Supporting Information 2: Worked examples

1 Introduction

This document contains supplementary material to worked example 1 and worked example 2, from the main manuscript. It contains the workflow and code used to load and explore data, fit and check all models, produce Figures 4 and 5 in the main manuscript, as well as additional figures and model outputs.

2 Setup

The code below acts as a common setup for both worked examples.

```
library(readxl) # read excel files
library(dplyr) # data manipulation
library(tidyr) # data manipulation
library(gllvm) # GLLVM modeling
library(ggplot2) # data visualization
library(ggpubr) # data visualization
library(mefa) # abundance-occupancy plots
library(corrplot) # correlation plots
library(gclus) # order correlation matrix
set.seed(1) # set seed
```

We also include a line of code that allows the `gllvm` package (through the underlying `TMB C++` interface) to fit the model using multiple CPU cores simultaneously, which might improve computation time:

```
TMB::openmp(parallel::detectCores()-1, autopar = TRUE, DLL = "gllvm")
```

3 Worked example 1

3.1 Preparing data

Below is the code for importing and cleaning the species- and environmental data, using the raw data file downloaded from the *Figshare* repository associated with Fernandez et al. (2021) (URL).

```
# load and format species data
species <- read_xlsx("example_1/data/Fernandez et al. 2021_JVS_Data.xlsx",
                     sheet=1) |>
  tibble::column_to_rownames("Species") |> # set species column as rownames
  t() |> as.data.frame() # transpose to site X species format

# load environemntal data
env <- read_xlsx("example_1/data/Fernandez et al. 2021_JVS_Data.xlsx",
                 sheet=2)
colnames(env) <- c("plot","sample", "soil.organic.C", # clean column names
                   "soil.N","soil.C.N", "soil.moisture")
```

We see, however that there are some typos in the values of the species data frame. In the original paper, the highest abundance (Basal area) of *L. lucidum* is visualised as approximately 1.8, while it is considerably higher in the excel sheet:

```
max(species$L.luc)
```

```
[1] 1775992
```

It seems most likely that there is a missing dot after the first number. Indeed, this seems to be the case for quite a few entries:

```
sum(species>1000)
```

```
[1] 36
```

As all the correctly entered numbers have exactly 6 digits after zero, we thus assume that the e.g. the number `10559` is supposed to be `0.010559`, while e.g. `178449` is supposed to be `0.178449`. Therefore, we assume the problem will be fixed by dividing the problem values by `1e6`.

```
species[species>1000] <- species[species>1000] / 1000000
```

In the case of the environmental data, there is one duplicate plot name (`lig.60_C2`). We do not know whether this is due to a typo or two separate measurements at the same location. For the purposes of this paper, we will assume it is the former, and average the measurements of the two columns with the same name.

```
duplicates <- env[env$plot=="lig.60_C2",-c(1,2)]
env[42,-c(1,2)] <- t(as.matrix(colMeans(duplicates))) # replace first duplicate
env <- env[-44,] # remove second duplicate
```

We also see that the row names of the `species` data frame and the `plot` column in the `env` data frame do not always match in terms of case, and in whether periods are used to separate sites. We therefore also need to harmonise this:

```
# make both lowercase
rownames(species) <- stringr::str_to_lower(rownames(species))
env$plot <- stringr::str_to_lower(env$plot)

# remove all periods from names
rownames(species) <- lapply(rownames(species), FUN = gsub,
                            pattern=".", replacement="", fixed=T)
env$plot <- gsub(env$plot, pattern=".", replacement="", fixed=T)
```

We also scale and center the predictor variables

```
env[,-c(1,2)] <- scale(env[,-c(1,2)])
```

In order to fit the second (concurrent ordination) model (see main text), we also need a filtered dataset of only the sites for which we have environmental data.

```
# filter by common sites
species_sub <- species |> filter(rownames(species) %in% env$plot)
env_sub <- env |> filter(env$plot %in% rownames(species_sub))
```

3.2 Data exploration

3.2.1 Species data

We first explore the data by creating a table of the row-and column occurrences:

```
## data exploration ####
# look at row and species scores
table(rowSums(species>0)) # five sites with only 1 species
```

```
 1  2  3  4  5  6  7  8  9 10 11 12
 5 13 20 30 22 19 25 16  8  3  1  2
```

```
table(colSums(species>0)) # all species have at least 3 sites
```

```
  3   5   9  11  13  27  28  36  37  38  42  52  66  67  72 100 109 113
  1   2   1   1   1   1   1   1   1   1   2   1   1   1   1   1   1   1
```

We see that three sites only contain one species. We therefore decide to exclude these sites from the analysis:

```
# remove singleton sites
species_f <- species[rowSums(species>0)>1,]
```

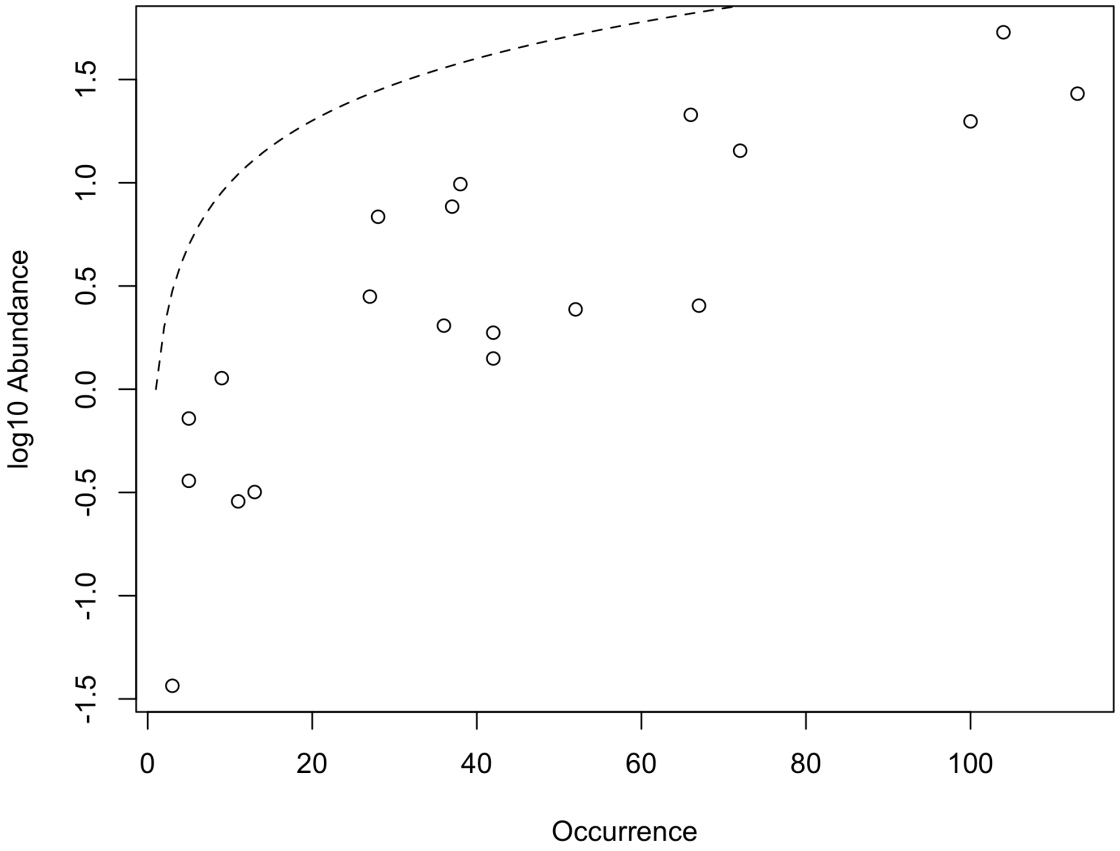
After removing the sites, all species still have at least three occurences:

```
# re-check species rarity
table(colSums(species_f>0)) # still >= three species in each site
```

```
  3   5   9  11  13  27  28  36  37  38  42  52  66  67  72 100 104 113
  1   2   1   1   1   1   1   1   1   1   2   1   1   1   1   1   1   1
```

Looking at the abundance-occupancy, we see that the pattern is approximately log-linear, as we would expect, with only one species being a potential low-abundance outlier at the left part of the spectrum.

```
# Abundance-occupancy
mefa::aoplot(species_f, log=T) # relatively log-linear
```



Supplementary Figure 2. 1: Abundance-occupancy plot for the species biomass data in worked example 1

As with the full dataset, we also need to look at the properties of the reduced dataset:

```
# species occupancy
table(rowSums(species_sub>0))
```

```
 1  2  3  4  5  6  7  8  9 12
 2  4  5  5  2  8  7  3  3  1
```

```
table(colSums(species_sub>0)) # two species with 0 obs!!
```

```
 0  1  2  3  6  7  9 10 11 12 13 14 17 18 23 27 31
 2  1  2  1  1  1  2  1  1  1  1  1  1  1  1  1  1
```

We see that two species have 0 observations, and one species have only 1 observation, in the reduced dataset. We filter these out, in addition to singleton sites, before we fit the concurrent model.

```
species_sub <- species_sub[,colSums(species_sub>0)>1]
env_sub <- env_sub[rowSums(species_sub>0)>1,]
species_sub <- species_sub[rowSums(species_sub>0)>1,]
```

We thus remove the most data-deficient sites and species.

```
table(rowSums(species_sub>0))
```

```
 2  3  4  5  6  7  8  9 12
 4  5  5  2  8  7  4  2  1
```

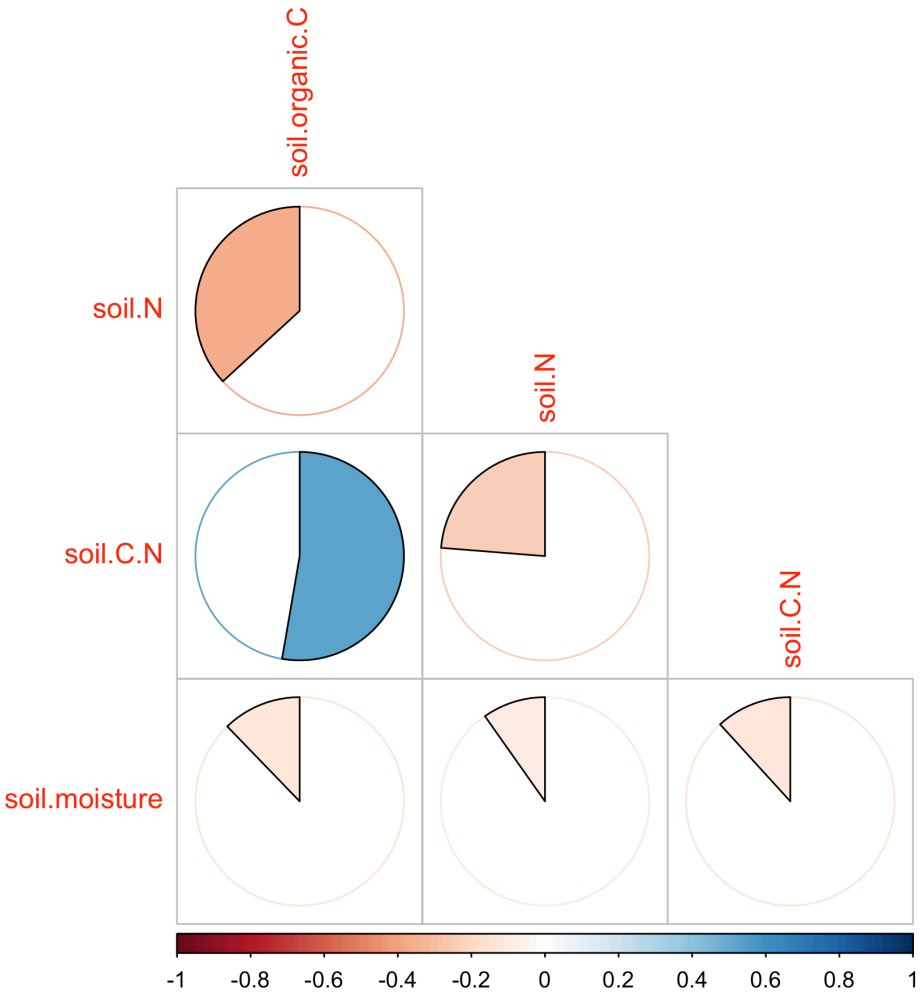
```
table(colSums(species_sub>0))
```

```
 2  3  6  7  9 10 11 12 13 14 17 18 23 27 29
 2  1  1  1  2  1  1  1  1  1  1  1  1  1  1
```

3.2.2 Environmental data

Sup. Figure 2. 2 shows the correlation between the (scaled) environmental predictors. As no predictors have a correlation coefficient higher than 0.56, we remain fairly confident that co-linearity will not have a significant effect on the model fit when including all available environmental predictors.

```
corrplot::corrplot(cor(env_sub[,-c(1,2)]), type = "lower",
                   method = "pie", diag= FALSE)
```



Supplementary Figure 2. 2: Correlation plot for the observed environmental predictors in worked example 1

3.3 Model fitting

Below is the code for the fitting of the unconstrained and concurrent GLLVMs. All models ran without giving errors, in less than five minutes. Code for the model fitting of the unconstrained model with one latent variable can also be found in the main text.

3.3.1 Unconstrained latent variable models

Below is the code to fit the model with unconstrained latent variables. We fit models with up to 6 latent variables, given an AIC minimum at 5 latent variables Sup. Table 2. 1.

```
we_1_unconstrained <- list() # make list for model object
for (n_lv in 1:6) { # run up to 6 LVs
  we_1_unconstrained[[n_lv]] <- gllvm(
  y = species_f,
  family = "tweedie", # response distribution family
  row.eff = "random", # to account for overall abundance in a plot
  num.lv = n_lv, # number of unconstrained LVs
  Power = NULL, # calculate power parameter for Tweedie family
  n.init = 10, # number of starting iterations (to ensure convergence)
  seed = 1, # starting seed (to ensure replicablilty)
  trace=T
  )
}
```

3.3.2 Concurrent latent variable models

Below is the code for fitting the concurrent latent variable models with up to three latent variables, revealing an AIC(c) minimum at two latent variables Sup. Table 2. 2. For the model with three latent variables, we had to play around with a few different seeds in order for the model to be able to calculate starting values for all five iterations.

```
we_1_concurrent <- list() # make list for model object
for (n_lv in 1:3) { # run up to 3 LVs
  we_1_concurrent[[n_lv]] <- gllvm(
    y = species_sub,
    X = env_sub,
    family = "tweedie",
    row.eff = "random",
    method = "LA", # Laplace approx. for better convergence
    lv.formula = ~ soil.organic.C + soil.N + # predictors for the ordination
      soil.C.N + soil.moisture,
    num.lv.c = n_lv, # number of concurrent LVs
    trace = F,
    seed = 123,
    n.init = 5,
    Power = NULL
```

```
  )
}
```

3.4 Model checking

3.4.1 Unconstrained latent variable model

Sup. Table 2. 1 shows the Aikake Information Criteria (AIC) for the resulting unconstrained latent variable models, while Sup. Figure 2. 3 and Sup. Figure 2. 4 show the diagnostic plots, obtained using the `plot()` function on the `gllvm` model object. We see tthat there is an AIC minimum for the model with 5 latent variables.

```
# only look at difference from lowest
ic_mods <- data.frame(
  "df" = unlist(lapply(we_1_unconstrained,
                       FUN = function(x){attributes(logLik(x))$df})), # DF
  "AIC" = unlist(lapply(we_1_unconstrained,
                        FUN = AIC)) # AIC
)
ic_mods$Difference <- ic_mods$AIC - min(ic_mods$AIC) # difference to min
rownames(ic_mods) <- paste(as.character(1:length(ic_mods[, 1])), # add row names
                           "latent variables")

knitr::kable(ic_mods |> round(1)) # render
```

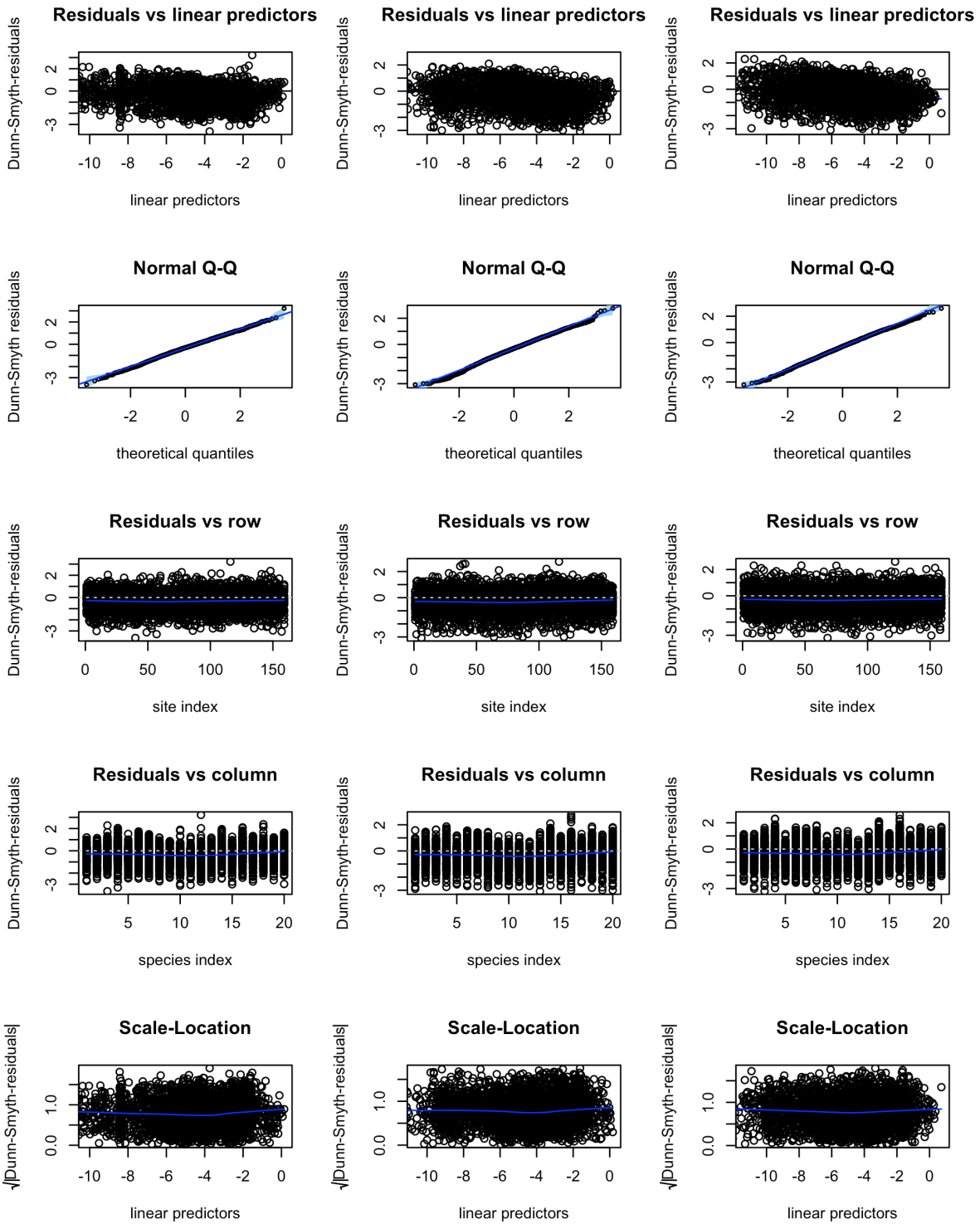
Supplementary Table 2. 1: Information criteria for the unconstrained models in worked example 1, with different number of latent variables

	df	AIC	Difference
1 latent variables	61	1225.7	134.1
2 latent variables	80	1170.0	78.5
3 latent variables	98	1128.6	37.1
4 latent variables	115	1102.5	10.9
5 latent variables	131	1091.6	0.0
6 latent variables	146	1110.2	18.7

As seen in the top row in both Sup. Figure 2. 3 and Sup. Figure 2. 4, there is a slight negative over-representation of negative residuals in all models for high values of the linear predictor. However, we assume it is not large enough to significantly impact the predictions of the model.

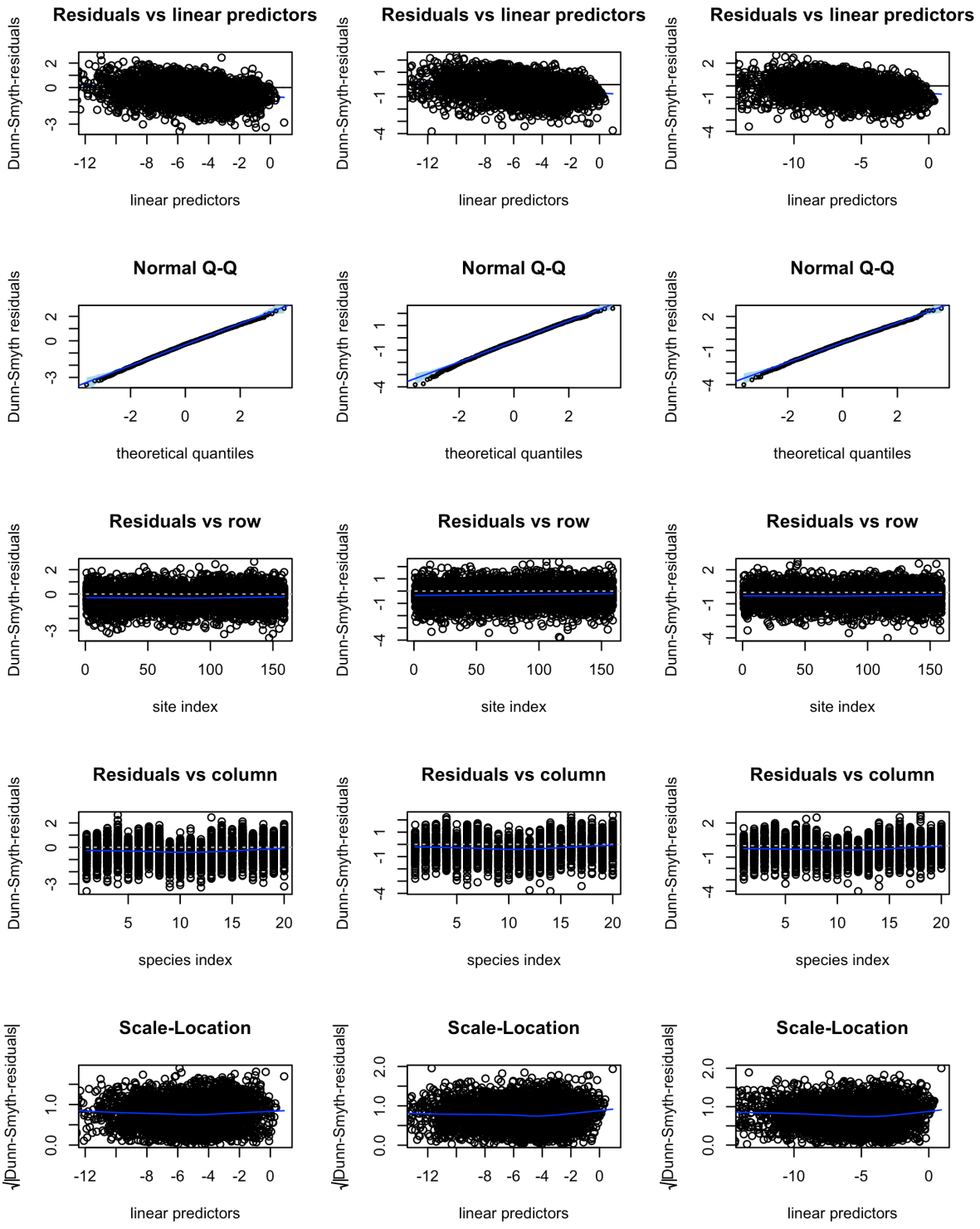
Otherwise, all the other residual diagnostics plots seem to indicate that the model assumptions hold for our data, for all models.

```
# make the points smaller and also no colors
layout(mat = matrix(1:15, nrow = 5, ncol = 3))
for (mod in we_1_unconstrained[1:3]) { # first 3 models
    plot(mod, var.colors=1)
}
```



Supplementary Figure 2. 3: Diagnostic plots of normal randomized quantile residuals for the unconstrained models in worked example 1. Colors represent different species in the dataset. Left column = 1 latent variable, middle = 2 and right = 3.

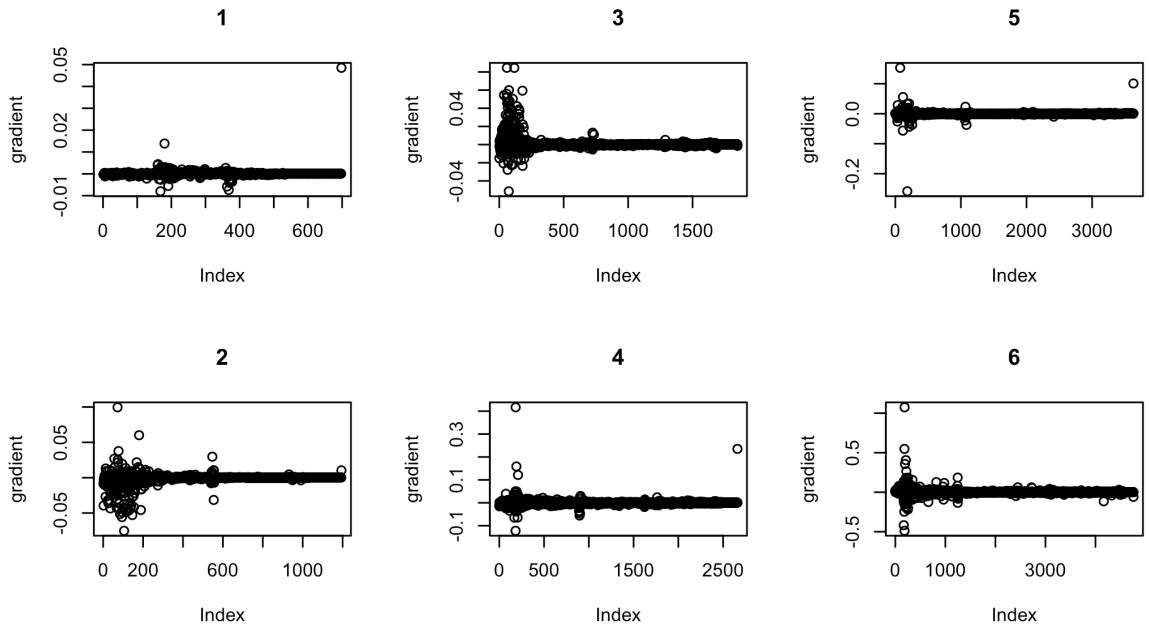
```
layout(mat = matrix(1:15, nrow = 5, ncol = 3))
for (mod in we_1_unconstrained[4:6]) { # last 3 models
    plot(mod, var.colors=1, add.smooth = T)
}
```



Supplementary Figure 2. 4: Diagnostic plots of normal randomized quantile residuals for the unconstrained models in worked example 1. Colors represent different species in the dataset. Left column = 4 latent variable, middle = 5 and right = 6.

As a supplement to the diagnostic plots, we also look at the gradients of the parameter estimators (Sup. Figure 2. 5) and the sign of the estimator variances, in order to get an indication of whether the model has converged stably. Here, we see that the gradients values are all very close to 0, which indicates stable convergence for all our models.

```
layout(mat = matrix(1:6, nrow = 2, ncol = 3))
for (mod in we_1_unconstrained) { # for each model
  plot(c(mod$TMBfn$gr(mod$TMBfn$par)), ylab = "gradient",
       main=mod$num.lv)
}
```



Supplementary Figure 2. 5: plots of the gradient values for the parameter estimators n the latent variable models. Left column = 1 latent variable, middle = 2 and right = 3

Looking at the sign of the diagonal of the covariance matrix for the parameter estimators, we see, however, that several of the models give negative values, which indicates poor model fit, even after running for several iterations. Based on this, we decide to stick with the model with three latent variables for further analysis, as it does not have any negative values, even though the 5 latent variable model had lower AIC..

```
# sign of the estimator variances
for (mod in we_1_unconstrained) { # for each model
  print(table(sign(diag(mod$Hess$cov.mat.mod))))
}
```

```
 1
61

 1
80

 1
98

  1
115

 -1   1
  3 128

 -1   1
 20 126
```

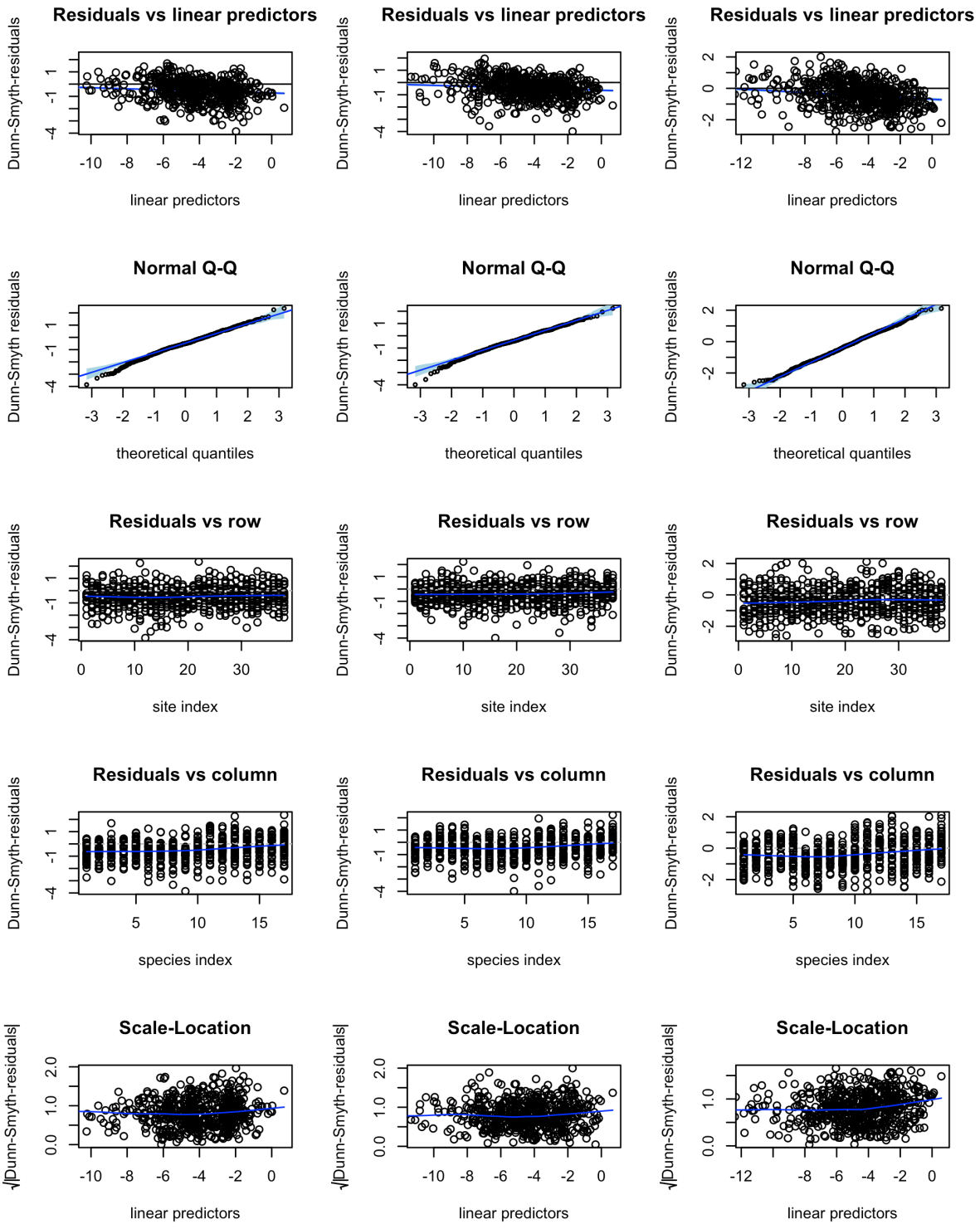
3.4.2 Concurrent latent variable model

The same procedure is done for the concurrent ordination models, and see that models have an AIC(c) minimum for two latent variables Sup. Table 2. 2. We here use the Corrected Aikake Information Criterion (AICc, **hurvichRegressionTimeSeries1989?**), due to the smaller sample size of our data.

Supplementary Table 2. 2: Information criteria for the concurrent models with different number of latent variables. Difference = difference in AICc between each model and the model with the lowest score

	df	AICc	Difference
1 latent variables	56	270.3	7.5
2 latent variables	75	262.7	0.0
3 latent variables	92	276.3	13.5

```
layout(mat = matrix(1:15, nrow = 5, ncol = 3))
for (mod in we_1_concurrent) { # first 3 models
    plot(mod, var.colors=1)
}
```



Supplementary Figure 2. 6: Diagnostic plots of normal randomized quantile residuals for the unconstrained models in worked example 1. Colors represent different species in the dataset. Left column = 1 latent variable, middle = 2 and right = 3.

```
layout(mat = matrix(1:3, nrow = 1, ncol = 3))
for (mod in we_1_concurrent) { # for each model
  plot(c(mod$TMBfn$gr(mod$TMBfn$par)))
}
```

Looking at the sign of the estimator variances, we see that the model with two latent variables does have some negative values, which is unfortunate. However, we will - at least in this instance, give it less weight compared to the AIC value, and thus stick with the two-variable model – primarily in order to showcase the two-dimensional environmental biplot. We do however acknowledge that this is perhaps somewhat inconsistent with our argument for the unconstrained model, and that one might make a plausible argument in this case to alternatively stick with the one-variable model, or alternatively run the two-variable model with more iterations to ensure convergence.

```
# sign of the estimator variances
for (mod in we_1_concurrent) { # for each model
  print(table(sign(diag(mod$Hess$cov.mat.mod))))
}
```

```
 1
56

-1  1
10 66

-1  1
 7 88
```

3.5 Visualizing results

3.5.1 Unconstrained latent variable model

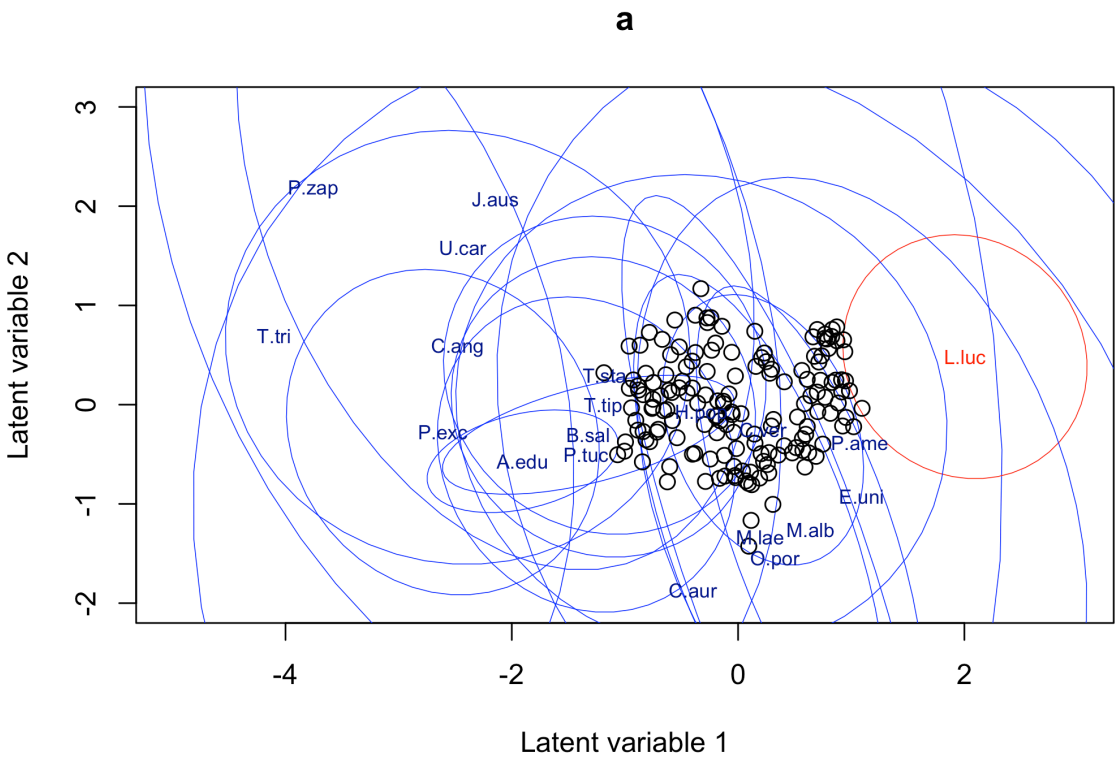
Here, we include all the code to make the figures that go into Figure 4 (see also the script `example_1/ex1_figures.R`), in addition to the model summaries of the two models (i.e. the unconstrained and concurrent), the species-specific coefficients for all the environmental predictors (Sup. Figure 2. 10), variance partitioning, and goodness-of-fit measures of the two models.Unconstrained latent variable model

```r
we_1_uc_mod <- we_1_unconstrained[[3]]
```

```r
summary(we_1_uc_mod)
```

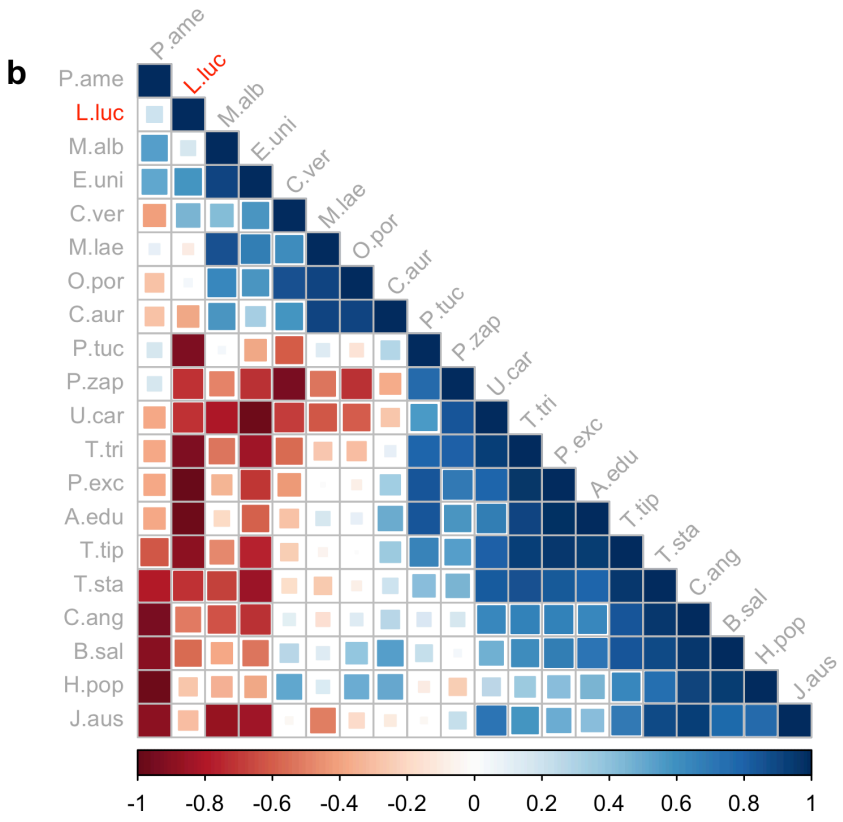
```
Call:
gllvm(y = species_f, family = "tweedie", num.lv = n_lv, row.eff = "random",
    Power = NULL, seed = 1, n.init = 10, trace = T)

Family:  tweedie

AIC:  1128.623 AICc:  1134.921 BIC:  1722.958 LL:  -466.3 df:  98

Informed LVs:  0
Constrained LVs:  0
Unconstrained LVs:  3

Formula:  ~ 1
LV formula:  ~ 0
Row effect:  ~(1 | sample)

 Random effects:
 Name                Variance Std.Dev
 (Intercept)|sample 1e-04    0.0096
```

```r
par(mfrow=c(1,1), adj=0.5) # set up plotting window

### Unconstrained ordination diagram ####
col_list <- c(rep("darkblue", 8), "red", rep(rep("darkblue", 11)))
col_list_2 <- c(rep("blue", 8), "red", rep(rep("blue", 11)))
ordiplot(we_1_uc_mod, biplot=T, symbols=T,
         predict.region = "species",
         spp.colors =  col_list, col.ellips = col_list_2,
         main="a", ylim=c(-2,3), xlim=c(-5,3))
```



Supplementary Figure 2. 7: Ordination biplot of the uinconstrained ordination model, equivalent to Figure 4a in the main manuscript.

```
cr0 <- getResidualCor(we_1_uc_mod)
b <- corrplot(cr0[order.single(cr0), order.single(cr0)],
              diag = TRUE, type = "lower",
              method = "square",
              tl.cex = 0.8, tl.srt = 45,
              tl.col = c("darkgrey","red", rep("darkgrey", 18)))
title(main = "b", adj = 0)
```



Supplementary Figure 2. 8: Between-species correlation from the unconstrained ordination. Equivalent to figure 4b in the main manuscript.

Looking at the built-in goodness-of-fit measures of the `gllvm` package, we see that the correlation between the predicted species observations and the acutual data is very high (`0.92`), and various error terms, such as root mean squared error (RMSE) and mean absolute error (MAE), are quite low, indicating that our model is a good fit to our original response data.

```
goodnessOfFit(we_1_uc_mod)
```

```
$cor
[1] 0.9099658

$RMSE
[1] 0.08198431

$MAE
```

```
[1] 0.03511619

$MARNE
[1] 0.05906984
```

3.5.2 Concurrent latent variable model

Below is the model summary of the concurrent ordination model with two concurrent latent variables. we see that the parameter estimates for the effects of soil moisture and soil nitrogen content on latent variable 1 are the only latent variable predictors significant at a 95% confidence level.

```
# extract the model for visualization
we_1_cc_mod <- we_1_concurrent[[2]]
```

```
summary(we_1_cc_mod)
```

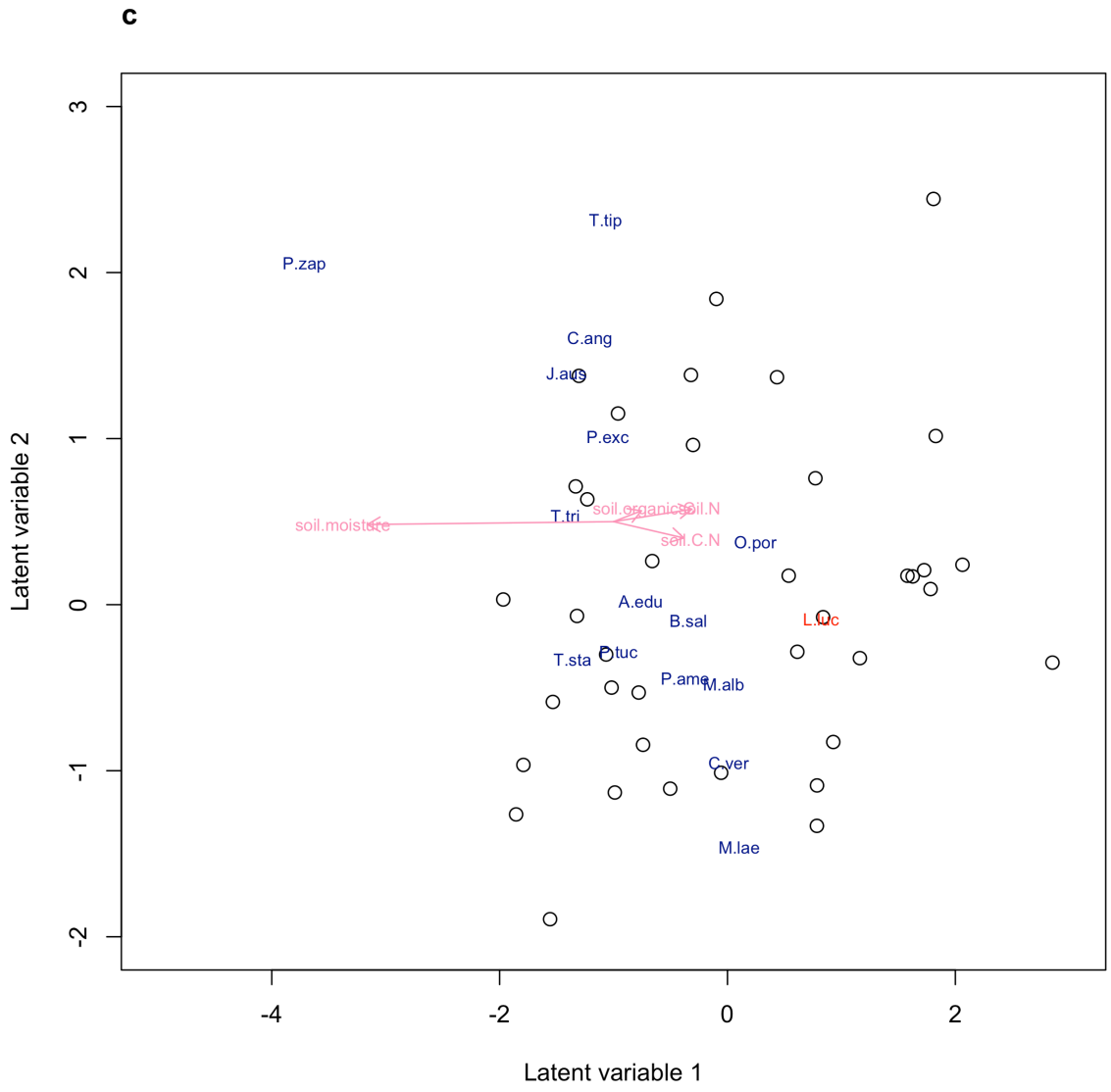
```
Warning in summary.gllvm(we_1_cc_mod): Negative diagonal entries detected in the covariance m


Call:
gllvm(y = species_sub, X = env_sub, family = "tweedie", num.lv.c = n_lv,
    lv.formula = ~soil.organic.C + soil.N + soil.C.N + soil.moisture,
    row.eff = "random", method = "LA", Power = NULL, seed = 123,
    trace = F, n.init = 5)

Family:  tweedie

AIC:  242.7136 AICc:  262.7136 BIC:  578.0236 LL:  -46.36 df:  75

Informed LVs:  2
Constrained LVs:  0
Unconstrained LVs:  0
Residual standard deviation of LVs:  0.4027 0.0999

Formula:  ~ 1
LV formula:  ~soil.organic.C + soil.N + soil.C.N + soil.moisture
Row effect:  ~(1 | sample)

 Random effects:
 Name                Variance Std.Dev
```

```
 (Intercept)|sample 0.0907   0.3012

Coefficients LV predictors:
                     Estimate Std. Error z value Pr(>|z|)
soil.organic.C(CLV1)  0.094792   0.134661   0.704  0.48148
soil.N(CLV1)          0.270208   0.133246   2.028  0.04257 *
soil.C.N(CLV1)        0.240018   0.155719   1.541  0.12323
soil.moisture(CLV1)  -0.805487   0.288147  -2.795  0.00518 **
soil.organic.C(CLV2)  0.025279   0.043485   0.581  0.56102
soil.N(CLV2)          0.029483   0.050007   0.590  0.55548
soil.C.N(CLV2)       -0.038353   0.021164  -1.812  0.06996 .
soil.moisture(CLV2)  -0.006437   0.079189  -0.081  0.93522
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
### Concurrent ordination diagram ####
col_list <- c(rep("darkblue", 5), "red", rep(rep("darkblue", 11)))
we_1_cc_mod$params$theta
```

```
             CLV1        CLV2
A.edu  1.00000000   0.0000000
B.sal  0.44697553   1.0000000
C.ang  1.64058020 -14.9147555
C.ver -0.04499237   9.0660183
J.aus  1.90698263 -12.9336632
L.luc -1.08567967   0.6280295
M.alb  0.02708833   4.5623707
M.lae -0.18359855  13.8804532
O.por -0.31098979  -3.5591910
P.exc  1.41632471  -9.3886164
P.ame  0.47309673   4.3278102
P.tuc  1.25188843   2.9332926
P.zap  4.95658873 -18.6024248
T.sta  1.77864456   3.4819518
T.tri  1.89143503  -4.7526245
T.tip  1.48325455 -21.7223435
U.car  8.00105647 -39.9133270
```

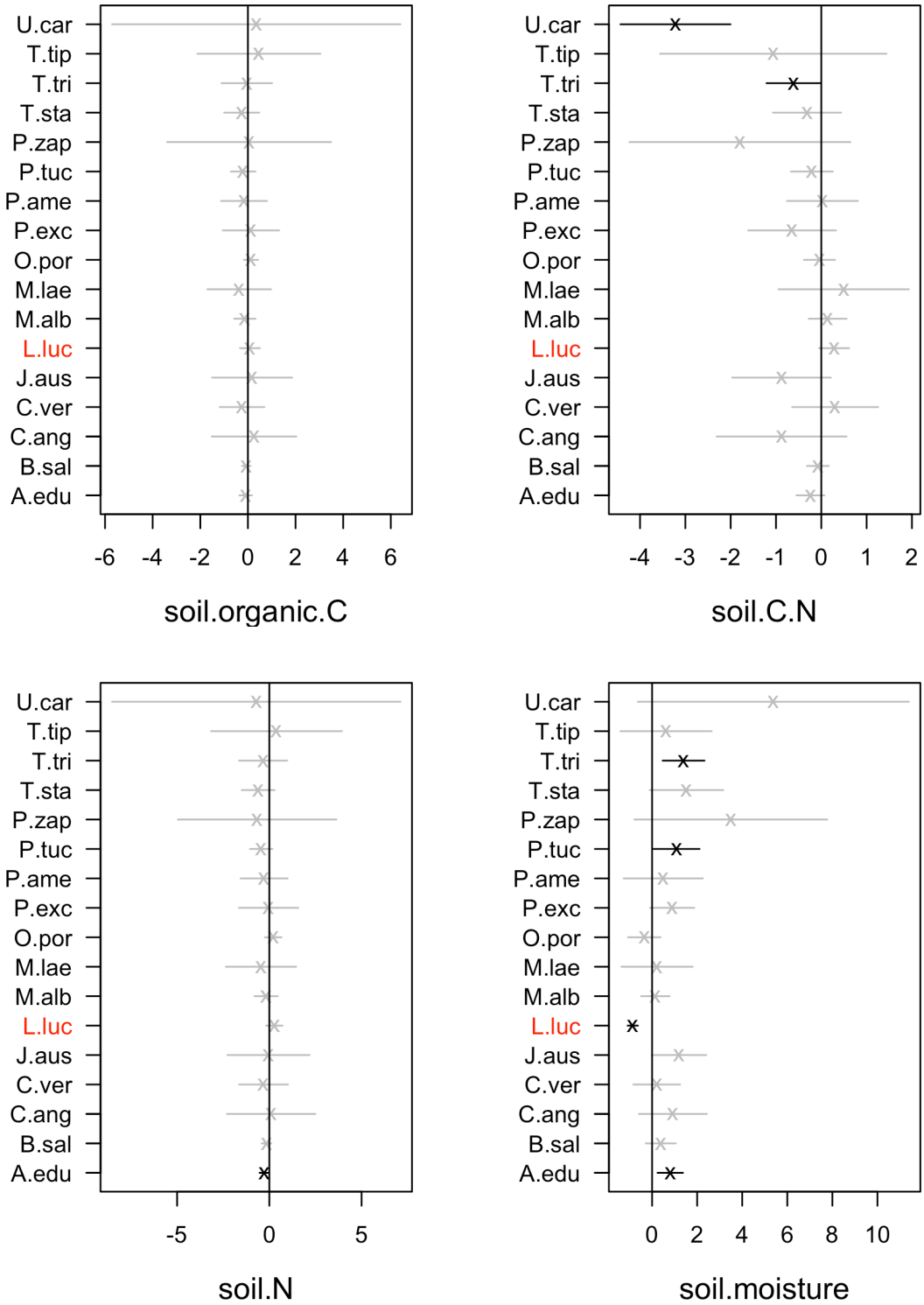
```
a <- ordiplot(we_1_cc_mod, biplot=T, symbols=T, rotate=T,
              spp.colors =  col_list, main="", ylim=c(-2,3), xlim=c(-5,3))
title(main = "c", adj = 0)
```



Supplementary Figure 2. 9: Ordination bipot for the concurrent ordination model. Equivalent to figure 4c in the main manuscript.

```
layout(mat = matrix(1:4, nrow = 2, ncol = 2))
for (p in 1:4) {
  col_list <- c(rep("black", 5), "red", rep(rep("black", 11))) # colormap to highlight L.luc
coefplot(we_1_cc_mod, which.Xcoef = p, order = F, cex.ylab =0.0001)
Map(axis, side=2, at=1:length(col_list), cex.lab =0.1, # highlight L.luc in red
```

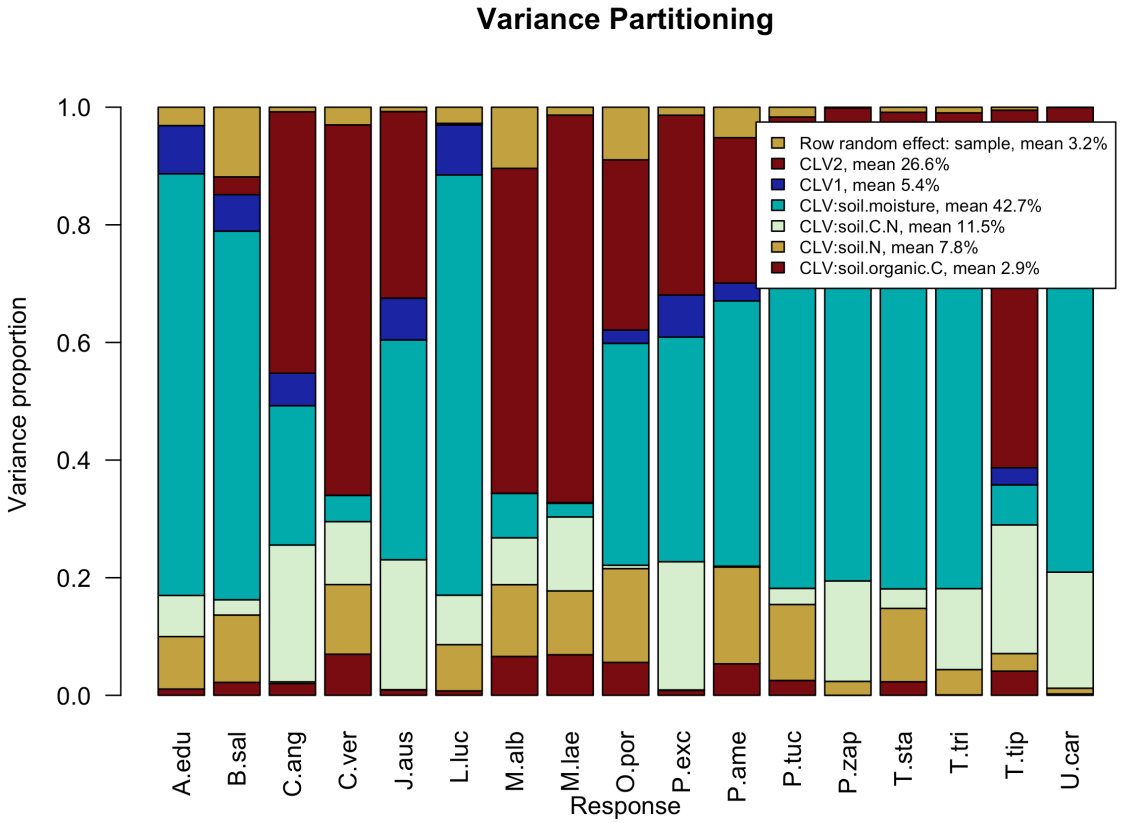
```
    col.axis=col_list, labels=rownames(we_1_cc_mod$params$theta),
    lwd=0, las=1)
axis(2,at=1:3,labels=FALSE)
}
```



Supplementary Figure 2. 10: Coefficient plots for the species-specific effects of the environmental predictors, from the concurrent latent variable model. The plot for soil moisture is equivalent to Figure 4d in the main manuscript

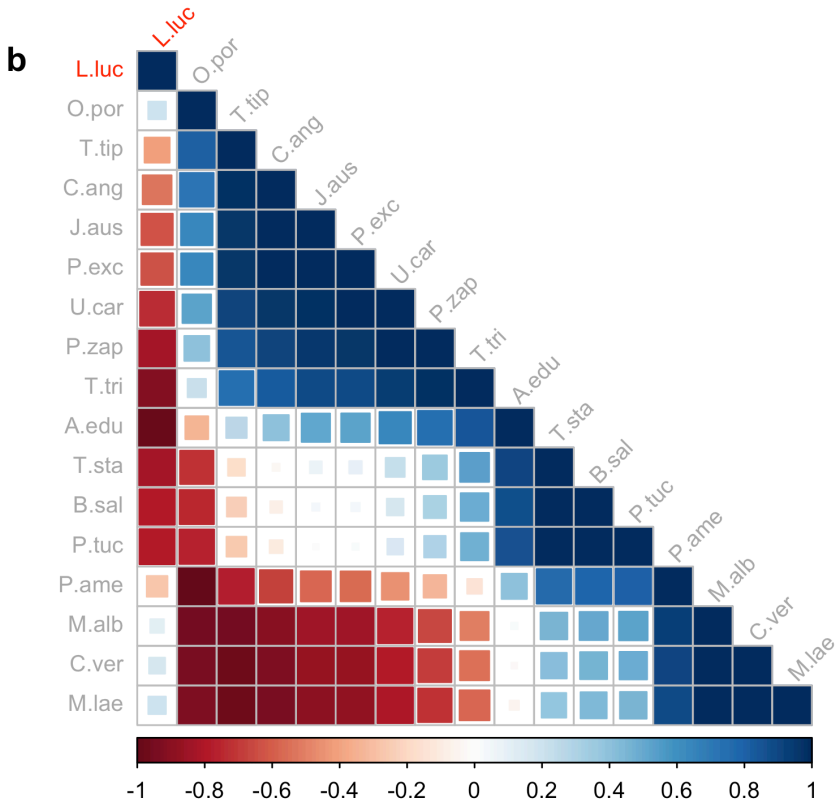
We can also look at the variance partitioning of the concurrent latent variable model Sup. Figure 2. 11. Here, we see that the linear predictor variance of *Ligustrum lucidum* is explained predominantly by the soil moisture content. We also note that a considerable amount of the variance for some species (26% in total) is due to residual (i.e. unexplained) variance in latent variable 2 (CLV2), indicating that their may be important additional environmental or biotic factors that were not included in the study that are important for shaping the species community. And as Sup. Figure 2. 11 shows, this residual variation is resonsible for significant negative correlation between L.luc and the other study species.

```
vp_we_1_cc_mod <- varPartitioning.gllvm(we_1_cc_mod)
plot(vp_we_1_cc_mod, args.legend = list(cex=0.7),
     col=hcl.colors(5, "Roma"), las=2)
```



Supplementary Figure 2. 11: Variance partitiining of the latent variable model in worked example 1

```
cr <- getResidualCor(we_1_cc_mod)
b <- corrplot(cr[order.single(cr), order.single(cr)],
              diag = TRUE, type = "lower", method = "square",
              tl.cex = 0.8, tl.srt = 45,
              tl.col = c("red",rep("darkgrey", 16)))
title(main = "b", adj = 0)
```



Supplementary Figure 2. 12: Residual between-species correlation from the concurrent ordination (after accounting for the latent variable predictors.

4 Worked example 2

4.1 Preparing data

We begin by loading the environmental and species data from Mehlhoop et al. (2022), downloaded from the author's Open Science repository (link) as `.rda` files.

```
# load species data
load("example_2/data/species.rda")
species <- species_sub1 # overwrite old species data frame

# load environemntal data
load("example_2/data/environmental_variables.rda")
```

Looking at the environmental data, we see that the variable `gf` (region) is coded with "ref" as its own category, for each reference plot within a `site`, regardless of which `gf` cateogry the other plots within that site belongs to. We change that to make sure that each site has exactly one `gf` category:

```
env <- env.var_sub1
for (s in unique(env.var_sub1$site)) {
  sb <- env.var_sub1[env.var_sub1$site==s,]
  env[env$site==s,]$gf <- sb[1,]$gf
}
env$gf <- droplevels(env$gf, "ref")
```

We also need to fix the data type of the grain size variable

```
env$grain_size_stand_f <- as.numeric(env$grain_size_stand_f)
```

4.2 Data exploration

4.2.1 Species data

We start by looking at the number of non-zero observations in our data set. This can be done quickly by using the `table` function:

```
table(colSums(species>0)) # number of observations per column (species)
```

```
  1   2   3   4   5   6   7   8   9  10  11  12  14  15  16  18  19  20  22  24
 53  24  13  10   7   6   5   5   4   5   4   4   4   1   4   4   2   1   5   1
 30  32  34  39  40  44  45  46  47  49  52  55  56  57  59  69  70  73  88 138
  1   1   2   1   1   2   1   1   1   1   2   2   1   1   2   1   1   1   1   1
163
  1
```

```
table(rowSums(species>0)) # number of observations per row (site)
```

```
 2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 21
 4 14 18 27 18 35 27 35 26 20 16  9 11 14  4  1  1  1  1
```

We see that 53 of the species are only observed once in the data.

In order to improve the stability of the model, and because our study is site-focused rather than species-focused (see main manuscript), we therefore decide to filter out all species that occur in three or fewer plots:

```
Y <- species[,colSums(species > 0)>3]
```

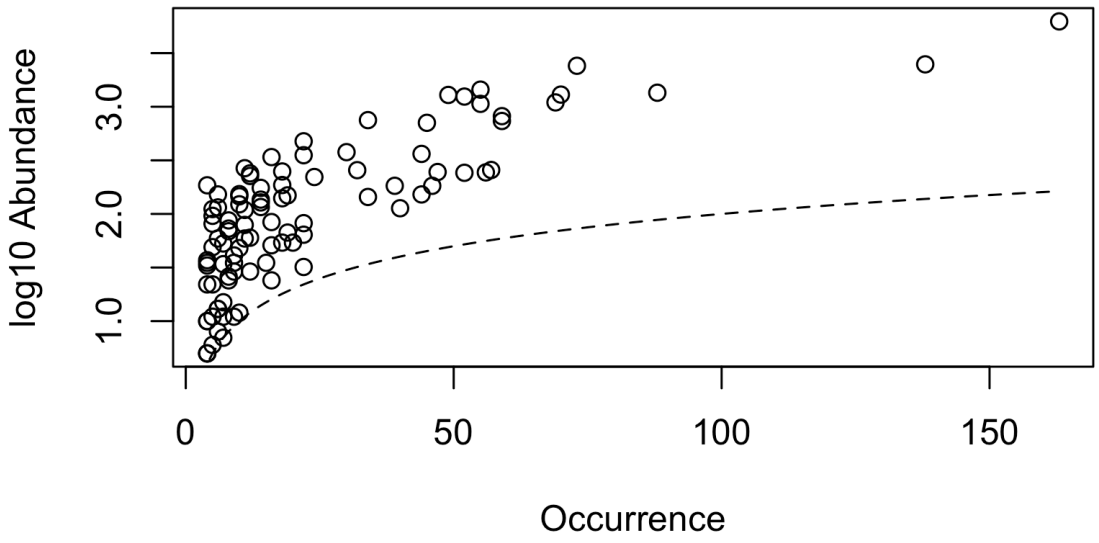
After removing the species, we see that each plot still has at least two species present:

```
table(rowSums(Y>0)) # number of observations per row (site)
```

```
 2  3  4  5  6  7  8  9 10 11 12 13 14 15 18
 6 14 24 29 24 28 32 37 18 26 13  8 11  9  3
```

Looking at an abundance occupancy-plot, we can see that the prevalence of species is roughly log-linear, and there are no clear outliers from this trend. In other words, there does not appear to be any individual species that might

```
mefa::aoplot(Y)
```



4.2.2 Environmental data

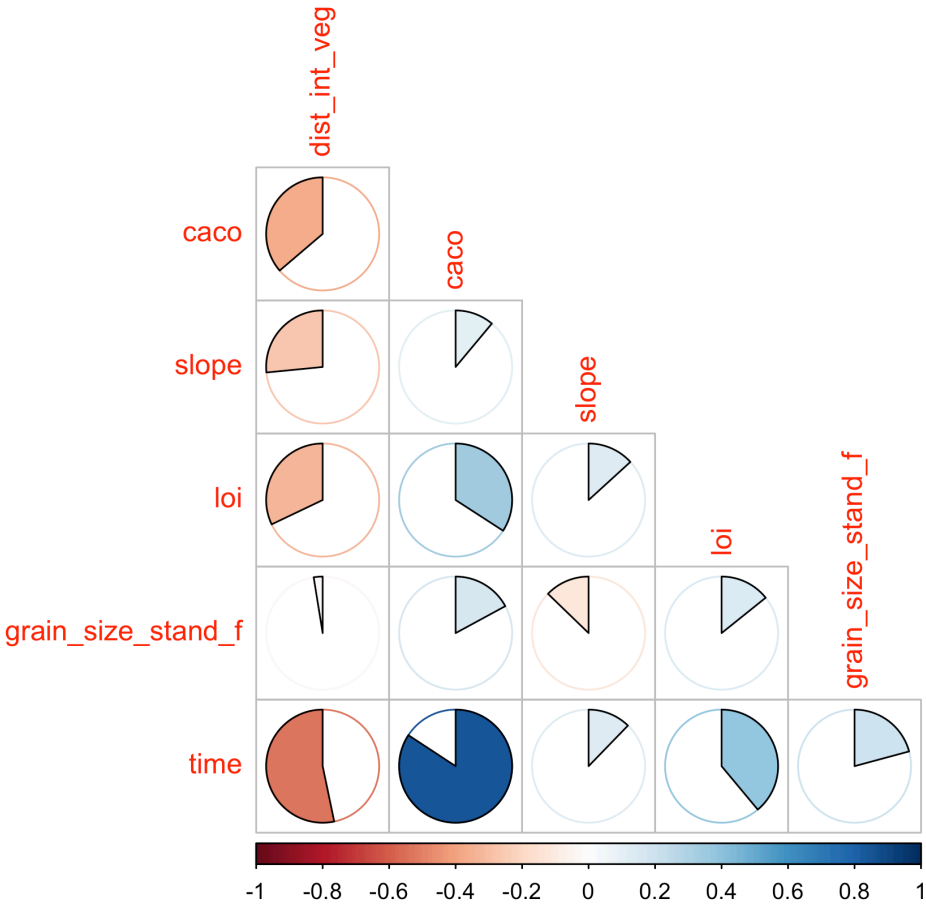
Next, we scale our environmental predictors

```
env_sub <- env |>
  select(dist_int_veg, caco, slope, loi, grain_size_stand_f) |>
  scale() |> as.data.frame()
env_sub$method <- env$method
env_sub$time <- env$years_since_n # rename the column to be shorter
```

Looking at the corrogram for the continuous predictors Sup. Figure 2. 13, we see no significant co-linearity, except for the relationship between canopy cover and time since restoration, which is likely due to the fact that all sites in the reference category are coded with a high "years since restotation" value of 40. And since we are not planning on using this value in our model, see below, it will not be a problem for us.

```
env_sub |>
  select(dist_int_veg, caco, slope,
         loi, grain_size_stand_f, time) |>
```

```
  cor() |>
  corrplot::corrplot(type = "lower",
                     method = "pie", diag= FALSE)
```



Supplementary Figure 2. 13: Correlation plot for the observed environmental predictors in worked example 1

Indeed, looking only at the correlation for the non-reference sites reveals that both the correlations between canopy cover and distance to vegetation with year disappears.

```
env_sub |>
  filter(method != "ref") |>
```

```
  select(dist_int_veg, caco, slope,
         loi, grain_size_stand_f, time) |>
  cor()
```

```
                   dist_int_veg         caco        slope          loi
dist_int_veg         1.00000000  0.240323679 -0.287203827 -0.166048097
caco                 0.24032368  1.000000000 -0.025436855 -0.003143336
slope               -0.28720383 -0.025436855  1.000000000 -0.007773989
loi                 -0.16604810 -0.003143336 -0.007773989  1.000000000
grain_size_stand_f   0.11215120  0.017015895 -0.188588521  0.100863684
time                 0.05681324  0.270426021 -0.104772695 -0.068541369
                   grain_size_stand_f        time
dist_int_veg                0.11215120  0.05681324
caco                        0.01701590  0.27042602
slope                      -0.18858852 -0.10477269
loi                         0.10086368 -0.06854137
grain_size_stand_f          1.00000000  0.01519351
time                        0.01519351  1.00000000
```

In order to scale the variable `years_since_n` (time since restoration intervention), we have to do some additional tweaking. Since the reference sites are coded in the data with a value of 40, and we are not including reference sites in our interaction effect, we scale the variable using only the non-reference sites:

```
env_sub$time <- (env_sub$time - mean(env_sub$time[env_sub$time<40]))/
  (sqrt(var(env_sub$time[env_sub$time<40])))
```

Next, we make a custom design matrix for the environmental predictors, with interactions between `years_since_n` and `method` , excluding interaction with the `reference` site level of `method`.

```
# making design matrix X
X <- model.matrix(~ method/time + dist_int_veg + caco +
                    slope + loi + grain_size_stand_f,
                  data = env_sub)

X <- as.data.frame(X[,-c(1,2,3,4,10)]) # remove time x reference interaction

colnames(X)[6:8] <- c("natXtime", "pnXtime", "seedXtime")
X$method <- factor(env_sub$method, levels=c("ref", "nat", "pn", "seed"))
X$gf <- env_sub$gf
```

4.3 Model fitting

We try to fit a concurrent model using the ordered beta distribution. To do this, we have to convert our species data into fractions

```
Y <- Y/100
```

We also order our species columns from most to least common, in order to help with convergence:

```
Y <- Y[,names(sort(colSums(Y), decreasing =T))] # sort by most common species first
```

We then attempt to fit the model with 1 to 3 latent variables. Because they took some time to fit with each iteration, we kept the `n.init = 2` for all runs.

```
we_2_concurrent <- list() # make list for model object
for (n_lv in 1:3) {  # run up to 3 LVs
  we_2_concurrent[[n_lv]] <- gllvm(
    y = Y,
    X = X,
    studyDesign = env, # study design matrix (for row.effects)
    num.lv.c = n_lv, # 1 concurrent latent variable
    family = "orderedBeta", # ordered beta response distribution
    formula = ~ (1|gf), # region as species-specific random effect
    row.eff = ~ (1|site), # site as community-wide random effect
    method = "EVA", # extended variational approximation method
    lv.formula = ~ method + natXtime + pnXtime + # predictors
      seedXtime + dist_int_veg + caco + slope + loi + grain_size_stand_f,
    disp.formula = rep(1, ncol(Y)), # same dispersion param for all species
    n.init = 2, # number of starting iterations (fewer due to slow fitting)
    seed = 1 # seed
  )
}
```

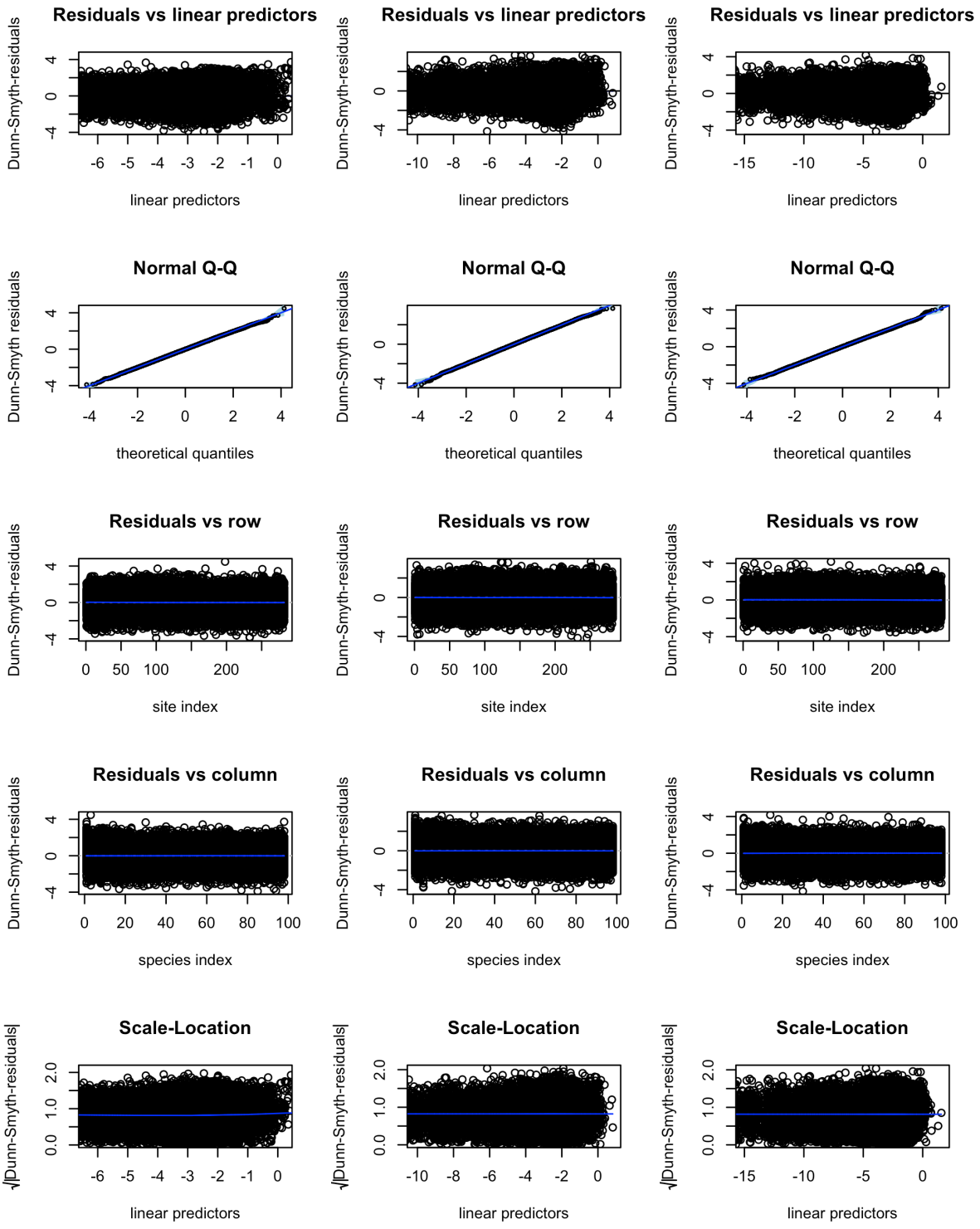
4.4 Model checking

Looking at the diagnostic plots for the two converged models Sup. Figure 2. 14, we see that none of them indicate notable deviations from the model assumptions. The model with three latent variables has the lowest AIC. However, we see that the gradient Sup. Figure 2. 15 have considerable outlier values for the two-and three- variable models, as well as >10 negative estimator variance estimates, indicating poor convergence.

Supplementary Table 2. 3: Information criteria for the concurrent models in worked example 2 with different number of latent variables

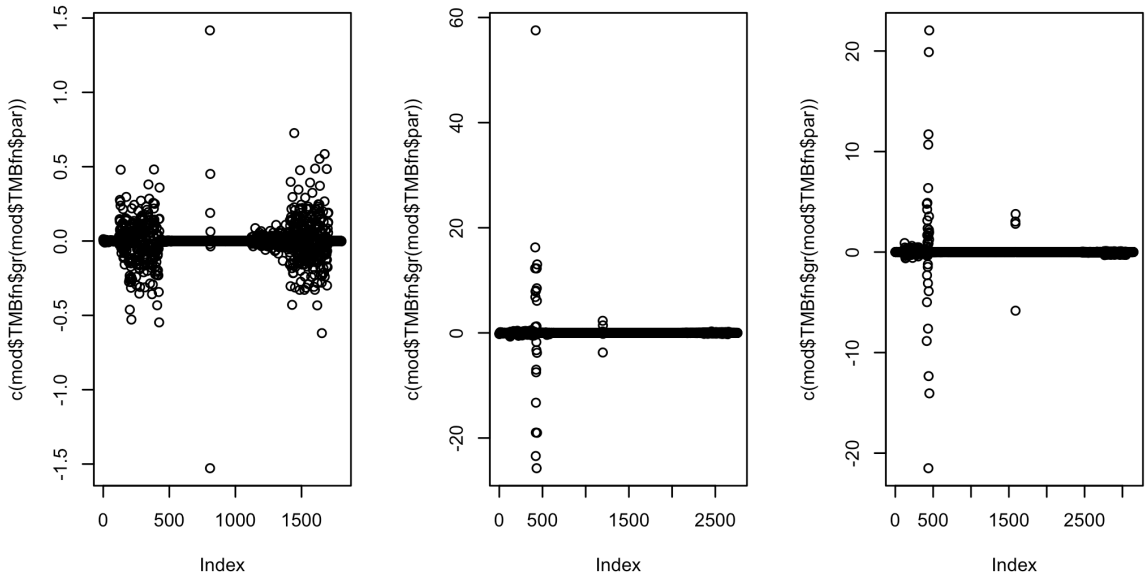
	df	AIC	Difference
1 latent variables	413	7514.8	459.4
2 latent variables	520	7179.2	123.8
3 latent variables	625	7055.4	0.0

```
layout(mat = matrix(1:15, nrow = 5, ncol = 3))
for (mod in we_2_concurrent) { # for each model
    plot(mod, var.colors=1)
}
```



Supplementary Figure 2. 14: Diagnostic plots of normal randomized quantile residuals for the concurrent models in worked example 2. Colors represent different species in the dataset. Left column = 1 latent variable, middle = 2 and right = 3.

```
layout(mat = matrix(1:3, nrow = 1, ncol = 3))
for (mod in we_2_concurrent) { # for each model
  plot(c(mod$TMBfn$gr(mod$TMBfn$par)))
}
```



Supplementary Figure 2. 15: plots of the gradient values for the parameter estimators n the latent variable models in worked example 2. Left column = 1 latent variable, right = 2

```
# sign of the estimator variances
for (mod in we_2_concurrent) { # for each model
  print(table(sign(diag(mod$Hess$cov.mat.mod))))
}
```

```
-1   1
 2 314

-1   1
15 409
```

```
 -1   1
  4 527
```

We also see that the residual variation in the latent variables is very small for all the constrained latent variables. Therefore, we decide to re-fit all the modes with constrained (i.e. fully predictor informed) latent variables.

```
for (mod in we_2_concurrent) { # for each model
  print(mod$params$sigma.lv)
}
```

```
        LV1
0.007552196
         LV1          LV2
1.113610e-06 4.894789e-04
         LV1          LV2          LV3
2.650746e-08 2.042170e-07 6.968475e-05
```

4.4.1 Model re-fitting

```
we_2_constrained <- list() # make list for model object
for (n_lv in 1:3) {  # run up to 3 LVs
  we_2_constrained[[n_lv]] <- gllvm(
    y = Y,
    X = X,
    studyDesign = env, # study design matrix (for row.effects)
    num.RR = n_lv, # constrained latent variables
    family = "orderedBeta", # ordered beta response distribution
    formula = ~ (1|gf), # region as species-specific random effect
    row.eff = ~ (1|site), # site as community-wide random effect
    method = "EVA", # extended variational approximation method
    lv.formula = ~ method + natXtime + pnXtime + # predictors
      seedXtime + dist_int_veg + caco + slope + loi + grain_size_stand_f,
    disp.formula = rep(1, ncol(Y)), # same dispersion param for all species
    n.init = 2, # number of starting iterations (fewer due to slow fitting)
    seed = 1 # seed
  )
}
```

4.4.2 Model re-checking

Re-fitting the models does however not seem to fully fix the convergence issues with the more latent variable-heavy models, as seen in Sup. Figure 2. 16 and the code output below. While the gradients of the 3-lv model look better, it has a greater number of negative variances, and the opposite is true for the 2 lv model.

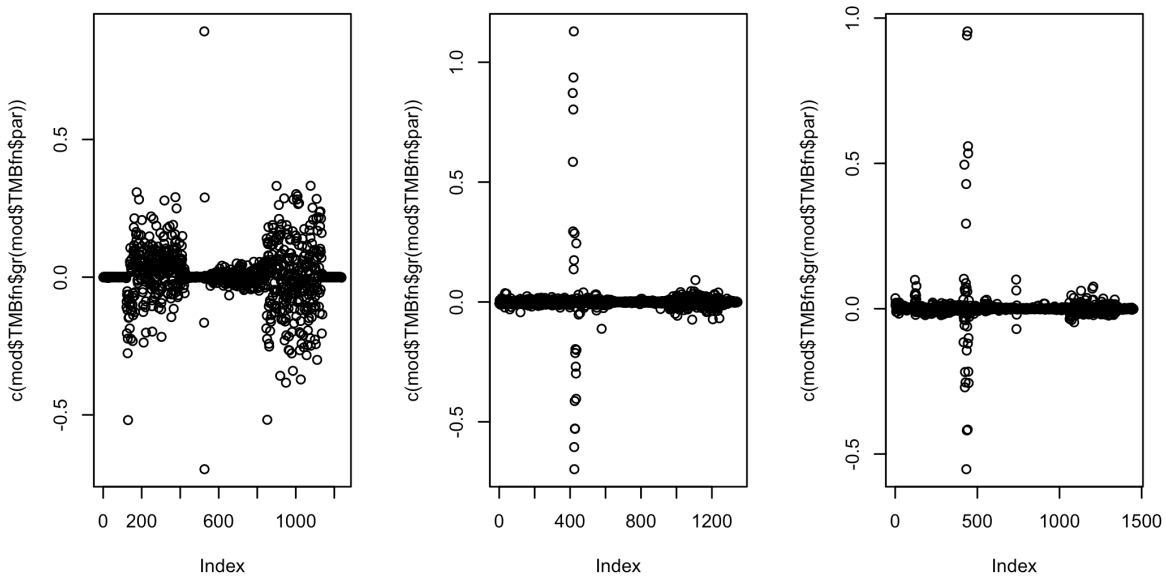
We therefore decide to process with the 1-latent variable model for the visualisations and analysis.

Supplementary Table 2. 4: Information criteria for the concurrent models in worked example 2 with different number of latent variables

	df	AIC	Difference
1 latent variables	412	7513.1	456.9
2 latent variables	518	7203.1	146.9
3 latent variables	622	7056.2	0.0

```
layout(mat = matrix(1:15, nrow = 5, ncol = 3))
for (mod in we_2_concurrent) { # for each model
    plot(mod, var.colors=1, add.smooth = T)
}
```

```
layout(mat = matrix(1:3, nrow = 1, ncol = 3))
for (mod in we_2_constrained) { # for each model
  plot(c(mod$TMBfn$gr(mod$TMBfn$par)))
}
```



Supplementary Figure 2. 16: plots of the gradient values for the parameter estimators n the latent variable models in worked example 2. Left column = 1 latent variable, right = 2

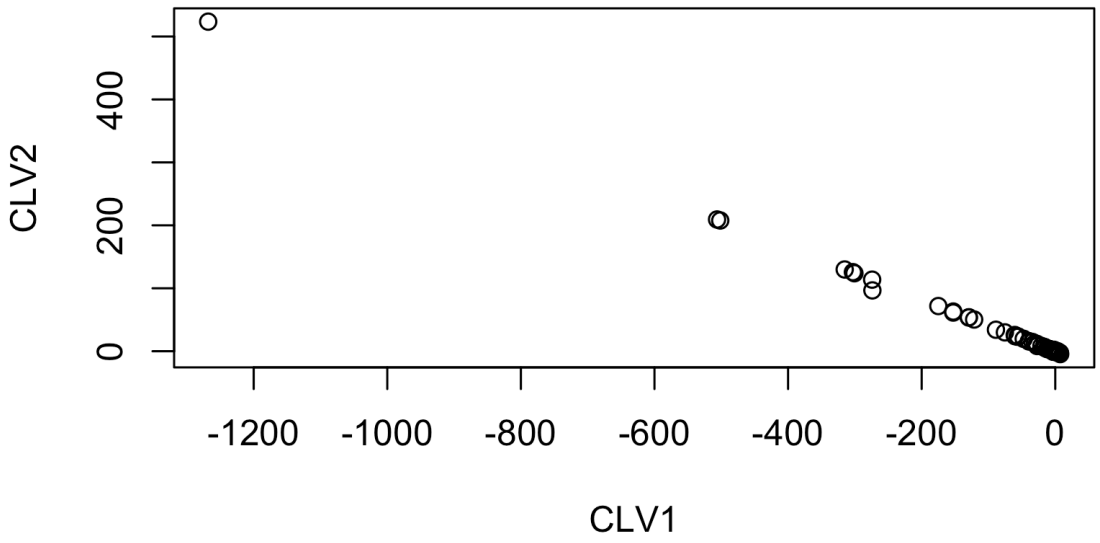
```
# sign of the estimator variances
layout(mat = matrix(1:3, nrow = 1, ncol = 3))
for (mod in we_2_constrained) { # for each model
  print(table(sign(diag(mod$Hess$cov.mat.mod))))
}
```

```
  1
315

 -1   1
  3 419

 -1   1
  5 523
```

We also see that many of the species loadings estimated by the 2-lv model are both severely "blown up" and almost perfectly positively correlated, supporting the notion that the model is

poorly converged, and that there is not separate second community gradient that is represented with one latent variable

```
plot(getLoadings(we_2_constrained[[2]]))
```



4.5 Visualizing results

Here, we include all the code that goes into making Figure 5 in the main manuscript, as well as the model summary (see below), variance partitioning and goodness-of-fit measures from the model. The code for Sup. Figure 2. 17 in particular makes use of some custom code adapted from the `ordiplot()` function in the `gllvm` package in order to scale the site scores, latent variable predictors and species loading in proportion to one another (see comments in code).

```
we_2_cn_mod <- we_2_constrained[[1]]
```

The model summary is shown below:

```
summary(we_2_cn_mod)
```

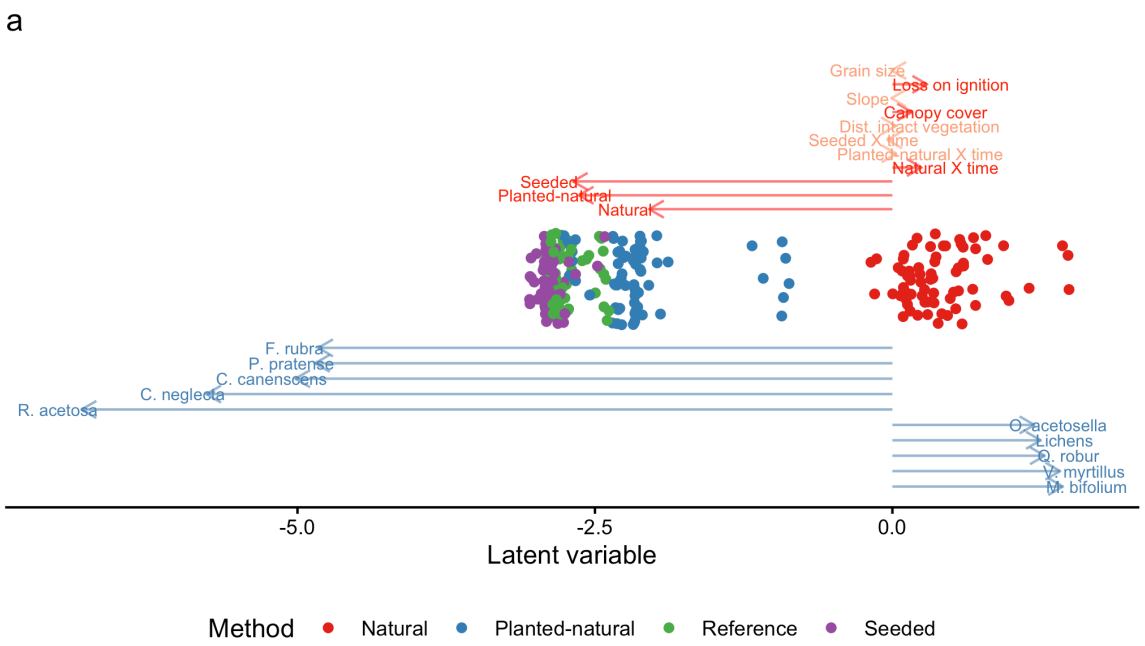
```
Call:
gllvm(y = Y, X = X, formula = ~(1 | gf), family = "orderedBeta",
    num.RR = n_lv, lv.formula = ~method + natXtime + pnXtime +
        seedXtime + dist_int_veg + caco + slope + loi + grain_size_stand_f,
    studyDesign = env, row.eff = ~(1 | site), method = "EVA",
    seed = 1, disp.formula = rep(1, ncol(Y)), n.init = 2)

Family:  orderedBeta

AIC:  7513.105 AICc:  7525.605 BIC:  10902.58 LL:  -3345 df:  412

Informed LVs:  0
Constrained LVs:  1
Unconstrained LVs:  0

Formula:  ~(1 | gf)
LV formula:  ~method + natXtime + pnXtime + seedXtime + dist_int_veg + caco + slope + loi + g
Row effect:  ~(1 | site)

 Multispecies random effects:
 Name Variance Std.Dev Corr
 gf1  0.3881   0.6230
 gf2  0.4271   0.6536  -0.5566
 gf3  0.0864   0.2940  -0.6001 -0.3304

 Random effects:
 Name             Variance Std.Dev
 (Intercept)|site 0.0540   0.2325

Coefficients predictors:
    Estimate Std. Error z value Pr(>|z|)
gf2   0.1891     0.2007   0.942   0.3461
gf3   0.3019     0.1735   1.740   0.0818 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Coefficients LV predictors:
                         Estimate Std. Error z value Pr(>|z|)
methodnat(CLV1)          -0.764685   0.105591  -7.242 4.42e-13 ***
methodpn(CLV1)           -0.985598   0.129906  -7.587 3.27e-14 ***
methodseed(CLV1)         -1.004025   0.132547  -7.575 3.60e-14 ***
natXtime(CLV1)            0.091919   0.026895   3.418 0.000631 ***
```

```
pnXtime(CLV1)              0.012401   0.010792   1.149 0.250496
seedXtime(CLV1)           -0.015854   0.009660  -1.641 0.100752
dist_int_veg(CLV1)         0.010920   0.006099   1.790 0.073387 .
caco(CLV1)                 0.061219   0.015769   3.882 0.000104 ***
slope(CLV1)               -0.002942   0.002922  -1.007 0.314036
loi(CLV1)                  0.109101   0.020795   5.247 1.55e-07 ***
grain_size_stand_f(CLV1) -0.002935   0.004149  -0.708 0.479212
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
## Make 1d ordiplot with arrows ####
## env loadings
lv_1RR <- getLV.gllvm(we_2_cn_mod, scale = T)
smry<-summary(we_2_cn_mod, rotate = F)
coefs <- smry$Coef.tableLV

## scaled species and site loadings
# code from the gllvm "ordiplot" function
choose.lv.coefs <- getLoadings.gllvm(we_2_cn_mod)
choose.lvs <- getLV.gllvm(we_2_cn_mod)
bothnorms <- vector("numeric",ncol(choose.lv.coefs))
for(i in 1:ncol(choose.lv.coefs)){
  bothnorms[i] <- sqrt(sum(choose.lvs[,i]^2)) *
    sqrt(sum(choose.lv.coefs[,i]^2))
}

## Scale both to unit norm then scale using bothnorms
scaled_cw_sites <- t(t(choose.lvs) / sqrt(colSums(choose.lvs^2)) *
                       (bothnorms^0.5))
scaled_cw_species <- choose.lv.coefs
for(i in 1:ncol(scaled_cw_species)){
  scaled_cw_species[,i] <- choose.lv.coefs[,i] /
    sqrt(sum(choose.lv.coefs[,i]^2)) * (bothnorms[i]^(0.5))
}

# Scale species coefs too
coef_scaled <- (coefs[,1] / sqrt(colSums(choose.lvs^2))) * (bothnorms^(0.5))

# extract species loadings for species with highest and lowest loadings
load_scaled_min <- scaled_cw_species[sort(scaled_cw_species[,1],
                                          decreasing = T,
                                          index.return=T)[[2]],] |>
```

```
  tail(6) |>
  head(5) # avoid a. sylvestris as it is a huge outlier
load_scaled_max <- scaled_cw_species[sort(scaled_cw_species[,1],
                                          index.return=T)[[2]],] |>
  tail(5)
load_scaled_ex <- c(load_scaled_min,load_scaled_max) # loadings to plot

# add together (names from inspecting the data)
arrow_names <- c(rev(c("Grain size", "Loss on ignition",
                   "Slope",  "Canopy cover", "Dist. intact vegetation",
                   "Seeded X time","Planted-natural X time", "Natural X time",
                   "Seeded", "Planted-natural", "Natural")),
                 c("F. rubra", "P. pratense", "C. canenscens",
                   "C. neglecta", "R. acetosa", "O. acetosella",
                   "Lichens", "Q. robur", "V. myrtillus", "M. bifolium"))

# y positions of loading arows and colors
y_pos <- c(seq(0.15,0.45, length.out=length(coefs[,1])),
           seq(-.15,-0.45, length.out=length(load_scaled_ex)))
colrs <- c(ifelse(coefs[,4]<0.05, "red", "lightsalmon"),
           rep("steelblue", length(load_scaled_ex))) # color = significance
x_pos <- c(coef_scaled, load_scaled_ex) # x pos of arrows

## make ggplot
ggplot() +
  geom_point(aes(x=scaled_cw_sites[,1], y=0, color=env$method),
             position = position_jitter(w = 0, h = 0.1)) +
  geom_segment(aes(x=0, y=y_pos, xend=x_pos, yend=y_pos),
               alpha = 0.6,
               color=colrs,
               arrow = arrow(type="open", length=unit(0.10, "inches"))) +
  geom_text(aes(x=(x_pos + ((x_pos/abs(x_pos)))*0.20), y=y_pos,
                label=arrow_names), size=2.5, alpha=1, color=colrs) +
  theme_classic() +
  theme(axis.line.y = element_blank(),
        axis.ticks.y = element_blank(),
        axis.text.y = element_blank(),
        axis.line.x = element_line(),
        legend.position = "bottom") +
  labs(y="", x="Latent variable", color="Method", title="a") +
  scale_color_brewer(labels = c("Natural", "Planted-natural",
                                "Reference","Seeded"),
```

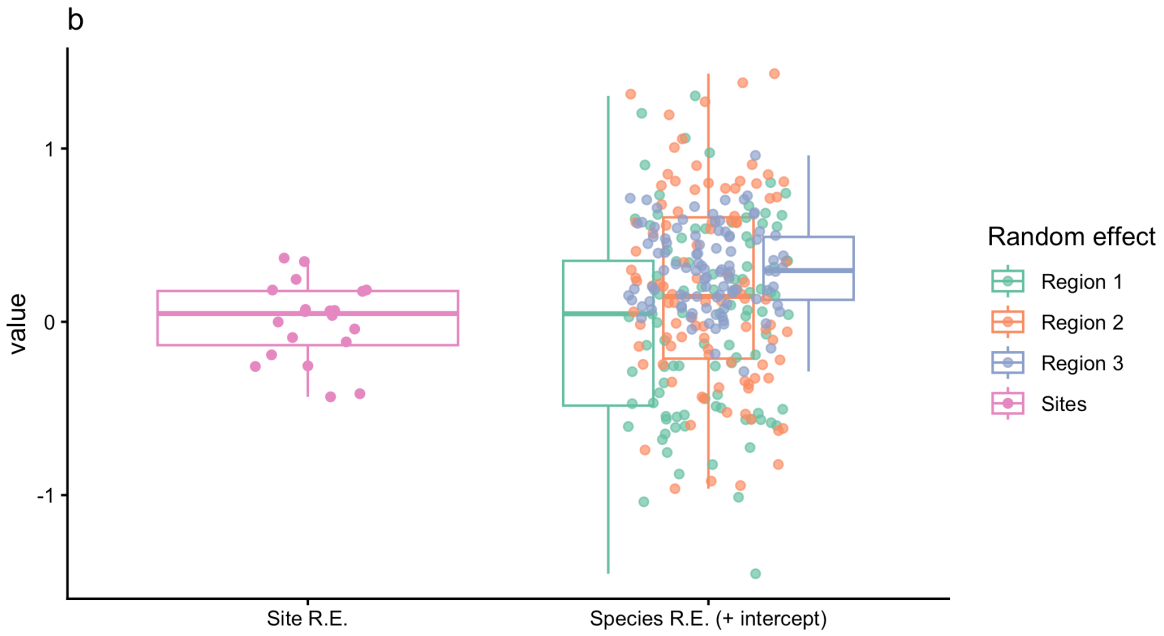
```
                    type="qual", palette = "Set1") +
  ylim(-.45,.45)
```



Supplementary Figure 2. 17: One-dimenstion ordination plot of the constrained ordination. Equivalent to Figure 5a in the main manuscript. Species loading and environemntal predictor labels were manually adjusted befor inclusion in the main manuscpript

```
# extract site and random effects
site_effects <- we_2_cn_mod$params$row.params.random
region_effects <- we_2_cn_mod$params$B
region_species_effects <- as.data.frame(we_2_cn_mod$params$Br)

# add intercepts to species random effects
region_species_effects[2,] <- region_species_effects[2,] + region_effects[1]
region_species_effects[3,] <- region_species_effects[3,] + region_effects[2]
region_species_effects$region <- rownames(region_species_effects)

# pivot for ggplot
region_species_effects <- region_species_effects |>
  pivot_longer(cols=1:98)
```

```
# plot
ggplot() +
  geom_boxplot(aes(x="Site R.E.", y=site_effects, color="sites")) +
  geom_jitter(aes(x="Site R.E.", y=site_effects, color="sites"),
              position=position_jitter(0.15)) +
  geom_boxplot(aes(x="Species R.E. (+ intercept)",
                   y=region_species_effects$value,
                   color=region_species_effects$region),
               outlier.shape = NA) +
  geom_jitter(aes(x="Species R.E. (+ intercept)",
                  y=region_species_effects$value,
                  color=region_species_effects$region),
              alpha=0.7, width=0.2) +
  labs(color="Random effect",x="", y="value", title="b") +
  theme_classic() +
  scale_color_brewer(palette="Set2",
                     labels = c("Region 1", "Region 2", "Region 3", "Sites"))
```



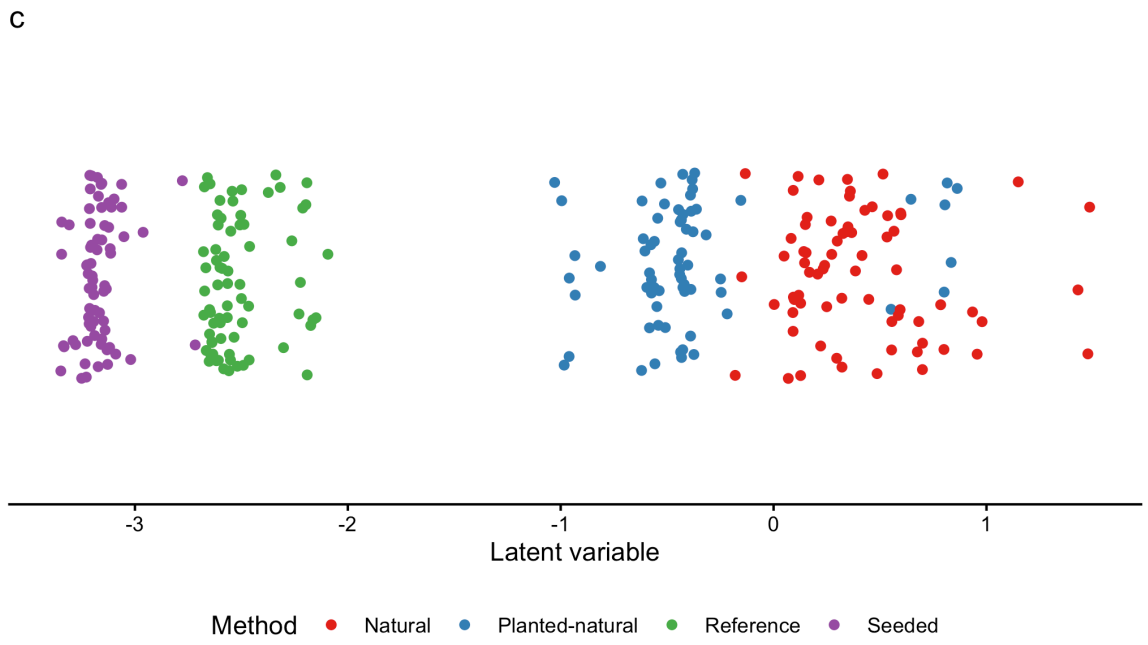
Supplementary Figure 2. 18: Boxplot of the site-specific and species-specific random effects of the sites and different study regions in Southern Norway, respectively. For the species-region random effects, the combined effect of the fixed-effect intercepts of region 2 and 3 and the random species effect are shown. Equivalent to Figure 5a in the main manuscript.

Below is the code for predicting the site scores after 20 years:

```
env_pred <- X
add_t <- 20/(sqrt(var(env$years_since_n[env$years_since_n<40])))
env_pred$natXtime <- ifelse(X$natXtime>0, X$natXtime+add_t, 0)
env_pred$pnXtime <- ifelse(X$pnXtime>0, X$pnXtime+add_t, 0)
env_pred$seedXtime <- ifelse(X$seedXtime>0, X$seedXtime+add_t, 0)
env_pred$gf <- env$gf # needed for prediction
```

```
lv_scores <- scaled_cw_sites
pars <- coef_scaled

lv_scores[X$method=="nat",] = lv_scores[X$method=="nat",] + pars[4]*add_t
lv_scores[X$method=="pn",] = lv_scores[X$method=="pn",] + pars[5]*add_t
lv_scores[X$method=="seed",] = lv_scores[X$method=="seed",] + pars[6]*add_t
```

```
# predict LV scores plot
ggplot() +
  geom_point(aes(x=lv_scores[,1], y=0, color=X$method),
             position = position_jitter(w = 0, h = 0.1)) +
  theme_classic() +
  theme(axis.line.y = element_blank(),
        axis.ticks.y = element_blank(),
        axis.text.y = element_blank(),
        axis.line.x = element_line(),
        legend.position = "bottom") +
  labs(y="", x="Latent variable", color="Method", title = "c") +
  scale_color_brewer(labels = c("Natural", "Planted-natural",
                                "Reference","Seeded"),
                     type="qual", palette = "Set1") +
  ylim(-.2,.2)
```



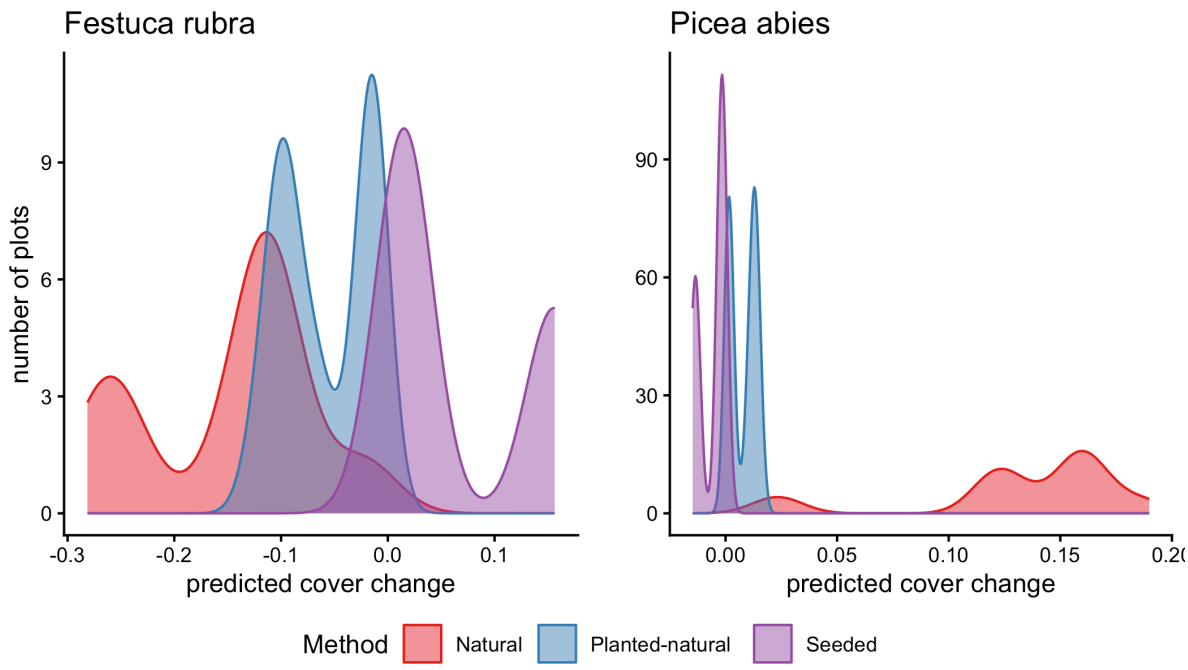
Supplementary Figure 2. 19: One-dimenstion ordination plot of the constrained ordination, with 20 years added to the time predictor. Equivalent to Figure 5c in the main manuscript.

We can also use the model to predict the future cover of specific species in the plots, using

the code below, and visualising he future predictions for two potential indicator species Sup. Figure 2. 20.

```
X$gf <- env_pred$gf # needed for prediction of baseline
pred_scores_0 <- predict.gllvm(newX=X, object = we_2_cn_mod,
                               type = "response")
pred_scores_20 <- predict.gllvm(newX=env_pred, object = we_2_cn_mod,
                                type = "response")
diffs <- as.data.frame(pred_scores_20-pred_scores_0) # calculate diff
diffs$method <- env_pred$method
diffs <- diffs |> filter(method != "ref") # remove reference sites from the data frame
```

```
p1 <- ggplot() +
  geom_density(aes(x=diffs$festuca_rubra, fill = diffs$method,
                   color=diffs$method), alpha=0.5) +
  labs(x="predicted cover change", y="number of plots",
       title = "Festuca rubra", fill="Method", color="Method") +
  scale_fill_manual(labels = c("Natural", "Planted-natural", "Seeded"),
                    values = c("#E41A1C", "#377EB8", "#984EA3")) +
  scale_color_manual(labels=c("Natural", "Planted-natural","Seeded"),
                     values = c("#E41A1C", "#377EB8", "#984EA3")) +
  theme_classic()

p2 <- ggplot() +
  geom_density(aes(x=diffs$picea_abies, fill = diffs$method,
                   color=diffs$method), alpha=0.5) +
  labs(x="predicted cover change", y="", title = "Picea abies",
       fill="Method", color="Method") +
  scale_fill_manual(labels = c("Natural", "Planted-natural", "Seeded"),
                    values = c("#E41A1C", "#377EB8", "#984EA3")) +
  scale_color_manual(labels=c("Natural", "Planted-natural", "Seeded"),
                     values = c("#E41A1C", "#377EB8", "#984EA3")) +
  theme_classic()

ggarrange(p1,p2, common.legend = T, legend = "bottom")
```



Supplementary Figure 2. 20: One-dimenstion ordination plot of the constrained ordination. Equivalent to Figure 5d in the main manuscript.

We can also look at some additional aspects of the model, such as the goodness-of-fit with the original data,

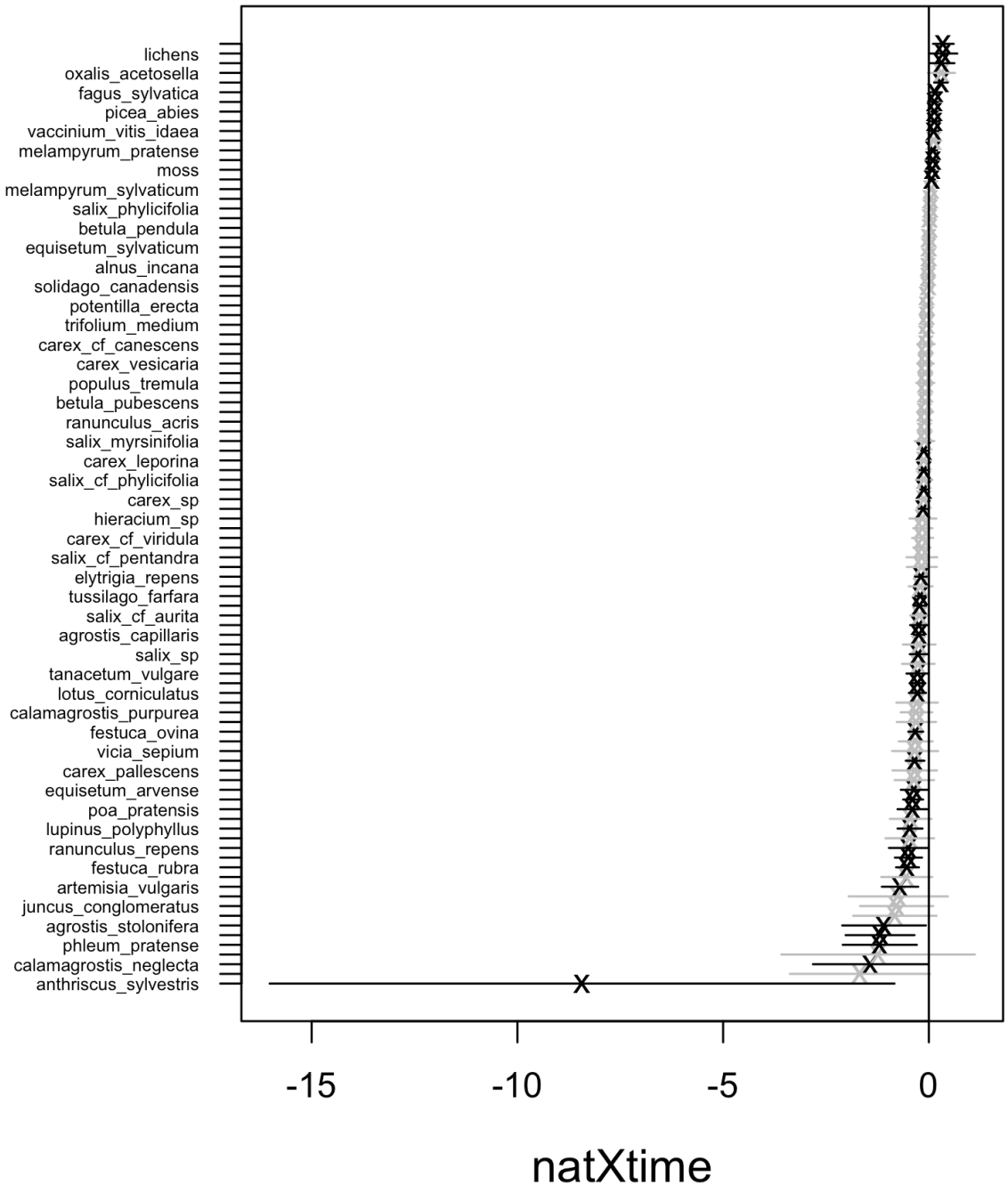
```
goodnessOfFit(we_2_cn_mod)
```

```
$cor
[1] 0.3797742

$RMSE
[1] 0.09896896

$MAE
[1] 0.06808131

$MARNE
[1] 0.4496327
```

where we see that the residual errors are quite low, although it does not correlate as strongly as the model in Worked example 1.

Like in Worked example 1, we can also look at the species-specific coefficients for the environmental predictors: as an example, for the natural re-vegetation treatment over time Sup. Figure 2. 21, where we see that *Anthriscus sylvestris* has by far the largest coefficient.

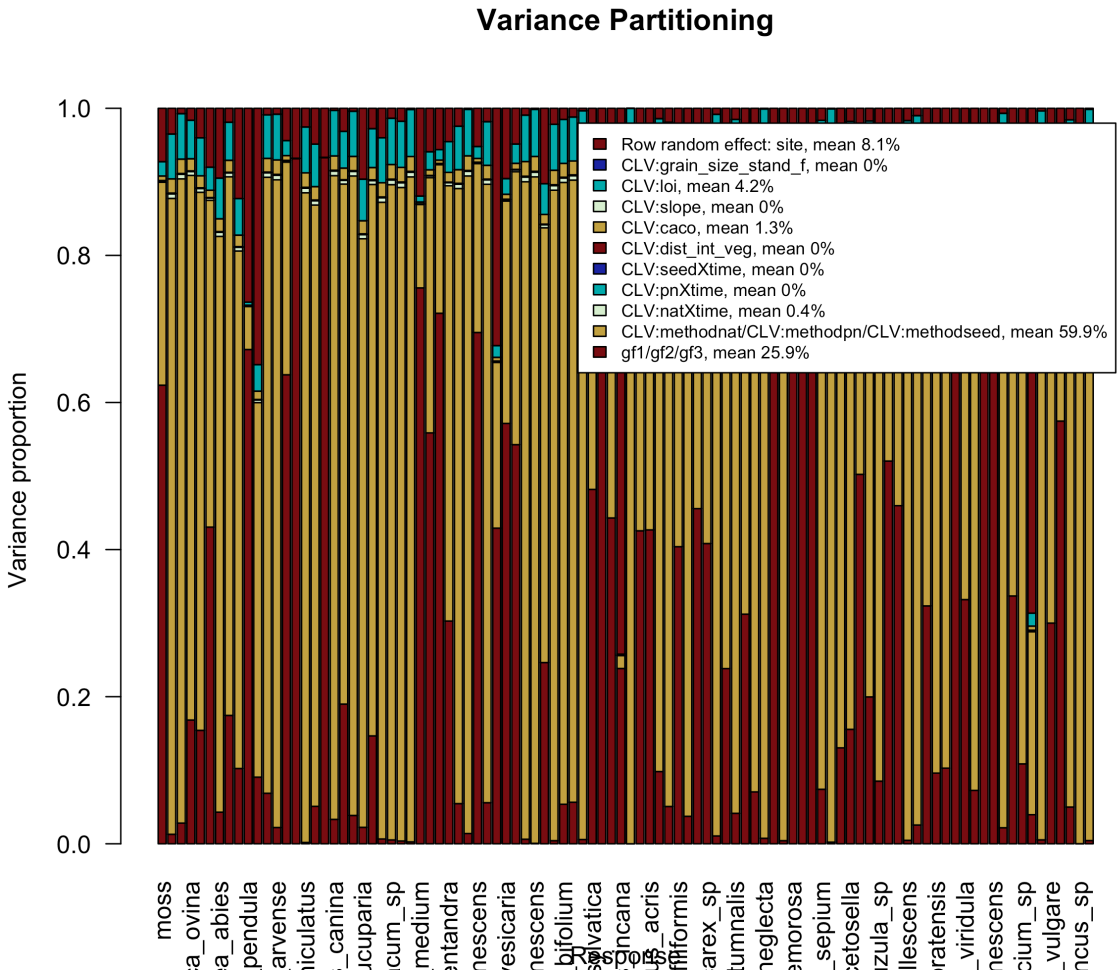
```
coefplot(we_2_cn_mod, which.Xcoef = 4)
```



Supplementary Figure 2. 21: Variance partitiining of the latent variable model in worked example 2

We can also visualize the variance partitioning Sup. Figure 2. 22, and note that the restoration treatment explain by far the most variance, followed by the species-specific effect of the study site and the community-wide random effect.

```
vp_we_2_cn_mod <- varPartitioning.gllvm(we_2_cn_mod)
plot(vp_we_2_cn_mod, args.legend = list(cex=0.7),
     col=hcl.colors(5, "Roma"), las=2)
```



Supplementary Figure 2. 22: Variance partitiining of the latent variable model in worked example 2

References

Fernandez, Romina D., Pilar Castro-Díez, Roxana Aragón, and Natalia Pérez-Harguindeguy. 2021. "Changes in Community Functional Structure and Ecosystem Properties Along an Invasion Gradient of Ligustrum Lucidum." *Journal of Vegetation Science* 32 (6): e13098. https://doi.org/10.1111/jvs.13098.

Mehlhoop, Anne Catriona, Astrid Brekke Skrindo, Marianne Evju, and Dagmar Hagen. 2022. "Best Practice—Is Natural Revegetation Sufficient to Achieve Mitigation Goals in Road Construction?" *Applied Vegetation Science* 25 (3): e12673. https://doi.org/10.1111/avsc.12673.