Wednesday, March 19

Likelihood Functions

The *likelihood function* computes the *likelihood* of $y_1, y_2, \ldots, y_n$ as a function of all unknown parameters. If the distribution of the response variable is *discrete* then the likelihood is the same as the *probability* of $y_1, y_2, \ldots, y_n$.

Example: Suppose Y_i has a Poisson distribution such that

$$P(Y_i = y) = \frac{\lambda_i^y e^{-\lambda_i}}{y!}$$

where

$$\lambda_i = \exp(\beta_0 + \beta_1 x_i),$$

so that the model for Y_i is a *Poisson regression model*. The likelihood function is then

$$L(\beta_0, \beta_1) = \frac{\lambda_1^{y_1} e^{-\lambda_1}}{y_1!} \times \frac{\lambda_2^{y_2} e^{-\lambda_2}}{y_2!} \times \cdots \times \frac{\lambda_n^{y_n} e^{-\lambda_n}}{y_n!},$$

or

$$L(\beta_0, \beta_1) = \prod_{i=1}^{n} \frac{\lambda_i^{y_i} e^{-\lambda_i}}{y_i!},$$

where, again,

$$\lambda_i = \exp(\beta_0 + \beta_1 x_i).$$

Example: Suppose Y_i has a *binomial* distribution such that

$$P(Y_i = y) = \frac{m_i!}{y_i!(m_i - y_i)!} p_i^y (1 - p_i)^{m_i - y},$$

where

$$p_i = \frac{e^{\beta_0 + \beta_1 x_i}}{1 + e^{\beta_0 + \beta_1 x_i}},$$

so that the model for Y_i is a *logistic regression model*. The likelihood function is then

$$L(\beta_0, \beta_1) = \frac{m_1!}{y_1!(m_1 - y_1)!} p_1^{y_1} (1-p_1)^{m_1 - y_1} \times \frac{m_2!}{y_2!(m_2 - y_2)!} p_2^{y_2} (1-p_2)^{m_2 - y_2} \times \cdots \times \frac{m_n!}{y_n!(m_n - y_n)!} p_n^{y_n} (1-p_n)^{m_n - y_n}$$

or

$$L(\beta_0, \beta_1) = \prod_{i=1}^{n} \frac{m_i!}{y_i!(m_i - y_i)!} p_i^{y_i} (1 - p_i)^{m_i - y_i},$$

where, again,

$$p_i = \frac{e^{\beta_0 + \beta_1 x_i}}{1 + e^{\beta_0 + \beta_1 x_i}}.$$

Note: If the response variable is *discrete* as it is in Poisson and logistic regression, then the likelihood function gives the *probability* of the observed responses as a function of the model parameters.

Example: Suppose Y_i has a normal distribution where the probability density function of Y_i is

$$f(y) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(y-\mu_i)^2}{2\sigma^2}},$$

where

$$\mu_i = \beta_0 + \beta_1 x_i,$$

so that the model for Y_i is a normal linear regression model. The likelihood function is then

$$L(\beta_0, \beta_1, \sigma^2) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(y_i-\mu_i)^2}{2\sigma^2}}.$$

Frequently we use the log of the likelihood function or the *log-likelihood function.*

Example: The log-likelihood for the Poisson regression model above is

$$\log L(\beta_0, \beta_1) = \sum_{i=1}^{n} \log\left(\frac{\lambda_i^{y_i} e^{-\lambda_i}}{y_i!}\right) = \sum_{i=1}^{n} y_i \log(\lambda_i) - \sum_{i=1}^{n} \lambda_i - \sum_{i=1}^{n} \log(y_i!).$$

Example: The log-likelihood for the normal regression model above is

$$\log L(\beta_0, \beta_1, \sigma^2) = -\frac{n}{2}\log(2\pi) - n\log(\sigma) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - \mu_i)^2.$$

Maximum Likelihood Estimation

The maximum likelihood estimates (MLE) of the model parameters is those values of the parameters that *maximize the likelihood of the data* — i.e., the parameter values that make the values of $y_1, y_2, \ldots, y_n$ we observed the most likely values to have occurred. Finding these estimates is a *optimization* problem — i.e., find the values of the parameters that maximize the likelihood (or log-likelihood).

Example: In the normal linear model above, *it can be shown that* the MLEs of β_0 and β_1 are those values that *minimize*

$$\sum_{i=1}^{n}(y_i - \mu_i)^2$$

where $\mu_i = \beta_0 + \beta_1 x_i$. So the MLEs are also the *least squares* estimators. But the MLE of σ^2 is

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$$

where $\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i$. But we typically use the unbiased estimator

$$\hat{\sigma}^2 = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n-p},$$

where p is the number of β_j parameters ($p = 2$ in the model above).

Except in the case of normal linear models and a few very simple nonlinear models, MLEs must be found *numerically* using *iterative* algorithms (similar to those used in nonlinear regression).

Inference Based On Maximum Likelihood

It is usually not possible to determine an *exact* method of estimating the standard error of a MLE, computing a confidence interval based on ML, or conducting a significance test based on ML. However there exist several *asymptotic* methods that give approximate results.

1. Likelihood ratio tests and profile likelihood confidence intervals.
2. Wald tests and confidence intervals.
3. Score (Lagrange multiplier) tests (not discussed).

Likelihood Ratio Tests

The likelihood ratio test statistic is

$$2\log\left(L/L_n\right) = 2\log(L) - 2\log(L_n) \overset{a}{\sim} \chi^2_{(r)},$$

where L and L_n are the likelihood functions for the model and the null model, respectively, evaluated at the MLEs, and r is the degrees of freedom.

Example: Consider the Poisson regression model for `ceriodaphniastrain` and the null hypothesis that there is no difference in the expected number of daphnia between the two strains when controlling for dose.

```
library(trtools) # for ceriodaphniastrain data
ceriodaphniastrain$strain <- factor(ceriodaphniastrain$strain, labels = c("a","b"))

m <- glm(count ~ strain + concentration, family = poisson, data = ceriodaphniastrain)
m.null <- glm(count ~ concentration, family = poisson, data = ceriodaphniastrain)

anova(m.null, m, test = "LRT") # has very little to do with ANOVA
```

```
Analysis of Deviance Table

Model 1: count ~ concentration
Model 2: count ~ strain + concentration
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1        68      119.0
2        67       86.4  1     32.6  1.1e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The "residual deviance" (or just "deviance") is related to the log-likelihood of the model. It is defined as

$$D = 2\log L_s - 2\log L \geq 0,$$

where L and L_s are the likelihoods of the model in question and a "saturated" model, respectively. The saturated model is a "best possible" model where each unique combination of covariate values is a distinct level of a factor. Thus the deviance can be viewed in a way as the "lack of fit" of the model to the data. The residual deviance can sometimes be obtained using the `summary` or `deviance` functions.

```
summary(m)  # shows residual deviance
```

```

Call:
glm(formula = count ~ strain + concentration, family = poisson,
    data = ceriodaphniastrain)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    4.4546     0.0391  113.82  < 2e-16 ***
strainb       -0.2750     0.0484   -5.68  1.3e-08 ***
concentration -1.5431     0.0466  -33.11  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 1359.381  on 69  degrees of freedom
Residual deviance:   86.376  on 67  degrees of freedom
```

```
AIC: 416

Number of Fisher Scoring iterations: 4
```

```
deviance(m) # extracts only residual deviance
```

```
[1] 86.4
```

The likelihood ratio test statistic can be expressed in terms of the (residual) deviance of the model and the null model.

$$\underbrace{2(\log_s - \log L_n)}_{D_n} - \underbrace{2(\log_s - \log L)}_{D} = 2\log L - 2\log L_n \overset{a}{\sim} \chi^2_{(r)},$$

where D and D_n are the deviance of the model and null model, respectively. The degrees of freedom (r) is the *difference* in the residual degrees of freedom between the two models. The residual degrees of freedom equal n minus the number of parameters for the expected response.

The following shows how the deviance relates to what is done by `anova`.

```
anova(m.null, m, test = "LRT")
```

```
Analysis of Deviance Table

Model 1: count ~ concentration
Model 2: count ~ strain + concentration
  Resid. Df Resid. Dev Df Deviance Pr(>Chi)
1        68      119.0
2        67       86.4  1     32.6  1.1e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
deviance(m)      # (residual) deviance of model
```

```
[1] 86.4
```

```
deviance(m.null) # (residual) deviance of null model
```

```
[1] 119
```

```
deviance(m.null) - deviance(m) # test statistic
```

```
[1] 32.6
```

Note: For models where Y_i is assumed to have a normal distribution, residual deviance essentially becomes the residual (error) sums of squares. Some in some sense the "analysis of deviance" can be viewed as a generalization of the "analysis of variance" in linear models.

Profile Likelihood Confidence Intervals

Consider a significance test with hypotheses $H_0 : \theta = \theta_0$ versus $H_a : \theta \neq \theta_0$ where θ is some parameter of the model. A confidence interval (θ_l, θ_h) for θ can be used to conduct the test.

1. If θ_0 is in (θ_l, θ_h), do not reject H_0.
2. If θ_0 is not in (θ_l, θ_h), reject H_0

Thus a confidence interval for θ can be *defined* as *all the values of* θ_0 *that are not rejected by a test* of $H_0 : \theta = \theta_0$ versus $H_a : \theta \neq \theta_0$.

A *profile likelihood confidence interval* is a confidence interval based on the likelihood ratio test. It is the set/range of all values of a parameter that would not be rejected by the likelihood ratio test. **Example**: Consider again the `ceriodaphniastrain` data.

```
confint(m)
```

```
Waiting for profiling to be done...
```

```
              2.5 % 97.5 %
(Intercept)    4.38   4.53
strainb       -0.37  -0.18
concentration -1.63  -1.45
```

For `glm` objects (and some others we will talk about in the future) `confint` will compute profile likelihood confidence intervals by default. A similar approach is used by `nls`, and `glm` when using quasi-likelihood (more on that later), but is based on a F test statistic.

Wald Tests and Confidence Intervals

Wald tests and confidence intervals are based on the fact that MLEs have approximately normal sampling distributions. The Wald *test statistic* for a parameter β_j is

$$z = \frac{\hat{\beta}_j - \beta_j}{\widehat{\text{SE}}(\hat{\beta}_j)} \overset{a}{\sim} \text{N}(0,1) \quad \text{or} \quad z^2 = \left[\frac{\hat{\beta}_j - \beta_j}{\widehat{\text{SE}}(\hat{\beta}_j)}\right]^2 \overset{a}{\sim} \chi^2_{(1)},$$

where β_j is the value hypothesized by the null (frequently $\beta_j = 0$). The Wald confidence interval for β_j is

$$\hat{\beta}_j \pm z \times \widehat{\text{SE}}(\hat{\beta}_j).$$

We can also apply this to a linear combinations of parameters by replacing β_j and $\hat{\beta}_j$ with ℓ and $\hat{\ell}$, respectively.

Wald test statistics and confidence intervals are reported by `contrast`, `lincon`, `glmint`, `nlsint` from the **trtools** package, and the functions in the **emmeans** package. The `summary` function reports Wald test statistics.

Note: In some cases a t-distribution is used as the approximate distribution of a Wald test statistic, and is also used when calibrating a confidence interval. In generalized linear models this usually happens for models in which we need to estimate the *dispersion parameter*. We *do not* need to estimate the dispersion parameter for Poisson or binomial/logistic regression.

Example: Consider again the `ceriodaphniastrain` data.

```
summary(m)$coefficients
```

```
              Estimate Std. Error z value  Pr(>|z|)
(Intercept)      4.455     0.0391  113.82  0.00e+00
strainb         -0.275     0.0484   -5.68  1.31e-08
concentration   -1.543     0.0466  -33.11 2.06e-240
```

```
lincon(m)
```

```
              estimate     se lower upper tvalue  df    pvalue
(Intercept)      4.455 0.0391  4.38  4.53 113.82 Inf  0.00e+00
strainb         -0.275 0.0484 -0.37 -0.18  -5.68 Inf  1.31e-08
concentration   -1.543 0.0466 -1.63 -1.45 -33.11 Inf 2.06e-240
```

```
trtools::contrast(m,
  a = list(strain = "a", concentration = c(0,1,2)),
  b = list(strain = "b", concentration = c(0,1,2)),
  cnames = c("0%","1%","2%"))
```

```
   estimate     se lower upper tvalue  df   pvalue
0%    0.275 0.0484  0.18  0.37   5.68 Inf 1.31e-08
```

```
1%    0.275 0.0484  0.18  0.37   5.68 Inf 1.31e-08
2%    0.275 0.0484  0.18  0.37   5.68 Inf 1.31e-08
```

```
trtools::contrast(m, tf = exp,
  a = list(strain = "a", concentration = c(0,1,2)),
  b = list(strain = "b", concentration = c(0,1,2)),
  cnames = c("0%","1%","2%"))
```

```
   estimate lower upper
0%     1.32   1.2  1.45
1%     1.32   1.2  1.45
2%     1.32   1.2  1.45
```

```
library(emmeans)
pairs(emmeans(m, ~strain|concentration,
  at = list(concentration = c(0,1,2))),
  type = "response", infer = TRUE)
```

```
concentration = 0:
 contrast ratio     SE  df asymp.LCL asymp.UCL null z.ratio p.value
 a / b     1.32 0.0637 Inf       1.2      1.45    1   5.680  <.0001

concentration = 1:
 contrast ratio     SE  df asymp.LCL asymp.UCL null z.ratio p.value
 a / b     1.32 0.0637 Inf       1.2      1.45    1   5.680  <.0001

concentration = 2:
 contrast ratio     SE  df asymp.LCL asymp.UCL null z.ratio p.value
 a / b     1.32 0.0637 Inf       1.2      1.45    1   5.680  <.0001

Confidence level used: 0.95
Intervals are back-transformed from the log scale
Tests are performed on the log scale
```

For Wald tests involving joint hypotheses, you can use the `waldtest` function from the **lmtest** package which works similarly to how the `anova` function can be used for likelihood ratio tests.

```
m <- glm(count ~ strain * concentration, family = poisson, data = ceriodaphniastrain)
summary(m)$coefficients
```

```
                     Estimate Std. Error z value  Pr(>|z|)
(Intercept)             4.481     0.0435  103.01  0.00e+00
strainb                -0.337     0.0670   -5.02  5.11e-07
concentration          -1.598     0.0624  -25.59 1.86e-144
strainb:concentration   0.125     0.0939    1.34  1.82e-01
```

```
m.null <- glm(count ~ concentration, family = poisson, data = ceriodaphniastrain)
summary(m.null)$coefficients
```

```
             Estimate Std. Error z value  Pr(>|z|)
(Intercept)      4.33     0.0331   130.7  0.00e+00
concentration   -1.54     0.0466   -33.1 2.06e-240
```

```
library(lmtest)
waldtest(m.null, m, test = "Chisq")
```

```
Wald test
```

```
Model 1: count ~ concentration
Model 2: count ~ strain * concentration
  Res.Df Df Chisq Pr(>Chisq)
1     68
2     66  2    34    4.1e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Likelihood Ratio Versus Wald

The two methods are equivalent *asymptotically* and tend to produce relatively similar results for larger n, but for smaller n they may produce somewhat different results.

1. Likelihood ratio tests/intervals are generally more accurate than Wald tests/intervals. The latter assumes that the sampling distribution of the estimator is normal. Likelihood ratio tests/intervals assume that the sampling distribution of the likelihood ratio test statistic has a χ^2 distribution. The latter is easier to meet since it does not depend on the parameterization of the model.
2. Likelihood ratio tests/intervals are a bit more computationally expensive and difficult to program in the case of functions of model parameters (e.g., linear combinations where we use `lincon`, `contrast`, or functions from the **emmeans** package).
3. Wald tests/intervals can be used in cases where a likelihood function is not specified because we do not or cannot assume a particular distribution for the response variable.

Assumptions

For inferences based on maximum likelihood to be correct, we are assuming we have the correct likelihood function. It is useful to note three aspects of the likelihood function where we need to be correct.

1. The *distribution* of the response variable.
2. The *relationship* between the parameter(s) of the distribution and the explanatory variable(s).
3. The *independence* of the n observations.

Interestingly, for GLMs the distribution actually isn't something that we need to be correct about, provided that we are correct about the following:

1. $E(Y_i)$ is related to the explanatory variables in the assumed way — e.g., $g[E(Y_i)] = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik}$.
2. $\text{Var}(Y_i)$ depends on $E(Y_i)$ in the assumed away — i.e., through the *variance structure* $\text{Var}(Y_i) = \phi V[E(Y_i)]$.
3. The observations are independent (although there are some special cases where we can violate this assumption and still get "asymptotically correct" inferences).

When you specify a distribution via the `family` argument to the `glm` function, all that matters is the *variance structure* implied by that family.