CHL5223 - Applied Bayesian Methods - Assignment 4

Luis Correia - Student No. 1006508566

April 6th 2020

Question 1

Simulating random variables

Introduction: For this problem, you are required to estimate the mean and variance for a random variable using some of the methods discussed in class. Consider a random variable X with density g(x). For each of the questions below, you are allowed to get random uniform variables using the R command `runif`. Also, you are just allowed to sample a total of 1000 uniform random variables. In each question, you will use a different technique which is described in the notes. For a complete answer, you should not simple provide the numeric value for your calculation. You need to describe why the algorithm you used is the correct algorithm.

Details of the random variable: Let X be a sample from the triangle distribution. That is, if $g(x)$ is the density function for X, then define $g(x)$ as:

$$g(x) = \begin{cases} 4x, & \text{for } 0.0 \leq x \leq 0.5 \\ 4 - 4x, & \text{for } 0.5 < x \leq 1.0 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

In the plot, this density is the solid line. Note, you could define the function $g(x)$ in R as:

```
g=function(x)(x>0)*(x<1)*((x<=0.5)*4*x+ (x>0.5)*(4-4*x))
```

Also, let f(x) be the density function for the uniform distribution from 0 to 1. Then, $f(x)$ is defined as:

$$f(x) = \begin{cases} 1, & \text{for } 0 \leq x \leq 1 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

In the plot, this density is the dashed line. Note that $2f(x) \geq g(x)$.

item (a)

Let U_1 and U_2 be two independent samples from a uniform distribution on $[0, 1]$. Define a sample random variable Z by $(U_1 + U_2)/2$. Note that Z is then a random sample from the triangle distribution and has density function g. Create a series of independent Z_i's in this manner in order to estimate $\mathbb{E}(X)$ and $\mathbb{V}ar(X)$.

{*Solution.*}

For these item the algorithm we will use is just generate the a new variable Z defined by $Z = \dfrac{U_1 + U_2}{2}$ with $U_1, U_2 \sim Unif[0, 1]$ to estimate $\mathbb{E}(X)$ and $\mathbb{V}ar(X)$ - details of implementation can be found on **Appendix**.

We have obtained the following results:

```
##
## ---- Summary Statistics for approx. Z=(U1+U2)/2 ---------------

##
## E(X) :  0.4978

##
## S.E.:  0.0062156

##
##
## 95% Credible Region for true-Mean ( 0.485336 , 0.510198 )

##
## Var(X) :  0.038634

##
## ------------------------------------------------------------
```

The histogram of the samples obtained by this method is as follows:

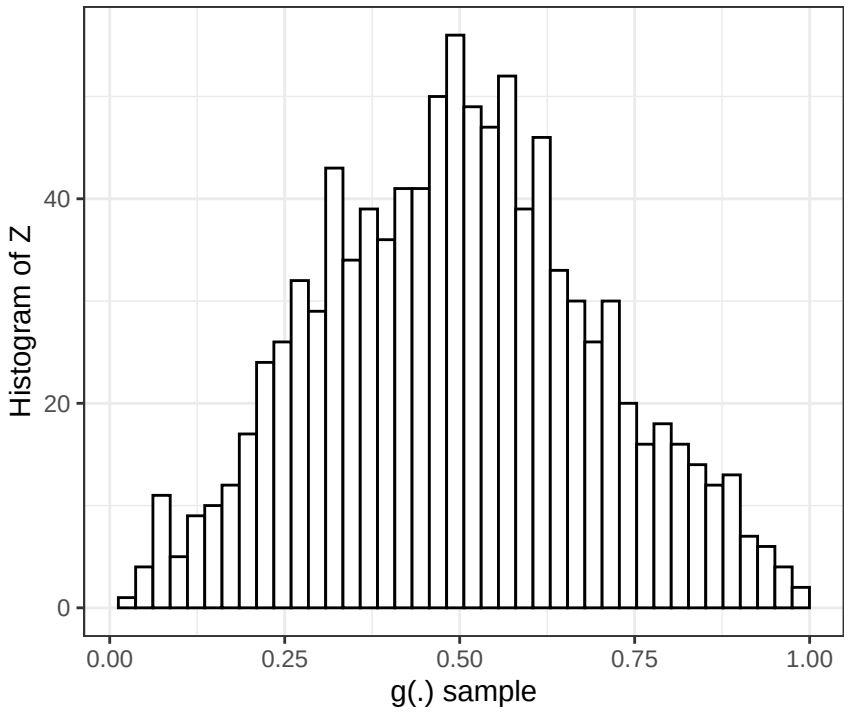

Figure 1: Histogram of simulated Z

item (b)

Just using sample from a uniform distribution on $[0, 1]$, use the importance sampler method. That is, do the following:

- i. Provide the weight function for the importance sampler when using sampled values from the uniform

distribution.

- ii. Give the estimates for $\mathbb{E}(X)$ and $\mathbb{V}ar(X)$. (Note: $\mathbb{V}ar(X) = \mathbb{E}(X^2) - [\mathbb{E}(X)^2]$.)

{*Solution.*}

Using the *Importance Sampling* method the *weight function* used was:

$$w(U_i) = g(U_i)/f(U_i), \text{ for } i = 1, \ldots, N \tag{3}$$

... where $U_i \sim Unif[0, 1]$ and $g(\cdot)$ is the *density function of the triangular distribution* defined in (1) and $f(\cdot)$ defined in (2).

This algorithm perfoms the estimation of $\mathbb{E}(X)$ via *Monte Carlo Integration Estimation*, and we can use the same approach to estimate the 2nd. momentum of X and then estimate $\mathbb{V}ar(X)$ - details of implementation can be found on **Appendix**.

We obtained the following results for estimates to $\mathbb{E}(X)$ and $\mathbb{V}ar(X)$:

```
##
## ---- Summary Statistics for Importance Sampling Algorithm -----

##
## E(X) :  0.5175

##
## S.E.:  0.010656

##
##
## 95% Credible Region for true-Mean ( 0.496215 , 0.538841 )

##
## Var(X) :  0.035897

##
## ----------------------------------------------------------------
```

As importance Sampling doesn't actually simulates a random variable distribution but consists in a *Monte Carlo Integration* method, we will skip from providing an histogram of simulated data in this case. Nonetheless, just for fun, we can estimate the error when calculating the integral under the density function $g(\cdot)$ which we expect to get a value close to 1, as N increases.

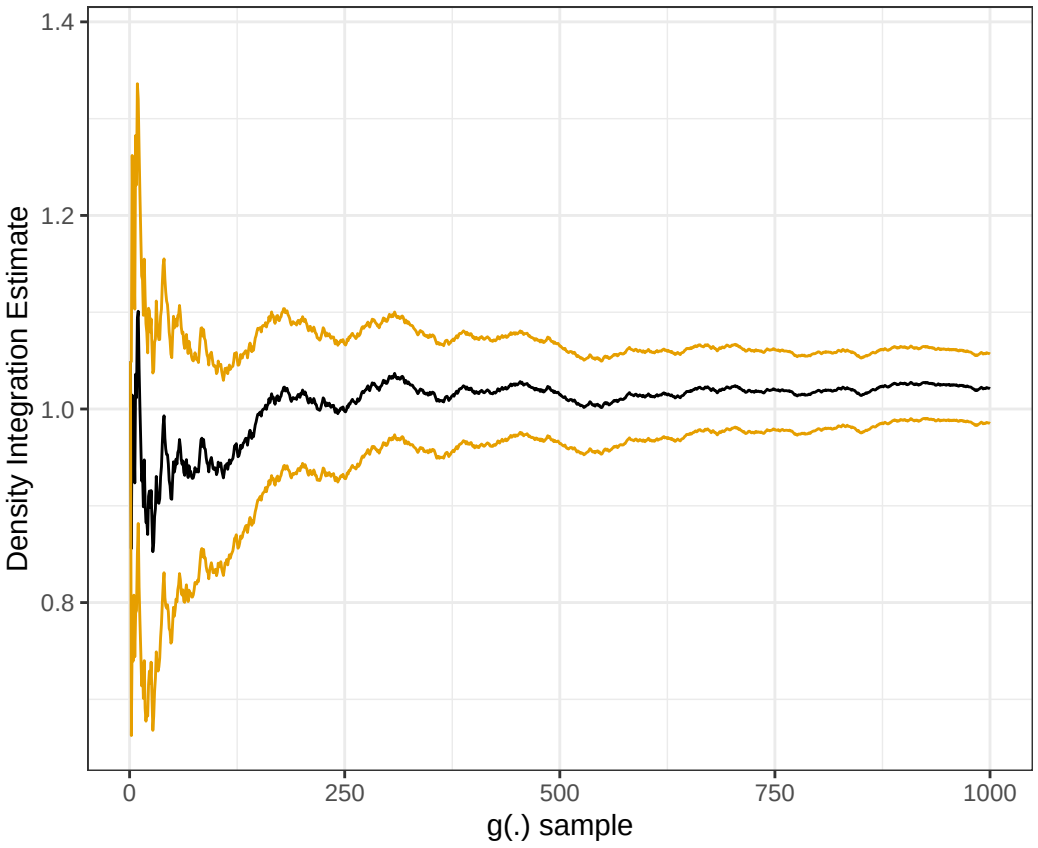

Figure 2: Estimated Density Integration obtained from Importance Sampling

item (c)

Using just samples from the uniform distribution, use the acceptance-rejection method to estimate $\mathbb{E}(X)$ and $\mathbb{V}ar(X)$. To do this, do the following:

- i. Generate a random variable X using the acceptance-rejection method. State how your algorithm works and that the acceptance test function is.
- ii. Give the rate of acceptance. That is, what percentage of proposed values is accepted.
- iii. Provide your estimates of $\mathbb{E}(X)$ and $\mathbb{V}ar(X)$ for this method.

{*Solution.*}

The Accept-Reject Method

The *pseudo-code* of Accept-Reject method applied to our problem can be summarized as follows:

Algorithm 1: Accept Reject Method

```
Set N = 1,000;
Set acc = rej = 0;
Set M = 2;
Set y = zeros(N);
while acc ≤ N do
    Propose X in support of g(·);
    begin
        Set x = runif(1, min = 0, max = 1);
        Set u = runif(1);
        if u ≤ g(x)/M then
            Increment acc;
            Set y[acc] = x;
        else
            Increment rej;
        end
    end
end
```

The corresponding `R-Code` can be found in **Appendix**.

Using this method, we have obtained the following results for the estimates to $\mathbb{E}(X)$ and $\mathbb{V}ar(X)$, including its *acceptance rate*:

```
##
## ---- Summary Statistics for Accept-Reject Algorithm -----

##
## E(X) :  0.4955

##
## S.E.:  0.0063373

##
##
## 95% Credible Region for true-Mean ( 0.482861 , 0.508211 )

##
## Var(X) :  0.040162

##
##
## Rate of Acceptance =  0.4926

##
## -------------------------------------------------------------
```

The histogram of the samples obtained by this method is as follows:

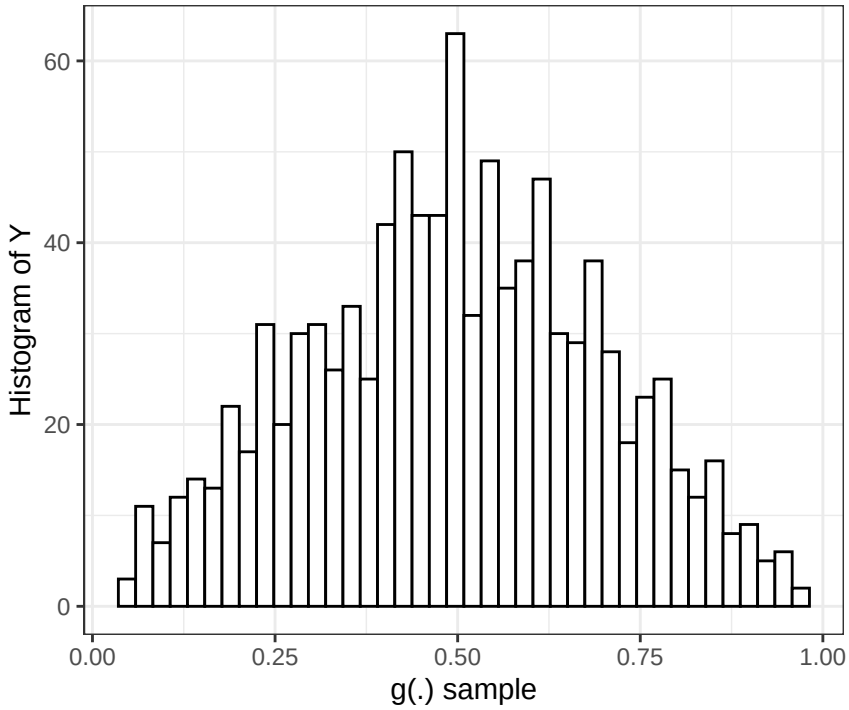

Figure 3: Histogram of simulated Accept/Reject

item (d)

Use the Metropolis-Hasting algorithm to generate an MCMC sequence of X_i's which have the triangle distribution as the invariant and the limiting distribution. Do this by doing the following:

- Let the transition function, $q(x, y)$, be uniform density and have it not depend on the value of x. (That is, the new proposed move is always a sample from the uniform distribution on $[0, 1]$ and this does not depend on the previous location. Therefore, $q(x, y) = 1$ for all values of x and for $y \in [0, 1]$.
- The invariant distribution is the triangle distribution. So, $u(x) = g(x)$.
- Don't burn in the chain and don't thin the chain.

  i What is the test function, $\alpha(x, y)$, for this chain given the above information?

  ii Provide the R code which samples the chain.

  iii What is the acceptance rate for the proposed moves in this chain?

  iv What are your estimates of $\mathbb{E}(X)$ and $\mathbb{V}ar(X)$?

{*Solution.*}

Using the *Metropolis-Hastings* method the *test function* $\alpha(x, y)$ used was:

$$\alpha(x, y) = \min\left[\frac{g(x)}{g(y)}, 1\right] \tag{4}$$

... where $g(\cdot)$ is the *density function of the triangular distribution* defined in (1).

The R-Code for M-H algorithm is as follows:

```
# Metropolis-Hastings Algorithm
alpha = function(x, y){
  min(1, g(x) / g(y))
}
x = rep(0, N)
acc <- 0

# Loop Sampling from the Chain
for(t in 2:N){
  ystar  <- runif(1, min=0, max=xrange)
  T <- runif(1)
  if( T <= alpha(ystar,x[t-1]))
  {
    x[t] <- ystar
    acc <- acc+1
  }
  else x[t] <- x[t-1]
}
```

... and we obtained the following results for the estimates to $\mathbb{E}(X)$ and $\mathbb{V}ar(X)$, including the *acceptance rate*:

```
##
## ---- Summary Statistics for Metropolis-Hastings Algorithm -----

##
## E(X) :  0.5030

##
## S.E.:  0.0066051

##
##
## 95% Credible Region for true-Mean ( 0.489791 , 0.516211 )

##
## Var(X) :  0.043628

##
##
## Rate of Acceptance =  0.6590

##
## ----------------------------------------------------------------
```

The histogram of the samples obtained by this method is as follows:

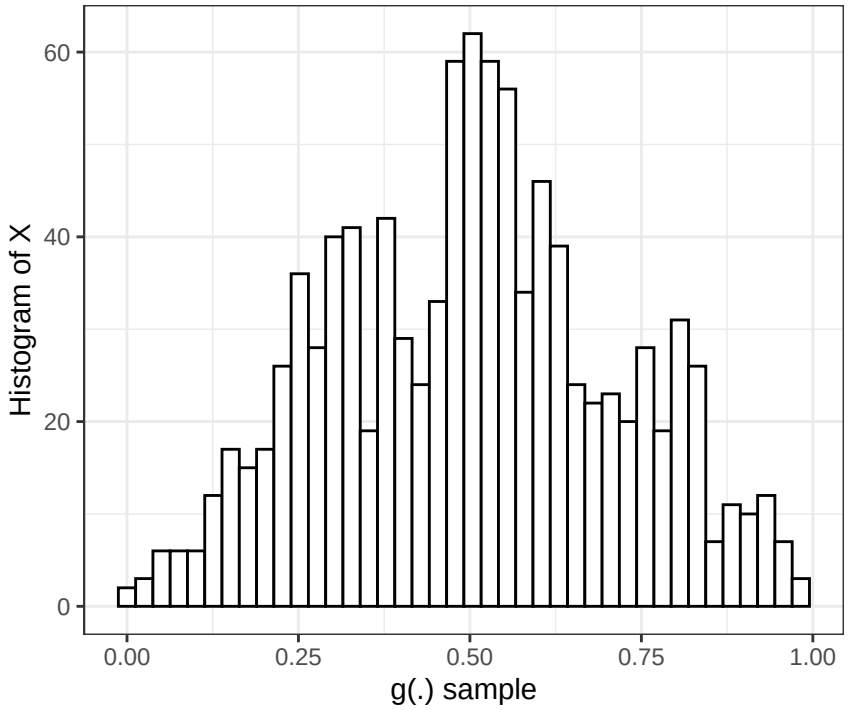

Figure 4: Histogram of simulated Metropolis-Hastings

Comparison between each method's estimate for $\mathbb{E}(X)$

We plotted each 95% Credible Region obtained on each method and the results are as follows:

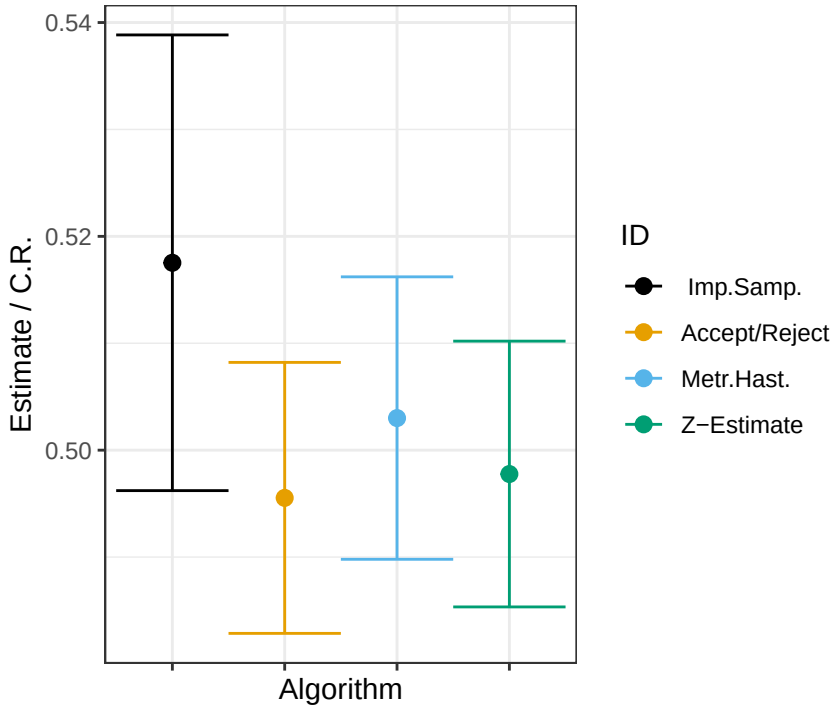

Figure 5: Comparative C.R. for E(X) between each method

Comment:- Comparing the four methods we can see that *Accept/Reject* and *Metropolis-Hastings* have provided very similar estimates and respective Credible Regions for $\mathbb{E}(X)$. The estimate that most approximate the *true-mean* was *Z-Estimate*[1]. On the other hand, *Importance Sampling* generated the poorest estimate of all methods for $\mathbb{E}(X)$ and the most elastic Credible Region, as well.

[1] Composed by $Z = \frac{U_1+U_2}{2}$, where U_1 and U_2 are $Uniform[0,1]$

Question 2

In this problem we will analysis an experiment which studies the effect of a dose of a drug on the growth of rats. The data is in the file `BigRatDat.txt` and is described below. This data consists of the growth of 50 rats, where 10 rats were randomly assigned to five different doses of a drug. (The dose levels are 0, .5, 1, 4, and 8 units of the drug.) The weights of the rats were obtained each week for 11 weeks. The data file is sent along with the file for this homework. The data file has the following structure:

```
Column 1: Doses levels
Column 2: Rat number (note: the rats are different for the different dose levels.
Column 3: Week 1 weight
Column 4: Week 2 weight
Column 5: Week 3 weight
Column 6: Week 4 weight
Column 7: Week 5 weight
Column 8: Week 6 weight
Column 9: Week 7 weight
Column 10: Week 8 weight
Column 11: Week 9 weight
Column 12: Week 10 weight
Column 13: Week 11 weight
```

The first several lines of the file `BigRatDat.txt` are below:

```
0 1 54 60 63 74 77 89 93 100 108 114 124
0 2 69 75 81 90 97 120 114 119 126 138 143
0 3 77 81 87 94 101 110 117 124 134 141 151
0 4 64 69 77 83 88 96 104 109 120 123 131
0 5 51 58 62 71 74 81 88 93 99 103 113
0 6 64 71 77 89 90 100 106 114 122 134 139
0 7 80 91 97 101 111 119 129 131 137 147 154
0 8 79 85 89 99 104 105 116 121 132 139 147
0 9 77 82 88 92 101 109 119 127 135 144 158
0 10 79 84 91 98 107 114 119 131 137 146 155
.5 1 62 71 75 79 87 91 100 105 111 121 124
.5 2 68 73 81 89 94 101 110 114 123 132 139
.5 3 94 102 109 110 128 133 147 151 153 171 184
.5 4 81 90 95 102 109 120 128 137 141 154 160
```

Do the following:

item (a)

Model this data with a growth curve model. So, for the ith rat, you should fit a regression line with the basic form: $Y_{ij} = \beta_{0i} + \beta_{1i} \times week_j + \epsilon_{ij}$. For the β's, you should model them as a linear relationship with the dose level. For example, you should model β_{0i} as a normal distribution with some precision and with $\mathbb{E}[\beta_{0i}] = \beta_{00} + \beta_{01} \times \text{DoseLevel}_i$. The coefficient β_{1i} is modelled similarly. (That is, the β_{0i} and β_{1i} each follow a normal distribution with some mean and precision.) Pick priors which are not restrictive. Show the WinBugs/OpenBUGS/JAGS code used to model this data. (If you run this through R, then please provide this code also.)

{*Solution.*}

Table 1: Data - Growth in Rats under Treatment

dose	IDinDose	week1	week2	week3	week4	week5	week6	week7	week8	week9	week10	week11
0.0	1	54	60	63	74	77	89	93	100	108	114	124
0.0	2	69	75	81	90	97	120	114	119	126	138	143
0.0	3	77	81	87	94	101	110	117	124	134	141	151
0.0	4	64	69	77	83	88	96	104	109	120	123	131
0.0	5	51	58	62	71	74	81	88	93	99	103	113
0.0	6	64	71	77	89	90	100	106	114	122	134	139
0.0	7	80	91	97	101	111	119	129	131	137	147	154
0.0	8	79	85	89	99	104	105	116	121	132	139	147
0.0	9	77	82	88	92	101	109	119	127	135	144	158
0.0	10	79	84	91	98	107	114	119	131	137	146	155
0.5	1	62	71	75	79	87	91	100	105	111	121	124
0.5	2	68	73	81	89	94	101	110	114	123	132	139
0.5	3	94	102	109	110	128	133	147	151	153	171	184
0.5	4	81	90	95	102	109	120	128	137	141	154	160
0.5	5	64	69	72	76	84	89	97	103	108	114	124
0.5	6	67	74	81	81	84	95	100	109	119	128	130
0.5	7	73	80	86	89	97	101	110	116	117	135	141
0.5	8	71	74	82	84	93	97	102	113	119	124	131
0.5	9	69	74	79	89	94	100	107	113	124	134	139
0.5	10	60	62	67	74	78	85	92	103	112	121	130
1.0	1	59	63	66	75	80	87	99	104	110	115	124
1.0	2	56	66	70	81	77	88	96	100	113	120	130
1.0	3	71	77	84	80	97	106	111	109	128	133	140
1.0	4	59	64	69	76	85	88	96	104	110	119	126
1.0	5	65	70	73	77	85	92	96	101	111	118	121
1.0	6	61	69	77	81	89	92	107	111	118	127	132
1.0	7	80	86	95	99	106	113	127	131	142	150	160
1.0	8	74	80	84	90	99	101	108	117	126	133	140
1.0	9	71	79	88	90	98	102	116	121	127	139	142
1.0	10	69	75	80	86	96	97	104	113	122	129	138
4.0	1	64	71	79	82	85	94	103	110	113	122	125
4.0	2	53	57	61	72	74	76	81	91	99	100	105
4.0	3	64	69	76	85	89	96	104	108	116	120	128
4.0	4	68	69	78	82	91	97	104	108	115	122	132
4.0	5	69	74	80	85	90	99	104	114	123	129	133
4.0	6	85	91	98	100	105	104	118	121	130	141	141
4.0	7	75	82	85	92	99	107	112	125	130	137	146
4.0	8	57	61	65	68	77	81	87	91	95	101	107
4.0	9	69	72	77	80	84	91	96	103	109	116	125
4.0	10	66	68	76	81	88	95	103	106	112	119	130
8.0	1	57	64	70	76	80	90	93	99	105	113	118
8.0	2	62	67	74	83	87	93	104	108	114	124	129
8.0	3	60	68	73	80	83	94	101	106	112	122	131
8.0	4	64	66	76	81	91	100	102	111	120	128	136
8.0	5	57	60	67	73	67	64	75	85	89	98	105
8.0	6	78	83	89	99	105	113	117	128	132	139	149
8.0	7	81	81	92	100	108	119	120	133	138	149	157
8.0	8	46	47	51	55	63	65	68	74	78	85	90
8.0	9	69	72	74	76	77	82	82	90	95	101	103

8.0	10	67	77	83	83	92	99	104	108	114	118	129

The model[2][3] created to this question is as follows:

Openbugs Model

```
model{
  for (i in 1:N) {
    for (j in 1:nj) {
      y[i,j] ~dnorm(mu[i,j], tau.w)
      mu[i,j] <- beta0[i]+beta1[i]*week[j]
    }
    beta0[i]~dnorm(mu0[i], tau.b0)
    beta1[i]~dnorm(mu1[i], tau.b1)
    mu0[i] <- beta00+beta01*doselevel[i]
    mu1[i] <- beta10+beta11*doselevel[i]
  }
  beta00 ~ dnorm(100.0, 0.00001)
  beta01 ~ dnorm(0.0, 0.0001)
  beta10 ~ dnorm(0.0, 0.0001)
  beta11 ~ dnorm(0.0, 0.0001)

  s.y ~dunif(0.0, 250.0)
  s.b0 ~dunif(0.0, 250.0)
  s.b1 ~dunif(0.0, 250.0)

  tau.w <- pow(s.y, -2)
  tau.b0 <- pow(s.b0, -2)
  tau.b1 <- pow(s.b1, -2)
}
```

item (b)

Provide some preliminary evidence that the model looks like it converged. You do not have to do the more advanced statistics (for the purpose of this assignment). It is sufficient to show trace plots and some auto correlation values. (Don't do this for each individual rat's β parameters, it is sufficient to show these values for the parameters of the hierarchical parameters.)

{*Solution.*}

In order to verify the convergence and auto-correlation of our estimates we will plot a sample trace-plots[4] and ACF-plots[5] and the results are as follows:

[2] Another possible approach for this question would be to consider the centered mean of the covariate W (which represents `week`), considering the recomendation of [5] which says when all of the possible values of the covariate are of the same sign and away from zero, the mathematical definition of the intercept may not make sense substantively.

[3] The counterparty of this approach is to deal with a different covariance matrix structure where identifying how the variance is growing at the dose level, for instance, could be troublesome. However, this would bring additional complexities on interpretability of the model which is beyond the scope of this question.

[4] In order to avoid too many points on each graph, affecting the display performance of PDF file, we limited the number of points to $5,000$ which is a good representation of parameter's convergence.

[5] For ACF-Plots we will use the chain-1 to verify for potential auto-correlations of parameter estimates.

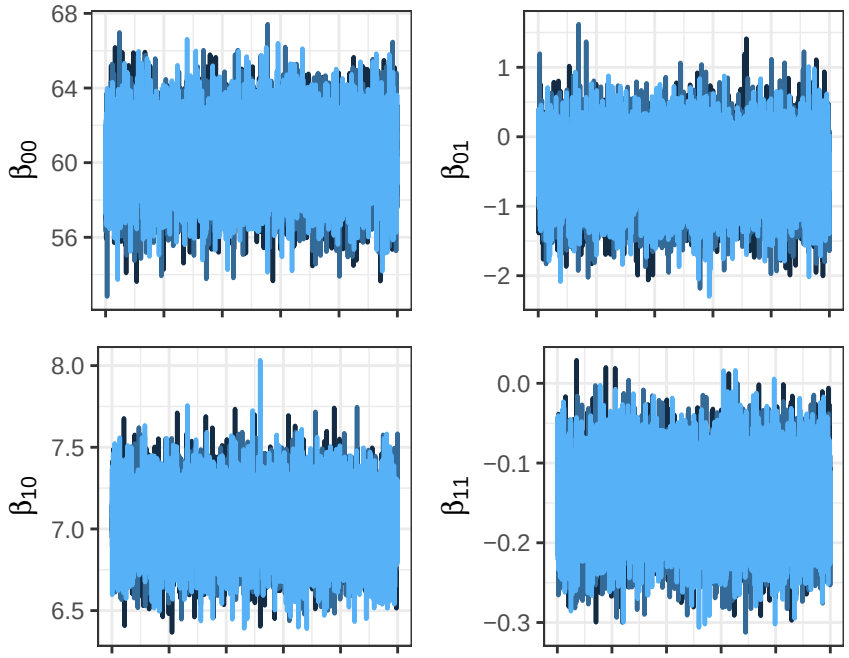

Figure 6: Trace-plots of β_{00}, β_{01}, β_{10} and β_{11} (Run-0)

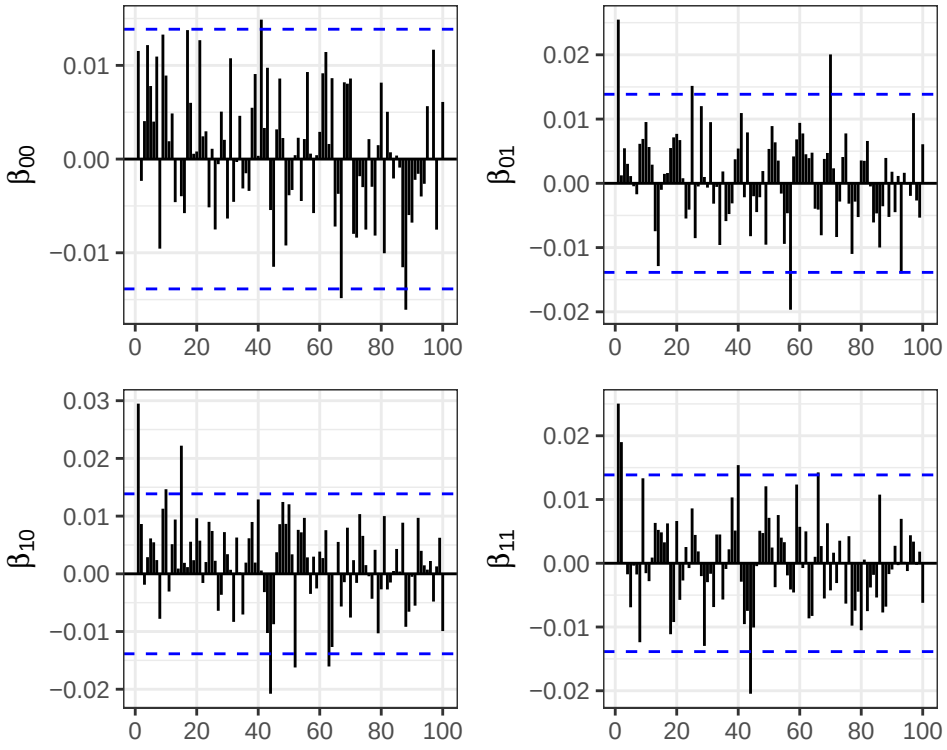

Figure 7: ACF-plots of β_{00}, β_{01}, β_{10} and β_{11} (max-Lag=100) (Run-0)

Comment:- We can see from trace and ACF plots that we have no evidence that our estimates of hierarchical

parameters have not converged or are auto-correlated.

item (c)

Provide the posterior distribution for the parameters of the distribution of the β's That is, the β_{0i}'s, β_{1i}'s and the β_{00}, β_{01}, β_{10}, and β_{11} parameters. (For this subquestion, you don't need to supply the density estimation. You may just provide the usual summary statistics for the distributions.) (This is for the mean and precision parameters which are discussed in part (a) of this question.) Also, provide the posterior distribution for the precision of ϵ_{ij} (which is defined in (a) above.)

{*Solution.*}

The posterior distribution of parameters β_{0i}'s, β_{1i}'s and the β_{00}, β_{01}, β_{10}, and β_{11} are as follows:

Table 2: Summary - Beta0i and Beta1i for Rat Growth (Run-0)

	mean	sd	2.5%	50%	97.5%
beta0[1]	45.4434	1.6504	42.2100	45.4300	48.680
beta0[2]	61.3815	1.6520	58.1400	61.3800	64.640
beta0[3]	65.5680	1.6429	62.3500	65.5700	68.790
beta0[4]	55.9494	1.6460	52.7000	55.9500	59.180
beta0[5]	45.2573	1.6478	41.9900	45.2600	48.490
beta0[6]	55.7574	1.6390	52.5697	55.7500	59.010
beta0[7]	74.3017	1.6406	71.0900	74.3000	77.500
beta0[8]	69.6671	1.6465	66.4300	69.6700	72.890
beta0[9]	64.2272	1.6458	61.0000	64.2200	67.460
beta0[10]	68.6077	1.6461	65.3800	68.6100	71.820
beta0[11]	55.8137	1.6403	52.5700	55.8200	59.010
beta0[12]	59.5719	1.6502	56.3300	59.5800	62.820
beta0[13]	82.4827	1.6622	79.2000	82.4800	85.720
beta0[14]	71.9096	1.6412	68.6900	71.9100	75.130
beta0[15]	54.8833	1.6494	51.6400	54.8800	58.100
beta0[16]	57.8412	1.6390	54.6100	57.8500	61.050
beta0[17]	64.4629	1.6348	61.2600	64.4700	67.650
beta0[18]	61.6838	1.6325	58.4700	61.6900	64.870
beta0[19]	59.2494	1.6397	56.0200	59.2500	62.490
beta0[20]	46.7688	1.6409	43.5700	46.7800	49.990
beta0[21]	49.0842	1.6369	45.8700	49.0900	52.270
beta0[22]	48.9641	1.6416	45.7600	48.9600	52.180
beta0[23]	61.1208	1.6397	57.9000	61.1300	64.340
beta0[24]	50.1858	1.6448	46.9400	50.1900	53.420
beta0[25]	56.1858	1.6325	53.0100	56.1800	59.410
beta0[26]	54.0667	1.6371	50.8200	54.0700	57.270
beta0[27]	69.2868	1.6395	66.0600	69.2900	72.500
beta0[28]	64.5755	1.6428	61.3300	64.5800	67.770
beta0[29]	63.4583	1.6328	60.2700	63.4500	66.680
beta0[30]	59.9138	1.6387	56.7000	59.9200	63.090
beta0[31]	57.8226	1.6419	54.6100	57.8200	61.040
beta0[32]	46.6662	1.6542	43.4100	46.6700	49.900
beta0[33]	57.4665	1.6450	54.2600	57.4700	60.700
beta0[34]	58.2843	1.6384	55.0900	58.2800	61.500

beta0[35]	59.6771	1.6390	56.4800	59.6800	62.900
beta0[36]	76.8083	1.6489	73.5700	76.8000	80.030
beta0[37]	65.2459	1.6455	62.0100	65.2400	68.480
beta0[38]	50.2179	1.6483	46.9700	50.2200	53.450
beta0[39]	59.1892	1.6424	55.9500	59.1900	62.400
beta0[40]	56.8971	1.6439	53.6600	56.9000	60.110
beta0[41]	51.6996	1.6500	48.4800	51.6900	54.940
beta0[42]	54.6003	1.6426	51.3700	54.6000	57.810
beta0[43]	52.8971	1.6476	49.6700	52.8800	56.130
beta0[44]	54.2744	1.6512	51.0400	54.2700	57.520
beta0[45]	49.2279	1.6506	46.0000	49.2200	52.460
beta0[46]	69.6788	1.6551	66.4000	69.6800	72.920
beta0[47]	69.3583	1.6547	66.1200	69.3600	72.600
beta0[48]	38.5631	1.6492	35.3400	38.5700	41.790
beta0[49]	61.6242	1.6590	58.3700	61.6200	64.880
beta0[50]	63.0403	1.6443	59.8100	63.0400	66.250
beta1[1]	6.9304	0.2418	6.4540	6.9290	7.407
beta1[2]	7.5273	0.2418	7.0520	7.5275	8.002
beta1[3]	7.5043	0.2410	7.0330	7.5050	7.976
beta1[4]	6.8012	0.2422	6.3270	6.8010	7.276
beta1[5]	6.0086	0.2410	5.5370	6.0090	6.483
beta1[6]	7.4709	0.2402	6.9950	7.4710	7.941
beta1[7]	7.2487	0.2415	6.7770	7.2490	7.721
beta1[8]	6.8007	0.2413	6.3280	6.7990	7.276
beta1[9]	7.9571	0.2418	7.4840	7.9580	8.430
beta1[10]	7.6609	0.2410	7.1910	7.6610	8.131
beta1[11]	6.2497	0.2403	5.7810	6.2500	6.722
beta1[12]	7.1026	0.2413	6.6290	7.1030	7.574
beta1[13]	8.6777	0.2442	8.1990	8.6770	9.156
beta1[14]	7.9544	0.2412	7.4830	7.9540	8.426
beta1[15]	6.0111	0.2419	5.5380	6.0110	6.489
beta1[16]	6.5443	0.2403	6.0760	6.5440	7.017
beta1[17]	6.5991	0.2407	6.1290	6.5980	7.071
beta1[18]	6.2329	0.2399	5.7630	6.2330	6.703
beta1[19]	7.1267	0.2407	6.6530	7.1270	7.598
beta1[20]	7.1323	0.2404	6.6600	7.1320	7.606
beta1[21]	6.7117	0.2400	6.2460	6.7110	7.182
beta1[22]	6.9602	0.2414	6.4870	6.9610	7.434
beta1[23]	7.0236	0.2401	6.5550	7.0220	7.498
beta1[24]	6.7393	0.2413	6.2660	6.7400	7.213
beta1[25]	5.9282	0.2400	5.4570	5.9290	6.396
beta1[26]	7.1173	0.2400	6.6510	7.1170	7.588
beta1[27]	7.9702	0.2414	7.4980	7.9700	8.446
beta1[28]	6.6860	0.2408	6.2150	6.6860	7.159
beta1[29]	7.1918	0.2397	6.7200	7.1920	7.661
beta1[30]	6.8173	0.2399	6.3480	6.8170	7.290
beta1[31]	6.2426	0.2405	5.7700	6.2440	6.713
beta1[32]	5.4049	0.2427	4.9320	5.4050	5.879
beta1[33]	6.4081	0.2415	5.9360	6.4070	6.883
beta1[34]	6.4373	0.2406	5.9660	6.4370	6.912

beta1[35]	6.7190	0.2406	6.2470	6.7200	7.187
beta1[36]	5.8717	0.2413	5.4000	5.8710	6.345
beta1[37]	7.1473	0.2415	6.6720	7.1470	7.621
beta1[38]	5.1258	0.2420	4.6520	5.1260	5.598
beta1[39]	5.6184	0.2420	5.1460	5.6180	6.094
beta1[40]	6.3377	0.2413	5.8630	6.3370	6.813
beta1[41]	6.0109	0.2422	5.5370	6.0120	6.485
beta1[42]	6.7353	0.2414	6.2610	6.7350	7.208
beta1[43]	6.7943	0.2415	6.3200	6.7950	7.268
beta1[44]	7.2461	0.2426	6.7690	7.2450	7.721
beta1[45]	4.5324	0.2432	4.0580	4.5320	5.008
beta1[46]	7.0363	0.2416	6.5630	7.0360	7.513
beta1[47]	7.7874	0.2435	7.3090	7.7860	8.264
beta1[48]	4.5361	0.2429	4.0580	4.5370	5.010
beta1[49]	3.6770	0.2448	3.1940	3.6780	4.153
beta1[50]	5.7569	0.2409	5.2860	5.7580	6.229

Table 3: Summary - beta00-beta11 for Rat Growth (Run-0)

	mean	sd	2.5%	50%	97.5%
beta00	60.2883	1.7888	56.760	60.2900	63.8100
beta01	-0.4698	0.4415	-1.334	-0.4699	0.3985
beta10	7.0434	0.1777	6.693	7.0430	7.3920
beta11	-0.1487	0.0440	-0.235	-0.1490	-0.0619

Table 4: Summary - Precisions for Rat Growth (Run-0)

	mean	sd	2.5%	50%	97.5%
tau.w	0.1420	0.0095	0.1242	0.1418	0.1612
tau.b0	0.0123	0.0026	0.0077	0.0121	0.0180
tau.b1	1.3160	0.2956	0.8106	1.2910	1.9670

item (d)

Is there any effect due to the drug dose level. Please provide your posterior belief that there is a difference. In justifying your answer, you should include the appropriate posterior distribution.

{*Solution.*}

In order to investigate the effect of each drug levels on rat's weight we first plotted the boxplot, grouped by drug level, per week.

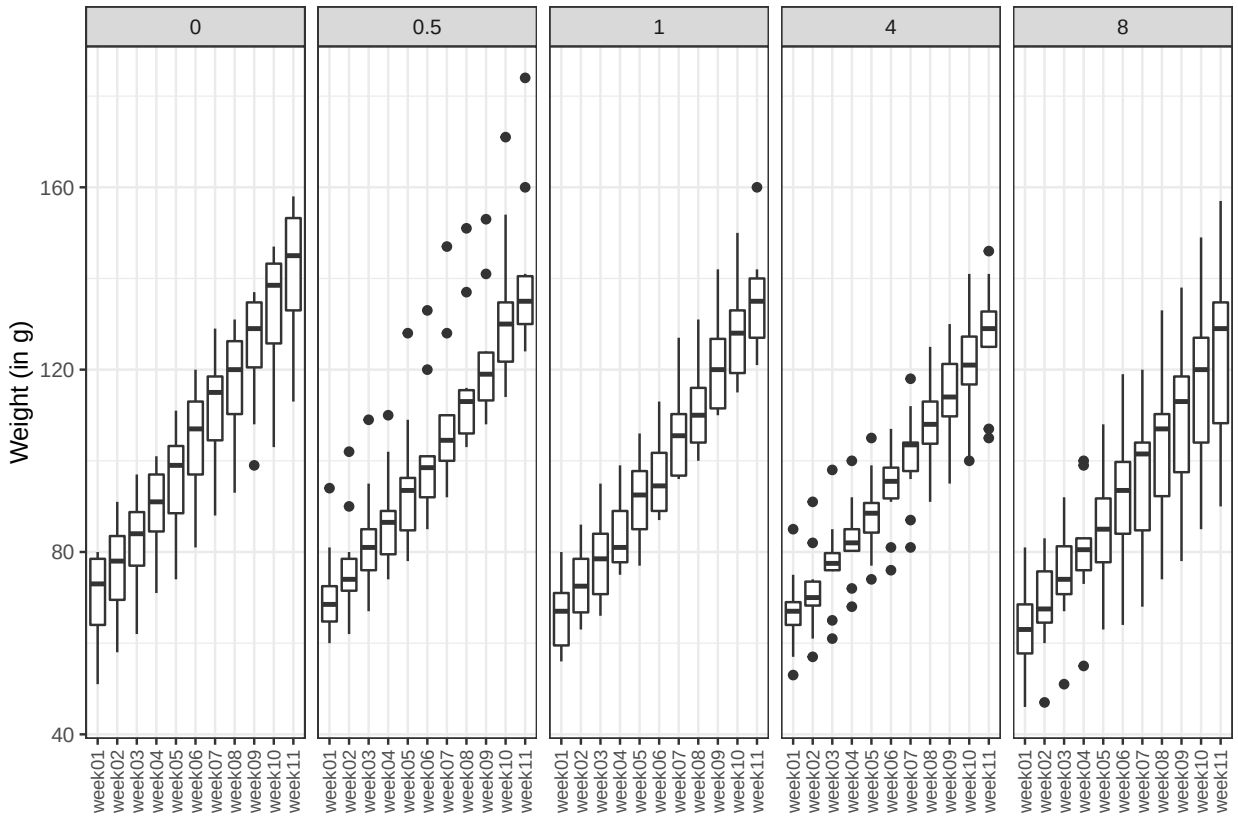

Figure 8: Comparative Weights by dose, by week

Comment:- Comparing the boxplots for each dose level and respective evolution of weight per week we noticed a slight decrease on weight levels at each grid of dose level. We will now verify if its plausible to assume there is a consistent difference between them.

The parameters associated with the effect of drug level are β_{00}, β_{01}, β_{10} and β_{11} and their respective posterior estimates are represented below.

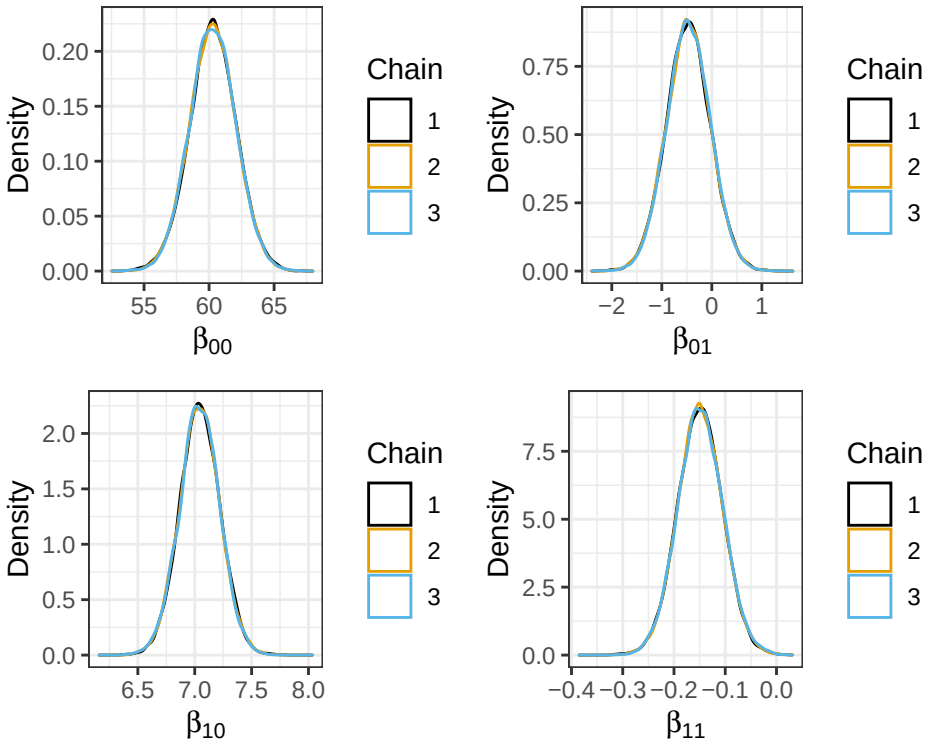

Figure 9: Posterior Distributions β_{00}, β_{01}, β_{10} and β_{11}

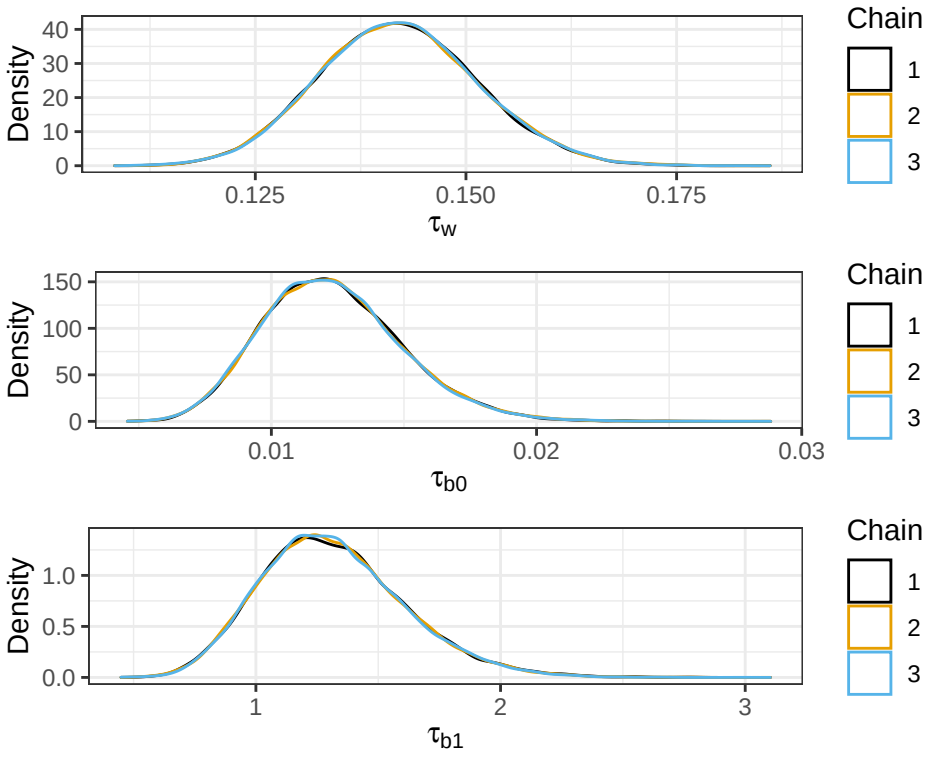

Figure 10: Posterior Distributions τ_w, τ_{β_0} and τ_{β_1}

Comments:- Using the **modeling results** and **posterior estimates of the parameters** we have obtained the following coefficients $\beta_{01} = -0.4698$ and $\beta_{11} = -0.1487$ which represents the expected effect of the dose level over the intercept and slope, respectively, of the predictive populational weight. As both posterior estimates of these parameters are *negative* this might indicate that *the increase of the number of units of the drug have a negative effect over rat's weight over the weeks.*

In other words, we can say that, for each increase in dose level, the expected *weekly growth rate* (β_{1i}) of the rat *decreases* 14.87%. Similar conclusion can be drawn for the intercept ruled by β_{0i}, i.e., the expected starting value for rat's weight is negatively affected and each increase of dose-level makes the *expected startig weight* decrease 46.98% in average.

item (e)

In a short paragraph, summarize your belief in the effect of the drug dose level. In this summary, you should state what the strength of the evidence from the posterior distributions.

{*Solution.*}

Comments:- From the posterior 95% Credible Region for these parameters, we have that $\beta_{01} \in [-1.3370, 0.4101]$ and $\beta_{11} \in [-0.2352, -0.0629]$. This suggests that, depending on the drug level, the effect of β_{11} is negative with 95% of certainty which negatively affects the slope of the populational weight, as the drug level increases. This makes us to believe that there is evidences from the posterior distribution of parameter β_{11} that *the drug indeed has an effect on rat's weight and this effect is negative, i.e., it is expected that their weights will decrease as the number of units of the drug increases.*

References

[1] Robert, C. P., Casella, G., *Introducing Monte Carlo Methods with R*. Chap.6, pp.167 - Springer, 2010.

[2] Gelman, A., Hill, J. *Data Analysis using Regression and Multilevel/Hierarchical Models*, Cambridge Press, 2007.

[3] Marin, J. M., Robert, C. *Bayesian Essentials with R*, 2nd. Ed. pp.49 - Springer, 2014.

[4] Congdon, P. *Applied Bayesian Modelling*, 2nd. Edition, Wiley, 2014

[5] Cowles, M.K. *Applied Bayesian Statistics with R and OpenBugs*, Springer, 2013

Appendix - R-Code

```r
library(tidyverse)
library(R2OpenBUGS)
library(kableExtra)
library(coda)


library(ggplot2)
library(forecast)

# The palette with black - Used in Graphs with :
cbp1 <- c("#999999", "#E69F00", "#56B4E9", "#009E73",
          "#F0E442", "#0072B2", "#D55E00", "#CC79A7")
cbp2 <- c("#000000", "#E69F00", "#56B4E9", "#009E73",
          "#F0E442", "#0072B2", "#D55E00", "#CC79A7")
cbp3 <- c("#FFDB6D", "#C4961A", "#F4EDCA", "#D16103",
          "#C3D7A4", "#52854C", "#4E84C4", "#293352")

# Triangular Distribution
g=function(x){
  (x>0)*(x<1)*((x<=0.5)*4*x+ (x>0.5)*(4-4*x))
}

# Set parameters
N = 1000

set.seed(123)

# Generating two independent samples U1 and U2 from Uniform[0,1]
U1 <- runif(N)
U2 <- runif(N)

# Calculating the exact distribution of Z:= (U1+U2)/2
Z <- (U1+U2)/2

# Estimate E(X) and E(Y) Considerin that Z ~Triang(g)
EZ <- mean(Z)
VarZ <- var(Z)
SE_Z <- sd(Z)/sqrt(N)

# Print Statistics for Approx. Z
cat("\n---- Summary Statistics for approx. Z=(U1+U2)/2 ---------------")
cat("\nE(X) : ",format(EZ, digits = 2, nsmall = 4))
cat("\nS.E.: ",format(SE_Z, digits = 5, nsmall = 4))
CR_Z_up <- EZ+2*SE_Z; CR_Z_lo <- EZ-2*SE_Z;
cat("\n\n95% Credible Region for true-Mean (",
    format(CR_Z_lo, digits = 6, nsmall = 4), ",",
    format(CR_Z_up, digits = 6, nsmall = 4),")\n")
cat("\nVar(X) : ",format(VarZ, digits = 5, nsmall = 4))
cat("\n-----------------------------------------------------------------\n")
```

```r
dfz <- data.frame (ZS = Z)
ggplot(data=dfz, aes(x=ZS))+
  geom_histogram(color="black", fill="white", bins=40)+
  scale_color_manual(values = cbp2)+
  labs(x = "g(.) sample", y = "Histogram of Z")+
  theme_bw()
remove(dfz)

# Item (b) - here h_1(X) = X => h_1(U3)=U3
set.seed(725)
U3 <- runif(N)

# Estimate E(X) via Importance Sampling
EX_IS <- (1/N)*sum((U3*g(U3))/(1))

# here h_2(X) = X^2 => h_1(U3)=U3^2
EX_IS2 <- (1/N)*sum((U3^2*g(U3))/(1))
VarX_IS <- EX_IS2-EX_IS^2
SE_IS <- sd((U3*g(U3))/(1))/sqrt(N)

# Print Statistics for Importance Sampling
cat("\n---- Summary Statistics for Importance Sampling Algorithm -----")
cat("\nE(X) : ",format(EX_IS, digits = 2, nsmall = 4))
cat("\nS.E.: ",format(SE_IS, digits = 5, nsmall = 4))
CR_IS_up <- EX_IS+2*SE_IS; CR_IS_lo <- EX_IS-2*SE_IS;
cat("\n\n95% Credible Region for true-Mean (",
    format(CR_IS_lo, digits = 6, nsmall = 4), ",",
    format(CR_IS_up, digits = 6, nsmall = 4),")\n")
cat("\nVar(X) : ",format(VarX_IS, digits = 5, nsmall = 4))
cat("\n-----------------------------------------------------------------\n")


# Adapted from Robert, Casella - pp.66
xx=g(U3)
mxx <- mean(xx)
estint=cumsum(xx)/(1:N)
esterr=sqrt(cumsum((xx-estint)^2))/(1:N)
dxx <- data.frame(yest = estint,
                  low = estint-2*esterr,
                  up = estint+2*esterr)
ggplot(data=dxx, aes(x=c(1:N)))+
  geom_line(aes(y=yest), size=0.5)+
  geom_line(aes(y=low), color="#E69F00", size=0.5)+
  geom_line(aes(y=up), color="#E69F00", size=0.5)+
  scale_color_manual(values = cbp2)+
  labs(x = "g(.) sample", y = "Density Integration Estimate")+
  ylim(mxx+20*c(-esterr[N],esterr[N]))+
  theme_bw()
remove(xx, mxx, estint, esterr, dxx)

# Initialize Variables
set.seed(923)
```

```r
xrange <- 1 # Only values in range from 0 to 'xrange' are of interest
M <- 2      # Upper Limit

acc <- rej <- 0
y <- rep(0,N)   # vector of sampled data

while (acc <= N) {
  # Propose a 'x' on support of g
  x <- runif(1, min = 0, max = xrange)

  # Generate Accept/Rejection criteria for each fitted value
  u <- runif(1)

  # Maximum of value for distribution 'g'
  if (u <= g(x)/(M)) {
    acc <- acc + 1
    y[acc] <- x
  }
  else rej <- rej + 1
}

# Estimate E(X) and Var(X) using the sample of accepted values
EX_AcpRej <- mean(y)
VarX_AcpRej <- var(y)
SE_AcpRej <- sd(y)/sqrt(N)

# Print Statistics for Accept/Reject
cat("\n---- Summary Statistics for Accept-Reject Algorithm -----")
cat("\nE(X) : ",format(EX_AcpRej, digits = 2, nsmall = 4))
cat("\nS.E.: ",format(SE_AcpRej, digits = 5, nsmall = 4))
CR_AcpRej_up <- EX_AcpRej+2*SE_AcpRej; CR_AcpRej_lo <- EX_AcpRej-2*SE_AcpRej;
cat("\n\n95% Credible Region for true-Mean (",
    format(CR_AcpRej_lo, digits = 6, nsmall = 4), ",",
    format(CR_AcpRej_up, digits = 6, nsmall = 4),")\n")
cat("\nVar(X) : ",format(VarX_AcpRej, digits = 5, nsmall = 4))
cat("\n\nRate of Acceptance = ", format(acc/(acc+rej), digits = 2, nsmall = 4))
cat("\n-----------------------------------------------------------------\n")


dfy <- data.frame (YS = y)
ggplot(data=dfy, aes(x=YS))+
  geom_histogram(color="black", fill="white", bins=40)+
  scale_color_manual(values = cbp2)+
  labs(x = "g(.) sample", y = "Histogram of Y")+
  theme_bw()
remove(dfy)

#Setup Variables
set.seed(936)

alpha = function(x, y){
  min(1, g(x) / g(y))
}
```

```
x = rep(0, N)
acc <- 0

# Loop Sampling from the Chain
for(t in 2:N){
  ystar  <- runif(1, min=0, max=xrange)
  T <- runif(1)
  if( T <= alpha(ystar,x[t-1]))
  {
    x[t] <- ystar
    acc <- acc+1
  }
  else x[t] <- x[t-1]
}

# Print Rate of Acceptance
EX_MH <- mean(x)
Var_MH <- var(x)
SE_MH <- sd(x)/sqrt(N)

# Print Statistics for Metropolis-Hastings
cat("\n---- Summary Statistics for Metropolis-Hastings Algorithm -----")
cat("\nE(X) : ",format(EX_MH, digits = 2, nsmall = 4))
cat("\nS.E.: ",format(SE_MH, digits = 5, nsmall = 4))
CR_MH_up <- EX_MH+2*SE_MH; CR_MH_lo <- EX_MH-2*SE_MH;
cat("\n\n95% Credible Region for true-Mean (",
    format(CR_MH_lo, digits = 6, nsmall = 4), ",",
    format(CR_MH_up, digits = 6, nsmall = 4),")\n")
cat("\nVar(X) : ",format(Var_MH, digits = 5, nsmall = 4))
cat("\n\nRate of Acceptance = ", format(acc/N, digits = 2, nsmall = 4))
cat("\n-----------------------------------------------------------------\n")


dfx <- data.frame (XS = x)
ggplot(data=dfx, aes(x=XS))+
  geom_histogram(color="black", fill="white", bins=40)+
  scale_color_manual(values = cbp2)+
  labs(x = "g(.) sample", y = "Histogram of X")+
  theme_bw()
remove(dfx)

# Prepare Data - Get delta[1-12]
# D3_Work <- as.data.frame(DiabetDrug_M1[["summary"]][4:15,c(1:3,5,7)])

# Generates Studies labels
StudyLst <- rapply(list(c("Z-Estimate", "Imp.Samp.","Accept/Reject","Metr.Hast.")),
                   sprintf, fmt = "%10s", how = "replace")

# Collects C.R's from Summary report
CR_Estim <- data.frame(ID = StudyLst[[1]],
                       lower = c(CR_Z_lo, CR_IS_lo, CR_AcpRej_lo, CR_MH_lo),
                       estim = c(EZ, EX_IS, EX_AcpRej, EX_MH),
                       upper = c(CR_Z_up, CR_IS_up, CR_AcpRej_up, CR_MH_up))
```

```r
# Organize data to generate graphs & analysis
P1 <- CR_Estim %>%
  ggplot() +
  geom_point(aes(x=ID, y=estim, color=ID), size=2.5) +
  geom_errorbar(aes(x=ID, ymin=lower, ymax=upper, color=ID), width = 1) +
  xlab("Algorithm") +
  ylab(expression("Estimate / C.R.")) +
  scale_color_manual(values = cbp2)+
  theme_bw()

P1 + theme(axis.text.x=element_blank())

# Setup Data-set - Read Data
RatGrowth=read.table("Data/RatData11weeks.csv", header=TRUE, sep = ",")

# Setup Variables (ORIGINAL DATABASE)
N <- nrow(RatGrowth)  # Number of Items in data-set
ni <- length(unique(RatGrowth$IDinDose))  # No. of Rats on each dose level
nj <- ncol(RatGrowth)-2      # No. of Weeks of treatment
nk <- length(unique(RatGrowth$dose))      # No. of dose levels

ProdRun <- TRUE  # Control Variable for Testing/Production run

if (ProdRun) {
  # Production Setup OpenBugs running parameters
  NSim <- 30000    # No. of simulations for production
  NChain <- 3      # No. of chains for production
  NThin <- 8      # n.thin parameter for production
  Burnin <- 10000  # Burn-In parameter for production
  Sz <- 5000       # Size of samples for trace/acf plots
} else {
  # Testing Setup OpenBugs running parameters
  NSim <- 5000    # No. of simulations for production
  NChain <- 3      # No. of chains for production
  NThin <- 5      # n.thin parameter for production
  Burnin <- 1000  # Burn-In parameter for production
  Sz <- 1000       # Size of samples for trace/acf plots

}


# Printing the Data-Frame
RatGrowth %>%
  kbl(booktabs = TRUE, digits = 4, longtable = TRUE,
      caption = "Data - Growth in Rats under Treatment") %>%
  kable_styling(latex_options = "striped")


# Model 0 - Standard Model

# Setup variables
y <- as.matrix(RatGrowth[,c(paste0("week",c(1:nj)))])
doselevel <- RatGrowth[,"dose"]
```

```
doselevel.c <- RatGrowth[,"dose"]-mean(RatGrowth[,"dose"])
week <- c(1:nj)
week.c <- c(1:nj)-mean(c(1:nj))

# Setup Model in OpenBugs
cat("
model{
  for (i in 1:N) {
    for (j in 1:nj) {
      y[i,j] ~dnorm(mu[i,j], tau.w)
      mu[i,j] <- beta0[i]+beta1[i]*week[j]
    }
    beta0[i]~dnorm(mu0[i], tau.b0)
    beta1[i]~dnorm(mu1[i], tau.b1)
    mu0[i] <- beta00+beta01*doselevel[i]
    mu1[i] <- beta10+beta11*doselevel[i]
  }
  beta00 ~ dnorm(100.0, 0.00001)
  beta01 ~ dnorm(0.0, 0.0001)
  beta10 ~ dnorm(0.0, 0.0001)
  beta11 ~ dnorm(0.0, 0.0001)

  s.y ~dunif(0.0, 250.0)
  s.b0 ~dunif(0.0, 250.0)
  s.b1 ~dunif(0.0, 250.0)

  tau.w <- pow(s.y, -2)
  tau.b0 <- pow(s.b0, -2)
  tau.b1 <- pow(s.b1, -2)
}", file="RatsGrowthM0.txt")

# Setup Parameters
paramsM0=c("tau.w", "beta0", "beta1",
           "beta00", "beta01", "beta10", "beta11",
           "tau.b0", "tau.b1")

bugM0.dat=list("y", "week", "doselevel", "N", "nj")  # what variable you need in the model


# Setup Initial Values- Stochastic Components
set.seed(963)
initM0.fun=function(){ list(
  beta0 = rnorm(N,50.0,10.0),
  beta1 = rnorm(N,50.0,10.0),
  beta00 = rnorm(1,100.0, 0.00001),
  beta01 = rnorm(1,0.0, 0.0001),
  beta10 = rnorm(1,0.0, 0.0001),
  beta11 = rnorm(1,0.0, 0.0001),
  s.y = runif(1, 0.0, 250.0),
  s.b0 = runif(1, 0.0, 250.0),
  s.b1 = runif(1, 0.0, 250.0)
  ) }
```

```
# Run Open Bugs - Parameters according with 'ProdRun' flag
set.seed(2602)
attach(RatGrowth)
RatGrowthM0=bugs(bugM0.dat, initM0.fun, paramsM0, model.file="RatsGrowthM0.txt",
                 n.chains=NChain, n.iter=NSim, n.burnin=Burnin, n.thin=NThin, debug=FALSE)
detach(RatGrowth)

# Get Simulation from OpenBugs
SArrayM0 <-  RatGrowthM0$sims.array   # Data Arrays
vname <- attr(SArrayM0,"dimnames")[3][[1]]  # Variable Names

# Get Summary statistics of parameters of Interest
RNames <- rownames(RatGrowthM0[["summary"]]) # List of parameters
df_SummaryM0 <- data.frame(Parameter = RNames)
df_SummaryM0 <- cbind(df_SummaryM0, as_tibble(RatGrowthM0[["summary"]]))
rownames(df_SummaryM0) <- RNames


# Plot TracePlots * BETA00, BETA01, BETA10, BETA11*

# Sampling 5000 points to generate "thinner" traceplots
set.seed(312)

L <- NSim-Burnin

S <- if (Sz < L) sort(sample(1:L, Sz, replace = FALSE)) else 1:Sz

# Build working dataframe
D_WorkBeta00 <- data.frame(ValCh1=SArrayM0[S,1,paste0("beta00")],
                           ValCh2=SArrayM0[S,2,paste0("beta00")],
                           ValCh3=SArrayM0[S,3,paste0("beta00")])

D_WorkBeta01 <- data.frame(ValCh1=SArrayM0[S,1,paste0("beta01")],
                           ValCh2=SArrayM0[S,2,paste0("beta01")],
                           ValCh3=SArrayM0[S,3,paste0("beta01")])

D_WorkBeta10 <- data.frame(ValCh1=SArrayM0[S,1,paste0("beta10")],
                           ValCh2=SArrayM0[S,2,paste0("beta10")],
                           ValCh3=SArrayM0[S,3,paste0("beta10")])

D_WorkBeta11 <- data.frame(ValCh1=SArrayM0[S,1,paste0("beta11")],
                           ValCh2=SArrayM0[S,2,paste0("beta11")],
                           ValCh3=SArrayM0[S,3,paste0("beta11")])

# Plot Traceplots for selected data-frames
P1 <- D_WorkBeta00 %>%
  ggplot(aes(seq(from=1,to=Sz)))+
  geom_line(aes(y=ValCh1, colour=1), size=0.8)+
  geom_line(aes(y=ValCh2, colour=2), size=0.8)+
  geom_line(aes(y=ValCh3, colour=3), size=0.8)+
  labs(y = expression(beta['00'])) +
  theme_bw()
P2 <- D_WorkBeta01 %>%
```

```
  ggplot(aes(seq(from=1,to=Sz)))+
  geom_line(aes(y=ValCh1, colour=1), size=0.8)+
  geom_line(aes(y=ValCh2, colour=2), size=0.8)+
  geom_line(aes(y=ValCh3, colour=3), size=0.8)+
  labs(y = expression(beta['01'])) +
  theme_bw()
P3 <- D_WorkBeta10 %>%
  ggplot(aes(seq(from=1,to=Sz)))+
  geom_line(aes(y=ValCh1, colour=1), size=0.8)+
  geom_line(aes(y=ValCh2, colour=2), size=0.8)+
  geom_line(aes(y=ValCh3, colour=3), size=0.8)+
  labs(y = expression(beta['10'])) +
  theme_bw()
P4 <- D_WorkBeta11 %>%
  ggplot(aes(seq(from=1,to=Sz)))+
  geom_line(aes(y=ValCh1, colour=1), size=0.8)+
  geom_line(aes(y=ValCh2, colour=2), size=0.8)+
  geom_line(aes(y=ValCh3, colour=3), size=0.8)+
  labs(y = expression(beta['11'])) +
  theme_bw()

# Plot all Graphs in a same frame
ggpubr::ggarrange(P1+theme(axis.text.x=element_blank(),
                           axis.title.x=element_blank(),
                           legend.position="none"),
                  P2+theme(axis.text.x=element_blank(),
                           axis.title.x=element_blank(),
                           legend.position="none"),
                  P3+theme(axis.text.x=element_blank(),
                           axis.title.x=element_blank(),
                           legend.position="none"),
                  P4+theme(axis.text.x=element_blank(),
                           axis.title.x=element_blank(),
                           legend.position="none"), ncol = 2, nrow = 2)

# Remove Working Variable to free memory
remove(D_WorkBeta00, D_WorkBeta01, D_WorkBeta10, D_WorkBeta11 )

# Plot ACF Plots * BETA00, BETA01, BETA10, BETA11*

# Build working dataframe
D_WorkBeta00 <- data.frame(ValCh1=SArrayM0[,1,paste0("beta00")])
D_WorkBeta01 <- data.frame(ValCh1=SArrayM0[,1,paste0("beta01")])
D_WorkBeta10 <- data.frame(ValCh1=SArrayM0[,1,paste0("beta10")])
D_WorkBeta11 <- data.frame(ValCh1=SArrayM0[,1,paste0("beta11")])

# Plot ACF Plots for selected data-frames
P1 <- ggAcf(D_WorkBeta00$ValCh1, lag.max = 100)+
  labs(x = "Lag", y = expression(beta['00'])) +
  ggtitle(NULL)+
  theme_bw()
P2 <- ggAcf(D_WorkBeta01$ValCh1, lag.max = 100)+
  labs(x = "Lag", y = expression(beta['01'])) +
```

```
  ggtitle(NULL)+
  theme_bw()
P3 <- ggAcf(D_WorkBeta10$ValCh1, lag.max = 100)+
  labs(x = "Lag", y = expression(beta['10'])) +
  ggtitle(NULL)+
  theme_bw()
P4 <- ggAcf(D_WorkBeta11$ValCh1, lag.max = 100)+
  labs(x = "Lag", y = expression(beta['11'])) +
  ggtitle(NULL)+
  theme_bw()

# Plot all Graphs in a same frame
ggpubr::ggarrange(P1+theme(axis.title.x=element_blank(),
                           legend.position="none"),
                  P2+theme(axis.title.x=element_blank(),
                           legend.position="none"),
                  P3+theme(axis.title.x=element_blank(),
                           legend.position="none"),
                  P4+theme(axis.title.x=element_blank(),
                           legend.position="none"), ncol = 2, nrow = 2)
# Remove Working Variable to free memory
remove(D_WorkBeta00, D_WorkBeta01, D_WorkBeta10, D_WorkBeta11 )

# Print statistics of interest - Beta0i and Beta1i
df_Prt <- df_SummaryM0[c(paste0("beta0[",c(1:N),"]"),
                         paste0("beta1[",c(1:N),"]")),c(2:4,6,8)]

df_Prt %>%
  kbl(booktabs = TRUE, digits = 4, longtable = TRUE,
      caption = "Summary - Beta0i and Beta1i for Rat Growth (Run-0)") %>%
  kable_styling(latex_options = "striped")

# Print statistics of interest - beta00-beta11
df_Prt <- df_SummaryM0[c("beta00", "beta01", "beta10", "beta11"),c(2:4,6,8)]

df_Prt %>%
  kbl(booktabs = TRUE, digits = 4,
      caption = "Summary - beta00-beta11 for Rat Growth (Run-0)") %>%
  kable_styling(latex_options = "striped")

# Print statistics of interest - Precisions
df_Prt <- df_SummaryM0[c("tau.w", "tau.b0", "tau.b1"),c(2:4,6,8)]

df_Prt %>%
  kbl(booktabs = TRUE, digits = 4,
      caption = "Summary - Precisions for Rat Growth (Run-0)") %>%
  kable_styling(latex_options = "striped")

# Create Temporary data-frame
D1_Work <- RatGrowth
colnames(D1_Work) <- c("dose", "IDinDose", "week01","week02","week03","week04","week05",
                       "week06","week07","week08","week09","week10","week11")
```

```
P2 <- D1_Work %>%
  pivot_longer(cols = week01:week11, names_to = "weekStr", values_to = "rweight") %>%
  mutate(weekNo = as.integer(substr(weekStr,5,length(weekStr)-5))) %>%
  ggplot(aes(group=weekNo))+
  geom_boxplot(aes(x=weekStr, y=rweight)) +
  labs(y="Weight (in g)")+
  facet_grid(~dose)+
  theme_bw()
P2 + theme(axis.title.x=element_blank(),
           axis.text.x = element_text(angle = 90, hjust = 1, vjust = 0.5, size = 8))
remove(D1_Work)


# Prepare Data-Frame for next steps
D1_Work <- rbind(data.frame(beta00=SArrayM0[,1,paste0("beta00")],
                            beta01=SArrayM0[,1,paste0("beta01")],
                            beta10=SArrayM0[,1,paste0("beta10")],
                            beta11=SArrayM0[,1,paste0("beta11")], Chain=factor(1)),
                 data.frame(beta00=SArrayM0[,2,paste0("beta00")],
                            beta01=SArrayM0[,2,paste0("beta01")],
                            beta10=SArrayM0[,2,paste0("beta10")],
                            beta11=SArrayM0[,2,paste0("beta11")], Chain=factor(2)),
                 data.frame(beta00=SArrayM0[,3,paste0("beta00")],
                            beta01=SArrayM0[,3,paste0("beta01")],
                            beta10=SArrayM0[,3,paste0("beta10")],
                            beta11=SArrayM0[,3,paste0("beta11")], Chain=factor(3)))

# Plot the Graphs of beta00, beta01, beta10, beta11
P1 <- D1_Work %>%
  ggplot(mapping = aes(x = beta00, group = Chain))+
  geom_density(aes(colour=Chain), size=0.5)+
  labs(x = expression(beta['00']), y = "Density") +
  scale_color_manual(values = cbp2)+
  theme_bw()

P2 <- D1_Work %>%
  ggplot(mapping = aes(x = beta01, group = Chain))+
  geom_density(aes(colour=Chain), size=0.5)+
  labs(x = expression(beta['01']), y = "Density") +
  scale_color_manual(values = cbp2)+
  theme_bw()

P3 <- D1_Work %>%
  ggplot(mapping = aes(x = beta10, group = Chain))+
  geom_density(aes(colour=Chain), size=0.5)+
  labs(x = expression(beta['10']), y = "Density") +
  scale_color_manual(values = cbp2)+
  theme_bw()

# Plot the Graphs of Tau
P4 <-D1_Work %>%
  ggplot(mapping = aes(x = beta11, group = Chain))+
  geom_density(aes(colour=Chain), size=0.5)+
```

```
  labs(x = expression(beta['11']), y = "Density") +
  scale_color_manual(values = cbp2)+
  theme_bw()

ggpubr::ggarrange(P1, P2, P3, P4, ncol = 2, nrow = 2)

# Remove Working Variable to free memory
remove(D1_Work)

# Prepare Data-Frame for next steps
D2_Work <- rbind(data.frame(tau.w=SArrayM0[,1,paste0("tau.w")],
                            tau.b0=SArrayM0[,1,paste0("tau.b0")],
                            tau.b1=SArrayM0[,1,paste0("tau.b1")], Chain=factor(1)),
                 data.frame(tau.w=SArrayM0[,2,paste0("tau.w")],
                            tau.b0=SArrayM0[,2,paste0("tau.b0")],
                            tau.b1=SArrayM0[,2,paste0("tau.b1")], Chain=factor(2)),
                 data.frame(tau.w=SArrayM0[,3,paste0("tau.w")],
                            tau.b0=SArrayM0[,3,paste0("tau.b0")],
                            tau.b1=SArrayM0[,3,paste0("tau.b1")], Chain=factor(3)))


# Plot the Graphs of TAU_W, TAU_Beta0, TAU_Beta1
P1 <- D2_Work %>%
  ggplot(mapping = aes(x = tau.w, group = Chain))+
  geom_density(aes(colour=Chain), size=0.5)+
  labs(x = expression(tau['w']), y = "Density") +
  scale_color_manual(values = cbp2)+
  theme_bw()

P2 <- D2_Work %>%
  ggplot(mapping = aes(x = tau.b0, group = Chain))+
  geom_density(aes(colour=Chain), size=0.5)+
  labs(x = expression(tau['b0']), y = "Density") +
  scale_color_manual(values = cbp2)+
  theme_bw()

# Plot the Graphs of Tau
P3 <-D2_Work %>%
  ggplot(mapping = aes(x = tau.b1, group = Chain))+
  geom_density(aes(colour=Chain), size=0.5)+
  labs(x = expression(tau['b1']), y = "Density") +
  scale_color_manual(values = cbp2)+
  theme_bw()

ggpubr::ggarrange(P1, P2, P3, ncol = 1, nrow = 3)

# Remove Working Variable to free memory
remove(D2_Work)
```