Protocol for natural selection analysis

Supplementary Material. Measuring natural selection on multivariate phenotypic traits. A protocol for verifiable and reproducible analyses of natural selection. Facundo Palacio, Mariano Ordano, and Santiago Benitez-Vieyra.

Load packages and data

```
library(MVN)
library(car)
library(boot)
library(visreg)
library(ggplot2)
library(knitr)

data <- read.table("cyclop.txt", header = TRUE)
data <- na.omit(data)
attach(data)
```

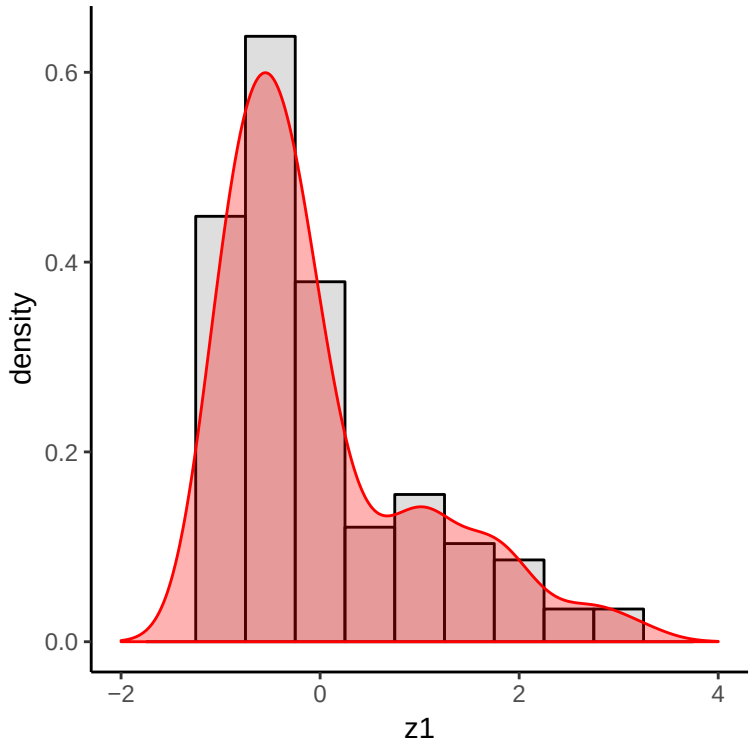
Step 1. Relative fitness and standardization of phenotypic traits (mean = 0, variance = 1)

```
W <- exported.pollinaria
wrel <- W/mean(W)
x1 <- flower.number
x2 <- nectary.depth
z1 <- (x1 - mean(x1))/sd(x1)
z2 <- (x2 - mean(x2))/sd(x2)
```

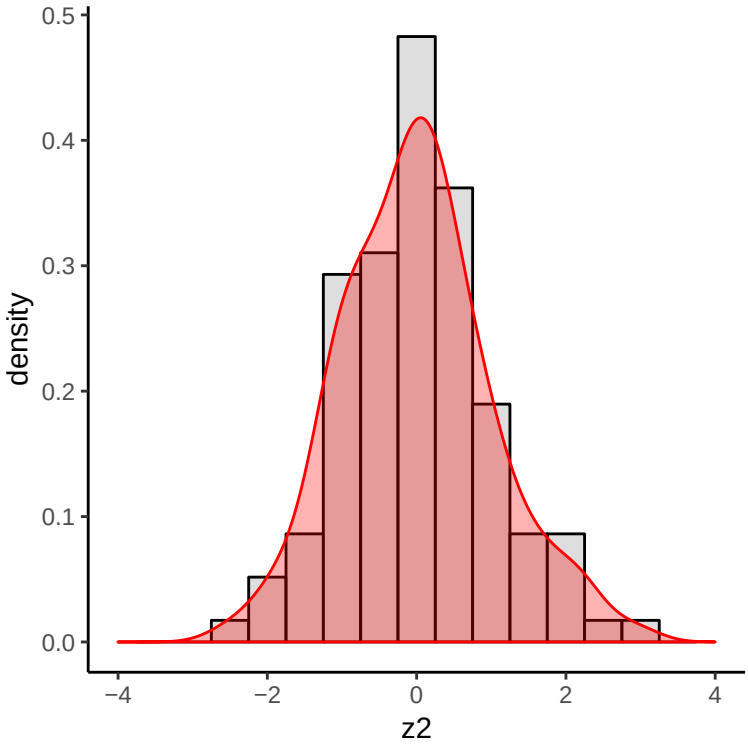
Step 2. Testing multivariate normality of phenotypic traits

2.1. Histograms

```
ggplot(data = data, aes(x = z1)) + xlim(-2, 4) +
  geom_histogram(aes(y = ..density..), col = "black",
                 fill = "gray", binwidth = 0.5, alpha = 0.5) +
  geom_density(col = "red", fill = "red", alpha = 0.3) + theme_classic()
```




```
ggplot(data = data, aes(x = z2)) + xlim(-4, 4) +
  geom_histogram(aes(y = ..density..), col = "black",
                 fill = "gray", binwidth = 0.5, alpha = 0.5) +
  geom_density(col = "red", fill = "red", alpha = 0.3) + theme_classic()
```



2.2. Multivariate normality tests

```
kable(mvn(data.frame(z1, z2), mvnTest = "hz")$multivariateNormality, digits = 3)
```

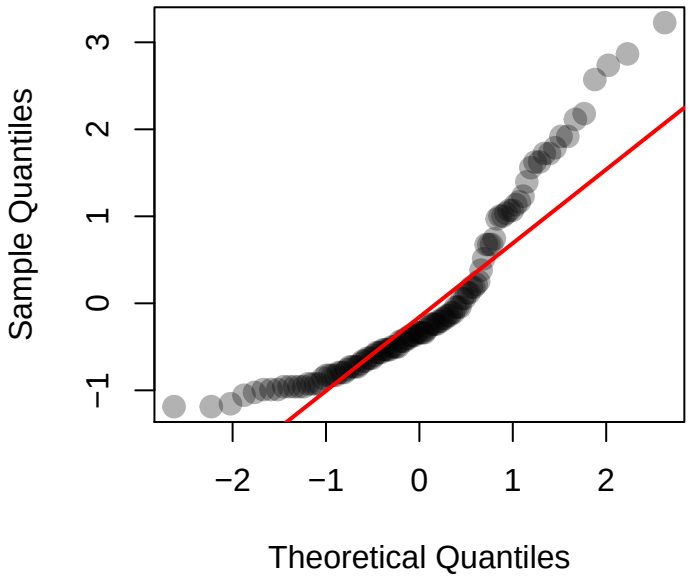
Test	HZ	p value	MVN
Henze-Zirkler	3.482	0	NO

```
kable(mvn(data.frame(z1, z2), mvnTest = "royston")$multivariateNormality, digits = 3)
```

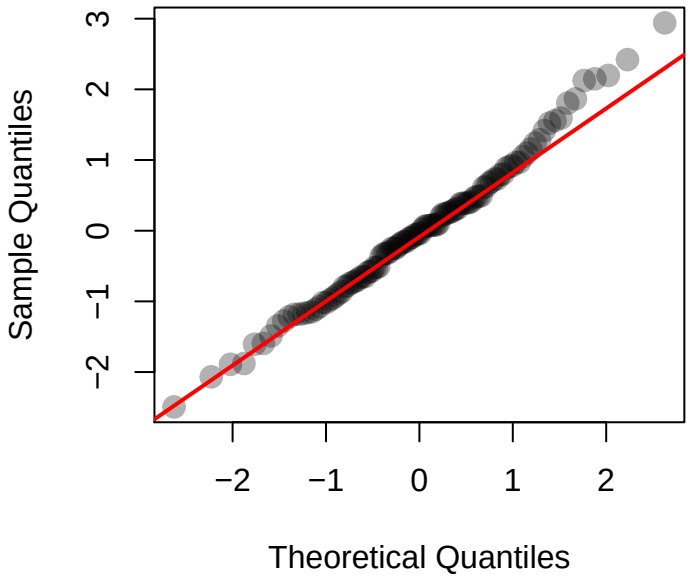
Test	H	p value	MVN
Royston	34.789	0	NO

2.3. Q-Q plots

```
qqnorm(z1, cex = 1.5, pch = 19, col = rgb(red = 0, green = 0, blue = 0, alpha = 0.3), main = "")
qqline(z1, col = "red", lwd = 2)
```




```
qqnorm(z2, cex = 1.5, pch = 19, col = rgb(red = 0, green = 0, blue = 0, alpha = 0.3), main = "")
qqline(z2, col = "red", lwd = 2)
```



Step 3. Assessing collinearity of phenotypic traits

3.1. Pearson correlation between traits

```
cor(z1, z2)
```

```
## [1] 0.2101184
```

3.2. Variance inflation factor on (linear) Lande and Arnold's model

```
lin.grad <- lm(wrel ~ z1 + z2)
nonlin.grad <- lm(wrel ~ z1 + z2 + I((1/2)*z1^2) + I((1/2)*z2^2) + z1:z2)

kable(vif(lin.grad), digits = 3)
```

	x
z1	1.046
z2	1.046

```
kable(summary(lin.grad)$coeff, digits = 3)
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.000	0.038	26.409	0
z1	0.652	0.039	16.752	0
z2	0.198	0.039	5.082	0

```
kable(summary(nonlin.grad)$coeff, digits = 3)
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	1.112	0.051	21.892	0.000
z1	0.780	0.055	14.289	0.000
z2	0.173	0.037	4.717	0.000
I((1/2) * z1^2)	-0.200	0.064	-3.109	0.002
I((1/2) * z2^2)	-0.086	0.056	-1.524	0.130
z1:z2	0.143	0.038	3.822	0.000

Step 4. Checking model residuals

4.1. Shapiro test

```
shapiro.test(resid(lin.grad))
```

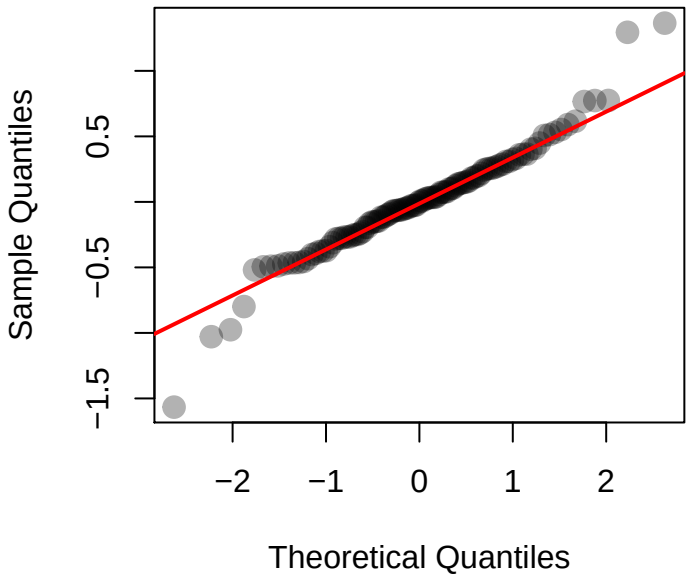
```
##
##  Shapiro-Wilk normality test
##
## data:  resid(lin.grad)
## W = 0.95743, p-value = 0.001003
```

```
shapiro.test(resid(nonlin.grad))
```

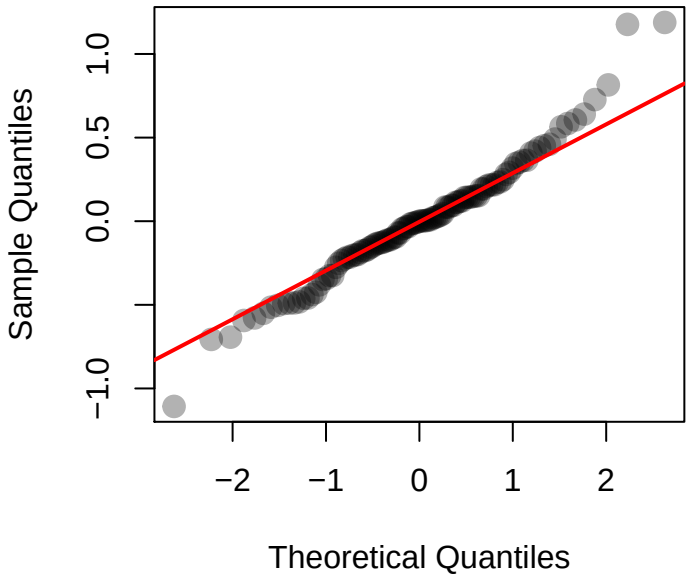
```
##
##  Shapiro-Wilk normality test
##
## data:  resid(nonlin.grad)
## W = 0.97573, p-value = 0.03342
```

4.2. Q-Q plots

```
qqnorm(resid(lin.grad), cex = 1.5, pch = 19,
       col = rgb(red = 0, green = 0, blue = 0, alpha = 0.3), main = "")
qqline(resid(lin.grad), col = "red", lwd = 2)
```




```
qqnorm(resid(nonlin.grad), cex = 1.5, pch = 19,
       col = rgb(red = 0, green = 0, blue = 0, alpha = 0.3), main = "")
qqline(resid(nonlin.grad), col = "red", lwd = 2)
```



Step 5. Standard error and confidence interval estimation

5.1. For the grad function, the data must have the relative fitness in the first column and standardized variables in the remaining columns. The function returns a vector with linear, quadratic and correlational gradients, and represents the input for the function boot.

```
grad <- function(data, original = c(1:nrow(data))){
  data <- data[original, ]
  vars  <- colnames(data)[-1]
  colnames(data)[1] <- "Wrel"
  model.lin <- as.formula(paste("Wrel", paste(vars, collapse=" + "), sep=" ~ "))
  m1 <- lm(formula = model.lin, data = data)
  part1 <- paste("(", paste(vars, collapse=" + "), ")^2", sep = "")
```

```
  part2 <- paste("I(0.5*(", vars, "^2))", sep = "", collapse = " + ")
  model.qua <- as.formula <- paste("Wrel", paste(part1, part2, sep = " + "), sep = " ~ ")
  m2 <- lm(formula = model.qua, data = data)
  sel.grad<-c(m1$coefficients[-1], m2$coefficients[-c(1:ncol(data))])
  return(sel.grad)
}

newdata <- data.frame(wrel, z1, z2)
selection.gradients <- grad(data = newdata)
boot.grad <- boot(data = newdata, statistic = grad, R = 999)
```

5.2. Create a list with 95% bias-corrected bootstrap confidence intervals for each gradient.

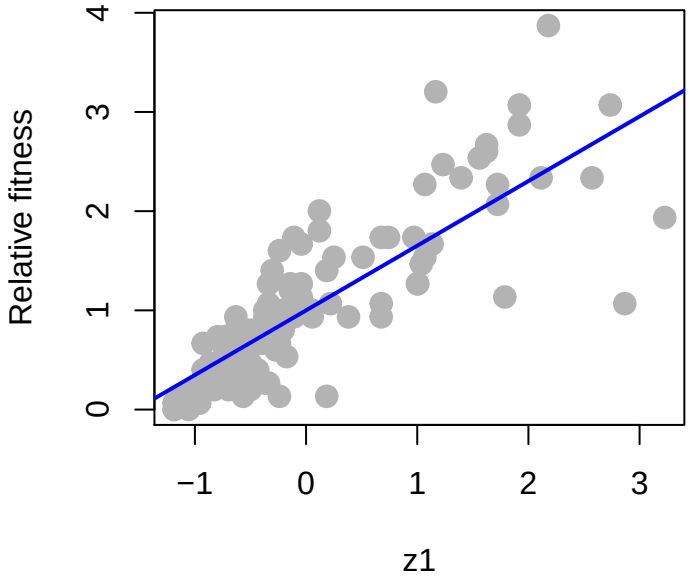
```
CI <- list()
for(i in 1:length(boot.grad$t0)){
  CI[[i]] <- boot.ci(boot.grad, conf = 0.95, type = "bca", index = i)$bca[4:5]
}

names(CI) <- names(boot.grad$t0)
CI <- do.call(rbind, CI)
colnames(CI) <-c("lower.ci", "upper.ci")
kable(CI, digits = 3)
```

	lower.ci	upper.ci
z1	0.505	0.777
z2	0.121	0.291
I(0.5 * (z1^2))	-0.354	-0.043
I(0.5 * (z2^2))	-0.239	0.049
z1:z2	0.024	0.267

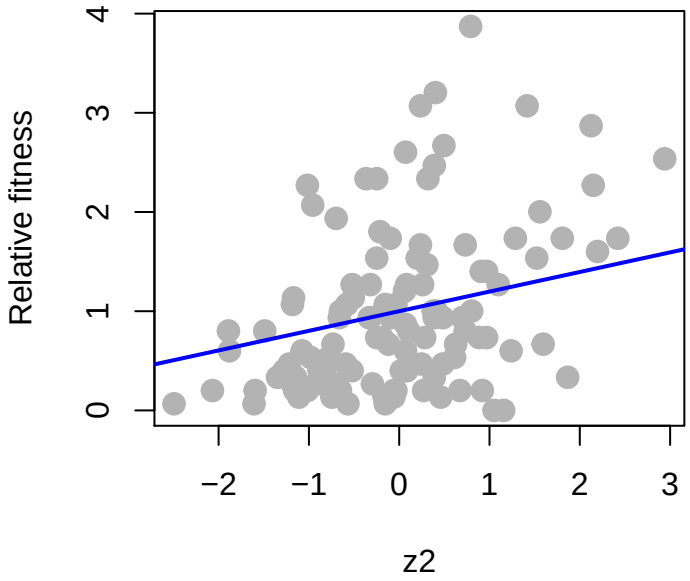
Step 6. Plotting Lande & Arnold's model results

6.1. Linear selection

```
new.z1 <- seq(-4, 4, length = 500)
plot(z1, wrel, pch = 19, cex = 1.5, col = "gray70", ylab = "Relative fitness")
pred.z1 <- predict(lin.grad, newdata = data.frame(z1 = new.z1, z2 = mean(z2)))
lines(new.z1, pred.z1, lwd = 2, col = "blue")
```

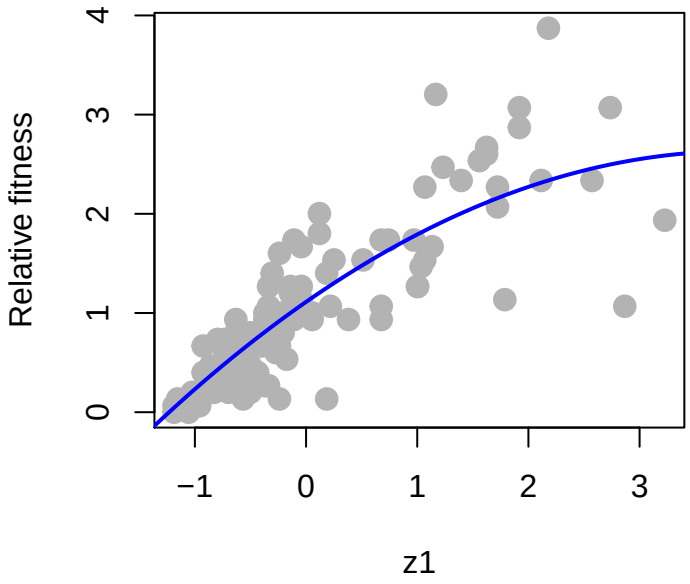



```
new.z2 <- seq(-4, 4, length = 500)
plot(z2, wrel, pch = 19, cex = 1.5, col = "gray70", ylab = "Relative fitness")
pred.z2 <- predict(lin.grad, newdata = data.frame(z1 = mean(z1), z2 = new.z2))
lines(new.z2, pred.z2, lwd = 2, col = "blue")
```

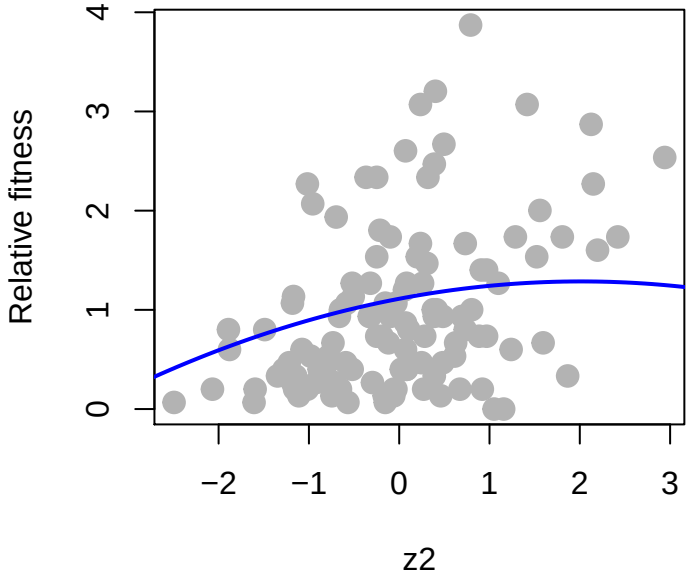


6.3. Quadratic selection

```
new.z1 <- seq(-4, 4, length = 500)
plot(z1, wrel, pch = 19, cex = 1.5, col = "gray70", ylab = "Relative fitness")
pred.z1 <- predict(nonlin.grad, newdata = data.frame(z1 = new.z1, z2 = mean(z2)))
lines(new.z1, pred.z1, lwd = 2, col = "blue")
```




```
new.z2 <- seq(-4, 4, length = 500)
plot(z2, wrel, pch = 19, cex = 1.5, col = "gray70", ylab = "Relative fitness")
pred.z2 <- predict(nonlin.grad, newdata = data.frame(z1 = mean(z1), z2 = new.z2))
lines(new.z2, pred.z2, lwd = 2, col = "blue")
```



6.4. Correlational selection

```
visreg2d(fit = nonlin.grad, xvar = "z1", yvar = "z2", scale = "response", plot.type = "image")
```