

5. Gene_expression_analysis

Fay

2022-08-09

load libraries

```
library(ggplot2)
library(tidyverse)

## -- Attaching packages ----- tidyverse 1.3.2 --
## v tibble 3.1.8      v dplyr 1.0.10
## v tidyr 1.2.1      v stringr 1.4.1
## v readr 2.1.3      v forcats 0.5.2
## v purrr 0.3.5
## -- Conflicts ----- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()     masks stats::lag()

library(optimx)
```

Import data:

Here, we have the experimental / field data

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

# Selecting genes
Gene_lab <- 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")

#add a suffix to represent changes in data file
Gene_lab_imp <- paste(Gene_lab, "imp", sep = "_")

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") # "IRG6")

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

Facs_lab <- c("Position", "CD4", "Treg", "Div_Treg", "Treg17", "Th1",
             "Div_Th1", "Th17", "Div_Th17", "CD8", "Act_CD8",
             "Div_Act_CD8", "IFNy_CD4", "IFNy_CD8") #, "Treg_prop",
             # "IL17A_CD4")
```

```
Facs_wild <- c( "Treg", "CD4", "Treg17", "Th1", "Th17", "CD8",
               "Act_CD8", "IFNy_CD4", "IFNy_CD8") #"IL17A_CD4",
```

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 across genes

```
# Selecting the field samples

field <- hm %>%
  dplyr::filter(origin == "Field")

field <- unique(field)

#make a factor out of the melting curves (important for later visualization)
field <- field %>%
  dplyr::mutate(MC.Eimeria = as.factor(MC.Eimeria))

genes_mouse <- field %>%
  dplyr::select(all_of(Genes_wild))

genes <- genes_mouse

#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 same rows in the first table
field <- field[row.names(genes), ]
```

Install the package

```
##
## * checking for file '/tmp/RtmpDu5az5/remotes1173903596a86d/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'
```

Applying Alice's package on every gene

```
x <- field$ID01
```

```

# 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
## 14.9971943  4.3383944
## ( 0.2366788) ( 0.1673572)
##
## $loglik
## [1] -969.8448
##
## $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")

## Warning in densfun(x, parm[1], parm[2], ...): NaNs produced

## Warning in densfun(x, parm[1], parm[2], ...): NaNs produced

## $fit
##      shape      scale
## 3.5858906 16.6222536
## ( 0.1428207) ( 0.2686505)
##
## $loglik
## [1] -974.6869
##
## $AIC
## NULL
tryDistrib(x, "weibullshifted")

## $fit
## [1] "fit failed"
##
## $loglik
## [1] "no loglik computed"
##
## $AIC
## [1] "no aic computed"
# remove NA in HI
field <- field %>%
  drop_na(HI)

field$Sex <- as.factor(field$Sex)

parasiteLoad::getParamBounds("weibull", data = field, response = "IFNy")

##      L1start      L1LB      L1UB      L2start      L2LB      L2UB
## 20.311647815 0.000000001 29.607514874 20.311647815 0.000000001 29.607514874
## alphaStart   alphaLB   alphaUB myshapeStart myshapeLB myshapeUB
## 0.000000000 -5.000000000 5.000000000 1.000000000 0.000000001 5.000000000
IFNy <- parasiteLoad::analyse(data = field,
                             response = "IFNy",
                             model = "weibull",
                             group = "Sex")

```

```

## [1] "Analysing data for response: IFNy"
## [1] "Fit for the response: IFNy"
## [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"

```

```

## [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 0.46   1 0.335786
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF   pvalue
## 1 0.28   1 0.4547721
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF   pvalue
## 1 0.01   1 0.8922156
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF   pvalue
## 1 0.79   1 0.2074947
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF   pvalue
## 1 0.05   1 0.7456197
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF   pvalue
## 1 0.49   1 0.3223774
## [1] "Testing H1 vs H0"
##      dLL dDF   pvalue
## 1 0.16   1 0.5764781
## [1] "Testing H2 vs H0"
##      dLL dDF   pvalue
## 1 0.67   3 0.7185038
## [1] "Testing H3 vs H1"
##      dLL dDF   pvalue
## 1 1.53   4 0.5477734
## [1] "Testing H3 vs H2"
##      dLL dDF   pvalue
## 1 1.01   2 0.3628641

parasiteLoad::analyse(data = field,
                      response = "IFNy",
                      model = "weibull",
                      group = "Sex")

## [1] "Analysing data for response: IFNy"
## [1] "Fit for the response: IFNy"
## [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"
## [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 0.46    1 0.335786
## [1] "Testing H1 no alpha vs alpha"

```

```

##      dLL dDF      pvalue
## 1 0.28      1 0.4547721
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.01      1 0.8922156
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.79      1 0.2074947
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.05      1 0.7456197
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.49      1 0.3223774
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 0.16      1 0.5764781
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 0.67      3 0.7185038
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 1.53      4 0.5477734
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 1.01      2 0.3628641

## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 20.57149376  0.07228781  5.00000000
##
## Log-likelihood: -985.28
## 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
## 20.71696489 20.35233186 0.05940179 5.00000000
##
## Log-likelihood: -985.12
## 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
## 20.43567608 0.01425417 5.00000000
##
## Log-likelihood: -484.91
## 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
## 20.6716975 0.1399482 4.8667226
##
## Log-likelihood: -499.7
## 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
## 20.26996359 20.74628573  0.03686576  5.00000000
##
## Log-likelihood: -484.79
## 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
## 21.2507542 20.0210272  0.1136491  4.8912903
##
## Log-likelihood: -498.8
## Best method: bobyqa

```

```

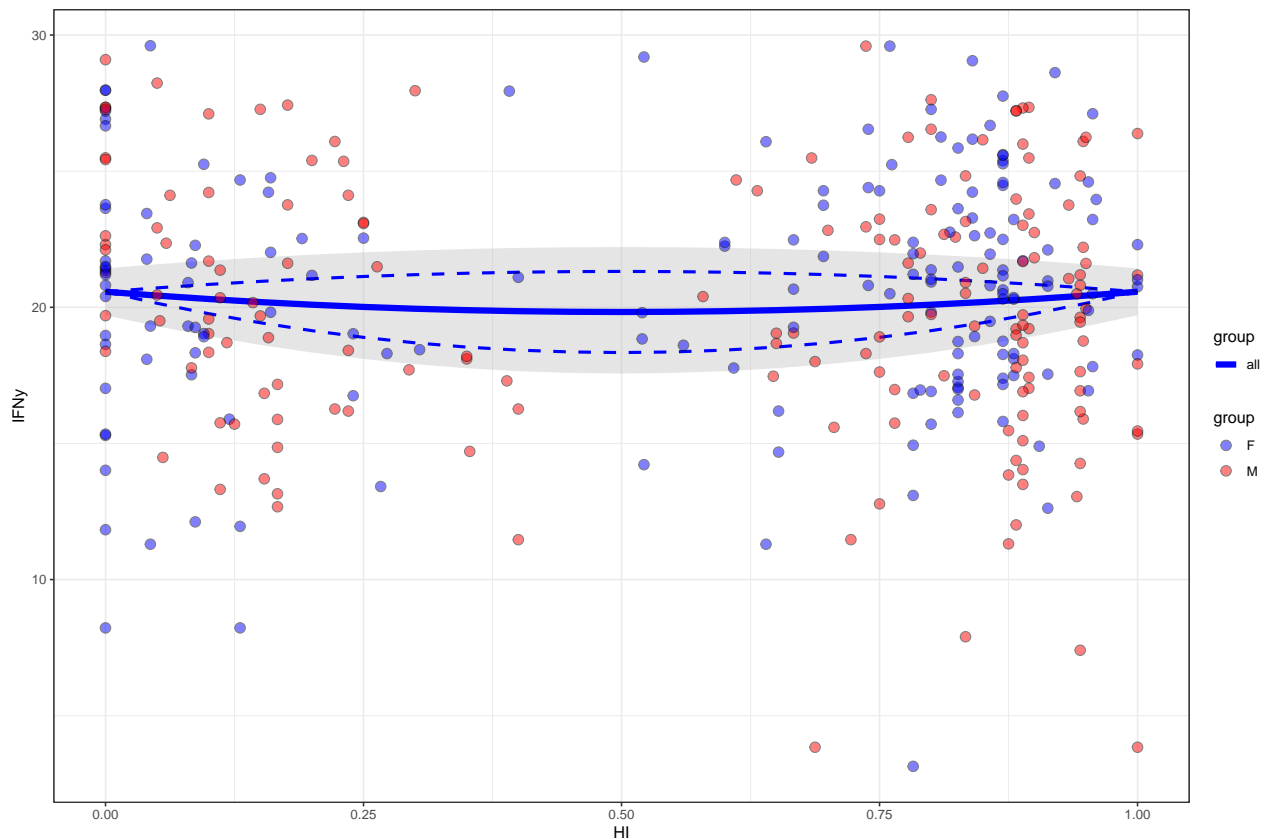
bananaPlot(mod = IFNy$H0,
            data = field,
            response = "IFNy",
            group = "Sex") +
  scale_fill_manual(values = c("blue", "red")) +
  scale_color_manual(values = c("blue", "red")) +
  theme_bw()

```

```

## 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.

```



```
CXCR3 <- parasiteLoad::analyse(data = field,
                               response = "CXCR3",
                               model = "weibull",
                               group = "Sex")

## [1] "Analysing data for response: CXCR3"
## [1] "Fit for the response: CXCR3"
## [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"

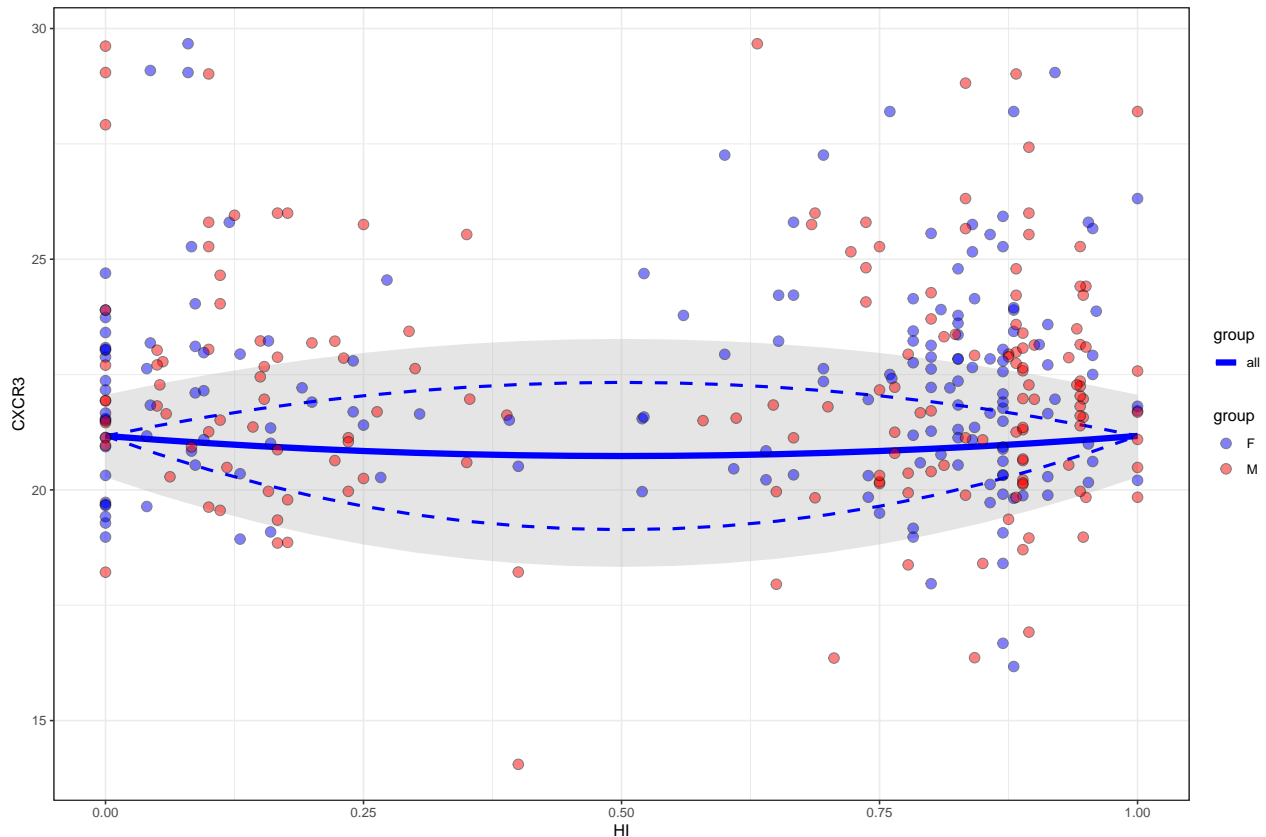
## Warning in bbmle::mle2(response ~ dweibull(shape = myshape, scale =
## MeanLoad(L1, : some parameters are on the boundary: variance-covariance
## calculations based on Hessian may be unreliable

```

```
## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.14    1 0.5968162
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.1    1 0.6483795
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1  0    1 0.974345
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.31    1 0.432465
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1  0    1 0.9802984
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.27    1 0.4636948
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 0.04    1 0.7889353
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 0.19    3 0.9463332
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 0.26    4 0.97082
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 0.11    2 0.8920405

bananaPlot(mod = CXCR3$H0,
            data = field,
            response = "CXCR3",
            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.
```



```
IL.6 <- parasiteLoad::analyse(data = field,
                             response = "IL.6",
                             model = "weibull",
                             group = "Sex")

## [1] "Analysing data for response: IL.6"
## [1] "Fit for the response: IL.6"
## [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 0.01    1 0.8956001
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1    0    1 0.9259378
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1    0    1 0.976793
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.01    1 0.8759232
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.02    1 0.8477779
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.01    1 0.8983205
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 0.02    1 0.830629
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 0.45    3 0.8251681
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 0.99    4 0.7406528
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 0.56    2 0.5717611
```

```
##All
```

```
print(IL.6)
```

```
## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 22.48064809  0.01041931  5.00000000
##
## Log-likelihood: -904
## 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
## 22.55041317 22.40211765  0.00750784  5.00000000
##
## Log-likelihood: -903.98
## 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
## 22.670875252  0.003217022  5.000000000
##
## Log-likelihood: -453.17
## 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
## 22.28299624  0.01785347  5.000000000
##
## Log-likelihood: -450.38
## 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
## 22.44113746 23.01284560  0.02202816  5.00000000
##
## Log-likelihood: -453.01
## Best method: L-BFGS-B
##
## $H3$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
## 22.7580806 21.8962172  0.0147476  5.0000000
##
## Log-likelihood: -449.98
## Best method: bobyqa

```

```

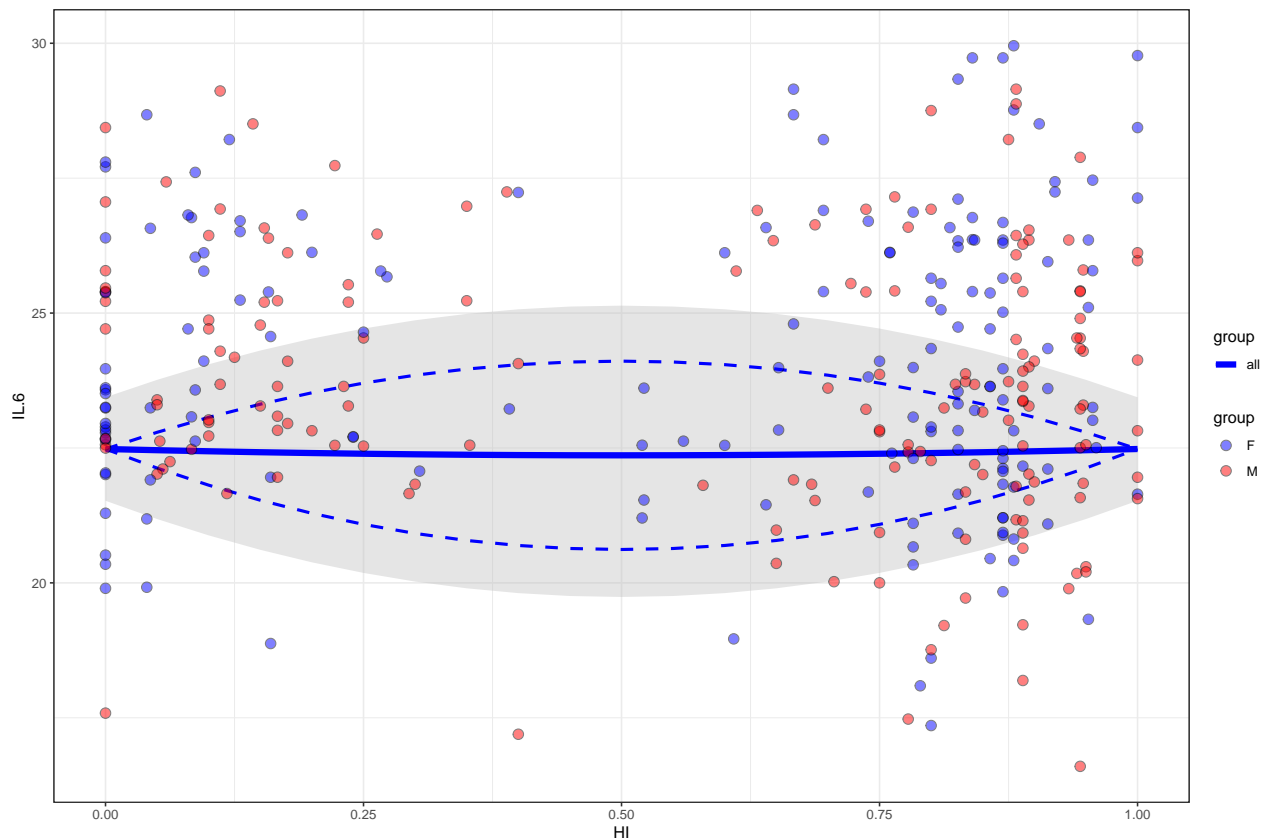
bananaPlot(mod = IL.6$H0,
           data = field,
           response = "IL.6",
           group = "Sex") +
  scale_fill_manual(values = c("blue", "red")) +
  scale_color_manual(values = c("blue", "red")) +
  theme_bw()

```

```

## 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.

```



```
IL.10 <- parasiteLoad::analyse(data = field,
                               response = "IL.10",
                               model = "weibull",
                               group = "Sex")
```

```
## [1] "Analysing data for response: IL.10"
## [1] "Fit for the response: IL.10"
## [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 0.1    1 0.6504249
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.05    1 0.7401676
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.02    1 0.8548983
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.11    1 0.6343149
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.03    1 0.8230158
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.08    1 0.6805082
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 0.15    1 0.5780891
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 0.08    3 0.9836571
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 0.4     4 0.9372508
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 0.48    2 0.6195723
```

```
##All
```

```
print(IL.10)
```

```
## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 23.50009295  0.03578727  5.00000000
##
## Log-likelihood: -913.82
## 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
## 23.68206223 23.27738820  0.02689144  5.00000000
##
## Log-likelihood: -913.66
## 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
## 23.49501930  0.01955936  5.00000000
##
## Log-likelihood: -455.69
## 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
## 23.52562017  0.05580705  5.00000000
##
## Log-likelihood: -458.05
## 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
## 23.4308738 23.5976487  0.0252457  5.0000000
##
## Log-likelihood: -455.68
## 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
## 24.04514310 23.06545122  0.04892828  5.00000000
##
## Log-likelihood: -457.58
## Best method: bobyqa

```

```

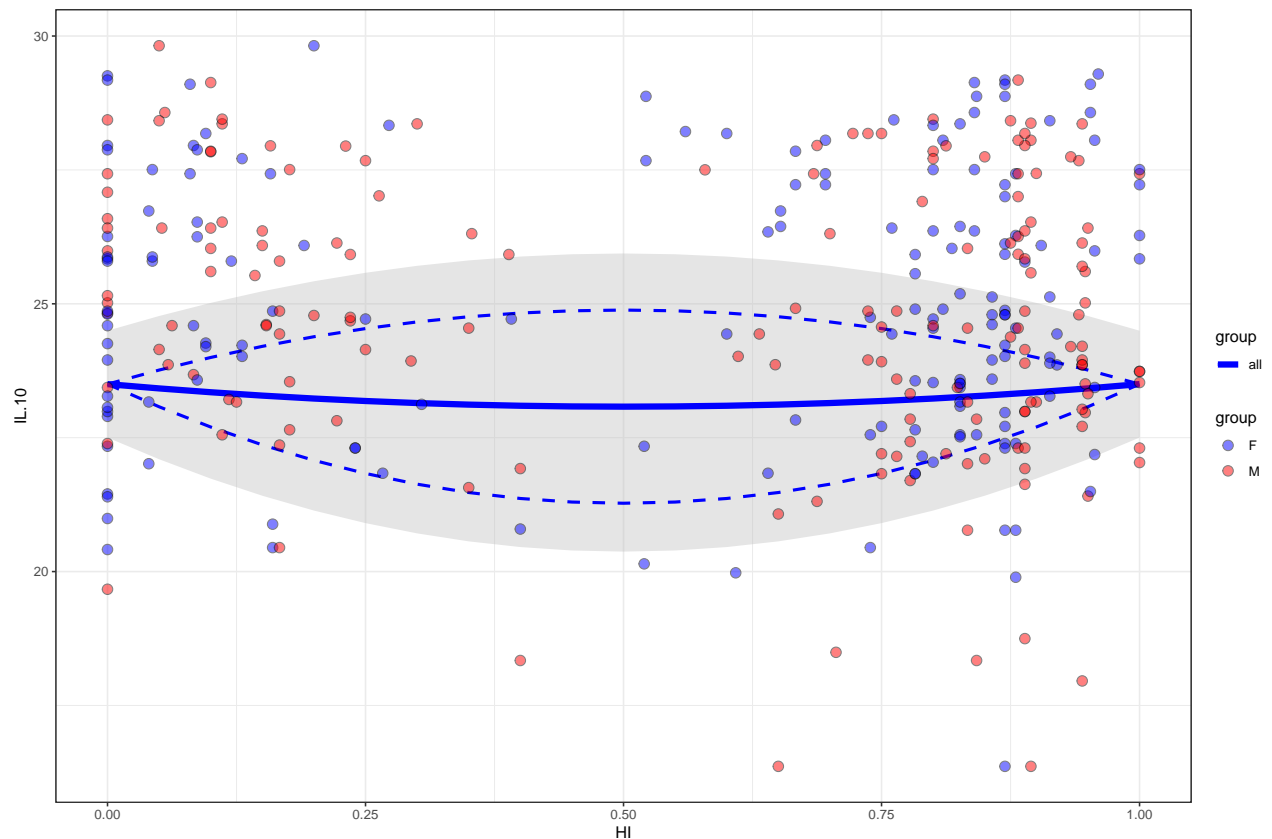
bananaPlot(mod = IL.10$H0,
           data = field,
           response = "IL.10",
           group = "Sex") +
  scale_fill_manual(values = c("blue", "red")) +
  scale_color_manual(values = c("blue", "red")) +
  theme_bw()

```

```

## 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.

```



```
IL.13 <- parasiteLoad::analyse(data = field,
                               response = "IL.13",
                               model = "weibull",
                               group = "Sex")
```

```
## [1] "Analysing data for response: IL.13"
## [1] "Fit for the response: IL.13"
## [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 0.37    1 0.3867057
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.18    1 0.5450651
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.27    1 0.4646559
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.1     1 0.6540817
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.32    1 0.4272694
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.04    1 0.7689236
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 1.07    1 0.1440993
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 1.31    3 0.4538533
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 3.23    4 0.1678807
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 2.98    2 0.05068666

##All
print(IL.13)

## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 16.5830387 0.1008805 3.1570989

```

```

##
## Log-likelihood: -1049.36
## 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
## 17.13828456 15.96431128  0.07251031  3.16085161
##
## Log-likelihood: -1048.3
## 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
## 17.1339369  0.1127506  3.2889148
##
## Log-likelihood: -523.15
## 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:

```

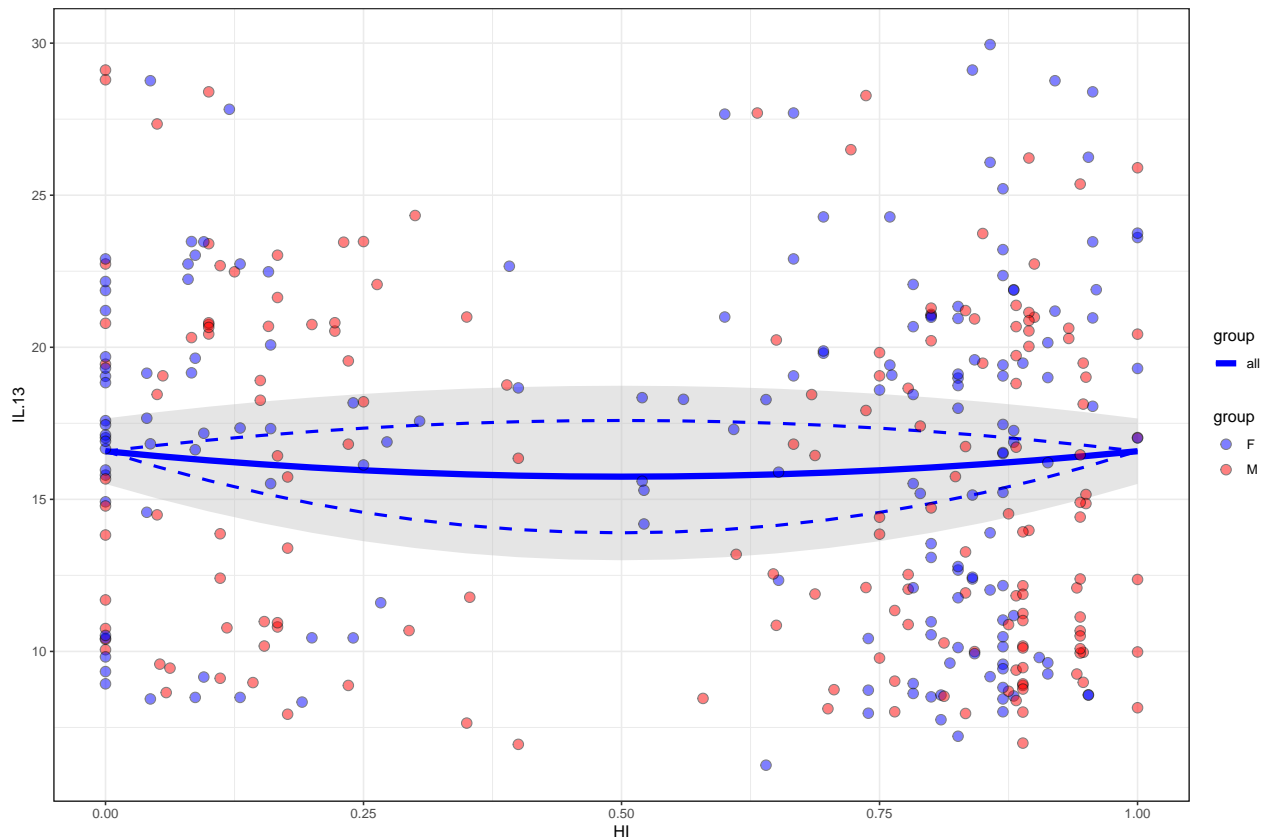
```

##           L1           alpha      myshape
## 15.99771644 0.07941324 3.05116675
##
## Log-likelihood: -524.9
## 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
## 16.9685635 17.3603225 0.1258806 3.2939699
##
## Log-likelihood: -523.09
## 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$control)
##
## Coefficients:
##           L1           L2           alpha      myshape
## 17.45548900 14.75418554 0.05245925 3.10641003
##
## Log-likelihood: -521.98
## Best method: bobyqa
bananaPlot(mod = IL.13$H0,
           data = field,
           response = "IL.13",
           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.
```



```
IL1RN <- parasiteLoad::analyse(data = field,
                               response = "IL1RN",
                               model = "weibull",
                               group = "Sex")
```

```
## [1] "Analysing data for response: IL1RN"
## [1] "Fit for the response: IL1RN"
## [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 1.02   1 0.1539407
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.89   1 0.1812934
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.78   1 0.2112293
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.29   1 0.44976
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.59   1 0.277405
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.29   1 0.4432021
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 0.02   1 0.8400309
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 1.88   3 0.2884104
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 1.94   4 0.4238122
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 0.07   2 0.9277952

##All
print(IL1RN)

## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 13.6962498 -0.1521167  3.5255931
##
## Log-likelihood: -959.99
## 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
## 13.6413104 13.7647220 -0.1465446  3.5245649
##
## Log-likelihood: -959.97
## 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
## 13.9473740 -0.1965823  3.5272765
##
## Log-likelihood: -483.92
## 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"]]),

```

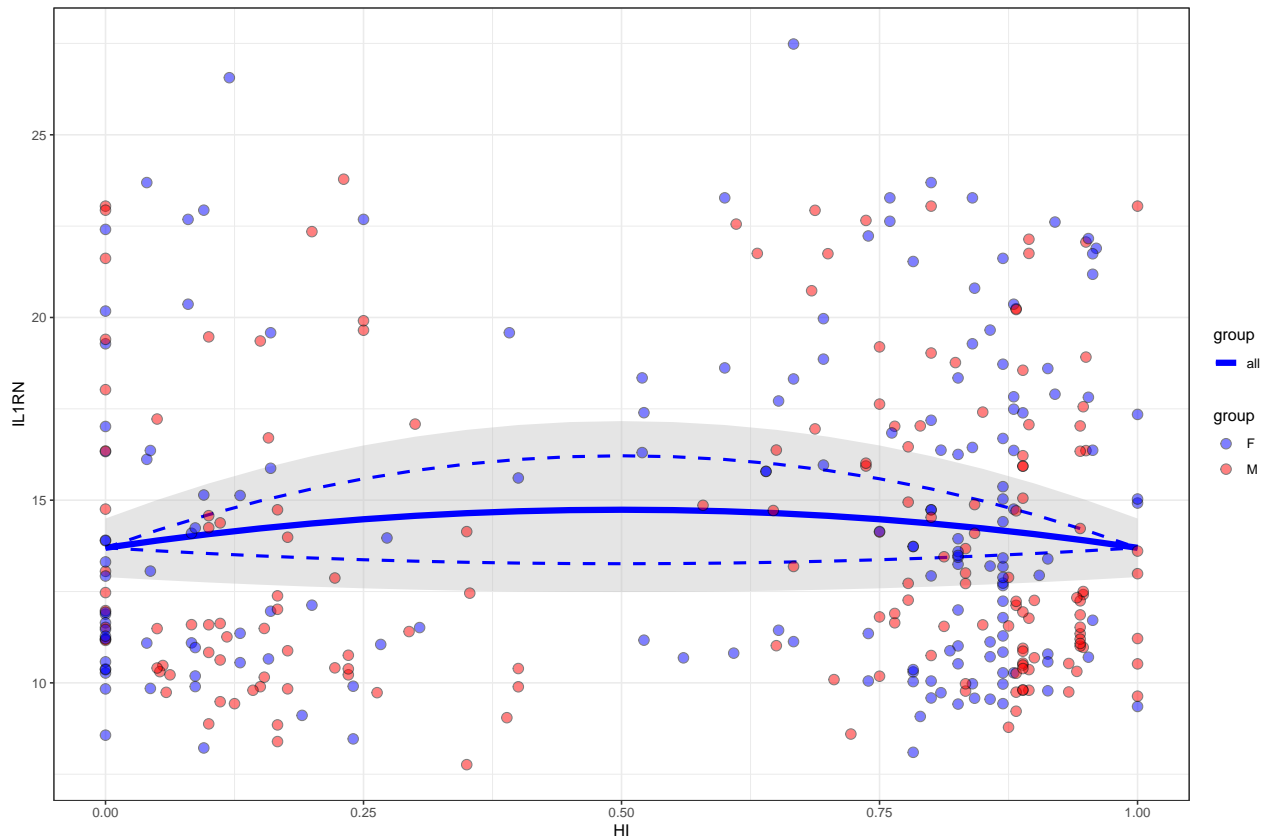
```

##      control = config$control)
##
## Coefficients:
##      L1      alpha    myshape
## 13.4426709 -0.1080279  3.5669779
##
## Log-likelihood: -474.18
## 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
## 13.8161556 14.1425746 -0.1777213  3.5249678
##
## Log-likelihood: -483.86
## Best method: L-BFGS-B
##
## $H3$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$control)
##
## Coefficients:
##      L1      L2      alpha    myshape
## 13.495226 13.388033 -0.111451  3.568166
##
## Log-likelihood: -474.18
## Best method: bobyqa
bananaPlot(mod = IL1RN$H0,
          data = field,
          response = "IL1RN",
          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.
```



```
CXCR3 <- parasiteLoad::analyse(data = field,
                               response = "CXCR3",
                               model = "weibull",
                               group = "Sex")
```

```
## [1] "Analysing data for response: CXCR3"
## [1] "Fit for the response: CXCR3"
## [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 0.14    1 0.5968162
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.1    1 0.6483795
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1  0    1 0.974345
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.31    1 0.432465
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1  0    1 0.9802984
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.27    1 0.4636948
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 0.04    1 0.7889353
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 0.19    3 0.9463332
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 0.26    4 0.97082
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 0.11    2 0.8920405

##All
print(CXCR3)

## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 21.1690733  0.0410894  5.0000000
##
## Log-likelihood: -885.46
## 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
## 21.24704448 21.07054670  0.03643424  5.00000000
##
## Log-likelihood: -885.42
## Best method: L-BFGS-B
##
## $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
## 20.910739433 -0.003525322  5.000000000
##
## Log-likelihood: -438.75
## 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)

```

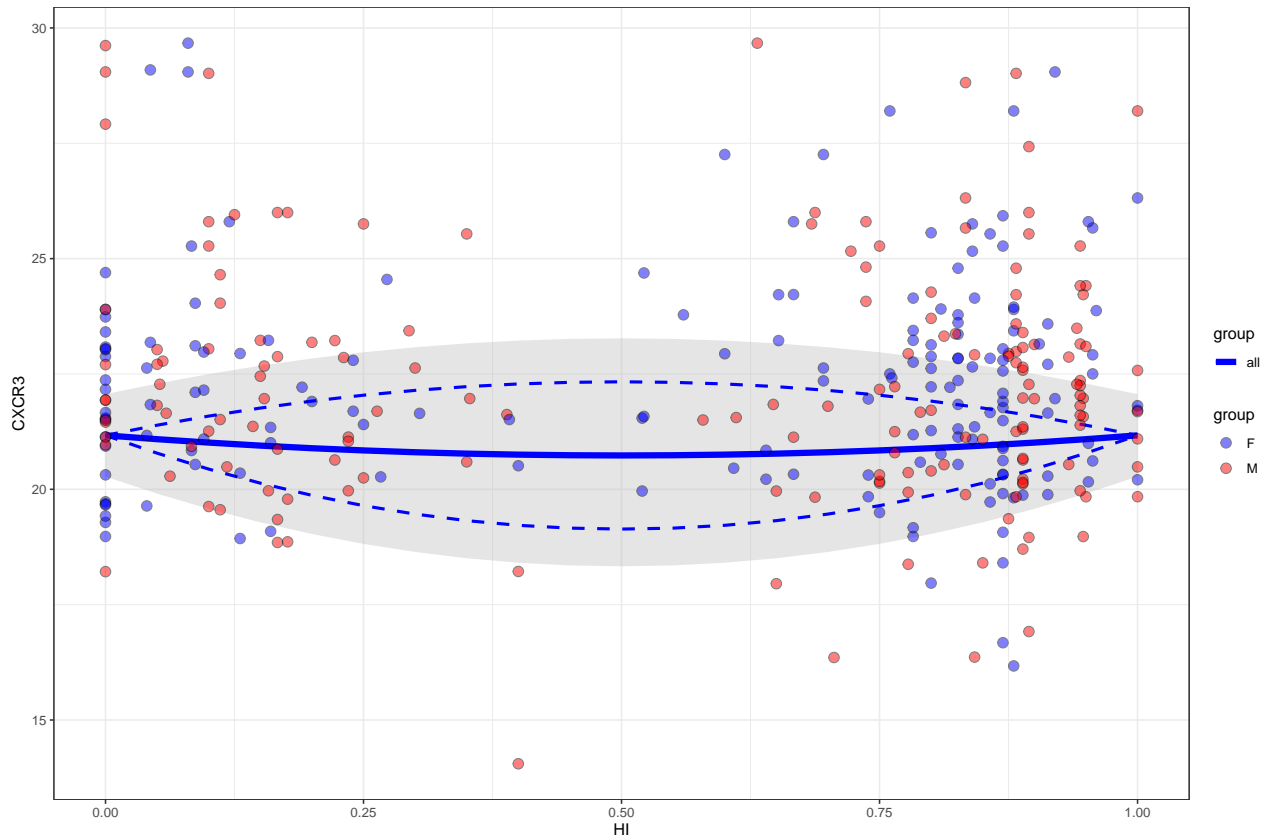
```

##
## Coefficients:
##      L1      alpha      myshape
## 21.43461412 0.08649651 5.00000000
##
## Log-likelihood: -446.53
## 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
## 20.905222427 20.920321762 -0.002885935 5.000000000
##
## Log-likelihood: -438.75
## 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$control)
##
## Coefficients:
##      L1      L2      alpha      myshape
## 21.66419446 21.22097080 0.08157629 5.00000000
##
## Log-likelihood: -446.41
## Best method: bobyqa
bananaPlot(mod = CXCR3$H0,
          data = field,
          response = "CXCR3",
          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.
```



```
CASP1 <- parasiteLoad::analyse(data = field,
                               response = "CASP1",
                               model = "weibull",
                               group = "Sex")
```

```
## [1] "Analysing data for response: CASP1"
## [1] "Fit for the response: CASP1"
## [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 0.02   1 0.8613874
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.07   1 0.7033055
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.31   1 0.4317221
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.55   1 0.2934197
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.04   1 0.7875807
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.55   1 0.2929039
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 0.69   1 0.2403023
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 0.85   3 0.6352717
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 1.21   4 0.6574302
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 1.05   2 0.3499148

##All
print(CASP1)

## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 21.51369846  0.01349478  5.00000000

```

```

##
## Log-likelihood: -888.24
## 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
## 21.13609855 21.92158161  0.02956312  5.00000000
##
## Log-likelihood: -887.55
## 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
## 20.99184039 -0.08670588  5.00000000
##
## Log-likelihood: -441.82
## 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:

```



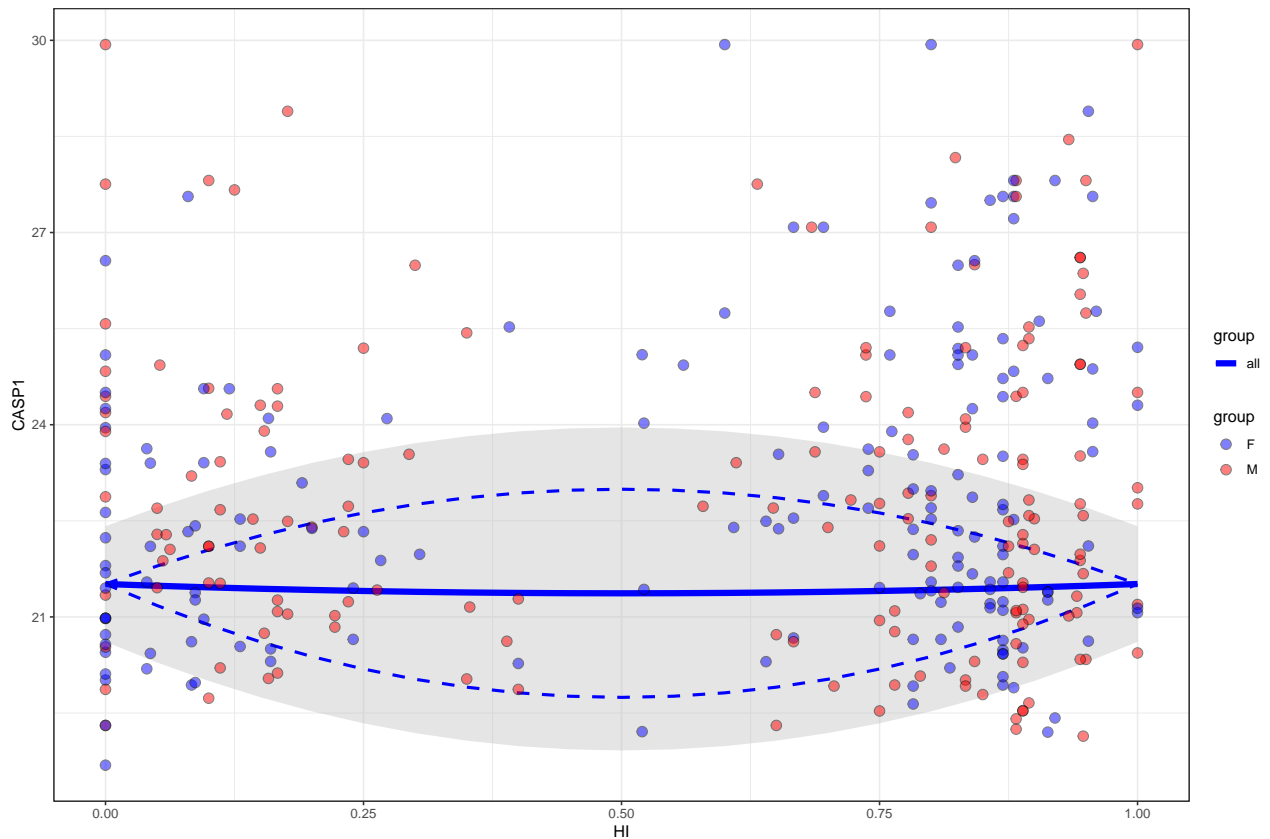
```

##      L1      alpha  myshape
## 22.04853 0.11412 5.00000
##
## Log-likelihood: -445.57
## 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
## 20.44004737 21.84537103 -0.03053329 5.00000000
##
## Log-likelihood: -440.77
## 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$control)
##
## Coefficients:
##      L1      L2      alpha  myshape
## 22.0211720 22.0695717 0.1142584 5.0000000
##
## Log-likelihood: -445.57
## Best method: L-BFGS-B
bananaPlot(mod = CASP1$H0,
           data = field,
           response = "CASP1",
           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.
```



```
CXCL9 <- parasiteLoad::analyse(data = field,
                                response = "CXCL9",
                                model = "weibull",
                                group = "Sex")
```

```
## [1] "Analysing data for response: CXCL9"
## [1] "Fit for the response: CXCL9"
## [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 0.04    1 0.7882417
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.09    1 0.6763625
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0      1 0.9572749
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.11    1 0.6428421
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.01    1 0.9011579
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.12    1 0.6215996
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 0.5     1 0.3191675
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 0.34    3 0.8787729
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 0.32    4 0.9590395
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 0.48    2 0.6213929

##All
print(CXCL9)

## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 20.09493697  0.02114074  5.00000000
##
## Log-likelihood: -907.84

```

```

## 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
## 19.79053110 20.40965300  0.03306302  5.00000000
##
## Log-likelihood: -907.35
## 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
## 20.127184346 -0.005827761  5.000000000
##
## Log-likelihood: -453.44
## 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
## 20.08120044  0.05307939  5.000000000

```

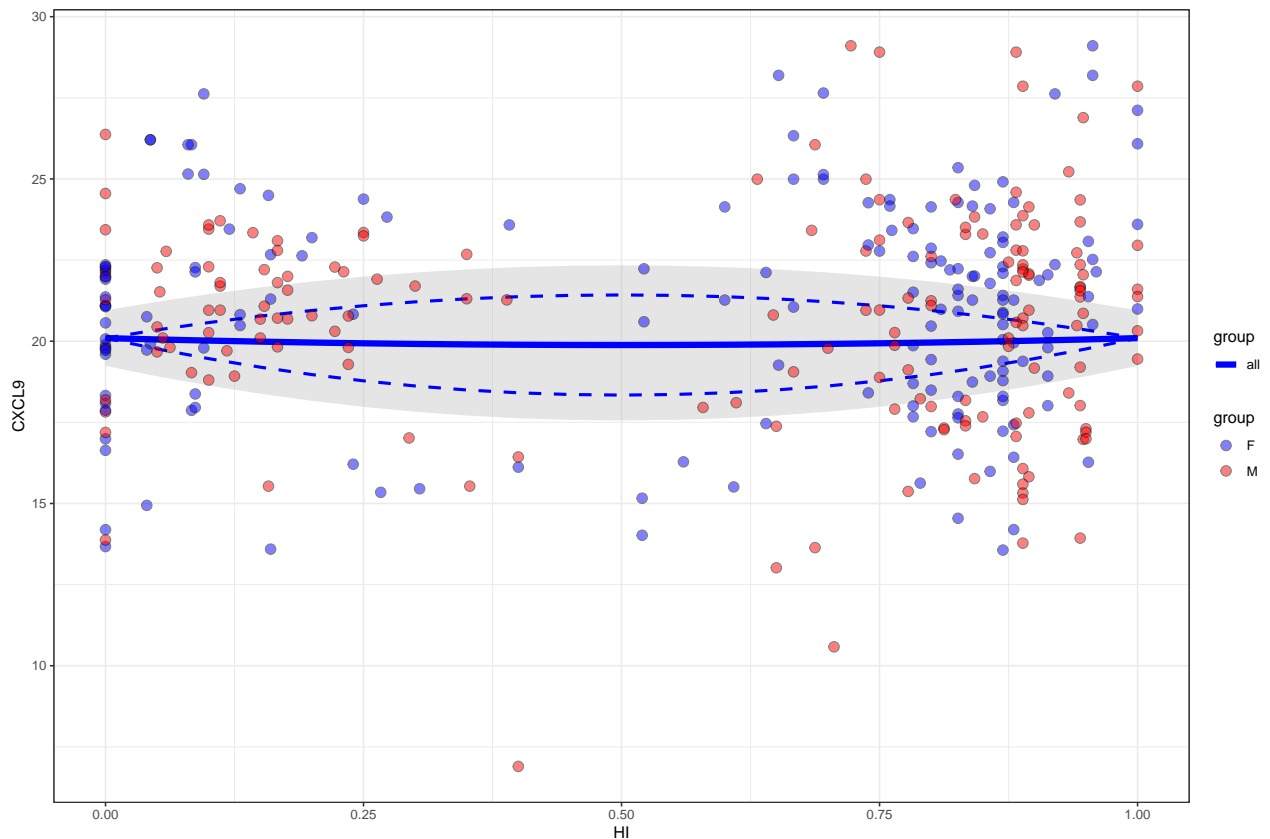
```

##
## Log-likelihood: -454.07
## 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
## 19.84795524 20.49114120  0.01385524  5.00000000
##
## Log-likelihood: -453.18
## 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$control)
##
## Coefficients:
##           L1           L2          alpha      myshape
## 19.75872453 20.33378407  0.05627667  5.00000000
##
## Log-likelihood: -453.85
## Best method: bobyqa
bananaPlot(mod = CXCL9$H0,
           data = field,
           response = "CXCL9",
           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.
```



```
ID01 <- parasiteLoad::analyse(data = field,
                               response = "ID01",
                               model = "weibull",
                               group = "Sex")
```

```
## [1] "Analysing data for response: ID01"
## [1] "Fit for the response: ID01"
## [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 0.12    1 0.6259242
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.06    1 0.7352657
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.72    1 0.2311111
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.06    1 0.724172
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.37    1 0.3910143
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.05    1 0.7560623
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 0.13    1 0.6039164
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 1.69    3 0.3368372
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 1.99    4 0.4083069
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 0.44    2 0.6461479

##All
print(ID01)

## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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

```



```

## 14.81790158 -0.04981861 3.59112899
##
## Log-likelihood: -971.56
## 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
## 14.66922289 15.00792143 -0.03552656 3.58941057
##
## Log-likelihood: -971.43
## 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
## 14.6668265 -0.1725365 3.8310148
##
## Log-likelihood: -478.76
## 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)
##

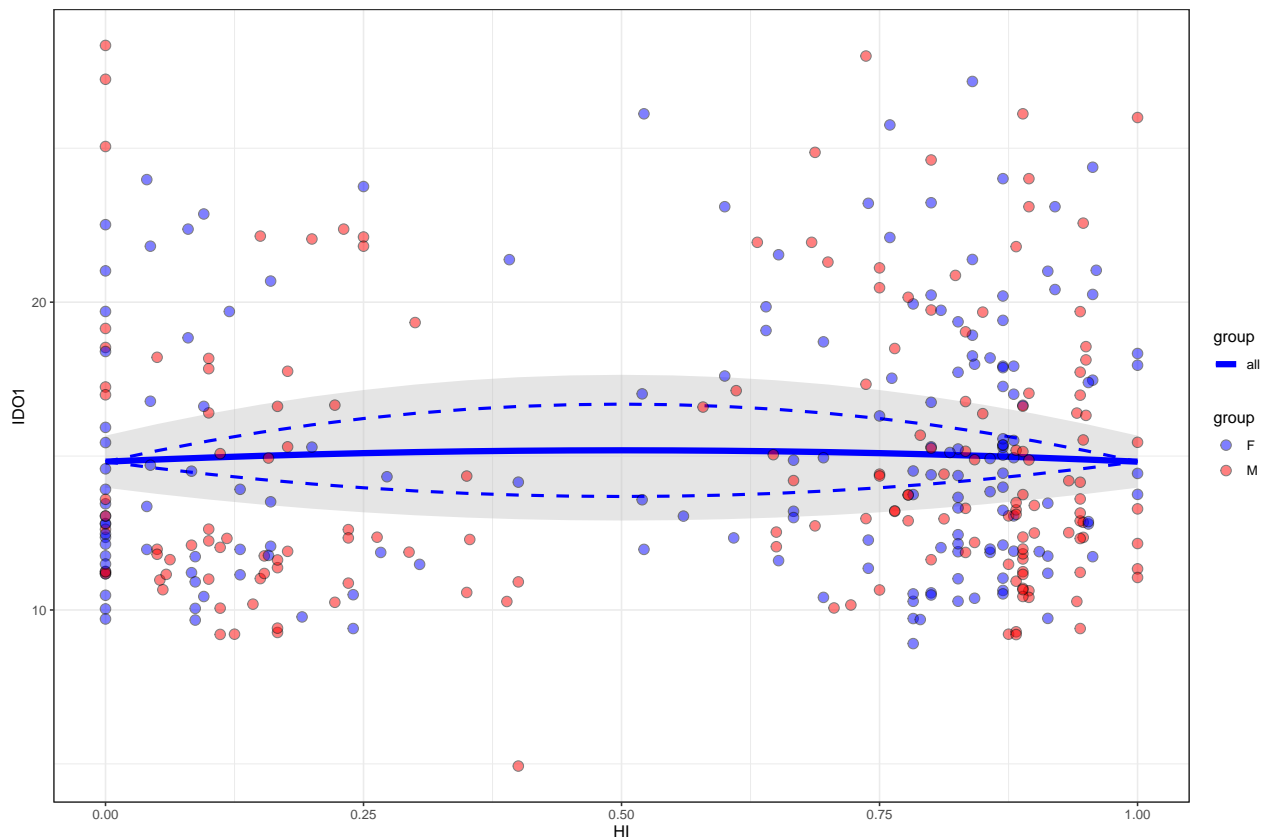
```

```

## Coefficients:
##      L1      alpha    myshape
## 14.91952835  0.05175987  3.41604254
##
## Log-likelihood: -491.11
## 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
## 14.3513581 15.1533300 -0.1273001  3.8325219
##
## Log-likelihood: -478.34
## 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$control)
##
## Coefficients:
##      L1      L2      alpha    myshape
## 15.00603485 14.82713455  0.04652613  3.41774066
##
## Log-likelihood: -491.09
## Best method: bobyqa
bananaPlot(mod = ID01$H0,
           data = field,
           response = "ID01",
           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.
```



```
IRGM1 <- parasiteLoad::analyse(data = field,
                                response = "IRGM1",
                                model = "weibull",
                                group = "Sex")
```

```
## [1] "Analysing data for response: IRGM1"
## [1] "Fit for the response: IRGM1"
## [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 0.25    1 0.4824964
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.29    1 0.4429557
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.21    1 0.5143831
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1  0     1 0.9946508
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.47    1 0.332391
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1  0     1 0.9633837
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 0.09    1 0.6691499
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 6.11    3 0.006689442
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 7.04    4 0.007066188
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 1.02    2 0.3600738

##All
print(IRGM1)

## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 11.19663862 -0.07653225  3.30954989
##
## Log-likelihood: -891.03
## 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
## 11.30449081 11.08114592 -0.08510521  3.31204995
##
## Log-likelihood: -890.94
## 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
## 11.3996701 -0.1048473  3.0986403
##
## Log-likelihood: -459.77
## Best method: L-BFGS-B
##
## $H2$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"]],

```

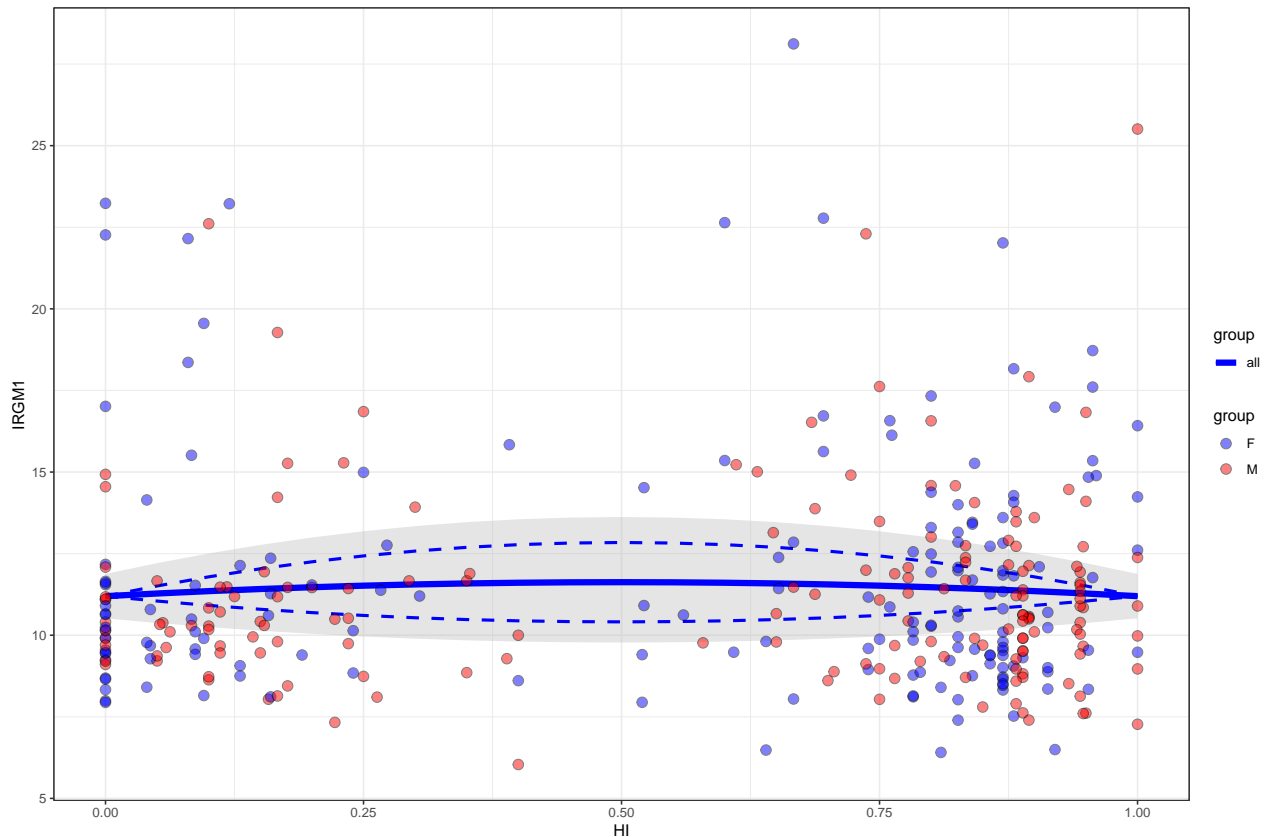
```

##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##         control = config$control)
##
## Coefficients:
##           L1           alpha           myshape
## 11.119321003  0.000966198  3.673978612
##
## Log-likelihood: -425.15
## 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
## 11.7552429 10.8759554 -0.1669563  3.1220092
##
## Log-likelihood: -459.21
## 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$control)
##
## Coefficients:
##           L1           L2           alpha           myshape
## 10.740355233 11.364936129 -0.006628154  3.687915923
##
## Log-likelihood: -424.7
## Best method: bobyqa
bananaPlot(mod = IRGM1$H0,
           data = field,
           response = "IRGM1",
           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.
```



```
MPO <- parasiteLoad::analyse(data = field,
                             response = "MPO",
                             model = "weibull",
                             group = "Sex")
```

```
## [1] "Analysing data for response: MPO"
## [1] "Fit for the response: MPO"
## [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 0.43   1 0.3553302
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.11   1 0.6392998
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.14   1 0.6009081
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.32   1 0.4206766
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1  0    1 0.9548616
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.23   1 0.4983949
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 1.58   1 0.07559372
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 0.77   3 0.6725683
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 1.14   4 0.6830476
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 1.95   2 0.1420977

##All
print(MP0)

## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 18.8929878 -0.0779708  4.5694505
##
## Log-likelihood: -980.13
## 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
## 18.38755818 19.55197808 -0.03975772  4.58953574
##
## Log-likelihood: -978.55
## 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
## 19.24079606 -0.06406299  4.59464382
##
## Log-likelihood: -491.15
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,

```

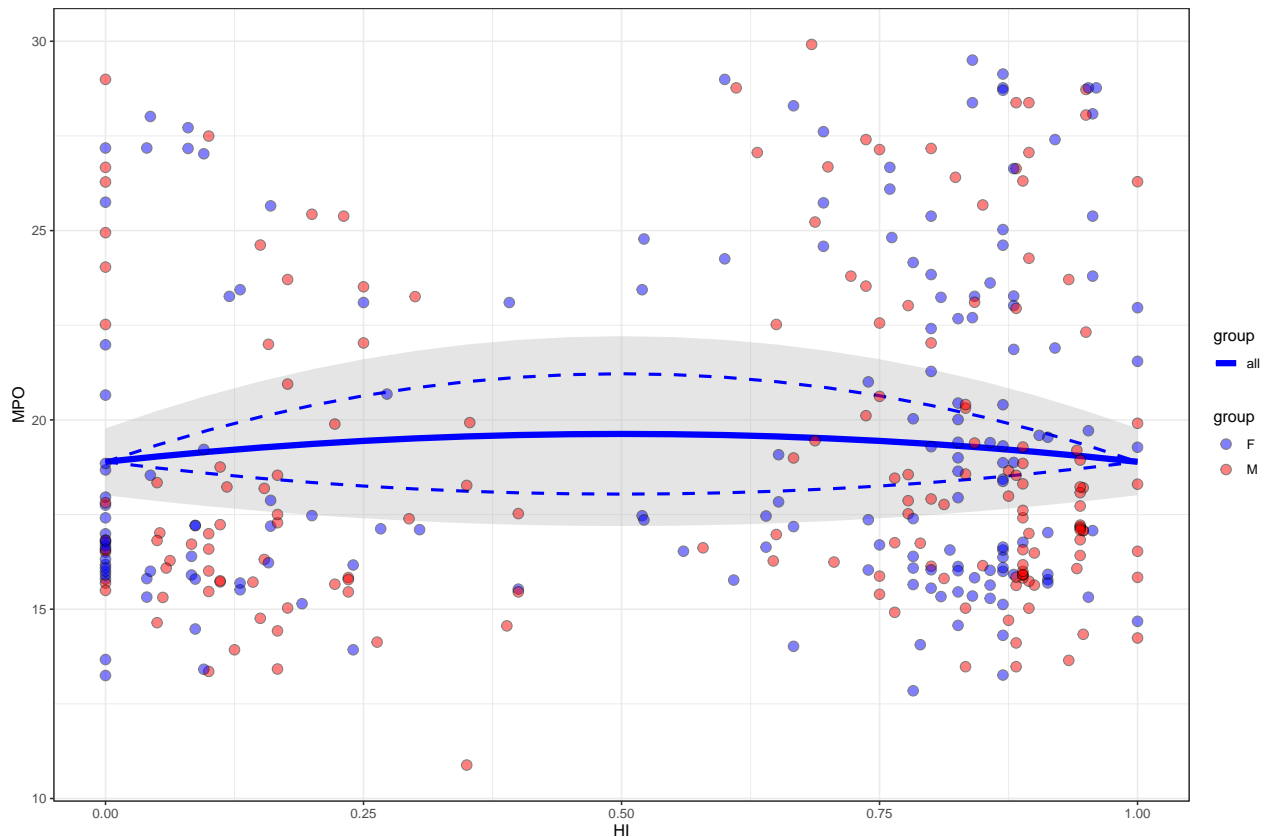
```

##      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
## 18.54266578 -0.09305452  4.56459925
##
## Log-likelihood: -488.2
## 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
## 18.532768765 20.322965617  0.006954789  4.637593898
##
## Log-likelihood: -489.4
## 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
## 18.27386610 18.84005584 -0.07876292  4.56936271
##
## Log-likelihood: -488
## Best method: bobyqa

```

```
bananaPlot(mod = MPO$H0,
           data = field,
           response = "MPO",
           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.
```



```
MUC2 <- parasiteLoad::analyse(data = field,
                             response = "MUC2",
                             model = "weibull",
                             group = "Sex")
```

```
## [1] "Analysing data for response: MUC2"
## [1] "Fit for the response: MUC2"
## [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 0.26   1 0.4705569
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.24   1 0.4898213
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1  0    1 0.9660975
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.6    1 0.273491
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1  0    1 0.9475567
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.55   1 0.2931829
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1  0    1 0.942364
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 1.29   3 0.4619501
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 1.36   4 0.606864
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 0.07   2 0.9305234
##All
print(MUC2)

```

```

## $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
## 10.15321980 -0.09796352  2.69939603
##
## Log-likelihood: -925.03
## 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
## 10.13346071 10.17587347 -0.09581511  2.69908187
##
## Log-likelihood: -925.02
## 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
## 10.62503173 -0.00799989  2.67764223
##
## Log-likelihood: -468.38
## Best method: bobyqa
##

```

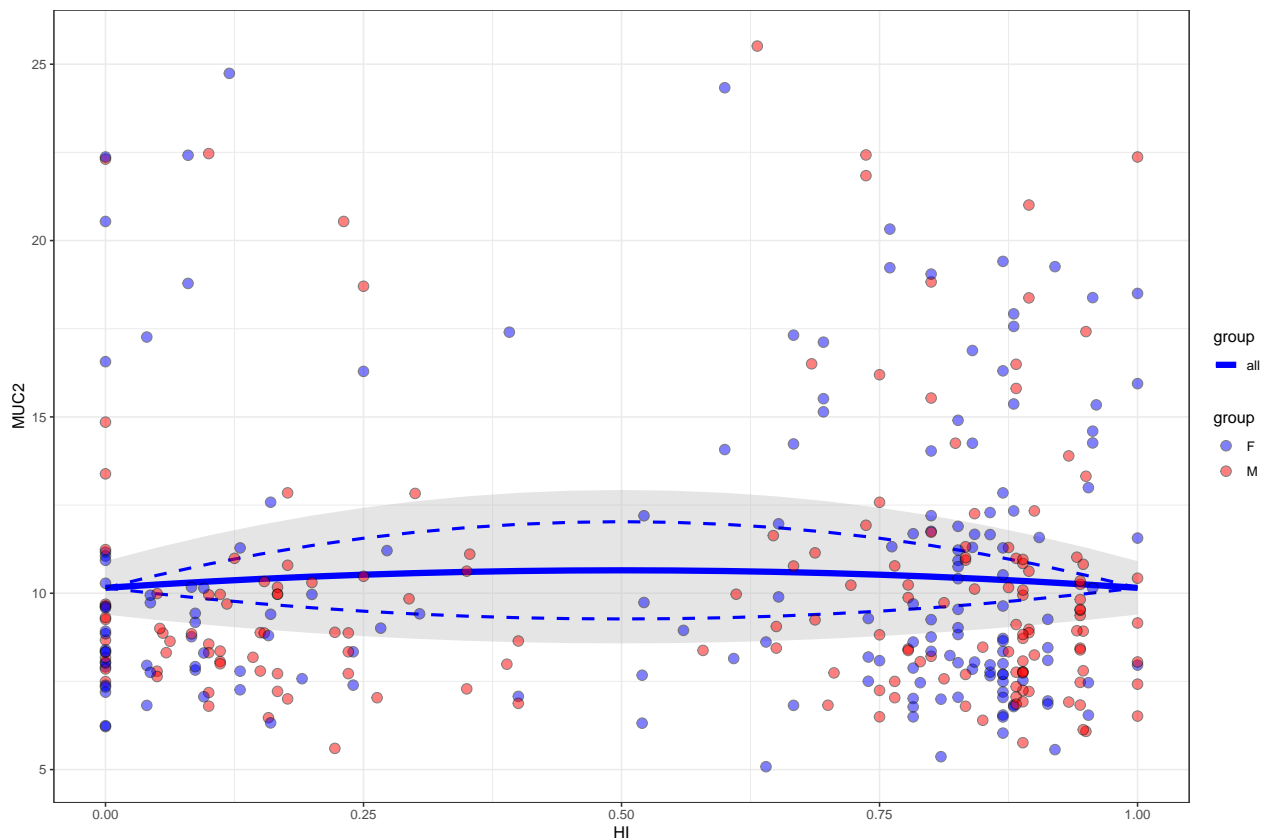
```

## $H2$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
## 9.6396173 -0.2143766  2.7380694
##
## Log-likelihood: -455.36
## 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
## 10.65738981 10.58062783 -0.01297116  2.67861166
##
## Log-likelihood: -468.38
## 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
## 9.4920606  9.7757655 -0.2060047  2.7383730

```

```
##
## Log-likelihood: -455.29
## Best method: bobyqa
bananaPlot(mod = MUC2$H0,
           data = field,
           response = "MUC2",
           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.
```



```
MUC5AC <- parasiteLoad::analyse(data = field,
                                response = "MUC5AC",
                                model = "weibull",
                                group = "Sex")
```

```
## [1] "Analysing data for response: MUC5AC"
## [1] "Fit for the response: MUC5AC"
## [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 0.07   1 0.7017173
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1   0   1 0.9710789
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.09   1 0.6655797
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.01   1 0.9182591
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1   0   1 0.9663915
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1   0   1 0.9762206
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 1.61   1 0.07306285
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 0.27   3 0.9089798
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 0.32   4 0.9586072
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 1.65   2 0.1913394

```



```

##All
print(MUC5AC)

## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 11.00011135 -0.06073295  2.48644239
##
## Log-likelihood: -973.27
## 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
## 10.430603054 11.670451820 -0.005651988  2.501298609
##
## Log-likelihood: -971.66
## 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

```

```

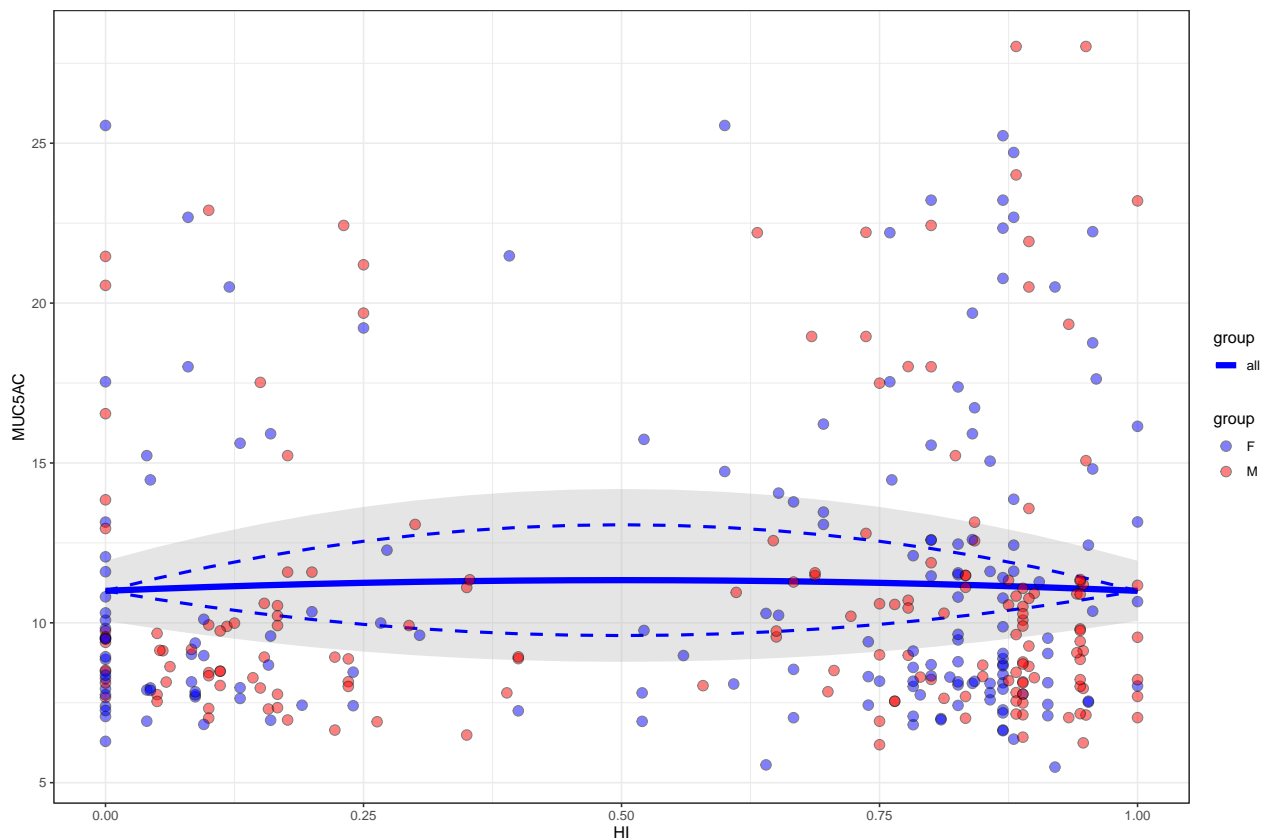
## 11.0653132 -0.0986965 2.4803643
##
## Log-likelihood: -489.19
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$method, optimizer = config$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
## 10.93512774 -0.02263148 2.49576978
##
## Log-likelihood: -483.81
## 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
## 10.460127250 12.006978371 0.009542628 2.496301231
##
## Log-likelihood: -488.04
## 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
## 10.424676839 11.384198154 -0.006464748  2.508128139
##
## Log-likelihood: -483.3
## Best method: bobyqa
```

```
bananaPlot(mod = MUC5AC$H0,
            data = field,
            response = "MUC5AC",
            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.
```



```
MYD88 <- parasiteLoad::analyse(data = field,
                                response = "MYD88",
                                model = "weibull",
                                group = "Sex")
```

```
## [1] "Analysing data for response: MYD88"
## [1] "Fit for the response: MYD88"
```

```

## [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 0.46   1 0.3389877
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.14   1 0.5902788
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.77   1 0.2140341
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1  0    1 0.9286848
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.32   1 0.4262514
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1  0    1 0.9611359
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 1.93   1 0.04956749
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 1.31   3 0.4543981
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 1.4    4 0.5923446

```

```
## [1] "Testing H3 vs H2"
##      dLL dDF   pvalue
## 1 2.02   2 0.132962
```

```
##All
```

```
print(MYD88)
```

```
## $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
## 16.3248706 -0.1332213  3.1642223
```

```
##
```

```
## Log-likelihood: -1067.93
```

```
## 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
## 15.60479123 17.19981597 -0.07464472  3.16485661
```

```
##
```

```
## Log-likelihood: -1066
```

```
## 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)
```

```

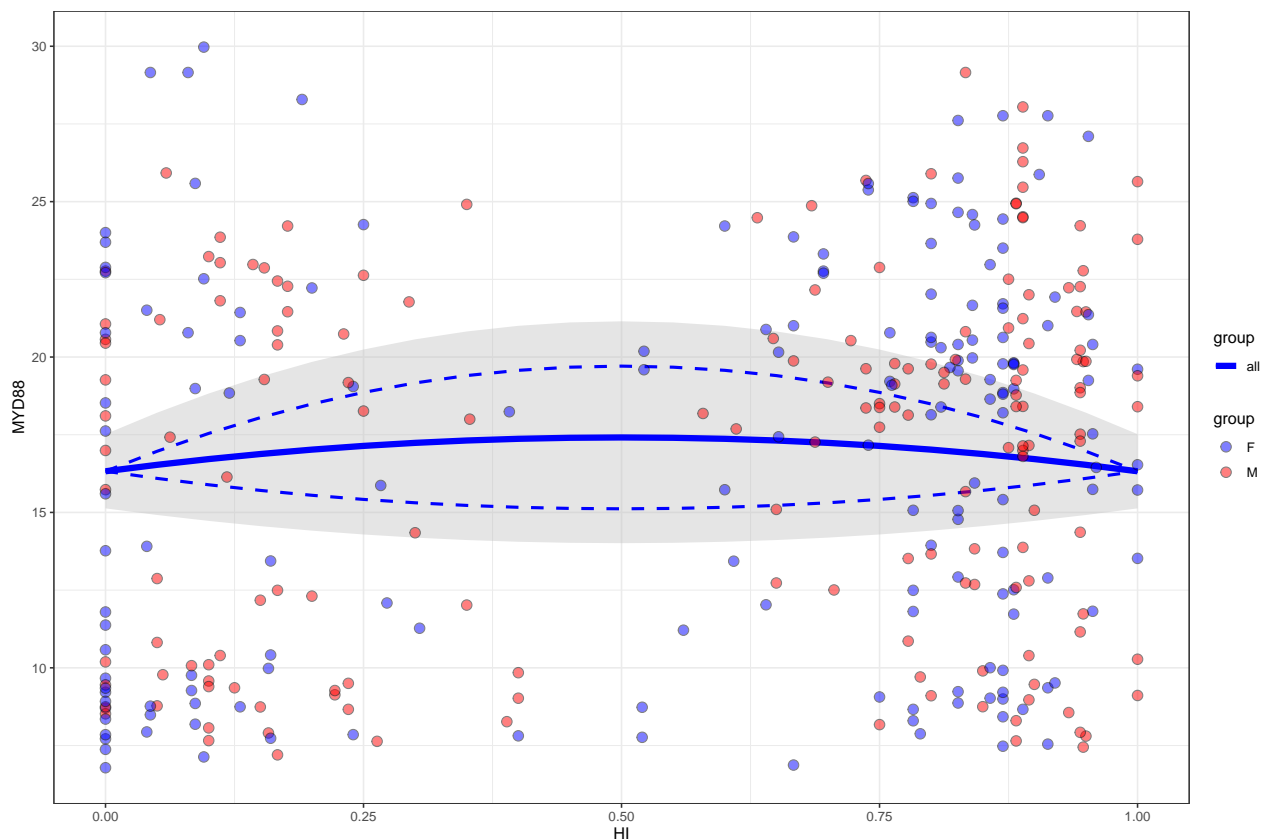
##
## Coefficients:
##      L1      alpha    myshape
## 15.7747180 -0.2685204  2.9916236
##
## Log-likelihood: -538.69
## Best method: L-BFGS-B
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      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
## 16.84217958 -0.01595002  3.37353617
##
## Log-likelihood: -527.93
## 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
## 15.3974288 16.5570754 -0.1813072  2.9837423
##
## Log-likelihood: -538.26
## 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
## 15.744768364 17.663805330 -0.008490156  3.396399982
##
## Log-likelihood: -526.34
## Best method: bobyqa
```

```
bananaPlot(mod = MYD88$H0,
           data = field,
           response = "MYD88",
           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.
```



```
NCR1 <- parasiteLoad::analyse(data = field,
                             response = "NCR1",
                             model = "weibull",
```

```

group = "Sex")

## [1] "Analysing data for response: NCR1"
## [1] "Fit for the response: NCR1"
## [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 0.25    1 0.4768117
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.17    1 0.5640037
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.12    1 0.6265969
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.13    1 0.6039129
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.09    1 0.6684493
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.11    1 0.6392631
## [1] "Testing H1 vs H0"
##      dLL dDF    pvalue
## 1 0.12    1 0.6208204
## [1] "Testing H2 vs H0"
##      dLL dDF    pvalue
## 1 0.04    3 0.9934031
## [1] "Testing H3 vs H1"
##      dLL dDF    pvalue
## 1 0.13    4 0.9925998
## [1] "Testing H3 vs H2"

```

```
##      dLL dDF      pvalue
## 1 0.21    2 0.8138524
```

```
##All
```

```
print(NCR1)
```

```
## $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
```

```
## 23.27700633 0.05537369 5.00000000
```

```
##
```

```
## Log-likelihood: -899.75
```

```
## 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
```

```
## 23.43303122 23.07422885 0.04645388 5.00000000
```

```
##
```

```
## Log-likelihood: -899.62
```

```
## 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)
```

```
##
```

```

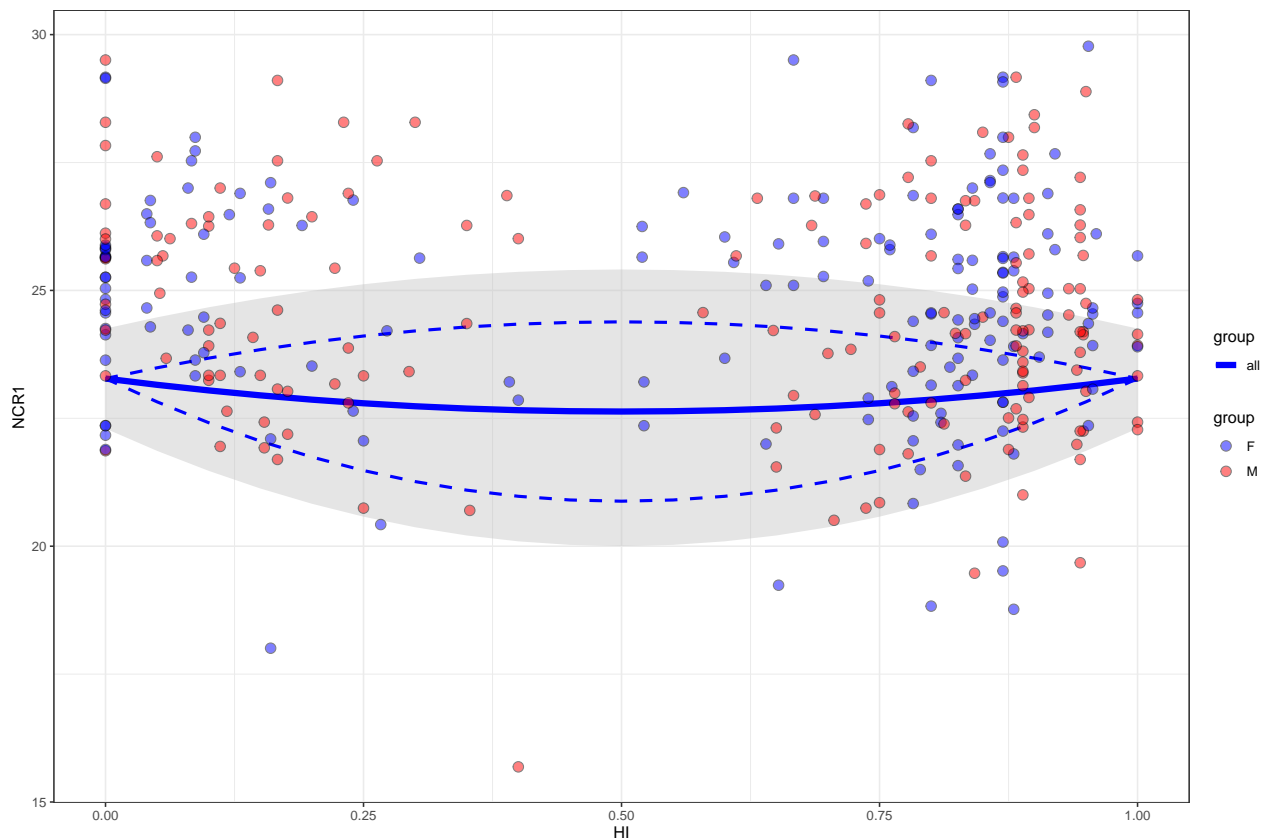
## Coefficients:
##      L1      alpha    myshape
## 23.33275536 0.05208696 5.00000000
##
## Log-likelihood: -448.49
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      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
## 23.22203167 0.05888617 5.00000000
##
## Log-likelihood: -451.21
## 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
## 23.35566151 23.29088212 0.04953958 5.00000000
##
## Log-likelihood: -448.49
## Best method: L-BFGS-B
##
## $H3$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
## 23.56136780 22.92055292  0.05382362  5.00000000
##
## Log-likelihood: -451.01
## Best method: bobyqa
```

```
bananaPlot(mod = NCR1$H0,
           data = field,
           response = "NCR1",
           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.
```



```
PRF1 <- parasiteLoad::analyse(data = field,
                             response = "PRF1",
                             model = "weibull",
                             group = "Sex")
```

```

## [1] "Analysing data for response: PRF1"
## [1] "Fit for the response: PRF1"
## [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 0.23    1 0.4973722
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.14    1 0.5987007
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.32    1 0.4221251
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.01    1 0.9005308
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.21    1 0.5127973
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.01    1 0.9079084
## [1] "Testing H1 vs H0"
##      dLL dDF    pvalue
## 1 0.25    1 0.4801175
## [1] "Testing H2 vs H0"
##      dLL dDF    pvalue
## 1 0.12    3 0.9718372
## [1] "Testing H3 vs H1"
##      dLL dDF    pvalue
## 1 0.1     4 0.9952185
## [1] "Testing H3 vs H2"
##      dLL dDF    pvalue
## 1 0.23    2 0.7920118

```

```

##All
print(PRF1)

## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 23.0571958 0.0518239 5.0000000
##
## Log-likelihood: -901.29
## 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
## 23.28648383 22.78332371 0.04126848 5.00000000
##
## Log-likelihood: -901.04
## 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

```

```

## 23.26863040  0.08321886  5.00000000
##
## Log-likelihood: -448.38
## 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
## 22.81170048  0.01415199  5.00000000
##
## Log-likelihood: -452.79
## 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
## 23.37124801 23.09162055  0.07306736  5.00000000
##
## Log-likelihood: -448.35
## 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"]]),

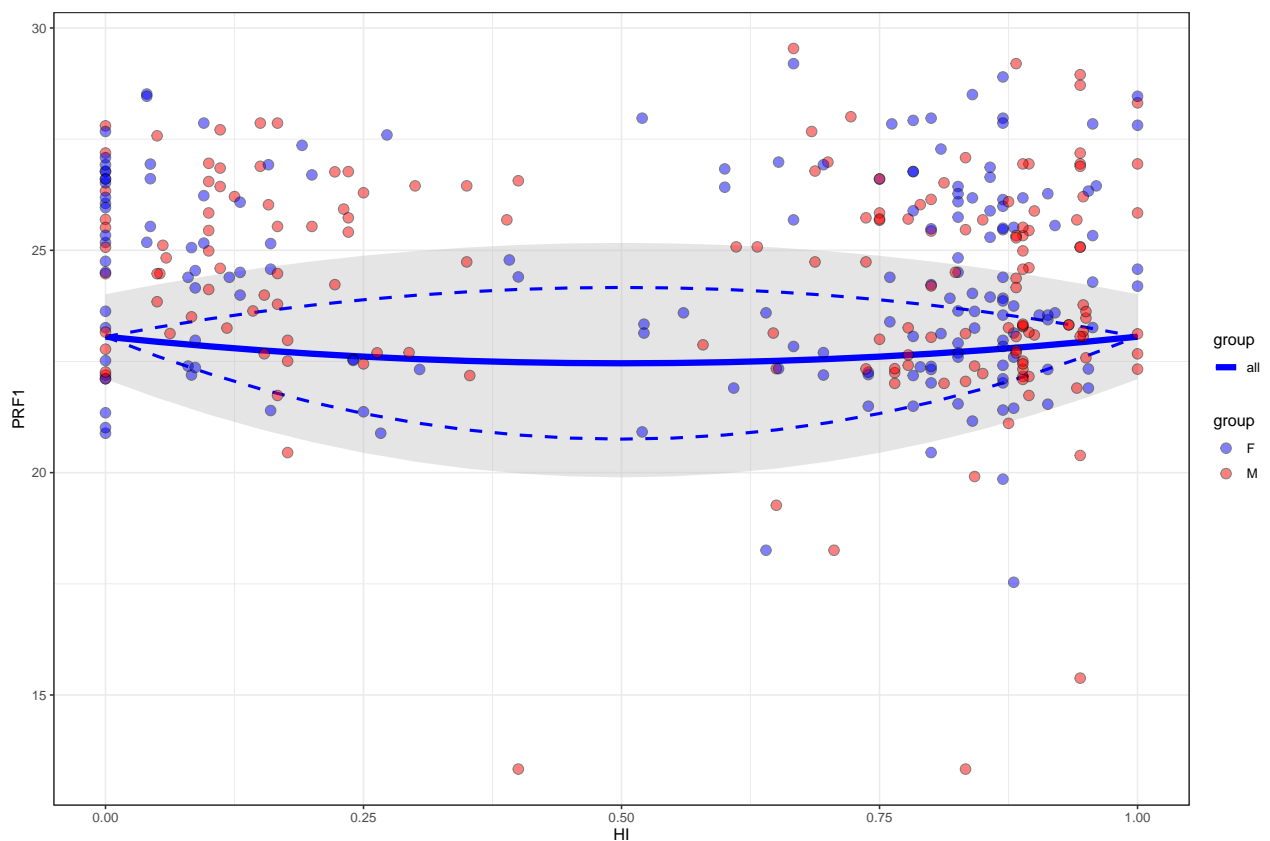
```



```
## control = config$control)
##
## Coefficients:
##      L1      L2      alpha    myshape
## 23.16156016 22.54154197 0.01314789 5.00000000
##
## Log-likelihood: -452.59
## Best method: bobyqa
```

```
bananaPlot(mod = PRF1$H0,
            data = field,
            response = "PRF1",
            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.
```



```
RETNLB <- parasiteLoad::analyse(data = field,
                                response = "RETNLB",
                                model = "weibull",
                                group = "Sex")
```

```
## [1] "Analysing data for response: RETNLB"
## [1] "Fit for the response: RETNLB"
```

```

## [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 0.53    1 0.3043467
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.21    1 0.5187508
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.4     1 0.3724035
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.15    1 0.5786035
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.03    1 0.795828
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.13    1 0.6058367
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 1.54    1 0.07927098
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
## 1 2.84    3 0.1281921
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 3.2     4 0.1716212

```

```
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 1.9    2 0.1501587
```

```
##All
```

```
print(RETNLB)
```

```
## $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
## 11.3900853 -0.1449763  2.7334993
```

```
##
```

```
## Log-likelihood: -966.62
```

```
## 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
## 10.8700153 12.0197805 -0.0905373  2.7474637
```

```
##
```

```
## Log-likelihood: -965.08
```

```
## 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)
```

```

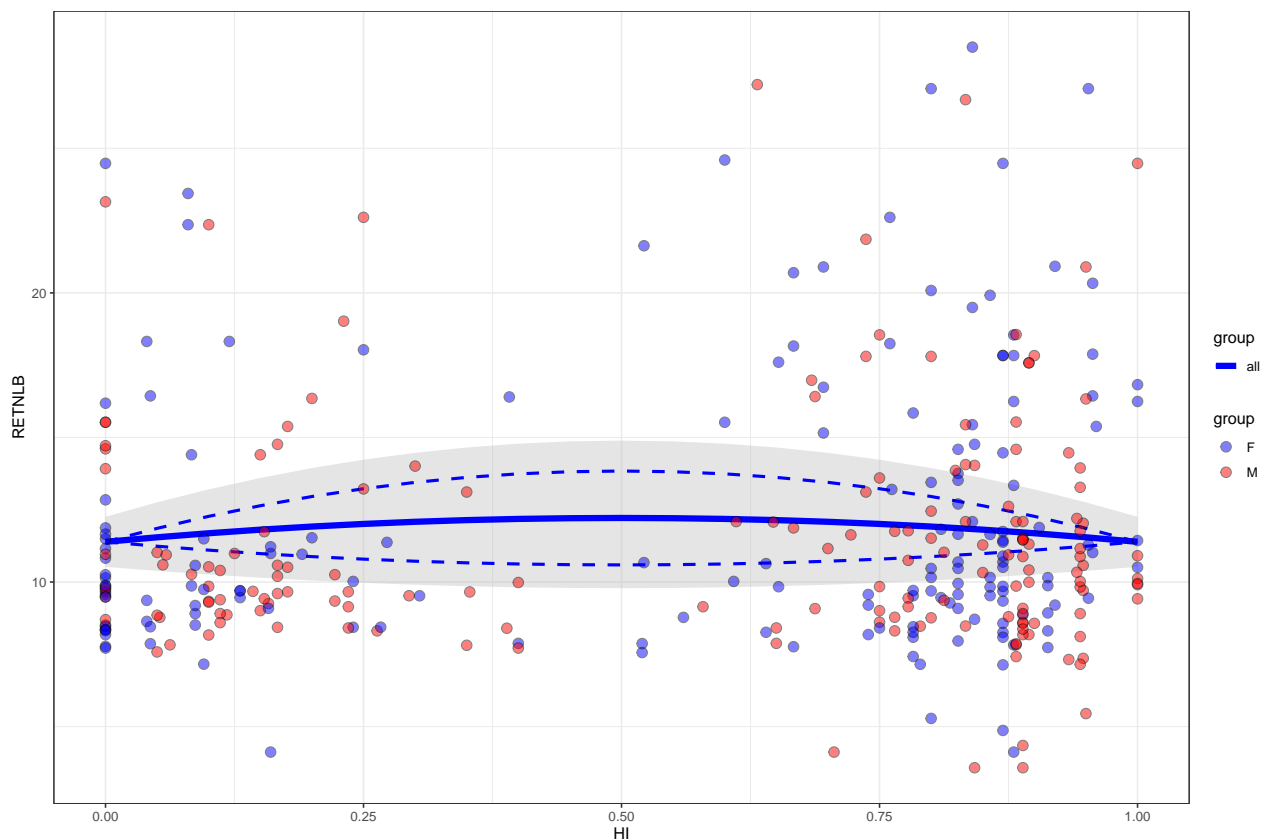
##
## Coefficients:
##      L1      alpha    myshape
## 11.6547765 -0.1885339  2.6477369
##
## Log-likelihood: -492.66
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      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
## 11.1162663 -0.1024407  2.8716587
##
## Log-likelihood: -471.12
## 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
## 10.90892814 12.83908997 -0.05428917  2.67546844
##
## Log-likelihood: -490.86
## 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
## 10.91661606 11.29472900 -0.09492913  2.87433575
##
## Log-likelihood: -471.02
## Best method: bobyqa
```

```
bananaPlot(mod = RETNLB$H0,
           data = field,
           response = "RETNLB",
           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.
```



```
SOCS1 <- parasiteLoad::analyse(data = field,
                               response = "SOCS1",
                               model = "weibull",
```

```
group = "Sex")
```

```
## [1] "Analysing data for response: SOCS1"
## [1] "Fit for the response: SOCS1"
## [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 0.08    1 0.6837654
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.06    1 0.7331601
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.52    1 0.3057468
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.39    1 0.3770379
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.26    1 0.4738698
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF      pvalue
## 1 0.42    1 0.3608173
## [1] "Testing H1 vs H0"
##      dLL dDF      pvalue
## 1 0.04    1 0.7685942
## [1] "Testing H2 vs H0"
##      dLL dDF      pvalue
```

```

## 1 3.77    3 0.05658546
## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 4.77    4 0.04892391
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 1.04    2 0.3519082

##All
print(SOCS1)

## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 12.48886179  0.03969697  3.57238951
##
## Log-likelihood: -895.99
## 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
## 12.5660220 12.4059954  0.0339433  3.5756682
##
## Log-likelihood: -895.95
## 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
## 13.0320597  0.1379951  3.4402765
##
## Log-likelihood: -457.2
## 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
## 11.7826291 -0.1259445  3.7892913
##
## Log-likelihood: -435.02
## 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
## 13.3254259 12.6389598  0.1025675  3.4633546
##
## Log-likelihood: -456.88
## Best method: bobyqa
##
## $H3$groupB
##
## Call:

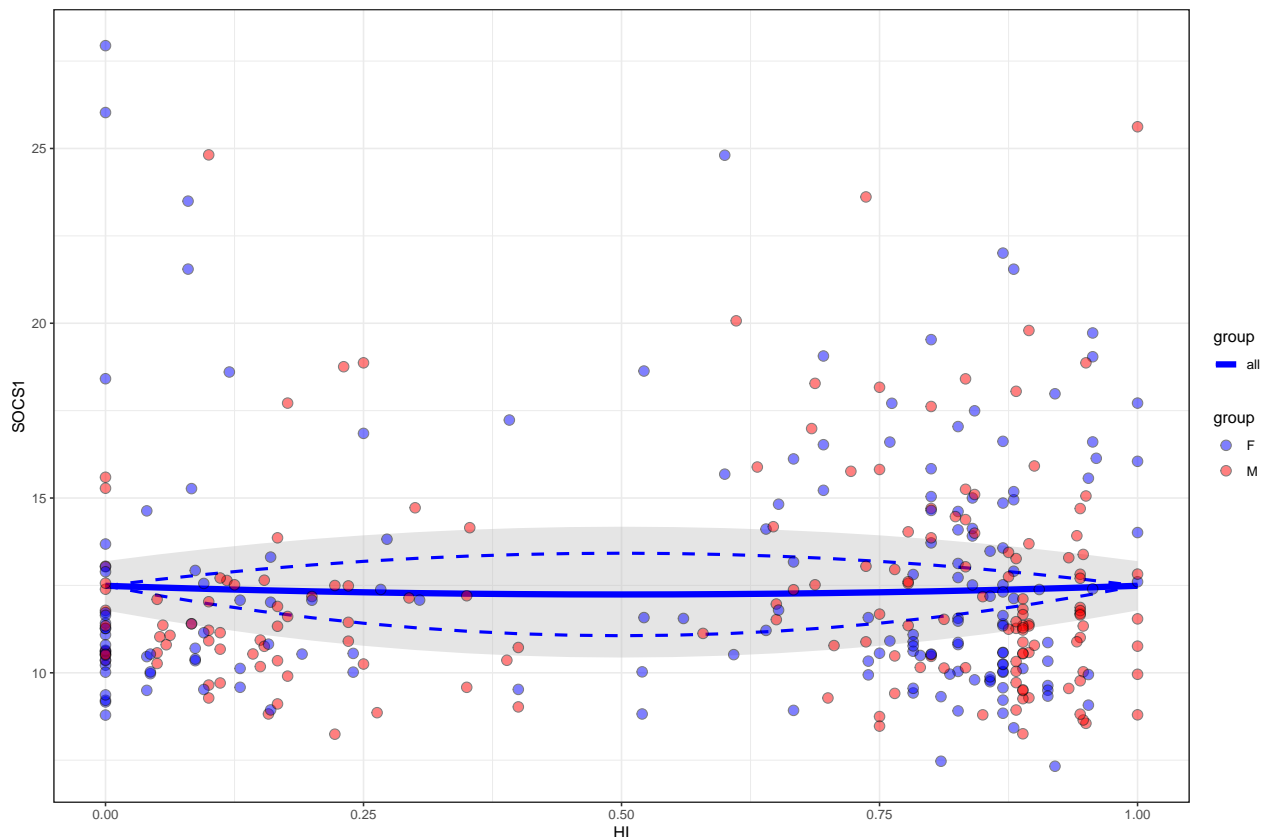
```



```
## 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
## 11.3052367 12.1141714 -0.1297315  3.8074983
##
## Log-likelihood: -434.3
## Best method: bobyqa
```

```
bananaPlot(mod = SOCS1$H0,
           data = field,
           response = "SOCS1",
           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.
```



```

TICAM1 <- parasiteLoad::analyse(data = field,
                                response = "TICAM1",
                                model = "weibull",
                                group = "Sex")

## [1] "Analysing data for response: TICAM1"
## [1] "Fit for the response: TICAM1"
## [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 0.29    1 0.4478463
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.31    1 0.4311544
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.06    1 0.7236212
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.26    1 0.4727852
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.11    1 0.643292
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.26    1 0.4742987
## [1] "Testing H1 vs H0"
##      dLL dDF    pvalue
## 1 0.03    1 0.8153707
## [1] "Testing H2 vs H0"
##      dLL dDF    pvalue
## 1 0.04    3 0.9946535

```

```

## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 0.1   4 0.9954505
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 0.09  2 0.9154868

##All
print(TICAM1)

## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 21.07394018  0.06239457  5.00000000
##
## Log-likelihood: -901.9
## 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
## 21.00180296 21.15377529  0.06549594  5.00000000
##
## Log-likelihood: -901.88
## 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"]],

```

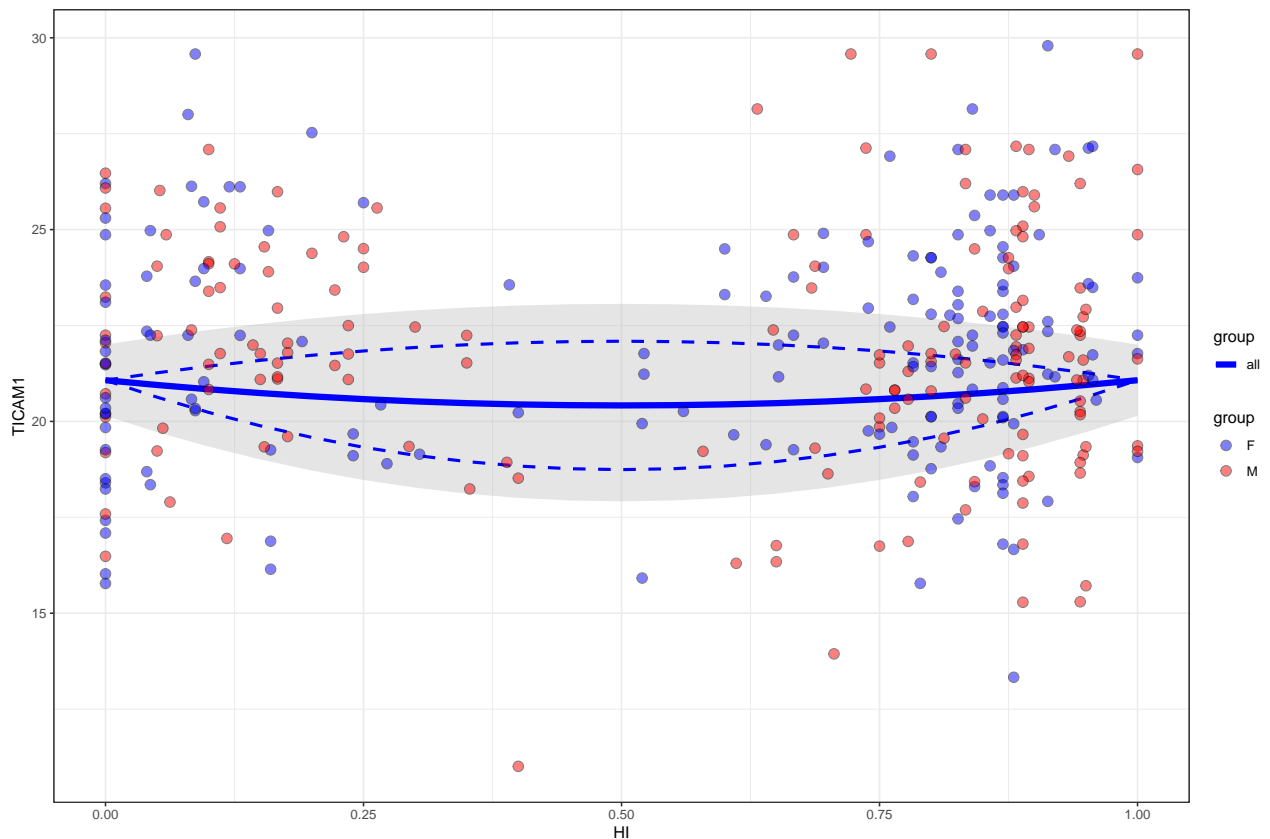
```

##      myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##      alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##      control = config$control)
##
## Coefficients:
##      L1      alpha    myshape
## 20.94781248  0.04141839  5.00000000
##
## Log-likelihood: -447.9
## 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
## 21.19891582  0.08290944  5.00000000
##
## Log-likelihood: -453.96
## 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
## 20.80207919 21.18724427  0.05669406  5.00000000
##
## Log-likelihood: -447.82
## 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$control)
##
## Coefficients:
##          L1          L2          alpha      myshape
## 21.25628492 21.15526894  0.08269444  5.00000000
##
## Log-likelihood: -453.96
## Best method: bobyqa
bananaPlot(mod = TICAM1$H0,
           data = field,
           response = "TICAM1",
           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.
```



```

TNF <- parasiteLoad::analyse(data = field,
                             response = "TNF",
                             model = "weibull",
                             group = "Sex")

## [1] "Analysing data for response: TNF"
## [1] "Fit for the response: TNF"
## [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 0.02    1 0.8611049
## [1] "Testing H1 no alpha vs alpha"
##      dLL dDF    pvalue
## 1    0    1 0.9749966
## [1] "Testing H2 groupA no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.07    1 0.7054059
## [1] "Testing H2 groupB no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.19    1 0.5332046
## [1] "Testing H3 groupA no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.04    1 0.7696155
## [1] "Testing H3 groupB no alpha vs alpha"
##      dLL dDF    pvalue
## 1 0.14    1 0.5925895
## [1] "Testing H1 vs H0"
##      dLL dDF    pvalue
## 1 0.2     1 0.5251758
## [1] "Testing H2 vs H0"
##      dLL dDF    pvalue
## 1 0.27    3 0.9090064

```



```

## [1] "Testing H3 vs H1"
##      dLL dDF      pvalue
## 1 0.76   4 0.8222382
## [1] "Testing H3 vs H2"
##      dLL dDF      pvalue
## 1 0.69   2 0.5005737

##All
print(TNF)

## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##      scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##      start = start, method = config$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
## 20.80863020  0.01350267  5.00000000
##
## Log-likelihood: -896.07
## 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
## 20.989587380 20.578828188  0.002497248  5.000000000
##
## Log-likelihood: -895.87
## 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
## 20.58943265 -0.04230928  5.00000000
##
## Log-likelihood: -443.24
## 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
## 21.01654488  0.06648745  5.00000000
##
## Log-likelihood: -452.56
## 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
## 20.53577940 20.68886287 -0.03512274  5.00000000
##
## Log-likelihood: -443.22
## 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$control)
##
## Coefficients:
##           L1           L2          alpha      myshape
## 21.57257783 20.51752253  0.05799377  5.00000000
##
## Log-likelihood: -451.88
## Best method: bobyqa
bananaPlot(mod = TNF$H0,
           data = field,
           response = "TNF",
           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.
```

