Modelling Count Variables with R

- This talks is about regression methods in which the dependent variable takes nonnegative integer or count values.
- The dependent variable is usually the number of times an event occurs.

- Linear regression is used to model and predict continuous measurement variables.
- Poisson regression is used to model and predict discrete count variables.

Poisson regression assumes the response variable Y has a Poisson distribution, and assumes the logarithm of its expected value can be modeled by a linear combination of unknown parameters. A Poisson regression model is sometimes known as a log-linear model, especially when used to model contingency tables.

Examples of Poisson regression

(1) The number of persons killed by mule or horse kicks in the Prussian army per year. Ladislaus Bortkiewicz collected data from 20 volumes of *Preussischen Statistik*. These data were collected on 10 corps of the Prussian army in the late 1800s over the course of 20 years.

(2) The number of people in line in front of you at the grocery store. Predictors may include the number of items currently offered at a special discounted price and whether a special event (e.g., a holiday, a big sporting event) is three or fewer days away.

(3) The number of awards earned by students at one high school. Predictors of the number of awards earned include the type of program in which the student was enrolled (e.g., vocational, general or academic) and the score on their final exam in math.

Overview

Some examples of event counts are:

- number of claims per year on a particular car owners insurance policy,
- number of workdays missed due to sickness of a dependent in a one-year period,
- number of papers published per year by a researcher.

Poisson Distribution

- The number of persons killed by mule or horse kicks in the Prussian army per year.
- Ladislaus Bortkiewicz collected data from 20 volumes of Preussischen Statistik.
- These data were collected on 10 corps of the Prussian army in the late 1800s over the course of 20 years, giving a total of 200 observations of one corps for a one year period. The period or module of observation is thus one year.

Poisson Distribution: Prussian Cavalary

- The total deaths from horse kicks were 122, and the average number of deaths per year per corps was thus $122/200 = 0.61$.
- In any given year, we expect to observe, well, not exactly 0.61 deaths in one corps
- Here, then, is the classic Poisson situation: a rare event, whose average rate is small, with observations made over many small intervals of time.

```
rpois(200,lambda=0.61)
> X
[1]   1 2 0 1 0 3 0 0 1 0 0 4 0 0 0 1 0 1 0 2
[21]  0 0 0 2 2 0 0 0 1 0 0 0 0 1 0 0 0 1 2 0
[41]  0 0 1 0 1 0 1 0 0 1 1 0 1 0 0 1 0 0 3 1
.......
.......
[141] 0 0 0 0 1 2 0 1 0 1 0 0 0 0 0 0 0 1 0 0
[161] 1 0 1 0 0 0 0 1 0 0 0 0 0 1 1 1 0 2 0 1
[181] 0 0 2 0 2 0 0 1 0 0 3 1 0 0 0 1 1 0 0 0
>
> mean(X)
[1] 0.53
> var(X)
[1] 0.5317588
```

Overview

- Poisson regression is main technique used to model count variables.
- Poisson Distribution : Mean and Variance are equal

$$\mathrm{E}(X) = \mathrm{Var}(X)$$

- Sometimes conventional Poisson Regression is not an appropriate technique, and alternative or variant techniques are used instead.
- For example, Negative Binomial regression is for modelling count variables, usually for over-dispersed count outcome variables.

Generalized Linear Models

- In statistics, the problem of modelling count variables is an example of generalized linear modelling.
- Generalized linear models are fit using the `glm()` function.
- The form of the `glm` function is

```
glm(formula, family=familytype(link=linkfunction),
  data=dataname)
```

Generalized Linear Models

Family	Default Link Function
binomial	`(link = "logit")`
gaussian	`(link = "identity")`
Gamma	`(link = "inverse")`
inverse.gaussian	`(link = "1/`mu^2`")`
poisson	`(link = "log")`
quasibinomial	`(link = "logit")`
quasipoisson	`(link = "log")`

Texts on GLMs

- Dobson, A. J. (1990) An Introduction to Generalized Linear Models. (*London: Chapman and Hall.*)
- Hastie, T. J. and Pregibon, D. (1992) Generalized linear models. Chapter 6 of Statistical Models in S eds J. M. Chambers and T. J. Hastie, Wadsworth & Brooks/Cole.
- McCullagh P. and Nelder, J. A. (1989) Generalized Linear Models. (*London: Chapman and Hall.*)
- Venables, W. N. and Ripley, B. D. (2002) Modern Applied Statistics with S. *New York: Springer.*

glm2: Fitting Generalized Linear Models

Author(s): Ian Marschner

Fits generalized linear models using the same model specification as glm in the stats package, but with a modified default fitting method that provides greater stability for models that may fail to converge using glm

VGAM: Vector Generalized Linear and Additive Models

Author(s): Thomas W. Yee (t.yee@auckland.ac.nz)

URL: http://www.stat.auckland.ac.nz/∼ yee/VGAM

Vector generalized linear and additive models, and associated models (Reduced-Rank VGLMs, Quadratic RR-VGLMs, Reduced-Rank VGAMs).

This package fits many models and distribution by maximum likelihood estimation (MLE) or penalized MLE. Also fits constrained ordination models in ecology.

Poisson Regression

- Poisson regression is used to model count variables.
- Poisson regression has a number of extensions useful for count models.

Examples of Poisson regression

- The number of awards earned by students at a secondary or high school.
- Predictors of the number of awards earned include the type of program in which the student was enrolled (e.g., vocational, general or academic) and the score on their final exam in math.

Conventional OLS regression

- Count outcome variables are sometimes log-transformed and analyzed using OLS regression.
- Many issues arise with this approach, including loss of data due to undefined values generated by taking the log of zero (which is undefined) and biased estimates.

Description of the data

- For the purpose of illustration, we have simulated a data set for the last example.
- The data set is called *poissonreg.csv*
- In this example, **num_awards** is the outcome variable and indicates the number of awards earned by students at a high school in a year

Predictor Variables

- **math** is a continuous predictor variable and represents students' scores on their math final exam,
- **prog** is a categorical predictor variable with three levels indicating the type of program in which the students were enrolled.
- **prog** is coded as $1 =$ "*General*", $2 =$ "*Academic*" and $3 =$ "*Vocational*".

Poisson Regression with R

```
       id         num_awards          prog          math      
 1      :  1   Min.   :0.00   General   : 45   Min.   :33.0  
 2      :  1   1st Qu.:0.00   Academic  :105   1st Qu.:45.0  
 3      :  1   Median :0.00   Vocational: 50   Median :52.0  
 4      :  1   Mean   :0.63                   Mean   :52.6  
 5      :  1   3rd Qu.:1.00                   3rd Qu.:59.0  
 6      :  1   Max.   :6.00                   Max.   :75.0  
   (Other):194
```

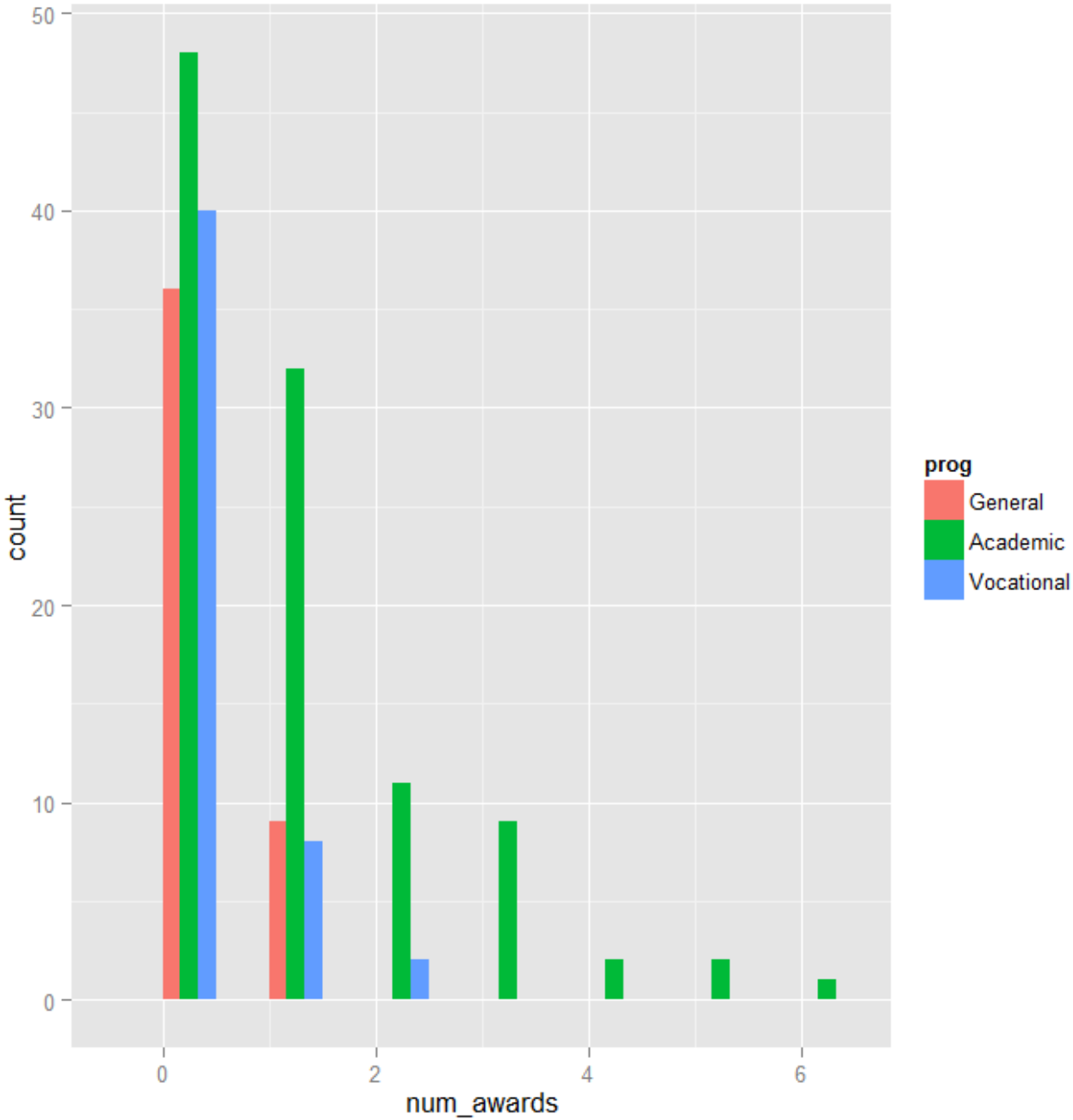

50
40
30
20
10
0
count
0
2
4
6
num_awards
prog
General
Academic
Vocational

Figure:

Poisson Regression with R

- Each variable has 200 valid observations and their distributions seem quite reasonable.
- The mean and variance of our outcome variable are more or less the same.
- Our model assumes that these values, conditioned on the predictor variables, will be equal (or at least roughly so).

Poisson Regression with R

- Additionally, the means and variances within each level of prog–*the conditional means and variances*–are similar.
- A conditional histogram separated out by program type is plotted to show the distribution.

Poisson regression

- At this point, we are ready to perform our Poisson regression model analysis using the `glm` function.
- We fit the model and save it in the object `model1` and get a summary of the model.

Poisson Regression with R

```
model1 <- glm(num_awards ~ prog + math,
family="poisson", data=poissonreg)

summary(model1)
```

Poisson Regression with R

```
Call:
glm(formula = num_awards ~ prog + math,
      family = "poisson",
      data = poissonreg)

Deviance Residuals:
Min       1Q   Median       3Q      Max
-2.204  -0.844  -0.511    0.256    2.680
```

Poisson Regression with R

```
Coefficients:
Estimate Std. Error z value Pr(>|z|)
(Intercept)     -5.2471     0.6585   -7.97  1.6e-15 ***
progAcademic     1.0839     0.3583    3.03   0.0025 **
progVocational   0.3698     0.4411    0.84   0.4018
math             0.0702     0.0106    6.62  3.6e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '
```

Poisson Regression with R

```
(Dispersion parameter for poisson family taken to be 1)

Null deviance: 287.67  on 199  degrees of freedom
Residual deviance: 189.45  on 196  degrees of freedom
AIC: 373.5

Number of Fisher Scoring iterations: 6
```

Poisson Regression with R

glm function output

- The output begins with echoing the function call. Then the information on deviance residuals is displayed.
- Deviance residuals are approximately normally distributed if the model is specified correctly.
- Here it shows a little bit of skeweness since median is not quite zero.

Poisson Regression with R

```
Call:
glm(formula = num_awards ~ prog + math,
family = "poisson",
data = poissonreg)

Deviance Residuals:
Min       1Q     Median       3Q      Max
-2.204  -0.844  -0.511    0.256    2.680
```

glm function output

- The Poisson regression coefficients for each of the variables along with the standard errors, z-scores, p-values and 95% confidence intervals for the coefficients.
- The coefficient for math is 0.07.
- This means that the expected log count for a one-unit increase in math is 0.07.

Poisson Regression with R

```
Coefficients:
Estimate Std. Error z value Pr(>|z|)
(Intercept)     -5.2471     0.6585   -7.97  1.6e-15 ***
progAcademic     1.0839     0.3583    3.03   0.0025 **
progVocational   0.3698     0.4411    0.84   0.4018
math             0.0702     0.0106    6.62  3.6e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '
```

Poisson Regression with R

glm function output

- The indicator variable **progAcademic** compares between **prog = Academic** and **prog = "General"** , the expected log count for **prog = Academic** increases by about 1.1.
- The indicator variable **prog.Vocational** is the expected difference in log count (≈ 0.37) between **prog = "Vocational"** and the reference group (**prog = "General"**).

Deviance

- In statistics, deviance is a quality of fit statistic for a model that is often used for statistical hypothesis testing.
- It is a generalization of the idea of using the sum of squares of residuals in ordinary least squares to cases where model-fitting is achieved by maximum likelihood.

Poisson Regression with R

glm function output

- The information on deviance is also provided.
- We can use the residual deviance to perform a goodness of fit test for the overall model.
- The residual deviance is the difference between the deviance of the current model and the maximum deviance of the ideal model where the predicted values are identical to the observed.

Poisson Regression with R

glm function output

- Therefore, if the residual difference is small enough, the goodness of fit test will not be significant, indicating that the model fits the data.
- We conclude that the model fits reasonably well because the goodness-of-fit chi-squared test is not statistically significant.

Poisson Regression with R

glm function output

- If the test had been statistically significant, it would indicate that the data do not fit the model well.
- We could try to determine if there are omitted predictor variables, if our linearity assumption holds and/or if there is an issue of over-dispersion.

Poisson Regression with R

```
with(m1, cbind(res.deviance = deviance,
df = df.residual,
        p = pchisq(deviance, df.residual,
            lower.tail=FALSE)))

      res.deviance  df      p
 [1,]        189.4 196 0.6182
```

Poisson Regression with R

- We can also test the overall effect of prog by comparing the deviance of the full model with the deviance of the model excluding prog.
- The two degree-of-freedom chi-square test indicates that prog, taken together, is a statistically significant predictor of **num_awards**.

Poisson Regression with R

Comparing Models

```
# update m1 model dropping prog
m2 <- update(m1, . ~ . - prog)

# test model differences with chi square test
anova(m2, m1, test="Chisq")
```

Poisson Regression with R

```
Analysis of Deviance Table

Model 1: num_awards ~ math
Model 2: num_awards ~ prog + math
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1       198        204
2       196        189  2     14.6  0.00069 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
```

Incident Rate Ratios

- Sometimes, we might want to present the regression results as **incident rate ratios** (IRRs) and their standard errors, together with the confidence interval.
- To compute the standard error for the incident rate ratios, we will use the **Delta method** (Numerical Computation Method).
- To this end, we make use the function `deltamethod` implemented in R package **msm**.

Incident Rate Ratios

A rate ratio (sometimes called an incidence density ratio) in epidemiology, is a relative difference measure used to compare the incidence rates of events occurring at any given point in time. A common application for this measure in analytic epidemiologic studies is in the search for a causal association between a certain risk factor and an outcome.

$$\text{Incidence Rate Ratio} = \frac{\text{Incidence Rate 1}}{\text{Incidence Rate 2}}$$

Incident Rate Ratios

Incidence rate is the occurrence of an event over person-time, for example person-years.

$$\text{Incidence Rate} = \frac{\text{events}}{\text{Person Time}}$$

Note: the same time intervals must be used for both incidence rates.

Poisson Regression with R

```
s <- deltamethod(list(~ exp(x1), ~ exp(x2), ~ exp(x3), ~

#exponentiate old estimates dropping the p values

rexp.est <- exp(r.est[, -3])

# replace SEs with estimates for exponentiated coefficien

rexp.est[, "Robust SE"] <- s
```

Poisson Regression with R

```
rexp.est

                Estimate Robust SE       LL      UL
 (Intercept)    0.005263   0.00340 0.001484 0.01867
 progAcademic   2.956065   0.94904 1.575551 5.54620
 progVocational 1.447458   0.57959 0.660335 3.17284
 math           1.072672   0.01119 1.050955 1.09484
```

Poisson Regression with R

- The output above indicates that the incident rate for prog = "Academic" is 2.96 times the incident rate for the reference group (prog = "General").
- Likewise, the incident rate for prog = "Vocational" is 1.45 times the incident rate for the reference group holding the other variables at constant.

Poisson Regression with R

- The percent change in the incident rate of num_awards is by 7% for every unit increase in math.

Poisson Regression with R

- Sometimes, we might want to look at the expected marginal means.
- For example, what are the expected counts for each program type holding math score at its overall mean?
- To answer this question, we can make use of the predict function.
- First off, we will make a small data set to apply the predict function to it.

Poisson Regression with R

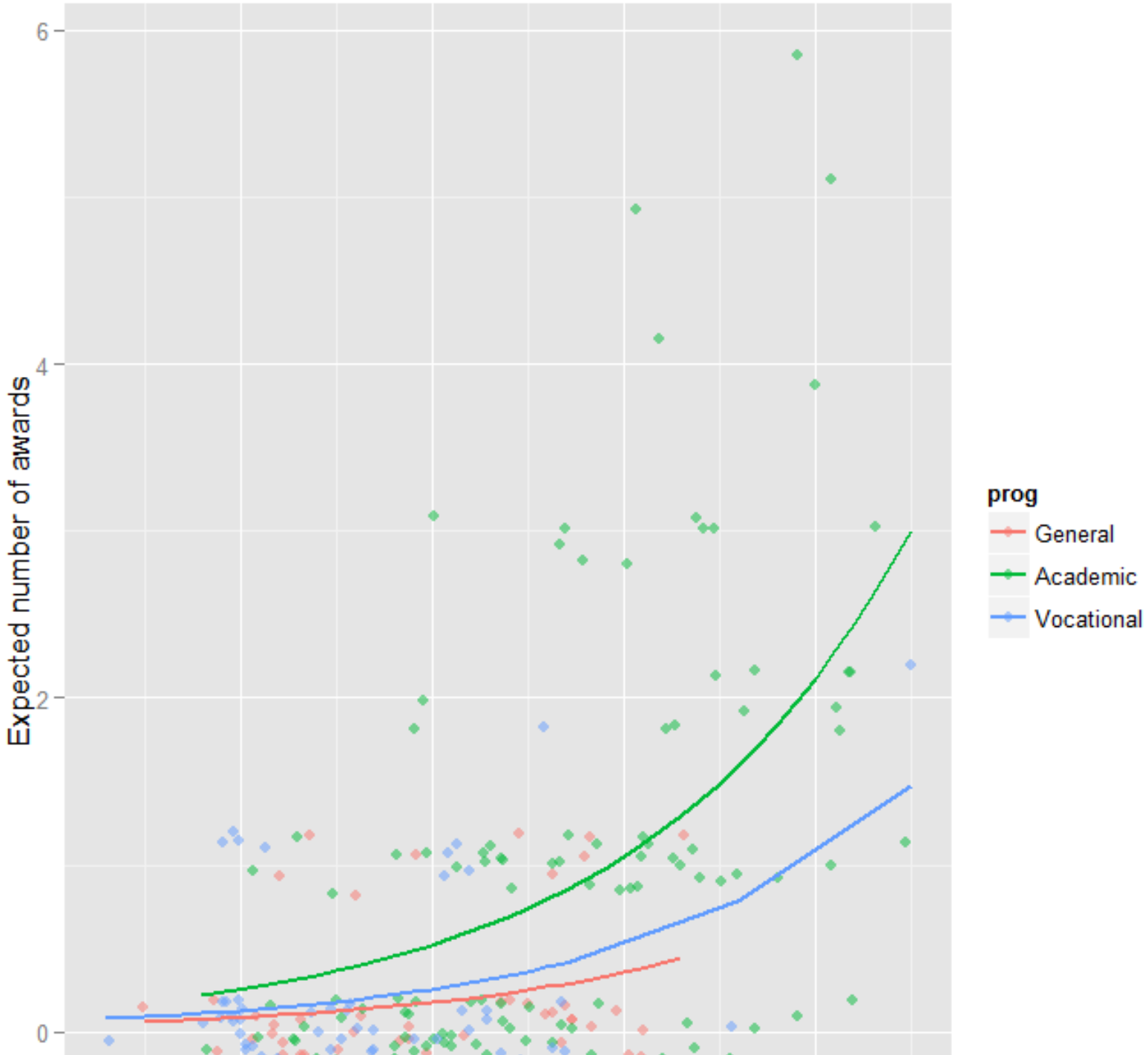
```
(s1 <- data.frame(math = mean(p$math),
  prog = factor(1:3, levels = 1:3,
  labels = levels(p$prog))))

    math        prog
1 52.65     General
2 52.65    Academic
3 52.65 Vocational
```

Poisson Regression with R

```
predict(m1, s1, type="response", se.fit=TRUE)

 $fit
      1       2       3
 0.2114 0.6249 0.3060

 $se.fit
       1        2        3
 0.07050 0.08628 0.08834

 $residual.scale
 [1] 1
```

Poisson Regression with R

- In the output above, we see that the predicted number of events for level 1 of prog is about 0.21, holding math at its mean.
- The predicted number of events for level 2 of prog is higher at 0.62, and the predicted number of events for level 3 of prog is about .31.
- The ratios of these predicted counts ($\frac{0.625}{0.211} = 2.96$, $\frac{0.306}{0.211} = 1.45$) match what we saw looking at the IRR.


Expected number of awards
6
4
2
0
prog
General
Academic
Vocational

Poisson Regression with R

- We can also graph the predicted number of events with the commands below.
- The graph indicates that the most awards are predicted for those in the academic program (prog $= 2$), especially if the student has a high math score.
- The lowest number of predicted awards is for those students in the general program (prog $=$ 1).
- The graph overlays the lines of expected values onto the actual points, although a small amount of random noise was added vertically to lessen overplotting.

Poisson Regression with R

```
# Calculate and store predicted values
p$phat <- predict(m1, type="response")

# order by program and then by math
p <- p[with(p, order(prog, math)), ]
```

Poisson Regression with R

```
ggplot(p, aes(x = math, y = phat, colour = prog)) +
  geom_point(aes(y = num_awards), alpha=.5, position=
  geom_line(size = 1) +
  labs(x = "Math Score", y = "Expected number of awar
```

Over-Dispersion

- Overdispersion is the presence of greater variability in a data set than would be expected based on a given simple statistical model.
- Poisson Distribution:

$$\mathrm{Var}(X) > \mathrm{E}(X)$$

Zero-Inflation

- One common cause of over-dispersion is excess zeros, which in turn are generated by an additional data generating process.
- In this situation, zero-inflated model should be considered.
- If the data generating process does not allow for any 0s (such as the number of days spent in the hospital), then a zero-truncated model may be more appropriate.

Over-Dispersion

- When there seems to be an issue of dispersion, we should first check if our model is appropriately specified, such as omitted variables and functional forms.
- For example, if we omitted the predictor variable prog in the example above, our model would seem to have a problem with over-dispersion.
- In other words, a misspecified model could present a symptom like an over-dispersion problem.

Poisson Regression with R

- Assuming that the model is correctly specified, the assumption that the conditional variance is equal to the conditional mean should be checked.
- There are several tests including the likelihood ratio test of over-dispersion parameter alpha by running the same model using negative binomial distribution.
- The R package **pscl** (Political Science Computational Laboratory, Stanford University) provides many functions for binomial and count data including `odTest` for testing over-dispersion.

Poisson Regression with R

- Count data often have an exposure variable, which indicates the number of times the event could have happened.
- This variable should be incorporated into a Poisson model with the use of the offset option.
- The outcome variable in a Poisson regression cannot have negative numbers, and the exposure cannot have 0s.

Poisson Regression with R

- Many different measures of pseudo-R-squared exist. They all attempt to provide information similar to that provided by R-squared in OLS regression, even though none of them can be interpreted exactly as R-squared in OLS regression is interpreted.
- Poisson regression is estimated via maximum likelihood estimation. It usually requires a large sample size.

Poisson Regression : Crabs Example

The crabs data set The crabs data set is derived from Agresti *(2007, Table 3.2, pp.76-77)*. It gives 4 variables for each of 173 female horseshoe crabs.

- **Satellites** number of male partners in addition to the female's primary partner
- **Width** width of the female in centimeters
- **Dark** a binary factor indicating whether the female has dark coloring (yes or no)
- **GoodSpine** a binary factor indicating whether the female has good spine condition (yes or no)

Poisson Regression : Crabs Example

```
require(glm2)
data(crabs)
head(crabs)
summary(crabs[,1:4])
```

Poisson Regression : Crabs Example

```
> head(crabs)
Satellites Width Dark GoodSpine Rep1 Rep2
1          8  28.3   no         no    2    2
2          0  22.5  yes         no    4    5
3          9  26.0   no        yes    5    6
4          0  24.8  yes         no    6    6
5          4  26.0  yes         no    6    8
....
```

Poisson Regression : Crabs Example

```
> summary(crabs[,1:4])
Satellites          Width        Dark     GoodSpine
Min.   : 0.000   Min.   :21.0   no :107   no :121
1st Qu.: 0.000   1st Qu.:24.9   yes: 66   yes: 52
Median : 2.000   Median :26.1
Mean   : 2.919   Mean   :26.3
3rd Qu.: 5.000   3rd Qu.:27.7
Max.   :15.000   Max.   :33.5
```

Poisson Regression : Crabs Example

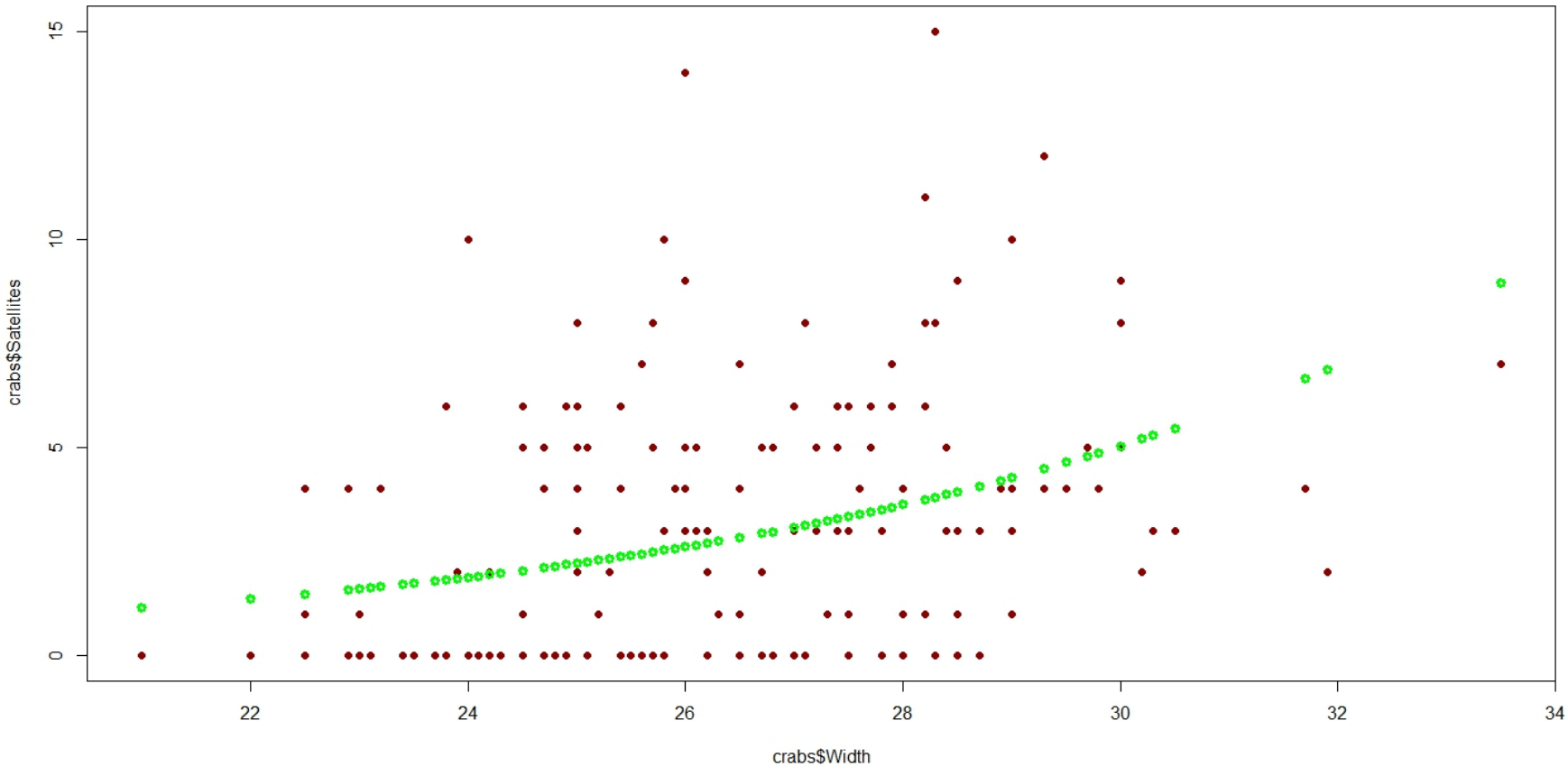
Fit a Poisson regression model with the number of Satellites as the outcome and the width of the female as the covariate. What is the multiplicative change in the expected number of crabs for each additional centimeter of width?

```
crabs.pois <- glm2(Satellites ~ Width,
data=crabs, family="poisson")
summary(crabs.pois)
exp(0.164)
```

Poisson Regression : Crabs Example

```
> summary(crabs.pois)

Call:
glm2(formula = Satellites ~ Width,
family = "poisson", data = crabs)

......
......
Coefficients:
Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.30476    0.54224  -6.095  1.1e-09 ***
Width        0.16405    0.01997   8.216  < 2e-16 ***
---
......
```

Poisson Regression : Crabs Example



Poisson Regression : Crabs Example

```
plot(crabs$Width, crabs$Satellites,
pch=16, col="darkred")
points(crabs$Width, crabs.pois$fitted.values,
col="green", lwd=3)
```

Introduction

Negative binomial regression is for modeling count variables, usually for over-dispersed count outcome variables.

Negative Binomial regression with R

- Negative binomial regression can be used for over-dispersed count data, that is when the conditional variance exceeds the conditional mean.
- It can be considered as a generalization of Poisson regression since it has the same mean structure as Poisson regression and it has an extra parameter to model the over-dispersion.

Negative Binomial regression with R

- If the conditional distribution of the outcome variable is over-dispersed, the confidence intervals for Negative binomial regression are likely to be narrower as compared to those from a Poisson regression.

Examples of negative binomial regression

- **Example 1** School administrators study the attendance behavior of high school juniors at two schools.
  Predictors of the number of days of absence include the type of program in which the student is enrolled and a standardized test in math.
- **Example 2** A health-related researcher is studying the number of hospital visits in past 12 months by senior citizens in a community based on the characteristics of the individuals and the types of health plans under which each one is covered.

Negative Binomial Regression with R

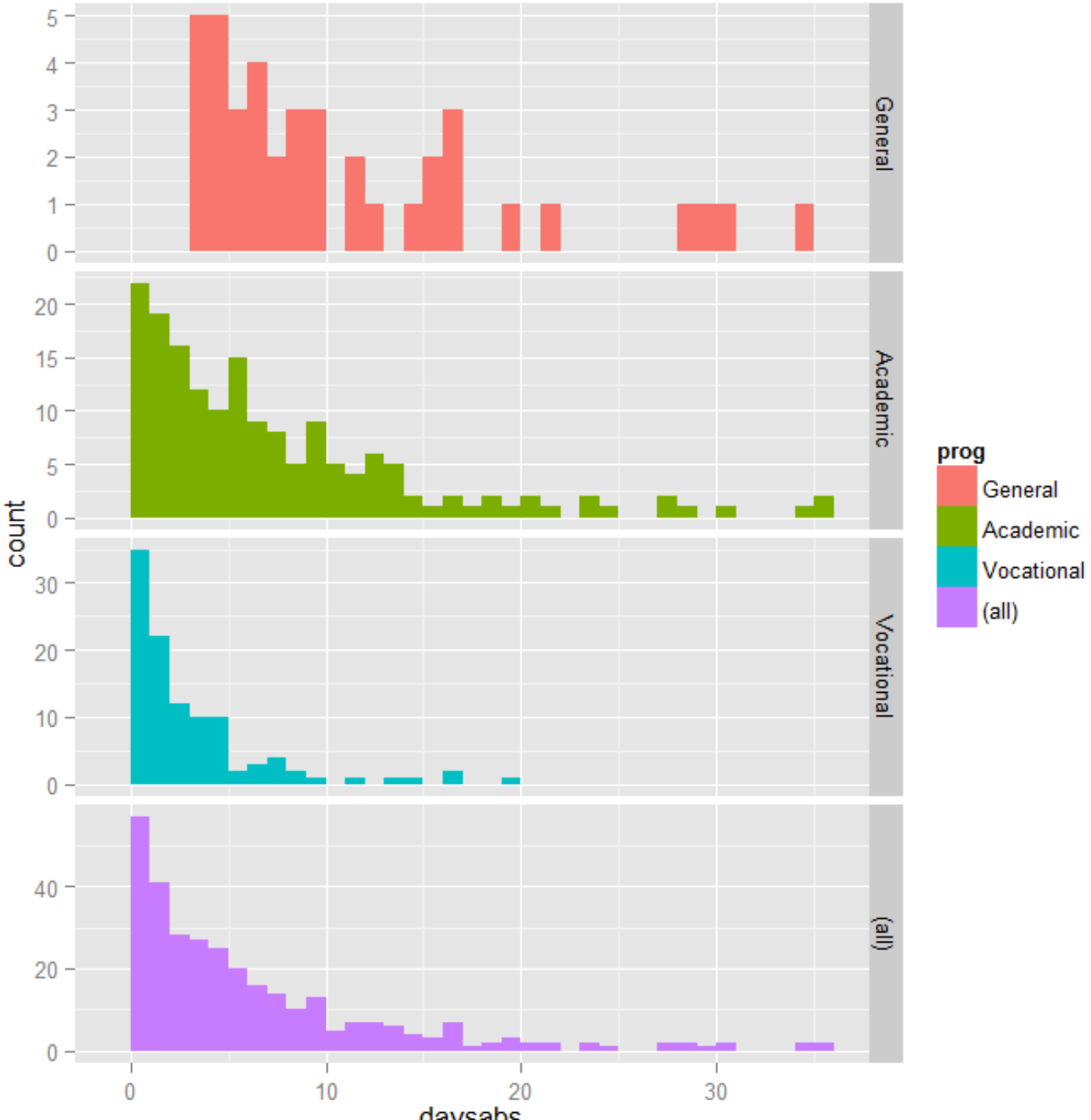
Description of the data Let's pursue Example 1 from above.

- We have attendance data on 314 high school juniors from two urban high schools in the file **negbin.csv** .
- The response variable of interest is days absent, **daysabs**.
- The variable **math** gives the standardized math score for each student.
- The variable **prog** is a three-level nominal variable indicating the type of instructional program in which the student is enrolled.

Negative Binomial Regression with R

Exploratory Data Analysis

```
summary(dat)
       id          gender       math          daysabs
  1001   :  1   female:160   Min.   : 1.0   Min.   : 0.00
  1002   :  1   male  :154   1st Qu.:28.0   1st Qu.: 1.00
  1003   :  1                Median :48.0   Median : 4.00
  1004   :  1                Mean   :48.3   Mean   : 5.96
  1005   :  1                3rd Qu.:70.0   3rd Qu.: 8.00
  1006   :  1                Max.   :99.0   Max.   :35.00
  (Other):308
        prog
  General   : 40
  Academic  :167
  Vocational:107
```


General
Academic
Vocational
(all)
count
daysabs
0
10
20
30
prog
General
Academic
Vocational
(all)

Negative Binomial Regression with R

Each variable has 314 valid observations and their distributions seem quite reasonable. The unconditional mean of our outcome variable is much lower than its variance.

Data Set

- The table below shows the average numbers of days absent by program type and seems to suggest that program type is a good candidate for predicting the number of days absent, our outcome variable, because the mean value of the outcome appears to vary by **prog**.

Data Set

- The variances within each level of prog are higher than the means within some of the levels.
- These are the conditional means and variances. These differences suggest that over-dispersion is present and that a Negative Binomial model would be appropriate.

Negative Binomial Regression with R

```
with(dat, tapply(daysabs, prog, function(x) {
sprintf("M (SD) = %1.2f (%1.2f)", mean(x), sd(x))
}))
##                   General
## "M (SD) = 10.65 (8.20)"
##                  Academic
## "M (SD) = 6.93 (7.45)"
##                Vocational
## "M (SD) = 2.67 (3.73)"
```

Negative Binomial Regression with R

Negative binomial regression analysis

We will use the `glm.nb` function from the MASS package to estimate a negative binomial regression.

```
summary(m1 <- glm.nb(daysabs ~ math + prog,
 data = negbinom))
##
## Call:
## glm.nb(formula = daysabs ~ math + prog,
##         data = dat, init.theta = 1.032713156,
##      link = log)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.155  -1.019  -0.369   0.229   2.527
```

Negative Binomial Regression with R

- R first displays the call and the deviance residuals.
- Next, we see the regression coefficients for each of the variables, along with standard errors, z-scores, and p-values.

Negative Binomial Regression with R

```
Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)      2.61527    0.19746   13.24  < 2e-16 ***
math            -0.00599    0.00251   -2.39    0.017 *
progAcademic    -0.44076    0.18261   -2.41    0.016 *
progVocational -1.27865    0.20072   -6.37  1.9e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
```

Negative Binomial Regression with R

- The variable **math** has a coefficient of -0.006, which is statistically significant.
- This means that for each one-unit increase in **math**, the expected log count of the number of days absent decreases by 0.006.
- The indicator variable shown as **progAcademic** is the expected difference in log count between group 2 and the reference group (prog=1).

Negative Binomial Regression with R

```
## (Dispersion parameter for Negative Binomial(1.033) famil
##
##     Null deviance: 427.54  on 313  degrees of freedom
## Residual deviance: 358.52  on 310  degrees of freedom
## AIC: 1741
##
## Number of Fisher Scoring iterations: 1
##
##
##                Theta:  1.033
##            Std. Err.:  0.106
##
##  2 x log-likelihood:  -1731.258
```

Negative Binomial Regression with R

- The expected log count for level 2 of prog is 0.44 lower than the expected log count for level 1.
- The indicator variable for progVocational is the expected difference in log count between group 3 and the reference group.

Negative Binomial Regression with R

- The expected log count for level 3 of prog is 1.28 lower than the expected log count for level 1.
- To determine if prog itself, overall, is statistically significant, we can compare a model with and without prog.
- The reason it is important to fit separate models, is that unless we do, the overdispersion parameter is held constant.

Negative Binomial Regression with R

```
m2 <- update(m1, . ~ . - prog)
anova(m1, m2)
## Likelihood ratio tests of Negative Binomial Models
##
## Response: daysabs
##          Model  theta Resid. df    2 x log-lik.   Test
## 1         math 0.8559         312           -1776
## 2 math + prog 1.0327         310           -1731 1 vs 2
##     Pr(Chi)
## 1
## 2 1.652e-10
```

Negative Binomial Regression with R

- The two degree-of-freedom chi-square test indicates that prog is a statistically significant predictor of daysabs.
- The null deviance is calculated from an intercept-only model with 313 degrees of freedom.
- Then we see the residual deviance, the deviance from the full model.
- We are also shown the AIC and 2*log likelihood.

Negative Binomial Regression with R

- The theta parameter shown is the dispersion parameter.
- Note that R parameterizes this differently from SAS, Stata, and SPSS.
- The R parameter (theta) is equal to the inverse of the dispersion parameter (alpha) estimated in these other software packages.
- Thus, the theta value of 1.033 seen here is equivalent to the 0.968 value seen in the Stata Negative Binomial Data Analysis Example because $1/0.968 = 1.033$.

Checking model assumption

- As we mentioned earlier, negative binomial models assume the conditional means are not equal to the conditional variances.
- This inequality is captured by estimating a dispersion parameter (not shown in the output) that is held constant in a Poisson model.
- Thus, the Poisson model is actually nested in the negative binomial model.
- We can then use a likelihood ratio test to compare these two and test this model assumption.
- To do this, we will run our model as a Poisson.

Negative Binomial Regression with R

```
m3 <- glm(daysabs ~ math + prog,
          family = "poisson", data = dat)
pchisq(2 * (logLik(m1) - logLik(m3)),
          df = 1, lower.tail = FALSE)
## 'log Lik.' 2.157e-203 (df=5)
```

Negative Binomial Regression with R

- In this example the associated chi-squared value is 926.03 with one degree of freedom.
- This strongly suggests the negative binomial model, estimating the dispersion parameter, is more appropriate than the Poisson model.

Negative Binomial Regression with R

We can get the confidence intervals for the coefficients by profiling the likelihood function.

```
(est <- cbind(Estimate = coef(m1), confint(m1)))
## Waiting for profiling to be done...
##                  Estimate   2.5 %    97.5 %
## (Intercept)      2.615265  2.2421  3.012936
## math            -0.005993 -0.0109 -0.001067
## progAcademic    -0.440760 -0.8101 -0.092643
## progVocational -1.278651 -1.6835 -0.890078
```

Incidence Rate Ratios

- We might be interested in looking at incident rate ratios rather than coefficients.
- To do this, we can exponentiate our model coefficients. The same applies to the confidence intervals.

Negative Binomial Regression with R

```
exp(est)
##                  Estimate  2.5 %  97.5 %
## (Intercept)       13.6708 9.4127 20.3470
## math               0.9940 0.9892  0.9989
## progAcademic       0.6435 0.4448  0.9115
## progVocational     0.2784 0.1857  0.4106
```

Negative Binomial Regression with R

- The output above indicates that the incident rate for prog = 2 is 0.64 times the incident rate for the reference group (prog = 1).
- Likewise, the incident rate for prog = 3 is 0.28 times the incident rate for the reference group holding the other variables constant.
- The percent change in the incident rate of **daysabs** is a 1% decrease for every unit increase in math.

Negative Binomial Regression with R

- The form of the model equation for negative binomial regression is the same as that for Poisson regression.
- The log of the outcome is predicted with a linear combination of the predictors:

$$ln(\widehat{daysabs_i}) = Intercept + b_1(prog_i = 2) + b_2(prog_i = 3) + b_3 math_i$$

Negative Binomial Regression with R

$$\widehat{daysabs}_i = e^{Intercept + b_1(prog_i=2) + b_2(prog_i=3) + b_3 math_i}$$

$$= e^{Intercept} e^{b_1(prog_i=2)} e^{b_2(prog_i=3)} e^{b_3 math_i}$$

The coefficients have an additive effect in the $ln(y)$ scale and the IRR have a multiplicative effect in the y scale. The dispersion parameter in negative binomial regression does not effect the expected counts, but it does effect the estimated variance of the expected counts.

Predicted values

- For assistance in further understanding the model, we can look at predicted counts for various levels of our predictors.
- Below we create new datasets with values of math and prog and then use the predict command to calculate the predicted number of events.

Negative Binomial Regression with R

- First, we can look at predicted counts for each value of prog while holding math at its mean.
- To do this, we create a new dataset with the combinations of prog and math for which we would like to find predicted values, then use the `predict` command.

Negative Binomial Regression with R

```
newdata1 <- data.frame(math = mean(dat$math),
prog = factor(1:3, levels = 1:3,
labels = levels(dat$prog)))

newdata1$phat <- predict(m1, newdata1,
 type = "response")

newdata1
    math        prog   phat
 1 48.27     General 10.237
 2 48.27    Academic  6.588
 3 48.27 Vocational  2.850
```

Negative Binomial Regression with R

- In the output above, we see that the predicted number of events (e.g., days absent) for a general program is about 10.24, holding math at its mean.
- The predicted number of events for an academic program is lower at 6.59, and the predicted number of events for a vocational program is about 2.85.

Negative Binomial Regression with R

Below we will obtain the mean predicted number of events for values of math across its entire range for each level of prog and graph these.

Negative Binomial Regression with R

```
newdata2 <- data.frame(
math = rep(seq(from = min(dat$math),
       to = max(dat$math), length.out = 100), 3),
prog = factor(rep(1:3, each = 100), levels = 1:3, lab
levels(dat$prog)))
```

Negative Binomial Regression with R

```
newdata2 <- cbind(newdata2, predict(m1, newdata2, type
newdata2 <- within(newdata2, {
DaysAbsent <- exp(fit)
LL <- exp(fit - 1.96 * se.fit)
UL <- exp(fit + 1.96 * se.fit)
})
```

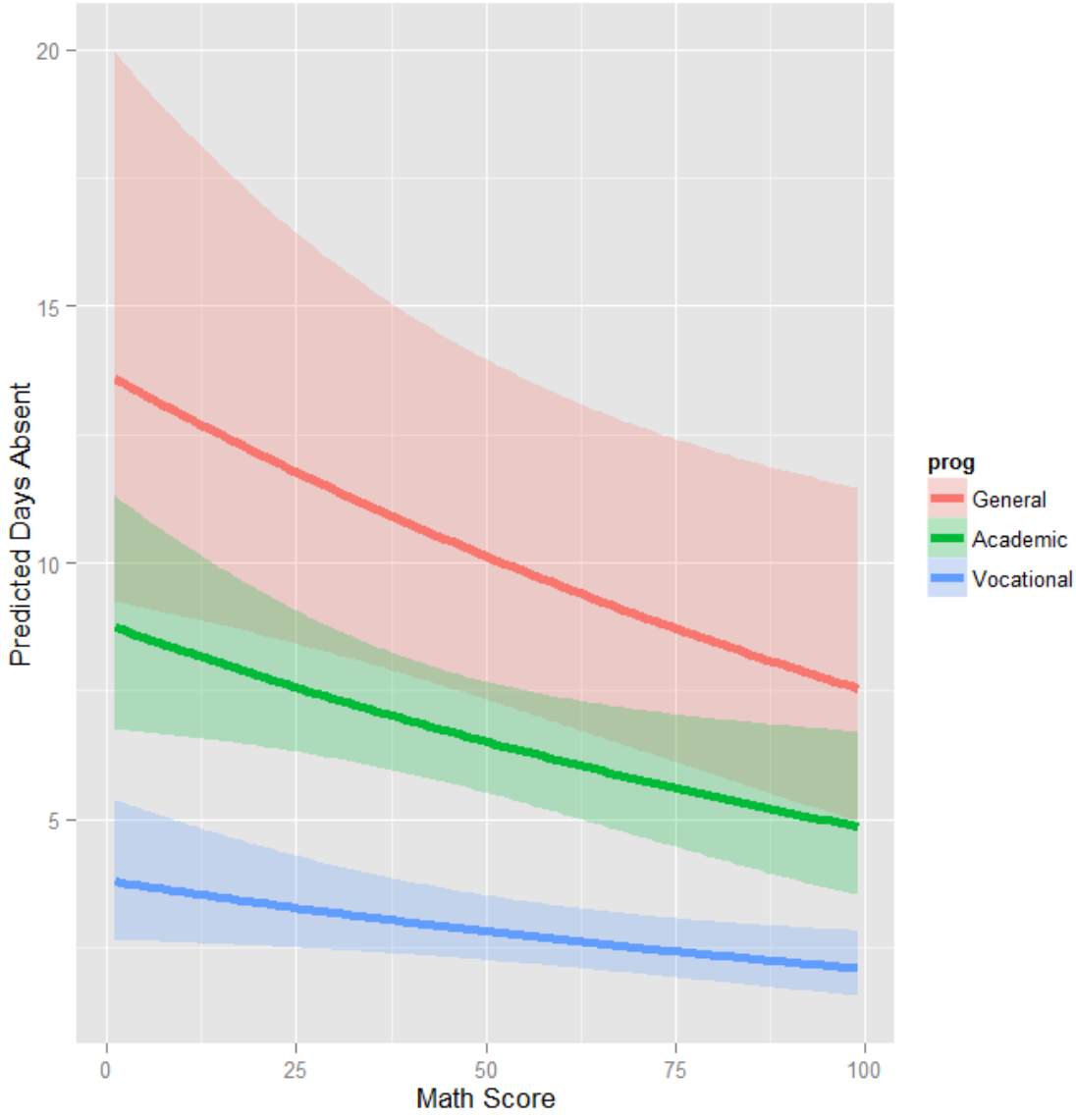

20
15
10
5
Predicted Days Absent
0
25
50
75
100
Math Score
prog
General
Academic
Vocational

Figure:

Negative Binomial Regression with R

- The graph shows the expected count across the range of math scores, for each type of program along with 95 percent confidence intervals.
- Note that the lines are not straight because this is a log linear model, and what is plotted are the expected values, not the log of the expected values.

Things to consider

- It is not recommended that negative binomial models be applied to small samples.
- One common cause of over-dispersion is excess zeros by an additional data generating process. In this situation, **zero-inflated model** should be considered.
- If the data generating process does not allow for any 0s (such as the number of days spent in the hospital), then a **zero-truncated model** may be more appropriate.

Negative Binomial Regression with R

- Count data often have an exposure variable, which indicates the number of times the event could have happened.
- This variable should be incorporated into your negative binomial regression model with the use of the `offset` option.

Negative Binomial Regression with R

- The outcome variable in a negative binomial regression cannot have negative numbers.
- You will need to use the `m1$resid` command to obtain the residuals from our model to check other assumptions of the negative binomial model

Zero-inflated Regression models - Summary

- Zero-inflated models attempt to account for excess zeros.
- In other words, two kinds of zeros are thought to exist in the data, "**true zeros**" and "**excess zeros**".

Zero-inflated Regression models

Two Distinct Processes

- The two parts of the a zero-inflated model are a binary model, usually a logit model to model which of the two processes the zero outcome is associated with and a count model, in this case, a negative binomial model, to model the count process.
- In other words, the excess zeros are generated by a separate process from the count values and that the excess zeros can be modelled independently.
- Zero-inflated models estimate two equations simultaneously, one for the count model and one for the excess zeros.
- The expected count is expressed as a combination of the two processes.

Zero-inflated Regression models

Fishing Data Set

- We have data on 250 groups that went to a park.
- Each group was questioned about how many fish they caught (**count**), how many children were in the group (**child**), how many people were in the group (**persons**), and whether or not they brought a camper to the park (**camper**).
- In addition to predicting the number of fish caught, there is interest in predicting the existence of excess zeros, i.e., the probability that a group caught zero fish.
- We will use the variables child, persons, and camper in our model.

Fishing Data Set

- In addition to predicting the number of fish caught, there is interest in predicting the existence of excess zeros, i.e., the probability that a group caught zero fish.
- We will use the variables child, persons, and camper in our model.

```
> head(fish)
nofish livebait camper persons child          xb
1      1        0      0       1     0 -0.8963146
2      0        1      1       1     0 -0.5583450
3      0        1      0       1     0 -0.4017310
4      0        1      1       2     1 -0.9562981
5      0        1      0       1     0  0.4368910
6      0        1      1       4     2  1.3944855
zg count
1  3.0504048     0
2  1.7461489     0
3  0.2799389     0
4 -0.6015257     0
5  0.5277091     1
6 -0.7075348     0
```

What is a Zero-Inflated Model?

The Fishing Example

- A zero-inflated model assumes that zero outcome is due to two different processes.
- For instance, in the example of fishing presented here, the two processes are that a subject has *gone fishing* vs. *not gone fishing*.
- If not gone fishing, the only outcome possible is zero.
- If gone fishing, it is then a count process.

$$E(nfishcaught = k) = P(notgonefishing) \times 0 + P(gonefishing) \times E(y = k|g$$

Zero-inflated Poisson regression

Though we can run a Poisson regression in `R` using the `glm` function in one of the core packages, we need another package to run the zero-inflated poisson model. We use the **pscl** package.

```
summary(m1 <- zeroinfl(count ~ child + camper |
      persons, data = zinb))
```

Zero-inflated Poisson regression

```
##
## Call:
## zeroinfl(formula = count ~ child + camper | persons, dat
##
## Pearson residuals:
##     Min      1Q  Median      3Q     Max
## -1.237 -0.754 -0.608 -0.192 24.085
```

Zero-inflated Poisson regression

```
## Count model coefficients (poisson with log link):
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.5979     0.0855   18.68   <2e-16 ***
## child        -1.0428     0.1000  -10.43   <2e-16 ***
## camper1       0.8340     0.0936    8.91   <2e-16 ***
```

Zero-inflated Poisson regression

```
## Zero-inflation model coefficients (binomial with logit l
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)    1.297      0.374    3.47  0.00052 ***
## persons       -0.564      0.163   -3.46  0.00053 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
##
## Number of iterations in BFGS optimization: 12
## Log-likelihood: -1.03e+03 on 5 Df
```

Zero-inflated Poisson regression

- Below the model call, you will find a block of output containing Poisson regression coefficients for each of the variables along with standard errors, z-scores, and p-values for the coefficients.
- A second block follows that corresponds to the inflation model.
- This includes logit coefficients for predicting excess zeros along with their standard errors, z-scores, and p-values.

- All of the predictors in both the count and inflation portions of the model are statistically significant.

Vuong Testing

- Note that the model output above does not indicate in any way if our zero-inflated model is an improvement over a standard Poisson regression.
- We can determine this by running the corresponding standard Poisson model and then performing a Vuong test of the two models.

```
summary(p1 <- glm(count ~ child + camper,
family = poisson, data = fishing))
```

- The Vuong test compares the zero-inflated model with an ordinary Poisson regression model.
- In this example, we can see that our test statistic is significant, indicating that the zero-inflated model is superior to the standard Poisson model.

```
vuong(p1, m1)
## Vuong Non-Nested Hypothesis Test-Statistic: -3.574
## (test-statistic is asymptotically distributed N(0,1)
##  null that the models are indistinguishible)
## in this case:
## model2 > model1, with p-value 0.0001756
```

Zero-Inflated Negative Binomial regression

- We are going to use the variables: **child** and **camper** to model the count in the part of negative binomial model and the variable **persons** in the logit part of the model.
- We use the **pscl** to run a zero-inflated negative binomial regression.
- We begin by estimating the model (called `m1`) with the variables of interest.

```
m1 <- zeroinfl(count ~ child + camper | persons,
  data = fishing, dist = "negbin",
  EM = TRUE)

summary(m1)
```

```
## Call:
## zeroinfl(formula = count ~ child + camper | persons,
##     data = fishing,
##     dist = "negbin", EM = TRUE)
##
## Pearson residuals:
##    Min     1Q Median     3Q    Max
## -0.586 -0.462 -0.389 -0.197 18.013
```

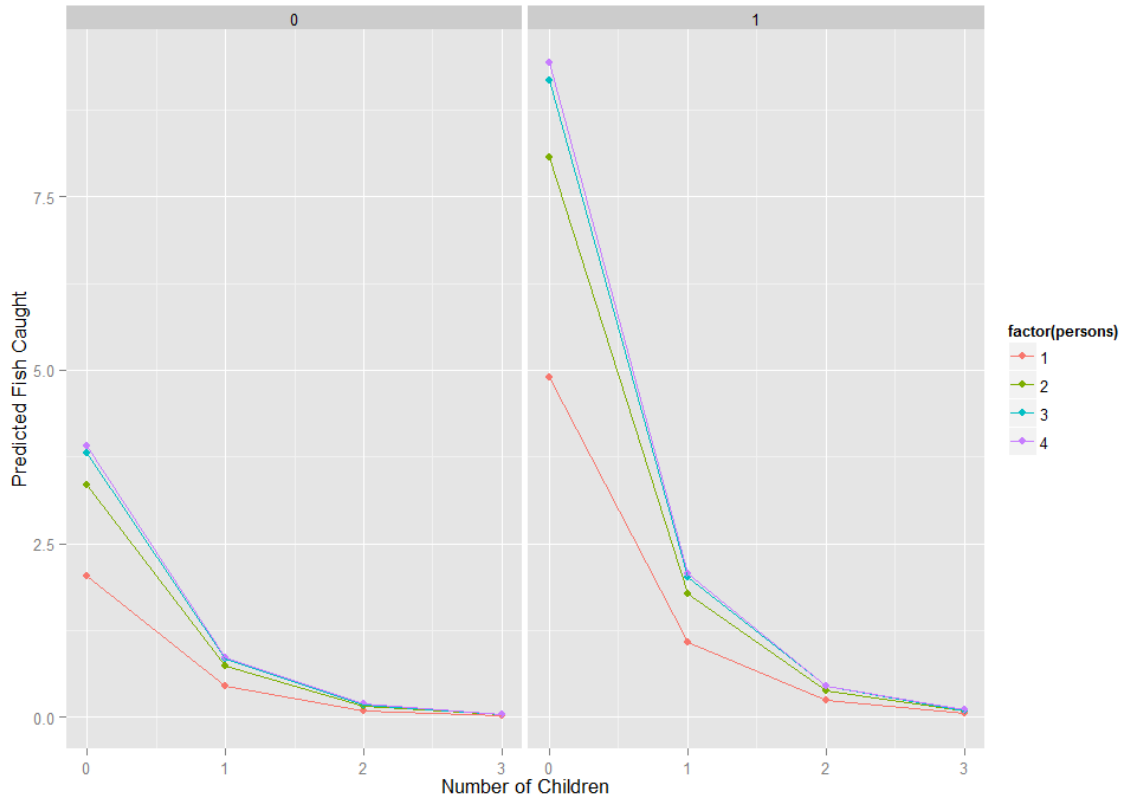
- Below the model call, you will find a block of output containing negative binomial regression coefficients for each of the variables along with standard errors, z-scores, and p-values for the coefficients.
- A second block follows that corresponds to the inflation model. This includes logit coefficients for predicting excess zeros along with their standard errors, z-scores, and p-values.

```
## Count model coefficients (negbin with log link):
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)     1.371      0.256    5.35  8.6e-08 ***
## child          -1.515      0.196   -7.75  9.4e-15 ***
## camper1         0.879      0.269    3.26   0.0011 **
## Log(theta)     -0.985      0.176   -5.60  2.1e-08 ***
```

```
## Zero-inflation model coefficients (binomial with logit l
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)    1.603      0.836    1.92    0.055 .
## persons       -1.666      0.679   -2.45    0.014 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
##
## Theta = 0.373
## Number of iterations in BFGS optimization: 2
## Log-likelihood: -433 on 6 Df
```

Tests of Significance

- All of the predictors in both the count and inflation portions of the model are statistically significant.
- This model will fit the data significantly better than the null model, i.e., the intercept-only model.
- To show that this is the case, we could compare with the current model to a null model without predictors using chi-squared test on the difference of log likelihoods.


0
1
7.5
5.0
2.5
0.0
Predicted Fish Caught
0
1
2
3
Number of Children
factor(persons)
1
2
3
4

- Note that the model output above does not indicate in any way if our zero-inflated model is an improvement over a standard negative binomial regression.
- We can determine this by running the corresponding standard negative binomial model and then performing a Vuong test of the two models.
- We use the MASS package to run the standard negative binomial regression.

```
library(MASS)
summary(m2 <- glm.nb(count ~ child + camper, data = zinb))
.....
```

```
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)    1.073      0.242    4.42  9.7e-06 ***
## child         -1.375      0.196   -7.03  2.1e-12 ***
## camper1        0.909      0.284    3.21   0.0013 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
```

```
vuong(m1, m2)

## Vuong Non-Nested Hypothesis Test-Statistic: 1.702
## (test-statistic is asymptotically distributed N(0,1) unc
##  null that the models are indistinguishible)
## in this case:
## model1 > model2, with p-value 0.0444
```

- The predictors child and camper in the part of the negative binomial regression model predicting number of fish caught (count) are both significant predictors.
- The predictor person in the part of the logit model predicting excessive zeros is statistically significant.
- For these data, the expected change in log(count) for a one-unit increase in child is -1.515255 holding other variables constant.
- A camper (camper $= 1$) has an expected log(count) of 0.879051 higher than that of a non-camper (camper $= 0$) holding other variables constant.

- The log odds of being an excessive zero would decrease by 1.67 for every additional person in the group.
- In other words, the more people in the group the less likely that the zero would be due to not gone fishing.
- Put plainly, the larger the group the person was in, the more likely that the person went fishing.
- The Vuong test suggests that the zero-inflated negative binomial model is a significant improvement over a standard negative binomial model.

Zero Truncated Poisson Distribution

Zero-Truncated Poisson Regression

- Zero-truncated Modelling is used to model count data for which the value zero cannot occur.
- Zero Truncated Poisson Model
- Zero Truncated Negative Binomial Model (Over Dispersion)

Examples of Zero-Truncated Model

Example 1.

- A study of length of hospital stay, in days, as a function of age, kind of health insurance and whether or not the patient died while in the hospital.
- Length of hospital stay is recorded as a minimum of at least one day.

Example 2.

- A study of the number of journal articles published by tenured faculty as a function of discipline (fine arts, science, social science, humanities, medical, etc).
- To get tenure faculty must publish, therefore, there are no tenured faculty with zero publications.

Examples of Zero-Truncated Model

Example 3.

- A study by the county traffic court on the number of tickets received by teenagers as predicted by school performance, amount of driver training and gender.
- Only individuals who have received at least one citation are in the traffic court files.

Example 4.

- Consider for example the random variable of the number of items in a shopper's basket at a supermarket checkout line.
- Presumably a shopper does not stand in line with nothing to buy (i.e. the minimum purchase is 1 item), so this phenomenon may follow a ZTP distribution

Zero-Truncated Poisson regression

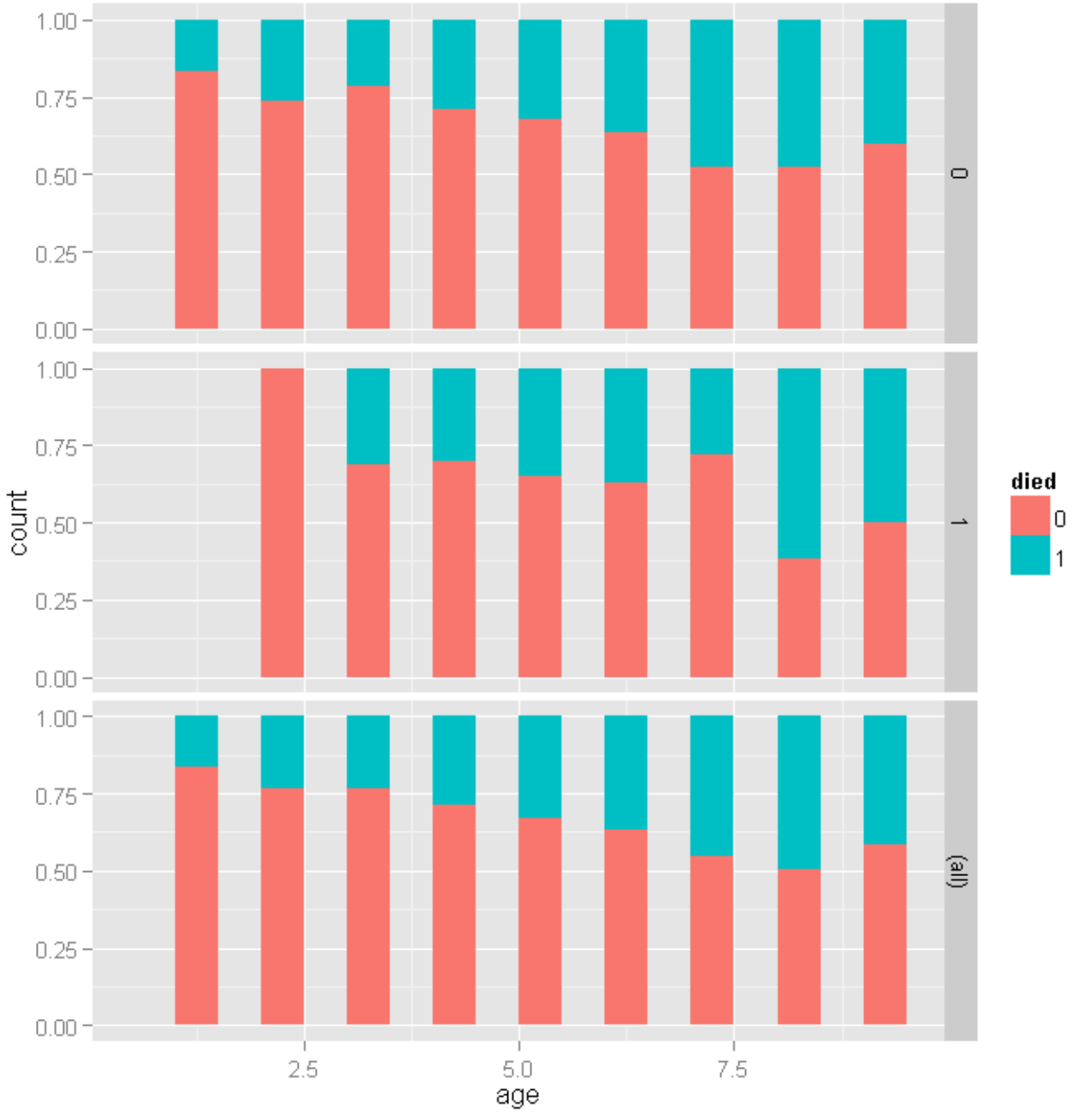
Data Set : hospitalstay

- We have a hypothetical data file, **hospitalstay** with 1,493 observations.
- The length of hospital stay variable is **stay**.
- The variable **age** gives the age group from 1 to 9 which will be treated as interval in this example.
- The variables **hmo** and **died** are binary indicator variables for HMO insured patients and patients who died while in the hospital, respectively.

Zero-Truncated Poisson regression

Data Set : hospitalstay

```
##       stay             age        hmo       died
##  Min.   : 1.00   Min.   :1.00   0:1254   0:981
##  1st Qu.: 4.00   1st Qu.:4.00   1: 239   1:512
##  Median : 8.00   Median :5.00
##  Mean   : 9.73   Mean   :5.23
##  3rd Qu.:13.00   3rd Qu.:6.00
##  Max.   :74.00   Max.   :9.00
```


1.00
0.75
0.50
0.25
0.00
count
0
1
(all)
2.5
5.0
7.5
age
died
0
1

Zero-Truncated Poisson regression

Data Set : hospitalstay

- For the lowest ages, a smaller proportion of people in HMOs died, but for higher ages, there does not seem to be a huge difference, with a slightly higher proportion in HMOs dying if anything.
- Overall, as age group increases, the proportion of those dying increases, as expected.

Zero-Truncated Poisson regression

- To fit the zero-truncated Poisson model, we use the `vglm` function in the VGAM package.
- This function fits a very flexible class of models called **vector generalized linear models** to a wide range of assumed distributions.
- In our case, we believe the data are Poisson, but without zeros.
- Thus the values are strictly positive Poisson, for which we use the positive Poisson family via the `pospoisson` function passed to `vglm`.

Zero-Truncated Poisson regression

Fitting the Model with R

We will use the *hospitalstay* data.

```
m1 <- vglm(stay ~ age + hmo + died,
    family = pospoisson(),
    data = hospitalstay)
summary(m1)
```

Zero-Truncated Poisson regression

Fitting the Model with R

Model Summary

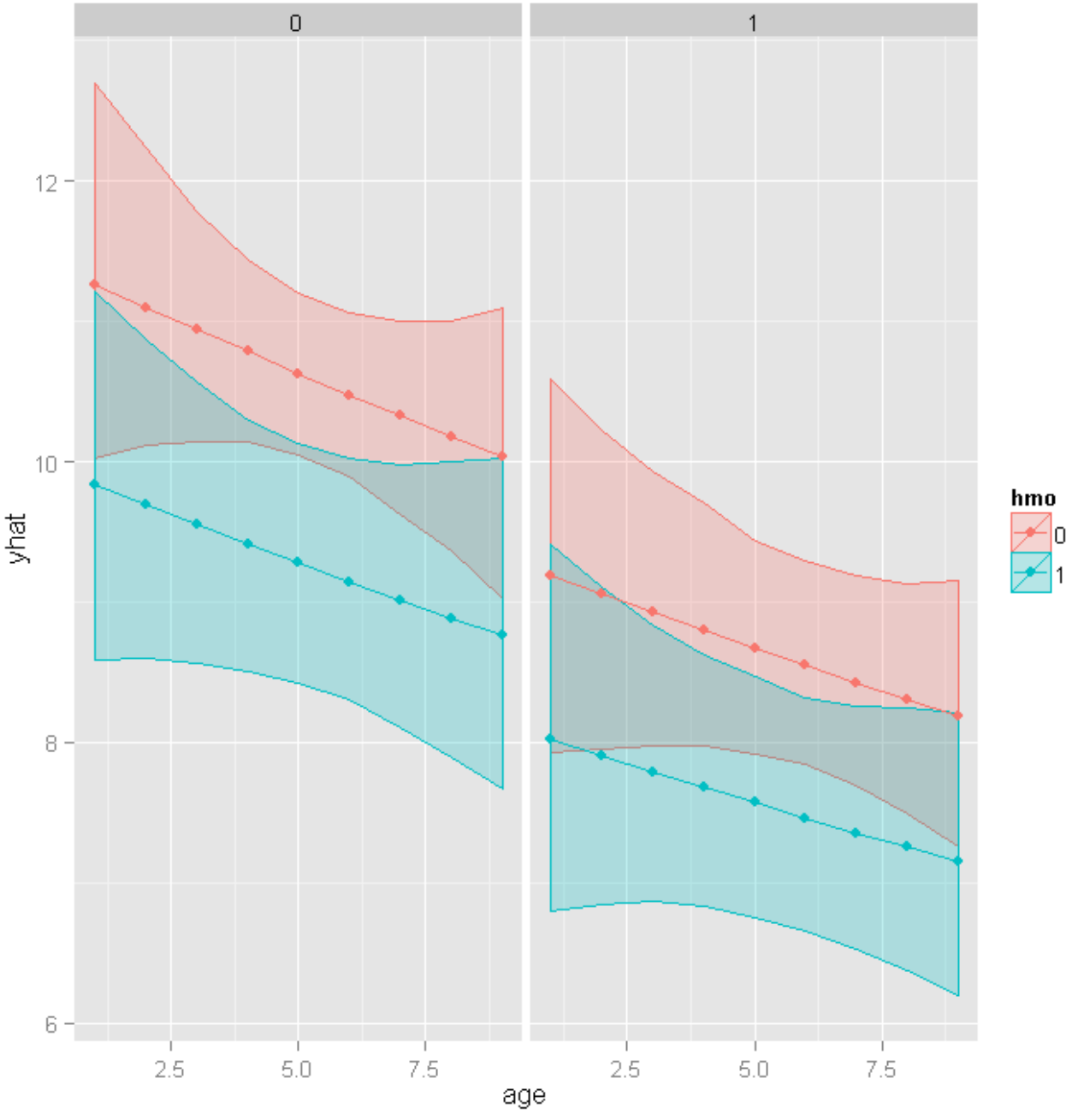
```
## Coefficients:
##             Estimate Std. Error z value
## (Intercept)    2.436      0.027    89.1
## age           -0.014      0.005    -2.9
## hmo1          -0.136      0.024    -5.7
## died1         -0.204      0.018   -11.1
```

Zero-Truncated Poisson regression

- The value of the coefficient for age, -0.0144 suggests that the log count of stay decreases by 0.0144 for each year increase in age.
- The coefficient for hmo, -0.1359 indicates that the log count of stay for HMO patient is 0.1359 less than for non-HMO patients.
- The log count of stay for patients who died while in the hospital was 0.2038 less than those patients who did not die.
- Finally, the value of the constant 2.4358 is the log count of the stay when all of the predictors equal zero.

Zero-Truncated Poisson regression

- Can compute CIs using **boot** package
- Age does not have a significant effect, but hmo and died both do.


0
1
12
10
8
6
yhat
2.5
5.0
7.5
age
hmo
0
1

Zero-truncated negative binomial regression

- Zero-truncated negative binomial regression is used to model count data for which the value zero cannot occur and for which over dispersion exists.

Zero-truncated negative binomial regression

- To fit the zero-truncated negative binomial model, we use the `vglm` function in the VGAM package.
- This function fits a very flexible class of models called vector generalized linear models to a wide range of assumed distributions.
- In our case, we believe the data come from the negative binomial distribution, but without zeros.
- Thus the values are strictly positive poisson, for which we use the positive negative binomial family via the `posnegbinomial` function passed to `vglm`.

Zero-truncated negative binomial regression

Fitting the Model with R

We will use the *hospitalstay* data again.

```
m1 <- vglm(stay ~ age + hmo + died,
   family = posnegbinomial(),
   data = hospitalstay)
```

Zero Truncated Negative Binomial Regression

```
summary(m1)
##
## Call:
## vglm(formula = stay ~ age + hmo + died,
##     family = posnegbinomial(),
##     data = hospitalstay)
##
## Pearson Residuals:
##                 Min     1Q Median    3Q Max
## log(munb)  -1.4 -0.70  -0.23 0.45 9.8
## log(size) -14.1 -0.27   0.45 0.76 1.0
```

Zero Truncated Negative Binomial Regression

```
## Coefficients:
##                 Estimate Std. Error z value
## (Intercept):1      2.408      0.072    33.6
## (Intercept):2      0.569      0.055    10.4
## age               -0.016      0.013    -1.2
## hmo1              -0.147      0.059    -2.5
## died1             -0.218      0.046    -4.7
```

Zero Truncated Negative Binomial Regression

-
- The first intercept is what we know as the typical intercept.
- The second is the **over dispersion parameter**, α.
- The number of linear predictors is two, one for the expected mean λ and one for the over dispersion.
- Next the dispersion parameter is printed, assumed to be one after accounting for overdispersion.

Zero Truncated Negative Binomial Regression

- The value of the coefficient for age, -0.0157 suggests that the log count of stay decreases by 0.0157 for each year increase in age.
- The coefficient for hmo, -0.1471 indicates that the log count of stay for HMO patient is 0.1471 less than for non-HMO patients.
- The log count of stay for patients who died while in the hospital was 0.2178 less than those patients who did not die.

Zero Truncated Negative Binomial Regression

- The value of the constant 2.4083 is the log count of the stay when all of the predictors equal zero.
- The value of the second intercept, the over dispersion parameter, α is 0.5686.
- To test whether we need to estimate over dispersion, we could fit a zero-truncated Poisson model and compare the two. (Not Covered).

Standard Errors for Poisson Regression

- It is recommended using robust standard errors for the parameter estimates to control for mild violation of the distribution assumption that the variance equals the mean.
- The R package **sandwich** can be used to obtain the robust standard errors and calculated the p-values accordingly.
- Together with the p-values, we have also calculated the 95% confidence interval using the parameter estimates and their robust standard errors.

Standard Errors for Poisson Regression

sandwich `R` Package

- Robust Covariance Matrix Estimators
- Model-robust standard error estimators for cross-sectional, time series, and longitudinal data.

Poisson Regression with R

Robust Standard Errors

```
cov.model1 <- vcovHC(model1, type="HC0")
std.err <- sqrt(diag(cov.model1))

r.est <- cbind(Estimate= coef(model1),
"Robust SE" = std.err,
"Pr(>|z|)" = 2 * pnorm(abs(coef(model1)/std.err),
lower.tail=FALSE),
 LL = coef(model1) - 1.96 * std.err,
 UL = coef(model1) + 1.96 * std.err)
```

Poisson Regression with R

Robust Standard Errors

```
r.est

                Estimate Robust SE   Pr(>|z|)      LL        UL
(Intercept)    -5.24712   0.64600 4.567e-16 -6.5133 -3.
progAcademic    1.08386   0.32105 7.355e-04  0.4546  1.
progVocational  0.36981   0.40042 3.557e-01 -0.4150  1.
math            0.07015   0.01044 1.784e-11  0.0497  0.
```

Poisson Regression with R

```
with(p, tapply(num_awards, prog, function(x) {
  sprintf("M (SD) = %1.2f (%1.2f)", mean(x), sd(x))
}))
```

Poisson Regression with R

```
##                  General                   Academic
## "M (SD) = 0.20 (0.40)" "M (SD) = 1.00 (1.28)" "M (SD)

ggplot(p, aes(num_awards, fill = prog)) +
  geom_histogram(binwidth=.5, position="dodge")
```

Negative Binomial Regression with R

```
ggplot(dat, aes(daysabs, fill = prog)) + geom_histogram(b
    ., margins = TRUE, scales = "free")
```

```
with(dat, tapply(daysabs, prog, function(x) {
    sprintf("M (SD) = %1.2f (%1.2f)", mean(x), sd(x))
}))
```

```
##                   General                    Academic
## "M (SD) = 10.65 (8.20)"  "M (SD) = 6.93 (7.45)"  "M (S
```