

Problem Set #1

INSERT YOUR NAME HERE

Invalid Date

Table of contents

Overview:	2
Question 1: <code>geom_point()</code>	2
Question 2: <code>geom_smooth()</code>	3
Question 3: <code>geom_vline()</code> and <code>geom_hline()</code> :	4
Question 4: <code>geom_bar()</code>	4
Question 5: <code>geom_boxplot()</code> and <code>geom_density()</code>	5
Question 6: Aesthetics	6
Render to html and submit problem set	7

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ----- tidyverse 2.0.0 --
```

```
v dplyr      1.1.4      v readr      2.1.5
```

```
v forcats    1.0.0      v stringr    1.5.1
```

```
v ggplot2    3.5.0      v tibble     3.2.1
```

```
v lubridate  1.9.3      v tidyr      1.3.1
```

```
v purrr      1.0.2
```

```
-- Conflicts ----- tidyverse_conflicts() --
```

```
x dplyr::filter() masks stats::filter()
```

```
x dplyr::lag()     masks stats::lag()
```

```
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become
```

Overview:

In this problem set, you will be using the **ggplot2** package (part of tidyverse) to practice the basics of plotting. Unlike later homeworks, this is just a basic set of exercises, so you will not be asked use your own data (although you're welcome to if you'd really like to).

For demonstration, we'll use the **starwars** dataset from the **dplyr** package, which you will have access to after loading the **tidyverse** package.

```
data(starwars)
head(starwars)
```

```
# A tibble: 6 x 14
  name      height  mass hair_color skin_color eye_color birth_year sex  gender
<chr>    <int> <dbl> <chr>    <chr>    <chr>    <dbl> <chr> <chr>
1 Luke Sky~   172    77 blond    fair     blue      19  male masculi~
2 C-3PO      167    75 <NA>     gold     yellow    112 none masculi~
3 R2-D2       96    32 <NA>     white, bl~ red       33 none masculi~
4 Darth Va~   202   136 none     white    yellow    41.9 male masculi~
5 Leia Org~   150    49 brown    light    brown     19  fema~ feminin~
6 Owen Lars   178   120 brown, gr~ light    blue      52  male masculi~
# i 5 more variables: homeworld <chr>, species <chr>, films <list>,
#   vehicles <list>, starships <list>
```

Question 1: geom_point()

1. Plot the relationship between mass and height using `geom_point()`.

```
# your code here
```

2. What an outlier! Use the `arrange()` function to sort the data by mass (descending) to figure out what it is

```
# your code here
```

3. Now, plot the relationship between mass and height again, removing that outlier (hint: use `filter()`).

```
# your code here
```

4. It's possible that different species in the starwars universe have different weight-height patterns. Let's test that by setting `color = species`:

```
# your code here
```

5. Oops – that's a lot of species, let's reduce that to humans, Droids, and Wookiees and collapse the others to "Other" (hint create a new variable with `mutate`; consider using `ifelse()`, `if_else()` or `case_when()`). Then replot. Once you're done, assign that plot to object `p1`. Remember that `ggplot` is a layered grammar of graphics, so assigning this plot to an object will let us layer additional things on top of this base plot.

```
# your code here
```

Question 2: `geom_smooth()`

Now that we've got our scatterplot, let's layer a line of best fit on top. We're going to test out different fits here. You can get a sense of this by typing `?geom_smooth` in your console.

1. First, let's test a linear fit between height and weight using `geom_smooth()`. To do this, you'll set `method = "lm"`:

```
# your code here
```

2. Hmm, that maybe isn't super linear. Let's test out a non-linear fit. To get a better sense of the general pattern, let's start with a loess line (hint: set `method = "loess"`):

```
# your code here
```

3. That's not totally clear – what about quadratic? We can change the formula that links `x` and `y` via the formula argument (`formula = y ~ x + I(x^2)`)

```
# your code here
```

4. Let's try one more. Set the method to "gam":

```
# your code here
```

5. Choose one of these and save it as object `p2`.

Question 3: `geom_vline()` and `geom_hline()`:

Now, let's practice adding vertical and horizontal lines. Let's add a line at the mean of both height (vertical) and weight (horizontal) using `geom_vline()` and `geom_hline()`, respectively.

1. Add a vertical line at the mean of height. Make it dashed and increase the thickness. Assign this to p3.

```
# your code here
```

2. Add a horizontal line at the mean of weight. Make it dashed and increase the thickness. Assign this to p4.

```
# your code here
```

Question 4: `geom_bar()`

But maybe we do actually just care about the means, so let's plot the mean and SDs of height and weight across species. Here's code to get the descriptives to help you get started:

```
starwars2 <- starwars %>%
  mutate(species_cat = ifelse(species %in% c("Human", "Droid", "Wookiee"), species, "Other"))
  filter(mass < 200) %>%
  select(name, height, mass, species_cat) %>%
  pivot_longer(
    cols = c(height, mass)
    , names_to = "measure"
    , values_to = "value"
  )

starwars_desc <- starwars2 %>%
  group_by(species_cat, measure) %>%
  summarize_at(vars(value), lst(mean, sd), na.rm = T) %>%
  ungroup()
starwars_desc
```

```
# A tibble: 8 x 4
  species_cat measure  mean    sd
  <chr>         <chr>   <dbl> <dbl>
```

1	Droid	height	140	52.0
2	Droid	mass	69.8	51.0
3	Human	height	180.	11.5
4	Human	mass	81.3	19.3
5	Other	height	171.	40.4
6	Other	mass	69.7	29.5
7	Wookiee	height	231	4.24
8	Wookiee	mass	124	17.0

1. Plot the mean of both height and mass using `geom_col()` or `geom_bar()`, splitting the two measures (height & weight using `facet_grid()`), filling by species and setting `color = "black"` to add an outline:

```
# your code here
```

2. Now add the SD using `geom_errorbar()`. Your key new arguments are `ymin = mean - sd` and `ymax = mean + sd` (hint: set the width to a smaller value to improve the aesthetic):

```
# your code here
```

3. Now let's re-add the raw data back in using `geom_jitter()` (jittering in the x direction only). Note the following hints:

- You will need to use a different data set. You can do this by using the `data` argument within `geom_jitter()` (`data = starwars2`)
- You want to jitter the x direction, not y, which you can do by setting `height = 0`
- Don't forget to change the color by setting `color = species_cat`

```
# your code here
```

4. Hmm, we can't really see the points. We'll do three things here. We'll change the `shape`, change fill for color, set `color = "black"`, and adjust the alpha (transparency):

```
# your code here
```

Question 5: `geom_boxplot()` and `geom_density()`

Lastly, let's do some quick practice with distributions of data using `geom_density()` and `geom_boxplot()`.

1. Make a boxplot of mass and height using `geom_boxplot()` and the `starwars2` dataset

- hint: `y = species_cat` and `x = value`
- Don't forget to use `facet_grid` again!
- set `fill = species_cat`
- remove the unnecessary legend using `theme(legend.position = "none")`

your code here

1. Make a histogram of mass and height using `geom_histogram()` and the `starwars2` dataset

- hint: `x = value`
- Don't forget to use `facet_grid` again; this time, you also need to add `species_cat` to it!
- set `fill = species_cat`
- set `color = "black"`
- remove the unnecessary legend using `theme(legend.position = "none")`

your code here

Question 6: Aesthetics

Choose any plot above that has some sort of color or fill mapping to improve it's aesthetic appearance.

1. **Axis labels:**

- Adjust the x and y labels using the `labs()` function.
- Modify their appearance using `theme(axis.text = element_text(face = "bold"), axis.title = element_text(face = "bold", size = rel(1.4)))`

2. **Plot title:**

- Add a plot title using the `labs()` function.
- Change the appearance of the title using `theme(plot.title = element_text())`

3. **Legend:**

- Redundant legend? Remove it
- Side legend? Move it to the bottom
- Weird title for the legend? Adjust it by updating the title for the relevant aesthetic in `labs()`

4. **Facets:**

- Weird facet range for one panel? Play around with setting the argument `scale` to `"free"`, `"free_x"`, and `"free_y"`.
- Change their appearance using `theme`. Try `theme(strip.background = element_rect(fill = "black"))` to set the background color. Then change the font color and appearance using `strip.text = element_text(color = "white", face = "bold")`

your code here

Render to html and submit problem set

Render to html by clicking the “Render” button near the top of your RStudio window (icon with blue arrow)

- Go to the Canvas -> Assignments -> Problem Set 1
- Submit both .qmd and .html files
- Use this naming convention “lastname_firstname_ps#” for your .qmd and html files (e.g. beck_emorie_ps1.qmd & beck_emorie_ps1.html)