3.Gene_expressions_analysis - Applying random forests on field data

Fay Webster

2022-07-18

Aim:

- Applying the models established in the script: 2.Gene_expression_analysis on the field data.
- How are hybrid mice different to the parental species?

Load necessary libraries:

```
#install.packages("optimx", version = "2021-10.12") # this package is required for
#the parasite load package to work
library(tidyverse)
library(tidyr)
library(dplyr)
library(cowplot)
library(randomForest)
library(ggplot2)
library(VIM) # visualizing missing data
library(mice) # imputing missing data without predictors
library(ggpubr)
library(optimx)
library(rfUtilities) # Implements a permutation test cross-validation for
# Random Forests models
library(mice) #imputations
library(fitdistrplus) #testing distributions
library(logspline)
library(caret)
```

Field data

Import field data

```
Field <- read.csv("https://raw.githubusercontent.com/derele/Mouse_Eimeria_Field/master/data_products/SOTA_
```

Clean data

```
Field %>% summarise(length(Mouse_ID))
```

```
##   length(Mouse_ID)
## 1             1921
```

```
Field <- Field %>%
    drop_na(HI)
```

We have 1921 mice in total.

```
EqPCR.cols      <- c("delta_ct_cewe_MminusE", "MC.Eimeria", "Ct.Eimeria") #,"Ct.Mus""delta_ct_ilwe_MminusE

EimGeno.cols    <- c("n18S_Seq", "COI_Seq", "ORF470_Seq", "eimeriaSpecies")

Gene.Exp.cols   <- c("IFNy",  "CXCR3", "IL.6", #"GBP2", "IL.12", "IRG6",
                     "IL.10", "IL.13", "IL1RN",
                     "CXCR3", "CASP1", "CXCL9",
                     "IDO1", "IRGM1", "MPO", "MUC2", "MUC5AC", "MYD88",
                     "NCR1", "PRF1", "RETNLB", "SOCS1", "TICAM1", "TNF")

House.Keeping.cols <- c("GAPDH", "PPIB", "B.actin", "B-actin")
```

Prepare columns for selecting

Actual Cleaning

```
#which are the numbers of the columns of Field
names <- data.frame(colnames(Field))

#how many nas in each column
sapply(Field, function(x) sum(is.na(x)))
```

```
##                      Mouse_ID                           Sex
##                             0                             0
##                     Longitude                      Latitude
##                             0                             0
##                          Year                        mtBamH
##                             0                            47
##                         YNPAR                         X332
##                           940                            83
##                          X347                           X65
##                            59                           147
##                           Tsx                           Btk
##                           105                            41
##                         Syap1                           Es1
##                            79                           414
##                          Gpd1                          Idh1
##                           172                           167
##                           Mpi                            Np
##                           161                           168
##                          Sod1                         Es1C
##                           177                           646
##                         Gpd1C                        Idh1C
##                           412                           409
##                          MpiC                          NpC
##                           403                           408
##                         Sod1C                     HI_NLoci
##                           419                            94
##                            HI                Dissection_Date
##                             0                           289
##                        Spleen                    Aspiculuris
##                           387                           585
##                      Syphacia                Trichuris_muris
##                           585                           585
##             Taenia_taeniformis                          Zfy2
```

```
##                         1308                         1192
##                            Y             Mastophorus_muris
##                         1502                          860
##          Hymenolepis_microstoma           Catenotaenia_pusilla
##                          860                          861
##                   Cysticercus                      Address
##                         1194                          386
##                       Status                  Left_Embryo
##                         1036                         1454
##                  Right_Embryo                Worms_presence
##                         1454                         1271
##           Hymenolepis_diminiuta                 Taenia_martis
##                         1359                         1025
##       Heligmosomoides_polygurus                   Hymenolepis
##                         1359                         1194
##                       Taenia           Aspiculuris_Syphacia
##                          522                          919
##                  Heterakis_sp                   Mastophorus
##                          860                         1194
##                       counter                    Date_count
##                         1110                         1111
##                N_oocysts_sq1                N_oocysts_sq2
##                          963                          963
##                N_oocysts_sq3                N_oocysts_sq4
##                          963                          963
##                N_oocysts_sq5                N_oocysts_sq6
##                         1119                         1118
##                N_oocysts_sq7                N_oocysts_sq8
##                         1118                         1118
##                mean_neubauer                PBS_dil_in_mL
##                          963                          956
##                          OPG                       Ncells
##                          967                          954
##                       Region                   Body_Weight
##                          399                          289
##                  Body_Length          Ectoparasites_Logical
##                          298                         1340
##               Left_Epididymis                 Feces_Weight
##                         1333                          769
##                        Fleas                        Liver
##                          884                         1601
##           Right_Ovarium_Weight           Left_Ovarium_Weight
##                         1566                         1565
##         Seminal_Vesicles_Weight                  Left_Testis
##                         1253                         1322
##                  Right_Testis                  Tail_Length
##                         1321                          325
##                     Trap_Date                Ectoparasites
##                          222                         1268
##           Hymenolepis_diminuta                eimeriaSpecies
##                         1268                         1446
##                    MC.Eimeria          delta_ct_cewe_MminusE
##                         1289                         1289
##                    Ct.Eimeria                       Ct.Mus
##                         1445                          1445
##                        CASP1                        CXCL9
```

```
##                         1395                         1309
##                        CXCR3                        GAPDH
##                         1376                         1277
##                         IDO1                          IFNy
##                         1296                         1273
##                        IL.10                       IL.12A
##                         1483                         1313
##                        IL.13                        IL1RN
##                         1294                         1298
##                        IRGM1                          MPO
##                         1278                         1306
##                         MUC2                       MUC5AC
##                         1281                         1297
##                        MYD88                         NCR1
##                         1287                         1393
##                         PPIB                         PRF1
##                         1479                         1401
##                       RETNLB                        SOCS1
##                         1375                         1278
##                       TICAM1                          TNF
##                         1384                         1306
##                         IL.6                       IL.17A
##                         1365                         1352
##                          CD4                         Treg
##                         1507                         1507
##                     Div_Treg                       Treg17
##                         1507                         1507
##                          Th1                      Div_Th1
##                         1507                         1507
##                         Th17                     Div_Th17
##                         1507                         1507
##                          CD8                      Act_CD8
##                         1507                         1507
##                  Div_Act_CD8                      IFNy_CD4
##                         1507                         1507
##                   IL17A_CD4                      IFNy_CD8
##                         1507                         1507
##                     Position                        delta
##                         1507                         1445
##                    EXP_type          Oocyst_Predict_Crypto
##                         1445                          957
##              ILWE_Crypto_Ct ILWE_DNA_Content_ng.microliter
##                          957                         1318
##                         COCE                         CEWE
##                         1602                         1602
##                         ILWE                         CECE
##                         1602                         1602
##                         COWE                         SICE
##                         1602                         1602
##                        Ticks                         Host
##                         1602                         1602
##                          FEC                         SPL2
##                         1602                         1602
##                Aspiculuris_sp                  Syphacia_sp
##                          585                          585
##                     Taenia_sp                Hymenolepis_sp
```

```
##                             188                             860
##                      FEC_Eim_Ct                  MC.Eimeria.FEC
##                            1333                            1333
##                             MCs
##                            1333
```

```
f <- Field %>%
  dplyr::select(all_of(Gene.Exp.cols))

#remove rows with only nas
f <- f[rowSums(is.na(f)) != ncol(f), ]

# subset only the rows without any nas everywhere from the first data frame
Field <- Field[row.names(f), ]

# select columns to be included in imputation
# Including data on body size, adult / pregnant, and on parasites other
# than eimeria
f.1 <- Field %>% dplyr::select(c("Body_Weight", "Body_Length", "Tail_Length",
                                 "Aspiculuris_sp", "Syphacia_sp",
                                 "Trichuris_muris", "Taenia_sp",
                                 "Heterakis_sp", "Mastophorus_muris",
                                 "Hymenolepis_sp", "Catenotaenia_pusilla",
                                 all_of(Gene.Exp.cols)))


f.1  <- f.1 %>% mutate_if(is.character, as.factor)
f.1  <- f.1 %>% mutate_if(is.integer, as.numeric)
```

‘

Imputing missing data

For the lab data I have used the function rfimpute from the package random forest. I can't use the same function for our lab data as the function requires the data set to contain predictor variable and response variables.

Therefore I will be using the package MICE (multivariate Imputation by chained Equations) which only requires a data frame of missing observations.

Description: *Multiple imputation using Fully Conditional Specification (FCS)*

implemented by the MICE algorithm as described in Van Buuren and Groothuis-Oudshoorn (2011) doi:10.18637/jss .v045.i03. Each variable has its own imputation model. Built-in imputation models are provided for continuous data (predictive mean matching, normal), binary data (logistic regression), unordered categorical data (polytomous logistic regression) and ordered categorical data (proportional odds). MICE can also impute continuous two-level data (normal model, pan, second-level variables). Passive imputation can be used to maintain consistency between variables. Various diagnostic plots are available to inspect the quality of the imputations.

https://www.jstatsoft.org/article/view/v045i03

tutorial: https://www.youtube.com/watch?v=WPiYOS3qK70

https://datascienceplus.com/imputing-missing-data-with-r-mice-package/

https://datascienceplus.com/handling-missing-data-with-mice-package-a-simple-approach/

Missing data can be classified into three categories:

1. Missing completely at random (MCAR) We can't probably predict that value from any other value in the data. MCAR implies the reason for the missingness of a field is completely random, and that we probably can't predict that value from any other value in the data.

2. Missing at Random (MAR) Missingess can be explained by other values in other columns, but not from that column.

3. Missing NOT at random (MNAR) The basic MICE assumption is that the data is missing at random, and that we can make a guess about its true value by looking at other data samples.

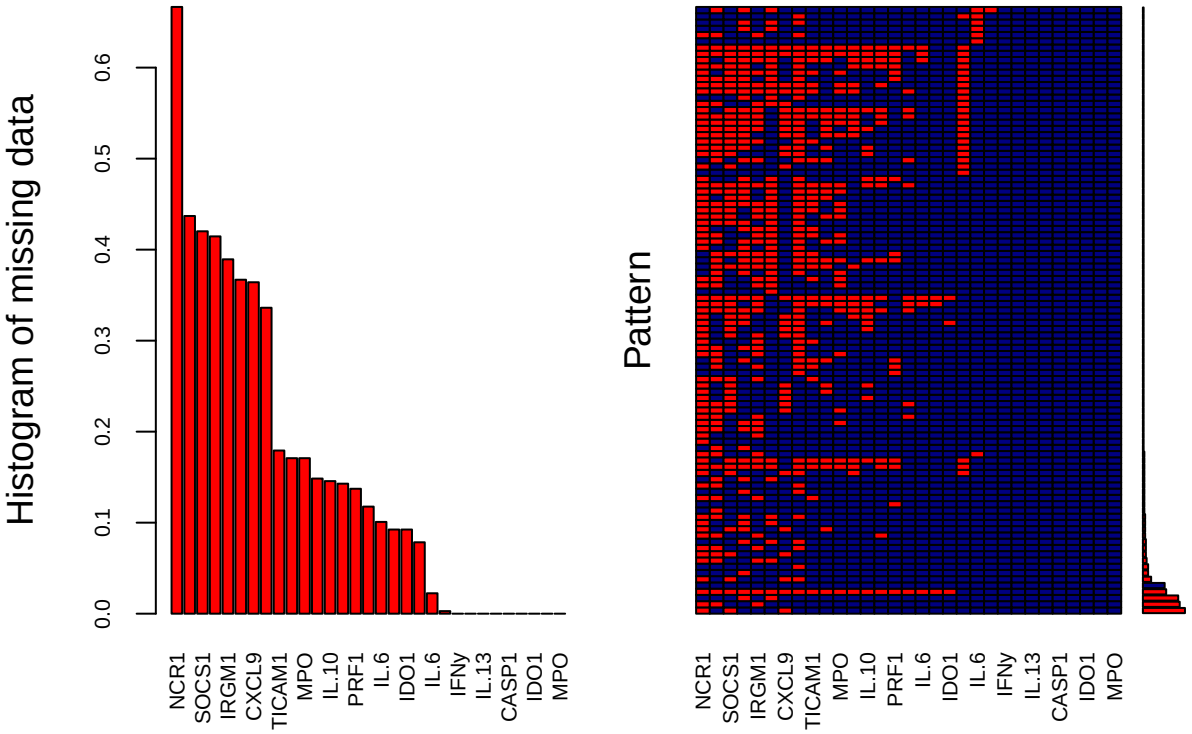
Step1 : cleaning and checking the missing data points in our field data.

```
# check the data for missing values
sapply(f.1, function(x) sum(is.na(x)))
```

```
##          Body_Weight          Body_Length          Tail_Length
##                    0                    1                    8
##       Aspiculuris_sp          Syphacia_sp      Trichuris_muris
##                    0                    0                    0
##            Taenia_sp         Heterakis_sp    Mastophorus_muris
##                    0                    0                    0
##       Hymenolepis_sp Catenotaenia_pusilla                 IFNy
##                    0                    0                   28
##                CXCR3                 IL.6                IL.10
##                  131                  120                  238
##                IL.13                IL1RN                CASP1
##                   49                   53                  150
##                CXCL9                 IDO1                IRGM1
##                   64                   51                   33
##                  MPO                 MUC2               MUC5AC
##                   61                   36                   52
##               MYD88                 NCR1                 PRF1
##                   42                  148                  156
##              RETNLB                SOCS1               TICAM1
##                  130                   33                  139
##                  TNF
##                   61
```

```
f.1 %>%
  aggr(col = c('navyblue', 'red'), numbers = TRUE, sortVars = TRUE,
       labels=names(f), cex.axis=.7, gap=3,
       ylab=c("Histogram of missing data","Pattern"))
```

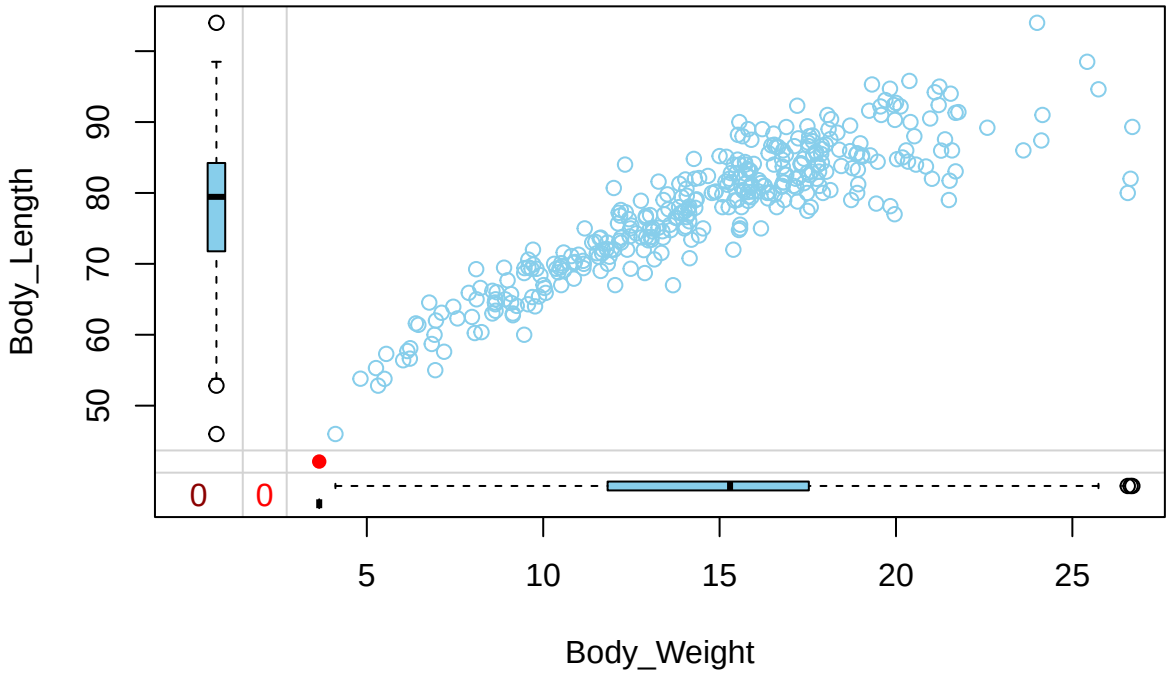
```
## Warning in plot.aggr(res, ...): not enough vertical space to display frequencies
## (too many combinations)
```




```
##
##  Variables sorted by number of missings:
##  Variable      Count
##       NCR1 0.66666667
##      CASP1 0.43697479
##      SOCS1 0.42016807
##      IL1RN 0.41456583
##      IRGM1 0.38935574
##     MUC5AC 0.36694678
##      CXCL9 0.36414566
##      MYD88 0.33613445
##     TICAM1 0.17927171
##      CXCR3 0.17086835
##        MPO 0.17086835
##     RETNLB 0.14845938
##      IL.10 0.14565826
##        TNF 0.14285714
##       PRF1 0.13725490
##      IL.13 0.11764706
##       IL.6 0.10084034
##       IFNy 0.09243697
##       IDO1 0.09243697
##       MUC2 0.07843137
##       IL.6 0.02240896
##      CXCR3 0.00280112
##       IFNy 0.00000000
##      IL.10 0.00000000
##      IL.13 0.00000000
##      IL1RN 0.00000000
##      CASP1 0.00000000
##      CXCL9 0.00000000
##       IDO1 0.00000000
##      IRGM1 0.00000000
```

```
##       MPO 0.00000000
```

```
marginplot(f.1[c(1,2)])
```



Now let's coninue by ussing the package MICE to impute the data

```
# The frequency distribution of the missing cases per variable can be obtained
# as:
init <- mice(f.1, maxit = 0)

# table of amount of variables with the amount of missing values
table(init$nmis)
```

```
##
##   0   1   8  28  33  36  42  49  51  52  53  61  64 120 130 131 139 148 150 156
##   9   1   1   1   2   1   1   1   1   1   1   2   1   1   1   1   1   1   1   1
## 238
##   1
```

```
# which method is used for imputation? In this case the package mice
# uses the default method for continuous variable,
# which is pmm, or predictive mean matching

meth <- init$method


# now impute the immune gene expression for the field and save it as the oject:
# igf
# m=5 refers to the number of imputed datasets. Five is the default value.
igf <- mice(f.1, method = meth, m = 5, seed = 500)
```

```
##
##  iter imp variable
##    1   1  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##    1   2  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##    1   3  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
```

```
##   1   4  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   1   5  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   2   1  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   2   2  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   2   3  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   2   4  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   2   5  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   3   1  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   3   2  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   3   3  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   3   4  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   3   5  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   4   1  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   4   2  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   4   3  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   4   4  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   4   5  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   5   1  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   5   2  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   5   3  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   5   4  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
##   5   5  Body_Length  Tail_Length  IFNy  CXCR3  IL.6  IL.10  IL.13  IL1RN  CASP1  CXCL9  IDO1  IRGM1  M
```

```
summary(igf)
```

```
## Class: mids
## Number of multiple imputations:  5
## Imputation methods:
##          Body_Weight          Body_Length          Tail_Length
##                   ""                "pmm"                "pmm"
##       Aspiculuris_sp          Syphacia_sp      Trichuris_muris
##                   ""                  ""                   ""
##            Taenia_sp         Heterakis_sp    Mastophorus_muris
##                   ""                  ""                   ""
##       Hymenolepis_sp Catenotaenia_pusilla                 IFNy
##                   ""                  ""                "pmm"
##                CXCR3                IL.6                IL.10
##                "pmm"               "pmm"                "pmm"
##                IL.13               IL1RN                CASP1
##                "pmm"               "pmm"                "pmm"
##                CXCL9                IDO1                IRGM1
##                "pmm"               "pmm"                "pmm"
##                  MPO                MUC2               MUC5AC
##                "pmm"               "pmm"                "pmm"
##                MYD88                NCR1                 PRF1
##                "pmm"               "pmm"                "pmm"
##               RETNLB               SOCS1               TICAM1
##                "pmm"               "pmm"                "pmm"
##                  TNF
##                "pmm"
## PredictorMatrix:
##                Body_Weight Body_Length Tail_Length Aspiculuris_sp Syphacia_sp
## Body_Weight              0           1           1              1           1
## Body_Length              1           0           1              1           1
## Tail_Length              1           1           0              1           1
## Aspiculuris_sp           1           1           1              0           1
## Syphacia_sp              1           1           1              1           0
```

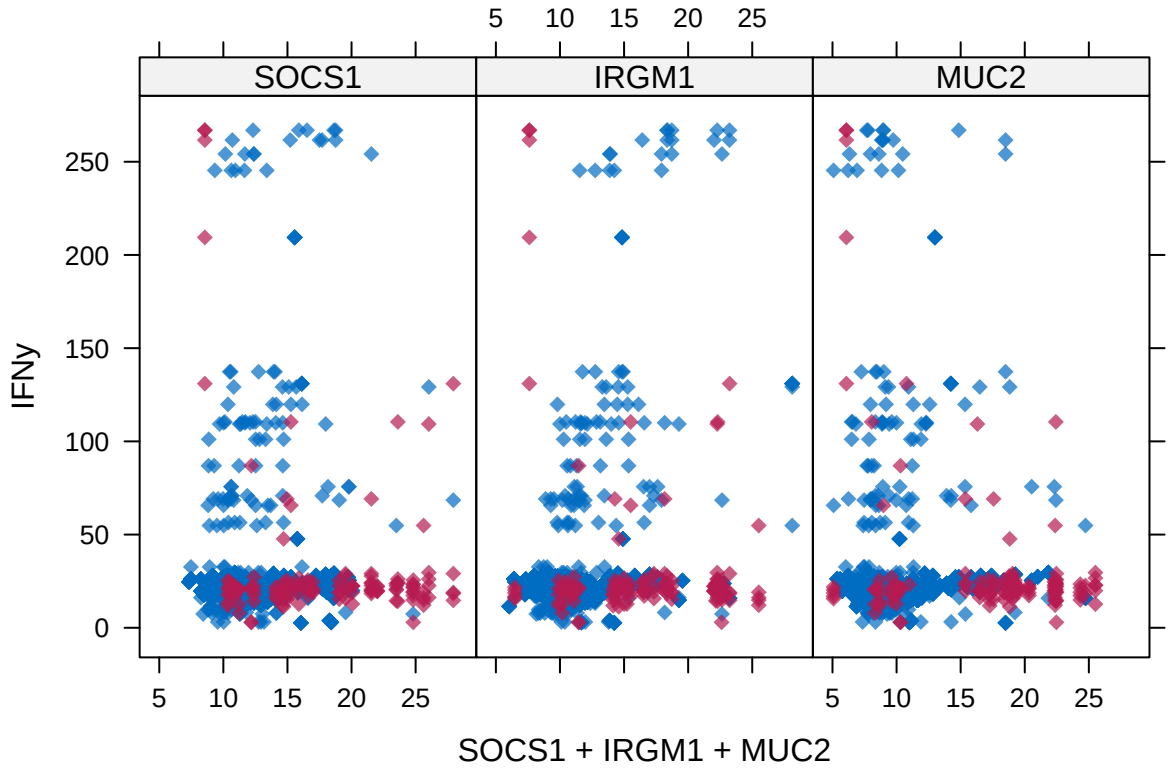
```
## Trichuris_muris           1           1           1              1           1
##                 Trichuris_muris Taenia_sp Heterakis_sp Mastophorus_muris
## Body_Weight                   1         1            1                 1
## Body_Length                   1         1            1                 1
## Tail_Length                   1         1            1                 1
## Aspiculuris_sp                1         1            1                 1
## Syphacia_sp                   1         1            1                 1
## Trichuris_muris               0         1            1                 1
##                 Hymenolepis_sp Catenotaenia_pusilla IFNy CXCR3 IL.6 IL.10 IL.13
## Body_Weight                  1                     1    1     1    1     1     1
## Body_Length                  1                     1    1     1    1     1     1
## Tail_Length                  1                     1    1     1    1     1     1
## Aspiculuris_sp               1                     1    1     1    1     1     1
## Syphacia_sp                  1                     1    1     1    1     1     1
## Trichuris_muris              1                     1    1     1    1     1     1
##                 IL1RN CASP1 CXCL9 IDO1 IRGM1 MPO MUC2 MUC5AC MYD88 NCR1 PRF1
## Body_Weight         1     1     1    1     1   1    1      1     1    1    1
## Body_Length         1     1     1    1     1   1    1      1     1    1    1
## Tail_Length         1     1     1    1     1   1    1      1     1    1    1
## Aspiculuris_sp      1     1     1    1     1   1    1      1     1    1    1
## Syphacia_sp         1     1     1    1     1   1    1      1     1    1    1
## Trichuris_muris     1     1     1    1     1   1    1      1     1    1    1
##                 RETNLB SOCS1 TICAM1 TNF
## Body_Weight          1     1      1   1
## Body_Length          1     1      1   1
## Tail_Length          1     1      1   1
## Aspiculuris_sp       1     1      1   1
## Syphacia_sp          1     1      1   1
## Trichuris_muris      1     1      1   1
```

```
# to check each column with imputed data
## igf$imp$IFNy

#Now we can get back the completed dataset using the complete()
completeField <- complete(igf, 1)
```

Predictive mean matching with $d = 5$ is the default in mice() for continuous data. The method is robust against misspecification of the imputation model, yet performs as well as theoretically superior methods. In the context of missing covariate data, Marshall, Altman, and Holder (2010) concluded that predictive mean matching "produced the least biased estimates and better model performance measures." Another simulation study that addressed skewed data concluded that predictive mean matching "may be the preferred approach provided that less than 50% of the cases have missing data and the missing data are not MNAR" (Marshall et al. 2010). Kleinke (2017) found that the method works well across a wide variety of scenarios, but warned the default cannot address severe skewness or small samples.

Let's compare the distributions of original and imputed data using a some useful plots.First of all we can use a scatterplot and plot Ozone against all the other variables. Let's first plot the variables for which we have few missing values.

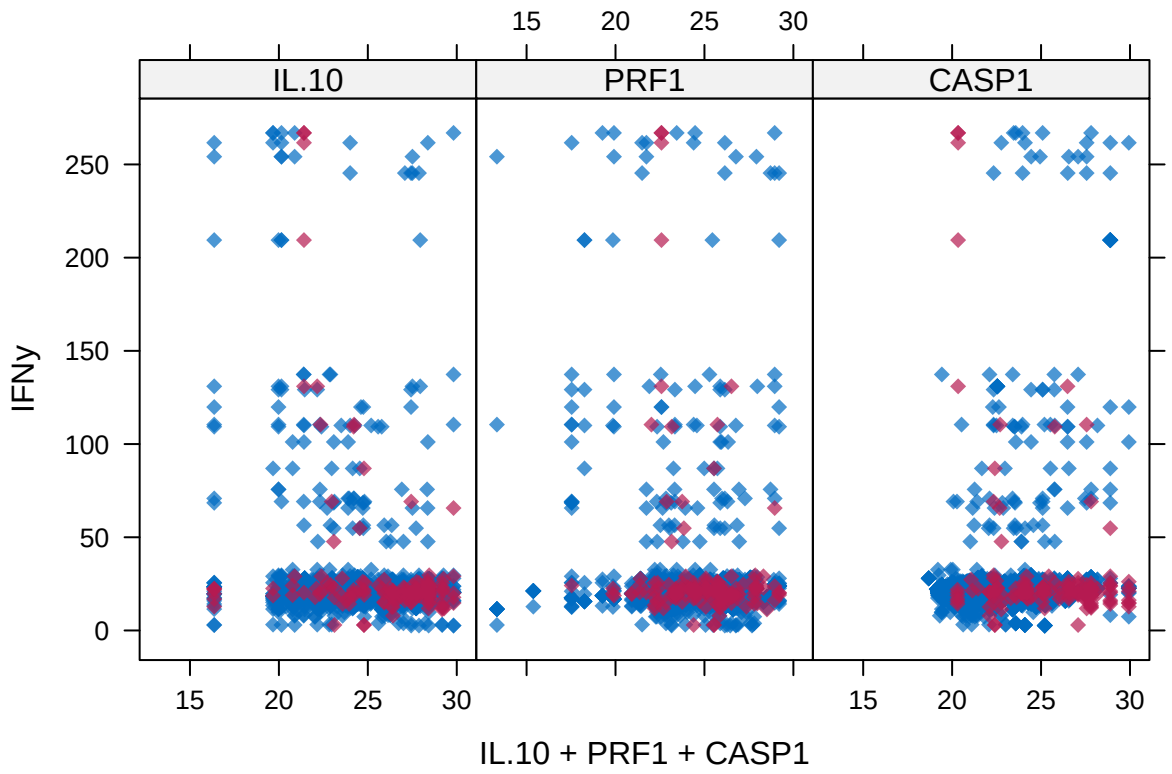
```
xyplot(igf,IFNy ~ SOCS1 + IRGM1 + MUC2, pch=18,cex=1)
```



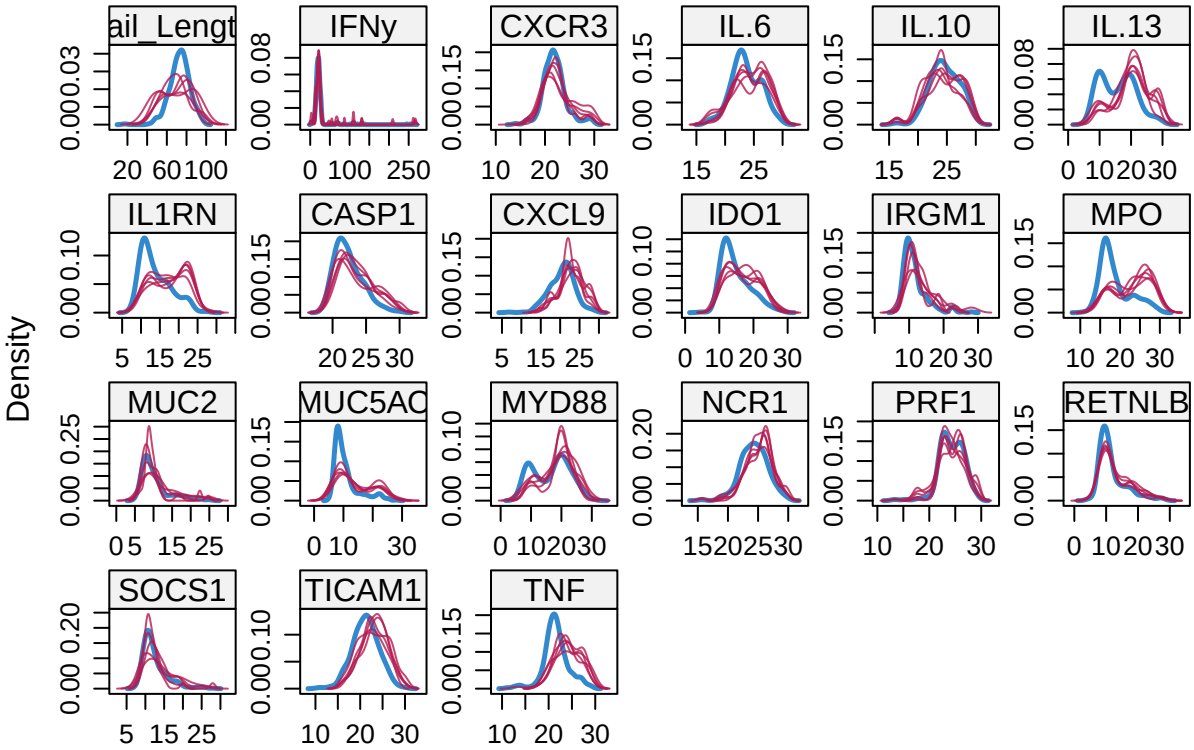
What we would like to see is that the shape of the magenta points (imputed) matches the shape of the blue ones (observed). The matching shape tells us that the imputed values are indeed "plausible values".

Now let's plot the variables with many missing data points.

```
xyplot(igf,IFNy ~ IL.10 + PRF1 + CASP1, pch=18,cex=1)
```



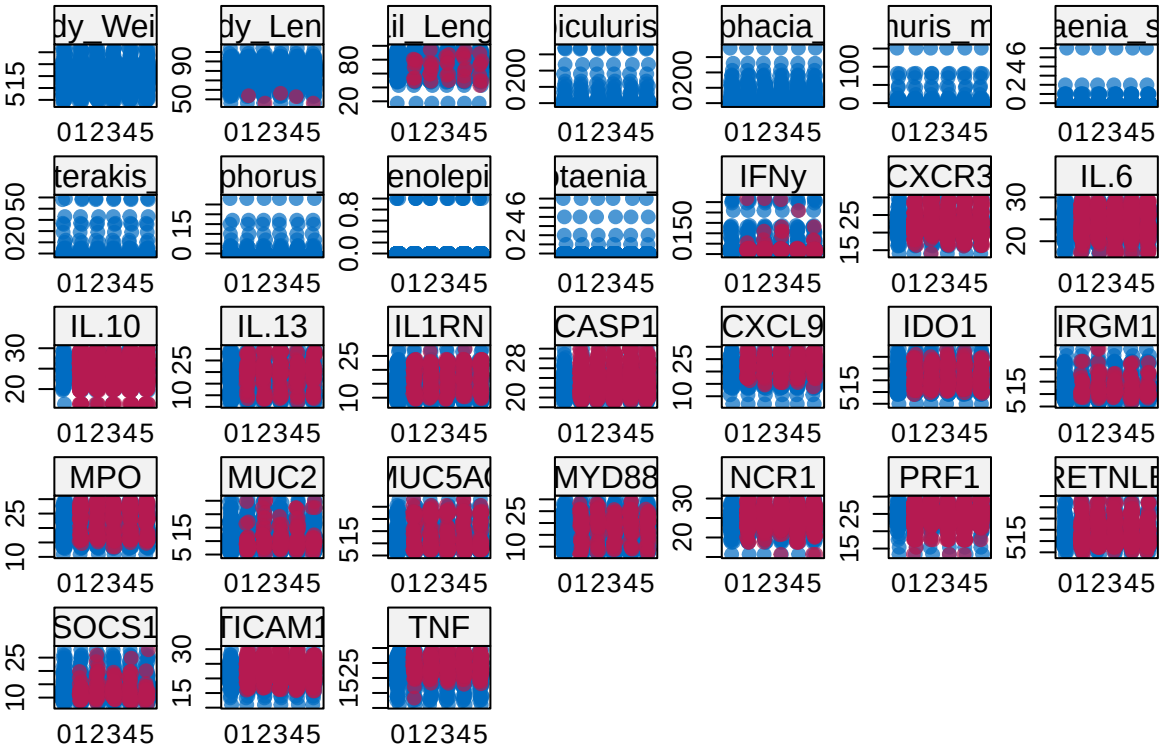
```
densityplot(igf)
```



The density of the imputed data for each imputed dataset is showed in magenta while the density of the observed data is showed in blue. Again, under our previous assumptions we expect the distributions to be similar.

Another useful visual take on the distributions can be obtained using the stripplot() function that shows the distributions of the variables as individual points

```
stripplot(igf, pch = 20, cex = 1.2)
```



Predicting weight loss in our imputed field data

Start by making the predictions for the field data.

```
# Start by selecting the columns that appear in both the training data set and
# the field data set
completeField <- completeField %>%
  dplyr::select(intersect(colnames(completeField), c("IFNy", "IL.6", "IL.10",
                                                     "IL.13", "IL1RN", "CASP1",
                                                     "CXCL9", "IDO1", "IRGM1",
                                                     "MPO", "MUC2", "MUC5AC",
                                                     "MYD88", "NCR1", "PRF1",
                                                     "RETNLB", "SOCS1", "TICAM1",
                                                     "TNF")))


# load predicting weight loss model
load("r_scripts/models/predict_weight_loss.RData")

set.seed(540)


#The predict() function in R is used to predict the values based on the input data.
predictions_field <- predict(weight_loss_predict, completeField)

# assign test.data to a new object, so that we can make changes
result_field <- completeField

#add the new variable of predictions to the result object
result_field <- cbind(result_field, predictions_field)

# add it to the field data
Field <- cbind(Field, predictions_field)
```

It is time to apply the package of Alice Balard et al. on our predictions!

Let's see if we indeed have differences across the hybrid index with our predicted weight loss.

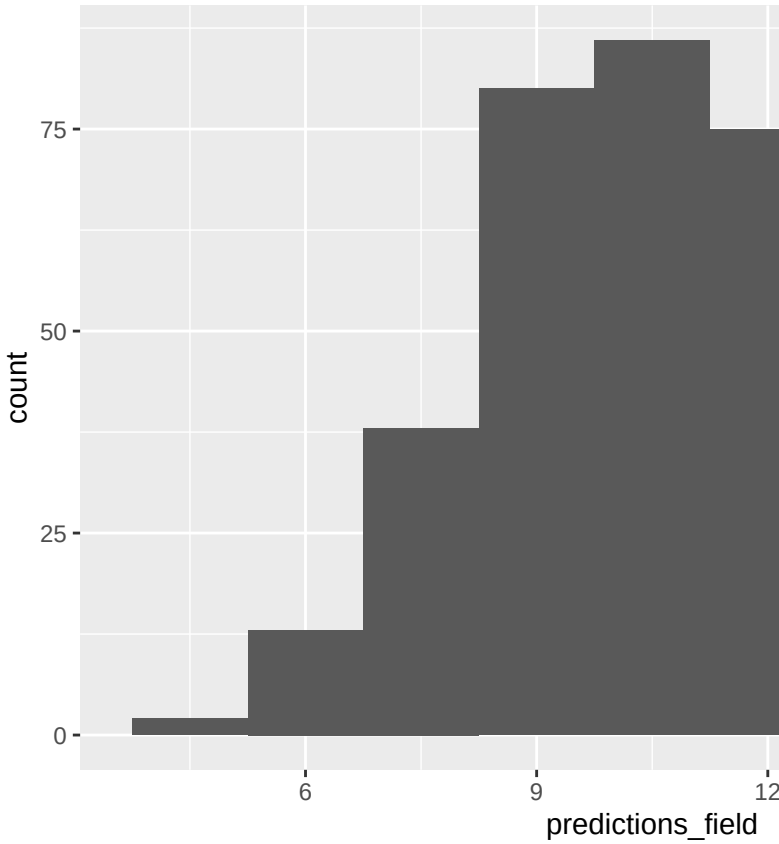
Install the package

```
##
## * checking for file '/tmp/Rtmp5mKu4R/remotes16deeb112fa30f/alicebalard-parasiteLoad-1b43216/DESCRIPTION
## * preparing 'parasiteLoad':
## * checking DESCRIPTION meta-information ... OK
## * checking for LF line-endings in source and make files and shell scripts
## * checking for empty or unneeded directories
## * building 'parasiteLoad_0.1.0.tar.gz'
```

Data diagnostics

Visualizations

```
Field %>% ggplot(aes(x = predictions_field)) +
  geom_histogram(binwidth = 1.5)
```

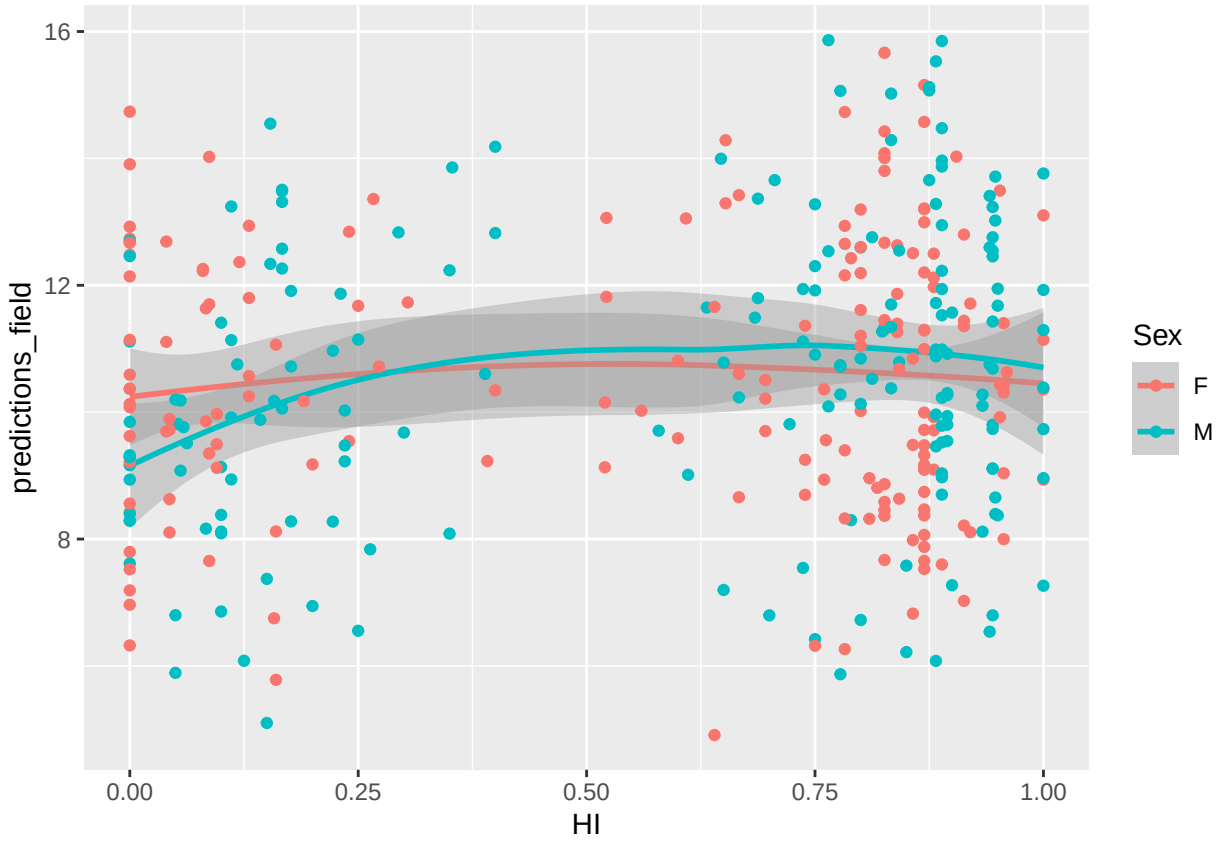


What is the distribution of the predicted weight loss?

Rough graph of our predictions against the hybrid index and against the #### body length

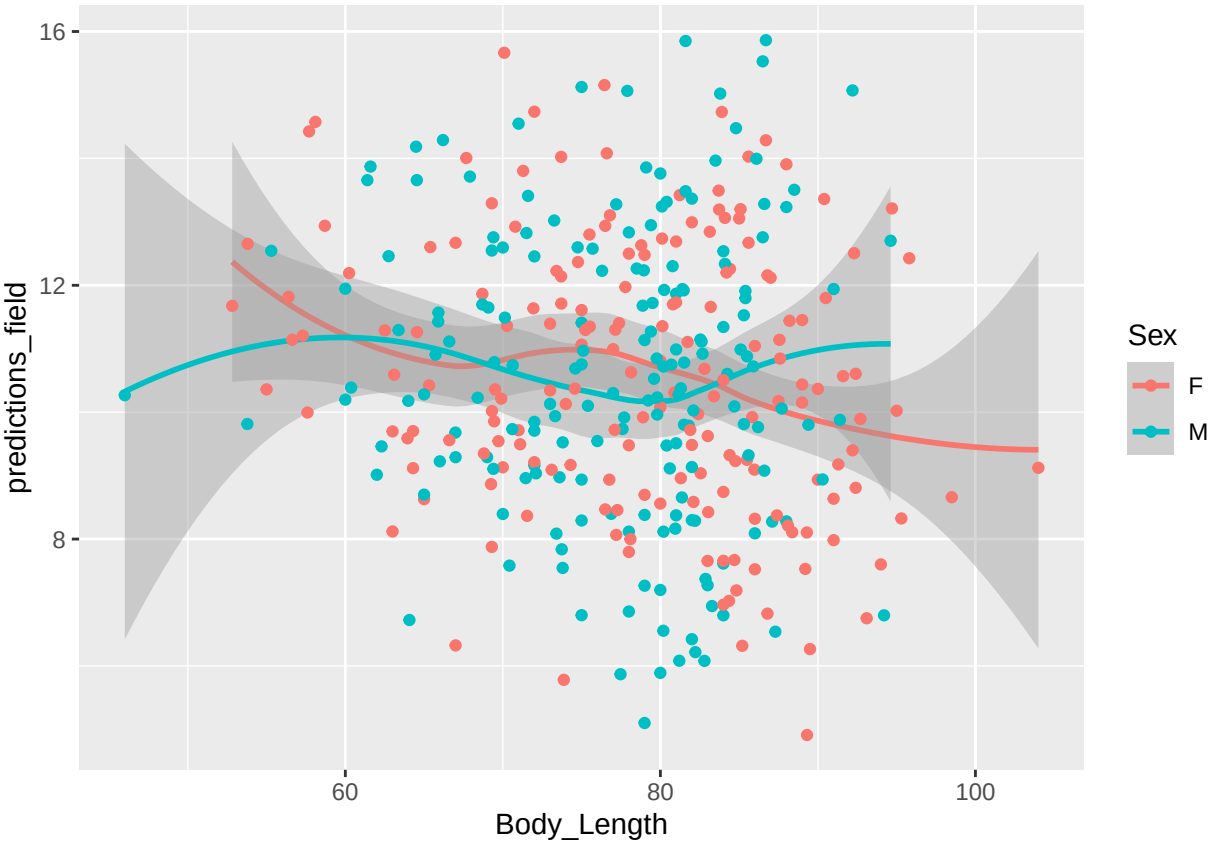
```
Field %>%
    ggplot(aes(x = HI , y = predictions_field , color = Sex)) +
    geom_smooth() +
    geom_point()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
Field %>%
    ggplot(aes(x = Body_Length , y = predictions_field , color = Sex)) +
    geom_smooth() +
    geom_point()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## Warning: Removed 1 rows containing non-finite values (stat_smooth).
## Warning: Removed 1 rows containing missing values (geom_point).
```

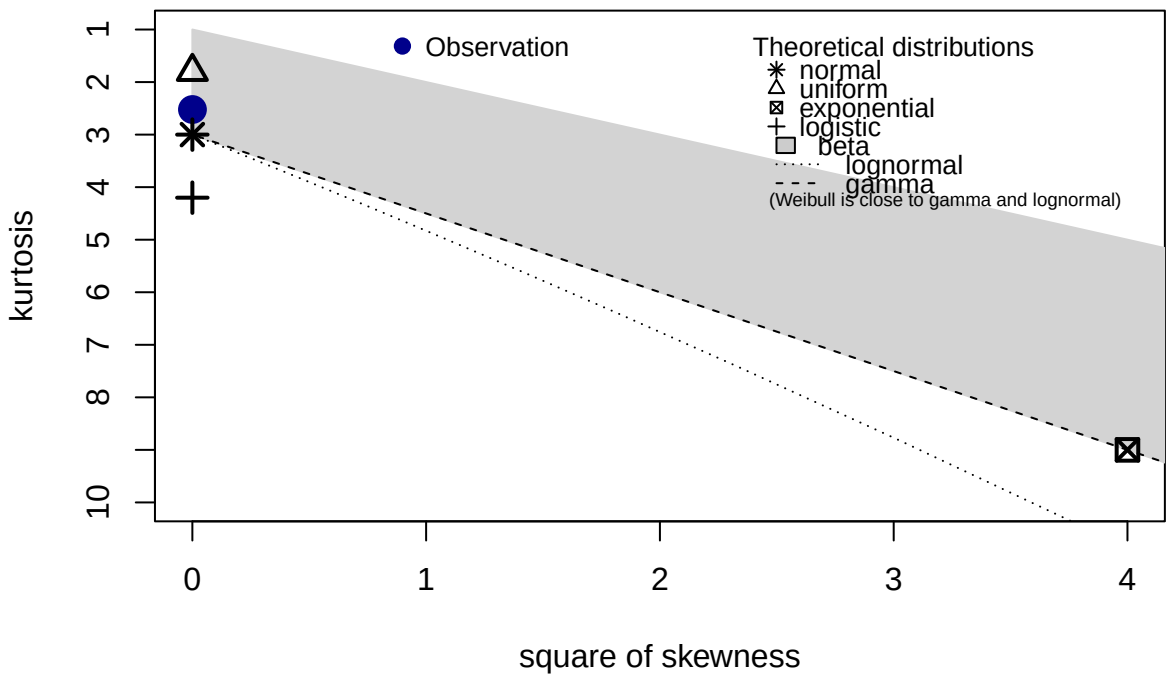


Fitting distributions??

Ratios / Percentages are not normally distributed. Weibull is a good distributions.

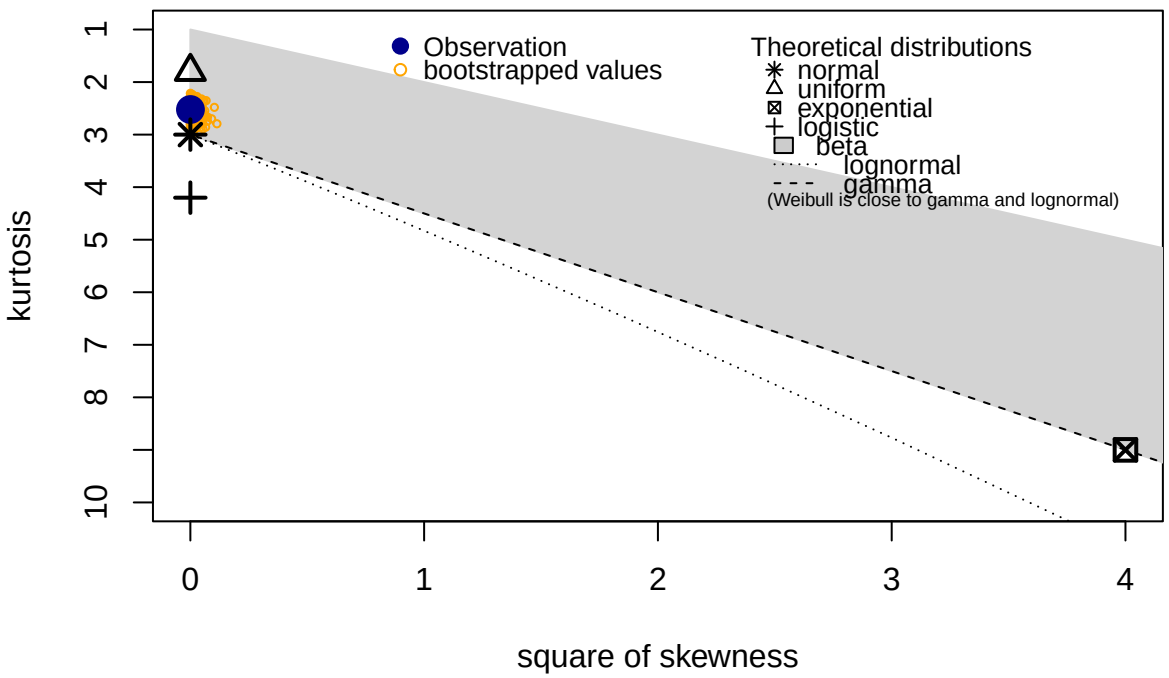
Alice used weibull for the qpcr data. (paper)

```
Field <- Field %>%
dplyr::mutate(WL = predictions_field)

x <- Field$WL

descdist(data = x, discrete = FALSE)
```




```
## summary statistics
## ------
## min:  4.911371   max:  15.86317
## median:  10.37672
## mean:  10.54954
## estimated sd:  2.217851
## estimated skewness:  0.01808838
## estimated kurtosis:  2.522255
```

```
descdist(data = x, discrete = FALSE, #data is continuous
         boot = 1000)
```



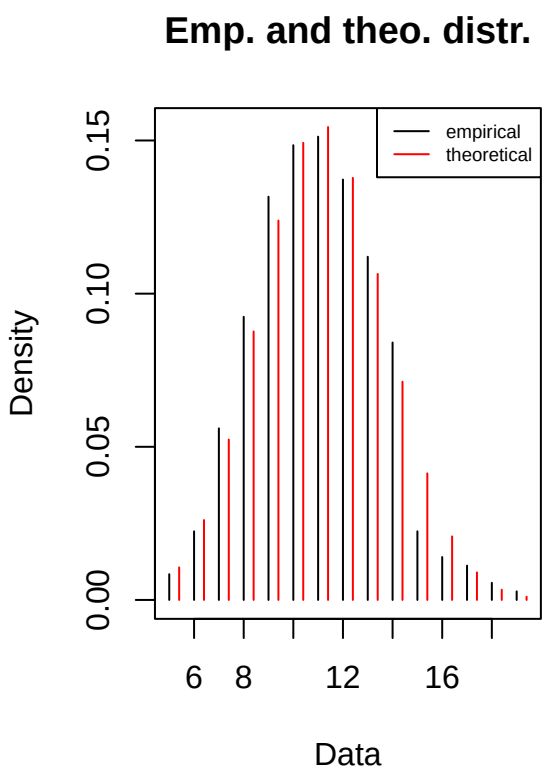

```
## summary statistics
## ------
## min:  4.911371   max:  15.86317
## median:  10.37672
## mean:  10.54954
## estimated sd:  2.217851
## estimated skewness:  0.01808838
## estimated kurtosis:  2.522255
```

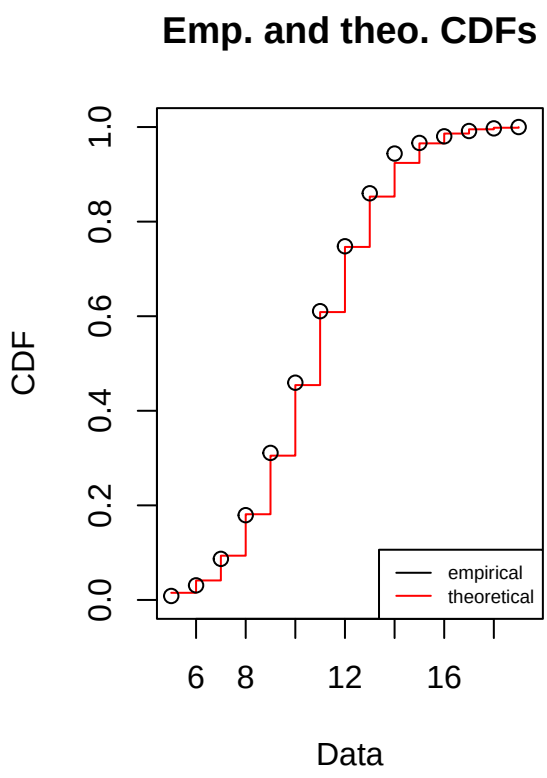
Test for binomial distribution

```
set.seed(10)
n = 25
size = 27
prob = .4
data = rbinom(x, size = size, prob = prob)
fit = fitdist(data = data, dist="binom",
                   fix.arg=list(size = size),
                   start=list(prob = 0.1))

summary(fit)
```

```
## Fitting of the distribution ' binom ' by maximum likelihood
## Parameters :
##       estimate  Std. Error
## prob 0.4009753 0.004991868
## Fixed parameters:
##      value
## size    27
## Loglikelihood:  -829.9393   AIC:  1661.879   BIC:  1665.756
```

```
plot(fit)
```






```
normal_ <- fitdist(x, "norm")
weibull_ <- fitdist(x, "weibull")
gamma_ <- fitdist(x, "gamma")


# Define function to be used to test, get the log lik and aic
tryDistrib <- function(x, distrib){
  # deals with fitdistr error:
  fit <- tryCatch(MASS::fitdistr(x, distrib), error=function(err) "fit failed")
  return(list(fit = fit,
              loglik = tryCatch(fit$loglik, error=function(err) "no loglik computed"),
              AIC = tryCatch(fit$aic, error=function(err) "no aic computed")))
}



findGoodDist <- function(x, distribs, distribs2){
  l =lapply(distribs, function(i) tryDistrib(x, i))
  names(l) <- distribs
  print(l)
  listDistr <- lapply(distribs2, function(i){
    if (i %in% "t"){
      fitdistrplus::fitdist(x, i, start = list(df =2))
    } else {
      fitdistrplus::fitdist(x,i)
    }}
  )
  par(mfrow=c(2,2))
```

```
  denscomp(listDistr, legendtext=distribs2)
  cdfcomp(listDistr, legendtext=distribs2)
  qqcomp(listDistr, legendtext=distribs2)
  ppcomp(listDistr, legendtext=distribs2)
  par(mfrow=c(1,1))
}
```

```
tryDistrib(x, "normal")
```

Functions for testing distributions

```
## $fit
##       mean           sd
##   10.54953557    2.21474293
##  ( 0.11721663) ( 0.08288467)
##
## $loglik
## [1] -790.4247
##
## $AIC
## NULL
```

```
tryDistrib(x, "binomial")
```

```
## $fit
## [1] "fit failed"
##
## $loglik
## [1] "no loglik computed"
##
## $AIC
## [1] "no aic computed"
```

```
tryDistrib(x, "student")
```

```
## $fit
## [1] "fit failed"
##
## $loglik
## [1] "no loglik computed"
##
## $AIC
## [1] "no aic computed"
```

```
tryDistrib(x, "weibull")
```

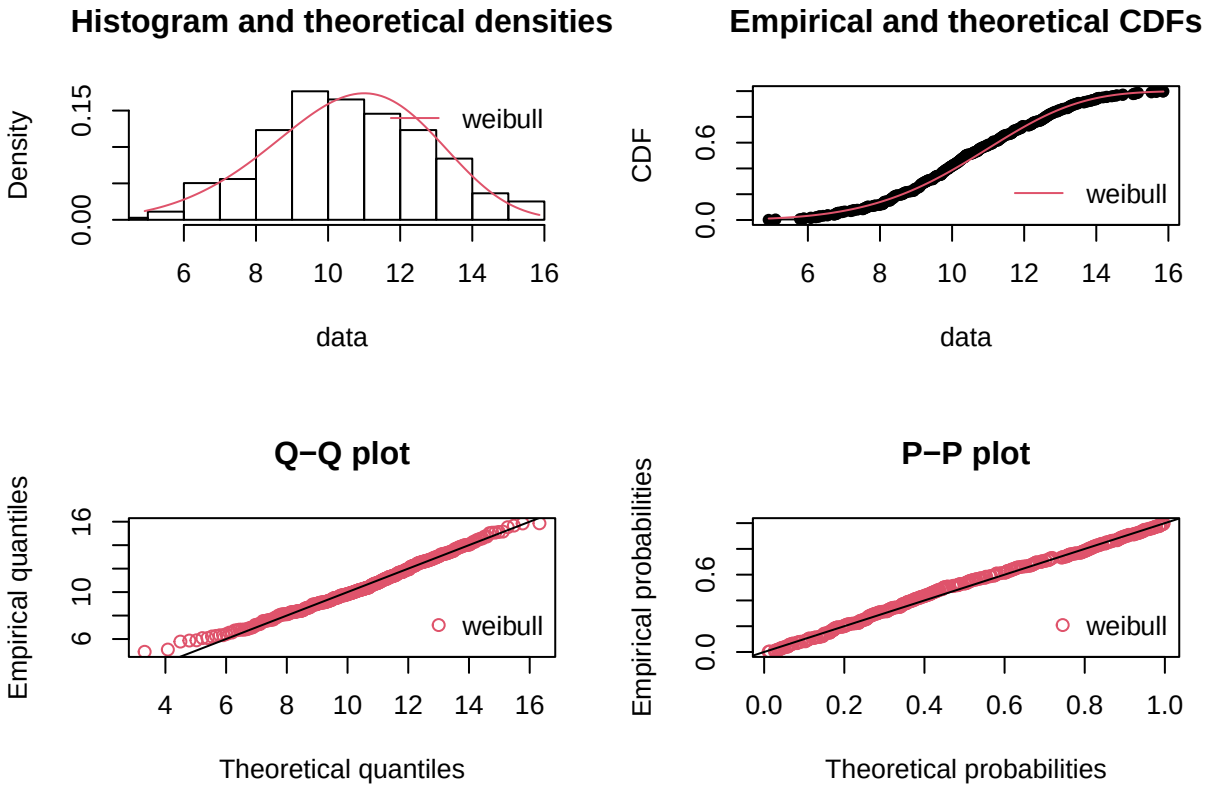
```
## $fit
##      shape        scale
##    5.3033464   11.4449547
##  ( 0.2154724) ( 0.1206725)
##
## $loglik
## [1] -792.6391
##
## $AIC
## NULL
```

```
tryDistrib(x, "weibullshifted")
```

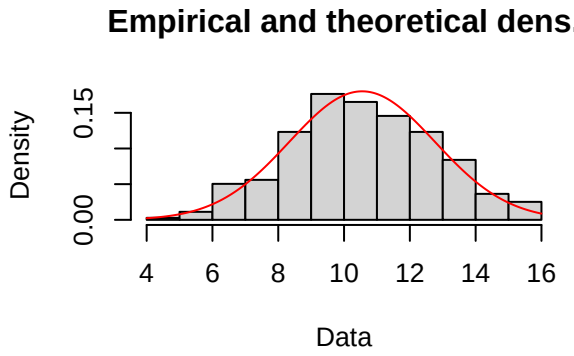
```
## $fit
## [1] "fit failed"
##
## $loglik
## [1] "no loglik computed"
##
## $AIC
## [1] "no aic computed"
```

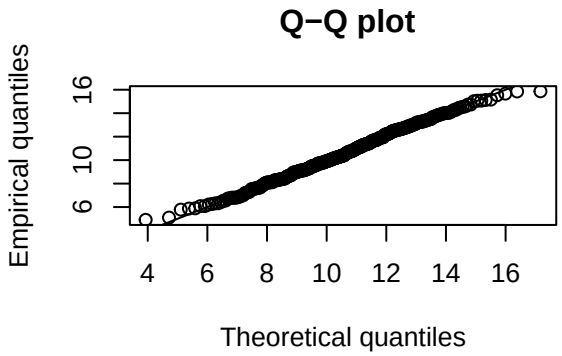
```
findGoodDist(x, "normal", "weibull")
```

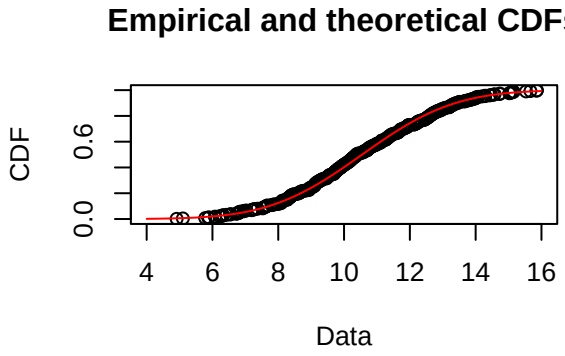
```
## $normal
## $normal$fit
##       mean           sd
##   10.54953557    2.21474293
##  ( 0.11721663) ( 0.08288467)
##
## $normal$loglik
## [1] -790.4247
##
## $normal$AIC
## NULL
```

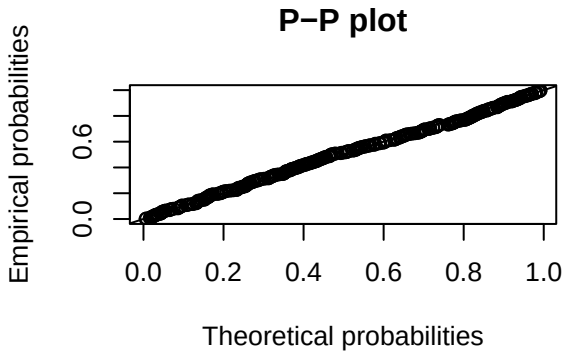



```
plot(normal_)
```





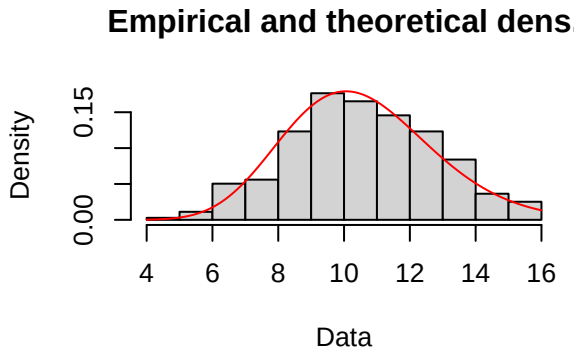


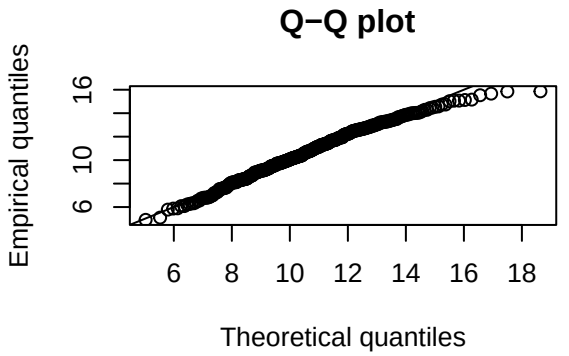


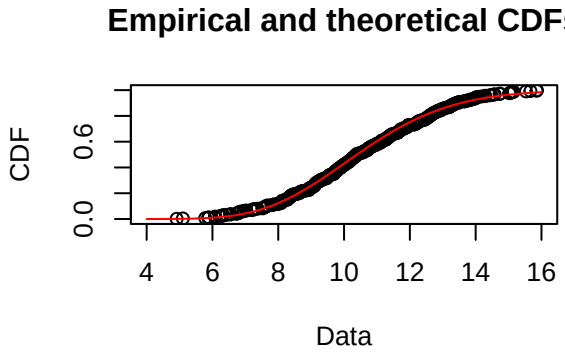

```
summary(normal_)
```

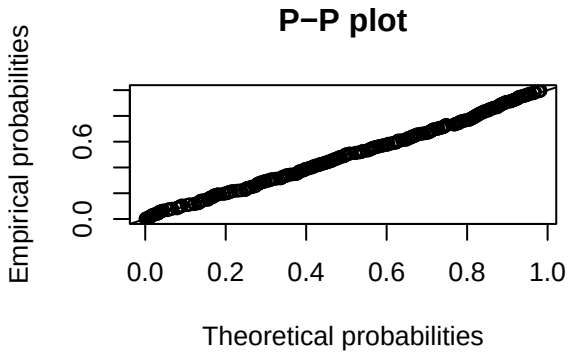
```
## Fitting of the distribution ' norm ' by maximum likelihood
## Parameters :
##       estimate Std. Error
## mean 10.549536 0.11721663
## sd    2.214743 0.08288459
## Loglikelihood:  -790.4247   AIC:  1584.849   BIC:  1592.605
## Correlation matrix:
##      mean sd
## mean    1  0
## sd      0  1
```

```
plot(gamma_)
```





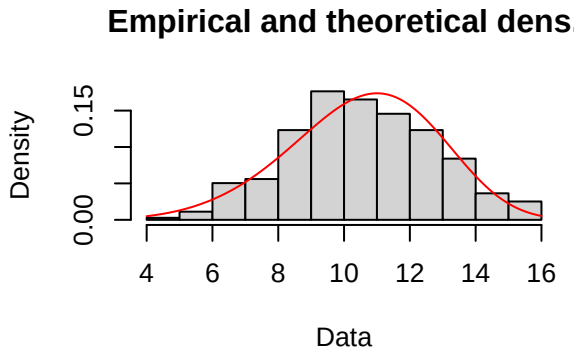


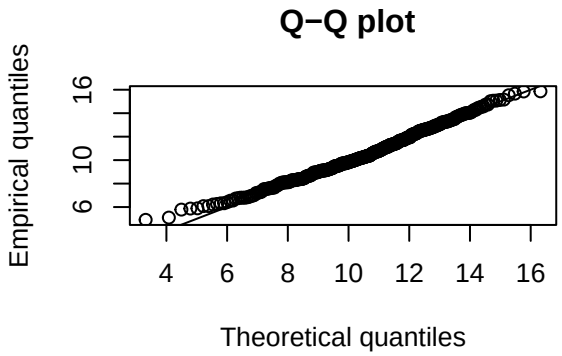


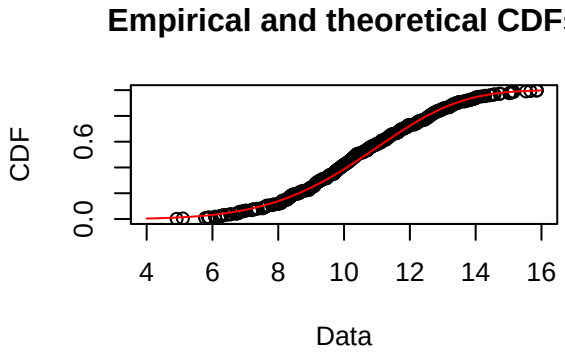

```
summary(gamma_)
```

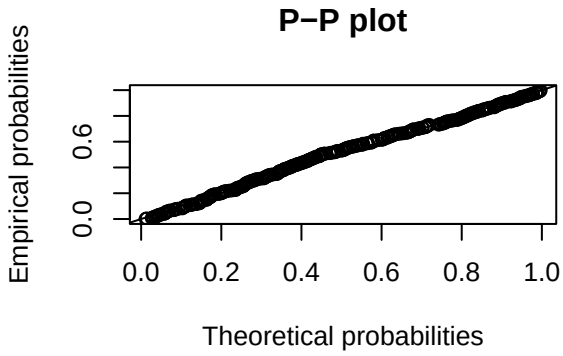
```
## Fitting of the distribution ' gamma ' by maximum likelihood
## Parameters :
##        estimate Std. Error
## shape 21.575010  1.6025112
## rate   2.045179  0.1536853
## Loglikelihood:  -793.8075   AIC:  1591.615   BIC:  1599.371
## Correlation matrix:
##           shape      rate
## shape 1.0000000 0.9884371
## rate  0.9884371 1.0000000
```

```
plot(weibull_)
```










```
summary(weibull_)
```

```
## Fitting of the distribution ' weibull ' by maximum likelihood
## Parameters :
##        estimate Std. Error
## shape  5.303975  0.2154915
## scale 11.444974  0.1206599
## Loglikelihood:  -792.6391   AIC:  1589.278   BIC:  1597.034
## Correlation matrix:
##           shape     scale
## shape 1.0000000 0.3228038
## scale 0.3228038 1.0000000
```

We have a weibull distribution!

Is alpha significant for each hypothesis?

```
Field$Sex <- as.factor(Field$Sex)



parasiteLoad::getParamBounds("weibull", data = Field, response = "WL")
```

```
##      L1start         L1LB         L1UB      L2start         L2LB         L2UB
## 10.549535572  0.000000001 15.863165435 10.549535572  0.000000001 15.863165435
##   alphaStart      alphaLB      alphaUB myshapeStart    myshapeLB    myshapeUB
##  0.000000000 -5.000000000  5.000000000  1.000000000  0.000000001  5.000000000
```

```
speparam <- c(L1start = 10,
                      L1LB = 1e-9,
                      L1UB = 20,
                      L2start = 10,
                      L2LB = 1e-9,
```

```
                    L2UB = 20,
                    alphaStart = 0, alphaLB = -5, alphaUB = 5,
                    myshapeStart = 1, myshapeLB = 1e-9, myshapeUB = 5)

##All
fitWL_Sex <- parasiteLoad::analyse(data = Field,
                        response = "WL",
                        model = "weibull",
                        group = "Sex")
```

```
## [1] "Analysing data for response: WL"
## [1] "Fit for the response: WL"
## [1] "Fitting for all"
## [1] "Fitting model basic without alpha"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable
```

```
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable
```

```
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable
```

```
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable
```

```
## [1] "Did converge"
## [1] "Fitting for groupA : F"
## [1] "Fitting model basic without alpha"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable
```

```
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable
```

```
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable
```

```
## [1] "Did converge"
```

```
## [1] "Fitting model advanced with alpha"

## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable

## [1] "Did converge"
## [1] "Fitting for groupB : M"
## [1] "Fitting model basic without alpha"

## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable

## [1] "Did converge"
## [1] "Fitting model basic with alpha"

## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable

## [1] "Did converge"
## [1] "Fitting model advanced without alpha"

## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable

## [1] "Did converge"
## [1] "Fitting model advanced with alpha"

## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable

## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.99   1 0.04590541
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 1.15   1 0.1298102
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.29   1 0.4434102
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF     pvalue
## 1 2.19   1 0.03625053
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.13   1 0.6074602
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF     pvalue
## 1 2.58   1 0.02311645
## [1] "Testing H1 vs H0"
##    dLL dDF     pvalue
## 1 2.84   1 0.01718919
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.75   3 0.6831234
## [1] "Testing H3 vs H1"
```

```
##    dLL dDF    pvalue
## 1 3.24   4 0.1666385
## [1] "Testing H3 vs H2"
##    dLL dDF      pvalue
## 1 5.33   2 0.004864787
```

```
fitWL_Sex
```

```
## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 10.0668456 -0.1662865  5.0000000
##
## Log-likelihood: -791.67
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
##  9.7207432 10.4517151 -0.1244582  5.0000000
##
## Log-likelihood: -788.83
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
```

```
## Coefficients:
##          L1       alpha     myshape
## 10.17726651 -0.08449818  5.00000000
##
## Log-likelihood: -388.15
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
##  9.9163603 -0.2665775  5.0000000
##
## Log-likelihood: -402.77
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 10.09908263 10.32600553 -0.06085799  5.00000000
##
## Log-likelihood: -388.03
## Best method: bobyqa
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
```

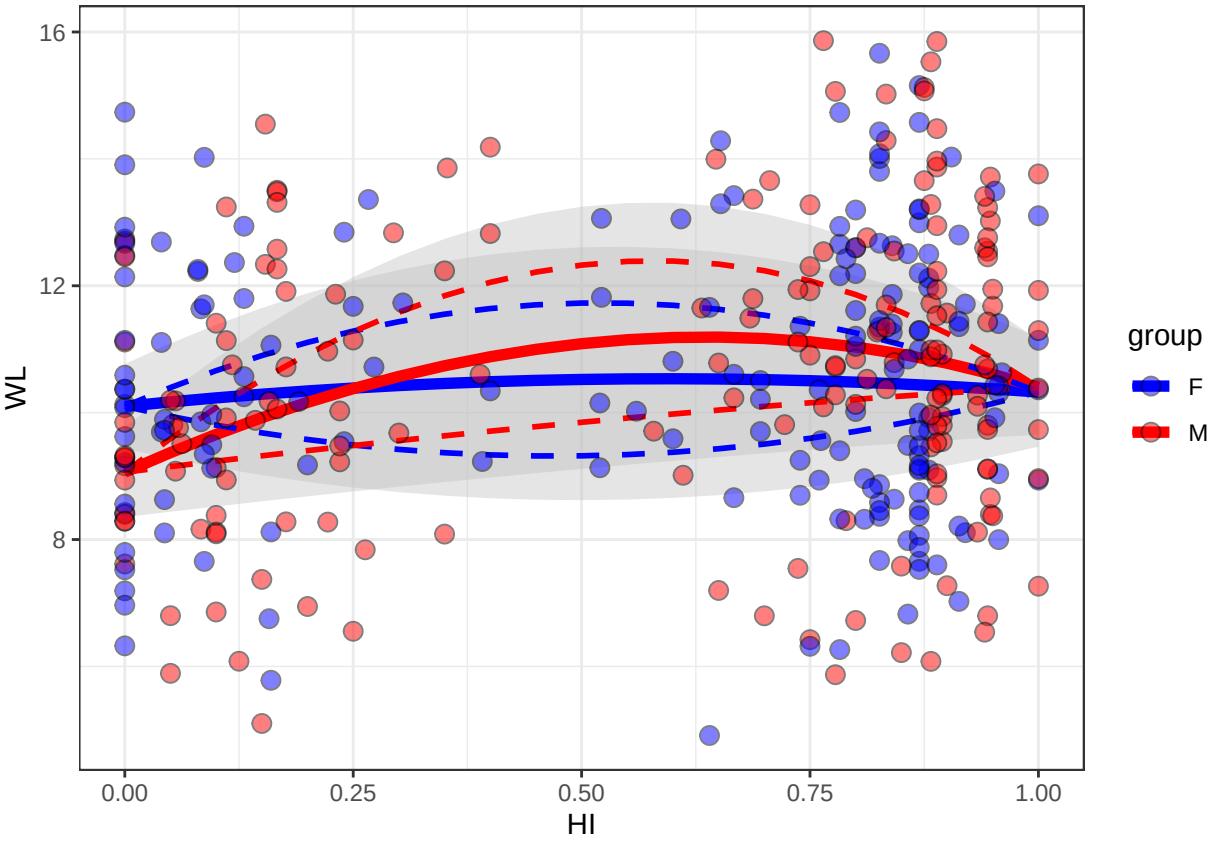
```
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
##  9.0616688 10.3980246 -0.2789579  5.0000000
##
## Log-likelihood: -397.56
## Best method: bobyqa
```

```
plot_WL_Sex<- bananaPlot(mod = fitWL_Sex$H3,
              data = Field,
              response = "WL",
              group = "Sex") +
    scale_fill_manual(values = c("blue", "red")) +
  scale_color_manual(values = c("blue", "red")) +
  theme_bw()
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```

```
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
```

```
plot_WL_Sex
```



H0: the expected load for the subspecies and between 2 groups is the same

H1: the mean load across 2 groups is the same, but can differ across subspecies

H2: the mean load across subspecies is the same, but can differ between the 2 groups

H3: the mean load can differ both across subspecies and between 2 groups

Summary stats for the field

Can we test the hybrid index, WL and the infection ?

```
Field %>%
  dplyr::group_by(MC.Eimeria) %>%
  dplyr::summarise(length(Mouse_ID))
```

```
## # A tibble: 3 x 2
##   MC.Eimeria `length(Mouse_ID)`
##   <lgl>                   <int>
## 1 FALSE                     111
## 2 TRUE                       92
## 3 NA                        154
```

```
Field %>%
    dplyr::group_by(eimeriaSpecies) %>%
    dplyr::summarize(length(Mouse_ID))
```

```
## # A tibble: 4 x 2
##   eimeriaSpecies `length(Mouse_ID)`
##   <chr>                       <int>
## 1 E_falciformis                   8
## 2 E_ferrisi                      10
## 3 Negative                       31
## 4 <NA>                          308
```

Reproducing for melting curve

```
Field_mc <- Field %>%
    drop_na("MC.Eimeria")

parasiteLoad::getParamBounds("weibull", data = Field_mc, response = "WL")
```

```
##      L1start         L1LB         L1UB      L2start         L2LB         L2UB
## 10.702131780  0.000000001 15.863165435 10.702131780  0.000000001 15.863165435
##   alphaStart      alphaLB      alphaUB myshapeStart    myshapeLB    myshapeUB
##  0.000000000 -5.000000000  5.000000000  1.000000000  0.000000001  5.000000000
```

```
speparam <- c(L1start = 10,
                      L1LB = 1e-9,
                      L1UB = 20,
                      L2start = 10,
                      L2LB = 1e-9,
                      L2UB = 20,
                      alphaStart = 0, alphaLB = -5, alphaUB = 5,
                      myshapeStart = 1, myshapeLB = 1e-9, myshapeUB = 5)


Field_mc <- Field_mc %>%
    dplyr::mutate(Eimeria = case_when(
        MC.Eimeria == "TRUE" ~ "positive",
        MC.Eimeria == "FALSE" ~ "negative",
        TRUE ~ ""
    ))

Field_mc$Eimeria <- as.factor(Field_mc$Eimeria)
```

```r
##All
fitWL_Eimeria <- parasiteLoad::analyse(data = Field_mc,
                          response = "WL",
                          model = "weibull",
                          group = "Eimeria")
```

```
## [1] "Analysing data for response: WL"
## [1] "Fit for the response: WL"
## [1] "Fitting for all"
## [1] "Fitting model basic without alpha"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable
```

```
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable
```

```
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable
```

```
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable
```

```
## [1] "Did converge"
## [1] "Fitting for groupA : negative"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Fitting for groupB : positive"
## [1] "Fitting model basic without alpha"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable
```

```
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable
```

```
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable
```

```
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable
```

```
## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.55   1 0.07805335
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 1.03   1 0.1520287
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.74   1 0.06200102
## [1] "Testing H2 groupB no alpha vs alpha"
##   dLL dDF    pvalue
## 1   0   1 0.9701239
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.47   1 0.08614847
## [1] "Testing H3 groupB no alpha vs alpha"
##   dLL dDF    pvalue
## 1   0   1 0.9532852
## [1] "Testing H1 vs H0"
##   dLL dDF     pvalue
## 1 1.9   1 0.05099885
## [1] "Testing H2 vs H0"
##    dLL dDF     pvalue
## 1 5.03   3 0.01809188
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 3.66   4 0.1197474
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.54   2 0.5841089
```

```
fitWL_Eimeria
```

```
## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 10.1293877 -0.2141842  5.0000000
```

```
##
## Log-likelihood: -454.29
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
##  9.7034446 10.5163844 -0.1700546  5.0000000
##
## Log-likelihood: -452.39
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
##  9.5295889 -0.2862507  4.8104963
##
## Log-likelihood: -250.8
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##           L1        alpha      myshape
## 11.098604487  0.007472937  5.000000000
```

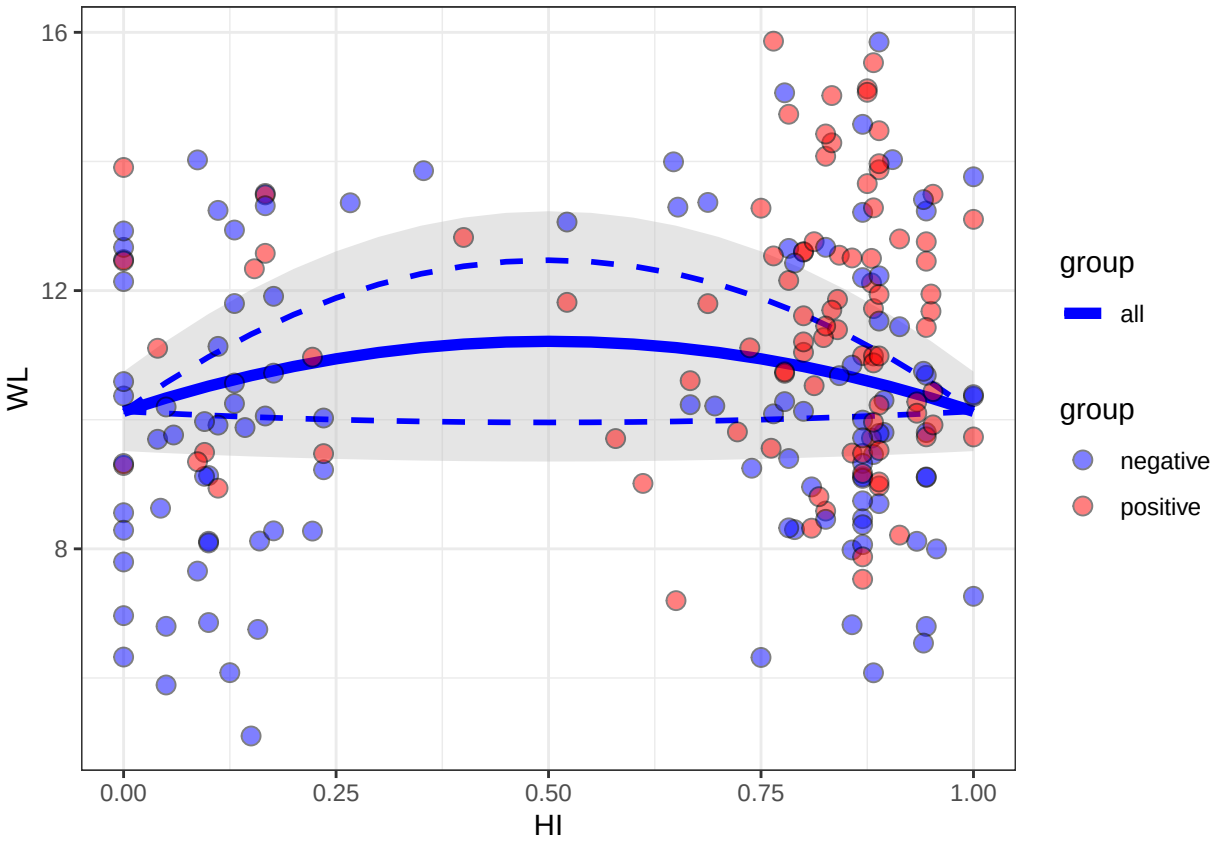
```
##
## Log-likelihood: -198.46
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
##  9.3673664  9.7655252 -0.2624788  4.8327103
##
## Log-likelihood: -250.49
## Best method: bobyqa
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 10.66851162 11.25663336  0.01138984  5.00000000
##
## Log-likelihood: -198.24
## Best method: bobyqa
```

```
plot_WL_Eimeria <- bananaPlot(mod = fitWL_Eimeria$H0,
             data = Field_mc,
             response = "WL",
             group = "Eimeria") +
  scale_fill_manual(values = c("blue", "red")) +
  scale_color_manual(values = c("blue", "red")) +
  theme_bw()
```

```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```

```
## Scale for 'colour' is already present. Adding another scale for 'colour',
## which will replace the existing scale.
```

```
plot_WL_Eimeria
```



Applying the classification model on the field data

Predicting melting curve

```
# load model
load("r_scripts/models/predicting_mc.RData")

#The predict() function in R is used to predict the values based on the input
# data.
predictions_mc_field <- predict(model_mc, completeField)

#add the new variable of predictions to the result object
Field <- cbind(Field, predictions_mc_field)


#turn the variable of melting curve into a factor so that you can compare to
# the predictions
Field$predictions_mc_field <-
  as.factor(as.character(Field$predictions_mc_field))

Field$MC.Eimeria <-
  as.factor(as.character(Field$MC.Eimeria))
```

Confusion matrix

```
Field_mc <- Field %>%
  filter(!MC.Eimeria == "NA")
```

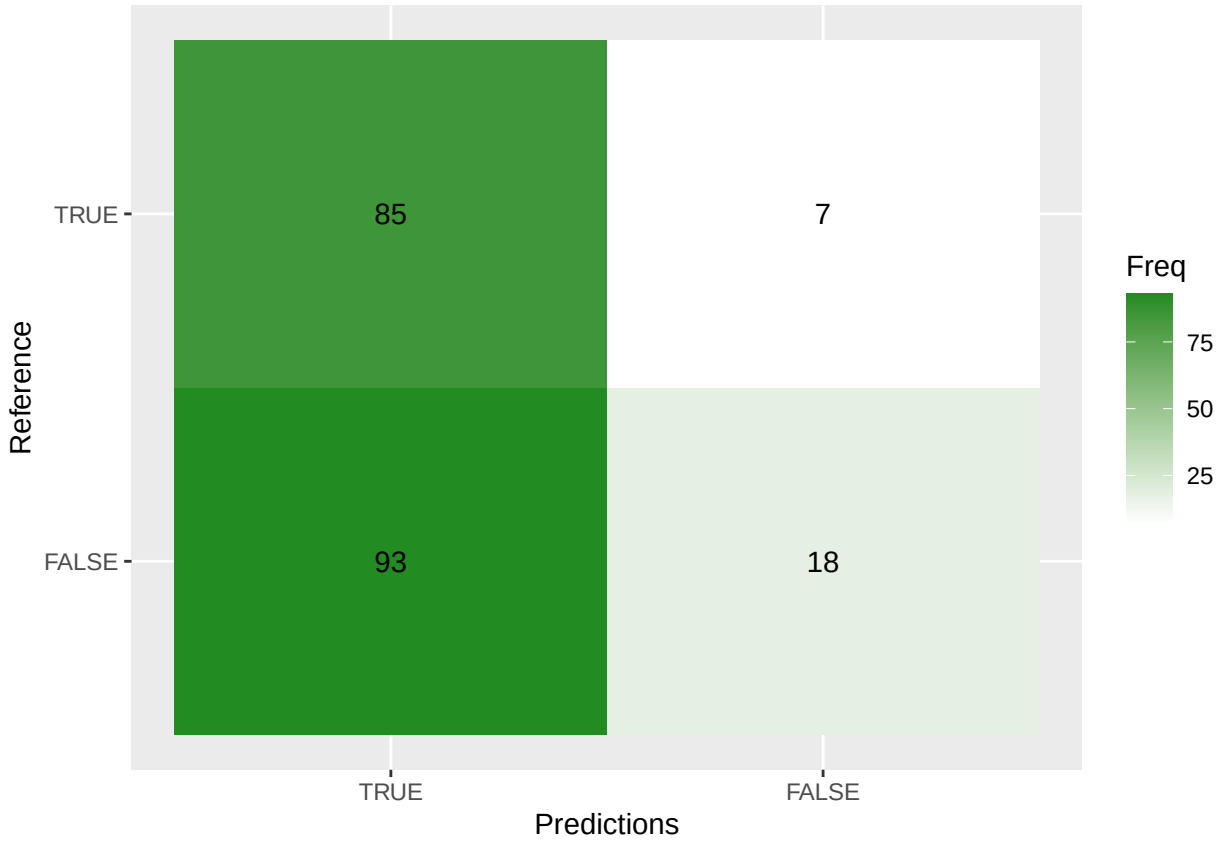
```
conf_matrix_mc <-
  confusionMatrix(Field_mc$predictions_mc_field,
                  reference = Field_mc$MC.Eimeria)

print(conf_matrix_mc)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE    18    7
##      TRUE     93   85
##
##                Accuracy : 0.5074
##                  95% CI : (0.4365, 0.5781)
##     No Information Rate : 0.5468
##     P-Value [Acc > NIR] : 0.8845
##
##                   Kappa : 0.0797
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.16216
##             Specificity : 0.92391
##          Pos Pred Value : 0.72000
##          Neg Pred Value : 0.47753
##              Prevalence : 0.54680
##          Detection Rate : 0.08867
##    Detection Prevalence : 0.12315
##       Balanced Accuracy : 0.54304
##
##        'Positive' Class : FALSE
##
```

```
conf_matrix_mc$table
```

```
##           Reference
## Prediction FALSE TRUE
##      FALSE    18    7
##      TRUE     93   85
```

```
plt <- as.data.frame(conf_matrix_mc$table)

plt$Prediction <- factor(plt$Prediction, levels=rev(levels(plt$Prediction)))
ggplot(plt, aes(x = Prediction, y =  Reference, fill= Freq)) +
        geom_tile() + geom_text(aes(label=Freq)) +
        scale_fill_gradient(low="white", high="forestgreen") +
        labs(x = "Predictions",y = "Reference")
```



Test the other model for parasite species

Making predictions

```
load("r_scripts/models/predict_infecting_parasite.RData")

#The predict() function in R is used to predict the values based on
# the input data.
predictions_parasite_field <- predict(model_Parasite, completeField)

# edit the infecting parasite names to correspond the parasite names in the
# model
Field$eimeriaSpecies <-
  gsub("Negative", "uninfected", Field$eimeriaSpecies)


Field$eimeriaSpecies <- as.factor(Field$eimeriaSpecies)

#add the new variable of predictions to the result object
Field_Eim <- cbind(Field, predictions_parasite_field)

# drop the rows with nas in the parasite
Field_Eim <- Field_Eim %>%
  drop_na(eimeriaSpecies)
```

Visualizations

```
conf_matrix_parasite <-
  confusionMatrix(Field_Eim$predictions_parasite_field,
```

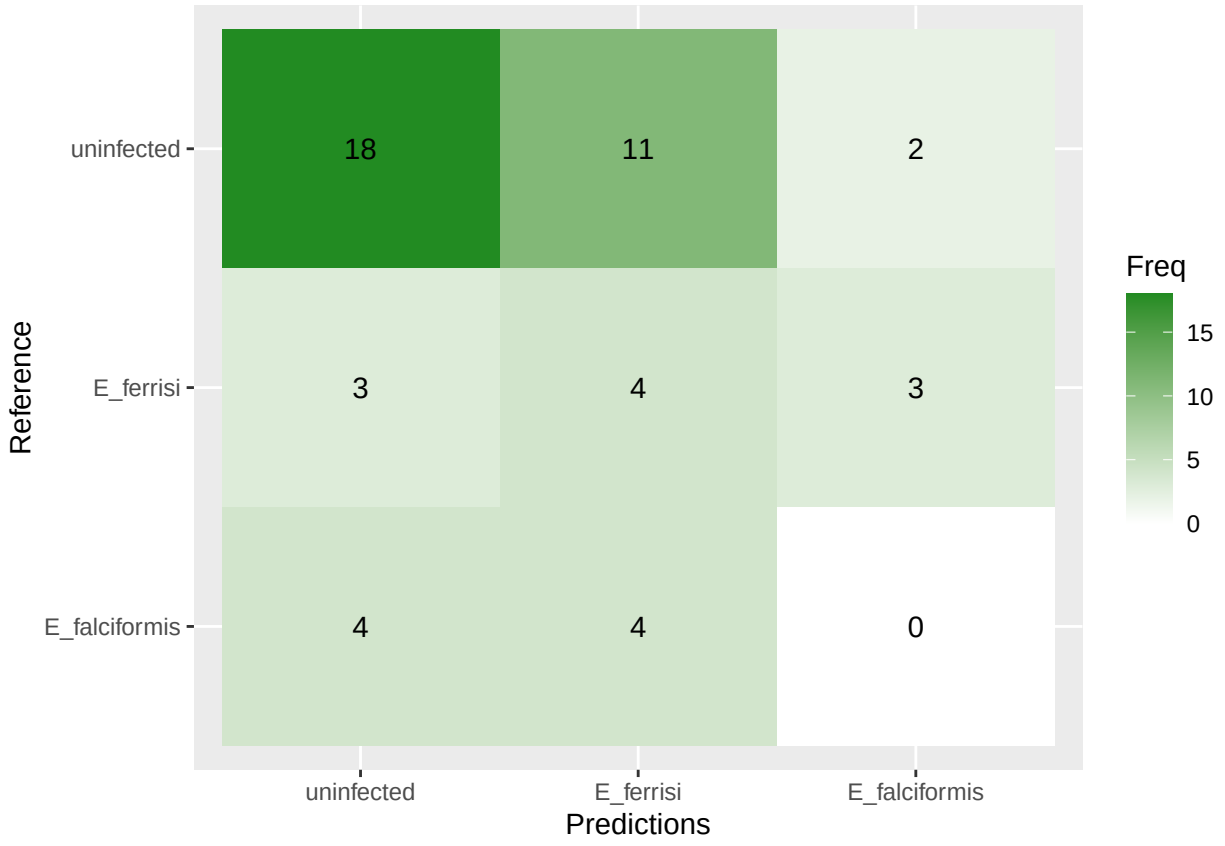
```
                  reference = Field_Eim$eimeriaSpecies)

print(conf_matrix_parasite)
```

```
## Confusion Matrix and Statistics
##
##                Reference
## Prediction      E_falciformis E_ferrisi uninfected
##   E_falciformis             0         3          2
##   E_ferrisi                 4         4         11
##   uninfected                4         3         18
##
## Overall Statistics
##
##                Accuracy : 0.449
##                  95% CI : (0.3067, 0.5977)
##     No Information Rate : 0.6327
##     P-Value [Acc > NIR] : 0.9971
##
##                   Kappa : 0.0523
##
##  Mcnemar's Test P-Value : 0.1459
##
## Statistics by Class:
##
##                      Class: E_falciformis Class: E_ferrisi Class: uninfected
## Sensitivity                        0.0000          0.40000            0.5806
## Specificity                        0.8780          0.61538            0.6111
## Pos Pred Value                     0.0000          0.21053            0.7200
## Neg Pred Value                     0.8182          0.80000            0.4583
## Prevalence                         0.1633          0.20408            0.6327
## Detection Rate                     0.0000          0.08163            0.3673
## Detection Prevalence               0.1020          0.38776            0.5102
## Balanced Accuracy                  0.4390          0.50769            0.5959
```

```
conf_matrix_parasite$table
```

```
##                Reference
## Prediction      E_falciformis E_ferrisi uninfected
##   E_falciformis             0         3          2
##   E_ferrisi                 4         4         11
##   uninfected                4         3         18
```

```
plt <- as.data.frame(conf_matrix_parasite$table)
plt$Prediction <- factor(plt$Prediction, levels=rev(levels(plt$Prediction)))

ggplot(plt, aes(x = Prediction, y =  Reference, fill= Freq)) +
        geom_tile() + geom_text(aes(label=Freq)) +
        scale_fill_gradient(low="white", high="forestgreen") +
        labs(x = "Predictions",y = "Reference")
```


Reference
uninfected
E_ferrisi
E_falciformis
18
11
2
3
4
3
4
4
0
uninfected
E_ferrisi
E_falciformis
Predictions
Freq
15
10
5
0