Working example for generating multiple plots inside a map2 call

Ronald (Ryy) Glenn Thomas

2024-02-23

Table of contents

1 Introduction

Figure 1: purrr

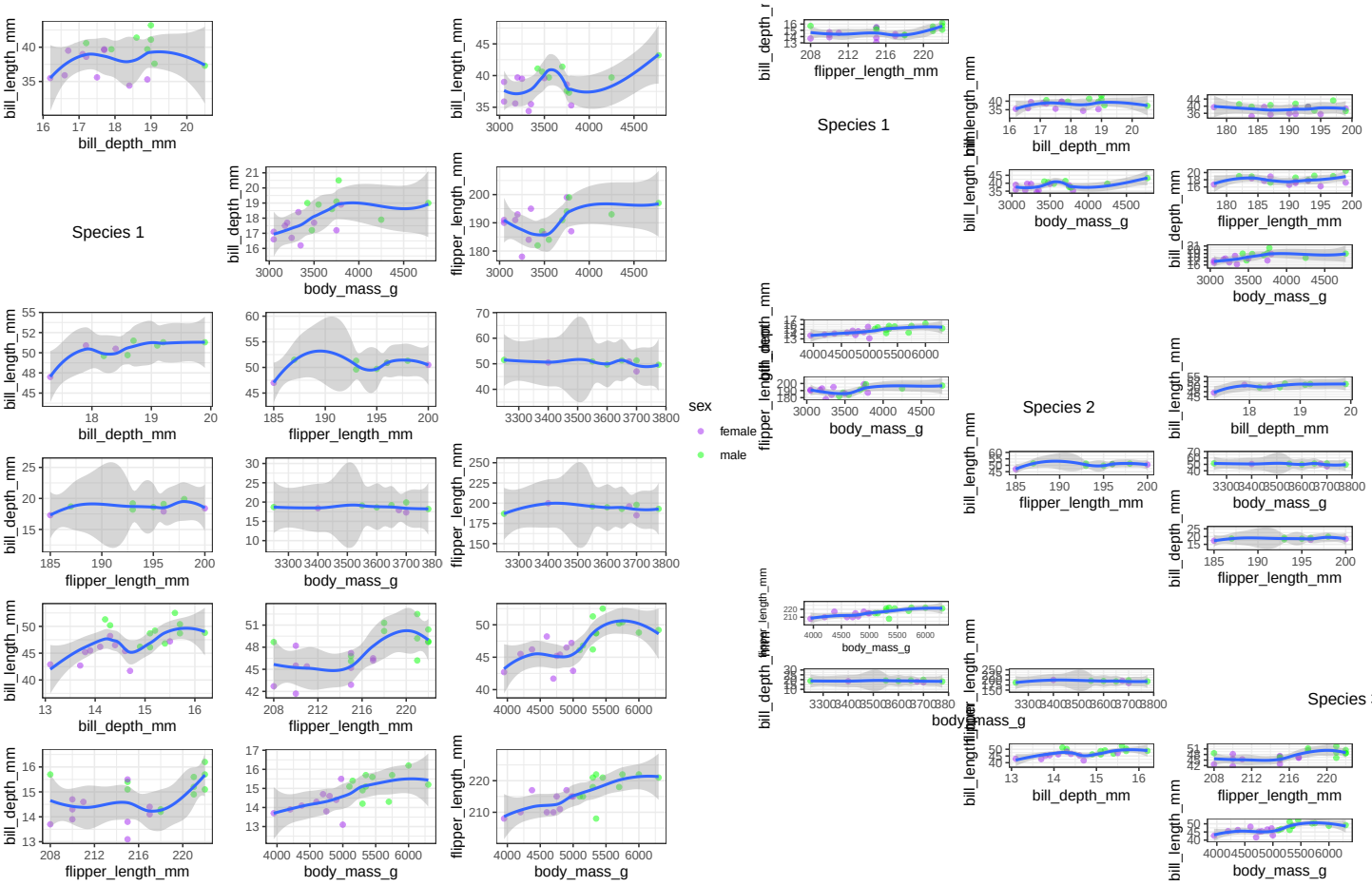
Consider the problem of running a data analysis requiring a separate analysis for each of `n` strata. For example consider an effort to model the relationship between `Bill length` and `Flipper length` across three different species of penguins.

We can work with the dataset `penguins` included in the package `palmerpenguins`

```
library(palmerpenguins)
```

One naive approach is to split the dataset and do three separate analyses:

The R package **purrr** provides a straightforward method to conduct the analyses with a single command. Assume the set of data tables are contained in a list of dataframes. Also assume the analysis is a simple visualization of a potential linear association between two features,



2 Plots for every variable and each species map inside map see ref 2 below

3 combine plots in a upper triangular grid with correlation coefs

4 Code

```
library(pacman)
p_load(grid, patchwork, rlang, purrr, palmerpenguins, tidyverse, knitr)

opts_chunk$set(
  warning = FALSE, message = FALSE, echo = FALSE, fig.width = 8,
  fig.height = 9, results = "asis", dev = "pdf"
)
df0 <- sample_n(penguins, 50) |> na.omit()
# nn = 50 ; df1 = sample_n(penguins, 50)
df1 <- split(df0, df0$species)
# df2 = penguins |> group_by(species)


ct <- names(df0)[3:6]
# mm = expand.grid(names(df1[3:6]), names(df1[3:6]))
nn <- t(combn(ct, 2))
colnames(nn) <- letters[1:2]
nn2 <- data.frame(nn) |> cbind(g = "sex")

zz.scatter <- function(data, formula, ...) {
  # function to take a dataframe, and a formula with potentially a '|' group
  #     option and return scatterplot matrix with optional R^2, loess smooth,
  #     or least squares line.
}

plt1 <- function(a, b, g, spc, df_split) {
  out_plot <- df_split |> ggplot(aes(x = .data[[a]], y = .data[[b]])) +
    geom_point(aes(color = .data[[g]]), alpha = .5) +
    geom_smooth() +
```

```
    scale_color_manual(values = c("purple", "green", "red")) +
    theme_bw()
  assign(paste0(spc, "_", a, "_", b), value = out_plot, envir = .GlobalEnv)
  return(out_plot)
}


temp <- df1 |> map2(names(df1), function(df_split, spc) {
  nn2 |> pmap(function(a, b, g) {
    plt1(b, a, g, spc, df_split)
  })
})

temp2 = list_flatten(temp)
X <- grid::textGrob("Species 1")
temp2[[2]] = plot_spacer()
temp2[[4]] = X
wrap_plots(temp2, ncol=3, nrow=6) +
  plot_layout(
    guides = "collect",
    axis_titles = "collect"
  )
A <- temp[[1]][[1]]
p2 <- temp[[1]][[2]]
p3 <- temp[[1]][[3]]
p4 <- temp[[1]][[4]]
p5 <- temp[[1]][[5]]
p6 <- temp[[1]][[6]]
p7 <- temp[[2]][[1]]
p8 <- temp[[2]][[2]]
p9 <- temp[[2]][[3]]
p10 <- temp[[2]][[4]]
p11 <- temp[[2]][[5]]
p12 <- temp[[2]][[6]]
p13 <- temp[[3]][[1]]
p14 <- temp[[3]][[2]]
p15 <- temp[[3]][[3]]
p16 <- temp[[3]][[4]]
p17 <- temp[[3]][[5]]
p18 <- temp[[3]][[6]]
```

```r
# names(temp)

layout <- "
X##
ABC
#DE
##F
Y##
GHI
#JK
##L
Z##
MNO
#PQ
##R
"
X <- grid::textGrob("Species 1")
t2 <- grid::textGrob("Species 2")
t3 <- grid::textGrob("Species 3")

out <- wrap_plots(
  X , A, B = p2, C = p3, D = p4, E = p5, F = p6, Y = t2,
  G = p7, H = p8, I = p9, J = p10, K = p11, L = p12, Z = t3,
  M = p13, N = p14, O = p15, P = p16, Q = p17, R = p18,
  design = layout
) +
  plot_layout(
    guides = "collect",
    axis_titles = "collect"
  ) +
  theme(
    legend.position = "bottom",
    legend.direction = "horizontal",
    text = element_text(size = 8)
  )

out
```

5 References

principal components analysis

Automating exploratory plots with ggplot2 and purrr