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")
```

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/RtmpKFzPnY/remotes33ddc65db62f33/alicebalard-parasiteLoad-1b43216/DESCRIPTI
## * preparing 'parasiteLoad':
## * checking DESCRIPTION meta-information ... OK
## * checking for LF line-endings in source and make files and shell scripts
## * checking for empty or unneeded directories
## * building 'parasiteLoad_0.1.0.tar.gz'
```

Applying Alice's package on every gene

```
x <- field$IDO1



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



findGoodDist <- function(x, distribs, distribs2){
  l =lapply(distribs, function(i) tryDistrib(x, i))
  names(l) <- distribs
  print(l)
  listDistr <- lapply(distribs2, function(i){
    if (i %in% "t"){
      fitdistrplus::fitdist(x, i, start = list(df =2))
    } else {
      fitdistrplus::fitdist(x,i)
    }}
  )
  par(mfrow=c(2,2))
  denscomp(listDistr, legendtext=distribs2)
  cdfcomp(listDistr, legendtext=distribs2)
  qqcomp(listDistr, legendtext=distribs2)
  ppcomp(listDistr, legendtext=distribs2)
  par(mfrow=c(1,1))
}
```

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

Functions for testing distributions

```
## $fit
##       mean          sd
##   15.0575257    4.4097438
##  ( 0.2409300) ( 0.1703633)
##
## $loglik
## [1] -972.423
##
## $AIC
## NULL
```

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

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

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

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

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

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

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

```
## $fit
##      shape        scale
##    3.5549795   16.6979975
##  ( 0.1421786) ( 0.2724857)
##
## $loglik
## [1] -976.5934
##
## $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"
```

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

```
## [1] "Did converge"
## [1] "Fitting for groupB : M"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.67   1 0.2464973
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.46   1 0.3364279
## [1] "Testing H2 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1   0   1 0.9640624
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.64   1 0.06993746
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.9198046
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 1.25   1 0.1142013
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.14   1 0.5927666
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 1.51   3 0.3873923
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 2.21   4 0.3529437
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.84   2 0.4334773
```

```
##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
## 20.66436487  0.08694602  5.00000000
##
## Log-likelihood: -985.61
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 20.80840234 20.45982142  0.07573465  5.00000000
##
## Log-likelihood: -985.47
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##           L1        alpha      myshape
## 20.397246611 -0.004673532  5.000000000
##
## Log-likelihood: -483.7
## Best method: bobyqa
```

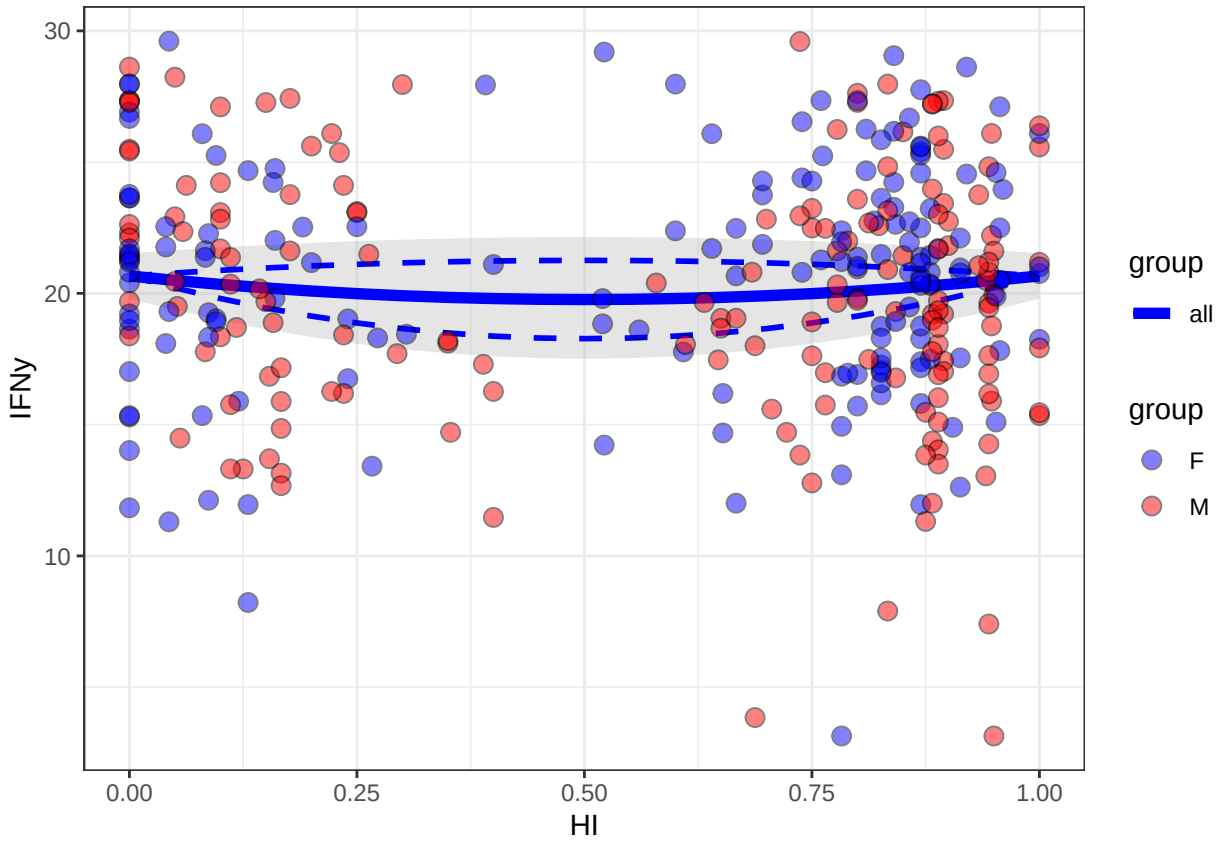
```
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 20.9652515  0.2076485  4.8276501
##
## Log-likelihood: -500.4
## 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
## 20.26647283 20.62522076  0.01117735  5.00000000
##
## Log-likelihood: -483.63
## Best method: bobyqa
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
```

```
## 21.5237023 20.3616480  0.1863146  4.8446615
##
## Log-likelihood: -499.64
## Best method: bobyqa
```

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

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

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




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

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

IFNy <- 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 0.23   1 0.4941464
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.26   1 0.4693894
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.13   1 0.6142887
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.11   1 0.6444226
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.24   1 0.492078
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.08   1 0.687482
## [1] "Testing H1 vs H0"
##    dLL dDF   pvalue
## 1 0.03   1 0.802799
## [1] "Testing H2 vs H0"
##   dLL dDF    pvalue
## 1   0   3 0.9998966
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.25   4 0.9732499
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.28   2 0.7559332
```

```
##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
## 21.13757351  0.05184824  5.00000000
##
## Log-likelihood: -887.03
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 21.06456046 21.22999921  0.05633644  5.00000000
##
## Log-likelihood: -887
## 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
```

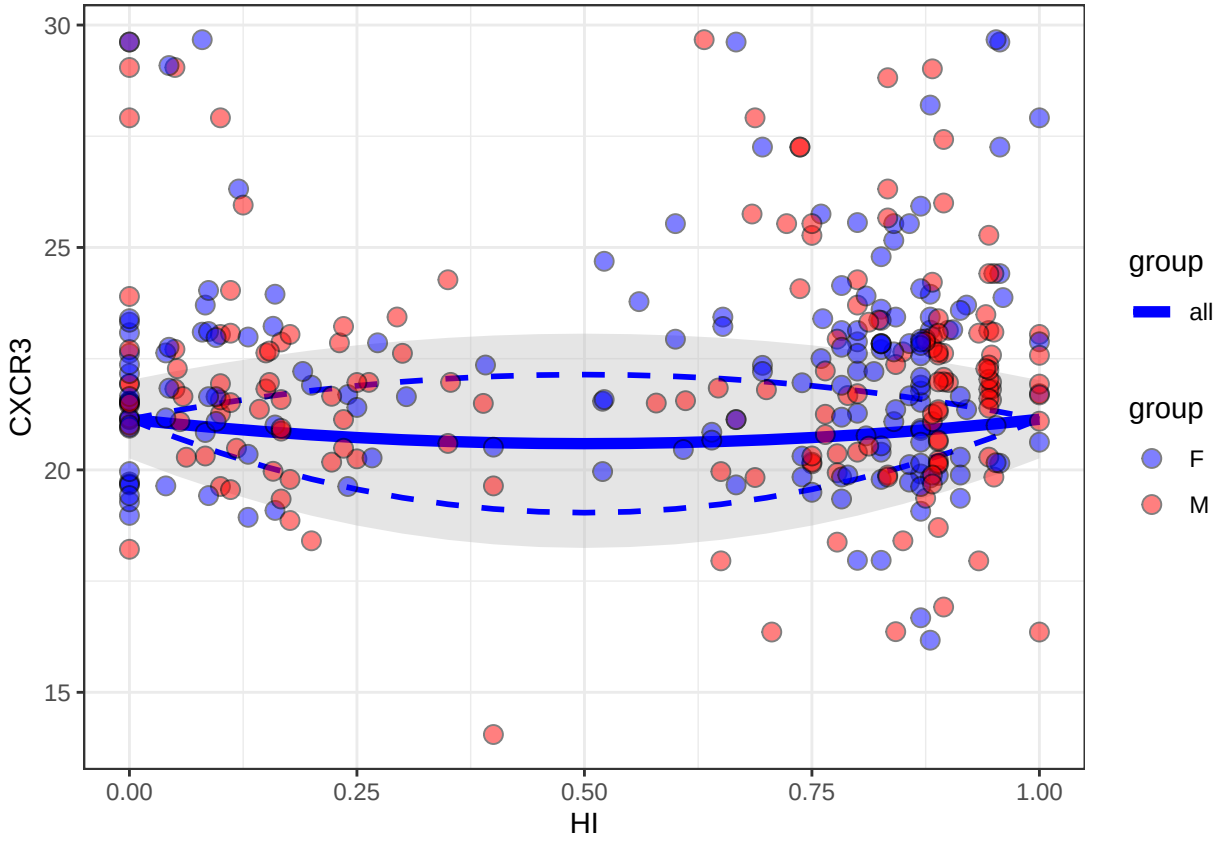
```
## 21.16012651  0.05309422  5.00000000
##
## Log-likelihood: -441.11
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$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
## 21.11417149  0.05042914  5.00000000
##
## Log-likelihood: -445.92
## 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
## 20.89486805 21.54677628  0.07518605  5.00000000
##
## Log-likelihood: -440.88
## Best method: L-BFGS-B
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
```

```
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 21.25139030 20.96471235  0.04478293  5.00000000
##
## Log-likelihood: -445.87
## Best method: L-BFGS-B
```

```
bananaPlot(mod = IFNy$H0,
             data = field,
             response = "CXCR3",
             group = "Sex") +
  scale_fill_manual(values = c("blue", "red")) +
  scale_color_manual(values = c("blue", "red")) +
  theme_bw()
```

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

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




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

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

IFNy <- 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   1 0.976873
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.9058748
## [1] "Testing H2 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1   0   1 0.9545683
## [1] "Testing H2 groupB no alpha vs alpha"
##   dLL dDF    pvalue
## 1   0   1 0.9881301
## [1] "Testing H3 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1   0   1 0.9770896
## [1] "Testing H3 groupB no alpha vs alpha"
##   dLL dDF    pvalue
## 1   0   1 0.9771681
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.12   1 0.6259119
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.18   3 0.9485655
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.37   4 0.9459875
## [1] "Testing H3 vs H2"
```

```
##    dLL dDF    pvalue
## 1 0.31   2 0.7330492
```

```
##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
## 22.358033634 -0.002280237  5.000000000
##
## Log-likelihood: -904.04
## 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
## 22.512843288 22.176246390 -0.009510556  5.000000000
##
## Log-likelihood: -903.92
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
```

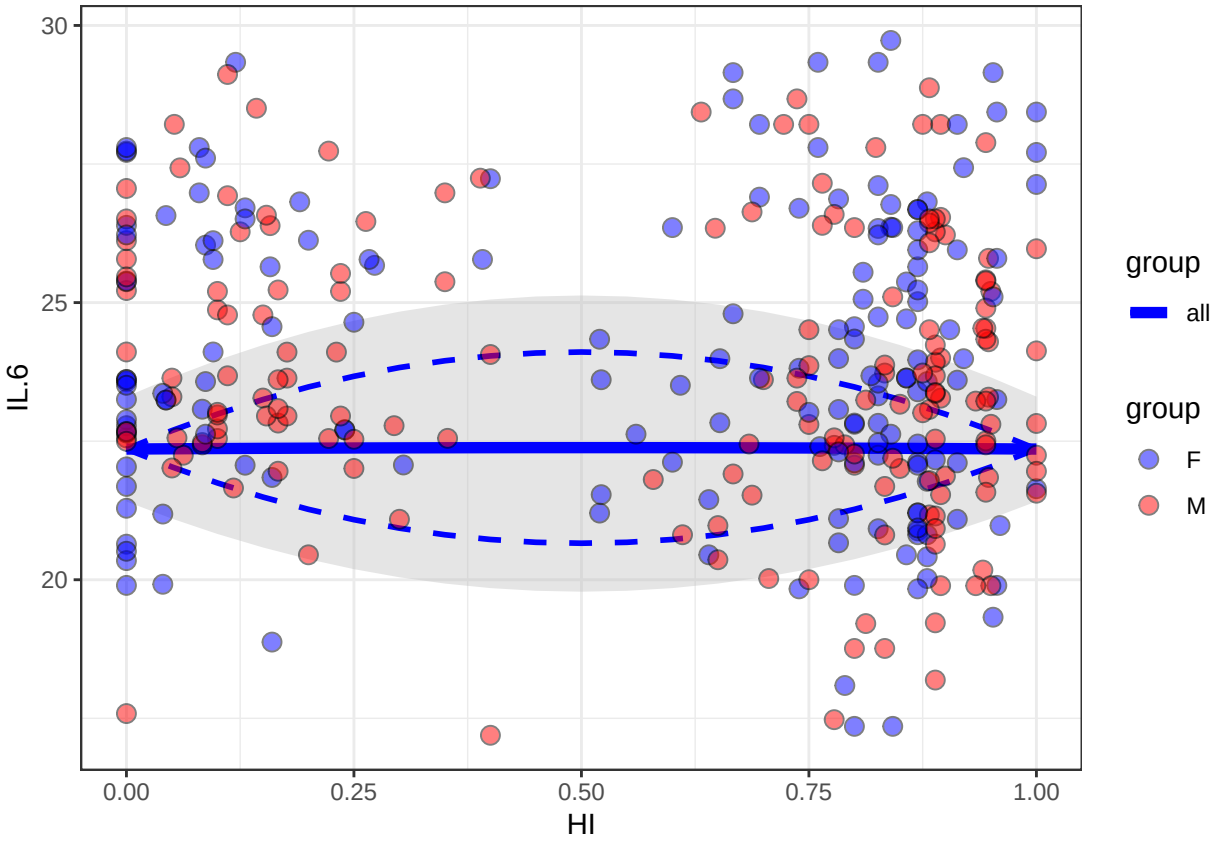
```
## Coefficients:
##           L1        alpha      myshape
## 22.482401564 -0.006154352  5.000000000
##
## Log-likelihood: -452.81
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (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
## 22.231268733  0.001715418  5.000000000
##
## Log-likelihood: -451.05
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##           L1           L2        alpha      myshape
## 22.448010801 22.534765599 -0.003246811  5.000000000
##
## Log-likelihood: -452.81
## 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
## 22.634336663 21.883191191 -0.003330059  5.000000000
##
## Log-likelihood: -450.74
## Best method: bobyqa
```

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

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

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




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

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

```
IFNy <- 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 0.28   1 0.4523162
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.15   1 0.580233
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.17   1 0.5646307
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.12   1 0.625714
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.06   1 0.7246361
## [1] "Testing H3 groupB no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.1   1 0.6590938
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.36   1 0.3961769
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.04   3 0.9934943
```

```
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.05   4 0.9987232
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.37   2 0.6917753
```

```
##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
## 23.57396321  0.05860765  5.00000000
##
## Log-likelihood: -915.52
## 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
## 23.8477593 23.2281731  0.0444881  5.0000000
##
## Log-likelihood: -915.16
## 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"]],
```

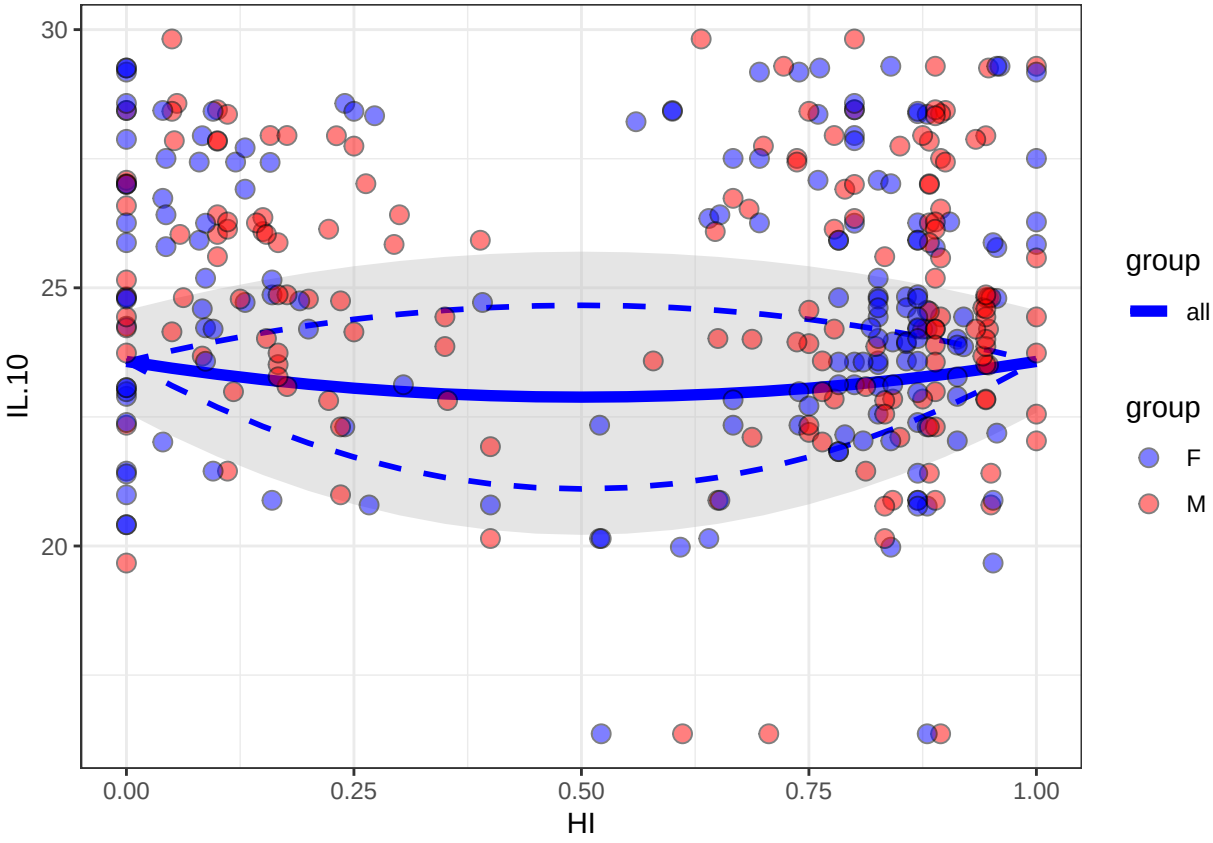
```
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1       alpha     myshape
## 23.50707937  0.06024782  5.00000000
##
## Log-likelihood: -457.14
## 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
## 23.63920972  0.05694935  5.00000000
##
## Log-likelihood: -458.34
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 23.71854225 23.14229445  0.03961731  5.00000000
##
## Log-likelihood: -457
## 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
## 24.00122226 23.32014918  0.05194454  5.00000000
##
## Log-likelihood: -458.11
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$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.
```



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

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

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

```
## [1] "Analysing data for response: IL.13"
## [1] "Fit for the response: IL.13"
## [1] "Fitting for all"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Fitting for groupA : F"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Fitting for groupB : M"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.21   1 0.5200098
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.07   1 0.7073971
## [1] "Testing H2 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.2   1 0.5291421
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.02   1 0.8375112
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.17   1 0.5569559
## [1] "Testing H3 groupB no alpha vs alpha"
```

```
##   dLL dDF    pvalue
## 1   0   1 0.9623027
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 1.19   1 0.1231585
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 1.61   3 0.3579706
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 2.72   4 0.2448702
## [1] "Testing H3 vs H2"
##   dLL dDF    pvalue
## 1 2.3   2 0.1006604
```

```
##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
## 16.55284103  0.07616907  3.09803071
##
## Log-likelihood: -1055.58
## 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
## 17.1494569 15.8898297  0.0456834  3.1023105
##
## Log-likelihood: -1054.39
## Best method: bobyqa
##
```

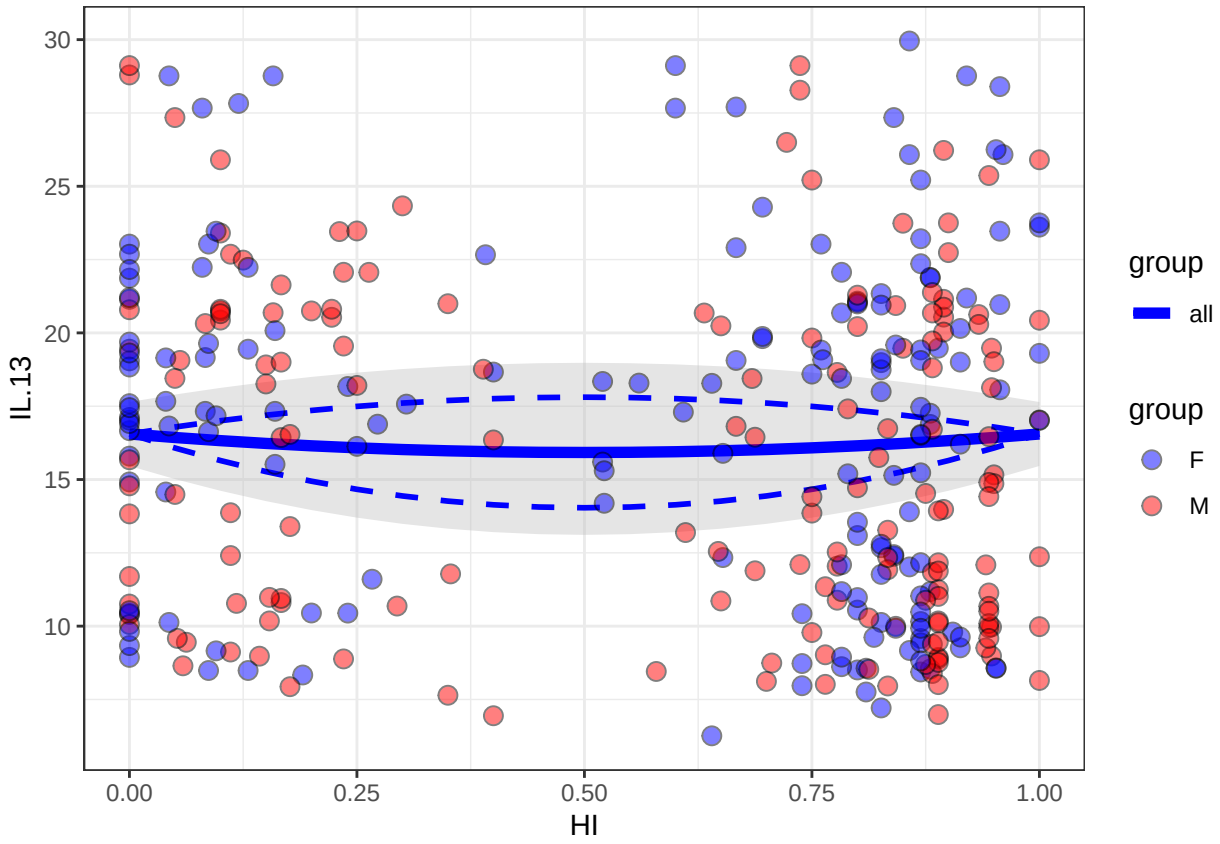
```
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1       alpha     myshape
## 17.20862483  0.09777046  3.20485373
##
## Log-likelihood: -527.65
## 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
## 15.83805312  0.03732138  3.02065015
##
## Log-likelihood: -526.32
## 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
## 17.25774384 17.14480046  0.09423363  3.20425311
```

```
##
## Log-likelihood: -527.64
## 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
## 17.110706241 14.713148410  0.008718456  3.056444824
##
## Log-likelihood: -524.03
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$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")

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

```
## [1] "Analysing data for response: IL1RN"
## [1] "Fit for the response: IL1RN"
## [1] "Fitting for all"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Fitting for groupA : F"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
```

```
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Fitting for groupB : M"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##    dLL dDF   pvalue
## 1 1.54   1 0.079619
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF     pvalue
## 1 1.39   1 0.09573598
## [1] "Testing H2 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.9   1 0.1785661
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.65   1 0.2535602
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.58   1 0.2822587
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.75   1 0.221989
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.02   1 0.8594302
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 1.79   3 0.3114628
## [1] "Testing H3 vs H1"
##    dLL dDF   pvalue
## 1 2.21   4 0.352885
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.44   2 0.6464306
```

```
##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
## 13.6859746 -0.1914273  3.5477182
##
## Log-likelihood: -961.86
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 13.6383627 13.7456601 -0.1865142  3.5469568
##
## Log-likelihood: -961.84
## 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
## 13.9590093 -0.2154608  3.5120743
##
## Log-likelihood: -485.85
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
```

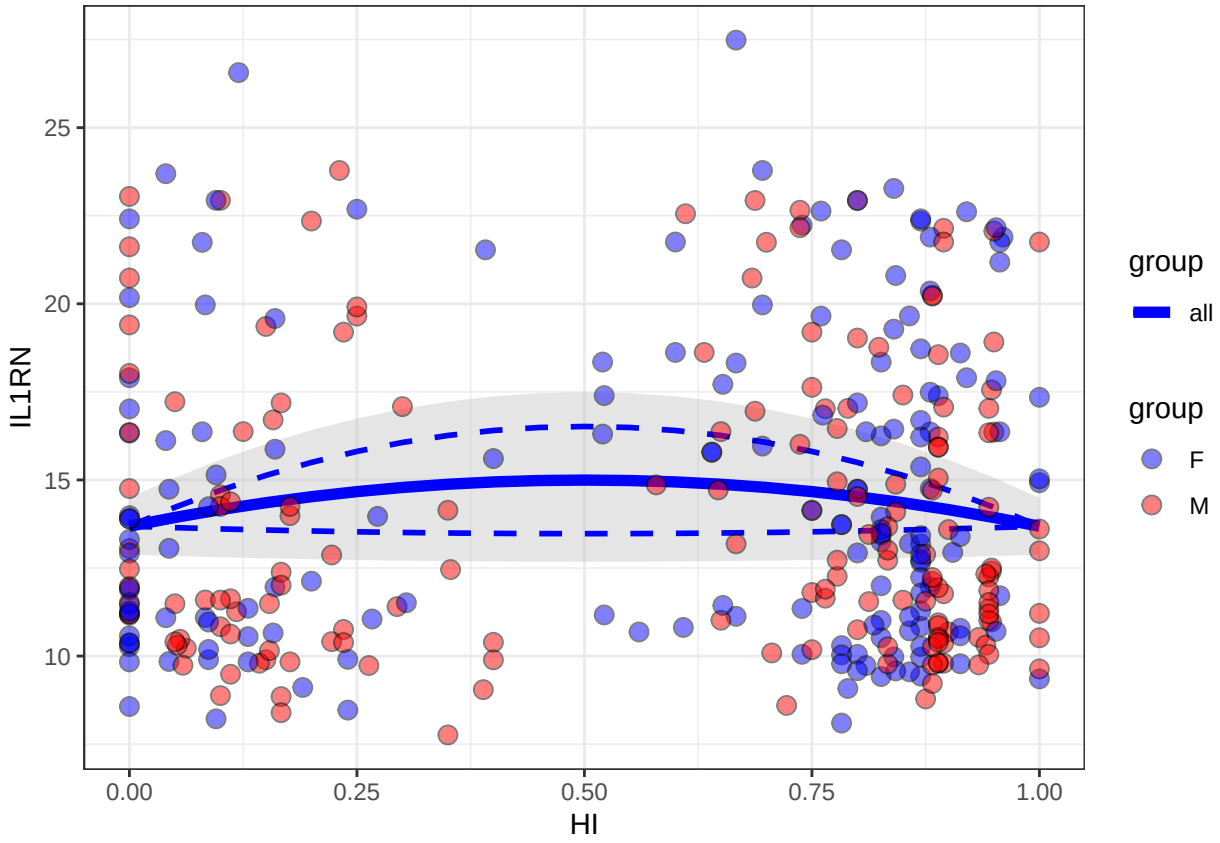
```
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 13.4130727 -0.1676594  3.6279845
##
## Log-likelihood: -474.22
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## 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
## 13.6959865 14.3516835 -0.1775076  3.5107781
##
## Log-likelihood: -485.59
## Best method: bobyqa
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 13.6461956 13.1720271 -0.1827691  3.6358216
##
## Log-likelihood: -474.05
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$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")

IFNy <- 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 0.23   1 0.4941464
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.26   1 0.4693894
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.13   1 0.6142887
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.11   1 0.6444226
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.24   1 0.492078
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.08   1 0.687482
## [1] "Testing H1 vs H0"
##    dLL dDF   pvalue
## 1 0.03   1 0.802799
## [1] "Testing H2 vs H0"
##   dLL dDF    pvalue
## 1   0   3 0.9998966
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.25   4 0.9732499
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.28   2 0.7559332
```

```
##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,
```

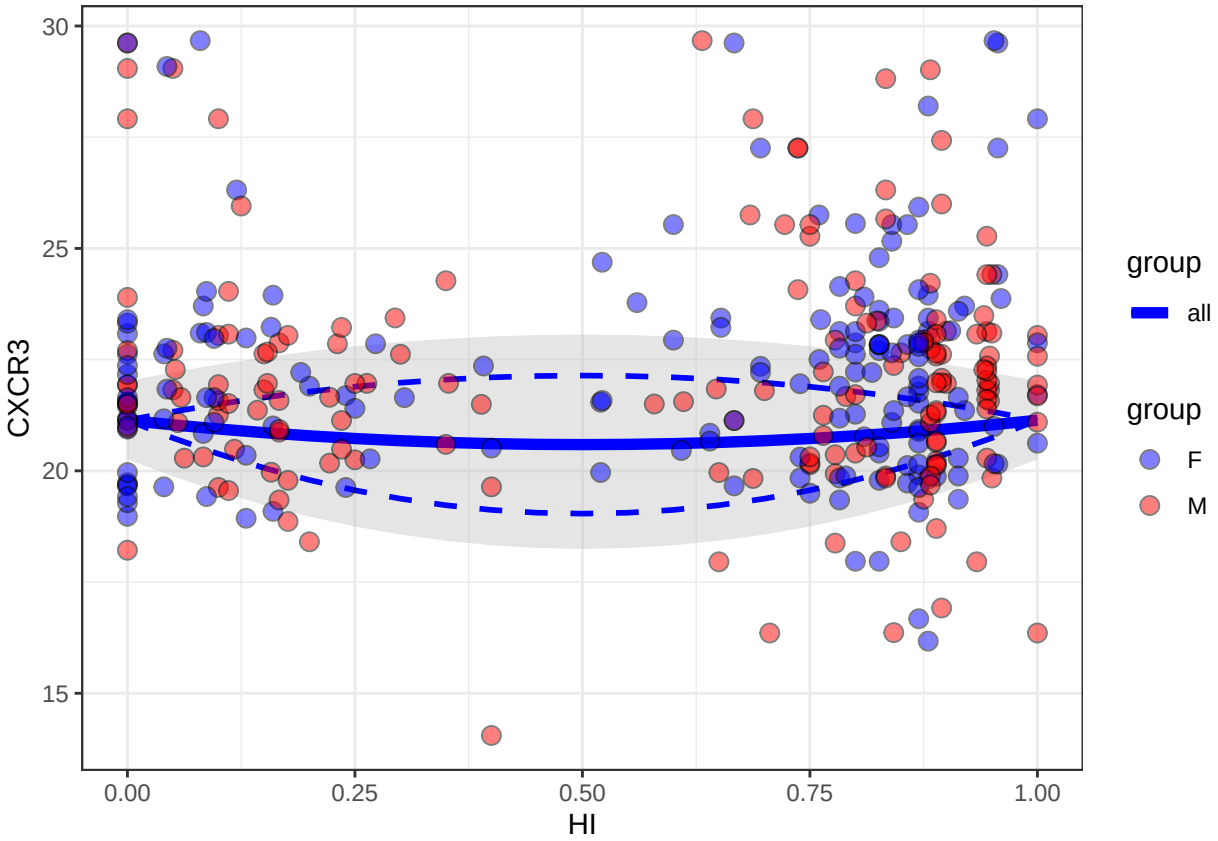
```
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1       alpha     myshape
## 21.13757351  0.05184824  5.00000000
##
## Log-likelihood: -887.03
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 21.06456046 21.22999921  0.05633644  5.00000000
##
## Log-likelihood: -887
## 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
## 21.16012651  0.05309422  5.00000000
##
## Log-likelihood: -441.11
## 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
## 21.11417149  0.05042914  5.00000000
##
## Log-likelihood: -445.92
## 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
## 20.89486805 21.54677628  0.07518605  5.00000000
##
## Log-likelihood: -440.88
## Best method: L-BFGS-B
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 21.25139030 20.96471235  0.04478293  5.00000000
##
## Log-likelihood: -445.87
## Best method: L-BFGS-B
```

```
bananaPlot(mod = IFNy$HO,
             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")

IFNy <- 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 0.09   1 0.6713713
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.13   1 0.6058085
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.02   1 0.8234527
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.07   1 0.7057156
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.11   1 0.6453932
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.07   1 0.710826
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.13   1 0.6067087
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.02   3 0.9982353
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.21   4 0.9803564
## [1] "Testing H3 vs H2"
##    dLL dDF   pvalue
## 1 0.33   2 0.720831
```

```
##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
## 21.50099623  0.03297412  5.00000000
##
## Log-likelihood: -885.58
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 21.34079401 21.68489302  0.04068176  5.00000000
##
## Log-likelihood: -885.45
## 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
## 21.49396241  0.02443874  5.00000000
##
## Log-likelihood: -442.38
## Best method: bobyqa
##
## $H2$groupB
```

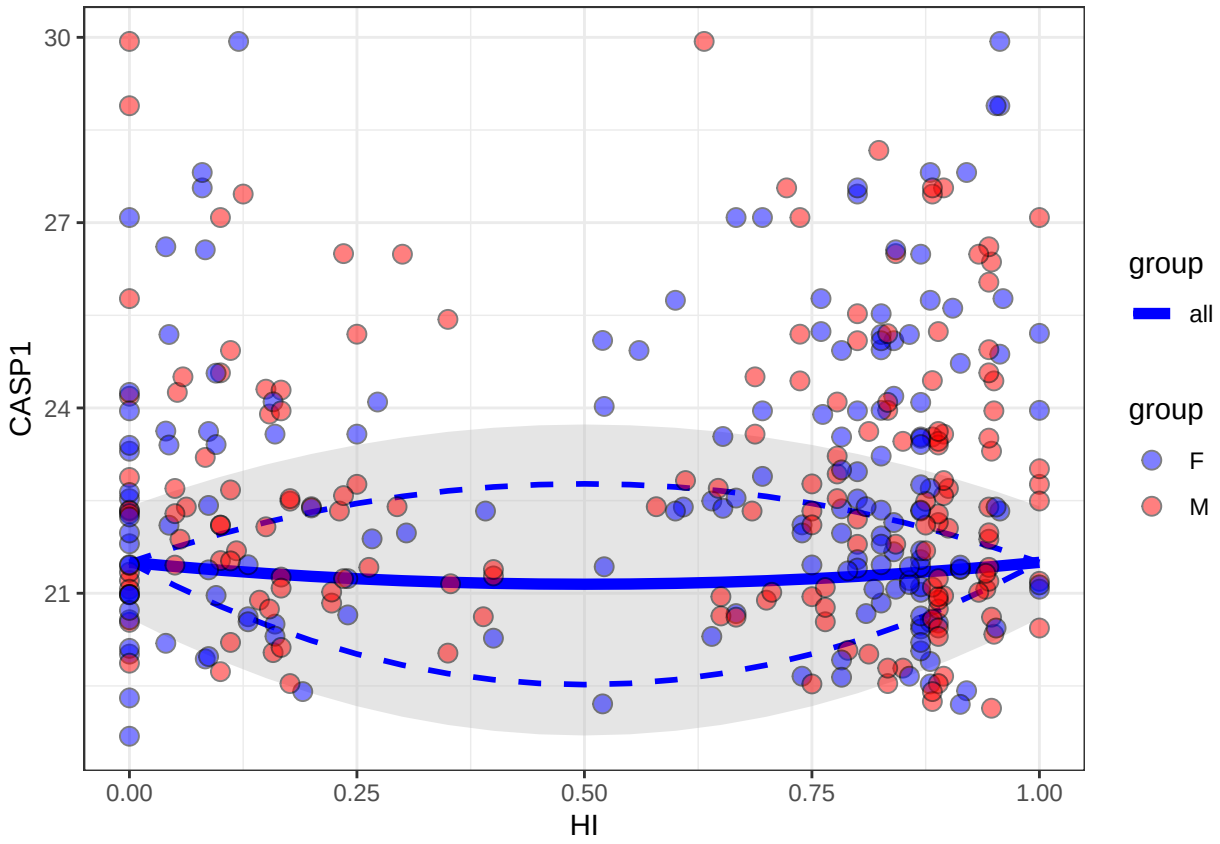
```
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1       alpha     myshape
## 21.50902537  0.04166358  5.00000000
##
## Log-likelihood: -443.18
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 21.1797205 21.9665025  0.0522642  5.0000000
##
## Log-likelihood: -442.06
## 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
## 21.54566954 21.47719556  0.04106758  5.00000000
##
```

```
## Log-likelihood: -443.18
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$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")

IFNy <- 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 0.9447407
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.02   1 0.8563652
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.14   1 0.5997119
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.26   1 0.4744637
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.04   1 0.7645265
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.26   1 0.4733661
## [1] "Testing H1 vs H0"
##    dLL dDF   pvalue
## 1 0.28   1 0.452086
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.72   3 0.6981508
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.86   4 0.7881596
## [1] "Testing H3 vs H2"
##    dLL dDF   pvalue
## 1 0.42   2 0.654451
```

```
##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
## 19.981630484  0.005451623  5.000000000
##
## Log-likelihood: -906.19
## 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
## 19.75363306 20.21762770  0.01433242  5.00000000
##
## Log-likelihood: -905.91
## 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
## 19.86502220 -0.05730404  5.00000000
##
## Log-likelihood: -455
```

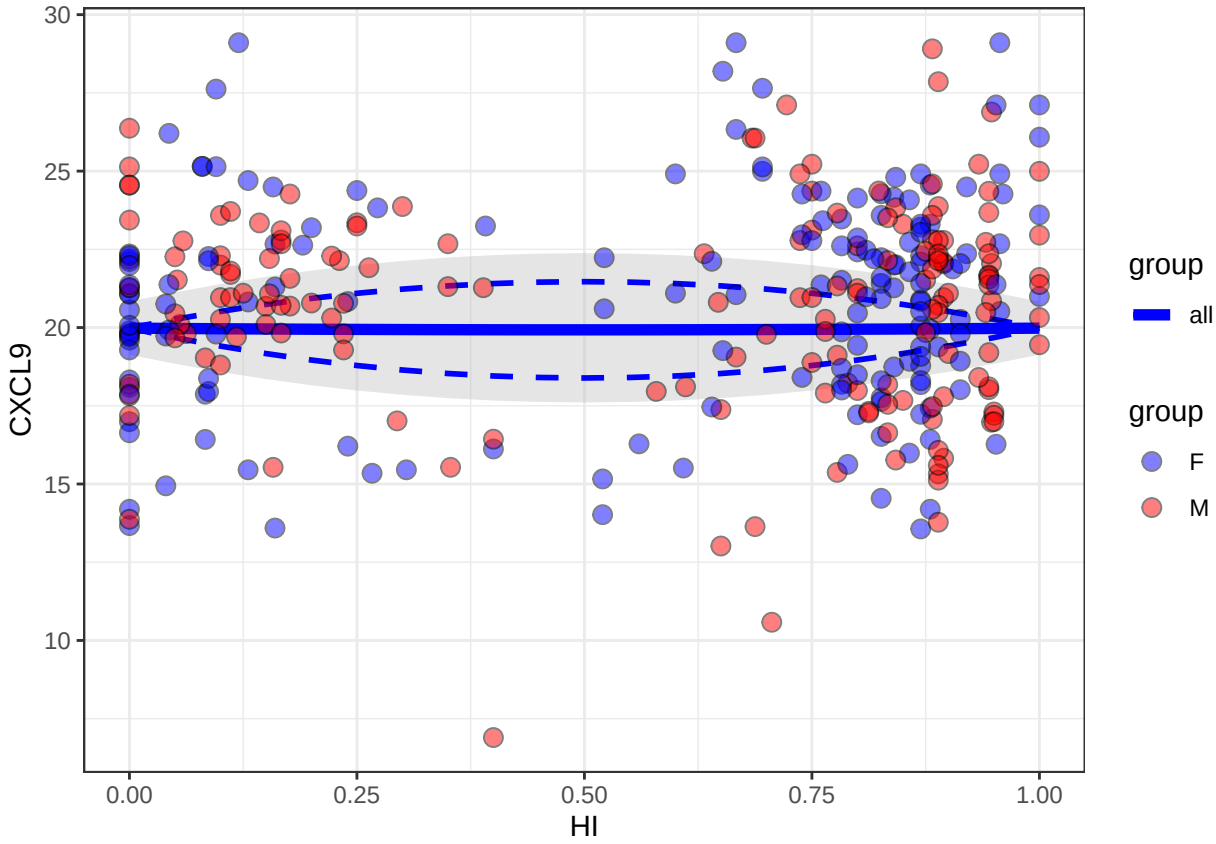
```
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1       alpha     myshape
## 20.15461841  0.08152077  5.00000000
##
## Log-likelihood: -450.47
## 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
## 19.5035914 20.3150787 -0.0333836  5.0000000
##
## Log-likelihood: -454.58
## 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
## 20.12991161 20.17488360  0.08181266  5.00000000
##
## Log-likelihood: -450.47
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$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")

IFNy <- 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 0.09   1 0.6683474
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.09   1 0.668404
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.32   1 0.4266815
## [1] "Testing H2 groupB no alpha vs alpha"
##   dLL dDF   pvalue
## 1   0   1 0.937834
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.22   1 0.5080867
## [1] "Testing H3 groupB no alpha vs alpha"
##   dLL dDF    pvalue
## 1   0   1 0.9951221
## [1] "Testing H1 vs H0"
##   dLL dDF    pvalue
## 1   0   1 0.9561119
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.98   3 0.5798636
## [1] "Testing H3 vs H1"
```

```
##    dLL dDF    pvalue
## 1 1.12   4 0.6926173
## [1] "Testing H3 vs H2"
##    dLL dDF   pvalue
## 1 0.14   2 0.872036
```

```
##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
## 14.8795486 -0.0453282  3.5547258
##
## Log-likelihood: -976.5
## 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
## 14.89546892 14.85915732 -0.04688506  3.55516770
##
## Log-likelihood: -976.5
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
```

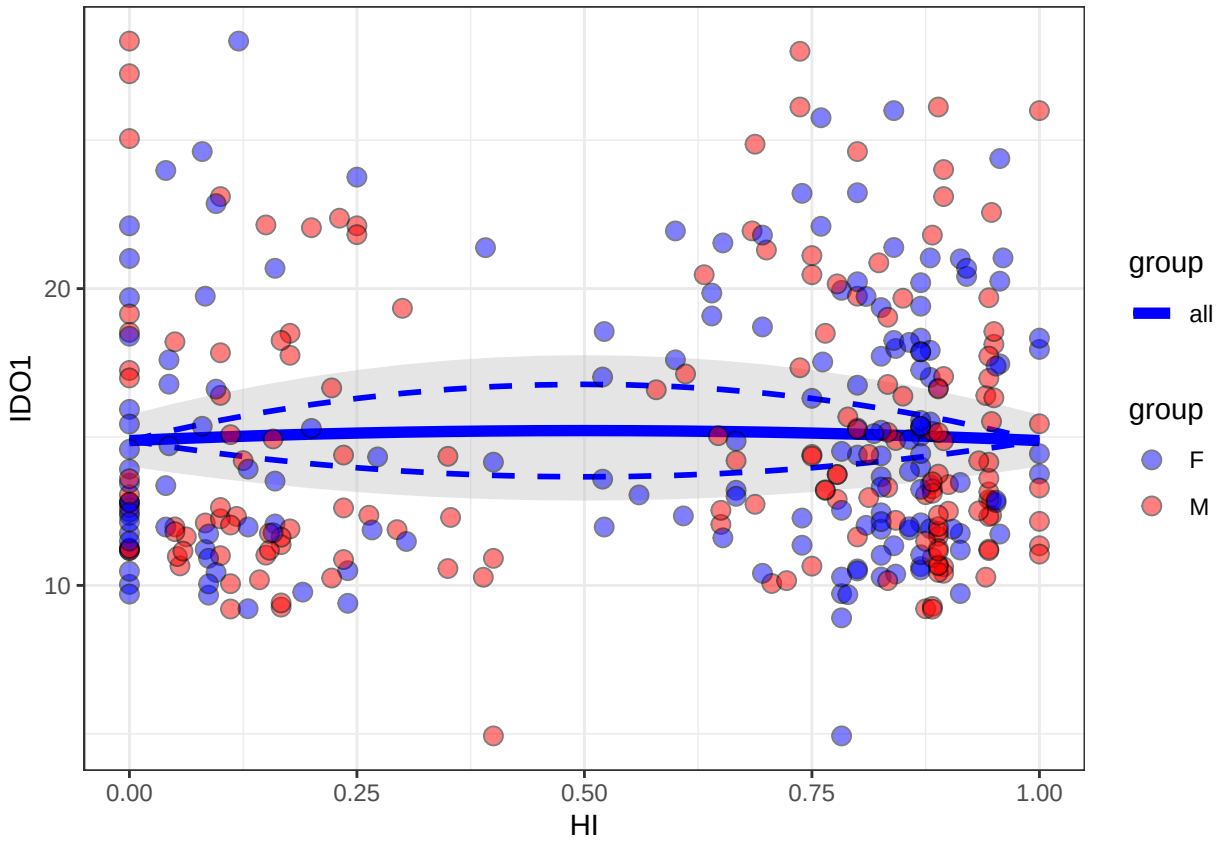
```
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 14.7557708 -0.1199757  3.7550175
##
## Log-likelihood: -481.26
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1       alpha     myshape
## 14.95514457  0.01171584  3.39652130
##
## Log-likelihood: -494.26
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##        L1        L2     alpha   myshape
## 14.646811 14.921180 -0.104478  3.750816
##
## Log-likelihood: -481.21
## 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
## 15.1510029708 14.7414325430 -0.0009399171  3.4010182803
##
## Log-likelihood: -494.17
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$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.
```



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

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

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

```
## [1] "Analysing data for response: IRGM1"
## [1] "Fit for the response: IRGM1"
## [1] "Fitting for all"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Fitting for groupA : F"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Fitting for groupB : M"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.24   1 0.4927543
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.27   1 0.4609059
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.23   1 0.4978844
## [1] "Testing H2 groupB no alpha vs alpha"
##   dLL dDF    pvalue
## 1   0   1 0.9569908
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.47   1 0.3297621
## [1] "Testing H3 groupB no alpha vs alpha"
```

```
##   dLL dDF    pvalue
## 1   0   1 0.9999517
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.06   1 0.7262511
## [1] "Testing H2 vs H0"
##    dLL dDF      pvalue
## 1 5.85   3 0.008523426
## [1] "Testing H3 vs H1"
##    dLL dDF      pvalue
## 1 6.74   4 0.009134627
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.96   2 0.3834328
```

```
##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
## 11.19559941 -0.07493732  3.30034466
##
## Log-likelihood: -891.91
## 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
## 11.28436166 11.10078855 -0.08197282  3.30240109
##
## Log-likelihood: -891.85
## Best method: bobyqa
##
```

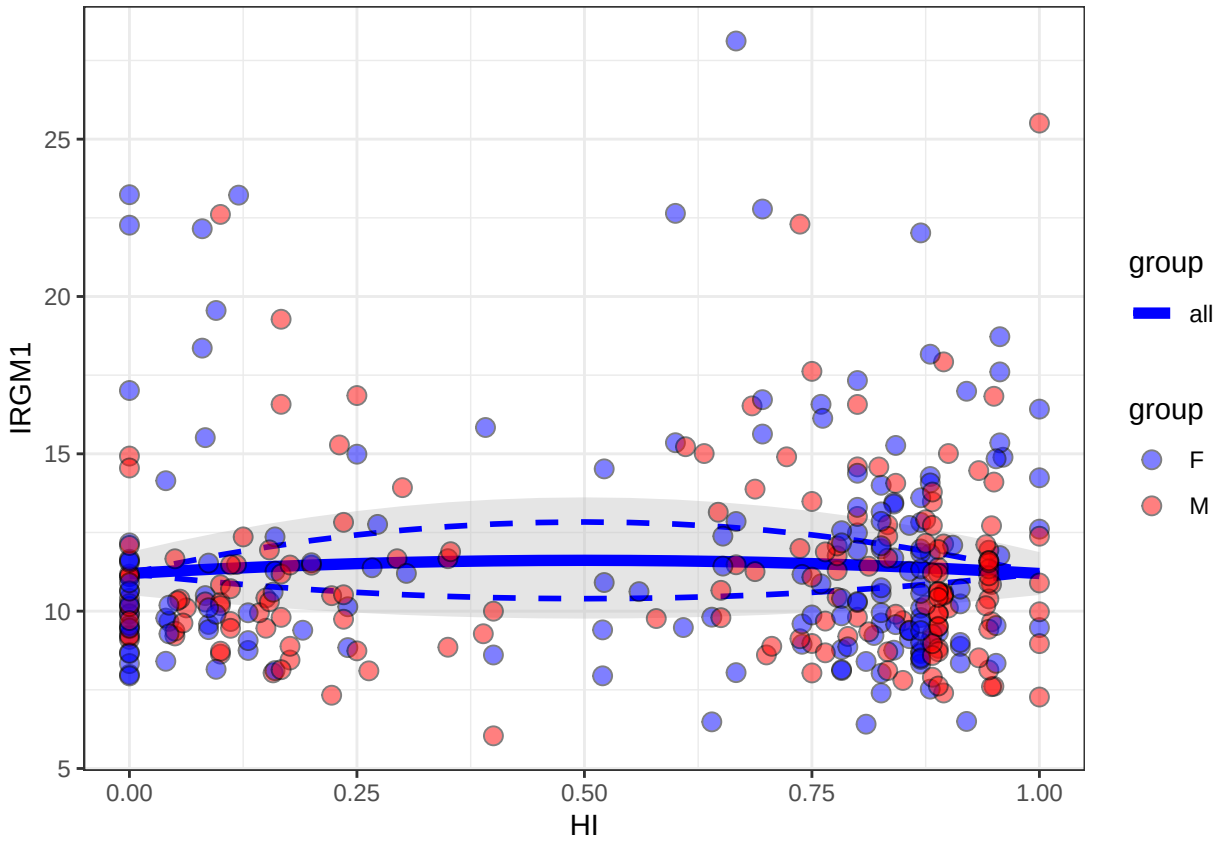
```
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 11.3905791 -0.1090206  3.0999499
##
## Log-likelihood: -459.72
## 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
## 11.126190811  0.007833011  3.648704592
##
## Log-likelihood: -426.34
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 11.7250901 10.8968237 -0.1679786  3.1220485
```

```
##
## Log-likelihood: -459.22
## 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
##  1.074277e+01  1.137318e+01 -9.759829e-06  3.662007e+00
##
## Log-likelihood: -425.88
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$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")

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

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

```
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Fitting for groupB : M"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.88   1 0.1836948
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.68   1 0.2431323
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.36   1 0.397991
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.56   1 0.2887894
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.12   1 0.6208357
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.64   1 0.2592147
## [1] "Testing H1 vs H0"
##    dLL dDF   pvalue
## 1 0.11   1 0.633915
## [1] "Testing H2 vs H0"
##    dLL dDF   pvalue
## 1 0.98   3 0.579382
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 1.51   4 0.5534792
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.64   2 0.5256382
```

```
##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
## 18.754871 -0.110056  4.629210
##
## Log-likelihood: -976.21
## 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
## 18.62382075 18.92884013 -0.09929406  4.62917360
##
## Log-likelihood: -976.1
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 19.0549443 -0.1032107  4.5931937
##
## Log-likelihood: -491.09
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
```

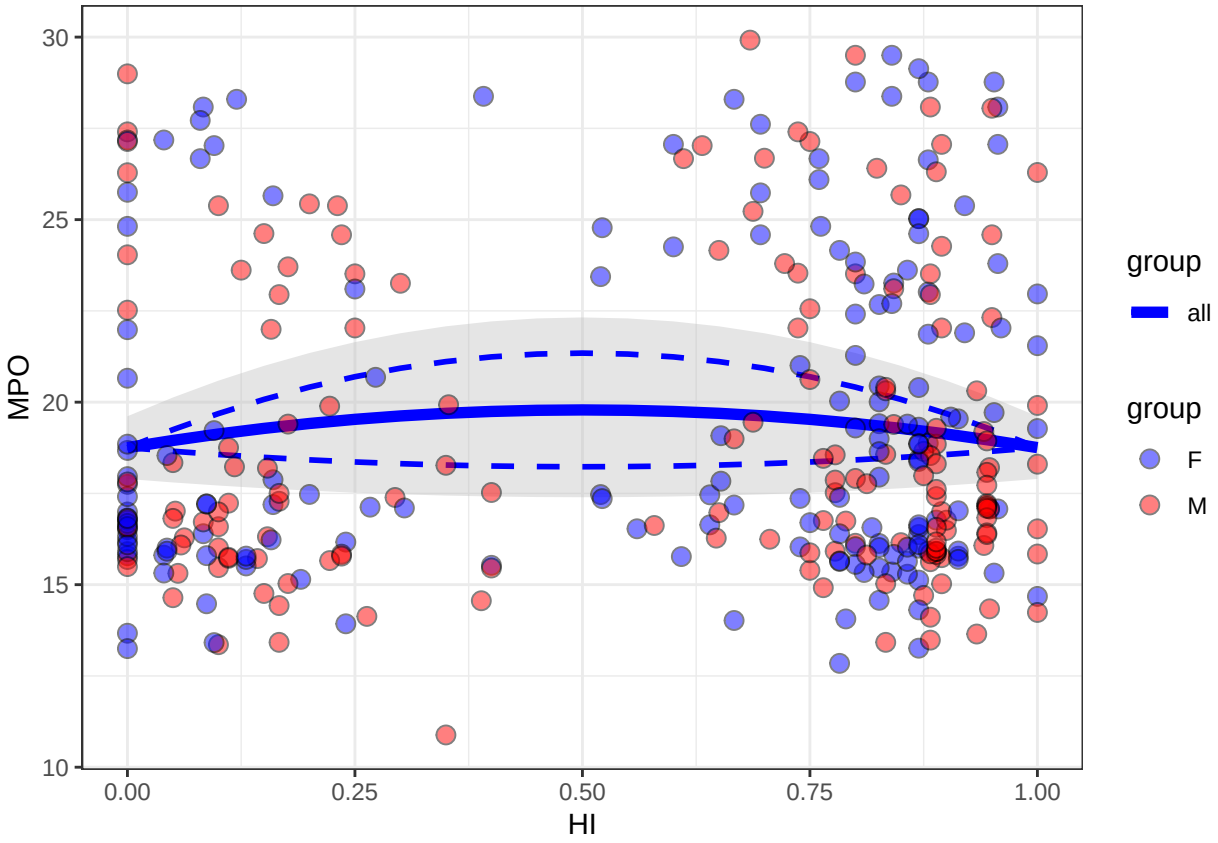
```
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##        L1     alpha   myshape
## 18.452376 -0.117932  4.693600
##
## Log-likelihood: -484.14
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 18.66472726 19.65178839 -0.06199369  4.59998139
##
## Log-likelihood: -490.54
## Best method: bobyqa
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 18.6295086 18.2485621 -0.1288016  4.6979042
##
## Log-likelihood: -484.05
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$H0,
             data = field,
             response = "MPO",
             group = "Sex") +
```

```
  scale_fill_manual(values = c("blue", "red")) +
  scale_color_manual(values = c("blue", "red")) +
  theme_bw()
```

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

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




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

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

IFNy <- 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 0.34   1 0.4119459
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.31   1 0.4293845
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.03   1 0.818322
## [1] "Testing H2 groupB no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.5   1 0.3168754
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.03   1 0.8167992
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.47   1 0.3318034
## [1] "Testing H1 vs H0"
##   dLL dDF    pvalue
## 1   0   1 0.9439104
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.91   3 0.6091541
## [1] "Testing H3 vs H1"
##    dLL dDF   pvalue
## 1 0.94   4 0.756035
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.03   2 0.9665639
```

```
##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
## 10.1482011 -0.1138808  2.6959899
##
## Log-likelihood: -927.03
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 10.1288812 10.1700892 -0.1118408  2.6956908
##
## Log-likelihood: -927.02
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
## 10.5576903 -0.0442586  2.6942756
##
## Log-likelihood: -468
## Best method: L-BFGS-B
```

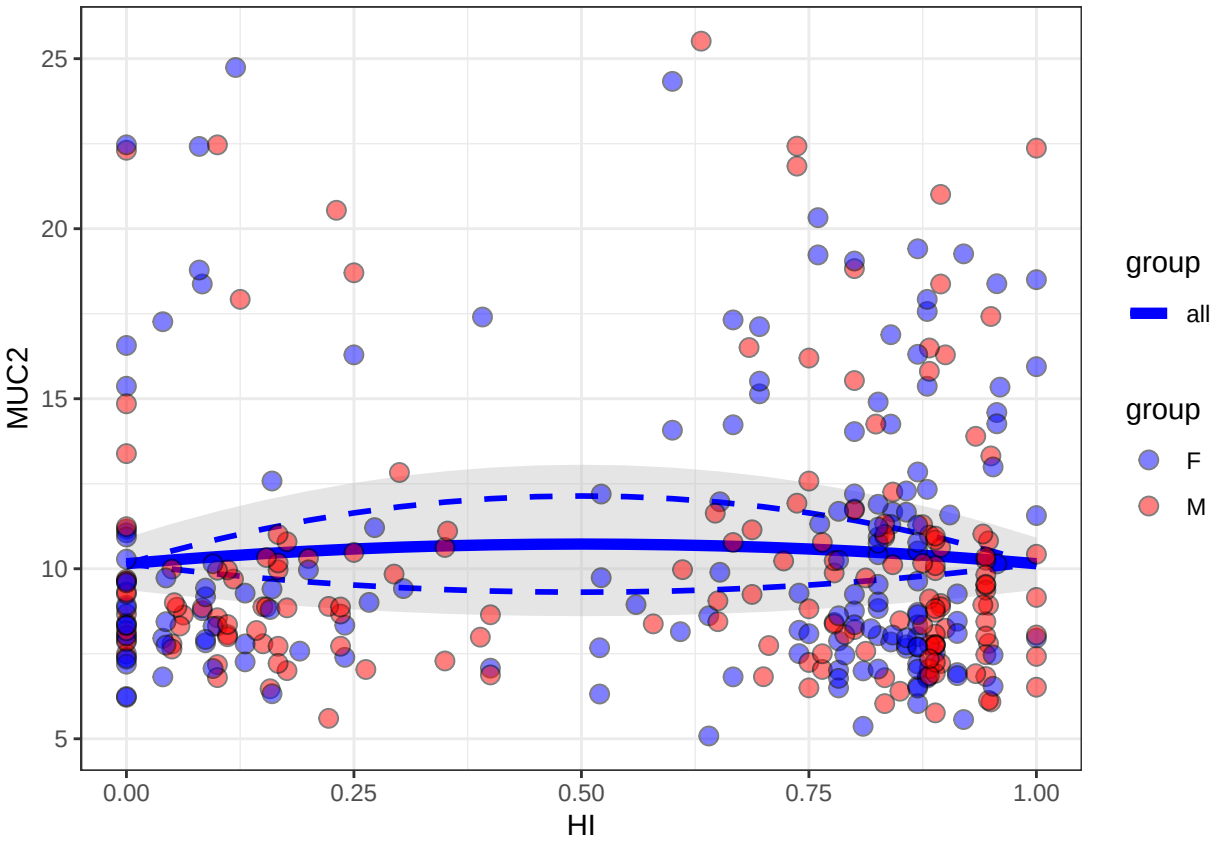
```
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##        L1     alpha   myshape
##  9.712777 -0.200091  2.709400
##
## Log-likelihood: -458.11
## 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
## 10.57250704 10.53754890 -0.04652781  2.69471223
##
## Log-likelihood: -468
## 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
```

```
##  9.6076350  9.8088333 -0.1944353  2.7093414
##
## Log-likelihood: -458.08
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$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")

IFNy <- 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 0.04   1 0.7774737
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.9163143
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.9163009
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.05   1 0.7572557
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.8757905
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.04   1 0.7811917
## [1] "Testing H1 vs H0"
##   dLL dDF    pvalue
## 1 0.4   1 0.3710776
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.42   3 0.8390002
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
```

```
## 1 0.48   4 0.9147444
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.46   2 0.6301823
```

```
##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
## 11.19018063 -0.04570668  2.43259875
##
## Log-likelihood: -984.71
## 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
## 10.89482803 11.53606022 -0.01706037  2.43489799
##
## Log-likelihood: -984.31
## 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"]]),
```

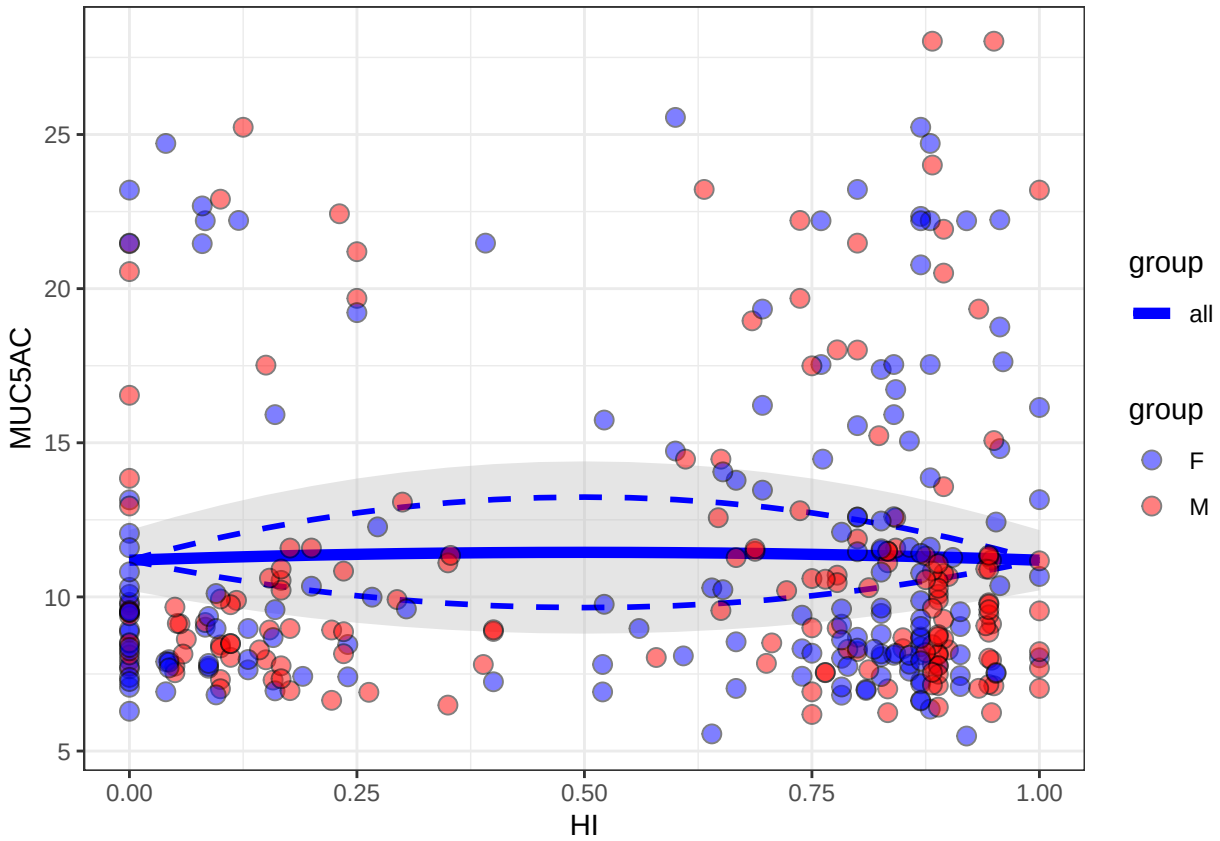
```
##     control = config$control)
##
## Coefficients:
##          L1       alpha     myshape
## 11.43407356 -0.02433222  2.40856703
##
## Log-likelihood: -496.3
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, 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
## 10.93963605 -0.06977184  2.46002231
##
## Log-likelihood: -487.98
## 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
## 11.08524393 11.97805583  0.03706757  2.40963793
##
## Log-likelihood: -495.96
## 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
## 10.68317077 11.15961692 -0.06233649  2.46303965
##
## Log-likelihood: -487.86
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$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)
```

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

IFNy <- 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 0.51   1 0.3130745
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.13   1 0.611203
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.81   1 0.2034581
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.9031706
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.24   1 0.4902722
## [1] "Testing H3 groupB no alpha vs alpha"
##   dLL dDF   pvalue
## 1   0   1 0.942868
```

```
## [1] "Testing H1 vs H0"
##    dLL dDF     pvalue
## 1 2.47   1 0.02625978
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 1.17   3 0.5060472
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 1.14   4 0.6835506
## [1] "Testing H3 vs H2"
##    dLL dDF     pvalue
## 1 2.44   2 0.08674323
```

```
##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
## 16.2986324 -0.1374078  3.2122525
##
## Log-likelihood: -1064.23
## 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
## 15.50631266 17.28908021 -0.06862561  3.21673665
##
## Log-likelihood: -1061.76
## Best method: bobyqa
##
## $H2
## $H2$groupA
```

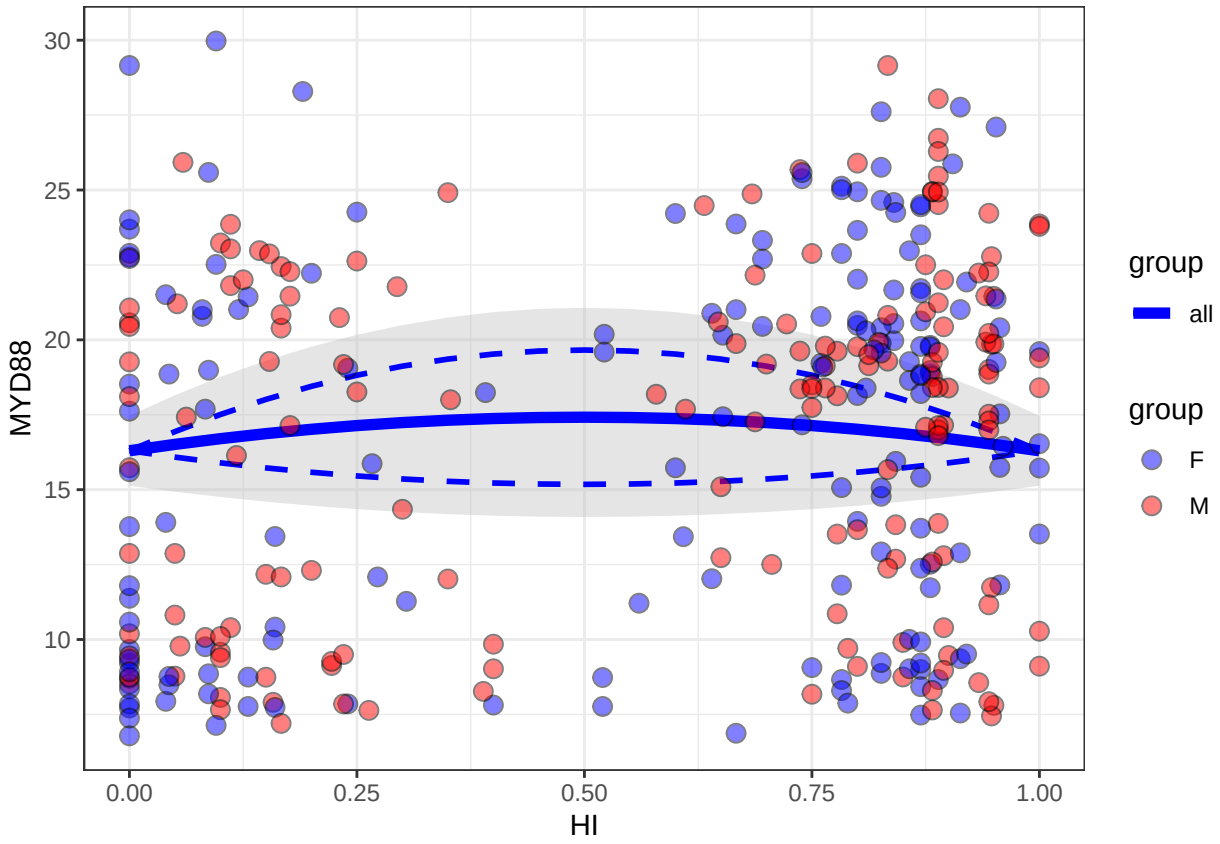
```
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##        L1     alpha   myshape
## 15.749626 -0.261844  3.042196
##
## Log-likelihood: -536.29
## 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
## 16.84579190 -0.02174222  3.41265792
##
## Log-likelihood: -526.78
## 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
## 15.2607632 16.7667979 -0.1492716  3.0359027
##
## Log-likelihood: -535.55
```

```
## 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
## 15.72515033 17.69501413 -0.01248977  3.43791893
##
## Log-likelihood: -525.07
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$H0,
             data = field,
             response = "MYD88",
             group = "Sex") +
  scale_fill_manual(values = c("blue", "red")) +
  scale_color_manual(values = c("blue", "red")) +
  theme_bw()
```

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

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




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

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

IFNy <- 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 0.85   1 0.1926329
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.76   1 0.2183278
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.14   1 0.5949384
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.92   1 0.1757295
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.13   1 0.6105077
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.87   1 0.1875805
## [1] "Testing H1 vs H0"
##    dLL dDF  pvalue
## 1 0.01   1 0.87117
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.21   3 0.9351275
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 0.27   4 0.9696534
## [1] "Testing H3 vs H2"
##    dLL dDF   pvalue
## 1 0.07   2 0.932305
```

```
##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
## 23.47968070  0.09858866  5.00000000
##
## Log-likelihood: -898.46
```

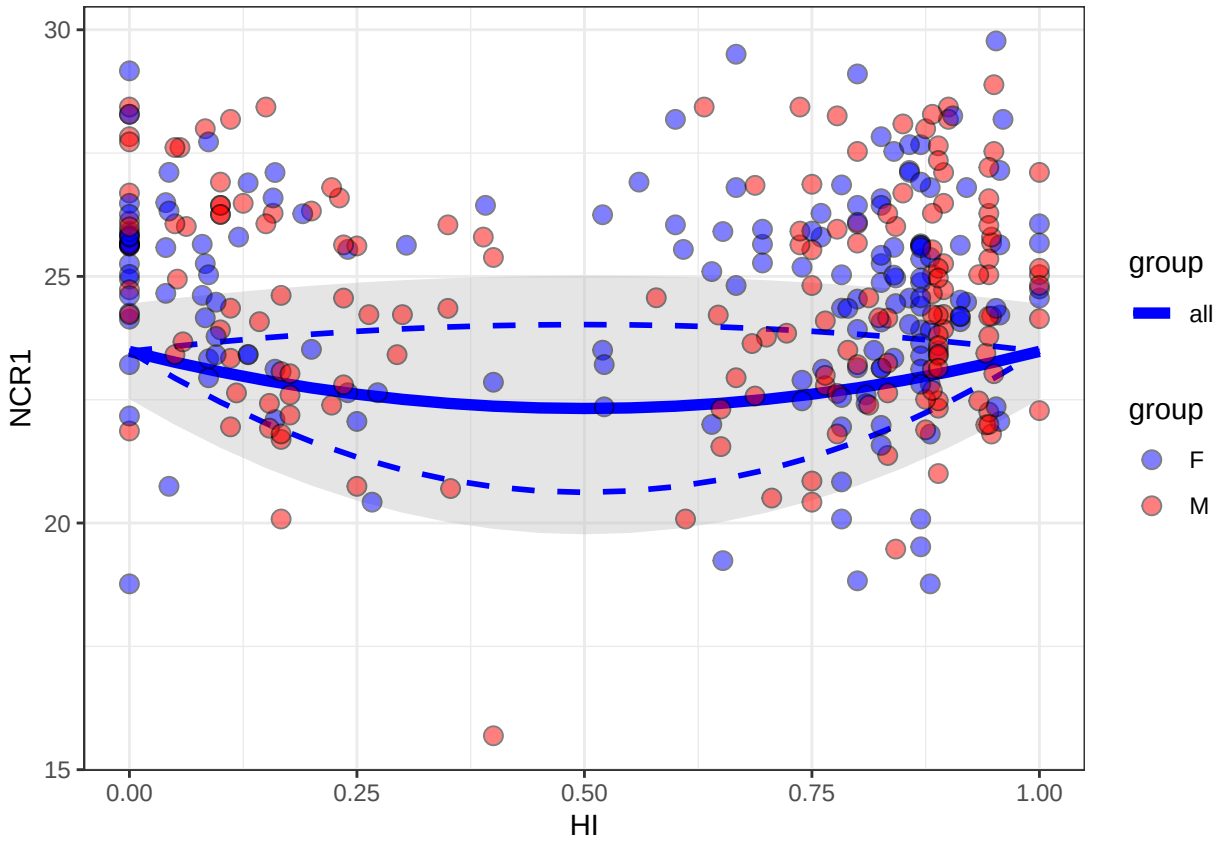
```
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 23.53250341 23.41385461  0.09587579  5.00000000
##
## Log-likelihood: -898.45
## 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
## 23.26482586  0.05503497  5.00000000
##
## Log-likelihood: -447.4
## 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
## 23.7424488  0.1502043  5.0000000
```

```
##
## Log-likelihood: -450.85
## 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
## 23.2495452 23.2909717  0.0565567  5.0000000
##
## Log-likelihood: -447.4
## Best method: bobyqa
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 23.9458833 23.5635635  0.1473126  5.0000000
##
## Log-likelihood: -450.78
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$H0,
             data = field,
             response = "NCR1",
             group = "Sex") +
  scale_fill_manual(values = c("blue", "red")) +
  scale_color_manual(values = c("blue", "red")) +
  theme_bw()
```

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

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




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

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

IFNy <- 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 0.57   1 0.2839376
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.39   1 0.374593
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.92   1 0.1751379
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.01   1 0.900514
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.66   1 0.2494584
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.9025852
## [1] "Testing H1 vs H0"
##   dLL dDF   pvalue
## 1 0.3   1 0.435094
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.37   3 0.8655757
## [1] "Testing H3 vs H1"
##   dLL dDF    pvalue
## 1 0.3   4 0.9640454
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.23   2 0.7913691
```

```
##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
## 23.09514365  0.08269669  5.00000000
```

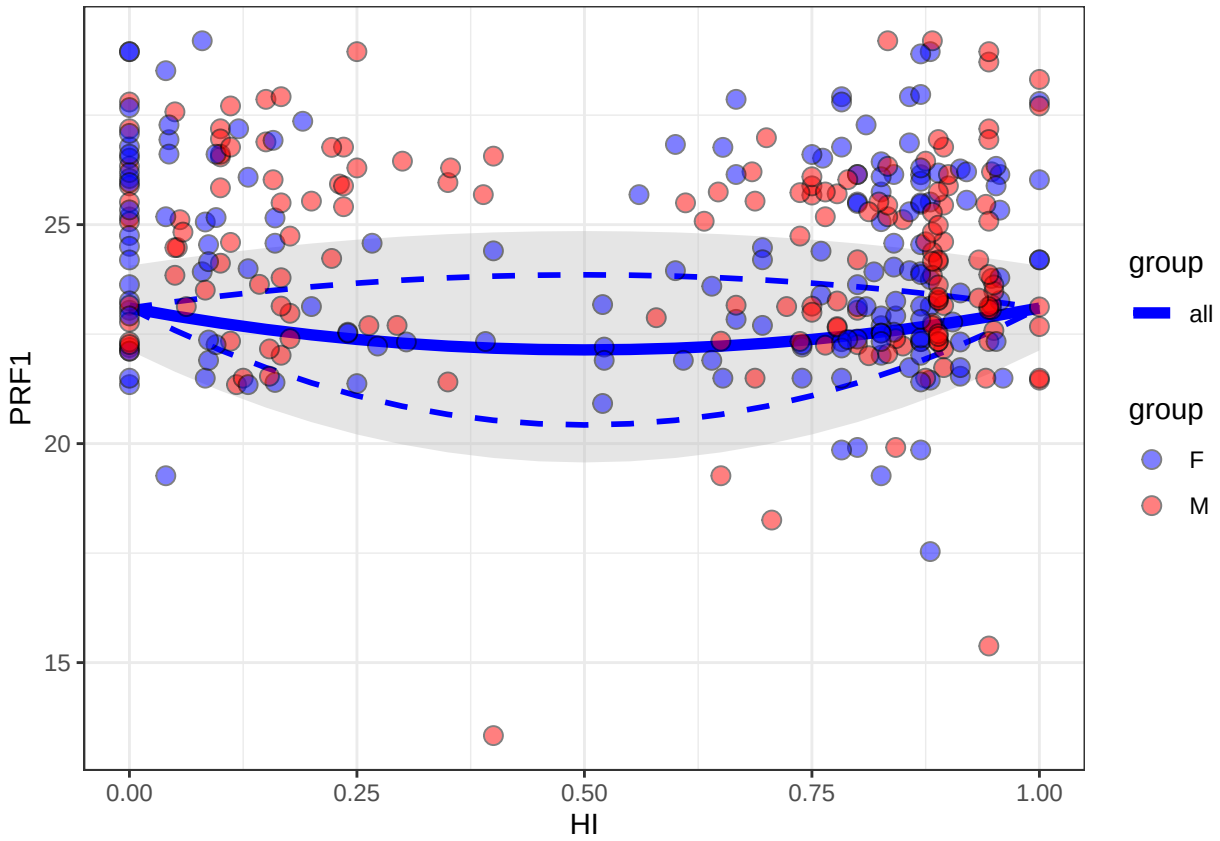
```
##
## Log-likelihood: -898.26
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 23.34591766 22.78849290  0.07060213  5.00000000
##
## Log-likelihood: -897.96
## 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
## 23.3547596  0.1405258  5.0000000
##
## Log-likelihood: -446.86
## 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
## 22.77637819  0.01448086  5.00000000
##
## Log-likelihood: -451.03
## 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
## 23.4503911 23.1790306  0.1301858  5.0000000
##
## Log-likelihood: -446.83
## 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
## 23.13315274 22.50826018  0.01425636  5.00000000
##
## Log-likelihood: -450.83
## Best method: L-BFGS-B
```

```
bananaPlot(mod = IFNy$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")

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

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

```
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Fitting for groupB : M"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.09   1 0.6752996
## [1] "Testing H1 no alpha vs alpha"
##   dLL dDF    pvalue
## 1   0   1 0.9602801
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.02   1 0.8274163
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.08   1 0.6940514
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.12   1 0.6274213
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.09   1 0.6773944
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.93   1 0.1729385
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.81   3 0.6567631
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 1.97   4 0.4140644
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 2.09   2 0.1232809
```

```
##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,
```

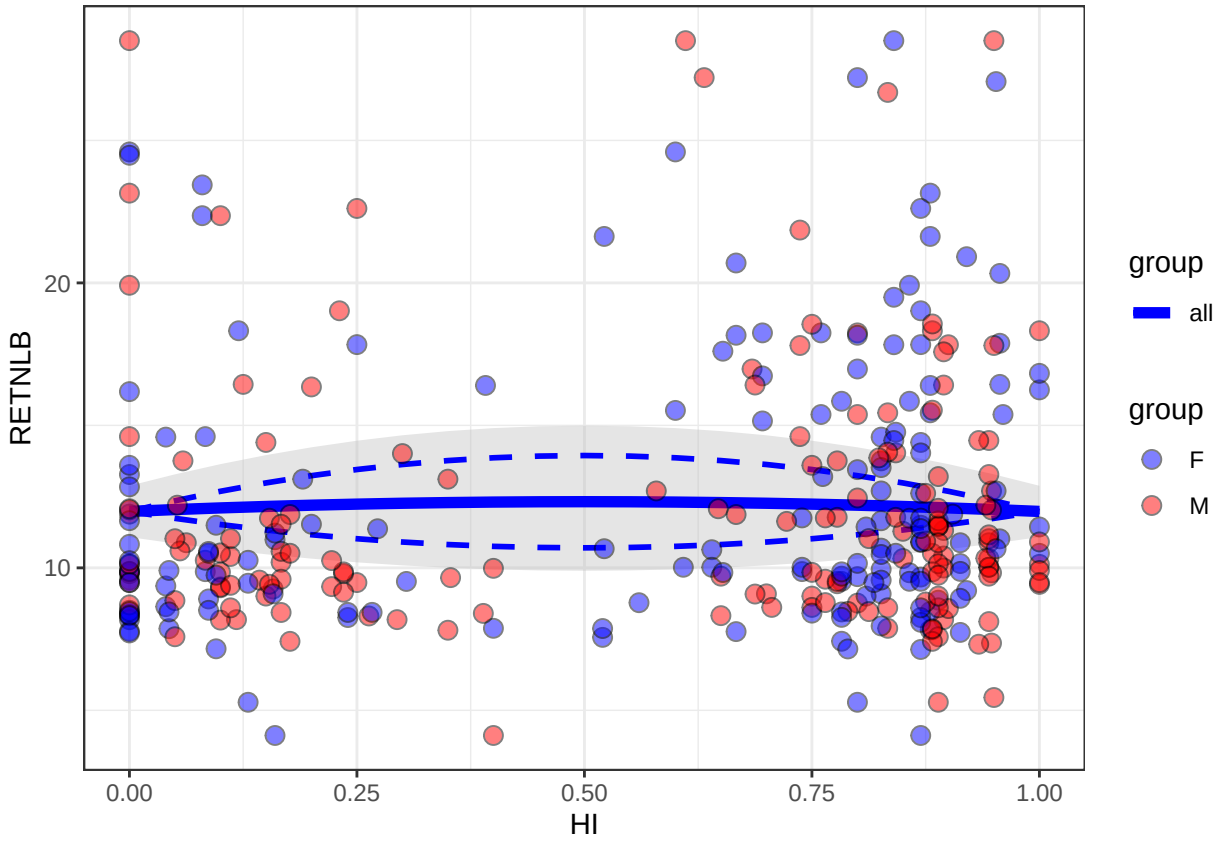
```
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1       alpha     myshape
## 11.98025381 -0.05703314  2.69643002
##
## Log-likelihood: -979.82
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##           L1           L2        alpha      myshape
## 11.560733567 12.523947050 -0.006854438  2.699198022
##
## Log-likelihood: -978.89
## 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
## 12.31694795 -0.04355707  2.69496495
##
## Log-likelihood: -494.58
## 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
## 11.6371020 -0.0727024  2.7070206
##
## Log-likelihood: -484.43
## 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
## 11.47963497 13.65113222  0.09626788  2.71933993
##
## Log-likelihood: -492.5
## 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
## 11.71141177 11.55903491 -0.07836459  2.70810144
##
## Log-likelihood: -484.42
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$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")

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

```
## [1] "Analysing data for response: SOCS1"
## [1] "Fit for the response: SOCS1"
## [1] "Fitting for all"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
```

```
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Fitting for groupA : F"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Fitting for groupB : M"
## [1] "Fitting model basic without alpha"
## [1] "Did converge"
## [1] "Fitting model basic with alpha"
## [1] "Did converge"
## [1] "Fitting model advanced without alpha"
## [1] "Did converge"
## [1] "Fitting model advanced with alpha"
## [1] "Did converge"
## [1] "Testing H0 no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.1   1 0.6485448
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.08   1 0.6827193
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.52   1 0.3095676
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.31   1 0.4302486
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.26   1 0.4733257
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.33   1 0.4189738
## [1] "Testing H1 vs H0"
##    dLL dDF   pvalue
## 1 0.02   1 0.853247
## [1] "Testing H2 vs H0"
##    dLL dDF     pvalue
## 1 3.81   3 0.05453768
## [1] "Testing H3 vs H1"
##    dLL dDF     pvalue
## 1 4.96   4 0.04192149
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 1.16   2 0.3124842
```

```
##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
## 12.47978310  0.04433325  3.56671359
##
## Log-likelihood: -895.85
## 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
## 12.52829370 12.42740777  0.04064683  3.56873122
##
## Log-likelihood: -895.83
## Best method: bobyqa
##
## $H2
## $H2$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1      alpha    myshape
```

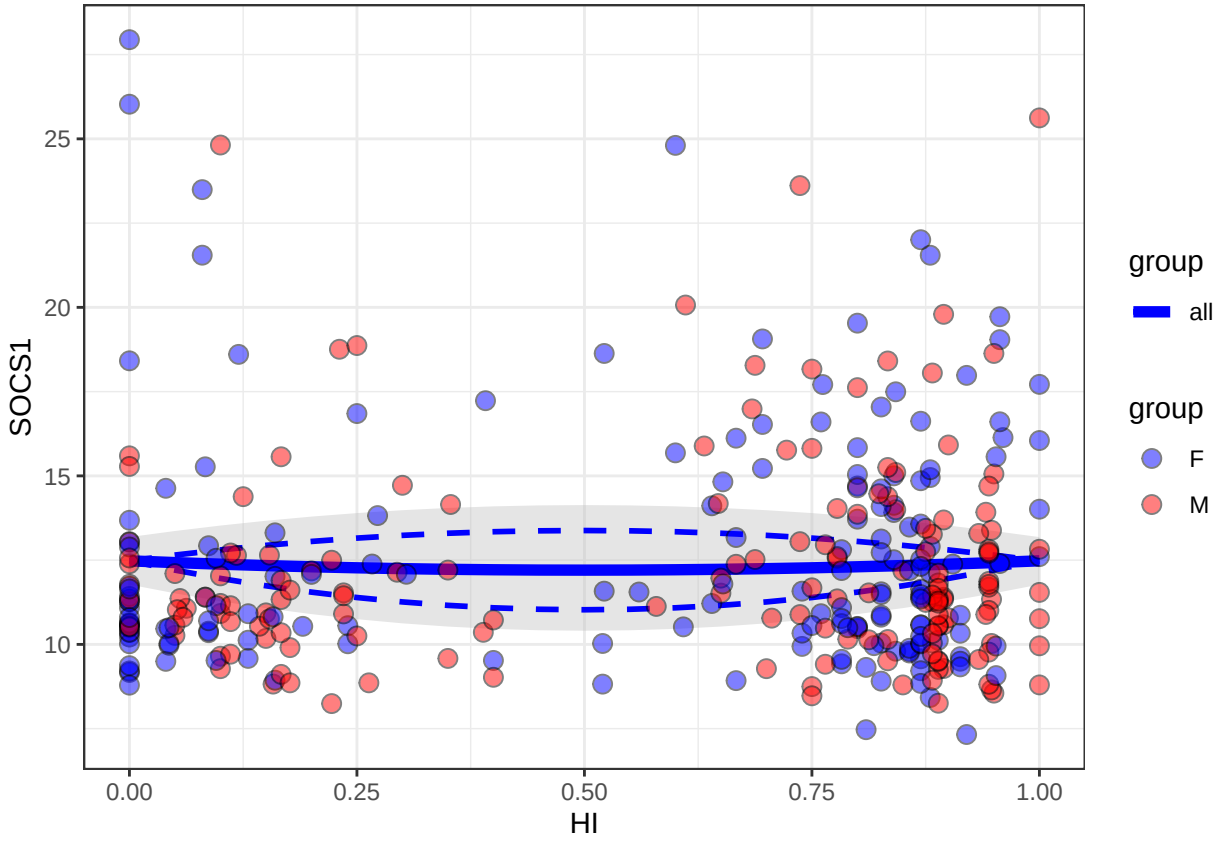
```
## 13.0361320  0.1368053  3.4447276
##
## Log-likelihood: -457.09
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$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
## 11.7697137 -0.1119764  3.7746229
##
## Log-likelihood: -434.94
## 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
## 13.3177711 12.6583063  0.1026318  3.4670416
##
## Log-likelihood: -456.8
## 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
## 11.2507889 12.1387258 -0.1140703  3.7955853
##
## Log-likelihood: -434.07
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$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")

IFNy <- 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 0.22   1 0.5037225
## [1] "Testing H1 no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.25   1 0.4795001
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.01   1 0.9159583
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.57   1 0.2875548
## [1] "Testing H3 groupA no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.06   1 0.7298372
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.51   1 0.3121757
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.04   1 0.7824371
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.36   3 0.8686206
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 1.96   4 0.4163236
## [1] "Testing H3 vs H2"
```

```
##    dLL dDF    pvalue
## 1 1.64   2 0.1938556
```

```
##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
## 20.61868464  0.05446208  5.00000000
##
## Log-likelihood: -898.42
## 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
## 20.53603913 20.71213328  0.05835373  5.00000000
##
## Log-likelihood: -898.38
## 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)
##
```

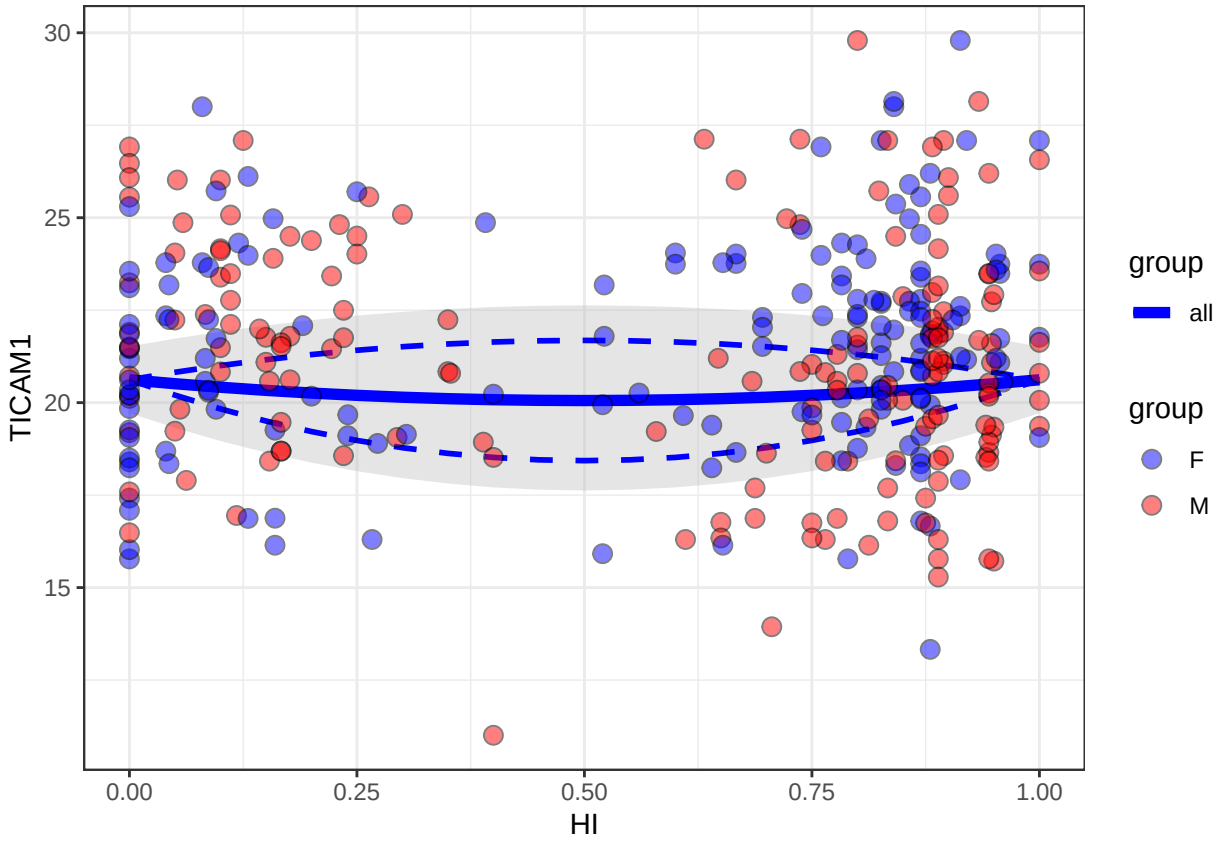
```
## Coefficients:
##          L1       alpha     myshape
## 20.33826707 -0.01232009  5.00000000
##
## Log-likelihood: -444.13
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     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
## 20.9029777  0.1210809  5.0000000
##
## Log-likelihood: -453.93
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 19.80926933 21.15065608  0.04130248  5.00000000
##
## Log-likelihood: -443.09
## Best method: bobyqa
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
```

```
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##         L1         L2      alpha    myshape
## 21.4417557 20.4513218  0.1164115  5.0000000
##
## Log-likelihood: -453.32
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$H0,
             data = field,
             response = "TICAM1",
             group = "Sex") +
  scale_fill_manual(values = c("blue", "red")) +
  scale_color_manual(values = c("blue", "red")) +
  theme_bw()
```

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

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




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

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

```
IFNy <- 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 0.13   1 0.6150504
## [1] "Testing H1 no alpha vs alpha"
##   dLL dDF    pvalue
## 1 0.1   1 0.6559595
## [1] "Testing H2 groupA no alpha vs alpha"
##    dLL dDF   pvalue
## 1 0.01   1 0.876648
## [1] "Testing H2 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.38   1 0.3810045
## [1] "Testing H3 groupA no alpha vs alpha"
##   dLL dDF    pvalue
## 1   0   1 0.9271205
## [1] "Testing H3 groupB no alpha vs alpha"
##    dLL dDF    pvalue
## 1 0.32   1 0.4206925
## [1] "Testing H1 vs H0"
##    dLL dDF    pvalue
## 1 0.03   1 0.8035592
## [1] "Testing H2 vs H0"
##    dLL dDF    pvalue
## 1 0.27   3 0.9097406
```

```
## [1] "Testing H3 vs H1"
##    dLL dDF    pvalue
## 1 1.16   4 0.6784149
## [1] "Testing H3 vs H2"
##    dLL dDF    pvalue
## 1 0.92   2 0.3998039
```

```
##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
## 20.96793671  0.03843686  5.00000000
##
## Log-likelihood: -897.06
## Best method: bobyqa
##
## $H1
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 21.04243399 20.88091647  0.03477237  5.00000000
##
## Log-likelihood: -897.03
## 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
## 20.69112183 -0.01680606  5.00000000
##
## Log-likelihood: -441.49
## Best method: bobyqa
##
## $H2$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L1, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], alpha = paramBounds[["alphaLB"]],
##         myshape = paramBounds[["myshapeLB"]]), upper = c(L1 = paramBounds[["L1UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1       alpha     myshape
## 21.25337880  0.09463402  5.00000000
##
## Log-likelihood: -455.3
## Best method: bobyqa
##
##
## $H3
## $H3$groupA
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 20.42062926 21.11497687  0.01036138  5.00000000
##
## Log-likelihood: -441.22
## Best method: bobyqa
##
## $H3$groupB
##
## Call:
## bbmle::mle2(minuslogl = response ~ dweibull(shape = myshape,
```

```
##     scale = MeanLoad(L1, L2, alpha, HI)/gamma(1 + (1/myshape))),
##     start = start, method = config$method, optimizer = config$optimizer,
##     data = data, lower = c(L1 = paramBounds[["L1LB"]], L2 = paramBounds[["L2LB"]],
##         alpha = paramBounds[["alphaLB"]], myshape = paramBounds[["myshapeLB"]]),
##     upper = c(L1 = paramBounds[["L1UB"]], L2 = paramBounds[["L2UB"]],
##         alpha = paramBounds[["alphaUB"]], myshape = paramBounds[["myshapeUB"]]),
##     control = config$control)
##
## Coefficients:
##          L1          L2       alpha     myshape
## 21.81181965 20.77037818  0.08825218  5.00000000
##
## Log-likelihood: -454.65
## Best method: bobyqa
```

```
bananaPlot(mod = IFNy$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.
```