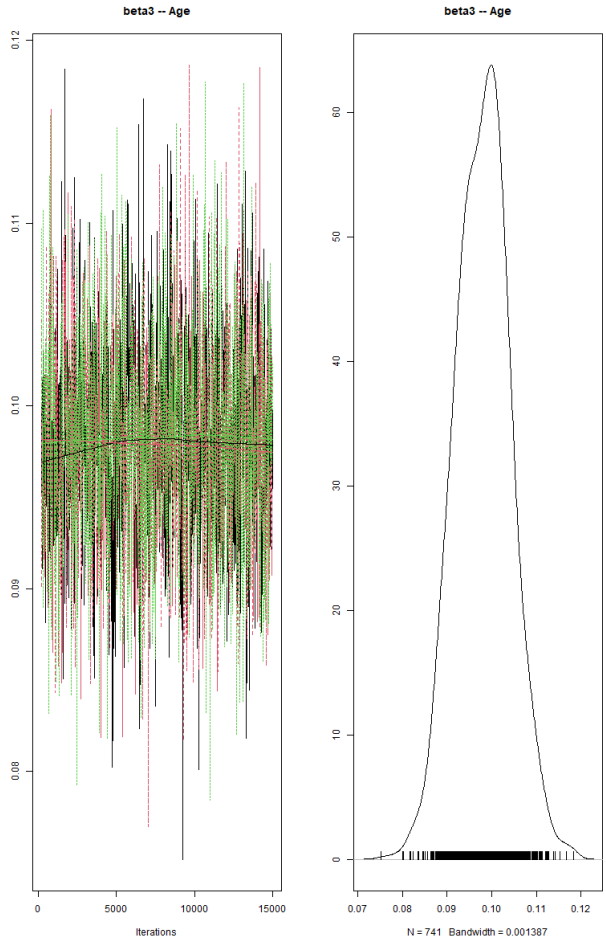


Figure 9: Trace and posterior density plots for the parameter β_3 (Age). The trace plot on the left displays three MCMC chains,exhibiting stable mixing with consistent fluctuations around a common central value, thus indicating convergence. The density plot on the right shows a unimodal posterior distribution concentrated between approximately 0.085 and 0.110. While the distribution has a clearly defined peak near 0.098, a slight asymmetry is visible near the mode, with a more gradual slope on the right side, indicating mild skewness. The rug plot confirms a dense and even sampling of values, supporting a reliable and stable estimate for the effect of age on compressive strength.

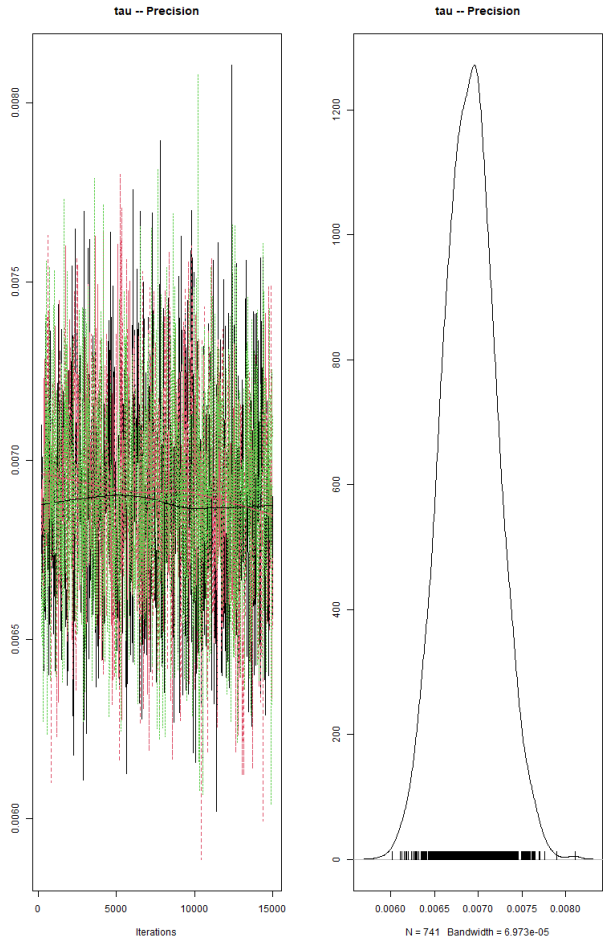


Figure 10: Trace and posterior density plots for the parameter τ (precision of the normal likelihood). The trace plot shows good chain mixing and stationarity across all three chains, with no divergence, drift, or separation after burn-in. Fluctuations are consistently centred around a stable mean, showing convergence. The posterior density plot reveals a narrow and sharply peaked unimodal distribution concentrated between approximately 0.0063 and 0.0078, peaking near 0.0069, showing a tightly constrained estimate of the model's residual precision. The rug plot below the density curve confirms dense and stable sampling from the posterior. This supports reliable inference for the precision parameter τ.

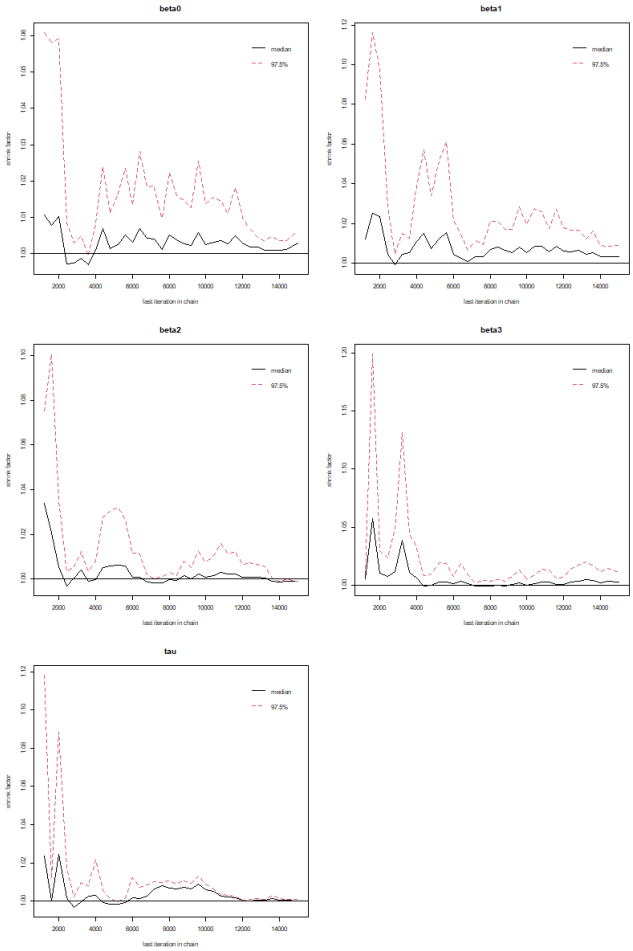


Figure 11: Gelman–Rubin diagnostic plots for all monitored parameters: intercept, cement, superplasticizer, age, and precision. The plots show the evolution of the shrink factor across increasing chain lengths. All parameters converge to Potential Scale Reduction Factor values close to 1.00 with their 97.5% confidence intervals stabilising below 1.05. Convergence for all parameters appears to occur around iteration 6000, after which the shrink factors remain flat and close to unity. This shows that the chains for all parameters have likely converged to the same posterior distribution and that no substantial between-chain variation remains.

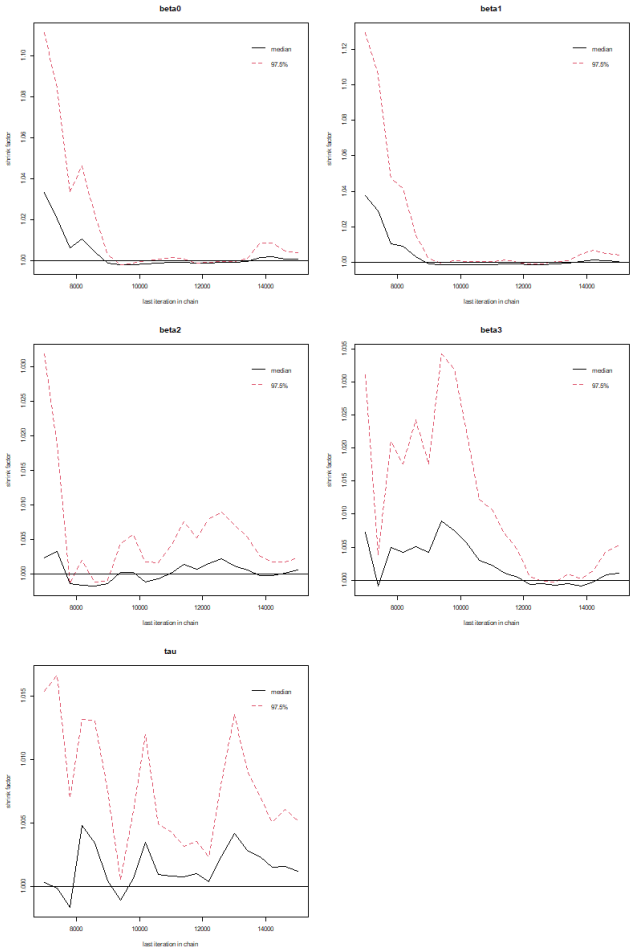


Figure 12: Gelman-Rubin diagnostic plots for all monitored parameters following the removal of the first 6000 iterations as burn-in. After burn-in, all parameters show Potential Scale Reduction Factor values stabilising near 1.00, with their 97.5% confidence intervals remaining below 1.05, indicating convergence. This confirms that discarding the initial 6000 iterations was effective in eliminating non-stationary behaviour, and that the retained samples are representative of the joint posterior distribution.

The Gelman-Rubin Potential Scale Reduction Factor convergence diagnostic was also considered for accuracy. As shown in Figure all parameters have Potential Scale Reduction Factor point estimates equal to or very close to 1.00, with upper confidence limits not exceeding 1.03, confirming that the chains have mixed well and that the sampling process is stable and reliable for posterior inference.

```
Potential scale reduction factors:

      Point est. Upper C.I.
beta0       1.01       1.02
beta1       1.01       1.03
beta2       1.00       1.01
beta3       1.00       1.00
tau         1.00       1.00
```

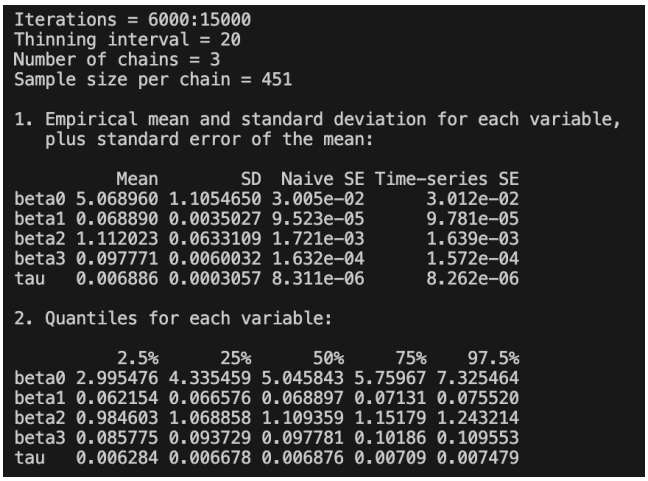
Figure 13: Potential Scale Reduction Factor values for all monitored parameters. All point estimates are close to 1.00, and upper confidence intervals remain below the typically accepted threshold of 1.05, indicating convergence across chains for each parameter.

To assess the reliability and efficiency of the Markov Chain Monte Carlo sampling process, we examined two key accuracy diagnostics: effective sample size and autocorrelation of the samples.

The effective sample sizes for all parameters were obtained using the summary() function on the Markov Chain Monte Carlo output object.

The effective sample size tells us how many of the samples from the Markov Chain Monte Carlo simulation are as useful as completely independent values. This is important because, in Markov Chain Monte Carlo, the samples are usually correlated, meaning each one is slightly influenced by the one before it. So, even if we run many iterations, the actual amount of "independent information" is lower. In this analysis, each chain had about 451 effective samples for each parameter after discarding the burn-in and applying thinning (keeping every 20th sample). Since we ran three chains, that means we had over 1300 effective samples per parameter, meaning that the Markov Chain Monte Carlo sampling worked well and gave us enough reliable information to make accurate conclusions from the posterior distributions.

The time-series standard errors were small for all parameters (e.g., β_0: 0.030, β_1: 0.000098, β_2: 0.00164), confirming high sampling precision and low Monte Carlo error. These diagnostics collectively indicate that the model's parameter estimates are stable, reliable, and based on an efficient sampling process.



```
Iterations = 6000:15000
Thinning interval = 20
Number of chains = 3
Sample size per chain = 451

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

          Mean        SD  Naive SE Time-series SE
beta0 5.068960 1.1054650 3.005e-02      3.012e-02
beta1 0.068890 0.0035027 9.523e-05      9.781e-05
beta2 1.112023 0.0633109 1.721e-03      1.639e-03
beta3 0.097771 0.0060032 1.632e-04      1.572e-04
tau   0.006886 0.0003057 8.311e-06      8.262e-06

2. Quantiles for each variable:

          2.5%      25%      50%     75%    97.5%
beta0 2.995476 4.335459 5.045843 5.75967 7.325464
beta1 0.062154 0.066576 0.068897 0.07131 0.075520
beta2 0.984603 1.068858 1.109359 1.15179 1.243214
beta3 0.085775 0.093729 0.097781 0.10186 0.109553
tau   0.006284 0.006678 0.006876 0.00709 0.007479
```

Figure 14: Summary statistics for all model parameters after burn-in and thinning.

Autocorrelation plots are used to assess how much each sample in a Markov Chain Monte Carlo simulation depends on the samples that came before it. In a well-mixed Markov Chain Monte Carlo chain, we expect the correlation between successive samples to decline rapidly as the lag increases; that is, samples become effectively independent as we move further apart along the chain. In an autocorrelation plot, each vertical bar represents the correlation at a particular lag. A plot showing high autocorrelation across many lags indicates poor mixing and slow convergence, while a plot where autocorrelation drops quickly to near zero suggests good mixing and efficient sampling. Ideally, the autocorrelation should be close to zero after just a few lags, indicating that the chain is exploring the posterior distribution effectively and that the resulting samples are reliable for inference.

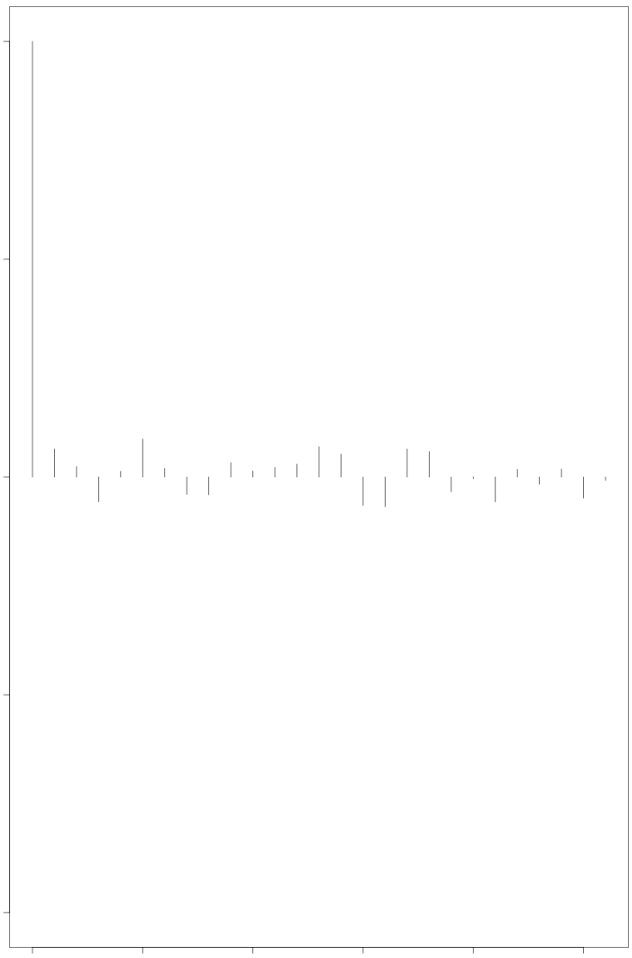


Figure 15: Autocorrelation plot for MCMC samples of β_0. The plot shows a rapid decline in autocorrelation after lag 1, with subsequent lags fluctuating closely around zero. This pattern indicates minimal serial correlation and suggests that the chain for β_0 mixes well, resulting in effectively independent samples suitable for posterior inference.

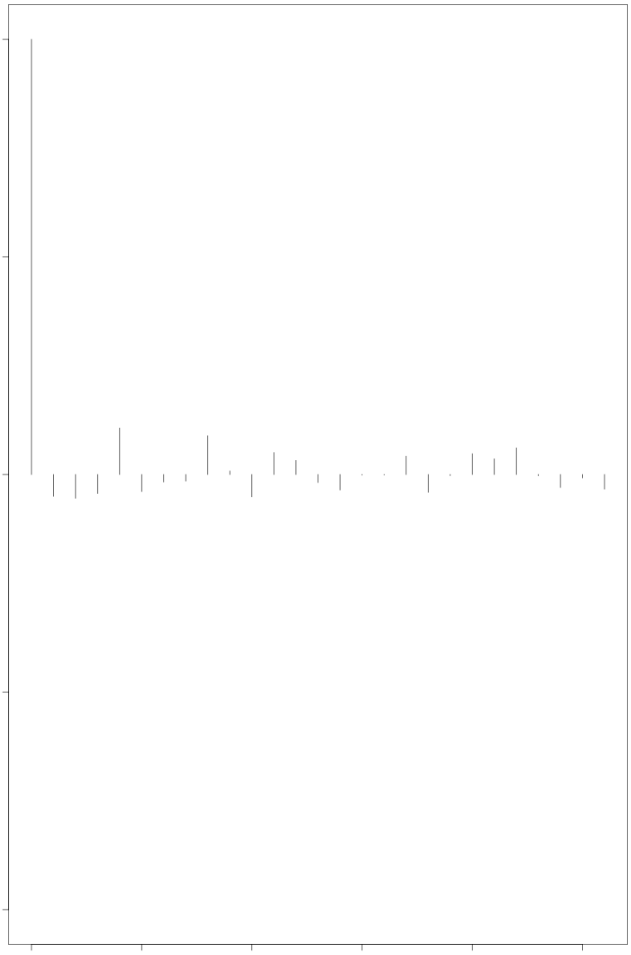


Figure 16: Autocorrelation plot for MCMC samples of β_1. As with β_0, the autocorrelation drops steeply after the first lag and remains near zero thereafter, confirming low intra-chain dependence and high sampling efficiency, implying that the posterior samples for β_1 are reliable and representative of the target distribution.

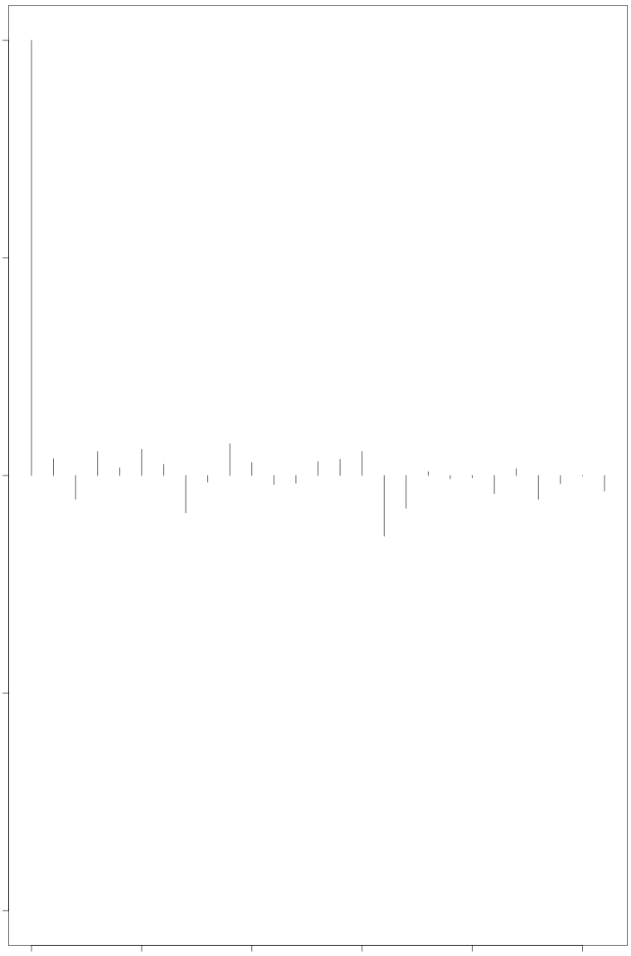


Figure 17: Autocorrelation plot for MCMC samples of β_2. The autocorrelation declines sharply after lag 1 and quickly levels off near zero for subsequent lags. This pattern indicates low dependence between successive samples and effective mixing of the chain, supporting the reliability of the posterior estimates for β_2.

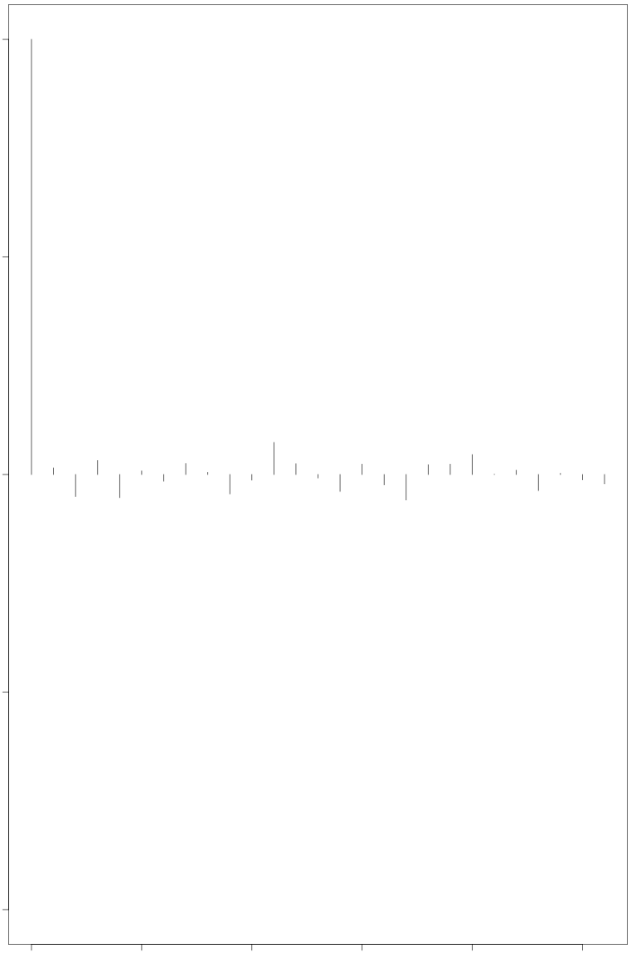


Figure 18: Autocorrelation plot for MCMC samples of β_3. The autocorrelation rapidly decays after lag 1 and remains close to zero for all subsequent lags, indicating minimal correlation between samples. This suggests that the MCMC chains for β_3 are mixing efficiently and producing effectively independent samples.

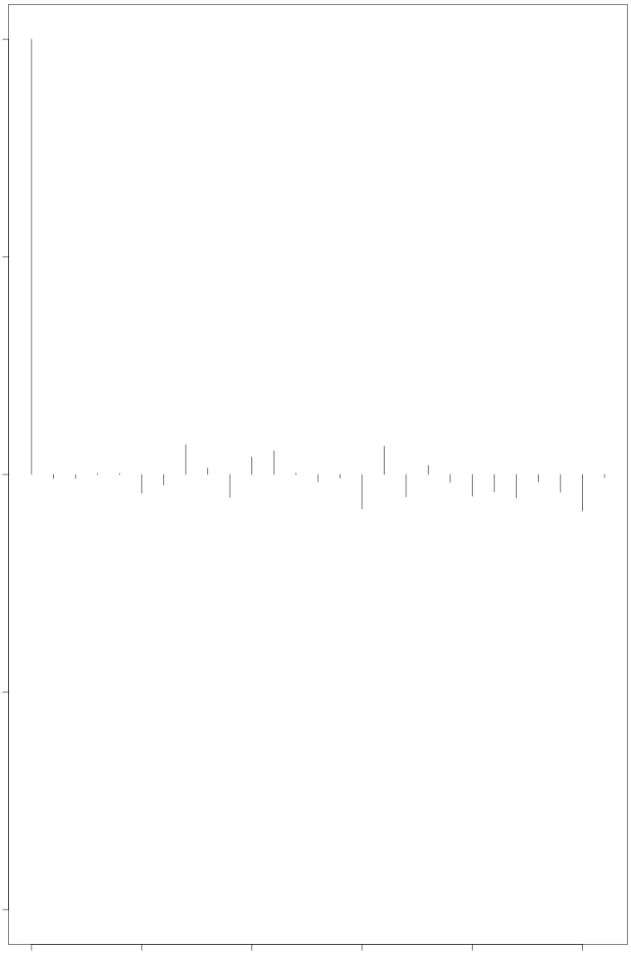


Figure 19: Autocorrelation plot for MCMC samples of τ (precision parameter). The autocorrelation drops steeply after lag 1 and fluctuates around zero for all subsequent lags, suggesting low serial correlation. This indicates that the MCMC sampler produced nearly independent draws, supporting efficient posterior estimation for the precision.

1.5 A discussion of the posterior distributions obtained from the analysis based on the plots and summary statistics (which need to be presented in the text). Discuss also how the posterior distributions of the model parameters can be used to predict new values of the response variable Y, given only values for the explanatory variables are available (prediction problem). Include both the interpretation of the parameter uncertainty and how it propagates into predictions for new observations.

The Bayesian linear regression model produced posterior distributions for all parameters; the intercept β_0, and the coefficients for cement (β_1), superplasticizer (β_2), and age (β_3), along with the residual precision parameter τ. The posterior summaries indicate how each predictor is associated with the concrete compressive strength after accounting for uncertainty and prior beliefs.

Table 2 presents the posterior mean, standard deviation, and the 95% credible interval for each parameter. None of the 95% credible intervals include 0, which suggests that all predictors have a statistically meaningful association with the response variable. The posterior distributions reveal meaningful associations between the predictors and the response variable. The cement coefficient (β_1) has a positive mean effect with a narrow credible interval, indicating a consistent and stable contribution of cement to compressive strength. The effect of superplasticizer (β_2) remains the most prominent, with its posterior centred substantially away from zero, suggesting a strong and reliably positive influence. The age coefficient (β_3) is modest but tightly concentrated, implying that even small increases in curing time yield measurable improvements in strength. The intercept term (β_0) exhibits greater variability, reflecting the model's baseline uncertainty when predictors are near zero. Finally, the residual precision parameter (τ) results in a relatively narrow distribution, indicating a well-estimated error structure and providing confidence in the model's predictive consistency. Collectively, these posterior characteristics reinforce the significance of each predictor while quantifying their uncertainty, supporting robust and interpretable inference.

Parameter	**Posterior Mean**	**SD**	**95% Credible Interval**
β_0 (Intercept)	5.069	1.105	[2.995, 7.325]
β_1 (Cement)	0.0689	0.0035	[0.062, 0.076]
β_2 (Superplasticizer)	1.112	0.063	[0.984, 1.243]
β_3 (Age)	0.0980	0.0061	[0.086, 0.110]
τ (Precision)	0.00689	0.00031	[0.00628, 0.00750]

Table 2: Posterior means, standard deviations, and 95% credible intervals.

The following code was implemented to predict new values of the response

variable:

```
1
2  #extract post samples -- generate a bunch of samples for each
       posterior distribution
3  samples_matrix <- as.matrix(post_burned)
4
5  #new values --
6  x_new <- c(300, 8, 28)
7
8  #compute mu for each -- scale the samples with the example values
9  mu_pred <- samples_matrix[, "beta0"] +
10 samples_matrix[, "beta1"] * x_new[1] +
11 samples_matrix[, "beta2"] * x_new[2] +
12 samples_matrix[, "beta3"] * x_new[3]
13
14 #distrib of pred mean --> distr of outcomes by adding resdl noise
15 tau_samples <- samples_matrix[, "tau"]
16 y_pred <- rnorm(length(mu_pred), mean = mu_pred, sd = 1 /
       sqrt(tau_samples))
17
18 print("Summary of prediction")
19 print(summary(y_pred))
20 print(quantile(y_pred, c(0.025, 0.5, 0.975)))
21
22
23 df <- data.frame(
24 predictive = y_pred,
25 expected = mu_pred
26 )
27
28
29 plt <- ggplot(df, aes(x = predictive)) +
30 geom_density(fill = "skyblue", alpha = 0.5) +
31 geom_vline(aes(xintercept = mean(predictive)), color = "blue",
       linetype = "dashed") +
32 geom_vline(aes(xintercept = quantile(predictive, 0.025)), linetype
       = "dotted") +
33 geom_vline(aes(xintercept = quantile(predictive, 0.975)), linetype
       = "dotted") +
34 labs(title = "Posterior Predictive Distribution",
35 x = "Predicted Strength (MPa)", y = "Density") +
36 theme_minimal()
37 plot(plt)
```

- To predict the compressive strength of a new concrete mix, the posterior distributions of the regression coefficients obtained from Markov Chain Monte Carlo sampling were used.
- Values for example predictor variables were specified (cement = 300, superplasticizer = 8, age = 28).
- For each posterior sample, a predicted linear mean μ_{pred} was computed using:

$$\mu_{\text{pred}} = \beta_0 + \beta_1 x_{\text{cement}} + \beta_2 x_{\text{superplasticizer}} + \beta_3 x_{\text{age}}$$

- Using μ_{pred}, the corresponding predicted value y_{pred} was then drawn from a normal distribution:

$$y_{\text{pred}} \sim \mathcal{N}(\mu_{\text{pred}}, \tau^{-1})$$

  where τ is the posterior precision from each sample.

- This procedure accounts for both parameter uncertainty and residual noise in generating predictive values.

- A summary of the predictive distribution, including mean and 95% credible interval, was computed and visualised.

Figure 20 shows the distributions of the expected mean response and posterior prediction. ThFigure 20 illustrates the distributions of the expected mean response and posterior predictions. The expected mean response is tightly concentrated, reflecting certainty around the average response. In contrast, the predictive distribution is flatter due to noise in the observations.e former is, as expected, tightly concentrated, thus reflecting certainty around the average response. The predictive distribution is flatter due to noise in the observations.

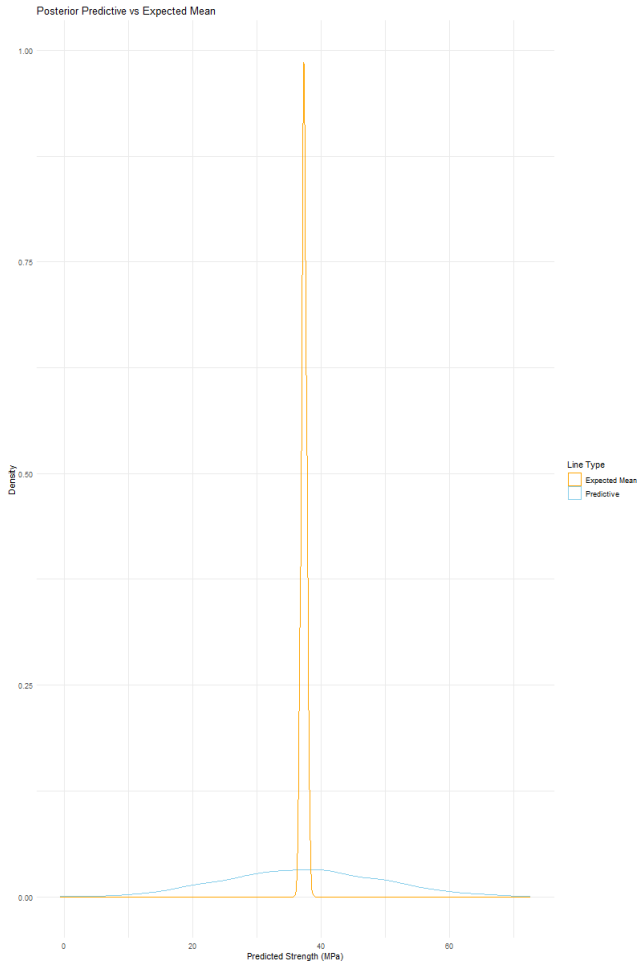


Figure 20: Comparison between the posterior predictive distribution and the distribution of the expected mean response. The narrow orange curve represents the distribution of the predicted mean response μ based on posterior samples of the model parameters. The wider blue curve shows the posterior predictive distribution for new observations, which incorporates both parameter uncertainty and residual noise. The increased spread in the predictive distribution reflects the variability expected in real-world observations.

1.6 Also present the R script, including comments that explain what each section does in an appendix.

- The code is available in the following GitHub repository: `https://github.com/carmelgafa/sci5020`.
- It was developed using Visual Studio Code alongside the VSCode R ex-

tension. Some minor precautions, such as printing to a variable, were necessary during development.

- The code was tested using R version 4.4.2.
- The code can be found in Appendix A

2 Question 2

Let $\mathbf{x} = (x_1, \ldots, x_n)'$ be a set of observations. Consider the following Bayesian model:

$$p(\mu, \sigma^2 \mid \mathbf{x}) \propto \left[\prod_{i=1}^{n} \mathcal{N}(x_i \mid \mu, \sigma^2)\right] \mathcal{N}(\mu \mid m_0, v_0^2)\Gamma(\tau \mid a, b), \quad \tau = \sigma^{-2}$$

We do not know explicitly the posterior here but we know through the theory of conjugate priors that:

$$p(\mu \mid \mathbf{x}, \sigma^2) \propto \left[\prod_{i=1}^{n} \mathcal{N}(x_i \mid \mu, \sigma^2)\right] \mathcal{N}(\mu \mid m_0, v_0^2) = \mathcal{N}\left(\frac{n\bar{x} + v_0^{-2}m_0}{n + v_0^{-2}}, \left(n + v_0^{-2}\right)^{-1}\right)$$

$$p(\tau \mid \mathbf{x}, \sigma^2) \propto \left[\prod_{i=1}^{n} \mathcal{N}(x_i \mid \mu, \sigma^2)\right] \Gamma(\tau \mid a, b) = \Gamma\left(a + \frac{n}{2}, \frac{1}{2}\sum_{i=1}^{n}(x_i - \mu)^2 + b\right)$$

Recap of problem statement:

Let $\mathbf{x} = (x_1, x_2, \ldots, x_n)'$ be defined as a vector of observed data points. A Bayesian model is considered, in which the data are assumed to be generated from a Normal distribution with an unknown mean μ and unknown variance σ^2:

$$x_i \mid \mu, \sigma^2 \sim \mathcal{N}(\mu, \sigma^2), \quad i = 1, \ldots, n$$

A Bayesian approach is adopted, and prior distributions are specified for the unknown parameters μ and σ^2. Specifically, a Normal prior is placed on the mean μ, and a Gamma prior is assigned to the precision $\tau = \sigma^{-2}$:

$$\mu \mid \tau \sim \mathcal{N}(m_0, v_0^2), \quad \tau \sim \text{Gamma}(a, b)$$

In this hierarchical Bayesian model, a Normal prior is assigned to the mean μ conditional on the precision τ, and a Gamma prior is placed on τ. This results in the following joint prior:

$$p(\mu, \tau) = p(\mu \mid \tau) \cdot p(\tau)$$

By using conjugate priors, it is ensured that the posterior distributions for μ and τ remain in standard, tractable forms (Normal and Gamma), thereby facilitating efficient sampling through the Gibbs sampling method.

The complete joint posterior distribution $p(\mu, \tau \mid \mathbf{x})$ is not known in closed form. However, the theory of conjugate priors allows the full conditional distributions to be derived analytically:

- The conditional posterior distribution of μ given τ and $\mathbf{x}$ is a Normal distribution:

$$\mu \mid \mathbf{x}, \tau \sim \mathcal{N}\left(\frac{n\bar{x} + v_0^{-2} m_0}{n + v_0^{-2}}, \left(n + v_0^{-2}\right)^{-1}\right)$$

- The conditional posterior distribution of τ given μ and $\mathbf{x}$ is a Gamma distribution:

$$\tau \mid \mathbf{x}, \mu \sim \text{Gamma}\left(a + \frac{n}{2},\ b + \frac{1}{2}\sum_{i=1}^{n}(x_i - \mu)^2\right)$$

These conditional distributions constitute the foundation of a Gibbs sampling algorithm, by which samples from the joint posterior can be obtained through iterative sampling.

The implementation of this Bayesian model is now undertaken through the following steps:

(i) **Simulation of data.** Fixed values are selected for μ and σ^2, and 100 observations are drawn from a Normal distribution with the specified parameters:

$$x_i \sim \mathcal{N}(\mu, \sigma^2), \quad i = 1, \ldots, 100$$

(ii) **Specification of priors.** Hyperparameters m_0, v_0^2, a, and b are chosen to reflect non-informative prior beliefs about μ and τ. For instance:

$$m_0 = 0, \quad v_0^2 = 10^6, \quad a = b = 0.01$$

(iii) **Execution of Gibbs sampling.** An initial value $\tau^{(0)}$ is selected. Gibbs sampling is then used to generate 10,000 samples from the joint posterior of μ and τ. At each iteration t, the following steps are performed:

$$\mu^{(t)} \sim \mathcal{N}\left(\frac{n\bar{x} + v_0^{-2} m_0}{n + v_0^{-2}}, \left(\tau^{(t-1)}(n + v_0^{-2})\right)^{-1}\right)$$

$$\tau^{(t)} \sim \text{Gamma}\left(a + \frac{n}{2},\ b + \frac{1}{2}\sum_{i=1}^{n}(x_i - \mu^{(t)})^2\right)$$

A suitable number of initial samples are discarded as burn-in.

(iv) **Assessment of convergence.** Diagnostic tools are used to evaluate whether the chains have converged and whether the burn-in period has been sufficient. In particular:

- A `traceplot` is produced for the sampled values of μ and τ.
- The `autocorrelation function (ACF)` is examined to assess correlation between successive samples.
- The `effective sample size (ESS)` is computed to estimate the number of effectively independent draws.

These diagnostics are used to determine the adequacy of convergence and burn-in.

(v) **Summarization of posterior distributions.** The `epdfPlot` function from the `EnvStats` package in R is employed to produce empirical posterior density estimates for μ, τ, and $\sigma^2 = \tau^{-1}$, using only post-burn-in samples.

2.1 Decide on values for μ and σ^2 and simulate 100 values from a normal distribution with this mean and variance.

Part (i): Simulation of Data

A dataset consisting of $n = 100$ observations was simulated from a normal distribution with mean $\mu = 5$ and variance $\sigma^2 = 4$.

The resulting histogram of the simulated data confirms the approximate bell-shaped structure expected of normal distributions, centered around the mean.

The following code was used for simulation and visualization:

```
1
2  ## Question i
3
4  mu <- 5
5  sigma_2 <- 4
6  sigma <- sqrt(sigma_2)
7  n <- 100
8
9  set.seed(1001)
10 x <- rnorm(n, mean = mu, sd = sigma)
11
12 # Save histogram object (does not plot automatically)
13 plt <- hist(x, breaks = 50, col = "skyblue", main = "Histogram of
       Simulated Data",
14 xlab = "x values", border = "white", freq = FALSE)
15 plot(plt)
16
17 curve(dnorm(x, mean = mu, sd = sigma), col = "blue", lwd = 2, add
       = TRUE)
```

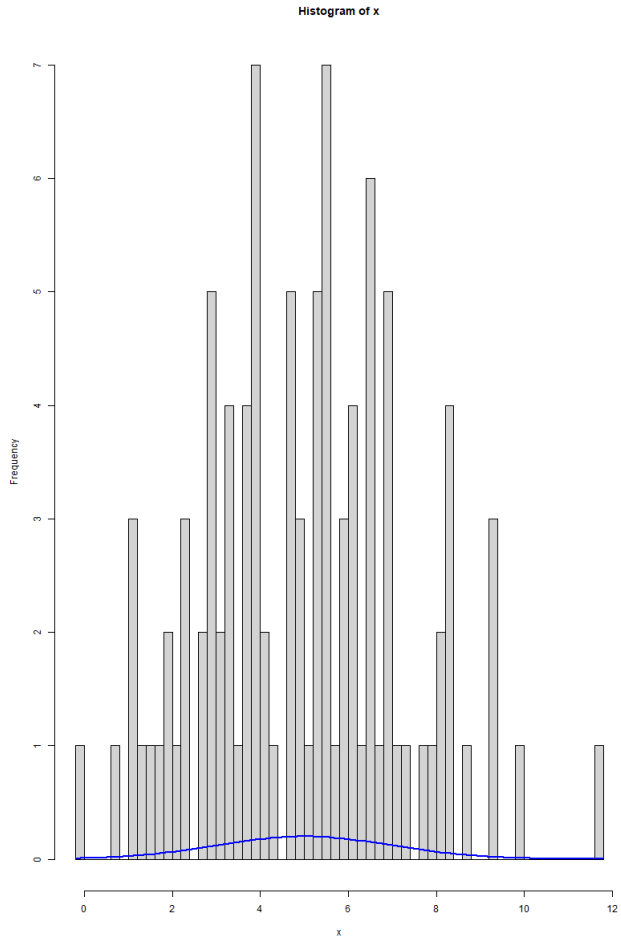


Figure 21: Histogram of 100 simulated observations from a normal distribution with mean $\mu = 5$ and variance $\sigma^2 = 4$. The blue curve represents the theoretical density function of $\mathcal{N}(5, 2^2)$ superimposed on the histogram.

2.2 Decide on hyperparameters m_0, v_0^2, a, b which yield a non-informative prior.

To use the Bayesian model, we need to define prior distributions for the unknown parameters μ and $\tau = \sigma^{-2}$. We will use a conjugate prior structure, which includes:

$$\mu \mid \tau \sim \mathcal{N}(m_0, v_0^2), \quad \tau \sim \text{Gamma}(a, b)$$

In this implementation, vague priors are used to show limited prior knowledge about the parameters. The following hyperparameter values are provided:

$$m_0 = 0, \quad v_0^2 = 10^6, \quad a = 0.01, \quad b = 0.01$$

This choice allows for a wide range of possible values for μ, giving more weight to the data when making inferences. The Gamma prior on τ has a small shape and rate, creating a flat and broad distribution over all positive real numbers.
These prior choices ensure that the posterior distributions remain proper while contributing little information relative to the likelihood. This is a common strategy when the objective is to let the data primarily inform the inference.

```
m0 <- 0
v0_sq <- 1e6
a <- 0.01
b <- 0.01
```

2.3 Pick an initial value of τ. Use Gibbs' sampling to simulate 10000 values of μ and τ (excluding an appropriately chosen burn-in period).

With the conditional posterior distributions derived analytically, a Gibbs sampling algorithm is implemented to simulate from the joint posterior of (μ, τ). The full conditionals are:

$$\mu \mid \mathbf{x}, \tau \sim \mathcal{N}\left(\frac{n\bar{x} + v_0^{-2} m_0}{n + v_0^{-2}},\ \left(\tau(n + v_0^{-2})\right)^{-1}\right)$$
$$\tau \mid \mathbf{x}, \mu \sim \text{Gamma}\left(a + \frac{n}{2},\ b + \frac{1}{2}\sum_{i=1}^{n}(x_i - \mu)^2\right)$$

The Gibbs sampler iteratively draws samples from the full conditional distributions of μ and τ, constructing a Markov chain that converges to the joint posterior $p(\mu, \tau \mid \mathbf{x})$. The procedure is implemented as follows:

- **Initialization**
  - Set the number of iterations T (10,000 in this case) and burn-in length (2000).
  - Create storage vectors for sampled values of μ and τ.
  - Choose an initial value for $\tau^{(0)}$, e.g., the reciprocal of the sample variance.
- **Repeat for each iteration** $t = 1, \ldots, T$**:**
  1. **Sample** $\mu^{(t)} \mid \tau^{(t-1)}, \mathbf{x}$**:**

- Compute the conditional posterior variance:

$$\text{Var}(\mu \mid \cdot) = \left(\tau^{(t-1)} n + \frac{1}{v_0^2}\right)^{-1}$$

- Compute the conditional posterior mean:

$$\text{Mean}(\mu \mid \cdot) = \text{Var}(\mu \mid \cdot) \cdot \left(\tau^{(t-1)} n \bar{x} + \frac{m_0}{v_0^2}\right)$$

- Draw $\mu^{(t)}$ from $\mathcal{N}(\text{Mean}, \text{Var})$.

2. **Sample $\tau^{(t)} \mid \mu^{(t)}, \mathbf{x}$:**
   - Compute the shape parameter:

$$a + \frac{n}{2}$$

   - Compute the rate parameter:

$$b + \frac{1}{2} \sum_{i=1}^{n} (x_i - \mu^{(t)})^2$$

   - Draw $\tau^{(t)}$ from $\text{Gamma}(\text{shape}, \text{rate})$.
3. **Store the draws:**
   - Save $\mu^{(t)}$ and $\tau^{(t)}$ to their respective storage vectors.

- **Post-sampling:**
  - Discard the first 2,000 iterations as burn-in to mitigate the effect of the initial values.
  - Use the remaining samples for inference and convergence diagnostics.

Code used for this question:

```
## Question iii

#sampler setting
n_iter <- 10000
burn_in <- 2000

#storeage for mu and tao
mu_samples <- numeric(n_iter)
tau_samples <- numeric(n_iter)

#init tao
tau_current <- 1 / var(x)  # start with reasonable value

x_bar <- mean(x)

for (t in 1:n_iter) {
        var_mu <- 1 / (tau_current * n + 1 / v0_sq)
```

```
[1] 4.9937
  2.5%    50%  97.5%
4.5471 4.9935 5.4460
[1] 5.354895
    2.5%      50%    97.5%
4.040837 5.283988 7.062571
```

Figure 22: Posterior summaries based on 8,000 post-burn-in samples from the Gibbs sampler. The estimated posterior mean of μ is approximately 4.99 with a 95% credible interval of [4.55, 5.45]. The posterior mean of σ^2 is approximately 5.35 with a 95% credible interval of [4.04, 7.06].

```
        mean_mu <- var_mu * (tau_current * n * x_bar + m0 / v0_sq)
        mu_current <- rnorm(1, mean = mean_mu, sd = sqrt(var_mu))

        #sample tao given mu
        shape_tau <- a + n / 2
        rate_tau <- b + sum((x - mu_current)^2) / 2
        tau_current <- rgamma(1, shape = shape_tau, rate =
            rate_tau)

        #store
        mu_samples[t] <- mu_current
        tau_samples[t] <- tau_current
}

#burn-in
mu_post <- mu_samples[(burn_in + 1):n_iter]
tau_post <- tau_samples[(burn_in + 1):n_iter]
sigma2_post <- 1 / tau_post  # for inference on variance


print(mean(mu_post))
print(quantile(mu_post, c(0.025, 0.5, 0.975)))
print(mean(sigma2_post))
print(quantile(sigma2_post, c(0.025, 0.5, 0.975)))
```

Figure 22 shows the posterior summary obtained after burn-in was executed.

1. The posterior estimate for μ is very close to the true value used in simulation: $\mu = 5$.

2. The posterior estimate for σ^2 is slightly higher than the true value ($\sigma^2 = 4$), but it falls within the 95% credible interval. This is expected given the relatively small sample size ($n = 100$). When the sample size was increased to $n = 1000$, the posterior estimate of σ^2 moved closer to the true value, as reflected in the summary statistics presented in Figure23

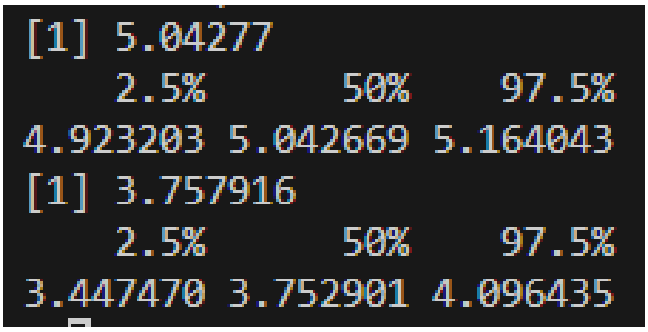


Figure 23: Posterior summaries based on 10,000 iterations of Gibbs sampling (with 2,000 burn-in discarded) using a simulated dataset of size $n = 1000$ from $\mathcal{N}(5, 4)$. The posterior mean of μ is 5.04, with a 95% credible interval of [4.92, 5.16], closely matching the true value $\mu = 5$. The posterior mean of σ^2 is 3.76, with a 95% credible interval of [3.45, 4.10], tightly capturing the true variance $\sigma^2 = 4$.

2.4 How can you use `traceplot`, the autocorrelation function and the effective sample size to determine whether the chain has converged, and whether the appropriate burn-in period has been eliminated? Use one or more of these techniques to determine whether your chain has converged and an adequate burn-in has been taken.

2.5 Make use of the `epdfPlot` command in the package `EnvStats` to determine the empirical pdf for μ, τ and σ^2 (of course, make sure that your chain has converged and do not include the burn-in period).

A Question 1 code

```
library(rjags)
library(lattice)
library(ggplot2)
library(coda)

#load dataset
script_dir <- getwd()
concrete_cleansed_path <- file.path(
script_dir,
"project_2_code/concrete_cleansed.csv"
)
concrete_cleansed <- read.csv(concrete_cleansed_path)

```

```
jags_data <- list(
x_cement = concrete_cleansed$cement,
x_superplasticizer = concrete_cleansed$superplasticizer,
x_age = concrete_cleansed$age,
y_strength = concrete_cleansed$strength,
n = nrow(concrete_cleansed)
)

#define model
model_description <- "
model {
        for (i in 1:n) {
                y_strength[i] ~ dnorm(mu[i], tau)
                mu[i] <- beta0 +
                (beta1 * x_cement[i]) +
                (beta2 * x_superplasticizer[i]) +
                (beta3 * x_age[i])
        }
        beta0 ~ dnorm(0, 0.01)
        beta1 ~ dnorm(0, 0.01)
        beta2 ~ dnorm(0, 0.01)
        beta3 ~ dnorm(0, 0.01)
        tau ~ dgamma(0.01, 0.01)
} "

#params
n_chains <- 3
n_burnin <- 6000
n_samples <- 15000
parameters_to_monitor <- c("beta0", "beta1", "beta2", "beta3",
    "tau")
initial_values <- list(
list(beta0 = -10, beta1 = 0.5, beta2 = 0.1, beta3 = 0.05, tau =
    0.5),
list(beta0 = 0,   beta1 = 1,   beta2 = 0.3, beta3 = 0.1,  tau = 1),
list(beta0 = 10,  beta1 = 2,   beta2 = 0.5, beta3 = 0.2,  tau = 2)
)

#create the jags model
model <- jags.model(
textConnection(model_description),
data = jags_data,
inits = initial_values,
n.chains = n_chains
)

#posterior samples from jgs model
post <- coda.samples(model = model,
variable.names = parameters_to_monitor,
n.iter = n_samples,
thin = 20)


#throw away burnin samples
post_burned <- window(post, start = n_burnin)

#posterior summary
```

```
print(summary(post_burned))

#autocorr mcmc samples
autocorr.plot(post_burned[, "beta0"])
autocorr.plot(post_burned[, "beta1"])
autocorr.plot(post_burned[, "beta2"])
autocorr.plot(post_burned[, "beta3"])
autocorr.plot(post_burned[, "tau"])

#density plts
plot(post_burned[, "beta0"], main="beta0 -- Intercept")
plot(post_burned[, "beta1"], main="beta1 -- Cement")
plot(post_burned[, "beta2"], main="beta2 -- Superplasticizer")
plot(post_burned[, "beta3"], main="beta3 -- Age")
plot(post_burned[, "tau"], main="tau -- Precision")


print(gelman.diag(post))
gelman.plot(post_burned[, "beta0"])
gelman.plot(post_burned)



print(effectiveSize(post_burned))

#extract post samples -- generate a bunch of samples for each
    posterior distribution
samples_matrix <- as.matrix(post_burned)

#new values --
x_new <- c(300, 8, 28)

#compute mu for each -- scale the samples with the example values
mu_pred <- samples_matrix[, "beta0"] +
samples_matrix[, "beta1"] * x_new[1] +
samples_matrix[, "beta2"] * x_new[2] +
samples_matrix[, "beta3"] * x_new[3]

#distrib of pred mean --> distr of outcomes by adding resdl noise
tau_samples <- samples_matrix[, "tau"]
y_pred <- rnorm(length(mu_pred), mean = mu_pred, sd = 1 /
    sqrt(tau_samples))

print("Summary of prediction")
print(summary(y_pred))
print(quantile(y_pred, c(0.025, 0.5, 0.975)))


df <- data.frame(
predictive = y_pred,
expected = mu_pred
)


plt <- ggplot(df) +
geom_density(aes(x = predictive, color = "Predictive")) +
geom_density(aes(x = expected, color = "Expected Mean")) +
```

```
labs(title = "Posterior Predictive vs Expected Mean",
x = "Predicted Strength (MPa)",
y = "Density",
color = "Line Type") +
theme_minimal() +
scale_color_manual(values = c("Predictive" = "skyblue", "Expected
    Mean" = "orange")) +
scale_x_continuous(limits = c(min(df), max(df)))
print(plt)


plt <- ggplot(df, aes(x = predictive)) +
geom_density(fill = "skyblue", alpha = 0.5) +
geom_vline(aes(xintercept = mean(predictive)), color = "blue",
    linetype = "dashed") +
geom_vline(aes(xintercept = quantile(predictive, 0.025)), linetype
    = "dotted") +
geom_vline(aes(xintercept = quantile(predictive, 0.975)), linetype
    = "dotted") +
labs(title = "Posterior Predictive Distribution",
x = "Predicted Strength (MPa)", y = "Density") +
theme_minimal()
plot(plt)
```