5.Gene_expression_analysis

Fay

2022-08-09

load libraries

```
library(ggplot2)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v tibble  3.1.8     v dplyr   1.0.9
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
## v purrr   0.3.4
```

```
## -- 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, inlcuding imputed data

```
lab <- read.csv("output_data/gene_expression/data_products/lab_imputed_gene_expression.csv")
field <- read.csv("output_data/gene_expression/data_products/field_imputed_gene_expression.csv")


field <- field %>% filter(MC.Eimeria == "TRUE")
```

Selecting genes

```
# vectors for selecting gene columns
Genes_lab <- c("IFNy", "CXCR3", "IL.6", "IL.10", "IL.13", "IL1RN", "CASP1",
               "CXCL9", "IDO1", "IRGM1", "MPO", "MUC2", "MUC5AC",
           "MYD88", "NCR1", "PRF1", "RETNLB", "SOCS1", "TICAM1", "TNF")

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

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

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

Install the package

```
##
## * checking for file '/tmp/RtmpuIaAeA/remotes3463dd2850a7f/alicebalard-parasiteLoad-1b43216/DESCRIPTI
## * 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$IDO1



# 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
##   14.7081383    3.7764064
##  ( 0.3937176) ( 0.2784004)
##
## $loglik
## [1] -252.7894
##
## $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
```

```
## $fit
##      shape        scale
##    4.0902380   16.1639498
##  ( 0.3132009) ( 0.4364843)
##
## $loglik
## [1] -254.8264
##
## $AIC
## NULL
```

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

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

```
## $AIC
## [1] "no aic computed"
```

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

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

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"
## [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"
```

```
## 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"
```

```
## 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"
## [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 0.51   1 0.3140693
## [1] "Testing H1 no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.5   1 0.3150261
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.41   1 0.3674579
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF     pvalue
## 1 3.23   1 0.01102272
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.41   1 0.3646888
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF     pvalue
## 1 3.25   1 0.01083619
## [1] "Testing H1 vs H0"
##   dLL dDF    pvalue
## 1   0   1 0.9898643
## [1] "Testing H2 vs H0"
##    dLL dDF   pvalue
## 1 5.09   3 0.017123
## [1] "Testing H3 vs H1"
##    dLL dDF     pvalue
## 1 5.12   4 0.03645614
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.04   2 0.9650189
```

```
##All
print(IFNy)
```

```
## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$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
## 20.6227445  0.1590249  4.9682096
##
## Log-likelihood: -267.78
## 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
## 20.6077462 20.6284759  0.1591186  4.9680389
##
## Log-likelihood: -267.78
## 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
## 19.5349115 -0.2573751  5.0000000
##
## Log-likelihood: -106.63
## 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"]],
```

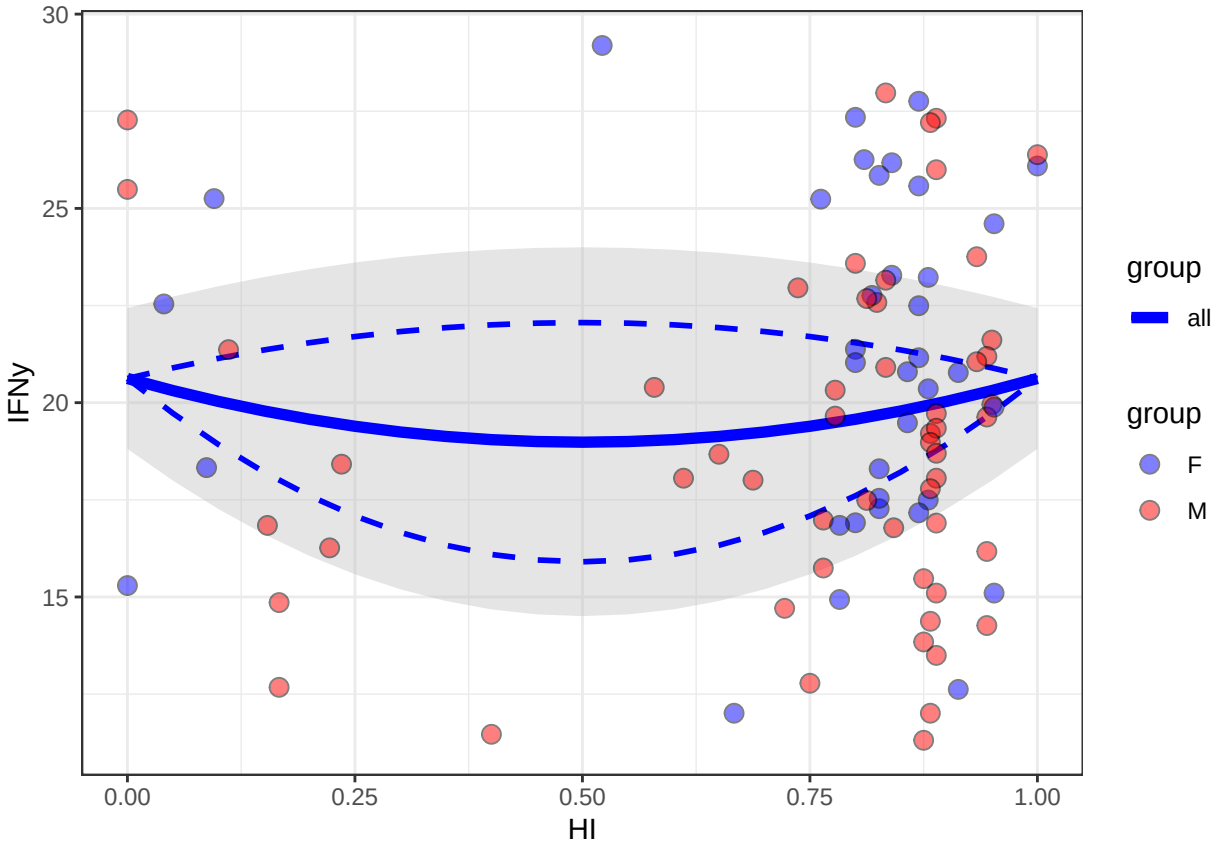
```
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 21.5378404  0.4887126  5.0000000
##
## Log-likelihood: -156.06
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## 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
## 19.3394960 19.5847997 -0.2596393  5.0000000
##
## Log-likelihood: -106.63
## 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
## 21.1614881 21.6974542  0.4917037  5.0000000
##
## Log-likelihood: -156.03
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "CXCR3")

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 0.56   1 0.2910144
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.53   1 0.3020375
## [1] "Testing H2 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.3   1 0.4390994
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.24   1 0.4929463
## [1] "Testing H3 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.3   1 0.4405921
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.22   1 0.5086422
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.18   1 0.5438574
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.16   3 0.9555063
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.15   4 0.9892984
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.18   2 0.8383611
```

```
##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,
```

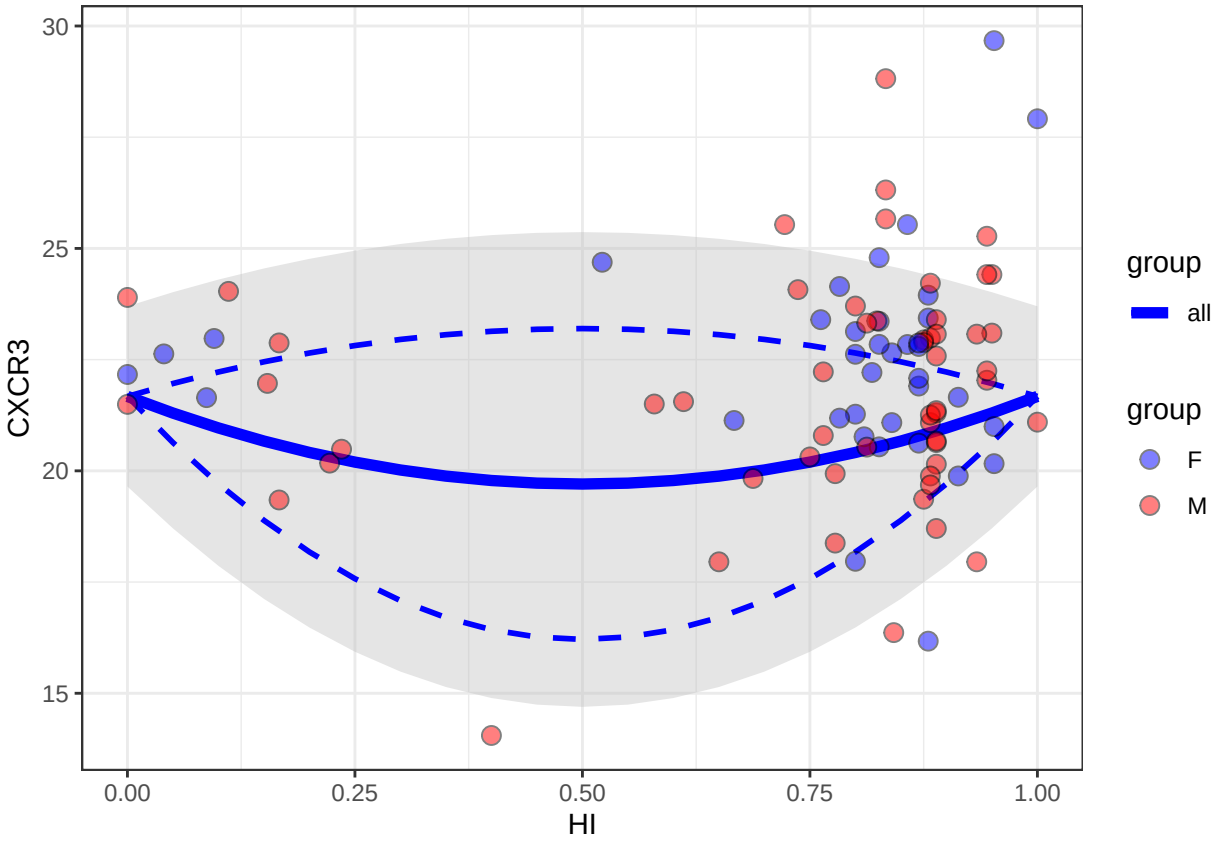
```
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 21.6732106  0.1817282  5.0000000
##
## Log-likelihood: -241.54
## 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
## 20.8584384 21.9008762  0.1755243  5.0000000
##
## Log-likelihood: -241.35
## 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
## 22.0599578  0.1971908  5.0000000
##
## Log-likelihood: -97.15
## 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
## 21.3483699  0.1614789  5.0000000
##
## Log-likelihood: -144.23
## 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
## 21.0851269 22.3418208  0.1941723  5.0000000
##
## Log-likelihood: -97.05
## 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
## 20.6775502 21.5297102  0.1539939  5.0000000
##
## Log-likelihood: -144.15
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "IL.6")

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 0.32   1 0.4268484
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.32   1 0.4231142
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.22   1 0.5106855
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.08   1 0.6979755
## [1] "Testing H3 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.2   1 0.5309636
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.08   1 0.6881683
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.03   1 0.8123636
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.48   3 0.8127799
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.48   4 0.9150859
## [1] "Testing H3 vs H2"
##    dLL dDF   pvalue
## 1 0.03   2 0.966186
```

```
##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
## 22.7616033  0.1398616  5.0000000
##
## Log-likelihood: -248.8
## 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
## 23.0866529 22.6621818  0.1418067  5.0000000
##
## Log-likelihood: -248.77
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 23.4467736  0.1661802  5.0000000
##
## Log-likelihood: -101.04
## Best method: bobyqa
##
## $H2$groupB
```

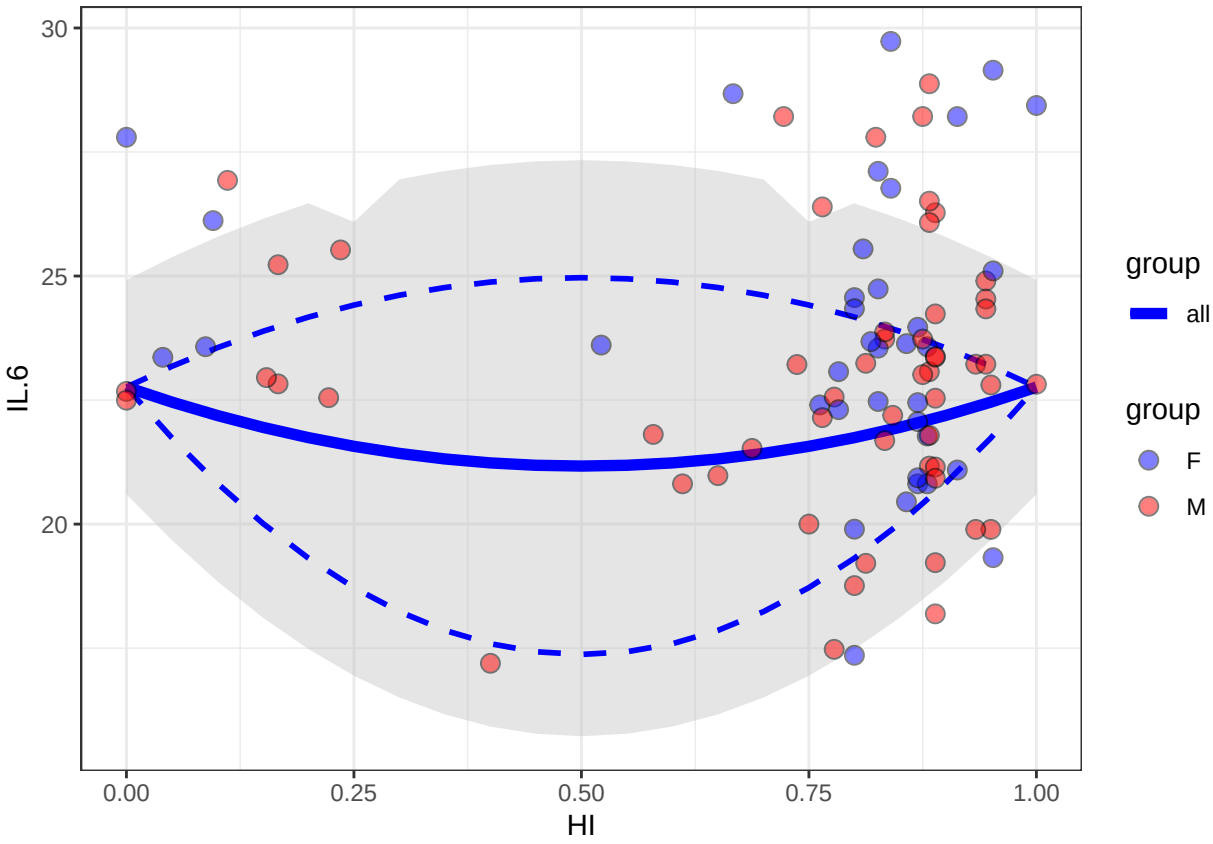
```
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1       alpha     myshape
## 22.13328956  0.09768941  5.00000000
##
## Log-likelihood: -147.28
## 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
## 23.9477460 23.2371968  0.1606118  5.0000000
##
## Log-likelihood: -101.01
## 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
## 22.3446252 22.0950213  0.1028157  5.0000000
##
```

```
## Log-likelihood: -147.27
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "IL.10")

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 0.54   1 0.3006752
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.48   1 0.3293379
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.18   1 0.5481586
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.36   1 0.3979259
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.02   1 0.8241312
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.37   1 0.3870322
## [1] "Testing H1 vs H0"
##    dLL dDF   pvalue
## 1 0.28   1 0.453702
## [1] "Testing H2 vs H0"
##   dLL dDF    pvalue
## 1   0   3 0.9998728
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.26   4 0.9714315
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.54   2 0.5838581
```

```
##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
## 23.8843774  0.1904109  5.0000000
##
## Log-likelihood: -249.64
## 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
## 24.8672587 23.4718637  0.1837463  5.0000000
##
## Log-likelihood: -249.36
## 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
## 23.7967557  0.1737669  5.0000000
##
## Log-likelihood: -101.27
```

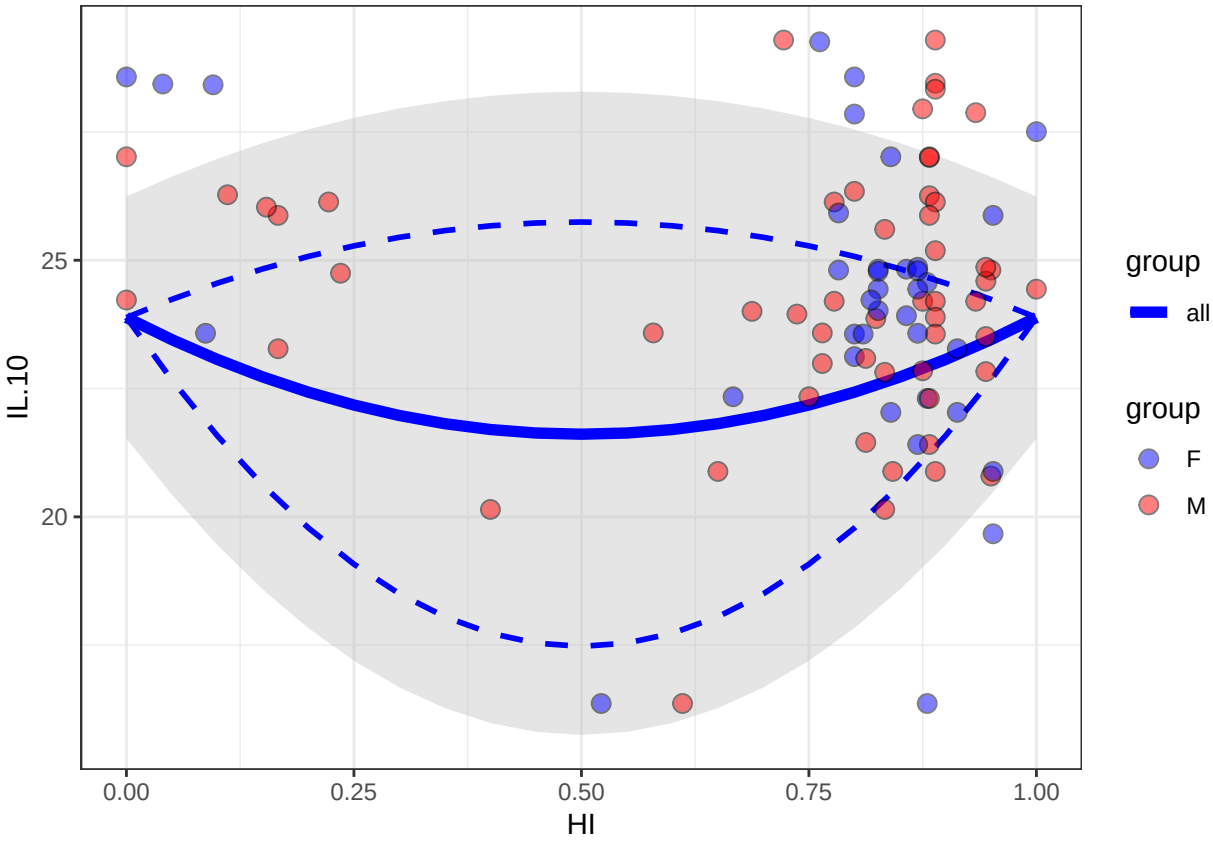
```
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 23.9422058  0.2010488  5.0000000
##
## Log-likelihood: -148.37
## 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
## 25.5002047 22.3747258  0.0694721  5.0000000
##
## Log-likelihood: -100.75
## 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
## 24.3609442 23.8538614  0.2087777  5.0000000
##
## Log-likelihood: -148.34
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "IL.13")

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 0.32   1 0.4245346
## [1] "Testing H1 no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.4   1 0.3725193
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 1.29   1 0.1080088
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.23   1 0.4995603
## [1] "Testing H3 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1 1.3   1 0.1065411
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.15   1 0.5820534
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.17   1 0.5601067
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 1.86   3 0.2937696
## [1] "Testing H3 vs H1"
```

```
##    dLL dDF    pvalue
## 1 1.78   4 0.4675144
## [1] "Testing H3 vs H2"
##   dLL dDF    pvalue
## 1 0.1   2 0.9085288
```

```
##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
## 14.2498398  0.1970013  3.1550306
##
## Log-likelihood: -268.95
## 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
## 15.1314829 14.0712499  0.2248117  3.1526025
##
## Log-likelihood: -268.78
## 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"]],
```

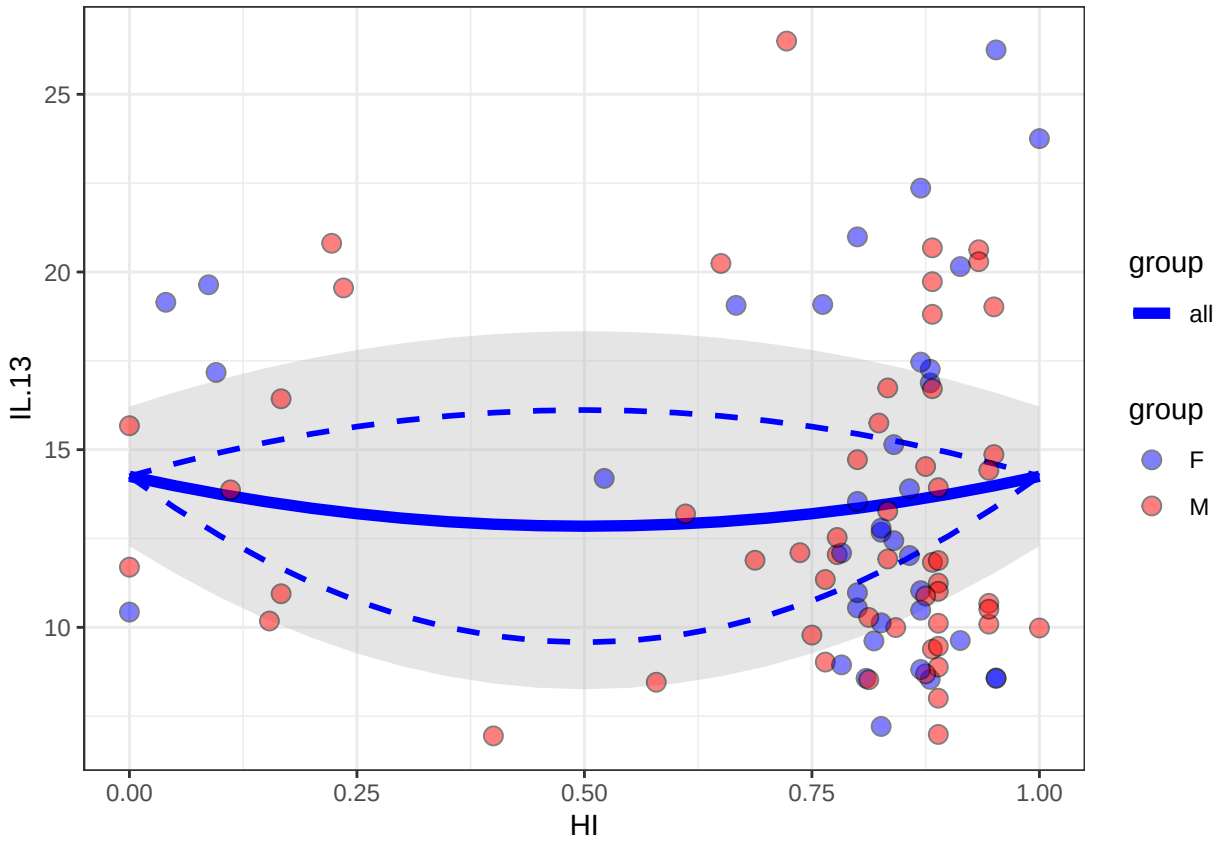
```
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 16.2485984  0.5541885  3.2349832
##
## Log-likelihood: -109.48
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = 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
## 12.4867172 -0.2464643  3.2712047
##
## Log-likelihood: -157.62
## 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
## 16.8007106 16.0902470  0.5592221  3.2295201
##
## Log-likelihood: -109.45
## 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
## 13.1327301 12.3861417 -0.2102881  3.2684064
##
## Log-likelihood: -157.55
## 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.
```



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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "IL1RN")

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 0.29   1 0.4464026
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.21   1 0.517137
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.32   1 0.4205679
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.03   1 0.7920651
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.19   1 0.533665
## [1] "Testing H3 groupB no alpha vs alpha"
```

```
##    dLL dDF    pvalue
## 1 0.03   1 0.8108416
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.61   1 0.2675825
## [1] "Testing H2 vs H0"
##    dLL dDF     pvalue
## 1 3.75   3 0.05765003
## [1] "Testing H3 vs H1"
##    dLL dDF     pvalue
## 1 4.03   4 0.08913755
## [1] "Testing H3 vs H2"
##   dLL dDF    pvalue
## 1 0.9   2 0.4065424
```

```
##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
## 14.9217625  0.1501938  3.9521434
##
## Log-likelihood: -256
## 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
## 16.0700305 14.4430762  0.1315936  3.9978080
##
## Log-likelihood: -255.39
## Best method: bobyqa
##
```

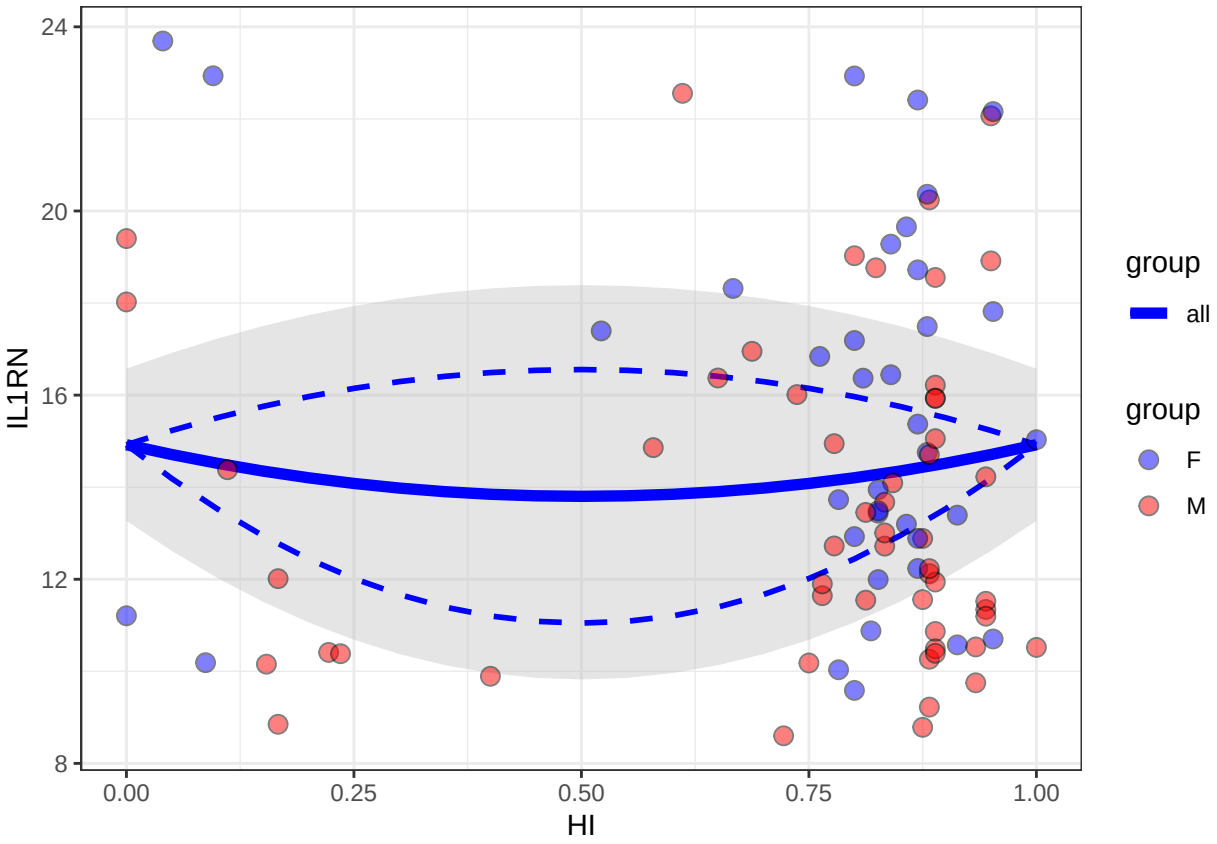
```
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 16.7485350  0.2653339  4.2873145
##
## Log-likelihood: -104.12
## 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
## 13.69654054  0.05859052  4.02144442
##
## Log-likelihood: -148.14
## 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
## 18.9279687 15.7624450  0.2119666  4.4296194
```

```
##
## Log-likelihood: -103.25
## 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
## 14.02619341 13.55606315  0.05405688  4.02506977
##
## Log-likelihood: -148.1
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "CXCR3")

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 0.56   1 0.2910144
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.53   1 0.3020375
## [1] "Testing H2 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.3   1 0.4390994
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.24   1 0.4929463
## [1] "Testing H3 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.3   1 0.4405921
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.22   1 0.5086422
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.18   1 0.5438574
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.16   3 0.9555063
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.15   4 0.9892984
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.18   2 0.8383611
```

```
##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
## 21.6732106  0.1817282  5.0000000
##
## Log-likelihood: -241.54
```

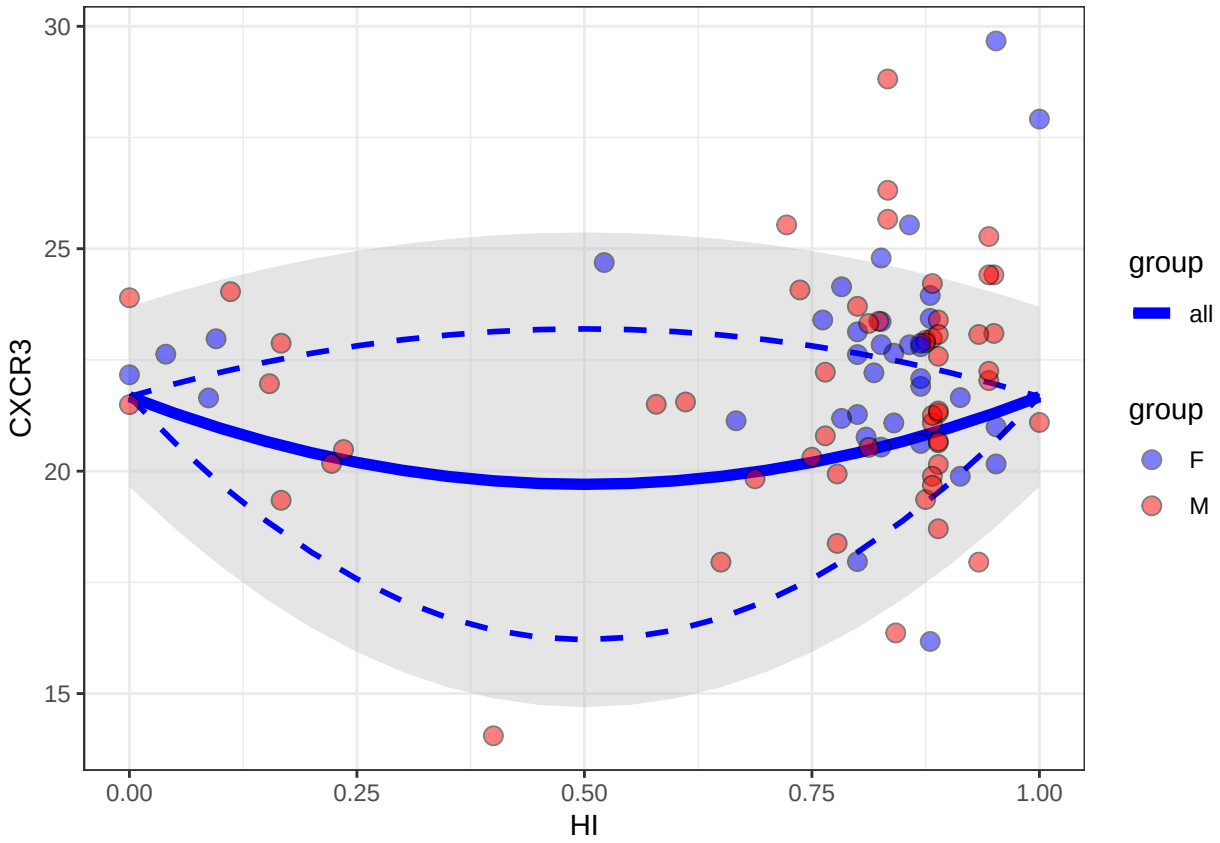
```
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 20.8584384 21.9008762  0.1755243  5.0000000
##
## Log-likelihood: -241.35
## 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
## 22.0599578  0.1971908  5.0000000
##
## Log-likelihood: -97.15
## 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
## 21.3483699  0.1614789  5.0000000
```

```
##
## Log-likelihood: -144.23
## 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
## 21.0851269 22.3418208  0.1941723  5.0000000
##
## Log-likelihood: -97.05
## 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
## 20.6775502 21.5297102  0.1539939  5.0000000
##
## Log-likelihood: -144.15
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "CASP1")

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 0.12   1 0.618711
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.12   1 0.6181879
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.08   1 0.6834257
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.05   1 0.7538205
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.09   1 0.6777975
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.04   1 0.7800683
## [1] "Testing H1 vs H0"
##   dLL dDF    pvalue
## 1 0.1   1 0.6533382
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.01   3 0.9994072
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.05   4 0.9988812
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.14   2 0.8690434
```

```
##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
## 21.88852929  0.08646794  5.00000000
```

```
##
## Log-likelihood: -242.39
## 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
## 21.30280035 22.08009640  0.08552456  5.00000000
##
## Log-likelihood: -242.29
## 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
## 22.0175911  0.1139856  5.0000000
##
## Log-likelihood: -97.46
## 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:
```

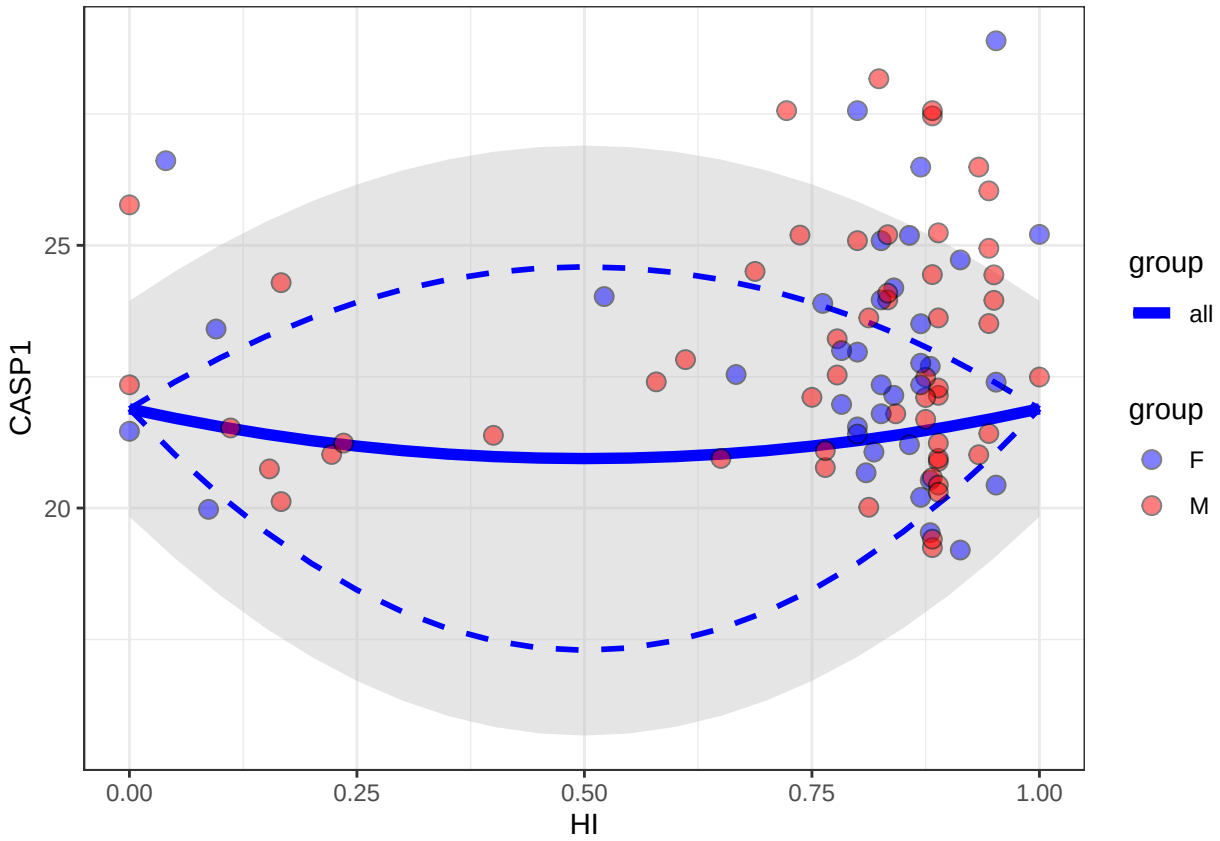
```
##          L1       alpha     myshape
## 21.80873537  0.06970773  5.00000000
##
## Log-likelihood: -144.93
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     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
## 21.8307183 22.0952789  0.1164336  5.0000000
##
## Log-likelihood: -97.45
## 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
## 20.91318966 22.05766988  0.06093675  5.00000000
##
## Log-likelihood: -144.79
## Best method: bobyqa
```

```
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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "CXCL9")

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 0.06   1 0.7214733
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.07   1 0.7162716
## [1] "Testing H2 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1   0   1 0.9844341
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.18   1 0.5531062
## [1] "Testing H3 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1   0   1 0.9988702
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.08   1 0.6889052
## [1] "Testing H1 vs H0"
##   dLL dDF    pvalue
## 1   0   1 0.9301337
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.93   3 0.6027429
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 1.47   4 0.5688134
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.54   2 0.5810038
```

```
##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
## 20.22464470  0.06369245  5.00000000
##
## Log-likelihood: -255.25
## 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
## 20.33660617 20.19645318  0.06531311  5.00000000
##
## Log-likelihood: -255.25
## 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
## 20.507347492 -0.004760387  5.000000000
##
## Log-likelihood: -100.77
## 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"]]),
```

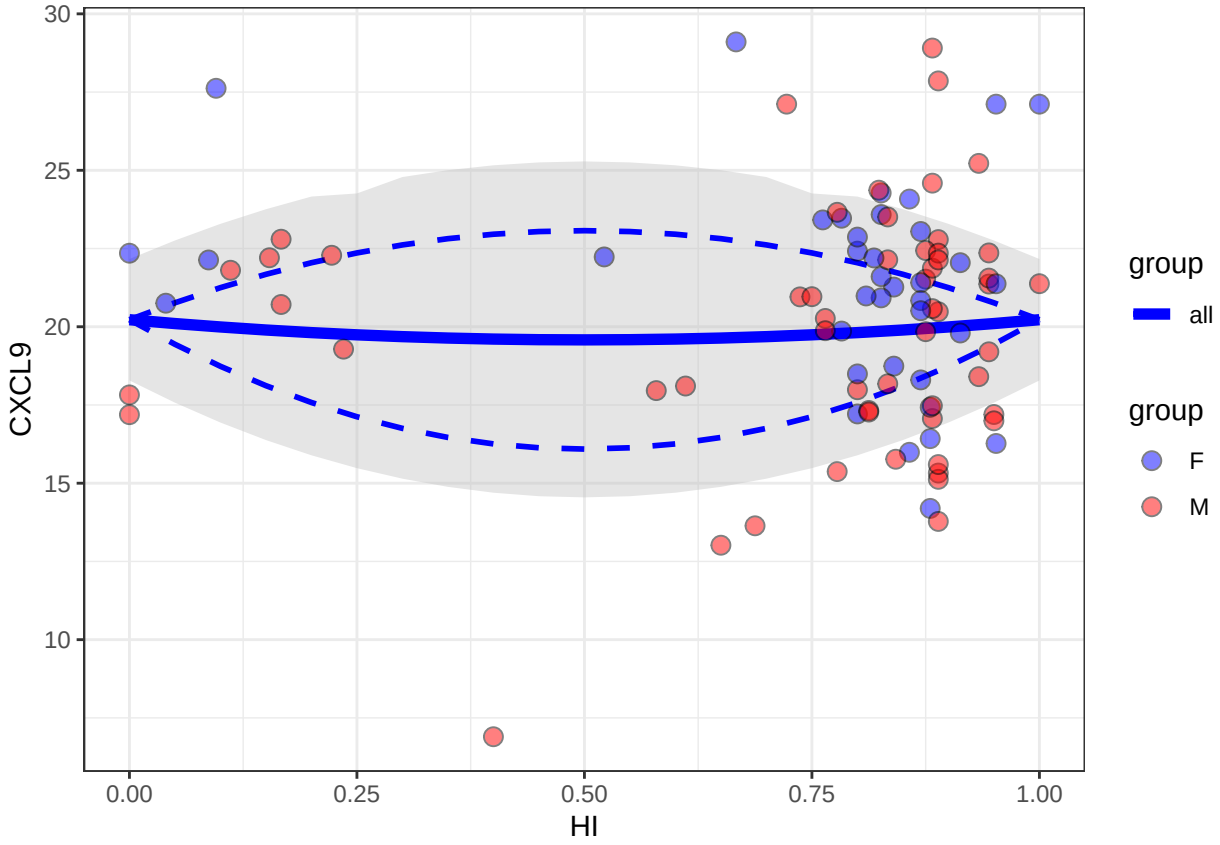
```
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 20.2483056  0.1619977  5.0000000
##
## Log-likelihood: -153.55
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, 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
## 22.0907265549 19.9822899813 -0.0003491458  5.0000000000
##
## Log-likelihood: -100.43
## 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
## 18.988841 20.348135  0.113251  5.000000
##
## Log-likelihood: -153.34
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "IDO1")

IDO1 <- parasiteLoad::analyse(data = field,
                          response = "IDO1",
                          model = "weibull",
                          group = "Sex")
```

```
## [1] "Analysing data for response: IDO1"
## [1] "Fit for the response: IDO1"
## [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"
```

```
## 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 0.32   1 0.4268605
## [1] "Testing H1 no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.2   1 0.5291766
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.8665802
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.52   1 0.3096423
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.07   1 0.7153135
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.62   1 0.2670407
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.29   1 0.4443023
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 4.31   3 0.0346746
## [1] "Testing H3 vs H1"
##    dLL dDF     pvalue
## 1 6.13   4 0.01546302
```

```
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 2.11   2 0.1208777
```

```
##All
print(IDO1)
```

```
## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$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
## 15.3102796  0.1718163  4.1102826
##
## Log-likelihood: -254.51
## 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
## 16.0384189 14.9127185  0.1420189  4.1374358
##
## Log-likelihood: -254.22
## 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)
```

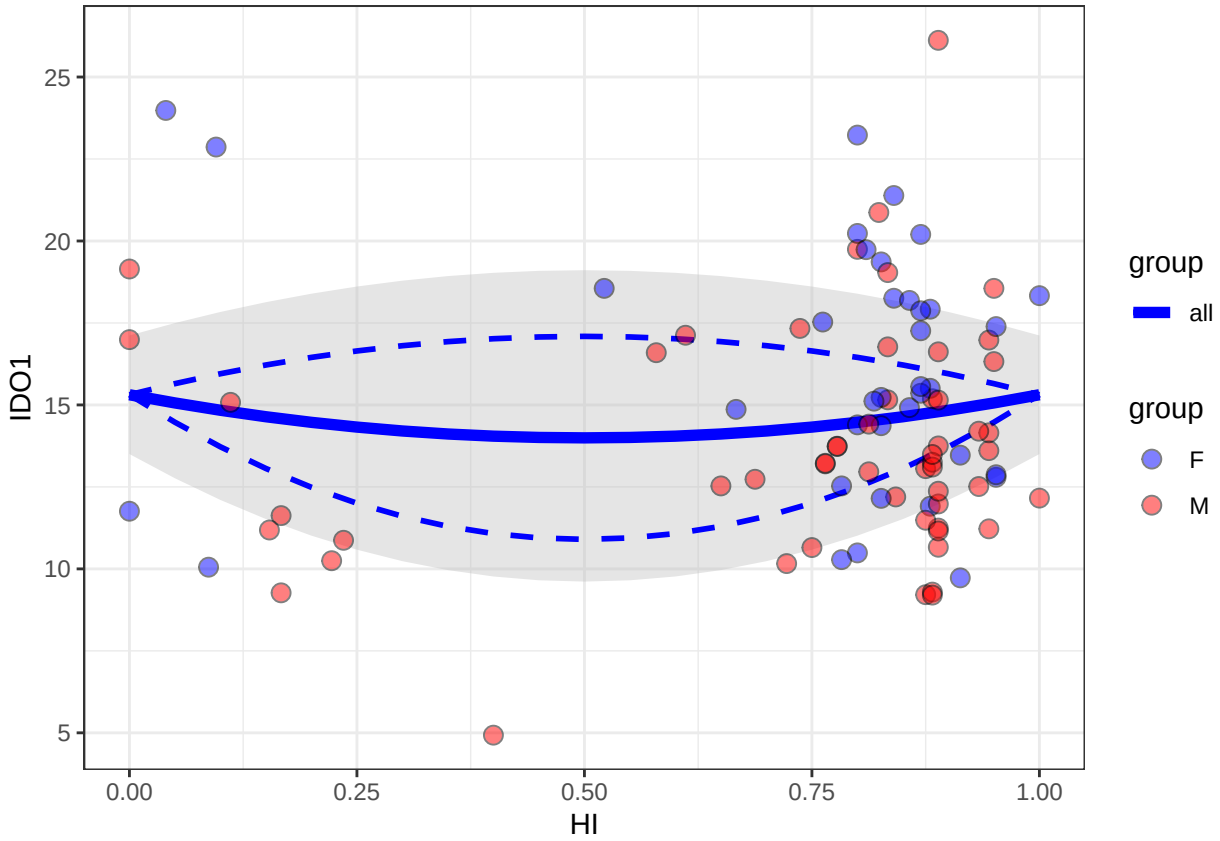
```
##
## Coefficients:
##          L1       alpha     myshape
## 16.31130944  0.05430951  4.71766930
##
## Log-likelihood: -101.8
## Best method: L-BFGS-B
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (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
## 14.6433014  0.2648944  4.0163443
##
## Log-likelihood: -148.4
## 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
## 18.6843857 14.6598882 -0.1228565  5.0000000
##
## Log-likelihood: -99.91
## Best method: bobyqa
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
```

```
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 13.8154878 15.0403007  0.2809686  4.0437361
##
## Log-likelihood: -148.18
## Best method: bobyqa
```

```
bananaPlot(mod = IDO1$H0,
             data = field,
             response = "IDO1",
             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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "IRGM1")
```

```r
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 2.79   1 0.01814648
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF     pvalue
## 1 2.92   1 0.01568958
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF     pvalue
## 1 2.72   1 0.01962714
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.01   1 0.899677
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF     pvalue
## 1 2.89   1 0.01611965
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.06   1 0.7251936
## [1] "Testing H1 vs H0"
```

```
##    dLL dDF    pvalue
## 1 2.49   1 0.0256257
## [1] "Testing H2 vs H0"
##    dLL dDF      pvalue
## 1 7.77   3 0.001410241
## [1] "Testing H3 vs H1"
##    dLL dDF      pvalue
## 1 8.71   4 0.001595116
## [1] "Testing H3 vs H2"
##    dLL dDF     pvalue
## 1 3.44   2 0.03219086
```

```
##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
## 10.2975857 -0.5679546  3.7481100
##
## Log-likelihood: -237.22
## 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
## 12.0434976  9.6644856 -0.5852502  3.8853613
##
## Log-likelihood: -234.73
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
```

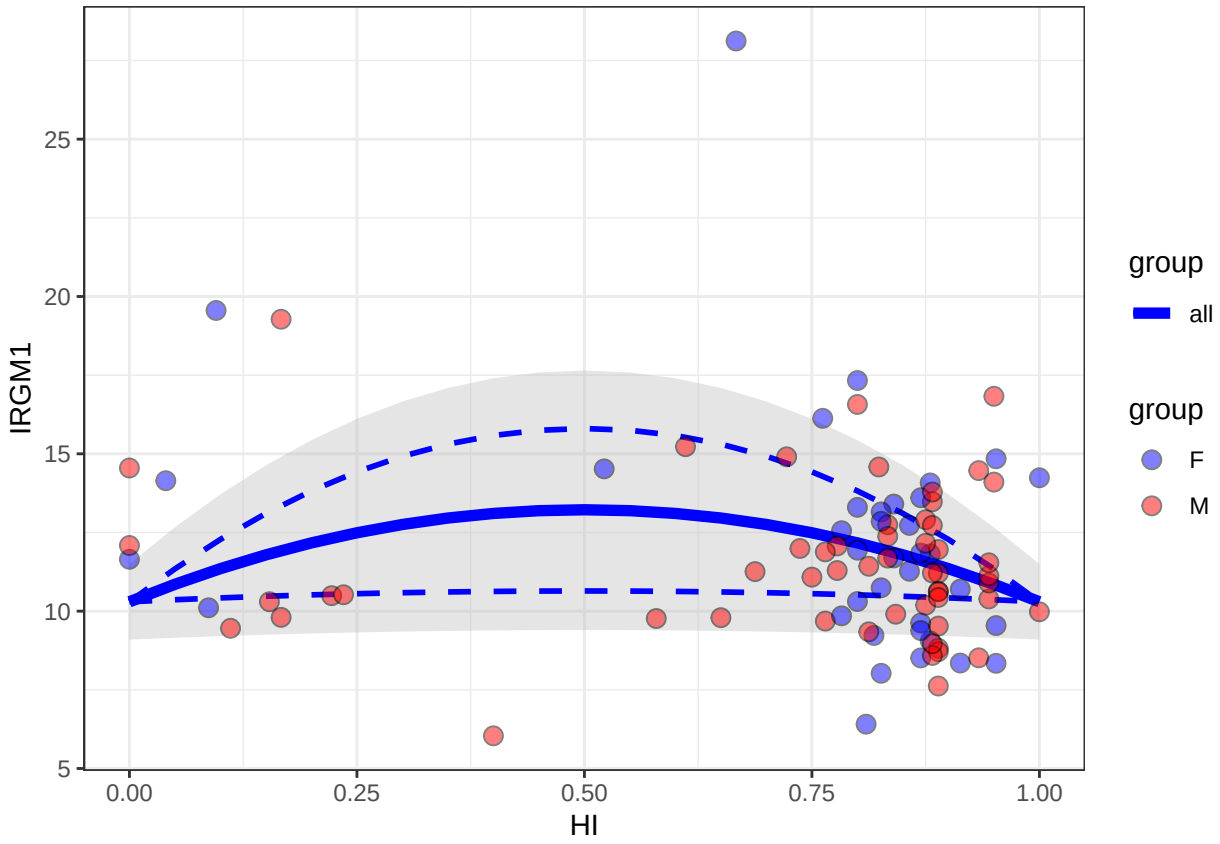
```
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##        L1     alpha   myshape
##  9.836522 -1.013484  3.444972
##
## Log-likelihood: -100.27
## 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
## 11.49474903  0.02960304  4.76895365
##
## Log-likelihood: -129.18
## 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
## 12.731552  8.894424 -1.012116  3.661200
##
## Log-likelihood: -98.03
## Best method: bobyqa
```

```
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 13.0603316 11.1839610  0.0827654  4.9718090
##
## Log-likelihood: -127.98
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "MPO")

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"
```

```
## 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 0.12   1 0.6229743
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.08   1 0.6810662
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.9136166
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.12   1 0.618121
## [1] "Testing H3 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1   0   1 0.9958121
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.11   1 0.6361416
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.14   1 0.5940009
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 2.18   3 0.2251199
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 2.27   4 0.3368593
```

```
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.24   2 0.7896226
```

```
##All
print(MPO)
```

```
## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$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
## 20.22502836  0.08170258  4.96865481
##
## Log-likelihood: -265.61
## 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
## 20.79902633 19.94745228  0.07028684  4.97551453
##
## Log-likelihood: -265.47
## 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)
```

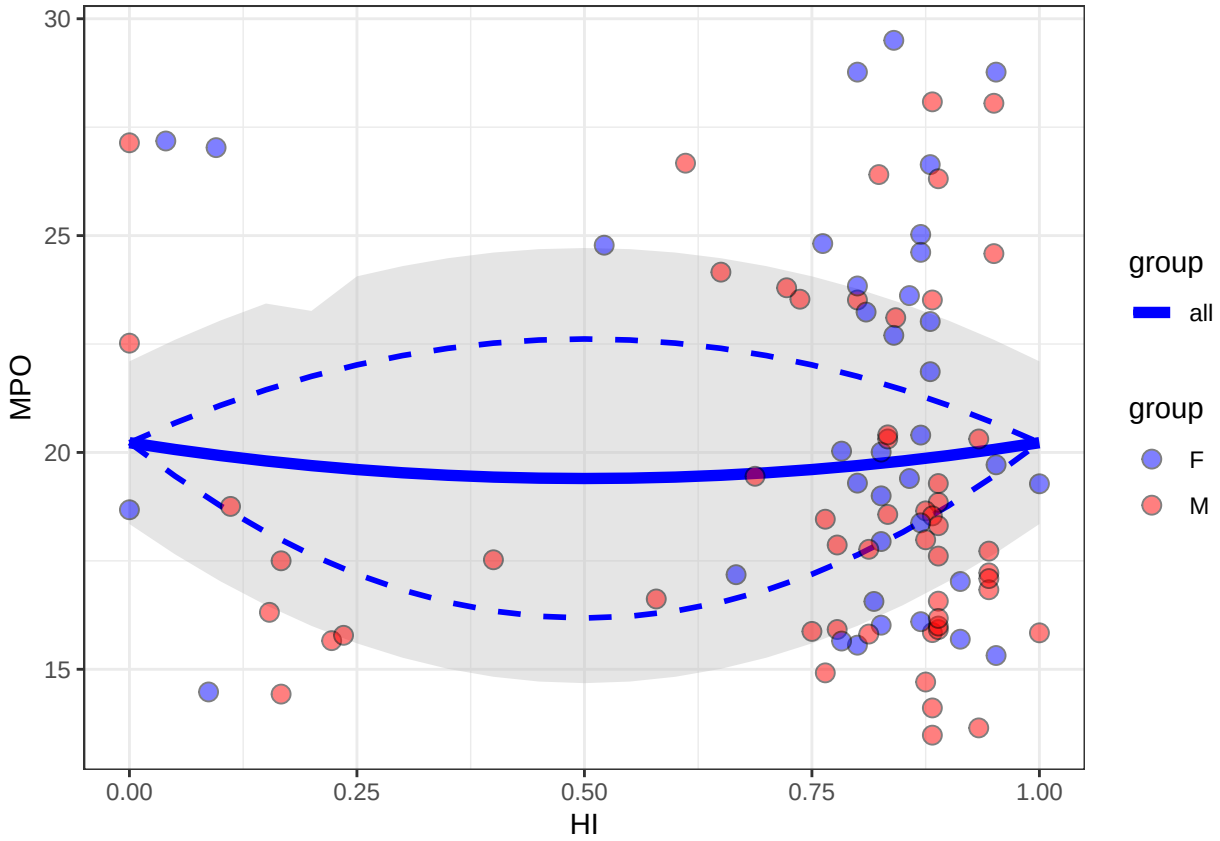
```
##
## Coefficients:
##          L1       alpha     myshape
## 20.98487545  0.03364238  5.00000000
##
## Log-likelihood: -107.32
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (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
## 19.48225359  0.09426582  4.94830866
##
## Log-likelihood: -156.11
## 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
## 22.12774842 20.35317639 -0.00170931  5.00000000
##
## Log-likelihood: -107.1
## 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
## 19.7070372 19.3772955  0.0909447  4.9484610
##
## Log-likelihood: -156.09
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "MUC2")
```

```r
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 1.74   1 0.06228684
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.82   1 0.05633974
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.23   1 0.499535
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 2.61   1 0.0223056
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.22   1 0.5114254
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF     pvalue
## 1 2.89   1 0.01622584
## [1] "Testing H1 vs H0"
```

```
##    dLL dDF    pvalue
## 1 0.13   1 0.6103351
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 4.07   3 0.0433805
## [1] "Testing H3 vs H1"
##    dLL dDF     pvalue
## 1 4.48   4 0.06186702
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.55   2 0.5774805
```

```
##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
## 11.5136079  0.4044631  3.6020566
##
## Log-likelihood: -232.74
## 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
## 11.0576662 11.6996777  0.4097291  3.6134916
##
## Log-likelihood: -232.61
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
```

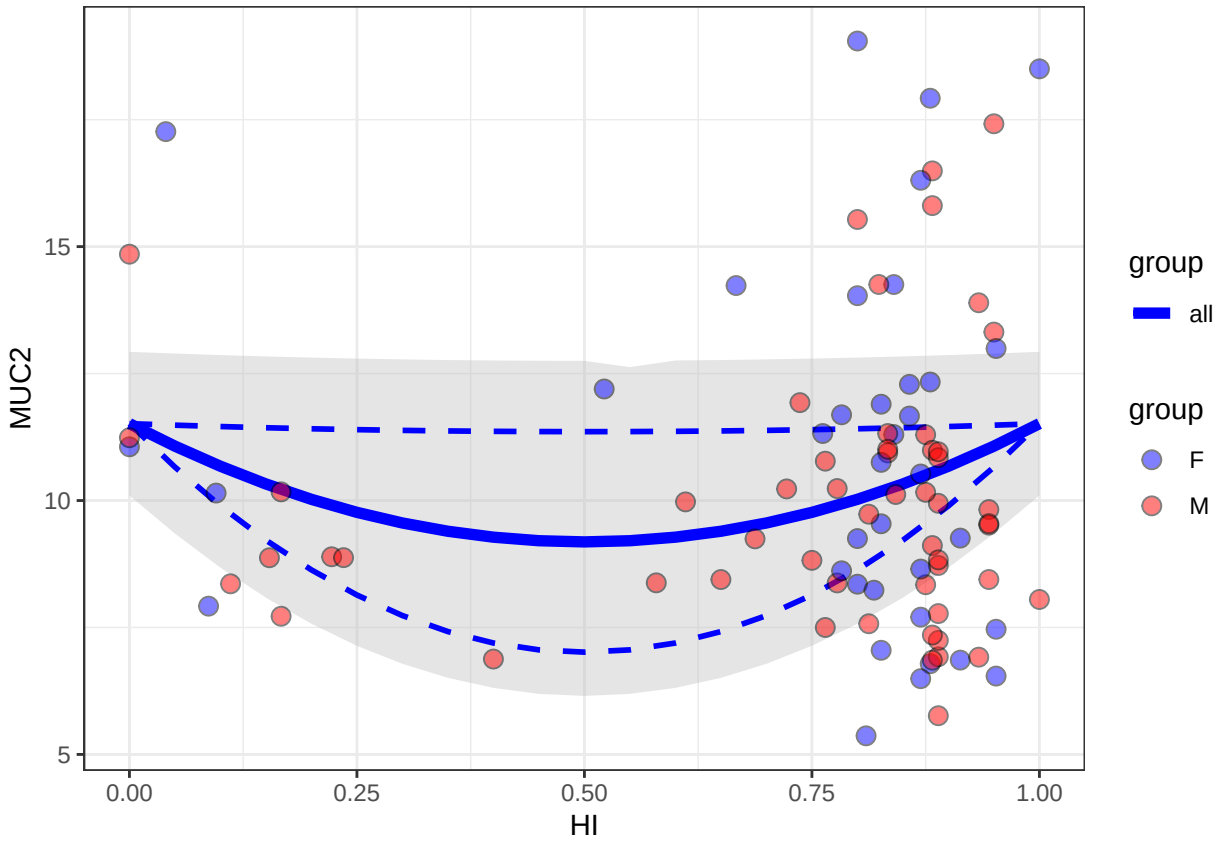
```
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##        L1     alpha   myshape
## 11.662169  0.244167  3.325369
##
## Log-likelihood: -99.03
## 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
## 11.6712168  0.5842741  4.1469840
##
## Log-likelihood: -129.65
## 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
## 11.9102368 11.5614899  0.2397356  3.3235714
##
## Log-likelihood: -99.01
## 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
## 10.6156147 12.0892153  0.5857167  4.2295717
##
## Log-likelihood: -129.12
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "MUC5AC")

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 0.9   1 0.1805306
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.95   1 0.1678589
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.01   1 0.915071
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 1.26   1 0.1128188
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.07   1 0.7002116
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.36   1 0.09970125
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.08   1 0.6856931
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.53   3 0.7854645
## [1] "Testing H3 vs H1"
##    dLL dDF   pvalue
## 1 2.32   4 0.326839
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 1.87   2 0.1546241
```

```
##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
## 13.02514  0.42150  2.61105
##
## Log-likelihood: -267.73
## 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
## 12.4669087 13.2612341  0.4290868  2.6166118
##
## Log-likelihood: -267.65
## 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
## 12.15383634  0.06028135  2.68224659
##
## Log-likelihood: -108.07
## 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"]],
```

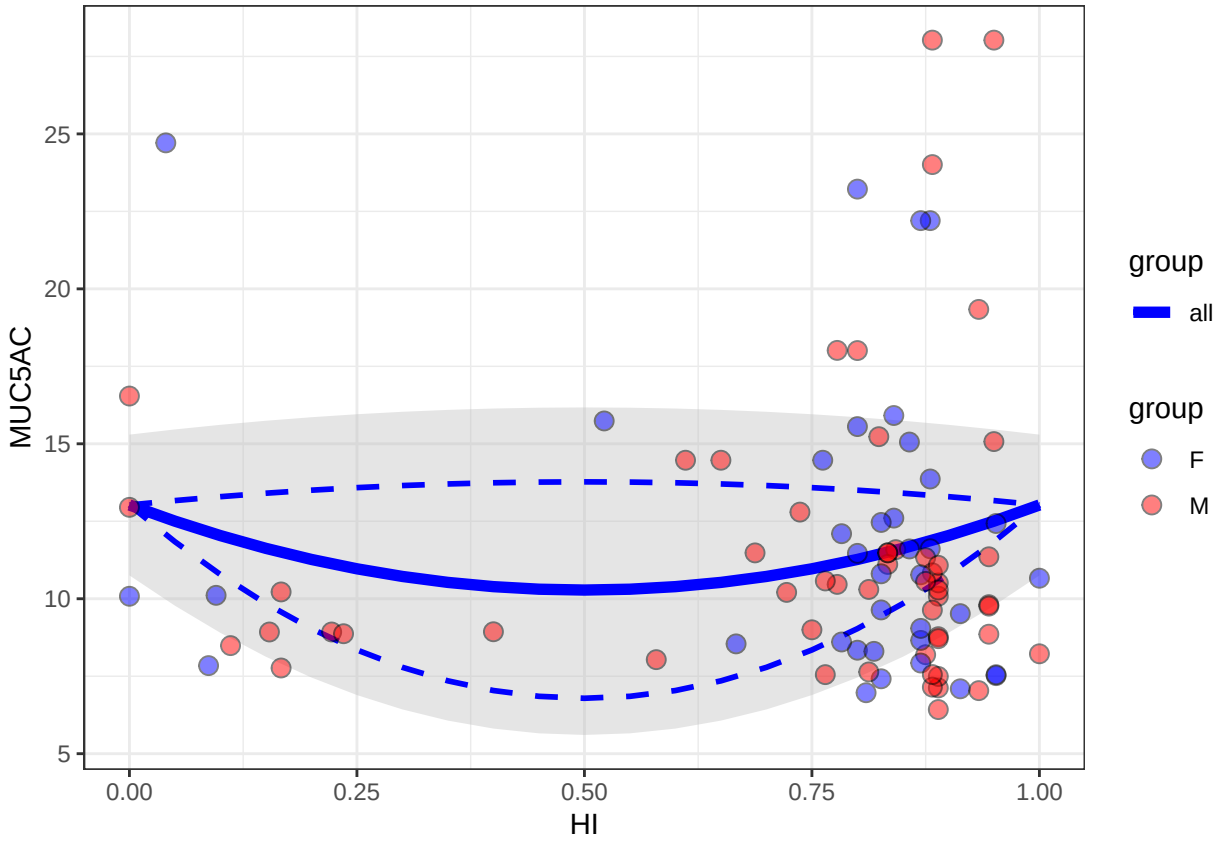
```
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 13.4584060  0.5925573  2.5917131
##
## Log-likelihood: -159.13
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## 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
## 14.1896513 10.3362083 -0.2673906  2.7611820
##
## Log-likelihood: -107.21
## 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
## 10.7635320 14.3618007  0.5667373  2.6783429
##
## Log-likelihood: -158.12
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "MYD88")

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"

## 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 2.61   1 0.02229377
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF     pvalue
## 1 2.37   1 0.02955345
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.76   1 0.2171589
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF     pvalue
## 1 2.26   1 0.03357069
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.25   1 0.4765959
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF     pvalue
## 1 2.19   1 0.03618594
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.19   1 0.5370658
## [1] "Testing H2 vs H0"
```

```
##    dLL dDF   pvalue
## 1 1.73   3 0.326372
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 2.89   4 0.2162854
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 1.35   2 0.2588828
```

```
##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
## 21.961439  0.434820  4.858284
##
## Log-likelihood: -273.32
## 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
## 22.716590 21.620033  0.424435  4.878195
##
## Log-likelihood: -273.13
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
```

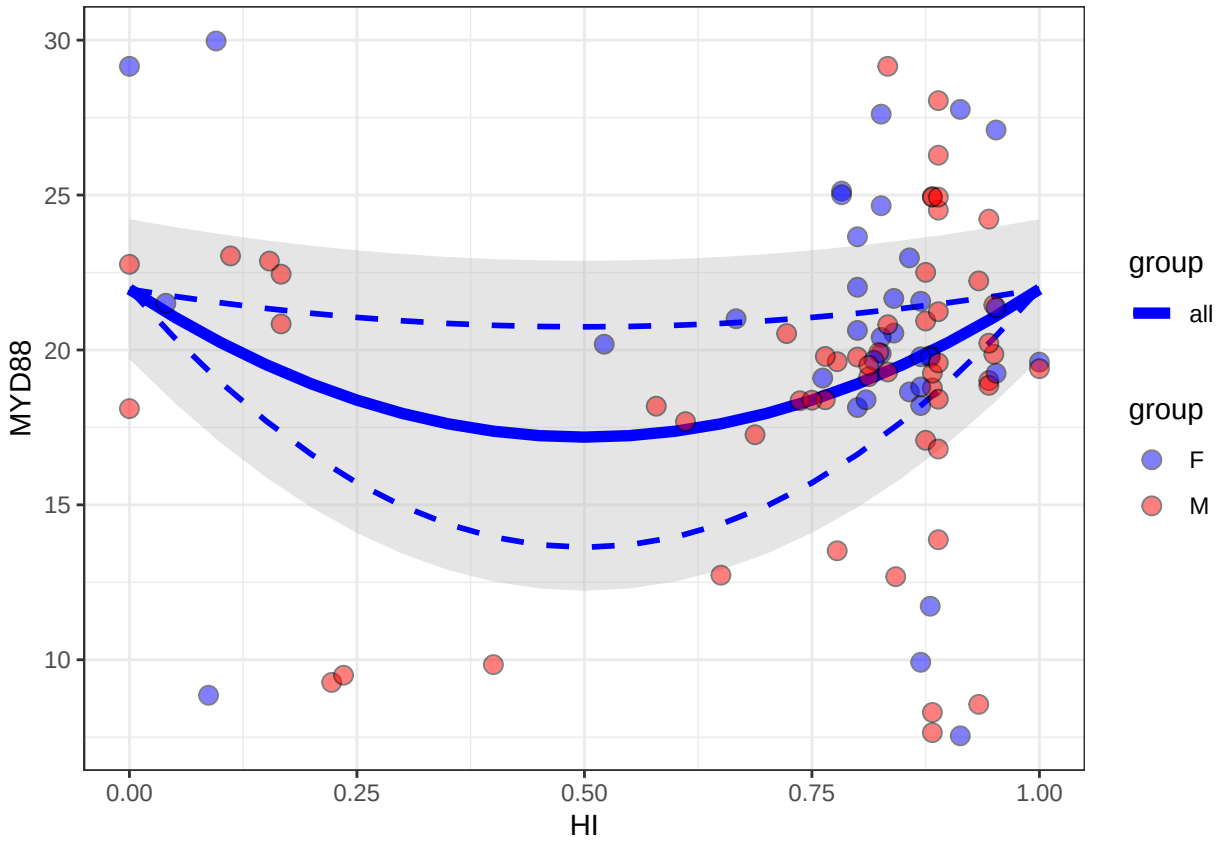
```
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 22.2823999  0.3263479  5.0000000
##
## Log-likelihood: -110.42
## 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
## 22.0417760  0.5609467  4.8647644
##
## Log-likelihood: -161.17
## 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
## 24.7794716 20.4470891  0.2100189  5.0000000
##
## Log-likelihood: -109.21
## 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
## 21.0544588 22.2739552  0.5460723  4.8849575
##
## Log-likelihood: -161.03
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "NCR1")

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 0.76   1 0.218994
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.77   1 0.215416
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.04   1 0.7907401
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.93   1 0.1723222
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.04   1 0.7763321
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.89   1 0.1815362
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.08   1 0.6926342
## [1] "Testing H2 vs H0"
##    dLL dDF   pvalue
## 1 0.33   3 0.884444
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.29   4 0.9646723
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.04   2 0.9561407
```

```
##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
## 23.7770011  0.2145699  5.0000000
##
## Log-likelihood: -246.25
```

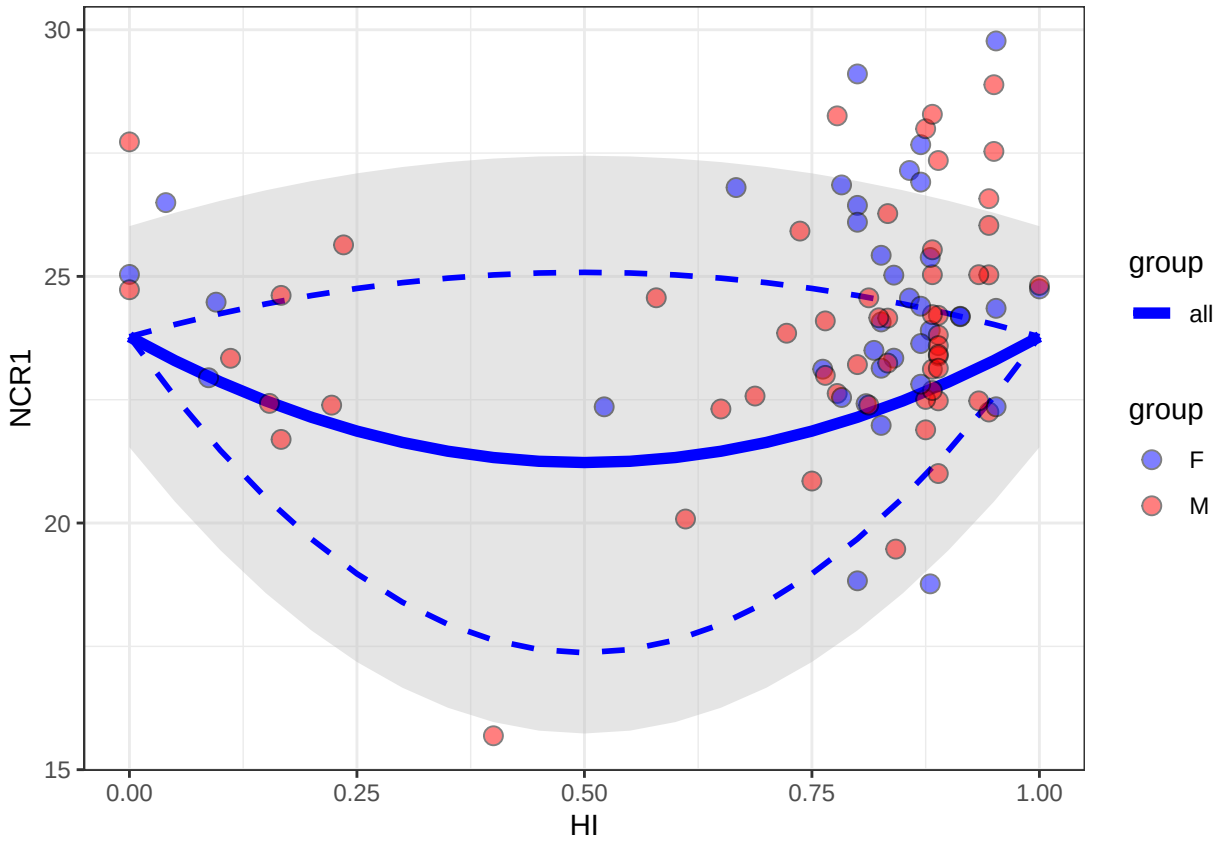
```
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 23.2186353 23.9614784  0.2139292  5.0000000
##
## Log-likelihood: -246.17
## 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
## 23.27355293  0.07873994  5.00000000
##
## Log-likelihood: -99.54
## 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
## 24.0376803  0.2914757  5.0000000
```

```
##
## Log-likelihood: -146.39
## 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
## 23.02335741 23.40382690  0.08539786  5.00000000
##
## Log-likelihood: -99.53
## 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
## 23.5018531 24.1622979  0.2838846  5.0000000
##
## Log-likelihood: -146.35
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "PRF1")

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 0.02   1 0.8459611
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.02   1 0.8487255
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.8966031
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.8800951
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.9167281
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.9117635
## [1] "Testing H1 vs H0"
##   dLL dDF    pvalue
## 1   0   1 0.9481792
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.01   3 0.9989601
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.05   4 0.9985552
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.04   2 0.9565873
```

```
##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
## 21.83464210 -0.03882146  5.00000000
```

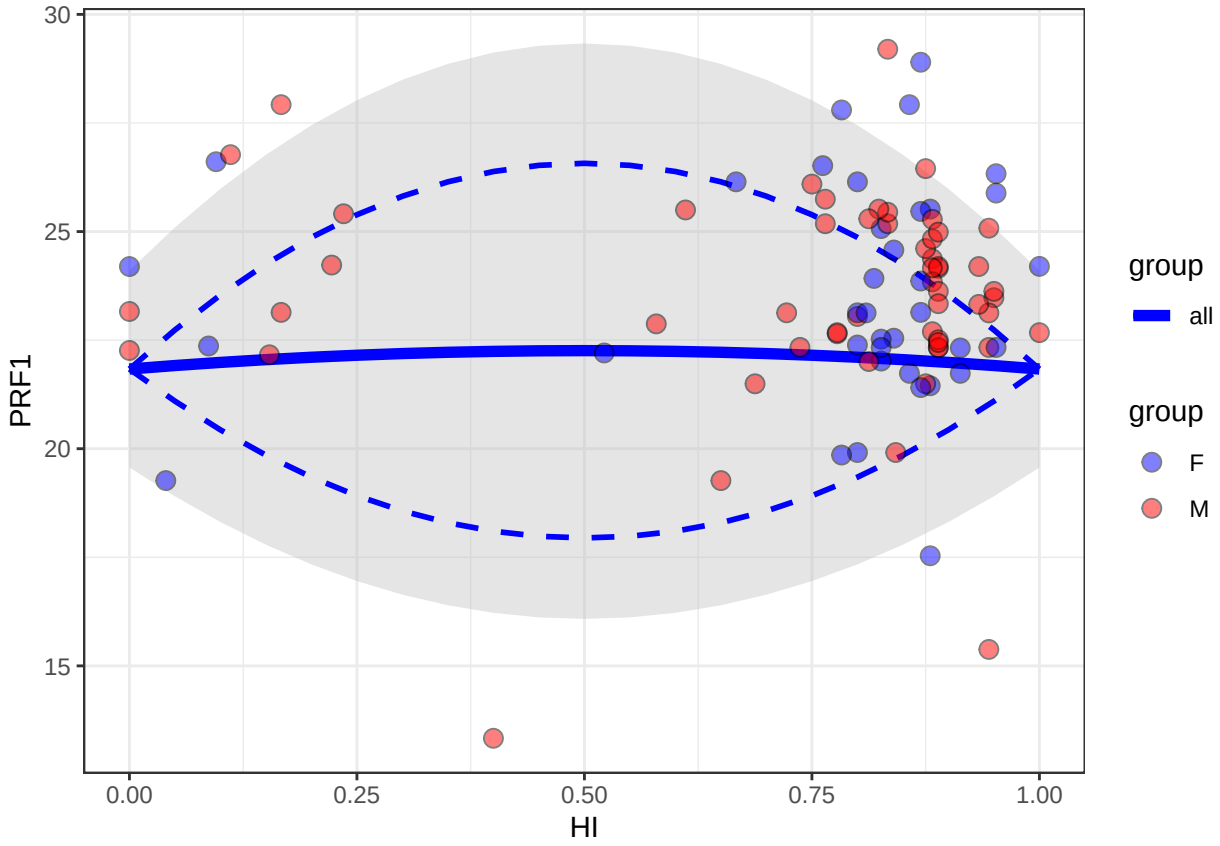
```
##
## Log-likelihood: -246.1
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 21.92053663 21.80889896 -0.03822195  5.00000000
##
## Log-likelihood: -246.1
## 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
## 21.91505837 -0.04011753  5.00000000
##
## Log-likelihood: -99.26
## 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
## 21.77114496 -0.03959978  5.00000000
##
## Log-likelihood: -146.83
## 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
## 21.55771788 22.08205939 -0.03235849  5.00000000
##
## Log-likelihood: -99.24
## 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
## 22.19961425 21.68455722 -0.02966573  5.00000000
##
## Log-likelihood: -146.8
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "RETNLB")

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 0.1   1 0.6588306
## [1] "Testing H1 no alpha vs alpha"
##   dLL dDF   pvalue
## 1 0.1   1 0.658107
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.05   1 0.7442577
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.31   1 0.4328414
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.06   1 0.7237122
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.27   1 0.4591174
## [1] "Testing H1 vs H0"
##   dLL dDF    pvalue
## 1   0   1 0.9762545
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 1.24   3 0.4779817
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 1.77   4 0.4726806
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.53   2 0.5915401
```

```
##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,
```

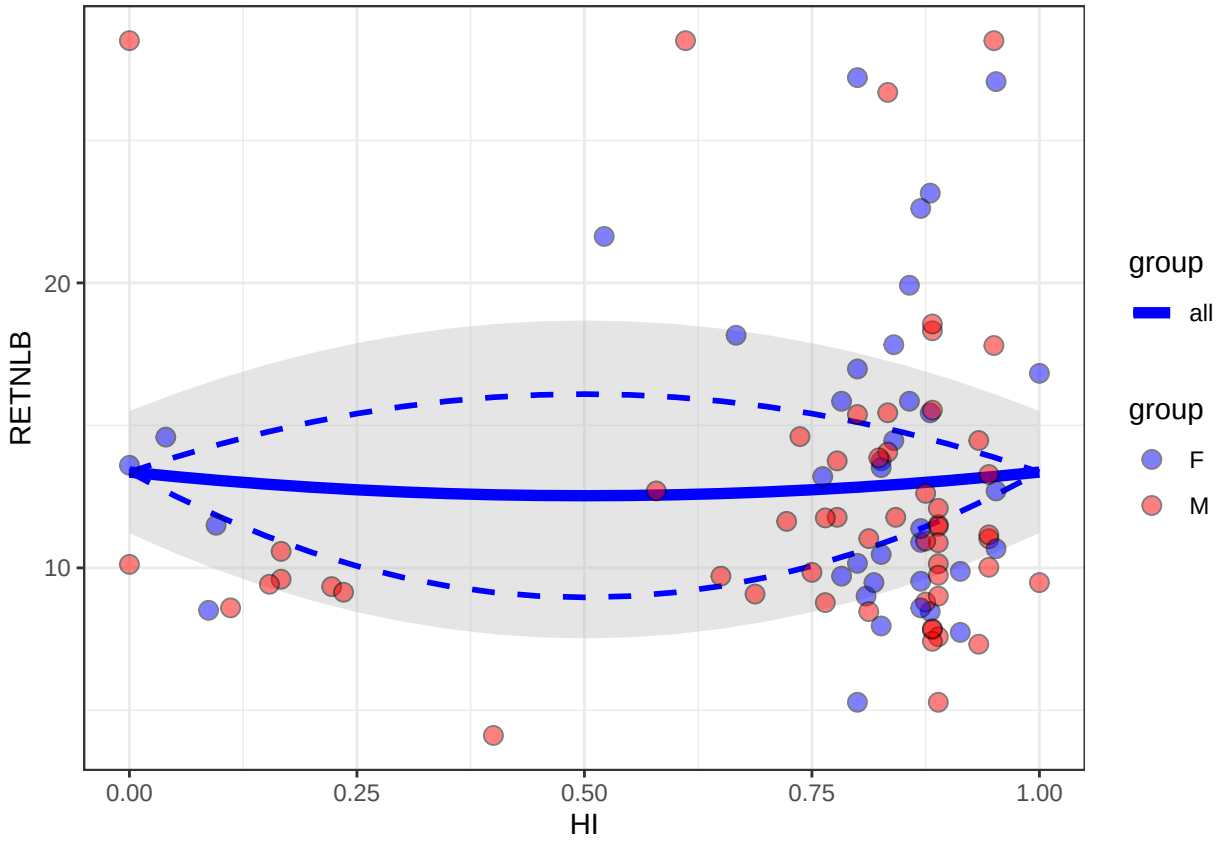
```
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 13.3615675  0.1243401  2.5166167
##
## Log-likelihood: -281.02
## Best method: L-BFGS-B
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 13.3166088 13.3787930  0.1247884  2.5163601
##
## Log-likelihood: -281.02
## 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
## 13.3980155 -0.1579974  2.7520614
##
## Log-likelihood: -113.48
## 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
## 13.1791194  0.2679829  2.4174675
##
## Log-likelihood: -166.3
## 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
## 11.3190443 14.0955075 -0.1606394  2.8083617
##
## Log-likelihood: -113.09
## 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
## 14.1682584 12.7908738  0.2587244  2.4322914
##
## Log-likelihood: -166.17
## Best method: bobyqa
```

```
bananaPlot(mod = RETNLB$HO,
             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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "SOCS1")

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"
## 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"
## [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 1.85   1 0.0546402
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.88   1 0.05231154
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.79   1 0.05857387
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.39   1 0.3784439
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.76   1 0.06039194
## [1] "Testing H3 groupB no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.4   1 0.3686053
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.12   1 0.6171367
## [1] "Testing H2 vs H0"
```

```
##    dLL dDF    pvalue
## 1 0.59   3 0.7585025
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.73   4 0.8328556
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.27   2 0.7642852
```

```
##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
## 11.6741289 -0.3284837  4.9004614
##
## Log-likelihood: -224.35
## 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
## 11.3294123 11.8043893 -0.3262144  4.9122668
##
## Log-likelihood: -224.22
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
```

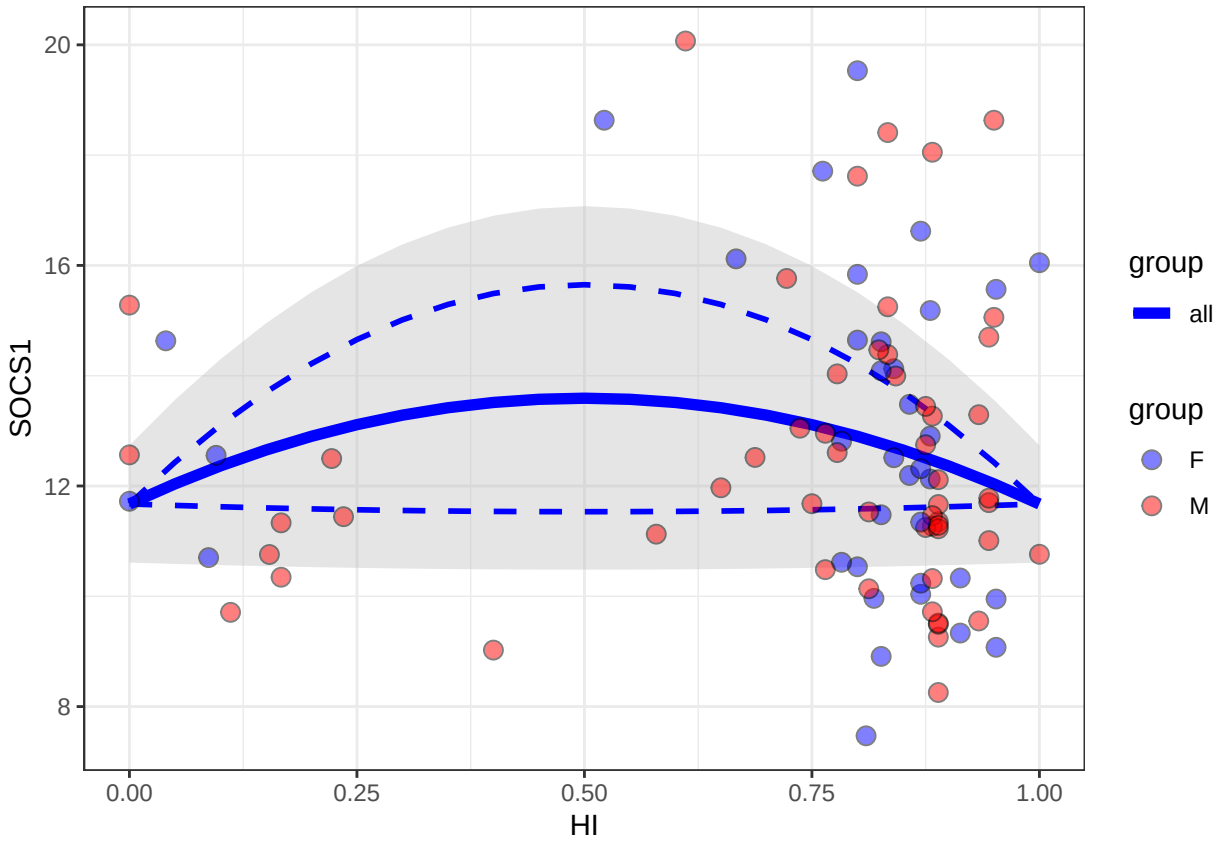
```
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 11.4394665 -0.5027339  5.0000000
##
## Log-likelihood: -90.42
## 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
## 11.8688017 -0.1933214  4.8161410
##
## Log-likelihood: -133.34
## 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
## 11.5217085 11.4142943 -0.5021997  5.0000000
##
## Log-likelihood: -90.41
## 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
## 11.2132014 12.1225114 -0.1895998  4.8416736
##
## Log-likelihood: -133.07
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "TICAM1")

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 2.94   1 0.01527628
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF     pvalue
## 1 2.94   1 0.01528152
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 1.27   1 0.1113205
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.57   1 0.07617043
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 1.28   1 0.1094055
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.58   1 0.07510119
## [1] "Testing H1 vs H0"
##   dLL dDF    pvalue
## 1   0   1 0.9689175
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.43   3 0.8370765
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.45   4 0.9241376
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.03   2 0.9739394
```

```
##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
## 22.0216488  0.4027216  5.0000000
##
## Log-likelihood: -244.39
```

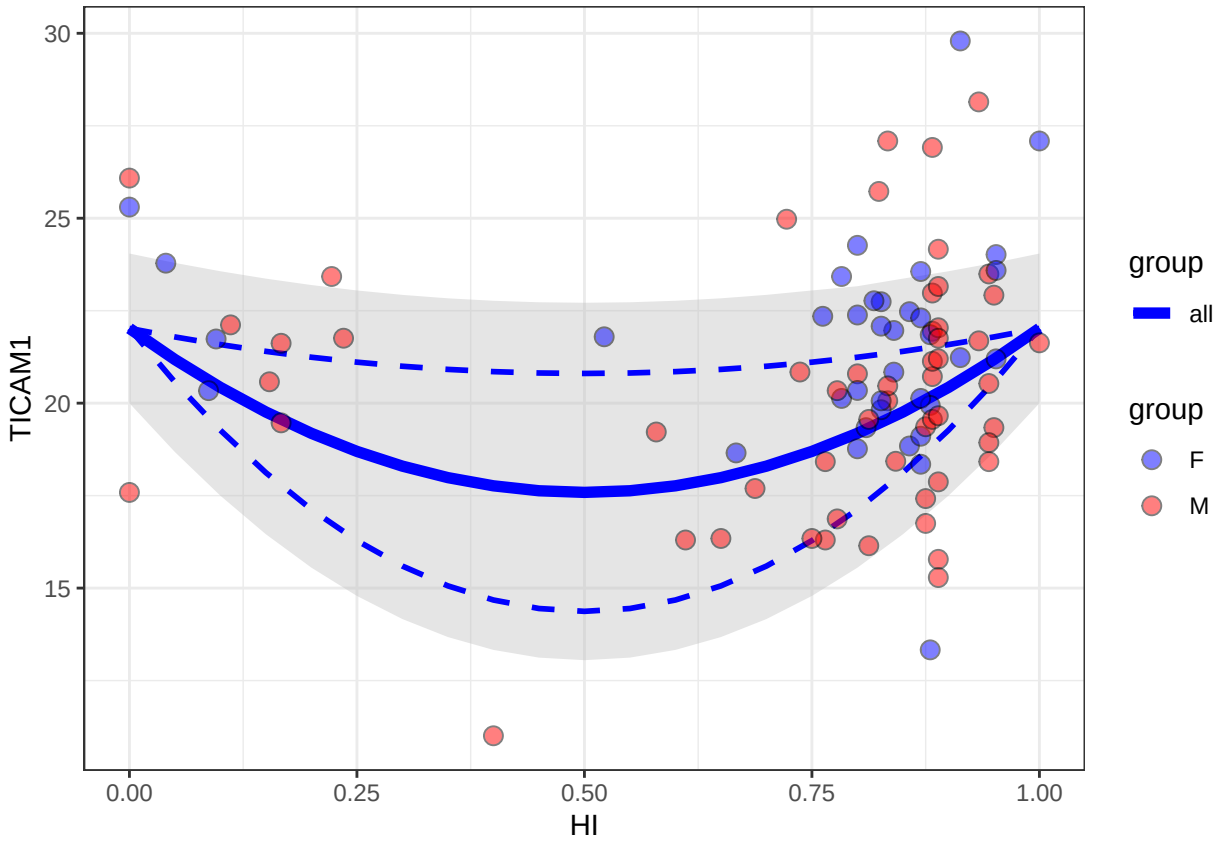
```
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 21.9688426 22.0365533  0.4023044  5.0000000
##
## Log-likelihood: -244.39
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 22.4672364  0.3940033  5.0000000
##
## Log-likelihood: -96.92
## 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
## 21.653890  0.401876  5.000000
```

```
##
## Log-likelihood: -147.05
## 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
## 22.1162202 22.5998810  0.3965689  5.0000000
##
## Log-likelihood: -96.9
## 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
## 21.9429451 21.5967115  0.4078008  5.0000000
##
## Log-likelihood: -147.03
## 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.
```




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

speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "TNF")

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 2.7   1 0.0202517
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF     pvalue
## 1 2.65   1 0.02129112
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 1.13   1 0.1331975
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 1.43   1 0.0910977
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.98   1 0.1605106
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.41   1 0.09265794
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.02   1 0.8343673
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 1.08   3 0.5404548
## [1] "Testing H3 vs H1"
##   dLL dDF    pvalue
## 1 1.2   4 0.6631581
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.14   2 0.8677416
```

```
##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
## 22.7806319  0.3726511  5.0000000
```

```
##
## Log-likelihood: -246.87
## 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
## 23.0550609 22.6818111  0.3723836  5.0000000
##
## Log-likelihood: -246.85
## 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
## 23.5611851  0.3690028  5.0000000
##
## Log-likelihood: -98.56
## 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
## 22.1039144  0.3620964  5.0000000
##
## Log-likelihood: -147.23
## 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
## 24.5965003 23.0938754  0.3552385  5.0000000
##
## Log-likelihood: -98.43
## 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
## 21.8004199 22.1835953  0.3585539  5.0000000
##
## Log-likelihood: -147.22
## 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.
```