Session 07

GLM extensions

The Negative Binomial distribution

- Probability function

$$\Pr(Y = y) = \frac{\Gamma(\theta + y)}{\Gamma(\theta)\, y!} \frac{\theta^{\theta} \mu^{y}}{(\theta + \mu)^{\theta + y}}, \quad y = 0, 1, 2, \ldots$$

- Mean-variance relationship

$$\mathrm{Var}[Y] = \mu + \mu^2 / \theta$$

- Software: MASS library function **`glm.nb`** (can also be fitted by optimisation functions, e.g. **`optim`**)

Genesis

- Gamma mixture of Poissons (GLMM)

$$Y \mid G \sim \mathrm{Po}(\mu G), \quad G \sim \gamma(\theta,\theta), \quad \mathrm{E}[G] = 1, \quad \mathrm{Var}[G] = 1/\theta$$

- Compound Poisson

$$Y = X_1 + X_2 + \cdots + X_N, \quad N \sim \mathrm{Po}, \; X_i \sim \text{logarithmic}$$

- Consider the Quine data example: both geneses have some credibility.

The Quine data again: an initial Poisson fit

```
quine.po1 <- glm(Days ~ .^4, poisson, quine, trace = T)

GLM    linear loop 1: deviance = 1373.243
GLM    linear loop 2: deviance = 1178.451
GLM    linear loop 3: deviance = 1173.905
GLM    linear loop 4: deviance = 1173.899
GLM    linear loop 5: deviance = 1173.899

summary(quine.po1, cor = F)

Call: glm(formula = Days ~ (Eth + Sex + Age + Lrn)^4, family =
   poisson, data = quine, trace = T)
…
(Dispersion Parameter for Poisson family taken to be 1 )

   Null Deviance: 2073.533 on 145 degrees of freedom
   Residual Deviance: 1173.899 on 118 degrees of freedom
```

An initial value for theta

- Heuristic: $G \approx Y/\mu$.
- Use the fitted value from the Poisson fit as an estimate of μ.
- $\text{Var}[G] = 1/\theta \Rightarrow \theta \approx 1/\text{Var}[G]$
- *Well, it's worth a try!*

```
t0 <- 1/var(quine$Days/fitted(quine.po1))
t0
[1] 1.966012
```

Initial NB fit and test

```
quine.nb1 <- glm.nb(Days ~ Eth * Lrn * Age * Sex, data
   = quine, init.theta = t0, trace = T)

GLM    linear loop 1: deviance = 176.1057
GLM    linear loop 2: deviance = 169.9369
GLM    linear loop 3: deviance = 169.8431
GLM    linear loop 4: deviance = 169.8431
GLM    linear loop 5: deviance = 169.8431
GLM    linear loop 1: deviance = 167.4535
Theta( 1 ) = 1.92836 , 2(Ls - Lm) = 167.453

quine.nb1$call$trace <- F  # turn off tracing
dropterm(quine.nb1, test = "Chisq")

Single term deletions

Model:
Days ~ Eth * Lrn * Age * Sex
                Df      AIC      LRT   Pr(Chi)
         <none>    1095.324
Eth:Lrn:Age:Sex  2 1092.728 1.403843 0.4956319
```

Backwards elmination to a final model

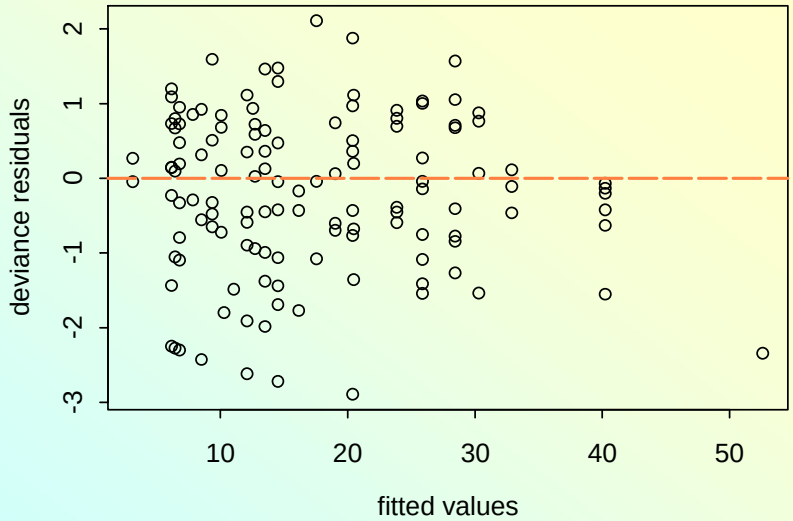
```
quine.nb2 <- update(quine.nb1, . ~ . - Eth:Lrn:Age:Sex)
dropterm(quine.nb2, test = "Chisq", k = log(nrow(quine)))
Single term deletions
…
             Df      AIC      LRT   Pr(Chi)
     <none>    1170.302
Eth:Lrn:Age  2 1166.308 5.973579 0.0504491
Eth:Lrn:Sex  1 1167.914 2.595925 0.1071389
Eth:Age:Sex  3 1158.032 2.680925 0.4434787
Lrn:Age:Sex  2 1166.614 6.279241 0.0432992
quine.nb3 <- update(quine.nb2, . ~ . - Eth:Age:Sex)
dropterm(quine.nb3, test = "Chisq", k = log(nrow(quine)))
Single term deletions
…
Df      AIC      LRT    Pr(Chi)
     <none>    1158.032
Eth:Lrn:Age  2 1153.833 5.768399 0.05589953
Eth:Lrn:Sex  1 1158.087 5.038374 0.02479174
Lrn:Age:Sex  2 1153.766 5.701942 0.05778817
```

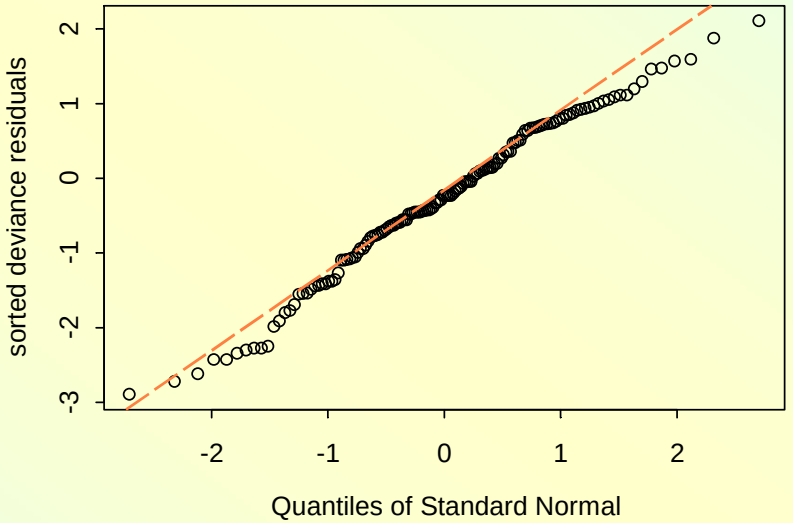
```
quine.nb4 <- update(quine.nb3, . ~ . - Lrn:Age:Sex)
dropterm(quine.nb4, test = "Chisq", k = log(nrow(quine)))
Single term deletions
…
Df      AIC      LRT   Pr(Chi)
     <none>    1153.766
    Age:Sex  3 1158.505 19.68971 0.0001968
Eth:Lrn:Age  2 1148.119  4.32009 0.1153202
Eth:Lrn:Sex  1 1154.271  5.48811 0.0191463
quine.nb5 <- update(quine.nb4, . ~ . - Lrn:Age:Eth)
dropterm(quine.nb5, test = "Chisq", k = log(nrow(quine)))
Single term deletions
…
Df      AIC      LRT   Pr(Chi)
     <none>    1148.119
    Eth:Age  3 1138.559  5.39070 0.1453244
    Lrn:Age  2 1141.940  3.78782 0.1504820
    Age:Sex  3 1154.312 21.14342 0.0000983
Eth:Lrn:Sex  1 1152.251  9.11539 0.0025347
quine.nb6 <- update(quine.nb5, . ~ . - Lrn:Age)
dropterm(quine.nb6, test = "Chisq", k = log(nrow(quine)))
Single term deletions
Df      AIC      LRT   Pr(Chi)
     <none>    1141.940
    Eth:Age  3 1132.796  5.80638 0.1214197
    Age:Sex  3 1145.429 18.43993 0.0003569
Eth:Lrn:Sex  1 1145.395  8.43894 0.0036727
```

```
quine.nb7 <- update(quine.nb6, . ~ . - Eth:Age)
dropterm(quine.nb7, test = "Chisq", k =
   log(nrow(quine)))
Single term deletions

Model:
Days ~ Eth + Lrn + Age + Sex + Eth:Lrn + Eth:Sex +
   Lrn:Sex + Age:Sex + Eth:Lrn:Sex
           Df      AIC      LRT      Pr(Chi)
     <none>    1132.796
    Age:Sex  3 1136.464 18.61934 0.0003276936
Eth:Lrn:Sex  1 1140.234 12.42160 0.0004243969

quine.check <- glm.nb(Days ~ Sex/(Age + Eth * Lrn),
   quine); deviance(quine.nb7); deviance(quine.check)
[1] 167.5558
[1] 167.5558

range(fitted(quine.nb7) - fitted(quine.check))
[1] -0.00006764941  0.00002037681
```

Diagnostic checks

```
fv <- fitted(quine.nb7)
rs <- resid(quine.nb7, type = "deviance")
pv <- predict(quine.nb7)
par(mfrow = c(2,2))
plot(fv, rs, xlab = "fitted values",
       ylab = "deviance residuals")
abline(h = 0, lty = 4, lwd = 2, col = 3)
qqnorm(rs, ylab = "sorted deviance residuals")
qqline(rs, col = 3, lwd = 2, lty = 4)
```





Notes

- We are led to the same model as with the transformed data
- The big advantage we have with this analysis is that it is on the original scale, so predictions would be direct.
- Diagnostic analyses are still useful here, though they are less so with small count data
- Often the value for theta is not critical. One alternative to this is to fit the models with a fixed value for theta as ordinary glm's. See next

Fixing theta at a constant value

```
quine.glm1 <- glm(Days ~ Eth * Sex * Lrn * Age,
   negative.binomial(theta = t0), data = quine, trace = F)
quine.step <- stepAIC(quine.glm1, k = log(nrow(quine)),
   trace = F)
dropterm(quine.step, test = "Chisq")

Single term deletions

Model:
Days ~ Eth + Sex + Lrn + Age + Eth:Sex + Eth:Lrn + Sex:Lrn +
   Sex:Age + Eth:Sex:Lrn
           Df Deviance      AIC scaled dev.      Pr(Chi)
     <none>    195.9901 201.5854
    Sex:Age  3 219.6959 216.5812    20.99584 0.0001054859
Eth:Sex:Lrn  1 211.5179 213.3381    13.75267 0.0002085244
```

- We are led to the same model. This is a common occurrence if theta is a reasonable value to use

Multinomial models (V&R, p. 199 ff)

- Surrogate Poisson models offer a powerful way of analysing frequency data, even if the distribution is not Poisson.
- This is possible because the multinomial distribution can be viewed as a conditional distribution of independent Poisson variables, given their sum
- In multiply classified frequency data, it is important to separate “response” and “stimulus” classifications (which may change according to viewpoint).
- With only one “response” classification, multinomial models may be fitted directly using `multinom`

Example: Copenhagen housing data

- Three ‘stimulus’ classifications: **`Influence`**, **`Type`** and **`Contact`**
- One ‘response’ classificaltion: **`Satisfaction`**
- Null model is **`Influence*Type*Contact`**, which corresponds to equal probabilities of 1/3 for each satisfaction class.
- Simplest real model is **`Influence*Type*Contact+Satisfaction`**, which corresponds to a homogeneity model
- More complex models are tested by their interactions with **`Satisfaction`**

Homogeneity is not adequate

```
hous.glm0 <- glm(Freq ~ Infl*Type*Cont, poisson, housing)
hous.glm1 <- update(hous.glm0, .~.+Sat)
anova(hous.glm0, hous.glm1, test = "Chisq")
```

- (Difference in deviance is 44.65689 on 2 d.f.)

```
addterm(hous.glm1, . ~ . + Sat * (Infl + Type + Cont), test =
   "Chisq")
Single term additions

Model:
Freq ~ Infl + Type + Cont + Sat + Infl:Type + Infl:Cont +
   Type:Cont + Infl:Type:Cont
         Df Deviance      AIC      LRT    Pr(Chi)
  <none>    217.4560 269.4560
Sat:Infl  4 111.0846 171.0846 106.3714 0.00000000
Sat:Type  6 156.7872 220.7872  60.6687 0.00000000
Sat:Cont  2 212.3301 268.3301   5.1258 0.07708018
```

Housing data, cont'd.

- All three terms are necessary, but no more.

```
hous.glm2 <- update(hous.glm1, .~.+Sat*(Infl+Type+Cont))
```

- To find a table of estimated probabilities we need to arrange the fitted values in a table (matrix) and normalize to have row sums unity.
- How do we do this?

```
levs <- lapply(housing[, -5], levels)
dlev <- sapply(levs, length)

ind <- do.call("cbind", lapply(housing[, -5],
  function(x) match(x, levels(x))))

RF <- Pr <- array(0, dim = dlev, dimnames = levs)
RF[ind] <- housing$Freq
tots <- rep(apply(RF, 2:4, sum), each = 3)

RF <- RF/as.vector(tots)
RF

Pr[ind] <- fitted(hous.glm2)
Pr <- Pr/as.vector(tots)
Pr
```

Table 7.4: Estimated probabilities from a main effects model for the Copenhagen housing conditions study.

Contact		Low			High		
Satisfaction		Low	Med.	High	Low	Med.	High
Housing	**Influence**						
Tower blocks	Low	0.40	0.26	0.34	0.30	0.28	0.42
	Medium	0.26	0.27	0.47	0.18	0.27	0.54
	High	0.15	0.19	0.66	0.10	0.19	0.71
Apartments	Low	0.54	0.23	0.23	0.44	0.27	0.30
	Medium	0.39	0.26	0.34	0.30	0.28	0.42
	High	0.26	0.21	0.53	0.18	0.21	0.61
Atrium houses	Low	0.43	0.32	0.25	0.33	0.36	0.31
	Medium	0.30	0.35	0.36	0.22	0.36	0.42
	High	0.19	0.27	0.54	0.13	0.27	0.60
Terraced houses	Low	0.65	0.22	0.14	0.55	0.27	0.19
	Medium	0.51	0.27	0.22	0.40	0.31	0.29
	High	0.37	0.24	0.39	0.27	0.26	0.47

Fitting as a multinomial model

- The function **`multinom`** is set up to take either a factor or a matrix with k columns as the response
- In our case we have frequencies already supplied. These act as case weights.
- “fitted values” from a multinomial fit are the matrix of probability estimates, with the columns corresponding to the response classes. Hence in our case they will occur three times over.
- Fit a multinomial model and check that the fitted values agree with our surrogate Poisson estimates.

```
hous.mult <- multinom(Sat ~ Infl + Type + Cont, data =
   housing, weights = Freq, trace = T)
# weights:  24 (14 variable)
initial  value 1846.767257
iter  10 value 1747.477617
final  value 1735.041934
converged
round(fitted(hous.mult), 2)
    Low Medium High
 1 0.40   0.26 0.34
 2 0.40   0.26 0.34
 3 0.40   0.26 0.34
 4 0.26   0.27 0.47
…
71 0.27   0.26 0.47
72 0.27   0.26 0.47

h1 <- t(fitted(hous.mult)[seq(3, 72, 3),  ])
range(h1 - as.vector(Pr))
[1] -3.763807e-006  3.948444e-006
```

Proportional odds models (V&R p. 204)

- A parametrically economic version of the multinomial
- The response classification is assumed *ordered*.
- The model may be specified as

$$\pi(\mathbf{x}) = \Pr(Y \leq k \mid \mathbf{x}), \quad \log \frac{\pi(\mathbf{x})}{1 - \pi(\mathbf{x})} = \zeta_k - \mathbf{x}^T \boldsymbol{\beta}$$

- Hence the cumulative probabilities conform to a logistic model, with parallelism in the logistic scale.
- MASS library contains a function polr to fit such models

Fitting a Prop. Odds model and checking

```
hous.polr <- polr(Sat ~ Infl+Type+Cont, data = housing,
   weights = Freq)

plot(fitted(hous.polr), fitted(hous.mult))
abline(0, 1, col=3, lty=4, lwd=1)

hous.polr2 <- stepAIC(hous.polr, ~.^2, k = log(24))
hous.polr2$call$formula

polr(formula = Sat ~ Infl + Type + Cont +
   Infl:Type, data = housing, weights = Freq)
```

- With a more parsimonious model the automatic selection procedure uncovers a possible extra term.