Hypothesis Tests and Confidence Intervals Involving Fitness Landscapes fit by Aster Models

By
Charles J. Geyer and Ruth G. Shaw

Technical Report No. 674

School of Statistics
University of Minnesota

March 24, 2009

Abstract

This technical report explores some issues left open in Technical Reports 669 and 670 (Geyer and Shaw, 2008a,b): for fitness landscapes fit using an aster models, we propose hypothesis tests of whether the landscape has a maximum and confidence regions for the location of the maximum.

All analyses are done in R (R Development Core Team, 2008) using the `aster` contributed package described by Geyer, Wagenius and Shaw (2007) and Shaw, Geyer, Wagenius, Hangelbroek, and Etterson (2008). Furthermore, all analyses are done using the `Sweave` function in R, so this entire technical report and all of the analyses reported in it are exactly reproducible by anyone who has R with the `aster` package installed and the R noweb file specifying the document.

1 R Package Aster

We use R statistical computing environment (R Development Core Team, 2008) in our analysis. It is free software and can be obtained from `http://cran.r-project.org`. Precompiled binaries are available for Windows, Macintosh, and popular Linux distributions. We use the contributed package `aster`. If R has been installed, but this package has not yet been installed, do

```
install.packages("aster")
```

from the R command line (or do the equivalent using the GUI menus if on Apple Macintosh or Microsoft Windows). This may require root or administrator privileges.

Assuming the `aster` package has been installed, we load it

```
> library(aster)
```

The version of the package used to make this document is 0.7-7 (which is available on CRAN). The version of R used to make this document is 2.8.1.

This entire document and all of the calculations shown were made using the R command `Sweave` and hence are exactly reproducible by anyone who has R and the R noweb (RNW) file from which it was created. Both the RNW file and and the PDF document produced from it are available at `http://www.stat.umn.edu/geyer/aster`. For further details on the use of Sweave and R see Chapter 1 of the technical report by Shaw, et al. (2007) available at the same web site.

Not only can one exactly reproduce the results in the printable document, one can also modify the parameters of the simulation and get different results.

Finally, we set the seeds of the random number generator so that we obtain the same results every time. To get different results, obtain the RNW file, change this statement, and reprocess using `Sweave` and LaTeX.

```
> set.seed(42)
```

2 Data Structure

We use the data simulated in Technical Report 669 (Geyer and Shaw, 2008a, herein after TR 669), because this simulated data has features not present in any currently available real data yet shows the full possibilities of aster modeling.

Temporarily, we load the data from a file. We intend that these data will be incorporated in a future version of the aster package.

```
> load("sim.rda")
> ls()

[1] "beta.true"  "fam"        "ladata"     "mu.true"
[5] "phi.true"   "pred"       "redata"     "theta.true"
[9] "vars"
```

For a full description of the graphical structure of these data see Section 2.1 of TR 669. For a full description of the variables and their conditional distributions see Section 2.2 of TR 669. For a full description of the sum of fitness components that is deemed the best surrogate of fitness for these data see Section 2.3 of TR 669.

3 Hypothesis Tests about Maxima

3.1 Asymptotic

First, we consider asymptotic (large sample, approximate) tests based on the well known likelihood ratio test, which has asymptotic chi-square distribution.

We fit the same model fit in TR 669 in which the unconditional canonical parameter corresponding to best surrogate of fitness is a quadratic function of phenotype data. In short, fitness is quadratic on the canonical parameter scale.

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

We also fit the model in which fitness is linear on the canonical parameter scale.

```
> out5 <- aster(resp ~ varb + 0 + z1 + z2, pred, fam,
+     varb, id, root, data = redata)
```

Then we compare these two models using the conventional likelihood ratio test.

```
> anova(out5, out6)

Analysis of Deviance Table

Model 1: resp ~ varb + 0 + z1 + z2
Model 2: resp ~ varb + 0 + z1 + z2 + I(z1^2) + I(z1 * z2) + I(z2^2)
  Model Df Model Dev Df Deviance P(>|Chi|)
1       22    -84959
2       25    -84975  3       16  0.000957
```

The P-value calculated here is highly statistically significantly ($P = 9.6 \times 10^{-4}$). It is a correct asymptotic approximation to the P-value for comparing the linear and quadratic models. It says the quadratic model clearly fits better.

A linear model cannot have a stationary point. A quadratic model does have a stationary point. Hence this is also a test of whether the fitness landscape has a stationary point, which may be a maximum, a minimum, or a saddle point.

If we wish to turn this into a test for the presence of a maximum, we should make the alternative be that the model is quadratic and the the quadratic surface has a maximum. Since this requires the coefficients of both `I(z1^2)` and `I(z2^2)` to be negative, this restricts the alternative to 1/4 of the parameter space. Hence the appropriate P-value for this test is 1/4 of the P-value for the general quadratic alternative, that is, ($P = 2.4 \times 10^{-4}$).

This test, although seemingly liberal, dividing the P-value of the conventional test by 4, or, more generally, by 2^p where p is the number of phenotypic variables, is in fact (asymptotically) conservative. Our claim that the alternative is only 1/4 of the parameter space, or 2^{-p} of the parameter space for general p, is conservative, since it only takes account of the diagonal terms of the Hessian matrix of the quadratic function.

3.2 Parametric Bootstrap

Second, we consider tests based on the distribution under the null hypothesis determined by simulation. These are still large sample approximate in a weak sense in that we should simulate the distribution for the true unknown parameter vector $\boldsymbol{\beta}$ but cannot and must use our best approximation, which is the distribution for the parameter vector $\hat{\boldsymbol{\beta}}$ that is the MLE for the null hypothesis. Since this only makes sense when $\hat{\boldsymbol{\beta}}$ is close to $\boldsymbol{\beta}$, this procedure is only approximate. To remind everyone of this fact, we call the procedure a parametric bootstrap rather than a simulation test. However, this procedure is much less approximate than the procedure of the preceding section, because it does not use the chi-square approximation for the distribution of the test statistic but instead calculates its exact sampling distribution when $\hat{\boldsymbol{\beta}}$ is the true parameter value.

The test in the preceding section also does not fully account for the restriction that the Hessian matrix be negative definite. Thus an even more correct P-value can be obtained using a parametric bootstrap that goes like this.

```
> nsim <- 249
> theta.boot <- predict(out5, parm.type = "canonical",
+     model.type = "conditional")
> nind <- length(unique(redata$id))
> theta.boot <- matrix(theta.boot, nrow = nind)
> phi.boot <- out6$coef * 0
> phi.boot[1:length(out5$coef)] <- out5$coef
> pvalsim <- double(nsim)
> eigmaxsim <- double(nsim)
> save.time <- proc.time()
> for (i in 1:nsim) {
+     ystar <- raster(theta.boot, pred, fam, root = theta.boot^0)
+     redatastar <- redata
+     redatastar$resp <- as.vector(ystar)
+     out6star <- aster(resp ~ varb + 0 + z1 + z2 +
+         I(z1^2) + I(z1 * z2) + I(z2^2), pred, fam,
```

```
+         varb, id, root, parm = phi.boot, data = redatastar)
+     out5star <- aster(resp ~ varb + 0 + z1 + z2,
+         pred, fam, varb, id, root, parm = out5$coef,
+         data = redatastar)
+     Afoo <- matrix(NA, 2, 2)
+     Afoo[1, 1] <- out6star$coef["I(z1^2)"]
+     Afoo[2, 2] <- out6star$coef["I(z2^2)"]
+     Afoo[1, 2] <- out6star$coef["I(z1 * z2)"]/2
+     Afoo[2, 1] <- out6star$coef["I(z1 * z2)"]/2
+     pvalsim[i] <- anova(out5star, out6star)[2, 5]
+     eigmaxsim[i] <- max(eigen(Afoo, symmetric = TRUE,
+         only.values = TRUE)$values)
+ }
> elapsed.time <- proc.time() - save.time
> pval.obs <- anova(out5, out6)[2, 5]
> pvalsim.corr <- pvalsim
> pvalsim.corr[eigmaxsim > 0] <- 1
> mean(c(pvalsim.corr, pval.obs) <= pval.obs)

[1] 0.004
```

The parametric bootstrap P-value, here $P = 0.004$, cannot be lower than $1/(n+1)$, where n is the number of simulations, here `nsim` $= 249$. We have gotten the lowest bootstrap P-value we could have with this number of simulations, which took 24 minutes and 52.1 seconds.

Hence there is little point in using the parametric bootstrap here where the asymptotic P-value ($P = 2.4 \times 10^{-4}$) is so small. If the asymptotic P-value were equivocal, somewhere in the vicinity of 0.05, then there would be much more reason to calculate a parametric bootstrap P-value, and the code above shows how to do it right.

We can see that the correction of dividing the conventional P-value by 2^p is conservative here. The fraction of the sample in which the matrix A is negative definite is 0.145, a good deal less than $2^{-p} = 0.25$.

4 Confidence Regions about Maxima

Now we consider the MLE of the location of the maximum, which is calculated in TR 669 as follows

```
> Afoo <- matrix(NA, 2, 2)
> Afoo[1, 1] <- out6$coef["I(z1^2)"]
> Afoo[2, 2] <- out6$coef["I(z2^2)"]
> Afoo[1, 2] <- out6$coef["I(z1 * z2)"]/2
> Afoo[2, 1] <- out6$coef["I(z1 * z2)"]/2
> bfoo <- rep(NA, 2)
> bfoo[1] <- out6$coef["z1"]
> bfoo[2] <- out6$coef["z2"]
```

```
> cfoo <- solve(-2 * Afoo, bfoo)
> cfoo

[1] 3.335738 1.626314
```

The explanation for this is that the estimated regression function, mapped to the natural parameter scale, is

$$g(\mathbf{z}) = c + \mathbf{b}^T \mathbf{z} + \mathbf{z}^T \mathbf{A} \mathbf{z}$$

where c is an arbitrary constant, $\mathbf{b}$ is the R vector `bfoo` above and $\mathbf{A}$ is the R matrix `Afoo` above. The first derivative vector is

$$\nabla g(\mathbf{z}) = \mathbf{b}^T + 2 \mathbf{z}^T \mathbf{A}$$

and setting this equal to zero and solving for $\mathbf{z}$ gives

$$\mathbf{z} = -\tfrac{1}{2} \mathbf{A}^{-1} \mathbf{b}$$

For future reference, we also compute the maximum of the simulation truth fitness landscape.

```
> Abar <- matrix(NA, 2, 2)
> Abar[1, 1] <- beta.true["I(z1^2)"]
> Abar[2, 2] <- beta.true["I(z2^2)"]
> Abar[1, 2] <- beta.true["I(z1 * z2)"]/2
> Abar[2, 1] <- beta.true["I(z1 * z2)"]/2
> bbar <- rep(NA, 2)
> bbar[1] <- beta.true["z1"]
> bbar[2] <- beta.true["z2"]
> cbar <- solve(-2 * Abar, bbar)
```

In order to apply the multivariable delta method, we need to differentiate the function of the parameter β that gives the maximum,

$$h(\boldsymbol{\beta}) = -\tfrac{1}{2} \mathbf{A}(\boldsymbol{\beta})^{-1} \mathbf{b}(\boldsymbol{\beta}),$$

where we have now written the matrix $\mathbf{A}$ and the vector $\mathbf{b}$ as functions of the regression coefficient vector $\boldsymbol{\beta}$, which they are, each component of $\mathbf{A}$ and each component of $\mathbf{b}$ being a component of $\boldsymbol{\beta}$. The partial derivatives are

$$\frac{\partial h(\boldsymbol{\beta})}{\partial \beta_i} = \tfrac{1}{2} \mathbf{A}(\boldsymbol{\beta})^{-1} \frac{\partial \mathbf{A}(\beta)^{-1}}{\partial \beta_i} \mathbf{A}(\boldsymbol{\beta})^{-1} \mathbf{b}(\boldsymbol{\beta}) - \tfrac{1}{2} \mathbf{A}(\boldsymbol{\beta})^{-1} \frac{\partial \mathbf{b}(\boldsymbol{\beta})}{\partial \beta_i}$$

The asymptotic variance of the components of $\boldsymbol{\beta}$ is the submatrix of the inverse Fisher information matrix corresponding to the components of $\boldsymbol{\beta}$ that enter into $\mathbf{A}(\boldsymbol{\beta})$ and $\mathbf{b}(\boldsymbol{\beta})$

```
> beta.sub.names <- c("I(z1^2)", "I(z2^2)", "I(z1 * z2)",
+     "z1", "z2")
> beta.sub.idx <- match(beta.sub.names, names(out6$coef))
> asymp.var <- solve(out6$fisher)
> asymp.var <- asymp.var[beta.sub.idx, ]
> asymp.var <- asymp.var[, beta.sub.idx]
```

The derivative matrix is set up as follows

```
> jack <- matrix(NA, 2, 5)
> jack[, 1] <- (1/2) * solve(Afoo) %*% matrix(c(1,
+     0, 0, 0), 2, 2) %*% solve(Afoo) %*% cbind(bfoo)
> jack[, 2] <- (1/2) * solve(Afoo) %*% matrix(c(0,
+     0, 0, 1), 2, 2) %*% solve(Afoo) %*% cbind(bfoo)
> jack[, 3] <- (1/2) * solve(Afoo) %*% matrix(c(0,
+     1, 1, 0), 2, 2) %*% solve(Afoo) %*% cbind(bfoo)
> jack[, 4] <- (-(1/2) * solve(Afoo) %*% matrix(c(1,
+     0), 2, 1))
> jack[, 5] <- (-(1/2) * solve(Afoo) %*% matrix(c(0,
+     1), 2, 1))
```

Finally, we finish applying the delta method

```
> asymp.var <- jack %*% asymp.var %*% t(jack)
> asymp.var

         [,1]     [,2]
[1,] 2.739692 3.997176
[2,] 3.997176 7.401151
```

So now we plot a confidence region for the maximum based on the delta method calculation above. The following R statements make Figure 1 (page 8)

```
> par(mar = c(2, 2, 1, 1) + 0.1)
> plot(ladata$z1, ladata$z2, xlab = "", ylab = "",
+     pch = 20, axes = FALSE, xlim = range(ladata$z1,
+         cfoo[1]), ylim = range(ladata$z2, cfoo[2]))
> title(xlab = "z1", line = 1)
> title(ylab = "z2", line = 1)
> box()
> z1 <- cos(seq(0, 2 * pi, length = 101))
> z2 <- sin(seq(0, 2 * pi, length = 101))
> z <- rbind(z1, z2)
> points(cfoo[1], cfoo[2], col = "blue", pch = 19)
> fred <- eigen(asymp.var)
> sally <- fred$vectors %*% diag(sqrt(fred$values)) %*%
+     t(fred$vectors)
> points(cbar[1], cbar[2], col = "green3", pch = 19)
> jane <- qchisq(0.5, 2) * sally %*% z
> lines(cfoo[1] + jane[1, ], cfoo[2] + jane[2, ], col = "blue",
+     lwd = 2)
> jane <- qchisq(0.75, 2) * sally %*% z
> lines(cfoo[1] + jane[1, ], cfoo[2] + jane[2, ], col = "blue",
+     lwd = 2, lty = "dotted")
```

```
> jane <- qchisq(0.9, 2) * sally %*% z
> lines(cfoo[1] + jane[1, ], cfoo[2] + jane[2, ], col = "blue",
+     lwd = 2, lty = "dashed")
```

The confidence regions are huge, the 90% confidence region containing most of the range of the data. One would need much larger sample sizes than the 500 used here to get precise confidence regions.

One might think we should have a section on how to calculate a confidence region based on the parametric bootstrap rather than asymptotic normality, and this would be expected for a confidence interval. However, it is an open research question how best to make a confidence region in this situation. The elliptical (large sample, approximate, delta method) confidence regions shown in Figure 1 get their shape from the asymptotic bivariate normal distribution of the two-dimensional vector (location of the maximum) being estimated. A bivariate normal distribution has elliptical contours of its probability density function, hence elliptical confidence regions make sense. If we drop the "assumption" of normality (not really an assumption but an asymptotic approximation), then there is no reason to make elliptical confidence regions. In fact, the main point of parametric bootstrap confidence intervals is to drop the "assumption" of normality and use intervals that are not centered at the MLE and reflect the skewness of the simulation distribution of the estimates. So in order to do a parametric bootstrap correctly in this situation, we should also use a non-elliptical confidence region that reflects the non-normality of the simulation distribution of the estimates. But how? That is the open research question. All of the bootstrap literature known to us is about confidence intervals, not about confidence regions.

References

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

Geyer, C. J. and Shaw, R. G. (2008b) Commentary on Lande-Arnold Analysis. University of Minnesota School of Statistics Technical Report No. 670. `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.

R Development Core Team (2008). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. `http://www.R-project.org`.

Shaw, R. G., Geyer, C. J., Wagenius, S., Hangelbroek, H. H., and Etterson, J. R. (2007). Supporting data analysis for "Unifying life history analysis for inference of fitness and population growth". University of Minnesota School of Statistics Technical Report No. 658. `http://www.stat.umn.edu/geyer/aster/`

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.

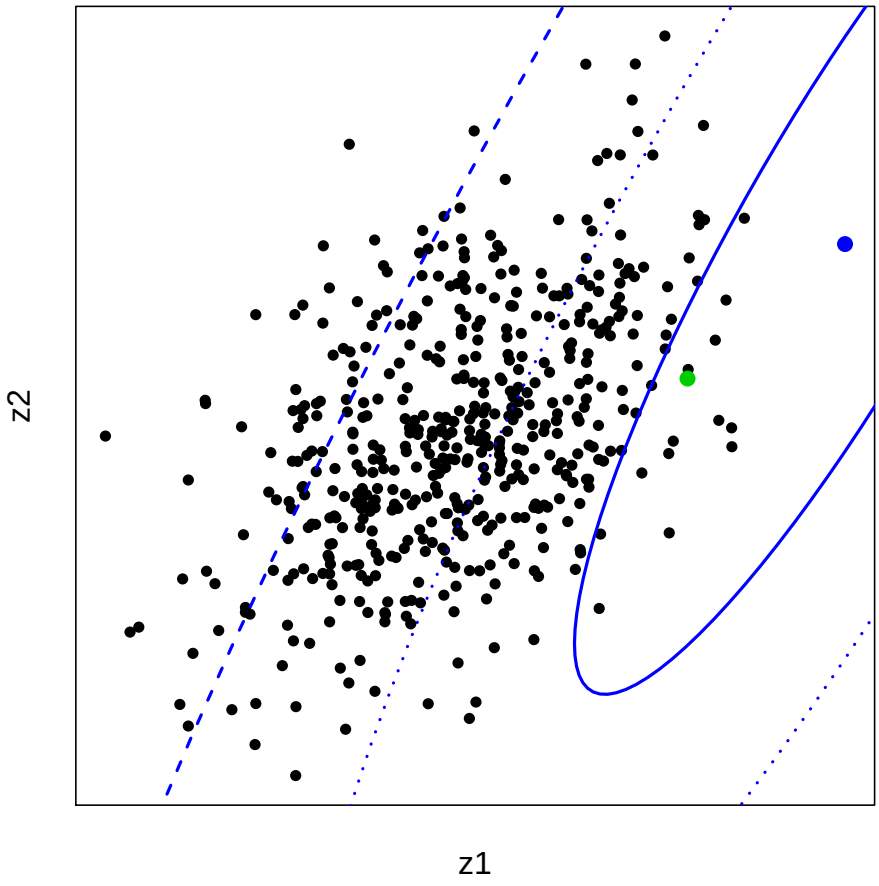

Figure 1: Scatterplot of `z1` versus `z2` with location of MLE of maximum of the fitness landscape (blue), boundary of asymptotic 50% confidence region for the maximum (solid blue), boundary of the asymptotic 75% confidence region (dotted blue), and boundary of the asymptotic 90% confidence region (dashed blue). Also shown is the simulation truth maximum (green).