R-Code for BCI-Coexistence Analysis

Oliver Purschke

21 September, 2017

Contents

1 Load required packages

```r
library(knitr)
library(vegan)
library(doParallel)
library(dplyr)
library(rosalia)
library(mistnet)
library(phytools)
library(gridExtra)
library(picante)
library(adephylo)
library(ggplot2)
```

2 Phylogeny

```
path.phylo <- "/home/oliver/Dokumente/PhD/PostPhD/Projects/BCI_Coexistence/Data/Phylo/"
## load BCI phylogeny
bci.tree <- read.tree(paste(path.phylo, "dated.tree.tre", sep = ""))
bci.tree.OP <- read.tree(paste(path.phylo, "dated.tree.OP.tre", sep = ""))

# [1] "phyfull"     465              "bci.tree.node.OP"  495
# [3] "traits.abund.all"          "phy.308"
```

Check names in phylogeny (do they correspond to nate's and joe' new nomenclature?)

```
bci.data$phyfull <- bci.tree
```

Rename one species

```
bci.tree$tip.label[bci.tree$tip.label=="Appunia_seibertii"] <- "Morinda_seibertii"
```

2.1 Manually add missing species to the phylogeny based on taxonomy

Take Joe's list to get info about family-affiliation.

2.1.1 Add to congeners

- "Alchornea_costaricensis" # added
- "Apeiba_hybrid" # added
- "Bactris_coloradonis" # added, check whether it belongs to b.col or b.bar
- "Banara_guianensis" # # added to genus Hasseltia (Flacourtiace)
- "Bertiera_guianensis" # to be added to Rubiaceae (Borojoa, Calycophyllu, Chimarrhis,Chomelia, Cosmibuena, Coussarea, Coutarea, Elaeagia, Exostema, Faramea, Genipa (#), Guettarda (#), Hamelia, Isertia, Macrocnemum, Morinda, Palicourea, Pentagonia, Pogonopus (#), Posoqueria (#), Psychotria, Randia (##), Rosenbergiod (##), Rondeletia (#), Rudgea, Tocoyena (##), Alibertia, Alseis, Amaioua, Pittoniotis, Appunia) # added

2.1.2 Add to most closely related genus

- "Cestrum_megalophyllum" # family Solanaceae: Solanum (add to Solanum), Cestrum (does Markea belong to Solanaceae, yes!)
- "Cyathea_petiolata" # Cyatheaceae family not represented (!! tree fern, maybe skip?!), decided to include tree ferns (but might be kicked out later)
- "Colubrina_glandulosa" # to be added to "Rhamnaceae" (family not represented). Needed to find most closely related family. Sister to clade that holds Ficus and Cecropia (N143) # added

- “Drypetes_standleyi” # (family Euphorbiacea: Hura, Hieronyma, Mabea, Maprounea, Margaritaria, Pera, Sapium, Tetrorchidiu, Acalypha, Adelia, Alchornea, Amanoa, Croton, Drypetes) # added to Margaritina
- “Ficus_colubrinae” # two groups within genus Ficus add to one of them # added
- “Ficus_pertusa” # two groups within genus ficus add to one of them # added
- “Geonoma_interrupta” # family Aracaceae: Oenocarpus (##), Attalea (), Socratea (), Synechanthus (), Astrocaryum (), Bactris (), Chamaedorea, Elaeis # added to Oenocarpus (Arecoideae)
- “Inga_mucuna” # added
- “Koanophyllon_wetmorei” # family: Asteraceae (Verbesina, Vernonanthur) # added to family: Asteraceae N326
- “Lacmellea_panamensis” # family Apocynaceae: Rauvolfia, Stemmadenia, Tabernaemont, Thevetia # Aspidosperma renamed to “Lacmellea_panamensis”
- “Pavonia_dasypetala” # search for family Malvacea (Hampea) # added
- “Lycianthes_maxonii” # family Solanaceae: Solanum (add to Solanum), Cestrum (does Markea belong to Solanaceae) # added
- “Maclura_tinctoria” # family Moraceae: Maquira (#), Trophis, Perebea (#), Poulsenia (#), Pseudolmedia (#), Sorocea, Brosimum, Castilla (#), Ficus, Trophis # added
- “Miconia_prasina” # added
- “Ocotea_whitei” # added
- “Piper_schiedeanum” # added
- “Pachira_sessilis” # added
- “Prioria_copaifera” # family Fabaceae : Schizolobium, Senna, Tachigali, Brownea, Cassia, Copaifera (#), Dialium, Hymenaea (#) # added
- “Psychotria_hoffmannseggiana”, see Sedio (2012) in J.Ecol. for Psychotria phylogeny # added to Psychotria_acuminata
- “Quararibea_asterolepis” # family Bombacaceae: add to N73 (holds Ceiba and Pseudobombax) # added
- “Schefflera_morototoni” # added to Dendropanax
- “Simarouba_amara” # family Simaroubacea: Picramnia, Quassia (#) (both distantly related in phylogeny) # added to Quassia
- “Talisia_nervosa” # added
- “Tetragastris_panamensis” # family Burseraceae: Trattinnicki, Bursera, Protium # added to Protium
- “Trichospermum_galeottii” # add to Apeiba # added
- “Vismia_macrophylla” # added
- “Xylopia_macrantha” # family Annonaceae: Anaxagorea, Annona (#), Desmopsis, Guatteria, Mosannona, Oxandra, Rollinia (#), Unonopsis
- “Xylosma_chlorantha” # added

2.1.3 Example how to add species based on node information

```r
bci.tree.nodes <- makeNodeLabel(drop.tip(bci.tree,
"Gnetum_leyboldii"), prefix = "N")

write.tree(bci.tree.nodes, file = "bci.tree.node.tre")

pdf("bci.tree.nodes.pdf", height = 25, width = 25)
plot(bci.tree.nodes, type = "f", cex = .6, show.node.label = TRUE)
dev.off()
```

2.1.4 Plotting the extended tree (495 taxa)

```
bci.tree.node.OP <- read.tree("bci.tree.node.OP.tre")
pdf("bci.tree.node.OP.pdf", height = 25, width = 25)
plot(bci.tree.node.OP, type = "f", cex = .6, show.node.label = TRUE)
dev.off()
```

2.2 Match phylogeny to trait data

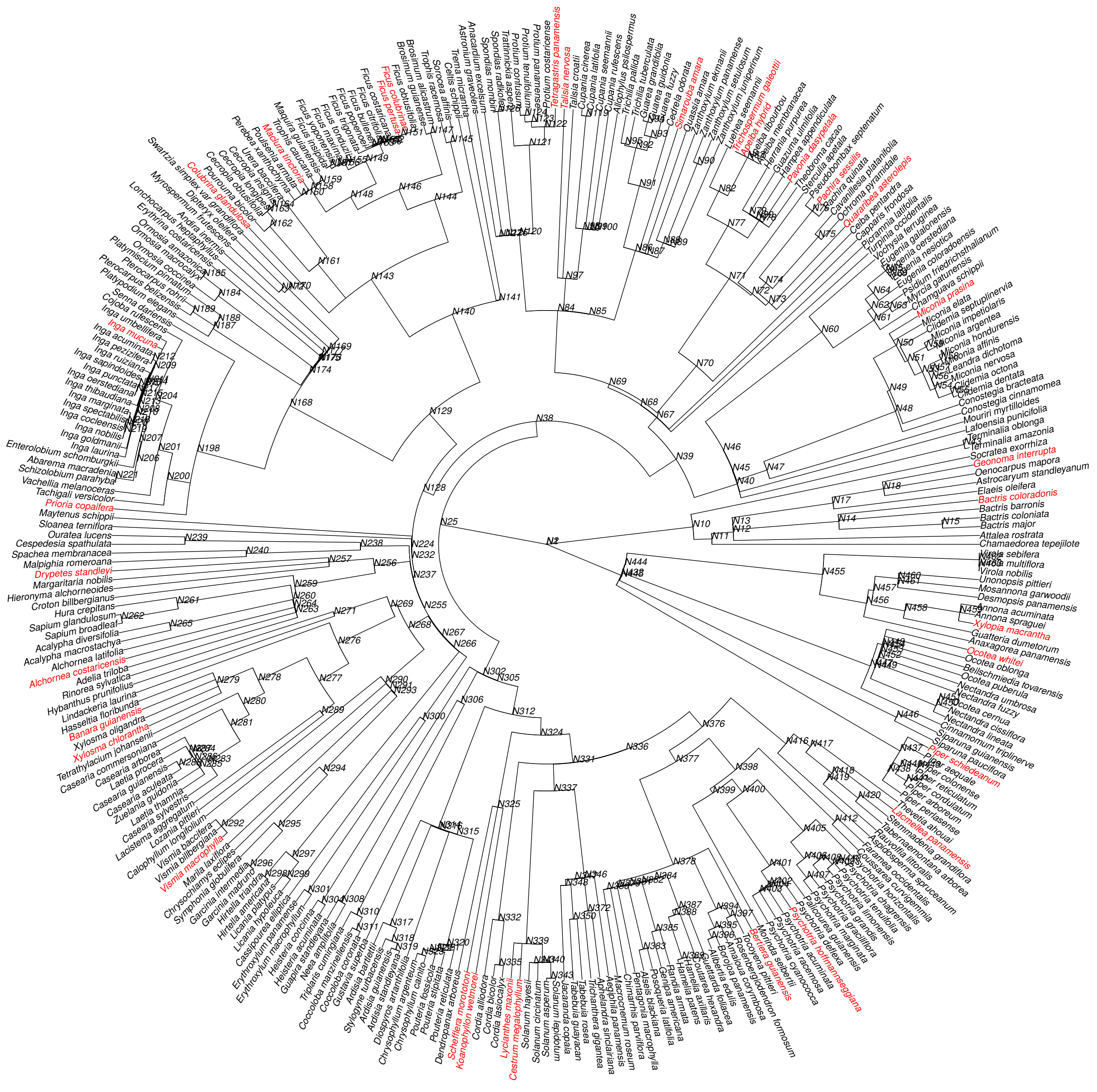
```
traits.abund <-
    read.table(paste(path.traits,
                     "Nadja_Traits/abun/demographic_traits_abun.txt", sep = ""), sep = "\t")

rownames(traits.abund)[rownames(traits.abund)=="Appunia_seibertii"] <- "Morinda_seibertii"

rownames(traits.abund)[which(rownames(traits.abund) %in%
                              bci.data$bci.tree.node.OP$tip.label ==
                              FALSE)]
```

Delete “Cyathea_petiolata”, which is a fern.

```
traits.abund.wo.Cyathea <-
traits.abund[rownames(traits.abund)!="Cyathea_petiolata", ]
rownames(traits.abund.wo.Cyathea)[which(rownames(traits.abund.wo.Cyathea)
%in% bci.data$bci.tree.node.OP$tip.label == FALSE)]

bci.data$traits.abund.all <- traits.abund.wo.Cyathea

mat <- match.phylo.comm(phy=bci.tree.node.OP, comm=t(bci.data$traits.abund.all))
mat$phy$edge.length <- mat$phy$edge.length+.1

bci.data$phy.308 <- mat$phy
bci.data$traits.abu.308 <- t(mat$comm)

save(bci.data, file = paste(path.data, "bci.data.Rdata", sep = ""))
```

2.2.1 Plot added species with different tip.color

```
names(bci.tree.node.OP)
ss <- bci.tree.node.OP$tip.label[c(1,3,5,7)]

ss <- c("Alchornea_costaricensis", "Apeiba_hybrid" , "Bactris_coloradonis",
        "Banara_guianensis", "Bertiera_guianensis", "Cestrum_megalophyllum",
        "Colubrina_glandulosa", "Colubrina_glandulosa", "Drypetes_standleyi",
        "Ficus_colubrinae", "Ficus_pertusa", "Geonoma_interrupta",
        "Inga_mucuna","Koanophyllon_wetmorei", "Lacmellea_panamensis", "Pavonia_dasypetala",
        "Lycianthes_maxonii", "Maclura_tinctoria", "Miconia_prasina" ,"Ocotea_whitei",
        "Piper_schiedeanum" ,"Pachira_sessilis", "Prioria_copaifera",
        "Psychotria_hoffmannseggiana", "Quararibea_asterolepis", "Schefflera_morototoni",
        "Simarouba_amara", "Talisia_nervosa", "Tetragastris_panamensis",
        "Trichospermum_galeottii", "Vismia_macrophylla", "Xylopia_macrantha",
        "Xylosma_chlorantha")

pdf("bci.tree.308.pdf", height = 25, width = 25)
plot(mat$phy, type = "f", cex = .9, show.node.label = TRUE,
tip.color=ifelse(mat$phy$tip.label %in% ss, "red","black"))
dev.off()
```



2.3 Phylogenetic signal in demographic rates

2.3.1 Blomberg's K

```
MultiK<- function(tre, traits){
    require(phytools)
    mat <- matrix(NA, ncol = 2, nrow = dim(traits)[2])
    colnames(mat) <- c("K","P")
    rownames(mat) <- colnames(traits)
    for (i in 1:dim(traits)[2]){
        x <- phylosig(tre, traits[tre$tip.label,i], method="K", test=TRUE,
                      nsim=1000)
        mat[i,] <- round(as.numeric(x), 3)
    }
    mat
}

mat2 <- match.phylo.comm(phy=mat$phy, comm=t(res))
                                        # comm = transposed species x traits matrix

multk <- MultiK(bci.data$phy206, bci.data$traits.pca[,c(16,18)])
multk
write.csv(multk, file = "multk.csv")
```

```
kable(read.csv("multk.csv"))
```

X	K	P
abun.rec2	0.072	0.344
rec.nr.mean2	0.020	0.520
rec.light.mean2	0.040	0.030
growth.mean2	0.069	0.001
growth.light.mean2	0.037	0.037
mort.mean2	0.022	0.395
mort.light.mean2	0.024	0.316

2.3.2 Pagel's λ

```
MultiLambda <- function(tre, traits){
    require(phytools)
    mat <- matrix(NA, ncol = 4, nrow = dim(traits)[2])
    colnames(mat) <- c("lambda","logL","logL0","P")
    rownames(mat) <- colnames(traits)
    for (i in 1:dim(traits)[2]){
        x <- phylosig(tre, traits[tre$tip.label,i], method="lambda", test=TRUE)
        mat[i,] <- round(as.numeric(x), 3)
    }
    mat
}

multlam <- MultiLambda(phy=mat2$phy, comm=t(mat2$comm)[,c(1,2,4,6,8,10,12,14:19)])
                                        # comm = transposed species x traits matrix
multlam
write.csv(multlam, file = "multlam.csv")
```

```
kable(read.csv("multlam.csv"))
```

X	lambda	logL	logL0	P
abun.rec2	0.941	-1941.671	-1953.767	0.000

X	lambda	logL	logL0	P
rec.nr.mean2	0.166	-401.625	-402.322	0.238
rec.light.mean2	0.247	-184.106	-184.505	0.371
growth.mean2	0.654	-123.385	-135.680	0.000
growth.light.mean2	0.461	50.587	43.612	0.000
mort.mean2	0.563	-263.038	-270.492	0.000
mort.light.mean2	0.348	54.061	51.666	0.029

2.3.3 Blomberg's K, but including measurement error

```
MultiKerror<- function(tre, traits, error){
    require(phytools)
    mat <- matrix(NA, ncol = 4, nrow = dim(traits)[2])
    colnames(mat) <- c("K","P","sig2","logL")
    rownames(mat) <- colnames(traits)
    for (i in 1:dim(traits)[2]){
        x <- phylosig(tre, traits[tre$tip.label,i], method="K", se=error[tre$tip.label,i],
                      test=TRUE, nsim=1000)
        mat[i,] <- round(as.numeric(x), 3)
    }
    mat
}

multkerr <- MultiKerror(bci.data$phy206, bci.data$traits.pca[,c(2,4,6,8,10,12)],
                        error=bci.data$traits.pca[,c(3,5,7,9,11,13)])

multkerr
write.csv(multkerr, file = "multkerr.csv")
```

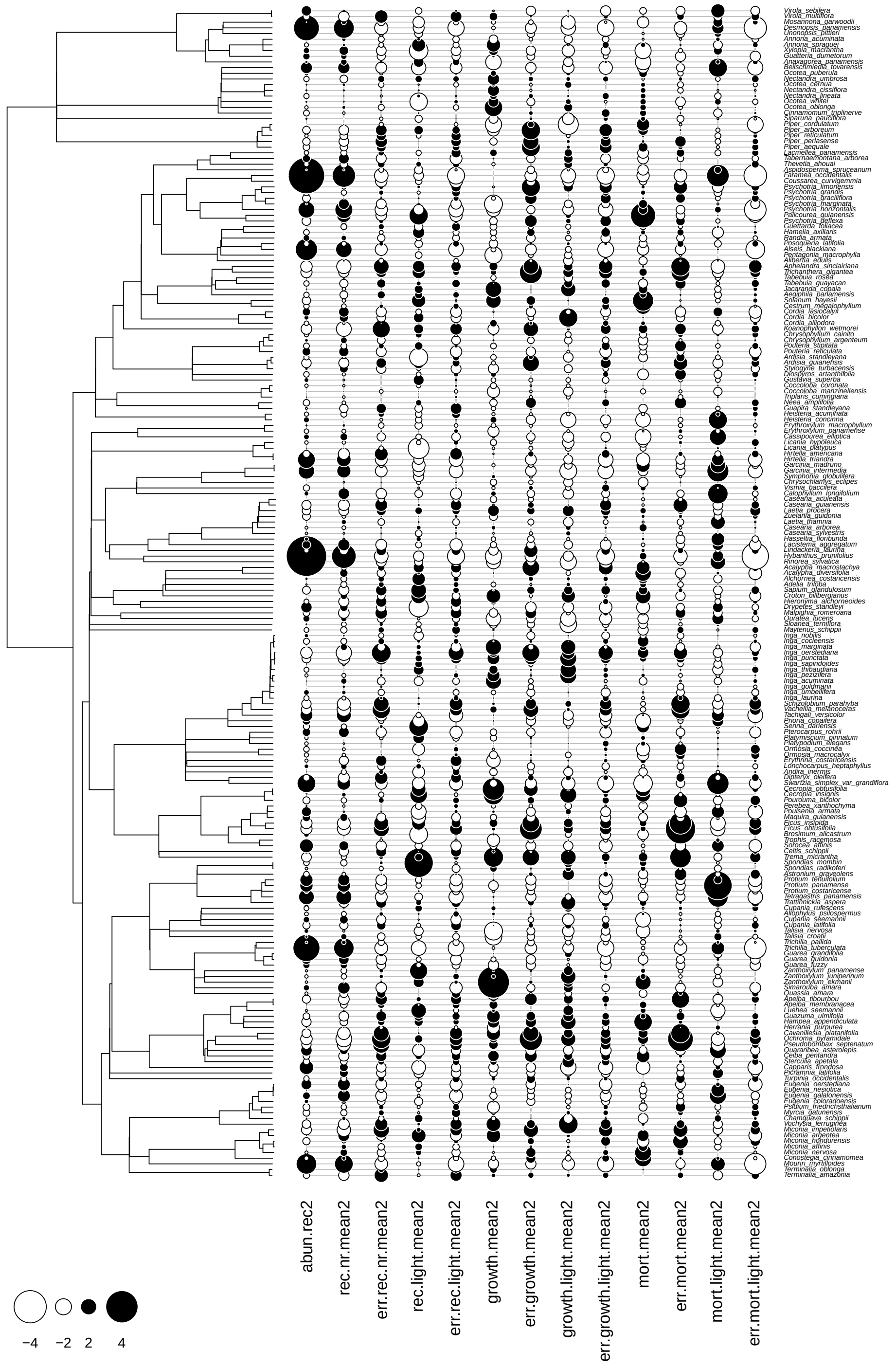
```
kable(read.csv("multkerr.csv"))
```

X	K	P	sig2	logL
rec.nr.mean2	0.384	0.101	0.051	-427.558
rec.light.mean2	0.479	0.045	0.001	-272.057
growth.mean2	0.460	0.037	0.001	-159.899
growth.light.mean2	0.746	0.017	0.000	-45.587
mort.mean2	0.464	0.005	0.007	-307.206
mort.light.mean2	0.720	0.336	0.000	-88.420

2.4 Plot abundance + 6 traits + 6 measurement error (n=13) on the phylogeny

```
comm2 <- t(mat2$comm)
comm2[,1] <- comm2[,1]^.25

herb <- phylo4d(mat2$phy, comm2)

pdf("BCI.traits.phylo.pdf", width = 12, height = 20)
# postscript(file = "BCI.traits.phylo.eps",width = 12, height = 20,
paper = "special", onefile = FALSE, horizontal = FALSE, pointsize=12)
```

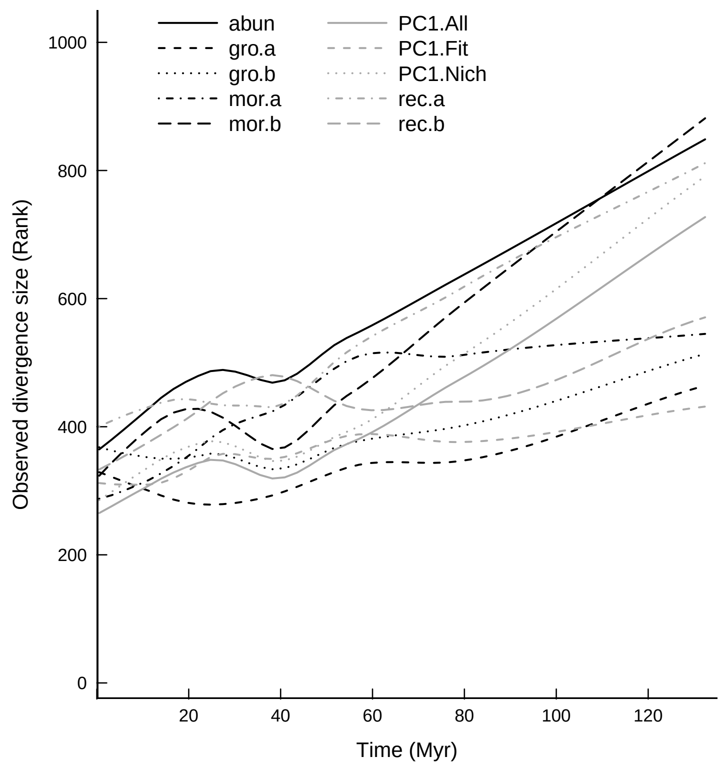
```r
phytab <- table.phylo4d(herb, treetype="phylogram", show.node.label=F,
box=F, ratio.tree=1/3, font=3, cex.label=.5, cex.symbol=1.2,
cex.legend = .8)
dev.off()
```


Virola_sebifera
Virola_multiflora
Mosannona_garwoodii
Desmopsis_panamensis
Unonopsis_pittieri
Annona_acuminata
Annona_spraguei
Xylopia_macrantha
Guatteria_dumetorum
Anaxagorea_panamensis
Beilschmiedia_tovarensis
Ocotea_puberula
Nectandra_umbrosa
Ocotea_cernua
Nectandra_cissiflora
Nectandra_lineata
Ocotea_whitei
Ocotea_oblonga
Cinnamomum_triplinerve
Siparuna_pauciflora
Piper_cordulatum
Piper_arboreum
Piper_reticulatum
Piper_perlasense
Piper_aequale
Lacmellea_panamensis
Tabernaemontana_arborea
Thevetia_ahouai
Aspidosperma_spruceanum
Faramea_occidentalis
Coussarea_curvigemmia
Psychotria_limonensis
Psychotria_grandis
Psychotria_graciliflora
Psychotria_marginata
Psychotria_horizontalis
Palicourea_guianensis
Psychotria_deflexa
Guettarda_foliacea
Hamelia_axillaris
Randia_armata
Posoqueria_latifolia
Alseis_blackiana
Pentagonia_macrophylla
Alibertia_edulis
Aphelandra_sinclairiana
Trichanthera_gigantea
Tabebuia_rosea
Tabebuia_guayacan
Jacaranda_copaia
Aegiphila_panamensis
Solanum_hayesii
Cestrum_megalophyllum
Cordia_lasiocalyx
Cordia_bicolor
Cordia_alliodora
Koanophyllon_wetmorei
Chrysophyllum_cainito
Chrysophyllum_argenteum
Pouteria_stipitata
Pouteria_reticulata
Ardisia_standleyana
Ardisia_guianensis
Stylogyne_turbacensis
Diospyros_artanthifolia
Gustavia_superba
Coccoloba_coronata
Coccoloba_manzinellensis
Triplaris_cumingiana
Neea_amplifolia
Guapira_standleyana
Heisteria_acuminata
Heisteria_concinna
Erythroxylum_macrophyllum
Erythroxylum_panamense
Cassipourea_elliptica
Licania_hypoleuca
Licania_platypus
Hirtella_americana
Hirtella_triandra
Garcinia_madruno
Garcinia_intermedia
Symphonia_globulifera
Chrysochlamys_eclipes
Vismia_baccifera
Calophyllum_longifolium
Casearia_aculeata
Casearia_guianensis
Laetia_procera
Zuelania_guidonia
Laetia_thamnia
Casearia_arborea
Casearia_sylvestris
Hasseltia_floribunda
Lacistema_aggregatum
Lindackeria_laurina
Hybanthus_prunifolius
Rinorea_sylvatica
Acalypha_macrostachya
Acalypha_diversifolia
Alchornea_costaricensis
Adelia_triloba
Sapium_glandulosum
Croton_billbergianus
Hieronyma_alchorneoides
Drypetes_standleyi
Malpighia_romeroana
Ouratea_lucens
Sloanea_terniflora
Maytenus_schippii
Inga_nobilis
Inga_cocleensis
Inga_marginata
Inga_oerstediana
Inga_punctata
Inga_sapindoides
Inga_thibaudiana
Inga_pezizifera
Inga_acuminata
Inga_goldmanii
Inga_umbellifera
Inga_laurina
Schizolobium_parahyba
Vachellia_melanoceras
Tachigali_versicolor
Prioria_copaifera
Senna_dariensis
Pterocarpus_rohrii
Platymiscium_pinnatum
Platypodium_elegans
Ormosia_coccinea
Ormosia_macrocalyx
Erythrina_costaricensis
Lonchocarpus_heptaphyllus
Andira_inermis
Dipteryx_oleifera
Swartzia_simplex_var_grandiflora
Cecropia_obtusifolia
Cecropia_insignis
Pourouma_bicolor
Perebea_xanthochyma
Poulsenia_armata
Maquira_guianensis
Ficus_insipida
Ficus_obtusifolia
Brosimum_alicastrum
Trophis_racemosa
Sorocea_affinis
Celtis_schippii
Trema_micrantha
Spondias_mombin
Spondias_radlkoferi
Astronium_graveolens
Protium_tenuifolium
Protium_panamense
Protium_costaricense
Tetragastris_panamensis
Trattinnickia_aspera
Cupania_rufescens
Allophylus_psilospermus
Cupania_seemannii
Cupania_latifolia
Talisia_nervosa
Talisia_croatii
Trichilia_pallida
Trichilia_tuberculata
Guarea_grandifolia
Guarea_guidonia
Guarea_fuzzy
Zanthoxylum_panamense
Zanthoxylum_juniperinum
Zanthoxylum_ekmanii
Simarouba_amara
Quassia_amara
Apeiba_tibourbou
Apeiba_membranacea
Luehea_seemannii
Guazuma_ulmifolia
Hampea_appendiculata
Herrania_purpurea
Cavanillesia_platanifolia
Ochroma_pyramidale
Pseudobombax_septenatum
Quararibea_asterolepis
Ceiba_pentandra
Sterculia_apetala
Capparis_frondosa
Picramnia_latifolia
Turpinia_occidentalis
Eugenia_oerstediana
Eugenia_nesiotica
Eugenia_galalonensis
Eugenia_coloradoensis
Psidium_friedrichsthalianum
Myrcia_gatunensis
Chamguava_schippii
Vochysia_ferruginea
Miconia_impetiolaris
Miconia_argentea
Miconia_hondurensis
Miconia_affinis
Miconia_nervosa
Conostegia_cinnamomea
Mouriri_myrtilloides
Terminalia_oblonga
Terminalia_amazonia
abun.rec2
rec.nr.mean2
err.rec.nr.mean2
rec.light.mean2
err.rec.light.mean2
growth.mean2
err.growth.mean2
growth.light.mean2
err.growth.light.mean2
mort.mean2
err.mort.mean2
mort.light.mean2
err.mort.light.mean2
−4
−2
2
4

2.5 Nodewise phylogenetic signal (divergence size)

This test has been carried out in Phylocom in the bash terminal. For interpretation of results, see Appendix in Purschke et al. (2013) J.Ecol..

```
div <- xyplot(Tsd.rankLow ~ age, groups=trait.name, data =
divergence.bci, type = "smooth", xlim = c(0, 135), ylim = c(-25,
1050), lty = c(1,2,3,4,5,1,2,3,4,5), par.settings = list(axis.line =
list(col = 0)),scales=list(col=1,tck=c(-1,0)),  # remove top and right
axes
              panel=function(...){
                lims <- current.panel.limits()
                panel.xyplot(...)
                panel.abline(h=lims$ylim[1],v=lims$xlim[1], lwd = 2.5)
              },
              layout.heights=list(axis.xlab.padding = 1),
lwd = 1.5, col =
               c("black","black","black","black","black","darkgrey","darkgrey","darkgrey","darkgrey",
                 "darkgrey"),
xlab = "Time (Myr)", ylab = "Observed divergence size (Rank)",
key=list(space="inside",  between = 1, padding.text = 2, just = c(.7,
.5), columns = 2, lines = list(lty = c(1,2,3,4,5,1,2,3,4,5), lwd =
1.5, col = c("black","black","black","black","black","darkgrey","darkgrey","darkgrey","darkgrey",
           "darkgrey")),text
= list(levels(divergence.bci$trait.name))))
plot(div)

pdf(file = "BCI.DivergenceSize.pdf",width = 6, height = 6.5,
pointsize=12)
# postscript(file = "BCI.DivergenceSize.eps",width = 6, height = 6.5,
paper = "special", onefile = FALSE, horizontal = FALSE, pointsize=12)
plot(div)
dev.off()
```



3 New Markov-network analyses (August 2016)

3.1 Read in all environmental data

If there are "." in the name, run the following script in the bash-shell to rename the files accordingly:

```
for filename in *_txt; do newname=`echo $filename | sed 's/\_txt$/.txt/g'`;
mv $filename $newname; done
```

```
path <- "/home/oliver/Dokumente/PhD/PostPhD/Projects/BCI_Coexistence/Data/Environment/
Kupers_grids_bci-2016-06-27"
files <- list.files(path, pattern="*.txt")
for (i in 1:length(files))
    assign(files[i], read.table(file.path(path, files[i])))
```

3.1.0.1 Bring environmental data in the same order as the species data:

```
env.to.spec <- function(x, ncol){
    as.vector(matrix(x, ncol = ncol, byrow = T))
}
```

3.1.0.2 Load all env.files for a particular scale and put them into one dataframe

```
library(foreach)
env.scale <- list()
```

```r
for (j in c("5m","10m","20m","40m","60m","80m","100m")){
    filenames <- list.files(path, pattern=j, full.names=TRUE)
    ldf <- lapply(filenames, read.table)
    res <- foreach(i = 1:length(filenames), .combine = cbind) %dopar% {
        as.vector(ldf[[i]])
    }
    env.scale <- c(env.scale, list(res))
}

str(env.scale)
names(env.scale) <- c("5m","10m","20m","40m","60m","80m","100m")
```

Make sure data are in the same order as species data (check for slope, ph and light):

```r
names(env.scale[[7]])
```

All grids are 1000x500, apart from:

- 40m: 1000 x 480
- 60m: 960 x 480
- 80m: 960 x 480

Set up ncol-values for the `env.to.spec`-function:

```r
ncol.vec <- c(200,100,50,25,16,12,10)

env.scale.2 <- env.scale

for(i in 1:7){
    for (j in 1:dim(env.scale[[i]])[2]){
        env.scale.2[[i]][,j] <- env.to.spec(env.scale[[i]][,j], ncol = ncol.vec[i])
    }
}

str(env.scale.2)

env.scale.correct <- env.scale
save(env.scale.correct, file = "env.scale.correct.Rdata")
```

3.1.1 Explore whether single environmental variables and abundance grids match up

3.1.1.1 Environmental variable

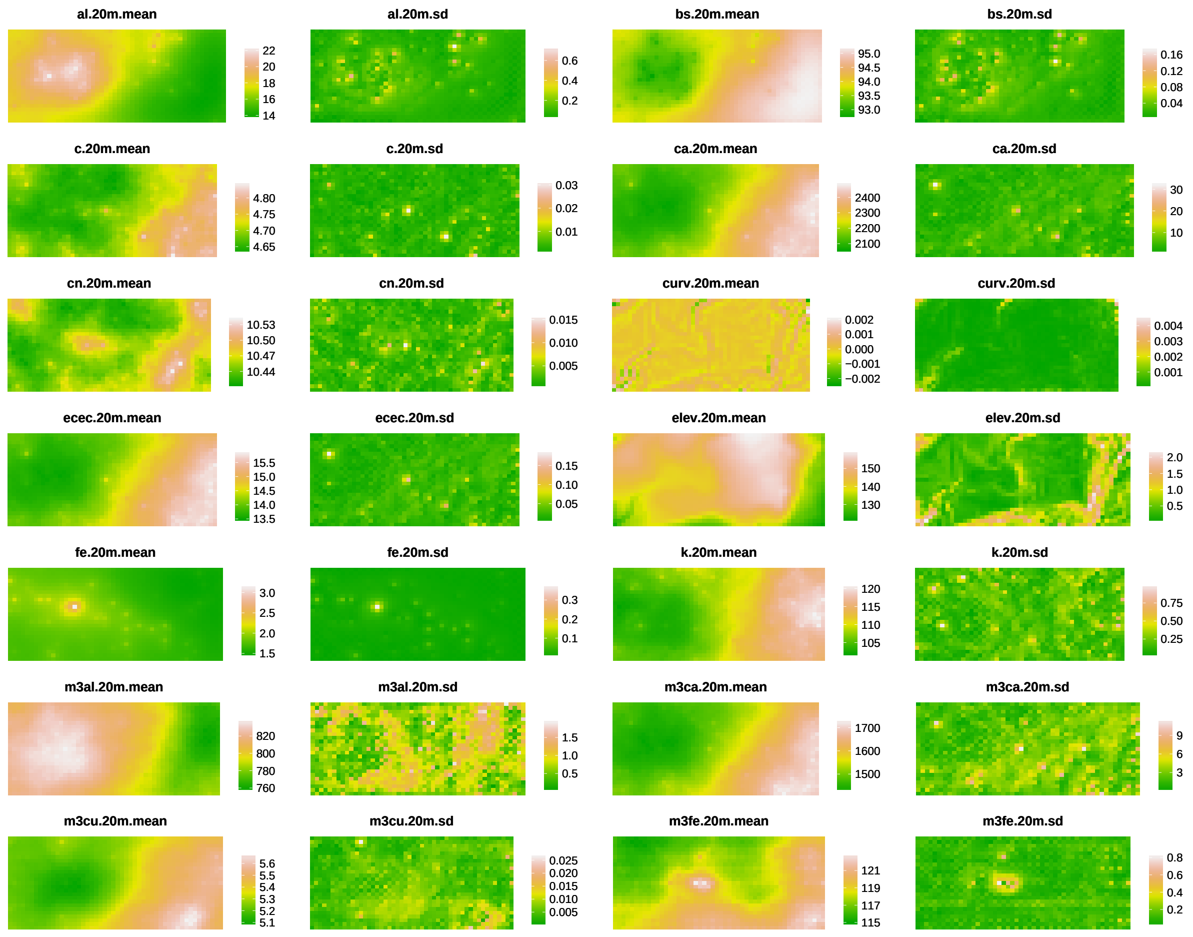
```r
filled.contour(matrix(env.scale[[3]][,55], ncol = 25, byrow = T),
               color.palette = terrain.colors, main = "slope.20m")

grid.213.7 <- bci.data$grid.213[c(1,2,4,8,12,16,20)]
length(grid.213.7)
names(grid.213.7)
lapply(grid.213.7, FUN=dim)
save(grid.213.7, file = "grid.213.7.Rdata")

filled.contour(log(matrix(grid.213.7[[3]][,147], ncol = 25, byrow = T)),
               color.palette = terrain.colors, main = "Slope.20m")
```

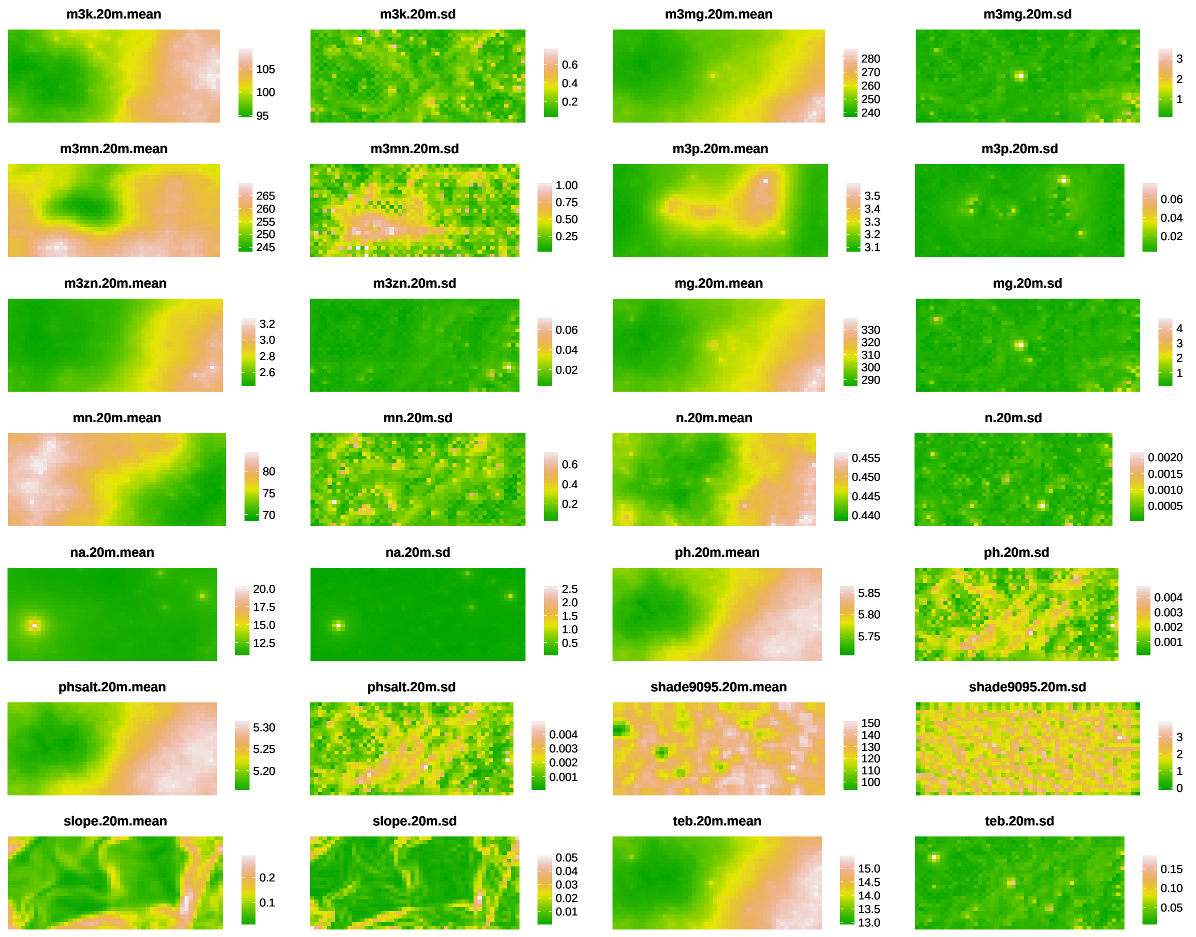
3.2 Plot environmental variables

```r
df <- expand.grid(x = 0:99, y = 0:49) # initialize coordinates
df$z <- env.scale[[2]][,47] # set up fill values
# default is compatible with geom_tile()
ggplot(df, aes(x, y, fill = z)) + geom_raster()
```

Using `ggplot` in loop (only works for one page), e.g. at 100m scale (grain size):

```r
p = list()
df <- expand.grid(x = 0:49, y = 0:24) # adjust for other scales

for(i in 1:28) {
  df$z = env.to.spec(env.scale[[3]][,i], ncol = 50) # adjust ncol for other scales
  p[[i]] = ggplot(df, aes(x, y, fill = z)) +
      geom_raster() +
      ggtitle(names(env.scale[[3]])[i]) +
      scale_fill_gradientn(colours = terrain.colors(10)) +
      theme(axis.title.x=element_blank(),
        axis.text.x=element_blank(),
        axis.title.y=element_blank(),
        axis.text.y=element_blank(),
        axis.ticks=element_blank(),
        axis.line = element_blank(),
        legend.position="right")+
      labs(fill="") # remove legend title
}

ggsave(file="env.100m.maps1.new.pdf", do.call("grid.arrange", c(p[1:28], ncol=4)),
       width = 45, height = 35, units = "cm")
```

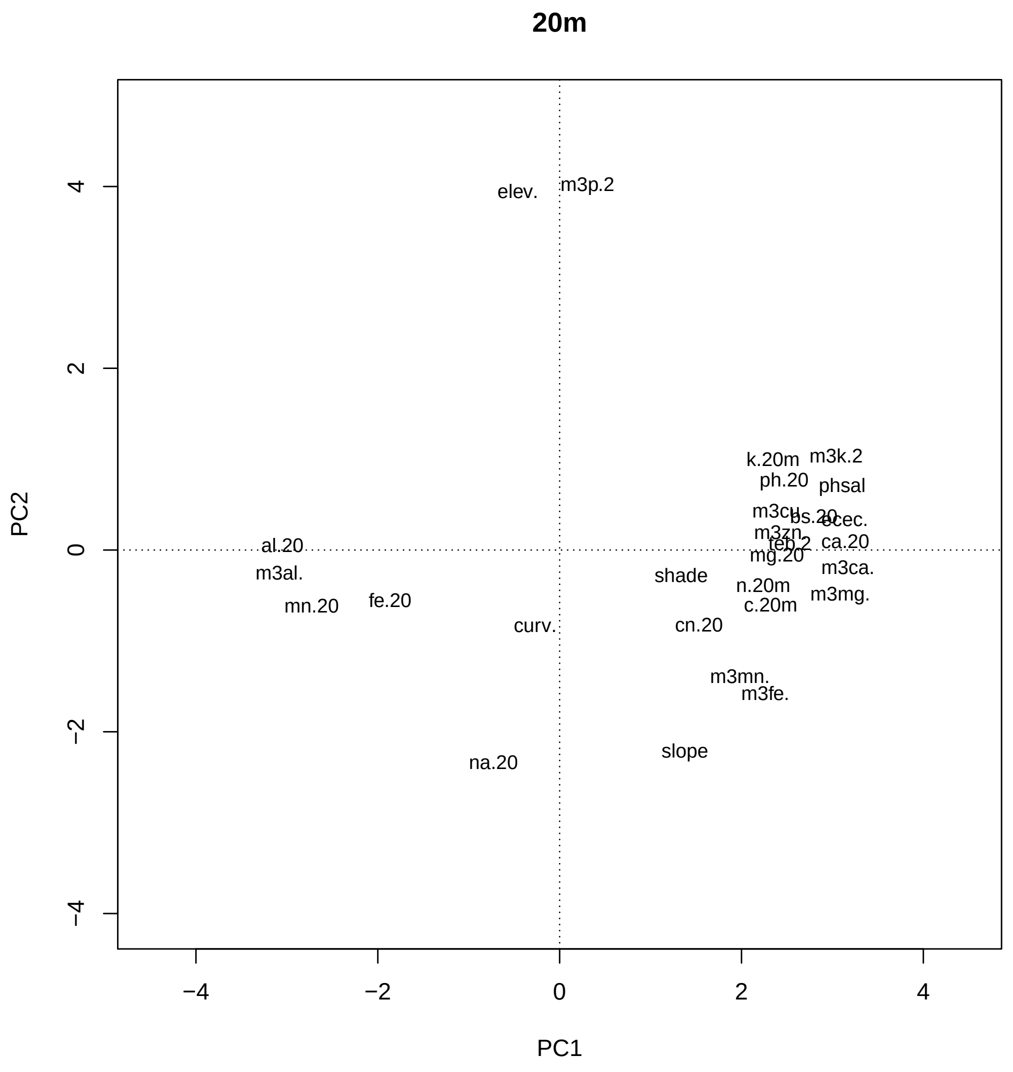

al.20m.mean
22
20
18
16
14
al.20m.sd
0.6
0.4
0.2
bs.20m.mean
95.0
94.5
94.0
93.5
93.0
bs.20m.sd
0.16
0.12
0.08
0.04
c.20m.mean
4.80
4.75
4.70
4.65
c.20m.sd
0.03
0.02
0.01
ca.20m.mean
2400
2300
2200
2100
ca.20m.sd
30
20
10
cn.20m.mean
10.53
10.50
10.47
10.44
cn.20m.sd
0.015
0.010
0.005
curv.20m.mean
0.002
0.001
0.000
−0.001
−0.002
curv.20m.sd
0.004
0.003
0.002
0.001
ecec.20m.mean
15.5
15.0
14.5
14.0
13.5
ecec.20m.sd
0.15
0.10
0.05
elev.20m.mean
150
140
130
elev.20m.sd
2.0
1.5
1.0
0.5
fe.20m.mean
3.0
2.5
2.0
1.5
fe.20m.sd
0.3
0.2
0.1
k.20m.mean
120
115
110
105
k.20m.sd
0.75
0.50
0.25
m3al.20m.mean
820
800
780
760
m3al.20m.sd
1.5
1.0
0.5
m3ca.20m.mean
1700
1600
1500
m3ca.20m.sd
9
6
3
m3cu.20m.mean
5.6
5.5
5.4
5.3
5.2
5.1
m3cu.20m.sd
0.025
0.020
0.015
0.010
0.005
m3fe.20m.mean
121
119
117
115
m3fe.20m.sd
0.8
0.6
0.4
0.2



3.3 Explore env.data (pairwise corr. plots, PCA), e.g. at 100m scale

```r
pdf("pca.env.correct.100.pdf", width=7.5, height=8)
dat <- env.scale.correct.mean[[7]]
colnames(dat) <- substr(colnames(dat),1, 5)
rda.60 <- rda(dat, scale = T)
plot(rda.60, type = "none", main="100m", scaling = 3)
ordipointlabel(rda.60, display = "species", scaling = 3, add = TRUE)
dev.off()

pdf("cor100.correct.pdf", width=25, height=25)
CorTestPlot(dat)
dev.off()
```

Example for environmental PCA at the 20m scale. Covariation among predictors doesnt change much across scales.



```
lapply(env.scale.correct, names)
```

There are different variables in the data sets for the different scales

Extract only those variables that are contained at the smallest spatial scale Extract before the (\.) any character (.) any number of times (*) until the end of the string

```
ind <- sub("\\..*$", "", names(env.scale.correct[[1]]))
```

Kick out variables that do not occur in the 5m scale data set.

```
which(sub("\\..*$", "", names(env.scale.correct[[4]])) %in% ind)

env.scale.correct.2 <- env.scale.correct
for (i in 1:7){
    env.scale.correct.2[[i]] <-
        env.scale.correct[[i]][, which(sub("\\..*$", "", names(env.scale.correct[[i]])) %in% ind)]
}

names(env.scale.correct.2)
```

Select soil types:

```
ind.soil.type <- c("ava","fairchild","marron","swamp")
```

Delete soil type columns (which occur throughout the data) to data

```
env.scale.correct.3 <- env.scale.correct.2

for (i in 1:7){
    env.scale.correct.3[[i]] <-
        env.scale.correct.2[[i]][, -which(sub("\\..*$", "",
                                             names(env.scale.correct.2[[i]])) %in% ind.soil.type)]
}

lapply(env.scale.correct.3, names)
```

Append soil type:

```
env.scale.correct.4 <- env.scale.correct.3
for (i in 1:7){
    env.scale.correct.4[[i]] <-
        cbind(env.scale.correct.3[[i]], env.scale.correct.2[[i]]
              [, which(sub("\\..*$", "", names(env.scale.correct.2[[i]])) %in% ind.soil.type)])
}

lapply(env.scale.correct.4, names)
```

3.4 Extract the mean columnms and select four sets of predictors at each spatial scale:

```
env.scale.correct.mean <- env.scale.correct.4
for (i in 2:7){
    env.scale.correct.mean[[i]] <- env.scale.correct.mean[[i]][,c(seq(1, 56, 2), 57:60)]
}
names(env.scale.correct.mean)
lapply(env.scale.correct.mean, names)
```

3.4.1 1st and 2nd PCA axis on all predictors (27 vars without shade)

```
env.scale.2.mean.PC <- env.scale.correct.mean
names(env.scale.2.mean.PC[[2]])[-26] # exclude shade

pca.test <- prcomp(scale(env.scale.correct.mean[[7]][ ,-26]))
summary(pca.test)
rownames(pca.test$rotation)

for (i in 1:7){
    env.scale.2.mean.PC[[i]] <-
        cbind(env.scale.correct.mean[[i]],
              scores(prcomp(scale(env.scale.correct.mean[[i]][ ,-26])), choices=c(1,2)))
}

dim(env.scale.2.mean.PC[[4]])
```

```r
tail(env.scale.2.mean.PC[[3]][,33:34])
```

3.4.2 Just PC1 (without shade)

```r
tail(env.scale.2.mean.PC[[3]][,33])
```

3.4.3 Just pH

```r
colnames(env.scale.2.mean.PC[[3]][24])
```

3.4.4 Just shade

```r
colnames(env.scale.2.mean.PC[[3]][26])

env.scale.2.mean.PC.correct <- env.scale.2.mean.PC
save(env.scale.2.mean.PC.correct, file = "env.scale.2.mean.PC.correct.Rdata")
```

3.4.5 Generate reduced data set with just 5 columns

```r
env.scale.small <- env.scale.2.mean.PC.correct

for (i in 1:7){
    env.scale.small[[i]] <- env.scale.2.mean.PC.correct[[i]][,c(24,8,26,33,34)]
}
lapply(env.scale.small, names)

save(env.scale.small, file = "env.scale.small.Rdata")
```

3.5 Generate dataset for standard dev. of env. variables

```r
lapply(env.scale.correct.4, names)
env.scale.correct.sd <- env.scale.correct.4

for (i in 2:7){
    env.scale.correct.sd[[i]] <- env.scale.correct.4[[i]][,seq(2, 56, 2)]
}
names(env.scale.correct.sd)

lapply(env.scale.correct.sd, names)

env.scale.correct.sd <- env.scale.correct.sd[-1]

# 1st and 2nd PCA axis on all predictors (27 vars without shade)
env.scale.2.sd.PC <- env.scale.correct.sd

names(env.scale.2.sd.PC[[2]])[-26] # exclude shade
```

3.5.1 Explore standard deviation (sd) of env.data (pairwise corr. plots, PCA)

```
pdf("pca.env.correct.sd.10.pdf", width=7.5, height=8)
dat <- env.scale.correct.sd[[1]]
colnames(dat) <- substr(colnames(dat),1, 5)
rda.60 <- rda(dat, scale = T)
plot(rda.60, type = "none", main="10m", scaling = 3)
ordipointlabel(rda.60, display = "species", scaling = 3, add = TRUE)
dev.off()

pdf("cor10.correct.sd.pdf", width=25, height=25)
CorTestPlot(dat)
dev.off()
```

3.5.2 PCA on sd data

```
pca.test <- prcomp(scale(env.scale.correct.sd[[6]][ ,-26]))
summary(pca.test)
rownames(pca.test$rotation)

for (i in 1:6){
    env.scale.2.sd.PC[[i]] <- cbind(env.scale.correct.sd[[i]],
                                    scores(prcomp(scale(env.scale.correct.sd[[i]][ ,-26])),
                                           choices=c(1,2)))
}

lapply(env.scale.2.sd.PC, names)


dim(env.scale.2.sd.PC[[4]])

tail(env.scale.2.sd.PC[[2]][,29:30])
summary(env.scale.2.sd.PC[[2]][,29:30])

filled.contour((matrix(env.scale.2.sd.PC[[2]][,30], ncol = 25, byrow = T)),
               color.palette = terrain.colors, main = "slope.20m")
                                        # ncol here corresponds to 500m / grain size (short side of bci

env.scale.2.sd.PC.correct <- env.scale.2.sd.PC
save(env.scale.2.sd.PC.correct, file = "env.scale.2.sd.PC.correct.Rdata")

lapply(env.scale.2.sd.PC.correct, names)
```

3.5.3 Generate reduced data set with just 5 columns:

- pH
- elevation
- shade
- PC1
- PC2

The latter two based on all (but not shade) environmental variables, incl. the 4 soil types.

```
env.scale.small.sd <- env.scale.2.sd.PC.correct
lapply(env.scale.small.sd, names)

for (i in 1:6){
    env.scale.small.sd[[i]] <- env.scale.2.sd.PC.correct[[i]][,c(24,8,26,29,30)]
}

lapply(env.scale.small.sd, names)
lapply(env.scale.small.sd, str)

save(env.scale.small.sd, file = "env.scale.small.sd.Rdata")
```

3.5.4 Match demographic rates and trait data (from Joe Wright) with species data (all)

```
traits.new <- read.table("/home/oliver/Dokumente/PhD/PostPhD/Projects/BCI_Coexistence/Data/Traits/
Nadja_Traits/abun/new_fitness_measures_relpopchange.txt", stringsAsFactors = F)
dim(traits.new)
str(traits.new)

names(traits.new)
CorTestPlot(traits.new[,-c(1,7)])
```

Match to species data:

```
index <- which(traits.new$species %in% colnames(grid.213.7[[7]]))
traits213new <- traits.new[index,]

summary(traits213new)

colnames(grid.213.7[[7]])
traits213new$species
match(traits213new$species, colnames(grid.213.7[[7]]))

traits213new2 <- traits213new[,c(1,7,4,5,6,2,3,8:12)]
rownames(traits213new2) <- traits213new2$species
```

3.5.5 Joe Wright's trait data:

```
traits.bci <- read.table("/home/oliver/Dokumente/PhD/PostPhD/Projects/BCI_Coexistence/Data/Traits/
Nadja_Traits/Traits_BCI.txt")
head(traits.bci)
dim(traits.bci)
match(traits213new$species, traits.bci$species)
summary(traits.bci)

library(dplyr)
traits.213.all <- dplyr::inner_join(traits213new2, traits.bci, by = "species")
bci.data$traits.213.all <- traits.213.all[,-13]
rownames(bci.data$traits.213.all) <- bci.data$traits.213.all$species
colnames(bci.data$traits.213.all)[1] <- "sp"
```

```r
save(bci.data, file = "bci.data.Rdata")
```

Complete trait data only are available for 164 out of the 213 species for which complete demographic rate data are available, mainly because data on seed mass for 46 species is missing.

Generate the trait data set based on the set of 164 species for which complete traits information (demographic rates and wright traits) are available:

```r
traits.wright <- bci.data$traits.213.all[!is.na(rowSums(bci.data$traits.213.all[,c(13:20)])),]
bci.data$traits.164.all <- traits.wright
```

4 Estimating species interactions using

After having generated the environmental, abundance (and trait) data sets, species interactions were inferred using the Markov-network approach by Harris 2016 (Ecology). Because the analytical approach by Harris (2016) is intractable for large networks >20 species, and does not account for environmental heterogeneity, I used the approximation approach to estimate interaction strength for the BCI-community containing 302 species. This approach also accounts for abiotic structure in the community data.

Although data on demographic rates and traits are not available for all BCI tree species, interactions were estimated based on the entire community (which in our case was 302 species) for each of the 7 spatial scales (grain sizes): 5m, 10m , 20m, 40m, 60m, 80m, 100m), and three size classes: 1-5cm (dbh), >5-10cm (dbh), >10cm (dbh):

4.1 Generating abundance grids

Although data on demographic rates and traits are not available for all BCI tree species, interactions were estimated based on the entire community (which in our case was 302 species) for each of the 7 spatial scales (grain sizes): 5m, 10m , 20m, 40m, 60m, 80m, 100m), and three size classes: 1-5cm (dbh), >5-10cm (dbh), >10cm (dbh):

4.1.1 Abundance grids for 302 species, across scales

```r
path <- "/home/oliver/Dokumente/PhD/PostPhD/Projects/BCI_Coexistence/Data/Traits/
Nadja_Traits/abundance_gridcells/5m_Steps"

grid.302.7 <- list()

for (j in c("gridcells5_1990","gridcells10_1990","gridcells20_1990","gridcells40_1990",
            "gridcells60_1990","gridcells80_1990","gridcells100_1990")){
    filenames <- list.files(path, pattern=j, full.names=TRUE)
    ldf <- lapply(filenames, read.table)

    grid.302.7 <- c(grid.302.7, (ldf))
}

lapply(grid.302.7, dim)

str(grid.302.7)
names(grid.302.7) <- c("5m","10m","20m","40m","60m","80m","100m")
```

Prune the 305 species data sets to 302 species (40-80m scales):

```
grid.302 <- foreach(i = 1:7) %do% {
    index <- which(toupper(colnames(grid.302.7[[i]])) %in% colnames(grid.302.7[[4]]))
    grid.302.7[[i]] <- grid.302.7[[i]][,index]
}
lapply(grid.302, dim)

names(grid.302) <- c("5m","10m","20m","40m","60m","80m","100m")
colnames(grid.302[[1]]) <- toupper(colnames(grid.302[[1]]))

grid.302 <- grid.302.7
save(grid.302.7, file = "grid.302.7.Rdata")
```

4.1.2 Abundance grids, size classes, across scales (e.g. >10 cm dbh)

```
path <- "/home/oliver/Dokumente/PhD/PostPhD/Projects/BCI_Coexistence/Data/Traits/
Nadja_Traits/abundance_gridcells/SizeClasses"

grid.size.10 <- list()

for (j in c("gridcells5_1990_10cm","gridcells10_1990_10cm","gridcells20_1990_10cm",
            "gridcells40_1990_10cm","gridcells60_1990_10cm","gridcells80_1990_10cm",
            "gridcells100_1990_10cm")){
    filenames <- list.files(path, pattern=j, full.names=TRUE)
    ldf <- lapply(filenames, read.table)
    grid.size.10 <- c(grid.size.10, (ldf))
}

lapply(grid.size.10, dim)
names(grid.size.10) <- c("5m","10m","20m","40m","60m","80m","100m")

grid.full.size.10 <- grid.size.10
save(grid.full.size.10, file = "grid.full.size.10.Rdata")
```

Repeat for size classes `>5-10cm (dbh)` and `>10cm (dbh)`.

Because the estimation of interactions using markov networks is computationally very expensive, particularily for the small spatial scales, the respective calculations were carried out on the EVE HPC, using the following environmental and species data sets:

Environmental data:

- env.scale.small.mean
- env.scale.small.sd

Species abundance grids:

- grid.full.302.7 (302 species)
- grid.full.size.10 (228 species)
- grid.full.size.5 (241 species)
- grid.full.size.1 (281 species)

4.1.3 Stochastic approximation of markov network models

Prior to analyses, I got the following email feedback from Dave Harris (11th Dec. 2015):

"Hi Oliver, thanks for your interest and your question. If I understand correctly, you should be able to do it with a few minor modifications to my code.

- Run the first chunk with no modifications. You might need to install the mistnet package for the `%plus%` function. The easiest way to do that is with the `devtools` package: devtools::install_github("davharris/mistnet")
- load your "x" and "y" data matrices and set n_loc, n_spp, and n_env to match your data (n_loc = nrow(y); n_spp = ncol(y); n_env = ncol(x))
- Skip the second and third chunks, since they involve the generation of a simulated y matrix and you already have all your data.
- Run the fourth chunk ("Calculate sufficient statistics of the data") without modification
- Run the fifth chunk (priors) without modification unless you have prior knowledge about the alpha and beta coefficients
- Remove the lines about r-squared from the next chunk (R-squared can't be calculated unless the "true" coefficients are known) and then run the remaining code. This might take a while, depending on how many iterations you choose and how big your data set is.

When you're done, you should have:

- an object called "alpha_species": this is a vector of intercept terms for each species.
- an object called "alpha_env": this is a matrix of species' responses to the environmental variables (one row per variable, one column per species)
- an object called "beta": this is a symmetric matrix with each species' responses to one another.

I don't think these objects will include names for the species or environmental variables, but they'll match the ordering of the columns in x and y. So the first row of alpha_env will refer to the first column of x, and so on.

Depending on the properties of your data set, the 50,000 iterations might not be enough for convergence. I'd try running it with 25,000 and 100,000 to make sure you get similar answers.

Hope this is useful for you! Let me know if you have any more questions or if you run into any cryptic errors.

I hope you have a great holiday. Say hi to Jon for me.

Dave"

4.1.3.1 General code for the stochastic approximation of markov network models (incl. Dave's comments above)

```r
library(rosalia)
library(mistnet)
library(vegan)
```

Steps according to email by Dave Harris, 11th Dec. 2016:

Run the first chunk with no modifications.

1) convenience function for adding intercepts to each column:

```r
`%plus%` = mistnet:::`%plus%`
logistic = binomial()$linkinv # logistic inverse link
```

Random bernoulli trial

```r
rbern = function(p){rbinom(length(p), size = 1, prob = p)}
```

2) load your "x" and "y" data matrices and set n_loc, n_spp, and n_env to match your data (n_loc = nrow(y); n_spp = ncol(y); n_env = ncol(x))

start with 20m scale since i have the env. variables for that scale:

```r
load("bci.data.Rdata")
```

i) y-matric (species data):

```r
names(bci.data)
dim(bci.data$grid.213[[4]])
```

1250 sites x 213 species

```r
names(bci.data$grid.213)
y <- as.matrix(bci.data$grid.213[[4]])
```

Transform into presence-absence matrix:

```r
y <- decostand(y, "pa")
```

ii) x-matric (environmental data). As did not have the proper soil data at that point, I downloaded some data from David Zeleny's webpage: http://www.davidzeleny.net/anadat-r/doku.php/en:data:bci:script-soil

```r
soil20x20 <-
read.delim('http://www.davidzeleny.net/anadat-r/lib/exe/
fetch.php?media=data:bci-soil-20x20.txt')
names(soil20x20[,c(8,11,13:15)])
soil.13 <- soil20x20[,c(3:15)]
env = true_env <- x <- soil.13 <- scale(soil20x20[,c(3:15)])
```

Set some parameters

```r
set.seed(1)
n_spp = ncol(y)    # number of species
n_loc = nrow(y)     # number of locations
n_env = ncol(x)      # number of environmental predictors
n_gibbs = 5000  # number of Gibbs sampling iterations
```

3) Skip the second and third chunks, since they involve the generation of a simulated y matrix and you already have all your data.
4) Run the fourth chunk ("Calculate sufficient statistics of the data") without modification

Calculate sufficient statistics of the data

```r
y_stats = crossprod(y)
y_env_stats = t(true_env) %*% y
```

Initialize the simulated landscape for stochastic approximation

```r
y_sim = matrix(0.5, nrow = nrow(y), ncol = ncol(y))
```

In this example, the true state of the environment is known without error

```r
env = true_env
```

Initialize species' responses to environment at 0. Also initialize the delta (change in parameter values from the previous optimization iteration) to zero, since no optimization has occurred yet

```r
alpha_env = delta_alpha_env = matrix(0, nrow = n_env, ncol = n_spp)
```

Initialize species' intercepts to match observed occurrence rates plus a small amount of regularization

```r
alpha_species = qlogis((colSums(y) + 1) / (nrow(y) + 2))
```

Initialize the deltas for the intercepts to zero

```
delta_alpha_species = rep(0, n_spp)
```

Initialize pairwise interactions and deltas to zero

```
beta = delta_beta = matrix(0, nrow = n_spp, ncol = n_spp)
```

Overall alpha depends on alpha_species and alpha_env. Will be filled in later, so can initialize it with zeros no delta alpha to initialize b/c alpha not optimized directly.

```
alpha = matrix(0, nrow = n_spp, ncol = n_spp)
```

5.) Run the fifth chunk (priors) without modification unless you have prior knowledge about the alpha and beta coefficients.

Very weak priors on alpha terms, somewhat stronger on beta terms.

```
alpha_env_prior = rosalia::make_logistic_prior(scale = 2)$log_grad
alpha_species_prior = rosalia::make_logistic_prior(scale = 2)$log_grad
beta_prior = rosalia::make_logistic_prior(scale = 0.5)$log_grad
```

6) Remove the lines about r-squared from the next chunk (R-squared can't be calculated unless the "true" coefficients are known) and then run the remaining code. This might take a while, depending on how many iterations you choose and how big your data set is.

```
initial_learning_rate = 1 # step size at start of optimization
maxit = 25000             # Number of rounds of optimization # try 50000 and 100000 as well
start_time = as.integer(Sys.time())
```

Record the timing history in this vector

```
times = integer(maxit)

for(i in 1:maxit){
  ##############################
  # Gibbs sampling for predicted species composition
  ##############################

  # Update alpha
  alpha = env %*% alpha_env %plus% alpha_species

  # Sample entries in y_sim from their conditional
  # distribution (Gibbs sampling)
    for(j in sample.int(n_spp)){
    y_sim[,j] = rbern(logistic(alpha[ , j] + y_sim %*% beta[ , j]))
  }
```

Stochastic approximation for updating alpha and beta

Update learning rate and momentum

```
learning_rate = initial_learning_rate * 1000 / (998 + 1 + i)
momentum = .9 * (1 - 1/(.1 * i + 2))
```

Calculate sufficient statistics

```
  y_sim_stats = crossprod(y_sim)
  y_sim_env_stats = t(env) %*% y_sim
```

Calculate the gradient with respect to alpha and beta. Gradients are differences in sufficient statistics plus prior gradients, all divided by the number of locations.

```
  stats_difference = y_stats - y_sim_stats
  beta_grad = (stats_difference + beta_prior(beta)) / n_loc

  alpha_species_grad = (
    diag(stats_difference) +
      alpha_species_prior(alpha_species)
  ) / n_loc
  diag(beta_grad) = 0 # beta_ii is 0 by convention
  y_env_difference = y_env_stats - y_sim_env_stats
  alpha_env_grad = (y_env_difference +
                    alpha_env_prior(alpha_env))  / n_loc
```

Calculate parameter updates: gradient times learning rate plus momentum times delta.

```
  delta_beta = beta_grad * learning_rate + momentum * delta_beta
  delta_alpha_species = alpha_species_grad * learning_rate +
    momentum  * delta_alpha_species
  delta_alpha_env = alpha_env_grad * learning_rate +
    momentum  * delta_alpha_env
```

Add the deltas to the previous parameter values.

```
  beta = beta + delta_beta
  alpha_species = alpha_species + delta_alpha_species
  alpha_env = alpha_env + delta_alpha_env

  times[i] = as.integer(Sys.time()) - start_time
}
```

Running time for 25000 iterations: 206 min

Save results:

```
MarkovNet213_20_env13_25000 <- list("alpha_species"=alpha_species, "alpha_env"=alpha_env,
                                    "beta"=beta)

save(MarkovNet213_20_env13_25000, file = "MarkovNet213_20_env13_25000.Rdata")
```

5 Scripts for analysis on EVE

5.1 bci-markov-submit-wrapper-correct-full-long.sh

```
#!/bin/bash

ITERATIONS=$1
JOB_NAME_SUFFIX=$2

for grid in /data/idiv_chase/OliverPurschke/bci-input/grid.full.*.Rdata ; do
  for env in /data/idiv_chase/OliverPurschke/bci-input/env.*.Rdata ; do
    for columns in "1" "2" "3" "c(1,2)" "4" "c(4,5)" ; do

      case $env in
    *.mean*)
          lenind=0
          ;;

    *.sd*)
          lenind=1
          ;;
      esac

      qsub -N bci.markov-$JOB_NAME_SUFFIX ~/bci/bci-markov-submit-correct-full-long.sh $grid $env $colu
    done
  done
done
```

To submit, use:

```
bash bci-markov-submit-wrapper-correct-full-long.sh 1000 analyse-1
```

5.2 bci-markov-submit-correct-full-long.sh

```
#!/bin/bash

#$ -N bci.markov

#$ -o /work/$USER/$JOB_NAME-$JOB_ID.out
#$ -e /work/$USER/$JOB_NAME-$JOB_ID.err

#$ -l h_rt=400:00:00
#$ -l h_vmem=2G
##$ -l avx # avx is cpu feature that is only available in the newer generation of the hardware

#$ -pe smp 2

export MC_CORES=${NSLOTS:-1}

module load R

Rscript ~/bci/bci.markov.correct.full.r "$@" /work/purschke/bci-output/$JOB_NAME-$JOB_ID-$(basename $1
```

5.3 bci.markov.correct.full.r

```
args <- commandArgs(trailingOnly = TRUE)
grid <- args[1] # e.g. "grid.213.7.Rdata"
env.name <- args[2] # e.g. "env.scale.2.sd.PC.Rdata"
columns <- args[3] # Arguments have type character, so coerce to numeric
# see https://wiki.csiro.au/display/ASC/Run+R+script+as+a+batch+job+on+a+Linux+cluster
len.ind <- as.numeric(args[4]) # length index: 0 if env. data has all scales; 1 if the smallest scale i
iters <- as.numeric(args[5]) # markov iterations, e.g. 25000
outpath <- args[6]

### for an example:
```

```r
## DO NOT RUN:
# 1) grid <- "grid.213.7.Rdata"
# 2) env.name <- "env.scale.2.sd.PC.Rdata"
# 3) columns <- "-c(26,29,30)"
# 4) len.ind <- as.numeric("1")
# 5) iters <- as.numeric("10")

# s <- 5
####

library(vegan)
library(rosalia)
library(mistnet)
library(doParallel)
# Find out how many cores are available (if you don't already know)
detectCores()
# Create cluster with desired number of cores
cl <- makeForkCluster() # !!! set to 2, since the first process (20,000 sites)takes ages on one core, w
# Register cluster
registerDoParallel(cl)
# Find out how many cores are being used
getDoParWorkers()

########
# # outside the loop
`%plus%` = mistnet:::`%plus%`
logistic = binomial()$linkinv # logistic inverse link
rbern = function(p){rbinom(length(p), size = 1, prob = p)}

# load("env.scale.2.sd.PC.Rdata") ### !!! change to "mean"
env2 <- get(load(env.name))
# load("grid.213.7.Rdata") # !!! change to "size"
grid2 <- get(load(grid))

######################
out <- foreach (s = 1:length(env2), .packages=c("rosalia", "mistnet","vegan")) %dopar% {
# !!! change to "1:7" if means are used; 1:6 if sd is used
y <- as.matrix(grid2[[s+len.ind]]) # !!! for "sd" change to [[s+1]]...
# y <- as.matrix(grid.213.7[[s]])                                    # !!! change to just "s", if mean
# transform into presence-absence matrix:
y <- decostand(y, "pa")

####################################################################
####### to be changed depending on which env. data should be used
env = true_env <- x <- soil.13 <- scale(env2[[s]][,eval(parse(text=columns))]) #!!!! change to the righ
    # 1 pH
    # 2 elevation
    # 3 shade
    # c(1,2) pH & elevation
    # 4 PC1
    # c(4,5) PC1 & PC2
####################################################################
```

```
set.seed(1)
n_spp = ncol(y)   # number of species
n_loc = nrow(y)    # number of locations
n_env = ncol(x)     # number of environmental predictors

n_gibbs = 5000  # number of Gibbs sampling iterations
######################

y_stats = crossprod(y)
y_env_stats = t(true_env) %*% y
y_sim = matrix(0.5, nrow = nrow(y), ncol = ncol(y))
env = true_env
alpha_env = delta_alpha_env = matrix(0, nrow = n_env, ncol = n_spp)
alpha_species = qlogis((colSums(y) + 1) / (nrow(y) + 2))
delta_alpha_species = rep(0, n_spp)
beta = delta_beta = matrix(0, nrow = n_spp, ncol = n_spp)
alpha = matrix(0, nrow = n_spp, ncol = n_spp)

alpha_env_prior = rosalia::make_logistic_prior(scale = 2)$log_grad
alpha_species_prior = rosalia::make_logistic_prior(scale = 2)$log_grad
beta_prior = rosalia::make_logistic_prior(scale = 0.5)$log_grad

initial_learning_rate = 1 # step size at start of optimization

###############################################
### needs to be changed
maxit = iters
# !!! 25000, Number of rounds of optimization # try 50000 and 100000 as well
########################################
start_time = as.integer(Sys.time())
times = integer(maxit)
        for(i in 1:maxit){
  ##############################
  # Gibbs sampling for predicted species composition
  ##############################

  # Update alpha
  alpha = env %*% alpha_env %plus% alpha_species

  # Sample entries in y_sim from their conditional
  # distribution (Gibbs sampling)
    for(j in sample.int(n_spp)){
    y_sim[,j] = rbern(logistic(alpha[ , j] + y_sim %*% beta[ , j]))
  }

  ##############################
  # Stochastic approximation for updating alpha and beta
  ##############################

  # Update learning rate and momentum

    learning_rate = initial_learning_rate * 1000 / (998 + 1 + i)
 momentum = .9 * (1 - 1/(.1 * i + 2))
```

```
  # Calculate sufficient statistics
  y_sim_stats = crossprod(y_sim)
  y_sim_env_stats = t(env) %*% y_sim


  stats_difference = y_stats - y_sim_stats
  beta_grad = (stats_difference + beta_prior(beta)) / n_loc

  alpha_species_grad = (
    diag(stats_difference) +
      alpha_species_prior(alpha_species)
  ) / n_loc
  diag(beta_grad) = 0 # beta_ii is 0 by convention
  y_env_difference = y_env_stats - y_sim_env_stats
  alpha_env_grad = (y_env_difference +
                    alpha_env_prior(alpha_env))  / n_loc


  delta_beta = beta_grad * learning_rate + momentum * delta_beta
  delta_alpha_species = alpha_species_grad * learning_rate +
    momentum  * delta_alpha_species
  delta_alpha_env = alpha_env_grad * learning_rate +
    momentum  * delta_alpha_env

  beta = beta + delta_beta
  alpha_species = alpha_species + delta_alpha_species
  alpha_env = alpha_env + delta_alpha_env
  times[i] = as.integer(Sys.time()) - start_time
}
        list("alpha_species"=alpha_species, "alpha_env"=alpha_env, "beta"=beta)

}

##  give name here
#save(out, file = paste(grid,env.name,columns,iters,"Rdata", sep = "."))
save(out, file = outpath)
```

6 Grand analysis function

```
mn.path <- "/home/oliver/Dokumente/PhD/PostPhD/Projects/BCI_Coexistence/Results/MarkovNetworks/"
traits <- bci.data$traits.164.all
phylo <- bci.data$phy.164
# only in some cases (traits.213.all): give short names as rownames:
rownames(traits) <- bci.data$traits.164.all$sp

i <- "size.1_"
v <- "(4,5)_"
s <- 1
t <-  "1:3"

mn.list <- list.files(mn.path, pattern=glob2rx(paste0("*grid.full","*",i,"*mean*",v,"*100000*")))
```

```
                                         # to be changed if sd or 100000 iterations are used
mn.list

# start loop
library(doParallel)
out <- foreach(i = c("302.7", "size.1_", "size.5_", "size.10_")) %:%
    foreach(v = c("_1_", "_2_", "_3_", "(1,2)_", "_4_", "(4,5)_"), .combine = rbind) %do% {

        mn.path <- "/home/oliver/Dokumente/PhD/PostPhD/Projects/BCI_Coexistence/Results/
MarkovNetworks/"
        traits <- bci.data$traits.164.all
        phylo <- bci.data$phy.164
# only in some cases (traits.164.all): give short names as rownames:
        rownames(traits) <- bci.data$traits.164.all$sp

        mn.list <- list.files(mn.path, pattern=glob2rx(paste0("*grid.full","*",i,"*mean*",
                                                              v,"*100000*")))
                                         # to be changed if sd or 100000 iterations are used
        mn.list
        mn.list <- get(load(paste0(mn.path, mn.list)))
                                         # match cooccurence matrices to traits
        coocc <- foreach(s = 1:length(mn.list)) %do% {
            beta <- mn.list[[s]]$beta
            rownames(beta) <- toupper(rownames(beta))
            colnames(beta) <- toupper(colnames(beta))
            index.beta <- which(colnames(beta) %in% rownames(traits))
            beta.small <- beta[index.beta, index.beta]
            index.traits <- which(rownames(traits) %in% rownames(beta.small))
            traits.small <- traits[index.traits,]
            mat.dis <- match.comm.dist(comm = t(traits.small[,c(3:dim(traits.small)[2])]), dis = as.dis
            }
        names(coocc) <- c("5m", "10m", "20m", "40m", "60m", "80m", "100m")

        ## to be deleted later on: some extra stuff for the veech analysis
        #traits <- bci.data$traits.213.all[,c(3:dim(bci.data$traits.213.all)[2])]
        #coocc <- veech.ses
        ######################

                                         # match traits to cooccurrence
        beta <- mn.list[[1]]$beta
        rownames(beta) <- toupper(rownames(beta))
        colnames(beta) <- toupper(colnames(beta))
        index.beta <- which(colnames(beta) %in% rownames(traits))
        beta.small <- beta[index.beta, index.beta]
        index.traits <- which(rownames(traits) %in% rownames(beta.small))
        traits.small <- traits[index.traits,]
        traits <- t(match.comm.dist(comm = t(traits.small[,c(3:dim(traits.small)[2])]), dis = coocc[[1]
        dim(traits)

#################################
# for the demographic rates and wright traits:
#################################
#1 "1:3": "demo.rec.light.mean2"    "demo.growth.light.mean2" "demo.mort.light.mean2"
```

```
#2 "4:5": "demo.growth.mean2"  "demo.mort.mean2"
#3 "7": "fitnessLogratio"
#4 "10": "relpopchange"
#5 "19": "fit_gro.mor_ratio"
#6 "20": "Niche.PC1"
#7 "23": "Fit.PC1"
#8 11: "WoodDensity"
#9 12: "MAXHEIGHT"
#10 13 "SeedMass"
#11 14: "LMA"
#12 15: "LeafArea"
#13 25: "NP.PC1"
#14 27: "Wright.8.PC1"

        trait.dist <- foreach(i = c("1:3", "4:5", "7", "20", "23",
                                    "11", "12", "13", "14", "15", "25", "27",
                                    "27:34",
                                    "35", "36",
                                    "37",
                                    "37:44")) %do% {
            dist(traits[,eval(parse(text=i))])
        }
        #lapply(trait.dist, range)
        names(trait.dist) <- c("Niche", "Fitness", "Fit.Log.Ratio", "Niche.PC1", "Fit.PC1",
                               "WoodDensity", "MAXHEIGHT", "SeedMass", "LMA", "LeafArea", "NP.PC1", "Wri
                               "Wright.8.PC1.8",
                               "SeedMass.log", "LeafArea.log",
                               "wright.8.trans.PC1",
                               "wright.8.trans.PC1.8")

##################################
# just for demo traits:
##################################
#1 "1:3": "demo.rec.light.mean2"    "demo.growth.light.mean2" "demo.mort.light.mean2"
#2 "4:5": "demo.growth.mean2"  "demo.mort.mean2"
#3 "7": "fitnessLogratio"
#4 "10": "relpopchange"
#5 "19": "fit_gro.mor_ratio"
#6 "20": "Niche.PC1"
#7 "23": "Fit.PC1"

#        trait.dist <- foreach(t = c("1:3", "4:5", "7", "10", "19", "20", "23")) %do% {
#            dist(traits[,eval(parse(text=t))])
#        }
        #lapply(trait.dist, range)
#        names(trait.dist) <- c("Niche", "Fitness", "Fit.Log.Ratio", "Rel.pop.change", "Fit.gro.mor.rat

# scale the data for multivariate analysis:
#trait.dist.scale <- lapply(trait.dist, scale)
#lapply(trait.dist.scale, mean)
#lapply(trait.dist.scale, sd)

# dis.1 <- dist(traits[,1:3])
```

```
#dis.2 <- dist(traits[,4:5])

# do names in coocc and trait distance matrices match up?:
#match(attr(coocc[[1]], "Labels"), attr(trait.dist[[1]], "Labels"))

#######################################
# add phylogenetic distance matrix
#######################################

# to match traits with phylogeny, give full species names as rownames to trait data:
#str(traits)
#class(traits)
        traits.2 <- traits
        traits.2 <- as.data.frame(traits.2)
        traits.2$sp <- rownames(traits.2)

#names(bci.data$traits.164.all)

# 17-17.30: match phylo with traits
        library(dplyr)
        traits.2.match <- inner_join(traits.2, bci.data$traits.164.all[,c(1:2)])
        rownames(traits.2.match) <- traits.2.match$species

        traits <- traits.2.match[, -c(25:26)]

        mat <- match.phylo.data(phy = bci.data$phy.164, data = traits)
        #mat$phy
#
        phy.mat <- as.dist(cophenetic(mat$phy))

        phy.sort <- match.comm.dist(comm = t(traits[,c(3:dim(traits)[2])]), dis = phy.mat)$dis

#match(attr(coocc[[1]], "Labels"), attr(phy.sort, "Labels"))
#match(attr(trait.dist[[1]], "Labels"), attr(phy.sort, "Labels"))

        trait.dist.phy <- trait.dist
        trait.dist.phy$phylo <- phy.sort

#trait.dist.phy.scale <- trait.dist.scale
#trait.dist.phy.scale$phylo <- scale(phy.sort)

#lapply(trait.dist.phy.scale, mean)
#lapply(trait.dist.phy.scale, sd)


## univariate regression analysis

        coocc <- coocc
        traits <- trait.dist.phy # for veech, just use "trait.dist" here

        ####################################################
        ## the following chunk might be deleted later on
        # put transformation here:
```

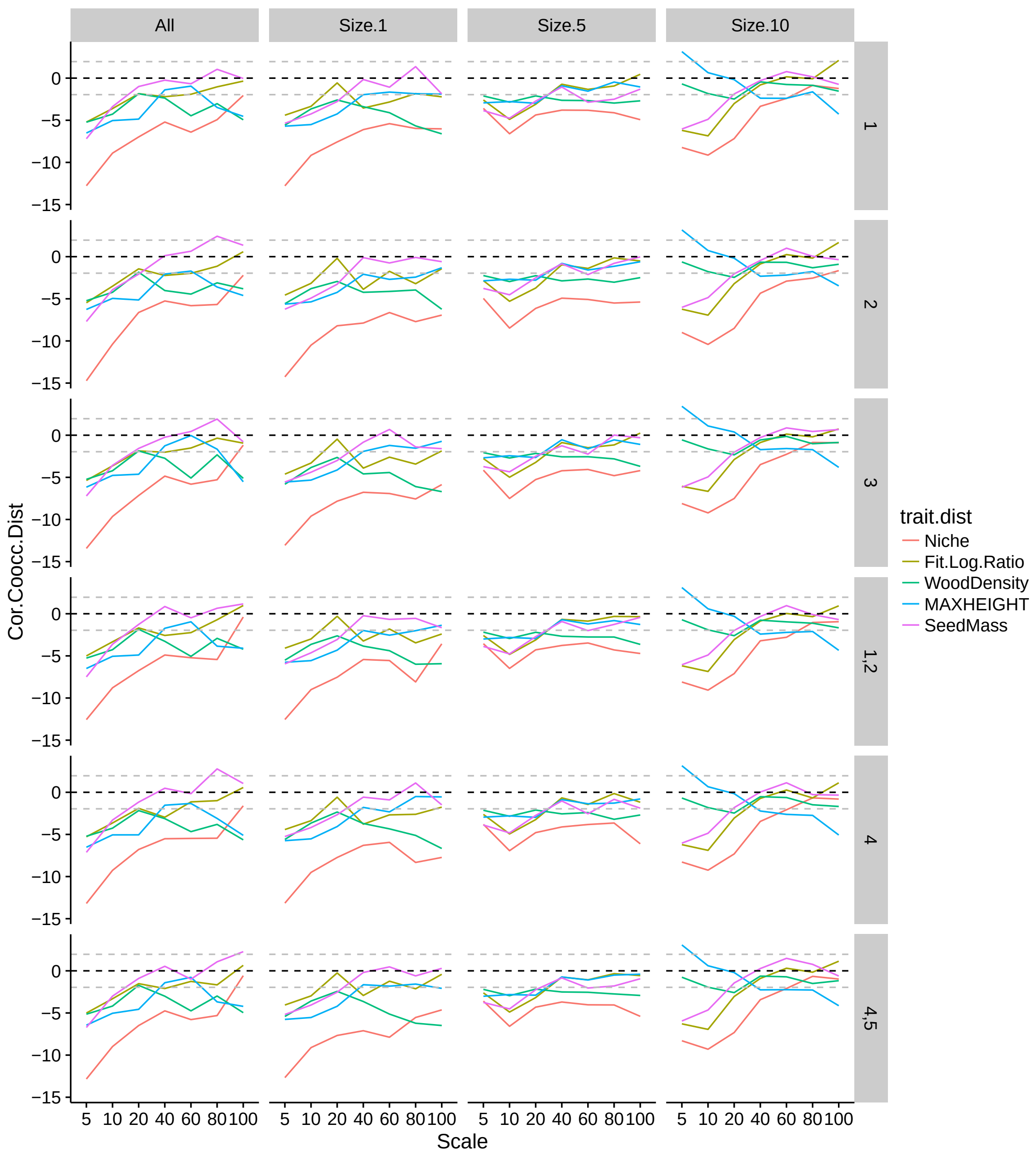
```
        # log-transform seed mass and leaf area, sqrt-transform the rest:
        #names(traits)[c(8,10)]

        for(i in c(1:7,9,11:18)){
            traits[[i]] <- sqrt(traits[[i]])
        }

        for(i in c(8,10)){
            traits[[i]] <- log(traits[[i]]+0.0000001) # there are zero distances for seed mass
        }

        #########################################

    # include the data matching cleaning and matching bits in here
            modelcoef <- foreach (s = 1:length(coocc)) %:%
                foreach(t = 1:length(traits), .combine = rbind) %do% {
                    round(summary(lm(coocc[[s]] ~ traits[[t]]))$coefficients[2,], 4)
                    #rownames(res) <- names(traits)

            }
            #rownames(res) <- names(traits)

       names(modelcoef) <- names(coocc)
       ldply(modelcoef)
    }

out
```

6.1 Assign names to output from grand analysis function

```
names(out) <- c("302.7", "size.1", "size.5", "size.10")
out.2 <- ldply(out)

nam <- expand.grid(
    trait.dist = c("Niche", "Fitness", "Fit.Log.Ratio", "Niche.PC1", "Fit.PC1",
                              "WoodDensity", "MAXHEIGHT", "SeedMass", "LMA", "LeafArea", "NP.PC1", "Wri
                              "Wright.8.PC1.8",
                              "SeedMass.log", "LeafArea.log",
                              "wright.8.trans.PC1",
                              "wright.8.trans.PC1.8","Phylo"),
    scale = c("5", "10", "20", "40", "60", "80", "100"),
    env = c("1", "2", "3", "1,2", "4", "4,5"),
    size = c("All", "Size.1", "Size.5", "Size.10")
)

out.3 <- cbind(nam, out.2)
head(out.3)
out.4 <- out.3[,-c(5:7)]

str(out.4)
names(out.4)[5] <- "t.val"
```

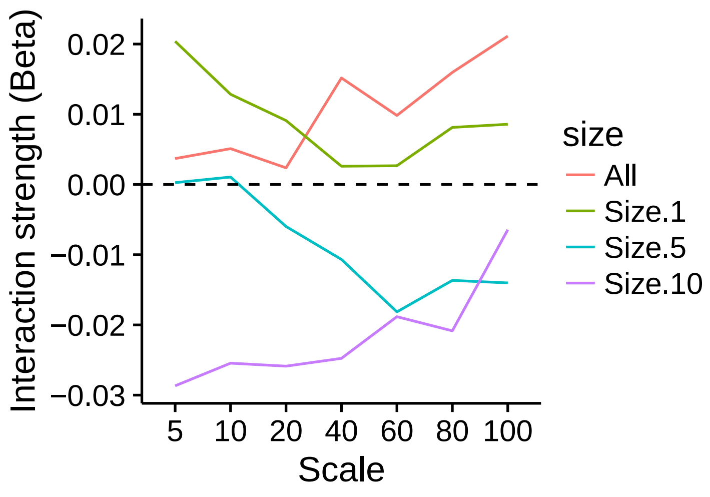
6.2 Plot correlation between pairwise interactions and demographic rate (and trait) differences

```
sub.5 <- subset(out.4, trait.dist %in% c("Niche", "Fit.Log.Ratio", "WoodDensity", "MAXHEIGHT", "SeedMas

sub.6 <- subset(out.4, trait.dist %in% c("LMA", "LeafArea", "NP.PC1", "Wright.8.PC1", "Wright.8.PC1.8")

sub.7 <- subset(out.4, trait.dist %in% c("SeedMass.log", "LeafArea.log", "wright.8.trans.PC1", "wright.

pdf("Cor.Coocc.Dist.wright.7.trans.100000.pdf", width = 10, height = 11)
ggplot(sub.7, aes(scale, t.val, color = trait.dist, group = trait.dist)) +
    #geom_point() +
    geom_line() +
    facet_grid(env ~ size, margins=F) +
    labs(y="Cor.Coocc.Dist", x= "Scale") +
    #scale_colour_manual(values=gs.pal(8)) +
    geom_hline(yintercept=c(0), color = c("black"), linetype="dashed") +
    geom_hline(yintercept=c(-1.96, 1.96), color = c("grey"), linetype="dashed") +
    theme_bw() + theme(panel.border = element_blank(), panel.grid.major = element_blank(),
                       panel.grid.minor = element_blank(),
                       axis.line = element_line(colour = "black"))
dev.off()
```



6.3 Plot scale dependent changes in interaction strength

```
inter <- foreach(i = c("302.7", "size.1_", "size.5_", "size.10_")) %:%
    foreach(v = c("_1_", "_2_", "_3_", "(1,2)_", "_4_", "(4,5)_"), .combine = rbind) %do% {

        mn.path <- "/home/oliver/Dokumente/PhD/PostPhD/Projects/BCI_Coexistence/Results/MarkovNetworks/
        mn.list <- list.files(mn.path, pattern=glob2rx(paste0("*grid.full","*",i,"*mean*",v,"*50000*")))
                                 # to be changed if sd or 100000 iterations are used
        mn.list
        mn.list <- get(load(paste0(mn.path, mn.list)))

        coocc <- foreach(s = 1:length(mn.list)) %do% {
            as.dist(mn.list[[s]]$beta)
            }
        names(coocc) <- c("5m", "10m", "20m", "40m", "60m", "80m", "100m")
        ldply(lapply(coocc, sd))
        }

names(inter) <- c("All", "Size.1", "Size.5", "Size.10")
inter.2 <- ldply(inter)

nam.inter <- expand.grid(
    scale = c("5", "10", "20", "40", "60", "80", "100"),
    env = c("1", "2", "3", "1,2", "4", "4,5"),
    size = c("All", "Size.1", "Size.5", "Size.10")
)

inter.3 <- cbind(nam.inter, inter.2)
head(inter.3)

inter.4 <- inter.3[,-c(4)]
names(inter.4)[4] <- "Beta"


pdf("Inter.scales.env.sd.pdf", width = 6, height = 5)
ggplot(inter.4, aes(scale, Beta, color = env, group = env)) +
    #geom_point() +
    geom_line() +
    facet_wrap(~size) +
    labs(y="Interaction strength (Beta)", x= "Scale") +
    #scale_colour_manual(values=gs.pal(8)) +
    geom_hline(yintercept=c(0), color = c("black"), linetype="dashed")
    #geom_hline(yintercept=c(-1.96, 1.96), color = c("grey"), linetype="dashed")
dev.off()
```

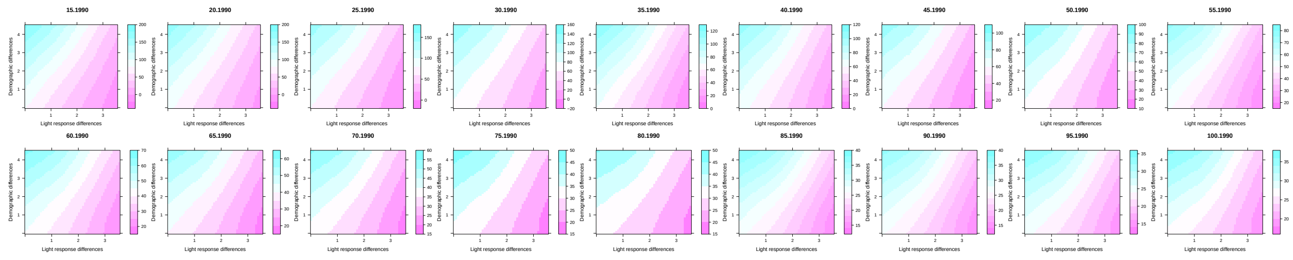


7 Some additional figures

7.1 Contour plots of cooccurrence vs. niche and fitness differences across scales

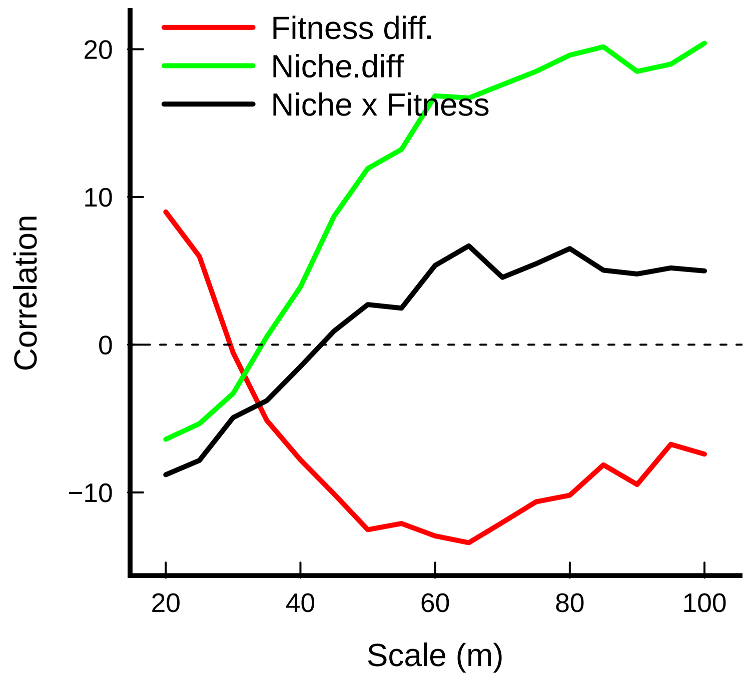
```
bci.data$traits.abu.213 <- traits.abu.213

colnames(bci.data$traits.abu.213)

# for "fitnessSMA"
fit.dist <- dist(bci.data$traits.abu.213[,8])

####### continue here ###!!1
# for "logratio"
fit.dist <- dist(bci.data$traits.abu.213[,9])
str(fit.dist)

# resp <- check.213.df # to be changed accordingly
niche <- bci.data$niche.dist.213.df[,1]
fitness <- (as.vector(fit.dist))

library(doParallel)
# Find out how many cores are available (if you don't already know)
detectCores()
# Create cluster with desired number of cores
cl <- makeCluster(3)

# Register cluster
registerDoParallel(cl)

# Find out how many cores are being used
getDoParWorkers()


for (j in 1:7){ # loop across cooc-response variable (7 variants)
    coocc <- bci.data$coocc.213[[j]]
    figlist <- foreach (i = 1:18, combine = list, .packages='lattice') %dopar% { # loop across scales (
        resp <-coocc[,i]
        phylodistance <- loess(resp ~ niche*fitness, span = 1, degree = 1)
        n.marginal <- seq(min(niche), max(niche), length.out = 50)
        f.marginal <- seq(min(fitness), max(fitness), length.out = 50)
        nf.marginal <- list(niche = n.marginal, fitness = f.marginal)
        grid <- expand.grid(nf.marginal)
        grid[, "fit"] <- c(predict(phylodistance, grid))
        fig <- lattice::contourplot(fit ~ niche * fitness, data = grid, cuts = 10, region = TRUE, label
            #,
            #panel=function(...){
            #panel.contourplot(...)
            #grid.points(niche, fitness, pch=1, size=unit(.001, "char"))
          #}
           #,
            #       panel = function(x,y,z,...){
            #  panel.contourplot(x,y,z,...)
            #  panel.abline(0,1,lwd=1,col="blue")
```

```
                                             #       }
                                        )
    #fig

# pdf(file = paste(paste("Check.213.", i, sep=""), "pdf", sep="."), width = 5, height = 4.6, pointsize=
#jpeg(filename = paste(paste("Coocc.check.wo.rec.a.spat.", i, sep=""), "jpeg", sep="."), width = 240, h
#plot(fig)
#dev.off()
                                             #list(fig)
        fig
    }

    pdf(file = paste(paste("Coocc.213.fitLogRatio", j, sep=""), "pdf", sep="."), width = 41, height = 8
    grid.arrange(figlist[[1]],figlist[[2]],figlist[[3]],figlist[[4]],figlist[[5]],figlist[[6]],figlist[
    dev.off()
}
```



7.2 Plotting correlation between cooccurrence vs. niche and fitness differences across scales

```
niche.fit.scaled <- as.data.frame((cbind(niche, fitness)))

modelcoef <- NA

for (i in 1:7){
resp.vec <- bci.data$check.205.ses.1s[,i]
mod <- lm(resp.vec ~ niche*fitness, data = niche.fit.scaled)
modelcoef <- rbind(modelcoef, melt(summary(mod)$coefficients))
}
modelcoef <- modelcoef[-1,]
modelcoef$scale <- rep(seq(10,100,5), each=16)
colnames(modelcoef)[c(1:2)] <- c("variable", "coefficient")
str(modelcoef)

modelcoef$variable <- as.character(modelcoef$variable)
modelcoef$coefficient <- as.character(modelcoef$coefficient)

 ##

modelcoef.estim <- modelcoef[modelcoef$coefficient=="t value",]
modelcoef.estim.2 <-
modelcoef.estim[modelcoef.estim$variable!="(Intercept)",]
modelcoef.estim.2 <-
modelcoef.estim.2[modelcoef.estim.2$variable!="niche.vec:fit.vec",]

modelcoef.estim.3 <- modelcoef.estim.2[,-2]
# scale >15
modelcoef.estim.4 <- modelcoef.estim.3[modelcoef.estim.3$scale > 15, ]
modelcoef.estim.4$variable <- as.factor(modelcoef.estim.4$variable)
modelcoef.estim.4$value <- -(modelcoef.estim.4$value)
modelcoef.estim.5 <- modelcoef.estim.4

# rerun for t-value

# do the lattice plot
# include interaction
div <- xyplot(value ~ scale, groups=variable, data =
modelcoef.estim.4, type = "l", lty = c(1), par.settings =
list(axis.line = list(col = 0)),scales=list(col=1,tck=c(-1,0)),  #
remove top and right axes
              panel=function(...){
                lims <- current.panel.limits()
                panel.xyplot(...)
                panel.abline(h=lims$ylim[1],v=lims$xlim[1], lwd = 4.5)
                panel.abline(h=0,lwd=1, lty=2, col="black")
              },
              layout.heights=list(axis.xlab.padding = 1),
lwd = 2.5, col = c("red","green","black"), xlab = "Scale (m)", ylab =
"Correlation", key=list(space="inside",  between = 1, padding.text =
2, just = c(.6, .5), columns = 1, lines = list(lty = c(1), lwd = 2.5,
col = c("red","green","black")),text = list(c("Fitness diff.",
"Niche.diff", "Niche x Fitness"))))
plot(div)


pdf(file = "Scales.t_value.inter.PCA1-3.coef.scale.pdf",width = 4.2, height = 4, pointsize=12)
# postscript(file = "BCI.DivergenceSize.eps",width = 6, height = 6.5,
paper = "special", onefile = FALSE, horizontal = FALSE, pointsize=12)
plot(div)
dev.off()
```



8 R-settings

```r
sessionInfo()
```

```
## R version 3.4.1 (2017-06-30)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 14.04.5 LTS
##
## Matrix products: default
## BLAS: /usr/lib/openblas-base/libopenblas.so.0
## LAPACK: /usr/lib/lapack/liblapack.so.3.0
##
## locale:
##  [1] LC_CTYPE=de_DE.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=de_DE.UTF-8        LC_COLLATE=de_DE.UTF-8
##  [5] LC_MONETARY=de_DE.UTF-8    LC_MESSAGES=de_DE.UTF-8
##  [7] LC_PAPER=de_DE.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=de_DE.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] parallel  stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] rmarkdown_1.6     ggplot2_2.2.1     adephylo_1.1-10
##  [4] ade4_1.7-8        picante_1.6-2     nlme_3.1-131
##  [7] gridExtra_2.2.1   phytools_0.6-20   maps_3.2.0
## [10] ape_4.1           mistnet_0.3.0     rosalia_0.1.0
## [13] dplyr_0.7.2.9000  doParallel_1.0.10 iterators_1.0.8
## [16] foreach_1.4.3     vegan_2.4-4       lattice_0.20-35
## [19] permute_0.9-4     knitr_1.17
##
## loaded via a namespace (and not attached):
##  [1] bold_0.5.0                gmodels_2.16.2
##  [3] progress_1.1.2            httr_1.3.1
##  [5] rprojroot_1.2             numDeriv_2016.8-1
##  [7] backports_1.1.0           tools_3.4.1
##  [9] R6_2.2.2                  DBI_0.7
## [11] lazyeval_0.2.0            mgcv_1.8-19
## [13] colorspace_1.3-2          sp_1.2-5
## [15] prettyunits_1.0.2         mnormt_1.5-5
## [17] phangorn_2.2.0            curl_2.8.1
## [19] compiler_3.4.1            animation_2.5
## [21] expm_0.999-2              xml2_1.1.1
## [23] scales_0.5.0              mvtnorm_1.0-6
## [25] quadprog_1.5-5            stringr_1.2.0
## [27] digest_0.6.12             pkgconfig_2.0.1
## [29] htmltools_0.3.6           plotrix_3.6-6
## [31] highr_0.6                 rlang_0.1.2
## [33] shiny_1.0.5               bindr_0.1
## [35] combinat_0.0-8            jsonlite_1.5
## [37] gtools_3.5.0              spdep_0.6-15
## [39] magrittr_1.5              Matrix_1.2-11
## [41] Rcpp_0.12.12              munsell_0.4.3
## [43] yaml_2.1.14               scatterplot3d_0.3-40
## [45] stringi_1.1.5             clusterGeneration_1.3.4
## [47] MASS_7.3-47               plyr_1.8.4
## [49] grid_3.4.1                gdata_2.18.0
## [51] adegenet_2.0.1            deldir_0.1-14
## [53] rncl_0.8.2                splines_3.4.1
## [55] igraph_1.1.2              uuid_0.1-2
## [57] taxize_0.8.9              boot_1.3-20
## [59] seqinr_3.4-5              reshape2_1.4.2
## [61] codetools_0.2-15          fastmatch_1.1-0
## [63] LearnBayes_2.15           crul_0.3.8
## [65] XML_3.98-1.9              GPArotation_2014.11-1
## [67] glue_1.1.1                evaluate_0.10.1
## [69] RcppArmadillo_0.7.960.1.2 RNeXML_2.0.7
## [71] msm_1.6.4                 data.table_1.10.4
## [73] httpuv_1.3.5              gtable_0.2.0
## [75] purrr_0.2.3               tidyr_0.7.1
## [77] reshape_0.8.7             assertthat_0.2.0
## [79] mime_0.5                  phylobase_0.8.4
## [81] xtable_1.8-2              coda_0.19-1
## [83] survival_2.41-3           tibble_1.3.4
## [85] bindrcpp_0.2              cluster_2.0.6
```

9 References