Time varying autoregressive state space modeling of multi-site community dynamics

Eric Ward, Mark Scheuerell, Steve Katz

December 3, 2015

Overview

This markdown file is intended to accompany the Scheuerell et al. paper, describing new methods for allowing time varying interactions. The general framework of these methods follows from the MAR(1) framework, as well as the state space versions (MARSS, VAR).

Loading in the data

Read in the raw data on the guild abundances. These are in log-space, and the few 0s in the dataset have been replaced with NAs, because species are likely present in low density (just not detected).

```
library(R2jags)
dat <- read.csv("biannual_var_flat_by_obs_zero.csv", header=TRUE, sep=",")
# Get rid of infinite values
for(i in 1:dim(dat)[2]) {
    dat[which(is.finite(dat[,i])==FALSE),i] = NA
}
# De-mean each of the individual time series
for(i in 1:dim(dat)[2]) {
    dat[,i] = dat[,i] - mean(dat[,i],na.rm=T)
}

spec.names = c("Abalone","Cleaner.fish","Giant.kelp","Herb.fish","Large.invert.eating.fish",
  "Limpets", "Omni.inverts", "Plankt.fish", "Pred.inverts", "Small.invert.eating.fish",
  "Small.pisc.fish", "Snails", "Understory.kelp", "Urchins")
```

Number of sites, time steps, and species/guilds/groups:

```
nSites = 7
n <- dim(dat)[1] # number of time points
m <- dim(dat)[2]/nSites # number of guilds/ groups
```

We have data from 7 sites, but we're only going to model the dynamics of 6 of those sites (the 7th, at the navy pier has lots of missing data). For the remaining 6 sites, we're going to assume that there are 4 underlying 'states', or realizations of the San Nicolas community. In the West region, we're going to assume sites 2 and 3 are samples from the same state. In the North region, we're going to assume site 1 represents a single realization of that state. In the South region, we're going to model 2 states, grouping sites 4 and 5, and modeling site 6 separately (because of spatial location and differences in habitat).

```
Y1 = t(dat[,matrix(seq(1,m*m),m,m)[,4]]) # site 1, state 2
Y2 = t(dat[,matrix(seq(1,m*m),m,m)[,1]]) # site 2, state 1
Y3 = t(dat[,matrix(seq(1,m*m),m,m)[,2]]) # site 3, state 1
Y4 = t(dat[,matrix(seq(1,m*m),m,m)[,5]]) # site 4, state 3
```

```
Y5 = t(dat[,matrix(seq(1,m*m),m,m)[,6]]) # site 5, state 3
Y6 = t(dat[,matrix(seq(1,m*m),m,m)[,7]]) # site 6, state 4
```

Now we'll read in the covariates. For this application, there are 3 covariates (ENSO, urchin harvest, otter counts), and 2 of the 3 have the same values for all regions becasue they are recorded at a coarser spatial scale than the sampling sites (ENSO, urchin harvest).

```
covars = read.csv("covariates_by_region.csv")

enso.wns = matrix(0,dim(covars)[1],m)
otters.w = matrix(0,dim(covars)[1],m) # dd.1
otters.n = matrix(0,dim(covars)[1],m)
otters.s = matrix(0,dim(covars)[1],m)
for(i in 1:m) {
    enso.wns[,i] = covars[,"enso_W"]
    otters.w[,i] = covars[,"otter_W"]
    otters.n[,i] = covars[,"otter_N"]
    otters.s[,i] = covars[,"otter_S"]
}

# For harvest, we're only including removals of urchins, hence
# 0s for other species
harvest.wns = matrix(0,dim(covars)[1],m)
harvest.wns[,14] = covars[,"harvest_W"]
```

Data preparation

We need to vectorize the B matrix in our code, so we need to do some housekeeping to create some indices.

```
m2 = m*m
rowIndices = rep(seq(1,m), m)
colIndices = sort(rowIndices)
Bindices = matrix(seq(1,m2),m,m)
Bprior = diag(m2)
BR = m2
Bz = rep(0,m2)
Bdiag = seq(1,m*m,by=(m+1)) # these are the indices of vecB for diagonal
Boffdiag = seq(1,m*m)[-Bdiag]
```

Writing the JAGS model

```
model = cat("

model {

    # B1 is interaction matrix
    B1.tau ~ dwish(Bprior, m2); # B1.tau is precision matrix of interactions
    B1.1[1:m2,1] ~ dmnorm(Bz,B1.tau);   # interactions at time 1

    # convert the B1 vec to a matrix for initial time step
```

```
  for(cols in 1:m) {
   # go from vec space -> matrix, but i think it's only way in jags
   Bmat.1[1:m,cols,1] <- B1.1[Bindices[1,cols]:Bindices[m,cols],1]; # this is by column
  }

  # initial X0, or state of nature, varies by state, but shared prior
  X0.tau ~ dwish(Bprior[1:m,1:m], m); # prior precision matrix
    X.1[1:m,1] ~ dmnorm(Bz[1:m],X0.tau); # state 1
    X.2[1:m,1] ~ dmnorm(Bz[1:m],X0.tau); # state 2
    X.3[1:m,1] ~ dmnorm(Bz[1:m],X0.tau); # state 3
    X.4[1:m,1] ~ dmnorm(Bz[1:m],X0.tau); # state 4

    # process variance is independent/unequal by region
  # and independent/unequal across spp
    for(i in 1:m) {
        for(j in 1:3) {
           tauQ[i,j] ~ dgamma(0.01,0.01);
        }
    }

  B1.offDiagTau ~ dgamma(0.001,0.001);
  B1.diagTau ~ dgamma(0.001,0.001);

  # Priors for coefficients
    for(i in 1:m) {
        CC[i,1] ~ dnorm(0,1); # effect of enso, varies by spp
        DD[i,1] ~ dnorm(0,1); # effect of otters, varies by spp
    }
    for(i in 1:13) {
        EE[i,1] <- 0; # effect of urchin harvest on non-urchin
    }
    EE[14,1] ~ dnorm(0,1); # effect of urchin harvest on urchins

  for(time in 2:n) {
    for(cols in 1:m) {
       # go from vec space -> matrix, but i think it's only way in jags
       Bmat.1[1:m,cols,time] <- B1.1[Bindices[1,cols]:Bindices[m,cols],time-1];
    }

    # calculate predicted state vector for states 1-4
    predX.1[1:m,time] <- Bmat.1[1:m,1:m,time-1] %*% (X.1[1:m,time-1]) +
CC[1:m,1] * enso.wns[time-1,] + DD[1:m,1] * otters.w[time-1,] +
EE[1:m,1]*harvest.wns[time-1,];#site 2/3

    predX.2[1:m,time] <- Bmat.1[1:m,1:m,time-1] %*% (X.2[1:m,time-1]) +
CC[1:m,1] * enso.wns[time-1,] + DD[1:m,1] * otters.n[time-1,] +
EE[1:m,1]*harvest.wns[time-1,];#site 1

    predX.3[1:m,time] <- Bmat.1[1:m,1:m,time-1] %*% (X.3[1:m,time-1]) +
CC[1:m,1] * enso.wns[time-1,] + DD[1:m,1] * otters.s[time-1,] +
EE[1:m,1]*harvest.wns[time-1,];#site 4/5

    predX.4[1:m,time] <- Bmat.1[1:m,1:m,time-1] %*% (X.4[1:m,time-1]) +
```

```
CC[1:m,1] * enso.wns[time-1,] + DD[1:m,1] * otters.s[time-1,] +
EE[1:m,1]*harvest.wns[time-1,];#site 6

    # include process variation - normally distributed errors
    # independent across spp and site
    for(spp in 1:m) {
      X.1[spp,time] ~ dnorm(predX.1[spp,time], tauQ[spp,1]);
      X.2[spp,time] ~ dnorm(predX.2[spp,time], tauQ[spp,2]);
      X.3[spp,time] ~ dnorm(predX.3[spp,time], tauQ[spp,3]);
      X.4[spp,time] ~ dnorm(predX.4[spp,time], tauQ[spp,3]);
    }

    # update B0 and B1
      for(i in 1:(m*m-m)) {
        # off diagonal elements of B1 -- interactions
        B1.1[Boffdiag[i],time] ~ dnorm(B1.1[Boffdiag[i],time-1], B1.offDiagTau);
      }
      for(i in 1:m) {
        # Diagonal elements of B1 -- density dep
        B1.1[Bdiag[i],time] ~ dnorm(B1.1[Bdiag[i],time-1], B1.diagTau);
      }
  } # end time loop

    # observation / data model, assume R = diag and equal
    for(i in 1:m) {
    tauR[i] ~ dgamma(0.001,0.001);
    }

    for(time in 1:n) {
        for(spp in 1:m) {
            Y2[spp,time] ~ dnorm(X.1[spp,time],tauR[spp]);
            Y3[spp,time] ~ dnorm(X.1[spp,time],tauR[spp]);
            Y1[spp,time] ~ dnorm(X.2[spp,time],tauR[spp]);
            Y4[spp,time] ~ dnorm(X.3[spp,time],tauR[spp]);
            Y5[spp,time] ~ dnorm(X.3[spp,time],tauR[spp]);
            Y6[spp,time] ~ dnorm(X.4[spp,time],tauR[spp]);
        }
    }

}

", file = "TVVARSS.txt")
```

Running the model

```
jags.data = list("Y1", "Y2", "Y3", "Y4", "Y5", "Y6", "m", "m2", "n", "Bindices", "Bprior",
"Bz", "Boffdiag", "Bdiag", "enso.wns", "otters.w", "otters.s", "otters.n", "harvest.wns")
jags.params=c("B1.1", "tauQ", "tauR", "B1.offDiagTau", "B1.diagTau", "CC", "DD", "EE",
"X.1", "X.2", "X.3", "X.4")
model.loc = paste("TVARSS.txt",sep="")
```

```
library(R2jags)

jags.model = jags.parallel(jags.data, inits = NULL, parameters.to.save= jags.params,
model.file=model.loc, n.chains = 4, n.burnin = 100000,
n.thin = 50, n.iter = 120000, DIC = TRUE)
```

Diagnostics

Observed versus predicted values

Estimates from state space models can be used to produce two kinds of residuals; process error residuals (differences between predicted and 'true' or latent states of nature), and observation error residuals (differences between latent states of nature and observed data).

We'll start by examining the observation error residuals, which are interpreted more as how well the model matches observed data.

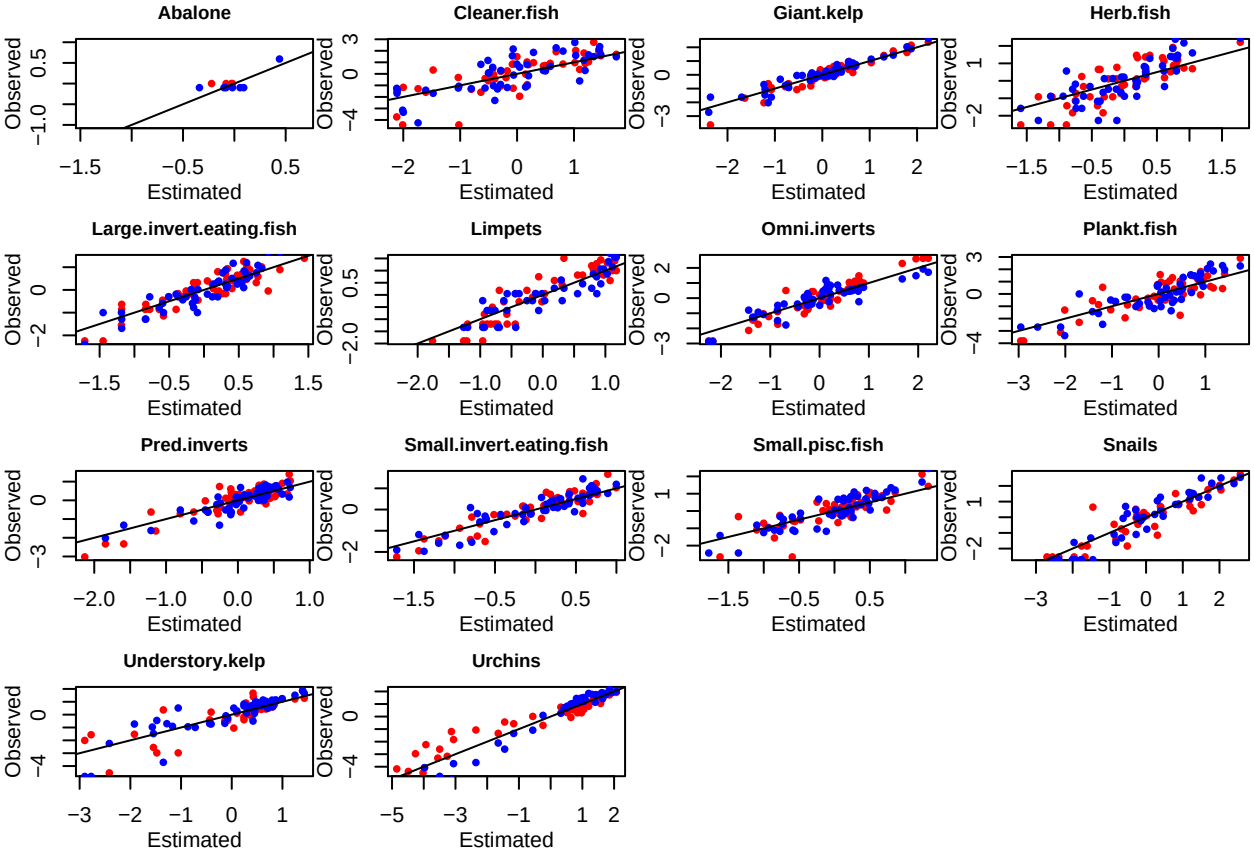

Figure 1: Observed data (y-axis) versus predicted values (x-axis, posterior means) for West region, sites 2 (red) and 3 (blue). The reference line is the 1:1 line.

Time series of observed and predicted values

We'll examine first how the observed data matches up with the estimated states. This can be done for each of the 4 states.

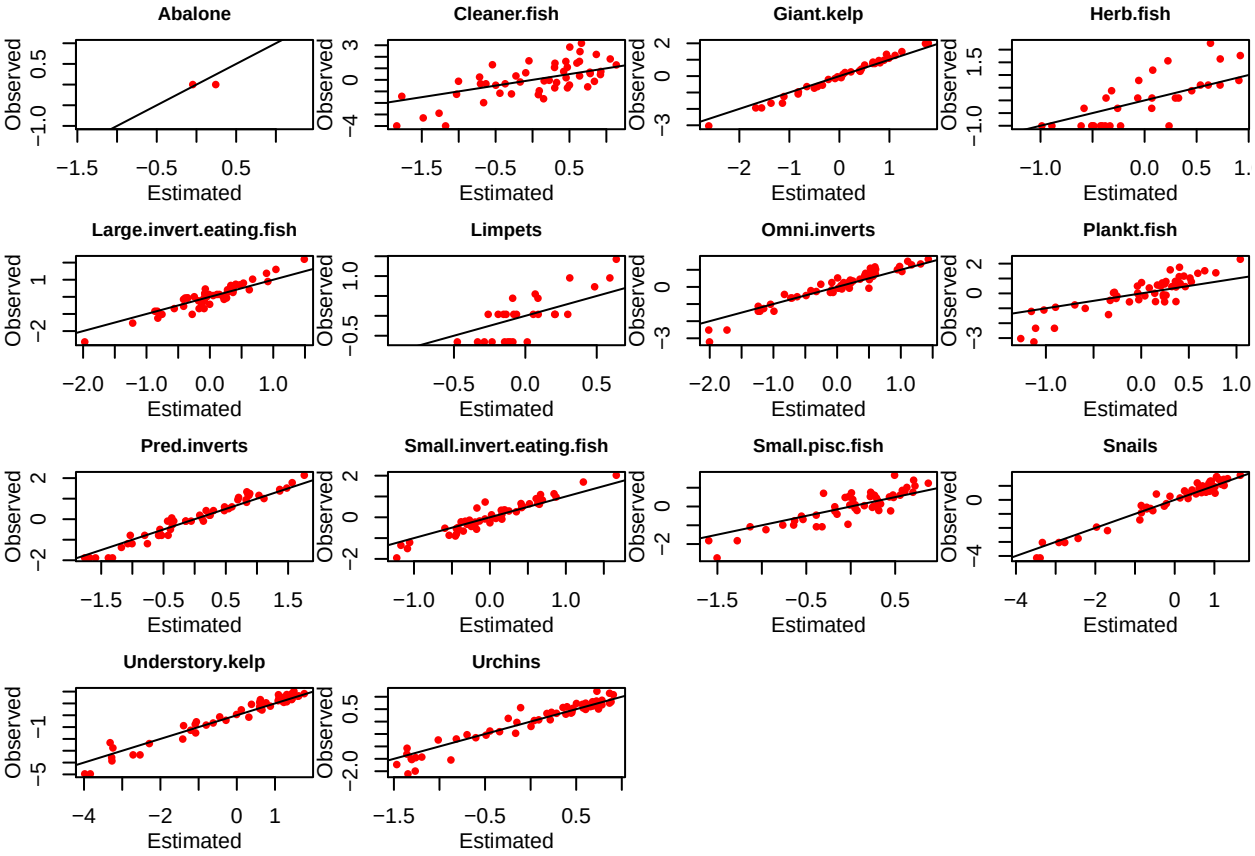

Figure 2: Observed data (y-axis) versus predicted values (x-axis, posterior means) for North region, sites 1 (red). The reference line is the 1:1 line.

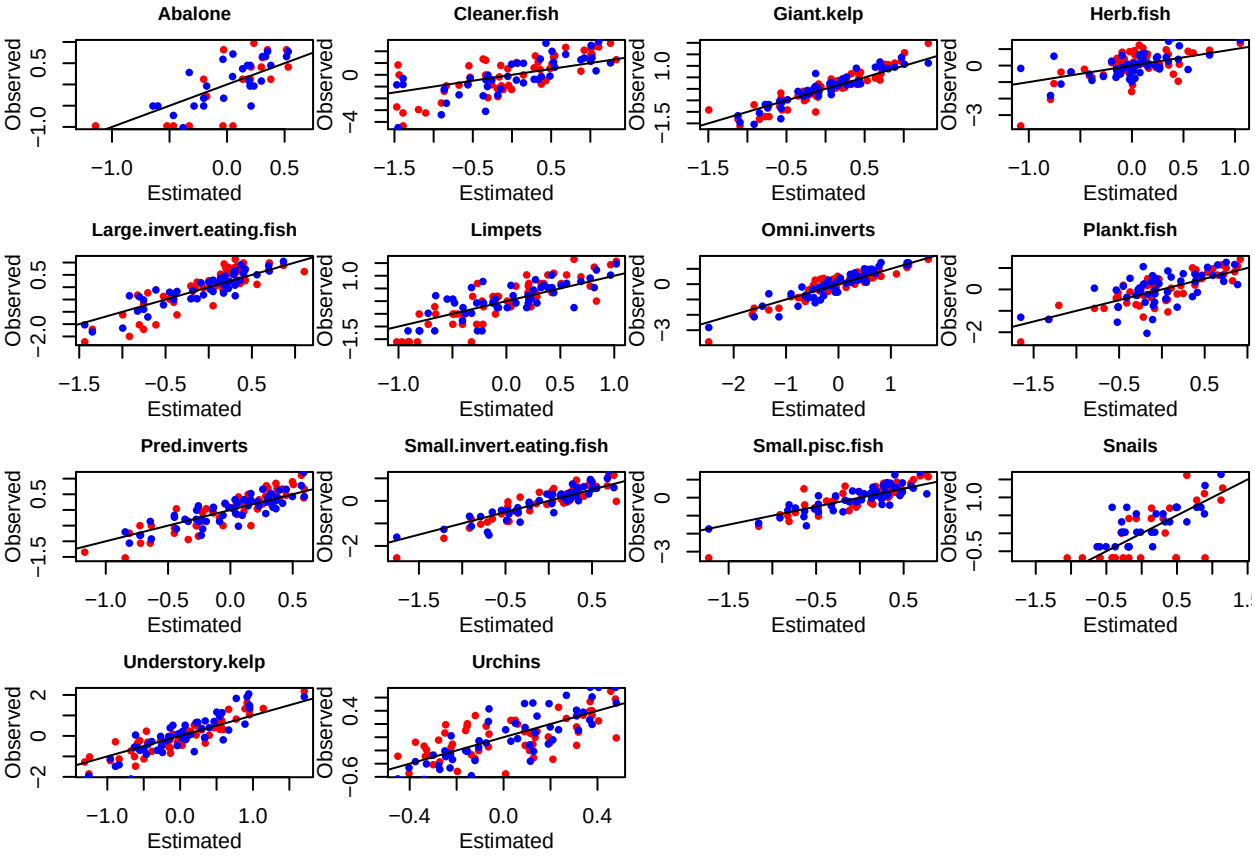

Figure 3: Observed data (y-axis) versus predicted values (x-axis, posterior means) for South region, sites 4 (red) and 5 (blue). The reference line is the 1:1 line.

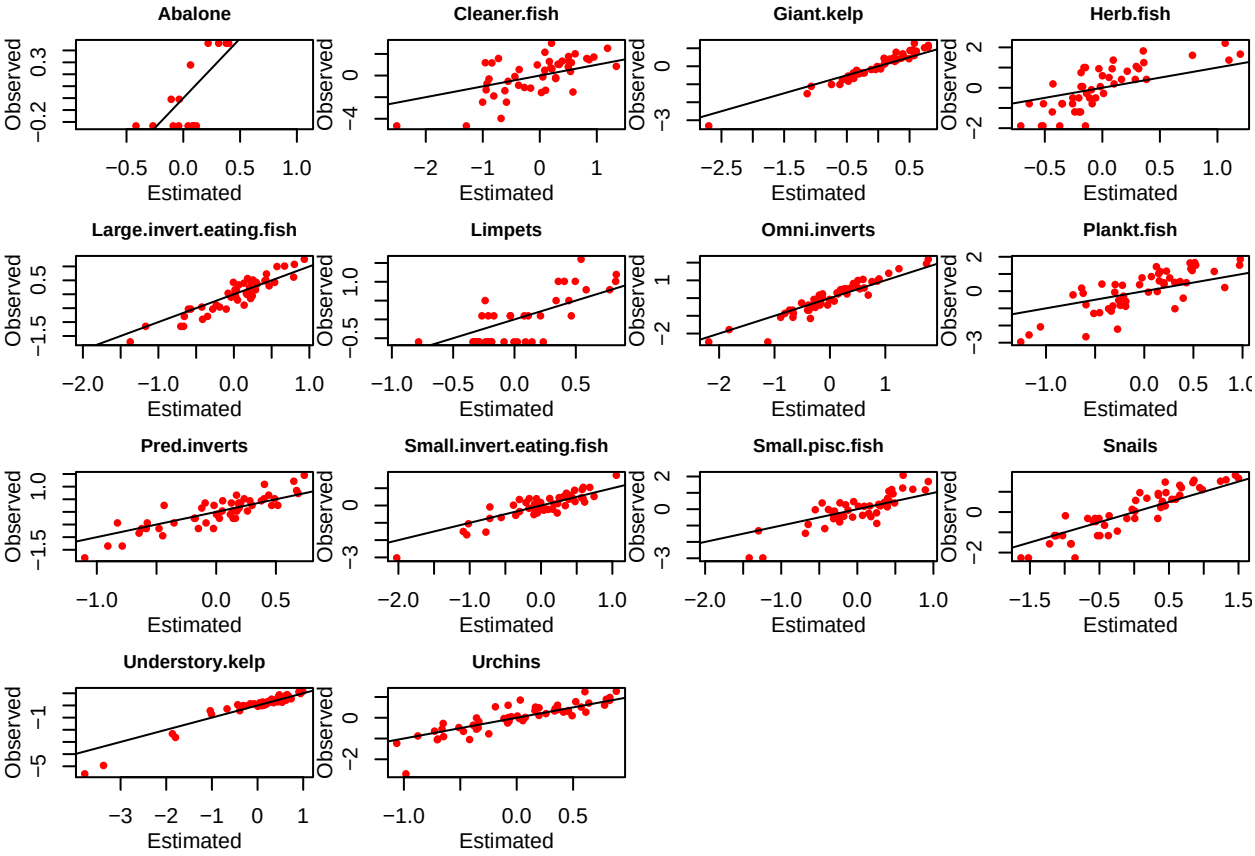

Figure 4: Observed data (y-axis) versus predicted values (x-axis, posterior means) for South region, sites 6 (red). The reference line is the 1:1 line.

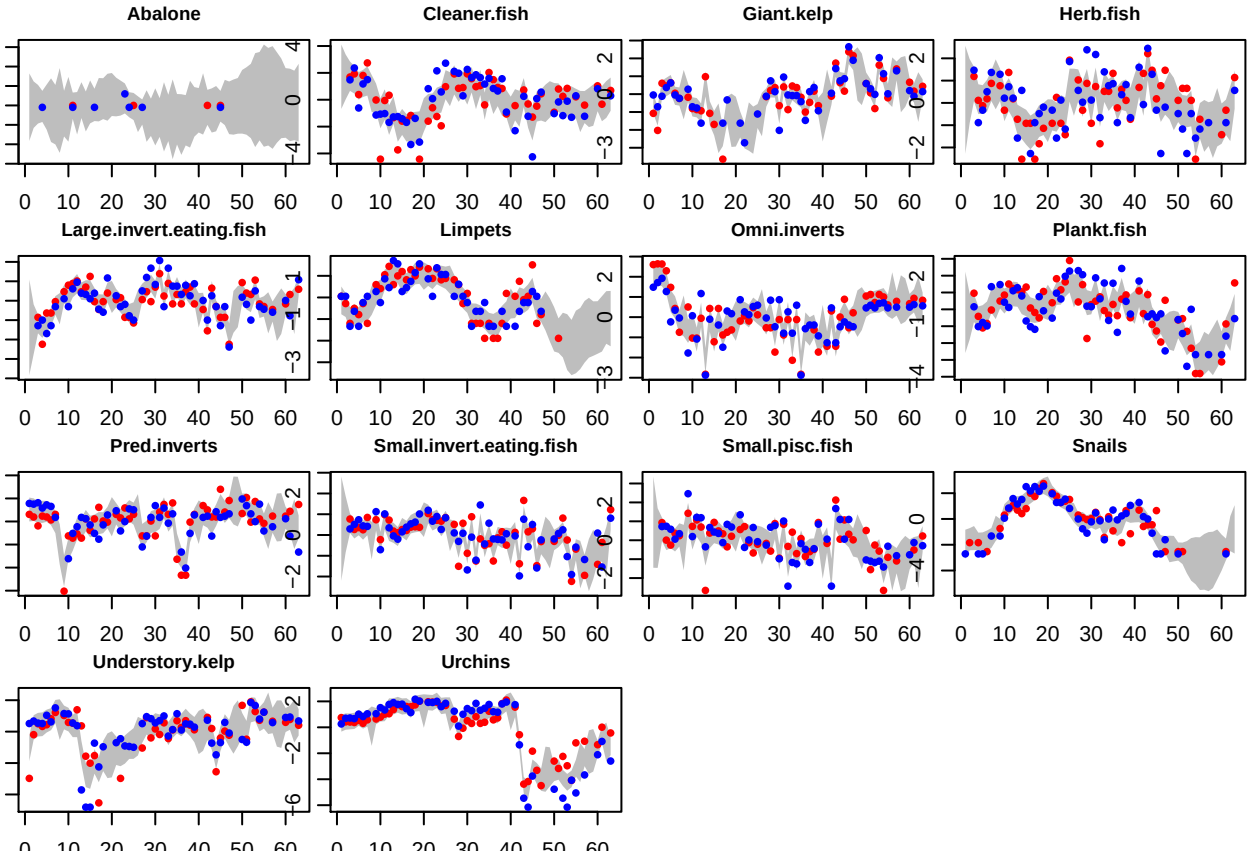

Figure 5: Time series estimated states (95% CIs in grey) and observed data for West region, sites 2 (red) and 3 (blue). The scale of the y-axis is log abundances that have been centered or de-meaned.

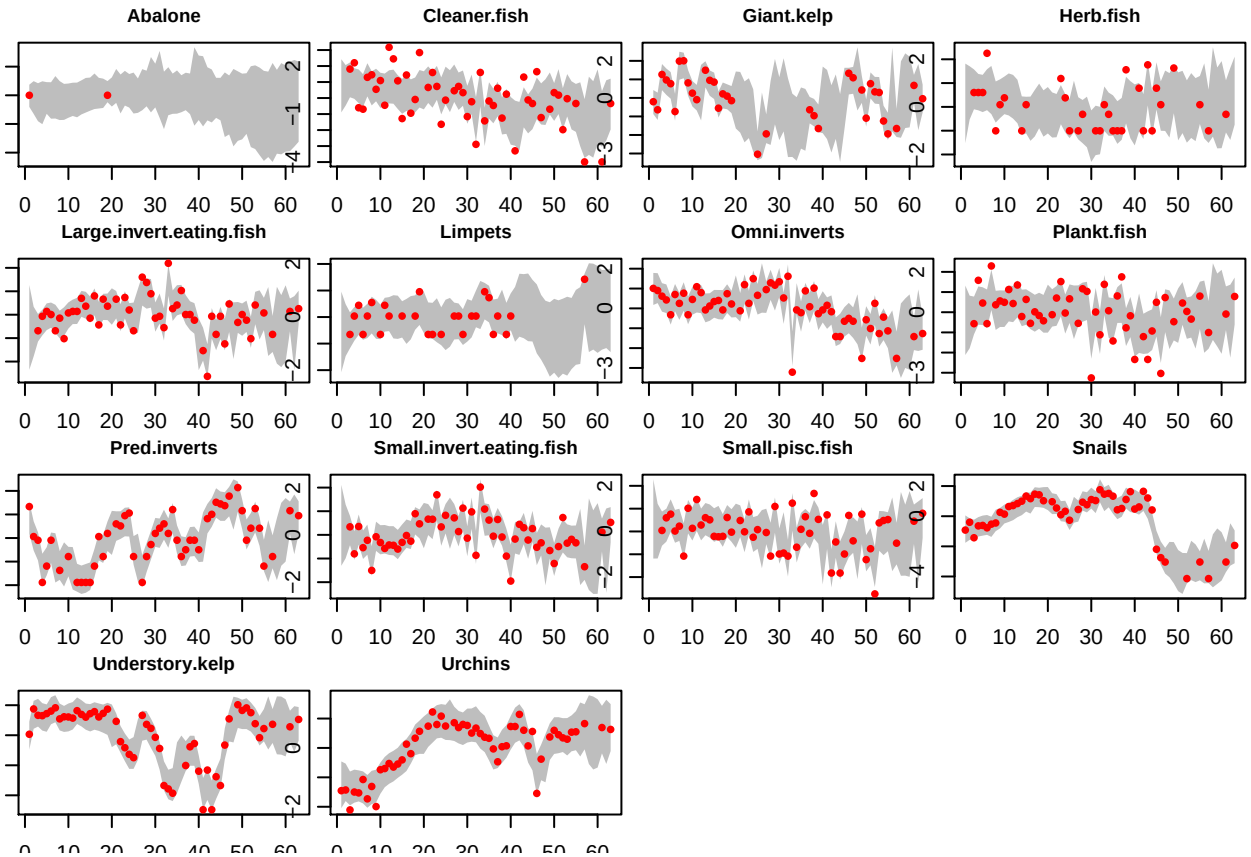

Figure 6: Time series estimated states (95% CIs in grey) and observed data for North region, site 1 (red). The scale of the y-axis is log abundances that have been centered or de-meaned.

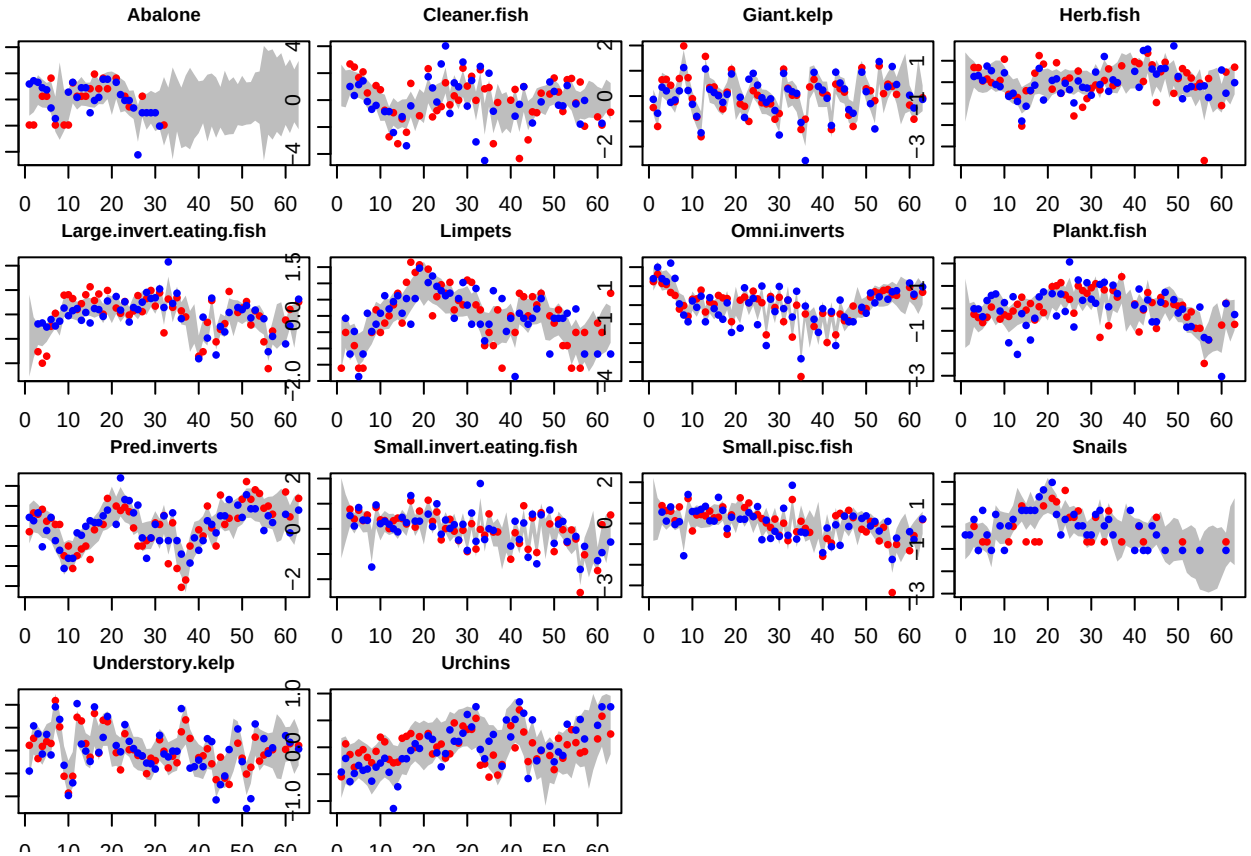

Figure 7: Time series estimated states (95% CIs in grey) and observed data for the first state in the South region, sites 4 (red) and 5 (blue). The scale of the y-axis is log abundances that have been centered or de-meaned.

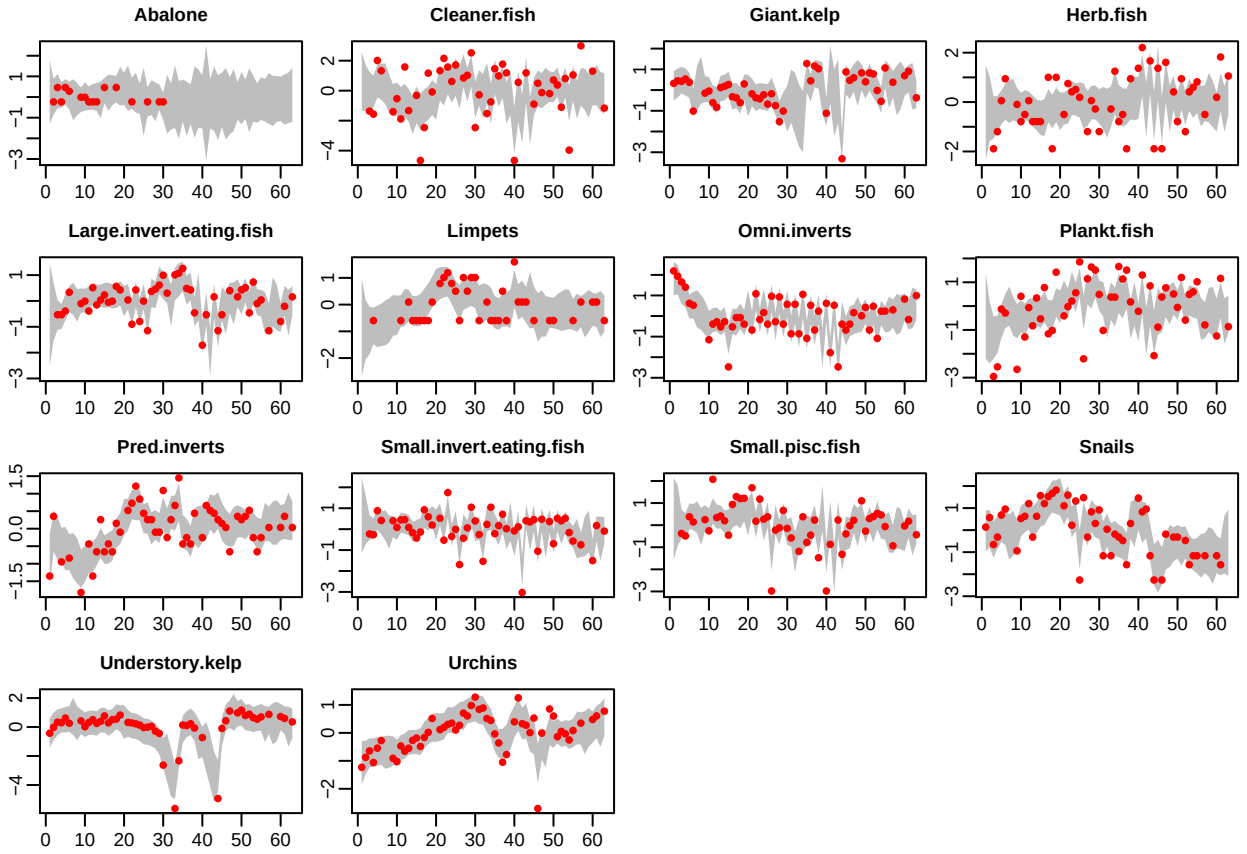

Figure 8: Time series estimated states (95% CIs in grey) and observed data for the second state in the South region, site 6 (red). The scale of the y-axis is log abundances that have been centered or de-meaned.