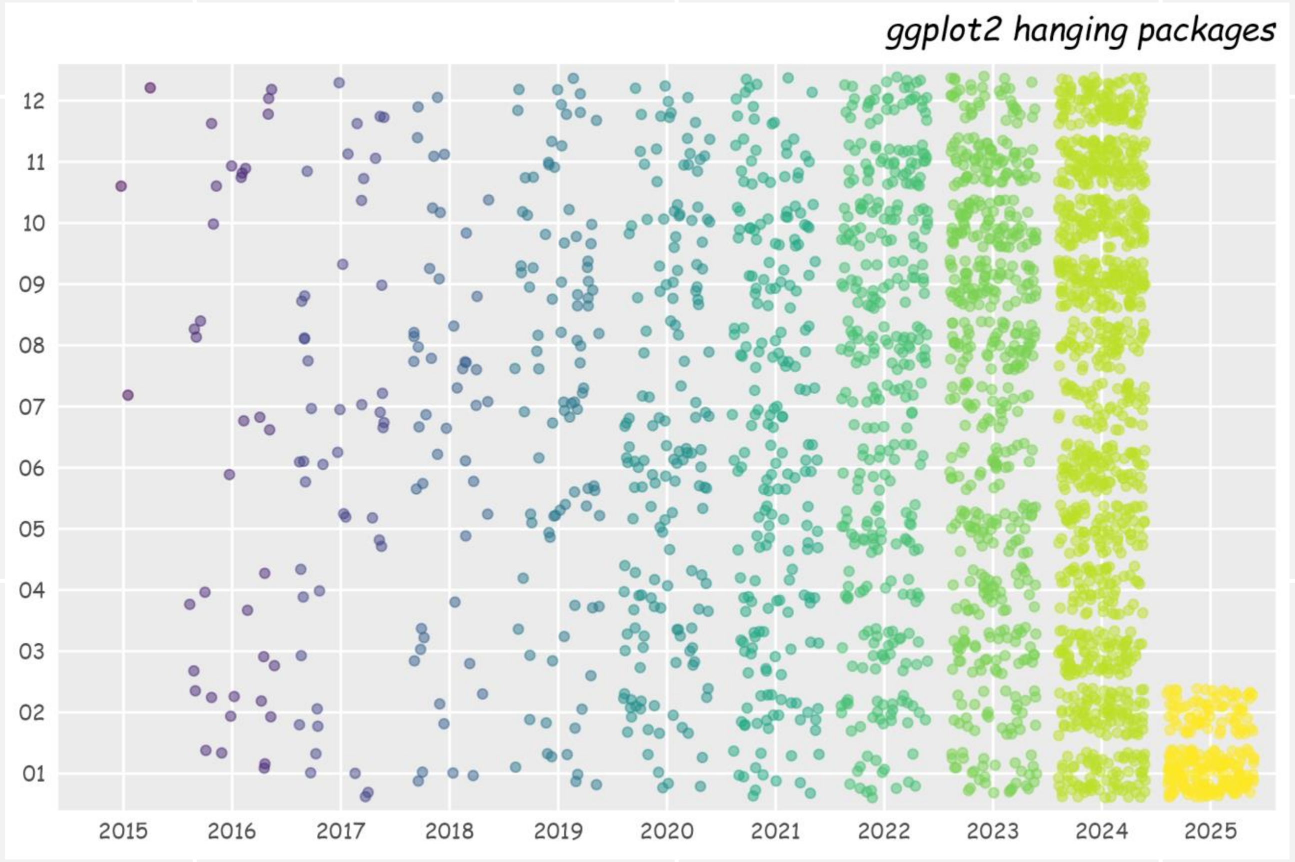
Visualización de datos en R: una aproximación a través del paquete **ggplot2**



https://ggplot2.tidyverse.org/

grammar of graphics:

“capas de información”

DATA

MAPPING

STATISTICS

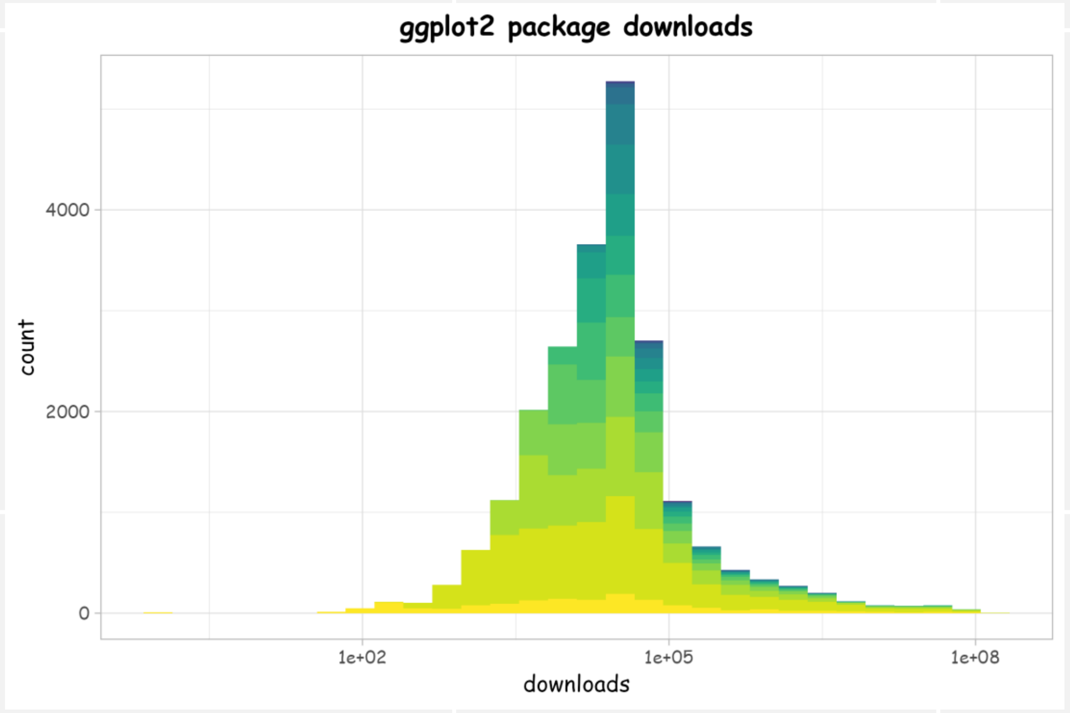
SCALES

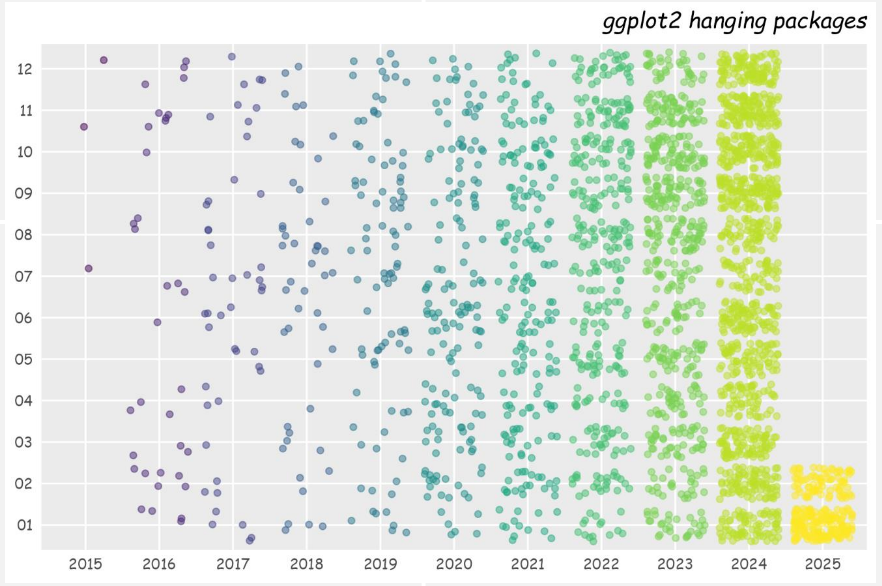
GEOMETRIES

FACETS

COORDINATES

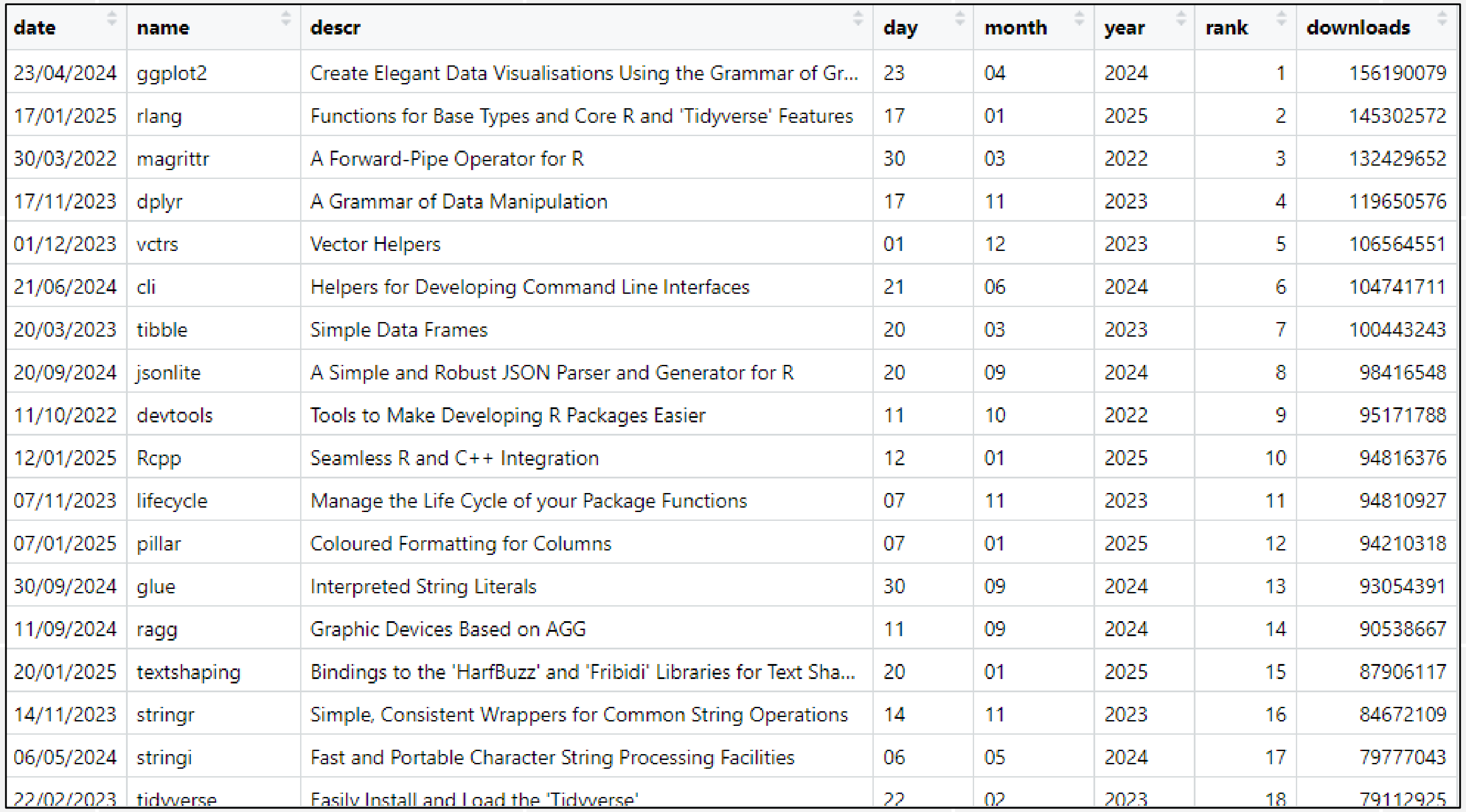
THEME





DATA

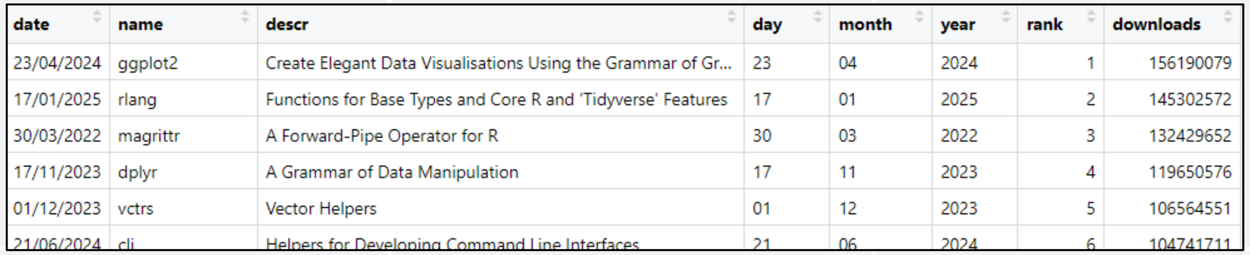
```r
ggplot(data)
```



date	name	descr	day	month	year	rank	downloads
23/04/2024	ggplot2	Create Elegant Data Visualisations Using the Grammar of Gr...	23	04	2024	1	156190079
17/01/2025	rlang	Functions for Base Types and Core R and 'Tidyverse' Features	17	01	2025	2	145302572
30/03/2022	magrittr	A Forward-Pipe Operator for R	30	03	2022	3	132429652
17/11/2023	dplyr	A Grammar of Data Manipulation	17	11	2023	4	119650576
01/12/2023	vctrs	Vector Helpers	01	12	2023	5	106564551
21/06/2024	cli	Helpers for Developing Command Line Interfaces	21	06	2024	6	104741711
20/03/2023	tibble	Simple Data Frames	20	03	2023	7	100443243
20/09/2024	jsonlite	A Simple and Robust JSON Parser and Generator for R	20	09	2024	8	98416548
11/10/2022	devtools	Tools to Make Developing R Packages Easier	11	10	2022	9	95171788
12/01/2025	Rcpp	Seamless R and C++ Integration	12	01	2025	10	94816376
07/11/2023	lifecycle	Manage the Life Cycle of your Package Functions	07	11	2023	11	94810927
07/01/2025	pillar	Coloured Formatting for Columns	07	01	2025	12	94210318
30/09/2024	glue	Interpreted String Literals	30	09	2024	13	93054391
11/09/2024	ragg	Graphic Devices Based on AGG	11	09	2024	14	90538667
20/01/2025	textshaping	Bindings to the 'HarfBuzz' and 'Fribidi' Libraries for Text Sha...	20	01	2025	15	87906117
14/11/2023	stringr	Simple, Consistent Wrappers for Common String Operations	14	11	2023	16	84672109
06/05/2024	stringi	Fast and Portable Character String Processing Facilities	06	05	2024	17	79777043
22/02/2023	tidyverse	Easily Install and Load the 'Tidyverse'	22	02	2023	18	79112925

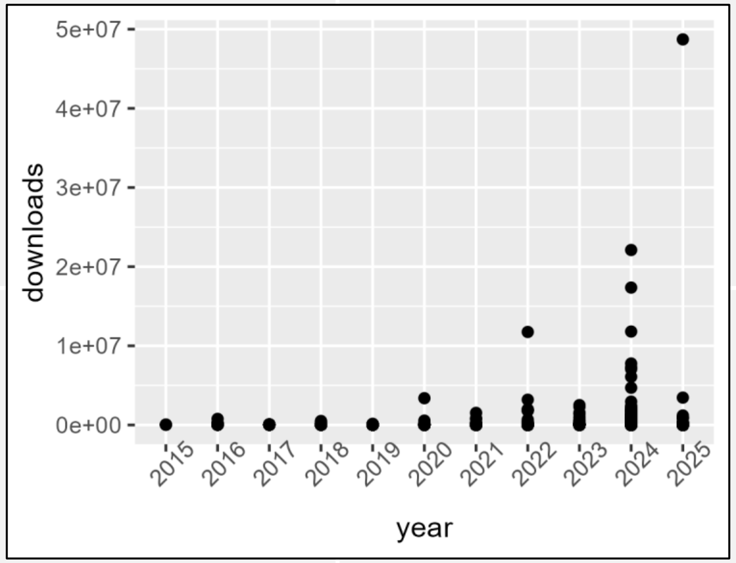
MAPPING

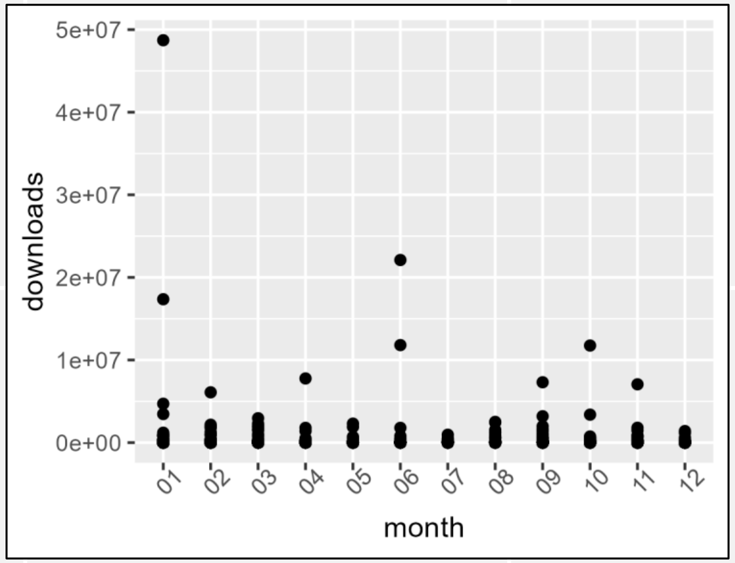
```
ggplot(data, mapping = aes(x = year, y = downloads))+
  geom_point()
```

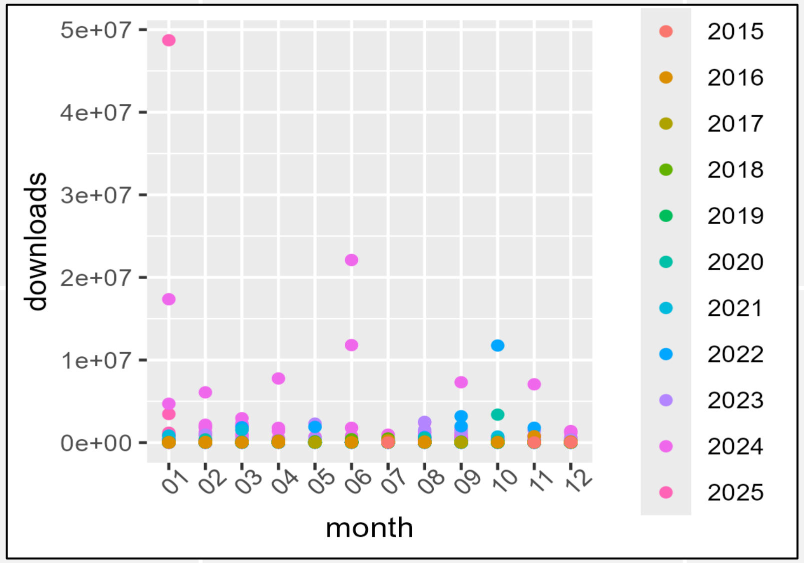


date	name	descr	day	month	year	rank	downloads
23/04/2024	ggplot2	Create Elegant Data Visualisations Using the Grammar of Gr...	23	04	2024	1	156190079
17/01/2025	rlang	Functions for Base Types and Core R and 'Tidyverse' Features	17	01	2025	2	145302572
30/03/2022	magrittr	A Forward-Pipe Operator for R	30	03	2022	3	132429652
17/11/2023	dplyr	A Grammar of Data Manipulation	17	11	2023	4	119650576
01/12/2023	vctrs	Vector Helpers	01	12	2023	5	106564551
21/06/2024	cli	Helpers for Developing Command Line Interfaces	21	06	2024	6	104741711

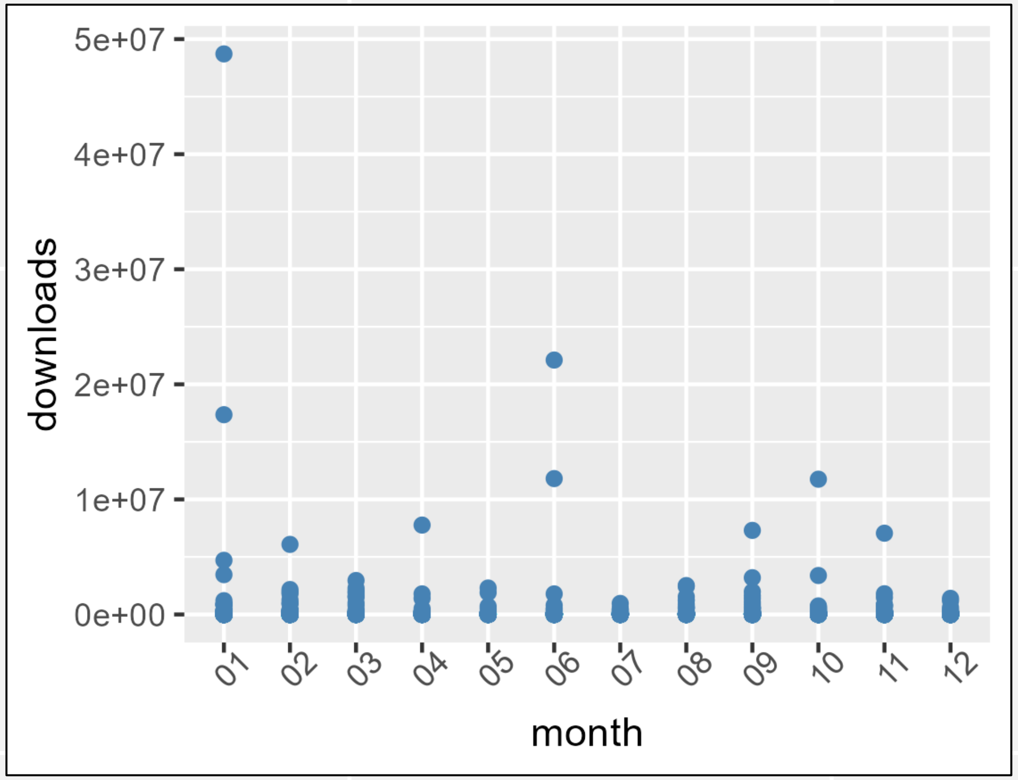
¿eje x?
¿eje y?
Otras variables…

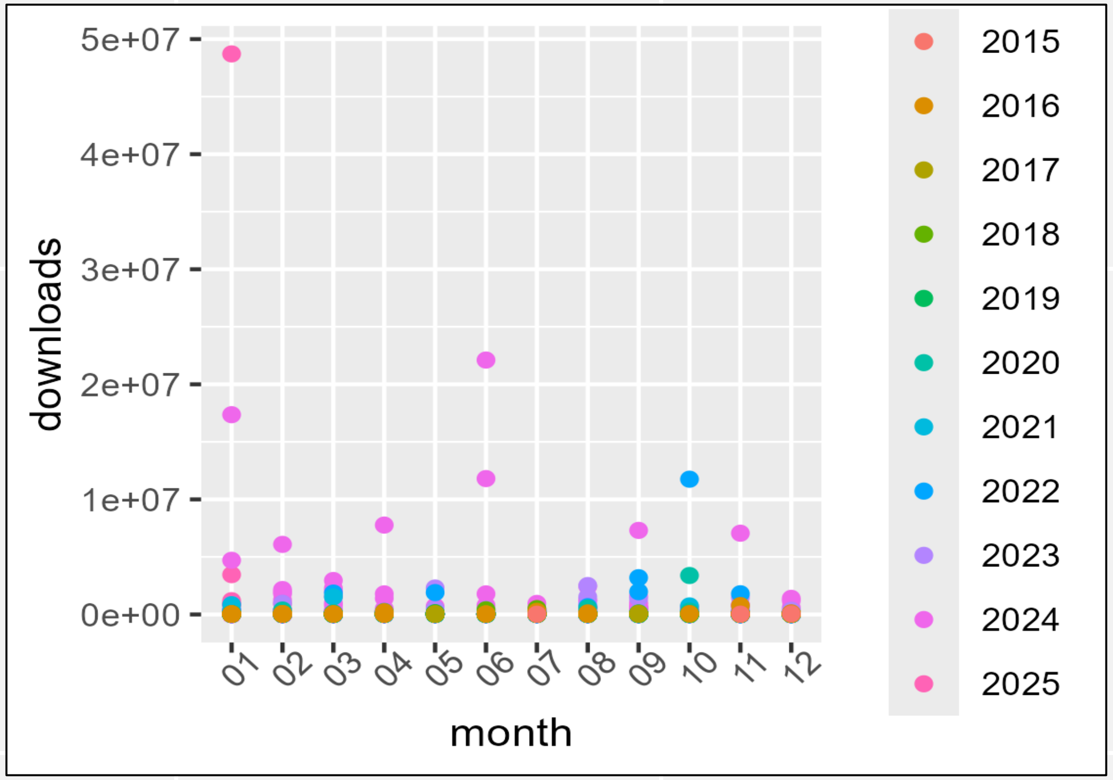






MAPPING



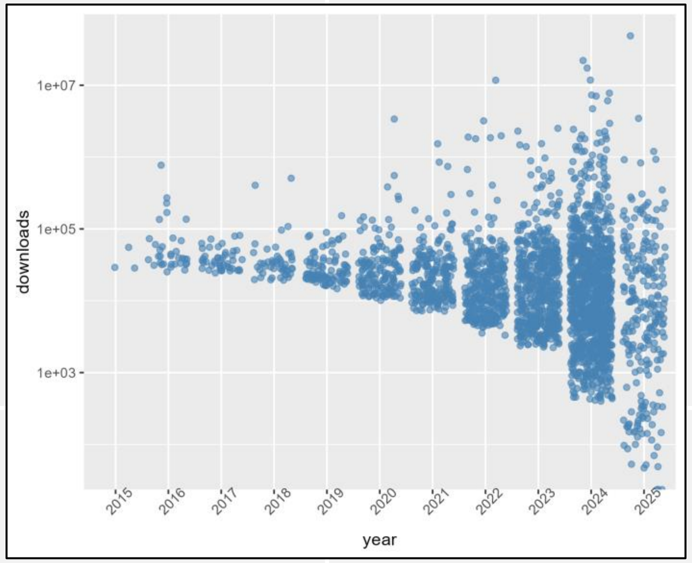


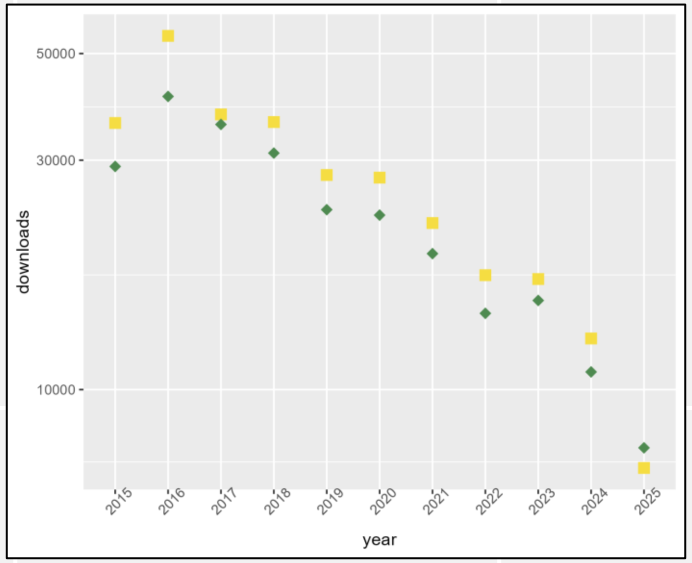
```
ggplot(data)+
  geom_point(aes(month, downloads) , colour="blue")
```

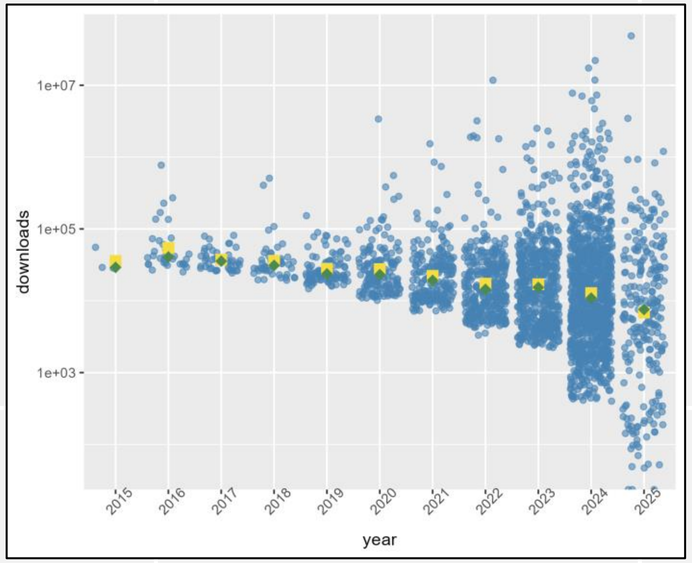
```
ggplot(data)+
  geom_point(aes(month, downloads, colour=year))
```

STATISTICS

```
ggplot(data)+
  stat_summary(aes(x = year, y = downloads), fun = mean)
```



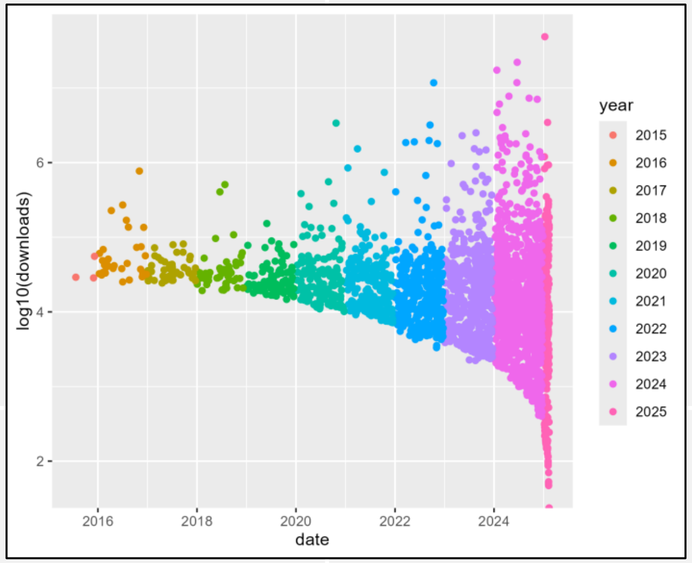


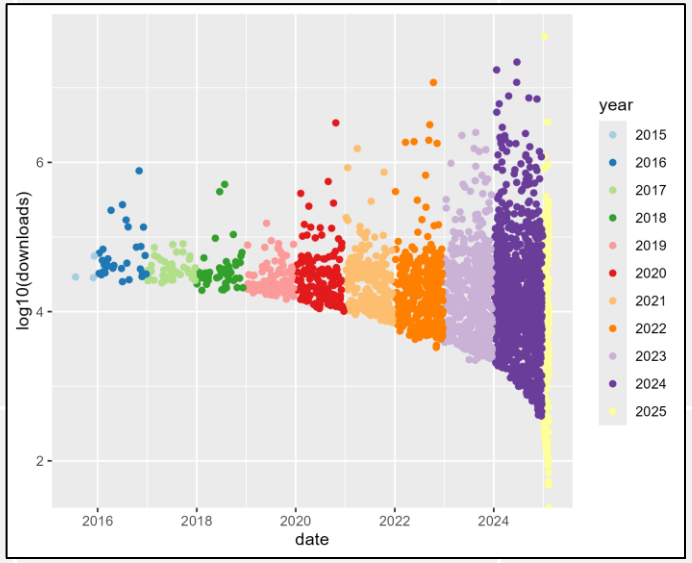


SCALES

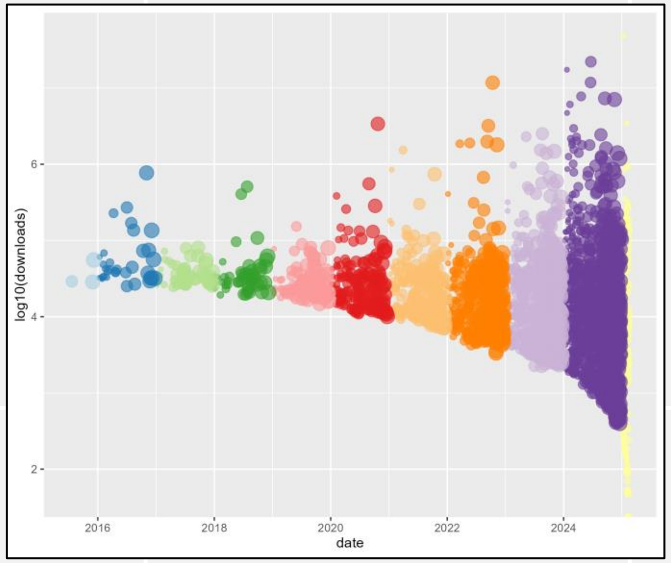
```
ggplot(data, mapping)+
  geom_xxx()+
  scale_<aes>_<type>()
```

```
ggplot(data)+
  geom_point(aes(date, log10(downloads), colour=year))
```



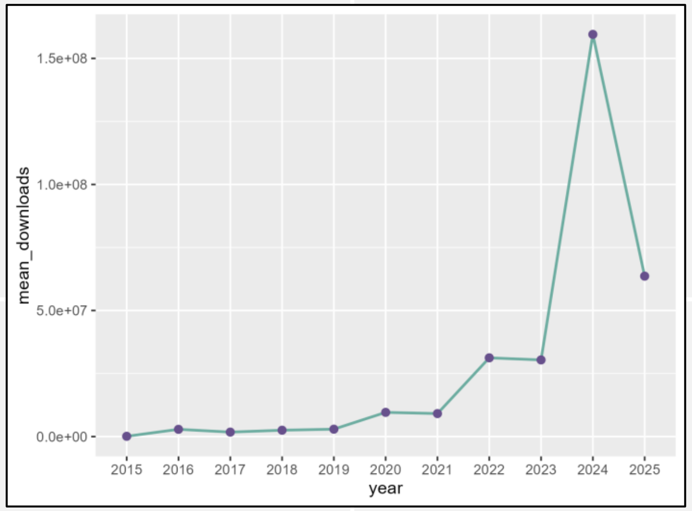


+ scale_colour_brewer(palette)

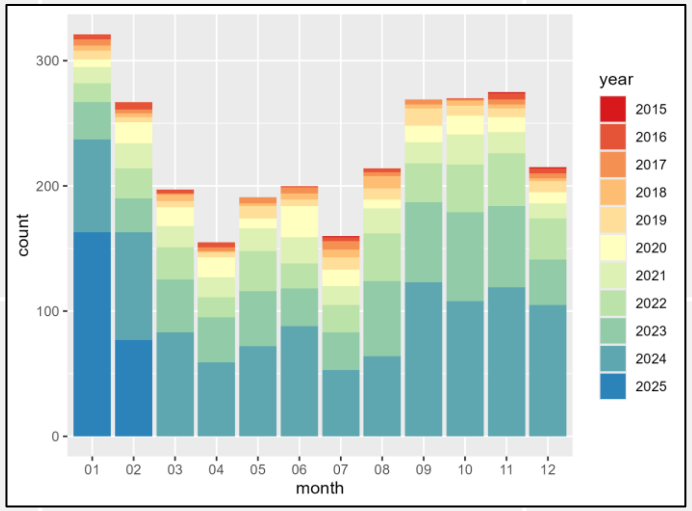


+ scale_size_area()

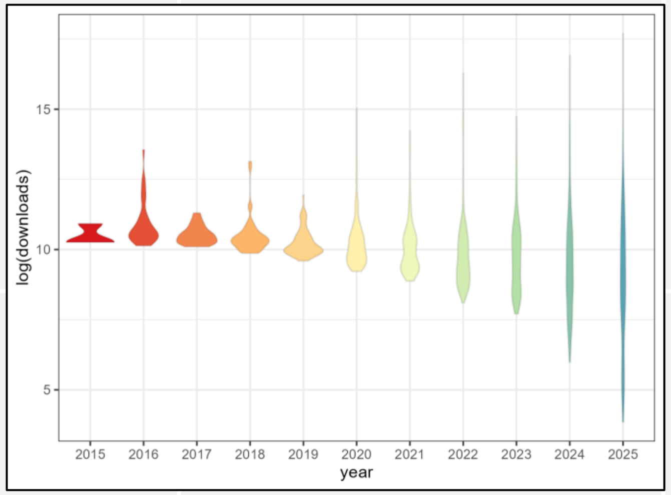
GEOMETRIES



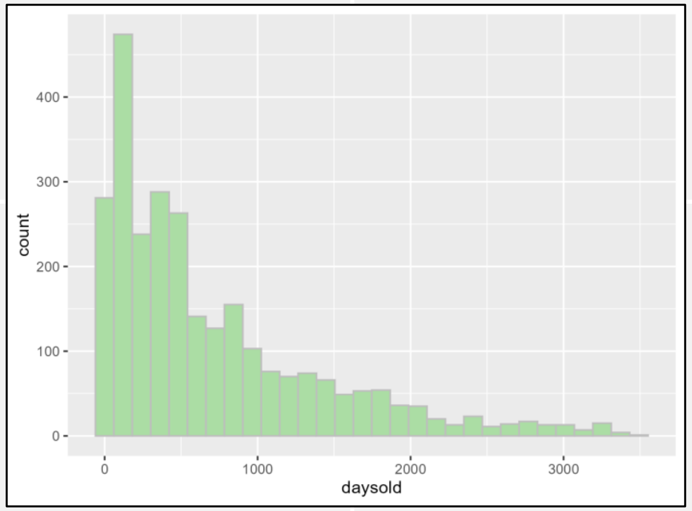
geom_point( ) & geom_line( )



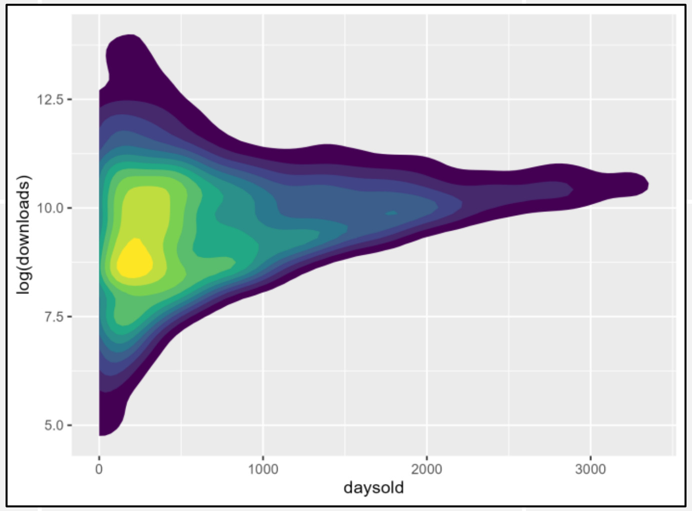
geom_bar( )



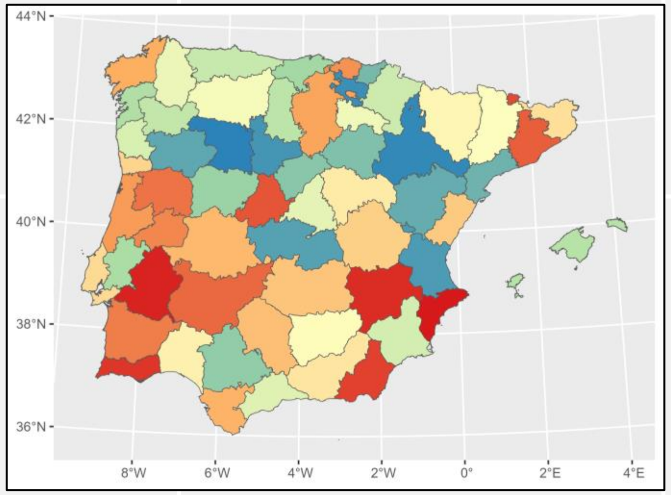
geom_violin( )



geom_histogram( )

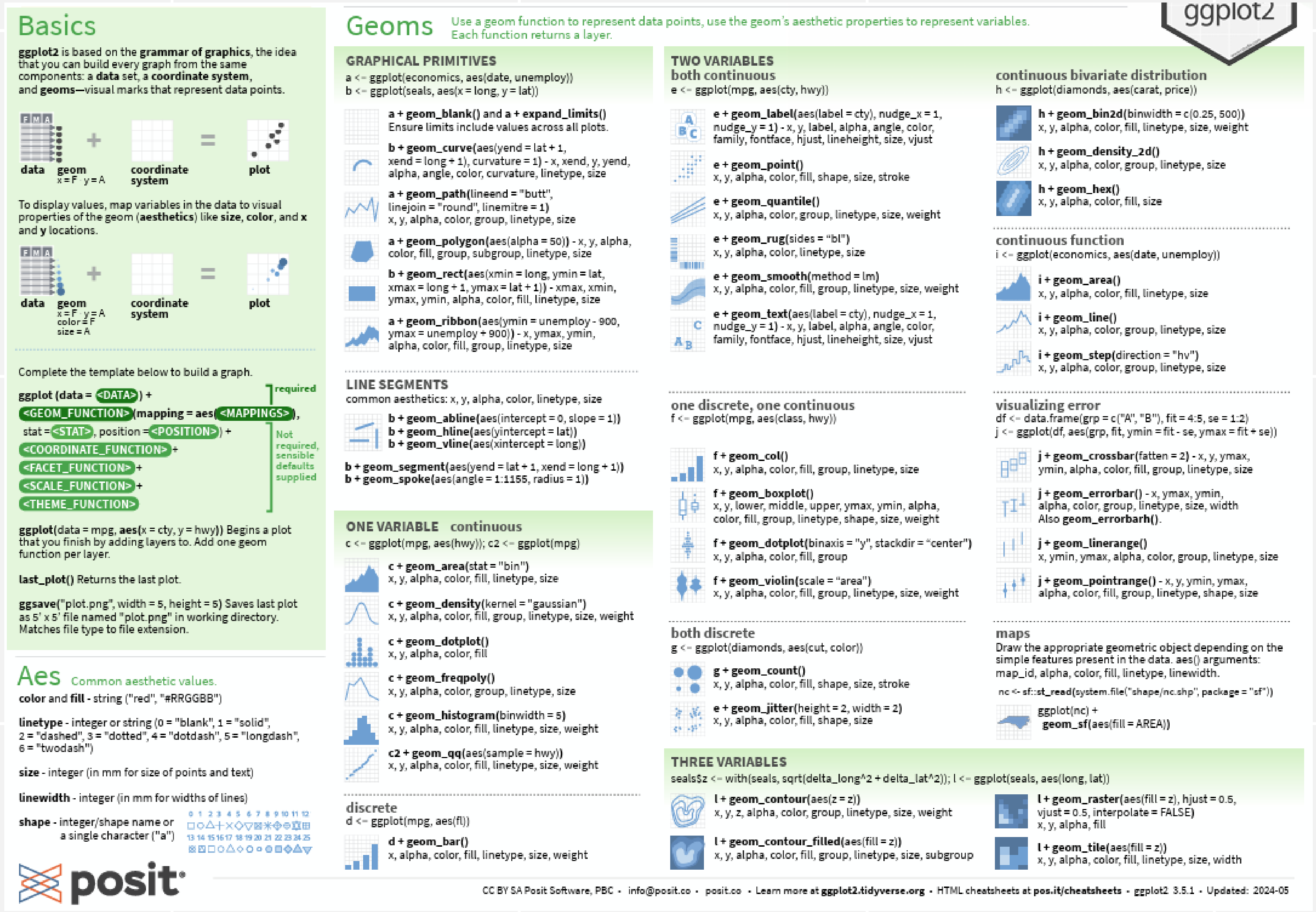


stat_density_2d ( )



geom_spatvect( )

GEOMETRIES



Basics

ggplot2 is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data** set, a **coordinate system**, and **geoms**—visual marks that represent data points.

F M A
data
geom
x = F · y = A
+
coordinate system
=
plot

To display values, map variables in the data to visual properties of the geom (**aesthetics**) like **size**, **color**, and **x** and **y** locations.

F M A
data
geom
x = F · y = A
color = F
size = A
+
coordinate system
=
plot

Complete the template below to build a graph.

```
ggplot (data = <DATA>) +                                  required
<GEOM_FUNCTION>(mapping = aes(<MAPPINGS>),
 stat = <STAT>, position = <POSITION>) +                  Not
<COORDINATE_FUNCTION> +                                   required,
<FACET_FUNCTION> +                                        sensible
<SCALE_FUNCTION> +                                        defaults
<THEME_FUNCTION>                                          supplied
```

ggplot(data = mpg, **aes**(x = cty, y = hwy)) Begins a plot that you finish by adding layers to. Add one geom function per layer.

last_plot() Returns the last plot.

ggsave("plot.png", width = 5, height = 5) Saves last plot as 5' x 5' file named "plot.png" in working directory. Matches file type to file extension.

Aes Common aesthetic values.

color and **fill** - string ("red", "#RRGGBB")

linetype - integer or string (0 = "blank", 1 = "solid", 2 = "dashed", 3 = "dotted", 4 = "dotdash", 5 = "longdash", 6 = "twodash")

size - integer (in mm for size of points and text)

linewidth - integer (in mm for widths of lines)

shape - integer/shape name or a single character ("a")

posit

Geoms

Use a geom function to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

ggplot2

GRAPHICAL PRIMITIVES

a <- ggplot(economics, aes(date, unemploy))
b <- ggplot(seals, aes(x = long, y = lat))

a + **geom_blank**() and a + **expand_limits**()
Ensure limits include values across all plots.

b + **geom_curve**(aes(yend = lat + 1, xend = long + 1), curvature = 1) - x, xend, y, yend, alpha, angle, color, curvature, linetype, size

a + **geom_path**(lineend = "butt", linejoin = "round", linemitre = 1)
x, y, alpha, color, group, linetype, size

a + **geom_polygon**(aes(alpha = 50)) - x, y, alpha, color, fill, group, subgroup, linetype, size

b + **geom_rect**(aes(xmin = long, ymin = lat, xmax = long + 1, ymax = lat + 1)) - xmax, xmin, ymax, ymin, alpha, color, fill, linetype, size

a + **geom_ribbon**(aes(ymin = unemploy - 900, ymax = unemploy + 900)) - x, ymax, ymin, alpha, color, fill, group, linetype, size

LINE SEGMENTS

common aesthetics: x, y, alpha, color, linetype, size

b + **geom_abline**(aes(intercept = 0, slope = 1))
b + **geom_hline**(aes(yintercept = lat))
b + **geom_vline**(aes(xintercept = long))

b + **geom_segment**(aes(yend = lat + 1, xend = long + 1))
b + **geom_spoke**(aes(angle = 1:1155, radius = 1))

ONE VARIABLE continuous

c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)

c + **geom_area**(stat = "bin")
x, y, alpha, color, fill, linetype, size

c + **geom_density**(kernel = "gaussian")
x, y, alpha, color, fill, group, linetype, size, weight

c + **geom_dotplot**()
x, y, alpha, color, fill

c + **geom_freqpoly**()
x, y, alpha, color, group, linetype, size

c + **geom_histogram**(binwidth = 5)
x, y, alpha, color, fill, linetype, size, weight

c2 + **geom_qq**(aes(sample = hwy))
x, y, alpha, color, fill, linetype, size, weight

discrete

d <- ggplot(mpg, aes(fl))

d + **geom_bar**()
x, alpha, color, fill, linetype, size, weight

TWO VARIABLES

both continuous

e <- ggplot(mpg, aes(cty, hwy))

e + **geom_label**(aes(label = cty), nudge_x = 1, nudge_y = 1) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

e + **geom_point**()
x, y, alpha, color, fill, shape, size, stroke

e + **geom_quantile**()
x, y, alpha, color, group, linetype, size, weight

e + **geom_rug**(sides = "bl")
x, y, alpha, color, linetype, size

e + **geom_smooth**(method = lm)
x, y, alpha, color, fill, group, linetype, size, weight

e + **geom_text**(aes(label = cty), nudge_x = 1, nudge_y = 1) - x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, vjust

one discrete, one continuous

f <- ggplot(mpg, aes(class, hwy))

f + **geom_col**()
x, y, alpha, color, fill, group, linetype, size

f + **geom_boxplot**()
x, y, lower, middle, upper, ymax, ymin, alpha, color, fill, group, linetype, shape, size, weight

f + **geom_dotplot**(binaxis = "y", stackdir = "center")
x, y, alpha, color, fill, group

f + **geom_violin**(scale = "area")
x, y, alpha, color, fill, group, linetype, size, weight

both discrete

g <- ggplot(diamonds, aes(cut, color))

g + **geom_count**()
x, y, alpha, color, fill, shape, size, stroke

e + **geom_jitter**(height = 2, width = 2)
x, y, alpha, color, fill, shape, size

continuous bivariate distribution

h <- ggplot(diamonds, aes(carat, price))

h + **geom_bin2d**(binwidth = c(0.25, 500))
x, y, alpha, color, fill, linetype, size, weight

h + **geom_density_2d**()
x, y, alpha, color, group, linetype, size

h + **geom_hex**()
x, y, alpha, color, fill, size

continuous function

i <- ggplot(economics, aes(date, unemploy))

i + **geom_area**()
x, y, alpha, color, fill, linetype, size

i + **geom_line**()
x, y, alpha, color, group, linetype, size

i + **geom_step**(direction = "hv")
x, y, alpha, color, group, linetype, size

visualizing error

df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)
j <- ggplot(df, aes(grp, fit, ymin = fit - se, ymax = fit + se))

j + **geom_crossbar**(fatten = 2) - x, y, ymax, ymin, alpha, color, fill, group, linetype, size

j + **geom_errorbar**() - x, ymax, ymin, alpha, color, group, linetype, size, width
Also **geom_errorbarh**().

j + **geom_linerange**()
x, ymin, ymax, alpha, color, group, linetype, size

j + **geom_pointrange**() - x, y, ymin, ymax, alpha, color, fill, group, linetype, shape, size

maps

Draw the appropriate geometric object depending on the simple features present in the data. aes() arguments: map_id, alpha, color, fill, linetype, linewidth.

nc <- sf::st_read(system.file("shape/nc.shp", package = "sf"))

ggplot(nc) +
 geom_sf(aes(fill = AREA))

THREE VARIABLES

seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2)); l <- ggplot(seals, aes(long, lat))

l + **geom_contour**(aes(z = z))
x, y, z, alpha, color, group, linetype, size, weight

l + **geom_contour_filled**(aes(fill = z))
x, y, alpha, color, fill, group, linetype, size, subgroup

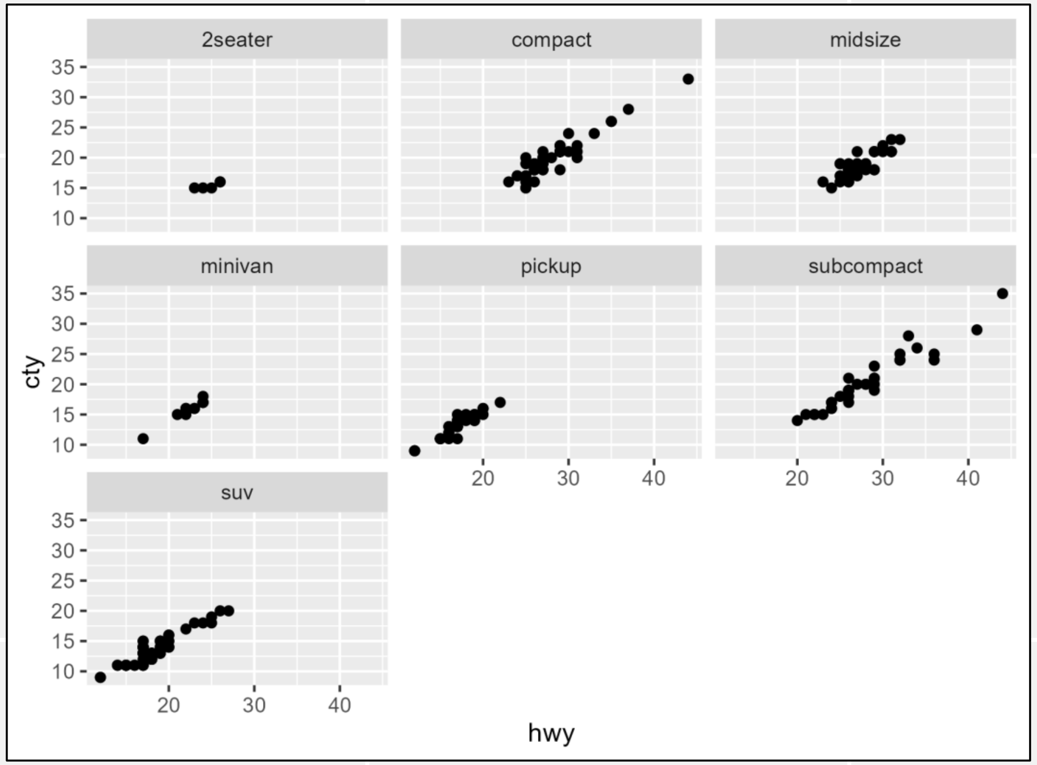
l + **geom_raster**(aes(fill = z), hjust = 0.5, vjust = 0.5, interpolate = FALSE)
x, y, alpha, fill

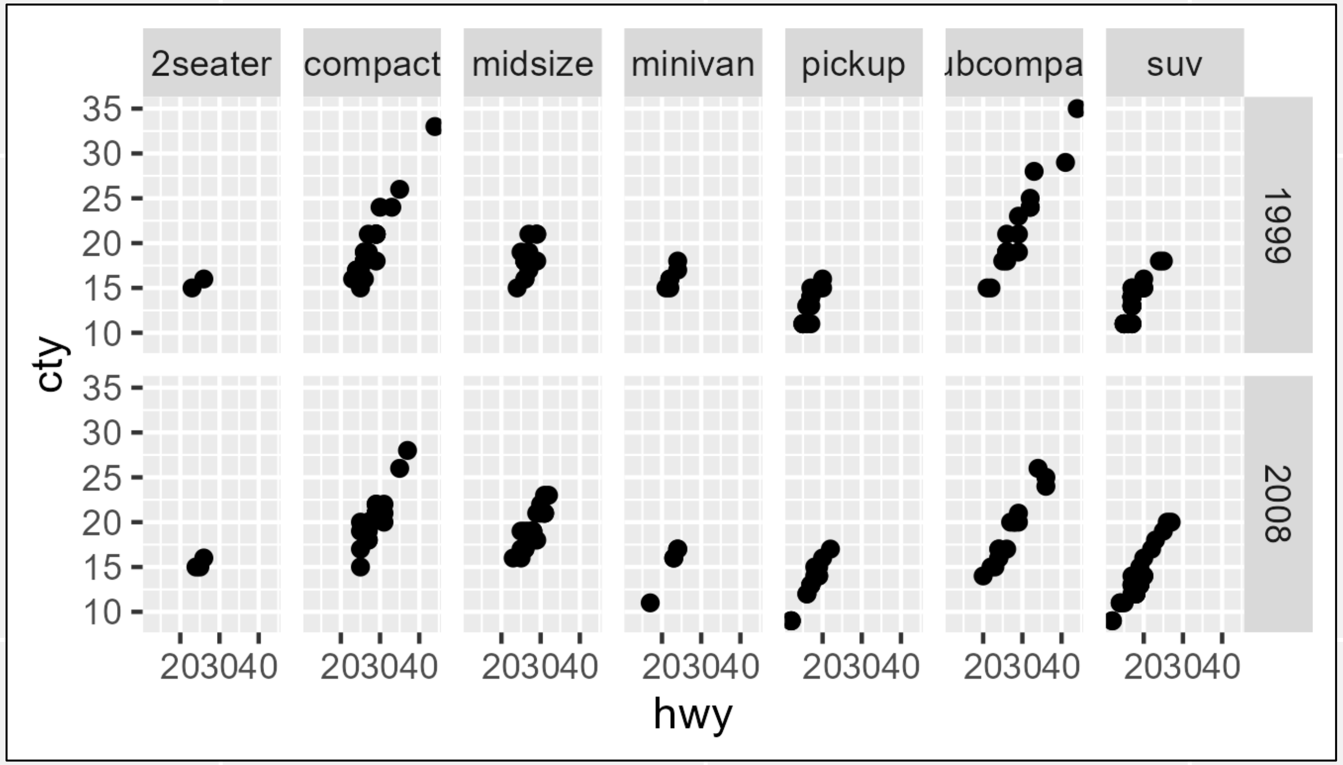
l + **geom_tile**(aes(fill = z))
x, y, alpha, color, fill, linetype, size, width

https://rstudio.github.io/cheatsheets/data-visualization.pdf

FACETS

```
ggplot(data, mapping)+
  geom_point()+
```



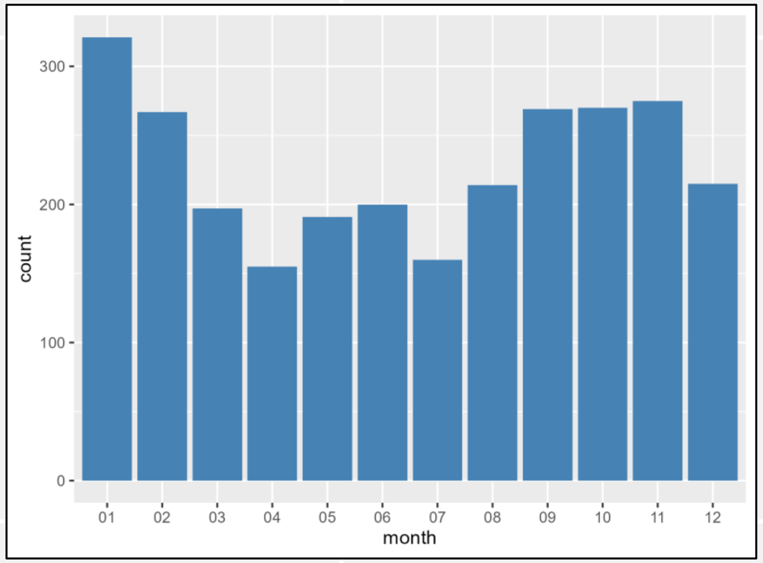


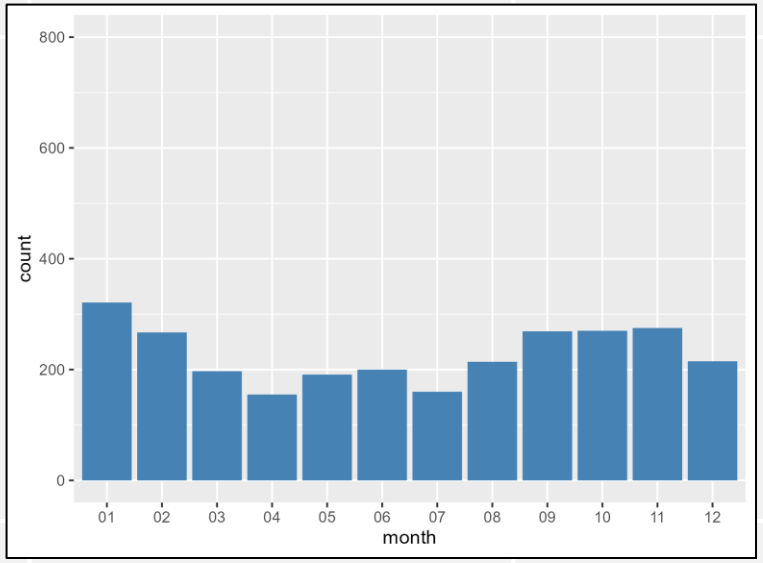
```
facet_wrap( ~var1 )
```

```
facet_grid(var1~var2)
```

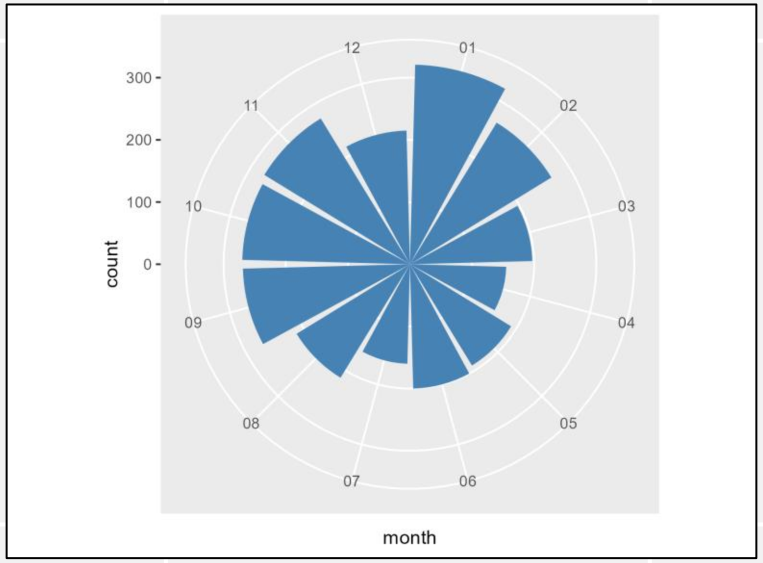
COORDINATES

```
ggplot(data, mapping)+
  geom_bar(fill)+
```



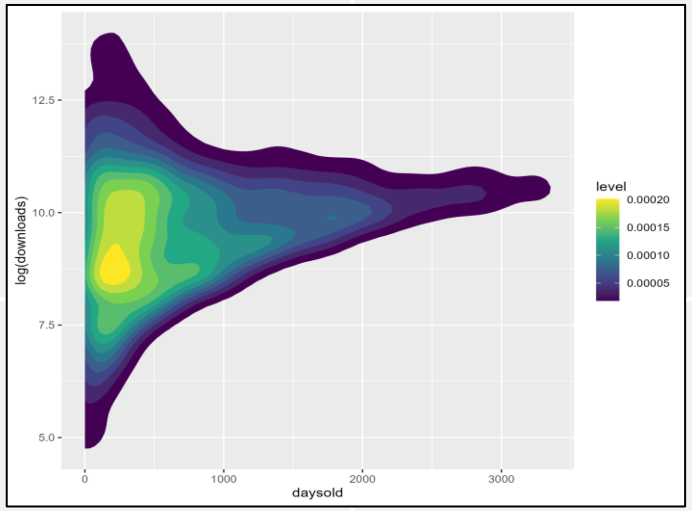


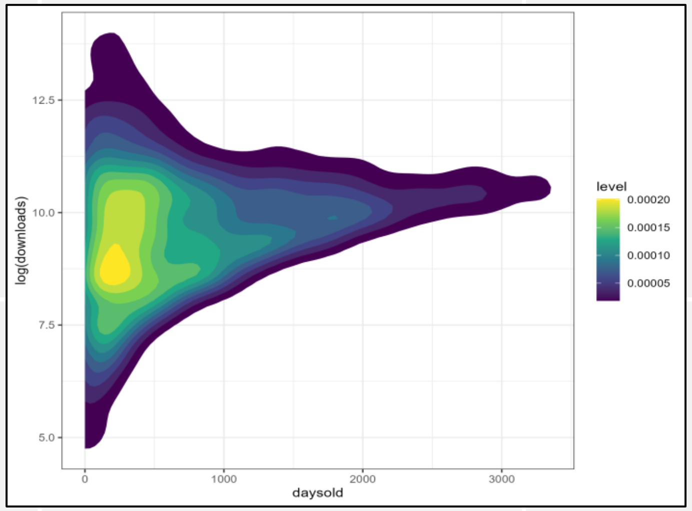
+ coords_trans(ylim = c(0, 800))



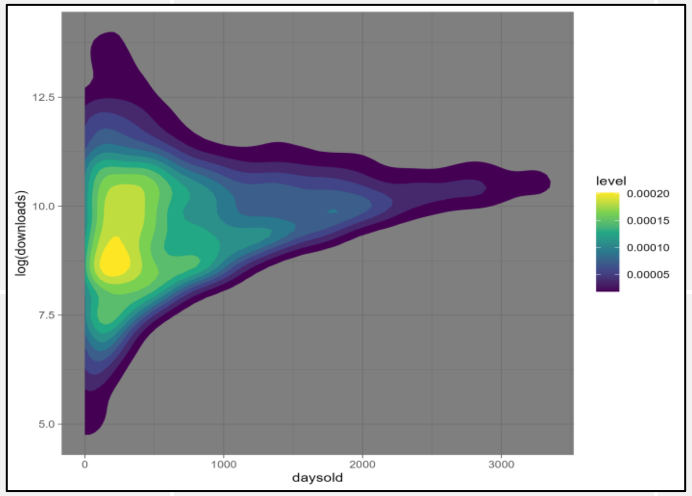
+ coord_polar()

THEME

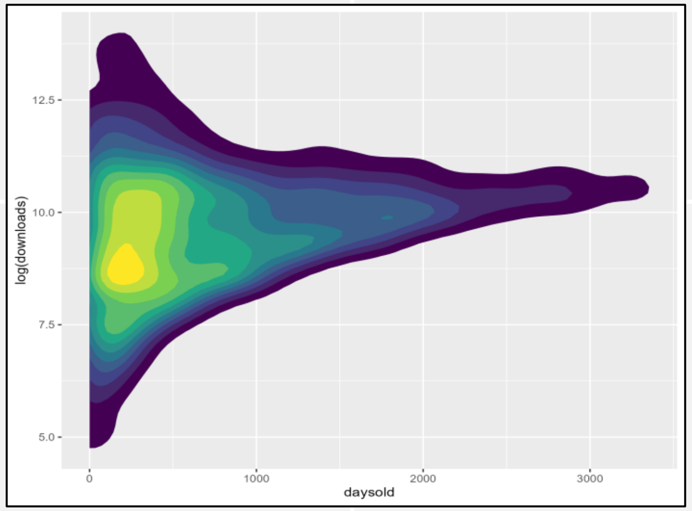




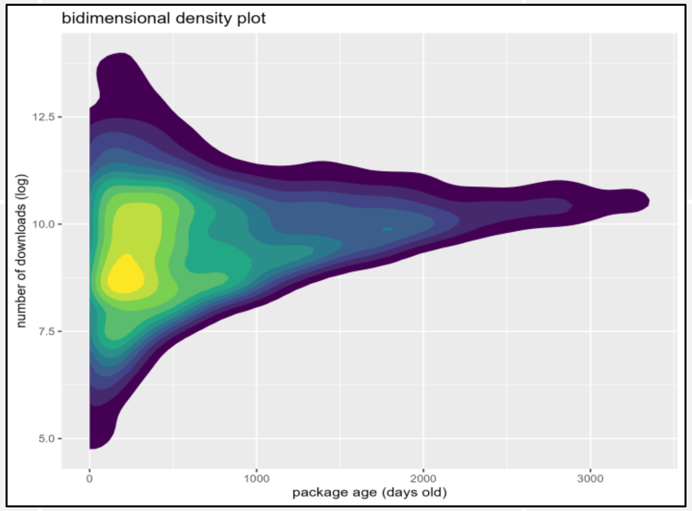
+theme_bw( )



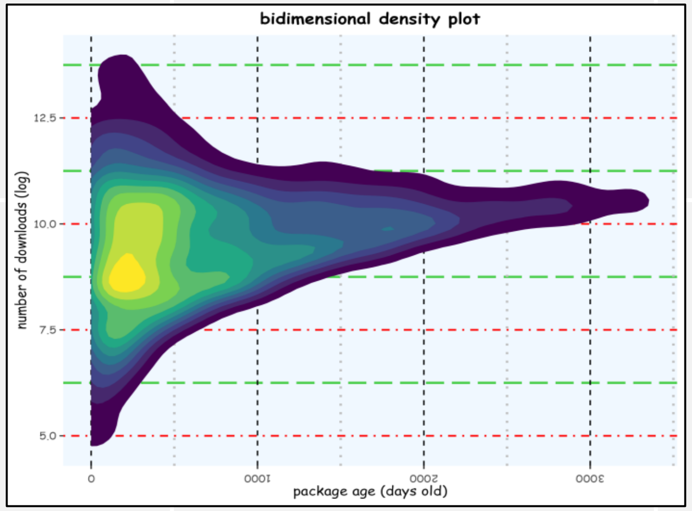
+theme_dark( )



+theme(legend.position = "none")



+labs(title, x, y)



+labs(...) + theme(...)