Supporting Data Analysis for "An Integrated Analysis of Phenotypic Selection on Insect Body Size and Development Time"

By
Daniel J. Eck, Ruth G. Shaw,
Charles J. Geyer, and Joel G. Kingsolver

Technical Report No. 698

School of Statistics
University of Minnesota

May 13, 2015
Revised July 2, 2015

Abstract

This technical report (TR) gives details of the data analysis backing up a paper having the same authors as this TR and having the title that is quoted in the title of this TR. It uses the R package `knitr` to process its source file (Rnw file) containing LaTeX and R, so that all R that appears in this document is not cut-and-pasted but actually run and actually produces the results that it appears to produce. This document reproduces all tables and figures from the paper and hence the entire data analysis reported in the paper. The data for the analysis are in the R object `hornworm` in the R package `aster2`, which is a CRAN package. An appendix of this technical report shows how this R object of class `"asterdata"` was created from the raw data, a comma separated values (CSV) file.

1 Introduction

In the paper that this technical report supplements, we describe a new generation of aster models (Geyer, Wagenius, and Shaw, 2007) that incorporate age of first reproduction in the model for fitness. Aster models are implemented in the R statistical computing language (R Development Core Team, 2014) via the CRAN package `aster` (Geyer, 2014, originally available in 2005). This package does not allow accounting for variation in generation time in the expression of fitness. To allow for this component of fitness, the CRAN package `aster2` (Geyer, 2015) has the added capability of specifying "dependence groups," such that life history stages, e. g., insect larval instars, pupa, and adult, and, in particular, variation in the age at which individuals reach these stages, are included in the model for fitness. Here we use `aster2` to take these stages into account in conducting selection analyses for a dataset from a previous study of phenotypic selection on body size and age at different developmental stages in *Manduca sexta* (Kingsolver, et al., 2012). That study estimated selection on each fitness component separately, and therefore could not quantify how selection operates over the lifecycle, nor identify potential trade-offs in selection among fitness components. Our aster analyses of these data provide an integrated view of phenotypic selection on size and age across development and fitness components in this study system. We discuss how these methods can be applied to selection analyses in other systems.

2 R

This document was processed with R, version 4.3.2, using the CRAN packages `aster2`, version 0.3, and `knitr`, version 1.45.

```
library(aster2)

## Loading required package:  Matrix
## This is beta software.
## Unless you need to do aster models with dependence groups,
##     use package "aster" instead.
## See help(aster2-package) for differences from package "aster"
##     and examples.
```

3 Data

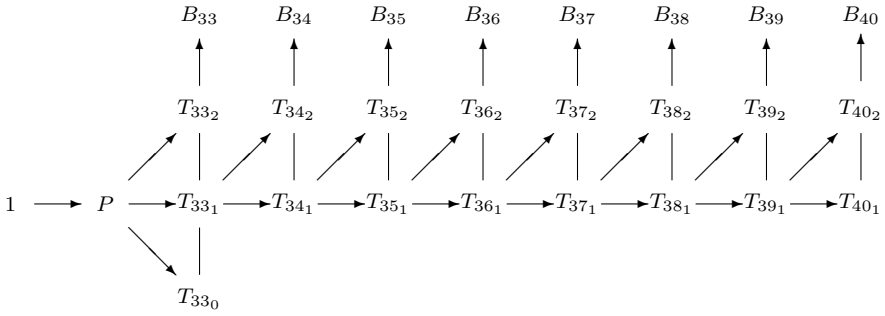
Data are in the R object `hornworm` in the `aster2` package. We are also going to use `data` as a synonym of this data object (because that was what we called it when we started this analysis, the name `hornworm` came later).

```
data(hornworm)
data <- hornworm
```

We show how this data set, an R object of class `"asterdata"`, was created in the appendix.

4 Graph

The aster graph for females in these data is shown below.



The aster graph for males and unknown (sex not determined because died before pupation) is the same except there are no B_x nodes.

Arrows go from predecessor nodes to successor nodes. Lines (that are not arrows) link dependence groups. Nodes are labeled by their associated variables. P node is pupation indicator, T nodes are survival and eclosion indicators, B nodes are ovariole counts. Subscripts indicate age (in days), subsubscripts indicate variables in the same dependence group (0 = death, 1 = surviving but pre-eclosion, 2 = eclosion at this time). All deaths before reproduction were modeled as occurring on day 33 because the day of death for individuals who died after pupation but before eclosion was not recorded.

As in all aster graphs, predecessor equals zero implies successor equals zero. If the variable at any node is zero, then the variables at all downstream nodes are also zero (all nodes one gets to by following arrows from a node where the variable is zero). Thus at most one variable of any T_x dependence group is nonzero, and the nonzero T_{x_y} variables indicate the path the life history for an individual takes. Similarly, at most one B_x variable is nonzero (because at most one T_{x_2} variable is nonzero), the one where x is the day the individual actually eclosed (if it did).

Our measure of Darwinian fitness for this graph is at first (Section 7 below) total ovariole count, the sum of all the B_x nodes. Since at most one of these nodes is nonzero (as explained above), this is just the number of ovarioles the individual had if it survived to eclosion (and zero otherwise). Later (Section 9 below) our measure of Darwinian fitness is a weighted sum of ovariole counts, weighted according to age and population growth rate, with weights given by (7). This is fitness adjusted for population growth rate.

5 A Digression on Aster Model Theory

Because a referee of the paper this technical report accompanies asked for it, we give a brief explanation of aster models and their theory. This section can be skipped by readers who want to get on to the analysis.

The full aster graph has a node for every measured component of fitness for every individual and also a constant node for every individual. The figure on p. 2 is only the part of the full graph for one female individual. The node marked 1 indicates that this part of the graph is for one individual (in some data sets data on multiple individuals is lumped together and then the constant node says how many individuals that is).

The *response vector* y of an aster model contains every measured compo-

nent of fitness for every individual, one component for every node in the full graph except for constant nodes. Every component of y belongs to exactly one dependence group G. Some components are in dependence groups by themselves. For example, in the figure on p. 2 the nodes T_{33_0}, T_{33_1}, and T_{33_2} form a dependence group of size 3, the nodes T_{34_0} and T_{34_1} form a dependence group of size 2, the node B_{35} forms a dependence group of size 1 (a group by itself), and the node P is another group by itself. Each dependence group G has exactly one predecessor node $q(G)$. In the figure on p. 2 the predecessor of $\{T_{33_0}, T_{33_1}, T_{33_2}\}$ is P, the predecessor of $\{T_{34_0}, T_{34_1}\}$ is T_{33_1}, the predecessor of B_{35} is T_{35_2}, and the predecessor of P is the constant node.

The *saturated aster model* has one parameter for every component of the response vector. The part of the response vector y_G containing the components in dependence group G is exponential family with sample size $q(G)$ and hence makes contribution to the log likelihood

$$y_G^T \theta_G - y_{q(G)} c_G(\theta_G)$$

(Geyer, et al., 2007, equation (2), and Geyer, 2010, Section 2.4). The vector θ whose parts are θ_G is the *conditional canonical parameter vector*. The function c_G is the *cumulant function* for dependence group G. The whole aster model is also an exponential family (not just the conditional distributions of its dependence groups). The whole log likelihood is the sum over all dependence groups for all individuals

$$l(\theta) = \sum_{G \in \mathcal{G}} \left[y_G^T \theta_G - y_{q(G)} c_G(\theta_G) \right]$$

where $\mathcal{G}$ is the family of all dependence groups. To see that this has exponential family form, we rewrite it as

$$l(\theta) = \sum_{j \in J} y_j \left[\theta_j - \sum_{\substack{G \in \mathcal{G} \\ q(G) = j}} c_G(\theta_G) \right] - \sum_{\substack{G \in \mathcal{G} \\ j \notin J}} y_{q(G)} c_G(\theta_G)$$

where $J = \bigcup \mathcal{G}$ the set of all nonconstant nodes of the full graph, and match it to the exponential family form

$$l(\varphi) = y^T \varphi - c(\varphi)$$

seeing that this works with

$$\varphi_j = \theta_j - \sum_{\substack{G \in \mathcal{G} \\ q(G) = j}} c_G(\theta_G) \tag{1}$$

and

$$c(\varphi) = \sum_{\substack{G \in \mathcal{G} \\ j \notin J}} y_{q(G)} c_G(\theta_G) \tag{2}$$

The vector φ having components φ_j is the *unconditional canonical parameter vector*, and the function c is the cumulant function of the saturated model. Equation (1) is called the *aster transform*. It defines an invertible infinitely differentiable change of parameter (Geyer, et al., 2007, Section 2.3). It may seem odd that (2) purports to define a function of φ by giving it as a function of θ, but this is valid because of the invertibility of the aster transform.

Neither θ nor φ has a simple connection with the components of y. For that we want mean value parameters. The *unconditional mean value parameter* is the vector μ having components

$$\mu_j = E(y_j)$$

and the *conditional canonical parameter* is the vector ξ having components

$$\xi_j = E(y_j | y_{p(j)} = 1)$$

where $p(j) = q(G)$ for the unique G that contains j, so $y_{p(j)}$ is the predecessor of y_j like $y_{q(G)}$ is the predecessor of y_G. It is a fundamental property of aster models that means multiply

$$\mu_j = \xi_j \mu_{p(j)} \tag{3}$$

(Geyer, et al., 2007, eq. (12)), so applying (3) recursively we have

$$\begin{aligned}
\mu_j &= \xi_j \xi_{p(j)} \mu_{p(p(j))} \\
\mu_j &= \xi_j \xi_{p(j)} \xi_{p(p(j))} \mu_{p(p(p(j)))} \\
\mu_j &= \xi_j \xi_{p(j)} \xi_{p(p(j))} \xi_{p(p(p(j)))} \mu_{p(p(p(p(j))))}
\end{aligned}$$

and so forth going all the way back to when the μ on the right-hand side is at a constant node and hence is just the constant (the expectation of a constant is that constant). In short, unconditional means are products of conditional means.

We now have four parameterizations of the saturated model θ, φ, ξ, and μ. All have their purposes. Any may be needed in some application. In most aster models all of the transformations between any of these parameters are invertible and infinitely differentiable and can be calculated by the computer. For models having multinomial dependence groups like our hornworm data,

the θ and φ parameterizations are not identifiable because each part of the response vector y_G for a multinomial dependence group satisfies the linear constraint

$$\sum_{j \in G} y_j = y_{q(G)}$$

and any linear constraint satisfied by the canonical statistic of an exponential family causes nonidentifiability (Geyer, 2009, Theorem 1 and the following discussion). This nonidentifiability is broken by fixing one component of θ_G for each multinomial dependence group G if we are using the θ parameterization and similarly for φ. With that proviso all of the parameter transformations are invertible and infinitely differentiable.

Of course, saturated models are uninteresting for data analysis because they have too many parameters. They are just the largest model within which we seek parsimonious submodels. Modeling mean values as linear functions of means, like linear models (LM) do, is nonsense because of the constraints on means (conditional means for both Bernoulli and multinomial are between zero and one, those for zero-truncated Poisson are nonnegative). Even generalized linear models (GLM) don't do that. Instead GLM that are exponential families (logistic regression and Poisson regression with log link) model canonical parameters linearly. Aster models do the same. The question is which canonical parameter vector θ or φ? The best answer is φ because that is the parameter that makes the whole aster model an exponential family. An *unconditional canonical affine submodel* has

$$\varphi = a + M\beta$$

where a is a known vector and M a known matrix, called the *offset vector* and *model matrix* in the terminology of the R functions `lm` which fits LM and `glm` which fits GLM and of the R package `aster2` (the R package `aster` says "origin" instead of "offset"), and where β is the submodel parameter vector. This makes the submodel a full exponential family. The submodel log likelihood is

$$l(\beta) = y^T(a + M\beta) - c(a + M\beta) = y^T a + y^T M\beta - c(a + M\beta)$$

and the term not containing the parameter β can be dropped giving

$$l(\beta) = y^T M\beta - c(a + M\beta) = (M^T y)^T \beta - c_{\text{sub}}(\beta)$$

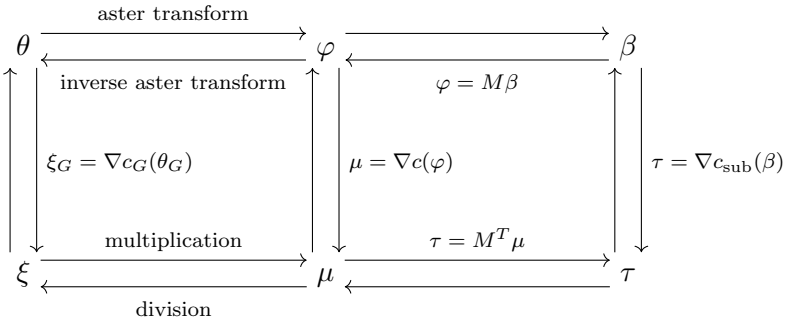
and this shows we have another exponential family with canonical statistic $M^T y$, canonical parameter β and cumulant function c_{sub}. The submodel

mean value parameter is the expectation of the canonical statistic

$$\tau = E(M^T y) = M^T \mu$$

With the proviso about fixing one component of φ or θ in each multinomial dependence group and with the additional proviso that M is full rank, we have six identifiable parameterizations θ, φ, ξ, μ, β, and τ and all of the transformations between them are invertible and infinitely differentiable.

The following diagram shows these parameters and some of their transformations.



As the diagram shows, transformations from canonical to mean value parameters are given by differentiating cumulant functions. The inverses of these transformations have no closed form expression but can be done by solving optimization problems, they are essentially equivalent to doing maximum likelihood.

Maximum likelihood estimation for the submodel differentiates the submodel log likelihood

$$\nabla l(\beta) = M^T y - \nabla c_{\text{sub}}(\beta) = M^T y - M^T E_\beta(y)$$

sets the derivative equal to zero and solves for β. Thus the maximum likelihood estimate (MLE) for β satisfies

$$M^T y = \nabla c_{\text{sub}}(\hat{\beta}) = M^T E_{\hat{\beta}}(y)$$

By invariance of maximum likelihood the MLE for τ is what the MLE for β maps to under the parameter transformation

$$\hat{\tau} = \nabla c_{\text{sub}}(\hat{\beta}) = M^T y$$

Everything in this paragraph holds for every full exponential family (nothing special about aster models here). MLE for them satisfy the "observed equals

expected" property (the MLE of the mean value parameter is the observed value of the canonical statistic, in this case $\hat{\tau} = M^T y$).

Exponential families give aster unconditional canonical affine submodels many more good properties: the multivariate monotonicity property of maps from canonical to mean value parameters (Geyer, 2010, sec. 2.9 and Shaw and Geyer, 2010, appendix), the sufficient dimension reduction property (all MLE are sufficient statistics, Geyer, 2010, sec. 2.10), and the maximum entropy property (Geyer, 2010, sec. 2.12). In particular, the multivariate monotonicity property of the maps $\varphi \longleftrightarrow \mu$ enables much important theory (Shaw and Geyer, 2010, appendix). All of these properties and more (Geyer, 2010, discussion) make unconditional canonical affine submodels by far the most useful.

Conditional canonical affine submodels model θ affinely

$$\theta = a + M\beta$$

These are not full exponential families. They are, of course curved exponential families because they are smooth submodels of the saturated model. But that doesn't give them any of the desirable properties of the unconditional canonical affine submodels. Generally, conditional submodels are a lot less parsimonious than unconditional submodels (Geyer, et al., 2007; Geyer, 2010). As far as we know there are only two published examples of conditional submodels: Example 1 in Shaw, et al. (2008), which arguably could have been done with an unconditional submodel, and Shaw, et al. (2015), which had to use a conditional submodel because of time-dependent covariates (the first essential use of such models). We ignore conditional submodels for the rest of this technical report because the R package `aster2` currently has no way to fit them.

6 Model Fitting

Fitting an aster model using the `aster2` package is not as easy as using the `aster` package. The `aster2` package has no model fitting function that takes a formula and fits a model, like the `aster` function in the `aster` package. Nor does it have generic `summary`, `anova`, and `predict` functions that support doing hypothesis tests and confidence intervals. The `aster2` package is the only way to do aster models with dependence groups. Currently, for everything else, use the `aster` package.

The `aster2` package has parameter transformation functions that do all transformations between all parameterizations of aster models shown in

the diagram on p. 7. In being able to do all of these transformations, the `aster2` package is superior to the `aster` package, which only does some of them (using the functions `predict` and `astertransform`).

But it does require more work to do maximum likelihood using the `aster2` package than the `aster` package. The observed equals expected property applied to aster models says that the MLE of the submodel mean value parameter $\hat{\tau}$ is equal to the submodel sufficient statistic $M^T y$. Then by invariance of maximum likelihood, the MLE of the submodel canonical parameter $\hat{\beta}$ is the transformation of $\hat{\tau}$. Thus maximum likelihood in the `aster2` package is done as follows.

1. Make the model matrix M. The R function `model.matrix` can be used to do this.

2. Make the submodel sufficient statistic $M^T y = \hat{\tau}$.

3. Transform $\hat{\tau}$ to $\hat{\beta}$ using the R function `transformUnconditional`.

Then, the MLE having been obtained, there is no support for hypothesis tests or confidence intervals. But the `aster2` package can calculate Fisher information for any of these parameters. The Fisher information matrix for β is

$$I(\beta) = \nabla^2 c_{\text{sub}}(\beta) = \nabla h(\beta)$$

where h is the transformation $\beta \longrightarrow \tau$. The Fisher information matrix for τ is

$$I(\tau) = \left[\nabla^2 c_{\text{sub}}(\beta)\right]^{-1} = \nabla h^{-1}(\tau)$$

And the Jacobian matrices $\nabla h(\beta)$ and $\nabla h^{-1}(\tau)$ are computed by the R function `jacobian`.

Then from Fisher information one can compute various hypothesis tests and confidence intervals. In this document we only do Rao tests. The `aster2` package does not have a function that calculates log likelihoods (in the `aster` package both R functions `aster` and `mlogl` do this), so we have no easy way to calculate likelihood ratios and likelihood ratio test statistics. That is why we do not do likelihood ratio tests.

7 Model selection

Our analysis selects a best model via backwards selection. We start with a model that is full quadratic in all three covariates (age at second instar, mass at second instar, and mass at eclosion). We then show by doing a Rao

test of model comparison that we can drop quadratic terms involving age. We then show by doing more Rao tests of model comparison that we cannot drop any other terms.

To conduct such a test, the score function and inverse observed Fisher information need to be calculated under the null model. The score function is

$$M_{\text{alter}}^T Y - \nabla c(M_{\text{alter}} \hat{\beta}_{\text{null}}) \tag{4}$$

where M_{alter} and $\hat{\beta}_{\text{null}}$ are the model matrix for the full quadratic model and the MLE for β using the null model, respectively (we do not have an offset vector in any of our models). We first calculate the term on the left in (4).

```
offspring <- as.numeric(grepl("B",data$redata$varb))
modmat.alt <- model.matrix(resp ~ varb +
  offspring:(Mass_Repro + Mass_2nd +
  I(Mass_Repro^2) + I(Mass_2nd^2) + I(Mass_2nd*Mass_Repro) +
  Time_2nd + I(Time_2nd^2) + I(Mass_2nd*Time_2nd) +
  I(Time_2nd*Mass_Repro)), data = data$redata)
tau.alt <- crossprod(modmat.alt, data$redata$resp)
```

We now calculate $\hat{\beta}_{\text{null}}$ by mapping $\hat{\tau}_{\text{null}}$ to the β parameterization using the `transformUnconditional` function, this null hypothesis being the one in which `Time_Repro` is linear and the other two covariates are quadratic.

```
modmat.null <- model.matrix(resp ~ varb +
  offspring:(Mass_Repro + Mass_2nd +
  I(Mass_Repro^2) + I(Mass_2nd^2) + I(Mass_2nd*Mass_Repro) +
  Time_2nd), data = data$redata)
tau <- crossprod(modmat.null, data$redata$resp)
beta <- beta.null <- transformUnconditional(tau, modmat.null,
  data, from = "tau", to = "beta", tolerance = 1e-100)
```

The term on the right in (4) is calculated using the `transformUnconditional` function.

```
tau.null <- transformUnconditional(c(beta.null,rep(0,3)),
  modmat.alt, data, from = "beta", to = "tau")
score <- tau.alt - tau.null
```

Observed Fisher information is calculated using the `jacobian` function. Three zeros are added to $\hat{\beta}_{\text{null}}$ in order to calculate observed Fisher information for the alternative model at the MLE for the null model. The three

zeros are for the terms (the quadratic term for age at second instar and the two crossproduct terms of age at second instar with one of the mass covariates) that are present in the alternative but absent in the null (which is the same as being set to zero).

```
Fisher.null <- jacobian(c(beta.null,rep(0,3)), data,
  transform = "unconditional", from = "beta",
  to = "tau", modmat = modmat.alt)
```

We now construct the Rao test statistic. The reference distribution for this test is χ^2_3 and the p-value suggests that we fail to reject the null hypothesis when testing at the 0.05 significance level.

```
Rao <- t(score) %*% solve(Fisher.null, tol = 1e-100) %*% score
pchisq(Rao, df = 3, lower = FALSE)

##           [,1]
## [1,] 0.1048548
```

All less complicated models are found to be insignificant when testing at the 0.05 significance level. Our final model includes the full quadratic structure between the two mass terms and a linear term for the age at which an individual *M. sexta* reaches its second instar larval stage as useful predictors of unconditional expected ovariole counts. First, we consider removing the quadratic terms for mass at second instar. The p-value is small enough to reject this smaller model at any reasonable significance level. The steps to conduct this test are similar to those that conducted our first hypothesis test.

```
modmat.null <- model.matrix(resp ~ varb +
  offspring:(Mass_Repro + I(Mass_Repro^2) + Time_2nd +
    Mass_2nd),
  data = data$redata)
tau <- crossprod(modmat.null, data$redata$resp)
beta.null <- transformUnconditional(tau, modmat.null,
  data, from = "tau", to = "beta", tolerance = 1e-100)

modmat.alt <- model.matrix(resp ~ varb +
  offspring:(Mass_Repro + I(Mass_Repro^2) + Time_2nd +
  Mass_2nd + I(Mass_Repro*Mass_2nd) + I(Mass_2nd^2)),
```

```
  data = data$redata)
tau.alt <- crossprod(modmat.alt, data$redata$resp)

tau.null <- transformUnconditional(c(beta.null,rep(0,2)),
  modmat.alt, data, from = "beta", to = "tau")
score <- tau.alt - tau.null

Fisher.null <- jacobian(c(beta.null,rep(0,2)), data,
  transform = "unconditional", from = "beta",
  to = "tau", modmat = modmat.alt)
Rao <- t(score) %*% solve(Fisher.null, tol = 1e-100) %*% score
pchisq(Rao, df = 2, lower = FALSE)

##              [,1]
## [1,] 3.371453e-08
```

We now consider removing the quadratic terms for mass at eclosion. The p-value is small enough to reject this smaller model at any reasonable significance level.

```
modmat.null <- model.matrix(resp ~ varb +
  offspring:(Mass_2nd + I(Mass_2nd^2) + Time_2nd +
    Mass_Repro),
  data = data$redata)
tau <- crossprod(modmat.null, data$redata$resp)
beta.null <- transformUnconditional(tau, modmat.null,
  data, from = "tau", to = "beta", tolerance = 1e-100)

modmat.alt <- model.matrix(resp ~ varb +
  offspring:(Mass_2nd + I(Mass_2nd^2) + Time_2nd +
    Mass_Repro + I(Mass_Repro^2) + I(Mass_Repro*Mass_2nd)),
  data = data$redata)
tau.alt <- crossprod(modmat.alt, data$redata$resp)

tau.null <- transformUnconditional(c(beta.null,rep(0,2)),
  modmat.alt, data, from = "beta", to = "tau")
score <- tau.alt - tau.null

Fisher.null <- jacobian(c(beta.null,rep(0,2)), data,
  transform = "unconditional", from = "beta",
```

```
  to = "tau", modmat = modmat.alt)
Rao <- t(score) %*% solve(Fisher.null, tol = 1e-100) %*% score
pchisq(Rao, df = 2, lower = FALSE)

##              [,1]
## [1,] 6.739063e-89
```

Finally, we consider removing the linear term for age that individuals reach their second instar stage. The p-value is small enough to reject this smaller model at any reasonable significance level.

```
modmat.null <- model.matrix(resp ~ varb +
  offspring:(Mass_2nd + I(Mass_2nd^2) + Mass_Repro +
  I(Mass_Repro^2) + I(Mass_Repro*Mass_2nd)),
  data = data$redata)
tau <- crossprod(modmat.null, data$redata$resp)
beta.null <- transformUnconditional(tau, modmat.null,
  data, from = "tau", to = "beta", tolerance = 1e-100)

modmat.alt <- model.matrix(resp ~ varb +
  offspring:(Mass_2nd + I(Mass_2nd^2) + Mass_Repro +
  I(Mass_Repro^2) + I(Mass_Repro*Mass_2nd) + Time_2nd),
  data = data$redata)
tau.alt <- crossprod(modmat.alt, data$redata$resp)

tau.null <- transformUnconditional(c(beta.null,0),
  modmat.alt, data, from = "beta", to = "tau")
score <- tau.alt - tau.null

Fisher.null <- jacobian(c(beta.null,0), data,
  transform = "unconditional", from = "beta",
  to = "tau", modmat = modmat.alt)
Rao <- t(score) %*% solve(Fisher.null, tol = 1e-100) %*% score
pchisq(Rao, df = 1, lower = FALSE)

##              [,1]
## [1,] 7.878988e-05
```

The results of the tests that compare our final model to smaller models are concisely summarized in Table 1.

null model	df	P-value
removes quadratic terms for mass at second instar	2	3.37×10^{-8}
removes quadratic terms for mass at eclosion	2	$< 10^{-10}$
removes linear term for age at second instar	1	7.88×10^{-5}

Table 1: Rao tests for smaller models. P-values and degrees of freedom for Rao tests of three smaller models against the larger model that includes linear, quadratic, and interaction term for the two mass traits and a linear term for age at second larval instar stage.

8 Fitness landscapes

Now that we have selected the best model, we want to plot fitness landscape, unconditional expected fitness as a function of covariates. We do this by predicting, on the basis of the model, values for fitness for all observed values of age at second instar and samples of 101 values of mass at second instar and 101 values of mass at eclosion. Since this is a four-dimensional graph (fitness versus three covariates), it cannot be visualized, and we make one three-dimensional graph for each age at 2nd instar using the R function `contour`. Covariate values that maximize expected fitness are also of interest.

Before we do this we need to extract some useful information from the `hornworm` dataset.

```
vars <- levels(hornworm$redata$varb)
pred <- hornworm$pred
group <- hornworm$group
code <- hornworm$code
families <- hornworm$families
```

Now the 101^2 hypothetical individuals with unique mass traits reaching their second instar larval stage at age 2 are generated

```
x <- seq(from = 0, to = 0.016, by = 0.016/100)
y <- seq(from = 0, to = 2.3, by = 2.3/100)
days <- 2:6 / 7
long.sex <- factor(rep("F", 101^2), levels = c("F", "M", "U"))
mnew <- data.frame(long.sex, rep(days[1], 101^2),
    rep(x, times = 101), rep(y, each = 101))
mnew <- cbind(mnew, matrix(rep(0, 26*101^2), ncol = 26))
```

```
mnames <- c("Sex", "Time_2nd", "Mass_2nd", "Mass_Repro", vars)
names(mnew) <- mnames
```

We then make the object of class `asterdata` for this new data and the model matrix for our best model and these new data.

```
data.mnew <- asterdata(mnew, vars = vars, pred = pred,
  group = group, code = code, families = families)
foo <- as.character(data.mnew$redata$varb)
offspring.mnew <- as.numeric(grepl("B", foo))
data.mnew$redata <- transform(data.mnew$redata,
  offspring.mnew = offspring.mnew)

modmat.mnew <- model.matrix(resp ~ varb +
  offspring.mnew:(Mass_Repro + Mass_2nd +
  I(Mass_Repro^2) + I(Mass_2nd^2) + I(Mass_2nd*Mass_Repro) +
  Time_2nd), data = data.mnew$redata)
```

This model matrix is then used to find the MLE of the saturated model unconditional canonical parameter φ.

```
phi <- modmat.mnew %*% beta
```

Estimates on the canonical scale are not directly interpretable. We need estimates for the mean-value parameters of the unconditional model (μ). These are found using the `transformSaturated` function. Only the estimates corresponding to ovariole count nodes are of interest. For every hypothetical individual there are eight predictions of ovariole count, one for every age that the individual can reach its reproduction stage. These predictions are summed to arrive at an estimate of expected ovariole count for each of the 101^2 hypothetical individuals.

Of course, in this biological system observed fitness for a single individual is nonzero at only one age, the age at which the individual reproduces (at most one Bx variable in the graph on p. 2 is nonzero). But expected fitness is nonzero at all ages (all components of the unconditional mean value parameter vector μ are nonzero). Adding the components of μ for all the Bx nodes of the graph adds the contribution to expected fitness of reproduction at all ages.

```
mus2 <- transformSaturated(phi, data.mnew, from = "phi",
  to = "mu")
ind <- which(grepl("B",rownames(data.frame(modmat.mnew))))
qux2 <- matrix(mus2[ind], nrow = 101^2, ncol = 8)
colnames(qux2) <- paste("c", 1:8, sep = "")
rownames(qux2) <- c(1:101^2)
sums2 <- as.numeric(apply(qux2, 1, sum))
zday2 <- matrix(sums2, nrow = 101)
```

The `maximum` function, defined below, is used to find the two mass values yielding the highest expected fitness.

```
maximum <- function(sums){
  max.ind <- which(sums == max(sums))
  column <- floor(max.ind / 101)
  row <- max.ind - 101 * column
  point <- c(x[row], y[column])
  return(point)
}
```

To make the plot we need to pull more things out of the `hornworm` dataset. The data are in the `redata` component of the `asterdata` object. We pull it out. This is an object in so-called long format (in the terminology of the R function `reshape`) so all of the covariates are repeated as many times as there are nodes in the graph. We get rid of this repetition because we are only interested in covariates here. And then we keep the data for females only.

```
mass.second <- hornworm$redata$Mass_2nd
mass.reprod <- hornworm$redata$Mass_Repro
sex <- as.character(hornworm$redata$Sex)
unique(sex)

## [1] "F" "M" "U"

time.second <- hornworm$redata$Time_2nd
id <- data$redata$id
u <- unique(id)
idx <- match(u, id)
mass.second <- mass.second[idx]
```

```
mass.reprod <- mass.reprod[idx]
sex <- sex[idx]
time.second <- time.second[idx]
mass.second <- mass.second[sex == "F"]
mass.reprod <- mass.reprod[sex == "F"]
```

The fitness landscape for these 101^2 hypothetical individuals reaching the second instar stage at age 2 is plotted in Figure 1.

The code below constructs the fitness landscape plots for hypothetical individuals reaching the second instar larval stage at ages 3, 4, and 5.

```
mnew <- data.frame(long.sex, rep(days[2], 101^2),
    rep(x, times = 101), rep(y, each = 101))
mnew <- cbind(mnew, matrix(rep(0, 26*101^2), ncol = 26))
mnames <- c("Sex", "Time_2nd", "Mass_2nd", "Mass_Repro", vars)
names(mnew) <- mnames
data.mnew <- asterdata(mnew, vars = vars, pred = pred,
  group = group, code = code, families = families)
foo <- as.character(data.mnew$redata$varb)
offspring.mnew <- as.numeric(grepl("B", foo))
data.mnew$redata <- transform(data.mnew$redata,
  offspring.mnew = offspring.mnew)

modmat.mnew <- model.matrix(resp ~ varb +
  offspring.mnew:(Mass_Repro + Mass_2nd +
  I(Mass_Repro^2) + I(Mass_2nd^2) + I(Mass_2nd*Mass_Repro) +
  Time_2nd), data = data.mnew$redata)
phi <- crossprod(t(modmat.mnew), beta)
mus3 <- transformSaturated(phi, data.mnew, from = "phi",
  to = "mu")

ind <- which(grepl("B",rownames(data.frame(modmat.mnew))))
qux3 <- matrix(mus3[ind], nrow = 10201, ncol = 8)
colnames(qux3) <- paste("c", 1:8, sep = "")
rownames(qux3) <- c(1:10201)
sums3 <- as.numeric(apply(qux3, 1, sum))
zday3 <- matrix(sums3, nrow = 101)

mnew <- data.frame(long.sex, rep(days[3], 101^2),
```

```
plot(mass.second, pch = 20,
  mass.reprod, xlab = "Mass at
  2nd instar", ylab = "Mass at Reproduction",
  main = "Fitness Landscape for Ovariole Counts vs. Mass")
points(maximum(sums2)[1], maximum(sums2)[2], col = "red",
  pch = 19)
contour(x,y,zday2, add = TRUE, nlevels = 8, levels =
  c(100,150,200,250,300,325,350,360) )
```

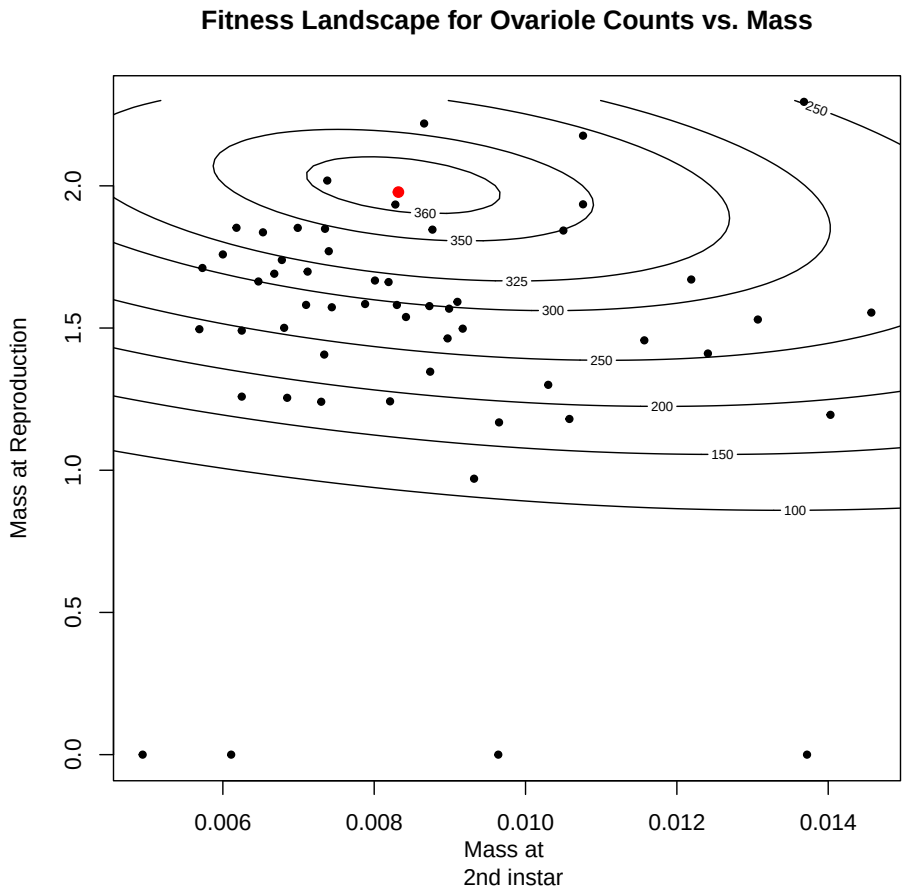


Figure 1: Fitness Landscape for Those Reaching Second Instar at Day 2.

```
    rep(x, times = 101), rep(y, each = 101))
mnew <- cbind(mnew, matrix(rep(0, 26*101^2), ncol = 26))
mnames <- c("Sex", "Time_2nd", "Mass_2nd", "Mass_Repro", vars)
names(mnew) <- mnames
data.mnew <- asterdata(mnew, vars = vars, pred = pred,
  group = group, code = code, families = families)
foo <- as.character(data.mnew$redata$varb)
offspring.mnew <- as.numeric(grepl("B", foo))
data.mnew$redata <- transform(data.mnew$redata,
  offspring.mnew = offspring.mnew)

modmat.mnew <- model.matrix(resp ~ varb +
  offspring.mnew:(Mass_Repro + Mass_2nd +
  I(Mass_Repro^2) + I(Mass_2nd^2) + I(Mass_2nd*Mass_Repro) +
  Time_2nd), data = data.mnew$redata)
phi <- crossprod(t(modmat.mnew), beta)
mus4 <- transformSaturated(phi, data.mnew, from = "phi",
  to = "mu")

ind <- which(grepl("B",rownames(data.frame(modmat.mnew))))
qux4 <- matrix(mus4[ind], nrow = 10201, ncol = 8)
colnames(qux4) <- paste("c", 1:8, sep = "")
rownames(qux4) <- c(1:10201)
sums4 <- as.numeric(apply(qux4, 1, sum))
zday4 <- matrix(sums4, nrow = 101)

mnew <- data.frame(long.sex, rep(days[4], 101^2),
    rep(x, times = 101), rep(y, each = 101))
mnew <- cbind(mnew, matrix(rep(0, 26*101^2), ncol = 26))
mnames <- c("Sex", "Time_2nd", "Mass_2nd", "Mass_Repro", vars)
names(mnew) <- mnames
data.mnew <- asterdata(mnew, vars = vars, pred = pred,
  group = group, code = code, families = families)
foo <- as.character(data.mnew$redata$varb)
offspring.mnew <- as.numeric(grepl("B", foo))
data.mnew$redata <- transform(data.mnew$redata,
  offspring.mnew = offspring.mnew)

modmat.mnew <- model.matrix(resp ~ varb +
```

```
  offspring.mnew:(Mass_Repro + Mass_2nd +
  I(Mass_Repro^2) + I(Mass_2nd^2) + I(Mass_2nd*Mass_Repro) +
  Time_2nd), data = data.mnew$redata)
phi <- crossprod(t(modmat.mnew), beta)
mus5 <- transformSaturated(phi, data.mnew, from = "phi",
  to = "mu")

ind <- which(grepl("B",rownames(data.frame(modmat.mnew))))
qux5 <- matrix(mus5[ind], nrow = 10201, ncol = 8)
colnames(qux5) <- paste("c", 1:8, sep = "")
rownames(qux5) <- c(1:10201)
sums5 <- as.numeric(apply(qux5, 1, sum))
zday5 <- matrix(sums5, nrow = 101)
```

The code below builds the fitness landscapes. These fitness landscapes appear in the left hand column of Figure 2 in the paper, which is called Figure 4 in this technical report. In the paper, the fitness landscapes appear smaller than those on display in Figure 2.

```
par(mfrow = c(2, 2), mar = rep(0.5, 4),
  oma = c(4, 4, 0, 0) + 0.1)
# day 2
plot(mass.second,
  mass.reprod,
  axes = FALSE, xlab = "", ylab = "", pch = 20)
box()
axis(side = 2, outer = TRUE)
mtext("mass at eclosion", outer = TRUE, line = 3,
  side = 2, at = 0.50)
mtext("mass at 2nd instar stage", outer = TRUE, line = 3,
  side = 1, at = 0.50)
points(maximum(sums2)[1], maximum(sums2)[2], pch = 0)
levels <- seq(50, 350, by = 50)
contour(x, y, zday2, add = TRUE, levels = levels)
# day 3
plot(mass.second,
  mass.reprod,
  axes = FALSE, xlab = "", ylab = "", pch = 20)
box()
```

```
points(maximum(sums3)[1], maximum(sums3)[2], pch = 0)
contour(x, y, zday3, add = TRUE, levels = levels)
# day 4
plot(mass.second,
  mass.reprod,
  axes = FALSE, xlab = "", ylab = "", pch = 20)
box()
axis(side = 1, outer = TRUE)
axis(side = 2, outer = TRUE)
points(maximum(sums4)[1], maximum(sums4)[2], pch = 0)
contour(x, y, zday4, add = TRUE, levels = levels)
# day 5
plot(mass.second,
  mass.reprod,
  axes = FALSE, xlab = "", ylab = "", pch = 20)
box()
axis(side = 1, outer = TRUE)
points(maximum(sums5)[1], maximum(sums5)[2], pch = 0)
contour(x, y, zday5, add = TRUE, levels = levels)
```

Figure 2 displays the entire fitness landscape for all hypothetical individuals. The fitness landscapes generated show that expected unconditional ovariole count is predominantly explained by an individual's mass at eclosion. As mass at eclosion increases, expected fitness increases with a maximum fitness found for female *M. sexta* weighing roughly 2 grams at eclosion. Expected fitness is also influenced by age at which individuals reach the second instar larval stage. The contours show that individuals reaching the second instar larval stage earlier have higher expected numbers of offspring. The contours suggest that fitness depends only weakly on mass at the second instar stage. However, the formal hypothesis test shows that this relationship is highly significant. In addition, expected unconditional ovariole count declines with increasing age to 2nd instar (Fig. 2). For example, maximum expected ovariole count declines by 16% as age to 2nd instar increases from 2 (upper left panel) to 5 (lower right panel) days. This effect is largely due to the effects of age at 2nd instar on survival to eclosion: slower development (later age at 2nd instar) is associated with lower survival.

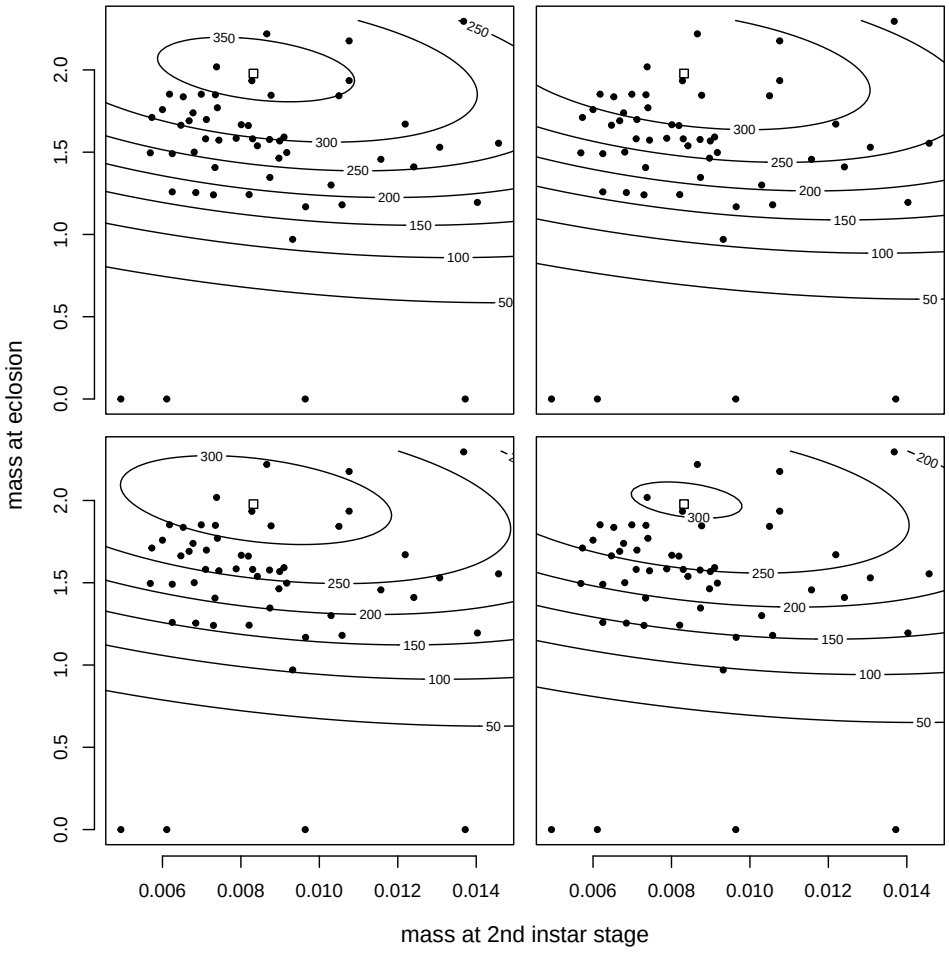


Figure 2: Fitness landscapes for expected unconditional ovariole counts vs. mass at eclosion and mass at 2nd instar stage. Different panels are for different ages (in days since hatching) at which individuals reached the second instar larval stage (age 2 is top left, age 3 is top right, age 4 is bottom left, and age 5 is bottom right). Mass at second instar stage is on the x-axis, and mass at eclosion is on the y-axis. The boxes denote the maxima. The maximum values are 363.6 (top left), 342.4 (top right), 322.4 (bottom left), and 303.6 (bottom-right).

9 Population growth rate

The preceding analysis accounts for survival and ovariole count as components of fitness, but does not take into account the role of variation in timing of reproduction in fitness variation. To incorporate this effect we must consider the expected population growth rate. The population growth rate parameter (λ) for the observed population of *M. sexta* is estimated from the stable age equation discussed in (Fisher, 1930, p. 26), as the basis of accounting for individuals' age at reproduction in its lifetime fitness. In our context, this is

$$1 = \frac{1}{n} \sum_{i=1}^{n} \sum_{x=33}^{40} \mu_{ix} e^{-\lambda x} \tag{5}$$

where μ_{ix} is the unconditional expected ovariole count for individual i at day x which is given below.

```
mu.star <- tau.alt[grepl("varbB", rownames(tau.alt))]
```

Having $e^{-\lambda x}$ instead of ρ^x in (5) follows Charlesworth (1980). The population growth rate parameter λ is calculated here using the `uniroot` function.

```
nind <- length(unique(hornworm$redata$id))
nind

## [1] 162

n <- nind
gr <- function(lam){
  n - sum(mu.star * exp(-(lam * (32 + seq(along = mu.star)))))
}
lout <- uniroot(gr, lower = -105, upper = 105)
lambda.hat <- lout$root
lambda.hat

## [1] 0.1215653
```

In most treatments of the stable age equation, starting with Fisher, the term μ_{ix} in (5) is written as the product of probability of survival to age x and the conditional expectation of number of offspring at age x given survival to age x, but we do not do that because μ_{ix} is calculated directly by the aster software. Most treatments of the stable age equation, starting

with Fisher, do not average over all individuals in the data, the $(1/n)\sum_{i=1}^{n}$ in (5). That is because those treatments do not allow for variation among individuals. Consequently, they use the same model for all individuals and apply the stable age equation to one individual (and hence to all because all are the same according to the model). Here, where μ_{ix} is different for different individuals because of the covariates in the model (mass at second instar, mass at eclosion, and age at second instar), we replace the means for a typical individual with the average over all individuals in the data.

From the $\hat{\mu}_{ix}$ produced by the aster model and (5) we obtain the estimate $\hat{\lambda} = 0.122$. The positive value of λ indicates a growing population. This very large value of $\hat{\lambda}$, indicating that the population grows by a factor of $\exp(\hat{\lambda}) = 1.129$ per day, results from the fact that many sources of mortality in natural populations were excluded from the experiment (e. g., low densities of larvae, the netting to exclude predation by birds, and lack of predation before larvae were moved from the lab to the field garden). Such overestimates of population growth rate are typical of experiments that do not have all sources of natural mortality and all sources of failure to reproduce.

We examine the effects that λ has on expected fitness by reweighting the fitness landscape according to

$$w(z) = \sum_{x=33}^{40} \mu_x(z) e^{-\lambda x}, \tag{6}$$

where $\mu_x(z)$ is now expected reproduction at age x for a hypothetical individual having trait values given by z (Charlesworth, 1980, p. 134). The weights according to the population growth rate used are

```
weight <- function(t) exp(-lambda.hat * (32 + t))
weight(c(1:8))

## [1] 0.018103415 0.016031176 0.014196139 0.012571153 0.011132173 0.009857909
## [7] 0.008729506 0.007730268
```

Fitness is now reweighted to incorporate the population growth rate. We refit the model using weights

$$w_j = f_j e^{-\lambda t_j} \tag{7}$$

where f_j are the zero or one weights indicating nodes that contribute directly to fitness and t_j is the age of the individual at node j. This weighting

accounts for population growth rate (Charlesworth, 1980, p. 134). We incorporate the weights using the code below.

```
foo <- data$redata$varb
offspring <- as.numeric(grepl("B", foo))
bar <- grepl("B", foo)
baz <- offspring
baz[bar] <- sub("B", "", foo)[bar]
baz <- as.numeric(baz)
baz[bar] <- weight(baz[bar] - 32)
offspring <- baz
offspring <- offspring * 100
```

A new model matrix is created that incorporates reweighted fitness.

```
modmat.full <- model.matrix(resp ~ varb +
  offspring:(Mass_Repro + Mass_2nd +
  I(Mass_Repro^2) + I(Mass_2nd^2) + I(Mass_2nd*Mass_Repro) +
  Time_2nd), data = data$redata)
```

We now obtain the new aster submodel mean-value parameter τ associated with reweighted fitness. The `transformUnconditional` function is used to find the corresponding submodel canonical parameter vector β.

```
tau.full <- crossprod(modmat.full, data$redata$resp)
beta.full <- transformUnconditional(tau.full,
  modmat.full, data, from = "tau", to = "beta")
```

The same hypothetical individuals used to build the fitness landscape before adjusting for the population growth rate are used to build the fitness landscape after we adjust for the population growth rate.

```
x <- seq(from = 0, to = 0.016, by = 0.016/100)
y <- seq(from = 0, to = 2.3, by = 2.3/100)
days <- 2:6 / 7
mnew <- data.frame(long.sex, rep(days[1], 101^2),
    rep(x, times = 101), rep(y, each = 101))
mnew <- cbind(mnew, matrix(rep(0, 26*101^2), ncol = 26))
mnames <- c("Sex", "Time_2nd", "Mass_2nd", "Mass_Repro", vars)
names(mnew) <- mnames
```

This model matrix is converted into an `asterdata` object in order to find estimates of fitness using the `aster2` package.

```
data.mnew <- asterdata(mnew, vars = vars, pred = pred,
  group = group, code = code, families = families)
```

The population growth rate is now incorporated into the model matrix for the hypothetical individuals.

```
foo <- as.character(data.mnew$redata$varb)
bar <- grepl("B", foo)
offspring.mnew <- as.numeric(bar)

baz <- offspring.mnew
baz[bar] <- sub("B", "", foo)[bar]
baz <- as.numeric(baz)
baz[bar] <- weight(baz[bar] - 32)
offspring.mnew <- baz
offspring.mnew <- 100 * offspring.mnew
data.mnew$redata <- transform(data.mnew$redata,
  offspring.mnew = offspring.mnew)
```

$\mu_x(z)$ is calculated in two steps. We first transform from β to φ manually and then use the `transformSaturated` function to transform from φ to μ as done previously.

```
modmat.mnew <- model.matrix(resp ~ varb +
  offspring.mnew:(Mass_Repro + Mass_2nd +
  I(Mass_Repro^2) + I(Mass_2nd^2) + I(Mass_2nd*Mass_Repro) +
  Time_2nd), data = data.mnew$redata)
phi <- crossprod(t(modmat.mnew), beta.full)
mus.fit2 <- transformSaturated(phi, data.mnew, from = "phi",
  to = "mu")
```

We now build the fitness landscape after adjusting for the population growth rate. The fitness landscape calculated using the code below is only for individuals that reach their second instar stage at age 2. The figure generated compares the fitness landscape constructed here to the fitness landscape unadjusted for λ for individuals that reach their second instar stage at age 2.

```
ind <- which(grepl("B",rownames(data.frame(modmat.mnew))))
qux.fit2 <- matrix(mus.fit2[ind], nrow = 10201, ncol = 8)
sumslam.fit2 <- as.numeric(apply(qux.fit2, 1, sum))
zday.fit2 <- matrix(sumslam.fit2, nrow = 101)
```

We can see that the contours of the two fitness landscapes in Figure 3 differ. We now change the fitness landscape adjusted for the population growth rate to a relative fitness landscape. This is done by dividing estimated expected ovariole counts by the mean of estimated expected ovariole counts.

```
mymu.full <- transformUnconditional(beta.full,
  modmat.full, data, from = "beta", to = "mu")
names(mymu.full) <- rownames(modmat.full)
mymu.full.births <- mymu.full[grepl("B", names(mymu.full))]
mean.fitness <- mean(mymu.full.births)
mean.fitness <- mean.fitness * 8
zday.fit2 <- zday.fit2 / mean.fitness
```

The same routine is performed for the 30603 individuals that reach their second instar life stage at ages 3, 4, and 5.

```
mnew <- data.frame(long.sex, rep(days[2], 101^2),
    rep(x, times = 101), rep(y, each = 101))
mnew <- cbind(mnew, matrix(rep(0, 26*101^2), ncol = 26))
mnames <- c("Sex", "Time_2nd", "Mass_2nd", "Mass_Repro", vars)
names(mnew) <- mnames
data.mnew <- asterdata(mnew, vars = vars, pred = pred,
  group = group, code = code, families = families)
foo <- as.character(data.mnew$redata$varb)
offspring.mnew <- as.numeric(bar)

baz <- offspring.mnew
baz[bar] <- sub("B", "", foo)[bar]
baz <- as.numeric(baz)
baz[bar] <- weight(baz[bar] - 32)
offspring.mnew <- baz
offspring.mnew <- 100 * offspring.mnew
data.mnew$redata <- transform(data.mnew$redata,
```

```
plot(mass.second, pch = 20,
  mass.reprod, xlab = "Mass at
  2nd instar", ylab = "Mass at Reproduction",
  main = "Fitness Landscape for Ovariole Counts vs. Mass")
points(maximum(sums2)[1], maximum(sums2)[2], col = "red",
  pch = 19)
contour(x,y,zday2, add = TRUE, nlevels = 8, levels =
  c(100,150,200,250,300,325,350,360) )


plot(mass.second, pch = 20,
  mass.reprod, xlab = "Mass at
  2nd instar", ylab = "Mass at Reproduction",
  main = "Fitness Landscape for Ovariole Counts vs. Mass")
points(maximum(sumslam.fit2)[1], maximum(sumslam.fit2)[2],
  col = "red", pch = 19)
contour(x,y,zday.fit2, add = TRUE, nlevels = 8)
```

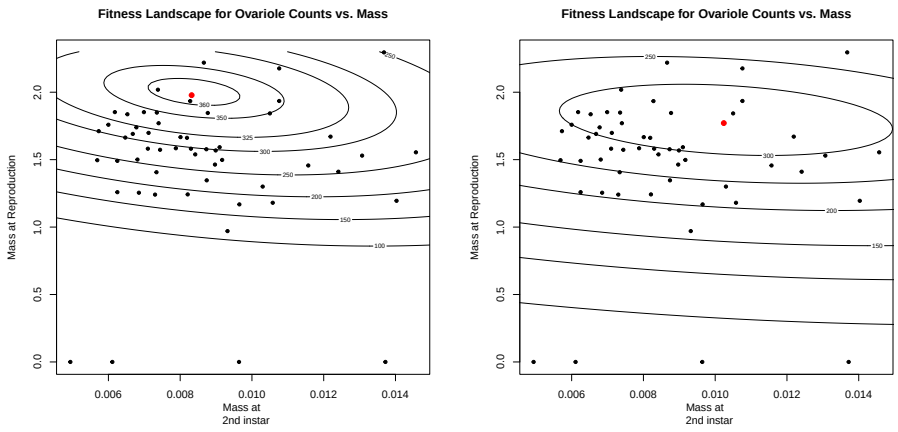


Figure 3: Fitness landscapes without (left panel) and with (right panel) adjustment for population growth rate λ for hypothetical individuals reaching their second instar stage at age 2.

```
  offspring.mnew = offspring.mnew)

modmat.mnew <- model.matrix(resp ~ varb +
  offspring.mnew:(Mass_Repro + Mass_2nd +
  I(Mass_Repro^2) + I(Mass_2nd^2) + I(Mass_2nd*Mass_Repro) +
  Time_2nd), data = data.mnew$redata)
phi <- crossprod(t(modmat.mnew), beta.full)
mus.fit3 <- transformSaturated(phi, data.mnew, from = "phi",
  to = "mu")

births.fit3 <- mus.fit3[ind]
quxlam.fit3 <- matrix(births.fit3, nrow = 10201)
sumslam.fit3 <- apply(quxlam.fit3, 1, sum)
zday.fit3 <- matrix(sumslam.fit3, nrow = 101)
zday.fit3 <- zday.fit3 / mean.fitness


mnew <- data.frame(long.sex, rep(days[3], 101^2),
    rep(x, times = 101), rep(y, each = 101))
mnew <- cbind(mnew, matrix(rep(0, 26*101^2), ncol = 26))
mnames <- c("Sex", "Time_2nd", "Mass_2nd", "Mass_Repro", vars)
names(mnew) <- mnames
data.mnew <- asterdata(mnew, vars = vars, pred = pred,
  group = group, code = code, families = families)
foo <- as.character(data.mnew$redata$varb)
offspring.mnew <- as.numeric(bar)

baz <- offspring.mnew
baz[bar] <- sub("B", "", foo)[bar]
baz <- as.numeric(baz)
baz[bar] <- weight(baz[bar] - 32)
offspring.mnew <- baz
offspring.mnew <- 100 * offspring.mnew
data.mnew$redata <- transform(data.mnew$redata,
  offspring.mnew = offspring.mnew)

modmat.mnew <- model.matrix(resp ~ varb +
  offspring.mnew:(Mass_Repro + Mass_2nd +
  I(Mass_Repro^2) + I(Mass_2nd^2) + I(Mass_2nd*Mass_Repro) +
```

```
  Time_2nd), data = data.mnew$redata)
phi <- crossprod(t(modmat.mnew), beta.full)
mus.fit4 <- transformSaturated(phi, data.mnew, from = "phi",
  to = "mu")

births.fit4 <- mus.fit4[ind]
quxlam.fit4 <- matrix(births.fit4, nrow = 10201)
sumslam.fit4 <- apply(quxlam.fit4, 1, sum)
zday.fit4 <- matrix(sumslam.fit4, nrow = 101)
zday.fit4 <- zday.fit4 / mean.fitness


mnew <- data.frame(long.sex, rep(days[4], 101^2),
    rep(x, times = 101), rep(y, each = 101))
mnew <- cbind(mnew, matrix(rep(0, 26*101^2), ncol = 26))
mnames <- c("Sex", "Time_2nd", "Mass_2nd", "Mass_Repro", vars)
names(mnew) <- mnames
data.mnew <- asterdata(mnew, vars = vars, pred = pred,
  group = group, code = code, families = families)
foo <- as.character(data.mnew$redata$varb)
offspring.mnew <- as.numeric(bar)

baz <- offspring.mnew
baz[bar] <- sub("B", "", foo)[bar]
baz <- as.numeric(baz)
baz[bar] <- weight(baz[bar] - 32)
offspring.mnew <- baz
offspring.mnew <- 100 * offspring.mnew
data.mnew$redata <- transform(data.mnew$redata,
  offspring.mnew = offspring.mnew)

modmat.mnew <- model.matrix(resp ~ varb +
  offspring.mnew:(Mass_Repro + Mass_2nd +
  I(Mass_Repro^2) + I(Mass_2nd^2) + I(Mass_2nd*Mass_Repro) +
  Time_2nd), data = data.mnew$redata)
phi <- crossprod(t(modmat.mnew), beta.full)
mus.fit5 <- transformSaturated(phi, data.mnew, from = "phi",
  to = "mu")
```

```
births.fit5 <- mus.fit5[ind]
quxlam.fit5 <- matrix(births.fit5, nrow = 10201)
sumslam.fit5 <- apply(quxlam.fit5, 1, sum)
zday.fit5 <- matrix(sumslam.fit5, nrow = 101)
zday.fit5 <- zday.fit5 / mean.fitness
```

We now make the eight-paneled plot that appears in the paper, see Figure 4. The plots in these panels are slightly different than those that appeared in Figure 3. We have zoomed out in order to capture more of the contour shapes. This allows one to easily see the differences in the contours before and after adjusting for the population growth rate.

```
xlim <- range(x)
ylim <- range(y)

par(mfrow = c(4, 2), mar = rep(0.5, 4),
  oma = c(4, 4, 0, 4) + 0.1)
# day 2
plot(mass.second,
    mass.reprod,
    axes = FALSE, xlab = "", ylab = "", pch = 20,
    xlim = xlim, ylim = ylim)
box()
axis(side = 2, outer = TRUE)
mtext("mass at eclosion", outer = TRUE, line = 3, side = 2,
  at = 0.50)
mtext("mass at 2nd instar stage", outer = TRUE, line = 3,
  side = 1, at = 0.50)
mtext("age second instar stage reached", outer = TRUE,
  line = 1, side = 4, at = 0.50)
mtext(as.character(2:5), outer = TRUE, line = 3,
  side = 4, at = c(7, 5, 3, 1) / 8)
points(maximum(sums2)[1], maximum(sums2)[2], pch = 0)
levels <- seq(50, 350, by = 50)
contour(x, y, zday2, add = TRUE, levels = levels)


plot(mass.second,
    mass.reprod,
```

```
    axes = FALSE, xlab = "", ylab = "", pch = 20,
    xlim = xlim, ylim = ylim)
box()
points(maximum(sumslam.fit2)[1], maximum(sumslam.fit2)[2],
  pch = 0)
contour(x, y, zday.fit2, nlevels = 7, add = TRUE)


# day 3
plot(mass.second,
    mass.reprod,
    axes = FALSE, xlab = "", ylab = "", pch = 20,
    xlim = xlim, ylim = ylim)
box()
axis(side = 2, outer = TRUE)
mtext("mass at eclosion", outer = TRUE, line = 3,
  side = 2, at = 0.50)
points(maximum(sums3)[1], maximum(sums3)[2], pch = 0)
contour(x, y, zday3, add = TRUE, levels = levels)


plot(mass.second,
    mass.reprod,
    axes = FALSE, xlab = "", ylab = "", pch = 20,
    xlim = xlim, ylim = ylim)
box()
points(maximum(sumslam.fit3)[1], maximum(sumslam.fit3)[2],
  pch = 0)
contour(x, y, zday.fit3, nlevels = 7, add = TRUE)


# day 4
plot(mass.second,
    mass.reprod,
    axes = FALSE, xlab = "", ylab = "", pch = 20,
    xlim = xlim, ylim = ylim)
box()
axis(side = 2, outer = TRUE)
mtext("mass at eclosion", outer = TRUE, line = 3,
```

```
  side = 2, at = 0.50)
points(maximum(sums4)[1], maximum(sums4)[2], pch = 0)
contour(x, y, zday4, add = TRUE, levels = levels)


plot(mass.second,
    mass.reprod,
    axes = FALSE, xlab = "", ylab = "", pch = 20,
    xlim = xlim, ylim = ylim)
box()
points(maximum(sumslam.fit4)[1], maximum(sumslam.fit4)[2],
  pch = 0)
contour(x, y, zday.fit4, nlevels = 7, add = TRUE)


# day 5
plot(mass.second,
    mass.reprod,
    axes = FALSE, xlab = "", ylab = "", pch = 20,
    xlim = xlim, ylim = ylim)
box()
axis(side = 1, outer = TRUE)
axis(side = 2, outer = TRUE)
points(maximum(sums5)[1], maximum(sums5)[2], pch = 0)
contour(x, y, zday5, add = TRUE, levels = levels)


plot(mass.second,
    mass.reprod,
    axes = FALSE, xlab = "", ylab = "", pch = 20,
    xlim = xlim, ylim = ylim)
box()
axis(side = 1, outer = TRUE)
points(maximum(sumslam.fit5)[1], maximum(sumslam.fit5)[2],
  pch = 0)
contour(x, y, zday.fit5, nlevels = 7, add = TRUE)
```

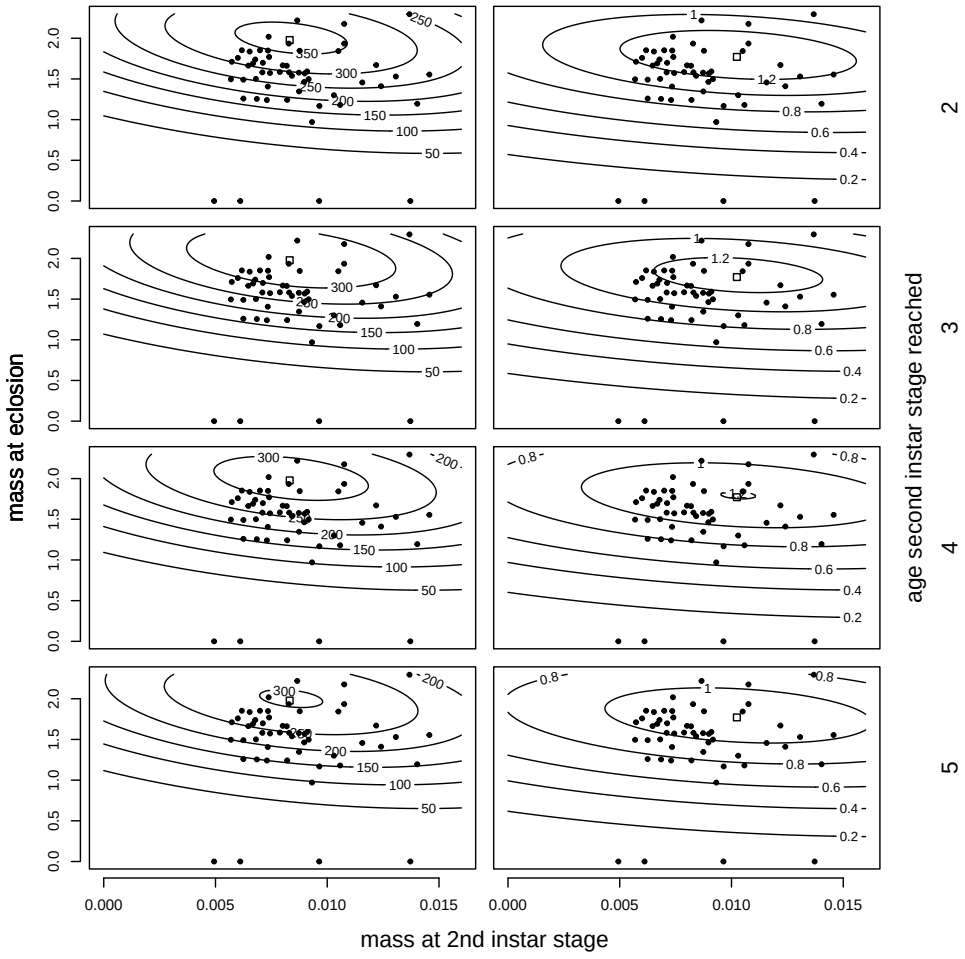


Figure 4: Fitness landscapes without (left column) and with (right column) adjustment for population growth rate λ. Rows top to bottom are 2nd instar stage reached at age 2, 3, 4, and 5. Numbers on contours are absolute fitness (unconditional expected ovariole counts) in the left column and are relative fitness (absolute fitness divided by its average over the population) in the right column. All plots display fitness as contours vs. mass at eclosion and mass at 2nd instar stage. Boxes denote locations of maxima. Maximum values are (left column, from top to bottom) 363.6, 342.4, 322.4, 303.6, (right column, from top to bottom) 1.39, 1.28, 1.20, 1.14.

A Make the Data

In this appendix we show how the `hornworm` dataset in the `aster2` package was initially created starting from the data for the original analysis (Kingsolver, et al., 2012), which was a comma separated values (CSV) file. This file can be found at the same location at the University of Minnesota Digital Conservancy (`http://conservancy.umn.edu/`) where this technical report is found.

```
MSexta <- read.csv("MSexta.Aster.csv")
names(MSexta)

## [1] "LarvaID"     "Sex"         "Surv_Eclose" "Time_2nd"    "Mass_2nd"
## [6] "Time_Eclose" "Mass_eclose" "total_eggs"
```

Some of the column names are changed for convenience. Any column name that ends in `Repro` means that the data in that column pertains to the eclosion stage of *M. sexta*.

```
names(MSexta)[c(3,6,7,8)] <- c("Surv_Repro","Time_Repro",
    "Mass_Repro","Eggs")
names(MSexta)

## [1] "LarvaID"    "Sex"        "Surv_Repro" "Time_2nd"   "Mass_2nd"
## [6] "Time_Repro" "Mass_Repro" "Eggs"
```

This dataset has two individuals that have an ovariole count but have sex declared as `NA`. We change sex from `NA` to female for these individuals.

```
sex.change <- (! is.na(MSexta$Eggs)) & is.na(MSexta$Sex)
sum(sex.change)

## [1] 2

MSexta[sex.change, "Sex"] <- "F"
```

Sex cannot be determined for *M. sexta* that die before pupation so the dataset at this point has `NA` values recorded for the sex of such individuals. The `aster` and `aster2` packages do not allow `NA` values in covariates, so change sex from `NA` to `U` for unobservable.

```
fred <- as.character(MSexta$Sex)
fred[is.na(fred)] <- "U"
MSexta$Sex <- as.factor(fred)
```

There is some data on female *M. sexta* that reached eclosion but had `NA` recorded for ovariole count. Since the fitness of such individuals is unknown (missing data), they provide no information about fitness, so we remove them from the data.

```
condition <- (MSexta$Sex == "F") & is.na(MSexta$Eggs) &
    (MSexta$Surv_Repro == 1)
sum(condition)

## [1] 7

MSexta <- MSexta[!condition,]
```

We now add the survival to pupation variable, called P in the aster graph (p. 2). This variable is constructed using the `Sex` variable. Every individual that has sex recorded as male or female survived to pupation, and every individual that has no sex recorded died before pupation.

```
P <- as.numeric(MSexta$Sex != "U")
MSexta <- data.frame(MSexta, P = P)
```

Other nodes in the graph are constructed using the survival to reproduction variables `Surv_Repro` and `Time_Repro`.

```
surv.repro <- MSexta$Surv_Repro
time.repro <- MSexta$Time_Repro
```

Time of death is not recorded for individuals that do not reach eclosion.

```
all(is.na(time.repro) == (surv.repro == 0))

## [1] TRUE
```

So that `NA` values do not propagate into the variables we are creating, we eliminate these `NA` values.

```
time.repro[is.na(time.repro)] <- 0
```

Since the day of death for individuals that did not survive to eclosion was not recorded, we cannot say on what day death occurred for these individuals and hence record the death at age 33 days. For all other individuals, we record when they reached eclosion by creating all the indicator variables in the graph. For each of the "T with subscripts" variables in the graph, we create an R variable of that name except we flatten the subscripts (T_{35_1} becomes `T351`). Because all deaths after pupation are recorded on day 33, that day is special. There is a `T330` variable but no `T`x`0` variable for $x > 33$. There are `T`x`1` and `T`x`2` variables for all x between 33 and 40.

```
T330 <- as.numeric((MSexta$Surv_Repro == 0) & (MSexta$P == 1))
for (j in 33:40) {
    varname1 <- paste("T", j, "1", sep = "")
    varname2 <- paste("T", j, "2", sep = "")
    assign(varname1, as.numeric(time.repro > j))
    assign(varname2, as.numeric(time.repro == j))
}
```

Now we also need the ovariole count variables, and we change `NA` to zero there too, which is harmless because all females that survive to eclosion now have egg counts.

We have another problem that will not arise until we use the `asterdata` function, but it is easiest to fix here. The graph for females is different from the graph for males and unobserved. Females have egg counts, which are the `B`x nodes of the graph. Others don't. But the R function `asterdata` only deals with data such that every individual has the same graph. The idea is do that first, and fix it later using the `subset` function to delete any unwanted nodes. But if we do that, the males and unknown must have valid egg counts (even though they are bogus and will be deleted later) or the `asterdata` function will complain. Since we are using the zero-truncated Poisson distribution for egg counts, every male who reaches eclosion must have at least one egg (even though this is bogus and we are going to delete his egg count node later). The unknowns do not reach eclosion, so no fix-up is needed for them.

```
eggs <- MSexta$Eggs
sex <- as.character(MSexta$Sex)
```

```
eggs[sex == "M"] <- NA
all(is.na(eggs) == ((surv.repro == 0) | (sex != "F")))

## [1] TRUE

eggs[is.na(eggs)] <- 0
eggs[(sex == "M") & (surv.repro == 1)] <- 1
```

The variable Bx is equal to the egg count for that individual if the individual reached eclosion on day x and is zero otherwise.

```
for (j in 33:40) {
    varname <- paste("B", j, sep = "")
    assign(varname, ifelse(time.repro == j, eggs, 0))
}
```

Finally we need to fix the one covariate that still has `NA` values.

```
all(is.na(MSexta$Mass_Repro) == (MSexta$Surv_Repro == 0))

## [1] TRUE

MSexta$Mass_Repro[is.na(MSexta$Mass_Repro)] <- 0
```

We now stuff all these variables into a data frame, in the process changing the units on some of the variables: the units for `Time_2nd` change from days to weeks, and the units for `Mass_2nd` and `Mass_Repro` change from milligrams to grams. These changes of units were perhaps unnecessary, but were done with the idea that they might make computations more stable.

```
Msextadata <- data.frame(LarvaID = MSexta$LarvaID, Sex = MSexta$Sex,
    Time_2nd = MSexta$Time_2nd / 7, Mass_2nd = MSexta$Mass_2nd / 1000,
    Mass_Repro = MSexta$Mass_Repro / 1000, P, T330, T331, T332, B33,
    T341, T342, B34, T351, T352, B35, T361, T362, B36, T371, T372,
    B37, T381, T382, B38, T391, T392, B39, T401, T402, B40)
```

And check for no `NA` values.

```
! any(is.na(Msextadata))

## [1] TRUE
```

Now we need to specify the rest of the graph shown in Figure 1.

```
vars <- c("P", "T330", "T331", "T332", "B33", "T341", "T342",
    "B34", "T351", "T352", "B35", "T361", "T362", "B36",
    "T371", "T372", "B37", "T381", "T382", "B38", "T391",
    "T392", "B39", "T401", "T402", "B40")
pred <- c(0, 1, 1, 1, 4, 3, 3, 7, 6, 6, 10, 9, 9, 13, 12, 12,
    16, 15, 15, 19, 18, 18, 22, 21, 21, 25)
group <- c(0, 0, 2, 3, 0, 0, 6, 0, 0, 9, 0, 0, 12, 0, 0, 15,
    0, 0, 18, 0, 0, 21, 0, 0, 24, 0)
families <- list("bernoulli", fam.multinomial(3),
    fam.zero.truncated.poisson(), fam.multinomial(2))
code <- c(1, 2, 2, 2, 3, 4, 4, 3, 4, 4, 3, 4, 4, 3, 4, 4, 3,
    4, 4, 3, 4, 4, 3, 4, 4, 3)
```

One of these variables that specify the graph is obvious. `vars` names the variables that are pasted together to form the response vector (all the variables at all the nodes of the graph). The others are less obvious. `pred` specifies the arrows of the graph, and `group` specifies the lines. If `pred[j]` is not zero, then there is an arrow from node `pred[j]` to node `j`. Otherwise, there is a line from the initial node (marked 1 in the aster graph) to node `j`. Similarly, if `group[j]` is nonzero, then there is a line from node `group[j]` to node `j`. `code` is an index vector into the families vector; `families[code[j]]` is the family for the conditional distribution of the dependence group containing node `j` given its predecessor node. Since these are so hard to follow, we check that these variables are set right.

```
charpred <- c("Initial", vars)[pred + 1]
groupidx <- seq(along = group)
for (j in seq(along = groupidx))
    if (group[j] != 0)
        groupidx[j] <- groupidx[group[j]]
data.frame(predecessor = charpred, successor = vars,
     group = groupidx, code)

##    predecessor successor group code
```

```
## 1      Initial         P     1    1
## 2            P      T330     2    2
## 3            P      T331     2    2
## 4            P      T332     2    2
## 5         T332       B33     5    3
## 6         T331      T341     6    4
## 7         T331      T342     6    4
## 8         T342       B34     8    3
## 9         T341      T351     9    4
## 10        T341      T352     9    4
## 11        T352       B35    11    3
## 12        T351      T361    12    4
## 13        T351      T362    12    4
## 14        T362       B36    14    3
## 15        T361      T371    15    4
## 16        T361      T372    15    4
## 17        T372       B37    17    3
## 18        T371      T381    18    4
## 19        T371      T382    18    4
## 20        T382       B38    20    3
## 21        T381      T391    21    4
## 22        T381      T392    21    4
## 23        T392       B39    23    3
## 24        T391      T401    24    4
## 25        T391      T402    24    4
## 26        T402       B40    26    3
```

In each line here, `predecessor` is the predecessor (variable at the tail of an arrow) and `successor` is the successor (variable at the head of an arrow) for some arrow in the graph, `group` is now an arbitrary number that is the same for successor nodes in the same dependence group and different for successor nodes in different dependence groups, and `code` is the same as before. We see that only {`T330`, `T331`, `T332`} form a three-node dependence group, and their distribution is (same for all three, as it must be, because this is their joint distribution) three-dimensional multinomial. We see that {`T`x`1`, `T`x`2`} form a two-node dependence group for $x > 33$, and their distribution is (same for both) two-dimensional multinomial. We see that P is a dependence group all by itself, and its distribution is Bernoulli. We see that `B`x is a dependence group all by itself for $x \geq 33$, and its distribution is zero-truncated Poisson. We see that the predecessor of P is the initial node,

the predecessor of the `T33` dependence group is P, the predecessor of the `T`x dependence group is the `T`w`1` node where $w = x - 1$ (the predecessor of T_{x_y} is T_{w_1}), and the predecessor of `B`x is `T`x`2` for $x \geq 33$. It all checks.

Now we make the asterdata object.

```
fred <- asterdata(Msextadata, vars = vars, pred = pred,
    group = group, code = code, families = families)
dim(fred$redata)

## [1] 4212    8

nrow(Msextadata) * length(vars)

## [1] 4212

names(fred$redata)

## [1] "LarvaID"    "Sex"        "Time_2nd"   "Mass_2nd"   "Mass_Repro"
## [6] "varb"       "resp"       "id"
```

And then we remove nodes of the graph that are not supposed to be there: ovariole count nodes for males and unknown.

```
condition2 <- (fred$redata$Sex == "F") | (! grepl("B", fred$redata$varb))
fred <- subset(fred, subset = condition2, successors = FALSE)
dim(fred$redata)

## [1] 3348    8

nrow(Msextadata) * length(vars) -
    sum(Msextadata$Sex != "F") * sum(grepl("B", vars))

## [1] 3348

names(fred$redata)

## [1] "LarvaID"    "Sex"        "Time_2nd"   "Mass_2nd"   "Mass_Repro"
## [6] "varb"       "resp"       "id"
```

Now we check this against the `hornworm` dataset from the `aster2` package.

```
identical(fred, hornworm)

## [1] FALSE
```

If `FALSE`, this is i18n insanity, which is not really R's fault. The details section of `help(Comparison)` has an explicit disclaimer that sort order is not guaranteed to work the same way on different machines.

```
foo <- fred$redata$LarvaID
bar <- hornworm$redata$LarvaID
identical(foo, bar)

## [1] FALSE

foo <- as.character(foo)
bar <- as.character(bar)
identical(foo, bar)

## [1] TRUE

baz <- levels(hornworm$redata$LarvaID)
all(foo %in% baz)

## [1] TRUE

qux <- factor(foo, levels = baz)
fred$redata$LarvaID <- qux
identical(fred, hornworm)

## [1] FALSE
```

References

Charlesworth, B. 1980. *Evolution in age-structured populations*. Cambridge Univ. Press, Cambridge, U. K.

Fisher, R. A. 1930. *The genetical theory of natural selection*. Clarendon Press, Oxford, U. K.

Geyer, C. J. (2009). Likelihood inference in exponential families and directions of recession. *Electronic Journal of Statistics*, **3**, 259–289.

Geyer, C. J. (2010). A philosophical look at aster models. Technical Report No. 676. School of Statistics, University of Minnesota. `http://purl.umn.edu/57163`.

Geyer, C. J. 2014. R package `aster` (aster models), version 0.8-30. `http://cran.r-project.org/package=aster`.

Geyer, C. J. 2015. R package aster2 (aster models), version 0.2-1. `http://cran.r-project.org/package=aster2`.

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

Kingsolver, J. G., S. E. Diamond, S. A. Seiter, and J. K. Higgins. 2012. Direct and indirect phenotypic selection on developmental trajectories in *Manduca sexta*. *Funct. Ecol.* 26:598–607.

R Development Core Team. 2014. R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. `http://www.R-project.org`.

Shaw, R. G., and C. J. Geyer. 2010. Inferring fitness landscapes. *Evolution* 64:2510–2520.

Shaw, R. G., C. J. Geyer, S. Wagenius, H. Hangelbroek, and J. R. Etterson. 2008. Unifying life-history analyses for inference of fitness and population growth. *Am. Nat.* 172:E35–E47.

Shaw, R. G., Wagenius, S., and Geyer, C. J. (2015). The susceptibility of *Echinacea angustifolia* to a specialist aphid: eco-evolutionary perspective on genotypic variation and demographic consequences. *Journal of Ecology*, **103**, 809–818.