

10. Applying random forest on field data - gene

Fay

2022-11-04

Aim:

- Applying the models established in the script: 9
- 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

```
hm <- read.csv("output_data/imputed_mice.csv")
```

Clean data

```
Field <- hm %>%  
  filter(origin == "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_Mmin

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

Genes_wild_imp  <- paste(Genes_wild, "imp", sep = "_")
```

Prepare vectors for selecting

Actual Cleaning

```
#remove the unimputed columns
Field <- unique(Field) %>%
  dplyr::select(-all_of(Genes_wild)) %>%
  rename_with(~str_remove(., '_imp'))

#select the imputed gene columns
gene <- Field %>%
  dplyr::select(c(Mouse_ID, all_of(Genes_wild)))

genes <- gene %>%
  dplyr::select(-Mouse_ID)

#remove rows with only nas
genes <- genes[,colSums(is.na(genes))<nrow(genes)]

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

# select the same rows from the gene data
gene <- gene[row.names(genes),]

# select the same rows from the field data
Field <- Field[row.names(genes),]
```

Predicting weight loss in our imputed field data

Start by making the predictions for the field data.

```
# load predicting weight loss model
weight_loss_predict <- readRDS("r_scripts/models/predict_WL.rds")

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, genes)

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

```
#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.

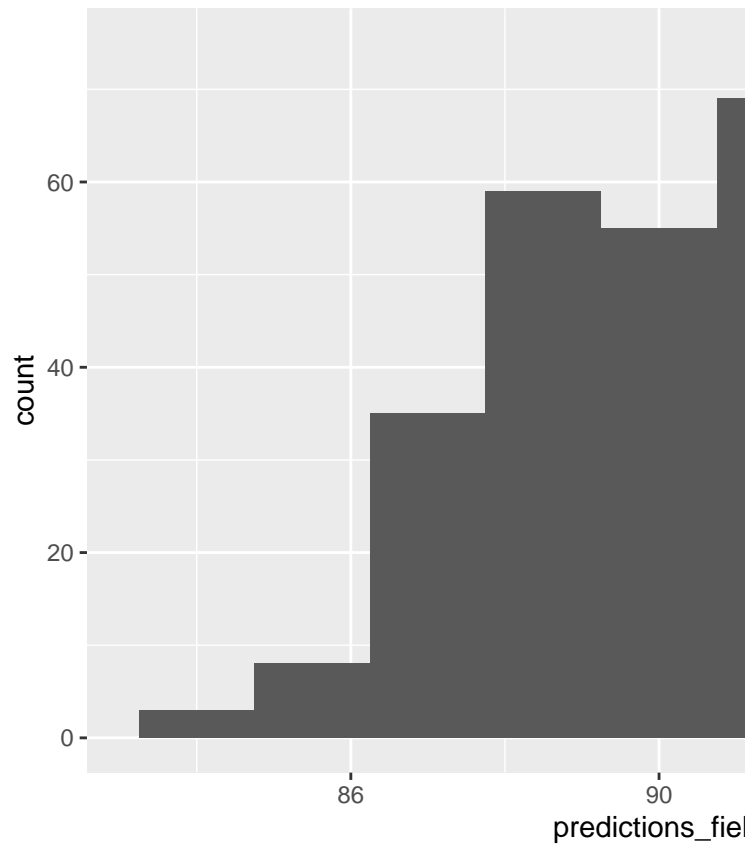
Install the package

```
## ggplot2 (5aee4a6ba... -> b27acf49c...) [GitHub]
##
## * checking for file '/tmp/Rtmp4imQjc/remotesaf3aa460e5868/cran-ggplot2-b27acf4/DESCRIPTION' ... OK
## * preparing 'ggplot2':
## * checking DESCRIPTION meta-information ... OK
## * checking vignette meta-information ... OK
## * checking for LF line-endings in source and make files and shell scripts
## * checking for empty or unneeded directories
## * building 'ggplot2_3.4.0.tar.gz'
##
## * checking for file '/tmp/Rtmp4imQjc/remotesaf3aa210d18d/alicebalard-parasiteLoad-1b43216/DESCRIPTION' ... OK
## * 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)
```



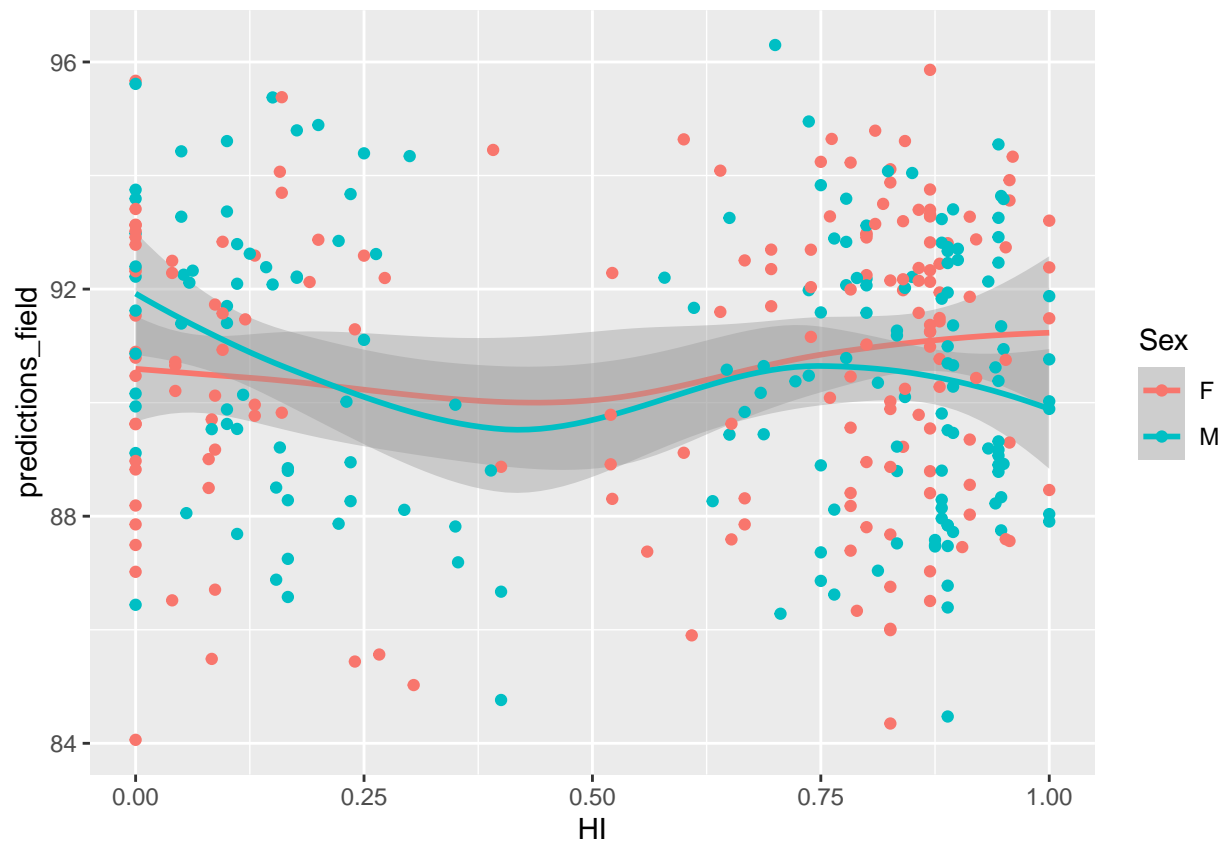
What is the distribution of the predicted weight loss?

Rough graph of our predictions against the hybrid index and against the

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

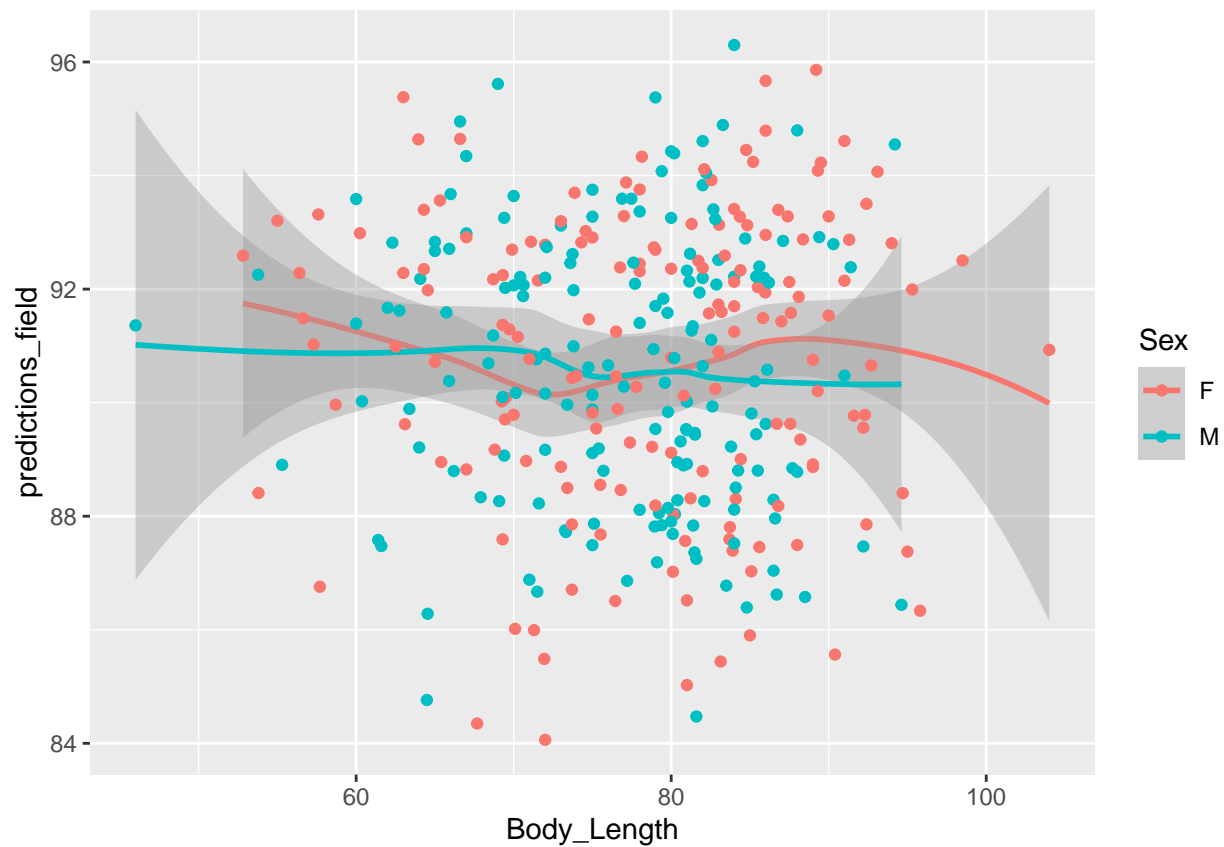
body length

```
## `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.

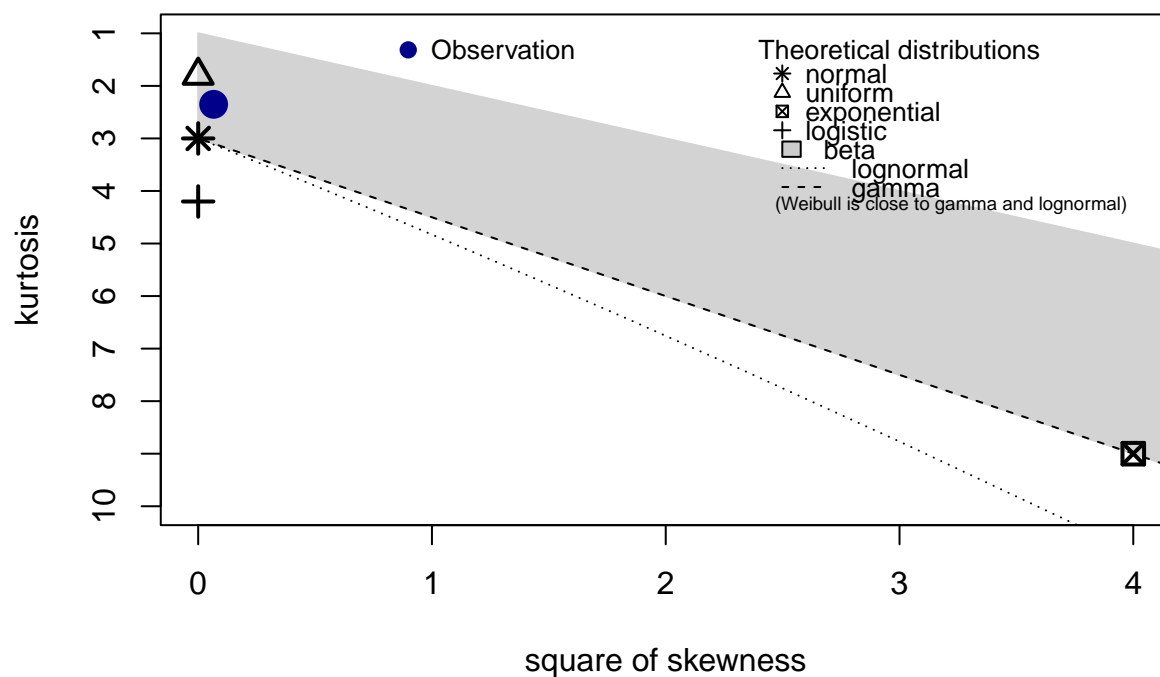
Alice used weibull for the qpcr data. (paper)

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

x <- Field$WL

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

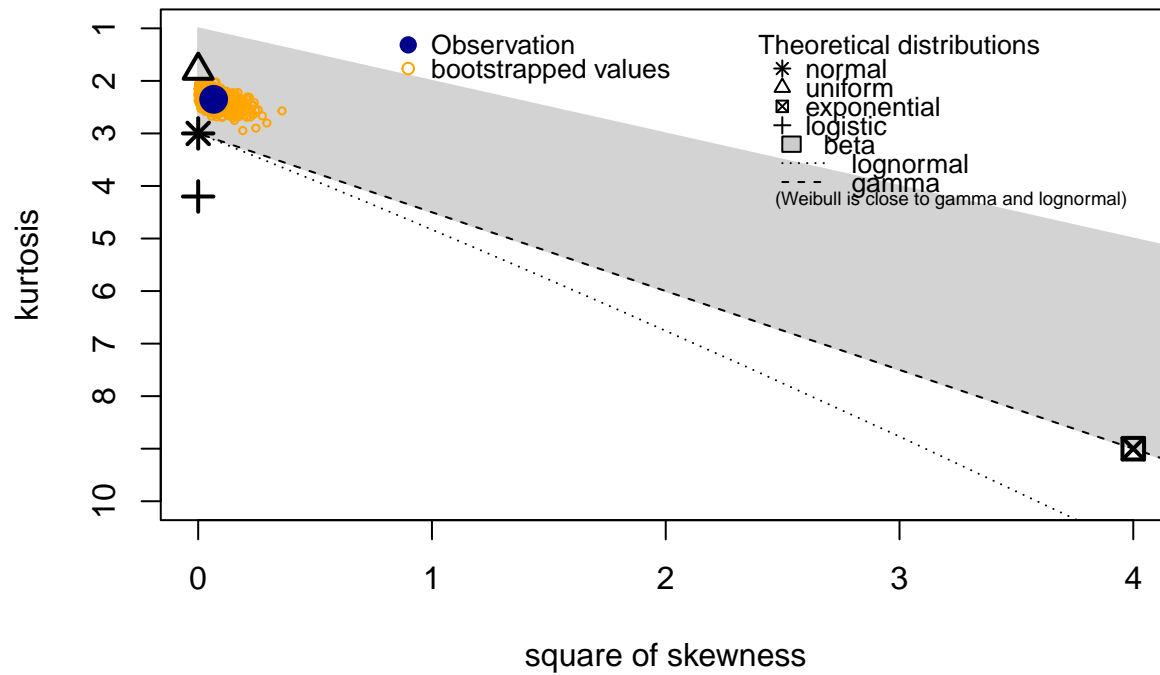
Cullen and Frey graph



```
## summary statistics
## -----
## min: 84.06162 max: 96.29867
## median: 90.93037
## mean: 90.69378
## estimated sd: 2.492399
## estimated skewness: -0.2572451
## estimated kurtosis: 2.350916
```

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

Cullen and Frey graph



```
## summary statistics
## -----
## min: 84.06162 max: 96.29867
## median: 90.93037
## mean: 90.69378
## estimated sd: 2.492399
## estimated skewness: -0.2572451
## estimated kurtosis: 2.350916
```

Test for binomial distribution

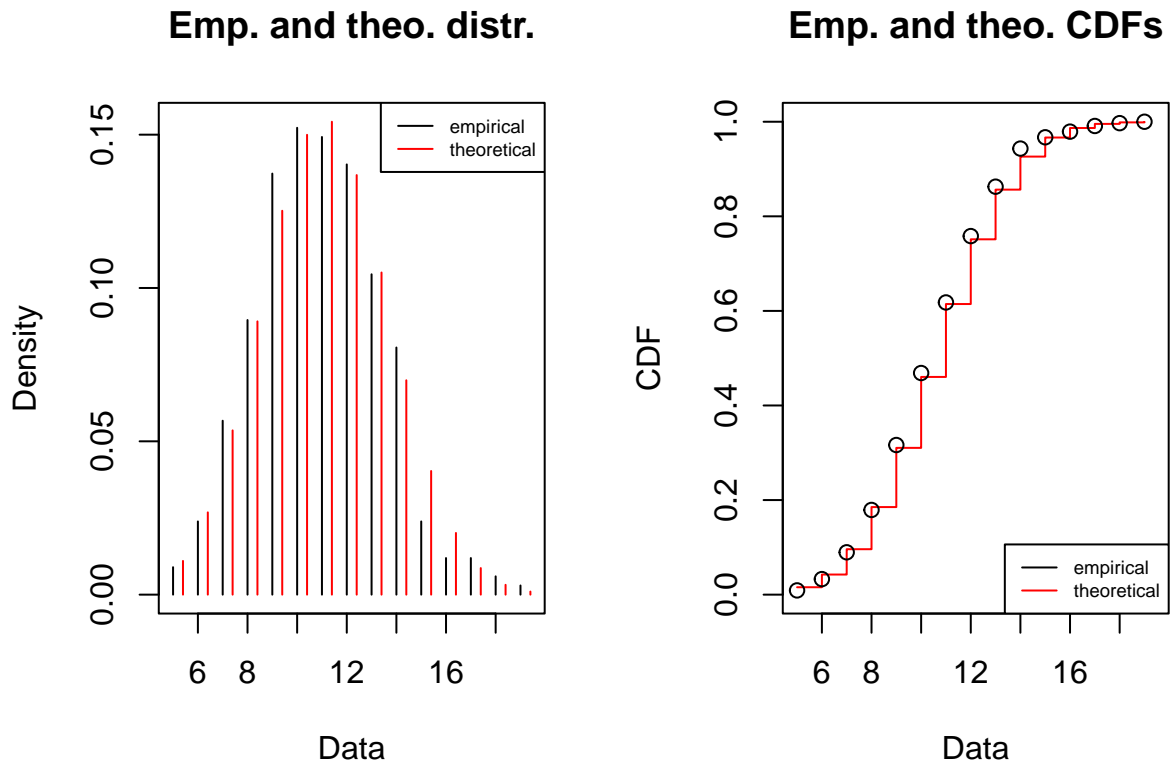
```
set.seed(10)
n = 25
size = 27
prob = .4
data = rbinom(x, size = size, prob = prob)
fit = fitdist(data = data, dist="binom",
              fix.arg=list(size = size),
              start=list(prob = 0.1))

summary(fit)
```

```
## Fitting of the distribution ' binom ' by maximum likelihood
## Parameters :
##      estimate Std. Error
## prob 0.399558 0.005150141
## Fixed parameters:
##      value
## size    27
```


Loglikelihood: -779.317 AIC: 1560.634 BIC: 1564.448

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, distrib, distrib2){
  l =lapply(distrib, function(i) tryDistrib(x, i))
  names(l) <- distrib
  print(l)
  listDistr <- lapply(distrib2, 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=distrib2)
  cdfcomp(listDistr, legendtext=distrib2)
  qqcomp(listDistr, legendtext=distrib2)
  ppcomp(listDistr, legendtext=distrib2)
  par(mfrow=c(1,1))
}

```

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

Functions for testing distributions

```

## $fit
##      mean      sd
##  90.69377795  2.48867665
##  ( 0.13597093) ( 0.09614597)
##
## $loglik
## [1] -780.781
##
## $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
##  41.4653052  91.8758382
##  ( 1.7368487) ( 0.1279802)
##
## $loglik
## [1] -782.1914

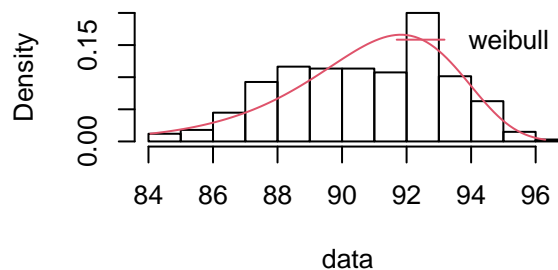
```

```
##
## $AIC
## NULL
tryDistrib(x, "weibullshifted")

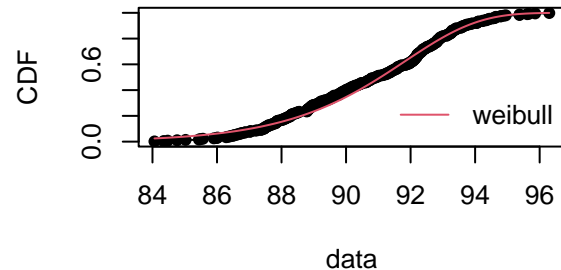
## $fit
## [1] "fit failed"
##
## $loglik
## [1] "no loglik computed"
##
## $AIC
## [1] "no aic computed"
findGoodDist(x, "normal", "weibull")

## $normal
## $normal$fit
##      mean      sd
##  90.69377795  2.48867665
##  ( 0.13597093) ( 0.09614597)
##
## $normal$loglik
## [1] -780.781
##
## $normal$AIC
## NULL
```

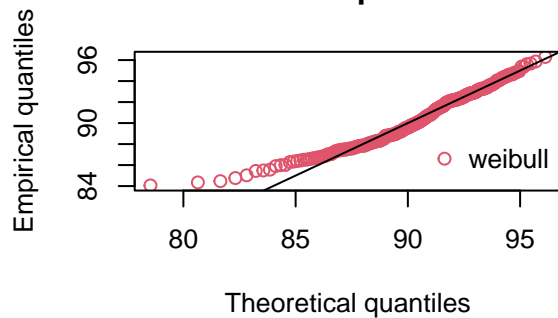
Histogram and theoretical densities



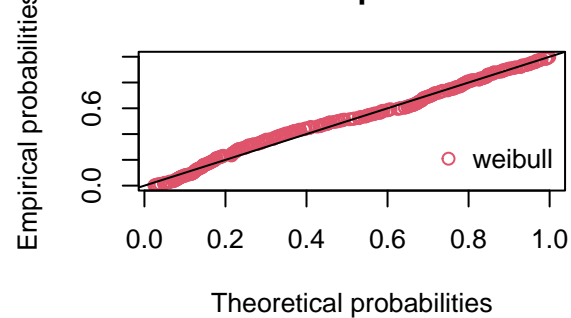
Empirical and theoretical CDFs



Q-Q plot

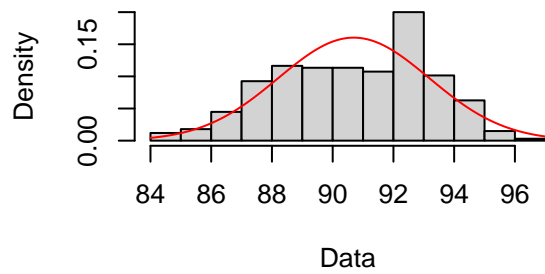


P-P plot

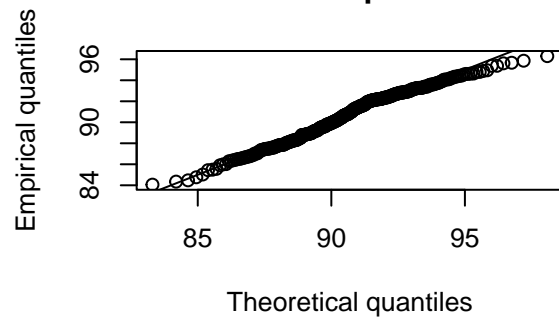


```
plot(normal_)
```

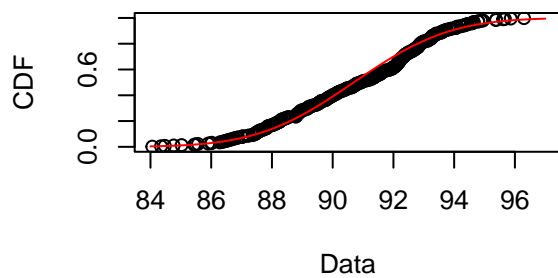
Empirical and theoretical dens.



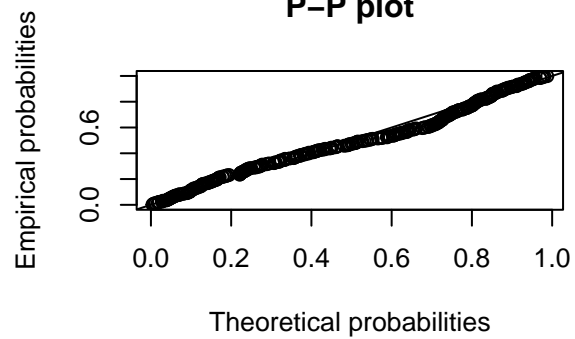
Q-Q plot



Empirical and theoretical CDFs



P-P plot

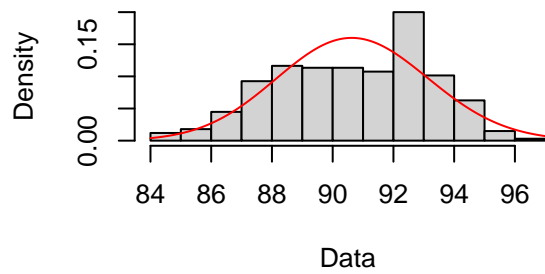


```
summary(normal_)
```

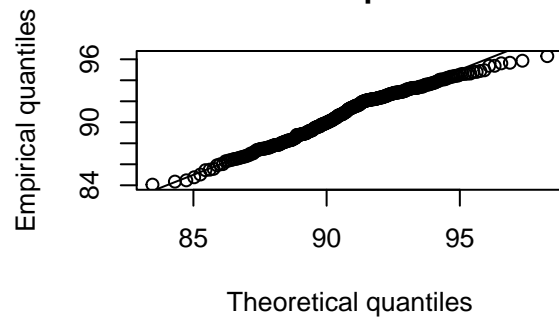
```
## Fitting of the distribution ' norm ' by maximum likelihood
## Parameters :
##      estimate Std. Error
## mean 90.693778  0.1359709
## sd   2.488677  0.0961459
## Loglikelihood: -780.781   AIC: 1565.562   BIC: 1573.19
## Correlation matrix:
##      mean sd
## mean  1  0
## sd    0  1
```

```
plot(gamma_)
```

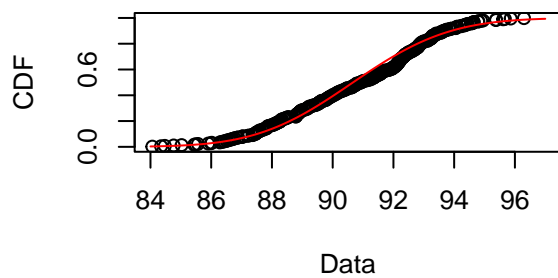
Empirical and theoretical dens.



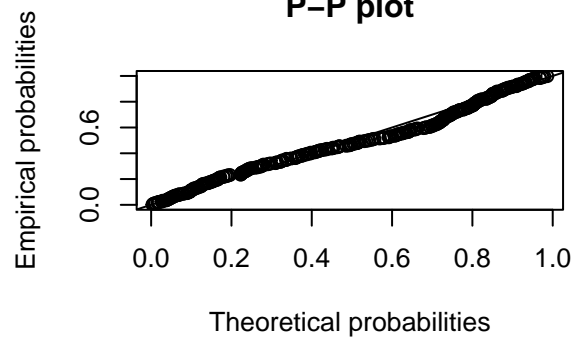
Q-Q plot



Empirical and theoretical CDFs



P-P plot

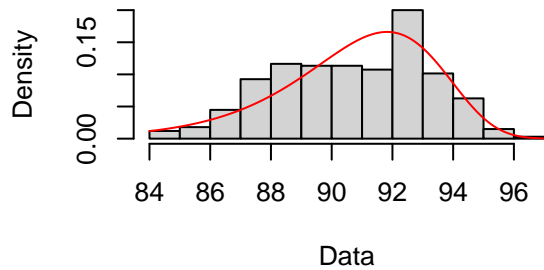


```
summary(gamma_)
```

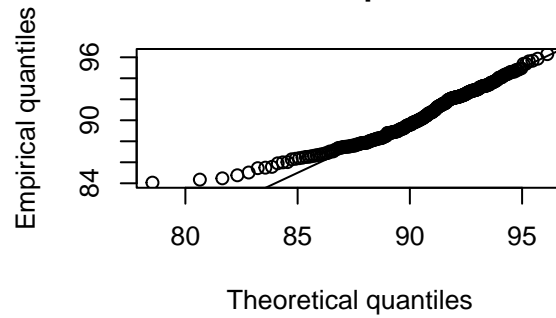
```
## Fitting of the distribution ' gamma ' by maximum likelihood
## Parameters :
##      estimate Std. Error
## shape 1321.02729  99.375775
## rate   14.56579   1.095947
## Loglikelihood: -781.6076  AIC:  1567.215  BIC:  1574.843
## Correlation matrix:
##      shape      rate
## shape 1.0000000 0.9998004
## rate   0.9998004 1.0000000
```

```
plot(weibull_)
```

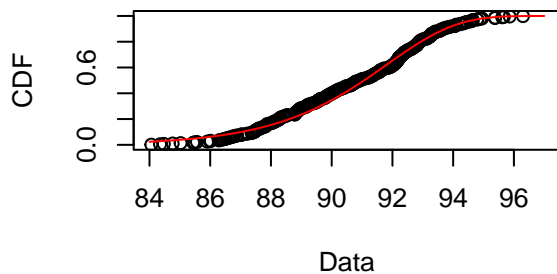
Empirical and theoretical dens.



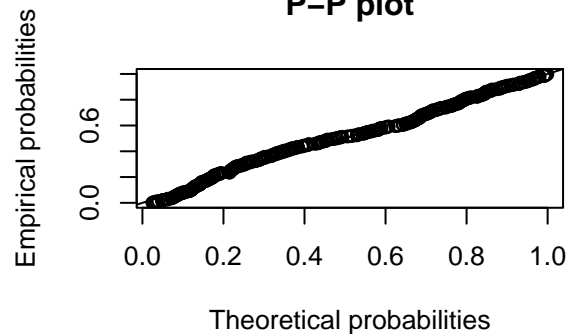
Q-Q plot



Empirical and theoretical CDFs



P-P plot



```
summary(weibull_)
```

```
## Fitting of the distribution ' weibull ' by maximum likelihood
## Parameters :
##      estimate Std. Error
## shape 41.46612   1.736871
## scale 91.87592   0.127978
## Loglikelihood: -782.1914   AIC:  1568.383   BIC:  1576.011
## Correlation matrix:
##      shape    scale
## shape 1.000000 0.324389
## scale 0.324389 1.000000
```

We have a weibull distribution!

Is alpha significant for each hypothesis?

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

```
Field <- Field %>%
  mutate(WL = 100 - WL)
```

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

```
##      L1start      L1LB      L1UB      L2start      L2LB      L2UB
## 9.306222052 0.000000001 15.938381859 9.306222052 0.000000001 15.938381859
## 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"
## [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 groupA : F"
## [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 : M"
## [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] "Testing H0 no alpha vs alpha"
##      dLL dDF   pvalue
## 1 1.16    1 0.1283225
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF   pvalue
## 1 1.23    1 0.116121
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF   pvalue
## 1 0.2     1 0.5246522
## [1] "Testing H2 groupB no alpha vs alpha"

```

```
##      dLL dDF      pvalue
## 1 1.34      1 0.1011852
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.75      1 0.2210683
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 1.56      1 0.07695409
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 0.1       1 0.6624668
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 1.38      3 0.4306275
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 3.72      4 0.1140242
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 2.44      2 0.08709369
```

```
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
## 9.0079495 -0.1450148 4.1118351
##
## Log-likelihood: -780.86
## Best method: L-BFGS-B
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
```



```

## 9.0783810 8.9308463 -0.1526132 4.1153069
##
## Log-likelihood: -780.77
## 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
## 9.02274359 -0.08485842 3.88199642
##
## Log-likelihood: -394.37
## 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
## 8.9559551 -0.2250568 4.4073411
##
## Log-likelihood: -385.11
## Best method: L-BFGS-B
##
##
## $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"]]),

```

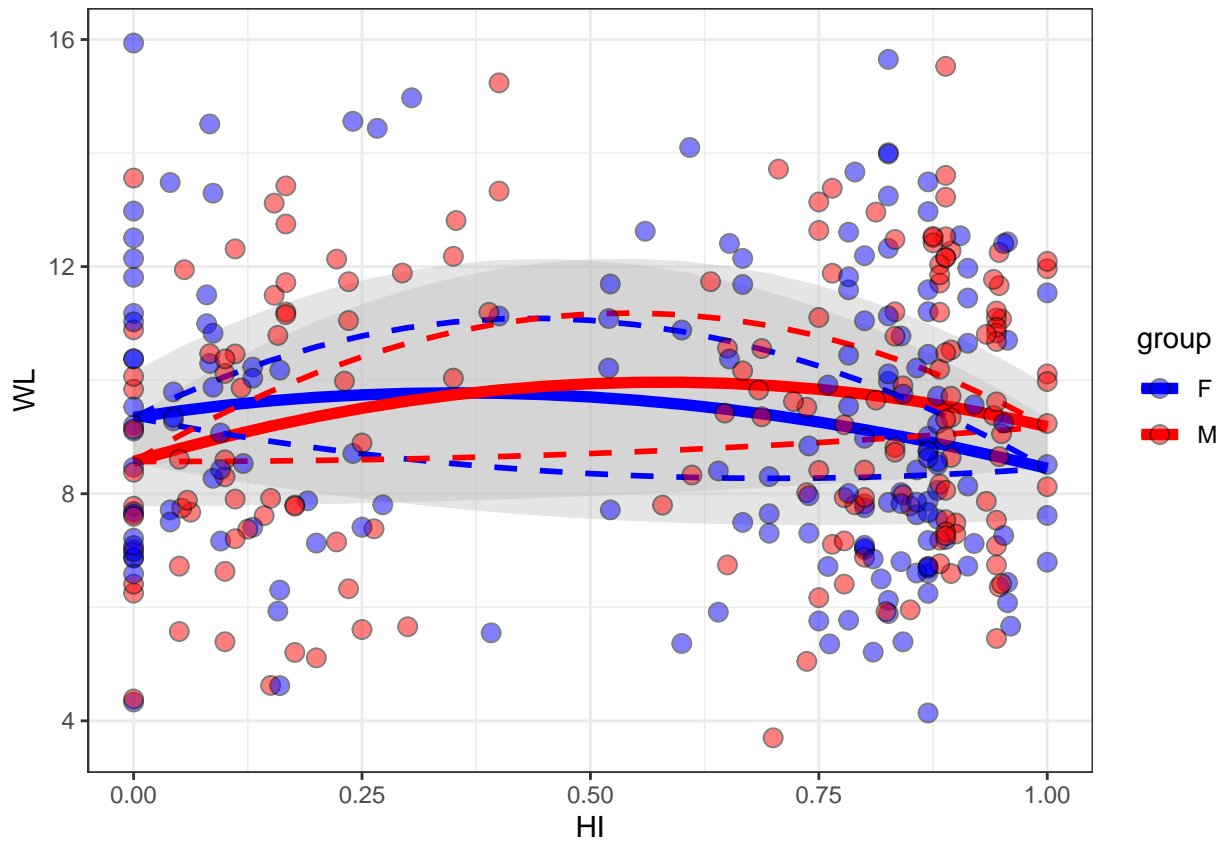
```

##      control = config$control)
##
## Coefficients:
##      L1      L2      alpha    myshape
##  9.3434134  8.4473983 -0.1819505  3.9233537
##
## Log-likelihood: -392.91
## 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$control)
##
## Coefficients:
##      L1      L2      alpha    myshape
##  8.5687035  9.1829740 -0.2393737  4.4256067
##
## Log-likelihood: -384.13
## Best method: bobyqa
plot_WL_Sex<- bananaPlot(mod = fitWL_Sex$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

```



[1]
 “Testing H1 no alpha vs alpha” dLL dDF pvalue 1 1.23 1 0.116121 [1] “Testing H2 groupA no alpha vs alpha” dLL dDF pvalue 1 1.34 1 0.1011852 [1] “Testing H2 groupB no alpha vs alpha” dLL dDF pvalue 1 1.56 1 0.07695409 [1] “Testing H1 vs H0” dLL dDF pvalue 1 0.1 1 0.6624668 [1] “Testing H2 vs H0” dLL dDF pvalue 1 1.38 3 0.4306275 [1] “Testing H3 vs H1” dLL dDF pvalue 1 3.72 4 0.1140242 [1] “Testing H3 vs H2” dLL dDF pvalue 1 2.44 2 0.08709369

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

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

```
Field1 <- Field %>%
  drop_na(MC.Eimeria)
Field$MC.Eimeria
```

##	[1]	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
##	[13]	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
##	[25]	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
##	[37]	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
##	[49]	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
##	[61]	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
##	[73]	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
##	[85]	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
##	[97]	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA

```
## [109] NA NA NA NA NA NA NA NA NA NA NA NA
## [121] NA NA NA NA NA NA NA NA NA NA NA NA
## [133] NA NA NA NA NA NA NA TRUE TRUE FALSE FALSE FALSE
## [145] FALSE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## [157] TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE TRUE
## [169] TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE
## [181] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE FALSE
## [193] TRUE TRUE TRUE TRUE TRUE FALSE FALSE FALSE TRUE TRUE TRUE TRUE
## [205] FALSE FALSE FALSE TRUE TRUE FALSE FALSE FALSE FALSE FALSE NA FALSE
## [217] FALSE FALSE FALSE NA FALSE FALSE FALSE TRUE FALSE NA TRUE FALSE
## [229] TRUE TRUE FALSE NA FALSE FALSE FALSE TRUE FALSE FALSE TRUE TRUE
## [241] FALSE TRUE TRUE TRUE TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE
## [253] NA TRUE FALSE NA FALSE FALSE TRUE FALSE TRUE TRUE TRUE FALSE
## [265] FALSE TRUE NA FALSE FALSE FALSE FALSE FALSE TRUE FALSE FALSE FALSE
## [277] FALSE FALSE FALSE TRUE NA FALSE TRUE FALSE FALSE FALSE FALSE FALSE
## [289] FALSE FALSE TRUE TRUE FALSE TRUE TRUE TRUE NA TRUE TRUE FALSE
## [301] FALSE TRUE TRUE FALSE TRUE NA TRUE FALSE FALSE TRUE TRUE TRUE
## [313] TRUE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE FALSE TRUE
## [325] TRUE TRUE TRUE NA TRUE TRUE TRUE FALSE TRUE FALSE TRUE
```

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

```
##      L1start      L1LB      L1UB      L2start      L2LB      L2UB
## 9.306222052 0.000000001 15.938381859 9.306222052 0.000000001 15.938381859
## 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"
## [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 groupA : F"
## [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 : M"
## [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] "Testing H0 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 1.16   1 0.1283225
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 1.23   1 0.116121
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.2    1 0.5246522
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 1.34   1 0.1011852
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.75   1 0.2210683
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 1.56   1 0.07695409
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 0.1    1 0.6624668
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 1.38   3 0.4306275
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 3.72   4 0.1140242
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 2.44   2 0.08709369
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
## 9.0079495 -0.1450148  4.1118351
##
## Log-likelihood: -780.86
## Best method: L-BFGS-B
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 9.0783810  8.9308463 -0.1526132  4.1153069
##
## Log-likelihood: -780.77
## 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
## 9.02274359 -0.08485842  3.88199642
##
## Log-likelihood: -394.37
## 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
## 8.9559551 -0.2250568  4.4073411
##
## Log-likelihood: -385.11
## Best method: L-BFGS-B
##
##
## $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
## 9.3434134  8.4473983 -0.1819505  3.9233537
##
## Log-likelihood: -392.91
## 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
## 8.5687035  9.1829740 -0.2393737  4.4256067
##
## Log-likelihood: -384.13
## 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
```

