Generalized Linear Modeling Part III: Binomial GLMs

ANTH 3720-001 Archaeological and Forensic Science Lab Methods: Data Analysis with R

Elic Weitzel

Jan. 29-31, 2021

If you would like the original R Markdown file, find it on my GitHub page at https://github.com/weitzele/Basic-R-Tutorial.

1 Binomial Generalized Linear Models

And finally, the fourth family of models that we'll discuss are binomial family generalized linear models.

1.1 Binomial Violations of Model Assumptions

Binomial GLMs are useful when you're modeling either presence/absence data or proportions. Similar to the count data from above, such data violate several assumptions of Gaussian linear models by definition. As proportions and presence/absence data are bounded by zero and one, residuals from such models are unlikely to be normally distributed (especially for presence/absence data). Similarly, the deterministic model fits from a Gaussian linear model will predict unrealistic values beyond the range of possible values for such types of data. And similar to count data, presence/absence and proportion data are commonly heteroskedastic, which is why the binomial distribution accomodates that fact.

1.2 Fitting Binomial GLMs

Let's go back to some data from our trusty `archdata` package and explore ceramic types from a pueblo in New Mexico. This pueblo, called Pueblo San Cristobal, has ten stratigraphic layers which are each 1 foot in depth.

```r
library(archdata)

data("Nelson")
```

Let's try to understand the occurrence of a rare type of pottery through time at this site. Type III ceramics - a three-color glazed ceramic - is the least common type at the pueblo. We might wonder what the probability of recovering a three-color glazed ceramic is for each stratigraphic unit, and we could model this to find out.

```r
#depth values
Nelson$Depth
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
#Type III ceramic counts
Nelson$Type_III
```

```
##  [1] 5 6 3 0 0 1 0 0 0 0
```

But as we can see in our data frame, we only have the counts of Type III ceramics by stratum. We're not interested in the frequency of Type III ceramics right now, only the probability of a stratum containing one. So let's convert this column into a new one that simply reports presence or absence of Type III ceramics. Let's call this new column `$PA` (presence/absence), and in this column, let's note whether Type III is present with a 1 or absent with a 0.

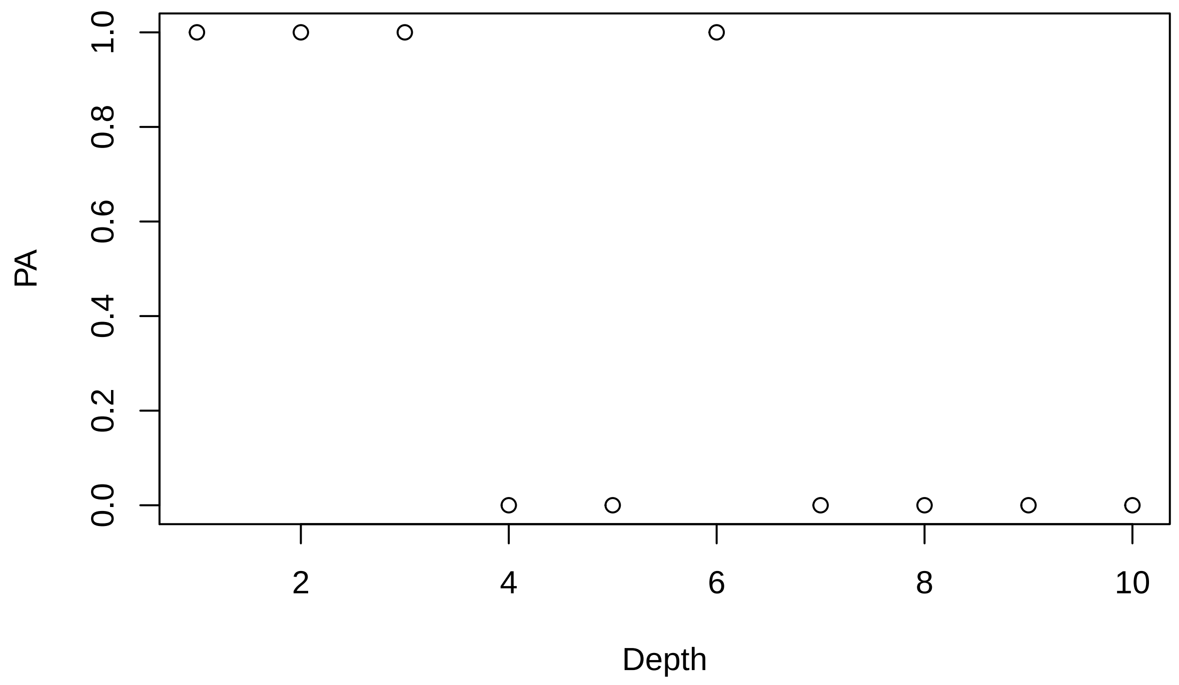
We can use a `dplyr` function called `case_when()` to do this by setting any Type_III value greater than 0 to a 1 in our new `$PA` column, while keeping those which are equal to 0 as 0. Make sure you have the `dplyr` package installed and loaded with the `library` function, or the `case_when` function won't work.

```
library(dplyr)

Nelson$PA <- case_when(Nelson$Type_III > 0 ~ 1,
                       Nelson$Type_III == 0 ~ 0)
```

Now that we've manipulated our data and calculated the variable we're interested in predicting, let's visualize this pattern.

```
plot(PA ~ Depth, data = Nelson)
```



It looks like Type III ceramics are more common higher up in the site, between 0-6 ft deep. There are more 0 values, denoting absence of Type III, at deeper levels in the site. But let's model this and see what's going on.

Since our data are presence/absence, taking only values of 0 and 1, they're bounded by 0 and 1. This makes them perfect for a binomial family GLM, so let's specify that `family = binomial` in our glm function.

But we also need to specify a new type of link function: *logit*. A logit function basically takes 0s and 1s and extends them to negative and positive infinity. You can check this out using the R function `qlogis()`, which runs a logit transformation.

```
qlogis(0)
```

```
## [1] -Inf
```

```
qlogis(1)
```

```
## [1] Inf
```

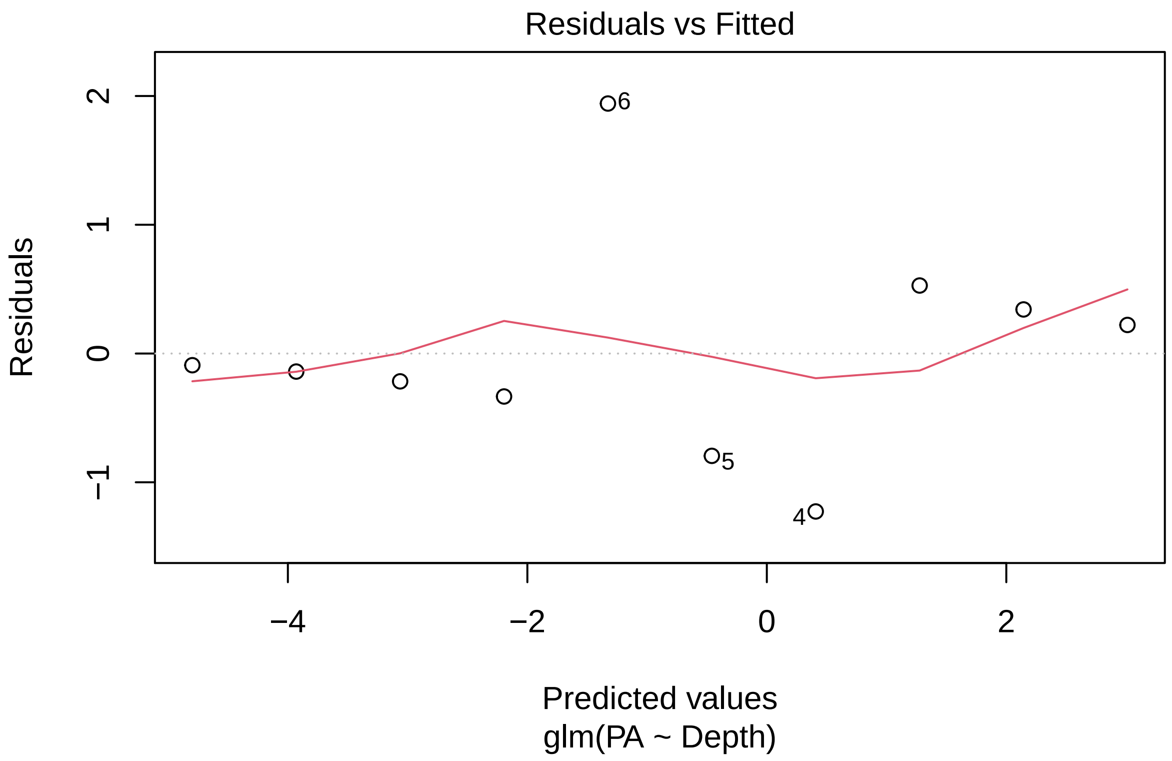
The logit of 0 is negative infinity, and the logit of 1 is positive infinity! This is how a logit transformation takes data bounded by 0 and 1 and permits such data to be used in a linear model. This accomplishes the same thing that a log link function did for log-normal and Poisson data above.

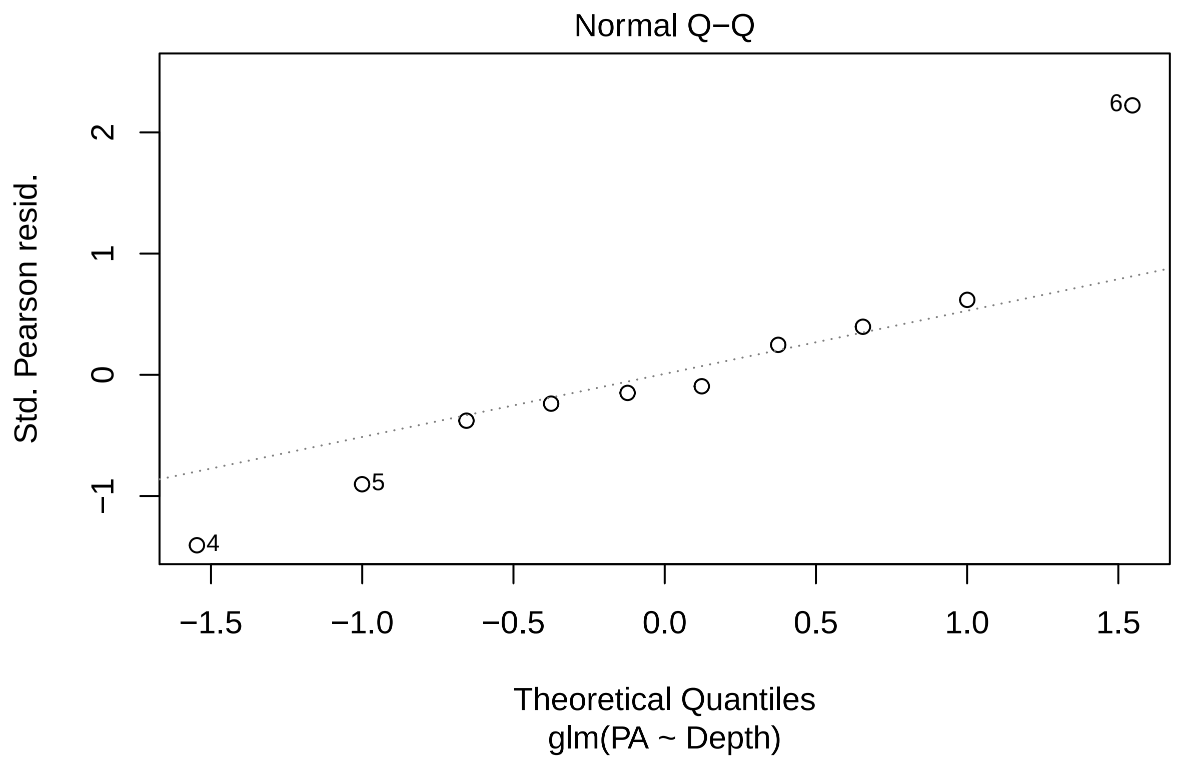
So now that we understand what our model is doing, let's built it!

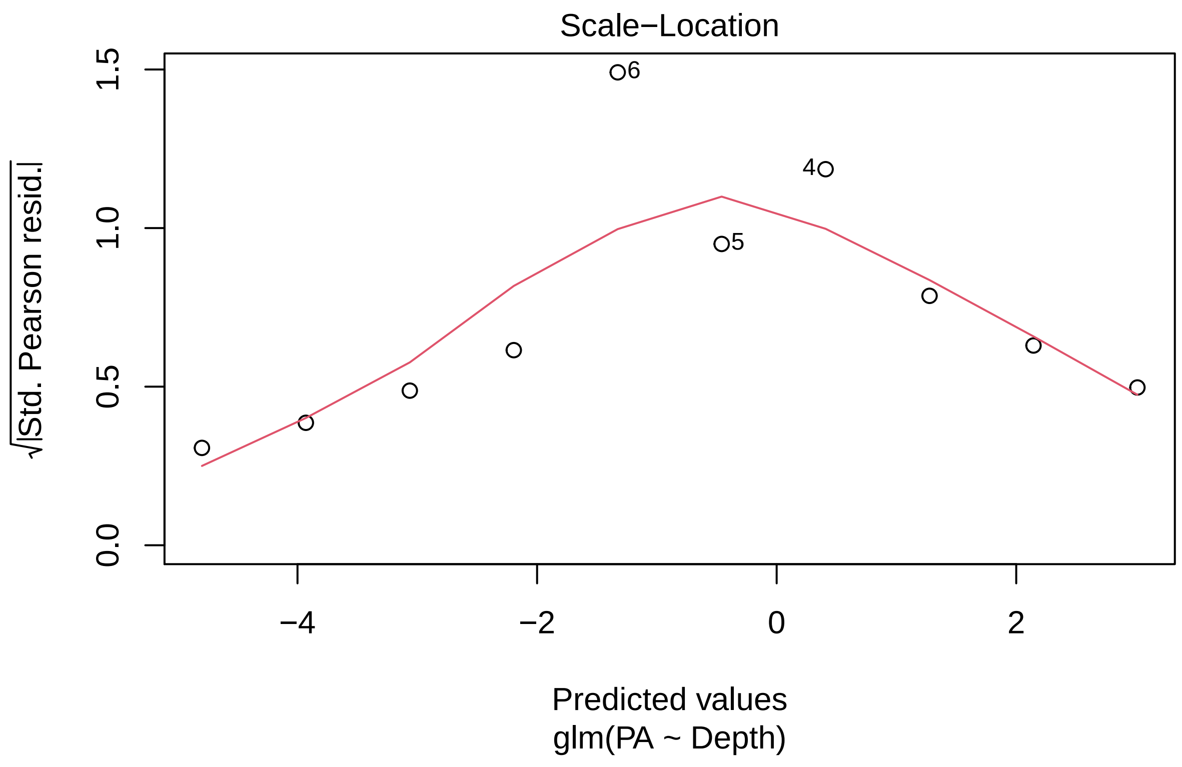
```
type3.mod <- glm(PA ~ Depth, data = Nelson, family = binomial(link = "logit"))
```

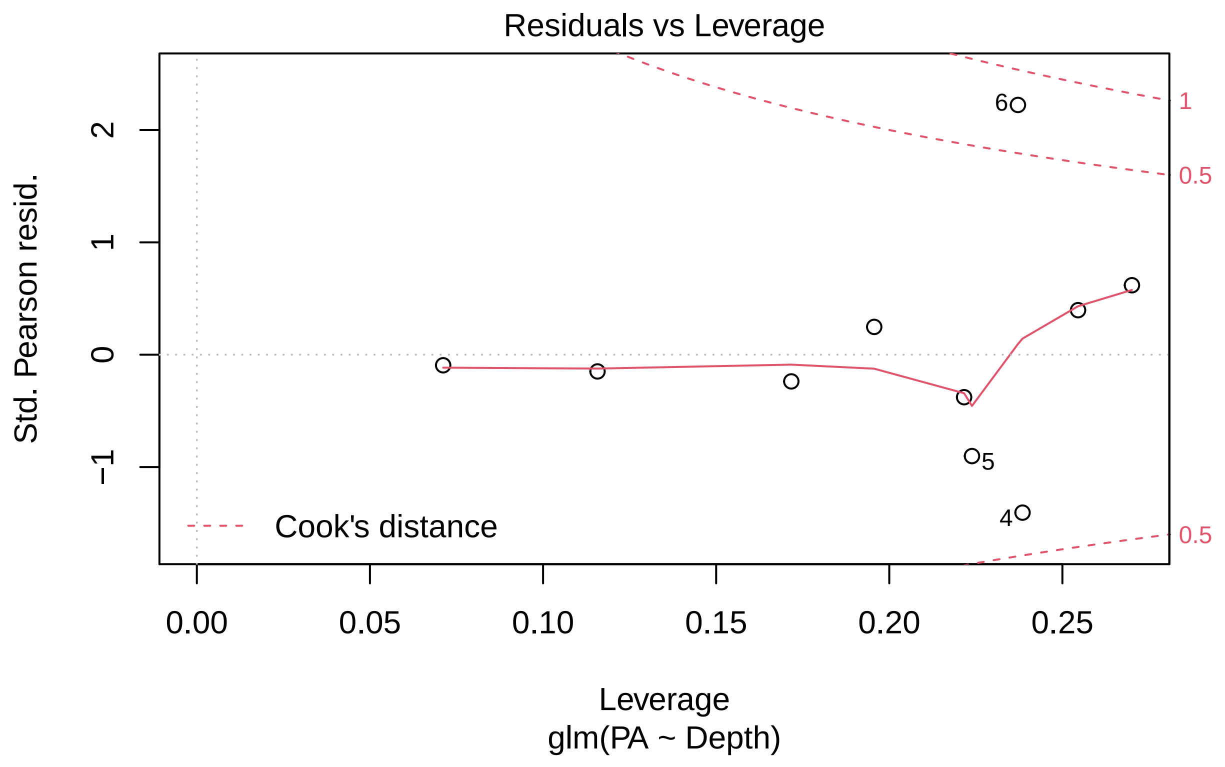
If we were to inspect the model diagnostic plots, we would see that our data very clearly violate the assumptions of Gaussian linear modeling. But that's why we're using a GLM! So it's important to remember that these diagnostic plots are based on the assumptions of a Gaussian linear model, not a GLM. These plots really don't have much use for determining anything about heteroskedasticity or normality of residuals because we already know that our residuals are heteroskedastic and binomially distributed, not Gaussian. But we can inspect the plots anyways, as they could clue us in to any non-independence of residuals or non-linearity (but these things are often apparent simply from plotting the data, too).

```
plot(type3.mod)
```


Residuals vs Fitted
Residuals
Predicted values
glm(PA ~ Depth)
2
1
0
−1
−4
−2
0
2
6
5
4


Normal Q−Q
Std. Pearson resid.
2
1
0
−1
6
5
4
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
Theoretical Quantiles
glm(PA ~ Depth)


Scale−Location
√|Std. Pearson resid.|
0.0
0.5
1.0
1.5
−4
−2
0
2
6
4
5
Predicted values
glm(PA ~ Depth)



1.3 Summary and Interpretation

So now let's interpret the model summary.

```
summary(type3.mod)
```

```
##
## Call:
## glm(formula = PA ~ Depth, family = binomial(link = "logit"),
##     data = Nelson)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.3551  -0.4203  -0.1627   0.4305   1.7676
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   3.8795     2.5650   1.512   0.1304
## Depth        -0.8677     0.5187  -1.673   0.0944 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
```

```
##     Null deviance: 13.460  on 9  degrees of freedom
## Residual deviance:  7.108  on 8  degrees of freedom
## AIC: 11.108
##
## Number of Fisher Scoring iterations: 5
```

We can see that our intercept estimate is 3.88 and our depth coefficient is -0.868. But! Remember that these are on the link scale! These values are not real-world values, but logit-link values. To get these values back into reality, we need to use the inverse logit function, which in R is `plogis()`.

Let's then un-transform our intercept value to see what the predicted probability of finding a Type III ceramic would be for a depth of 0.

```r
plogis(type3.mod$coefficients[1])
```

```
## (Intercept)
##   0.9797578
```

That value of 0.98 makes sense given the plot from above: Type III ceramics seem to only occur later in time (i.e. at higher depths). But what if we wanted to predict the probability of recovering a Type III ceramic at a depth of 5 feet? We'd use these coefficients in our model equation, as usual, but remember to then take the inverse logit of the answer we get. Remember that you cannot take the inverse logit of each individual term and add them together! You must first add the terms, and then take the inverse logit.

```r
plogis(type3.mod$coefficients[1] + (type3.mod$coefficients[2] * 5))
```

```
## (Intercept)
##   0.3871837
```

It looks like at a depth of 5 ft, we'd have a 39% chance of finding a Type III ceramic. Let's use the `predict()` function to confirm this.

```r
predict(type3.mod, data.frame("Depth" = 5), type = "response")
```

```
##         1
## 0.3871837
```

Indeed, we did our math correctly and the `predict` function confirms our result of 0.39. Since the intercept, for a depth of 0, was 0.98 and the predicted value for 5 feet was 0.39, our trend is downwards. There is a negative relationship between depth and probability of a Type III ceramic. This is evident in the summary output by the negative coefficient for depth.

Now that we understand the coefficients and the direction of our model fit, let's calculate a model p-value and D^2 value.

```r
library(lmtest)
library(modEvA)

lrtest(type3.mod)
```

```
## Likelihood ratio test
##
## Model 1: PA ~ Depth
## Model 2: PA ~ 1
##   #Df  LogLik Df  Chisq Pr(>Chisq)
## 1   2 -3.5540
## 2   1 -6.7301 -1 6.3522    0.01172 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
Dsquared(type3.mod)
```

```
## [1] 0.4719223
```

The `lrtest` function tells us that our model is significant ($X^2 = 6.35$, df $= 1$, $p < 0.05$), meaning that is it unlikely that we'd get this result if an intercept-only null model were "true".

Then our D^2 value of 0.472 tells us that there is a moderately strong relationship between the probability of Type III ceramic recovery and depth. Including depth in our model accounts for 47% of the residual deviance, which is pretty good for archaeological data.

1.4 Plotting

Finally, let's plot this model. We'll need to use the same `predict()` function we did previously. We need to specify a range of predictor data that we'll feed into the function, ideally extending beyond our real data and with many more point values. The `predict` function will predict over these predictor values based on our model object. We'll also need to specify that we want the standard errors (`se = TRUE`), and that `type = "link"` not `type = "response"`. This is because we cannot calculate 95% confidence intervals on the response scale - we have to calculate those on the link scale, then convert them later.

```r
#create the newdata object over which to predict
new.depths <- data.frame("Depth" = seq(-5, 15, length.out = 100))

#predict over new.depths
pa.pred <- predict(type3.mod, new.depths, type = "link", se = TRUE)
```

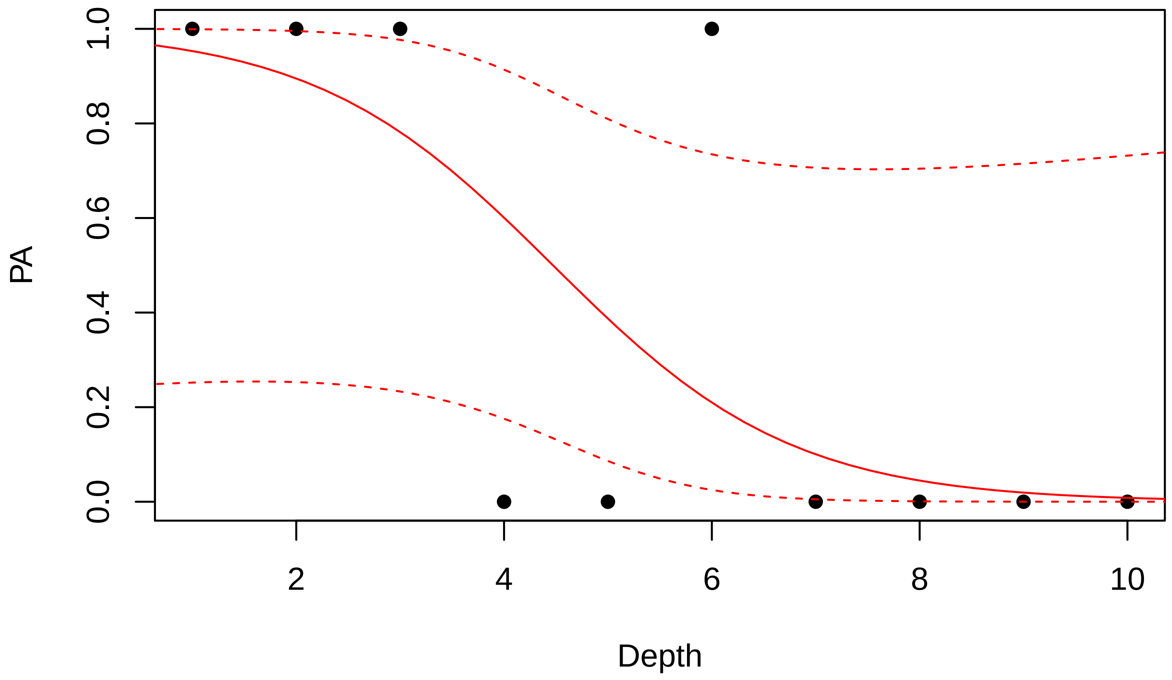
Now that we've got our predicted model fit and standard errors on the link scale (saved in the pa.pred object), let's construct the confidence intervals.

```r
#calculate the confidence intervals and save these values alongside the model fit
#making sure to un-transform the values from the link scale to the response scale
#using the inverse logit function plogis()
type3.mod.fit <- data.frame("fit" = plogis(pa.pred$fit),
                            "upper" = plogis(pa.pred$fit + (pa.pred$se.fit * 1.96)),
                            "lower" = plogis(pa.pred$fit - (pa.pred$se.fit * 1.96)))
```

Now we can visualize the model, confidence intervals, and data points all together.

```r
plot(PA ~ Depth, data = Nelson, pch = 16, col = "black")

lines(type3.mod.fit$fit ~ new.depths$Depth, col = "red")
lines(type3.mod.fit$upper ~ new.depths$Depth, col = "red", lty = 2)
lines(type3.mod.fit$lower ~ new.depths$Depth, col = "red", lty = 2)
```



And voila! We now have a beautiful binomial family GLM showing that as depth increases, the probability of finding a Type III ceramic goes down.

As a final note, remember that your data for binomial GLMs don't actually have to be discrete values of either 0 or 1. It's a somewhat flexible distribution in this sense, so you can plug in proportions as your response variable - continuous values between 0 and 1. You will get a warning message that says "non integer successes in a binomial glm", but you can ignore it and the model will still work.