An Analysis of Life History Data on the Purple Coneflower (*Echinacea angustifolia*) using the R Package Aster

Charles J. Geyer Ruth G. Shaw Stuart Wagenius

August 6, 2005

1 Preliminaries

Load aster and data.

```
> library(aster)
> data(echinacea)
> names(echinacea)
```

```
 [1] "hdct02" "hdct03" "hdct04" "site"   "ewloc"  "nsloc"
 [7] "ld02"   "fl02"   "ld03"   "fl03"   "ld04"   "fl04"
```

the variables with numbers in the names are the columns of the response matrix of the aster model. The variables `ld0`x (where x is a digit) are the survival indicator variables for year 200x (one for alive, zero for dead). The variables `fl0`x are the flowering indicator variables (one for any flowers, zero for none). The variables `hdct0`x are the inflorescence (flower head) count variables (number of flower heads). The variables without numbers are other predictors. The variables `ewloc` and `nsloc` are spatial predictions (east-west and north-south location, respectively). The variable `site` is the site of origin of the plant, so plants with different values of `site` may be more genetically diverse than those with the same values of `site`.

Make graph (of the graphical model, specified by a function p given by an R vector `pred`).

```
> pred <- c(0, 1, 2, 1, 2, 3, 4, 5, 6)
> fam <- c(1, 1, 1, 1, 1, 1, 3, 3, 3)
```

Reshape data.

```
> vars <- c("ld02", "ld03", "ld04", "fl02", "fl03",
+     "fl04", "hdct02", "hdct03", "hdct04")
> redata <- reshape(echinacea, varying = list(vars),
+     direction = "long", timevar = "varb", times = as.factor(vars),
+     v.names = "resp")
> redata <- data.frame(redata, root = 1)
> names(redata)

[1] "site"  "ewloc" "nsloc" "varb"  "resp"  "id"    "root"
```

2 Modeling

2.1 First Model

For our first model we try something simple (moderately simple). We have no site effects. We put in a mean and effect of north-south and east-west position for each of the nine variables. That gives us $3 \times 9 = 27$ parameters.

```
> out1 <- aster(resp ~ varb + varb:nsloc + varb:ewloc,
+     pred, fam, varb, id, root, data = redata)
> summary(out1, show.graph = TRUE)

Call:
aster.formula(formula = resp ~ varb + varb:nsloc + varb:ewloc,
    pred = pred, fam = fam, varvar = varb, idvar = id, root = root,
    data = redata)


Graphical Model:
 variable predecessor family
 ld02     root        bernoulli
 ld03     ld02        bernoulli
 ld04     ld03        bernoulli
 fl02     ld02        bernoulli
 fl03     ld03        bernoulli
 fl04     ld04        bernoulli
 hdct02   fl02        non.zero.poisson
 hdct03   fl03        non.zero.poisson
 hdct04   fl04        non.zero.poisson

                  Estimate Std. Error z value Pr(>|z|)
```

```
(Intercept)      -1.6418322  0.1928540  -8.513  < 2e-16 ***
varbfl03         -0.2585034  0.2961442  -0.873  0.38272
varbfl04         -0.3368137  0.2601966  -1.294  0.19551
varbhdct02        1.9661735  0.2675730   7.348 2.01e-13 ***
varbhdct03        1.9185775  0.2203482   8.707  < 2e-16 ***
varbhdct04        2.4696479  0.2001443  12.339  < 2e-16 ***
varbld02         -0.9772646  0.3502261  -2.790  0.00526 **
varbld03          1.0373523  0.4281242   2.423  0.01539 *
varbld04          4.1243999  0.3469256  11.888  < 2e-16 ***
varbfl02:nsloc    0.0838694  0.0265213   3.162  0.00157 **
varbfl03:nsloc    0.0655407  0.0302581   2.166  0.03031 *
varbfl04:nsloc    0.0698900  0.0244358   2.860  0.00423 **
varbhdct02:nsloc -0.0179987  0.0116776  -1.541  0.12324
varbhdct03:nsloc  0.0001156  0.0138761   0.008  0.99335
varbhdct04:nsloc -0.0034831  0.0074230  -0.469  0.63891
varbld02:nsloc   -0.0463586  0.0376371  -1.232  0.21805
varbld03:nsloc    0.0291040  0.0527152   0.552  0.58088
varbld04:nsloc    0.0339060  0.0405000   0.837  0.40249
varbfl02:ewloc    0.0460555  0.0267844   1.719  0.08553 .
varbfl03:ewloc   -0.0060486  0.0290874  -0.208  0.83527
varbfl04:ewloc   -0.0094399  0.0239818  -0.394  0.69385
varbhdct02:ewloc  0.0024187  0.0120993   0.200  0.84156
varbhdct03:ewloc  0.0230630  0.0135295   1.705  0.08826 .
varbhdct04:ewloc  0.0084179  0.0072661   1.159  0.24665
varbld02:ewloc    0.0149197  0.0357336   0.418  0.67629
varbld03:ewloc   -0.0285742  0.0517154  -0.553  0.58059
varbld04:ewloc    0.0048502  0.0409964   0.118  0.90582
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1
```

2.2 Second Model

So now we put in site.

```
> levels(echinacea$site)

[1] "AA"      "Eriley"  "Lf"      "Nessman" "NWLF"    "SPP"
[7] "Stevens"
```

Let us put `site` in only at the top level in this model (just to see what happens). In order to do that we have to add a predictor that "predicts" the top level.

```
> hdct <- grep("hdct", as.character(redata$varb))
> hdct <- is.element(seq(along = redata$varb), hdct)
> redata <- data.frame(redata, hdct = as.integer(hdct))
> names(redata)

[1] "site"  "ewloc" "nsloc" "varb"  "resp"  "id"    "root"
[8] "hdct"

> out2 <- aster(resp ~ varb + varb:nsloc + varb:ewloc +
+     hdct * site - site, pred, fam, varb, id, root,
+     data = redata)
> summary(out2)

Call:
aster.formula(formula = resp ~ varb + varb:nsloc + varb:ewloc +
    hdct * site - site, pred = pred, fam = fam, varvar = varb,
    idvar = id, root = root, data = redata)

                  Estimate Std. Error z value Pr(>|z|)
(Intercept)      -1.6259835  0.1925029  -8.447  < 2e-16 ***
varbfl03         -0.2591673  0.2953707  -0.877  0.38025
varbfl04         -0.3272159  0.2594037  -1.261  0.20716
varbhdct02        2.0516689  0.2751442   7.457 8.87e-14 ***
varbhdct03        2.0032366  0.2298569   8.715  < 2e-16 ***
varbhdct04        2.5544660  0.2106706  12.125  < 2e-16 ***
varbld02         -0.9847880  0.3499238  -2.814  0.00489 **
varbld03          1.0268923  0.4279181   2.400  0.01641 *
varbld04          4.1086115  0.3467427  11.849  < 2e-16 ***
varbfl02:nsloc    0.0835490  0.0264359   3.160  0.00158 **
varbfl03:nsloc    0.0655031  0.0301469   2.173  0.02980 *
varbfl04:nsloc    0.0698326  0.0243307   2.870  0.00410 **
varbhdct02:nsloc -0.0184434  0.0116321  -1.586  0.11284
varbhdct03:nsloc -0.0004162  0.0138183  -0.030  0.97597
varbhdct04:nsloc -0.0038684  0.0073788  -0.524  0.60010
varbld02:nsloc   -0.0459235  0.0376414  -1.220  0.22245
varbld03:nsloc    0.0295182  0.0527428   0.560  0.57571
varbld04:nsloc    0.0339706  0.0405350   0.838  0.40200
varbfl02:ewloc    0.0462769  0.0267337   1.731  0.08345 .
varbfl03:ewloc   -0.0054404  0.0289865  -0.188  0.85112
varbfl04:ewloc   -0.0089080  0.0239051  -0.373  0.70942
```

```
varbhdct02:ewloc  0.0028066  0.0120847   0.232  0.81635
varbhdct03:ewloc  0.0231490  0.0134693   1.719  0.08568 .
varbhdct04:ewloc  0.0086106  0.0072716   1.184  0.23636
varbld02:ewloc    0.0153283  0.0357260   0.429  0.66789
varbld03:ewloc   -0.0283036  0.0517141  -0.547  0.58417
varbld04:ewloc    0.0048550  0.0409914   0.118  0.90572
hdct:siteEriley  -0.1782285  0.0893919  -1.994  0.04618 *
hdct:siteLf      -0.1617692  0.0960825  -1.684  0.09225 .
hdct:siteNessman -0.3152202  0.1387855  -2.271  0.02313 *
hdct:siteNWLF    -0.1076672  0.0832346  -1.294  0.19582
hdct:siteSPP      0.0198694  0.0862319   0.230  0.81777
hdct:siteStevens -0.1288083  0.0891936  -1.444  0.14870
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

Original predictor variables dropped (aliased)
     hdct

> anova(out1, out2)

Analysis of Deviance Table

Model 1: resp ~ varb + varb:nsloc + varb:ewloc
Model 2: resp ~ varb + varb:nsloc + varb:ewloc + hdct * site - site
  Model Df Model Dev Df Deviance P(>|Chi|)
1       27   2717.39
2       33   2701.26  6    16.13      0.01
```

Comment The last bit of the summary that the "original predictor" `hdct` was "dropped (aliased)" means just what it (tersely) says. The R formula mini-language as implemented in the R functions `model.frame` and `model.matrix` produces a model matrix that is not full rank. In order to estimate anything we must drop some dummy variable that it constructed. In this particular case the (dummy variable that is the indicator of) `hdct` is equal to the sum of the (dummy variables that are the indicators of) `varbhdct02`, `varbhdct03`, and `varbhdct04`. Thus we must drop one of these variables for the model to be identifiable. So `aster` does.

Comment The reason for the `- site` in the formula is not obvious. In fact, we originally did not write the formula this way and got the wrong

model (see "Tenth Model" in Section 2.10 below). It took some grovelling in various bits of R documentation to come up with this `- site` trick, but once you see it, the effect is clear.

We want site effects only at the "hdct" level. But the `hdct * site` crosses the `hdct` indicator variable, which has two values (zero and one) with the `site` variable, which has seven values (the seven sites), giving 14 parameters, one of which R drops (because it is aliased with the intercept). But that's not what we want. We don't want `site` effects at the "non-hdct" levels. The way the R formula mini-language allows us to specify that is `- site` which means to leave out the site main effects (7 fewer parameters, leaving 6) and we see that we do indeed have 6 degrees of freedom difference between models one and two.

2.3 Third Model

Let us now put `site` in at all levels in this model.

```
> level <- gsub("[0-9]", "", as.character(redata$varb))
> redata <- data.frame(redata, level = as.factor(level))
> out3 <- aster(resp ~ varb + varb:nsloc + varb:ewloc +
+     level * site, pred, fam, varb, id, root, data = redata)
> summary(out3)

Call:
aster.formula(formula = resp ~ varb + varb:nsloc + varb:ewloc +
    level * site, pred = pred, fam = fam, varvar = varb, idvar = id,
    root = root, data = redata)

                     Estimate Std. Error z value Pr(>|z|)
(Intercept)          -1.937453   0.423538  -4.574 4.77e-06 ***
varbfl03             -0.268508   0.293765  -0.914  0.36071
varbfl04             -0.303001   0.257928  -1.175  0.24009
varbhdct02            2.477882   0.546517   4.534 5.79e-06 ***
varbhdct03            2.433586   0.525941   4.627 3.71e-06 ***
varbhdct04            2.972531   0.516328   5.757 8.56e-09 ***
varbld02             -0.739801   0.582421  -1.270  0.20401
varbld03              1.256976   0.632563   1.987  0.04691 *
varbld04              4.322854   0.581362   7.436 1.04e-13 ***
siteEriley            0.807390   0.451694   1.787  0.07386 .
siteLf                0.877245   0.481248   1.823  0.06833 .
siteNessman          -0.602180   0.681430  -0.884  0.37686
```

```
siteNWLF              -0.111507   0.434671  -0.257  0.79754
siteSPP                0.541625   0.450808   1.201  0.22958
siteStevens            0.112023   0.465345   0.241  0.80976
varbfl02:nsloc         0.083795   0.026329   3.183  0.00146 **
varbfl03:nsloc         0.066937   0.029982   2.233  0.02558 *
varbfl04:nsloc         0.070146   0.024162   2.903  0.00370 **
varbhdct02:nsloc      -0.018769   0.011491  -1.633  0.10240
varbhdct03:nsloc      -0.001417   0.013619  -0.104  0.91715
varbhdct04:nsloc      -0.004397   0.007261  -0.606  0.54477
varbld02:nsloc        -0.044854   0.037640  -1.192  0.23340
varbld03:nsloc         0.030181   0.052719   0.572  0.56699
varbld04:nsloc         0.034831   0.040522   0.860  0.39003
varbfl02:ewloc         0.035972   0.026717   1.346  0.17817
varbfl03:ewloc        -0.015152   0.028934  -0.524  0.60050
varbfl04:ewloc        -0.018685   0.023867  -0.783  0.43371
varbhdct02:ewloc       0.006926   0.012016   0.576  0.56435
varbhdct03:ewloc       0.026914   0.013389   2.010  0.04441 *
varbhdct04:ewloc       0.012554   0.007241   1.734  0.08298 .
varbld02:ewloc         0.013009   0.035812   0.363  0.71641
varbld03:ewloc        -0.030830   0.051730  -0.596  0.55119
varbld04:ewloc         0.002422   0.040976   0.059  0.95287
levelhdct:siteEriley  -1.341288   0.587473  -2.283  0.02242 *
levelld:siteEriley    -0.560351   0.553094  -1.013  0.31100
levelhdct:siteLf      -1.408221   0.631418  -2.230  0.02573 *
levelld:siteLf        -0.674658   0.588417  -1.147  0.25156
levelhdct:siteNessman  0.418459   0.893928   0.468  0.63970
levelld:siteNessman    0.922668   0.778683   1.185  0.23605
levelhdct:siteNWLF     0.046822   0.553566   0.085  0.93259
levelld:siteNWLF       0.140229   0.531548   0.264  0.79192
levelhdct:siteSPP     -0.702929   0.573865  -1.225  0.22061
levelld:siteSPP       -0.479659   0.561548  -0.854  0.39301
levelhdct:siteStevens -0.215472   0.593594  -0.363  0.71661
levelld:siteStevens   -0.240029   0.569320  -0.422  0.67331
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

Original predictor variables dropped (aliased)
     levelhdct
     levelld
```

```
> anova(out1, out2, out3)

Analysis of Deviance Table

Model 1: resp ~ varb + varb:nsloc + varb:ewloc
Model 2: resp ~ varb + varb:nsloc + varb:ewloc + hdct * site - site
Model 3: resp ~ varb + varb:nsloc + varb:ewloc + level * site
  Model Df Model Dev Df Deviance P(>|Chi|)
1       27   2717.39
2       33   2701.26  6    16.13      0.01
3       45   2663.55 12    37.71 0.0001715
```

Comment This would finish a sensible analysis, but we're really not sure we have dealt with the "geometry" (the variables `ewloc` and `nsloc`) correctly.

2.4 Fourth Model, Less Geometry

Thus we experiment with different ways to put in the spatial effects. First we reduce the geometry to a product, either year or level.

```
> year <- gsub("[a-z]", "", as.character(redata$varb))
> year <- paste("yr", year, sep = "")
> redata <- data.frame(redata, year = as.factor(year))
> out4 <- aster(resp ~ varb + (level + year):(nsloc +
+     ewloc) + level * site, pred, fam, varb, id, root,
+     data = redata)
> summary(out4)

Call:
aster.formula(formula = resp ~ varb + (level + year):(nsloc +
    ewloc) + level * site, pred = pred, fam = fam, varvar = varb,
    idvar = id, root = root, data = redata)

                     Estimate Std. Error z value Pr(>|z|)
(Intercept)          -1.875208   0.419723  -4.468 7.91e-06 ***
varbfl03             -0.384991   0.268534  -1.434  0.15166
varbfl04             -0.373643   0.244457  -1.528  0.12640
varbhdct02            2.387001   0.541799   4.406 1.05e-05 ***
varbhdct03            2.399402   0.522036   4.596 4.30e-06 ***
varbhdct04            2.913906   0.513754   5.672 1.41e-08 ***
varbld02             -0.679149   0.562506  -1.207  0.22729
```

```
varbld03               1.114165   0.611123   1.823  0.06828 .
varbld04               4.225862   0.573484   7.369 1.72e-13 ***
siteEriley             0.809271   0.451391   1.793  0.07300 .
siteLf                 0.878454   0.480895   1.827  0.06774 .
siteNessman           -0.600230   0.681076  -0.881  0.37816
siteNWLF              -0.111683   0.434221  -0.257  0.79702
siteSPP                0.540528   0.450468   1.200  0.23017
siteStevens            0.110291   0.464961   0.237  0.81250
levelfl:nsloc          0.068603   0.015126   4.535 5.75e-06 ***
levelhdct:nsloc       -0.014500   0.007501  -1.933  0.05324 .
levelld:nsloc          0.001078   0.007299   0.148  0.88258
levelfl:ewloc          0.004590   0.015092   0.304  0.76102
levelhdct:ewloc        0.020550   0.007740   2.655  0.00793 **
levelld:ewloc         -0.001435   0.007502  -0.191  0.84825
yearyr03:nsloc         0.009125   0.007155   1.275  0.20224
yearyr04:nsloc         0.009184   0.006137   1.496  0.13453
yearyr03:ewloc        -0.001812   0.007242  -0.250  0.80241
yearyr04:ewloc        -0.011085   0.006324  -1.753  0.07965 .
levelhdct:siteEriley  -1.344673   0.587258  -2.290  0.02204 *
levelld:siteEriley    -0.561644   0.552345  -1.017  0.30923
levelhdct:siteLf      -1.410639   0.631143  -2.235  0.02541 *
levelld:siteLf        -0.674982   0.587591  -1.149  0.25067
levelhdct:siteNessman  0.416160   0.893648   0.466  0.64144
levelld:siteNessman    0.919968   0.777808   1.183  0.23690
levelhdct:siteNWLF     0.046781   0.553089   0.085  0.93259
levelld:siteNWLF       0.140718   0.530656   0.265  0.79087
levelhdct:siteSPP     -0.701296   0.573572  -1.223  0.22145
levelld:siteSPP       -0.477678   0.560739  -0.852  0.39429
levelhdct:siteStevens -0.213370   0.593238  -0.360  0.71909
levelld:siteStevens   -0.237795   0.568455  -0.418  0.67571
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

Original predictor variables dropped (aliased)
     levelhdct
     levelld
```

> anova(out4, out3)

Analysis of Deviance Table

```
Model 1: resp ~ varb + (level + year):(nsloc + ewloc) + level * site
Model 2: resp ~ varb + varb:nsloc + varb:ewloc + level * site
  Model Df Model Dev Df Deviance P(>|Chi|)
1       37   2668.39
2       45   2663.55  8     4.83      0.78
```

So we have goodness of fit, and this can be our "big model".

2.5 Fifth Model, Much Less Geometry

Now we reduce the geometry to just two predictors.

```
> out5 <- aster(resp ~ varb + nsloc + ewloc + level *
+     site, pred, fam, varb, id, root, data = redata)
> summary(out5)

Call:
aster.formula(formula = resp ~ varb + nsloc + ewloc + level *
    site, pred = pred, fam = fam, varvar = varb, idvar = id,
    root = root, data = redata)

                      Estimate Std. Error z value Pr(>|z|)
(Intercept)          -1.740778   0.415402  -4.191 2.78e-05 ***
varbfl03             -0.334899   0.264510  -1.266 0.205473
varbfl04             -0.328208   0.240321  -1.366 0.172032
varbhdct02            2.190026   0.536935   4.079 4.53e-05 ***
varbhdct03            2.209167   0.516225   4.279 1.87e-05 ***
varbhdct04            2.712799   0.509537   5.324 1.01e-07 ***
varbld02             -0.769927   0.562247  -1.369 0.170882
varbld03              0.982700   0.611052   1.608 0.107789
varbld04              4.110627   0.573689   7.165 7.76e-13 ***
nsloc                 0.013506   0.001725   7.828 4.97e-15 ***
ewloc                 0.005965   0.001722   3.465 0.000531 ***
siteEriley            0.755567   0.448815   1.683 0.092284 .
siteLf                0.809411   0.478696   1.691 0.090862 .
siteNessman          -0.721910   0.677652  -1.065 0.286735
siteNWLF             -0.138061   0.433020  -0.319 0.749854
siteSPP               0.516341   0.448626   1.151 0.249757
siteStevens           0.080025   0.464014   0.172 0.863074
levelhdct:siteEriley -1.259136   0.585222  -2.152 0.031433 *
levelld:siteEriley   -0.548422   0.554771  -0.989 0.322881
```

```
levelhdct:siteLf      -1.308189   0.629591  -2.078 0.037724 *
levelld:siteLf        -0.635814   0.590884  -1.076 0.281910
levelhdct:siteNessman  0.581044   0.890106   0.653 0.513898
levelld:siteNessman    1.048141   0.779258   1.345 0.178609
levelhdct:siteNWLF     0.072362   0.552482   0.131 0.895795
levelld:siteNWLF       0.183780   0.534492   0.344 0.730967
levelhdct:siteSPP     -0.662518   0.572464  -1.157 0.247147
levelld:siteSPP       -0.474797   0.564778  -0.841 0.400528
levelhdct:siteStevens -0.171288   0.593123  -0.289 0.772742
levelld:siteStevens   -0.216441   0.572985  -0.378 0.705622
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

Original predictor variables dropped (aliased)
     levelhdct
     levelld
```

```
> anova(out5, out4, out3)

Analysis of Deviance Table

Model 1: resp ~ varb + nsloc + ewloc + level * site
Model 2: resp ~ varb + (level + year):(nsloc + ewloc) + level * site
Model 3: resp ~ varb + varb:nsloc + varb:ewloc + level * site
  Model Df Model Dev Df Deviance P(>|Chi|)
1       29   2696.56
2       37   2668.39  8    28.18 0.0004416
3       45   2663.55  8     4.83      0.78
```

So we do not have goodness of fit, and this cannot be our "big model".

2.6 Sixth Model, Intermediate Geometry, Levels

So we try again with the geometry.

```
> out6 <- aster(resp ~ varb + level:(nsloc + ewloc) +
+     level * site, pred, fam, varb, id, root, data = redata)
> summary(out6)

Call:
aster.formula(formula = resp ~ varb + level:(nsloc + ewloc) +
```

```
    level * site, pred = pred, fam = fam, varvar = varb, idvar = id,
    root = root, data = redata)

                      Estimate Std. Error z value Pr(>|z|)
(Intercept)          -1.907186   0.420138  -4.539 5.64e-06 ***
varbfl03             -0.354186   0.266796  -1.328   0.1843
varbfl04             -0.319952   0.242895  -1.317   0.1878
varbhdct02            2.424075   0.542402   4.469 7.85e-06 ***
varbhdct03            2.447742   0.521950   4.690 2.74e-06 ***
varbhdct04            2.945476   0.514751   5.722 1.05e-08 ***
varbld02             -0.626114   0.562017  -1.114   0.2653
varbld03              1.128493   0.611062   1.847   0.0648 .
varbld04              4.254223   0.573494   7.418 1.19e-13 ***
siteEriley            0.812524   0.451595   1.799   0.0720 .
siteLf                0.882088   0.481103   1.833   0.0667 .
siteNessman          -0.593161   0.680863  -0.871   0.3837
siteNWLF             -0.110570   0.434822  -0.254   0.7993
siteSPP               0.543515   0.450805   1.206   0.2280
siteStevens           0.114072   0.465336   0.245   0.8063
levelfl:nsloc         0.070763   0.014568   4.857 1.19e-06 ***
levelhdct:nsloc      -0.006425   0.005465  -1.176   0.2398
levelld:nsloc         0.008036   0.005924   1.356   0.1749
levelfl:ewloc         0.007768   0.014546   0.534   0.5933
levelhdct:ewloc       0.011689   0.005561   2.102   0.0356 *
levelld:ewloc        -0.007240   0.006114  -1.184   0.2363
levelhdct:siteEriley -1.350424   0.587897  -2.297   0.0216 *
levelld:siteEriley   -0.563945   0.552117  -1.021   0.3071
levelhdct:siteLf     -1.416802   0.631793  -2.243   0.0249 *
levelld:siteLf       -0.678297   0.587394  -1.155   0.2482
levelhdct:siteNessman  0.405845   0.893667   0.454   0.6497
levelld:siteNessman    0.912760   0.777262   1.174   0.2403
levelhdct:siteNWLF     0.044673   0.554325   0.081   0.9358
levelld:siteNWLF       0.139453   0.530726   0.263   0.7927
levelhdct:siteSPP     -0.705876   0.574445  -1.229   0.2191
levelld:siteSPP       -0.481472   0.560611  -0.859   0.3904
levelhdct:siteStevens -0.218838   0.594180  -0.368   0.7126
levelld:siteStevens   -0.242044   0.568322  -0.426   0.6702
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1
```

```
Original predictor variables dropped (aliased)
     levelhdct
     levelld
```

```
> anova(out5, out6, out4, out3)

Analysis of Deviance Table

Model 1: resp ~ varb + nsloc + ewloc + level * site
Model 2: resp ~ varb + level:(nsloc + ewloc) + level * site
Model 3: resp ~ varb + (level + year):(nsloc + ewloc) + level * site
Model 4: resp ~ varb + varb:nsloc + varb:ewloc + level * site
  Model Df Model Dev Df Deviance P(>|Chi|)
1       29   2696.56
2       33   2674.70  4    21.87  0.000213
3       37   2668.39  4     6.31      0.18
4       45   2663.55  8     4.83      0.78
```

So we have goodness of fit, and this can be our "big model". But why drop year rather than level?

2.7 Seventh Model, Intermediate Geometry, Years

So we try again with the geometry.

```
> out7 <- aster(resp ~ varb + year:(nsloc + ewloc) +
+     level * site, pred, fam, varb, id, root, data = redata)
> summary(out7)

Call:
aster.formula(formula = resp ~ varb + year:(nsloc + ewloc) +
    level * site, pred = pred, fam = fam, varvar = varb, idvar = id,
    root = root, data = redata)

                     Estimate Std. Error z value Pr(>|z|)
(Intercept)          -1.737420   0.416050  -4.176 2.97e-05 ***
varbfl03             -0.316825   0.265066  -1.195 0.231981
varbfl04             -0.337380   0.241251  -1.398 0.161973
varbhdct02            2.193871   0.538051   4.077 4.55e-05 ***
varbhdct03            2.168558   0.517159   4.193 2.75e-05 ***
varbhdct04            2.713968   0.510169   5.320 1.04e-07 ***
```

```
varbld02              -0.778587   0.562724  -1.384 0.166480
varbld03               0.992898   0.611831   1.623 0.104625
varbld04               4.099985   0.574053   7.142 9.19e-13 ***
siteEriley             0.741522   0.449483   1.650 0.099000 .
siteLf                 0.799687   0.479291   1.668 0.095221 .
siteNessman           -0.727352   0.678773  -1.072 0.283914
siteNWLF              -0.128513   0.433273  -0.297 0.766765
siteSPP                0.507401   0.449178   1.130 0.258636
siteStevens            0.075879   0.464452   0.163 0.870225
yearyr02:nsloc         0.009909   0.004372   2.266 0.023441 *
yearyr03:nsloc         0.019664   0.004995   3.937 8.27e-05 ***
yearyr04:nsloc         0.012311   0.003276   3.758 0.000172 ***
yearyr02:ewloc         0.010774   0.004435   2.430 0.015116 *
yearyr03:ewloc         0.008521   0.004798   1.776 0.075742 .
yearyr04:ewloc         0.001578   0.003246   0.486 0.626836
levelhdct:siteEriley  -1.239189   0.585903  -2.115 0.034429 *
levelld:siteEriley    -0.532712   0.555127  -0.960 0.337246
levelhdct:siteLf      -1.294188   0.630147  -2.054 0.039996 *
levelld:siteLf        -0.624375   0.591157  -1.056 0.290881
levelhdct:siteNessman  0.587678   0.891330   0.659 0.509686
levelld:siteNessman    1.055395   0.779991   1.353 0.176028
levelhdct:siteNWLF     0.059302   0.552518   0.107 0.914526
levelld:siteNWLF       0.174231   0.534494   0.326 0.744444
levelhdct:siteSPP     -0.650145   0.572908  -1.135 0.256452
levelld:siteSPP       -0.463775   0.564970  -0.821 0.411712
levelhdct:siteStevens -0.165745   0.593407  -0.279 0.780007
levelld:siteStevens   -0.211556   0.573114  -0.369 0.712028
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

Original predictor variables dropped (aliased)
     levelhdct
     levelld

> anova(out5, out7, out4, out3)

Analysis of Deviance Table

Model 1: resp ~ varb + nsloc + ewloc + level * site
Model 2: resp ~ varb + year:(nsloc + ewloc) + level * site
```

```
Model 3: resp ~ varb + (level + year):(nsloc + ewloc) + level * site
Model 4: resp ~ varb + varb:nsloc + varb:ewloc + level * site
  Model Df Model Dev Df Deviance P(>|Chi|)
1       29   2696.56
2       33   2691.95  4     4.61      0.33
3       37   2668.39  4    23.57 9.761e-05
4       45   2663.55  8     4.83      0.78
```

And we do not have goodness of fit! So Model Six is our "big model" and we have been logical in our model selection. Note that it is not valid to compare Models Six and Seven because they are not nested, but both fit between Models Five and Four, so Six and Seven can each be compared to both Five and Four (and this tells us what we want to know).

2.8 Eighth Model, Like Models Two and Six

We need to make a model with the structure of Model Two with respect to variables and like Model Six with respect to geometry.

```
> out8 <- aster(resp ~ varb + level:(nsloc + ewloc) +
+     hdct * site - site, pred, fam, varb, id, root,
+     data = redata)
> summary(out8)

Call:
aster.formula(formula = resp ~ varb + level:(nsloc + ewloc) +
    hdct * site - site, pred = pred, fam = fam, varvar = varb,
    idvar = id, root = root, data = redata)

                 Estimate Std. Error z value Pr(>|z|)
(Intercept)      -1.591968   0.184332  -8.636  < 2e-16 ***
varbfl03         -0.349096   0.267919  -1.303  0.19258
varbfl04         -0.344222   0.243899  -1.411  0.15815
varbhdct02        1.991976   0.265192   7.511 5.85e-14 ***
varbhdct03        2.013936   0.219375   9.180  < 2e-16 ***
varbhdct04        2.521890   0.205048  12.299  < 2e-16 ***
varbld02         -0.874272   0.315703  -2.769  0.00562 **
varbld03          0.895081   0.396189   2.259  0.02387 *
varbld04          4.036755   0.334266  12.076  < 2e-16 ***
levelfl:nsloc     0.070102   0.014652   4.785 1.71e-06 ***
levelhdct:nsloc  -0.005804   0.005550  -1.046  0.29564
```

```
levelld:nsloc     0.007165   0.005867   1.221  0.22196
levelfl:ewloc     0.017977   0.014413   1.247  0.21229
levelhdct:ewloc   0.007606   0.005561   1.368  0.17138
levelld:ewloc    -0.004787   0.005919  -0.809  0.41863
hdct:siteEriley  -0.178799   0.089411  -2.000  0.04553 *
hdct:siteLf      -0.162516   0.096116  -1.691  0.09087 .
hdct:siteNessman -0.315507   0.138823  -2.273  0.02304 *
hdct:siteNWLF    -0.108209   0.083110  -1.302  0.19292
hdct:siteSPP      0.019942   0.086198   0.231  0.81704
hdct:siteStevens -0.129238   0.089129  -1.450  0.14706
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

Original predictor variables dropped (aliased)
     hdct
```

```
> anova(out8, out6)

Analysis of Deviance Table

Model 1: resp ~ varb + level:(nsloc + ewloc) + hdct * site - site
Model 2: resp ~ varb + level:(nsloc + ewloc) + level * site
  Model Df Model Dev Df Deviance P(>|Chi|)
1       21   2712.54
2       33   2674.70 12    37.84 0.0001632
```

2.9 Ninth Model, Like Models One and Eight

We need to make a model with the structure of Model Eight except no sites.

```
> out9 <- aster(resp ~ varb + level:(nsloc + ewloc),
+     pred, fam, varb, id, root, data = redata)
> summary(out9)

Call:
aster.formula(formula = resp ~ varb + level:(nsloc + ewloc),
    pred = pred, fam = fam, varvar = varb, idvar = id, root = root,
    data = redata)

                Estimate Std. Error z value Pr(>|z|)
```

```
(Intercept)     -1.607690   0.184593  -8.709  < 2e-16 ***
varbfl03        -0.349426   0.268532  -1.301   0.1932
varbfl04        -0.353847   0.244529  -1.447   0.1479
varbhdct02       1.906022   0.257183   7.411 1.25e-13 ***
varbhdct03       1.929506   0.209331   9.217  < 2e-16 ***
varbhdct04       2.436417   0.194176  12.547  < 2e-16 ***
varbld02        -0.866520   0.315991  -2.742   0.0061 **
varbld03         0.905229   0.396379   2.284   0.0224 *
varbld04         4.052121   0.334412  12.117  < 2e-16 ***
levelfl:nsloc    0.070281   0.014705   4.779 1.76e-06 ***
levelhdct:nsloc -0.005378   0.005578  -0.964   0.3350
levelld:nsloc    0.006855   0.005868   1.168   0.2427
levelfl:ewloc    0.017686   0.014441   1.225   0.2207
levelhdct:ewloc  0.007312   0.005542   1.320   0.1870
levelld:ewloc   -0.005027   0.005932  -0.847   0.3968
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

> anova(out9, out8, out6)

Analysis of Deviance Table

Model 1: resp ~ varb + level:(nsloc + ewloc)
Model 2: resp ~ varb + level:(nsloc + ewloc) + hdct * site - site
Model 3: resp ~ varb + level:(nsloc + ewloc) + level * site
  Model Df Model Dev Df Deviance P(>|Chi|)
1       15   2728.72
2       21   2712.54  6    16.18      0.01
3       33   2674.70 12    37.84 0.0001632
```

2.10 Tenth Model, Between Models Six and Eight

We accidentally created a new tenth model by not understanding the "minus site" stuff in the formulae for Models Two and Eight.

```
> out10 <- aster(resp ~ varb + level:(nsloc + ewloc) +
+     hdct * site, pred, fam, varb, id, root, data = redata)
> summary(out10)

Call:
aster.formula(formula = resp ~ varb + level:(nsloc + ewloc) +
```

```
    hdct * site, pred = pred, fam = fam, varvar = varb, idvar = id,
    root = root, data = redata)

                 Estimate Std. Error z value Pr(>|z|)
(Intercept)      -1.726122   0.223462  -7.724 1.12e-14 ***
varbfl03         -0.349106   0.266693  -1.309  0.19053
varbfl04         -0.330903   0.242658  -1.364  0.17268
varbhdct02        2.203913   0.331242   6.653 2.86e-11 ***
varbhdct03        2.226234   0.296256   7.515 5.71e-14 ***
varbhdct04        2.732111   0.285814   9.559  < 2e-16 ***
varbld02         -0.861178   0.316060  -2.725  0.00644 **
varbld03          0.888715   0.396388   2.242  0.02496 *
varbld04          4.008350   0.334436  11.985  < 2e-16 ***
siteEriley        0.375414   0.154884   2.424  0.01536 *
siteLf            0.355237   0.164870   2.155  0.03119 *
siteNessman       0.231430   0.189693   1.220  0.22245
siteNWLF          0.012016   0.145094   0.083  0.93400
siteSPP           0.185498   0.158948   1.167  0.24319
siteStevens      -0.069680   0.153063  -0.455  0.64894
levelfl:nsloc     0.070918   0.014584   4.863 1.16e-06 ***
levelhdct:nsloc  -0.006532   0.005504  -1.187  0.23528
levelld:nsloc     0.007901   0.005959   1.326  0.18488
levelfl:ewloc     0.014308   0.014359   0.996  0.31904
levelhdct:ewloc   0.010083   0.005557   1.814  0.06961 .
levelld:ewloc    -0.009182   0.006100  -1.505  0.13231
hdct:siteEriley  -0.794844   0.264102  -3.010  0.00262 **
hdct:siteLf      -0.744006   0.282716  -2.632  0.00850 **
hdct:siteNessman -0.705433   0.355627  -1.984  0.04730 *
hdct:siteNWLF    -0.114923   0.245244  -0.469  0.63935
hdct:siteSPP     -0.270775   0.262750  -1.031  0.30276
hdct:siteStevens  0.003604   0.260305   0.014  0.98895
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ 1

Original predictor variables dropped (aliased)
     hdct

> anova(out9, out8, out10, out6)

Analysis of Deviance Table
```

```
Model 1: resp ~ varb + level:(nsloc + ewloc)
Model 2: resp ~ varb + level:(nsloc + ewloc) + hdct * site - site
Model 3: resp ~ varb + level:(nsloc + ewloc) + hdct * site
Model 4: resp ~ varb + level:(nsloc + ewloc) + level * site
  Model Df Model Dev Df Deviance P(>|Chi|)
1       15   2728.72
2       21   2712.54  6    16.18      0.01
3       27   2684.86  6    27.67 0.0001083
4       33   2674.70  6    10.17      0.12
```

So this says that Model Ten (which only has 12 d. f. for "site") can be the big model.

```
> save(redata, out6, out8, out9, out10, pred, fam,
+     file = "coneflr.RData")
```

3 Mean Value Parameters

3.1 Unconditional

As argued in *Aster Models*, canonical parameters are "meaningless." Only mean value parameters have real world, scientific interpretability.

So in this section we compare predicted values for a typical individual (say zero-zero geometry) in each site under both Models Six and Eight. The functional of mean value parameters we want is *total head count*, since this has the biological interpretation of the component of fitness measured in this data set. A biologist (at least an evolutionary biologist) is interested in the "predecessor variables" of head count only insofar as they contribute to head count. Two sets of parameter values that "predict" the same expected total head count (over the three years the data were collected) have the same contribution to fitness. So that is the "prediction" (really functional of mean value parameters) we "predict."

To do this we must construct "newdata" for these hypothetical individuals.

```
> newdata <- data.frame(site = levels(echinacea$site))
> for (v in vars) newdata[[v]] <- 1
> newdata$root <- 1
> newdata$ewloc <- 0
> newdata$nsloc <- 0
> renewdata <- reshape(newdata, varying = list(vars),
```

```
+     direction = "long", timevar = "varb", times = as.factor(vars),
+     v.names = "resp")
> names(redata)

 [1] "site"  "ewloc" "nsloc" "varb"  "resp"  "id"    "root"
 [8] "hdct"  "level" "year"

> names(renewdata)

[1] "site"  "root"  "ewloc" "nsloc" "varb"  "resp"  "id"

> hdct <- grep("hdct", as.character(renewdata$varb))
> hdct <- is.element(seq(along = renewdata$varb), hdct)
> renewdata$hdct <- as.integer(hdct)
> level <- gsub("[0-9]", "", as.character(renewdata$varb))
> renewdata$level <- as.factor(level)
> year <- gsub("[a-z]", "", as.character(renewdata$varb))
> year <- paste("yr", year, sep = "")
> renewdata$year <- as.factor(year)
> setequal(names(redata), names(renewdata))

[1] TRUE
```

We are using bogus data $x_{ij} = 1$ for all i and j because unconditional mean value parameters do not depend on x. We have to have an `x` argument because that's the way the aster package functions work (ultimately due to limitations of the R formula mini-language). So it doesn't matter what we make it. In the following section, the predictions will depend on `x`, but then (as we shall argue), this is the `x` we want.

```
> nind <- nrow(newdata)
> nnode <- length(vars)
> amat <- array(0, c(nind, nnode, nind))
> for (i in 1:nind) amat[i, grep("hdct", vars), i] <- 1
> pout6 <- predict(out6, varvar = varb, idvar = id,
+     root = root, newdata = renewdata, se.fit = TRUE,
+     amat = amat)
> pout8 <- predict(out8, varvar = varb, idvar = id,
+     root = root, newdata = renewdata, se.fit = TRUE,
+     amat = amat)
```

Figure 1 is produced by the following code

```
> conf.level <- 0.95
> crit <- qnorm((1 + conf.level)/2)

> sitenames <- as.character(newdata$site)
> fit8 <- pout8$fit
> i <- seq(along = sitenames)
> ytop <- fit8 + crit * pout8$se.fit
> ybot <- fit8 - crit * pout8$se.fit
> plot(c(i, i), c(ytop, ybot), type = "n", axes = FALSE,
+     xlab = "", ylab = "")
> segments(i, ybot, i, ytop)
> foo <- 0.1
> segments(i - foo, ybot, i + foo, ybot)
> segments(i - foo, ytop, i + foo, ytop)
> segments(i - foo, fit8, i + foo, fit8)
> axis(side = 2)
> title(ylab = "unconditional mean value parameter")
> axis(side = 1, at = i, labels = sitenames)
> title(xlab = "site")
```

and appears on p. 22.

Figure 2 is produced by the following code

```
> fit6 <- pout6$fit
> i <- seq(along = sitenames)
> foo <- 0.1
> y8top <- fit8 + crit * pout8$se.fit
> y8bot <- fit8 - crit * pout8$se.fit
> y6top <- fit6 + crit * pout6$se.fit
> y6bot <- fit6 - crit * pout6$se.fit
> plot(c(i - 1.5 * foo, i - 1.5 * foo, i + 1.5 * foo,
+     i + 1.5 * foo), c(y8top, y8bot, y6top, y6bot),
+     type = "n", axes = FALSE, xlab = "", ylab = "")
> segments(i - 1.5 * foo, y8bot, i - 1.5 * foo, y8top)
> segments(i - 2.5 * foo, y8bot, i - 0.5 * foo, y8bot)
> segments(i - 2.5 * foo, y8top, i - 0.5 * foo, y8top)
> segments(i - 2.5 * foo, fit8, i - 0.5 * foo, fit8)
> segments(i + 1.5 * foo, y6bot, i + 1.5 * foo, y6top,
+     lty = 2)
> segments(i + 2.5 * foo, y6bot, i + 0.5 * foo, y6bot)
> segments(i + 2.5 * foo, y6top, i + 0.5 * foo, y6top)
```

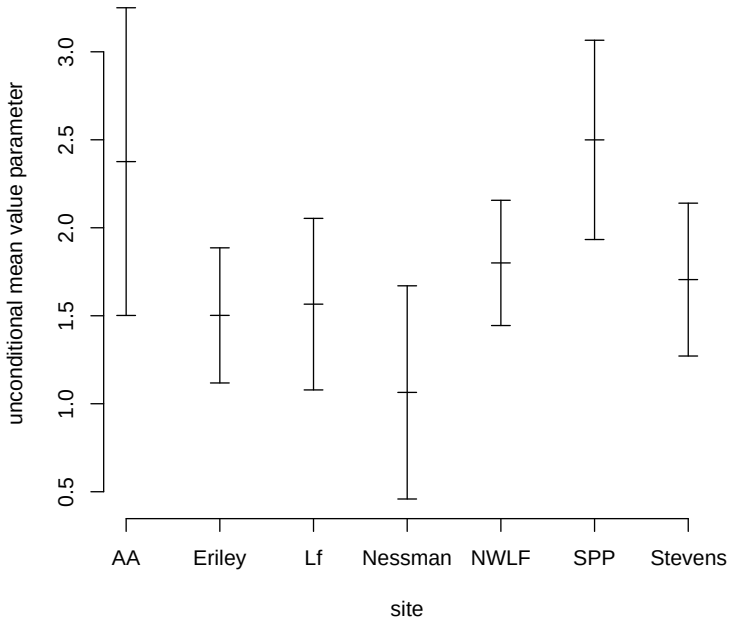

Figure 1: 95% confidence intervals for unconditional mean value parameter for fitness (sum of head count for all years) at each site for a "typical" individual having position zero-zero and having the parameterization of Model Eight. Tick marks in the middle of the bars are the center (the MLE).

```
> segments(i + 2.5 * foo, fit6, i + 0.5 * foo, fit6)
> axis(side = 2)
> title(ylab = "unconditional mean value parameter")
> axis(side = 1, at = i, labels = sitenames)
> title(xlab = "site")
```

and appears on p. 24.

3.2 Conditional

This section is very incomplete. We don't redo everything using conditional models. That's not the point. We only want to show that conditional models and conditional mean value parameters just don't do the same thing as unconditional models (which is obvious, but some people like examples, and in any case, this gives us an opportunity to show some options of aster model fitting).

3.2.1 Conditional Models

Let us redo Figure 2 based on conditional models with the same model matrices (a dumb idea, since the meaning of the models is entirely different despite the similarity in algebra, but we want to hammer the point home).

```
> cout6 <- aster(resp ~ varb + level:(nsloc + ewloc) +
+     level * site, pred, fam, varb, id, root, data = redata,
+     type = "conditional")
> cout8 <- aster(resp ~ varb + level:(nsloc + ewloc) +
+     hdct * site - site, pred, fam, varb, id, root,
+     data = redata, type = "conditional")
> pcout6 <- predict(cout6, varvar = varb, idvar = id,
+     root = root, newdata = renewdata, se.fit = TRUE,
+     amat = amat)
> pcout8 <- predict(cout8, varvar = varb, idvar = id,
+     root = root, newdata = renewdata, se.fit = TRUE,
+     amat = amat)
```

Note that these are exactly like the analogous statements making the analogous objects without the "c" in their names except for the `type = "conditional"` arguments in the two `aster` function calls. Then we make Figure 3 just like Figure 2 except for using `pcout6` and `pcout8` instead of `pout6` and `pout8`. It appears on p. 25.

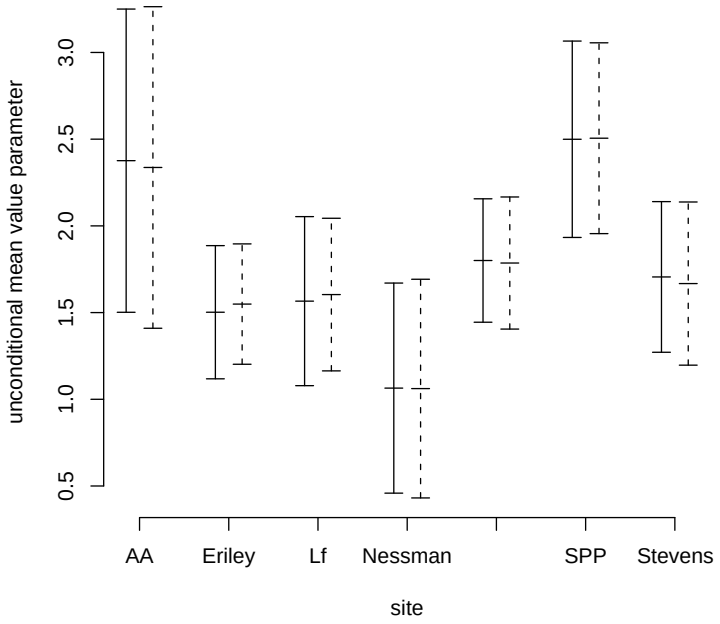

Figure 2: 95% confidence intervals for unconditional mean value parameter for fitness (sum of head count for all years) at each site for a "typical" individual having position zero-zero and having the parameterization of Model Eight (solid bar) or Model Six (dashed bar). Tick marks in the middle of the bars are the center (the MLE).

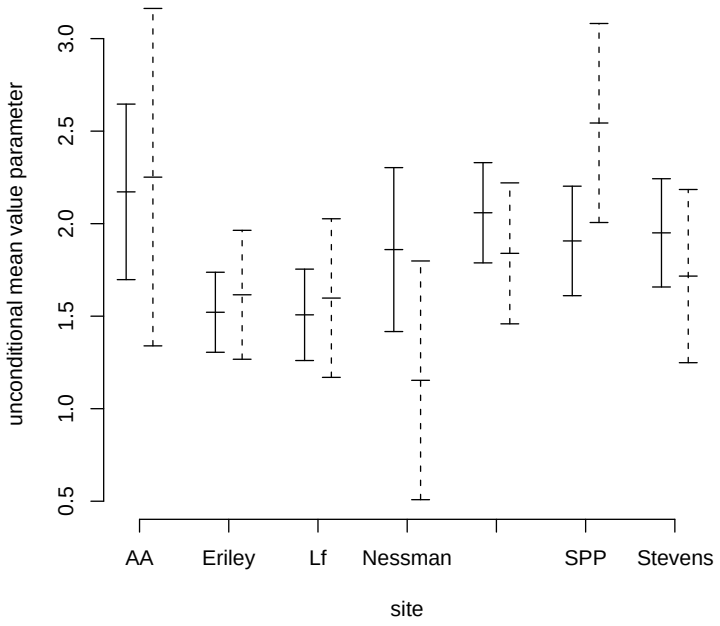

Figure 3: 95% confidence intervals for unconditional mean value parameter for fitness (sum of head count for all years) at each site for a "typical" individual having position zero-zero and having the parameterization of Model Eight (solid bar) or Model Six (dashed bar). Tick marks in the middle of the bars are the center (the MLE). The difference between this figure and Figure 2 is that the models fitted are *conditional* rather than *unconditional*.

Note the huge difference between Figure 2 and Figure 3. The same model matrices are used in both cases. The linear predictor satisfies $\boldsymbol{\eta} = \mathbf{M}\boldsymbol{\beta}$, but in one case (Figure 2) the linear predictor is the unconditional canonical parameter ($\boldsymbol{\eta} = \boldsymbol{\varphi}$) and in the other case (Figure 3) the linear predictor is the conditional canonical parameter ($\boldsymbol{\eta} = \boldsymbol{\theta}$). In one case (Figure 2) the predictions of a linear functional of the unconditional mean value parameter ($\boldsymbol{\tau}$) are nearly the same for the two models and in the other case (Figure 3) the predictions of the same linear functional of $\boldsymbol{\tau}$ are wildly different.

Conclusion: conditional models and unconditional models are different. That's the whole point. That's why unconditional models were invented, because conditional models can't be made to do the same thing.

3.2.2 Conditional Parameter

Let us redo Figure 2 now not changing the model (we still use the fits `out6` and `out8`) but changing the thingummy we "predict". In Figure 2 we "predict" a linear functional $\mathbf{A}'\boldsymbol{\tau}$ of the unconditional mean value parameter (the sum of three components of $\boldsymbol{\tau}$, those for flower head count). In Figure 4 we predict the same linear functional $\mathbf{A}'\boldsymbol{\xi}$ of the *conditional* mean value parameter $\boldsymbol{\xi}$.

```
> pxout6 <- predict(out6, varvar = varb, idvar = id,
+     root = root, newdata = renewdata, se.fit = TRUE,
+     amat = amat, model.type = "conditional")
> pxout8 <- predict(out8, varvar = varb, idvar = id,
+     root = root, newdata = renewdata, se.fit = TRUE,
+     amat = amat, model.type = "conditional")
```

Note that these are exactly like the analogous statements making the analogous objects without the "x" in their names except for the `model.type = "conditional"` arguments in the two `predict` function calls. Then we make Figure 4 just like Figure 2 except for using `pxout6` and `pxout8` instead of `pout6` and `pout8`. It appears on p. 27.

Note the huge difference between Figure 2 and Figure 4. The same models are used in both cases but in one case (Figure 2) we "predict" a linear functional $\mathbf{A}'\boldsymbol{\tau}$ of the unconditional mean value parameter ($\boldsymbol{\tau}$) and in the other case (Figure 4) we "predict" the *same* linear functional $\mathbf{A}'\boldsymbol{\xi}$ of the conditional mean value parameter ($\boldsymbol{\xi}$).

Conclusion: conditional expectations and unconditional expectations are different. (Duh!) The two sorts of predictions can't be made to do the same thing.

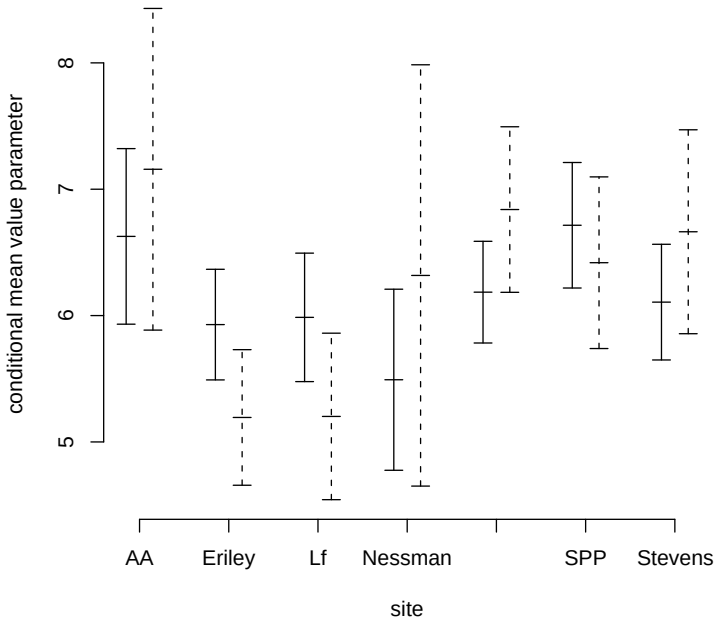

Figure 4: 95% confidence intervals for conditional mean value parameter for fitness (sum of head count for all years) at each site for a "typical" individual having position zero-zero and having the parameterization of Model Eight (solid bar) or Model Six (dashed bar). Tick marks in the middle of the bars are the center (the MLE). The difference between this figure and Figure 2 is that the parameters "predicted" are *conditional* rather than *unconditional*.

A Plot for the Paper

We redo Figure 2 changing the models compared to Model 8 and Model 10 (fits in `out8` and `out10`).

```
> pout10 <- predict(out10, varvar = varb, idvar = id,
+     root = root, newdata = renewdata, se.fit = TRUE,
+     amat = amat)
```

It appears on p. 29.

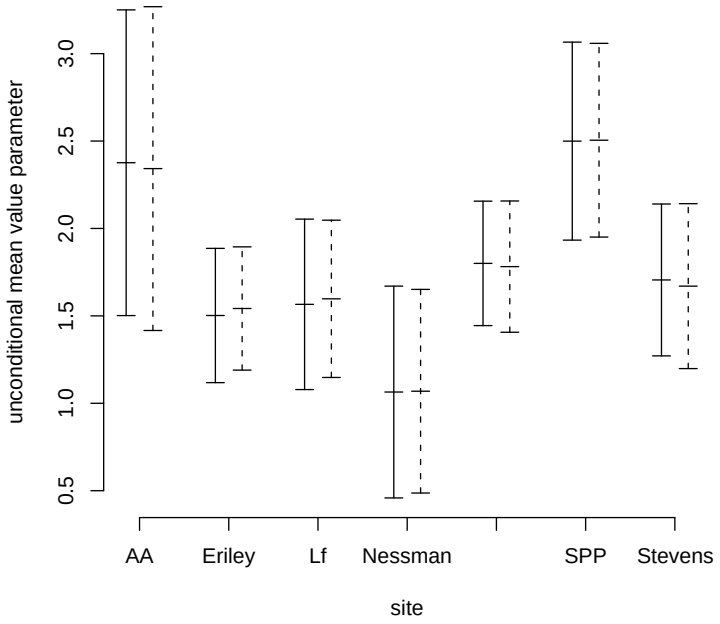

Figure 5: 95% confidence intervals for unconditional mean value parameter for fitness (sum of head count for all years) at each site for a "typical" individual having position zero-zero and having the parameterization of Model Eight (solid bar) or Model Ten (dashed bar). Tick marks in the middle of the bars are the center (the MLE).