Mpd reproducible start

Anthony Davidson

13/10/2019

Just the start. Here is the basic start and then I will contunie to move files into independent .rmd files and vignettes folder.

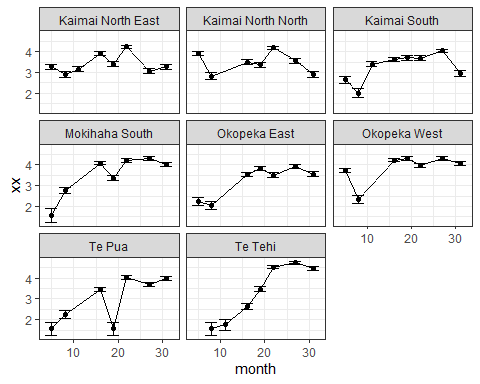
Data

## Joining, by = c("grid", "month")

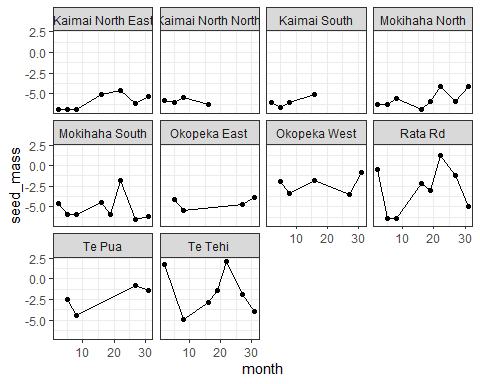
Rats

## Warning: Removed 6 rows containing missing values (geom\_point).

## Warning: Removed 6 rows containing missing values (geom\_errorbar).



Seed



Analysis

Current model

Gompertz model with same parameters for each grid here:

Gompertz model

If the resource changes over time (e.g. seed availability, rainfall….) then both the rate of increase and the carrying capacity will change over time.

An example with and starting value , with a fluctuating resource.

If we set and log-transform equation 2 we get:

This is the process model we can now fit from our discrete time-series data (e.g CR data or indices)

Model fit

This model assumes the same underlying population model at each site - observation error is associated with differences between grids at the same site

* missing values for seed are modelled at the site level
* specify the rainfall lag for the process model in the data statement

mod <- "model {  
  
 # Observation model  
 for(i in 1:nYears) {  
 for(j in 1:nGrids) {  
 Ytau[i, j] ~ dnorm(mu.Ytau, tau[5])  
 Y[i, j] ~ dnorm(X[i, j], Ytau[i, j])  
 # model seed as an overall mean (alpha1), an offset for each year (alpha2) and an offset for each grid (alpha3)  
 seed[i, j] ~ dnorm(mu.r[i, j], tau[1])  
 mu.r[i, j] <- alpha1 + alpha2[i] + alpha3[j]  
 }  
 }  
   
 # priors for the observation and seed models  
 for(i in 1:nYears) {  
 alpha2[i] ~ dnorm(0, tau[2])  
 }  
  
 for(j in 1:nGrids) {  
 alpha3[j] ~ dnorm(0, tau[3])  
 X[1, j] ~ dnorm(0, 0.1)   
 simX[1, j] <- X[1, j]   
 }  
   
 # Process model  
 for(j in 1:nGrids) {  
 for(i in 2:nYears) {  
  
 X[i, j] ~ dnorm(predX[i, j], tau[4])  
 # include rainfall on the raw scale (e.g. exp(log(rain)) - this fits much better  
 predX[i, j] <- X[i-1, j] + (r + beta.seed \* (seed[i-lag, j]) - d \* X[i-1, j]) \* td[i]  
  
 # get independent estimates for the fitted process model  
 simX[i, j] ~ dnorm(predsimX[i, j], tau[4])  
 predsimX[i, j] <- simX[i-1, j] + (r + beta.seed \* (seed[i-lag, j]) - d \* simX[i-1, j]) \* td[i]  
 }  
 }  
  
 # priors  
 mu.Ytau ~ dnorm(0, 0.001)T(0, )  
 r ~ dnorm(0, 0.001)T(0, )  
 beta.seed ~ dnorm(0, 0.01)  
   
 d ~ dnorm(0, 0.01)T(0, )  
 alpha1 ~ dnorm(0, 0.001)  
  
 for(i in 1:5) {  
 tau[i] <- 1 / (sigma[i] \* sigma[i])  
 sigma[i] ~ dunif(0, 100)  
 }  
  
 }"  
  
# write model  
write(mod, "model.txt")

##   
## Processing function input.......   
##   
## Done.   
##   
## Beginning parallel processing using 3 cores. Console output will be suppressed.  
##   
## Parallel processing completed.  
##   
## Calculating statistics.......

## Warning in doTryCatch(return(expr), name, parentenv, handler): At least one  
## Rhat value could not be calculated.

##   
## Done.

## mean 2.5% 97.5% Rhat n.eff  
## r 6.4042161 4.08661403 8.7846278 1.0032577 852  
## d 1.7464941 1.09469572 2.4183466 1.0032521 836  
## beta.seed 0.2726964 0.02675609 0.5213452 1.0000811 17646  
## sigma[1] 1.1078321 0.82724527 1.5140057 1.0019767 1066  
## sigma[2] 1.6255883 0.66724891 3.4928227 1.0008618 8271  
## sigma[3] 2.0729890 1.07572666 4.0880917 1.0009368 17646  
## sigma[4] 0.5692032 0.44587322 0.7266135 1.0000804 20501  
## sigma[5] 42.1684821 35.07943985 51.1404531 1.0000033 45000  
## X[1,1] 3.2577292 3.01980198 3.4948289 0.9999974 45000  
## X[2,1] 2.9426020 2.66634086 3.2230433 1.0000283 45000  
## X[3,1] 3.1724671 2.92418151 3.4168273 0.9999871 45000  
## X[4,1] 3.9140946 3.72900454 4.1014057 0.9999965 45000  
## X[5,1] 3.4236818 3.19245668 3.6532609 1.0000117 45000  
## X[6,1] 4.2191391 4.05609875 4.3812474 0.9999779 45000  
## X[7,1] 3.0873774 2.84496935 3.3316960 1.0000302 45000  
## X[8,1] 3.2728487 3.02760724 3.5159001 1.0000340 45000  
## X[1,2] 3.8952160 3.70393851 4.0875083 1.0000743 45000  
## X[2,2] 2.8930947 2.60264230 3.1852648 0.9999944 45000  
## X[3,2] 3.2348602 2.10454148 4.3613036 1.0000295 44543  
## X[4,2] 3.4847469 3.26542299 3.7048342 1.0001244 13893  
## X[5,2] 3.3902134 3.15608622 3.6223739 1.0000635 26089  
## X[6,2] 4.1688333 3.99546000 4.3435892 1.0000347 45000  
## X[7,2] 3.5508640 3.34384532 3.7609966 1.0000229 45000  
## X[8,2] 2.9529241 2.67716833 3.2293994 1.0000054 45000  
## X[1,3] 2.6033994 2.29475982 2.9111601 1.0001289 18887  
## X[2,3] 2.1763404 1.76583391 2.5806818 1.0000589 40513  
## X[3,3] 3.3793656 3.14850371 3.6122228 1.0000203 45000  
## X[4,3] 3.6253916 3.42123029 3.8306264 1.0000744 19764  
## X[5,3] 3.7013023 3.49500060 3.9075270 0.9999770 45000  
## X[6,3] 3.7054651 3.50005084 3.9107739 1.0000206 37178

Results

What is this for??

Wrangling results

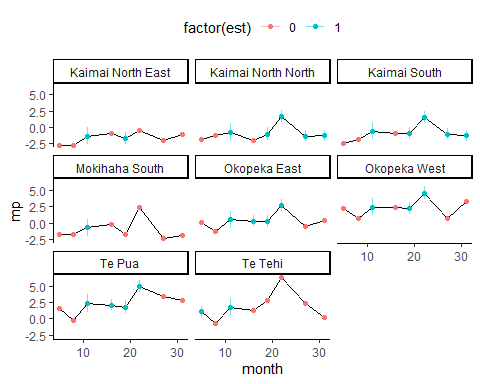
Extract model fit

## Joining, by = c("grid", "month")

## Warning: Column `grid` joining factor and character vector, coercing into  
## character vector

## Joining, by = c("grid", "month")

Figures

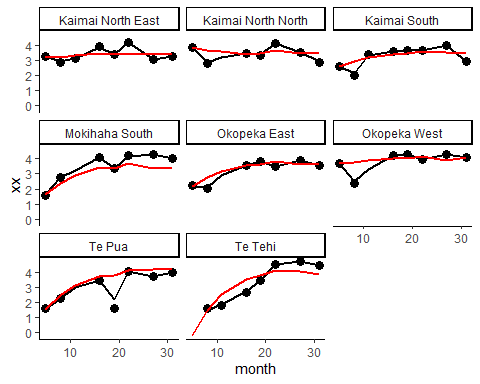


Overall fit

Notes

* fitted process model is the red line circles are the grid data
* black line is the estimate of the true population having accounted for observation error (e.g. variation between grids at the same site)

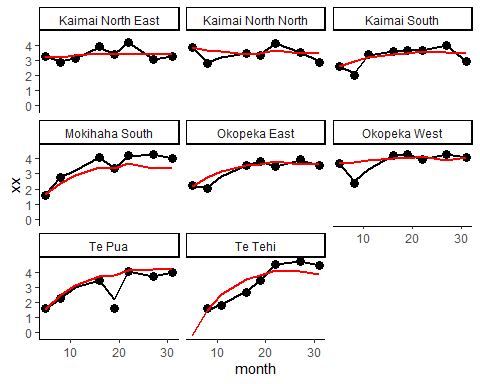
## Warning: Removed 6 rows containing missing values (geom\_point).



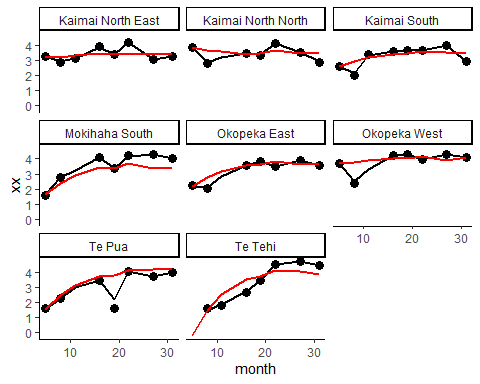
My notes

Other models

# run the model in JAGS with three chains  
mod1.jags <- jags(model = "model1.txt",  
 data = list(Y = Y, Ytau = Ytau, seed = nseed, nGrids = nGrids, nYears = nYears, lag = lag.sp, td = td),  
 param = c("r", "d", "beta.seed", "sigma", "X", "simX", "seed"),  
 n.chains = 3,  
 n.iter = 200000,  
 n.burnin = 5000,  
 parallel = TRUE)



# run the model in JAGS with three chains  
mod2.jags <- jags(model = "model1.txt",  
 data = list(Y = Y, Ytau = Ytau, seed = nseed, nGrids = nGrids, nYears = nYears, lag = lag.sp, td = td),  
 param = c("r", "d", "beta.seed", "sigma", "X", "simX", "seed"),  
 n.chains = 3,  
 n.iter = 100000,  
 n.burnin = 50000,  
 parallel = TRUE)



v3

# run the model in JAGS with three chains  
mod3.jags <- jags(model = "model1.txt",  
 data = list(Y = Y, Ytau = Ytau, seed = nseed, nGrids = nGrids, nYears = nYears, lag = lag.sp, td = td),  
 param = c("r", "d", "beta.seed", "sigma", "X", "simX", "seed"),  
 n.chains = 3,  
 n.iter = 2000,  
 n.burnin = 500,  
 parallel = TRUE)

