Tutorial: Phylogenetic Diversity and Endemism based on a phylogeny and Species Distribution Models

Timings

Section times as follows, based on using coarse environmental data (10 arc minutes):

1. Package install (ca. 3 min)
2. Download and prepare environmental data (ca. 2 min)
3. Process species occurrence data (ca. 5 min)
4. Build SDMs (ca. 5 min)
5. Calculate Phylogenetic Diversity and Endemism (from the SDMs and a provided phylogeny)

Other info

- If you are on a PC, directory paths need to be separated by back-slashes instead of forward-slashes
- Try not to resize windows/display panels while R is busy calculating/plotting - this tends to cause problems

1. Initial setup and package install

The following block of code will install all required R packages for this tutorial, you can copy and paste this in your R Studio and run it. All of this code works with the latest R version (4.1.0), and possibly older ones (e.g. 3.5.0), but I cannot guarantee it will work, so best to have R v.4.1.0.

```
# install all required R packages
install.packages(c(   'devtools', 'sp', 'rgbif', 'readr', 'ape', 'phylobase',
'foreach', 'doParallel', 'ggplot2', 'sp', 'stringr', 'gdistance') )
# This one installs from a github repository
devtools::install_github('jjvanderwal/SDMTools')
```

2. Script to prepare the environmental data

```
library(raster)
library (rgdal)
```

```
base.dir <- '/Users/cb76kecu/Dropbox (iDiv)/Micro-Macro_course/2021/Tutorial/
Part_2_PD_PE/'
setwd <- base.dir
```

Now we will create some directories where things will be saved

```
dir.create(paste0(base.dir))
dir.create(paste0(base.dir,'env_data/'))
dir.create(paste0(base.dir,'env_data/'))
dir.create(paste0(base.dir,'env_data/present_climate'))
dir.create(paste0(base.dir,'env_data/future_climate_2070_RCP_8.5'))
dir.create(paste0(base.dir,'sdm_R/'))
dir.create(paste0(base.dir,'sdm_R/ensembles'))
dir.create(paste0(base.dir,'sdm_R/ensembles/asc'))
dir.create(paste0(base.dir,'sdm_R/GBIF_data'))
```

download the data at 2.5 arc minute resolution globally

```
climate <- raster::getData('worldclim', var='bio', res=2.5)
```

Now we will clip the data to a specific extent (xmin,xmax,ymin,ymax)

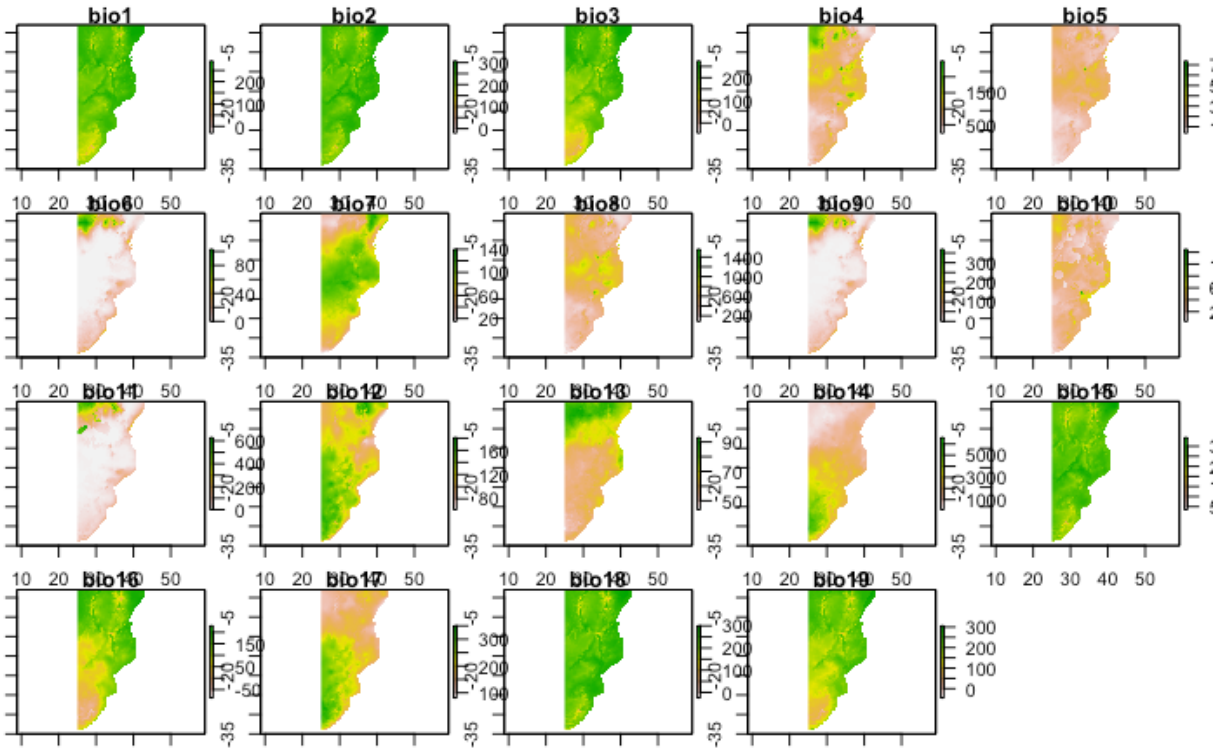
```
template <- extent(25, 43, -35, 2)
climate.crop <- crop(climate, template, snap="out")
```

And save each of the bioclim variables as .asc grid files

```
for(i in 1:nlayers(climate.crop)){
  writeRaster(climate.crop[[i]], paste0(base.dir, "env_data/present_climate/b
ioclim_", i), "ascii", overwrite = T)
  cat('Writing bioclim',i,
      '... \n')
}
```

Now load them from file and plot them, just to check they look ok

```
env_data <- paste0(base.dir,'env_data/present_climate')
lst <- list.files(path=env_data,pattern='asc$',full.names = T)
climate <- stack(lst)

par(mar=c(1,1,1,1))
par(mfrow=c(4,5))
for(i in 1:nlayers(climate)){
  plot(climate[[i]], main= paste0("bio",i))
}
```



And delete temporary files

```
unlink('../../wc2-5/', recursive=TRUE)
```

3. Process species occurrence data

```
library(raster)
library(rgbif)
library(readr)
library(dismo)
library(rgbif)
library(dplyr)

base.dir <- '/Users/cb76kecu/Dropbox (iDiv)/Micro-Macro_course/2021/Tutorial/Part_2_PD_PE/'
```

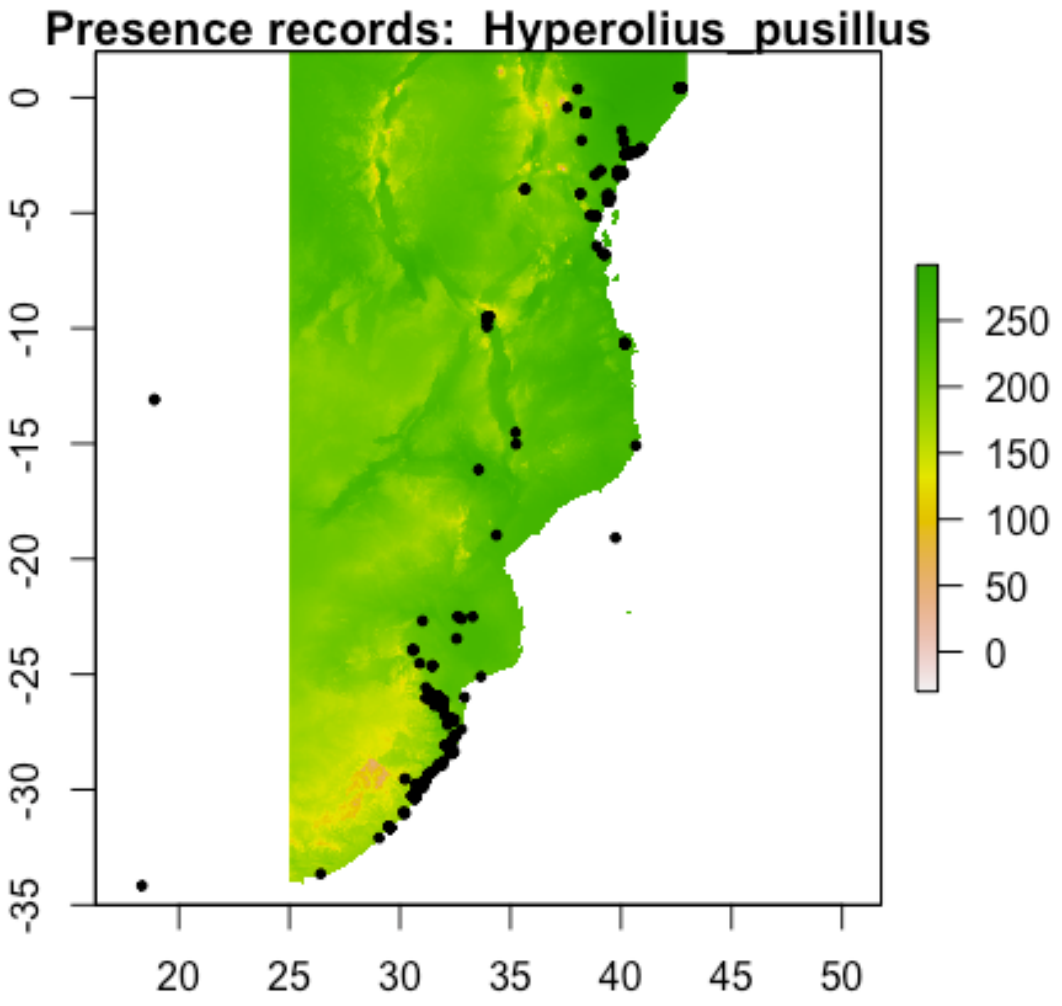
Read in species csv files with presence data. You don't need to do this here, but just so you know - it's possible to provide rgbif a long list of species and save the data for each species, see here: https://data-blog.gbif.org/post/downloading-long-species-lists-on-gbif/

```
setwd(paste0(base.dir,'/sdm_R/'))
input.dir = "./GBIF_data/per_species/"
setwd(input.dir)
input_files <-gsub("\\.csv$","", list.files(pattern="\\.csv$"))
for(i in input_files){
  filepath <- file.path("./",paste(i,".csv",sep=""))
  assign(i, read.csv(filepath, sep = ","))
```

```
  cat("\n Reading",i,"csv file... done!")
}
```

plot samples on a map

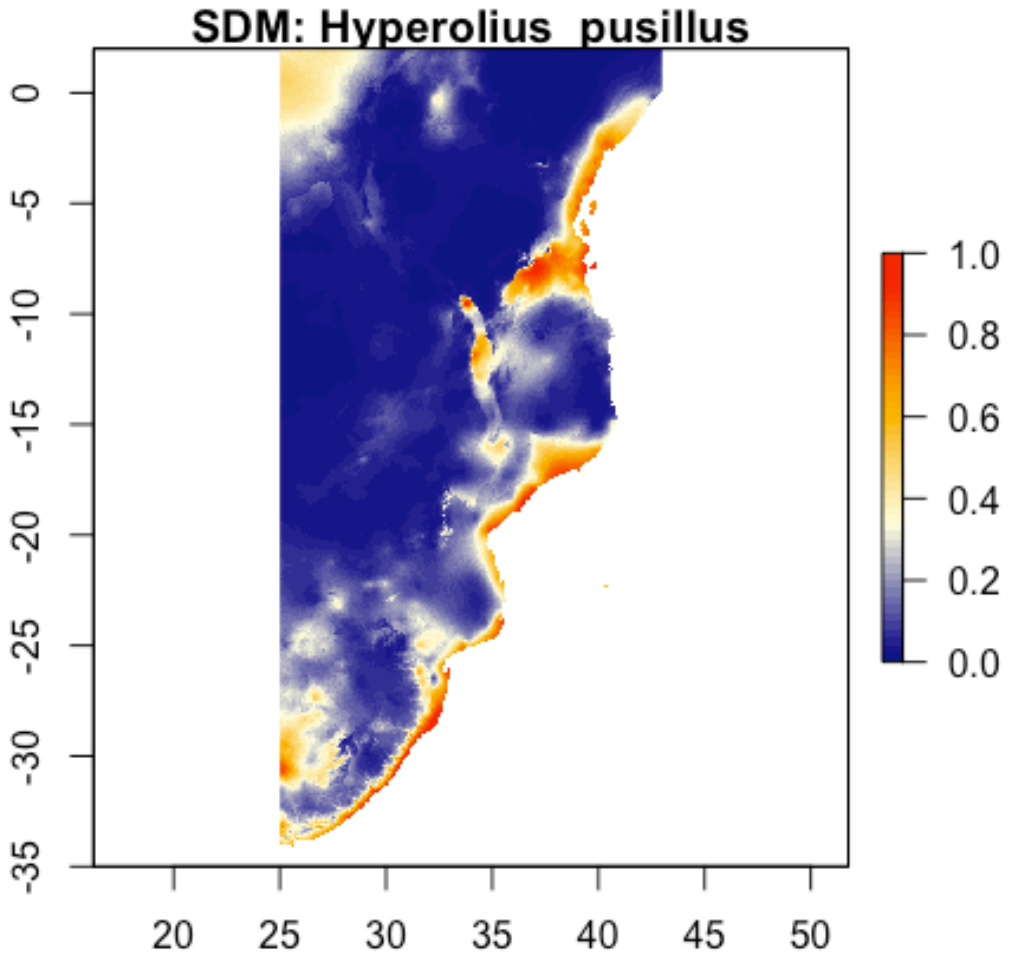
```
mar <- c(1,1,1,1)
for(i in input_files){
  filepath <- file.path("./",paste(i,".csv",sep=""))
  assign(i, read.csv(filepath, sep = ","))
  sp <- read.csv(filepath)
  sp <- sp[,c('decimalLongitude','decimalLatitude')]
  sp$species <- 1
  coordinates(sp) <- ~ decimalLongitude + decimalLatitude
  par(mfrow=c(1,1))
  plot(climate[[1]], main = paste("Presence records: ", i))
  plot(sp, pch=21, cex=0.5, add=T)
}
```



4. Build SDMs

```
library(sdm)
library(raster)
library(rgdal)
library(sp)
library(usdm)

# remove correlated predictor variables
# read in all the data points
sp <- read.csv('../../GBIF_data/GBIF_data.csv')
# tell package which colums represent long and lat
sp <- sp[,c('decimalLongitude','decimalLatitude')]
# extract the predictor variable values from these points and put in a data f
rame
spx <- extract(climate, sp)
spx <- data.frame(spx)
# measure variable inflation (similar to Pearson's correlation in sdm R packa
ge, and remove highly correlated variables)
v <- vifstep(spx)
v
# exclude these highly correlated variables and overwrite the bio data (repla
cing the 19 bioclim vars with a subset of uncorrelated predictors)
climate <- exclude(climate, v)
climate

setwd(paste0(base.dir,'sdm_R/ensembles/asc/'))

library(sdm)

wd <- getwd()
par(mfrow=c(1,1))
mar <- c(1,1,1,1)
for(i in input_files){
  filepath <- file.path("../../GBIF_data/per_species/",paste(i,".csv",sep="")
)
  assign(i, read.csv(filepath, sep = ","))
  sp <- read.csv(filepath)
  sp <- sp[,c('decimalLongitude','decimalLatitude')]
  sp$species <- 1
  head(sp)
  coordinates(sp) <- ~ decimalLongitude + decimalLatitude
  d <- sdmData(species~., train=sp, predictors= climate, bg=list(n=1000))
  methods <- c('maxlike','glm') # which methods required? - e.g. 'bioclim','b
ioclim.dismo','gam','glm','maxlike','rpart'
  m <- sdm(species ~ . , d, methods=methods, replication='sub', test.p=30, n=
1) # data partitioning/ test percentage?
  ensemble_model  <- ensemble(m,climate, setting=list(id=1:2, method='weighte
d', stat='AUC', opt=2)) # how to combine models?
  writeRaster(ensemble_model, filename=paste0(wd,'/SDM_',i,'.asc'), format="a
scii", overwrite=TRUE)
```

```
  plot(ensemble_model, zlim=c(0,1),main = paste0("SDM: ", i), col=colorRampPa
lette(c('navy','lightyellow','orange','red'))(50))
  lst1 <- list.files(path=getwd(),pattern='grd$',full.names = T)
  lst2 <- list.files(path=getwd(),pattern='gri$',full.names = T)
  file.remove(lst1)
  file.remove(lst2)
}
```



5. Calculate SR/WE/PD/PE from these SDMs using a phylogeny

This code calculates richness and endemism from modelled suitability surfaces, it requires all the model grids to have the same extent

```
rm(list=ls())

library(SDMTools)
library(raster)
```

```
library(ape)
library(phylobase)
library(foreach)
library(doParallel)
library(ggplot2)

base.dir <- '/Users/cb76kecu/Dropbox (iDiv)/Micro-Macro_course/2021/Tutorial/Part_2_PD_PE/'
phylo.dir <- (paste0(base.dir,'phylo/'))   # modify to the base directory
source(paste0(phylo.dir,'phylogenetic endemism.r'))
```

First define some functions

```
map_raster = function(raster, output_file, title) {
  p         <- rasterToPoints(raster)
  p         <- data.frame(p)
  names(p) <- c("x", "y", "Model")
  colour_gradient <- scale_fill_gradientn(colours = rainbow(15), values=p$model)
  colour_gradient <- scale_fill_gradient2(low="white", mid="yellow", high="red",
                                          limits=c(min(p$Model),max(p$Model)), midpoint=quantile(p$Model, 0.75), space='Lab')
  m <- ggplot(data=p) + geom_tile(aes(x, y, fill=Model)) + coord_equal() + labs(x=NULL, y=NULL) + colour_gradient
  # delete a previous file if needed
  if (file.exists(output_file)) {
    file.remove(output_file)
    cat("Previous", output_file, "removed\n")
  }
  m <- m + ggtitle(title)
  m <- m + theme(axis.title=element_text(face="bold", size="18"))
  m <- m + theme(axis.text=element_text(face="bold", size="14"))
  m <- m + theme(plot.title=element_text(face="bold", size="24"))
  m <- m + xlab("longitude") + ylab("latitude")
  png(output_file, width=image.width, height=image.height)
  print(m)
  dev.off()
  m <- NULL
}
```

Now some parameters

```
max.rows        <- 10000000
core_count      <- 4 # number of cores to use for parallel steps
# size in pixels for maps
image.width=1400
image.height=1400
```

Define directories

```
phylo.dir        <- (paste0(base.dir,'phylo/'))      # modify to the base dir
ectory for your lineage modelling
input.dir       <- paste(base.dir, 'sdm_R/ensembles/asc/', sep ='') # input l
ineage models
output.dir      <- paste(phylo.dir, 'output/', sep ='') # output location whe
re diversity results and maps will be saved
file_pattern    <- 'SDM_'                     # modify this to a match the st
art of the name of all lineage model asc files

group_lin_file  <- paste(phylo.dir, 'All_species_list.csv', sep ='')
```

Tree details - this works for one genus at a time

```
tree.file       <- paste(phylo.dir, '/species_tree.tre', sep ='')
outgroup        <- 'SDM_scolecomorphus_vittatus'
preface         <- ""
genus           <- ''
output_prefix   <- 'SDMs_'
threshold       <- 0.0000000000001  # this is not a species level threshold,
but one used for each lineage model
```

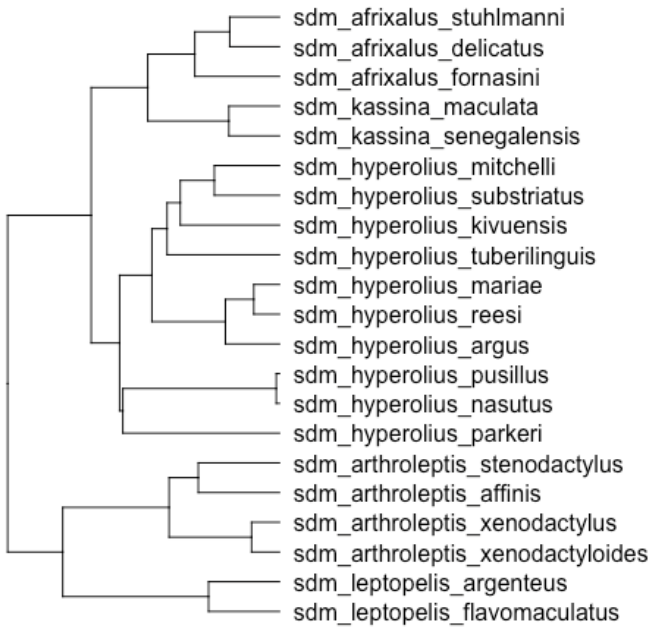
Start Calculations

```
setwd(base.dir)
files <- list.files(path = input.dir, pattern = file_pattern, recursive = FAL
SE,ignore.case = TRUE, include.dirs = FALSE)
setwd(input.dir)
template.asc = read.asc(paste0(input.dir,'SDM_Afrixalus_delicatus.asc'))
model_rows=nrow(template.asc)
model_cols=ncol(template.asc)
pos <- as.data.frame(which(is.finite(template.asc),arr.ind=TRUE)) #get all po
ints that have data
cat("\nLoading model rasters in parallel\n")
cl <- makeCluster(core_count)
registerDoParallel(cl)
pos_par <- foreach (j=1:length(files), .combine=cbind, .packages='SDMTools')
%dopar% {
  tfile <- files[j]
  pos_temp <- pos
  checkname = unlist(strsplit(tfile,".",fixed=T))
  if (checkname[length(checkname)]=="asc") {   # only accept filenames ending
in .asc
    cat(j)
    tasc = read.asc(tfile)                                #read in the data
    dataname=gsub(".asc",'',tfile)
    newname <- tolower(gsub(preface, "", dataname))
    cat("About to do pos\n")
    pos_temp[newname] <- tasc[cbind(pos_temp$row, pos_temp$col)]
    pos_temp[(which(pos_temp[newname]< threshold)), newname] <- 0    # set va
lues below the threshold to 0
    pos_temp[(which(is.na(pos_temp[newname]))), newname]     <- 0    # set th
```

```
e nulls to 0
    cat("\n", j, newname, "loaded")
    newcol <- data.frame(pos_temp[, newname])
    newcol <- round(newcol,3)
    names(newcol) <- newname
    newcol
  }
}

pos <- cbind(pos,pos_par)
rm(pos_par)
setwd(base.dir)
group_lin_list <-read.csv(group_lin_file, stringsAsFactors=F)
```

Read in the tree

```
mar=c(1,1,1,1)
tree <- read.nexus(tree.file)
plot(tree)
tree <- keep.tip(tree, c('SDM_afrixalus_delicatus','SDM_afrixalus_fornasini',
'SDM_afrixalus_stuhlmanni','SDM_arthroleptis_affinis','SDM_arthroleptis_steno
dactylus','SDM_arthroleptis_xenodactyloides','SDM_arthroleptis_xenodactylus',
'SDM_hyperolius_argus','SDM_hyperolius_kivuensis','SDM_hyperolius_mariae','SD
M_hyperolius_mitchelli','SDM_hyperolius_nasutus','SDM_hyperolius_parkeri','SD
M_hyperolius_pusillus','SDM_hyperolius_reesi','SDM_hyperolius_substriatus','S
DM_hyperolius_tuberilinguis','SDM_kassina_maculata','SDM_kassina_senegalensis
','SDM_leptopelis_argenteus','SDM_leptopelis_flavomaculatus'))
tree <- phylo4(tree)
labels(tree) <- tolower(labels(tree))
plot(tree)
```



Ensure that the tree tips match the model names

```
model.names <- names(pos)
model.names <- model.names[-(1:2)] # names of all columns except the 1st two
which are row, col
model.names <- tolower(gsub(preface,"",model.names))
model.groups <- data.frame(model.groups=vector("character",nTips(tree)),strin
gsAsFactors=F)

for (i in 1:nTips(tree)) {
  cat(labels(tree)[i],"\n")
  row <- group_lin_list[group_lin_list$lineage==labels(tree)[i],]
  if (nrow(row) > 0) {
    new_tip_name <- tolower(paste(row$genus, row$model_group, row$lineage, se
p="_"))
    labels(tree)[i] <- new_tip_name
    model.groups[i,1] <- as.character(row$model_group)
  }
}

plot(tree)
tree <- phylo4d(tree,tip.data=model.groups)
tree
tree.names  <- as.character(labels(tree)[1:nTips(tree)]) ## original line
matched.names <- intersect(model.names,tree.names)
matched.tips  <- which(labels(tree,"tip") %in% matched.names)
cat("\nNot in tree names:",setdiff(model.names,tree.names),"\n")
cat("\nNot in model names:",setdiff(tree.names,model.names),"\n")

# a subtree containing only the tips for which there is a corresponding model
subtree <- subset(tree,tips.include=matched.tips)
subtree <- tree
# limit the occurrence table to lineages which match the tree
matching_pos_columns <- which(names(pos) %in% matched.names)
matching_pos_columns <- unique(c(1, 2, matching_pos_columns))  # ensure that
row and column are included
pos <- pos[, matching_pos_columns]
cat("\n\nRemoving unoccupied cells\n")
cat("Before:",nrow(pos),"\n")
rowsums <- apply(pos[,3:ncol(pos)],1,sum,na.rm=T)
pos <- pos[which(rowsums>0),]
rm(rowsums)
cat("After:",nrow(pos),"\n")
max.rows <- min(max.rows,nrow(pos))
```

Perform the calculations

```
result <- calc_PE_from_models(subtree,pos[1:max.rows,which(names(pos) %in% ma
tched.names)], core_count = core_count)
cat("\nDiversity calculations completed. Now writing outputs to file.")
```

```
pos_output <- cbind(pos[1:max.rows,1:2],result)
pos_output <- pos_output[,-3] # omit the site column
```

```
Calculating species measures
SR done
WE done

22      b22     21
23      b23     6
24      b24     2
25      b25     4
```

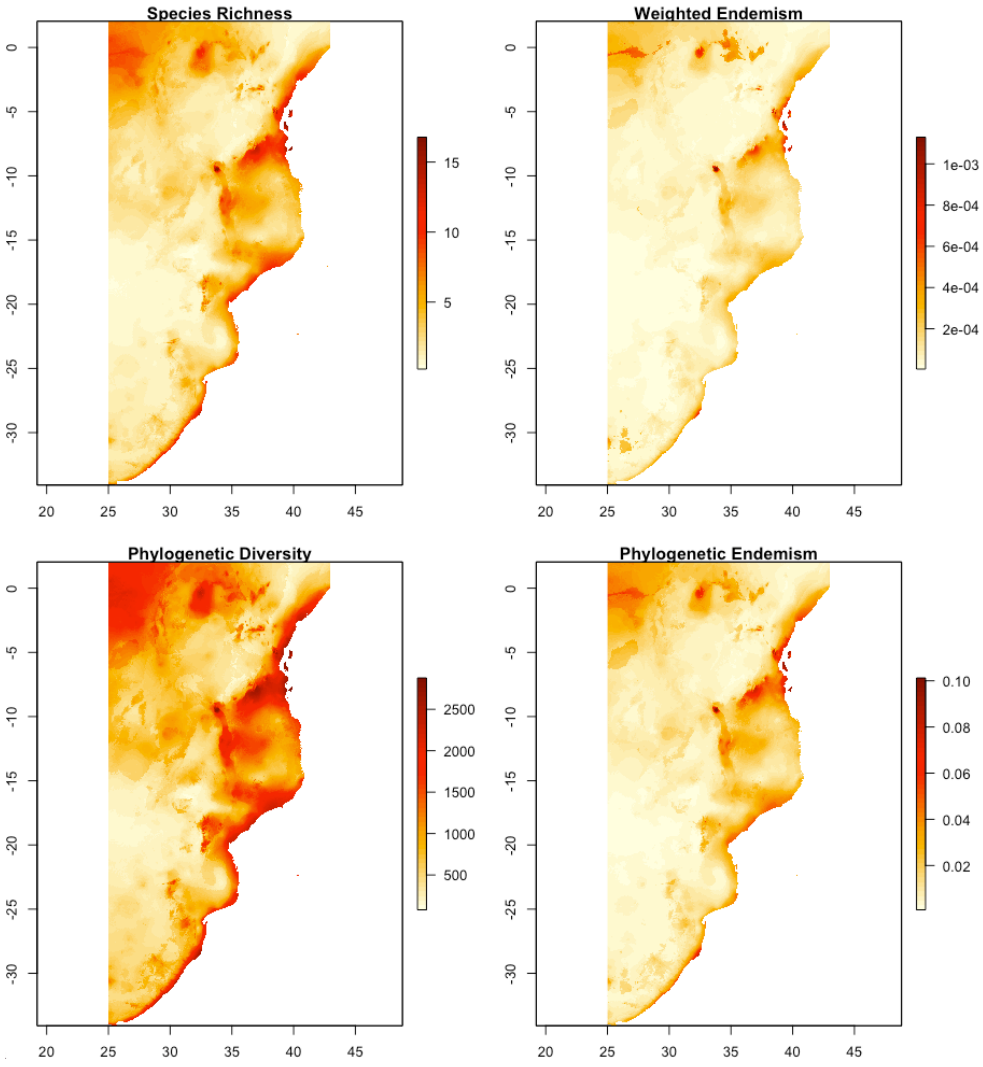
Plot and make output files

```
cellsize <- attr(template.asc,"cellsize")
ymin <- attr(template.asc,"yll")
xmin <- attr(template.asc,"xll")

x <- ((pos_output$row - 1) * cellsize) + xmin
y <- ((pos_output$col - 1) * cellsize) + ymin
pos_output <- cbind(pos_output[,1:2],x,y,pos_output[,-(1:2)])

filenames <- c(paste(output_prefix,"PE.asc",sep=""),paste(output_prefix,"PD.a
sc",sep=""),paste(output_prefix,"WE.asc",sep=""),paste(output_prefix,"SR.asc"
,sep=""))
dataframe2asc(pos_output[,c(4,3,5:8)],filenames,output.dir)

stopCluster(cl)
setwd(output.dir)
```

Save output images

```
PE.ras <- raster(filenames[1])
map_filename <- paste(output_prefix, "PE.png", sep="")
map_raster(PE.ras, map_filename, paste(output_prefix, "PE"))

PD.ras <- raster(filenames[2])
map_filename <- paste(output_prefix, "PD.png", sep="")
map_raster(PD.ras, map_filename, paste(output_prefix, "PD"))

WE.ras <- raster(filenames[3])
map_filename <- paste(output_prefix, "WE.png", sep="")
map_raster(WE.ras, map_filename, paste(output_prefix, "WE"))

SR.ras <- raster(filenames[4])
map_filename <- paste(output_prefix, "SR.png", sep="")
map_raster(SR.ras, map_filename, paste(output_prefix, "SR"))
dev.off()

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

```
mar=c(1,1,1,1)
colour_scale <- colorRampPalette(c('lightyellow','orange','red','darkred'))(5
0)
plot(SR.ras, main="Species Richness", col=colour_scale)
plot(WE.ras, main="Weighted Endemism", col=colour_scale)
plot(PD.ras, main="Phylogenetic Diversity", col=colour_scale)
plot(PE.ras, main="Phylogenetic Endemism", col=colour_scale)
```



Bonus tutorial assignment

You can use the code above as a basis to do the following things: 1. Rerun the SDMs but do it properly - we did a quick and dirty analysis here but I'd like you to try and make some better ensembles based on what you learned in Tutorial 1 2. Redo the code above but project your models into the future to see how much your Phylogenetic Indices (Phylogenetic Diversity and Endemism) change