

What do I mean by tidy data?

Data are often stored in **tabular** (or matrix) form:

```
1 library(palmerpenguins)
2 penguins |> slice(1:5)
```

A tibble: 5 × 8

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g
	<fct>	<fct>	<dbl>	<dbl>	<int>	<int>
1	Adelie	Torgersen	39.1	18.7	181	3750
2	Adelie	Torgersen	39.5	17.4	186	3800
3	Adelie	Torgersen	40.3	18	195	3250
4	Adelie	Torgersen	NA	NA	NA	NA
5	Adelie	Torgersen	36.7	19.3	193	3450

i 2 more variables: sex <fct>, year <int>

Each row == an observation

↑
each column is
variable / measurement

The Grammar of Graphics

Originally defined by Leland Wilkinson

1. **data**
2. **geometries**: type of geometric objects to represent data, e.g., points, lines
3. **aesthetics**: visual characteristics of geometric objects to represent data, e.g., position, size
4. **scales**: how each aesthetic is converted into values on the graph, e.g., color scales
5. **stats**: statistical transformations to summarize data, e.g., counts, means, regression lines
6. **facets**: split data and view as multiple graphs
7. **coordinate system**: 2D space the data are projected onto, e.g., Cartesian coordinates

Hadley Wickham created `ggplot2`

1. **data**
2. **geom_** *X geom-bar geom-point*
3. **aes**: mappings of columns to geometric objects
4. **scale**: one scale for each **aes** variable
5. **stat** *scale = size = manual()*
6. **facet**
7. **coord**
8. **labs**: labels/guides for each variable and other parts of the plot, e.g., title, subtitle, caption
9. **theme**: customization of plot layout

Bar charts

```
1 library(tidyverse)
2 penguins |>
3   ggplot(aes(x = species)) +
4   geom_bar()
```



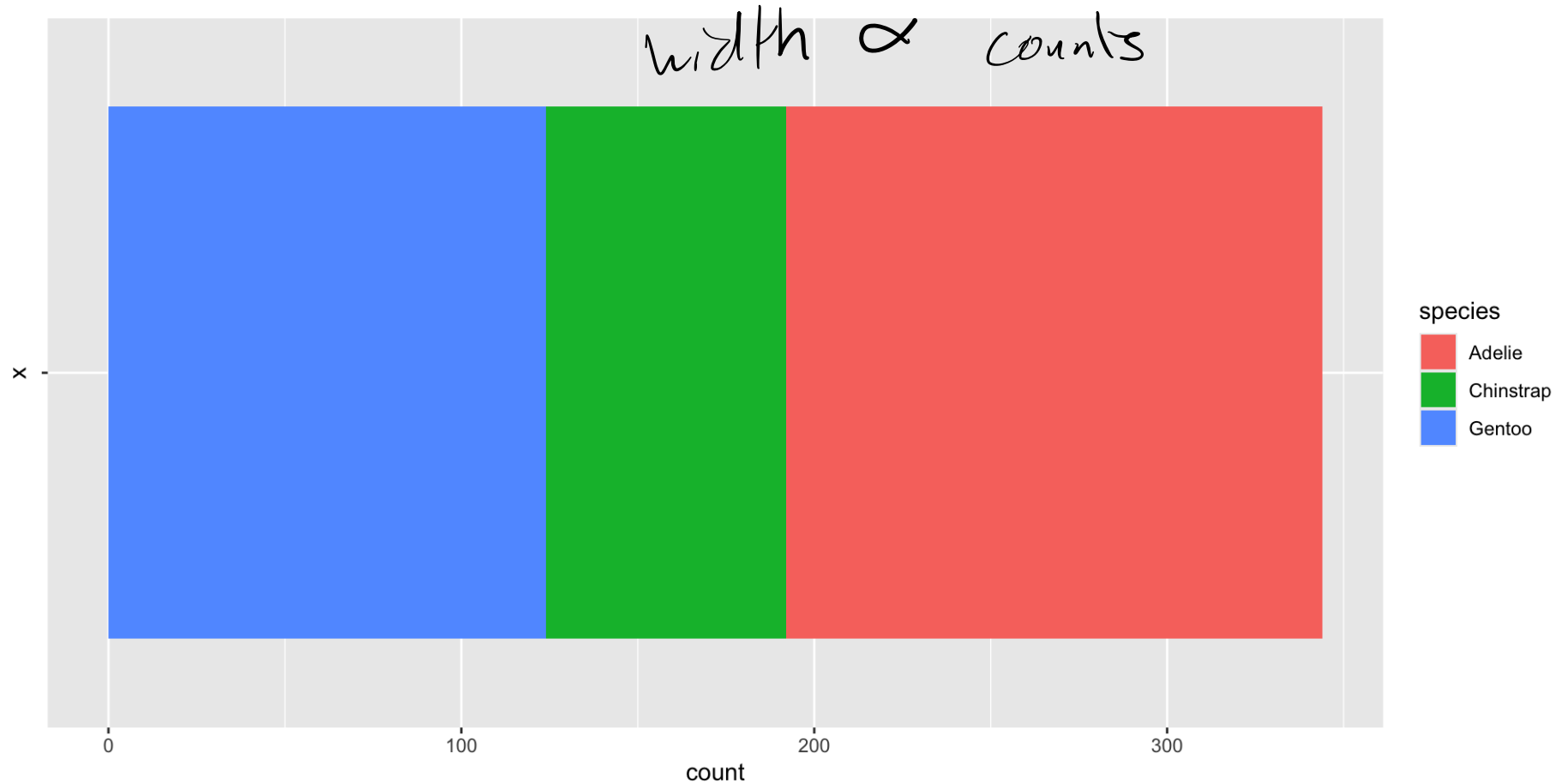
Spine charts - height version

```
1 penguins |>  
2   ggplot(aes(fill = species, x = "")) +  
3   geom_bar()
```



Spine charts - width version

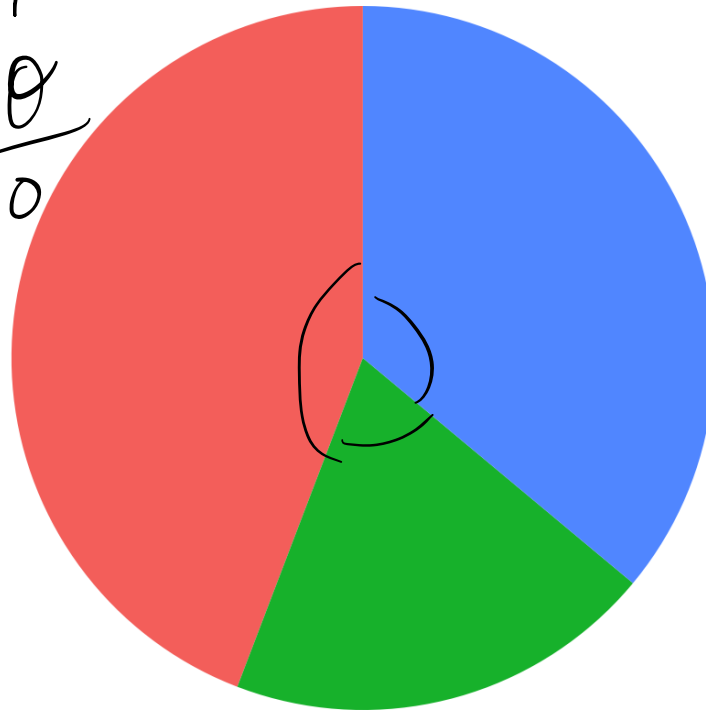
```
1 penguins |>  
2   ggplot(aes(fill = species, x = "")) +  
3   geom_bar() +  
4   coord_flip()
```



So you want to make pie charts...

```
1 penguins |>
2   ggplot(aes(fill = species, x = "")) +
3   geom_bar(aes(y = after_stat(count))) +
4   coord_polar(theta = "y") +
5   theme_void()
```

Total area = πr^2
slice area: $\frac{\pi r^2 \cdot \theta}{360}$
 $\theta \propto \text{counts}$
radius $\propto 1$

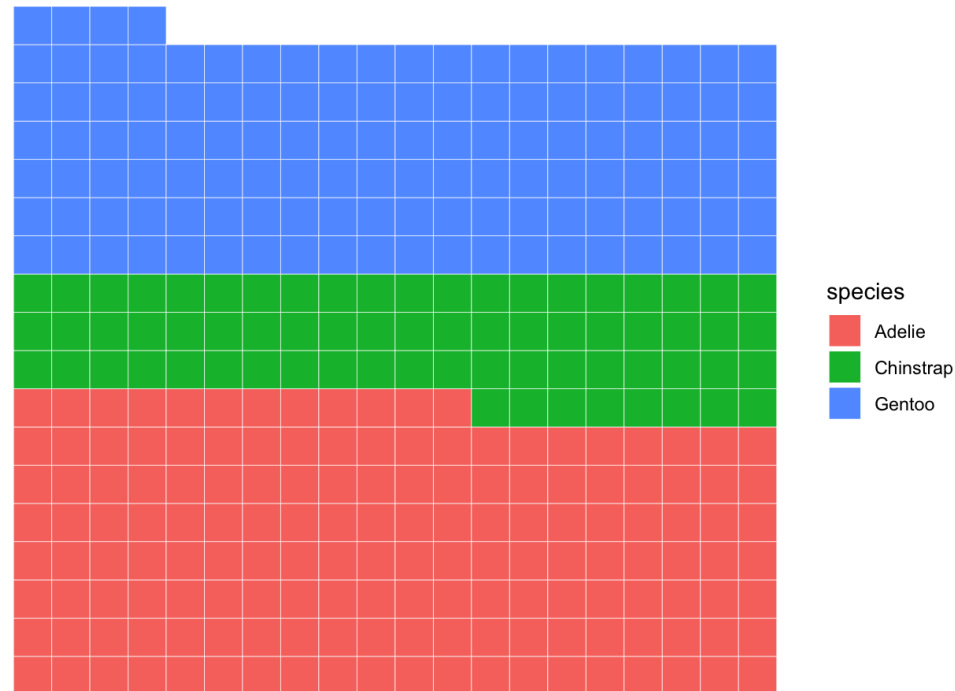


species

- Adelie
- Chinstrap
- Gentoo

Waffle charts are cooler anyway...

```
1 library(waffle)
2 penguins |>
3   group_by(species) |>
4   summarize(count = n(), .groups = "drop") |>
5   ggplot(aes(fill = species, values = count)) +
6   geom_waffle(n_rows = 20, color = "white", flip = TRUE) +
7   coord_equal() +
8   theme_void()
```



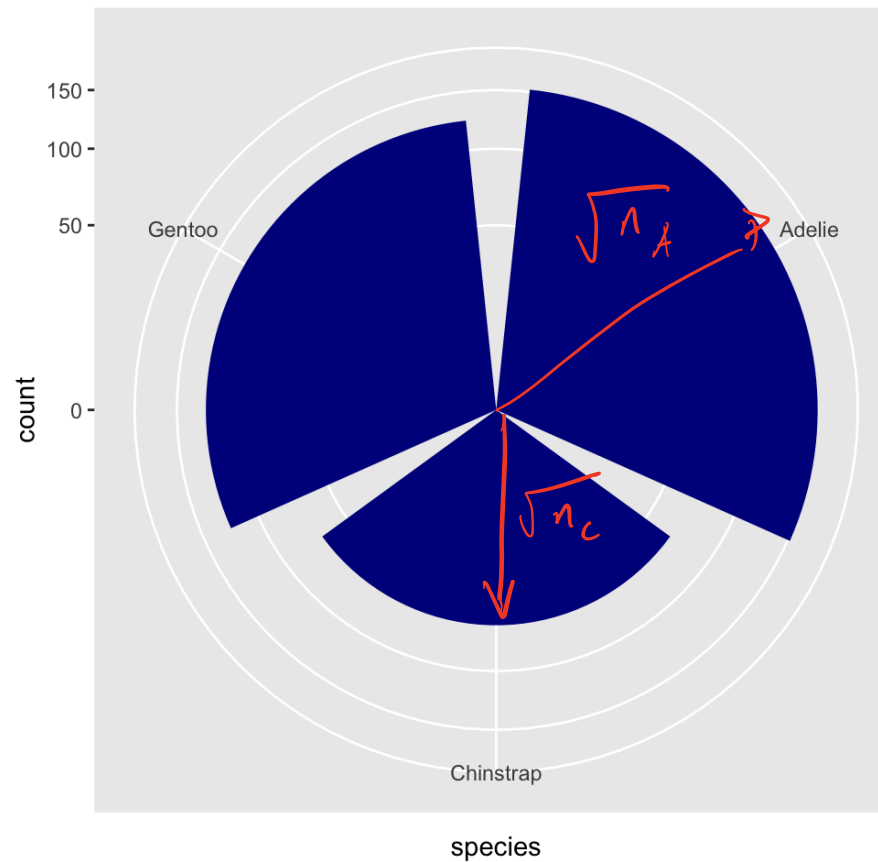
Rose diagrams

```
1 penguins |>  
2   ggplot(aes(x = species)) +  
3   geom_bar(fill = "darkblue") +  
4   coord_polar() +  
5   scale_y_sqrt()
```

$\theta \propto 1$
Area $\propto r^2$. $\theta \propto (\text{counts})^2$
if $r = \text{count}$

$r = \sqrt{\text{count}}$

$\Rightarrow \text{Area} \propto \text{counts}$

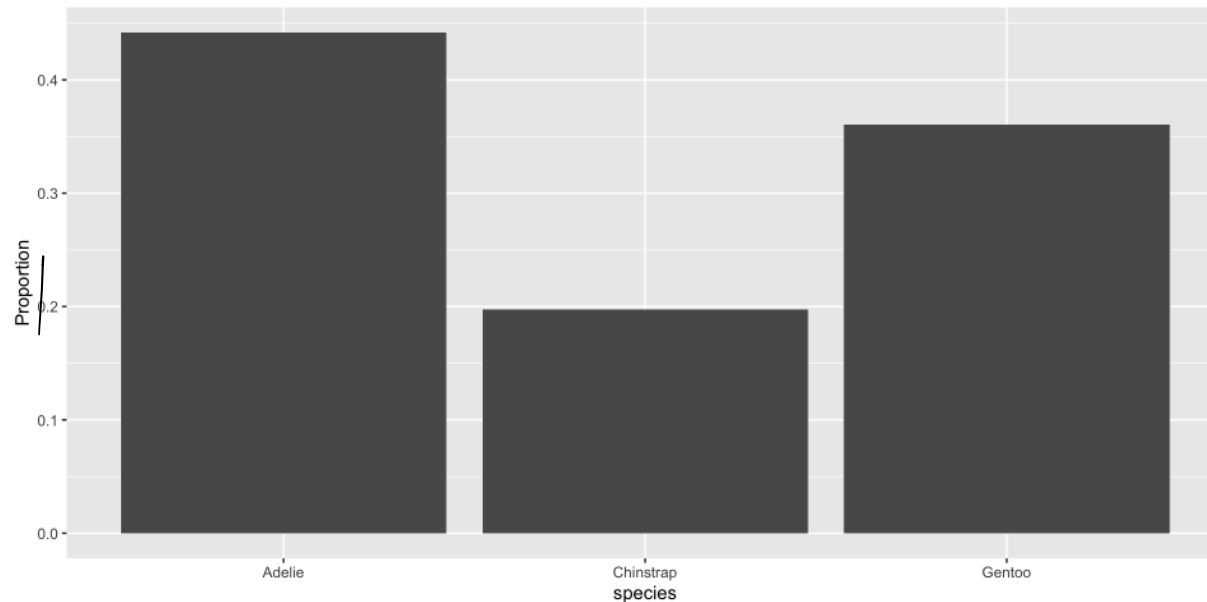


Bar charts with proportions

- `after_stat()` indicates the aesthetic mapping is performed after statistical transformation
- Use `after_stat(count)` to access the `stat_count()` called by `geom_bar()`

```
1 penguins |>
2   ggplot(aes(x = species)) +
3   geom_bar(aes(y = after_stat(count) / sum(after_stat(count)))) +
4   labs(y = "Proportion")
```

Handwritten note: `nrow(penguins)` is written above the denominator `sum(after_stat(count))` in the code, indicating that the denominator represents the total number of rows in the data frame.



Compute and display the proportions directly

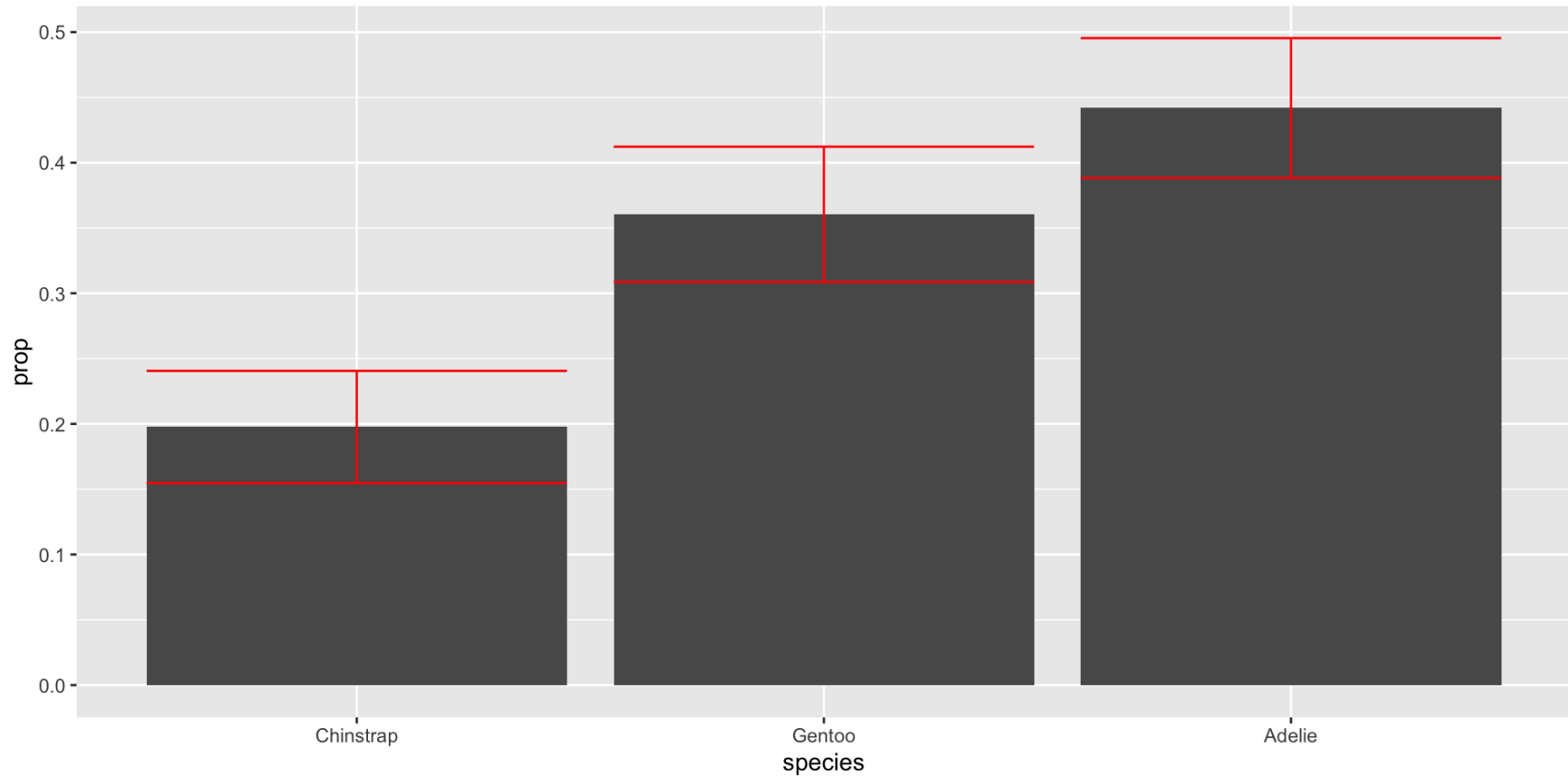
- Use `group_by()`, `summarize()`, and `mutate()` in a pipeline to compute then display the proportions directly
- Need to indicate we are displaying the `y` axis as given, i.e., the identity function

```
1 penguins |>
2   group_by(species) |>
3   summarize(count = n(), .groups = "drop") |>
4   mutate(total = sum(count),
5           prop = count / total) |>
6   ggplot(aes(x = species)) +
7   geom_bar(aes(y = prop), stat = "identity")
```

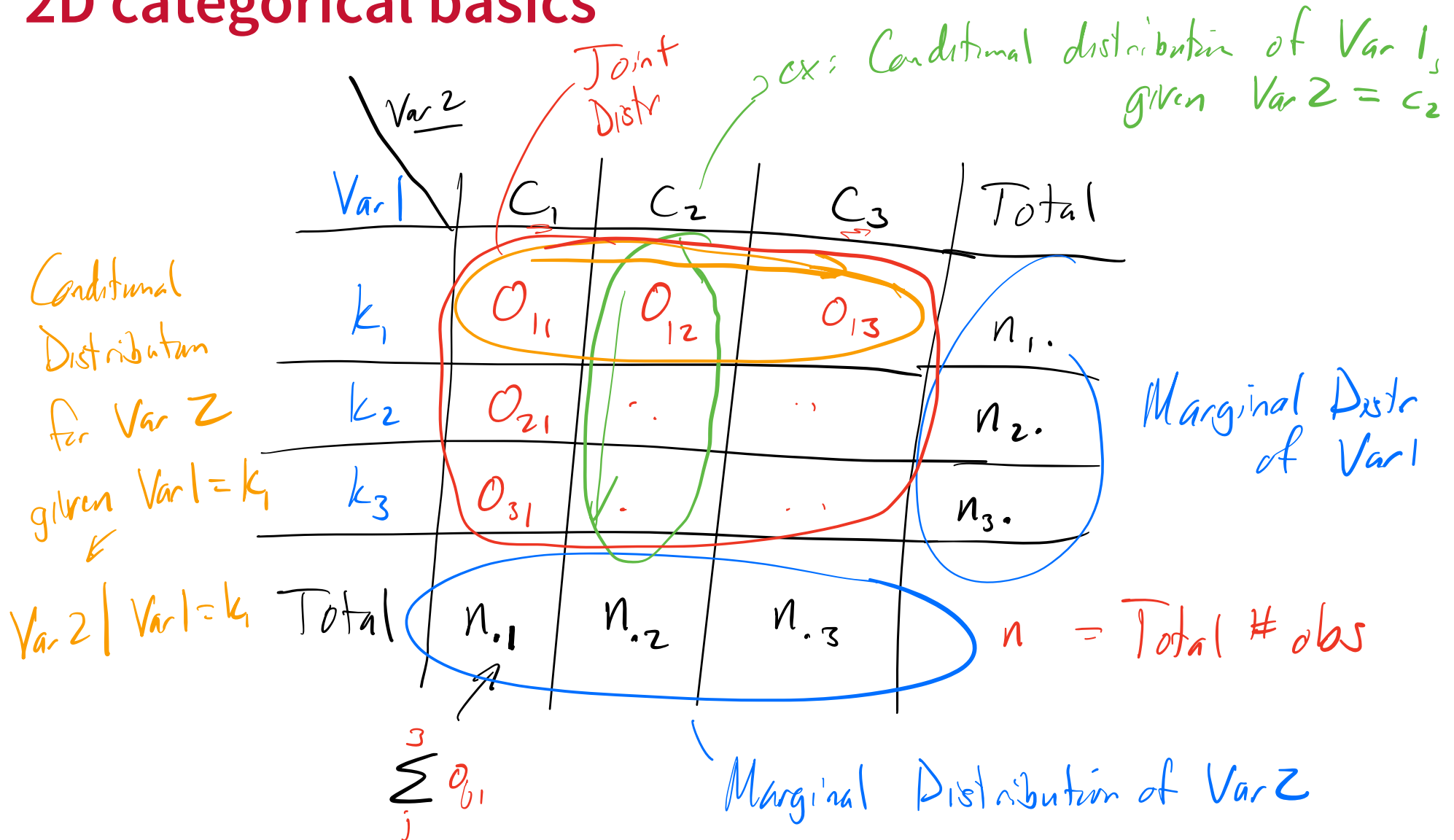
Useful to order categories by frequency with forcats

```
1 penguins |>
2   group_by(species) |>
3   summarize(count = n(), .groups = "drop") |>
4   mutate(total = sum(count),
5           prop = count / total,
6           se = sqrt(prop * (1 - prop) / total),
7           lower = prop - 2 * se,
8           upper = prop + 2 * se,
9           species = fct_reorder(species, prop)) |>
10  ggplot(aes(x = species)) +
11  geom_bar(aes(y = prop), stat = "identity") +
12  geom_errorbar(aes(ymin = lower, ymax = upper),
13              color = "red")
```

Useful to order categories by frequency with forcats



2D categorical basics



2D categorical basics

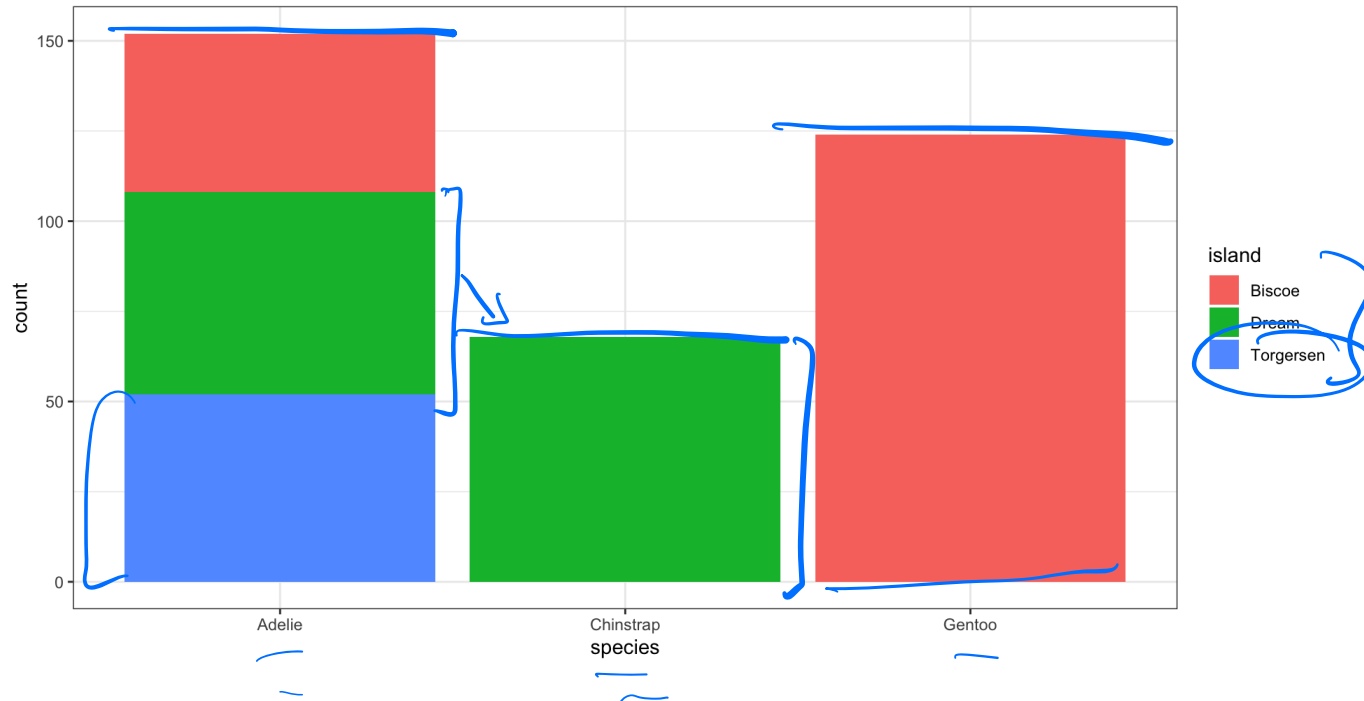
```
1 addmargins(table("Species" = penguins$species, "Island" = penguins$island))
```

Species	Island			Sum
	Biscoe	Dream	Torgersen	
Adelie	44	56	52	152
Chinstrap	0	68	0	68
Gentoo	124	0	0	124
Sum	168	124	52	344

- Column and row sums: marginal distributions
- Values within rows: conditional distribution for *Island* given *Species*
- Values within columns: conditional distribution for *Species* given *Island*
- Bottom right: total number of observations

Stacked bar charts - a bar chart of spine charts

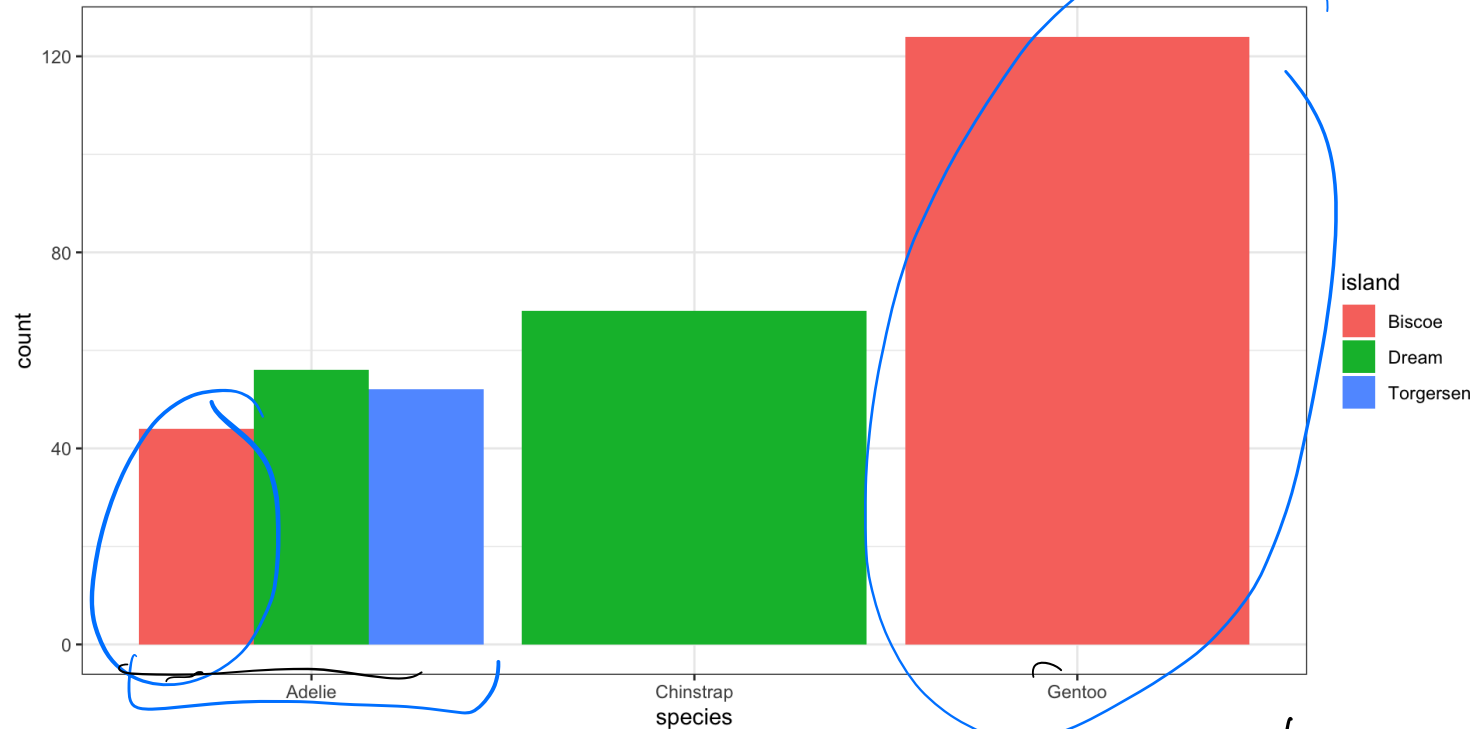
```
1 penguins |>
2   ggplot(aes(x = species, fill = island)) +
3   geom_bar() +
4   theme_bw()
```



- Easy to see marginal of species, i.e., $P(x)$
- Can see conditional of island | species, i.e., $P(\text{fill} | x)$
- Harder to see conditional of species | island, i.e., $P(x | \text{fill})$

Side-by-side bar charts

```
1 penguins |>
2   ggplot(aes(x = species, fill = island)) +
3   geom_bar(position = "dodge") +
4   theme_bw()
```



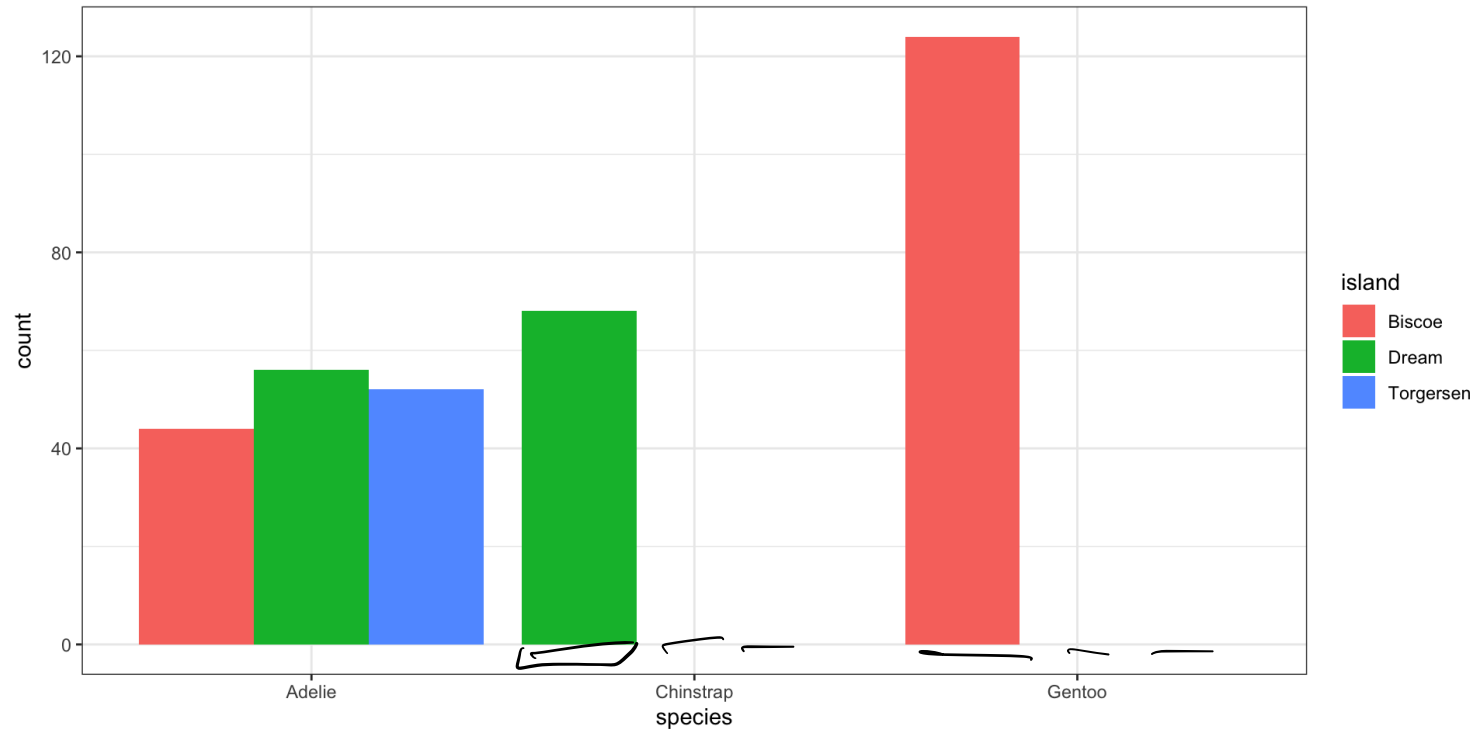
- Easy to see conditional of island | species, i.e., $P(\text{fill} | x)$
- Can see conditional of species | island, i.e., $P(x | \text{fill})$

statds-36315-spring25

We can also see: joint
Hard to see: marginal

Side-by-side bar charts

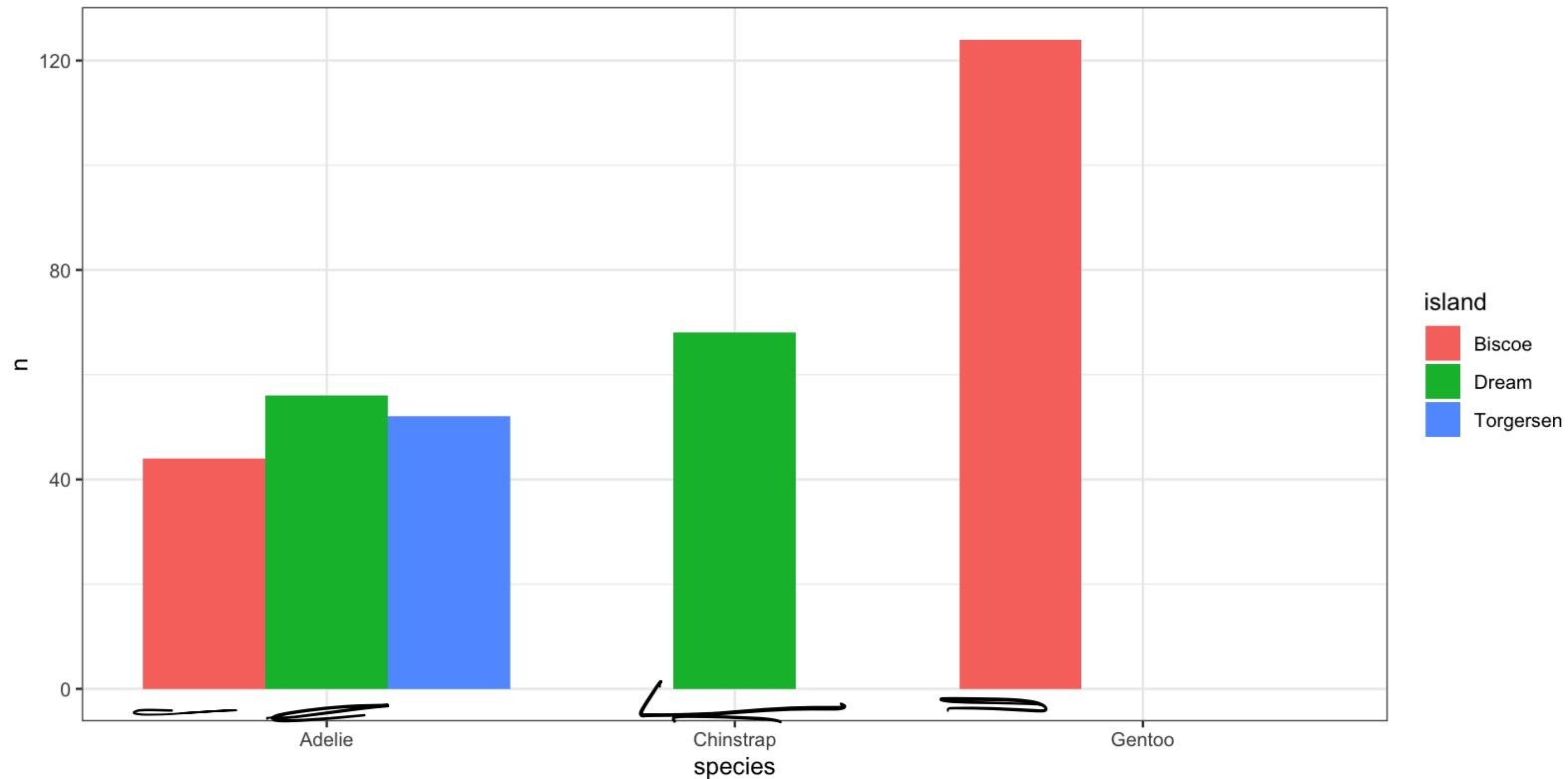
```
1 penguins |>
2   ggplot(aes(x = species, fill = island)) +
3   geom_bar(position = position_dodge(preserve = "single")) +
4   theme_bw()
```



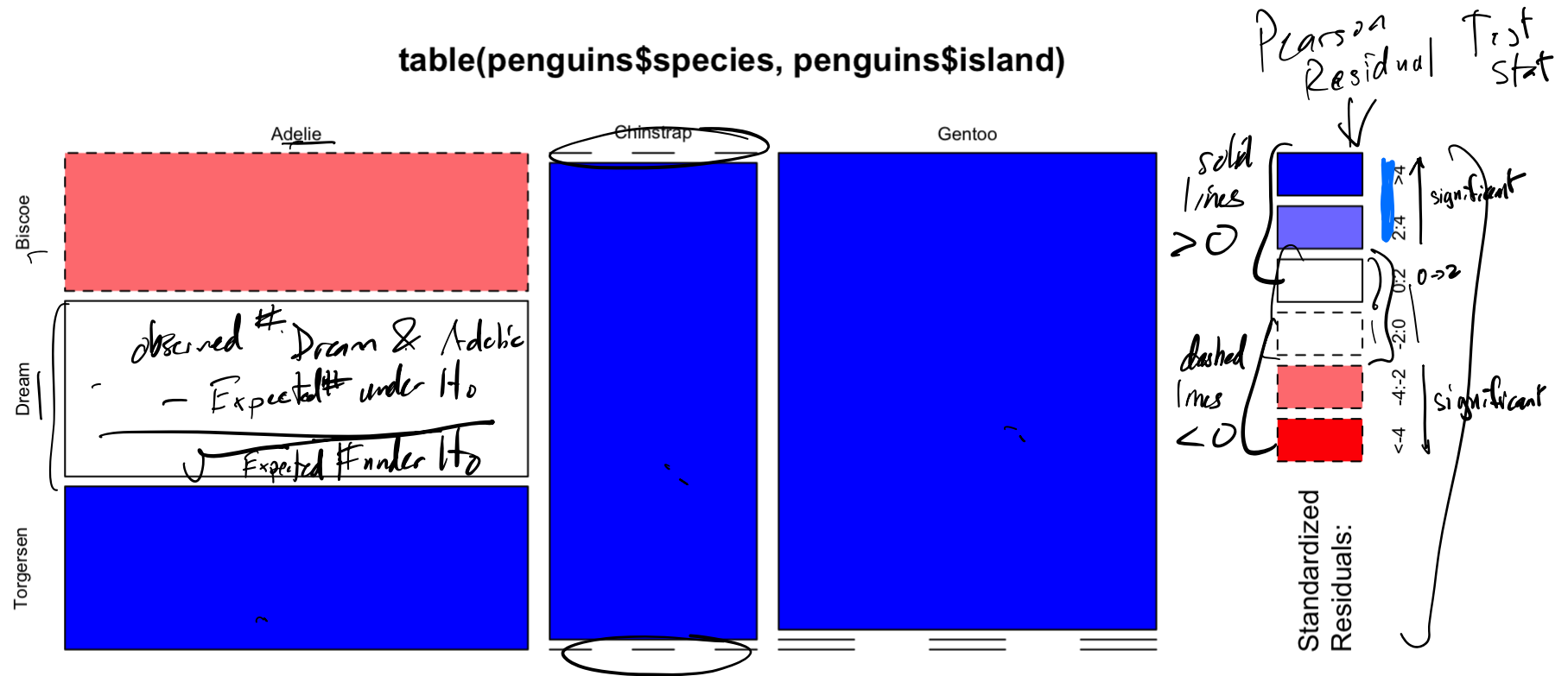
- Easy to see conditional of `island | species`, i.e., $P(\text{fill} | x)$
- Can see conditional of `species | island`, i.e., $P(x | \text{fill})$

Complete missing values to preserve location

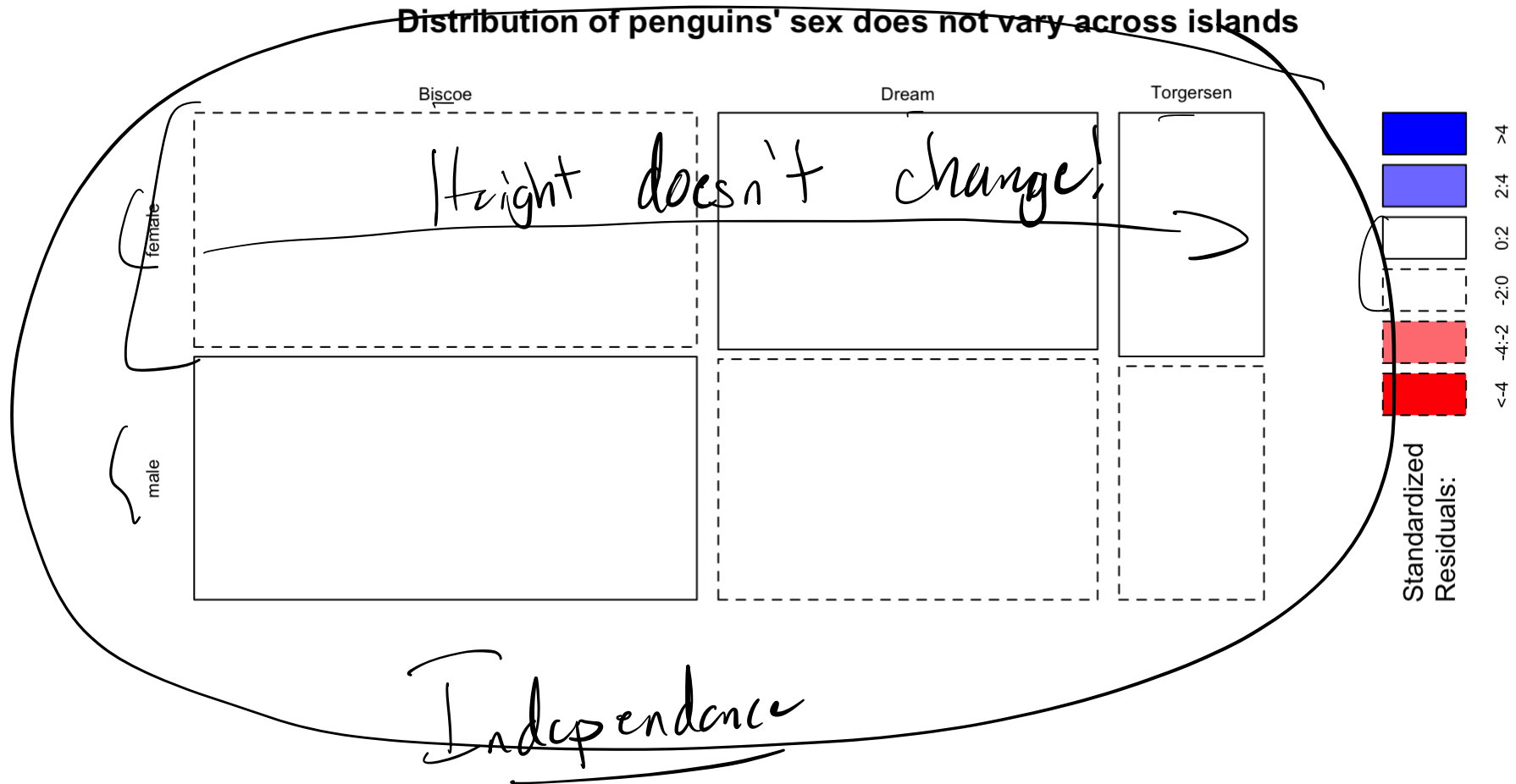
```
1 penguins |>
2   count(species, island) |>
3   complete(species = unique(species), island = unique(island),
4     fill = list(n = 0)) |>
5   ggplot(aes(x = species, y = n, fill = island)) +
6   geom_bar(stat = "identity", position = "dodge") +
7   theme_bw()
```



```
1 mosaicplot(table(penguins$species, penguins$island), shade = TRUE)
```

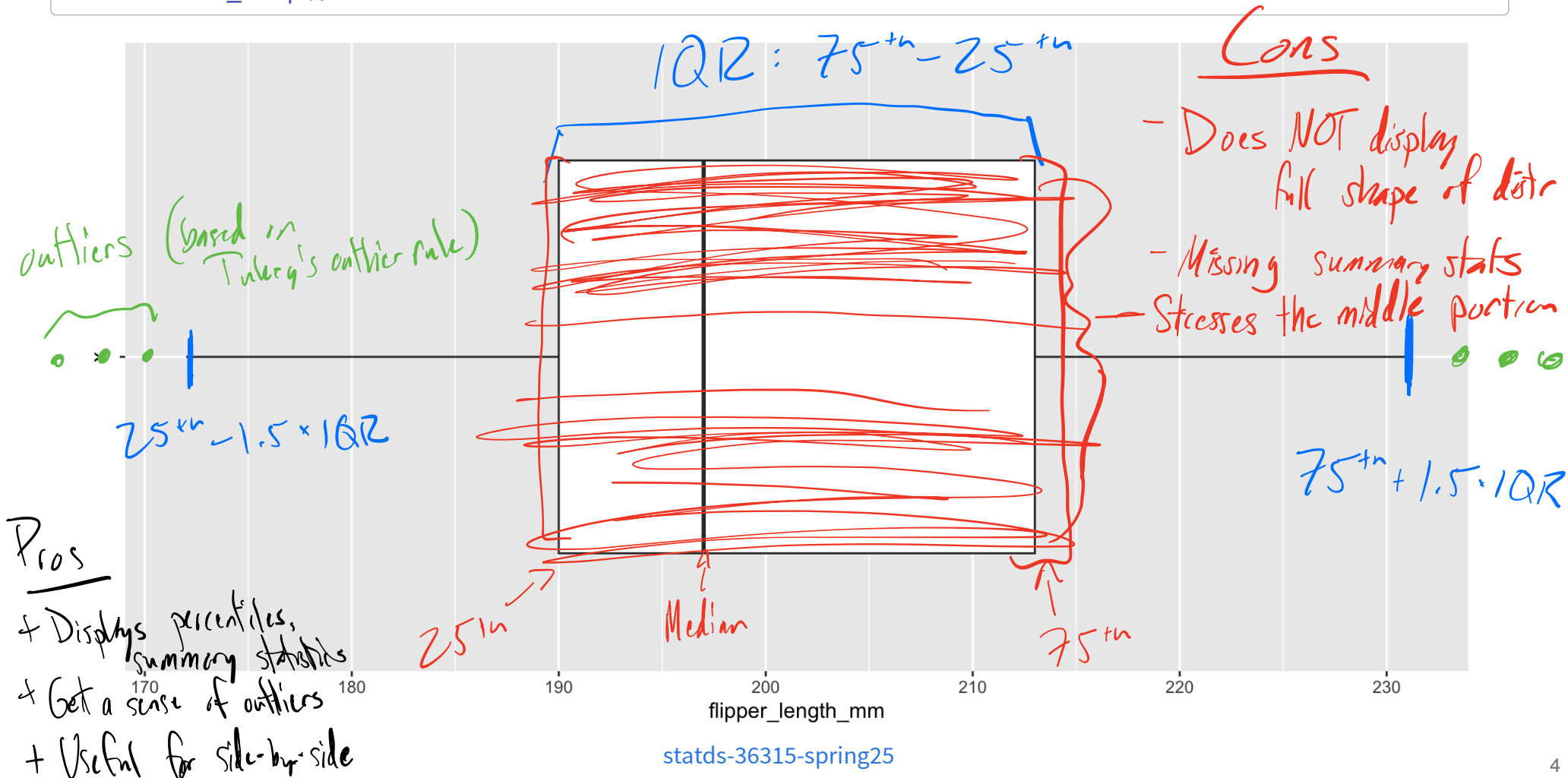


```
1 mosaicplot(table(penguins$island, penguins$sex), shade = TRUE,
2             main = "Distribution of penguins' sex does not vary across islands")
```



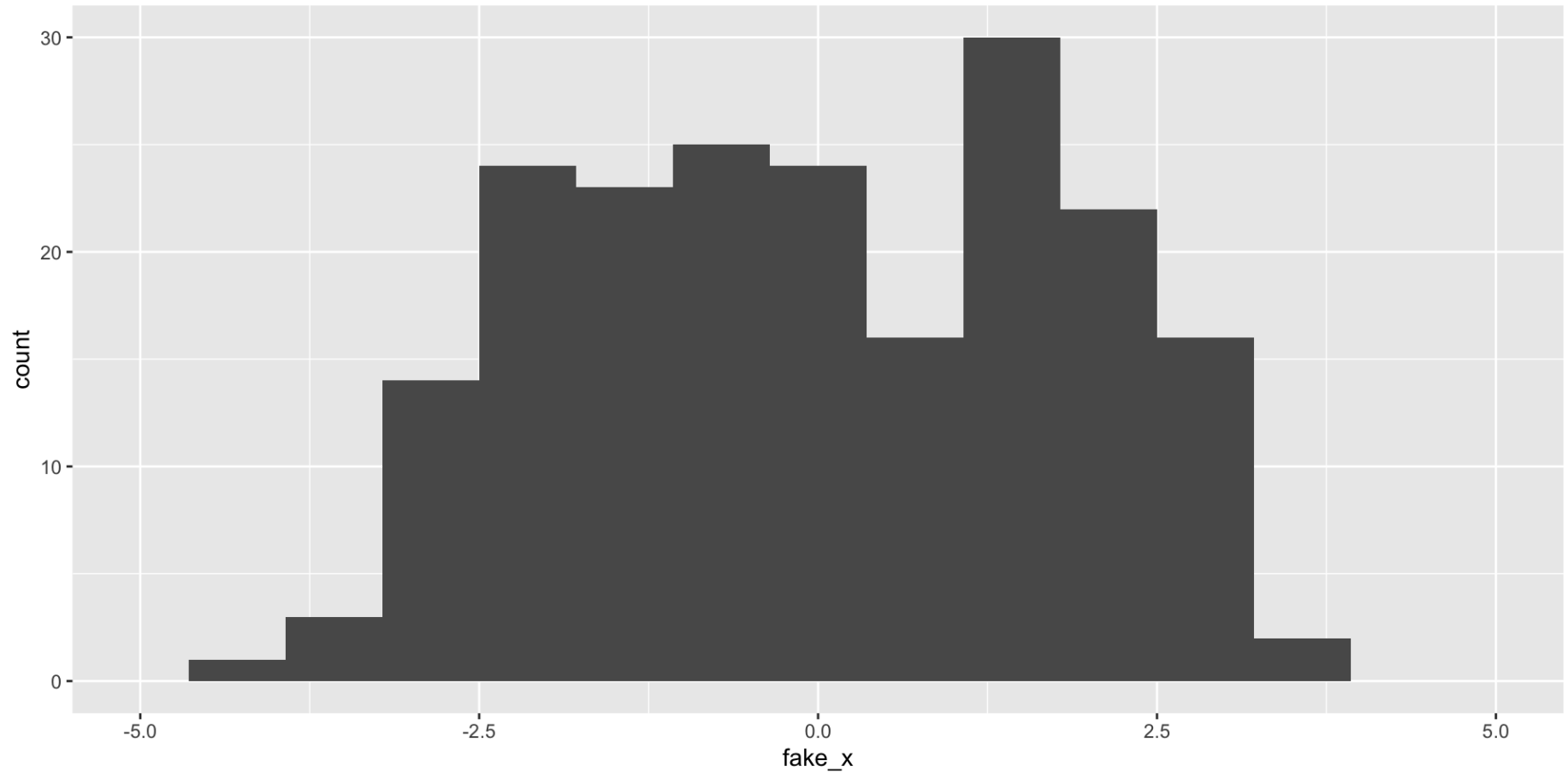
Box plots visualize summary statistics

```
1 penguins |>  
2   ggplot(aes(y = flipper_length_mm)) +  
3   geom_boxplot(aes(x = "")) +  
4   coord_flip()
```



What happens as we change the number of bins?

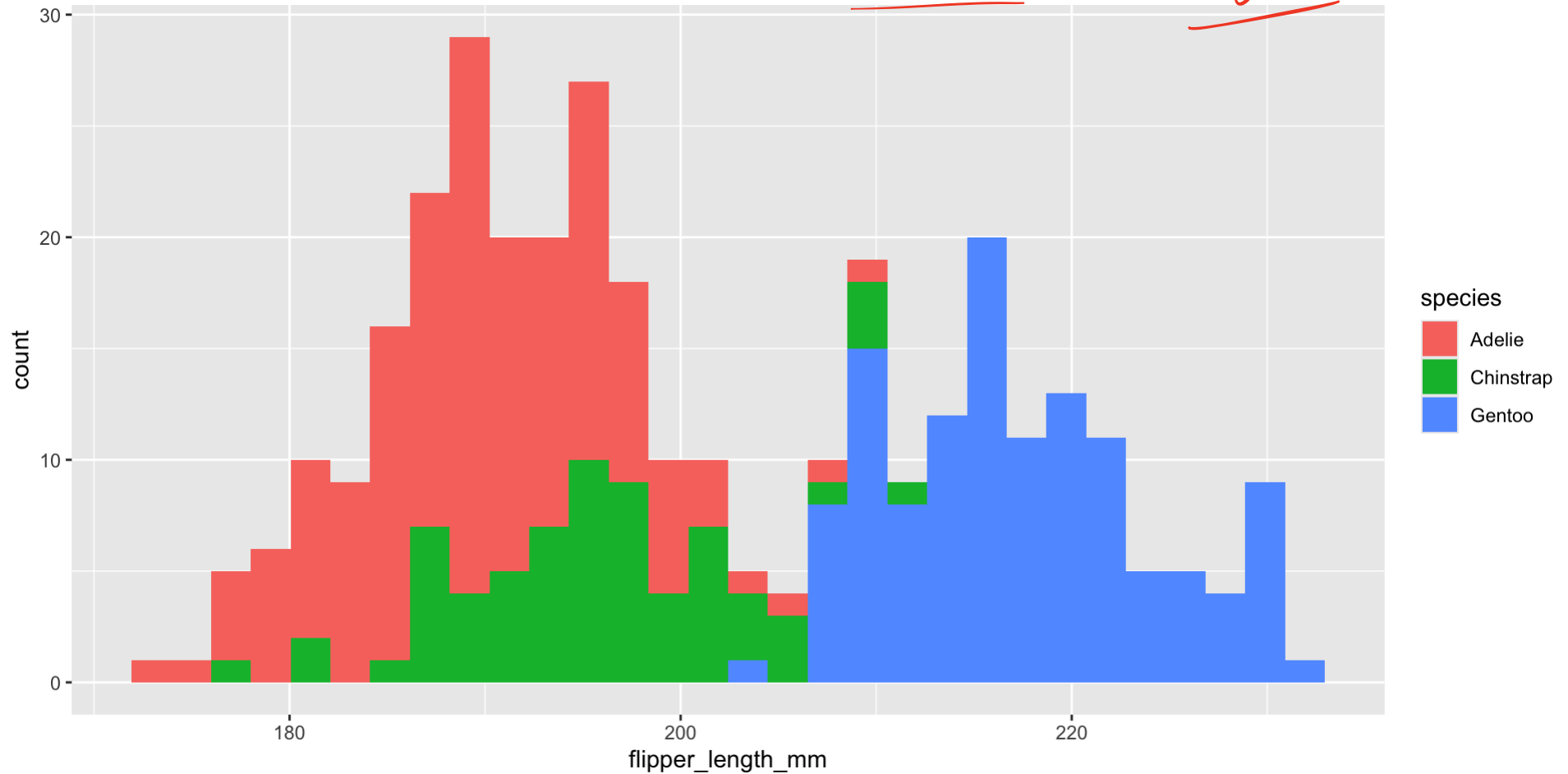
```
1 fake_data |>  
2   ggplot(aes(x = fake_x)) +  
3   geom_histogram(bins = 15) +  
4   scale_x_continuous(limits = c(-5, 5))
```



What about displaying conditional distributions?

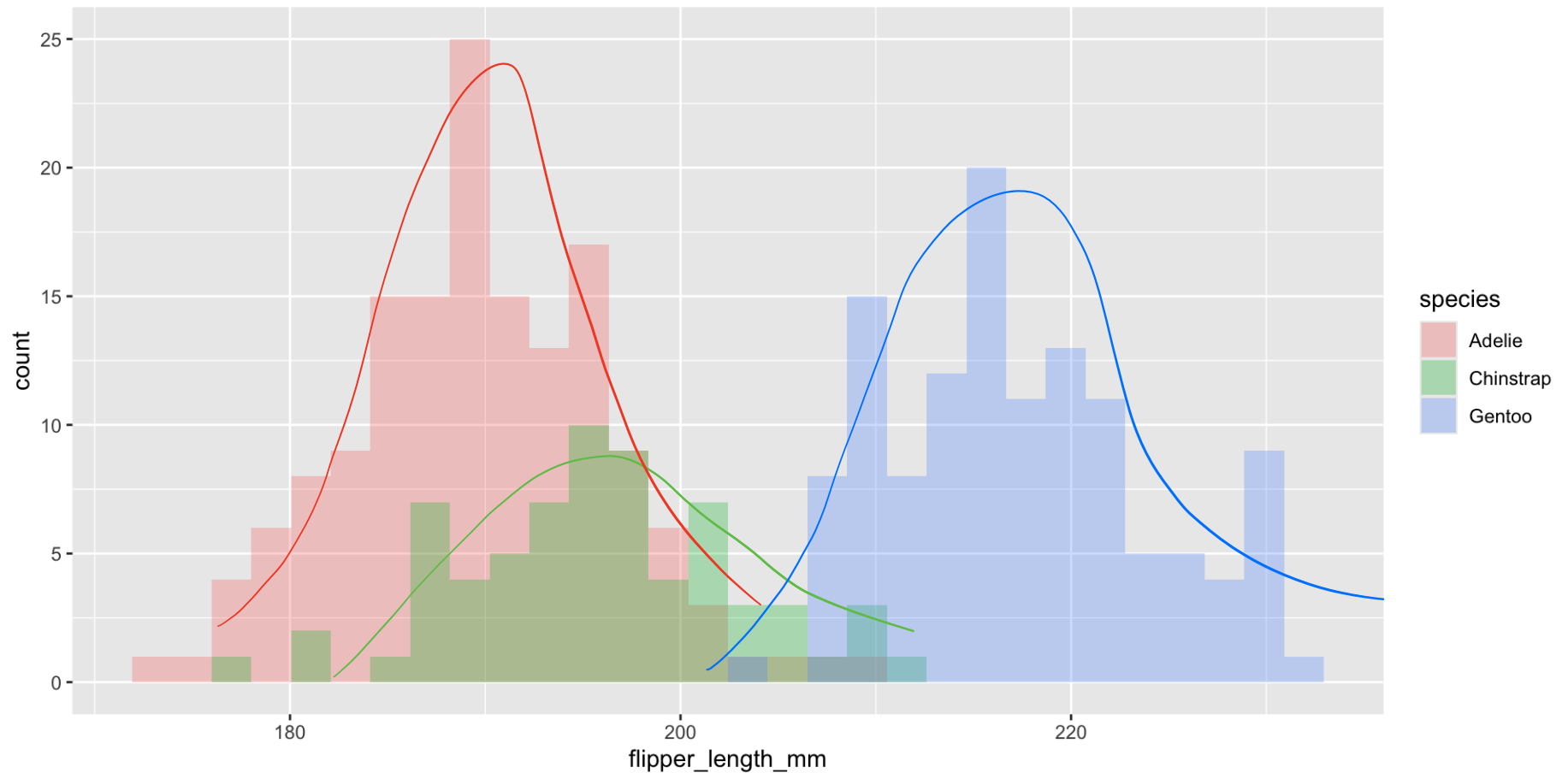
```
1 penguins |>  
2   ggplot(aes(x = flipper_length_mm)) +  
3   geom_histogram(aes(fill = species))
```

→ Stacking → Marginal



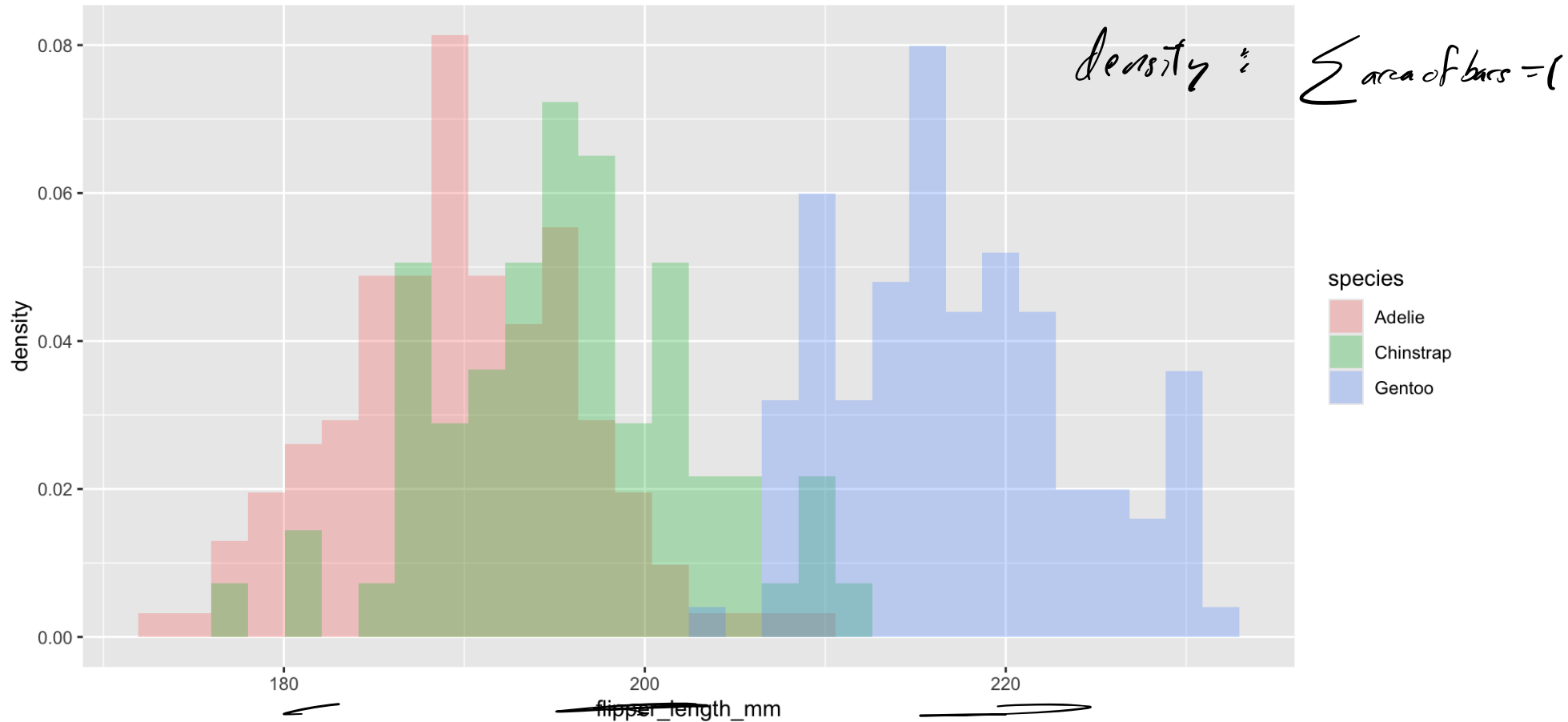
What about displaying conditional distributions?

```
1 penguins |>  
2   ggplot(aes(x = flipper_length_mm)) +  
3   geom_histogram(aes(fill = species),  
4                 position = "identity", alpha = 0.3)
```



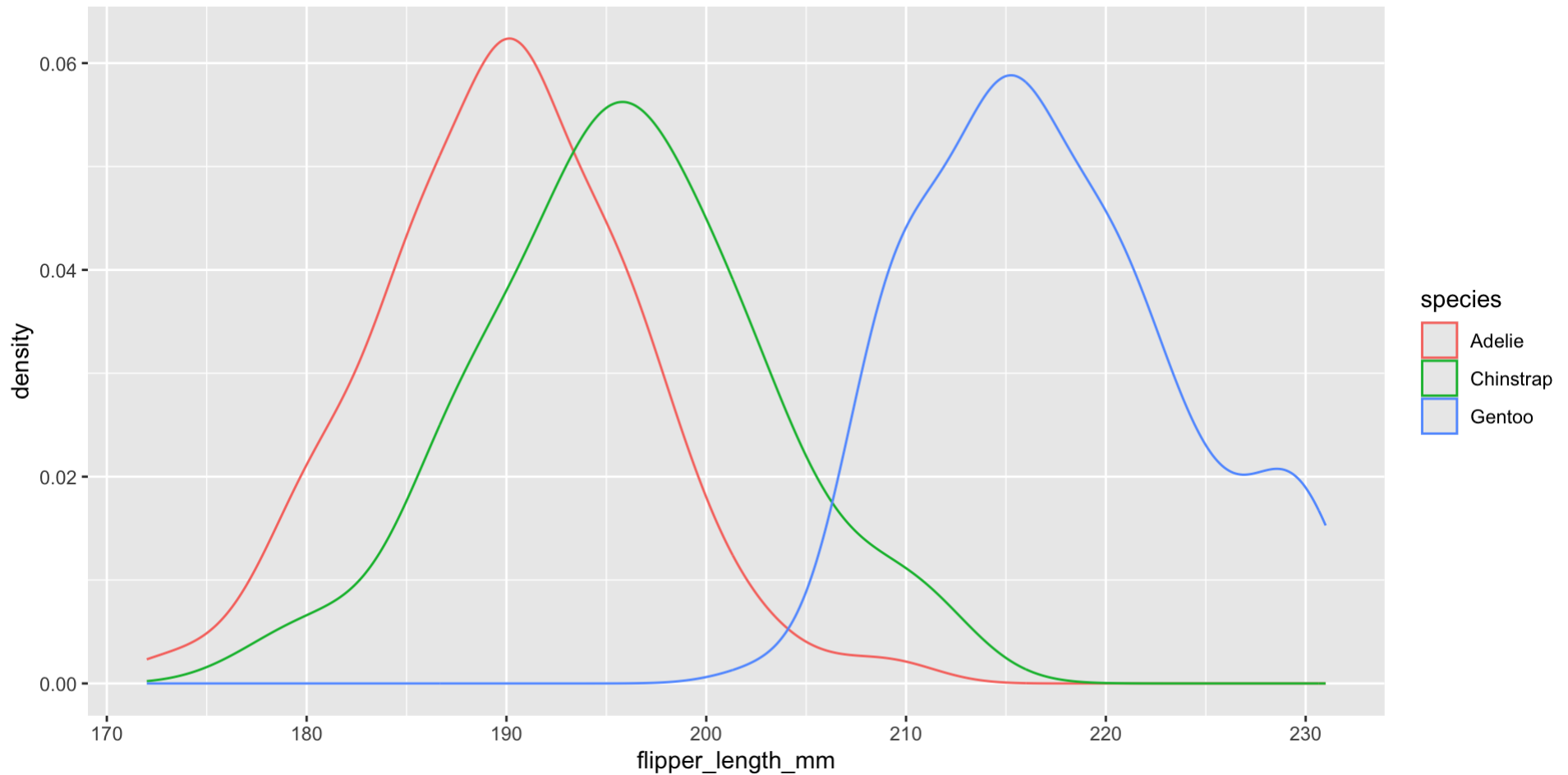
Normalize histogram frequencies with density

```
1 penguins |>  
2   ggplot(aes(x = flipper_length_mm)) +  
3   geom_histogram(aes(y = after_stat(density), fill = species),  
4                 position = "identity", alpha = 0.3)
```



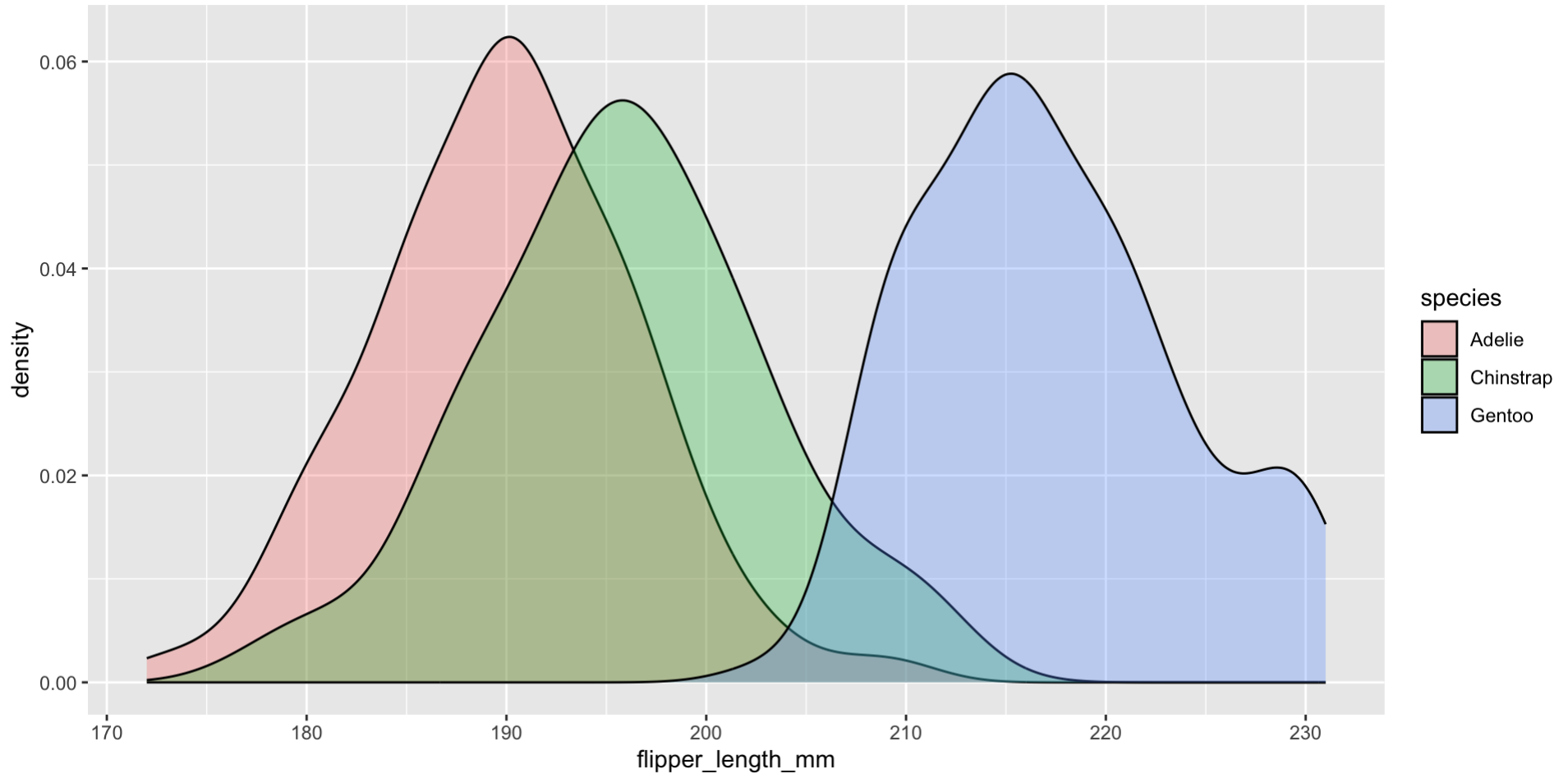
Can use density curves instead

```
1 penguins |>  
2   ggplot(aes(x = flipper_length_mm)) +  
3   geom_density(aes(color = species))
```



We should NOT fill the density curves

```
1 penguins |>
2   ggplot(aes(x = flipper_length_mm)) +
3   geom_density(aes(fill = species), alpha = .3)
```



Kernel density estimation

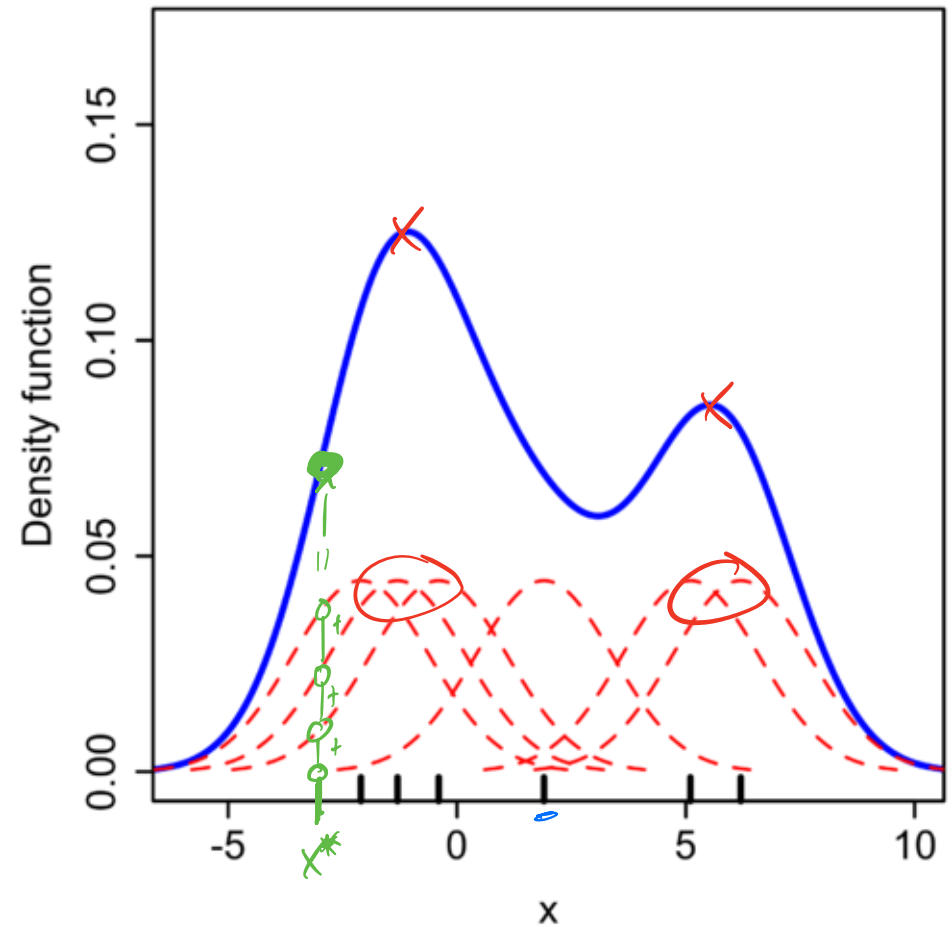
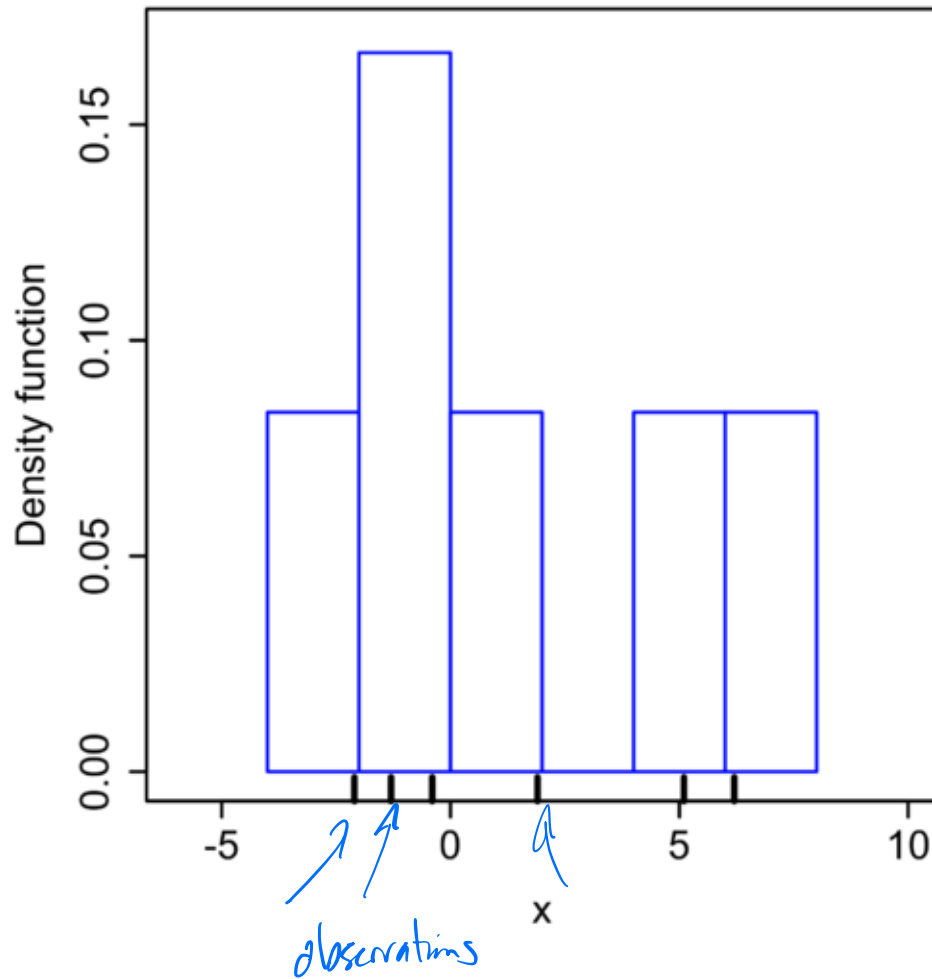
Goal: estimate PDF $f(x)$ for all possible values (assuming it is continuous & smooth)

$$\text{Kernel density estimate: } \hat{f}(x) = \frac{1}{n} \sum_{i=1}^n \frac{1}{h} K_h(\underline{x} - x_i)$$

Handwritten notes: A blue bracket is drawn over the K_h term, with an arrow pointing to a handwritten $K(\frac{x-x_i}{h})$ above it.

- n = sample size, x = new point to estimate $f(x)$ (does NOT have to be in dataset!)
- h = **bandwidth**, analogous to histogram bin width, ensures $\hat{f}(x)$ integrates to 1
- x_i = i th observation in dataset
- $K_h(x - x_i)$ is the **Kernel** function, creates **weight** given distance of i th observation from new point
 - as $|x - x_i| \rightarrow \infty$ then $K_h(x - x_i) \rightarrow 0$, i.e. further apart i th row is from x , smaller the weight
 - as **bandwidth** $h \uparrow$ weights are more evenly spread out (as $h \downarrow$ more concentrated around x)
 - typically use **Gaussian / Normal** kernel: $\propto e^{-(x-x_i)^2/2h^2}$
 - $K_h(x - x_i)$ is large when x_i is close to x

Wikipedia example

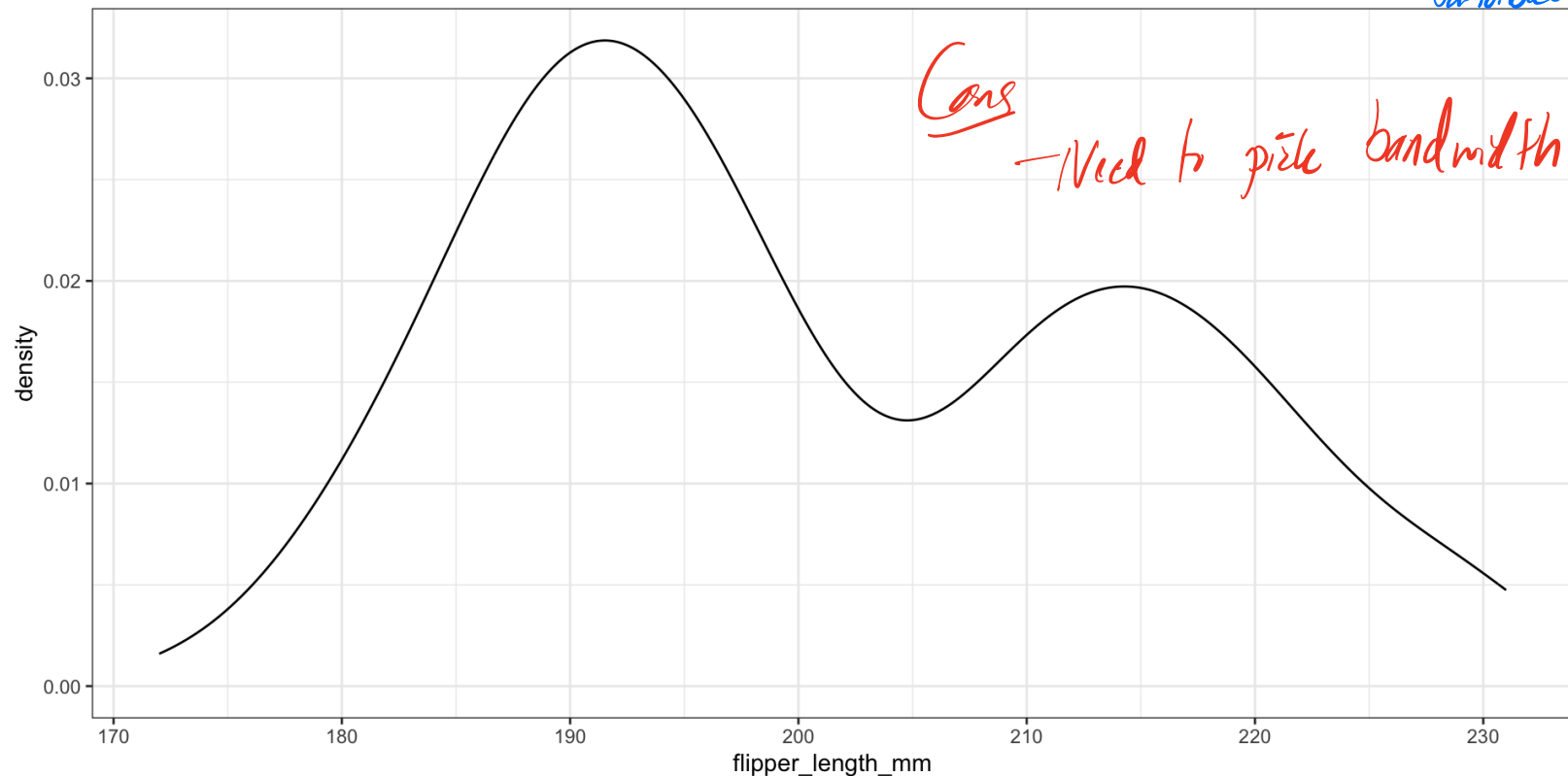


We display kernel density estimates with `geom_density()`

```
1 penguins |>  
2   ggplot(aes(x = flipper_length_mm)) +  
3   geom_density() +  
4   theme_bw()
```

Pros

- + Display full shape of distribution
- + Easily layer, add categorical variables w/ color

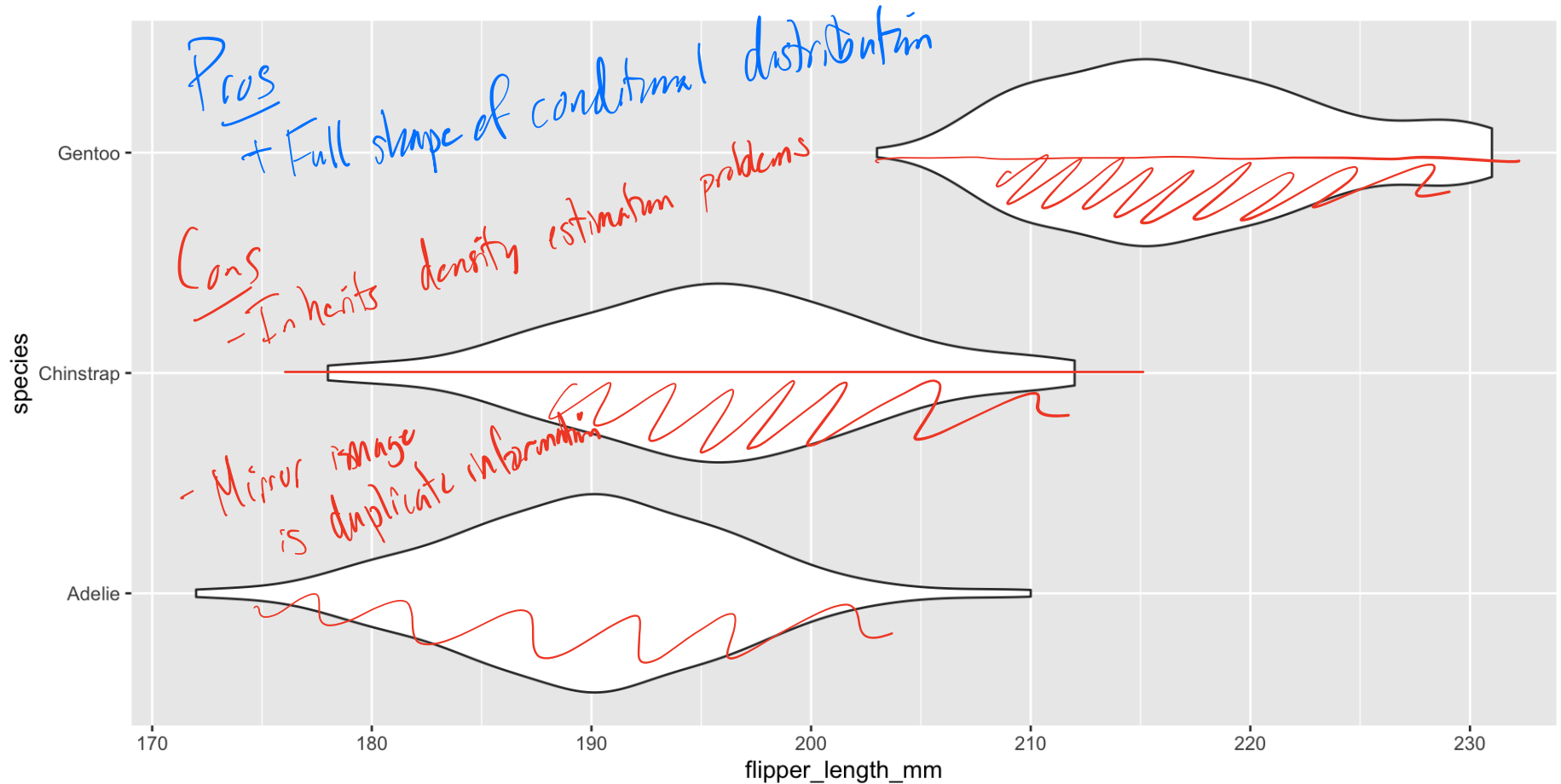


Cons

- Need to pick bandwidth & kernel

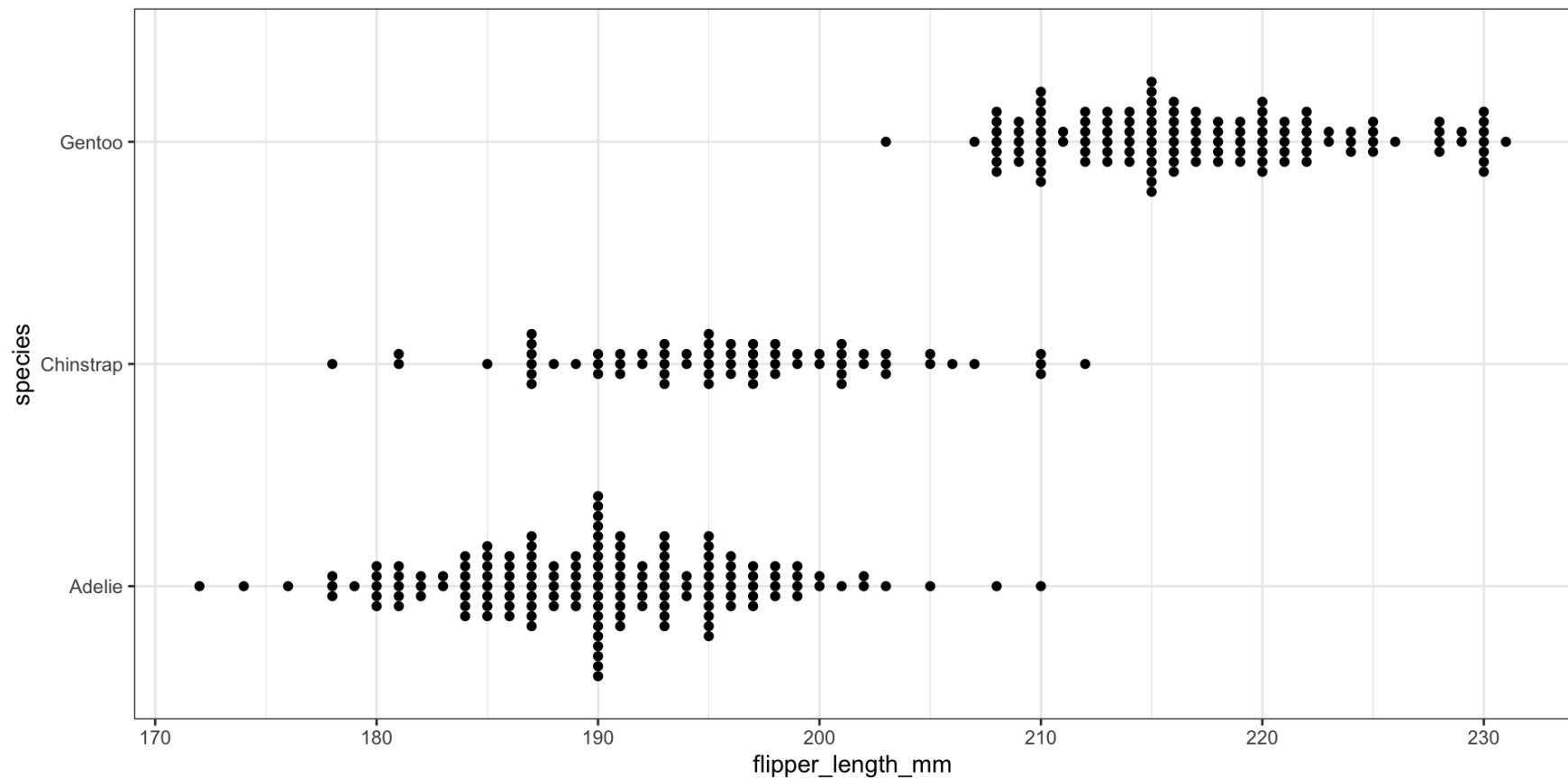
Visualizing conditional distributions: violin plots

```
1 penguins |>  
2   ggplot(aes(x = species, y = flipper_length_mm)) +  
3   geom_violin() +  
4   coord_flip()
```



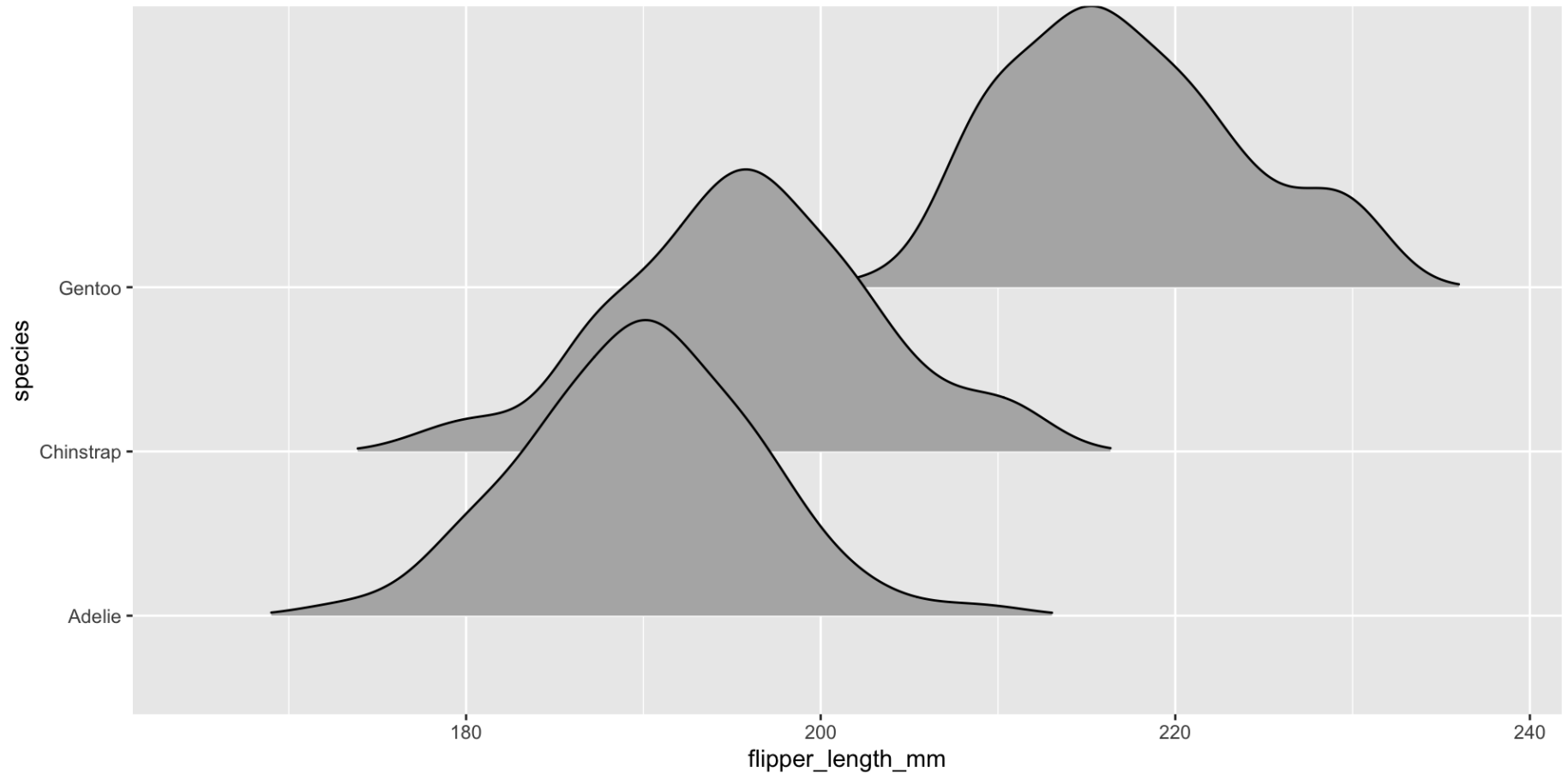
Visualizing conditional distributions: ggbeeswarm

```
1 library(ggbeeswarm)
2 penguins |>
3   ggplot(aes(x = flipper_length_mm, y = species)) +
4   geom_beeswarm(cex = 1.5) +
5   theme_bw()
```



Visualizing conditional distributions: `ggridges`

```
1 library(ggridges)
2 penguins |>
3   ggplot(aes(x = flipper_length_mm, y = species)) +
4   geom_density_ridges(rel_min_height = 0.01)
```



Visualizing conditional distributions: ggbeeswarm

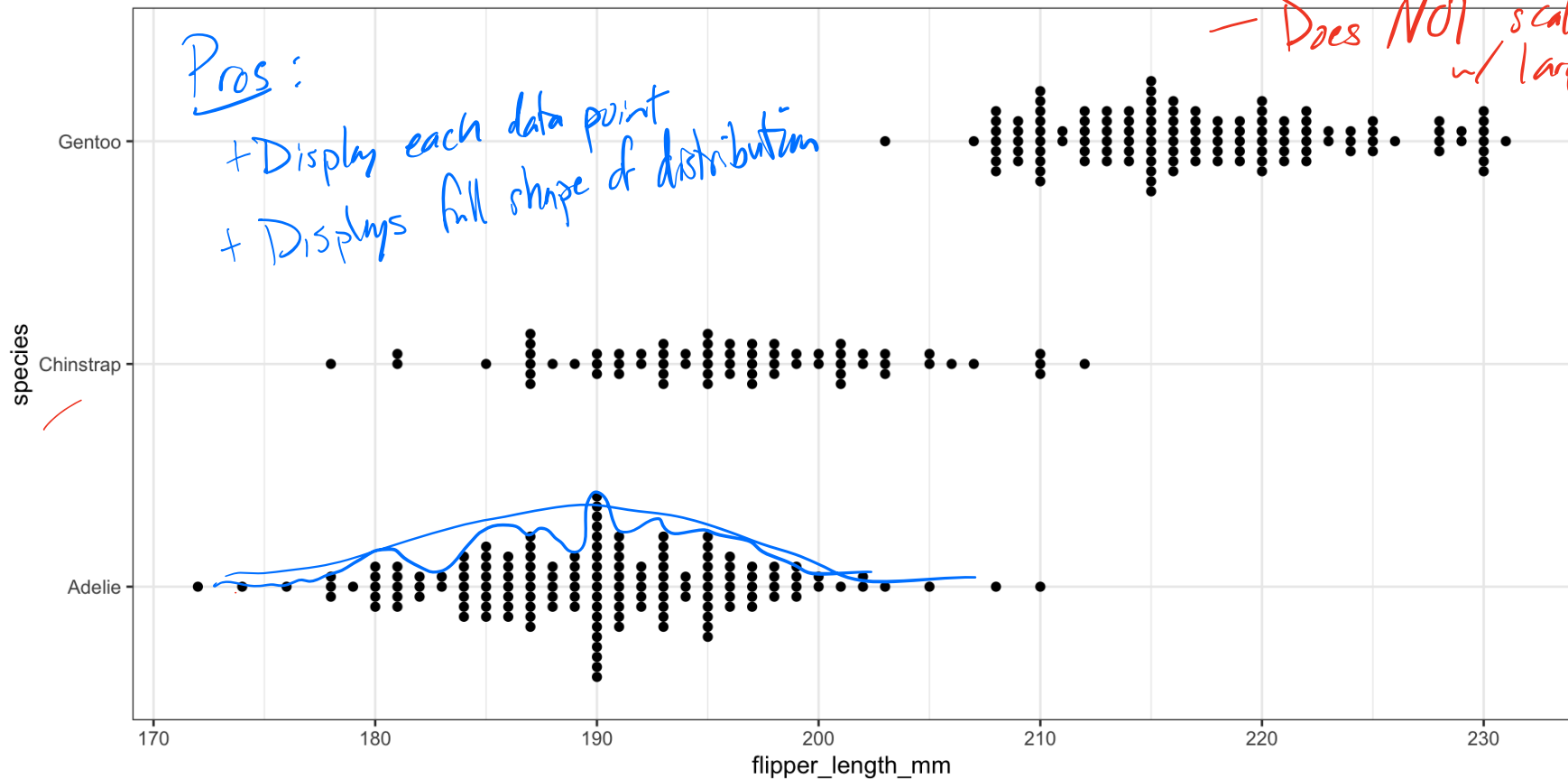
```
1 library(ggbeeswarm)
2 penguins |>
3   ggplot(aes(x = flipper_length_mm, y = species)) +
4   geom_beeswarm(cex = 1.5) +
5   theme_bw()
```

Cons
- Need some algorithm for arranging points

- Does NOT scale well w/ large data

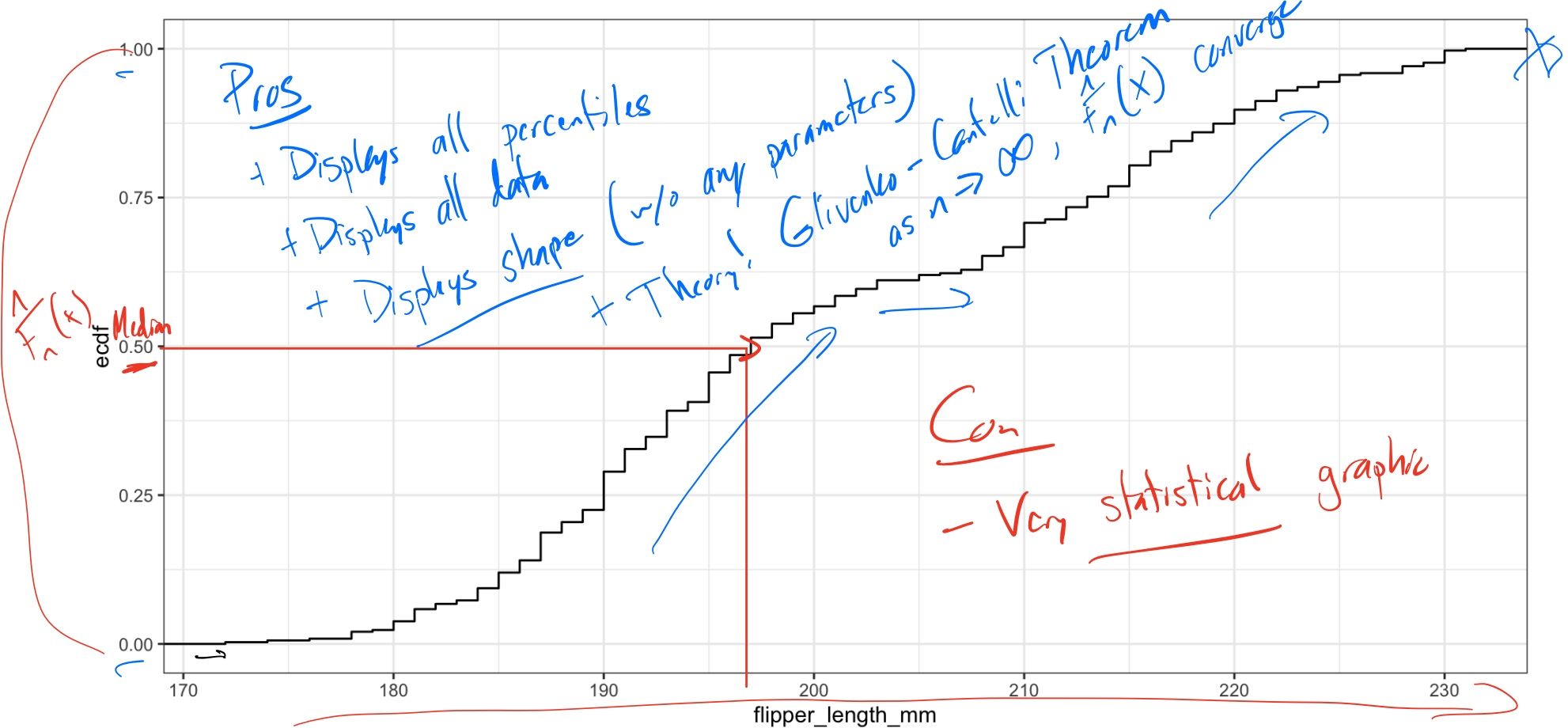
Pros:

+ Display each data point
+ Displays full shape of distribution



Display full distribution with ECDF plot

```
1 penguins |>
2   ggplot(aes(x = flipper_length_mm)) +
3   stat_ecdf() +
4   theme_bw()
```



What's the relationship between these two?

