ggplot2 session outline

- We go through plots built with points and lines, from a very basic plot, to plots of increasing complexity.
- I will give examples of useful functions within ggplot2 and go through the code.
- We will go through what to think about when it comes to the size of the figure, and how to export/write figures.
- After the lecture there are class exercises.

The penguins data set

str(penguins)

tibble [333 × 8] (S3: tbl_df/tbl/data.frame)

```
$ species:           Factor w/ 3 levels "Adelie","Chinstrap",..: 1 1 1 1 1 1 1 1 1 1 ...
$ island:            Factor w/ 3 levels "Biscoe","Dream",..: 3 3 3 3 3 3 3 3 3 3 ...
$ bill_length_mm:    num [1:333] 39.1 39.5 40.3 36.7 39.3 38.9 39.2 41.1 38.6 34.6 ...
$ bill_depth_mm:     num [1:333] 18.7 17.4 18 19.3 20.6 17.8 19.6 17.6 21.2 21.1 ...
$ flipper_length_mm: int [1:333] 181 186 195 193 190 181 195 182 191 198 ...
$ body_mass_g:       int [1:333] 3750 3800 3250 3450 3650 3625 4675 3200 3800 4400 ...
$ sex:               Factor w/ 2 levels "female","male": 2 1 1 1 2 1 2 1 2 2 ...
$ year:              int [1:333] 2007 2007 2007 2007 2007 2007 2007 2007 2007 2007 ...
```

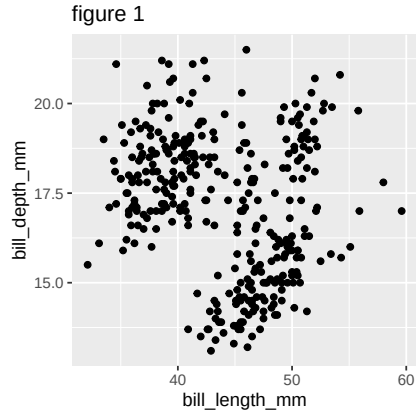
A default plot

- function `ggplot(data, aes(x=, y=))`
- the $+$ sign is used to add components to plot
- add layer to plot data `geom_point()`

data is either `data frame` or `tibble`

```
ggplot(data = penguins,
aes(x=bill_length_mm, y=bill_depth_mm))+
geom_point()
```

figure 1



Adding more than one variable to `aes()`, `geom_points()`

Which ones to use depends on your data

points continuous:

- color
- fill
- size

points discrete:

- color
- fill
- shape

Code for figure 2

```
ggplot(data = penguins, aes(x=bill_length_mm,
y=bill_depth_mm,
color=species,
shape=sex,
size=body_mass_g))+
labs(title = "figure 2")+
geom_point()
```

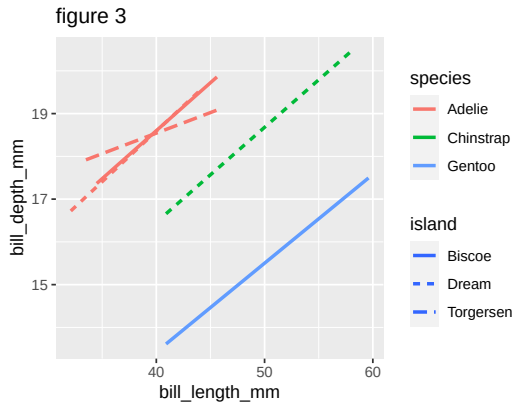
Adding more than one variable to `aes()`, `geom_line()` and `geom_smooth()`

lines numeric and discrete variables:

- color
- linetype
- group -specifies how groups (lines) are added to plot, for example: group=interaction(species, island)

  i.e. the unique combinations of species and island form the groups

figure 3



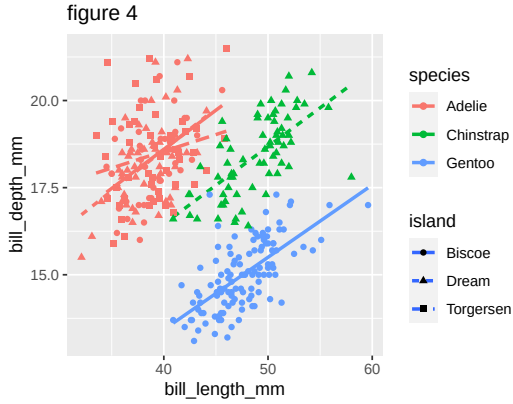
Code for figure 3

```
ggplot(data = penguins, aes(x=bill_length_mm,
y=bill_depth_mm,
color=species,
linetype=island))+
labs(title = "figure 3")+
geom_smooth(method = 'lm', se=F)
```

Adding more than one layer

- you can add points and lines to the same plot
- from the same data frame or from different ones
- when adding a new data frame, supply the data and mapping=aes() arguments in the new layer

figure 4



Code for figure 4

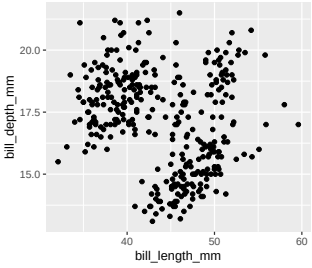
```
ggplot(data = penguins, aes(x=bill_length_mm,
y=bill_depth_mm,
color=species,
linetype=island))+
labs(title = "figure 4")+
geom_smooth(method = 'lm', se=F)+
geom_point(penguins, mapping=aes(shape=island))
```

Theme - changing from the default

The default plot have

- gray background
- white grid
- no panel borders
- default text size

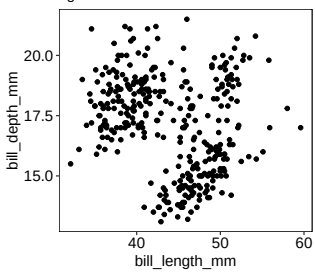
figure 5



Change to

- white background
- no grid
- black panel borders
- fitting text size

figure 6



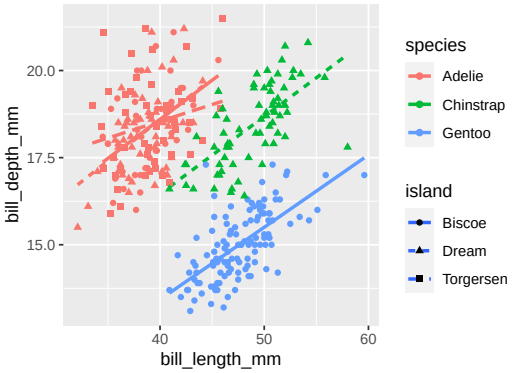
Code for figure 6

```
ggplot(data = penguins, aes(x=bill_length_mm,y=bill_depth_mm))+
geom_point()+
labs(title = "figure 6")+
theme(axis.text.x = element_text(colour="black",size=13),
      axis.text.y = element_text(colour="black",size=13),
      axis.title.x = element_text(colour="black",size=13),
      axis.title.y = element_text(colour="black",size=13),
      panel.grid = element_blank(),
      panel.background = element_rect(fill = "white", colour = "black"))
```

Customize colors, shapes, and lines

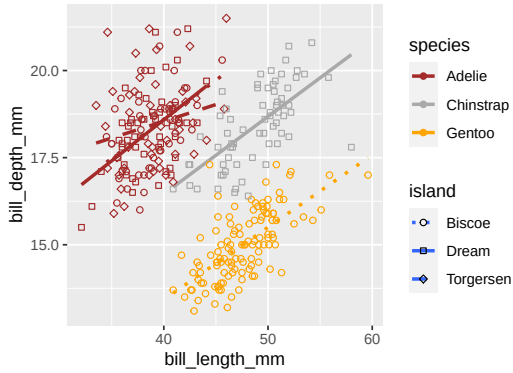
Default

figure 7



Custom

figure 8



Customise the aes arguments by using the scale_*_manual functions

* color, shape, linetype etc..

Code for figure 8

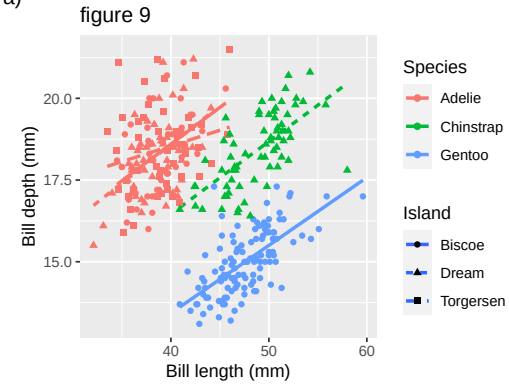
```
ggplot(data = penguins, aes(x=bill_length_mm,
y=bill_depth_mm,
color=species,
linetype=island))+
geom_smooth(method = 'lm', se=F)+
labs(title = "figure 8")+
scale_colour_manual(values = c("brown","darkgray","orange"))+
scale_linetype_manual(values = c("dotted","solid","dashed"))+
scale_shape_manual(values = c(21,22,23))+
geom_point(penguins, mapping=aes(shape=island))
```

Labels can be changed and added by the labs function

arguments in labs can:

- produce a tag "a)", that is useful for multiple plots
- change the x and y axis titles
- change the headings in the legend
- produce a plot title

a)



Code for figure 9

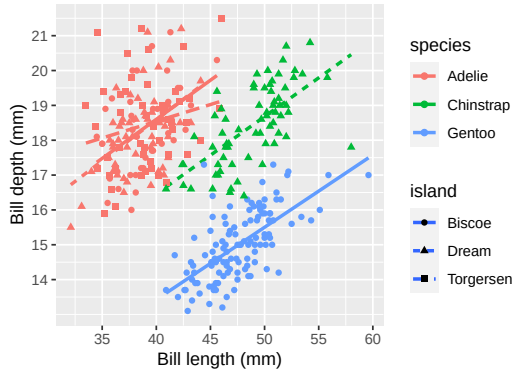
```
ggplot(data = penguins, aes(x=bill_length_mm, y=bill_depth_mm,
color=species, linetype=island))+
geom_smooth(method = 'lm', se=F)+
geom_point(penguins, mapping=aes(shape=island))+
labs(tag="a)",
     title = "figure 9",
     y="Bill depth (mm)",
     x="Bill length (mm)",
     color="Species",
     linetype="Island",
     shape="Island")
```

Breaks and limits of the x and y axis

breaks and limits are changed by the same function

- `scale_x_continuous(breaks=c(), limits=c() )`
- `scale_y_continuous(breaks=c(), limits=c() )`

figure 10

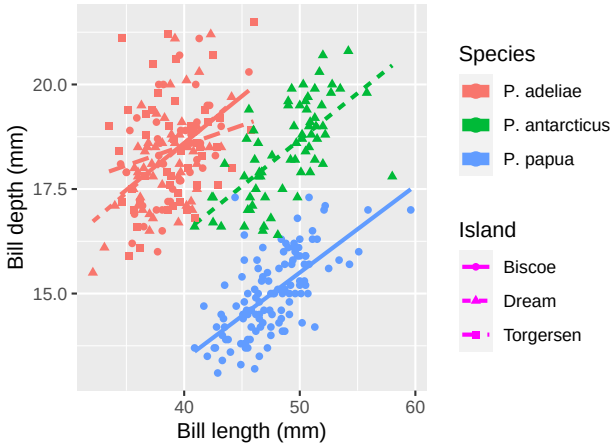


Legend

There are several functions for legends that can help you

- change the legend heading `-labs()`
- change the name of levels within legend `-scale_*_manual()`
- customize the symbols in the legend `-guides( guide_legend(overrride.aes()) )`
- turn off legends completely `-guides()`
- change position `-theme()`
- change from vertical to horizontal box `-theme()`

figure 11



Code for figure 11

```
ggplot(data = penguins,
aes(x=bill_length_mm,y=bill_depth_mm,color=species,linetype=island,shape=island))+
geom_smooth(method = 'lm', se=F)+
geom_point()+
labs(title = "figure 10",
      y="Bill depth (mm)",
      x="Bill length (mm)",
      color="Species",
       shape="Island",
       linetype="Island")+
scale_color_manual(values = default_colors, labels=c("P. adeliae","P. antarcticus",
       P. papua"))+
guides(linetype=guide_legend(override.aes = list(color="magenta")),
       color=guide_legend(override.aes = list(size=3)))
```

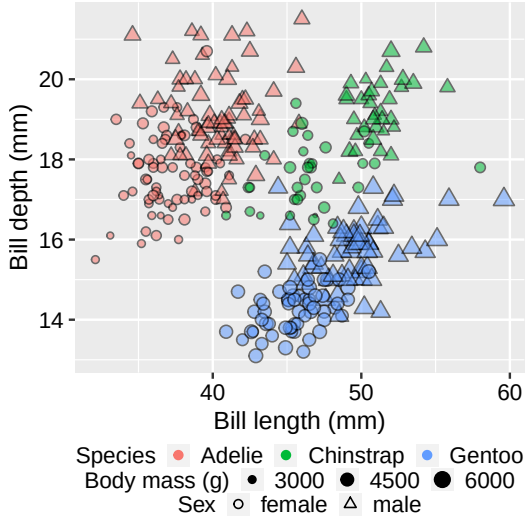
Dimensions of the exported plot in relation to size within plot

- one column figure
  -max width of 3.5 inches or 8.9 cm
- if written from R with default ggplot settings, the figure becomes disproportionate
- size of points are too large for the panel
- the legend take up much of the panel
- decimal numbers take up unnecessary space

Dimension of the exported plot in relation to size within plot

- placing the legend on the bottom frees up panel space
- clear white space from the legend box
- changing the size range of points make them appear more clearly
- overlapping points become more clearly visible if they are semi-transparent and have borders, that is, change pch and use fill and alpha in aes()

figure 13

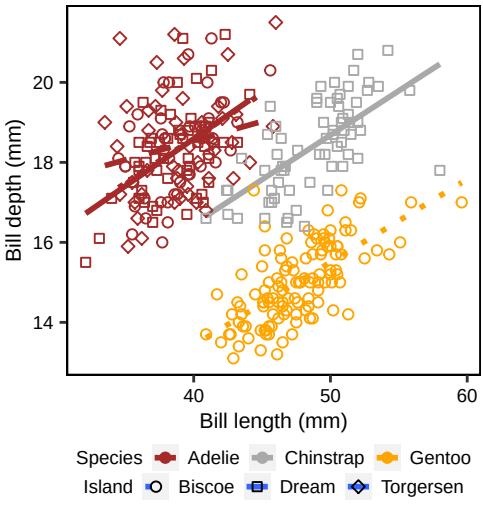


Code for figure 13

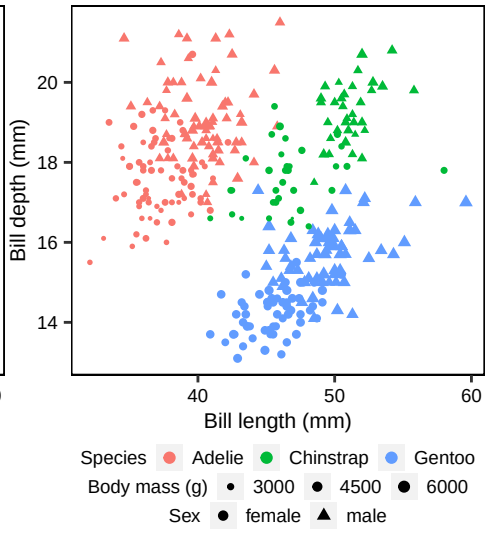
```
ggplot(data = penguins, aes(x=bill_length_mm,y=bill_depth_mm,fill=species,shape=sex,size=body_mass_g,alpha=0.5))+
geom_point(color="black")+
scale_shape_manual(values=c(21,24))+
scale_size_continuous(range = c(0.5, 3), breaks = c(3000,4500,6000))+
scale_y_continuous(breaks = c(14,16,18,20))+
labs(x="Bill length (mm)", y= "Bill depth (mm)", fill="Species", shape="Sex", size= "Body mass (g)", title = "figure 12")+
guides(alpha=F, fill=guide_legend(override.aes = list(color=default_colors)))+
theme(axis.text.x = element_text(colour="black",size=10),
      axis.text.y = element_text(colour="black",size=10),
      axis.title.x = element_text(colour="black",size=11),
      axis.title.y = element_text(colour="black",size=11),
      legend.position = "bottom",
      legend.text = element_text(size=10),
      legend.title = element_text(size = 10),
      legend.key.size = unit(10,"point"),
      legend.box = "vertical",
      legend.spacing.y = unit(-.25,"cm"),
      legend.margin=margin(0,5,8,5),
      legend.box.margin=margin(-10,10,-3,10),
      legend.box.spacing = unit(0.5,"cm"),

      plot.margin = margin(2,2,0,0))
```

Plots side by side with identical panel size

a) figure 14a



b) figure 14b

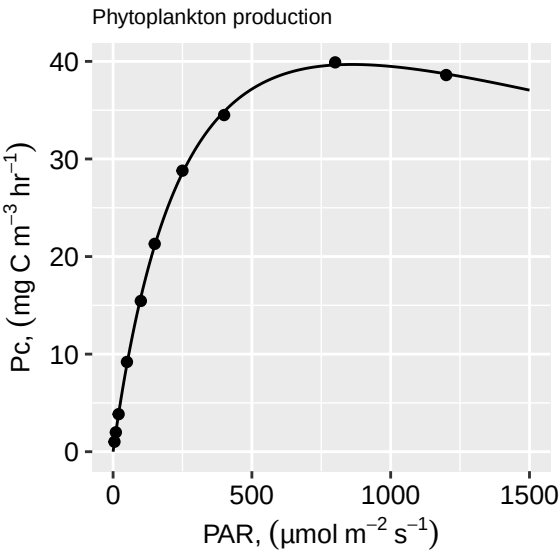


Write plots side by side

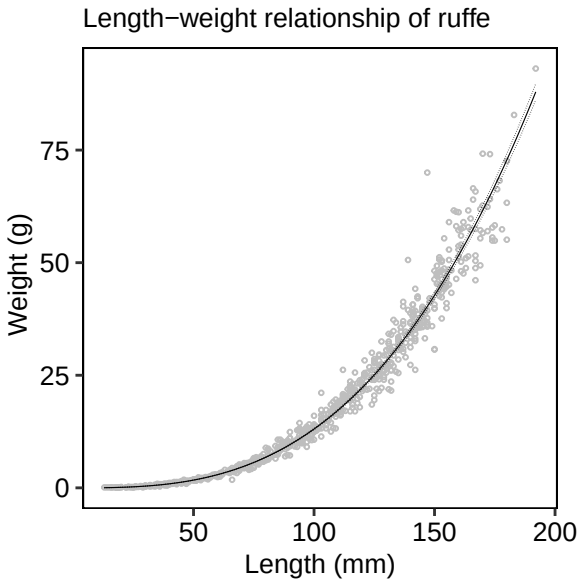
```
setwd(" ")

pdf("fig14.pdf", width = 5, height = 3,paper="special")

ggarrange(fig14a,fig14b, ncol=2, nrow=1 ,labels = c("a)","b)"),
label.args = list(gp = gpar(font = 1, cex = .8)))

dev.off()
```

You can also export your plot in other formats as well, like postscript() (for .eps) or png()

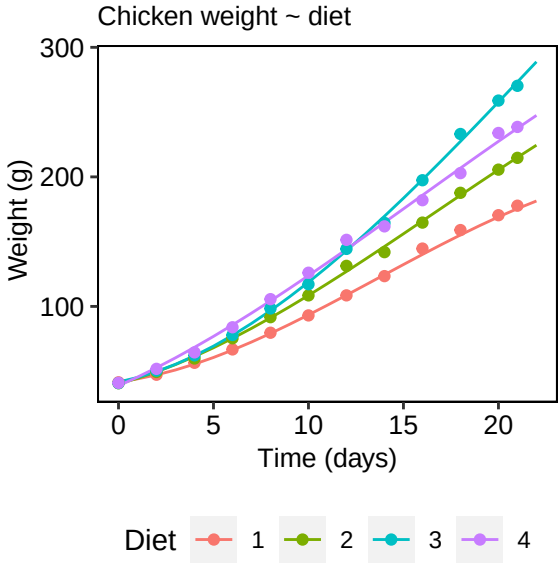
Create you own figures, exercise 1



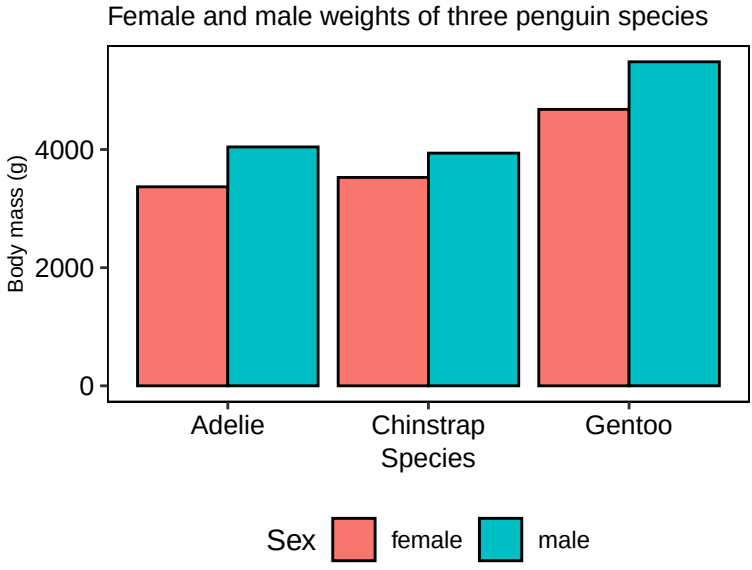
Create you own figures, exercise 2



Create you own figures, exercise 3



Create you own figures, exercise 4



Create you own figures, exercise 5