Lab 4

Name: Huiyu Hu

```
#CHANGE WORKING DIRECTORY
setwd("/Users/huiyuhu/Desktop/Study/UCLA_Biostat/BIOSTAT234/lab/Lab 4")
getwd()
```

```
## [1] "/Users/huiyuhu/Desktop/Study/UCLA_Biostat/BIOSTAT234/lab/Lab 4"
```

```
#LOAD NECESSARY PACKAGES
library(R2jags)
```

```
## Loading required package: rjags
```

```
## Loading required package: coda
```

```
## Linked to JAGS 4.3.0
```

```
## Loaded modules: basemod,bugs
```

```
##
## Attaching package: 'R2jags'
```

```
## The following object is masked from 'package:coda':
##
##     traceplot
```

```
load("AddBurnin.RData")
library(lattice)
library(knitr)
# useful function
mysummary = function(invector) {
c(mean(invector), sd(invector), quantile(invector, .025),
    quantile(invector,.975),
    length(invector[invector>0])/length(invector))
}

# load the data.

load("lab4_data.RData")
```

```
#Create the model

# sink("lab4model.txt")
# cat("
# model
#        {
#                for( i in 1 : 64 ) {
#                        for( j in 1 : 4 ) {
#                            s[i, j]<-4*(i-1)+j
#                            y[i, j] ~ dnorm(mu[i , j],tau.e)
#                                mu[i , j] <- inprod(x[s[i,j],],alpha[])+beta[i]
#                        }
#                        beta[i]~dnorm(0, tau.b)
#                }
#
# for( k in 1:8) {
#                alpha[k]~dnorm(m[k],varinv[k])
#                alphasign[k] <- step(alpha[k])
# }
#
#                tau.e ~ dgamma(ea,eb)
#                tau.b~dgamma(ba,bb)
#
#                sigma <- 1 /sqrt( tau.e)
#                sqrtD <- 1 /sqrt( tau.b)
#                rho <- sqrtD*sqrtD/(sigma*sigma + sqrtD *sqrtD)
#
#        }
#
#     ",fill = TRUE)
# sink()
```

```
run1 = jags(priordata, inits, parameters, "lab4model.txt",
    n.chains=5, n.iter=1100, n.burnin=0, n.thin=1)
```

```
## module glm loaded
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 245
##    Unobserved stochastic nodes: 85
##    Total graph size: 3256
##
## Initializing model
```

```
names(run1)
```

```
## [1] "model"              "BUGSoutput"          "parameters.to.save"
## [4] "model.file"         "n.iter"              "DIC"
```

```
Output1=AddBurnin(run1$BUGSoutput$sims.array,burnin=100,n.thin=1)
print(Output1$Burnin.Summary)
```

```
##                    mu.vect     sd.vect         2.5%        97.5%    P>0
## alpha[1]        3.13991168  0.12265553   2.90082146   3.37577693 1.0000
## alpha[2]        0.32949375  0.17206185  -0.01243767   0.65950190 0.9688
## alpha[3]        0.07747821  0.14328781  -0.20161267   0.35291316 0.7084
## alpha[4]        0.02451773  0.14610770  -0.26539296   0.29974675 0.5642
## alpha[5]       -0.07127580  0.14559454  -0.35855445   0.21022848 0.3072
## alpha[6]       -0.23910454  0.14376971  -0.52462382   0.03952644 0.0466
## alpha[7]        0.38836439  0.13736935   0.12411749   0.65377737 0.9978
## alpha[8]       -0.32343569  0.14851067  -0.61366522  -0.02624460 0.0164
## alphasign[1]    1.00000000  0.00000000   1.00000000   1.00000000 1.0000
## alphasign[2]    0.96880000  0.17387526   0.00000000   1.00000000 0.9688
## alphasign[3]    0.70840000  0.45454457   0.00000000   1.00000000 0.7084
## alphasign[4]    0.56420000  0.49591082   0.00000000   1.00000000 0.5642
## alphasign[5]    0.30720000  0.46137917   0.00000000   1.00000000 0.3072
## alphasign[6]    0.04660000  0.21080163   0.00000000   1.00000000 0.0466
## alphasign[7]    0.99780000  0.04685722   1.00000000   1.00000000 0.9978
## alphasign[8]    0.01640000  0.12702073   0.00000000   0.00000000 0.0164
## beta[1]        -0.21409041  0.22146941  -0.65319245   0.21432333 0.1684
## beta[2]        -0.44706980  0.22203640  -0.86679294  -0.01381099 0.0226
## beta[3]        -0.70175484  0.22506546  -1.15067284  -0.25315208 0.0010
## beta[4]        -0.44002733  0.28128124  -0.98765792   0.10469642 0.0586
## beta[5]         0.13898659  0.22534131  -0.29991760   0.57844514 0.7308
## deviance      253.82378759 13.72778552 229.04329447 282.42032430 1.0000
## rho             0.71841913  0.04473987   0.62339288   0.80098408 1.0000
## sigma           0.40517581  0.02127233   0.36655417   0.44781623 1.0000
## sqrtD           0.65277619  0.06287705   0.54021145   0.78487403 1.0000
## tau.b           2.41238879  0.46575929   1.62330483   3.42667113 1.0000
## tau.e           6.14164538  0.64316357   4.98655189   7.44258253 1.0000
## y[4,3]          3.02250112  0.48259079   2.09578165   3.94614860 1.0000
## y[4,4]          2.79114738  0.51207358   1.78315205   3.81804720 1.0000
```

```
# sink("lab4Tmodel.txt")
# cat("
# model
#         {
#                 for( i in 1 : 64 ) {
#                         for( j in 1 : 4 ) {
#                             s[i, j]<-4*(i-1)+j
#                             y[i, j] ~ dt(mu[i , j],tau.e, df1) # change to t
#
#                             mu[i , j] <- inprod(x[s[i,j],],alpha[])+beta[i]
#                         }
#                         beta[i]~dt(0, tau.b, df2) # change to t
#
#                 }
#
# for( k in 1:8) {
#                 alpha[k]~dnorm(m[k],varinv[k])
#                 alphasign[k] <- step(alpha[k])
# }
#
#                 df1 <- 1/invdf1
#                 invdf1 ~ dunif(0, 0.5)
#                 df2 <- 1/invdf2
#                 invdf2 ~ dunif(0, 0.5)
#
#                 tau.e ~ dgamma(ea,eb)
#                 tau.b~dgamma(ba,bb)
#
#                 sigma <- 1 /sqrt( tau.e)
#                 sqrtD <- 1 /sqrt( tau.b)
#                 rho <- sqrtD*sqrtD/(sigma*sigma + sqrtD *sqrtD)
#         }
#
#     ",fill = TRUE)
#
# sink()
```

```
# T
parameters = c("alpha", "alphasign", "tau.e", "tau.b", "sigma", "sqrtD", "rho", "beta[1:
5]", "y[4,3:4]", "df1", "df2")
run2 = jags(priordata, inits, parameters, "lab4Tmodel.txt",  n.chains=5, n.iter=11000,
 n.burnin=0, n.thin=1)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 245
##    Unobserved stochastic nodes: 87
##    Total graph size: 3261
##
## Initializing model
```

```
Output2 = AddBurnin(run2$BUGSoutput$sims.array,burnin=1000,n.thin=1)
```

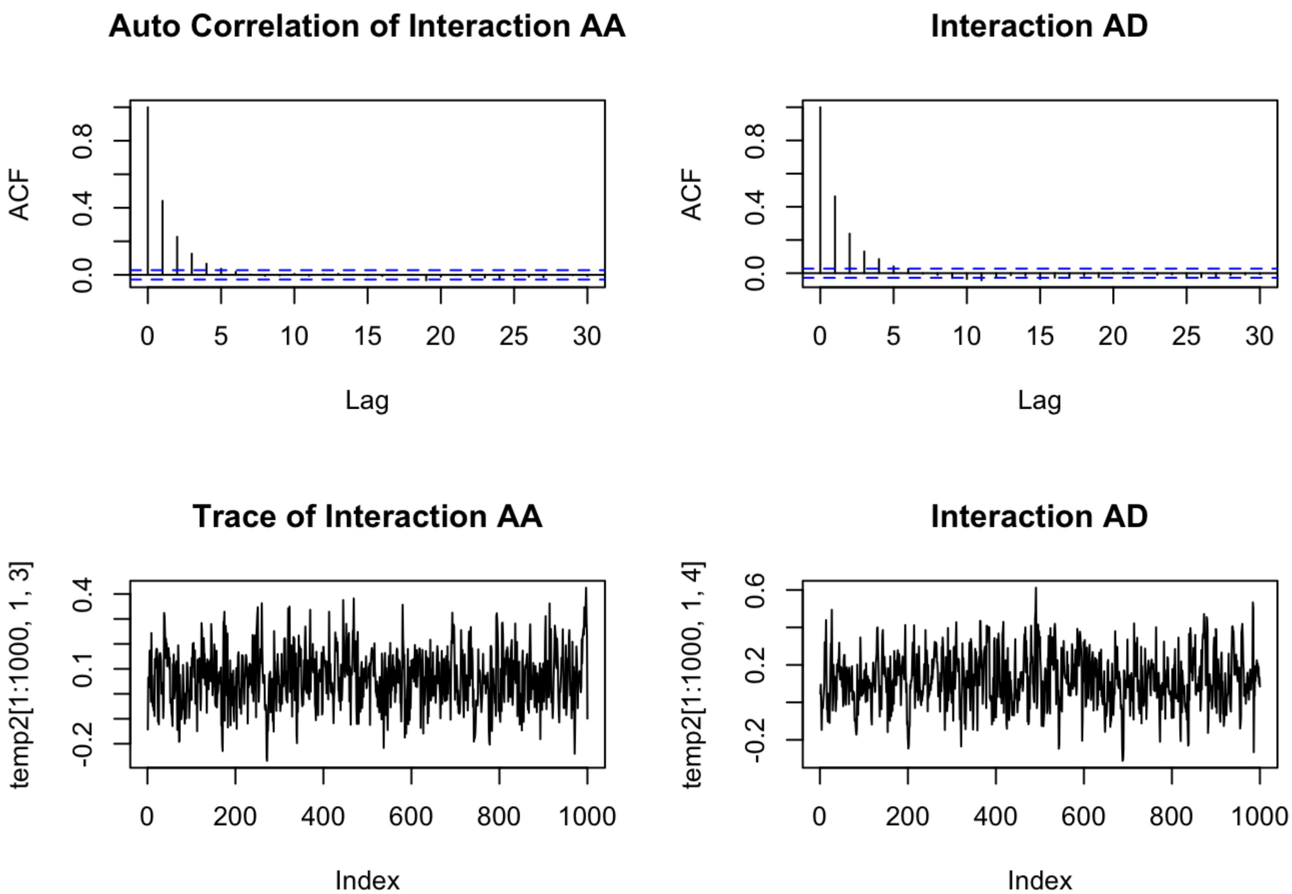
1. Turn in results from the t-model. Be sure to run sufficient iterations.

a. How is the convergence? Show an illustrative autocorrelation function and time-series plot for two parameters of interest.

```
temp2= Output2$Burnin.sims.array
# par(mfrow = c(1,2))
# acf(temp2[,1,"alpha[2]"], main="")
# acf(temp2[1:5000,1,1], main="beta1", lag.max = 200)

# auto-correlation
par(mfrow=c(2,2))
acf(temp2[1:5000,1,3], main="Auto Correlation of Interaction AA", lag.max = 30)
acf(temp2[1:5000,1,4], main="Interaction AD", lag.max = 30)

plot(temp2[1:1000,1,3], main="Trace of Interaction AA", type ="l")
plot(temp2[1:1000,1,4], main="Interaction AD", type ="l")
```



- The parameters shown above converge well.

b. Turn in a table of results for the fixed effects, the two standard deviations sqrtD and □, and the two degrees of freedom parameters. Label rows appropriately and format the table carefully.

```
temp2 = Output2$Burnin.Summary
temp = Output1$Burnin.Summary
desired_rows = c(paste0("alpha[", 1:8, "]"), "sigma", "sqrtD", "df1", "df2")
table = round(temp2[desired_rows, ],3)
table2 = temp[c(paste0("alpha[", 1:8, "]"), "sigma", "sqrtD"), ]
colnames(table2) = c("mean", "sd","2.5%", "97.5%", "P>0")
kable(table)
```

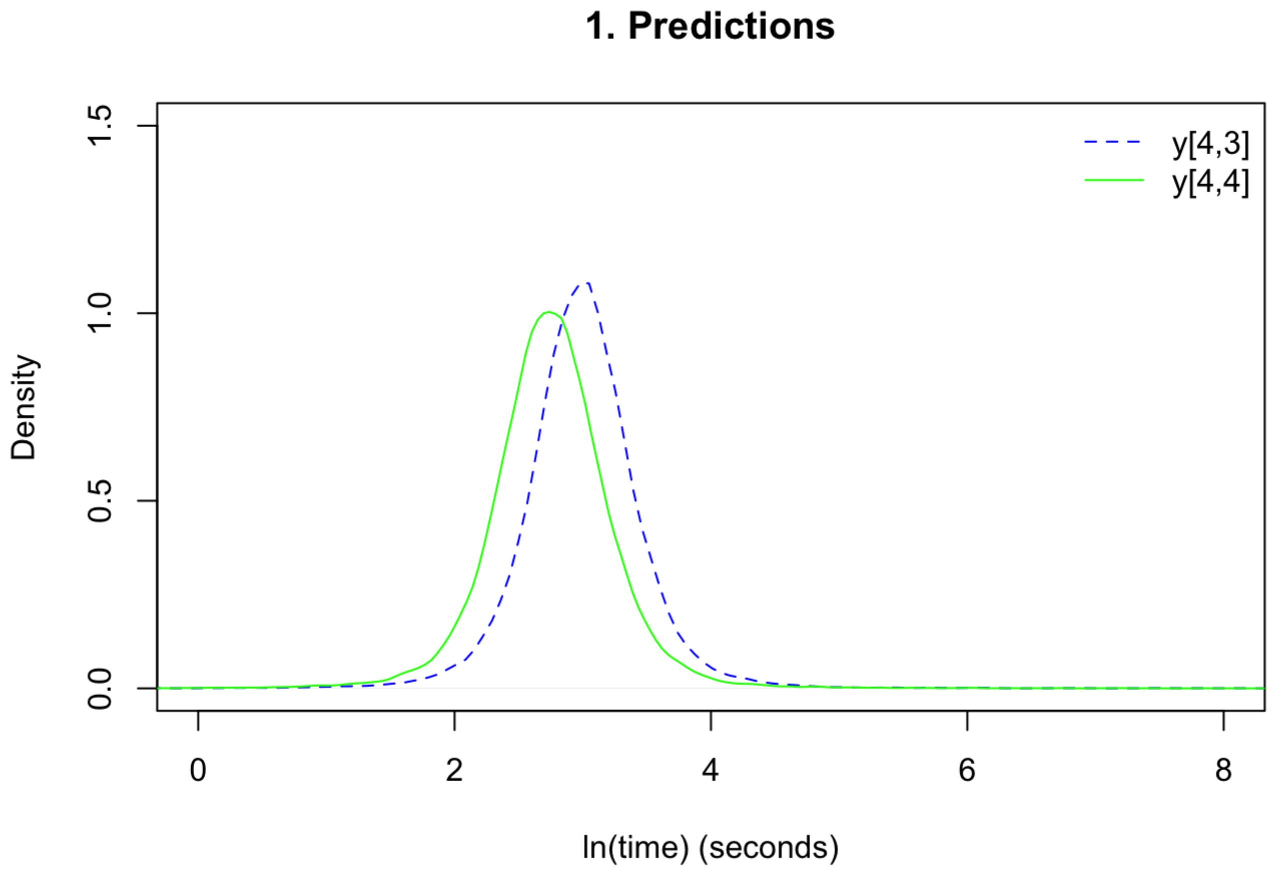
	mu.vect	sd.vect	2.5%	97.5%	P>0
alpha[1]	3.093	0.112	2.878	3.317	1.000
alpha[2]	0.277	0.165	-0.046	0.597	0.954
alpha[3]	0.060	0.115	-0.165	0.287	0.703
alpha[4]	0.120	0.141	-0.159	0.393	0.805
alpha[5]	-0.067	0.113	-0.290	0.151	0.276
alpha[6]	-0.240	0.121	-0.480	0.001	0.025
alpha[7]	0.202	0.155	-0.090	0.518	0.911

	mu.vect	sd.vect	2.5%	97.5%	P>0
alpha[8]	-0.260	0.125	-0.507	-0.016	0.018
sigma	0.256	0.025	0.212	0.309	1.000
sqrtD	0.561	0.073	0.434	0.717	1.000
df1	3.092	0.758	2.097	4.951	1.000
df2	14.006	558.398	2.294	37.078	1.000
2. Compare	the result	s from the	normal m	odel to t	he results from the t model: What changes are there? In particular, what scientific conclusions change?

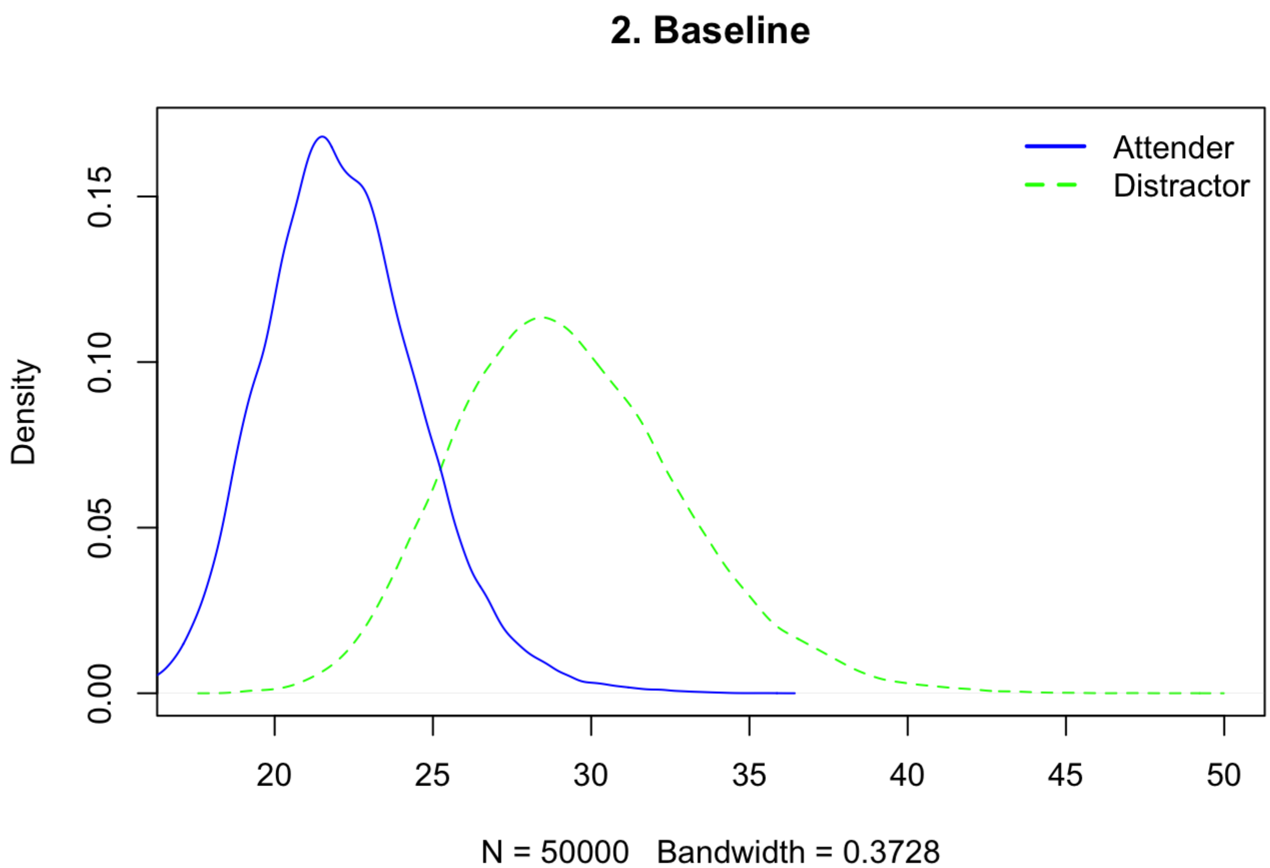
- According to the result, most of the parameter do not have big difference, expect alpha[7] (Normal: 0.39 -> t-model: 0.204). Therefore, the benefit of teaching distractors to distract may be less significant than estimated by the normal model.

3. Reproduce figures 1-5 (see below for the normal model figures) for your t model. Label your figures appropriately.

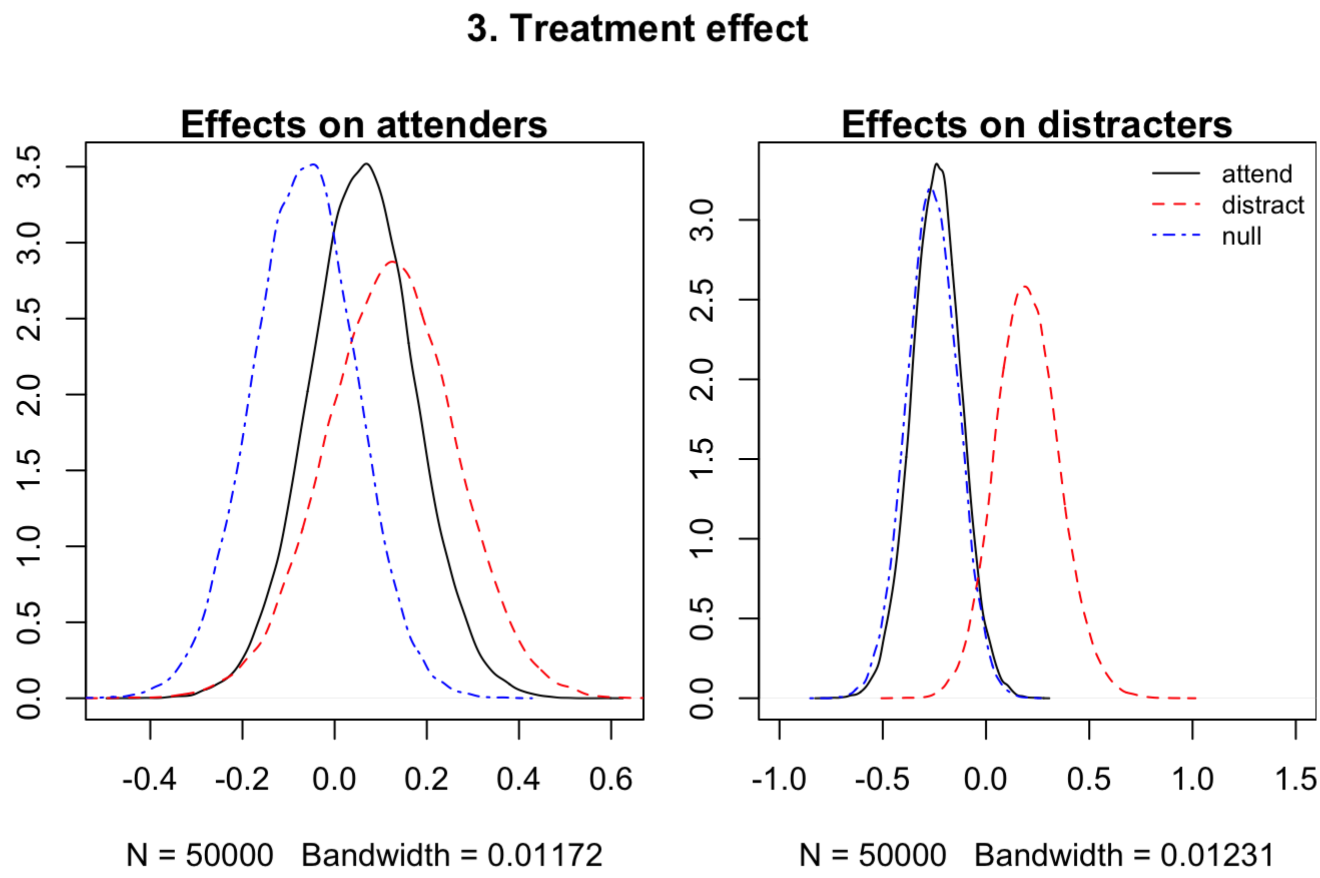
```
out.q3 = Output2$Burnin.sims.matrix
# Plot 1
plot(density((out.q3[,"y[4,3]"])), xlim = c(0, 8), ylim = c(0, 1.5), col = "blue", lty =
2, xlab = "ln(time) (seconds)", main = "")
lines(density((out.q3[,"y[4,4]"])), col = "green", lty = 1)
legend("topright", col = c("blue","green"), legend = c('y[4,3]','y[4,4]'), bty = 'n', lt
y = c(2,1))
title("1. Predictions")
```



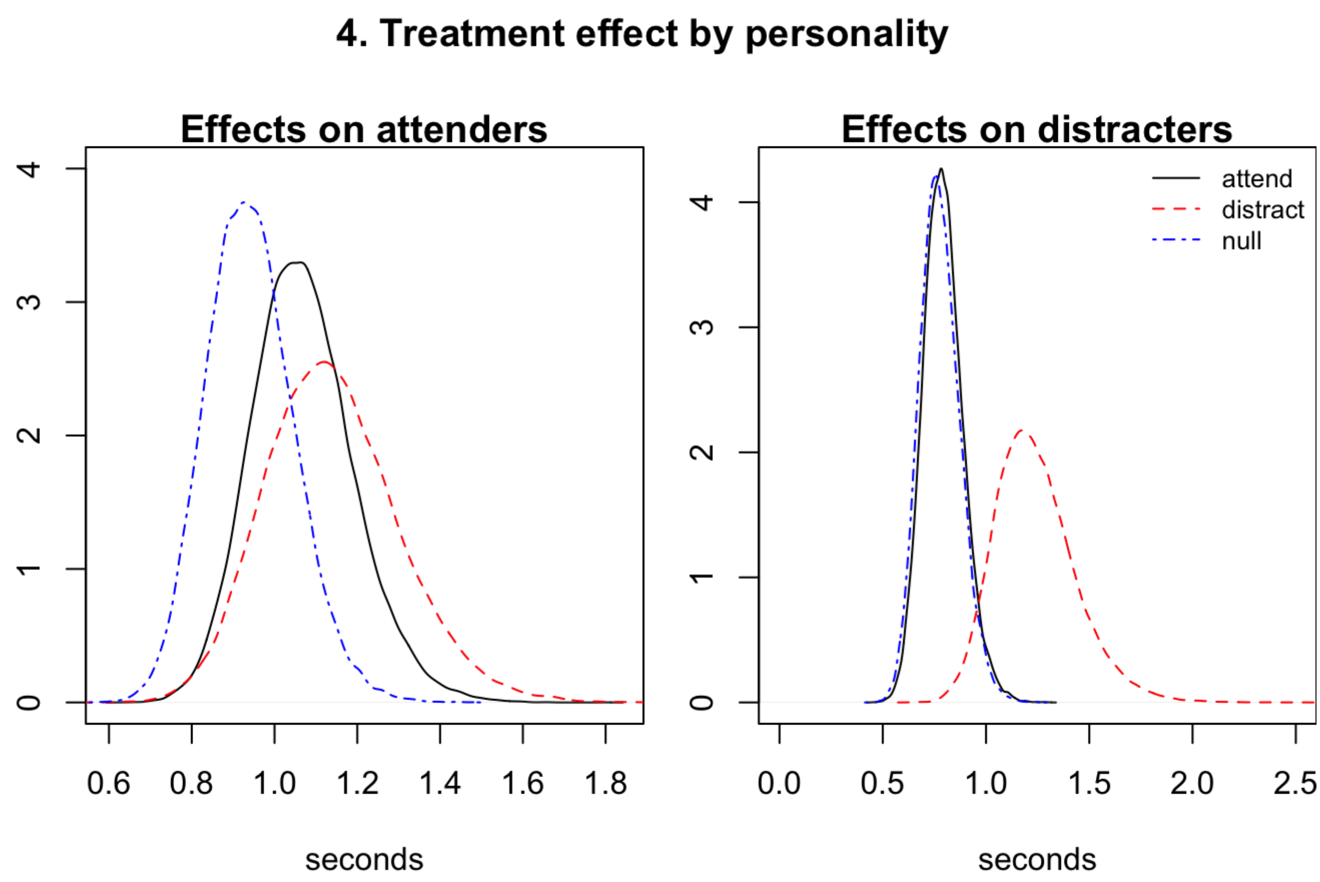

```
# Plot 2
plot(density(exp(out.q3[,c("alpha[1]")] +
               out.q3[,c("alpha[2]")])), ylim = c(0, .17),
     main = "2. Baseline", col = "green", lty = 2)
lines(density(exp(out.q3[,c("alpha[1]")])), col = "blue", lty = 1)
legend("topright", col = c("blue","green"), legend = c('Attender','Distractor'), bty =
'n', lty = c(1,2), lwd = 2)
```



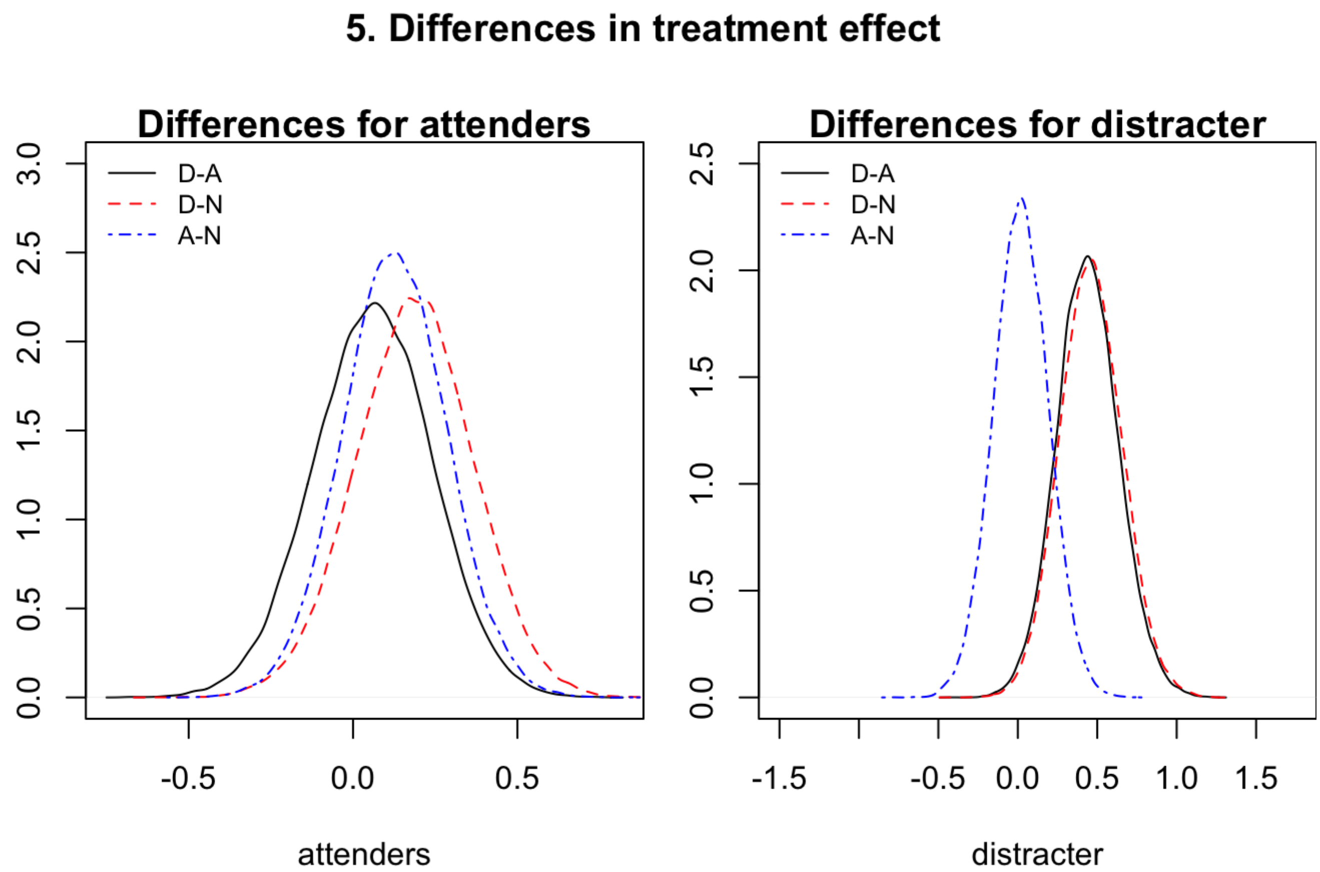

```
# Plot 3
# Treatment effect for attenders
# Log scale
par(mfrow = c(1,2), oma = c(0,0,4,0), mar = c(5,3,1,0))
plot(density((out.q3[, "alpha[3]"])), main = "Effects on attenders", lty = 1)
lines(density((out.q3[,"alpha[4]"])), col = "red", lty = 2)
lines(density((out.q3[,"alpha[5]"])), col = "blue", lty = 4)
plot(density((out.q3[, "alpha[6]"])), main = "Effects on distracters", lty = 1, xlim = c
(-1, 1.5))
lines(density((out.q3[,"alpha[7]"])), col = "red", lty = 2)
lines(density((out.q3[,"alpha[8]"])), col = "blue", lty = 4)
legend("topright", col = c("black", "red", "blue"), legend = c("attend", "distract", "nu
ll"), lty = c(1,2,4), bty = "n", cex = 0.8)
title("3. Treatment effect ", outer = T)
```




```
# Plot 4
# Multiplicative scale
par(mfrow = c(1,2), oma = c(0,0,4,0), mar = c(5,3,1,0))
plot(density(exp(out.q3[, "alpha[3]"])), main = "Effects on attenders", xlab = "seconds"
, ylim = c(0, 4), lty = 1)
lines(density(exp(out.q3[,"alpha[4]"])), col = "red", lty = 2)
lines(density(exp(out.q3[,"alpha[5]"])), col = "blue", lty = 4)
plot(density(exp(out.q3[, "alpha[6]"])), main = "Effects on distracters", xlab = "second
s", xlim = c(0,2.5), lty = 1)
lines(density(exp(out.q3[,"alpha[7]"])), col = "red", lty = 2)
lines(density(exp(out.q3[,"alpha[8]"])), col = "blue", lty = 4)
title("4. Treatment effect by personality", outer = T)
legend("topright", col = c("black", "red", "blue"), legend = c("attend", "distract", "nu
ll"), lty = c(1,2,4), bty = "n", cex = 0.8)
```




```
# Plot 5
# Difference
par(mfrow = c(1,2), oma = c(0,0,4,0), mar = c(5,3,1,0))
plot(density((out.q3[, "alpha[4]"]-out.q3[,"alpha[3]"])), main = "Differences for attend
ers", xlab = "attenders", ylim = c(0, 3), lty = 1)
lines(density((out.q3[,"alpha[4]"]-out.q3[,"alpha[5]"])), col = "red", lty = 2)
lines(density((out.q3[,"alpha[3]"]-out.q3[,"alpha[5]"])), col = "blue", lty = 4)
legend("topleft", col = c("black", "red", "blue"), legend = c("D-A", "D-N", "A-N"), lty
 = c(1,2,4), bty = "n", cex = 0.8)
plot(density((out.q3[, "alpha[7]"]-out.q3[,"alpha[6]"])), main = "Differences for distra
cter", xlab = "distracter", ylim = c(0, 2.5), xlim = c(-1.5, 1.75), lty = 1)
lines(density((out.q3[,"alpha[7]"]-out.q3[,"alpha[8]"])), col = "red", lty = 2)
lines(density((out.q3[,"alpha[6]"]-out.q3[,"alpha[8]"])), col = "blue", lty = 4)
legend("topleft", col = c("black", "red", "blue"), legend = c("D-A", "D-N", "A-N"), lty
 = c(1,2,4), bty = "n", cex = 0.8)
title("5. Differences in treatment effect" ,outer = T)
```



4. Invent another prior for the df, and in one sentence explain its properties (ie support, mean, sd or other characteristics) and why it is better than the above prior.

- I suggust that df = Gamma(3969/200, 63/200), since the distribution has mean of 63 (df = n-1 for y in our dataset), also has sd of 200. It may become better since the mean will become closer to sample size and keep wide range of df.