Worksheet 05 Group 2

Andrea Staub Emanuel Mauch Holly Vuarnoz Jan Hohenheim
Sophie Haldemann

```
library(tidyverse)
library(rjags)
library(bayesmeta)
```

Exercise 3

Data:

```
dat <-
  tibble(
    pl_total = c(107,44,51,39,139,20,78,35),
    pl_case = c(23,12,19,9,39,6,9,10),
    tr_total = c(208,38,150,45,138,20,201,34),
    tr_case = c(120,18,107,26,82,16,126,23),
  ) |>
  mutate(
    n_pl_case = pl_total - pl_case,
    n_tr_case = tr_total - tr_case,
    labels = c("Adalimumab 1", "Adalimumab 2", "Etanercept 1", "Etanercept 2", "Etanercept 3", "Etanerc
  ) |>
  mutate(
    or = (pl_case / n_pl_case) / (tr_case / n_tr_case),
  ) |>
  mutate(
    log_or = log(or),
    se_log_or = sqrt(1 / pl_case + 1 / n_pl_case + 1 / tr_case + 1 / n_tr_case),
  )
```

Analysis:

```
res1 <- bayesmeta(
  y = dat$log_or,
  sigma = dat$se_log_or,
  labels = dat$labels,
  mu.prior.mean = 0,
  mu.prior.sd = 4,
  tau.prior = \(t) dhalfnormal(t, scale = 0.5),
  interval.type = "central"
)
```

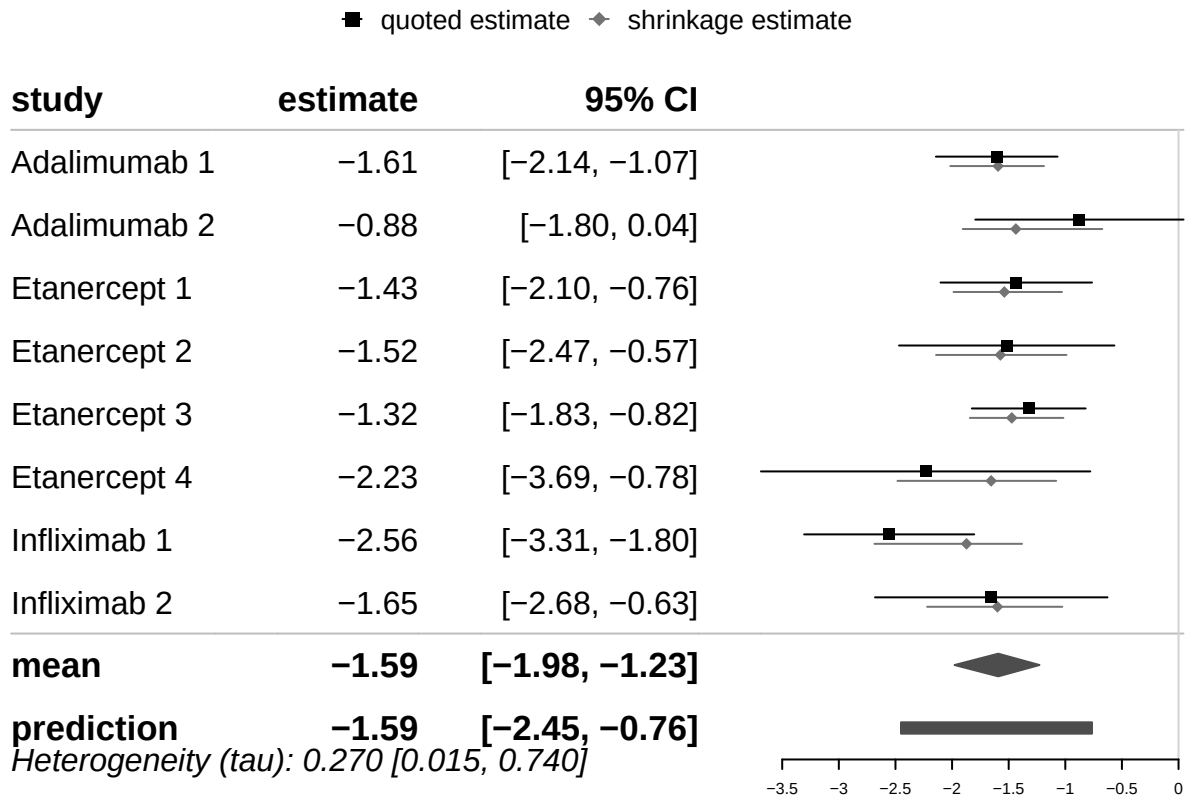
Plots:

```
res1 |> summary()
```

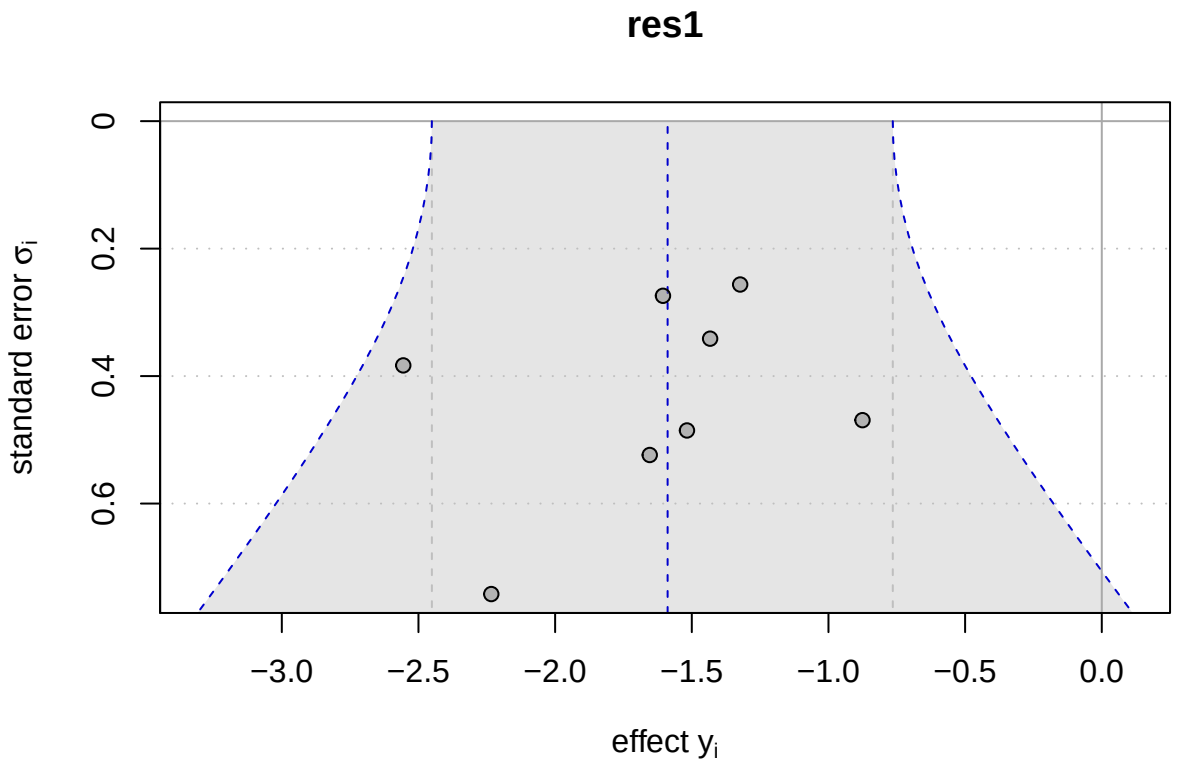
```
##  'bayesmeta' object.
```

```
## data (8 estimates):
##                         y     sigma
## Adalimumab 1 -1.6054775 0.2740073
## Adalimumab 2 -0.8754687 0.4691896
## Etanercept 1 -1.4329256 0.3412963
## Etanercept 2 -1.5176304 0.4853221
## Etanercept 3 -1.3229761 0.2563070
## Etanercept 4 -2.2335922 0.7420210
## Infliximab 1 -2.5556757 0.3832411
## Infliximab 2 -1.6538897 0.5238200
##
## tau prior (proper):
## \(t) dhalfnormal(t, scale = 0.5)
## <bytecode: 0x7fb5c3545a88>
##
## mu prior (proper):
## normal(mean=0, sd=4)
##
## ML and MAP estimates:
##                      tau        mu
## ML joint     0.2094171 -1.592280
## ML marginal  0.2852879 -1.590235
## MAP joint    0.1614761 -1.585174
## MAP marginal 0.2334117 -1.587618
##
## marginal posterior summary:
##                  tau         mu      theta
## mode       0.2334117 -1.5876182 -1.5805059
## median     0.2702386 -1.5919563 -1.5884808
## mean       0.2949284 -1.5946544 -1.5946544
## sd         0.1941244  0.1879906  0.4002409
## 95% lower 0.0153332 -1.9777397 -2.4509871
## 95% upper 0.7397310 -1.2281569 -0.7646060
##
## (quoted intervals are central, equal-tailed credible intervals.)
##
## Bayes factors:
##              tau=0        mu=0
## actual  1.0209152 3.11865e-05
## minimum 0.7030068 1.23805e-06
##
## relative heterogeneity I^2 (posterior median): 0.3343206
```

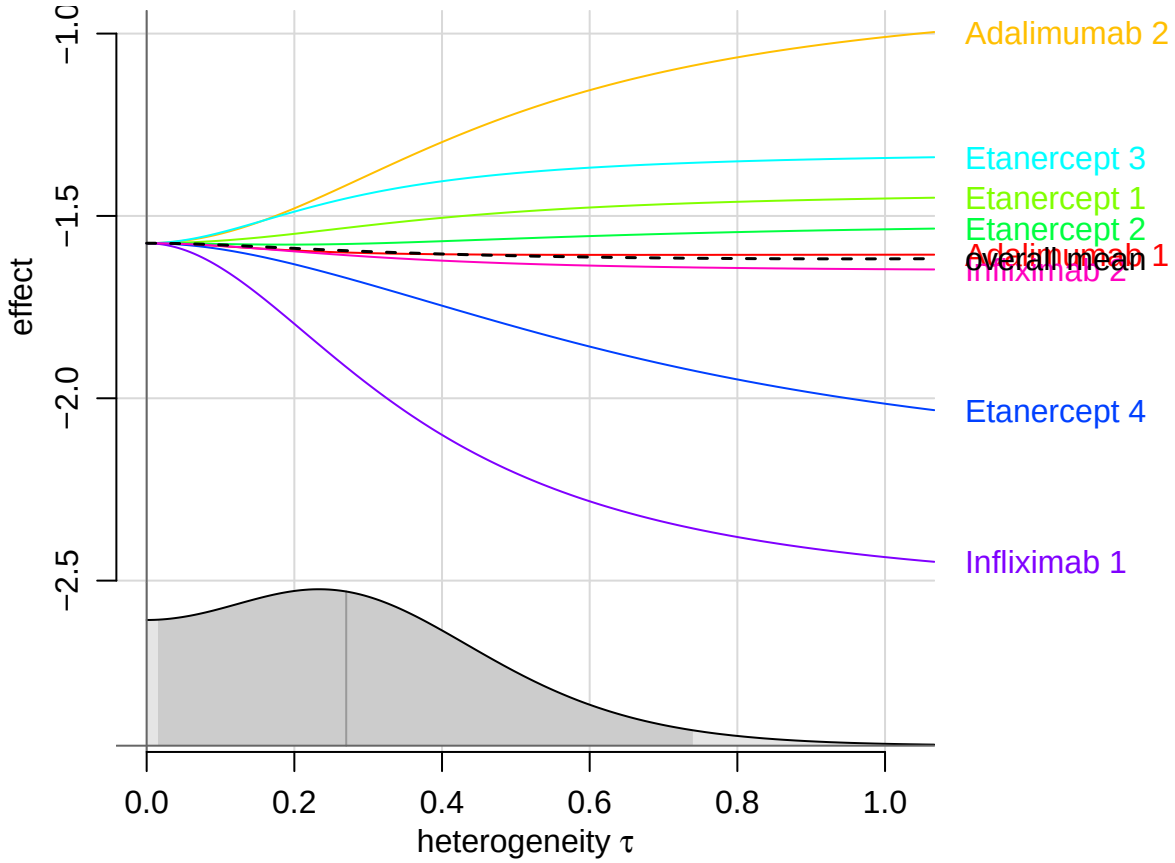
```
res1 |> forestplot()
```




```
res1 |> funnel()
```




```
res1 |> bayesmeta::traceplot()
```




```
res1 |> weights()
```

```
## Adalimumab 1 Adalimumab 2 Etanercept 1 Etanercept 2 Etanercept 3 Etanercept 4
##  0.192387131  0.097292478  0.147914947  0.092796783  0.207606031  0.048902618
## Infliximab 1 Infliximab 2   prior mean
##  0.127676598  0.083225820  0.002197593
```

The 95% CI for the posterior is [-2.45, -0.76]. This is fairly expected, considering most of the studies report pretty similar findings.

Exercise 4

(Bayesian meta-analysis with JAGS)

Run R code provided in the file 06worksheet_JAGSextension.R. This model provides an alternative analysis of data considered in the Exercise 3 above. Discuss similarities and differences of the model provided in this R code and models that were used for Bayesian meta-analyses in the Exercise 3 above and in the individual project (Exercise 1 of Worksheet 5).

```
pl1.data<-list(N = 16,
               y = c(23., 12., 19.,  9.,  39.,  6.,  9., 10., 120., 18., 107.,
                     26., 82., 16., 126., 23.),
               n = c(107., 44., 51., 39., 139., 20., 78., 35., 208., 38., 150.,
                     45., 138., 20., 201., 34.),
               C1 = c(0., 0., 0., 0., 0., 0., 0., 0., 1., 1., 1., 1., 1., 1., 1., 1.))


pl1.params<-c("mu", "beta", "tau", "p1.star", "p2.star")


pl1_modelString <- "
model
{

#   sampling model (likelihood)
for (j in 1:N)  {
y[j] ~ dbin(p[j],n[j])
logit(p[j]) <- mu + beta*C1[j] + eta[j]
eta[j] ~ dnorm(0, tau.prec)

#   prediction for posterior predictive checks
y.pred[j] ~ dbin(p[j],n[j])
PPC[j] <- step(y[j]-y.pred[j])-0.5*equals(y[j],y.pred[j])
}

#   priors
mu ~ dunif(-10,10)
beta ~ dunif(-10,10)
tau ~ dunif(0,10)
tau.prec <- 1/tau/tau

#   population effect
p1 <- 1/(1+exp(-mu))
p2 <- 1/(1+exp(-mu-beta))

#   predictive distribution for new study effect
eta.star ~ dnorm(0,tau.prec)
p1.star <- 1/(1+exp(-mu-eta.star))
p2.star <- 1/(1+exp(-mu-beta-eta.star))

}
"
```

```r
writeLines(pl1_modelString, con="TempModel.txt") # write to a file

# model initiation
rjags.pl1 <- jags.model(
  file = "TempModel.txt",
  data = pl1.data,
  n.chains = 4,
  n.adapt = 4000
)
```

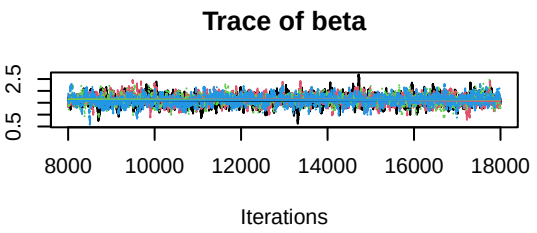
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 16
##    Unobserved stochastic nodes: 36
##    Total graph size: 222
##
## Initializing model
```

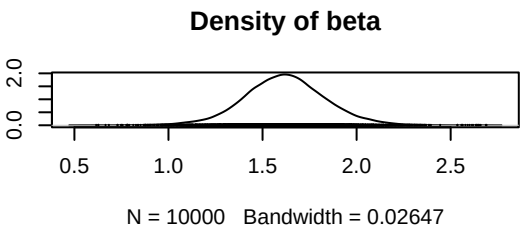
```r
# str(rjags.pl1)
# class(rjags.pl1)
# attributes(rjags.pl1)

# burn-in

update(rjags.pl1, n.iter = 4000)

# sampling/monitoring
fit.rjags.pl1.coda <- coda.samples(
  model = rjags.pl1,
  variable.names = pl1.params,
  n.iter = 10000,
  thin = 1
)

summary(fit.rjags.pl1.coda)
```

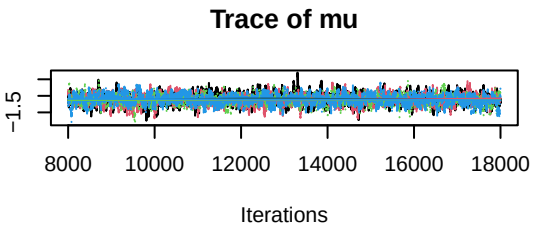
```
##
## Iterations = 8001:18000
## Thinning interval = 1
## Number of chains = 4
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##            Mean      SD  Naive SE Time-series SE
## beta     1.6191 0.21964 0.0010982      0.0058688
## mu      -1.1137 0.15977 0.0007989      0.0038363
## p1.star  0.2529 0.06747 0.0003373      0.0007978
## p2.star  0.6202 0.08071 0.0004035      0.0006783
## tau      0.2980 0.13614 0.0006807      0.0032845
##
```

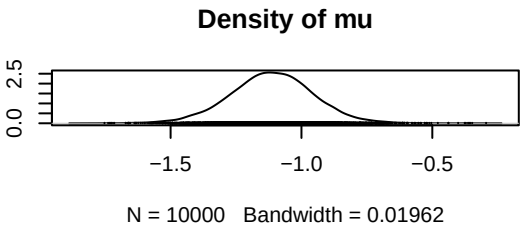
```
## 2. Quantiles for each variable:
##
##              2.5%     25%     50%     75%   97.5%
## beta      1.18051  1.4781  1.6166  1.7567  2.0643
## mu       -1.43342 -1.2162 -1.1124 -1.0097 -0.8011
## p1.star   0.13336  0.2111  0.2474  0.2871  0.4096
## p2.star   0.44327  0.5767  0.6232  0.6683  0.7763
## tau       0.06401  0.2073  0.2851  0.3746  0.6010
```

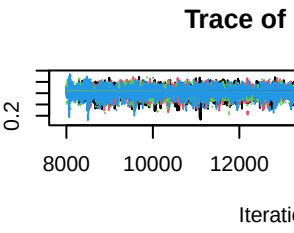
```
plot(fit.rjags.pl1.coda)
```

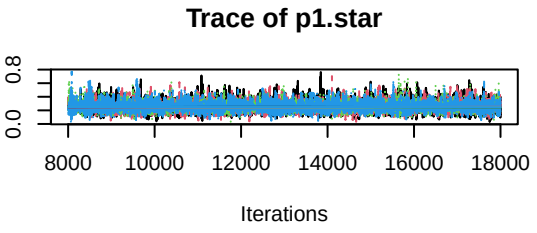


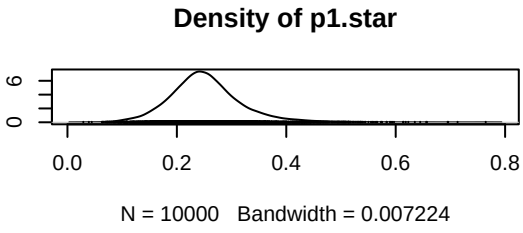


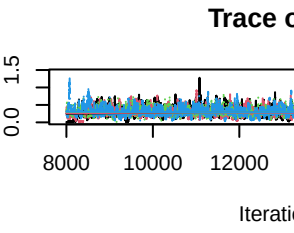












Comparison of models

	W06 Ex. 4	**W06 Ex. 3**	**W05 Ex.1**
Framework	Bayesian	Bayesian	Bayesian
Purpose	Random-effects meta-analysis	Random-effects meta-analysis	Random-effects meta-analysis
Data	accommodates treatment and placebo data	accommodates treatment and placebo data	does not accommodate treatment data, only placebo
Transformation of data	models number of responders as binomially distributed with study-specific p_j and n_j	models log-odds as normally distributed	models log-odds as normally distributed centered around study-specific θ_j with study-specific precision prec_s
Priors	uniform priors for μ, β, and τ	- normal prior for μ - halfnormal prior for τ	- normal prior for μ - gamma prior for prec_tau
Random effects	study-specific p_j are transformed into log-odds and modeled as a linear regression $\mu + \beta * c_j + \eta_j$, study-specific error terms η_j are modelled as normally distributed around 0 with precision τ.prec	- study-specific effects are normally distributed around μ	random effects θ_j are normally distributed, centered around the global effect μ with precision prec_ τ
PPC	included	not included	not included
Output	- predictive distributions of placebo response rate p1.star and treatment response rate p2.star; which are derived by back-transforming the log-odds onto a probability scale - posterior distributions of μ, β and τ	- posterior distributions of τ, μ and θ	predictive distribution of placebo response rate, which is derived by back-transforming the log-odds $\theta \sim N(\mu, prec\_\tau)$ onto a probability scale
Normal parameterization	mean, precision	mean, standard deviation	mean, precision
Approach	MCMC	Numerical	MCMC
Ease of use	requires more coding	easier to apply	requires more coding

Figure 1: Exercise 4.

Exercise 5

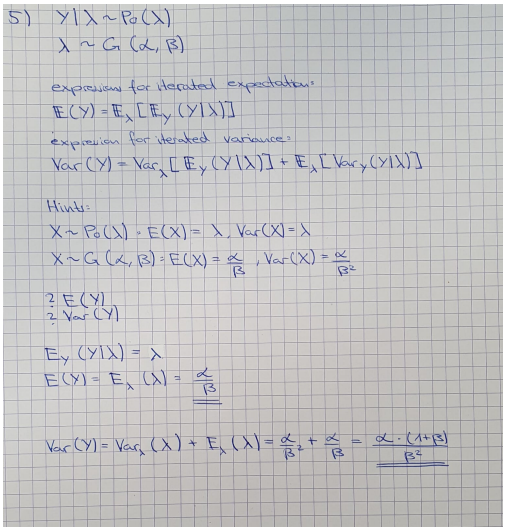


Figure 2: Exercise 5.

Exercise 6

a) Numerical approximation

Numerical maximisation of the log-likelihood corresponding to the Poisson-gamma distribution as described by (Held and Sabanés Bové, 2020, p. 210–211 and Fig. 6.13) to obtain the marginal maximum likelihood estimator.

```
# Data: observed counts of lip cancer cases
y <- c(11, 5, 15, 9, 6, 9, 2, 3, 26, 39, 20, 31, 9, 16, 6, 16, 19, 17, 15, 11, 19,
       7, 10, 0, 7, 7, 9, 2, 8, 8, 11, 6, 28, 4, 1, 1, 1, 8, 6, 3, 2, 1, 7, 10, 9, 11, 3,
       11, 5, 8, 3, 7, 0, 8, 7, 13)
# Sanity check: number of districts
n <- length(y)
```

The log-likelihood function for the Poisson-Gamma model is given by (from the Book):

$$l(\alpha, \beta) = \sum_{i=1}^{n} \left[\alpha \log(\beta) + \log\left(\frac{\Gamma(\alpha + x_i)}{\Gamma(\alpha)} \right) - (\alpha + x_i) \log(\beta + e_i) \right]$$

We want to maximize this log-likelihood function.

```
# note that e_i = 1 and xi is our data y
# Log-likelihood function
log.lik <- function(x) {
  alpha <- x[1]
  beta <- x[2]
  log.lik <- sum(alpha * log(beta) + log( (gamma(alpha + y))/(gamma(alpha)) ) -
              (alpha + y) * log(beta + 1))
  return(log.lik)
  }
```

```r
# Maximize log-likelihood function
opt <- optim(par=c(0.1, 0.1), fn=log.lik, control = list(fnscale = -1))
```

```
## Warning in log(beta): NaNs produced
## Warning in log(beta): NaNs produced
## Warning in log(beta): NaNs produced
```

```r
# Print the maximum likelihood estimates
opt$par
```

```
## [1] 1.8321593 0.1914292
```

```r
alpha1 <- opt$par[1]
beta1 <- opt$par[2]
```

b) Moments matching

Matching of moments based on the Exercise 5 above, which provides the marginal moment estimator.

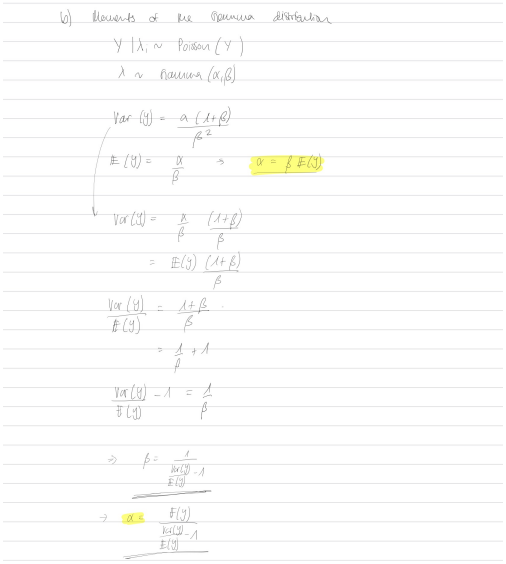


Figure 3: Exercise 6.

```r
# Moment-matching function
moment.matching <- function(mean, var) {
  alpha <- mean / (var/mean - 1)
  beta <- 1 / (var/mean - 1)
  return(params = c(alpha=alpha, beta=beta))
  }

params <- moment.matching(mean = mean(y), var = var(y))
alpha2 <- params[1]
beta2 <- params[2]
```

Compare means and the lengths of equi-tailed 95%CrI obtained by both approaches. Report your results.

```r
# Function to compute 95% credible intervals
credible_interval <- function(alpha, beta, y) {
  lower <- qgamma(0.025, alpha + y, rate = beta + 1)
  upper <- qgamma(0.975, alpha + y, rate = beta + 1)
  return(c(lower, upper))
}

# Compute credible intervals using MLE
CrI1 <- t(sapply(y, credible_interval, alpha = alpha1, beta = beta1))
CrI1
```

```
##             [,1]      [,2]
##  [1,]  5.7075164 17.414051
##  [2,]  2.2741767 10.766990
##  [3,]  8.2047971 21.633617
##  [4,]  4.5100265 15.252482
##  [5,]  2.8074244 11.914749
##  [6,]  4.5100265 15.252482
##  [7,]  0.8441860  7.145953
##  [8,]  1.2845305  8.391500
##  [9,] 15.5016904 32.804718
## [10,] 24.5757174 45.554557
## [11,] 11.4607635 26.772546
## [12,] 18.9491521 37.751177
## [13,]  4.5100265 15.252482
## [14,]  8.8457674 22.671700
## [15,]  2.8074244 11.914749
## [16,]  8.8457674 22.671700
## [17,] 10.8001162 25.754277
## [18,]  9.4922591 23.704218
## [19,]  8.2047971 21.633617
## [20,]  5.7075164 17.414051
## [21,] 10.8001162 25.754277
## [22,]  3.3598004 13.042827
## [23,]  5.1036432 16.338470
## [24,]  0.1613126  4.429515
## [25,]  3.3598004 13.042827
## [26,]  3.3598004 13.042827
## [27,]  4.5100265 15.252482
## [28,]  0.8441860  7.145953
## [29,]  3.9281299 14.154573
## [30,]  3.9281299 14.154573
## [31,]  5.7075164 17.414051
## [32,]  2.8074244 11.914749
## [33,] 16.8731620 34.790841
## [34,]  1.7644360  9.594871
## [35,]  0.4596559  5.838925
## [36,]  0.4596559  5.838925
## [37,]  0.4596559  5.838925
## [38,]  3.9281299 14.154573
## [39,]  2.8074244 11.914749
## [40,]  1.2845305  8.391500
## [41,]  0.8441860  7.145953
## [42,]  0.4596559  5.838925
```

```
## [43,]  3.3598004 13.042827
## [44,]  5.1036432 16.338470
## [45,]  4.5100265 15.252482
## [46,]  5.7075164 17.414051
## [47,]  1.2845305  8.391500
## [48,]  5.7075164 17.414051
## [49,]  2.2741767 10.766990
## [50,]  3.9281299 14.154573
## [51,]  1.2845305  8.391500
## [52,]  3.3598004 13.042827
## [53,]  0.1613126  4.429515
## [54,]  3.9281299 14.154573
## [55,]  3.3598004 13.042827
## [56,]  6.9415114 19.538633
```

```
# Compute credible intervals using Method of Moments
CrI2 <- t(sapply(y, credible_interval, alpha = alpha2, beta = beta2))
CrI2
```

```
##             [,1]      [,2]
##  [1,]  5.6964194 17.461385
##  [2,]  2.2408836 10.744702
##  [3,]  8.2133283 21.722397
##  [4,]  4.4901897 15.277997
##  [5,]  2.7767726 11.905168
##  [6,]  4.4901897 15.277997
##  [7,]  0.8090843  7.078868
##  [8,]  1.2485628  8.340993
##  [9,] 15.5718617 32.999720
## [10,] 24.7255127 45.868275
## [11,] 11.4963185 26.910612
## [12,] 19.0493616 37.992428
## [13,]  4.4901897 15.277997
## [14,]  8.8595245 22.770518
## [15,]  2.7767726 11.905168
## [16,]  8.8595245 22.770518
## [17,] 10.8301051 25.882649
## [18,]  9.5113410 23.812974
## [19,]  8.2133283 21.722397
## [20,]  5.6964194 17.461385
## [21,] 10.8301051 25.882649
## [22,]  3.3323517 13.045362
## [23,]  5.0880612 16.375013
## [24,]  0.1388657  4.314686
## [25,]  3.3323517 13.045362
## [26,]  3.3323517 13.045362
## [27,]  4.4901897 15.277997
## [28,]  0.8090843  7.078868
## [29,]  3.9043162 14.168775
## [30,]  3.9043162 14.168775
## [31,]  5.6964194 17.461385
## [32,]  2.7767726 11.905168
## [33,] 16.9552412 35.004450
## [34,]  1.7292628  9.559065
## [35,]  0.4282177  5.751860
```

```
## [36,]  0.4282177  5.751860
## [37,]  0.4282177  5.751860
## [38,]  3.9043162 14.168775
## [39,]  2.7767726 11.905168
## [40,]  1.2485628  8.340993
## [41,]  0.8090843  7.078868
## [42,]  0.4282177  5.751860
## [43,]  3.3323517 13.045362
## [44,]  5.0880612 16.375013
## [45,]  4.4901897 15.277997
## [46,]  5.6964194 17.461385
## [47,]  1.2485628  8.340993
## [48,]  5.6964194 17.461385
## [49,]  2.2408836 10.744702
## [50,]  3.9043162 14.168775
## [51,]  1.2485628  8.340993
## [52,]  3.3323517 13.045362
## [53,]  0.1388657  4.314686
## [54,]  3.9043162 14.168775
## [55,]  3.3323517 13.045362
## [56,]  6.9399388 19.606984
```

```r
# Length of the credible intervals
lengths_1 <- CrI1[, 2] - CrI1[, 1]
lengths_2 <- CrI2[, 2] - CrI2[, 1]

# Compute posterior means
lambda_eb_1 <- (alpha1 + y) / (beta1 + 1)
lambda_eb_2 <- (alpha2 + y) / (beta2 + 1)

# Compare means and lengths of 95% credible intervals
mean_comparison <- data.frame(
  Statistic = c("Posterior Mean (MLE)", "Posterior Mean (MM)", "Length of 95% CrI (MLE)", "Length of 95
  Mean = c(mean(lambda_eb_1), mean(lambda_eb_2), mean(lengths_1), mean(lengths_2))
)

print(mean_comparison)
```

```
##                 Statistic      Mean
## 1    Posterior Mean (MLE)  9.571351
## 2     Posterior Mean (MM)  9.571429
## 3 Length of 95% CrI (MLE) 10.430665
## 4  Length of 95% CrI (MM) 10.464384
```

Practically the same.