GLM I - Count Regression

November 5th, 2020

Poisson Regression

Set up

- A Poisson distribution is often used as a count
- When the response is an unbounded count (for example integers: 0,1,2,3…) We can use a count regression model.
- What about when count is bounded? For example if the count always goes from 0-100 or 0-500? What strategy can you use to model that? The beta addition is good to use because if you have a bounded count because all you do is you rescale everything to run between zero and 1. Now the outcome will be bounded between zero and 1.
- As opposed to in the binomial, yes you have the parameter key that is bounded between 0 and 1 bur the outcome is not. The outcome is a number.
- So it's just a different way of setting it up that you could use.

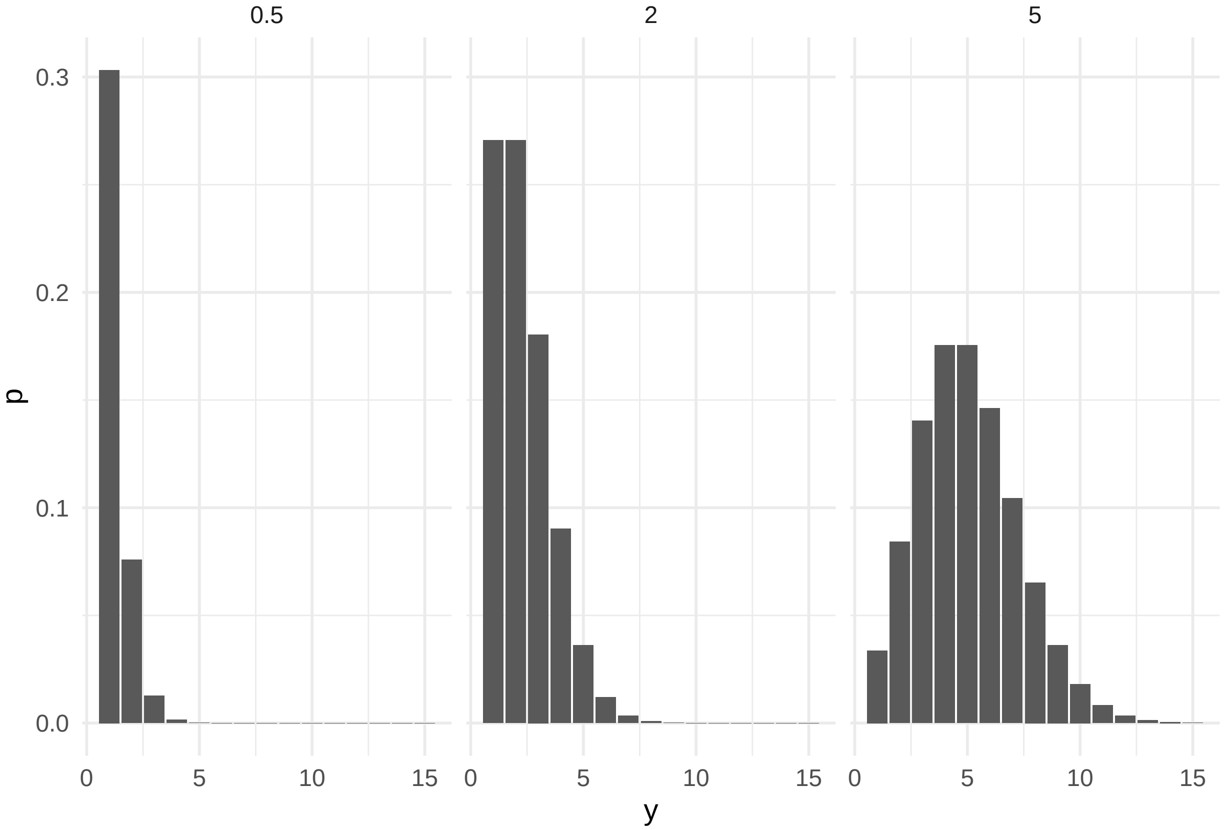
If Y is Poisson with mean $\mu > 0$ then:

$$P(Y = y) = \frac{e^{-\mu}\mu^y}{y!} y = 0, 1, 2, ...EY = varY = \mu$$

- The thing that we see is that there is only one parameter, just a mu. Mu is a single parameter of a Poisson distribution.
- Another thing that we see is that both the expected value of the distribution and the variance of the distribution are both equal to mu. That's what the Poisson distribution is.
- Based on some of the things we're talked about in the semester, can someone think of problems we may have with this regression using Poisson distribution? What's another single variable distribution that we've worked with so far? Binomial, because we just have the p. What were some things that we learned about when dealing with the Binomial or Bernoulli distribution? If the group size isn't large enough the chi-squared approximation for testing for significance of variables is not going to work. That was definitely a problem that was part of the Bernoulli. We shouldn't see that with the Poisson distribution.
- What were some of the things that we talking about towards the end? what happens when the mean and variance are tied together very intimately? If the model assumes the mean and variance are the same, what if the observed data actually has a higher variance? Then we'll see overdispersion, because there is no other parameter to capture that variability. So just like we saw some issues in the binomial case, we're see it again in the Poisson case.

Poisson Distribution

- Here's some examples of the Poisson distribution



- What you see here on the x-axis, is the y. And you have the histogram of the values. Firstly you see that it's bounded to zero. The mu parameters are across the top here, mu of .5, mu of 2, and mu of 5. What you see here is something sort of interesting. As your mu starts to move to higher numbers, you actually end up approximating a normal distribution. It's kind of a good thing, because often times you model something as normal even though they might be physical mini sections of something that can't be less than zero. For example if you model height, we would model it as normal because we expect the estimates to be far enough from zero that we don't have to really deal with the edge cases.
- The Poisson can sort of smooth that out a little bit where you can model things as a count. And if the counts are very high then you effectively get a normal distribution. And if the counts are low, you're at least bounded by zero.

Poisson distribution

- If count is number of possible total then binomial (or beta) is better, but is a good approximation when n is large and p is small.
- Because that approach is the Poisson distribution
- The Poisson distribution naturally arises when probability of event is proportional to length of time and independent of other events.
- Time between events is exponentially distributed (basically the same as above)
- That actually mean the same thing above because exponential distribution is sort of memoryless. Meaning that it doesn't matter where you start on the distribution, the chance of seeing the next thing is the same no matter where you started. Its consistent like that.
- Sum of Poisson variables is Poisson
- So if you have a bunch of Poisson distribution, you sum them up, you get a Poisson distribution that has a parameter that is the sum of all the mu's from the original Poisson distribution.

$$Y_i \sim Pois(\mu_i) i = 1, 2, ... \sum_i Y_i \sim Pois(\sum_i \mu_i)$$

Galapagos Data

- For 30 Galapagos Islands we have a count of number of plants species found on each island
- We're going to be looking at this Galapagos island data set and which has the outcome variable that we're going to be looking at is species, the number of species found on an island.
- And you have these 5 different variables for each island. They dropped out the endemics one. Scruz are stats from Santa Cruz island.

```r
data(gala, package = "faraway")
d <- select(gala, -Endemics)
summary(d)
```

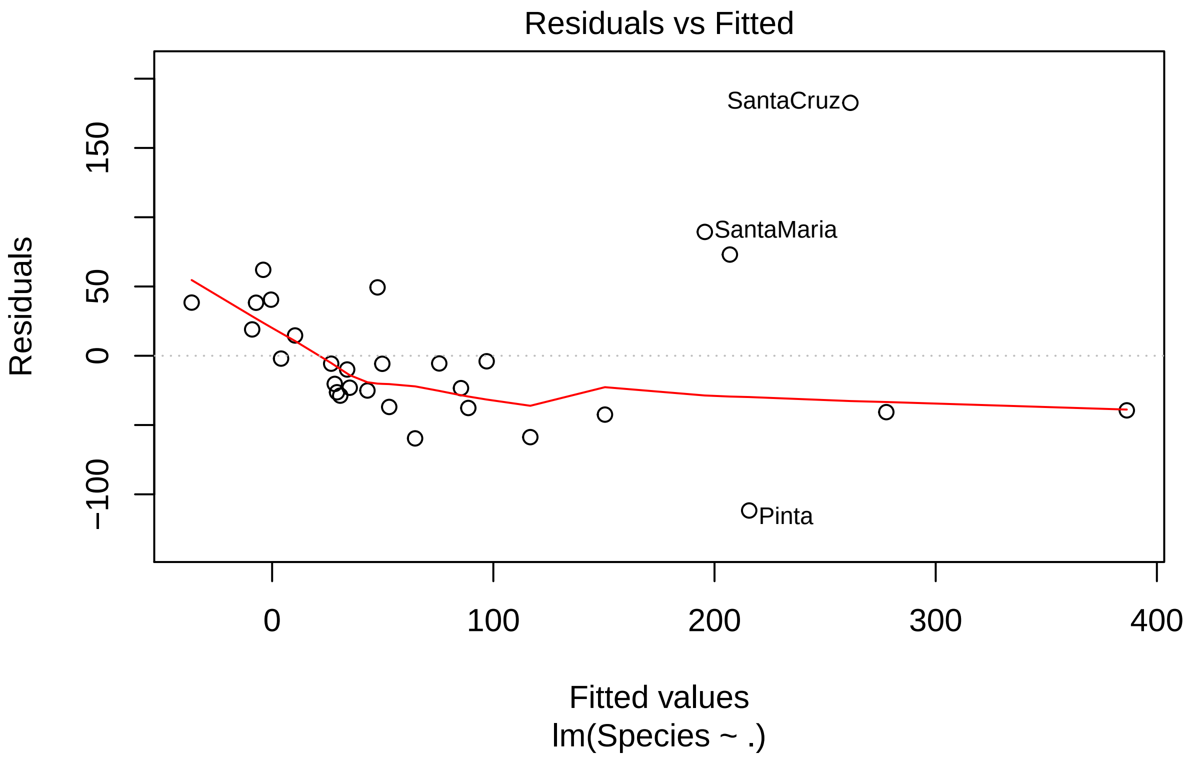
Species	Area	Elevation	Nearest	Scruz	Adjacent
Min. : 2.00	Min. : 0.010	Min. : 25.00	Min. : 0.20	Min. : 0.00	Min. : 0.03
1st Qu.: 13.00	1st Qu.: 0.258	1st Qu.: 97.75	1st Qu.: 0.80	1st Qu.: 11.03	1st Qu.: 0.52
Median : 42.00	Median : 2.590	Median : 192.00	Median : 3.05	Median : 46.65	Median : 2.59
Mean : 85.23	Mean : 261.709	Mean : 368.03	Mean :10.06	Mean : 56.98	Mean : 261.10
3rd Qu.: 96.00	3rd Qu.: 59.237	3rd Qu.: 435.25	3rd Qu.:10.03	3rd Qu.: 81.08	3rd Qu.: 59.24
Max. :444.00	Max. :4669.320	Max. :1707.00	Max. :47.40	Max. :290.20	Max. :4669.32

- It's a surprise that species is a literal count. The min is 2 and the max is 444, the mean of 85. you can see that the mean and median are very far apart and suggests a skewed to the left dataset.
- All the other variables have an min of 0. All of the data is skewed expect scruz.

Galapagos Data | Normal Linear Regression

- We're going to start by modeling the data in a normal linear model. Linear model with species as the outcome variable with everything else to look at it. We're going to plot the residuals against the fitted values.

```r
modl <- lm(Species ~ . , d)
plot(modl, 1)
```

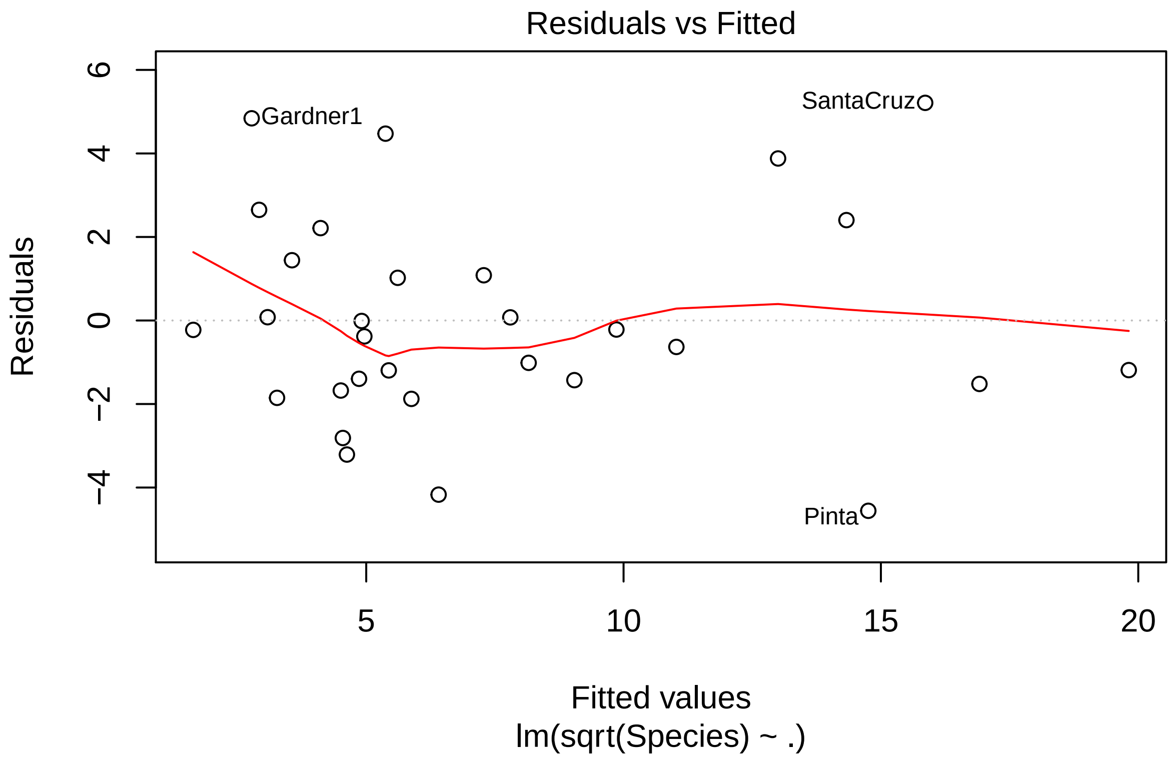


- Can anyone identify the issues with this data? It looks like the residuals gets much larger as it gets father out from zero.
- What helped me is looking at the scale of this on the x and y axis. These are huge. If the actual fitted value is 250 and we were wrong by 150 that's pretty bad. But absolute residuals are father apart but it's hard to say because they are fewer points.
- There's also this weird downward slope.
- So we know there's an issue and we also know that this is truly count data and now that we know that there are regressions for count type data we should go ahead and use it, and see what we can do.

Galapagos Data | Square root transformation

- This is the square root transformation. What this is doing is remember when we talked about how you can log a variable and it helps to work with the outliers. Square rooted in a similar procedure one of the benefits that I know of square rooting is that you can deal with a zero so if you do a log of a zero you can an invalid number but square rooting you don't have to deal with that.
- You can look at this and say this looks a lot better. The residuals are on the square root scale so we have to square these to get them similar but you know that seems a lot better than what we had before.

```
modt <- lm(sqrt(Species) ~ . , d)
plot(modt, 1)
```



- How can we defend why we choose square root it, our reason behind square rooting it? The way you sort of deal with it is i do think it is a way to deal with heteroscedasticity sometimes, because what it does is it makes your outliers affect the model less, in the same way log does. So really what you're doing is that your taking all the data that's sort of carry out there and you're sort of bringing it in and making it a little better distributed. And you can see that happening in the x=5 sections over here. And why you square root instead of fourth or cube root or log of whatever, there's no real difference that my residuals plots look better. Or you know there is an actual reason to do it.
- But we lose something when we square root this, because it makes it a lot harder to interpret the other coefficients of the model. Because now instead of directly modeling species it's modeling the square root of species, which is tougher to deal with and explain.
- When we square root it does it make it harder to find a good fit for the model? Because it's not like the actual values but since we're zooming on it, does it make it harder to find a proper fit? No it shouldn't, numerically it shouldn't matter. Because you have fewer outliers when you square root something it makes it easier to fit stuff for the algorithms to work. Where it becomes harder the interpretation of it. When you square root of something, you're effecting the square root of the count of species. Yes you can you show that something maybe statically significant and effects the square root of the count of species but that's a hard thing to say to someone and make it make sense to them.

Galapagos Data | Square root transformation

```
summary(modt)
```

```
##
## Call:
## lm(formula = sqrt(Species) ~ ., data = d)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.5572 -1.4969 -0.3031  1.3527  5.2110
##
```

```
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  3.3919243  0.8712678   3.893 0.000690 ***
## Area        -0.0019718  0.0010199  -1.933 0.065080 .
## Elevation    0.0164784  0.0024410   6.751 5.55e-07 ***
## Nearest      0.0249326  0.0479495   0.520 0.607844
## Scruz       -0.0134826  0.0097980  -1.376 0.181509
## Adjacent    -0.0033669  0.0008051  -4.182 0.000333 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.774 on 24 degrees of freedom
## Multiple R-squared:  0.7827, Adjusted R-squared:  0.7374
## F-statistic: 17.29 on 5 and 24 DF,  p-value: 2.874e-07
```

- That's sort of what we're showing here, this estimate of 0.0164784. Not really sure what that means. It can means that for every meter you go up in elevation you gain 0.2 of species.
- Not really sure of how to deal with that. So let's look at another strategy and that's creating a Poisson model.

Create Poisson Model

We talked about Poisson being the only parameter is mu.

- If $Y_i \sim Pois(\mu_i)$ what constraint is on μ? For example, can mu be a negative number? Remember what mu represents (in the set up part), mu is = to the expected value and the variance. So if we're talking about counts, can we have a neg value for counts? No. So what we need is mu to be positive.

- We talked about logistic link function before. Do we need to use the logit link function here with mu? Why or why not? No, we don't need to use the logit function because we don't need to be bounded between zero and one. But we do need to be positive. Part of the logit function is the log, that's one of the things we used to make the values positive. So what we're going to do is use the log link function in the same way.

- $\eta_i = x_i^T\beta$

- First our linear predictor is going to be the same, as we have been seeing this entire time. The only thing we're going to change is the distribution but we're also going to change the link function.

  - $log\mu_i = \eta_i = x_i^T\beta$
  - So now we're going to say log of mu is = to eta is = the values here.

Here is the loglikelihood equation:

$$l(\beta) = \sum_{i=1}^{n}(y_i x_i^T\beta - exp(x_i^T\beta) - log(y_i!))$$

- Now that we know the link function, and we see a beta of zero, how are we going to actual interpret it? We want to invert it so we want to exponentiate both sides. It's going to be similar how we're been looking at coefficients of binominal or Bernoulli regression. When we exponentiate the betas then they are multiplicative influence on the mean. So a beta of 0, you exponentiating 0, e^0 =1, that is no effect. That means you are multiplying the current estimate of mu times 1 and you're getting mu out again. The idea is that we're going to exponentiate the betas to make them interpreting just like we were doing with the binomial. - The only thing now is that it's not a hazard ratio it's just a rate ratio. It's really just multiplication on the current value of mu. - The T is a transpose, all it's doing is it's making it so you can multiply, it's sort of the matrix notation of part of it. It's making it so you can multiply the two things together. If you have multiply x's or rows and multiply columns for x, therefore you have multiple betas.

Fit Poisson Model

```
modp <- glm(Species ~ ., family=poisson, d)
summary(modp)
```

```
##
## Call:
## glm(formula = Species ~ ., family = poisson, data = d)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -8.2752  -4.4966  -0.9443   1.9168  10.1849
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.155e+00  5.175e-02  60.963  < 2e-16 ***
## Area        -5.799e-04  2.627e-05 -22.074  < 2e-16 ***
## Elevation    3.541e-03  8.741e-05  40.507  < 2e-16 ***
## Nearest      8.826e-03  1.821e-03   4.846 1.26e-06 ***
## Scruz       -5.709e-03  6.256e-04  -9.126  < 2e-16 ***
## Adjacent    -6.630e-04  2.933e-05 -22.608  < 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: 3510.73  on 29  degrees of freedom
## Residual deviance:  716.85  on 24  degrees of freedom
## AIC: 889.68
##
## Number of Fisher Scoring iterations: 5
```

We use the glm call, but our family is now Poisson. Now we take a look at the summary function. These values we can't really interpret. They are very close to one. But let's take this value.

In general for the original summary function of the dataset, we can see that in elevation we might scale this to something since these values are pretty high. Like maybe if this is meters, maybe we scale it so its meters' divided by 100. So the rise of every 100 meters, what's our change?

Basically in this case, we do e^3.541 and it was 1.003. So for every elevation, every meter you go up, you multiple you're expected number of species by 1.003. That is how you can kind of interpret these values, while holding everything else at constant.

All the tools we have been talking about so far are still applicable. You can see here the residual deviance, and it's actually still pretty high still. We have 716 on 24 degrees of freedom and we generally want them to be equal to each other. So there's still something clearly not working super well in this particular regression equation.

Deviance

- You can use it the same way as before (goodness of fit and compare nested models)
- Deviance (or G-statistic)

$$D = 2\sum_{i=1}^{n}(y_i log(y_i/\hat{\mu}_i) - (y_i - \hat{\mu}_i))$$

- Pearson's X^2, Pearson chi square test

$$X^2 = \sum_{i=1}^{n} \frac{(O-E)^2}{E} = \frac{(y_i - \hat{\mu}_i)^2}{\hat{\mu}_i}$$

- All those same tools, you can test nested models using the deviance, you can compare to the null model and get a feel for it.
- The beauty of these regression equations is that a lot of the tools we used for the one we can use the another one.

Dispersed Poisson Model

Overdispersion in Poisson

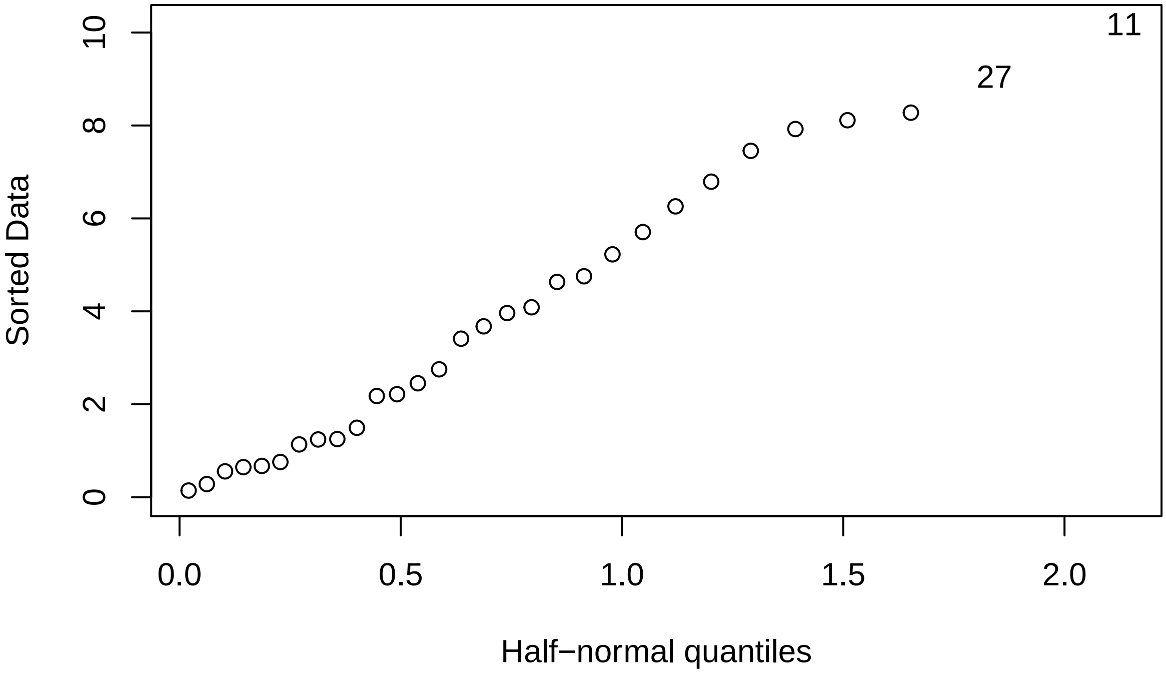
There can be similar issues in the Poisson model, as we saw in the binomial model with overdispersion. Because we only have that one parameter we don't have a way of varying the variance throughout the model.

- Same issues as binomial - variance is tied to mean

```
broom::glance(modp)
```

null.deviance	df.null	logLik	AIC	BIC	deviance	df.residual	nobs
3510.729	29	-438.8384	889.6767	898.0839	716.8458	24	30

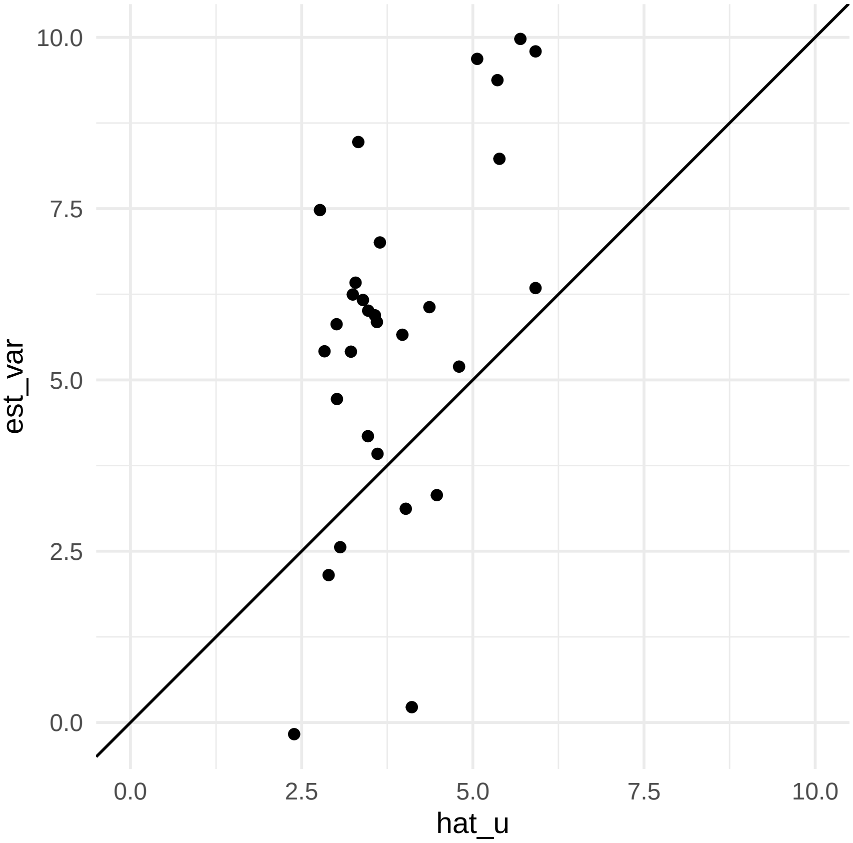
```
halfnorm(residuals(modp))
```



- The book sort of goes through the other ways we talked about on make sure there aren't any outliers. This is showing what we saw at the bottom of the summary before that the current deviance and the residuals of the deviance is not fitting the data very well.
- This half normal plot of the residuals and they aren't really any outstanding points that suggest strong outliers.

Variance vs Mean

- We're left with the same idea we saw before, that the variance might be too high.
- Use $(y - \hat{\mu})^2$ as an estimate of variance
- This is a way you can sort of estimate the variance sing the model you have looking at the observed - expected squared just as a rough estimate of the variance. vs the expected value, the x-axis hat_mu, because the expected values is the same as the variance we expect for that data.



- And so when we look at the model here, a lot of the estimated variance is consistently higher than our expected variance. This is not a rule but it's sort of pointing us to the issues of overdispersion.

Overdispersion

We can deal with it the same exact way we did we dealt with it with the binomial. And what we do is introduce a dispersion parameter through the variance, the same way we did with the binomial and we estimate the exact same way using the person residual divided by n-p. And we can use that in the model.

- What can we say about the point estimates for parameters in the model?
- What can we say about the standard error around those parameters?
- We can add dispersion parameter such that $varY = \phi\mu$
- Estimate with:

$$\hat{\phi} = \frac{X^2}{n-p} = \frac{\frac{(y_i - \hat{\mu}_i)^2}{\hat{\mu}_i}}{n-p}$$

Overdispersion Application

- The same sort of ideas apply as before.

```r
dp <- sum(residuals(modp,type="pearson")^2)/modp$df.res
summary(modp, dispersion = dp)
```

```
##
## Call:
## glm(formula = Species ~ ., family = poisson, data = d)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -8.2752  -4.4966  -0.9443   1.9168  10.1849
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.1548079  0.2915897  10.819  < 2e-16 ***
## Area        -0.0005799  0.0001480  -3.918 8.95e-05 ***
## Elevation    0.0035406  0.0004925   7.189 6.53e-13 ***
## Nearest      0.0088256  0.0102621   0.860    0.390
## Scruz       -0.0057094  0.0035251  -1.620    0.105
## Adjacent    -0.0006630  0.0001653  -4.012 6.01e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 31.74914)
##
##     Null deviance: 3510.73  on 29  degrees of freedom
## Residual deviance:  716.85  on 24  degrees of freedom
## AIC: 889.68
##
## Number of Fisher Scoring iterations: 5
```

- The actual estimates don't change, but the standard error will change when we introduce a dispersion parameter.
- And we can do it the same way as before, there's actually a quasi-Poisson family or we can do it by hand by calculating the dispersion parameter by hand and adding it in the regression.

Overdispersion Application

- You can use `quasipoisson` to do it all in one go
- Use F test to compare nested models

Rate Models

- This is again getting at this idea that there may be something, not necessarily a bound to a regression, but there is some sort of size type variable that the counts depend on. And it's not like a count where a person is counted that's sort of it. It can happen to a person multiple times.
- The example the book gives is that is your modeling house burgherly, it's possible that a house gets burgled more than once.

Set up

- The number of variables may depend on a 'size' variable - but a binomial rate may not make sense

- Any ideas for examples?
- Some scaler (e.g. population or time)
- When may a binomial not make sense?
  - rate > 1
  - multiple events per row (items your looking at)
- Can you think of any example where this type of model might come into play? A rate model, where you have a count of something but it depends on some other scale of value. Either it's the number of people or maybe the amount of time where if you need to take into some scaling factor into account.
- For example what if you're modeling the number of words someone types per min. What you have is the number words they've typed and the amount of time they've taken to type them. If that's the case, someone can be a very slow typer and type have a word per min or they can be faster and typer 5 words per min.
- If you have data on the professor, and you only have two mins of my typing and you have data on yourself and have 5 mins of typing. You can't compare the actual type words, you have to think about it in the compact of the scalar.
- The outcome variable with is a count, if very closely related to some sort of scalar. Not in a way that we care about its effect on the outcome. But we just know that there's this scalar value that we have to account for somehow when we do the regression.
- What if the number of birthdays that are celebrated that we celebrated in USA vs the number of birthdays celebrated in Sweden. That is very tied to the population.
- This is way of modeling a count, when there is some sort of scalar in the background.

Gamma Radiation on Chromosomal Abnormalities

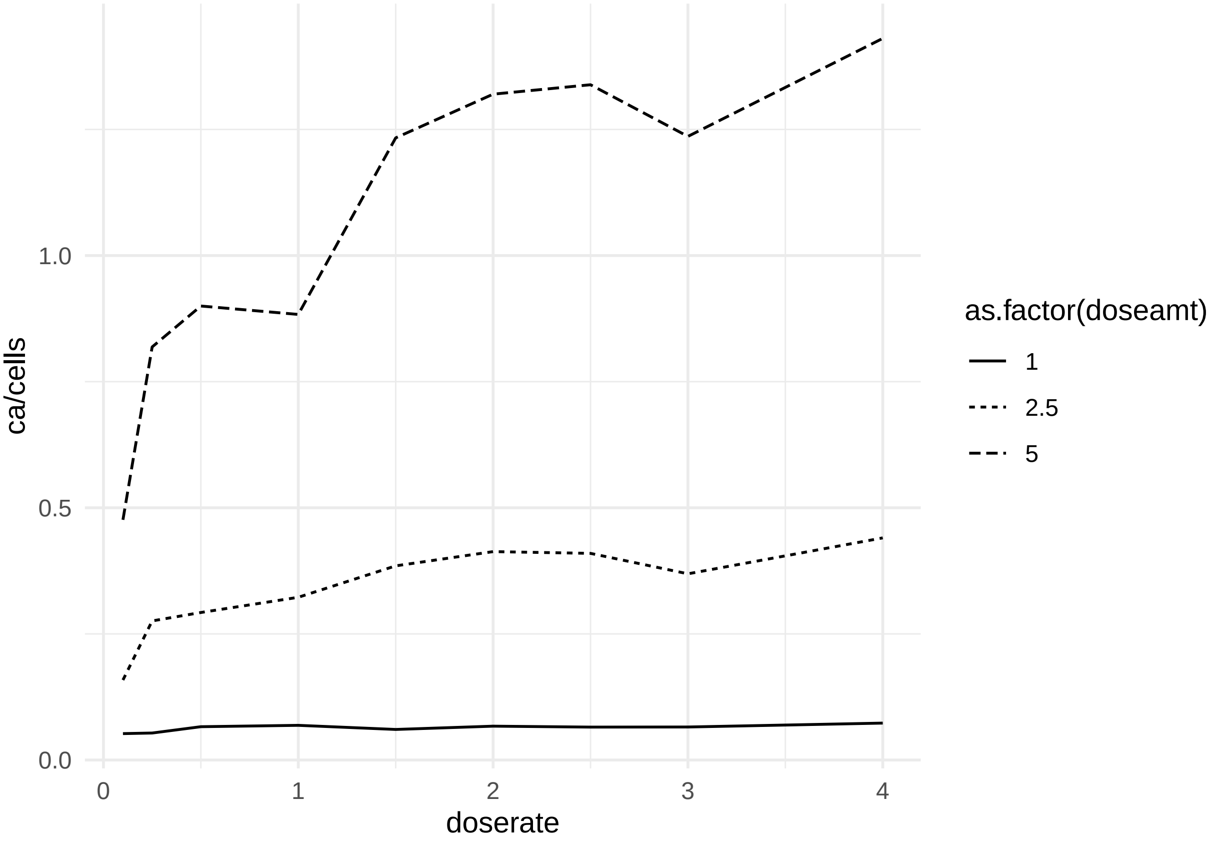
- Outcome is the number of chromosomal abnormalities/cell
- So the idea here is that we have two variables. The dose rate is how quickly you give this gamma ration and how much was given.
- In each of these values we have the number of chromosomal abnormalities per cell. This is sort of the rate of chromosomal abnormalities in a cell, but each time you do this trail you have a different number of cells. So the range goes from something very low, much less that 1 abnormal per cell to greater than 1 abnormality per cell.
- This is why you can't use a binomial or beta regression because this is an unbounded number, it can be more than 1 abnormal in a cell.

doseamt/doserate	0.1	0.25	0.5	1	1.5	2	2.5	3	4
1	0.05	0.05	0.07	0.07	0.06	0.07	0.07	0.07	0.07
2.5	0.16	0.28	0.29	0.32	0.38	0.41	0.41	0.37	0.44
5	0.48	0.82	0.90	0.88	1.23	1.32	1.34	1.24	1.43

Gamma Radiation on Chromosomal Abnormalities | Cont.

- This is an interaction term here. There's no model here. This is just looking at the dose amount and the dose rate vs the rate of chromosomal abnormalities.
- This is an interaction plot, where you're showing two sets of information on it. What it does do it is point you in the direction there may be an interaction happening here.
- We might see this playing out in this plot is that as the dose rate increases at the different dose amounts they are changing at different rates.
- What we're seeing here is that the dose rate, as it goes up. basically the slope of these three lines don't have the same slope. That is effectively what we're seeing here. If these 3 had exactly the same slope then we wouldn't bother with looking for interactions. But we definitely see relationship here
- If we're talking about radiation, we can say that the number of abnormalities you see probably go up with the rate of dosing. I guess how quickly or how long you dose these cells with radiation. Similarly, the chromosomal abnormalities go up given a higher dose level.

- What we're then saying as well is in fact the number of abnormalities goes up faster the higher the dose amount. Which makes sense, imagine you have a dose of 1, what if a dose of 1 is effectively what you get from the sun. If it's a low enough number, it doesn't matter how much of it you get. Like if you sit outside all day, it might not be that big of a deal. But when you start to get to these higher thresholds, then it becomes worse and worse for you to be in this environment.

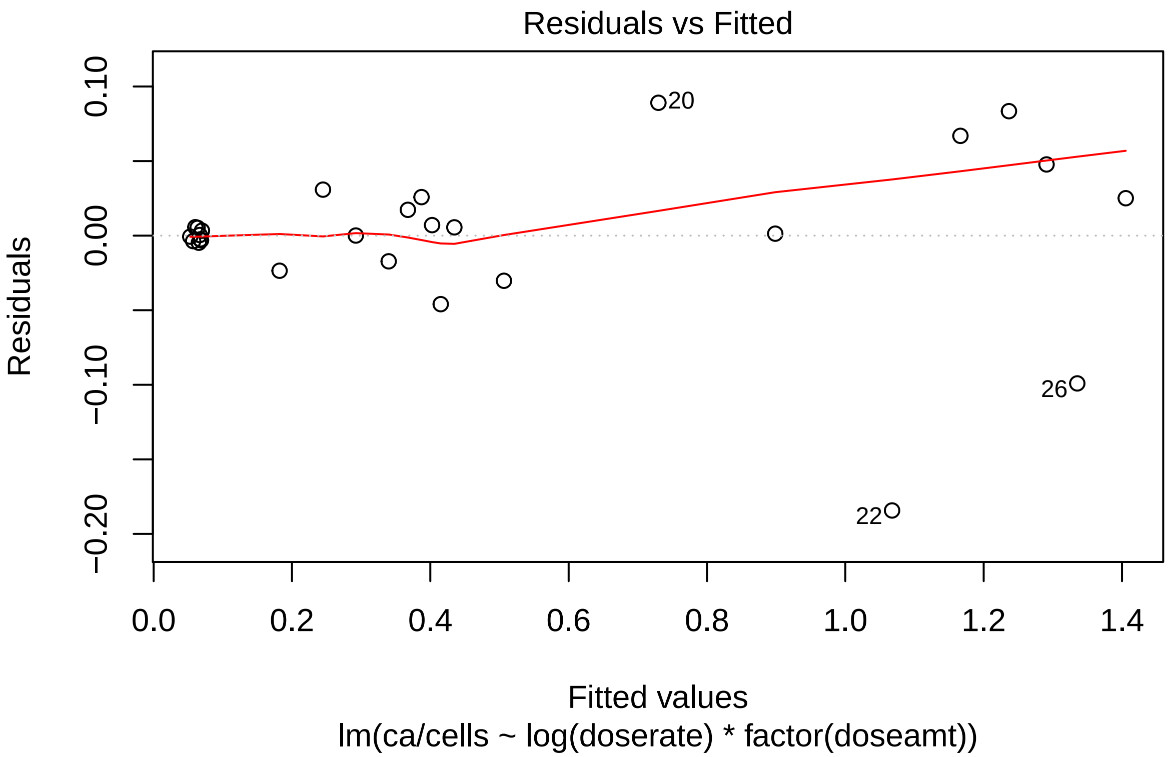


- Notice that chromosomal abnormalities is divided by the number of cells.

Model

- First, what you can do is model it as a linear model but I think again you see some heteroscedasticity here where the variance is higher in these terms over here on the left side. where the variance is a little higher up here.

```
## [1] 0.9844421
```



Use Poisson

- Two things first, we're going to be modeling the log(doserate). The justification there is that there is the theory that it's not just a linear effect of dose rate on the data. It's not super obvious in the gamma radiation on ca interaction plot, other than the fact that you kind of see these curves level off a little bit on these higher ones. But that could just be something that you learn because you know about the physical phenomena happening in the background.
- What we're going to do it that we're going to add this funky multiplication term. Before we were just adding different variables. here we're going to add a multiplication term. and in this case we actually make doseef a factor, named dosef
- This is our way of saying to include an interaction term, in fact in includes both an interaction term and a regular plus term as well.

```
dicentric$dosef <- factor(dicentric$doseamt)
# interactoint term with the *
pmod <- glm(ca ~ offset(log(cells)) +log(doserate)*dosef, family=poisson,data = dicentric)
summary(pmod)
```

```
##
## Call:
## glm(formula = ca ~ offset(log(cells)) + log(doserate) * dosef,
##     family = poisson, data = dicentric)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.49101  -0.62473  -0.05078   0.76786   1.59115
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -2.74671    0.03426 -80.165  < 2e-16 ***
## log(doserate)           0.07178    0.03518   2.041 0.041299 *
## dosef2.5                1.62542    0.04946  32.863  < 2e-16 ***
## dosef5                  2.76109    0.04349  63.491  < 2e-16 ***
## log(doserate):dosef2.5  0.16122    0.04830   3.338 0.000844 ***
## log(doserate):dosef5    0.19350    0.04243   4.561  5.1e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 4753.00  on 26  degrees of freedom
## Residual deviance:   21.75  on 21  degrees of freedom
## AIC: 209.16
##
## Number of Fisher Scoring iterations: 4
```

- We have our intercept, even though we don't explicitly put it in the formula we know that there's always one. Notice that there is no term for offset. There is no value here where we have estimates for the offset of the log of cells. Because it's an offset, we don't estimate a parameter.
- We know that dosef has 3 values, 1, 2.5 and 5. And we see those here. The base level is dose factor 1. In relationship to dose factor 1, we have dose factor 2.5 and dose factor 5.
- Likewise, doserate is a continuous variable. We take the log of it, that's fine, nothing wrong with that. And here is our estimate of log(doserate)
- Now is where it gets a little funkier. We see this down here which is the log(doserate):dosef2.5 and log(doserate):dosef5. Both are interaction terms, because the dosef has 3 separate values in total and we see the two here.
- The way we can interpret this is, and we can sort of add our thinking because we looked at the graph and we know what to expect. This is saying that effectively you add .16 to the slope of the log(doserate) when the dose is 2.5.
- So if we didn't have these two interaction terms here, every time the log(doserate) goes up by one value, we expect the output to change, on the log scale because it's a Poisson distribution, by 1.7. It doesn't matter what's going on in the other variables. If we just have the first 4 lines, this is always the relationship between the dose rate and the outcome variable, regardless of what the effective dose is.
- Now when you add an interaction term, that no longer holds. Because you're changing dose rate is effectively higher when your dose amount is 2.5, and the same thing when you're dose rate is 5, it's higher than .07.
- This is one of those things, interaction terms are very unwieldy and very hard to deal with. While you can use them to build a model without any problem, interpreting that model becomes a lot harder.
- This is a clean example where we have one continuous variable and one categorical variable. But if we had kept dose as a factor, as we continuous variable, it would be very hard to interpret.
- And so one of the things you want to do is you really want to draw this stuff out. Because what we want to show is on a graph like this, what we're doing makes sense with the data.
- So let's look at the full equation. If your regular dose amount is 1, you take your log(doserate) estimate is how much you're outcome changes based off of log of doserate. If you dose amount is 1, you just look at this log(doserate) coefficients.
- If you then are at a dose amount of 2.5, you can't just look at that value 0.07 anymore. First of all you have to add the constant 1.62542 that doesn't depend on dose rate at all. If your doserate goes up from 1 to 2 in dose of 1, this goes up by this constant amount 0.07178. But if your dose is 2.5, you then also

need to look at this interaction term here. So the amount that you raise for each log(doserate) = .07 + .16, instead of just 0.07.

- What that translates to is basically it translates to these different slopes. So when you're at dose 1, your log of doserate is barely significant, barely changing your outcome variable based off of this estimate. But when you add in that you are not at a 2.5 dose amount, you see that there is a bit of a steeper slope here. Not only is your base a little higher, which comes from the constant estimate 1.62542, but also your slope here is higher. And that comes from the fact that you have this interaction term.

How does offset work?

- What you can do is sort of think about it in a different way. If we instead model the chromosomal abnormalities/cells and put a log link effectively on that. What you can do is actually rearrange the variables by moving the log of the number of cells over to the right side of the equation. Because log of ca/cell is the same as log (ca) - the log (cells).

$$log(ca/cells) = X\beta log(ca) = log(cells) + X\beta$$

- We don't want to estimate another parameter here. It's not an independent variable. It is just this value here what we want to take into account when we're putting the model together.

Interaction Term

- The way we can do that is there is special function called offset that you can put directly into the formula that you put together. What we can do is include that term here in the equation, and then model chromosomal abnormalities directly. Because there's a log link already there we don't need to put a log around the chromosomal abnormalities. What we're doing is that we can basically model it like that. It's pretty cool.
- That's what this offset does, it allows you to effectively offset your equation without adding another variable to it. It's also sort of the same thing of adding variable but that has a fixed beta equal to 1.

```
# Here I just have the addition term. There's no interaction here
pmod_no_inter <- glm(ca ~ offset(log(cells)) +log(doserate) + dosef, family=poisson,data = dicentric)
dicentric_fit <- dicentric
# Added fitted value for both the interaction model and the non-interaction model and we're going to lo
dicentric_fit$fitted <- fitted.values(pmod)
dicentric_fit$fitted_no_inter <- fitted.values(pmod_no_inter)
```

- When you have your normal regression equation and you write something like dosef (the dosage amount) you're going to get a beta, and estimate for this value. When you wrap the call in an offset, there will actually won't be an estimate for this value. It's just basically modifying the equation as a whole with no variable, with nothing else sort of estimating it. You're just offsetting this equation here when you're modeling against chromosomal abnormalities.

- That's why we have the log of cells out here and it's not part of the x terms that we would normally model.

- The x terms here include the log oof dose rate and the effect dose. That's sort of wrapped up in this X because its supposed to be a matrix. And this sort of lives outside of it. Because we don't want to estimate any beta's for it we just want to have a way of controlling for the number of cell we're looking at.

- Let's say what you do is you blast this number of cells with this radiation and you look at them under a microscope. And maybe you stain them and that's how you count the chromosomal abnormalities. So what we do is we maybe look at a microscope slide, every time we look at a new trial, we'd prob see a different number of cells. Like maybe we see 20 in 1, 50 in the other. Let's say we look at the version with 20 in it and we see 5 chromosomal abnormalities in there or we look at the 50 cells and we also see

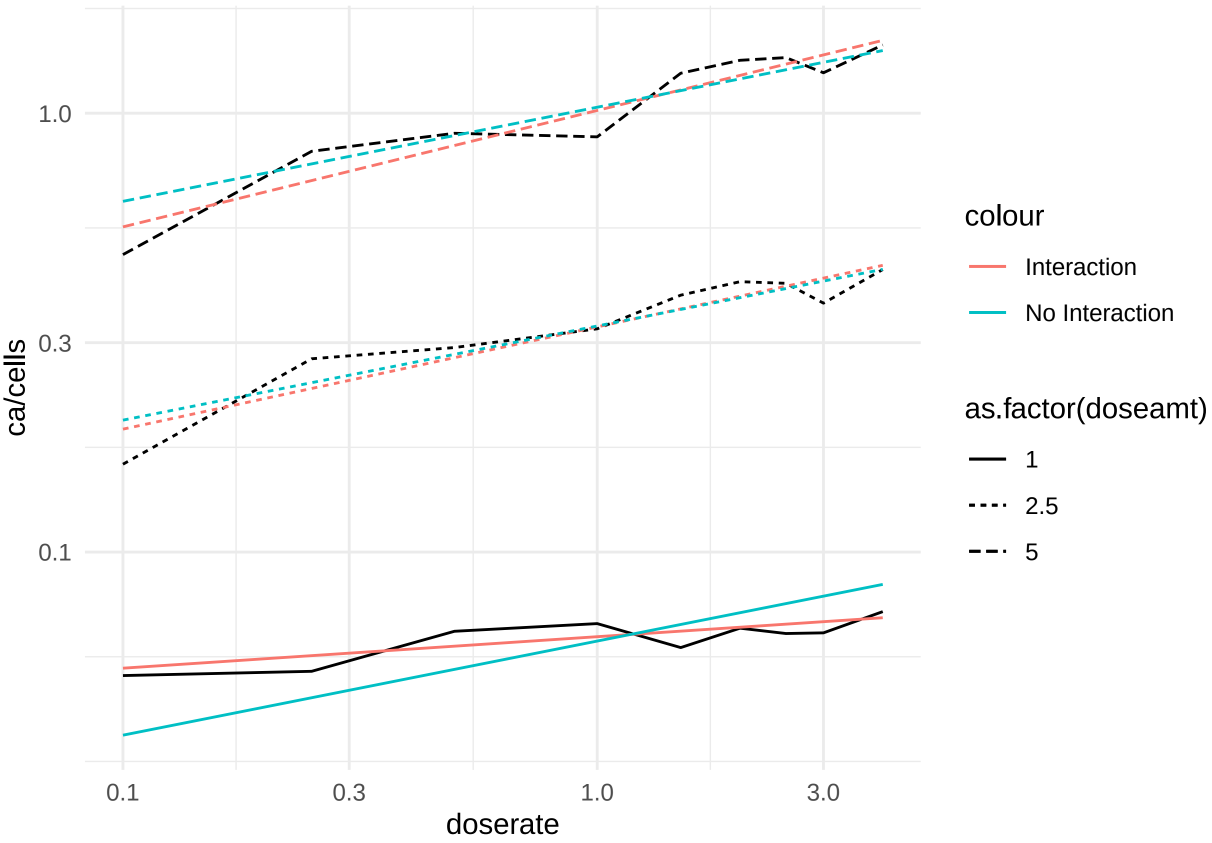
5 chromosomal abnormalities. So you're outcome variable is chromosomal abnormalities, it is the same in both cases, 5 in both. But you know that's not full story, because in one you see 5/20 and the other 5/50.

- So when we're modeling this phenomena, it's important to see how many cells are actually in view.
- So in this example, why wouldn't we just make it a proportion before you modeled it? What are the benefits of modeling the count instead of the proportion? So it's not necessarily that there's always going to be a benefit. Part of the benefit is that it sort of makes sense. The proportion can be greater than 1, that's sort of want their saying here where one cell can have more than 1 chromosomal abnormality. So sometimes you can have more than 1, so you're sort of out of luck on the binomial way of looking at things.
- Yes you can model it in a normal regression, but it's not super great because you have all of these that are very close to zero. So when you are modeling this in a normal regression, you might have predictions that are less than zero. But in reality those predictions are impossible, you can have less than zero chromosomal abnormality. This is way of dealing with that.
- But also more importantly it is a more accurate representation of what is happening with the data. You are sort of counting these cells and you are also counting these cells out of a total number. So you just want to sort of use as much of that info as you can.
- The other thing you could probability do is that you can take the log of this, instead of modeling ca/cells directly, you can model the log(ca/cells). Then again anytime you do a transformation of the response variable it becomes harder to interpret. So then you're effect is a multiplicative effect on the rate of cells, which you could probably justify and talk about it.
- But it might make more sense to use this strategy to model it.

Interaction Term | Plot

```r
p <- ggplot(dicentric, aes(x = doserate, y = ca/cells, linetype = as.factor(doseamt))) +
  geom_line() +
  geom_line(data = dicentric_fit, aes(y = fitted/cells, color = "Interaction")) +
  geom_line(data = dicentric_fit, aes(y = fitted_no_inter/cells, color = "No Interaction")) +
  scale_x_log10() +
  scale_y_log10()
# Notice that I scale both the x and y to log in ggplot. That'll help us just see this so we don’t have
```

Interaction Term | Plot



- The black lines are the same as before. we're just showing them on the log scale, both on the x-axis and y-axis. And so you'll see immediately that they are very straight now. It's easier for us to sort of see patterns because they are straight and not curved. It's a little harder to see slope changes and things like that when everything is curved
- The red is the outcome from the interaction model where the interaction changes, and the blue is where there is no interaction. First of all you can see that up here that they look very similar, the red and blue. But where they are different is down here. So the idea is the slope of each of the blue lines is actually the same, it's not super obviously because they are far apart but you can trust me on it. The scale on the all of the blue lines are equivalent.
- But now let's look at this relationship between the blue line to the black line. It doesn't look like it's fitting the data very well on the bottom part, it's sort of just bisecting it. It feels like it needs to flatten out a little bit.
- Now let's compare it to the interaction model. In the interaction model, the slope of this line has the ability to change. The slope of all of the red lines is not actually the same, they're slightly different. And you can see that sort of play out because in this lower black line, the red line actually flows it very well. However, the blue line is not able to do so because it also has to fit these other two sets of data as well.
- And so that's where you see the interaction term play out, it is allows you to modify the slopes of each of these lines. This is easier to see when you plot it out like this, and easier to interpret then when you are looking at the table of values. But this only works because we only have a couple of dose amount, and one continuous variable. The second you add more different interactions, it becomes a lot harder to sort of deal with. We can conclude that the interaction model is better.
- Looking at this data I would say that the interaction term is better. And I would also lean into it a little bit more because I think there is a physical reason for the interaction to be there. Sort of how I was taking about dose amount vs doserate. Dose amount is very low.

- The downside of adding an interaction is the interpretation. And the fact that we added two extra terms to the model. So what we're doing is that is it possible that we might overfit the data. This would be a good way of doing a nested chi squared test and see if the model improves or not.

Negative Binomial

Setup

- The way that you could sort of imagine having a negative binomial data set is here:
- Given a series of independent trials, each with probability success p, let Z be the number of trials until the kth success.
- Like maybe you were talking about basketball games, how many games does the team have to play before there is a 50% chance that they'll win 5 games.
- And really it's a little bit hard to interpret these sections here in the formula.

$$P(Z = z) = \binom{z-1}{k-1} p^k (1-p)^{z-k} z = k, k+1, ...$$

- Reparametrize so that $Y = Z - k$ and $p = (1 + \alpha)^{-1}$
- $EY = \mu = k\alpha$ and $varY = k\alpha + k\alpha^2 = \mu + \mu^2/k$

Solder Data

- This is looking at soldering data, which is looking at a dataset that models the number of times a soldering joint was skipped on an integrated circuit and they have a bunch of different parameters

```
# t
data(solder, package="faraway")
modp <- glm(skips ~ . , family=poisson, data=solder)
c(deviance(modp), df.residual(modp))
```

```
## [1] 1829.002  882.000
```

- So basically, we're modeling the number of skips. And so if we do a regular simple Poisson, we see a big difference between the deviance and the residual deviance.
- And so what we can do is, there's two ways of doing it. We have 2 parameters, we have the k parameter and the p parameter. You can do one of two things, you can either specify a k to use or you can actually fit it like any other value.

Fit model with fixed K

- Here if you use just the regular glm, and here we're using the negative binomial family which is built in, you can specify a k to use, which we used as 1.

```
library(MASS)
```

```
##
## Attaching package: 'MASS'
```

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

```
modn <- glm(skips ~ .,negative.binomial(1),solder)
summary(modn)
```

```
##
```

```
## Call:
## glm(formula = skips ~ ., family = negative.binomial(1), data = solder)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9628  -0.8021  -0.2373   0.3204   2.1097
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -1.69933    0.16334 -10.404  < 2e-16 ***
## OpeningM     0.50854    0.08797   5.781 1.03e-08 ***
## OpeningS     1.99966    0.08069  24.783  < 2e-16 ***
## SolderThin   1.04894    0.06364  16.483  < 2e-16 ***
## MaskA3       0.65710    0.10463   6.280 5.29e-10 ***
## MaskA6       2.52649    0.12123  20.841  < 2e-16 ***
## MaskB3       1.27261    0.10780  11.806  < 2e-16 ***
## MaskB6       2.08026    0.10482  19.845  < 2e-16 ***
## PadTypeD6   -0.46118    0.13788  -3.345 0.000859 ***
## PadTypeD7    0.01608    0.13350   0.120 0.904185
## PadTypeL4    0.46883    0.13046   3.594 0.000344 ***
## PadTypeL6   -0.47115    0.13799  -3.414 0.000669 ***
## PadTypeL7   -0.29494    0.13620  -2.165 0.030618 *
## PadTypeL8   -0.08493    0.13432  -0.632 0.527372
## PadTypeL9   -0.52125    0.13854  -3.763 0.000179 ***
## PadTypeW4   -0.14250    0.13481  -1.057 0.290781
## PadTypeW9   -1.48361    0.15317  -9.686  < 2e-16 ***
## Panel        0.16932    0.03811   4.443 1.00e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(1) family taken to be 0.5560534)
##
##     Null deviance: 1743.01  on 899  degrees of freedom
## Residual deviance:  558.67  on 882  degrees of freedom
## AIC: 3883.7
##
## Number of Fisher Scoring iterations: 9
```

- The estimates of the model are interpreted the same way as for a regular Poisson, you exponentiate it and that's the effect it has on the number of skips.

Fit model with parameter K

- If you don't have a way of fitting the k, you can fit it using the function called glm.nb, which is in the MASS package. All it does it is it allows you to actually using maximum likelihood to fit that k, they call it theta here which would be k-4.
- Unless you have a very good reason for having a fixed k, I wouldn't set it. Nevertheless you can. I wouldn't even know how to go about defending a k value, I would just use maximum likelihood estimation to find the best value of k to fit it. It tested a bunch of different k values and landed on 4.39 being the best one.
- It's a way of dealing with overdispersion. So the model where we do fit k, you know that model has residual deviance of 1008 on 882 degrees of freedom while the regular Poisson model has 1829 on 882 df, almost double. It fits the data a lot better and helps deal with overdispersion.

- That's actually the most common way that I've seen people deal with overdispersion in a Poisson or count scenario, you just use a negative binomial.

```
modn <- glm.nb(skips ~ .,solder)
summary(modn)
```

```
##
## Call:
## glm.nb(formula = skips ~ ., data = solder, init.theta = 4.397157245,
##     link = log)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7376  -1.0068  -0.3834   0.4460   2.7829
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.42245    0.14274  -9.965  < 2e-16 ***
## OpeningM     0.50294    0.07976   6.306 2.87e-10 ***
## OpeningS     1.91317    0.07152  26.750  < 2e-16 ***
## SolderThin   0.93932    0.05362  17.517  < 2e-16 ***
## MaskA3       0.58981    0.09651   6.112 9.87e-10 ***
## MaskA6       2.26734    0.10182  22.269  < 2e-16 ***
## MaskB3       1.21101    0.09637  12.566  < 2e-16 ***
## MaskB6       1.99037    0.09223  21.580  < 2e-16 ***
## PadTypeD6   -0.46592    0.11238  -4.146 3.38e-05 ***
## PadTypeD7   -0.03315    0.10673  -0.311 0.756114
## PadTypeL4    0.38268    0.10265   3.728 0.000193 ***
## PadTypeL6   -0.57844    0.11413  -5.068 4.01e-07 ***
## PadTypeL7   -0.36656    0.11094  -3.304 0.000953 ***
## PadTypeL8   -0.15890    0.10821  -1.468 0.141986
## PadTypeL9   -0.56600    0.11393  -4.968 6.77e-07 ***
## PadTypeW4   -0.20044    0.10873  -1.844 0.065255 .
## PadTypeW9   -1.56460    0.13621 -11.486  < 2e-16 ***
## Panel        0.16369    0.03139   5.214 1.85e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Negative Binomial(4.3972) family taken to be 1)
##
##     Null deviance: 4043.3  on 899  degrees of freedom
## Residual deviance: 1008.3  on 882  degrees of freedom
## AIC: 3683.3
##
## Number of Fisher Scoring iterations: 1
##
##
##               Theta:  4.397
##           Std. Err.:  0.495
##
##  2 x log-likelihood:  -3645.309
```

Zero Inflated Count Models

Set up

- We often see situations where there are more 0's than would be expected by either of these models.
- We consider a sample of 915 biochemistry graduate students as analyzed by Long (1990). The response is the number of articles produced during the last three years of the PhD. We are interested in how this is related to the gender, marital status, number of children, prestige of the department and productivity of the advisor of the student.

```
library(pscl)
```

```
## Classes and Methods for R developed in the
## Political Science Computational Laboratory
## Department of Political Science
## Stanford University
## Simon Jackman
## hurdle and zeroinfl functions by Achim Zeileis
```

```
modp <- glm(art ~ ., data=bioChemists, family=poisson)
summary(modp)
```

```
##
## Call:
## glm(formula = art ~ ., family = poisson, data = bioChemists)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.5672  -1.5398  -0.3660   0.5722   5.4467
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.304617   0.102981   2.958   0.0031 **
## femWomen    -0.224594   0.054613  -4.112 3.92e-05 ***
## marMarried   0.155243   0.061374   2.529   0.0114 *
## kid5        -0.184883   0.040127  -4.607 4.08e-06 ***
## phd          0.012823   0.026397   0.486   0.6271
## ment         0.025543   0.002006  12.733  < 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: 1817.4  on 914  degrees of freedom
## Residual deviance: 1634.4  on 909  degrees of freedom
## AIC: 3314.1
##
## Number of Fisher Scoring iterations: 5
```