Innovations in aesthetic evaluation semantics:

Where ggplot2 users and developers meet

June Choe

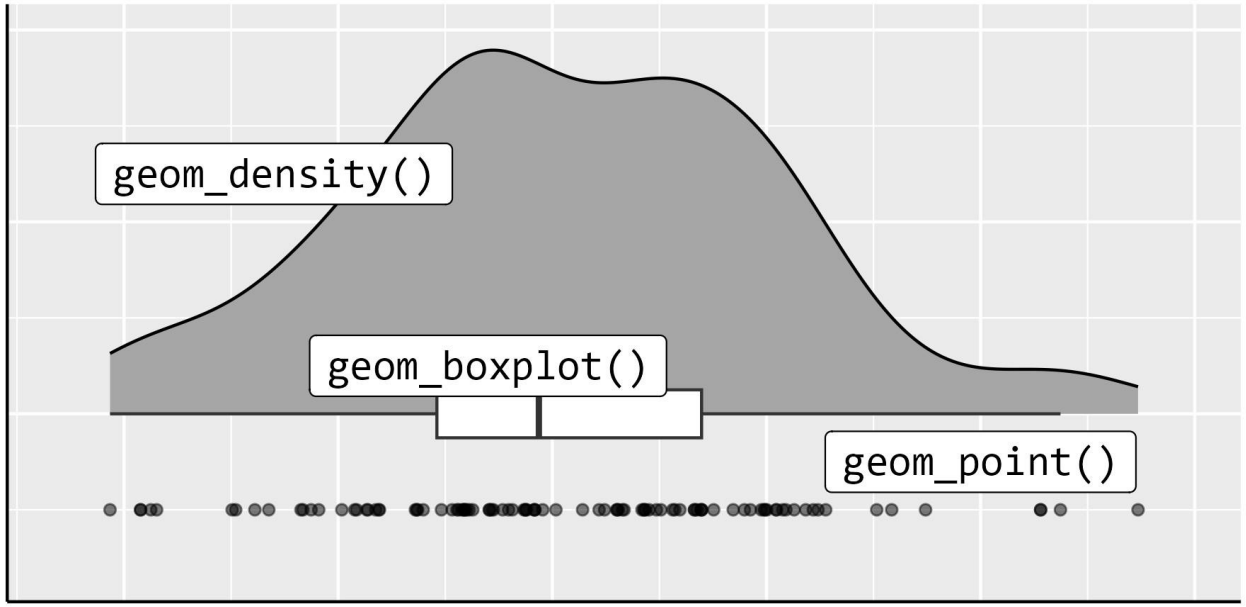
University of Pennsylvania

JSM 2025

The grammar is **composable**

Across layers: “raincloud plot”

```
ggplot(...) +
  geom_density() +
  geom_boxplot() +
  geom_point()
```



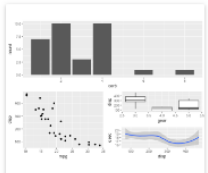
Within a layer: “Sub-layer modularity”

Wilkinson’s Grammar

- DATA
- TRANS
- ELEMENT
- GUIDE
- SCALE
- COORD

GG	ggplot2	vega-lite	seaborn
Element	Layer	Layer	Layer
	Stat	Transform	Stat
	Geom	Mark	Mark
	Position	Position	Move

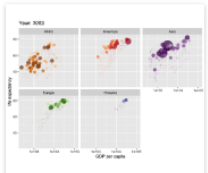
127 registered extensions available to explore



patchwork ★ 2249

Easy composition of ggplot plots using arithmetic operators

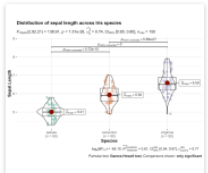
- author: thomasp85
- tags: visualization, composition
- js libraries:



gganimate ★ 1883

A Grammar of Animated Graphics.

- author: thomasp85
- tags: visualization, general
- js libraries:



ggstatsplot ★ 1726

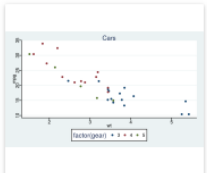
'ggstatsplot' provides a collection of functions to enhance 'ggplot2' plots with results from statistical tests.

- author: IndrajeetPatil
- tags: visualization, statistics
- js libraries:

esquisse ★ 1656

Explore and Visualize Your Data Interactively with ggplot2

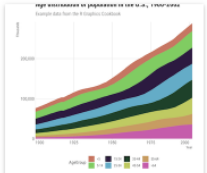
- author: dreamrs
- tags: visualization, interface
- js libraries:



ggthemes ★ 1247

Some extra geoms, scales, and themes for ggplot.

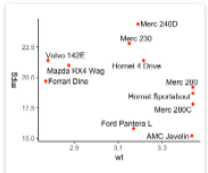
- author: jrnold
- tags: visualization, general, themes
- js libraries:



hrbrthemes ★ 1128

A compilation of extra (ggplot2) themes, scales and utilities, including a spell check function for plot label fields and an overall emphasis on typography.

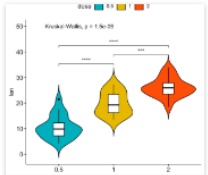
- author: hrbrmstr
- tags: theme, typography
- js libraries:



ggrepel ★ 1122

Repel overlapping text labels away from each other.

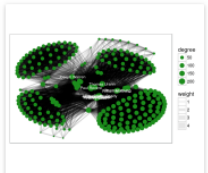
- author: slowkow
- tags: visualization, general
- js libraries:



ggpubr ★ 1018

'ggplot2' Based Publication Ready Plots

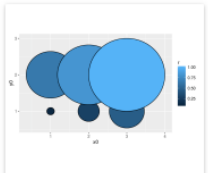
- author: kassambara
- tags: visualization, statistics
- js libraries:



ggraph ★ 1000

ggraph is tailored at plotting graph-like data structures (graphs, networks, trees, hierarchies...).

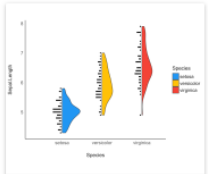
- author: thomasp85
- tags: visualization, general
- js libraries:



ggforce ★ 947

ggforce is aimed at providing missing functionality to ggplot2 through the extension system introduced with ggplot2 v2.0.0.

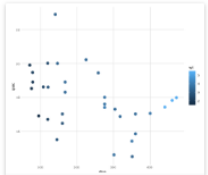
- author: thomasp85
- tags: visualization, general
- js libraries:



see ★ 727

Visualisation Toolbox for 'easystats' and Extra Geoms, Themes and Color Palettes for 'ggplot2'

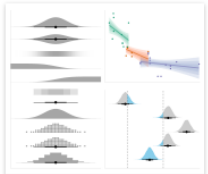
- author: easystats
- tags: visualizations, statistics
- js libraries:



ggiraph ★ 924

htmlwidget to make 'ggplot' graphics interactive.

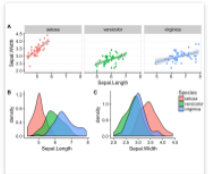
- author: davidgohel
- tags: visualization, general
- js libraries:



ggdist ★ 675

'ggdist' provides stats and geoms for visualizing distributions and uncertainty.

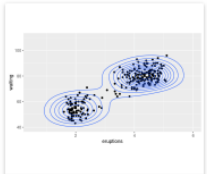
- author: mjskay
- tags: visualization, uncertainty, confidence, probability
- js libraries:



cowplot ★ 650

Streamlined plot theme and plot annotations for 'ggplot2'

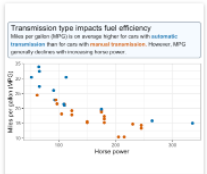
- author: clauswilke
- tags: visualization, general, themes
- js libraries:



ggalt ★ 638

A compendium of 'geoms', 'coords' and 'stats' for 'ggplot2'.

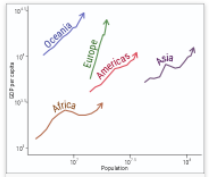
- author: hrbrmstr
- tags: visualization, general
- js libraries:



ggtext ★ 606

Improved text rendering support for ggplot2

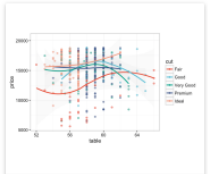
- author: Claus Wilke
- tags: general, theme, typography
- js libraries:



geomtextpath ★ 565

Create curved text and directly label lines in ggplot

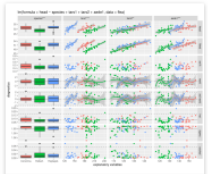
- author: AllanCameron
- tags: typography, plot-labelling, visualization
- js libraries:



ggsci ★ 562

A collection of 'ggplot2' color palettes inspired by scientific journals and science fiction TV shows.

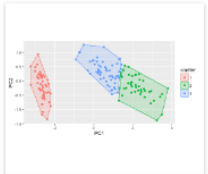
- author: road2stat
- tags: visualization, general
- js libraries:



GGally ★ 552

ggally extends 'ggplot2' by adding several functions to reduce the complexity of combining geometric objects with transformed data.

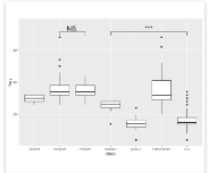
- author: ggobi
- tags: visualization, general
- js libraries:



ggfortify ★ 518

The unified interface to ggplot2 many popular statistical pakackage results.

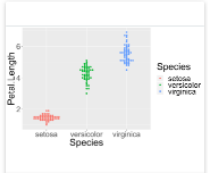
- author: terrytangyuan
- tags: visualization, general
- js libraries:



ggsignif ★ 512

Significance Brackets for 'ggplot2'.

- author: const-ae and IndrajeetPatil
- tags: visualization, multiple comparisons



ggbeeswarm ★ 501

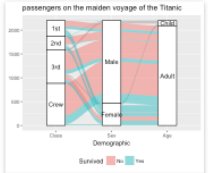
Create beeswarm plots, which avoids overlapping datapoints.

- author: Erik Clarke and Scott Sherrill-Mix

ggbump ★ 455

Bump Chart and Sigmoid Curves.

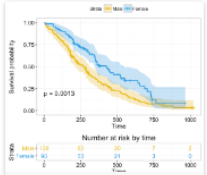
- author: davidsjoberg
- tags: visualization, general, geoms



ggalluvial ★ 451

A 'ggplot2' extension for alluvial diagrams.

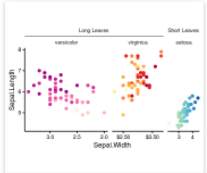
- author: corybrunson
- tags: visualization, categorical, time series



survminer ★ 438

Drawing Survival Curves using 'ggplot2'

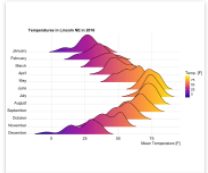
- author: kassambara
- tags: visualization, survival



ggh4x ★ 426

Options for tailored facets, multiple colourscales and miscellaneous

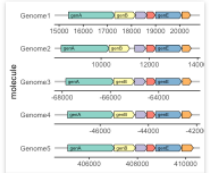
- author: teunbrand



ggridges ★ 390

Ridgeline plot geoms for 'ggplot2'

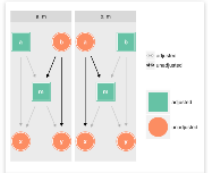
- author: clauswilke
- tags: visualization, general



gggenes ★ 330

ggplot2 geoms to draw gene arrow maps

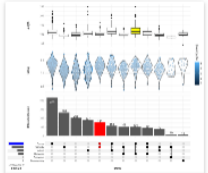
- author: wilkox
- tags: visualization, general, genetics



ggdag ★ 385

Causal directed acyclic graphs (DAGs) in ggplot2

- author: malcolmbarrett



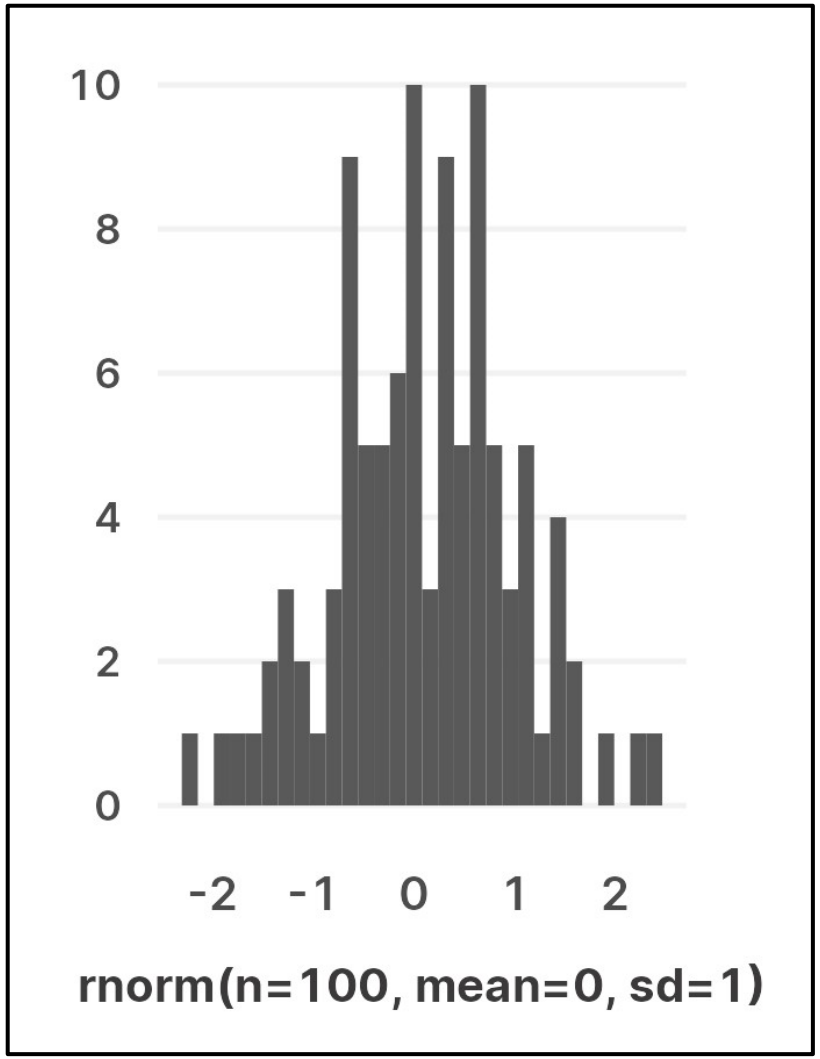
ComplexUpset ★ 378

Visualize set intersections and add ggplot2 annotations

- author: krassowski

The user experience

```
ggplot(data, aes(x)) +
  geom_histogram()
```



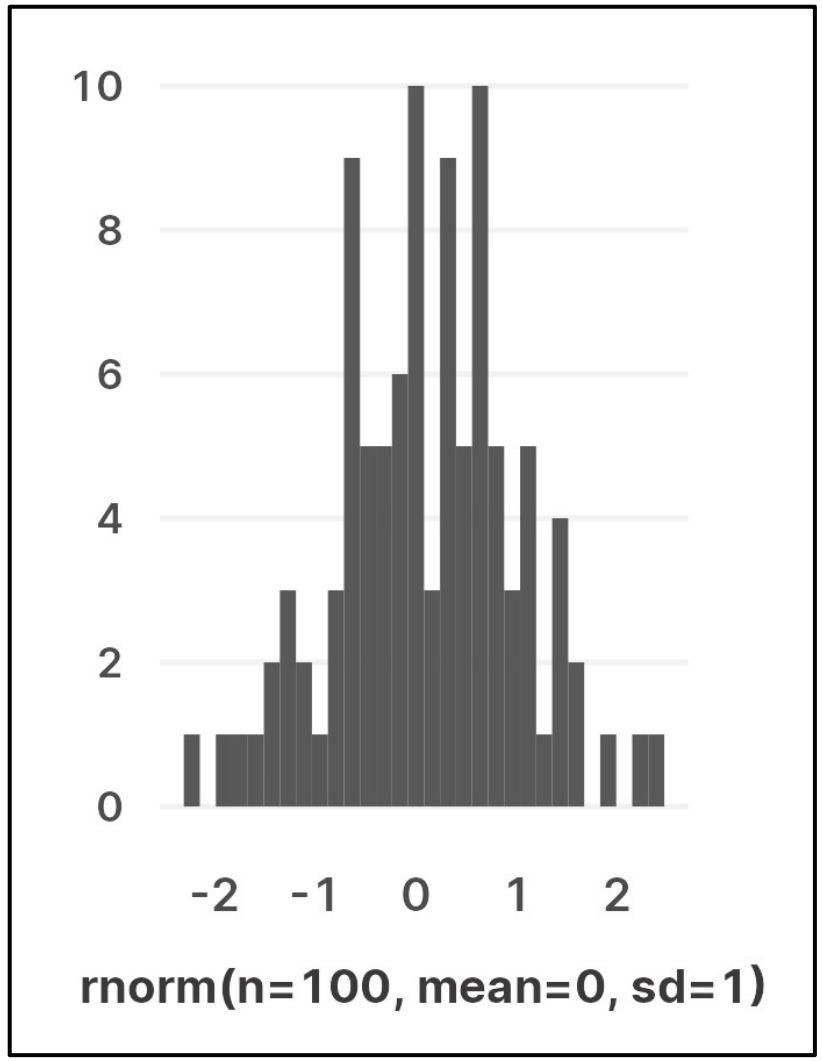
The user experience

```
ggplot(data, aes(x)) +
  geom_histogram()
```

LAYER `geom_histogram()`

STAT: `StatBin`

GEOM: `GeomBar`



The **aesthetic evaluation** semantics in modern ggplot2

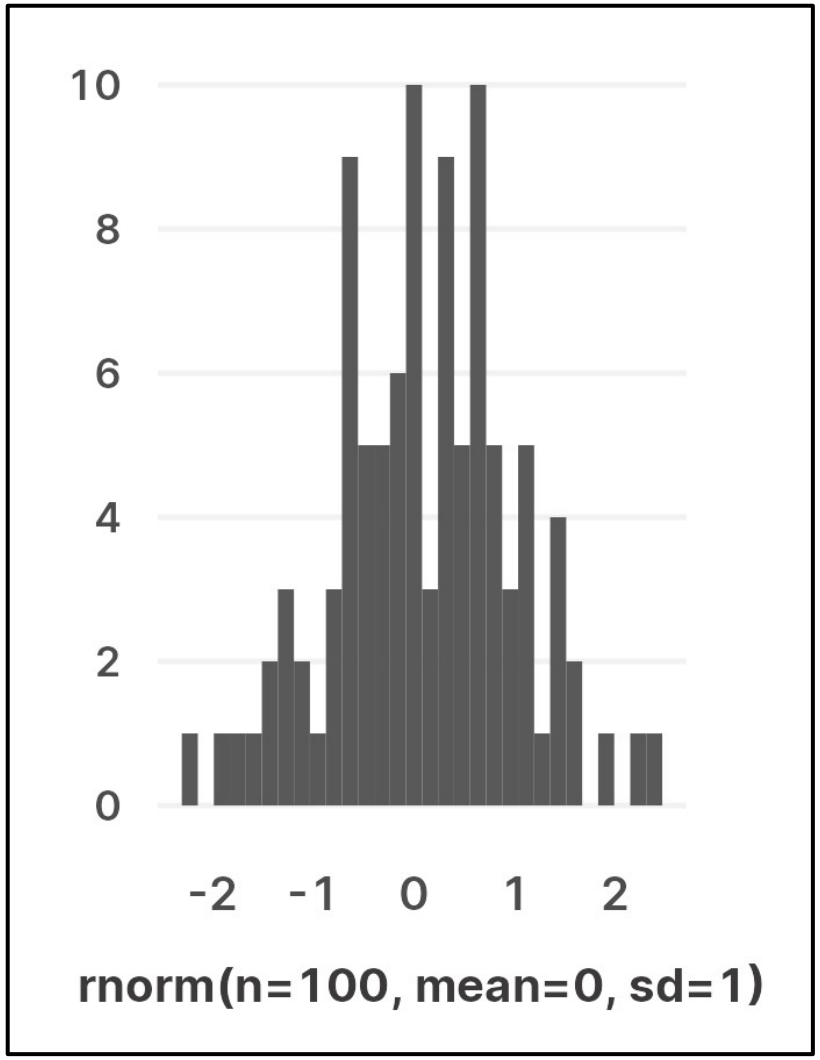
Aesthetic evaluation in modern ggplot2

Functions that let users reference **internal snapshots** of a **layer's data**:

- `after_stat()`
- `after_scale()`
- `stage()`

The user experience

```
ggplot(data, aes(x)) +
  geom_histogram()
```

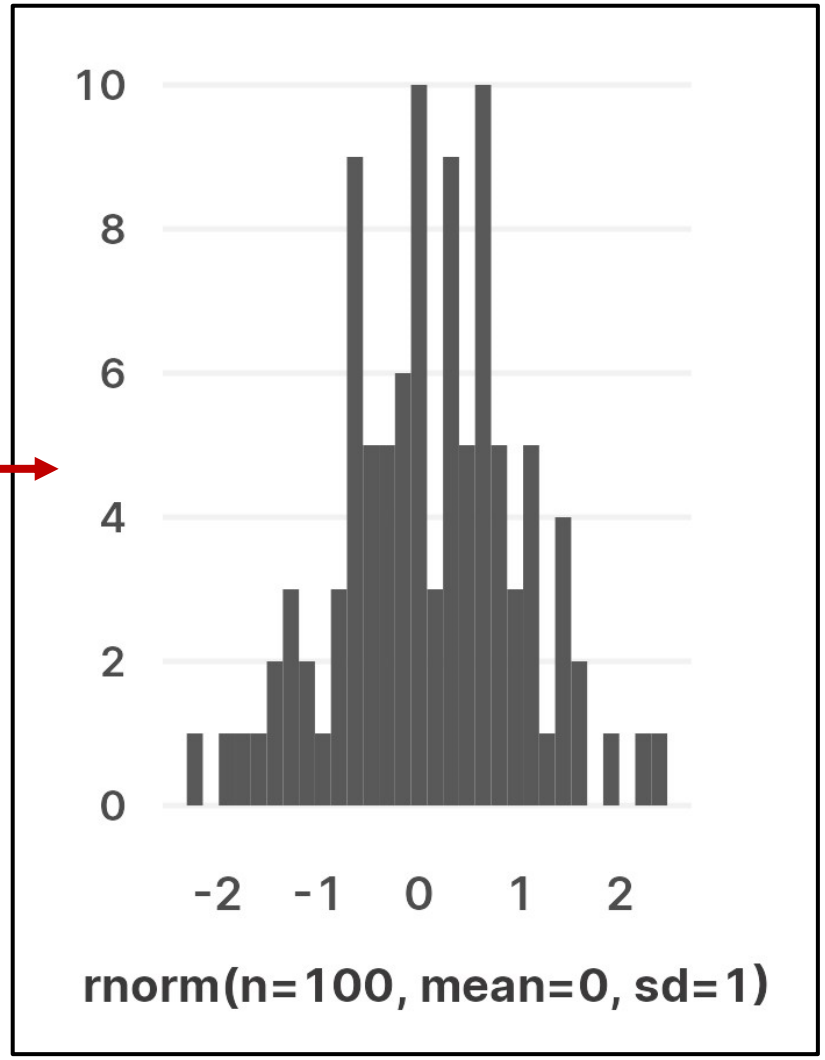


A piece of the grammar is missing from the code

The user experience

```
ggplot(data, aes(x)) +
  geom_histogram()
```

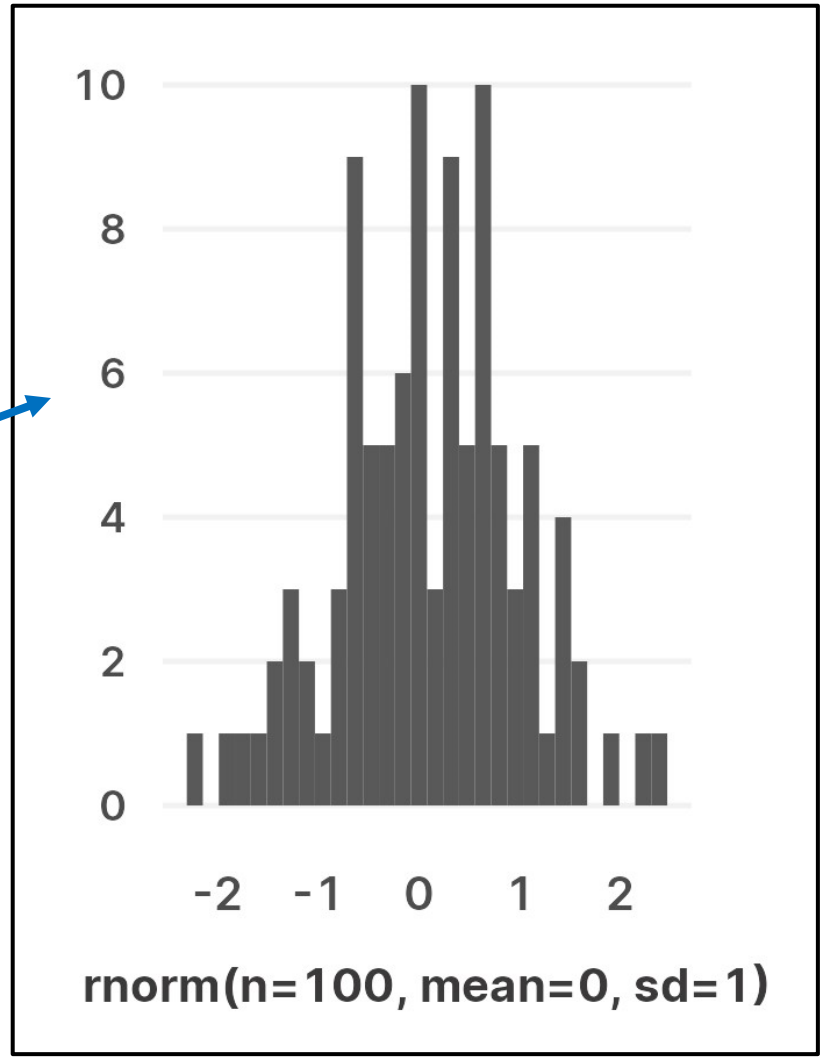
?



A piece of the grammar is missing from the code: No data is mapped to the **y** aesthetic

The user experience

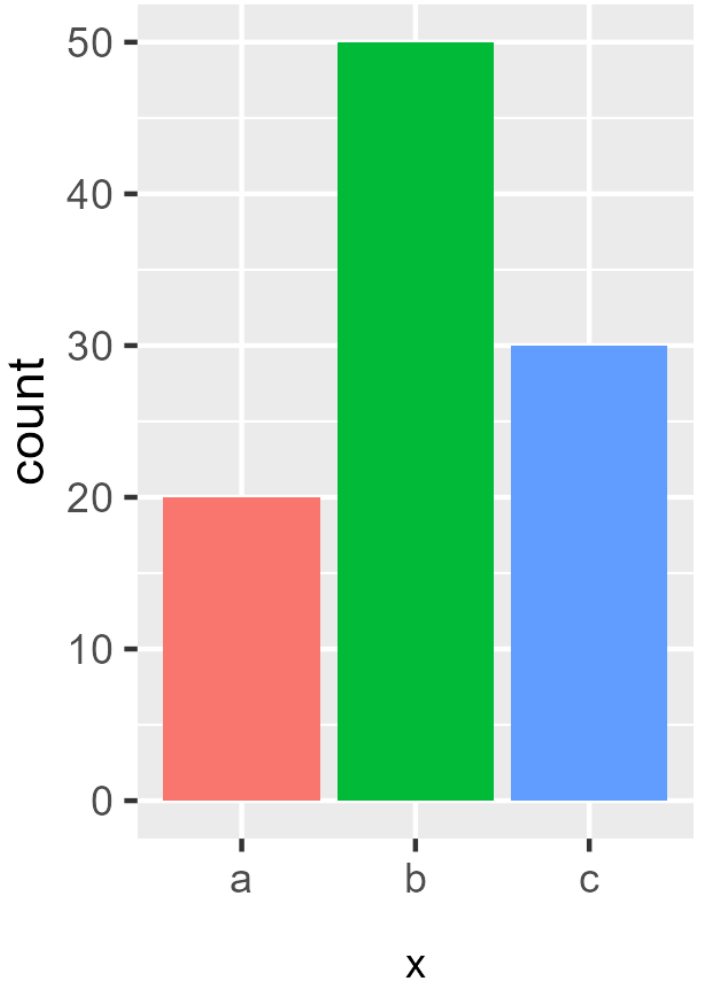
```
ggplot(data, aes(x)) +
  geom_histogram(
    aes(y = after_stat(count))
  )
```



Functions like `after_stat` fill that gap, in a way that is **available to the user**.

The evaluation semantics

```
geom_bar(
   aes(y = after_stat(count))
)
```



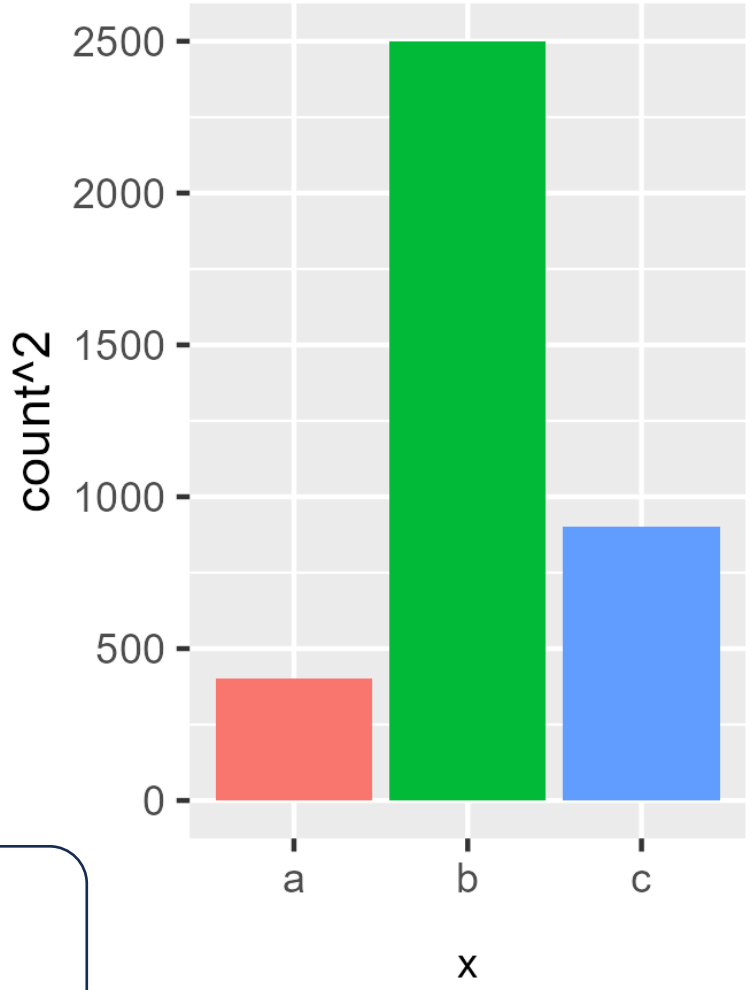
LAYER `geom_bar()`

STAT: `StatCount`

GEOM: `GeomBar`

The evaluation semantics

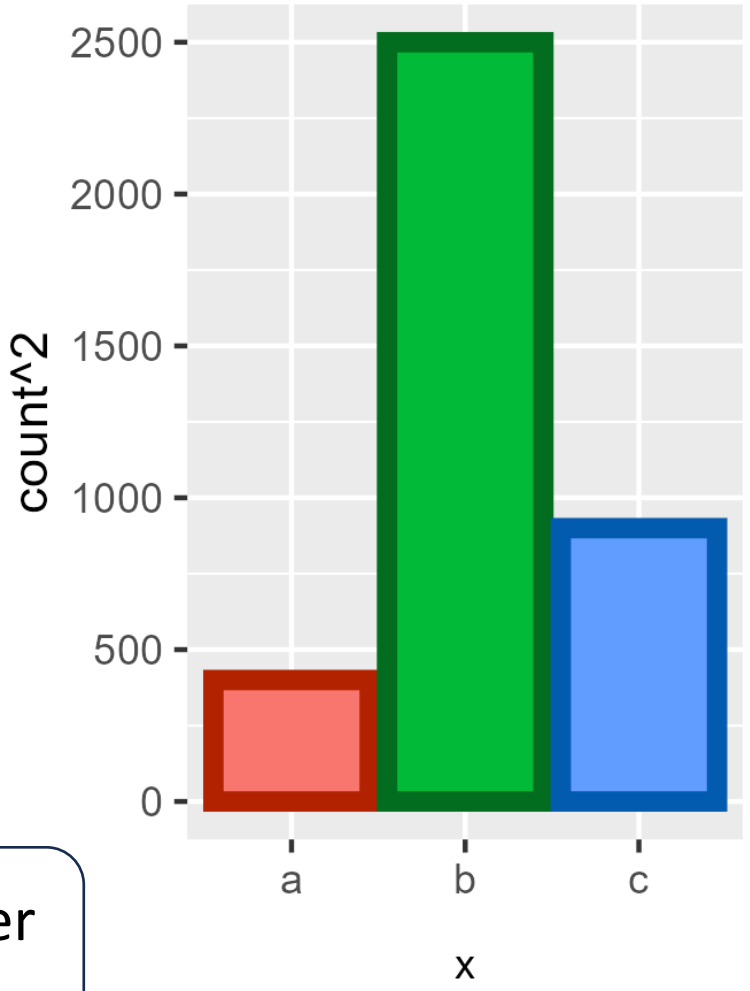
```
geom_bar(
   aes(y = after_stat(count^2))
)
```



"Take the **`count`** variable when it's available after the layer data's **STATistical transformation**, and square it"

The evaluation semantics

```
geom_bar(
  aes(
    y = after_stat(count^2)
    color = after_scale(darken(fill))
  )
)
```



"Take the **`fill`** variable when it's available after the layer data's **SCALE transformation**, and darken it"

The evaluation semantics

```
geom_bar(
  aes(
    y = after_stat(print(count))
    color = after_scale(browser())
  )
)
```

```
[1] 20 50 30
```

```
Browse[1]> fill
[1] "#F8766D" "#00BA38" "#619CFF"
```

Interim summary

What is special about ggplot's design?

A rich aesthetic mapping interface that exposes some of the intermediate stages of a layer's data to the user

Why does it allow users to do?

- Intervene in the process of how a layer materializes from the data
- Build intuitions about the internals and start thinking like a developer

Case study:

Annotating a boxplot variable

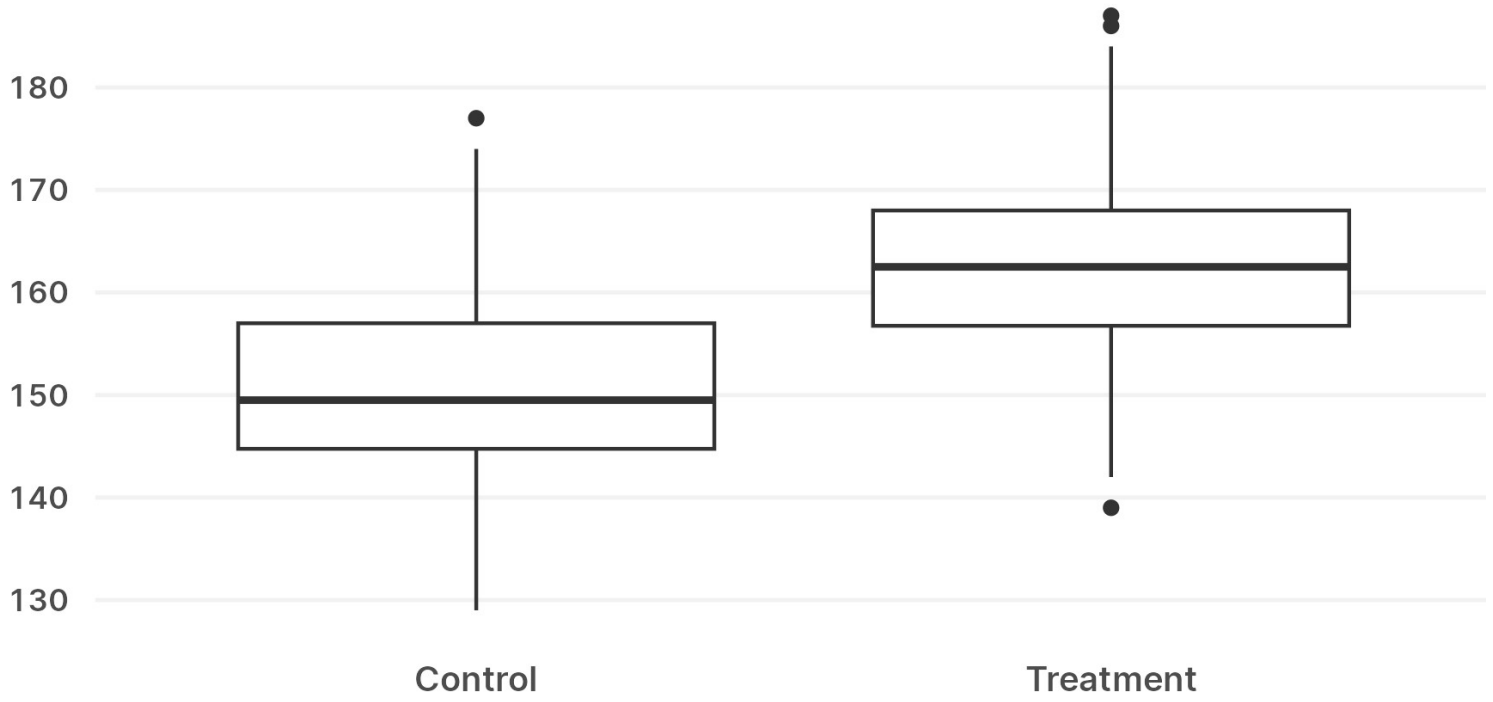
The goal

Draw a boxplot for a data with a discrete x and a continuous y:

```
data
#>    xvar     yvar
#> 1 Control   149
#> 2 Control   140
#> 3 Control   131
#> 4 Control   148
#> 5 Control   144
#> # ℹ 195 more rows
```

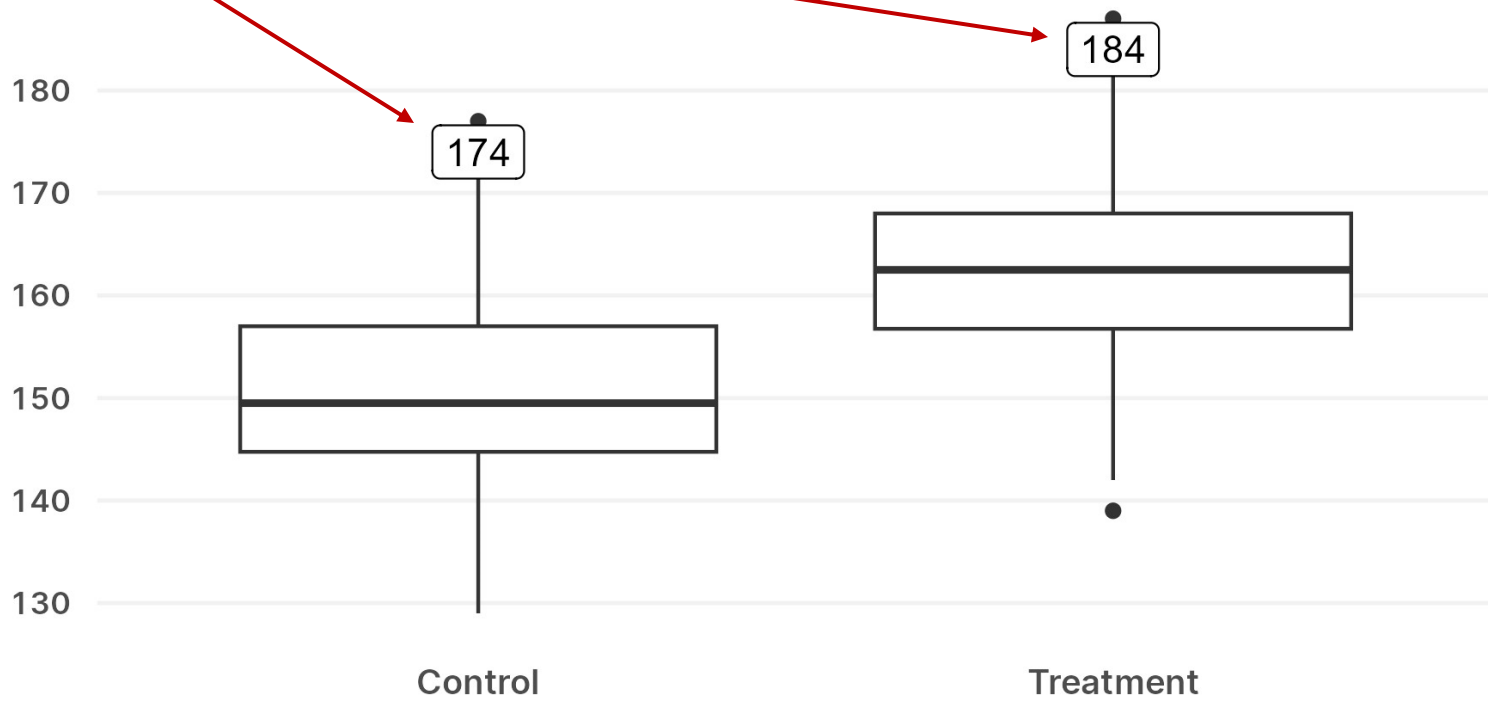
The boxplot

```
box_plot <- ggplot(data, aes(x = xvar, y = yvar)) +
  geom_boxplot()
```

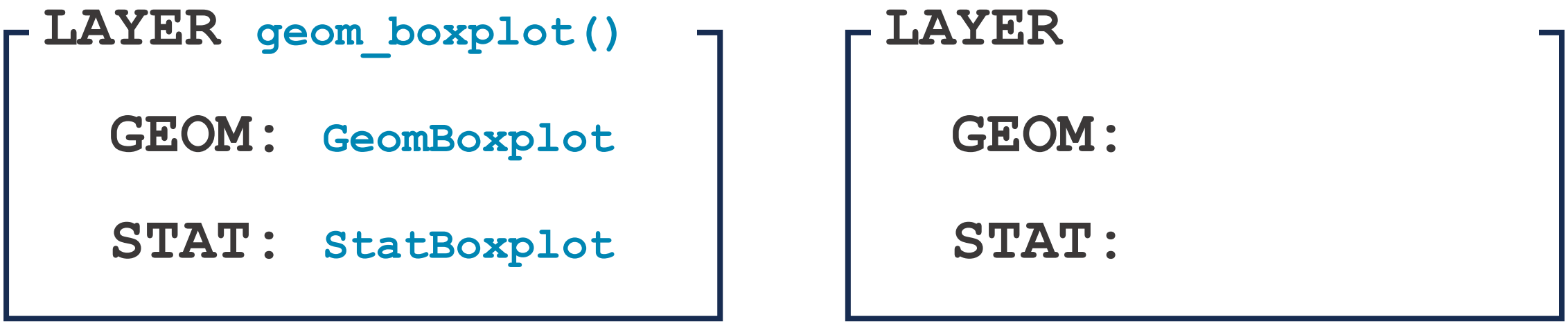


A problem: annotating upper whiskers

```
box_plot +
   LAYER
```



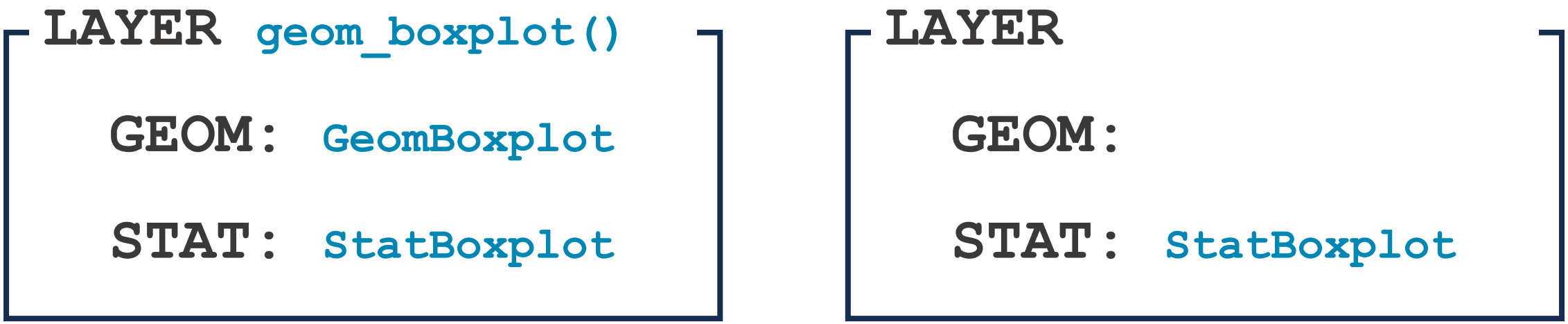
Deriving the annotation layer



First layer (boxplot)

Second layer (annotation)

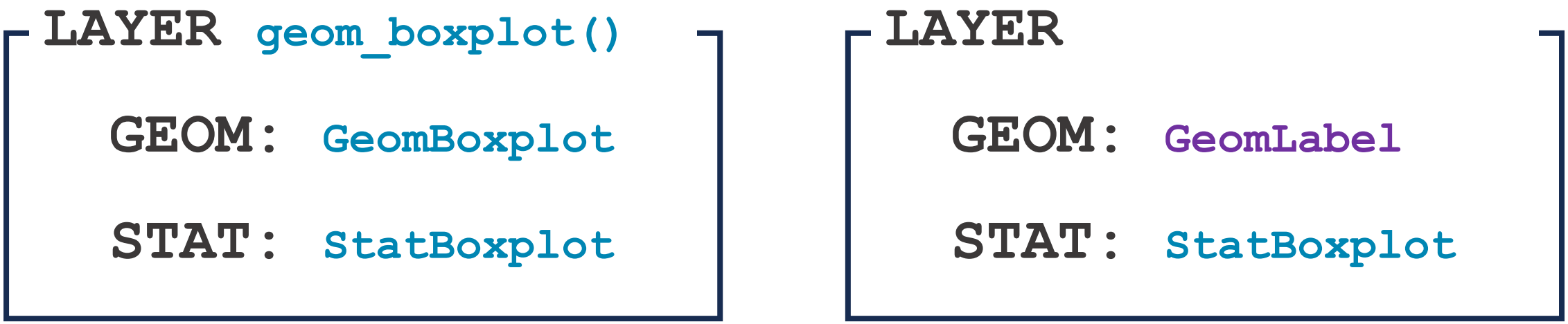
Deriving the annotation layer



First layer (boxplot)

Second layer (annotation)

Deriving the annotation layer



First layer (boxplot)

Second layer (annotation)

Deriving the annotation layer

```
box_plot +
   geom_label(stat = StatBoxplot)
```

```
#> Error in `geom_label()`:
#> ! Problem while setting up geom.
#> ℹ Error occurred in the 2nd layer.
#> Caused by error in `compute_geom_1()`:
#> ! `geom_label()` requires the following
missing aesthetics: y and label
```

Deriving the annotation layer

```
box_plot +
  geom_label(stat = StatBoxplot)
```

User thinking:

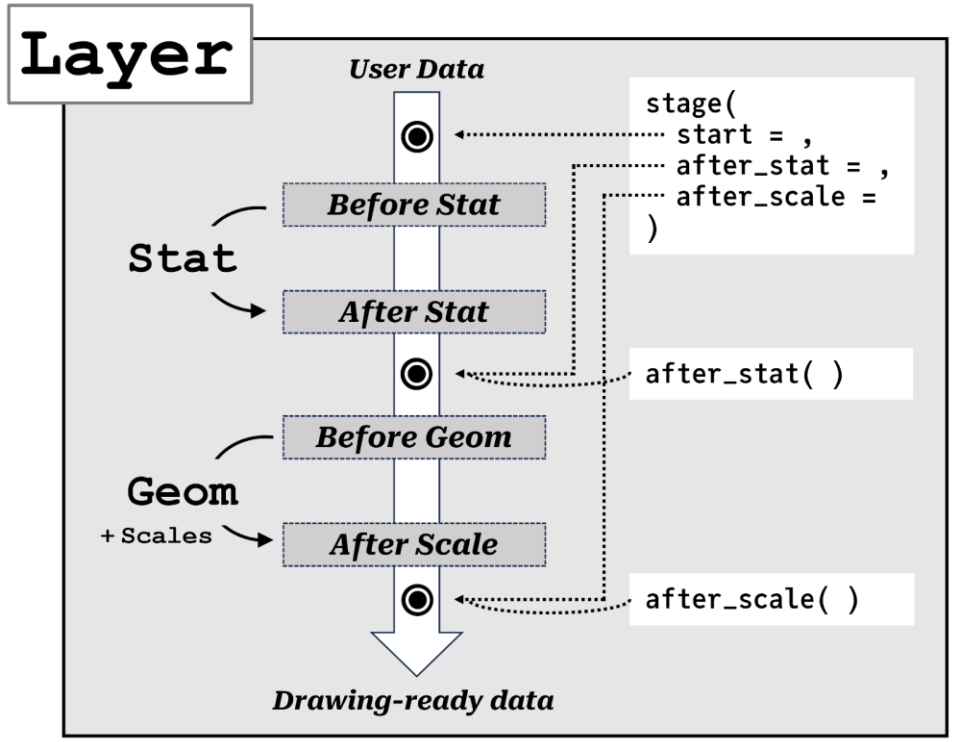
"I should calculate these values manually myself."

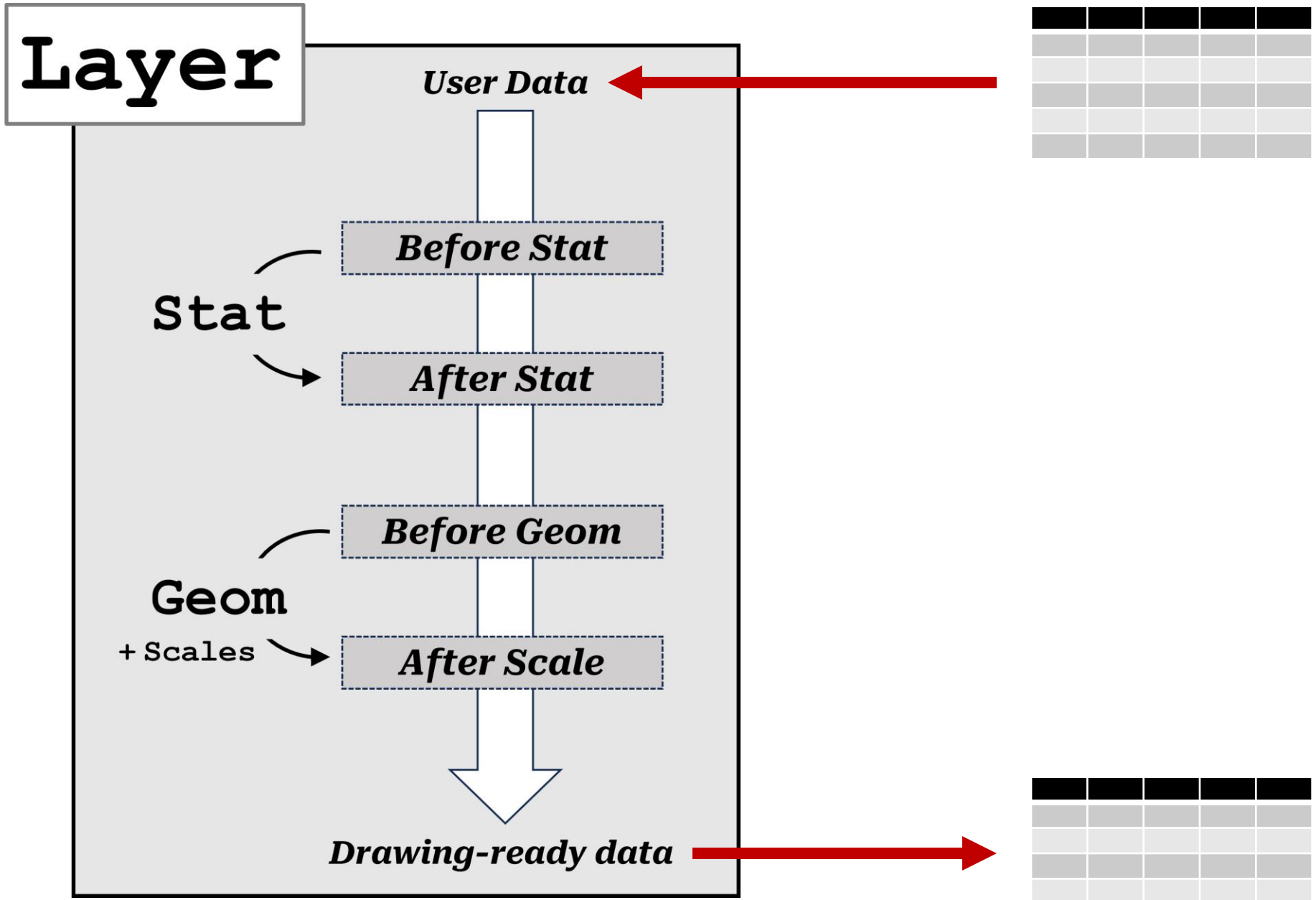
Developer thinking:

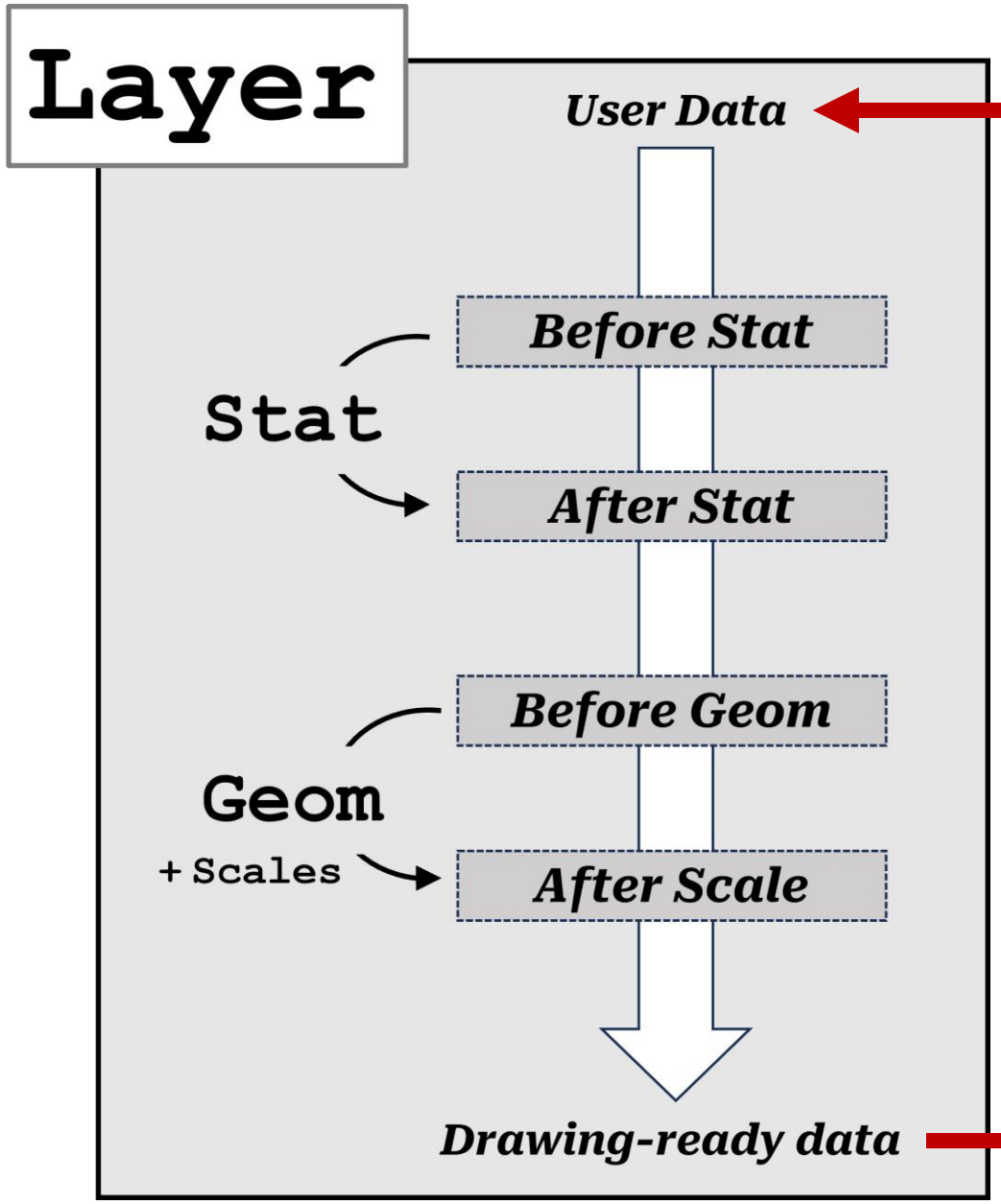
"How do I access the values that the boxplot stat must have already computed?"

Requirements for developer thinking

1) Basic data wrangling skills for **tabular data**
2) A mental model of sub-layer processes as a **data pipeline**

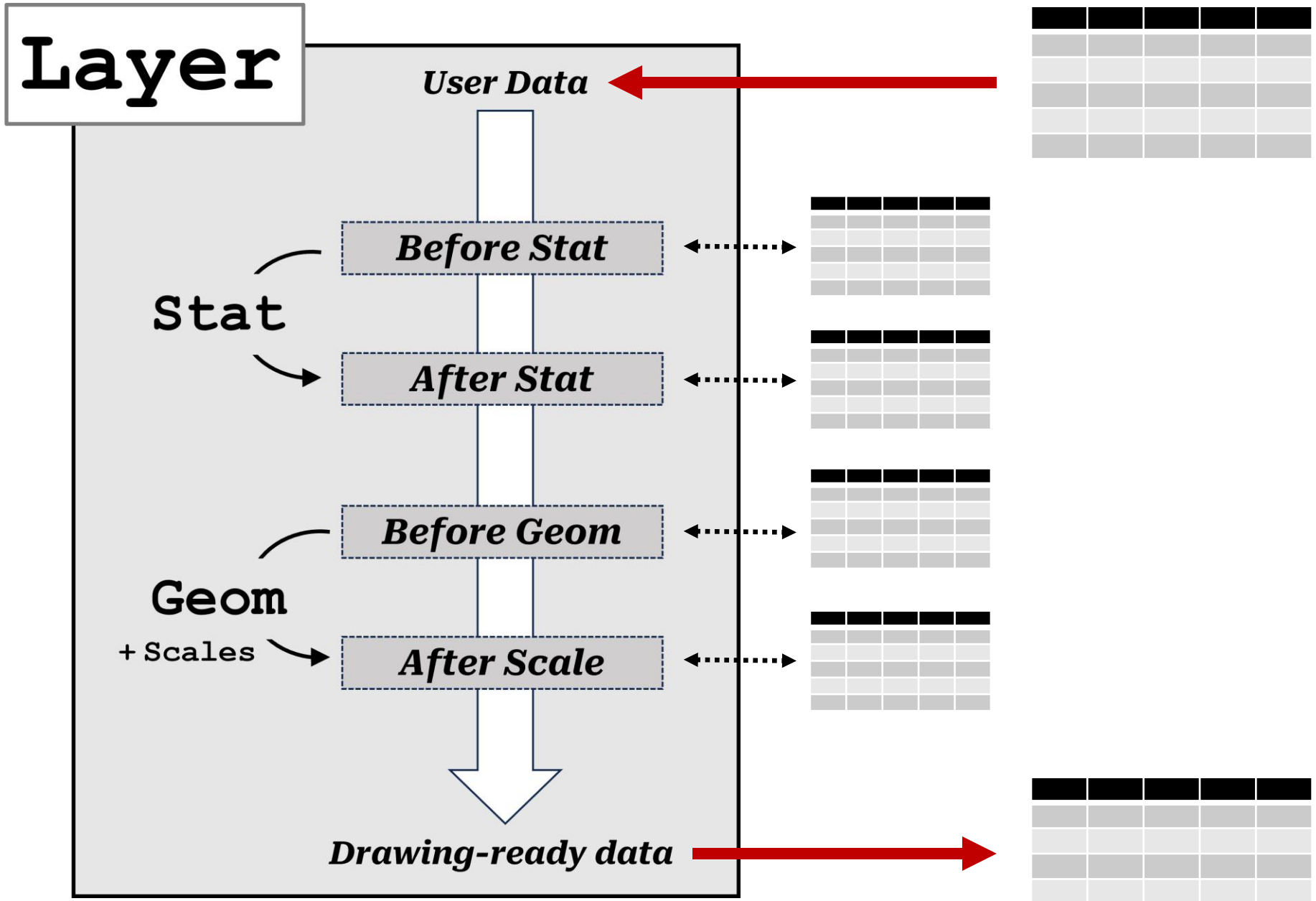


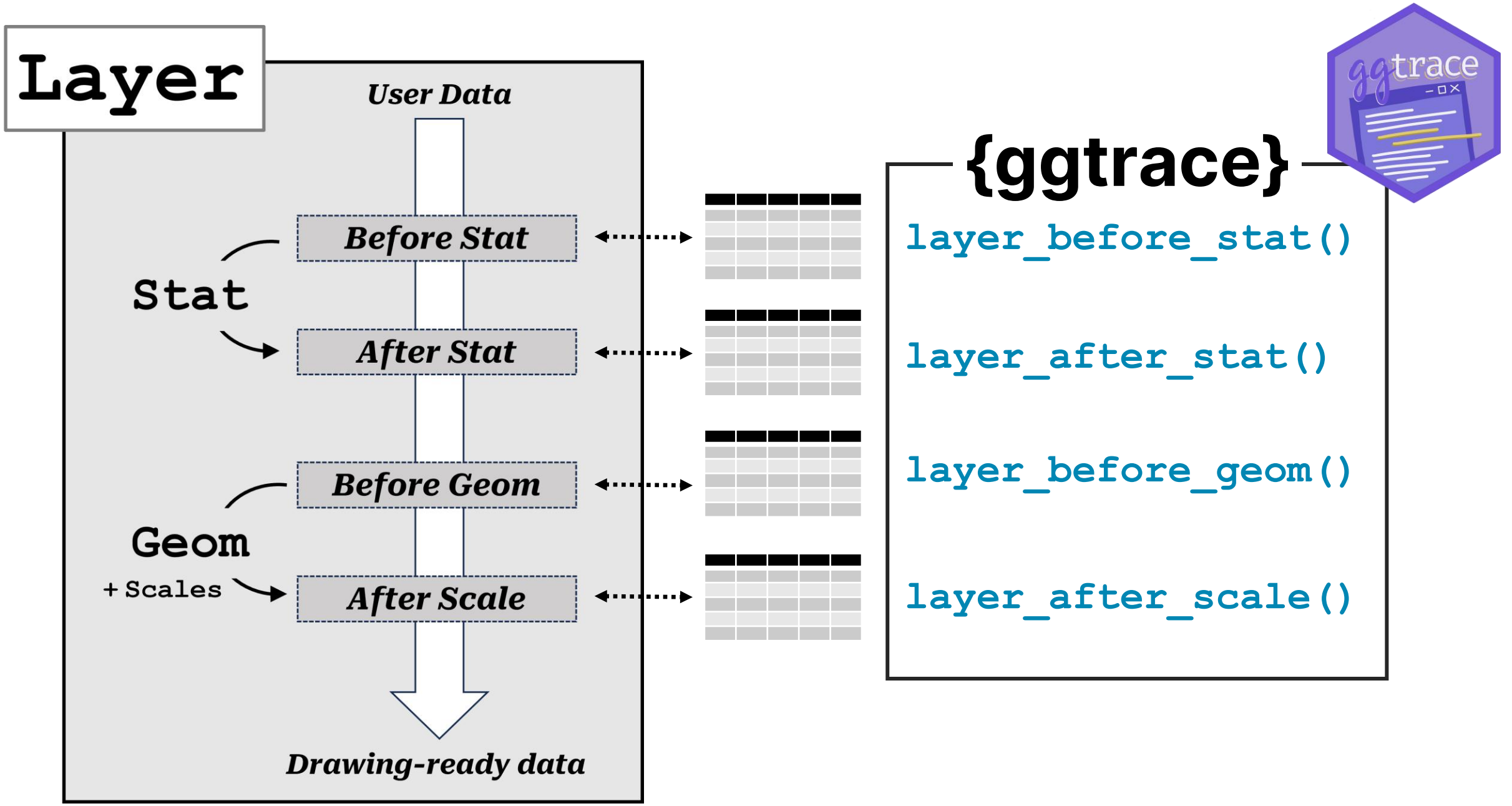

Layer
User Data
Before Stat
Stat
After Stat
Before Geom
Geom
+ Scales
After Scale
Drawing-ready data

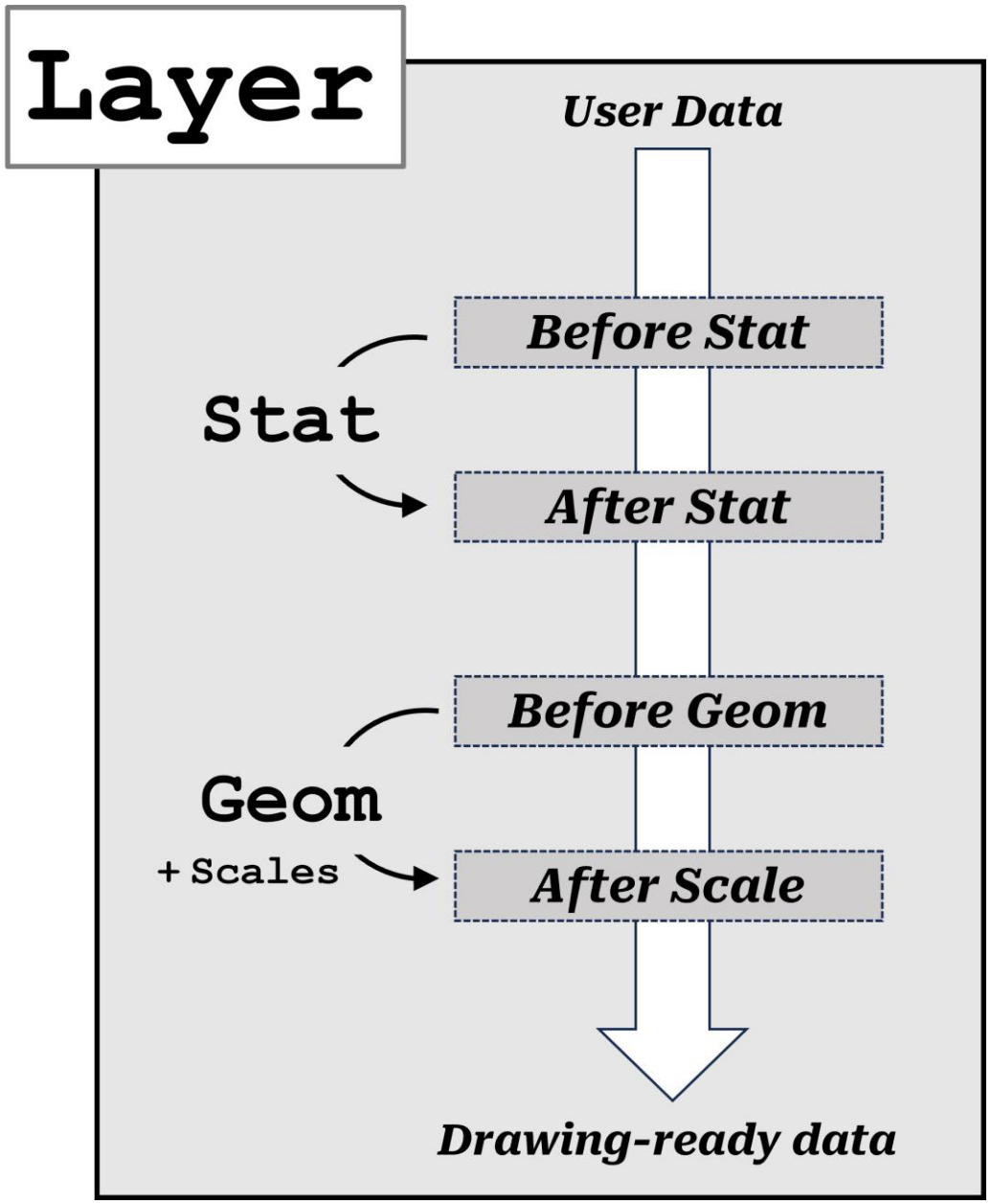



```
data
#>    xvar     yvar
#> 1 Control   149
#> 2 Control   140
#> 3 Control   131
#> 4 Control   148
#> 5 Control   144
#> # i 195 more rows
```

```
ggplot2::layer_data(box_plot, i = 1)
#>    ymin  lower middle upper ymax ...
#> 1   129 144.75  149.5   157  174 ...
#> 2   142 156.75  162.5   168  184 ...
```


Layer
User Data
Before Stat
Stat
After Stat
Before Geom
Geom
+ Scales
After Scale
Drawing-ready data


Layer
User Data
Before Stat
Stat
After Stat
Before Geom
Geom
+ Scales
After Scale
Drawing-ready data
{ggtrace}
layer_before_stat()
layer_after_stat()
layer_before_geom()
layer_after_scale()

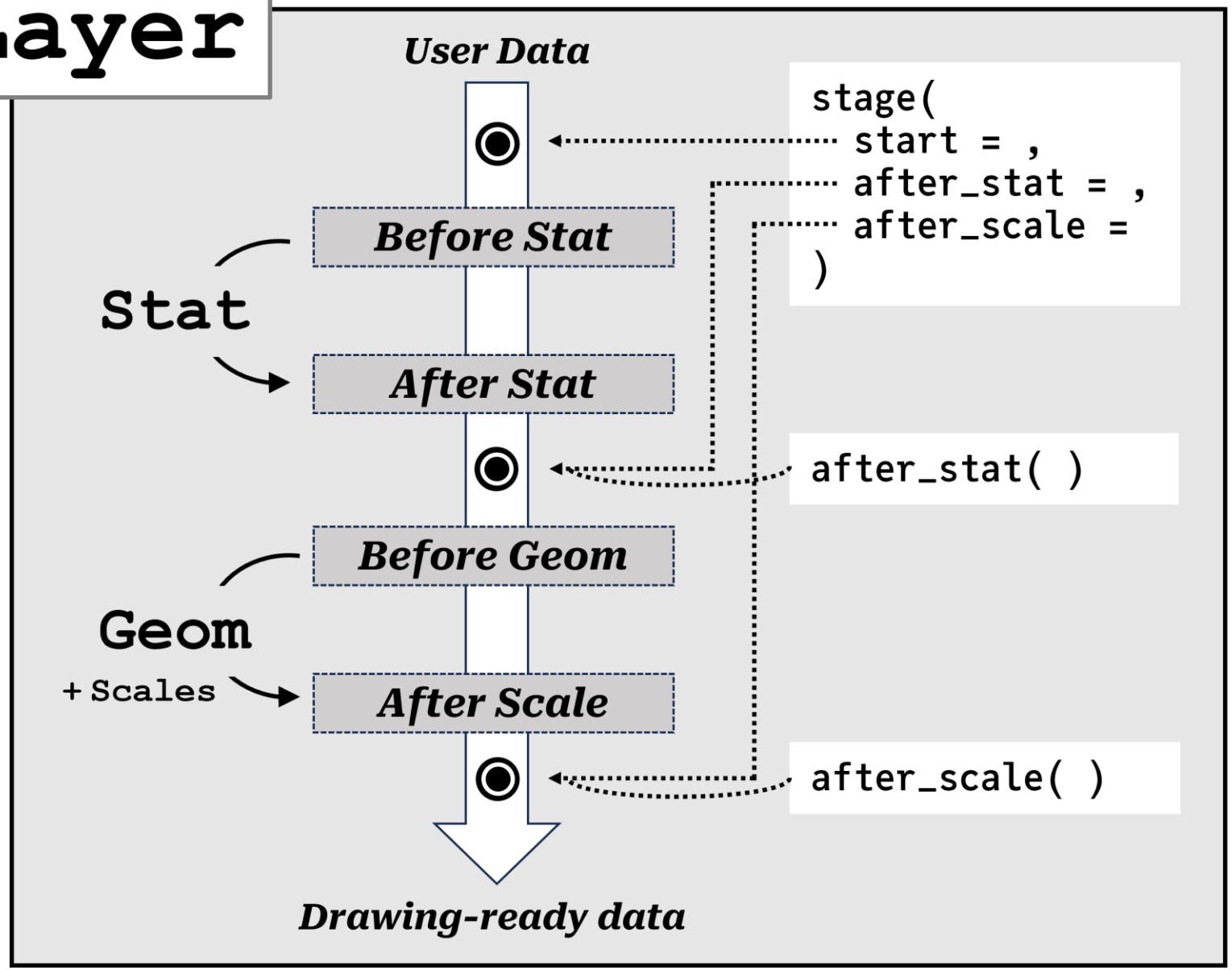



```
after_stat()

after_scale()

stage()
```

Layer


User Data
stage(
start = ,
after_stat = ,
after_scale =
)
Before Stat
Stat
After Stat
after_stat( )
Before Geom
Geom
+ Scales
After Scale
after_scale( )
Drawing-ready data

Debugging the annotation layer

```
box_plot +
  geom_label(stat = StatBoxplot)
```

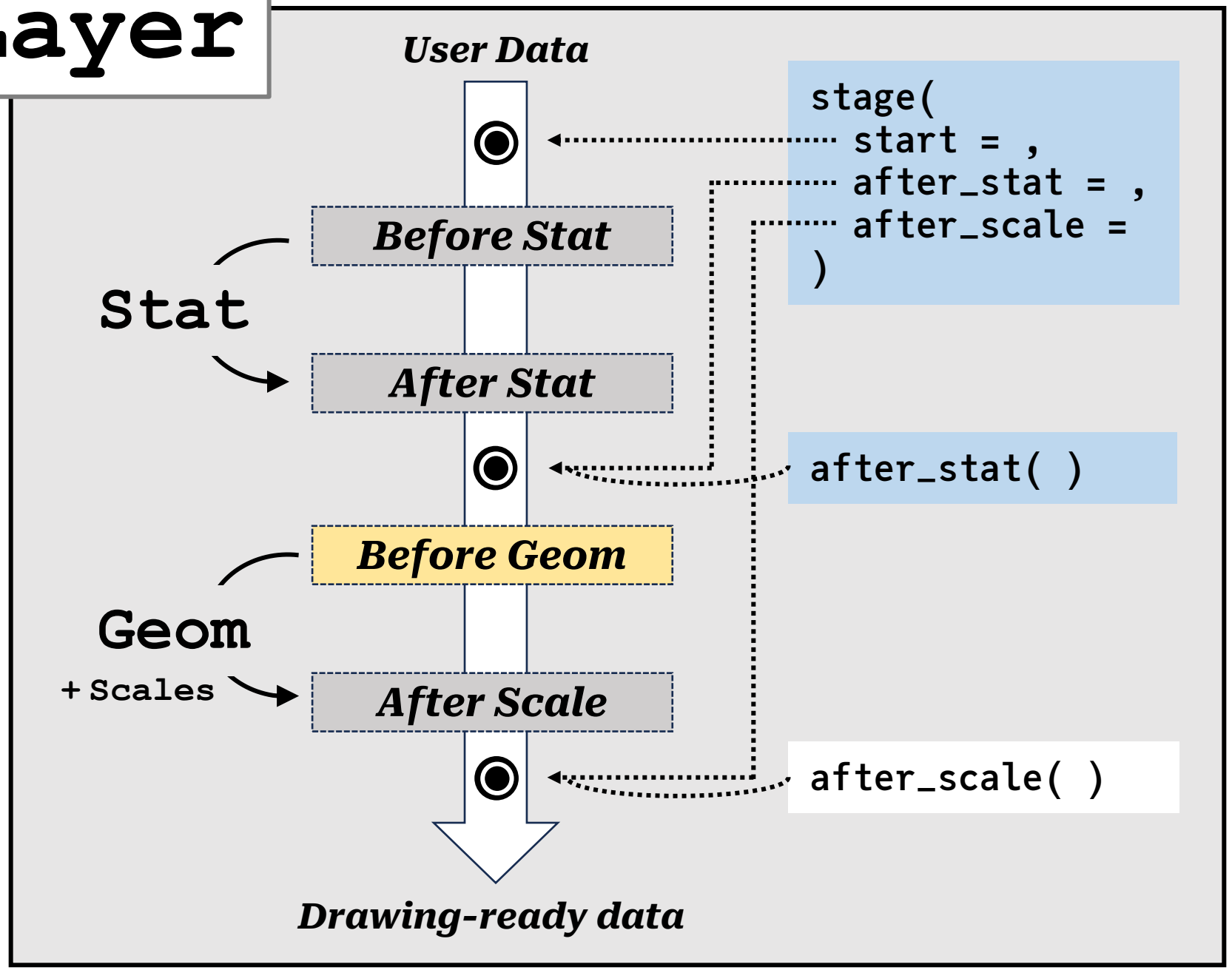
```
#> Error in `geom_label()`:
#> ! Problem while setting up geom.
#> ℹ Error occurred in the 2nd layer.
#> Caused by error in `compute_geom_1()`:
#> ! `geom_label()` requires the following missing aesthetics: y and label
```

Debugging the annotation layer

Something is wrong with the **dataframe** that the **Geom** receives as **input**

I must ensure that **columns `y`** and **`label`** are present by the time the Geom receives the data.

Layer



1) See what is going on in the **Before Geom** snapshot of the data

2) Use **`stage()`** and/or **`after_stat()`** to ensure **`y`** and **`label`** are present

Before Geom of the defective layer

```
ggtrace::layer_before_geom(box_plot, i = 2, error = TRUE)
```

```
#>   ymin  lower middle upper ymax      outliers notchupper notchlower
#> 1  129 144.75  149.5   157  174           177   151.4355   147.5645
#> 2  142 156.75  162.5   168  184 139, 186, 187   164.2775   160.7225
#>   x width relvarwidth flipped_aes PANEL group
#> 1 1  0.75          10       FALSE     1     1
#> 2 2  0.75          10       FALSE     1     2
```

Before Geom of the defective layer

```
ggtrace::layer_before_geom(box_plot, i = 2, error = TRUE)
```

```
#>   ymin  lower middle upper ymax      outliers notchupper notchlower
#> 1  129 144.75  149.5   157  174           177   151.4355   147.5645
#> 2  142 156.75  162.5   168  184 139, 186, 187   164.2775   160.7225
#>   x width relvarwidth flipped_aes PANEL group
#> 1 1  0.75          10       FALSE     1     1
#> 2 2  0.75          10       FALSE     1     2
```

Fix: Two mappings (= column modifications) in the after-stat stage:

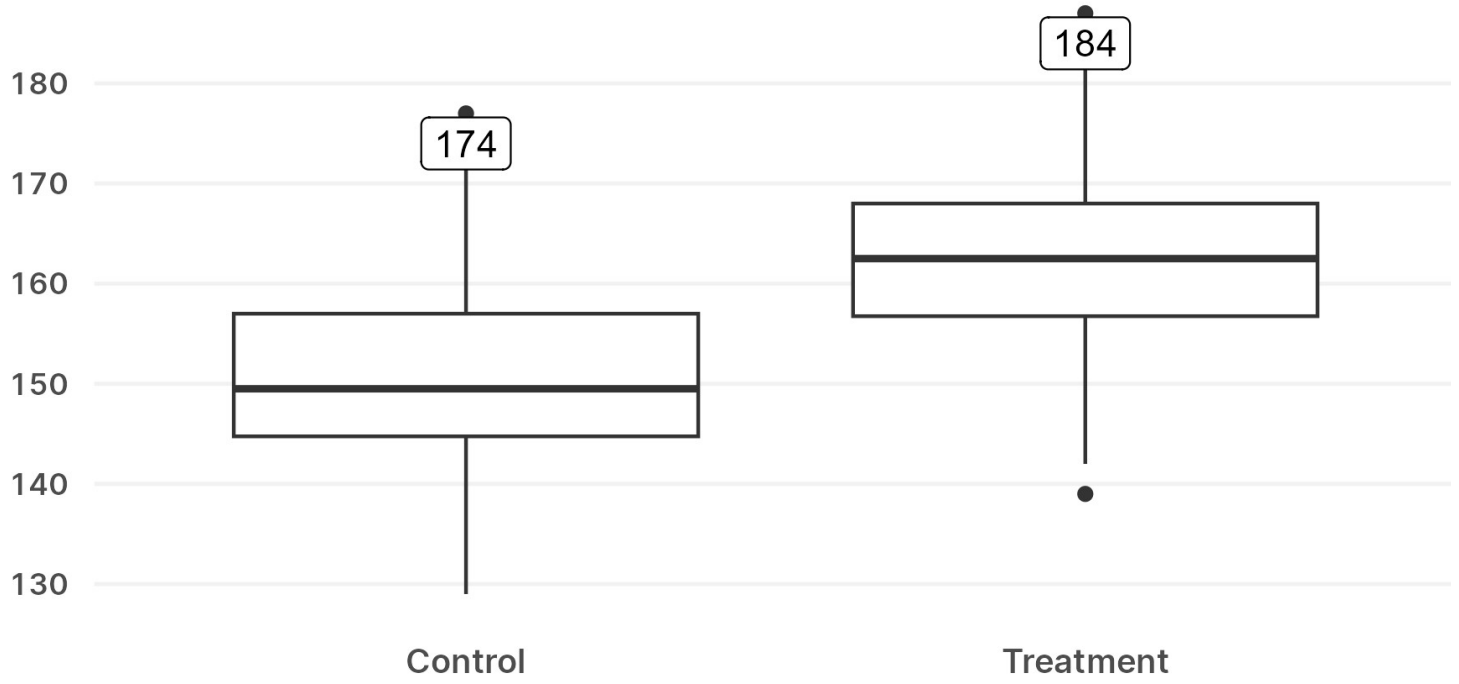
- **`y = ymax`**
- **`label = ymax`**

Annotation layer

```
geom_label(
  aes(y = stage(start = yvar, after_stat = ymax),
      label = after_stat(ymax))
  stat = StatBoxplot
)
```

"In the layer data pipeline, use the column **`yvar`** to map to **`y`** at the start. Then, after the **`STAT`** transforms data, use the computed variable **`ymax`** to re-map to **`y`** and map to **`label`**."

```
box_plot2 <- box_plot +
  geom_label(
    aes(y = stage(start = yvar, after_stat = ymax),
        label = after_stat(ymax))
    stat = StatBoxplot
  )
```



The fixed layer's Before Geom data

```
ggtrace::layer_before_geom(box_plot2, i = 2)
```

```
#>     y label ymin  lower middle upper ymax      outliers notchupper
#> 1 174   174  129 144.75  149.5   157  174           177   151.4355
#> 2 184   184  142 156.75  162.5   168  184 139, 186, 187   164.2775
#>   notchlower x width relvarwidth flipped_aes PANEL group
#> 1   147.5645 1  0.75          10       FALSE     1     1
#> 2   160.7225 2  0.75          10       FALSE     1     2
```

Conclusion

ggplot’s unique innovations in aesthetic evaluation semantics empower users to see and touch internal processes

We can leverage this design to not only write more powerful user code, but also to start thinking like a developer

More at: **github.com/yjunechoe/ggtrace**