Datacamp_Data Visualization with ggplot2 (Part 2)___Coordinates and Facets

dizhen

2019/4/11

Coordinates

1. Coordinates Layer

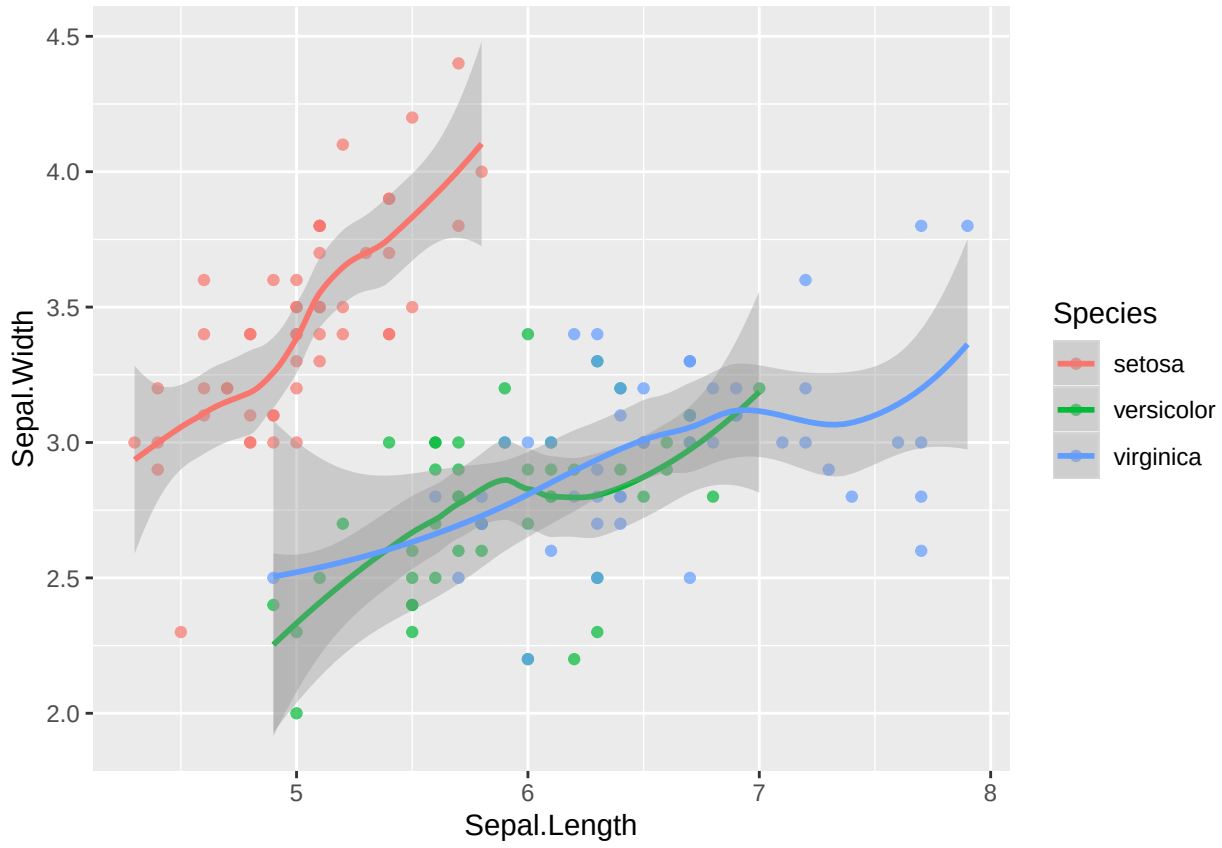
- Controls plot dimensions
- coord_
- coord_cartesian()

2. Zooming in

- scale_x_continuous(limits = ...)
- xlim()
- coord_cartesian(xlim = ...)

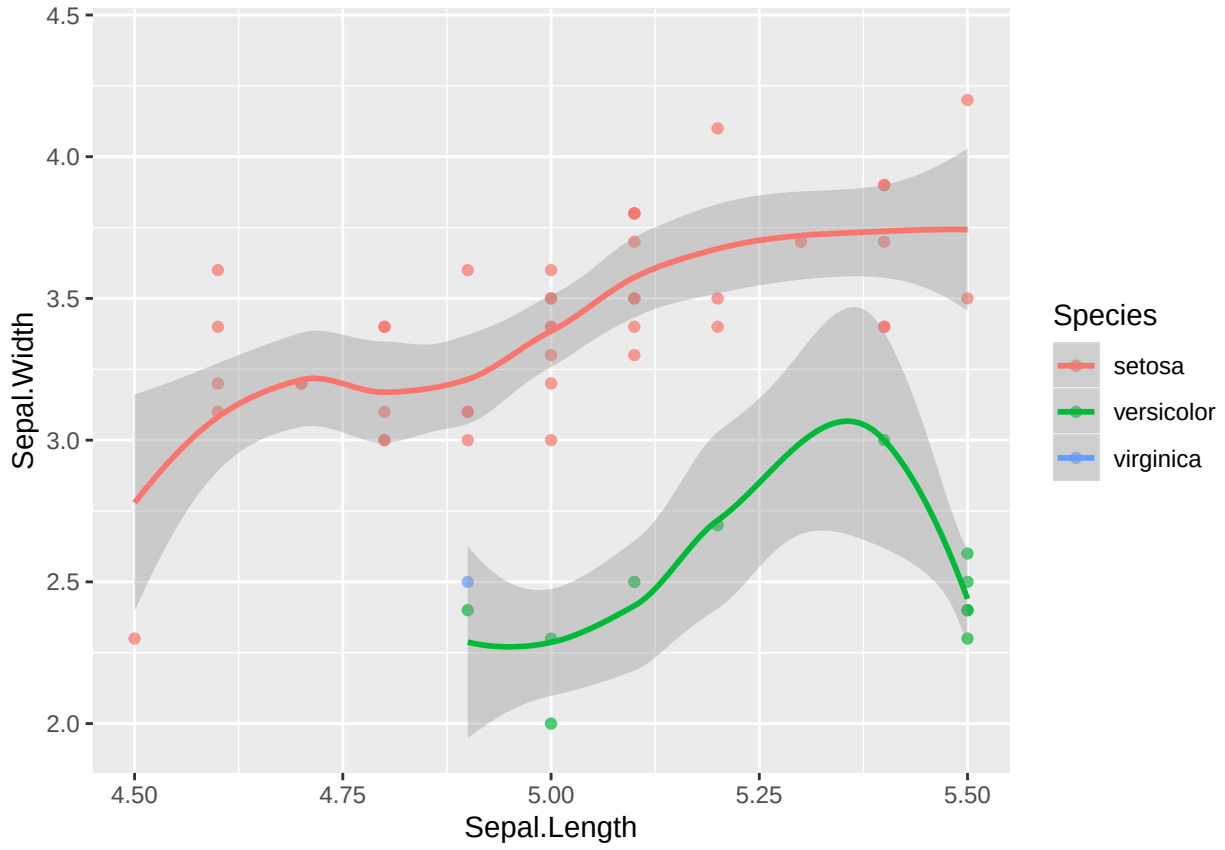
```
library("ggplot2")

# Original Plot
iris.smooth <- ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, col = Species)) +
  geom_point(alpha = 0.7) +
  geom_smooth()
iris.smooth
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```




```
# scale_x_continous
iris.smooth + scale_x_continuous(limits = c(4.5, 5.5))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'

## Warning: Removed 95 rows containing non-finite values (stat_smooth).

## Warning: Removed 95 rows containing missing values (geom_point).
```

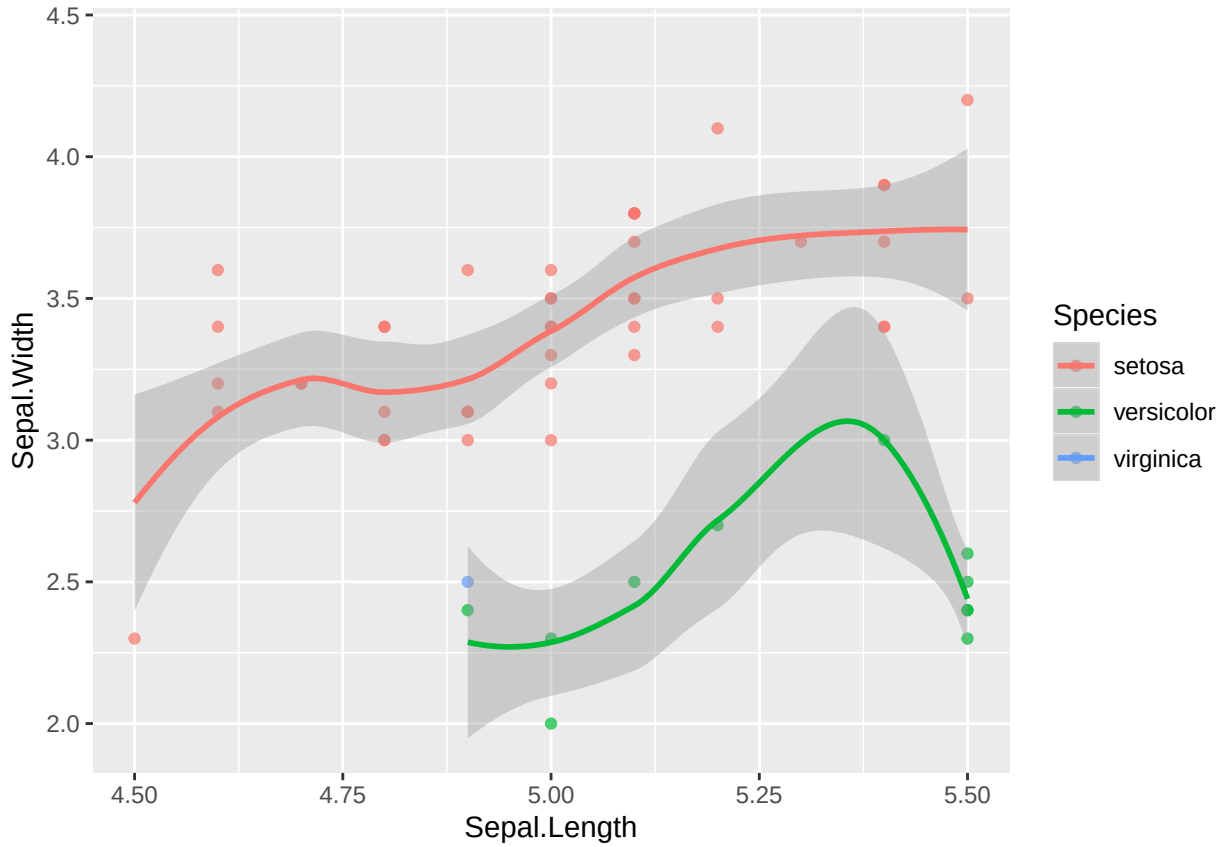



```
# xlim
iris.smooth + xlim(c(4.5, 5.5))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```

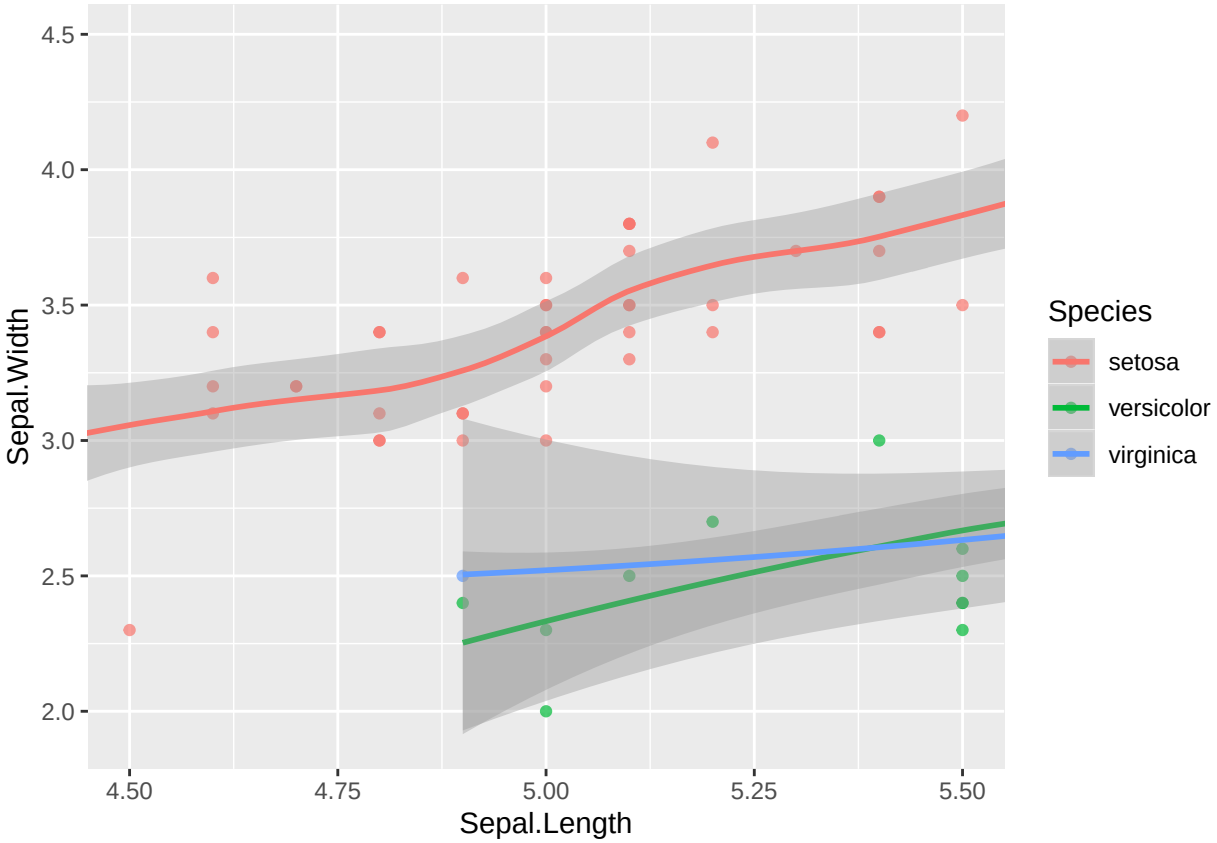
```
## Warning: Removed 95 rows containing non-finite values (stat_smooth).
```

```
## Warning: Removed 95 rows containing missing values (geom_point).
```




```
# coord_cartesian
iris.smooth + coord_cartesian(xlim = c(4.5, 5.5))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



3. Aspect Ratio

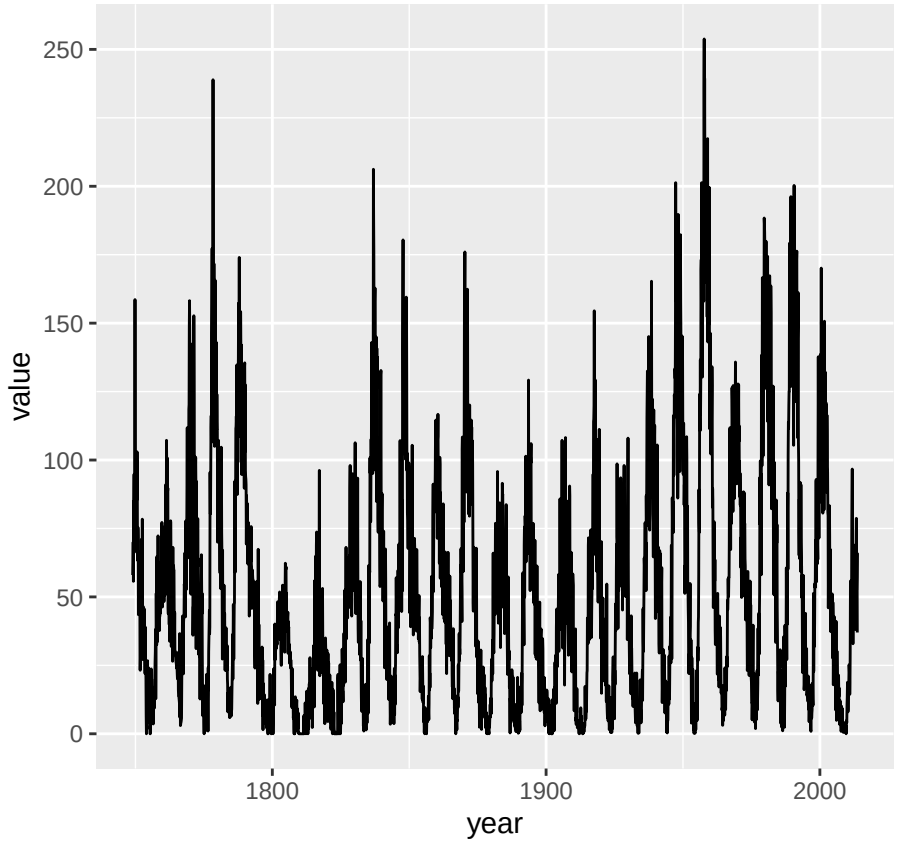
- Height-to-width ratio
- Deception
- Standardization a!empts
- Typically 1:1

```
# Sunspots
library(reshape2); library(zoo)
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

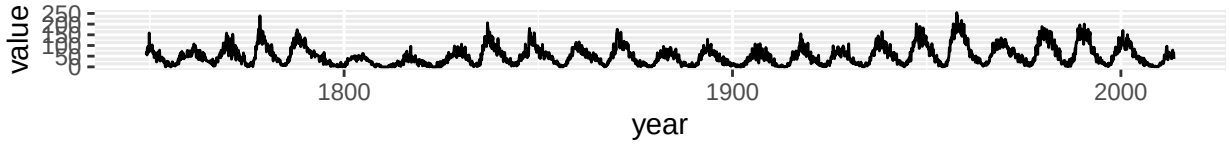
```
sunspots.m <- data.frame(year = index(sunspot.month), value = melt(sunspot.month)$value)

ggplot(sunspots.m, aes(x = year, y = value)) +
  geom_line() +
  coord_equal() # a 1:1 aspect ratio
```

```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```




```
ggplot(sunspots.m, aes(x = year, y = value)) +
  geom_line() +
  coord_fixed(0.055)
```

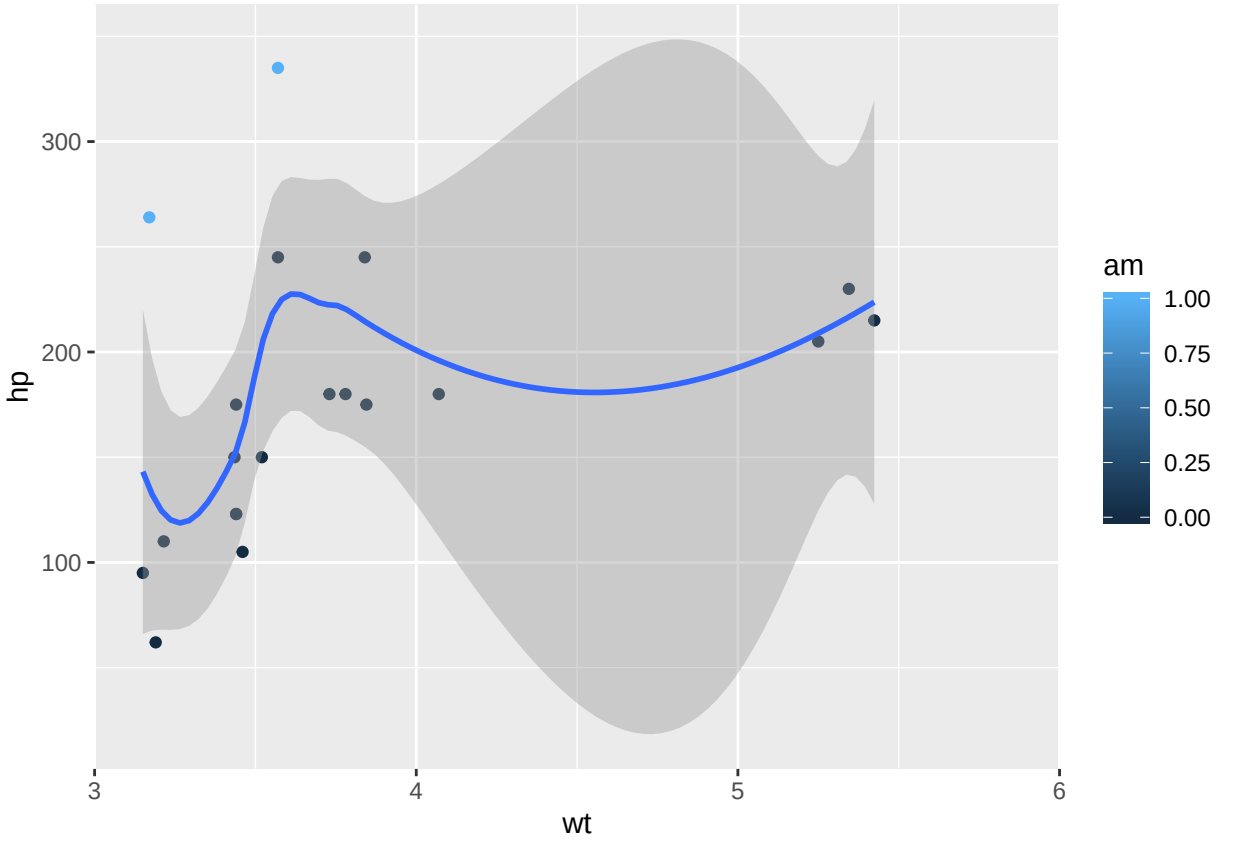
```
## Don't know how to automatically pick scale for object of type ts. Defaulting to continuous.
```



Practice

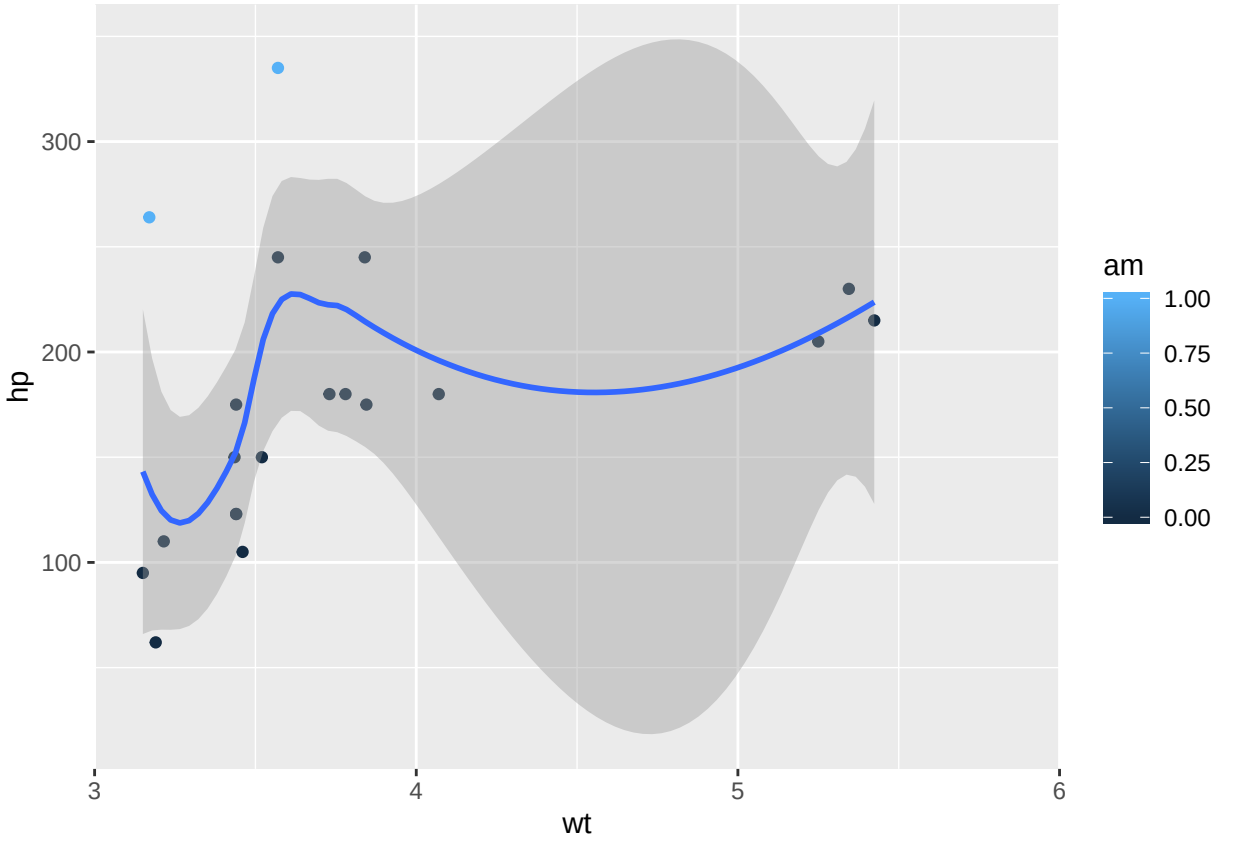
```
# Basic ggplot() command, coded for you
p <- ggplot(mtcars, aes(x = wt, y = hp, col = am)) + geom_point() + geom_smooth()

# Add scale_x_continuous()
p + scale_x_continuous(limit = c(3,6),expand = c(0,0))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'

## Warning: Removed 12 rows containing non-finite values (stat_smooth).

## Warning: Removed 12 rows containing missing values (geom_point).
```




```
# Add coord_cartesian(): the proper way to zoom in
p + scale_x_continuous(limit = c(3,6),expand = c(0,0)) + coord_cartesian(xlim = c(3,6))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'

## Warning: Removed 12 rows containing non-finite values (stat_smooth).

## Warning: Removed 12 rows containing missing values (geom_point).
```



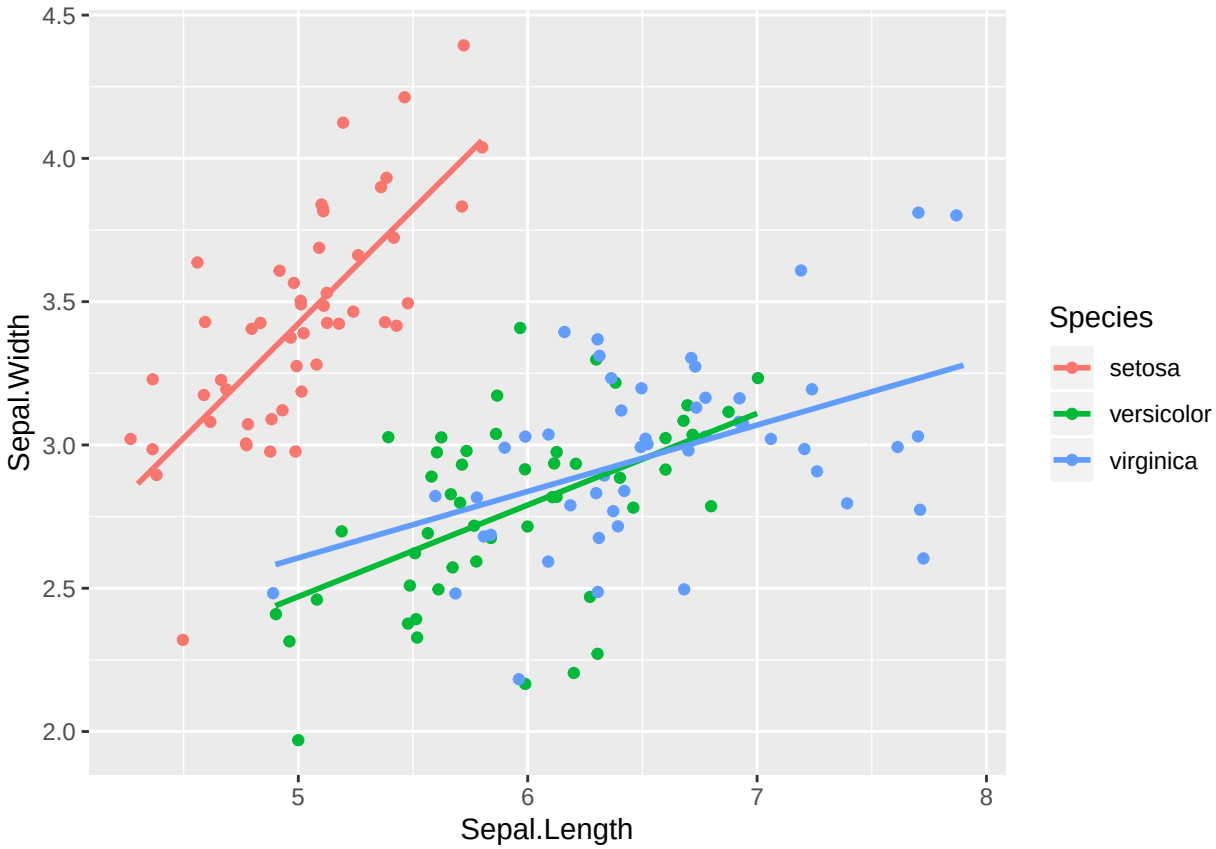
Aspect Ratio

We can set the aspect ratio of a plot with coord_fixed() or coord_equal(). Both use ratio = 1 as a default. A 1:1 aspect ratio is most appropriate when two continuous variables are on the same scale, as with the iris dataset.

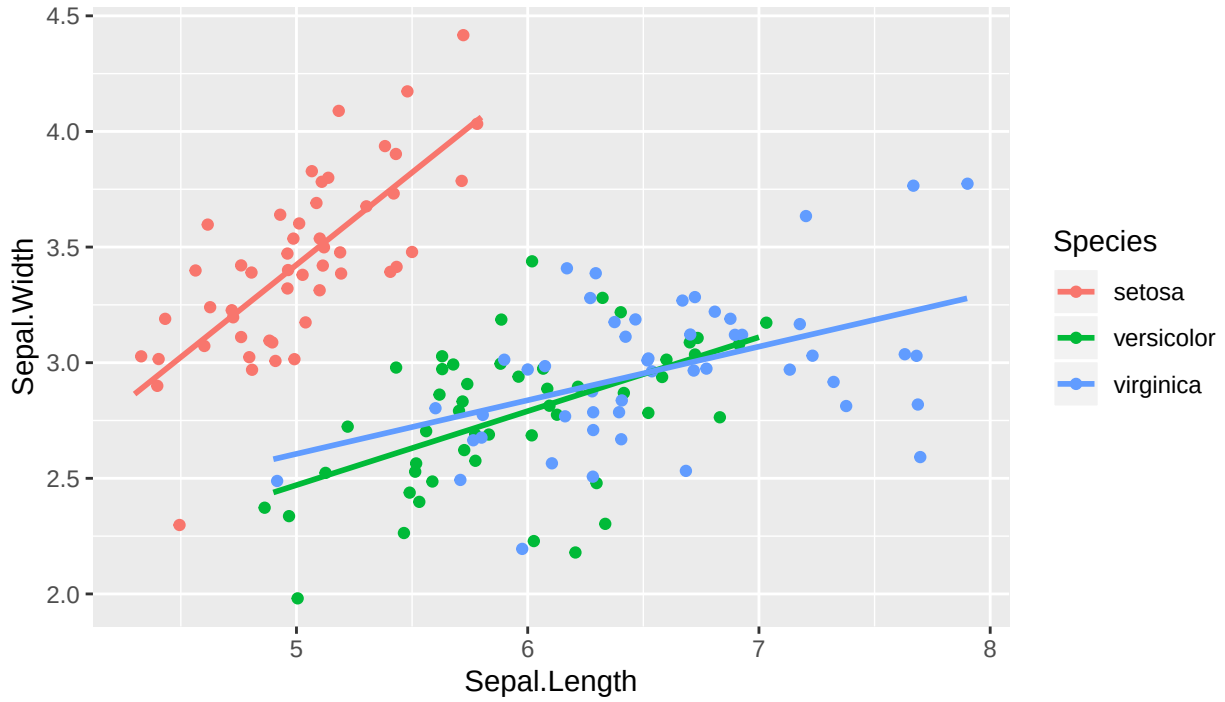
All variables are measured in centimeters, so it only makes sense that one unit on the plot should be the same physical distance on each axis. This gives a more truthful depiction of the relationship between the two variables since the aspect ratio can change the angle of our smoothing line. This would give an erroneous impression of the data.

Of course the underlying linear models don't change, but our perception can be influenced by the angle drawn.

```
# Complete basic scatter plot function
base.plot <- ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, col = Species)) +
               geom_jitter() +
               geom_smooth(method = "lm", se = FALSE)

# Plot base.plot: default aspect ratio
base.plot
```




```
# Fix aspect ratio (1:1) of base.plot
base.plot + coord_equal()
```



Pie Charts

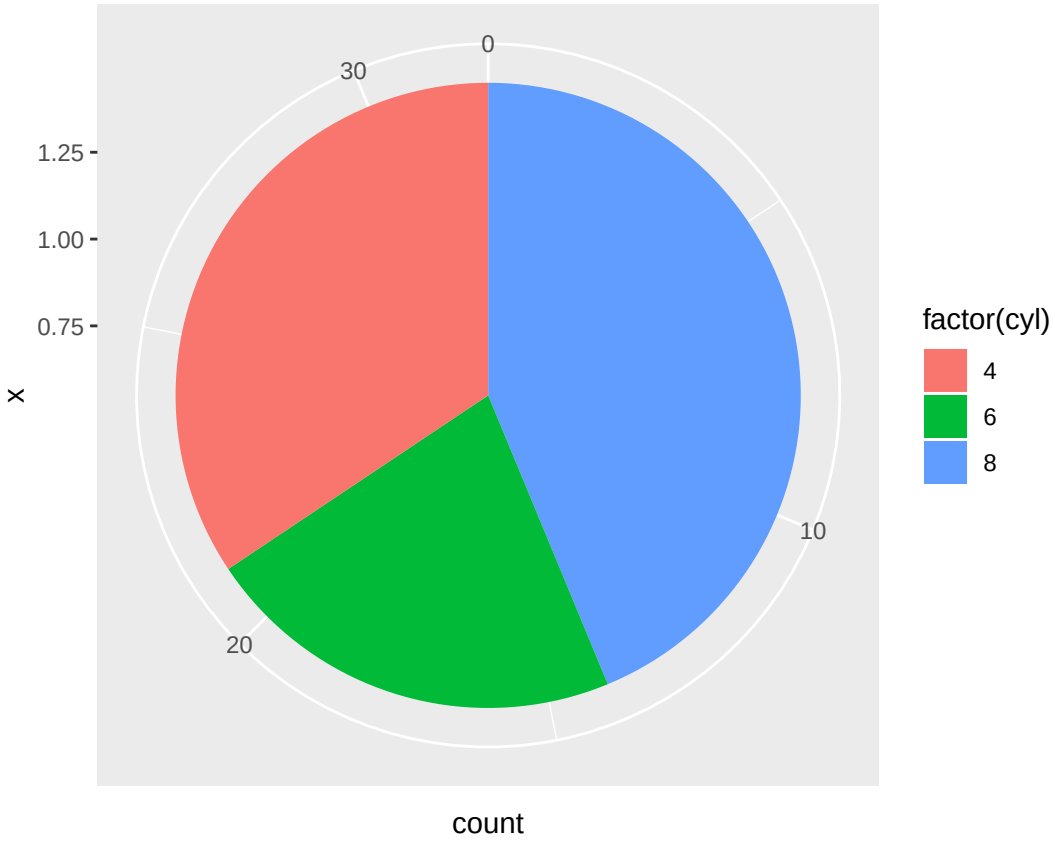
The coord_polar() function converts a planar x-y Cartesian plot to polar coordinates. This can be useful if you are producing pie charts.

We can imagine two forms for pie charts - the typical filled circle, or a colored ring.

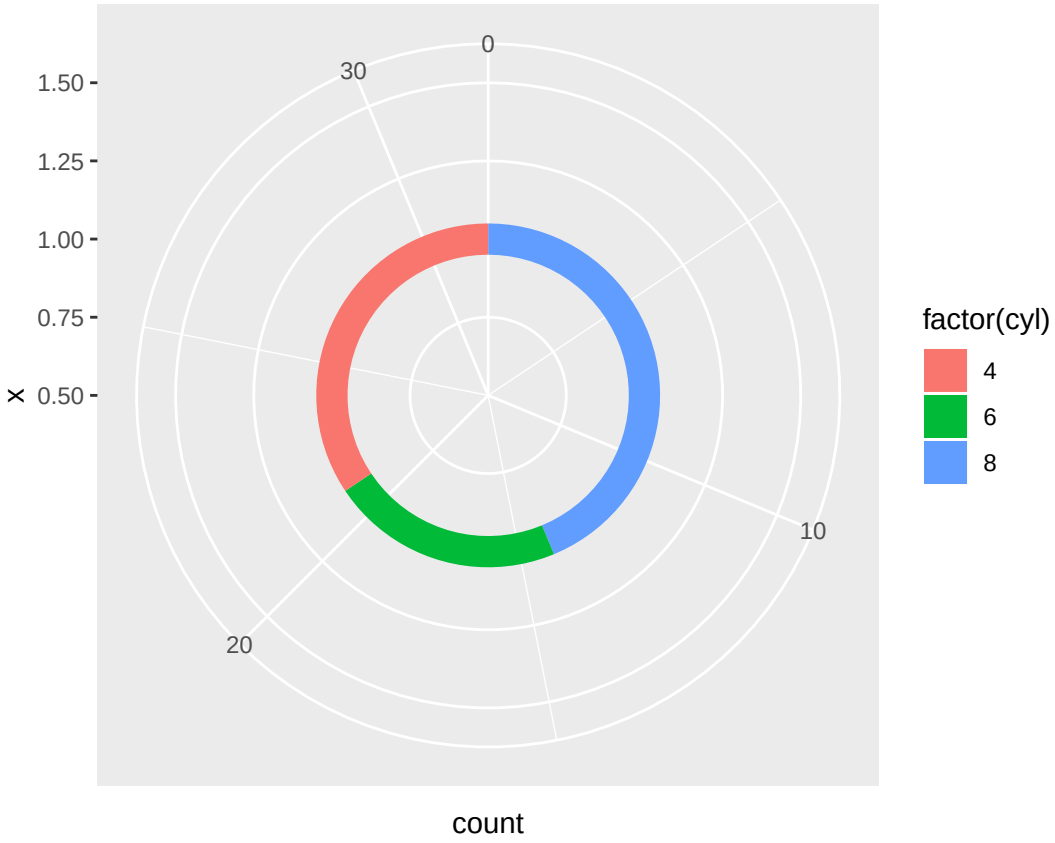
As an example, consider the stacked bar chart shown in the viewer. Imagine that we just take the y axis on the left and bend it until it loops back on itself, while expanding the right side as we go along. We'd end up with a pie chart - it's simply a bar chart transformed onto a polar coordinate system.

Typical pie charts omit all of the non-data ink.

```
# Create a stacked bar plot: wide.bar
wide.bar <- ggplot(mtcars, aes(x = 1, fill = factor(cyl))) +
              geom_bar()

# Convert wide.bar to pie chart
wide.bar +
  coord_polar(theta = "y")
```




```
# Create stacked bar plot: thin.bar
thin.bar <- ggplot(mtcars, aes(x = 1, fill = factor(cyl))) +
              geom_bar(width = 0.1) +
              scale_x_continuous(limits = c(0.5,1.5))

# Convert thin.bar to "ring" type pie chart
thin.bar +
  coord_polar(theta = "y")
```



Facets

1. Facets

- Straight-forward yet useful
- Concept of Small Multiples

Edward Tufte -Visualization of Quantitative Information, 1983

```
library(tidyr)
```

```
##
## Attaching package: 'tidyr'
```

```
## The following object is masked from 'package:reshape2':
##
##     smiths
```

```
library(tidyverse)
```

```
## -- Attaching packages -------- tidyverse 1.2.1 --
```

```
## v tibble  2.1.3     v dplyr   0.8.3
## v readr   1.3.1     v stringr 1.4.0
## v purrr   0.3.2     v forcats 0.4.0

## -- Conflicts ----------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
iris$Flower <- 1:nrow(iris)

iris.wide <- iris %>%
  gather(key, value, -Flower, -Species) %>%
  separate(key, c("Part", "Measure"), "\\.") %>%
  spread(Measure, value)

iris.tidy <- iris %>%
  gather(key, Value, -Flower, -Species) %>%
  separate(key, c("Part", "Measure"), "\\.")

iris.wide2 <- rbind(
   data.frame(Measure = "Length", Part = "Petal", Setosa = iris[iris$Species == "setosa","Petal.Length"]
   data.frame(Measure = "Width", Part = "Petal", Setosa = iris[iris$Species == "setosa","Petal.Width"],
   data.frame(Measure = "Length", Part = "Sepal", Setosa = iris[iris$Species == "setosa","Sepal.Length"]
   data.frame(Measure = "Width", Part = "Sepal", Setosa = iris[iris$Species == "setosa","Sepal.Width"],
```

For iris.wide2, each plot has separate y axis. Three different plot functions

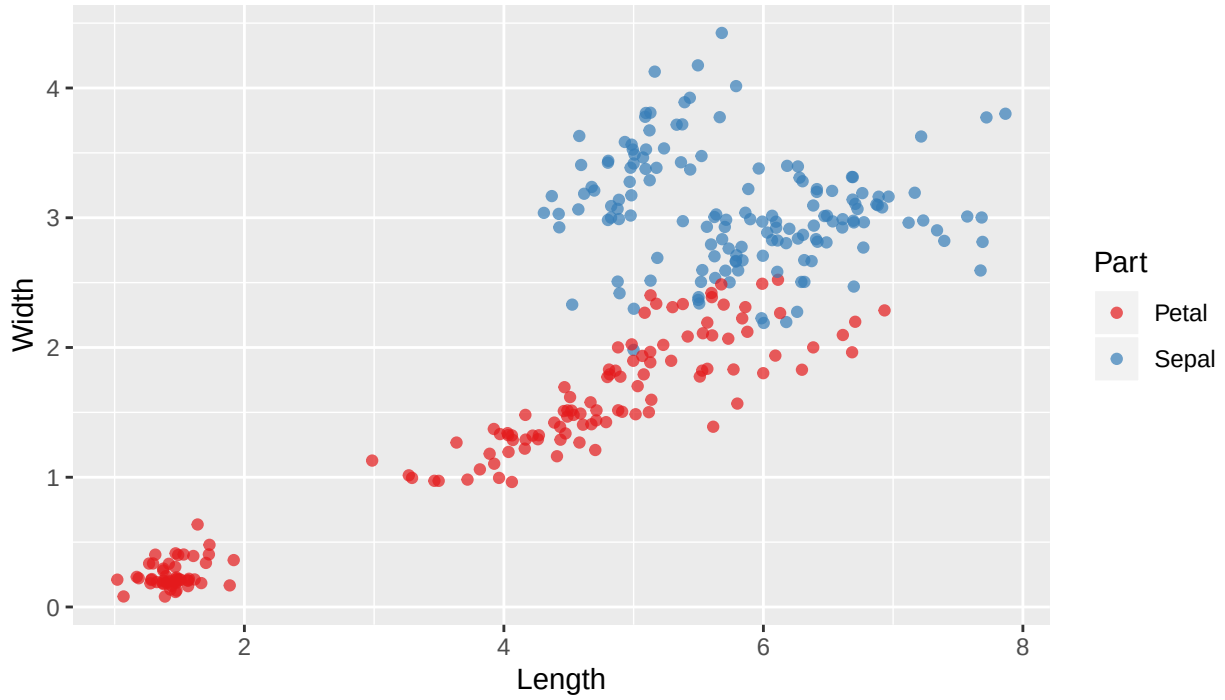
```
head(iris.wide2)
```

```
##   Measure  Part Setosa Versicolor Virginica
## 1  Length Petal    1.4        4.7       6.0
## 2  Length Petal    1.4        4.5       5.1
## 3  Length Petal    1.3        4.9       5.9
## 4  Length Petal    1.5        4.0       5.6
## 5  Length Petal    1.4        4.6       5.8
## 6  Length Petal    1.7        4.5       6.6
```

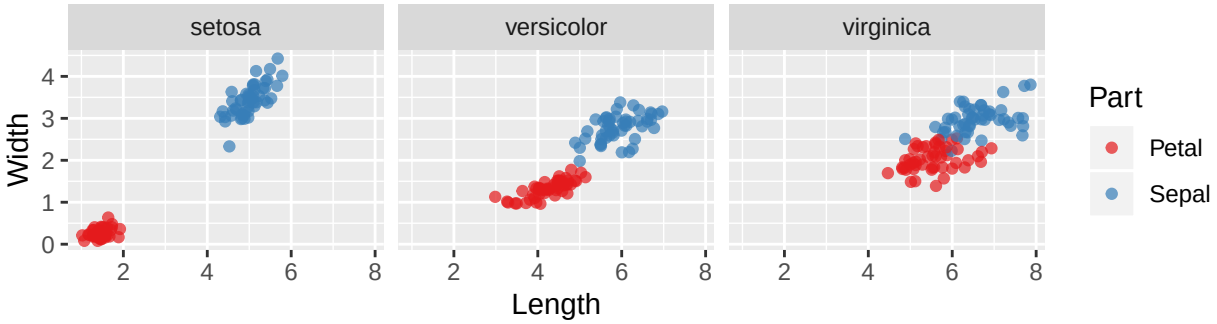
```
str(iris.wide2)
```

```
## 'data.frame':    200 obs. of  5 variables:
##  $ Measure   : Factor w/ 2 levels "Length","Width": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Part      : Factor w/ 2 levels "Petal","Sepal": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Setosa    : num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Versicolor: num  4.7 4.5 4.9 4 4.6 4.5 4.7 3.3 4.6 3.9 ...
##  $ Virginica : num  6 5.1 5.9 5.6 5.8 6.6 4.5 6.3 5.8 6.1 ...
```

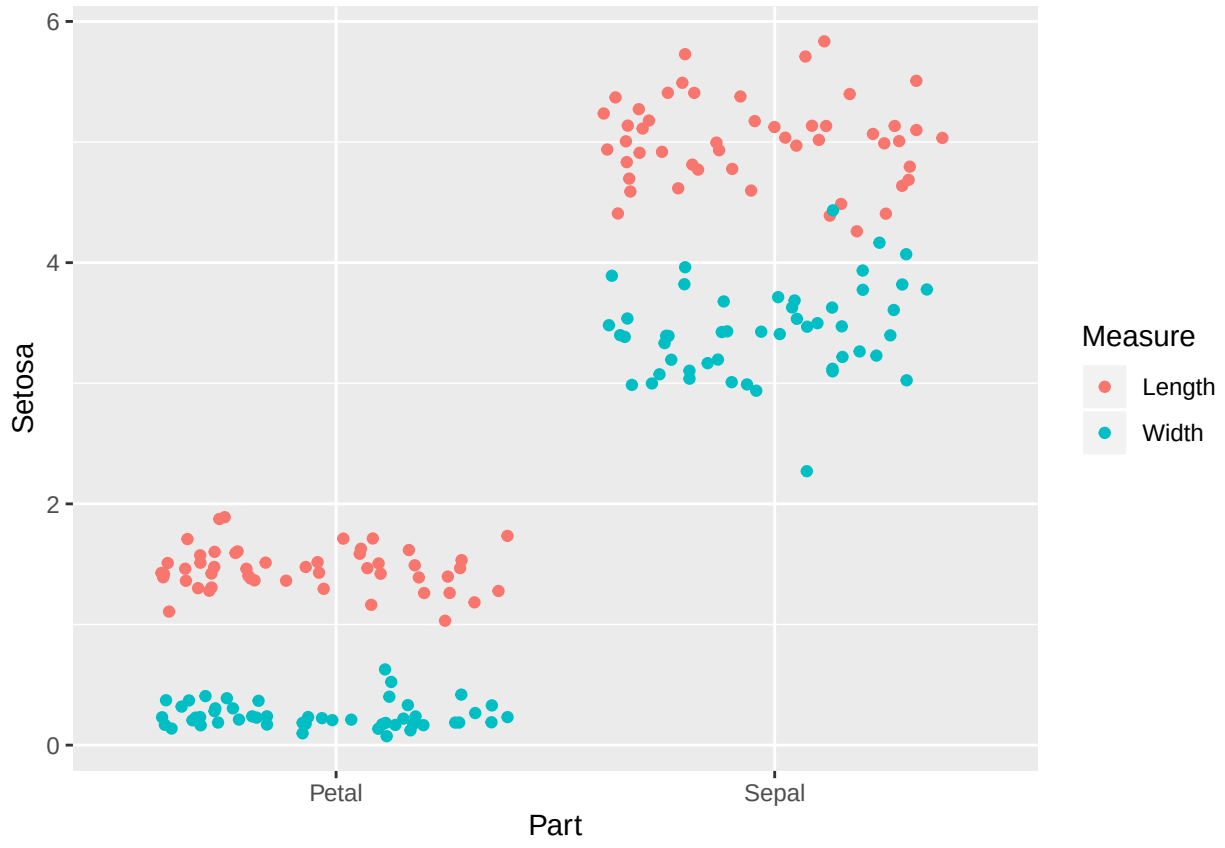
```
# iris.wide
p <- ggplot(iris.wide, aes(x = Length, y = Width, col = Part)) +
 geom_point(position = position_jitter(), alpha = 0.7) +
 scale_color_brewer(palette = "Set1") +
 coord_fixed()
p
```



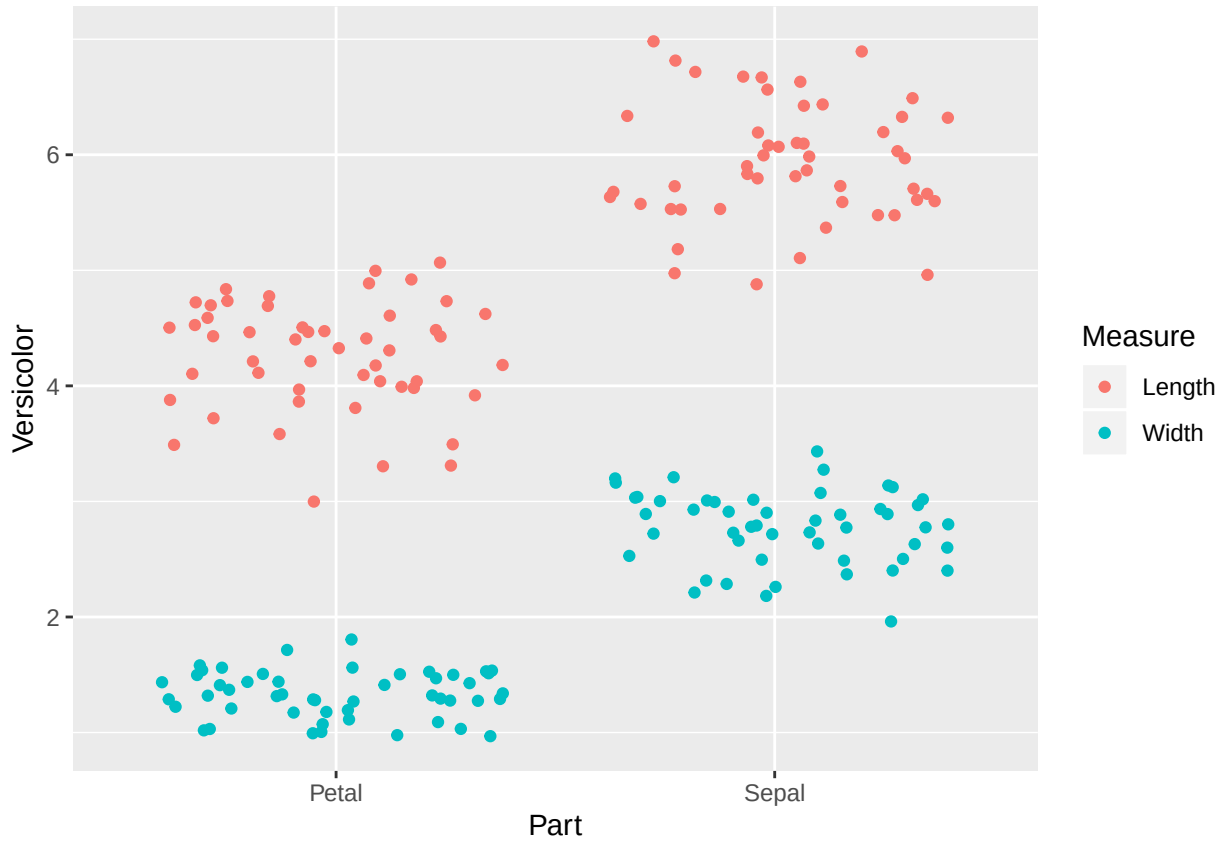

```
p + facet_grid(. ~ Species) # row ~ column
```



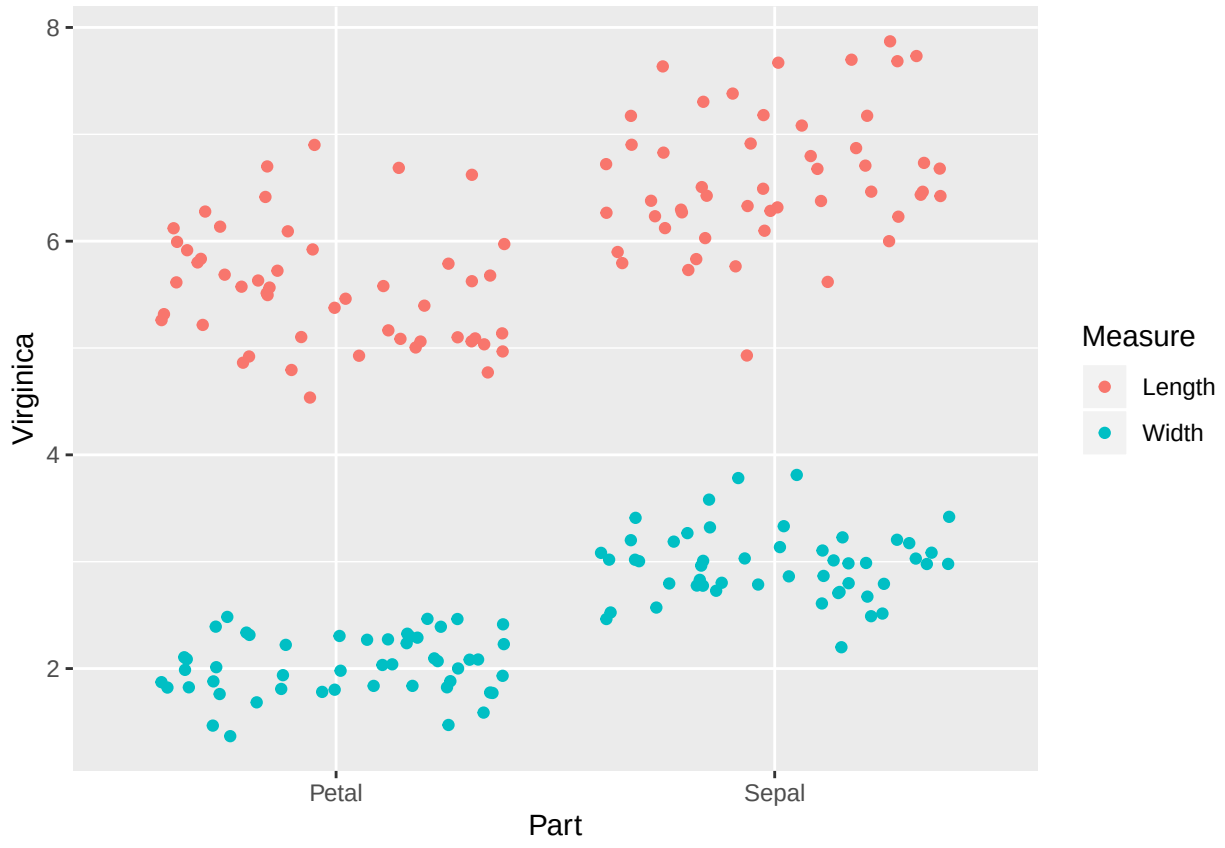

```
# iris.wide2
ggplot(iris.wide2, aes(x = Part, y = Setosa, col = Measure)) +
  geom_jitter()
```



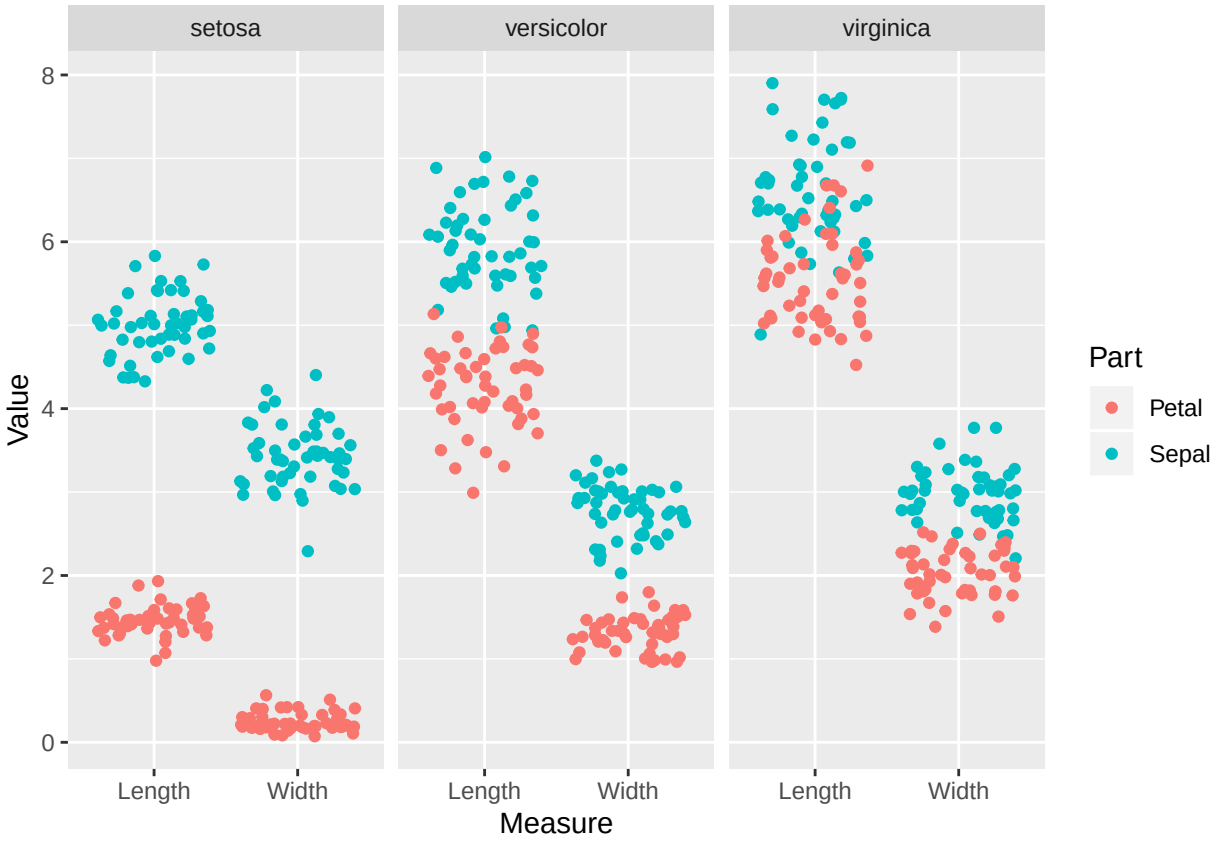

```
ggplot(iris.wide2, aes(x = Part, y = Versicolor, col = Measure)) +
  geom_jitter()
```



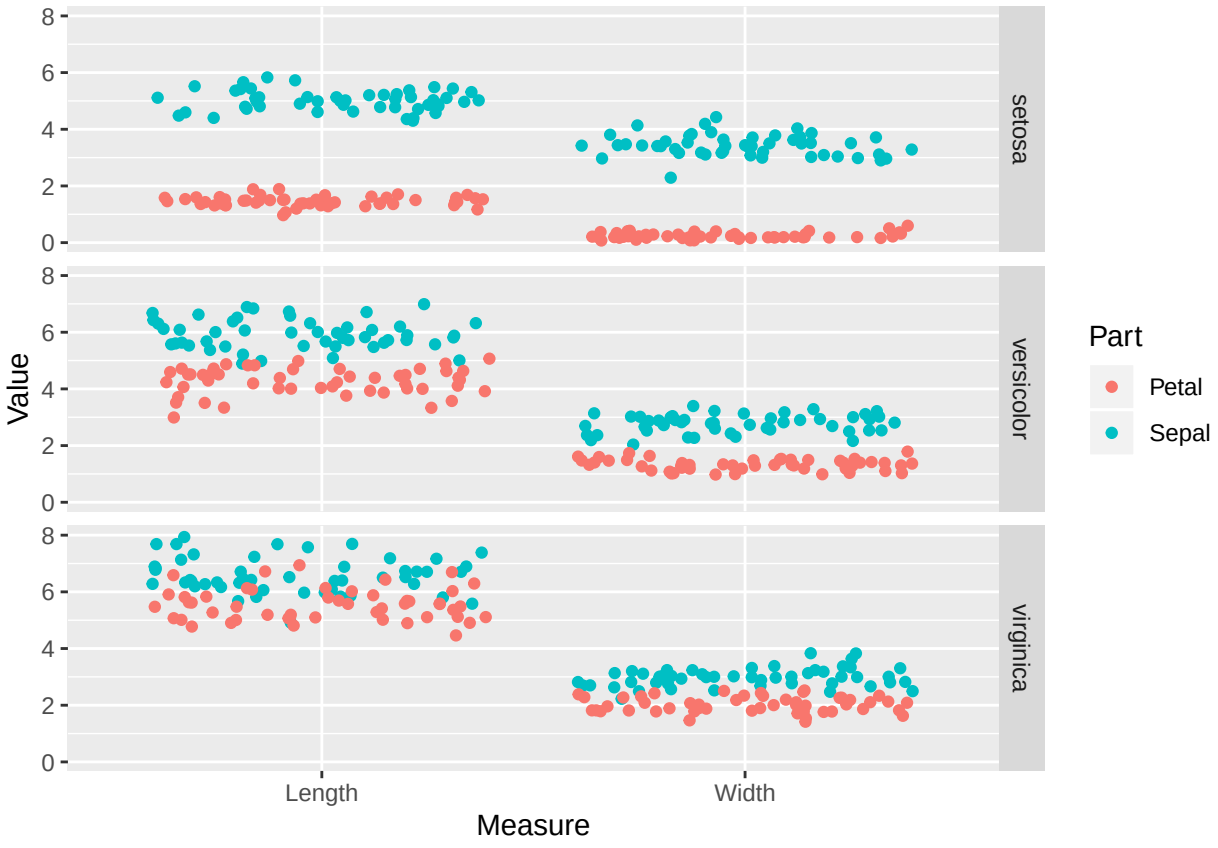

```
ggplot(iris.wide2, aes(x = Part, y = Virginica, col = Measure)) +
  geom_jitter()
```




```
# iris.tidy
ggplot(iris.tidy, aes(x = Measure, y = Value, col = Part)) +
  geom_jitter() +
  facet_grid(. ~ Species)
```

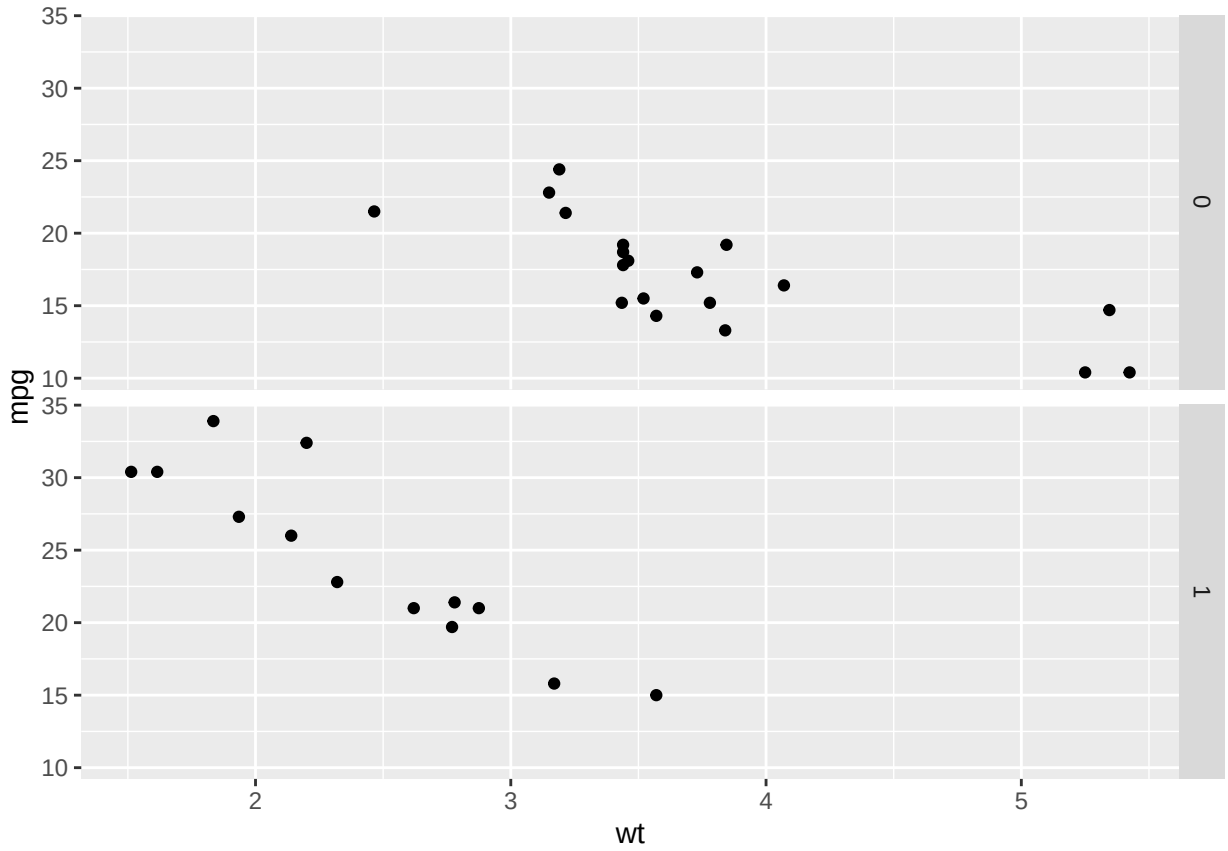



```
# iris.tidy - wrong
ggplot(iris.tidy, aes(x = Measure, y = Value, col = Part)) +
  geom_jitter() +
  facet_grid(Species ~ .)
```

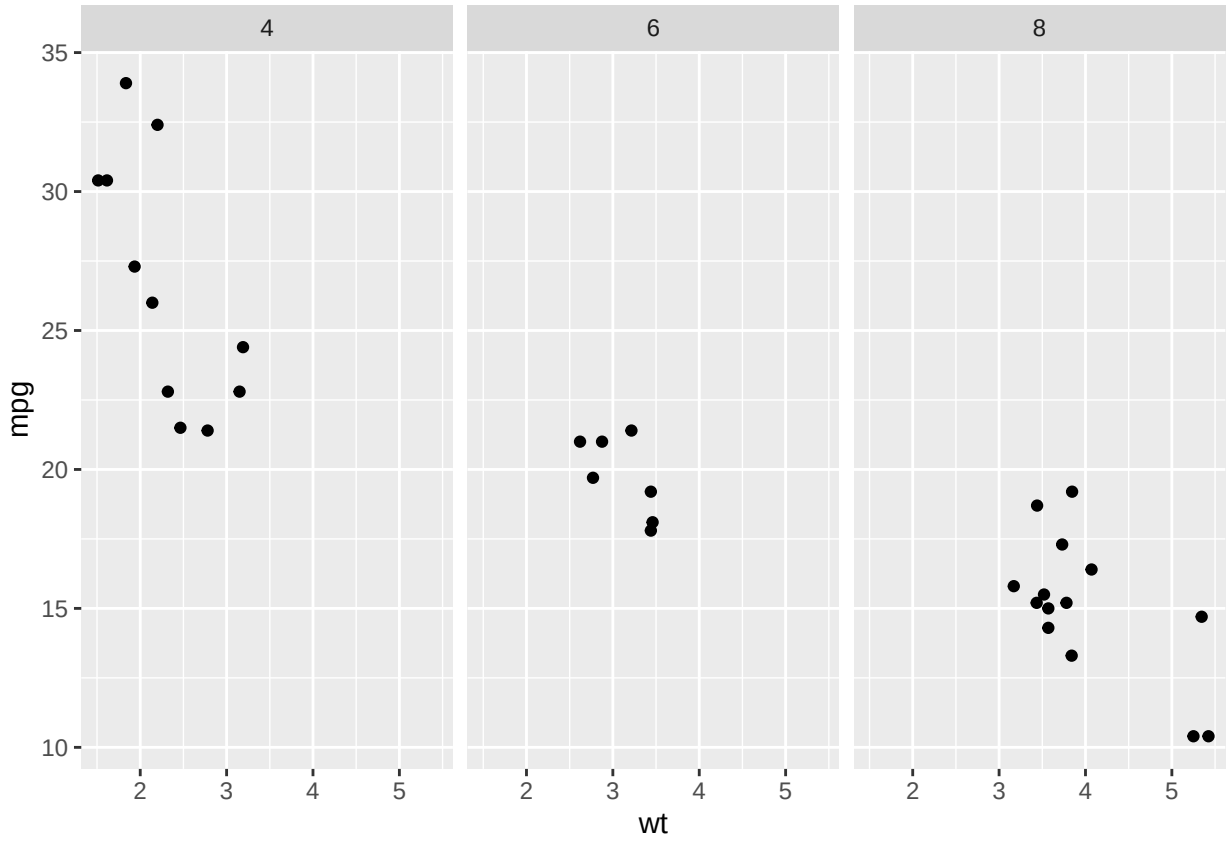


Practice

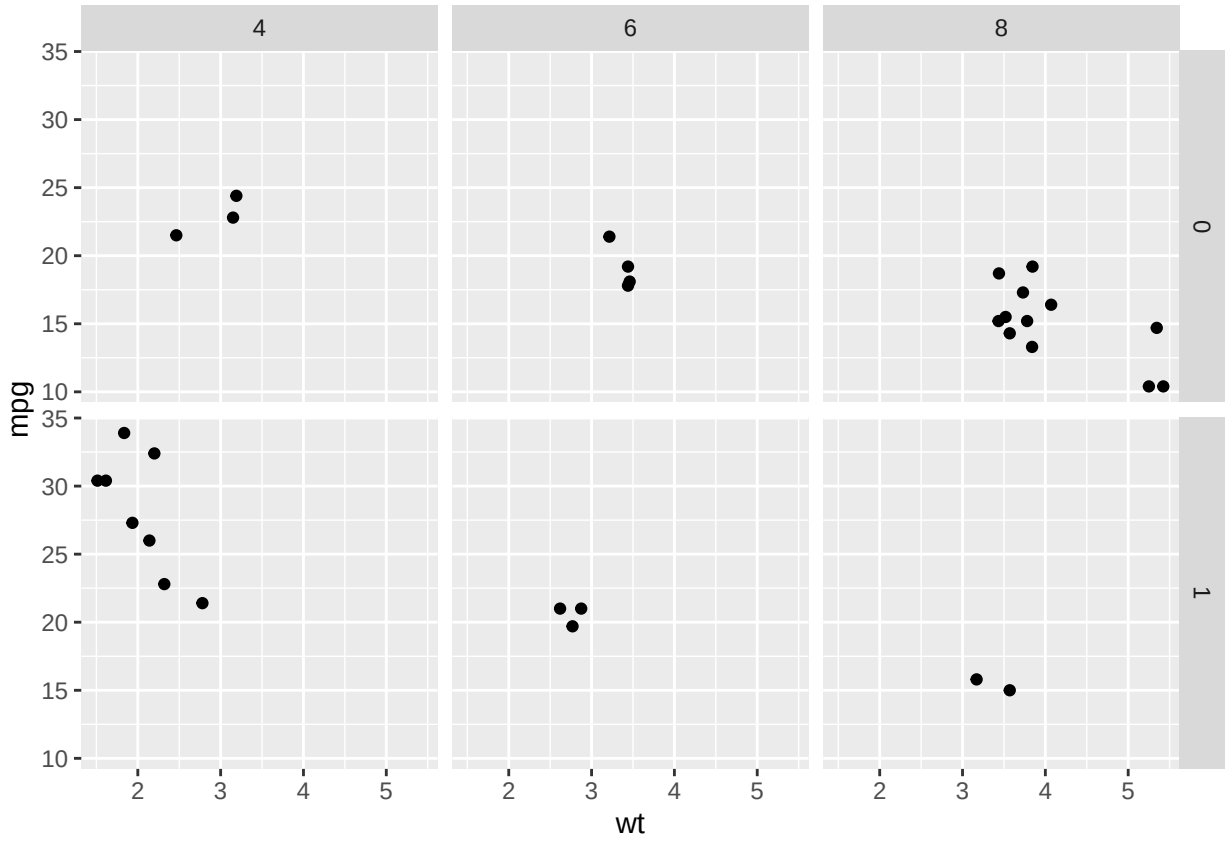
```
# Basic scatter plot
p <- ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point()

# 1 - Separate rows according to transmission type, am
p +
  facet_grid(am ~ .)
```




```
# 2 - Separate columns according to cylinders, cyl
p +
  facet_grid(. ~ cyl)
```

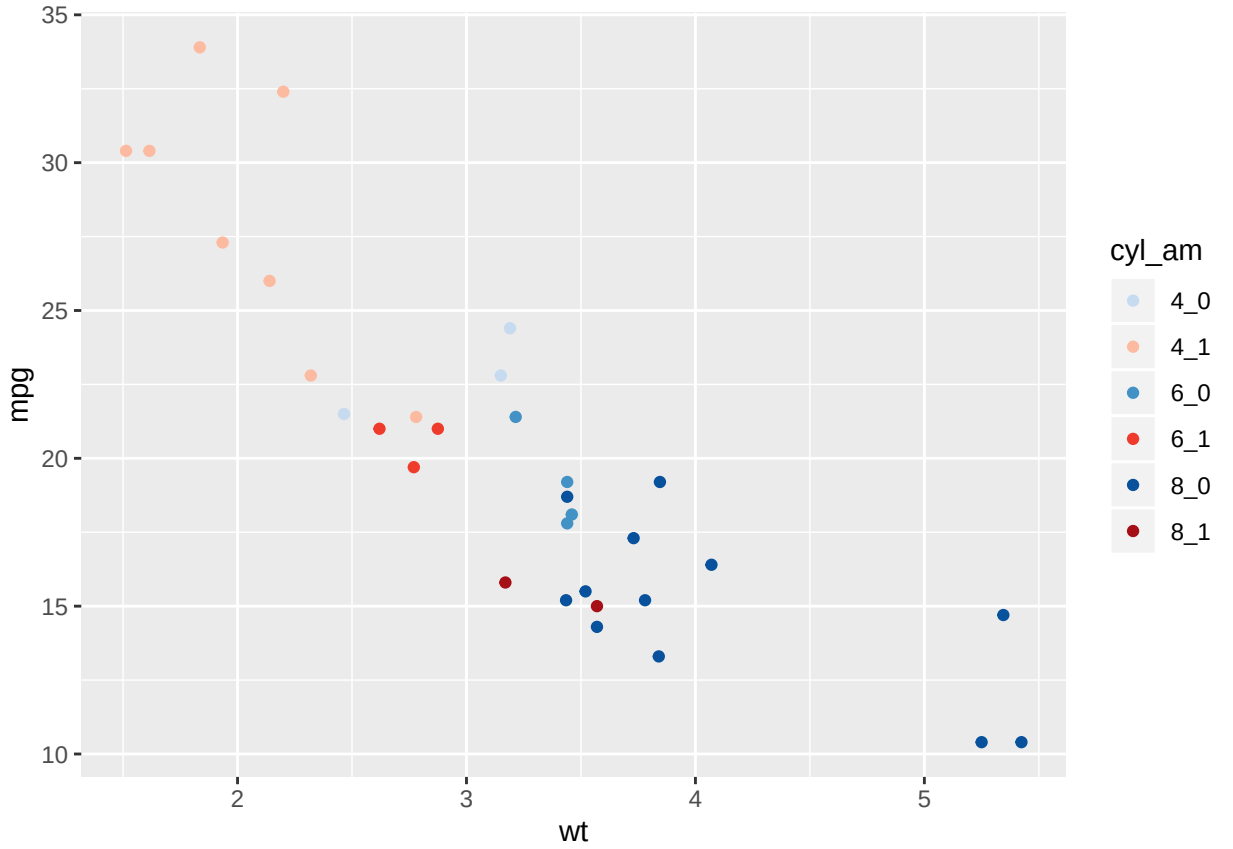



```
# 3 - Separate by both columns and rows
p +
  facet_grid(am ~ cyl)
```

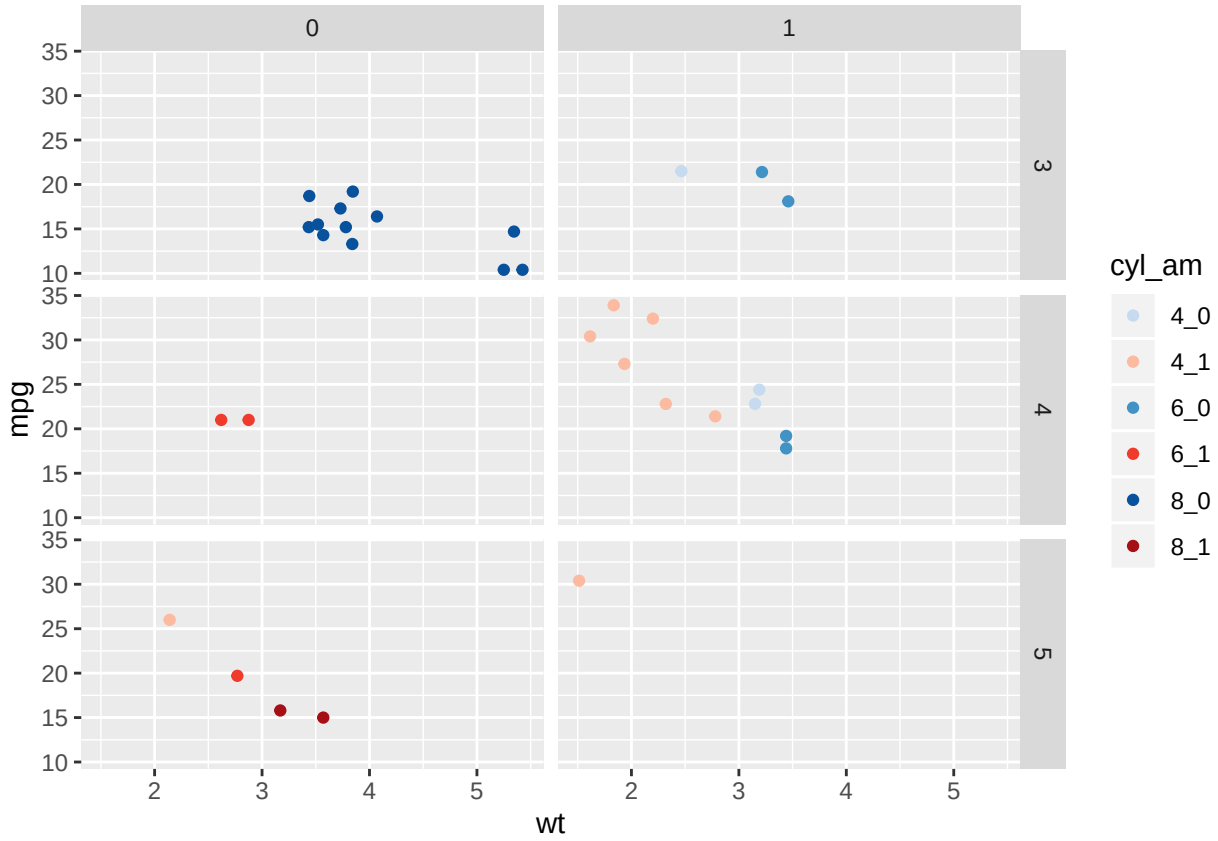


Many variables

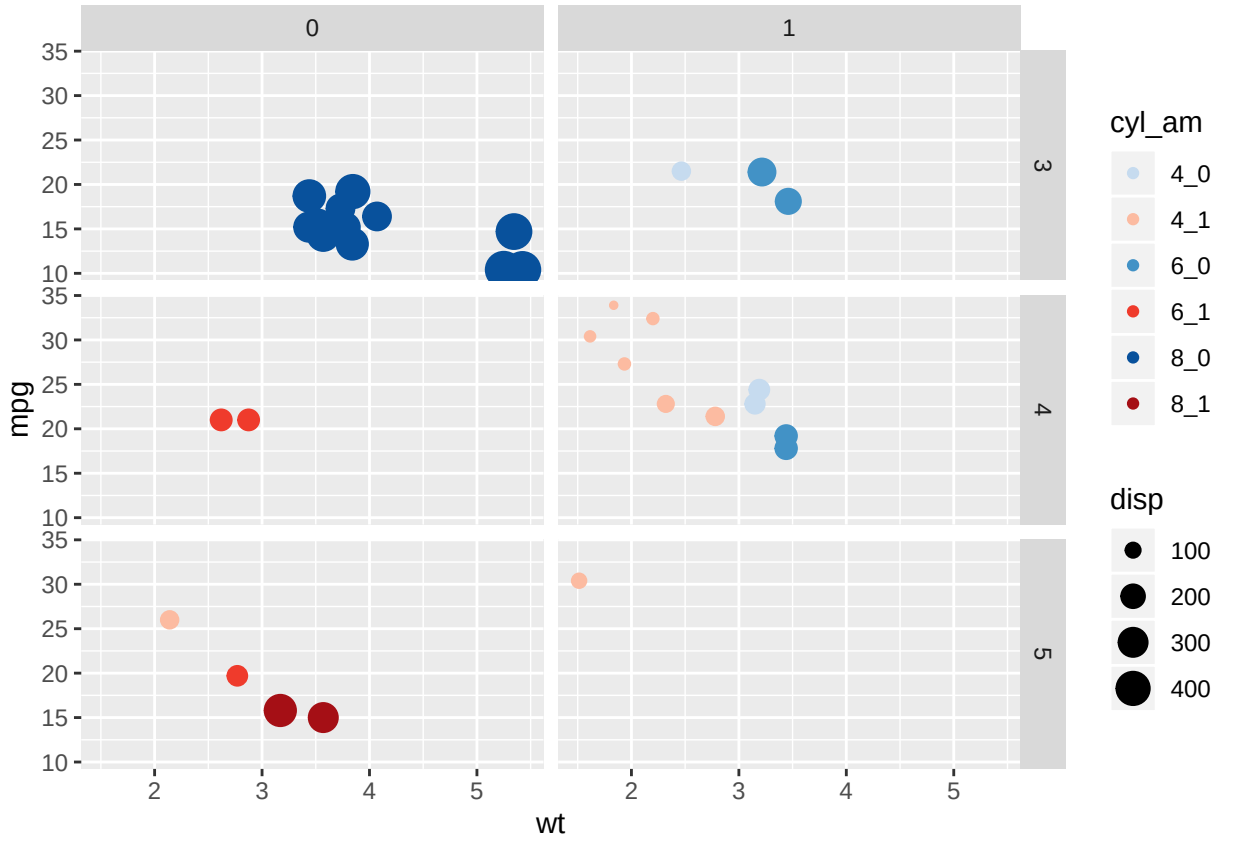
```
library("RColorBrewer")

# Code to create the cyl_am col and myCol vector
mtcars$cyl_am <- paste(mtcars$cyl, mtcars$am, sep = "_")
myCol <- rbind(brewer.pal(9, "Blues")[c(3,6,8)],
               brewer.pal(9, "Reds")[c(3,6,8)])

# Map cyl_am onto col
ggplot(mtcars, aes(x = wt, y = mpg, col = cyl_am)) +
  geom_point() +
  # Add a manual colour scale
  scale_color_manual(values = myCol)
```




```
# Grid facet on gear vs. vs
ggplot(mtcars, aes(x = wt, y = mpg, col = cyl_am)) +
  geom_point() +
  # Add a manual colour scale
  scale_color_manual(values = myCol) + facet_grid(gear ~ vs)
```



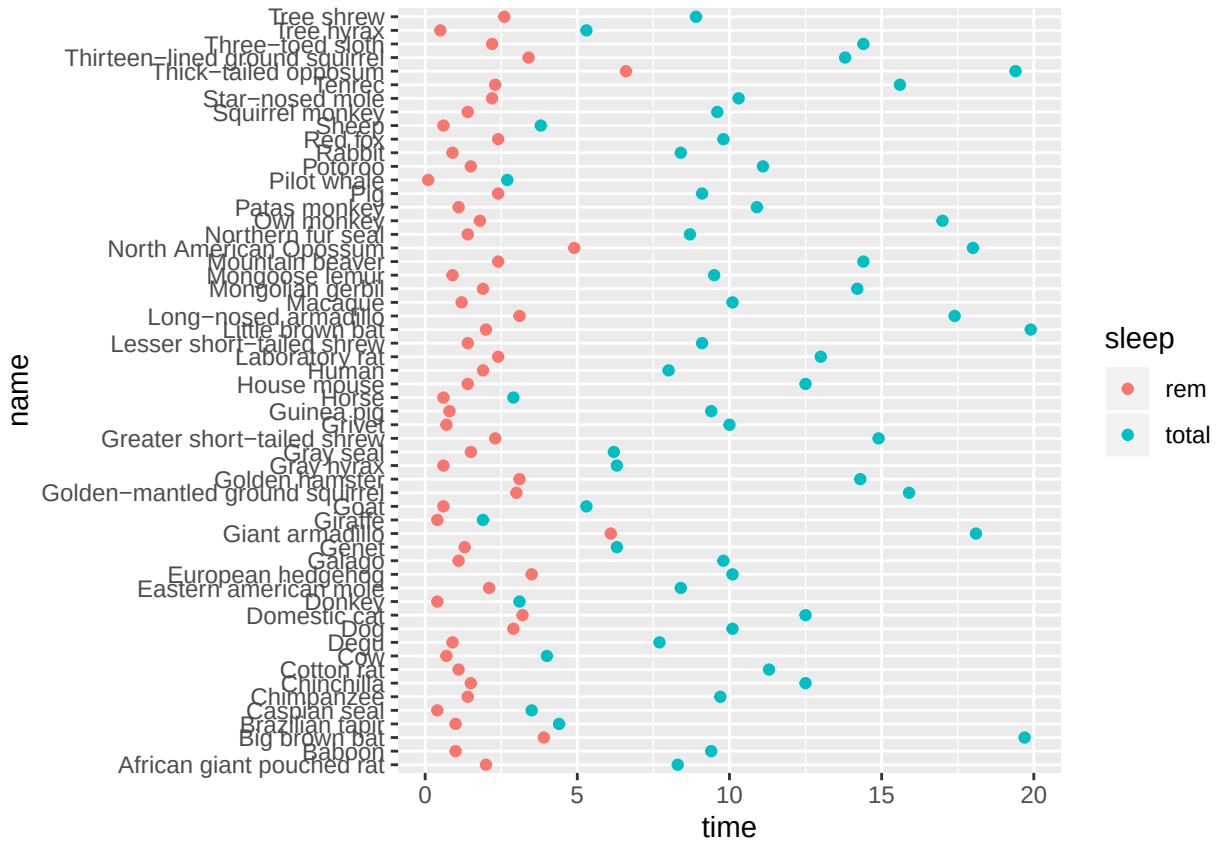

```
# Also map disp to size
ggplot(mtcars, aes(x = wt, y = mpg, col = cyl_am, size = disp)) +
  geom_point() +
  # Add a manual colour scale
  scale_color_manual(values = myCol) + facet_grid(gear ~ vs)
```



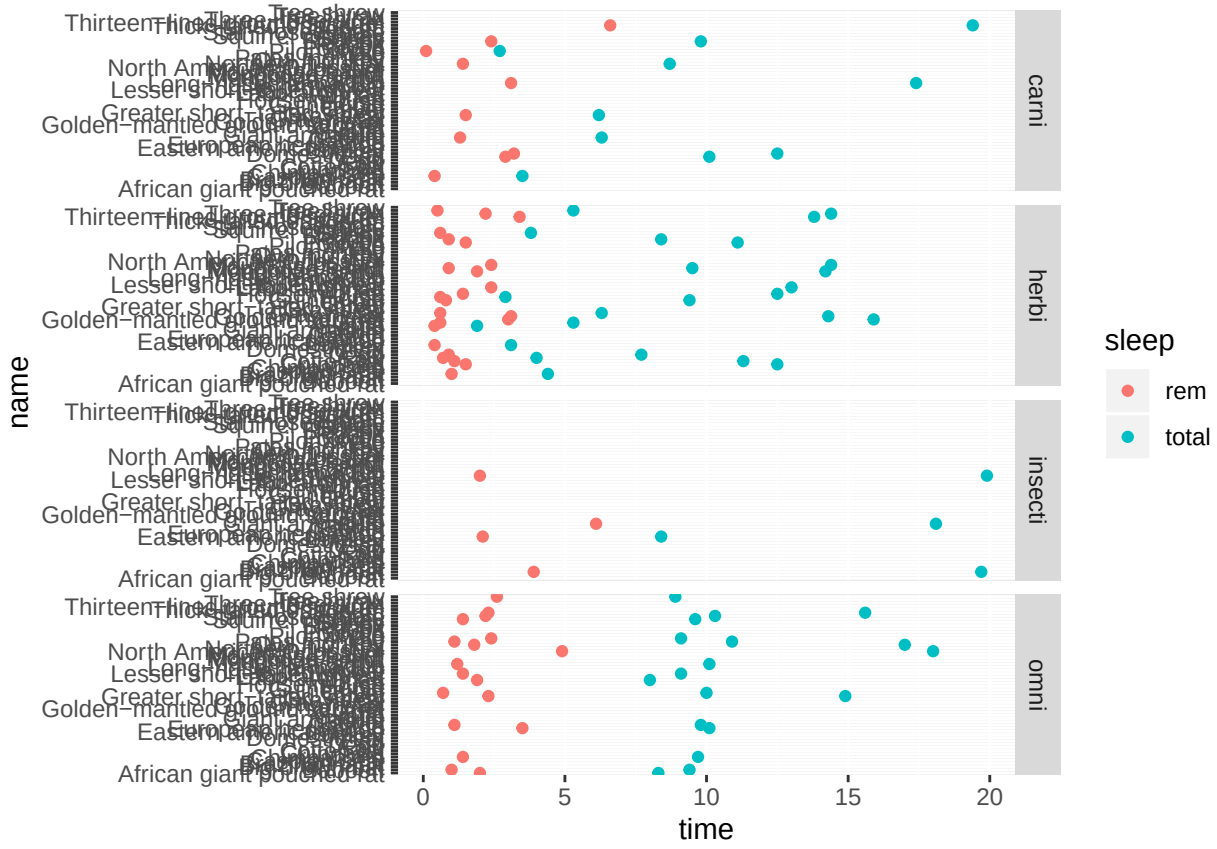
Dropping levels

Extend facet_grid with scale = “free_y” and space = “free_y” to leave out rows for which there is no data.

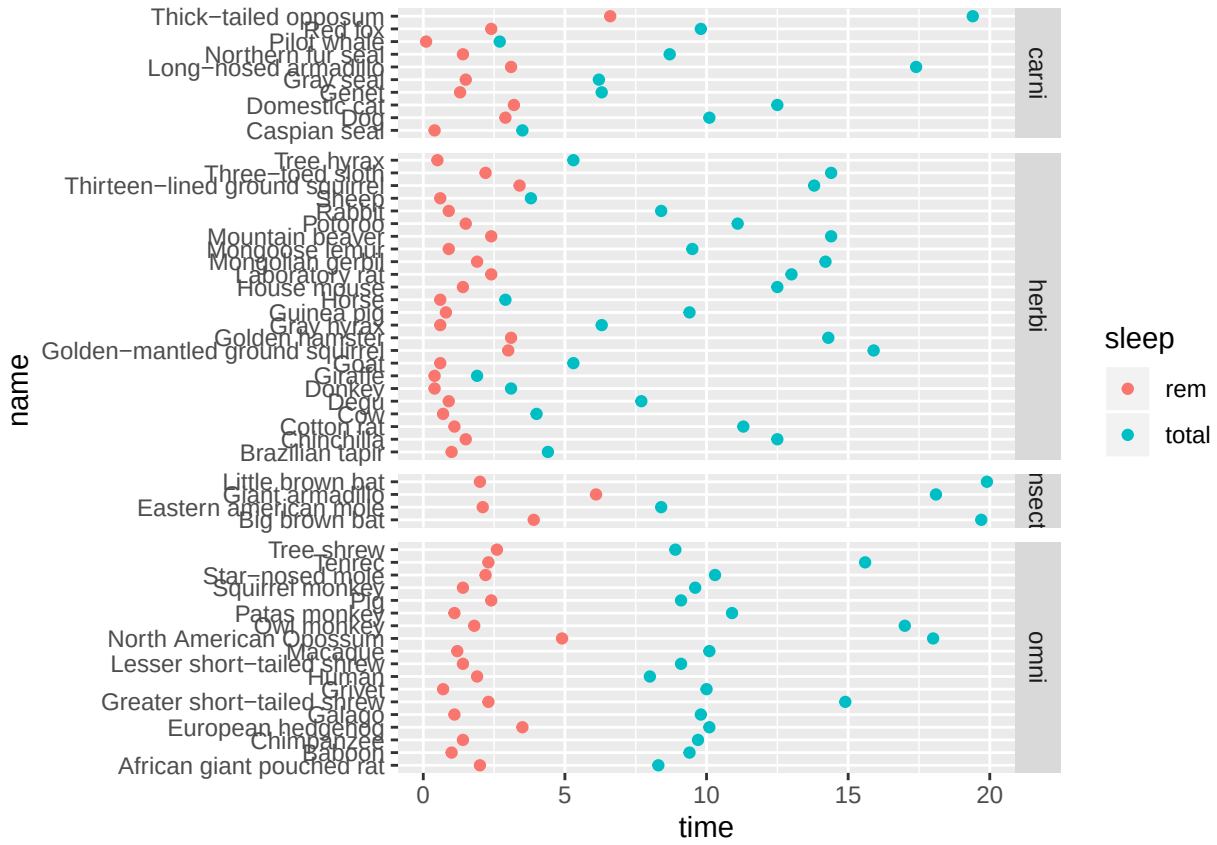
```
data(msleep)
mamsleep_clean <- na.omit(msleep[,c("vore", "name", "sleep_total", "sleep_rem")])
mamsleep <- mamsleep_clean %>% gather(sleep, time, -vore, -name) %>%
  separate(sleep, c("nouse","sleep"), "\\_")
mamsleep$nouse <- NULL

# Basic scatter plot
p <- ggplot(mamsleep, aes(x = time, y = name, col = sleep)) +
  geom_point()

# Execute to display plot
p
```




```
# Facet rows accoding to vore
p +
  facet_grid(vore ~ .)
```




```
# Specify scale and space arguments to free up rows
p +
  facet_grid(vore ~ ., scale= "free_y", space = "free_y")
```


Thick−tailed opposum
Red fox
Pilot whale
Northern fur seal
Long−nosed armadillo
Gray seal
Genet
Domestic cat
Dog
Caspian seal
Tree hyrax
Three−toed sloth
Thirteen−lined ground squirrel
Sheep
Rabbit
Proboci
Mountain beaver
Mongoose lemur
Mongolian gerbil
Laboratory rat
House mouse
Horse
Guinea pig
Gray hyrax
Golden hamster
Golden−mantled ground squirrel
Goat
Giraffe
Donkey
Degu
Cow
Cotton rat
Chinchilla
Brazilian tapir
Little brown bat
Giant armadillo
Eastern american mole
Big brown bat
Tree shrew
Tenrec
Star−nosed mole
Squirrel monkey
Pig
Patas monkey
Owl monkey
North American Opossum
Macaque
Lesser short−tailed shrew
Human
Grivet
Greater short−tailed shrew
Galago
European hedgehog
Chimpanzee
Baboon
African giant pouched rat
carni
herbi
nsect
omni
name
0
5
10
15
20
time
sleep
rem
total