Using hierarchical joint models to detect reproductive interactions in plant communities: Plot-level visitation data

Øystein H. Opedal & Stein Joar Hegland

9 May 2019

Contact: oystein.opedal@helsinki.fi

Introduction

Here, we demonstrate how to set up a HMSC analysis of data on bumblebee visitation to 9 plant species cooccurring on 20 plots, collected during 176 10-min censuses. Data are from Hegland et al. 2009 (Ecological Research). A summary of these results are reported in Opedal & Hegland 2019 (in prep.).

For technical details about the HMSC model, see Ovaskainen et al. 2017 (Ecology Letters) and Tikhonov et al. 2019 (bioRxiv).

We will fit four different models, which will allow us to assess different aspects of potential reproductive interactions.

Model 1: Latent variables and environmental covariates only

We start by fitting a model with temperature as a single environmental covariate, which allows assessing raw associations among species after controlling for effects of temperature on insect activity.

In the HMSC model, we include in the response matrix ($\mathbf{Y}$) the number of pollinator visits to each species, with `NA` for species not flowering in the focal plot during a census. The number of pollinator visits are $\log(x+1)$-transformed to place the response variables on a proportional scale and to reduce the leverage of large values.

Model setup and MCMC sampling

Load packages and read data files

```
library(Hmsc)
library(corrplot)

Y = read.csv("model1/Y1.csv")
XData = read.csv("model1/XData1.csv")
studyDesign = read.csv("model1/studyDesign1.csv")
studyDesign$plot = as.factor(studyDesign$plot)
```

Define HMSC random levels

HMSC handles an arbitrary number of random factors represented by latent variables. Here, we set two random levels, `rL1` for the plots, and `rL2` for the sampling units (censuses).

```
head(studyDesign)
```

```
##   plot               su
## 1   17 17_11.06.2003_17
## 2    3  3_15.06.2003_13
## 3   20 20_15.06.2003_12
## 4    7   7_15.06.2003_9
## 5   16 16_18.06.2003_12
## 6    5  5_18.06.2003_11
```

```
rL1 = HmscRandomLevel(units = unique(studyDesign[,1]))
rL2 = HmscRandomLevel(units = unique(studyDesign[,2]))
```

Set model formula for covariates

The regression formula for the covariates is specified using standard `R` formula syntax, and can include linear terms, polynomials, and interactions.

```
XFormula = ~ poly(Temp, degree=2, raw=TRUE)
```

Set up the HMSC model

In this step we also set the error distribution of the model, here Gaussian (`distr="normal"`)

```
m = Hmsc(Y = as.matrix(log(Y+1)), XData = XData,  XFormula = XFormula,
         distr="normal", studyDesign=studyDesign, ranLevels = list(plot=rL1, su=rL2))
```

Run MCMC and save the model object

As with all MCMC-based Bayesian analyses, the posterior needs to be sampled until convergence. A good strategy is to start with a small number of iterations, and then increase the number of iterations until the results converge. We run 2 replicate MCMC chains to assess whether these converge. Because the sampling may take a long time, it is always advisable to save the model object including the posterior distribution to a local file.

```
thin = 200
samples = 1000
nChains = 2
adaptNf = ceiling(0.4*samples*thin)
transient = ceiling(0.5*samples*thin)

a = Sys.time()
m = sampleMcmc(m, samples = samples, thin = thin,
               adaptNf = rep(adaptNf, m$nr),
               transient = transient,
               nChains = nChains, nParallel = 1)
Sys.time() - a

save(m, file = "model1/mod1_thin200_samples1000_chains2.RData")
```

Evaluating chain convergence and model fit

After running the MCMC sampling scheme, we need to evaluate whether the chains converged, so that the results can be trusted.

Load the model object

```
load("model1/mod1_thin200_samples1000_chains2.RData")
```

Compute effective sample sizes and potential scale reduction factors

The effective sample sizes of the beta (regression coefficients) and omega (residual covariances) parameters are (in most cases) close to the expected (2000 samples), indicating adequate chain mixing. The potential scale reduction factors are all <1.1, indicating convergence of the two independent chains.

```
post = convertToCodaObject(m)

esBeta = effectiveSize(post$Beta)
summary(esBeta)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1752    1918    2000    1977    2000    2162
```

```
psrfBeta = gelman.diag(post$Beta)
summary(psrfBeta$psrf)
```

```
##    Point est.       Upper C.I.
##  Min.   :0.9999   Min.   :0.9999
##  1st Qu.:1.0004   1st Qu.:1.0016
##  Median :1.0007   Median :1.0032
##  Mean   :1.0011   Mean   :1.0057
##  3rd Qu.:1.0012   3rd Qu.:1.0058
##  Max.   :1.0058   Max.   :1.0306
```

```
esOmega = effectiveSize(post$Omega[[1]])
summary(esOmega)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1700    1913    2000    2012    2096    2391
```

Write posterior trace plots to pdf

Because the HMSC model includes many parameters, it is advisable to write the posterior trace plots to a pdf file rather then plotting them directly within R.

```
pdf("model1/posterior_plots/BetaPost.pdf")
plot(post$Beta)
dev.off()

pdf("model1/posterior_plots/OmegaPost.pdf")
plot(post$Omega[[1]])
plot(post$Omega[[2]])
dev.off()
```

Extract and assess parameter estimates

Compute predicted values

```
predY = computePredictedValues(m)
```

Evaluate model fit

We can evaluate the explanatory power of the model by computing r^2 values for each species.

```
MF = evaluateModelFit(m, predY)
round(MF$R2, 2)
```

```
## [1] 0.44 0.29 0.25 0.17 0.29 0.25 0.26 0.18 0.26
```

```
round(mean(MF$R2), 2)
```

```
## [1] 0.27
```

Compute and plot variance partitioning

HMSC comes with tools for performing variance component analyses, i.e. partitioning the explained variance into components related to each random effect and each (group of) fixed effect(s).

```
groups = c(1,1,1)
groupnames = "Temp"
VP = computeVariancePartitioning(m, groups, groupnames)

names = gsub("_", " ", colnames(m$Y))

outvals = VP$vals
ng = dim(outvals)[1]
leg = groupnames
for (r in 1:m$nr){leg = c(leg, paste("Random: ", m$rLNames[r], sep = ""))}
means = round(100 * rowMeans(outvals), 1)
for (i in 1:ng){leg[i] = paste(leg[i], " (mean = ", toString(means[i]), ")", sep = "")}

par(mar = c(8,4,2,11), xpd=T)
barplot(outvals, xlab = "", ylab = "Variance proportion",axisnames=F,
        args.legend=list(x=17.5,y=1, bty="n", cex=.8),
        las = 1, legend = leg, col = topo.colors(ng, alpha = 1))
text(x = seq(.5,10, length.out = 9), par("usr")[3] - 0.05, srt = 45, adj = .9, cex = .8,
     labels = names, xpd = TRUE)
```

Extract beta parameters

The regression coefficients describing the effects of the covariates on each response variable can be extracted using the `getPostEstimate` function. The support values corresponds to the proportion of posterior estimates that are above zero, with values close to 1 indicating strong support for a positive effect. The supportNeg values corresponds to the proportion of posterior estimates that are below zero, with values close to 1 indicating strong support for a negative effect.

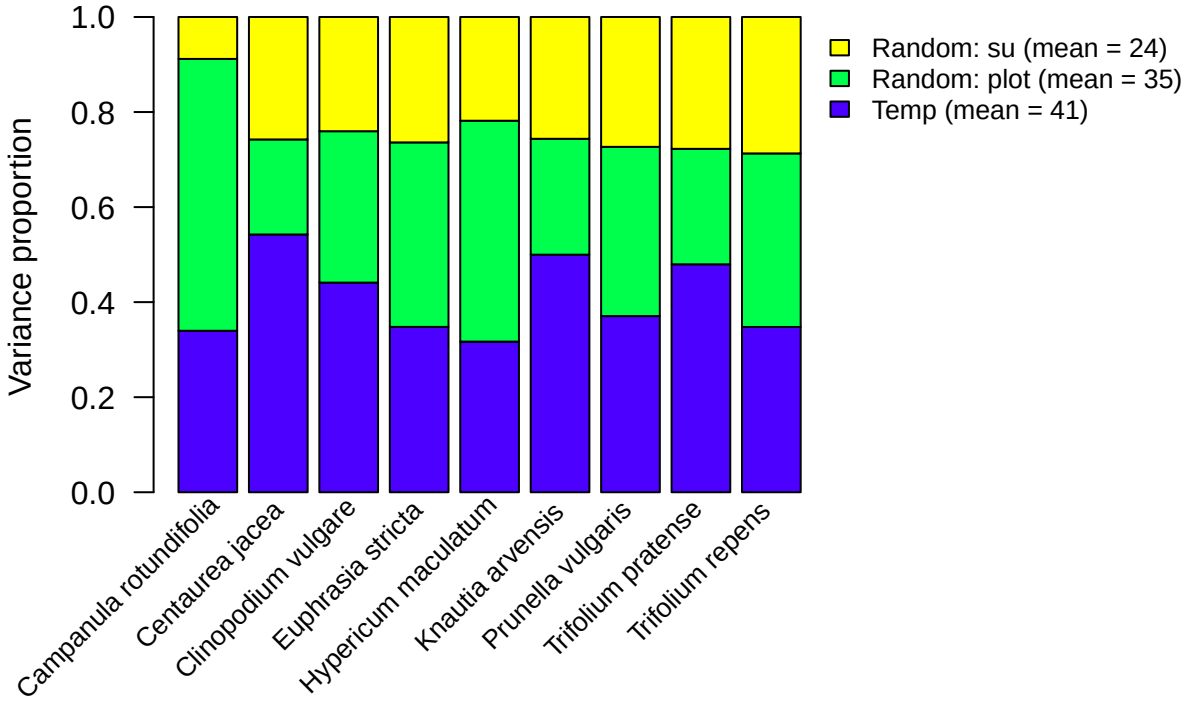


Figure 1: Variance partitioning for each species for Model 1

```
pBeta = getPostEstimate(m, "Beta")
pBeta
```

```
## $mean
##      Campanula_rotundifolia Centaurea_jacea Clinopodium_vulgare
## [1,]           -2.573626758    -0.544688926         -1.94491814
## [2,]            0.168350457     0.117491242          0.15360813
## [3,]           -0.002065701    -0.002270569         -0.00211104
##      Euphrasia_stricta Hypericum_maculatum Knautia_arvensis
## [1,]      -1.595791505        -1.677607659     -1.252126339
## [2,]       0.123041021         0.151947043      0.127106179
## [3,]      -0.001721741        -0.002362996     -0.002180371
##      Prunella_vulgaris Trifolium_pratense Trifolium_repens
## [1,]      -1.764549401       -1.037755465     -1.522772195
## [2,]       0.146559633        0.102471461      0.132416830
## [3,]      -0.001976021       -0.001158058     -0.002271367
##
## $support
##      Campanula_rotundifolia Centaurea_jacea Clinopodium_vulgare
## [1,]                 0.0300          0.3575               0.081
## [2,]                 0.9645          0.8830               0.950
## [3,]                 0.1030          0.1090               0.086
##      Euphrasia_stricta Hypericum_maculatum Knautia_arvensis
## [1,]            0.1125              0.1245           0.2255
## [2,]            0.9195              0.9475           0.8905
## [3,]            0.1305              0.0835           0.1100
##      Prunella_vulgaris Trifolium_pratense Trifolium_repens
## [1,]            0.1195             0.2055           0.1410
## [2,]            0.9340             0.8755           0.9210
## [3,]            0.1065             0.2105           0.0735
##
## $supportNeg
##      Campanula_rotundifolia Centaurea_jacea Clinopodium_vulgare
## [1,]                 0.9700          0.6425               0.919
## [2,]                 0.0355          0.1170               0.050
## [3,]                 0.8970          0.8910               0.914
##      Euphrasia_stricta Hypericum_maculatum Knautia_arvensis
## [1,]            0.8875              0.8755           0.7745
## [2,]            0.0805              0.0525           0.1095
## [3,]            0.8695              0.9165           0.8900
##      Prunella_vulgaris Trifolium_pratense Trifolium_repens
## [1,]            0.8805             0.7945           0.8590
## [2,]            0.0660             0.1245           0.0790
## [3,]            0.8935             0.7895           0.9265
```

Extract and plot the Omega matrix

The Omega matrices describing residual associations among response variables can be extracted using the `computeAssociations` function.

```
OmegaCor = computeAssociations(m)

par(mfrow = c(1,2))
```

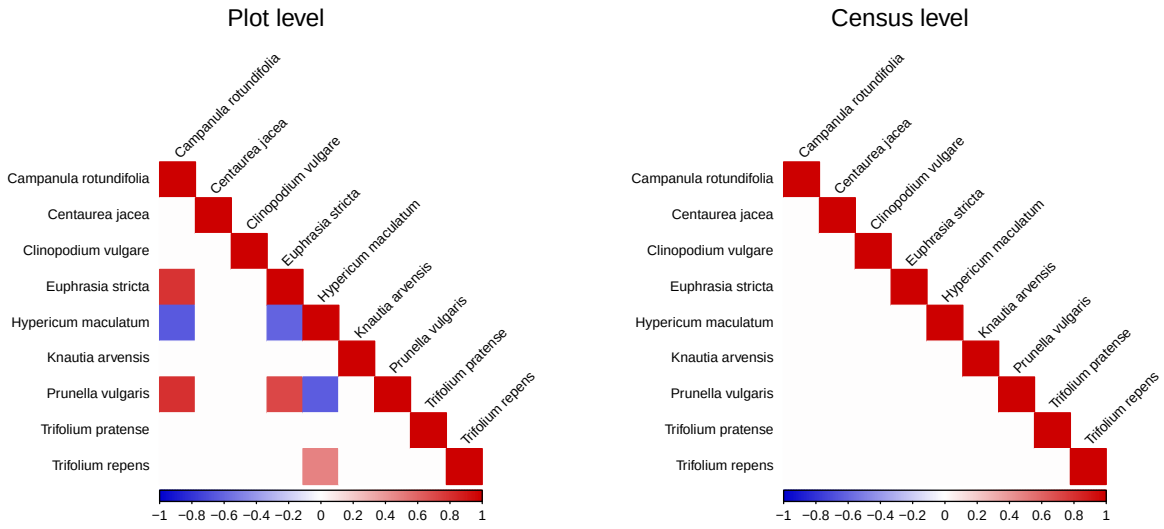


Figure 2: Residual associations for pollinator visitation to each species after accounting for the effect of temperature on visitation. Associations with at least 75% posterior support are shown

```
plotOrder = 1:m$ns
supportLevel = 0.75

toPlot = ((OmegaCor[[1]]$support>supportLevel) +
            (OmegaCor[[1]]$support<(1-supportLevel))>0)*OmegaCor[[1]]$mean
rownames(toPlot)=colnames(toPlot)=gsub("_"," ",rownames(toPlot))
corrplot(toPlot[plotOrder,plotOrder], type="lower", tl.cex=.7, tl.col="black",
         tl.srt=45, method = "color", col=colorRampPalette(c("blue3","white","red3"))(200),
         title=expression("Plot level"), cl.cex=.7, mar=c(0,0,1,0))

toPlot = ((OmegaCor[[2]]$support>supportLevel) +
            (OmegaCor[[2]]$support<(1-supportLevel))>0)*OmegaCor[[2]]$mean
rownames(toPlot)=colnames(toPlot)=gsub("_"," ",rownames(toPlot))
corrplot(toPlot[plotOrder,plotOrder], type="lower", tl.cex=.7, tl.col="black",
         tl.srt=45, method = "color", col=colorRampPalette(c("blue3","white","red3"))(200),
         title=expression("Census level"), cl.cex=.7, mar=c(0,0,1,0))
```

Construct and plot a gradient for temperature

HMSC-R also comes with tools for constructing gradients illustrating the response of a response variable to covariates. Here, we construct a gradient for temperature, and plot the results for the complete community (the sum of the log number of visits to each species).

```
Gradient = constructGradient(m, focalVariable = "Temp")

predY = predict(m, Gradient=Gradient,expected=TRUE, predictEtaMean=FALSE)

plotGradient(m, Gradient, predY, measure ="S", showData=F, las=1)
```

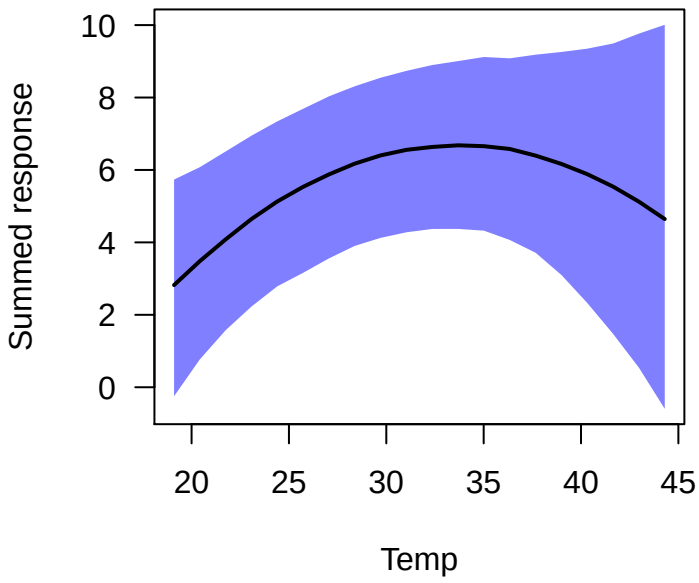


Figure 3: Effect of temperature on the total number of pollinator visits to all species (sum of log visits)

Model 2: Conspecific flower counts as covariates

Pollinator visitation is likely to depend on the phenotype of the focal species. Here, we assess how the number of flowers in a plot during a census influences the overall number of visits to those flowers. To do so, we now add to the model the number of flowers of the focal species in each plot during each census as a species-specific covariate.

Model setup and MCMC sampling

Read data files

```
Y = read.csv("model2/Y2.csv")
XData = read.csv("model2/XData2.csv")
studyDesign = read.csv("model2/studyDesign2.csv")
studyDesign$plot = as.factor(studyDesign$plot)
```

Define HMSC random levels

```
rL1 = HmscRandomLevel(units = unique(studyDesign[,1]))
rL2 = HmscRandomLevel(units = unique(studyDesign[,2]))
```

Compile a list containing XData for each species

In this model the values of the covariate `nflowers` (log number of flowers) are unique for each species. Therefore, instead of providing a single data frame, we provide a list of data frames containing the covariates for each species.

```
head(XData)
```

```
##   Plot               su Temp Campanula_rotundifolia Centaurea_jacea
## 1   17 17_11.06.2003_17 21.8                      0              0
## 2    3  3_15.06.2003_13 22.6                      0              0
## 3   20 20_15.06.2003_12 19.7                      0              0
## 4    7   7_15.06.2003_9 23.5                      0              0
## 5   16 16_18.06.2003_12 28.1                      0              0
## 6    5  5_18.06.2003_11 27.6                      0              0
##   Clinopodium_vulgare Euphrasia_stricta Hypericum_maculatum
## 1                   0                 0                   0
## 2                   0                 0                   0
## 3                   0                 0                   0
## 4                   0                 0                   0
## 5                   0                 0                   0
## 6                   0                 0                   0
##   Knautia_arvensis Prunella_vulgaris Trifolium_pratense Trifolium_repens
## 1                0                 0          1.0986123        0.0000000
## 2                0                 0          1.0986123        0.6931472
## 3                0                 0          1.3862944        0.0000000
## 4                0                 0          0.6931472        0.0000000
## 5                0                 0          0.6931472        0.0000000
## 6                0                 0          0.0000000        0.6931472
```

```
xList = list()
for(i in 1:ncol(Y)){
  xList[[i]] = data.frame(Temp=XData$Temp, nflowers=c(XData[i+3]))
  names(xList[[i]]) = c("Temp","nflowers")
}

head(xList[[8]],5)
```

```
##   Temp  nflowers
## 1 21.8 1.0986123
## 2 22.6 1.0986123
## 3 19.7 1.3862944
## 4 23.5 0.6931472
## 5 28.1 0.6931472
```

Set model formula for covariates

```
XFormula = ~ nflowers + poly(Temp, degree=2, raw=TRUE)
```

Set up the HMSC model

```
m = Hmsc(Y = as.matrix(log(Y+1)), XData = xList,  XFormula = XFormula,
         distr="normal", studyDesign = studyDesign, ranLevels = list(plot=rL1,su=rL2))
```

Run MCMC and save the model object

```
thin = 200
samples = 1000
nChains = 2
adaptNf = ceiling(0.4*samples*thin)
transient = ceiling(0.5*samples*thin)

a = Sys.time()
m = sampleMcmc(m, samples = samples, thin = thin,
               adaptNf = rep(adaptNf, m$nr),
               transient = transient,
               nChains = nChains, nParallel = 1)
Sys.time() - a

save(m, file ="model2/mod2_thin200_samples1000_chains2.RData")
```

Evaluating chain convergence and model fit

Load the model object

```
load("model2/mod2_thin200_samples1000_chains2.RData")
```

Compute effective sample sizes and potential scale reduction factors

```
post = convertToCodaObject(m)

esBeta = effectiveSize(post$Beta)
summary(esBeta)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1736    1996    2000    2018    2004    3030
```

```
psrfBeta = gelman.diag(post$Beta)
summary(psrfBeta$psrf)
```

```
##    Point est.       Upper C.I.
##  Min.   :0.9994   Min.   :0.9994
##  1st Qu.:1.0000   1st Qu.:1.0008
##  Median :1.0004   Median :1.0020
##  Mean   :1.0012   Mean   :1.0043
##  3rd Qu.:1.0020   3rd Qu.:1.0059
##  Max.   :1.0076   Max.   :1.0316
```

```
esOmega = effectiveSize(post$Omega[[1]])
summary(esOmega)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1829    2000    2000    2058    2104    2843
```

Write posterior trace plots to pdf

```
pdf("model2/posterior_plots/BetaPost.pdf")
plot(post$Beta)
dev.off()

pdf("model2/posterior_plots/OmegaPost.pdf")
plot(post$Omega[[1]])
plot(post$Omega[[2]])
dev.off()
```

Extract and assess parameter estimates

Compute predicted values

```
predY = computePredictedValues(m)
```

Evaluate model fit

```
MF = evaluateModelFit(m, predY)
round(MF$R2, 2)
```

```
## [1] 0.46 0.32 0.44 0.26 0.33 0.24 0.34 0.36 0.27
```

```
round(mean(MF$R2), 2)
```

```
## [1] 0.34
```

Compute and plot variance partitioning

```
groups = c(1,1,2,2)
groupnames = c("Conspecific flowers", "Temperature")
VP = computeVariancePartitioning(m, groups, groupnames)

names = gsub("_", " ", colnames(m$Y))

outvals = VP$vals
ng = dim(outvals)[1]
leg = groupnames
for (r in 1:m$nr) {leg = c(leg, paste("Random: ", m$rLNames[r], sep = ""))}
means = round(100 * rowMeans(outvals), 1)
for (i in 1:ng) {leg[i] = paste(leg[i], " (mean = ", toString(means[i]), ")", sep = "")}

par(mar = c(8,4,2,12), xpd=T)
barplot(outvals, xlab = "", ylab = "Variance proportion",axisnames=F,
        args.legend=list(x=19.5,y=1, bty="n", cex=.8),
        las = 1, legend = leg, col = topo.colors(ng, alpha = 1))
text(x = seq(.5,10, length.out = 9), par("usr")[3] - 0.05, srt = 45, adj = .9, cex = .8,
     labels = names, xpd = TRUE)
```

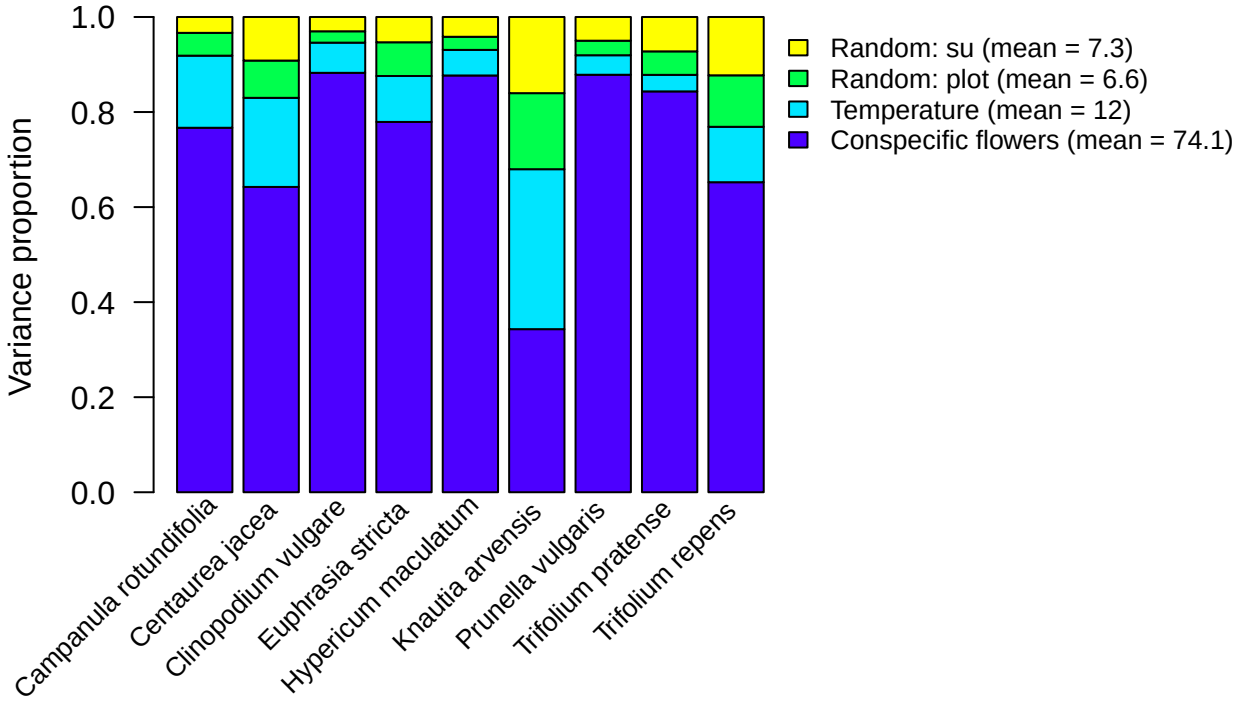


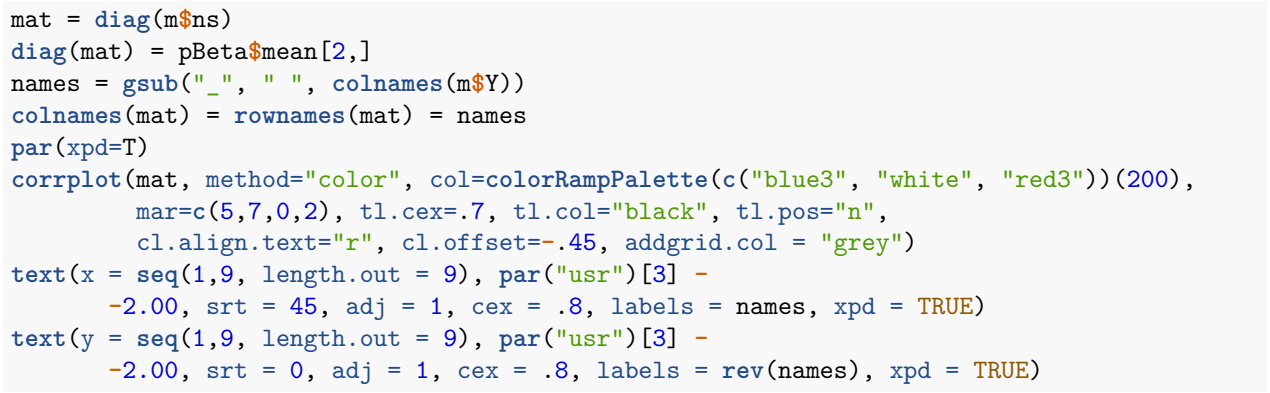
Figure 4: Variance partitioning for each species for Model 2

Extract and plot beta parameters

```
pBeta = getPostEstimate(m, "Beta")
pBeta$mean
```

```
##      Campanula_rotundifolia Centaurea_jacea Clinopodium_vulgare
## [1,]           -3.633437075    -1.808526548        -3.349894377
## [2,]            0.518408181     0.739896829         0.587100790
## [3,]            0.177010937     0.133277872         0.159311565
## [4,]           -0.002200119    -0.002324611        -0.002162262
##      Euphrasia_stricta Hypericum_maculatum Knautia_arvensis
## [1,]      -2.647962617        -2.720143661     -1.688232930
## [2,]       0.273259256         0.537360257      0.542787606
## [3,]       0.124818893         0.152024848      0.130968958
## [4,]      -0.001602942        -0.002335278     -0.002405238
##      Prunella_vulgaris Trifolium_pratense Trifolium_repens
## [1,]      -2.469160601       -1.858824881     -1.914858511
## [2,]       0.569543692        0.465905059      0.295652921
## [3,]       0.145244544        0.099551952      0.120329515
## [4,]      -0.002364748       -0.001465622     -0.002081686
```

```
mat = diag(m$ns)
diag(mat) = pBeta$mean[2,]
names = gsub("_", " ", colnames(m$Y))
colnames(mat) = rownames(mat) = names
par(xpd=T)
corrplot(mat, method="color", col=colorRampPalette(c("blue3", "white", "red3"))(200),
         mar=c(5,7,0,2), tl.cex=.7, tl.col="black", tl.pos="n",
         cl.align.text="r", cl.offset=-.45, addgrid.col = "grey")
text(x = seq(1,9, length.out = 9), par("usr")[3] -
       -2.00, srt = 45, adj = 1, cex = .8, labels = names, xpd = TRUE)
text(y = seq(1,9, length.out = 9), par("usr")[3] -
       -2.00, srt = 0, adj = 1, cex = .8, labels = rev(names), xpd = TRUE)
```



Credibility intervals for the beta parameters

```
post = convertToCodaObject(m, spNamesNumbers=c(F,T),covNamesNumbers=c(T,F))
summary(post$Beta)$quantiles[seq(2,34,4),]
```

```
##                           2.5%       25%       50%       75%     97.5%
## B[nflowers, (S1)]  0.34735783 0.4630861 0.5179493 0.5760077 0.6874612
## B[nflowers, (S2)]  0.28483437 0.5523727 0.7343571 0.9139686 1.2539406
## B[nflowers, (S3)]  0.40012778 0.5222846 0.5872142 0.6489166 0.7810989
## B[nflowers, (S4)]  0.16329602 0.2311310 0.2724727 0.3134225 0.3968774
## B[nflowers, (S5)]  0.33755314 0.4717873 0.5423461 0.6013974 0.7223218
## B[nflowers, (S6)] -0.02670231 0.3361168 0.5333113 0.7440691 1.1494186
## B[nflowers, (S7)]  0.36631085 0.4992351 0.5709772 0.6403006 0.7714587
## B[nflowers, (S8)]  0.34697994 0.4244774 0.4656549 0.5062328 0.5906866
## B[nflowers, (S9)]  0.10672268 0.2301845 0.2940219 0.3573870 0.4857707
```

Extract and plot the Omega matrices

```
OmegaCor = computeAssociations(m)
```

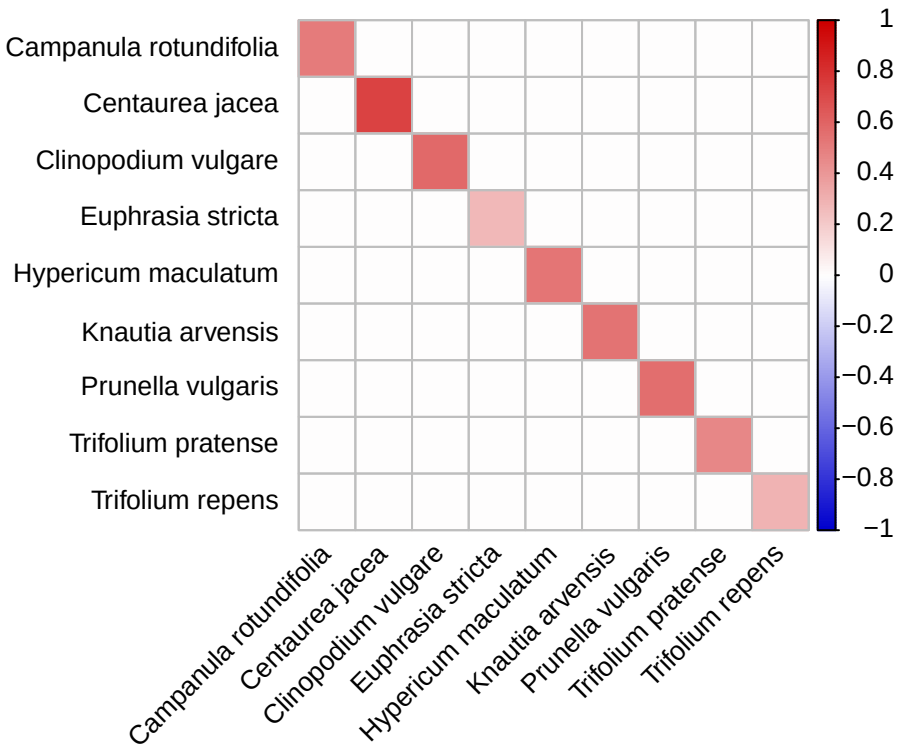


Figure 5: Effects of log(conspecific flower abundance) on log(number of visits)

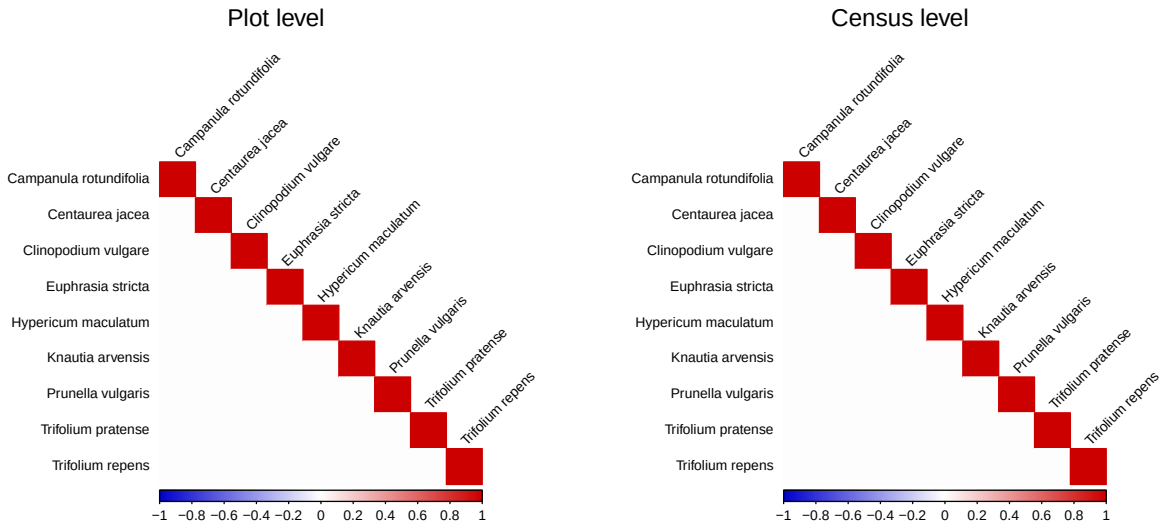


Figure 6: Residual associations for pollinator visitation to each species after accounting for the effect of temperature and conspecific flower abundances on visitation

```
par(mfrow = c(1,2))
plotOrder = 1:m$ns
supportLevel = 0.75

toPlot = ((OmegaCor[[1]]$support>supportLevel) +
            (OmegaCor[[1]]$support<(1-supportLevel))>0)*OmegaCor[[1]]$mean
rownames(toPlot)=colnames(toPlot)=gsub("_"," ",rownames(toPlot))
corrplot(toPlot[plotOrder,plotOrder], type="lower", tl.cex=.7, tl.col="black",
         tl.srt=45, method = "color", col=colorRampPalette(c("blue3","white","red3"))(200),
         title=expression("Plot level"), cl.cex=.7, mar=c(0,0,1,0))

toPlot = ((OmegaCor[[2]]$support>supportLevel) +
            (OmegaCor[[2]]$support<(1-supportLevel))>0)*OmegaCor[[2]]$mean
rownames(toPlot)=colnames(toPlot)=gsub("_"," ",rownames(toPlot))
corrplot(toPlot[plotOrder,plotOrder], type="lower", tl.cex=.7, tl.col="black",
         tl.srt=45, method = "color", col=colorRampPalette(c("blue3","white","red3"))(200),
         title=expression("Census level"), cl.cex=.7, mar=c(0,0,1,0))
```

Construct a gradient for conspecific flower number

```
Gradient = constructGradient(m, focalVariable = "nflowers",
                             non.focalVariables = list(
                             Temp = list(1),
                             nflowers = list(1)))

predY = predict(m, Gradient=Gradient, expected=TRUE, predictEtaMean=FALSE)

plotGradient(m, Gradient, predY, measure = "S", showData=F)
```

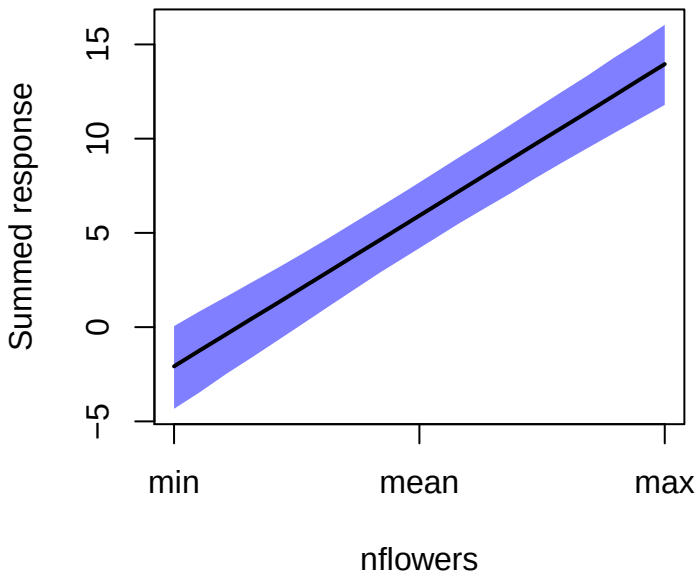


Figure 7: Effect of log(conspecific flower abundance) on pollinator visitation (sum log visits)

```
par(mfrow=c(3,3))
for(i in 1:m$ns){
  plotGradient(m, Gradient, predY, measure = "Y", index = i, showData = F)
}
```

Model 3: Conspecific + heterospecific flower counts as covariates

We will now ask whether and how the number of pollinator visits to a focal species depend on the flower abundances of coflowering species. We include the log $(x+1)$-transformed flower abundances of all species in each plot as covariates.

Model setup and MCMC sampling

Read data files

```
rm(list=ls())
Y = read.csv("model3/Y2.csv")
XData = read.csv("model3/XData2.csv")
studyDesign = read.csv("model3/studyDesign2.csv")
studyDesign$plot = as.factor(studyDesign$plot)
```

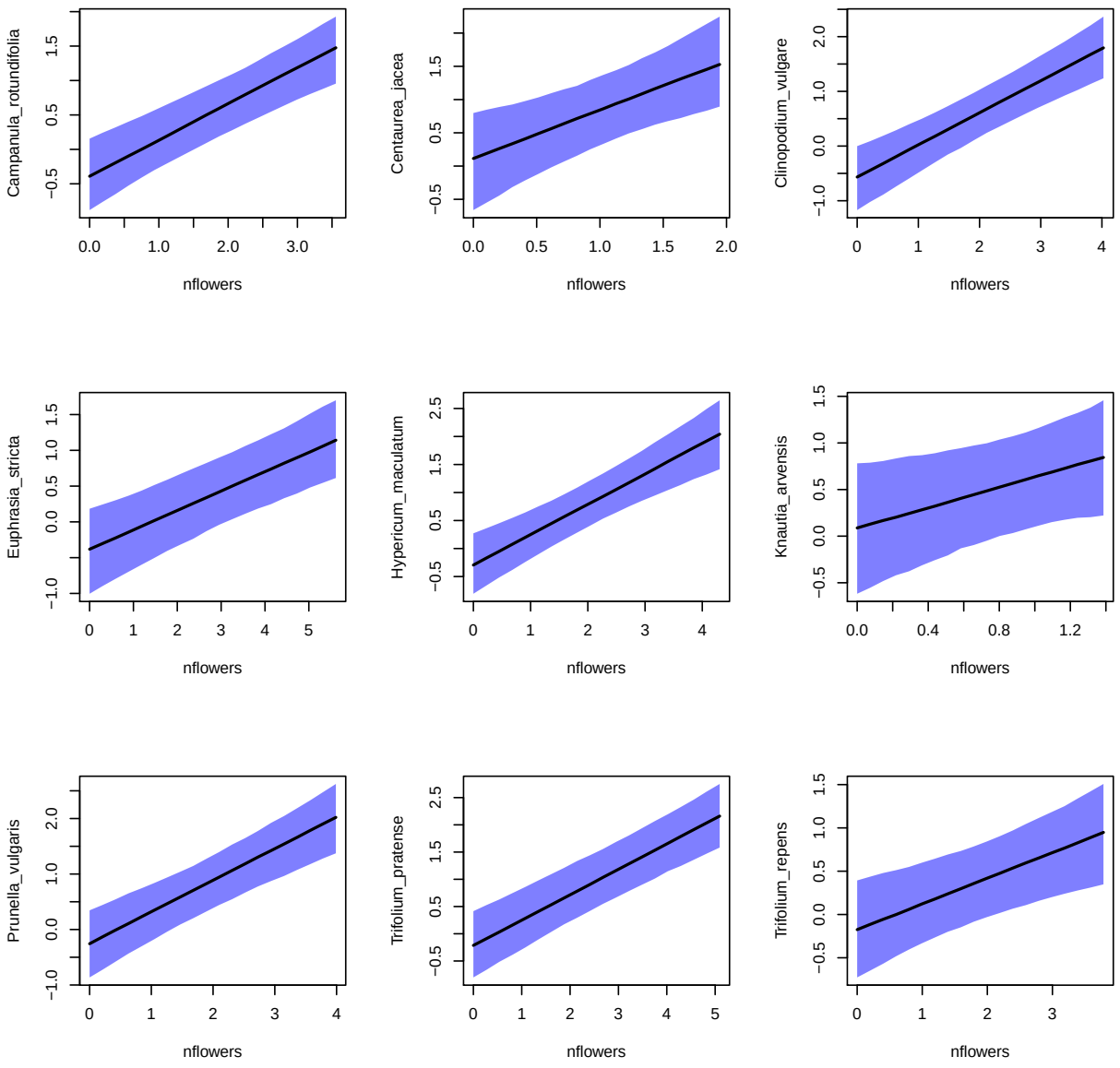


Figure 8: Effect of conspecific flower abundance on pollinator visitation to each species

Define HMSC random levels

```
rL1 = HmscRandomLevel(units = unique(studyDesign[,1]))
rL2 = HmscRandomLevel(units = unique(studyDesign[,2]))
```

Compile a list containing `XData` for each species

To keep the conspecific flower abundances separate from heterospecific flower abundances, we model the conspecific flower abundances by the covariate `nflowers`, and set the abundance of the focal species to 0 in the `xData` for each focal species. This format allows us (and the HMSC model) to consider similarity among species in the response to conspecific vs. heterospecific flower abundances.

```
xList = list()
for(i in 1:ncol(Y)){
  xList[[i]] = data.frame(data.frame(XData[,4:12]), Temp=XData$Temp, nflowers=c(XData[i+3]))
  xList[[i]][,i] = 0
  names(xList[[i]])[11] = "nflowers"
}
tail(xList[[8]],5)
```

```
##     Campanula_rotundifolia Centaurea_jacea Clinopodium_vulgare
## 172               0.000000       0.6931472            0.000000
## 173               1.098612       0.0000000            0.000000
## 174               0.000000       0.0000000            3.135494
## 175               1.098612       0.0000000            2.708050
## 176               0.000000       0.0000000            0.000000
##     Euphrasia_stricta Hypericum_maculatum Knautia_arvensis
## 172          4.753590            3.332205        0.0000000
## 173          2.944439            0.000000        0.0000000
## 174          5.153292            1.945910        0.6931472
## 175          5.030438            1.945910        0.0000000
## 176          4.060443            3.135494        0.0000000
##     Prunella_vulgaris Trifolium_pratense Trifolium_repens Temp  nflowers
## 172         0.0000000                  0                0 31.8 1.6094379
## 173         0.0000000                  0                0 32.6 0.6931472
## 174         0.6931472                  0                0 30.4 1.0986123
## 175         1.3862944                  0                0 24.8 1.0986123
## 176         0.0000000                  0                0 29.8 2.0794415
```

Set model formula for covariates

```
XFormula = as.formula(paste("~",paste(colnames(Y),collapse="+"),
                            "+nflowers + poly(Temp, degree=2, raw=TRUE)"))
XFormula
```

```
## ~Campanula_rotundifolia + Centaurea_jacea + Clinopodium_vulgare +
##     Euphrasia_stricta + Hypericum_maculatum + Knautia_arvensis +
##     Prunella_vulgaris + Trifolium_pratense + Trifolium_repens +
##     nflowers + poly(Temp, degree = 2, raw = TRUE)
```

Set up the HMSC model

```
m = Hmsc(Y=as.matrix(log(Y+1)), XData = xList,  XFormula = XFormula,
         distr="normal", studyDesign=studyDesign, ranLevels=list(plot=rL1,su=rL2))
```

Run MCMC and save the model object

```
thin = 200
samples = 1000
nChains = 2
adaptNf = ceiling(0.4*samples*thin)
transient = ceiling(0.5*samples*thin)

a=Sys.time()
m = sampleMcmc(m, samples = samples, thin = thin,
               adaptNf = rep(adaptNf,m$nr),
               transient = transient,
               nChains = nChains, nParallel = 1)
Sys.time() - a

save(m, file ="model3/mod3_thin200_samples1000_chains2.RData")
```

Evaluating chain convergence and model fit

Load the model object

```
load("model3/mod3_thin200_samples1000_chains2.RData")
```

Compute effective sample sizes and potential scale reduction factors

```
post = convertToCodaObject(m)

esBeta = effectiveSize(post$Beta)
summary(esBeta)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1683    2000    2000    2039    2103    2498
```

```
psrfBeta = gelman.diag(post$Beta)
summary(psrfBeta$psrf)
```

```
##    Point est.       Upper C.I.
##  Min.   :0.9993   Min.   :0.9994
##  1st Qu.:0.9997   1st Qu.:1.0001
##  Median :1.0003   Median :1.0021
##  Mean   :1.0009   Mean   :1.0045
##  3rd Qu.:1.0014   3rd Qu.:1.0064
##  Max.   :1.0086   Max.   :1.0400
```

```
esOmega = effectiveSize(post$Omega[[1]])
summary(esOmega)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1647    2000    2000    2048    2090    2756
```

Write posterior trace plots to pdf

```
pdf("model3/posterior_plots/BetaPost.pdf")
plot(post$Beta)
dev.off()

pdf("model3/posterior_plots/OmegaPost.pdf")
plot(post$Omega[[1]])
plot(post$Omega[[2]])
dev.off()
```

Extract and assess parameter estimates

Compute predicted values

```
predY = computePredictedValues(m)
```

Evaluate model fit

```
MF = evaluateModelFit(m, predY)
round(MF$R2, 2)
```

```
## [1] 0.49 0.52 0.57 0.32 0.37 0.52 0.39 0.40 0.29
```

```
round(mean(MF$R2), 2)
```

```
## [1] 0.43
```

Compute and plot variance partitioning

In this case we are interested in the contributions of conspecific vs. all heterospecific floral abundances on visitation to the focal species. We therefore estimate the variance explained by all non-focal species as a single group.

```
groups=c(rep(1,10), 2, 3, 3)
groupnames=c("Heterospecific flowers", "Conspecific flowers", "Temperature")
VP = computeVariancePartitioning(m, groups, groupnames)

outvals = VP$vals
ng = dim(outvals)[1]
leg = groupnames
m$rLNames = c("Plot", "Census")
for (r in 1:m$nr) {leg = c(leg, paste("Random: ", m$rLNames[r], sep = ""))}
means = round(100 * rowMeans(outvals), 1)
```

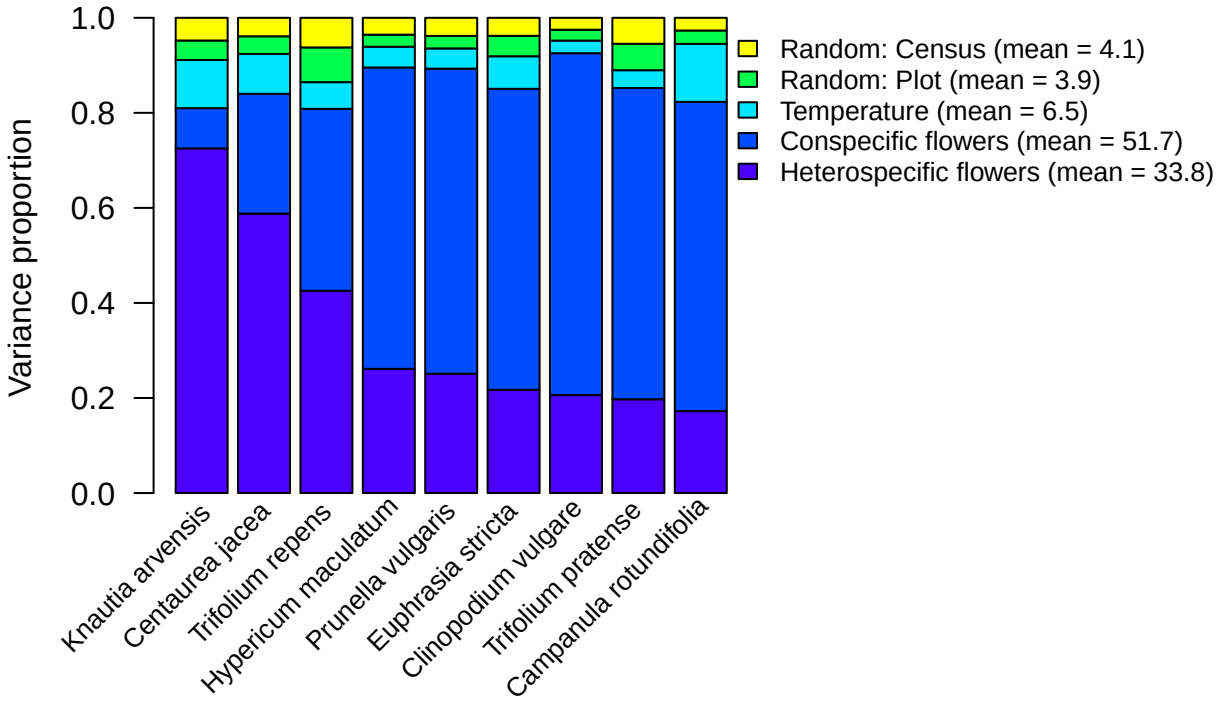


Figure 9: Variance partitioning for each species for Model 3

```
for (i in 1:ng) {leg[i] = paste(leg[i], " (mean = ", toString(means[i]), ")", sep = "")}

plotorder = order(outvals[1,],decreasing=T)
names= gsub("_", " ", colnames(m$Y))

par(mar = c(8,4,2,13), xpd=T)
barplot(outvals[,plotorder], xlab = "", ylab = "Variance proportion",
        args.legend=list(x=20.4,y=1, bty="n", cex=.8),
        axisnames=F,
        las = 1, legend = leg, col = topo.colors(ng, alpha = 1))
text(x = seq(.5,10, length.out = 9), par("usr")[3] - 0.05, srt = 45, adj = .9, cex = .8,
     labels = names[plotorder], xpd = TRUE)
```

Extract and plot beta parameters

The effects of conspecific and heterospecific flower abundances on pollinator visitation to each species are described by the beta coefficients. For a quick visual overview, we plot the posterior support of the beta parameters describing the effects of conspecific and heterospecific flower abundances. For visual clarity, we set those parameters with less than 85% posterior support to zero.

```
pBeta = getPostEstimate(m, "Beta")
```

```
mat = pBeta$mean[2:10,]
diag(mat) = pBeta$mean[11,]

smat = 2*pBeta$support[2:10,] - 1
diag(smat) = 2*pBeta$support[11,] - 1

supp = pBeta$support[2:10,]
diag(supp) = pBeta$support[11,]

supportLevel = .85
mat = smat * ((supp > supportLevel) + (supp < (1 - supportLevel)) > 0)

names = gsub("_", " ", colnames(m$Y))
colnames(mat) = rownames(mat) = names
par(xpd=T)
corrplot(t(mat), method="color", col=colorRampPalette(c("blue3", "white", "red3"))(200),
         mar=c(5,7,0,2), tl.cex=.7, tl.col="black", tl.pos="n",
         cl.align.text="r", cl.offset=-.45, addgrid.col = "grey")
text(x = seq(1,9, length.out = 9), par("usr")[3] -
       -2.00, srt = 45, adj = 1, cex = .8, labels = names, xpd = TRUE)
text(y = seq(1,9, length.out = 9), par("usr")[3] -
       -2.00, srt = 0, adj = 1, cex = .8, labels = rev(names), xpd = TRUE)
```

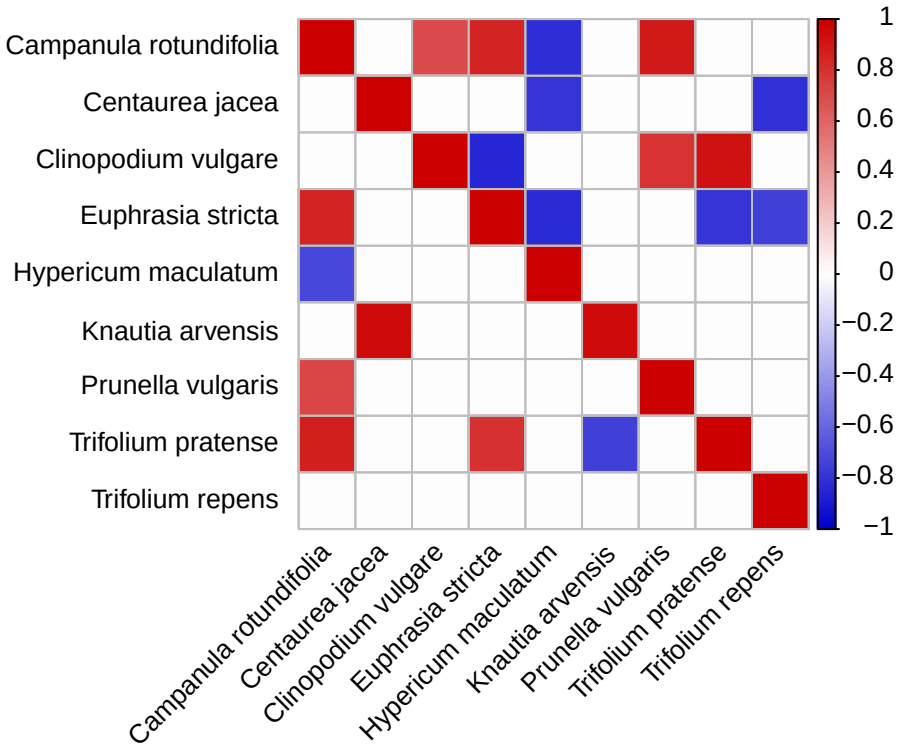


Figure 10: Effects of conspecific (diagonal) and heterospecific (off-diagonal) log(flower abundance) on log(number of visits), shown as the posterior support for positive and negative effects. Parameters with at least 85% posterior support are shown

Such plots are useful for visualising many parameters. However, to interpret the results in terms of biology, it is also necessary to look at the parameter estimates. In this model, the beta parameters describe the change in log(visits) per change in log(flowers).

Table 1: Parameter estimates for the effect of the log(floral abundance) of the species given in rows on the log(pollinator visitation) to the species given in columns

	S1	S2	S3	S4	S5	S6	S7	S8	S9
Campanula rotundifolia (S1)	0.51	0.01	-0.02	0.11	-0.10	0.08	0.11	0.12	0.09
Centaurea jacea (S2)	-0.05	0.70	-0.13	-0.01	-0.03	0.53	0.17	0.08	0.11
Clinopodium vulgare (S3)	0.07	-0.06	0.63	0.00	-0.04	-0.04	-0.01	0.00	-0.05
Euphrasia stricta (S4)	0.07	0.02	-0.07	0.31	0.01	0.06	0.01	0.06	0.03
Hypericum maculatum (S5)	-0.11	-0.17	0.00	-0.11	0.49	-0.09	-0.05	0.03	0.07
Knautia arvensis (S6)	0.04	-0.41	-0.31	0.10	-0.10	0.47	-0.33	-0.25	-0.11
Prunella vulgaris (S7)	0.11	-0.14	0.10	0.07	-0.07	0.06	0.52	-0.01	0.02
Trifolium pratense (S8)	-0.03	0.08	0.16	-0.10	-0.04	0.05	0.10	0.45	-0.04
Trifolium repens (S9)	0.00	-0.38	-0.04	-0.11	-0.03	-0.12	-0.09	-0.03	0.34

To assess parameter uncertainty, we can access the 95% credible intervals of each parameter. For example, here are the quantiles for the effect of each species on pollinator visitation to *Clinopodium vulgare* (Species 3). The intra-speficic effect `nflowers` is well supported, while the effects of *Euphrasia stricta*, *Prunella vulgaris* and *Trifolium pratense* are reasonably well supported.

```
post = convertToCodaObject(m, spNamesNumbers=c(F, T), covNamesNumbers=c(T, F))
round(summary(post$Beta)$quantiles[c(37, 28:29, 31:36),], 3)
```

```
##                                    2.5%    25%    50%    75% 97.5%
## B[nflowers, (S3)]                 0.423  0.549  0.625  0.708 0.856
## B[Campanula_rotundifolia, (S3)] -0.208 -0.092 -0.024  0.044 0.165
## B[Centaurea_jacea, (S3)]        -0.493 -0.250 -0.122 -0.002 0.239
## B[Euphrasia_stricta, (S3)]      -0.175 -0.106 -0.070 -0.036 0.027
## B[Hypericum_maculatum, (S3)]    -0.179 -0.063 -0.005  0.055 0.169
## B[Knautia_arvensis, (S3)]       -0.883 -0.519 -0.310 -0.097 0.297
## B[Prunella_vulgaris, (S3)]      -0.068  0.046  0.101  0.154 0.266
## B[Trifolium_pratense, (S3)]     -0.015  0.098  0.159  0.220 0.342
## B[Trifolium_repens, (S3)]       -0.301 -0.128 -0.044  0.042 0.209
```

Construct a gradient for effects of *Campanula* flower density

```
Gradient = constructGradient(m, focalVariable = "Campanula_rotundifolia")

predY = predict(m, Gradient=Gradient, expected=TRUE, predictEtaMean=FALSE)

par(mfrow = c(3,3))
for(i in 2:m$ns){
  plotGradient(m, Gradient, predY, measure = "Y", index=i, showData=F)
}
```

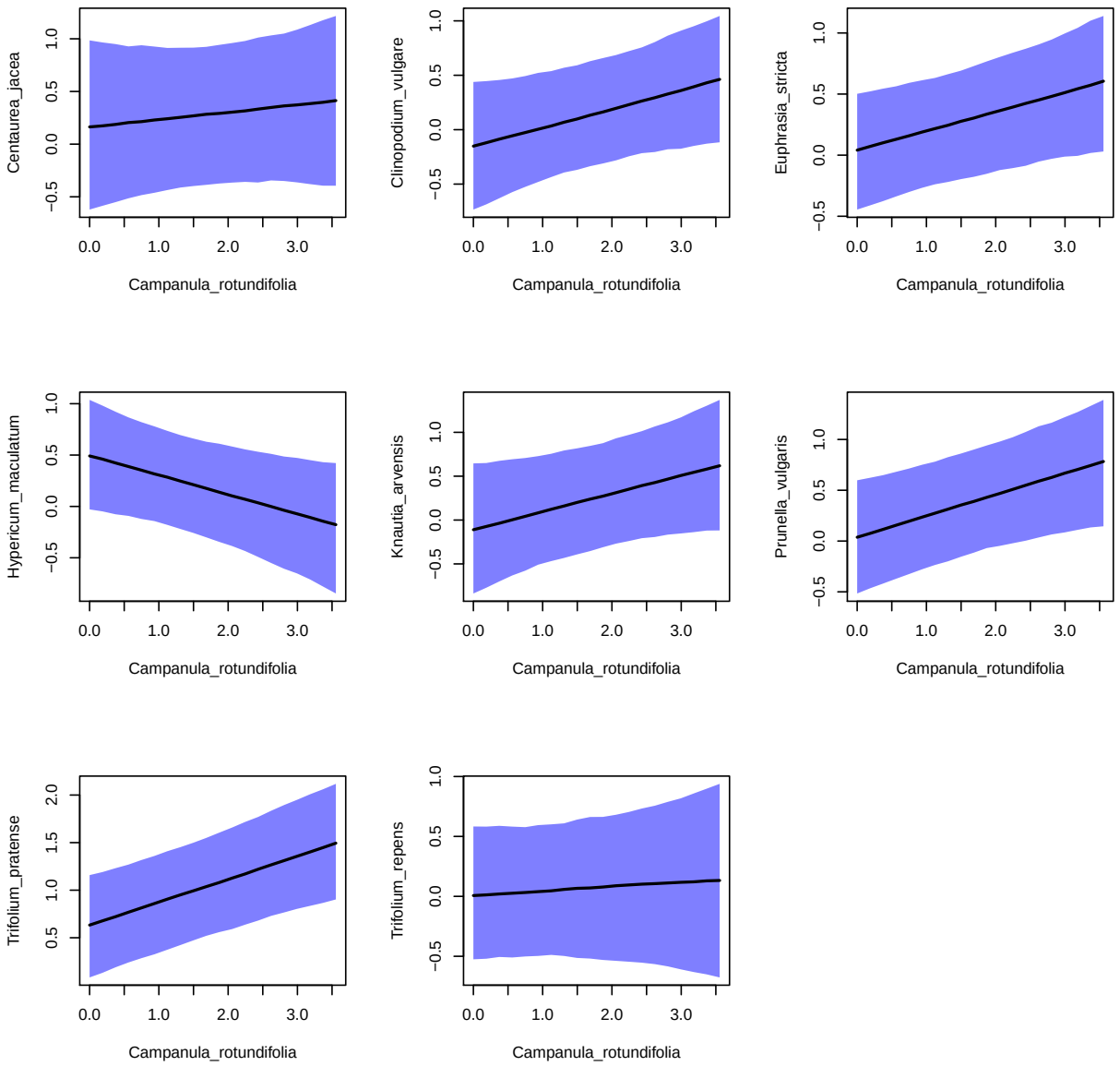


Figure 11: Effect of *Campanula* flower abundance on pollinator visitation (log visits)

Model 4: Visitation rates with conspecific + heterospecific flower counts as covariates

Finally, it may be of interest to know whether the abundance of conspecific and heterospecific flowers affects the number of visits to individual flowers, rather than the total number of visits to all flowers. Thus, we fit a model similar to Model 3, but with the number of visits translated into visitation rates (i.e. number of visits/number of flowers).

Model setup and MCMC sampling

Read data files

To translate the number of visits into visitation rates, we simple divide by the number of flowes of each species in each sampling unit. We then $\log(x+1)$-transform the flower abundances (covariates).

```
rm(list=ls())
Y = read.csv("model4/Y3.csv")
XData = read.csv("model4/XData1.csv")
studyDesign = read.csv("model4/studyDesign3.csv")
studyDesign$plot = as.factor(studyDesign$plot)

total_flowers = rowSums(XData[,4:12],na.rm=T)
total_visits = rowSums(Y,na.rm=T)
freqs = colSums(XData[,4:12],na.rm=T)/sum(colSums(XData[,4:12],na.rm=T))*100

Y = Y/XData[,4:12]
XData[,4:12] = apply(XData[,4:12], 2, function(x) log(x+1))
```

Define HMSC random levels

```
rL1 = HmscRandomLevel(units = unique(studyDesign[,1]))
rL2 = HmscRandomLevel(units = unique(studyDesign[,2]))
```

Compile a list containing `XData` for each species

```
xList = list()
for(i in 1:ncol(Y)){
  xList[[i]] = data.frame(data.frame(XData[,4:12]), Temp=XData$Temp, nflowers=c(XData[i+3]))
  xList[[i]][,i] = 0
  names(xList[[i]])[11] = "nflowers"
}
```

Set model formula for covariates

```
XFormula = as.formula(paste("~",paste(colnames(Y),collapse="+"),
                      "+ nflowers + poly(Temp, degree=2, raw=TRUE)"))
XFormula
```

```
## ~Campanula_rotundifolia + Centaurea_jacea + Clinopodium_vulgare +
##     Euphrasia_stricta + Hypericum_maculatum + Knautia_arvensis +
##     Prunella_vulgaris + Trifolium_pratense + Trifolium_repens +
##     nflowers + poly(Temp, degree = 2, raw = TRUE)
```

Set up the HMSC model

```
m = Hmsc(Y=as.matrix(Y), XData = xList,  XFormula = XFormula,
         distr="normal", studyDesign=studyDesign, ranLevels=list(plot=rL1,su=rL2))
```

Run MCMC and save the model object

```
thin = 200
samples = 1000
nChains = 2
adaptNf = ceiling(0.4*samples*thin)
transient = ceiling(0.5*samples*thin)

a = Sys.time()
m = sampleMcmc(m, samples = samples, thin = thin,
               adaptNf = rep(adaptNf,m$nr),
               transient = transient,
               nChains = nChains, nParallel = 1)
Sys.time() - a

save(m, file ="model4/mod4_thin200_samples1000_chains2.RData")
```

Evaluating chain convergence and model fit

Load the model object

```
load("model4/mod4_thin200_samples1000_chains2.RData")
```

Compute effective sample sizes and potential scale reduction factors

```
post = convertToCodaObject(m)

esBeta = effectiveSize(post$Beta)
summary(esBeta)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1714    2000    2000    2013    2000    2540
```

```
psrfBeta = gelman.diag(post$Beta)
summary(psrfBeta$psrf)
```

```
##    Point est.       Upper C.I.
##  Min.   :0.9993   Min.   :0.9993
##  1st Qu.:0.9997   1st Qu.:1.0004
##  Median :1.0002   Median :1.0025
##  Mean   :1.0011   Mean   :1.0055
##  3rd Qu.:1.0019   3rd Qu.:1.0060
##  Max.   :1.0096   Max.   :1.0425
```

```
esOmega = effectiveSize(post$Omega[[1]])
summary(esOmega)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1629    2000    2000    2043    2098    2890
```

Write posterior trace plots to pdf

```
pdf("model4/posterior_plots/BetaPost.pdf")
plot(post$Beta)
dev.off()

pdf("model4/posterior_plots/OmegaPost.pdf")
plot(post$Omega[[1]])
plot(post$Omega[[2]])
dev.off()
```

Extract and assess parameter estimates

Compute predicted values

```
predY = computePredictedValues(m)
```

Evaluate model fit

The explanatory power is somewhat lower than for the model of total visit number.

```
MF = evaluateModelFit(m, predY)
round(MF$R2, 2)
```

```
## [1] 0.32 0.30 0.33 0.18 0.20 0.45 0.25 0.20 0.14
```

```
round(mean(MF$R2), 2)
```

```
## [1] 0.26
```

Compute and plot variance partitioning

Less variance is now explained by conspecific flower density, which partly explains the reduced explanatory power.

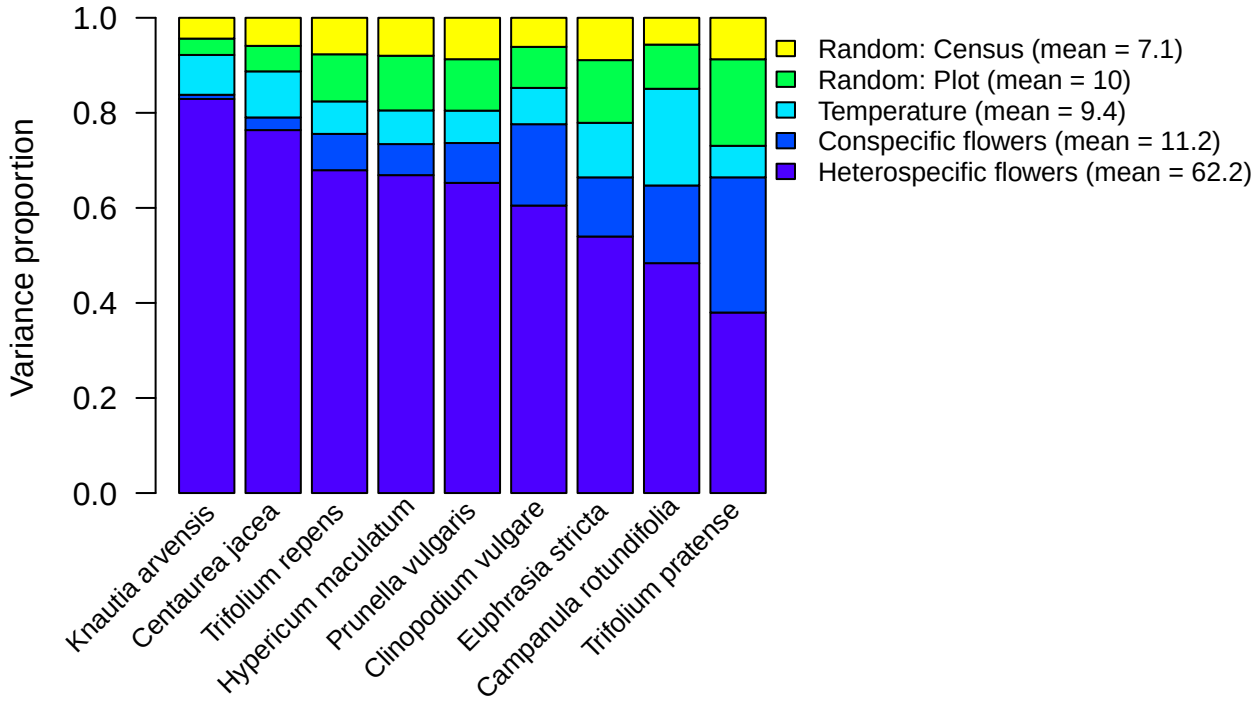


Figure 12: Variance partitioning for each species for Model 4

```
groups = c(rep(1,10), 2, 3, 3)
groupnames = c("Heterospecific flowers", "Conspecific flowers","Temperature")
VP = computeVariancePartitioning(m, groups, groupnames)

outvals = VP$vals
ng = dim(outvals)[1]
leg = groupnames
m$rLNames = c("Plot", "Census")
for (r in 1:m$nr) {leg = c(leg, paste("Random: ", m$rLNames[r], sep = ""))}
means = round(100 * rowMeans(outvals), 1)
for (i in 1:ng) {leg[i] = paste(leg[i], " (mean = ", toString(means[i]), ")", sep = "")}

plotorder = order(outvals[1,],decreasing=T)
names = gsub("_", " ", colnames(m$Y))

par(mar = c(8,4,2,12), xpd=T)
barplot(outvals[,plotorder], xlab = "", ylab = "Variance proportion",
        args.legend=list(x=19.8,y=1, bty="n", cex=.8),
        axisnames=F,
        las = 1, legend = leg, col = topo.colors(ng, alpha = 1))
text(x = seq(.5,10, length.out = 9), par("usr")[3] - 0.05, srt = 45, adj = .9, cex = .8,
     labels = names[plotorder], xpd = TRUE)
```

Extract and plot beta parameters

The main qualitative difference between the models occurs for the intraspecific effects of flower abundance, which are now often weak, and has changed to negative for *Trifolium pratense*. We have also changed the units of the regression coefficients. While in model 3 the betas described the change in the log number of visits per unit change in log number of flowers (roughly an elasticity), in model 4 the betas describe the change in the number of visits per flower per unit change in log number of flowers. For *Trifolium pratense*, this means that larger floral displays are visited more often, but the increase in visitation per flower is limited, so that the number of visits per flower decreases in large displays (intraspecific negative density dependence). For *Campanula rotundifolia* we observe the opposite, i.e. intraspecific positive density dependence.

```
pBeta = getPostEstimate(m, "Beta")

mat = pBeta$mean[2:10,]
diag(mat) = pBeta$mean[11,]

smat = 2*pBeta$support[2:10,] - 1
diag(smat) = 2*pBeta$support[11,] - 1

supp = pBeta$support[2:10,]
diag(supp) = pBeta$support[11,]

supportLevel = .85
mat = smat * ((supp > supportLevel) + (supp < (1 - supportLevel)) > 0)

names = gsub("_", " ", colnames(m$Y))
colnames(mat) = rownames(mat) = names
par(xpd=T)
corrplot(t(mat), method="color", col=colorRampPalette(c("blue3", "white", "red3"))(200),
         mar=c(5,7,0,2), tl.cex=.7, tl.col="black", tl.pos="n",
         cl.align.text="r", cl.offset=-.45, addgrid.col = "grey")
text(x = seq(1,9, length.out = 9), par("usr")[3] -
       -2.00, srt = 45, adj = 1, cex = .8, labels = names, xpd = TRUE)
text(y = seq(1,9, length.out = 9), par("usr")[3] -
       -2.00, srt = 0, adj = 1, cex = .8, labels = rev(names), xpd = TRUE)
```

To further understand the difference between the two models, and thus the consequences of translating the number of visits into visitation rates, we plot the estimated beta parameters from the two models.

The estimates tend to fall in the same quadrant, indicating similar sign in both models. The change in sign for *Trifolium pratense* is visible in the upper left quadrant.

```
load("model3/mod3_thin200_samples1000_chains2.RData")
mod3_mat = getPostEstimate(m, "Beta")$mean[2:10,]
diag(mod3_mat) = getPostEstimate(m, "Beta")$mean[11,]

load("model4/mod4_thin200_samples1000_chains2.RData")
mod4_mat = getPostEstimate(m, "Beta")$mean[2:10,]
diag(mod4_mat) = getPostEstimate(m, "Beta")$mean[11,]

diag_mod3 = diag(mod3_mat)
diag_mod4 = diag(mod4_mat)
diag(mod3_mat) = NA
diag(mod4_mat) = NA
```

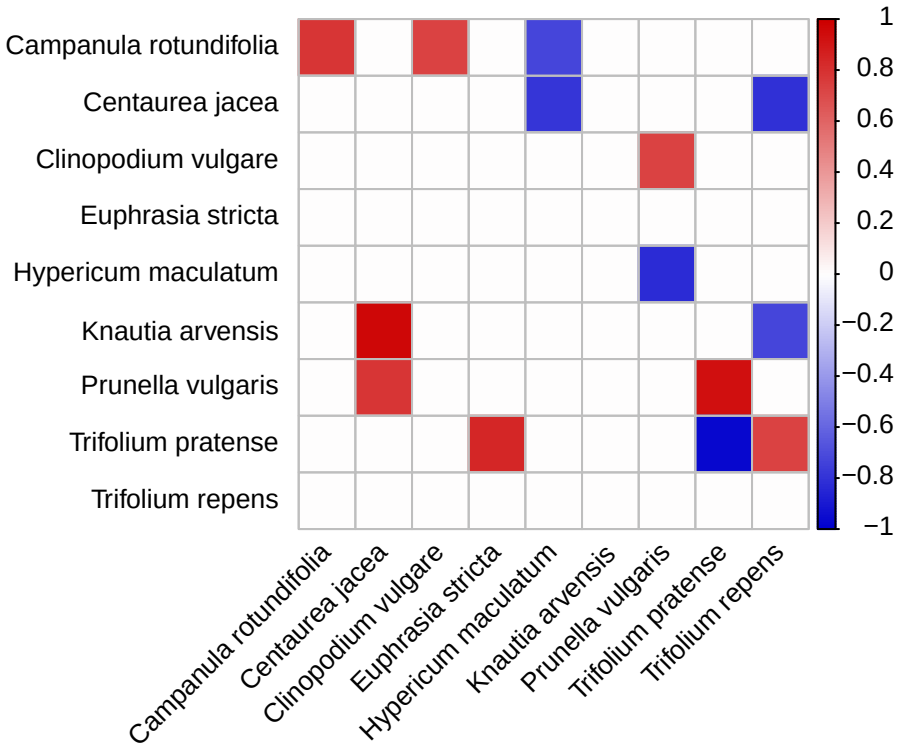


Figure 13: Effects of conspecific (diagonal) and heterospecific (off-diagonal) log(flower abundance) on visitation rates (number of visits per flower), shown as the posterior support for positive and negative effects. Parameters with at least 85% posterior support are shown

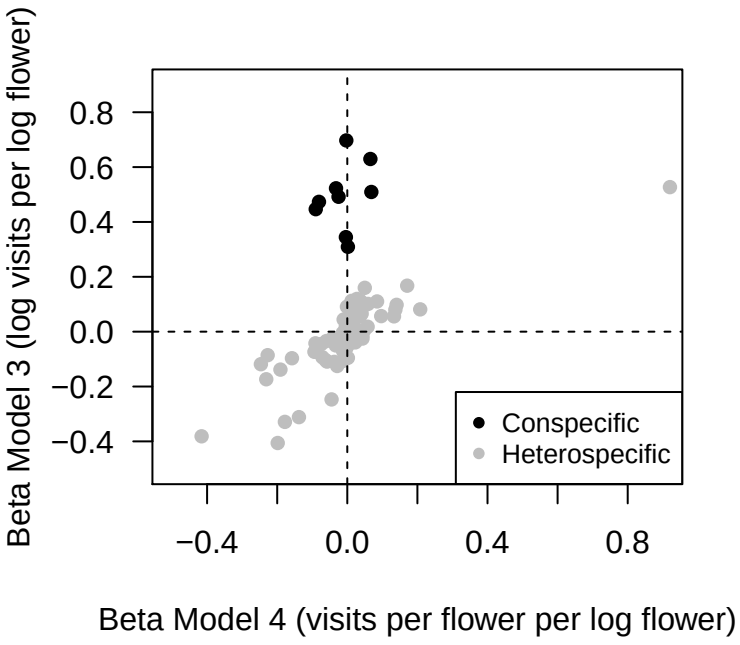


Figure 14: Comparison of regression coefficients for effects of conspecific (black) and heterospecific (grey) flower abundances estimated from Model 3 and 4

```
plot(mod4_mat,mod3_mat,col="grey",pch=16,ylim=c(-.5,.9),xlim=c(-.5,.9),las=1,
     xlab="Beta Model 4 (visits per flower per log flower)",
     ylab="Beta Model 3 (log visits per log flower)")
points(diag_mod4,diag_mod3,col="black",pch=16)
abline(h=0, lty=2)
abline(v=0, lty=2)
legend("bottomright", pch=c(16,16), col=c("black","grey"),
       legend=c("Conspecific","Heterospecific"), cex=.8)
```

Comparing the betas from the two models, ~80% have the same sign in both models.

```
t = (mod3_mat>0 & mod4_mat>0 | mod3_mat<0 & mod4_mat<0)
round(sum(t,na.rm=T)/(sum(t>-1,na.rm=T))*100, 2)
```

```
## [1] 79.17
```

```
f = (mod3_mat<0 & mod4_mat>0 | mod3_mat>0 & mod4_mat<0)
round(sum(f,na.rm=T)/(sum(f>-1,na.rm=T))*100, 2)
```

```
## [1] 20.83
```

The estimates for the interspecific effects are strongly correlated.

```
cor(c(mod3_mat),c(mod4_mat),"pairwise")
```

```
## [1] 0.8241469
```

```
cor(diag_mod3,diag_mod4,"pairwise")
```

```
## [1] 0.2694395
```