Lifetime fitness through female and male function: the influence of density and genetically effective population size

Mason W. Kulbaba & Ruth G. Shaw

April 5, 2019

Introduction

The following code performs fixed-effects aster analyses on data examining the effects of density and effective genetic population size (Ne) on female (seeds seet) fitness. LM analysis of biomass (above and below ground) and number of aborted ovules follows
the aster analyses.

Please send any questions to Mason Kulbaba (mason.kulbaba@gmail.com)

Preliminaries

Set working directory and load data.

Examine data

```
fin<- read.csv("data/aster.dat.csv")

names(fin)
```

```
##  [1] "Treat"     "plotID"    "Den"       "Gen"       "plantID"
##  [6] "familyID"  "surv"      "flw"       "frt"       "frt.2"
## [11] "seeds"     "aborted"   "rel.seeds" "seed.wt"   "mass.a"
## [16] "mass.b"
```

```
head(fin)
```

```
##   Treat plotID Den Gen plantID familyID surv flw frt frt.2 seeds aborted
## 1  HDHG      5   H  HG       1        4    1   1   3     2    33      14
## 2  HDHG      5   H  HG       2        6    1   1  21     6    69      22
## 3  HDHG      5   H  HG       3       11    1   1  16     7    73      19
## 4  HDHG      5   H  HG       4       14    1   1  18     8   107      23
## 5  HDHG      5   H  HG       5        6    1   1   1     1    16       4
## 6  HDHG      5   H  HG       6       14    1   1   0     0     0       0
##   rel.seeds seed.wt mass.a mass.b
## 1 0.5714286   6.792    5.8    0.8
## 2 1.1948052   6.995    1.9    0.2
## 3 1.2640693   7.413   20.8    1.3
## 4 1.8528139   8.554   18.2    1.1
## 5 0.2770563   7.771   65.8    6.1
## 6 0.0000000   0.000    6.6    1.2
```

Make sure Den (density treatment), Gen (Ne treatment), plotID, plantID, and familyID are all classified as factors. Otherwise

```
fin$Den<- as.factor(fin$Den)
fin$Gen<- as.factor(fin$Gen)
fin$plotID<- as.factor(fin$plotID)
```

```
fin$plantID<- as.factor(fin$plantID)
fin$familyID<- as.factor(fin$familyID)
```

Setting up for aster analyses

Load aster package (make sure you have most current version)

```
library(aster)
```

```
## Loading required package: trust
```

Begin by naming variables that will be used in the graphical model of the aster analyses:

flw - total number of flowers produced frt - total number of fruits produced frt.2 - subsetted number of fruits collected seeds - total number of seeds collected from subsetted fruits

```
vars<- c( "flw", "frt", "frt.2","seeds")
```

Reshape the data so that all response variables are located in a single vector, in a new data set called "redata"

```
redata <- reshape(fin, varying = list(vars), direction = "long",timevar = "varb", times = as.factor(vars
```

Designate the terminal fitness variable "seeds" (make it numberic), and then add it to the reshaped data

```
fit <- grepl("seeds", as.character(redata$varb))
fit<- as.numeric(fit)
redata$fit <- fit
```

Check

```
with(redata, sort(unique(as.character(varb)[fit == 0])))
```

```
## [1] "flw"   "frt"   "frt.2"
```

```
with(redata, sort(unique(as.character(varb)[fit == 1])))
```

```
## [1] "seeds"
```

Add a variable "root" to redata, where value is 1. This is the "starting point" of the aster graphical model (i.e. a seed planted)

```
redata<- data.frame(redata, root=1)
```

Set up the graphical model and designate the statistical distribution for each node. This graphical model has four nodes (in object `pred`) described earlier. Statistical family for each node is described by object `fam`.

```
pred<- c(0,1,2,3)
fam<- c(1,2,2,2)
```

Show distribution family for each node

```
sapply(fam.default(), as.character)[fam]
```

```
## [1] "bernoulli" "poisson"   "poisson"   "poisson"
```

Aster Analyses

First aster analysis with only fitness data. Note, `aster` reads the `redata` version of the data.

```
aout<- aster(resp~varb , pred, fam, varb, id, root, data=redata)

summary(aout, show.graph=T)
```

```
##
## Call:
## aster.formula(formula = resp ~ varb, pred = pred, fam = fam,
##     varvar = varb, idvar = id, root = root, data = redata)
##
##
## Graphical Model:
##  variable predecessor family
##  flw      root        bernoulli
##  frt      flw         poisson
##  frt.2    frt         poisson
##  seeds    frt.2       poisson
##
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -44.6476     0.5507  -81.08   <2e-16 ***
## varbfrt      49.1734     0.5599   87.82   <2e-16 ***
## varbfrt.2    34.2048     0.5538   61.76   <2e-16 ***
## varbseeds    46.9877     0.5507   85.32   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Add density to the model as a fixed effect. Note that we follow the "no naked predictors" rule with `fit:(Den)`

```
aout.d<- aster(resp~varb + fit:(Den), pred, fam, varb, id, root, data=redata)

summary(aout.d, show.graph=T)
```

```
##
## Call:
## aster.formula(formula = resp ~ varb + fit:(Den), pred = pred,
##     fam = fam, varvar = varb, idvar = id, root = root, data = redata)
##
##
## Graphical Model:
##  variable predecessor family
##  flw      root        bernoulli
##  frt      flw         poisson
##  frt.2    frt         poisson
##  seeds    frt.2       poisson
##
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -39.755770   0.597861 -66.497   <2e-16 ***
## varbfrt      44.287436   0.606131  73.066   <2e-16 ***
## varbfrt.2    29.362530   0.598590  49.053   <2e-16 ***
## varbseeds    42.077712   0.597597  70.412   <2e-16 ***
## fit:DenH      0.041854   0.003121  13.411   <2e-16 ***
## fit:DenL     -0.039658   0.004752  -8.346   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Original predictor variables dropped (aliased)
##      fit:DenM
```

Do not try to determine the significance `Density` from the `summary` statement. These are not reliable. Instead, perform a liklihood ratio test. The results of the liklihood ratio test show that the model with `Density` explains significantly more variation than the model without.

```
anova(aout, aout.d)
```

```
## Analysis of Deviance Table
##
## Model 1: resp ~ varb
## Model 2: resp ~ varb + fit:(Den)
##   Model Df Model Dev Df Deviance P(>|Chi|)
## 1        4    132606
## 2        6    133112  2   505.61 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Now, add density and Ne (`Gen`) to the model, and perform a liklihood ratio test

```
aout.dg<- aster(resp~varb + fit:(Den + Gen), pred, fam, varb, id, root, data=redata)

summary(aout.dg, show.graph=T)
```

```
##
## Call:
## aster.formula(formula = resp ~ varb + fit:(Den + Gen), pred = pred,
##     fam = fam, varvar = varb, idvar = id, root = root, data = redata)
##
##
## Graphical Model:
##  variable predecessor family
##  flw      root        bernoulli
##  frt      flw         poisson
##  frt.2    frt         poisson
##  seeds    frt.2       poisson
##
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -38.436651   0.570184  -67.41   <2e-16 ***
## varbfrt      42.971860   0.578764   74.25   <2e-16 ***
## varbfrt.2    28.072418   0.570263   49.23   <2e-16 ***
## varbseeds    40.775220   0.569845   71.56   <2e-16 ***
## fit:DenH      0.040797   0.003083   13.23   <2e-16 ***
## fit:DenL     -0.046588   0.004664   -9.99   <2e-16 ***
## fit:GenLG    -0.048047   0.003018  -15.92   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Original predictor variables dropped (aliased)
##      fit:DenM
```

```
anova(aout.d, aout.dg)
```

```
## Analysis of Deviance Table
##
## Model 1: resp ~ varb + fit:(Den)
## Model 2: resp ~ varb + fit:(Den + Gen)
##   Model Df Model Dev Df Deviance P(>|Chi|)
## 1        6    133112
## 2        7    133406  1   294.19 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Include interaction between `Den` & `Gen` and test for significance of interaction with liklihood ratio test.

```
aout.dg2<- aster(resp~varb + fit:(Den + Gen + Den*Gen), pred, fam, varb, id, root, data=redata)

summary(aout.dg2)
```

```
##
## Call:
## aster.formula(formula = resp ~ varb + fit:(Den + Gen + Den *
##     Gen), pred = pred, fam = fam, varvar = varb, idvar = id,
##     root = root, data = redata)
##
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -39.685580   0.573110 -69.246  < 2e-16 ***
## varbfrt         44.223124   0.581733  76.020  < 2e-16 ***
## varbfrt.2       29.337161   0.574020  51.108  < 2e-16 ***
## varbseeds       42.019372   0.572764  73.362  < 2e-16 ***
## fit:DenH         0.050631   0.003726  13.589  < 2e-16 ***
## fit:DenL        -0.061064   0.005868 -10.407  < 2e-16 ***
## fit:GenLG       -0.076651   0.004327 -17.716  < 2e-16 ***
## fit:DenL:GenLG   0.081842   0.006936  11.799  < 2e-16 ***
## fit:DenM:GenLG   0.039494   0.006678   5.914 3.34e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Original predictor variables dropped (aliased)
##      fit:DenM
```

```
anova(aout.dg, aout.dg2)
```

```
## Analysis of Deviance Table
##
## Model 1: resp ~ varb + fit:(Den + Gen)
## Model 2: resp ~ varb + fit:(Den + Gen + Den * Gen)
##   Model Df Model Dev Df Deviance P(>|Chi|)
## 1        7    133406
## 2        9    133542  2   135.58 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Finally, include plotID to model and perform yet anohter liklihood ratio test

```
aoutc<- aster(resp~varb + fit:(plotID + Den + Gen + Den*Gen), pred, fam, varb, id, root, data=redata)

summary(aoutc, show.graph=T)
```

```
##
## Call:
## aster.formula(formula = resp ~ varb + fit:(plotID + Den + Gen +
##     Den * Gen), pred = pred, fam = fam, varvar = varb, idvar = id,
##     root = root, data = redata)
##
##
## Graphical Model:
##  variable predecessor family
##  flw      root        bernoulli
##  frt      flw         poisson
```

```
##  frt.2    frt         poisson
##  seeds    frt.2       poisson
##
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -37.862132   0.594544 -63.683  < 2e-16 ***
## varbfrt       42.414140   0.602792  70.363  < 2e-16 ***
## varbfrt.2     27.620364   0.594622  46.450  < 2e-16 ***
## varbseeds     40.099155   0.596376  67.238  < 2e-16 ***
## fit:plotID1    0.107193   0.010629  10.085  < 2e-16 ***
## fit:plotID2    0.018053   0.012738   1.417  0.15641
## fit:plotID3   -0.055465   0.018987  -2.921  0.00349 **
## fit:plotID4   -0.028087   0.014469  -1.941  0.05223 .
## fit:plotID5    0.007564   0.012785   0.592  0.55408
## fit:plotID6    0.027366   0.012667   2.160  0.03074 *
## fit:plotID7    0.059572   0.011864   5.021 5.13e-07 ***
## fit:plotID8    0.061736   0.011796   5.233 1.66e-07 ***
## fit:plotID9    0.067894   0.011607   5.849 4.94e-09 ***
## fit:plotID10   0.087198   0.011071   7.876 3.38e-15 ***
## fit:plotID11   0.056926   0.011947   4.765 1.89e-06 ***
## fit:plotID12   0.100938   0.010755   9.385  < 2e-16 ***
## fit:plotID13   0.180417   0.009807  18.398  < 2e-16 ***
## fit:plotID14   0.076877   0.011345   6.776 1.23e-11 ***
## fit:plotID15   0.033981   0.012566   2.704  0.00685 **
## fit:plotID16   0.014427   0.012752   1.131  0.25793
## fit:plotID17   0.014738   0.012751   1.156  0.24777
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Original predictor variables dropped (aliased)
##      fit:plotID18
##      fit:DenL
##      fit:DenM
##      fit:GenLG
##      fit:DenL:GenLG
##      fit:DenM:GenLG
```

```
anova(aout.dg2, aoutc)
```

```
## Analysis of Deviance Table
##
## Model 1: resp ~ varb + fit:(Den + Gen + Den * Gen)
## Model 2: resp ~ varb + fit:(plotID + Den + Gen + Den * Gen)
##   Model Df Model Dev Df Deviance P(>|Chi|)
## 1        9    133542
## 2       21    134616 12   1074.7 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

That was fun. The results of these aster models (of entire data set) and liklihood ratio tests are presetned in Table 1 of the manuscript. The next steps will produce mean fitness and standard errors for these factors.

Calculation of Mean Fitness and Standard Errors

As the effects of `GEN` was significant in the above analyses, we can divide the data into high and low Ne data sets, and perform additional analsyis to calcualte mean fitness and standard errors for the density treatmetns.

The aster analyses for high and low Ne are performed in parallel below (i.e. each step is performed twice, once for high Ne and once for low Ne analses).

First, isloate high (HG) and low (LG) data from the main redata file, and drop unused levels. Therefore, don't have to do "reshape" data step. `HG`/`hg` = High Ne, `LG`/`lg` = Low Ne

```
redataHG<- subset(redata, Gen=="HG")
redataLG<- subset(redata, Gen=="LG")

redataHG<- droplevels(redataHG)
redataLG<- droplevels(redataLG)
```

Perform aster analysis on `HG` and `LG` data with just fitness data (no predictors), then add `Den` and perform a liklihood ratio test.

```
aoutHG<- aster(resp~varb, pred, fam, varb, id, root, data=redataHG)
aoutHG2<- aster(resp~varb + fit:(Den), pred, fam, varb, id, root, data=redataHG)

summary(aoutHG, show.graph = T)
```

```
##
## Call:
## aster.formula(formula = resp ~ varb, pred = pred, fam = fam,
##     varvar = varb, idvar = id, root = root, data = redataHG)
##
##
## Graphical Model:
##  variable predecessor family
##  flw      root        bernoulli
##  frt      flw         poisson
##  frt.2    frt         poisson
##  seeds    frt.2       poisson
##
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -46.7514     0.8383  -55.77   <2e-16 ***
## varbfrt      51.2764     0.8499   60.33   <2e-16 ***
## varbfrt.2    35.6147     0.8417   42.31   <2e-16 ***
## varbseeds    49.1697     0.8383   58.66   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(aoutHG2, show.graph=T)
```

```
##
## Call:
## aster.formula(formula = resp ~ varb + fit:(Den), pred = pred,
##     fam = fam, varvar = varb, idvar = id, root = root, data = redataHG)
##
##
## Graphical Model:
##  variable predecessor family
##  flw      root        bernoulli
##  frt      flw         poisson
##  frt.2    frt         poisson
##  seeds    frt.2       poisson
##
##                Estimate Std. Error z value Pr(>|z|)
```

```
## (Intercept) -37.702589   0.851082  -44.30   <2e-16 ***
## varbfrt      42.242416   0.862010   49.01   <2e-16 ***
## varbfrt.2    26.664567   0.850417   31.36   <2e-16 ***
## varbseeds    40.096837   0.850770   47.13   <2e-16 ***
## fit:DenH      0.048489   0.003650   13.28   <2e-16 ***
## fit:DenL     -0.070015   0.006872  -10.19   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Original predictor variables dropped (aliased)
##      fit:DenM
```

```
anova(aoutHG, aoutHG2)
```

```
## Analysis of Deviance Table
##
## Model 1: resp ~ varb
## Model 2: resp ~ varb + fit:(Den)
##   Model Df Model Dev Df Deviance P(>|Chi|)
## 1        4     89573
## 2        6     90187  2   614.18 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Perform same sequence of analyses on LG dataset

```
aoutLG<- aster(resp~varb, pred, fam, varb, id, root, data=redataLG)
aoutLG2<- aster(resp~varb + fit:(Den), pred, fam, varb, id, root, data=redataLG)

summary(aoutLG, show.graph = T)
```

```
##
## Call:
## aster.formula(formula = resp ~ varb, pred = pred, fam = fam,
##     varvar = varb, idvar = id, root = root, data = redataLG)
##
##
## Graphical Model:
##  variable predecessor family
##  flw      root        bernoulli
##  frt      flw         poisson
##  frt.2    frt         poisson
##  seeds    frt.2       poisson
##
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -42.0953     0.7497  -56.15   <2e-16 ***
## varbfrt      46.6262     0.7638   61.05   <2e-16 ***
## varbfrt.2    32.9264     0.7552   43.60   <2e-16 ***
## varbseeds    44.2799     0.7497   59.06   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(aoutLG2, show.graph=T)
```

```
##
## Call:
## aster.formula(formula = resp ~ varb + fit:(Den), pred = pred,
```

```
##     fam = fam, varvar = varb, idvar = id, root = root, data = redataLG)
##
##
## Graphical Model:
##  variable predecessor family
##  flw      root        bernoulli
##  frt      flw         poisson
##  frt.2    frt         poisson
##  seeds    frt.2       poisson
##
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -41.520615   0.776686 -53.459  < 2e-16 ***
## varbfrt      46.051828   0.790274  58.273  < 2e-16 ***
## varbfrt.2    32.355588   0.781521  41.401  < 2e-16 ***
## varbseeds    43.704518   0.776742  56.266  < 2e-16 ***
## fit:DenH      0.010151   0.005359   1.894  0.05822 .
## fit:DenL     -0.015065   0.005258  -2.865  0.00417 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Original predictor variables dropped (aliased)
##      fit:DenM
```

```
anova(aoutLG, aoutLG2)
```

```
## Analysis of Deviance Table
##
## Model 1: resp ~ varb
## Model 2: resp ~ varb + fit:(Den)
##   Model Df Model Dev Df Deviance P(>|Chi|)
## 1        4     43589
## 2        6     43612  2   23.643 7.345e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The effectds of density (`Den`) was significant in both the high (`HG`) and low (`LG`) effective genetic population size data sets.

First step is to generate MLE of saturated model mean value parameter vector: mu. Again, all step are performed twice, once for high and once for low Ne data. Because we want treatment-level estiamtes of fitness, we generate these estimates from the analyses that included density: `aoutHG` and `aoutLG`

```
pout.HG<- predict(aoutHG, se.fit=TRUE)

pout.LG<- predict(aoutLG, se.fit=TRUE)
```

Make design matrix data.frame of indivudals for each density level (low, med., high), that has a 1 for each element of the matrix. These will eventually be replaced with actual fitness values in later steps.

```
fred.hg <- data.frame( Den=levels(redataHG$Den), flw=1, frt=1,frt.2=1, seeds=1,root = 1)

fred.lg <- data.frame( Den=levels(redataLG$Den), flw=1, frt=1,frt.2=1, seeds=1,root = 1)
```

Reshape the design matrix just as the actual data

```
renewdata.hg <- reshape(fred.hg, varying = list(vars),
                     direction = "long", timevar = "varb",
                     times = as.factor(vars), v.names = "resp")
```

```
renewdata.lg <- reshape(fred.lg, varying = list(vars),
                        direction = "long", timevar = "varb",
                        times = as.factor(vars), v.names = "resp")
```

Make character string from "varb" of renewdata without actual values (i.e., the layers of varb in renewdata), and add it to each renewdata object

```
layer<- gsub("[0-9]", "", as.character(renewdata.hg$varb))


renewdata.hg<- data.frame(renewdata.hg, layer= layer)
renewdata.lg<- data.frame(renewdata.lg, layer= layer)
```

Add "seeds" in new layer column of renewdata as numeric, called fit Note: only need one `fit` object as it is the same for both High and Low Ne data, and add to each renew data file

```
fit<- as.numeric(layer=="seeds")

renewdata.gh<- data.frame(renewdata.hg, fit = fit)
renewdata.lg<- data.frame(renewdata.lg, fit = fit)
```

Rerun prediction of aster analyses, with the reshaped design matrices

```
pout.hg<- predict(aoutHG2, newdata= renewdata.hg, varvar= varb,
                  idvar = id, root = root, se.fit = TRUE)

pout.lg<- predict(aoutLG2, newdata= renewdata.lg, varvar= varb,
                  idvar = id, root = root, se.fit = TRUE)
```

Check class of each column in prediction outputs

```
sapply(pout.hg, class)
```

```
##       fit    se.fit  gradient    modmat
## "numeric" "numeric"  "matrix"   "array"
```

```
sapply(pout.lg, class)
```

```
##       fit    se.fit  gradient    modmat
## "numeric" "numeric"  "matrix"   "array"
```

Lengths of fit and se.fit (12) match row number of renewdata (as should be with predict.aster)

```
sapply(pout.hg, length)
```

```
##      fit   se.fit gradient   modmat
##       12       12       72       72
```

```
sapply(pout.lg, length)
```

```
##      fit   se.fit gradient   modmat
##       12       12       72       72
```

Therefore, we can make 12 CIs, one for each of 4 nodes of graphical model, and 3 density treatmetns (4 nodes x 3 treatments =12 estimates).

Put the parameter estimates into a matrix with individuals in rows and nodes in columns

Extract HG resutls, and produce a 3 x 4 matrix (3 density treatments by 4 nodes)

```
nnode<- length(vars)
sally.hg<- matrix(pout.hg$fit, ncol = nnode)
dim(sally.hg)
```

```
## [1] 3 4
```

Name the rows (by `Den` treatments) and columns (as nodes), and view the matrix

```
rownames(sally.hg)<- unique(as.character(renewdata.hg$Den))
colnames(sally.hg)<- unique(as.character(renewdata.hg$varb))

round(sally.hg, 3)
```

```
##     flw    frt  frt.2   seeds
## H 1.000 61.951 36.336 418.028
## L 0.861 34.901  5.659  57.833
## M 1.000 48.425 16.477 180.583
```

Now generate matrix of standard errors, and name rows and columns just as fitness estimates

```
nnode2<- length(vars)
sally2<- matrix(pout.hg$se.fit, ncol = nnode)
dim(sally2)
```

```
## [1] 3 4
```

```
rownames(sally2)<- unique(as.character(renewdata.hg$Den))
colnames(sally2)<- unique(as.character(renewdata.hg$varb))

round(sally2, 3)
```

```
##     flw   frt frt.2  seeds
## H 0.000 1.012 1.253 14.952
## L 0.058 2.412 0.560  5.972
## M 0.000 0.821 0.765  8.874
```

Combine estimates with standard errors for only final node: seeds

```
ests<- sally.hg[,grepl("seeds", colnames(sally.hg))]
se<-  sally2[,grepl("seeds", colnames(sally2))]

HG<- cbind(ests, se)
```

Perform the same steps for `LG` results

```
nnode<- length(vars)
sally.lg<- matrix(pout.lg$fit, ncol = nnode)
dim(sally.lg)
```

```
## [1] 3 4
```

```
rownames(sally.lg)<- unique(as.character(renewdata.lg$Den))
colnames(sally.lg)<- unique(as.character(renewdata.lg$varb))

round(sally.lg, 3)
```

```
##     flw    frt  frt.2   seeds
## H 0.977 45.194 13.701 122.917
## L 0.735 31.994  7.757  67.861
## M 0.927 41.784 11.570 102.750
```

Extract standard errors

```
nnode2<- length(vars)
sally.lg2<- matrix(pout.lg$se.fit, ncol = nnode)
dim(sally.lg2)
```

```
## [1] 3 4
```

```
rownames(sally.lg2)<- unique(as.character(renewdata.lg$Den))
colnames(sally.lg2)<- unique(as.character(renewdata.lg$varb))

round(sally.lg2, 3)
```

```
##     flw   frt frt.2 seeds
## H 0.014 1.188 0.772 7.298
## L 0.068 3.086 0.924 8.279
## M 0.033 1.840 0.825 7.632
```

Combine estimates with standard errors for only final node: seeds

```
ests<- sally.lg[,grepl("seeds", colnames(sally.lg))]
se<-  sally.lg2[,grepl("seeds", colnames(sally.lg2))]

LG<- cbind(ests, se)
```

These are the fitness and standard errors for HG and LG treatments (across densities)

```
HG
```

```
##        ests        se
## H 418.02778 14.951711
## L  57.83334  5.971818
## M 180.58333  8.874330
```

```
LG
```

```
##       ests       se
## H 122.9167 7.298401
## L  67.8611 8.278818
## M 102.7500 7.631958
```