Stat 8931 (Aster Models)
Lecture Slides Deck 4
Large Sample Theory and Estimating Population Growth Rate

Charles J. Geyer

School of Statistics
University of Minnesota

October 8, 2018

R and License

- The version of R used to make these slides is 3.5.1.
- The version of R package `aster` used to make these slides is 1.0.2.

The Delta Method

The **delta method** is a method (duh!) of deriving the approximate distribution of a nonlinear function of an estimator from the approximate distribution of the estimator itself.

What it does is linearize the nonlinear function. If g is a nonlinear, differentiable vector-to-vector function, the best linear approximation, which is the Taylor series up through linear terms, is

$$g(y) - g(x) \approx \nabla g(x)(y - x),$$

where $\nabla g(x)$ is the matrix of partial derivatives, sometimes called the Jacobian matrix. If $g_i(x)$ denotes the i-th component of the vector $g(x)$, then the (i, j)-th component of the Jacobian matrix is $\partial g_i(x)/\partial x_j$.

The Delta Method (cont.)

The delta method is particularly useful when $\hat{\theta}$ is an estimator and θ is the unknown true (vector) parameter value it estimates, and the delta method says

$$g(\hat{\theta}) - g(\theta) \approx \nabla g(\theta)(\hat{\theta} - \theta)$$

It is not necessary that θ and $g(\theta)$ be vectors of the same dimension. Hence it is not necessary that $\nabla g(\theta)$ be a square matrix.

The Delta Method (cont.)

The delta method gives good or bad approximations depending on whether the spread of the distribution of $\hat{\theta} - \theta$ is small or large compared to the nonlinearity of the function g in the neighborhood of θ.

The Taylor series approximation the delta method uses is a good approximation for sufficiently small values of $\hat{\theta} - \theta$ and a bad approximation for sufficiently large values of $\hat{\theta} - \theta$.

So the overall method is good if those "sufficiently large" values have small probability. And bad otherwise.

The Delta Method (cont.)

As with nearly every application of approximation in statistics, we rarely (if ever) do the (very difficult) analysis to know whether the approximation is good or bad.

We just use the delta method and hope it gives good results.

If we are really worried, we can check it using simulation (also called the **parametric bootstrap**). This method will be illustrated in Deck 7 of the course slides.

The Delta Method (cont.)

The delta method is particularly easy to use when the distribution of $\hat{\theta} - \theta$ is multivariate normal, exactly or approximately.

If it is only approximately normal, then this is another approximation in addition to the Taylor series approximation.

The reason this is easy is that normal distributions are determined by their mean vector and variance matrix, and there is a theorem which gives the mean vector and variance matrix of a linear function of a random vector.

The Delta Method (cont.)

Theorem

Suppose X is a random vector, a is a nonrandom vector, and B is a nonrandom matrix such that $a + BX$ makes sense (because a, B, and X have dimensions such that the indicated vector addition and matrix-vector multiplication are defined). Then

$$E(a + bX) = a + BE(X)$$
$$\text{var}(a + bX) = B \,\text{var}(X) B^T$$

A proof is given on slides 64–67 of deck 2 of my Stat 5101 course slides.

Another way to say this is that if $E(X) = \mu$ and $\text{var}(X) = V$, then

$$E(a + bX) = a + B\mu$$
$$\text{var}(a + bX) = BVB^T$$

The Delta Method (cont.)

So suppose $\hat{\theta}$ is normal with mean vector θ and variance matrix V, and write $B = \nabla g(\theta)$, then $\hat{\theta} - \theta$ has mean vector 0 and variance matrix V, and

$$E\{g(\hat{\theta}) - g(\theta)\} \approx 0$$
$$\text{var}\{g(\hat{\theta}) - g(\theta)\} \approx BVB^T$$

The Delta Method (cont.)

The Delta Method for Approximately Normal Estimators. Suppose $\hat{\theta}$ is approximately normal with mean vector θ and variance matrix $V(\theta)$. Suppose g is a vector-to-vector function with derivative $\nabla g(\theta) = B(\theta)$. Then $g(\hat{\theta})$ is approximately normal with mean vector $g(\theta)$ and variance matrix $B(\theta)V(\theta)B(\theta)^T$.

The Delta Method (cont.)

An approximate confidence region for $g(\theta)$ is centered at $g(\hat{\theta})$ and has extent determined by $B(\theta)V(\theta)B(\theta)^T$. But we do not know that because we do not know θ (the true unknown parameter value).

Thus we make a last approximation and **plug-in** $\hat{\theta}$ for θ in the variance and use $B(\hat{\theta})V(\hat{\theta})B(\hat{\theta})^T$.

This is known as the **plug-in principle**.

(For the statisticians in the audience, it is an application of Slutsky's theorem.)

The Delta Method (cont.)

Recall from deck 2 of these slides that the maximum likelihood estimator in an unconditional canonical affine submodel of an aster model can be written

$$\hat{\beta} = h^{-1}(M^T y)$$

where h is the transformation from canonical to mean value parameters given by

$$h(\beta) = \nabla c_{\text{sub}}(\beta) = M^T \nabla c(a + M\beta)$$

and has derivative

$$\nabla h(\beta) = \nabla^2 c_{\text{sub}}(\beta) = M^T \nabla^2 c(a + M\beta) M$$

The Delta Method (cont.)

And by the inverse function theorem of real analysis, the derivative of the inverse function is the (matrix) inverse of the derivative of the forward function

$$\nabla h^{-1}(\tau) = \left[\nabla h(\beta)\right]^{-1}, \qquad \text{when } \tau = h(\beta) \text{ and } \beta = h^{-1}(\tau).$$

Fisher Information

The matrix that appeared in the derivative of the canonical-to-mean-value parameter map plays a very important role in likelihood inference.

The **observed Fisher information matrix** is minus the second derivative matrix of the log likelihood.

The **expected Fisher information matrix** is the expectation of the observed Fisher information matrix.

Fisher Information (cont.)

What Fisher information is depends on what the parameter is (what you are differentiating with respect to).

It also depends on what the model is (what the log likelihood is).

Thus, to be pedantically correct, we need decoration to indicate

- observed or expected,
- the model, and
- the parameter

Sometimes we are not so fussy and let the context indicate what we mean.

Fisher Information (cont.)

For log likelihood l for parameter φ, observed Fisher information (for this model and parameter) is

$$I_{\text{obs}}(\varphi) = -\nabla^2 l(\varphi)$$

and expected Fisher information (for this model and parameter) is

$$I_{\text{exp}}(\varphi) = E_\varphi\{I_{\text{obs}}(\varphi)\} = E_\varphi\{-\nabla^2 l(\varphi)\}$$

Fisher Information (cont.)

If this is the log likelihood for a regular full exponential family

$$l(\varphi) = \langle y, \varphi \rangle - c(\varphi),$$

then

$$I_{\text{obs}}(\varphi) = -\nabla^2 l(\varphi) = \nabla^2 c(\varphi)$$

and since this is a nonrandom quantity, it is its own expectation (expectation of a constant is that constant), so

$$I_{\text{exp}}(\varphi) = \nabla^2 c(\varphi)$$

too.

Fisher Information (cont.)

Thus for a regular full exponential family, in general, and for saturated aster models and their unconditional canonical affine submodels, in particular, there is no difference between observed and expected Fisher information for the unconditional canonical parameter, and we can just write

$$I(\varphi) = \nabla^2 c(\varphi)$$

Fisher Information (cont.)

But even restricting to Fisher information for the unconditional canonical parameter, we distinguish Fisher information for saturated models and canonical affine submodels

$$
\begin{aligned}
I_{\mathsf{sat}}(\varphi) &= \nabla^2 c(\varphi) \\
I_{\mathsf{sub}}(\beta) &= \nabla^2 c_{\mathsf{sub}}(\beta) \\
&= M^T \nabla^2 c(a + M\beta) M
\end{aligned}
$$

Fisher Information (cont.)

To figure out Fisher information for other parameters, there are two ways to go:

- Write the log likelihood in terms of the new parameter, differentiate it twice, negate it, and take an expectation, if expected Fisher information is wanted.
- Prove a theorem about how Fisher information transforms under change-of-parameter.

(The latter is just the former done abstractly and once and for all, rather than concretely and repeated for each problem.)

Fisher Information Transforms by Covariance

If ψ is another parameter, then

$$\frac{\partial I(\psi)}{\partial \psi_i} = \sum_k \frac{\partial I(\varphi)}{\partial \varphi_k} \frac{\partial \varphi_k}{\partial \psi_i}$$

(the multivariable chain rule), and

$$\frac{\partial^2 I(\psi)}{\partial \psi_i \partial \psi_j} = \sum_k \sum_l \frac{\partial^2 I(\varphi)}{\partial \varphi_k \partial \varphi_l} \frac{\partial \varphi_k}{\partial \psi_i} \frac{\partial \varphi_l}{\partial \psi_j} + \sum_k \frac{\partial I(\varphi)}{\partial \varphi_k} \frac{\partial^2 \varphi_k}{\partial \psi_i \partial \psi_j}$$

This is somewhat ugly. But if we plug in the MLE for φ, the second term is zero because $\nabla I(\hat{\varphi}) = 0$ (the first derivative is zero at the maximum). The second term also goes away for expected Fisher information because $E_\varphi\{\nabla I(\varphi)\} = 0$ by a differentiation under the integral sign argument proved in theoretical statistics courses (slides 33–35 and 86 of my 5102 course slides).

Fisher Information Transforms by Covariance (cont.)

This gives the tranformation rules

$$I_{\text{exp},\psi}(\psi) = B(\psi)^T I_{\text{exp},\varphi}(\varphi) B(\psi)$$

where

$$\varphi = h(\psi)$$
$$B(\psi) = \nabla h(\psi)$$

and

$$I_{\text{obs},\psi}(\hat{\psi}) = B(\hat{\psi})^T I_{\text{obs},\varphi}(\hat{\varphi}) B(\hat{\psi})$$

with the same conditions and $\hat{\varphi} = h(\hat{\psi})$.

Fisher Information and MLE

The so-called "usual" asymptotics of maximum likelihood says the asymptotic (large sample, approximate) distribution of the MLE is normal with mean vector the true unknown parameter value and variance inverse Fisher information (either observed or expected, but for that particular model and parameter).

For regular full exponential families, this is an application of the delta method.

Fisher Information and MLE (cont.)

Recall again (from just before we started talking about Fisher information) for a unconditional canonical affine submodel of an aster model

$$\hat{\beta} = h^{-1}(M^T y)$$

where

$$h(\beta) = \nabla c_{\text{sub}}(\beta) = M^T \nabla c(a + M\beta)$$
$$\nabla h(\beta) = \nabla^2 c_{\text{sub}}(\beta) = M^T \nabla c(a + M\beta) M$$

and

$$\nabla h^{-1}(\tau) = \bigl[\nabla h(\beta)\bigr]^{-1}, \qquad \text{when } \tau = h(\beta) \text{ and } \beta = h^{-1}(\tau).$$

Fisher Information and MLE (cont.)

The mean vector and variance matrix of the submodel canonical statistic

$$E\{M^T y\} = M^T \mu$$
$$\text{var}\{M^T y\} = M^T \nabla^2 c(a + M\beta) M = I(\beta)$$

(the latter is the submodel Fisher information matrix for β).

Assume (more on this later) that the distribution is approximately multivariate normal with this mean vector and variance matrix.

Fisher Information and MLE (cont.)

Recognize that in the change of parameter $\tau \longleftrightarrow \beta$ we have

$$\nabla h(\beta) = I(\beta)$$
$$\nabla h^{-1}(\tau) = I(\beta)^{-1}$$

(when $\tau = h(\beta)$ and $\beta = h^{-1}(\tau)$).

And recognize $\hat{\tau} = M^T y$ (observed equals expected).

So the delta method says the approximate normal distribution of $\hat{\beta} = h^{-1}(\hat{\tau})$ has mean vector

$$\beta = h^{-1}(\tau) = h^{-1}(M^T \mu)$$

and variance matrix

$$\left[\nabla h^{-1}(\tau)\right] \operatorname{var}(\hat{\tau}) \left[\nabla h^{-1}(\tau)\right] = I(\beta)^{-1} I(\beta) I(\beta)^{-1} = I(\beta)^{-1}$$

Asymptotics

Suppose X_1, X_2, ... is an infinite sequence of IID random vectors, each having mean vector μ and variance matrix V. Define

$$\overline{X}_n = \frac{1}{n} \sum_{i=1}^{n} X_i.$$

Then

$$\sqrt{n}(\overline{X}_n - \mu) \xrightarrow{\mathcal{D}} \text{Normal}(0, V). \tag{1}$$

Statement (1) is called the **central limit theorem** (CLT).

The type of convergence it uses is called **convergence in distribution**. It means the distribution of the random vector on the left-hand side (which is a different distribution for each n) gets closer and closer to the distribution on the right-hand side as n goes to infinity.

Asymptotics (cont.)

The sense in which it gets "closer and closer" we leave vague. Roughly speaking, probabilities and expectations "nice enough" events and random variables calculated with respect to the distribution of the left-hand side (which is a different distribution for each n) get closer and closer to the corresponding probabilities and expectations calculated with respect to the distribution of the right-hand side.

The story about n going to infinity is not really interesting.

In practice we have a distribution we want to calculate, the exact distribution of some random quantity for the actual n of the data we are analyzing. But this is too hard, so we use the asymptotic approximation. But the actual n of our actual data is not going to infinity or anywhere else.

Asymptotics (cont.)

Asymptotic approximation does not approximate all probabilities and expectations with the same accuracy at the same n. It may be fairly good for some and very poor for others.

The accuracy of asymptotic approximation is absolute not relative. An asymptotic approximation $P = 1.35 \times 10^{-9}$ has good absolute accuracy if the true probability being approximation is any fairly small number, perhaps $P = 0.001$.

Asymptotics (cont.)

Despite its limitations, asymptotic approximation is the only game in town. For all models more complicated than LM, including GLM and aster models, no exact sampling distributions are available.

We have to use asymptotic approximation even though we have no idea what its accuracy is.

We will later learn (deck 7 of these slides) about a very time-consuming method called the **parametric bootstrap** which tells us something about how well asymptotics work but it itself justified asymptotically.

Most researchers do not use the parametric bootstrap routinely. Many never use it.

Asymptotics Summary

The two main tools of asymptotic approximation are the central limit theorem and the delta method. For an unconditional aster model they say $\hat{\beta}$ has approximately the multivariate normal distribution with mean β (the true unknown parameter value) and variance $I(\hat{\beta})^{-1}$ (inverse Fisher information).

If worried about whether the asymptotic approximation is good, parametric bootstrap.

The same recipe works for any other parameter, just replace β everywhere with ξ or whatever, but one has to use inverse Fisher information *for that parameter*.

Asymptotics Summary (cont.)

Alternatively, one can use the delta method again.

If $\psi = g(\beta)$ for any differentiable function g, and $G(\beta) = \nabla g(\beta)$ and $\hat{\psi} = g(\hat{\beta})$, then $\hat{\psi}$ has approximately the multivariate normal distribution with mean ψ (the true unknown parameter value) and variance

$$G(\hat{\beta}) I(\hat{\beta})^{-1} G(\hat{\beta})^T$$

This is the method actually used by the R functions `predict.aster` and `predict.aster.formula`.

If worried about whether the asymptotic approximation is good, parametric bootstrap.

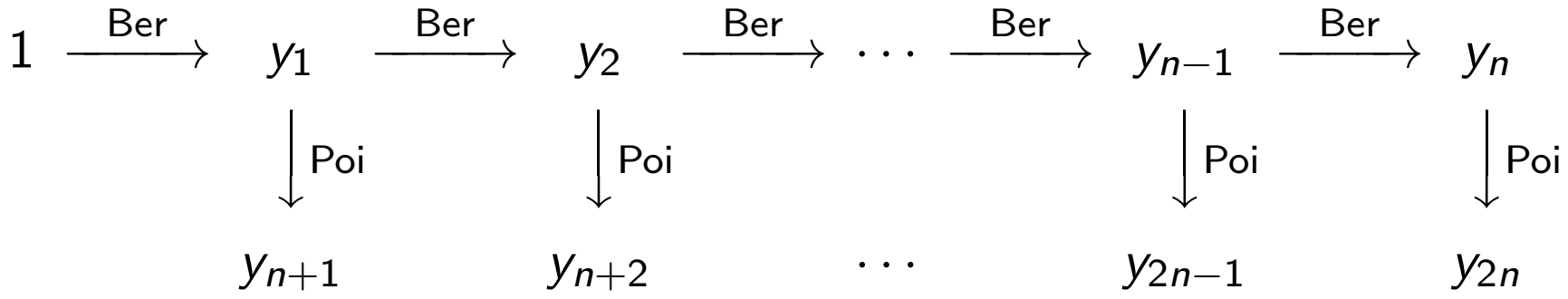
The Theory of Population Growth Rates

Example 1 in the second paper about aster models (Shaw, Geyer, Wagenius, Hangelbroek and Etterson, *American Naturalist*, 2008) is about estimating population growth rate.

This theory has its origins in books by Lotka (1925) and Fisher (1930) and papers by Leslie (1945), but we were following a paper by Lenski and Service (*Ecology*, 1982) that proposed using a statistical technique called the "jackknife" to estimate standard errors for estimates of the population growth rate. They in turn cite a paper by Goodman (1968) for derivation of the theory, so that is where the theory presented here comes from.

The Theory of Population Growth Rates (cont.)

Suppose we have a graph



where the dots indicate more of the same. The variables in the first row are survival indicators and the variables in the second row are offspring counts (we could more layers between survival and offspring counts).

The Theory of Population Growth Rates (cont.)

Write

$$\xi_j^* = \xi_{n+j}, \qquad j = 1, \ldots, n$$

We consider j to index age classes rather than time. (Individuals are born at different times, but y_j and y_{n+j} refer to survival and fecundity, respectively, at age j for all individuals.)

So

- ξ_j is the conditional probability that an individual alive at age $j-1$ survives to age j, and
- ξ_j^* is the conditionally expected number of offspring produced at age j given survival to age j.

The Theory of Population Growth Rates (cont.)

We introduce the same convention for unconditional mean values

$$\mu_j^* = \mu_{n+j}, \qquad j = 1, \ldots, n$$

The relation between conditional and unconditional mean values is

$$\mu_j = \prod_{k=1}^{j} \xi_k$$

and

$$\mu_j^* = \mu_j \xi_j^*$$

The Theory of Population Growth Rates (cont.)

We want to consider a population of individuals undergoing exponential growth. Let $\pi_j(t)$ denote the expected number of individuals of age j at time t, and suppose that we only observe at times t spaced in the same way as the age classes. Then we have

$$\pi_j(t) = \pi_{j-1}(t-1)\xi_j^*, \qquad j = 1, \ldots, n \tag{2a}$$

with

$$\pi_0(t) = \sum_{j=1}^{n} \pi_j(t-1)\xi_j^* \tag{2b}$$

being the number of individuals born at time t (all offspring when just born go into age class zero where they have no past mortality).

All of this is a bit crude, ignoring the variations of age within an age class, but it is the basis of a huge literature.

The Theory of Population Growth Rates (cont.)

The exponential growth assumption is

$$\pi_j(t) = \pi_j \lambda^t, \quad \text{for all } j \text{ and } t \tag{3}$$

where π_j is simplified notation for $\pi_j(0)$.

This too is an oversimplification. It can be shown that if we do not introduce this assumption that we will have

$$\pi_j(t) \approx \pi_j \lambda^t$$

for very large t, where now π_j is not $\pi_j(0)$. Rather π_j are components of the eigenvector of the "Leslie matrix" corresponding to the largest eigenvalue, which is λ (with the eigenvector normalized to make this equation work).

The Theory of Population Growth Rates (cont.)

But exponential growth always comes up against resource limits and stops, so the value of this asymptotic theory is limited and unlike statistical asymptotic theory (the central limit theorem and the delta method) there is no method analogous to the parametric bootstrap of checking validity of exponential growth. So we will just assume it.

The Theory of Population Growth Rates (cont.)

$$\pi_j(t) = \pi_{j-1}(t-1)\xi_j \tag{2a}$$

Iterating (2a) we get

$$\begin{aligned}
\pi_j(t) &= \pi_{j-1}(t-1)\xi_j \\
&= \pi_{j-2}(t-2)\xi_{j-1}\xi_j \\
&= \pi_{j-3}(t-3)\xi_{j-2}\xi_{j-1}\xi_j \\
&\vdots \\
&= \pi_0(t-j)\prod_{k=1}^{j}\xi_k \\
&= \pi_0(t-j)\mu_j
\end{aligned} \tag{4}$$

The Theory of Population Growth Rates (cont.)

$$\pi_j(t) = \pi_j \lambda^t \tag{3}$$

$$\pi_j(t) = \mu_j \pi_0(t-j) \tag{4}$$

From the exponential growth assumption (3) we get

$$\frac{\pi_0(t)}{\pi_0(t-j)} = \lambda^j \tag{5}$$

Hence, combining (4) and (5), we get

$$\frac{\pi_j(t)}{\pi_0(t)} = \frac{\mu_j \pi_0(t-j)}{\pi_0(t)} = \frac{\mu_j}{\lambda^j} \tag{6}$$

The Theory of Population Growth Rates (cont.)

$$\frac{\pi_j(t)}{\pi_0(t)} = \frac{\mu_j}{\lambda^j} \tag{6}$$

Define

$$\nu(t) = \sum_{j=0}^{n} \pi_j(t)$$

(the total population size at time t). Combining this with (6) gives

$$\frac{\nu(t)}{\pi_0(t)} = \sum_{j=0}^{n} \frac{\pi_j(t)}{\pi_0(t)} = \sum_{j=0}^{n} \frac{\mu_j}{\lambda^j} \tag{7}$$

Divide (7) into (6) to get

$$\frac{\pi_j(t)}{\nu(t)} = \frac{\mu_j \lambda^{-j}}{\sum_{k=1}^{n} \mu_k \lambda^{-k}}$$

The Theory of Population Growth Rates (cont.)

$$\pi_0(t) = \sum_{j=1}^{n} \pi_j(t-1)\xi_j^* \tag{2b}$$

$$\pi_j(t) = \mu_j \pi_0(t-j) \tag{4}$$

Combine (2b) and (4) to get

$$\pi_0(t) = \sum_{j=1}^{n} \mu_j \pi_0(t-j-1)\xi_j^*$$

Plugging in $\mu_j^* = \mu_j \xi_j^*$ (relation between conditional and unconditional means) and dividing through by $\pi_0(t)$, this becomes

$$1 = \sum_{j=1}^{n} \mu_j^* \frac{\pi_0(t-j-1)}{\pi_0(t)} \tag{8}$$

The Theory of Population Growth Rates (cont.)

$$\frac{\pi_0(t)}{\pi_0(t-j)} = \lambda^j \tag{5}$$

$$1 = \sum_{j=1}^{n} \mu_j^* \frac{\pi_0(t-j-1)}{\pi_0(t)} \tag{8}$$

Combining (5) and (8) we get

$$1 = \sum_{j=1}^{n} \mu_j^* \lambda^{-(j+1)} \tag{$\star$}$$

We have arrived at the *stable age equation*, equation (27) in Goodman (*Demography*, 1968), equation (1) in Lenski and Service (*Ecology*, 1982) and equation (5.1) in the technical report (U. of M. School of Statistics TR 658) that does the calculations for the paper Shaw, Geyer, Wagenius, Hangelbroek and Etterson (*American Naturalist*, 2008).

The Theory of Population Growth Rates (cont.)

$$1 = \sum_{j=1}^{n} \mu_j^* \lambda^{-(j+1)} \qquad (\star)$$

As $\lambda \to 0$ the right-hand side of $(\star)$ goes to ∞.

As $\lambda \to \infty$ the right-hand side of $(\star)$ goes to zero.

Moreover the right-hand side of $(\star)$ is a strictly decreasing function of λ.

Hence $(\star)$ always has exactly one solution for λ (thought of as a function of $\mu_1^*, \ldots, \mu_n^*$).

The Theory of Population Growth Rates (cont.)

$$1 = \sum_{j=1}^{n} \mu_j^* \lambda^{-(j+1)} \qquad (\star)$$

Equation $(\star)$ is the key to all life history analysis that uses population growth rates.

Life history analyses that do not involve the population growth rate λ or subpopulation growth rates (with a different λ for each subpopulation) do not need $(\star)$. Those that do, do.

By $(\star)$ the population growth rate λ is an implicitly defined function of mean value parameters. Aster can estimate the latter, so between aster and $(\star)$ we can estimate λ.

Aphids

The data used by Lenski and Service (*Ecology*, 1982) and by Shaw, Geyer, Wagenius, Hangelbroek and Etterson (*American Naturalist*, 2008, Example 1) is on the brown ambrosia aphid *Uroleucon rudbeckiae*.

```
> library(aster)
> data(aphid)
> class(aphid)

[1] "data.frame"

> names(aphid)

[1] "root" "varb" "resp" "id"
```

We see this is a "long format" data set (as the help page says). No `reshape` necessary.

Aphids (cont.)

```
> levels(aphid$varb)

 [1] "B2"  "B3"  "B4"  "B5"  "B6"  "B7"  "B8"  "B9"
 [9] "S1"  "S10" "S11" "S12" "S13" "S2"  "S3"  "S4"
[17] "S5"  "S6"  "S7"  "S8"  "S9"
```

The "Sx" variables are the survival variables and the "Bx" variables are the fecundity variables.

Note that B1, B10 B11 B12, and B13 are missing. The reason for this is that they were zero for all individuals and would have caused problems for the models fit by Shaw, Geyer, Wagenius, Hangelbroek and Etterson (*American Naturalist*, 2008, Example 1). (More on this later.)

Aphids (cont.)

For no particular reason, partly because the *American Naturalist* paper fit conditional aster models and we haven't covered them yet and don't want to stop to introduce them, but also partly just because we can, we are going fit different models.

So we are going to restore the data missing from this dataset.

```
> aphid.wide <- reshape(aphid, direction = "wide",
+     timevar = "varb", v.names = "resp")
> head(names(aphid.wide))
```

```
[1] "root"    "id"      "resp.S1" "resp.S2" "resp.B2"
[6] "resp.S3"
```

```
> fred <- sub("resp.", "", names(aphid.wide))
> names(aphid.wide) <- fred
```

Aphids (cont.)

```
> z <- rep(0, nrow(aphid.wide))
> aphid.wide <- transform(aphid.wide, B1 = z, B10 = z,
+     B11 = z, B12 = z, B13 = z)
> aphid.wide$id <- NULL
> names(aphid.wide)

 [1] "root" "S1"   "S2"   "B2"   "S3"   "B3"   "S4"
 [8] "B4"   "S5"   "B5"   "S6"   "B6"   "S7"   "B7"
[15] "S8"   "B8"   "S9"   "B9"   "S10"  "S11"  "S12"
[22] "S13"  "B1"   "B10"  "B11"  "B12"  "B13"
```

Aphids (cont.)

Now we do the graph,

```
> vars <- outer(c("S", "B"), 1:13, paste, sep = "")
> vars <- as.vector(t(vars))
> vars

 [1] "S1"  "S2"  "S3"  "S4"  "S5"  "S6"  "S7"  "S8"
 [9] "S9"  "S10" "S11" "S12" "S13" "B1"  "B2"  "B3"
[17] "B4"  "B5"  "B6"  "B7"  "B8"  "B9"  "B10" "B11"
[25] "B12" "B13"

> pred <- c(0:12, 1:13)
> fam <- rep(1:2, each = 13)
```

Aphids (cont.)

And check that the graph is what we want

```
> foo <- rbind(vars, c("initial", vars)[pred + 1])
> rownames(foo) <- c("successor", "predecessor")
> foo
```

```
             [,1]      [,2] [,3] [,4] [,5] [,6] [,7]
successor   "S1"      "S2" "S3" "S4" "S5" "S6" "S7"
predecessor "initial" "S1" "S2" "S3" "S4" "S5" "S6"
             [,8] [,9] [,10] [,11] [,12] [,13] [,14]
successor   "S8" "S9" "S10" "S11" "S12" "S13" "B1"
predecessor "S7" "S8" "S9"  "S10" "S11" "S12" "S1"
             [,15] [,16] [,17] [,18] [,19] [,20] [,21]
successor   "B2"  "B3"  "B4"  "B5"  "B6"  "B7"  "B8"
predecessor "S2"  "S3"  "S4"  "S5"  "S6"  "S7"  "S8"
             [,22] [,23] [,24] [,25] [,26]
successor   "B9"  "B10" "B11" "B12" "B13"
predecessor "S9"  "S10" "S11" "S12" "S13"
```

Aphids (cont.)

And check that the graph is what we want

```
> rbind(vars, fam)
```

```
     [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
vars "S1" "S2" "S3" "S4" "S5" "S6" "S7" "S8" "S9"
fam  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "1"  "1"
     [,10] [,11] [,12] [,13] [,14] [,15] [,16] [,17]
vars "S10" "S11" "S12" "S13" "B1"  "B2"  "B3"  "B4"
fam  "1"   "1"   "1"   "1"   "2"   "2"   "2"   "2"
     [,18] [,19] [,20] [,21] [,22] [,23] [,24] [,25]
vars "B5"  "B6"  "B7"  "B8"  "B9"  "B10" "B11" "B12"
fam  "2"   "2"   "2"   "2"   "2"   "2"   "2"   "2"
     [,26]
vars "B13"
fam  "2"
```

Aphids (cont.)

```
> redata <- reshape(aphid.wide, varying = list(vars),
+     direction = "long", timevar = "varb",
+     times = as.factor(vars), v.names = "resp")
> fred <- as.character(redata$varb)
> surv <- as.numeric(grepl("S", fred))
> fecund <- as.numeric(grepl("B", fred))
> age <- as.numeric(sub("[SB]", "", fred))
> redata <- transform(redata, surv = surv,
+     fecund = fecund, age = age)
> names(redata)

[1] "root"   "varb"   "resp"   "id"     "surv"
[6] "fecund" "age"
```

Aphids (cont.)

Now we are going to fit some aster models without `varb`. We are still going to obey the "no naked predictors" dictum by "interacting" every predictor with either `surv`, which indicates the "Sx" nodes of the graph, or `fecund`, which indicates the "Bx" nodes of the graph.

Aphids (cont.)

```
> aout.0.0 <- aster(resp ~ 0 + surv + fecund,
+     pred, fam, varb, id, root, data = redata)
> summary(aout.0.0)

Call:
aster.formula(formula = resp ~ 0 + surv + fecund, pred = pr
    fam = fam, varvar = varb, idvar = id, root = root, data

       Estimate Std. Error z value Pr(>|z|)
surv   -0.20297    0.13828  -1.468    0.142
fecund  0.62690    0.06757   9.277   <2e-16 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Aphids (cont.)

We try polynomial functions of age. Since we know that fecundity is low (actually zero observed fecundity) at both ends, we only try even degree polynomials for that. Theory says survival, should, perhaps, go the same way, but that is less clear, especially in a laboratory experiment.

```
> aout.0.2 <- aster(resp ~ 0 + surv + fecund +
+     fecund : poly(age, d=2),
+     pred, fam, varb, id, root, data = redata)
> summary(aout.0.2)

Call:
aster.formula(formula = resp ~ 0 + surv + fecund + fecund:poly(age,
    d = 2), pred = pred, fam = fam, varvar = varb, idvar = id,
    root = root, data = redata)

                           Estimate Std. Error z value Pr(>|z|)
surv                         0.3288     0.2258   1.456    0.145
fecund                      -1.9329     0.4523  -4.273 1.93e-05 ***
fecund:poly(age, d = 2)1 -60.6462    10.4293  -5.815 6.06e-09 ***
fecund:poly(age, d = 2)2 -41.8304     6.0271  -6.940 3.91e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Aphids (cont.)

```
> aout.0.4 <- aster(resp ~ 0 + surv + fecund +
+     fecund : poly(age, d=4),
+     pred, fam, varb, id, root, data = redata)
> summary(aout.0.4)

Call:
aster.formula(formula = resp ~ 0 + surv + fecund + fecund:poly(age,
    d = 4), pred = pred, fam = fam, varvar = varb, idvar = id,
    root = root, data = redata)

                             Estimate Std. Error z value Pr(>|z|)
surv                           0.2262     0.2118   1.068  0.28560
fecund                        -6.9527     3.4996  -1.987  0.04695 *
fecund:poly(age, d = 4)1    -207.2506   101.9484  -2.033  0.04206 *
fecund:poly(age, d = 4)2    -189.5867    84.4659  -2.245  0.02480 *
fecund:poly(age, d = 4)3     -83.1739    43.3301  -1.920  0.05492 .
fecund:poly(age, d = 4)4     -45.4436    15.9402  -2.851  0.00436 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Aphids (cont.)

```
> aout.0.6 <- aster(resp ~ 0 + surv + fecund +
+     fecund : poly(age, d=6),
+     pred, fam, varb, id, root, data = redata)
> aout.0.8 <- aster(resp ~ 0 + surv + fecund +
+     fecund : poly(age, d=8),
+     pred, fam, varb, id, root, data = redata,
+     maxiter = 5000)
```

(We had to add the argument `maxiter = 5000` because otherwise we got a warning "did not converge" in the default for `maxiter`.)

Aphids (cont.)

```
> anova(aout.0.0, aout.0.2, aout.0.4, aout.0.6, aout.0.8)

Analysis of Deviance Table

Model 1: resp ~ 0 + surv + fecund
Model 2: resp ~ 0 + surv + fecund + fecund:poly(age, d = 2)
Model 3: resp ~ 0 + surv + fecund + fecund:poly(age, d = 4)
Model 4: resp ~ 0 + surv + fecund + fecund:poly(age, d = 6)
Model 5: resp ~ 0 + surv + fecund + fecund:poly(age, d = 8)
  Model Df Model Dev Df Deviance P(>|Chi|)
1        2   -260.72
2        4   -152.95  2  107.772 < 2.2e-16 ***
3        6   -120.69  2   32.262 9.874e-08 ***
4        8   -117.69  2    2.992    0.2241
5       10   -116.51  2    1.182    0.5537
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Aphids (cont.)

Stop at degree = 4 for `fecund`.

```
> aout.1.4 <- aster(resp ~ 0 + surv + fecund +
+     surv : age + fecund : poly(age, d=4),
+     pred, fam, varb, id, root, data = redata)
> aout.2.4 <- aster(resp ~ 0 + surv + fecund +
+     surv : poly(age, d=2) + fecund : poly(age, d=4),
+     pred, fam, varb, id, root, data = redata)
> aout.3.4 <- aster(resp ~ 0 + surv + fecund +
+     surv : poly(age, d=3) + fecund : poly(age, d=4),
+     pred, fam, varb, id, root, data = redata)
> aout.4.4 <- aster(resp ~ 0 + surv + fecund +
+     surv : poly(age, d=4) + fecund : poly(age, d=4),
+     pred, fam, varb, id, root, data = redata)
```

Aphids (cont.)

```
> anova(aout.0.4, aout.1.4, aout.2.4, aout.3.4, aout.4.4)

Analysis of Deviance Table

Model 1: resp ~ 0 + surv + fecund + fecund:poly(age, d = 4)
Model 2: resp ~ 0 + surv + fecund + surv:age + fecund:poly(
Model 3: c("resp ~ 0 + surv + fecund + surv:poly(age, d = 2
Model 4: c("resp ~ 0 + surv + fecund + surv:poly(age, d = 3
Model 5: c("resp ~ 0 + surv + fecund + surv:poly(age, d = 4
  Model Df Model Dev Df Deviance P(>|Chi|)
1        6  -120.686
2        7   -81.540  1   39.146 3.933e-10 ***
3        8   -76.683  1    4.857   0.02754 *
4        9   -74.540  1    2.143   0.14327
5       10   -74.190  1    0.350   0.55392
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Aphids (cont.)

Looks like `aout.2.4` is the best model.

Exercise for the reader. The formula

```
resp ~ poly(age, d=2) + fecund + fecund:poly(age, d=4)
```

looks like it violates our "no naked predictors" dictum, but it fits exactly the same model as `aout.2.4`. Why?

Thus this really illustrates the "canonical parameters are meaningless quantities" dictum.

Aphids (cont.)

Now we need mean value parameters for a typical individual, so we use `predict.aster.formula` on `newdata` for one individual

```
> renewdata <- subset(redata, id == 1)
> pout <- predict(aout.2.4, varvar = varb, idvar = id,
+     root = root, newdata = renewdata)
> names(pout) <- as.character(renewdata$varb)
> round(pout, 3)
```

```
   S1    S2    S3    S4    S5    S6    S7    S8    S9
0.798 0.794 0.794 0.792 0.778 0.733 0.637 0.473 0.306
  S10   S11   S12   S13    B1    B2    B3    B4    B5
0.191 0.113 0.061 0.031 0.026 0.982 3.234 3.188 2.132
   B6    B7    B8    B9   B10   B11   B12   B13
1.412 0.872 0.298 0.024 0.000 0.000 0.000 0.000
```

Aphids (cont.)

We only want the "Bx" components of the mean value parameter vector

```
> mu.star <- pout[grepl("B", names(pout))]
> round(mu.star, 3)

   B1    B2    B3    B4    B5    B6    B7    B8    B9
0.026 0.982 3.234 3.188 2.132 1.412 0.872 0.298 0.024
  B10   B11   B12   B13
0.000 0.000 0.000 0.000

> sally <- function(lambda)
+     1 - sum(mu.star / lambda^(1 + seq(along = mu.star)))
```

Aphids (cont.)

```
> sally(1)

[1] -11.16667

> sally(2)

[1] 0.5208323

> lout <- uniroot(sally, lower = 1, upper = 2)
> names(lout)

[1] "root"       "f.root"     "iter"       "init.it"
[5] "estim.prec"

> lout$root

[1] 1.683472

> lout$estim.prec

[1] 6.103516e-05
```

Aphids (cont.)

```
> lambda.hat <- lout$root
> lambda.hat

[1] 1.683472
```

Jacobian Matrix for Lambda as a Function of Mu

$$1 = \sum_{j=1}^{n} \mu_j^* \lambda^{-(j+1)} \qquad (\star)$$

Differentiating $(\star)$ with respect to μ_k^* gives

$$\begin{aligned} 0 &= \lambda^{-k-1} + \sum_{j=1}^{n} \mu_j^* (-j-1)\lambda^{-j-2} \frac{\partial \lambda}{\partial \mu_k^*} \\ &= \lambda^{-k-1} - \frac{\partial \lambda}{\partial \mu_k^*} \sum_{j=1}^{n} (j+1)\mu_j^* \lambda^{-j-2} \end{aligned}$$

$$\frac{\partial \lambda}{\partial \mu_k^*} = \frac{\lambda^{-k-1}}{\sum_{j=1}^{n} (j+1)\mu_j^* \lambda^{-j-2}} \qquad (\star\star)$$

Aphids (cont.)

Now we are ready to apply the delta method "by hand" (using R but not having everything done for us by some R function).

First we need the Jacobian of the map $\beta \to \mu$ which is the `gradient` component of the object returned by `predict.aster.formula` when the optional argument `gradient = TRUE` is given.

```
> pout <- predict(aout.2.4, varvar = varb, idvar = id,
+     root = root, newdata = renewdata, gradient = TRUE)
> dim(pout$gradient)
```

```
[1] 26  8
```

```
> nrow(renewdata)
```

```
[1] 26
```

```
> length(aout.2.4$coefficients)
```

```
[1] 8
```

Aphids (cont.)

```
> jacobian.beta2mu <- pout$gradient
> inies <- grepl("B", as.character(renewdata$varb))
> as.character(renewdata$varb)[inies]

 [1] "B1"  "B2"  "B3"  "B4"  "B5"  "B6"  "B7"  "B8"
 [9] "B9"  "B10" "B11" "B12" "B13"

> jacobian.beta2mu <- jacobian.beta2mu[inies, ]
> j <- 1:13
> jacobian.mu2lambda <- lambda.hat^(- j - 1) /
+     sum((j + 1) * mu.star * lambda.hat^(- j - 2))
> round(jacobian.mu2lambda, 3)

 [1] 0.136 0.081 0.048 0.028 0.017 0.010 0.006 0.004
 [9] 0.002 0.001 0.001 0.000 0.000
```

Aphids (cont.)

```
> jacobian.mu2lambda <- rbind(jacobian.mu2lambda)
> jacobian <- jacobian.mu2lambda %*% jacobian.beta2mu
> dim(jacobian)

[1] 1 8

> as.vector(round(jacobian, 3))

[1]  0.660  1.614 -0.017 -0.005 -0.058 -0.020  0.058
[8] -0.036
```

Aphids (cont.)

```
> se.lambda <- jacobian %*%
+     solve(aout.2.4$fisher) %*% t(jacobian)
> se.lambda <- sqrt(se.lambda)
> se.lambda <- as.vector(se.lambda)
> se.lambda

[1] 0.0558462
```

Aphids (cont.)

There is an alternative way of computing standard errors for nonlinear functions of parameters. Linearize them. Hand that linearization to `predict.aster.formula` as the `amat` optional argument. This will give incorrect "predictions" but *correct standard errors*.

```
> nnode <- nrow(renewdata)
> amat <- array(c(rep(0, nnode/2), jacobian.mu2lambda),
+     c(1, nnode, 1))
> pout.amat <- predict(aout.2.4, varvar = varb, idvar = id,
+     root = root, newdata = renewdata, se.fit = TRUE,
+     amat = amat)
> pout.amat$se.fit

[1] 0.0558462
```

Standard Errors

Both methods (using the `gradient` component or using an artificial `amat`) have their advantages and disadvantages.

Neither is trivial.

The former follows the delta method more closely and transparently.

The latter is less code.

Either works.

Example 1 Revisited

Recall that in deck 1 of these slides we redid the analysis of the first aster paper (Geyer, Wagenius, and Shaw, *Biometrika*, 2007), and we fit three models, did the following analysis of deviance

```
> anova(aout.smaller, aout, aout.bigger)

Analysis of Deviance Table

Model 1: resp ~ varb + fit:(nsloc + ewloc + pop)
Model 2: resp ~ varb + layer:(nsloc + ewloc) + fit:pop
Model 3: resp ~ varb + layer:(nsloc + ewloc + pop)
  Model Df Model Dev Df Deviance P(>|Chi|)
1       17   -2746.7
2       21   -2712.5  4   34.203 6.772e-07 ***
3       33   -2674.7 12   37.838 0.0001632 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Example 1 Revisited (cont.)

But the authors of that paper blithely ignored the P-value $P = 0.00016$ comparing the two models `aout` and `aout.bigger` and chose the smaller model because it had a more direct interpretation about the effect of population of origin (predictor `pop`) on fitness.

How can they do that?

That is what we now propose to examine.

Example 1 Revisited (cont.)

First, a disclaimer. They did not actually make this choice "blithely" as the preceding slide said. They did consider carefully.

The argument was that they were modeling fitness, and the distribution of fitness (actually best surrogate of fitness in their data) is not very different between the two models.

The distribution of other components of fitness (other than the final one) may differ quite a lot, but that was not the question of scientific interest. (More on this later).

So what do these models say about the distribution of fitness?

Example 1 Revisited (cont.)

```
> pop <- levels(redata$pop)
> nind <- length(unique(redata$id))
> nnode <- nlevels(redata$varb)
> npop <- length(pop)
> amat <- array(0, c(nind, nnode, npop))
> amat.ind <- array(as.character(redata$pop),
+     c(nind, nnode, npop))
> amat.node <- array(as.character(redata$varb),
+     c(nind, nnode, npop))
> amat.fit <- grepl("hdct", amat.node)
> amat.fit <- array(amat.fit,
+     c(nind, nnode, npop))
> amat.pop <- array(pop, c(npop, nnode, nind))
> amat.pop <- aperm(amat.pop)
> amat[amat.pop == amat.ind & amat.fit] <- 1
```

Example 1 Revisited (cont.)

```
> pout <- predict(aout,
+      varvar = varb, idvar = id, root = root,
+      se.fit = TRUE, amat = amat)
> pout.bigger <- predict(aout.bigger,
+      varvar = varb, idvar = id, root = root,
+      se.fit = TRUE, amat = amat)
```

Example 1 Revisited (cont.)

The first interesting thing about these "predictions" (actually point estimates of parameters with standard errors) is that the point estimates are exactly the same for the two models.

```
> pout$fit

[1]  81 171 112  31 286 218 167

> pout.bigger$fit

[1]  81 171 112  31 286 218 167

> all.equal(pout$fit, pout.bigger$fit)

[1] TRUE
```

Example 1 Revisited (cont.)

And why is that? These are submodel canonical statistics (components of $M^T y$). Thus by the observed-equals-expected property of exponential families their MLE are equal to their observed values and hence equal to each other.

So that is certainly not a reason to prefer one model to the other.

If the (estimated) means are exactly the same how about (estimated, asymptotic, approximate) variances?

Example 1 Revisited (cont.)

The asymptotic variance matrix of these canonical statistics is actually diagonal for each model. (This was not obvious to me. I had to calculate it to see this.) The reason is that different populations of origin have different individuals in the sample, and only individuals from one population contribute to estimating one of these canonical statistics.

Example 1 Revisited (cont.)

Thus it is enough to look at the asymptotic standard errors (all the covariances are zero).

```
> pout$se.fit

[1] 13.617532 19.984170 16.267065  8.524453 25.968492
[6] 22.227096 19.884556

> pout.bigger$se.fit

[1] 14.521691 17.870387 14.513433  9.105173 27.857509
[6] 21.589790 21.642168
```

Not that different.

Example 1 Revisited (cont.)

If, like in Example 2 (Deck 3), we were interested in the effect of `pop` on the different components of fitness, then the P-value $P = 0.00016$ does indicate that the model `aout.bigger`, which has different `pop` effects in different "layers" of the graph, does show a statistically significant difference in the way the components of fitness combine to make up fitness in the various population of origin groups.

But if we are only interested in overall fitness rather than the separate components, then there is hardly any difference in the models.