AdvancedR_Final

Hai Long, Le

August 20, 2019

```
# Required Libraries
library(readr)
library(dplyr)
library(magrittr)
library(ggplot2)
library(shiny)
library(rstan)
library(shinystan)
library(ggiraph)
library(gganimate)
library(reshape2)
library(tidyr)
library(ggmap)
```

```
data <- read_csv("D:/UCD/Advanced R/Assignment/Final Project/exo_data.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   id = col_character(),
##   age = col_logical(),
##   meth = col_character(),
##   recency = col_character(),
##   r_asc = col_character(),
##   decl = col_character(),
##   lists = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
## Warning: 2 parsing failures.
##  row col           expected actual
file
## 1712 age 1/0/T/F/TRUE/FALSE 0.0055 'D:/UCD/Advanced R/Assignment/Final Project/exo_
data.csv'
## 2970 age 1/0/T/F/TRUE/FALSE 3.0    'D:/UCD/Advanced R/Assignment/Final Project/exo_
data.csv'
```

```
str(data)
```

```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 3659 obs. of  25 variable
s:
##  $ id       : chr  "KOI-1843.03" "Kepler-974 b" "KOI-1843.02" "Kepler-9 b" ...
##  $ flag     : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ mass     : num  0.0014 NA NA 0.25 0.17 0.022 0.0321 NA 0.6 5.21 ...
##  $ radius   : num  0.054 0.14 0.071 0.84 0.82 0.147 NA 0.192 1.24 NA ...
##  $ period   : num  0.177 4.194 6.356 19.224 39.031 ...
##  $ axis     : num  0.0048 0.039 0.052 0.143 0.229 0.0271 0.053 NA 0.0449 1.33 ...
##  $ ecc      : num  NA NA NA 0.0626 0.0684 NA 0.06 NA NA 0.15 ...
##  $ per      : num  NA NA NA NA NA ...
##  $ lon      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ asc      : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ incl     : num  72 89.4 88.2 87.1 87.2 ...
##  $ temp     : num  NA NA NA 707 558 ...
##  $ age      : logi  NA NA NA NA NA NA ...
##  $ meth     : chr  "transit" "transit" "transit" "transit" ...
##  $ year     : num  2012 NA NA 2010 2010 ...
##  $ recency  : chr  "13/07/15" "17/11/28" NA "15/12/03" ...
##  $ r_asc    : chr  "19 00 03.14" "19 00 03.14" "19 00 03.14" "19 02 17" ...
##  $ decl     : chr  "+40 13 14.7" "+40 13 14.7" "+40 13 14.7" "+38 24 03" ...
##  $ dist     : num  NA NA NA 650 650 ...
##  $ host_mass: num  0.52 0.52 0.52 1.07 1.07 1.07 0.69 0.83 1.07 0.82 ...
##  $ host_rad : num  0.5 0.5 0.5 1.02 1.02 1.02 NA 0.79 NA NA ...
##  $ host_met : num  0.07 0.07 0.07 0.12 0.12 0.12 NA -0.01 -0.02 -0.18 ...
##  $ host_temp: num  3687 3687 3687 5777 5777 ...
##  $ host_age : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ lists    : chr  "Controversial" "Confirmed planets" "Controversial" "Confirmed p
lanets" ...
##  - attr(*, "problems")=Classes 'tbl_df', 'tbl' and 'data.frame': 2 obs. of  5 varia
bles:
##   ..$ row     : int  1712 2970
##   ..$ col     : chr  "age" "age"
##   ..$ expected: chr  "1/0/T/F/TRUE/FALSE" "1/0/T/F/TRUE/FALSE"
##   ..$ actual  : chr  "0.0055" "3.0"
##   ..$ file    : chr  "'D:/UCD/Advanced R/Assignment/Final Project/exo_data.csv'"
"'D:/UCD/Advanced R/Assignment/Final Project/exo_data.csv'"
##  - attr(*, "spec")=
##   .. cols(
##   ..   id = col_character(),
##   ..   flag = col_double(),
##   ..   mass = col_double(),
##   ..   radius = col_double(),
##   ..   period = col_double(),
##   ..   axis = col_double(),
##   ..   ecc = col_double(),
##   ..   per = col_double(),
##   ..   lon = col_double(),
##   ..   asc = col_double(),
```

```
##   ..   incl = col_double(),
##   ..   temp = col_double(),
##   ..   age = col_logical(),
##   ..   meth = col_character(),
##   ..   year = col_double(),
##   ..   recency = col_character(),
##   ..   r_asc = col_character(),
##   ..   decl = col_character(),
##   ..   dist = col_double(),
##   ..   host_mass = col_double(),
##   ..   host_rad = col_double(),
##   ..   host_met = col_double(),
##   ..   host_temp = col_double(),
##   ..   host_age = col_double(),
##   ..   lists = col_character()
##   .. )
```

1) Import the dataset exo_data.csv as a tibble. Columns 1, 16, 17, 18, 25 should be characters. Columns 2, 14 should be factors. Column 15 should be integers.The remaining columns should be doubles.

```
#Columns 1, 16, 17, 18, 25 ALREADY be characters.

data$year %<>% as.integer # column 15 = year

data$flag %<>% as.factor  # column 2 = flag

#unique(data[,14]) # There are 5 different levels and NA for col14=meth.

data$meth %<>% as.factor
```

2) Exclude the exoplanets with an unknown method of discovery.

```
data <- data %>% drop_na(meth)
```

Dataset now reduce to 3596 obs after remove NA of “meth”.

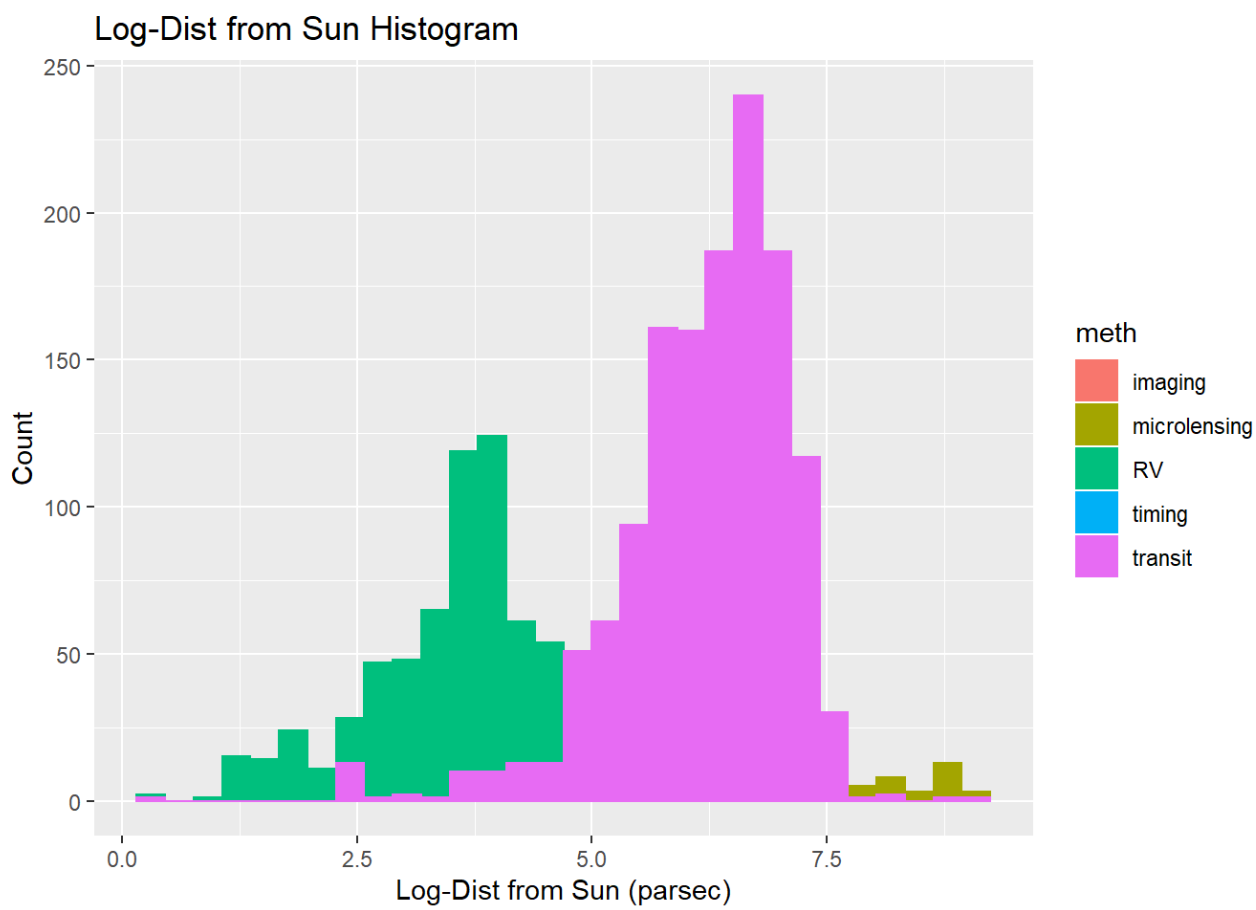
3) Create a histogram for the log-distances from

the Sun, highlighting the methods of discovery.

```
ggplot(data, aes(x=log(dist), fill=meth, color=meth)) +
  geom_histogram(position="identity") +
  labs(title="Log-Dist from Sun Histogram",x="Log-Dist from Sun (parsec)", y = "Coun
t")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 1409 rows containing non-finite values (stat_bin).
```



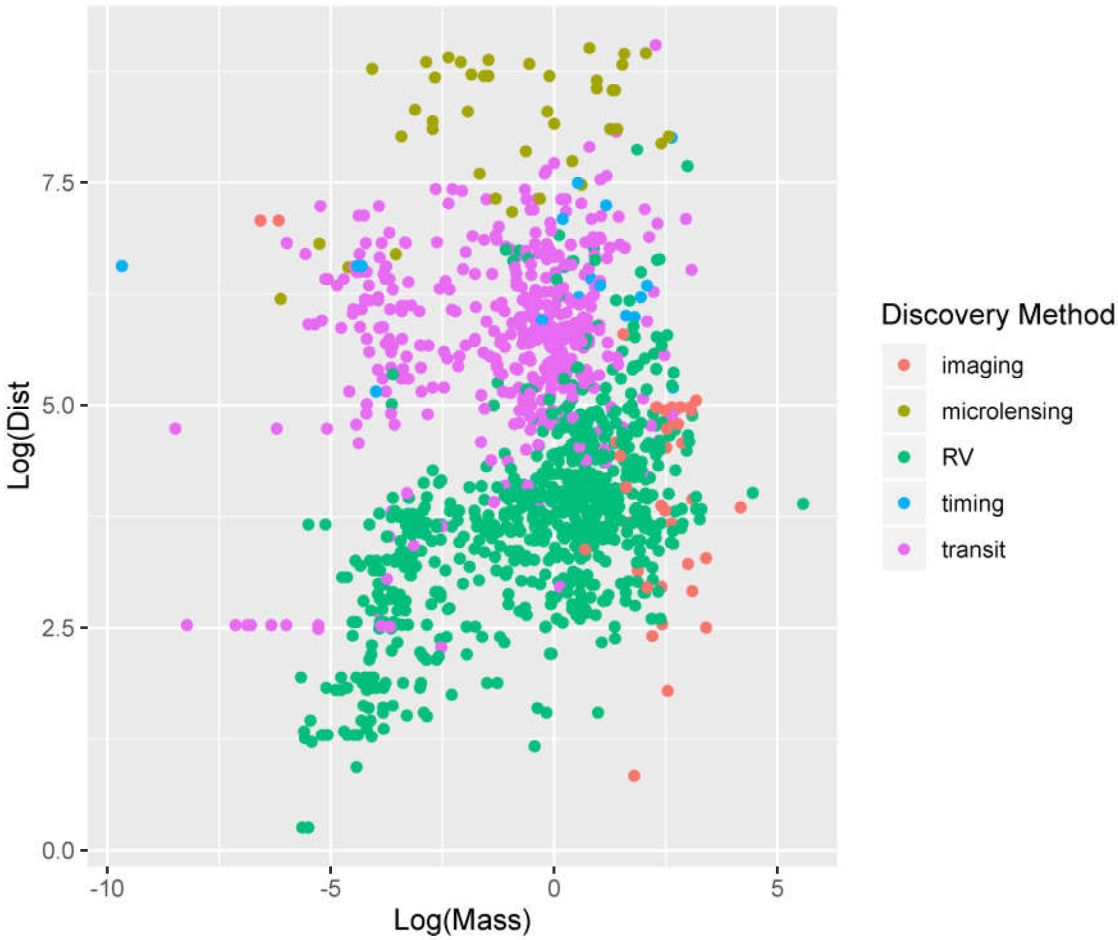
4) Create scatterplots of the log-mass versus log-distances, separating by methods of discovery. Hovering with the cursor highlights

the point and displays its name, and, if you click, the exoplanet's page on the Open Exoplanet Catalogue will be opened.

(paste the id after http://www.openexoplanetcatalogue.com/planet/ (http://www.openexoplanetcatalogue.com/planet/)).

```
data$onclick <- sprintf("window.open(\"%s%s\")",
                        "http://www.openexoplanetcatalogue.com/planet/",
                        data$id)

gg_graph = ggplot(data,
                  aes(x = log(mass),
                      y = log(dist),
                      color = meth)) +
                  xlab('Log(Mass)') +
                  ylab('Log(Dist') +
                  scale_color_discrete(name="Discovery Method")+
                  geom_point_interactive(aes(data_id = id,
                             tooltip = id,
                             onclick = onclick))

ggiraph(code = print(gg_graph))
```

```
## Warning: Removed 2355 rows containing missing values
## (geom_interactive_point).
```



5) Rename the radius into jupiter_radius, and create a new column called earth_radius which is 11.2 / the Jupiter radius.

```
data <- data %>%
          rename(jupiter_radius = radius ) # rename() function from tidyverse with pip
e.

data <- data %>%
          mutate(earth_radius = jupiter_radius/11.2 )
```

6) Focus only on the rows where log-radius of Earth and log-period have no missing values, and perform kmeans with four clusters on these two columns.

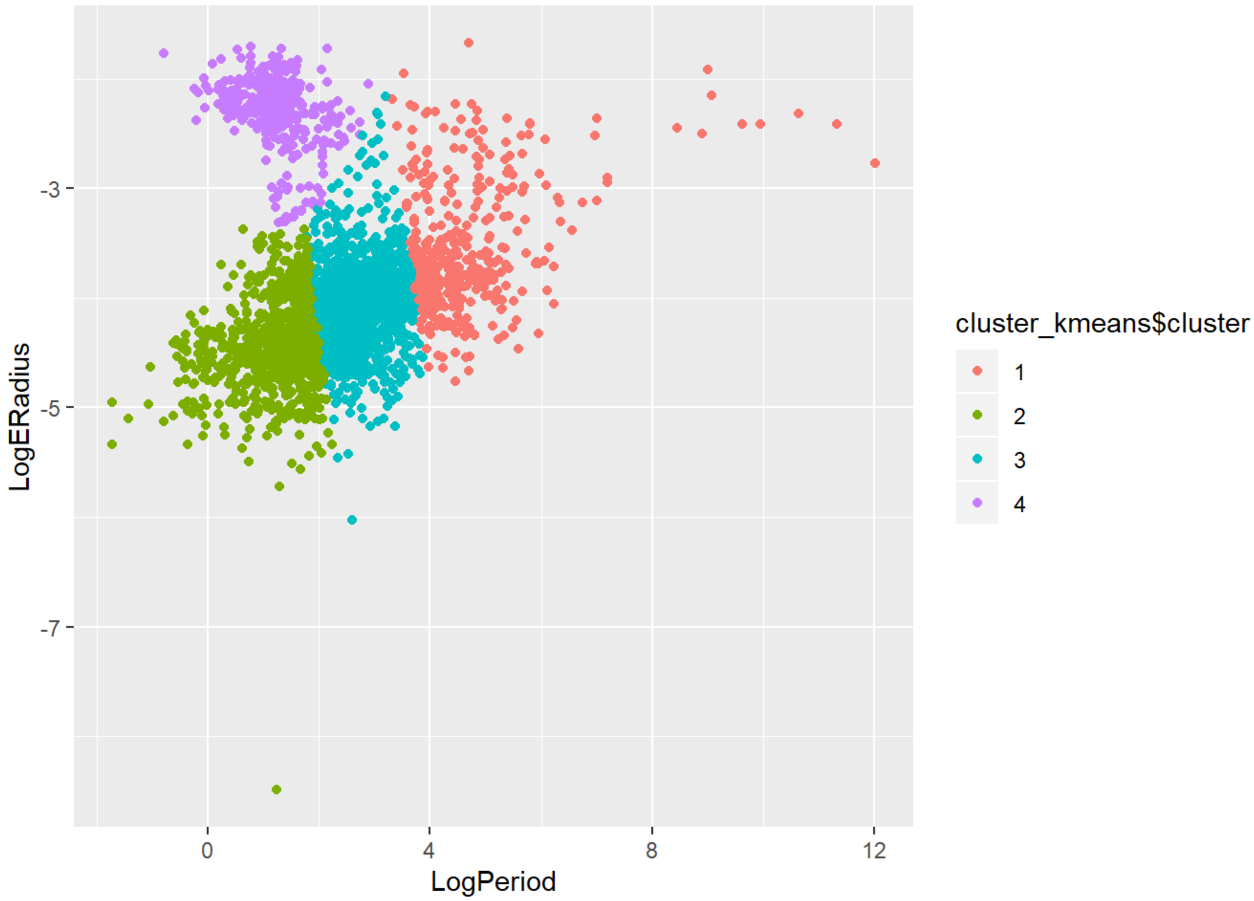
```
data_clustering <- data # create new df for clustering from data

# Focus only on the rows where radius of Earth and period have no missing values
data_clustering <- data %>% drop_na(earth_radius, period)  # 2732 obs

#log-radius of Earth and log-period
data_clustering <- data_clustering %>%
                      mutate(LogERadius = log(earth_radius),
                             LogPeriod  = log(period))



# data to perform Kmeans
data_kmeans <- data_clustering %>%
                      select(LogERadius,LogPeriod)

# perform k-means
set.seed(123)
cluster_kmeans  <- kmeans(data_kmeans, 4)

table(cluster_kmeans$cluster)
```

```
##
##    1    2    3    4
##  416  798 1133  385
```

7*) Add the clustering labels to the dataset through a new factor column called ‘type’, with levels ‘rocky’, ‘hot_jupiters’, ‘cold_gas_giants’, ‘others’;

similarly to https://en.wikipedia.org/wiki/Exoplanet#/media/File:ExoplanetPopulations-20170616.png (https://en.wikipedia.org/wiki/Exoplanet#/media/File:ExoplanetPopulations-20170616.png)

```
cluster_kmeans$cluster <- as.factor(cluster_kmeans$cluster)

ggplot(data_kmeans, aes(LogPeriod,LogERadius ,color = cluster_kmeans$cluster)) + geom_
point()
```




```
# Using https://en.wikipedia.org/wiki/Exoplanet#/media/File:ExoplanetPopulations-20170
616.png we have:
# 1 = Rocky 1133
# 2 = cold_gas_giants 416
# 3 = hot_jupiters 385
# 4 = others 798


data_clustering$type <- cluster_kmeans$cluster

data_clustering$type <- as.numeric(data_clustering$type)

data_clustering$type[data_clustering$type == 1] <- "Rocky"
data_clustering$type[data_clustering$type == 2] <- "cold_gas_giants"
data_clustering$type[data_clustering$type == 3] <- "hot_jupiters"
data_clustering$type[data_clustering$type == 4] <- "others"
```

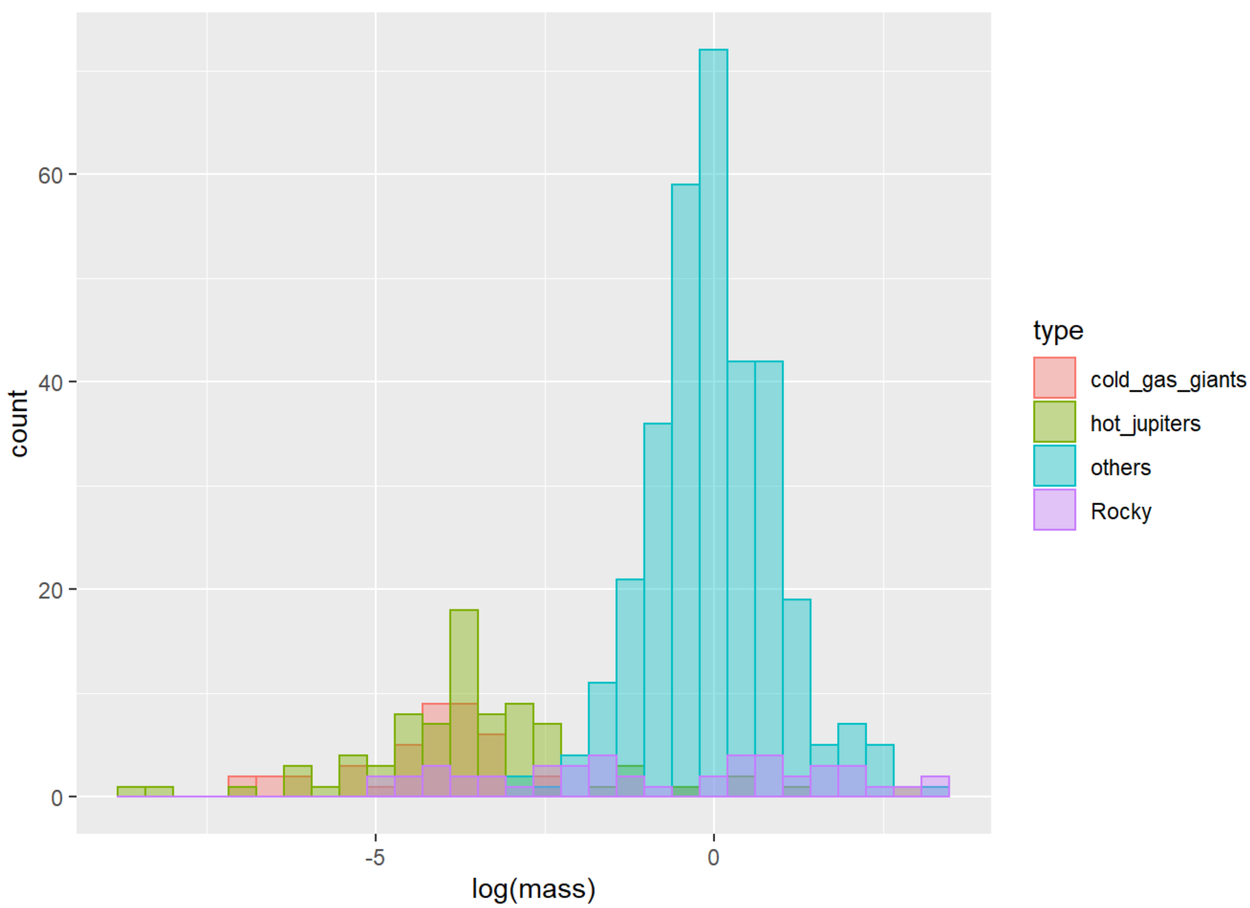
```
table(data_clustering$type) ## checking
```

```
##
## cold_gas_giants     hot_jupiters          others           Rocky
##             798             1133             385             416
```

8) Use a histogram and a violin plot to illustrate how these clusters relate to the log-mass of the exoplanet.

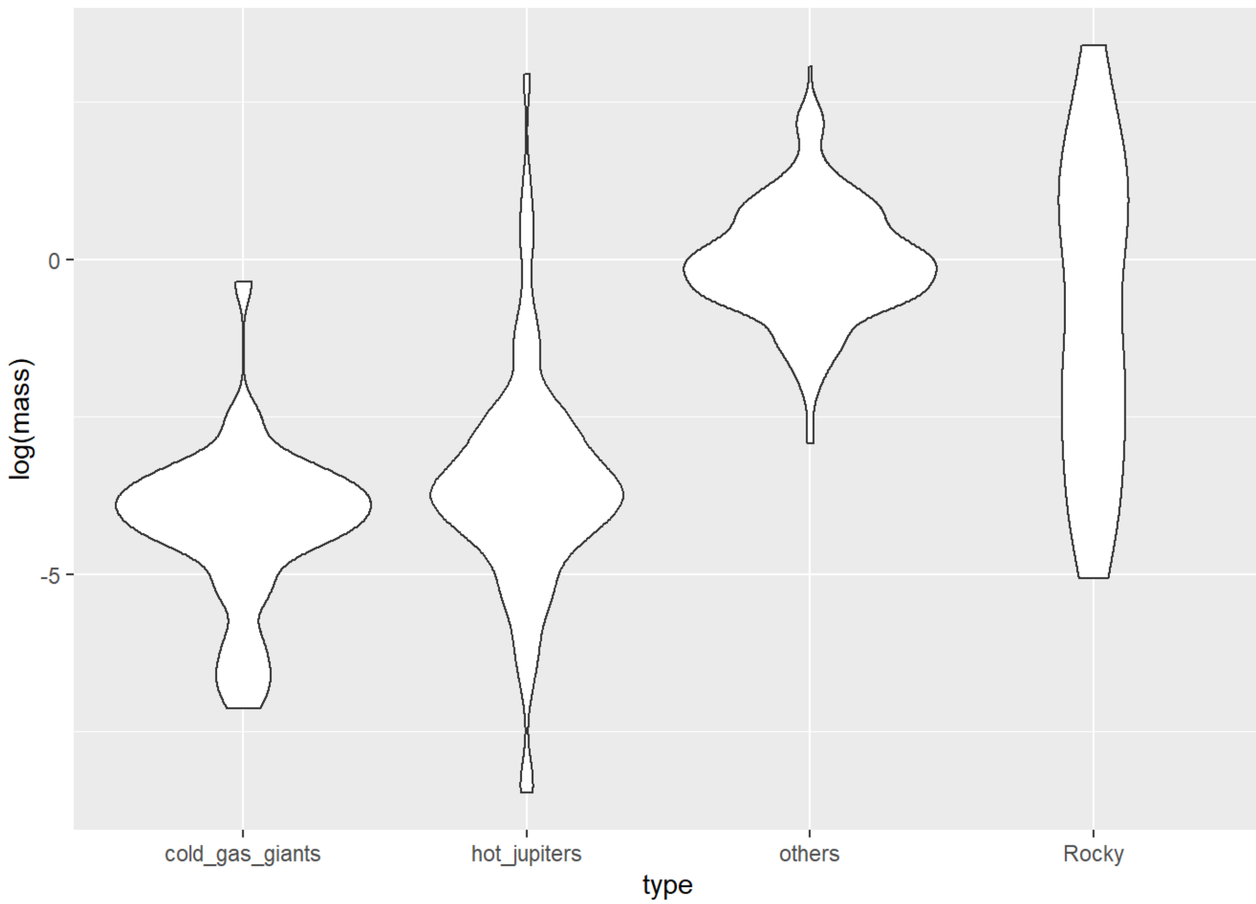
```
# Histogram
ggplot(data_clustering, aes(x = log(mass))) +
                            geom_histogram(aes(color = type, fill = type),
                                            position = "identity", bins = 30, alpha =
0.4)
```

```
## Warning: Removed 2236 rows containing non-finite values (stat_bin).
```



```
# Violin
ggplot(data_clustering, aes(x = type, y = log(mass))) +
  geom_violin()
```

```
## Warning: Removed 2236 rows containing non-finite values (stat_ydensity).
```



9*) transform r_asc and decl into the equivalent values in seconds and use these as coordinates to represent a celestial map for the exoplanets.

```
head(data$r_asc) # [hh mm ss]
```

```
## [1] "19 00 03.14" "19 00 03.14" "19 00 03.14" "19 02 17"    "19 02 17"
## [6] "19 02 17"
```

```
head(data$decl)   #Declination [+/-dd mm ss]
```

```
## [1] "+40 13 14.7" "+40 13 14.7" "+40 13 14.7" "+38 24 03"   "+38 24 03"
## [6] "+38 24 03"
```

```
library(lubridate)
```

```
# conver r_asc to seconds and save as r_asc_sec
data$r_asc <- gsub(" ", ":", data$r_asc, fixed=TRUE) # convert to hh:mm:ss
data$r_asc <- hms(data$r_asc)
```

```
## Warning in .parse_hms(..., order = "HMS", quiet = quiet): Some strings
## failed to parse, or all strings are NAs
```

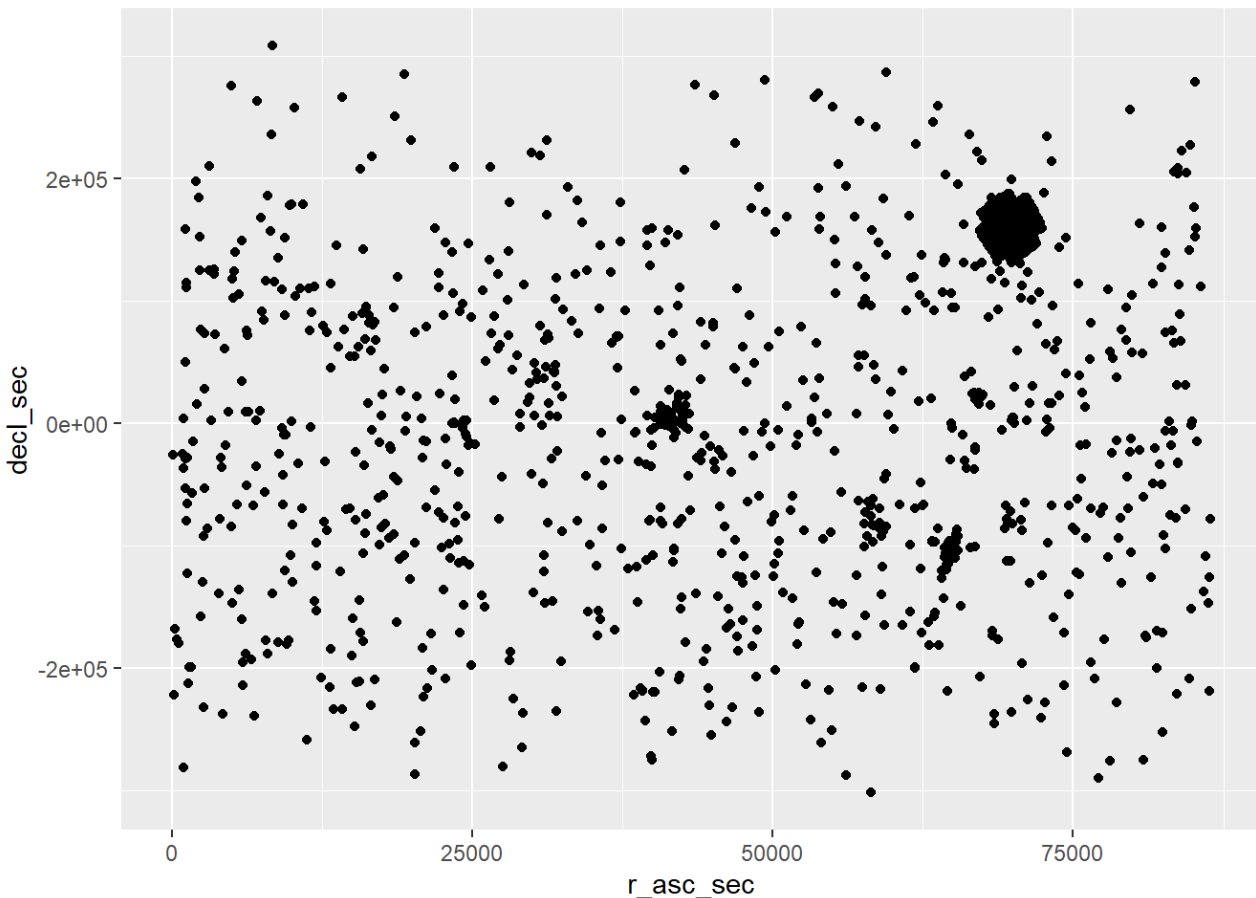
```
data$r_asc_sec <- period_to_seconds(data$r_asc)

# convert Declination to seconds and save as decl_sec
data$decl <- gsub(" ", ":", data$decl, fixed=TRUE) # convert to dd:mm:ss, where dd=360
0ss
data$decl <- hms(data$decl) # for Decl, dd is similar to hh where :=3600ss
```

```
## Warning in .parse_hms(..., order = "HMS", quiet = quiet): Some strings
## failed to parse, or all strings are NAs
```

```
data$decl_sec <- period_to_seconds(data$decl)
```

```
# scatter plot represents a celestial map for the exoplanets
ggplot(data, aes(r_asc_sec, decl_sec)) +
                            geom_point()
```

```
## Warning: Removed 1 rows containing missing values (geom_point).
```



10) create an animated time series where multiple lines illustrate the evolution over time of the total number of exoplanets discovered for each method up to that year.

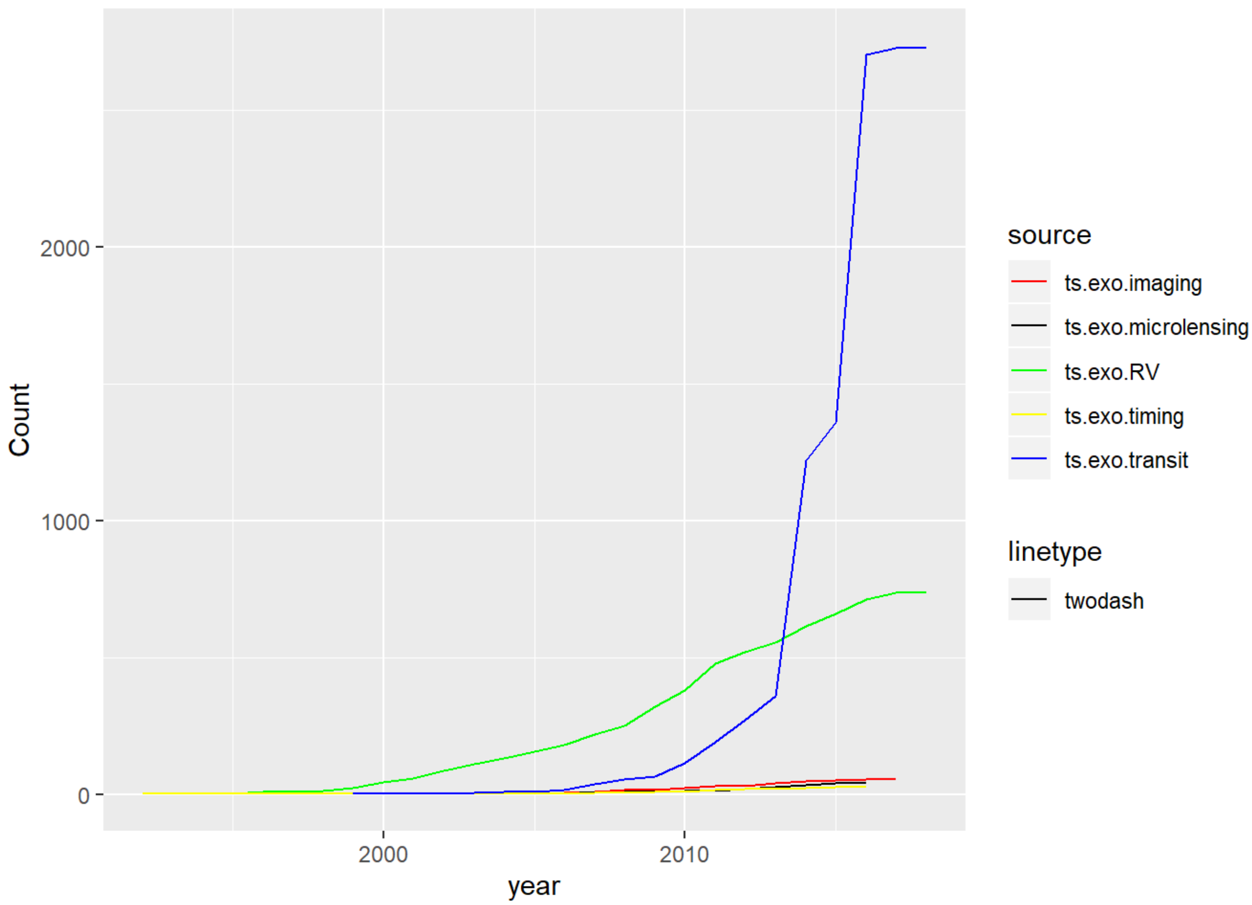
```
methods <- c()
methods <- levels(data$meth)
```

```
for (i in 1:length(methods))
{
  assign(paste0("ts.exo.", methods[i]),  data%>%
                                       filter(meth == methods[i]) %>%
                                               group_by(year) %>%  # group by "year"
                                                   summarise(Count = length(meth)) %>%
                                                           mutate(Count = cumsum(Coun
t))) #cummulative-count thru each year
}
```

```
ts.exo <- bind_rows(list(ts.exo.transit = ts.exo.transit, ts.exo.timing = ts.exo.timin
g, ts.exo.RV = ts.exo.RV,
                         ts.exo.microlensing= ts.exo.microlensing,ts.exo.imaging=ts.ex
o.imaging), .id = 'source')
```

```
ggplot(ts.exo, aes(x = year, y = Count)) +
  geom_line(aes(color = source, linetype = "twodash")) +
  scale_color_manual(values = c("red", "black", "green", "yellow", "blue"))
```

```
## Warning: Removed 3 rows containing missing values (geom_path).
```



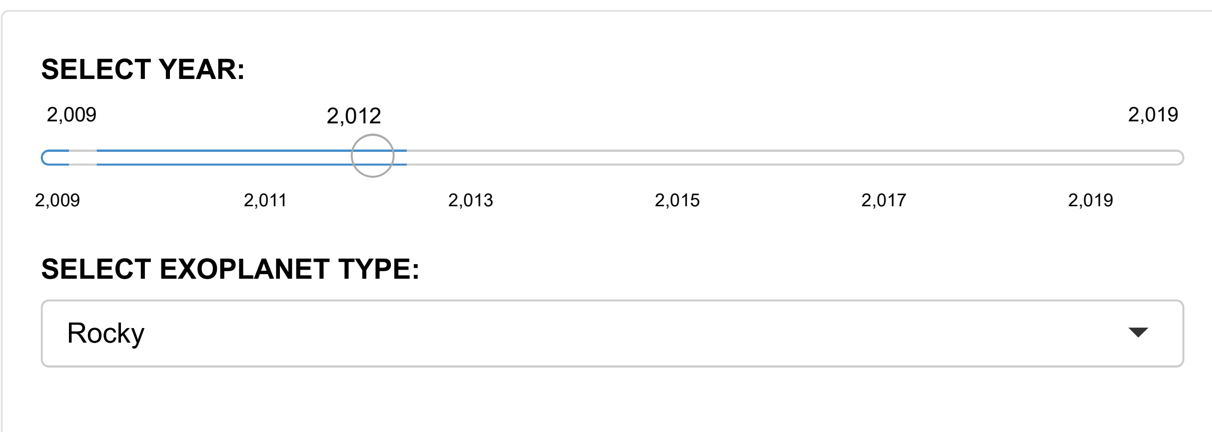
11*) create an interactive plot with Shiny where you can select the year (slider widget, with values >= 2009) and exoplanet type. Exoplanets appear as points on a scatterplot (log-mass vs log-distance coloured by method) only if they have already been discovered. If type is equal to “all” all types are plotted together.

Shiny code:

```r
# Define UI for application that draws a histogram
ui <- fluidPage(

  # Application title
  titlePanel("scatterplot of log-mass vs log-distance of Exoplanet"),

  # Sidebar with a slider input for number of bins
  sidebarLayout(
    sidebarPanel(
      sliderInput("id1",
                  "SELECT YEAR:",
                  min = 2009,
                  max = 2019,
                  value = 2012)
    ,

    # Let user select "exoplanet type" by SelectInput
    selectInput("id2",
                "SELECT EXOPLANET TYPE:",
                choices = c("Rocky","cold_gas_giants", "hot_jupiters", "others", "al
l"))
    ),

    # Show a plot of the generated distribution
    mainPanel(
      plotOutput("scatterPlot") # was distPlot
    )
  )
)

# Define server logic required to draw a histogram
server <- function(input, output) {

  output$scatterPlot <- renderPlot({
    # Get the User Input and cols will be used
    xdata    <- data_clustering[,c("mass","dist","year", "meth", "type")]

    idyear = input$id1
    idmeth = input$id2

    # Filter the data based on user input.
    xdata <- xdata %>% filter(year <= idyear)

    if (idmeth == "all") # If the user want to see all "Type"
    {
      ggplot(xdata,aes(x = log(mass),
                       y = log(dist),
                       color = meth)) +
```

```r
        xlab('Log(Mass)') +
        ylab('Log(Dist') +
        geom_point() +
        facet_wrap( ~ type, ncol=2)
    }

    else # if not, filter out the "type"
    {

    xdata <- xdata %>% filter(type == idmeth)

    # draw the Scatter Plot with the specified Year and Type
    ggplot(xdata,aes(x = log(mass),
                     y = log(dist),
                     color = meth)) +
                     xlab('Log(Mass)') +
                     ylab('Log(Dist') +
                   geom_point()
    }

  })
}

# Run the application
shinyApp(ui = ui, server = server)
```

scatterplot of log-mass vs log-distance of Exoplanet



12) Use STAN to perform likelihood maximisation on a regression model where log-period is the response variable and the logs of host_mass, host_temp and axis are the covariates (exclude rows that contain at least one missing value). Include an intercept term in the regression model.

```
fileName <- "D:/UCD/Advanced R/Assignment/Final Project/MLR_Project.stan"
stan_code <- readChar(fileName, file.info(fileName)$size)
cat(stan_code)
```

```
## data {
##   // Define all the data here
##   int<lower=0> N; // number of observations
##   int<lower=0> K; // number of explanatory variables
##   matrix[N, K] x; // explanatory variables
##   vector[N] y; // response variable
## }
## parameters {
##   // Define parameters here
##   real alpha; // intercept
##   vector[K] beta; // slope
##   real<lower=0> sigma; // residual sd
## }
## model {
##   // Model Likelihood
##   y ~ normal(alpha + x * beta, sigma);
## }
##
```

```
stan.data <- data_clustering[,c("host_mass","host_temp","axis", "period")]

stan.data.complete <- na.omit(stan.data) #exclude rows that contain at least one missi
ng value

stan.data.complete <- stan.data.complete %>%                          # Log scale of a
ll Variables.
                                  mutate(host_mass = log(host_mass),
                                         host_temp = log(host_temp),
                                         axis = log(axis),
                                         period = log(period))



# to save time when you recompile an already compiled file:
rstan_options(auto_write = TRUE)

# Always good to enable parallel running if available:
options(mc.cores = parallel::detectCores())

# Set up your data into the correct format, save it as a list with the same names as i
n the stan file
data_mlr = list(N = nrow(stan.data.complete), #  number of observations
               K = 3,                        #  number of explanatory variables
               y = stan.data.complete$period,
               x =  as.matrix(stan.data.complete[,c("host_mass","host_temp","axis")]))
# x now is matrix with [N,K] dimensions



# Call Model from separate Stan file
stan_model_mlr = stan_model('D:/UCD/Advanced R/Assignment/Final Project/MLR_Project.st
an')

#Fit the model with either the optimizing (Maximum likelihood version because have not
specified Prior)
stan_run_mlr = optimizing(stan_model_mlr, data = data_mlr)
```

```
# Print the output
print(stan_run_mlr)
```

```
## $par
##      alpha    beta[1]    beta[2]    beta[3]      sigma
##  7.2181628 -0.2599741 -0.1578366  1.4830355  0.2237078
##
## $value
## [1] 618.4012
##
## $return_code
## [1] 0
##
## $theta_tilde
##         alpha    beta[1]    beta[2]  beta[3]     sigma
## [1,] 7.218163 -0.2599741 -0.1578366 1.483035 0.2237078
```

13) Extend the model in (12) by specifying standard Gaussian priors for the intercept and slope terms, and a Gamma(1,1) prior for the standard deviation of errors. Obtain approximate samples from the posterior distribution of the model.

```
fileName1 <- "D:/UCD/Advanced R/Assignment/Final Project/MLR_Project_Prior.stan"
stan_code1 <- readChar(fileName1, file.info(fileName1)$size)
cat(stan_code1)
```

```
## data {
##   // Define all the data here
##   int<lower=0> N; // number of observations
##   int<lower=0> K; // number of explanatory variables
##   matrix[N, K] x; // explanatory variables
##   vector[N] y; // response variable
## }
## parameters {
##   // Define parameters here
##   real alpha; // intercept
##   vector[K] beta; // slope
##   real<lower=0> sigma; // residual sd
## }
## model {
##   // Prior for Intercept alpha, standard Gaussian
##   alpha ~ normal(0,1);
##
##   // Prior for slope beta, standard Gaussian
##   beta ~ normal(0,1);
##
##   // Prior for Sigma, Gamma(1,1)
##   sigma ~ gamma(1,1);
##
##   // Model Likelihood
##   y ~ normal(alpha + x * beta, sigma);
## }
##
```

```
stan_model_mlr_prior = stan_model('D:/UCD/Advanced R/Assignment/Final Project/MLR_Proj
ect_Prior.stan')


# The full Bayesian way
stan_run_lr_bayes = sampling(stan_model_mlr_prior,
                             data = data_mlr)
```

```
print(stan_run_lr_bayes)
```

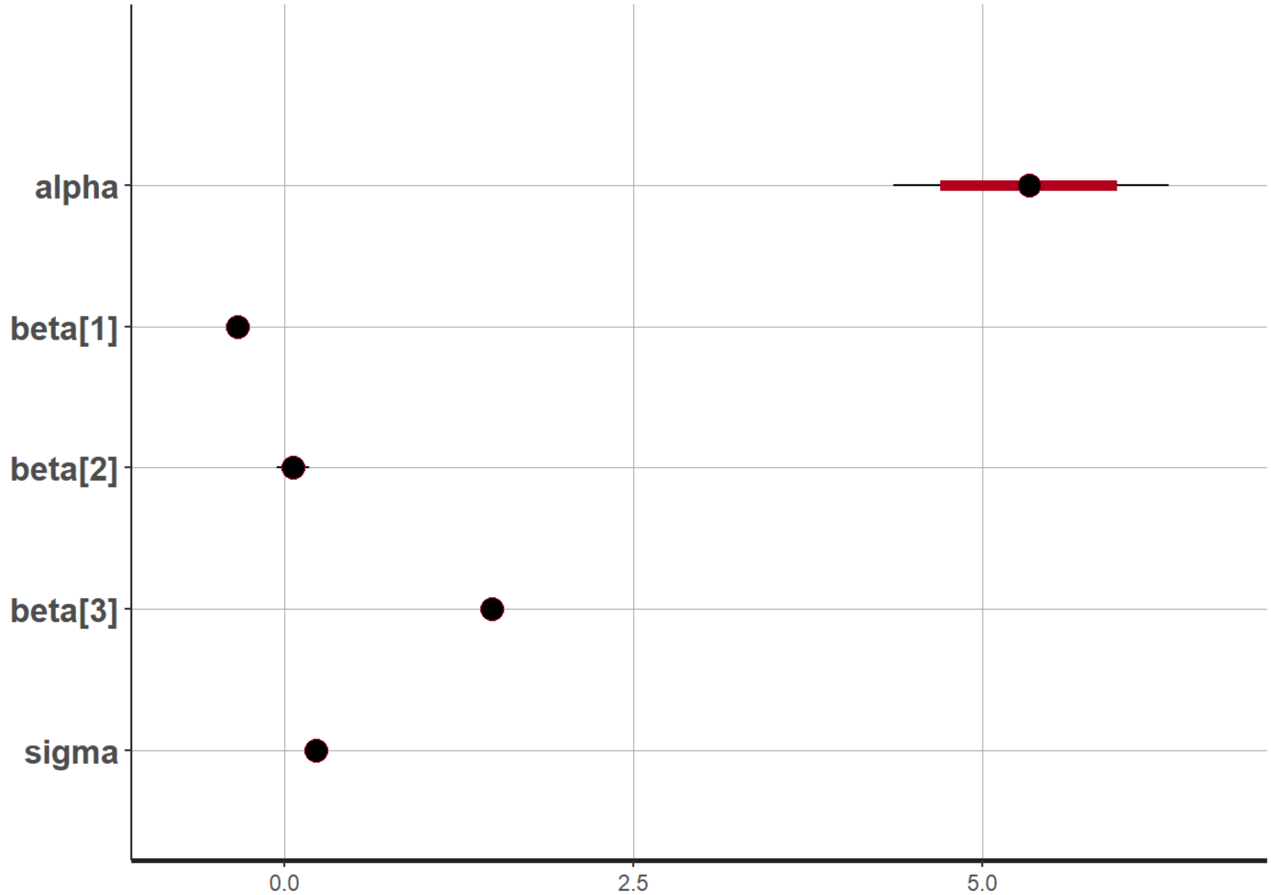
```
## Inference for Stan model: MLR_Project_Prior.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##           mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
## alpha     5.34    0.01 0.50   4.36   5.00   5.33   5.69   6.33  1264 1.00
## beta[1]  -0.33    0.00 0.03  -0.39  -0.35  -0.34  -0.32  -0.28  1464 1.00
## beta[2]   0.06    0.00 0.06  -0.06   0.02   0.06   0.10   0.17  1270 1.00
## beta[3]   1.48    0.00 0.01   1.47   1.48   1.48   1.49   1.50  2777 1.00
## sigma     0.23    0.00 0.01   0.21   0.22   0.23   0.23   0.24  2455 1.00
## lp__    593.74    0.04 1.57 589.77 592.94 594.05 594.90 595.79  1499 1.01
##
## Samples were drawn using NUTS(diag_e) at Tue Aug 20 10:57:13 2019.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

14) Include in your RMarkdown document a few posterior summaries plots (e.g. estimated posterior densities) from (13) for the parameters of interest.

```
plot(stan_run_lr_bayes) # Not always helpful if parameters on very different scales
```

```
## ci_level: 0.8 (80% intervals)
```

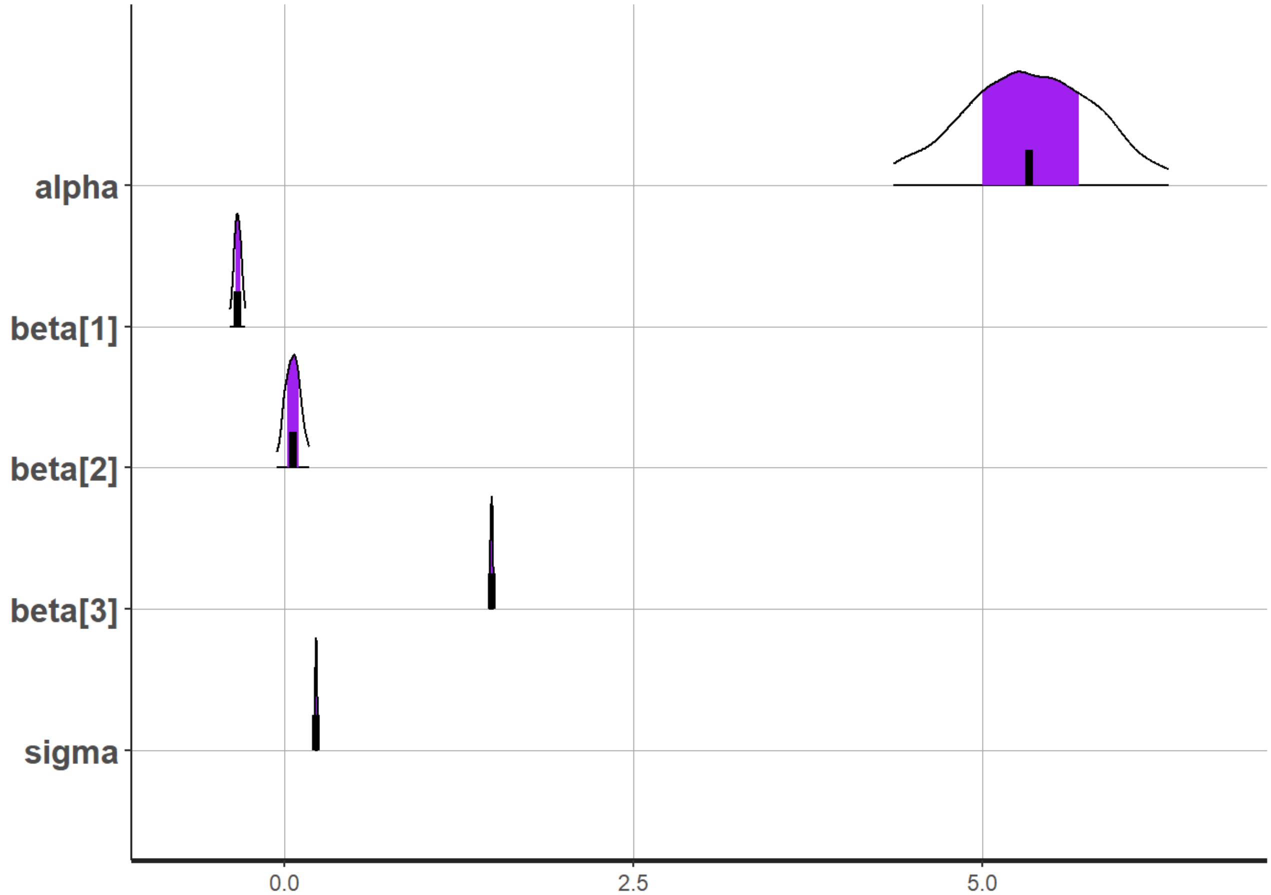
```
## outer_level: 0.95 (95% intervals)
```




```
plot(stan_run_lr_bayes, show_density = TRUE, ci_level = 0.5, fill_color = "purple")
```

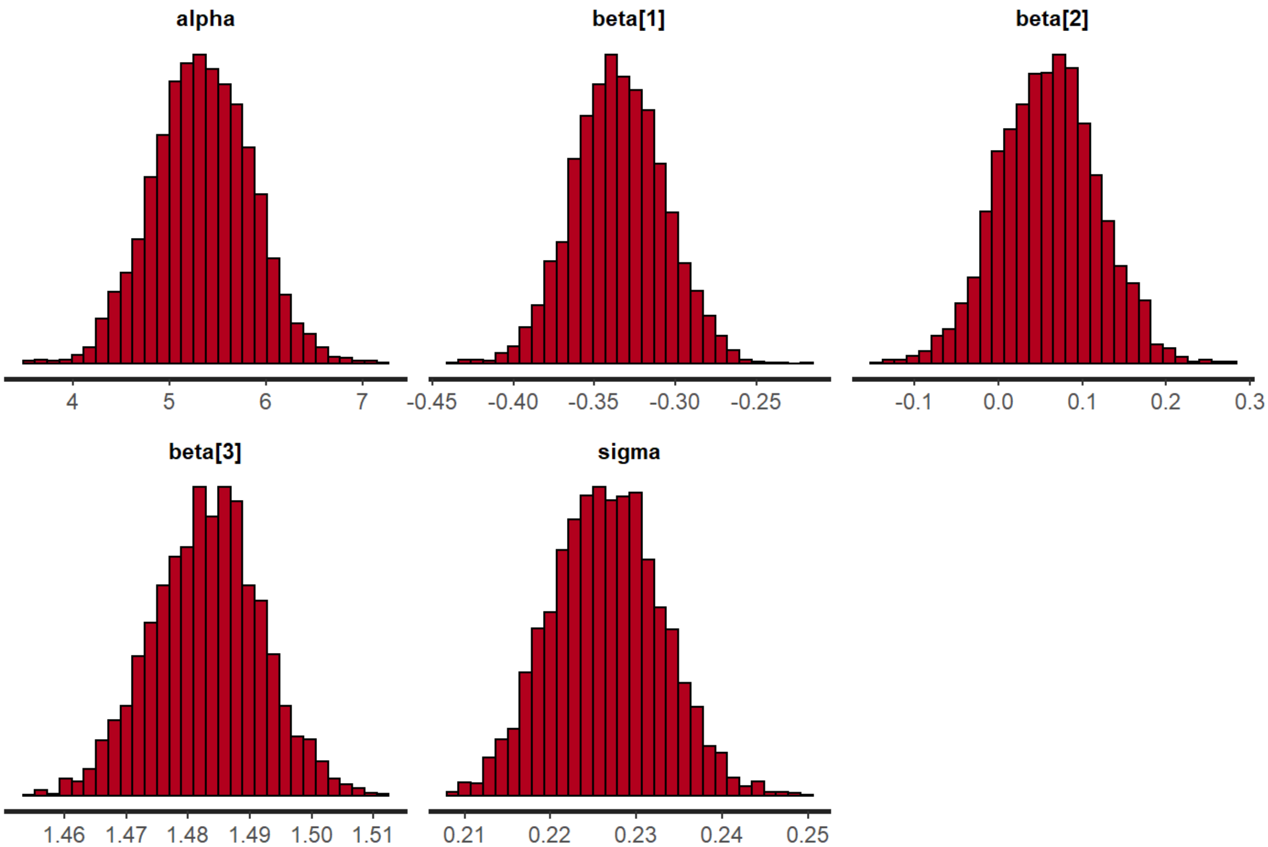
```
## ci_level: 0.5 (50% intervals)
```

```
## outer_level: 0.95 (95% intervals)
```

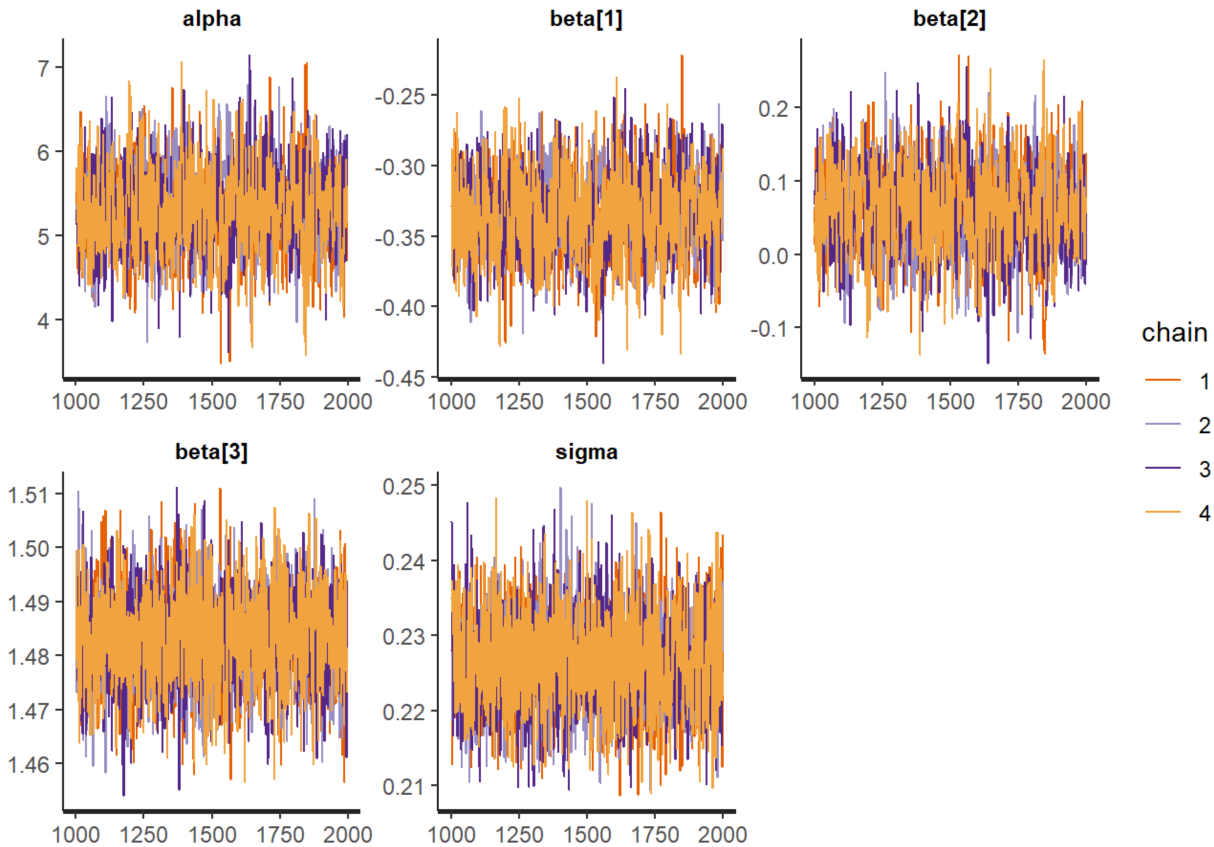



```
stan_hist(stan_run_lr_bayes)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
plot(stan_run_lr_bayes, plotfun = "trace")
```


alpha
beta[1]
beta[2]
beta[3]
sigma
chain
1
2
3
4