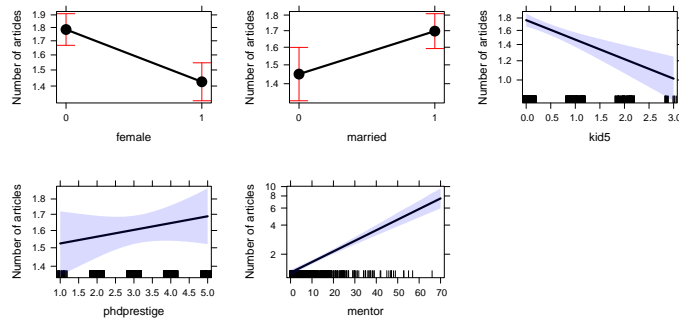


GLMs for Count Data

Michael Friendly

Psych 6136

November 28, 2017



Outline

- 1 Generalized linear models
- 2 GLMs for count data
 - Example: phdpubs
- 3 Model diagnostics
 - Interactions
 - Nonlinearity
 - Outliers, leverage and influence
- 4 Overdispersion
 - Quasi-poisson models
 - Negative-binomial models
- 5 Excess zeros
 - Zero-inflated models
 - Hurdle models
 - Example
- 6 Wrapup

2/74

Generalized linear models

Generalized linear models

Generalized linear models

Generalized linear models

We have used generalized linear models (`glm()`) in two contexts so far:

Loglinear models

- the outcome variable is the **vector of frequencies** \mathbf{y} in a table cross-classified by factors in a design matrix \mathbf{X}
- The model is expressed as a linear model for $\log \mathbf{y}$

$$\log(\mathbf{y}) = \mathbf{X}\beta$$

- The random (or unexplained) variation is expressed as a Poisson distribution for $\mathcal{E}(\mathbf{y} | \mathbf{X})$

Logistic regression

- the outcome variable is a **categorical response** \mathbf{y} , with predictors \mathbf{X}
- The model is expressed as a linear model for the log odds that $y = 1$ vs. $y = 0$.

$$\text{logit}(\mathbf{y}) \equiv \log \left[\frac{\Pr(y = 1)}{\Pr(y = 0)} \right] = \mathbf{X}\beta$$

- The random (or unexplained) variation is expressed as a Binomial distribution for $\mathcal{E}(\mathbf{y} | \mathbf{X})$

Hey, aren't these both very like the familiar, classical linear model,

$$\mathbf{y} = \mathbf{X}\beta + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I}) \quad ?$$

Yes, for some transformation, $g(\mathbf{y})$, and with different distributions!

3/74

4/74

Generalized linear models

Nelder & Wedderburn (1972) said, “Let there be light!”, a **generalized linear model**, encompassing them all, and many more. This has 3 components:

- A **random component**, specifying the conditional distribution of \mathbf{y} given the explanatory variables in \mathbf{X} , with mean $\mathcal{E}(y_j | \mathbf{x}_j) = \mu_j$
 - The normal (Gaussian), binomial, and Poisson are already familiar
 - But, these are all members of an **exponential family**
 - GLMs now include an even wider family: negative-binomial and others
- The **systematic component**, a linear function of the predictors called the **linear predictor**

$$\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta} \quad \text{or} \quad \eta_j = \beta_0 + \beta_1 X_{j1} + \cdots + \beta_p X_{jp}$$

- An invertible **link function**, $g(\mu_i) = \eta_i = \mathbf{x}_i^\top \boldsymbol{\beta}$ that transforms the expected value of the response to the linear predictor
 - The link function is invertible, so we can go back to the **mean function** $g^{-1}(\eta_i) = \mu_i$

5/74

Mean functions

Standard GLM link functions and their inverses:

Table 11.1: Common link functions and their inverses used in generalized linear models

Link name	Function: $\eta_i = g(\mu_i)$	Inverse: $\mu_i = g^{-1}(\eta_i)$
identity	μ_i	η_i
square-root	$\sqrt{\mu_i}$	η_i^2
log	$\log_e(\mu_i)$	$\exp(\eta_i)$
inverse	μ_i^{-1}	η_i^{-1}
inverse-square	μ_i^{-2}	$\eta_i^{-1/2}$
logit	$\log_e \frac{\mu_i}{1-\mu_i}$	$\frac{1}{1+\exp(-\eta_i)}$
probit	$\Phi^{-1}(\mu_i)$	$\Phi(\eta_i)$
log-log	$-\log_e[-\log_e(\mu_i)]$	$\exp[-\exp(-\eta_i)]$
comp. log-log	$\log_e[-\log_e(1-\mu_i)]$	$1 - \exp[-\exp(\eta_i)]$

- The top section recognizes standard transformations often used with traditional linear models
- The bottom section is for binomial data, where y_i represents an observed proportion in n_i trials

6/74

Canonical links and variance functions

- For every distribution family, there is a default, **canonical** link function
- Each one also specifies the expected relationship between mean and variance

Table 11.2: Common distributions in the exponential family used with generalized linear models and their canonical link and variance functions

Family	Notation	Canonical link	Range of y	Variance function, $\mathcal{V}(\mu \eta)$
Gaussian	$N(\mu, \sigma^2)$	identity: μ	$(-\infty, +\infty)$	ϕ
Poisson	$\text{Pois}(\mu)$	$\log_e(\mu)$	$0, 1, \dots, \infty$	μ
Negative-Binomial	$\text{NBin}(\mu, \theta)$	$\log_e(\mu)$	$0, 1, \dots, \infty$	$\mu + \mu^2/\theta$
Binomial	$\text{Bin}(n, \mu)/n$	logit(μ)	$\{0, 1, \dots, n\}/n$	$\mu(1-\mu)/n$
Gamma	$G(\mu, \nu)$	μ^{-1}	$(0, +\infty)$	$\phi\mu^2$
Inverse-Gaussian	$IG(\mu, \nu)$	μ^2	$(0, +\infty)$	$\phi\mu^3$

7/74

Variance functions and over-dispersion

- In the classical Gaussian linear model, the conditional variance is constant, $\phi = \sigma_e^2$.
- For binomial data, the variance function is $\mathcal{V}(\mu_i) = \mu_i(1 - \mu_i)/n_i$, with ϕ fixed at 1
- In the Poisson family, $\mathcal{V}(\mu_i) = \mu_i$ and the dispersion parameter is fixed at $\phi = 1$.
- In practice, it is common for count data to exhibit **overdispersion**, meaning that $\mathcal{V}(\mu_i) > \mu_i$.
- One way to correct for this is to allow the dispersion parameter to be estimated from the data, giving what is called the **quasi-Poisson** family, with $\mathcal{V}(\mu_i) = \hat{\phi}\mu_i$.

8/74

Variance functions and over-dispersion

Overdispersion often results from failures of the assumptions of the model:

- supposedly independent observations may be correlated
- the probability of an event may not be constant, or
- it may vary with unmeasured or unmodeled variables

9/74

ML Estimation

- GLMs are fit by the method of **maximum likelihood**.
- For the Poisson distribution with mean μ , the probability that the random variable Y takes values $y = 0, 1, 2, \dots$ is

$$\Pr(Y = y) = \frac{e^{-\mu} \mu^y}{y!}$$

- In the GLM with a log link, the mean, μ_i depends on the predictors in \mathbf{x} through

$$\log_e(\mu_i) = \mathbf{x}_i^\top \boldsymbol{\beta}$$

- The log-likelihood function (ignoring a constant) for n independent observations has the form

$$\log_e \mathcal{L}(\boldsymbol{\beta}) = \sum_{i=1}^n \{y_i \log_e(\mu_i) - \mu_i\}$$

- It can be shown that the maximum likelihood estimators are solutions to the **estimating equations**,

$$\mathbf{X}^\top \mathbf{y} = \mathbf{X}^\top \boldsymbol{\mu}$$

- The solutions are found by iteratively re-weighted least squares.

10/74

Goodness of fit

- The **residual deviance** defined as twice the difference between the maximum log-likelihood for the **saturated model** that fits perfectly and maximized log-likelihood for the fitted model.

$$D(\mathbf{y}, \hat{\boldsymbol{\mu}}) \equiv 2[\log_e \mathcal{L}(\mathbf{y}; \mathbf{y}) - \log_e \mathcal{L}(\mathbf{y}; \hat{\boldsymbol{\mu}})] .$$

- For classical (Gaussian) linear models, this is just the **residual sum of squares**
- For Poisson models with a log link giving $\boldsymbol{\mu} = \exp(\mathbf{X}^\top \boldsymbol{\beta})$, the deviance takes the form

$$D(\mathbf{y}, \hat{\boldsymbol{\mu}}) = 2 \sum_{i=1}^n \left[y_i \log_e \left(\frac{y_i}{\hat{\mu}_i} \right) - (y_i - \hat{\mu}_i) \right] .$$

- For a GLM with p parameters, both the Pearson and residual deviance statistics follow approximate χ_{n-p}^2 distributions with $n - p$ degrees of freedom.

11/74

GLMs for count data

- Typically, these are fit using: `glm(y ~ x1 + x2 + x3, family=poisson, data=mydata)`
- As in other linear models, the predictors x_j can be discrete factors, quantitative variables, and so forth.
- This fixes the dispersion parameter ϕ to 1, assuming that the count variable y conditional on x_1, x_2, \dots is Poisson distributed.
- It is possible to fit a **quasi Poisson** model, allowing ϕ to be estimated from the data. Specify: `family=quasipoisson`. This allows the variance to be proportional to the mean,

$$\mathcal{V}(y_i | \eta_i) = \phi \mu_i$$

- Another possibility is the negative-binomial model, which has

$$\mathcal{V}(y_i | \eta_i) = \mu_i + \mu_i^2 / \theta$$

12/74

Example: Publications of PhD Candidates

Example 3.24 in DDAR gives data on the number of publications by PhD candidates in biochemistry in the last 3 years of study

```
data("PhdPubs", package = "vcdExtra")
table(PhdPubs$articles)
```

```
##
##  0  1  2  3  4  5  6  7  8  9 10 11 12 16 19
## 275 246 178 84 67 27 17 12 1 2 1 1 2 1 1
```

- Predictors are: gender, marital status, number of young children, prestige of the doctoral department, and number of publications by the student's mentor.

13/74

Example: Publications of PhD Candidates

- Initially, ignore the predictors.
- For the Poisson, equivalent to an intercept-only model:
`glm(articles ~ 1, data=PhdPubs, family="poisson")`

As a quick check on the Poisson assumption:

```
with(PhdPubs, c(mean=mean(articles),
                 var=var(articles),
                 ratio=var(articles)/mean(articles)))

##  mean  var  ratio
## 1.6929 3.7097 2.1914
```

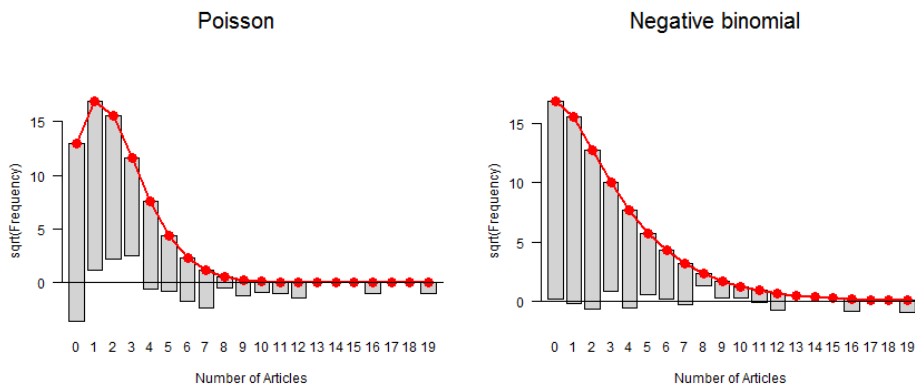
The assumption that mean = variance could be met when we add predictors.

14/74

Example: Publications of PhD Candidates

First, look at rootograms:

```
plot(goodfit(PhdPubs$articles), xlab = "Number of Articles",
     main = "Poisson")
plot(goodfit(PhdPubs$articles, type = "nbinomial"),
     xlab = "Number of Articles", main = "Negative binomial")
```



One reason the Poisson doesn't fit: excess 0s (some never published?)

15/74

Fitting the Poisson model

Fit the model with all main effects:

```
# predictors: female, married, kid5, phdprestige, mentor
phd.pois <- glm(articles ~ ., data=PhdPubs, family=poisson)
Anova(phd.pois)
```

```
## Analysis of Deviance Table (Type II tests)
##
## Response: articles
##          LR Chisq Df Pr(>Chisq)
## female      17.1  1  3.6e-05 ***
## married       6.6  1   0.01 *
## kid5         22.1  1  2.6e-06 ***
## phdprestige  1.0  1   0.32
## mentor     126.8  1 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Only `phdprestige` is NS; it does no harm to keep it, for now.

16/74

Interpreting coefficients

β_j is the **increment** in log (articles) for a 1 unit change in x_j ; $\exp(\beta_j)$ is the **multiple** of articles:

```
round(cbind(beta = coef(phd.pois),
            expbeta = exp(coef(phd.pois)),
            pct = 100 * (exp(coef(phd.pois)) - 1)), 3)
```

```
##          beta expbeta    pct
## (Intercept)  0.266   1.304  30.425
## female1     -0.224   0.799 -20.102
## married1     0.157   1.170  17.037
## kid5        -0.185   0.831 -16.882
## phdprestige  0.025   1.026   2.570
## mentor       0.025   1.026   2.555
```

Thus:

- females publish -0.224 fewer log (articles), or $0.8 \times$ that of males
- married publish 0.157 more log (articles); or $1.17 \times$ unmarried (17% increase)
- each additional young child decreases this by 0.185; or $0.831 \times$ articles (16.9% decrease)
- each mentor pub multiplies student pub by 1.026, a 2.6% increase

17/74

Model diagnostics

Model diagnostics

Diagnostic tests for count data GLMs are similar to those used for classical linear models

- Test for presence of interactions
 - Fit model(s) with some or all two-way interactions
- Non-linear effects of quantitative predictors?
 - Component-plus-residual plots— `car::crPlot()` are useful here
- Outliers? Influential observations?
 - `car::influencePlot()` is your friend

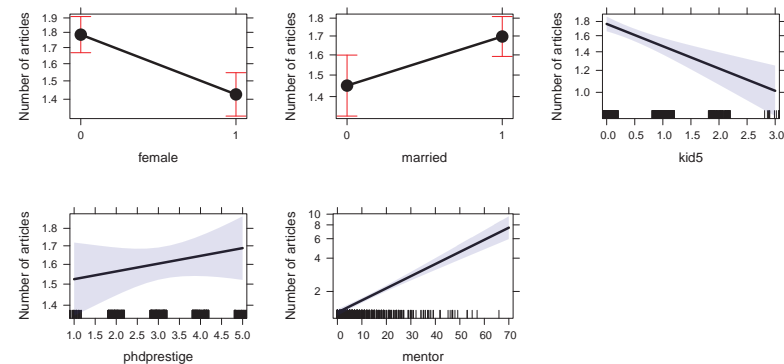
For count data models, we should also check for **over-dispersion**. This is similar to **homogeneity of variance** checks in `lm()`

19/74

Effect plots

As usual, we can understand the fitted model from predicted values for the model effects:

```
library(effects); plot(allEffects(phd.pois))
```



These are better visual summaries for a model than a table of coefficients.

18/74

Model diagnostics Interactions

Testing for interactions

As a quick check for interactions, fit the model with all two-way terms

```
phd.pois1 <- update(phd.pois, . ~ .^2)
Anova(phd.pois1)
```

```
## Analysis of Deviance Table (Type II tests)
##
## Response: articles
##          LR Chisq Df Pr(>Chisq)
## female          14.5  1  0.00014 ***
## married          6.2  1  0.01277 *
## kid5            19.5  1  9.8e-06 ***
## phdprestige      1.0  1  0.32655
## mentor          128.1  1  < 2e-16 ***
## female:married    0.3  1  0.60995
## female:kid5       0.1  1  0.72929
## female:phdprestige 0.2  1  0.63574
## female:mentor     0.0  1  0.91260
## married:kid5      0    0
## married:phdprestige 1.7  1  0.19153
## married:mentor    1.2  1  0.28203
## kid5:phdprestige  0.2  1  0.68523
## kid5:mentor       2.8  1  0.09290 .
## phdprestige:mentor 3.8  1  0.05094 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

20/74

Compare models I

Compare models: LR tests for nested models (`anova()`), and AIC/BIC (`LRstats()`)

```
anova(phd.pois, phd.pois1, test="Chisq")

## Analysis of Deviance Table
##
## Model 1: articles ~ female + married + kid5 + phdprestige + mentor
## Model 2: articles ~ female + married + kid5 + phdprestige + mentor + fe
##   female:kid5 + female:phdprestige + female:mentor + married:kid5 +
##   married:phdprestige + married:mentor + kid5:phdprestige +
##   kid5:mentor + phdprestige:mentor
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1      909      1634
## 2      900      1618  9      15.2   0.086 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

21/74

Compare models II

```
LRstats(phd.pois, phd.pois1)

## Likelihood summary table:
##      AIC  BIC LR Chisq Df Pr(>Chisq)
## phd.pois 3313 3342  1634 909 <2e-16 ***
## phd.pois1 3316 3388  1618 900 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

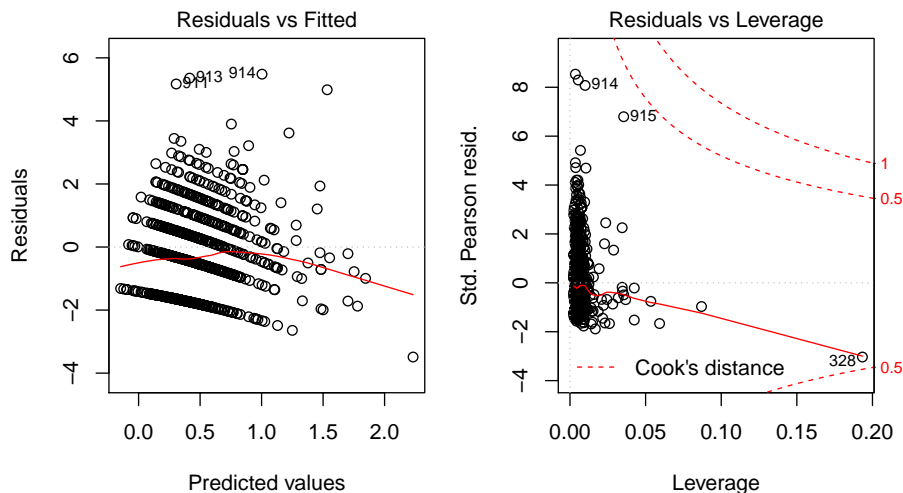
- There seems to be no reason to include interactions in the model
- We might want to re-visit this, after examining other models for the basic count distribution (quasi-poisson, negative-binomial)

22/74

Basic model plots

Only two of the standard model plots are informative for count data models

```
plot(phd.pois, which=c(1,5))
```



23/74

Nonlinearity diagnostics

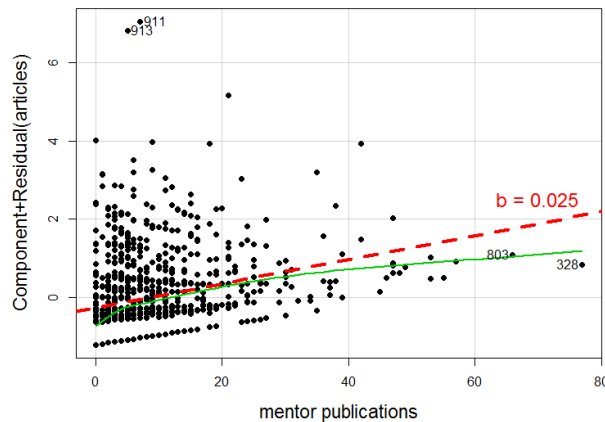
- Non-linear relations are difficult to assess in **marginal plots**, because they don't control (or adjust) for other predictors
- **Component-plus-residual** plots (also called **partial residual plots**) can show non-linear relations for numeric predictors
 - These graph the value of $\hat{\beta}_i x_i + \text{residual}_i$ vs. the predictor, x_i .
 - In this plot, the slope of the points is the coefficient, $\hat{\beta}_i$ in the full model
 - The residual is $y_i - \hat{y}_i$ in the full model
- A non-parametric (e.g., `loess()`) smooth makes it easy to detect non-linearity

24/74

Nonlinearity diagnostics: car::crPlot()

Is the relationship between articles published by the student and the mentor adequately represented as linear?

```
crPlot(phd.pois, "mentor", pch=16, lwd=4, id.n=2)
```



25/74

Residuals I

Several types of residuals can be defined based on the Pearson and deviance goodness-of-fit measures

- the **Pearson residual** is the case-wise contribution to Pearson χ^2

$$r_i^P = \frac{y_i - \hat{\mu}_i}{\sqrt{\hat{v}(y_i)}}$$

- the **deviance residual** is the signed square root of the contribution to the deviance G^2

$$r_i^D = \text{sign}(y_i - \hat{\mu}_i) \sqrt{d_i}$$

- Both of these have **standardized** forms that correct for conditional variance and leverage, and have approx. $\mathcal{N}(0, 1)$ distributions.

$$\tilde{r}_i^P = \frac{r_i^P}{\sqrt{\hat{\phi}(1 - h_i)}}$$

$$\tilde{r}_i^D = \frac{r_i^D}{\sqrt{\hat{\phi}(1 - h_i)}}$$

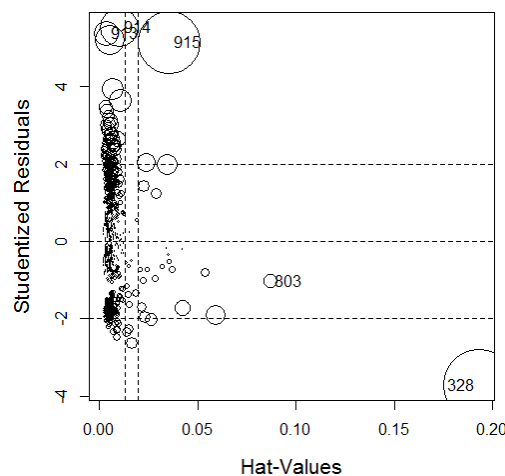
26/74

Residuals II

- The most useful is the **studentized residual** (or deletion residual), `rstudent()` in R. This estimates the standardized residual resulting from omitting each observation in turn. An approximation is:

$$\tilde{r}_i^S = \text{sign}(y_i - \hat{\mu}_i) \sqrt{(1 - h_i)(\tilde{r}_i^D)^2 + h_i(\tilde{r}_i^P)^2}$$

Outliers, leverage and influence



`influencePlot(phd.pois)`

- Several observations (913–915) stand out with large + residuals
- One observation (328) has a large leverage
- Why are they unusual? Do they affect our conclusions?
- Look back at data & decide what to do!

27/74

28/74

Outliers, leverage and influence

At the very least, we should look at these observations in the data:

```
PhdPubs[c(328, 913:915), ]
##      articles female married kid5 phdprestige mentor
## 328         1       0       1     1           2      77
## 913        12       0       1     1           2       5
## 914        16       0       1     0           2      21
## 915        19       0       1     0           2      42
```

- case 328: Mentor published 77 papers! Student, only 1
- 913–915: all published >> predicted

29/74

Overdispersion

Overdispersion

- The Poisson model for counts assumes $\mathcal{V}(\mu_i) = \mu_i$, i.e., the dispersion parameter $\phi = 1$
- But often, the counts exhibit greater variance than the Poisson distribution allows, $\mathcal{V}(\mu_i) > \mu_i$ or $\phi > 1$
 - The observations (counts) may not be independent (clustering)
 - The probability of an “event” may not be constant
 - There may be unmeasured influences, not accounted for in the model
 - These effects are sometimes called “unmodeled heterogeneity”
- The consequences are:
 - Standard errors of the coefficients, $se(\hat{\beta}_j)$ are optimistically small
 - Wald tests, $z_j = \hat{\beta}_j / se(\hat{\beta}_j)$, are too large, and thus overly liberal.

31/74

Outlier test

- A formal test for outliers can be based on the [studentized residuals](#), `rstudent(model)`, using the standard normal distribution for p -values
- A Bonferroni correction should be applied, because interest focuses on the [largest \$n\$ absolute residuals](#).

For the Poisson model, 4 observations are nominated as large + outliers:

```
outlierTest(phd.pois, cutoff=0.001)
##      rstudent unadjusted p-value Bonferonni p
## 914     5.5423         2.9852e-08  2.7315e-05
## 913     5.3821         7.3617e-08  6.7360e-05
## 911     5.2074         1.9153e-07  1.7525e-04
## 915     5.1504         2.5988e-07  2.3779e-04
```

30/74

Overdispersion

Testing overdispersion

- Statistical tests for overdispersion are described in DDAR §11.3.4.
- They test $H_0 : \mathcal{V}(y) = \mu$, vs. H_1 that variance depends on the mean according to some function $f(\mu)$

$$\mathcal{V}(y) = \mu + \alpha \times f(\mu)$$

- This is implemented in `dispersiontest()` in the [AER](#) package.
 - If significant, overdispersion should not be ignored
 - Alternatively, you can try fitting a more general model to see what difference it makes.

32/74

Overdispersion: Quasi-poisson models

- Instead, we can fit another version of the model in which the dispersion ϕ is a free parameter, estimated along with the other coefficients. That is, the conditional variance is allowed to be

$$\mathcal{V}(y_i | \eta_i) = \phi \mu_i$$

- This model is fit with `glm()` using `family=quasipoisson`
 - the estimated coefficients $\hat{\beta}$ are **unchanged**
 - the standard errors are multiplied by $\hat{\phi}^{1/2}$
 - peace, order, and good governance is restored!

33/74

Fitting the quasi-poisson model

The quasi-Poisson model is can be fit using `glm()` as:

```
phd.qpois <- glm(articles ~ ., data=PhdPubs, family=quasipoisson)
```

The dispersion parameter estimate $\hat{\phi}$ can be obtained as follows:

```
(phi <- summary(phd.qpois)$dispersion)
```

```
## [1] 1.8304
```

This is much better than variance/mean ratio of 2.91 calculated for the marginal distribution ignoring the predictors.

35/74

Overdispersion: Quasi-poisson models

- One estimate of the dispersion parameter is the residual deviance divided by degrees of freedom $\hat{\phi} = D(\mathbf{y}, \hat{\mu})/df$
- The Pearson χ^2 statistic has better statistical properties and is more commonly used

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

For the PhdPubs data, these estimates are quite similar: about 80% overdispersion

```
with(phd.pois, deviance / df.residual)
## [1] 1.7971

sum(residuals(phd.pois, type = "pearson")^2) / phd.pois$df.residual
## [1] 1.8304
```

34/74

Coefficients unchanged; std. errors multiplied by $\hat{\phi}^{1/2} = \sqrt{1.83} = 1.35$.

```
summary(phd.qpois)

##
## Call:
## glm(formula = articles ~ ., family = quasipoisson, data = PhdPubs)
##
## Deviance Residuals:
##   Min       1Q   Median       3Q      Max
## -3.488  -1.538  -0.365   0.577   5.483
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.26562    0.13478    1.97  0.04906 *
## female1     -0.22442    0.07384   -3.04  0.00244 **
## married1     0.15732    0.08287    1.90  0.05795 .
## kid5        -0.18491    0.05427   -3.41  0.00069 ***
## phdprestige  0.02538    0.03419    0.74  0.45815
## mentor      0.02523    0.00275    9.19 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for quasipoisson family taken to be 1.8304)
##
## Null deviance: 1817.4  on 914  degrees of freedom
## Residual deviance: 1633.6  on 909  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 5
```

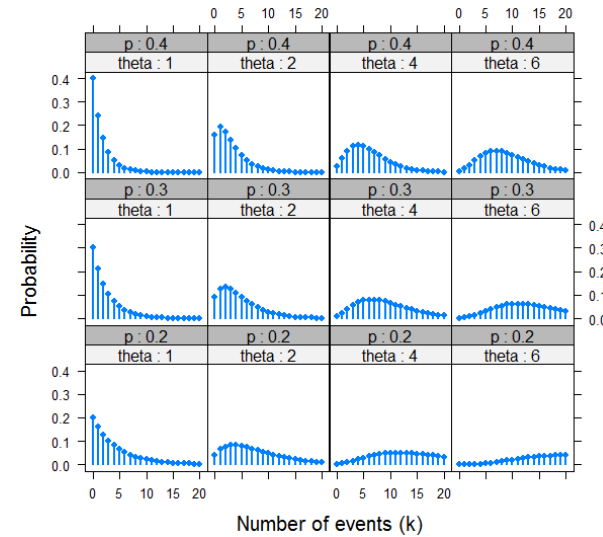
The negative-binomial model

- The negative-binomial model is a different generalization of the Poisson that allows for over-dispersion
- Mathematically, it allows the mean $\mu | \mathbf{x}_i$ to vary across observations as a gamma distribution with a shape parameter θ .
- The variance function, $\mathcal{V}(y_i) = \mu_i + \mu_i^2/\theta$, allows the variance of y to increase more rapidly than the mean.
- Another parameterization uses $\alpha = 1/\theta$

$$\mathcal{V}(y_i) = \mu_i + \mu_i^2/\theta = \mu_i + \alpha\mu_i^2,$$

- As $\alpha \rightarrow 0$, $\mathcal{V}(y_i) \rightarrow \mu_i$ and the negative-binomial converges to the Poisson.

The negative-binomial model



Overdispersion decreases as θ increases

37/74

38/74

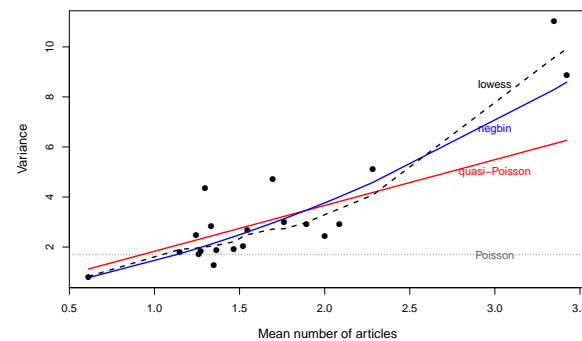
The negative-binomial model: Fitting

- For fixed θ , the negative-binomial is another special case of the GLM
- This is handled in the **MASS** package, with `family=negative.binomial(theta)`
- But most often, θ is unknown, and must be estimated from the data
- This is implemented in `glm.nb()` in the **MASS** package.

```
library(MASS)
phd.nbin <- glm.nb(articles ~ ., data=PhdPubs)
```

Visualizing the mean variance relation

One way to see the difference among models is to plot the variance vs. mean for **grouped** values of the fitted linear predictor.



- The smoothed (loess) curve gives the **empirical mean–variance** relationship
- Also plot the theoretical mean–variance from different models
- For PhdPubs, the data is most similar to the negative-binomial
- The models differ most for those with > 3 articles

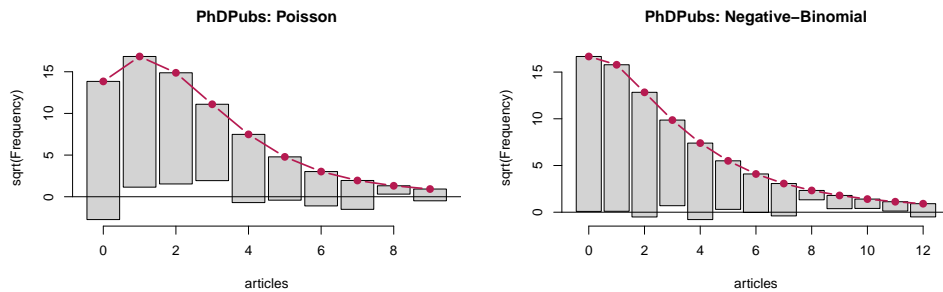
39/74

40/74

Visualizing goodness-of-fit

The `countreg` package extends the `rootogram()` function to work with fitted models:

```
countreg::rootogram(phd.pois, main="PhDPubs: Poisson")
countreg::rootogram(phd.nbin, main="PhDPubs: Negative-Binomial")
```



The Poisson model shows a systematic, wave-like pattern with excess zeros, too few observed frequencies for counts of 1–3.

What difference does it make?

The NB is certainly a better fit than the Poisson; the QP cannot be distinguished by standard tests

```
LRstats(phd.pois, phd.qpois, phd.nbin)

## Likelihood summary table:
##           AIC  BIC LR Chisq  Df Pr(>Chisq)
## phd.pois  3313 3342    1634  909    <2e-16 ***
## phd.qpois
## phd.nbin  3135 3169    1004  909     0.015 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Can also compare standard errors of the coefficients:

```
##           pois  qpois  nbin
## (Intercept) 0.100 0.135 0.133
## female1     0.055 0.074 0.073
## married1    0.061 0.083 0.082
## kid5        0.040 0.054 0.053
## phdprestige 0.025 0.034 0.034
## mentor      0.002 0.003 0.003
```

41/74

42/74

What have we learned?

A summary for an article to this point would use the result of negative-binomial model, from `summary(phd.nbin)`

- The number of articles published by these PhD candidates is most strongly influenced by publications of their mentor
- Increasing young children (`kids5`) results in fewer publications.
- Being married is marginally non-significant— don't interpret
- The prestige of the university doesn't make a difference
- There are still some remaining doubts:
 - Several cases (328, 913–915) appeared unusual in earlier diagnostic plots. Refit without them to see if any conclusions change.
 - The NB model seems to account for the zero counts— students who never published.
 - Is there a better way?

Excess zero counts

- A common problem in count data models is that many sets of data have more observed zero counts than the (quasi) Poisson or NB models can handle.
 - In the PhdPubs data, 275 of 915 (30%) candidates published zilch, bupkis
 - The expected count of 0 articles in the Poisson model is only 191 (21%)
- Maybe there are two types of students giving zero counts:
 - Those who never intend to publish (non-academic career path?)
 - The rest, who do intend to publish, but have not yet done so
 - This suggests the idea of **zero inflation**
- An alternative idea is that there is some **hurdle** to overcome before attaining a positive count, e.g., external pressure from the mentor.

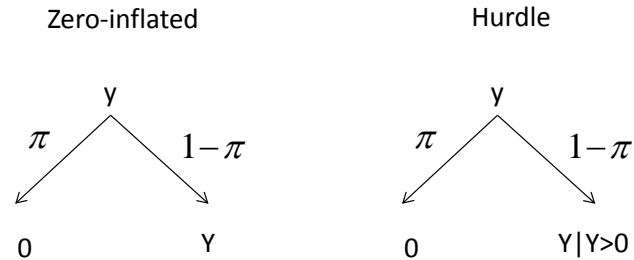
Beyond simply identifying this as a problem of lack-of-fit, understanding the **reasons** for excess zero counts can contribute to a more complete explanation of the phenomenon of interest.

43/74

44/74

Two model types for excess zeros

- **zero-inflated models:** The responses with $y_i = 0$ arise from a mixture of **structural**, always 0 values, with $\Pr(y_i = 0) = \pi_i$ and the rest, which are **random** 0s, with $\Pr(y_i = 0) = 1 - \pi_i$
- **hurdle models:** One process determines whether $y_i = 0$ with $\Pr(y_i = 0) = \pi_i$. A second process determines the distribution of values of positive counts, $\Pr(y_i | y_i > 0)$



45/74

Zero-inflated models

The **zero-inflated Poisson** (ZIP) model has two components:

- A logistic regression model for membership in the unobserved (latent) class of those for whom y_i is necessarily zero

$$\text{logit}(\pi_i) = \mathbf{z}_i^T \boldsymbol{\gamma} = \gamma_0 + \gamma_1 z_{i1} + \gamma_2 z_{i2} + \cdots + \gamma_q z_{iq} .$$

- A Poisson model for the other class (e.g., “publishers”), for whom y_i may be 0 or positive.

$$\log_e \mu(y_i | \mathbf{x}_i) = \mathbf{x}_i^T \boldsymbol{\beta} = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_q x_{ip} .$$

In applications, the same predictors can be (and often are) used in both models ($\mathbf{x} = \mathbf{z}$).

46/74

Zero-inflated models

In the ZIP model, the probabilities of observing counts of $y_i = 0$ and $y_i > 0$ are:

$$\Pr(y_i = 0 | \mathbf{x}, \mathbf{z}) = \pi_i + (1 - \pi_i) e^{-\mu_i}$$

$$\Pr(y_i | \mathbf{x}, \mathbf{z}) = (1 - \pi_i) \times \left[\frac{\mu_i^{y_i} e^{-\mu_i}}{y_i!} \right], \quad y_i \geq 0 .$$

The conditional expectation and variance of y_i then are:

$$\mathcal{E}(y_i) = (1 - \pi_i) \mu_i$$

$$\mathcal{V}(y_i) = (1 - \pi_i) \mu_i (1 + \mu_i \pi_i) .$$

When $\pi_i > 0$, the mean of y is always less than μ_i ; the variance of y is greater than its mean by a dispersion factor of $(1 + \mu_i \pi_i)$.

The model for the count variable could also be negative-binomial, giving a **zero-inflated negative-binomial** (ZINB) model using $\text{NBin}(\mu, \theta)$

47/74

Zero-inflated data

Generate some random data from $\text{Pois}(3) = \text{ZIP}(3, \pi = 0)$ and $\text{ZIP}(3, \pi = 0.3)$. This uses **rzipois()** in the **VGAM**.

```
library(VGAM)
set.seed(1234)
data1 <- rzipois(200, 3, 0)
data2 <- rzipois(200, 3, .3)
```

Tables of the counts:

```
table(data1)

## data1
##  0  1  2  3  4  5  6  7  8  9
## 10 31 46 54 24 20  7  3  4  1

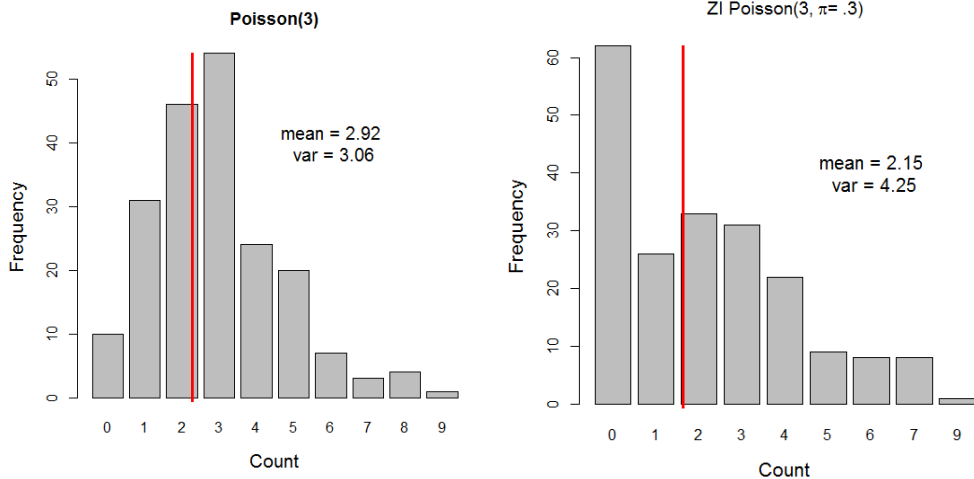
table(data2)

## data2
##  0  1  2  3  4  5  6  7  9
## 62 26 33 31 22  9  8  8  1
```

48/74

Zero-inflated data

Bar plots of the counts:



The 30% extra zeros decrease the mean and inflate the variance

Hurdle models

The Hurdle model also has two components:

- A logistic regression model, for the probability that $y_i = 0$ vs. $y_i > 0$

$$\text{logit} \left[\frac{\Pr(y_i = 0)}{\Pr(y_i > 0)} \right] = \mathbf{z}_i^T \boldsymbol{\gamma} = \gamma_0 + \gamma_1 z_{i1} + \gamma_2 z_{i2} + \dots + \gamma_q z_{iq}$$

- A model for the positive counts, taken as a left-truncated Poisson or negative-binomial, excluding the zero counts
- Comparing the ZIP and Hurdle models:
 - In ZIP models, the first (latent) process generates extra zeros (with probability π_i).
 - In Hurdle models, $y_i = 0$ and $y_i > 0$ are fully observed. The first process generates all the zeros.

Fitting ZIP and Hurdle models

In R, these models can be fit using the `pscl` and `countreg` packages.

`countreg` is more mature, but is only available on R-Forge, not on CRAN. Use:

```
install.packages("countreg", repos="http://R-Forge.R-project.org")
```

The functions have the following arguments:

```
zeroinfl(formula, data, subset, na.action, weights, offset,
         dist = c("poisson", "negbin", "geometric", "binomial"),
         ...)
hurdle(formula, data, subset, na.action, weights, offset,
       dist = c("poisson", "negbin", "geometric", "binomial"),
       ...)
```

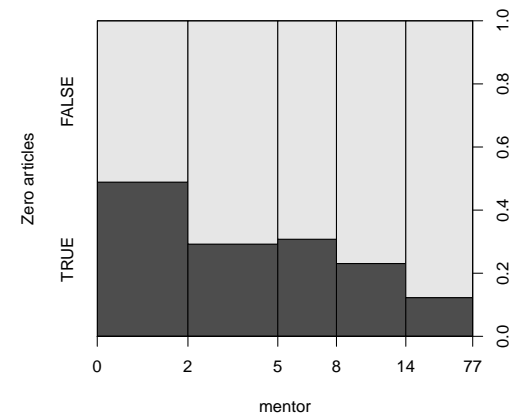
The formula, $y \sim x_1 + x_2 + \dots$ uses the same predictors for both models.

Using $y \sim x_1 + x_2 + \dots | z_1 + z_2 + \dots$ allows separate predictors for the 0 submodel.

Visualizing zero counts

It is often useful to plot the data for the binary distinction between $y_i = 0$ vs. $y_i > 0$ as in logistic regression models.

```
plot(factor(articles==0) ~ mentor, data=PhdPubs,
     ylevels=2:1, ylab="Zero articles",
     breaks=quantile(mentor, probs=seq(0,1,.2)), cex.lab=1.25)
```



Example: Phd Publications

Just to illustrate, we fit all four models, the combinations of (ZI, Hurdle) × (Poisson, NBin) to the PhdPubs data.

For simplicity, we use all predictors for both the zero model and the non-zero model.

```
library(countreg)
phd.zip <- zeroinfl(articles ~ ., data=PhdPubs, dist="poisson")
phd.znb <- zeroinfl(articles ~ ., data=PhdPubs, dist="negbin")

phd.hp <- hurdle(articles ~ ., data=PhdPubs, dist="poisson")
phd.hnb <- hurdle(articles ~ ., data=PhdPubs, dist="negbin")
```

53/74

Test the coefficients in the ZIP model using `lmtest::coefstest()`

```
library(lmtest)
coefstest(phd.zip)

##
## t test of coefficients:
##
##              Estimate Std. Error t value Pr(>|t|)
## count_(Intercept)  0.59918    0.11861    5.05 5.3e-07 ***
## count_female1     -0.20879    0.06353   -3.29 0.0011 **
## count_married1     0.10623    0.07097    1.50 0.1348
## count_kid5         -0.14271    0.04744   -3.01 0.0027 **
## count_phdprestige  0.00700    0.02981    0.23 0.8145
## count_mentor       0.01785    0.00233    7.65 5.3e-14 ***
## zero_(Intercept)  -0.56332    0.49405   -1.14 0.2545
## zero_female1      0.10816    0.28173    0.38 0.7011
## zero_married1     -0.35558    0.31796   -1.12 0.2637
## zero_kid5          0.21974    0.19658    1.12 0.2639
## zero_phdprestige  -0.00537    0.14118   -0.04 0.9697
## zero_mentor       -0.13313    0.04643   -2.87 0.0042 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Only `mentor` is significant for the zero model!

55/74

Example: Phd Publications

Compare models, sorting by BIC:

```
LRstats(phd.pois, phd.nbin, phd.zip, phd.znb, phd.hp, phd.hnb,
        sortby="BIC")

## Likelihood summary table:
##           AIC  BIC LR Chisq  Df Pr(>Chisq)
## phd.pois 3313 3342    3301 909    <2e-16 ***
## phd.hp   3235 3292    3211 903    <2e-16 ***
## phd.zip  3234 3291    3210 903    <2e-16 ***
## phd.hnb  3131 3194    3105 902    <2e-16 ***
## phd.znb  3126 3188    3100 902    <2e-16 ***
## phd.nbin 3135 3169    3121 909    <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The standard negative binomial looks best by BIC. Why do you think this is?

54/74

Re-fit the ZIP and ZNB models using only `mentor` for the zero models:

```
phd.zip1 <- zeroinfl(articles ~ . | mentor, data=PhdPubs, dist="poisson")
phd.znb1 <- zeroinfl(articles ~ . | mentor, data=PhdPubs, dist="negbin")
```

Compare again:

```
LRstats(phd.pois, phd.nbin, phd.zip, phd.znb, phd.hp, phd.hnb,
        phd.zip1, phd.znb1, sortby="BIC")

## Likelihood summary table:
##           AIC  BIC LR Chisq  Df Pr(>Chisq)
## phd.pois 3313 3342    3301 909    <2e-16 ***
## phd.hp   3235 3292    3211 903    <2e-16 ***
## phd.zip  3234 3291    3210 903    <2e-16 ***
## phd.zip1 3227 3266    3211 907    <2e-16 ***
## phd.hnb  3131 3194    3105 902    <2e-16 ***
## phd.znb  3126 3188    3100 902    <2e-16 ***
## phd.nbin 3135 3169    3121 909    <2e-16 ***
## phd.znb1 3124 3168    3106 906    <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Now, the `phd.znb1` model is best by BIC. Why?

56/74

Model interpretation: Coefficients

Ignoring NS coefficients in the revised ZNB model (`phd.znb1`)

```
coef(phd.znb1)[c(1,2,4,6,7,8)]
```

```
## count_(Intercept)      count_female1      count_kid5      count_mentor
##      0.357194         -0.211573         -0.167527         0.024057
## zero_(Intercept)      zero_mentor
##      -0.816912         -0.608024
```

- Count model:

$$\log(\text{articles}) = 0.357 - 0.21 \text{ female} - 0.17 \text{ kids5} + 0.024 \text{ mentor}$$

- Zero model:

$$\text{logit}(\text{articles} = 0) = -0.817 - 0.608 \text{ mentor}$$

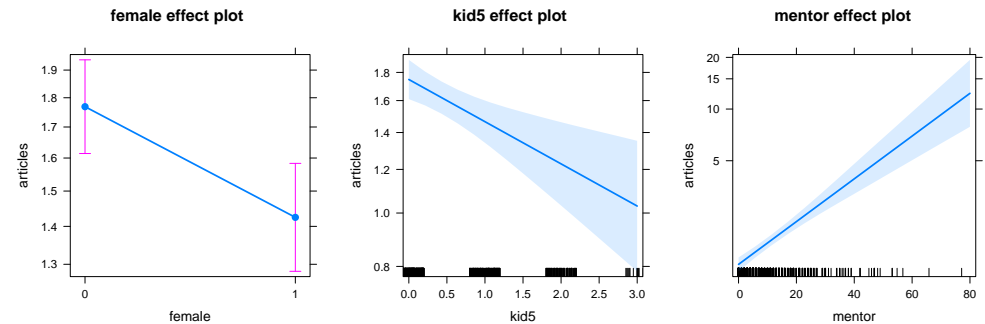
Can you describe these in words?

57/74

Model interpretation: Effect plots

- The `effects` package cannot yet handle zero-inflated or hurdle models.
- But the fitted values don't differ very much among these models
- Here, I use the `phd.nbin` model, and just show the effects for the important terms

```
plot(allEffects(phd.nbin)[c(1,3,5)], rows=1, cols=3)
```

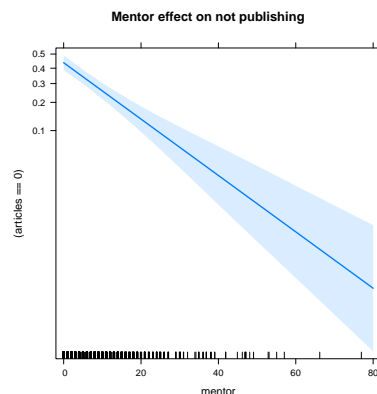


58/74

The ZIP sub-model for the zero counts ("did not publish") can also be interpreted visually

- As an approximation, fit a separate logistic model for `articles==0`
- The effect plot for that gives an interpretation of the zero model.

```
phd.zero <- glm((articles==0) ~ mentor, data=PhdPubs, family=binomial)
plot(allEffects(phd.zero), main="Mentor effect on not publishing")
```



59/74

What have we learned?

- The simple Poisson regression model fits very badly
 - Standard errors do not reflect overdispersion
 - Inference about model effects is compromised by **overly liberal** tests
- The quasi-poisson model corrects for overdispersion.
 - But doesn't account for excess 0s
- The negative-binomial model provides valid tests and fits the 0 counts well.
 - But it doesn't provide any insight into why there are so many 0s
- The ZIP and ZNB models fit well, and account for the 0s.
 - But they lose here on BIC (and AIC) measures, because they have $2\times$ the number of parameters.
 - For simplicity, I have slighted the analogous hurdle models

60/74

What have we learned?

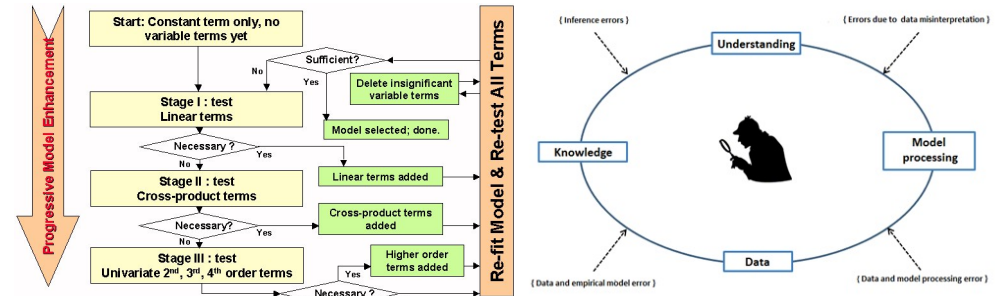
- The revised ZNB model ([phd.znb1](#)), with only `mentor` predicting 0s, wins on parsimony, and has a simple interpretation.
 - The log odds that a student **does not publish** decrease by 0.61 for every article published by the mentor
 - Each mentor pub increases student publications by about 2.5%
 - ⇒ Encourage or help your supervisor to publish!
 - (Or, choose a high publishing one.)
- For this data set, the main substantive interpretation and predicted effects are similar across models. But **details matter!**
- In data sets where there are substantive reasons for excess 0s, the ZI and hurdle models provide **different** explanations.
 - It is not always just a matter of model fit!
 - Hurdle models make the distinction between 0 and > 0 more explicit
 - In ZI models, the interpretation of the mean count is clearer.

61/74

What have we forgotten?

“All models are wrong, but some are useful” — GEP Box

- Model **building** and model **criticism** go hand in hand
- But they don't form a **linear** series of steps, or steps you can put into a flow chart
- Sometimes, you have to go back and re-visit decisions made earlier: Re-think → Re-fit → Re-interpret



62/74

What I missed

- In the initial model, `phdprestige` was NS; I decided to keep it
- In the check for two way interactions, the interaction `phdprestige:mentor` was borderline ($p = 0.051$)
 - I did a global test for all interactions together.
 - That was NS ($p = 0.08$), so I decided to dismiss them all.
 - (I wanted to keep the model simple, to go on to other topics: overdispersion, models for excess zeros.)

63/74

Back to square TWO

- A question in class made me reconsider the `phdprestige:mentor` interaction
- Perhaps the effect of `mentor` varied with `phdprestige`?

Try this, starting with the negative-binomial model, `phd.nbin`

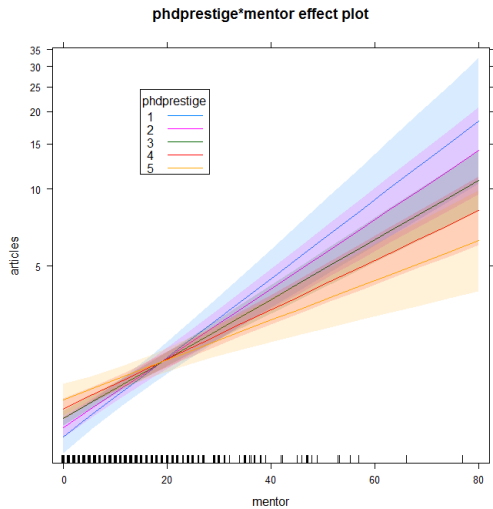
```
phd.nbin2 <- update(phd.nbin, . ~ . + phdprestige:mentor)
Anova(phd.nbin2)

## Analysis of Deviance Table (Type II tests)
##
## Response: articles
##
##          LR      Chisq Df Pr(>Chisq)
## female          9.1   1  0.0026 **
## married          3.1   1  0.0762 .
## kid5            10.7   1  0.0011 **
## phdprestige       0.7   1  0.3921
## mentor          72.8   1 <2e-16 ***
## phdprestige:mentor 5.6   1  0.0179 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

64/74

Visualize the interaction

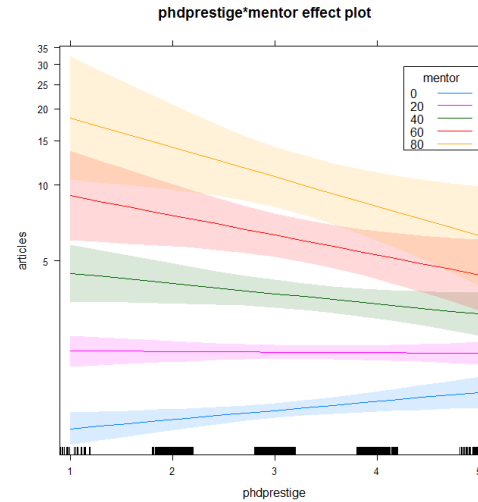
```
phd.effnb2 <- allEffects(phd.nbin2)
plot(phd.effnb2[4], x.var="mentor", multiline=TRUE, ci.style="bands", ...)
```



- An effect plot for `phdprestige*mentor` shows the average over other predictors
- This plot, with `mentor` on the X-axis shows that the slope for `mentor` increases with higher prestige of the student's university

Visualize the interaction—the other way

```
plot(phd.effnb2[4], multiline=TRUE, ci.style="bands", ...)
```



- This plot, with `phdprestige` on the X-axis shows that the slopes change sign depending on the value of `mentor`.
- It explains why the main effect of `phdprestige` is near 0.
- The widths of the confidence bands indicate **model uncertainty**— they get wider as `mentor` pubs increase, and `phdprestige` differs from average.

Back to square ONE

Aren't we done yet?

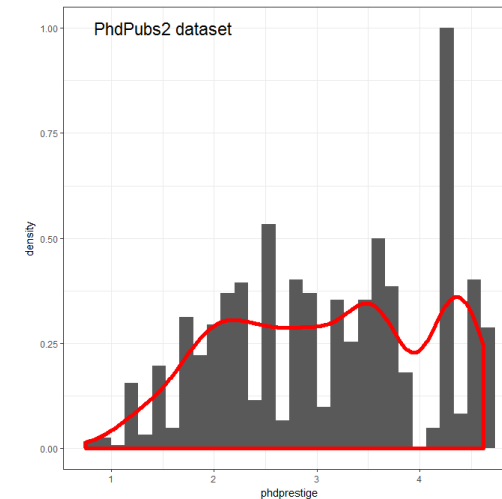
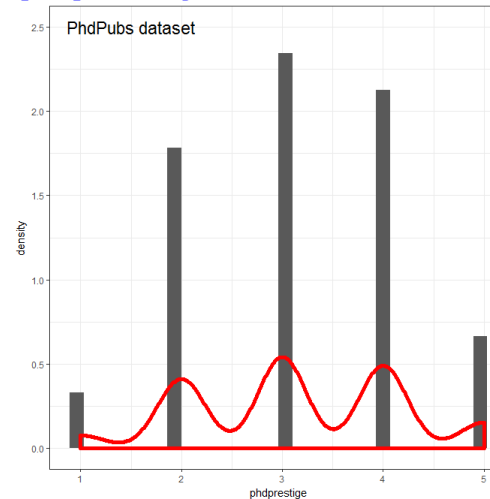
"All data are wrong, but some are useful" — Sitsafe Tsagbey et al., *TAS*, 2017

- A nagging doubt: what is the coding for `phdprestige`?
 - Email from Scott Long: "The higher the number the *more* prestigious the program."
 - "PS: The data I used did not categorize the continuous phd scale into discrete categories"
- Found the original Stata data set:

```
library(foreign)
PhdPubs2 <-
read.dta("http://www.stata-press.com/data/1f2/couart2.dta")
```

Compare distributions

Histograms with smoothed density estimate of the two versions of `phdprestige`:



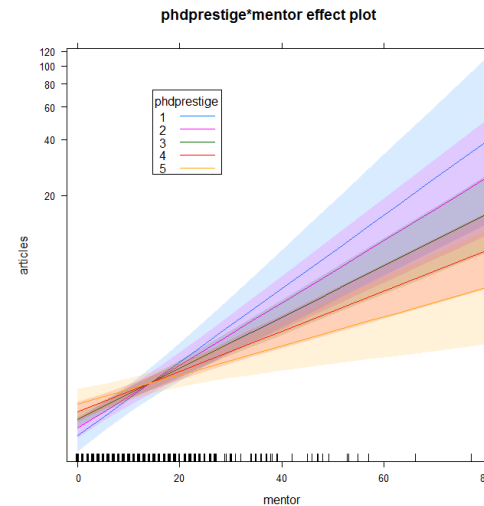
What to do?

Re-run the analysis with the new dataset, `PhdPubs2`

- Are the results for the `phd.nbin2` and `phd.znb2` models about the same? YES!
- Is the interaction `phdprestige:mentor` about the same? YES!
- Does the effect plot look about the same? YES!

Re-interpret the interaction

```
phd.effnb2 <- allEffects(phd.nbin2)
plot(phd.effnb2[4], x.var="mentor", multiline=TRUE, ci.style="bands", ...)
```



- For students from low prestige universities, `mentor` pubs has a large effect on articles published
- As `phdprestige` increases, the slope for `mentor` decreases
- The range of predicted values is much greater than that with the discrete version.
- DONE!

69/74

70/74

What else is there?

The `PhdPubs` example was rather simple, in that:

- There were only a few predictors
 - Model selection methods could be based on simple `Anova()` s or `coefstest()` s
 - No need for more complex model selection methods, or cross-validation
- Of the quantitative predictors, only `mentor` and `kids5` had important effects
 - The effects of `mentor` and `kids5` were sufficiently linear.
 - No need to try polynomial (`poly(mentor, 2)`) or other non-linear effects
- There turned out to be one important interaction.
 - In Psychology, these are often called `moderator` effects
 - Interpretation is often based on post-hoc tests of `simple slopes` or `regions of significance`
 - Interpretation is usually simplified in effect plots.

71/74

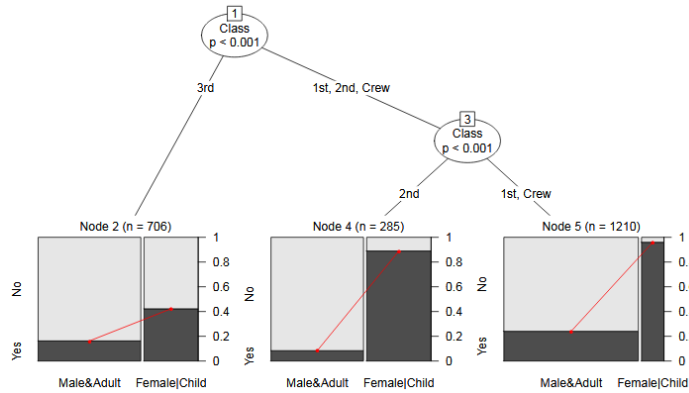
What else is there?

- The response variable, `articles` was measured only once, i.e., there is no `longitudinal` aspect of the analysis.
 - One extension might track the number of articles published by these students over stages in their career.
 - Longitudinal models are examples of `multilevel` or hierarchical linear models
 - Well-developed for classical, Gaussian models (`lm()` → `lme4::lmer()`)
 - These models are now extended to GLMs for count data (e.g., `lm()` → `lme4::glmer()`)
- There was only one response variable: `articles`.
 - Another extension might analyse articles published and the number of job interviews upon graduation as a `multivariate GLM`
 - Yet another, could try to develop a `structural equation model` (SEM) or `path analysis model`, with a variable like “hired within one year?” as the ultimate binary outcome.

72/74

Other methods: Recursive partitioning

- Recursive partitioning, or **regression trees** are often an attractive alternative to linear models
 - Interactions are handled by partitioning the ranges of variables
 - Or, models can be fit to **subsets** of the data defined by recursive partitioning



Logistic regression tree fit to the Titanic data with `partykit::glmtree()`

Other methods: Recursive partitioning

Maybe an even simpler model?

```
install.packages("partykit")
library(partykit)
phd.tree <- glmtree(articles ~ mentor | female+married+kid5+phdprestige,
                    data=PhdPubs, family=poisson)
plot(phd.tree)
```

