

# Introduction to ggplot2

# The concept behind ggplot2

Traditional graphics treats each type of plot separately

- Many different functions, each with distinct syntax and arguments

- Hard to learn and remember; hard to read / update code later

- Hard to experiment with different plot types



ggplot2 is built around the idea of a “grammar of graphics” (Wilkinson 1999)

- Core concept: define plots based on semantic components

Important advantages:

- Uses consistent syntax and arguments for all plot types: easier to learn / code

- Allows combination of plot types through layering, multiple plots through faceting

- Provides customizable themes: consistency is easier for audiences / readers to follow

# ggplot2 basic syntax

*Geometries*  
*Aesthetics*  
*Data*



building “up”  
metaphor

# Data visualizations can be generalized as consisting of three parts

Data

+

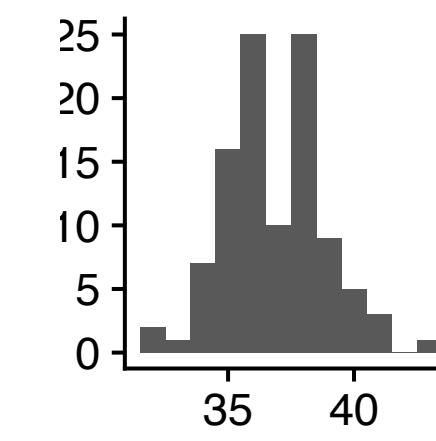
Geometric  
Mapping

+

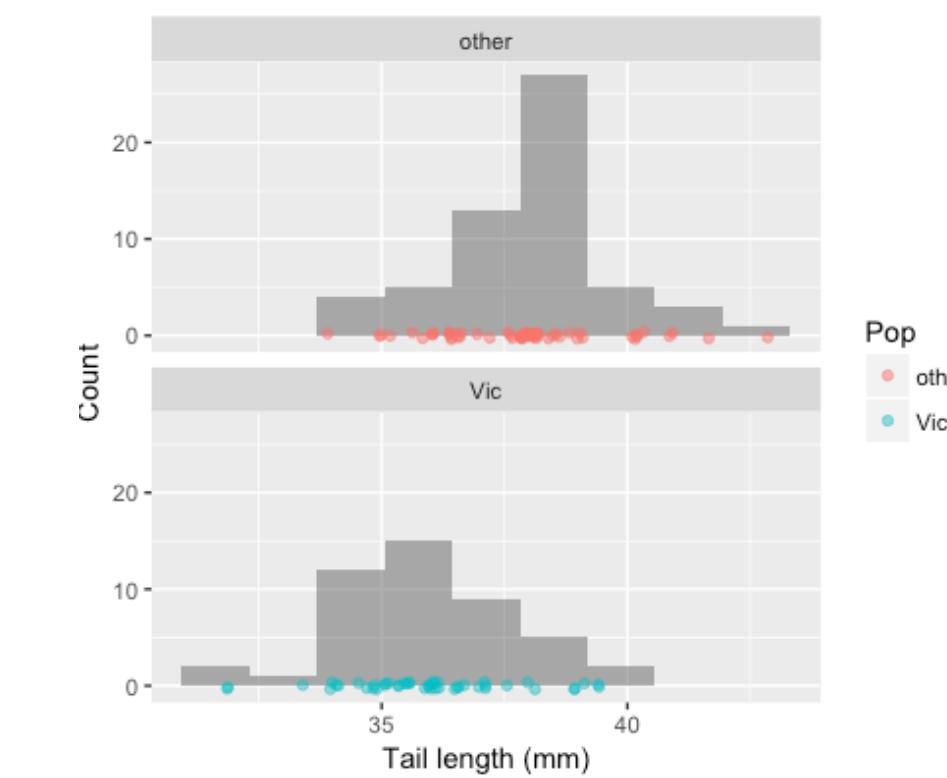
Aesthetic  
Properties of the  
Geometric Mapping

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1      5.1        3.5       1.4        0.2   setosa
## 2      4.9        3.0       1.4        0.2   setosa
## 3      4.7        3.2       1.3        0.2   setosa
## 4      4.6        3.1       1.5        0.2   setosa
## 5      5.0        3.6       1.4        0.2   setosa
## 6      5.4        3.9       1.7        0.4   setosa
```

+



+



```
ggplot(data = <DATA>) + <GEOM_FUNCTION>(mapping = aes(<MAPPINGS>))
```

# Example 1: Creating a histogram

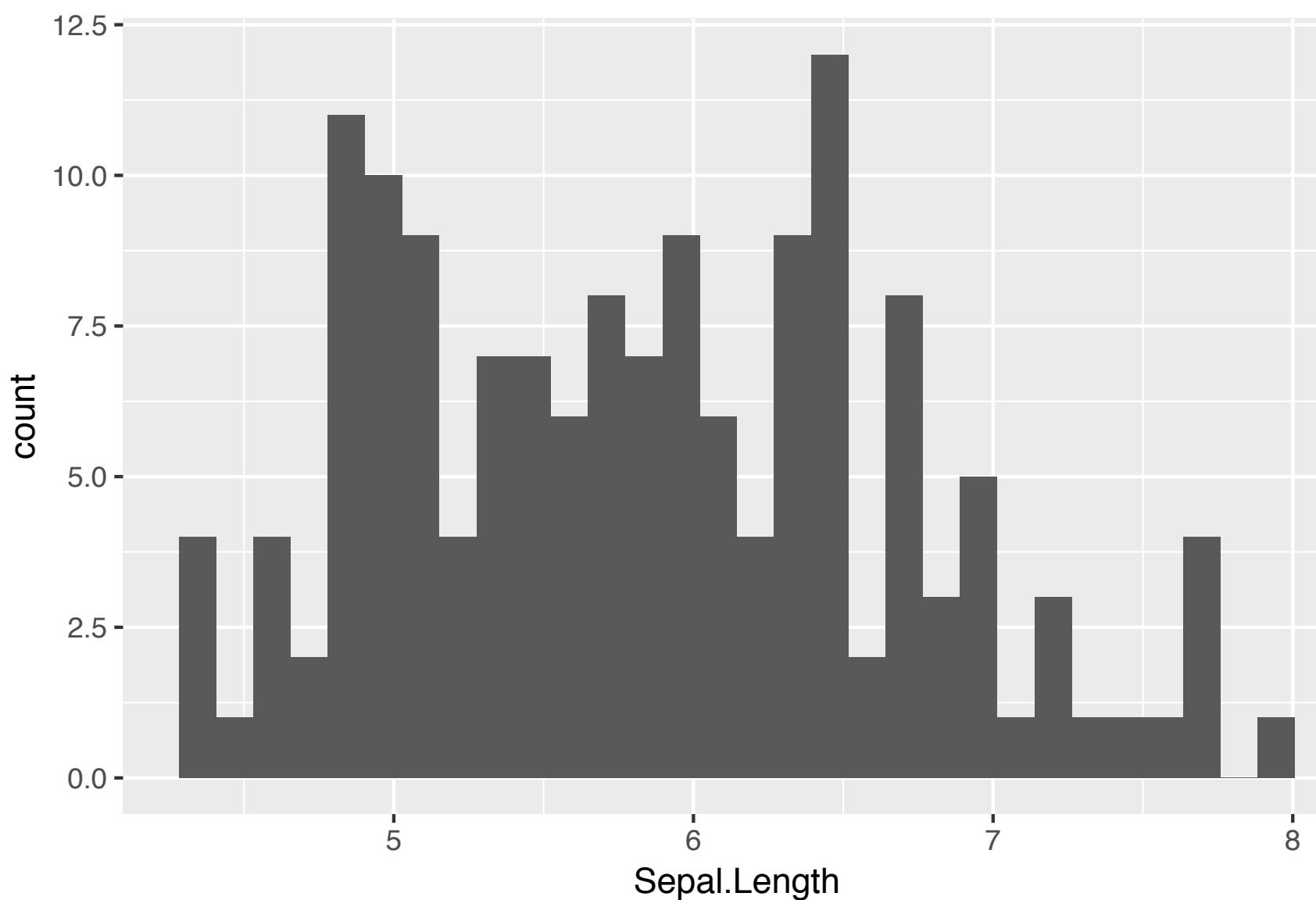
Data

+

Geometric  
Mapping

Aesthetic  
Properties of the  
Geometric Mapping

```
ggplot(data = iris) + geom_histogram(mapping = aes(x = Sepal.Length))
```



## Example 2: Creating a Boxplot

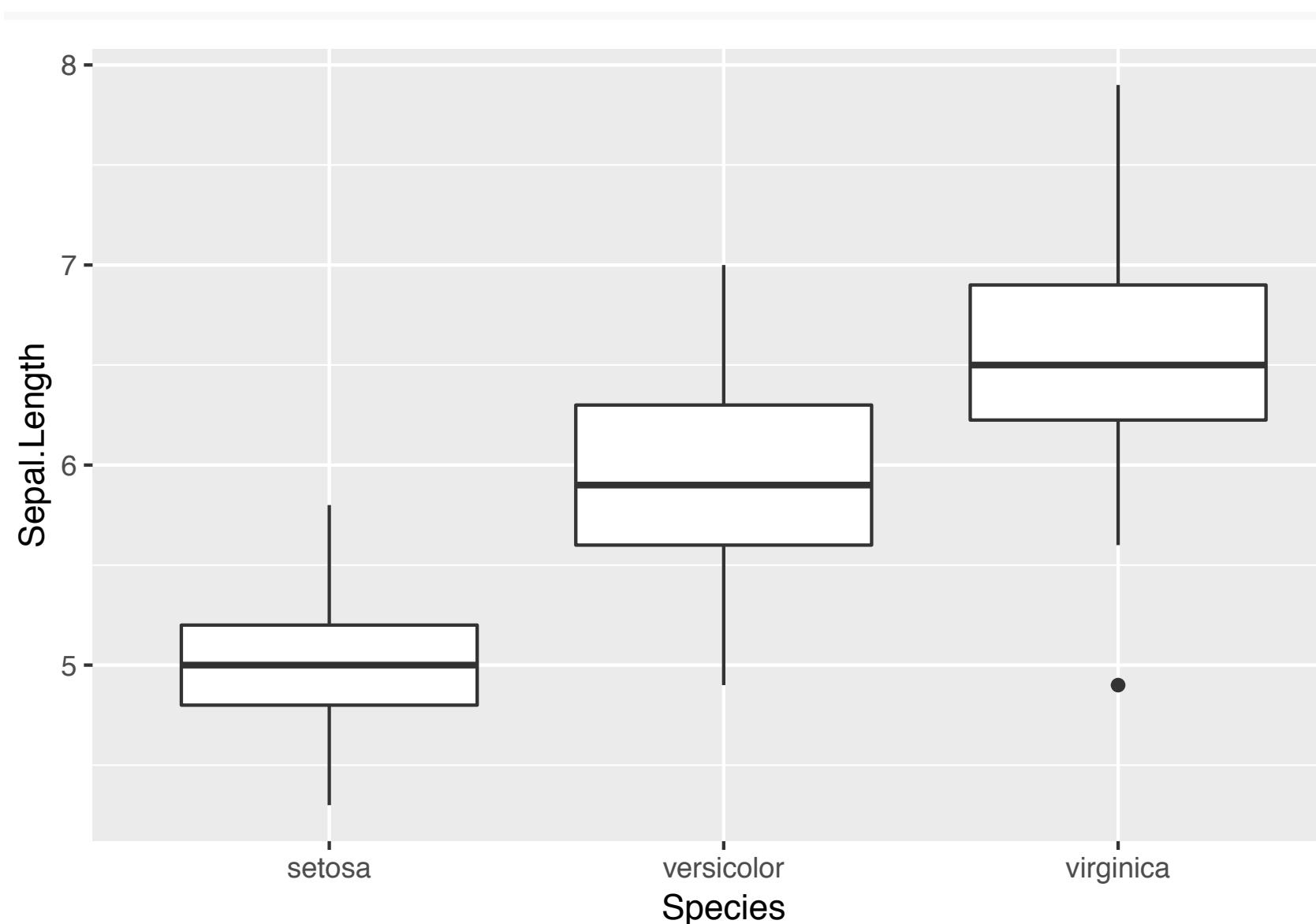
Data

+

Geometric  
Mapping

Aesthetic  
Properties of the  
Geometric Mapping

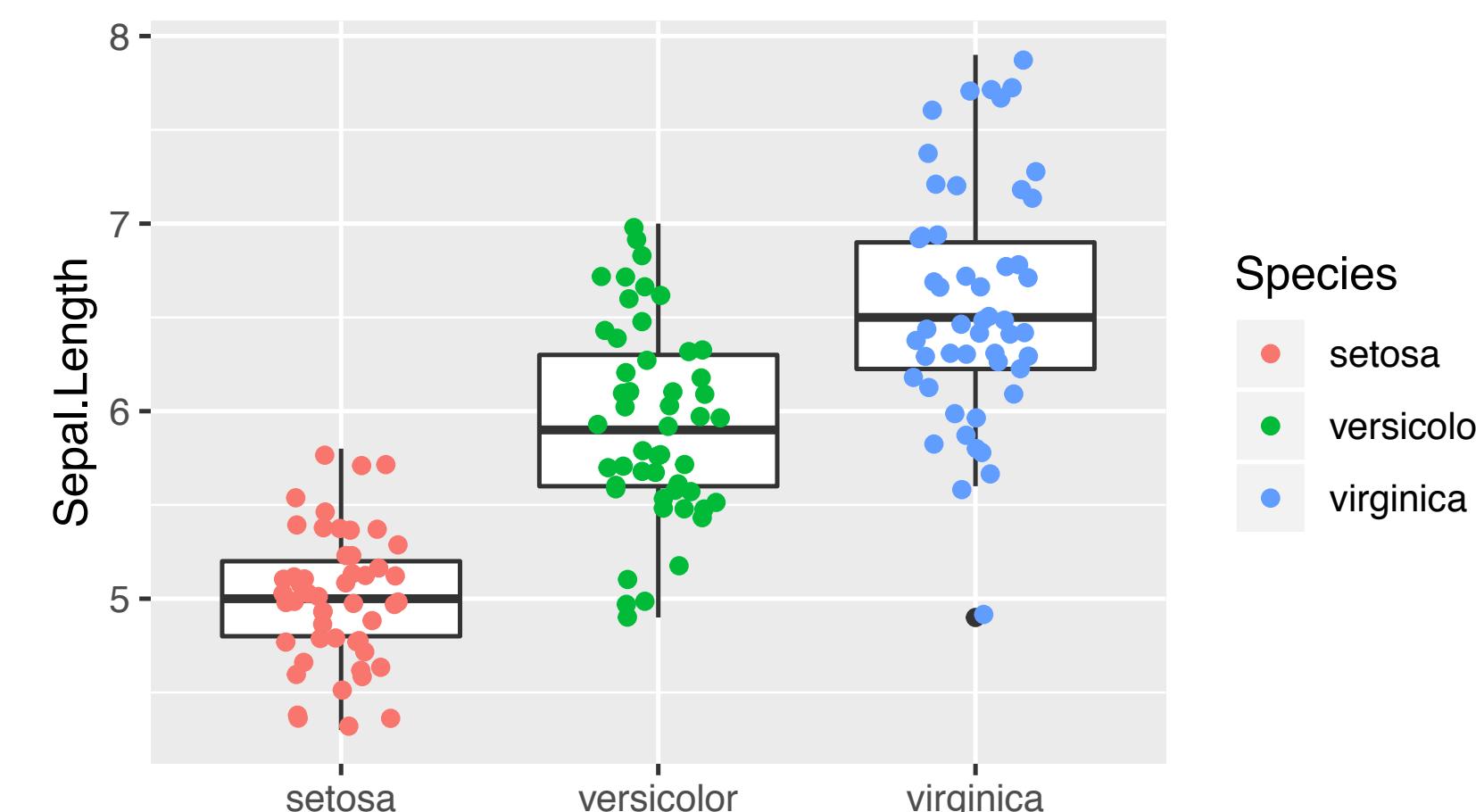
```
ggplot(data = iris) + geom_boxplot(mapping = aes(x = Species,  
y = Sepal.Length))
```



# Example 3: Combining Geometric Representations

Data + Geometric Mapping + Aesthetic Properties of the Geometric Mapping

```
ggplot(data = iris) + geom_boxplot(mapping = aes(x = Species,  
y = Sepal.Length)) +  
  geom_jitter(mapping = aes(x = Species,  
y = Sepal.Length,  
color = Species),  
width = 0.2)
```



# Shared aesthetics properties can be specified in `ggplot()` or in an independent `aes()` call

```
ggplot(data = iris, aes(x = Species, y = Sepal.Length)) +  
  geom_boxplot() +  
  geom_jitter(mapping = aes(color = Species),  
              width = 0.2)
```

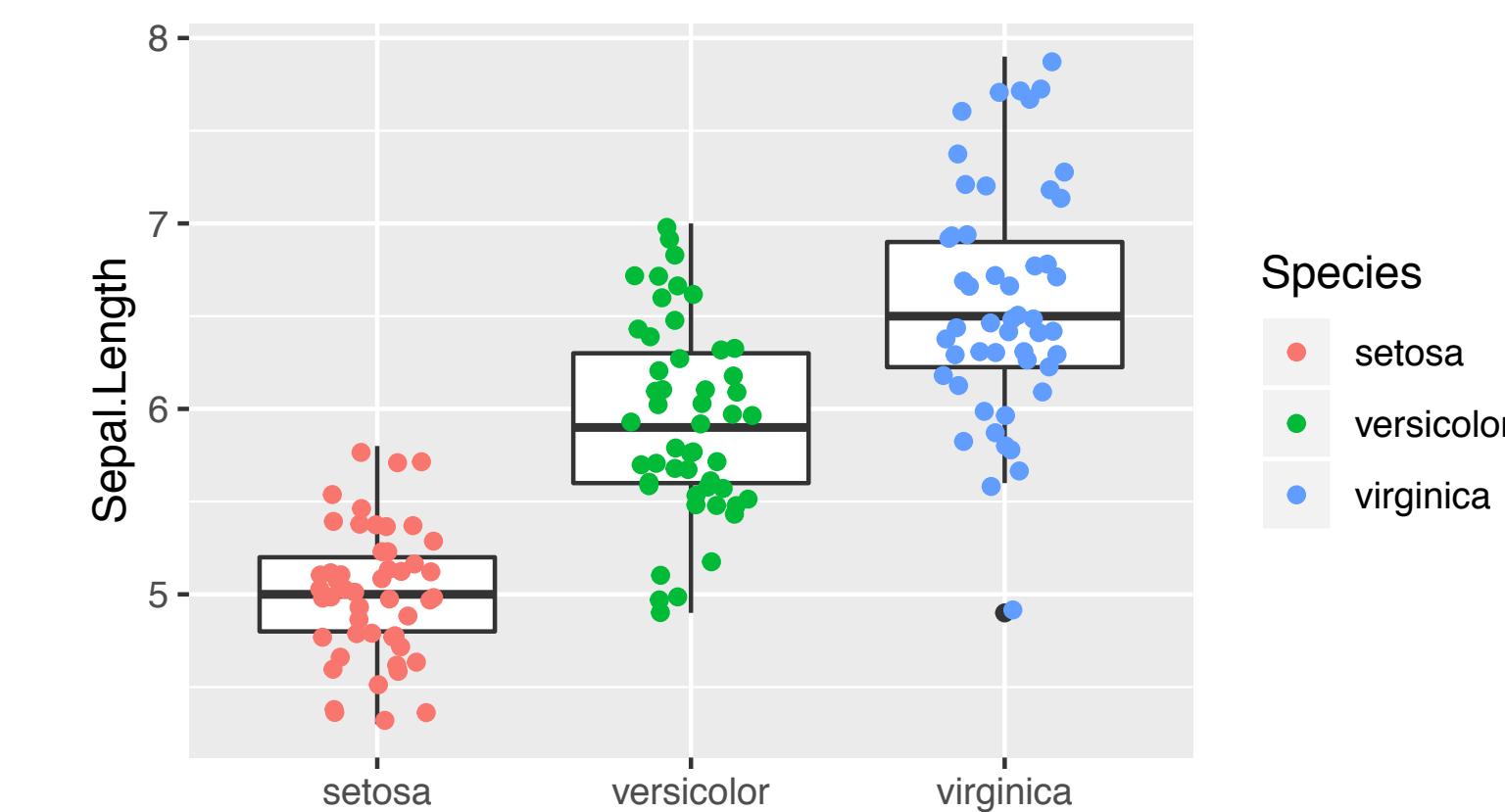
or

```
ggplot(data = iris) +  
  aes(x = Species, y = Sepal.Length) +  
  geom_boxplot() +  
  geom_jitter(mapping = aes(color = Species),  
              width = 0.2)
```

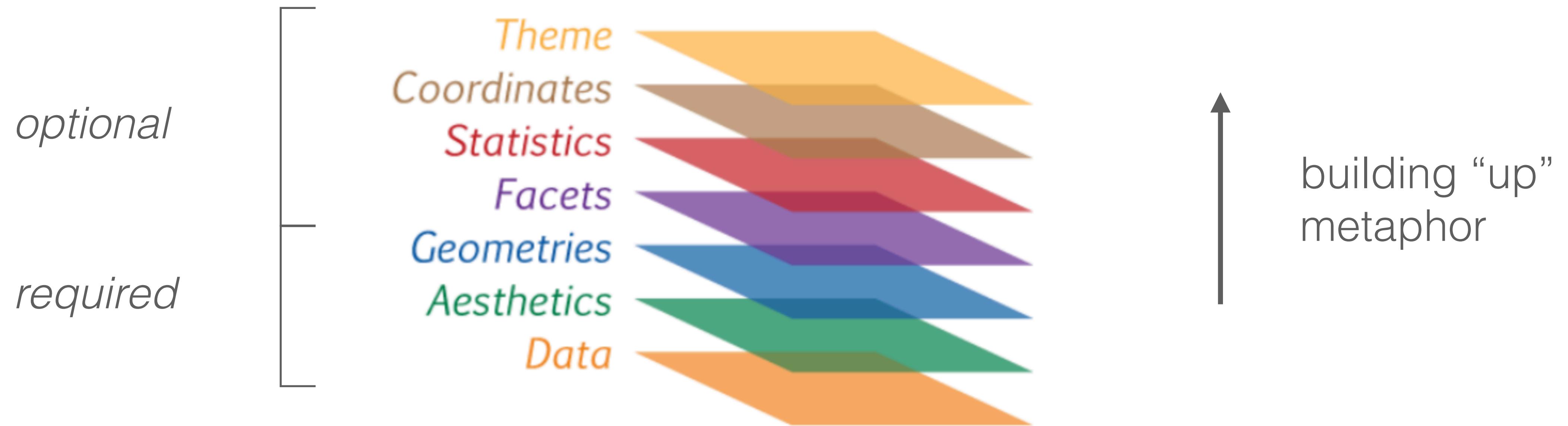
Subsequent geoms  
inherit the default aesthetics  
but can specify additional ones

If you don't want to  
inherit the default aesthetics  
can specify with argument  
`inherit.aes = FALSE`

Default  
aesthetic  
properties



# ggplot2 required and optional components



**Theme:** every part of the graph *not* related directly to data; the “look and feel”

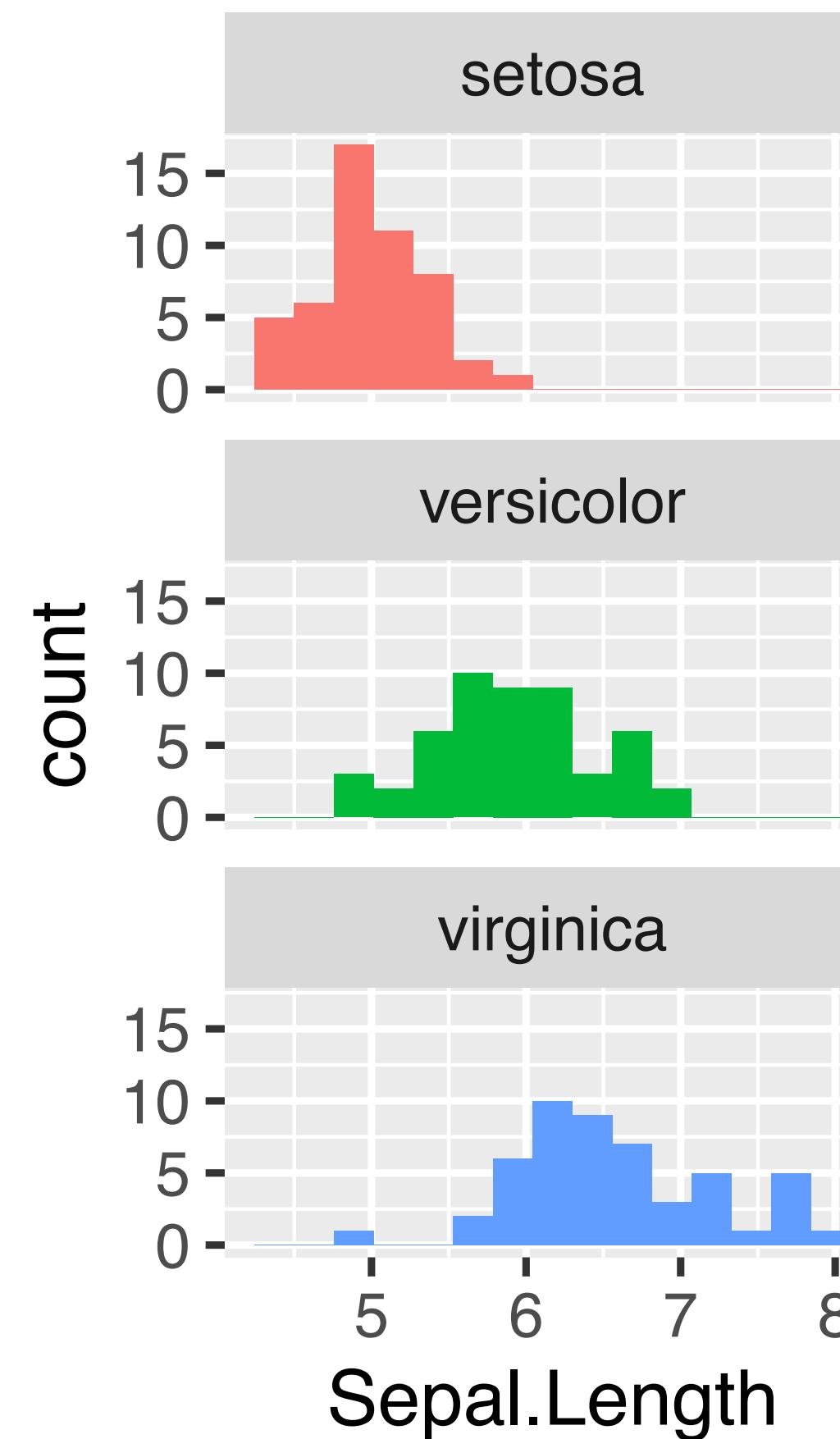
**Coordinates:** how variables are mapped, scaled, and applied to a particular geometry

**Statistics:** calculations of derived values; linked to specific geoms

**Facets:** specify multiple panels with equivalent logic, split data among panels

# Faceting creates subplots based on conditioning of one or more variables

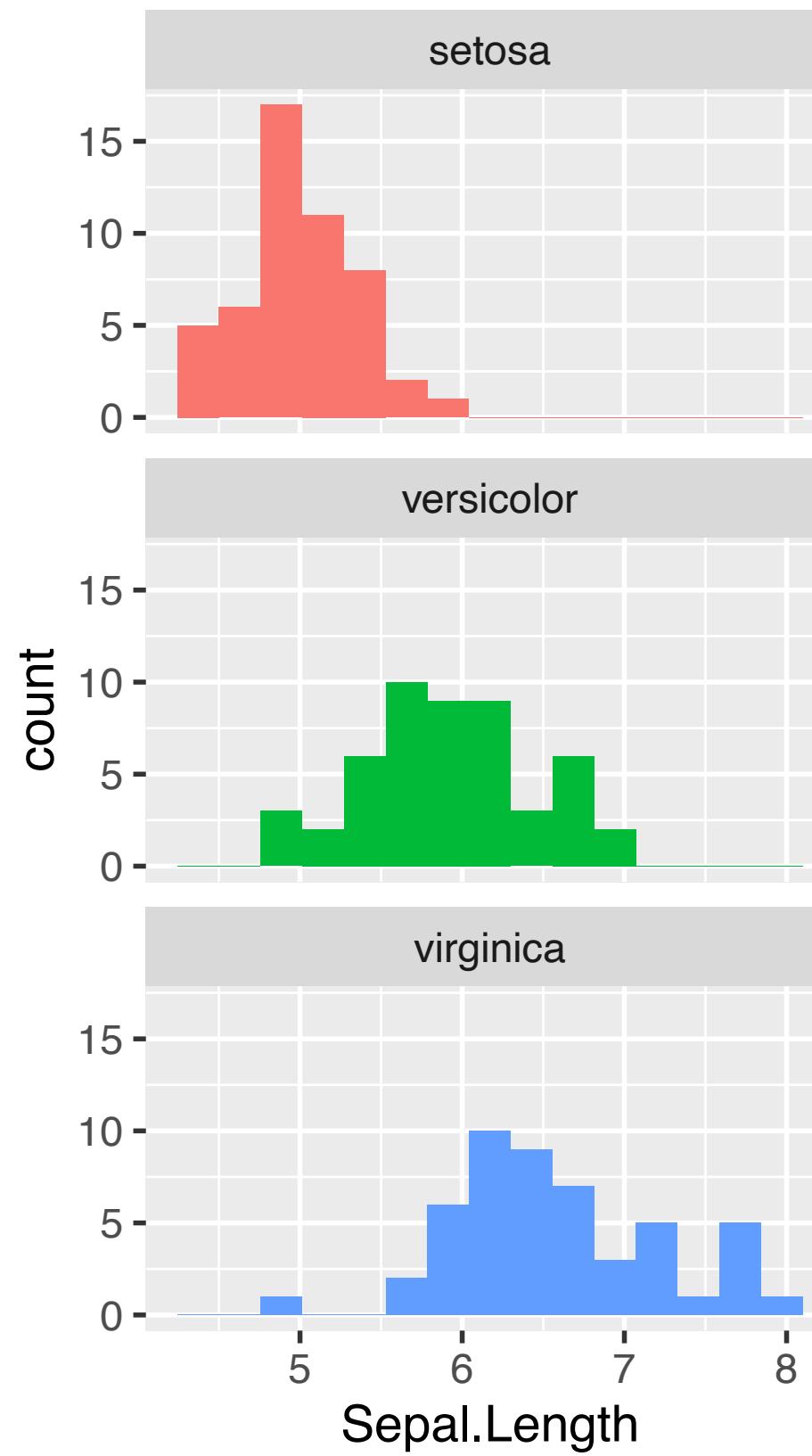
```
ggplot(data = iris, aes(x = Sepal.Length, fill=Species)) +  
  geom_histogram(bins=15) + facet_wrap(~Species,ncol=1)
```



# Themes change look-and-feel across the plot

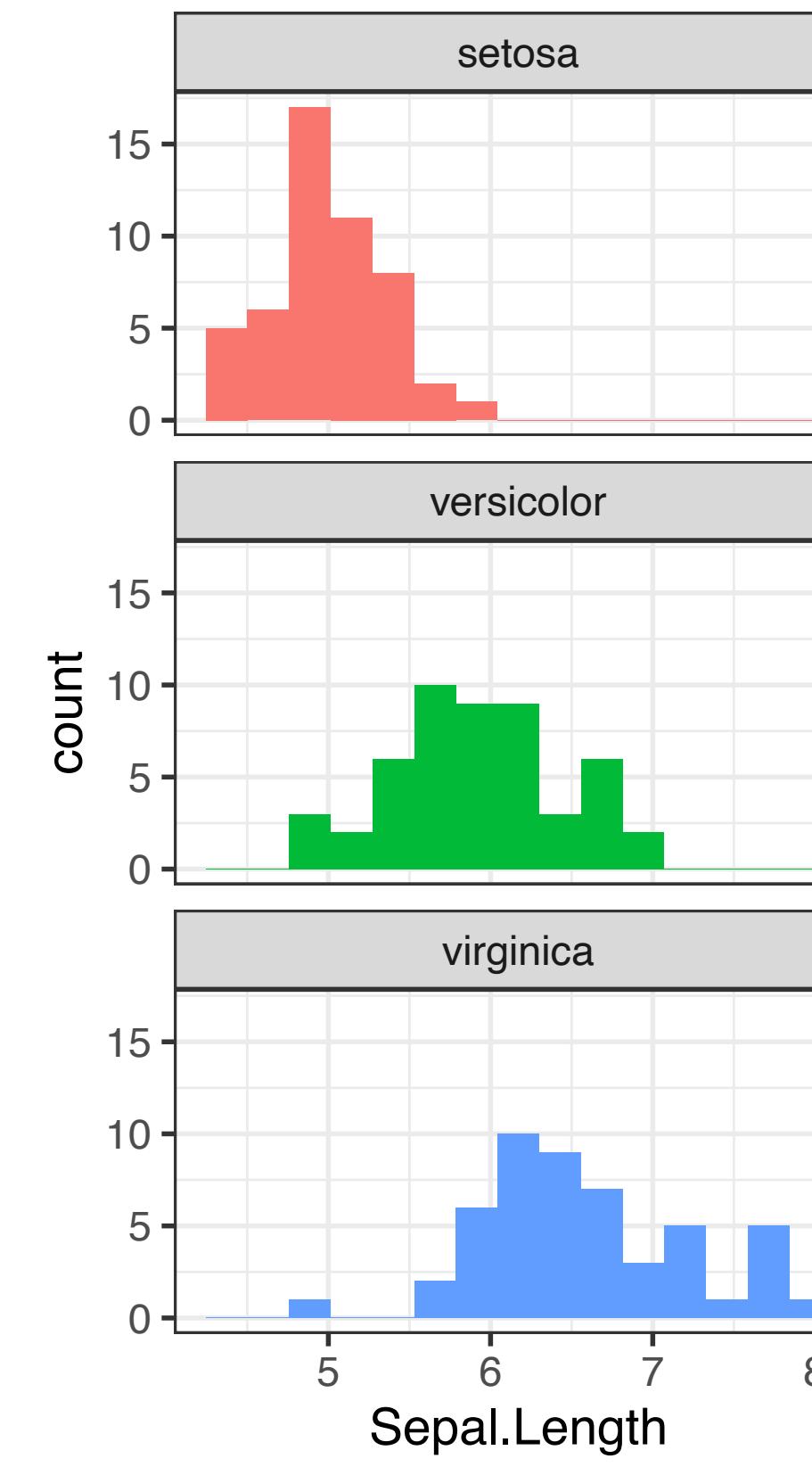
default theme

```
ggplot(data = iris, aes(x = Sepal.Length, fill=Species)) +  
  geom_histogram(bins=15) + facet_wrap(~Species,ncol=1) +  
  theme(aspect.ratio = 0.5, legend.position = "none")
```



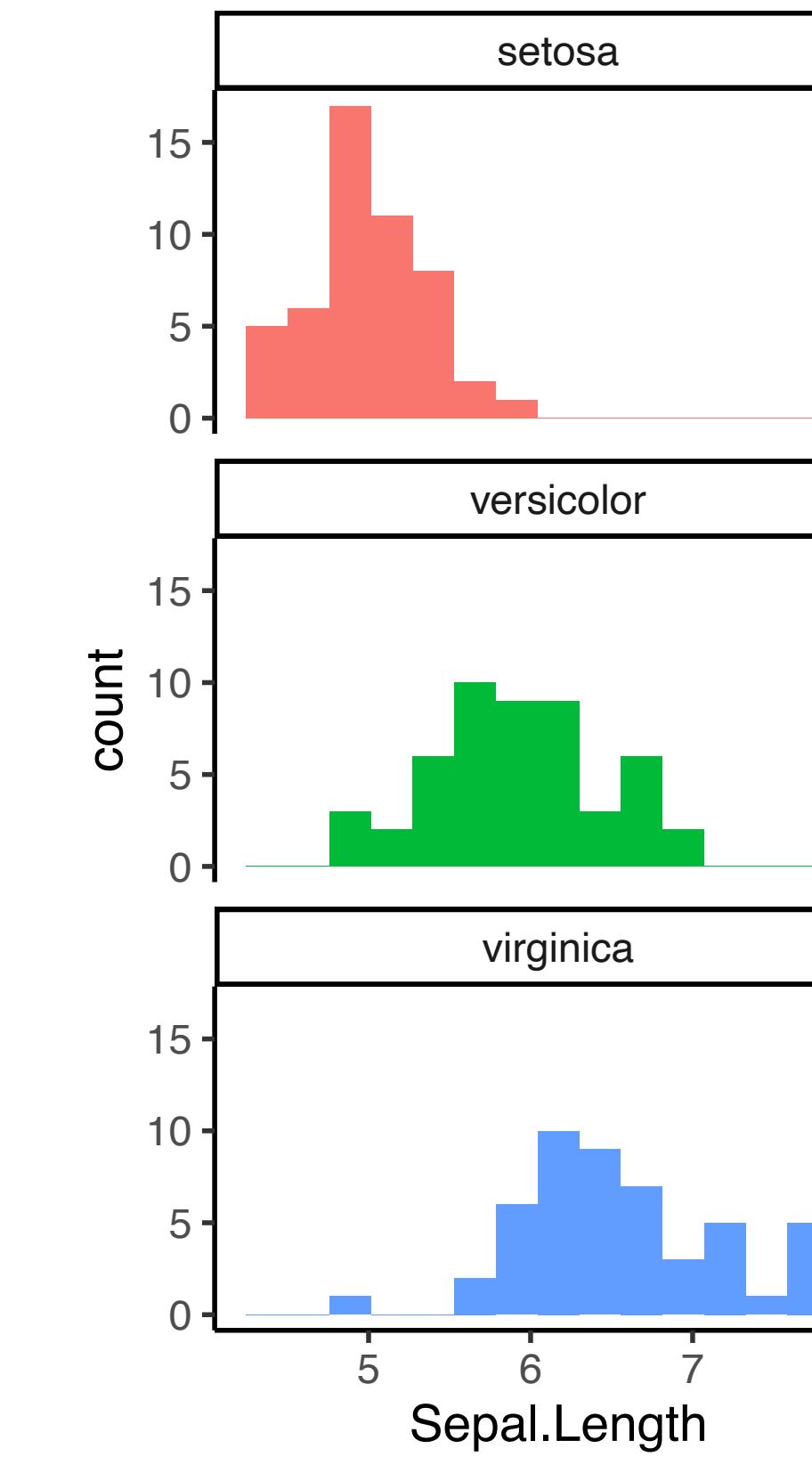
+theme\_bw()

```
ggplot(data = iris, aes(x = Sepal.Length, fill=Species)) +  
  geom_histogram(bins=15) + facet_wrap(~Species,ncol=1) +  
  theme_bw() + theme(aspect.ratio = 0.5, legend.position = "none")
```



+theme\_classic()

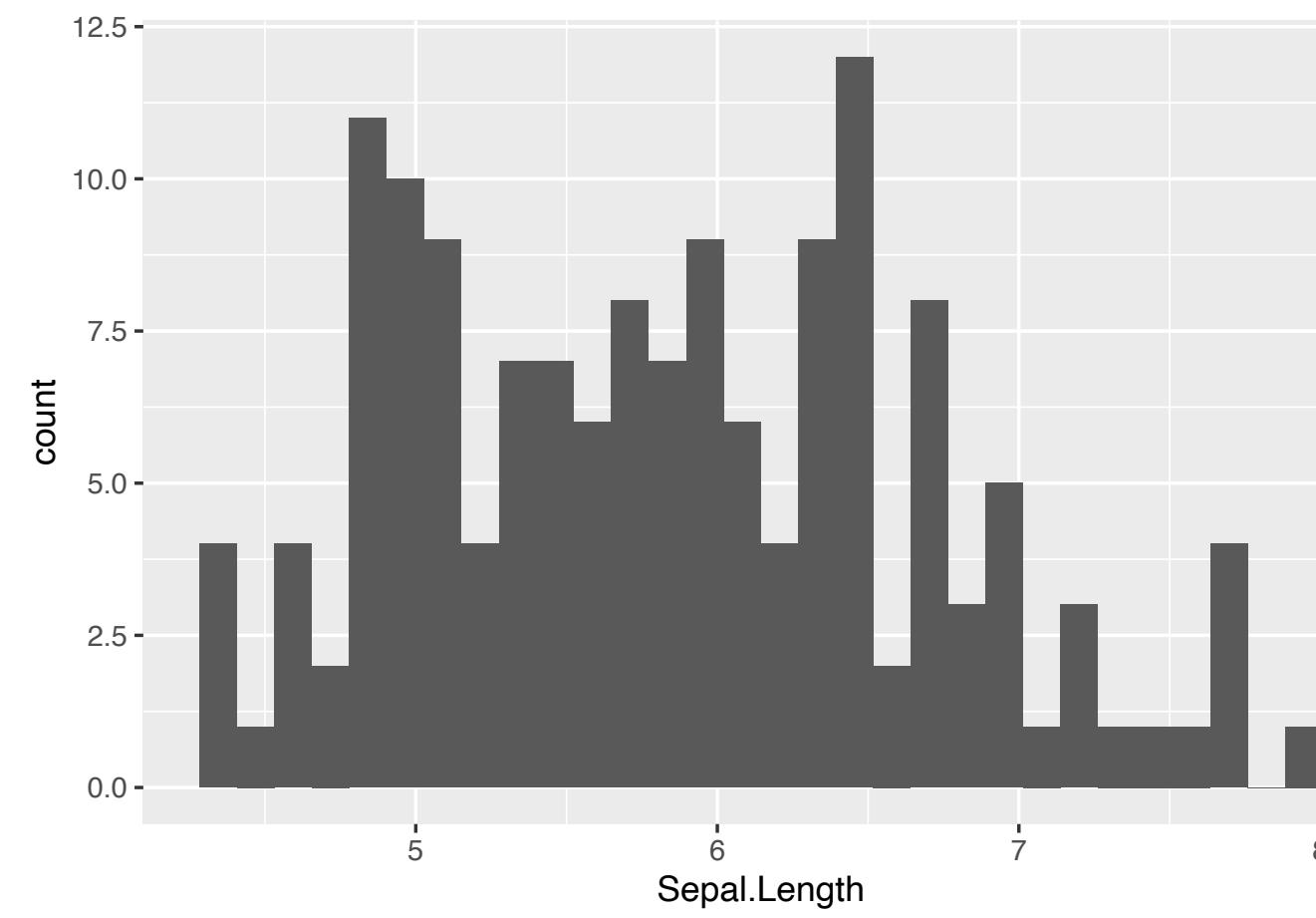
```
ggplot(data = iris, aes(x = Sepal.Length, fill=Species)) +  
  geom_histogram(bins=15) + facet_wrap(~Species,ncol=1) +  
  theme_classic() + theme(aspect.ratio = 0.5, legend.position = "none")
```



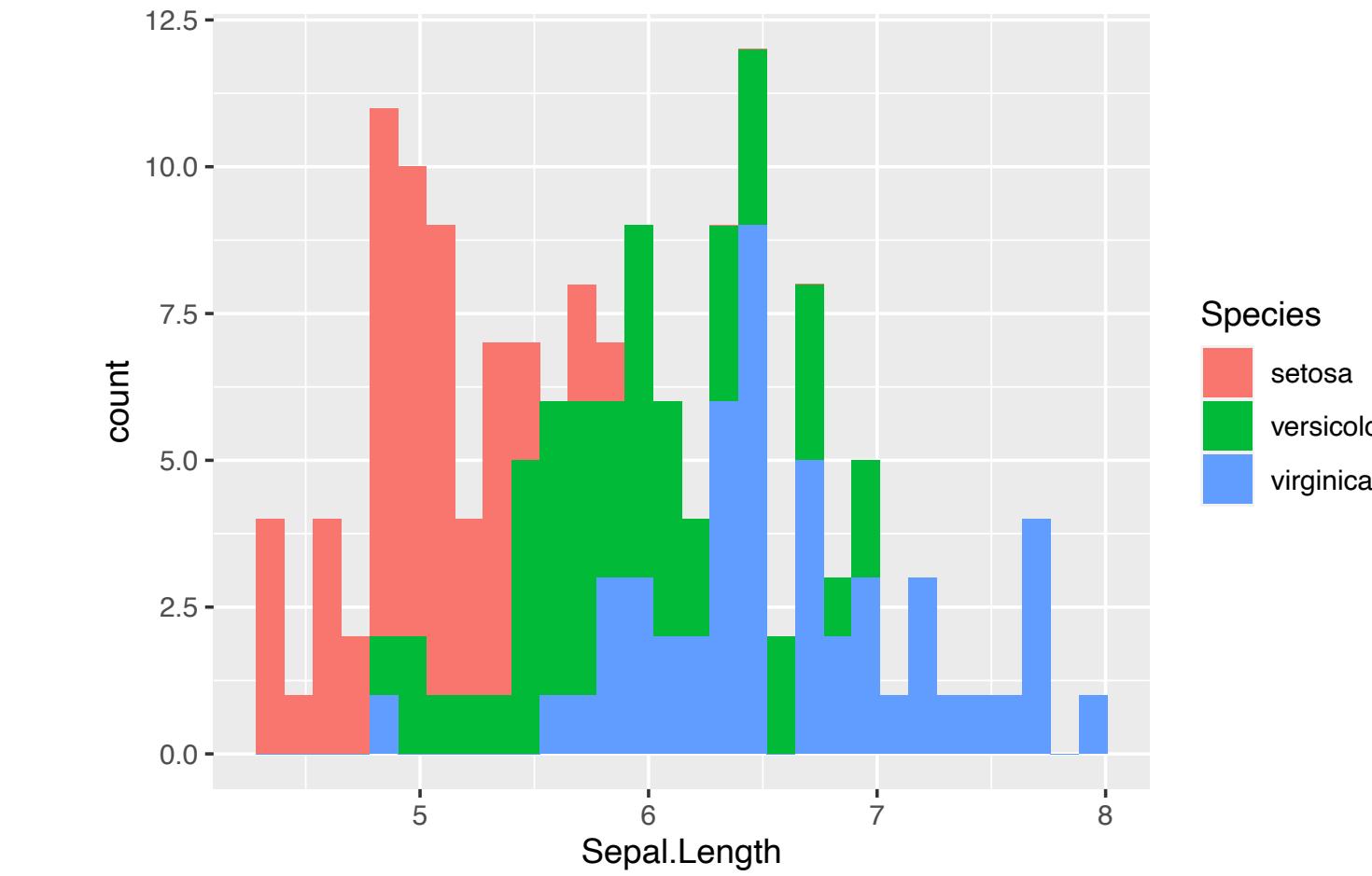
# "Geom" tour: Histograms

## geom\_histogram()

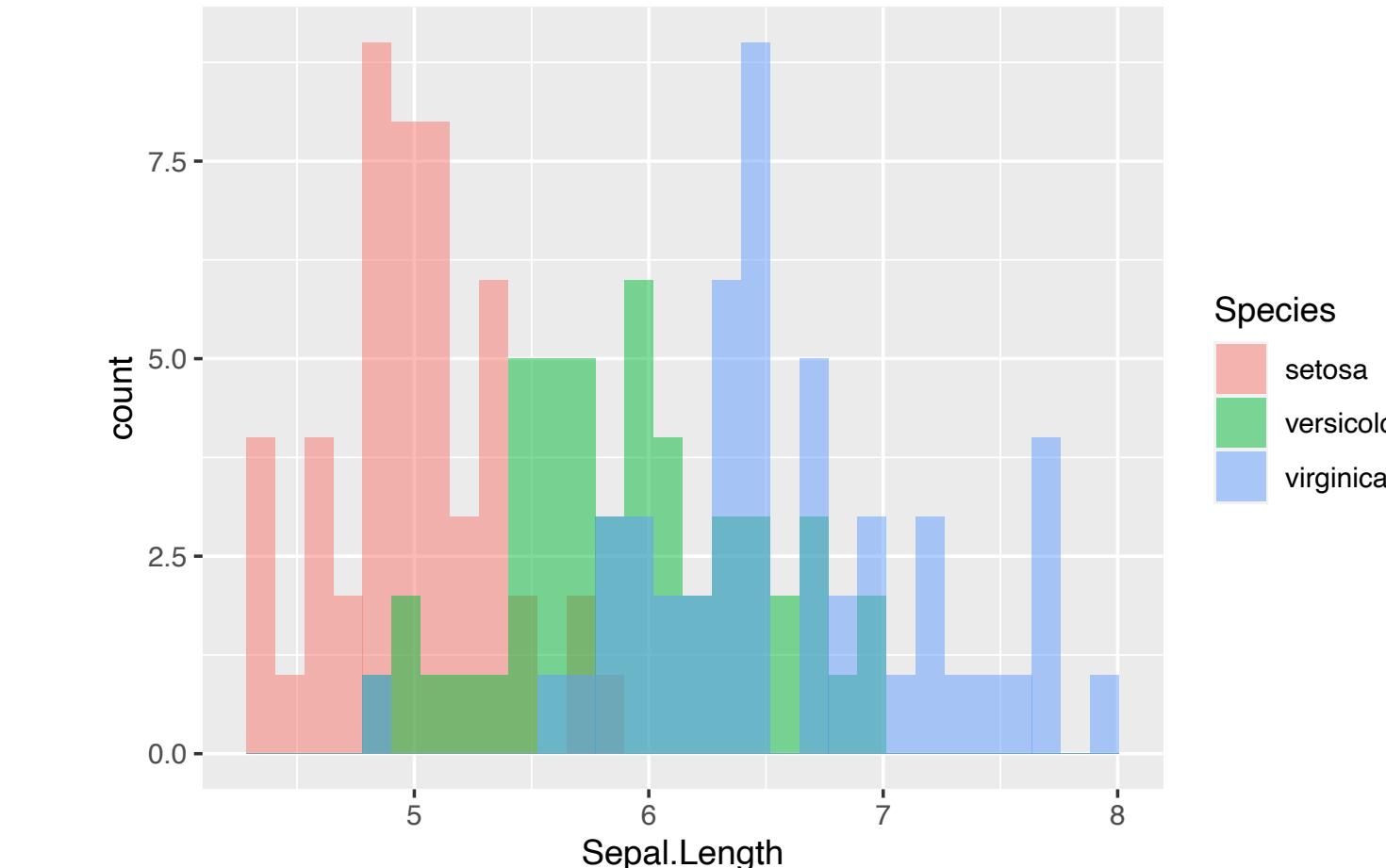
```
ggplot(iris) +  
  aes(x = Sepal.Length) +  
  geom_histogram()
```



```
ggplot(iris) +  
  aes(x = Sepal.Length, fill = Species) +  
  geom_histogram()
```



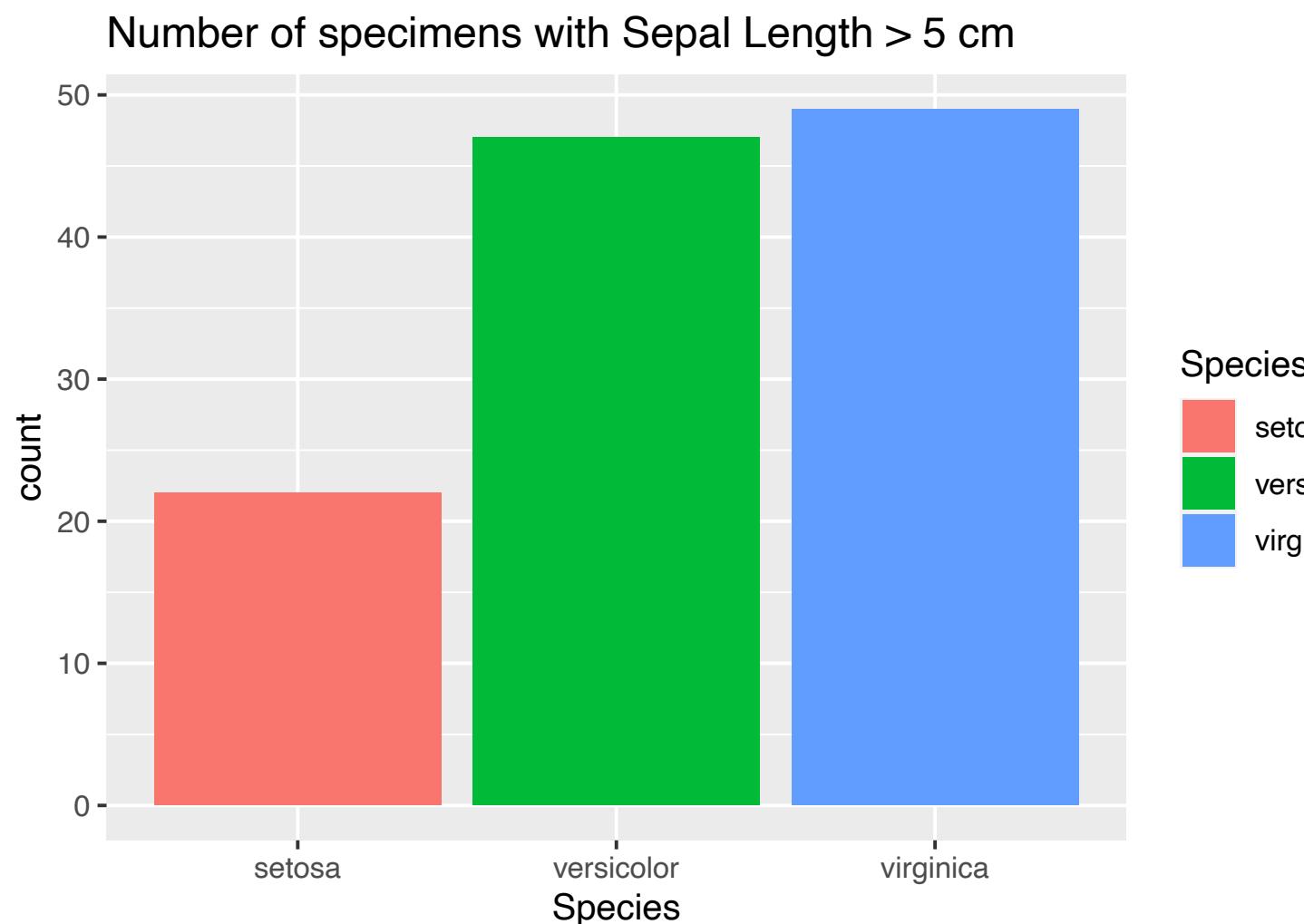
```
ggplot(iris) +  
  aes(x = Sepal.Length, fill = Species) +  
  geom_histogram(position = "identity", alpha=0.5)
```



# "Geom" tour: Bar charts

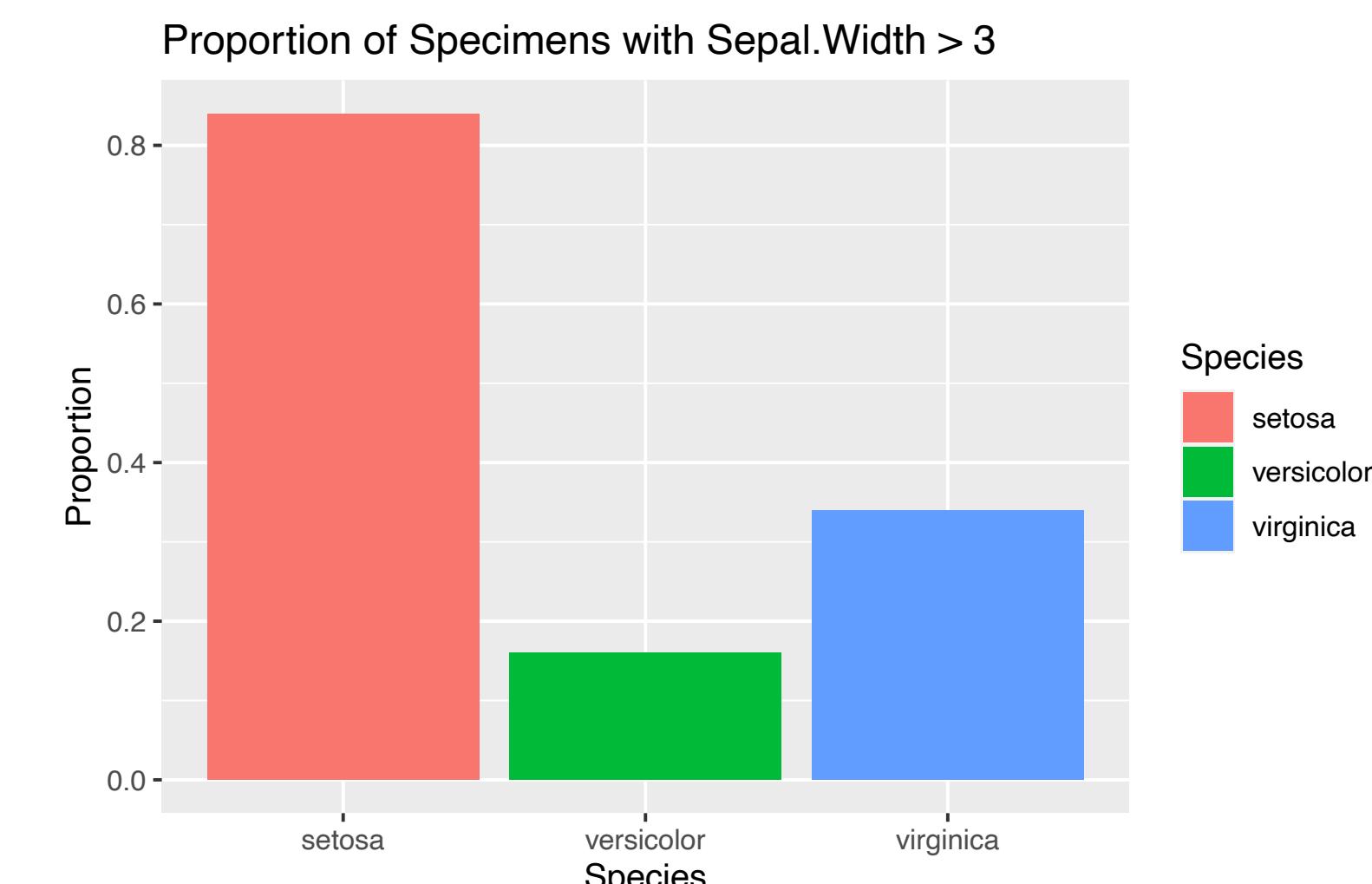
geom\_bar()

```
iris |>
  filter(Sepal.Length > 5) |>
  ggplot(aes(x = Species, fill = Species)) +
  geom_bar() +
  labs(title = "Number of specimens with Sepal Length > 5 cm")
```



geom\_col()

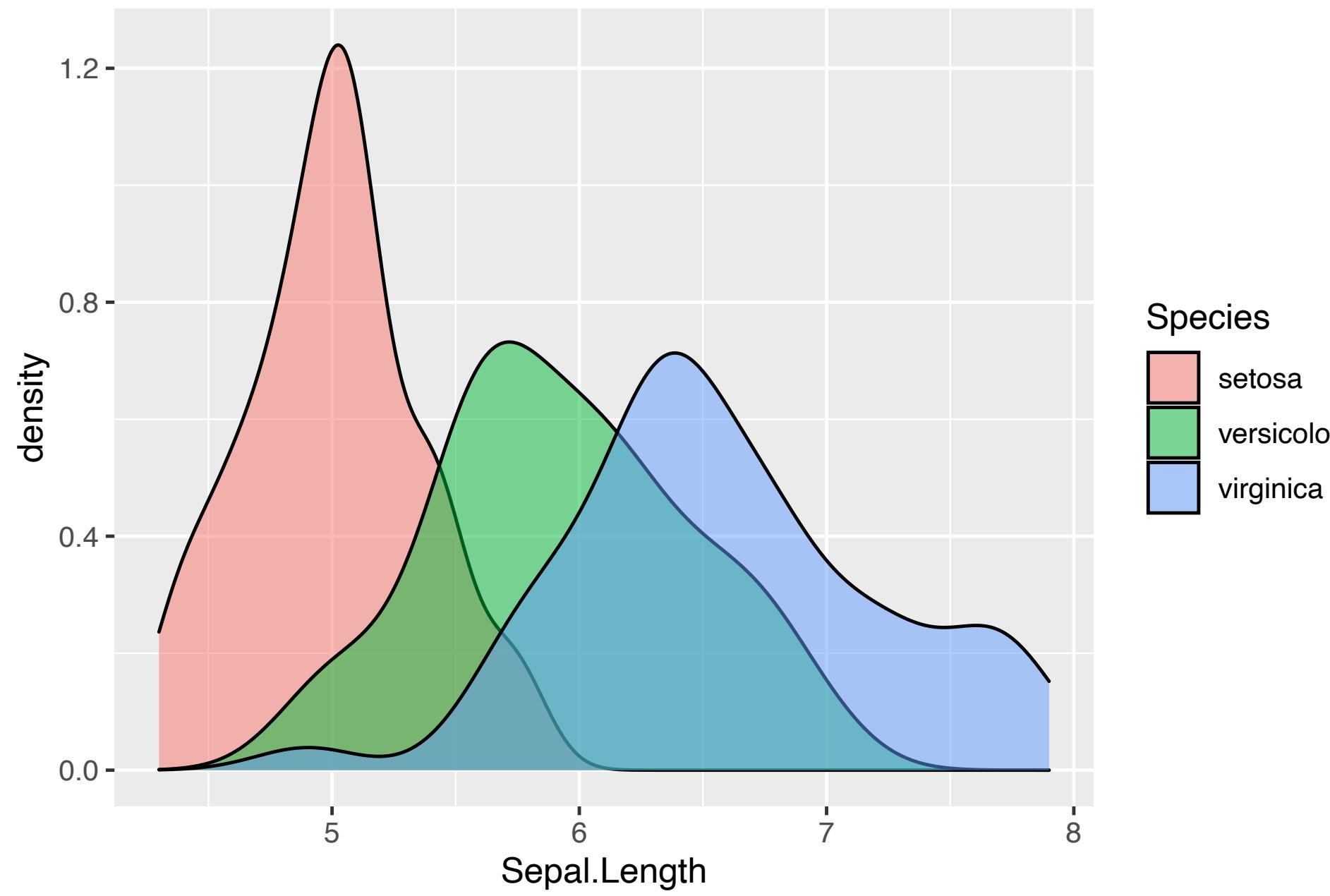
```
iris |>
  group_by(Species) |>
  summarize(prop_Sepal.Width_gt3 = sum(Sepal.Width > 3)/n()) |>
  ggplot(aes(x = Species, fill = Species,
             y = prop_Sepal.Width_gt3)) +
  geom_col() +
  labs(y = "Proportion",
       title = "Proportion of Specimens with Sepal.Width > 3")
```



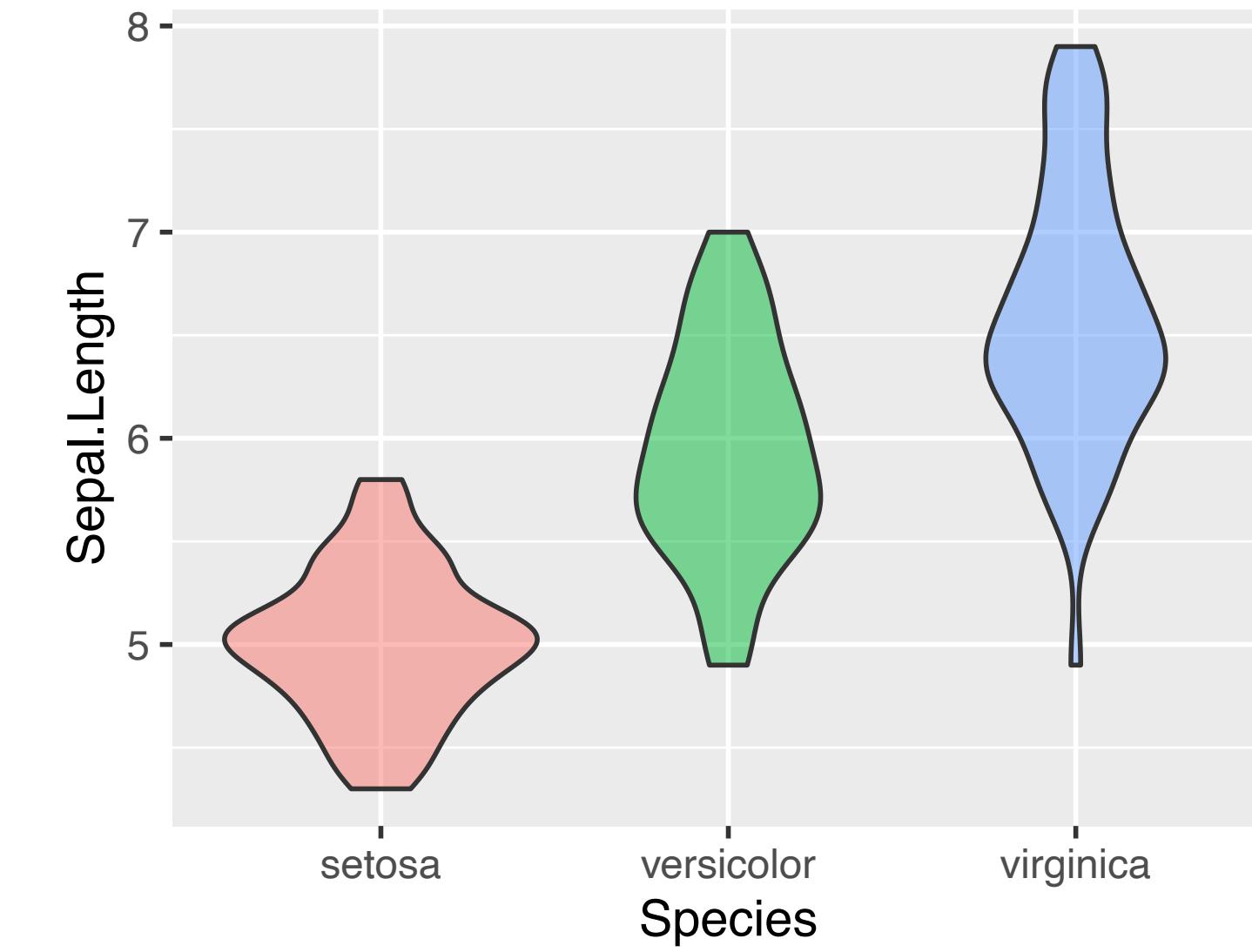
Use `geom_col()` when  
you've pre-computed the values  
you want to depict

# "Geom" tour: Density and violin plots

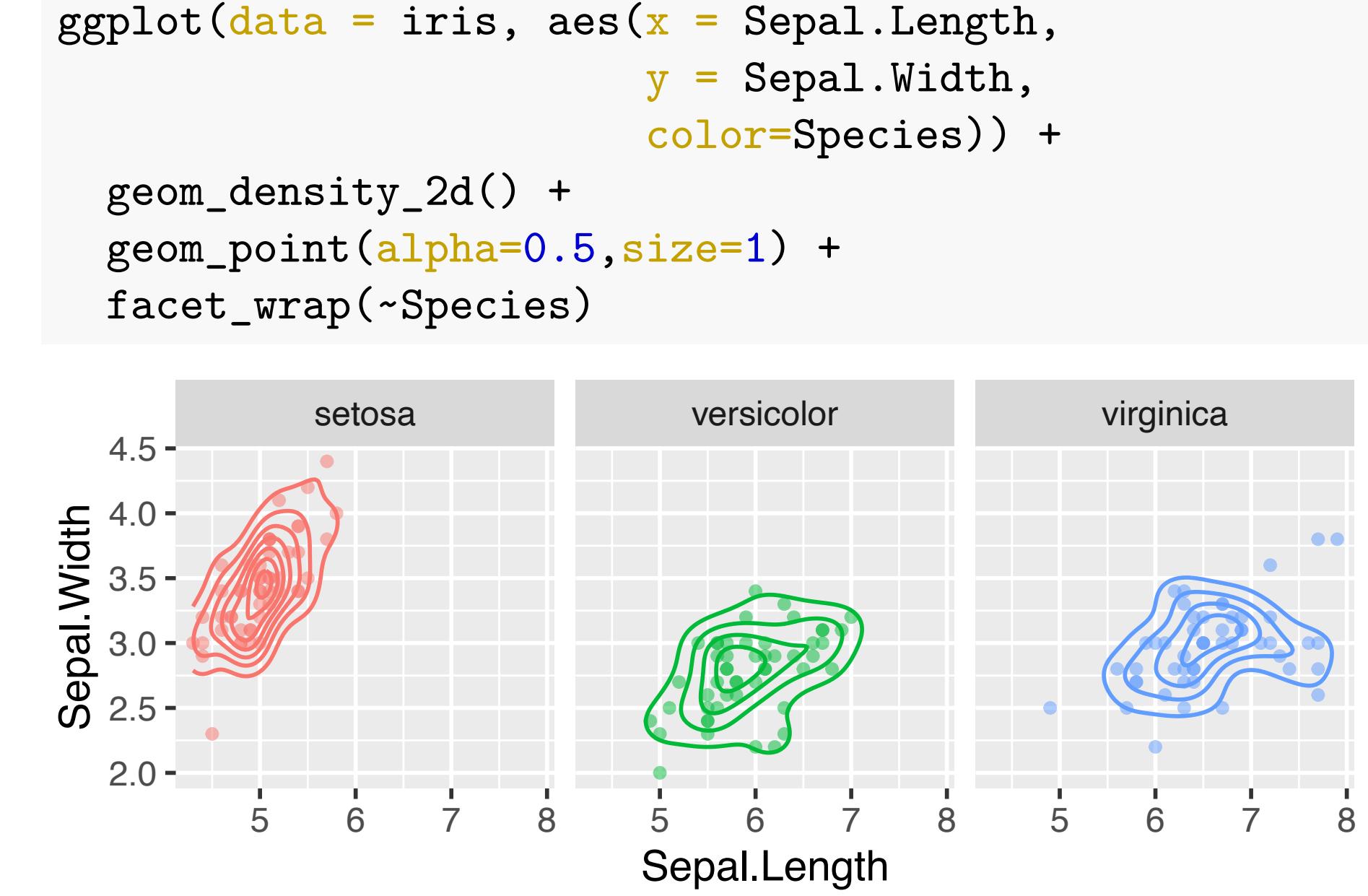
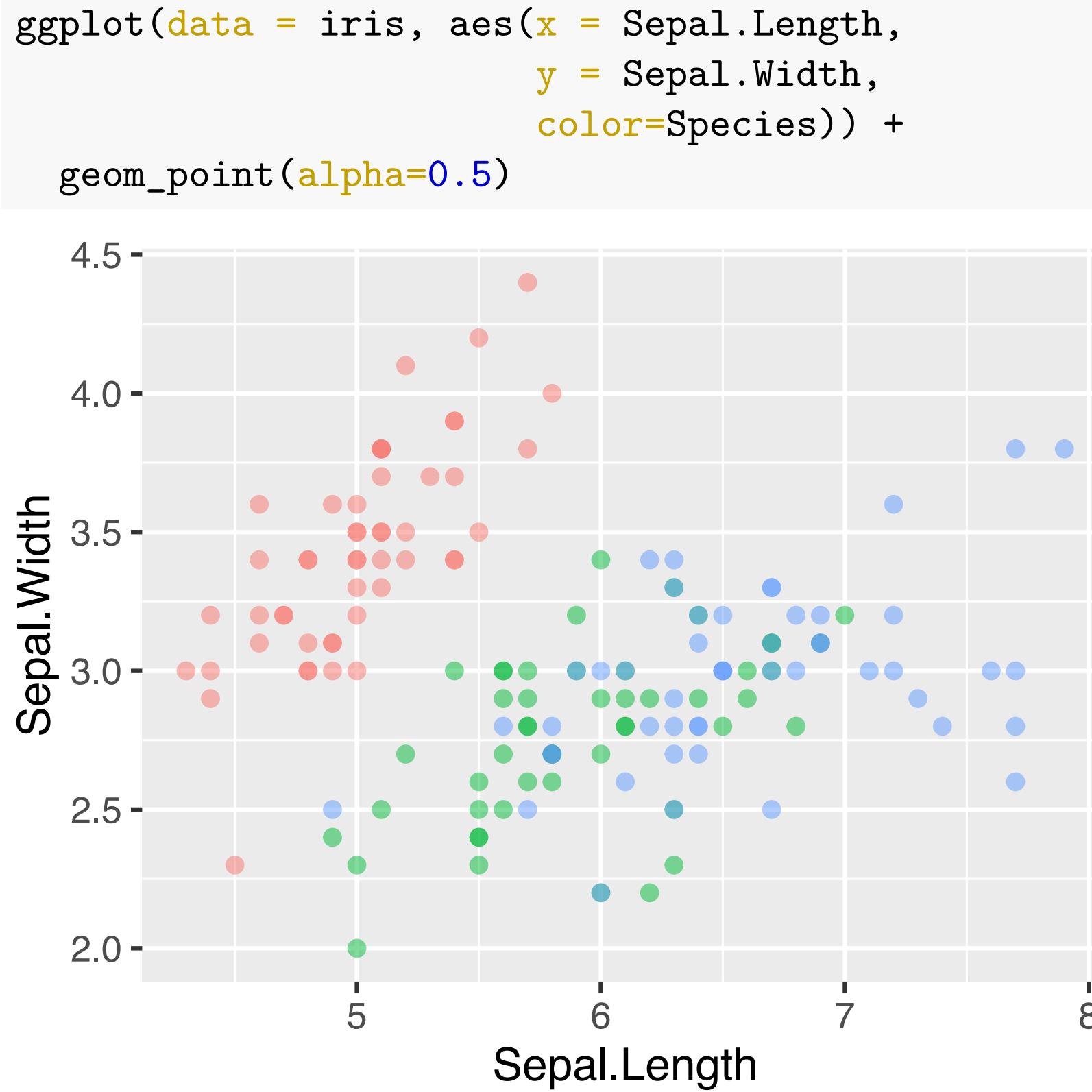
```
ggplot(data = iris, aes(x = Sepal.Length, fill=Species)) +  
  geom_density(alpha=0.5)
```



```
ggplot(data = iris, aes(x = Species,  
y = Sepal.Length,  
fill=Species)) +  
  geom_violin(alpha=0.5)
```



# "Geom" tour: Scatter and 2d density

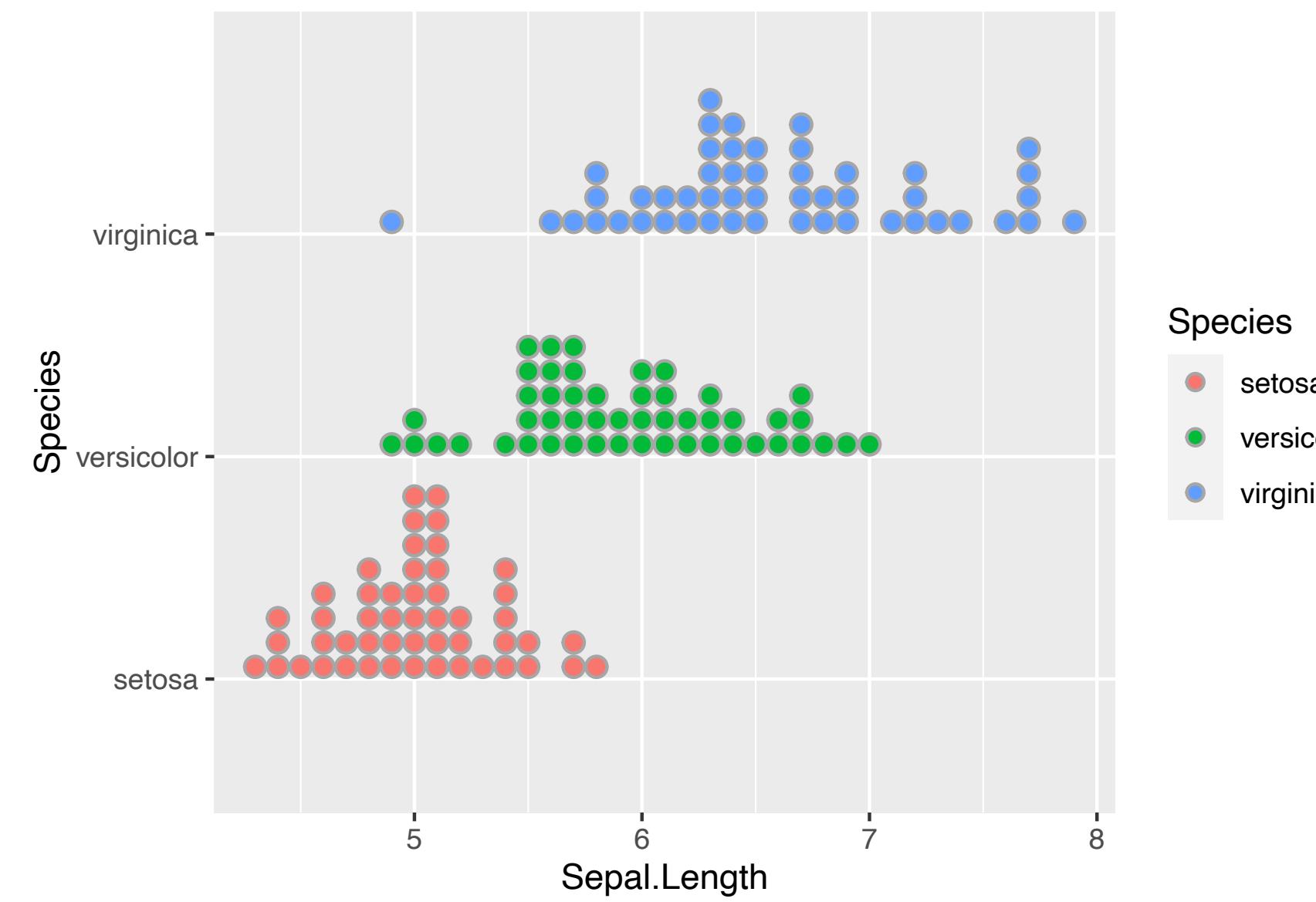


# ggplot2 extension: ggdist for comparing distributions

geom\_dots()

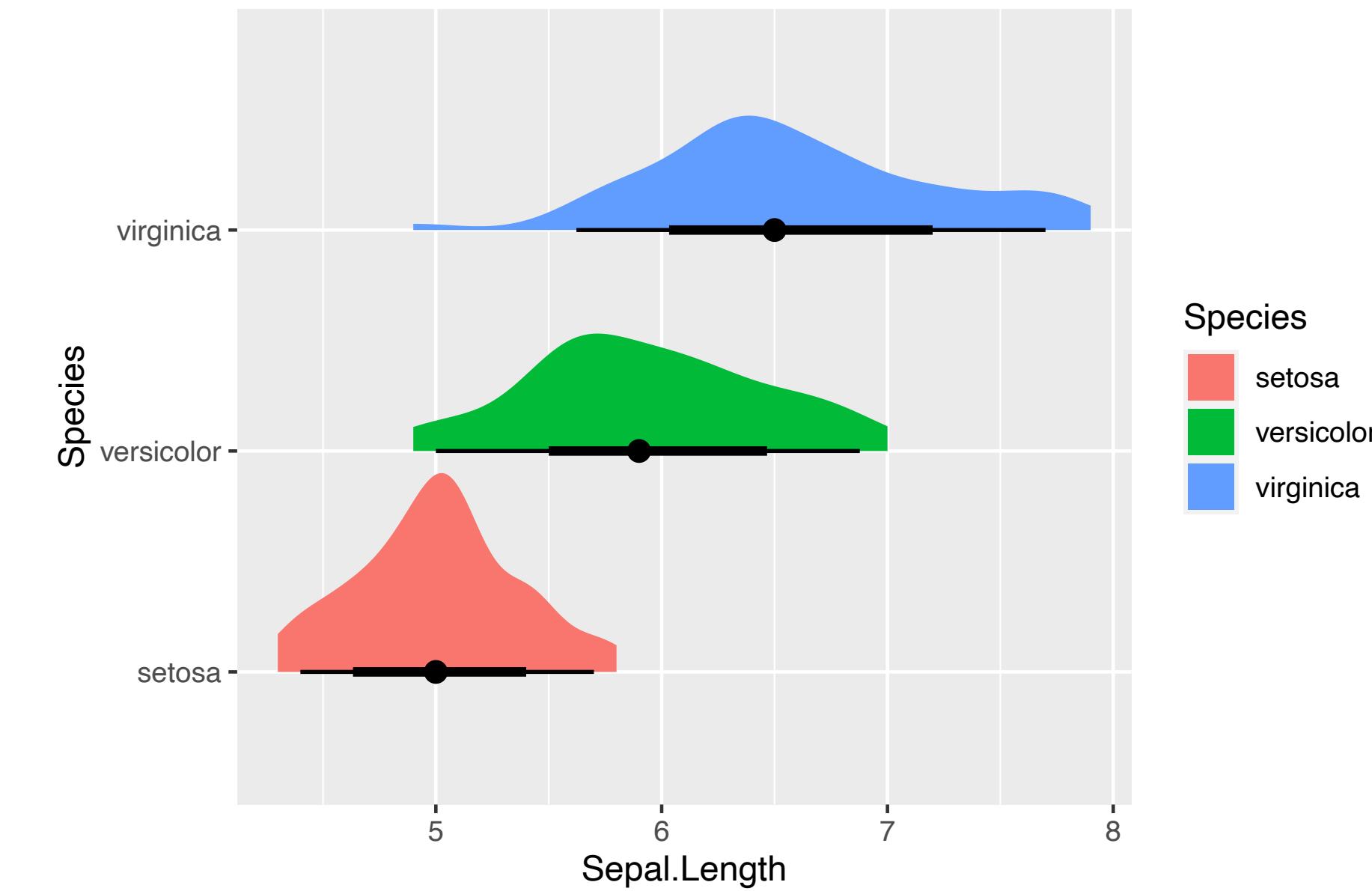
```
library(ggdist)

ggplot(iris, aes(x = Sepal.Length,
                  y = Species,
                  fill=Species)) +
  geom_dots()
```



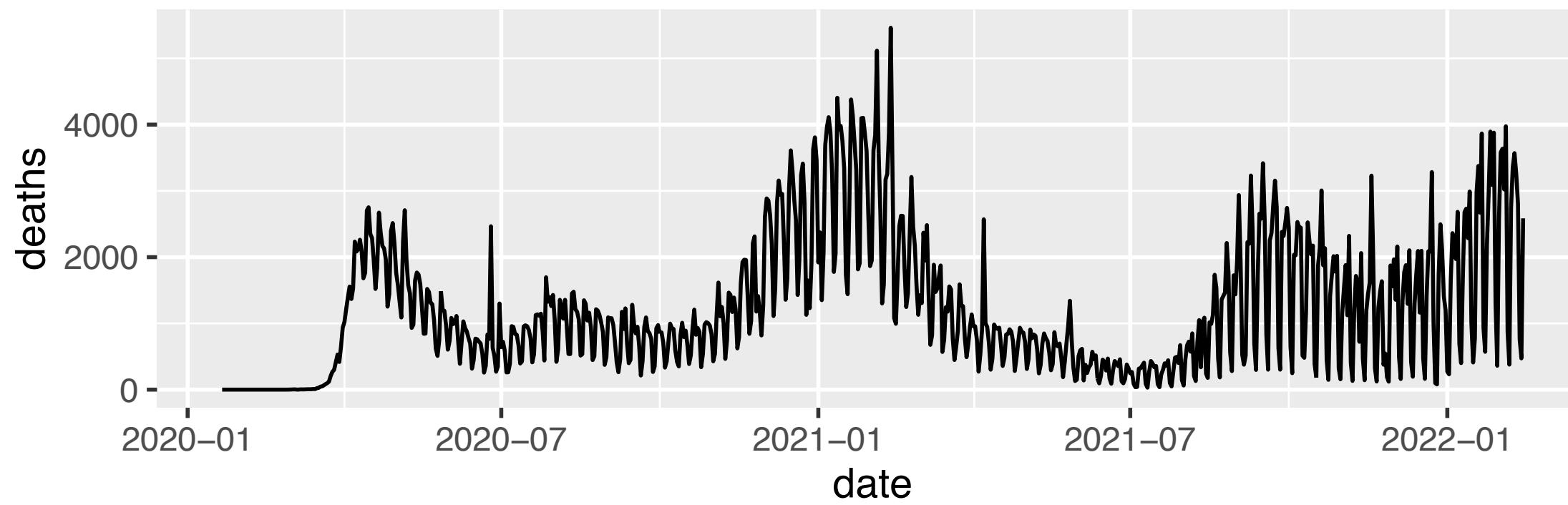
stat\_slabinterval()

```
ggplot(iris, aes(x = Sepal.Length,
                  y = Species,
                  fill=Species)) +
  stat_slabinterval()
```

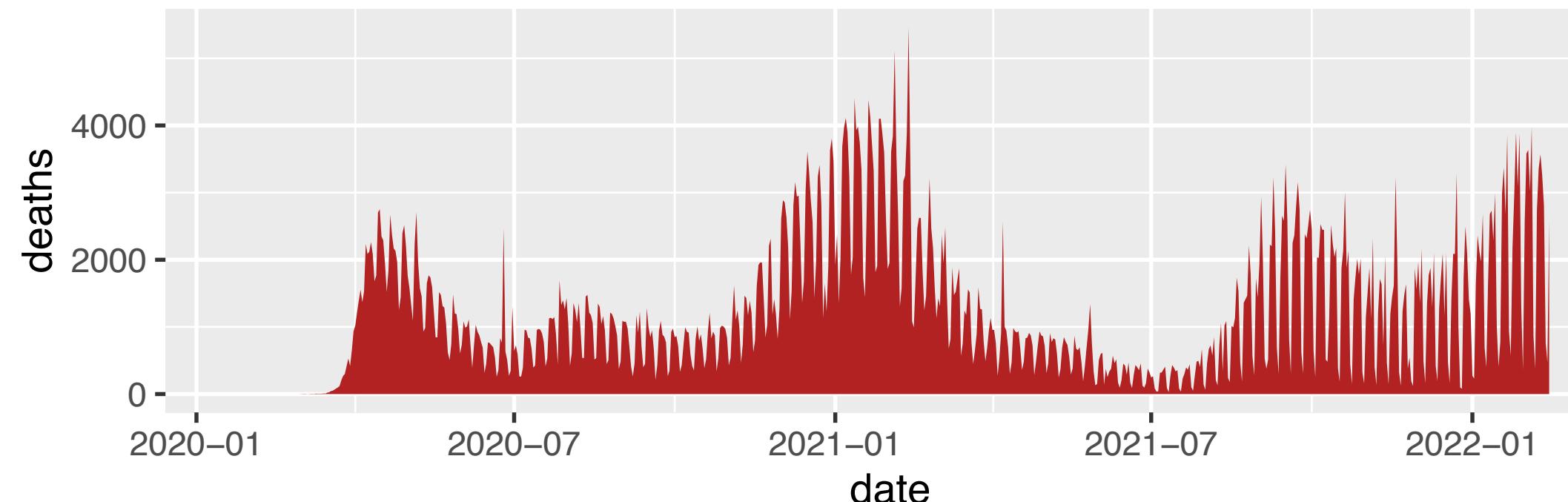


# "Geom" tour: Line and Area plots

```
ggplot(data = covid, aes(x = date, y = deaths)) +  
  geom_line()
```



```
ggplot(data = covid, aes(x = date, y = deaths)) +  
  geom_area(fill="firebrick")
```

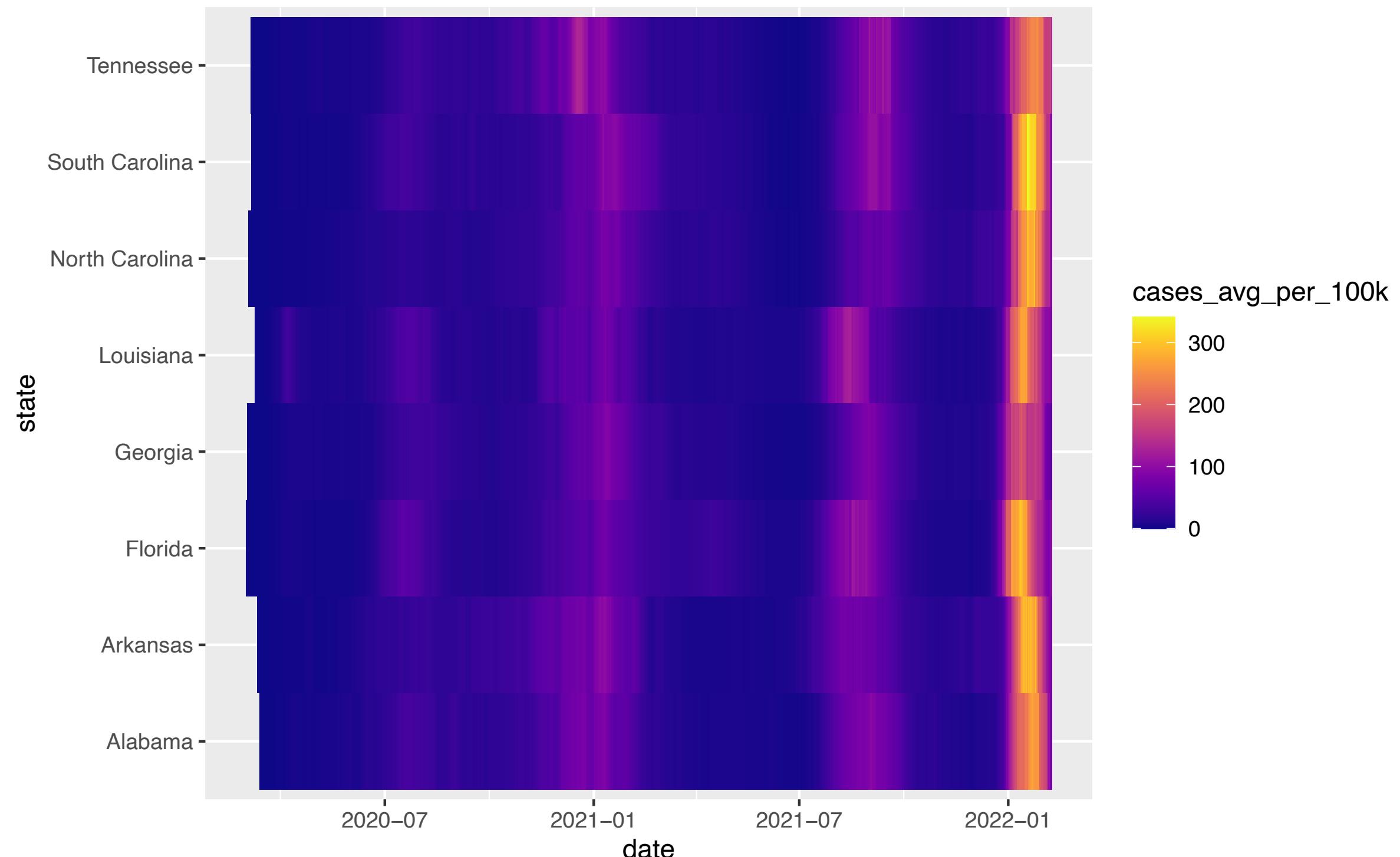


# "Geom" tour: Heat maps

geom\_tile() or geom\_raster()

```
SE_states <- c("North Carolina", "South Carolina",
               "Arkansas", "Georgia", "Tennessee",
               "Louisiana", "Alabama", "Florida")

us_states %>%
  filter(state %in% SE_states) %>%
  arrange(state) %>%
  ggplot(aes(x=date, y=state, fill=cases_avg_per_100k)) +
  geom_tile() +
  scale_fill_viridis(option = "C")
```



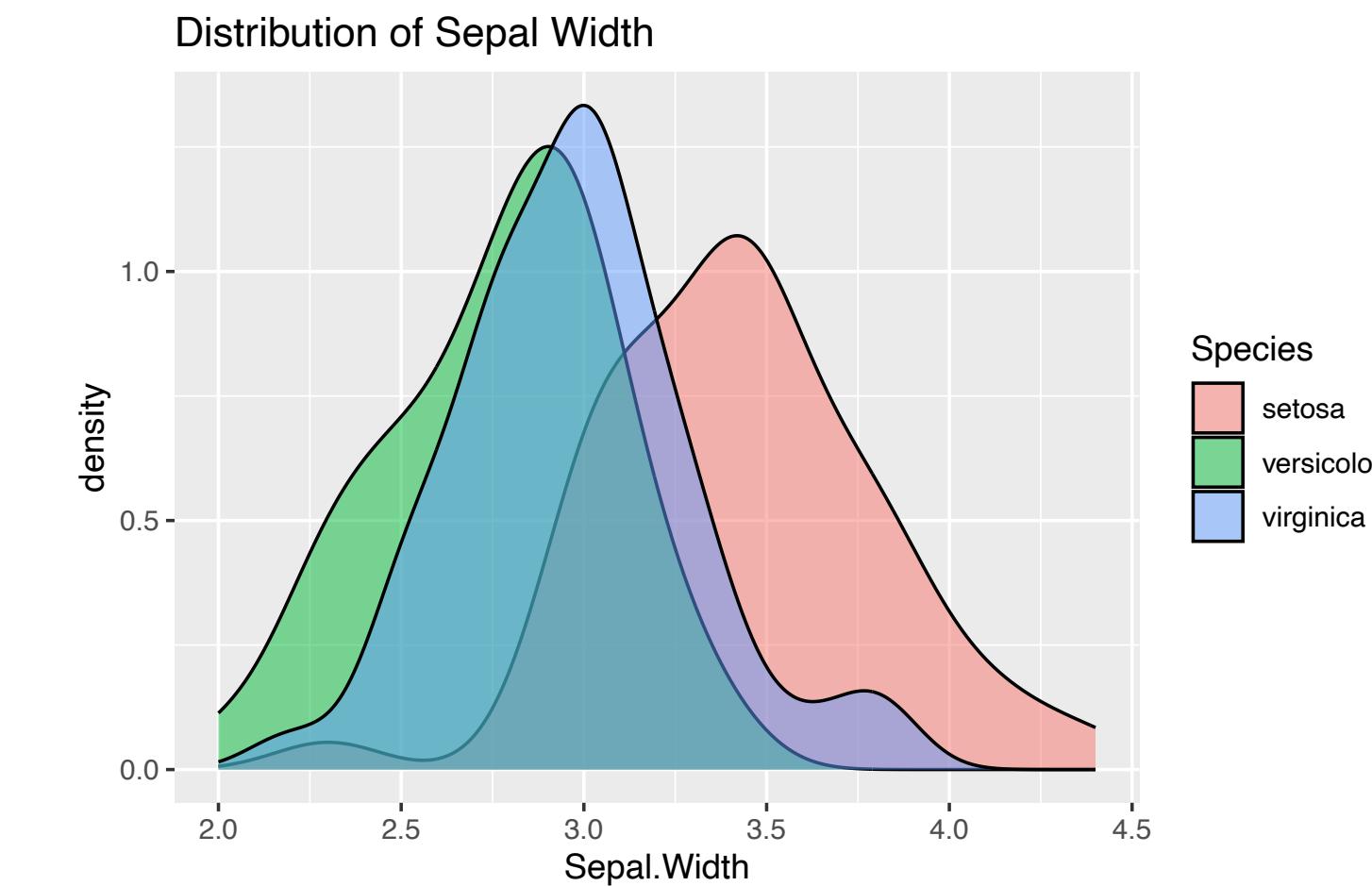
# Assigning plots to variables

```
plot_Sepal.Length <-
  ggplot(data = iris, aes(x = Sepal.Length, fill=Species)) +
  geom_density(alpha=0.5) +
  labs(title = "Distribution of Sepal Length")

plot_Sepal.Width <-
  ggplot(data = iris, aes(x = Sepal.Width, fill=Species)) +
  geom_density(alpha=0.5) +
  labs(title = "Distribution of Sepal Width")

plot_Sepal.Ratio <-
  ggplot(data = iris, aes(x = Sepal.Width/Sepal.Length, fill=Species)) +
  geom_density(alpha=0.5) +
  labs(x = "Sepal Ratio (Width / Length)",
       title = "Distribution of Sepal Ratios")

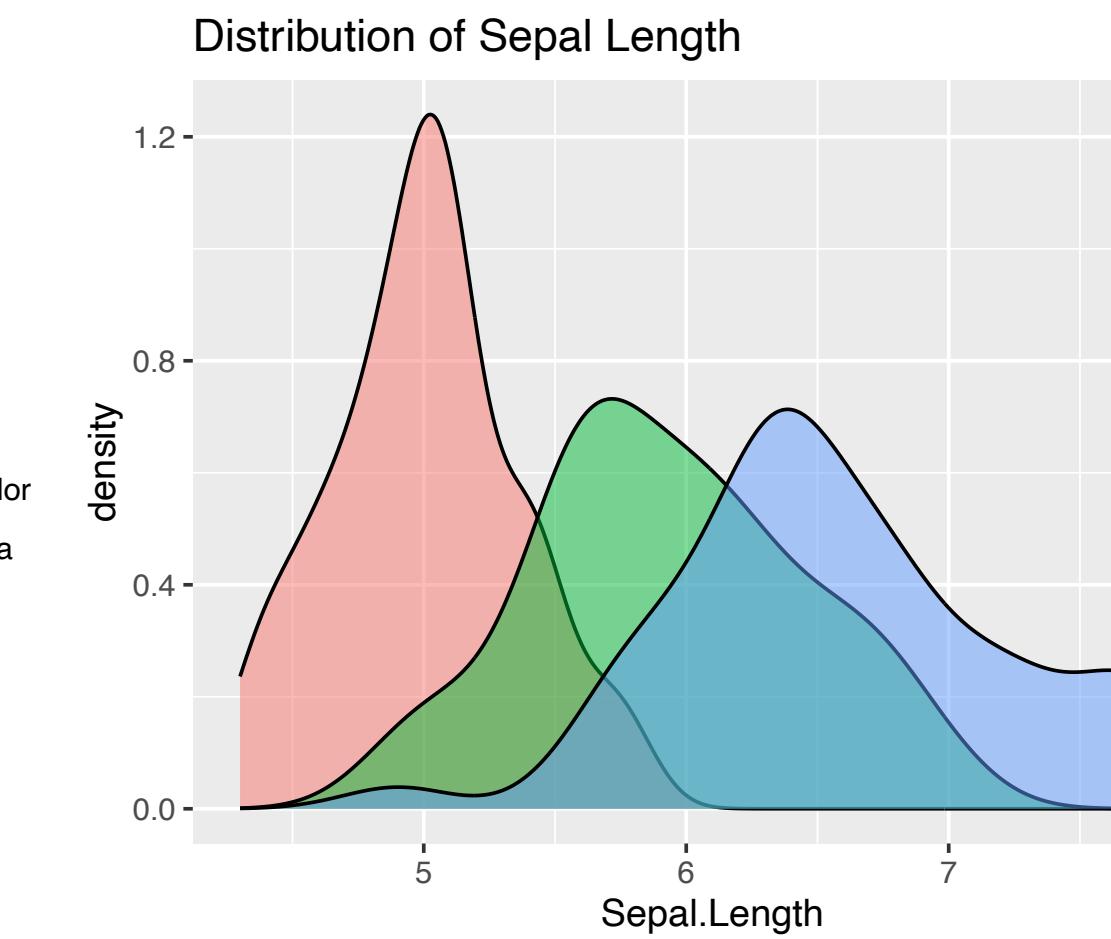
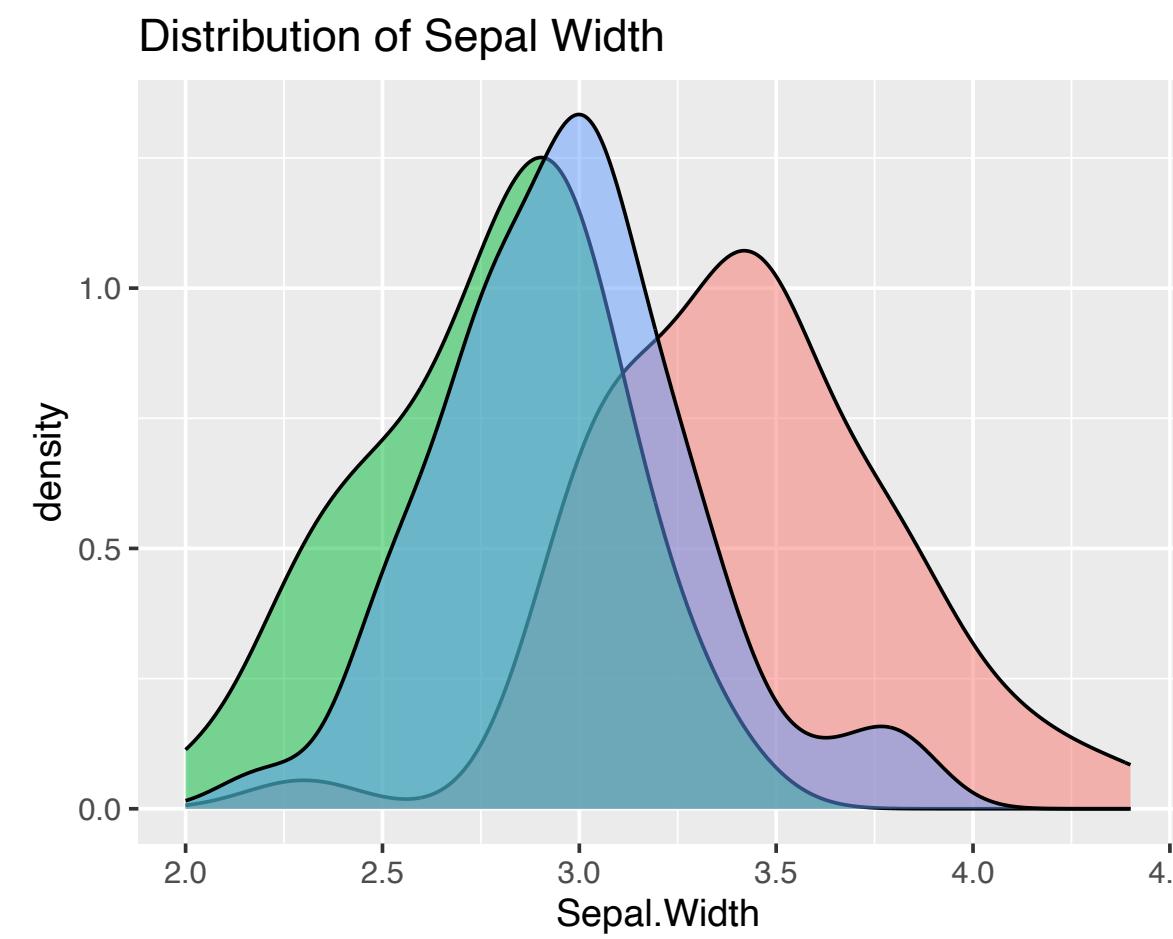
# Note that a plot is not constructed until
# we evaluate one of the variables holding a plot
plot_Sepal.Width
```



# Multi-plot layouts with the patchwork library

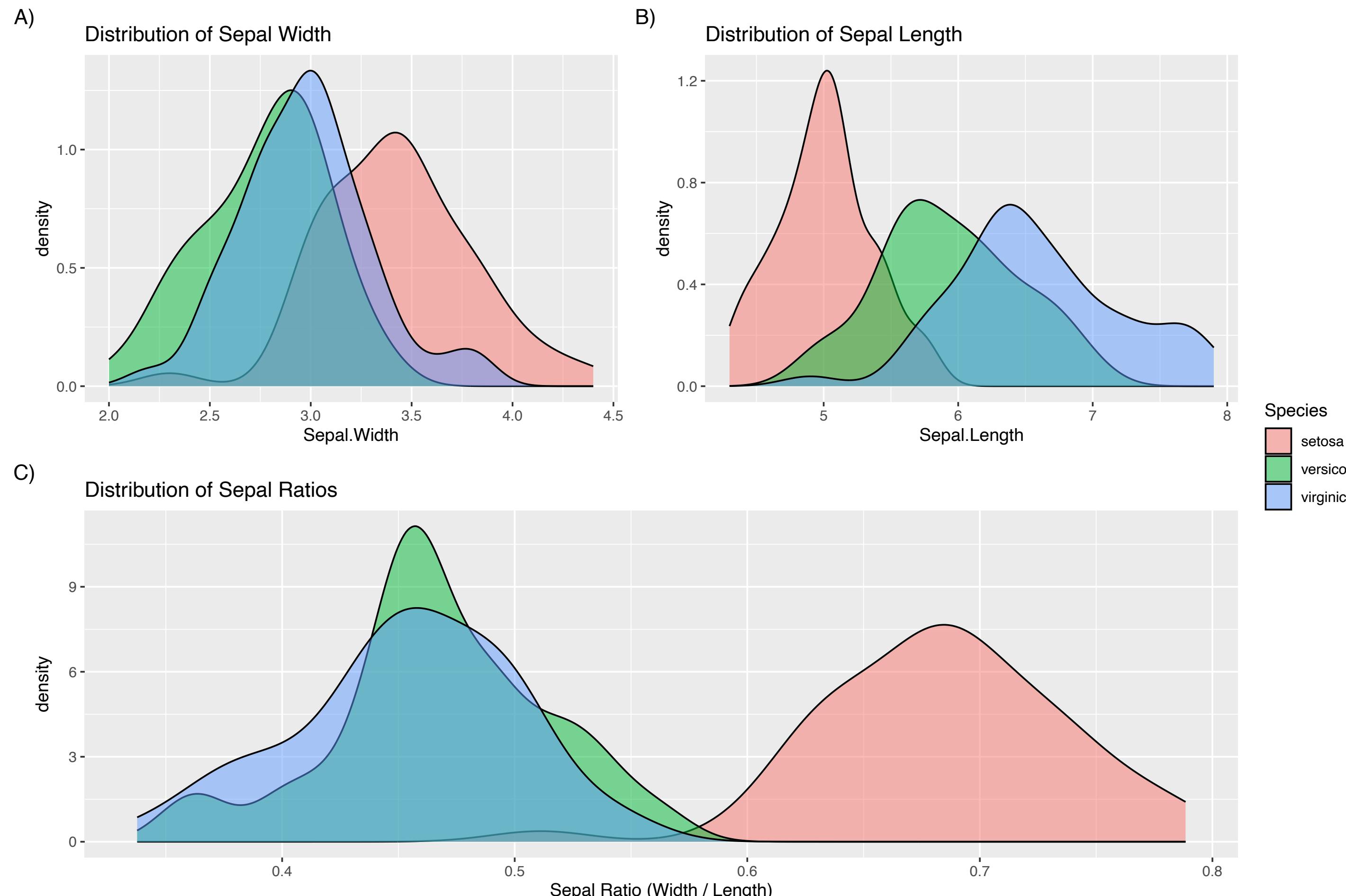
```
library(patchwork)

# Plots can be laid out side by side using the + operator
plot_Sepal.Width + plot_Sepal.Length
```



# Even more complex layouts with patchwork

```
(plot_Sepal.Width | plot_Sepal.Length) / plot_Sepal.Ratio +  
  plot_layout(guides = "collect") +  
  plot_annotation(tag_levels = 'A', tag_suffix = ")")
```



# ggtext allows for simple HTML and markdown formatting of text and labels

```
library(ggtext)

# a named vector provides an explicit map
# between names as given in the data frame
# and the labels we want printed
species_names = c("setosa" = "<i>I. setosa</i>",
                  "versicolor" = "<i>I. versicolor</i>",
                  "virginica" = "<i>I. virginica</i>")

ggplot(iris, aes(x = Sepal.Length, fill=Species)) +
  geom_density(alpha=0.5) +
  labs(title = "Distribution of Sepal Length in three *Iris* species",
       x = "Sepal Length", y = "Density") +
  scale_fill_discrete(labels = species_names) +
  theme(plot.title = element_markdown(),
        legend.text = element_markdown())
```

