Appendix S4. The relative importance of evolution in populations and communities

Jelena H. Pantel* Ruben J. Hermann†

03 November, 2024

1 Logistic growth and trait evolution

We can create simulations of evolutionary rescue in a population, where adaptive evolution rescues a population from extinction, at a speed that depends on the heritability level in the system. Population dynamics are modelled as:

$$N_{i,t+1} = \frac{\hat{W} e^{\frac{-[(\frac{w+(1-h^2)P}{P+w})(E-x_{i,t})]^2}{2(P+w)}} N_{i,t}}{1 + \alpha_{ii} N_{i,t} + \alpha_{ij} N_{j,t}}$$

where $\hat{W}$ is calculated as $\hat{W} = W_{max}\sqrt{(\frac{w}{P+w})}$, W_{max} is the species' maximum per-capita growth rate, w is the width of the Gaussian fitness function (which determines the strength of selection, as increasing values indicate a weaker reduction in fitness with distance from optimum trait value), P is the width of the distribution of the phenotype x, and h^2 is the heritability of the trait x. For the simulation we use $W_{max} = 2$, $P = 1$, and $w = 2$.

The change in the average trait value each time step is given by:

$$d_{i,t+1} = k d_{i,t}$$

where $k = \frac{w+(1-h^2)P}{w+P}$ and $d_{i,t} = E_t - x_{i,t}$.

1.1 Population dynamics simulation

```
### 1 species, h2=0 ### Initial conditions
N0 <- 10
alpha.11 <- 0.01
alpha <- matrix(alpha.11, byrow = FALSE)
E.0 <- 0.8
P <- 1
w <- 0.5
# Draw initial trait value using degree of initial maladaptation
```

*Laboratoire Chrono-environnement,UMR 6249 CNRS-UFC, 16 Route de Gray, 25030 Besançon cedex, France, jelena.pantel@univ-fcomte.fr

†University of Duisburg-Essen, Universitätsstraße 5, 45141 Essen, Germany, ruben.hermann@uni-due.de

```
B0 <- 1
d0 = sqrt(B0 * (w + P))
x.0 = E.0 - d0
Wmax <- 2
h2 <- 0
k <- (w + (1 - h2) * P)/(P + w)
# Simulation of model for t time steps Simulation of model for t time steps
t <- 60
N <- array(NA, dim = c(t, length(N0)))
N <- as.data.frame(N)
colnames(N) <- paste0("N", 1:length(N0))
N[1, ] <- N0
E <- rep(NA, t)
E[1] <- E.0
x <- array(NA, dim = c(t, length(N0)))
x <- as.data.frame(x)
colnames(x) <- paste0("x", 1:length(N0))
x[1, ] <- x.0
r <- array(NA, dim = c(t, length(N0)))
r <- as.data.frame(r)
colnames(r) <- paste0("r", 1:length(N0))
What <- Wmax * sqrt(w/(P + w))
r[1, ] <- What * exp((-(((w + (1 - h2) * P)/(P + w)) * (E[1] - x[1, ]))^2)/(2 * (P +
    w)))
for (i in 2:t) {
    res <- disc_LV_evol(N0 = N[i - 1, ], alpha = alpha, E = E[i - 1], x = x[i - 1,
        ], P = P, w = w, Wmax = Wmax, h2 = h2)
    N[i, ] <- res$Nt1
    r[i, ] <- res$r
    # trait change
    d <- E[i - 1] - x[i - 1, ]
    d1 <- k * d
    x[i, ] <- E[i - 1] - d1
    # environmental change
    E[i] <- E[i - 1] + abs(rnorm(1, 0, 0))
}
# Arrange to save across scenarios
gdat <- N
gdat$time <- 1:t
gdat$h2 <- h2
gdat$x1 <- x$x1
gdat$E <- E
```

```
### 1 species, h2=0.01 ### Initial conditions
N0 <- 10
alpha.11 <- 0.01
alpha <- matrix(alpha.11, byrow = FALSE)
E.0 <- 0.8
P <- 1
w <- 0.5
# Draw initial trait value using degree of initial maladaptation
B0 <- 1
d0 = sqrt(B0 * (w + P))
```

```
x.0 = E.0 - d0
Wmax <- 2
h2 <- 0.01
k <- (w + (1 - h2) * P)/(P + w)
# Simulation of model for t time steps Simulation of model for t time steps
t <- 60
N <- array(NA, dim = c(t, length(N0)))
N <- as.data.frame(N)
colnames(N) <- paste0("N", 1:length(N0))
N[1, ] <- N0
E <- rep(NA, t)
E[1] <- E.0
x <- array(NA, dim = c(t, length(N0)))
x <- as.data.frame(x)
colnames(x) <- paste0("x", 1:length(N0))
x[1, ] <- x.0
r <- array(NA, dim = c(t, length(N0)))
r <- as.data.frame(r)
colnames(r) <- paste0("r", 1:length(N0))
What <- Wmax * sqrt(w/(P + w))
r[1, ] <- What * exp((-(((w + (1 - h2) * P)/(P + w)) * (E[1] - x[1, ]))^2)/(2 * (P +
    w)))
for (i in 2:t) {
    res <- disc_LV_evol(N0 = N[i - 1, ], alpha = alpha, E = E[i - 1], x = x[i - 1,
        ], P = P, w = w, Wmax = Wmax, h2 = h2)
    N[i, ] <- res$Nt1
    r[i, ] <- res$r
    # trait change
    d <- E[i - 1] - x[i - 1, ]
    d1 <- k * d
    x[i, ] <- E[i - 1] - d1
    # environmental change
    E[i] <- E[i - 1] + abs(rnorm(1, 0, 0))
}
# Arrange to save across scenarios
gdat <- rbind(gdat, data.frame(N, time = 1:t, h2 = rep(h2, t), x1 = x$x1, E = E))
```

```
### 1 species, h2=0.02 ### Initial conditions
N0 <- 10
alpha.11 <- 0.01
alpha <- matrix(alpha.11, byrow = FALSE)
E.0 <- 0.8
P <- 1
w <- 0.5
# Draw initial trait value using degree of initial maladaptation
B0 <- 1
d0 = sqrt(B0 * (w + P))
x.0 = E.0 - d0
Wmax <- 2
h2 <- 0.02
k <- (w + (1 - h2) * P)/(P + w)
# Simulation of model for t time steps Simulation of model for t time steps
t <- 60
```

```
N <- array(NA, dim = c(t, length(N0)))
N <- as.data.frame(N)
colnames(N) <- paste0("N", 1:length(N0))
N[1, ] <- N0
E <- rep(NA, t)
E[1] <- E.0
x <- array(NA, dim = c(t, length(N0)))
x <- as.data.frame(x)
colnames(x) <- paste0("x", 1:length(N0))
x[1, ] <- x.0
r <- array(NA, dim = c(t, length(N0)))
r <- as.data.frame(r)
colnames(r) <- paste0("r", 1:length(N0))
What <- Wmax * sqrt(w/(P + w))
r[1, ] <- What * exp((-(((w + (1 - h2) * P)/(P + w)) * (E[1] - x[1, ]))^2)/(2 * (P +
    w)))
for (i in 2:t) {
    res <- disc_LV_evol(N0 = N[i - 1, ], alpha = alpha, E = E[i - 1], x = x[i - 1,
        ], P = P, w = w, Wmax = Wmax, h2 = h2)
    N[i, ] <- res$Nt1
    r[i, ] <- res$r
    # trait change
    d <- E[i - 1] - x[i - 1, ]
    d1 <- k * d
    x[i, ] <- E[i - 1] - d1
    # environmental change
    E[i] <- E[i - 1] + abs(rnorm(1, 0, 0))
}
# Arrange to save across scenarios
gdat <- rbind(gdat, data.frame(N, time = 1:t, h2 = rep(h2, t), x1 = x$x1, E = E))
```

```
### 1 species, h2=0.03 ### Initial conditions
N0 <- 10
alpha.11 <- 0.01
alpha <- matrix(alpha.11, byrow = FALSE)
E.0 <- 0.8
P <- 1
w <- 0.5
# Draw initial trait value using degree of initial maladaptation
B0 <- 1
d0 = sqrt(B0 * (w + P))
x.0 = E.0 - d0
Wmax <- 2
h2 <- 0.03
k <- (w + (1 - h2) * P)/(P + w)
# Simulation of model for t time steps Simulation of model for t time steps
t <- 60
N <- array(NA, dim = c(t, length(N0)))
N <- as.data.frame(N)
colnames(N) <- paste0("N", 1:length(N0))
N[1, ] <- N0
E <- rep(NA, t)
E[1] <- E.0
```

```
x <- array(NA, dim = c(t, length(N0)))
x <- as.data.frame(x)
colnames(x) <- paste0("x", 1:length(N0))
x[1, ] <- x.0
r <- array(NA, dim = c(t, length(N0)))
r <- as.data.frame(r)
colnames(r) <- paste0("r", 1:length(N0))
What <- Wmax * sqrt(w/(P + w))
r[1, ] <- What * exp((-(((w + (1 - h2) * P)/(P + w)) * (E[1] - x[1, ]))^2)/(2 * (P +
    w)))
for (i in 2:t) {
    res <- disc_LV_evol(N0 = N[i - 1, ], alpha = alpha, E = E[i - 1], x = x[i - 1,
        ], P = P, w = w, Wmax = Wmax, h2 = h2)
    N[i, ] <- res$Nt1
    r[i, ] <- res$r
    # trait change
    d <- E[i - 1] - x[i - 1, ]
    d1 <- k * d
    x[i, ] <- E[i - 1] - d1
    # environmental change
    E[i] <- E[i - 1] + abs(rnorm(1, 0, 0))
}
# Arrange to save across scenarios
gdat <- rbind(gdat, data.frame(N, time = 1:t, h2 = rep(h2, t), x1 = x$x1, E = E))
```

```
### 1 species, h2=0.04 ### Initial conditions
N0 <- 10
alpha.11 <- 0.01
alpha <- matrix(alpha.11, byrow = FALSE)
E.0 <- 0.8
P <- 1
w <- 0.5
# Draw initial trait value using degree of initial maladaptation
B0 <- 1
d0 = sqrt(B0 * (w + P))
x.0 = E.0 - d0
Wmax <- 2
h2 <- 0.04
k <- (w + (1 - h2) * P)/(P + w)
# Simulation of model for t time steps Simulation of model for t time steps
t <- 60
N <- array(NA, dim = c(t, length(N0)))
N <- as.data.frame(N)
colnames(N) <- paste0("N", 1:length(N0))
N[1, ] <- N0
E <- rep(NA, t)
E[1] <- E.0
x <- array(NA, dim = c(t, length(N0)))
x <- as.data.frame(x)
colnames(x) <- paste0("x", 1:length(N0))
x[1, ] <- x.0
r <- array(NA, dim = c(t, length(N0)))
r <- as.data.frame(r)
```

```
colnames(r) <- paste0("r", 1:length(N0))
What <- Wmax * sqrt(w/(P + w))
r[1, ] <- What * exp((-(((w + (1 - h2) * P)/(P + w)) * (E[1] - x[1, ]))^2)/(2 * (P +
    w)))
for (i in 2:t) {
    res <- disc_LV_evol(N0 = N[i - 1, ], alpha = alpha, E = E[i - 1], x = x[i - 1,
        ], P = P, w = w, Wmax = Wmax, h2 = h2)
    N[i, ] <- res$Nt1
    r[i, ] <- res$r
    # trait change
    d <- E[i - 1] - x[i - 1, ]
    d1 <- k * d
    x[i, ] <- E[i - 1] - d1
    # environmental change
    E[i] <- E[i - 1] + abs(rnorm(1, 0, 0))
}
# Arrange to save across scenarios
gdat <- rbind(gdat, data.frame(N, time = 1:t, h2 = rep(h2, t), x1 = x$x1, E = E))
```

```
### 1 species, h2=0.05 ### Initial conditions
N0 <- 10
alpha.11 <- 0.01
alpha <- matrix(alpha.11, byrow = FALSE)
E.0 <- 0.8
P <- 1
w <- 0.5
# Draw initial trait value using degree of initial maladaptation
B0 <- 1
d0 = sqrt(B0 * (w + P))
x.0 = E.0 - d0
Wmax <- 2
h2 <- 0.05
k <- (w + (1 - h2) * P)/(P + w)
# Simulation of model for t time steps Simulation of model for t time steps
t <- 60
N <- array(NA, dim = c(t, length(N0)))
N <- as.data.frame(N)
colnames(N) <- paste0("N", 1:length(N0))
N[1, ] <- N0
E <- rep(NA, t)
E[1] <- E.0
x <- array(NA, dim = c(t, length(N0)))
x <- as.data.frame(x)
colnames(x) <- paste0("x", 1:length(N0))
x[1, ] <- x.0
r <- array(NA, dim = c(t, length(N0)))
r <- as.data.frame(r)
colnames(r) <- paste0("r", 1:length(N0))
What <- Wmax * sqrt(w/(P + w))
r[1, ] <- What * exp((-(((w + (1 - h2) * P)/(P + w)) * (E[1] - x[1, ]))^2)/(2 * (P +
    w)))
for (i in 2:t) {
    res <- disc_LV_evol(N0 = N[i - 1, ], alpha = alpha, E = E[i - 1], x = x[i - 1,
```

```
        ], P = P, w = w, Wmax = Wmax, h2 = h2)
    N[i, ] <- res$Nt1
    r[i, ] <- res$r
    # trait change
    d <- E[i - 1] - x[i - 1, ]
    d1 <- k * d
    x[i, ] <- E[i - 1] - d1
    # environmental change
    E[i] <- E[i - 1] + abs(rnorm(1, 0, 0))
}
# Arrange to save across scenarios
gdat <- rbind(gdat, data.frame(N, time = 1:t, h2 = rep(h2, t), x1 = x$x1, E = E))
```

```
### 1 species, h2=0.075 ### Initial conditions
N0 <- 10
alpha.11 <- 0.01
alpha <- matrix(alpha.11, byrow = FALSE)
E.0 <- 0.8
P <- 1
w <- 0.5
# Draw initial trait value using degree of initial maladaptation
B0 <- 1
d0 = sqrt(B0 * (w + P))
x.0 = E.0 - d0
Wmax <- 2
h2 <- 0.075
k <- (w + (1 - h2) * P)/(P + w)
# Simulation of model for t time steps Simulation of model for t time steps
t <- 60
N <- array(NA, dim = c(t, length(N0)))
N <- as.data.frame(N)
colnames(N) <- paste0("N", 1:length(N0))
N[1, ] <- N0
E <- rep(NA, t)
E[1] <- E.0
x <- array(NA, dim = c(t, length(N0)))
x <- as.data.frame(x)
colnames(x) <- paste0("x", 1:length(N0))
x[1, ] <- x.0
r <- array(NA, dim = c(t, length(N0)))
r <- as.data.frame(r)
colnames(r) <- paste0("r", 1:length(N0))
What <- Wmax * sqrt(w/(P + w))
r[1, ] <- What * exp((-(((w + (1 - h2) * P)/(P + w)) * (E[1] - x[1, ]))^2)/(2 * (P +
    w)))
for (i in 2:t) {
    res <- disc_LV_evol(N0 = N[i - 1, ], alpha = alpha, E = E[i - 1], x = x[i - 1,
        ], P = P, w = w, Wmax = Wmax, h2 = h2)
    N[i, ] <- res$Nt1
    r[i, ] <- res$r
    # trait change
    d <- E[i - 1] - x[i - 1, ]
    d1 <- k * d
```

```
    x[i, ] <- E[i - 1] - d1
    # environmental change
    E[i] <- E[i - 1] + abs(rnorm(1, 0, 0))
}
# Arrange to save across scenarios
gdat <- rbind(gdat, data.frame(N, time = 1:t, h2 = rep(h2, t), x1 = x$x1, E = E))
```

```
### 1 species, h2=0.1 ### Initial conditions
N0 <- 10
alpha.11 <- 0.01
alpha <- matrix(alpha.11, byrow = FALSE)
E.0 <- 0.8
P <- 1
w <- 0.5
# Draw initial trait value using degree of initial maladaptation
B0 <- 1
d0 = sqrt(B0 * (w + P))
x.0 = E.0 - d0
Wmax <- 2
h2 <- 0.1
k <- (w + (1 - h2) * P)/(P + w)
# Simulation of model for t time steps Simulation of model for t time steps
t <- 60
N <- array(NA, dim = c(t, length(N0)))
N <- as.data.frame(N)
colnames(N) <- paste0("N", 1:length(N0))
N[1, ] <- N0
E <- rep(NA, t)
E[1] <- E.0
x <- array(NA, dim = c(t, length(N0)))
x <- as.data.frame(x)
colnames(x) <- paste0("x", 1:length(N0))
x[1, ] <- x.0
r <- array(NA, dim = c(t, length(N0)))
r <- as.data.frame(r)
colnames(r) <- paste0("r", 1:length(N0))
What <- Wmax * sqrt(w/(P + w))
r[1, ] <- What * exp((-(((w + (1 - h2) * P)/(P + w)) * (E[1] - x[1, ]))^2)/(2 * (P +
    w)))
for (i in 2:t) {
    res <- disc_LV_evol(N0 = N[i - 1, ], alpha = alpha, E = E[i - 1], x = x[i - 1,
        ], P = P, w = w, Wmax = Wmax, h2 = h2)
    N[i, ] <- res$Nt1
    r[i, ] <- res$r
    # trait change
    d <- E[i - 1] - x[i - 1, ]
    d1 <- k * d
    x[i, ] <- E[i - 1] - d1
    # environmental change
    E[i] <- E[i - 1] + abs(rnorm(1, 0, 0))
}
# Arrange to save across scenarios
gdat <- rbind(gdat, data.frame(N, time = 1:t, h2 = rep(h2, t), x1 = x$x1, E = E))
```

```r
### 1 species, h2=0.4 ### Initial conditions
N0 <- 10
alpha.11 <- 0.01
alpha <- matrix(alpha.11, byrow = FALSE)
E.0 <- 0.8
P <- 1
w <- 0.5
# Draw initial trait value using degree of initial maladaptation
B0 <- 1
d0 = sqrt(B0 * (w + P))
x.0 = E.0 - d0
Wmax <- 2
h2 <- 0.4
k <- (w + (1 - h2) * P)/(P + w)
# Simulation of model for t time steps Simulation of model for t time steps
t <- 60
N <- array(NA, dim = c(t, length(N0)))
N <- as.data.frame(N)
colnames(N) <- paste0("N", 1:length(N0))
N[1, ] <- N0
E <- rep(NA, t)
E[1] <- E.0
x <- array(NA, dim = c(t, length(N0)))
x <- as.data.frame(x)
colnames(x) <- paste0("x", 1:length(N0))
x[1, ] <- x.0
r <- array(NA, dim = c(t, length(N0)))
r <- as.data.frame(r)
colnames(r) <- paste0("r", 1:length(N0))
What <- Wmax * sqrt(w/(P + w))
r[1, ] <- What * exp((-(((w + (1 - h2) * P)/(P + w)) * (E[1] - x[1, ]))^2)/(2 * (P +
    w)))
for (i in 2:t) {
    res <- disc_LV_evol(N0 = N[i - 1, ], alpha = alpha, E = E[i - 1], x = x[i - 1,
        ], P = P, w = w, Wmax = Wmax, h2 = h2)
    N[i, ] <- res$Nt1
    r[i, ] <- res$r
    # trait change
    d <- E[i - 1] - x[i - 1, ]
    d1 <- k * d
    x[i, ] <- E[i - 1] - d1
    # environmental change
    E[i] <- E[i - 1] + abs(rnorm(1, 0, 0))
}
# Arrange to save across scenarios
gdat <- rbind(gdat, data.frame(N, time = 1:t, h2 = rep(h2, t), x1 = x$x1, E = E))
```

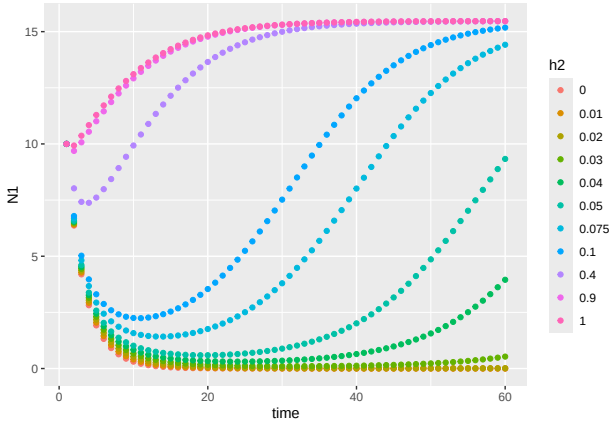
```r
### 1 species, h2=0.9 ### Initial conditions
N0 <- 10
alpha.11 <- 0.01
alpha <- matrix(alpha.11, byrow = FALSE)
E.0 <- 0.8
P <- 1
```

```
w <- 0.5
# Draw initial trait value using degree of initial maladaptation
B0 <- 1
d0 = sqrt(B0 * (w + P))
x.0 = E.0 - d0
Wmax <- 2
h2 <- 0.9
k <- (w + (1 - h2) * P)/(P + w)
# Simulation of model for t time steps Simulation of model for t time steps
t <- 60
N <- array(NA, dim = c(t, length(N0)))
N <- as.data.frame(N)
colnames(N) <- paste0("N", 1:length(N0))
N[1, ] <- N0
E <- rep(NA, t)
E[1] <- E.0
x <- array(NA, dim = c(t, length(N0)))
x <- as.data.frame(x)
colnames(x) <- paste0("x", 1:length(N0))
x[1, ] <- x.0
r <- array(NA, dim = c(t, length(N0)))
r <- as.data.frame(r)
colnames(r) <- paste0("r", 1:length(N0))
What <- Wmax * sqrt(w/(P + w))
r[1, ] <- What * exp((-(((w + (1 - h2) * P)/(P + w)) * (E[1] - x[1, ]))^2)/(2 * (P +
    w)))
for (i in 2:t) {
    res <- disc_LV_evol(N0 = N[i - 1, ], alpha = alpha, E = E[i - 1], x = x[i - 1,
        ], P = P, w = w, Wmax = Wmax, h2 = h2)
    N[i, ] <- res$Nt1
    r[i, ] <- res$r
    # trait change
    d <- E[i - 1] - x[i - 1, ]
    d1 <- k * d
    x[i, ] <- E[i - 1] - d1
    # environmental change
    E[i] <- E[i - 1] + abs(rnorm(1, 0, 0))
}
# Arrange to save across scenarios
gdat <- rbind(gdat, data.frame(N, time = 1:t, h2 = rep(h2, t), x1 = x$x1, E = E))
```

```
### 1 species, h2=1 ### Initial conditions
N0 <- 10
alpha.11 <- 0.01
alpha <- matrix(alpha.11, byrow = FALSE)
E.0 <- 0.8
P <- 1
w <- 0.5
# Draw initial trait value using degree of initial maladaptation
B0 <- 1
d0 = sqrt(B0 * (w + P))
x.0 = E.0 - d0
Wmax <- 2
```

```
h2 <- 1
k <- (w + (1 - h2) * P)/(P + w)
# Simulation of model for t time steps Simulation of model for t time steps
t <- 60
N <- array(NA, dim = c(t, length(N0)))
N <- as.data.frame(N)
colnames(N) <- paste0("N", 1:length(N0))
N[1, ] <- N0
E <- rep(NA, t)
E[1] <- E.0
x <- array(NA, dim = c(t, length(N0)))
x <- as.data.frame(x)
colnames(x) <- paste0("x", 1:length(N0))
x[1, ] <- x.0
r <- array(NA, dim = c(t, length(N0)))
r <- as.data.frame(r)
colnames(r) <- paste0("r", 1:length(N0))
What <- Wmax * sqrt(w/(P + w))
r[1, ] <- What * exp((-(((w + (1 - h2) * P)/(P + w)) * (E[1] - x[1, ]))^2)/(2 * (P +
    w)))
for (i in 2:t) {
    res <- disc_LV_evol(N0 = N[i - 1, ], alpha = alpha, E = E[i - 1], x = x[i - 1,
        ], P = P, w = w, Wmax = Wmax, h2 = h2)
    N[i, ] <- res$Nt1
    r[i, ] <- res$r
    # trait change
    d <- E[i - 1] - x[i - 1, ]
    d1 <- k * d
    x[i, ] <- E[i - 1] - d1
    # environmental change
    E[i] <- E[i - 1] + abs(rnorm(1, 0, 0))
}
# Arrange to save across scenarios
gdat <- rbind(gdat, data.frame(N, time = 1:t, h2 = rep(h2, t), x1 = x$x1, E = E))
```

```
### Plot of time series ###
gdat$h2 <- as.factor(gdat$h2)
ggplot2::ggplot(gdat, aes(time, N1, col = h2)) + geom_point()
```



17

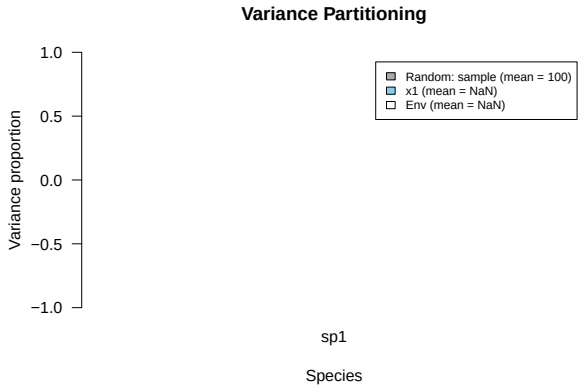
```
### ###
```

1.2 HMSC model fit

The goal of fitting the data to a statistical model is to estimate the relative important of trait evolution for the population dynamics.

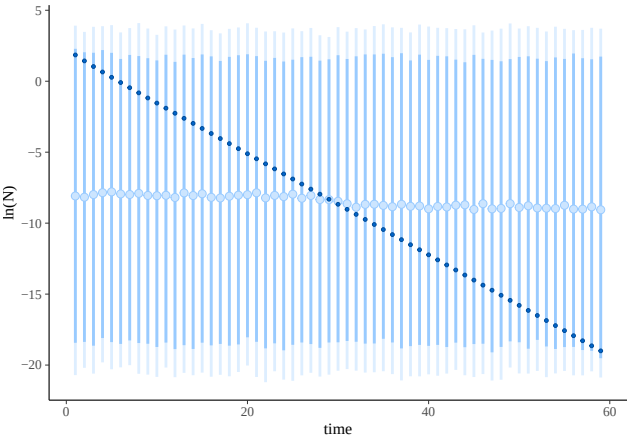
```
### Model 1, h2=00 ###
dat <- cbind(log(gdat$N1[gdat$h2 == 0]), gdat$x1[gdat$h2 == 0])
dat <- as.data.frame(dat)
colnames(dat) <- c("N1", "x1")
dat$time <- 1:t
dat <- as.data.frame(dat)
df <- data.frame(dat[(2:t), -3])
colnames(df) <- c("Nt1", "xt1")
df$dx1 <- abs(dat$x1[2:t] - dat$x1[1:(t - 1)])
# prepare data in HMSC format
Y <- as.matrix(df$Nt1)
XData <- data.frame(cbind(E[1:(t - 1)], E[1:(t - 1)]^2, abs(dat$x1[2:t] - dat$x1[1:(t -
    1)])))
colnames(XData) <- c("E", "Esq", "dx1")

studyDesign = data.frame(sample = as.factor(1:(t - 1)))
rL = HmscRandomLevel(units = studyDesign$sample)

m.1.hmsc = Hmsc(Y = Y, XData = XData, XFormula = ~E + Esq + dx1, studyDesign = studyDesign,
    ranLevels = list(sample = rL))
# Bayesian model parameters
nChains <- 2
thin <- 5
samples <- 2000
transient <- 1000 * thin
verbose <- 500 * thin
# sample MCMC
m.1.sample <- sampleMcmc(m.1.hmsc, thin = thin, sample = samples, transient = transient,
    nChains = nChains, verbose = verbose)
#> Computing chain 1
#> Chain 1, iteration 2500 of 15000 (transient)
#> Chain 1, iteration 5000 of 15000 (transient)
#> Chain 1, iteration 7500 of 15000 (sampling)
#> Chain 1, iteration 10000 of 15000 (sampling)
#> Chain 1, iteration 12500 of 15000 (sampling)
#> Chain 1, iteration 15000 of 15000 (sampling)
#> Computing chain 2
#> Chain 2, iteration 2500 of 15000 (transient)
#> Chain 2, iteration 5000 of 15000 (transient)
#> Chain 2, iteration 7500 of 15000 (sampling)
#> Chain 2, iteration 10000 of 15000 (sampling)
#> Chain 2, iteration 12500 of 15000 (sampling)
#> Chain 2, iteration 15000 of 15000 (sampling)
m1.post.hmsc <- convertToCodaObject(m.1.sample)
```

```
VP1 <- computeVariancePartitioning(m.1.sample, group = c(1, 1, 1, 2), groupnames = c("Env",
    "x1"))
plotVariancePartitioning(m.1.sample, VP1, cols = c("white", "skyblue", "darkgrey"),
    args.legend = list(cex = 0.75, bg = "transparent"))
```



21

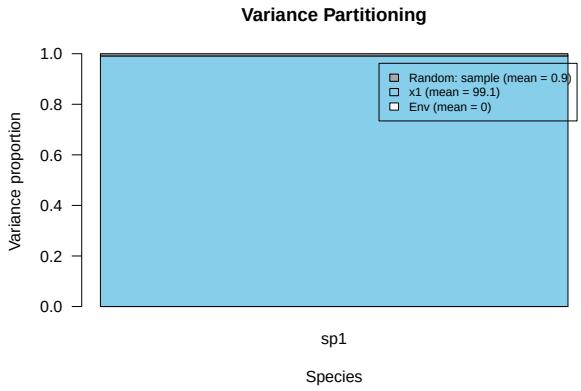
```
# prepare for plot
pred2 <- predict(m.1.sample)
hold1 <- list()
hold2 <- list()
for (i in 1:length(N0)) {
    hold1[[i]] <- as.numeric(Y[, i])
    data <- lapply(pred2, function(x) x[, colnames(x)[i], drop = FALSE])
    hold2[[i]] <- matrix(unlist(data), nrow = length(pred2), byrow = TRUE)
}
y <- unlist(hold1)  # vector of each species' observed N across time points
yrep <- as.matrix(data.frame(hold2))
### Mod2 plot ###
par(mgp = c(2, 0.45, 0), tcl = -0.4, mar = c(1.3, 1.2, 0, 0))
color_scheme_set("brightblue")
# ppc_dens_overlay(y, yrep[1:50, ])
ppc_intervals(y, yrep, x = dat$time[1:(t - 1)], prob = 0.95) + labs(x = "time", y = "ln(N)",
    ) + theme(plot.margin = unit(c(0, 0, 0, 0), "cm")) + theme(legend.position = "none")
```



22

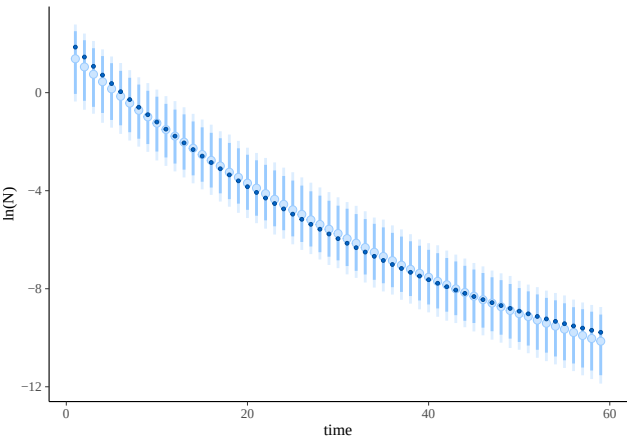
```
### Model 2, h2=0.01 ###
dat <- cbind(log(gdat$N1[gdat$h2 == 0.01]), gdat$x1[gdat$h2 == 0.01])
dat <- as.data.frame(dat)
colnames(dat) <- c("N1", "x1")
dat$time <- 1:t
dat <- as.data.frame(dat)
df <- data.frame(dat[(2:t), -3])
colnames(df) <- c("Nt1", "xt1")
df$dx1 <- abs(dat$x1[2:t] - dat$x1[1:(t - 1)])
# prepare data in HMSC format
Y <- as.matrix(df$Nt1)
XData <- data.frame(cbind(E[1:(t - 1)], E[1:(t - 1)]^2, abs(dat$x1[2:t] - dat$x1[1:(t -
    1)])))
colnames(XData) <- c("E", "Esq", "dx1")

studyDesign = data.frame(sample = as.factor(1:(t - 1)))
rL = HmscRandomLevel(units = studyDesign$sample)

m.2.hmsc = Hmsc(Y = Y, XData = XData, XFormula = ~E + Esq + dx1, studyDesign = studyDesign,
    ranLevels = list(sample = rL))
# Bayesian model parameters
nChains <- 2
thin <- 5
samples <- 2000
transient <- 1000 * thin
verbose <- 500 * thin
# sample MCMC
m.2.sample <- sampleMcmc(m.2.hmsc, thin = thin, sample = samples, transient = transient,
    nChains = nChains, verbose = verbose)
#> Computing chain 1
#> Chain 1, iteration 2500 of 15000 (transient)
#> Chain 1, iteration 5000 of 15000 (transient)
#> Chain 1, iteration 7500 of 15000 (sampling)
#> Chain 1, iteration 10000 of 15000 (sampling)
#> Chain 1, iteration 12500 of 15000 (sampling)
#> Chain 1, iteration 15000 of 15000 (sampling)
#> Computing chain 2
#> Chain 2, iteration 2500 of 15000 (transient)
#> Chain 2, iteration 5000 of 15000 (transient)
#> Chain 2, iteration 7500 of 15000 (sampling)
#> Chain 2, iteration 10000 of 15000 (sampling)
#> Chain 2, iteration 12500 of 15000 (sampling)
#> Chain 2, iteration 15000 of 15000 (sampling)
m2.post.hmsc <- convertToCodaObject(m.2.sample)
```

```
VP2 <- computeVariancePartitioning(m.2.sample, group = c(1, 1, 1, 2), groupnames = c("Env",
    "x1"))
plotVariancePartitioning(m.2.sample, VP2, cols = c("white", "skyblue", "darkgrey"),
    args.legend = list(cex = 0.75, bg = "transparent"))
```



23

```
# prepare for plot
pred2 <- predict(m.2.sample)
hold1 <- list()
hold2 <- list()
for (i in 1:length(N0)) {
    hold1[[i]] <- as.numeric(Y[, i])
    data <- lapply(pred2, function(x) x[, colnames(x)[i], drop = FALSE])
    hold2[[i]] <- matrix(unlist(data), nrow = length(pred2), byrow = TRUE)
}
y <- unlist(hold1)  # vector of each species' observed N across time points
yrep <- as.matrix(data.frame(hold2))
### Mod2 plot ###
par(mgp = c(2, 0.45, 0), tcl = -0.4, mar = c(1.3, 1.2, 0, 0))
color_scheme_set("brightblue")
# ppc_dens_overlay(y, yrep[1:50, ])
ppc_intervals(y, yrep, x = dat$time[1:(t - 1)], prob = 0.95) + labs(x = "time", y = "ln(N)",
    ) + theme(plot.margin = unit(c(0, 0, 0, 0), "cm")) + theme(legend.position = "none")
```

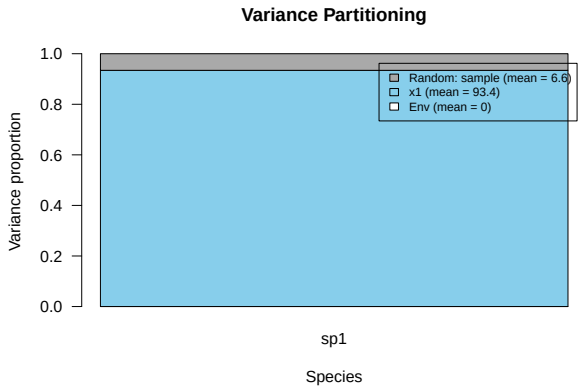


24

```
### Model 3, h2=0.02 ###
dat <- cbind(log(gdat$N1[gdat$h2 == 0.02]), gdat$x1[gdat$h2 == 0.02])
dat <- as.data.frame(dat)
colnames(dat) <- c("N1", "x1")
dat$time <- 1:t
dat <- as.data.frame(dat)
```

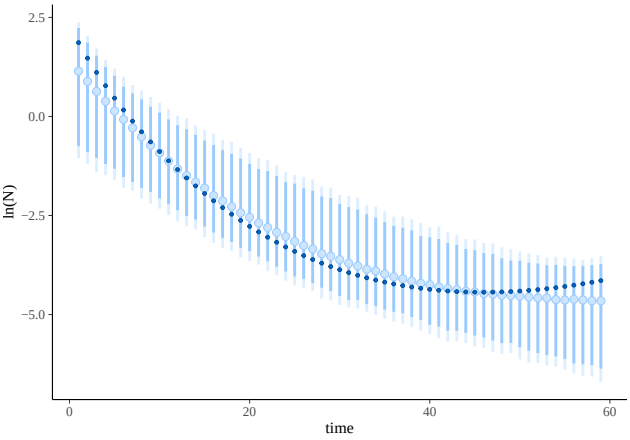
```
df <- data.frame(dat[(2:t), -3])
colnames(df) <- c("Nt1", "xt1")
df$dx1 <- abs(dat$x1[2:t] - dat$x1[1:(t - 1)])
# prepare data in HMSC format
Y <- as.matrix(df$Nt1)
XData <- data.frame(cbind(E[1:(t - 1)], E[1:(t - 1)]^2, abs(dat$x1[2:t] - dat$x1[1:(t -
    1)])))
colnames(XData) <- c("E", "Esq", "dx1")

studyDesign = data.frame(sample = as.factor(1:(t - 1)))
rL = HmscRandomLevel(units = studyDesign$sample)

m.3.hmsc = Hmsc(Y = Y, XData = XData, XFormula = ~E + Esq + dx1, studyDesign = studyDesign,
    ranLevels = list(sample = rL))
# Bayesian model parameters
nChains <- 2
thin <- 5
samples <- 2000
transient <- 1000 * thin
verbose <- 500 * thin
# sample MCMC
m.3.sample <- sampleMcmc(m.3.hmsc, thin = thin, sample = samples, transient = transient,
    nChains = nChains, verbose = verbose)
#> Computing chain 1
#> Chain 1, iteration 2500 of 15000 (transient)
#> Chain 1, iteration 5000 of 15000 (transient)
#> Chain 1, iteration 7500 of 15000 (sampling)
#> Chain 1, iteration 10000 of 15000 (sampling)
#> Chain 1, iteration 12500 of 15000 (sampling)
#> Chain 1, iteration 15000 of 15000 (sampling)
#> Computing chain 2
#> Chain 2, iteration 2500 of 15000 (transient)
#> Chain 2, iteration 5000 of 15000 (transient)
#> Chain 2, iteration 7500 of 15000 (sampling)
#> Chain 2, iteration 10000 of 15000 (sampling)
#> Chain 2, iteration 12500 of 15000 (sampling)
#> Chain 2, iteration 15000 of 15000 (sampling)
m3.post.hmsc <- convertToCodaObject(m.3.sample)
```

```
VP3 <- computeVariancePartitioning(m.3.sample, group = c(1, 1, 1, 2), groupnames = c("Env",
    "x1"))
plotVariancePartitioning(m.3.sample, VP3, cols = c("white", "skyblue", "darkgrey"),
    args.legend = list(cex = 0.75, bg = "transparent"))
```



25

```
# prepare for plot
pred2 <- predict(m.3.sample)
hold1 <- list()
hold2 <- list()
for (i in 1:length(N0)) {
    hold1[[i]] <- as.numeric(Y[, i])
    data <- lapply(pred2, function(x) x[, colnames(x)[i], drop = FALSE])
    hold2[[i]] <- matrix(unlist(data), nrow = length(pred2), byrow = TRUE)
}
y <- unlist(hold1)  # vector of each species' observed N across time points
yrep <- as.matrix(data.frame(hold2))
### Mod2 plot ###
par(mgp = c(2, 0.45, 0), tcl = -0.4, mar = c(1.3, 1.2, 0, 0))
color_scheme_set("brightblue")
# ppc_dens_overlay(y, yrep[1:50, ])
ppc_intervals(y, yrep, x = dat$time[1:(t - 1)], prob = 0.95) + labs(x = "time", y = "ln(N)",
    ) + theme(plot.margin = unit(c(0, 0, 0, 0), "cm")) + theme(legend.position = "none")
```

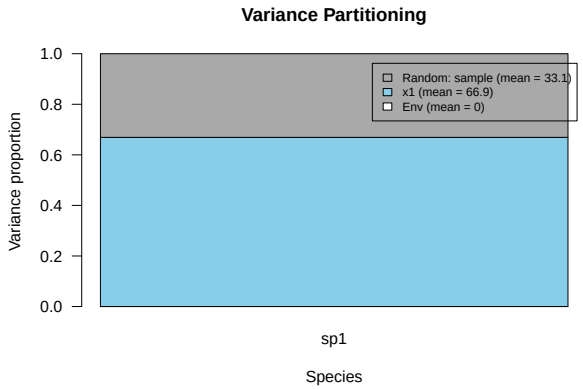


26

```
### Model 4, h2=0.03 ###
dat <- cbind(log(gdat$N1[gdat$h2 == 0.03]), gdat$x1[gdat$h2 == 0.03])
dat <- as.data.frame(dat)
colnames(dat) <- c("N1", "x1")
dat$time <- 1:t
dat <- as.data.frame(dat)
```

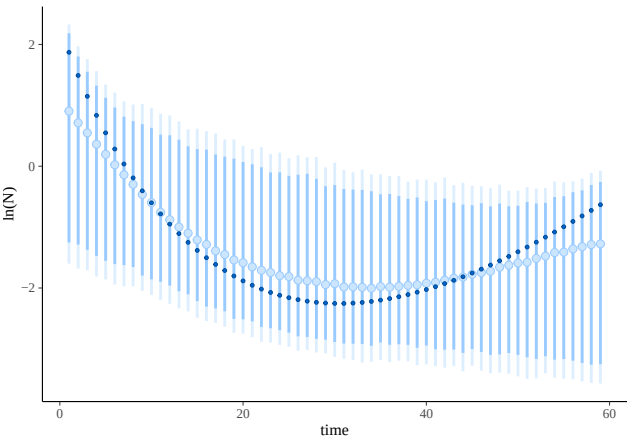
```
df <- data.frame(dat[(2:t), -3])
colnames(df) <- c("Nt1", "xt1")
df$dx1 <- abs(dat$x1[2:t] - dat$x1[1:(t - 1)])
# prepare data in HMSC format
Y <- as.matrix(df$Nt1)
XData <- data.frame(cbind(E[1:(t - 1)], E[1:(t - 1)]^2, abs(dat$x1[2:t] - dat$x1[1:(t -
    1)])))
colnames(XData) <- c("E", "Esq", "dx1")

studyDesign = data.frame(sample = as.factor(1:(t - 1)))
rL = HmscRandomLevel(units = studyDesign$sample)

m.4.hmsc = Hmsc(Y = Y, XData = XData, XFormula = ~E + Esq + dx1, studyDesign = studyDesign,
    ranLevels = list(sample = rL))
# Bayesian model parameters
nChains <- 2
thin <- 5
samples <- 2000
transient <- 1000 * thin
verbose <- 500 * thin
# sample MCMC
m.4.sample <- sampleMcmc(m.4.hmsc, thin = thin, sample = samples, transient = transient,
    nChains = nChains, verbose = verbose)
#> Computing chain 1
#> Chain 1, iteration 2500 of 15000 (transient)
#> Chain 1, iteration 5000 of 15000 (transient)
#> Chain 1, iteration 7500 of 15000 (sampling)
#> Chain 1, iteration 10000 of 15000 (sampling)
#> Chain 1, iteration 12500 of 15000 (sampling)
#> Chain 1, iteration 15000 of 15000 (sampling)
#> Computing chain 2
#> Chain 2, iteration 2500 of 15000 (transient)
#> Chain 2, iteration 5000 of 15000 (transient)
#> Chain 2, iteration 7500 of 15000 (sampling)
#> Chain 2, iteration 10000 of 15000 (sampling)
#> Chain 2, iteration 12500 of 15000 (sampling)
#> Chain 2, iteration 15000 of 15000 (sampling)
m4.post.hmsc <- convertToCodaObject(m.4.sample)
```

```
VP4 <- computeVariancePartitioning(m.4.sample, group = c(1, 1, 1, 2), groupnames = c("Env",
    "x1"))
plotVariancePartitioning(m.4.sample, VP4, cols = c("white", "skyblue", "darkgrey"),
    args.legend = list(cex = 0.75, bg = "transparent"))
```



27

```
# prepare for plot
pred2 <- predict(m.4.sample)
hold1 <- list()
hold2 <- list()
for (i in 1:length(N0)) {
    hold1[[i]] <- as.numeric(Y[, i])
    data <- lapply(pred2, function(x) x[, colnames(x)[i], drop = FALSE])
    hold2[[i]] <- matrix(unlist(data), nrow = length(pred2), byrow = TRUE)
}
y <- unlist(hold1)  # vector of each species' observed N across time points
yrep <- as.matrix(data.frame(hold2))
### Mod2 plot ###
par(mgp = c(2, 0.45, 0), tcl = -0.4, mar = c(1.3, 1.2, 0, 0))
color_scheme_set("brightblue")
# ppc_dens_overlay(y, yrep[1:50, ])
ppc_intervals(y, yrep, x = dat$time[1:(t - 1)], prob = 0.95) + labs(x = "time", y = "ln(N)",
    ) + theme(plot.margin = unit(c(0, 0, 0, 0), "cm")) + theme(legend.position = "none")
```

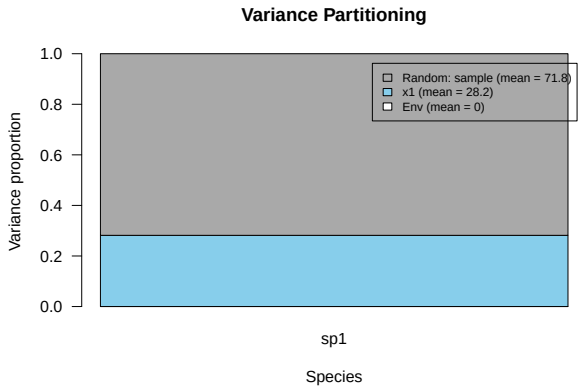


28

```
### Model 5, h2=0.04 ###
dat <- cbind(log(gdat$N1[gdat$h2 == 0.04]), gdat$x1[gdat$h2 == 0.04])
dat <- as.data.frame(dat)
colnames(dat) <- c("N1", "x1")
dat$time <- 1:t
dat <- as.data.frame(dat)
```

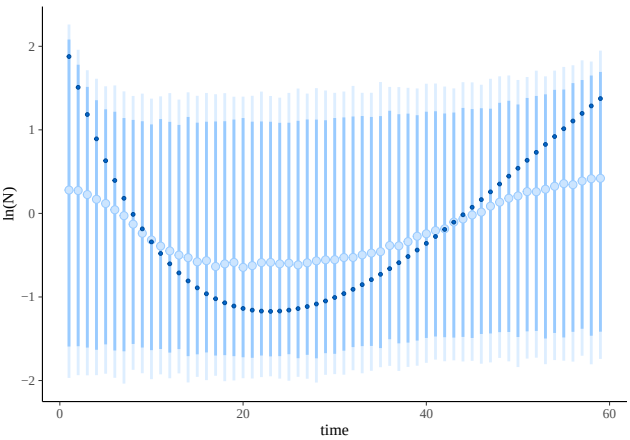
```
df <- data.frame(dat[(2:t), -3])
colnames(df) <- c("Nt1", "xt1")
df$dx1 <- abs(dat$x1[2:t] - dat$x1[1:(t - 1)])
# prepare data in HMSC format
Y <- as.matrix(df$Nt1)
XData <- data.frame(cbind(E[1:(t - 1)], E[1:(t - 1)]^2, abs(dat$x1[2:t] - dat$x1[1:(t -
    1)])))
colnames(XData) <- c("E", "Esq", "dx1")

studyDesign = data.frame(sample = as.factor(1:(t - 1)))
rL = HmscRandomLevel(units = studyDesign$sample)

m.5.hmsc = Hmsc(Y = Y, XData = XData, XFormula = ~E + Esq + dx1, studyDesign = studyDesign,
    ranLevels = list(sample = rL))
# Bayesian model parameters
nChains <- 2
thin <- 5
samples <- 2000
transient <- 1000 * thin
verbose <- 500 * thin
# sample MCMC
m.5.sample <- sampleMcmc(m.5.hmsc, thin = thin, sample = samples, transient = transient,
    nChains = nChains, verbose = verbose)
#> Computing chain 1
#> Chain 1, iteration 2500 of 15000 (transient)
#> Chain 1, iteration 5000 of 15000 (transient)
#> Chain 1, iteration 7500 of 15000 (sampling)
#> Chain 1, iteration 10000 of 15000 (sampling)
#> Chain 1, iteration 12500 of 15000 (sampling)
#> Chain 1, iteration 15000 of 15000 (sampling)
#> Computing chain 2
#> Chain 2, iteration 2500 of 15000 (transient)
#> Chain 2, iteration 5000 of 15000 (transient)
#> Chain 2, iteration 7500 of 15000 (sampling)
#> Chain 2, iteration 10000 of 15000 (sampling)
#> Chain 2, iteration 12500 of 15000 (sampling)
#> Chain 2, iteration 15000 of 15000 (sampling)
m5.post.hmsc <- convertToCodaObject(m.5.sample)
```

```
VP5 <- computeVariancePartitioning(m.5.sample, group = c(1, 1, 1, 2), groupnames = c("Env",
    "x1"))
plotVariancePartitioning(m.5.sample, VP5, cols = c("white", "skyblue", "darkgrey"),
    args.legend = list(cex = 0.75, bg = "transparent"))
```



29

```
# prepare for plot
pred2 <- predict(m.5.sample)
hold1 <- list()
hold2 <- list()
for (i in 1:length(N0)) {
    hold1[[i]] <- as.numeric(Y[, i])
    data <- lapply(pred2, function(x) x[, colnames(x)[i], drop = FALSE])
    hold2[[i]] <- matrix(unlist(data), nrow = length(pred2), byrow = TRUE)
}
y <- unlist(hold1)  # vector of each species' observed N across time points
yrep <- as.matrix(data.frame(hold2))
### Mod2 plot ###
par(mgp = c(2, 0.45, 0), tcl = -0.4, mar = c(1.3, 1.2, 0, 0))
color_scheme_set("brightblue")
# ppc_dens_overlay(y, yrep[1:50, ])
ppc_intervals(y, yrep, x = dat$time[1:(t - 1)], prob = 0.95) + labs(x = "time", y = "ln(N)",
    ) + theme(plot.margin = unit(c(0, 0, 0, 0), "cm")) + theme(legend.position = "none")
```

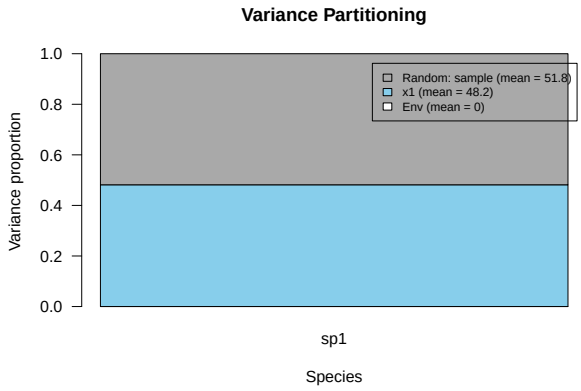


30

```
### Model 6, h2=0.05 ###
dat <- cbind(log(gdat$N1[gdat$h2 == 0.05]), gdat$x1[gdat$h2 == 0.05])
dat <- as.data.frame(dat)
colnames(dat) <- c("N1", "x1")
dat$time <- 1:t
dat <- as.data.frame(dat)
```

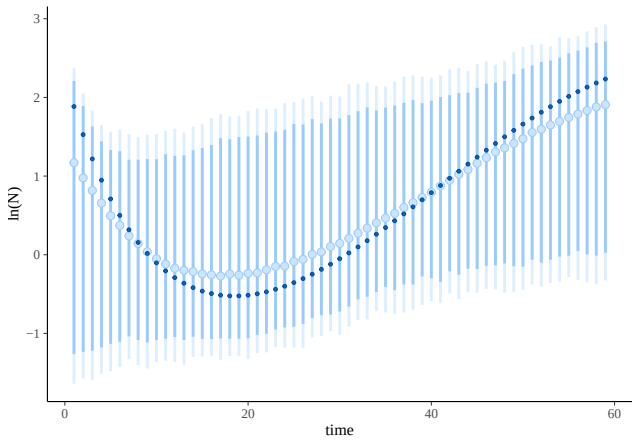
```
df <- data.frame(dat[(2:t), -3])
colnames(df) <- c("Nt1", "xt1")
df$dx1 <- abs(dat$x1[2:t] - dat$x1[1:(t - 1)])
# prepare data in HMSC format
Y <- as.matrix(df$Nt1)
XData <- data.frame(cbind(E[1:(t - 1)], E[1:(t - 1)]^2, abs(dat$x1[2:t] - dat$x1[1:(t -
    1)])))
colnames(XData) <- c("E", "Esq", "dx1")

studyDesign = data.frame(sample = as.factor(1:(t - 1)))
rL = HmscRandomLevel(units = studyDesign$sample)

m.6.hmsc = Hmsc(Y = Y, XData = XData, XFormula = ~E + Esq + dx1, studyDesign = studyDesign,
    ranLevels = list(sample = rL))
# Bayesian model parameters
nChains <- 2
thin <- 5
samples <- 2000
transient <- 1000 * thin
verbose <- 500 * thin
# sample MCMC
m.6.sample <- sampleMcmc(m.6.hmsc, thin = thin, sample = samples, transient = transient,
    nChains = nChains, verbose = verbose)
#> Computing chain 1
#> Chain 1, iteration 2500 of 15000 (transient)
#> Chain 1, iteration 5000 of 15000 (transient)
#> Chain 1, iteration 7500 of 15000 (sampling)
#> Chain 1, iteration 10000 of 15000 (sampling)
#> Chain 1, iteration 12500 of 15000 (sampling)
#> Chain 1, iteration 15000 of 15000 (sampling)
#> Computing chain 2
#> Chain 2, iteration 2500 of 15000 (transient)
#> Chain 2, iteration 5000 of 15000 (transient)
#> Chain 2, iteration 7500 of 15000 (sampling)
#> Chain 2, iteration 10000 of 15000 (sampling)
#> Chain 2, iteration 12500 of 15000 (sampling)
#> Chain 2, iteration 15000 of 15000 (sampling)
m6.post.hmsc <- convertToCodaObject(m.6.sample)
```

```
VP6 <- computeVariancePartitioning(m.6.sample, group = c(1, 1, 1, 2), groupnames = c("Env",
    "x1"))
plotVariancePartitioning(m.6.sample, VP6, cols = c("white", "skyblue", "darkgrey"),
    args.legend = list(cex = 0.75, bg = "transparent"))
```



31

```
# prepare for plot
pred2 <- predict(m.6.sample)
hold1 <- list()
hold2 <- list()
for (i in 1:length(N0)) {
    hold1[[i]] <- as.numeric(Y[, i])
    data <- lapply(pred2, function(x) x[, colnames(x)[i], drop = FALSE])
    hold2[[i]] <- matrix(unlist(data), nrow = length(pred2), byrow = TRUE)
}
y <- unlist(hold1)  # vector of each species' observed N across time points
yrep <- as.matrix(data.frame(hold2))
### Mod2 plot ###
par(mgp = c(2, 0.45, 0), tcl = -0.4, mar = c(1.3, 1.2, 0, 0))
color_scheme_set("brightblue")
# ppc_dens_overlay(y, yrep[1:50, ])
ppc_intervals(y, yrep, x = dat$time[1:(t - 1)], prob = 0.95) + labs(x = "time", y = "ln(N)",
    ) + theme(plot.margin = unit(c(0, 0, 0, 0), "cm")) + theme(legend.position = "none")
```

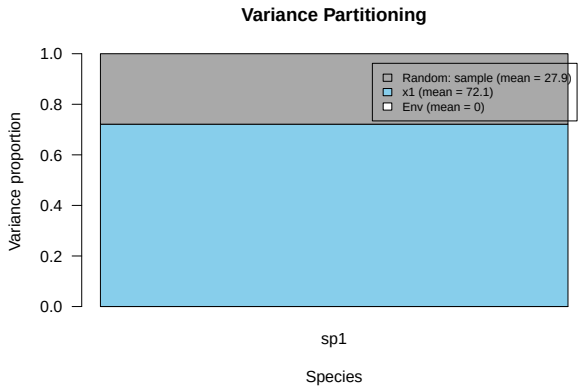


32

```
### Model 7, h2=0.075 ###
dat <- cbind(log(gdat$N1[gdat$h2 == 0.075]), gdat$x1[gdat$h2 == 0.075])
dat <- as.data.frame(dat)
colnames(dat) <- c("N1", "x1")
dat$time <- 1:t
dat <- as.data.frame(dat)
```

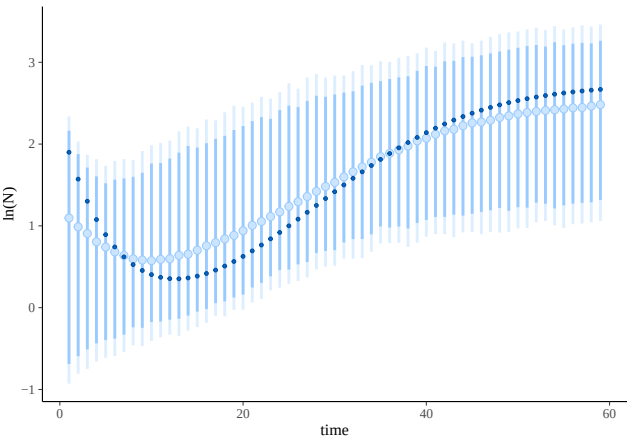
```
df <- data.frame(dat[(2:t), -3])
colnames(df) <- c("Nt1", "xt1")
df$dx1 <- abs(dat$x1[2:t] - dat$x1[1:(t - 1)])
# prepare data in HMSC format
Y <- as.matrix(df$Nt1)
XData <- data.frame(cbind(E[1:(t - 1)], E[1:(t - 1)]^2, abs(dat$x1[2:t] - dat$x1[1:(t -
    1)])))
colnames(XData) <- c("E", "Esq", "dx1")

studyDesign = data.frame(sample = as.factor(1:(t - 1)))
rL = HmscRandomLevel(units = studyDesign$sample)

m.7.hmsc = Hmsc(Y = Y, XData = XData, XFormula = ~E + Esq + dx1, studyDesign = studyDesign,
    ranLevels = list(sample = rL))
# Bayesian model parameters
nChains <- 2
thin <- 5
samples <- 2000
transient <- 1000 * thin
verbose <- 500 * thin
# sample MCMC
m.7.sample <- sampleMcmc(m.7.hmsc, thin = thin, sample = samples, transient = transient,
    nChains = nChains, verbose = verbose)
#> Computing chain 1
#> Chain 1, iteration 2500 of 15000 (transient)
#> Chain 1, iteration 5000 of 15000 (transient)
#> Chain 1, iteration 7500 of 15000 (sampling)
#> Chain 1, iteration 10000 of 15000 (sampling)
#> Chain 1, iteration 12500 of 15000 (sampling)
#> Chain 1, iteration 15000 of 15000 (sampling)
#> Computing chain 2
#> Chain 2, iteration 2500 of 15000 (transient)
#> Chain 2, iteration 5000 of 15000 (transient)
#> Chain 2, iteration 7500 of 15000 (sampling)
#> Chain 2, iteration 10000 of 15000 (sampling)
#> Chain 2, iteration 12500 of 15000 (sampling)
#> Chain 2, iteration 15000 of 15000 (sampling)
m7.post.hmsc <- convertToCodaObject(m.7.sample)
```

```
VP7 <- computeVariancePartitioning(m.7.sample, group = c(1, 1, 1, 2), groupnames = c("Env",
    "x1"))
plotVariancePartitioning(m.7.sample, VP7, cols = c("white", "skyblue", "darkgrey"),
    args.legend = list(cex = 0.75, bg = "transparent"))
```



33

```
# prepare for plot
pred2 <- predict(m.7.sample)
hold1 <- list()
hold2 <- list()
for (i in 1:length(N0)) {
    hold1[[i]] <- as.numeric(Y[, i])
    data <- lapply(pred2, function(x) x[, colnames(x)[i], drop = FALSE])
    hold2[[i]] <- matrix(unlist(data), nrow = length(pred2), byrow = TRUE)
}
y <- unlist(hold1)  # vector of each species' observed N across time points
yrep <- as.matrix(data.frame(hold2))
### Mod2 plot ###
par(mgp = c(2, 0.45, 0), tcl = -0.4, mar = c(1.3, 1.2, 0, 0))
color_scheme_set("brightblue")
# ppc_dens_overlay(y, yrep[1:50, ])
ppc_intervals(y, yrep, x = dat$time[1:(t - 1)], prob = 0.95) + labs(x = "time", y = "ln(N)",
    ) + theme(plot.margin = unit(c(0, 0, 0, 0), "cm")) + theme(legend.position = "none")
```

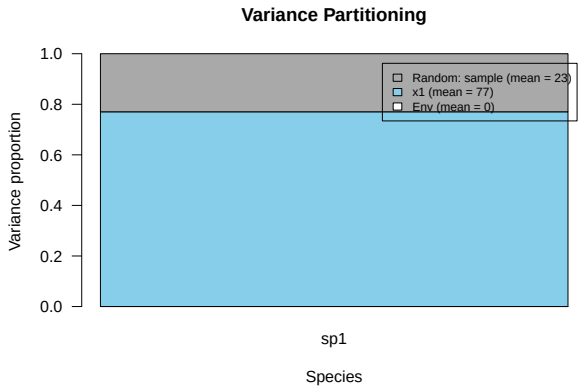


34

```
### Model 8, h2=0.1 ###
dat <- cbind(log(gdat$N1[gdat$h2 == 0.1]), gdat$x1[gdat$h2 == 0.1])
dat <- as.data.frame(dat)
colnames(dat) <- c("N1", "x1")
dat$time <- 1:t
dat <- as.data.frame(dat)
```

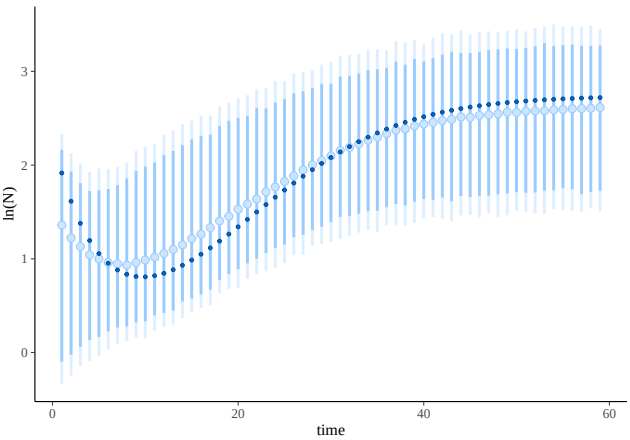
```
df <- data.frame(dat[(2:t), -3])
colnames(df) <- c("Nt1", "xt1")
df$dx1 <- abs(dat$x1[2:t] - dat$x1[1:(t - 1)])
# prepare data in HMSC format
Y <- as.matrix(df$Nt1)
XData <- data.frame(cbind(E[1:(t - 1)], E[1:(t - 1)]^2, abs(dat$x1[2:t] - dat$x1[1:(t -
    1)])))
colnames(XData) <- c("E", "Esq", "dx1")

studyDesign = data.frame(sample = as.factor(1:(t - 1)))
rL = HmscRandomLevel(units = studyDesign$sample)

m.8.hmsc = Hmsc(Y = Y, XData = XData, XFormula = ~E + Esq + dx1, studyDesign = studyDesign,
    ranLevels = list(sample = rL))
# Bayesian model parameters
nChains <- 2
thin <- 5
samples <- 2000
transient <- 1000 * thin
verbose <- 500 * thin
# sample MCMC
m.8.sample <- sampleMcmc(m.8.hmsc, thin = thin, sample = samples, transient = transient,
    nChains = nChains, verbose = verbose)
#> Computing chain 1
#> Chain 1, iteration 2500 of 15000 (transient)
#> Chain 1, iteration 5000 of 15000 (transient)
#> Chain 1, iteration 7500 of 15000 (sampling)
#> Chain 1, iteration 10000 of 15000 (sampling)
#> Chain 1, iteration 12500 of 15000 (sampling)
#> Chain 1, iteration 15000 of 15000 (sampling)
#> Computing chain 2
#> Chain 2, iteration 2500 of 15000 (transient)
#> Chain 2, iteration 5000 of 15000 (transient)
#> Chain 2, iteration 7500 of 15000 (sampling)
#> Chain 2, iteration 10000 of 15000 (sampling)
#> Chain 2, iteration 12500 of 15000 (sampling)
#> Chain 2, iteration 15000 of 15000 (sampling)
m8.post.hmsc <- convertToCodaObject(m.8.sample)
```

```
VP8 <- computeVariancePartitioning(m.8.sample, group = c(1, 1, 1, 2), groupnames = c("Env",
    "x1"))
plotVariancePartitioning(m.8.sample, VP8, cols = c("white", "skyblue", "darkgrey"),
    args.legend = list(cex = 0.75, bg = "transparent"))
```



35

```
# prepare for plot
pred2 <- predict(m.8.sample)
hold1 <- list()
hold2 <- list()
for (i in 1:length(N0)) {
    hold1[[i]] <- as.numeric(Y[, i])
    data <- lapply(pred2, function(x) x[, colnames(x)[i], drop = FALSE])
    hold2[[i]] <- matrix(unlist(data), nrow = length(pred2), byrow = TRUE)
}
y <- unlist(hold1)  # vector of each species' observed N across time points
yrep <- as.matrix(data.frame(hold2))
### Mod2 plot ###
par(mgp = c(2, 0.45, 0), tcl = -0.4, mar = c(1.3, 1.2, 0, 0))
color_scheme_set("brightblue")
# ppc_dens_overlay(y, yrep[1:50, ])
ppc_intervals(y, yrep, x = dat$time[1:(t - 1)], prob = 0.95) + labs(x = "time", y = "ln(N)",
    ) + theme(plot.margin = unit(c(0, 0, 0, 0), "cm")) + theme(legend.position = "none")
```

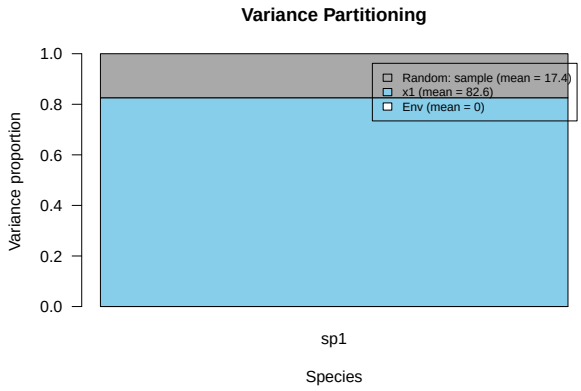


36

```
### Model 9, h2=0.4 ###
dat <- cbind(log(gdat$N1[gdat$h2 == 0.4]), gdat$x1[gdat$h2 == 0.4])
dat <- as.data.frame(dat)
colnames(dat) <- c("N1", "x1")
dat$time <- 1:t
dat <- as.data.frame(dat)
```

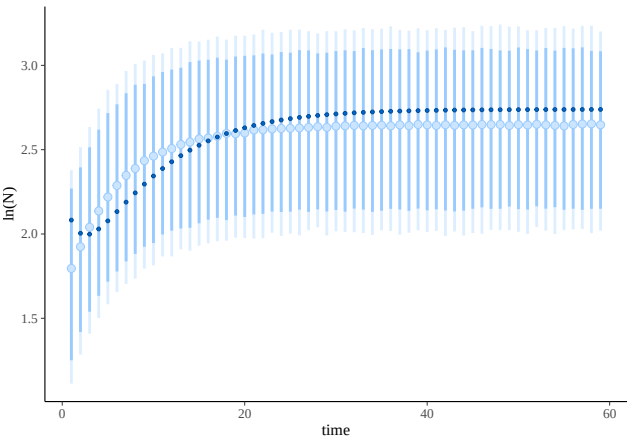
```
df <- data.frame(dat[(2:t), -3])
colnames(df) <- c("Nt1", "xt1")
df$dx1 <- abs(dat$x1[2:t] - dat$x1[1:(t - 1)])
# prepare data in HMSC format
Y <- as.matrix(df$Nt1)
XData <- data.frame(cbind(E[1:(t - 1)], E[1:(t - 1)]^2, abs(dat$x1[2:t] - dat$x1[1:(t -
    1)])))
colnames(XData) <- c("E", "Esq", "dx1")

studyDesign = data.frame(sample = as.factor(1:(t - 1)))
rL = HmscRandomLevel(units = studyDesign$sample)

m.9.hmsc = Hmsc(Y = Y, XData = XData, XFormula = ~E + Esq + dx1, studyDesign = studyDesign,
    ranLevels = list(sample = rL))
# Bayesian model parameters
nChains <- 2
thin <- 5
samples <- 2000
transient <- 1000 * thin
verbose <- 500 * thin
# sample MCMC
m.9.sample <- sampleMcmc(m.9.hmsc, thin = thin, sample = samples, transient = transient,
    nChains = nChains, verbose = verbose)
#> Computing chain 1
#> Chain 1, iteration 2500 of 15000 (transient)
#> Chain 1, iteration 5000 of 15000 (transient)
#> Chain 1, iteration 7500 of 15000 (sampling)
#> Chain 1, iteration 10000 of 15000 (sampling)
#> Chain 1, iteration 12500 of 15000 (sampling)
#> Chain 1, iteration 15000 of 15000 (sampling)
#> Computing chain 2
#> Chain 2, iteration 2500 of 15000 (transient)
#> Chain 2, iteration 5000 of 15000 (transient)
#> Chain 2, iteration 7500 of 15000 (sampling)
#> Chain 2, iteration 10000 of 15000 (sampling)
#> Chain 2, iteration 12500 of 15000 (sampling)
#> Chain 2, iteration 15000 of 15000 (sampling)
m9.post.hmsc <- convertToCodaObject(m.9.sample)
```

```
VP9 <- computeVariancePartitioning(m.9.sample, group = c(1, 1, 1, 2), groupnames = c("Env",
    "x1"))
plotVariancePartitioning(m.9.sample, VP9, cols = c("white", "skyblue", "darkgrey"),
    args.legend = list(cex = 0.75, bg = "transparent"))
```



37

```
# prepare for plot
pred2 <- predict(m.9.sample)
hold1 <- list()
hold2 <- list()
for (i in 1:length(N0)) {
    hold1[[i]] <- as.numeric(Y[, i])
    data <- lapply(pred2, function(x) x[, colnames(x)[i], drop = FALSE])
    hold2[[i]] <- matrix(unlist(data), nrow = length(pred2), byrow = TRUE)
}
y <- unlist(hold1)  # vector of each species' observed N across time points
yrep <- as.matrix(data.frame(hold2))
### Mod2 plot ###
par(mgp = c(2, 0.45, 0), tcl = -0.4, mar = c(1.3, 1.2, 0, 0))
color_scheme_set("brightblue")
# ppc_dens_overlay(y, yrep[1:50, ])
ppc_intervals(y, yrep, x = dat$time[1:(t - 1)], prob = 0.95) + labs(x = "time", y = "ln(N)",
    ) + theme(plot.margin = unit(c(0, 0, 0, 0), "cm")) + theme(legend.position = "none")
```

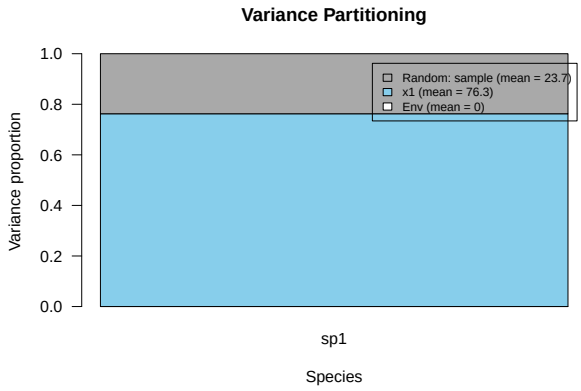


38

```
### Model 10, h2=0.9 ###
dat <- cbind(log(gdat$N1[gdat$h2 == 0.9]), gdat$x1[gdat$h2 == 0.9])
dat <- as.data.frame(dat)
colnames(dat) <- c("N1", "x1")
dat$time <- 1:t
dat <- as.data.frame(dat)
```

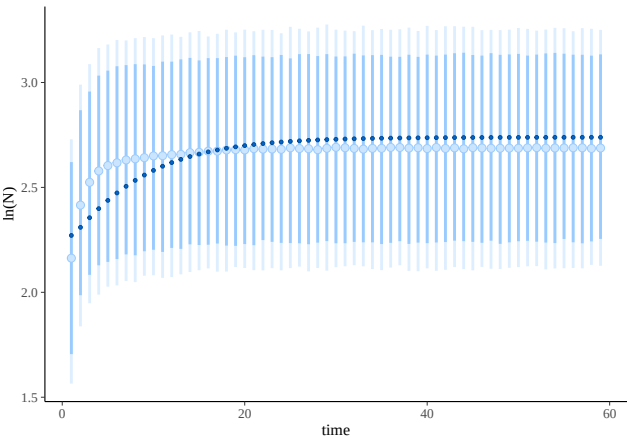
```
df <- data.frame(dat[(2:t), -3])
colnames(df) <- c("Nt1", "xt1")
df$dx1 <- abs(dat$x1[2:t] - dat$x1[1:(t - 1)])
# prepare data in HMSC format
Y <- as.matrix(df$Nt1)
XData <- data.frame(cbind(E[1:(t - 1)], E[1:(t - 1)]^2, abs(dat$x1[2:t] - dat$x1[1:(t -
    1)])))
colnames(XData) <- c("E", "Esq", "dx1")

studyDesign = data.frame(sample = as.factor(1:(t - 1)))
rL = HmscRandomLevel(units = studyDesign$sample)

m.10.hmsc = Hmsc(Y = Y, XData = XData, XFormula = ~E + Esq + dx1, studyDesign = studyDesign,
    ranLevels = list(sample = rL))
# Bayesian model parameters
nChains <- 2
thin <- 5
samples <- 2000
transient <- 1000 * thin
verbose <- 500 * thin
# sample MCMC
m.10.sample <- sampleMcmc(m.10.hmsc, thin = thin, sample = samples, transient = transient,
    nChains = nChains, verbose = verbose)
#> Computing chain 1
#> Chain 1, iteration 2500 of 15000 (transient)
#> Chain 1, iteration 5000 of 15000 (transient)
#> Chain 1, iteration 7500 of 15000 (sampling)
#> Chain 1, iteration 10000 of 15000 (sampling)
#> Chain 1, iteration 12500 of 15000 (sampling)
#> Chain 1, iteration 15000 of 15000 (sampling)
#> Computing chain 2
#> Chain 2, iteration 2500 of 15000 (transient)
#> Chain 2, iteration 5000 of 15000 (transient)
#> Chain 2, iteration 7500 of 15000 (sampling)
#> Chain 2, iteration 10000 of 15000 (sampling)
#> Chain 2, iteration 12500 of 15000 (sampling)
#> Chain 2, iteration 15000 of 15000 (sampling)
m10.post.hmsc <- convertToCodaObject(m.10.sample)
```

```
VP10 <- computeVariancePartitioning(m.10.sample, group = c(1, 1, 1, 2), groupnames = c("Env",
    "x1"))
plotVariancePartitioning(m.10.sample, VP10, cols = c("white", "skyblue", "darkgrey"),
    args.legend = list(cex = 0.75, bg = "transparent"))
```



39

```
# prepare for plot
pred2 <- predict(m.10.sample)
hold1 <- list()
hold2 <- list()
for (i in 1:length(N0)) {
    hold1[[i]] <- as.numeric(Y[, i])
    data <- lapply(pred2, function(x) x[, colnames(x)[i], drop = FALSE])
    hold2[[i]] <- matrix(unlist(data), nrow = length(pred2), byrow = TRUE)
}
y <- unlist(hold1)  # vector of each species' observed N across time points
yrep <- as.matrix(data.frame(hold2))
### Mod2 plot ###
par(mgp = c(2, 0.45, 0), tcl = -0.4, mar = c(1.3, 1.2, 0, 0))
color_scheme_set("brightblue")
# ppc_dens_overlay(y, yrep[1:50, ])
ppc_intervals(y, yrep, x = dat$time[1:(t - 1)], prob = 0.95) + labs(x = "time", y = "ln(N)",
    ) + theme(plot.margin = unit(c(0, 0, 0, 0), "cm")) + theme(legend.position = "none")
```

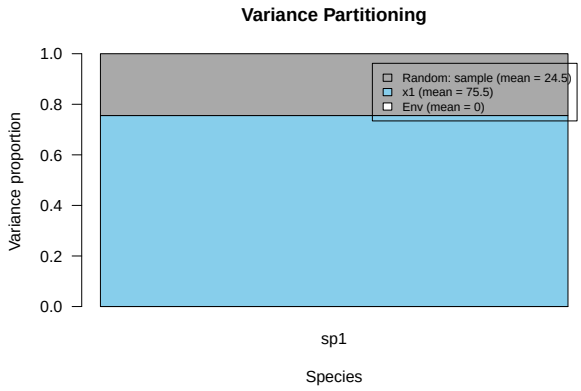


40

```
### Model 11, h2=1 ###
dat <- cbind(log(gdat$N1[gdat$h2 == 1]), gdat$x1[gdat$h2 == 1])
dat <- as.data.frame(dat)
colnames(dat) <- c("N1", "x1")
dat$time <- 1:t
dat <- as.data.frame(dat)
```

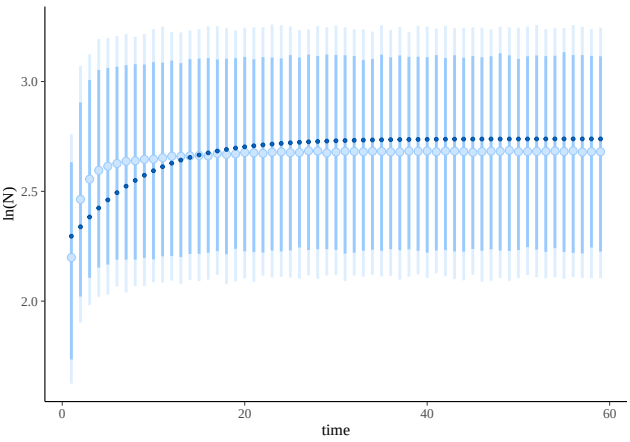
```
df <- data.frame(dat[(2:t), -3])
colnames(df) <- c("Nt1", "xt1")
df$dx1 <- abs(dat$x1[2:t] - dat$x1[1:(t - 1)])
# prepare data in HMSC format
Y <- as.matrix(df$Nt1)
XData <- data.frame(cbind(E[1:(t - 1)], E[1:(t - 1)]^2, abs(dat$x1[2:t] - dat$x1[1:(t -
    1)])))
colnames(XData) <- c("E", "Esq", "dx1")

studyDesign = data.frame(sample = as.factor(1:(t - 1)))
rL = HmscRandomLevel(units = studyDesign$sample)

m.11.hmsc = Hmsc(Y = Y, XData = XData, XFormula = ~E + Esq + dx1, studyDesign = studyDesign,
    ranLevels = list(sample = rL))
# Bayesian model parameters
nChains <- 2
thin <- 5
samples <- 2000
transient <- 1000 * thin
verbose <- 500 * thin
# sample MCMC
m.11.sample <- sampleMcmc(m.11.hmsc, thin = thin, sample = samples, transient = transient,
    nChains = nChains, verbose = verbose)
#> Computing chain 1
#> Chain 1, iteration 2500 of 15000 (transient)
#> Chain 1, iteration 5000 of 15000 (transient)
#> Chain 1, iteration 7500 of 15000 (sampling)
#> Chain 1, iteration 10000 of 15000 (sampling)
#> Chain 1, iteration 12500 of 15000 (sampling)
#> Chain 1, iteration 15000 of 15000 (sampling)
#> Computing chain 2
#> Chain 2, iteration 2500 of 15000 (transient)
#> Chain 2, iteration 5000 of 15000 (transient)
#> Chain 2, iteration 7500 of 15000 (sampling)
#> Chain 2, iteration 10000 of 15000 (sampling)
#> Chain 2, iteration 12500 of 15000 (sampling)
#> Chain 2, iteration 15000 of 15000 (sampling)
m11.post.hmsc <- convertToCodaObject(m.11.sample)
```

```
VP11 <- computeVariancePartitioning(m.11.sample, group = c(1, 1, 1, 2), groupnames = c("Env",
    "x1"))
plotVariancePartitioning(m.11.sample, VP11, cols = c("white", "skyblue", "darkgrey"),
    args.legend = list(cex = 0.75, bg = "transparent"))
```



41

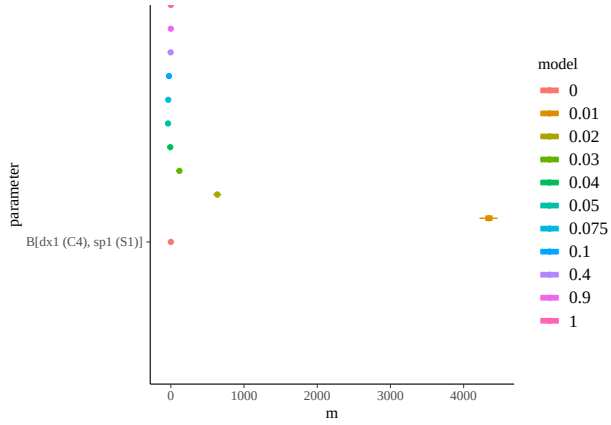
```
# prepare for plot
pred2 <- predict(m.11.sample)
hold1 <- list()
hold2 <- list()
for (i in 1:length(N0)) {
    hold1[[i]] <- as.numeric(Y[, i])
    data <- lapply(pred2, function(x) x[, colnames(x)[i], drop = FALSE])
    hold2[[i]] <- matrix(unlist(data), nrow = length(pred2), byrow = TRUE)
}
y <- unlist(hold1)  # vector of each species' observed N across time points
yrep <- as.matrix(data.frame(hold2))
### Mod2 plot ###
par(mgp = c(2, 0.45, 0), tcl = -0.4, mar = c(1.3, 1.2, 0, 0))
color_scheme_set("brightblue")
# ppc_dens_overlay(y, yrep[1:50, ])
ppc_intervals(y, yrep, x = dat$time[1:(t - 1)], prob = 0.95) + labs(x = "time", y = "ln(N)",
    ) + theme(plot.margin = unit(c(0, 0, 0, 0), "cm")) + theme(legend.position = "none")
```



42

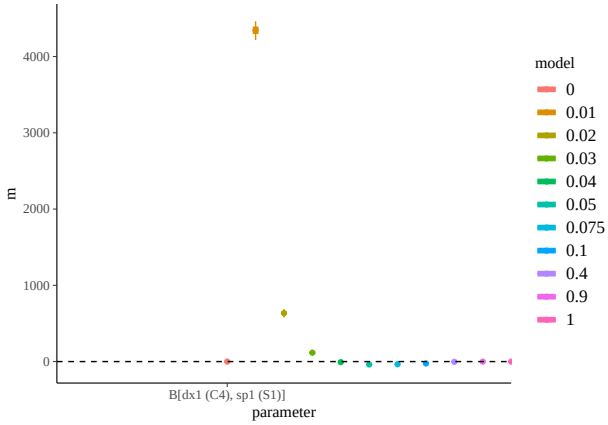
```
mc_dat_1 <- mcmc_intervals_data(m1.post.hmsc$Beta, pars = "B[dx1 (C4), sp1 (S1)]")
mc_dat_2 <- mcmc_intervals_data(m2.post.hmsc$Beta, pars = "B[dx1 (C4), sp1 (S1)]")
mc_dat_3 <- mcmc_intervals_data(m3.post.hmsc$Beta, pars = "B[dx1 (C4), sp1 (S1)]")
mc_dat_4 <- mcmc_intervals_data(m4.post.hmsc$Beta, pars = "B[dx1 (C4), sp1 (S1)]")
mc_dat_5 <- mcmc_intervals_data(m5.post.hmsc$Beta, pars = "B[dx1 (C4), sp1 (S1)]")
mc_dat_6 <- mcmc_intervals_data(m6.post.hmsc$Beta, pars = "B[dx1 (C4), sp1 (S1)]")
```

```
mc_dat_7 <- mcmc_intervals_data(m7.post.hmsc$Beta, pars = "B[dx1 (C4), sp1 (S1)]")
mc_dat_8 <- mcmc_intervals_data(m8.post.hmsc$Beta, pars = "B[dx1 (C4), sp1 (S1)]")
mc_dat_9 <- mcmc_intervals_data(m9.post.hmsc$Beta, pars = "B[dx1 (C4), sp1 (S1)]")
mc_dat_10 <- mcmc_intervals_data(m10.post.hmsc$Beta, pars = "B[dx1 (C4), sp1 (S1)]")
mc_dat_11 <- mcmc_intervals_data(m11.post.hmsc$Beta, pars = "B[dx1 (C4), sp1 (S1)]")


combined <- rbind(mc_dat_1, mc_dat_2, mc_dat_3, mc_dat_4, mc_dat_5, mc_dat_6, mc_dat_7,
    mc_dat_8, mc_dat_9, mc_dat_10, mc_dat_11)
combined$model <- rep(c("0", "0.01", "0.02", "0.03", "0.04", "0.05", "0.075", "0.1",
    "0.4", "0.9", "1"), times = c(dim(mc_dat_1)[1], dim(mc_dat_2)[1], dim(mc_dat_3)[1],
    dim(mc_dat_4)[1], dim(mc_dat_5)[1], dim(mc_dat_6)[1], dim(mc_dat_7)[1], dim(mc_dat_8)[1],
    dim(mc_dat_9)[1], dim(mc_dat_10)[1], dim(mc_dat_11)[1]))

# make the plot using ggplot
theme_set(bayesplot::theme_default())
pos <- position_nudge(y = c(0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1))
ggplot(combined, aes(x = m, y = parameter, color = model)) + geom_linerange(aes(xmin = l,
    xmax = h), position = pos, size = 2) + geom_linerange(aes(xmin = ll, xmax = hh),
    position = pos) + geom_point(aes(color = model), position = pos)
#> Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
#> i Please use `linewidth` instead.
#> This warning is displayed once every 8 hours.
#> Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
#> generated.
```



43

```
ggplot(combined, aes(x = m, y = parameter, color = model)) + geom_linerange(aes(xmin = l,
    xmax = h), position = pos, size = 2) + geom_linerange(aes(xmin = ll, xmax = hh),
    position = pos) + geom_point(aes(color = model), position = pos) + coord_flip() +
    geom_vline(xintercept = 0, linetype = "dashed")
```



2 References