Stat 8931 (Aster Models)
Lecture Slides Deck 6

Charles J. Geyer

School of Statistics
University of Minnesota

June 7, 2015

Random Effects Models

Abandon all hope, ye who enter here.

sign on the gates of hell in Dante's *Inferno*

Random Effects Models (cont.)

A bit overdramatic, perhaps, but I want to warn against using random effect aster models.

They have none of the good properties of fixed effect aster models.

The models have questionable (and untestable) validity.

Their parameter estimates have questionable validity.

They should be used only when irreplaceable, when nothing else addresses the issues of scientific interest.

Random Effects Models (cont.)

Bad reasons for using random effects.

- They make a good story. It sounds better when certain effects are described as “random”.
- They supposedly make the inference stronger, describing an imaginary “population” from which the “random” effects are drawn.

Random Effects Models (cont.)

The only good reason for using random effects.

- They are the only way to address important questions of scientific interest.

Here are two such issues:

- nested predictors and
- quantitative genetics.

There may be more, but those are the only ones in published examples.

Paper and Tech Report

Geyer, et al. (*Annals of Applied Statistics*, 2013) and School of Statistics Technical Report 692.

Nested Predictors

We say two categorical predictor variables (R calls them factors) are **nested** if each value of one only occurs with a single value of the other. We say the former is "nested within" the latter.

In the first example in the paper and TR (data on the invasive California wild radish *Raphanus sativus* collected and first analyzed by Caroline Ridley) there are two nested predictors.

There were two experimental sites, and there were blocks within sites.

There were two regions (northern, coastal California and southern, inland California) and there were three ancestral populations collected from each region.

One experimental site was in each region.

Nested Predictors (cont.)

We wish (the scientists wished) to make inference about site and region. In particular, a statistically significant site by region interaction (this time “interaction” really means interaction) may indicate local adaptation.

Certainly, there is no evidence of local adaptation if this interaction is not statistically significant.

But even if this interaction is statistically significant, there is still no evidence of local adaptation unless each “region” does better (has higher fitness, outcompetes) than other “regions” in the corresponding site.

“Region” is in scare quotes because “regions” don’t have fitness, only individual organisms do. So this is about parameters of a statistical model rather than direct observation of reality.

Nested Predictors (cont.)

The problem that random effects addresses here is that if we put both the region predictor variable and the population predictor variable in the model as fixed effects (the usual kind of aster analysis that everything in the course up to here has been about), the model isn't full rank and some of these predictors need to be dropped.

Nested Predictors (cont.)

And which ones get dropped is arbitrary (parameters are meaningless quantities) so fixed effect models (all of the aster models we have discussed so far) leave us with three options, none of them good,

- Put both the nested effects and the effects they are nested within in the model, let R drop whatever it wants, and try to interpret that.
- Put only the nested effects (but not the effects they are nested within) in the model, and try to interpret that.
- Put only the effects they are nested within (omitting the nested effects) in the model, and try to interpret that.

Nested Predictors (cont.)

In the radish example, only the latter is guaranteed to have the effects of scientific interest (site and region) in the model (not dropped to obtain full rank model matrix), and that is what Ridley and Ellstrand (*Evolutionary Applications*, 2010) did (random effects aster models weren't available then).

But this ignores the "nested within" effects (block and population), and it ignores them in a way that fans of random effects models do not approve.

Nested Predictors (cont.)

The usual thing to do is treat nested effects as random effects.

This is not motivated by the idea of generalizing to the "population" of block effects or population effects (different meaning for "population").

All possible blocks? Who cares about that?

All possible populations might be scientifically interesting (assuming it is even a meaningful concept), but the actual populations (six of them) are not a large sample, even if they were chosen randomly in some sense, which they were not.

Nested Predictors (cont.)

It is motivated by the idea that just using some blocks and some populations (and not very many) gives a less than ideal estimates of the effects of scientific interest they are nested within (site and region). And the random effects story at least tries to deal with that.

Radishes

The invasive California wild radish (*Raphanus sativus*) data (Ridley and Ellstrand, 2010) is in the dataset `radish` in the `aster` package.

```
> library(aster)
> data(radish)
> dim(radish)

[1] 858  11
```

Radishes (cont.)

```
> levels(radish$varb)
```

```
[1] "Flowering" "Flowers"   "Fruits"
```

The graph is

$$1 \xrightarrow{\text{Ber}} y_1 \xrightarrow{\text{0-Poi}} y_2 \xrightarrow{\text{Poi}} y_3$$

- y_1 (`Flowering`), indicator variable of any flowers produced
- y_2 (`Flowers`), count of the number of flowers produced
- y_3 (`Fruits`), count of the number of fruits produced

As usual, y_1 is just the indicator of $y_2 > 0$, and the combination of the first two arrows is zero-inflated Poisson.

Radishes (cont.)

```
> names(radish)

 [1] "Site"             "Block"            "Region"
 [4] "Pop"              "varb"             "resp"
 [7] "id"               "root"             "varbFlowering"
[10] "varbFlowers"      "fit"
```

`Site`, `Block`, `Region`, and `Pop` (population) are the predictor variables of scientific interest. `Block` is nested within `Site`. `Pop` is nested within `Region`.

`varb`, `resp`, `id`, and `root` are as usual.

`fit` is, as usual, the indicator of the terminal node, here `Fruits`, so it is the same as `varbFruits` when we put `varb` in the model.

Not sure what `varbFlowering` and `varbFlowers` are good for.

R function reaster

The R function reaster in the aster package fits random effects aster models (much more about how later in this deck).

It fits aster models having the "model equation"

$$\varphi = a + M\alpha + \sum_{k=1}^{K} Z_k b_k \tag{$*$}$$

where a is a known vector (offset), M is a known matrix (model matrix for fixed effects), each Z_k is a known matrix (model matrix for random effects), α is an unknown vector (fixed effects), and the b_k are independent random vectors whose components are assumed IID mean-zero normal with variance σ_k^2 (variance component).

The Greek letters (fixed effects and the variance components) are the unknown parameters to be estimated.

R function `reaster` (cont.)

$$\varphi = a + M\alpha + \sum_{k=1}^{K} Z_k b_k \tag{$*$}$$

So there are a bunch of model matrices we need to specify, one for fixed effects and one or more for random effects.

How do we do that?

Existing R packages that do random effects models, such as `nlme` and `lme4` have tricky (IMHO) ways to specify fixed and random effects in one formula. The schemes used by the different packages are different, even though those packages have some of the same authors. What is more, those schemes are not general enough to specify all models specified by $(*)$.

R function `reaster` (cont.)

$$\varphi = a + M\alpha + \sum_{k=1}^{K} Z_k b_k \qquad (*)$$

So `reaster` uses a different scheme, with one formula per model matrix. Obviously, our scheme can specify all the models specified by $(*)$.

Our scheme is more general and more transparent, but more clumsy.

Other schemes are (perhaps) more user-friendly, but trickier and less general.

Anyway, like it or not, our scheme is the only one available for aster models.

R function `reaster` (cont.)

The formulas for random effects are always fairly simple, often only one categorical predictor (factor), which specifies one random effect per category.

One almost always (the “almost” is just academic weasel wording, AFAIK always always) wants to specify no intercept. Like

```
~ 0 + foo
```

if `foo` is a categorical predictor (factor).

The formula has a twiddle in it (like all R formulas) but does not need to specify the response on the left hand side because `reaster` gets the response from the formula for fixed effects (and it doesn’t need to be specified more than once).

R function `reaster` (cont.)

Unlike what happens with fixed effects, the formulas

```
~ 0 + foo
```

and

```
~ foo
```

specify *different random effects models*.

The first specifies a model matrix with as many columns as levels of `foo`, and each column is the dummy variable indicating one category. Call that Z_1.

The second specifies a model matrix of the same dimension of Z_1 but having the first column ("intercept") all ones and the rest of the columns the same as Z_1. Call that Z_2.

R function `reaster` (cont.)

Considered as fixed effect model matrices, Z_1 and Z_2 would specify the same models because they have the same column spaces

$$
\begin{gathered}
V_1 = \{\, Z_1 \beta : \beta \in \mathbb{R}^p \,\} \\
V_2 = \{\, Z_2 \beta : \beta \in \mathbb{R}^p \,\}
\end{gathered}
$$

Considered as random effect model matrices, Z_1 and Z_2 would specify different models because the $Z_1 b$ and $Z_2 b$ are mean zero normal random vectors with variance matrices

$$
\begin{gathered}
\sigma^2 Z_1 Z_1^T \\
\sigma^2 Z_2 Z_2^T
\end{gathered}
$$

where σ^2 is the variance of the components of b, and these are different.

R function `reaster` (cont.)

Just one way that fixed effects models and random effects models are different.

Z_1 and Z_2 having the same column space (so specifying the same fixed effects models) does not mean $Z_1 Z_1^T = Z_2 Z_2^T$ (so they specify the same random effects models).

So we do have to be careful with formulas for random effects.

But we also had to be careful with formulas for fixed effects.

Specification of model matrices using the R formula mini-language is tricky! This is just another aspect of the trickiness.

Radishes (cont.)

$$1 \xrightarrow{\text{Ber}} y_1 \xrightarrow{\text{0-Poi}} y_2 \xrightarrow{\text{Poi}} y_3$$

```
> pred <- c(0,1,2)
> fam <- c(1,3,2)
> sapply(fam.default(), as.character)[fam]

[1] "bernoulli"
[2] "truncated.poisson(truncation = 0)"
[3] "poisson"
```

Radishes (cont.)

```
> rout <- reaster(resp ~ varb + fit : (Site * Region),
+     list(block = ~ 0 + fit : Block,
+         pop = ~ 0 + fit : Pop),
+     pred, fam, varb, id, root, data = radish)
```

Here we have two variance components because we have two groups of nested effects (`Block` within `Site` and `Pop` within `Region`).

For the same reasons as always, we want “no naked predictors” so our formulas for random effects have `fit : Block` instead of just `Block` and so forth.

We collect the formulas for random effects in a list (one formula for each variance component). The names in the list are not necessary, they only make the `summary` printout nicer.

Radishes (cont.)

```
> summary(rout)

Call:
reaster.formula(fixed = resp ~ varb + fit:(Site * Region), random = list(block
    fit:Block, pop = ~0 + fit:Pop), pred = pred, fam = fam, varvar = varb,
    idvar = id, root = root, data = radish)


Fixed Effects:
                            Estimate Std. Error  z value Pr(>|z|)
(Intercept)               -467.24230    1.75183 -266.717   <2e-16 ***
varbFlowers                474.13821    1.75416  270.293   <2e-16 ***
varbFruits                 466.11036    1.76038  264.779   <2e-16 ***
fit:SitePoint Reyes         -0.03620    0.20781   -0.174    0.862
fit:RegionS                 -0.12249    0.07892   -1.552    0.121
fit:SiteRiverside:RegionS    0.49930    0.01211   41.223   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Square Roots of Variance Components (P-values are one-tailed):
      Estimate Std. Error z value Pr(>|z|)/2
block  0.32820    0.07358   4.461   4.09e-06 ***
pop    0.09619    0.02992   3.214   0.000654 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Radishes (cont.)

Recall that the question of scientific interest is local adaptation which is shown by

- the fixed effects for interaction (here `Site` by `Region`) are statistically significantly different from zero and
- the pattern of means is as expected (each population is the best in the locality it is from).

We have the first. The `fit:SiteRiverside:RegionS` fixed effect is highly statistically significantly different from zero ($P \approx 0$).

For the second, it is difficult to even say what "prediction" of mean value parameters means for random effects models (more on this later), so we use the fixed effects model for that.

Radishes (cont.)

```
> aout <- aster(resp ~ varb + fit : (Site * Region),
+     pred, fam, varb, id, root, data = radish)
> summary(aout, info.tol = 1e-9)

Call:
aster.formula(formula = resp ~ varb + fit:(Site * Region), pred = pred,
    fam = fam, varvar = varb, idvar = id, root = root, data = radish)

                            Estimate Std. Error  z value Pr(>|z|)
(Intercept)               -5.198e+02  1.590e+00 -326.949  < 2e-16 ***
varbFlowers                5.267e+02  1.593e+00  330.727  < 2e-16 ***
varbFruits                 5.188e+02  1.591e+00  326.145  < 2e-16 ***
fit:SitePoint Reyes       -3.257e-03  2.370e-03   -1.374  0.16934
fit:RegionS               -5.339e-03  1.972e-03   -2.707  0.00678 **
fit:SiteRiverside:RegionS  3.853e-01  7.470e-03   51.574  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Original predictor variables dropped (aliased)
     fit:SiteRiverside
```

Radishes (cont.)

```
> pout <- predict(aout)
> varb <- as.character(radish$varb)
> site <- as.character(radish$Site)
> region <- as.character(radish$Region)
> unique(varb)

[1] "Flowering" "Flowers"   "Fruits"

> unique(site)

[1] "Riverside"   "Point Reyes"

> unique(region)

[1] "N" "S"
```

Radishes (cont.)

```
> unique(pout[varb == "Fruits" & site == "Riverside" &
+     region == "N"])

[1] 171.4521

> unique(pout[varb == "Fruits" & site == "Riverside" &
+     region == "S"])

[1] 338.6892
```

For those who don't know, University of California at Riverside is in southern California (La la land), and indeed the southern populations do better there.

Radishes (cont.)

```
> unique(pout[varb == "Fruits" & site == "Point Reyes" &
+     region == "N"])

[1] 154.2576

> unique(pout[varb == "Fruits" & site == "Point Reyes" &
+     region == "S"])

[1] 111.7123
```

By elimination, Point Reyes National Seashore must be in northern California (actually in Marin county, north of the Golden Gate), and indeed the northern populations do better there.

So that is "local adaptation" each variety does better (has higher fitness, outcompetes) the other varieties in it own bailiwick.

Radishes (cont.)

We could also “predict” using the random effects model, but what does that mean?

- We could say the random effects are zero on average, so we should put in zero for the random effects.
- We could try to account for the nonlinearity of the $\varphi \to \mu$ transformation somehow.
- If we are predicting for observed individuals, we should take into account the data for that individual, perhaps by using estimated random effects (more on this later).
- And in the latter we could try to account for nonlinearity.

All are reasonable, we will just illustrate the first.

Radishes (cont.)

```
> identical(names(aout$coefficients), names(rout$alpha))

[1] TRUE

> prout <- predict(aout, newcoef = rout$alpha)
> site <- site[varb == "Fruits"]
> region <- region[varb == "Fruits"]
> prout <- prout[varb == "Fruits"]
> pout <- pout[varb == "Fruits"]
> fred <- paste(site, region, sep = ":")
> sapply(split(prout, fred), unique)

Point Reyes:N Point Reyes:S   Riverside:N   Riverside:S
     154.2742      131.6803      161.8043      273.3486

> sapply(split(pout, fred), unique)

Point Reyes:N Point Reyes:S   Riverside:N   Riverside:S
     154.2576      111.7123      171.4521      338.6892
```

Radishes (cont.)

Not that we need it for these data, but if there were more than two regions or more than two sites so there would be more than one parameter for the “interaction” of region and site, we would have to do a general test of model comparison like so

```
> rout0 <- reaster(resp ~ varb + fit : (Site + Region),
+     list(block = ~ 0 + fit : Block,
+         pop = ~ 0 + fit : Pop),
+     pred, fam, varb, id, root, data = radish)
> anova(rout0, rout)
```

```
Analysis of Deviance Table

Model 1: resp ~ varb + fit:(Site + Region), ~0 + fit:Block, ~0 + fit:Pop
Model 2: resp ~ varb + fit:(Site * Region), ~0 + fit:Block, ~0 + fit:Pop
  Mod Df Fix Mod Df Rand Mod Dev Df Fix Df Rand Deviance P-value
1          5           2 1175212
2          6           2 1176704      1       0   1492.3       0
```

In fact, the R function `anova.asterOrReaster` that does this was written at the request of a user who had just this problem.

Oats

The second example in Geyer, et al. (2013) is about data on the slender wild oat (*Avena barbata*) collected and first analyzed by Bob Latta. It is in the dataset `oats` in the R contributed package `aster`.

The graph is

$$1 \xrightarrow{\text{Ber}} y_1 \xrightarrow{\text{0-Poi}} y_2$$

where

- y_1 is the indicator of whether any spikelets (compound flowers) were produced
- y_2 is the count of the number of spikelets produced

So here the unconditional distribution of the number of spikelets is zero-inflated Poisson.

Oats (cont.)

We load the data

```
> data(oats)
> names(oats)

 [1] "Plant.id" "Env"        "Gen"        "Fam"
 [5] "Site"     "Year"       "varb"       "resp"
 [9] "id"       "root"       "fit"

> levels(oats$varb)

[1] "Spike" "Surv"
```

`varb`, `resp`, `id`, `root`, and `fit` are the same as usual (the last, the indicator of “fitness” nodes, here `Spike` nodes).

Oats (cont.)

```
> sapply(oats, class)

 Plant.id       Env       Gen       Fam      Site
 "factor"  "factor"  "factor"  "factor"  "factor"
     Year      varb      resp        id      root
 "factor"  "factor" "integer" "integer" "numeric"
      fit
"numeric"
```

- Gen is the “ecotype” "X" for xeric (from more dry region) or "M" for mesic (from less dry region)
- Fam is the “accession” the particular variety, nested withing Gen
- Site is the site (more on next slide)
- Year is the year

Oats (cont.)

The experimental sites were at the Sierra Foothills Research and Extension Center (`Site == "SF"`), which is northeast of Sacramento on the east side of the Central Valley of California, and at the Hopland Research and Extension center (`Site = "Hop"`), which is in the California Coastal Ranges north of San Francisco. Hopland receives 30% more rainfall and has a less severe summer drought than Sierra foothills.

So if we have local adaptation we expect the xeric varieties to do better at Sierra Foothills and the mesic varieties to do better at Hopland.

In fact, the pattern of means is not right for local adaptation (Latta, *Molecular Ecology*, 2009). The mesic ecotype had higher fitness in both environments.

Oats (cont.)

So we are doing this analysis not to show local adaptation, but just to show that we can do a complicated random effects aster analysis.

Latta (2009) did not do a random effects aster analysis, because it wasn't available yet. And he did want random effects. So he had to "assume" fitness is normally distributed and use conventional normal response, normal random effects models.

We redo his analysis but using the aster model with the same fixed and random effects.

Oats (cont.)

These fixed and random effects were

Effect	Type
Site	fixed
Year	random
Gen	fixed
Fam	random
Gen : Site	fixed
Gen : Year	random
Site : Fam	random
Year : Fam	random

An interaction of a fixed effect and a random effect is a random effect. So is an interaction of a random effect and a random effect.

There is one variance component for each random effect term.

Oats (cont.)

```
> data(oats)
> pred <- c(0,1)
> fam <- c(1,3)
> woof <- suppressWarnings(try(load("rout.rda"),
+     silent = TRUE))
> if (inherits(woof, "try-error")) {
+ rout <- reaster(resp ~ varb + fit : (Gen * Site),
+     list(year = ~ 0 + fit : Year, fam = ~ 0 + fit : Fam,
+         fam.site = ~ 0 + fit : Fam : Site,
+         fam.year = ~ 0 + fit : Fam : Year,
+         gen.year = ~ 0 + fit : Gen : Year),
+     pred, fam, varb, id, root, data = oats)
+ save(rout, file = "rout.rda")
+ }
```

Oats (cont.)

Always use : rather than * in random effects formulas.

Main effects and "interaction" effects are in different variance components (have different variance), hence in a different formulas.

Also there wouldn't even be the same number of random effects!

```
> cc <- sample(c("red", "blue", "green"), 100,
+     replace = TRUE)
> dd <- sample(c("N", "S", "E", "W"), 100, replace = TRUE)
> foo <- data.frame(color = cc, direction = dd)
```

Oats (cont.)

```
> ncol(model.matrix(~ 0 + color, data = foo))

[1] 3

> ncol(model.matrix(~ 0 + direction, data = foo))

[1] 4

> ncol(model.matrix(~ 0 + color : direction, data = foo))

[1] 12

> ncol(model.matrix(~ 0 + color * direction, data = foo))

[1] 12
```

First three do not add up to the last!

Oats (cont.)

```
> z1 <- model.matrix(~ 0 + color : direction, data = foo)
> z2 <- model.matrix(~ 0 + color * direction, data = foo)
> identical(dim(z1), dim(z2))

[1] TRUE

> all.equal(z1 %*% t(z1), z2 %*% t(z2))

[1] "Mean relative difference: 6.805674"
```

So `~ 0 + color : direction` and `~ 0 + color * direction` do not specify the same random effects!

Oats (cont.)

```
> summary(rout)


Call:
reaster.formula(fixed = resp ~ varb + fit:(Gen * Site), random = list(year = ~0 +
    fit:Year, fam = ~0 + fit:Fam, fam.site = ~0 + fit:Fam:Site,
    fam.year = ~0 + fit:Fam:Year, gen.year = ~0 + fit:Gen:Year),
    pred = pred, fam = fam, varvar = varb, idvar = id, root = root,
    data = oats)


Fixed Effects:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)       2.86833    0.36873   7.779 7.31e-15 ***
varbSurv        -15.15044    0.48600 -31.174  < 2e-16 ***
fit:GenM          0.27250    0.13975   1.950  0.05118 .
fit:SiteSF       -0.32606    0.09609  -3.393  0.00069 ***
fit:GenX:SiteSF   0.09138    0.14293   0.639  0.52259
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Square Roots of Variance Components (P-values are one-tailed):
         Estimate Std. Error z value Pr(>|z|)/2
year      0.70794    0.25524   2.774    0.00277 **
fam       0.00000         NA      NA         NA
fam.site  0.17502    0.03013   5.809   3.13e-09 ***
fam.year  0.18193    0.02538   7.168   3.81e-13 ***
gen.year  0.10986    0.06078   1.807    0.03535 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Oats (cont.)

Wouldn't need to do this except for slides

```
> sout <- summary(rout)
> printCoefmat(sout$alpha)

                  Estimate Std. Error  z value  Pr(>|z|)    
(Intercept)       2.868332   0.368725   7.7790 7.307e-15 ***
varbSurv        -15.150443   0.486001 -31.1737 < 2.2e-16 ***
fit:GenM          0.272503   0.139750   1.9499 0.0511837 .  
fit:SiteSF       -0.326056   0.096087  -3.3933 0.0006905 ***
fit:GenX:SiteSF   0.091382   0.142927   0.6394 0.5225896    
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Coefficient for local adaptation (`fit:GenX:SiteSF`) isn't statistically significantly different from zero.

Oats (cont.)

```
> printCoefmat(sout$sigma)

         Estimate Std. Error z value Pr(>|z|)/2
year     0.707939   0.255239  2.7736   0.002772 **
fam      0.000000         NA      NA         NA
fam.site 0.175019   0.030126  5.8095  3.133e-09 ***
fam.year 0.181932   0.025382  7.1679  3.809e-13 ***
gen.year 0.109865   0.060785  1.8074   0.035347 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Variance component `fam` is estimated to be exactly zero. If the true unknown parameter value is actually zero (on the boundary), the estimator is not asymptotically normal and “standard error” makes no sense. So everything else in the row is `NA`.

Oats (cont.)

Of course, we do not confuse the estimator and the true unknown parameter ($\hat{\sigma}_k \neq \sigma_k$) so we don't know whether the true unknown parameter is zero or not.

So we could do a one-sided confidence interval for σ_k, but the notion of "standard error" is no help. There is nothing useful to print in this kind of table except `NA`.

For most purposes, it is enough that $\hat{\sigma}_k = 0$ is certainly not evidence against the null hypothesis $\sigma_k = 0$, so we certainly have no evidence in favor of the alternative hypothesis $\sigma_k > 0$.

Oats (cont.)

If we really wanted that one-sided confidence interval, we could do something.

But it would be some work and there is no function in the `aster` package that just does it.

If two or more variance components were estimated to be zero, things get very tricky theoretically. The theory is understood (Chernoff, *Annals of Mathematical Statistics*, 1954; Geyer, *Annals of Statistics*, 1994), but there is no easy way to calculate the theoretical asymptotic joint distribution of the parameter estimates nor its marginals.

This is an interesting open research question.

Oats (cont.)

Why is `Year` a random effect?

```
> levels(oats$Year)
```

```
[1] "2003" "2004" "2006" "2007"
```

Does making `Year` a random effect make data about 4 years say everything there is to be said about all years (forever)?

If that were the case, random effects models would be true magic, voodoo statistics.

There is nothing wrong with making `Year` a random effect if you want to, but there is no magic.

Exponential Family Mixed Models

At first, random effects models (also called “mixed” models because they have fixed effects too) were only normal response and normal random effects.

To distinguish them from generalized linear mixed models (GLMM) and exponential family mixed models (EFMM) that appeared later, we call the original ones linear mixed models (LMM).

Exponential Family Mixed Models (cont.)

LMM start with R. A. Fisher (*Proceedings of the Royal Society of Edinburgh*, 1918) and Sewall Wright (*Genetics*, 1918; *Proceedings of the National Academy of Sciences*, 1920; *Journal of Agricultural Research*, 1921). All are genetics papers.

LMM were soon turned into a more general methodology (Fisher, *Statistical Methods for Research Workers*, 1925; *Design of Experiments*, 1935).

However, according to the authoritative book on the subject (Searle, Casella, and McCulloch, *Variance Components*, 1992) the full theory of maximum likelihood estimation and restricted maximum likelihood estimation (REML) for general LMM doesn't appear until the seventies (Hartley and Rao, *Biometrika*, 1967; Patterson and Thompson, *Biometrika*, 1971; *Proceedings of the 8th International Biometric Conference*, 1974).

Exponential Family Mixed Models (cont.)

GLMM appear in Stiratelli, Laird, and Ware (*Biometrics*, 1984) but do not really start to catch on until Breslow and Clayton (*Journal of the American Statistical Association*, 1993) provided a useful but only approximate method of calculating maximum likelihood estimates.

For simple enough GLMM, those in which only one random effect is involved in the "model equation" for each individual, the likelihood can be calculated by numerical integration, and this is also done (not sure who deserves the credit for this).

Exponential Family Mixed Models (cont.)

For models with "crossed" random effects, like those we have already looked at, there is no hope of exact calculation of the likelihood (more on this later). Only two methods for approximate calculation have any sizable literature: Breslow and Clayton (1992, Google Scholar says cited by 3040) and Markov chain Monte Carlo (MCMC) (Thompson and Guo, *IMA Journal of Mathematics Applied in Medicine and Biology*, 1991, Google Scholar says cited by 88; Geyer, *Journal of the Royal Statistical Society, Series B*, 1994, Google Scholar says cited by 194).

Of course, even more papers, cite papers that cite these (without necessarily citing the originals), and so forth.

Exponential Family Mixed Models (cont.)

MCMC is hard for ordinary users to get right (hard even for experts in non-toy models). No widely used software implements it for general GLMM.

R and SAS and other statistical computing languages have packages for GLMM that use either numerical integration (for simple enough models) or Breslow-Clayton (for the rest).

The R function `reaster` uses Breslow-Clayton.

Exponential Family Mixed Models (cont.)

A third method using good old-fashioned Monte Carlo (ordinary Monte Carlo, OMC) was introduced by Sung and Geyer (*Annals of Statistics*, 2007), but it does not work for complicated models.

Exponential Family Mixed Models (cont.)

As we shall see, Breslow-Clayton is far from ideal.

IMHO, it is an open research question to really do maximum likelihood (or other non-maximum-likelihood) estimation right for GLMM.

Maybe MCMC or OMC will make a comeback. Maybe something completely different (but have no idea what that would be).

Exponential Family Mixed Models (cont.)

There is an even worse problem with GLMM than difficulty of computing estimates.

Even the simple toy models that much of the literature analyze and which the OMC method (Sung and Geyer, 2007) handle perfectly are far from asymptopia.

Very large sample sizes are needed for these very simple models to have approximately normal parameter estimates (examples in Sung and Geyer, 2007).

So, in general, even if you can get estimates, it is unclear whether your P-values and confidence intervals have any validity.

Exponential Family Mixed Models (cont.)

Hence Geyer, et al. (2013) recommend doing a parametric bootstrap if one has any doubt about the asymptotics and wants to be sure about the inference. (That's hardly profound advice. *Whenever one doubts asymptotics*, do a parametric bootstrap!)

But bootstrapping an already slow method like MCMC or OMC is intolerably slow!

Hence (so far) the preference for the (possibly bad) Breslow-Clayton estimators.

Exponential Family Mixed Models (cont.)

IMHO this whole area has a long way to go and the very optimistic literature on the subject (and there is a lot of that) is *foolishly* optimistic.

A lot more caution is warranted than most users understand.

Exponential Family Mixed Models (cont.)

EFMM were introduced by Geyer, et al. (2013). They don't call them that. They do say "Exponential Family Models with Random Effects" in the title of the paper, but are really only interested in aster models with random effects.

The only reason for EFMM is to be like GLMM and LMM. (Of course, nobody says LMM either. That too is to be like GLMM.)

Exponential Family Mixed Models (cont.)

$$\varphi = a + M\alpha + \sum_{k=1}^{K} Z_k b_k \qquad (*)$$

So what is an EFMM? It is a regular full exponential family statistical model in which the canonical parameter φ is specified using the "model equation" $(*)$, in which, recall, the b_k are independent vectors whose components are IID normal, mean zero, variance σ_k^2.

The unknown parameters to be estimated are the Greek letters: the vector α of fixed effects and the vector of variance components σ_k^2.

I don't like differentiating with respect to parameters that don't have explicit notation, so write $\nu_k = \sigma_k^2$ and ν for the vector with components ν_k.

Missing Data, Latent Variables, Empirical Bayes

And the title doesn't have enough keywords. . . . Hierarchical Models, Multi-Level Models, Structural Equation Models, Causal Models, Hidden this and that (like Hidden Markov Models), and probably a lot more I can't think of or haven't heard of.

But there is only one idea in all of this. The probability model has two kinds of variables. The PDF (or PMF or PMDF) is

$$f_\theta(x, y)$$

where y is observed and x is not observed (missing, latent, random, whatever).

Thus the probability model for the actual observed data is the marginal

$$f_\theta(y) = \int f_\theta(x, y) \, dx \qquad (**)$$

(or sum if PMF or sum and integrate if PMDF).

Missing Data, Latent Variables, Empirical Bayes (cont.)

$$f_\theta(y) = \int f_\theta(x, y) \, dx \qquad (**)$$

And that's it! That's the whole theory!

In LMM where everything is normal, the integral can be done analytically, because every marginal of normal is normal.

In GLMM and EFMM, the integral can never be done analytically. And only in the simplest models can it be done exactly by numerical integration.

So we need approximations, and then we need to theorize about approximations.

Exponential Family Mixed Models (cont.)

$$\varphi = a + M\alpha + \sum_{k=1}^{K} Z_k b_k \tag{$*$}$$

For doing theory we want to simplify $(*)$. Write

$$Z = \begin{pmatrix} Z_1 & Z_2 & \cdots & Z_K \end{pmatrix}$$

and

$$b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_K \end{pmatrix}$$

so

$$Zb = \sum_{k=1}^{K} Z_k b_k$$

Exponential Family Mixed Models (cont.)

So now the "model equation" is

$$\varphi = a + M\alpha + Zb \tag{$***$}$$

where

- a is the offset vector
- M is the model matrix for fixed effects
- Z is the model matrix for random effects
- α is the vector of fixed effects
- b is the vector of random effects

We assume the distribution of b is multivariate normal with mean zero and variance matrix D, which is diagonal.

Exponential Family Mixed Models (cont.)

By putting all of the random effects in one vector b we have lost track of which random effects go with which variance component.

Since all of the random effects are independent, D is diagonal.

And each diagonal element of D is one of the ν_k, but the diagonal elements are not all different, the random effects come in blocks with the same variance.

$$E_k = \frac{\partial D}{\partial \nu_k}$$

is a diagonal zero-or-one-valued matrix the same shape as D.

The ones on the diagonal of E_k tell us which random effects have variance ν_k.

Why are Random Effects Assumed Normal?

Like the title asks!

In the assume-everything-is-normal era (approximately 1800 to 1970) assumptions of normality were more or less unquestioned.

> Everybody believes in the law of errors, the experimenters because they think it is a mathematical theorem, the mathematicians because they think it is an experimental fact.
>
> — Lippman, quoted by Poincaré, quoted by Cramér

"The law of errors" is an old name for the normal distribution.

Note that it builds into the name "*the* law . . ." the idea that it is *the* only distribution we consider for "errors".

Also note the Lippman quote is sarcastic. Justification for this belief was always known to be, at best, shaky.

Why are Random Effects Assumed Normal? (cont.)

The central limit theorem does say that if each data point is the sum of lots and lots of little independent thingummies all having about the same variance, then the data point will be approximately normal.

But if one actually *looks* at data instead of just *theorizing* about it, one finds a lot of data does not *look* very normal!

Tukey's (1977) book *Exploratory Data Analysis* was revolutionary.

But what was revolutionary in it? Partly just looking at data. It introduced boxplots and stem-and-leaf plots. It also introduced a bunch of robust and nonparametric methods that aren't used much any more (median polish smoothing, for example) but only because better methods have been invented since then.

Why are Random Effects Assumed Normal? (cont.)

But IMHO what was really revolutionary about EDA (*Exploratory Data Analysis*) was that it drove a stake through the heart of the idea of statistical analysis based on assumptions that can be shown to be radically wrong by a simple plot.

It changed statistics forever.

Why are Random Effects Assumed Normal? (cont.)

But nobody applies EDA to purely hypothetical random variables that no one can see. You cannot prove they are not normal with a plot!

But if we now admit that a lot of real observable data is not normal, why would we continue to maintain that completely imaginary unobservable data is normal?

You hear people say maybe it doesn't make much difference. Perhaps the inference would be more or less the same with non-normal random effects.

But perhaps not. It's not like anybody ever uses non-normal random effects models! AFAIK there is no theory about this.

Why are Random Effects Assumed Normal? (cont.)

In theory, semiparametric inference applies to this situation. One could be parametric about the fixed effects and nonparametric about the random effects.

But (again AFAIK) nobody has ever done this. It is another open research question.

So we just don't know how much the assumption of normal random effects affects statistical inference.

If anybody tells you there isn't much effect, ask them what theorem says so.

Exponential Family Mixed Models (cont.)

So that is three reasons to be skeptical about GLMM and EFMM.

- Estimates are only approximate MLE of unknown accuracy.
- Models are not in asymptopia. P-values and confidence intervals have unknown accuracy.

Parametric bootstrap helps with these.

- Unknown effect of normality assumptions, which are uncheckable.

Parametric bootstrap no help with this one.

Exponential Family Mixed Models (cont.)

And if that isn't enough issues, here's another.

In GLM and EFM we have strictly concave log likelihood so the MLE is unique if it exists.

This is no longer true for LMM or GLMM or EFMM. The log likelihood can be multimodal so there is no guarantee that a local maximum is the global maximum.

Exponential Family Mixed Models (cont.)

Actually, the “usual” asymptotics of maximum likelihood does not require global maximizers. The theory describes the “right” local maximizer. It proves there is only one root-n-consistent local maximizer and describes it.

The theory also says a one-step-Newton update of a root-n-consistent preliminary estimator is asymptotically equivalent to the MLE (a fully efficient estimator).

But we don’t know any provably root-n-consistent estimators for LMM or GLMM or EFMM. So that’s no help.

There is no reason to believe that a local maximum of the true log likelihood — even it we can calculate one — is a good estimator.

Exponential Family Mixed Models (cont.)

Enough gloom and doom.

We've said it. We don't need to keep saying it.

From now on, it goes without saying.

We'll just present the theory and then more examples.

Exponential Family Mixed Models (cont.)

The log likelihood for a regular full exponential family model (and, in particular, for a saturated aster model in the unconditional canonical parameterization) is

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

We turn this into the so-called "complete data" log likelihood by plugging in the "model equation"

$$l_c(\alpha, b, \nu) = l(a + M\alpha + Zb) - \tfrac{1}{2} b^T D^{-1} b - \tfrac{1}{2} \log \det(D) \quad (****)$$

The extra terms are the log of the marginal normal PDF of b.

If we observed b (which we don't), this would be the log likelihood.

Exponential Family Mixed Models (cont.)

$$f_\theta(y) = \int f_\theta(x, y) \, dx \qquad (**)$$

$$l_c(\alpha, b, \nu) = l(a + M\alpha + Zb) - \tfrac{1}{2} b^T D^{-1} b - \tfrac{1}{2} \log \det(D) \qquad (****)$$

$$l_m(\alpha, \nu) = \log \left(\int e^{l_c(\alpha, b, \nu)} \, db \right)$$

This is the actual log likelihood for the actual (observed) data.

To distinguish this from the "complete data" log likelihood, I sometimes call this the "missing data" log likelihood, but that isn't a good name. It is just the actual correct log likelihood for this problem.

Exponential Family Mixed Models (cont.)

$$l_m(\alpha, \nu) = \log \left(\int e^{l_c(\alpha, b, \nu)} \, db \right)$$

If this integral breaks up into a product of univariate integrals, which happens when there is only one random effect per individual, this integral can be done by numerical integration.

Otherwise, it is a high-dimensional integral, and intractable.

Thus we seek approximations.

Laplace Approximation

Breslow and Clayton (1993) introduced Laplace approximation for GLMM log likelihoods.

As one can tell from the eponym, it is an old method of evaluating integrals.

Laplace Approximation (cont.)

$$l_m(\alpha, \nu) = \log \left(\int e^{l_c(\alpha, b, \nu)} \, db \right)$$

- Expand the log integrand, here $l_c(\alpha, b, \nu)$, in a Taylor series in the variable of integration, here b,
- expand around a local maximum b^*, where the first derivative is zero,
- throw away all terms higher-order than second derivative,
- then the integrand is "e to a quadratic" in b, which is analytically tractable, essentially a multivariate normal distribution integral.

Laplace Approximation (cont.)

The idea is that if b^* is the actual global maximum of the integrand we get fairly good approximation where the integrand is high and maybe the rest doesn't matter.

For EFMM we know that $l_c(\alpha, b, \nu)$ is a concave function of b and the tails decrease quadratically. So Laplace approximation may do fairly well.

Laplace Approximation (cont.)

$$\begin{aligned} l(\varphi) &= \langle y, \varphi \rangle - c(\varphi) \\ l_c(\alpha, b, \nu) &= l(a + M\alpha + Zb) - \tfrac{1}{2} b^T D^{-1} b - \tfrac{1}{2} \log \det(D) \quad (****) \end{aligned}$$

$$\begin{aligned} \mu(\varphi) &= \nabla c(\varphi) \\ W(\varphi) &= \nabla^2 c(\varphi) \end{aligned}$$

(unconditional mean value parameter vector and Fisher information matrix)

$$\begin{aligned} \nabla_b l_c(\alpha, b, \nu) &= Z^T \big[y - \mu(a + M\alpha + Zb) \big] - D^{-1} b \\ \nabla_b^2 l_c(\alpha, b, \nu) &= -Z^T W(a + M\alpha + Zb) Z - D^{-1} \end{aligned}$$

Laplace Approximation (cont.)

$$\begin{aligned}
l_c(\alpha, b, \nu) &= l(a + M\alpha + Zb) - \tfrac{1}{2} b^T D^{-1} b - \tfrac{1}{2} \log \det(D) \\
\nabla_b l_c(\alpha, b, \nu) &= Z^T \bigl[y - \mu(a + M\alpha + Zb) \bigr] - D^{-1} b \\
\nabla_b^2 l_c(\alpha, b, \nu) &= -Z^T W(a + M\alpha + Zb) Z - D^{-1}
\end{aligned}$$

$$\begin{aligned}
l_c(\alpha, b, \nu) &\approx l_c(\alpha, b^*, \nu) \\
&\qquad - \tfrac{1}{2}(b - b^*)^T \bigl[Z^T W(a + M\alpha + Zb^*) Z + D^{-1} \bigr] (b - b^*)
\end{aligned}$$

considered as a function of b only, this looks like a log likelihood for a multivariate normal distribution with mean vector b^* and variance matrix

$$\bigl[Z^T W(a + M\alpha + Zb^*) Z + D^{-1} \bigr]^{-1}$$

Laplace Approximation (cont.)

The d-variate normal PDF with mean vector μ and variance matrix Σ is

$$f(x) = (2\pi)^{-d/2} \det(\Sigma)^{-1/2} \exp\left(-\tfrac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right)$$

so

$$\int \exp\left(-\tfrac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right) \, dx = (2\pi)^{d/2} \det(\Sigma)^{1/2}$$

Laplace Approximation (cont.)

$$\begin{aligned} l_c(\alpha, b, \nu) \approx{}& l_c(\alpha, b^*, \nu) \\ &- \tfrac{1}{2}(b - b^*)^T \bigl[Z^T W(a + M\alpha + Zb^*) Z + D^{-1} \bigr] (b - b^*) \end{aligned}$$

$$\begin{aligned} \int \exp\bigl(l_c(\alpha, b, \nu)\bigr)\, db \approx{}& \exp\bigl(l_c(\alpha, b^*, \nu)\bigr) \\ & (2\pi)^{d/2} \det\bigl(Z^T W(a + M\alpha + Zb^*) Z + D^{-1}\bigr)^{-1/2} \end{aligned}$$

the $-1/2$ is because the matrix in the determinant is the analog of Σ^{-1} and because $\det(A^{-1}) = \det(A)^{-1}$.

Laplace Approximation (cont.)

$$l_c(\alpha, b, \nu) = l(a + M\alpha + Zb) - \tfrac{1}{2} b^T D^{-1} b - \tfrac{1}{2} \log\det(D)$$

Thus we get

$$\begin{aligned} l_m(\alpha, \nu) &\approx l_c(\alpha, b^*, \nu) - \tfrac{1}{2} \log\det\bigl(Z^T W(a + M\alpha + Zb^*)Z + D^{-1}\bigr) \\ &= l(a + M\alpha + Zb^*) - \tfrac{1}{2}(b^*)^T D^{-1} b^* - \tfrac{1}{2} \log\det(D) \\ &\qquad - \tfrac{1}{2} \log\det\bigl(Z^T W(a + M\alpha + Zb^*)Z + D^{-1}\bigr) \\ &= l(a + M\alpha + Zb^*) - \tfrac{1}{2}(b^*)^T D^{-1} b^* \\ &\qquad - \tfrac{1}{2} \log\det\bigl(Z^T W(a + M\alpha + Zb^*)ZD + I\bigr) \end{aligned}$$

the last equality being $\det(AB) = \det(A)\det(B)$, and I denotes the identity matrix.

Laplace Approximation (cont.)

So that is our Laplace approximation approximate log likelihood

$$\begin{aligned} q(\alpha, \nu) = l(a + M\alpha + Zb^*) - \tfrac{1}{2}(b^*)^T D^{-1} b^* \\ - \tfrac{1}{2} \log \det\bigl(Z^T W(a + M\alpha + Zb^*) ZD + I\bigr) \end{aligned} \qquad (\dagger)$$

where b^* is a function of α and ν and D is a function of ν although the notation does not indicate this.

Recall that b^* is the maximizer of

$$l_c(\alpha, b, \nu) = l(a + M\alpha + Zb) - \tfrac{1}{2} b^T D^{-1} b - \tfrac{1}{2} \log \det(D)$$

and the last term on the right doesn't matter because it does not contain b.

Laplace Approximation (cont.)

Thus b^* is the maximizer of the penalized log likelihood

$$l(a + M\alpha + Zb) - \tfrac{1}{2} b^T D^{-1} b \qquad (\star)$$

where the "penalty" is the second term on the right-hand side, which looks like the penalty for ridge regression, although it arises from completely different reasoning.

For GLMM this is called penalized quasi-likelihood (PQL) because GLM allow quasi-likelihood rather than likelihood. For EFMM, PQL is a misnomer. Should be just PL, but we'll keep PQL.

Because $(\star)$ is strictly concave and decreases quadratically at infinity, the maximum b^* always exists and is unique.

The PQL step is very well behaved.

Laplace Approximation (cont.)

$$\begin{aligned} q(\alpha, \nu) = l(a + M\alpha + Zb^*) - \tfrac{1}{2}(b^*)^T D^{-1} b^* \\ - \tfrac{1}{2} \log \det\bigl(Z^T W(a + M\alpha + Zb^*) Z D + I\bigr) \end{aligned} \qquad (\dagger)$$

In contrast, the whole procedure is very ill-behaved.

To calculate the objective function $q(\alpha, \nu)$ we have to first calculate $b^*(\alpha, \nu)$ via a PQL step. This will not be exact, and the sloppier the convergence tolerance in this optimization, the sloppier our evaluation of (†).

Thus we need extreme precision in the PQL step optimization.

Even so, our evaluation of (†) will be sloppy, so "calculating" derivatives of (†) by finite difference approximation will be very sloppy.

Laplace Approximation (cont.)

$$
\begin{aligned}
q(\alpha, \nu) = l(a + M\alpha + Zb^*) - \tfrac{1}{2}(b^*)^T D^{-1} b^* \\
- \tfrac{1}{2} \log\det\bigl(Z^T W(a + M\alpha + Zb^*) ZD + I\bigr)
\end{aligned}
\qquad (\dagger)
$$

Optimizing (†) is very different from the nice smooth functions with nice smooth derivatives that can be evaluated to high precision that optimization software likes (and is designed for).

So we're not out of the woods yet.

Laplace Approximation (cont.)

$$
\begin{aligned}
q(\alpha, \nu) = l(a + M\alpha + Zb^*) - \tfrac{1}{2}(b^*)^T D^{-1} b^* & \\
- \tfrac{1}{2} \log \det\bigl(Z^T W(a + M\alpha + Zb^*) Z D + I\bigr) & \qquad (\dagger)
\end{aligned}
$$

Worse. For either Newton's method in optimization or for using the "usual" asymptotics of maximum likelihood (asymptotic variance of MLE is inverse Fisher information) we need two derivatives of (†).

But W is already the second derivative of the cumulant function c, so second derivatives of (†) involve fourth derivatives of c.

O. K. in principle. Cumulant functions are infinitely differentiable.

Bad in practice. The function `mlogl` in the `aster` package calculates the log likelihood, the first derivative vector, and the second derivative matrix. But no higher-order derivatives.

Laplace Approximation (cont.)

After all this, I have to tell you that this whole Laplace approximation scheme is a wild goose chase.

Breslow and Clayton (1993) introduce it, but don't use it, not exactly.

No other GLMM software does either AFAIK.

The `reaster` function doesn't either.

All make further approximations and kludges.

So it makes a nice story to tell people about what we're doing, but we actually do even messier things. (As if this wasn't already messy enough!)

Laplace Approximation (cont.)

The intractability of differentiating the objective function q leads to the idea of just ignoring third and higher-order derivatives of the cumulant function (they aren't really zero but we just treat them as zero).

This is equivalent to assuming that first derivatives of W are zero, which is the same as assuming that it is a constant matrix.

Breslow and Clayton (1993) use the same idea.

Of course, W isn't constant, so we adopt the schizophrenic attitude of assuming it is constant in some parts of our argument and non-constant in other parts.

Laplace Approximation (cont.)

$$
\begin{aligned}
q(\alpha, \nu) = l(a + M\alpha + Zb^*) - \tfrac{1}{2}(b^*)^T D^{-1} b^* \\
- \tfrac{1}{2} \log\det\bigl(Z^T W(a + M\alpha + Zb^*) Z D + I\bigr)
\end{aligned}
\tag{$\dagger$}
$$

Replace (†) with

$$
\begin{aligned}
q(\alpha, \nu) = l(a + M\alpha + Zb^*) - \tfrac{1}{2}(b^*)^T D^{-1} b^* \\
- \tfrac{1}{2} \log\det\bigl(Z^T \widetilde{W} Z D + I\bigr)
\end{aligned}
\tag{$\dagger\dagger$}
$$

where $\widetilde{W}$ is a constant, positive definite, symmetric matrix.

We want $\widetilde{W}$ to be close to $W(a + M\hat{\alpha} + Z\hat{b})$, where $\hat{\alpha}$ and $\hat{\nu}$ maximize (†) and $\hat{b} = b^*(\hat{\alpha}, \hat{\nu})$, but for most of the argument it doesn't matter what $\widetilde{W}$ is.

Laplace Approximation (cont.)

$$\begin{aligned} q(\alpha, \nu) = l(a + M\alpha + Zb^*) - \tfrac{1}{2}(b^*)^T D^{-1} b^* \\ - \tfrac{1}{2} \log \det\bigl(Z^T \widetilde{W} Z D + I\bigr) \end{aligned} \tag{††}$$

Now we notice that b^* maximizes (††), that is,

$$q(\alpha, \nu) = \sup_{b \in \mathbb{R}^d} p(\alpha, b, \nu)$$

where

$$\begin{aligned} p(\alpha, b, \nu) = l(a + M\alpha + Zb) - \tfrac{1}{2} b^T D^{-1} b \\ - \tfrac{1}{2} \log \det\bigl(Z^T \widetilde{W} Z D + I\bigr) \end{aligned} \tag{†††}$$

Hence we can (so long as we are imagining $\widetilde{W}$ is constant) find $\hat{\alpha}$, $\hat{b}$, and $\hat{\nu}$ by jointly maximizing (†††).

Laplace Approximation (cont.)

$$\begin{aligned} p(\alpha, b, \nu) = l(a + M\alpha + Zb) - \tfrac{1}{2} b^T D^{-1} b \\ - \tfrac{1}{2} \log \det\bigl(Z^T \widetilde{W} Z D + I\bigr) \end{aligned} \tag{†††}$$

$$q(\alpha, \nu) = \sup_{b \in \mathbb{R}^d} p(\alpha, b, \nu)$$

Key ideas:

- Can maximize q by maximizing p.
- Can calculate derivatives of q from those of p by the implicit function theorem.

(assuming $\widetilde{W}$ is a constant matrix, which it isn't).

Implicit Function Theorem

Suppose f is a differentiable vector-valued function of two vector variables, write the partial derivatives $\nabla_x f(x, y)$ and $\nabla_y f(x, y)$ (these are matrices, with row dimension the dimension of $f(x, y)$ and column dimension the dimension of x and y, respectively).

Suppose $\nabla_y f(x_0, y_0)$ is square and invertible. Then there exists a function g defined on some open neighborhood O of x_0 such that

$$f\bigl(x, g(x)\bigr) = f(x_0, y_0), \qquad x \in O,$$

and

$$\nabla g(x_0) = -\bigl[\nabla_y f(x_0, y_0)\bigr]^{-1} \nabla_x f(x_0, y_0)$$

and g is differentiable as many times as f is.

Implicit Function Theorem (cont.)

To get into the right notation for the implicit function theorem, we change notation letting ψ be the vector of all the parameters (α and ν) and write

$$\begin{aligned} p(\psi, b) &= l(a + M\alpha + Zb) - \tfrac{1}{2} b^T D^{-1} b \\ &\qquad - \tfrac{1}{2} \log \det\bigl(Z^T \widetilde{W} Z D + I\bigr) \\ q(\psi) &= p(\psi, b^*) \end{aligned}$$

where, as always, b^* is the maximizer of $p(\psi, b)$ and is a function of ψ although the notation does not explicitly indicate this.

A Digression on Notation for Derivatives

We are going to simplify notation for partial derivatives, so the notation does not get intolerably messy.

- f is a scalar-valued function f of two vector variables
- $f_x(x, y)$ is the column vector whose components are partial derivatives with respect to components of x
- $f_y(x, y)$ similarly
- $f_{xx}(x, y)$ is the square matrix whose components are second partial derivatives with respect to components of x
- $f_{yy}(x, y)$ similarly
- $f_{xy}(x, y)$ is the non-square matrix whose i, j component is $\partial^2 f(x, y) / \partial x_i \partial y_j$
- $f_{yx}(x, y) = f_{xy}(x, y)^T$

A Digression on Notation for Derivatives (cont.)

- f is a vector-valued function f of two vector variables
- $f_x(x, y)$ is the non-square matrix whose row dimension is the dimension of $f(x, y)$ and whose column dimension is the dimension of x.
- $f_y(x, y)$ similarly

We don't need to discuss second derivatives here. They would be three-index thingummies (tensors) and matrix notation wouldn't work.

Implicit Function Theorem (cont.)

Since b^* is the maximizer of $p(\psi, b)$, it solves

$$p_b(\psi, b) = 0$$

so the implicit function theorem says

$$b^*_\psi(\psi) = -p_{bb}(\psi, b^*)^{-1} p_{b\psi}(\psi, b^*)$$

where on the right-hand side b^* is a function of ψ although the notation does not explicitly indicate this.

Implicit Function Theorem (cont.)

$$q(\psi) = p(\psi, b^*)$$

Now we are ready to differentiate q using the chain rule

$$q_\psi(\psi) = p_\psi(\psi, b^*) + p_b(\psi, b^*) b^*_\psi(\psi)$$

but by definition of b^*

$$p_b(\psi, b^*) = 0$$

where, as always, b^* is a function of ψ although the notation does not explicitly indicate this. So

$$q_\psi(\psi) = p_\psi(\psi, b^*)$$

Implicit Function Theorem (cont.)

$$
\begin{aligned}
b^*_\psi(\psi) &= -p_{bb}(\psi, b^*)^{-1} p_{b\psi}(\psi, b^*) \\
q_\psi(\psi) &= p_\psi(\psi, b^*)
\end{aligned}
$$

$$
\begin{aligned}
q_{\psi\psi}(\psi) &= p_{\psi\psi}(\psi, b^*) + p_{\psi b}(\psi, b^*) b^*_\psi(\psi) \\
&= p_{\psi\psi}(\psi, b^*) - p_{\psi b}(\psi, b^*) p_{bb}(\psi, b^*)^{-1} p_{b\psi}(\psi, b^*)
\end{aligned}
$$

Implicit Function Theorem (cont.)

$$q_{\psi\psi}(\psi) = p_{\psi\psi}(\psi, b^*) - p_{\psi b}(\psi, b^*) p_{bb}(\psi, b^*)^{-1} p_{b\psi}(\psi, b^*)$$

This gives us approximate second derivatives of the approximate log likelihood which is minus the approximate observed Fisher information matrix.

Implicit Function Theorem (cont.)

$$q_{\psi\psi}(\psi) = p_{\psi\psi}(\psi, b^*) - p_{\psi b}(\psi, b^*) p_{bb}(\psi, b^*)^{-1} p_{b\psi}(\psi, b^*)$$

Why is $-q_{\psi\psi}(\psi)$ positive definite?

This form looks familiar. Suppose

$$\Sigma = \begin{pmatrix} \Sigma 11 & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}$$

is the partitioned variance matrix of a normal random vector, then (look in any multivariate statistics book) the conditional variance of the first block of variables x_1 given the second x_2 is

$$\Sigma_{11} - \Sigma_{12} \Sigma_{22}^{-1} \Sigma_{21}$$

Implicit Function Theorem (cont.)

If the conditional distribution of x_1 given x_2 is degenerate, then so is the joint distribution of x_1 and x_2.

Conversely, if Σ is positive definite, then so is $\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21}$.

Since any positive definite matrix is the variance matrix of some normal random vector, this must be a general fact about matrices. If

$$\begin{pmatrix} p_{\psi\psi}(\psi, b^*) & p_{\psi b}(\psi, b^*) \\ p_{b\psi}(\psi, b^*) & p_{bb}(\psi, b^*) \end{pmatrix}$$

is negative definite, then so is

$$q_{\psi\psi}(\psi) = p_{\psi\psi}(\psi, b^*) - p_{\psi b}(\psi, b^*) p_{bb}(\psi, b^*)^{-1} p_{b\psi}(\psi, b^*)$$

Implicit Function Theorem (cont.)

Let $\hat{\psi}$ and $\hat{b}$ be the maximizers of p. Then $\hat{b} = b^*(\hat{\psi})$.

$$\begin{pmatrix} p_{\psi\psi}(\hat{\psi}, \hat{b}) & p_{\psi b}(\hat{\psi}, \hat{b}) \\ p_{b\psi}(\hat{\psi}, \hat{b}) & p_{bb}(\hat{\psi}, \hat{b}) \end{pmatrix}$$

must be negative semidefinite and usually will be negative definite, in which case our approximation of the observed Fisher information matrix $-q_{\psi\psi}(\hat{\psi})$ will be positive definite.

Sadly, this beautiful theorem does not work with computer arithmetic (instead of real real numbers). We have run into the problem of $-q_{\psi\psi}(\hat{\psi})$ not being positive definite or even positive semidefinite when calculated with inexact computer arithmetic.

A Digression on Matrix Calculus

Differentiation of Matrix Inverse

$$AA^{-1} = I$$

so by the product rule

$$\frac{\partial A}{\partial \psi} A^{-1} + A \frac{\partial A^{-1}}{\partial \psi} = 0$$

hence

$$\frac{\partial A^{-1}}{\partial \psi} = -A^{-1} \frac{\partial A}{\partial \psi} A^{-1}$$

A Digression on Matrix Calculus (cont.)

Differentiation of Matrix Determinant

Laplace's rule for determinants says

$$\det(A) = \sum_{i=1}^{d} (-1)^{i+j} a_{ij} m_{ij}$$

where a_{ij} are the elements of A and m_{ij} (a "minor") is the determinant of A with the i-th row and j-th column deleted.

Hence

$$\frac{\partial \det(A)}{\partial a_{ij}} = (-1)^{i+j} m_{ij}$$

A Digression on Matrix Calculus (cont.)

$$\frac{\partial \det(A)}{\partial a_{ij}} = (-1)^{i+j} m_{ij}$$

Differentiation of Matrix Determinant (cont.)

Cramer's rule for determinants says that if $B = A^{-1}$, then

$$b_{ji} = \frac{(-1)^{i+j} m_{ij}}{\det(A)}$$

Putting these two together gives

$$\frac{\partial \det(A)}{\partial a_{ij}} = \det(A) b_{ji}$$

A Digression on Matrix Calculus (cont.)

$$B = A^{-1}$$
$$\frac{\partial \det(A)}{\partial a_{ij}} = \det(A) b_{ji}$$

Differentiation of Matrix Determinant (cont.)

Now the chain rule says

$$\begin{aligned}
\frac{\partial \det(A)}{\partial \psi} &= \sum_{i=1}^{d} \sum_{j=1}^{d} \frac{\partial \det(A)}{\partial a_{ij}} \frac{\partial a_{ij}}{\partial \psi} \\
&= \det(A) \sum_{i=1}^{d} \sum_{j=1}^{d} b_{ji} \frac{\partial a_{ij}}{\partial \psi} \\
&= \det(A) \operatorname{tr}\left(A^{-1} \frac{\partial A}{\partial \psi} \right)
\end{aligned}$$

A Digression on Matrix Calculus (cont.)

$$\frac{\partial \det(A)}{\partial \psi} = \det(A) \operatorname{tr}\left(A^{-1} \frac{\partial A}{\partial \psi}\right)$$

Differentiation of Matrix Determinant (cont.)

And, finally, the rule for the determinant of a logarithm says

$$\begin{aligned} \frac{\partial \log \det(A)}{\partial \psi} &= \frac{1}{\det(A)} \cdot \frac{\partial \det(A)}{\partial \psi} \\ &= \operatorname{tr}\left(A^{-1} \frac{\partial A}{\partial \psi}\right) \end{aligned}$$

A Digression on Matrix Calculus (cont.)

Summary

$$\frac{\partial A^{-1}}{\partial \psi} = -A^{-1}\frac{\partial A}{\partial \psi}A^{-1}$$

$$\frac{\partial \log \det(A)}{\partial \psi} = \operatorname{tr}\left(A^{-1}\frac{\partial A}{\partial \psi}\right)$$

Implicit Function Theorem (cont.)

$$\begin{aligned} p(\alpha, b, \nu) = l(a + M\alpha + Zb) - \tfrac{1}{2} b^T D^{-1} b \\ - \tfrac{1}{2} \log\det\bigl(Z^T \widetilde{W} Z D + I\bigr) \end{aligned} \qquad (\dagger\dagger\dagger)$$

$$\begin{aligned} p_\alpha(\alpha, b, \nu) &= M^T \bigl[y - \mu(a + M\alpha + Zb)\bigr] \\ p_b(\alpha, b, \nu) &= Z^T \bigl[y - \mu(a + M\alpha + Zb)\bigr] - D^{-1} b \\ p_{\nu_k}(\alpha, b, \nu) &= \tfrac{1}{2} b^T D^{-1} E_k D^{-1} b - \operatorname{tr}\Bigl(\bigl[Z^T \widetilde{W} Z D + I\bigr]^{-1} Z^T \widetilde{W} Z E_k\Bigr) \end{aligned}$$

Implicit Function Theorem (cont.)

$$
\begin{aligned}
p_{\alpha}(\alpha, b, \nu) &= M^T \big[y - \mu(a + M\alpha + Zb) \big] \\
p_{b}(\alpha, b, \nu) &= Z^T \big[y - \mu(a + M\alpha + Zb) \big] - D^{-1} b \\
p_{\nu_k}(\alpha, b, \nu) &= \tfrac{1}{2} b^T D^{-1} E_k D^{-1} b - \operatorname{tr}\big(\big[Z^T \widetilde{W} Z D + I \big]^{-1} Z^T \widetilde{W} Z E_k \big)
\end{aligned}
$$

$$
\begin{aligned}
p_{\alpha\alpha}(\alpha, b, \nu) &= -M^T W(a + M\alpha + Zb) M \\
p_{\alpha b}(\alpha, b, \nu) &= -M^T W(a + M\alpha + Zb) Z \\
p_{\alpha\nu_k}(\alpha, b, \nu) &= 0 \\
p_{bb}(\alpha, b, \nu) &= -Z^T W(a + M\alpha + Zb) Z - D^{-1} \\
p_{b\nu_k}(\alpha, b, \nu) &= D^{-1} E_k D^{-1} b \\
p_{\nu_k\nu_m}(\alpha, b, \nu) &= -b^T D^{-1} E_k D^{-1} E_m D^{-1} b \\
&\qquad + \operatorname{tr}\big(\big[Z^T \widetilde{W} Z D + I \big]^{-1} Z^T \widetilde{W} Z E_k \\
&\qquad\qquad \times \big[Z^T \widetilde{W} Z D + I \big]^{-1} Z^T \widetilde{W} Z E_m \big)
\end{aligned}
$$

Implicit Function Theorem (cont.)

$$p_{bb}(\alpha, b, \nu) = -Z^T W(a + M\alpha + Zb)Z - D^{-1}$$

Note that $p_{bb}(\alpha, b, \nu)$ is negative definite because $W(\,\cdot\,)$ is positive definite valued (by strict convexity of cumulant functions).

Hence it is invertible, which is a good thing because we have been writing $p_{bb}(\psi, b)^{-1}$ a lot, and invertibility is the condition for the implicit function theorem to hold.

Implicit Function Theorem (cont.)

So that finishes our treatment of derivatives of q.

It is horribly messy, but doable.

We have an approximation to observed Fisher information.

Laplace Approximation (cont.)

Should any of this actually work? If so, when?

In essence, we are "assuming" (hoping?) that the complete data log likelihood is approximately quadratic in the random effects, that is, $l_c(\alpha, b, \nu)$ is approximately quadratic in b.

All of the "usual" asymptotics of maximum likelihood is about approximating log likelihoods by quadratic functions. That theory says that as n goes to infinity (for aster models n would be the number of individuals, the number of independent bits of the data, not number of individuals times number of nodes per individual) and the number of parameters stays fixed, then the log likelihood does get closer and closer to a quadratic function.

Laplace Approximation (cont.)

Thus so long as

$$\text{number of random effects} \ll \text{number of individuals} \approx \infty$$

where $\ll$ means "much larger than" and $\approx \infty$ means "very large," we can expect that the Laplace approximation will be not bad.

Laplace Approximation (cont.)

How about that other approximation, that $W(\,\cdot\,)$ is a constant function of its arguments?

Since $W(\,\cdot\,)$ is the second derivative of the cumulant function and the second derivative of the log likelihood, that is equivalent to assuming that the saturated model log likelihood is "in asymptopia" (almost quadratic). And we have no reason to assume that, since there is no $p \ll n$ working here, where p is the number of (saturated model parameters and n is the number of individuals (in fact we have $p \gg n$ because p is number of individuals times number of nodes per individual).

Laplace Approximation (cont.)

But we didn't quite need that $W(\,\cdot\,)$ is constant.

More precisely, what we actually used ("assumed") was that $W(a + M\alpha + Zb)$ is a constant function of α and b.

And that is equivalent to assuming that the log likelihood for the aster model that treats all the random effects as fixed effects is approximately quadratic, and we expect that if

$$\text{number of effects} \ll \text{number of individuals} \approx \infty$$

where "number of effects" means number of fixed effects + number of random effects.

Laplace Approximation (cont.)

So both approximations follow the same logic and can be “justified” by appeal to the “usual” asymptotics of maximum likelihood.

What this analysis tells us that putting in one random effect per individual so

$$\text{number of random effects} \geq \text{number of individuals}$$

should not work well.

Laplace Approximation (cont.)

Presumably, there are some theorems that could be proved here if anyone wants to do it.

For now we make do with these heuristic arguments.

We repeat: if you don't believe the asymptotics, do a parametric bootstrap!

Zero Variance Components

$$
\begin{aligned}
q(\alpha, \nu) = l(a + M\alpha + Zb^*) - \tfrac{1}{2} (b^*)^T D^{-1} b^* \\
- \tfrac{1}{2} \log \det\bigl(Z^T W(a + M\alpha + Zb^*) Z D + I\bigr)
\end{aligned}
\tag{$\dagger$}
$$

D^{-1} is undefined when any of the variance components ν_k are zero.

Define σ_k such that $\nu_k = \sigma_k^2$ but, contrary to the usual practice, allow both positive and negative signs for σ_k.

Then define

$$
A = \sum_{k=1}^{K} \sigma_k E_k
$$

a diagonal matrix whose diagonal elements are elements of the vector σ having elements σ_k. Then

$$
D = A^2
$$

Zero Variance Components (cont.)

$$\begin{aligned} q(\alpha, \nu) = l(a + M\alpha + Zb^*) - \tfrac{1}{2}(b^*)^T D^{-1} b^* \\ - \tfrac{1}{2} \log \det\bigl(Z^T W(a + M\alpha + Zb^*) Z D + I\bigr) \end{aligned} \qquad (\dagger)$$

Now define

$$b = Ac$$

where c are IID standard normal, and b are normal with mean vector zero and variance matrix $D = A^2$, as required. Then

$$\begin{aligned} q(\alpha, \sigma) = l(a + M\alpha + ZAc^*) - \tfrac{1}{2}(c^*)^T c^* \\ - \tfrac{1}{2} \log \det\bigl(A Z^T W(a + M\alpha + ZAc^*) Z A + I\bigr) \end{aligned} \qquad (\spadesuit)$$

where c^* is the maximizer of the penalized likelihood for c

$$l(a + M\alpha + ZAc) - \tfrac{1}{2} c^T c$$

Zero Variance Components (cont.)

$$\begin{aligned} q(\alpha, \sigma) = l(a + M\alpha + ZAc^*) - \tfrac{1}{2}(c^*)^T c^* \\ - \tfrac{1}{2} \log \det\bigl(AZ^T W(a + M\alpha + ZAc^*)ZA + I\bigr) \end{aligned} \tag{$\spadesuit$}$$

As before, if we make W constant,

$$\begin{aligned} q(\alpha, \sigma) = l(a + M\alpha + ZAc^*) - \tfrac{1}{2}(c^*)^T c^* \\ - \tfrac{1}{2} \log \det\bigl(AZ^T \widetilde{W} ZA + I\bigr) \end{aligned} \tag{$\heartsuit$}$$

we have

$$q(\alpha, \sigma) = \sup_{c \in \mathbb{R}^d} p(\alpha, c, \sigma)$$

where

$$\begin{aligned} p(\alpha, c, \sigma) = l(a + M\alpha + ZAc) - \tfrac{1}{2} c^T c \\ - \tfrac{1}{2} \log \det\bigl(AZ^T \widetilde{W} ZA + I\bigr) \end{aligned} \tag{$\diamondsuit$}$$

Zero Variance Components (cont.)

This change-of-parameter-and-random-effects has virtues and vices.

Virtues

- Approximate log likelihood defined and continuously differentiable for all parameter values (no problem with zero variance components).
- No need for constrained optimization (σ_k allowed to be negative).

Vices

- $q_{\sigma_k}(\alpha, \sigma) = 0$ by symmetry. Cannot use first derivative equal to zero as test for solution.
- Analog of $-q_{\psi\psi}(\psi)$ for this parameterization even more computationally unstable than in the other parameterization (more likely to get failure to be positive definite due to inexactness of computer arithmetic).

Zero Variance Components (cont.)

To obtain the virtues and avoid the vices, we use this change-of-parameter-and-random-effects in optimization.

But we derived a test for whether variance components are zero using the original parameterization and random effects.

And we calculate approximate observed Fisher information using the original parameterization.

Overall Algorithm

Back to schizophrenia.

Define

$$\begin{aligned} q(\sigma) = l(a + M\tilde{\alpha} + ZA\tilde{c}) - \tfrac{1}{2}\tilde{c}^T\tilde{c} \\ - \tfrac{1}{2}\log\det\bigl(AZ^TW(a + M\tilde{\alpha} + ZA\tilde{c})ZA + I\bigr) \end{aligned} \qquad (\clubsuit)$$

where $\tilde{\alpha}$ and $\tilde{c}$ are the joint maximizers of the penalized likelihood

$$l(a + M\alpha + ZAc) - \tfrac{1}{2}c^Tc$$

so $\tilde{\alpha}$ and $\tilde{c}$ and A are functions of σ although the notation does not explicitly indicate this.

The maximizers $\tilde{\alpha}$ and $\tilde{c}$ are unique if they exist by strict concavity of l.

Overall Algorithm (cont.)

$$\begin{aligned} q(\sigma) = {} & l(a + M\tilde{\alpha} + ZA\tilde{c}) - \tfrac{1}{2} \tilde{c}^T \tilde{c} \\ & - \tfrac{1}{2} \log \det\bigl(A Z^T W(a + M\tilde{\alpha} + ZA\tilde{c}) Z A + I\bigr) \end{aligned} \qquad (\clubsuit)$$

Maximize (♣) using a no-derivative method of optimization (the `optim` function using the default `"Nelder-Mead"` method) giving solutions $\hat{\sigma}$, $\hat{\alpha} = \tilde{\alpha}(\hat{\sigma})$ and $\hat{c} = \tilde{c}(\hat{\sigma})$.

There is no justification for our optimizing jointly over α and c in the penalized likelihood step rather than only over c. These produce the same results when we assume W is constant. But here we are not assuming W is constant.

Breslow and Clayton (1993) do the same thing we do. Again, Laplace approximation is only a motivation, not actually used.

Overall Algorithm (cont.)

$$
\begin{aligned}
p(\alpha, c, \sigma) = l(a + M\alpha + ZAc) - \tfrac{1}{2} c^T c & \\
- \tfrac{1}{2} \log \det\bigl(A Z^T \widetilde{W} Z A + I\bigr) &
\end{aligned}
\tag{$\diamondsuit$}
$$

The fact that we did the wrong thing does not matter because we only use it as a preliminary estimate.

Set

$$\widetilde{W} = W\bigl(a + M\hat{\alpha} + ZA(\hat{\sigma})\hat{c}\bigr)$$

where $\hat{\alpha}$, $\hat{c}$, and $\hat{\sigma}$ are the solutions from the previous estimate.

Minimize ($\diamondsuit$) in α, c, and σ jointly using a high-precision optimization algorithm that uses second derivatives (the `trust` function in the `trust` package).

Repeat until estimates don't change (because $\widetilde{W}$ doesn't).

Overall Algorithm (cont.)

Now have putative solutions $\hat{\alpha}$, $\hat{c}$, and $\hat{\sigma}$.

Patch up, changing signs of $\hat{\sigma}$ so all are positive and reversing the signs of the corresponding components of $\hat{c}$ so

$$\hat{b} = A(\hat{\sigma})\hat{c}$$

still makes sense. Estimates of variance components are $\hat{\nu}_k = \hat{\sigma}_k^2$.

Calculate approximate observed Fisher information in the original parameterization $-q_{\psi\psi}(\hat{\psi})$.

Overall Algorithm (cont.)

Test for Zero Variance Components

$$\bar{p}_{\nu_j}(\alpha, b, \nu) - \frac{1}{4} \sum_{\substack{i \in I \\ d_{ii} = \nu_j}} \bar{p}_{b_i}(\alpha, b, \nu)^2 \qquad (\Box)$$

where $\bar{p}$ is the part of p that does not contain D^{-1}.

A sufficient condition for $\nu_j > 0$ is $(\Box)$ being negative. Conversely, we appear to have $\hat{\nu}_j = 0$ if $(\Box)$ is nonnegative.

For once we will spare you the derivation (which is long and involves a lot of optimization theory that statisticians don't know). A full derivation is in TR 692.

REML?

This is not the place to explain REML (restricted maximum likelihood because we do not use it).

The general idea is that it produces inflated estimates of variance components that are roughly analogous to dividing by $n - p$ in regression. But it does not produce unbiased estimators of variance components.

REML is based on a marginal likelihood that is a function of variance components only (so, strictly speaking, it does not estimate fixed effects).

There is always a question whether REML or plain ML is preferred.

We like ML, especially when inference about fixed effects of primary interest.

REML? (cont.)

Breslow and Clayton (1993) depart even further from simple Laplace approximation, doing something they say is an analog of REML.

Of course, it has none of the properties of real REML for LMM.

AFAIK, no one uses exactly the REML-like scheme of Breslow and Clayton (1993), although many implementers of GLMM software use some REML-like scheme of their own.

We don't.

Our algorithm is as described above. No "REML-like" aspects.

A Formula Attributed to Lewis

For any missing data model with complete data density $f_\theta(x, y)$ with x missing and y observed and log likelihood

$$l(\theta) = \log \int f_\theta(x, y) \, dx$$

we have

$$\begin{aligned} \nabla^2 l(\theta) = E_\theta\{\nabla^2 \log f_\theta(X, Y) \mid Y = y\} \\ + \operatorname{var}_\theta\{\nabla \log f_\theta(X, Y) \mid Y = y\} \end{aligned}$$

(a formula attributed to Lewis, *Journal of the Royal Statistical Society, Series B*, 1982, in the EM literature, although also found in Sundberg, *Scandinavian Journal of Statistics*, 1974).

A Formula Attributed to Lewis (cont.)

$$
\begin{aligned}
\nabla^2 l(\theta) = E_\theta\{ & \nabla^2 \log f_\theta(X, Y) \mid Y = y\} \\
& + \operatorname{var}_\theta\{\nabla \log f_\theta(X, Y) \mid Y = y\}
\end{aligned}
$$

The left-hand side is a negative definite matrix for θ near the MLE. The second term on the right-hand side is positive definite (like all variance matrices). Hence the first term on the right-hand side must be even more negative definite than the left-hand side.

The left-hand side may be close to singular although neither term on the right-hand side is.

Why missing data is hard: observed Fisher information is much less than if missing data were observed.

A Formula Attributed to Lewis (cont.)

$$
\begin{aligned}
\nabla^2 l(\theta) = E_\theta\{\nabla^2 \log f_\theta(X, Y) \mid Y = y\} \\
+ \operatorname{var}_\theta\{\nabla \log f_\theta(X, Y) \mid Y = y\}
\end{aligned}
$$

Our analog

$$
\begin{aligned}
q_{\psi\psi}(\psi) &= p_{\psi\psi}(\psi, b^*) + p_{\psi b}(\psi, b^*) b^*_\psi(\psi) \\
&= p_{\psi\psi}(\psi, b^*) - p_{\psi b}(\psi, b^*) p_{bb}(\psi, b^*)^{-1} p_{b\psi}(\psi, b^*)
\end{aligned}
$$

Not the same, but right-hand side is the difference of negative definite matrices. The left-hand side may be close to singular although neither term on the right-hand side is.

Why random effects are hard: approximate observed Fisher information is much less than if random effects were observed.

Peas

Partridge pea (*Chamaecrista fasciculata*) collected and first analyzed by Julie Etterson (Etterson, *Evolution*, 2004; Etterson and Shaw, *Science*, 2001). (The *Evolution* reference is two papers, a "I" and a "II".)

These data have been re-analyzed using aster models twice in Shaw, et al. (2008) and once more in Geyer, et al. (2013). Only the latter uses random effects, and that is the one we present here.

There are several R data sets containing parts of these data. The one we want here is in the dataset `chamae3` in the `aster` package.

```
> library(aster)
> data(chamae3)
> dim(chamae3)
```

```
[1] 19062    11
```

Peas (cont.)

The graph is

$$1 \xrightarrow{\text{Ber}} y_1 \xrightarrow{\text{0-Poi}} y_2$$

y_1 being an indicator of whether any fruits were produced, y_2 being the count of the number of fruits produced, the unconditional distribution of y_1 being Bernoulli, and the conditional distribution of y_2 given y_1 being zero-truncated Poisson.

```
> levels(chamae3$varb)
```

```
[1] "fecund" "fruit"
```

Peas (cont.)

```
> sapply(chamae3, class)

     SIRE       DAM       POP      SITE       ROW
 "factor"  "factor"  "factor"  "factor" "integer"
      BLK      varb      resp        id      root
 "factor"  "factor" "numeric" "integer" "numeric"
      fit
"numeric"
```

The variables `varb`, `resp`, `id`, `root`, and `fit` are as usual for an aster model (the latter the indicator of “fitness nodes” which here are terminal nodes).

Peas (cont.)

C. fasciculata grows in the Great Plains of North America from southern Minnesota to Mexico. Three populations were sampled in the following locations

1. Kellog-Weaver Dunes, Wabasha County, Minnesota
2. Konza Prairie, Riley County, Kansas
3. Pontotoc Ridge, Pontotoc County, Oklahoma

(the numbers for the list items correspond to the levels of the variable `POP` in the dataset). These sites are progressively more arid from north to south and also differ in other characteristics.

Peas (cont.)

Seed pods were collected from 200 plants in each population, crosses were done, germinated, and raised in the greenhouse. The parent plants are indicated by the variables `SIRE` and `DAM` in the dataset. The seedlings from the greenhouse were planted using a randomized block design (Etterson, 2004) in three field sites

- `"O"` Robert S. Kerr Environmental Research Center, Ada, Oklahoma
- `"K"` Konza Prairie Research Natural Area, Manhatten, Kansas
- `"M"` University of Minnesota, St. Paul Minnesota

(the characters for the list items correspond to the levels of the variable `SITE` in the dataset). The Oklahoma field site was 30 km northwest of the Oklahoma population; the Kansas field site was 5 km from the Kansas population; the Minnesota field site was 110 km northwest of the Minnesota population.

Peas (cont.)

That explains `SIRE`, `DAM`, `POP`, and `SITE`.

`BLK` (block) is nested within site. We did not use `ROW` in our re-analysis.

In our re-analysis, we analyze each of the nine population-site pairs independently. So we split the data on `POP` and `SITE`. We do not include these variables in the model.

All nine re-analyses are in TR 692. Here we show only one (Kansas-Kansas).

```
> inies <- as.character(chamae3$SITE) == "K"
> inies <- inies & as.character(chamae3$POP) == "2"
> subdata <- subset(chamae3, inies)
> dim(subdata)
```

```
[1] 2342   11
```

Peas (cont.)

Now that we are analyzing only one site and only one population, our remaining predictors are `BLK`, `SIRE`, and `DAM`.

Quantitative genetics theory says we should treat `SIRE` and `DAM` as random effects representing the genetic contribution of parents to offspring.

Peas (cont.)

```
> pred <- c(0,1)
> fam <- c(1,3)
> woof <- suppressWarnings(try(load("peas.rda"),
+     silent = TRUE))
> if (inherits(woof, "try-error")) {
+ rout <- reaster(resp ~ varb + fit : BLK,
+     list(sire = ~ 0 + fit:SIRE, dam = ~ 0 + fit:DAM),
+     pred, fam, varb, id, root, data = subdata)
+ save(rout, file = "peas.rda")
+ }
```

Peas (cont.)

```
> summary(rout)

Call:
reaster.formula(fixed = resp ~ varb + fit:BLK, random = list(sire = ~0 +
    fit:SIRE, dam = ~0 + fit:DAM), pred = pred, fam = fam, varvar = varb,
    idvar = id, root = root, data = subdata)



Fixed Effects:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -88.178261   1.429153  -61.70   <2e-16 ***
varbfruit    93.779086   1.429651   65.60   <2e-16 ***
fit:BLK1     -0.563040   0.005827  -96.63   <2e-16 ***
fit:BLK2     -0.277439   0.005319  -52.16   <2e-16 ***
fit:BLK3     -0.068424   0.005062  -13.52   <2e-16 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Square Roots of Variance Components (P-values are one-tailed):
     Estimate Std. Error z value Pr(>|z|)/2
sire  0.12085    0.02999   4.029    2.8e-05 ***
dam   0.23771    0.01700  13.986    < 2e-16 ***
---
Signif. codes:
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Peas (cont.)

Everything is highly statistically significant.

This was not true in all nine of the separate population-site analyses. Geyer, et al. (2013) say

> We found that the sire variance components for the Minnesota and Oklahoma natural population are not close to statistically significant at the Minnesota field site. All the other sire variance components are at least borderline statistically significant.

All nine aster analyses are all in Appendix B of TR 692.

The *P*-values called not close to statistically significant are 0.5 and 0.381.

The *P*-values called at least borderline statistically significant are 0.0565, 0.0406, 0.0305, 0.00681, 0.00092. 7.99×10^{-5}, and 2.8×10^{-5}.

Peas (cont.)

Geyer et al. (2013) also say

> We focus on sire effects although dam effects are also treated as random effects because in this experimental design sire effects are expected to correspond closely to pure breeding values but dam effects will be confounded with maternal and dominance effects.

"Breeding values" are another name for additive genetic effects. They are the ones that figure in evolutionary theory.

The dam effects were all highly statistically significant (all reported as $P < 2 \times 10^{-16}$), so we need them in the model, but they are not of scientific interest with this experimental design.

Peas (cont.)

So what are these sire effects? Since they are not parameters, we do not have parameter estimates of them. But we do have the $\hat{b}$ values — the PQL estimates when α and ν are at the MLE.

```
> bhat <- rout$b
> head(names(bhat))

[1] "fit:SIRE2001" "fit:SIRE2003" "fit:SIRE2010"
[4] "fit:SIRE2012" "fit:SIRE2016" "fit:SIRE2020"

> bhat.sire <- bhat[grep("SIRE", names(bhat))]
> length(bhat.sire)

[1] 50
```

Peas (cont.)

```
> stem(bhat.sire)

  The decimal point is 1 digit(s) to the left of the |

  -2 | 30
  -1 |
  -1 | 32000
  -0 | 88865555
  -0 | 3221111000
   0 | 222233344
   0 | 55566667889
   1 | 0014
   1 | 5
```

Peas (cont.)

Looks fairly normal.

Of course they would because the ridge-type penalty tries to make them so.

This is a well known fact about least squares. The residuals from least squares look a lot more normal than the errors.

This is the reason for looking at leave-one-out residuals and at residuals from robust regression methods.

Geyer, et al. (2013) checked this by redoing the PQL step with much smaller penalty, that is, much larger ν than $\hat{n}$. This spreads out the resulting b^*, but it just approximately multiplies them by a constant so they still look normal (Figures 7 through 9 in TR 692).

Peas (cont.)

We also want to map these $\hat{b}$ values from the canonical parameter scale to the mean value parameter scale (actual expected fitness).

This is a different form of "prediction" than we looked at before. Here it is the random effects that are of interest. We don't want to just make them zero in the "prediction".

Or, more precisely, we do want to set the dam effects to zero (because we are not interested in them) but do not want to set the sire effects to zero.

Again we see why there is no function `predict.reaster`. The many jobs that are some sort of "prediction" are too much for any one function. Worse, it seems that every new application needs a new kind of "prediction" so we cannot yet imagine all the kinds of "prediction" that someone might want to do.

Peas (cont.)

We need the nonlinear mapping $\varphi \to \mu$. There are two functions in the `aster` package that do this.

- The function `predict.aster` goes from β to θ, φ, ξ, or μ (with some fooling around necessary to get ξ).
- The function `astertransform` goes from either θ or φ to any of θ, φ, ξ, and μ.

Essentially, the first does parameter transformations for submodels (including submodels for hypothetical data), and the second does parameter transformations for the saturated model.

So for this job we use the former, following TR 692, Section 8.6.

Peas (cont.)

To use `predict.aster` we need a fixed effect model fit and there is one in `rout`

```
> class(rout$obj)
```

```
[1] "aster"           "asterOrReaster"
```

Peas (cont.)

```
> hoom <- predict(rout$obj, newcoef = rout$alpha)
> hoom <- matrix(hoom, ncol = 2)
> hoom <- hoom[ , 2]
> unique(hoom)
```

```
[1] 154.1282 205.0772 252.7500 270.6497
```

These are predictions for the second node of the graph, unconditional expected fitness for each block.

Peas (cont.)

Here we use the prediction for the first individual in the data.

```
> hoom[1]
```

```
[1] 154.1282
```

```
> as.character(subdata$BLK)[1]
```

```
[1] "1"
```

who appears to be in block 1.

Peas (cont.)

```
> rout$alpha
 (Intercept)    varbfruit     fit:BLK1     fit:BLK2
-88.17826056  93.77908583  -0.56304041  -0.27743889
    fit:BLK3
 -0.06842436
```

Our strategy is to add a component of $\hat{b}$ that we want to map to the mean value parameter scale to the `fit:BLK1` fixed effect, predict, and take the value of the second node for the first individual.

Peas (cont.)

Here's a function to do this.

```
> map <- function(b) {
+     stopifnot(length(b) == 1)
+     stopifnot(is.finite(b))
+     alpha <- rout$alpha
+     alpha[3] <- alpha[3] + b
+     hoom <- predict(rout$obj, newcoef = alpha)
+     hoom <- matrix(hoom, ncol = 2)
+     return(hoom[1, 2])
+ }
```

Peas (cont.)

The following code makes the figure on the next slide

```
> fred <- Vectorize(map)
> curve(fred, from = - 1 / 2, to = 1 / 2, xlab = "b",
+     ylab = expression(mu(b)))
```

Peas (cont.)

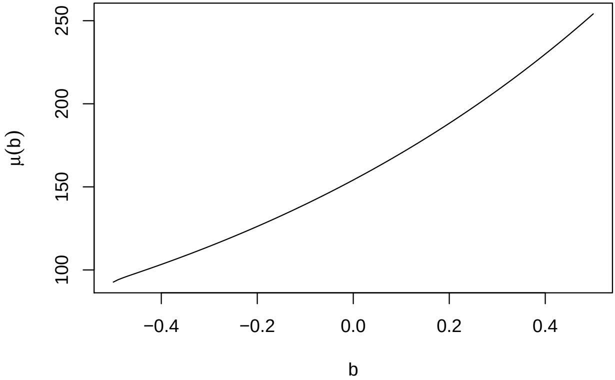

Figure : Mapping random effects from canonical parameter scale to mean value parameter scale.

Peas (cont.)

And voila!

```
> bhat.sire.mu <- fred(bhat.sire)
> stem(bhat.sire.mu)

  The decimal point is 1 digit(s) to the right of the |

  12 | 26
  13 | 57
  14 | 000222567779
  15 | 122333444777899
  16 | 00022333445779
  17 | 0127
  18 | 0
```

Peas (cont.)

We can, perhaps, consider the differences of these from the value for $b = 0$ to be the additive genetic effects for fitness

```
> stem(bhat.sire.mu - map(0))

  The decimal point is 1 digit(s) to the right of the |

  -3 | 2
  -2 | 8
  -1 | 97444222
  -0 | 987775422111000
   0 | 33345556688999
   1 | 000235678
   2 | 36
```

The Fundamental Theorem of Natural Selection

In R. A. Fisher's 1930 book *The Genetical Theory of Natural Selection* he formulated the "fundamental theorem of natural selection" (FTNS) which states that the expected change in fitness per generation is equal to the additive genetic variance for fitness divided by mean fitness.

The Fundamental Theorem of Natural Selection (cont.)

FTNS is like Lande-Arnold analysis but without phenotypic trait “predictors”.

Lande-Arnold beta “predicts” change in *those phenotypic traits* in one generation of selection.

FTNS “predicts” change in *expected fitness itself* in one generation of selection.

The Fundamental Theorem of Natural Selection (cont.)

FTNS makes qualitative sense. If there were no genetic variation there could be no evolution. It is not so obvious that only *additive* genetic variance matters, but that is what quantitative genetic theory says.

The Fundamental Theorem of Natural Selection (cont.)

We follow TR 696 (yet another TR) in this bit.

Here is the main idea.

We know additive genetic variance on the canonical parameter scale.

We need to transfer it to the mean value parameter scale.

How do we do that? The delta method.

How do we get the Jacobian matrix we need for the delta method?

The R function `predict.aster` returns it as the `gradient` component of the returned object when `se.fit = TRUE`.

The Fundamental Theorem of Natural Selection (cont.)

Let us return to our $\beta \to \mu$ mapping and add the gradient.

```
> hoom <- predict(rout$obj, newcoef = rout$alpha,
+     se.fit = TRUE)
> goom <- hoom$gradient
> dim(goom)

[1] 2342    5

> colnames(goom)

NULL
```

The Fundamental Theorem of Natural Selection (cont.)

```
> names(rout$alpha)

[1] "(Intercept)" "varbfruit"   "fit:BLK1"
[4] "fit:BLK2"    "fit:BLK3"

> moom <- goom[ , 3]
> moom <- matrix(moom, ncol = 2)
> moom[1 , 2]

[1] 154.1282
```

There is our derivative of mean fitness for an individual in block 1 with respect to the beta for block 1, the beta to which we added *b* in our mapping of *b* from β to μ.

The Fundamental Theorem of Natural Selection (cont.)

Now the “Jacobian” (here just an ordinary derivative) is squared in the delta method

```
> names(rout$nu)

[1] "sire" "dam"

> 4 * moom[1 , 2]^2 * rout$nu[1] / map(0)

    sire
9.004205
```

The factor of 4 comes from the fact that for this particular experimental design (not in general) the variance due to sires is 1/4 of the additive genetic variance.

Aster Models with Random Effects

And that's it.

Everything is moderately straightforward except:

- Cannot do models with one random effect per individual.
- Cannot really trust this stuff. Have to parametric bootstrap for complete confidence.
- "Prediction" means different things in different contexts, and we probably haven't even thought of them all. There is no `predict.reaster` that just does what you want.