# Appendix S1. Linear regression models and non-linear population dynamics

Jelena H. Pantel* Ruben J. Hermann†

04 May, 2024

## 1 One species, logistic growth

Population growth over time in a single species is first modelled using a Beverton-Holt (discrete-time, logistic) model (Beverton and Holt (1957)), using an intra-specific competition coefficient for density-dependent growth (Hart and Marshall (2013)).

$$N_{i,t+1} = \frac{r_i N_{i,t}}{1 + \alpha_{ii} N_{i,t}}$$

Note that in this model, the system is at equilibrium when $N_{i,t+1} = N_{i,t}$, and therefore:

$$N^* = N^* \frac{r_i}{1 + \alpha_{ii} N^*}$$

$$1 = \frac{r_i}{1 + \alpha_{ii} N^*}$$

$$N^* = \frac{r_i - 1}{\alpha_{ii}}$$

### 1.1 Population dynamics simulation

In the metacommunity simulation in the main text, a species resides in a site with an initial population size $N_{i,0} \sim Pois(10)$, a growth rate $r_i$ that depends on the local environmental value $E_k$ and the species trait $x_i$, and a fixed intra-specific competition coefficient of $\alpha_{ii} = 0.00125$. We simulate population growth here:

```
set.seed(42)
# Simulate initial species population growth
N1.0 <- rpois(1, 10)
r1.0 <- 1.67
alpha.11 <- 0.00125
# model function
disc_log <- function(r, N0, alpha) {
    Nt1 <- (r * N0)/(1 + alpha * N0)
```

*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

```
    return(Nt1)
}
# Simulation of model for t time steps
t <- 30
N <- rep(NA, t)
N[1] <- N1.0
for (i in 2:t) {
    N[i] <- disc_log(r = r1.0, N0 = N[i - 1], alpha = alpha.11)
}
```

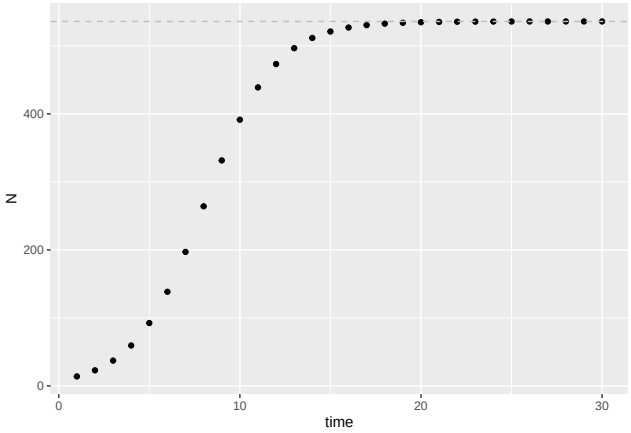



Figure 1: Population size $N$ over time $t$ for a discrete-time logistic growth model, with parameters $r_i = 1.67$, $N_{1,0} = 14$, and $\alpha_{11} = 0.00125$.

## 1.2 Linear statistical model

We fit the population time series data to a first-order auto-regressive model to predict $N_{t+1}$ as a function of $N_t$, and compare that to a linear regression:

$$N_{t+1} = \beta_0 + \beta_1 N_t + \epsilon_t$$

```
# Fit the model
m.1.ar <- arima(x = log(N), order = c(1, 0, 0), include.mean = T, method = "CSS")
m.1.lm <- lm(log(dat$N[2:t]) ~ log(dat$N[1:(t - 1)]))
# plotting the series along with the fitted values
m.1.ar.fit <- log(N) - residuals(m.1.ar)
m.1.lm.fit <- log(dat$N[2:t]) - m.1.lm$resid
dat$ar1.fit <- m.1.ar.fit
dat$lm.fit <- NA
dat$lm.fit[2:t] <- m.1.lm.fit
```

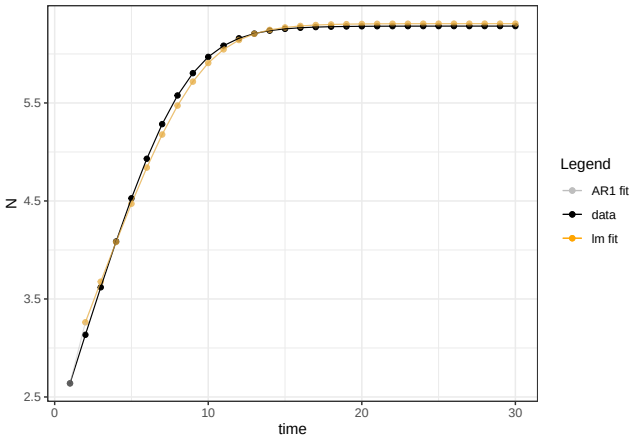



Figure 2: Population size over time (black line) with fitted values from a first-order autoregressive model (red dashed line).

17 The linear model is a good fit, and $N_{t+1}$ and $N_t$ are well-represented by a linear function:

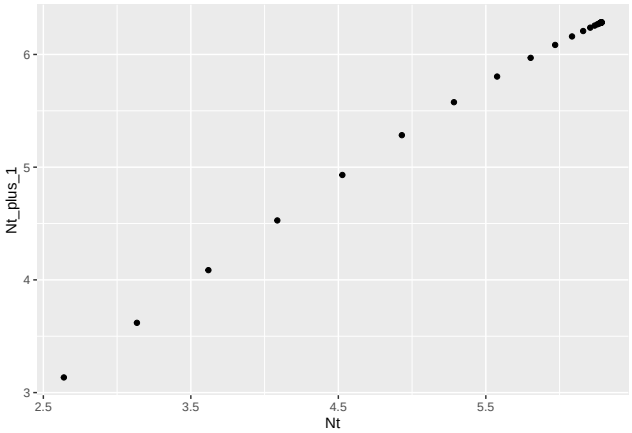



Figure 3: Population size (logarithm) at one time step $N_{t+1}$ as a function of log-population size in the previous time step $N_t$.

18 We also examine density dependence by plotting $\Delta N = N_{t+1} - N_t$ vs. $N_t$:

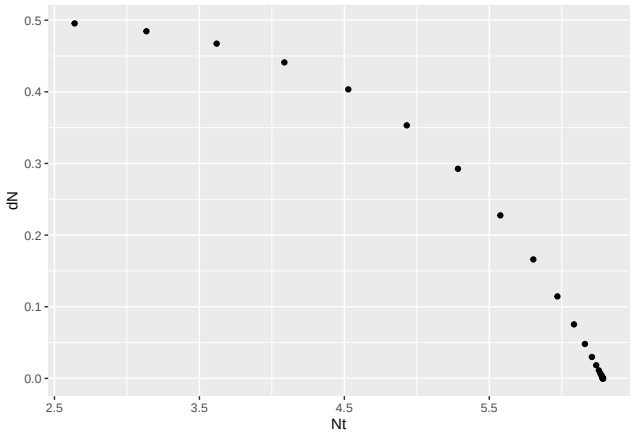



Figure 4: Change in population size from one time step to the next $N_{t+1}$ as a function of $N_{t+1}$

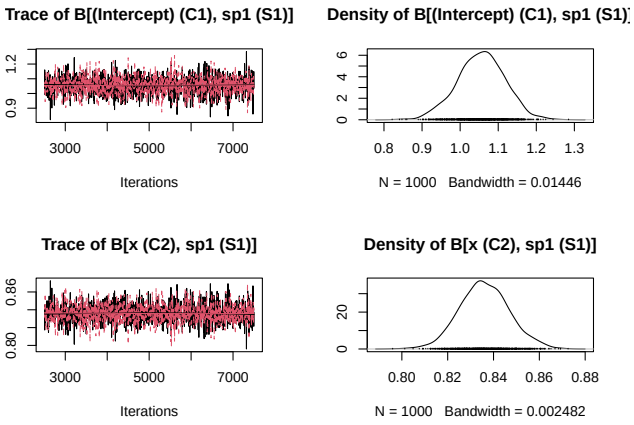
## 1.3 Bayesian linear statistical model: HMSC

We can estimate the same model parameters using HMSC:

```
# prepare data in HMSC format
Y <- as.matrix(log(dat$N[2:t]))
XData <- data.frame(x = log(dat$N[1:(t - 1)]))
m.1.hmsc <- Hmsc(Y = Y, XData = XData, XFormula = ~x)
# Bayesian model parameters
nChains <- 2
thin <- 5
samples <- 1000
transient <- 500 * thin
verbose <- 500 * thin
# sample MCMC
m.1.sample <- sampleMcmc(m.1.hmsc, thin = thin, sample = samples, transient = transient,
    nChains = nChains, verbose = verbose)
#> setting updater$GammaEta=FALSE due to absence of random effects included to the model
#> Computing chain 1
#> Chain 1, iteration 2500 of 7500 (transient)
#> Chain 1, iteration 5000 of 7500 (sampling)
#> Chain 1, iteration 7500 of 7500 (sampling)
#> Computing chain 2
#> Chain 2, iteration 2500 of 7500 (transient)
#> Chain 2, iteration 5000 of 7500 (sampling)
#> Chain 2, iteration 7500 of 7500 (sampling)
```

```
m.post.hmsc <- convertToCodaObject(m.1.sample)
summary(m.post.hmsc$Beta)
#>
#> Iterations = 2505:7500
#> Thinning interval = 5
#> Number of chains = 2
#> Sample size per chain = 1000
#>
#> 1. Empirical mean and standard deviation for each variable,
```

```
#>    plus standard error of the mean:
#>
#>                                    Mean      SD  Naive SE Time-series SE
#> B[(Intercept) (C1), sp1 (S1)] 1.0550 0.06240 0.0013952      0.0013954
#> B[x (C2), sp1 (S1)]           0.8361 0.01083 0.0002422      0.0002415
#>
#> 2. Quantiles for each variable:
#>
#>                                    2.5%    25%    50%    75%  97.5%
#> B[(Intercept) (C1), sp1 (S1)] 0.9288 1.0132 1.0564 1.0974 1.1693
#> B[x (C2), sp1 (S1)]           0.8154 0.8287 0.8358 0.8431 0.8582
plot(m.post.hmsc$Beta)
```




These estimates match well with those from the AR1 and linear model:

```
# AR1 coefficients (recall that the intercept is the term below multipled by 1
# - phi1)
m.1.ar$coef
#>       ar1 intercept
#> 0.8359839 6.4371388
m.1.ar$coef[2] * (1 - m.1.ar$coef[1])
#> intercept
#>  1.055794
# linear model
summary(m.1.lm)$coefficients[1:2, 1:2]
#>                        Estimate  Std. Error
#> (Intercept)           1.0557944 0.054425861
#> log(dat$N[1:(t - 1)]) 0.8359839 0.009441702
# Bayesian estimates
summary(m.post.hmsc$Beta)$statistics[1:2, 1:2]
#>                                    Mean         SD
#> B[(Intercept) (C1), sp1 (S1)] 1.0549596 0.06239605
#> B[x (C2), sp1 (S1)]           0.8360545 0.01082936
```

```
Gradient <- constructGradient(m.1.sample, focalVariable = "x", ngrid = 29)
predY <- predict(m.1.sample, Gradient = Gradient, expected = TRUE)
# preds <-computePredictedValues(m.1.sample)
plotGradient(m.1.sample, Gradient, pred = predY, showData = T, measure = "Y", main = "",
    xlab = "N_t", ylab = "predicted N_t+1")
```

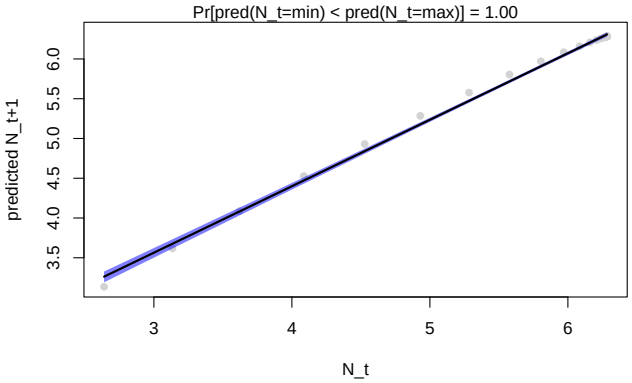
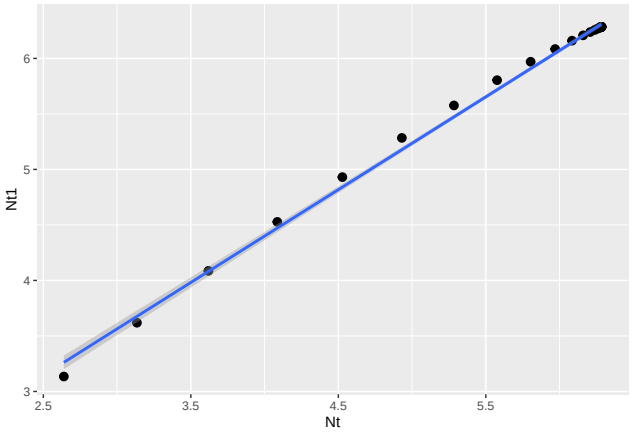



Figure 5: Observed (grey) and model-fit (blue) values for population size at time t (x-axis) and t+1 (y-axis).

```
lm_dat <- data.frame(cbind(log(dat$N[2:t]), log(dat$N[1:(t - 1)])))
colnames(lm_dat) <- c("Nt1", "Nt")
ggplot(lm_dat, aes(Nt, Nt1)) + stat_summary(fun.data = mean_cl_normal) + geom_smooth(method = "lm")
#> `geom_smooth()` using formula = 'y ~ x'
#> Warning: Removed 29 rows containing missing values (`geom_segment()`).
```




## 1.4 Conclusions

In this example, a first-order auto-regressive model works well, bypassing the need to estimate logistic growth parameters $r_i$ and $\alpha_{ii}$. The density-dependence dynamics ($\Delta N \sim f(N_t)$) show an overall declining trend over time. The Bayesian estimation implemented in HMSC gives good parameter estimates.

# 2 One species, logistic growth, environmental covariate

We now consider using a linear model to analyze population growth when the species growth rate is impacted by a single environmental covariate.

## 2.1 Growth depends on environment

First we add environment-dependent growth rate. The growth rate $r_i$ becomes:

$$r_i = \hat{W} e^{-(E - x_{i,t})^2}$$

Here, $\hat{W}$ is the maximal population growth rate (set to 1.67 as above), $E$ is the local environmental trait optimum value, and $x_{i,t}$ is species $i$ trait value at time $t$. We see that if $E = x_{i,t}$ then the growth rate is at the value $r = 1.67$. Here, we begin with $E = x_{i,t} = 0.8$, then simulate the environment $E$ value fluctuating randomly over time, and finally use a linear model to fit $E$ as a covariate.

```
# Simulate initial species population growth with environment fluctuations
N1.0 <- 10
r1.0 <- 1.67
alpha.11 <- 0.00125
E.0 <- 0.8
x1.0 <- 0.8
# model function
disc_log_E <- function(r, N0, alpha, E, x) {
    Nt1 <- ((r * exp(-(E - x)^2)) * N0)/(1 + alpha * N0)
    return(Nt1)
}
# Simulation of model for t time steps
t <- 40
N <- rep(NA, t)
N[1] <- N1.0
E <- rep(NA, t)
E[1] <- E.0
for (i in 2:t) {
    N[i] <- disc_log_E(r = r1.0, N0 = N[i - 1], alpha = alpha.11, E = E[i - 1], x = x1.0)
    E[i] <- E[i - 1] + rnorm(1, 0, 0.1)
}
```

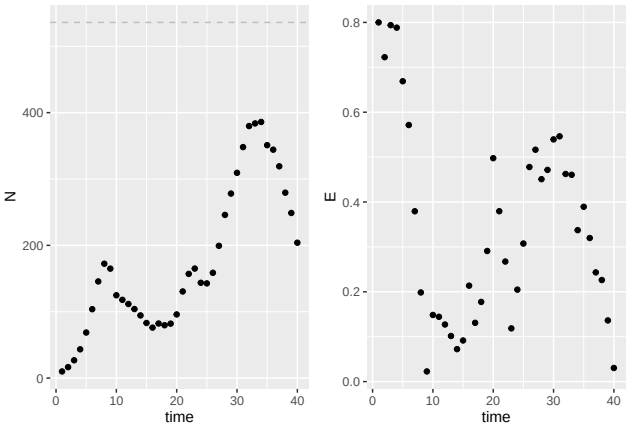



Figure 6: Population size $N$ over time $t$ for a discrete-time logistic growth model, with parameters $r_i = 1.67$, $N_{1,0} = 14$, and $\alpha_{11} = 0.00125$. Relationship between E and Nt is also shown.

## 2.2 Linear statistical model with environmental covariate

We now include environment $E$ as a covariate in the linear model:

$$N_t = \beta_0 + \beta_1 N_{t-1} + \beta_2 E_{t-1} + \epsilon_t$$

```r
# Fit the model
m.2.ar <- arima(x = log(N), order = c(1, 0, 0), include.mean = T, method = "CSS",
    xreg = E)
m.2.lm <- lm(log(dat$N[2:t]) ~ log(dat$N[1:(t - 1)]) + log(E[1:(t - 1)]))
# plotting the series along with the fitted values
m.2.ar.fit <- log(N) - residuals(m.2.ar)
m.2.lm.fit <- log(dat$N[2:t]) - m.2.lm$resid
dat$ar2.fit <- m.2.ar.fit
dat$lm2.fit <- NA
dat$lm2.fit[2:t] <- m.2.lm.fit
```

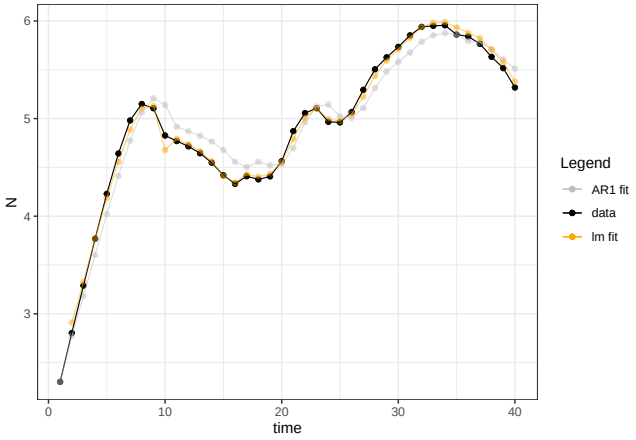



Figure 7: Population size over time (black line) with fitted values from a first-order autoregressive model (red dashed line).

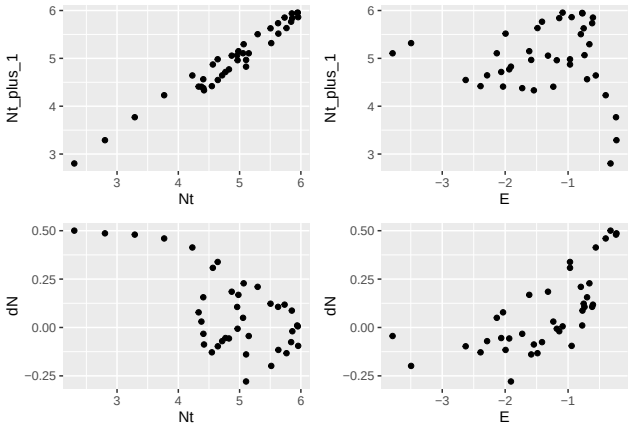



Figure 8: Population size (logarithm) at one time step $N_{t+1}$ as a function of log-population size in the previous time step $N_t$.

The linear model is a good fit when including the environmental covariate. $N_{t+1}$ and $N_t$ can still be captured by a linear relationship. However we see that the relationship between $N_{t+1}$ and $E_t$ is non-linear. This tells us that the lm is good for predictions, but not for inference (for capturing well the relationsip between the predictor and response variable). The use of linear relationships in JSDMs is discussed in (Ingram et al. 2020), and in many applications (e.g. (Erickson and Smith 2023)) quadratic terms are used, which create bell-shaped response curves that may better match species with optimal niches (as opposed to linear,

monotonically increasing relationships between population size and environmental predictors). We thus include a quadratic term for $E_t$ to provide a better fit to the data.

```
df <- data.frame(cbind(log(dat$N[2:t]), log(dat$N[1:(t - 1)]), E[1:(t - 1)], E[1:(t -
    1)]^2))
colnames(df) <- c("Nt1", "Nt", "E", "Esq")
m.2.lm <- lm(Nt1 ~ Nt + E + Esq, data = df)
```

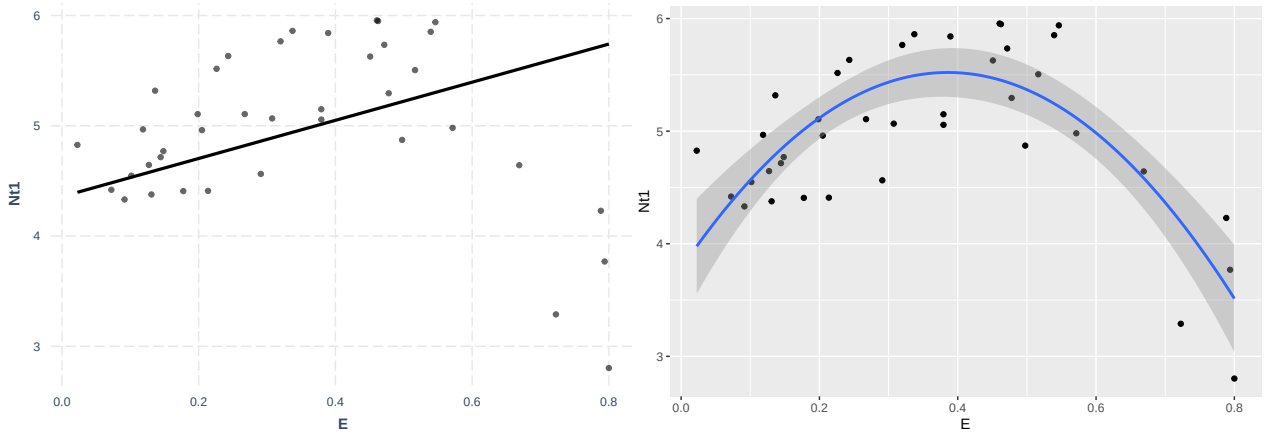


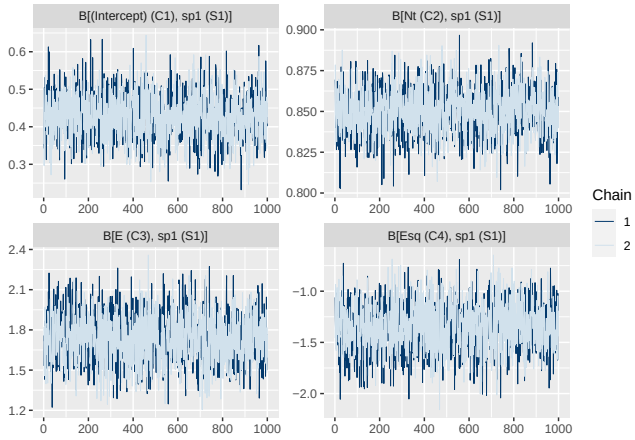

Figure 9: Population size (logarithm) at one time step $N_{t+1}$ as a function of log-population size in the previous time step $N_t$.

## 2.3 Bayesian linear statistical model: HMSC

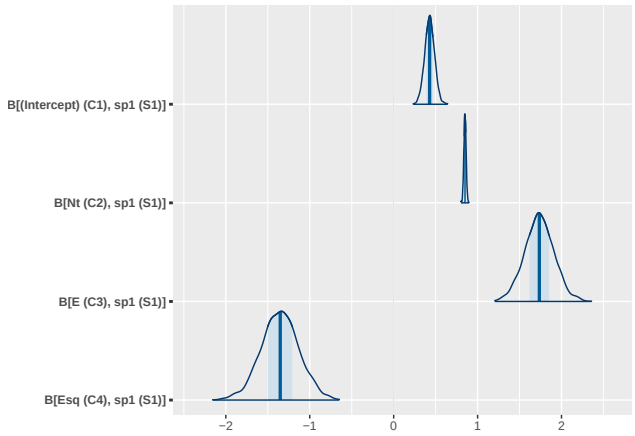
We can estimate the same model parameters using HMSC:

```
# prepare data in HMSC format
Y <- as.matrix(log(dat$N[2:t]))
XData <- df
m.2.hmsc <- Hmsc(Y = Y, XData = XData, XFormula = ~Nt + E + Esq)
# Bayesian model parameters
nChains <- 2
thin <- 5
samples <- 1000
transient <- 500 * thin
verbose <- 500 * thin
# sample MCMC
m.2.sample <- sampleMcmc(m.2.hmsc, thin = thin, sample = samples, transient = transient,
    nChains = nChains, verbose = verbose)
#> setting updater$GammaEta=FALSE due to absence of random effects included to the model
#> Computing chain 1
#> Chain 1, iteration 2500 of 7500 (transient)
#> Chain 1, iteration 5000 of 7500 (sampling)
#> Chain 1, iteration 7500 of 7500 (sampling)
#> Computing chain 2
#> Chain 2, iteration 2500 of 7500 (transient)
#> Chain 2, iteration 5000 of 7500 (sampling)
#> Chain 2, iteration 7500 of 7500 (sampling)
```

```
m2.post.hmsc <- convertToCodaObject(m.2.sample)
summary(m2.post.hmsc$Beta)
#>
#> Iterations = 2505:7500
#> Thinning interval = 5
#> Number of chains = 2
#> Sample size per chain = 1000
#>
#> 1. Empirical mean and standard deviation for each variable,
#>    plus standard error of the mean:
#>
#>                                    Mean      SD  Naive SE Time-series SE
#> B[(Intercept) (C1), sp1 (S1)]  0.4287 0.05840 0.0013059      0.0013051
#> B[Nt (C2), sp1 (S1)]           0.8500 0.01357 0.0003034      0.0003136
#> B[E (C3), sp1 (S1)]            1.7365 0.17407 0.0038922      0.0039946
#> B[Esq (C4), sp1 (S1)]         -1.3544 0.22541 0.0050403      0.0051805
#>
#> 2. Quantiles for each variable:
#>
#>                                   2.5%     25%     50%    75%   97.5%
#> B[(Intercept) (C1), sp1 (S1)]  0.3125  0.3902  0.4285  0.467  0.5435
#> B[Nt (C2), sp1 (S1)]           0.8235  0.8413  0.8500  0.859  0.8768
#> B[E (C3), sp1 (S1)]            1.3967  1.6194  1.7361  1.853  2.0816
#> B[Esq (C4), sp1 (S1)]         -1.8157 -1.5001 -1.3518 -1.207 -0.9151
bayesplot::mcmc_trace(m2.post.hmsc$Beta)
```




```
bayesplot::mcmc_areas(m2.post.hmsc$Beta, area_method = c("equal height"))
```




These estimates match well with those from the AR1 and linear model:

```
# AR1 coefficients (recall that the intercept is the term below multipled by 1
# - phi1)
m.2.ar$coef
#>        ar1  intercept          E
#>  0.8471120  5.4224546 -0.1096173
m.2.ar$coef[2] * (1 - m.2.ar$coef[1])
#> intercept
#>  0.829028
# linear model
summary(m.2.lm)$coefficients[1:4, 1:2]
#>               Estimate Std. Error
#> (Intercept)  0.4286437 0.04917405
#> Nt           0.8502417 0.01151860
#> E            1.7296394 0.14595151
#> Esq         -1.3462240 0.18902070
# Bayesian estimates
summary(m2.post.hmsc$Beta)$statistics[1:4, 1:2]
#>                                      Mean         SD
#> B[(Intercept) (C1), sp1 (S1)]  0.4287409 0.05840051
#> B[Nt (C2), sp1 (S1)]           0.8500472 0.01356690
#> B[E (C3), sp1 (S1)]            1.7365191 0.17406609
#> B[Esq (C4), sp1 (S1)]         -1.3543664 0.22541119
```

We recall that the interpretation of the coefficients in an arimaX (arima with covariates) model is difficult. They do not give the impact on $N_t$ per unit increase in X as in a regression. So we do not interpret the causation implied by the coefficient in the arimaX model. In the regression model, we can see that $E$ has a positive impact on $N_t$.

```
Gradient <- constructGradient(m.2.sample, focalVariable = "E", non.focalVariables = list(Nt = list(2),
    Esq = list(2)), ngrid = 39)
predY <- predict(m.2.sample, XData = Gradient$XDataNew, expected = TRUE)
plotGradient(m.2.sample, Gradient, pred = predY, showData = T, measure = "Y", main = "",
    xlab = "E_t", ylab = "predicted N_t+1")
# Can't figure out curved gradient using E and E^2 pr <-
# predict(m.2.hmsc,m.2.sample,)
```

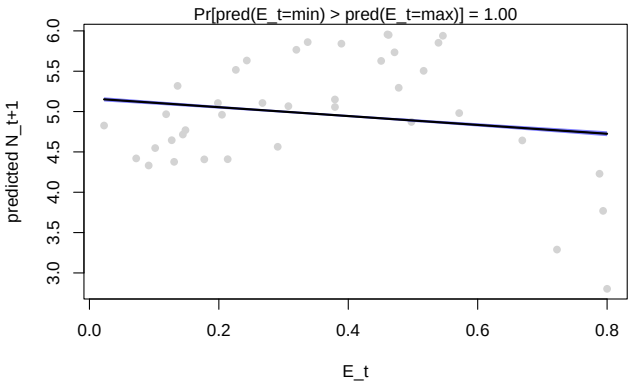



Figure 10: Observed (grey) and model-fit (blue) values for population size at time t (x-axis) and t+1 (y-axis).

## 2.4 Conclusions

In this example, the linear regression again works well to describe the impact of $E_t$ for $N_t$ when using the quadratic formulation. The arimaX model works well for fitting and subsequent prediction, but less well for inference about the impacts of $E$. From the quadratic regression terms for $E$, we correctly see that the population size is maximal at the species trait value and decreases away from that value. We will continue to use log-transformed abundance and now introduce quadratic terms for the environmental parameter.

```
knitr::knit_exit()
```

Beverton, R. J., and S. J. Holt. 1957. On the dynamics of exploited fish populations (Vol. 11). Springer Science & Business Media.

Erickson, K. D., and A. B. Smith. 2023. Modeling the rarest of the rare: A comparison between multi-species distribution models, ensembles of small models, and single-species models at extremely low sample sizes. Ecography 2023:e06500.

Hart, S. P., and D. J. Marshall. 2013. Environmental stress, facilitation, competition, and coexistence. Ecology 94:2719–2731.

Ingram, M., D. Vukcevic, and N. Golding. 2020. Multi-output gaussian processes for species distribution modelling. Methods in Ecology and Evolution 11:1587–1598.