Supplementary Information (SI) for Lab on a Chip.

OoTrap: Enhancing Oocyte Collection and Maturation with a Field-Deployable Fluidic Device

R. Franko[1,2], M.A.M.M. Ferraz[1,2*]

[1]Clinic of Ruminants, Faculty of Veterinary Medicine, Ludwig-Maximilians-Universität München, Sonnenstr. 16, Oberschleißheim, 85764, Germany
[2]Gene Center, Ludwig-Maximilians-Universität München, Feodor-Lynen Str. 25, Munich, 81377, Germany.

*Corresponding author:
Phone: +49(0)89 2180-78683
Email: m.ferraz@lmu.de

Supplementary File 2

allcodes.R

marciaferraz

2024-08-06

```
setwd("~/Desktop/R/Roksan/OoTrap/final codes")

library(reshape2)
library(nlme)

## Warning: package 'nlme' was built under R version 4.3.3

library(multcomp)

## Loading required package: mvtnorm

## Warning: package 'mvtnorm' was built under R version 4.3.3

## Loading required package: survival

## Warning: package 'survival' was built under R version 4.3.3

## Loading required package: TH.data

## Loading required package: MASS

## Warning: package 'MASS' was built under R version 4.3.1

##
## Attaching package: 'TH.data'

## The following object is masked from 'package:MASS':
##
##     geyser

library(viridis)

## Warning: package 'viridis' was built under R version 4.3.1

## Loading required package: viridisLite

library(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.3.1

library(gridExtra)
library(survival)
library(survminer)

## Loading required package: ggpubr

##
## Attaching package: 'survminer'
```

```
## The following object is masked from 'package:survival':
##
##     myeloma
```

```
library(lme4)
```

```
## Warning: package 'lme4' was built under R version 4.3.1
```

```
## Loading required package: Matrix
```

```
## Warning: package 'Matrix' was built under R version 4.3.1
```

```
##
## Attaching package: 'lme4'
```

```
## The following object is masked from 'package:nlme':
##
##     lmList
```

```
library(MuMIn)
library(MASS)
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.3.1
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:gridExtra':
##
##     combine
```

```
## The following object is masked from 'package:MASS':
##
##     select
```

```
## The following object is masked from 'package:nlme':
##
##     collapse
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

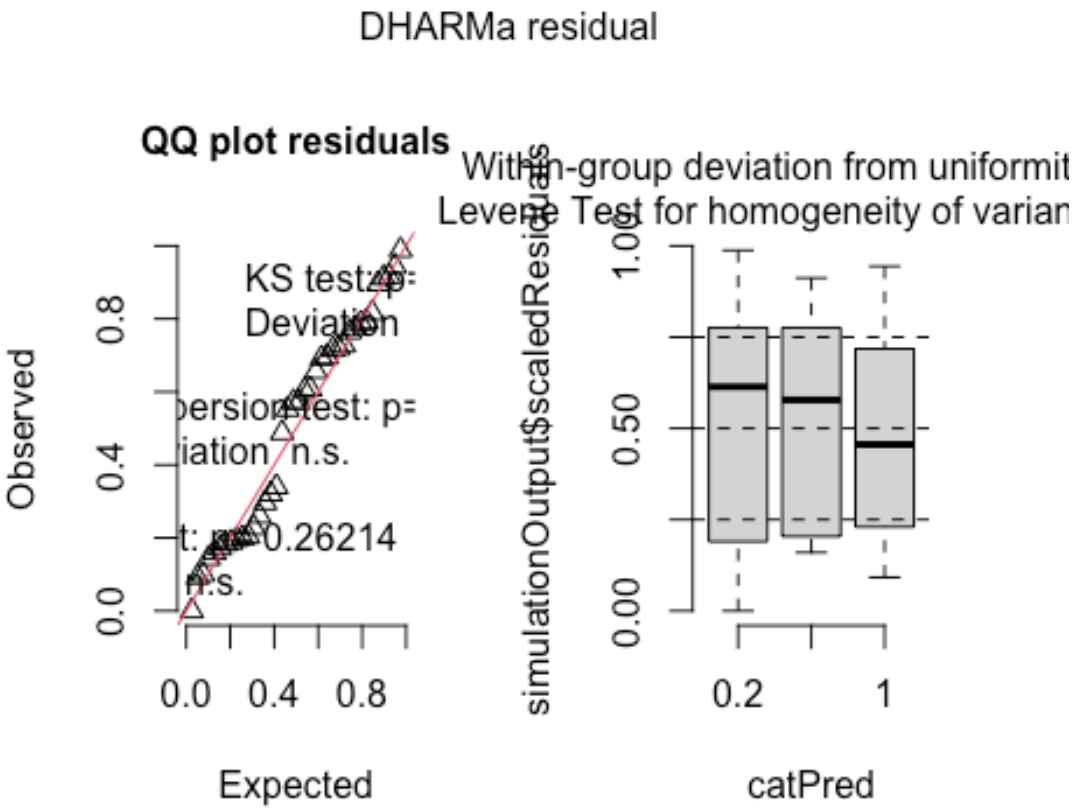
```
library(glmmTMB)
```

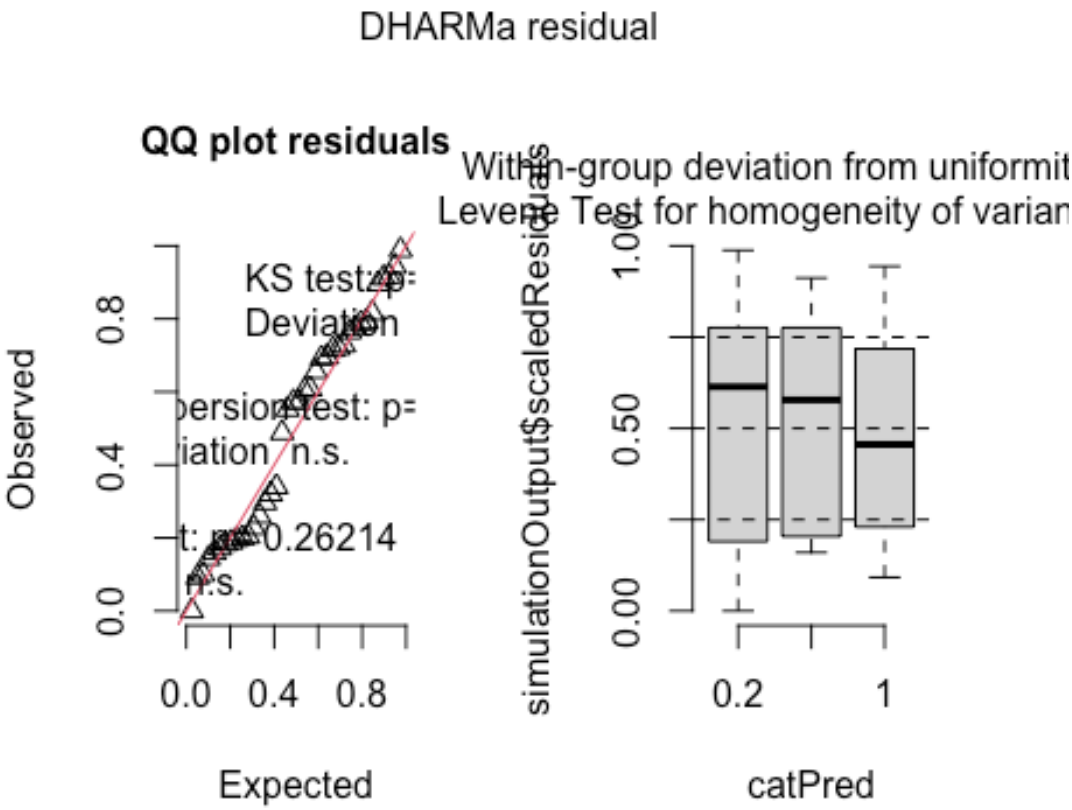
```
## Warning: package 'glmmTMB' was built under R version 4.3.1
```

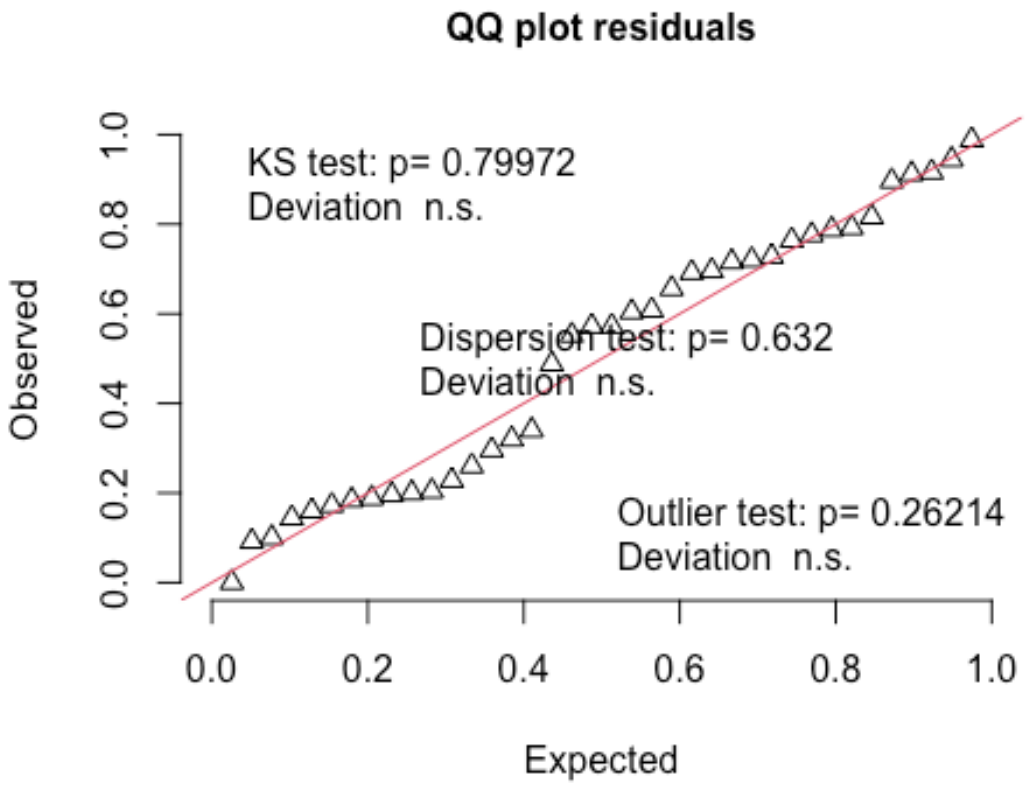
```
## Warning in checkDepPackageVersion(dep_pkg = "TMB"): Package version incons
istency detected.
```

```
## glmmTMB was built with TMB version 1.9.10
## Current TMB version is 1.9.14
## Please re-install glmmTMB from source or restore original 'TMB' package (s
ee '?reinstalling' for more information)

library(DHARMa)

## This is DHARMa 0.4.6. For overview type '?DHARMa'. For recent changes, typ
e news(package = 'DHARMa')

library(emmeans)

## Warning: package 'emmeans' was built under R version 4.3.3

## Welcome to emmeans.
## Caution: You lose important information if you filter this package's resul
ts.
## See '? untidy'

################Follicular Fluid experiments##################

#data input
data <- read.csv("IVM1_FF.csv")
data$Treatment <- as.factor(data$Treatment)
data$Replicate <- as.factor(data$Replicate)



# Means
means <- aggregate(Maturation ~ Treatment, data = data, FUN = "mean")

# SDs
sds <- aggregate(Maturation ~ Treatment, data = data, FUN = "sd")

# Combine means and SDs
means$SD <- sds$Maturation

# Update column names
colnames(means)[2:3] <- c("Mean", "SD")  # Update column names

write.csv(means, file = "Summary_Statistics_maturation_FF.csv", row.names = F
ALSE)



# Stats

# Adjust the data to avoid exact 0 and 1 values
epsilon <- 1e-5
data$AdjustedMaturation <- (data$Maturation / 100) * (1 - 2 * epsilon) + epsi
lon
```

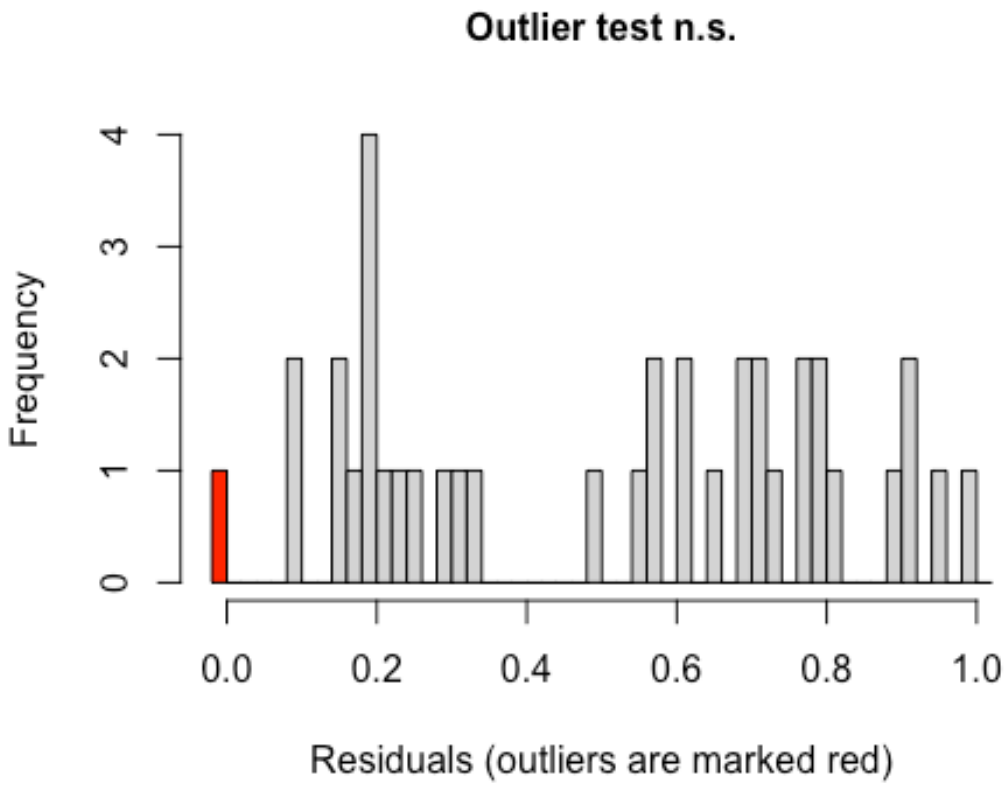
```r
# Function to fit model, check diagnostics, and perform Tukey test for beta f
amily
fit_check_tukey_beta <- function(data) {
  # Fit the model using beta family
  fit <- glmmTMB(AdjustedMaturation ~ Treatment + (1|Replicate),
                 family = beta_family(),
                 data = data)

  # Simulate residuals and perform diagnostic checks
  sim_res <- simulateResiduals(fit, plot = TRUE)
  plot(sim_res)
  print(testUniformity(sim_res))
  print(testOutliers(sim_res))
  print(testDispersion(sim_res))

  # Tukey post-hoc test
  tukey <- glht(fit, linfct = mcp(Treatment = "Tukey"))
  tukey_summary <- summary(tukey)

  return(list(summary = summary(fit), tukey_summary = tukey_summary))
}

# Adjust the data to avoid exact 0 and 1 values
epsilon <- 1e-5
data$AdjustedMaturation <- (data$Maturation / 100) * (1 - 2 * epsilon) + epsi
lon

# Apply to the whole dataset
results_beta <- fit_check_tukey_beta(data)
```


DHARMa residual
QQ plot residuals
KS test: p=
Deviation
ersion test: p=
iation n.s.
t: p= 0.26214
n.s.
Observed
Expected
0.0
0.4
0.8
Within-group deviation from uniformit
Levene Test for homogeneity of varian
simulationOutput$scaledResiduals
0.00
0.50
1.00
0.2
1
catPred

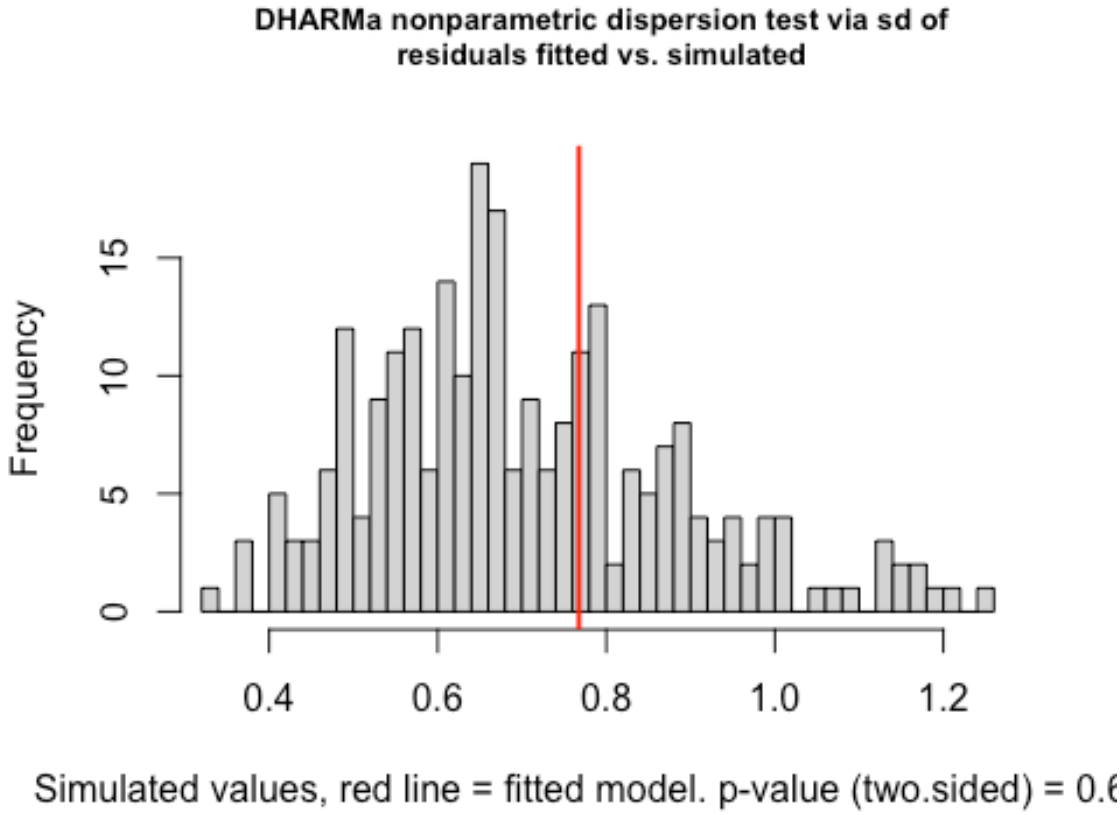

DHARMa residual
QQ plot residuals
KS test: p=
Deviation
ersion test: p=
iation n.s.
t: p= 0.26214
n.s.
Observed
Expected
0.0
0.4
0.8
Within-group deviation from uniformit
Levene Test for homogeneity of varian
simulationOutput$scaledResiduals
0.00
0.50
1.00
0.2
1
catPred




```
##
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  simulationOutput$scaledResiduals
## D = 0.10463, p-value = 0.7997
## alternative hypothesis: two-sided
```




```
##
##  DHARMa outlier test based on exact binomial test with approximate
##  expectations
##
## data:  sim_res
## outliers at both margin(s) = 1, observations = 38, p-value = 0.2621
## alternative hypothesis: true probability of success is not equal to 0.0079
68127
## 95 percent confidence interval:
##  0.0006660362 0.1380990298
## sample estimates:
## frequency of outliers (expected: 0.00796812749003984 )
##                                                0.02631579
```




```
##
##  DHARMa nonparametric dispersion test via sd of residuals fitted vs.
##  simulated
##
## data:  simulationOutput
## dispersion = 1.0938, p-value = 0.632
## alternative hypothesis: two.sided
```

```
# Print summaries and Tukey results for beta family
print("Beta Family Model Summary:")
```

```
## [1] "Beta Family Model Summary:"
```

```
print(results_beta$summary)
```

```
##  Family: beta  ( logit )
## Formula:          AdjustedMaturation ~ Treatment + (1 | Replicate)
## Data: data
##
##      AIC      BIC   logLik deviance df.resid
##    -70.4    -62.2     40.2    -80.4       33
##
## Random effects:
##
```

```
## Conditional model:
##  Groups    Name        Variance Std.Dev.
##  Replicate (Intercept) 0.03029  0.174
## Number of obs: 38, groups:  Replicate, 7
##
## Dispersion parameter for beta family (): 36.1
##
## Conditional model:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)          0.2582     0.1172   2.203   0.0276 *
## TreatmentPerfusion   0.5709     0.1439   3.967 7.29e-05 ***
## TreatmentStatic      0.4211     0.1342   3.138   0.0017 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
print("Beta Family Tukey Summary:")
```

```
## [1] "Beta Family Tukey Summary:"
```

```
print(results_beta$tukey_summary)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: glmmTMB(formula = AdjustedMaturation ~ Treatment + (1 | Replicate),
##     data = data, family = beta_family(), ziformula = ~0, dispformula = ~1)
##
## Linear Hypotheses:
##                           Estimate Std. Error z value Pr(>|z|)
## Perfusion - 2D == 0         0.5709     0.1439   3.967  < 0.001 ***
## Static - 2D == 0            0.4211     0.1342   3.138  0.00476 **
## Static - Perfusion == 0    -0.1498     0.1450  -1.033  0.55572
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

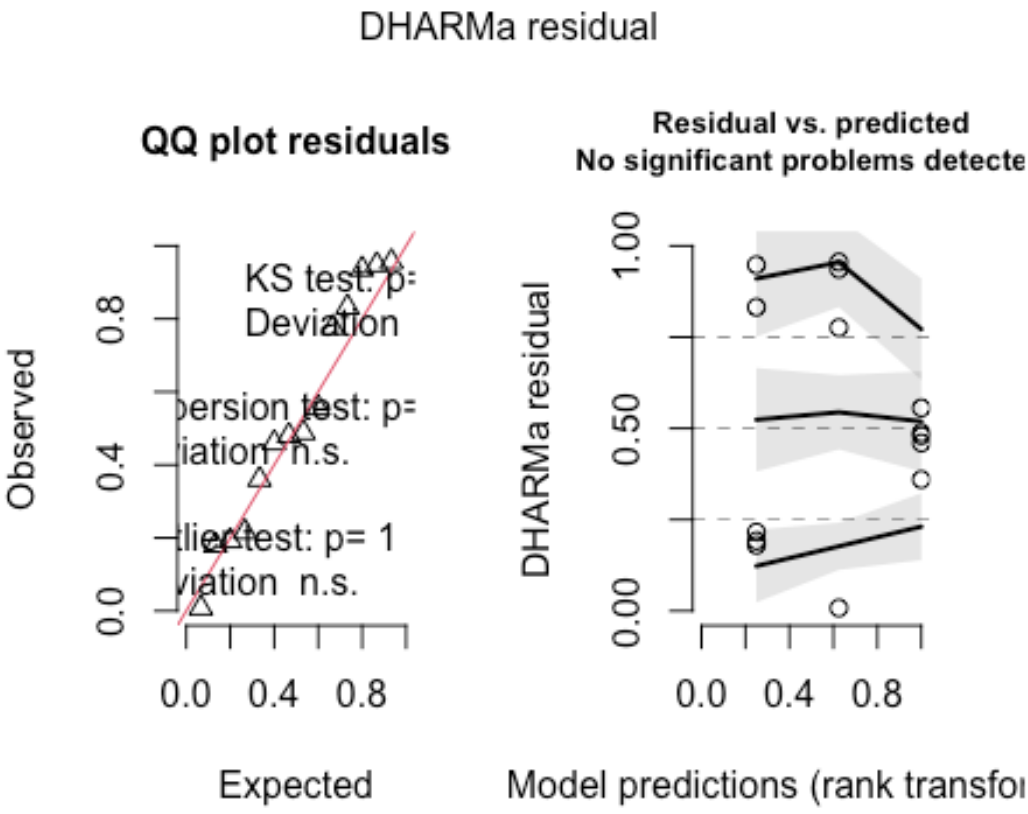
```
# Plot

png(filename="maturationFF.png",
    width = 4, height = 4, units = "in",
    res = 600)

par(mgp=c(1.5,0.5,0),
    mar = c(4, #bottom
            4, #left
            3, #top
            2),
    cex.main = 0.8,
```

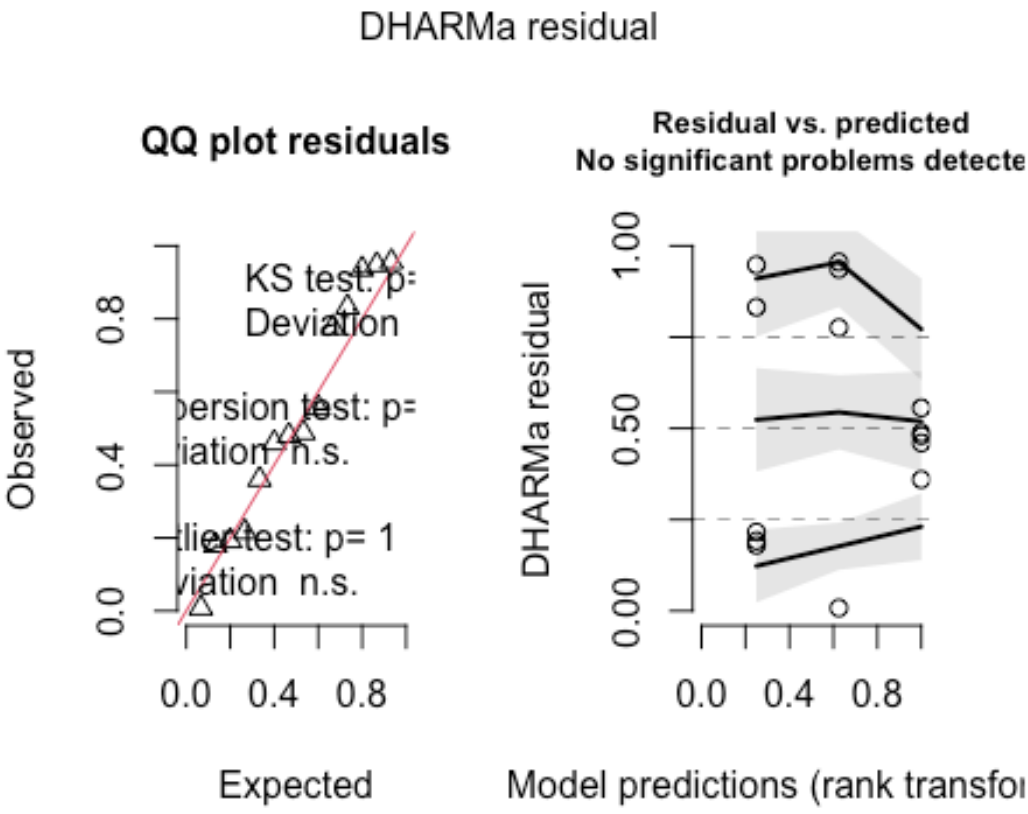
```
    family = "sans" #right
)


# Define the desired order of X-axis labels
TIME <- c("2D", "Perfusion", "Static")

boxplot(Maturation ~ Treatment,
        data = data,
        cex.axis = 0.8,
        col = viridis(4, alpha = 0.7),
        lwd = 0.7,
        ylab = "",
        ylim = c(0, 100),
        axes = FALSE,
        frame.plot = TRUE,
        xlab = "")

groups <- 3
numbox <- 1
total <- groups * numbox
xpoints <- seq(median(1:numbox), total, numbox)

axis(2, seq(0, 100, by = 10), las = 1, cex.axis = 0.7)
axis(1, labels = TIME, at = xpoints, las = 1, cex.axis = 0.7)

stripchart(data$Maturation ~ data$Treatment,
           method = "jitter",
           pch = 19,
           col = "black",
           vertical = TRUE,
           add = TRUE,
           cex = 0.3)

title(ylab = expression("Mature oocytes (%)"), line = 1.7)




dev.off()

## quartz_off_screen
##                 2

# Spindle anaysis

#data input
data <- read.csv("Spindle.csv")
data$Treatment <- as.factor(data$Treatment)
```

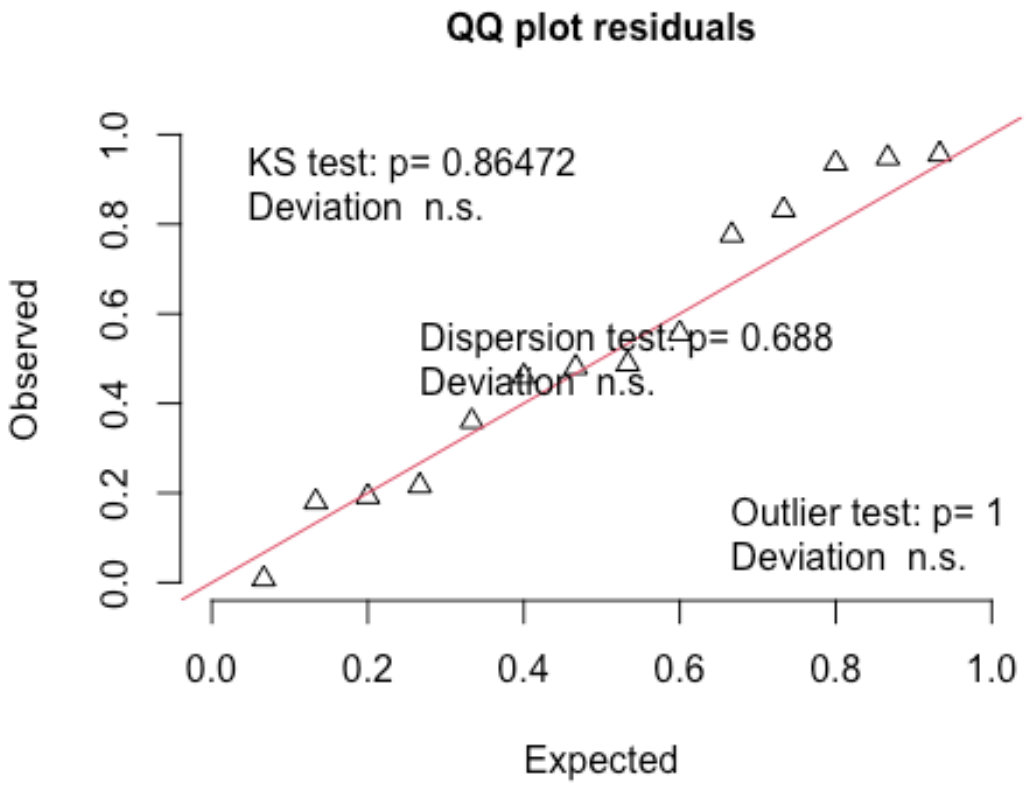
```r
data$Replicate <- as.factor(data$Replicate)
data$Device <- as.factor(data$Device)


# Filter the data for 'Spindle' with only 'N' and 'A' and for 'Stage' with 'M
2'
filtered_data <- data %>%
  filter(Stage == 'M2')

# Calculate the total count of 'M2' stage observations for each treatment, re
plicate, and device
total_counts <- filtered_data %>%
  group_by(Treatment, Replicate, Device) %>%
  summarise(Total_M2 = n(), .groups = 'drop')

# Join the total counts back to the filtered data for 'A' and 'N' spindles
filtered_data <- filtered_data %>%
  filter(Spindle %in% c('N', 'A')) %>%
  left_join(total_counts, by = c("Treatment", "Replicate", "Device"))

# Now calculate the frequency as a percentage of the total 'M2' for each trea
tment
frequency_data <- filtered_data %>%
  group_by(Treatment, Replicate, Device, Spindle) %>%
  summarise(Frequency = n() / first(Total_M2) * 100, .groups = 'drop')  # Cal
culate the proportion


# Plot abnormal spindle

#data input
data <- read.csv("spindle_freq.csv")
data$Treatment <- as.factor(data$Treatment)
data$Replicate <- as.factor(data$Replicate)
data$Device <- as.factor(data$Device)


# Open the PNG device
png(filename="spindle.png",
    width = 4, height = 4, units = "in",
    res = 600)

# Set graphical parameters
par(mgp=c(1.5,0.5,0),
    mar = c(4, 4, 3, 2), # bottom, left, top, right
    cex.main = 0.8,
    family = "sans")

# Filter the data to include only Spindle A
```

```r
frequency_data_A <- data[data$Spindle == 'A',]


# Define the desired order of X-axis labels
TIME <- c("2D", "Perfusion", "Static")

boxplot(Frequency ~ Treatment,
        data = frequency_data_A,
        cex.axis = 0.8,
        col = viridis(4, alpha = 0.7),
        lwd = 0.7,
        ylab = "",
        ylim = c(0, 100),
        axes = FALSE,
        frame.plot = TRUE,
        xlab = "")

groups <- 3
numbox <- 1
total <- groups * numbox
xpoints <- seq(median(1:numbox), total, numbox)

axis(2, seq(0, 100, by = 10), las = 1, cex.axis = 0.7)
axis(1, labels = TIME, at = xpoints, las = 1, cex.axis = 0.7)

stripchart(frequency_data_A$Frequency ~ frequency_data_A$Treatment,
           method = "jitter",
           pch = 19,
           col = "black",
           vertical = TRUE,
           add = TRUE,
           cex = 0.3)

title(ylab = expression(atop("Abnormal Spindle", "and/or Chromosome alignment
(%)")), line = 1.5)



dev.off()

## quartz_off_screen
##                 2

# Stats Spindle


# Means
means <- aggregate(Frequency ~ Treatment, data = frequency_data_A, FUN = "mea
n")
```

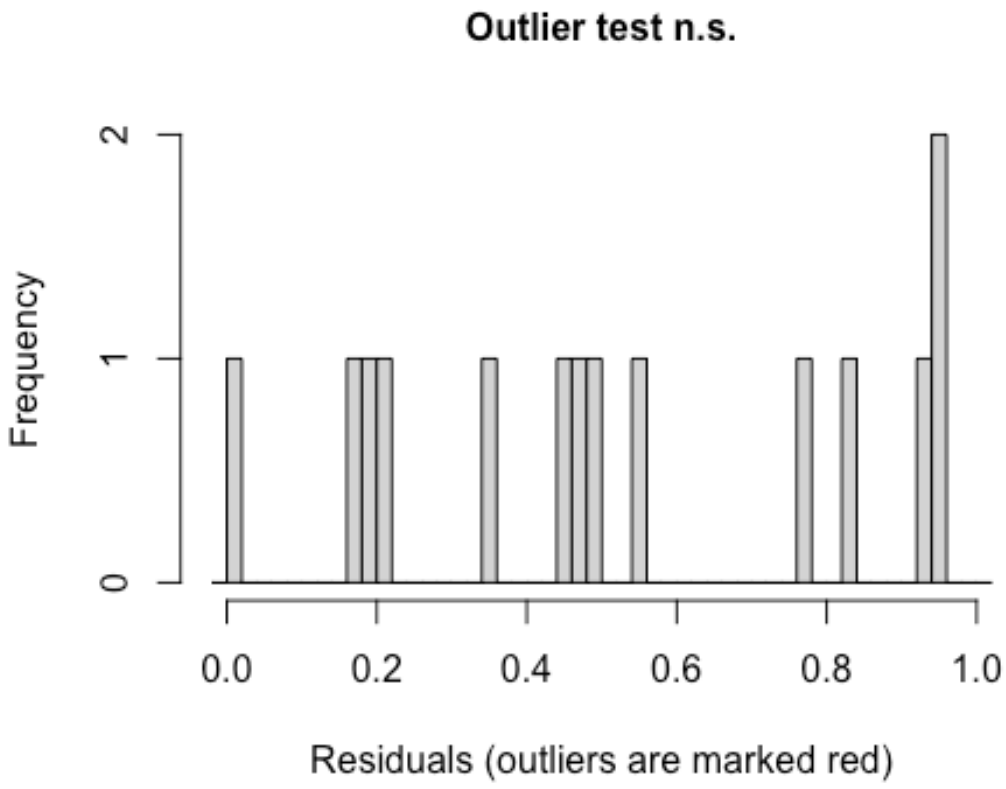
```r
# SDs
sds <- aggregate(Frequency ~ Treatment, data = frequency_data_A, FUN = "sd")

# Combine means and SDs
means$SD <- sds$Frequency

# Update column names
colnames(means)[2:3] <- c("Mean", "SD")  # Update column names

write.csv(means, file = "Summary_Statistics_spindle.csv", row.names = FALSE)


# Adjust the data to avoid exact 0 and 1 values
epsilon <- 1e-5
frequency_data_A$AdjustedFrequency <- (frequency_data_A$Frequency / 100) * (1
- 2 * epsilon) + epsilon

# Function to fit model, check diagnostics, and perform Tukey test for beta f
amily
fit_check_tukey_beta <- function(data) {
  # Fit the model using beta family
  fit <- glmmTMB(AdjustedFrequency ~ Treatment + (1|Replicate),
                 family = beta_family(),
                 data = data)

  # Simulate residuals and perform diagnostic checks
  sim_res <- simulateResiduals(fit, plot = TRUE)
  plot(sim_res)
  print(testUniformity(sim_res))
  print(testOutliers(sim_res))
  print(testDispersion(sim_res))

  # Tukey post-hoc test
  tukey <- glht(fit, linfct = mcp(Treatment = "Tukey"))
  tukey_summary <- summary(tukey)

  return(list(summary = summary(fit), tukey_summary = tukey_summary))
}

# Apply to the adjusted frequency data
results_beta <- fit_check_tukey_beta(frequency_data_A)
```


DHARMa residual
QQ plot residuals
KS test: p=
Deviation
ersion test: p=
iation n.s.
test: p= 1
viation n.s.
Observed
Expected
0.0
0.4
0.8
Residual vs. predicted
No significant problems detecte
DHARMa residual
0.00
0.50
1.00
Model predictions (rank transfo

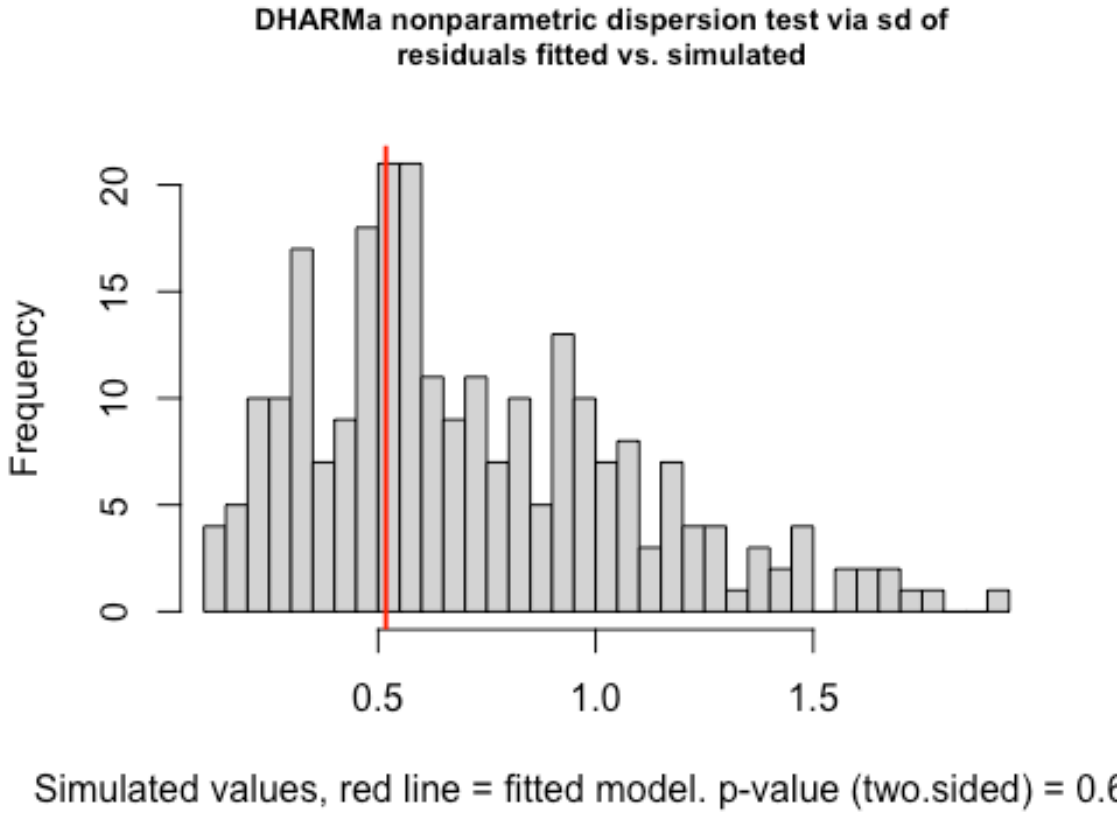

DHARMa residual
QQ plot residuals
KS test: p=
Deviation
ersion test: p=
iation n.s.
test: p= 1
viation n.s.
Observed
Expected
0.0
0.4
0.8
Residual vs. predicted
No significant problems detecte
DHARMa residual
0.00
0.50
1.00
Model predictions (rank transfo




```
##
##  Exact one-sample Kolmogorov-Smirnov test
##
## data:  simulationOutput$scaledResiduals
## D = 0.15029, p-value = 0.8647
## alternative hypothesis: two-sided
```




```
##
##  DHARMa outlier test based on exact binomial test with approximate
##  expectations
##
## data:  sim_res
## outliers at both margin(s) = 0, observations = 14, p-value = 1
## alternative hypothesis: true probability of success is not equal to 0.0079
68127
## 95 percent confidence interval:
##  0.0000000 0.2316358
## sample estimates:
## frequency of outliers (expected: 0.00796812749003984 )
##                                                      0
```




```
##
##  DHARMa nonparametric dispersion test via sd of residuals fitted vs.
##  simulated
##
## data:  simulationOutput
## dispersion = 0.72968, p-value = 0.688
## alternative hypothesis: two.sided
```

```
# Print summaries and Tukey results for beta family
print("Beta Family Model Summary:")
```

```
## [1] "Beta Family Model Summary:"
```

```
print(results_beta$summary)
```

```
##  Family: beta  ( logit )
## Formula:          AdjustedFrequency ~ Treatment + (1 | Replicate)
## Data: data
##
##      AIC      BIC   logLik deviance df.resid
##    -51.4    -48.2     30.7    -61.4        9
##
## Random effects:
##
```

```
## Conditional model:
##  Groups    Name        Variance  Std.Dev.
##  Replicate (Intercept) 8.448e-10 2.907e-05
## Number of obs: 14, groups:  Replicate, 3
##
## Dispersion parameter for beta family (): 3.36
##
## Conditional model:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -1.9848     0.5794  -3.426 0.000614 ***
## TreatmentDynamic  -1.0067     0.6945  -1.450 0.147173
## TreatmentStatic    1.0894     0.6877   1.584 0.113185
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
print("Beta Family Tukey Summary:")
```

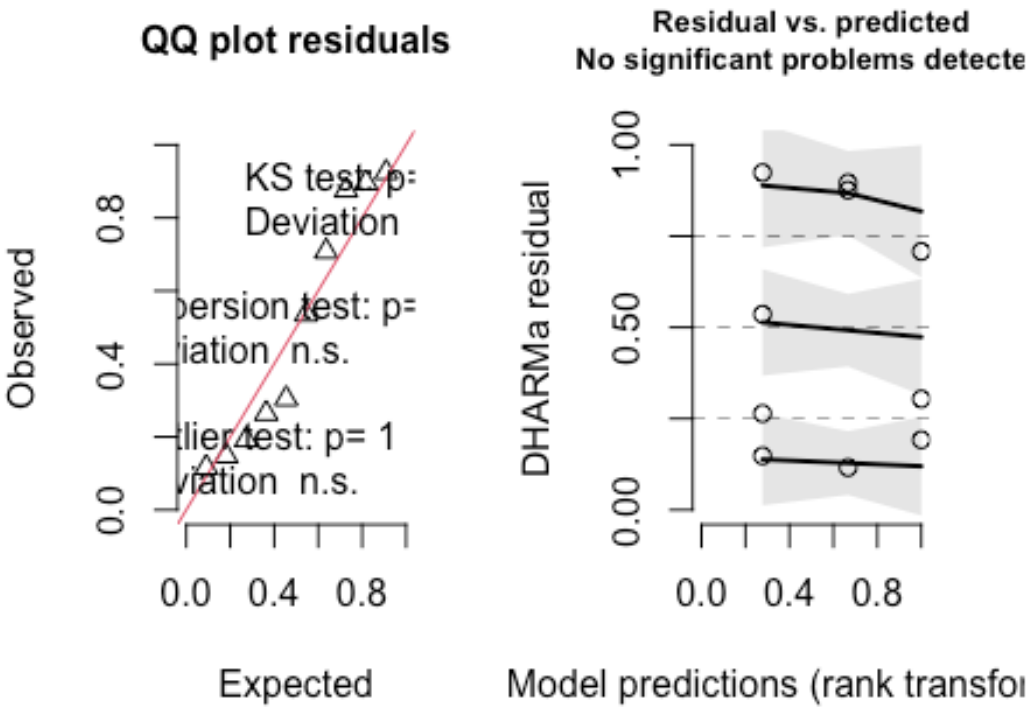
```
## [1] "Beta Family Tukey Summary:"
```

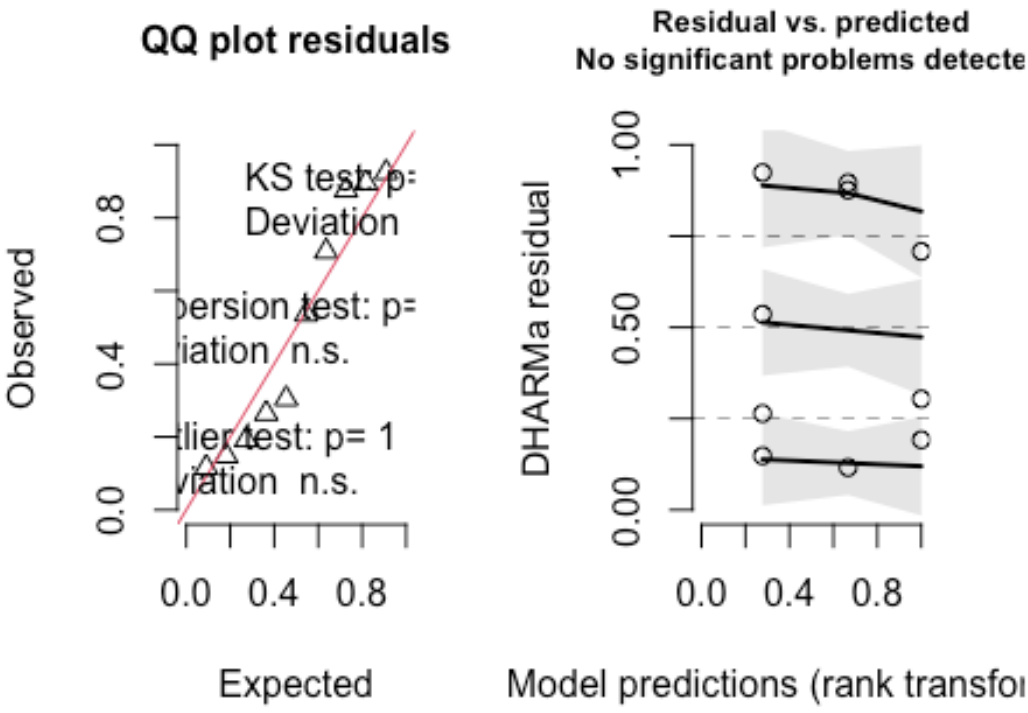
```
print(results_beta$tukey_summary)
```

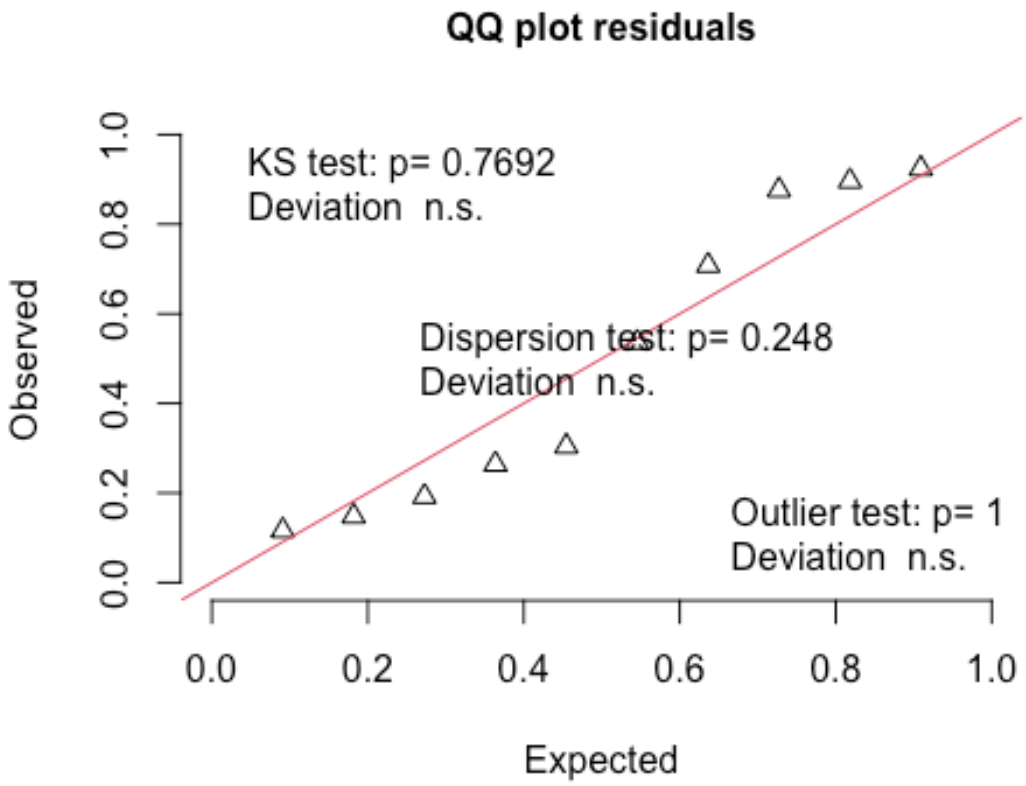
```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: glmmTMB(formula = AdjustedFrequency ~ Treatment + (1 | Replicate),
##     data = data, family = beta_family(), ziformula = ~0, dispformula = ~1)
##
## Linear Hypotheses:
##                        Estimate Std. Error z value Pr(>|z|)
## Dynamic - 2D == 0      -1.0067     0.6945  -1.450   0.3155
## Static - 2D == 0        1.0894     0.6877   1.584   0.2525
## Static - Dynamic == 0   2.0961     0.6780   3.091   0.0057 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

```
#################Selected COCs experiments###################

#data input
data <- read.csv("IVM1.csv")
data$Treatment <- as.factor(data$Treatment)
data$Replicate <- as.factor(data$Replicate)



# Means
means <- aggregate(Maturation_Adj ~ Treatment, data = data, FUN = "mean")

# SDs
```

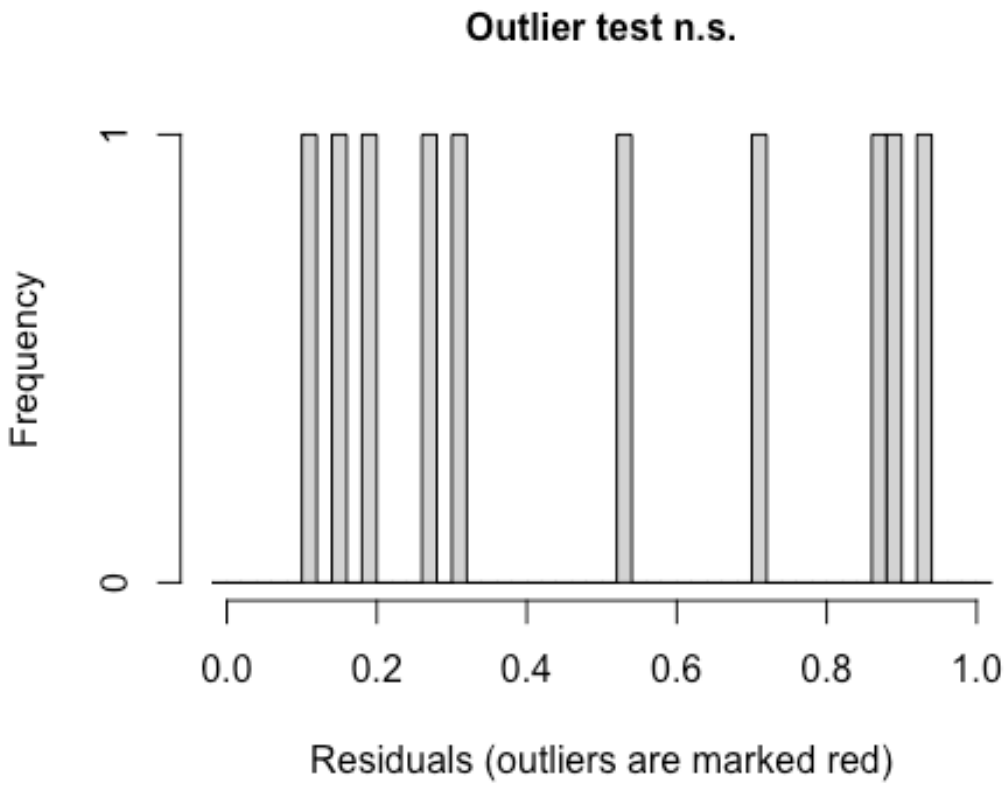
```r
sds <- aggregate(Maturation_Adj ~ Treatment, data = data, FUN = "sd")

# Combine means and SDs
means$SD <- sds$Maturation_Adj

# Update column names
colnames(means)[2:3] <- c("Mean", "SD")  # Update column names

write.csv(means, file = "Summary_Statistics_maturation_selected.csv", row.nam
es = FALSE)


# Stats

# Adjust the data to avoid exact 0 and 1 values
epsilon <- 1e-5
data$AdjustedMaturation <- (data$Maturation / 100) * (1 - 2 * epsilon) + epsi
lon

# Function to fit model, check diagnostics, and perform Tukey test for beta f
amily
fit_check_tukey_beta <- function(data) {
  # Fit the model using beta family
  fit <- glmmTMB(AdjustedMaturation ~ Treatment + (1|Replicate),
                 family = beta_family(),
                 data = data)

  # Simulate residuals and perform diagnostic checks
  sim_res <- simulateResiduals(fit, plot = TRUE)
  plot(sim_res)
  print(testUniformity(sim_res))
  print(testOutliers(sim_res))
  print(testDispersion(sim_res))

  # Tukey post-hoc test
  tukey <- glht(fit, linfct = mcp(Treatment = "Tukey"))
  tukey_summary <- summary(tukey)

  return(list(summary = summary(fit), tukey_summary = tukey_summary))
}

# Adjust the data to avoid exact 0 and 1 values
epsilon <- 1e-5
data$AdjustedMaturation <- (data$Maturation / 100) * (1 - 2 * epsilon) + epsi
lon

# Apply to the whole dataset
results_beta <- fit_check_tukey_beta(data)
```


DHARMa residual
QQ plot residuals
KS test: p=
Deviation
ersion test: p=
iation n.s.
lier test: p= 1
viation n.s.
Observed
Expected
0.0
0.4
0.8
Residual vs. predicted
No significant problems detecte
DHARMa residual
0.00
0.50
1.00
Model predictions (rank transfo

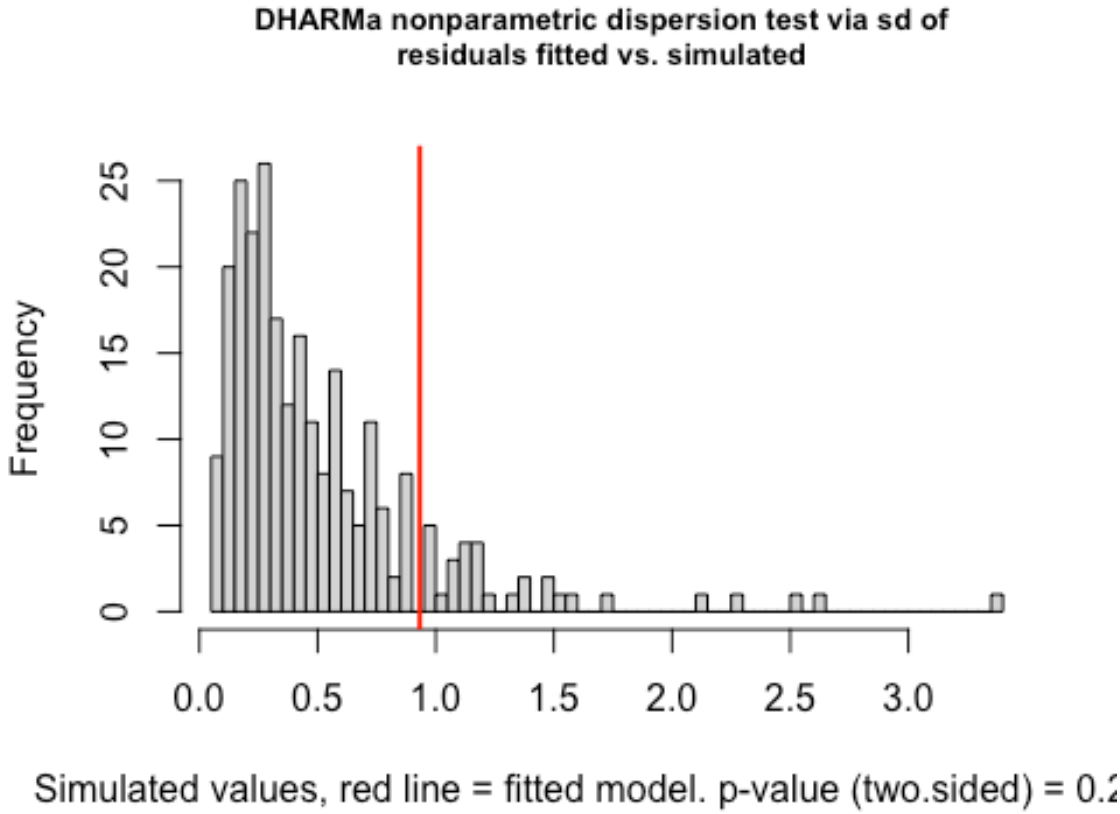

DHARMa residual
QQ plot residuals
KS test: p=
Deviation
ersion test: p=
iation n.s.
lier test: p= 1
viation n.s.
Observed
Expected
0.0
0.4
0.8
Residual vs. predicted
No significant problems detecte
DHARMa residual
0.00
0.50
1.00
Model predictions (rank transfo




```
##
##  Exact one-sample Kolmogorov-Smirnov test
##
## data:  simulationOutput$scaledResiduals
## D = 0.196, p-value = 0.7692
## alternative hypothesis: two-sided
```




```
##
##  DHARMa outlier test based on exact binomial test with approximate
##  expectations
##
## data:  sim_res
## outliers at both margin(s) = 0, observations = 10, p-value = 1
## alternative hypothesis: true probability of success is not equal to 0.0079
68127
## 95 percent confidence interval:
##  0.0000000 0.3084971
## sample estimates:
## frequency of outliers (expected: 0.00796812749003984 )
##                                                      0
```




```
##
##  DHARMa nonparametric dispersion test via sd of residuals fitted vs.
##  simulated
##
## data:  simulationOutput
## dispersion = 1.8316, p-value = 0.248
## alternative hypothesis: two.sided
```

```
# Print summaries and Tukey results for beta family
print("Beta Family Model Summary:")
```

```
## [1] "Beta Family Model Summary:"
```

```
print(results_beta$summary)
```

```
##  Family: beta  ( logit )
## Formula:          AdjustedMaturation ~ Treatment + (1 | Replicate)
## Data: data
##
##      AIC      BIC   logLik deviance df.resid
##    -10.8     -9.3     10.4    -20.8        5
##
## Random effects:
##
```

```
## Conditional model:
##  Groups    Name        Variance Std.Dev.
##  Replicate (Intercept) 0.4274   0.6538
## Number of obs: 10, groups:  Replicate, 2
##
## Dispersion parameter for beta family (): 42.1
##
## Conditional model:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)          0.7125     0.4928   1.446   0.1483
## TreatmentPerfusion   0.8016     0.2800   2.862   0.0042 **
## TreatmentStatic      0.2173     0.2668   0.814   0.4154
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
print("Beta Family Tukey Summary:")
```

```
## [1] "Beta Family Tukey Summary:"
```

```
print(results_beta$tukey_summary)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: glmmTMB(formula = AdjustedMaturation ~ Treatment + (1 | Replicate),
##     data = data, family = beta_family(), ziformula = ~0, dispformula = ~1)
##
## Linear Hypotheses:
##                           Estimate Std. Error z value Pr(>|z|)
## Perfusion - 2D == 0         0.8016     0.2800   2.862   0.0118 *
## Static - 2D == 0            0.2173     0.2668   0.814   0.6927
## Static - Perfusion == 0    -0.5843     0.3152  -1.854   0.1513
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

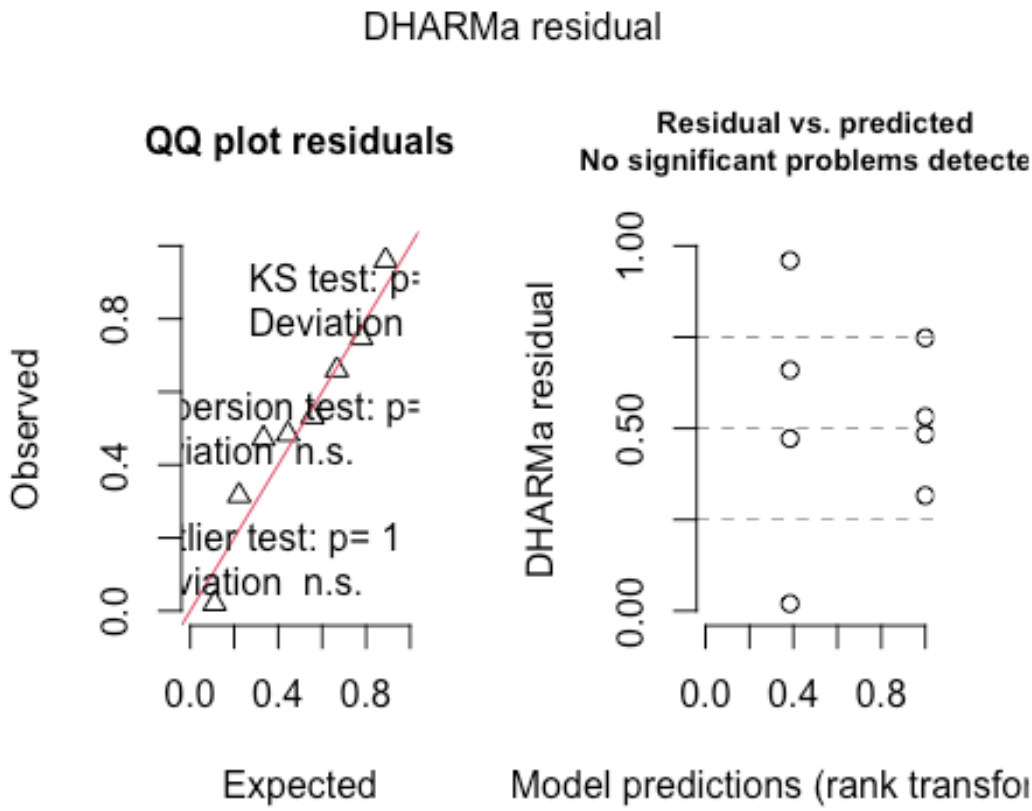
```
# Plot

png(filename="maturation_selected.png",
    width = 4, height = 4, units = "in",
    res = 600)

par(mgp=c(1.5,0.5,0),
    mar = c(4, #bottom
            4, #left
            3, #top
            2),
    cex.main = 0.8,
```

```r
    family = "sans" #right
)


# Define the desired order of X-axis labels
TIME <- c("2D", "Perfusion", "Static")

boxplot(Maturation_Adj ~ Treatment,
        data = data,
        cex.axis = 0.8,
        col = viridis(4, alpha = 0.7),
        lwd = 0.7,
        ylab = "",
        ylim = c(0, 100),
        axes = FALSE,
        frame.plot = TRUE,
        xlab = "")

groups <- 3
numbox <- 1
total <- groups * numbox
xpoints <- seq(median(1:numbox), total, numbox)

axis(2, seq(0, 100, by = 10), las = 1, cex.axis = 0.7)
axis(1, labels = TIME, at = xpoints, las = 1, cex.axis = 0.7)

stripchart(data$Maturation_Adj ~ data$Treatment,
           method = "jitter",
           pch = 19,
           col = "black",
           vertical = TRUE,
           add = TRUE,
           cex = 0.3)

title(ylab = expression("Mature oocytes (%)"), line = 1.7)

dev.off()

## quartz_off_screen
##                 2

################ Heated mini OoTrap experiments###################


# IVM
#data input
data <- read.csv("IVM_mini.csv")
data$Treatment <- as.factor(data$Treatment)
data$Replicate <- as.factor(data$Replicate)
```

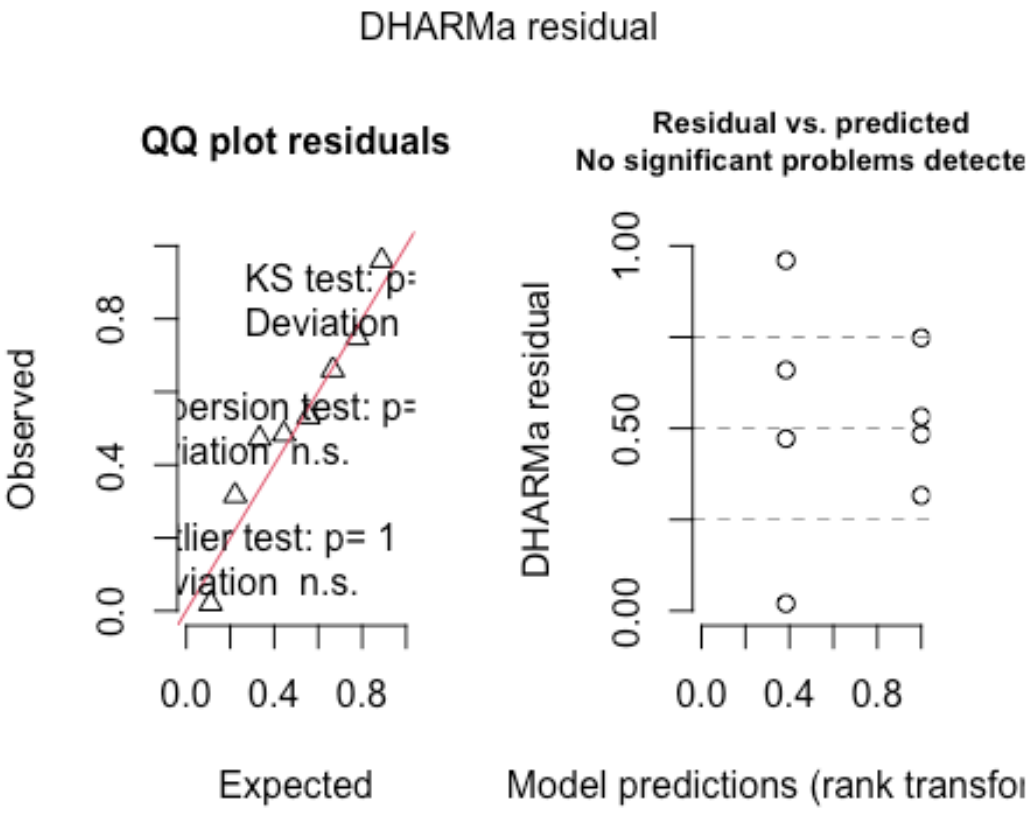
```r
# Means
means <- aggregate(Maturation ~ Treatment, data = data, FUN = "mean")

# SDs
sds <- aggregate(Maturation ~ Treatment, data = data, FUN = "sd")

# Combine means and SDs
means$SD <- sds$Maturation

# Update column names
colnames(means)[2:3] <- c("Mean", "SD")  # Update column names

write.csv(means, file = "Summary_Statistics_maturation_selected.csv", row.nam
es = FALSE)


# Stats

# Adjust the data to avoid exact 0 and 1 values
epsilon <- 1e-5
data$AdjustedMaturation <- (data$Maturation / 100) * (1 - 2 * epsilon) + epsi
lon

# Function to fit model, check diagnostics, and perform Tukey test for beta f
amily
fit_check_tukey_beta <- function(data) {
  # Fit the model using beta family
  fit <- glmmTMB(AdjustedMaturation ~ Treatment + (1|Replicate),
                 family = beta_family(),
                 data = data)

  # Simulate residuals and perform diagnostic checks
  sim_res <- simulateResiduals(fit, plot = TRUE)
  plot(sim_res)
  print(testUniformity(sim_res))
  print(testOutliers(sim_res))
  print(testDispersion(sim_res))

  # Tukey post-hoc test
  tukey <- glht(fit, linfct = mcp(Treatment = "Tukey"))
  tukey_summary <- summary(tukey)

  return(list(summary = summary(fit), tukey_summary = tukey_summary))
}

# Adjust the data to avoid exact 0 and 1 values
epsilon <- 1e-5
```

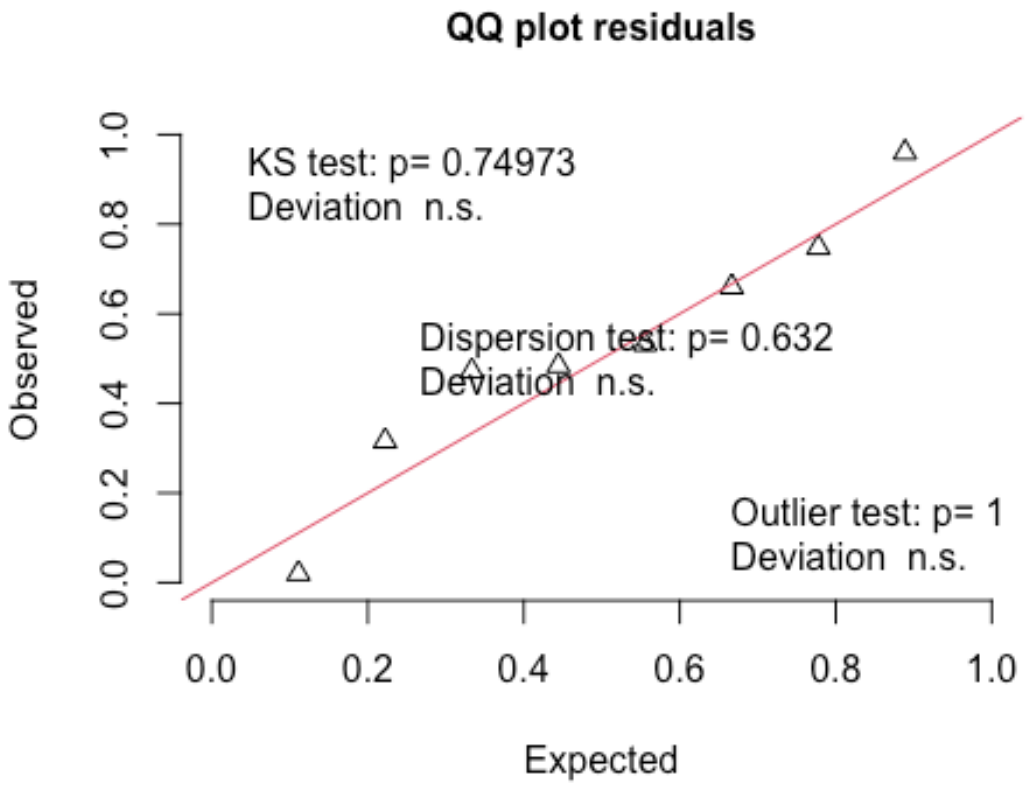
```r
data$AdjustedMaturation <- (data$Maturation / 100) * (1 - 2 * epsilon) + epsi
lon

# Apply to the whole dataset
results_beta <- fit_check_tukey_beta(data)
```

```
## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): bas
is dimension, k, increased to minimum possible

## Unable to calculate quantile regression for quantile 0.25. Possibly to few
(unique) data points / predictions. Will be ommited in plots and significance
calculations.

## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): bas
is dimension, k, increased to minimum possible

## Unable to calculate quantile regression for quantile 0.5. Possibly to few
(unique) data points / predictions. Will be ommited in plots and significance
calculations.

## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): bas
is dimension, k, increased to minimum possible

## Unable to calculate quantile regression for quantile 0.75. Possibly to few
(unique) data points / predictions. Will be ommited in plots and significance
calculations.
```

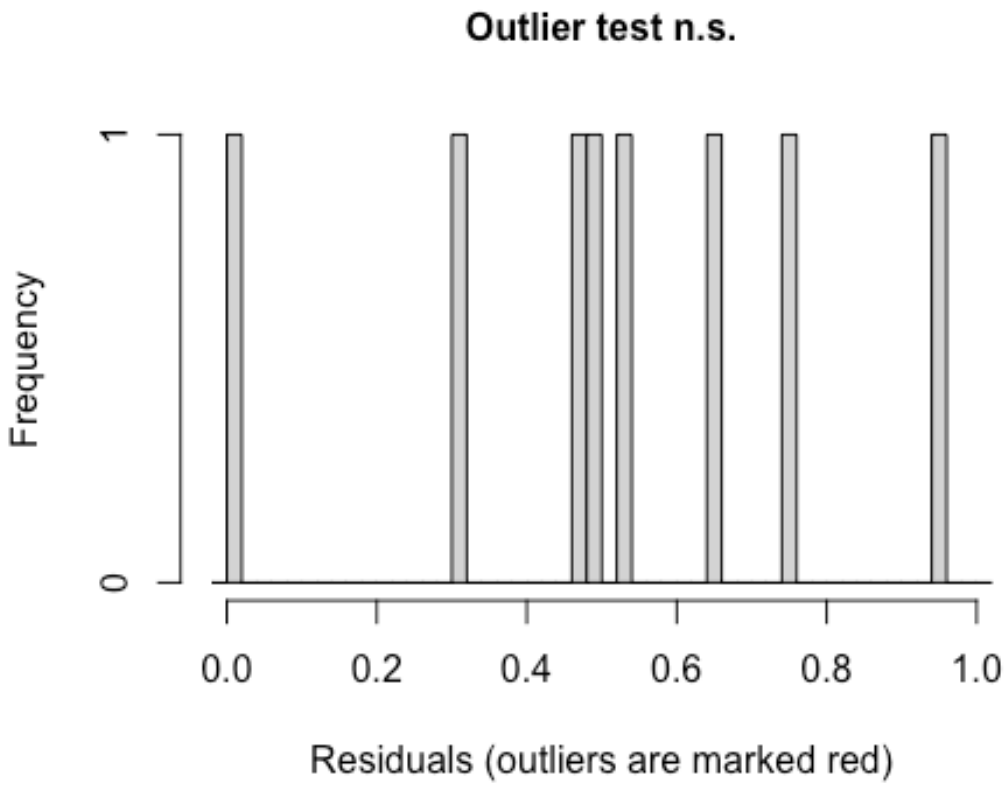



```
## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): bas
is dimension, k, increased to minimum possible

## Unable to calculate quantile regression for quantile 0.25. Possibly to few
(unique) data points / predictions. Will be ommited in plots and significance
calculations.

## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): bas
is dimension, k, increased to minimum possible

## Unable to calculate quantile regression for quantile 0.5. Possibly to few
(unique) data points / predictions. Will be ommited in plots and significance
calculations.

## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): bas
is dimension, k, increased to minimum possible

## Unable to calculate quantile regression for quantile 0.75. Possibly to few
(unique) data points / predictions. Will be ommited in plots and significance
calculations.
```

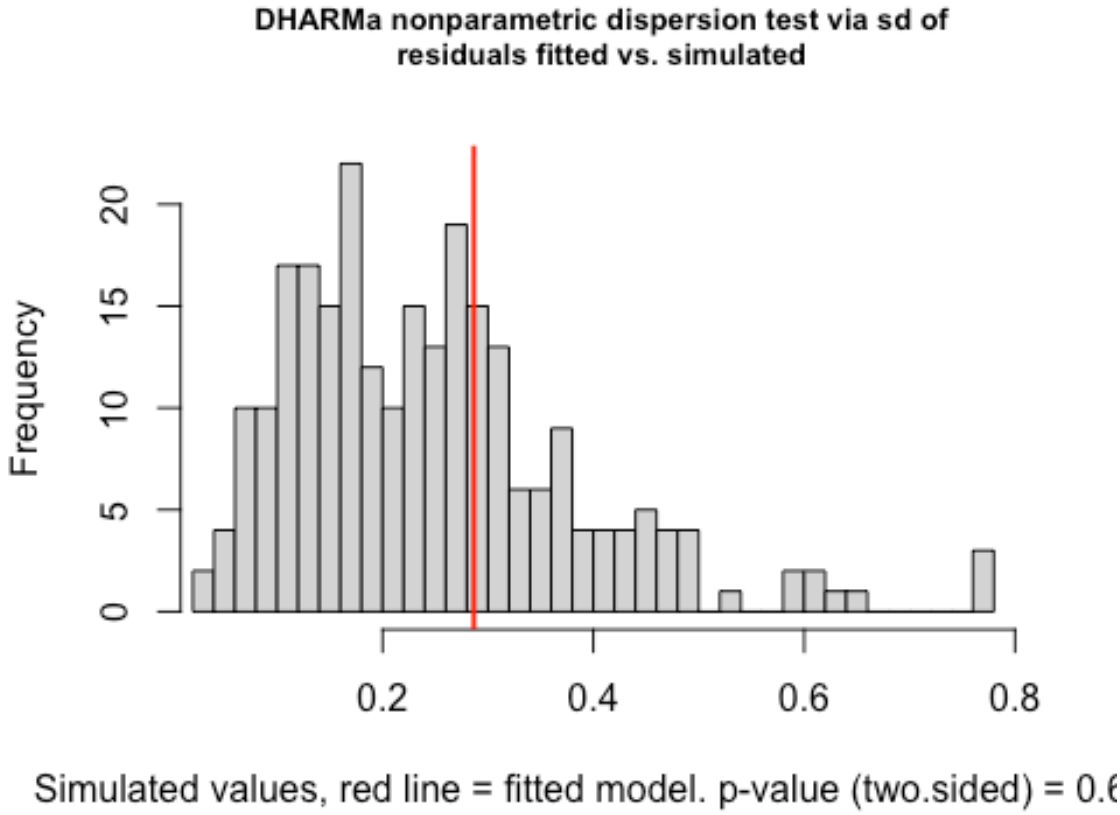

DHARMa residual
QQ plot residuals
KS test: p=
Deviation
ersion test: p=
iation n.s.
lier test: p= 1
viation n.s.
Observed
Expected
0.0
0.4
0.8
Residual vs. predicted
No significant problems detecte
DHARMa residual
0.00
0.50
1.00
Model predictions (rank transfo




```
##
##  Exact one-sample Kolmogorov-Smirnov test
##
## data:  simulationOutput$scaledResiduals
## D = 0.222, p-value = 0.7497
## alternative hypothesis: two-sided
```




```
##
##  DHARMa outlier test based on exact binomial test with approximate
##  expectations
##
## data:  sim_res
## outliers at both margin(s) = 0, observations = 8, p-value = 1
## alternative hypothesis: true probability of success is not equal to 0.0079
68127
## 95 percent confidence interval:
##  0.0000000 0.3694166
## sample estimates:
## frequency of outliers (expected: 0.00796812749003984 )
##                                                      0
```




```
##
##  DHARMa nonparametric dispersion test via sd of residuals fitted vs.
##  simulated
##
## data:  simulationOutput
## dispersion = 1.1639, p-value = 0.632
## alternative hypothesis: two.sided
```

```
# Print summaries and Tukey results for beta family
print("Beta Family Model Summary:")
```

```
## [1] "Beta Family Model Summary:"
```

```
print(results_beta$summary)
```

```
##  Family: beta  ( logit )
## Formula:          AdjustedMaturation ~ Treatment + (1 | Replicate)
## Data: data
##
##      AIC      BIC   logLik deviance df.resid
##    -13.7    -13.4     10.9    -21.7        4
##
## Random effects:
##
```

```
## Conditional model:
##  Groups    Name        Variance Std.Dev.
##  Replicate (Intercept) 0.01806  0.1344
## Number of obs: 8, groups:  Replicate, 4
##
## Dispersion parameter for beta family (): 78.1
##
## Conditional model:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -0.3423     0.1323  -2.588  0.00966 **
## TreatmentPerfusion   0.8965     0.1638   5.474  4.4e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
print("Beta Family Tukey Summary:")
```

```
## [1] "Beta Family Tukey Summary:"
```

```
print(results_beta$tukey_summary)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: glmmTMB(formula = AdjustedMaturation ~ Treatment + (1 | Replicate),
##     data = data, family = beta_family(), ziformula = ~0, dispformula = ~1)
##
## Linear Hypotheses:
##                      Estimate Std. Error z value Pr(>|z|)
## Perfusion - 2D == 0   0.8965     0.1638   5.474  4.4e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

```
# Plot

png(filename="maturation_mini.png",
    width = 4, height = 4, units = "in",
    res = 600)

par(mgp=c(1.5,0.5,0),
    mar = c(4, #bottom
            4, #left
            3, #top
            2),
    cex.main = 0.8,
    family = "sans" #right
)
```

```r
# Define the desired order of X-axis labels

TIME <- c("2D", "Perfusion")

boxplot(Maturation ~ Treatment,
        data = data,
        cex.axis = 0.8,
        col = viridis(4, alpha = 0.7),
        lwd = 0.7,
        ylab = "",
        ylim = c(0, 100),
        axes = FALSE,
        frame.plot = TRUE,
        xlab = "")

groups <- 2
numbox <- 1
total <- groups * numbox
xpoints <- seq(median(1:numbox), total, numbox)

axis(2, seq(0, 100, by = 10), las = 1, cex.axis = 0.7)
axis(1, labels = TIME, at = xpoints, las = 1, cex.axis = 0.7)

stripchart(data$Maturation ~ data$Treatment,
           method = "jitter",
           pch = 19,
           col = "black",
           vertical = TRUE,
           add = TRUE,
           cex = 0.3)

title(ylab = expression("Mature oocytes (%)"), line = 1.7)

dev.off()

## quartz_off_screen
##                 2

# Plot abnormal spindle mini

#data input
data <- read.csv("Mini_spindlefreq.csv")
data$Treatment <- as.factor(data$Treatment)
data$Replicate <- as.factor(data$Replicate)
data$Device <- as.factor(data$Device)


# Open the PNG device
```

```r
png(filename="spindle_mini.png",
    width = 4, height = 4, units = "in",
    res = 600)

# Set graphical parameters
par(mgp=c(1.5,0.5,0),
    mar = c(4, 4, 3, 2), # bottom, left, top, right
    cex.main = 0.8,
    family = "sans")

# Filter the data to include only Spindle A
frequency_data_A <- data[data$Spindle == 'A',]


# Define the desired order of X-axis labels
TIME <- c("2D", "Perfusion")

boxplot(Frequency ~ Treatment,
        data = frequency_data_A,
        cex.axis = 0.8,
        col = viridis(4, alpha = 0.7),
        lwd = 0.7,
        ylab = "",
        ylim = c(0, 100),
        axes = FALSE,
        frame.plot = TRUE,
        xlab = "")

groups <- 2
numbox <- 1
total <- groups * numbox
xpoints <- seq(median(1:numbox), total, numbox)

axis(2, seq(0, 100, by = 10), las = 1, cex.axis = 0.7)
axis(1, labels = TIME, at = xpoints, las = 1, cex.axis = 0.7)

stripchart(frequency_data_A$Frequency ~ frequency_data_A$Treatment,
           method = "jitter",
           pch = 19,
           col = "black",
           vertical = TRUE,
           add = TRUE,
           cex = 0.3)

title(ylab = expression(atop("Abnormal Spindle", "and/or Chromosome alignment
(%)")), line = 1.5)
```

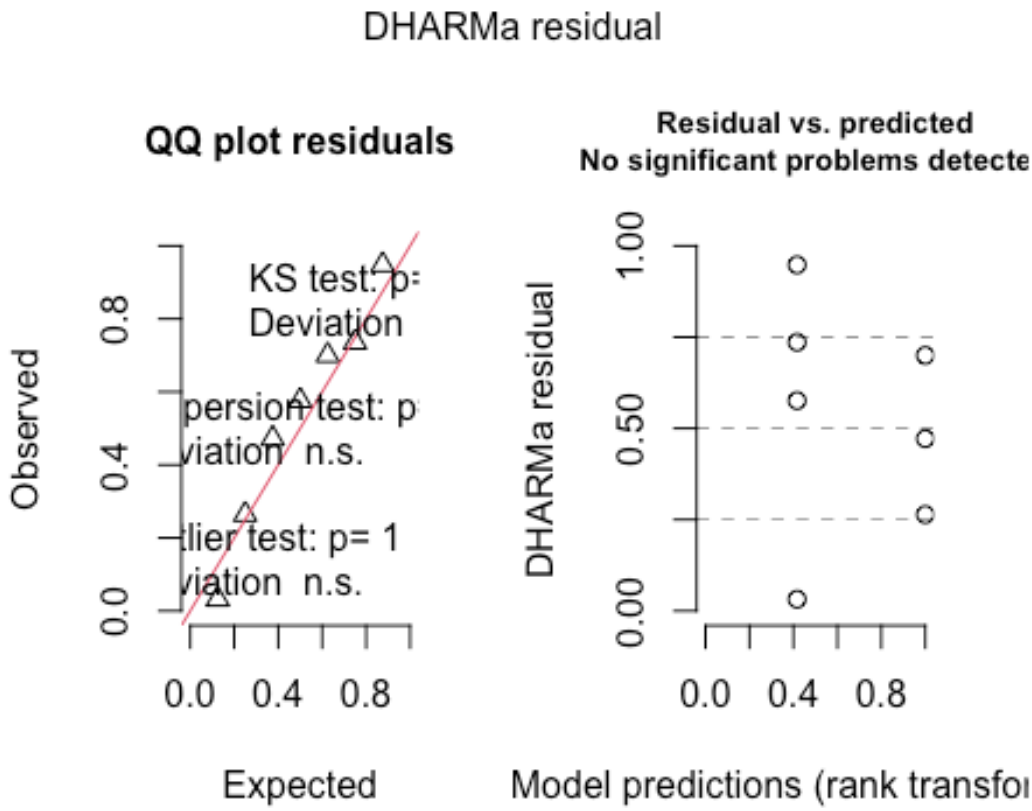
```r
dev.off()
```

```
## quartz_off_screen
##                 2
```

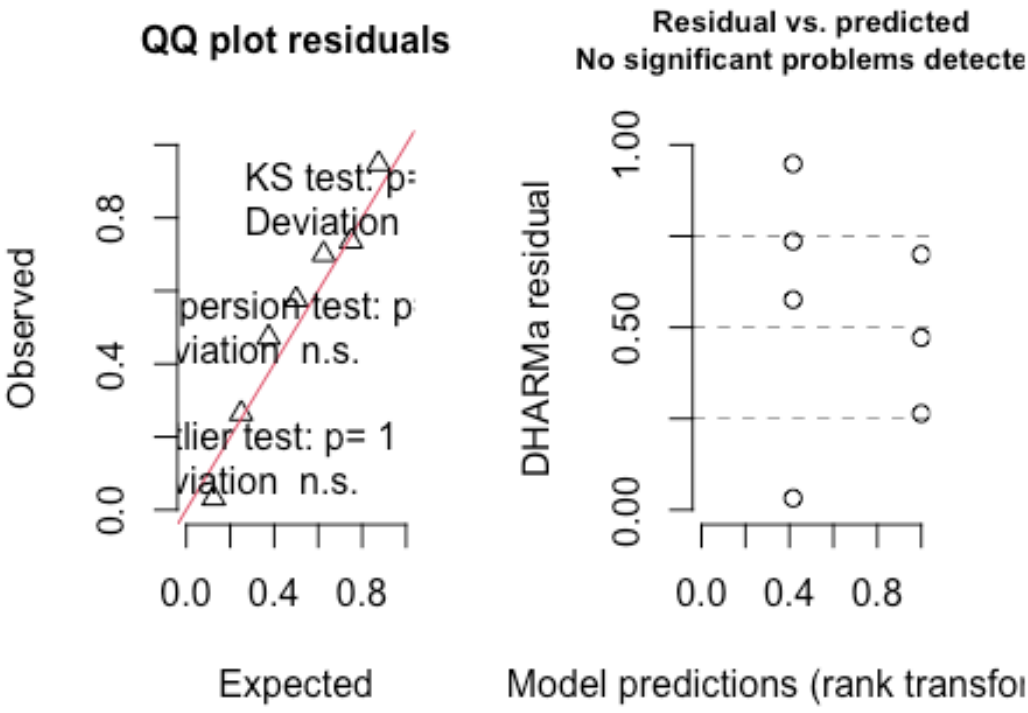
```r
# Stats Spindle


# Means
means <- aggregate(Frequency ~ Treatment, data = frequency_data_A, FUN = "mea
n")

# SDs
sds <- aggregate(Frequency ~ Treatment, data = frequency_data_A, FUN = "sd")

# Combine means and SDs
means$SD <- sds$Frequency

# Update column names
colnames(means)[2:3] <- c("Mean", "SD")  # Update column names

write.csv(means, file = "Summary_Statistics_spindle.csv", row.names = FALSE)


# Adjust the data to avoid exact 0 and 1 values
epsilon <- 1e-5
frequency_data_A$AdjustedFrequency <- (frequency_data_A$Frequency / 100) * (1
- 2 * epsilon) + epsilon

# Function to fit model, check diagnostics, and perform Tukey test for beta f
amily
fit_check_tukey_beta <- function(data) {
  # Fit the model using beta family
  fit <- glmmTMB(AdjustedFrequency ~ Treatment + (1|Replicate),
                 family = beta_family(),
                 data = data)

  # Simulate residuals and perform diagnostic checks
  sim_res <- simulateResiduals(fit, plot = TRUE)
  plot(sim_res)
  print(testUniformity(sim_res))
  print(testOutliers(sim_res))
  print(testDispersion(sim_res))

  # Tukey post-hoc test
  tukey <- glht(fit, linfct = mcp(Treatment = "Tukey"))
  tukey_summary <- summary(tukey)
```

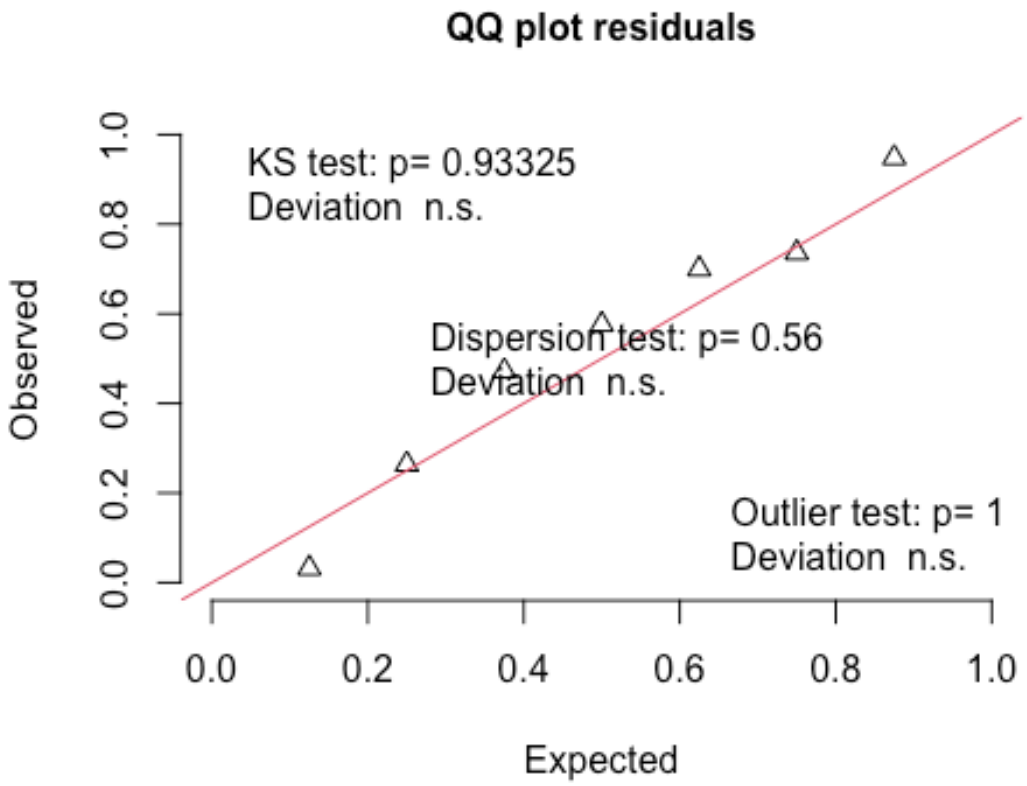
```r
  return(list(summary = summary(fit), tukey_summary = tukey_summary))
}

# Apply to the adjusted frequency data
results_beta <- fit_check_tukey_beta(frequency_data_A)
```

```
## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): bas
is dimension, k, increased to minimum possible

## Unable to calculate quantile regression for quantile 0.25. Possibly to few
(unique) data points / predictions. Will be ommited in plots and significance
calculations.

## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): bas
is dimension, k, increased to minimum possible

## Unable to calculate quantile regression for quantile 0.5. Possibly to few
(unique) data points / predictions. Will be ommited in plots and significance
calculations.

## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): bas
is dimension, k, increased to minimum possible

## Unable to calculate quantile regression for quantile 0.75. Possibly to few
(unique) data points / predictions. Will be ommited in plots and significance
calculations.
```

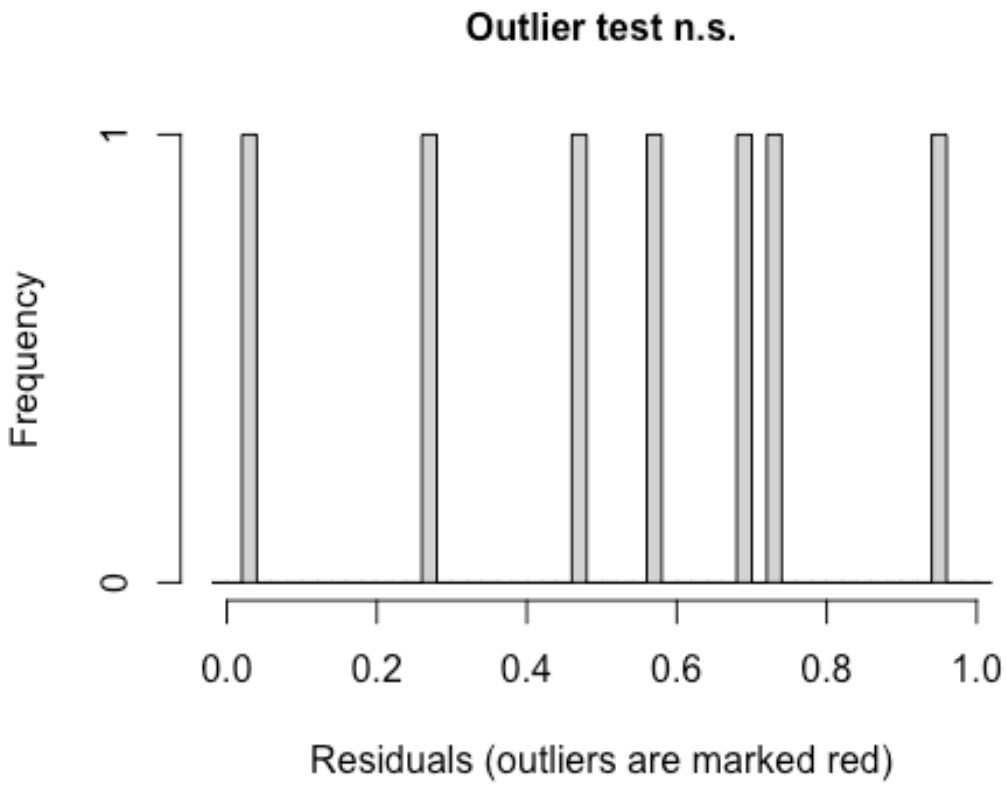



```
## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): bas
is dimension, k, increased to minimum possible

## Unable to calculate quantile regression for quantile 0.25. Possibly to few
(unique) data points / predictions. Will be ommited in plots and significance
calculations.

## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): bas
is dimension, k, increased to minimum possible

## Unable to calculate quantile regression for quantile 0.5. Possibly to few
(unique) data points / predictions. Will be ommited in plots and significance
calculations.

## Warning in smooth.construct.tp.smooth.spec(object, dk$data, dk$knots): bas
is dimension, k, increased to minimum possible

## Unable to calculate quantile regression for quantile 0.75. Possibly to few
(unique) data points / predictions. Will be ommited in plots and significance
calculations.
```

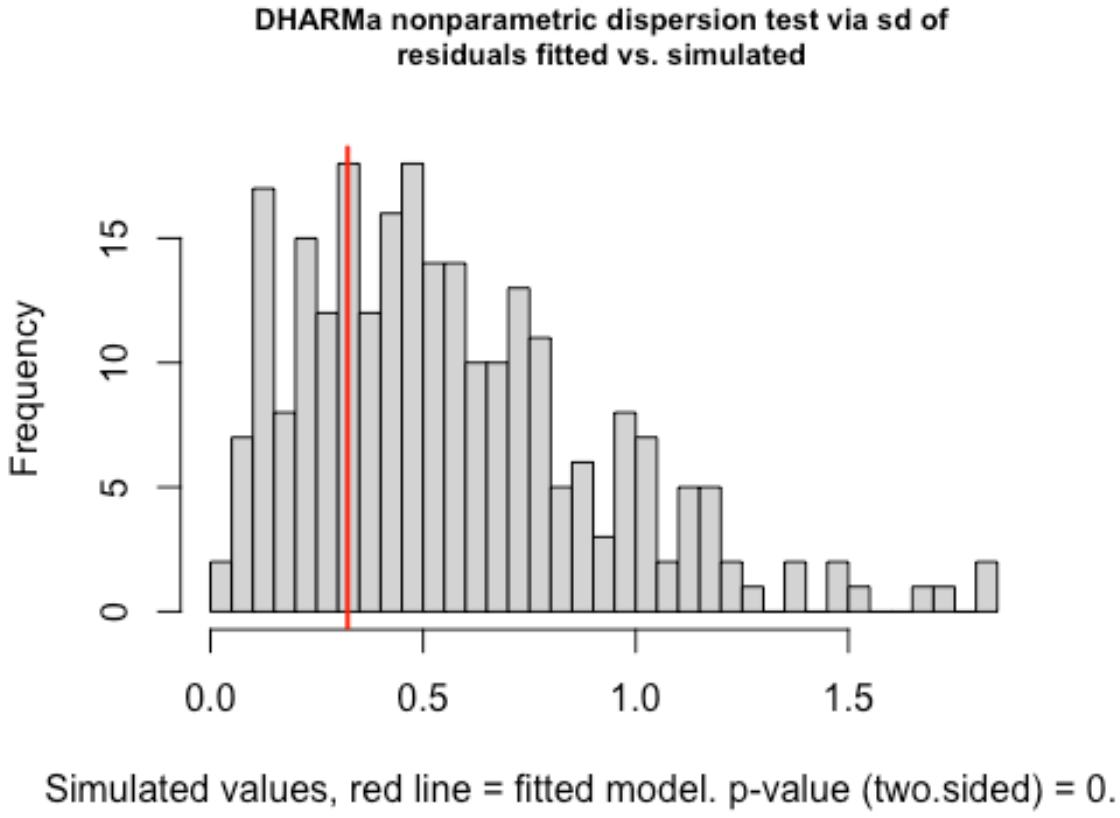

DHARMa residual
QQ plot residuals
KS test: p=
Deviation
persion test: p
viation n.s.
lier test: p= 1
viation n.s.
Observed
Expected
0.0
0.4
0.8
Residual vs. predicted
No significant problems detecte
DHARMa residual
0.00
0.50
1.00
Model predictions (rank transfo




```
##
##  Exact one-sample Kolmogorov-Smirnov test
##
## data:  simulationOutput$scaledResiduals
## D = 0.18629, p-value = 0.9332
## alternative hypothesis: two-sided
```




```
##
##  DHARMa outlier test based on exact binomial test with approximate
##  expectations
##
## data:  sim_res
## outliers at both margin(s) = 0, observations = 7, p-value = 1
## alternative hypothesis: true probability of success is not equal to 0.0079
68127
## 95 percent confidence interval:
##  0.0000000 0.4096164
## sample estimates:
## frequency of outliers (expected: 0.00796812749003984 )
##                                                      0
```




```
##
##  DHARMa nonparametric dispersion test via sd of residuals fitted vs.
##  simulated
##
## data:  simulationOutput
## dispersion = 0.56818, p-value = 0.56
## alternative hypothesis: two.sided
```

```
# Print summaries and Tukey results for beta family
print("Beta Family Model Summary:")
```

```
## [1] "Beta Family Model Summary:"
```

```
print(results_beta$summary)
```

```
##  Family: beta  ( logit )
## Formula:          AdjustedFrequency ~ Treatment + (1 | Replicate)
## Data: data
##
##      AIC      BIC   logLik deviance df.resid
##     -9.6     -9.8      8.8    -17.6        3
##
## Random effects:
##
```

```
## Conditional model:
##  Groups    Name        Variance Std.Dev.
##  Replicate (Intercept) 0.4563   0.6755
## Number of obs: 7, groups:  Replicate, 4
##
## Dispersion parameter for beta family (): 7.26
##
## Conditional model:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -0.4587     0.5955  -0.770  0.44114
## TreatmentDynamic  -1.8022     0.6954  -2.592  0.00955 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
print("Beta Family Tukey Summary:")
```

```
## [1] "Beta Family Tukey Summary:"
```

```
print(results_beta$tukey_summary)
```

```
##
##   Simultaneous Tests for General Linear Hypotheses
##
## Multiple Comparisons of Means: Tukey Contrasts
##
##
## Fit: glmmTMB(formula = AdjustedFrequency ~ Treatment + (1 | Replicate),
##     data = data, family = beta_family(), ziformula = ~0, dispformula = ~1)
##
## Linear Hypotheses:
##                   Estimate Std. Error z value Pr(>|z|)
## Dynamic - 2D == 0  -1.8022     0.6954  -2.592  0.00955 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## (Adjusted p values reported -- single-step method)
```

```
####### Difuusion COMSOL Simulations #######

###BSA


#data input
data <- read.csv("BSA_diff2.csv")




png(filename="BSA_diff.png",
    width = 6, height = 4, units = "in",
```

```
    res = 600)



# Ensure 'Time' is numeric and remove NA values
data <- data %>%
  mutate(Time = as.numeric(Time)) %>%
  filter(!is.na(Time))

# Calculate mean values for each combination of Time and Flow
mean_data <- data %>%
  group_by(Time, Flow) %>%
  summarize(mean_concentration = mean(Concentration), .groups = 'drop')

# Define specific colors manually
flow_colors <- c("0" = "#35b779", "20" = "#31688e")

# Create the plot using ggplot
ggplot(data = data, aes(x = Time, y = Concentration, color = factor(Flow))) +
  geom_jitter(size = 0.2, alpha = 0.05, width = 2) +  # Jitter the points
  geom_line(data = mean_data, aes(x = Time, y = mean_concentration, group = F
low), size = 1) +  # Mean line
  geom_point(data = mean_data, aes(x = Time, y = mean_concentration), shape =
21, size = 3, fill = "white") +  # Add mean points
  geom_text(data = mean_data, aes(x = Time, y = mean_concentration, label = r
ound(mean_concentration, 2)),
            vjust = -1, hjust = 1, size = 3, color = "red") +  # Add mean val
ues next to mean points
  labs(x = "Time (min)", y = expression(paste("Concentration (mol ", m^{-3},
")")), color = "Flow") +
  theme_minimal() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.border = element_rect(color = "black", fill = NA),
        axis.title = element_text(size = 14),
        axis.text = element_text(size = 12)) +
  scale_x_continuous(breaks = seq(0, max(data$Time, na.rm = TRUE), by = 5)) +
# Set interval of 5 for x-axis
  scale_color_manual(values = flow_colors)  # Manually set colors for Flow va
riable

## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## ℹ Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.

dev.off()
```

```
## quartz_off_screen
##                 2

# Stats



# Ensure 'Flow' is a factor
data$Flow <- factor(data$Flow)

# Define time points
time_points <- c(0, 5, 10, 15, 20)

# Function to perform GLM and Tukey's HSD for each time point
perform_glm_tukey <- function(time_point) {
  # Subset data for the specific time point
  subset_data <- data %>% filter(Time == time_point)

  # Fit the generalized linear model
  glm_model <- glm(Concentration ~ Flow, data = subset_data, family = gaussia
n())

  # Calculate estimated marginal means
  emm <- emmeans(glm_model, ~ Flow)

  # Perform Tukey's HSD test for pairwise comparisons
  pairwise_results <- pairs(emm, adjust = "tukey")

  # Print the results
  cat("Results for Time Point:", time_point, "\n")
  print(summary(pairwise_results))
  cat("\n\n")
}

# Perform the analysis for each time point
lapply(time_points, perform_glm_tukey)

## Results for Time Point: 0
##  contrast       estimate       SE    df t.ratio p.value
##  Flow0 - Flow20 4.81e-05 2.48e-05 47926   1.939  0.0525
##
##
##
## Results for Time Point: 5
##  contrast       estimate      SE    df t.ratio p.value
##  Flow0 - Flow20  -0.0808 0.00137 48017 -59.111  <.0001
##
##
##
## Results for Time Point: 10
```

```
##  contrast       estimate       SE    df  t.ratio p.value
##  Flow0 - Flow20  -0.0314 0.000312 38386 -100.514  <.0001
##
##
##
## Results for Time Point: 15
##  contrast       estimate       SE    df  t.ratio p.value
##  Flow0 - Flow20  -0.0165 0.000151 37978 -109.325  <.0001
##
##
##
## Results for Time Point: 20
##  contrast       estimate       SE    df  t.ratio p.value
##  Flow0 - Flow20  -0.0224 0.000145 22686 -154.647  <.0001

## [[1]]
## NULL
##
## [[2]]
## NULL
##
## [[3]]
## NULL
##
## [[4]]
## NULL
##
## [[5]]
## NULL
```

###H2O2

```
#data input
data <- read.csv("H2O2_diff.csv")


png(filename="H2O2_diff.png",
    width = 6, height = 4, units = "in",
    res = 600)


# Ensure 'Time' is numeric and remove NA values
data <- data %>%
  mutate(Time = as.numeric(Time)) %>%
  filter(!is.na(Time))

# Calculate mean values for each combination of Time and Flow
mean_data <- data %>%
  group_by(Time, Flow) %>%
```

```r
  summarize(mean_concentration = mean(Concentration), .groups = 'drop')

# Define specific colors manually
flow_colors <- c("0" = "#35b779", "20" = "#31688e")

# Create the plot using ggplot
ggplot(data = data, aes(x = Time, y = Concentration, color = factor(Flow))) +
  geom_jitter(size = 0.2, alpha = 0.05, width = 2) +  # Jitter the points
  geom_line(data = mean_data, aes(x = Time, y = mean_concentration, group = F
low), size = 1) +  # Mean line
  geom_point(data = mean_data, aes(x = Time, y = mean_concentration), shape =
21, size = 3, fill = "white") +  # Add mean points
  geom_text(data = mean_data, aes(x = Time, y = mean_concentration, label = r
ound(mean_concentration, 2)),
            vjust = -1, hjust = 1, size = 3, color = "red") +  # Add mean val
ues next to mean points
  labs(x = "Time (min)", y = expression(paste("Concentration (mol ", m^{-3},
")")), color = "Flow") +
  theme_minimal() +
  theme(panel.grid.major = element_blank(),
        panel.grid.minor = element_blank(),
        panel.border = element_rect(color = "black", fill = NA),
        axis.title = element_text(size = 14),
        axis.text = element_text(size = 12)) +
  scale_x_continuous(breaks = seq(0, max(data$Time, na.rm = TRUE), by = 5)) +
# Set interval of 5 for x-axis
  scale_color_manual(values = flow_colors)  # Manually set colors for Flow va
riable

dev.off()

## quartz_off_screen
##                 2

# Stats

# Ensure 'Flow' is a factor
data$Flow <- factor(data$Flow)

# Define time points
time_points <- c(0, 5, 10, 15, 20)

# Function to perform GLM and Tukey's HSD for each time point
perform_glm_tukey <- function(time_point) {
  # Subset data for the specific time point
  subset_data <- data %>% filter(Time == time_point)

  # Fit the generalized linear model
  glm_model <- glm(Concentration ~ Flow, data = subset_data, family = gaussia
n())
```

```r
  # Calculate estimated marginal means
  emm <- emmeans(glm_model, ~ Flow)

  # Perform Tukey's HSD test for pairwise comparisons
  pairwise_results <- pairs(emm, adjust = "tukey")

  # Print the results
  cat("Results for Time Point:", time_point, "\n")
  print(summary(pairwise_results))
  cat("\n\n")
}

# Perform the analysis for each time point
lapply(time_points, perform_glm_tukey)
```

```
## Results for Time Point: 0
##  contrast        estimate       SE    df t.ratio p.value
##  Flow0 - Flow20 -1.21e-05 3.35e-05 50412  -0.362  0.7171
##
##
##
## Results for Time Point: 5
##  contrast       estimate       SE    df t.ratio p.value
##  Flow0 - Flow20 1.75e-05 0.000571 50412   0.031  0.9755
##
##
##
## Results for Time Point: 10
##  contrast        estimate       SE    df t.ratio p.value
##  Flow0 - Flow20 -8.94e-06 0.000191 50412  -0.047  0.9627
##
##
##
## Results for Time Point: 15
##  contrast       estimate       SE    df t.ratio p.value
##  Flow0 - Flow20 1.33e-05 9.53e-05 49433   0.140  0.8886
##
##
##
## Results for Time Point: 20
##  contrast       estimate      SE    df t.ratio p.value
##  Flow0 - Flow20 3.76e-06 4.7e-05 40075   0.080  0.9362

## [[1]]
## NULL
##
## [[2]]
## NULL
##
```

```
## [[3]]
## NULL
##
## [[4]]
## NULL
##
## [[5]]
## NULL
```