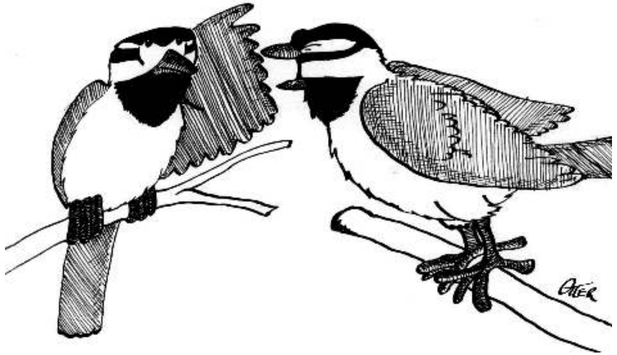
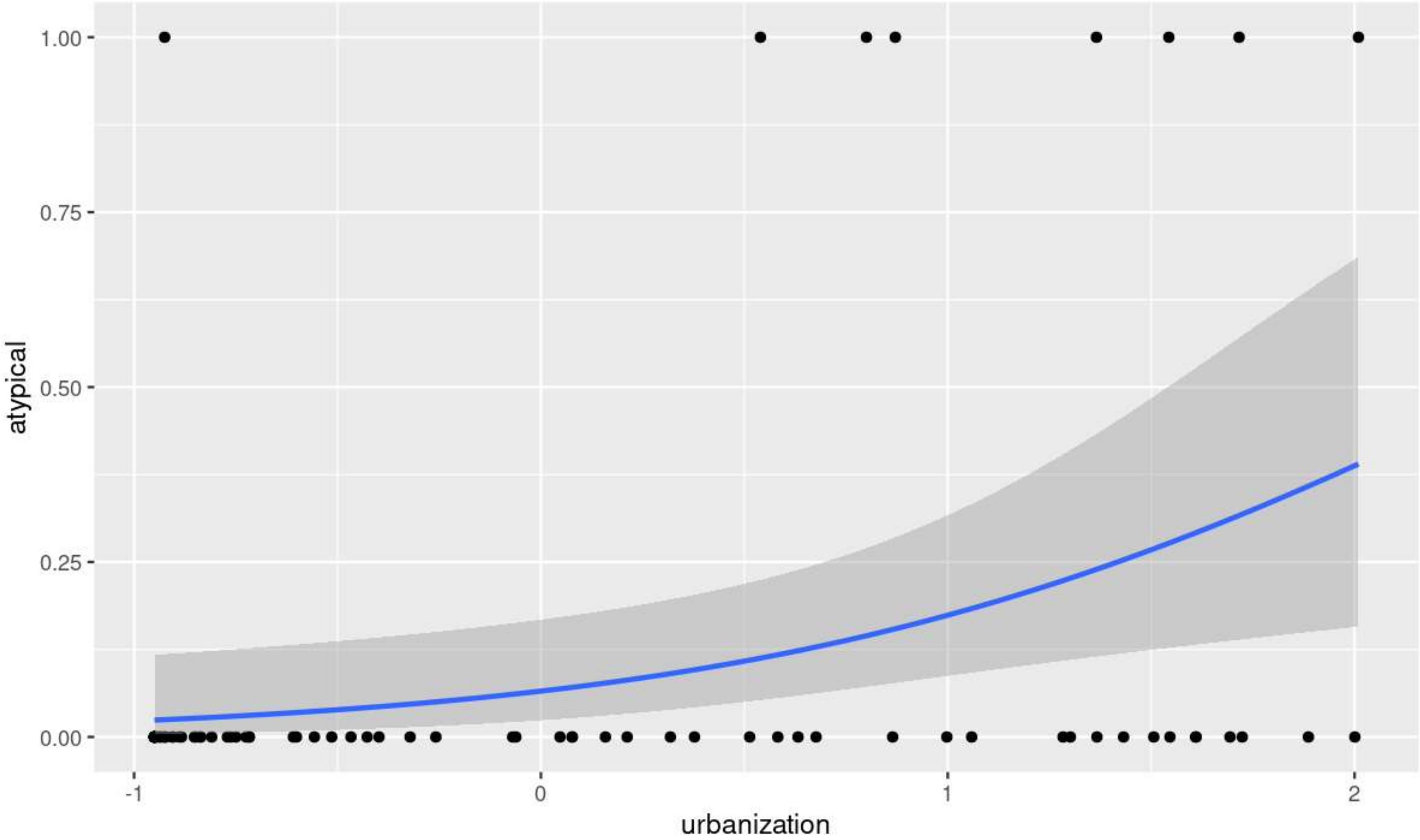
NRI 7350



Even more stats...

Generalized linear models,
Other advanced models,
Non-parametric stats



Getting started (again)

Open RStudio

Open your NRI project

Open a **new** script for today:

File > New File > R Script

Make sure to load packages at the top:

```
library(tidyverse)
library(DHARMa)
```

Reference Material

This lecture covers A HUGE subject area

It is not comprehensive

It is a place to start, with references to guide you later on

Diagnostics for complex models

Introducing DHARMa

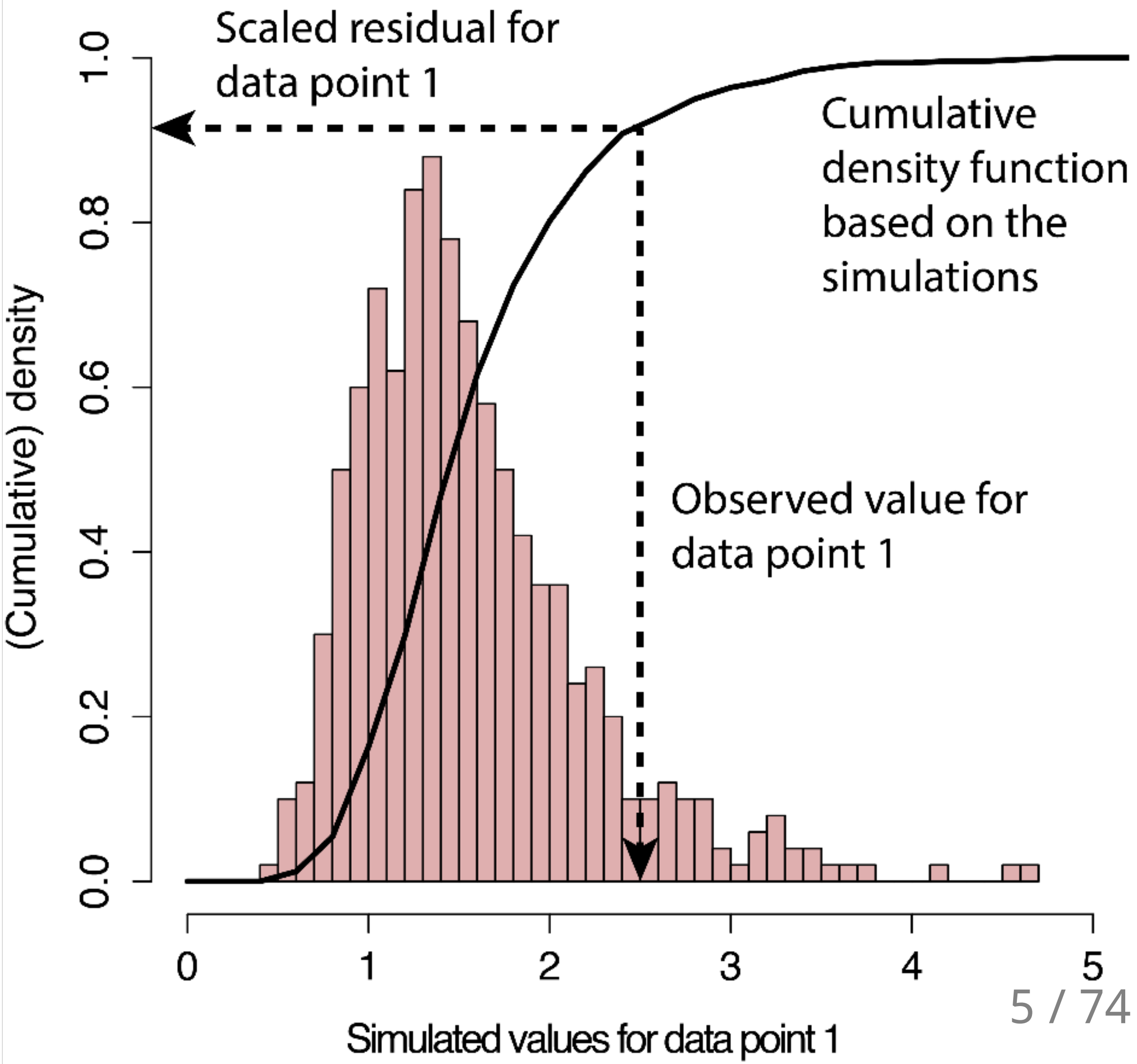
DHARMa Tutorial

(Many great examples of model checking)

DHARMa Package

Simulated Residuals

- **If** your data perfectly matched your model, what would the values look like?
  - Pink histogram values show repeatedly simulated values

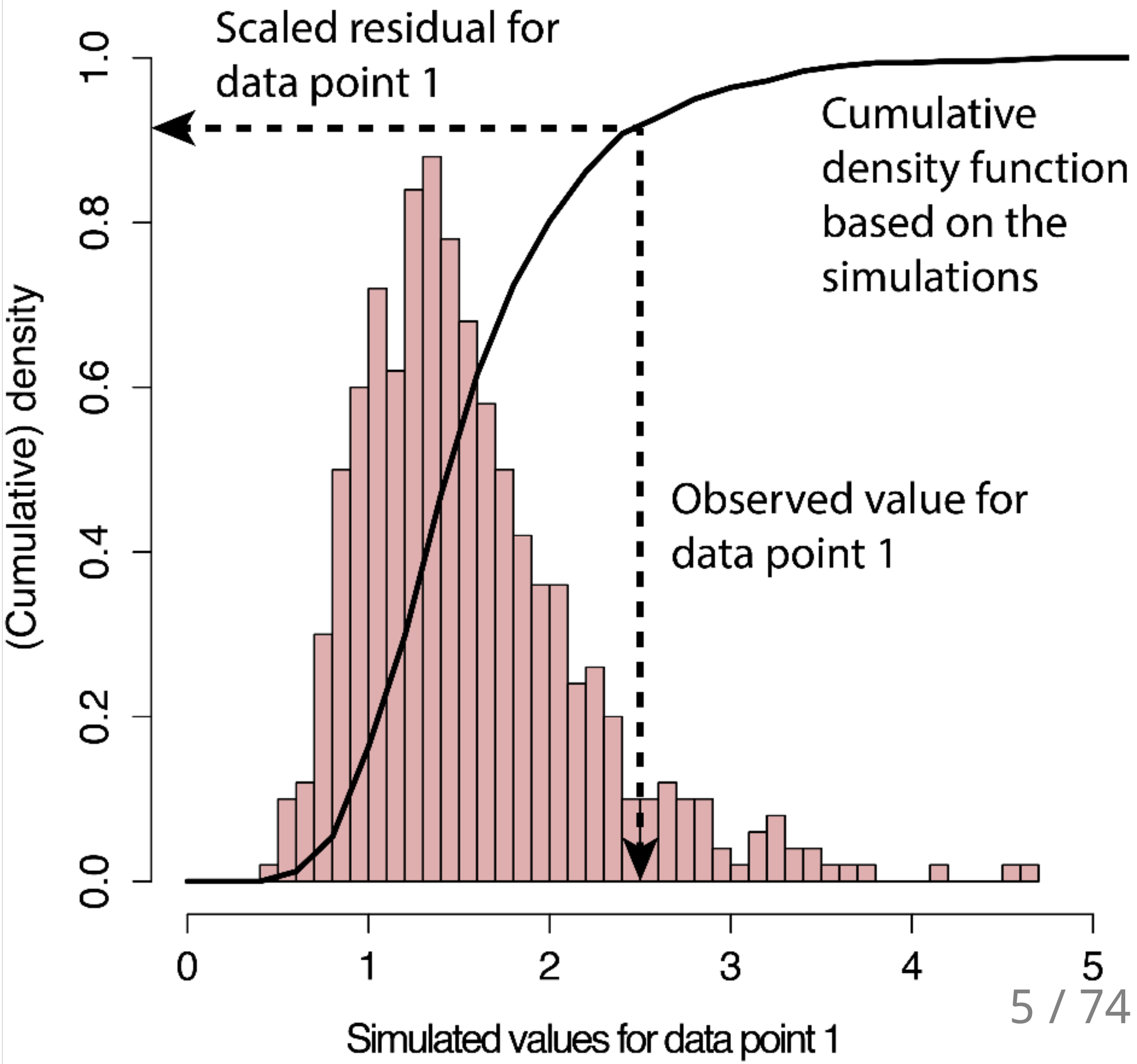


DHARMa Package

Simulated Residuals

- **If** your data perfectly matched your model, what would the values look like?
  - Pink histogram values show repeatedly simulated values

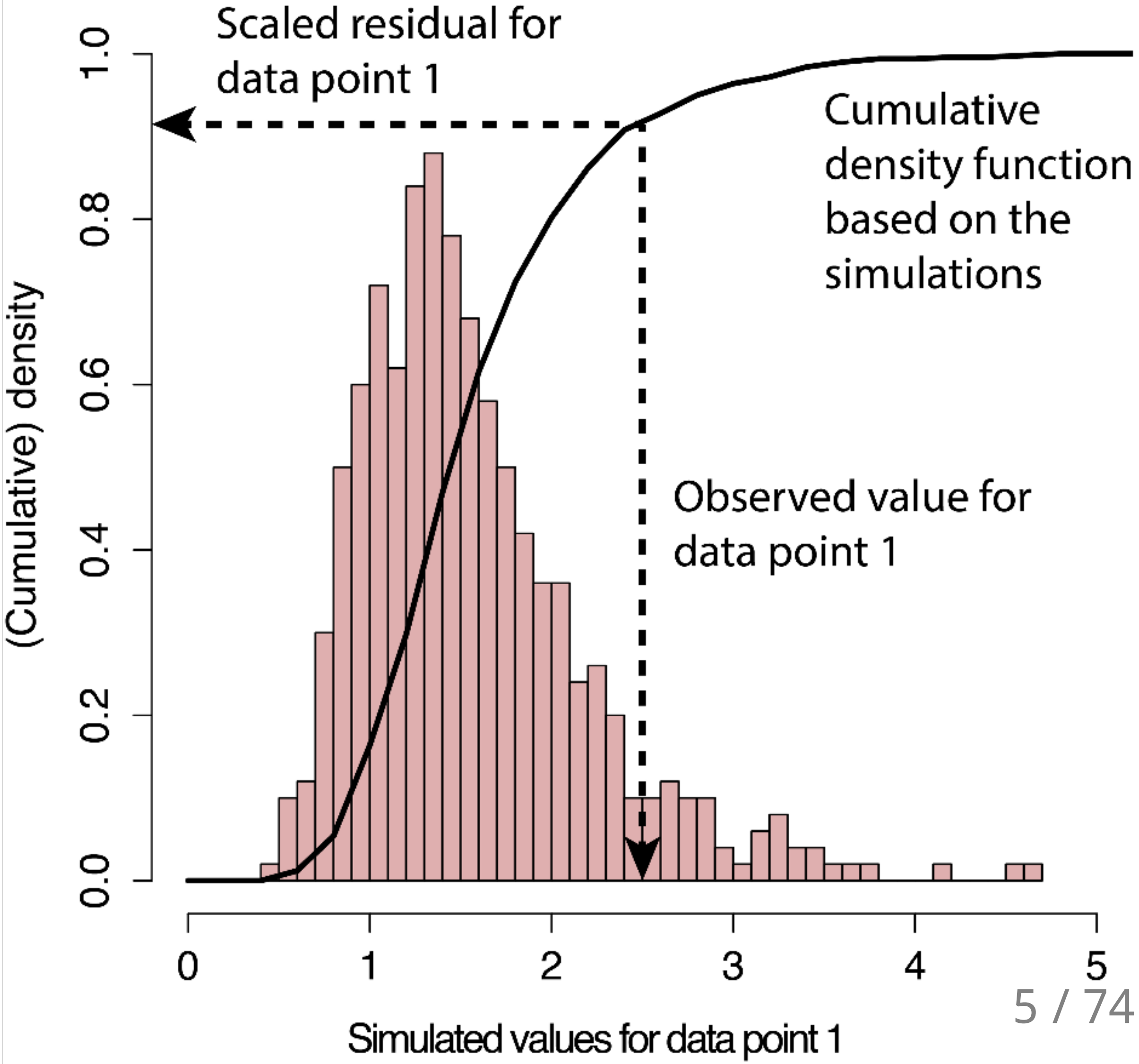
- How do your *actual* values compare?
  - Black arrow on x-axis shows *actual* value
  - Black arrow on y-axis shows residual: how *actual* value compares to distribution of simulated values



DHARMa Package

Simulated Residuals

- **If** your data perfectly matched your model, what would the values look like?
    - Pink histogram values show repeatedly simulated values
- How do your *actual* values compare?
    - Black arrow on x-axis shows *actual* value
    - Black arrow on y-axis shows residual: how *actual* value compares to distribution of simulated values
- Residuals are scaled (0 to 1)
    - If data fits model perfectly, expect all = 0.5
    - Good fit *always* = flat/uniform distribution



DHARMa Package

If data fits the model

- Residuals follow a flat (uniform) distribution (no matter what model!)

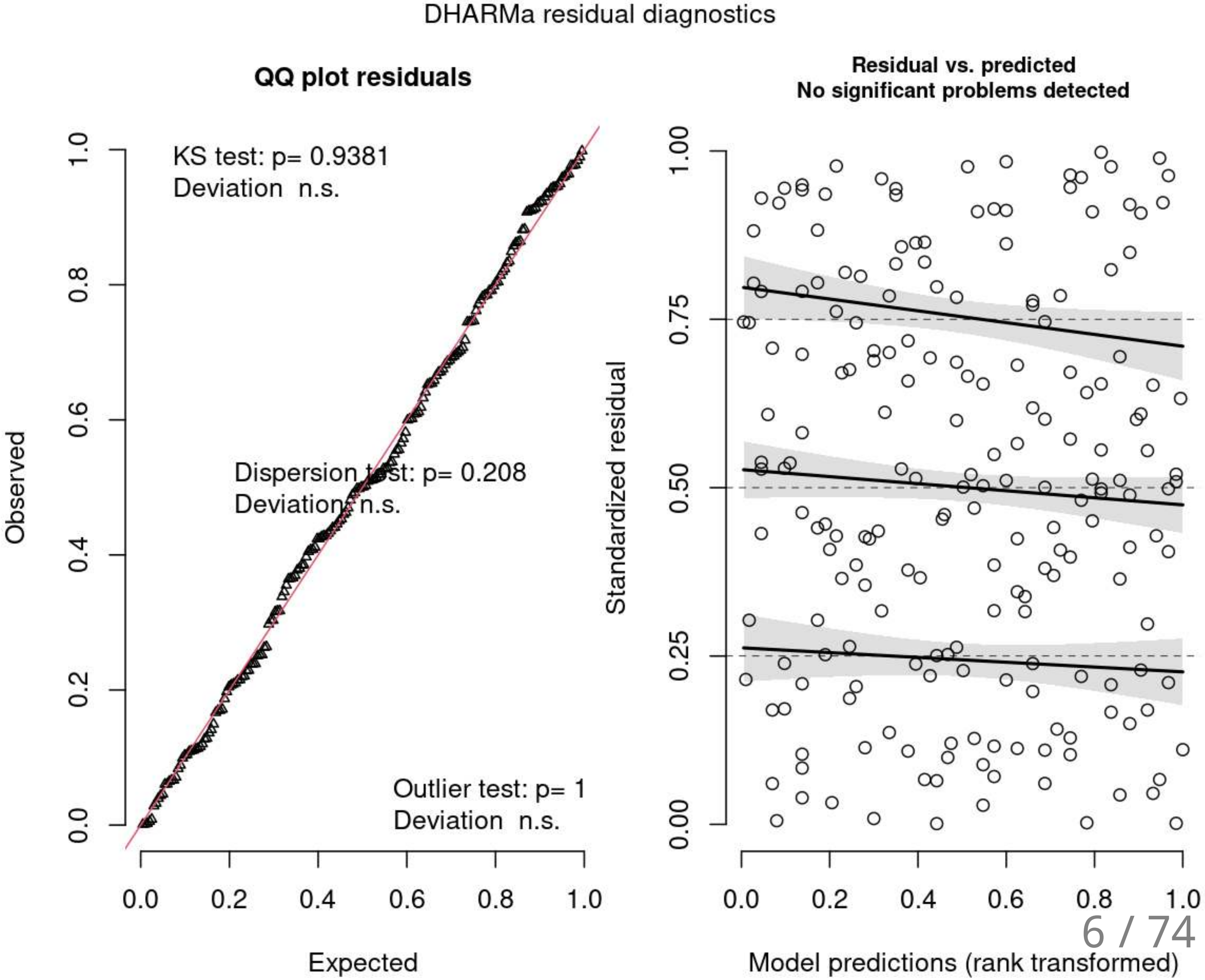
DHARMa Package

If data fits the model

- Residuals follow a flat (uniform) distribution (no matter what model!)
- Expect: Straight line on QQ plot of **uniform** distribution

  (similar to QQ Normal plot)
- Expect: No patterns between residuals and model predictions

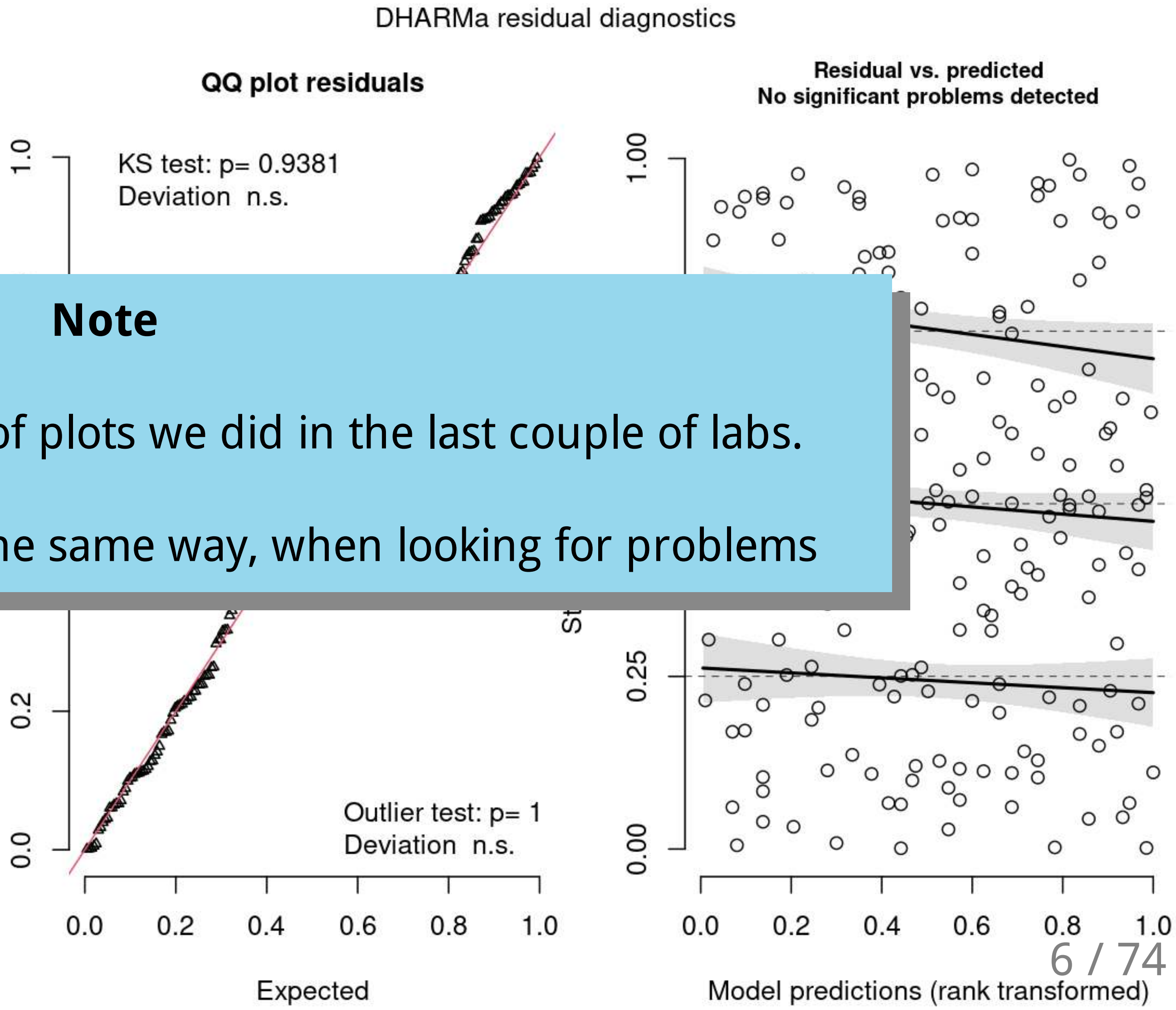
  (similar to heteroscedasticity plot, resid vs. fitted)



DHARMa Package

If data fits the model

- Residuals follow a flat (uniform) distribution (no matter what model!)
- Expect:
  uniform
  (similar to
- Expect: No patterns between residuals and model predictions
  (similar to heteroscedasticity plot, resid vs. fitted)



Note

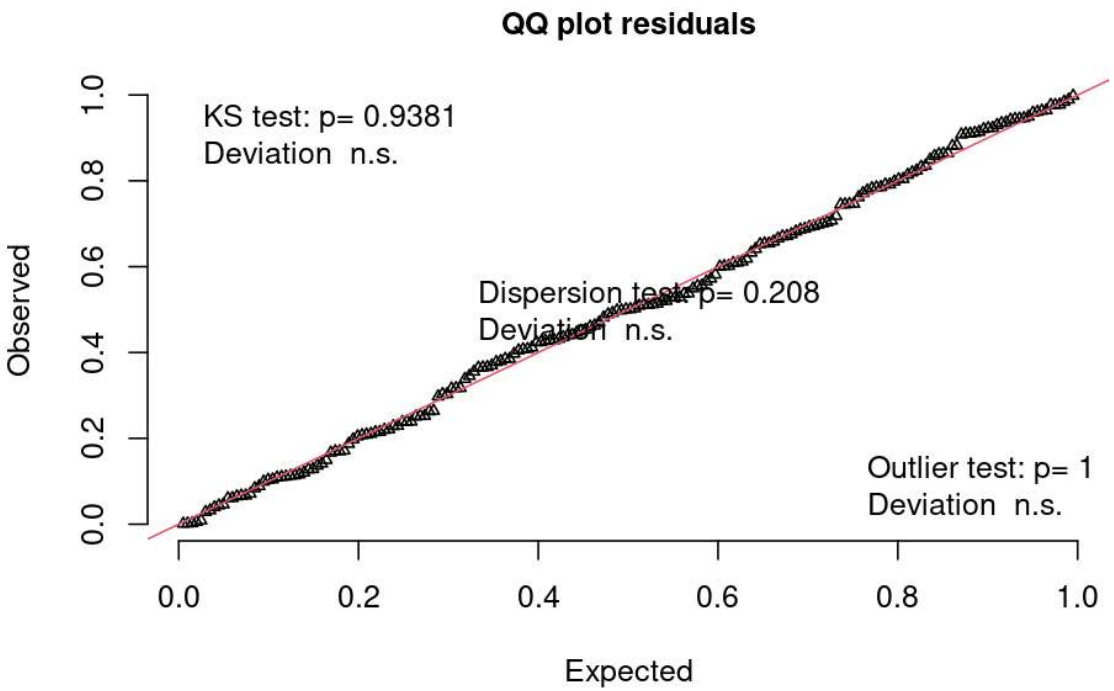
These aren't the same kinds of plots we did in the last couple of labs.

BUT you can interpret them the same way, when looking for problems

DHARMa Package

QQ Plot

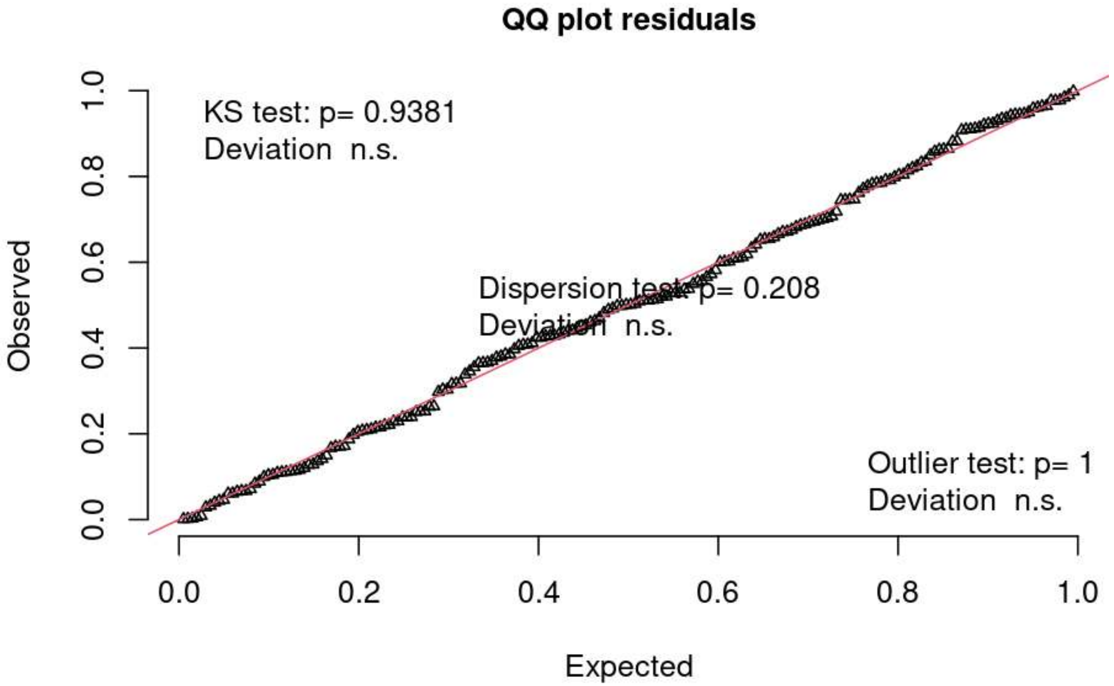
- Tests **Uniformity** with Kolmogorov-Smirnov (KS) test
  - (do the residuals match a Uniform distribution?)
- Tests for **Over/Underdispersion** with Dispersion Test
- Tests for more **Outliers** than expected with Outlier test



DHARMa Package

QQ Plot

- Tests **Uniformity** with Kolmogorov-Smirnov (KS) test
  - (do the residuals match a Uniform distribution?)
- Tests for **Over/Underdispersion** with Dispersion Test
- Tests for more **Outliers** than expected with Outlier test



Residuals vs. Predicted

- Check distribution of residuals (visually and with quantiles)
- Dotted lines show expected quantiles
- Black lines show simulated quantiles (want straight lines)
- Outliers would show up as red stars

DHARMa Package

Usage

```r
library(DHARMa)

m <- lm(body_mass_g ~ flipper_length_mm, data = penguins)
r <- simulateResiduals(m, n = 1000, plot = TRUE)
```

- **`simulateResiduals()`** function from
- Use **`plot = TRUE`** to produce diagnostic plots to see if simulated match expectation
- **`n = 1000`** isn't strictly necessary but runs more simulations to produce more stable results

DHARMa Package

Usage

```r
library(DHARMa)

m <- lm(body_mass_g ~ flipper_length_mm, data = penguins)
r <- simulateResiduals(m, n = 1000, plot = TRUE)
```

- **`simulateResiduals()`** function from
- Use **`plot = TRUE`** to produce diagnostic plots to see if simulated match expectation
- **`n = 1000`** isn't strictly necessary but runs more simulations to produce more stable results

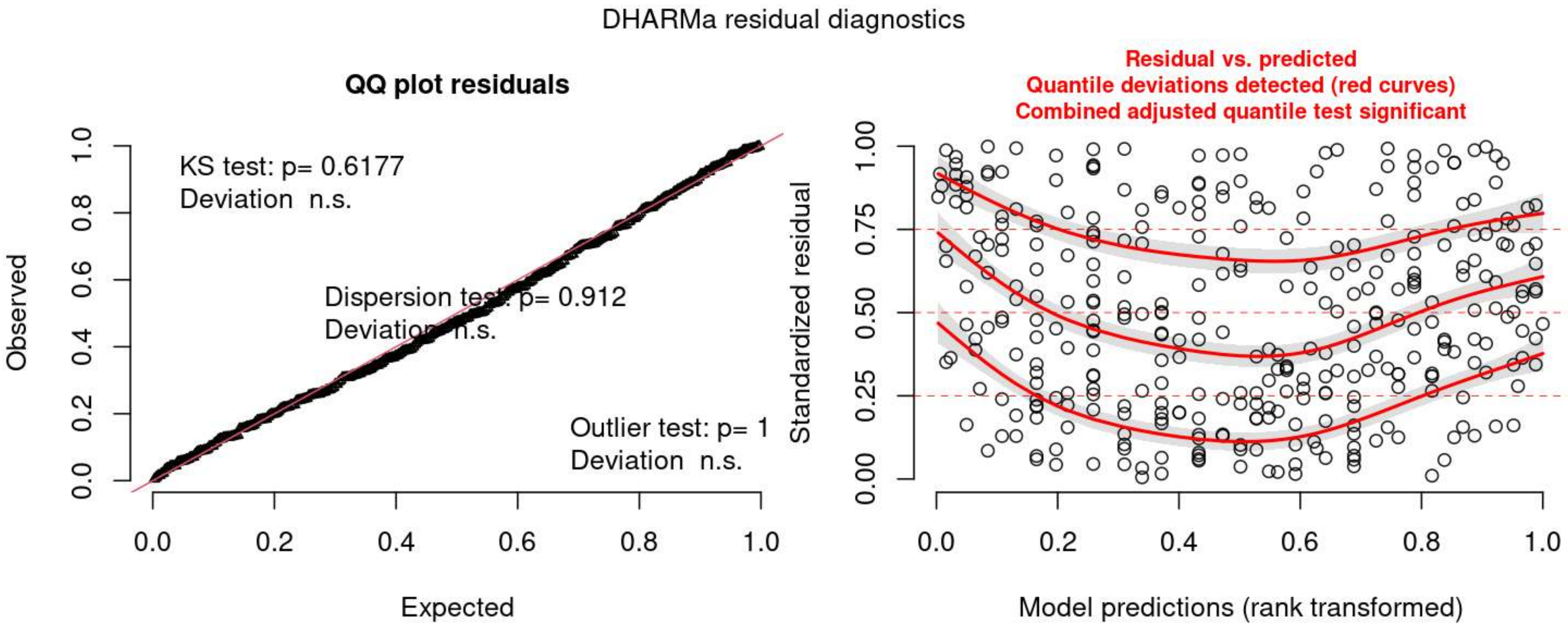
Applies to any model

- REMEMBER! Not assessing normality of residuals or heteroscedacity...
- Assessing whether data fits the model
- This *includes* assumptions, but *also* includes general fit, etc.
- So, we can use this to see if our model could be improved...

DHARMa Package

Check a previous model

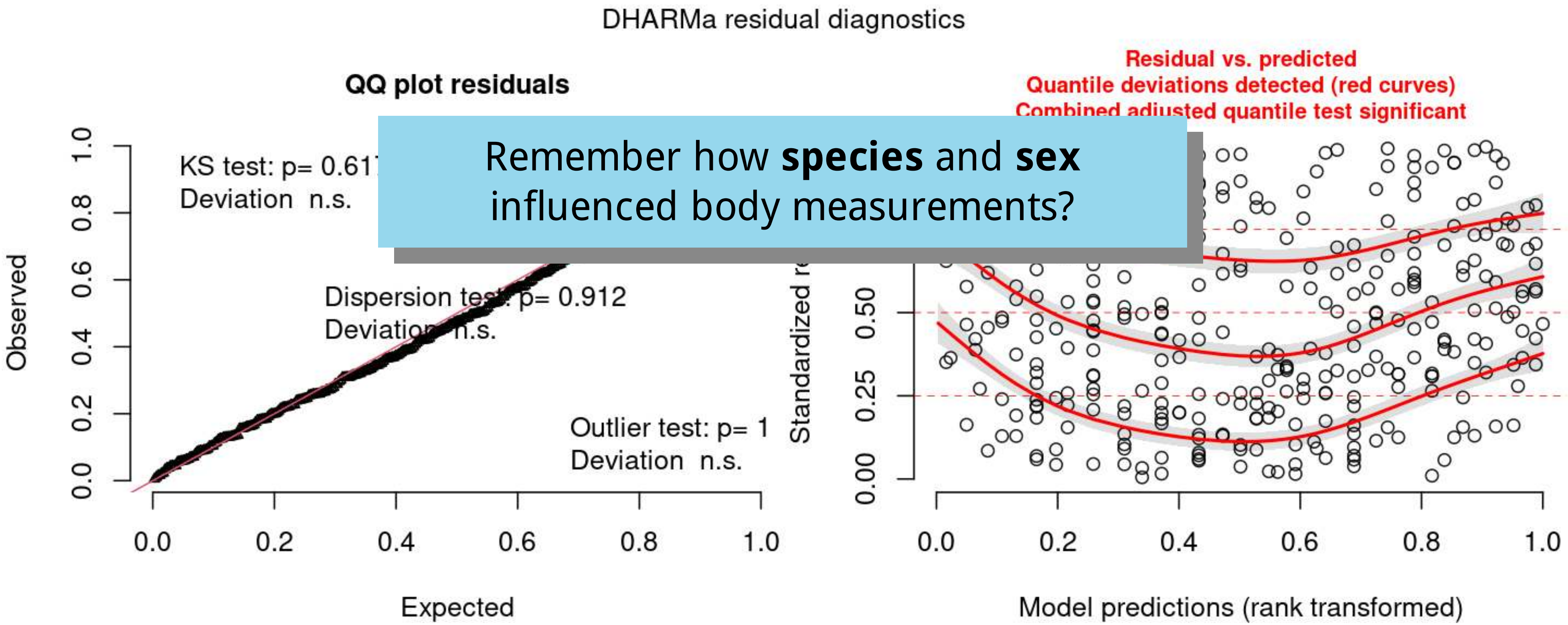
```
m <- lm(body_mass_g ~ flipper_length_mm, data = penguins)
r <- simulateResiduals(m, n = 1000, plot = TRUE)
```



DHARMa Package

Check a previous model

```
m <- lm(body_mass_g ~ flipper_length_mm, data = penguins)
r <- simulateResiduals(m, n = 1000, plot = TRUE)
```

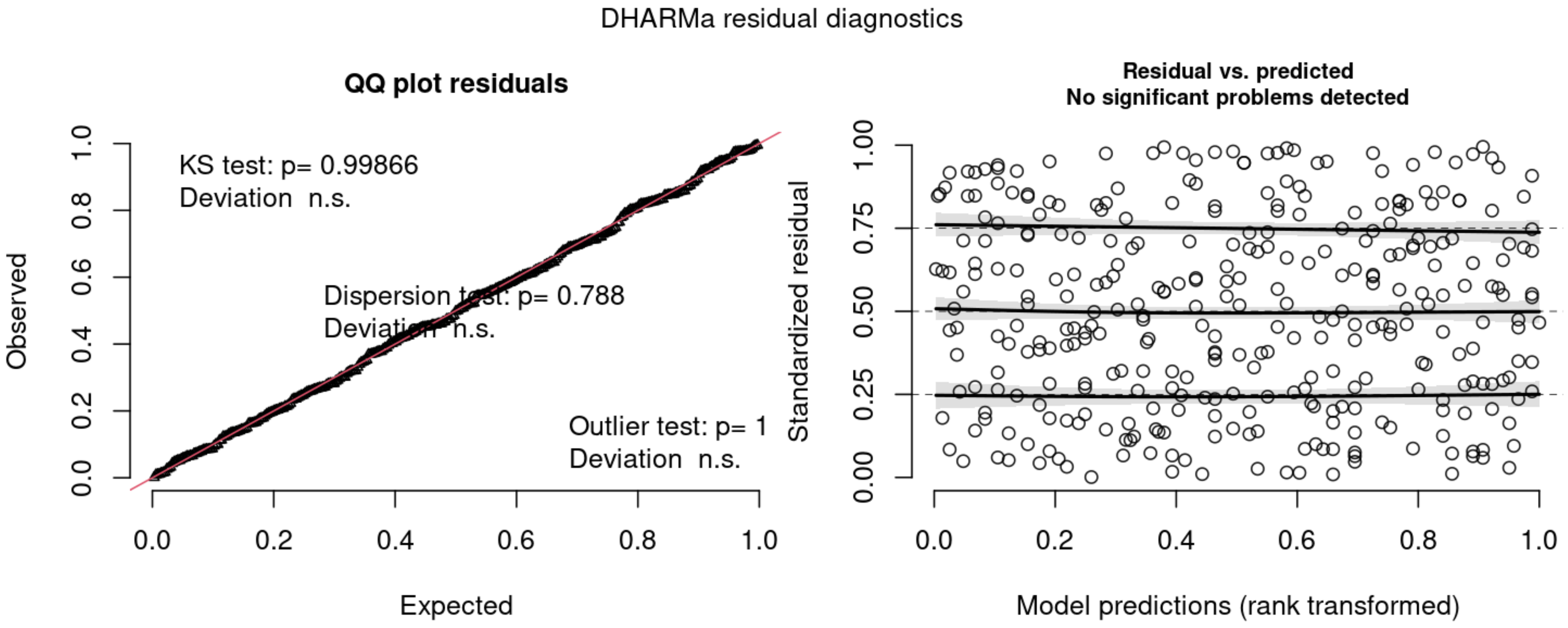


Remember how **species** and **sex** influenced body measurements?

DHARMa Package

Improve the fit…

```
m <- lm(body_mass_g ~ flipper_length_mm + species * sex, data = penguins)
r <- simulateResiduals(m, n = 1000, plot = TRUE)
```



Comparing Models

Model Selection

Model Selection

How do I know which model to use?

1. Check your diagnostics
    - Generally speaking, pick the model which has good diagnostics

Model Selection

How do I know which model to use?

1. Check your diagnostics
    - Generally speaking, pick the model which has good diagnostics

All models have reasonably good diagnostics?

Model Selection

How do I know which model to use?

1. Check your diagnostics
    - Generally speaking, pick the model which has good diagnostics

All models have reasonably good diagnostics?

2. Check model fit (how good is model at explaining data?)
    - Analysis of Variance/Deviance - `anova()`
    - Information Theoretic Approach (AIC) - `model.sel()` (MuMIn package)
        - (covered in class Nov 17th)

Model Selection

How do I know which model to use?

1. Check your diagnostics
    - Generally speaking, pick the model which has good diagnostics

All models have reasonably good diagnostics?

2. Check model fit (how good is model at explaining data?)
    - Analysis of Variance/Deviance - `anova()`
    - Information Theoretic Approach (AIC) - `model.sel()` (MuMIn package)
        - (covered in class Nov 17th)

> Many other methods, functions, packages

`anova()` - Analysis of Variance/Deviance

Very common in R

- Sequential analysis of variance for `lm()`
- Analysis of deviance for `glm()`

```
p <- na.omit(penguins)  # Must have exact same dataset for each model

m1 <- lm(body_mass_g ~ flipper_length_mm, data = p)
m2 <- lm(body_mass_g ~ flipper_length_mm + species + sex, data = p)
m3 <- lm(body_mass_g ~ flipper_length_mm + species * sex, data = p)

anova(m1, m2, m3)
```

`anova()` - Analysis of Variance/Deviance

```r
anova(m1, m2, m3)
```

```
## Analysis of Variance Table
##
## Model 1: body_mass_g ~ flipper_length_mm
## Model 2: body_mass_g ~ flipper_length_mm + species + sex
## Model 3: body_mass_g ~ flipper_length_mm + species * sex
##   Res.Df      RSS Df Sum of Sq      F    Pr(>F)
## 1    331 51211963
## 2    328 28653568  3  22558395 91.082 < 2.2e-16 ***
## 3    326 26913579  2   1739989 10.538 3.675e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

anova() - Analysis of Variance/Deviance

```
anova(m1, m2, m3)
```

```
## Analysis of Variance Table
##
## Model 1: body_mass_g ~ flipper_length_mm
## Model 2: body_mass_g ~ flipper_length_mm + species + sex
## Model 3: body_mass_g ~ flipper_length_mm + species * sex
##   Res.Df      RSS Df Sum of Sq      F    Pr(>F)
## 1    331 51211963
## 2    328 28653568  3  22558395 91.082 < 2.2e-16 ***
## 3    326 26913579  2   1739989 10.538 3.675e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Models being compared

`anova()` - Analysis of Variance/Deviance

```
anova(m1, m2, m3)
```

```
## Analysis of Variance Table
##
## Model 1: body_mass_g ~ flipper_length_mm
## Model 2: body_mass_g ~ flipper_length_mm + species + sex
## Model 3: body_mass_g ~ flipper_length_mm + species * sex
##   Res.Df      RSS Df Sum of Sq      F    Pr(>F)
## 1    331 51211963
## 2    328 28653568  3  22558395 91.082 < 2.2e-16 ***
## 3    326 26913579  2   1739989 10.538 3.675e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Models being compared
- How the models compare

`anova()` - Analysis of Variance/Deviance

```
anova(m1, m2, m3)
```

```
## Analysis of Variance Table
## 
## Model 1: body_mass_g ~ flipper_length_mm
## Model 2: body_mass_g ~ flipper_length_mm + species + sex
## Model 3: body_mass_g ~ flipper_length_mm + species * sex
##   Res.Df      RSS Df Sum of Sq      F    Pr(>F)    
## 1    331 51211963                                  
## 2    328 28653568  3  22558395 91.082 < 2.2e-16 ***
## 3    326 26913579  2   1739989 10.538 3.675e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Models being compared
- How the models compare
    - m2 has significantly ($P < 0.05$) lower Sums Squares than m1

`anova()` - Analysis of Variance/Deviance

```
anova(m1, m2, m3)
```

```
## Analysis of Variance Table
##
## Model 1: body_mass_g ~ flipper_length_mm
## Model 2: body_mass_g ~ flipper_length_mm + species + sex
## Model 3: body_mass_g ~ flipper_length_mm + species * sex
##   Res.Df      RSS Df Sum of Sq      F    Pr(>F)
## 1    331 51211963
## 2    328 28653568  3  22558395 91.082 < 2.2e-16 ***
## 3    326 26913579  2   1739989 10.538 3.675e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Models being compared
- How the models compare
  - m2 has significantly ($P < 0.05$) lower Sums Squares than m1
  - m3 has significantly ($P < 0.05$) lower Sums Squares than m2

anova() - Analysis of Variance/Deviance

```r
anova(m1, m2, m3)
```

```
## Analysis of Variance Table
##
## Model 1: body_mass_g ~ flipper_length_mm
## Model 2: body_mass_g ~ flipper_length_mm + species + sex
## Model 3: body_mass_g ~ flipper_length_mm + species * sex
##   Res.Df      RSS Df Sum of Sq      F    Pr(>F)
## 1    331 51211963
## 2    328 28653568  3  22558395 91.082 < 2.2e-16 ***
## 3    326 26913579  2   1739989 10.538 3.675e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Therefore
m3 > m2 > m1

- Models being compared
- How the models compare
  - m2 has significantly ($P < 0.05$) lower Sums Squares than m1
  - m3 has significantly ($P < 0.05$) lower Sums Squares than m2

`model.sel()` - AIC model selection

```r
library(MuMIn)
model.sel(m1, m2, m3)
```

```
## Model selection table
##       (Int) flp_lng_mm sex spc sex:spc             family df    logLik   AICc  delta weight
## m3   -478.3      20.49   +   +       + gaussian(identity)  8 -2353.956 4724.4   0.00      1
## m2   -365.8      20.02   +   +         gaussian(identity)  6 -2364.387 4741.0  16.67      0
## m1 -5872.0      50.15                  gaussian(identity)  3 -2461.073 4928.2 203.86      0
## Models ranked by AICc(x)
```

- m3 is the best (lowest AICc, and much lower than the others)
- For more specifics on how to use/interpret this table, stay tuned for Nicky's lecture

`model.sel()` - AIC model selection

```r
library(MuMIn)
model.sel(m1, m2, m3)
```

```
## Model selection table
##       (Int) flp_lng_mm sex spc sex:spc             family df    logLik   AICc  delta weight
## m3  -478.3       20.49   +   +       + gaussian(identity)  8 -2353.956 4724.4   0.00      1
## m2  -365.8       20.02   +   +         gaussian(identity)  6 -2364.387 4741.0  16.67      0
## m1 -5872.0       50.15                 gaussian(identity)  3 -2461.073 4928.2 203.86      0
## Models ranked by AICc(x)
```

- m3 is the best (lowest AICc, and much lower than the others)
- For more specifics on how to use/interpret this table, stay tuned for Nicky's lecture

> Always, remember that your 'best' model is only the best of what you've compared...

Generalized Linear Models

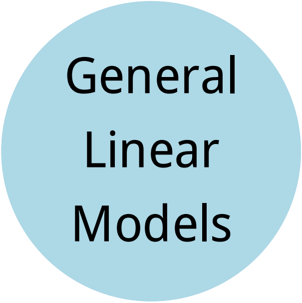
Chapter 13, "The R Book" by Michael J. Crawley

(Freely available online through University of Manitoba Library)

Generalized linear models

Simple
Linear
Models

Generalized linear models



General
Linear
Models

Generalized linear models

- Residuals are normally distributed
- Normally distributed also known as "Gaussian" distribution

General
Linear
Models

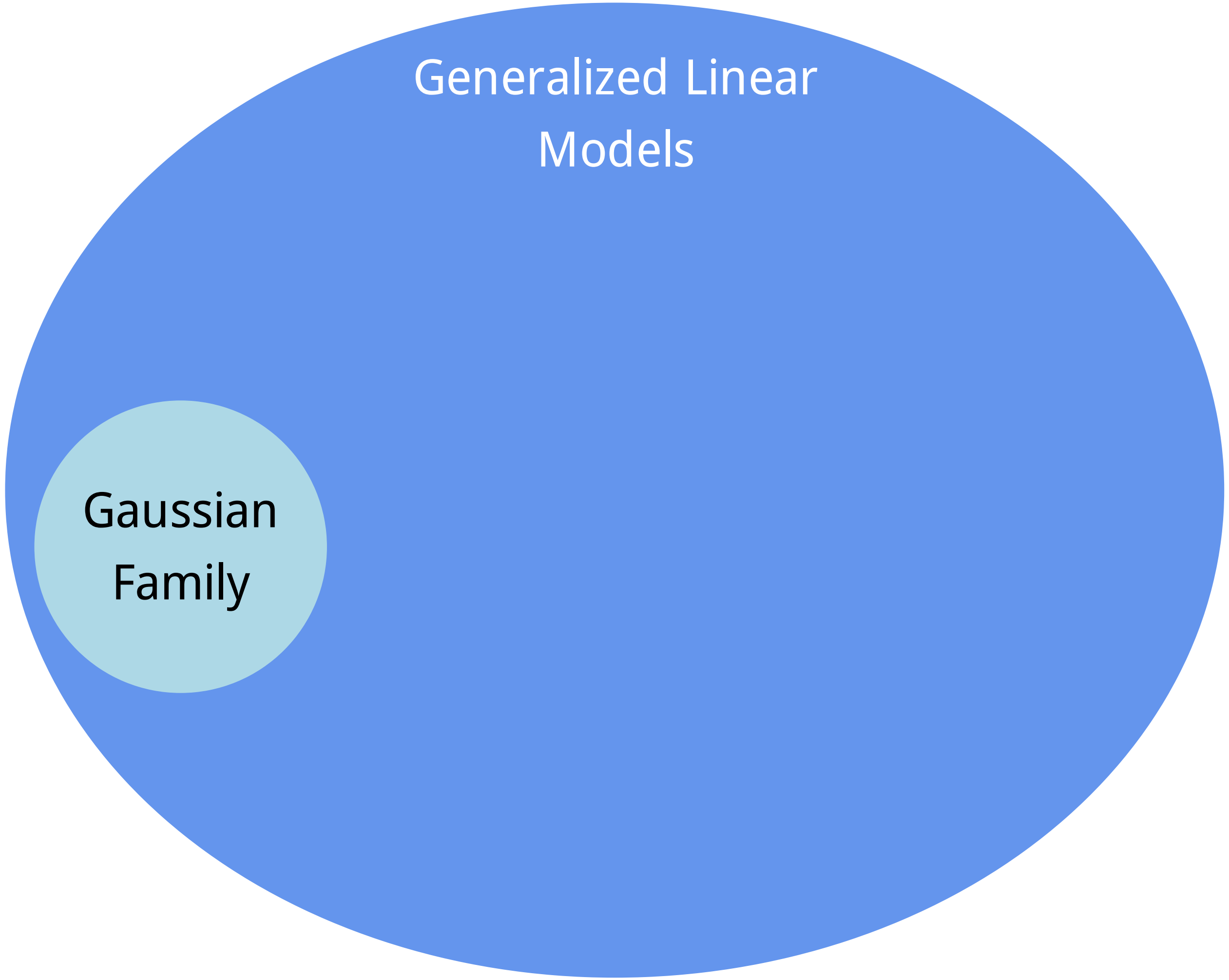
Generalized linear models

- Residuals are normally distributed
- Normally distributed also known as "Gaussian" distribution

Gaussian
Family

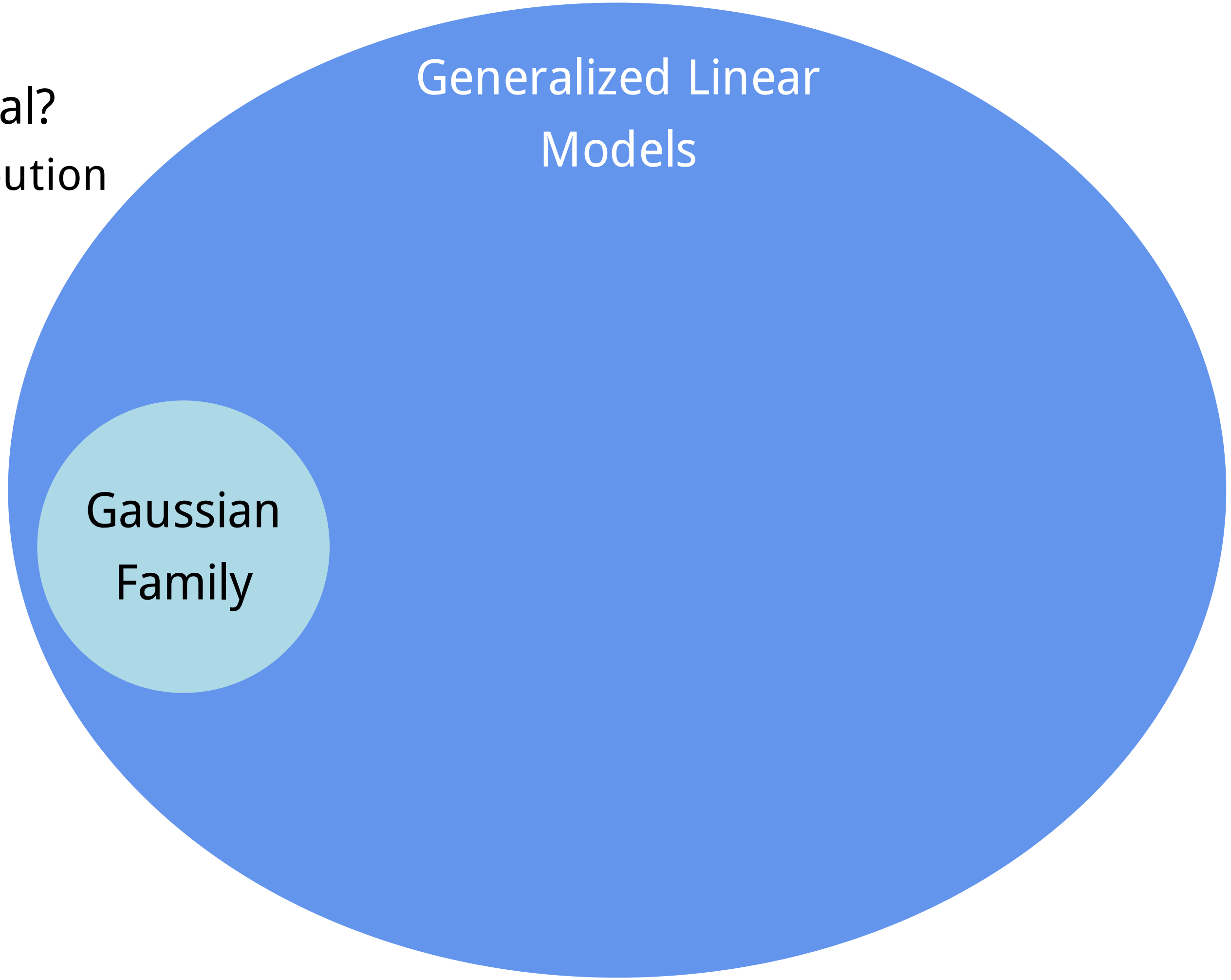
Generalized linear models

- One type of Generalized linear model



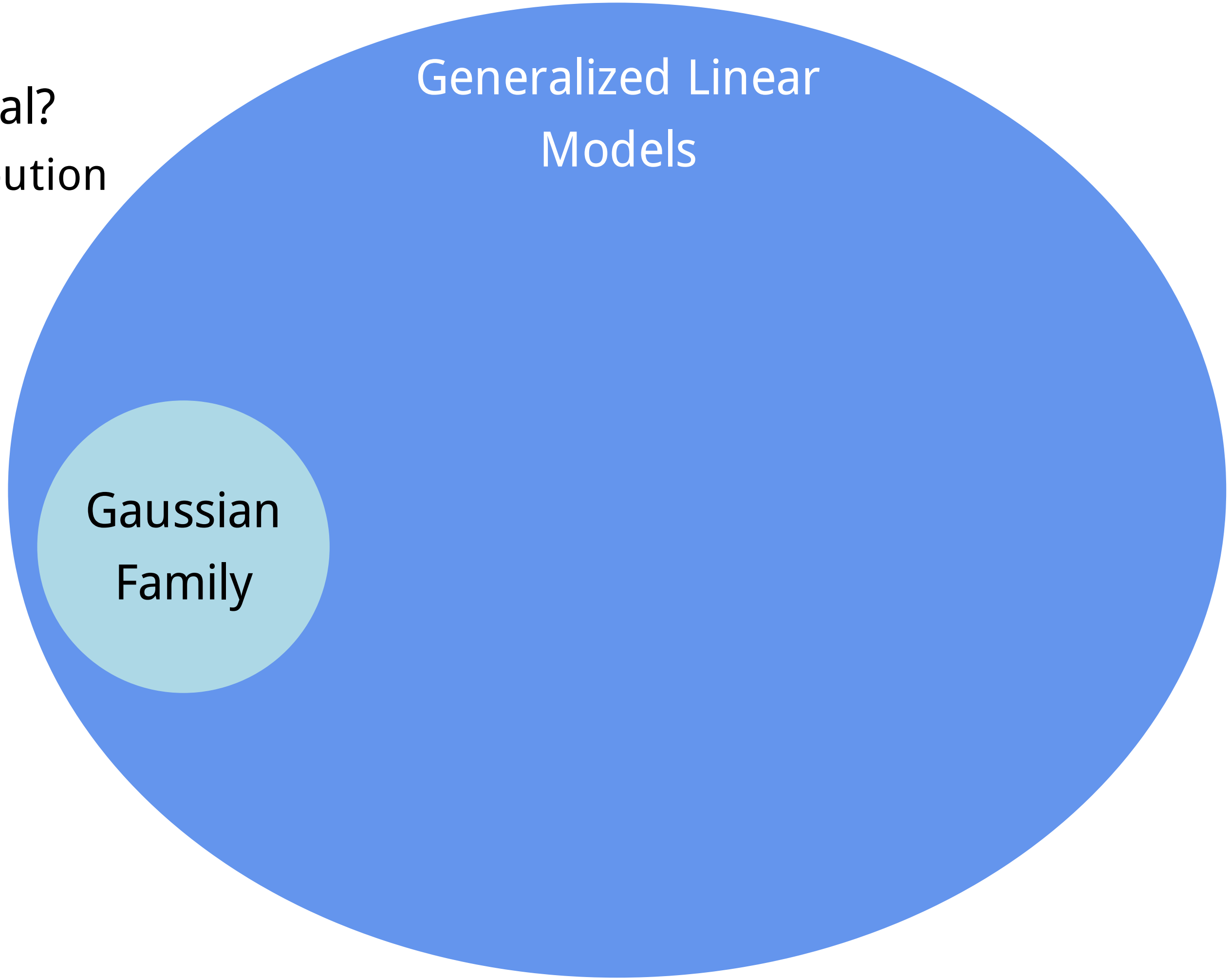
Generalized linear models

- One type of Generalized linear model
- So what if your residuals are *not* normal?
    - ie. they don't follow a Gaussian distribution



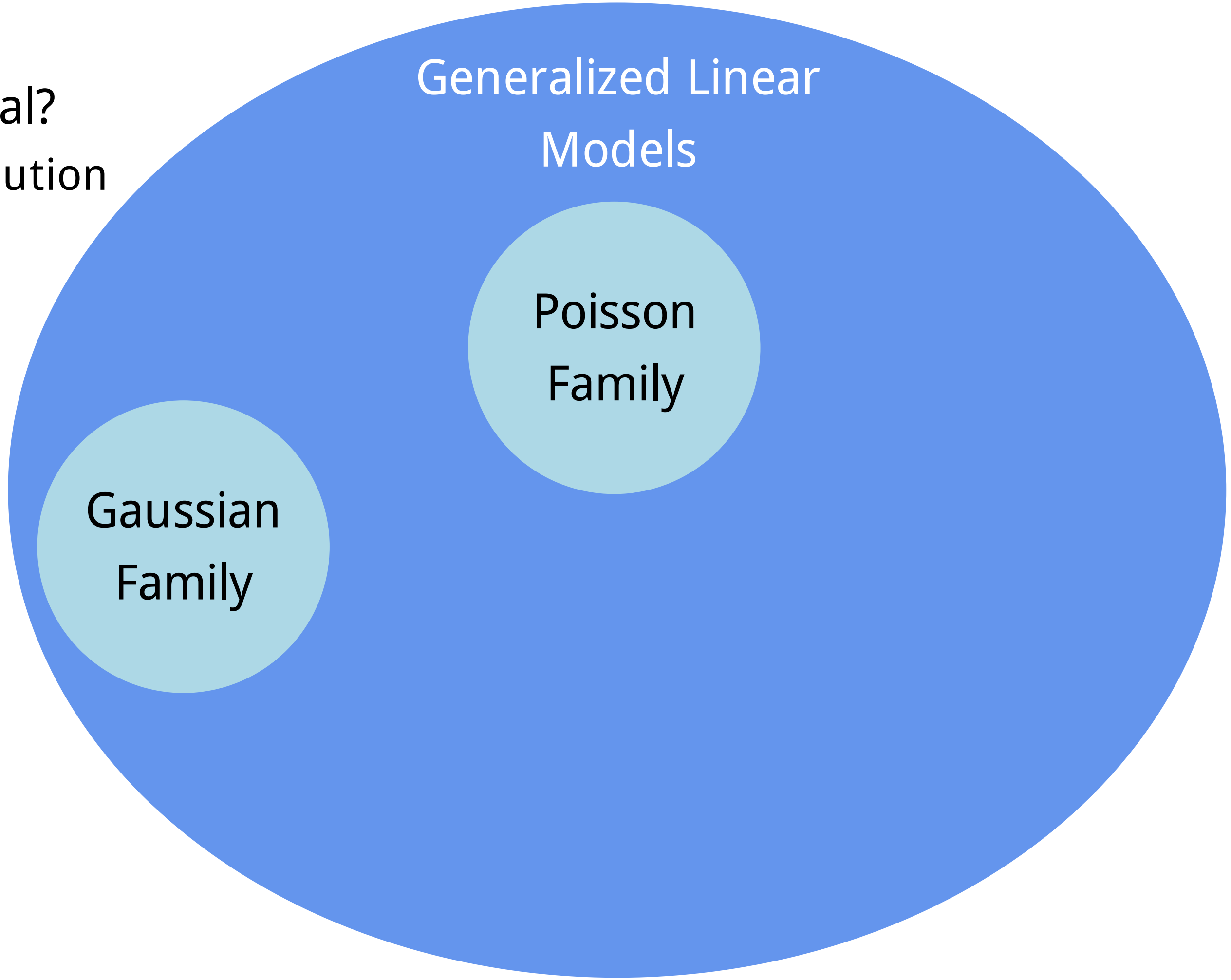
Generalized linear models

- One type of Generalized linear model
- So what if your residuals are *not* normal?
    - ie. they don't follow a Gaussian distribution
- Try a different family!



Generalized linear models

- One type of Generalized linear model
- So what if your residuals are *not* normal?
    - ie. they don't follow a Gaussian distribution
- Try a different family!



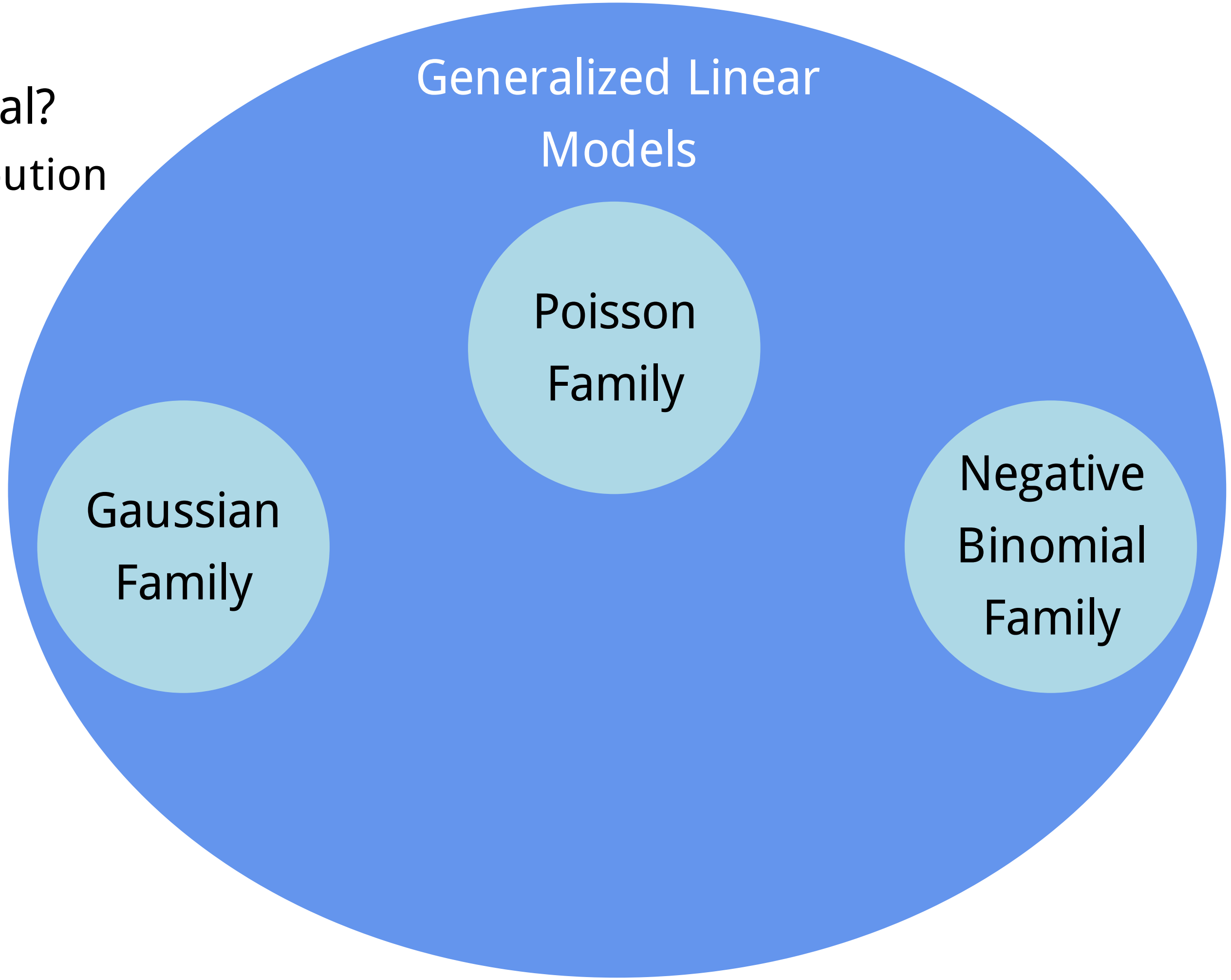
Generalized Linear Models

Poisson Family

Gaussian Family

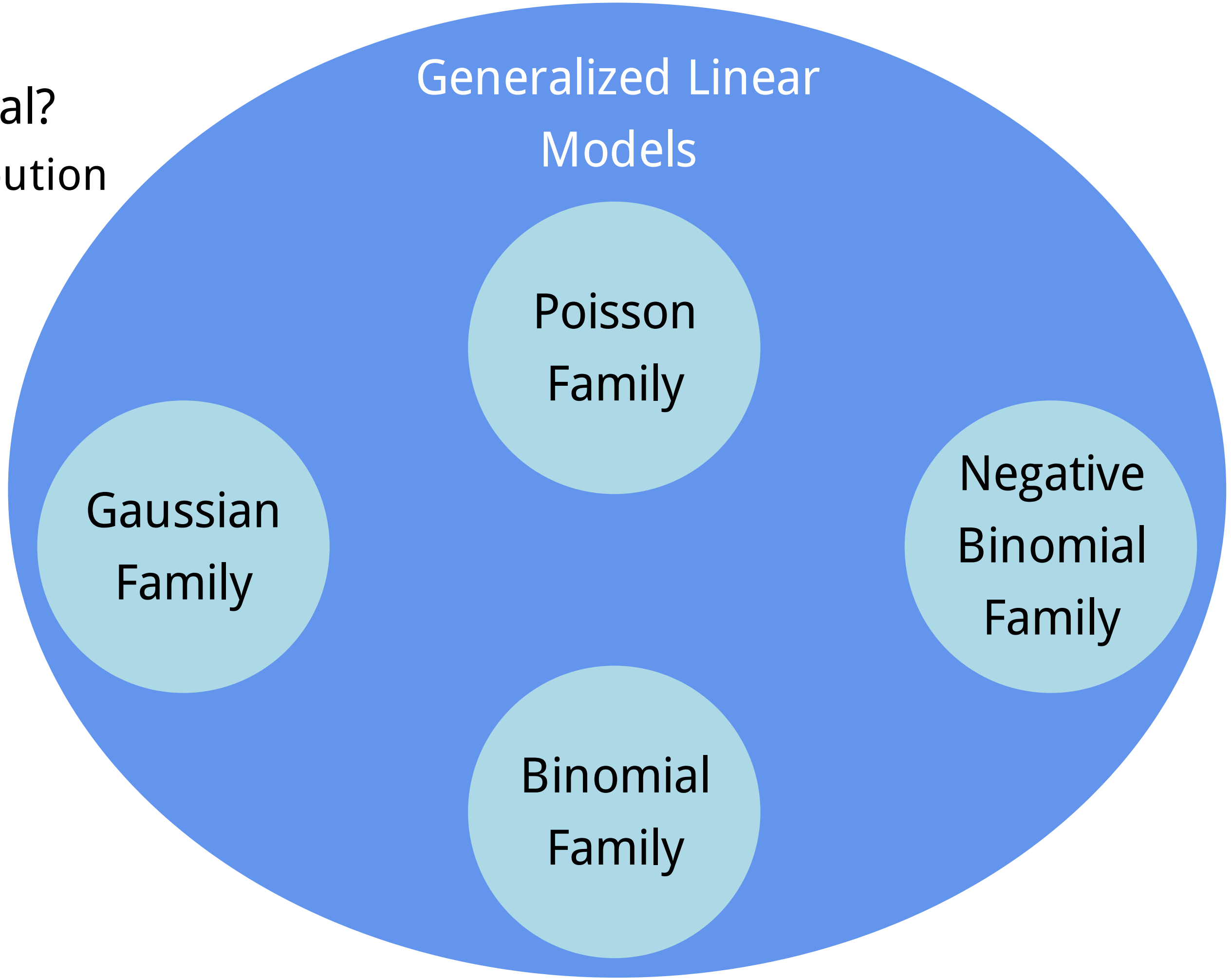
Generalized linear models

- One type of Generalized linear model
- So what if your residuals are *not* normal?
    - ie. they don't follow a Gaussian distribution
- Try a different family!



Generalized linear models

- One type of Generalized linear model
- So what if your residuals are *not* normal?
    - ie. they don't follow a Gaussian distribution
- Try a different family!



Generalized Linear Models

Poisson Family

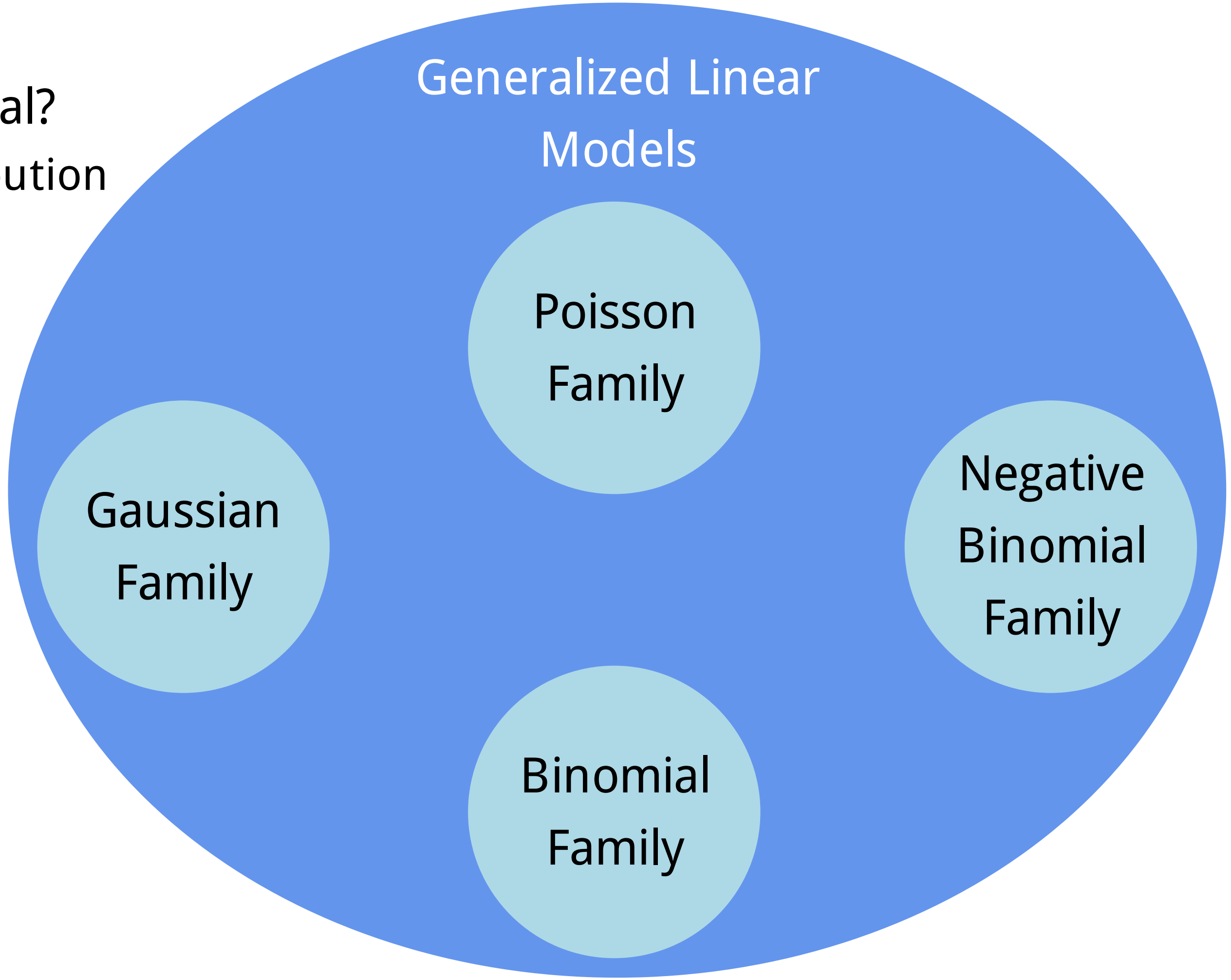
Gaussian Family

Negative Binomial Family

Binomial Family

Generalized linear models

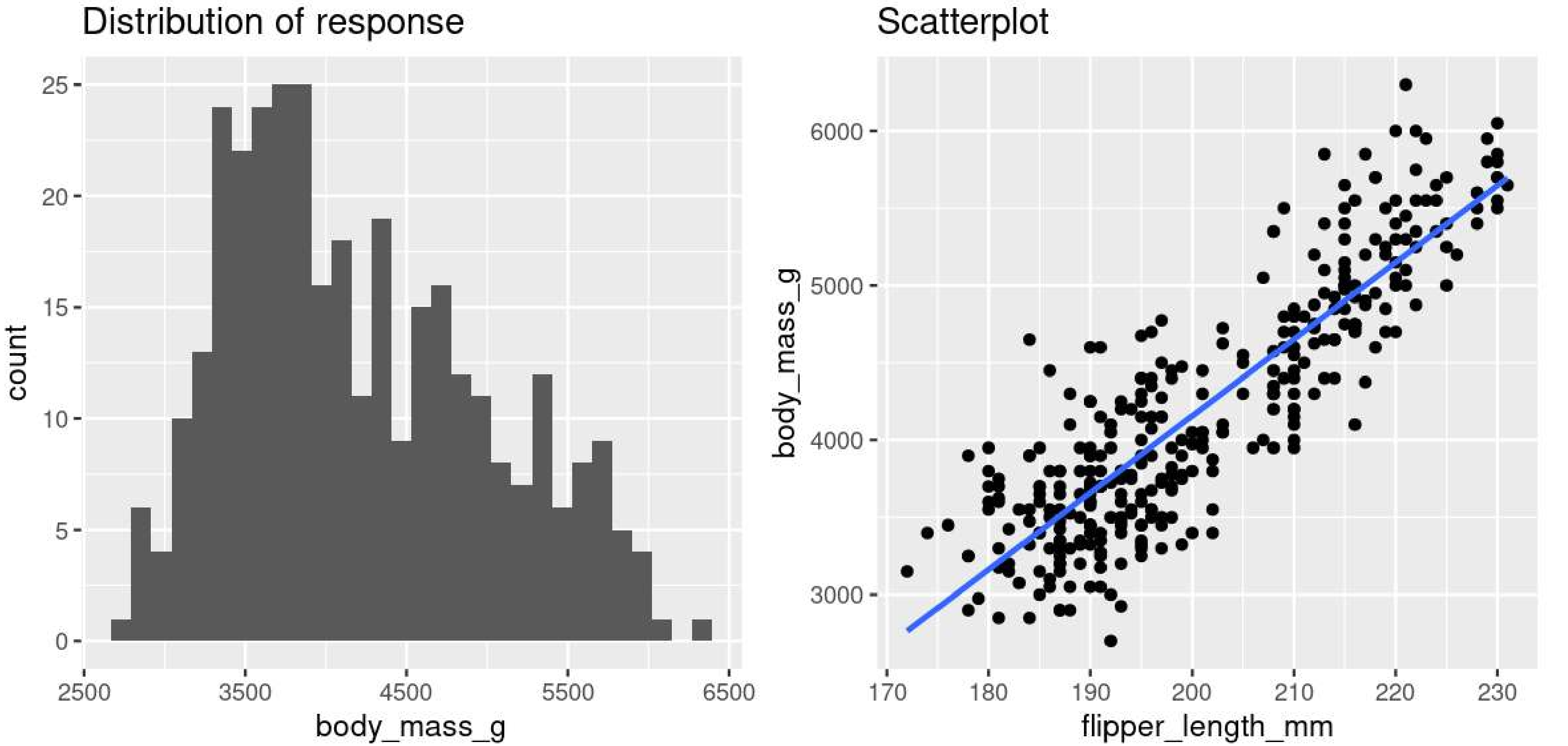
- One type of Generalized linear model
- So what if your residuals are *not* normal?
    - ie. they don't follow a Gaussian distribution
- Try a different family!
- Etc.



Generalized linear models

Gaussian Family (Normal data)

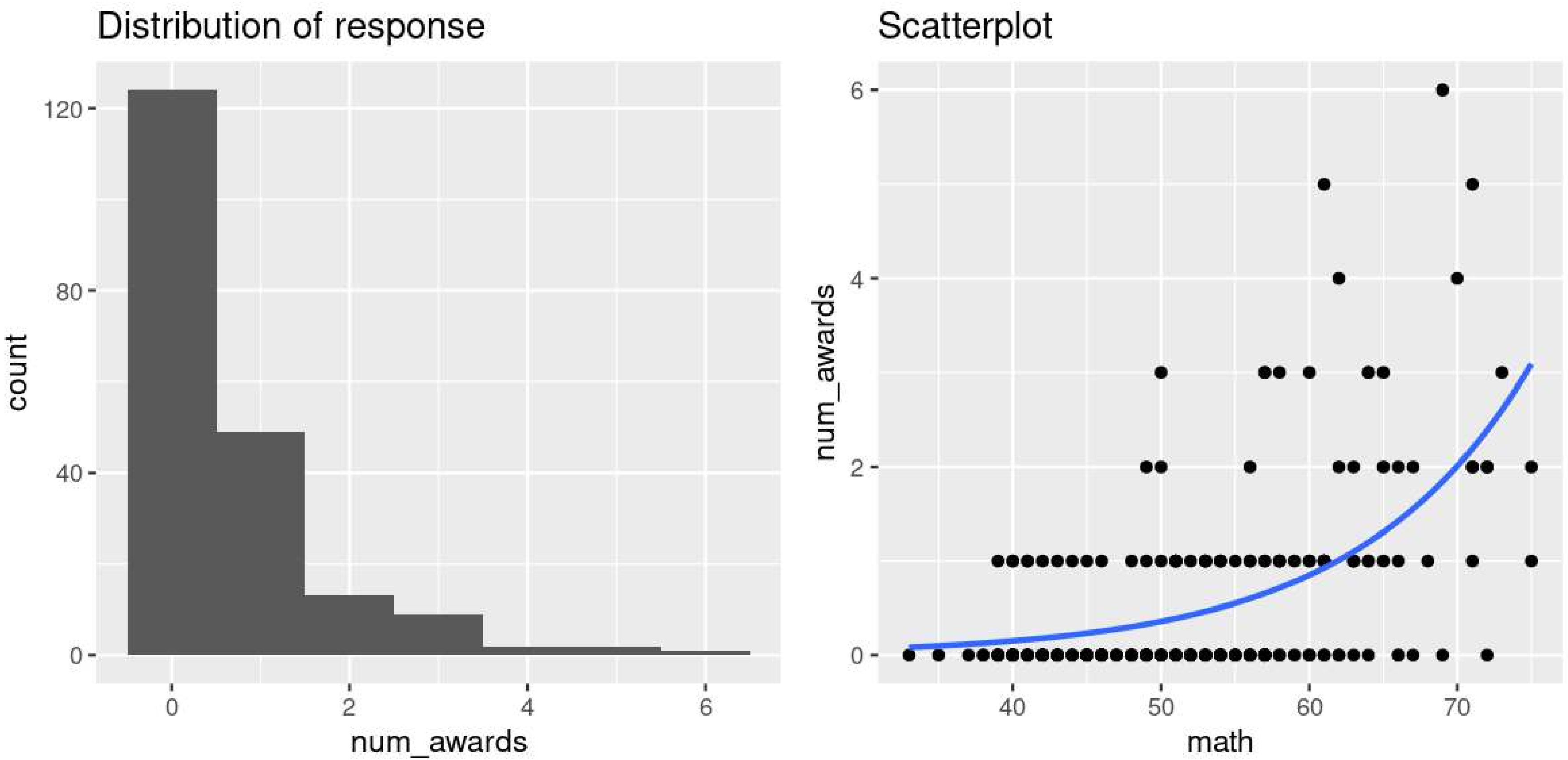
```
lm(y ~ x1 * x2, data = my_data)
```



Generalized linear models

Poisson Family (Count data)

```
glm(counts ~ x1 * x2, family = "poisson", data = my_data)
```



Generalized linear models

Negative Binomial Family (Overdispersed Count data)

```
MASS::glm.nb(counts ~ x1 * x2,  data = my_data)
```

- Do Poisson model first
- Check diagnostics, if necessary, you'll do a negative binomial model

Generalized linear models

Binomial Family (Binary data)

- TRUE/FALSE, 0/1, Logistic Regression

```
glm(y ~ x1 * x2, family = "binomial", data = my_data)
```

Generalized linear models

Binomial Family (Binary data)

- TRUE/FALSE, 0/1, Logistic Regression

```
glm(y ~ x1 * x2, family = "binomial", data = my_data)
```

Binomial Family (Proportion data)

- Two responses (binary outcomes), each in it's own column
    - (e.g., number of Yes, vs. No; Number of songs, vs. calls)

```
glm(cbind(yes, no) ~ x1 * x2, family = "binomial", data = my_data)
```

Examples

Count Data - Poisson Family

Get the data

```
p <- read_csv("https://stats.idre.ucla.edu/stat/data/poisson_sim.csv")
p <- mutate(p, program = factor(prog, levels = 1:3,
                                labels = c("General", "Academic", "Vocational")))

p
```

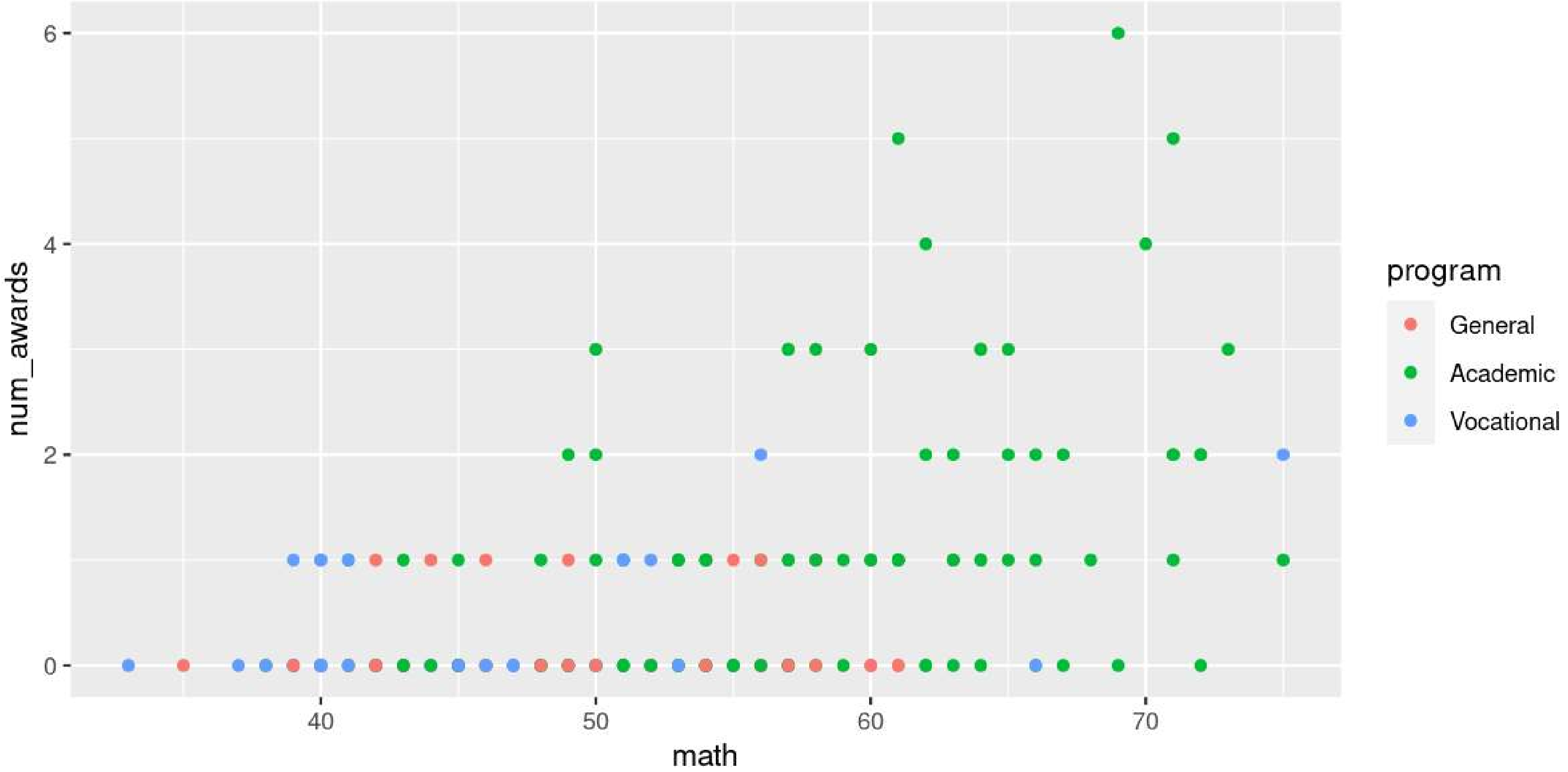
```
## # A tibble: 200 × 5
##         id num_awards  prog  math program
##      <dbl>      <dbl> <dbl> <dbl> <fct>
##  1     45          0     3    41 Vocational
##  2    108          0     1    41 General
##  3     15          0     3    44 Vocational
##  4     67          0     3    42 Vocational
##  5    153          0     3    40 Vocational
##  6     51          0     1    42 General
##  7    164          0     3    46 Vocational
##  8    133          0     3    40 Vocational
##  9      2          0     3    33 Vocational
## 10     53          0     3    46 Vocational
## # … with 190 more rows
```

How do marks in **`math`** as well as **`program`** of study influence the number of awards a student receives (**`num_awards`**)?

Count Data - Poisson Family

Look at the data

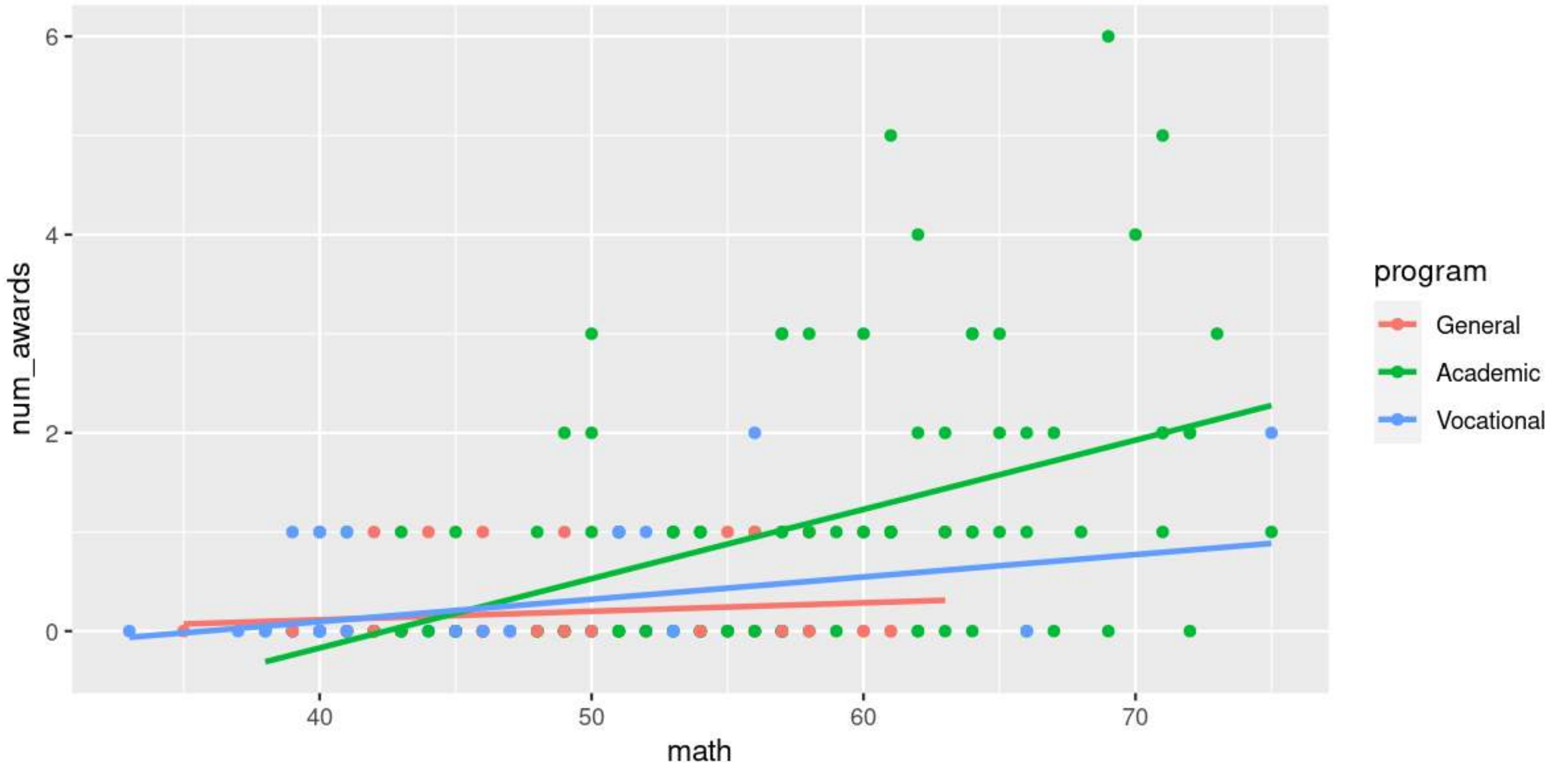
```
ggplot(data = p, aes(x = math, y = num_awards, colour = program)) +
  geom_point()
```



Count Data - Poisson Family

Look at the data

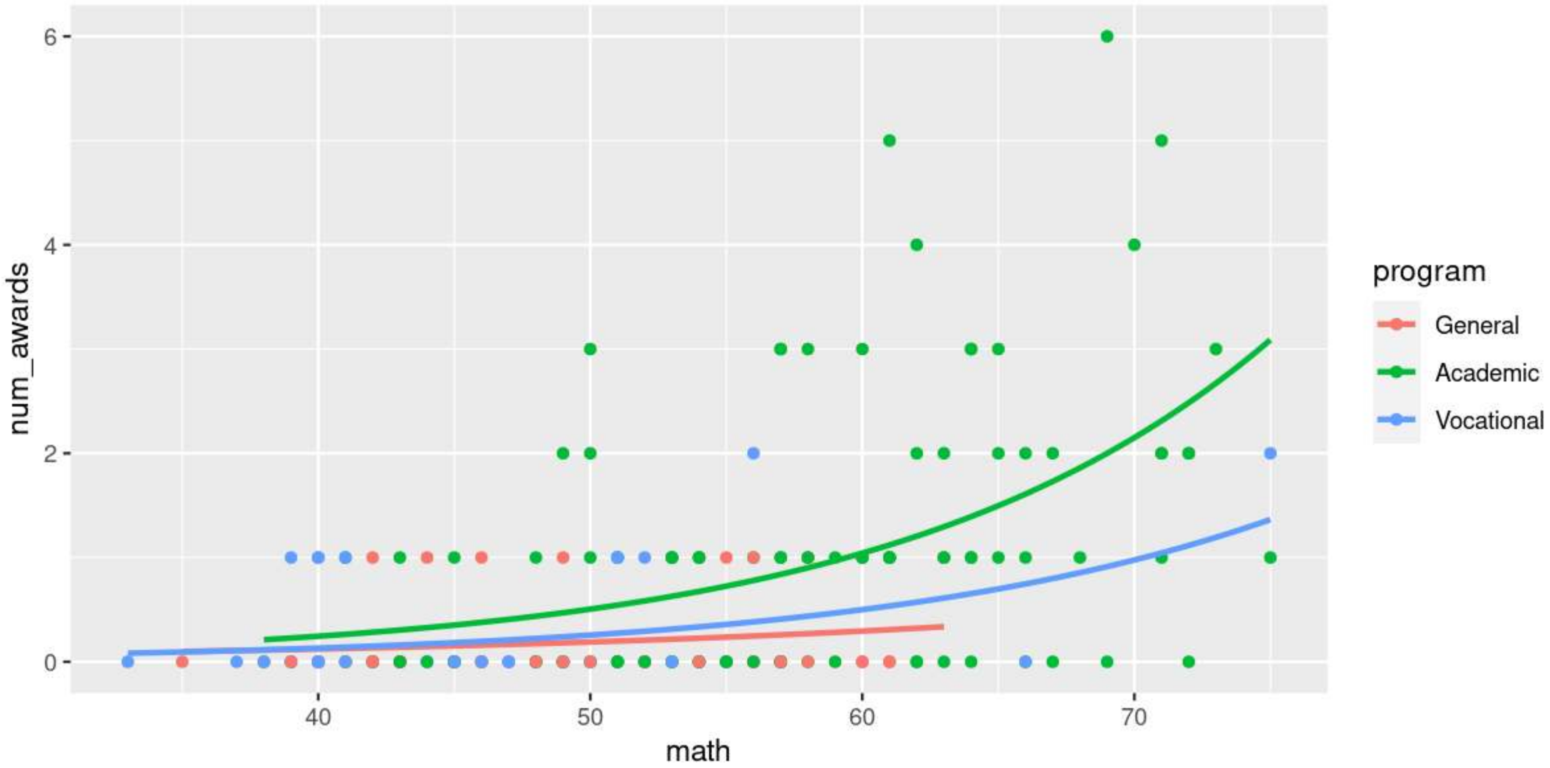
```
ggplot(data = p, aes(x = math, y = num_awards, colour = program)) +
  geom_point() +
  stat_smooth(method = "glm", se = FALSE)
```



Count Data - Poisson Family

Look at the data

```
ggplot(data = p, aes(x = math, y = num_awards, colour = program)) +
  geom_point() +
  stat_smooth(method = "glm", method.args = list(family = "poisson"), se = FALSE)
```



Count Data - Poisson Family

Run the model

```
m <- glm(num_awards ~ math + program, family = "poisson", data = p)
```

Count Data - Poisson Family

Diagnostics

```r
summary(m)
```

```
## Call:
## glm(formula = num_awards ~ math + program, family = "poisson",
##     data = p)
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -5.24712    0.65845  -7.969 1.60e-15 ***
## math                0.07015    0.01060   6.619 3.63e-11 ***
## programAcademic     1.08386    0.35825   3.025  0.00248 **
## programVocational  0.36981    0.44107   0.838  0.40179
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 287.67  on 199  degrees of freedom
## Residual deviance: 189.45  on 196  degrees of freedom
## AIC: 373.5
##
```

Overdispersion

- Dispersion = Residual Deviance / DF
- Expected to be 1 for Poisson
- Overdispersion > 1; Underdispersion < 1

Count Data - Poisson Family

Diagnostics

```
summary(m)
```

```
## Call:
## glm(formula = num_awards ~ math + program, family = "poisson",
##     data = p)
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -5.24712    0.65845  -7.969 1.60e-15 ***
## math                0.07015    0.01060   6.619 3.63e-11 ***
## programAcademic     1.08386    0.35825   3.025  0.00248 **
## programVocational   0.36981    0.44107   0.838  0.40179
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 287.67  on 199  degrees of freedom
## Residual deviance: 189.45  on 196  degrees of freedom
## AIC: 373.5
##
```

Overdispersion

- Dispersion = Residual Deviance / DF
- Expected to be 1 for Poisson
- Overdispersion > 1; Underdispersion < 1

Traditional check:

- Residual Deviance (189.45) vs. DF (196)

```
deviance(m) / df.residual(m)
```

```
## [1] 0.9665797
```

- Almost 1 (which it is good)

```
pchisq(deviance(m), df.residual(m),
       lower.tail = FALSE)
```

```
## [1] 0.6182274
```

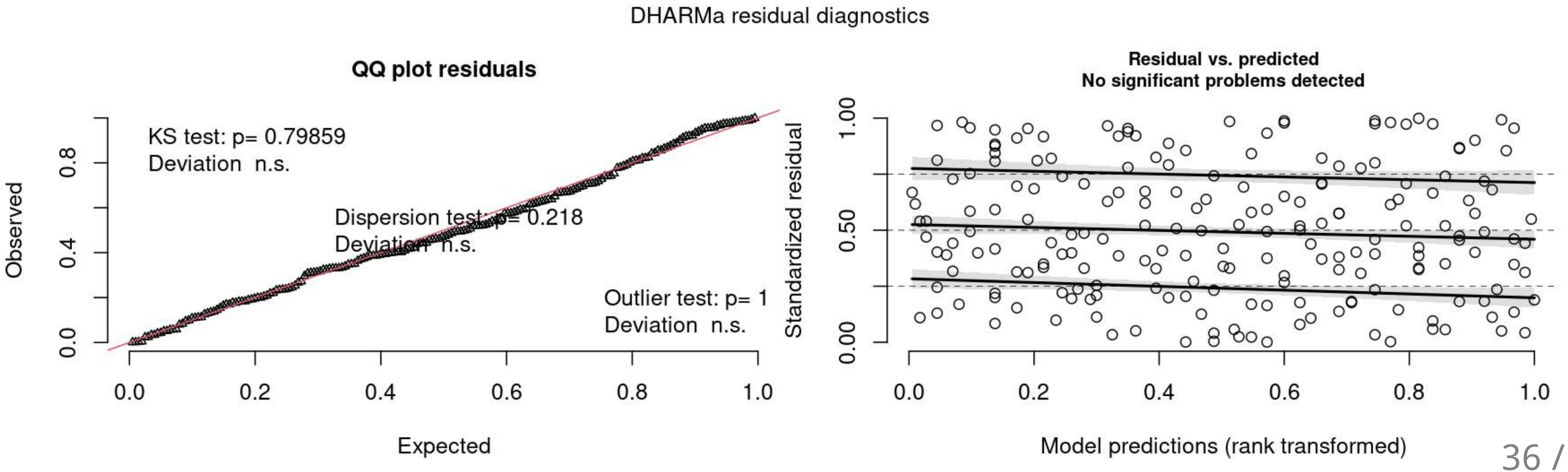
- Test shows no significant overdispersion (p = 0.62)

Count Data - Poisson Family

Diagnostics

- Residuals are complicated to assess in GLMs
- Therefore, use **`DHARMa`** package!

```
r <- simulateResiduals(m, n = 1000, plot = TRUE)
```



Count Data - Poisson Family

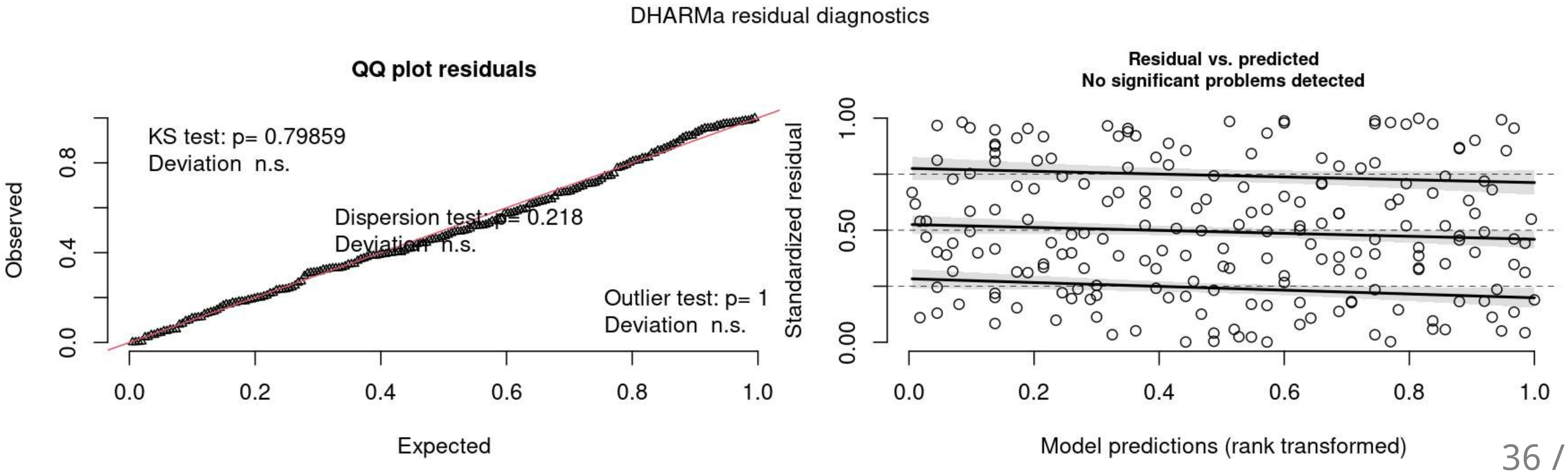
Diagnostics

- Residuals are complicated to assess in GLMs
- Therefore, use **DHARMa** package!

```
r <- simulateResiduals(m, n = 1000, plot = TRUE)
```

Also shows no overdispersion

(See Dispersion test)



Count Data - Poisson Family - Model Diagnostics

Zero-inflation (More zeros than expected)

- Will often pop up as iffy residuals
- Overdispersion can lead to false positives
- Here not a problem (non-significant P-value)

```
testZeroInflation(m, plot = FALSE)
```

```
## 
##      DHARMa zero-inflation test via comparison to expected zeros with simulation under H0 =
##      fitted model
## 
## data:  simulationOutput
## ratioObsSim = 1.0131, p-value = 0.832
## alternative hypothesis: two.sided
```

Count Data - Poisson Family

Interpretation

```
summary(m)
```

```
## Call:
## glm(formula = num_awards ~ math + program, family = "poisson",
##     data = p)
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -5.24712    0.65845  -7.969 1.60e-15 ***
## math                0.07015    0.01060   6.619 3.63e-11 ***
## programAcademic     1.08386    0.35825   3.025  0.00248 **
## programVocational   0.36981    0.44107   0.838  0.40179
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 287.67  on 199  degrees of freedom
## Residual deviance: 189.45  on 196  degrees of freedom
## AIC: 373.5
##
```

Effects

- Significantly more awards received with higher marks in math
    - i.e. Number of awards increases by 0.07 log-counts per 1 unit increase in Math mark
- Significantly more awards received in Academic Program compared to General
    - i.e. Number of awards greater by 1.08 log-counts for Academic compared to General
- No difference in amount of awards received in Vocational vs. General Program

Count Data - Poisson Family

Interpretation

```
summary(m)
```

```
## Call:
## glm(formula = num_awards ~ math + program, family = "poisson",
##     data = p)
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -5.24712    0.65845  -7.969 1.60e-15 ***
## math                0.07015    0.01060   6.619 3.63e-11 ***
## programAcademic     1.08386    0.35825   3.025  0.00248 **
## programVocational   0.36981    0.44107   0.838  0.40179
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 287.67  on 199  degrees of freedom
## Residual deviance: 189.45  on 196  degrees of freedom
## AIC: 373.5
##
```

Count Data - Poisson Family

Interpretation

```
summary(m)
```

```
## Call:
## glm(formula = num_awards ~ math + program, family = "poisson",
##     data = p)
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -5.24712    0.65845  -7.969 1.60e-15 ***
## math               0.07015    0.01060   6.619 3.63e-11 ***
## programAcademic    1.08386    0.35825   3.025  0.00248 **
## programVocational  0.36981    0.44107   0.838  0.40179
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 287.67  on 199  degrees of freedom
## Residual deviance: 189.45  on 196  degrees of freedom
## AIC: 373.5
##
```

Interpreting Results

- Convert to ratios with e^{est} (`exp()`)

```
data.frame(est = coef(m),
           ratios = exp(coef(m)))
```

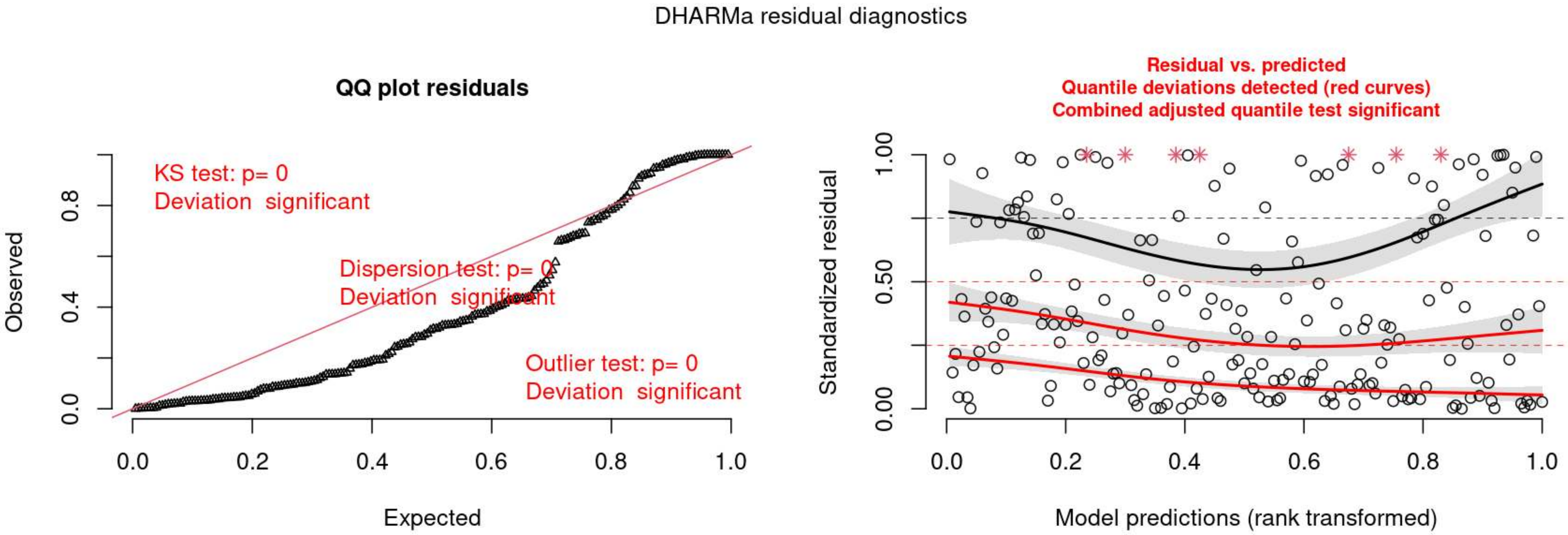
```
##                          est     ratios
## (Intercept)       -5.2471244 0.00526263
## math               0.0701524 1.07267164
## programAcademic    1.0838591 2.95606545
## programVocational  0.3698092 1.44745846
```

- No. awards increases by 1.07 *times* per 1 unit increase in Math mark (7%)

- No. awards received by Academic is 2.96 *times* greater than in General program

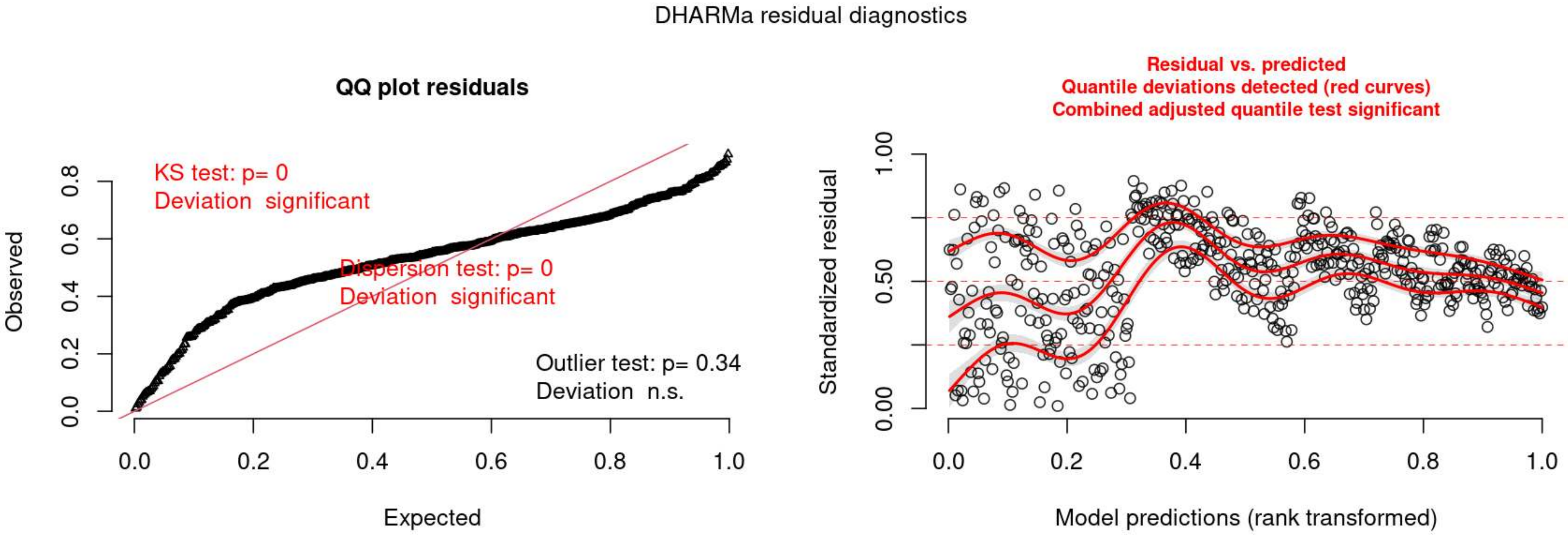
Overdispersion

- Overdispersion when data is spread out more than distribution would be (longer tails)
- Results in **highly** significant findings that are **not** valid!!
- Simulated residuals run from 0 to 1, but here more residuals around 0 and 1 (longer tails)



Underdispersion

- Underdispersion is less common
- When data is gathered towards the centre more than distribution would be (shorter tails)
- These simulated residuals run from 0 to 1
- Here, more residuals around 0.5 (shorter tails)



Overdispersed Count Data - Negative Binomial

```
quine <- MASS::quine
```

Poisson GLM

```
m1 <- glm(Days ~ Sex, data = quine, family =
"poisson")
summary(m1)
```

```
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.72294    0.02865  95.030  < 2e-16
***
## SexM         0.16490    0.04080   4.041 5.31e-05
***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05
'.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to
be 1)
##
##     Null deviance: 2073.5  on 145  degrees of
freedom
```

Negative Binomial GLM

```
m2 <- MASS::glm.nb(Days ~ Sex, data = quine)
summary(m2)
```

```
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.7229     0.1116  24.395   <2e-16
***
## SexM          0.1649     0.1656   0.996    0.319
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05
'.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(1.0741)
family taken to be 1)
##
##     Null deviance: 169.50  on 145  degrees of
freedom
## Residual deviance: 168.51  on 144  degrees of
freedom
```

Overdispersed Count Data - Negative Binomial

```
quine <- MASS::quine
```

Poisson GLM

```
m1 <- glm(Days ~ Sex, data = quine, family =
"poisson")
summary(m1)
```

```
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.72294    0.02865  95.030  < 2e-16
***
## SexM         0.16490    0.04080   4.041 5.31e-05
***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05
'.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to
be 1)
##
##     Null deviance: 2073.5  on 145  degrees of
freedom
```

Negative Binomial GLM

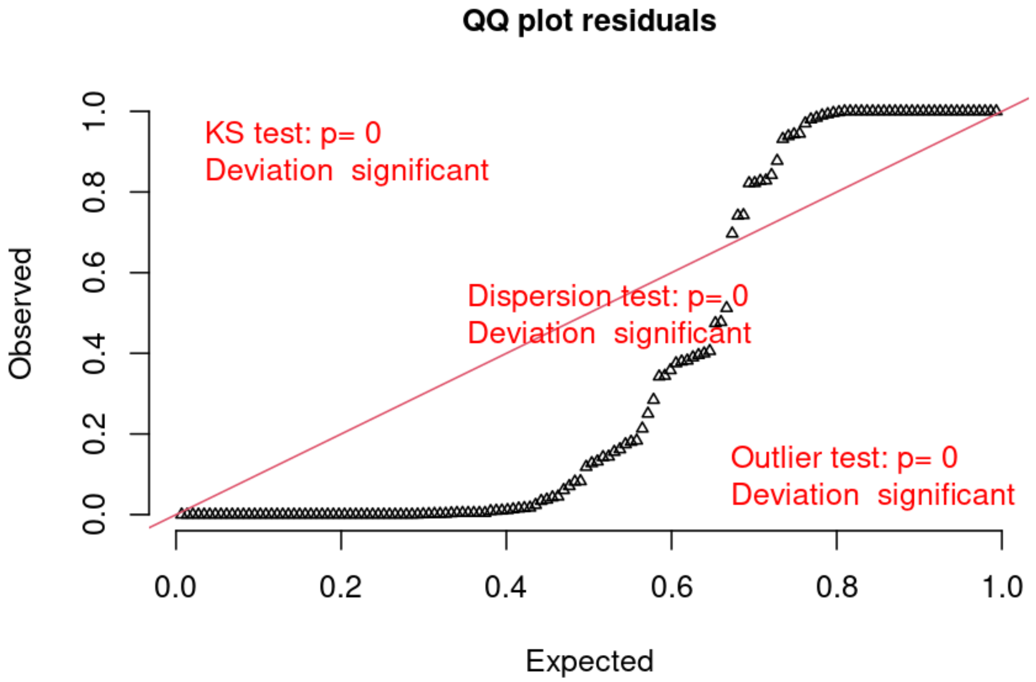
```
m2 <- MASS::glm.nb(Days ~ Sex, data = quine)
summary(m2)
```

```
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.7229     0.1116  24.395   <2e-16
***
## SexM          0.1649     0.1656   0.996    0.319
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05
'.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(1.0741)
family taken to be 1)
##
##     Null deviance: 169.50  on 145  degrees of
freedom
## Residual deviance: 168.51  on 144  degrees of
freedom
```

Overdispersed Count Data - Negative Binomial

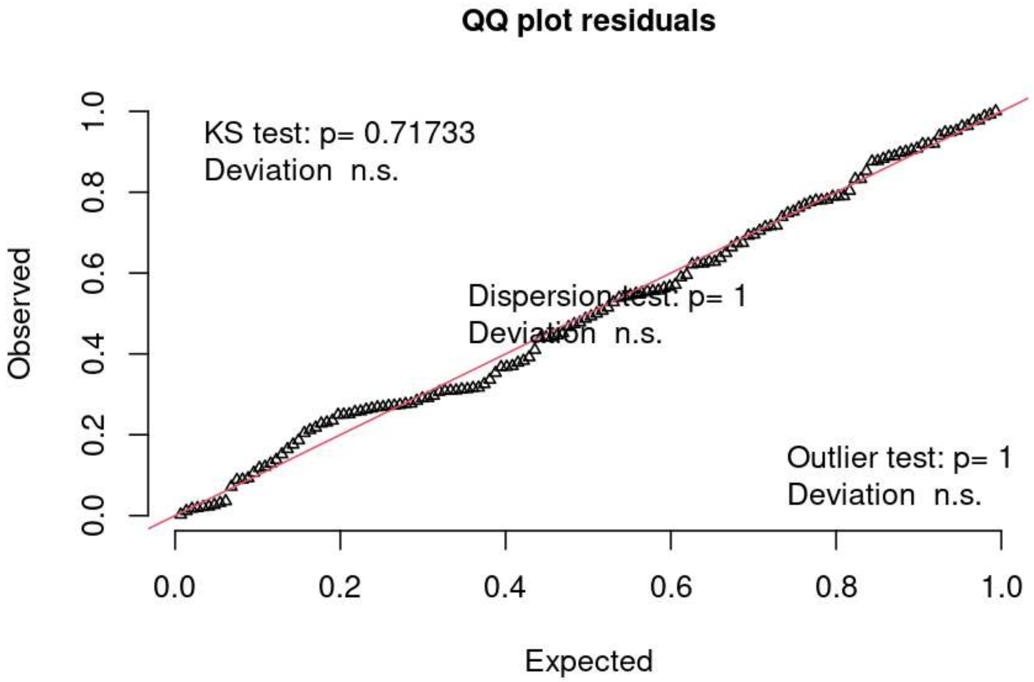
Poisson GLM

```
r <- simulateResiduals(m1)
plotQQunif(r) # Just the first Uniformity Plot
```



Negative Binomial GLM

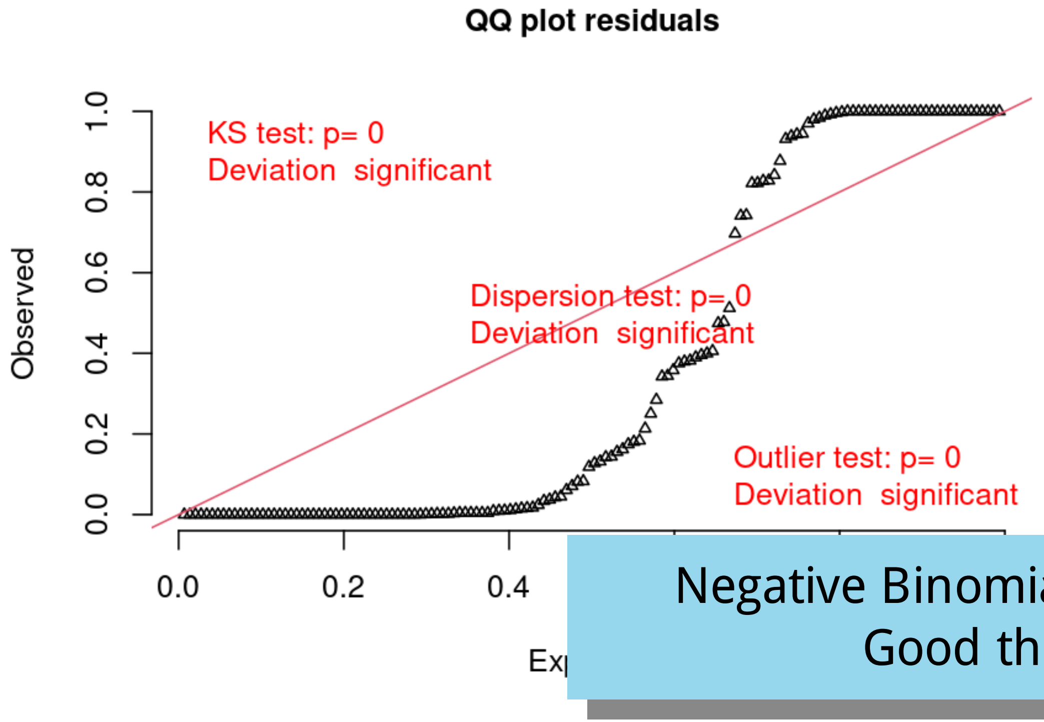
```
r <- simulateResiduals(m2)
plotQQunif(r) # Just the first Uniformity Plot
```



Overdispersed Count Data - Negative Binomial

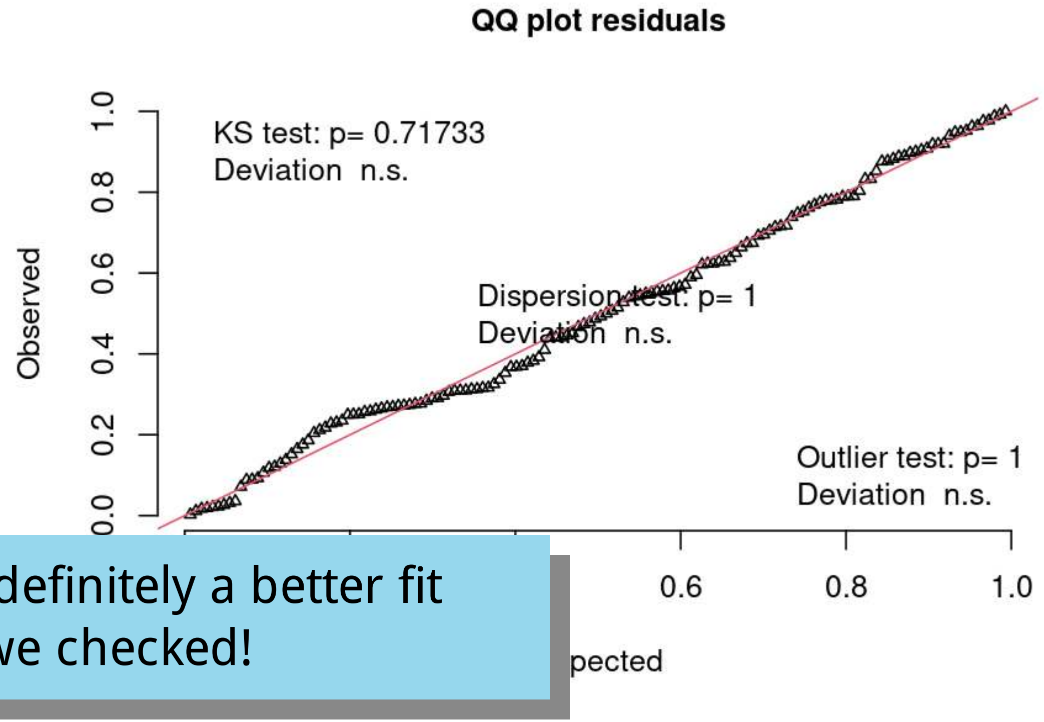
Poisson GLM

```r
r <- simulateResiduals(m1)
plotQQunif(r) # Just the first Uniformity Plot
```



Negative Binomial GLM

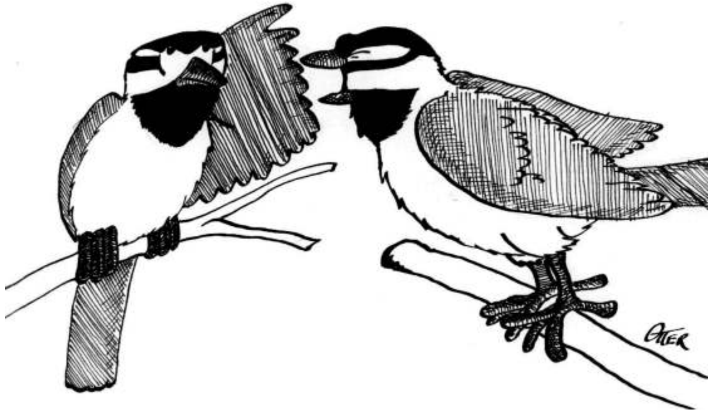
```r
r <- simulateResiduals(m2)
plotQQunif(r) # Just the first Uniformity Plot
```



Negative Binomial is definitely a better fit
Good thing we checked!

Binary Data (0/1) - Binomial Family (logistic regresion)

Get the data



- Mountain chickadees atypical songs by urbanization
- Negative **`urbanization`** more rural
- Positive **`urbanization`** more urban
- **`atypical_c`** atypical singer (1) or 'normal' singer (0)

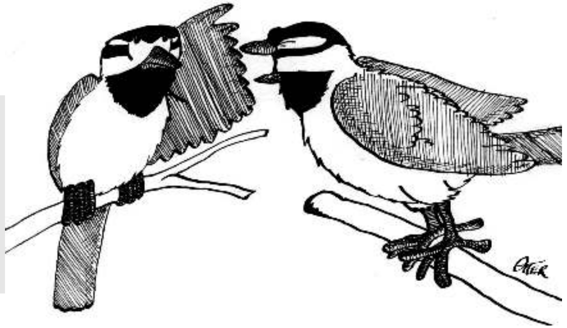
```
atypical <- read_csv("https://steffilazerte.ca/NRI_7350/data/atypical.csv")
atypical
```

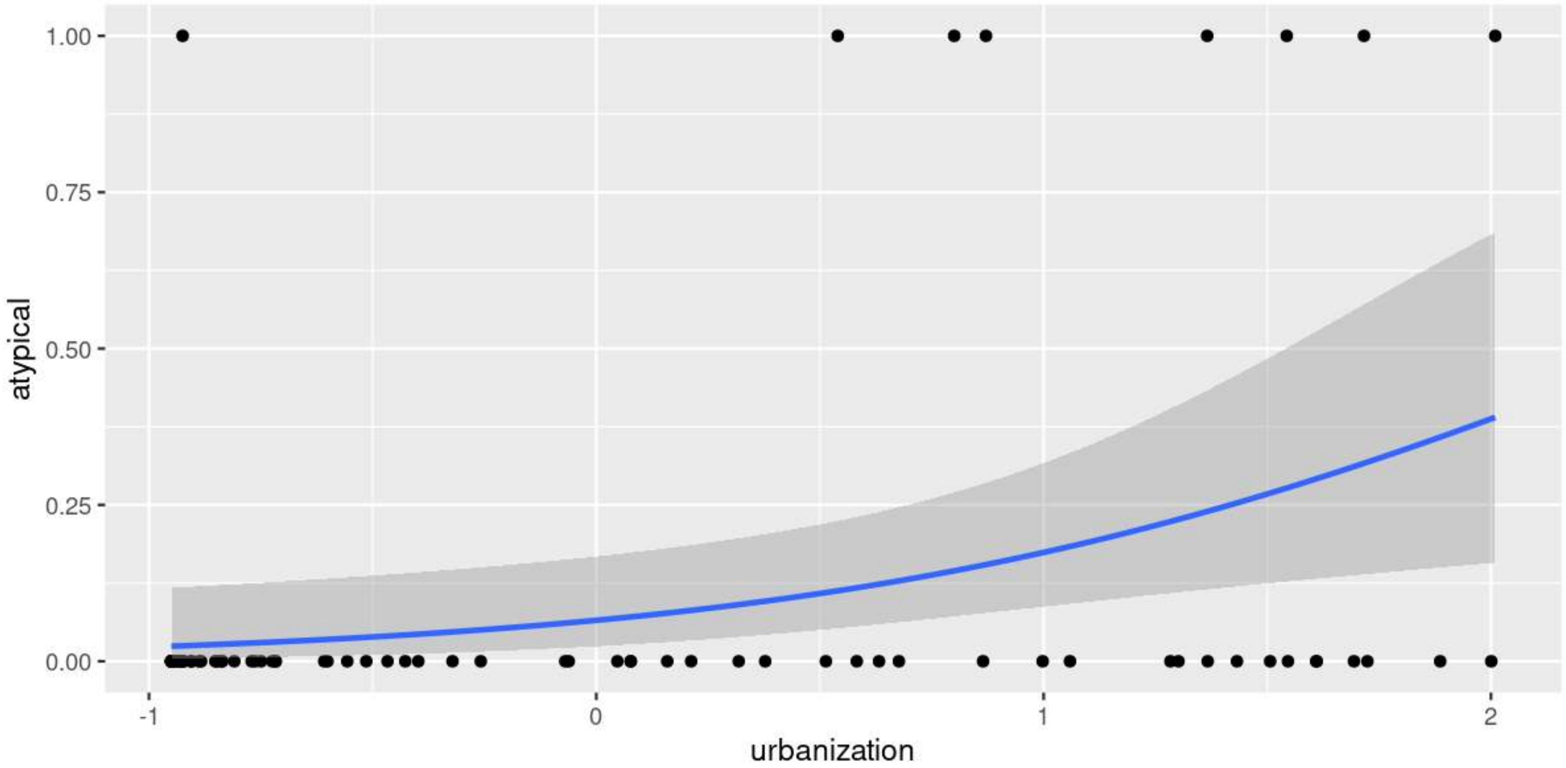
```
## # A tibble: 78 × 2
##    atypical urbanization
##       <dbl>        <dbl>
##  1        1        0.540
##  2        0        0.582
##  3        0       -0.950
##  4        0       -0.950
##  5        0        0.513
##  6        0        1.69
```

Binary Data (0/1) - Binomial Family (logistic regresion)

Look at the data

```
ggplot(data = atypical, aes(x = urbanization, y = atypical)) +
  geom_point() +
  stat_smooth(method = "glm", method.args = list(family = "binomial"))
```

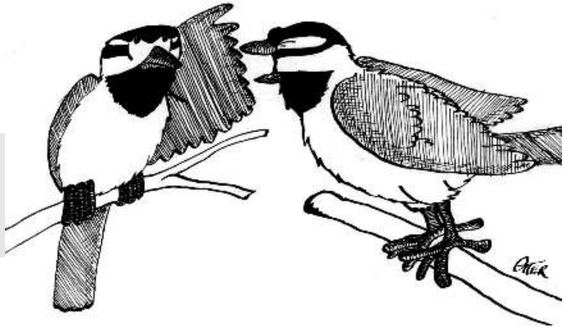


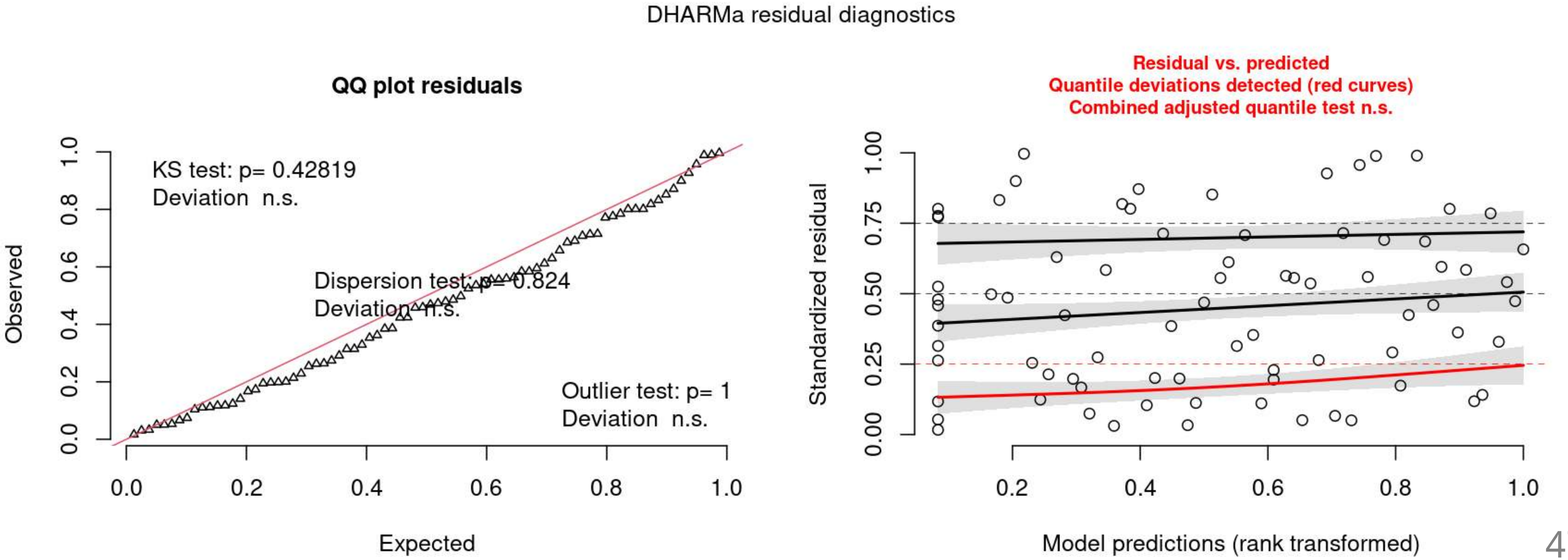


Binary Data (0/1) - Binomial Family (logistic regresion)

Run model and check diagnostics

```
m <- glm(atypical ~ urbanization, family = "binomial", data = atypical)
r <- simulateResiduals(m, plot = TRUE)
```





Binary Data (0/1) - Binomial Family (logistic regression)

```
summary(m)
```

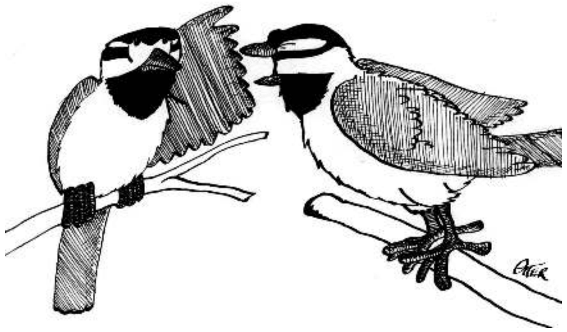
```
##
## Call:
## glm(formula = atypical ~ urbanization, family = "binomial",
data = atypical)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.9907  -0.4460  -0.2500  -0.2210   2.7201
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -2.6572     0.5380  -4.939 7.85e-07 ***
## urbanization   1.1000     0.4209   2.613  0.00897 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 51.586  on 77  degrees of freedom
## Residual deviance: 43.203  on 76  degrees of freedom
## AIC: 47.203
```

Interpreting Results

```
exp(coef(m))
```

```
##  (Intercept) urbanization
##   0.07014254   3.00406665
```

E.g., The odds of being an atypical singer increase by a factor of 3 (x3 times more likely) for every unit increase in Habit Urbanization.



Binary Outcomes - Binomial Family

Proportion with binary outcomes (e.g., 10 yes, 5 no)

Get the data

```
admissions <- as.data.frame(UCBAdmissions)
admissions <- pivot_wider(admissions,
                          names_from = Admit,
                          values_from = Freq)
admissions
```

```
## # A tibble: 12 × 4
##    Gender Dept  Admitted Rejected
##    <fct>  <fct>    <dbl>    <dbl>
##  1 Male   A          512      313
##  2 Female A           89       19
##  3 Male   B          353      207
##  4 Female B           17        8
##  5 Male   C          120      205
##  6 Female C          202      391
##  7 Male   D          138      279
```

Binary Outcomes - Binomial Family

Diagnostics

```
m <- glm(cbind(Admitted, Rejected) ~ Gender,
         family = "binomial", data = admissions)
summary(m)
```

```
## Call:
## glm(formula = cbind(Admitted, Rejected) ~ Gender, family =
"binomial",
##     data = admissions)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.22013    0.03879  -5.675 1.38e-08 ***
## GenderFemale -0.61035    0.06389  -9.553  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 877.06  on 11  degrees of freedom
## Residual deviance: 783.61  on 10  degrees of freedom
## AIC: 856.55
```

Overdispersion

Traditional check:

- Look at resid deviance (783.61) vs. df (10)

```
deviance(m) / df.residual(m)
```

```
## [1] 78.3607
```

- Very large (definitely not close to 1)

```
pchisq(deviance(m), df.residual(m),
       lower.tail = FALSE)
```

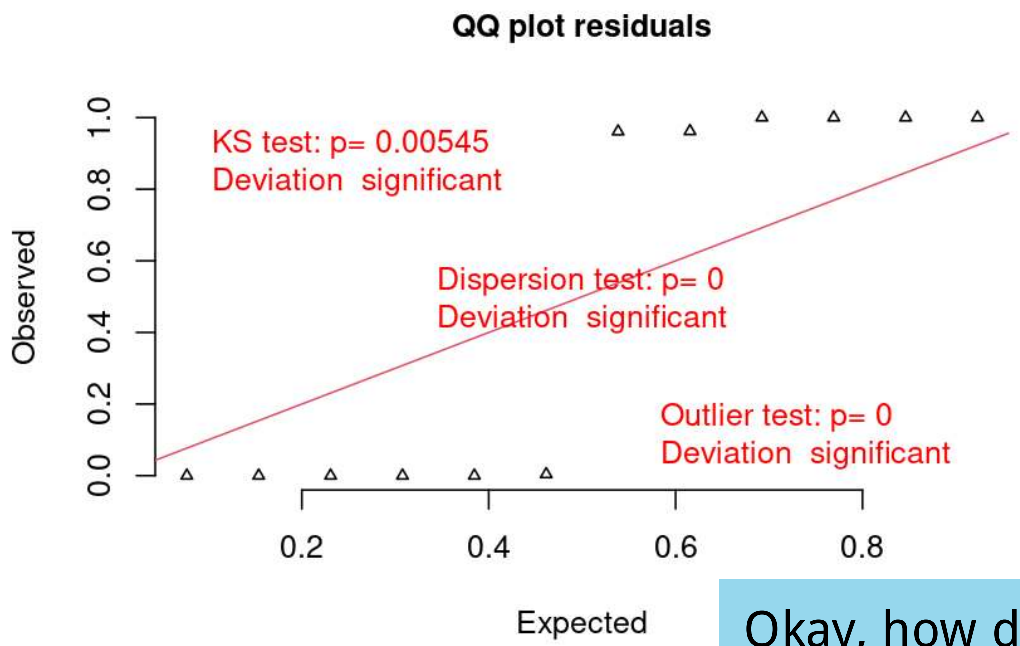
```
## [1] 6.892992e-162
```

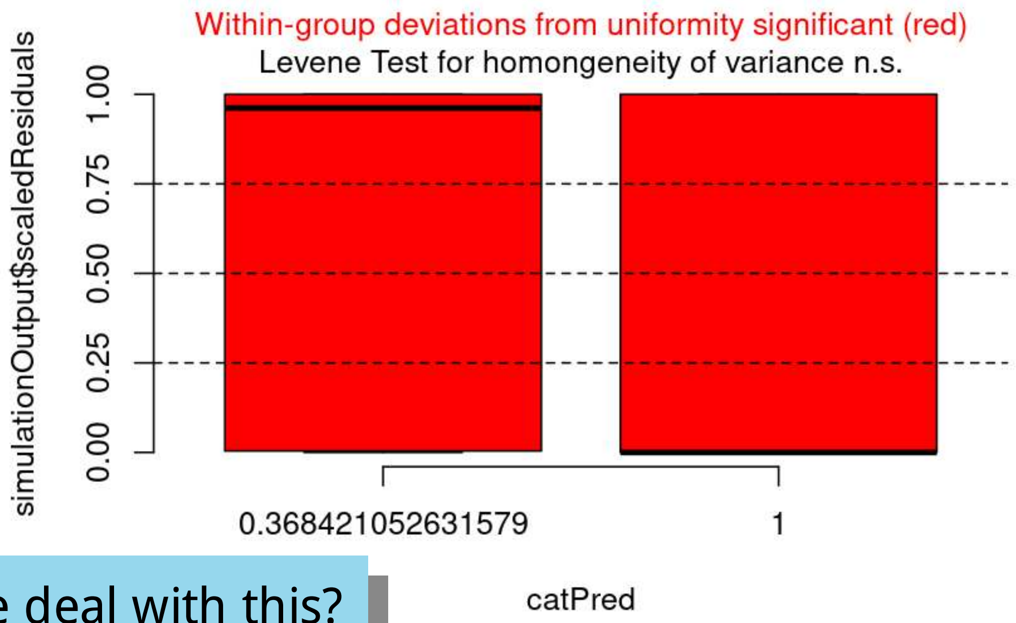
- Test shows significant overdispersion

Binary Outcomes - Binomial Family

Check with DHARMa

```r
plotQQunif(m)
plotResiduals(m, asFactor = TRUE) # to ensure Gender is treated as category
```





Okay, how do we deal with this?

Binary Outcomes - Quasi-binomial Family for overdispersion

```
m_quasi <- glm(cbind(Admitted, Rejected) ~ Gender, family = "quasibinomial", data = admissions)
summary(m_quasi)
```

```
## 
## Call:
## glm(formula = cbind(Admitted, Rejected) ~ Gender, family = "quasibinomial", 
##     data = admissions)
## 
## Deviance Residuals: 
##      Min        1Q    Median        3Q       Max  
## -16.7915   -4.7613   -0.4365    5.1025   11.2022  
## 
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -0.2201     0.3281  -0.671    0.517
## GenderFemale  -0.6104     0.5404  -1.129    0.285
## 
## (Dispersion parameter for quasibinomial family taken to be 71.52958)
## 
##     Null deviance: 877.06  on 11  degrees of freedom
## Residual deviance: 783.61  on 10  degrees of freedom
## AIC: NA
```

Binary Outcomes - Quasi-binomial Family for overdispersion

```
m_quasi <- glm(cbind(Admitted, Rejected) ~ Gender, family = "quasibinomial", data = admissions)
summary(m_quasi)
```

```
##
## Call:
## glm(formula = cbind(Admitted, Rejected) ~ Gender, family = "quasibinomial",
##     data = admissions)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -16.7915   -4.7613   -0.4365    5.1025   11.2022
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -0.2201     0.3281  -0.671    0.517
## GenderFemale  -0.6104     0.5404  -1.129    0.285
##
## (Dispersion parameter for quasibinomial family taken to be 71.52958)
##
##     Null deviance: 877.06  on 11  degrees of freedom
## Residual deviance: 783.61  on 10  degrees of freedom
## AIC: NA
```

Much more appropriate

But 'quasi' families don't always work with other functions/packages, like DHARMa!

Binary Outcomes - Mixed models for overdispersion

- Observation-level random effects (i.e assign each observation an ID number)

```r
library(lme4)
admissions <- mutate(admissions, ID = 1:n())
m_glmm <- glmer(cbind(Admitted, Rejected) ~ Gender + (1|ID), family = "binomial", data = admissions)
```

Binary Outcomes - Overdispersion

Either way, fixing overdispersion remove the 'significance'

Original

```
coef(summary(m))
```

```
##                Estimate Std. Error   z value     Pr(>|z|)
## (Intercept)  -0.2201340 0.03878810 -5.675297 1.384479e-08
## GenderFemale -0.6103524 0.06389305 -9.552720 1.263352e-21
```

Quasi fix

```
coef(summary(m_quasi))
```

```
##                Estimate Std. Error    t value
Pr(>|t|)
## (Intercept)  -0.2201340  0.3280510 -0.6710359
0.5173966
## GenderFemale -0.6103524  0.5403765 -1.1294947
0.2850545
```

GLMM fix

```
coef(summary(m_glmm))
```

```
##                Estimate Std. Error   z value
Pr(>|z|)
## (Intercept)  -0.6508845  0.4932596 -1.319558
0.1869827
## GenderFemale  0.1747094  0.7016048  0.249014
0.8033499
```

Your Turn!

We have the **`crabs`** dataset

```
crabs <-
read_csv("https://steffilazerte.ca/NRI_7350/data/crabs.csv")
```

Background

- Horseshoe crabs form pairs for spawning (mating)
- But extra, unattached males crowd around and try to get involved (Satellite males)

Your Job

You're interested in the effect of female size (**`width`**) on the number of male **`satellites`**

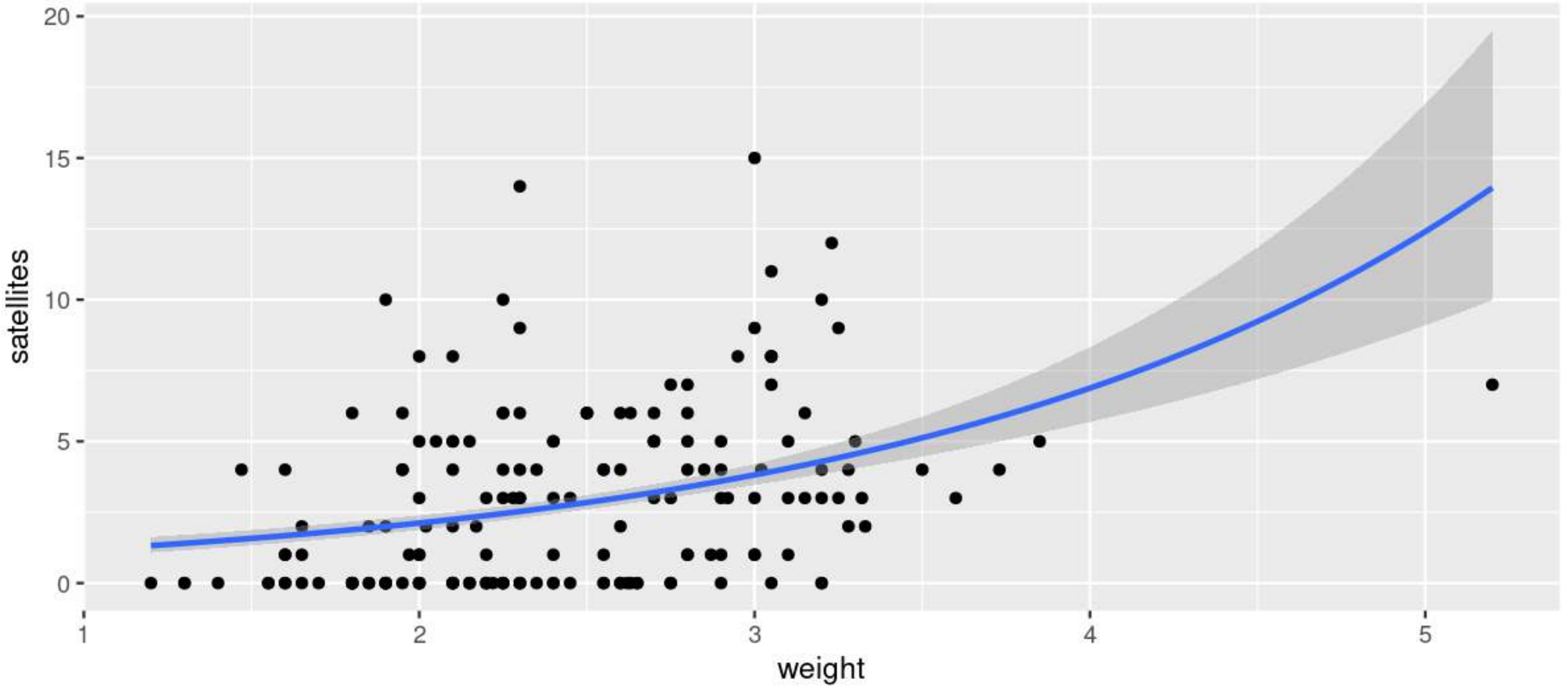
- Look at your data (make a plot)
- Run a **`glm()`** for count data
- Check your diagnostics. Do you have a problem? Check for overdispersion and zero-inflation
- Apply an overdispersion fix
- Check your diagnostics. Do you have a problem? Check for zero-inflation

Brockmann HJ (1996). “Satellite Male Groups in Horseshoe Crabs, *Limulus polyphemus*”, Ethology, 102(1), 1–21.

Your Turn!

Look at your data

```
ggplot(data = crabs, aes(x = weight, y = satellites)) +
  geom_point() +
  stat_smooth(method = "glm", method.args = list(family = "poisson"))
```



Your Turn!

Run a `glm()` for count data

```
m <- glm(satellites ~ weight, data = crabs, family = "poisson")
summary(m)
```

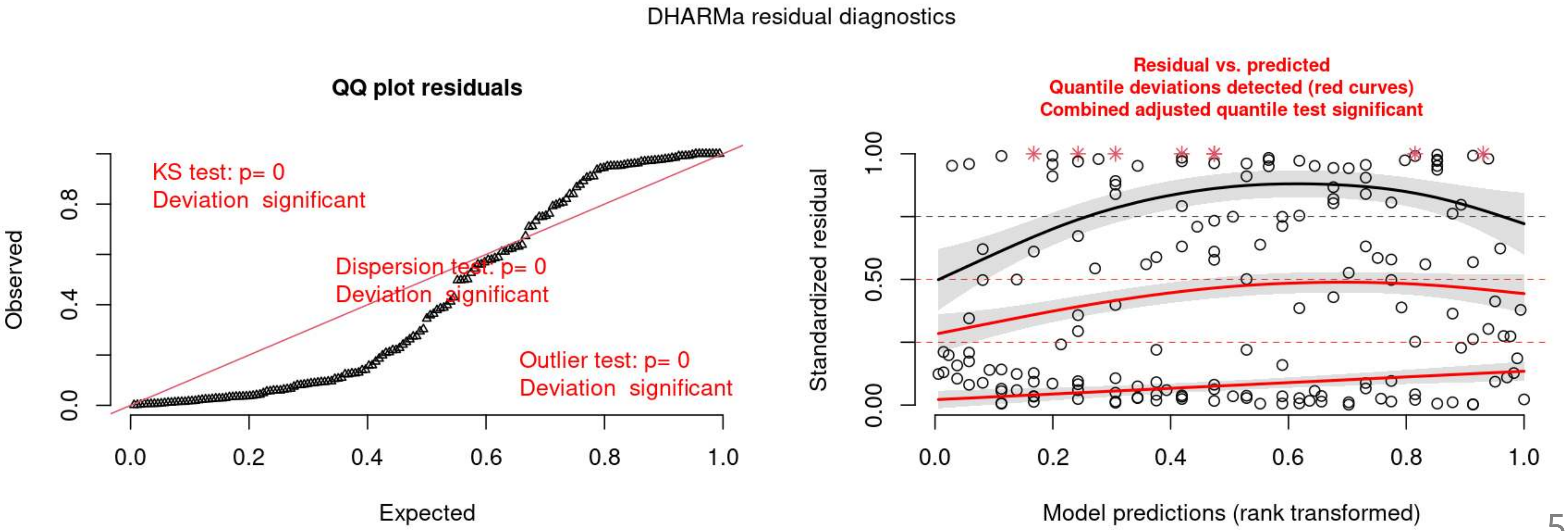
```
##
## Call:
## glm(formula = satellites ~ weight, family = "poisson", data = crabs)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9306  -1.9981  -0.5627   0.9299   4.9992
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.4282     0.1789  -2.394   0.0167 *
## weight        0.5892     0.0650   9.065   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 632.79  on 172  degrees of freedom
```

Your Turn!

Check your diagnostics

- Do you have a problem? Check for overdispersion and zero-inflation

```
r <- simulateResiduals(m, plot = TRUE)
```



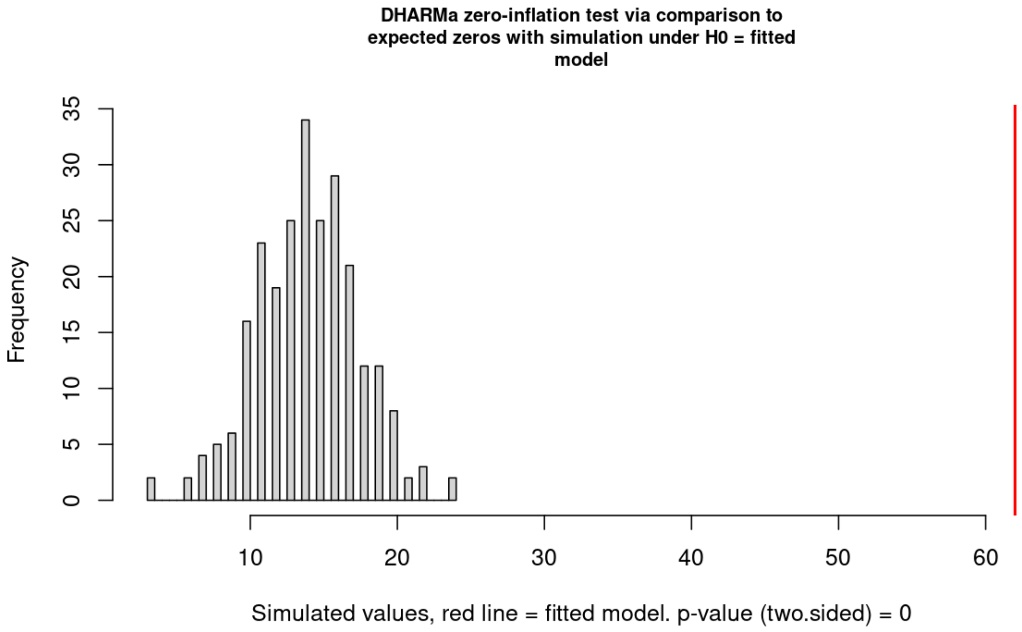
Your Turn!

Check your diagnostics

- Do you have a problem? Check for overdispersion and zero-inflation

```
testZeroInflation(m)
```

```
##
##      DHARMa zero-inflation test via comparison to
expected zeros with simulation under H0 =
##      fitted model
##
## data:  simulationOutput
## ratioObsSim = 4.381, p-value < 2.2e-16
## alternative hypothesis: two.sided
```



Your Turn!

Apply an overdispersion fix

```
m <- MASS::glm.nb(satellites ~ weight, data = crabs)
summary(m)
```

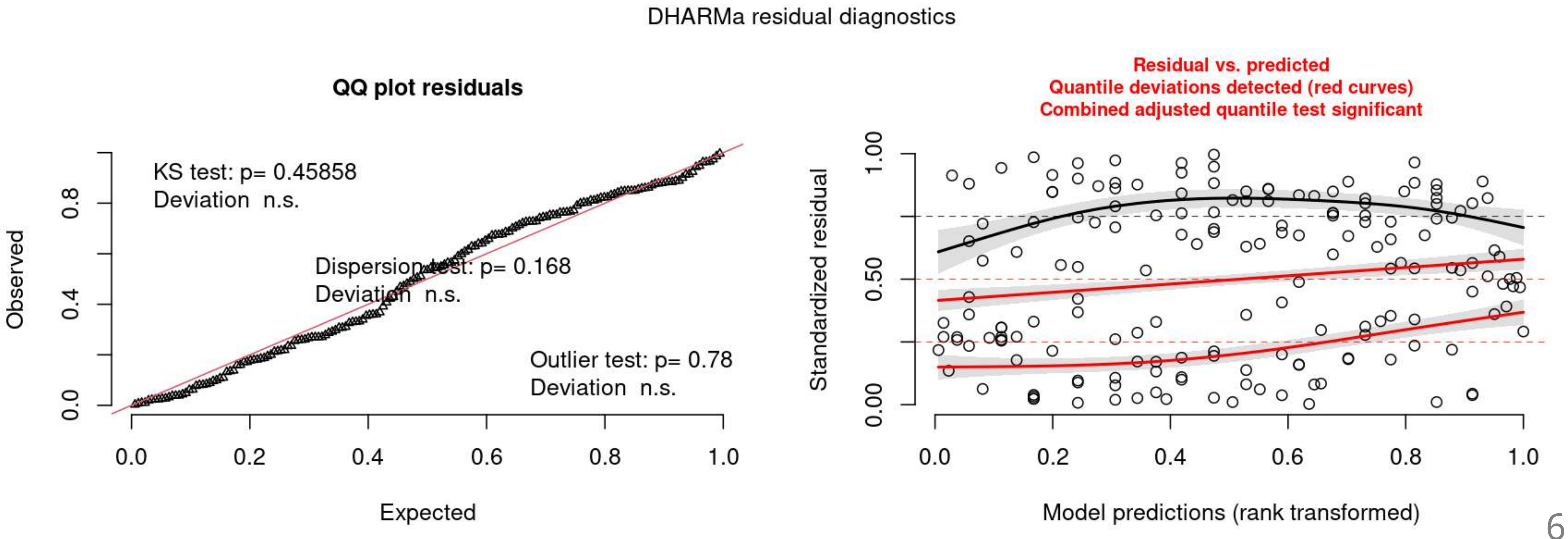
```
##
## Call:
## MASS::glm.nb(formula = satellites ~ weight, data = crabs, init.theta = 0.9310998551,
##     link = log)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8393  -1.4123  -0.3246   0.4747   2.1278
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.8637     0.4046  -2.135   0.0328 *
## weight        0.7599     0.1578   4.817 1.46e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(0.9311) family taken to be 1)
##
```

Your Turn!

Check your diagnostics

- Do you still have a problem? Check for overdispersion and zero-inflation

```r
r <- simulateResiduals(m, plot = TRUE)
```



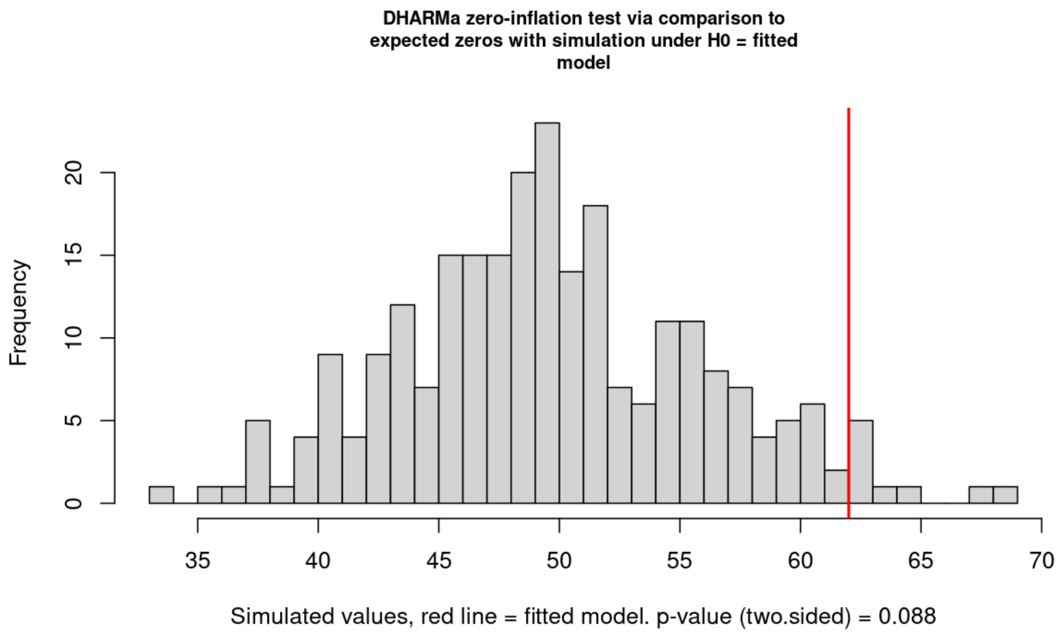
Your Turn!

Check your diagnostics

- Do you still have a problem? Check for overdispersion and zero-inflation

```
testZeroInflation(m)
```

```
##
##      DHARMa zero-inflation test via comparison to
expected zeros with simulation under H0 =
##      fitted model
##
## data:  simulationOutput
## ratioObsSim = 1.2358, p-value = 0.088
## alternative hypothesis: two.sided
```



Not bad...
Can we do better?

Zero-inflated Models (Advanced example! Above and beyond!)

- **`glmmTMB()`** function from **`glmmTMB`** package
- Allows modeling zero-inflation (**`ziformula`** says zeros are a function of the variable **`weight`**)

```
crabs <- read_csv("https://steffilazerte.ca/NRI_7350/data/crabs.csv")
library(glmmTMB)
m <- glmmTMB(satellites ~ weight, ziformula = ~ weight , family = "nbinom2", data = crabs)
summary(m)
```

```
## Dispersion parameter for nbinom2 family (): 4.96
##
## Conditional model:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.8979     0.3051   2.943  0.00325 **
## weight        0.2171     0.1118   1.942  0.05217 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Zero-inflation model:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   3.7546     0.9837   3.817 0.000135 ***
## weight       -1.9123     0.4320  -4.426 9.59e-06 ***
## ---
```

Zero-inflated Models (Advanced example! Above and beyond!)

- **`glmmTMB()`** function from **`glmmTMB`** package
- Allows modeling zero-inflation (**`ziformula`** says zeros are a function of the variable **`weight`**)

```
crabs <- read_csv("https://steffilazerte.ca/NRI_7350/data/crabs.csv")
library(glmmTMB)
m <- glmmTMB(satellites ~ weight, ziformula = ~ weight , family = "nbinom2", data = crabs)
summary(m)
```

```
## Dispersion parameter for nbinom2 family (): 4.96
##
## Conditional model:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.8979     0.3051   2.943  0.00325 **
## weight        0.2171     0.1118   1.942  0.05217 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
##
## Zero-inflation model:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   3.7546     0.9837   3.817 0.000135 ***
## weight       -1.9123     0.4320  -4.426 9.59e-06 ***
## ---
```

"Regular" effects
Heavier females have more satellite males
(P = 0.052)

Zero-inflated Models (Advanced example! Above and beyond!)

- **`glmmTMB()`** function from **`glmmTMB`** package
- Allows modeling zero-inflation (**`ziformula`** says zeros are a function of the variable **`weight`**)

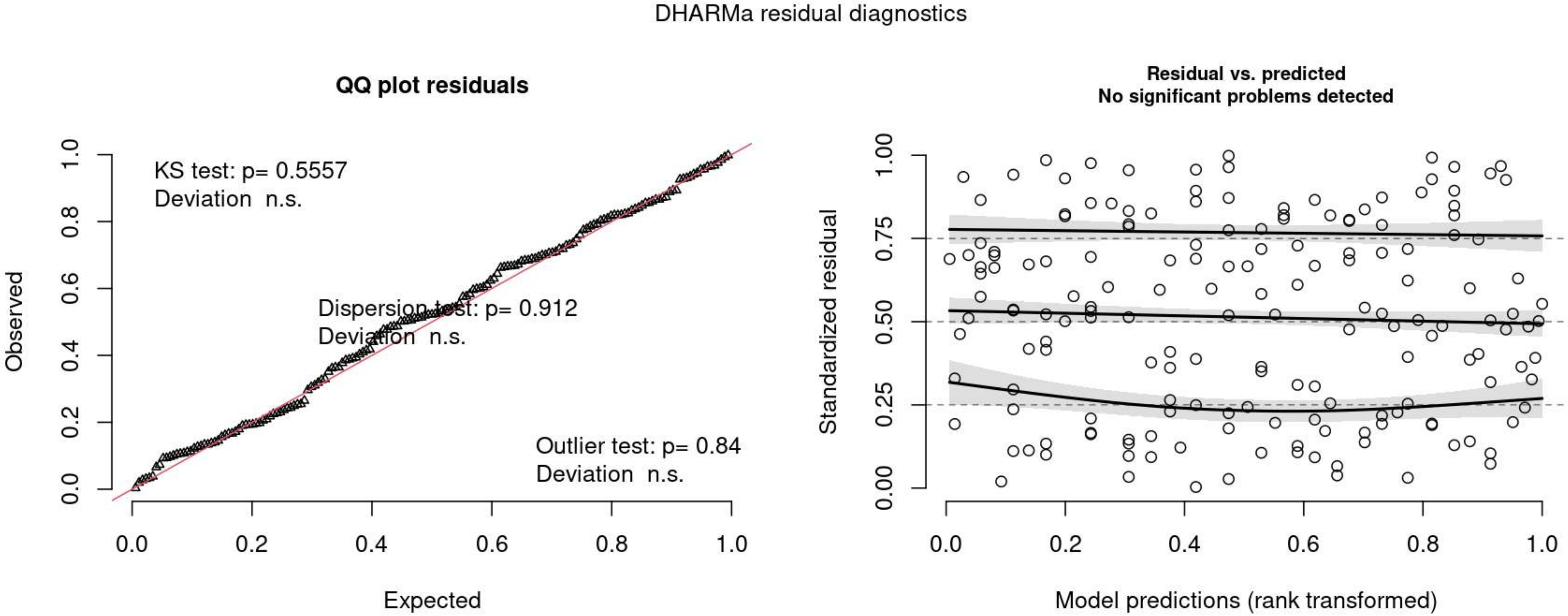
```r
crabs <- read_csv("https://steffilazerte.ca/NRI_7350/data/crabs.csv")
library(glmmTMB)
m <- glmmTMB(satellites ~ weight, ziformula = ~ weight , family = "nbinom2", data = crabs)
summary(m)
```

```
## Dispersion parameter for nbinom2 family (): 4.96
##
## Conditional model:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   0.8979     0.3051   2.943  0.00325 **
## weight        0.2171     0.1118   1.942  0.05217 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
##
## Zero-inflation model:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   3.7546     0.9837   3.817 0.000135 ***
## weight       -1.9123     0.4320  -4.426 9.59e-06 ***
## ---
```

"Zero-inflation" effects
more zeros counts as weight decreases
What we expected!

Zero-inflated Models (Advanced example! Above and beyond!)

```
r <- simulateResiduals(m, plot = TRUE)
```

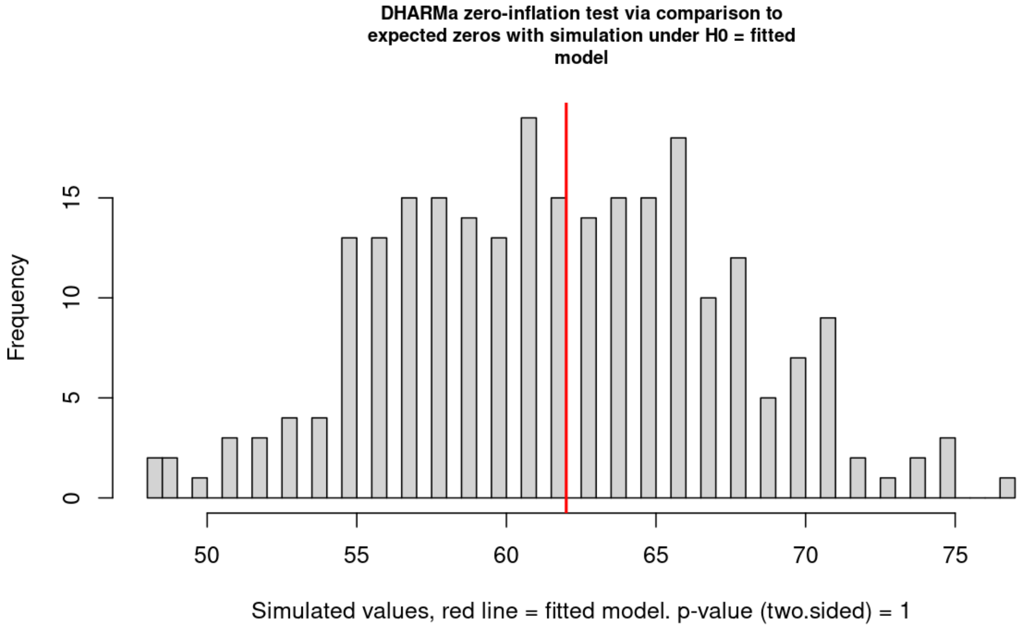


For more information on the glmmTMB package see their Journal Article

Zero-inflated Models (Advanced example! Above and beyond!)

```
testZeroInflation(m)
```

```
##
##      DHARMa zero-inflation test via comparison to
expected zeros with simulation under H0 =
##      fitted model
##
## data:  simulationOutput
## ratioObsSim = 1.0016, p-value = 1
## alternative hypothesis: two.sided
```



For more information on the glmmTMB package see their [Journal Article]

Packages and References for Other Advanced Models

Packages and References

(Generalized) Linear Mixed Models (LMM, GLMM)

- Also called generalized linear mixed effects models (GLME, LME)
- `lme4` - More advanced, crossed-random factors, Generalized (`glmer()`) and Gaussian (`lmer()`)
- `nlme` - Older but can specify auto-correlation structures, only Gaussian (`lme()`)
- `glmmTMB` - Zero-inflated models and other distribtions

References

- Ben Bolker's GLMM FAQ
- Chapter 19, "The R Book" by Michael J. Crawley

  (Freely available online through University of Manitoba Library)
- Mixed Effects Models and Extensions in Ecology with R by Alain Zuur

  (Freely available online through University of Manitoba Library)
- Generalized linear mixed models: a practical guide for ecology and evolution, 2009, Trends in ecology and evolution

Packages and References

General Additive Models (GAM)

- **`mgcv`** package (**`gam()`**, **`gamm()`**)
- **`gamm4`** package (**`gamm4()`**)

References

- Chapter 19, "The R Book" by Michael J. Crawley

  (Freely available online through University of Manitoba Library)
- "Generalized Additive Models: An Introduction with R" by Simon N. Wood

  (Hard-copy available from University of Manitoba Library)

Packages and References

Generalized Estimating Equations (GEE)

- **`gee`** package (**`gee()`**)
- **`geepack`** package (**`geeglm()`**)

References

- The R package geepack for Generalized Estimating equations, Journal of Statistical Software, 2005
- geepack Manual

Non-parametric Statistics

Non-parametric Statistics

Wilcoxon Rank Sum (Mann-Whitney) Test

```
air <- filter(airquality, Month %in% c(5, 8))
```

Is there a difference in air quality between May (5th month) and August (8th month)?

```
wilcox.test(Ozone ~ Month, data = air, exact = FALSE)
```

```
##
##      Wilcoxon rank sum test with continuity correction
##
## data:  Ozone by Month
## W = 127.5, p-value = 0.0001208
## alternative hypothesis: true location shift is not equal to 0
```

Yes!

Non-parametric Statistics

Kruskal-Wallis Rank Sum Test

Is there a difference in air quality among months?

```
kruskal.test(Ozone ~ Month, data = airquality)
```

```
##
##      Kruskal-Wallis rank sum test
##
## data:  Ozone by Month
## Kruskal-Wallis chi-squared = 29.267, df = 4, p-value = 6.901e-06
```

Yes, there is at least one month that is different from the rest.