Supporting Data Analysis for "Do defensive symbionts cause selection for greater pathogen virulence?"

Georgiana May, Ruth G. Shaw, Charles J. Geyer, Daniel J. Eck

April 7, 2021

In this document we reproduce the aster analyses that are presented in the "Do defensive symbionts cause selection for greater pathogen virulence?" manuscript. We begin with a description of the aster graphical structure that represents the lifecycle of the pathogen *Ustilago maydis* within the host plant *Zea mays*. We then describe how we converted the collected raw data into a form that is usable by `aster2` software (Geyer, 2010). From here we fit and tested candidate aster models, arriving at a final model using backwards selection. Pathogen fitness landscapes are then presented.

Contents

1 The aster graph

$$\begin{tikzcd}
GallWT.sd & & & & & & & & GallWT.sd \\
GallWT \arrow[u, dash] & PD1 \arrow[l] \arrow[ul] \arrow[dd, dash] & & & & & & PD1 \arrow[r] \arrow[ur] \arrow[dd, dash] & GallWT \arrow[u, dash] \\
& & per \arrow[ul] \arrow[dl] & LH3 \arrow[l] \arrow[r, dash] & LH1 \arrow[r, dash] & LH2 \arrow[r] & per \arrow[ur] \arrow[dr] & & \\
GallWT \arrow[d, dash] & PD0 \arrow[l] \arrow[dl] & & & & & & PD0 \arrow[r, dashed] \arrow[dr, dashed] & GallWT \arrow[d, dashed, dash] \\
GallWT.sd & & & & I \arrow[uul] \arrow[uu] \arrow[uur] & & & & GallWT.sd \\
& & & & 1 \arrow[u] & & & &
\end{tikzcd}$$

Figure 1: Modified Aster Graph for our data. Arrows go from predecessor nodes to successor nodes. Lines (that are not arrows) link dependence groups (Eck, et al., 2015 a). The dashed vectors and lines correspond to nodes where no data was observed. Developmental paths are designated LH. After infection (I) there is either no further disease development ($LH1$), development on leaves ($LH2$) or development on stems ($LH3$) and thus, $LH1$, $LH2$, $LH3$ form a dependency group. If the pathogen growth in the plant persists (per) after infection, it may either cause plant death ($PD0$) or not ($PD1$). Since plants either live or die but not both, $PD0$ and $PD1$ are a dependency group within each developmental path, $LH2$ or $LH3$. For both paths, the pathogen may produce galls filled with spores, a measure of reproduction (GallWT). In the LH3 path, galls were produced whether the plant lived or died ($PD1$, $PD0$) whereas in the $LH2$ path, plant death did not occur.

2 The Data

Before we begin manipulating the data into a useable form, we load the needed software. The `aster2` package (Geyer, 2010) has the capabilities to perform an aster analyses when nodes in the aster model graphical structure form a dependence group which is a requirement of this analysis.

```
library(Matrix)
library(methods)
library(MASS)
library(knitr)
library(aster2)
library(tidyverse)
```

We load the data and store it as `Patho`. The names of the variables and a snapshot of the information contained in `Patho` is seen below.

```
## load in data
dat <- read.csv("PathogenA4_6_7_15.csv", sep = ",",
  header = T)

## keep variables used in the analysis
Patho <- dat %>% dplyr::select("Block", "Treatment",
  "Fvert", "InVir", "Infect", "LH_stage", "Path_persist",
  "PlantLives2", "GallWT", "Umaydis", "Height19")

## snapshot of the data
head(Patho[, 1:6])

##   Block Treatment Fvert InVir Infect LH_stage
## 1     1     U2-21     0  8.21      1        0
## 2     1     U2-21     0  8.21      1        2
## 3     1     U2-21     0  8.21      0        0
## 4     1     U2-21     0  8.21      1        0
## 5     1     U2-21     0  8.21      1        3
## 6     1     U2-21     0  8.21      0        0

head(Patho[, 7:11])
```

```
##   Path_persist PlantLives2 GallWT Umaydis Height19
## 1            0           1      0      22       45
## 2            1           1   36.6      22       40
## 3            0           1      0      22     53.5
## 4            0           1      0      22       50
## 5            1           1  189.1      22       24
## 6            0           1      0      22       47
```

The "mock" treatments are filtered out from the dataset. Records with `Umaydis == 0` are removed as well.

```
Patho <- Patho %>% filter(Treatment != "Mock",
  Umaydis != 0)
```

This dataset has NA values present in various locations. All of these NA values are either associated with "mock" treatments or plants that did not grow (DNG) which was recorded in the `Height19` variable. We remove the DNG entries from the `Height19` variable.

```
Patho <- Patho %>% filter(Height19 != "DNG")
all(!is.na(Patho))

## [1] TRUE
```

We now convert the data into a format workable for aster software. We first specify the life history stage nodes (coded $LH1$, $LH2$, or $LH3$). The collection of nodes $LH1$, $LH2$, and $LH3$ are said to form a dependence group. Nodes that form a dependence group are intertwined in the sense that a particular individual can only progress to one of the nodes in the dependence group. Thus, these nodes individually act as switches that are coded 0 or 1 where a 1 indicates the path that a particular *U. maydis* was recorded to take. For example, if a particular *U. maydis* has $LH2 = 1$ recorded, then this pathogen is said to have limited growth and it will also have the values $LH3 = 0$ and $LH1 = 0$. All future realizations in the lifecycle for this particular *U. maydis* pathogen are conditional on the fact that it exhibited limited growth indicated by the switch $LH1 = 0$, $LH2 = 1$, $LH3 = 0$.

```r
assign("LH2", as.numeric(Patho$LH_stage == 2))
assign("LH3", as.numeric(Patho$LH_stage == 3))
unique(LH2 + LH3 - as.numeric(Patho$LH_stage > 0))

## [1] 0

cond <- LH2 + LH3 - as.numeric(Patho$Infect > 0) == -1
assign("LH1", as.numeric(cond))
unique(LH1 + LH2 + LH3 - as.numeric(Patho$Infect > 0))

## [1] 0
```

All remaining life stages have to be coded with respect to the growth the pathogen exhibited. Therefore, pathogen persistence, plant death, and gall weight nodes have to be constructed with respect to both the $LH2 = 1$ and $LH3 = 1$ cases. The $LH = 1$ case indicates no growth of pathogen within the plant and therefore this case has no ancestor nodes. We begin with pathogen persistence for both cases where the realization of both persistence and non-persistence is checked.

```r
assign("persists2", as.numeric(Patho$LH_stage == 2))
persists2[Patho$Path_persist == 0] <- 0
unique(persists2 - LH2)

## [1]  0 -1

assign("persists3", as.numeric(Patho$LH_stage == 3))
persists3[Patho$Path_persist == 0] <- 0
unique(persists3 - LH3)

## [1]  0 -1
```

Next, we construct the plant death nodes for both cases and check the realization of both life and death of the plant. In this experiment, the pathogen can still live while the plant dies. Therefore, plant death forms another dependence group.

```r
assign("PD2_1", persists2)
PD2_1[Patho$PlantLives2 == 0] <- 0
assign("PD2_0", persists2)
```

```
PD2_0[Patho$PlantLives2 == 1] <- 0
unique(PD2_1 + PD2_0 - persists2)

## [1] 0

assign("PD3_1", persists3)
PD3_1[Patho$PlantLives2 == 0] <- 0
assign("PD3_0", persists3)
PD3_0[Patho$PlantLives2 == 1] <- 0
unique(PD3_1 + PD3_0 - persists3)

## [1] 0
```

In the above, we observe no plant death when a pathogen persists and $LH = 2$. If we proceed further with this unacknowledged, we will end up fitting a degenerate aster model. We can handle this in one of two ways:

1) We can remove either the persistence node or plant death node when $LH = 2$.

2) We can specify the relevant directions of recession relating corresponding to this occurrence. This is complicated. but it can be handled by `aster2` software.

We go with the second avenue. The aster graph depicted in Figure 1 reflects this decision. This aster graph also specifies the $LH1$ node corresponding to when a plant was infected but no pathogen growth was observed. We now construct the $GallWT$ nodes.

```
Patho$GallWT <- as.numeric(as.character(Patho$GallWT))
Patho$GallWT <- Patho$GallWT/1000
assign("GallWT2_1", as.numeric(Patho$GallWT))
GallWT2_1[PD2_1 == 0] <- 0
unique(as.numeric(GallWT2_1 > 0) - PD2_1)

## [1] 0

assign("GallWT2_0", as.numeric(Patho$GallWT))
GallWT2_0[PD2_0 == 0] <- 0
unique(as.numeric(GallWT2_0 > 0) - PD2_0)
```

```
## [1] 0

assign("GallWT3_1", as.numeric(Patho$GallWT))
GallWT3_1[PD3_1 == 0] <- 0
unique(as.numeric(GallWT3_1 > 0) - PD3_1)

## [1] 0

assign("GallWT3_0", as.numeric(Patho$GallWT))
GallWT3_0[PD3_0 == 0] <- 0
unique(as.numeric(GallWT3_0 > 0) - PD3_0)

## [1] 0
```

All of the $GallWT$ nodes need to have an associated variance node since the normal distribution is a two parameter exponential family. The $GallWT$ nodes that follow the occurrence pathogen persistence when $LH = 2$ are trimmed off of the graphical structure, see Figure 1. This is because no individuals progressed to this life stage and `aster2` requires such a trimming in order to function properly.

```
GallWT2_0.sd <- GallWT2_0^2
GallWT2_1.sd <- GallWT2_1^2
GallWT3_0.sd <- GallWT3_0^2
GallWT3_1.sd <- GallWT3_1^2
```

We now have all nodes of the aster graph depicted in Figure 1 specified and can begin creating objects useable by `aster2` functions. We first specify the variables that appear in the graphical structure of the aster model, see Figure 1.

```
vars <- c("I","LH1","LH2","LH3","persists2","PD2_0",
  "PD2_1","GallWT2_1",
  "GallWT2_1.sd",
  "persists3","PD3_0","PD3_1","GallWT3_0",
  "GallWT3_0.sd","GallWT3_1","GallWT3_1.sd")
test <- cbind(Patho$Infect,LH1,LH2,LH3,persists2,
  PD2_0,PD2_1,GallWT2_1,GallWT2_1.sd,
  persists3,PD3_0,PD3_1,GallWT3_0,GallWT3_0.sd,
  GallWT3_1,GallWT3_1.sd,
  Patho$InVir,Patho$Block,Patho$Fvert)
colnames(test)[!(colnames(test) %in% vars)] <-
  c("I","Invir","Block","Fvert")
test <- as.data.frame(test)
test$Fvert <- as.factor(test$Fvert)
test$Block <- as.factor(test$Block)
```

The observed values corresponding to the nodes in the graphical structure depicted in Figure 1 are now specified. We now specify the arrows and the distributions corresponding to these arrows. The names of the nodes were collected into a single character vector `vars`. The `pred` vector indicates which node in `vars` precedes the node of `vars` in question. The `group` vector indicates which nodes comprise a dependence group. The `code` vector provides which exponential family, in the `families` list, is being used to model a particular node. The `delta` vector specifies the directions of recession that are present. These are now specified.

```
pred <- c(0,1,1,1,3,5,5,7,7,4,10,10,11,11,12,12)
group <- c(0,0,2,3,0,0,6,0,8,0,0,11,0,13,0,15)
families <- list("bernoulli",fam.multinomial(3),
  fam.multinomial(2),"normal.location.scale")
code <- c(1,2,2,2,1,3,3,4,4,1,3,3,4,4,4,4)
delta <- rep(0, length(pred)) # direction of recession
delta[grepl("PD2_0",vars)] <- -1
```

We now make the `asterdata` object which allows us to fit models, compare models, and estimate aster model parameters.

```
data <- asterdata(data = test, vars = vars,
  pred = pred, group = group, code = code,
  families = families, delta = delta)
is.validasterdata(data)

## [1] TRUE
```

In order to estimate fitness (gall weight), the locations of the `GallWT` nodes within the dataframe are specified.

```r
infection <- as.numeric(grepl("I",data$redata$varb))
GallWT <- as.numeric(grepl("GallWT",data$redata$varb))
GallWT.sd <- as.numeric(grepl(".sd",data$redata$varb))
GallWT.mean <- GallWT - GallWT.sd
```

We have now successfully converted the raw data into an analyzable form using `aster2` software.

3 Model comparison and selection

Aster models are now fit and compared. Models are selected via a Rao test at size $\alpha = 0.05$. The Rao test statistic is given by

$$R = U(\hat{\beta}_{\text{null}})' I(\hat{\beta}_{\text{null}})^{-1} U(\hat{\beta}_{\text{null}}),$$

where $\hat{\beta}_{\text{null}}$ is the maximum likelihood estimate (MLE) of β under the null (smaller) model, and U and I are the score function and Fisher information matrix under the alternative (larger) model. The Rao test has a χ^2 reference distribution with degrees of freedom equal to the difference in terms between the two models. Backward selection is used to compare model. This procedure follows the same outline as Eck, et al. (2015 b) for comparing aster models with dependency groups. All quantities are calculated using functionality in the `aster2` package.

We start with the full model which contains linear and quadratic `Invir` terms, the `Fvert` and `Block` main effects, the `Fvert` and `Block` interaction term, and the `Fvert` interaction with the linear and quadratic `Invir` terms. Our largest model includes these terms as interactions with the fitness and infection nodes.

We will walk through the steps of our testing procedure and then construct a testing function that performs these steps. We will test whether or not the `Fvert` and `Block` interaction terms belong in our final model for our walk through. We will first compute the score function. The score function for our aster model is

$$M_{\text{alt}}^T Y - \nabla c(M_{\text{alt}} \hat{\beta}_{\text{null}}) \tag{1}$$

where $\hat{\beta}_{\text{null}}$ is the MLE for β using the null model. The left hand side of the subtraction in (1) is $\hat{\tau}_{\text{alt}} = M_{alt}^T Y$ where $\hat{\tau}_{\text{alt}}$ is the MLE of the submodel mean-value parameter vector under the alternative model (see Figure 2). The right hand side of

$$\begin{tikzcd}
\theta \arrow[r, shift left, "\text{aster transform}"] \arrow[d, shift left, "\xi_j = c'_j(\theta)"] & \varphi \arrow[l, shift left, "\text{inverse aster transform}"] \arrow[r, shift left] \arrow[d, shift left, "\mu = \nabla c(\varphi)"] & \beta \arrow[l, shift left, "\varphi = M\beta"] \arrow[d, shift left, "\tau = \nabla c_{sub}(\beta)"] \\
\xi \arrow[u, shift left] \arrow[r, shift left, "\text{multiplication}"] & \mu \arrow[u, shift left] \arrow[l, shift left, "\text{division}"] \arrow[r, shift left, "\tau = M^T \mu"] & \tau \arrow[u, shift left] \arrow[l, shift left]
\end{tikzcd}$$

Figure 2: A depiction of the transformations necessary to change parameterizations. Unlabeled arrows require `aster` or `aster2` software to perform the transformation.

the subtraction in (1) is a bit more complicated. The model matrix M_{alt} has more columns than M_{null} because it corresponds to a larger model with more parameters specified. Thus $\hat{\beta}_{\text{null}}$ is the MLE of the submodel canonical parameter vector under the null model with additional zero entries that align with the columns of M_{alt} that are not in M_{null}. If the kth column of M_{alt} is not in M_{null}, then the kth entry of $\hat{\beta}_{\text{null}}$ is 0.

We now obtain the submodel mean-value parameter vector τ_{alt} for the alternative model.

```
modmat.alt <- model.matrix(resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2) + I(Fvert:Block)) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2) + I(Fvert:Block)),
  data = data$redata)
tau.alt <- crossprod(modmat.alt, data$redata$resp)
```

The aster submodel mean-value parameter vector for the null model is now estimated. This is done in three steps. The first step is to estimate τ_{null} with $M_{\text{null}}^T Y$ where M_{null} is the model matrix corresponding to the null aster submodel with the `Fvert` and `Block` interaction removed.

```
modmat.null <- model.matrix(resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)),
  data = data$redata)
tau.null <- crossprod(modmat.null, data$redata$resp)
```

We then map this estimate of τ_{null} to the canonical parameterization using `transformUnconditional` with the model matrix M_{null} specified, see Figure 2.

```
beta.null <- transformUnconditional(tau.null,
  modmat.null, data, from = "tau", to = "beta")
```

Finally, we map back to the mean-parameterization using `transformUnconditional` with the model matrix M_{alt} specified. In order to perform this mapping, a number of 0s equal to the number of degrees of freedom in the Rao test are added to the estimated canonical parameter vector obtained above. This is equivalent to the hypothesis that the null model is preferred.

```
foo <- grepl("I.Fvert.Block", colnames(modmat.alt))
bar <- rep(0, length(foo))
bar[!foo] <- beta.null
bar[foo] <- 0
tau.null <- transformUnconditional(bar, modmat.alt,
  data, from = "beta", to = "tau")
```

We now have the quantities required to perform the Rao test. We first build the score function and then calculate inverse Fisher information.

```
score <- tau.alt - tau.null
Fisher.null <- jacobian(bar, data,
  transform = "unconditional", from = "beta",
  to = "tau", modmat = modmat.alt)
```

The Rao test statistic is now calculated and the p-value for this test is computed.

```
df <- nrow(tau.alt) - length(beta.null)
Rao <- t(score) %*% ginv(Fisher.null) %*% score
pval1 <- as.numeric(pchisq(Rao, df = df, lower = FALSE))
pval1

## [1] 0.9991659
```

We reject the hypothesis that the alternative (larger) model fits the data better than the smaller model at the $\alpha = 0.05$ level. The `Fvert` and `Block` interaction terms are removed from our final model. We now streamline this testing procedure in the `Rao_test` function below.

```
Rao_test <- function(formula.null, formula.alt,
  data.null, data.alt)
{
  infection <- GallWT.mean <- NULL
  if(grepl("infection", formula.null)) infection <-
    as.numeric(grepl("I", data.null$redata$varb))
  if(grepl("GallWT", formula.null)){
    GallWT <- as.numeric(grepl("GallWT",
      data.null$redata$varb))
    GallWT.sd <- as.numeric(grepl(".sd",
      data.null$redata$varb))
    GallWT.mean <- GallWT - GallWT.sd
  }
  modmat.null <- model.matrix(as.formula(formula.null),
    data = data.null$redata)
  tau.null <- crossprod(modmat.null,
    data.null$redata$resp)
  beta.null <- transformUnconditional(tau.null,
    modmat.null, data.null, from = "tau", to = "beta")

  infection <- GallWT.mean <- NULL
  if(grepl("infection", formula.alt)) infection <-
    as.numeric(grepl("I", data.alt$redata$varb))
  if(grepl("GallWT", formula.alt)){
    GallWT <- as.numeric(grepl("GallWT",
      data.alt$redata$varb))
    GallWT.sd <- as.numeric(grepl(".sd",
      data.alt$redata$varb))
    GallWT.mean <- GallWT - GallWT.sd
  }
  modmat.alt <- model.matrix(as.formula(formula.alt),
    data = data.alt$redata)
  tau.alt <- crossprod(modmat.alt,
    data.alt$redata$resp)

  df <- nrow(tau.alt) - length(beta.null)
  foo <- !(colnames(modmat.alt) %in%
    colnames(modmat.null))
  bar <- rep(0, nrow(tau.alt))
  bar[!foo] <- beta.null
  tau.null <- transformUnconditional(bar,
    modmat.alt, data.null, from = "beta", to = "tau")
  score <- tau.alt - tau.null

  Fisher.null <- jacobian(bar, data.null,
    transform = "unconditional", from = "beta",
    to = "tau", modmat = modmat.alt)
  Rao <- t(score) %*% ginv(Fisher.null) %*% score
  pval <- pchisq(Rao, df = df, lower = FALSE)
  as.numeric(pval)

}

## data.null and data.alt are declared
data.null <- data.alt <- data
```

We now test whether or not the Block factor belongs in the final model. We first verify that our Rao_test function is doing the right thing by determining whether or not it can replicate the results of our walk though test.

```
## first test the Block and Fvert interaction
formula.null <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir +  Fvert*I(Invir^2)) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2))"
formula.alt <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  infection:(I(Fvert:Block)) +
  GallWT.mean:(I(Fvert:Block))"
pval2 <- Rao_test(formula.null, formula.alt,
  data.null, data.alt)
```

We can see that the `Rao_test` function is working properly.

```
all.equal(pval1, pval2)

## [1] TRUE
```

We now show that `Block` and `Fvert` interactions do not belong in the final model when testing first at the infection node and then at the gall weight node.

```
## we test the Fvert and Block interaction at the
## infection node
formula.null <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  GallWT.mean:(Block + Fvert + I(Fvert:Block) +
    Invir + I(Invir^2) + Fvert*Invir +
    Fvert*I(Invir^2))"
formula.alt <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  GallWT.mean:(Block + Fvert + I(Fvert:Block) +
    Invir + I(Invir^2) + Fvert*Invir +
    Fvert*I(Invir^2)) +
  infection:(I(Fvert:Block))"
Rao_test(formula.null, formula.alt, data.null, data.alt)

## [1] 0.9612011
## we test the Fvert and Block interaction at the
## gall weight nodes
formula.null <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2))"
formula.alt <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  GallWT.mean:(I(Fvert:Block))"
Rao_test(formula.null, formula.alt, data.null, data.alt)

## [1] 0.9755211
```

We now proceed with the test of whether or not the `Block` factor belongs in the final model.

```
## now test the Block main effect at the infection node
formula.null <- "resp ~ varb +
  infection:(Fvert + Invir + I(Invir^2) + Fvert*Invir +
    Fvert*I(Invir^2)) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2))"
formula.alt <- "resp ~ varb +
  infection:(Fvert + Invir + I(Invir^2) + Fvert*Invir +
    Fvert*I(Invir^2)) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  infection:Block"
Rao_test(formula.null, formula.alt, data.null, data.alt)

## [1] 1.484154e-05
## now test the Block main effect at gall weight nodes
formula.null <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  GallWT.mean:(Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2))"
formula.alt <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  GallWT.mean:(Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  GallWT.mean:Block"
Rao_test(formula.null, formula.alt, data.null, data.alt)

## [1] 0.01391618
```

Therefore, the `Block` and `Fvert` interaction should not be included in the final model, but the `Block` main effect belongs at both nodes. We now test whether or not the `Fvert` and `Invir` interaction terms should be included in the final model.

```
## now test the Fvert and quadratic Invir interaction
## at the infection node
formula.null <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2))"
formula.alt <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  infection:(Fvert*I(Invir^2))"
Rao_test(formula.null, formula.alt, data.null, data.alt)

## [1] 0.7373819
## now test the Fvert and quadratic Invir interaction
## at the gall weight nodes
formula.null <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir)"
formula.alt <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir) +
  GallWT.mean:(Fvert*I(Invir^2))"
Rao_test(formula.null, formula.alt, data.null, data.alt)

## [1] 0.9656977
```

```
## now test the Fvert and Invir interaction at
## the infection node
formula.null <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2)) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir)"
formula.alt <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2)) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir) +
  infection:(Fvert*Invir)"
Rao_test(formula.null, formula.alt, data.null, data.alt)

## [1] 0.730041
## now test the Fvert and Invir interaction at
## the gall weight nodes
formula.null <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2)) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2))"
formula.alt <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2)) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2)) +
  GallWT.mean:(Fvert*Invir)"
Rao_test(formula.null, formula.alt, data.null, data.alt)

## [1] 0.9185152
```

Therefore, the `Fvert` and `Invir` interaction terms should not be included in the final model. We now test whether or not the quadratic `Invir` term belongs in our final model.

```
## now test the quadratic Invir term at the infection
## node
formula.null <- "resp ~ varb +
  infection:(Block + Fvert + Invir) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2))"
formula.alt <- "resp ~ varb +
  infection:(Block + Fvert + Invir) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2)) +
  infection:(I(Invir^2))"
Rao_test(formula.null, formula.alt, data.null, data.alt)

## [1] 0.05257471
## now test the quadratic Invir term at the gall weight
## nodes
formula.null <- "resp ~ varb +
  infection:(Block + Fvert + Invir) +
  GallWT.mean:(Block + Fvert + Invir)"
formula.alt <- "resp ~ varb +
  infection:(Block + Fvert + Invir) +
  GallWT.mean:(Block + Fvert + Invir) +
  GallWT.mean:(I(Invir^2))"
Rao_test(formula.null, formula.alt, data.null, data.alt)

## [1] 1.14161e-06
```

We remove the quadratic `Invir` term at the infection node and keep the quadratic `Invir` term at the gall weight node. We now test the linear `Invir` at the infection node.

```
## now test the linear Invir term at the infection node
formula.null <- "resp ~ varb +
  infection:(Block + Fvert) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2))"
formula.alt <- "resp ~ varb +
  infection:(Block + Fvert) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2)) +
  infection:(Invir)"
Rao_test(formula.null, formula.alt, data.null, data.alt)

## [1] 8.310293e-57
```

The linear `Invir` term at the infection node remains in the final model. We now test whether or not the `Fvert` terms belong to the final model.

```
## now test the Fvert term at the infection node
formula.null <- "resp ~ varb +
  infection:(Block + Invir) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2))"
formula.alt <- "resp ~ varb +
  infection:(Block + Invir) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2)) +
  infection:(Fvert)"
Rao_test(formula.null, formula.alt, data.null, data.alt)

## [1] 1.05324e-13
```

```
## now test the Fvert term at the gall weight nodes
formula.null <- "resp ~ varb +
  infection:(Block + Fvert + Invir) +
  GallWT.mean:(Block + Invir + I(Invir^2))"
formula.alt <- "resp ~ varb +
  infection:(Block + Fvert + Invir) +
  GallWT.mean:(Block + Invir + I(Invir^2)) +
  GallWT.mean:(Fvert)"
Rao_test(formula.null, formula.alt, data.null, data.alt)

## [1] 0.8493302
```

We keep the `Fvert` term at the infection node and remove the `Fvert` term at the gall weight node. In accordance with following model hierarchy, there are no more terms to be tested. Our final model includes the `Block` factor and the linear `Invir` terms at both the infection and fitness nodes. Our final model also includes a quadratic `Invir` term at the fitness node and the `Fvert` factor at the infection node.

Preliminary analyses (and background knowledge) suggest that the second and third levels of the `Fvert` factor could be combined into a single level. We now test whether or not this is the case. The hypothesis testing procedure corresponding to this test is slightly different than the preceding tests. We first combine the second and third levels together in a new dataframe and then create a new `asterdata` object corresponding to the changes made to the `Fvert` factor.

```
test.null <- test
test.null <- test.null %>%
  mutate(Fvert = as.factor(ifelse(Fvert != 0, 1, 0)))
data.null <- asterdata(data = test.null, vars = vars,
  pred = pred, group = group, code = code,
  families = families, delta = delta)
is.validasterdata(data.null)

## [1] TRUE
```

We now test whether or not the `Fvert` levels should be combined.

```
## Test whether or not the second and third levels of
## Fvert should be combined into a single level
formula.null <- "resp ~ varb +
  infection:(Block + Fvert + Invir) +
  GallWT.mean:(Block + Invir + I(Invir^2))"
formula.alt <- "resp ~ varb +
  infection:(Block + Fvert + Invir) +
  GallWT.mean:(Block + Invir + I(Invir^2))"
Rao_test(formula.null, formula.alt,
  data.null = data.null, data.alt = data)

## [1] 1.610078e-25
```

Therefore, the second and third levels of `Fvert` should not be combined into a single level. We close this section with a table (Table 1) that summarises the results of the backward selection procedure that we used to arrive at our final model.

4 Fitness landscape

In order to build a fitness landscape, we create 50 hypothetical individuals possessing distinct values of `Invir` that were suggested by our observed data. The fitness landscape will consist of 2 lines, one for both levels of the `Fvert` factor. The hypothetical individuals used to form each plot posses the same 50 `Invir` values for a total of 100 individuals. The dataset for these hypothetical individuals

model	infection node terms	gall weight node terms	p-value
null	Fvert, Invir, Invir^2, Fvert∗Invir, Fvert∗Invir^2	Block, Fvert, Invir, Invir^2, Fvert∗Invir, Fvert∗Invir^2	
alt	**Block**, Fvert, Invir, Invir^2, Fvert∗Invir, Fvert∗Invir^2	Block, Fvert, Invir, Invir^2, Fvert∗Invir, Fvert∗Invir^2	1.48e-5
null	Block, Fvert, Invir, Invir^2, Fvert∗Invir, Fvert∗Invir^2	Fvert, Invir, Invir^2, Fvert∗Invir, Fvert∗Invir^2	
alt	Block, Fvert, Invir, Invir^2, Fvert∗Invir, Fvert∗Invir^2	**Block**, Fvert, Invir, Invir^2, Fvert∗Invir, Fvert∗Invir^2	0.014
null	Block, Fvert, Invir, Invir^2, Fvert∗Invir	Block, Fvert, Invir, Invir^2, Fvert∗Invir, Fvert∗Invir^2	
alt	Block, Fvert, Invir, Invir^2, Fvert∗Invir, **Fvert∗Invir**^{2}	Block, Fvert, Invir, Invir^2, Fvert∗Invir, Fvert∗Invir^2	0.74
null	Block, Fvert, Invir, Invir^2, Fvert∗Invir	Block, Fvert, Invir, Invir^2, Fvert∗Invir	
alt	Block, Fvert, Invir, Invir^2, Fvert∗Invir	Block, Fvert, Invir, Invir^2, Fvert∗Invir, **Fvert∗Invir**^{2}	0.97
null	Block, Fvert, Invir, Invir^2	Block, Fvert, Invir, Invir^2, Fvert∗Invir	
alt	Block, Fvert, Invir, Invir^2, **Fvert∗Invir**	Block, Fvert, Invir, Invir^2, Fvert∗Invir	0.73
null	Block, Fvert, Invir, Invir^2	Block, Fvert, Invir, Invir^2	
alt	Block, Fvert, Invir, Invir^2	Block, Fvert, Invir, Invir^2, **Fvert∗Invir**	0.92
null	Block, Fvert, Invir	Block, Fvert, Invir, Invir^2	
alt	Block, Fvert, Invir, **Invir**^{2}	Block, Fvert, Invir, Invir^2	0.053
null	Block, Fvert, Invir	Block, Fvert, Invir	
alt	Block, Fvert, Invir	Block, Fvert, Invir, **Invir**^{2}	1.14e-6
null	Block, Fvert,	Block, Fvert, Invir, Invir^2	
alt	Block, Fvert, **Invir**	Block, Fvert, Invir, Invir^2	≈ 0
null	Block, Invir	Block, Fvert, Invir, Invir^2	
alt	Block, **Fvert**, Invir	Block, Fvert, Invir, Invir^2	≈ 0
null	Block, Fvert, Invir	Block, Invir, Invir^2	
alt	Block, Fvert, Invir	Block, **Fvert**, Invir, Invir^2	0.85

Table 1: Summary of our model selection procedure. Our final model includes the `Block` factor and the linear `Invir` terms at both the infection and gall weight nodes. Our final model also includes a quadratic `Invir` term at the gall weight node and the `Fvert` factor at the infection node.

is called `cand`, and it is constructed below. Note that we will only display fitness landscapes for the first level of the `Block` factor. This variable is not of scientific interest, is only relevant to this experiment, and only has a linear effect that does not interact with any other quntities of scientific interest. The latter point means that only one level of the `Block` factor is necessary for interpreting the relation between estimated expected fitness and `Invir` and `Fvert`. That being said, we will include both levels of `Block` when constructing fitness landscapes since statistical quantities (canonical parameter vectors, model matrices, etc) include both levels of `Block`.

```
nInvir <- 50
nFvertlevels <- 3
nBlock <- 2
lwr <- round(min(unique(test$Invir)))
upr <- round(max(unique(test$Invir)))
cand.Invir <- rep(seq(from = lwr, to = upr,
  length = nInvir), nFvertlevels)
cand.Fvert <- rep(seq(from = 0, to = nFvertlevels - 1),
  each = nInvir)
cand <- cbind(cand.Fvert, cand.Invir,
  rep(1:nBlock, each = nInvir * nFvertlevels))
cand <- as.data.frame(cand)
colnames(cand) <- c("Fvert","Invir","Block")
cand$Block <- as.factor(cand$Block)
cand$root <- 1
blah <- test[1:(nFvertlevels*nInvir*nBlock),
  colnames(test) %in% vars]
cand <- cbind(cand, blah)
cand <- as.data.frame(cand)
cand$Fvert <- as.factor(cand$Fvert)
```

We now construct a corresponding `asterdata` object and specify which nodes relate to Darwinian fitness.

```
data.mnew <- asterdata(cand, vars = vars,
  pred = pred, group = group, code = code,
  delta = delta, families = families)
GallWT.mnew <- as.numeric(grepl("GallWT",
  data.mnew$redata$varb))
GallWT.mnew.sd <- as.numeric(grepl(".sd",
  data.mnew$redata$varb))
GallWT.mnew.mean <- GallWT.mnew -
  GallWT.mnew.sd
infection.mnew <- as.numeric(grepl("I",
  data.mnew$redata$varb))
data.mnew$redata <- transform(data.mnew$redata,
  GallWT.mean = GallWT.mnew.mean,
  GallWT.sd = GallWT.mnew.sd,
  infection = infection.mnew)
```

The model matrix using the aster model formula for our final aster model which corresponds to the hypothetical individuals, M_{new} is now constructed. This model matrix plays an important role in estimating expected Darwinian fitness.

```r
formula.final <- "resp ~ varb +
  infection:(Block + Fvert + Invir) +
  GallWT.mean:(Block + Invir + I(Invir^2))"
modmat.mnew <- model.matrix(as.formula(formula.final),
  data = data.mnew$redata)
```

Expected Darwinian fitness is now estimated for every hypothetical individual. The parameterization of expected Darwinian fitness is a function, in this case a sum, of aster model mean-value parameter values, see μ in Figure 2. The μ parameters are a function of the ϕ parameters and we estimate ϕ with $\hat{\phi}_{\text{new}} = M_{\text{new}}\hat{\beta}$ where $\hat{\beta}$ is the estimated aster submodel canonical parameter vector for our final model. The `transformSaturated` function in the `aster2` package maps $\hat{\phi}_{\text{new}}$ to $\hat{\mu}_{\text{new}}$.

```r
## now get mean-value estimates
infection <- GallWT.mean <- NULL
if(grepl("infection", formula.final)) infection <-
  as.numeric(grepl("I", data$redata$varb))
if(grepl("GallWT", formula.final)){
  GallWT <-
    as.numeric(grepl("GallWT",data$redata$varb))
  GallWT.sd <- as.numeric(grepl(".sd",data$redata$varb))
  GallWT.mean <- GallWT - GallWT.sd
}
modmat.data <- model.matrix(as.formula(formula.final),
  data = data$redata)
tau.hat <- crossprod(modmat.data, data$redata$resp)
beta.hat <- transformUnconditional(tau.hat,
  modmat.data, data, from = "tau", to = "beta")
stat <- modmat.mnew %*% beta.hat
mus <- transformSaturated(stat, data.mnew,
  from = "phi", to = "mu")
```

Expected Darwinian fitness is estimated by summing over the components of $\hat{\mu}_{\text{new}}$ which correspond to spore mass.

```r
ind <- which(GallWT.mnew.mean == 1)
qux <- matrix(mus[ind],
  nrow = nFvertlevels*nInvir*nBlock, ncol = 3)
sums <- as.numeric(apply(qux, 1, sum))
```

We now estimate 95% confidence bands for estimated expected spore mass. These confidence bands are of the form

$$A\hat{\mu} \pm 1.96 * \hat{\text{se}}(A\hat{\mu})$$

where the matrix A specifies the sum of components of $\hat{\mu}$ that correspond to *GallWT* nodes. In other words, $A\hat{\mu}$ is estimated expected Darwinian fitness. We already have computed estimated expected Darwinian fitness, the code below finds the estimated variance of the components of $\hat{\mu}$ that correspond to estimated expected Darwinian fitness. This variance is calculated using the `jacobian` function in the `aster2` package. In this calculation we consider the map ϕ to μ in Figure 2. The indices used to create `fubar` are the indices of the components of $\hat{\mu}$ that correspond to estimated expected Darwinian fitness for the hypothetical individuals.

```r
mus.jacob <- jacobian(stat, data.mnew,
  transform = "saturated", from = "phi", to = "mu",
  modmat = modmat.mnew)
fubar <- mus.jacob[ind,ind]
```

We now estimate the standard error of estimated expected Darwinian fitness by building A. There are 900 values of $\hat{\mu}$ that correspond to estimated expected Darwinian fitness of the hypothetical individuals. There are 3 estimated expected *GallWT* parameters for all 300 hypothetical individuals. Thus $A \in \mathbb{R}^{300\times 900}$ where the placement of a 1 in row i of A corresponds to one of the entries of $\hat{\mu}$ that is an estimated expected value of *GallWT* for individual i.

```r
A <- matrix(0, nrow = nFvertlevels*nInvir*nBlock,
  ncol = 3*nFvertlevels*nInvir*nBlock)
for(i in 1:(nFvertlevels*nInvir*nBlock)){
  A[i,i] <- A[i,i+nFvertlevels*nInvir*nBlock] <-
    A[i,i+2*nFvertlevels*nInvir*nBlock] <- 1
}
```

```
barbaz <- A %*% fubar %*% t(A)
var <- diag(barbaz)
se <- sqrt(var) / sqrt(nrow(test))
crit <- qnorm(1-0.025) * se
```

The matrix `barbaz` in the above code is the estimated covariance matrix of estimated expected Darwinian fitness. These derivations are supported by the Delta method where the asymptotic distribution of estimated expected Darwinian fitness is given by

$$\sqrt{n}\left(A\hat{\mu} - A\mu\right) \longrightarrow N\left(0,\ A\Sigma_\mu A^T\right)$$

where `barbaz` estimates $A\Sigma_\mu A^T$ and Σ_μ is estimated by `fubar`. The code below creates the fitness landscapes with confidence bands.

```
cand_block1 <- cand %>% mutate(sums, sums,
  lower = sums - crit, upper = sums + crit) %>%
  filter(Block == 1)
ggplot(data = cand_block1) +
  geom_line(mapping = aes(x = Invir, y = sums,
    group=Fvert, color=Fvert)) +
  geom_ribbon(mapping = aes(x = Invir, ymin = lower,
    ymax = upper, group=Fvert, color=Fvert),
    alpha = 0.5) +
  labs(x="Invir", y="Expected spore mass",
    color="Fvert", title = "Fitness landscape") +
  facet_wrap(~Block) +
  theme_minimal() +
  scale_x_continuous(minor_breaks = NULL) +
  scale_y_continuous(minor_breaks = NULL)
```

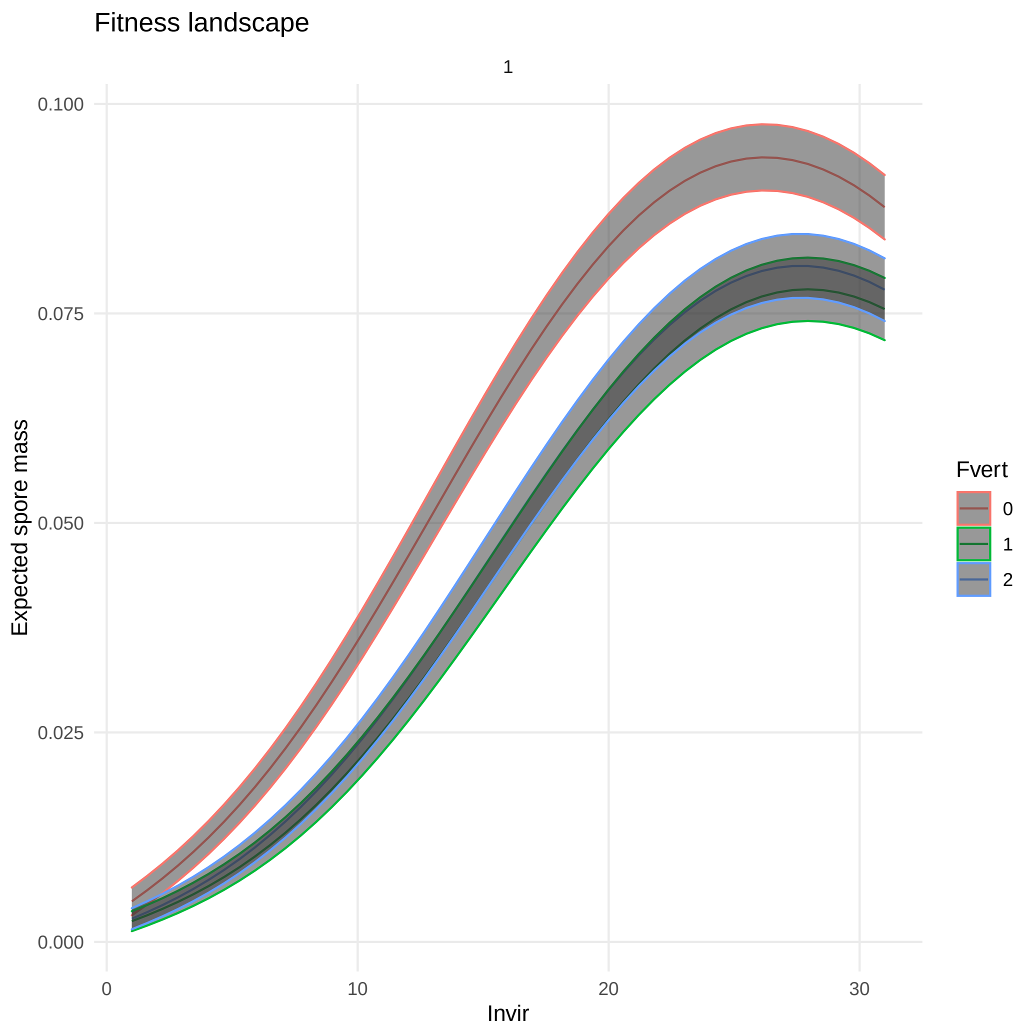

Figure 3: The fitness landscape with standard errors corresponding to our final model for hypothetical individuals possessing `Invir` values suggested by the original data. The model contains the following terms at the gall weight (fitness) node: a linear term for `Block` and `Invir`, and a quadratic term for `Invir`. The model contains the following terms at the infection node: a linear term for `Block`, `Fvert` and `Invir`. The confidence bands are at the 95% confidence level.

5 Additional fitness landscapes

We now streamline the process of constructing a fitness landscape with a fitting function `fitness_landscape` using `ggplot` functionality (the code is included in the .Rnw file). We use the `fitness_landscape` function to view some other (possibly) interesting fitness landscapes.

```
formula1 <- "resp ~ varb +
  infection:(Block + Fvert + Invir) +
  GallWT.mean:(Block + Invir + I(Invir^2))"
fitness_landscape(formula1, data = data, mat = test,
  nInvir = 50, alpha = 0.05)
```

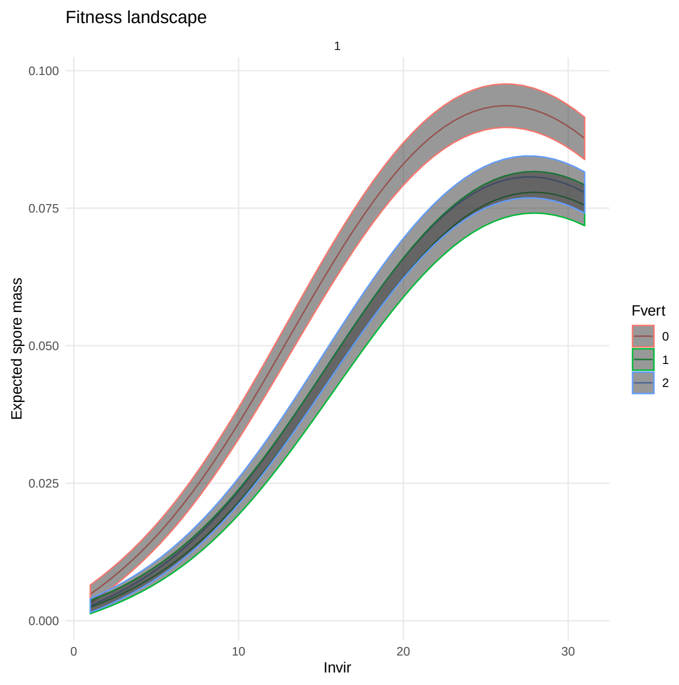

Figure 4: Same fitness landscape as Figure 3, except that it is produced using the `fitness_landscape` function.

```
formula2 <- "resp ~ varb +
  infection:(Fvert + Invir) +
  GallWT.mean:(Invir + I(Invir^2))"
fitness_landscape(formula2, data = data, mat = test,
  nInvir = 50, alpha = 0.05)
```

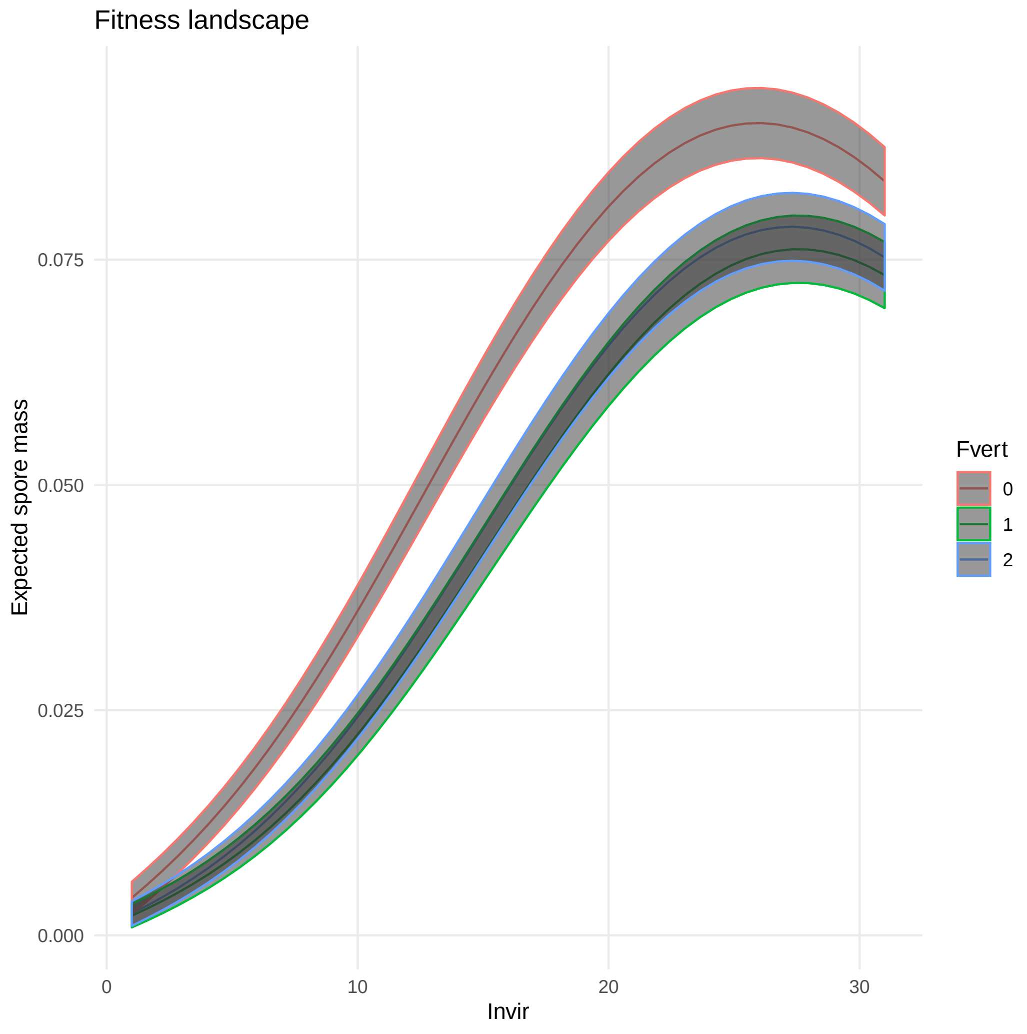

Figure 5: Fitness landscape corresponding to the final model with the `Block` factor excluded.

```
formula3 <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2))"
fitness_landscape(formula3, data = data, mat = test,
  nInvir = 50, alpha = 0.05)
```

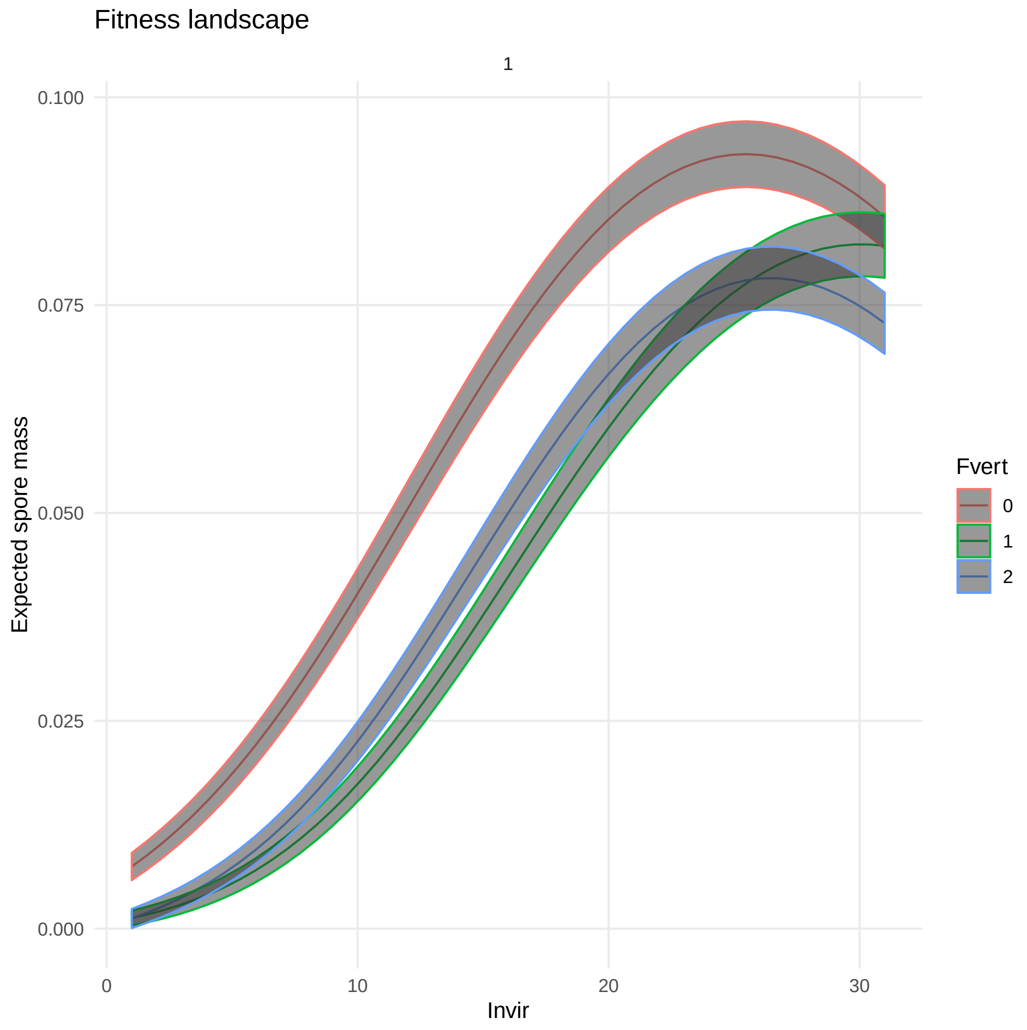

Figure 6: Fitness landscape for full model with the `Block` and `Fvert` interaction terms excluded.

5.1 Fitness landscape for gall weight nodes

We now consider aster models for terms at the gall weight node only. We first perform a backwards selection procedure similar to the one above. We then produce the fitness landscape corresponding to that final model.
The hypothesis tests are displayed below.

```
## lack of evidence for larger model: remove Fvert:Block
data.null <- data.alt <- data
formula.null <- "resp ~ varb +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2))"
formula.alt <- "resp ~ varb +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  GallWT.mean:(I(Fvert:Block))"
pval_gall_full <- Rao_test(formula.null, formula.alt,
  data.null, data.alt)
pval_gall_full

## [1] 0.9175423

## lack of evidence for larger model: remove Fvert*I(Invir^2)
formula.null <- "resp ~ varb +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir)"
formula.alt <- "resp ~ varb +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir) +
  GallWT.mean:(Fvert*I(Invir^2))"
pval_gall_fInv2 <- Rao_test(formula.null, formula.alt,
  data.null, data.alt)
pval_gall_fInv2

## [1] 0.6361912
## lack of evidence for larger model: remove Fvert*Invir
formula.null <- "resp ~ varb +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2))"
formula.alt <- "resp ~ varb +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2)) +
  GallWT.mean:(Fvert*Invir)"
pval_gall_fInv <- Rao_test(formula.null, formula.alt,
  data.null, data.alt)
pval_gall_fInv

## [1] 0.1632702

## evidence favors larger model: keep Invir^2
formula.null <- "resp ~ varb +
  GallWT.mean:(Block + Fvert + Invir)"
formula.alt <- "resp ~ varb +
  GallWT.mean:(Block + Fvert + Invir) +
  GallWT.mean:(I(Invir^2))"
pval_gall_Inv2 <- Rao_test(formula.null, formula.alt,
  data.null, data.alt)
pval_gall_Inv2

## [1] 2.534872e-08

## evidence favors larger model: keep Fvert
formula.null <- "resp ~ varb +
  GallWT.mean:(Block + Invir + I(Invir^2))"
formula.alt <- "resp ~ varb +
  GallWT.mean:(Block + Invir + I(Invir^2)) +
  GallWT.mean:(Fvert)"
pval_gall_f <- Rao_test(formula.null, formula.alt,
  data.null, data.alt)
pval_gall_f

## [1] 0.0002391985
## evidence favors larger model: remove block factor
formula.null <- "resp ~ varb +
  GallWT.mean:(Fvert + Invir + I(Invir^2))"
formula.alt <- "resp ~ varb +
  GallWT.mean:(Fvert + Invir + I(Invir^2)) +
  GallWT.mean:(Block)"
pval_gall_b <- Rao_test(formula.null, formula.alt,
  data.null, data.alt)
pval_gall_b

## [1] 0.4568961
```

The above model selection procedure is summarised in Table 2 below.

model	gall weight node terms	p-value
null	Block, Fvert, Invir, Invir^2, Fvert$*$Invir, Fvert$*$Invir^2	
alt	Block, Fvert, Invir, Invir^2, Fvert$*$Invir, Fvert$*$Invir^2, **Block$*$Fvert**	0.92
null	Block, Fvert, Invir, Invir^2, Fvert$*$Invir	
alt	Block, Fvert, Invir, Invir^2, Fvert$*$Invir, **Fvert$*$Invir^2**	0.64
null	Block, Fvert, Invir, Invir^2	
alt	Block, Fvert, Invir, Invir^2, **Fvert$*$Invir**	0.16
null	Block, Fvert, Invir	
alt	Block, Fvert, Invir, **Invir^2**	2.53e-8
null	Block, Invir, Invir^2	
alt	Block, **Fvert**, Invir, Invir^2	0.00024
null	Fvert, Invir, Invir^2	
alt	**Block**, Fvert, Invir, Invir^2	0.46

Table 2: Summary of our model selection procedure. Our final model includes main effects for `Fvert` and `Invir`, and a quadratic term for `Invir`.

We now produce the fitness landscape corresponding to the final model above that includes main effects for `Fvert` and `Invir`, and a quadratic term for `Invir`.

```
formula_gall <- "resp ~ varb +
  GallWT.mean:(Fvert + Invir + I(Invir^2))"
fitness_landscape(formula_gall, data = data, mat = test,
  nInvir = 50, alpha = 0.05)
```

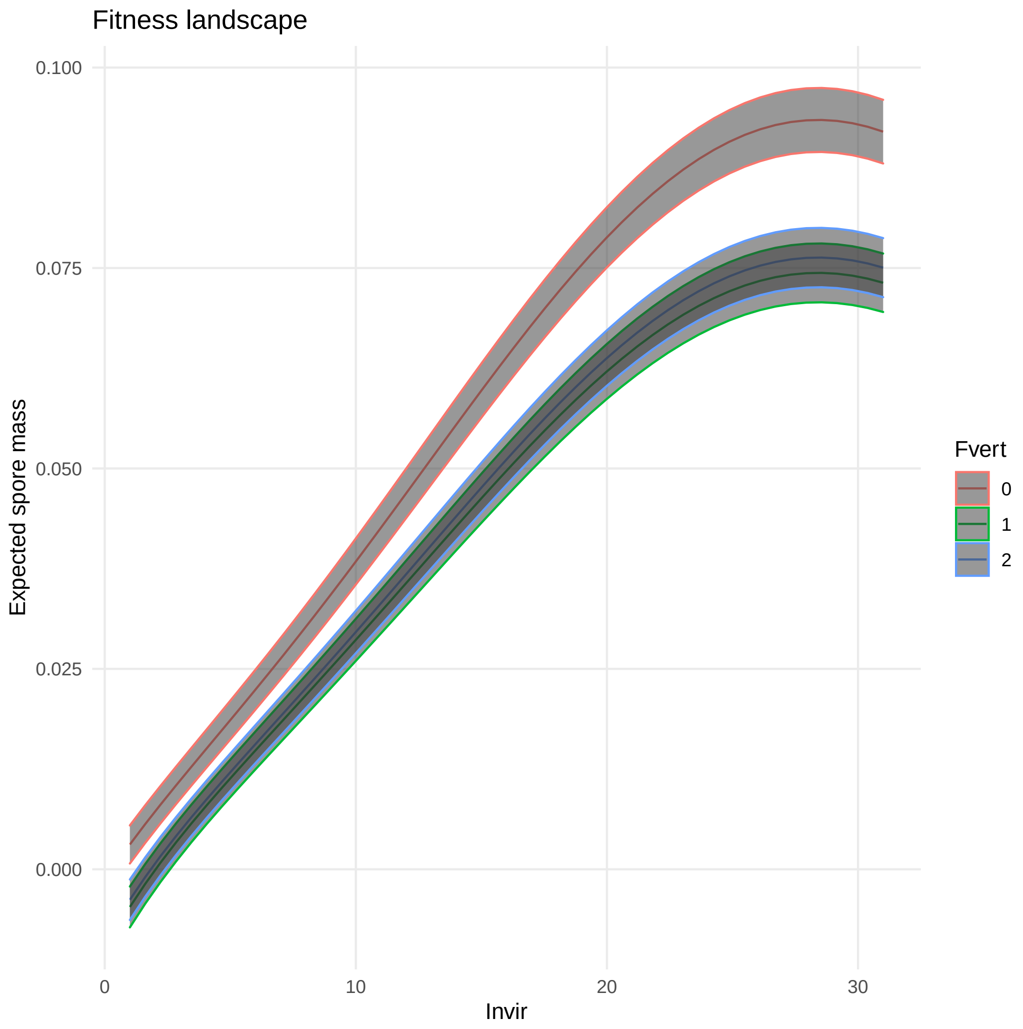

Figure 7: Fitness landscape for the model with a linear term for `Fvert` and a linear and quadratic term for `Invir`. This model only has terms for the gall weight nodes.

5.2 Fitness landscapes with 2 level `Fvert` factor

We make similar fitness landscapes corresponding to an analysis that combines the second and third levels of `Fvert`.

```
test2 <- test
test2 <- test2 %>% mutate(Fvert =
  as.factor(ifelse(Fvert != 0, 1, 0)))
data2 <- asterdata(data = test2, vars = vars,
  pred = pred, group = group, code = code,
  families = families, delta = delta)
is.validasterdata(data2)

## [1] TRUE
```

The fitness landscapes begin on the next page.

```
formula1 <- "resp ~ varb +
  infection:(Block + Fvert + Invir) +
  GallWT.mean:(Block + Invir + I(Invir^2))"
fitness_landscape(formula1, data = data2, mat = test2,
  nInvir = 50, alpha = 0.05)
```

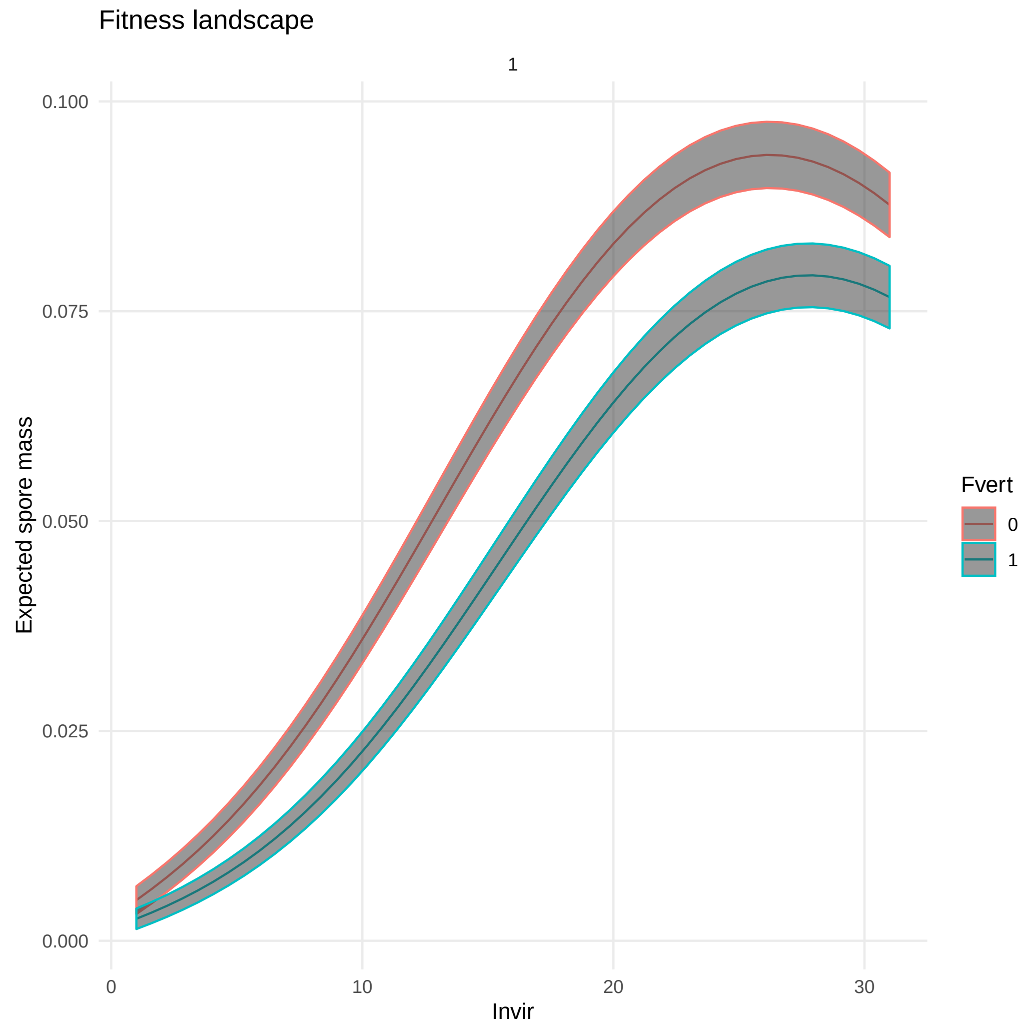

Figure 8: Fitness landscape corresponding to the final model where the second and third levels of `Fvert` are combined.

```
formula2 <- "resp ~ varb +
  infection:(Fvert + Invir) +
  GallWT.mean:(Invir + I(Invir^2))"
fitness_landscape(formula2, data = data2, mat = test2,
  nInvir = 50, alpha = 0.05)
```

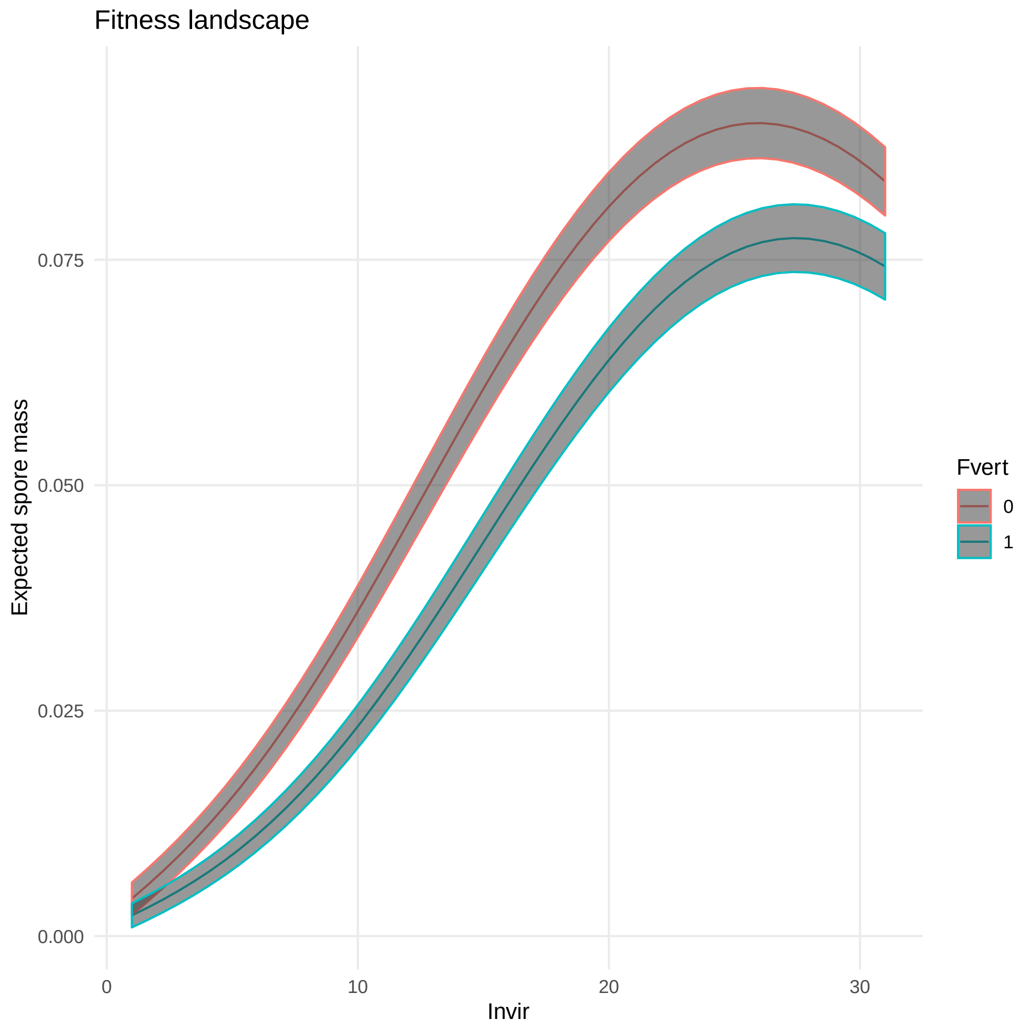

Figure 9: Fitness landscape for the final model with the `Block` factor excluded and the second and third levels of `Fvert` are combined.

```r
formula3 <- "resp ~ varb +
  infection:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2)) +
  GallWT.mean:(Block + Fvert + Invir + I(Invir^2) +
    Fvert*Invir + Fvert*I(Invir^2))"
fitness_landscape(formula3, data = data2, mat = test2,
  nInvir = 50, alpha = 0.05)
```

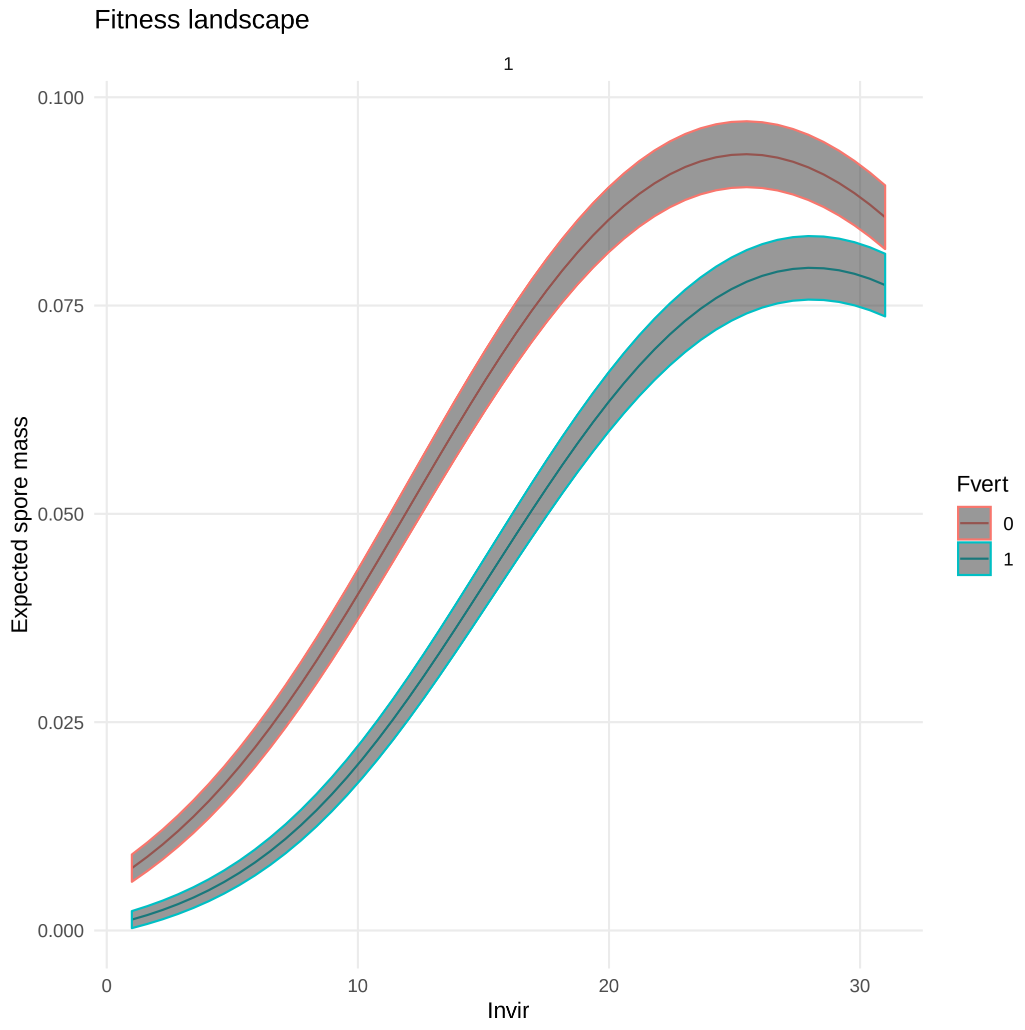

Figure 10: Fitness landscape for the full model with the `Block` and `Fvert` interaction terms excluded and the second and third levels of `Fvert` are combined.

6 Aster analysis without $LH2$ nodes

In this section, we fit the aster model corresponding to the graphical structure seen in Figure 1 with the $LH2$ node and all successor nodes excluded. The same variables used in the final aster model of the previous section are used to fit this model. The same hypothetical individuals used to estimate expected Darwinian fitness in the previous analysis are used here. Estimated expected Darwinian fitness for the hypothetical individuals using the aster model with the $LH2$ nodes excluded is depicted in Figure 11. Figure 12 compares estimated Darwinian fitness for the two aster models of interest. The dotted lines and solid lines correspond to estimates of Darwinian fitness using the aster model with the $LH2$ nodes excluded and the full aster model respectively.

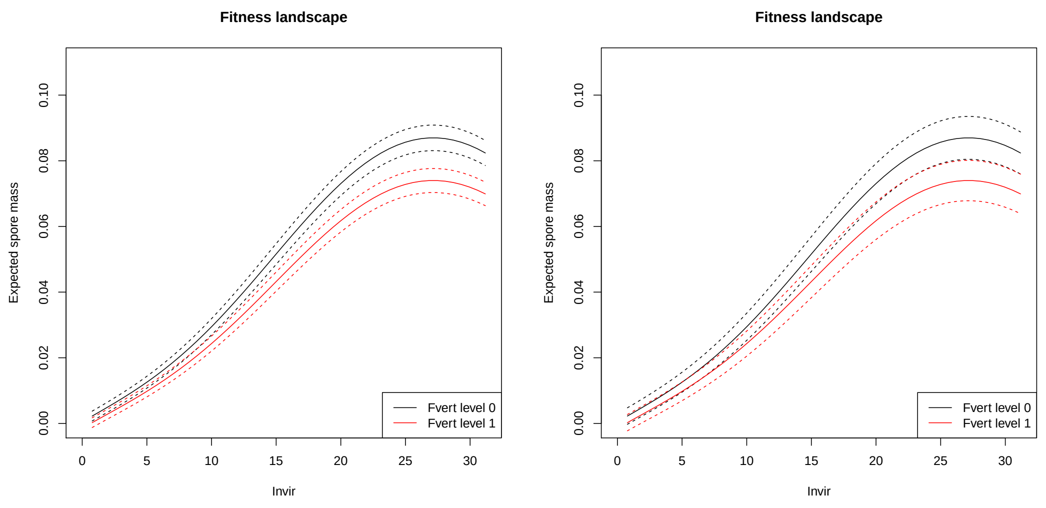

Figure 11: The fitness landscape for hypothetical individuals possessing `Invir` values suggested by the original data with standard errors. These fitness landscapes are from an aster model with no LH2 node. The left panel shows confidence bands at 95% confidence level. The right panel shows confidence bands at 95% confidence level adjusted for multiple comparisons.

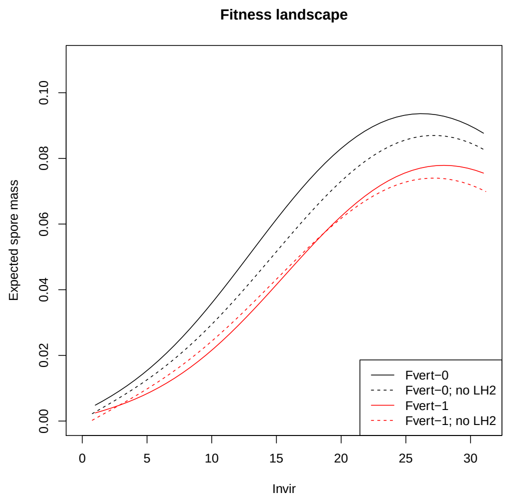

Figure 12: The fitness landscape for hypothetical individuals possessing `Invir` values suggested by the original data with standard errors. These fitness landscapes are from an aster model with no LH2 node. The dashled lines correspond to 95% confidence intervals.

References

Eck, D. J., Shaw, R. G., Geyer, C. J., and Kingsolver, J. G. (2015). An Integrated Analysis of Phenotypic Selection on Insect Body Size and Development Time. *Evolution*, **69**, 9, 2525–2532.

Eck, D. J., Shaw, R. G., Geyer, C. J., and Kingsolver, J. G. (2015). Supporting Data Analysis for "An Integrated Analysis of Phenotypic Selection on Insect Body Size and Development Time." Technical Report No. 698. School of Statistics, University of Minnesota. `http://conservancy.umn.edu/handle/11299/172272`.

Geyer, C. J. (2010). R package `aster2` (Aster Models), version 0.1-1. `http://cran.r-project.org/package=aster2`.

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