Primer on modelling species co-occurrence using latent variable models (LVM)

Supplement to 'Letten, Keith, Tozer and Hui (2015) Fine-scale hydrological niche differentiation through the lens of multi-species co-occurrence models'

In this brief primer we demonstrate the efficacy of latent variable models (LVM) for partitioning out the different drivers of species co-occurrence patterns into that which is attributable to shared/diverging environmental responses and that which may be attributable to unmeasured covariates and/or biotic interactions.

To this end, we first simulate multivariate abundance data along two hypothetical environmental gradients (pH and soil moisture), and then fit two LVMs using each of the gradients independently as model predictors. We then derive environmental and 'residual' correlation matrices from the fitted regression and latent variable values respectively to show that the residual correlations induced by the latent variables closely correspond to the correlations due to the environmental response of the missing covariate. This reflects the ability of LVMs to perform inference which is robust to missing covariates/biotic interactions.

Before proceeding, make sure JAGS is installed (http://mcmc-jags.sourceforge.net/), as well as the following packages (note that R2jags is the only critical package for fitting LVMs, with the remainder required for data simulation, data wrangling, aesthetics, and inference). You will also need to load `fitLVM-auxiliaryfunctions.r` which includes a number of functions required for post-processing / analysis.

```
packages <- c("R2jags","coenocliner",
              "ggplot2", "tidyr",
              "vegan", "corrplot",
              "MASS", "gridExtra")
if (length(setdiff(packages, rownames(installed.packages()))) > 0) {
  install.packages(setdiff(packages, rownames(installed.packages())))
}

sapply(packages, require, character.only = TRUE)
source("fitLVM-auxiliaryfunctions.r")
```

Data simulation

Simulate data using the `coenocliner` (http://cran.r-project.org/web/packages/coenocliner/vignettes/coenocliner.pdf) package, which, like it says on the tin, "can be used to generated random count or occurrence data from parametrised species response curve".

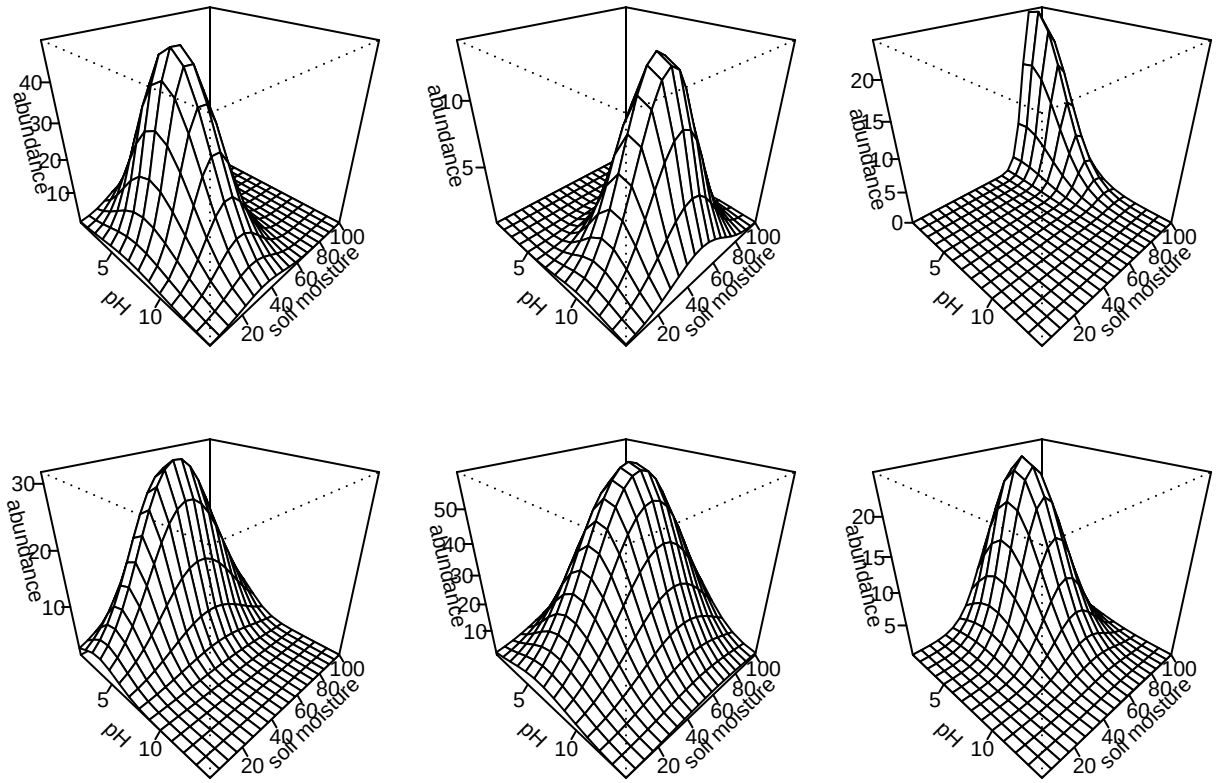
The very first step is to define the simulation parameters.

```
sp <- 6 # number of species

# Gradient 1: PH
grad1.locs <- seq(1, 14, length = 14) # pH gradient locations
grad1.opt <- runif(sp, min = min(grad1.locs), max = max(grad1.locs)) # species optima along gradient
grad1.tol <- ceiling(runif(sp, min = 0.5, max = 3)) # niche widths
h <- ceiling(rlnorm(sp, meanlog = 3)) # max abundances (only defined once)
grad1.par <- cbind(opt = grad1.opt, tol = grad1.tol, h = h) # combine into a matrix

# Gradient 2: soil moisture (% volume)
grad2.locs <- seq(1, 100, length = 20) # soil moisture gradient locations
```

```
grad2.opt <- runif(sp, min = min(grad2.locs), max = max(grad2.locs)) # species optima
grad2.tol <- ceiling(runif(sp, min = 5, max = 30)) # niche widths
grad2.par <- cbind(opt = grad2.opt, tol = grad2.tol) # combine into a matrix

pars <- list(px = grad1.par, py = grad2.par) # combine paremeters for both gradients
```

We now generate a matrix of simulated species abundances using `coenocline` and visualise their bivariate distribution along the two gradients with `persp`. Note that each simulation run will generate different bivariate relationships.

```
# List parameters
locs <- expand.grid(x = grad1.locs, y = grad2.locs) # put gradient locations together

multi.sim <- coenocline(locs, responseModel = "gaussian",
                   params = pars, extraParams = list(corr = 0),
                   expectation = TRUE)

layout(matrix(1:6, ncol = 3))
op <- par(mar = rep(1, 4))
for (i in c(1:6)) {
  persp(grad1.locs, grad2.locs, matrix(multi.sim[, i], ncol = length(grad2.locs)),
        ticktype = "detailed", zlab = "abundance", xlab = "pH", ylab = "soil moisture",
        theta = 45, phi = 30)
}
```



Simulate count (species abundance) data with negative-binomial errors.

```
multi.sim.count <- coenocline(locs, responseModel = "gaussian",
                   params = pars, extraParams = list(corr = 0),
                   expectation = FALSE, countModel = "negbin",
                   countParams = list(alpha = 1))
```

Model fitting

Bundle data into a JAGS friendly format.

```
grad1.X = model.matrix(~ poly(locs[,1],2) -1)

sim.data1 <- list(y    = as.matrix(multi.sim.count),
                  X    = grad1.X,
                  n.cov = ncol(grad1.X),
                  n.sites = nrow(multi.sim.count),
                  n.species = sp)

grad2.X = model.matrix(~ poly(locs[,2],2) -1)

sim.data2 <- list(y    = as.matrix(multi.sim.count),
                  X    = grad2.X,
                  n.cov = ncol(grad2.X),
                  n.sites = nrow(multi.sim.count),
                  n.species = sp)
```

Write JAGS model.

```
cat(
  '
model {

  ## Observation level ##
  for(i in 1:n.sites) { for(j in 1:n.species) {
    eta[i,j] <- spp.int[j] + inprod(X[i, ],spp.coef[j,]) +
      inprod(LV[i, ],lv.coef[j,])

    ## Parameterization of negbin as Poisson with multiplicative gamma random effect
    ## Var = mu + phi*mu^2
    u[i,j] ~ dgamma(1/spp.phi[j],1/spp.phi[j])
    y[i,j] ~ dpois(exp(eta[i,j])*u[i,j])
  }
  }


  ## Latent variables ##
  for(i in 1:n.sites) { LV[i,1] ~ dnorm(0,1); LV[i,2] ~ dnorm(0,1) }
  lv.coef[1,2] <- 0 ## Constraints
  lv.coef[1,1] ~ dunif(0,30) ## Constraints
  lv.coef[2,2] ~ dunif(0,30) ## Constraints
  lv.coef[2,1] ~ dnorm(0,0.1)
  for(j in 3:n.species) { lv.coef[j,1] ~ dnorm(0,0.01); lv.coef[j,2] ~ dnorm(0,0.01) }


  ## Process level ##
  for(j in 1:n.species) {
    ## Draw intercepts and overdispersion parameter independently
    spp.int[j] ~ dnorm(0,0.01); spp.phi[j] ~ dunif(0,30)
    ## Draw spp slopes from common distribution
    for(l in 1:n.cov) {spp.coef[j,l] ~ dnorm(mu.beta[l], tau.beta[l]) }
  }

  ## Prior level ##
  for(l in 1:n.cov) {mu.beta[l] <- 0; tau.beta[l] <- 0.01 }
  ## Species slopes drawn from independent distributions. To draw from
  ## a common distribution (see Ovaskainen & Soininen 2011) change
  ## to vague priors for hyperparameters for spp slopes e.g.
  ## for(l in 1:n.cov) { mu.beta[l] ~ dnorm(0, 0.01); tau.beta[l] ~ dunif(0,50) }
}
  '
  , file="LVM-simulate.txt")
```

We now run an individual LVM for each predictor (pH and soil moisture) by calling JAGS. Note this may take a fair bit of time (but hopefully under an hour!).

```
fit.LVM.1 <- jags(data=sim.data1,
                  inits = NULL,
                  parameters.to.save=c('spp.int', 'spp.coef', 'spp.phi', 'LV',
                                       'lv.coef', 'mu.beta', 'tau.beta'),
                  model.file="LVM-simulate.txt",
                  DIC=TRUE,
                  n.iter=10000,
                  n.burnin=1000,
                  n.chains=3,
                  n.thin=20)

fit.LVM.2 <- jags(data=sim.data2,
                  inits = NULL,
                  parameters.to.save=c('spp.int', 'spp.coef', 'spp.phi', 'LV',
                                       'lv.coef', 'mu.beta', 'tau.beta'),
                  model.file="LVM-simulate.txt",
                  DIC=TRUE,
                  n.iter=10000,
                  n.burnin=1000,
                  n.chains=3,
                  n.thin=20)

# Save data to model fits
fit.LVM.1$data <- sim.data1
fit.LVM.2$data <- sim.data2
```

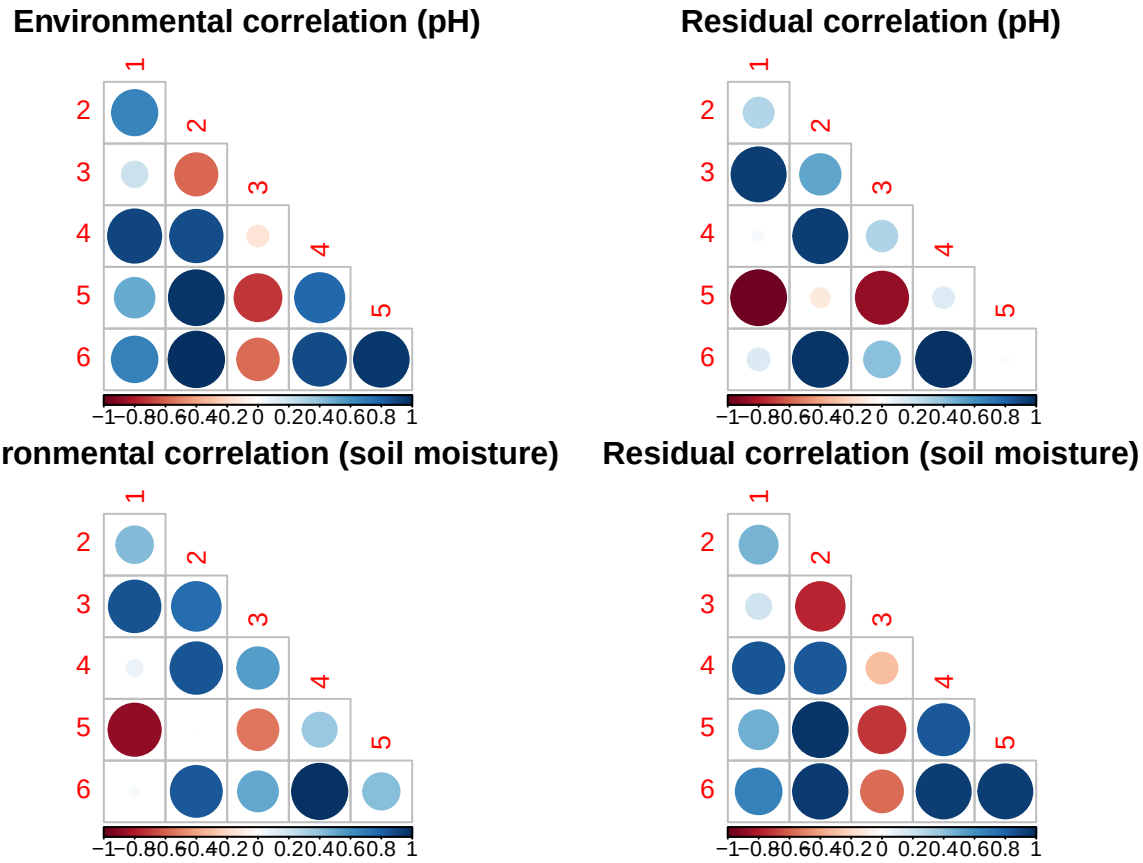
Model inference

Extract environmental and residual correlations (both significant and non-significant mean and median values).

```
enviro.cor.1 <- extract.env.cor(fit.mod = fit.LVM.1,
                                X = fit.LVM.1$data$X,
                                y = fit.LVM.1$data$y)
residual.cor.1 <- extract.residual.cor(fit.mod = fit.LVM.1,
                                       X = fit.LVM.1$data$X,
                                       y = fit.LVM.1$data$y)
enviro.cor.2 <- extract.env.cor(fit.mod = fit.LVM.2,
                                X = fit.LVM.2$data$X,
                                y = fit.LVM.2$data$y)
residual.cor.2 <- extract.residual.cor(fit.mod = fit.LVM.2,
                                       X = fit.LVM.2$data$X,
                                       y = fit.LVM.2$data$y)
```

Compare environmental and residual correlation plots for the two gradients. The residual correlation from the LVM for pH should provide a good approximation of the environmental correlation attributable to soil moisture, and vice versa.

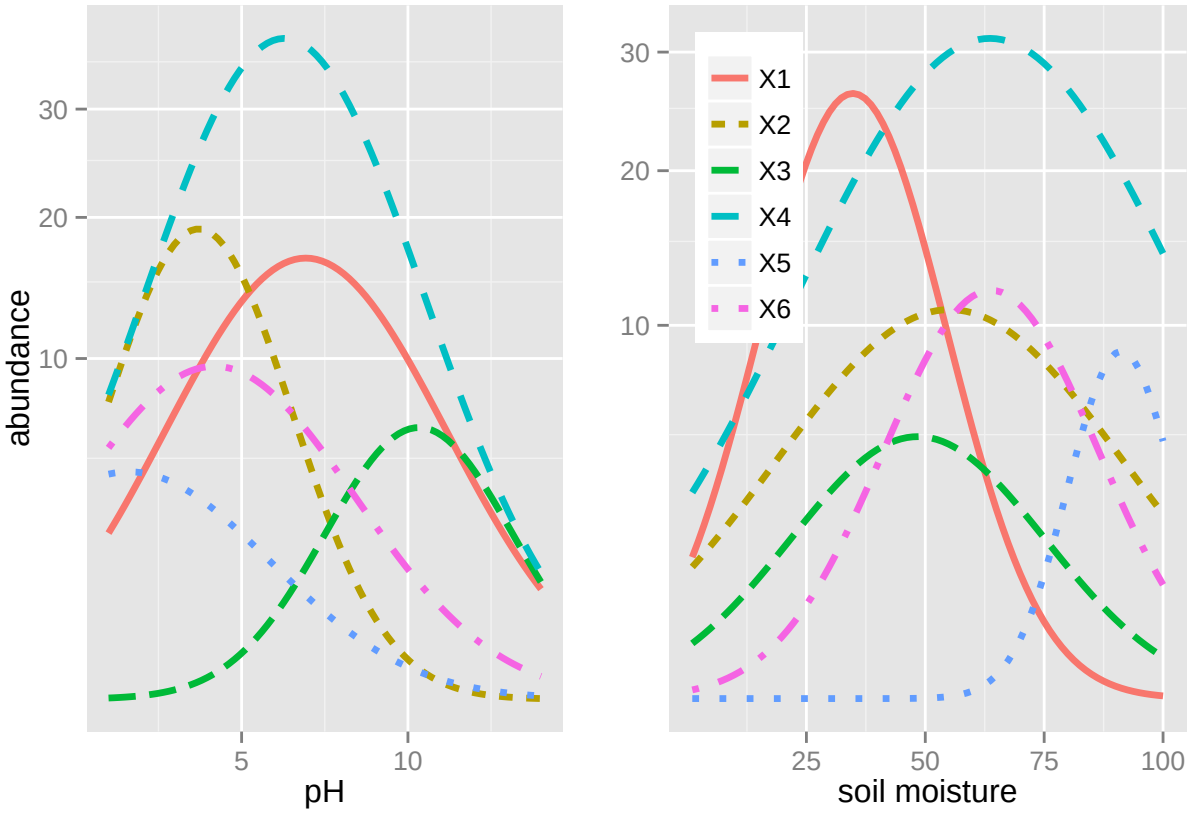
```
par(mfrow = c(2,2))
corrplot(enviro.cor.1$envcor.mean, diag = F, type = "lower",
         title = "Environmental correlation (pH)", mar=c(0,0,1,0))
corrplot(residual.cor.1$rescor.mean, diag = F, type = "lower",
         title = "Residual correlation (pH)", mar=c(0,0,1,0))

corrplot(enviro.cor.2$envcor.mean, diag = F, type = "lower",
         title = "Environmental correlation (soil moisture)", mar=c(0,0,1,0))
corrplot(residual.cor.2$rescor.mean, diag = F, type = "lower",
         title = "Residual correlation (soil moisture)", mar=c(0,0,1,0))
```



Large red circles indicate strong negative patterns of co-occurrence, while large blue circles indicate strong positive patterns of co-occurrence.

It is instructive to contrast the environmental and residual correlation plots with the species-specific responses to each gradient (note for simplicity these are fitted using the simulated data and `ggplot`'s internal model fitting functions, but could also be obtained via the predicted environmental responses derived from the LVMs).

```
sim.frame1 <- data.frame(multi.sim, grad1 = locs[,1])
sim.gg1 <- gather(sim.frame1, species, value, -grad1)

p1 <- ggplot(sim.gg1, aes(y = value, x = grad1)) +
  geom_smooth(method = "glm", formula = y ~ poly(x,2),
              aes(group = species, col = species, linetype = species),
              size = 1.2, family = negative.binomial(theta = 1), se = FALSE) +
  theme(legend.position="none") +
  coord_trans(y = "sqrt") +
  xlab("pH") + ylab("abundance")

sim.frame2 <- data.frame(multi.sim, grad2 = locs[,2])
sim.gg2 <- gather(sim.frame2, species, value, -grad2)

p2 <- ggplot(sim.gg2, aes(y = value, x = grad2)) +
  geom_smooth(method = "glm", formula = y ~ poly(x,2),
              aes(group = species, col = species, linetype = species),
              size = 1.2, family = negative.binomial(theta = 1), se = FALSE) +
  theme(legend.title=element_blank(), legend.justification=c(0,1), legend.position=c(0,1)) +
  coord_trans(y = "sqrt") +
  xlab("soil moisture") + ylab(NULL)

grid.arrange(p1, p2, ncol = 2)
```


abundance
30
20
10
5
10
pH
30
20
10
X1
X2
X3
X4
X5
X6
25
50
75
100
soil moisture

Mantel tests of the association between the environmental correlation for pH and residual correlation for soil moisture and vice versa.

```
mantel(enviro.cor.1$envcor.mean, residual.cor.2$rescor.mean)
```

```
## 'nperm' > set of all permutations; Resetting 'nperm'.

##
## Mantel statistic based on Pearson's product-moment correlation
##
## Call:
## mantel(xdis = enviro.cor.1$envcor.mean, ydis = residual.cor.2$rescor.mean)
##
## Mantel statistic r: 0.9931
##       Significance: 0.0041667
##
## Upper quantiles of permutations (null model):
##   90%   95% 97.5%   99%
## 0.809 0.851 0.928 0.961
## Permutation: free
## Number of permutations: 720
```

```
mantel(enviro.cor.2$envcor.mean, residual.cor.1$rescor.mean)
```

```
## 'nperm' > set of all permutations; Resetting 'nperm'.

##
## Mantel statistic based on Pearson's product-moment correlation
##
## Call:
## mantel(xdis = enviro.cor.2$envcor.mean, ydis = residual.cor.1$rescor.mean)
##
## Mantel statistic r: 0.9636
##       Significance: 0.0027778
##
## Upper quantiles of permutations (null model):
##   90%   95% 97.5%   99%
## 0.453 0.567 0.676 0.859
## Permutation: free
## Number of permutations: 720
```

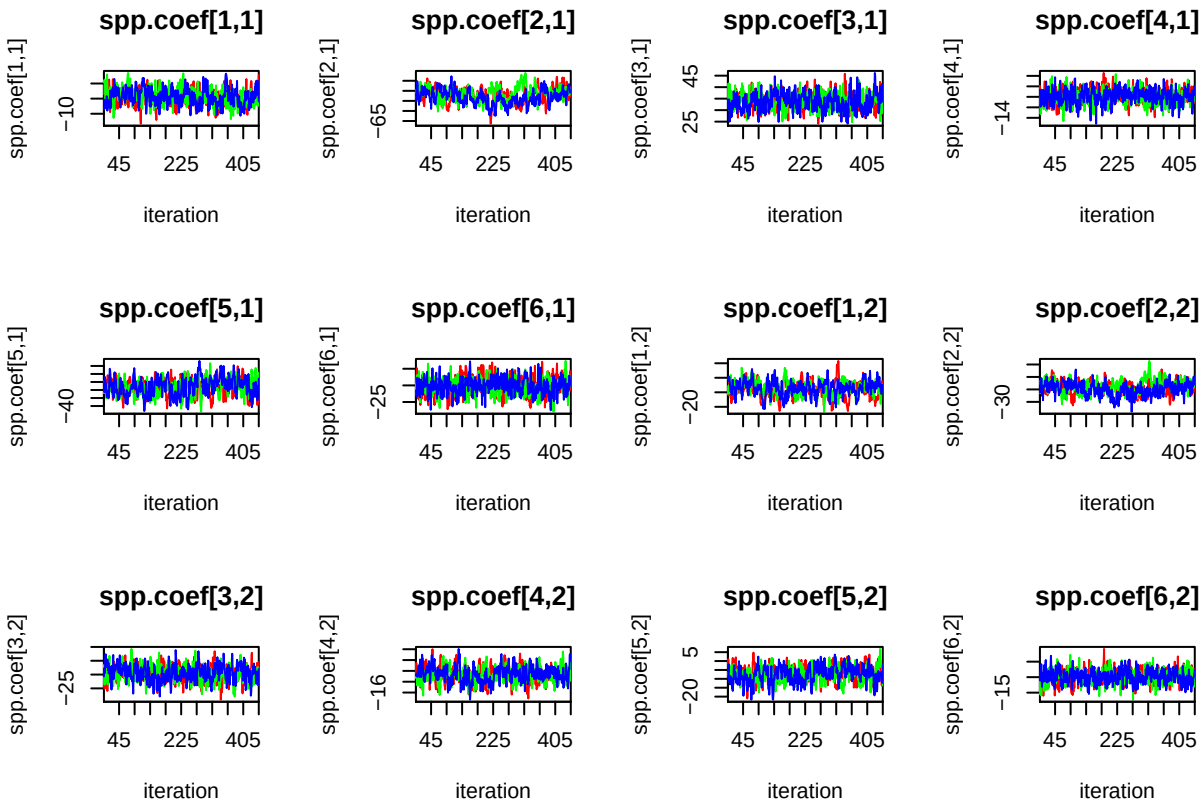
Model diagnostics

Remember to to assess mixing of chains, Rhats etc..

```
# e.g.
out <- fit.LVM.1$BUGSoutput # extract BUGS output

# view species coefficients, credible intervals and Rhats
out$summary[grep("spp.coef", rownames(out$summary)),]
```

```
##                      mean       sd       2.5%        25%        50%
## spp.coef[1,1]  -4.376308 2.658915  -9.623917  -6.211199  -4.368937
## spp.coef[2,1] -52.244569 3.696968 -59.665952 -54.642950 -52.134381
## spp.coef[3,1]  33.828181 3.625238  27.151757  31.306688  33.740443
## spp.coef[4,1]  -9.938732 1.412374 -12.817821 -10.836772  -9.902413
## spp.coef[5,1] -28.113718 5.286011 -38.512741 -31.773215 -27.867749
## spp.coef[6,1] -20.152278 2.512646 -25.208912 -21.722603 -20.155770
## spp.coef[1,2] -13.850570 2.661883 -19.201696 -15.574995 -13.783175
## spp.coef[2,2] -24.309061 2.676103 -29.866672 -26.093269 -24.186097
## spp.coef[3,2] -19.474867 2.839791 -25.224109 -21.336707 -19.436900
## spp.coef[4,2] -12.688023 1.474339 -15.530660 -13.673480 -12.719679
## spp.coef[5,2]  -7.328328 4.444105 -16.405253 -10.183379  -7.234160
## spp.coef[6,2]  -9.799089 2.251483 -14.159866 -11.275689  -9.820069
##                      75%      97.5%     Rhat n.eff
## spp.coef[1,1]  -2.624422   1.022885 1.001172  1400
## spp.coef[2,1] -49.668080 -45.426535 1.056914    40
## spp.coef[3,1]  36.280006  41.139177 1.000929  1400
## spp.coef[4,1]  -9.014537  -7.126312 1.004041   480
## spp.coef[5,1] -24.367833 -18.238094 1.005865   880
## spp.coef[6,1] -18.569281 -15.106468 1.006487   430
## spp.coef[1,2] -12.050343  -8.729608 1.017428   130
## spp.coef[2,2] -22.469606 -19.265690 1.041300    53
## spp.coef[3,2] -17.496958 -14.163351 1.002806   690
## spp.coef[4,2] -11.716850  -9.720445 1.002209   860
## spp.coef[5,2]  -4.249257   1.338408 1.016139   140
## spp.coef[6,2]  -8.249307  -5.495947 1.000578  1400
```

```
# view trace plots
traceplot(fit.LVM.1, varname = c("spp.coef"), mfrow = c(3,4))
```



References

Ovaskainen, O. & Soininen, J. (2011). Making more out of sparse data: hierarchical modeling of species communities. Ecology, 92, 289-295.