Yet More Supporting Data Analysis for "Unifying Life History Analysis for Inference of Fitness and Population Growth"

By
Ruth G. Shaw, Charles J. Geyer, Stuart Wagenius,
Helen H. Hangelbroek, and Julie R. Etterson

Technical Report No. 666
School of Statistics
University of Minnesota
January 2, 2008

Abstract

This technical report (TR) gives details of a data reanalysis backing up a paper having the same authors as this TR and having the title that is quoted in the title of this TR. Two previous TR, 658 and 661, have the bulk of the supporting data analysis this paper. This TR deals with one minor issue, Box-Cox transformation of predictor variables to make them more normal and the effect of such transformation on the estimation of fitness surfaces, in particular on Figure 3 of the paper, which is also Figure 14 of TR 661. The sole objective of this TR is to produce the analog of that figure using transformed predictors.

1 Discussion

Traditionally discussion goes at the end, but since the point of this technical report (TR) is very simple, we put it here. The job of this TR is to produce Figure 6. This figure is to be compared with Figure 3 in Shaw, Geyer, Wagenius, Hangelbroek, and Etterson (submitted), which was produced as Figure 14 of TR 661 (Shaw, et al., 2007)

The only difference between these two figures is that Figure 3 of the paper uses log transformation of the two predictor variables (the x- and y-axes of the plot) and Figure 6 of this TR uses Box-Cox transformations.

The point of the Box-Cox transformations is that Lande-Arnold theory (Lande and Arnold, 1983) requires joint multivariate normality of predictor variables. Aster theory does not, so as far as aster analysis is concerned, Figure 3 of the paper with its more conventional log transformation is just fine. It did, however, occur to us that someone might raise the issue that we are being unfair to Lande and Arnold (1983) in not making our best effort to transform to multivariate normality. There being no really good methods for transformation to multivariate normality (Andrews et al., 1971; Riani, 2004), we do Box-Cox transformation (Box and Cox, 1964; Venables and Ripley, 2002, pp. 170–172) of each predictor variable separately. This, of course, need not even produce univariate normality of each variable separately; it merely does the best job of any power transformation of producing univariate normality.

In this example, there seems to be little point to the Box-Cox transformation. Qualitatively, nothing changes.

- The aster estimate of the fitness surface still has a peak, the best quadratic approximation (Lande-Arnold estimate) has a saddle.
- The peak of the fitness landscape is near the edge of the distribution of predictor values, hence this should not be called "stabilizing selection" on leaf number but "directional selection."
- The disagreement between the aster estimate (peak) and the Lande-Arnold estimate (saddle) is entirely due to the inability of a quadratic function to fit both a peak and a flat region. Having to choose one or the other it chooses a saddle as its best approximation to the flat region to the left edge of the plot.

The Box-Cox transformation might have made a difference in all of these aspects, but in this particular example it did not.

2 Data and Box-Cox

We reanalyze a subset of the data analyzed by Etterson and Shaw (2001). These data are in the `chamae2` dataset in the `aster` con-

tributed package to the R statistical computing environment.

```
> library(aster)
> data(chamae2)
```

The only difference between the analysis in this technical report (TR) and the corresponding analysis in TR 661 is that we do a Box-Cox transformation (Box and Cox, 1964; Venables and Ripley, 2002, pp. 170–172) of the predictor variables called SLA and LN in the paper but called LOGSLA and LOGLVS in the dataset.

```
> sla <- 10^chamae2$LOGSLA
> lvs <- 10^chamae2$LOGLVS
> library(MASS)
> out.sla <- boxcox(sla ~ 1, plotit = FALSE)
> out.lvs <- boxcox(lvs ~ 1, plotit = FALSE)
> lambda.sla <- out.sla$x[out.sla$y == max(out.sla$y)]
> lambda.lvs <- out.lvs$x[out.lvs$y == max(out.lvs$y)]
> print(lambda.sla)

[1] -0.2

> print(lambda.lvs)

[1] 0.3

> chamae2$LOGSLA <- sla^lambda.sla
> chamae2$LOGLVS <- lvs^lambda.lvs
```

Figure 1 (page 3) shows the Box-Cox plot for SLA. Figure 2 (page 3) shows the Box-Cox plot for LN.

These data are already in "long" format, no need to use the `reshape` function on them to do aster analysis. We will, however, need the "wide" format for Lande-Arnold analysis. So we do that, before making any changes (we will add newly defined variables) to `chamae2`.

```
> chamae2w <- reshape(chamae2, direction = "wide", timevar = "varb",
+     v.names = "resp", varying = list(levels(chamae2$varb)))
> names(chamae2w)

[1] "id"     "root"   "STG1N"  "LOGLVS" "LOGSLA" "BLK"    "fecund"
[8] "fruit"
```

3 Aster Analysis

We need to choose the non-exponential-family parameter (size) for the negative binomial distribution, since the `aster` package only does maximum likelihood for exponential family parameters. We start with

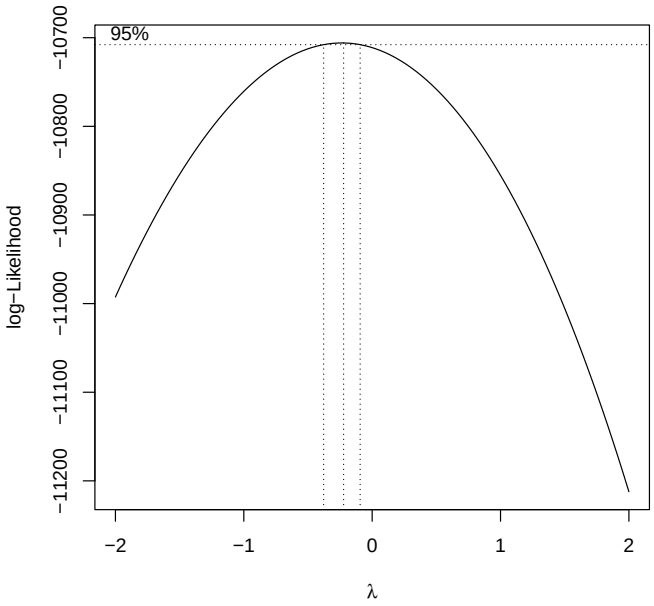

Figure 1: Box-Cox Plot for SLA.

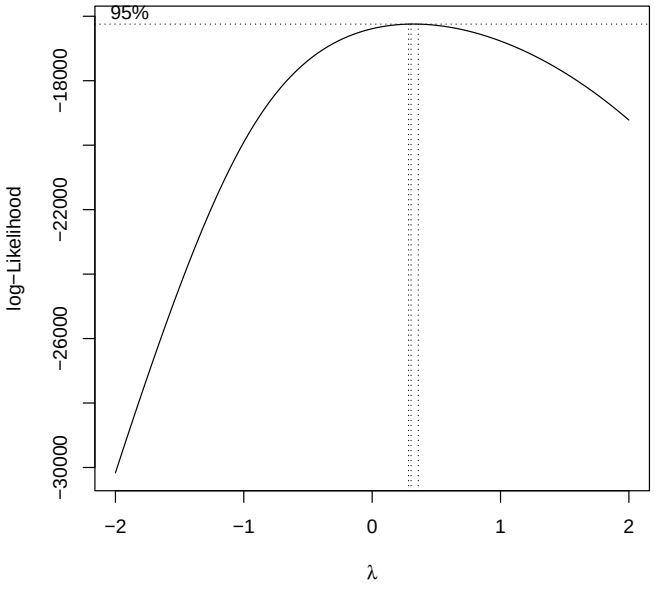

Figure 2: Box-Cox Plot for LN.

the following value, which was chosen with knowledge of the maximum likelihood estimate for this parameter, which we find in Section 3.1. The value that is found then is written out to a file and loaded here if the file exists, so after several runs (of `Sweave`) we are reading in here the maximum likelihood value of this non-exponential-family parameter.

```
> options(show.error.messages = FALSE, warn = -1)
> try(load("chamae2-alpha.rda"))
> options(show.error.messages = TRUE, warn = 0)
> ok <- exists("alpha.fruit")
> if (! ok) {
+     alpha.fruit <- 3.0
+ }
> print(alpha.fruit)

[1] 3
```

Then we set up the aster model framework.

```
> vars <- c("fecund", "fruit")
> pred <- c(0, 1)
> famlist <- list(fam.bernoulli(),
+     fam.truncated.negative.binomial(size = alpha.fruit, truncation = 0))
> fam <- c(1,2)
```

We make up new predictors that apply only to the variable `fruit`.

```
> foo <- as.numeric(as.character(chamae2$varb) == "fruit")
> chamae2$LOGLVSfr <- chamae2$LOGLVS * foo
> chamae2$LOGSLAfr <- chamae2$LOGSLA * foo
> chamae2$STG1Nfr <- chamae2$STG1N * foo
```

Now we fit the model called `out7` in TR 661, which is the one used for fitness surface estimation. The only difference is that here we have transformed the predictor variables.

```
> out7 <- aster(resp ~ varb + BLK + LOGLVSfr + LOGSLAfr + I(LOGLVSfr^2) +
+     I(LOGSLAfr^2) + I(LOGLVSfr * LOGSLAfr) + STG1Nfr,
+     pred, fam, varb, id, root, data = chamae2, famlist = famlist)
> summary(out7, info.tol = 1e-10)

Call:
aster.formula(formula = resp ~ varb + BLK + LOGLVSfr + LOGSLAfr +
    I(LOGLVSfr^2) + I(LOGSLAfr^2) + I(LOGLVSfr * LOGSLAfr) +
    STG1Nfr, pred = pred, fam = fam, varvar = varb, idvar = id,
    root = root, data = chamae2, famlist = famlist)
```

```
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)             -6.777e+00  1.466e-01 -46.223  < 2e-16 ***
varbfruit                6.887e+00  2.263e-01  30.437  < 2e-16 ***
BLK2                    -3.075e-03  4.474e-04  -6.873 6.30e-12 ***
BLK4                    -3.288e-05  4.136e-04  -0.079   0.9366
LOGLVSfr                 4.822e-02  7.043e-03   6.847 7.55e-12 ***
LOGSLAfr                 1.011e+00  2.246e-01   4.504 6.67e-06 ***
I(LOGLVSfr^2)           -1.914e-03  1.598e-04 -11.975  < 2e-16 ***
I(LOGSLAfr^2)           -3.296e-01  7.962e-02  -4.140 3.47e-05 ***
I(LOGLVSfr * LOGSLAfr) -1.317e-02  4.856e-03  -2.712   0.0067 **
STG1Nfr                 -1.448e-03  2.054e-04  -7.050 1.79e-12 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

3.1 Maximum Likelihood Estimation of Size

The `aster` function does not calculate the correct likelihood when the size parameters are considered unknown, because it drops terms that do not involve the exponential family parameters. However, the full log likelihood is easily calculated in R.

```
> x <- out7$x
> logl <- function(alpha.fruit, theta, x) {
+     x.fecund <- x[ , 1]
+     theta.fecund <- theta[ , 1]
+     p.fecund <- 1 / (1 + exp(- theta.fecund))
+     logl.fecund <- sum(dbinom(x.fecund, 1, p.fecund, log = TRUE))
+     x.fruit <- x[x.fecund == 1, 2]
+     theta.fruit <- theta[x.fecund == 1, 2]
+     p.fruit <- (- expm1(theta.fruit))
+     logl.fruit <- sum(dnbinom(x.fruit, size = alpha.fruit,
+         prob = p.fruit, log = TRUE) - pnbinom(0, size = alpha.fruit,
+         prob = p.fruit, lower.tail = FALSE, log = TRUE))
+     logl.fecund + logl.fruit
+ }
```

We then calculate the profile likelihood for the size parameter `alpha.fruit` maximizing over the other parameters, evaluating the profile log likelihood on a grid of points. We do not do this if the results would be the same as we got last time and have stored in the variable `logl.seq`.

```
> ok <- exists("alpha.fruit.save") && (alpha.fruit.save == alpha.fruit) &&
+     exists("coef.save") && isTRUE(all.equal(coef.save, coefficients(out7)))
> print(ok)

[1] FALSE
```

```
> alpha.fruit.seq <- seq(1.5, 4.5, 0.25)
> if (! ok) {
+ logl.seq <- double(length(alpha.fruit.seq))
+     for (i in 1:length(alpha.fruit.seq)) {
+         famlist.seq <- famlist
+         famlist.seq[[2]] <- fam.truncated.negative.binomial(size =
+             alpha.fruit.seq[i], truncation = 0)
+         out7.seq <- aster(out7$formula, pred, fam, varb, id, root,
+             data = chamae2, famlist = famlist.seq, parm = out7$coefficients)
+         theta.seq <- predict(out7.seq, model.type = "cond",
+             parm.type = "canon")
+         dim(theta.seq) <- dim(x)
+         logl.seq[i] <- logl(alpha.fruit.seq[i], theta.seq, x)
+     }
+ }
> ##### interpolate #####
> alpha.foo <- seq(min(alpha.fruit.seq), max(alpha.fruit.seq), 0.01)
> logl.foo <- spline(alpha.fruit.seq, logl.seq, n = length(alpha.foo))$y
> imax <- seq(along = alpha.foo)[logl.foo == max(logl.foo)]
> alpha.fruit.save <- alpha.fruit
> alpha.fruit <- alpha.foo[imax]
> coef.save <- coefficients(out7)
> ##### save #####
> if (! ok) {
+     save(alpha.fruit, alpha.fruit.save, coef.save, logl.seq,
+         file = "chamae2-alpha.rda", ascii = TRUE)
+ }
```

At the end of this chunk we save the maximum likelihood estimate in a file which is read in at the beginning of this document. We also save some extra information so there is no need to do this step every time if there is no change in the alpha.

Figure 3 (page 7) shows the profile log likelihood for the size parameter.

3.2 The Fitness Landscape

We calculate for just one value of `BLK` and `STG1N`.

```
> theblk <- "1"
> thestg <- 1
```

Figure 4 (page 8) shows the scatter plots of the two phenotypic variables (`LOGLVS` and `LOGSLA`, labeled `LN` and `SLA` because that is what they are called in the paper). It is made by the following code.

```
> xlab <- quote(LN^2)
> xlab[[3]] <- lambda.lvs
```

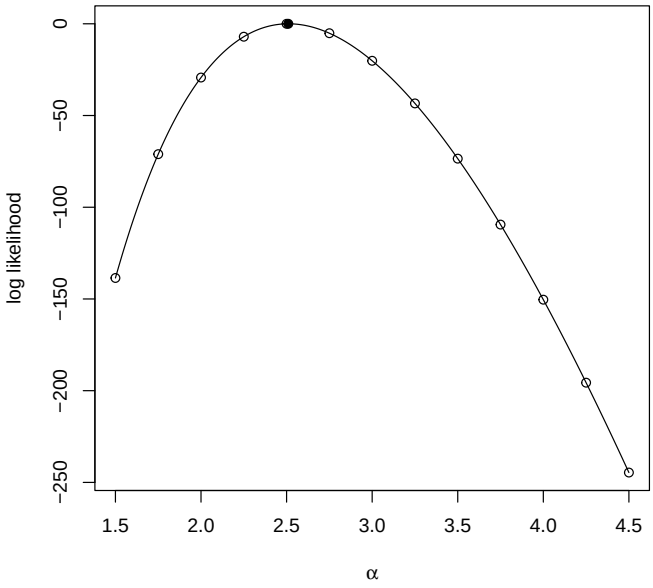

Figure 3: Profile log likelihood for size parameter for the (zero-truncated) negative binomial distribution of fruit. Hollow dots are points at which the log likelihood was evaluated exactly. Curve is the interpolating spline. Solid dot is maximum likelihood estimate.

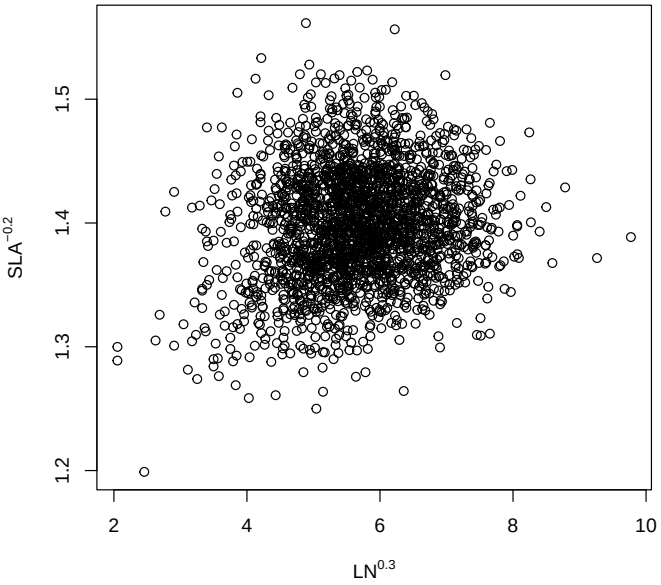

Figure 4: Scatterplot of phenotypic variables.

```
> xlab <- as.expression(xlab)
> ylab <- quote(SLA^2)
> ylab[[3]] <- lambda.sla
> ylab <- as.expression(ylab)
> plot(chamae2w$LOGLVS, chamae2w$LOGSLA, xlab = xlab, ylab = ylab)
```

The point of making the plot Figure 4 is that we want to add contour lines showing the estimated fitness landscape. To do that we first start with a grid of points across the figure.

```
> ufoo <- par("usr")
> nx <- 101
> ny <- 101
> z <- matrix(NA, nx, ny)
> x <- seq(ufoo[1], ufoo[2], length = nx)
> y <- seq(ufoo[3], ufoo[4], length = ny)
> xx <- outer(x, y^0)
> yy <- outer(x^0, y)
> xx <- as.vector(xx)
> yy <- as.vector(yy)
> n <- length(xx)
```

Then we create an appropriate `newdata` argument for the `predict.aster` function to "predict" at these points

```
> newdata <- data.frame(
+     BLK = factor(rep(theblk, n), levels = levels(chamae2$BLK)),
+     STG1N = rep(thestg, n), LOGLVS = xx, LOGSLA = yy, fecund = rep(1, n),
+     fruit = rep(3, n))
> renewdata <- reshape(newdata, varying = list(vars), direction = "long",
+     timevar = "varb", times = as.factor(vars), v.names = "resp")
> renewdata <- data.frame(renewdata, root = 1)
> foo <- as.numeric(as.character(renewdata$varb) == "fruit")
> renewdata$LOGLVSfr <- renewdata$LOGLVS * foo
> renewdata$LOGSLAfr <- renewdata$LOGSLA * foo
> renewdata$STG1Nfr <- renewdata$STG1N * foo
```

Then we predict the unconditional mean value parameter τ, for which the "fruit" component is expected fitness.

```
> tau <- predict(out7, newdata = renewdata, varvar = varb, idvar = id,
+     root = root)
> tau <- matrix(tau, nrow = nrow(newdata), ncol = ncol(out7$x))
> dimnames(tau) <- list(NULL, vars)
> zfit <- tau[ , "fruit"]
```

Figure 5 (page 10), which is made by the following code, shows it.

```
> plot(chamae2w$LOGLVS, chamae2w$LOGSLA, xlab = xlab, ylab = ylab, pch = ".")
> zfit <- matrix(zfit, nrow = length(x))
> contour(x, y, zfit, add = TRUE)
> contour(x, y, zfit, levels = c(5, 10, 25), add = TRUE)
```

3.3 Lande-Arnold Analysis

In contrast to the aster analysis, the Lande-Arnold analysis is very simple.

```
> lout <- lm(fruit ~ LOGLVS + LOGSLA + STG1N + I(LOGLVS^2) +
+     I(LOGLVS * LOGSLA) + I(LOGSLA^2), data = chamae2w)
> summary(lout)

Call:
lm(formula = fruit ~ LOGLVS + LOGSLA + STG1N + I(LOGLVS^2) +
    I(LOGLVS * LOGSLA) + I(LOGSLA^2), data = chamae2w)

Residuals:
    Min      1Q  Median      3Q     Max
-484.81  -67.74  -11.65   53.89  697.87

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

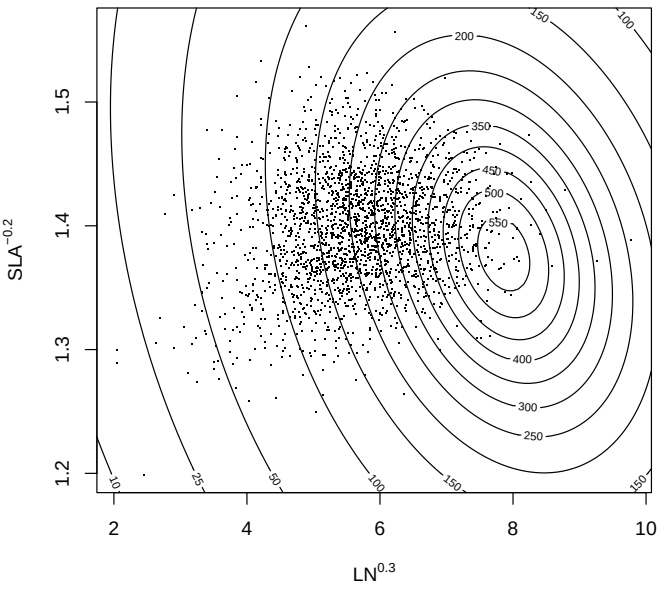

Figure 5: Scatterplot of phenotypic variables with contours of fitness landscape estimated by the aster model.

```
(Intercept)        -5691.028   1705.142  -3.338 0.000859 ***
LOGLVS               -12.787     81.954  -0.156 0.876026
LOGSLA              8126.090   2476.600   3.281 0.001050 **
STG1N                -17.479      2.832  -6.171 8.03e-10 ***
I(LOGLVS^2)           11.471      1.993   5.757 9.76e-09 ***
I(LOGLVS * LOGSLA)   -21.767     60.657  -0.359 0.719740
I(LOGSLA^2)        -2849.604    908.744  -3.136 0.001736 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 122.2 on 2232 degrees of freedom
Multiple R-squared:  0.3562,        Adjusted R-squared:  0.3545
F-statistic: 205.8 on 6 and 2232 DF,  p-value: < 2.2e-16
```

Figure 6 (page 12), which is made by the following code, shows the best quadratic approximation to the fitness landscape fit above by multiple regression together with the estimate from the aster model from Figure 5. It is made by the following code, first the prediction

```
> zzols <- predict(lout, newdata = data.frame(LOGLVS = xx, LOGSLA = yy,
+     STG1N = rep(thestg, length(xx))))
```

```
> plot(chamae2w$LOGLVS, chamae2w$LOGSLA, xlab = xlab, ylab = ylab, pch = ".")
> contour(x, y, zfit, add = TRUE)
> contour(x, y, zfit, levels = c(5, 10, 25), add = TRUE)
> zzols <- matrix(zzols, nrow = length(x))
> contour(x, y, zzols, add = TRUE, lty = "dotted")
```

Note that fitness is a positive quantity. Hence the negative contours in the best quadratic approximation are nonsense, although they are the inevitable result of approximating a surface that is not close to quadratic with a quadratic function. Note also that the best quadratic approximation has a saddle point and no maximum, whereas it appears that the actual fitness landscape does have a maximum, albeit near the edge of the distribution of phenotypes. Apparently, the saddle point is the result of the quadratic function trying to be nearly flat on the left hand side of the figure (a quadratic function cannot have an asymptote; the saddle point is the next best thing). A quadratic function cannot have both a saddle point and a maximum; it has to choose one or the other. Unfortunately, least squares makes the wrong choice from the biological point of view. It is more important to get the maximum right than the flat spot (where fitness is close to zero).

References

Andrews, D. F., Gnanadesikan, R. and Warner, J. L. (1971). Transformations of multivariate data. *Biometrics*, **27**, 825–840.

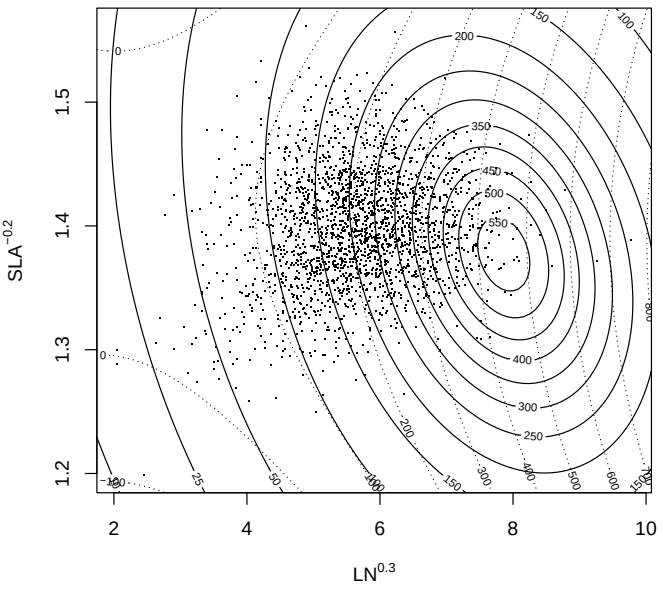

Figure 6: Scatterplot of phenotypic variables with contours of fitness landscape estimated by the aster model (solid) and the best quadratic approximation (dotted).

Box, G. E. P. and Cox, D. R. (1964). An analysis of transformations (with discussion). *Journal of the Royal Statistical Society, Series B*, **26**, 211–252.

Etterson, J. R. (2004) Evolutionary potential of *Chamaecrista fasciculata* in relation to climate change. I. Clinal patterns of selection along an environmental gradient in the great plains. *Evolution*, **58**, 1446–1458.

Etterson, J. R., and Shaw, R. G. (2001). Constraint to adaptive evolution in response to global warming. *Science*, **294**, 151–154.

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.

R Development Core Team (2006). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. `http://www.R-project.org`.

Riani, M. (2004). Robust multivariate transformations to normality: Constructed variables and likelihood ratio tests. *Statistical Methods & Applications*, **13**, 179–196.

Shaw, R. G., Geyer, C. J., Wagenius, S., Hangelbroek, H. H., and Etterson, J. R. (submitted). Unifying life history analysis for inference of fitness and population growth. `http://www.stat.umn.edu/geyer/aster/`

Shaw, R. G., Geyer, C. J., Wagenius, S., Hangelbroek, H. H., and Etterson, J. R. (2007). More supporting data analysis for "Unifying life history analysis for inference of fitness and population growth". University of Minnesota School of Statistics Technical Report No. 661 `http://www.stat.umn.edu/geyer/aster/`

Venables, W. N. and Ripley, B. D. (2002). *Modern Applied Statistics with S*, 4th ed. New York: Springer-Verlag.