Investigation on sub-sampling

Gan Yao

2022-11-03

Genral idea

This analysis attempts to handle the sub-sampling node in aster graph, by considering expected fitness as products of conditional mean value parameters.

Recall the proposed aster graph for Lilium data,

$$root \to flCt \to flNotConsumed \to capsuleCt \to isHarvested \to ovuleCt \to embryoCt(fitness)$$

Denote conditional mean value parameters of six non-root nodes sequentially as $\xi_1, \xi_2, \xi_3, \xi_4, \xi_5, \xi_6$. Then the expected fitness should be,

$$\xi_1\xi_2\xi_3\xi_5\xi_6$$

Also, to apply this procedure on random effects model. A little trick is used to transform variance components to the scale of conditional mean value parameters to enable later delta methods.

Above ideas are borrowed from Prof. Dan Eck's summer project write up and Prof. Geyer's TR696.

Analysis

Load libraries and read data.

```r
library(aster)
library(tidyverse)
redata <- read.csv("Lilium_processed.csv")
redata$site <- as.factor(redata$site)
redata$varb <- as.factor(redata$varb)
```

First fit the full random effects model.

```r
pred <- c(0,1,2,3,4,5)
fam <- c(2,1,2,1,2,1)
redata$Nid <- as.numeric(gsub("[^0-9.-]", "", redata$id))
remodel <- reaster(resp ~ varb + fit:(nn1Dist + nn2Dist + nn3Dist + nn4Dist + nn5Dist
                                       + nn6Dist + nn7Dist + nn8Dist + nn9Dist + nn10Dist),
                    list(site = ~0 + fit:site),
                    pred, fam,varb,Nid,root,data=redata)
```

And a fixed effects model that will be 'tricked' by us to do the transformation.

```r
femodel <- aster(resp ~ varb + fit:(site+nn1Dist + nn2Dist + nn3Dist + nn4Dist + nn5Dist
                                       + nn6Dist + nn7Dist + nn8Dist + nn9Dist + nn10Dist),
                    pred, fam,varb,Nid,root,data=redata)
```

Get the parameters of random effects model.

```
alpha.hat <- remodel$alpha
b.hat <- remodel$b
fred <- c(alpha.hat, b.hat)
idx <- match(names(femodel$coefficients), names(fred))
idx
```

```
##  [1]  1  2  3  4  5  6 17 18 19  7  8  9 10 11 12 13 14 15 16
```

```
fred[-idx]
```

```
## fit:sitesppw
##  0.002299888
```

Now we do the transformation by calling `predict` on fixed effects model, but with the parameters of random effects model.

```
pred <- predict(femodel, varvar=varb, idvar=id, root=root, se.fit=TRUE, newcoef=fred[idx],
                model.type = 'conditional')
```

```
## apparent null eigenvectors of information matrix
## directions of recession or constancy of log likelihood
##              [,1]       [,2]       [,3]
##  [1,]  0.0721329  0.4354807 -0.0140221
##  [2,] -0.0733104 -0.4358130  0.0140129
##  [3,] -0.6369888 -0.4341350  0.0140214
##  [4,]  0.7604961 -0.4889580  0.0147854
##  [5,]  0.0005299  0.0303749  0.9995350
##  [6,] -0.0728875 -0.4381186  0.0110262
##  [7,] -0.0001262  0.0000008 -0.0000119
##  [8,] -0.0004514 -0.0000385 -0.0000598
##  [9,]  0.0005397  0.0000090  0.0000220
## [10,]  0.0000052  0.0000000  0.0000003
## [11,] -0.0000073 -0.0000001 -0.0000003
## [12,] -0.0000075 -0.0000002 -0.0000003
## [13,]  0.0000035  0.0000002  0.0000002
## [14,] -0.0000056  0.0000002 -0.0000004
## [15,]  0.0000199  0.0000000  0.0000011
## [16,]  0.0000096  0.0000001  0.0000010
## [17,] -0.0000252 -0.0000008 -0.0000019
## [18,]  0.0000079  0.0000007  0.0000004
## [19,]  0.0000021  0.0000000  0.0000002
```

```
## Error in predict.aster(femodel, varvar = varb, idvar = id, root = root, : cannot compute standard er
```

Direction of recession problem pops out again. Set `info.tol`. Note that we jumped directly from canonical parameter scale to conditional mean value parameter scale. To make sure `predict` function output what we want, set `is.always.parameter = TRUE`.

```
pred <- predict(femodel, varvar=varb, idvar=id, root=root, se.fit=TRUE, newcoef=fred[idx],
                model.type = 'conditional', info.tol  = 1e-11, is.always.parameter = TRUE)

xis <- pred$fit
names(xis) <- redata$X
```

Get variation of conditional mean value parameters by delta method.

```
foo <- pred$gradient
rownames(foo) <- redata$X
colnames(foo) <- names(femodel$coefficients)
#thegradient <- foo[, "fit:sitebeng"]
thegradient <- foo[, c("fit:sitebeng", "fit:sitehegg", "fit:sitesppe")]
thevariance <- thegradient %*% t(thegradient) * remodel$nu
rownames(thevariance) <- redata$X
```

Expected fitness is products of conditional mean value parameters.

```
Ids <- unique(redata$id)
exp_fitness <- rep(0,697)
names(exp_fitness) <- Ids
for (id in Ids) {
  exp_fitness[id] <- prod(xis[grep(id, names(xis))][-4])
}
```

Call delta method again to get variation of expected fitness.

```
var_expfit <- rep(0,697)
names(var_expfit) <- Ids
for (id in Ids) {
  ind <- paste(id,'\\.', sep='')
  xi <- xis[grep(ind, names(xis))][-4]
  var <- thevariance[grep(ind, rownames(thevariance))[-4],
                     grep(ind, colnames(thevariance))[-4]]
  grad <- c(prod(xi[-1]), prod(xi[-2]), prod(xi[-3]), prod(xi[-4]), prod(xi[-5]))
  var_expfit[id] <- t(grad) %*% var %*% grad
}
```

Let's take at look at the expected fitness and variation of some individuals.

```
sd_expfit = sqrt(var_expfit)
head(cbind(exp_fitness,sd_expfit),20)
```

```
##         exp_fitness sd_expfit
## LP1000 1.358946e+01  9.039905
## LP1001 1.377828e+01  9.273702
## LP1002 1.386370e+01  9.379747
## LP1003 1.389156e+01  9.414372
## LP1004 1.360767e+01  9.062418
## LP1005 1.386802e+01  9.385117
## LP1006 1.373612e+01  9.221431
## LP1007 1.376949e+01  9.262795
## LP1008 1.385190e+01  9.365085
## LP1009 1.391322e+01  9.441304
## LP101  9.162291e+06  3.451148
## LP1010 1.456461e+01 10.256335
## LP1011 1.400945e+01  9.561089
## LP1012 1.526793e+01 11.147276
## LP1013 1.475858e+01 10.500928
## LP1014 1.545691e+01 11.388558
## LP1015 1.637282e+01 12.569138
## LP1016 1.497113e+01 10.769937
## LP102  8.937471e+00  3.569710
## LP103  9.030573e+00  3.673154
```

```r
sum(sd_expfit == 0)
```

```
## [1] 301
```