R Exercise

RS-eco

09.04.2019

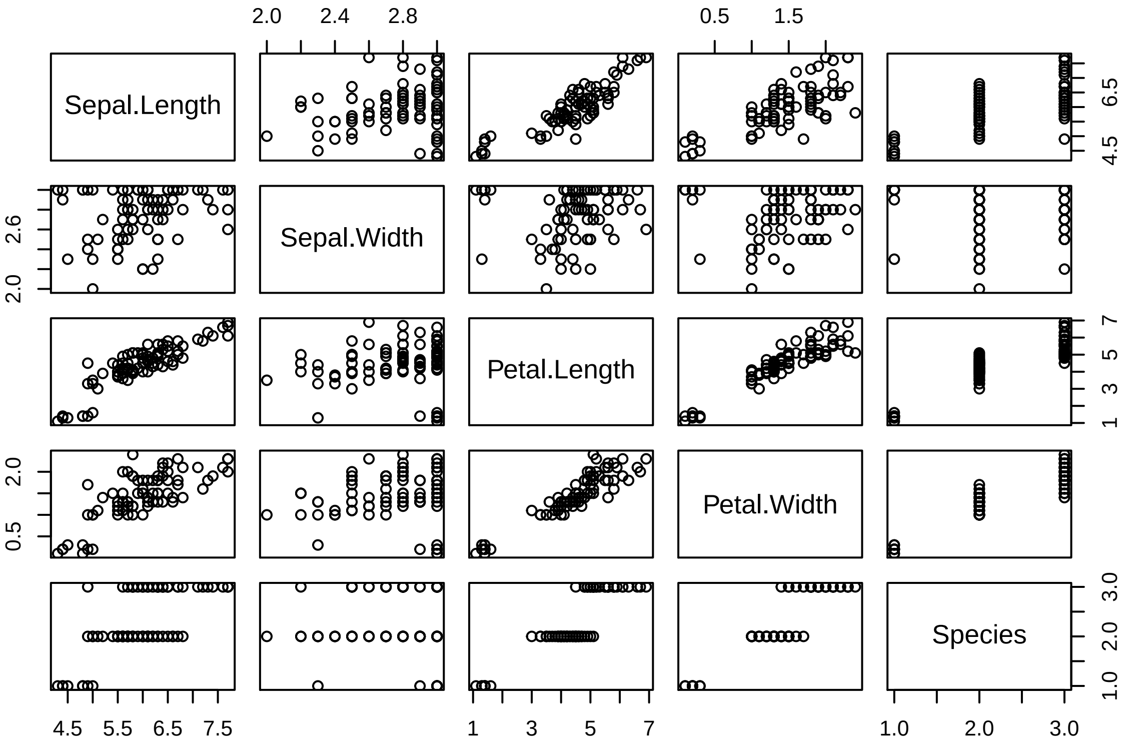
1. Piping

Install and load the magrittr package

```
#install.packages("magrittr", dep=T)
library(magrittr)
```

Turn the following code into a streamlined version using piping:

```
# Without piping
mean_width <- mean(iris$Sepal.Width)
iris_sub <- subset(iris, Sepal.Width < mean_width)
plot(iris_sub)
cor_iris <- cor(iris_sub$Sepal.Length,
    iris_sub$Sepal.Width)

# With piping
mean_width <- iris %$% mean(Sepal.Width)
iris %>%
  subset(Sepal.Width < mean_width) %>%
  plot()
```



```
cor_iris <- iris %>%
  subset(Sepal.Width < mean(Sepal.Width)) %T>%
  plot %$% cor(Sepal.Length, Sepal.Width)
```

```
######################################################################
```

2. Read data

Install and load the readr package

```
#install.packages(readr, dep=T)
library(readr)
```

Set working directory

```
#setwd("")
```

Read species_info.csv.xz into a data.frame called species_info Note: species_info contains information on more than 40000 vertebrate species

```
species_info <- read.csv("data/species_info.csv.xz")
library(readr)
species_info <- read_csv("data/species_info.csv.xz")
```

```
## Parsed with column specification:
## cols(
##   binomial = col_character(),
##   presence = col_double(),
##   origin = col_double(),
##   seasonal = col_double(),
##   kingdom_na = col_character(),
##   phylum_nam = col_character(),
##   class_name = col_character(),
##   order_name = col_character(),
##   family_nam = col_character(),
##   shape_Area = col_double()
## )
```

Check class and structure of species_info

```
class(species_info)
```

```
## [1] "spec_tbl_df" "tbl_df"      "tbl"         "data.frame"
```

```
str(species_info)
```

```
## Classes 'spec_tbl_df', 'tbl_df', 'tbl' and 'data.frame': 40584 obs. of  10 variables:
##  $ binomial  : chr  "Abavorana luctuosa" "Acanthixalus sonjae" "Acanthixalus spinosus" "Acris crepita
##  $ presence  : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ origin    : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ seasonal  : num  1 1 1 1 1 1 1 1 1 1 ...
##  $ kingdom_na: chr  "ANIMALIA" "ANIMALIA" "ANIMALIA" "ANIMALIA" ...
##  $ phylum_nam: chr  "CHORDATA" "CHORDATA" "CHORDATA" "CHORDATA" ...
##  $ class_name: chr  "AMPHIBIA" "AMPHIBIA" "AMPHIBIA" "AMPHIBIA" ...
##  $ order_name: chr  "ANURA" "ANURA" "ANURA" "ANURA" ...
##  $ family_nam: chr  "RANIDAE" "HYPEROLIIDAE" "HYPEROLIIDAE" "HYLIDAE" ...
##  $ shape_Area: num  9.399 0.465 98.687 322.578 63.883 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   binomial = col_character(),
##   ..   presence = col_double(),
##   ..   origin = col_double(),
##   ..   seasonal = col_double(),
##   ..   kingdom_na = col_character(),
##   ..   phylum_nam = col_character(),
##   ..   class_name = col_character(),
##   ..   order_name = col_character(),
##   ..   family_nam = col_character(),
##   ..   shape_Area = col_double()
##   .. )
```

Read first 10 entries of species_info.csv.xz

```
species_info_first10 <- read_csv("data/species_info.csv.xz", n_max=10)
```

```
## Parsed with column specification:
## cols(
##   binomial = col_character(),
##   presence = col_double(),
##   origin = col_double(),
##   seasonal = col_double(),
##   kingdom_na = col_character(),
##   phylum_nam = col_character(),
##   class_name = col_character(),
##   order_name = col_character(),
##   family_nam = col_character(),
##   shape_Area = col_double()
## )
```

Read last 10 lines of species_info.csv.xz

```
col_sp_info <- colnames(read_csv("data/species_info.csv.xz",
                                 n_max=1))
```

```
## Parsed with column specification:
## cols(
##   binomial = col_character(),
##   presence = col_double(),
##   origin = col_double(),
##   seasonal = col_double(),
##   kingdom_na = col_character(),
##   phylum_nam = col_character(),
##   class_name = col_character(),
##   order_name = col_character(),
##   family_nam = col_character(),
##   shape_Area = col_double()
## )
```

```
species_info_last10 <- read_csv("data/species_info.csv.xz",
                         col_names=col_sp_info,
                         skip=40574)
```

```
## Parsed with column specification:
## cols(
##   binomial = col_character(),
##   presence = col_logical(),
##   origin = col_logical(),
##   seasonal = col_logical(),
##   kingdom_na = col_character(),
##   phylum_nam = col_character(),
##   class_name = col_character(),
##   order_name = col_logical(),
##   family_nam = col_logical(),
##   shape_Area = col_double()
## )
```

```
# Skip the first 10 and skip the last 10 lines
species_info_other <- read_csv("data/species_info.csv.xz",
                         col_names=col_sp_info,
                         skip=10, n_max=40564)
```

```
## Parsed with column specification:
## cols(
##   binomial = col_character(),
##   presence = col_double(),
##   origin = col_double(),
##   seasonal = col_double(),
##   kingdom_na = col_character(),
##   phylum_nam = col_character(),
##   class_name = col_character(),
##   order_name = col_character(),
##   family_nam = col_character(),
##   shape_Area = col_double()
## )
```

```
######################################################################
```

3. tidyr

Install and load the tidyr package

```
#install.packages("tidyr", dep=T)
library(tidyr)
```

Using species_info, replace NAs of presence, origin and season column with 0s. Complete binomial, presence, origin and season columns of species_info Drop missing values from species_info Separate binomial into genus and species, but keep binomial

```
# Are there any NAs?
species_info %>% anyNA()
```

```
## [1] TRUE
```

```
# How many rows with NAs in origin column?
species_info %>%
  subset(is.na(origin)==TRUE) %>%
  nrow
```

```
## [1] 10064
```

```
species_new <- species_info %>%
  replace_na(list(presence=0, origin=0, seasonal=0)) %>%
  complete(binomial, kingdom_na, presence, origin, seasonal) %>%
  drop_na() %>%
  separate(binomial,
           into=c("genus", "species"),
           remove=F)
```

```
## Warning: Expected 2 pieces. Additional pieces discarded in 7 rows [19706, 19981,
## 23353, 23746, 24166, 25971, 25973].
```

```
# colnames(species_new)
```

Why do you get an error message and how could we avoid it? Because some species also include information about the subspecies

```
species_new <- species_info %>%
  replace_na(list(presence=0, origin=0, seasonal=0)) %>%
  complete(binomial, kingdom_na, presence, origin, seasonal) %>%
  drop_na() %>%
  separate(binomial,
           into=c("genus", "species", "subspecies"),
           remove=F)
```

```
## Warning: Expected 3 pieces. Additional pieces discarded in 4 rows [19706, 19981,
## 23746, 24166].
```

```
## Warning: Expected 3 pieces. Missing pieces filled with `NA` in 30509 rows [1, 2,
## 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, ...].
```

Read species_data.csv.xz into a data.frame called species_data Note: species_data contains gridded species occurrence data for Bavaria at a spatial resolution of 0.5 degree

```
species_data <- read_csv("data/species_data.csv.xz")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double()
## )
```

```
## See spec(...) for full column specifications.
```

2. Turn species_data from wide into long format, drop missing values and save to a data.frame called species_long

```
species_long <- species_data %>%
  gather(binomial, occurrence, -c(x,y)) %>% drop_na()

# Alternative
species_long <- gather(species_data,
                       colnames(species_data)[-c(1:2)],
       key="binomial", value="occurrence") %>% drop_na()
```

Why do we have to drop NAs after we turn data into a long format? Because, we get lots of unmeaningful data entries when turning the data into a long format #################### Turn species_long back into wide format and write to a data.frame called species_wide

```
species_wide <- species_long %>%
  spread(binomial, occurrence)

######################################################################
```

4. dplyr

```
# Install and load dplyr package
```

```
#install.packages("dplyr", dep=T)
library(dplyr)
```

Using species_long calculate the number of occurrences per species using group_by() and summarise

```
species_long %>% group_by(binomial) %>%
  summarise(sum=sum(occurrence))
```

```
## # A tibble: 322 x 2
##    binomial                       sum
##    <chr>                        <dbl>
##  1 Accipiter gentilis             189
##  2 Accipiter nisus                189
##  3 Acrocephalus arundinaceus      163
##  4 Acrocephalus dumetorum         121
##  5 Acrocephalus paludicola          6
##  6 Acrocephalus palustris         189
##  7 Acrocephalus schoenobaenus     156
##  8 Acrocephalus scirpaceus        189
##  9 Actitis hypoleucos             134
## 10 Aegithalos caudatus            189
## # ... with 312 more rows
```

```
species_long %>% group_by(binomial) %>%
  summarise(sum=n())
```

```
## # A tibble: 322 x 2
##    binomial                       sum
##    <chr>                        <int>
##  1 Accipiter gentilis             189
##  2 Accipiter nisus                189
##  3 Acrocephalus arundinaceus      163
##  4 Acrocephalus dumetorum         121
##  5 Acrocephalus paludicola          6
##  6 Acrocephalus palustris         189
##  7 Acrocephalus schoenobaenus     156
##  8 Acrocephalus scirpaceus        189
##  9 Actitis hypoleucos             134
## 10 Aegithalos caudatus            189
## # ... with 312 more rows
```

Of course, this could also be done with colSums() or count() #################### Using species_long, identify how many species have less than 10 occurrences?

```
species_long %>% group_by(binomial) %>%
  summarise(sum=sum(occurrence)) %>%
  arrange(desc(sum)) %>%
  filter(sum < 10)
```

```
## # A tibble: 34 x 2
##    binomial                 sum
##    <chr>                  <dbl>
##  1 Calidris alpina            9
##   2 Columba livia               9
##   3 Dryomys nitedula            9
##   4 Natrix tessellata           8
##   5 Pipistrellus kuhlii         8
##   6 Tichodroma muraria          8
##   7 Apodemus alpicola           7
##   8 Carduelis citrinella        7
##   9 Strix uralensis             7
##  10 Acrocephalus paludicola     6
## # ... with 24 more rows
```

Using species_long calculate species richness per grid cell

```
sr_xy <- species_long %>% group_by(x,y) %>%
  summarise(sr=sum(occurrence))
```

Extract data of the following 5 species from species_long:

```
species <- c("Anas crecca", "Lacerta agilis", "Mustela nivalis",
             "Pelophylax lessonae", "Vulpes vulpes")
# %in%

species_long %>% filter(binomial %in% species)
```

```
## # A tibble: 920 x 4
##          x     y binomial    occurrence
##      <dbl> <dbl> <chr>            <dbl>
##   1  6.25   50.2 Anas crecca          1
##   2  6.25   50.8 Anas crecca          1
##   3  6.25   51.2 Anas crecca          1
##   4  6.25   51.8 Anas crecca          1
##   5  6.75   49.2 Anas crecca          1
##   6  6.75   49.8 Anas crecca          1
##   7  6.75   50.2 Anas crecca          1
##   8  6.75   50.8 Anas crecca          1
##   9  6.75   51.2 Anas crecca          1
##  10  6.75   51.8 Anas crecca          1
## # ... with 910 more rows
```

```
species_long %>%
  filter(binomial %in% c("Anas crecca", "Lacerta agilis"))
```

```
## # A tibble: 365 x 4
##          x     y binomial    occurrence
##      <dbl> <dbl> <chr>            <dbl>
##   1  6.25   50.2 Anas crecca          1
##   2  6.25   50.8 Anas crecca          1
##   3  6.25   51.2 Anas crecca          1
##   4  6.25   51.8 Anas crecca          1
##   5  6.75   49.2 Anas crecca          1
##   6  6.75   49.8 Anas crecca          1
##   7  6.75   50.2 Anas crecca          1
##   8  6.75   50.8 Anas crecca          1
##   9  6.75   51.2 Anas crecca          1
##  10  6.75   51.8 Anas crecca          1
## # ... with 355 more rows
```

Using species_wide, extract data for all species starting with "Ac"

```
species_ac <- species_long %>% spread(binomial, occurrence) %>%
  select(starts_with("Ac"))
nrow(species_ac); ncol(species_ac)
```

```
## [1] 189
```

```
## [1] 9
```

Why do you have to use species_wide here? Because starts_with() only works with select() and not with filter() and with select you can only subset columns and not rows #################### Calculate the number of species per class_name in species_info and output as table using the kable function of the knitr package

```
library(knitr)
species_info %>% group_by(class_name) %>%
  distinct(binomial) %>% summarise(no_species=n()) %>%
  kable()
```

class_name	no_species
AMPHIBIA	6428
AVES	10066
MAMMALIA	5322
REPTILIA	10064

```
########################################################################
```

5. dplyr - Part II

Calculate the number of species per class in species_data Hint: To do this, you first need to join species_data with species_info

```
species_info %<>%
  distinct(binomial, class_name, order_name, family_nam)
#species_long %>%
#  left_join(species_info, by = "binomial") %>%
#  group_by(class_name) %>% distinct(binomial) %>%
#  summarise(no_species=n())
species_long %>%
  left_join(species_info, by = "binomial") %>%
  group_by(class_name) %>%
  summarise(no_species=length(unique(binomial)))
```

```
## # A tibble: 4 x 2
##   class_name no_species
##   <chr>           <int>
## 1 AMPHIBIA           21
## 2 AVES              213
## 3 MAMMALIA           77
## 4 REPTILIA           11
```

Calculate the number of species per grid cell for each class Hint: To do this, species_info must contain distinct species names (binomial)

```
species_long %>%
  left_join(species_info, by="binomial") %>%
  group_by(x,y,class_name) %>%
  summarise(sr=sum(occurrence))
```

```
## # A tibble: 756 x 4
## # Groups:   x, y [189]
##        x     y class_name    sr
##    <dbl> <dbl> <chr>      <dbl>
##  1  6.25  50.2 AMPHIBIA      16
##  2  6.25  50.2 AVES         126
##  3  6.25  50.2 MAMMALIA      54
##  4  6.25  50.2 REPTILIA       4
##  5  6.25  50.8 AMPHIBIA      17
##  6  6.25  50.8 AVES         132
##  7  6.25  50.8 MAMMALIA      54
##  8  6.25  50.8 REPTILIA       4
##  9  6.25  51.2 AMPHIBIA      17
## 10  6.25  51.2 AVES         133
## # ... with 746 more rows
```

Identify the most widely distributed species for each class

```
sp_wide <- species_long %>%
  left_join(species_info, by="binomial") %>%
  group_by(binomial, class_name, family_nam) %>%
  summarise(sum=sum(occurrence)) %>%
  group_by(class_name) %>% top_n(1,sum)
```

Note: This should not be higher than the maximum grid cells of Germany (189)

```
######################################################################
```

6. Making plots with ggplot2

Install and load ggplot2

```
#install.packages(ggplot2, dep=T)
library(ggplot2)
```

Plot barchart of the number of species per class

```
# Same as before
library(readr); library(dplyr);
library(tidyr); library(ggplot2)
species_long <- read_csv("data/species_data.csv.xz") %>%
  gather(binomial, occurrence, -c(x,y)) %>% drop_na()
```

```
## Parsed with column specification:
## cols(
##   .default = col_double()
## )
```

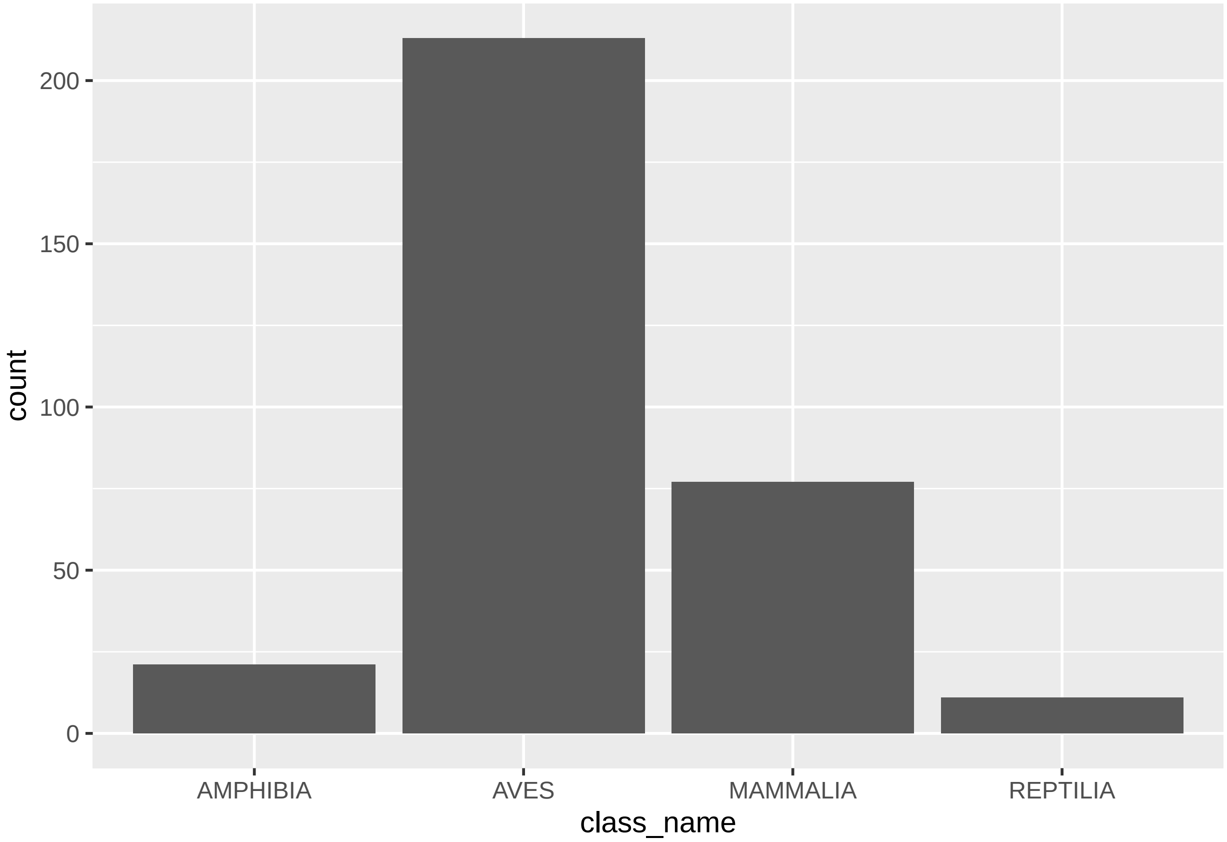
```
## See spec(...) for full column specifications.
```

```
species_info <- read_csv("data/species_info.csv.xz")
```

```
## Parsed with column specification:
```

```
## cols(
##   binomial = col_character(),
##   presence = col_double(),
##   origin = col_double(),
##   seasonal = col_double(),
##   kingdom_na = col_character(),
##   phylum_nam = col_character(),
##   class_name = col_character(),
##   order_name = col_character(),
##   family_nam = col_character(),
##   shape_Area = col_double()
## )
```

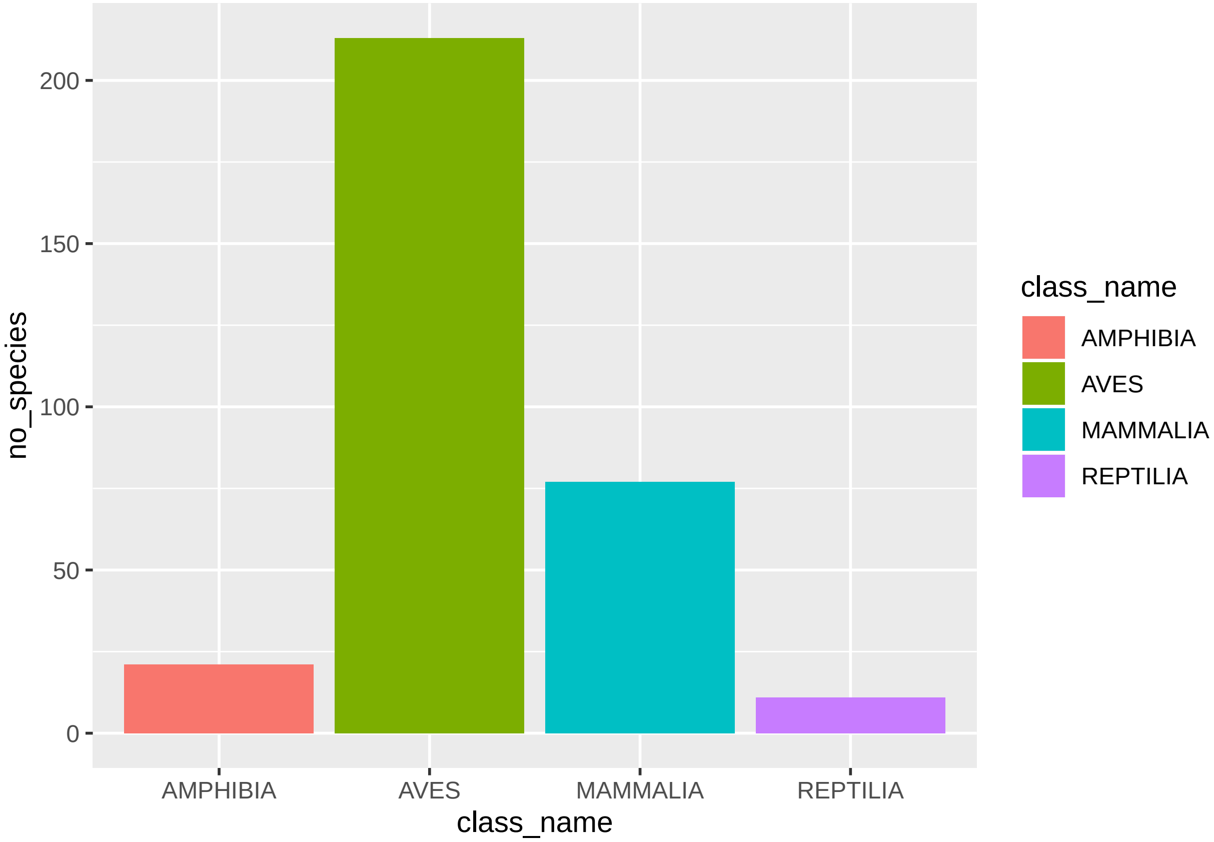
```
data_plot <- species_long %>%
  left_join(species_info, by=("binomial")) %>%
  group_by(class_name) %>% distinct(binomial) %>%
  summarise(no_species= n())

# Now, we add a plot
species_long %>%
  left_join(species_info, by=("binomial")) %>%
  group_by(class_name) %>% distinct(binomial) %>%
  ggplot(aes(class_name)) + geom_bar()
```



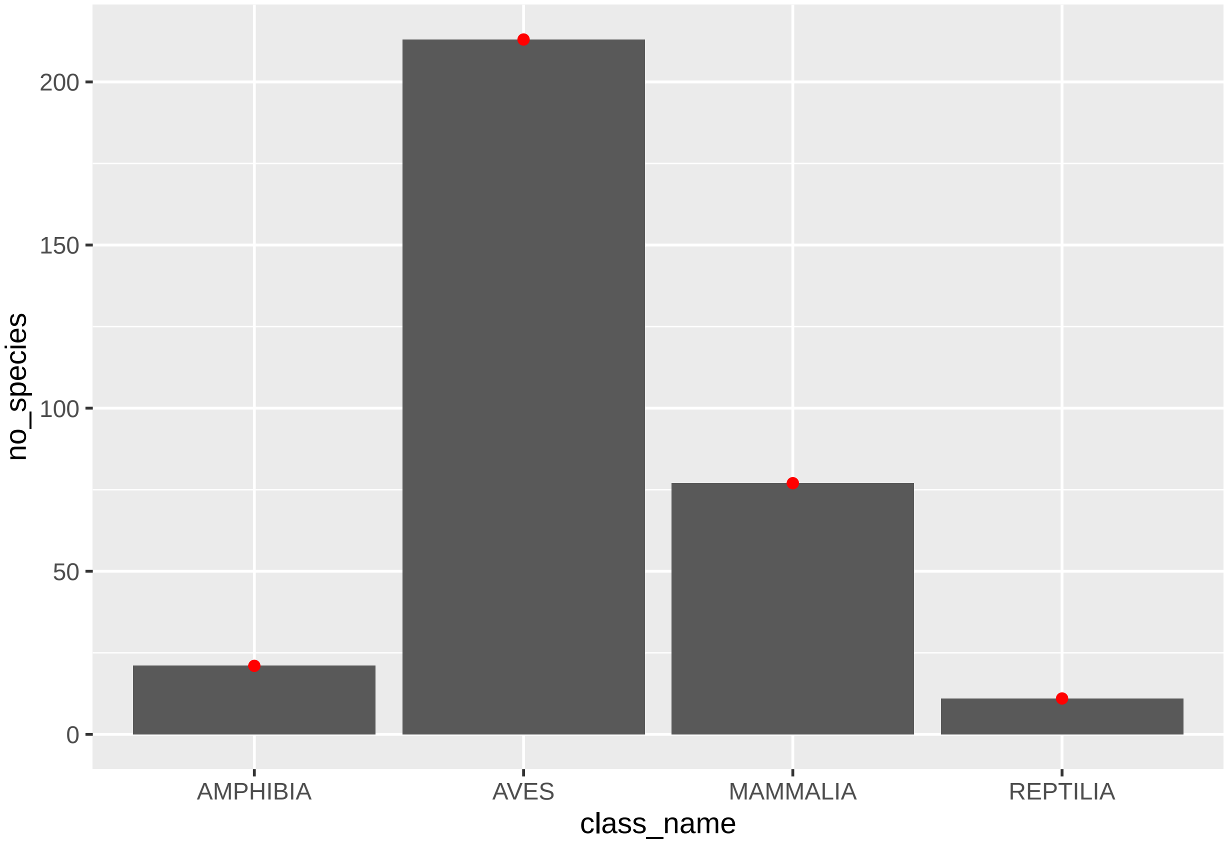
```
species_long %>%
  left_join(species_info, by=("binomial")) %>%
  group_by(class_name) %>% distinct(binomial) %>%
  summarise(no_species= n())
```

```
## # A tibble: 4 x 2
##   class_name no_species
##   <chr>           <int>
## 1 AMPHIBIA           21
## 2 AVES              213
## 3 MAMMALIA           77
## 4 REPTILIA           11
```

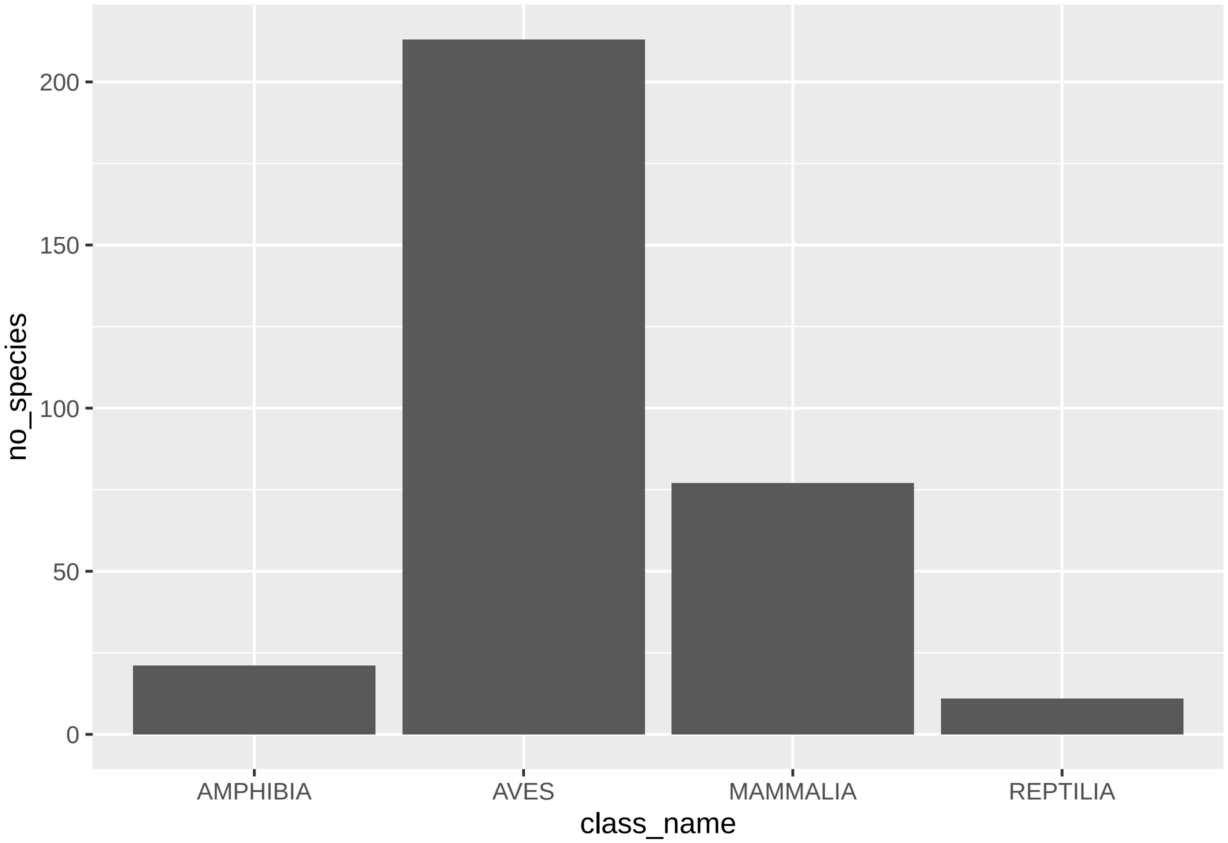
```
data_plot %>% ggplot(aes(class_name, no_species,
                         fill=class_name)) +
  geom_bar(stat="identity")
```



```
ggplot(data_plot, aes(class_name, no_species)) +
  geom_bar(stat="identity") + geom_point(colour="red")
```



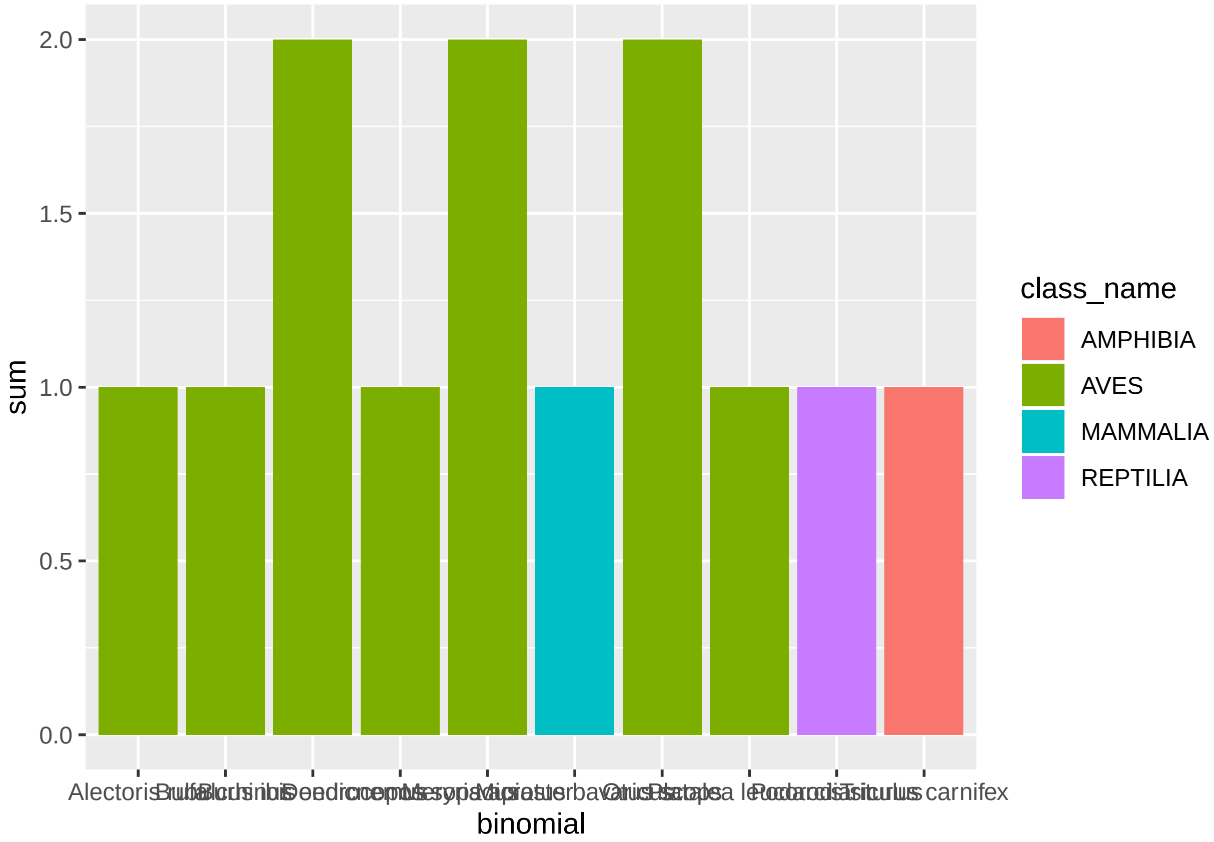
```
ggplot() +
  geom_bar(data=data_plot, aes(class_name, no_species),
           stat="identity")
```



Plot number of occurrences for the 10 least widely distributed species

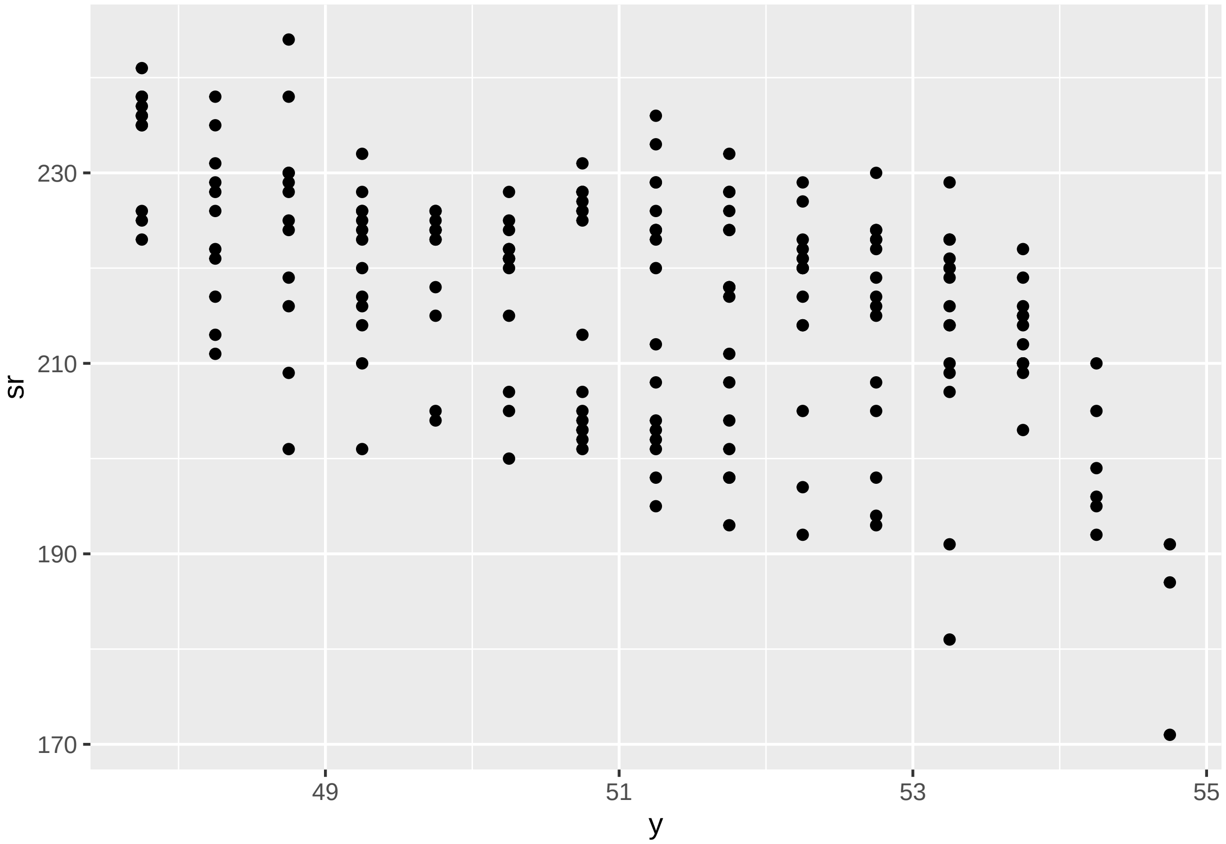
```
library(magrittr)
species_info %<>%
  distinct(binomial, class_name, order_name, family_nam)
species_long %>% drop_na() %>%
  left_join(species_info, by="binomial") %>%
  group_by(binomial, class_name, family_nam) %>%
  summarise(sum=sum(occurrence)) %>% ungroup() %>%
  top_n(-10) %>% # You can also use for example filter to extract the species
  ggplot(aes(binomial,sum, fill=class_name)) +
  geom_bar(stat="identity")
```

```
## Selecting by sum
```

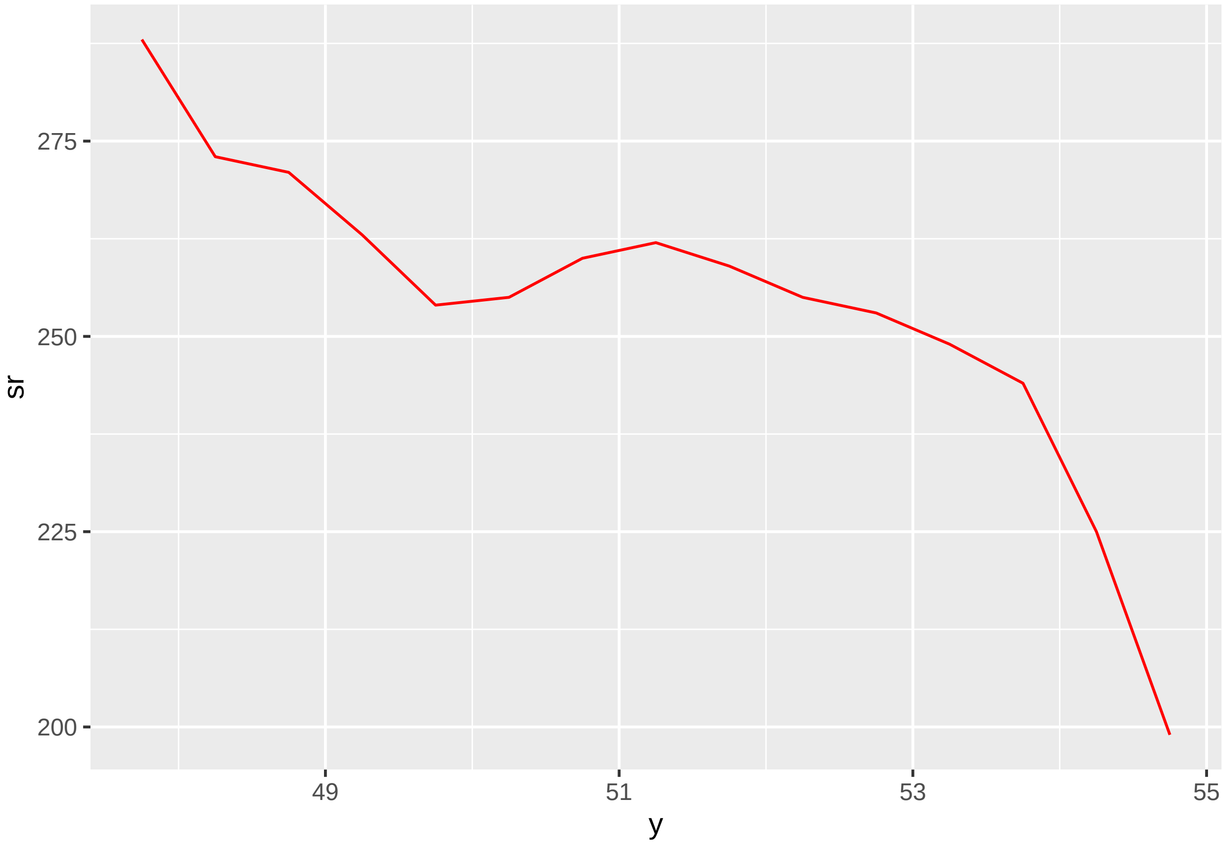


Plot species richness against latitude using different geoms Adjust labels and theme

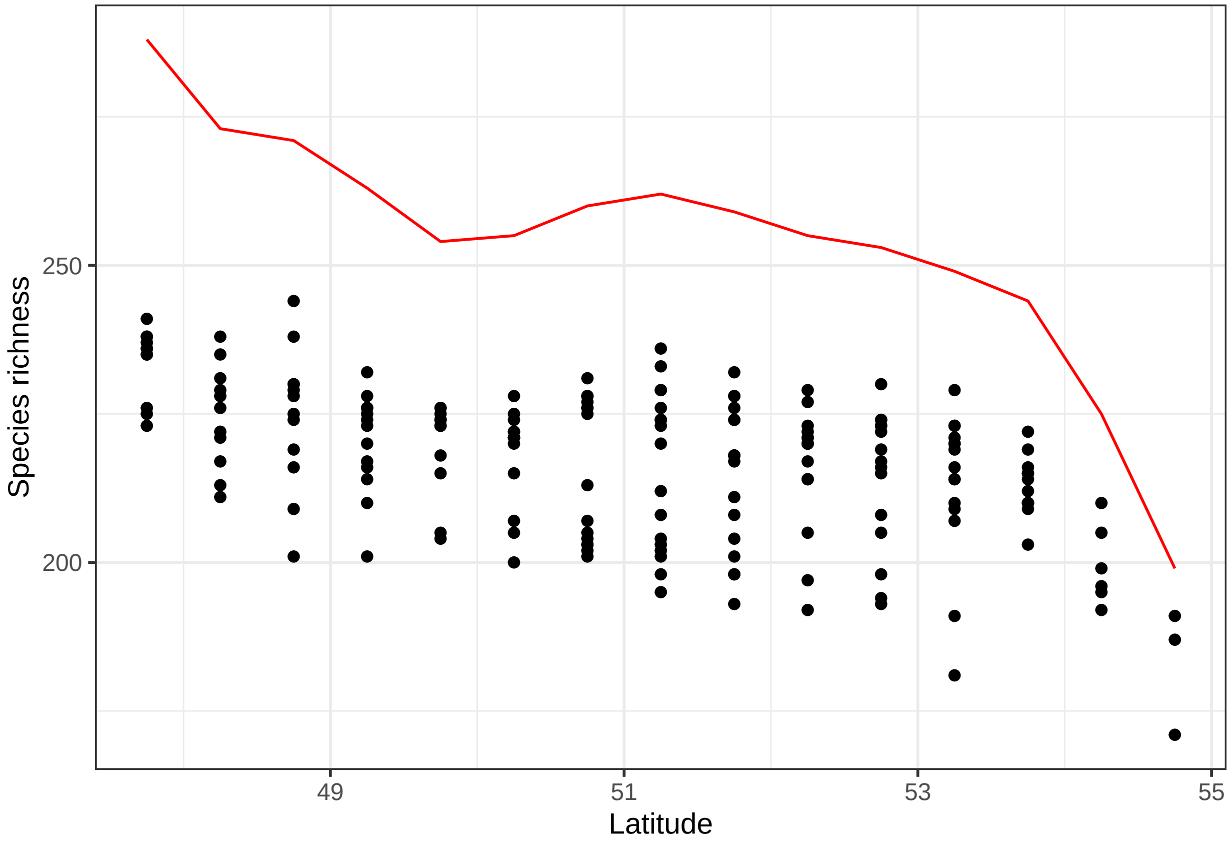
```
dat1 <- species_long %>% group_by(x,y) %>%
  summarise(sr=sum(occurrence))
ggplot(dat1, aes(y, sr)) + geom_point()
```



```
dat2 <- species_long %>% group_by(y) %>% distinct(binomial) %>%
  summarise(sr=n())
ggplot(dat2, aes(y,sr)) + geom_line(colour="red")
```

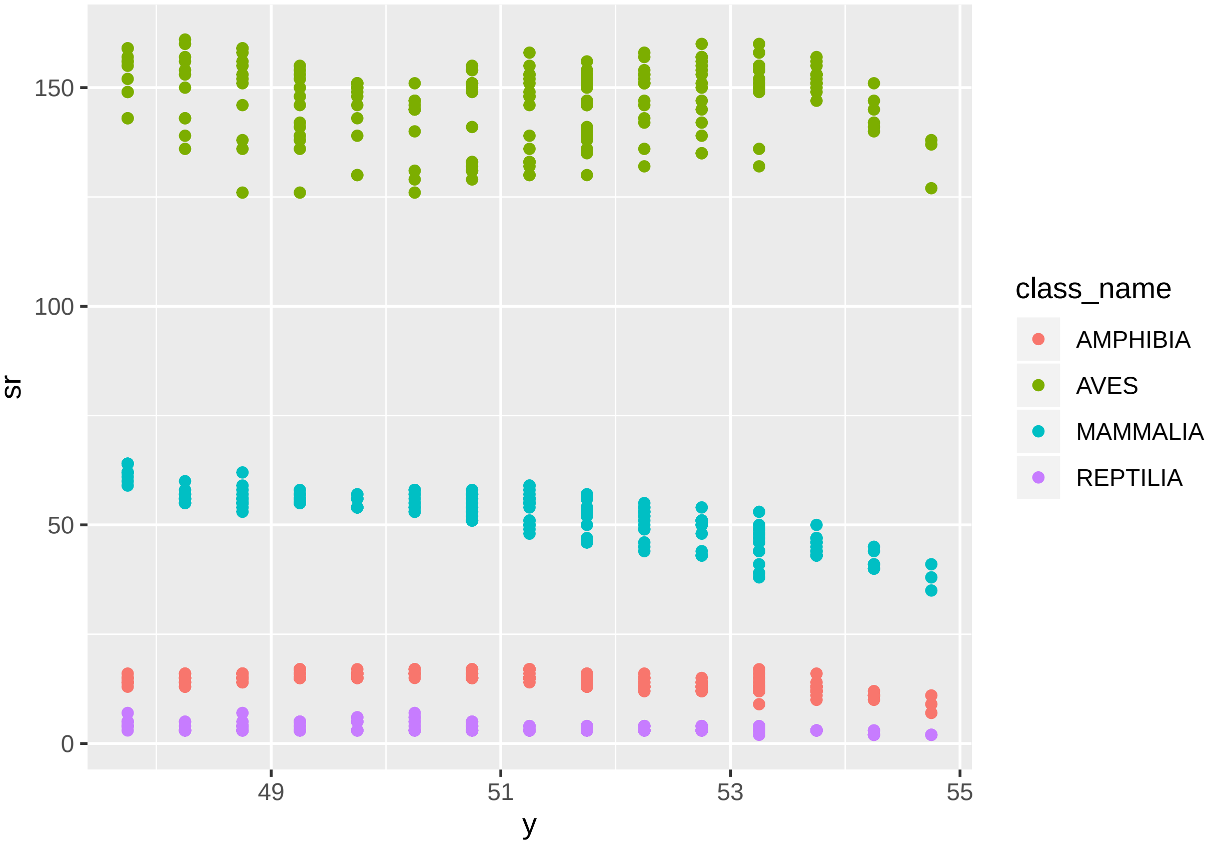


```
ggplot() + geom_point(data=dat1, aes(y, sr)) +
  geom_line(data=dat2, aes(y,sr), colour="red") +
  theme_bw() + labs(x="Latitude", y="Species richness")
```

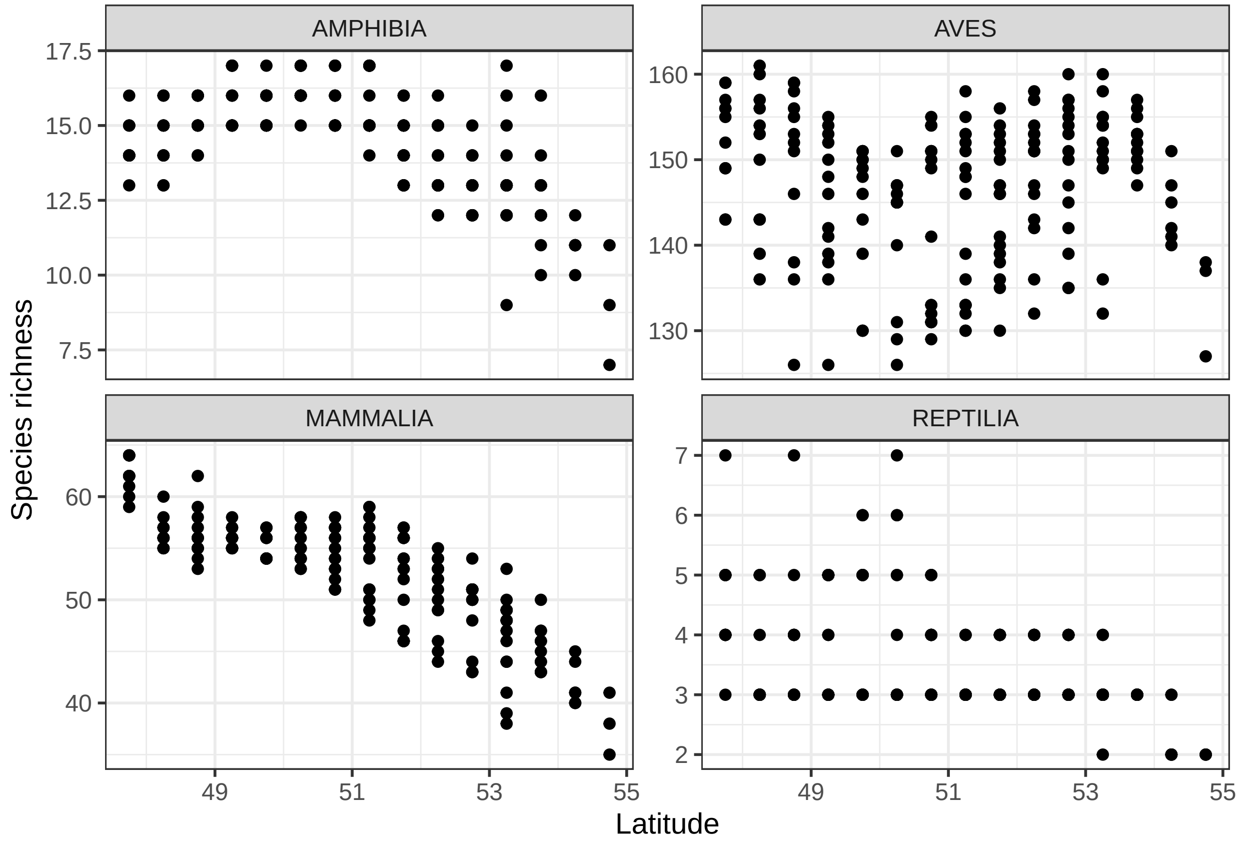


Use faceting to split the previous plots by class

```
species_long %>% drop_na() %>%
  left_join(species_info, by="binomial") %>%
  group_by(x,y,class_name) %>%
  summarise(sr=sum(occurrence)) %>%
  ggplot(aes(y, sr, colour=class_name)) + geom_point()
```



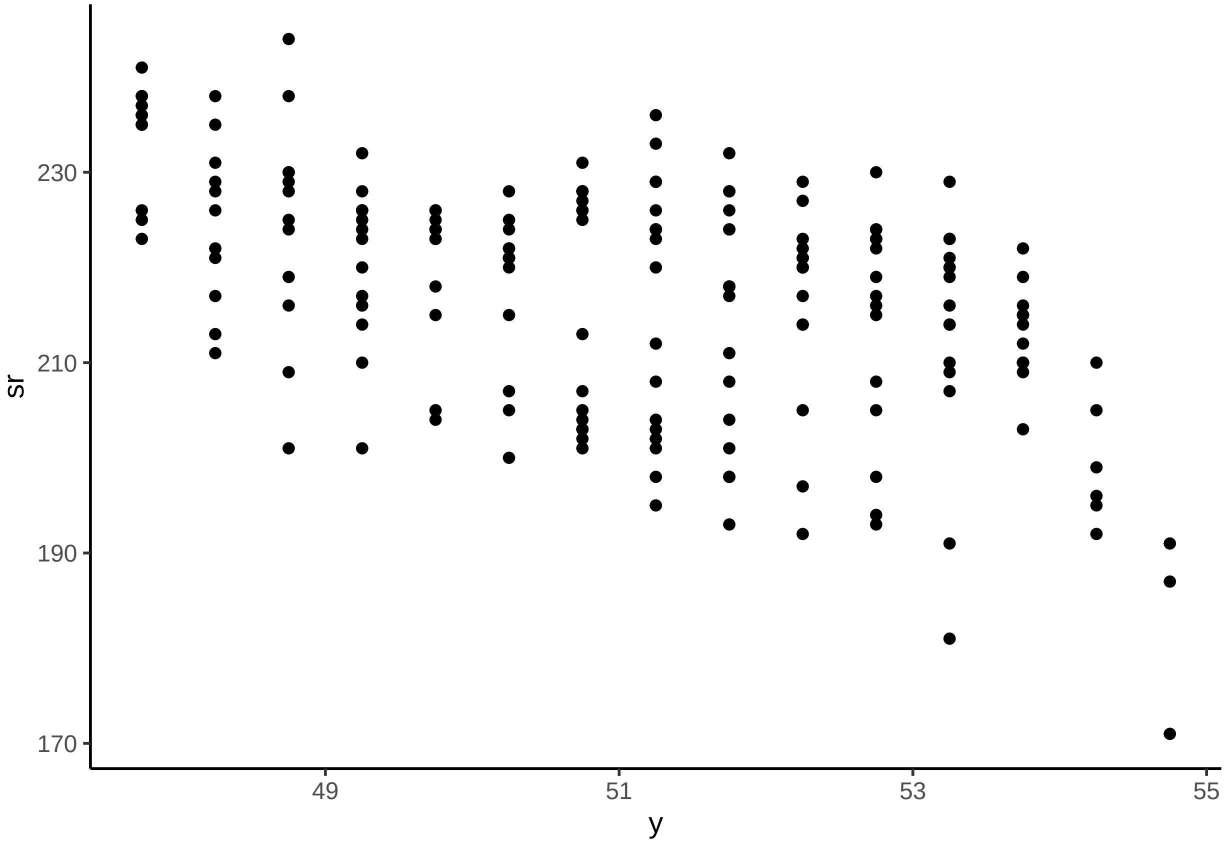
```
species_long %>% drop_na() %>%
  left_join(species_info, by="binomial") %>%
  group_by(x,y,class_name) %>%
  summarise(sr=sum(occurrence)) %>%
  ggplot(aes(y, sr)) + geom_point() +
  facet_wrap(~class_name, scales = "free_y") +
  theme_bw() + labs(x="Latitude", y="Species richness")
```



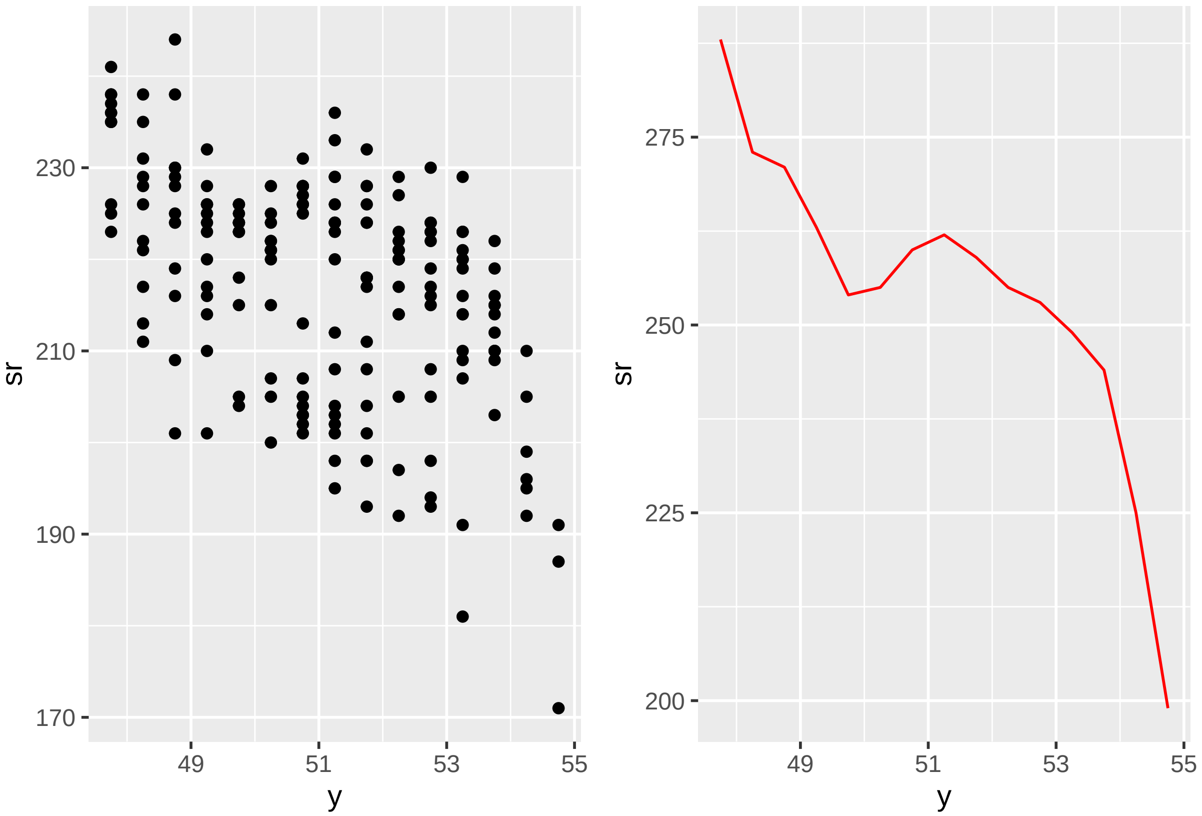
```
# Alternatively, you can also use facet_grid()
```

Use patchwork to create individual plots and combine them into one figure Install and load the patchwork package

```
#install.packages("devtools")
#devtools::install_github("thomasp85/patchwork")
library(patchwork)

dat1 <- species_long %>% group_by(x,y) %>%
  summarise(sr=sum(occurrence))
p1 <- ggplot(dat1, aes(y, sr)) + geom_point()
p1 + theme_classic()
```

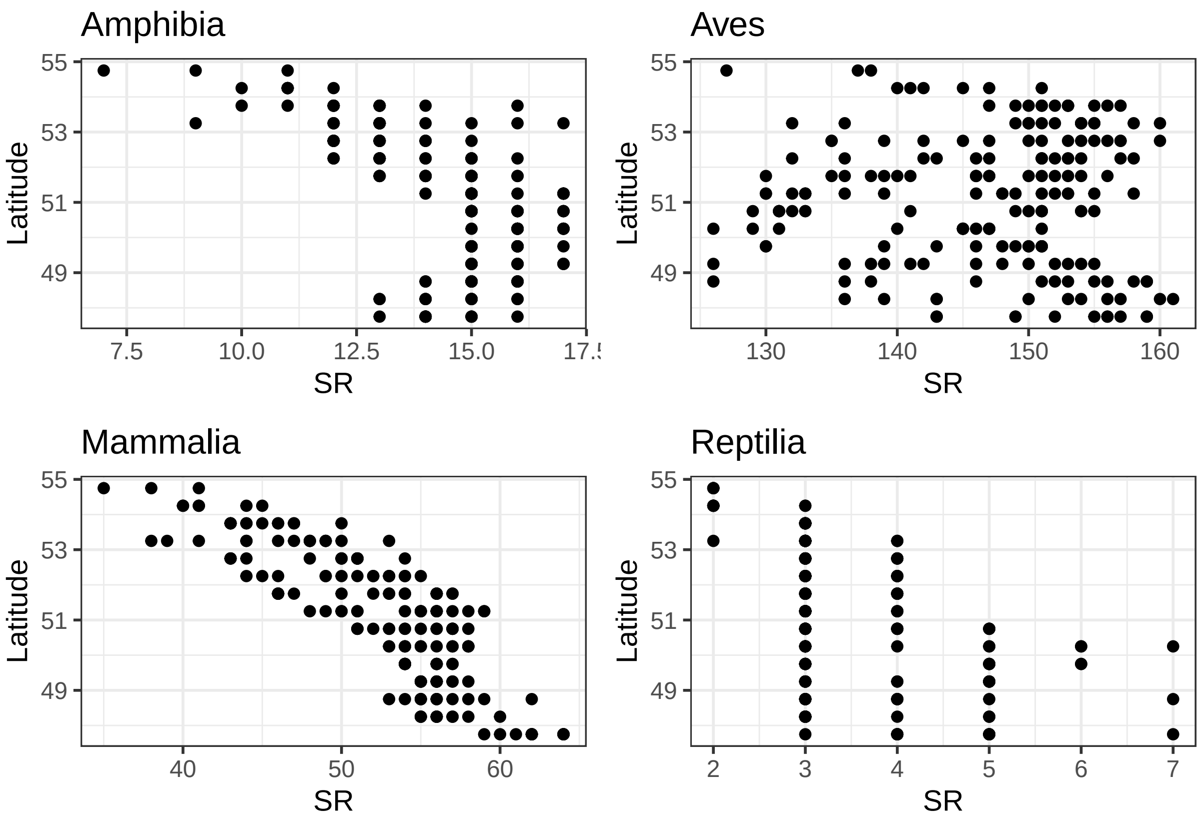


```
dat2 <- species_long %>% group_by(y) %>% distinct(binomial) %>%
  summarise(sr=n())
p2 <- ggplot(dat2, aes(y,sr)) + geom_line(colour="red")

p1 + p2
```



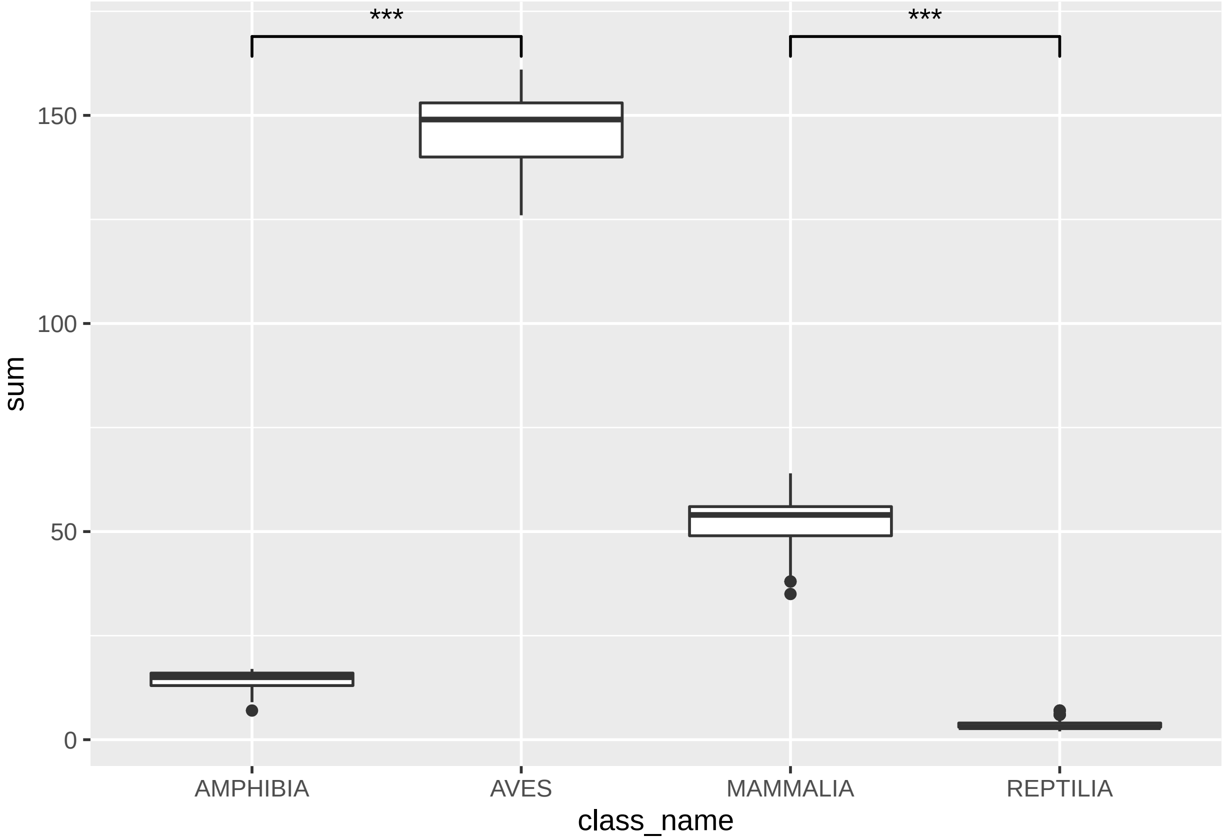
Create individual plots for each class and using patchwork display all plots combined in one

```
p1 <- species_long %>%
  left_join(species_info, by=("binomial")) %>%
  group_by(x,y,class_name) %>% filter(class_name == "AMPHIBIA") %>%
  summarise(sum=sum(occurrence)) %>% ggplot() +
  geom_point(aes(x=sum, y=y)) + theme_bw() +
  labs(x="SR", y="Latitude", title="Amphibia")
p2 <- species_long %>%
  left_join(species_info, by=("binomial")) %>%
  group_by(x,y,class_name) %>% filter(class_name == "AVES") %>%
  summarise(sum=sum(occurrence)) %>% ggplot() +
  geom_point(aes(x=sum, y=y)) + theme_bw() +
  labs(x="SR", y="Latitude", title="Aves")
p3 <- species_long %>%
  left_join(species_info, by=("binomial")) %>%
  group_by(x,y,class_name) %>% filter(class_name == "MAMMALIA") %>%
  summarise(sum=sum(occurrence)) %>% ggplot() +
  geom_point(aes(x=sum, y=y)) + theme_bw() +
  labs(x="SR", y="Latitude", title="Mammalia")
p4 <- species_long %>%
  left_join(species_info, by=("binomial")) %>%
  group_by(x,y,class_name) %>% filter(class_name == "REPTILIA") %>%
  summarise(sum=sum(occurrence)) %>% ggplot() +
  geom_point(aes(x=sum, y=y)) + theme_bw() +
  labs(x="SR", y="Latitude", title="Reptilia")

p1 + p2 + p3 + p4
```



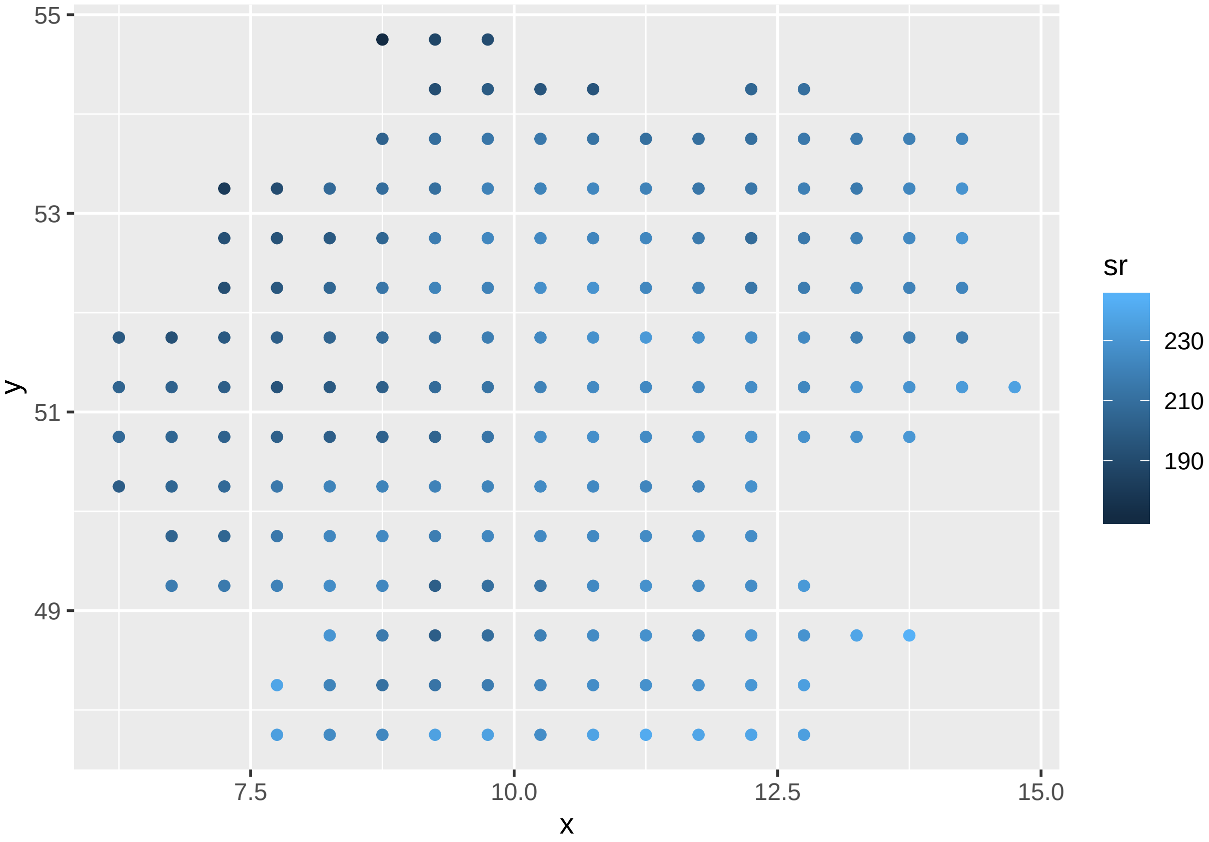
Plot boxplot of species richness against class and add significant differences with geom_signif() from the ggsignif package

```
#install.packages("ggsignif", dep=T)
library(ggsignif)

species_long %>%
  left_join(species_info, by=("binomial")) %>%
  group_by(x,y,class_name) %>%
  summarise(sum=sum(occurrence)) %>%
  ggplot(aes(x=class_name, y=sum)) +
  geom_boxplot() +
  geom_signif(comparisons = list(c("AMPHIBIA", "AVES"),
                                 c("REPTILIA", "MAMMALIA")),
              map_signif_level=TRUE)
```

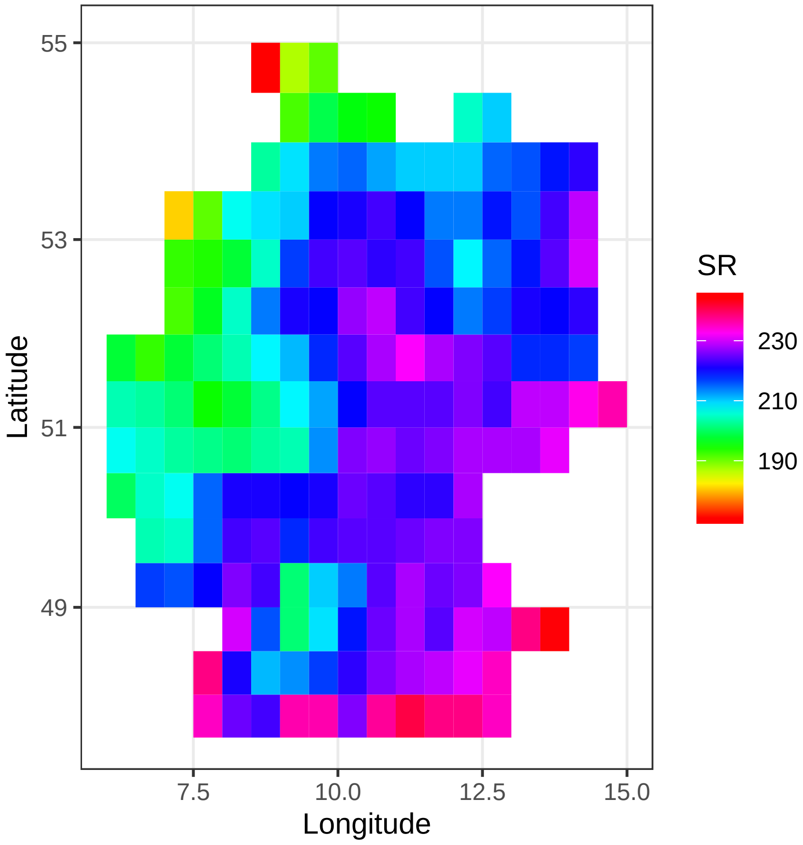


Make map of species richness

```
species_long %>% group_by(x,y) %>%
  summarise(sr=sum(occurrence)) %>%
  ggplot() + geom_point(aes(x,y,colour=sr))
```



```
species_long %>% group_by(x,y) %>%
  summarise(sr=sum(occurrence)) %>%
  ggplot() + geom_tile(aes(x,y,fill=sr)) +
  scale_fill_gradientn(name="SR", colours=rainbow(255)) +
  theme_bw() + coord_map() +
  labs(x="Longitude", y="Latitude")
```



```
##########################################################################
```

7. Data analysis with dplyr

Create linear model using do(), i.e. of species richness against latitude for each class_name

```
library(magrittr)
species_long <- read_csv("data/species_data.csv.xz") %>%
  gather(binomial, occurrence, -c(x,y)) %>% drop_na()
```

```
## Parsed with column specification:
## cols(
##   .default = col_double()
## )
```

```
## See spec(...) for full column specifications.
```

```
species_info <- read_csv("data/species_info.csv.xz")
```

```
## Parsed with column specification:
## cols(
##   binomial = col_character(),
##   presence = col_double(),
##   origin = col_double(),
##   seasonal = col_double(),
##   kingdom_na = col_character(),
##   phylum_nam = col_character(),
##   class_name = col_character(),
##   order_name = col_character(),
##   family_nam = col_character(),
##   shape_Area = col_double()
## )
```

```
species_info %<>%
  distinct(binomial, class_name, order_name, family_nam)

models <- species_long %>% left_join(species_info, by="binomial") %>%
  group_by(x,y,class_name) %>% summarise(sr=sum(occurrence)) %>%
  group_by(class_name) %>% do(model=lm(sr ~ y, data=.))
```

Extract coefficients of linear model

```
ctl <- c(4.17,5.58,5.18,6.11,4.50,4.61,5.17,4.53,5.33,5.14)
trt <- c(4.81,4.17,4.41,3.59,5.87,3.83,6.03,4.89,4.32,4.69)
group <- gl(2, 10, 20, labels = c("Ctl","Trt"))
weight <- c(ctl, trt)
mod <- lm(weight ~ group)
mod
```

```
##
## Call:
## lm(formula = weight ~ group)
##
## Coefficients:
## (Intercept)     groupTrt
##       5.032       -0.371
```

```
summary(mod)
```

```
##
## Call:
## lm(formula = weight ~ group)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.0710 -0.4938  0.0685  0.2462  1.3690
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   5.0320     0.2202  22.850 9.55e-15 ***
## groupTrt     -0.3710     0.3114  -1.191    0.249
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6964 on 18 degrees of freedom
## Multiple R-squared:  0.07308,    Adjusted R-squared:  0.02158
## F-statistic: 1.419 on 1 and 18 DF,  p-value: 0.249
```

```
#models$model[[1]]$coefficients
#coef(summary(models$model[[1]]))

models %>% do(data.frame(class = .$class_name,
                         var=names(coef(.$model)),
                         coef(.$model)))
```

```
## Warning in bind_rows_(x, .id): Unequal factor levels: coercing to character

## Warning in bind_rows_(x, .id): binding character and factor vector, coercing
## into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector, coercing
## into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector, coercing
## into character vector

## Warning in bind_rows_(x, .id): binding character and factor vector, coercing
## into character vector

## Source: local data frame [8 x 3]
## Groups: <by row>
##
## # A tibble: 8 x 3
##   class    var         coef...model.
## * <chr>    <fct>               <dbl>
## 1 AMPHIBIA (Intercept)       41.9
## 2 AMPHIBIA y                 -0.539
## 3 AVES     (Intercept)      151.
## 4 AVES     y                 -0.0814
## 5 MAMMALIA (Intercept)      179.
## 6 MAMMALIA y                 -2.48
## 7 REPTILIA (Intercept)       15.2
## 8 REPTILIA y                 -0.230
```

Alternatively

```
library(broom)
# glance(), tidy(), augment()
tidy(models$model[[1]])
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   41.9      2.85       14.7   4.40e-33
## 2 y             -0.539    0.0557     -9.68 3.24e-18
```

```
glance(models$model[[1]])
```

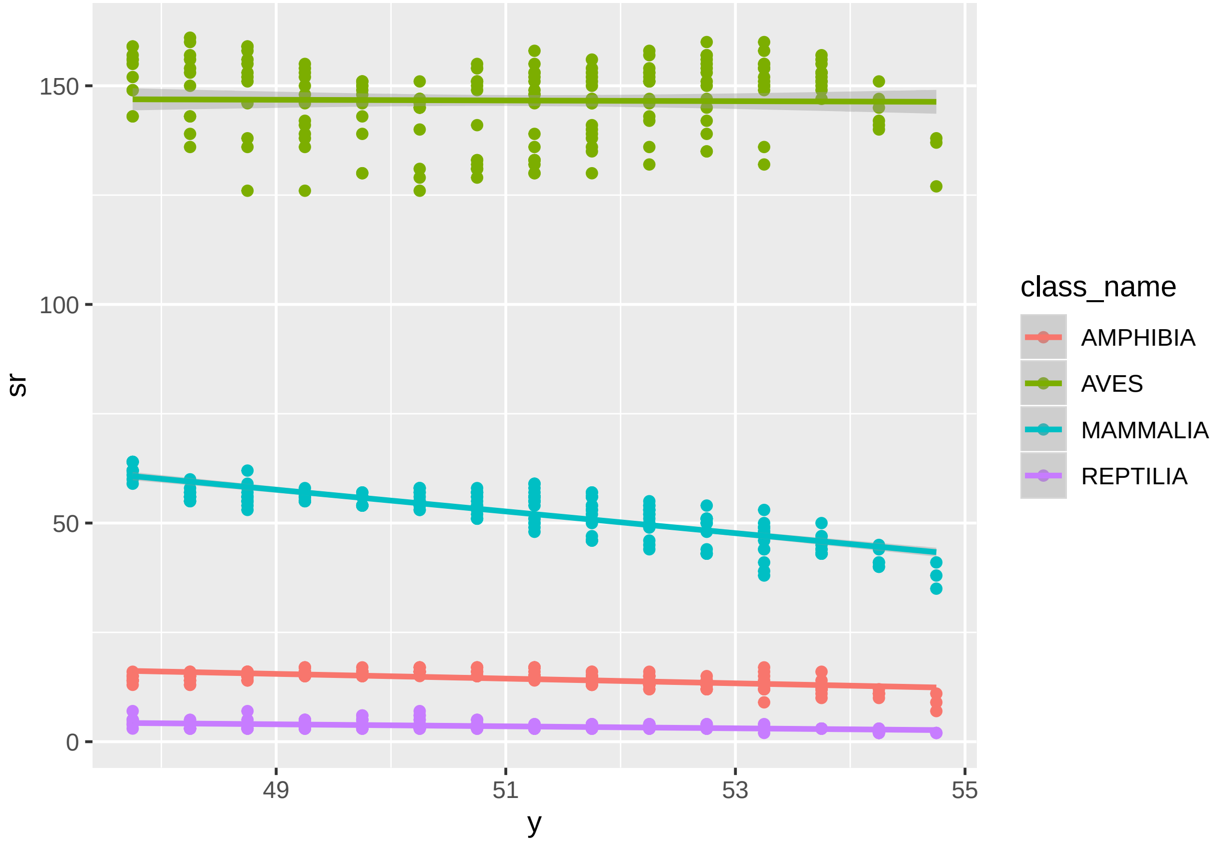
```
## # A tibble: 1 x 11
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <int>  <dbl> <dbl> <dbl>
## 1     0.334         0.330  1.45      93.7 3.24e-18     2  -338.  681.  691.
## # ... with 2 more variables: deviance <dbl>, df.residual <int>
```

```
models %>% do(tidy(.$model))
```

```
## Source: local data frame [8 x 5]
## Groups: <by row>
##
## # A tibble: 8 x 5
##   term        estimate std.error statistic  p.value
## * <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)  41.9       2.85      14.7   4.40e-33
## 2 y            -0.539     0.0557    -9.68  3.24e-18
## 3 (Intercept) 151.       17.2        8.77  1.08e-15
## 4 y            -0.0814    0.336     -0.242 8.09e- 1
## 5 (Intercept) 179.        6.39      28.1   5.47e-69
## 6 y            -2.48      0.125    -19.9   5.43e-48
## 7 (Intercept)  15.2       1.63       9.34  2.93e-17
## 8 y            -0.230     0.0319    -7.20  1.44e-11
```

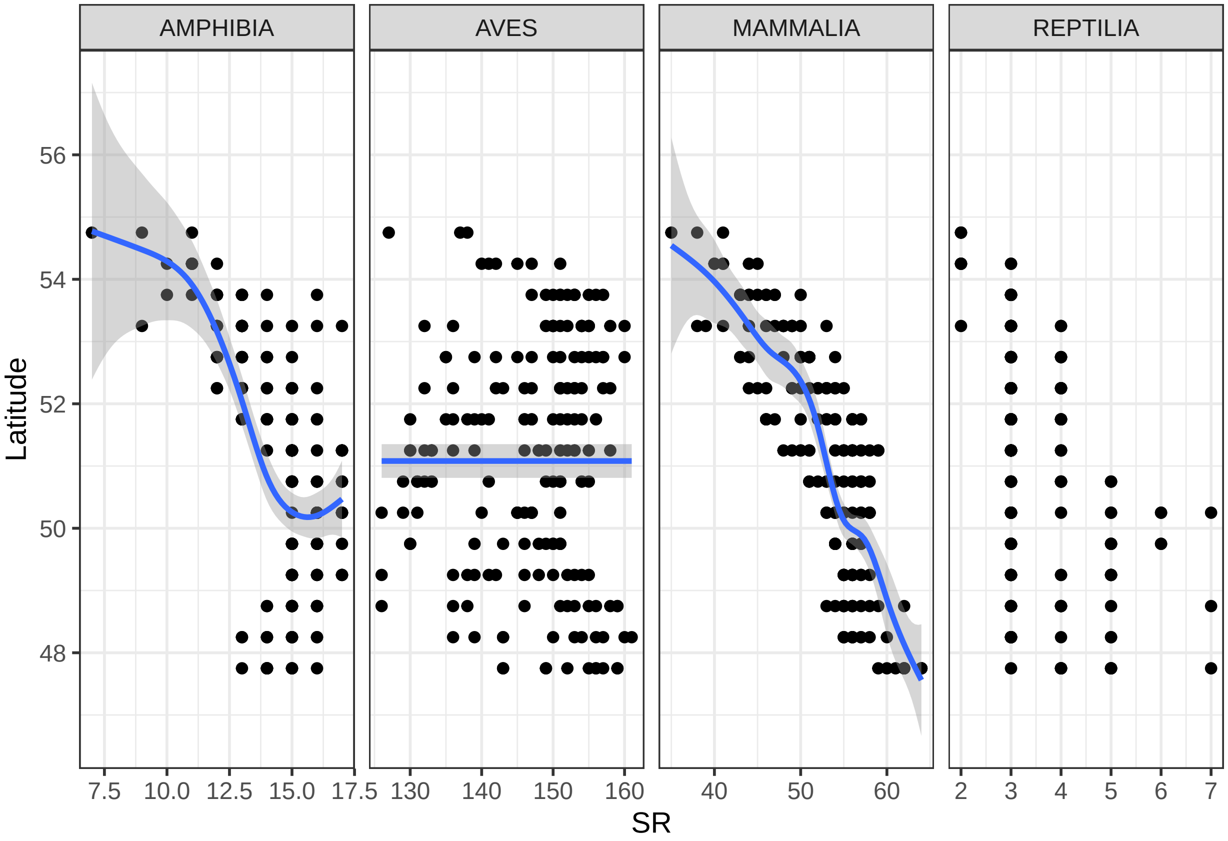
Run and plot gam of species richness against latitude for each class using ggplot2 (geom_smooth)

```
species_long %>% left_join(species_info, by="binomial") %>%
  group_by(x,y,class_name) %>% summarise(sr=sum(occurrence)) %>%
  ggplot(aes(y,sr, colour=class_name)) +
  geom_point() + geom_smooth(method="gam")
```



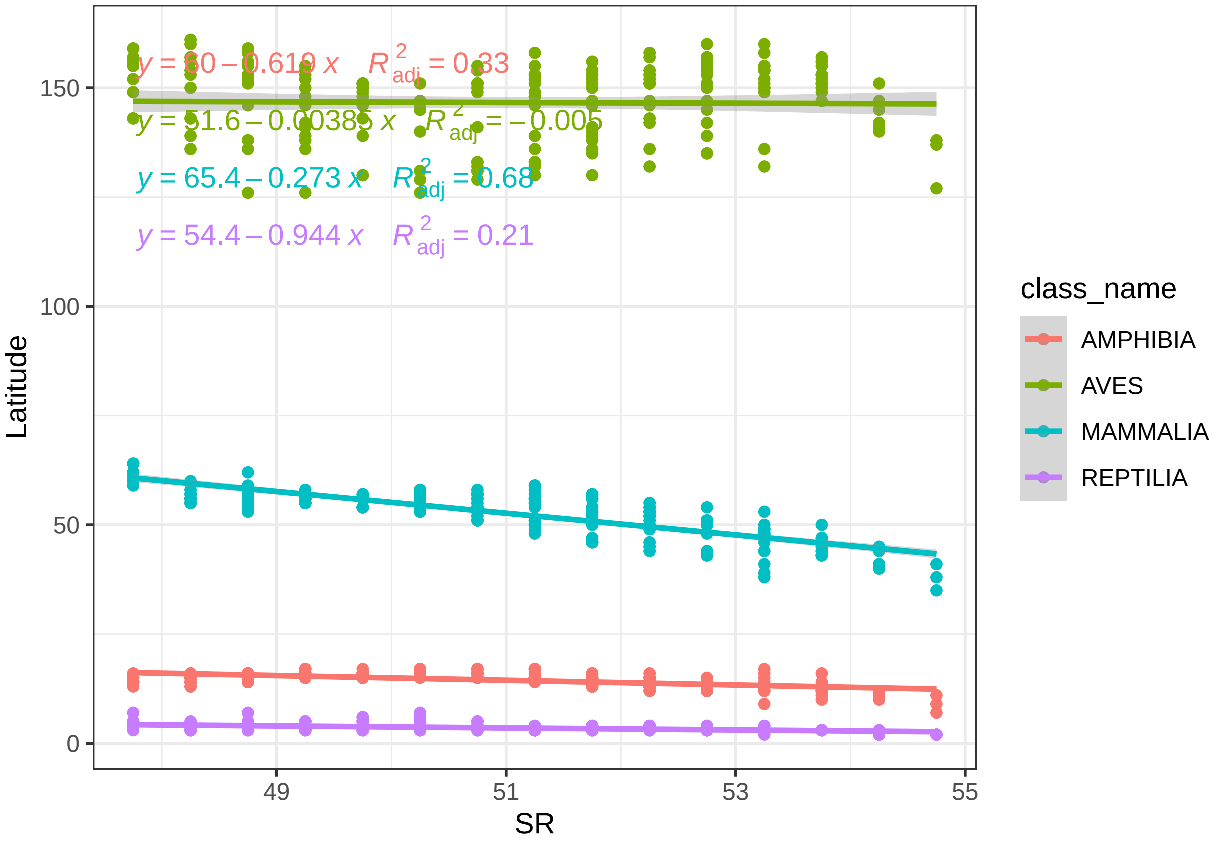
```
species_long %>%
  left_join(species_info, by=("binomial")) %>%
  group_by(x,y,class_name) %>% summarise(sr=sum(occurrence)) %>%
  ggplot(aes(x=sr, y=y)) + geom_point() +
  facet_grid(.~class_name, scales="free_x") +
  geom_smooth(method="gam", formula = y ~ s(x, bs = "cs")) +
  labs(x="SR", y="Latitude") + theme_bw()
```

```
## Warning: Computation failed in `stat_smooth()`:
## x has insufficient unique values to support 10 knots: reduce k.
```

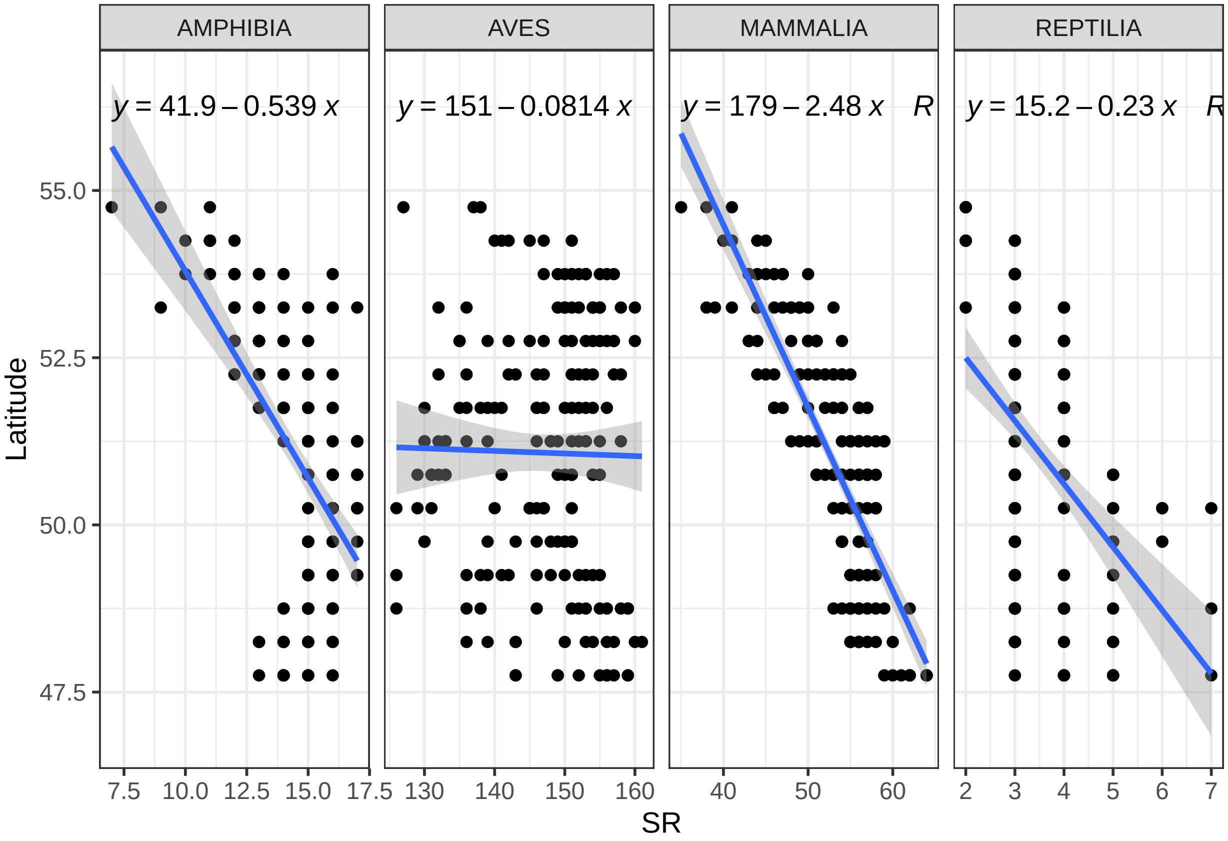


Run and plot lm of species richness against latitude for each class using ggplot2 and add equation to plot using stat_poly_eq() of the ggpmisc package

```r
#install.packages("ggpmisc", dep=T)
library(ggpmisc)

# One plot with different colours
species_long %>% left_join(species_info, by="binomial") %>%
  group_by(x,y,class_name) %>% summarise(sr=sum(occurrence)) %>%
  ggplot(aes(y,sr, colour=class_name)) +
  geom_point() + geom_smooth(method="lm") +
  stat_poly_eq(formula = x~y,
               aes(colour=class_name,
                   label =  paste(stat(eq.label),
                          stat(adj.rr.label), sep = "~~~~")),
               parse = TRUE) +
  labs(x="SR", y="Latitude") + theme_bw()
```

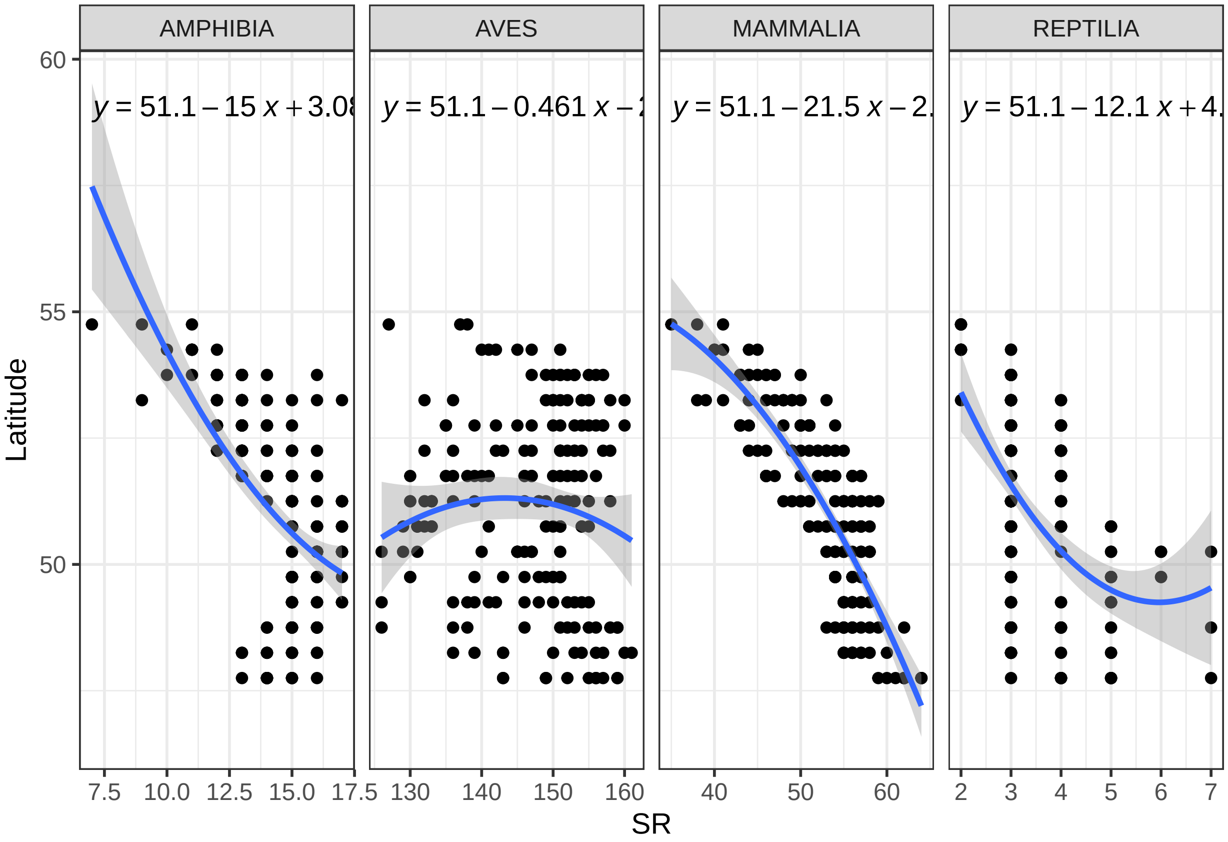


```
# Panel plot with different panels
species_long %>%
  left_join(species_info, by=("binomial")) %>%
  group_by(x,y,class_name) %>% summarise(sr=sum(occurrence)) %>%
  ggplot(aes(x=sr, y=y)) + geom_point() +
  facet_grid(.~class_name, scales="free_x") +
  geom_smooth(method="lm") +
  stat_poly_eq(formula = x~y,
               aes(label =  paste(stat(eq.label),
                                  stat(adj.rr.label), sep = "~~~~")),
               parse = TRUE) +
  labs(x="SR", y="Latitude") + theme_bw()
```



We can do the same using a quadratic term

```
species_long %>%
  left_join(species_info, by=("binomial")) %>%
  group_by(x,y,class_name) %>% summarise(sr=sum(occurrence)) %>%
  ggplot(aes(x=sr, y=y)) + geom_point() +
  facet_grid(.~class_name, scales="free_x") +
  geom_smooth(method="lm", formula= y ~ poly(x, 2)) +
  stat_poly_eq(formula = y ~ poly(x, 2),
               aes(label =  paste(stat(eq.label),
                                  stat(adj.rr.label), sep = "~~~~")),
               parse = TRUE) +
  labs(x="SR", y="Latitude") + theme_bw()
```



Note: Than you have to specify the quadratic term in the formula argument of both functions, geom_smooth and stat_poly_eq. #################### Run ANOVA to compare linear models of linear versus quadratic term using the tidy() function of the broom package

```
#install.packages("broom", dep=T)
library(broom)

species_long %>%
  left_join(species_info, by=("binomial")) %>%
  group_by(x,y,class_name) %>% summarise(sr=sum(occurrence)) %>%
  group_by(class_name) %>% do(
    mod_linear = lm(sr ~ y, data = .),
    mod_quad = lm(sr ~ poly(y, 2), data = .)) %>%
  do(aov = anova(.$mod_linear, .$mod_quad)) %>%
  rowwise %>% do(tidy(.$aov)) %>% tidyr::drop_na()
```

```
## Warning: Unknown or uninitialised column: 'term'.

## Warning: Unknown or uninitialised column: 'term'.

## Warning: Unknown or uninitialised column: 'term'.

## Warning: Unknown or uninitialised column: 'term'.

## # A tibble: 4 x 6
##   res.df   rss    df  sumsq statistic  p.value
##    <dbl> <dbl> <dbl>  <dbl>     <dbl>    <dbl>
## 1    186  231.     1 162.        131.  2.88e-23
## 2    186 13695.     1 648.        8.80 3.40e- 3
## 3    186  1764.     1 215.       22.7  3.87e- 6
## 4    186   128.     1   1.55       2.25 1.35e- 1
```

Note: For the Anova, you also need the rowwise function, which you do not need for extracting the output of lm().

```
######################################################################
```

8. Spatio-temporal data with stars and sf

Read tas_ewembi_deu_1981_2010.nc using the read_ncdf() function of the stars package, turn into data.frame and drop NAs tas_ewembi_deu_1981_2010.nc contains daily temperature data for Germany at a spatial resolution of 0.5 degree from 1981 - 2010

```
#install.packages("stars", dep=T)
library(stars)

env <- read_ncdf("data/tas_ewembi_deu_1981_2010.nc") %>%
  as.data.frame() %>% drop_na()
```

Define timestamp of date variable (z) using as.Date()

```
# Start date 1/01/1981, End date 31/12/2010
env$z <- factor(env$z, labels=seq(as.Date("1981/01/01"),
                                  as.Date("2010/12/31"), "days"))
head(env)
```

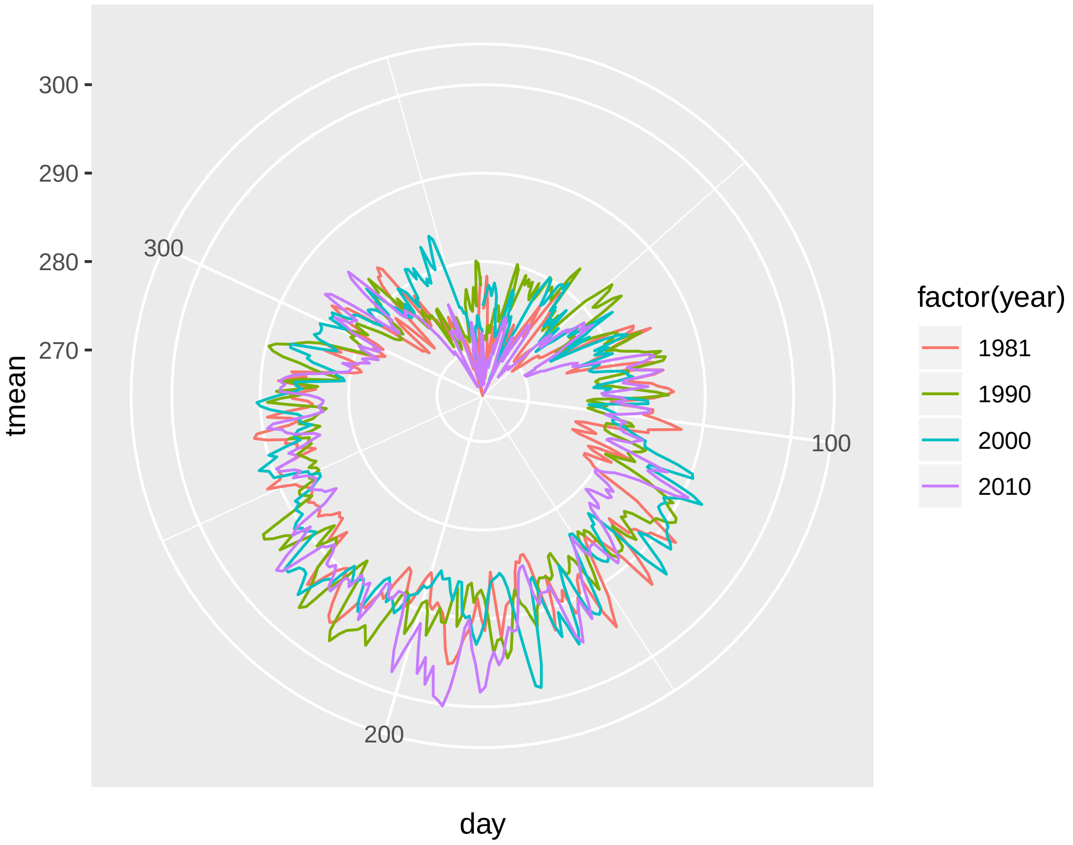
```
##    longitude latitude          z variable
## 6       8.75    54.75 1981-01-01 275.2337
## 7       9.25    54.75 1981-01-01 274.9547
## 8       9.75    54.75 1981-01-01 275.2550
## 25      9.25    54.25 1981-01-01 275.1317
## 26      9.75    54.25 1981-01-01 275.2247
## 27     10.25    54.25 1981-01-01 275.4385
```

Calculate mean temperature per date Split date into yday and year using the lubridate package And make a polar plot of daily mean temperature coloured by year

```
#install.packages("lubridate", dep=T)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date
```

```
env %>% group_by(z) %>%
  summarise(tmean=mean(variable)) %>%
  mutate(day = yday(z), year = year(z)) %>%
  filter(year %in% c(1981, 1990, 2000, 2010)) %>%
  ggplot(aes(x=day, y=tmean, colour=factor(year))) +
  geom_line() + coord_polar()
```

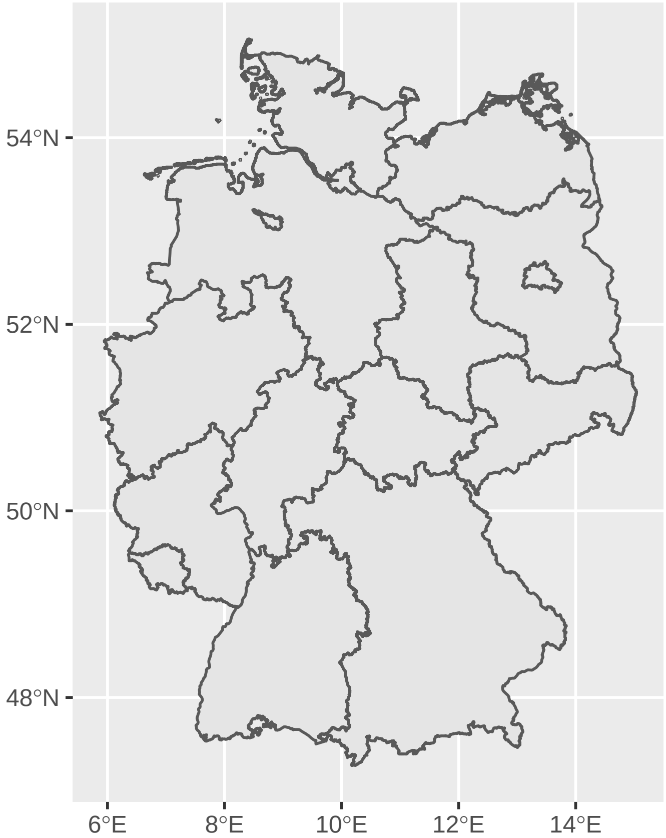


Plot map of 30-year average temperature Read gadm36_DEU_adm1.shp into R using st_read() of the sf package Add polygon of Germany to map using geom_sf

```
#install.packages("sf", dep=T)
library(sf)

# Read shapefile of Germany
deu <- st_read("data/gadm36_DEU_adm1.shp")
```

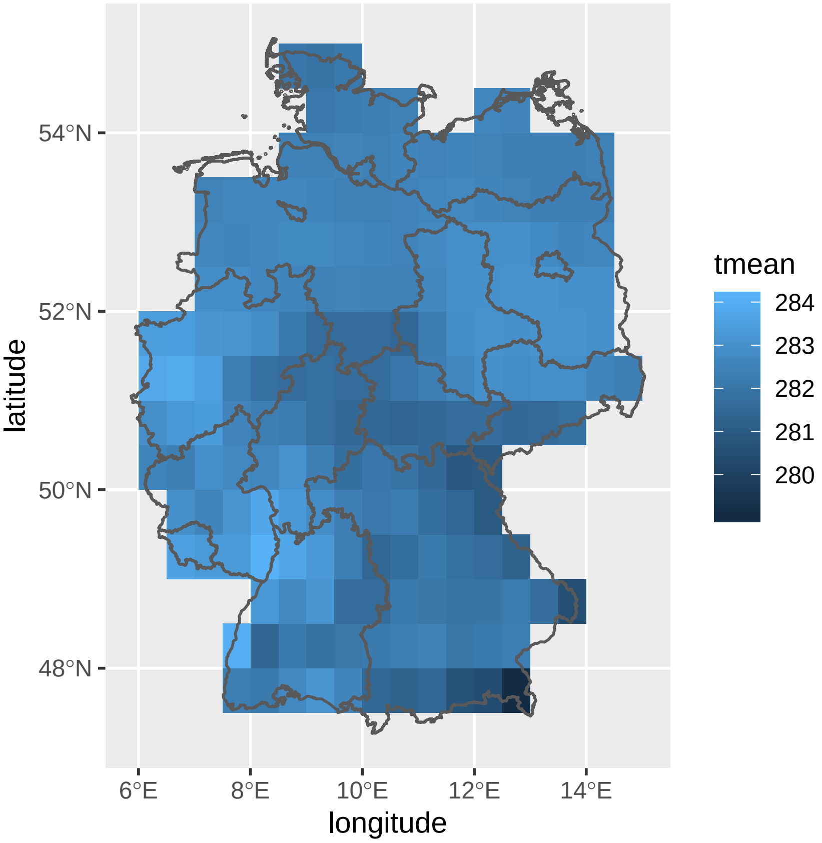
```
## Reading layer `gadm36_DEU_adm1' from data source `/home/matt/Documents/Github/dmvR/data/gadm36_DEU_ad
## Simple feature collection with 16 features and 0 fields
## geometry type:  MULTIPOLYGON
## dimension:      XY
## bbox:           xmin: 5.866251 ymin: 47.27012 xmax: 15.04181 ymax: 55.05653
## epsg (SRID):    4326
## proj4string:    +proj=longlat +datum=WGS84 +no_defs
```

```
#bav <- deu %>% filter(NAME_1=="Bayern")

# Plot the shapefile
deu %>% ggplot() + geom_sf()
```



Note: geom_sf() automatically plots the shapefile with a black outline and grey filling, which is not meaningful if we want to overlay the shapefile onto our data, thus we need to specify the fill argument, in the plot below

```
env %>% group_by(longitude,latitude) %>%
  summarise(tmean=mean(variable)) %>%
  ggplot() + geom_tile(aes(x=longitude, y=latitude,
                          fill=tmean)) +
  geom_sf(data=deu, fill="transparent")
```



For every grid cell calculate monthly mean temperature per year using as.yearmon() from the zoo package. Make a linear model using do() and extract Intercept, Slope and R2 values per grid cell.

```
#install.packages("zoo", dep=T)
library(zoo)
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
mod_lm <- env %>% mutate(yearmon = as.yearmon(z)) %>%
  group_by(yearmon, latitude, longitude) %>%
  summarise(tmean=mean(variable)) %>%
  group_by(longitude,latitude) %>%
  do(mod = lm(tmean ~ yearmon, data = .)) %>%
  mutate(Intercept = summary(mod)$coeff[1],
         Slope = summary(mod)$coeff[2],
         R2 = summary(mod)$r.squared) %>%
  dplyr::select(-mod)
```

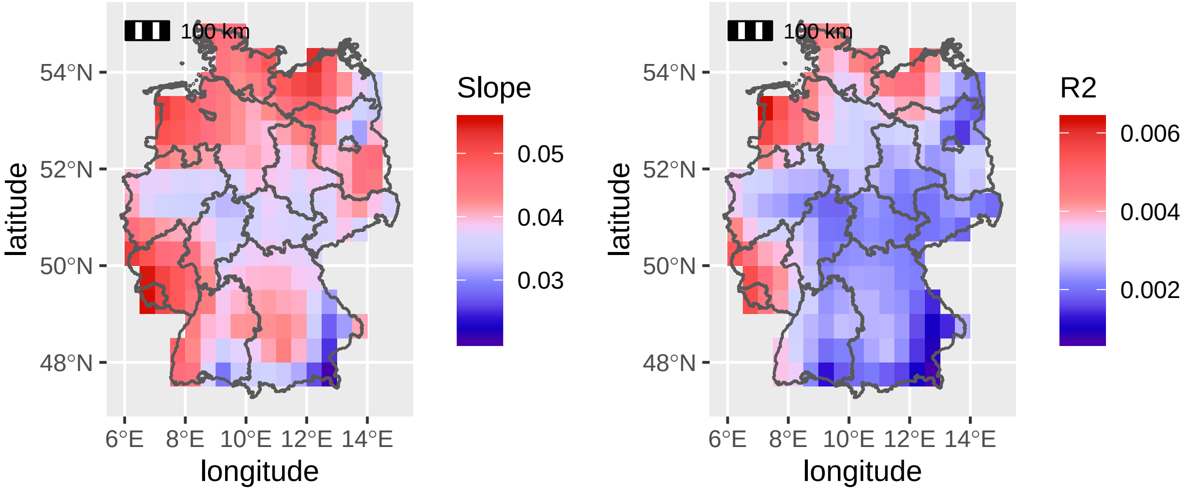
Plot map of slope and R2 With ggspatial, you can add a scale bar to your map using the annotation_scale() function

```
#install.packages("ggspatial", dep=T)
library(ggspatial)
```

The ggsci package, provides easy to use scientific color palettes for ggplot2

```
#install.packages("ggsci", dep=T)
library(ggsci)

p1 <- mod_lm %>% ggplot() +
  geom_tile(aes(x=longitude, y=latitude, fill=Slope)) +
  scale_fill_gsea() +
  geom_sf(data=deu, fill="NA") +
  annotation_scale(location="tl") +
  coord_sf()
p2 <- mod_lm %>% ggplot() +
  geom_tile(aes(x=longitude, y=latitude, fill=R2)) +
  scale_fill_gsea() +
  geom_sf(data=deu, fill="NA") +
  annotation_scale(location="tl") +
  coord_sf()

p1 + p2
```

```
## Scale on map varies by more than 10%, scale bar may be inaccurate
## Scale on map varies by more than 10%, scale bar may be inaccurate
```



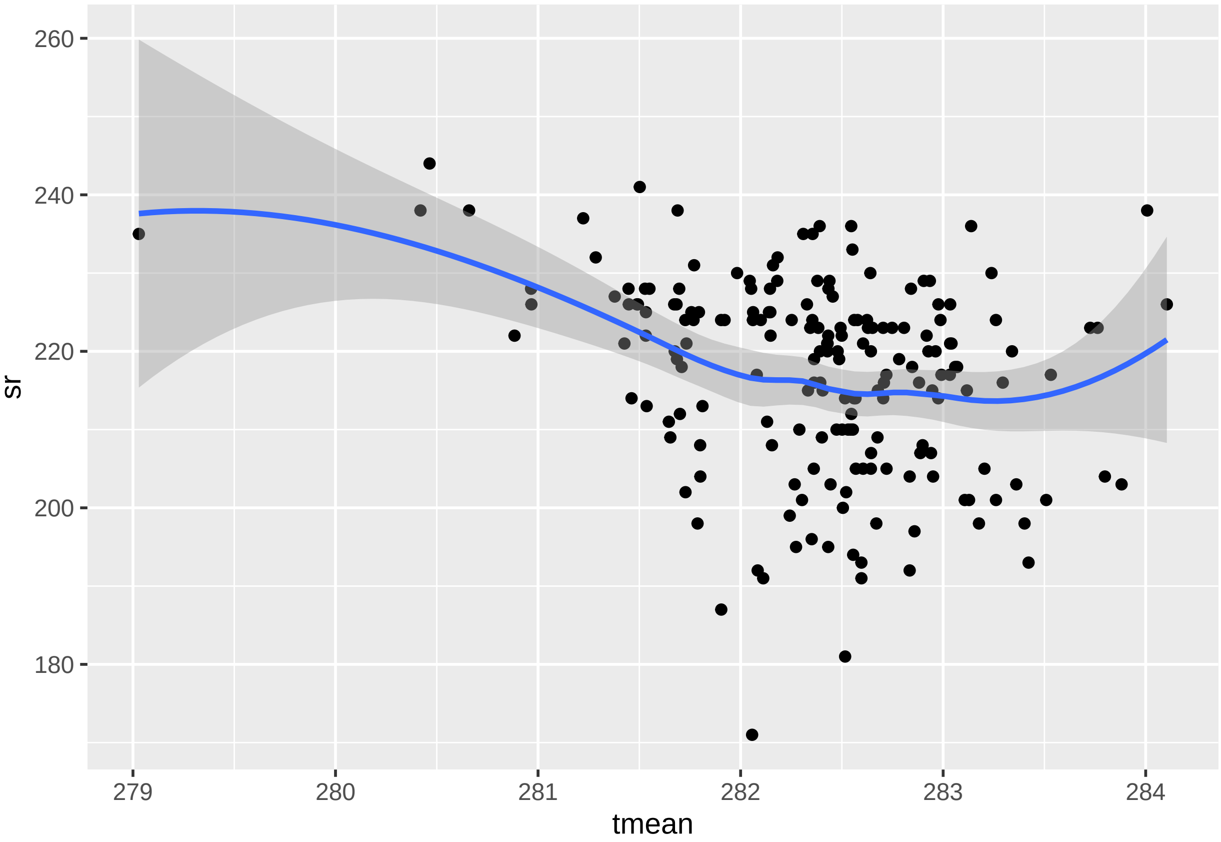
Join species_long with species_info and then join 30-year average temperature per year with species data

```
species_all <- species_long %>%
  left_join(species_info, by=("binomial"))
env_sp <- env %>% mutate(yearmon = as.yearmon(z)) %>%
  group_by(latitude, longitude) %>%
  summarise(tmean=mean(variable)) %>%
  right_join(species_all,
             by=c("longitude"="x", "latitude"="y"))
```

Plot richness against temperature

```
env_sp %>% group_by(longitude,latitude) %>%
  summarise(sr=sum(occurrence), tmean=mean(tmean)) %>%
  ggplot(aes(x=tmean, y=sr)) + geom_point() + geom_smooth()
```

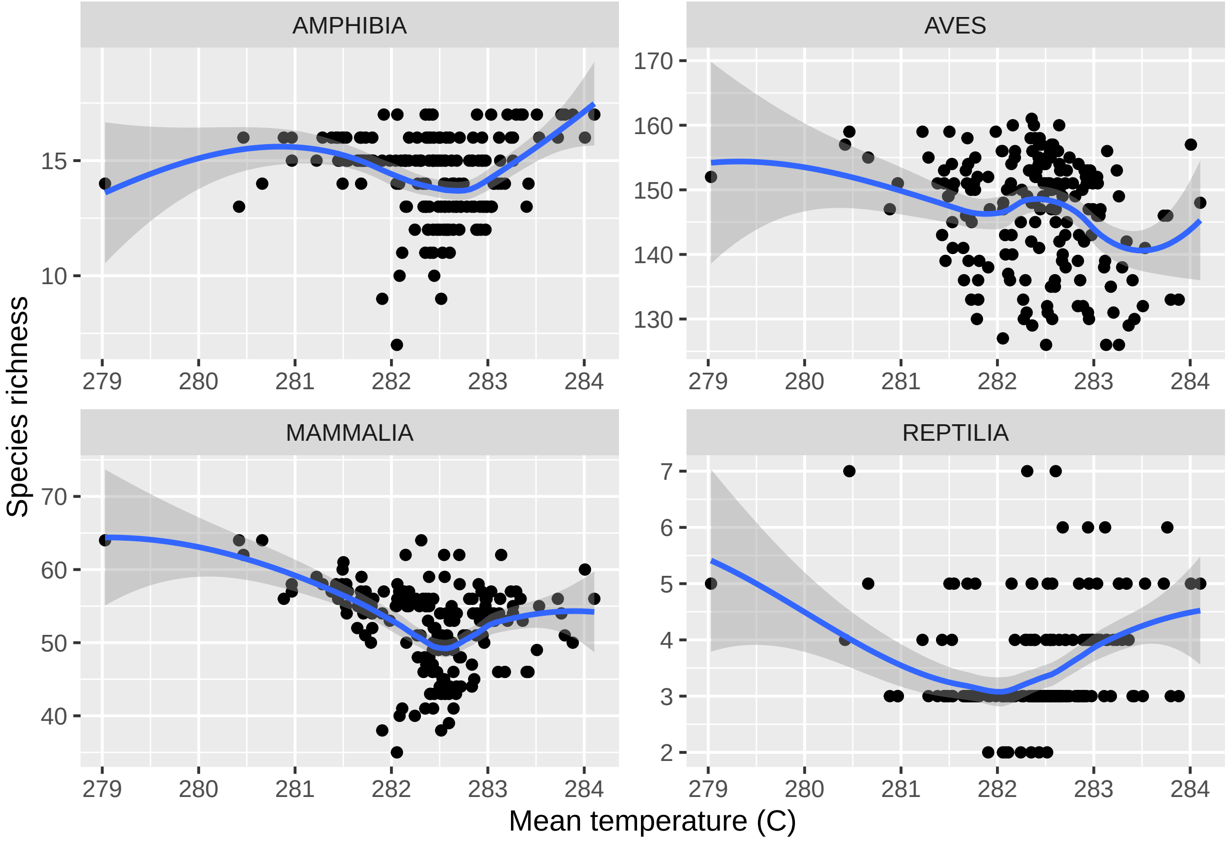
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Plot richness against temperature per class

```
env_sp %>% group_by(longitude,latitude, class_name) %>%
  summarise(sr=sum(occurrence), tmean=mean(tmean)) %>%
  ggplot(aes(x=tmean, y=sr)) + geom_point() +
  labs(x="Mean temperature (C)", y="Species richness") +
  geom_smooth() + facet_wrap(. ~ class_name, scales="free")
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```


AMPHIBIA
AVES
MAMMALIA
REPTILIA
Species richness
Mean temperature (C)
15
10
170
160
150
140
130
70
60
50
40
7
6
5
4
3
2
279
280
281
282
283
284