GLM Tutorial

#### Ana Miller-ter Kuile and Tatum Katz

#### 7/21/2020

Some useful formatting and visualization packages for this R Markdown:

library(ggplot2) #plotting  
library(tidyverse) #data tidying power  
library(gt) #tables  
library(here)

Learning goals

1. See and understand the **math of GLMs**
2. Know that different kinds of ecological data require different **GLM distributions** (with a nifty table as a resource)
3. Know how to **fit a GLM in R** and diagnose how well data fit the selected model distribution
4. Feel less intimidated by statistics! Specifically, remember: there are **multiple ways to approach statistics** and all of them are a little wrong, statistics is a tool to **bolster ecological questions**, and as long as you are **consistent** and **transparent** about your process (accessible data and code), you’ll be fine

A. GLM Math

Part A. Foundations

**Largely adapted from Dr. Wendy Meiring’s PSTAT220B course**

1. Motivation for extending the Linear Regression Model

Key assumptions in linear models are:

1. linear dependence of the Y on unkown parameters

2. additive, random error

3. errors are Gaussian (normal)

4. specific mean-variance relationship (Gaussian)

When do these assumptions break down? Proportion data, count data - i.e. MANY common ecological systems.

The “first step” of extending the linear model to be more flexible is to move to *generalized linear models*.

2. What IS a GLM?

All generalized linear models rely on the *natural exponential family of distributions*. Gaussian, Binomial, and Poisson distributions are special cases of the exponential family which assumes a general density function form for the Y:

\[ f(y\_i | \theta\_i, \phi) = exp \left\{ \frac{y\theta\_i - b(\theta\_i)}{a\_i(\phi)} + c(y\_i, \phi) \right\} \]

where \(\theta\) is the **canonical parameter** and \(\phi\) is the **dispersion parameter**, \(i\) ranges from \(1,...,n\), and \(n\) is your sample size. *Importantly, note which functions and parameters are specific to each observation, and which are true over the whole distribution*. This is some of where the GLM flexibility comes in!

You can think of the **canonical parameter** like a weird, flexible, GLM version of the Gaussian mean, and the **dispersion parameter** like a weird, flexible, GLM version of the Gaussian variance.

All members of the exponential family of distributions’ PDFs (probability density functions) can be re-written in the above form. In the Gaussian special case, \(\theta\_i\) is the mean and \(\phi\) is the variance.

A GLM has three components; the chosen distribution is just one of them!

1. **Random component**: specifies the distribution of the response variables \(Y\_i\) for likelihood computation. In GLM, the distributions must be from the exponential family, and the \(Y\_i\)’s are assumed independent with \(E(Y\_i | x\_i) = \mu\_i\) for the explanatory variables \(x\_i\).
2. **Systematic component**: specifies the linear function of \(\beta\_p\)’s that is a function of the explanatory variables. This is the traditional “right hand side” of the equation of a linear model.
3. **Link function**: specifies how the **mean of the random component** *relates* to the **systematic component**. You can think of this like a transformation which puts the right hand side of the equation on the same scale as the \(Y\_i\)’s. It is important to note however that the link function \(g(.)\) operates on the **mean** \(\mu\_i\), NOT the response \(Y\_i\).

Here is how you can write the Gaussian linear model in GLM form:

1. **Random component**  
   \[Y\_i \stackrel{independent}{\sim} N(\mu\_i, \sigma^2)\]
2. **Systematic component**  
   \[\eta\_i = \sum^{p-1}\_{j=0} \beta\_jx\_{ij}\]
3. **Link function**  
   \[g(\mu\_i) = \eta\_i \texttt{, where g(.) is the identity function.}\]

3. Estimating and comparing GLMs

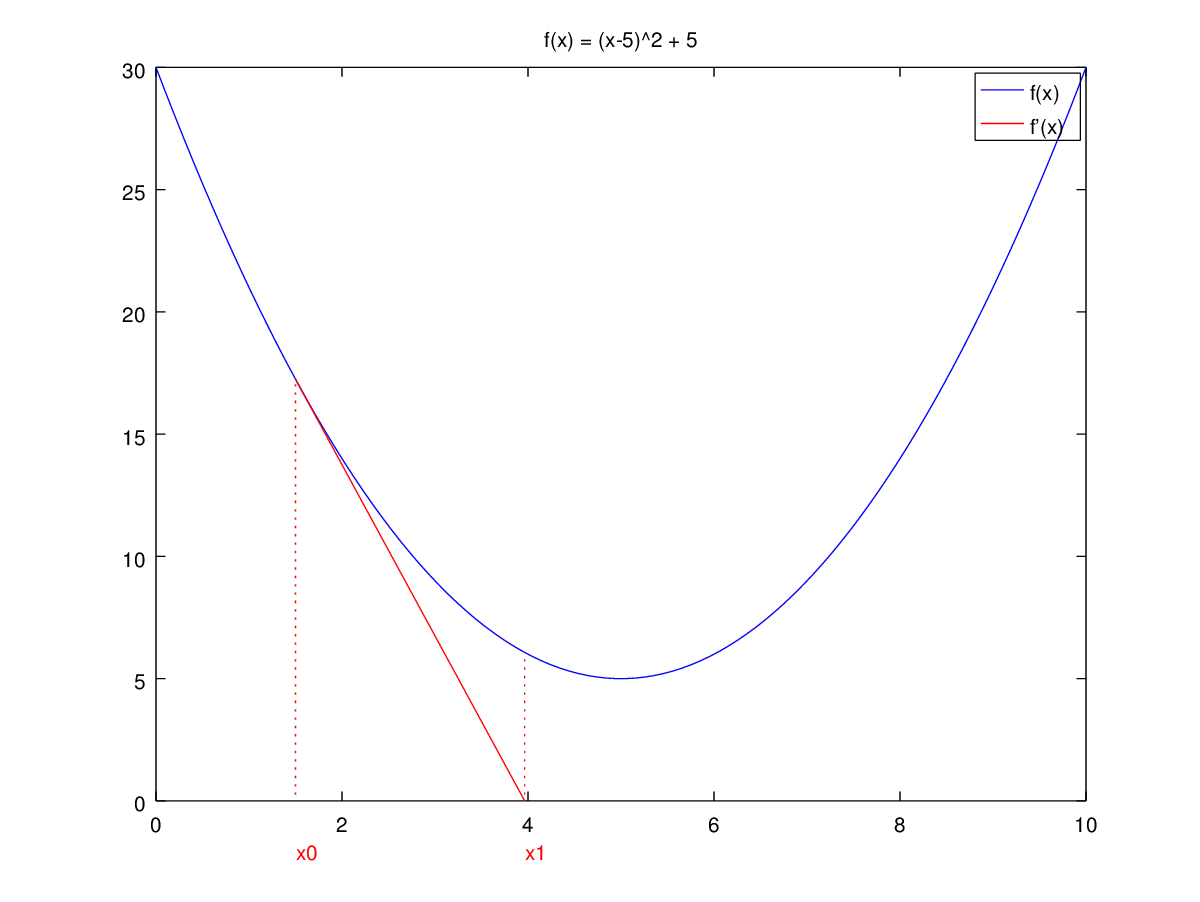
A. Solving for **\(\beta\)**’s

Remember how we can have a different function \(a\_i(\phi)\) for each observation? \(a\_i(\phi)\) typically has one of two forms:

1. Constant dispersion: \(a\_i(\phi)\) is independent of \(i\), and we give equal weight to each observation.
2. Unequal weights: \(a\_i(\phi) = \frac{\phi}{w\_i}\), where \(w\_i\) is unknown. This weights each observation differently, in terms of how much each observation influences estimation of the \(\beta\)s.

Following, GLM’s can have weird mean-variance relationships. In the Gaussian, the mean and variance are independent, but in many GLMs they depend on one another (think Poisson, for which both the mean and the variance is \(\lambda\)). Therefore, we cannot analytically solve for the traditional MLE, WLS, OLS estimations. The problem is that we wind up with a non-linear (self-referential) system of equations that cannot be solved analytically!

INSTEAD, we estimate the \(\beta\)s numerically using **iteration**. Remember the **Newton-Raphson algorithm**? This is (basically) what glm() and most other functions which estimate the \(\beta\)s use, and this is how you can solve for your \(\beta\)s by hand (but extremely not recommended, talk to Tatum if you want references).



If you ever got some warning message like “algorithm did not converge” after trying to run a glm() etc., that means that it wasn’t able to converge on a single estimate for your \(\beta\)s, or even any estimate at all! The NRA isn’t the best, or the fastest, or the most reliable approach, even, but it usually works. You can use other methods if you like, but that is an entirely different discussion.

B. Model selection for GLMs

Consider two extreme models:

1. **Null model**: \(\mu\) is a constant, i.e. \(Y\) is predicted only by an intercept. Any variations in the observations are due to the random component alone. Usually, this is far too simple.
2. **Saturated model**: the systematic component has \(n\) parameters (\(n\) is still the number of observations!). All variations in the observations are captured by the systematic component. Because this model completely reconstructs the data, it isn’t informative or useful.

To compare models against one another, you can use the **deviance**, \(D\_M\). This can be simply calculated with many R functions.

In practice, comparing nested models is simple. Let’s develop a formal hypothesis test! To test if a model \(M\) with \(p\) parameters can be reduced to a sub-model \(M\_0\) with \(q < p\) parameters, we can use the likelihood ratio statistic:

\[\frac{D\_{M\_0} - D\_M}{\phi} \sim X^2\_{p-q} \texttt{ if } H\_0 \texttt{: model }M\_0\texttt{ is true}\] But, this only works when \(\phi\) is known and fixed, such as in the Poisson or Binomial. These hypotheses can be stacked together to create an *Analysis of Deviance* table (Anode???).

We can also still use AIC and BIC (nested or otherwise), and instead of two-times the log-likelihood, you can just sub out the deviance!

If \(\phi\) is unknown, we can estimate it, and then compare models using **an approximate F test** instead:

\[ \frac{D\_{M\_0} - D\_M}{\hat{\phi}(p-q)} \sim F\_{(p-q),(n-p)} \texttt{ if } H\_0 \texttt{: model }M\_0\texttt{ is true} \]

C. Residuals in GLM world

There are two types of departures from a specified GLM:

1. Systematic departure: assumptions about any one of the three components can be wrong (could have chosen the wrong distribution, data are correlated, missing important covariates, or choosing the wrong link function), or you have over-dispersion (see next topic).
2. Isolated departure: you got outliers, my dude

Residual plots do a great job of finding model issues in linear regression world, but are more iffy in GLM world…still, they are the best we often have.

There are MANY types of residuals for GLM, where in LM you only have the one type. Here are a few important ones, all of which can be computed using R functions:

1. **Response residuals**: \(y\_i - \hat{\mu\_i}\). These are *not useful* for non-Gaussian data since they have different variances!!!
2. **Partial residuals**: \((y\_i - \hat{\mu\_i})g'(\hat{\mu\_i}) + \hat{\beta\_j}x\_{ij}\). For each fixed \(j\), the plot of these residuals vs. \(x\_{ij}\) should be a line if the model is correct. A non-linear pattern suggests a higher order term or transformation.
3. **Deviance residuals**: \(\texttt{sign}(y\_i - \hat{\mu\_i})\sqrt{d\_i}\). Commonly used for diagnostics.
4. **Pearson residuals**: \(\frac{y\_i - \hat{\mu\_i}}{\sqrt{V(\hat{\mu\_i})}}\) where \(V(\hat{\mu\_i})\) is the variance function for your chosen distribution. These are the response residuals scaled by the square root of the variance function.

In the Gaussian case, all of the above residuals are identical to the response residuals!

As with LM, we can also calculate leverage and influence for each point by simply extending the LM equations.

To check for an **isolated depature**, look for high values for any of the residuals, or high leverage of influence values.

To check for a **systematic departure**, you may make plots of the **residuals vs. predictions** (should have mean zero and constant range; curvature arises from anything from wrong choice of link or systematic component; *do not use for binary data*, since it is uninformative), the **residuals vs. covariates** (systematic trend may indicate wrong choice of link and/or missing covariates), or the **partial residuals vs. covariates** (to check scale; the null pattern is approx. a straight line).

D. Under- and over-dispersion

A sign things in your model, somewhere, somehow, have gone wrong:

1. **Over-dispersion**: more common of the two, occurs when the variance of the response variable exceeds the nomial value, i.e., \(Var(Y) > a(\phi)b''(\theta)\).
2. **Under-dispersion**: far less common, when the variance of the response variable falls short of the nomial value, i.e., \(Var(Y) < a(\phi)b''(\theta)\).

These definitions come from the *moment-generating functions*; you can easily find what the variance should be for any distribution by checking Wikipedia for your distribution and finding the Variance!

A basic approach is to compare your estimated \(\phi\) value with the nomial values (which are 1 for Binomial and Poisson). Large deviations are bad. Use this with ~extreme caution~.

**Clustering** is a common cause of over-dispersion. This can happen easily in ecological studies; let’s say you sample 10 salamanders from each of 10 ponds. Then you have clustering!

If you do have over-dispersion, you can often model the extra variation by converting Binomial models to Beta-Binomial models and converting Poisson models to Negative-Binomial models, or by moving on to **generalized linear mixed effects models**. You could also move to **quasi-likelihood** models, but these ignore the underlying mechanism and so might be useless for many ecology problems.

E. Poisson Rate Models - fishing example

Suppose \(Y\_i\) is the number of fish caught by a single trawl \(i\), which catches the fish in a volume \(vol\_i\) of water. Different trawls can pass their nets through different volumes of water. We want to use the collected data to estimate the expected “rate” of fish ber unit of water volume, i.e., we want to find \(E \left[ \frac{Y\_i}{vol\_i} \right]\).

This is a common ecological problem - the idea of search effort!

Using some algebra, we can identify a workable model:

Suppose \(Y\_i \stackrel{independent}{\sim} Poisson(\mu\_i)\), so that \(E[Y\_i] = \mu\_i\).

Then, \(ln \left( \frac{E[Y\_i]}{vol\_i} \right) = ln \left( \frac{\mu\_i}{vol\_i} \right) = ln(\mu\_i) - ln(vol\_i)\)

To turn it into a model now, \(ln \left( \frac{\mu\_i}{vol\_i} \right) = \underline{x}^T\_i\underline{\beta}\).

Based on the above, this is equivalent to modeling \(ln(\mu\_i) = ln(vol\_i) + \underline{x}^T\_i\underline{\beta}\)! You can include the \(ln(vol\_i)\) term in your glm() using offset().

B. Common Distributions

There are lots of different data distributions for GLMs for ecological data. The Zuur book has a great introduction to many of these and the [Kyle Edwards notes](https://sites.google.com/site/kyleedwardsresearch/lecture-notes?authuser=0) (Lectures 6-13) have great succinct explanations and worked examples. Many of these I (Ana) don’t actually use but there are a few that are really common in ecology which I use all the time.

You can find the model families for a given GLM-calling function in R by assessing the family object for that function. I usually get there by typing ?glm or ?glmmTMB and then selecting either the family or the family\_glmmTMB links from those help pages when they pop up.

We compiled a list of common GLM families, what kinds of data you might fit with them, packages that have that model family, some extra juicy math for each family, and where you might find more information on each of them in Kyle Edwards’ notes, or on the interwebs. You can find complementary information on page 205 of the Zuur book of in Kyle Edwards’ Lecture 11.

|  |  |  |  |
| --- | --- | --- | --- |
| GLM Distributions | | | |
| Find your data type for model families, R packages, resources, and math! | | | |
| **Distribution** | **Common Ecological Data** | **Packages** | **Resources** |
| ***Data type: Binary or proportional*** | | | |
| Binomial | Binary or proportional (e.g. presence-absence, percents) | glm, glmmTMB | Kyle Edwards lectures 8, 9, 11 |
| Quasi-binomial | Binary or proportional | glmmTMB | Kyle Edwards lectures 8, 9, 11 |
| Beta or Beta-family | Proportions or percents that are not ones or zeros | betareg, glmmTMB | Kyle Edwards lecture 11 |
| ***Data type: Count or abundance*** | | | |
| Poisson | Count or abundance data | stats, glmmTMB | Kyle Edwards lecture 6, 8, 10, 11 |
| Negative Binomial | Count or abundance data with overdispersion | MASS, glmmTMB | Kyle Edwards lecture 6, 8, 10, 11 |
| Quasi-Poisson | Count or abundance data with overdispersion | glmmTMB | Kyle Edwards lecture 6, 8, 10, 11 |
| Zero-Inflated Poisson | Count or abundance data with too many zeros | glmmTMB | Kyle Edwards lecture 6, 8, 10, 11 |
| Zero-Inflated Negative Binomial | Count or abundance data with too many zeros and overdispersion | glmmTMB | Kyle Edwards lecture 6, 8, 10, 11 |
| Truncated Poisson | Count or abundance data that won't have zeros (e.g. only counted when they were present) | glmmTMB | Kyle Edwards lecture 6, 8, 10, 11 |
| Truncated Negative Binomial | Count or abundance data that won't have zeros and that is overdispersed | glmmTMB | Kyle Edwards lecture 6, 8, 10, 11 |
| Poisson or negative binomial families with an offset term | Density data, Count or abundance data with unequal sample sizes | stats, MASS, glmmTMB | Kyle Edwards lecture 9, 11 |
| ***Data type: Continuous*** | | | |
| Gaussian | Normal continuous data (e.g. body measurements) | stats, glmmTMB | Kyle Edwards lecture 1, 11 |
| Gamma | Continuous positive data | stats, glmmTMB | Kyle Edwards lecture 11 |
| Tweedie | Continuous data derived from count data (e.g. tree biomasses) that will have too many zeros | glmmTMB | Kyle Edwards lecture 11, https://doi.org/10.1186/s13021-016-0051-z |
|  | | | |
| Weibull | survival/failure analysis, wind speed distributions | dweibull | https://www.r-bloggers.com/survival-analysis-fitting-weibull-models-for-improving-device-reliability-in-r/ |
| You can find all the distributions in a package by exploring the family object in any ?glm\* call. | | | |

From the survey results, we have a mix of datasets that *could* be fit with a basic GLM (or extensions into mixed models), including

* presence of many species across an environmental gradient
* species across an environmental gradient - classic ecology, haha
* CO2 emissions through the year from soil that received different treatments
* abundance of fish exhibiting different social behaviors
* Herbarium specimen records (collection dates, locations) ~ local climate
* physiological response of individuals across size and thermal gradient
* abundance/percent cover
* species abundance and diversity across various gradients
* Pathogen presence/absence, species composition and diversity
* Phenological data from herbarium specimens
* Condition of a species across environmental gradient and time

C. GLMs in R

I’m going to go through three really common GLMs in ecology (Poisson, negative binomial, and binomial). I also have several extensions for GLMs tacked on to the Extensions tab for you to explore at your leisure ( 1. zero-inflation for count data and 2. categorical predictor variables).

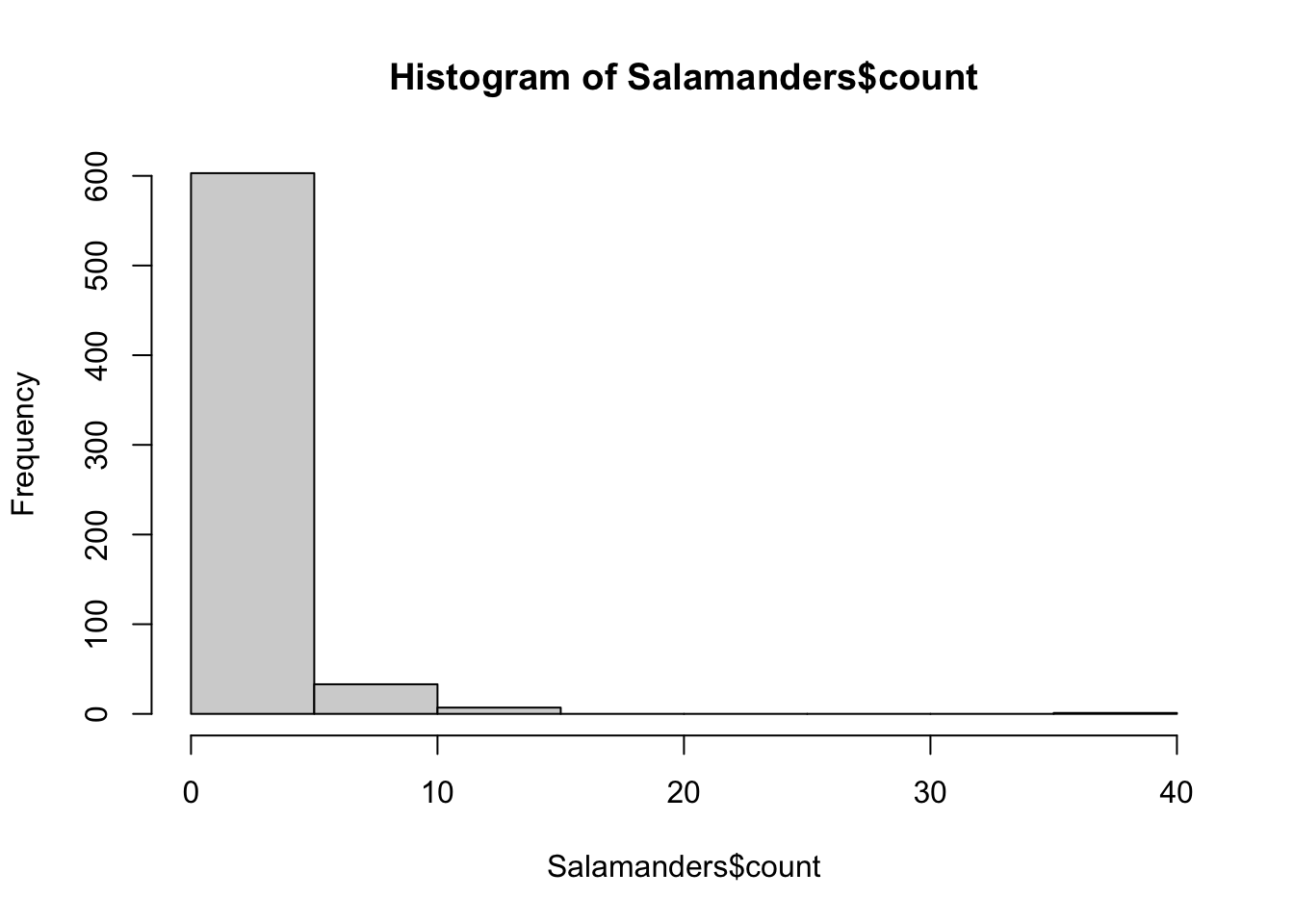
Why GLM and not a basic LM?

A lot of ecological data do not fit the basic assumptions of a linear model (specifically, normal data distributions).

Like a basic LM, GLMs do still have assumptions (heteroskedasticity, dispersion, and no zero-inflation).

We’ll be using a dataset on salamander abundances in streams that are either below mountaintop removal mining sites or in control (non-mined sites) to explore some GLM approaches. Let’s see why these data might be better suited for a GLM than a regular LM by looking at the frequency distribution (histogram) of the response variable (count) values. For an LM, the assumption is that these are normally distributed (e.g. have a nice bell curve shape).

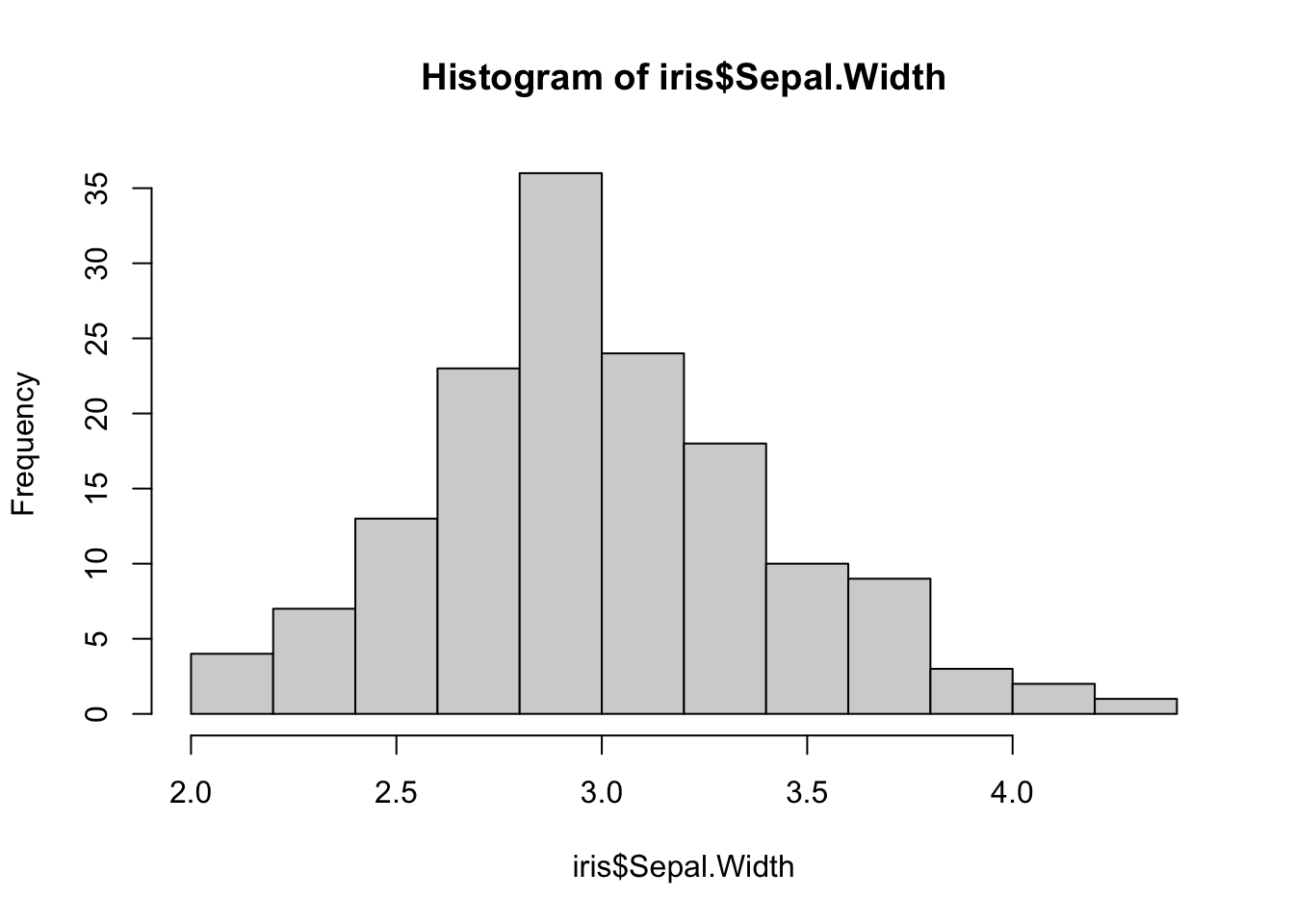
library(glmmTMB)  
data(Salamanders)  
hist(Salamanders$count)



The above histogram demonstrates a distribution I see in just about any of my ecological datasets that involve counts. Specifically, the data are *not* normally distributed (and there are a lot of zeros).

Conversely, the Sepal.Width measurements in the iris dataset from last week demonstrate a beautiful normally-distributed dataset that would be perfect for a basic linear model.

data(iris)  
hist(iris$Sepal.Width)



R Packages for GLMs

library(MASS) #glm.nb for negative binomial models  
library(glmmTMB) #Salamanders dataset and lots of families  
library(lme4) #grouseticks dataset and mixed model functionality  
library(DHARMa) #model diagnostics  
library(effects) #what do my marginal effects look like?  
library(performance) #binomial model diagnostics  
library(emmeans) #post hoc for categorical predictors

LMs, GLMs, and their extensions (e.g. mixed models) are used a TON in ecology and other fields so there are lots of R packages for regression, including some base packages and some more fancy ones.

**Ana uses:** stats (pre-loaded in R), MASS for the glm.nb function for negative binomial data, and glmmTMB for most GLM models and mixed models

**Tatum uses:** stat and MASS as above, plus stepAIC in boot to hone in on a best model quickly

There are also lots of packages for diagnosing model fits for GLMs and their extensions.

**Ana uses:** DHARMa for all the “do my data fit my model” questions, performance for “do my data fit my model” questions for binomial models, effects for looking at what the marginal effects are of the model (e.g. what is the relationship in my data?), and emmeans for models with categorical predictors (covered later)

**Tatum uses:** simulation and the boot package’s glm.diag functions to validate the model

Model-fitting packages

[stats](https://stat.ethz.ch/R-manual/R-devel/library/stats/html/00Index.html)

[MASS](https://cran.r-project.org/web/packages/MASS/index.html)

[glmmTMB](https://cran.r-project.org/web/packages/glmmTMB/index.html)

[boot](https://cran.r-project.org/web/packages/boot/index.html)

Model diagnostics packages

[DHARMa](https://cran.r-project.org/web/packages/DHARMa/index.html)

[performance](https://cran.r-project.org/web/packages/performance/index.html)

[effects](https://cran.r-project.org/web/packages/effects/index.html)

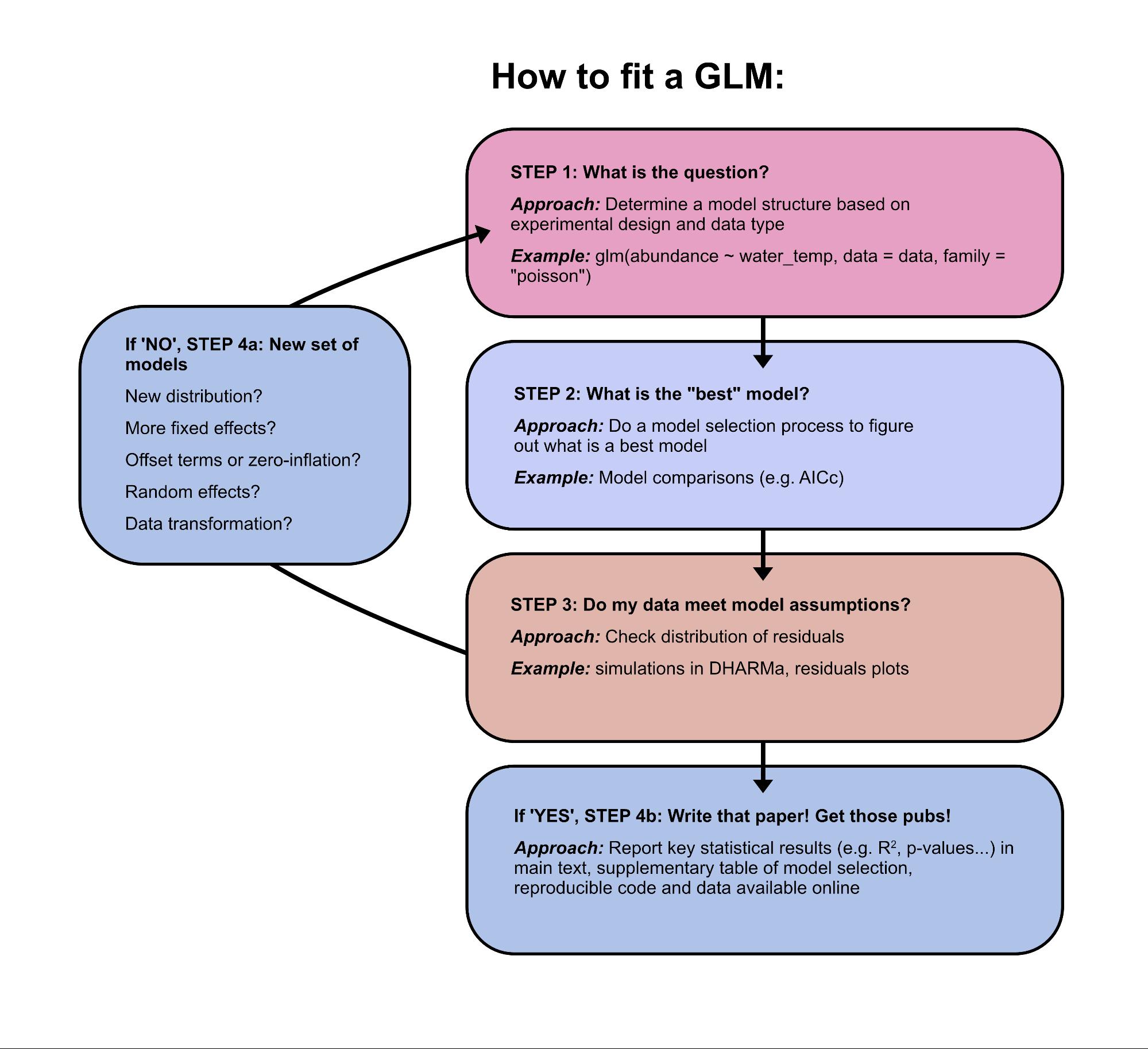
[emmeans](https://cran.r-project.org/web/packages/emmeans/index.html)

[boot](https://cran.r-project.org/web/packages/boot/index.html)

The GLM process

The process of fitting a GLM includes an iterative process of steps:

1. **Determine a model structure** based on experimental design (what is the effect of x (and potentially z, a, and b) on y (and potentially c and d)?)
   * What is the effect of water temperature and elevation on the abundance of caddisfly larvae?
   * What is the effect of habitat patch size and productivity on the presence or absence of spiders?
2. Do some **model selection process** to figure out what is a best model (teaser for next week, not part of today’s discussion!)
3. **Do my data meet model assumptions**, and if not, how do I fix this? (Ana: most of my time in statistics is spent on this step)



*Ana’s words of encouragement: I follow an approach I learned from the Zuur book when developing and fitting models to my data. Start a priori with a full model I want to test, some knowledge of the type of family it should fit (for counts, likely Poisson, negative binomial, or some derivative), and try to remember the ecological reasons for why I’m doing my statistics. Not only does this avoid finding significance just because I’ve thrown in everything and the kitchen sink, but it also helps me explain my statistics to myself and to others who might not be statisticians first. Once I dove in to model selecting, I realized that everyone on Stack Overflow has their own strong and differing opinions about the “right” way, which made me think there isn’t a “right” way and it’s more important to be able to justify what I did by being consistent and transparent with my process.*

Poisson GLM: Count data

Now the part we’ve all been waiting for - let’s fit one of these GLMs!

We’ll be using a dataset of salamander abundances from a paper on the effects of mountaintop removal on salamanders in streams ([Price et al. 2015](https://besjournals.onlinelibrary.wiley.com/doi/full/10.1111/1365-2664.12585)). The dataset is in the glmmTMB package.

data(Salamanders) #from glmmTMB

This dataset has 644 observations of 9 variables:

str(Salamanders)

## 'data.frame': 644 obs. of 9 variables:  
## $ site : Ord.factor w/ 23 levels "R-1"<"R-2"<"R-3"<..: 13 14 15 1 2 3 4 5 6 7 ...  
## $ mined : Factor w/ 2 levels "yes","no": 1 1 1 2 2 2 2 2 2 2 ...  
## $ cover : num -1.442 0.298 0.398 -0.448 0.597 ...  
## $ sample: int 1 1 1 1 1 1 1 1 1 1 ...  
## $ DOP : num -0.596 -0.596 -1.191 0 0.596 ...  
## $ Wtemp : num -1.2294 0.0848 1.0142 -3.0234 -0.1443 ...  
## $ DOY : num -1.497 -1.497 -1.294 -2.712 -0.687 ...  
## $ spp : Factor w/ 7 levels "GP","PR","DM",..: 1 1 1 1 1 1 1 1 1 1 ...  
## $ count : int 0 0 0 2 2 1 1 2 4 1 ...

**site**: name of a location where repeated samples were taken

**mined**: factor indicating whether the site was affected by mountain top removal coal mining

**cover**: amount of cover objects in the stream (scaled)

**sample**: repeated sample

**DOP**: Days since precipitation (scaled)

**Wtemp**: water temperature (scaled)

**DOY**: day of year (scaled)

**spp**: abbreviated species name, possibly also life stage

**count**: number of salamanders observed

Going back to the three steps we need to fit a GLM to our data:

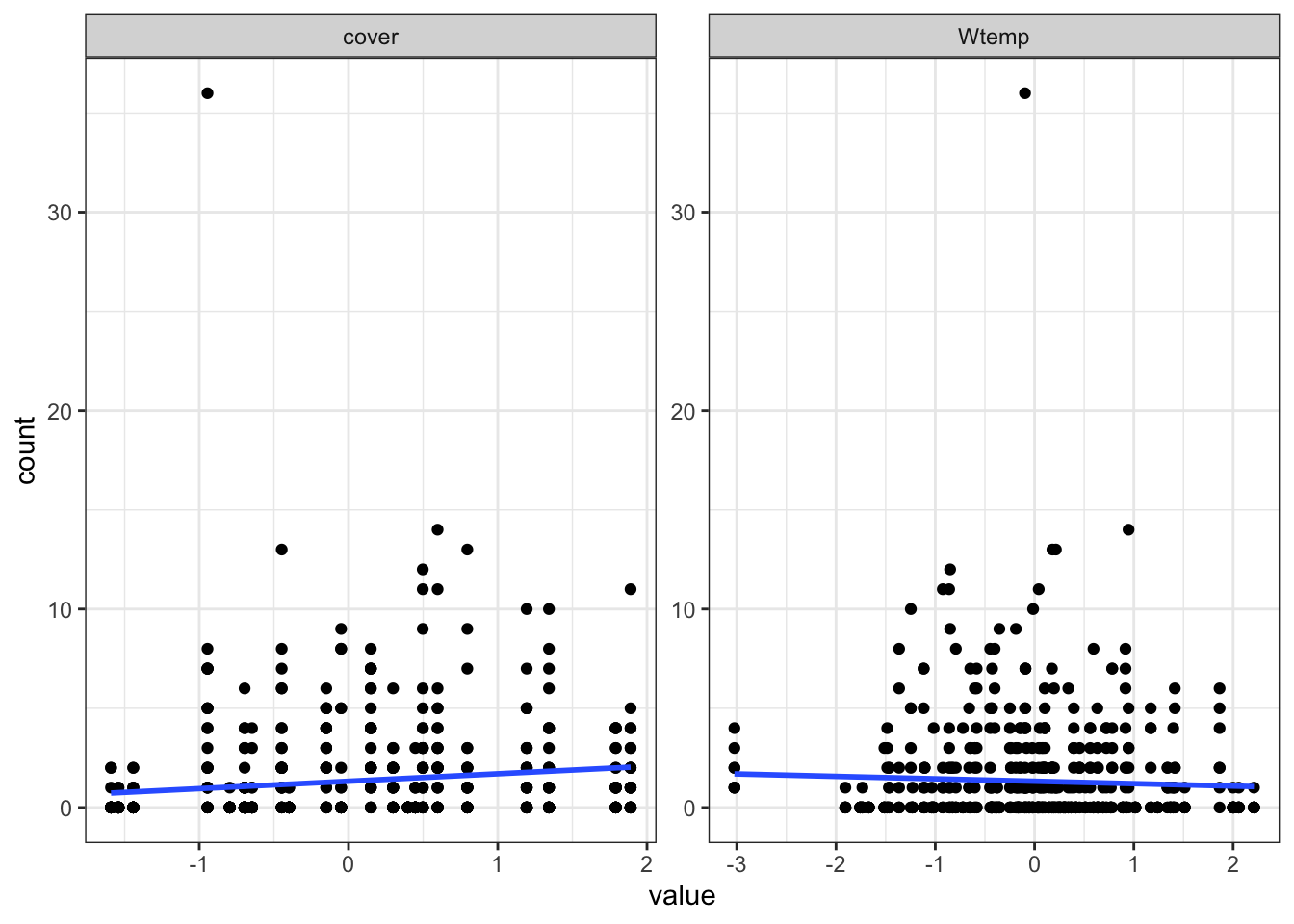
1. What is our question (e.g. what is the effect of x on y?)
2. What is the best model? (not addressed in this presentation)
3. Do my data meet model assumptions, and how do I fix this if not?

1. Poisson: What is the question?

Let’s say that this study was part of a project looking at how stream variables influenced salamander presence or abundance (we address [categorical predictors](#gjdgxs) in the Extensions and will revisit the mined vs. not mined question there and how to approach that with a GLM). Our question with a continuous predictor might be:

* What are the effects of cover and water temperature on salamander abundance?

Salamanders %>%  
 gather(-site, -mined, -sample, -DOP, -DOY, -spp, -count, key = "var", value = "value") %>%  
 ggplot(aes(x = value, y = count)) +  
 geom\_point() +  
 geom\_smooth(method = "lm", se =F) +  
 facet\_wrap(~ var, scales = "free") +  
 theme\_bw()



Based on this graph, *maybe* there is some positive influence of cover on salamander abundance and some negative influence of water temperature. (To answer this question conclusively and correctly, we’d need to do step 2. what is the *best* model, but stay tuned, that’s coming up next week!)

The same way we fit an LM in the last session, we can call a GLM, but now using a GLM formula. Again, from above, we have abundance (count) data with a left-skewed distribution (not normal), so we will be starting with a Poisson distribution and tinkering with fixes on that model family (e.g. negative binomial).

model <- glm(count ~ cover + Wtemp,  
 data = Salamanders,  
 family = "poisson")

We can look at the model summary, which looks really similar to the basic regression summary.

summary(model)

##   
## Call:  
## glm(formula = count ~ cover + Wtemp, family = "poisson", data = Salamanders)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.1270 -1.5988 -1.3516 0.3424 13.7366   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.24068 0.03558 6.764 1.34e-11 \*\*\*  
## cover 0.27085 0.03491 7.758 8.64e-15 \*\*\*  
## Wtemp -0.05746 0.03692 -1.557 0.12   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for poisson family taken to be 1)  
##   
## Null deviance: 2120.7 on 643 degrees of freedom  
## Residual deviance: 2053.8 on 641 degrees of freedom  
## AIC: 2783.1  
##   
## Number of Fisher Scoring iterations: 6

Based on this model, maybe there is an effect of cover on salamander abundance.

2. Poisson: What is the “best” model (Next week!)

Let’s pretend this is our best model, which means we can go on to step 3.

3. Poisson: Do my data fit the model assumptions?

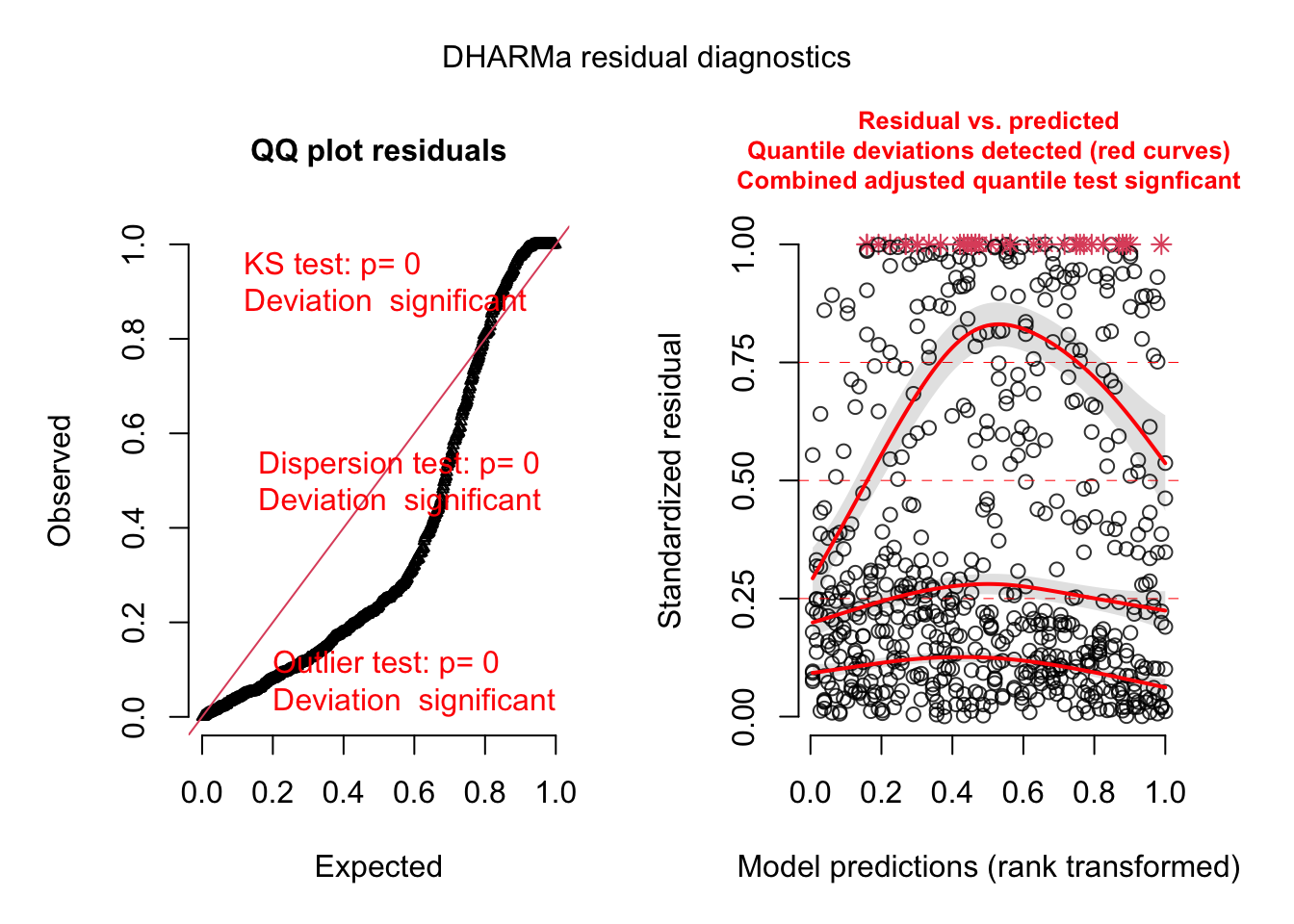
Now that we have our model that we want to report in our paper, we need some way to say whether our data actually meet the assumptions of this model. You can approach these in a similar way to the LM model tests of last week, but there are several extensions here (dispersion and zero-inflation) AND some reasons as you get to more complicated models (e.g. mixed models) as to why you might want something that gives you some confidence interval (e.g. p-value) to whether your visual assessments of your data warrant revisiting your model.

I use the DHARMa package for this in all my linear models (mixed and non-mixed). DHARMa simulates a bunch of sets of residuals from each observations based on the fitted model, including the probability of those values given the data. The outputs are AWESOME diagnostics that look really similar to the qqplots and residual plots from basic model diagnostics, except now with some confidence that the data fit model assumptions tacked on (and the ability to determine overdispersion and zero-inflation).

This is a very simplified explanation of the DHARMa package, but they have a great [tutorial](https://cran.r-project.org/web/packages/DHARMa/vignettes/DHARMa.html) with more in-depth explanations and lots more functions from the package.

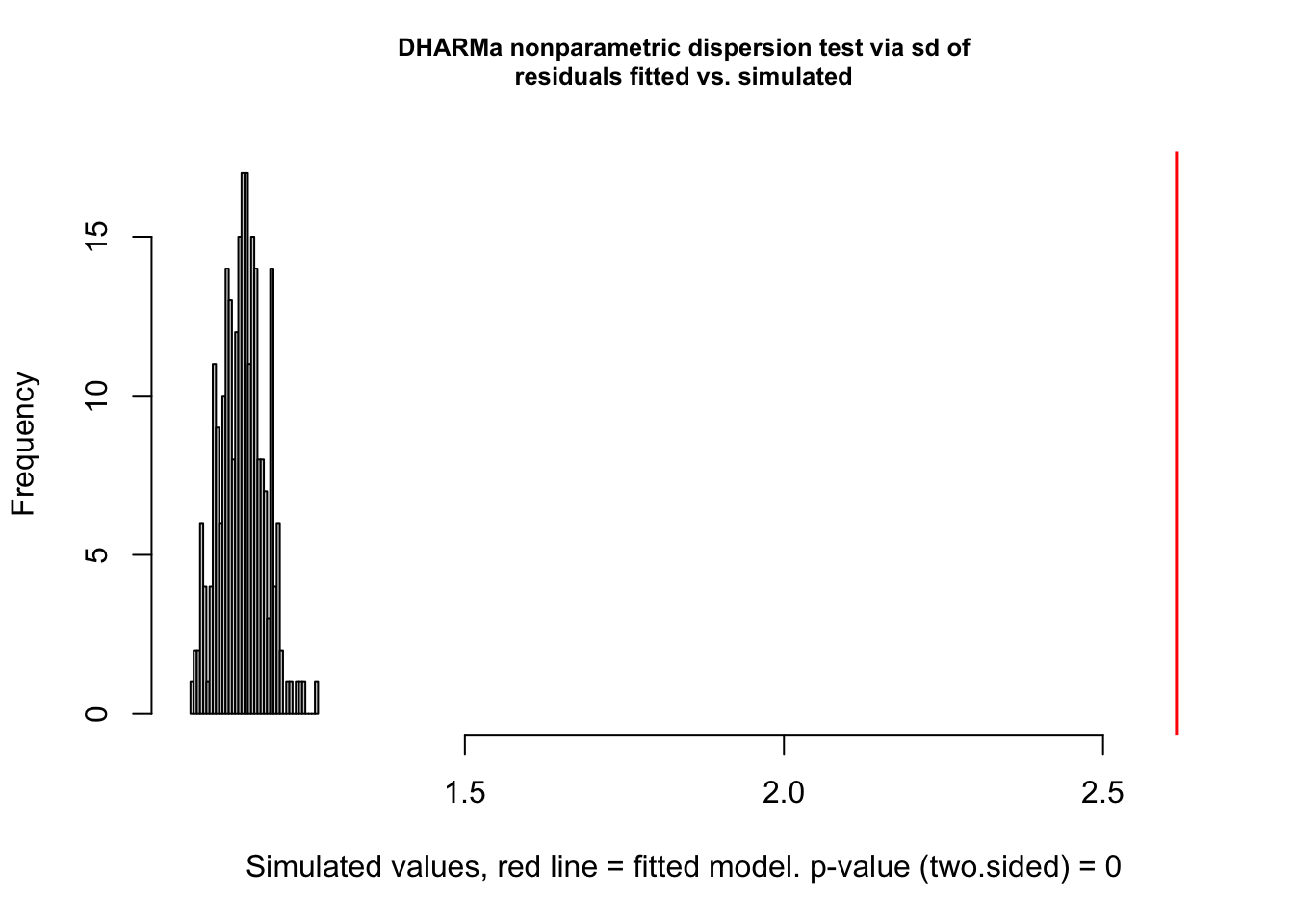
First, we’ll simulate our batches of residuals based on our fitted model:

simulationOutput <- simulateResiduals(fittedModel = model, plot = T)



YIKES. We’ve got some issues with model assumptions. This shape for a qqplot is definitely evidence of some potential overdispersion, but we can test that explicitly in the DHARMa package using the testDispersion() function.

testDispersion(simulationOutput)



##   
## DHARMa nonparametric dispersion test via sd of residuals fitted vs.  
## simulated  
##   
## data: simulationOutput  
## ratioObsSim = 2.2714, p-value < 2.2e-16  
## alternative hypothesis: two.sided

The testDispersion() function gives us a histogram of simulated *expected* values, and then a red line indicating our *observed* dispersion in our actual data. Additionally, this function performs a statistical test of our dispersion with the null hypothesis being “We do not have overdispersion”. Our very low p-value indicates that our dispersion value is not very likely to come from the distribution of expected dispersion values for our model. Simply: **we have overdispersion**.

Poisson: Common fixes for models that don’t meet assumptions

Overdispersion is a really common problem for count data and the good news is that there are some pretty easy fixes. *(A place where opinions on the order in which you apply fixes deviates, but I’ll show you mine. Kyle Edwards and the Zuur book both have great workflows for this, which I based mine off as well. You can also find my* [*flowchart for model selection*](#30j0zll) *in the Extensions tab)*.

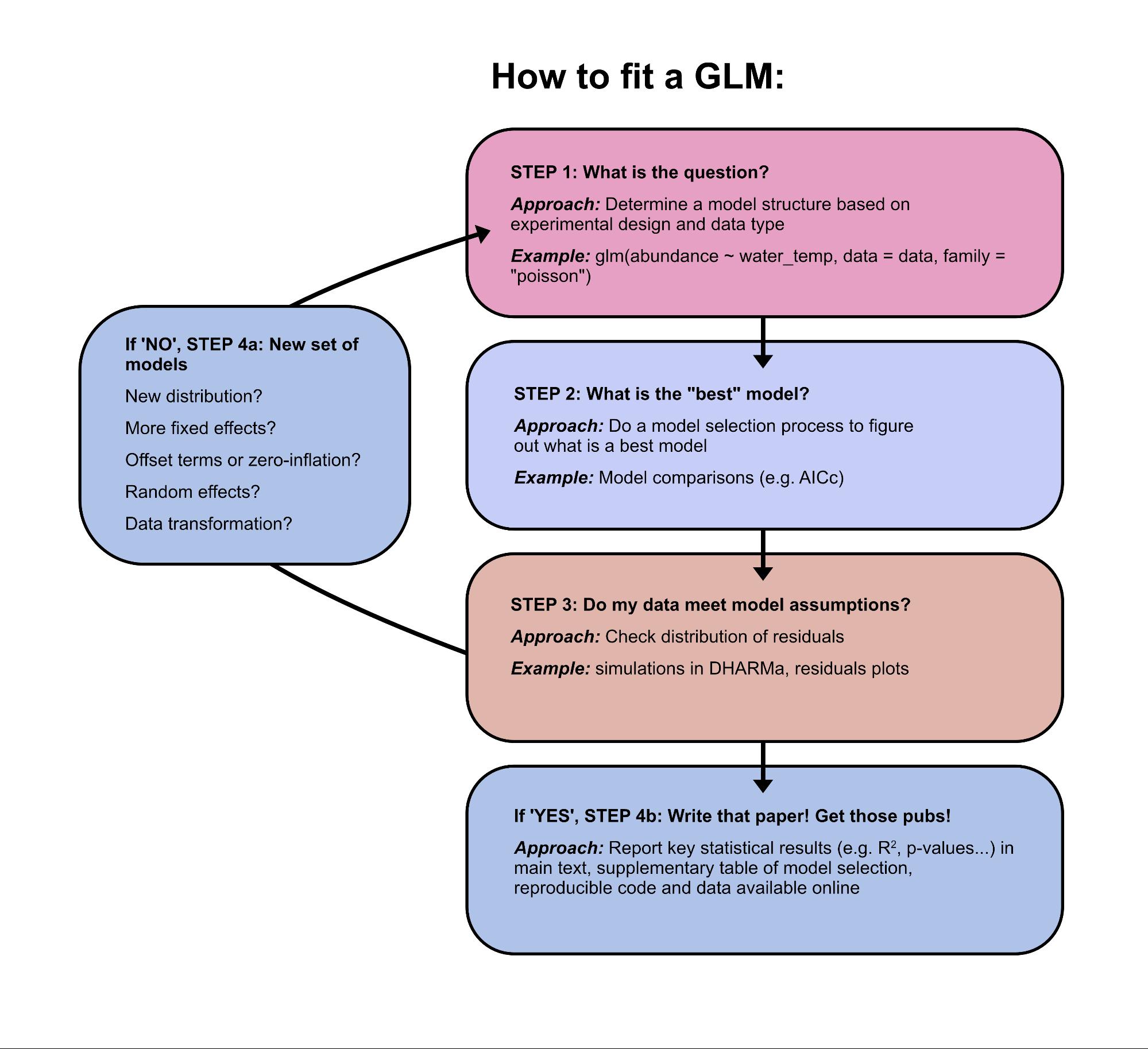
The fixes used can depend on your data and what your model looks like in the first place. There are some simple math-related reasons based on model distributions for why overdispersion could be occurring and there are also some more practical reasons based on the variables we put into (or didn’t put into) the model.

1. Did I include all the relevant variables?
   * Are there other fixed effects that may be needed in my model?
   * (Not covered today, and maybe a problem with this dataset: is there a need for grouping variables (random effects and mixed models)?)
2. Do my data actually fit a Poisson distribution? (lots of opinions and approaches to address this point and Tatum and Ana differ in their approaches)
   * Ana’s approach: 1) refit with a negative binomial model, 2) add a zero-inflation term, 3) think about data transformations. (some other useful overdispersion tips, specifically for GLMMs, but useful for GLMs too, on [Ben Bolker’s GLMM FAQ page](https://bbolker.github.io/mixedmodels-misc/glmmFAQ.html))
   * Tatum’s approach: 1) refit with a negative binomial or really any other function that has correct support (see table, may require transformation), 2) add a zero-inflation term or choose a hurdle model

Negative binomial GLM: overdispersed count data

Taking Ana’s approach, let’s start by refitting our data to a negative binomial distribution, which we can do with the glm.nb function from the MASS package.

This means we’re returning to Step 1 via Step 4A in our GLM process flowchart:



1. NB: What is the question?

In this case, the ecological question stays the same, we’re just using a different model family to better account for model assumptions.

nb.model <- glm.nb(count ~ cover + Wtemp,  
 data = Salamanders)

Looking at the summary of this model, it looks like cover is still important, but the model coefficients have changed.

summary(nb.model)

##   
## Call:  
## glm.nb(formula = count ~ cover + Wtemp, data = Salamanders, init.theta = 0.3343960957,   
## link = log)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.2138 -1.0258 -0.9076 0.1246 4.4182   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.22911 0.07721 2.967 0.003 \*\*   
## cover 0.34692 0.07955 4.361 1.29e-05 \*\*\*  
## Wtemp -0.09539 0.08050 -1.185 0.236   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for Negative Binomial(0.3344) family taken to be 1)  
##   
## Null deviance: 546.2 on 643 degrees of freedom  
## Residual deviance: 528.8 on 641 degrees of freedom  
## AIC: 1904  
##   
## Number of Fisher Scoring iterations: 1  
##   
##   
## Theta: 0.3344   
## Std. Err.: 0.0328   
##   
## 2 x log-likelihood: -1896.0100

(Just as a reminder, the summary for the Poisson model):

summary(model)

##   
## Call:  
## glm(formula = count ~ cover + Wtemp, family = "poisson", data = Salamanders)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.1270 -1.5988 -1.3516 0.3424 13.7366   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 0.24068 0.03558 6.764 1.34e-11 \*\*\*  
## cover 0.27085 0.03491 7.758 8.64e-15 \*\*\*  
## Wtemp -0.05746 0.03692 -1.557 0.12   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for poisson family taken to be 1)  
##   
## Null deviance: 2120.7 on 643 degrees of freedom  
## Residual deviance: 2053.8 on 641 degrees of freedom  
## AIC: 2783.1  
##   
## Number of Fisher Scoring iterations: 6

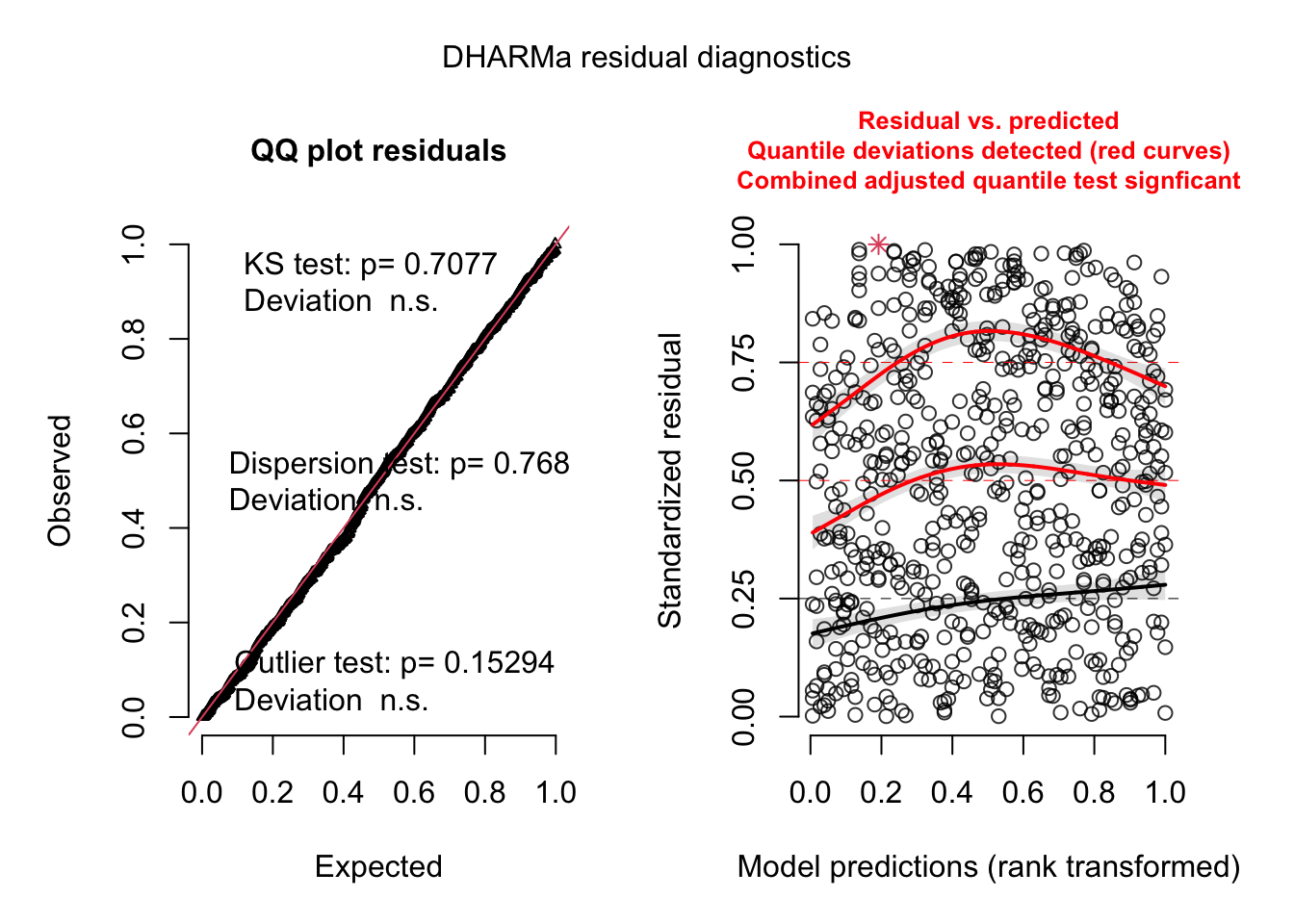
2. NB: What is the “best” model (Next week!)

Again, let’s say we went through the step 2 model selection process and this came out as the best model.

3. NB: Do my data fit the model assumptions?

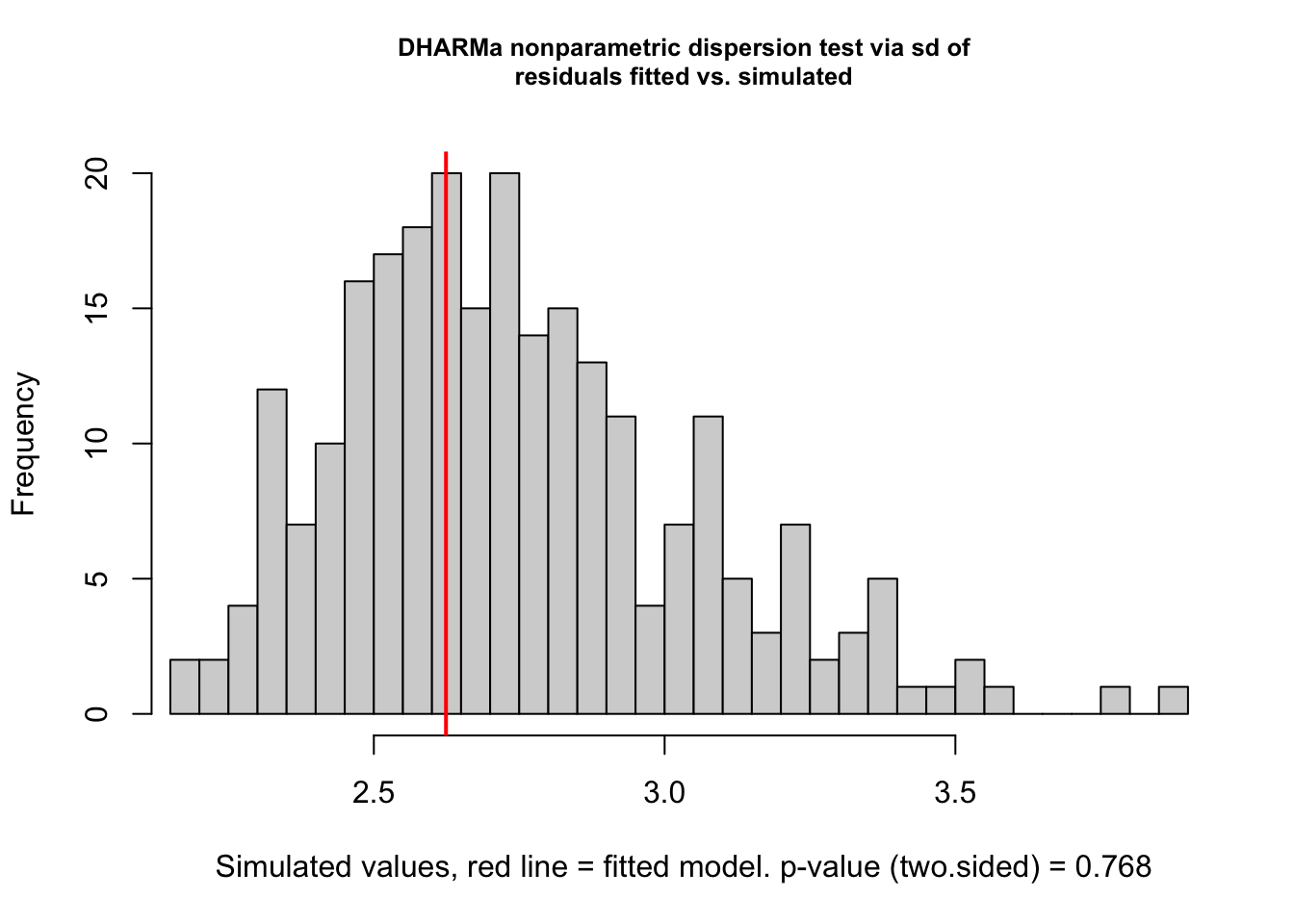
Let’s see how the model and data do in terms of model assumptions:

simulationOutput <- simulateResiduals(fittedModel = nb.model, plot = T)



It looks like we’ve fixed the overdispersion problem, but may still have some issues with this dataset based on the residuals plot. Let’s look at the dispersion problem specifically:

testDispersion(simulationOutput)



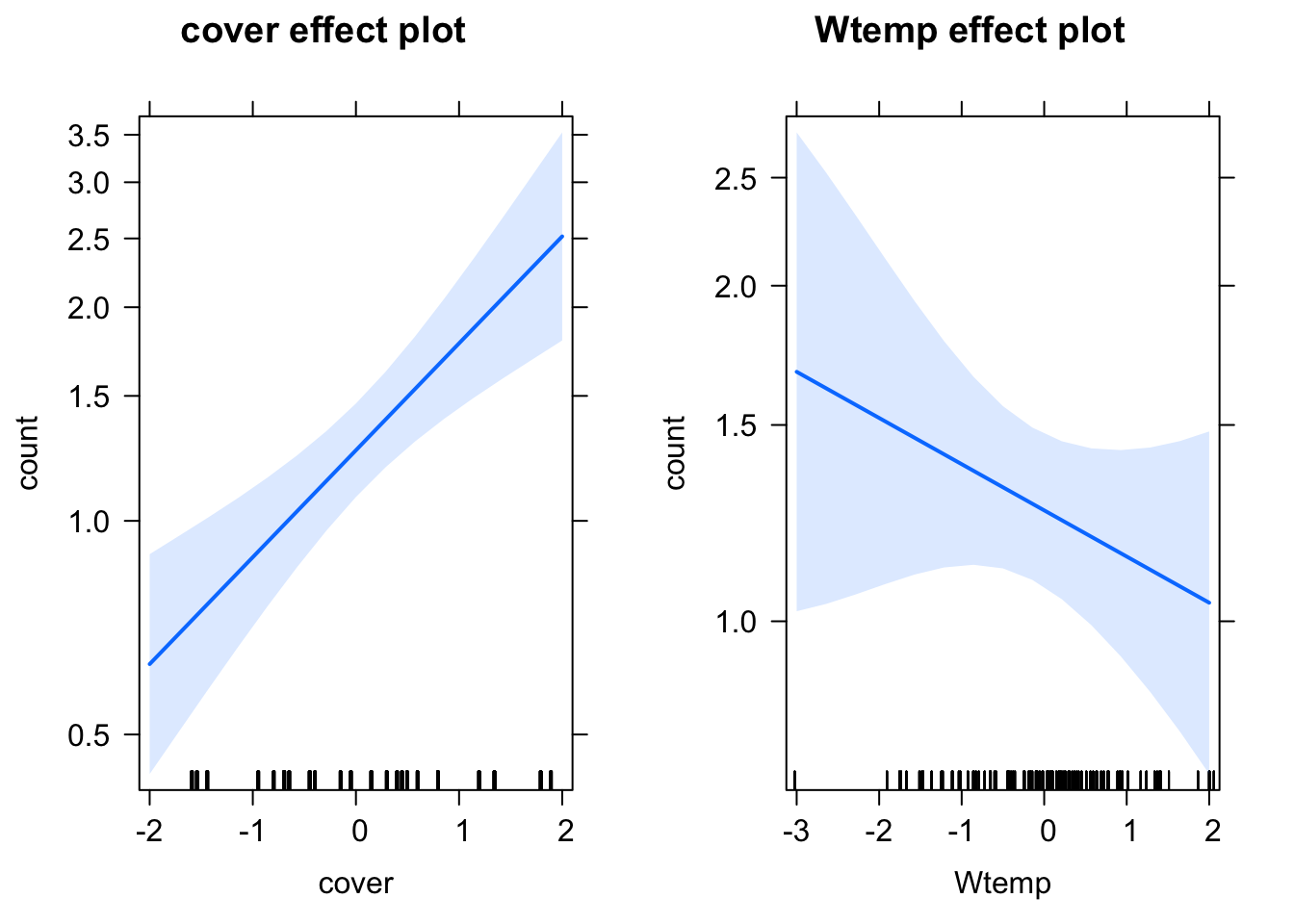
##   
## DHARMa nonparametric dispersion test via sd of residuals fitted vs.  
## simulated  
##   
## data: simulationOutput  
## ratioObsSim = 0.95545, p-value = 0.768  
## alternative hypothesis: two.sided

We have fixed the overdispersion! However, based on the output from this model, it still looks like we have some patterns in our residuals. I’m not surprised about this, given that it’s an example dataset in a mixed model package and so likely we are having more of a practical problem with missing effects in this model. (Maybe once we learn mixed models we can revisit?)

Count data wrap-up: a quick visualization

I’m a visual person, so at this point I always want to know what my model is actually telling me ecologically (e.g. what is the effect of cover??). I do quick diagnostics using the effects package and the allEffects function. This is just a line graph showing the marginal change in count for this model given a change in each of the predictors, while accounting for everything else in the model. You can get similar and more complex marginal effects plots with the ggeffects package, but I haven’t played with it much. I wouldn’t put this into my final paper, but it helps me to stay grounded in the ecology when I get lost in the R and statistics world.

plot(allEffects(nb.model))



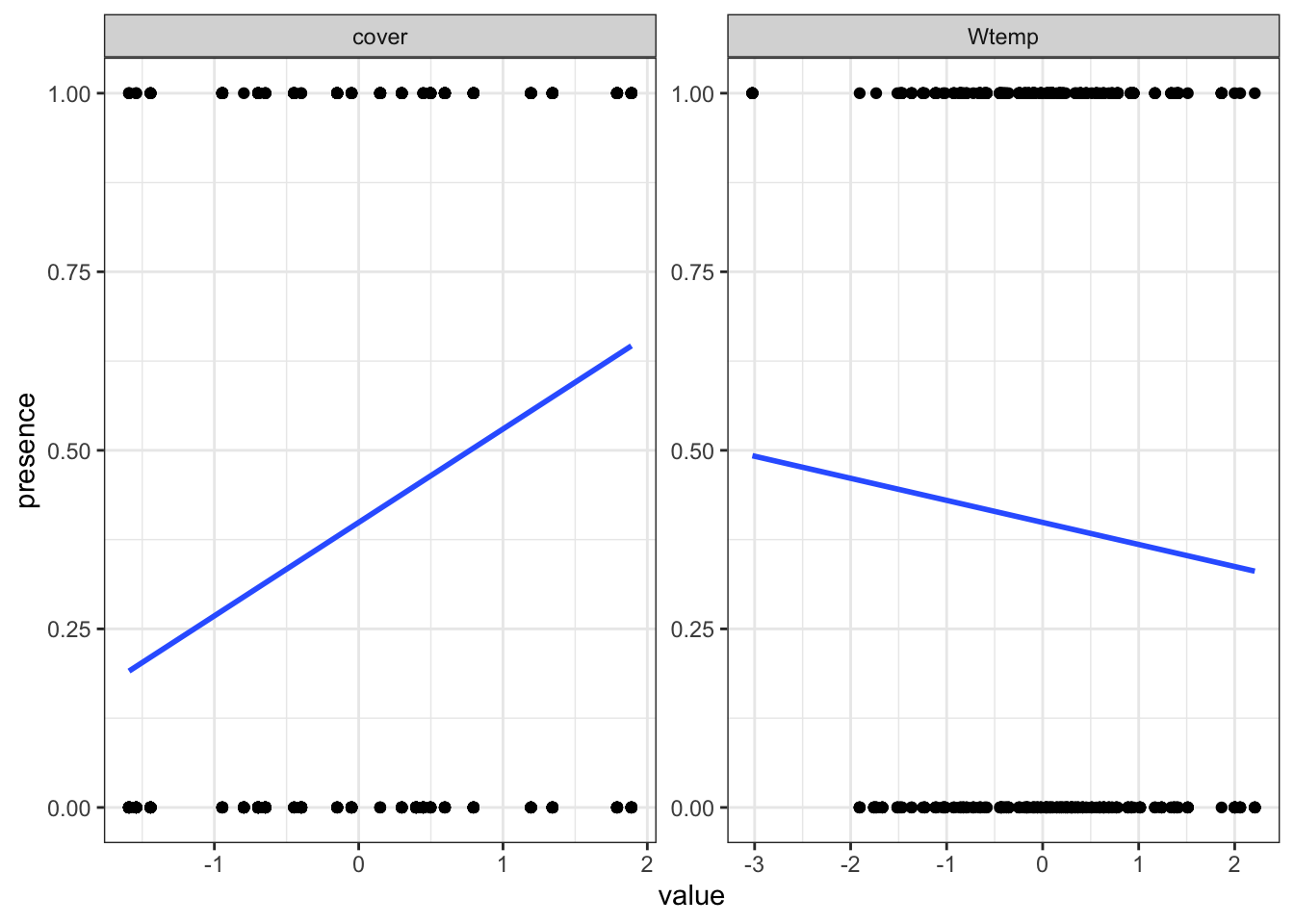
Binomial GLM: Presence-absence data

Now let’s say we were actually interested in just whether we find salamanders or not, not their actual abundances in each stream. This means we have a binary response of presence-absence and want to know whether presence is predicted by cover or water temperature. We can convert the count data to a binary variable of presence:

Salamanders <- Salamanders %>%  
 mutate(presence = ifelse(count > 0, 1, 0))

And then look at whether our two predictors (stream cover and water temperature) predict presence.

Salamanders %>%  
 gather(-site, -mined, -sample, -DOP, -DOY, -spp, -count, -presence, key = "var", value = "value") %>%  
 ggplot(aes(x = value, y = presence)) +  
 geom\_point() +  
 geom\_smooth(method = "lm", se =F) +  
 facet\_wrap(~ var, scales = "free") +  
 theme\_bw()



This is kind of a hard graph to visualize because of the binary response variable, but visually it does look like there might be more 1’s as cover increases and maybe more zeros as water temperature increases.

1. Binomial: What is the question

Our question here is really similar to the question above, except now we want to know whether stream cover and water temperature predict the *presence* of salamanders, rather than their abundance. Now that we have a binary presence-absence response variable, we will be using the binomial distribution to fit our data. You can run a binomial model by just changing the family = argument in the glm() function.

bi.model <- glm(presence ~ cover + Wtemp,  
 data = Salamanders,  
 family = binomial)

Our model summary is really similar to before and suggests some evidence that stream cover is, indeed, predictive of whether we see salamanders in streams or not.

summary(bi.model)

##   
## Call:  
## glm(formula = presence ~ cover + Wtemp, family = binomial, data = Salamanders)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.4782 -0.9944 -0.7274 1.2272 1.8225   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -0.43787 0.08380 -5.225 1.74e-07 \*\*\*  
## cover 0.56261 0.08914 6.311 2.76e-10 \*\*\*  
## Wtemp -0.05290 0.08758 -0.604 0.546   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 866.35 on 643 degrees of freedom  
## Residual deviance: 821.17 on 641 degrees of freedom  
## AIC: 827.17  
##   
## Number of Fisher Scoring iterations: 4

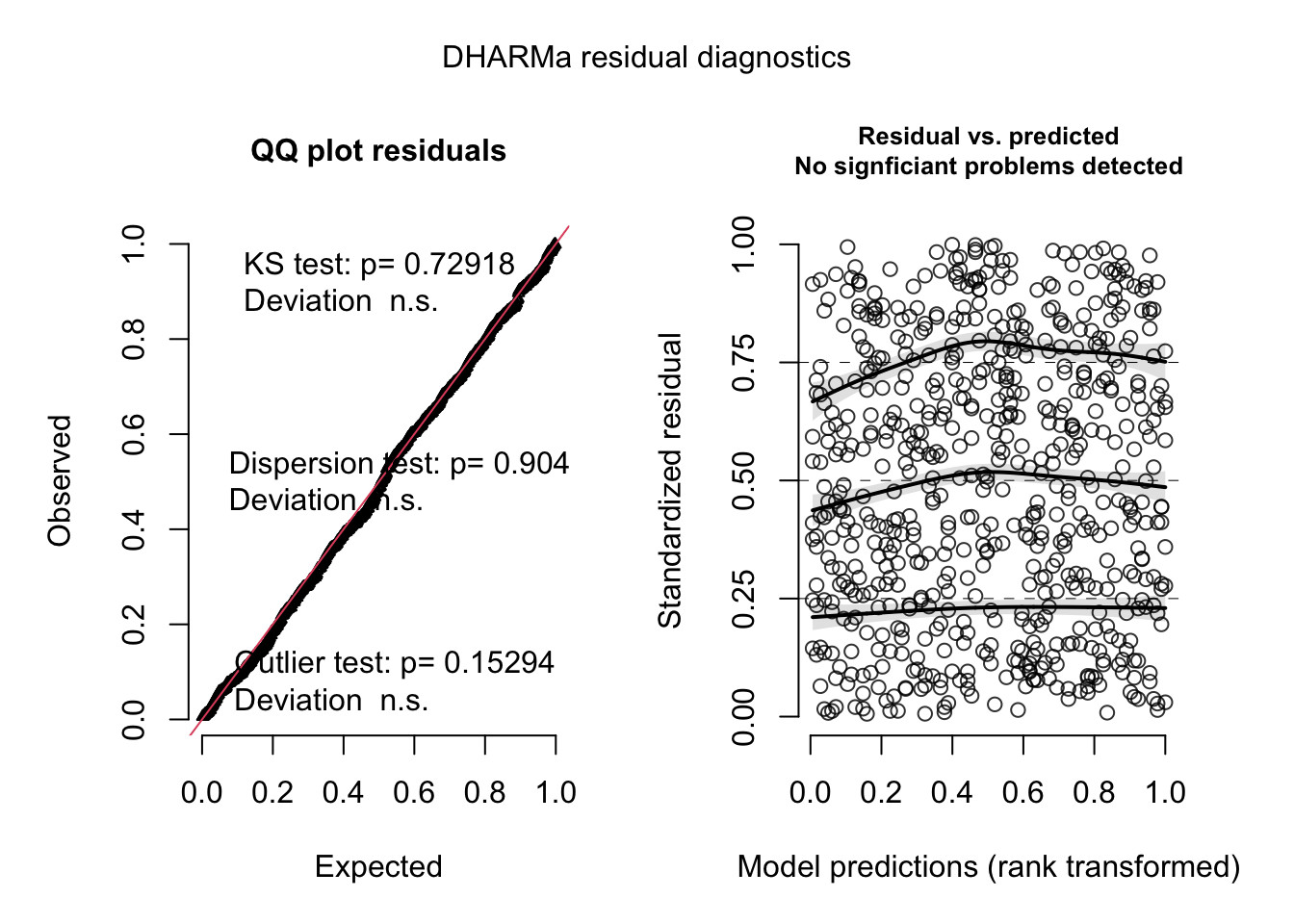
2. Binomial: What is the “best” model (Next week!)

Again skipping this step for now!

3. Binomial: Does my data fit the model assumptions

With this model, we can again use DHARMa to test model assumptions and we can also use the binned\_residuals() function in the performance package, which is specifically designed for binomial distributions.

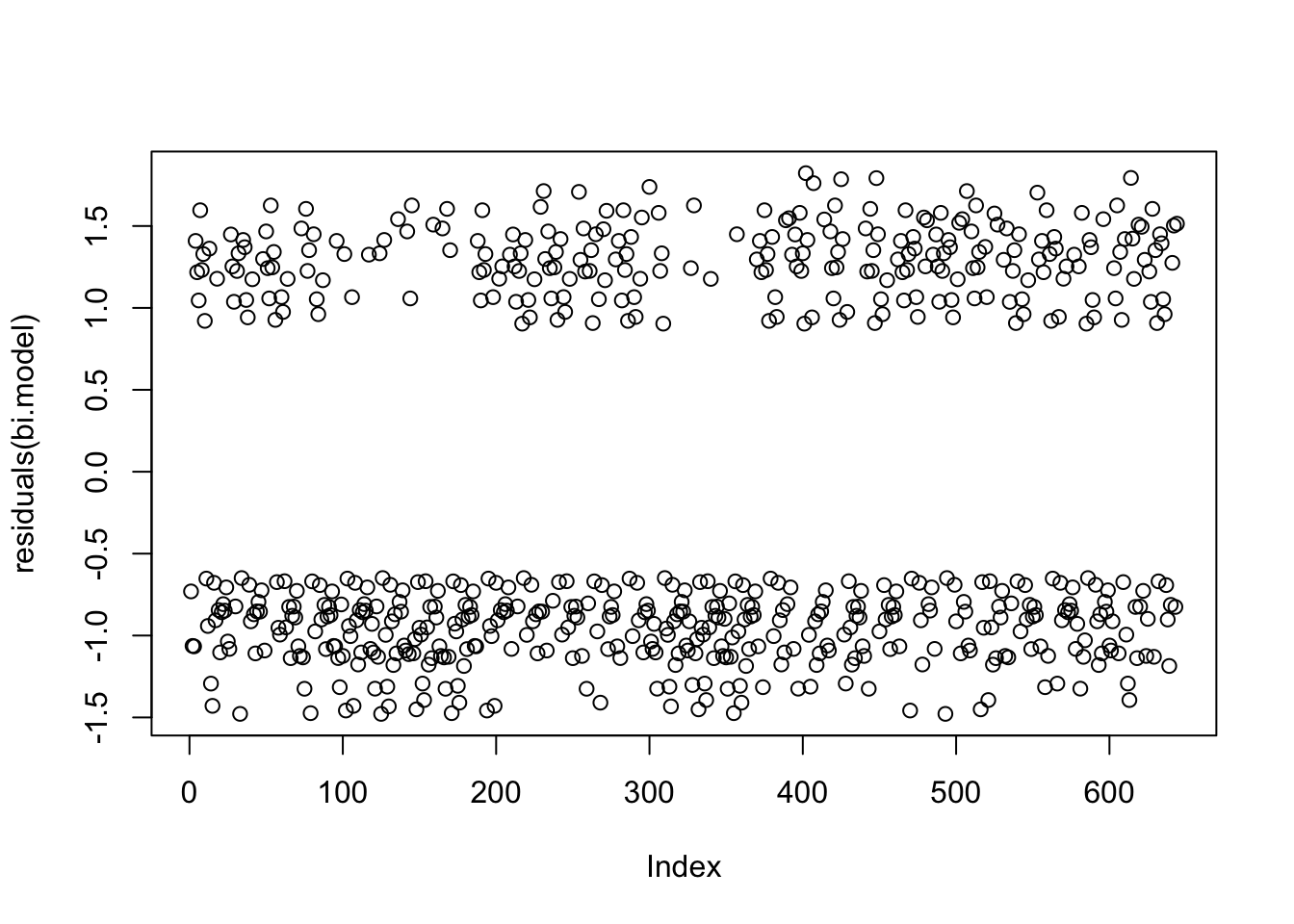
simulationOutput <- simulateResiduals(fittedModel = bi.model, plot = T)



According to this, this model actually performs pretty well.

The residuals for a binomial model always look pretty funky:

plot(residuals(bi.model))

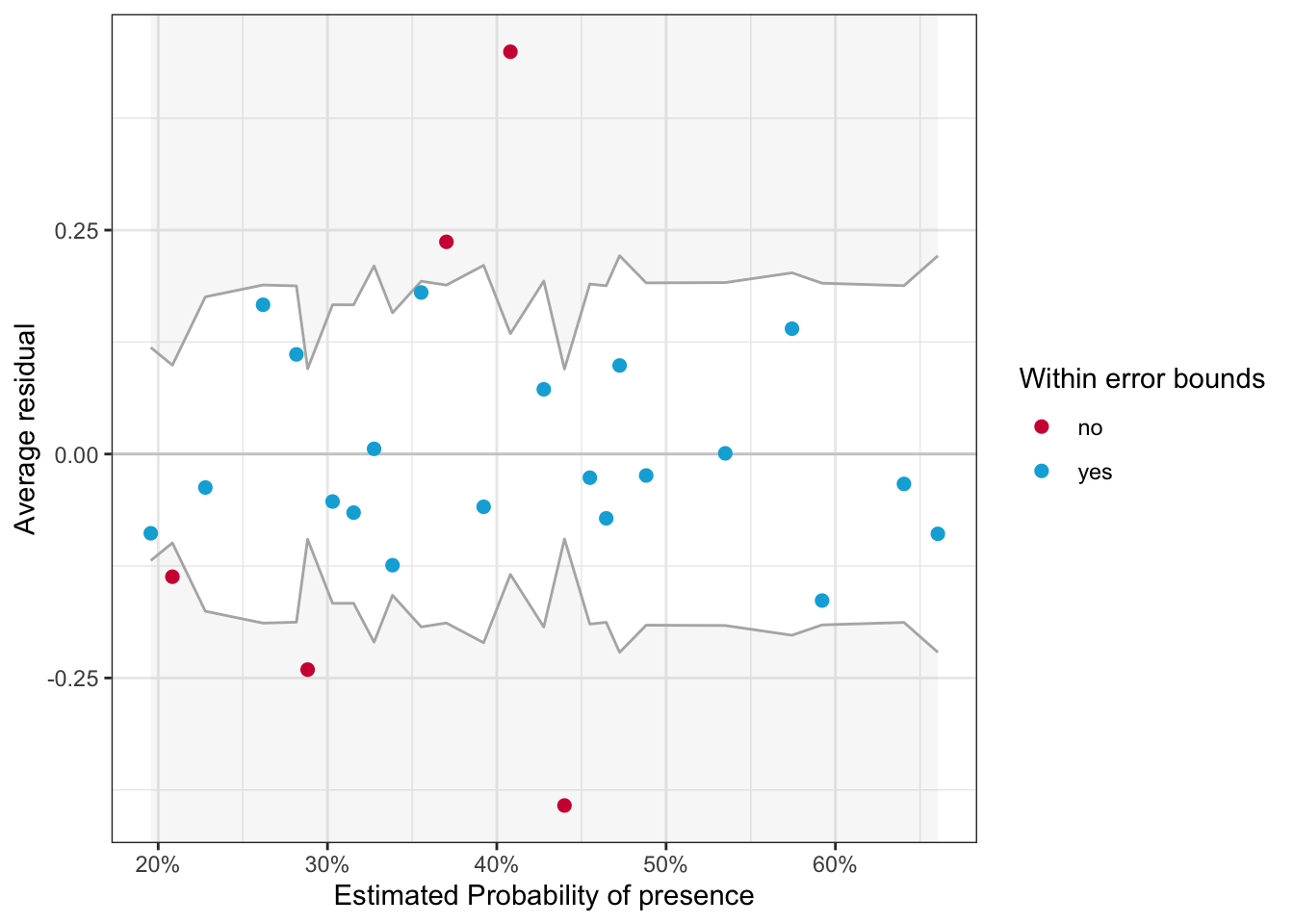


So we can use the binned\_residuals function to tell how well our model is doing, which is… ok, but not great.

From the vignette for the performance package: “Binned residual plots are achieved by “dividing the data into categories (bins) based on their fitted values, and then plotting the average residual versus the average fitted value for each bin.” (Gelman,Hill 2007: 97). If the model were true, one would expect about 95% of the residuals to fall inside the error bounds."

binned\_residuals(bi.model)

## Warning: About 80% of the residuals are inside the error bounds (~95% or higher would be good).



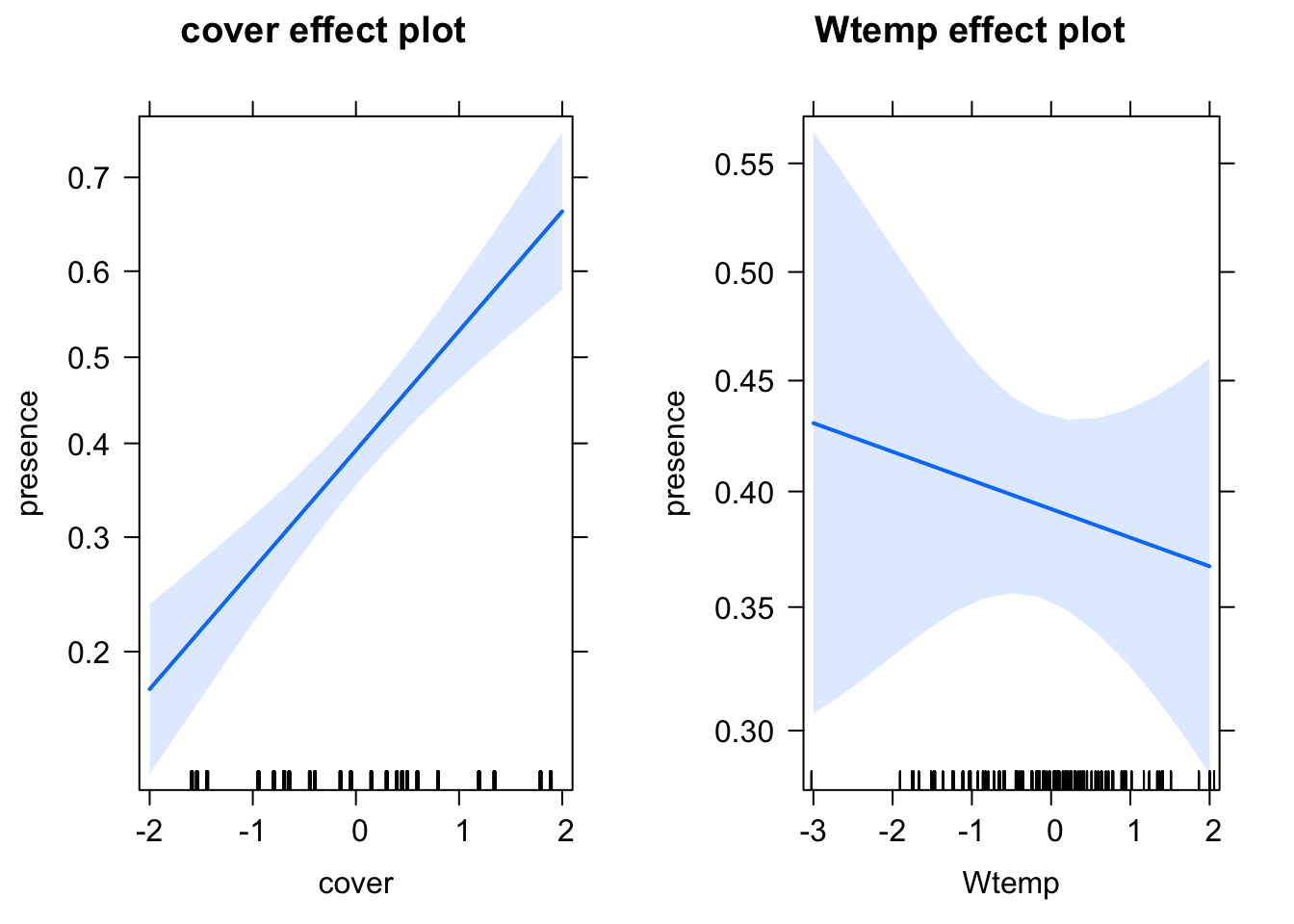
Based on these residuals, our data is *ok*, but not great at fitting model assumptions (95% of residuals). Again, this could be because of the practical or math reasons outlined above from the Poisson and negative binomial models:

1. Did I include the right variables in my model?
   * Are there other fixed effects that may be needed in my model?
   * (Not covered today, and maybe a problem with this dataset: is there a need for grouping variables (random effects and mixed models)?)
2. Do my data actually fit a binomial distribution?
   * variable transformations or quasi-binomial distribution

Presence data wrap-up: a quick visualization

We can do another quick diagnostic visualization to ground ourselves here as well!

plot(allEffects(bi.model))



D. Extensions

There are lots of kinds of ecological data and so many model distributions for asking statistical questions about them. It’s impossible to cover them all in an hour!

In this section, I’m going to go over a couple other really common types of data for GLMs, including **zero-inflated count data** and models with **categorical predictors**.

In addition, I’ve added a [flowchart](#30j0zll) I use for fitting, choosing, and diagnosing GLMs and GLMMs for count data. It’s a little bit of a teaser for the coming weeks, but hopefully it can be a resource!

Extension 1: Zero-inflated count data

As you may know from your own datasets, ecological data of counts often have a lot of zeros. Sometimes, these datasets have so many zeros that neither Poisson or negative binomial models will work. There are a set of models called “Zero-inflated” models (can be either Poisson or negative binomial), which are basically that - models that can fit data that have an “inflated” number of zeros. There are tons of resources on this, and I won’t cover it all here, just providing a practical worked example. (there are also two types of zero-inflated models that I won’t go into, but have to do with where the zeros are coming from, not going to try to have an exhaustive example here, but look into it if this is a problem for you!)

**Zuur chapter 11 is a go-to resource to get started!**

[Another set of worked examples](https://fukamilab.github.io/BIO202/04-C-zero-data.html#key_points)

[Books](https://highstat.com/index.php/beginner-s-guide-to-zero-inflated-models)

[More books](https://highstat.com/index.php/zero-inflated-models-and-generalized-linear-mixed-models-with-r)

[GOOGLE](https://www.google.com/search?sxsrf=ALeKk00BAogDZPbJ5yHQg810CFuTJ3HEgA:1595002818410&ei=ws8RX83OGMe0tAaxl474Aw&q=zero+inflated+models+ecology&oq=zero+inflated+models+ecology&gs_lcp=CgZwc3ktYWIQAzIFCAAQzQIyBQgAEM0CMgUIABDNAjIFCAAQzQIyBQgAEM0COgQIABBDOgYIABAHEB46BwgAEBQQhwI6AggAOgYIABAWEB46CAgAEBYQChAeOgUIIRCgAToHCCEQChCgAVDpJVjlNGC1NmgAcAB4AIAB3AGIAbsIkgEFMi42LjGYAQCgAQGqAQdnd3Mtd2l6&sclient=psy-ab&ved=0ahUKEwjNt6DD2NTqAhVHGs0KHbGLAz8Q4dUDCAw&uact=5)

Let’s play with a different dataset from the lme4 package on the number of ticks observed in grouse chicks in nests in different years, elevations, and locations.

data(grouseticks)  
str(grouseticks)

## 'data.frame': 403 obs. of 7 variables:  
## $ INDEX : Factor w/ 403 levels "1","2","3","4",..: 1 2 3 4 5 6 7 8 9 10 ...  
## $ TICKS : num 0 0 0 0 0 3 2 0 0 2 ...  
## $ BROOD : Factor w/ 118 levels "501","502","503",..: 1 1 2 3 3 3 3 4 4 4 ...  
## $ HEIGHT : num 465 465 472 475 475 475 475 488 488 488 ...  
## $ YEAR : Factor w/ 3 levels "95","96","97": 1 1 1 1 1 1 1 1 1 1 ...  
## $ LOCATION: Factor w/ 63 levels "1","2","3","4",..: 32 32 36 37 37 37 37 44 44 44 ...  
## $ cHEIGHT : num 2.76 2.76 9.76 12.76 12.76 ...

**INDEX**: (factor) chick number (observation level)

**TICKS**: number of ticks sampled

**BROOD**: (factor) brood number

**HEIGHT**: height above sea level (meters)

**YEAR**: year (-1900)

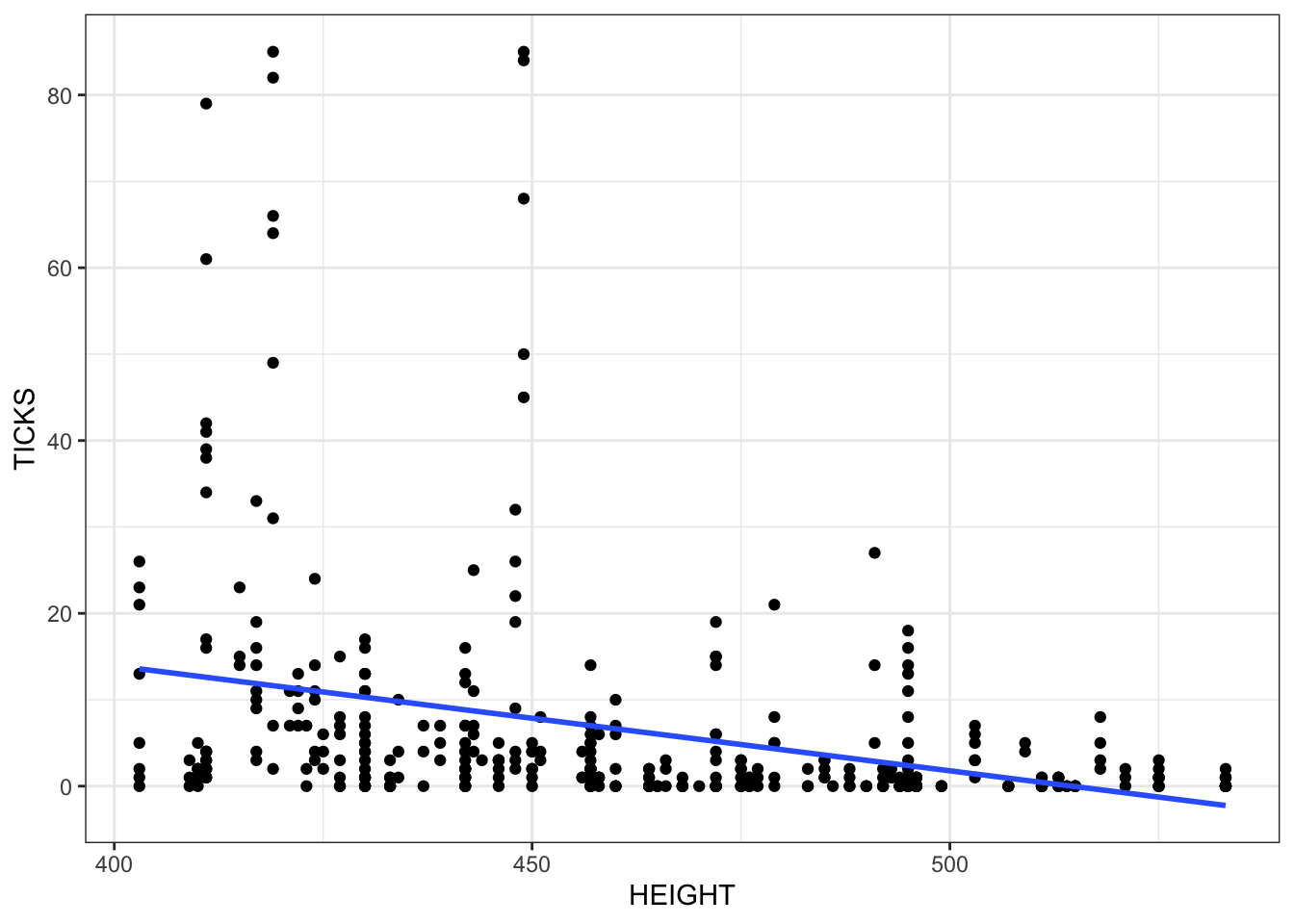
**LOCATION**: (factor) geographic location code

**cHEIGHT**: centered height, derived from HEIGHT

1. Zero-inflated count data: What is the question?

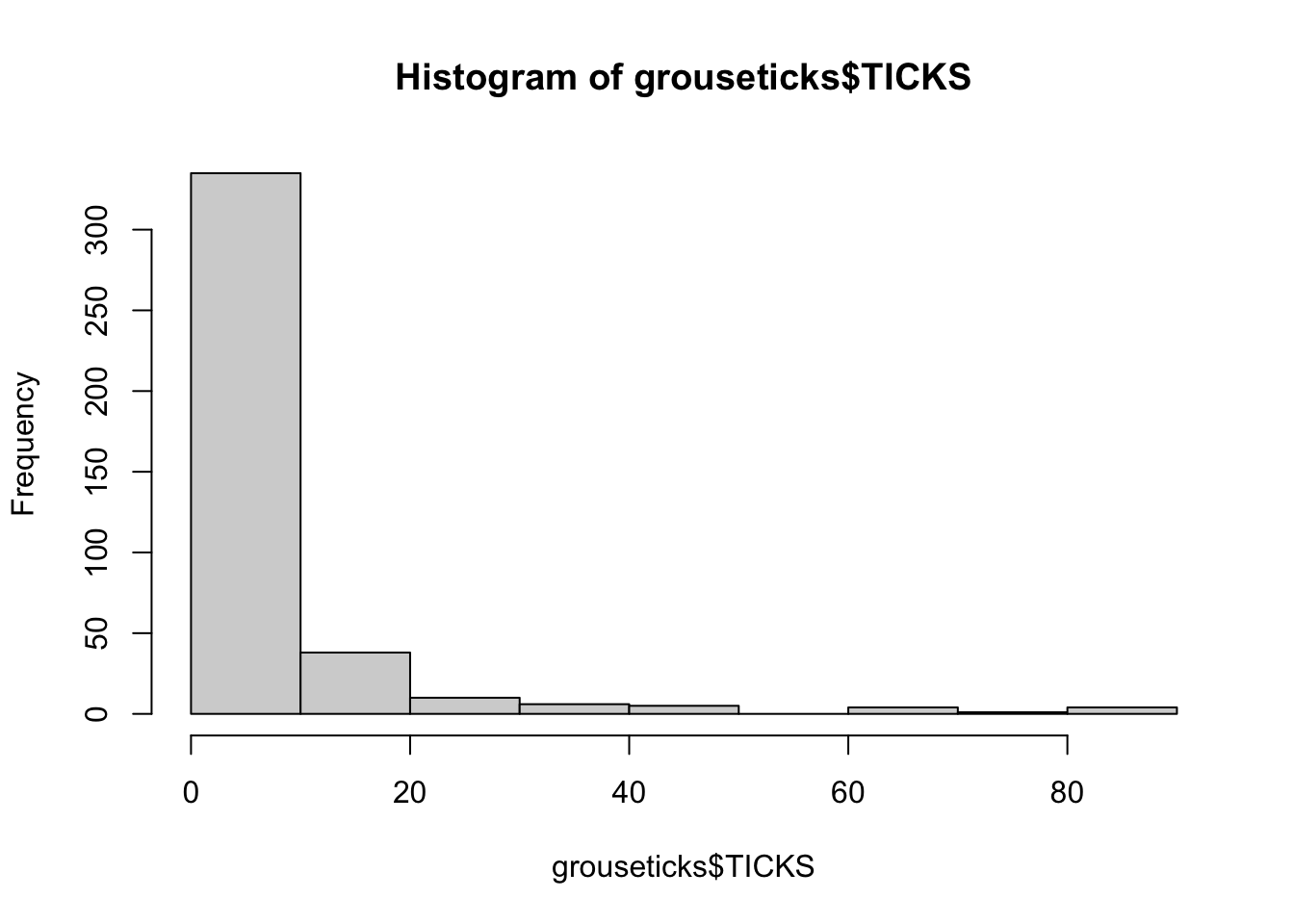
Let’s say our question was whether elevation has an effect on tick abundance in grouse chicks, maybe because lower elevation sites have a larger temperature window of tick activity.

ggplot(grouseticks, aes(x = HEIGHT, y = TICKS)) +  
 geom\_point() +  
 geom\_smooth(method = "lm", se = F) +  
 theme\_bw()

 It does look like there is some effect of elevation on tick abundance, and it *does* seem to be negative (e.g. higher elevation = lower tick abundance).

Like the previous dataset on salamanders, there are a lot of zeros in this dataset

hist(grouseticks$TICKS)



grouseticks %>%  
 filter(TICKS == 0) %>%  
 summarise(count = n(), total = 403, proportion = count/403)

## count total proportion  
## 1 126 403 0.3126551

Out of 403 observations, 126 observations are zeros (31%). That’s a lot!

Again, we will start with fitting a normal Poisson model:

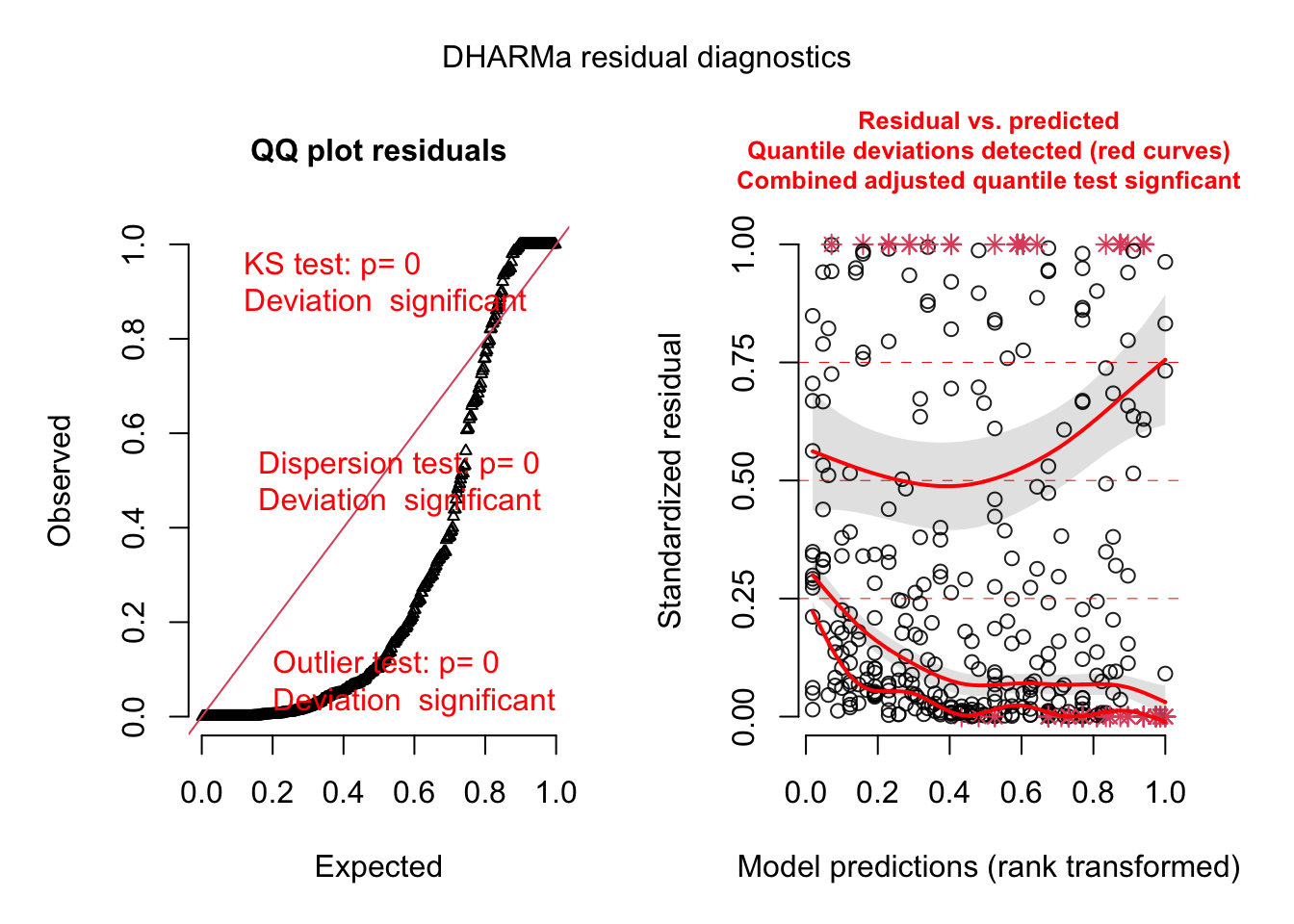
ticks\_poisson <- glm(TICKS ~ HEIGHT,  
 data = grouseticks,  
 family = "poisson")

2. Zero-inflated count data: What is the “best” model? (Skipping again)

If this was our best model (skipping step 2 again), we can assess the residuals:

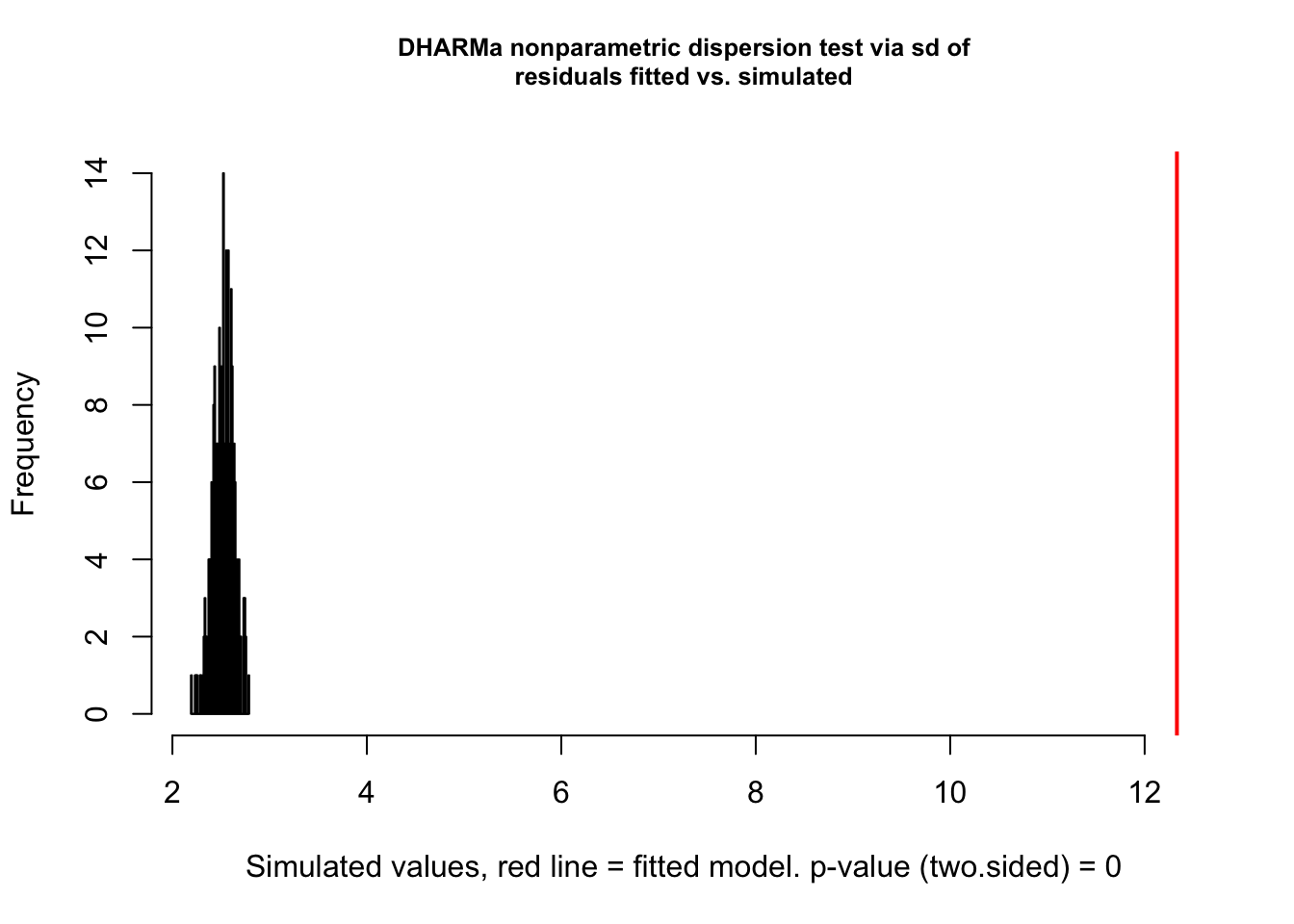
3. Zero-inflated count data: Does my data fit the model assumptions?

simulationOutput <- simulateResiduals(fittedModel = ticks\_poisson, plot = T)



And test for overdispersion again:

testDispersion(ticks\_poisson)

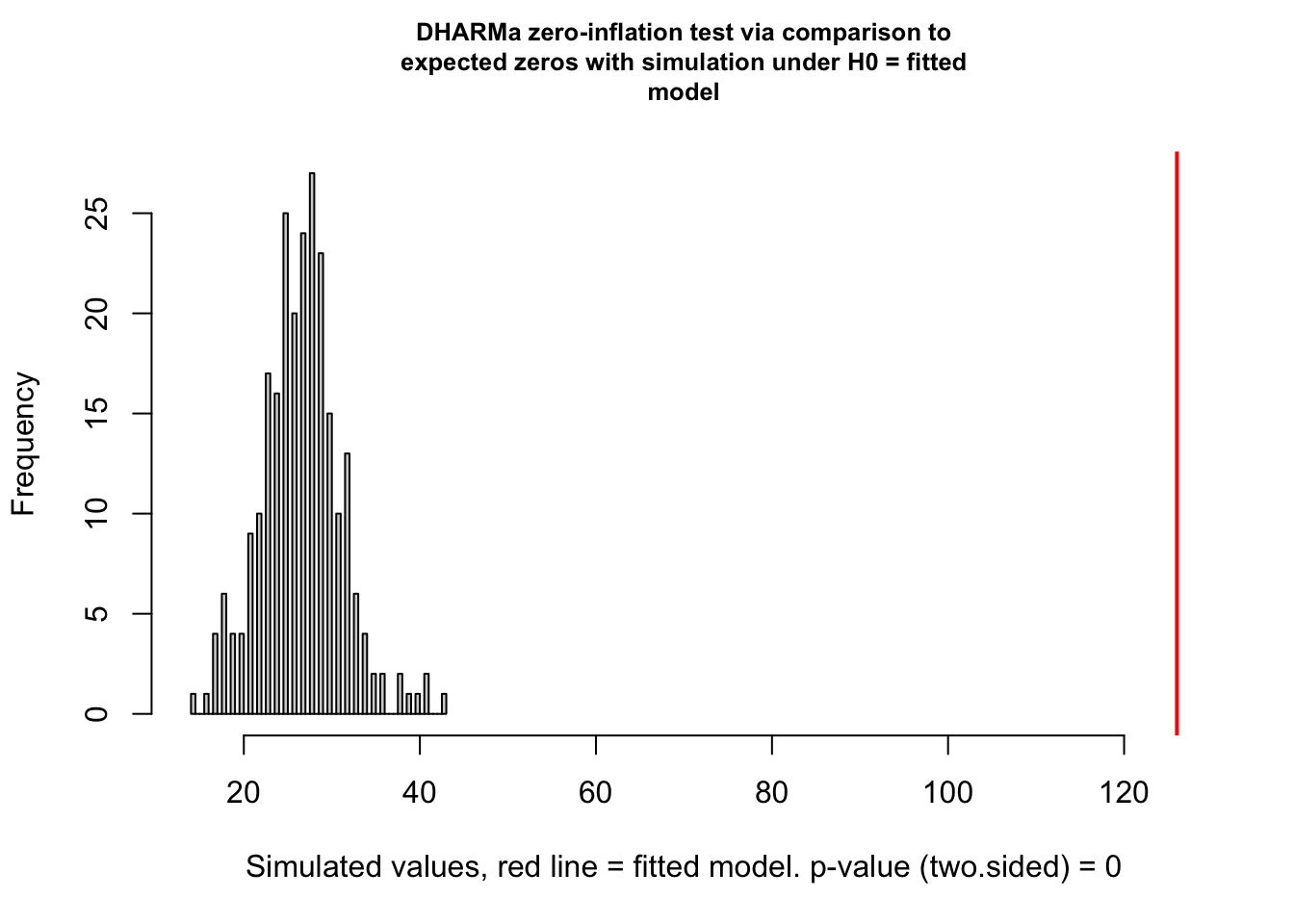


##   
## DHARMa nonparametric dispersion test via sd of residuals fitted vs.  
## simulated  
##   
## data: simulationOutput  
## ratioObsSim = 4.8843, p-value < 2.2e-16  
## alternative hypothesis: two.sided

Again, we have overdispersion. If it was me, I would start by trying to fit a negative binomial model (as before). Even with this number of zeros, often re-fitting with a negative binomial model will fix a lot of issues with zeros. However, for the sake of learning, let’s say that we can’t fit a negative binomial model to these data, or that it doesn’t fit our zero inflation problems.

**How do we determine we have zero inflation issues, you ask?** The DHARMa package has another useful function called testZeroInflation() that I always throw into my model diagnostics for count data.

testZeroInflation(ticks\_poisson)



##   
## DHARMa zero-inflation test via comparison to expected zeros with  
## simulation under H0 = fitted model  
##   
## data: simulationOutput  
## ratioObsSim = 4.7057, p-value < 2.2e-16  
## alternative hypothesis: two.sided

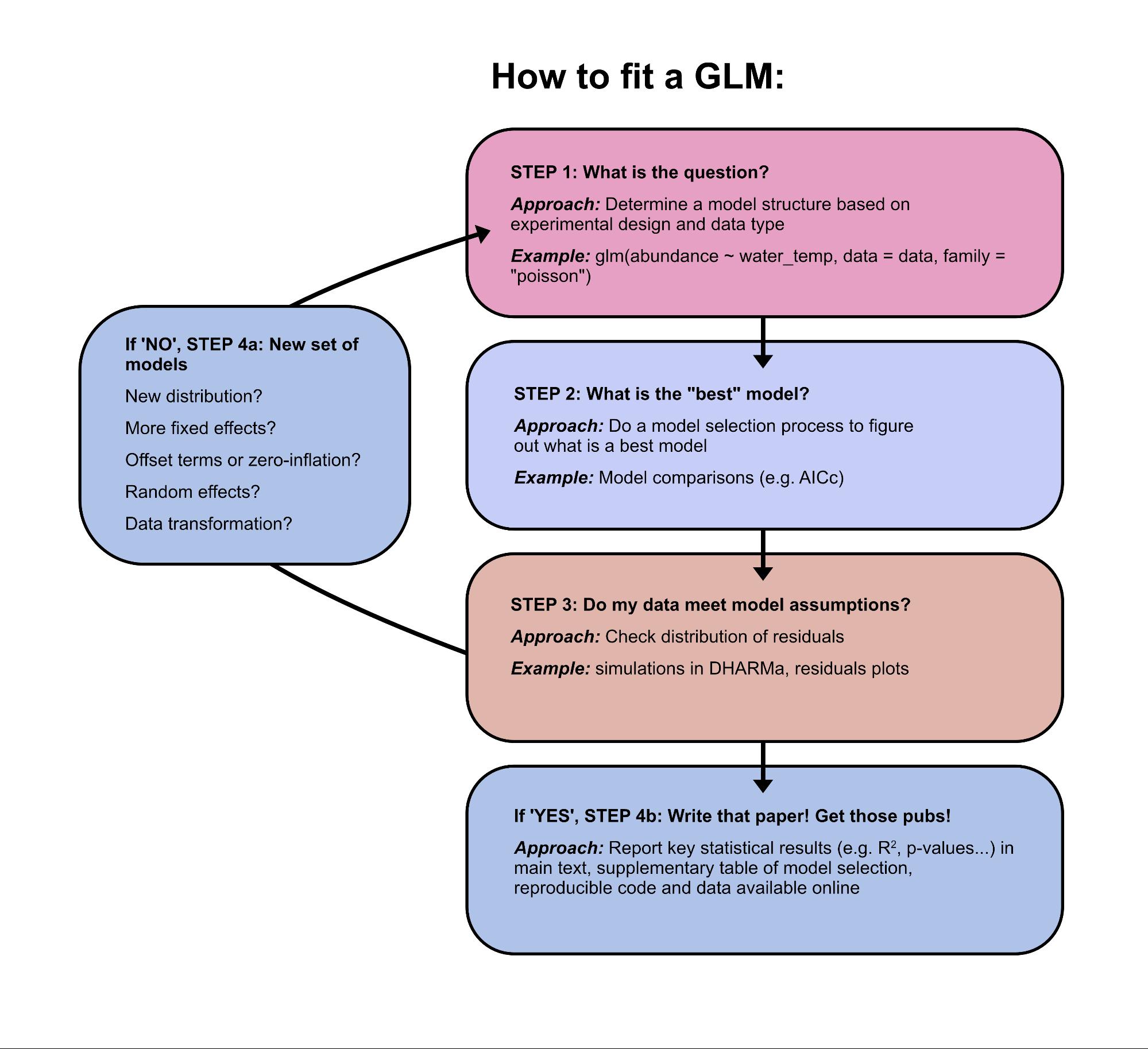
This output operates a lot like the testDispersion() function output. Again, DHARMa has created some simulated expected values for our model, and we see that our distribution has way more zeros than would be predicted by a simulated data distribution based on our model.

1.a. Zero-inflated count data: What is the question?

Again, we have the same question, we just need to fit a different model family that incorporates zero inflation. There are several packages that will let you fit a zero-inflated model (pscl, glmmTMB, ZIM). You can find a tutorial based on pscl at [this website](https://fukamilab.github.io/BIO202/04-C-zero-data.html#key_points). I use glmmTMB because it has a lot of flexibility and lets me extend to mixed models really easily. It is an actively-developing package that many of the pre-eminent mixed model people in ecology swear by.

In glmmTMB, the function to call a glm is glmmTMB() and we can add a zero-inflation term with an added argument (ziformula = 1). There are more complicated ways to call a zero-inflated formula, but this is the basic one.([more here](https://cran.r-project.org/web/packages/glmmTMB/vignettes/glmmTMB.pdf))

Referring again to our flowchart for GLMs, we’re doing Step 1 via Step 4A again, except this time we’re going to fit a zero-inflated model rather than a negative binomial model to see if we can fix our model assumption problems:



ticks\_zi <- glmmTMB(TICKS ~ HEIGHT,  
 data = grouseticks,  
 ziformula = ~1,  
 family = "poisson")

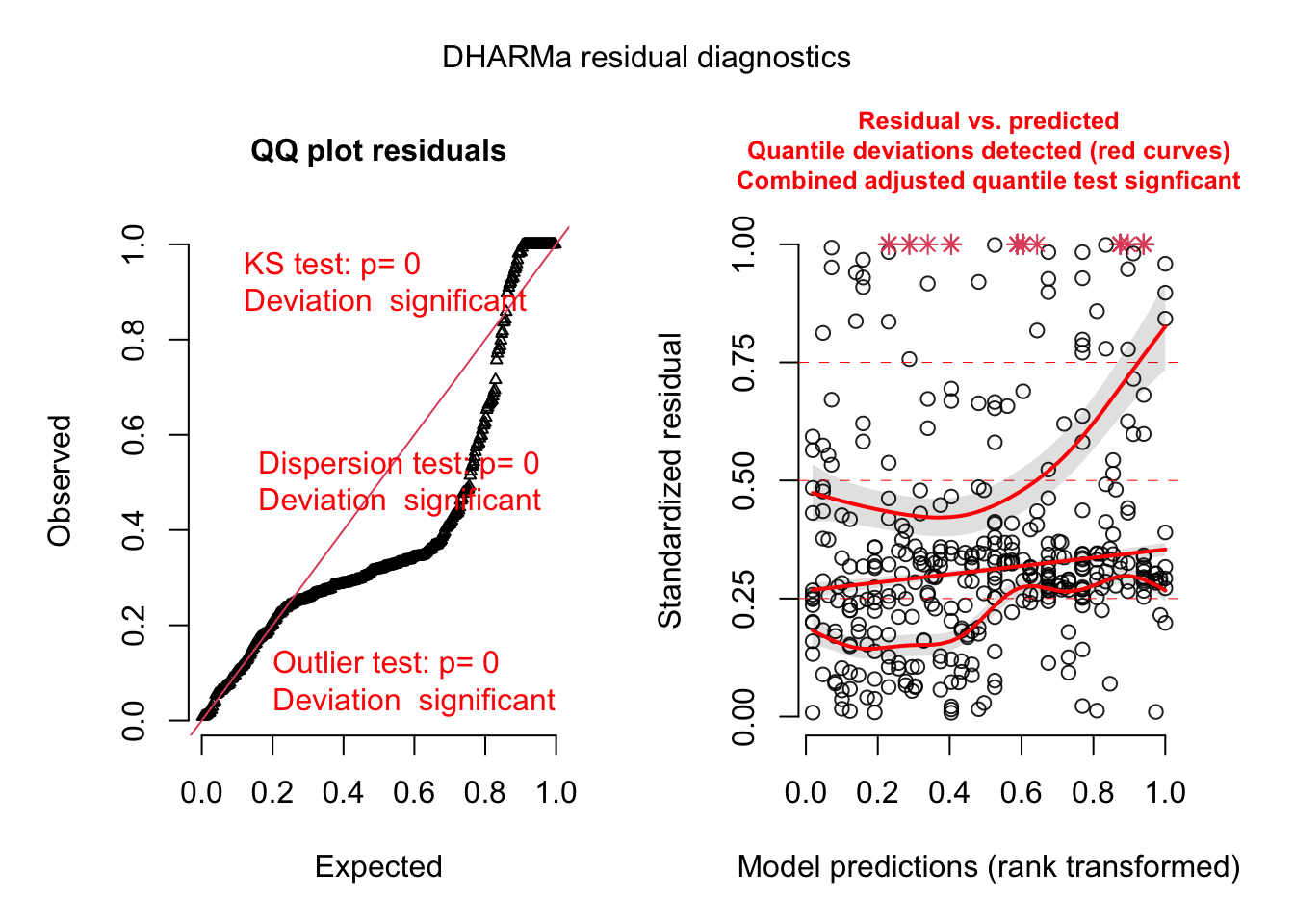
2.a. Zero-inflated count data: What is the “best” model?

Skip!

3.a. Zero-inflated count data: Do my data meet model assumptions?

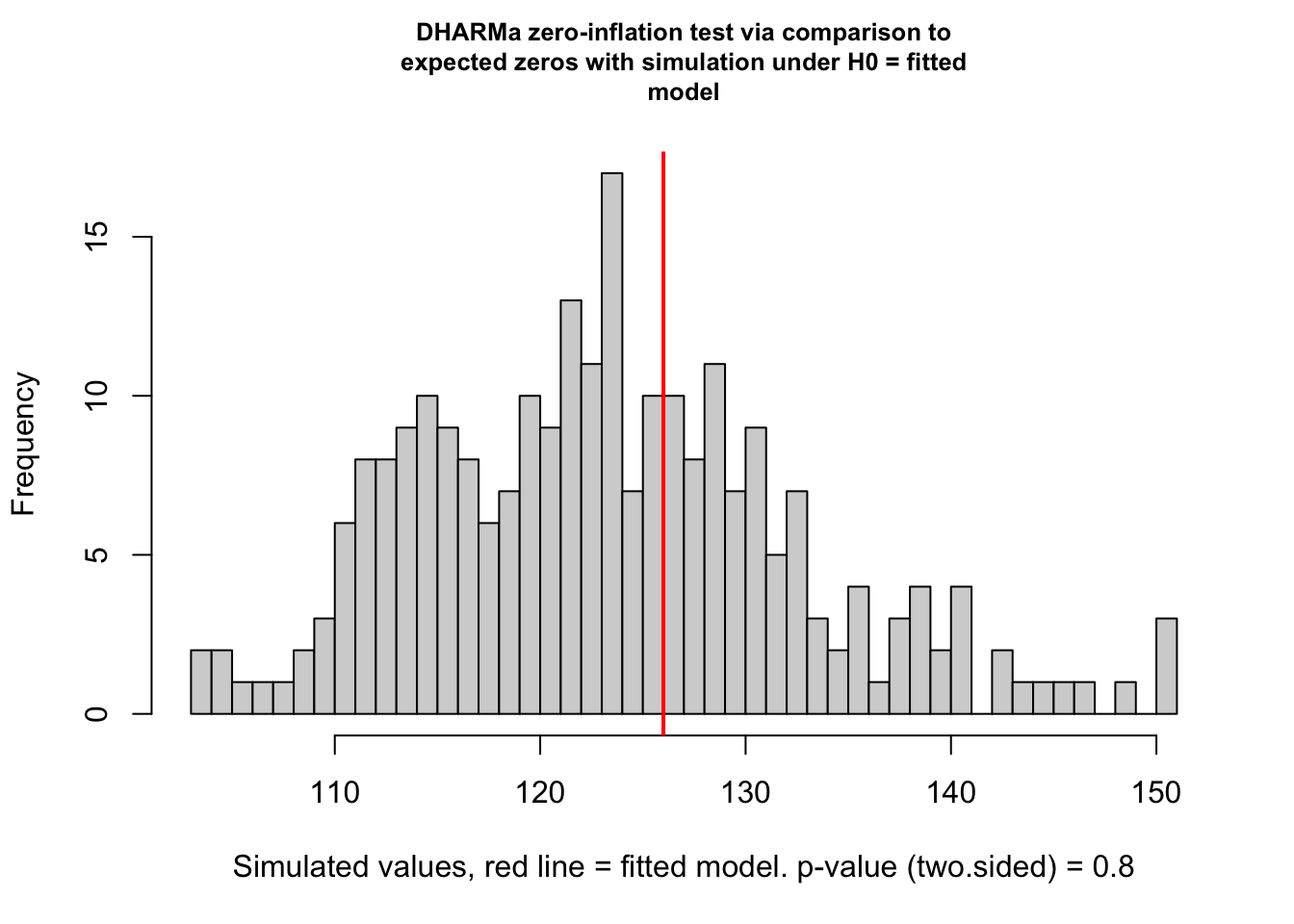
Repeating our diagnostics:

simulationOutput <- simulateResiduals(fittedModel = ticks\_zi, plot = T)



This is **NOT** a good model fit! But, let’s see if we fixed the zero-inflation by including a zero inflation term in our model:

testZeroInflation(ticks\_zi)



##   
## DHARMa zero-inflation test via comparison to expected zeros with  
## simulation under H0 = fitted model  
##   
## data: simulationOutput  
## ratioObsSim = 1.0178, p-value = 0.8  
## alternative hypothesis: two.sided

Yes we have! However, we still need to consider all the math and practical reasons why this model isn’t good (do we need to add variables, do we need a different distribution, do we need to transform?).

Extension 2: Categorical fixed effects

GLMs can sometimes look suspiciously like an ANOVA, which isn’t surprising and makes GLMs really versitile!

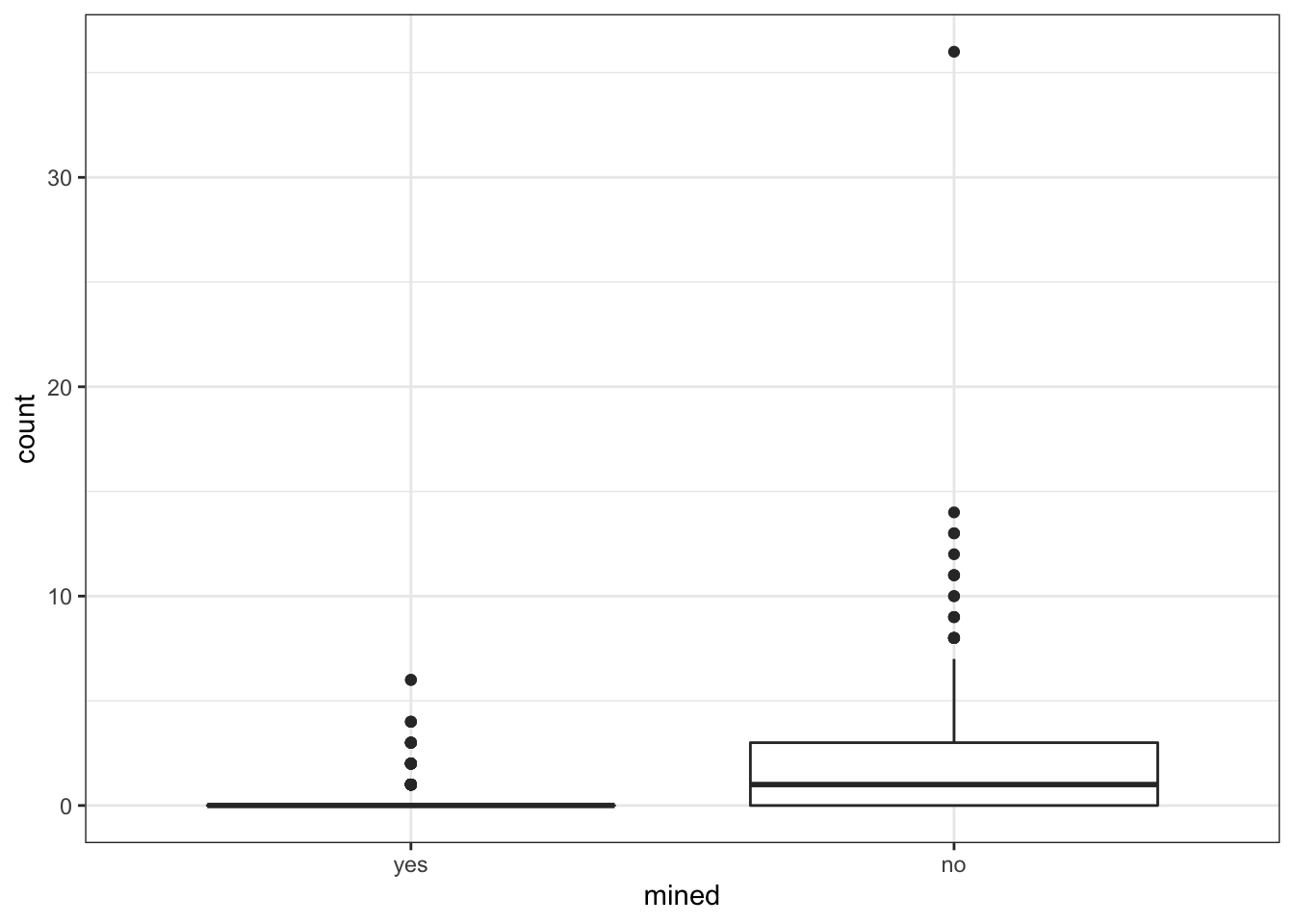
Let’s revisit the example of the salamanders in streams that either were or weren’t below mountaintop removal sites. In this example, if we want to know whether mining has an effect on salamander abundance, we are dealing with a categorical fixed effect (mined vs. unmined). We can approach this really similarly to any other GLM and can also get diagnostics that look suspiciously similar to ANOVA diagnostics. Let’s do it:

1. Categorical fixed effect, count response: what is my question?

Again, we have count data, so we’re going to be playing with Poisson or negative binomial models with this one. Our question is now **What is the effect of presence of mountaintop mining on the abundance of salamanders in streams?"**

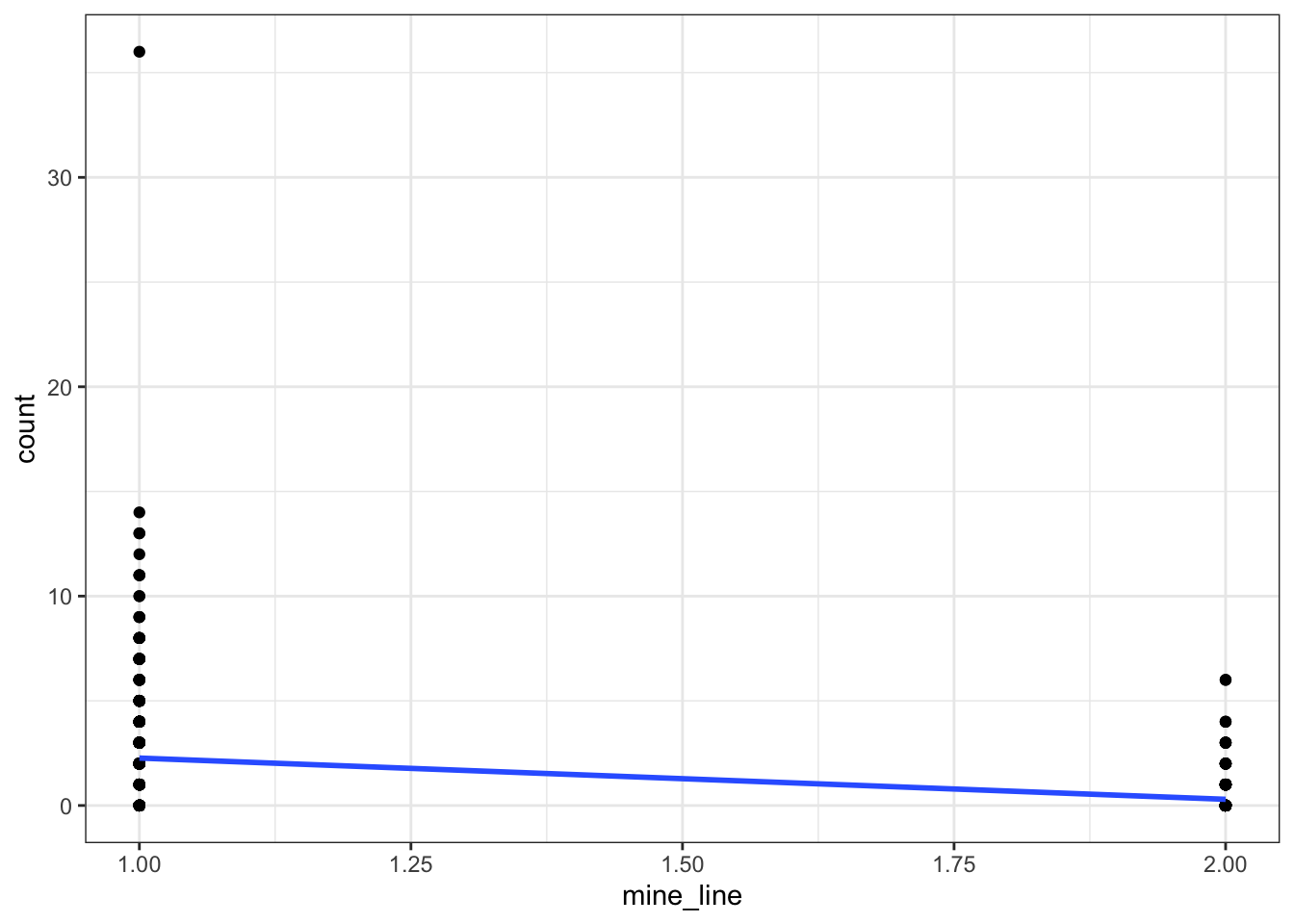
In this situation, it might be more intuitive to look at our data as a boxplot:

ggplot(Salamanders, aes(x = mined, y = count)) +  
 geom\_boxplot() +  
 theme\_bw()



But we could also coerce our yes/no predictor into a continuous-type variable and visualize it as a line graph, setting the “no” level to “1”, and the “yes” level to “2”.

Salamanders %>%  
 mutate(mine\_line = ifelse(mined == "no", 1, 2)) %>%  
 ggplot(aes(x = mine\_line, y = count)) +  
 geom\_point() +  
 geom\_smooth(method = "lm", se = FALSE) +  
 theme\_bw()

 In both of these graphs, it looks like mining may have a positive effect on salamander abundance.

In our model, the fixed effect will be mined (a binary yes/no) and our response will be count from the Salamanders dataset:

cat\_poisson <- glm(count ~ mined,   
 data = Salamanders,  
 family = "poisson")

We can look at our model summary again to see that the effect of mining does seem to be important for salamander abundance.

summary(cat\_poisson)

##   
## Call:  
## glm(formula = count ~ mined, family = "poisson", data = Salamanders)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -2.1283 -0.9459 -0.7687 -0.1796 11.4753   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.2192 0.1048 -11.63 <2e-16 \*\*\*  
## minedno 2.0368 0.1109 18.36 <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: 2120.7 on 643 degrees of freedom  
## Residual deviance: 1575.2 on 642 degrees of freedom  
## AIC: 2302.5  
##   
## Number of Fisher Scoring iterations: 6

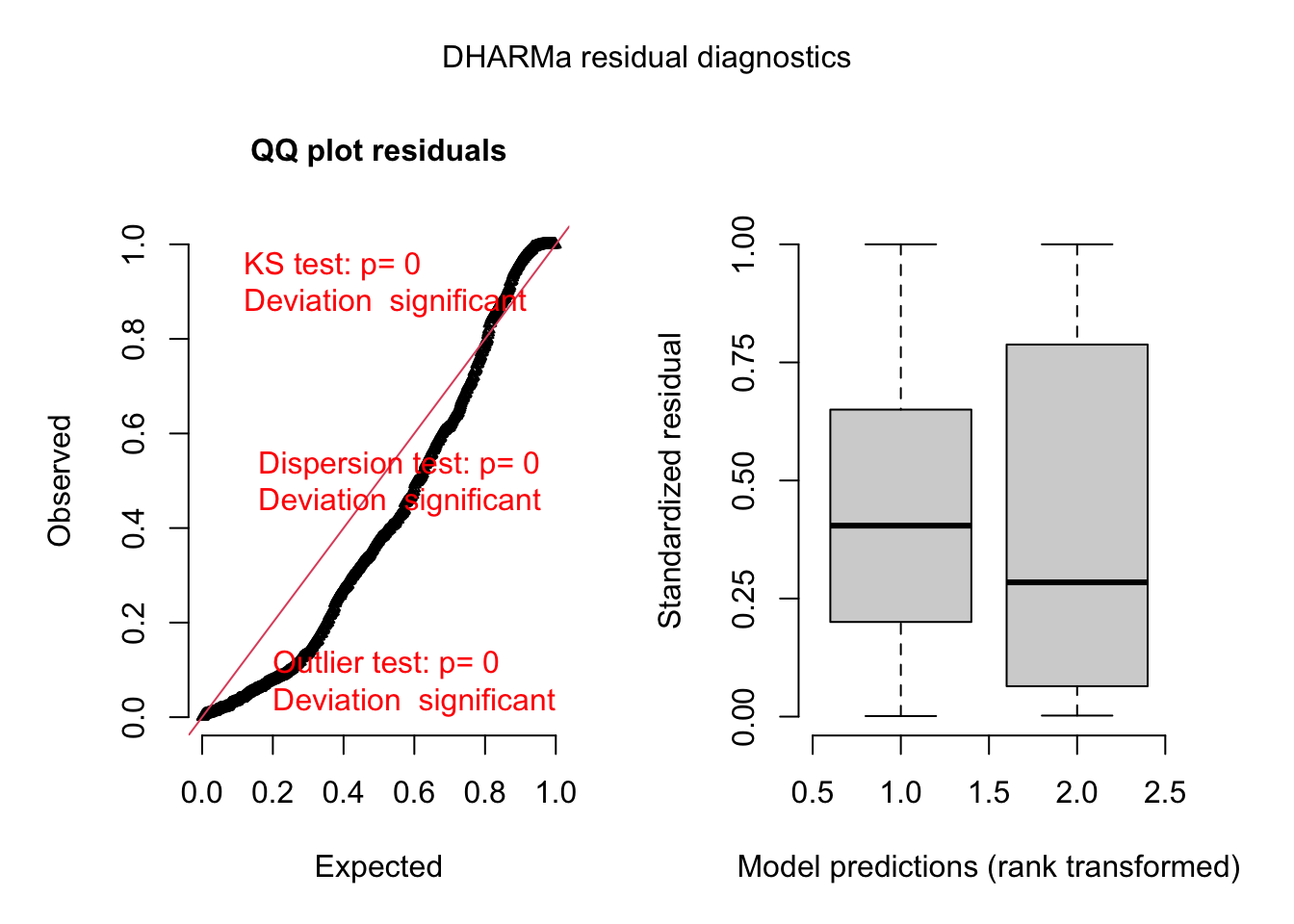
2. Categorical fixed effect, count response: what is the “best” model?

Covered next week

3. Categorical fixed effect, count response: do data meet assumptions?

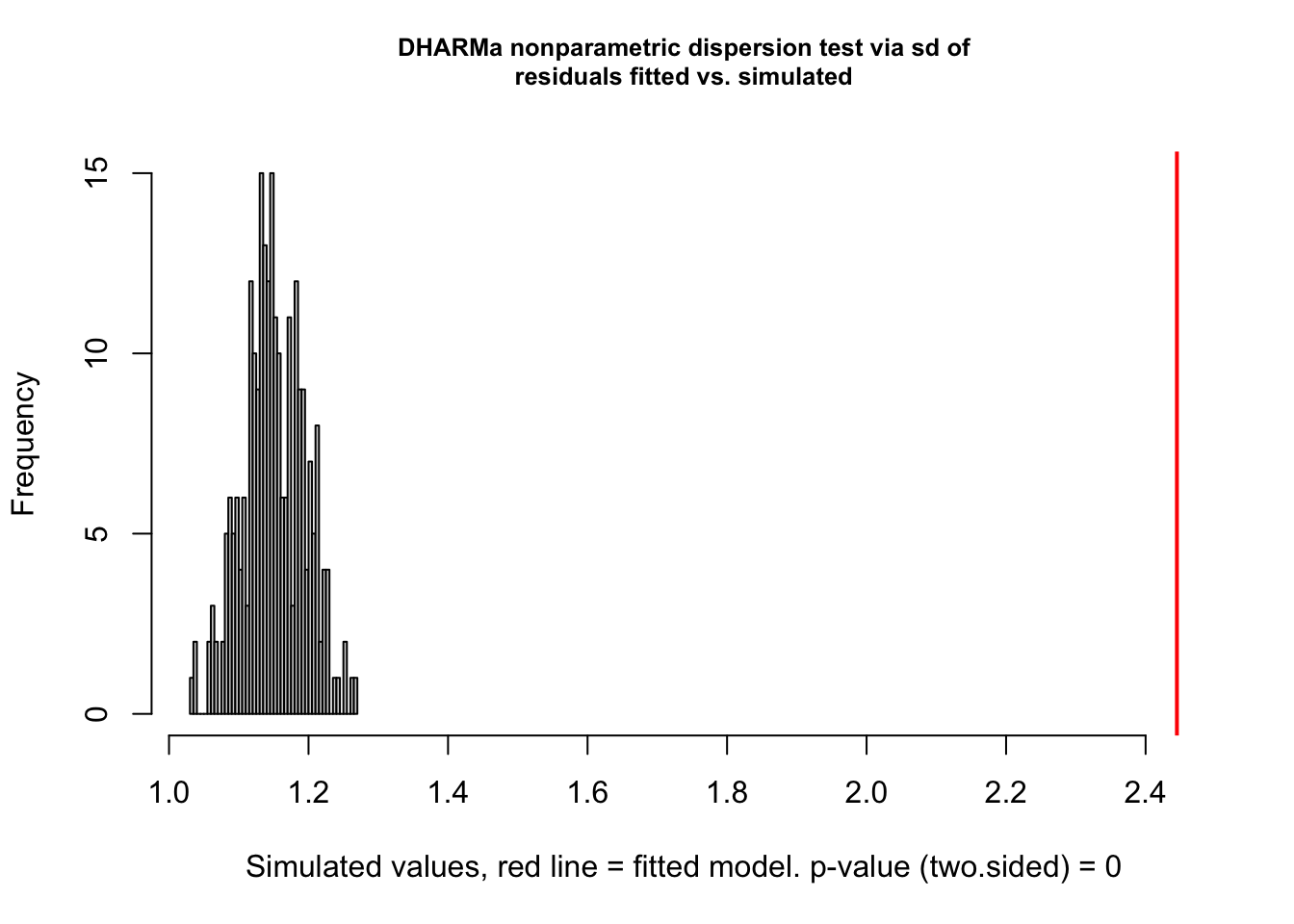
This probably seems repetitive by now - but we can also use DHARMa for this kind of model.

simulationOutput <- simulateResiduals(fittedModel = cat\_poisson, plot = T)



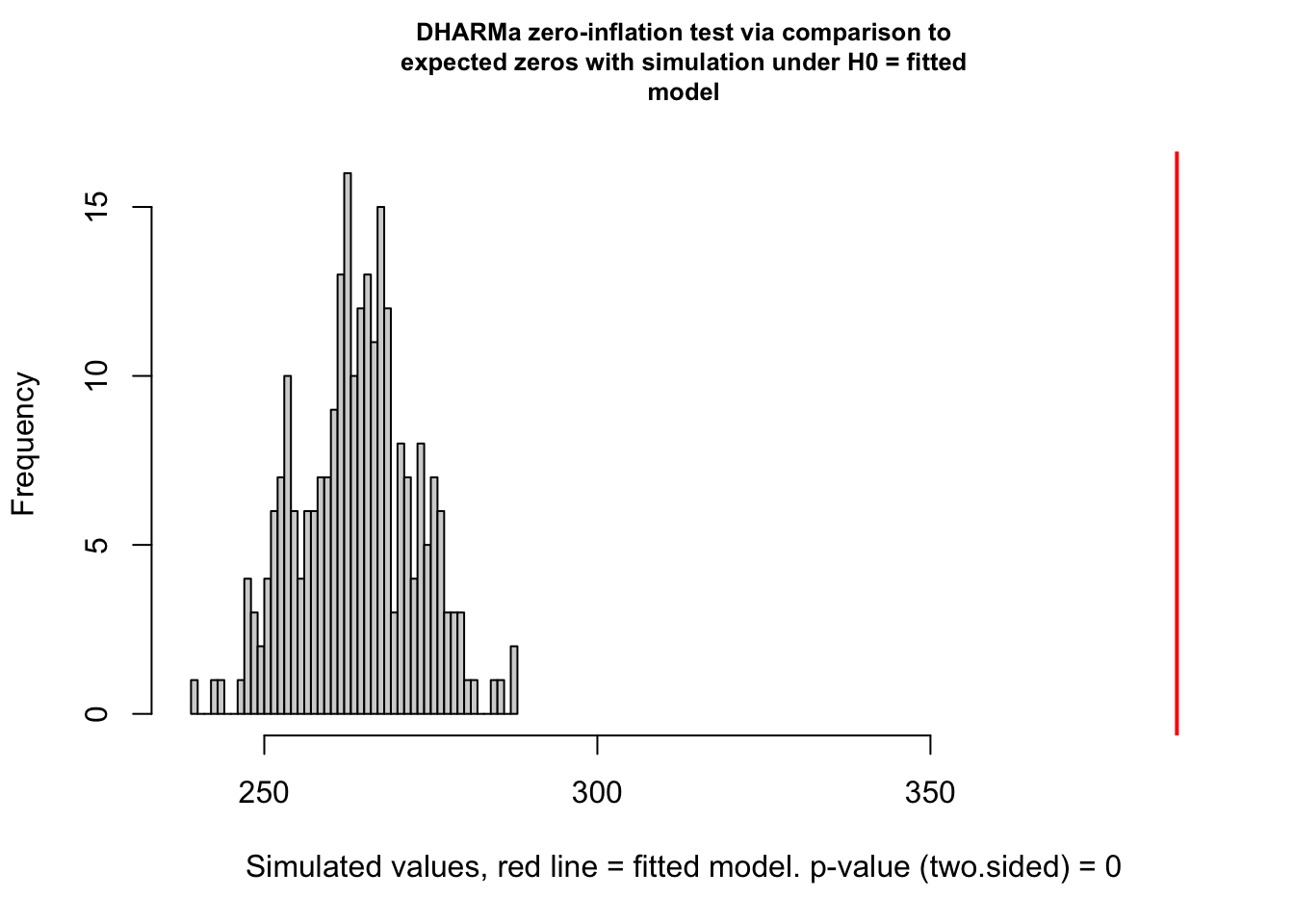
This model also has so many of the beautiful model assumption failures we’re getting good at assessing.

testDispersion(simulationOutput)



##   
## DHARMa nonparametric dispersion test via sd of residuals fitted vs.  
## simulated  
##   
## data: simulationOutput  
## ratioObsSim = 2.1261, p-value < 2.2e-16  
## alternative hypothesis: two.sided

testZeroInflation(simulationOutput)



##   
## DHARMa zero-inflation test via comparison to expected zeros with  
## simulation under H0 = fitted model  
##   
## data: simulationOutput  
## ratioObsSim = 1.4637, p-value < 2.2e-16  
## alternative hypothesis: two.sided

So we could approach it like we’ve been approaching any other model when it comes to fixing this!

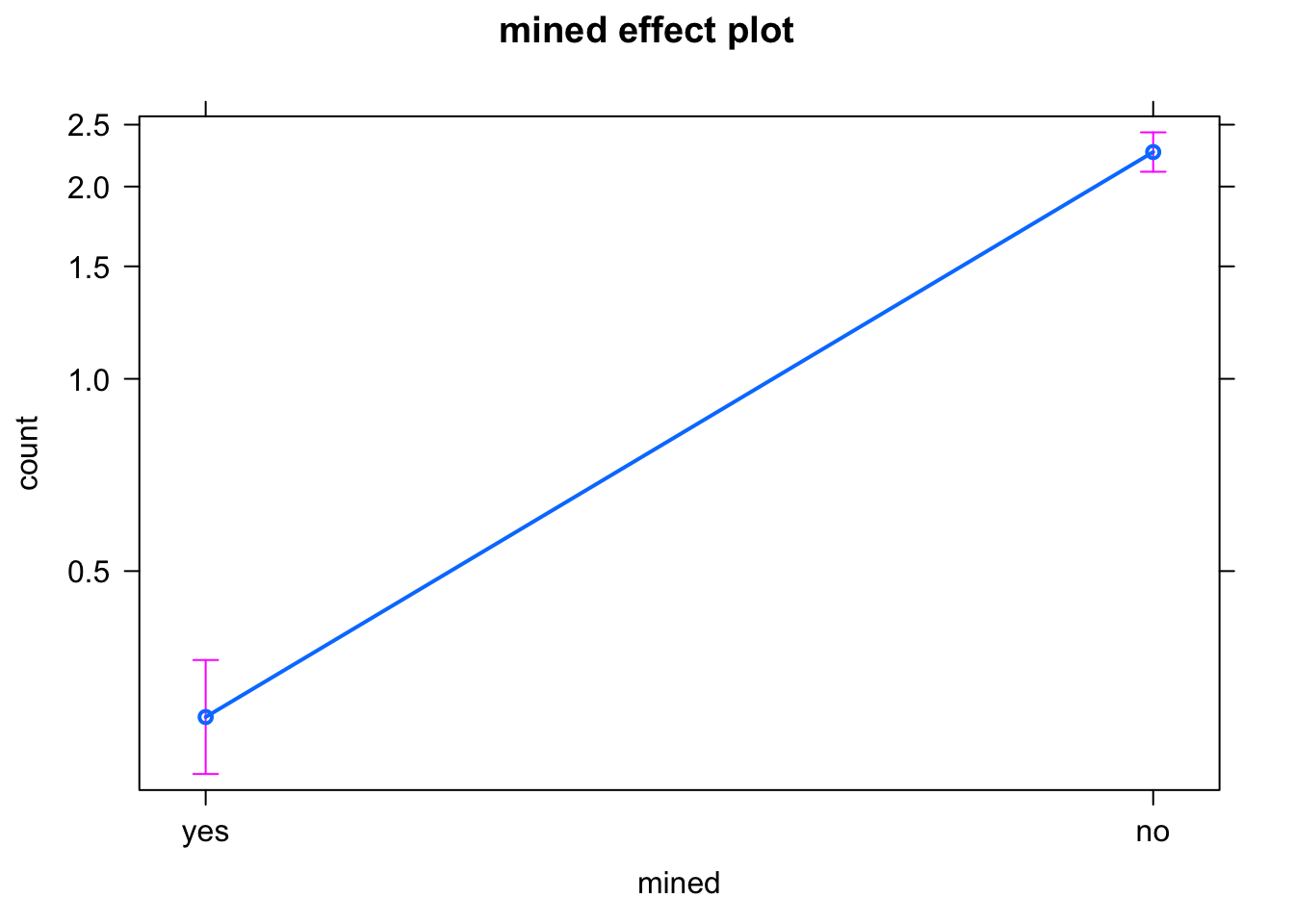
**On your own**: How would you attempt to fix some of these model assumption failures?

Catagorical predictor wrap-up: post-hoc and visualizations

In this example, our predictor had two categories, and so we can report model results (including the p-value) for the between-group differences (ala ANOVA) from the summary() for the model. However, if you had a categorical variables with more than two levels, there are some cool ways to do pair-wise comparisons between those levels (for example, if I had “full mountaintop removal”, “partial mountaintop removal”, and “no mountaintop removal” mining categories for the Salamander dataset).

Again, at this point I always like to do a quick visual of my data, which I can still do with the allEffects call:

plot(allEffects(cat\_poisson))



If I had more than two levels for my predictor, I could use the emmeans package and function to determine the pair-wise differences between groups.

em <- emmeans(cat\_poisson, "mined")

The emmeans() function calculates the marginal means (mean of a group given all the other groupings in the model) based on a specified predictor (here, “mined”):

em

## mined emmean SE df asymp.LCL asymp.UCL  
## yes -1.219 0.1048 Inf -1.425 -1.014  
## no 0.818 0.0362 Inf 0.746 0.889  
##   
## Results are given on the log (not the response) scale.   
## Confidence level used: 0.95

and then you can compare these means to each other:

pairs(em)

## contrast estimate SE df z.ratio p.value  
## yes - no -2.04 0.111 Inf -18.363 <.0001   
##   
## Results are given on the log (not the response) scale.

Flowchart for GLM and GLMM model selection for count data

Here’s a flow chart of my complete model selection process and troubleshooting for count data (for GLMM, but can easily be used for GLM as well)

