Study Guide: Data Visualization with R

Afshine Amidi and Shervine Amidi

August 9, 2020

General structure

❒ **Overview** – The general structure of the code that is used to plot figures is as follows:

R

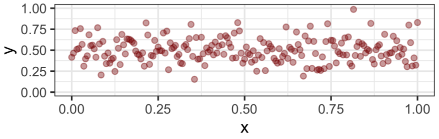
```
ggplot(...) +          # Initialization
  geom_...(...) +      # Main plot(s)
  facet_...(...) +     # Facets (optional)
  labs(...) +          # Legend (optional)
  scale_...(...) +     # Scales (optional)
  theme_...()          # Theme (optional)
```

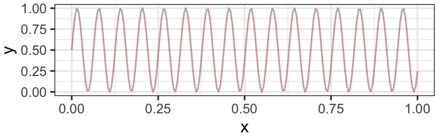
We note the following points:

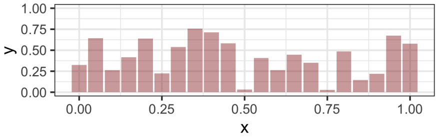
- The `ggplot()` layer is mandatory.
- When the `data` argument is specified inside the `ggplot()` function, it is used as default in the following layers that compose the plot command, unless otherwise specified.
- In order for features of a data frame to be used in a plot, they need to be specified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

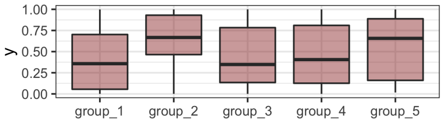
❒ **Basic plots** – The main basic plots are summarized in the table below:

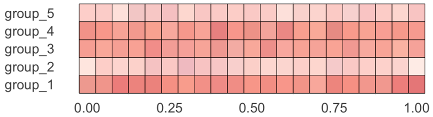
Type	Command	Illustration

Scatter plot	`geom_point(`
	`x, y, color, size,`
	`fill, alpha`
	`)`
		

Line plot	`geom_line(`
	`x, y, color, size,`
	`fill, alpha`
	`)`
		

Bar chart	`geom_bar(`
	`x, y, color, size,`
	`fill, alpha`
	`)`
		

Box plot	`geom_boxplot(`
	`x, y, color, size,`
	`fill, alpha`
	`)`
		

Heatmap	`geom_tile(`
	`x, y, color, size,`
	`fill, alpha`
	`)`
		

where the meaning of parameters are summarized in the table below:

Command	Description	Use case
`color`	Color of a line / point / border	`'red'`
`fill`	Color of an area	`'red'`
`size`	Size of a line / point	4
`shape`	Shape of a point	4
`linetype`	Shape of a line	`'dashed'`
`alpha`	Transparency, between 0 and 1	0.3

❒ **Maps** – It is possible to plot maps based on geometrical shapes. The following command gives the general structure of a map plot based on polygons:

R

```
ggplot() +
  geom_sf(data, aes(...)) +
  coord_sf(xlim, ylim) +
  annotation_north_arrow() +
  annotation_scale()
```

Action	Command
Draw boundaries	`geom_sf`
Add geographical directions	`annotation_north_arrow`
Add distance scale	`annotation_scale`

❒ **Animations** – Plotting animations can be made using the `gganimate` library. The following command gives the general structure of the code:

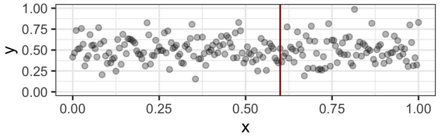
R

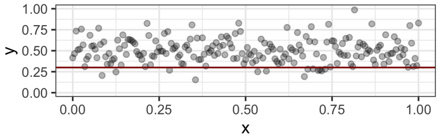
```
ggplot() + ... + transition_states(field, states_length)
animate(plot, duration, fps, width, height, units, res, renderer)
anim_save(filename)
```

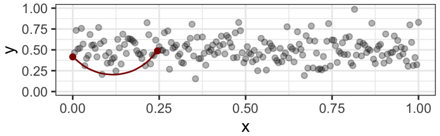
Advanced features

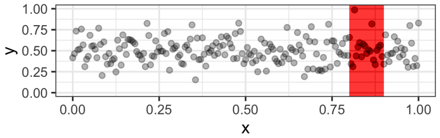
❒ **Additional elements** – We can add objects on the plot with the following commands:

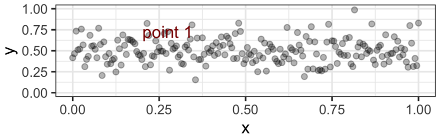
Type	Command	Illustration

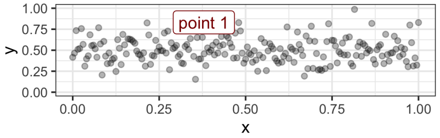
Line	`geom_vline(`
	`xintercept, linetype`
	`)`
		

	`geom_hline(`
	`yintercept, linetype`
	`)`
		

Curve	`geom_curve(`
	`x, y, xend, yend`
	`)`
		

Rectangle	`geom_rect(`
	`xmin, xmax, ymin, ymax`
	`)`
		

Text	`geom_text(`
	`x, y, label,`
	`hjust, vjust`
	`)`
		

	`geom_label_repel(`
	`x, y, label,`
	`nudge_x, nudge_y`
	`)`
		

Last touch

❒ **Legend** – The title of legends can be customized to the plot with the following command:

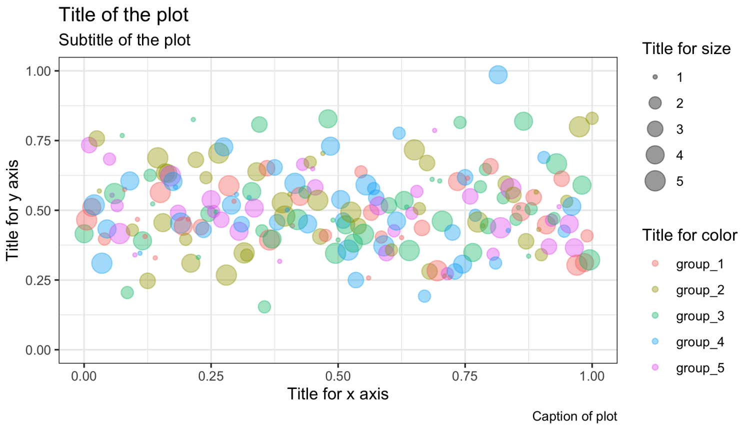
R

```
plot + labs(params)
```

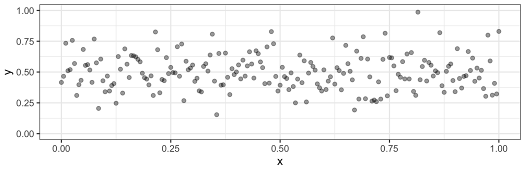
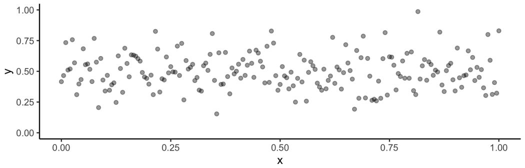
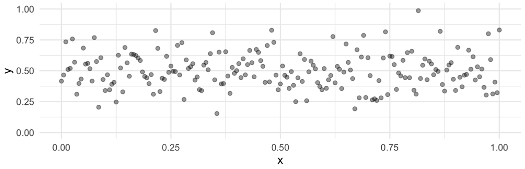
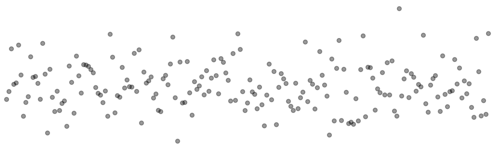
where the `params` are summarized below:

Element	Command
Title / subtitle of the plot	`title = 'text'` / `subtitle = 'text'`
Title of the x / y axis	`x = 'text'` / `y = 'text'`
Title of the size / color	`size = 'text'` / `color = 'text'`
Caption of the plot	`caption = 'text'`

This results in the following plot:



❒ **Theme** – The theme of a given plot can be set for all plots by adding the following command:

Type	Command	Illustration
Black and white	`theme_bw()`	
Classic	`theme_classic()`	
Minimal	`theme_minimal()`	
None	`theme_void()`	

In addition, `theme()` is able to adjust positions/fonts of elements of the legend.

❒ **Axes and scales** – Axes and scales can be changed with the following commands:

Category	Action	Command

Range	Specify range of x / y axis	`xlim(xmin, xmax)`
		`ylim(ymin, ymax)`

Nature	Display ticks in a customized manner	`scale_x_continuous()`
		`scale_x_discrete()`
		`scale_x_date()`

Magnitude	Transform axes	`scale_x_log10()`
		`scale_x_reverse()`
		`scale_x_sqrt()`

❒ **Double axes** – A plot can have more than one axis with the `sec.axis` option within a given scale function `scale_function()`. It is done as follows:

R

```
scale_function(sec.axis = sec_axis(~ .))
```

❒ **Figure saving** – Figures can be saved with a finetuned size of output image as follows:

R

```
ggsave(plot, filename, scale, width, height)
```