Lesson XX: Gemo Review (Decide where to put in 2023)

Doug Joubert

2022-03-02

# Introduction

## Packages Used for this Demo

library(gapminder)  
library(tidyverse)

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v ggplot2 3.3.5 v purrr 0.3.4  
## v tibble 3.1.6 v dplyr 1.0.8  
## v tidyr 1.2.0 v stringr 1.4.0  
## v readr 2.1.2 v forcats 0.5.1

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(socviz)  
library(ggrepel)  
library(RColorBrewer)  
library(nlme)

##   
## Attaching package: 'nlme'

## The following object is masked from 'package:dplyr':  
##   
## collapse

## Data Used in this Demo

* interviews plotting
* Oxboys

## Read in our Data

interviews\_plotting <- read\_csv("../raw\_data/interviews\_plotting.csv")

## Rows: 131 Columns: 45  
## -- Column specification --------------------------------------------------------  
## Delimiter: ","  
## chr (5): village, respondent\_wall\_type, memb\_assoc, affect\_conflicts, inst...  
## dbl (8): key\_ID, no\_membrs, years\_liv, rooms, liv\_count, no\_meals, number\_...  
## lgl (31): bicycle, television, solar\_panel, table, cow\_cart, radio, cow\_plo...  
## dttm (1): interview\_date  
##   
## i Use `spec()` to retrieve the full column specification for this data.  
## i Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

# ggplot and Layers

The general structure of the code that is used to plot figures is as follows:

ggplot(...) +             # Initialization  
  geom\_function(...) +    # Main plot(s)  
  facet\_function(...) +   # Facets (optional)  
  labs(...) +             # Legend (optional)  
  scale\_function(...) +   # Scales (optional)  
  theme\_function(...)     # Theme (optional)

What to consider, when working with these layers:

* The ggplot() layer is mandatory.
* When the data argument is specified inside the ggplot() function, it is used as default in the following layers that compose the plot command, unless otherwise specified.
* In order for features of a data frame to be used in a plot, they need to be specified inside the aes() function

# Geoms in ggplot

## Individual geoms

The geoms listed below are the fundamental building blocks of ggplot2. They are useful in their own right, but are also used to construct more complex geoms. Most of these geoms are associated with a named plot: when that geom is used by itself in a plot, that plot has a special name.

Each of these geoms is two dimensional and requires both x and y aesthetics. All of them understand colour (or color) and size aesthetics, and the filled geoms (bar, tile and polygon) also understand fill.

* [geom\_area()](https://ggplot2.tidyverse.org/reference/geom_ribbon.html) draws an **area plot**, which is a line plot filled to the y-axis (filled lines). Multiple groups will be stacked on top of each other.
* [geom\_bar(stat = "identity")](https://ggplot2-book.org/statistical-summaries.html#distributions) makes a **bar chart**. We need stat = "identity" because the default stat automatically counts values (so is essentially a 1d geom, see Section [5.4](https://ggplot2-book.org/statistical-summaries.html#distributions). The identity stat leaves the data unchanged. Multiple bars in the same location will be stacked on top of one another.
* [geom\_line()](https://ggplot2.tidyverse.org/reference/geom_path.html) makes a **line plot**. The group aesthetic determines which observations are connected; see Chapter [4](https://ggplot2-book.org/collective-geoms.html#collective-geoms) for more detail. [geom\_line()](https://ggplot2.tidyverse.org/reference/geom_path.html) connects points from left to right; [geom\_path()](https://ggplot2.tidyverse.org/reference/geom_path.html) is similar but connects points in the order they appear in the data. Both [geom\_line()](https://ggplot2.tidyverse.org/reference/geom_path.html) and [geom\_path()](https://ggplot2.tidyverse.org/reference/geom_path.html) also understand the aesthetic linetype, which maps a categorical variable to solid, dotted and dashed lines.
* [geom\_point()](https://ggplot2.tidyverse.org/reference/geom_point.html) produces a **scatterplot**. [geom\_point()](https://ggplot2.tidyverse.org/reference/geom_point.html) also understands the shape aesthetic.
* [geom\_polygon()](https://ggplot2.tidyverse.org/reference/geom_polygon.html) draws polygons, which are filled paths. Each vertex of the polygon requires a separate row in the data. It is often useful to merge a data frame of polygon coordinates with the data just prior to plotting. Section [6](https://ggplot2-book.org/maps.html#maps) illustrates this concept in more detail for map data.
* [geom\_rect()](https://ggplot2.tidyverse.org/reference/geom_tile.html), [geom\_tile()](https://ggplot2.tidyverse.org/reference/geom_tile.html) and [geom\_raster()](https://ggplot2.tidyverse.org/reference/geom_tile.html) draw rectangles. [geom\_rect()](https://ggplot2.tidyverse.org/reference/geom_tile.html) is parameterised by the four corners of the rectangle, xmin, ymin, xmax and ymax. [geom\_tile()](https://ggplot2.tidyverse.org/reference/geom_tile.html) is exactly the same, but parameterised by the center of the rect and its size, x, y, width and height. [geom\_raster()](https://ggplot2.tidyverse.org/reference/geom_tile.html) is a fast special case of [geom\_tile()](https://ggplot2.tidyverse.org/reference/geom_tile.html) used when all the tiles are the same size. .

## Collective Geoms in ggplot

An **individual** geom draws a distinct graphical object for each observation (row). For example, the point geom draws one point per row. A **collective** geom displays multiple observations with one geometric object. This may be a result of a statistical summary, like a boxplot, or may be fundamental to the display of the geom, like a polygon. How do we control the assignment of observations to graphical elements? This is the job of the group aesthetic.

By default, the group aesthetic is mapped to the interaction of all discrete variables in the plot. This often partitions the data correctly. However, when the mapping is not automatic, or when no discrete variable is used in a plot, you’ll need to explicitly define the grouping structure to a variable that has a different value for each group.

In the following examples, we will use a simple longitudinal dataset, [Oxboys](https://rdrr.io/cran/nlme/man/Oxboys.html) which is included in the nlmepackage. It looks at the height of a selection of boys from Oxford, England versus a standardized age. It records the heights (height) and centered ages (age) of 26 boys (Subject), measured on nine occasions (Occasion). Subject and Occassion are stored as ordered factors.

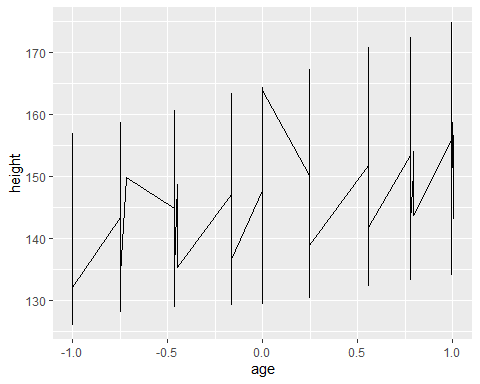
data(Oxboys, package = "nlme")  
head(Oxboys)

## Grouped Data: height ~ age | Subject  
## Subject age height Occasion  
## 1 1 -1.0000 140.5 1  
## 2 1 -0.7479 143.4 2  
## 3 1 -0.4630 144.8 3  
## 4 1 -0.1643 147.1 4  
## 5 1 -0.0027 147.7 5  
## 6 1 0.2466 150.2 6

### Multiple groups, one aesthetic

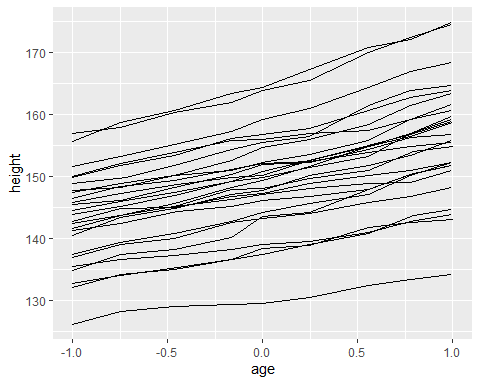
In many situations, you want to separate your data into groups, but render them in the same way. In other words, you want to be able to distinguish individual subjects, but not identify them. This is common in longitudinal studies with many subjects, where the plots are often descriptively called spaghetti plots. For example, the following plot shows the growth trajectory for each boy (each Subject). Without grouping, ggplot will make its best judgement, when displaying the graph

Oxboys %>%  
 ggplot(mapping = aes(x = age, y = height)) +   
 geom\_line()



This is certainly not what we want, and we get a graph that has a saw tooth appearance:

Oxboys %>%   
ggplot(mapping = aes(x = age, y = height, group = Subject)) +   
 geom\_line()

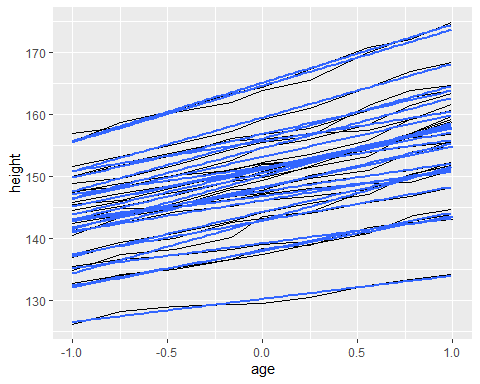


## Different groups on different layers

Sometimes we want to plot summaries that use different levels of aggregation: one layer might display individuals, while another displays an overall summary. Building on the previous example, suppose we want to add a single smooth line, showing the overall trend for *all* boys. If we use the same grouping in both layers, we get one smooth line per boy:

Oxboys %>%   
ggplot(mapping = aes(x = age, y = height, group = Subject)) +   
 geom\_line() +   
 geom\_smooth(method = "lm", se = FALSE)

## `geom\_smooth()` using formula 'y ~ x'

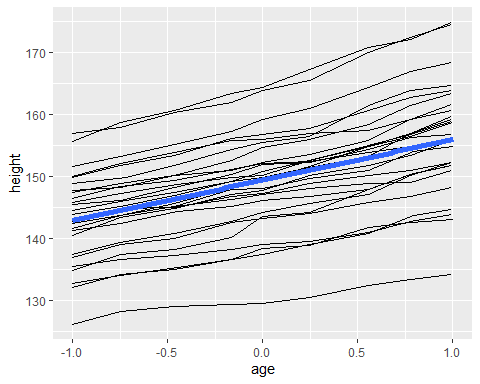


This is not what we wanted; we have added a smoothed line for each boy. Grouping controls both the display of the geoms, and the operation of the stats: one statistical transformation is run for each group.

Instead of setting the grouping aesthetic in [ggplot()](https://ggplot2.tidyverse.org/reference/ggplot.html), where it will apply to all layers, we set it in [geom\_line()](https://ggplot2.tidyverse.org/reference/geom_path.html) so it applies only to the lines. There are no discrete variables in the plot so the default grouping variable will be a constant and we get one smooth.

ggplot(Oxboys, aes(age, height)) +   
 geom\_line(aes(group = Subject)) +   
 geom\_smooth(method = "lm", size = 2, se = FALSE)

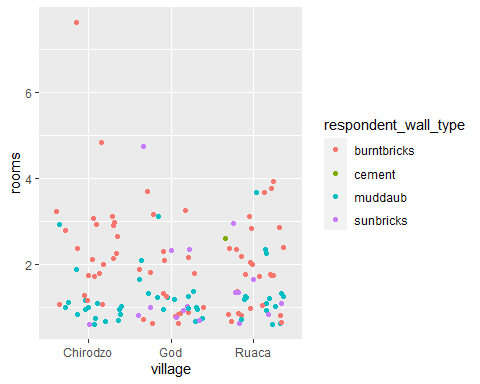
## `geom\_smooth()` using formula 'y ~ x'



# Examples from Class

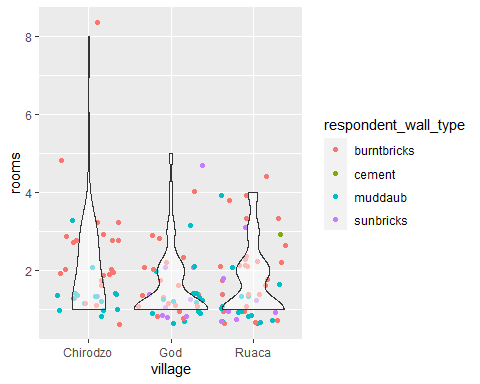
### Example 1: col = respondent\_wall\_type in ggplot statement (Doug Version)

interviews\_plotting %>%   
 ggplot(mapping = aes(x = village, y = rooms)) +  
 geom\_jitter(mapping = aes(col = respondent\_wall\_type))



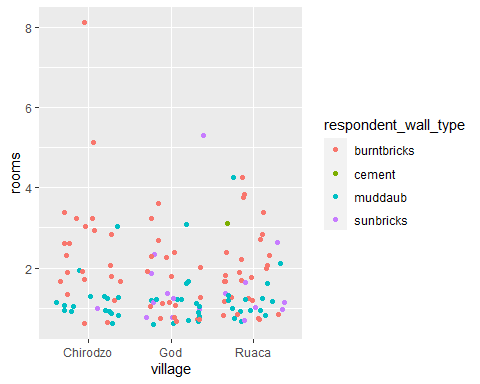
### Example 1: Implications when you add another layer (Doug version)

interviews\_plotting %>%   
 ggplot(mapping = aes(x = village, y = rooms)) +  
 geom\_jitter(mapping = aes(col = respondent\_wall\_type)) +  
 geom\_violin(alpha = 0.5)



## Example 2: col = respondent\_wall\_type in ggplot statement (Other Version)

interviews\_plotting %>%   
 ggplot(mapping = aes(x = village, y = rooms, col = respondent\_wall\_type)) +  
 geom\_jitter()



### Example 2: Implications when you add another layer (Other Version)

interviews\_plotting %>%   
 ggplot(mapping = aes(x = village, y = rooms, col = respondent\_wall\_type)) +  
 geom\_jitter() +  
 geom\_violin(alpha = 0.5)

## Warning: Groups with fewer than two data points have been dropped.  
## Groups with fewer than two data points have been dropped.

