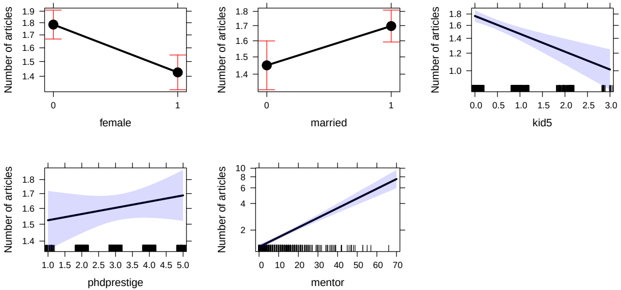
GLMs for Count Data

Michael Friendly

Psych 6136

November 21, 2017



Outline

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 $\boldsymbol{y}$ in a table cross-classified by factors in a design matrix $\boldsymbol{X}$
- The model is expressed as a linear model for log $\boldsymbol{y}$

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

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

Generalized linear models

Logistic regression

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

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

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

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

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

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

Generalized linear models

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

- A random component, specifying the conditional distribution of $\boldsymbol{y}$ given the explanatory variables in $\boldsymbol{X}$, with mean $\mathcal{E}(y_i \mid \boldsymbol{x}_i) = \mu_i$
  - 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

$$\boldsymbol{\eta} = \boldsymbol{X}\boldsymbol{\beta} \quad \text{or} \quad \eta_i = \beta_0 + \beta_1 X_{i1} + \cdots + \beta_p X_{ip}$$

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

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

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, $\mathcal{V}(\mu \mid \eta)$
Gaussian	$N(\mu, \sigma^2)$	identity: μ	$(-\infty, +\infty)$	ϕ
Poisson	$\mathrm{Pois}(\mu)$	$\log_e(\mu)$	$0, 1, \ldots, \infty$	μ
Negative-Binomial	$\mathrm{NBin}(\mu, \theta)$	$\log_e(\mu)$	$0, 1, \ldots, \infty$	$\mu + \mu^2/\theta$
Binomial	$\mathrm{Bin}(n, \mu)/n$	$\mathrm{logit}(\mu)$	$\{0, 1, \ldots, n\}/n$	$\mu(1-\mu)/n$
Gamma	$G(\mu, \nu)$	μ^{-1}	$(0, +\infty)$	$\phi\mu^2$
Inverse-Gaussian	$IG(\mu, \nu)$	μ^2	$(0, +\infty)$	$\phi\mu^3$

Variance functions and over-dispersion

- In the classical Gaussian linear model, the conditional variance is constant, $\phi = \sigma^2_\epsilon$.
- 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 $\mathcal{V}(\mu_i) = \widehat{\phi}\mu_i$.

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

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 $y = 0, 1, 2, \ldots$ 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 $\boldsymbol{x}$ through

$$\log_e(\mu_i) = \boldsymbol{x}_i^{\mathsf{T}} \boldsymbol{\beta}$$

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

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

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

$$\boldsymbol{X}^{\mathsf{T}} \boldsymbol{y} = \boldsymbol{X}^{\mathsf{T}} \boldsymbol{\mu}$$

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

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.

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

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

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

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

GLMs for count data

- Typicaly, 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 `x1`, `x2`, ... 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

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

Example: Publications of PhD Candidates

Example 3.24 in DDAR gives data on the number of publications by PhD candidates 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.

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:

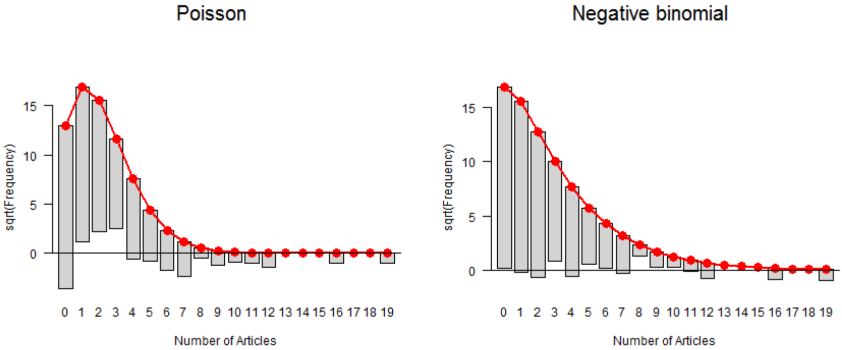
```
with(PhdPubs, c(mean=mean(articles),
                var=var(articles),
                ratio=var(articles)/mean(articles)))

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

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

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?)

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, fot now.

Interpreting coefficients

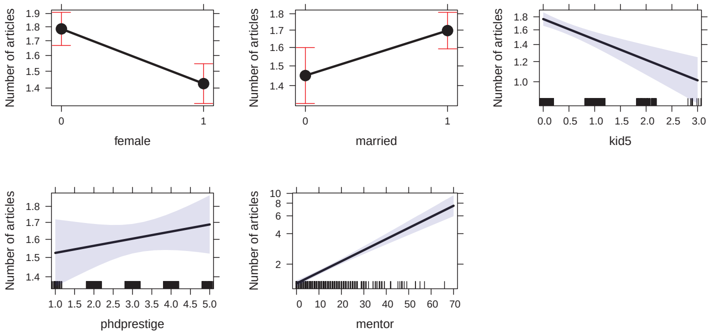
```
round(cbind(beta = coef(phd.pois),
            expbeta = exp(coef(phd.pois)),
            pct = 100 * (exp(coef(phd.pois)) - 1)), 3)

##               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
```

- 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 2.5% increase

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.

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()` is 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()`

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
## 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
```

Compare models I

Compare models:

```
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
```

Compare models II

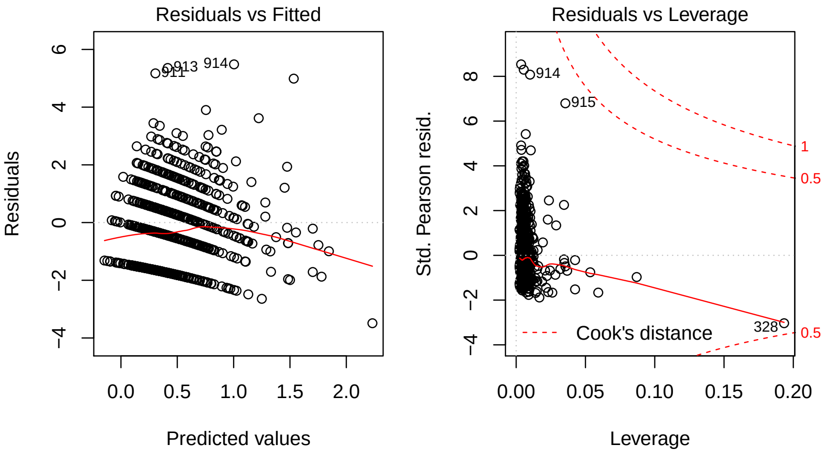
```
LRstats(phd.pois, phd.pois1)

## 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

Basic model plots

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

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



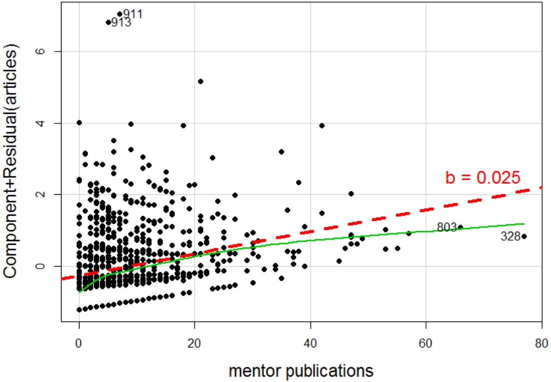
Nonlinearity diagnostics

- 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 $\hat{\beta}_i x_i + \mathrm{residual}_i$ 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

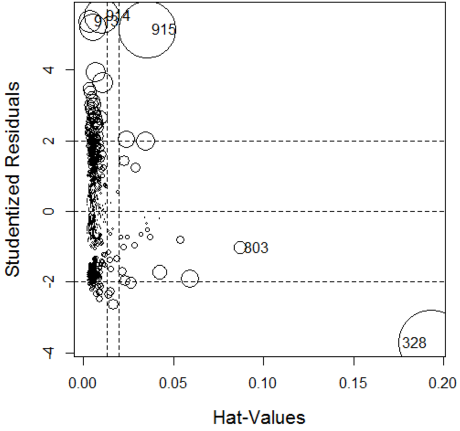
Nonlinearity diagnostics: car::crPlot()

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)
```



Outliers, leverage and influence



`influencePlot(phd.pois)`

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

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!
- 913–915: all published $>>$ predicted

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, $\mathrm{se}(\widehat{\beta}_j)$ are optimistically small
  - Wald tests, $z_j = \widehat{\beta}_j / \mathrm{se}(\widehat{\beta}_j)$, are too large, and thus overly liberal.

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.

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

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

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

Overdispersion: Quasi-poisson models

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

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

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
```

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 $\widehat{\phi}$ 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.

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

```
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
```

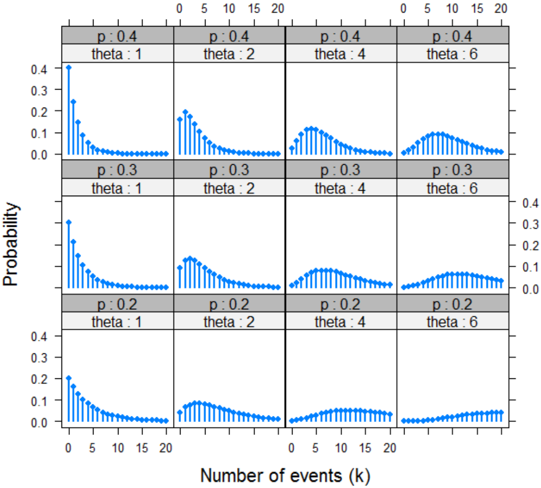
The negative-binomial model

- The negative-binomial model is a different generalization of the Poisson that allows for over-dispersion
- Mathematically, it allows the mean $\mu \mid \boldsymbol{x}_i$ 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 $\alpha \to 0$, $\mathcal{V}(y_i) \to \mu_i$ and the negative-binomial converges to the Poisson.

The negative-binomial model



Overdispersion decreases as θ increases

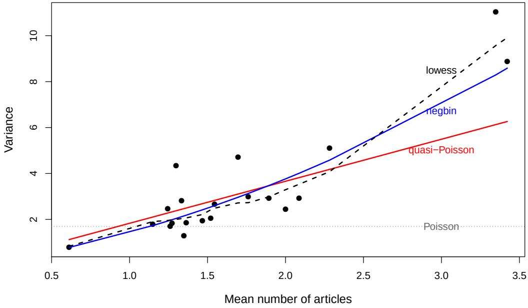
The negative-binomial model: Fitting

- 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.

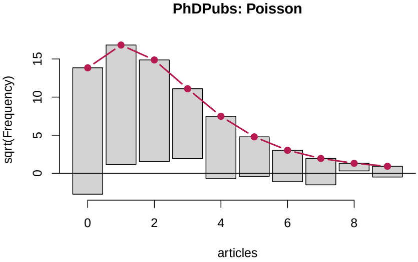


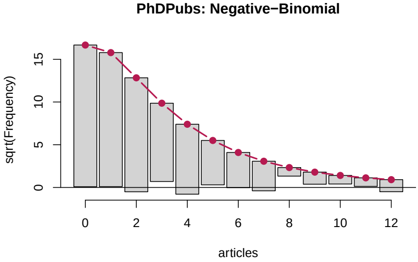
- The smoothed (loess) curve gives the empirical mean–variance relationship
- Also plot the theoretical mean–variance from different models
- For PhdPubs, the data is most similar to the negative-binomial
- The models differ most for those with > 3 articles

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                     909
## 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
```

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?

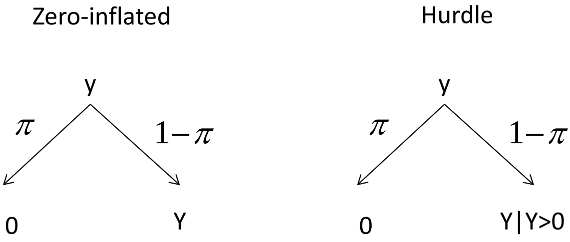
Excess zero counts

- 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.

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, $\Pr(y_i \mid y_i > 0)$



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

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

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

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

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

Zero-inflated models

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

$$
\begin{aligned}
\Pr(y_i = 0 \mid \boldsymbol{x}, \boldsymbol{z}) &= \pi_i + (1 - \pi_i) e^{-\mu_i} \\
\Pr(y_i \mid \boldsymbol{x}, \boldsymbol{z}) &= (1 - \pi_i) \times \left[\frac{{\mu_i}^{y_i} e^{-\mu_i}}{y_i!} \right], \qquad y_i \geq 0 \ .
\end{aligned}
$$

The conditional expectation and variance of y_i then are:

$$
\begin{aligned}
\mathcal{E}(y_i) &= (1 - \pi_i)\, \mu_i \\
\mathcal{V}(y_i) &= (1 - \pi_i)\, \mu_i (1 + \mu_i \pi_i) \ .
\end{aligned}
$$

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 $\text{NBin}(\mu, \theta)$

Zero-inflated data

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

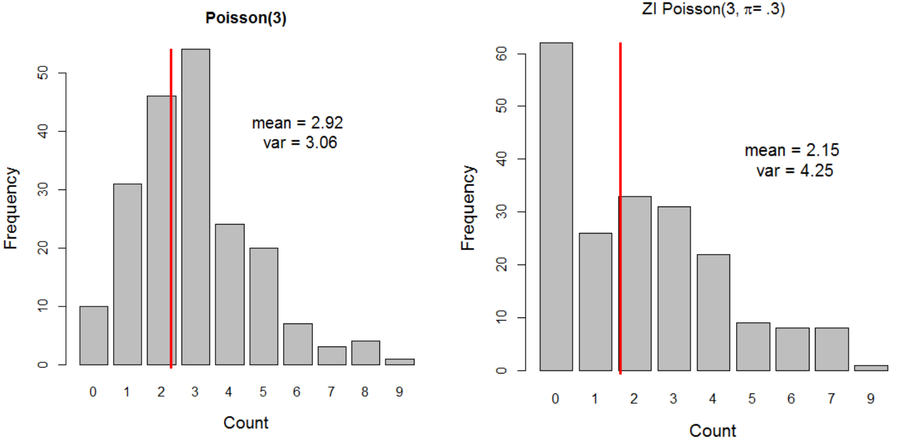
```
library(VGAM)
set.seed(1234)
data1 <- rzipois(200, 3, 0)
data2 <- rzipois(200, 3, .3)
```

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
```

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. $y_i > 0$

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

- 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:

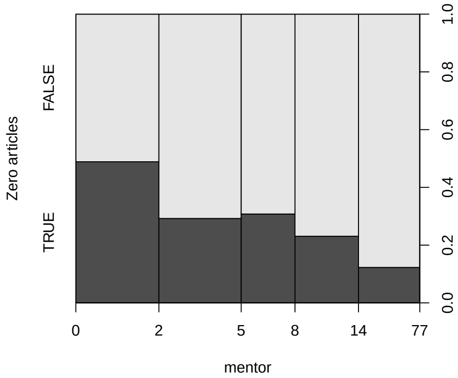
```
zeroinfl(formula, data, subset, na.action, weights, offset,
         dist = c("poisson", "negbin", "geometric", "binomial"),
          ...)

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

The formula, `y ~ x1 + x2 + ...` uses the same predictors for both models.
Using `y ~ x1 + x2 + ...| z1 + z2 + ...` 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) $\times$ (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")
```

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?

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

```
library(lmtest)
coeftest(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!

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?

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:

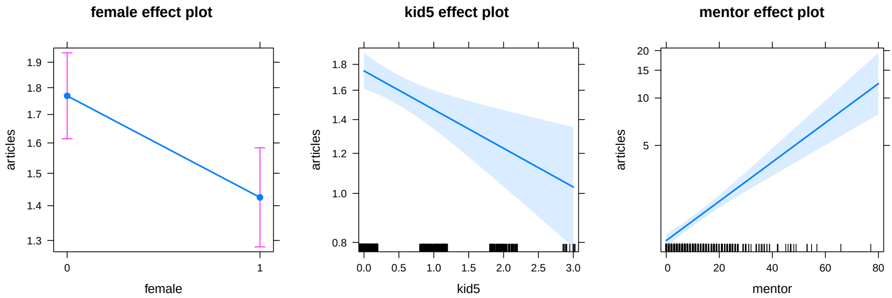
$$\text{logit}(\text{articles} = 0) = -0.817 - 0.608\ \text{mentor}$$

Can you describe these in words?

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)
```



- 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")
```