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 == "FALSE")
```

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/RtmpZyjaa0/remotes346f726d0322cf/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

```r
x <- field$IDO1
```

```r
# 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")))
}
```

```r
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))
}
```

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

Functions for testing distributions

```
## $fit
##        mean           sd
##    13.5366293    3.3682927
##   ( 0.3492757) ( 0.2469752)
##
## $loglik
## [1] -244.901
##
## $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
##     4.0246825    14.8671522
##   ( 0.2966289) ( 0.4074253)
##
## $loglik
## [1] -248.7806
##
## $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"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
```

```
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.25   1 0.4771524
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.13   1 0.6131639
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.08   1 0.695912
## [1] "Testing H2 groupB no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.2   1 0.5250571
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.68   1 0.2443054
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.17   1 0.5584521
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.07   1 0.6985756
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.26   3 0.9133874
## [1] "Testing H3 vs H1"
##    dLL dDF pvalue
## 1 2.26   4 0.33981
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 2.07   2 0.1256684
```

```
##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
## 20.5424780  0.1281996  4.8935819
##
## Log-likelihood: -277.96
## 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
## 20.64402696 20.16505099  0.09957123  4.90650573
## 
## Log-likelihood: -277.89
## 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
## 20.39525575  0.08870288  5.00000000
## 
## Log-likelihood: -134.8
## 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
## 20.701420   0.191281   4.605496
## 
## Log-likelihood: -142.9
## Best method: bobyqa
```
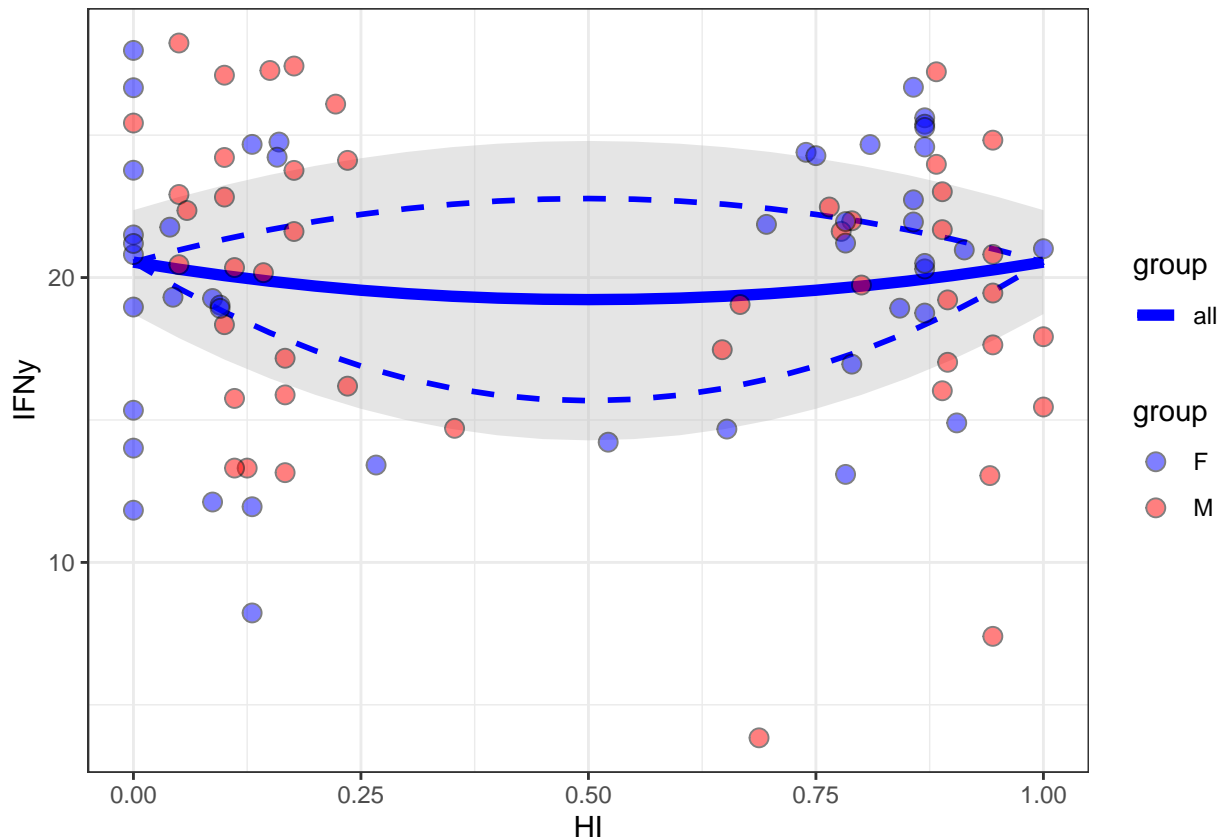
```
## 
## 
## $H3
## $H3$groupA
## 
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
## 
## Coefficients:
##         L1         L2      alpha    myshape
## 20.1269301 22.9708398  0.3325826  5.0000000
## 
## Log-likelihood: -133.95
## 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
## 21.9080596 19.1961242  0.1727863  4.7539251
## 
## Log-likelihood: -141.68
## Best method: bobyqa
```

```r
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 0.08   1 0.6928902
## [1] "Testing H1 no alpha vs alpha"
##     dLL dDF    pvalue
## 1 0.07   1 0.7050402
## [1] "Testing H2 groupA no alpha vs alpha"
##     dLL dDF    pvalue
## 1 0.15   1 0.5817916
## [1] "Testing H2 groupB no alpha vs alpha"
##     dLL dDF    pvalue
## 1 0.01   1 0.9009118
## [1] "Testing H3 groupA no alpha vs alpha"
##     dLL dDF    pvalue
## 1 0.06   1 0.7248703
## [1] "Testing H3 groupB no alpha vs alpha"
##     dLL dDF    pvalue
## 1 0.01   1 0.8946092
## [1] "Testing H1 vs H0"
##   dLL dDF   pvalue
## 1   0   1 0.998095
## [1] "Testing H2 vs H0"
##     dLL dDF    pvalue
## 1 0.46   3 0.8213957
## [1] "Testing H3 vs H1"
##     dLL dDF    pvalue
## 1 0.51   4 0.9077678
## [1] "Testing H3 vs H2"
##     dLL dDF    pvalue
## 1 0.05   2 0.9528307
```

```
##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
## 20.85139081  0.06288493  5.00000000
##
## Log-likelihood: -241.37
```

```
## 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
## 20.85057018 20.85341543  0.06299882  5.00000000
##
## Log-likelihood: -241.37
## 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
## 20.6509448  0.1145553  5.0000000
##
## Log-likelihood: -117.78
## 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
## 21.07334293  0.03046427  5.00000000
```
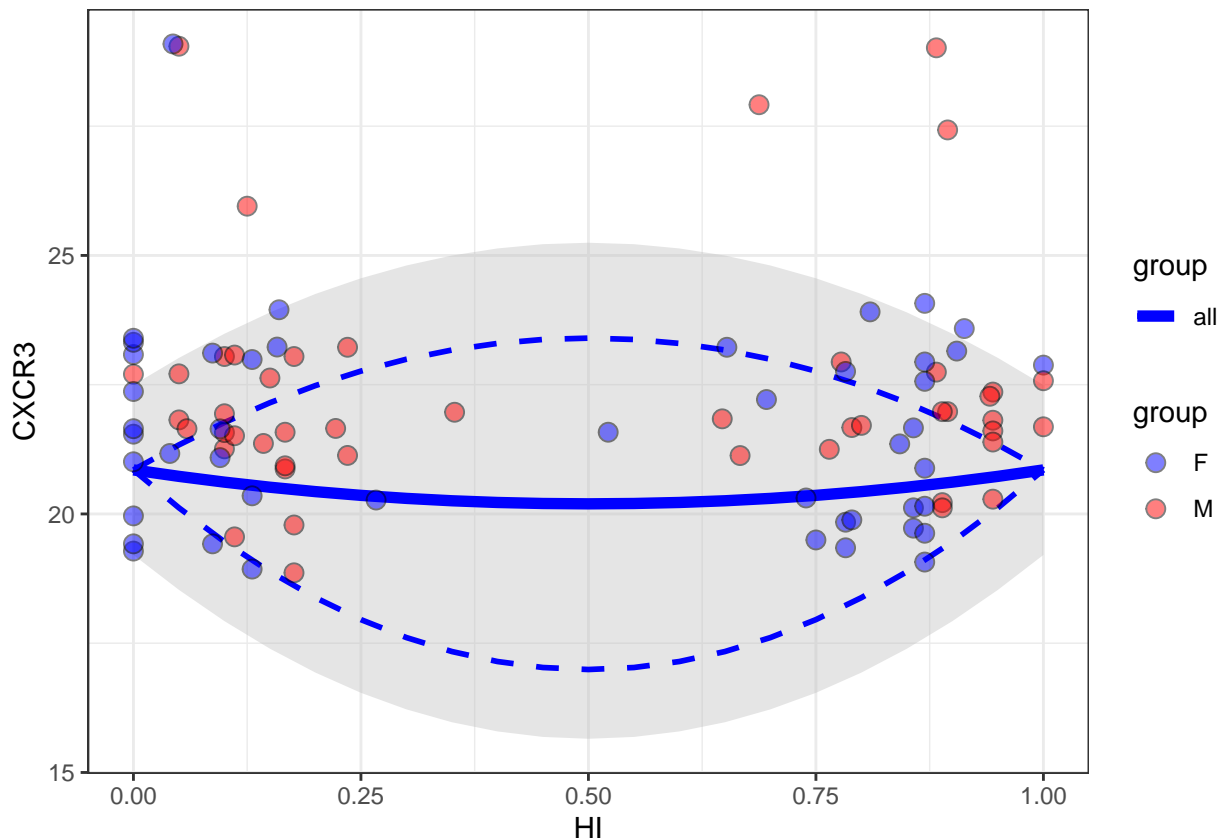
```
## 
## Log-likelihood: -123.13
## Best method: bobyqa
## 
## 
## $H3
## $H3$groupA
## 
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     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
## 20.71277797 20.32572499  0.08758269  5.00000000
## 
## Log-likelihood: -117.76
## 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
## 20.89918501 21.28004103  0.03252853  5.00000000
## 
## Log-likelihood: -123.1
## Best method: bobyqa
```

```r
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.
```



```r
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 0.07   1 0.7093301
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.06   1 0.7302717
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.8697426
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.04   1 0.7711233
## [1] "Testing H3 groupA no alpha vs alpha"
##   dLL dDF   pvalue
## 1   0   1 0.981184
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.04   1 0.7736477
## [1] "Testing H1 vs H0"
##   dLL dDF  pvalue
## 1   0   1 0.92726
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.18   3 0.9496095
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.21   4 0.9805638
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.04   2 0.9620527
##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
## 22.13526238 -0.06396086  5.00000000
```

```
##
## Log-likelihood: -248.38
## 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
## 22.09804980 22.21113912 -0.06049139  5.00000000
##
## Log-likelihood: -248.38
## 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
## 21.97515246 -0.03683734  5.00000000
##
## Log-likelihood: -122.71
## 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
## 22.32962840  -0.07696131  5.00000000
##
## Log-likelihood: -125.5
## 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
## 21.873080083 22.384847947 -0.005908704  5.000000000
##
## Log-likelihood: -122.67
## 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
## 22.3905167  22.2716247  -0.0760752  5.0000000
##
## Log-likelihood: -125.49
## Best method: L-BFGS-B
```

```
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.
```
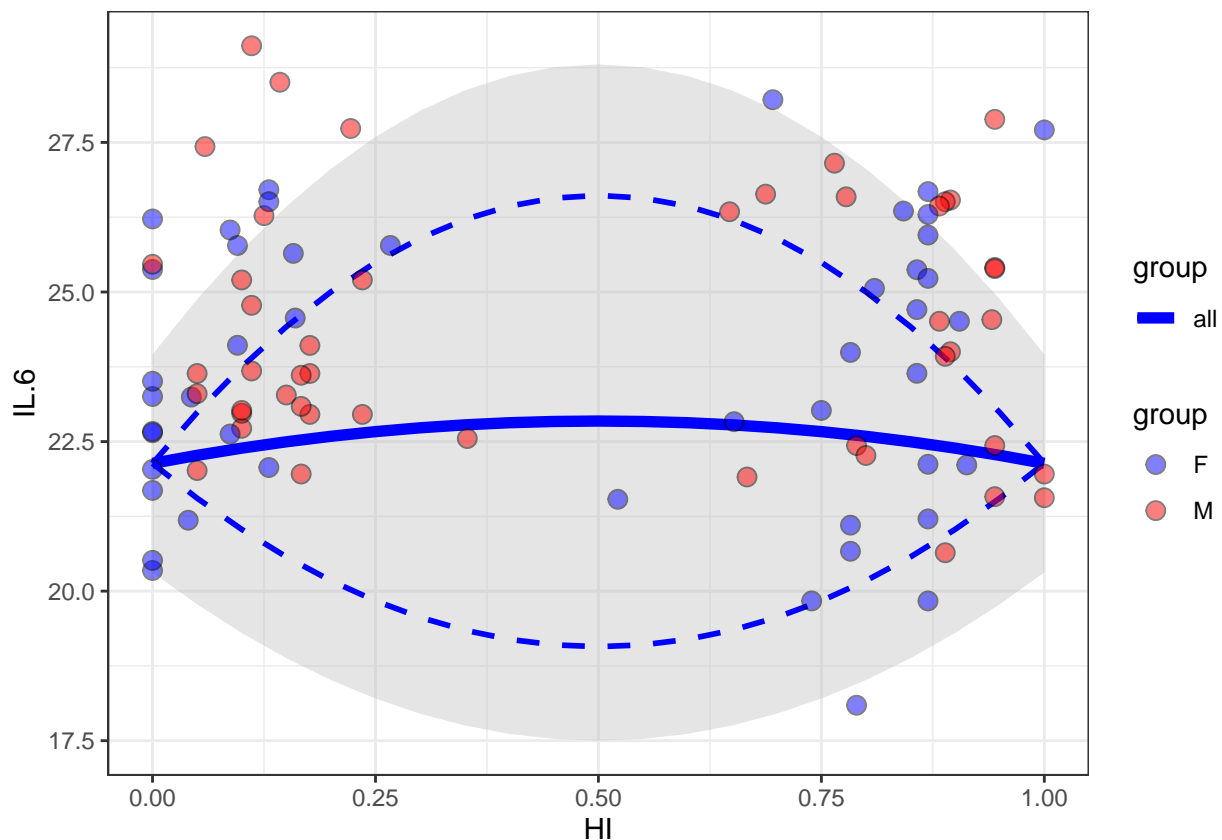


```r
field$Sex <- as.factor(field$Sex)

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 0.59   1 0.2767055
## [1] "Testing H1 no alpha vs alpha"
##     dLL dDF    pvalue
## 1 0.33   1 0.4157139
## [1] "Testing H2 groupA no alpha vs alpha"
##     dLL dDF    pvalue
## 1 0.59   1 0.2772499
## [1] "Testing H2 groupB no alpha vs alpha"
##     dLL dDF    pvalue
## 1 0.09   1 0.6729248
## [1] "Testing H3 groupA no alpha vs alpha"
##     dLL dDF    pvalue
## 1 0.25   1 0.4763257
## [1] "Testing H3 groupB no alpha vs alpha"
##     dLL dDF    pvalue
## 1 0.08   1 0.6833205
## [1] "Testing H1 vs H0"
##     dLL dDF    pvalue
## 1 0.26   1 0.4692568
## [1] "Testing H2 vs H0"
##   dLL dDF    pvalue
## 1 0.1   3 0.9776686
## [1] "Testing H3 vs H1"
##     dLL dDF    pvalue
## 1 0.06   4 0.9983191
## [1] "Testing H3 vs H2"
##     dLL dDF    pvalue
## 1 0.22   2 0.8015243
```

```r
##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
## 24.1144356  0.1669175   5.0000000
##
## Log-likelihood: -251.21
## 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
## 24.3841498 23.3885808  0.1317113   5.0000000
##
## Log-likelihood: -250.95
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha     myshape
## 24.1793776  0.2101329   5.0000000
##
## Log-likelihood: -124.3
## 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
## 23.9305399  0.1080356  5.0000000
##
## Log-likelihood: -126.81
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha     myshape
## 24.3342270 23.4592561  0.1628995  5.0000000
##
## Log-likelihood: -124.22
## Best method: bobyqa
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha     myshape
## 24.3645165 23.3781655  0.1032649  5.0000000
##
## Log-likelihood: -126.67
## 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.
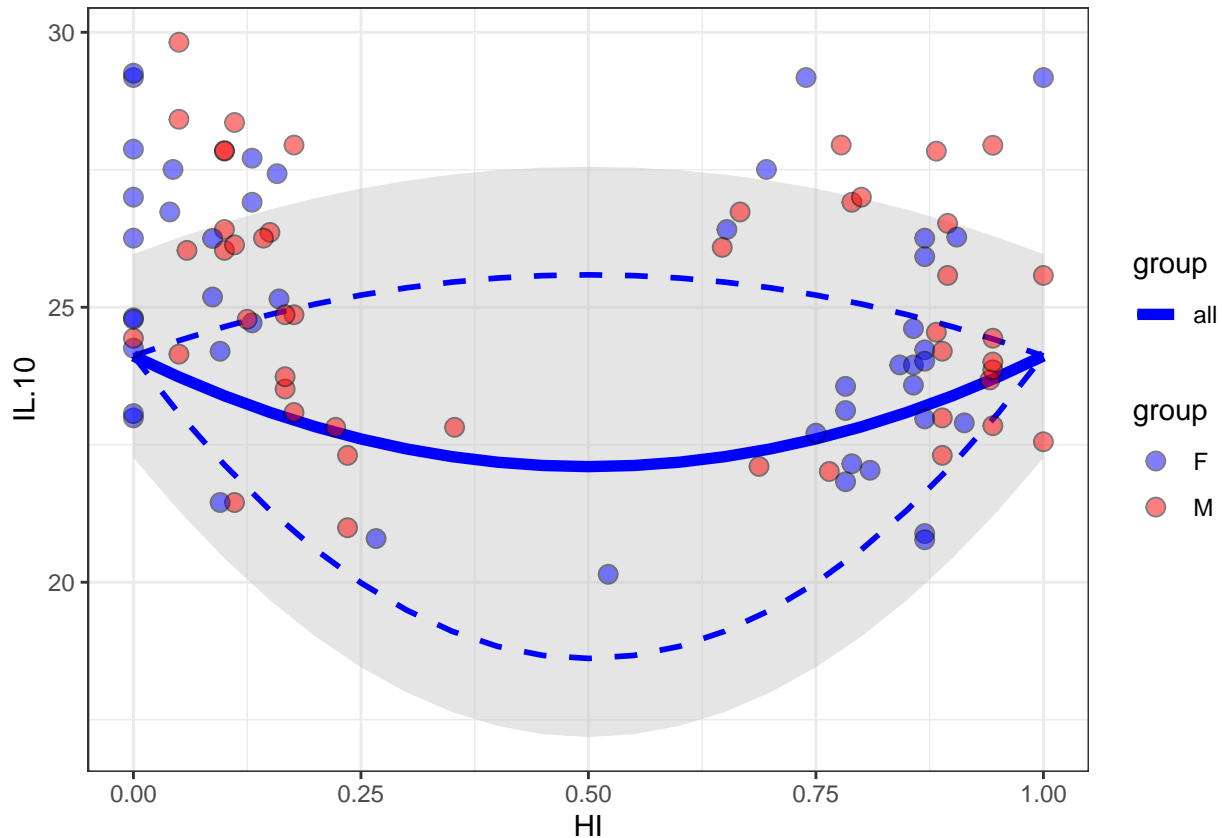

```
field$Sex <- as.factor(field$Sex)

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"

23
```

```
## [1] "Did converge"
## [1] "Fitting for groupA : F"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Fitting for groupB : M"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##     dLL dDF     pvalue
## 1 0.25   1 0.4808932
## [1] "Testing H1 no alpha vs alpha"
##     dLL dDF     pvalue
## 1 0.07   1 0.7122476
## [1] "Testing H2 groupA no alpha vs alpha"
##     dLL dDF     pvalue
## 1 0.13   1 0.6107557
## [1] "Testing H2 groupB no alpha vs alpha"
##     dLL dDF     pvalue
## 1 0.06   1 0.7302787
## [1] "Testing H3 groupA no alpha vs alpha"
##     dLL dDF     pvalue
## 1 0.04   1 0.7740159
## [1] "Testing H3 groupB no alpha vs alpha"
##     dLL dDF     pvalue
## 1 0.07   1 0.7039566
## [1] "Testing H1 vs H0"
##     dLL dDF     pvalue
## 1 0.59   1 0.2757684
## [1] "Testing H2 vs H0"
##    dLL dDF     pvalue
## 1 0.24   3 0.9240296
## [1] "Testing H3 vs H1"
##     dLL dDF   pvalue
## 1 0.47   4 0.920064
## [1] "Testing H3 vs H2"
##     dLL dDF     pvalue
## 1 0.82   2 0.4399012
```

```
##All
print(IL.13)
```

```
## $H0
##
```

```
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 16.5841991  0.2017999  3.0205159
##
## Log-likelihood: -292.89
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 17.0147414 15.3405674  0.1122689  3.0132983
##
## Log-likelihood: -292.29
## 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
## 16.934435  0.179060  3.108148
##
## Log-likelihood: -144.86
## 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
## 16.0729548   0.1787235   2.9469949
##
## Log-likelihood: -147.79
## 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
## 17.0641554  16.2763360   0.1162702   3.1010729
##
## Log-likelihood: -144.8
## Best method: bobyqa
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1          L2       alpha     myshape
## 17.3385300  14.7064298   0.1905704   2.9682903
```

```
##
## Log-likelihood: -147.02
## 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.
```



```
field$Sex <- as.factor(field$Sex)
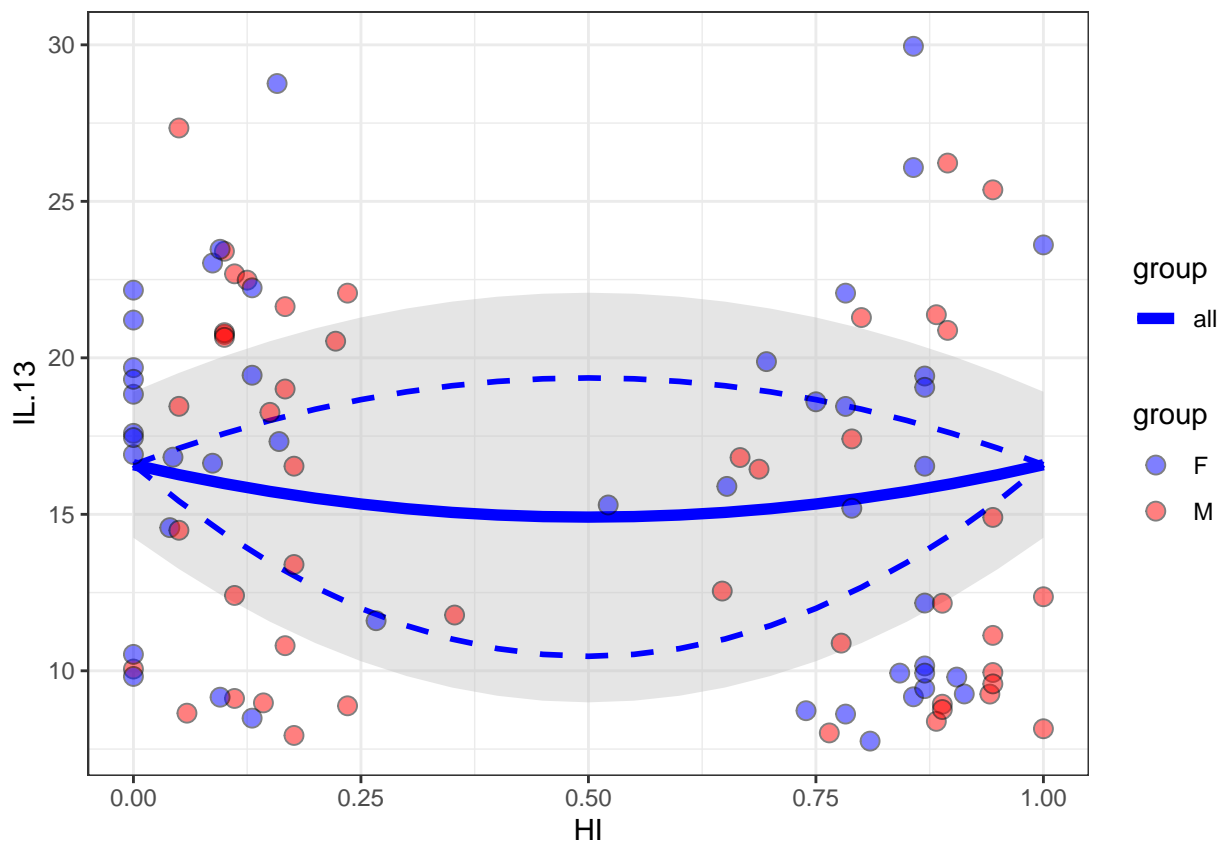
speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "IL1RN")

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 0.94   1 0.1710034
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.77   1 0.2143179
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.45   1 0.340802
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 2.55   1 0.0239246
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.41   1 0.3647958
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 2.72   1 0.0196231
## [1] "Testing H1 vs H0"
##   dLL dDF    pvalue
## 1 0.1   1 0.6515893
## [1] "Testing H2 vs H0"
##    dLL dDF     pvalue
## 1 4.33   3 0.03417428
## [1] "Testing H3 vs H1"
##    dLL dDF     pvalue
## 1 4.94   4 0.04264957
```

```
## [1] "Testing H3 vs H2"
##     dLL dDF    pvalue
## 1 0.71   2 0.4927214
```

```
##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
## 11.8558697 -0.2655501  4.0575474
##
## Log-likelihood: -241.47
## 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
## 11.7265857 12.0971682 -0.2458925  4.0590390
##
## Log-likelihood: -241.37
## 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
## 12.4585340   0.2198072   4.3013439
##
## Log-likelihood: -114.25
## 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
## 11.281564  -0.741879   4.269951
##
## Log-likelihood: -122.89
## 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
## 12.4346366  12.5241243   0.2268313   4.2984601
##
## Log-likelihood: -114.25
## 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"]],
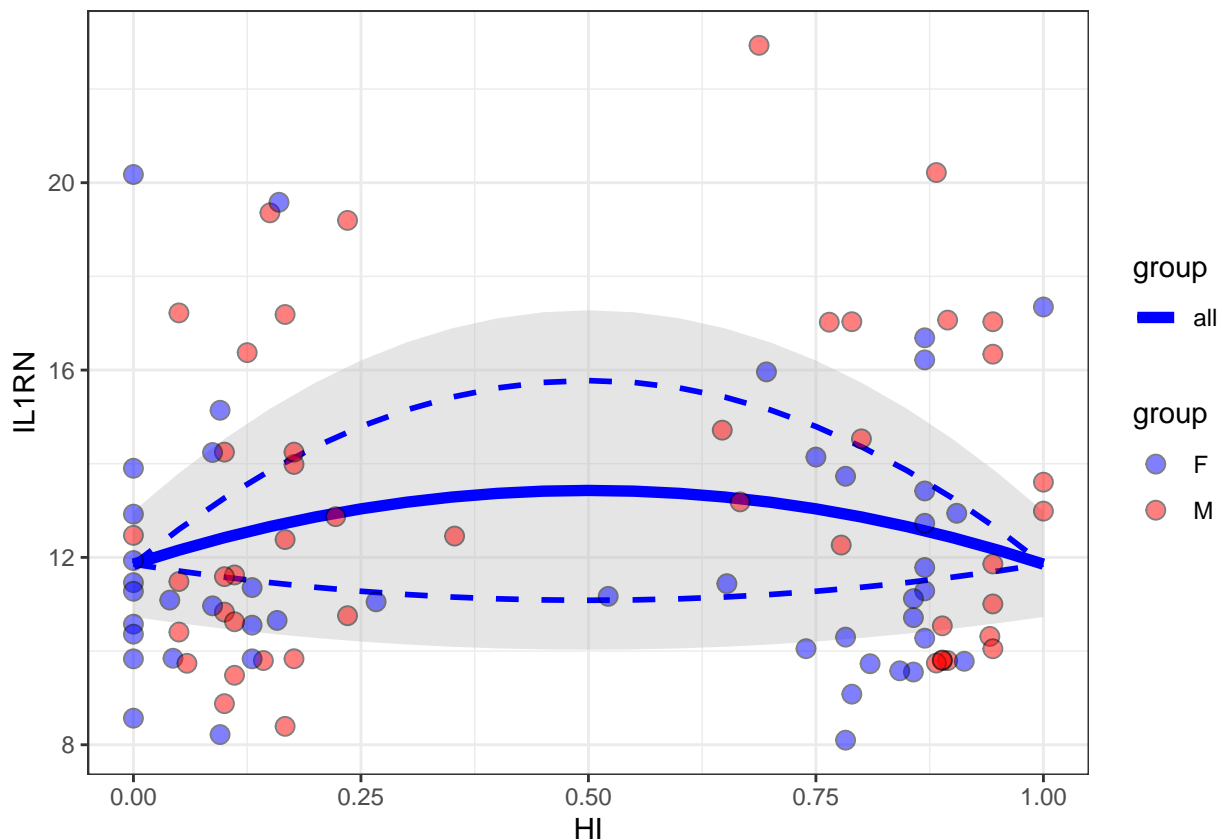```

```
##         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
## 10.6297652 11.8450706 -0.7770197   4.3399845
##
## Log-likelihood: -122.18
## Best method: bobyqa
```

```
bananaPlot(mod = IL1RN$H0,
           data = field,
           response = "IL1RN",
           group = "Sex") +
  scale_fill_manual(values = c("blue", "red")) +
  scale_color_manual(values = c("blue", "red")) +
  theme_bw()
```

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

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



```
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 0.08   1 0.6928902
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.07   1 0.7050402
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.15   1 0.5817916
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.9009118
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.06   1 0.7248703
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.8946092
## [1] "Testing H1 vs H0"
##   dLL dDF   pvalue
## 1   0   1 0.998095
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.46   3 0.8213957
```

```
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.51   4 0.9077678
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.05   2 0.9528307
```

```
##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
## 20.85139081  0.06288493  5.00000000
##
## Log-likelihood: -241.37
## 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
## 20.85057018 20.85341543  0.06299882  5.00000000
##
## Log-likelihood: -241.37
## 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"]],
```
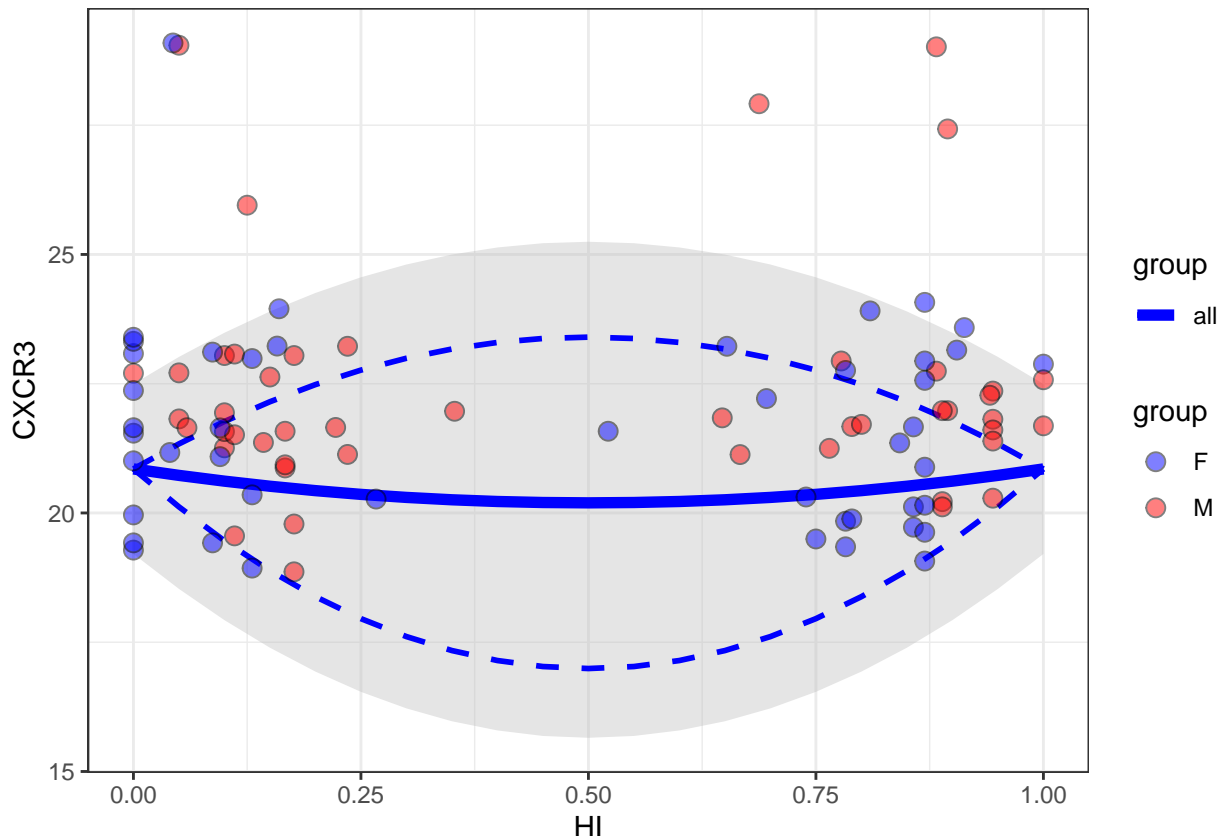
```
##        myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##            alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##       control = config$control)
##
## Coefficients:
##          L1       alpha     myshape
## 20.6509448   0.1145553   5.0000000
##
## Log-likelihood: -117.78
## 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
## 21.07334293   0.03046427   5.00000000
##
## Log-likelihood: -123.13
## 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
## 20.71277797   20.32572499   0.08758269   5.00000000
##
## Log-likelihood: -117.76
## 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
## 20.89918501 21.28004103  0.03252853  5.00000000
##
## Log-likelihood: -123.1
## Best method: bobyqa
```

```r
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 0.04   1 0.7752172
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.02   1 0.8497243
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.06   1 0.7191263
## [1] "Testing H2 groupB no alpha vs alpha"
##   dLL dDF   pvalue
## 1   0   1 0.930879
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.06   1 0.721972
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.8712246
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.09   1 0.6756336
## [1] "Testing H2 vs H0"
```

```
##     dLL dDF     pvalue
## 1 0.12    3 0.971501
## [1] "Testing H3 vs H1"
##   dLL dDF     pvalue
## 1 0.2    4 0.9822669
## [1] "Testing H3 vs H2"
##     dLL dDF     pvalue
## 1 0.17    2 0.8431625
```

```
##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
## 20.78367523  0.04593364  5.00000000
##
## Log-likelihood: -237.25
## 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
## 20.94298032 20.45795242  0.03128766  5.00000000
##
## Log-likelihood: -237.16
## 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
## 20.70188527   0.07494595   5.00000000
##
## Log-likelihood: -116.53
## 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
## 20.86194313   0.02191731   5.00000000
##
## Log-likelihood: -120.6
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##         scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##         start = start, method = config$method, optimizer = config$optimizer,
##         data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##             alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##         upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##             alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##         control = config$control)
##
## Coefficients:
##          L1           L2        alpha      myshape
## 20.67111071  20.84588499   0.08667083   5.00000000
##
## Log-likelihood: -116.52
## 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
## 21.39226455 20.43942387  0.04060682  5.00000000
##
## Log-likelihood: -120.43
## Best method: bobyqa
```

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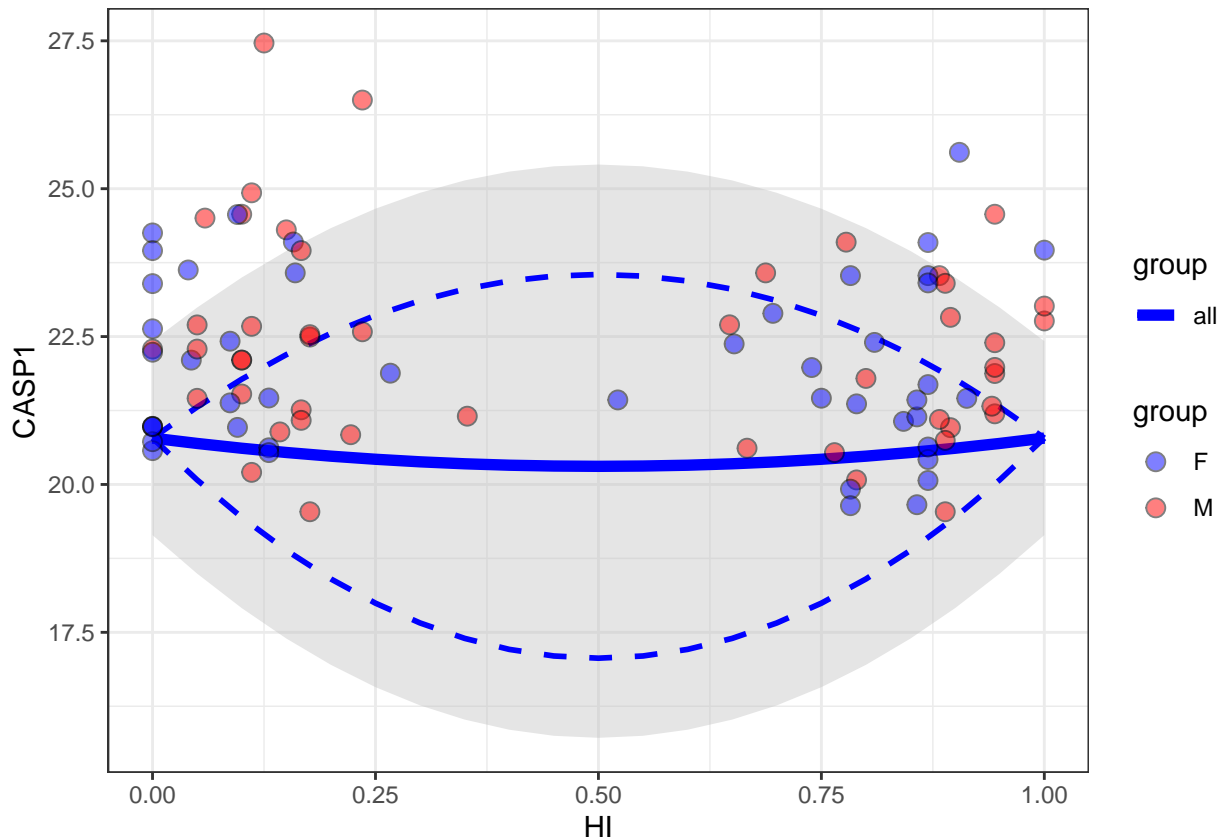
```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```

```
## 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   1 0.9963293
## [1] "Testing H1 no alpha vs alpha"
##   dLL dDF   pvalue
## 1   0   1 0.9253699
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.05   1 0.7504693
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.08   1 0.6980899
## [1] "Testing H3 groupA no alpha vs alpha"
##   dLL dDF   pvalue
## 1   0   1 0.9860847
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.07   1 0.7024633
## [1] "Testing H1 vs H0"
##    dLL dDF   pvalue
## 1 0.14   1 0.5979112
## [1] "Testing H2 vs H0"
##    dLL dDF   pvalue
## 1 0.16   3 0.9560483
## [1] "Testing H3 vs H1"
##    dLL dDF   pvalue
## 1 0.19   4 0.9838563
## [1] "Testing H3 vs H2"
##    dLL dDF   pvalue
## 1 0.17   2 0.8436226
```

```
##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
## 19.7549380715 -0.0007690849  5.0000000000
##
## Log-likelihood: -240.71
```

```
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 19.54773359 20.12907922  0.01591917  5.00000000
##
## Log-likelihood: -240.57
## 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
## 19.37582553 -0.06972773  5.00000000
##
## Log-likelihood: -120.1
## 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
## 20.29726112  0.09799529  5.00000000
```
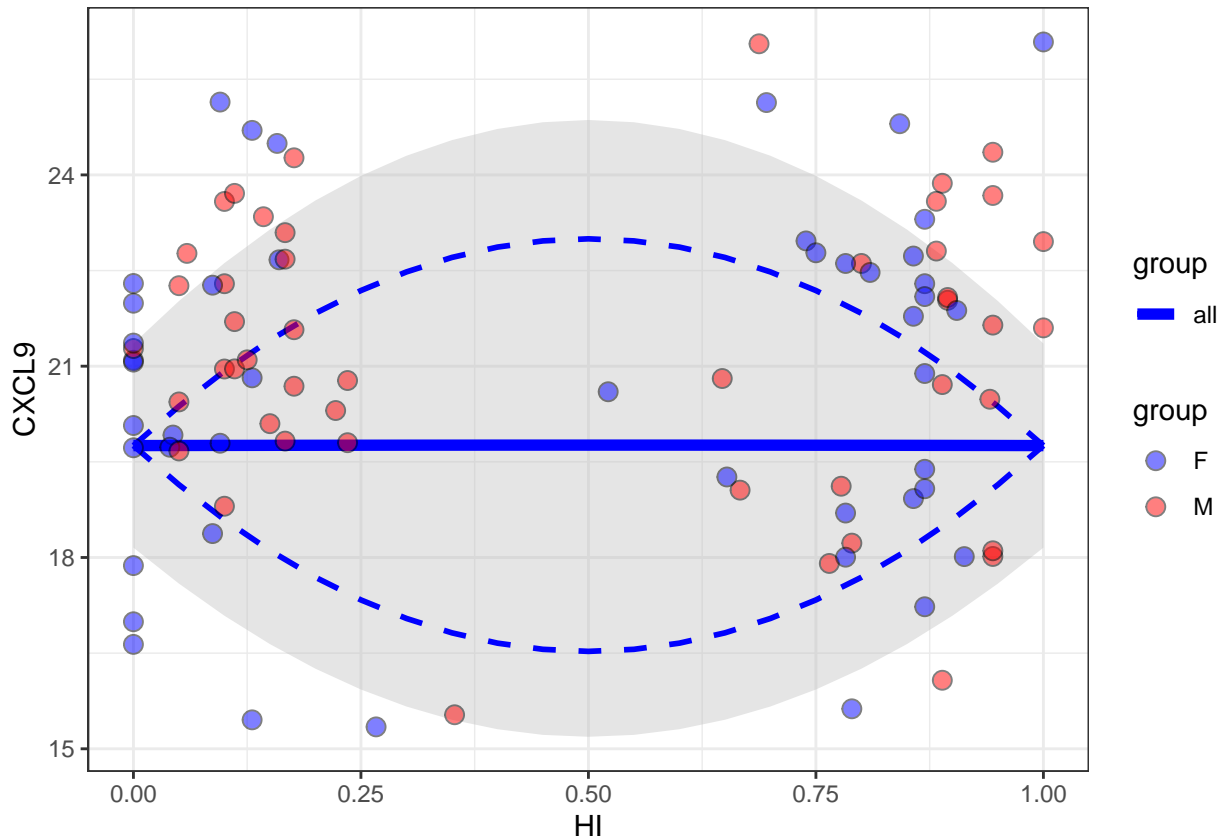
```
##
## Log-likelihood: -120.45
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     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
## 19.166764286 20.148690084 -0.004281179  5.000000000
##
## Log-likelihood: -119.93
## 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
## 20.27030561 20.31976523  0.09726483  5.00000000
##
## Log-likelihood: -120.45
## 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"
## [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 0.51   1 0.314253
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.38   1 0.3833321
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF  pvalue
## 1 0.82   1 0.201004
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 1.67   1 0.06752519
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.98   1 0.1613542
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 1.81   1 0.05711729
## [1] "Testing H1 vs H0"
##    dLL dDF   pvalue
## 1 0.15   1 0.5791198
## [1] "Testing H2 vs H0"
##   dLL dDF   pvalue
## 1 4.4   3 0.0321927
## [1] "Testing H3 vs H1"
##   dLL dDF    pvalue
## 1 4.9   4 0.04408805
## [1] "Testing H3 vs H2"
##    dLL dDF   pvalue
## 1 0.65   2 0.5200677
```

```
##All
print(ID01)
```

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

```
##        alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha     myshape
## 12.9431656 -0.2017249   4.0650201
##
## Log-likelihood: -248.27
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha     myshape
## 12.7712879 13.2665994 -0.1778444   4.0686265
##
## Log-likelihood: -248.12
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha     myshape
## 13.7330931  0.3141663   4.4386753
##
## Log-likelihood: -116.64
## 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
## 12.4519126 -0.6042104  4.1483562
##
## Log-likelihood: -127.24
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha     myshape
## 13.5571992 14.3183570  0.3747684  4.4287475
##
## Log-likelihood: -116.47
## 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$control)
##
## Coefficients:
##         L1         L2      alpha     myshape
## 11.8136110 12.9680793 -0.6425961  4.1991220
##
## Log-likelihood: -126.75
## Best method: bobyqa
bananaPlot(mod = IDO1$H0,
          data = field,
```

```
            response = "IDO1",
            group = "Sex") +
  scale_fill_manual(values = c("blue", "red")) +
  scale_color_manual(values = c("blue", "red")) +
  theme_bw()
```
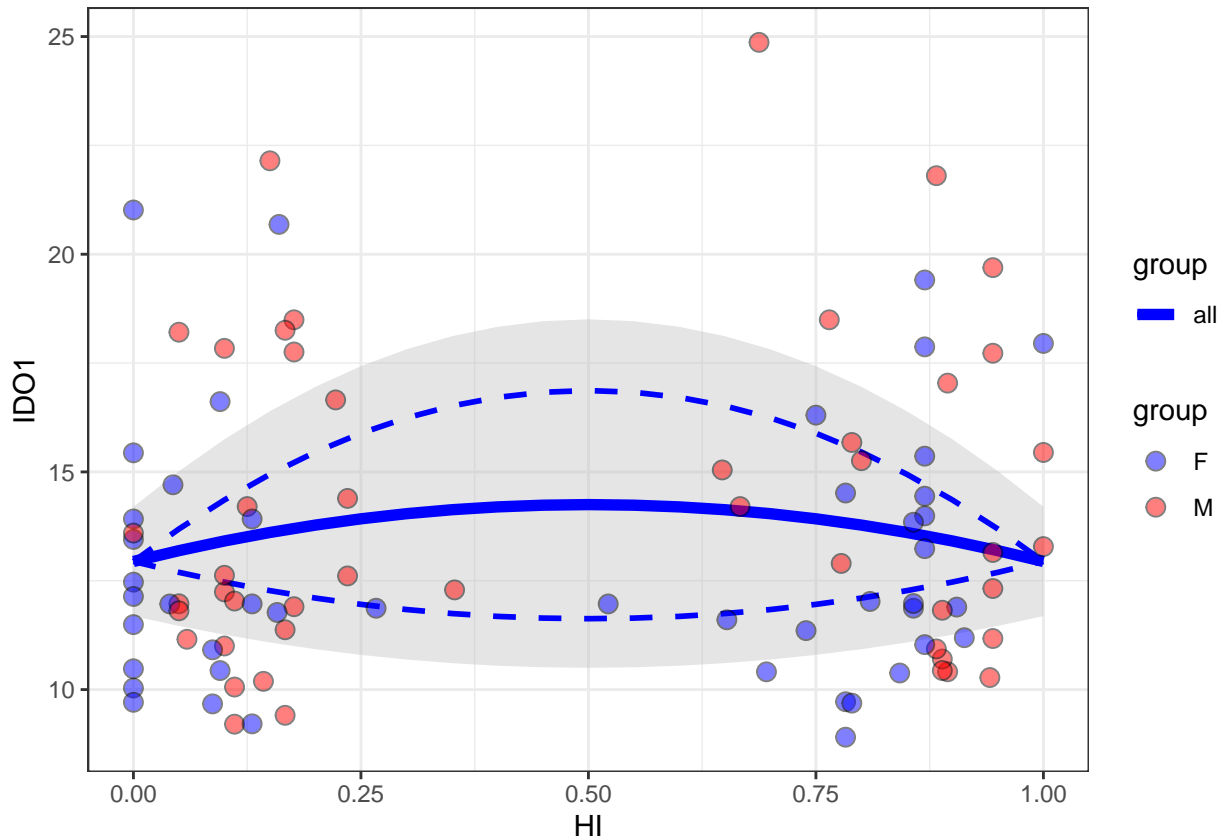
```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```

```
## 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")

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"

## Warning in bbmle::mle2(response ~ dweibull(shape = 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 0.8456101
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.02   1 0.8506396
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.32   1 0.4221519
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.74   1 0.06231174
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.46   1 0.3349712
## [1] "Testing H3 groupB no alpha vs alpha"
```

```
##    dLL dDF      pvalue
## 1 1.7    1 0.06489869
## [1] "Testing H1 vs H0"
##   dLL dDF     pvalue
## 1   0   1 0.9858853
## [1] "Testing H2 vs H0"
##    dLL dDF     pvalue
## 1 2.67   3 0.1482466
## [1] "Testing H3 vs H1"
##    dLL dDF     pvalue
## 1 2.84   4 0.2241561
## [1] "Testing H3 vs H2"
##    dLL dDF     pvalue
## 1 0.17   2 0.8443836
```

```
##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
## 10.25808578 -0.02544155  4.97270528
##
## Log-likelihood: -202.8
## 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
## 10.25423830 10.26439275 -0.02501915  4.97272252
##
## Log-likelihood: -202.8
## Best method: bobyqa
##
```

```
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha     myshape
## 10.5291394  0.1421704   4.5720509
##
## Log-likelihood: -103.27
## 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
##  9.1786913 -0.4858919   5.0000000
##
## Log-likelihood: -96.86
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha     myshape
## 10.3687249 10.9265708  0.1834953   4.5957871
```

```
##
## Log-likelihood: -103.1
## 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
##   9.1675163   9.1854042  -0.4873618   5.0000000
##
## Log-likelihood: -96.86
## Best method: L-BFGS-B
```

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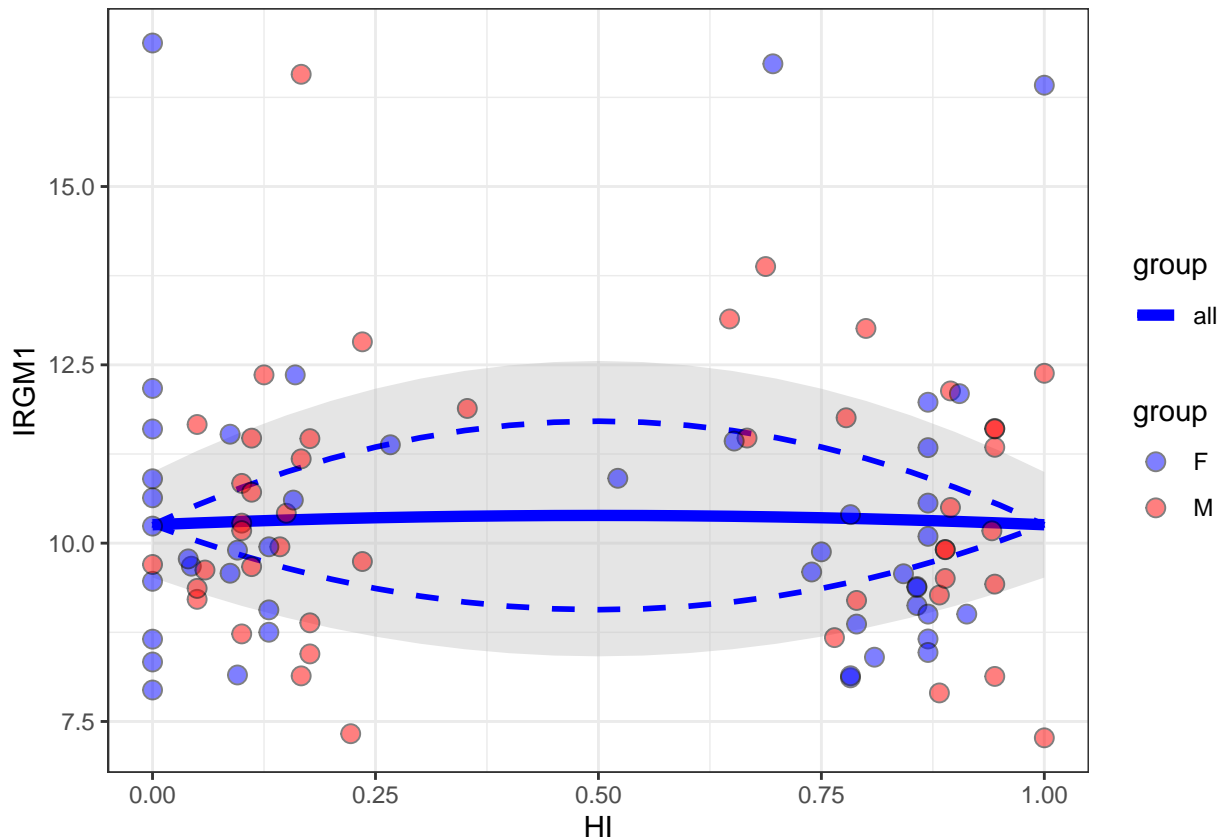
```
field$Sex <- as.factor(field$Sex)

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"

## Warning in bbmle::mle2(response ~ dweibull(shape = 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"
## [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"

## Warning in bbmle::mle2(response ~ dweibull(shape = 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 2.41   1 0.0282651
```

```
## [1] "Testing H1 no alpha vs alpha"
##     dLL dDF     pvalue
## 1 2.28   1 0.03254679
## [1] "Testing H2 groupA no alpha vs alpha"
##     dLL dDF     pvalue
## 1 0.53   1 0.3016715
## [1] "Testing H2 groupB no alpha vs alpha"
##     dLL dDF     pvalue
## 1 2.09   1 0.04092962
## [1] "Testing H3 groupA no alpha vs alpha"
##     dLL dDF     pvalue
## 1 0.09   1 0.6744969
## [1] "Testing H3 groupB no alpha vs alpha"
##     dLL dDF     pvalue
## 1 1.84   1 0.05521473
## [1] "Testing H1 vs H0"
##     dLL dDF     pvalue
## 1 0.16   1 0.5710236
## [1] "Testing H2 vs H0"
##     dLL dDF     pvalue
## 1 0.32   3 0.8892378
## [1] "Testing H3 vs H1"
##     dLL dDF   pvalue
## 1 1.36   4 0.606687
## [1] "Testing H3 vs H2"
##   dLL dDF     pvalue
## 1 1.2   2 0.3005683
```

```
##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
## 15.9632485 -0.3732209  5.0000000
##
## Log-likelihood: -244.31
## 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
## 15.7509728 16.2474384 -0.3651888   5.0000000
##
## Log-likelihood: -244.15
## 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
## 16.3142473 -0.2431962   4.9359337
##
## Log-likelihood: -123.32
## 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
## 15.4374870 -0.5363109   5.0000000
##
## Log-likelihood: -120.67
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
```

```
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##        L1         L2      alpha    myshape
## 15.7678582 17.8745354 -0.1011647  5.0000000
##
## Log-likelihood: -122.17
## Best method: bobyqa
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##        L1         L2      alpha    myshape
## 15.7231654 15.2883343 -0.5091311  5.0000000
##
## Log-likelihood: -120.61
## Best method: bobyqa
```

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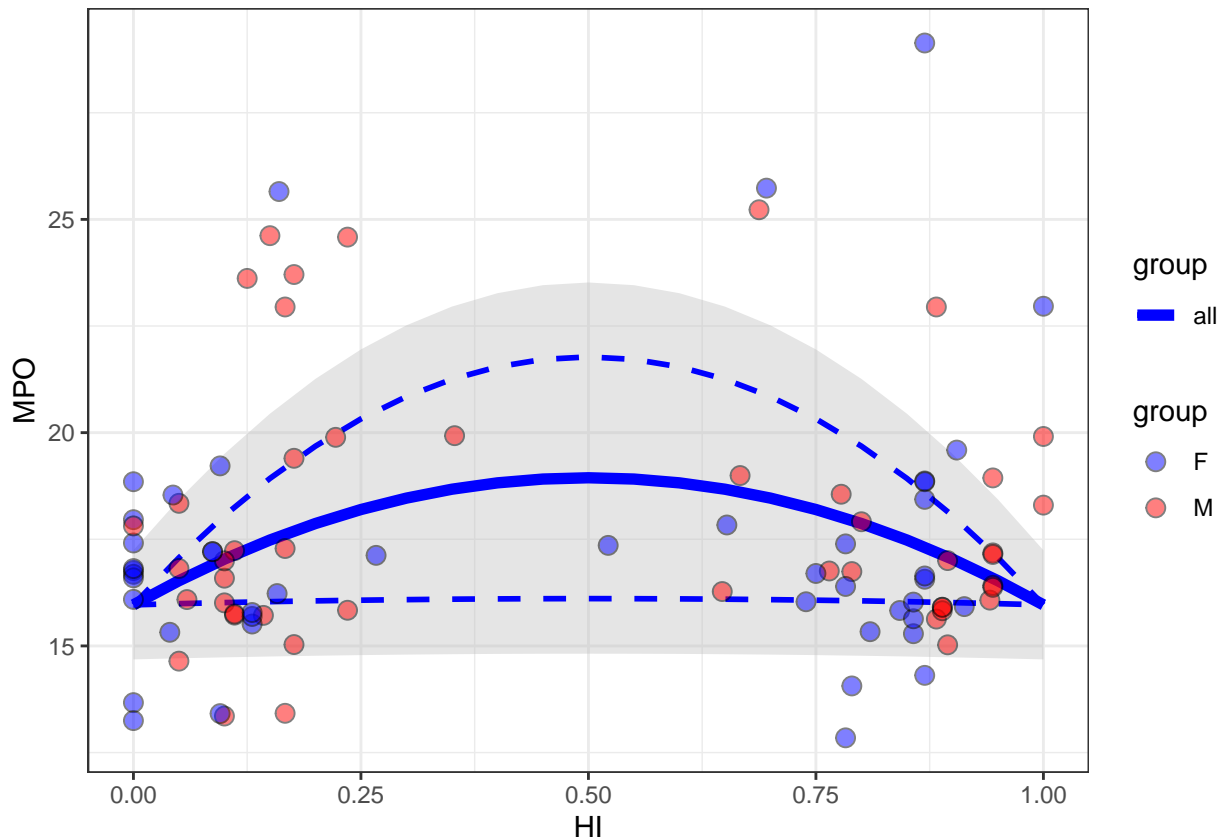
```r
field$Sex <- as.factor(field$Sex)

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

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

```
## [1] "Analysing data for response: MUC2"
## [1] "Fit for the response: MUC2"
## [1] "Fitting for all"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Fitting for groupA : F"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
```

```
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Fitting for groupB : M"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.09   1 0.6745602
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.05   1 0.7576221
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.65   1 0.2533174
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.84   1 0.1946314
## [1] "Testing H3 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.8   1 0.2052835
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.76   1 0.2187171
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.18   1 0.5501423
## [1] "Testing H2 vs H0"
##   dLL dDF    pvalue
## 1 1.8   3 0.3074465
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 2.74   4 0.2420038
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 1.11   2 0.3284586
```

```
##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
## 9.05387736 0.07257998 3.86015438
##
## Log-likelihood: -211
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2      alpha     myshape
## 9.20101661 8.81322458 0.05390445 3.87111345
##
## Log-likelihood: -210.82
## 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
## 9.4255700 0.2482641 3.7389279
##
## Log-likelihood: -106.36
## 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
##  8.0532855 -0.4514793   4.0931420
##
## Log-likelihood: -102.83
## 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
## 9.2532829 9.8332000 0.2919943 3.7575604
##
## Log-likelihood: -106.21
## 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$control)
##
## Coefficients:
##         L1         L2      alpha     myshape
##  8.7224354  7.6143447 -0.3864455   4.2656027
##
## Log-likelihood: -101.87
## Best method: L-BFGS-B
bananaPlot(mod = MUC2$H0,
           data = field,
           response = "MUC2",
           group = "Sex") +
```
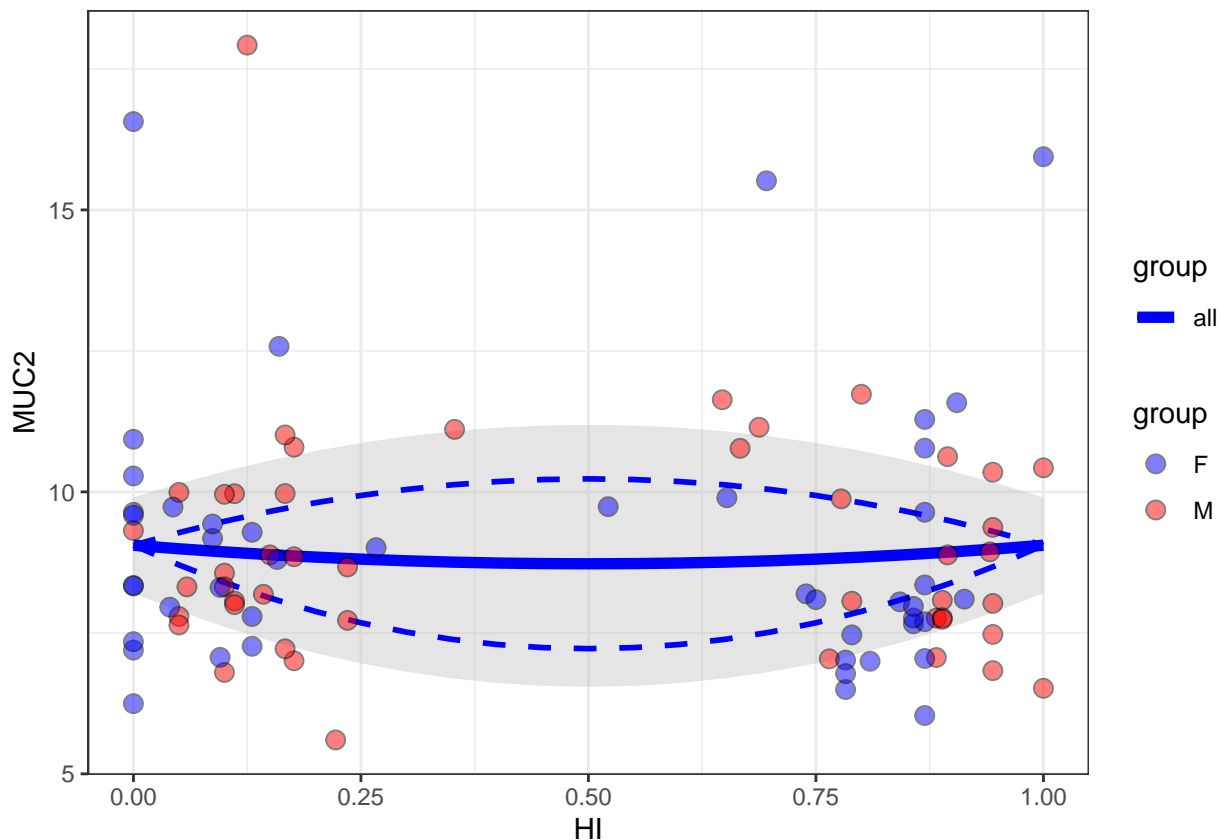
```
    scale_fill_manual(values = c("blue", "red")) +
    scale_color_manual(values = c("blue", "red")) +
    theme_bw()
```

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 0.17   1 0.5633805
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.21   1 0.5197812
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.06   1 0.7336679
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.71   1 0.2345556
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.46   1 0.3391632
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.38   1 0.3803574
## [1] "Testing H1 vs H0"
##   dLL dDF    pvalue
## 1 0.2   1 0.5268657
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 1.13   3 0.5206896
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 4.27   4 0.07365245
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 3.34   2 0.03536834
```

```
##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
##  9.0922421 -0.1642345  3.0215729
##
## Log-likelihood: -234.81
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha     myshape
##  9.3068756  8.7854275 -0.1830806  3.0344626
##
## Log-likelihood: -234.61
## 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
##  9.4952728  0.1088555  3.2095833
##
## Log-likelihood: -113.82
## Best method: bobyqa
```

```
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##        L1       alpha     myshape
##  8.3150025 -0.6611056   2.9318387
##
## Log-likelihood: -119.86
## 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
##  8.9284894 11.1171824   0.2974613   3.3786840
##
## Log-likelihood: -112.3
## Best method: bobyqa
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##        L1           L2       alpha     myshape
```

```
##  9.9411528  7.5926702 -0.4049394  3.1535572
##
## Log-likelihood: -118.04
## 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.
```
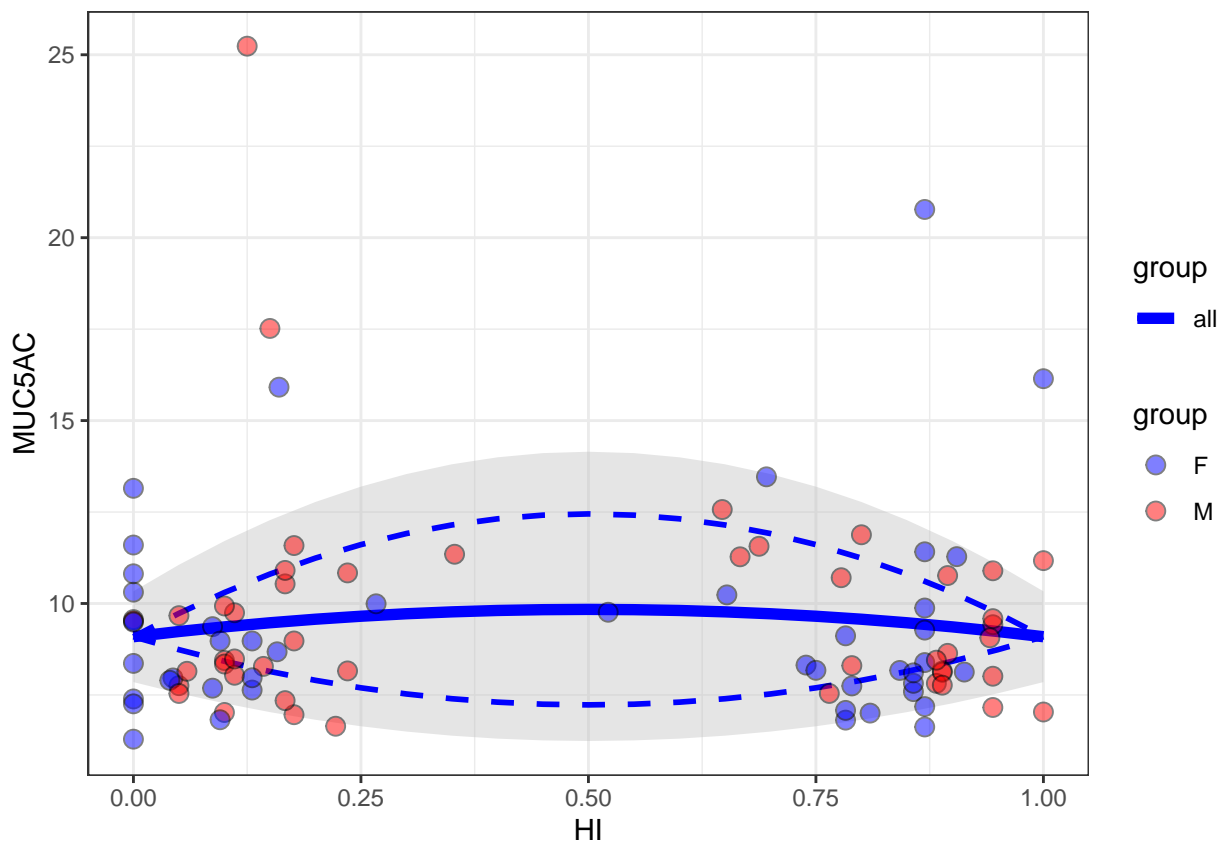


```
field$Sex <- as.factor(field$Sex)

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"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Fitting for groupB : M"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.25   1 0.4824313
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.09   1 0.6698607
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.62   1 0.2669381
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.03   1 0.7950275
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.07   1 0.7114594
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.02   1 0.8414895
## [1] "Testing H1 vs H0"
##   dLL dDF    pvalue
## 1 0.9   1 0.1798621
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 2.05   3 0.2501121
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
```

```
## 1 1.77    4 0.4724325
## [1] "Testing H3 vs H2"
##    dLL dDF     pvalue
## 1 0.61   2 0.5413296
```

```
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
## 15.0217055 -0.1978601  2.9286225
##
## Log-likelihood: -295.08
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 14.3230282 16.2637330 -0.1206625  2.9359550
##
## Log-likelihood: -294.19
## 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
## 13.4638020 -0.5159865  2.6047271
##
## Log-likelihood: -147.2
## Best method: L-BFGS-B
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1      alpha     myshape
## 16.74256244  0.08299642  3.43559695
##
## Log-likelihood: -145.83
## 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
## 13.122690 15.639937 -0.189355  2.622846
##
## Log-likelihood: -146.7
## 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
## 16.21659746 17.14167799  0.06564635  3.42296703
##
## Log-likelihood: -145.72
## 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)
```
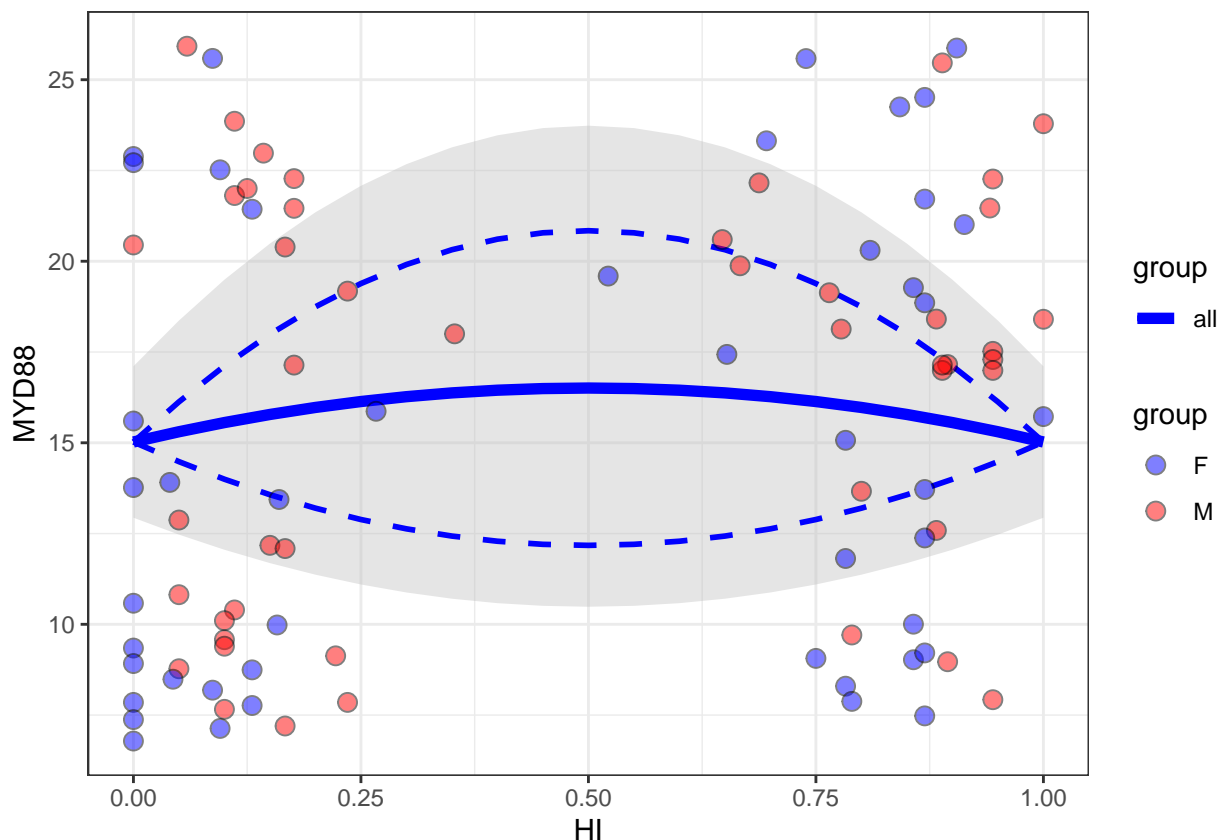
```r
speparam <- parasiteLoad::getParamBounds("weibull", data = field, response = "NCR1")

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 0.37   1 0.3911155
## [1] "Testing H1 no alpha vs alpha"
##     dLL dDF   pvalue
## 1 0.33   1 0.413491
## [1] "Testing H2 groupA no alpha vs alpha"
##     dLL dDF   pvalue
## 1 0.19   1 0.539366
## [1] "Testing H2 groupB no alpha vs alpha"
##     dLL dDF   pvalue
## 1 0.17   1 0.554883
## [1] "Testing H3 groupA no alpha vs alpha"
##     dLL dDF    pvalue
## 1 0.23   1 0.4940659
## [1] "Testing H3 groupB no alpha vs alpha"
##     dLL dDF    pvalue
## 1 0.18   1 0.5491787
## [1] "Testing H1 vs H0"
##   dLL dDF    pvalue
## 1   0   1 0.9935248
## [1] "Testing H2 vs H0"
##     dLL dDF    pvalue
## 1 0.01   3 0.9997142
```

```
## [1] "Testing H3 vs H1"
##   dLL dDF    pvalue
## 1 0.1   4 0.9956145
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.09   2 0.9125738
```

```
##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
## 23.4424587   0.1358494   5.0000000
##
## Log-likelihood: -247.89
## 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
## 23.4394250  23.4503534   0.1362557   5.0000000
##
## Log-likelihood: -247.89
## 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
## 23.4273344   0.1256819   5.0000000
##
## Log-likelihood: -122.77
## 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
## 23.4800814   0.1491375   5.0000000
##
## Log-likelihood: -125.11
## 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
## 23.317828  24.014454   0.169168   5.000000
##
## Log-likelihood: -122.72
## 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
## 23.7374122 23.2080893   0.1505759    5.0000000
##
## Log-likelihood: -125.07
## 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.
```
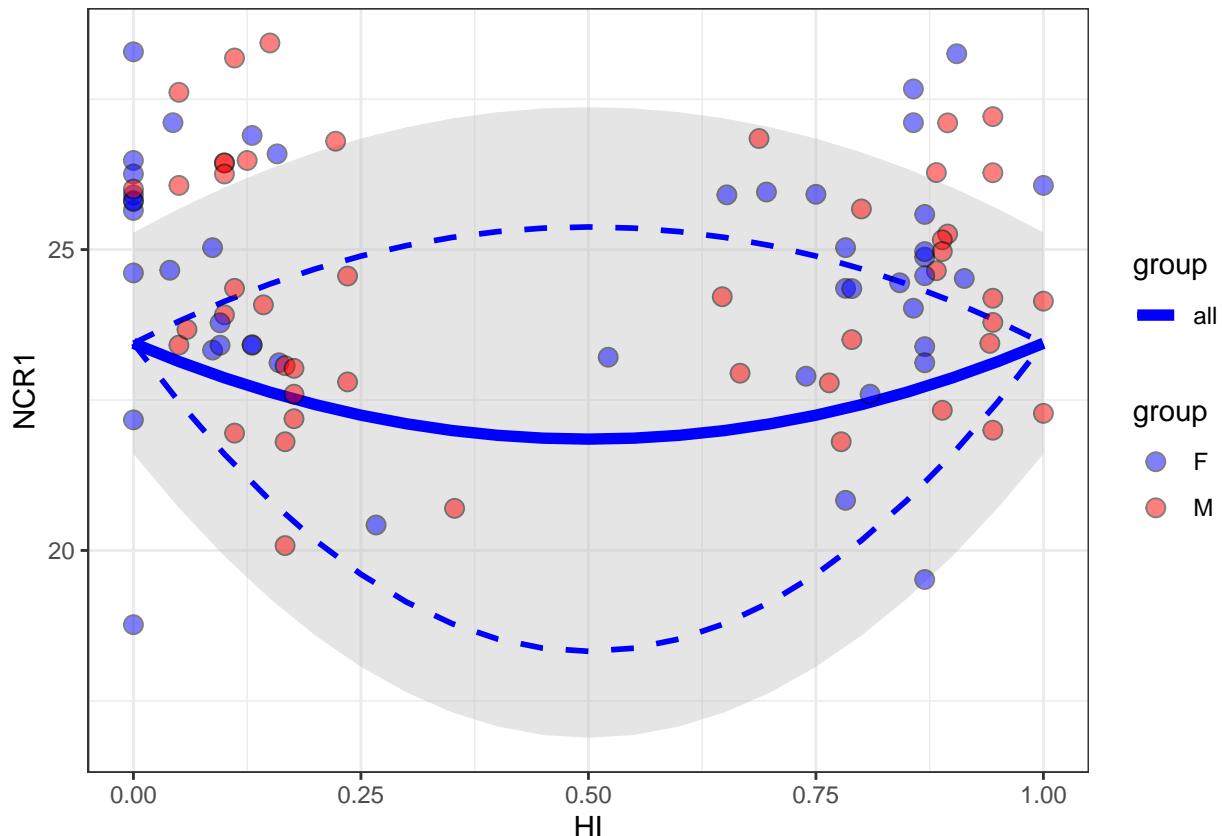
```r
field$Sex <- as.factor(field$Sex)

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 0.2   1 0.5302536
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.16   1 0.5756258
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.49   1 0.3220653
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.9076507
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.35   1 0.3999125
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.9022935
## [1] "Testing H1 vs H0"
##    dLL dDF   pvalue
## 1 0.01   1 0.879235
## [1] "Testing H2 vs H0"
```

```
##     dLL dDF  pvalue
## 1 0.36    3 0.86634
## [1] "Testing H3 vs H1"
##     dLL dDF    pvalue
## 1 0.36    4 0.9492029
## [1] "Testing H3 vs H2"
##     dLL dDF    pvalue
## 1 0.01    2 0.9944679
```

```
##All
print(PRF1)
```

```
## $H0
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1       alpha     myshape
## 23.39478957  0.09939632  5.00000000
##
## Log-likelihood: -249.42
## 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
## 23.45445067 23.25258957  0.09259684  5.00000000
##
## Log-likelihood: -249.41
## 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
## 23.6438599  0.2005803  5.0000000
##
## Log-likelihood: -122.74
## 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
## 22.92560282 -0.03064608  5.00000000
##
## Log-likelihood: -126.31
## 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
## 23.6178200 23.8036925  0.2123901  5.0000000
##
## Log-likelihood: -122.74
## 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
## 22.85870842 22.97670970 -0.03271014  5.00000000
##
## Log-likelihood: -126.31
## 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()
```
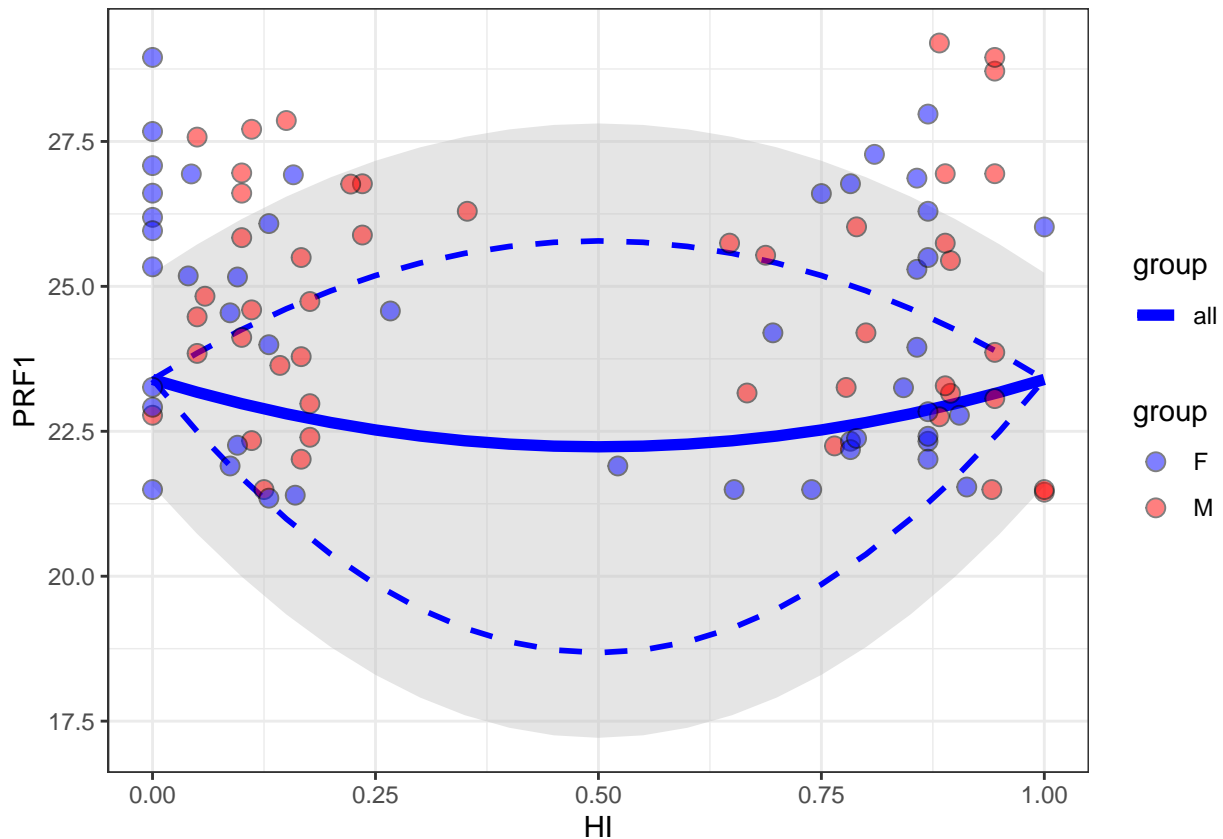
```
## Scale for 'fill' is already present. Adding another scale for 'fill', which
## will replace the existing scale.
```

```
## 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"

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

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

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

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

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

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

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

## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.8752896
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.11   1 0.6421224
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.54   1 0.3004127
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.53   1 0.3018806
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF     pvalue
## 1 2.17   1 0.03744553
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.54   1 0.3003223
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.93   1 0.1730573
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 1.63   3 0.3527014
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 3.88   4 0.1005275
```

```
## [1] "Testing H3 vs H2"
##    dLL dDF     pvalue
## 1 3.18   2 0.04161503
```

```
##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
## 10.41201667  0.02361581  4.76326918
##
## Log-likelihood: -209.56
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1          L2        alpha      myshape
## 10.11788114 10.95884684  0.07124636  4.79168115
##
## Log-likelihood: -208.63
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
```

```
## 
## Coefficients:
##         L1      alpha     myshape
## 10.5736608  0.2114386  4.5164904
## 
## Log-likelihood: -105.36
## Best method: bobyqa
## 
## $H2$groupB
## 
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
## 
## Coefficients:
##        L1      alpha     myshape
##  9.9619422 -0.2573729  5.0000000
## 
## Log-likelihood: -102.57
## Best method: bobyqa
## 
## 
## $H3
## $H3$groupA
## 
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
## 
## Coefficients:
##        L1         L2      alpha     myshape
##  9.901039 12.444111   0.424179   4.781805
## 
## Log-likelihood: -102.19
## 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"]],
```
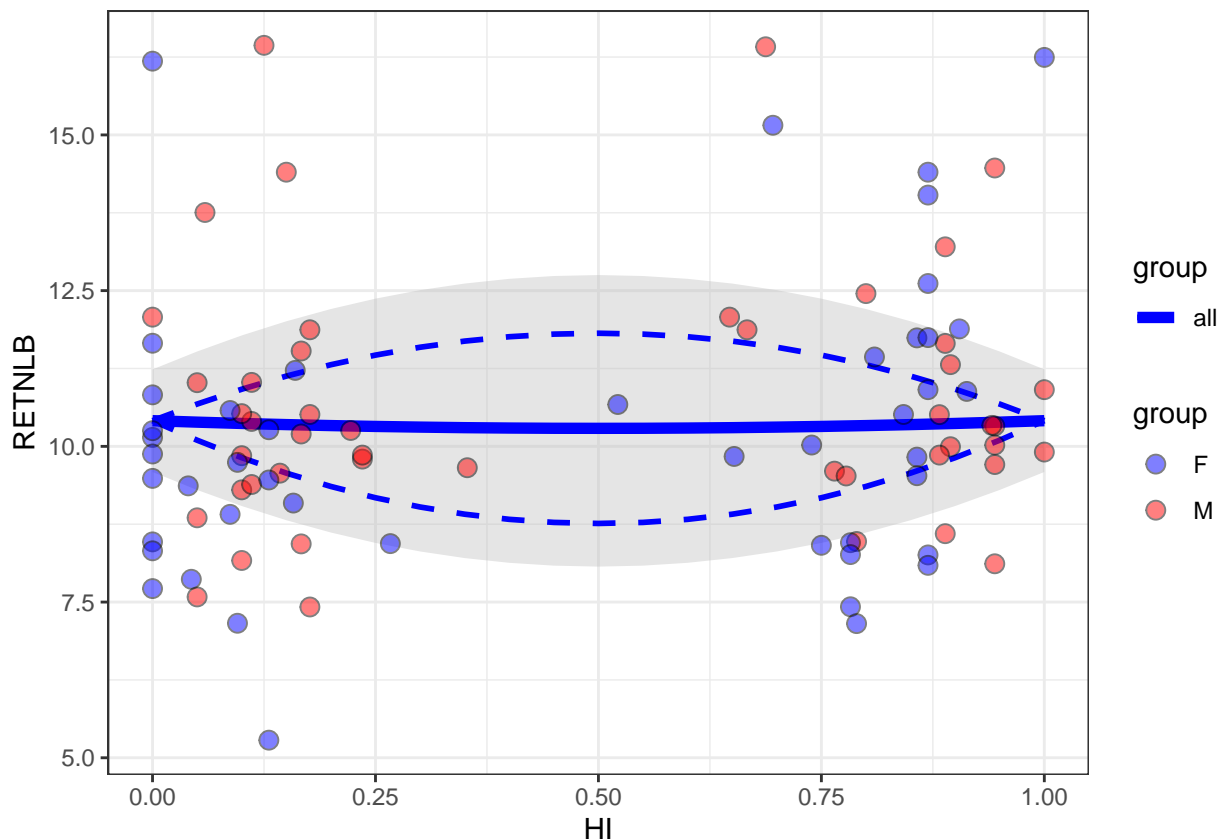
```
##           alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##       upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##           alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##       control = config$control)
##
## Coefficients:
##        L1        L2      alpha    myshape
## 10.002441   9.916132  -0.257629   5.000000
##
## Log-likelihood: -102.56
## Best method: bobyqa
```

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

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

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



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

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

```r
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"
```

```
## Warning in bbmle::mle2(response ~ dweibull(shape = 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"
## [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"
```

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

```
## calculations based on Hessian may be unreliable

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

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

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

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

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

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

## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.8892309
## [1] "Testing H1 no alpha vs alpha"
##   dLL dDF    pvalue
## 1   0   1 0.9524479
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.82   1 0.05654932
## [1] "Testing H2 groupB no alpha vs alpha"
##   dLL dDF     pvalue
## 1 3.3   1 0.01017372
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF     pvalue
## 1 2.01   1 0.04512266
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF      pvalue
## 1 3.33   1 0.009899352
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.06   1 0.7364755
## [1] "Testing H2 vs H0"
##    dLL dDF     pvalue
## 1 5.12   3 0.01663725
## [1] "Testing H3 vs H1"
##    dLL dDF     pvalue
## 1 5.49   4 0.02675789
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.43   2 0.6515867
```

```
##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
## 11.07829826 -0.01771689  5.00000000
##
## Log-likelihood: -206.19
## 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
## 11.002229171 11.211749068 -0.007794367  5.000000000
##
## Log-likelihood: -206.14
## 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
## 11.7621171  0.3092282  5.0000000
##
## Log-likelihood: -101.56
```

```
## 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
##  9.6390383 -0.6366352  5.0000000
##
## Log-likelihood: -99.52
## 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
## 11.5729872 12.1686563   0.3441928  5.0000000
##
## Log-likelihood: -101.37
## Best method: bobyqa
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
```

```
##            L1         L2       alpha     myshape
##    9.3511763  9.8722665  -0.6539532   5.0000000
##
## Log-likelihood: -99.28
## 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.
```
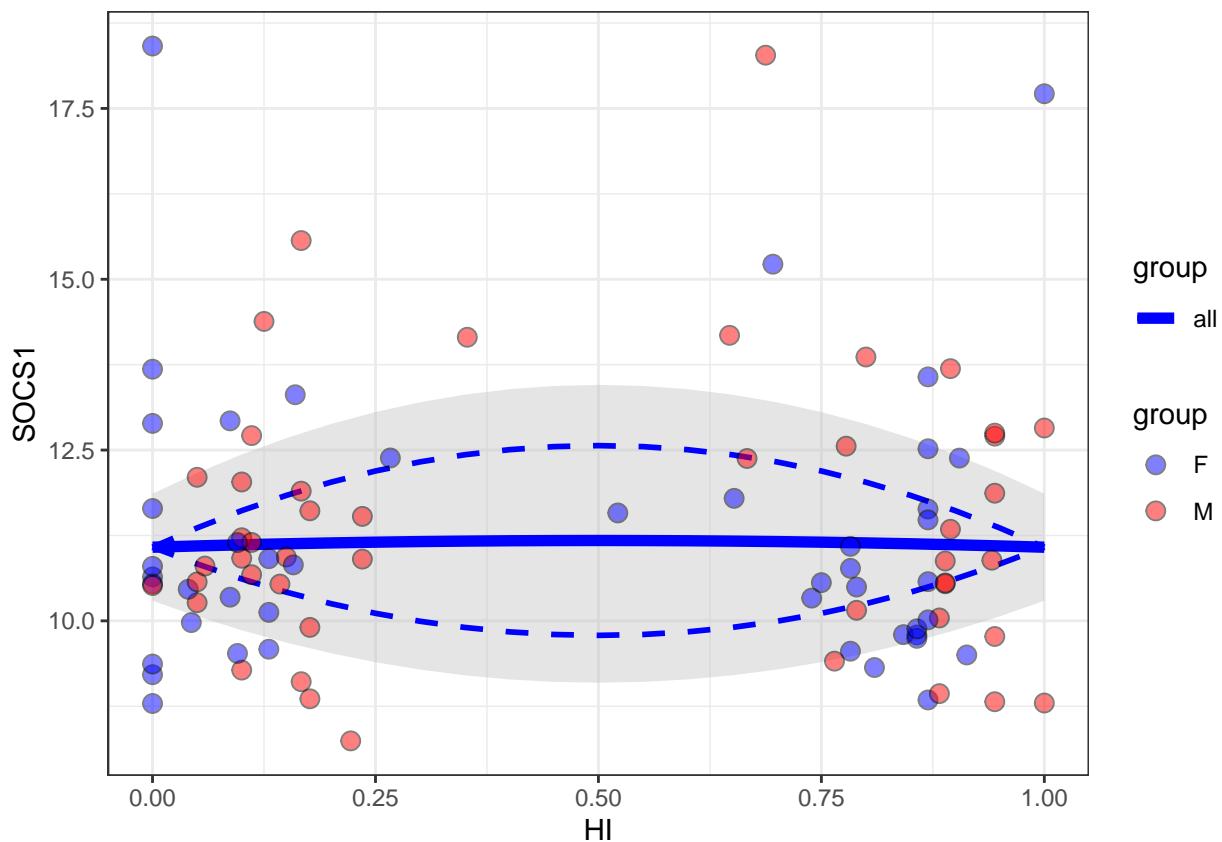


```
field$Sex <- as.factor(field$Sex)

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   0   1 0.9365761
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.01   1 0.877207
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.06   1 0.731288
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.03   1 0.793152
## [1] "Testing H3 groupA no alpha vs alpha"
##   dLL dDF   pvalue
## 1   0   1 0.9210294
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.05   1 0.7575529
## [1] "Testing H1 vs H0"
##    dLL dDF   pvalue
## 1 0.04   1 0.7695544
## [1] "Testing H2 vs H0"
##    dLL dDF   pvalue
## 1 0.09   3 0.9805297
## [1] "Testing H3 vs H1"
##    dLL dDF pvalue
## 1 0.47   4 0.9196
## [1] "Testing H3 vs H2"
##    dLL dDF   pvalue
## 1 0.42   2 0.6575008
```

```
##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
## 20.19908080 -0.01414287   5.00000000
##
## Log-likelihood: -244.81
## 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
## 20.29335826 19.95900717 -0.02856649   5.00000000
##
## Log-likelihood: -244.76
## 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
```

```
## 19.91190095 -0.08320177  5.00000000
##
## Log-likelihood: -121.83
## 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
## 20.57691756  0.06756058  5.00000000
##
## Log-likelihood: -122.89
## 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
## 19.84332139 20.45275282 -0.02866069  5.00000000
##
## Log-likelihood: -121.78
## 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
## 21.25921230 19.87679697  0.07780539  5.00000000
##
## Log-likelihood: -122.52
## Best method: bobyqa
```

```r
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.
```
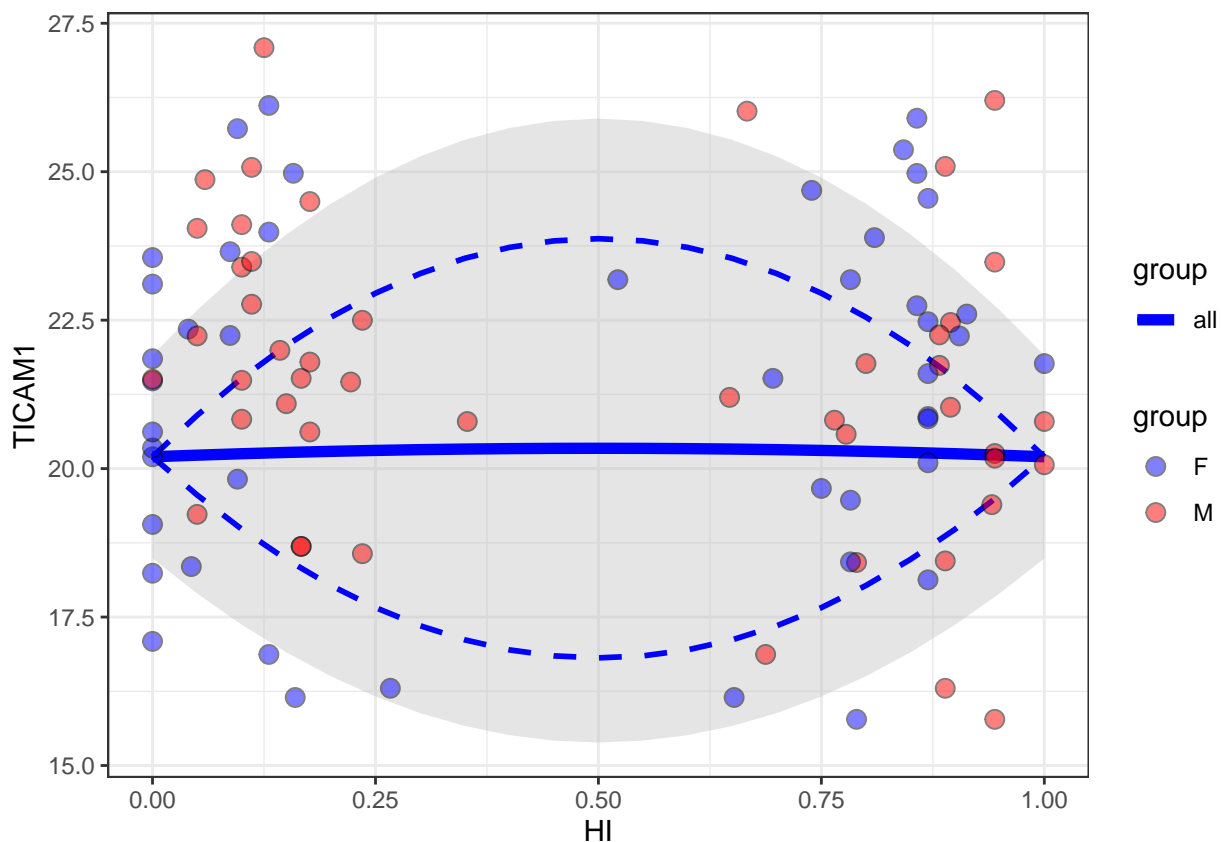


```r
field$Sex <- as.factor(field$Sex)

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   0   1 0.9956017
## [1] "Testing H1 no alpha vs alpha"
##   dLL dDF   pvalue
## 1   0   1 0.988477
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.05   1 0.7608506
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.02   1 0.8318057
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.13   1 0.6087703
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.02   1 0.8475079
## [1] "Testing H1 vs H0"
##   dLL dDF    pvalue
## 1   0   1 0.9670622
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.38   3 0.8590135
## [1] "Testing H3 vs H1"
##    dLL dDF  pvalue
## 1 0.52   4 0.90483
## [1] "Testing H3 vs H2"
```

```
##    dLL dDF   pvalue
## 1 0.14    2 0.871876
```

```
##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
## 2.020418e+01 8.692459e-04 5.000000e+00
##
## Log-likelihood: -236.79
## 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
## 20.188532880 20.235276078   0.002341657   5.000000000
##
## Log-likelihood: -236.79
## 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
## 20.09788132   0.06143294   5.00000000
##
## Log-likelihood: -116.03
## 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
## 20.27506469  -0.05368809   5.00000000
##
## Log-likelihood: -120.38
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##        L1        L2       alpha     myshape
## 19.923029  20.807409   0.118918   5.000000
##
## Log-likelihood: -115.91
## 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
## 20.4409428  20.1384496  -0.0485576   5.0000000
##
## Log-likelihood: -120.37
## Best method: bobyqa
```

```r
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.
```