

# Statistical estimation of ovule fates parameters

Matthew B. Routley

December 7, 2005

## Model 1

Model 1 is a general linear model with an identity link function with binomial variance. The seed-ovule ratio is the response variable ( $\mu$ ) and the proportion of cross-fertilized ovules is the predictor variable ( $x$ ).

$$g(\mu) = \beta_{1,0} + \beta_{1,1}x_i$$

```
> response <- cbind(seeds, ovules)
> model.1 <- glm(response ~ proportionCrossPollen, family = quasi(var = "mu(1-mu)",
+   link = "identity"))
```

## Model 2

Model 2 is also a GLM with an identity link function with binomial variance. In addition to the same response and predictor variables as Model 1, Model 2 includes the parameter  $\alpha$  which indicates the breakpoint in the response of the seed-ovule ratio to changes in the proportion of cross-pollen (Toms and Lesperance, 2003).

$$g(\mu) = \begin{cases} \beta_{2,0} + \beta_{2,1}x_i, & \text{if } x_i \leq \alpha \\ \beta_{2,0} + \beta_{2,1}x_i + \beta_{2,2}(x_i - \alpha), & \text{if } x_i > \alpha \end{cases}$$

```
> model.2 <- glm(response ~ bs(proportionCrossPollen, knots = alpha,
+   degree = 1), family = quasi(var = "mu(1-mu)", link = "identity"))
```

To choose the  $\alpha$  value to substitute into Model 2, we evaluate the expression above across a range of  $\alpha$  values and choose the value that provides the lowest model deviance.

```
> alpha.scale <- 0.1
> deviances <- unlist(lapply(seq(0 + alpha.scale, 1 - alpha.scale, alpha.scale),
+   function(alpha) glm(response ~ bs(proportionCrossPollen, knots = alpha,
+     degree = 1), family = quasi(var = "mu(1-mu)", link = "identity"))$deviance))
> best.alpha <- which.min(deviances) * alpha.scale + alpha.scale
```

## Parameter estimates

Before estimating the ovule fates parameters, the appropriate model must be chosen (Figure 1). If  $\beta_{1,1}$  is not significantly different from 0, then  $m < g_s < g_x$  (Figure 2a) and only  $m$  can be estimated. The estimate of  $m$  is the intercept of Model 1.

```
> if (summary(model.1)$coefficients[2, 4] > 0.05) {
+   m <- summary(model.1)$coefficients[1, 1]
+ }
```

If  $\beta_{1,1}$  is significantly different from 0, we compare the AICs of Model 1 and Model 2 to determine which model is the better fit to the data. In either case,  $g_s$  is estimated as  $\beta_{i,0}$  and  $g_x$  as  $\beta_{i,1} + \beta_{i,0}$ .

```
> gs <- the.model$coefficients[1]
> gx <- the.model$coefficients[2] + the.model$coefficients[1]
```

If Model 2 has the lowest AIC, then  $m$  is estimated as the mean seed-ovule ratio complete cross-pollination. Alternatively, if Model 1 has the lowest AIC, then  $m$  is 1.

```
> if (model.2$aic < model.1$aic) {
+   m <- mean(seeds[proportionCrossPollen == 1])/ovules
+ } else {
+   m <- 1
+ }
```

## Complete R functions

The R code described above is consolidated into the function `OFModels`. `OFModels` takes the proportion of cross-pollen, seed set, and ovule production as arguments and returns a named list of estimates for  $g_s$ ,  $g_x$ ,  $m$ , and  $\alpha$ .

```
> OFModels <- function(x, y, ovules) {
+   library(splines)
+   estimates <- c(NA, NA, NA, NA)
+   response <- y/ovules
+   parameterEstimates <- function(the.model) {
+     as.real(c(the.model$coefficients[1], the.model$coefficients[2] +
+       the.model$coefficients[1]))
+   }
+   family <- quasi(var = "mu(1-mu)", link = "identity")
+   model.1 <- glm(response ~ x, family = family)
+   if (summary(model.1)$coefficients[2, 4] > 0.05) {
+     estimates <- c(NA, NA, summary(model.1)$coefficients[1, 1],
+       NA)
+   }
+ }
```

```

+   }
+   else {
+     alpha.scale <- 0.1
+     deviances <- unlist(lapply(seq(0 + alpha.scale, 1 - alpha.scale,
+       alpha.scale), function(alpha) glm(response ~ bs(x, knots = alpha,
+       degree = 1), family = family)$deviance))
+     best.alpha <- which.min(deviances) * alpha.scale + alpha.scale
+     model.2 <- glm(response ~ bs(x, knots = best.alpha, degree = 1),
+       family = family)
+     if (model.2$deviance < model.1$deviance) {
+       condition <- x <= best.alpha
+       model.2.truncated <- glm(response[condition] ~ x[condition],
+         family = family)
+       estimates <- c(parameterEstimates(model.2.truncated), mean(y[x ==
+         1])/ovules, best.alpha)
+     }
+     else {
+       estimates <- c(parameterEstimates(model.1), 1, NA)
+     }
+   }
+   round(c(gs = estimates[1], gx = estimates[2], m = estimates[3],
+     alpha = estimates[4]), 3)
+ }

```

## Simulated data

Two functions are available for simulating data relevant to ovule fates: `OFSimulateData` and `OFPlot`. `OFSimulateData` creates a dataset containing replicate seed set data for given values of  $g_s$ ,  $g_x$ , and  $m$ . Random noise is taken from a normal distribution with a mean of 0 and specified standard deviation and added to the probability of an ovule becoming a seed. `OFPlot` takes the data generated from `OFSimulateData` and creates a plot of the simulated data. This function could also be used to generate plots of experimentally derived seed-ovule data.

```

> OFSimulateData <- function(gs, gx, m, sdev) {
+   ovules <- 100
+   parameters <- list(gs = gs, gx = gx, m = m)
+   replicates <- 3
+   if (gx > m & gs < m) {
+     parameters <- c(parameters, list(alpha = (m - gs)/(gx - gs)))
+   }
+   stepSize <- 0.1
+   crossPollen <- seq(0, 1, stepSize)
+   grp <- factor((1:replicates)[rep(1:replicates, each = length(crossPollen))])

```

```

+   crossPollen <- seq(0, 1, stepSize)
+   seeds <- rep((crossPollen * ovules * gx + (1 - crossPollen) * ovules *
+     gs), replicates)
+   conditions <- (seeds > m * ovules)
+   seeds[conditions] <- m * ovules
+   seeds <- seeds + ovules * rnorm(length(crossPollen) * replicates,
+     mean = 0, sd = sdev)
+   seeds[seeds > ovules] <- ovules
+   seeds[seeds < 0] <- 0
+   seeds <- round(seeds, 0)
+   data <- as.data.frame(cbind(grp, rep(crossPollen, replicates),
+     seeds))
+   names(data) <- c("replicates", "crossPollen", "seeds")
+   OFPlot(data$crossPollen, data$seeds, data$replicates, ovules)
+   c(parameters, data)
+ }
> OFPlot <- function(x, y, class, ovules) {
+   library(splines)
+   stepSize <- (max(x) - min(x))/(length(x)/length(levels(as.factor(class))) -
+     1)
+   symbols <- 21
+   fill <- c("white")
+   plot(c(0, 1), c(ovules, 0), type = "n", xlab = "Proportion cross pollen",
+     ylab = "Seed production", las = 1)
+   points(x, y, pch = symbols[1], bg = fill[class])
+   sequence <- seq(0, 1, stepSize)
+   loe.predict <- predict(loess(y ~ x), data.frame(x = sequence),
+     se = TRUE)
+   lines(sequence, loe.predict$fit)
+   lines(sequence, loe.predict$fit + loe.predict$se, lty = "dashed")
+   lines(sequence, loe.predict$fit - loe.predict$se, lty = "dashed")
+ }

```

## References

TOMS, J. D. AND LESPERANCE, M. L. 2003. Piecewise regression: a tool for identifying ecological thresholds. *Ecology* 84:2034–2041.

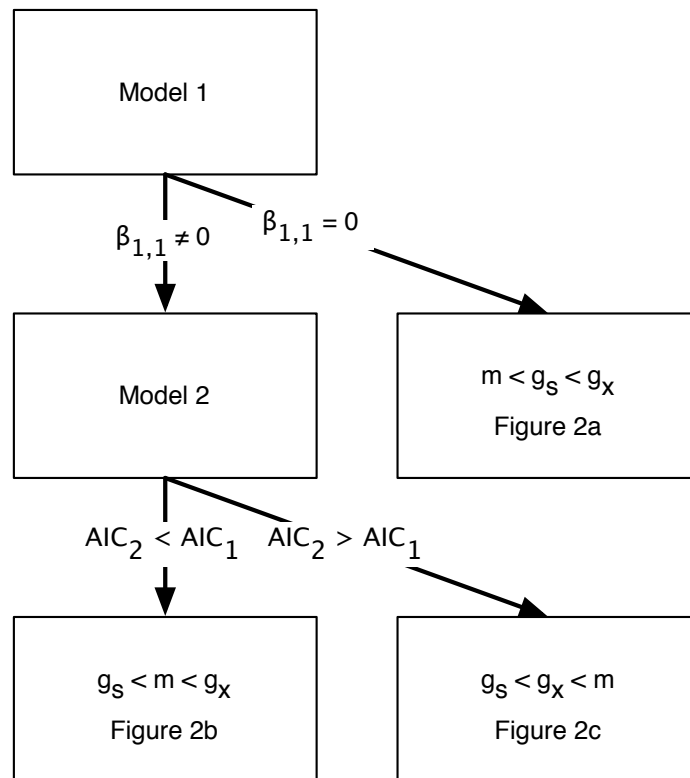


Figure 1: A schematic of the decisions required to estimate parameters of the ovule fates model from empirical data. Examples of the classes of data are illustrated in Figure 2.

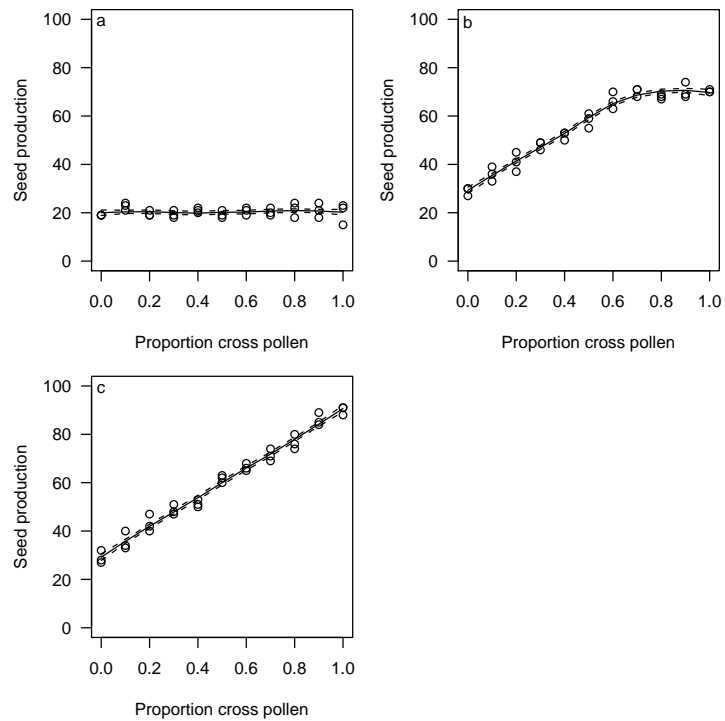


Figure 2: Simulated data created with the `OFSimulateData` and `OFPlot` functions. In each panel 100 ovules are available,  $g_s = 0.3$ ,  $g_x = 0.9$ , and  $\sigma = 0.025$ . a)  $m = 0.2$ , b)  $m = 0.7$ , and c)  $m = 1.0$ .