STSB6816 Test 3 of 2023

Mathematical Statistics and Actuarial Science; University of the Free State

2023/06/13

Time: 180 minutes; Marks: 50

MEMORANDUM

Instructions

- Answer all questions in a single R Markdown document. Please knit to PDF or Word at the end and submit both the PDF/Word document and the ".Rmd" file for assessment, in that order.
- Label questions clearly, as it is done on this question paper.
- All results accurate to about 3 decimal places.
- Show all derivations, formulas, code, sources, and reasoning.
- Intervals should cover 95% probability unless stated otherwise.
- No communication software, devices, or communication capable websites may be accessed prior to submission. You may not (nor even appear to) attempt to communicate or pass information to another student.

Question 1

The data is provided at https://ufs.blackboard.com. It contains data from an experiment on the "Pharmacokinetics of Theophylline". 12 subjects (Subject) were each weighed (Wt) and given a slightly different dose (Dose) of this substance at time 0. Their blood concentration (conc) was measured over time (Time). Your goal is to predict the log blood concentration curve of a random future subject.

We will assume that the log concentration curves follow the formula $\eta + \lambda t$. η (eta) measures where the curve would start if absorption was instantaneous, and λ (lambda) measures how the concentration drops over time (t).

We can then construct regression Model 1 by assuming an error distribution around the curve:

$$
\begin{aligned}
y_i &\sim t(\nu, \mu_i, \sigma),\ i = 1 \dots n \\
\mu_i &= \lambda_{s_i} t_i + \eta_{s_i} \\
\lambda_j &\sim N(\lambda_0, \tau_1^2),\ j = 1 \dots n_s \\
\eta_j &\sim N(\eta_0, \tau_2^2) \\
\ln\pi(\nu, \sigma, \tau_1, \tau_2, \lambda_0, \eta_0) &= -2\log\sigma + \log\nu - 3\log(\nu + 0.75) - 2\log\tau_1 - 2\log\tau_2 + k \\
\text{where } s_i &\text{ denotes the subject number of observation } i \\
n_s &\text{ denotes the number of subjects} \\
n &\text{ denotes the number of observations in total}
\end{aligned}
$$

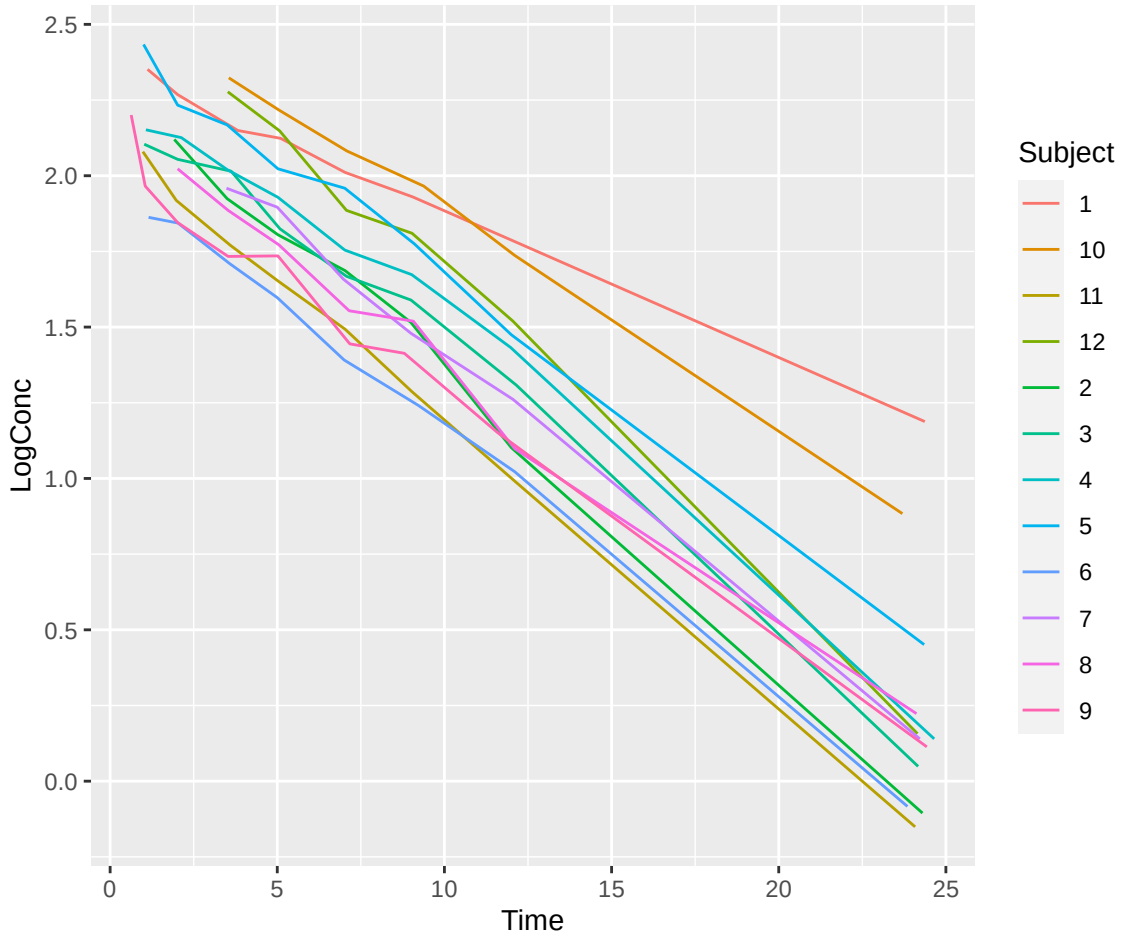
Note that Model 1 does not consider any explanatory variables other than the random effects induced by the assumption that each subject has their own curve. We are interested in the average curve, that will hopefully be indicative of a random future subject. Usually, one might model the correlation between the random intercept and random slope parameters explicitly, but the implied correlation will suffice today.

1.1) What does modelling the data on the log scale as in Model 1 imply with regard to the variation (in terms of standard deviation) around the curve on the two scales? **[3]**

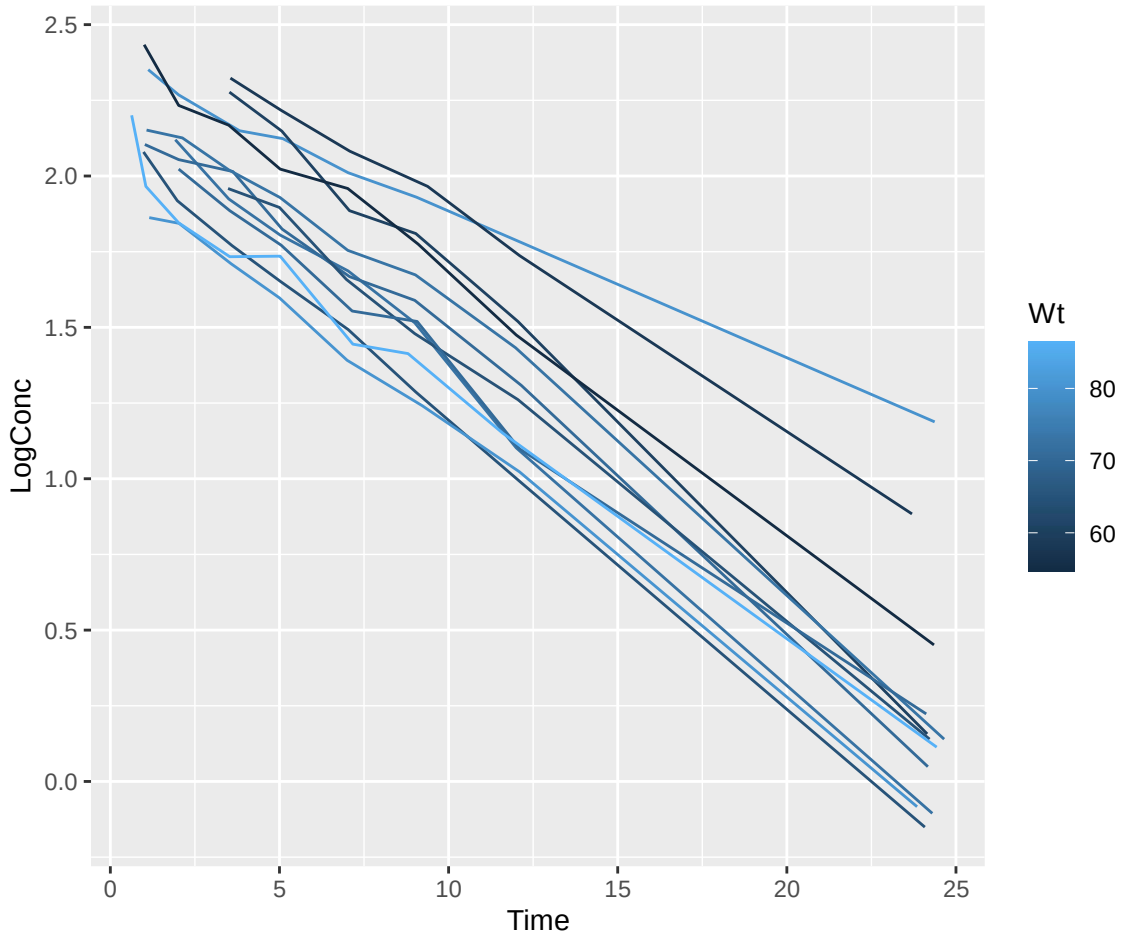
Discussion saying something about assuming a constant scale parameter around the line on the log scale [1], and that this implies a changing standard deviation on the original scale [2]. In this experiment the assumption seems valid.

1.2) Import the data set into R and explore it visually. You could draw line plots with a line for each subject, perhaps coloured by an explanatory variable; or a table of averages per subject next to their dose and weight. Discuss what you see. **[5]**

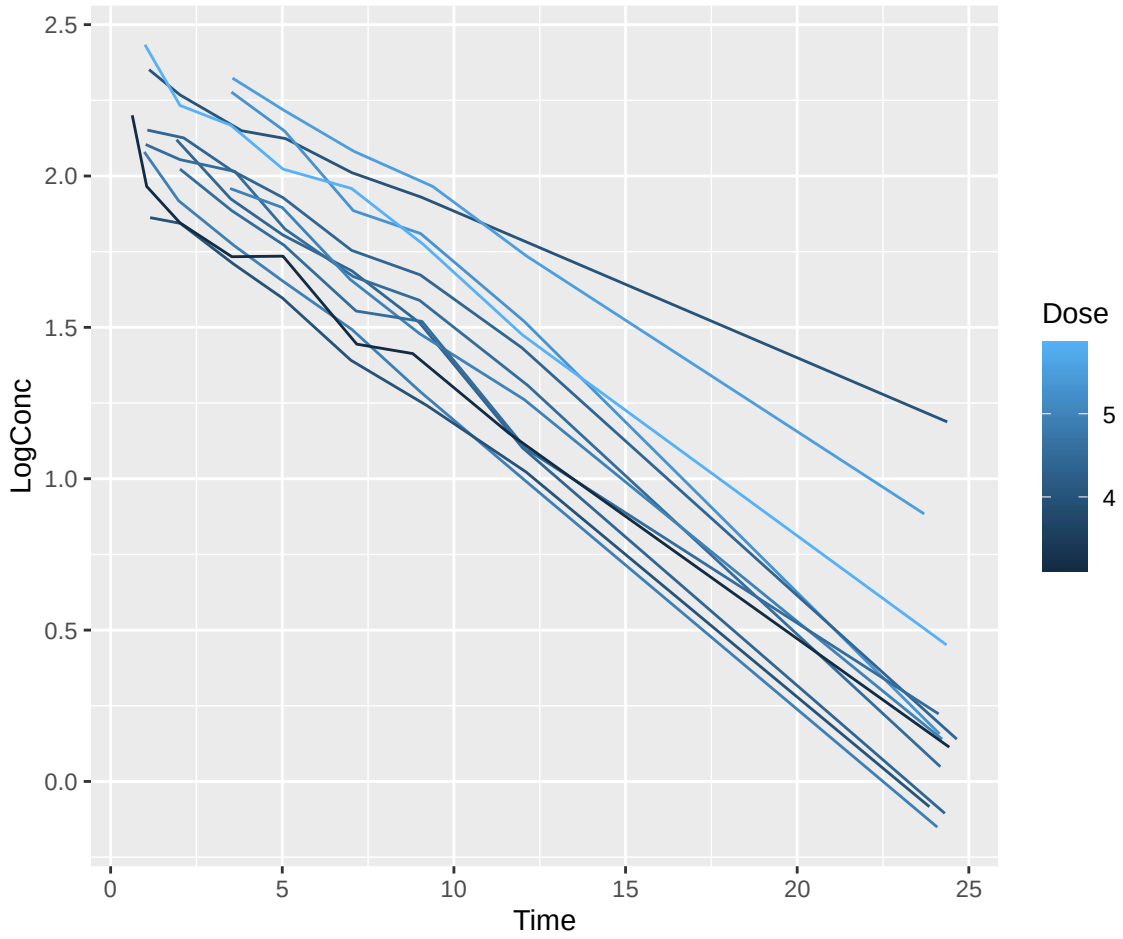
```
library(tidyverse)

"STSB6816Test3Data2023.xlsx" |> openxlsx::read.xlsx("TestData") -> d

d |> ggplot(aes(x = Time, y = LogConc, colour = Subject, group = Subject)) +
geom_line()
```




```
d |> ggplot(aes(x = Time, y = LogConc, colour = Wt, group = Subject)) + geom_line()
```




```
d |> ggplot(aes(x = Time, y = LogConc, colour = Dose, group = Subject)) +
geom_line()
```

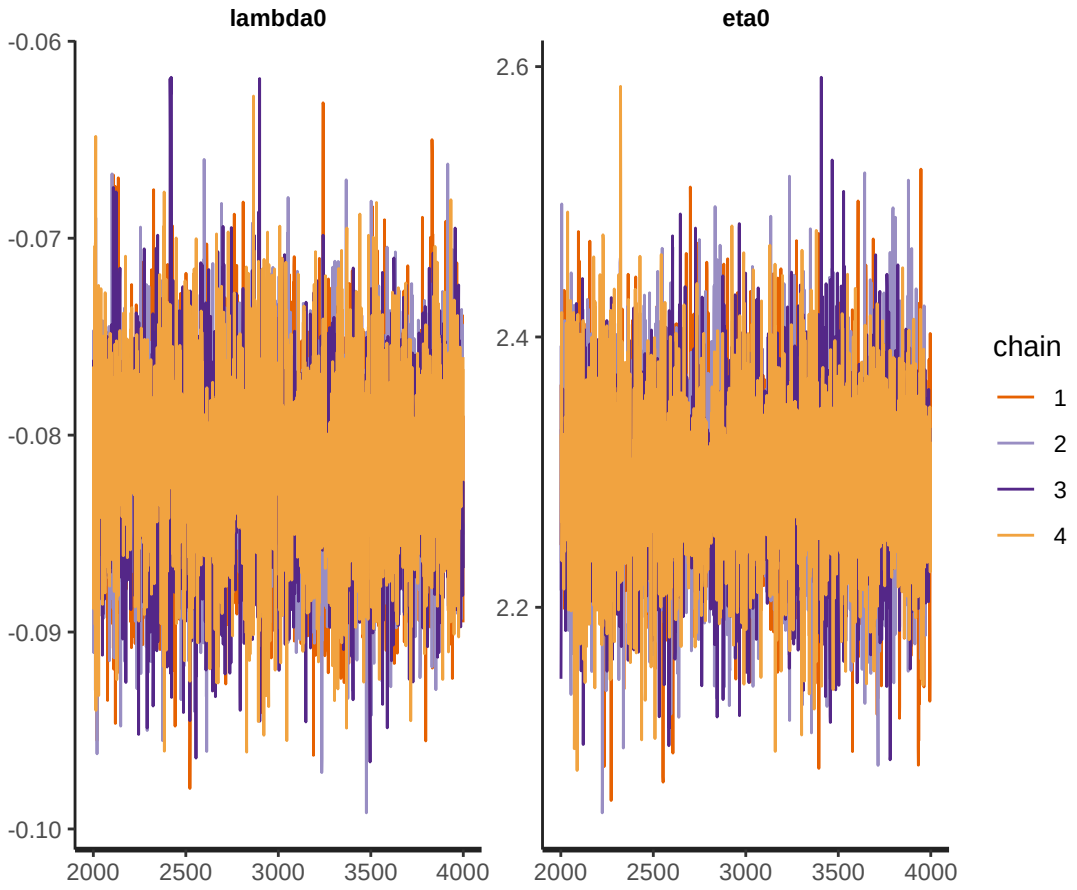


Loading data [1], line plot(s) or table(s) [2], and discussion saying something about higher doses having higher curves - significance cannot be determined yet [2].

1.3) Fit Model 1 on this data and discuss your estimates of η_0 and λ_0, along with their 95% intervals, in both statistical terms and practical terms. **[14]**

```
# First we load Stan:
library(rstan)
mycores <- max(1,floor(parallel::detectCores(logical = FALSE)*0.8))
options(mc.cores = mycores)
rstan_options(auto_write = TRUE)

// This Stan block defines a t regression model with random effects, by Sean van der
Merwe, UFS
data {
  int<lower=1> n;               // number of observations in total
  vector[n] y;                 // observations
  vector[n] time;
  int n_s;
  int subj_ind[n];
}
// The parameters of the model
parameters {
  real<lower = 0> sigma;          // error scale
  real<lower = 0.5> nu;           // error freedom
  real eta0;                        // intercept
  real lambda0;
  vector[n_s] lambda;
  vector[n_s] eta;
```

```
  real<lower=0> tau1;
  real<lower=0> tau2;
}
transformed parameters {
  vector[n] mu;
  for (i in 1:n) {
    mu[i] = eta[subj_ind[i]] + lambda[subj_ind[i]]*time[i];
  }
}
model {
  y ~ student_t(nu, mu, sigma);
  lambda ~ normal(lambda0, tau1);
  eta ~ normal(eta0, tau2);
  target += log(nu) - 3*log(nu + 0.75) - 2*log(sigma) - 2*log(tau1) - 2*log(tau2);
}
generated quantities {
  vector[n] log_lik;
  for (i in 1:n) {
    log_lik[i] = student_t_lpdf(y[i] | nu, mu[i], sigma);
  }
}

saveRDS(t_curves, file = 't_curves.Rds')

d$subjID <- d$Subject |> as.numeric()
n_s <- max(d$subjID)

t_curves |> sampling(data = list(n = nrow(d),
                                 y = d$LogConc,
                                 time = d$Time,
                                 n_s = n_s,
                                 subj_ind = d$subjID
                                ),
                     chains = mycores,
                     iter = 4000
                    ) -> Model1Fit

pars_of_interest <- c('lambda0', 'eta0')
Model1Fit |> traceplot(pars = pars_of_interest)
```




```
summary(Model1Fit, pars = pars_of_interest)$summary |> kable(digits = 3)
```

	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
lambda0	-0.081	0.000	0.005	-0.090	-0.084	-0.081	-0.079	-0.073	5998.293	1
eta0	2.295	0.001	0.060	2.175	2.257	2.294	2.332	2.418	8322.022	1

Specifying the fixed effect model components correctly, including the data, parameters, and model components relating to t regression [4]. Specifying the random effects model components correctly, including the data, parameters, and model components relating to random intercepts [2] and random slopes [2]. Implementing the model correctly using the provided data and giving a sensible summary of the key parameters [7]. Note that the generated log_lik is not required here and the associated marks fall under a later question.

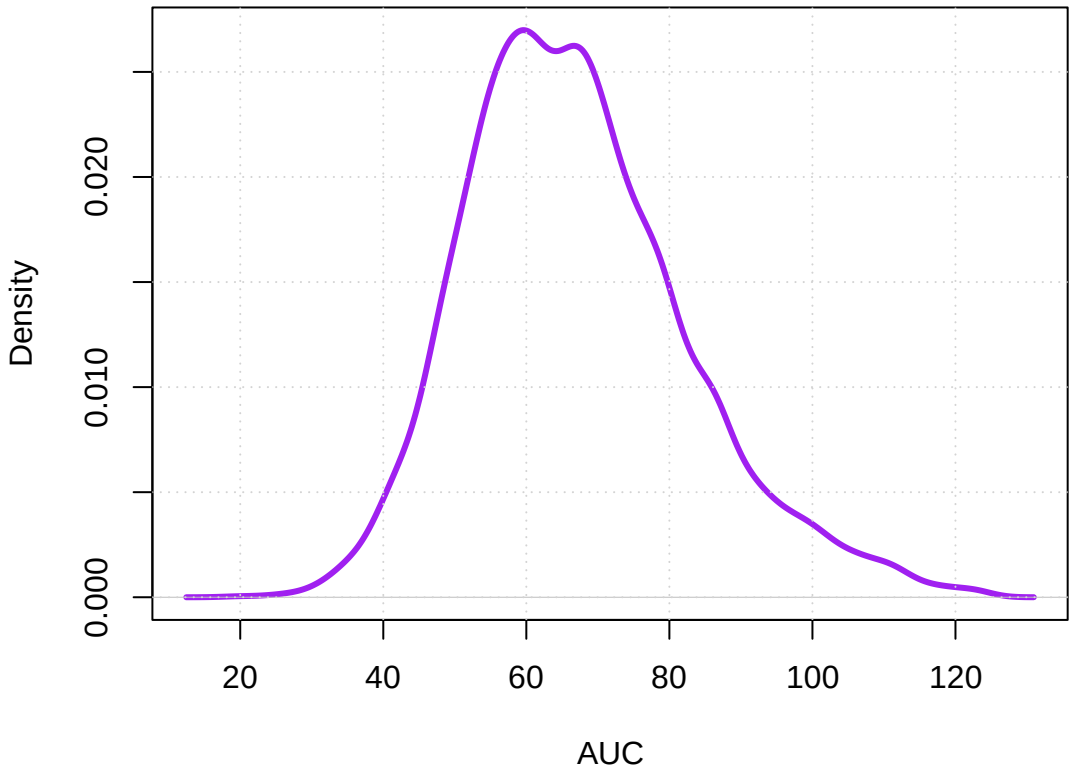
The bio-availability of the substance is related to the Area Under the Curve (AUC). We are most interested in the area under the blood concentration curve (not log) between hours 2 and 14 specifically. Assuming the model fits, the

$$AUC \approx 0.1 \sum_{i=1}^{121} exp\left(y_{t_i}^{new} | \mathbf{y}\right),\ t_i = 2, 2.1, 2.2, \ldots, 13.9, 14$$

1.4) Illustrate the posterior density of AUC for a random future subject. Please include only the lower 98% of predictions in any graphical illustration. [Partial credit will be given for a rough estimate of AUC.] **[8]**

```
sims <- rstan::extract(Model1Fit)
nsims <- length(sims$sigma)
```

```r
t_vec <- seq(2, 14, 0.1)
n_times <- length(t_vec)
new_lambda <- rnorm(nsims, sims$lambda0, sims$tau1)
new_eta <- rnorm(nsims, sims$eta0, sims$tau2)
new_mu <- new_lambda %*% t(t_vec) + new_eta
new_logy <- (rt(nsims*n_times, sims$nu)*rep(sims$sigma, n_times)) |> matrix(nsims) +
new_mu
AUC <- (new_logy |> exp() |> rowSums())*0.1
rm(new_lambda, new_eta, new_mu, new_logy)

AUC[AUC<quantile(AUC, 0.98)] |> density() |> plot(lwd = 3, main = '', col =
'purple', xlab = 'AUC')
grid()
```




```r
cat('A rough estimate of AUC is', 0.1*sum(exp(mean(sims$lambda0)*t_vec +
mean(sims$eta0))), 'while a more accurate estimate might be', median(AUC))
```

```
| A rough estimate of AUC is 65.13173 while a more accurate estimate might be 65.87401
```

Generating new random effects [2]. Implementing the linear expression at given times values [2]. Generating new random variation [2]. Implementing the AUC expression [2]. [Thus, a valid rough estimate can get up to 4 marks.]

Now consider the explanatory variables *weight* and *dose*. Including them as part of the intercept produces Model 2, which has the following changes:

$$\mu_i = \eta_{s_i} + \lambda_{s_i} t_i + \beta_1 * w_j + \beta_2 * d_j,\ j = 1 \ldots n_s$$

w_j denotes the standardised weight of subject j

d_j denotes the standardised dose of subject j

1.5) Standardise the explanatory variables using the mean-standard deviation approach, then fit the model with standardised explanatory variables and give estimates of those coefficients (betas). **[7]**

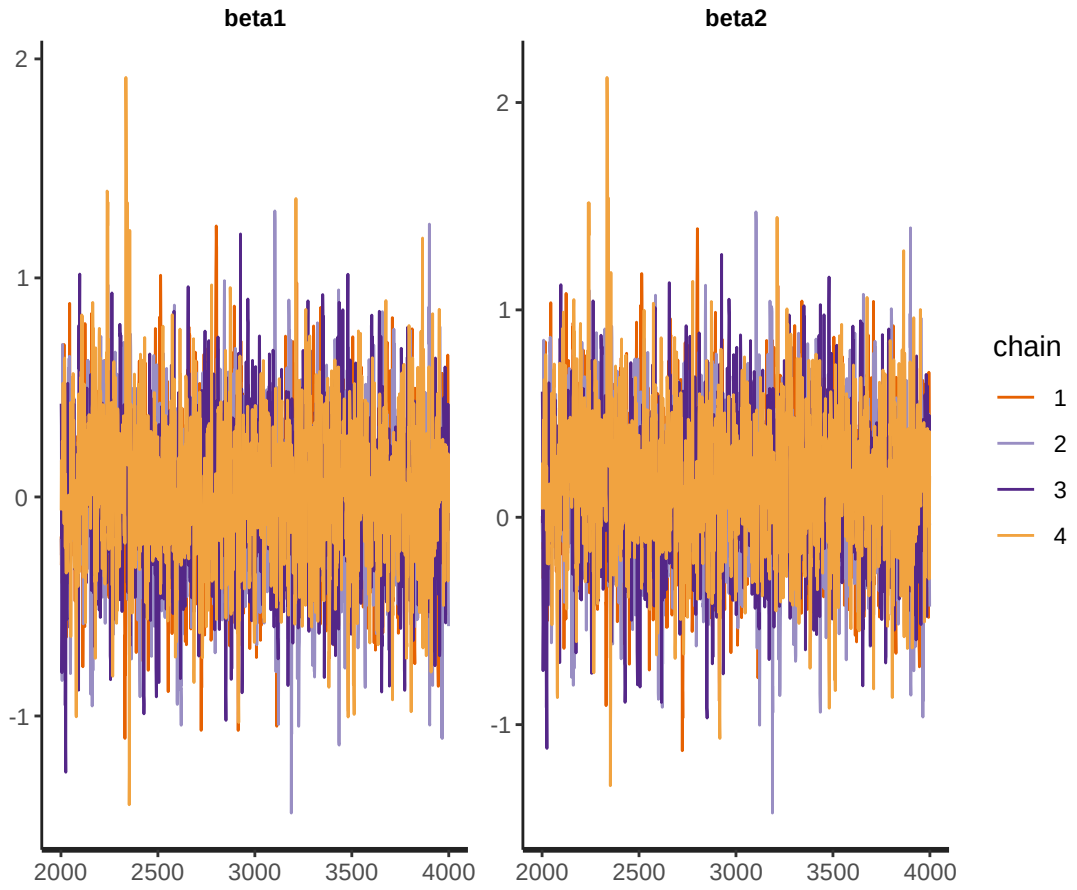
```
d$Wt_std <- d$Wt |> scale()
d$Dose_std <- d$Dose |> scale()

// This Stan block defines a t regression model with random effects and covariates,
by Sean van der Merwe, UFS
data {
  int<lower=1> n;              // number of observations in total
  vector[n] y;                // observations
  vector[n] time;
  vector[n] w;
  vector[n] d;
  int n_s;
  int subj_ind[n];
}
// The parameters of the model
parameters {
  real<lower = 0> sigma;          // error scale
  real<lower = 0.5> nu;           // error freedom
  real eta0;                        // intercept
  real lambda0;
  vector[n_s] lambda;
  vector[n_s] eta;
  real<lower=0> tau1;
  real<lower=0> tau2;
  real beta1;
  real beta2;
}
transformed parameters {
  vector[n] mu;
  for (i in 1:n) {
    mu[i] = eta[subj_ind[i]] + lambda[subj_ind[i]]*time[i] + beta1*w[i] +
beta2*d[i];
  }
}
model {
  y ~ student_t(nu, mu, sigma);
  lambda ~ normal(lambda0, tau1);
  eta ~ normal(eta0, tau2);
  target += log(nu) - 3*log(nu + 0.75) - 2*log(sigma) - 2*log(tau1) - 2*log(tau2);
}
generated quantities {
  vector[n] log_lik;
  for (i in 1:n) {
    log_lik[i] = student_t_lpdf(y[i] | nu, mu[i], sigma);
  }
}

saveRDS(t_expanded, file = 't_expanded.Rds')

t_expanded |> sampling(data = list(n = nrow(d),
                                   y = d$LogConc,
                                   time = d$Time,
                                   w = as.numeric(d$Wt_std),
                                   d = as.numeric(d$Dose_std),
```

```
                                n_s = n_s,
                                subj_ind = d$subjID
                               ),
                 chains = mycores,
                 iter = 4000
                ) -> Model2Fit
```

| Warning: There were 96 transitions after warmup that exceeded the maximum treedepth. Increase max_treedepth above 10. See
| https://mc-stan.org/misc/warnings.html#maximum-treedepth-exceeded

| Warning: Examine the pairs() plot to diagnose sampling problems

```
pars_of_interest <- c('beta1', 'beta2')
Model2Fit |> traceplot(pars = pars_of_interest)
```




```
summary(Model2Fit, pars = pars_of_interest)$summary |> kable(digits = 3)
```

	mean	se_mean	sd	2.5%	25%	50%	75%	97.5%	n_eff	Rhat
beta1	0.017	0.007	0.330	-0.656	-0.183	0.018	0.223	0.665	2016.527	1.001
beta2	0.160	0.007	0.333	-0.516	-0.039	0.161	0.367	0.807	2094.224	1.001

Adapting the model to use the variables correctly [2]. Standardising the two variables and sending them correctly to the model [2]. Implementing the model correctly using the provided data and giving a sensible summary of the beta parameters [3]. Note that the generated log_lik is not required here and the associated marks fall under a later question.

1.6) Compare the fit of the two models, and then explain what your model comparison implies regarding the significance of the explanatory variables as a set. **[6]**

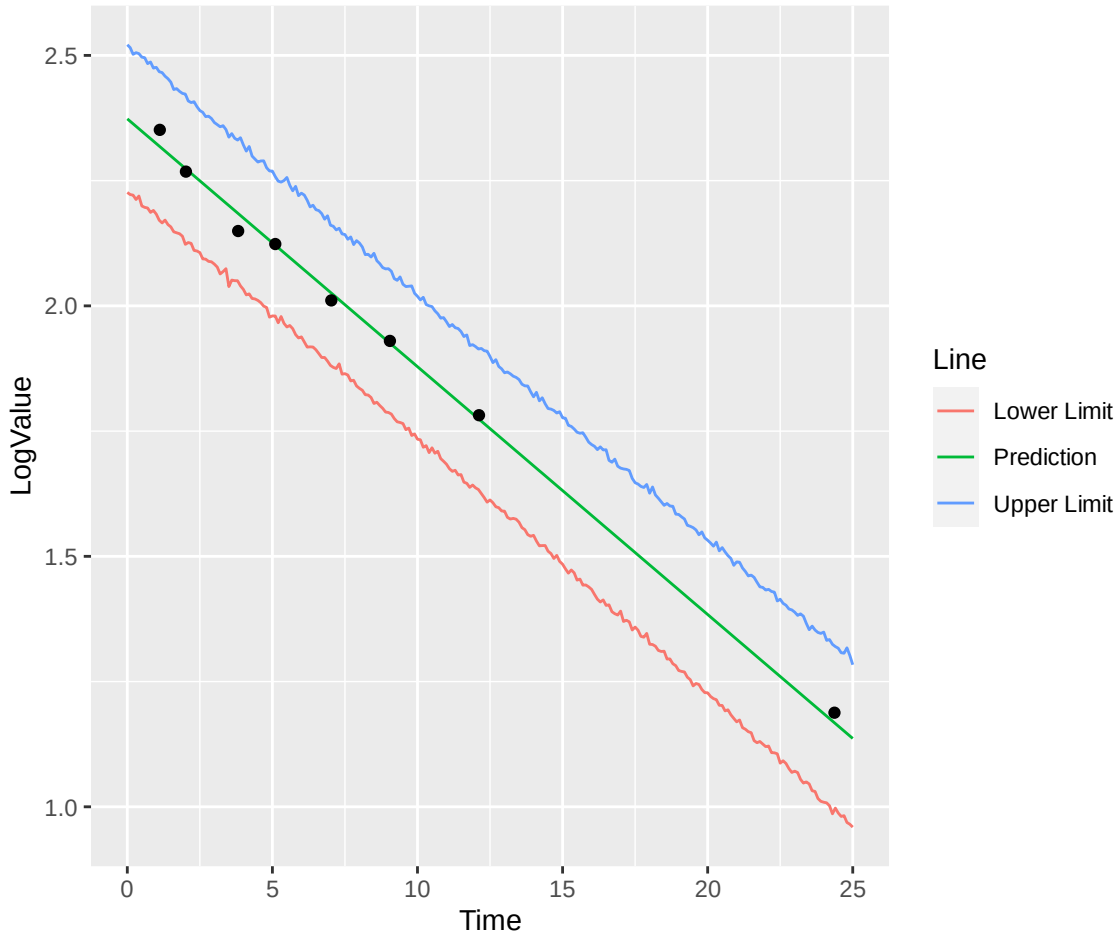
```
fits <- list(NoXs = Model1Fit, WithXs = Model2Fit)

library(loo)
fits |> lapply(\(fit) {extract_log_lik(fit, merge_chains = FALSE)}) -> log_lik
log_lik |> lapply(\(ll) {relative_eff(exp(ll), cores = 1)}) -> r_eff
fits |> length() |> seq_len() |>
  lapply(\(i) {loo(log_lik[[i]], r_eff = r_eff[[i]], cores = 1)}) |>
  loo_compare() -> comparison
rownames(comparison) <- names(fits)[order(rownames(comparison))]
comparison |> knitr::kable(digits = 1)
```

	elpd_diff	se_diff	elpd_loo	se_elpd_loo	p_loo	se_p_loo	looic	se_looic
WithXs	0	0.0	104.9	11.6	46.3	6.6	-209.7	23.1
NoXs	-1	0.8	103.9	11.4	46.6	6.7	-207.8	22.8

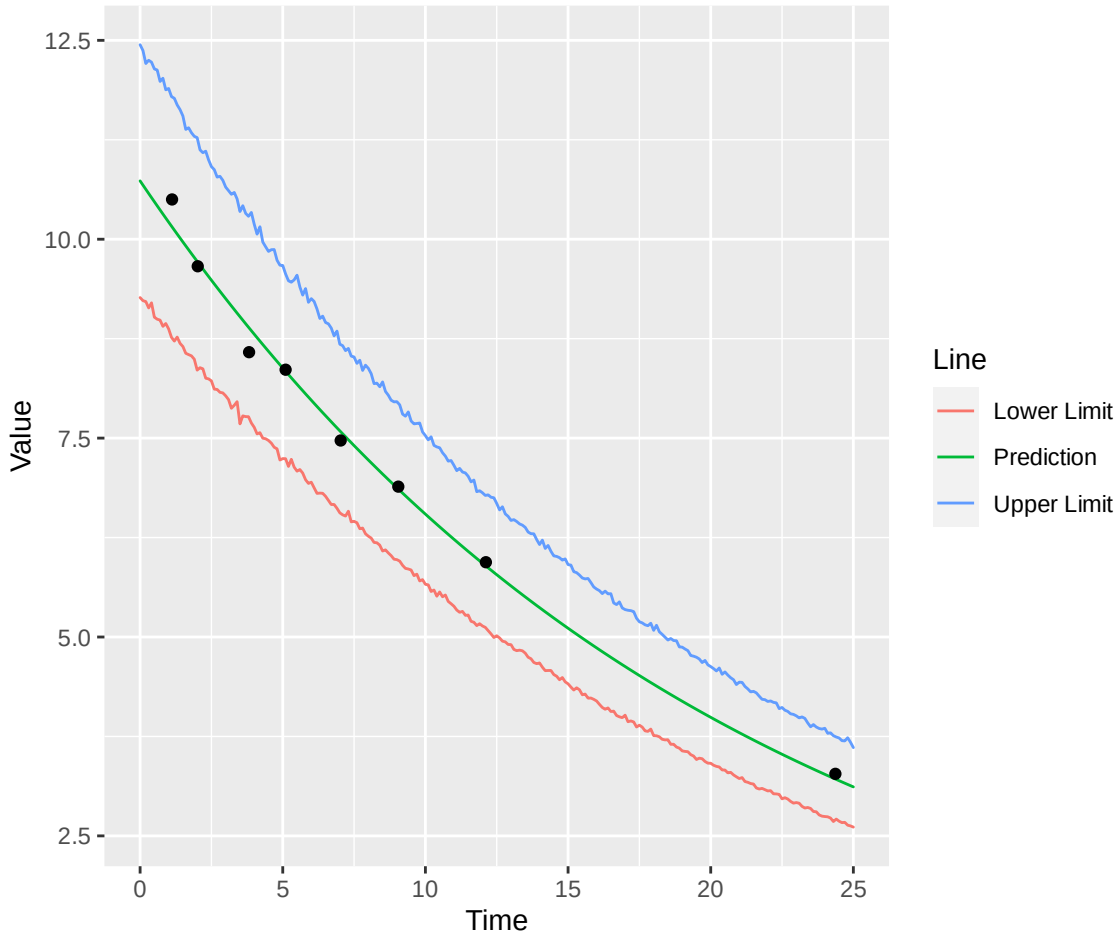
Calculating a suitable statistic for all models [3]. Note that most of these marks are for the code that calculates the log likelihoods, either in Stan or in R, not for the last bit of code above to get the statistics. Statement saying that the (correctly identified) model with the lowest criterion value is preferred [1]. Statement saying that the difference in criterion values is well within their standard errors, thus providing no evidence that the explanatory variables added value to the regression model [2].

1.7) Plot the data of any one observed subject from the experiment. On the same plot show the fitted curve of that subject and 95% prediction intervals around the curve. You may use either the log or original scale. **[7]**

```
sbj <- 1
plot_data_sbj <- d |> filter(subjID == sbj)
t_vec_long <- seq(0,25,0.1)
n_times_long <- length(t_vec_long)
sbj_mu <- sims$lambda[,sbj] %*% t(t_vec_long) + sims$eta[,sbj]
sbj_logy <- (rt(nsims*n_times_long, sims$nu)*rep(sims$sigma, n_times_long)) |>
matrix(nsims) + sbj_mu
middle <- colMeans(sbj_mu)
intervals <- sbj_logy |> apply(2, \(sims_at_t) {
  quantile(sims_at_t, c(0.025, 0.975))
}) |> t() |> c()
plot_data_curves <- data.frame(LogValue = c(middle, intervals),
                               Value = exp(c(middle, intervals)),
                               Time = rep(t_vec_long, times = 3),
                               Line = rep(c('Prediction','Lower Limit','Upper
Limit'),
                                          each = n_times_long))

plot_data_curves |> ggplot() +
  geom_line(aes(x = Time, y = LogValue, group = Line, colour = Line)) +
  geom_point(aes(x = Time, y = LogConc), data = plot_data_sbj)
```




```
plot_data_curves |> ggplot() +
  geom_line(aes(x = Time, y = Value, group = Line, colour = Line)) +
  geom_point(aes(x = Time, y = conc), data = plot_data_sbj)
```



Data of one subject plotted [2]. Estimate curve of that same subject plotted (not overall curve) [2]. Intervals generated and plotted for that subject, including available uncertainty [3].

Points total

The points on the test add up to **50**
