STAT 528 - Advanced Regression Analysis II

Count response regression (part II)

Daniel J. Eck
Department of Statistics
University of Illinois

Last time

- Poisson regression
- Residual diagnostics
- Data analysis

Learning Objectives Today

- Overdispersion
- gala data analysis
- Negative Binomial regression
- Zero Inflated Count Models

When the mechanism is not known, we can introduce a dispersion parameter ϕ such that $\mathrm{Var}(Y) = \phi\mathrm{E}(Y) = \phi\mu$. This decouples the variance from the mean.

The case $\phi = 1$ is the regular Poisson regression case, while $\phi > 1$ is overdispersion and $\phi < 1$ is underdispersion.

A common explanation for large deviance (or poor fit) is the presence of a few outliers.

When large number of points are identified as outliers, they become unexceptional, and it may be the case that the error distribution is misspecified.

In the presence of overdispersion, the exponential family takes on a different functional form

$$f(y|\theta,\phi) = \exp\left(\frac{\langle y,\theta\rangle - c(\theta)}{a(\phi)} - b(y,\phi)\right), \qquad (1)$$

where

- y, θ, and $c(\theta)$ are as before
- ϕ is a dispersion parameter
- $b(y,\phi)$ is a function of the data y and the dispersion parameter ϕ.

Notice that the density (1) is a generalization of the exponential family density which specifies that $a(\phi) = 1$ and $b(y,\phi) = \log(h(y))$.

This is not a canonical exponential family model. What happens to sufficiency?

Note that the dispersion parameter can be estimated using

$$\hat{\phi} = \frac{\sum_{i=1}^{n}(y - \hat{\mu}_i)^2/\hat{\mu}_i}{n - p}.$$

Notice that the estimation of the dispersion and the regression parameters is independent, so choosing a dispersion other than one has no effect on the regression parameter estimates.

We investigate the overdispersed Poisson regression model with respect to the Galapogas data.

```r
n <- nrow(gala)
p <- length(coef(m1))
y <- gala$Species

## estimate dispersion directly
fits <- predict(m1, type = "response")
dp <- sum((y - fits)^2/fits) / (n - p)
dp
```

```
## [1] 29.9553
```

```r
summary(m1, dispersion = dp)
```

```
##
## Call:
## glm(formula = Species ~ Elevation + Nearest + Scruz + Adjacent +
##     Size, family = "poisson", data = gala, x = TRUE)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -10.3723   -3.5214   -0.9947    1.7193   10.6627
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.7897965  0.4437393   6.287 3.24e-10 ***
## Elevation    0.0009361  0.0002957   3.166  0.00154 **
## Nearest      0.0064693  0.0095646   0.676  0.49880
## Scruz       -0.0062665  0.0034307  -1.827  0.06776 .
## Adjacent    -0.0002858  0.0001620  -1.764  0.07781 .
## Size2        1.1276155  0.5218686   2.161  0.03072 *
## Size3        2.0586771  0.5155262   3.993 6.51e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 29.9553)
##
##     Null deviance: 3510.73  on 29  degrees of freedom
## Residual deviance:  594.18  on 23  degrees of freedom
## AIC: 769.01
##
## Number of Fisher Scoring iterations: 5
```

```r
m2 <- glm(Species ~ Elevation + Nearest + Scruz + Adjacent + Size,
          family = "quasipoisson", data = gala, x = TRUE)
summary(m2)
```

```
##
## Call:
## glm(formula = Species ~ Elevation + Nearest + Scruz + Adjacent +
##     Size, family = "quasipoisson", data = gala, x = TRUE)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -10.3723   -3.5214   -0.9947    1.7193   10.6627
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.7897965  0.4437452   6.287 2.05e-06 ***
## Elevation    0.0009361  0.0002957   3.166 0.004314 **
## Nearest      0.0064693  0.0095648   0.676 0.505552
## Scruz       -0.0062665  0.0034308  -1.827 0.080777 .
## Adjacent    -0.0002858  0.0001621  -1.764 0.091094 .
## Size2        1.1276155  0.5218755   2.161 0.041376 *
## Size3        2.0586771  0.5155330   3.993 0.000572 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 29.95609)
##
##     Null deviance: 3510.73  on 29  degrees of freedom
## Residual deviance:  594.18  on 23  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

In this case the dispersion is quite large leading to an increase in standard errors of over a factor of 5

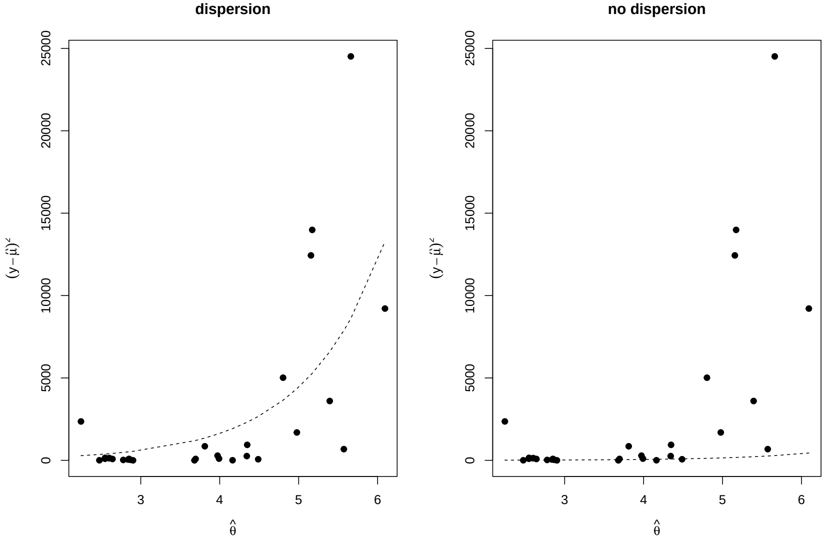
```
c(dp, sqrt(dp))
```

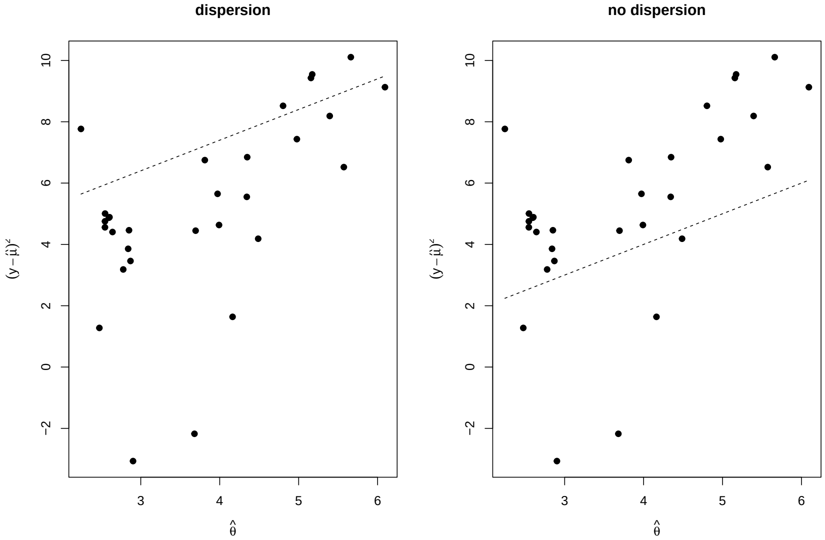
```
## [1] 29.955296  5.473143
```

```
se <- function(model) sqrt(diag(vcov(model)))
round(data.frame(coef.m1=coef(m1), coef.m2=coef(m2), se.m1=se(m1), se.m2=se(m2),
                 ratio=se(m2)/se(m1)), 4)
```

```
##             coef.m1 coef.m2  se.m1  se.m2  ratio
## (Intercept)  2.7898  2.7898 0.0811 0.4437 5.4732
## Elevation    0.0009  0.0009 0.0001 0.0003 5.4732
## Nearest      0.0065  0.0065 0.0017 0.0096 5.4732
## Scruz       -0.0063 -0.0063 0.0006 0.0034 5.4732
## Adjacent    -0.0003 -0.0003 0.0000 0.0002 5.4732
## Size2        1.1276  1.1276 0.0954 0.5219 5.4732
## Size3        2.0587  2.0587 0.0942 0.5155 5.4732
```

The overdispersed Poisson model clearly offers improvements to the residuals vs fitted values plot




dispersion
no dispersion
$(y-\hat{\mu})^2$
$(y-\hat{\mu})^2$
−2
0
2
4
6
8
10
3
4
5
6
$\hat{\theta}$
$\hat{\theta}$

The AER package includes a function that allows for one to test whether or not dispersion is present. We can see that dispersion is present at most reasonable significance testing levels.

```
library(AER)
dispersiontest(m1, trafo=1)
```

```
##
##  Overdispersion test
##
## data:  m1
## z = 2.5007, p-value = 0.006198
## alternative hypothesis: true alpha is greater than 0
## sample estimates:
##    alpha
## 21.92009
```

Note that the AER package defines dispersion differently than the glm function.

Let's consider our better performing model from last time.

```r
m3 <- glm(Species ~ Elevation + I(Elevation^2) + Nearest + Scruz +
            I(Scruz^2) + Adjacent +
            Area + I(Area^2), family = "poisson", data = gala, x = TRUE)
```

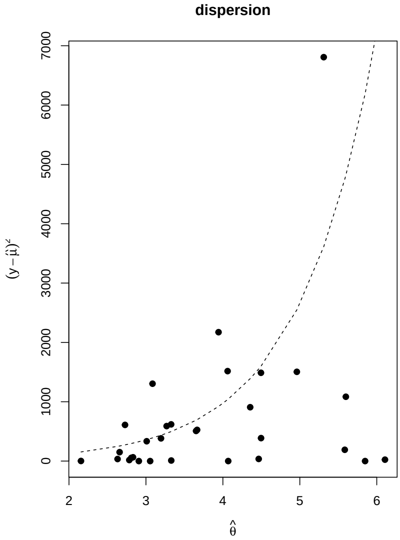
This model appears to also be over dispersed.

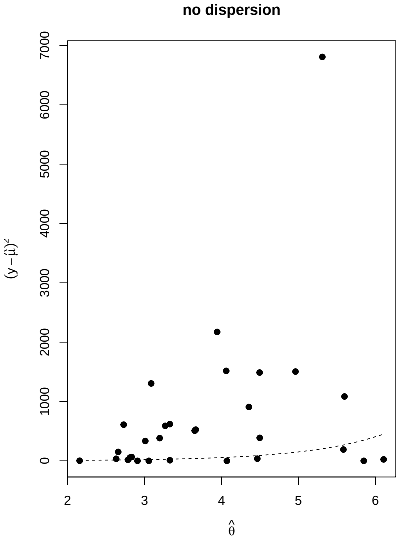
```r
p <- length(coef(m3))

fits <- predict(m3, type = "response")
dp3 <- sum((y - fits)^2/fits) / (n - p)
dp3
```

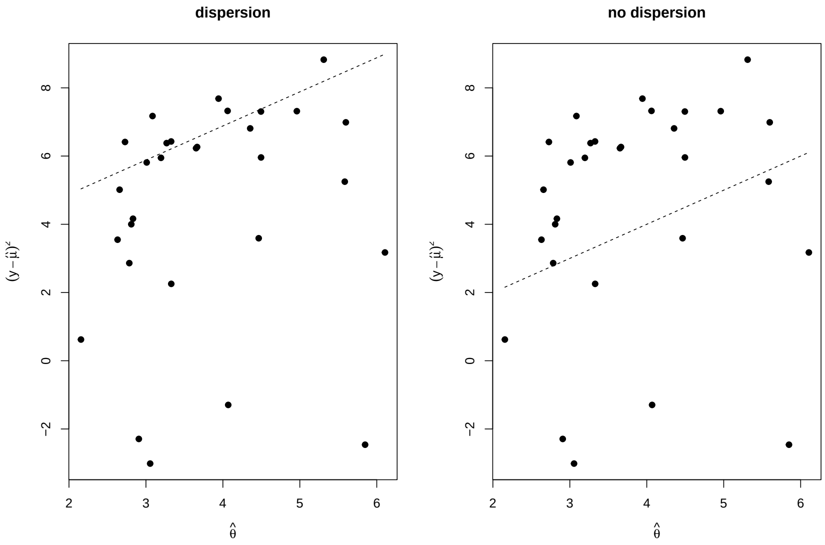
```
## [1] 17.84607
```

```r
dispersiontest(m3, trafo=1)
```

```
##
##  Overdispersion test
##
## data:  m3
## z = 4.2173, p-value = 1.236e-05
## alternative hypothesis: true alpha is greater than 0
## sample estimates:
##    alpha
## 11.47278
```


dispersion
$(y-\hat{\mu})^2$
0
1000
2000
3000
4000
5000
6000
7000
2
3
4
5
6
$\hat{\theta}$


no dispersion
$(y-\hat{\mu})^2$
0
1000
2000
3000
4000
5000
6000
7000
2
3
4
5
6
$\hat{\theta}$


dispersion
no dispersion
(y − μ̂)²
−2
0
2
4
6
8
2
3
4
5
6
θ̂

Look at residuals

We will now remove Santa Maria.

```r
sort(round((gala$Species - fitted(m3))^2, 2),
     decreasing = TRUE)
```

```
##   SantaMaria     Gardner2     Espanola        Pinta     Marchena     Gardner1
##      6807.27      2172.44      1516.23      1504.33      1487.96      1304.18
##  SanSalvador       Pinzon     Caldwell     Genovesa      Enderby      SantaFe
##      1084.19       907.83       618.86       609.66       589.81       525.89
##      Tortuga       Rabida      Seymour      Coamano SanCristobal       Onslow
##       507.92       387.17       382.86       334.32       190.45       150.49
##     Champion Daphne.Minor   Fernandina         Eden    SantaCruz   Las.Plazas
##        64.28        54.63        36.26        34.75        23.89        17.49
##    Bartolome       Darwin       Baltra Daphne.Major      Isabela         Wolf
##         9.52         1.86         0.27         0.10         0.09         0.05
```

```r
m4 <- glm(Species ~ Elevation + I(Elevation^2) + Nearest + Scruz +
            I(Scruz^2) + Adjacent + Area + I(Area^2), family = "poisson",
            data = gala[-27, ], x = TRUE)
summary(m4)
```

```
## 
## Call:
## glm(formula = Species ~ Elevation + I(Elevation^2) + Nearest + 
##     Scruz + I(Scruz^2) + Adjacent + Area + I(Area^2), family = "poisson", 
##     data = gala[-27, ], x = TRUE)
## 
## Deviance Residuals: 
##     Min       1Q   Median       3Q      Max  
## -7.4154  -2.4285  -0.0320   0.9377   6.7205  
## 
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)    
## (Intercept)     2.442e+00  1.042e-01  23.436  < 2e-16 ***
## Elevation       5.697e-03  4.611e-04  12.355  < 2e-16 ***
## I(Elevation^2) -4.090e-06  4.650e-07  -8.796  < 2e-16 ***
## Nearest        -4.035e-03  2.475e-03  -1.630 0.103063    
## Scruz           6.016e-03  1.685e-03   3.570 0.000357 ***
## I(Scruz^2)     -2.944e-05  5.956e-06  -4.943 7.70e-07 ***
## Adjacent        2.368e-04  8.877e-05   2.667 0.007648 ** 
## Area            2.160e-03  1.785e-04  12.103  < 2e-16 ***
## I(Area^2)      -2.191e-07  3.251e-08  -6.740 1.58e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for poisson family taken to be 1)
## 
##     Null deviance: 3205.62  on 28  degrees of freedom
## Residual deviance:  356.08  on 20  degrees of freedom
## AIC: 527.42
## 
## Number of Fisher Scoring iterations: 5
```

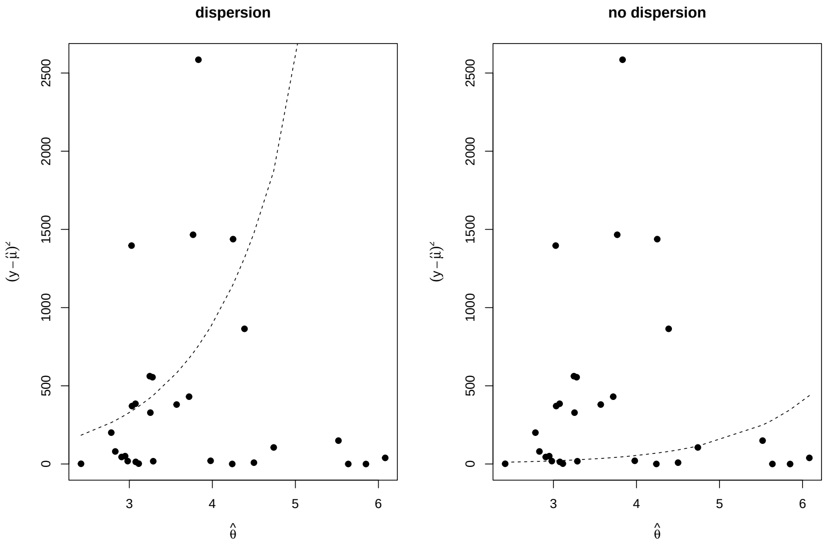
Let’s calculate dispersion for this new model

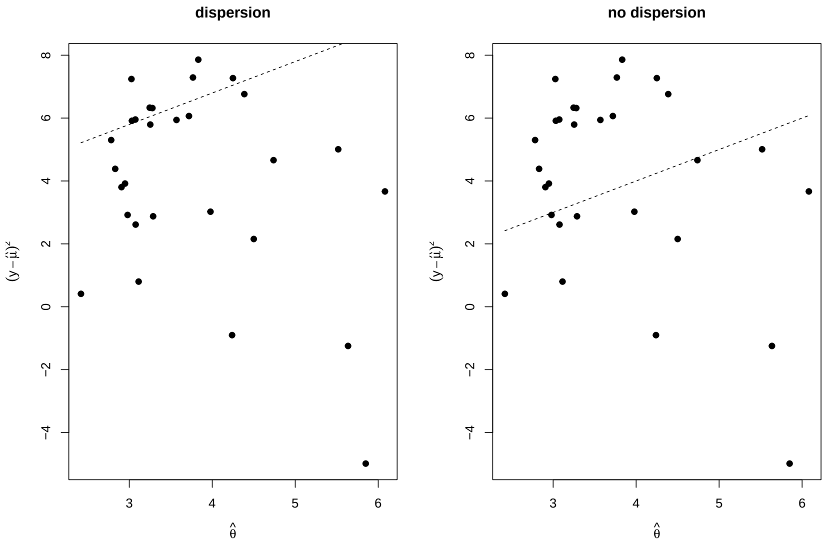
```
p <- length(coef(m4))
n <- nrow(gala) - 1
y <- gala[-27, ]$Species

## estimate dispersion directly
fits <- predict(m4, type = "response")
dp4 <- sum((y - fits)^2/fits) / (n - p)
dp4
```

```
## [1] 16.39359
```

```
dispersiontest(m4, trafo=1)
```

```
##
##  Overdispersion test
##
## data:  m4
## z = 3.3744, p-value = 0.0003698
## alternative hypothesis: true alpha is greater than 0
## sample estimates:
##    alpha
## 10.31668
```


dispersion
no dispersion
$(y-\hat{\mu})^2$
0
500
1000
1500
2000
2500
3
4
5
6
$\hat{\theta}$


dispersion
no dispersion
$(y-\hat{\mu})^2$
$\hat{\theta}$
8
6
4
2
0
−2
−4
3
4
5
6

Matching of large species counts is pretty good.

```
foo <- data.frame(names = rownames(gala[-27, ]),
                  Species = y,
                  pred = as.numeric(predict(m4, type = "response")),
                  resid = residuals(m4))
head(cbind(foo %>% arrange(desc(Species)) %>% pull(names),
           foo %>% arrange(desc(pred)) %>% pull(names)), 10)
```

```
##       [,1]          [,2]
##  [1,] "SantaCruz"   "SantaCruz"
##  [2,] "Isabela"     "Isabela"
##  [3,] "SanCristobal" "SanCristobal"
##  [4,] "SanSalvador" "SanSalvador"
##  [5,] "Pinzon"      "Pinta"
##  [6,] "Pinta"       "Fernandina"
##  [7,] "Espanola"    "Marchena"
##  [8,] "Fernandina"  "Pinzon"
##  [9,] "Rabida"      "Rabida"
## [10,] "SantaFe"     "Baltra"
```

There are still some misses, mostly among small counts.

```r
foo %>% arrange(desc(abs(resid))) %>% head(10)
```

```
##             names Species     pred     resid
## Gardner2 Gardner2       5 43.28636 -7.415439
## Gardner1 Gardner1      58 20.63096  6.720517
## Espanola Espanola      97 46.15761  6.510695
## Enderby   Enderby       2 25.69789 -6.097764
## Caldwell Caldwell       3 26.55618 -5.833392
## Coamano   Coamano       2 21.62974 -5.453054
## Onslow     Onslow       2 16.16162 -4.468251
## Pinzon     Pinzon     108 70.08212  4.192417
## Genovesa Genovesa      40 20.74950  3.742670
## Tortuga   Tortuga      16 35.49805 -3.673642
```

Which model?

We can additionally consider information criteria:

```
AIC(m1, m3, m4)
```

```
## Warning in AIC.default(m1, m3, m4): models are not all fitted to the same number
## of observations

##    df      AIC
## m1  7 769.0063
## m3  9 590.7201
## m4  9 527.4177
```

```
BIC(m1, m3, m4)
```

```
## Warning in BIC.default(m1, m3, m4): models are not all fitted to the same number
## of observations

##    df      BIC
## m1  7 778.8146
## m3  9 603.3309
## m4  9 539.7234
```

Negative Binomial regression

Given a series of independent trials, each trial with probability of success p, let Z be the number of trials until the kth success.

The negative binomial distribution can arise naturally in a few ways:

- One can envision a system that can withstand k hits before failure. The probability of a hit in a given time period is p.
- The negative binomial also arises from the generalization of the Poisson where the rate parameter is gamma distributed.

The mass function for the negative binomial distribution is:

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

We get a more convenient parameterization if we let $Y=Z-k$ and $p=(1+\alpha)^{-1}$ so that:

$$\mathbb{P}(Y=y)=\binom{y+k-1}{k-1}\frac{\alpha^y}{(1+\alpha)^{y+k}}, \qquad y=0,1,2,\dots.$$

Then $\mathrm{E}(Y)=\mu=k\alpha$ and $\mathrm{Var}(Y)=k\alpha+k\alpha^2=\mu+\mu^2/k$. The log-likelihood is then

$$\sum_{i=1}^{n}\left(y_i\log\left(\frac{\alpha}{1+\alpha}\right)-k\log(1+\alpha)+\sum_{j=0}^{y_i-1}\log(j+k)-\log(y_i!)\right).$$

Can the log likelihood on the previous slide be written in canonical form?

The most convenient way to link the mean response μ to a linear combination of the the predictors X in typical GLM fashion is through

$$\log\left(\frac{\mu}{\mu+k}\right) = \log\left(\frac{\alpha}{1+\alpha}\right) = \theta = x'\beta.$$

We can specify the change of parameters map $g : \theta \to \mu$ and the link function as $g^{-1} : \mu \to \theta$ as

$$g(\theta) = \frac{ke^{\theta}}{1-e^{\theta}} = \mu, \qquad g^{-1}(\mu) = \log\left(\frac{\mu}{\mu+k}\right) = x'\beta.$$

We can regard k as fixed and determined by the application or as an additional parameter to be estimated.

Example: Solder data

ATT ran an experiment varying five factors relevant to a wave-soldering procedure for mounting components on printed circuit boards.

The response variable, skips, is a count of how many solder skips appeared in a visual inspection.

The data comes from Comizzoli et al. (1990) (See the source material on the help page for the solder dataset in the faraway package).

We start with a Poisson regression:

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

```
## [1] 1796.973  881.000
```

We see that the full model has a residual deviance of 1797 on 881 degrees of freedom. This is not a good fit (as a rule of thumb, deviance should be less than degrees of freedom for a well-fitting submodel).

Perhaps including interaction terms will improve the fit:

```
modp2 <- glm(skips ~ (Opening +Solder + Mask + PadType + Panel)^2,
             family=poisson, data=solder)
c(deviance(modp2), df.residual(modp2))
```

```
## [1] 1007.736  773.000
```

```
pchisq(deviance(modp2), df.residual(modp2), lower=FALSE)
```

```
## [1] 2.238499e-08
```

The fit is improved but not enough to conclude that the model fits. We could investigate:

- adding more interactions but that would make interpretation increasingly difficult.
- A check for outliers reveals no problem.

Try negative binomial regression

The functions for fitting come from the `MASS` package. Note that the `MASS` package has a `select` function that will clash with the `select` function in `dplyr`. Write `dplyr::select` for data wrangling.

We can specify the link parameter k. Here we choose $k = 1$ to demonstrate the method, although there is no substantive motivation from this application to use this value.

Note that the $k = 1$ case corresponds to an assumption of a geometric distribution for the response.

```
library(MASS)
```

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

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

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

```
##
## Call:  glm(formula = skips ~ ., family = negative.binomial(1), data = solder)
##
## Coefficients:
## (Intercept)     OpeningM     OpeningS   SolderThin       MaskA3       MaskA6
##    -1.57251      0.50852      2.00093      1.04754      0.65065      2.52776
##      MaskB3       MaskB6    PadTypeD6    PadTypeD7    PadTypeL4    PadTypeL6
##     1.26631      2.07062     -0.45434      0.01979      0.46751     -0.46812
##   PadTypeL7    PadTypeL8    PadTypeL9    PadTypeW4    PadTypeW9       Panel2
##    -0.28999     -0.08057     -0.51864     -0.13917     -1.48133      0.29536
##      Panel3
##     0.34262
##
## Degrees of Freedom: 899 Total (i.e. Null);  881 Residual
## Null Deviance:       1743
## Residual Deviance: 556.7     AIC: 3884
```

```
## LRT test
pchisq(deviance(modn), df.residual(modn), lower=FALSE)
```

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

We can estimate the parameter k using maximum likelihood estimation in:

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

```
##
## Call:
## glm.nb(formula = skips ~ ., data = solder, init.theta = 4.52811339,
##     link = log)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7047  -1.0109  -0.3921   0.4480   2.8869
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.29859    0.13202  -9.837  < 2e-16 ***
## OpeningM     0.50353    0.07932   6.348 2.18e-10 ***
## OpeningS     1.91104    0.07111  26.876  < 2e-16 ***
## SolderThin   0.93513    0.05323  17.567  < 2e-16 ***
## MaskA3       0.58383    0.09592   6.087 1.15e-09 ***
## MaskA6       2.26096    0.10101  22.384  < 2e-16 ***
## MaskB3       1.20651    0.09572  12.605  < 2e-16 ***
## MaskB6       1.98172    0.09158  21.638  < 2e-16 ***
## PadTypeD6   -0.46189    0.11145  -4.144 3.41e-05 ***
## PadTypeD7   -0.03182    0.10584  -0.301 0.763655
## PadTypeL4    0.38119    0.10177   3.745 0.000180 ***
## PadTypeL6   -0.57860    0.11327  -5.108 3.25e-07 ***
## PadTypeL7   -0.36569    0.11006  -3.323 0.000891 ***
## PadTypeL8   -0.15882    0.10734  -1.480 0.138953
## PadTypeL9   -0.56554    0.11306  -5.002 5.67e-07 ***
## PadTypeW4   -0.19851    0.10783  -1.841 0.065630 .
## PadTypeW9   -1.56332    0.13538 -11.547  < 2e-16 ***
## Panel2       0.29574    0.06322   4.678 2.90e-06 ***
## Panel3       0.33380    0.06298   5.300 1.16e-07 ***
## ---
```

We see that $\hat{k} = 4.528$ with a standard error of 0.518.

We can compare negative binomial models using the usual inferential techniques. For instance, we see that the overall fit is much improved.

```
## LRT test
pchisq(deviance(modn), df.residual(modn), lower=FALSE)
```

```
## [1] 0.001367546
```

Zero Inflated Count Models

Sometimes we see count response data where the number of zeroes appearing is significantly greater than the Poisson or negative binomial models would predict.

This commonly arises in

- life history analyses of plants and animals where many subjects die before they reproduce, arrest
- bookings data where many people are either not arrested or receive zero day sentences,
- insurance claims data.

Modifying the Poisson by adding a dispersion parameter does not adequately model this divergence from the standard count distributions.

Example: Biochemistry graduate students

We consider a sample of 915 biochemistry graduate students.

The response variable is the number of articles produced during the last three years of the PhD. We are interested in how this is

influenced by the gender, marital status, number of children, prestige of the department and productivity of the advisor of the student.

The dataset may be found in the `pscl` package.

We start by fitting a Poisson regression model:

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

```
##                Estimate Std. Error z value  Pr(>|z|)
## (Intercept)  0.3046168  0.1029814  2.9580  0.003097
## femWomen    -0.2245942  0.0546135 -4.1124 3.915e-05
## marMarried   0.1552434  0.0613744  2.5294  0.011424
## kid5        -0.1848827  0.0401269 -4.6075 4.076e-06
## phd          0.0128226  0.0263970  0.4858  0.627139
## ment         0.0255427  0.0020061 12.7327 < 2.2e-16
##
## n = 915 p = 6
## Deviance = 1634.37098 Null Deviance = 1817.40530 (Difference = 183.03432)
```

We can see that deviance is significantly larger than the degrees of freedom (a rule of thumb indicated poor fit).

Some experimentation reveals that this cannot be solved by using a richer linear predictor or by eliminating some outliers (see Faraway).

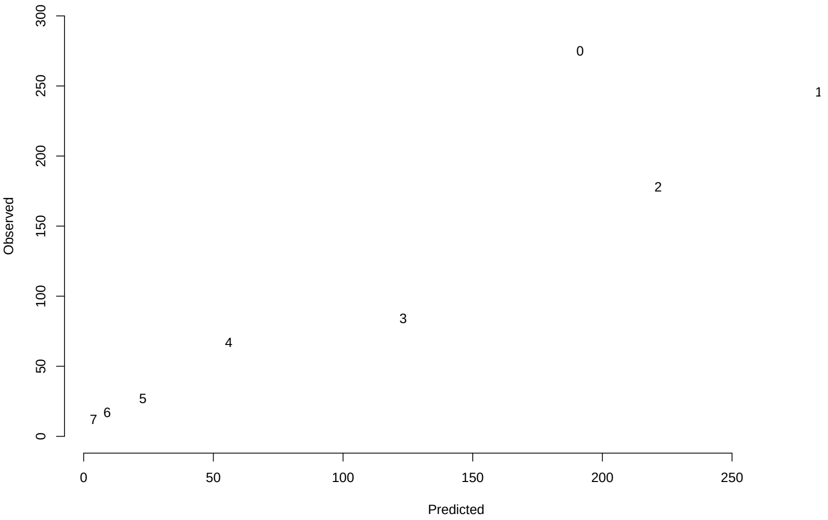
We might consider a dispersed Poisson model or negative binomial but some thought suggests that there are good reasons why a disproportionate number of students might produce no articles at all.

We count and predict how many students produce between zero and seven articles. Very few students produce more than seven articles so we ignore these.

The `predprob` function produces the predicted probabilities for each case. By summing these, we get the expected number for each article count.

```
ocount <- table(bioChemists$art)[1:8]
pcount <- colSums(predprob(modp)[,1:8])
```

```
plot(pcount, ocount, type="n", xlab="Predicted", ylab="Observed",
     ylim = c(0, 300), axes = FALSE)
axis(side = 1)
axis(side = 2)
text(pcount,ocount, 0:7)
```



We see that there are many more students with zero articles than would be predicted by the Poisson model. In contrast, the relationship between observed and predicted is linear for the students who produce at least one article.

We now consider a zero-inflated Poisson model. First, some motivation.

Suppose we ask the general public how many games of chess they have played in the last month:

- Some people will say zero because they do not play chess
- Some will say zero because they are chess players who did not play in the last month.

Circumstances such as these require a *mixture* model.

A general specification of this model takes the form:

$$\begin{aligned}\mathbb{P}(Y=0) &= \phi + (1-\phi)f(0)\\ \mathbb{P}(Y=j) &= (1-\phi)f(j), \qquad j>0.\end{aligned}$$

The parameter ϕ represents the proportion of subjects who will always respond zero (the non-chess players in the motivating example).

One can model the proportion ϕ using a binary response model.

The distribution f models the counts of those individuals that can have a positive response.

We can use a Poisson model for f in which case this is called zero-inflated Poisson model.

```
modz <- zeroinfl(art ~ ., data=bioChemists)
summary(modz)
```

```
##
## Call:
## zeroinfl(formula = art ~ ., data = bioChemists)
##
## Pearson residuals:
##     Min      1Q  Median      3Q     Max
## -2.3253 -0.8652 -0.2826  0.5404  7.2976
##
## Count model coefficients (poisson with log link):
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.640839   0.121307   5.283 1.27e-07 ***
## femWomen    -0.209144   0.063405  -3.299 0.000972 ***
## marMarried   0.103750   0.071111   1.459 0.144567
## kid5        -0.143320   0.047429  -3.022 0.002513 **
## phd         -0.006166   0.031008  -0.199 0.842376
## ment         0.018098   0.002294   7.888 3.07e-15 ***
##
## Zero-inflation model coefficients (binomial with logit link):
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.577060   0.509386  -1.133  0.25728
## femWomen     0.109752   0.280082   0.392  0.69517
## marMarried  -0.354018   0.317611  -1.115  0.26501
## kid5         0.217095   0.196483   1.105  0.26920
## phd          0.001275   0.145263   0.009  0.99300
## ment        -0.134114   0.045243  -2.964  0.00303 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Number of iterations in BFGS optimization: 19
## Log-likelihood: -1605 on 12 Df
```

We notice that the `ment` variable which counts the number of articles produced by the mentor is the most significant predictor in both the Poisson and binary regressions.

This is because the zero-inflated approach models the probability of a zero count (not the probability of a successful 1 count). Hence there is no contradiction.

We can use the standard likelihood testing theory to compare nested models. For example, suppose we consider a simplified version of the zero-inflated Poisson model where we now have different predictors for the two components of the model. The count part of the model is specified before the $|$ and the binary response model after.

```r
# smaller model
modz2 <- zeroinfl(art ~ fem+kid5+ment | ment, data=bioChemists)

# summary table for smaller model
summary(modz2)
```

```
##
## Call:
## zeroinfl(formula = art ~ fem + kid5 + ment | ment, data = bioChemists)
##
## Pearson residuals:
##     Min      1Q  Median      3Q     Max
## -2.2802 -0.8807 -0.2718  0.5131  7.4788
##
## Count model coefficients (poisson with log link):
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.694517   0.053025  13.098  < 2e-16 ***
## femWomen    -0.233857   0.058400  -4.004 6.22e-05 ***
## kid5        -0.126516   0.039668  -3.189  0.00143 **
## ment         0.018004   0.002224   8.096 5.67e-16 ***
##
## Zero-inflation model coefficients (binomial with logit link):
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.68488    0.20529  -3.336 0.000849 ***
## ment        -0.12680    0.03981  -3.185 0.001448 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Number of iterations in BFGS optimization: 22
## Log-likelihood: -1608 on 6 Df
```

```r
# test of nested models
pchisq(2*(modz$loglik-modz2$loglik), 6, lower = FALSE)
```

```
## [1] 0.4041153
```

For interpretation, the exponentiated coefficients are more useful:

```
exp(coef(modz2))
```

```
## count_(Intercept)    count_femWomen        count_kid5        count_ment 
##         2.0027411         0.7914748         0.8811604         1.0181669 
##  zero_(Intercept)         zero_ment 
##         0.5041522         0.8809081
```

We can also use the model to make predictions. Consider a single male with no children whose mentor produced six articles:

```
newman <- data.frame(fem="Men",mar="Single",kid5=0,ment=6)
predict(modz2, newdata=newman, type="prob")[1:6]
```

```
## [1] 0.27758792 0.19394034 0.21635997 0.16091421 0.08975799 0.04005363
```

We see that most likely outcome for this student is that no articles will be produced with a probability of 0.278. We can query the probability of no production from the zero part of the model:

```
predict(modz2, newdata=newman, type="zero")
```

```
##        1
## 0.190666
```

So the additional probability to make this up to 0.278 comes from the Poisson count part of the model. This difference might be attributed to students who had the potential to write an article.

We note that the zero-inflated Poisson distribution also arises as a special case of an aster model used in life history analyses for populations of plants or animals.

The aster model can be thought of as a generalized generalized linear regression model and it models individuals through stages of their lifecycle. One can think of a simple lifecycle:

$$1 \to Y_1 \to Y_2,$$

where

- Y_1 encodes subjects either reaching or not reaching a reproductive stage (modeled as a binary random variable) and then, conditional on reproduction
- Y_2 encodes how many offspring produced by the subjects in the reproductive stage (modeled as a 0-truncated Poisson random variable).