Aster Models and Lande-Arnold Beta

By
Charles J. Geyer and Ruth G. Shaw
Technical Report No. 675
School of Statistics
University of Minnesota

Original January 9, 2010
Revised January 13, 2010

Abstract

Lande and Arnold (1983) proposed an estimate of beta, the directional selection gradient, by ordinary least squares (OLS). Aster models (Geyer, Wagenius and Shaw, 2007; Shaw, Geyer, Wagenius, Hangelbroek, and Etterson, 2008) estimate exactly the same beta, so providing no improvement over the Lande-Arnold method in point estimation of this quantity. Aster models do provide correct confidence intervals, confidence regions, and hypothesis tests for beta; in contrast, such procedures derived from OLS are often invalid because the assumptions for OLS are grossly incorrect. This revision fixes a bug which made the figure incorrect in the original.

We use data simulated in Geyer and Shaw (2008) and available in the dataset `sim` in the R contributed package `aster`. Fitness landscapes for these data are computed in Geyer and Shaw (2008). Confidence regions for the maximum of the fitness landscape are computed in Geyer and Shaw (2010). Here we compute confidence regions for Lande-Arnold beta.

First we compute beta, which is found by OLS regression of relative fitness (fitness divided by average fitness) on centered phenotypic predictor variables.

```
> library(aster)
> data(sim)
> w <- ladata$y / mean(ladata$y)
> z1 <- ladata$z1 - mean(ladata$z1)
> z2 <- ladata$z2 - mean(ladata$z2)
```

The vector `w` is relative fitness. Vectors `z1` and `z2` are centered phenotypic predictor variables.

```
> bout <- lm(w ~ z1 + z2)
> summary(bout)

Call:
lm(formula = w ~ z1 + z2)

Residuals:
    Min      1Q  Median      3Q     Max
-1.8315 -0.5906 -0.1215  0.5026  3.4714

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
```

```
(Intercept)  1.00000    0.03844  26.013   <2e-16 ***
z1           0.69969    0.04449  15.729   <2e-16 ***
z2          -0.10355    0.04250  -2.436   0.0152 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.8596 on 497 degrees of freedom
Multiple R-squared:  0.3623,        Adjusted R-squared:  0.3598
F-statistic: 141.2 on 2 and 497 DF,  p-value: < 2.2e-16
```

Because we are using relative fitness the intercept term is known to be 1 Thus we refit fixing the intercept to be one (because the degrees of freedom are so large, this has negligible effect on standard errors).

```
> bout <- lm(w ~ 0 + z1 + z2, offset = rep(1, length(w)))
> summary(bout)

Call:
lm(formula = w ~ 0 + z1 + z2, offset = rep(1, length(w)))

Residuals:
    Min      1Q  Median      3Q     Max
-1.8315 -0.5906 -0.1215  0.5026  3.4714

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
z1  0.69969    0.04444  15.744   <2e-16 ***
z2 -0.10355    0.04246  -2.439   0.0151 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.8587 on 498 degrees of freedom
Multiple R-squared:  0.3623,        Adjusted R-squared:  0.3598
F-statistic: 141.5 on 2 and 498 DF,  p-value: < 2.2e-16

> beta <- as.numeric(bout$coefficients)
> print(beta)

[1]  0.6996907 -0.1035472
```

Now we convince ourselves that the “observed equals expected” property of maximum likelihood estimation in exponential families implies that the

best linear approximation (BLA) of the fitness landscape estimated by the aster model is the same as the same as the BLA fit by the Lande-Arnold method.

```
> out6 <- aster(resp ~ varb + 0 + z1 + z2 + I(z1^2) + I(z1*z2) + I(z2^2),
+    pred, fam, varb, id, root, data = redata)
> summary(out6)

Call:
aster.formula(formula = resp ~ varb + 0 + z1 + z2 + I(z1^2) +
    I(z1 * z2) + I(z2^2), pred = pred, fam = fam, varvar = varb,
    idvar = id, root = root, data = redata)

             Estimate Std. Error z value Pr(>|z|)
varbiflow1 -3.444251   0.180123 -19.122  < 2e-16 ***
varbiflow2 -3.064152   0.203311 -15.071  < 2e-16 ***
varbiflow3 -3.207467   0.218952 -14.649  < 2e-16 ***
varbiflow4 -3.284180   0.236597 -13.881  < 2e-16 ***
varbisurv1 -0.065167   0.160348  -0.406  0.68444
varbisurv2 -0.700847   0.225747  -3.105  0.00191 **
varbisurv3 -0.094013   0.275111  -0.342  0.73256
varbisurv4  1.217672   0.234288   5.197 2.02e-07 ***
varbnflow1 -7.264353   0.090581 -80.198  < 2e-16 ***
varbnflow2 -7.452760   0.102617 -72.627  < 2e-16 ***
varbnflow3 -7.227782   0.105711 -68.373  < 2e-16 ***
varbnflow4 -7.044131   0.107792 -65.349  < 2e-16 ***
varbngerm1 -2.264595   0.030308 -74.720  < 2e-16 ***
varbngerm2 -2.270312   0.033766 -67.237  < 2e-16 ***
varbngerm3 -2.325980   0.036102 -64.429  < 2e-16 ***
varbngerm4 -2.304824   0.036977 -62.332  < 2e-16 ***
varbnseed1  2.881224   0.009182 313.789  < 2e-16 ***
varbnseed2  2.895118   0.010258 282.241  < 2e-16 ***
varbnseed3  2.880964   0.010737 268.332  < 2e-16 ***
varbnseed4  2.864026   0.011117 257.619  < 2e-16 ***
z1          0.146950   0.013695  10.730  < 2e-16 ***
z2         -0.020598   0.009842  -2.093  0.03637 *
I(z1^2)    -0.027807   0.009508  -2.925  0.00345 **
I(z1 * z2)  0.023713   0.012352   1.920  0.05489 .
I(z2^2)    -0.017986   0.006536  -2.752  0.00593 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

This is the aster model fit in Geyer and Shaw (2008). Note that this fitness landscape is quadratic on the canonical parameter scale. We are not trying to fit a linear fitness landscape. Rather we are trying to estimate the fitness landscape as best we can.

Now we get the fitness landscape itself. The aster model has the dependence graph shown in Geyer and Shaw (2008, Section 2.1). There are 25 fitness component variables measured for each individual, of which only the last 4, number of seeds germinated in each of four time periods contribute directly to observed fitness (the other variables contribute indirectly in being predecessors, predecessors of predecessors, etc. of these four).

```
> pout6 <- predict(out6)
> pout6 <- matrix(pout6, nrow = nrow(out6$x), ncol = ncol(out6$x))
> colnames(pout6) <- colnames(out6$x)
> mufit <- pout6[ , grep("germ", colnames(pout6))]
> mufit <- apply(mufit, 1, "sum")
> length(mufit)
```

```
[1] 500
```

Now `mufit` is the (MLE of) expected fitness for each individual. When divided by its mean, it gives the (MLE of) expected relative fitness for each individual. We check that OLS regression of this on `z1` and `z2` giving the BLA of (the MLE of) the fitness landscape is the same as the Lande-Arnold beta.

```
> wmu <- mufit / mean(mufit)
> wmout <- lm(wmu ~ z1 + z2)
> all.equal(beta, as.numeric(wmout$coefficients[-1]))
```

```
[1] TRUE
```

Because this is a simulated dataset, we know the simulation truth fitness landscape, which is given by the R object `mu.true` in the `sim` dataset. So we treat `mu.true` as we did `pout6` above, to get the simulation truth beta.

```
> wmu.true <- matrix(mu.true, nrow = nrow(out6$x), ncol = ncol(out6$x))
> wmu.true <- wmu.true[ , grep("germ", colnames(pout6))]
> wmu.true <- apply(wmu.true, 1, "sum")
> wmu.true <- wmu.true / mean(wmu.true)
> wmout.true <- lm(wmu.true ~ z1 + z2)
> beta.true <- as.vector(wmout.true$coefficients[-1])
> print(beta.true)
```

```
[1]  0.7107269 -0.1341213
```

Summarizing where we are at this point, `beta.true` is the true (simulation truth) unknown (in real data unknown, in simulated data known) parameter value, the slope of the BLA of the relative fitness landscape, and `beta` is its MLE using the aster model. Of course, `beta` is also the MLE under the assumption that relative fitness is exactly homoscedastic normal, which is, of course, grossly incorrect. Confidence intervals for beta derived from the aster model are correct; those derived from OLS are invalid.

We now proceed to find confidence intervals for the components of beta and a confidence region for the vector beta derived from the aster model and the asymptotics of maximum likelihood estimation.

First we look at another way of estimating beta, using what Lande and Arnold (1983) call the "multivariate generalization of the results of Robertson (1966) and Price (1970, 1972) [see Lande and Arnold (1983) for citations]"

$$\beta = P^{-1} \operatorname{cov}(w, z),$$

where z is the random vector of phenotypic predictor values and P is its variance-covariance matrix. This shows we can estimate beta as follows

```
> z <- cbind(z1, z2, deparse.level = 0)
> zvarinv <- solve(t(z) %*% z / nrow(z))
> zwcov <- t(z) %*% cbind(mufit) / nrow(z)
> all.equal(beta, as.numeric(zvarinv %*% zwcov) / mean(mufit))
```

```
[1] TRUE
```

Now we want to do the same calculation starting with `predict(out6)`

```
> mu <- predict(out6)
> amat <- matrix(0, nrow = length(mufit), ncol = length(mu))
> blank <- matrix(0, nrow = nrow(pout6), ncol = ncol(pout6))
> blank.idx <- grep("germ", colnames(pout6))
> for (i in 1:nrow(amat)) {
+     boom <- blank
+     boom[i, blank.idx] <- 1
+     amat[i, ] <- as.vector(boom)
+ }
> all.equal(mufit, as.vector(amat %*% mu))
```

```
[1] TRUE
```

```
> bmat <- zvarinv %*% t(z) %*% amat / nrow(z)
> all.equal(beta, as.numeric(bmat %*% mu) / mean(as.numeric(amat %*% mu)))

[1] TRUE

> cmat <- apply(amat, 2, sum) / nrow(z)
> cmat <- rbind(cmat)
> all.equal(beta, as.numeric(bmat %*% mu) / as.numeric(cmat %*% mu))

[1] TRUE

> dmat <- rbind(bmat, cmat, deparse.level = 0)
> all.equal(beta, as.numeric(dmat %*% mu)[1:2] / as.numeric(dmat %*% mu)[3])

[1] TRUE

> d3way <- array(as.vector(t(dmat)), dim = c(dim(out6$modmat)[1:2], nrow(dmat)))
> dout <- predict(out6, amat = d3way, se.fit = TRUE)
> all.equal(beta, dout$fit[1:2] / dout$fit[3])

[1] TRUE
```

If we denote the R object `dout$fit` as the vector ζ in mathematical notation, then

$$\beta_i = \zeta_i / \zeta_3, \qquad i = 1, 2.$$

This is a non-linear transformation $\zeta \to \beta$ that has Jacobian matrix

$$\begin{pmatrix} 1/\zeta_3 & 0 & -\zeta_1/\zeta_3^2 \\ 0 & 1/\zeta_3 & -\zeta_2/\zeta_3^2 \end{pmatrix}$$

constructed in R as

```
> zeta1 <- dout$fit[1]
> zeta2 <- dout$fit[2]
> zeta3 <- dout$fit[3]
> jacobian <- rbind( c( 1 / zeta3, 0, - zeta1 / zeta3^2 ),
+                    c( 0, 1 / zeta3, - zeta2 / zeta3^2 ) )
```

Because of this nonlinearity, which arises from the fact that we measure actual fitness not relative fitness, hence must estimate it by dividing actual fitness by mean fitness, OLS would not calculate correct standard errors even

if actual fitness were homoscedastic normal. Our next step takes this issue into account correctly.

The function `predict.aster` does not give the full variance-covariance matrix for "predicted values" like `dout$fit`. It does, however, give a component `gradient` which is $\partial\zeta/\partial\beta$ and can be used to calculate the asymptotic variance-covariance matrix for ζ

```
> dvar <- dout$gradient %*% solve(out6$fisher) %*% t(dout$gradient)
> all.equal(dout$se.fit, sqrt(diag(dvar)))

[1] TRUE

> print(dvar)

            [,1]         [,2]         [,3]
[1,]  0.11074988 -0.049378460  0.051190895
[2,] -0.04937846  0.099797054 -0.008332417
[3,]  0.05119090 -0.008332417  0.092359357
```

Then the delta method is applied to get the asymptotic variance-covariance matrix for β

```
> bvar <- jacobian %*% dvar %*% t(jacobian)
> print(bvar)

              [,1]          [,2]
[1,]  0.0012646355 -0.0006739173
[2,] -0.0006739173  0.0014855497
```

The diagonal elements give standard errors for beta

```
> foo <- cbind(beta, sqrt(diag(bvar)))
> colnames(foo) <- c("beta", "se(beta)")
> print(foo)

           beta   se(beta)
[1,]  0.6996907 0.03556171
[2,] -0.1035472 0.03854283
```

If one compares these standard errors (derived from the aster model) with the putative (and erroneous) standard errors derived from OLS (page 2), one sees that the putative OLS standard errors are larger than correct standard

errors based on a defensible statistical model. A 95% confidence interval for β_2 is $(-0.187, -0.02)$ based on OLS and $(-0.179, -0.028)$ based on the aster model. The P-value for the two-tailed test with null hypothesis $\beta_2 = 0$ is $P = 0.015$ based on OLS and ($P = 0.007$) using the aster model. The incorrect OLS confidence interval is wider than it should be, and the incorrect OLS P-value is larger than it should be.

One might draw the lesson from this one example that OLS standard errors are always conservative, but there is no mathematics to justify this. The OLS standard errors (in the context of Lande-Arnold analysis) are just wrong and should never be used.

We can also make an elliptical confidence region that accounts for the correlation of the components of β. The following R statements make Figure 1 (page 9)

```
> fred <- eigen(bvar, symmetric = TRUE)
> sally <- fred$vectors %*% diag(sqrt(fred$values)) %*% t(fred$vectors)
> zoo1 <- cos(seq(0, 2 * pi, length = 101))
> zoo2 <- sin(seq(0, 2 * pi, length = 101))
> zoo <- rbind(zoo1, zoo2)
> jane <- sqrt(qchisq(0.95, 2)) * sally %*% zoo
> par(mar = c(5, 4, 1, 1) + 0.1)
> plot(beta[1] + jane[1, ], beta[2] + jane[2, ], type = "l",
+     xlab = expression(beta[1]), ylab = expression(beta[2]),
+     ylim = range(c(0, beta[2] + jane[2, ])))
> points(beta[1], beta[2], pch = 19)
> points(beta.true[1], beta.true[2])
> jane <- sqrt(qchisq(0.50, 2)) * sally %*% zoo
> lines(beta[1] + jane[1, ], beta[2] + jane[2, ], lty = "dashed")
```

Note that in Figure 1 the 95% confidence ellipse does not cross either coordinate axis. This says β_1 is statistically significantly greater than zero and β_2 is statistically significantly less than zero (at the 0.05 significance level), even accounting for doing two tests.

If one were going to use beta to plug into the multivariate breeder's equation (mean response to selection equals $G\beta$, where G is the variance-covariance matrix of the breeding values derived from a quantitative genetics model) one could use the variance-covariance matrix for $\hat{\beta}$ (the matrix `bvar`) with another application of the delta method involving the asymptotic variance-covariance matrix of G (if it is also estimated from data) to get a variance-covariance matrix for the response to selection.

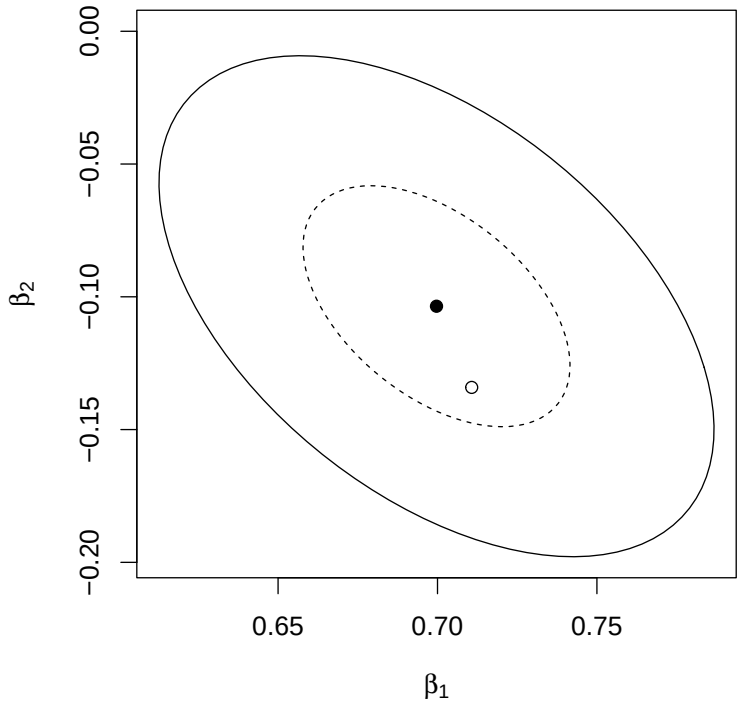

Figure 1: Confidence ellipses for beta. Solid curve is boundary of 95% confidence region, dashed curve is boundary of 50% confidence region, solid dot is location of MLE for beta, hollow dot is location of simulation truth value of beta.

References

Geyer, C. J., and Shaw, R. G. (2008). Supporting data analysis for a talk to be given at Evolution 2008 University of Minnesota, June 20–24. Technical Report No. 669. School of Statistics, University of Minnesota. `http://www.stat.umn.edu/geyer/aster/`.

Geyer, C. J., and Shaw, R. G. (2010). Hypothesis Tests and Confidence Intervals Involving Fitness Landscapes fit by Aster Models. Technical Report No. 674 revised. School of Statistics, University of Minnesota. `http://www.stat.umn.edu/geyer/aster/`.

Geyer, C. J., Wagenius, S., and Shaw, R. G. (2007). Aster models for life history analysis. *Biometrika* 94:415–426.

Lande, R., and Arnold, S. J. (1983). The measurement of selection on correlated characters. *Evolution* 37:1210–1226.

Shaw, R. G., Geyer, C. J., Wagenius, S., Hangelbroek, H. H., and Etterson, J. R. (2008). Unifying life history analysis for inference of fitness and population growth. *American Naturalist* 172:E35–E47.