Data Types, Probability Distributions, and ggplot2

Tim Frasier

Data Types

Data Types

- In general, data can take on one of three different forms
    1. Metric
    2. Ordinal
    3. Nominal

- Need to consider this when developing appropriate Bayesian models

Data Types

Metric

- When values are actual measurements that can take on a range of values (e.g., temperature, weight, counts, frequencies)

- Indicate 2 things
  - The order of values (which ones are largest and which are smallest)
  - The scale of difference between values

Data Types

Metric

Both metric

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Data Types

Metric

- Can tell order (which are larger and which are smaller)
- Can tell by how much values differ

Both metric

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

Data Types
Ordinal

- Provide information on order, but not scale

Data Types
Ordinal

- Provide information on order, but not scale
  - Places in a race

Ordinal

Person	Place
Emily	First
Victoria	Second
Tasha	Third
Ben	Fourth

- Can tell order, but not scale (e.g, time difference between individuals)

Data Types

Nominal

- Categorical
  - Sex
  - Political party
  - Others

- Can tell what category something is in, but not order or scale

Data Types

Nominal

Nominal

x	age	fat	sex
1	24	15.5	male
2	37	20.9	male
3	41	18.6	male
4	60	28	male
5	31	34.7	female
6	39	30.2	female
…	…	…	…

- Can tell category, but not order or scale

Probability Distributions In R and STAN

Probability Distributions

- To conduct Bayesian analyses, you have to be fairly comfortable with at least a few probability distributions

- Different ones suitable for different data, with different characteristics

Continuous Distributions

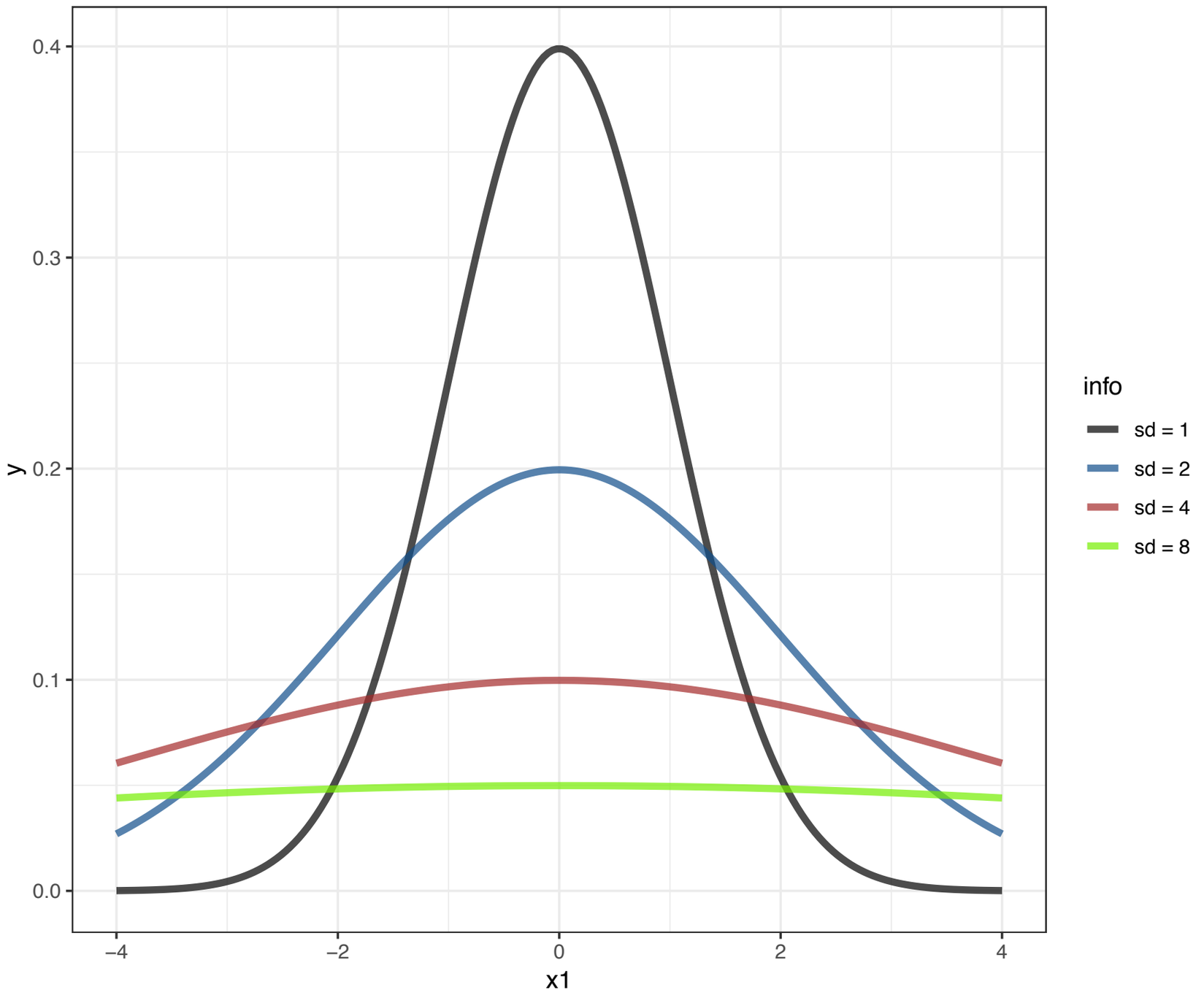
Probability Distributions

Normal distribution

- Symmetric around the mean (mean, median, and mode are all equal)
- Range is (-∞ to ∞)
- Has two parameters:
  - Mean: determines position of peak along x-axis
  - Standard Deviation: determines how wide the peak is around the mean

Probability Distributions

Normal distribution



Probability Distributions

Normal distribution

- In R, use the `dnorm` function

```
dnorm(x, mean, sd)
```

Probability Distributions

Normal distribution

- In R, use the `dnorm` function

```
dnorm(x, mean, sd)
```

x values over which to calculate probabilities

Probability Distributions

Normal distribution

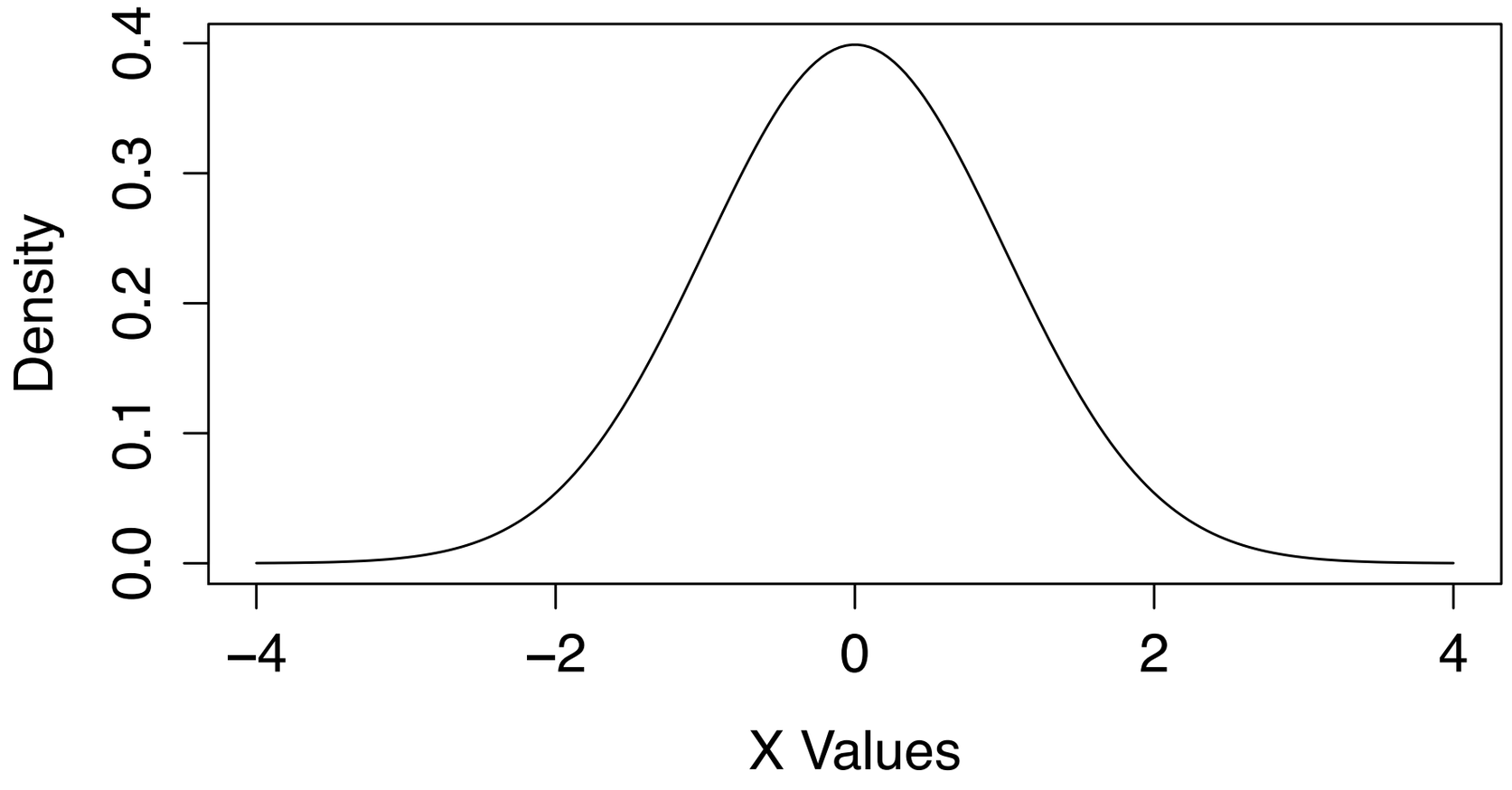
- In R, use the `dnorm` function

```
dnorm(x, mean, sd)
```

```
x = seq(from = -4, to = 4, length = 200)
y = dnorm(x, mean = 0, sd = 1)
plot(x, y, type = "l", lwd = 2, xlab = "X Values", ylab = "Density")
```

Probability Distributions

Normal distribution



Probability Distributions

Normal distribution

- In Stan, function is called `normal`, requires:
  - `mu (μ)` = mean
  - `sigma (σ)` = standard deviation

```
normal(mu, sigma)
```

Probability Distributions

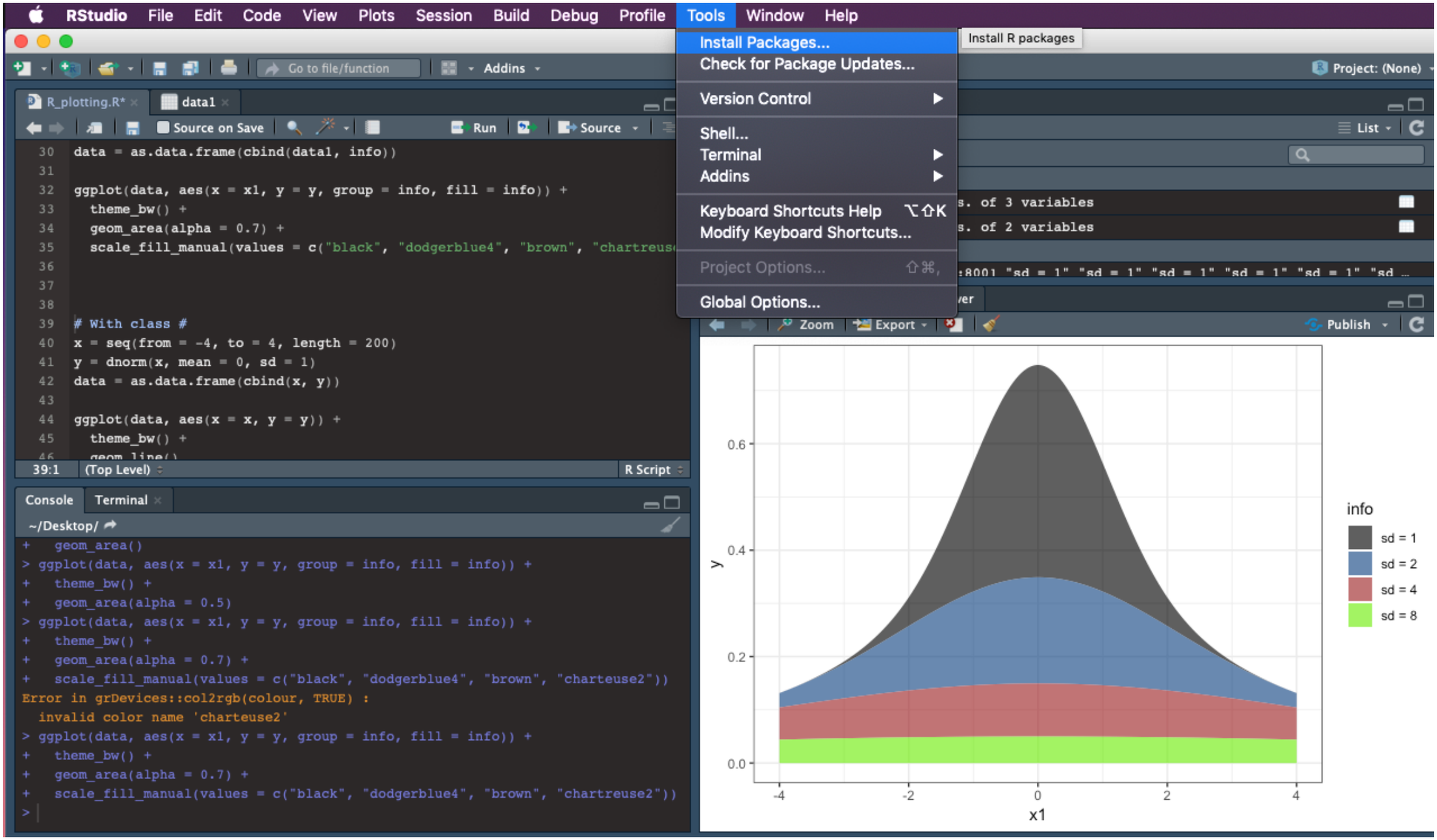
Normal distribution

- Because of the central limit theorem, normal distribution is appropriate for many cases when our goal is to estimate the mean response to certain conditions

https://vimeo.com/75089338

ggplot2

- Install



ggplot2

- "grammar of graphics"
- Build plots layer by layer, customizing as you go
  - A bit tricky to learn at first, but ultimately easier, with nicer results
- Data must be in a data frame
  - Based on the philosophy of "tidy data"

ggplot2

Tidyverse

Packages Articles Learn Help Contribute

R packages for data science

The tidyverse is an opinionated collection of R packages designed for data science. All packages share an underlying design philosophy, grammar, and data structures.

Install the complete tidyverse with:

```
install.packages("tidyverse")
```

ggplot2

- Load library

```
library(ggplot2)
```

ggplot2

- Load library

```
library(ggplot2)
```

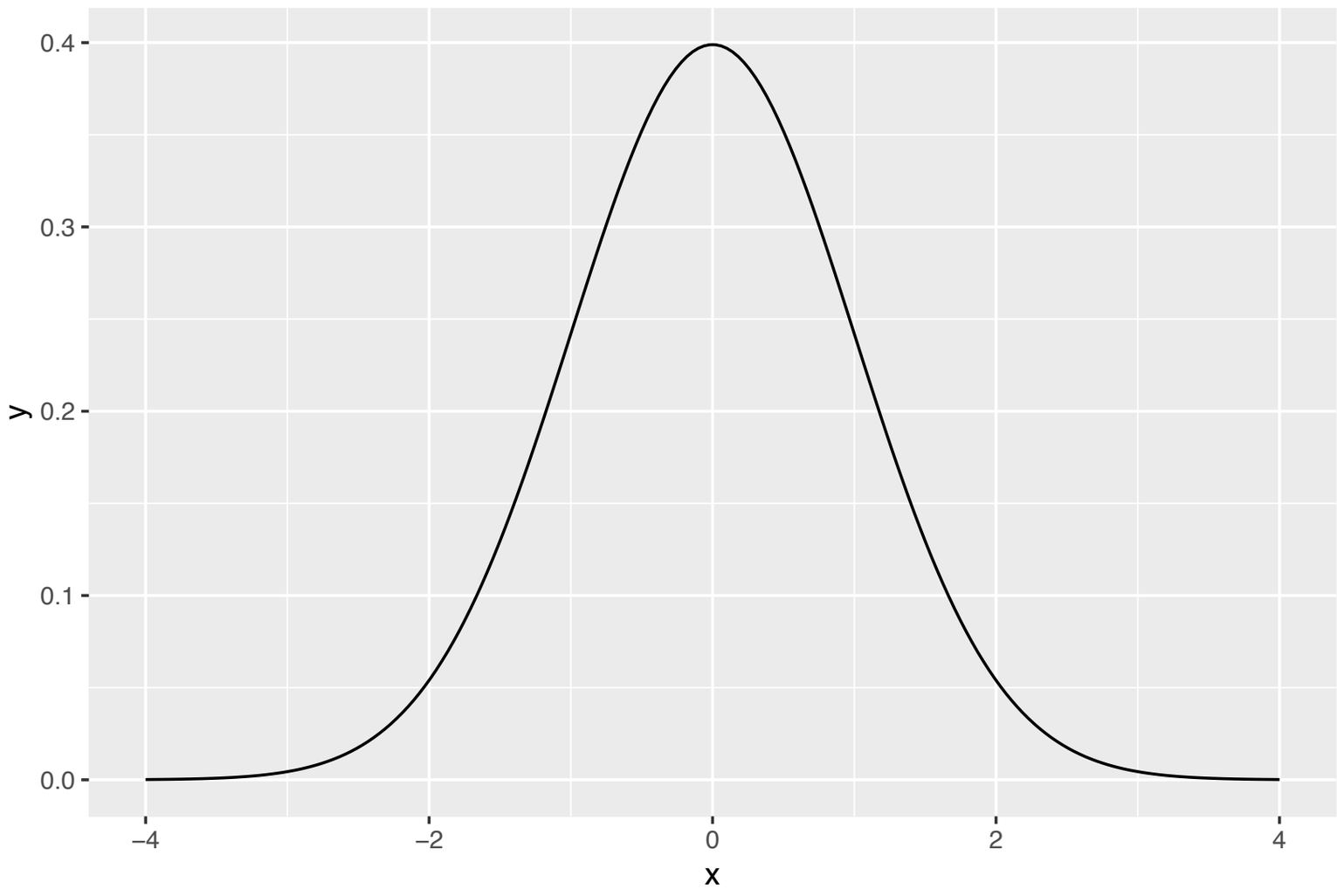
- Combine `x` and `y` data into a data frame

```
data = as.data.frame(cbind(x, y))
```

ggplot2

- Plot in ggplot2

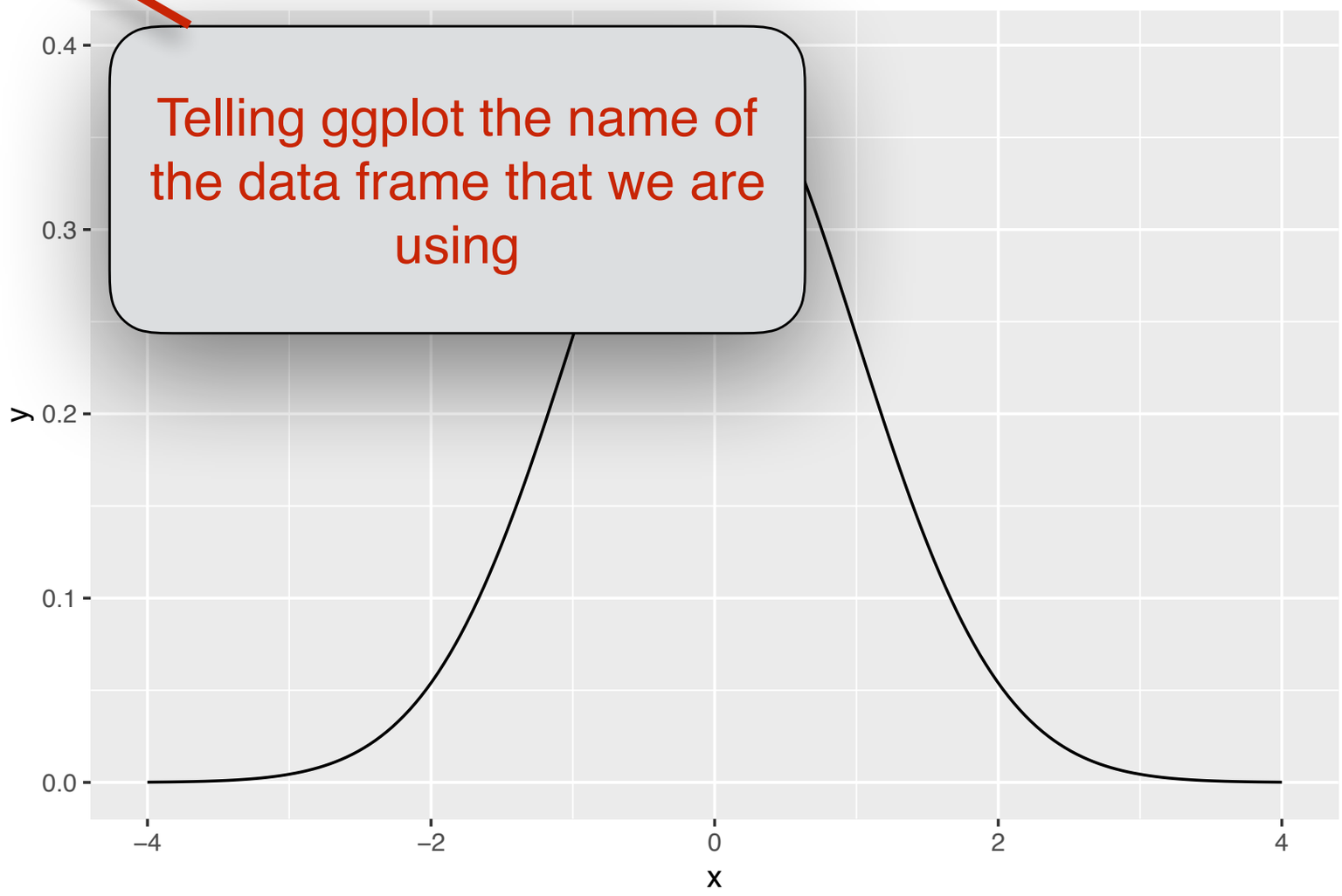
```
ggplot(data) +
   geom_line(aes(x = x, y = y))
```



ggplot2

- Plot in ggplot2

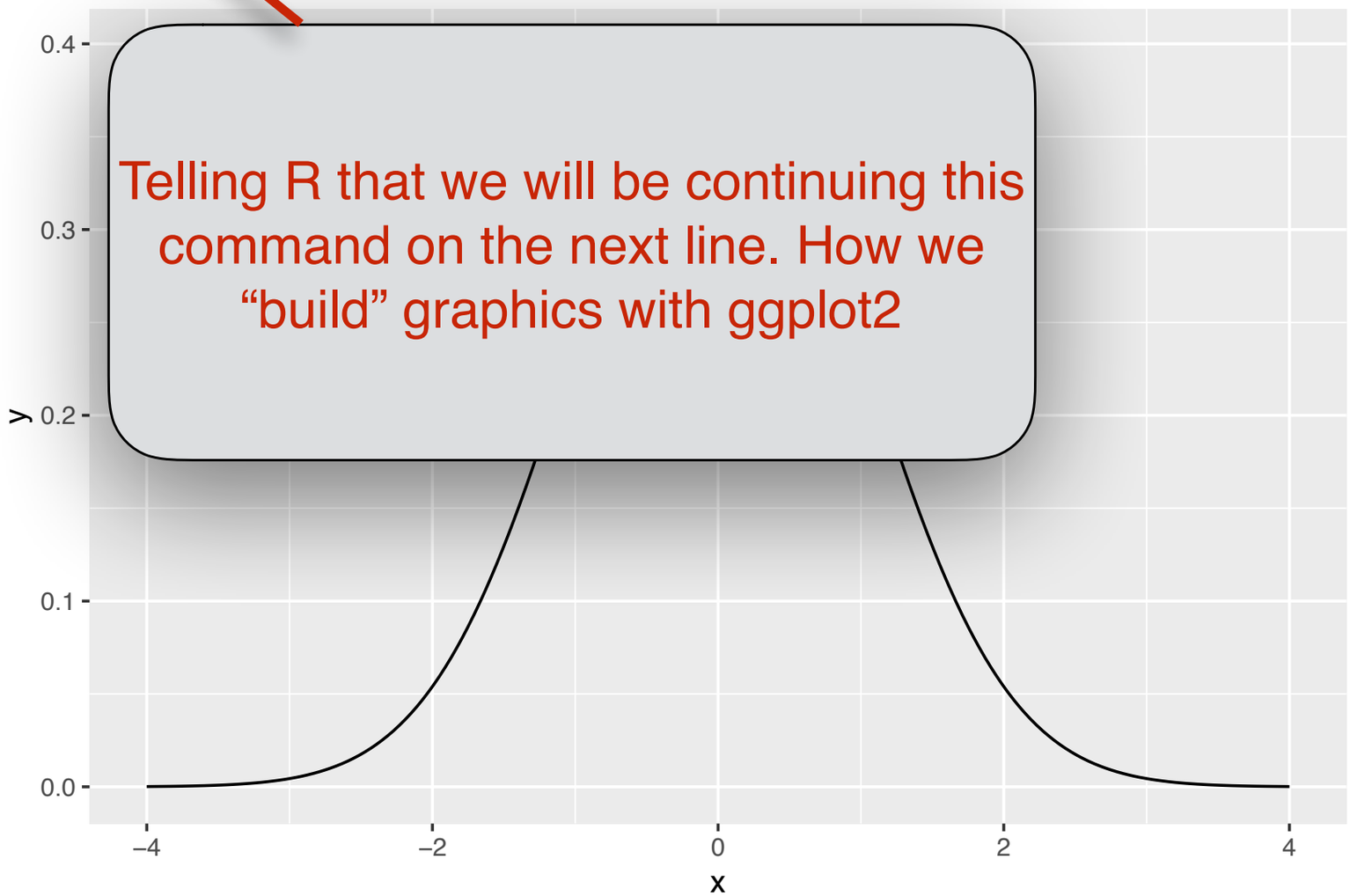
```
ggplot(data) +
    geom_line(aes(x = x, y = y))
```



ggplot2

- Plot in ggplot2

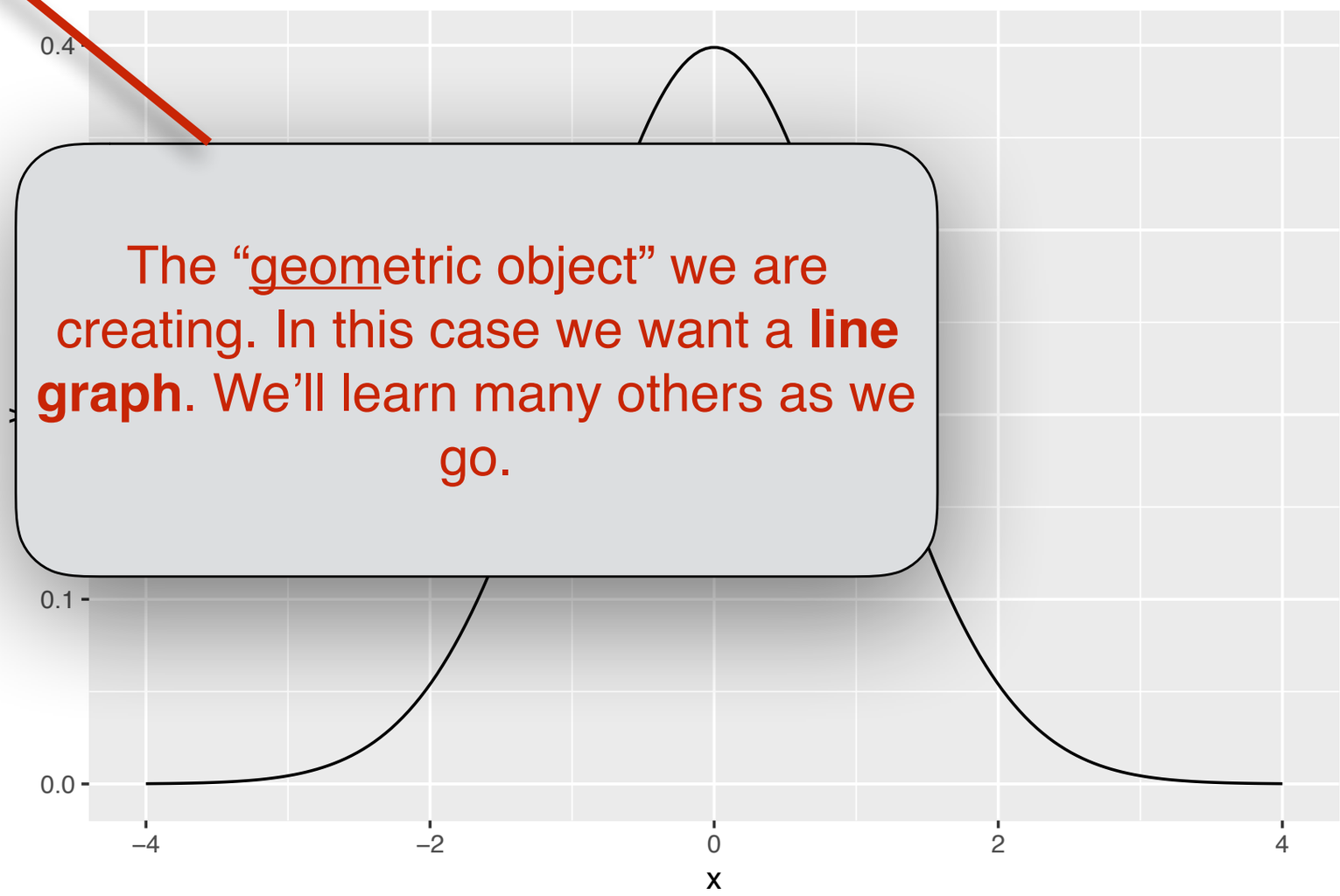
```
ggplot(data) +
   geom_line(aes(x = x, y = y))
```



ggplot2

- Plot in ggplot2

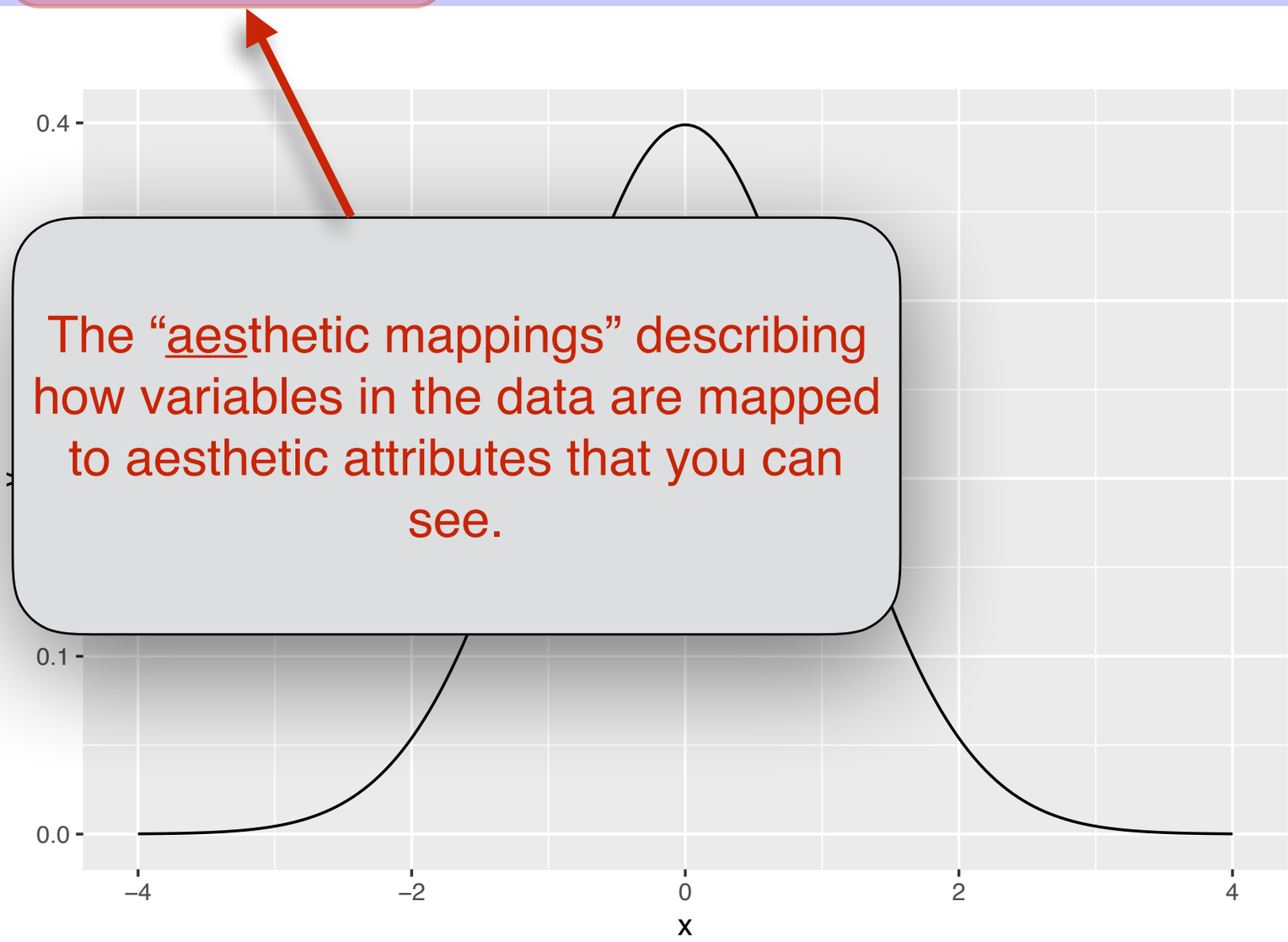
```
ggplot(data) +
   geom_line(aes(x = x, y = y))
```



ggplot2

- Plot in ggplot2

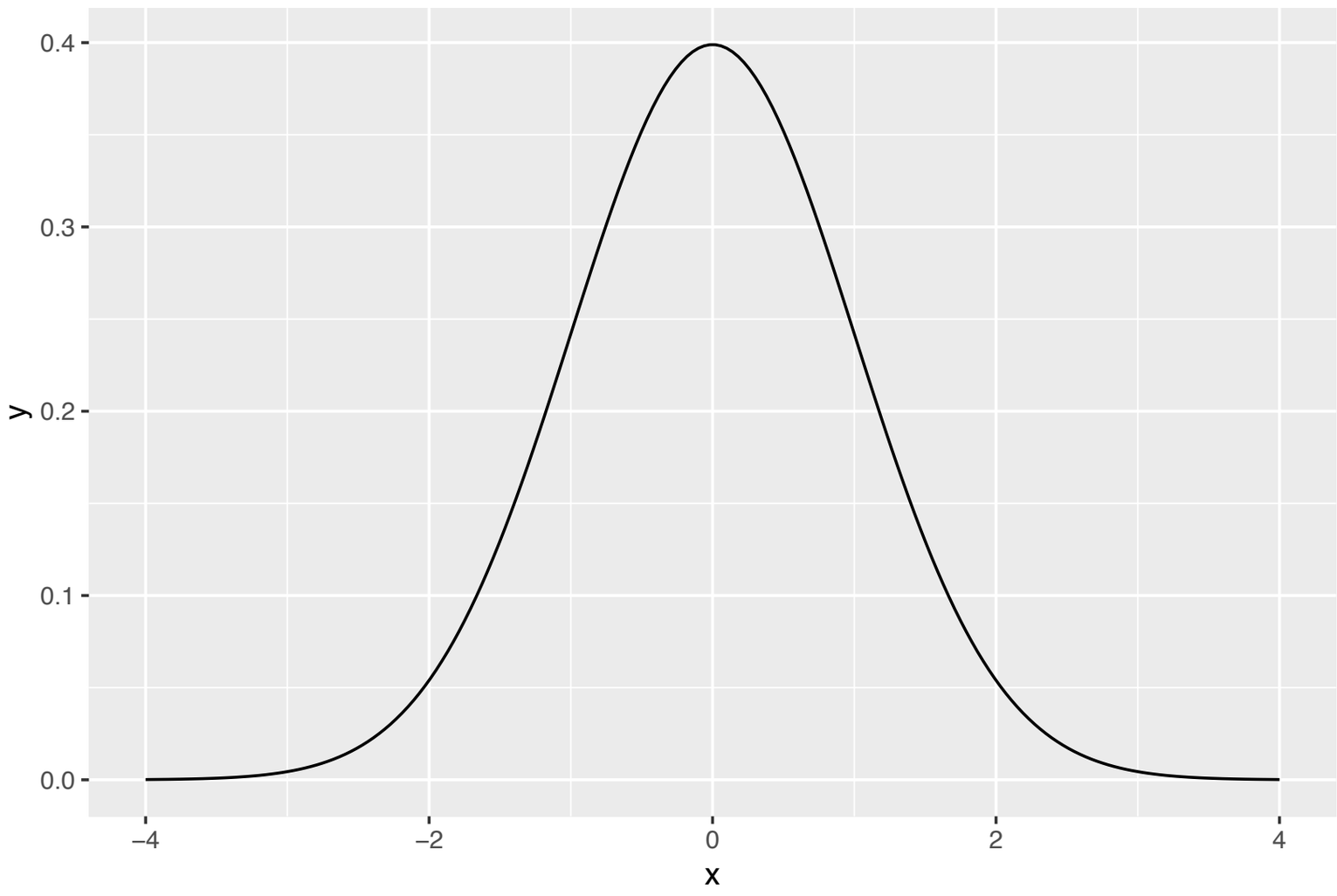
```
ggplot(data) +
   geom_line(aes(x = x, y = y))
```



ggplot2

- Plot in ggplot2

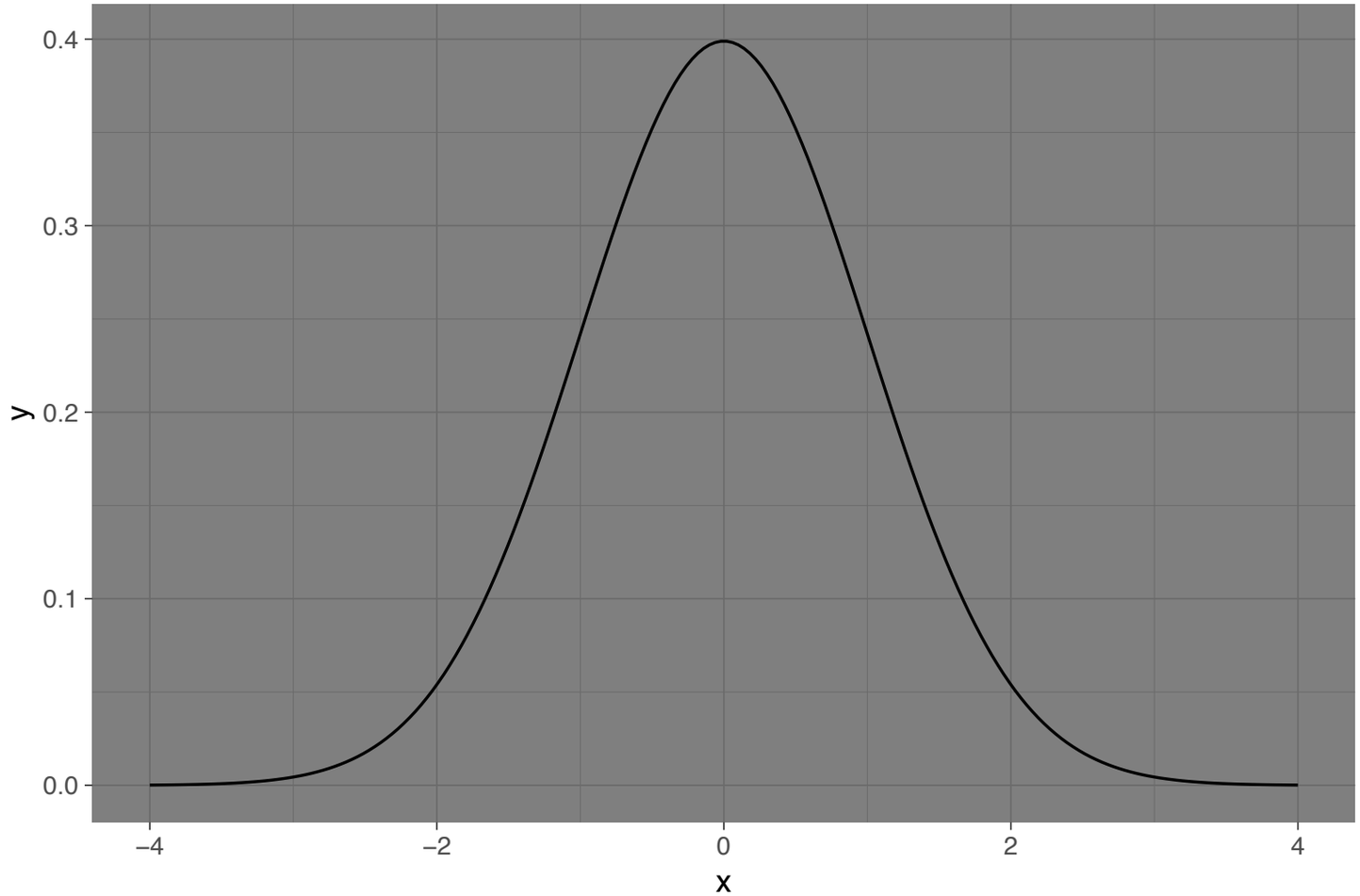
```
ggplot(data) +
   geom_line(aes(x = x, y = y))
```



ggplot2

- Are many different themes in ggplot2 that influence many aspects of how a plot looks

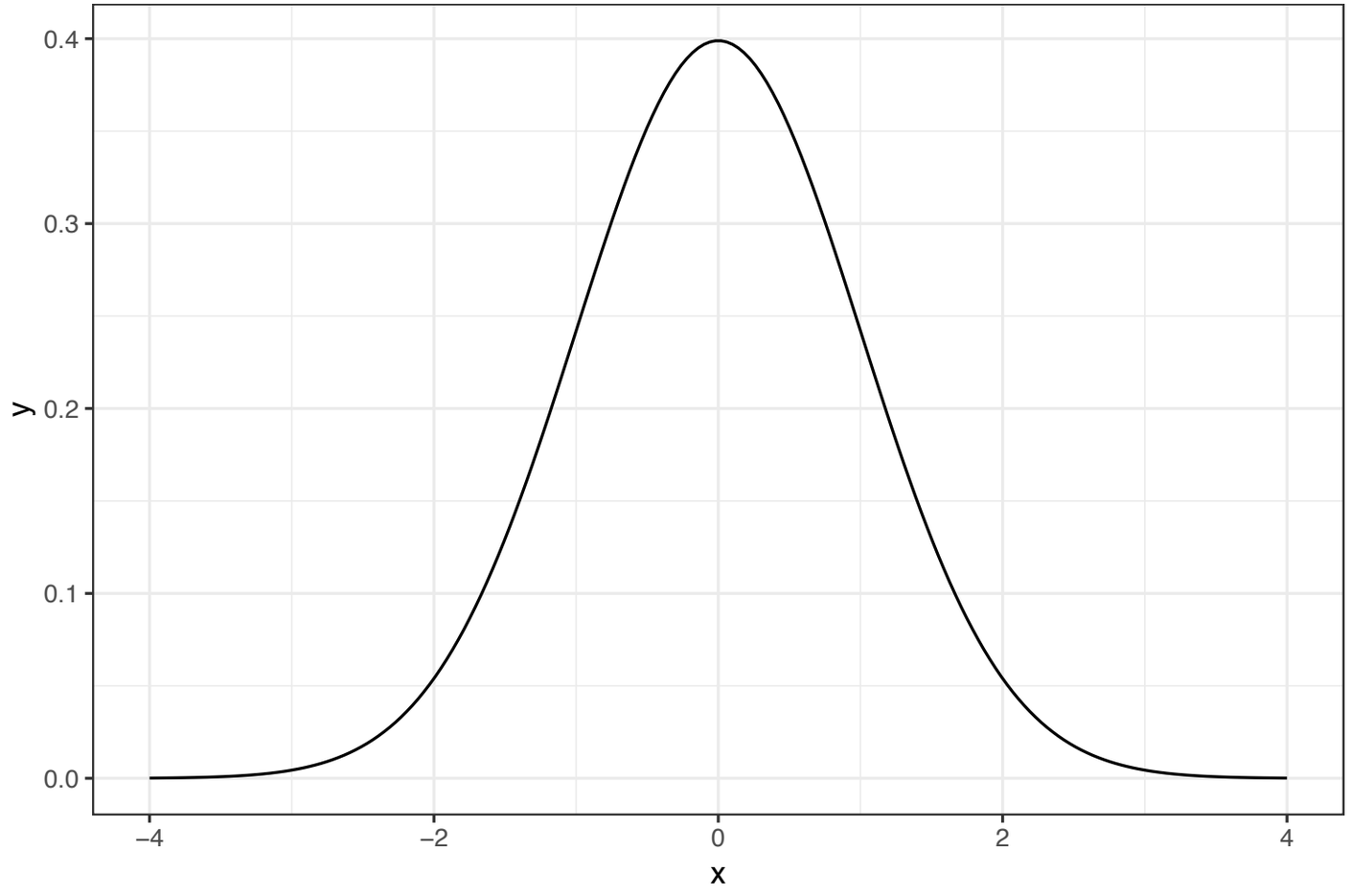
```
ggplot(data) +
   theme_dark() +
   geom_line(aes(x = x, y = y))
```



ggplot2

- Are many different themes in ggplot2 that influence many aspects of how a plot looks

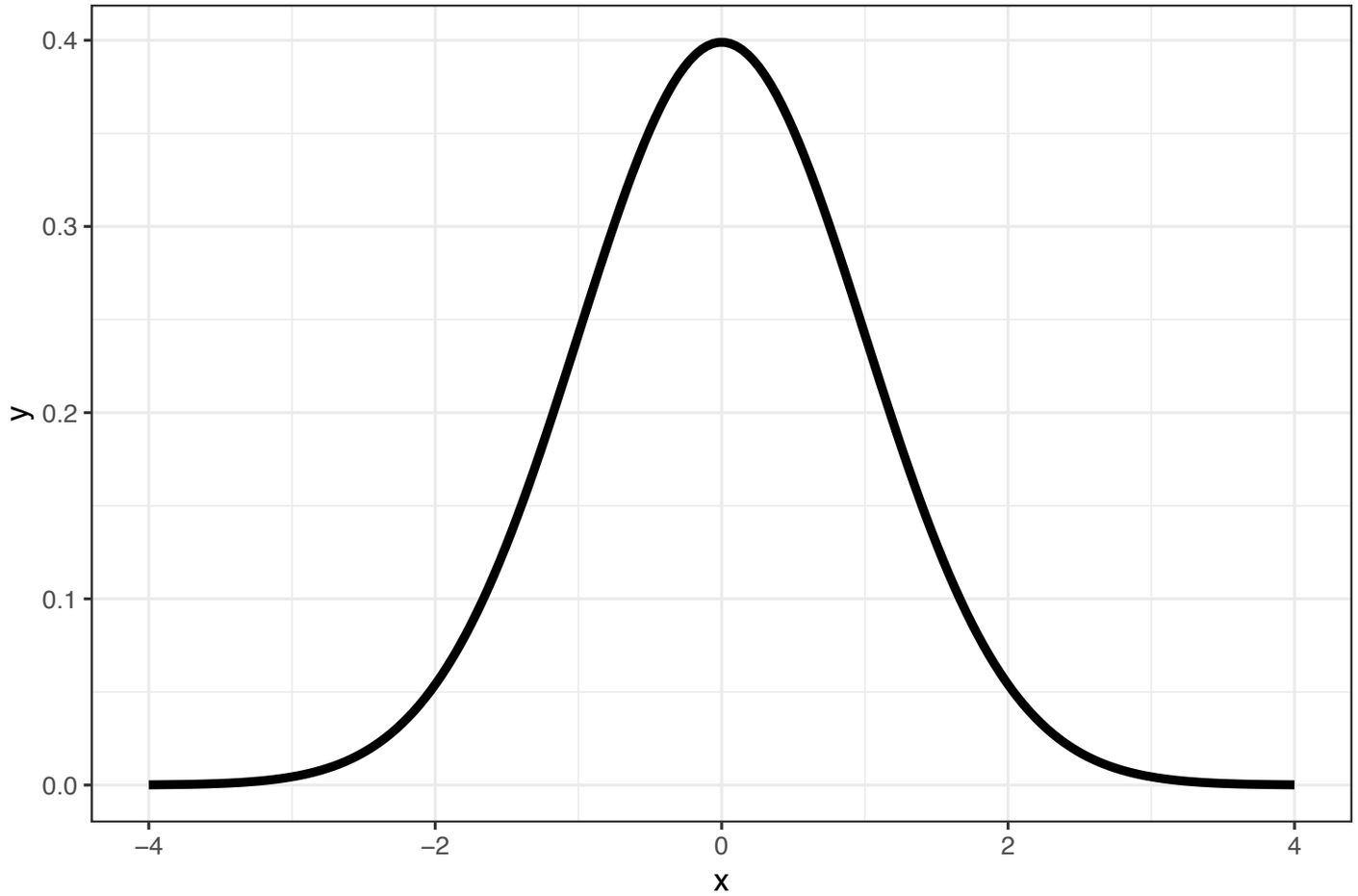
```
ggplot(data) +
   theme_bw() +
   geom_line(aes(x = x, y = y))
```



ggplot2

- Let's make the line a bit thicker

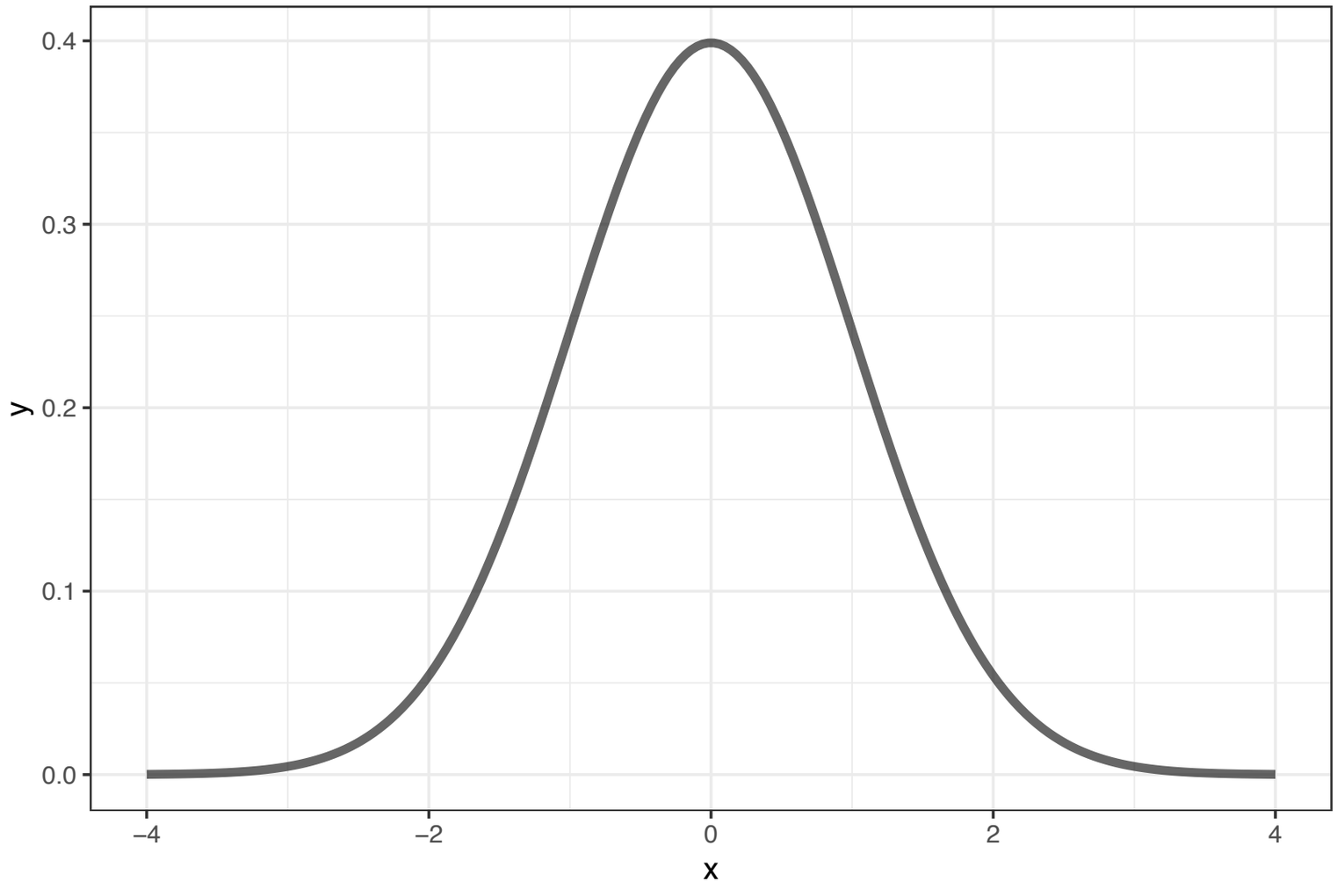
```
ggplot(data) +
   theme_bw() +
   geom_line(aes(x = x, y = y), size = 1.5)
```



ggplot2

- Let's make it slightly transparent

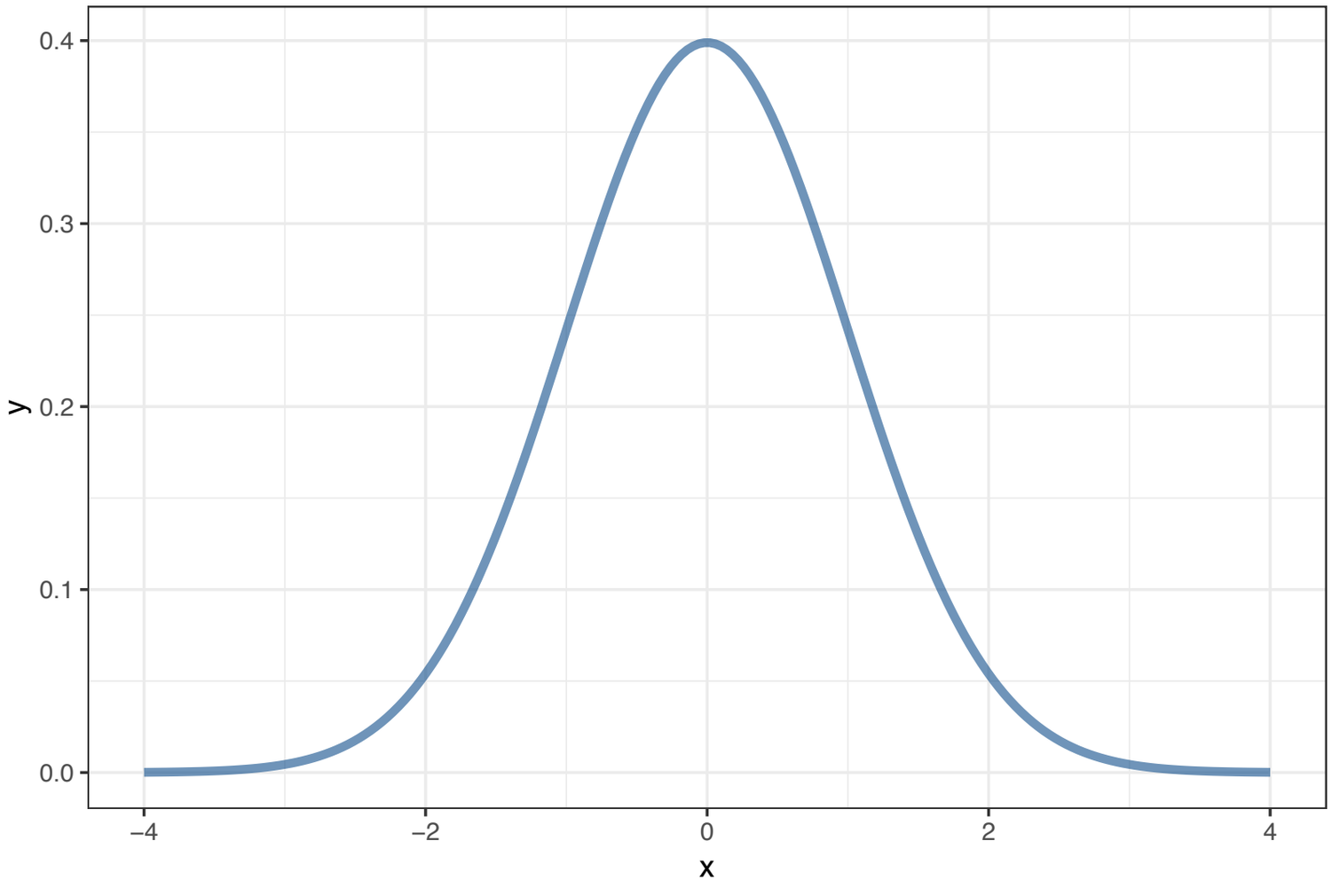
```
ggplot(data) +
   theme_bw() +
   geom_line(aes(x = x, y = y), size = 1.5, alpha = 0.6)
```



ggplot2

- Lastly, let's customize the colour

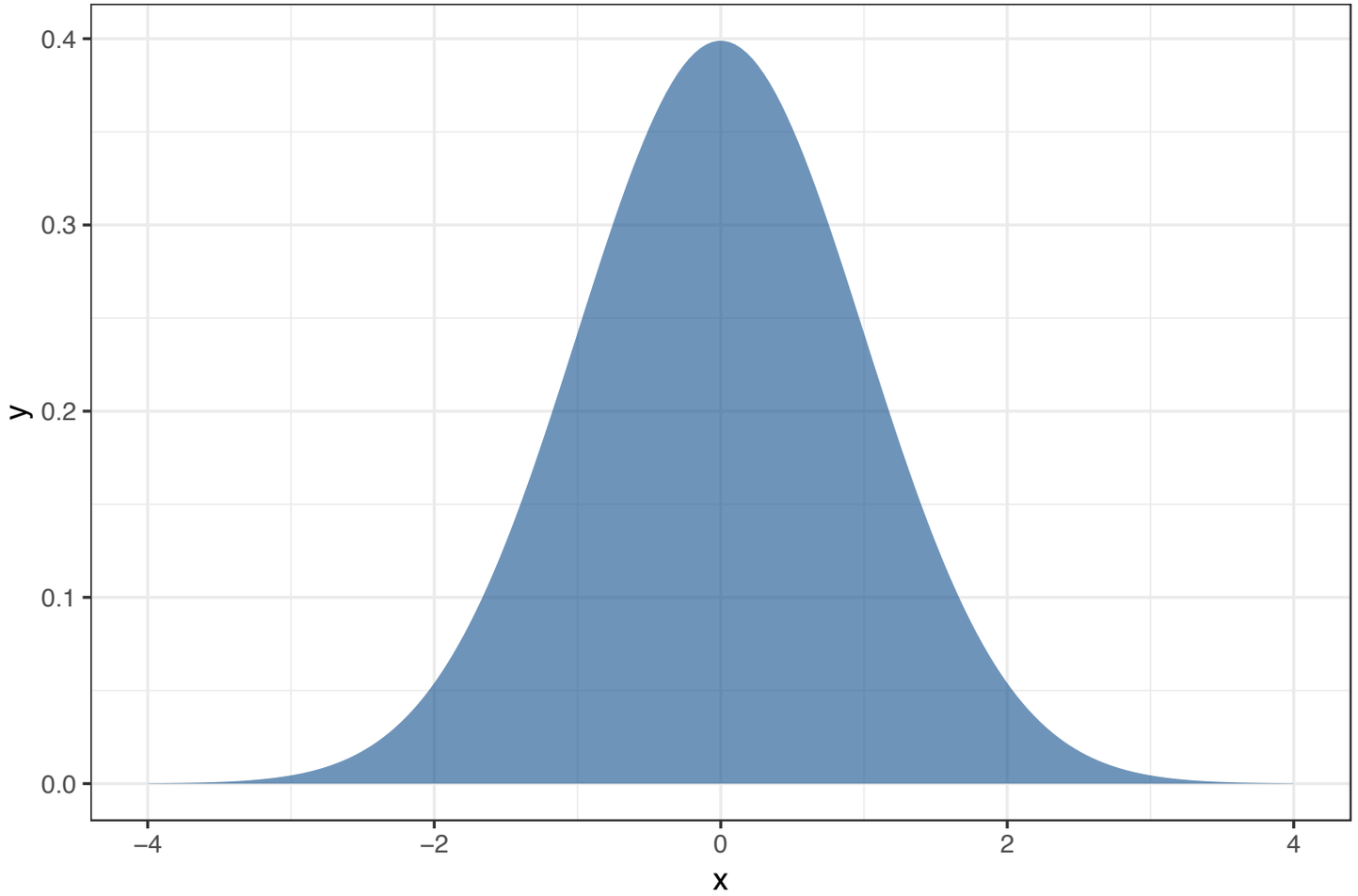
```
ggplot(data) +
   theme_bw() +
   geom_line(aes(x = x, y = y), colour = "dodgerblue4", size = 1.5, alpha
   = 0.6)
```



ggplot2

- Let's plot it as an "area" plot
  - ymax = y values, ymin = 0

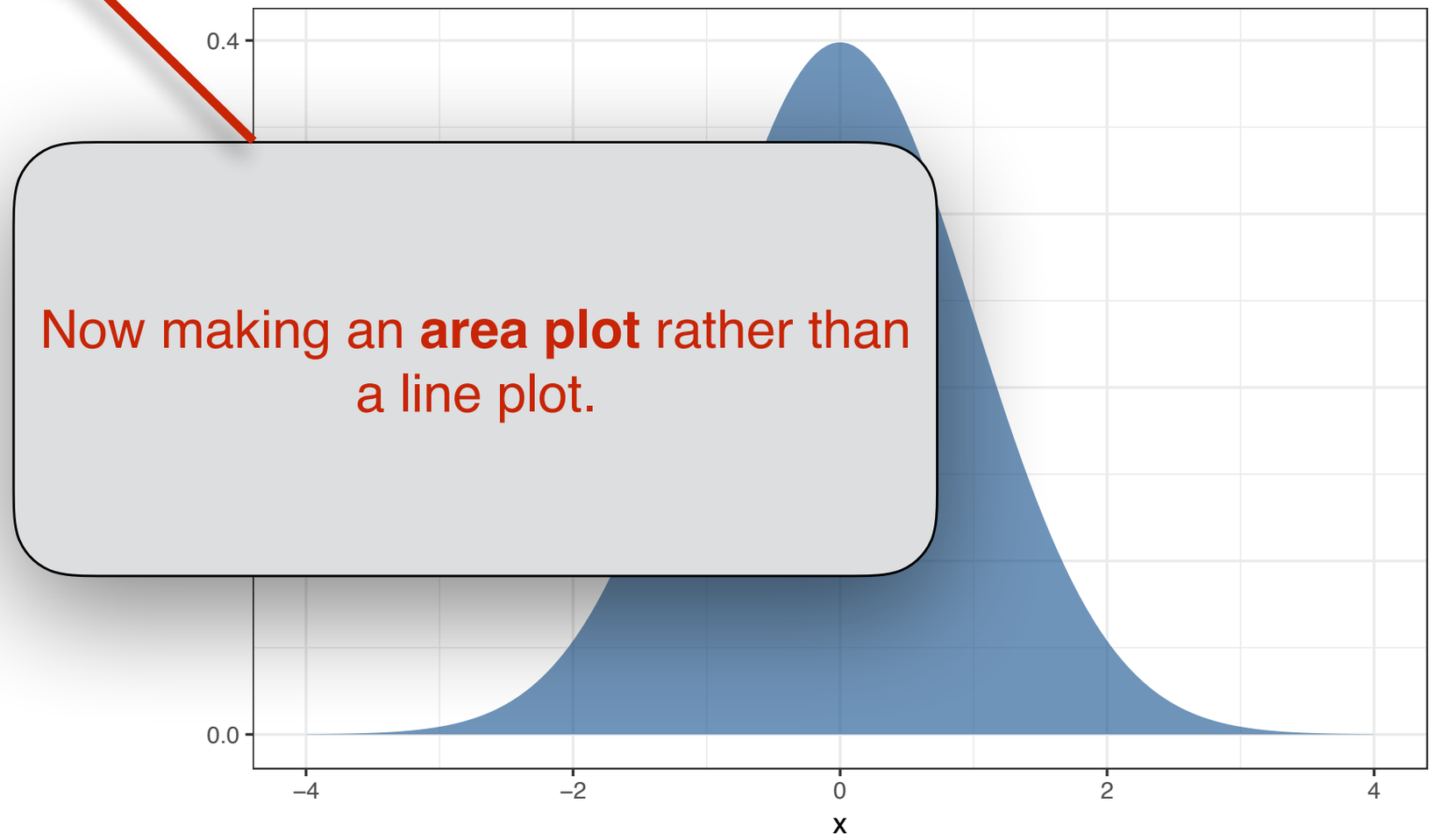
```
ggplot(data) +
   theme_bw() +
   geom_area(aes(x = x, y = y), fill = "dodgerblue4", alpha = 0.6)
```



ggplot2

- Let's plot it as an "area" plot
  - ymax = y values, ymin = 0

```
ggplot(data) +
   theme_bw() +
   geom_area(aes(x = x, y = y), fill = "dodgerblue4", alpha = 0.6)
```

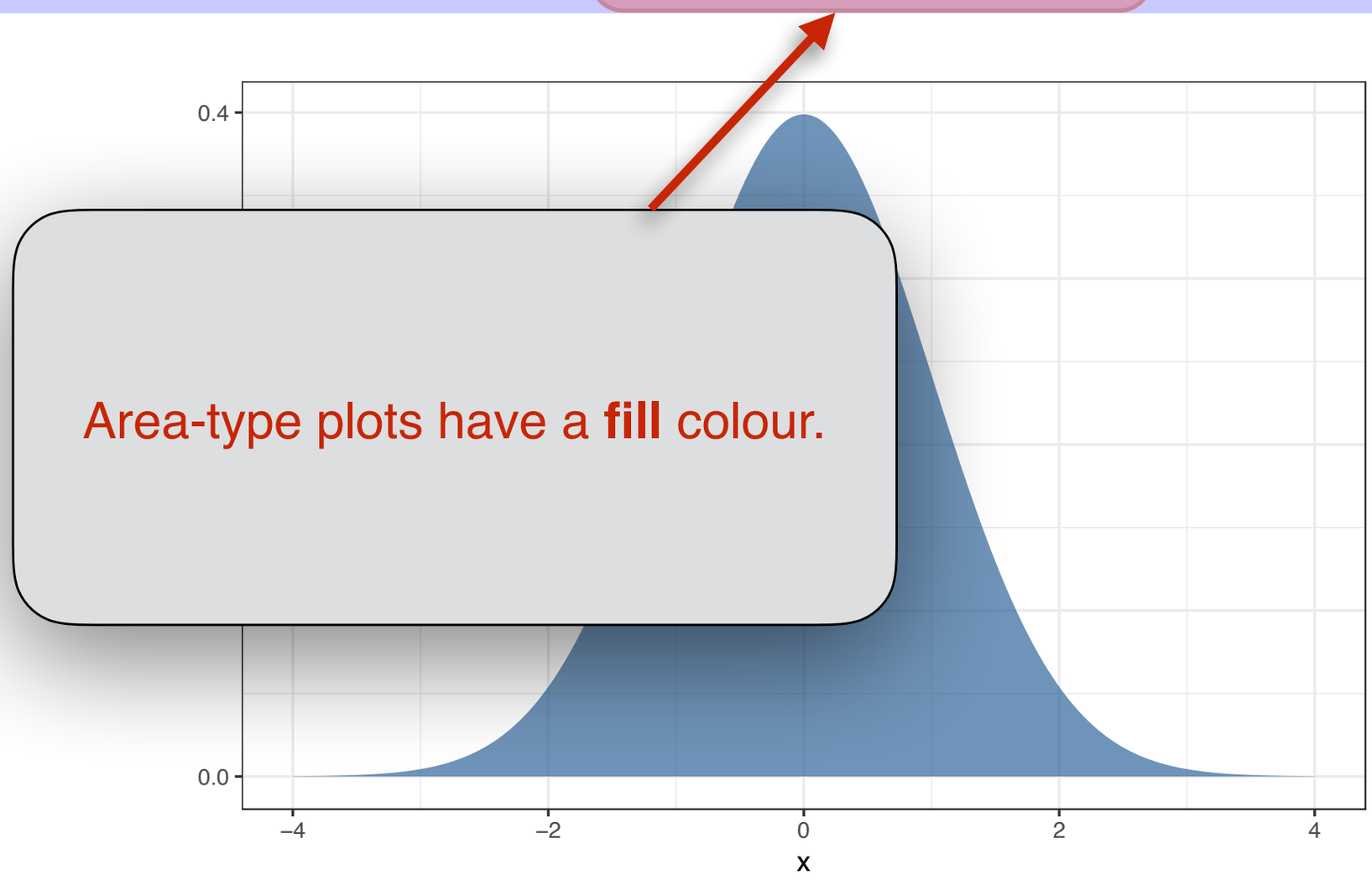


Now making an **area plot** rather than a line plot.

ggplot2

- Let's plot it as an "area" plot
  - ymax = y values, ymin = 0

```
ggplot(data) +
   theme_bw() +
   geom_area(aes(x = x, y = y), fill = "dodgerblue4", alpha = 0.6)
```

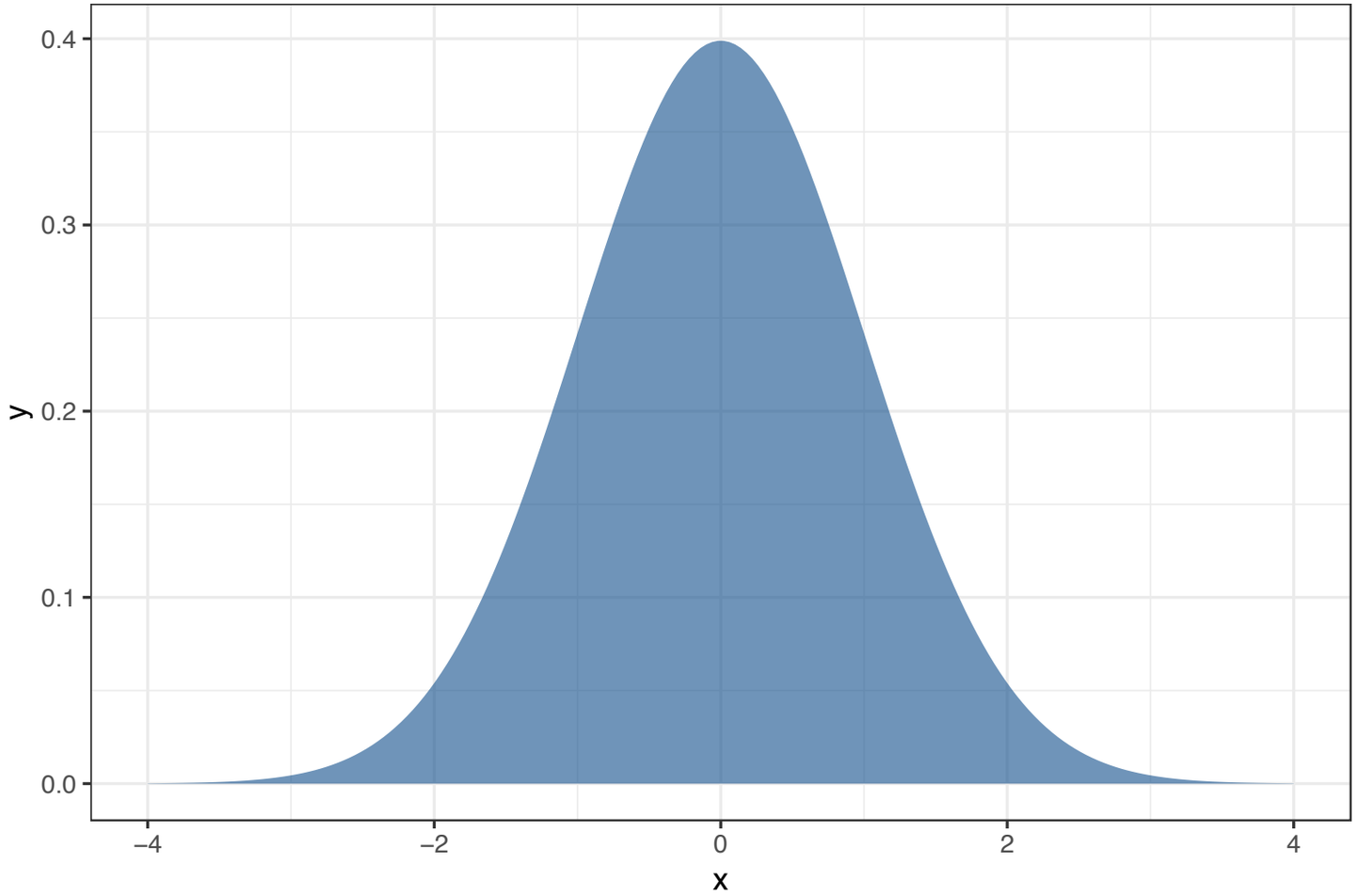


Area-type plots have a **fill** colour.

ggplot2

- Let's plot it as an "area" plot
  - ymax = y values, ymin = 0

```
ggplot(data) +
   theme_bw() +
   geom_area(aes(x = x, y = y), fill = "dodgerblue4", alpha = 0.6)
```



Google “Colors in R”

Probability Distributions

Uniform distribution

- A flat distribution (all values have the same probability)
- Range is -∞ to ∞ (you specify range)
- Has two parameters:
  - min: minimum value to consider
  - max: maximum value to consider

Probability Distributions

Uniform distribution

- In R, use the `dunif` function

```
dunif(x, min, max)
```

Probability Distributions

Uniform distribution

- In R, use the `dunif` function

```
dunif(x, min, max)
```

```
x = seq(from = -4, to = 4, length = 200)
y = dunif(x, min = -4, max = 4)
```

Probability Distributions

Uniform distribution

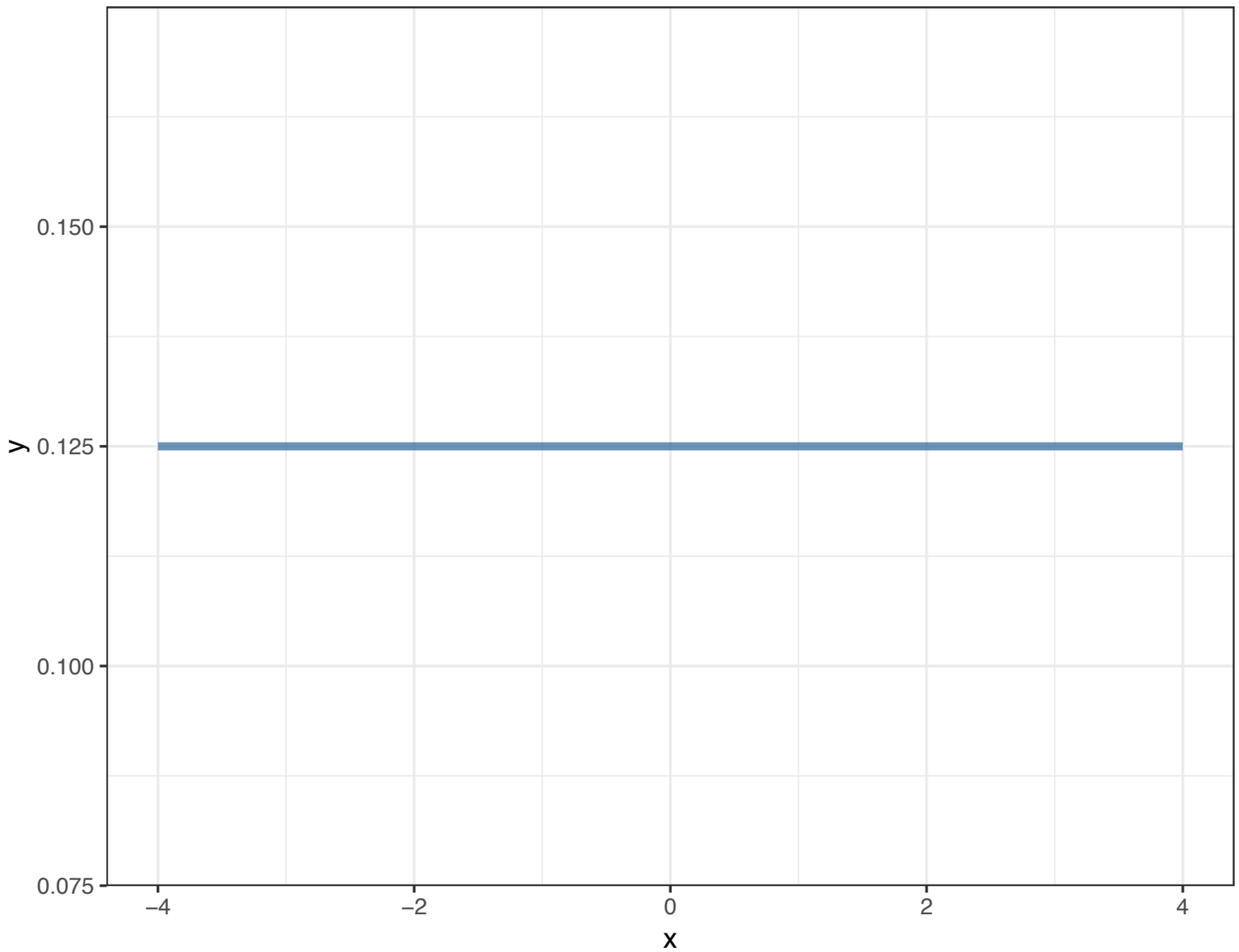
- Organize and plot the data

```
data = as.data.frame(cbind(x, y))

ggplot(data) +
   theme_bw() +
   geom_line(aes(x = x, y = y), colour = “dodgerblue4”, size = 1.5, alpha
   = 0.6)
```

Probability Distributions

Uniform distribution



Probability Distributions

Uniform distribution

- In Stan, function is called `uniform`, requires:
  - `alpha` = minimum value
  - `beta` = maximum value

```
uniform(alpha, beta)
```

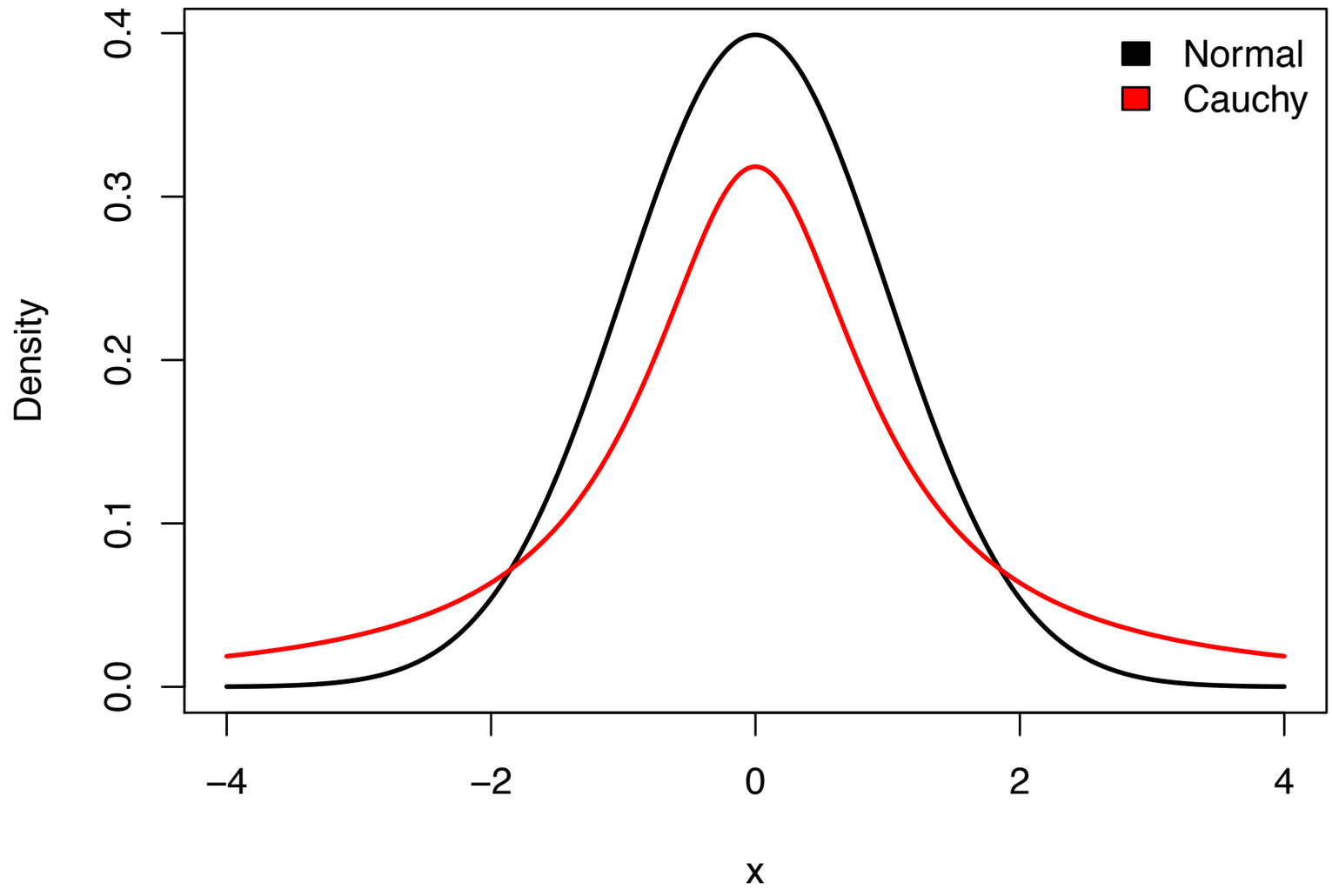
Probability Distributions

Cauchy distribution (pronounced *kō shee*)

- Like the normal distribution, but with heavier tails
- Symmetric around the mean (mean, median, and mode are all equal)
- Range is (-∞ to ∞)
- Has two parameters:
  - location: determines position of peak along x-axis
  - scale: determines how wide the peak is around the location

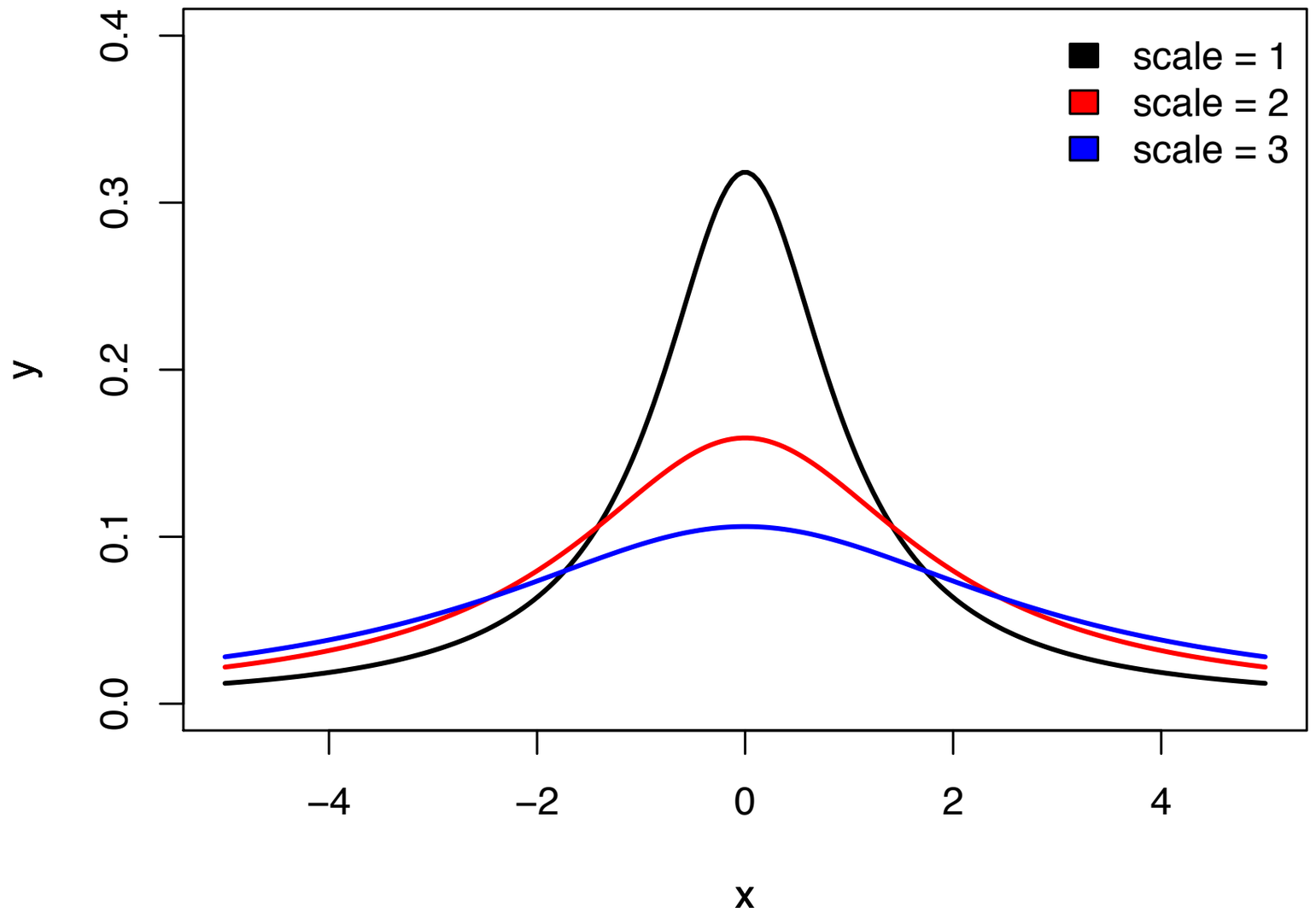
Probability Distributions

Cauchy distribution



Probability Distributions

Cauchy distribution



Probability Distributions

Cauchy distribution

- In R, use the `dcauchy` function

```
dcauchy(x, location, scale)
```

```
x = seq(from = -4, to = 4, length = 200)
y = dcauchy(x, location = 0, scale = 1)
```

Probability Distributions

Cauchy distribution

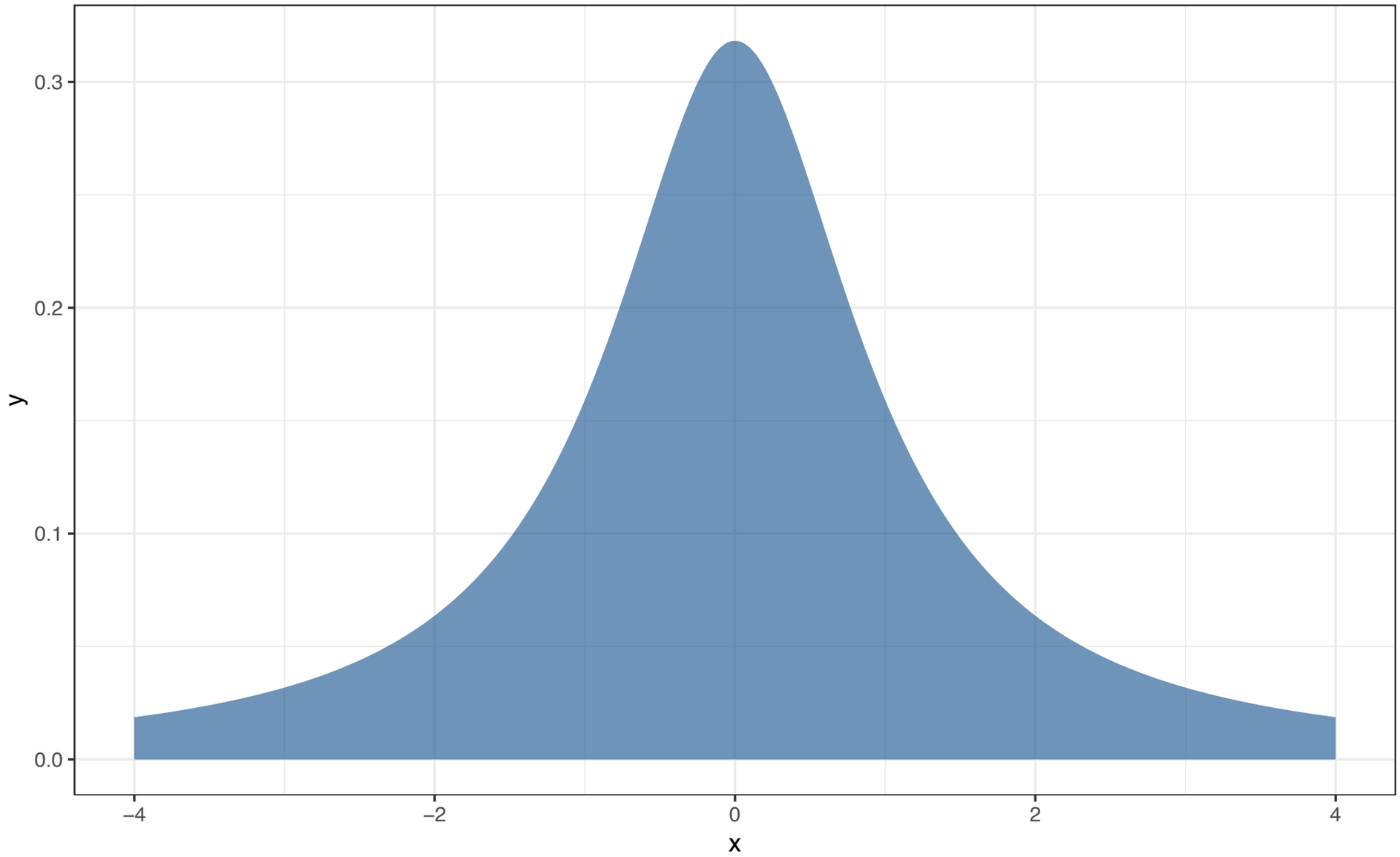
- Organize and plot the data (*exactly* the same as before)

```
data = as.data.frame(cbind(x, y))

ggplot(data) +
   theme_bw() +
   geom_area(aes(x = x, y = y), fill = “dodgerblue4”, alpha = 0.6)
```

Probability Distributions

Cauchy distribution



Probability Distributions

Cauchy distribution

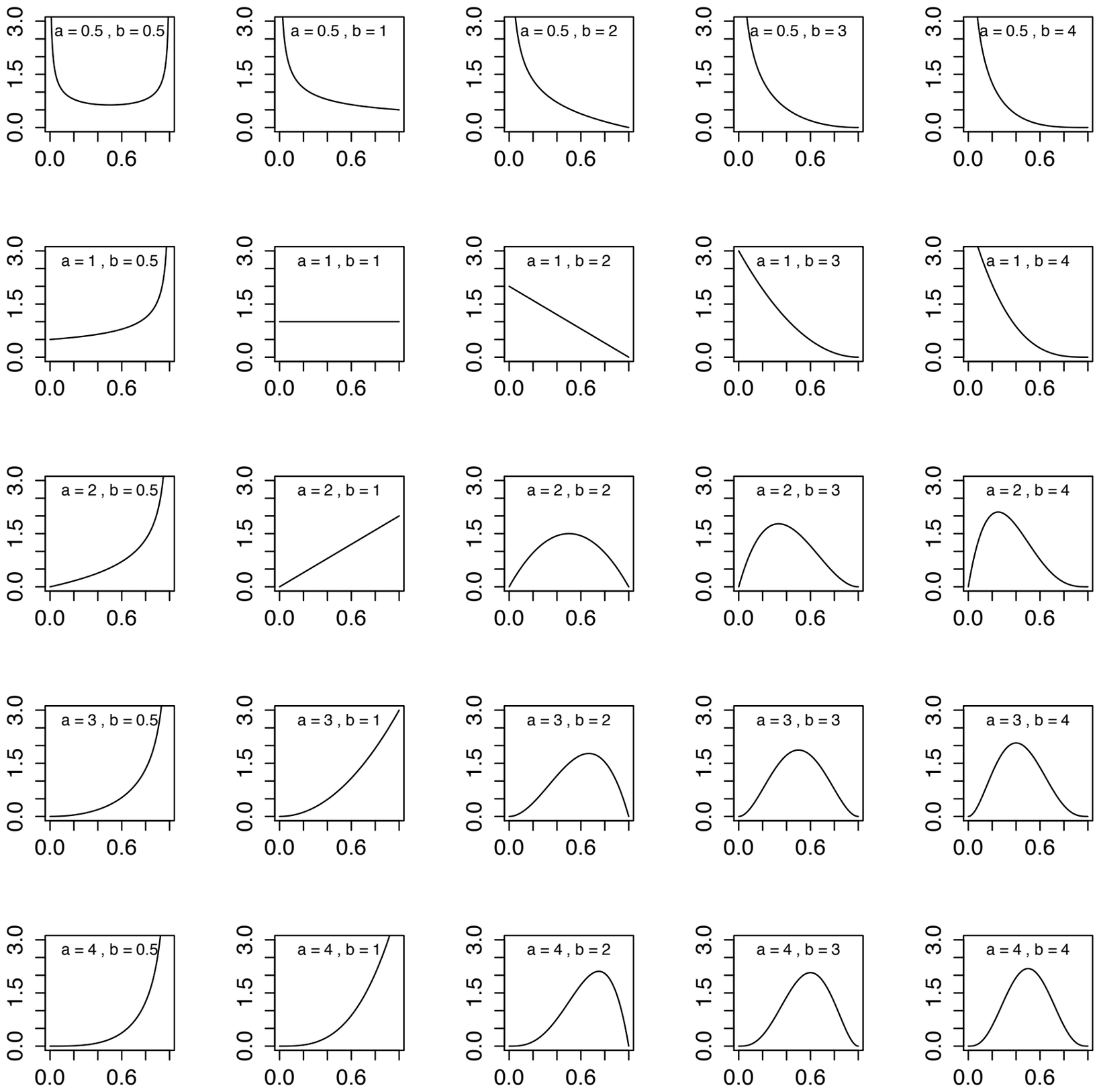
- In Stan, function is called `cauchy`, requires:
  - `location`
  - `scale`

```
cauchy(location, scale)
```

Probability Distributions

Beta distribution

- Can take on almost any shape, determined by two positive shape parameters
- Range is [0 to 1]
- Has two parameters:
  - First shape parameter (a): determines shape of curve
  - Second shape parameter (b): determines shape of curve


a = 0.5 , b = 0.5
a = 0.5 , b = 1
a = 0.5 , b = 2
a = 0.5 , b = 3
a = 0.5 , b = 4
a = 1 , b = 0.5
a = 1 , b = 1
a = 1 , b = 2
a = 1 , b = 3
a = 1 , b = 4
a = 2 , b = 0.5
a = 2 , b = 1
a = 2 , b = 2
a = 2 , b = 3
a = 2 , b = 4
a = 3 , b = 0.5
a = 3 , b = 1
a = 3 , b = 2
a = 3 , b = 3
a = 3 , b = 4
a = 4 , b = 0.5
a = 4 , b = 1
a = 4 , b = 2
a = 4 , b = 3
a = 4 , b = 4
0.0
1.5
3.0
0.6

Probability Distributions

Beta distribution

- In R, use the `dbeta` function

```
dbeta(x, shape1, shape2)
```

Probability Distributions

Beta distribution

- In R, use the `dbeta` function

```
dbeta(x, shape1, shape2)
```

```
x = seq(from = 0, to = 1, length = 200)
y = dbeta(x, shape1 = 2, shape2 = 3)
```

Probability Distributions

Beta distribution

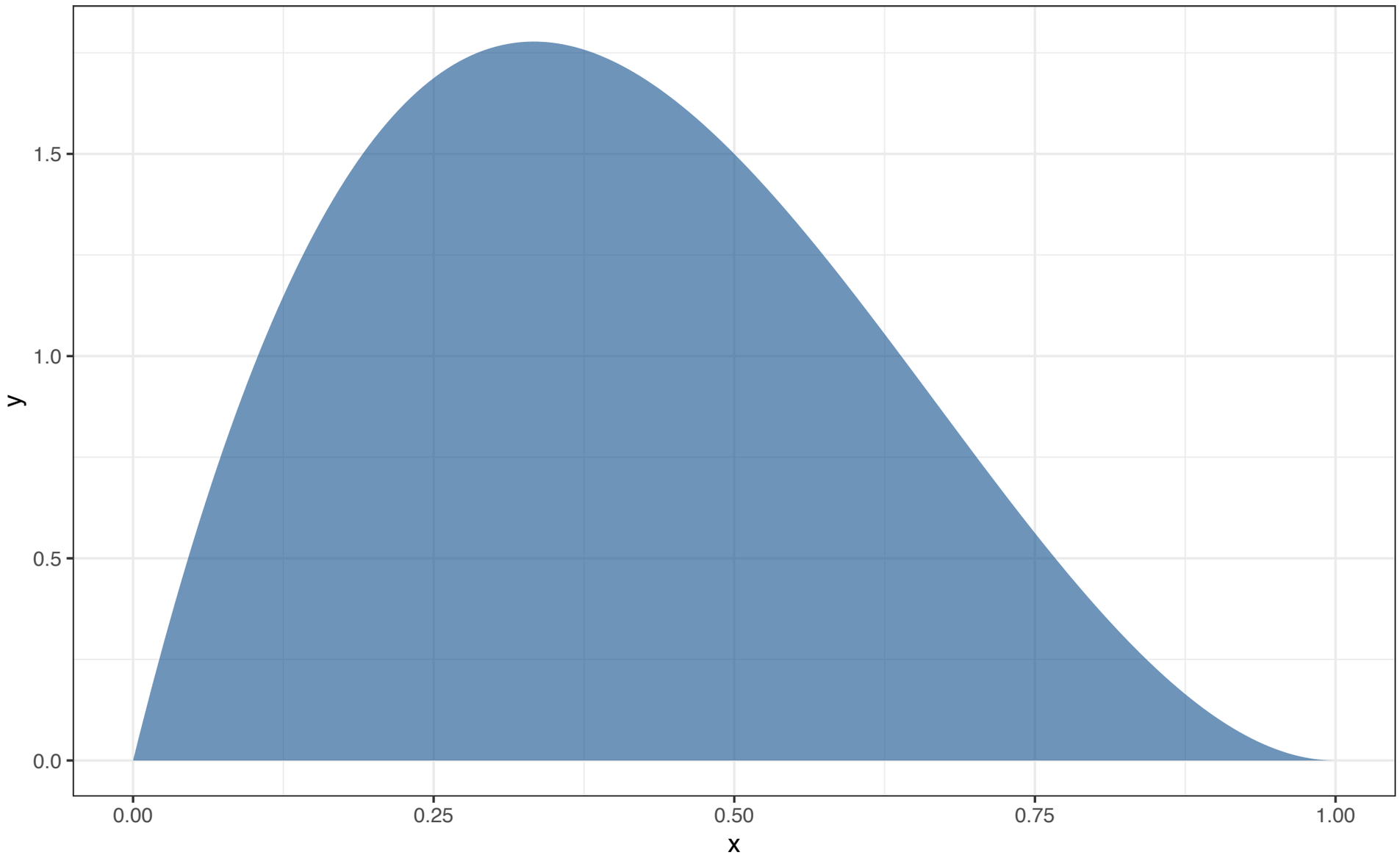
- Organize and plot the data

```
data = as.data.frame(cbind(x, y))

ggplot(data) +
   theme_bw() +
   geom_area(aes(x = x, y = y), fill = “dodgerblue4”, alpha = 0.6)
```

Probability Distributions

Beta distribution



Probability Distributions

Beta distribution

- In Stan, function is called `beta`, requires:
  - `alpha` = first shape parameter
  - `beta` = second shape parameter

```
beta(alpha, beta)
```

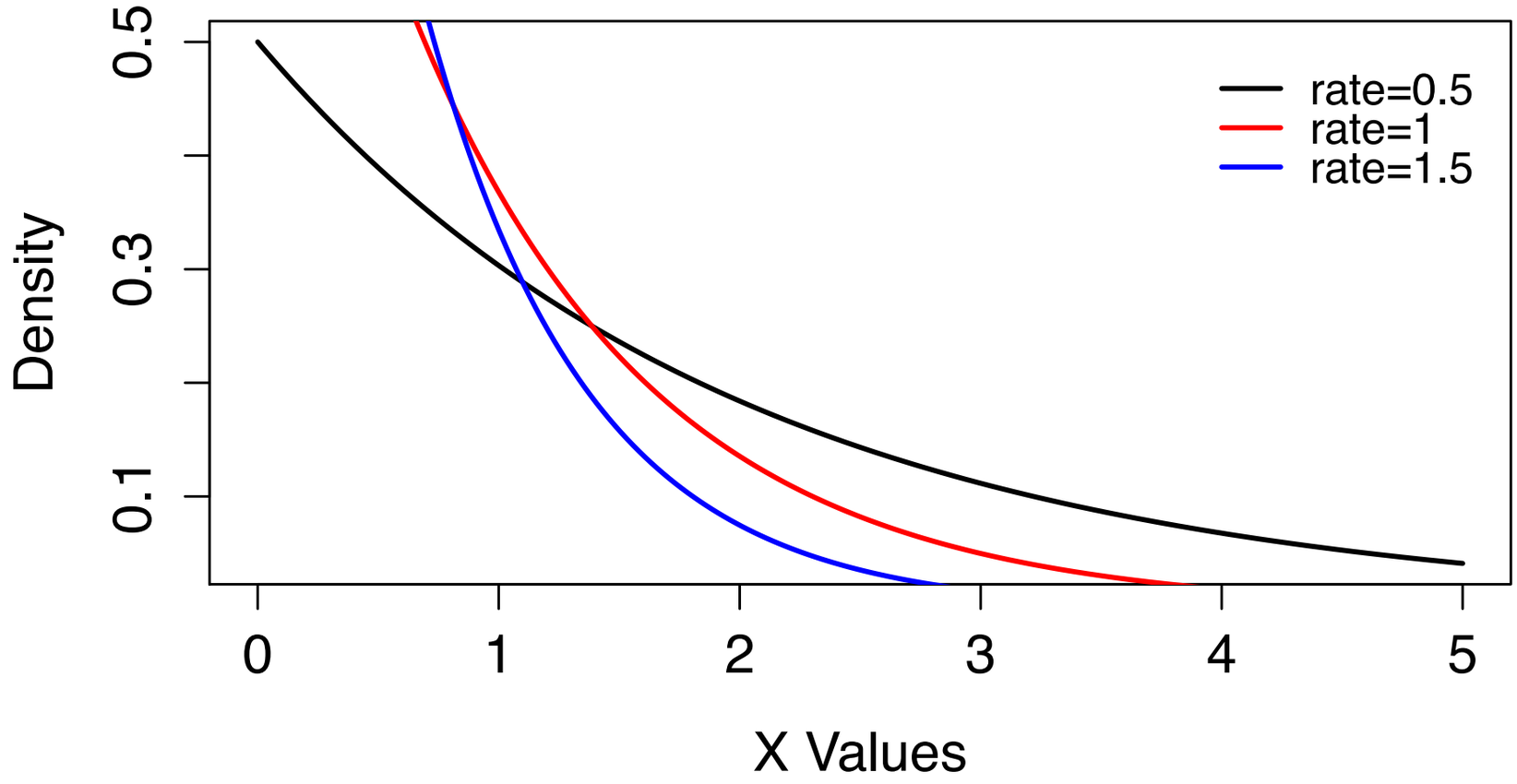
Probability Distributions

Exponential distribution

- An L-shaped distribution
- Range is 0 to ∞
- Has one parameter:
  - rate: determines shape of the curve

Probability Distributions

Exponential distribution



Probability Distributions

Exponential distribution

- In R, use the `dexp` function

```
dexp(x, rate)
```

Probability Distributions

Exponential distribution

- In R, use the `dexp` function

```
dexp(x, rate)
```

```
x = seq(from = 0, to = 20, length = 200)
y = dexp(x, rate = 0.5)
```

Probability Distributions

Exponential distribution

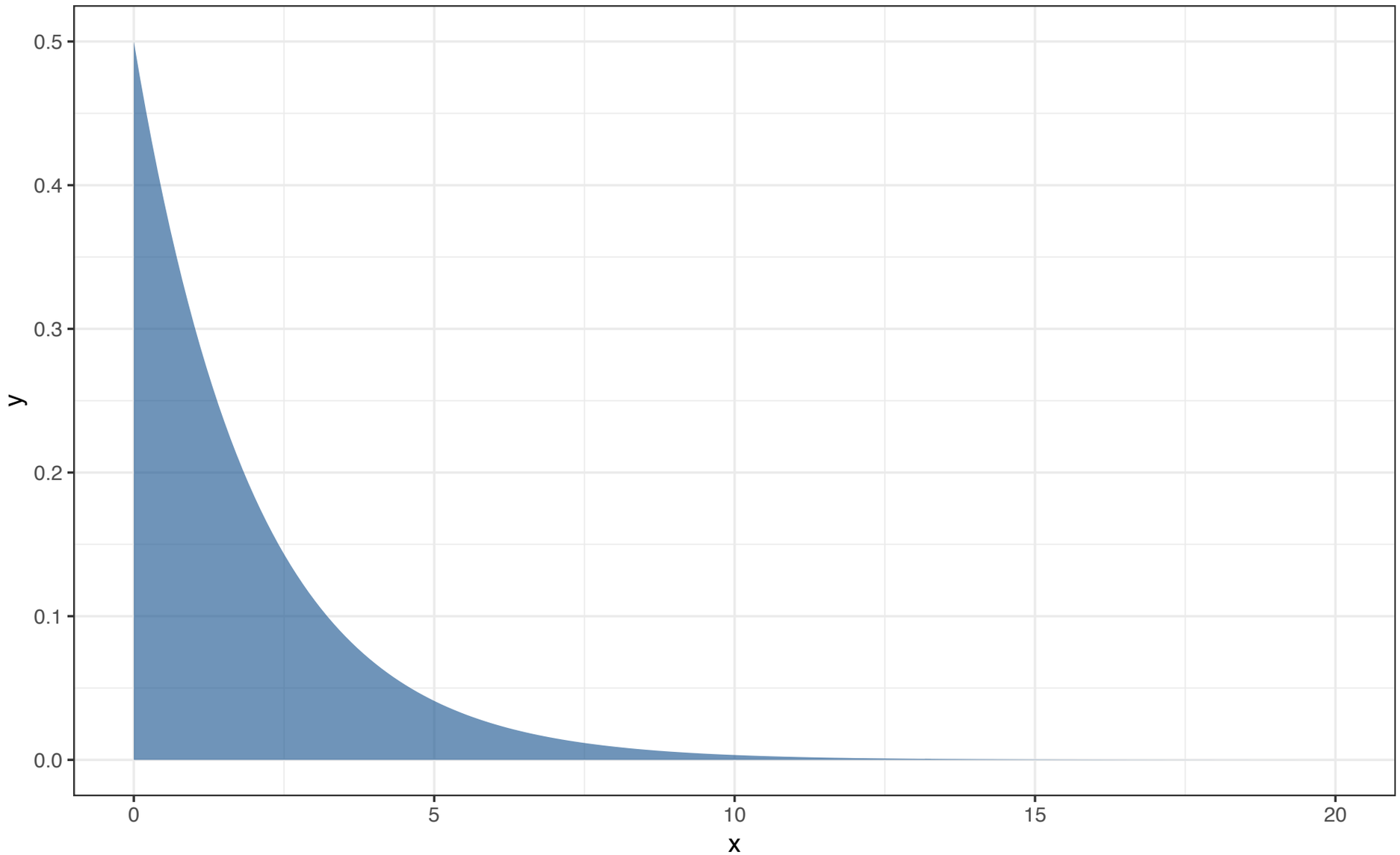
- Organize and plot the data

```
data = as.data.frame(cbind(x, y))

ggplot(data) +
   theme_bw() +
   geom_area(aes(x = x, y = y), fill = “dodgerblue4”, alpha = 0.6)
```

Probability Distributions

Exponential distribution



Probability Distributions

Exponential distribution

- In Stan, function is called `exponential`, requires:
  - `beta` = rate value

```
exponential(beta)
```

Discrete Distributions

Probability Distributions

Binomial distribution

- Used when you have multiple trials, and only two results possible: 0 or 1 (sometimes “failure” or “success”)

- Range is integers from [0 to ∞)
  - # of “successes”

- Has two parameters:
  - Size: The number of trials
  - Prob: the probability of “success”

Probability Distributions

Binomial distribution

- In R, use the `dbinom` function

```
dbinom(x, size, prob)
```

Probability Distributions

Binomial distribution

- In R, use the `dbinom` function

```
dbinom(x, size, prob)
```

```
x = 1:50
y = dbinom(x, size = 50, prob = 0.5)
```

Probability Distributions

Binomial distribution

- Organize and plot the data

```
data = as.data.frame(cbind(x, y))

ggplot(data) +
   theme_bw() +
   geom_col(aes(x = x, y = y), fill = “dodgerblue4”, alpha = 0.6)
```

Probability Distributions

Binomial distribution

- Organize and plot the data

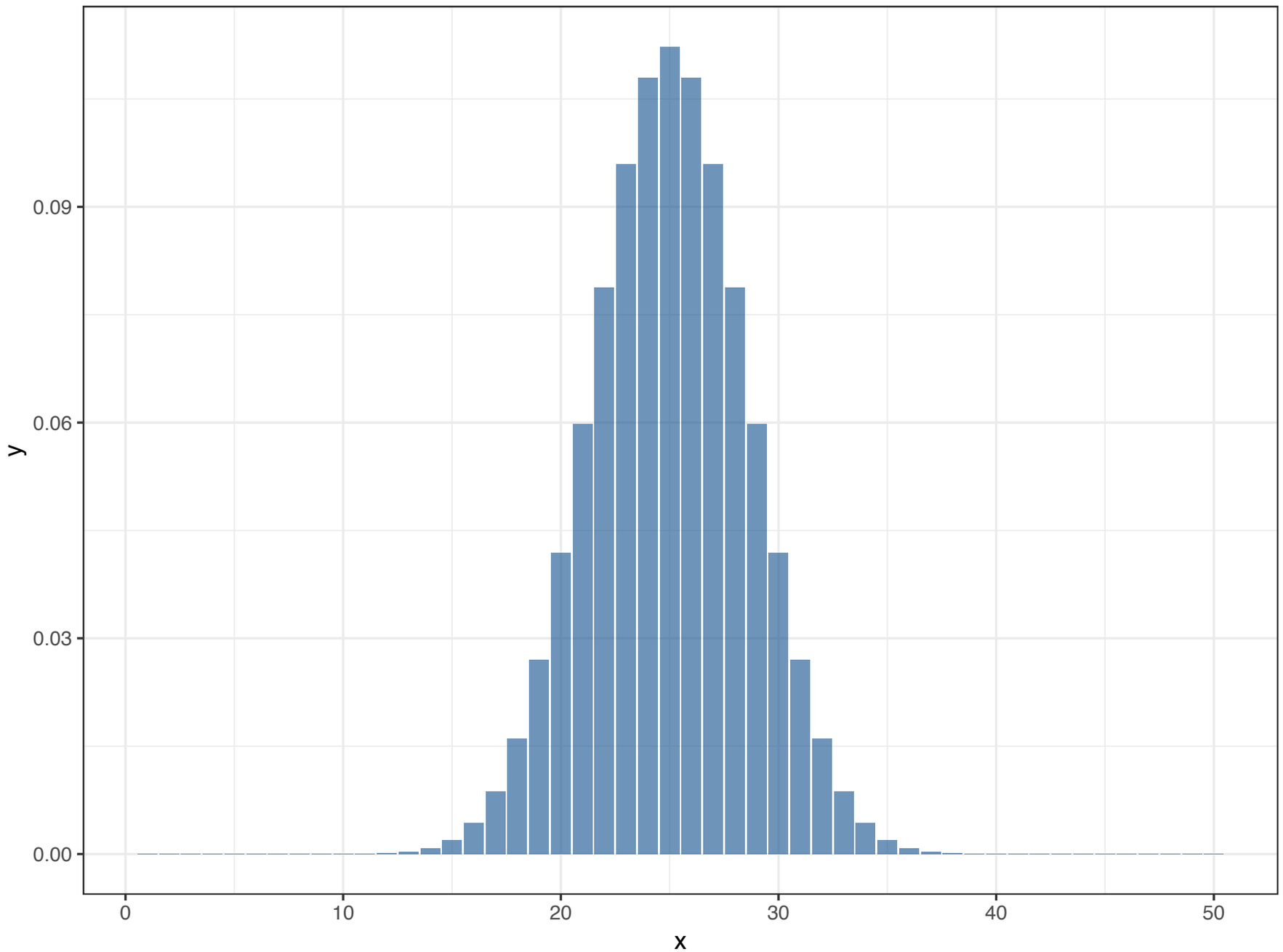
```
data = as.data.frame(cbind(x, y))

ggplot(data) +
   theme_bw() +
   geom_col(aes(x = x, y = y), fill = “dodgerblue4”, alpha = 0.6)
```

Since we have discrete data (integers), area curve no longer appropriate.

Probability Distributions

Binomial distribution



Probability Distributions

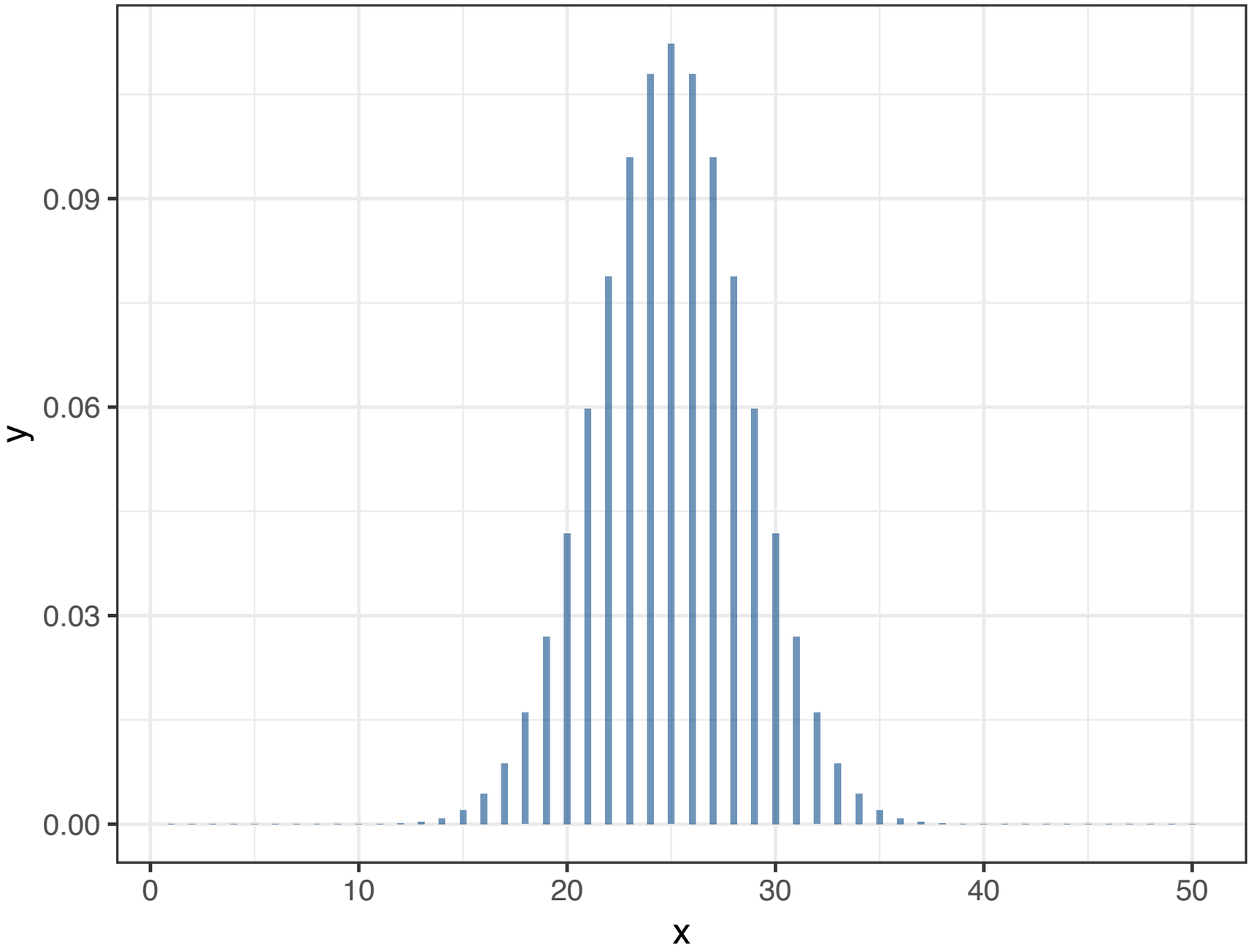
Binomial distribution

- Can change the width of the bars, if desired

```
ggplot(data) +
   theme_bw() +
   geom_col(aes(x = x, y = y), fill = “dodgerblue4”, alpha = 0.6, width =
   0.3)
```

Probability Distributions

Binomial distribution



Probability Distributions

Binomial distribution

- In Stan, function is called `binomial`, requires:
  - `N` = number of trials
  - `theta` = probability of "success"

```
binomial(N, theta)
```

Probability Distributions

Bernoulli distribution

- One trial of a binomial situation
  - For a case-by-case analysis of a binomial process

- Many of our models could be parameterized as:
  - One group of results (size = *N*) from a binomial process; or
  - A series (length = *N*) of Bernoulli processes

Probability Distributions

Bernoulli distribution

- In Stan, function is called `bernoulli`, requires:
  - `theta` = probability of "success"

```
bernoulli(theta)
```

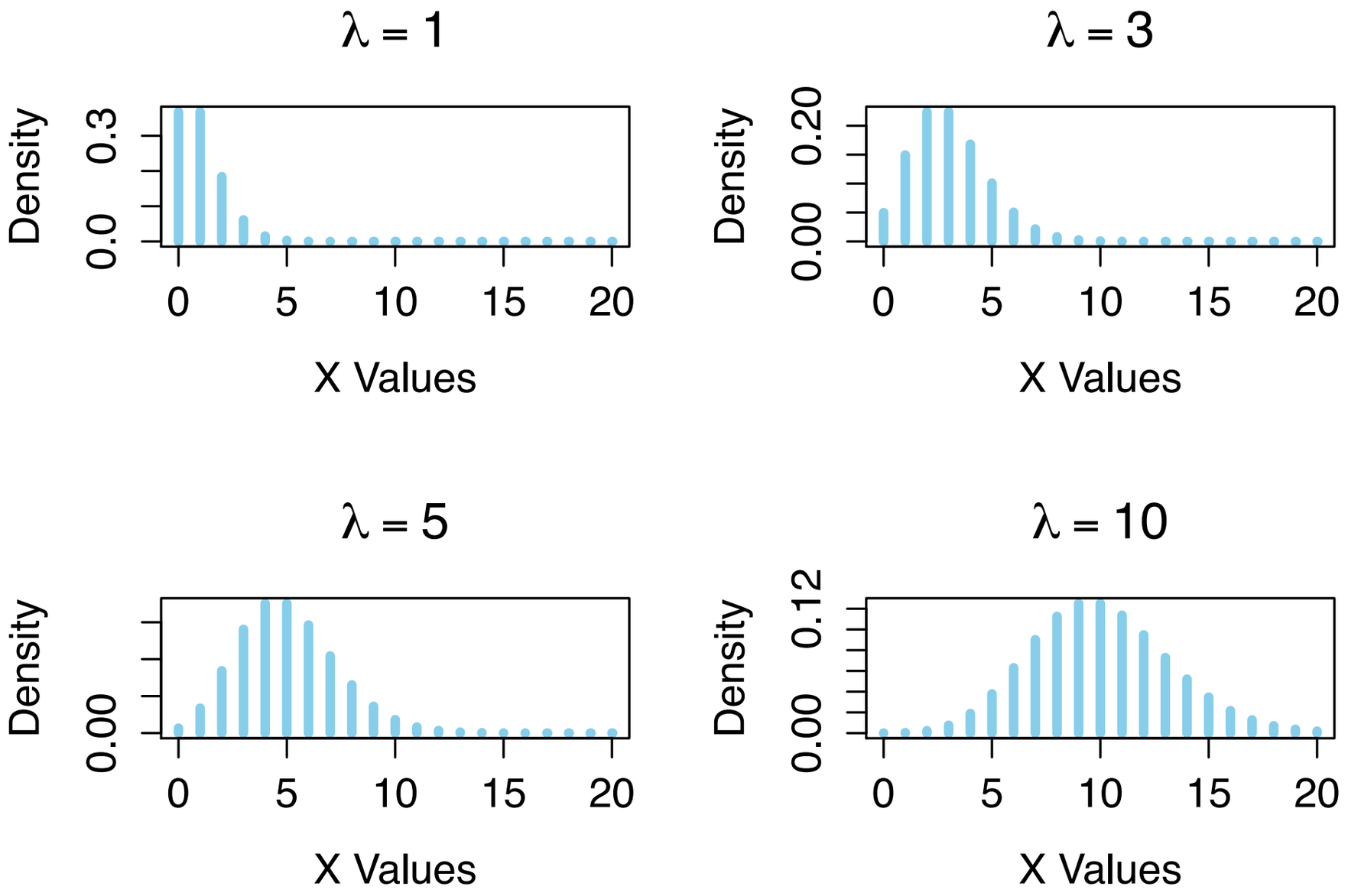
Probability Distributions

Poisson distribution

- Can range from L-shaped to a normal distribution
- Range is integers from 0 to ∞
  - Discrete values, not a continuous distribution
- Has one parameter:
  - lamdba: determines shape of the curve

Probability Distributions

Poisson distribution



Probability Distributions

Poisson distribution

- In R, use the `dpois` function

```
dpois(x, lambda)
```

Probability Distributions

Poisson distribution

- In R, use the `dpois` function

```
dpois(x, lambda)
```

```
x = seq(from = 0, to = 20, by = 1)
y = dpois(x, lambda = 4)
```

Probability Distributions

Poisson distribution

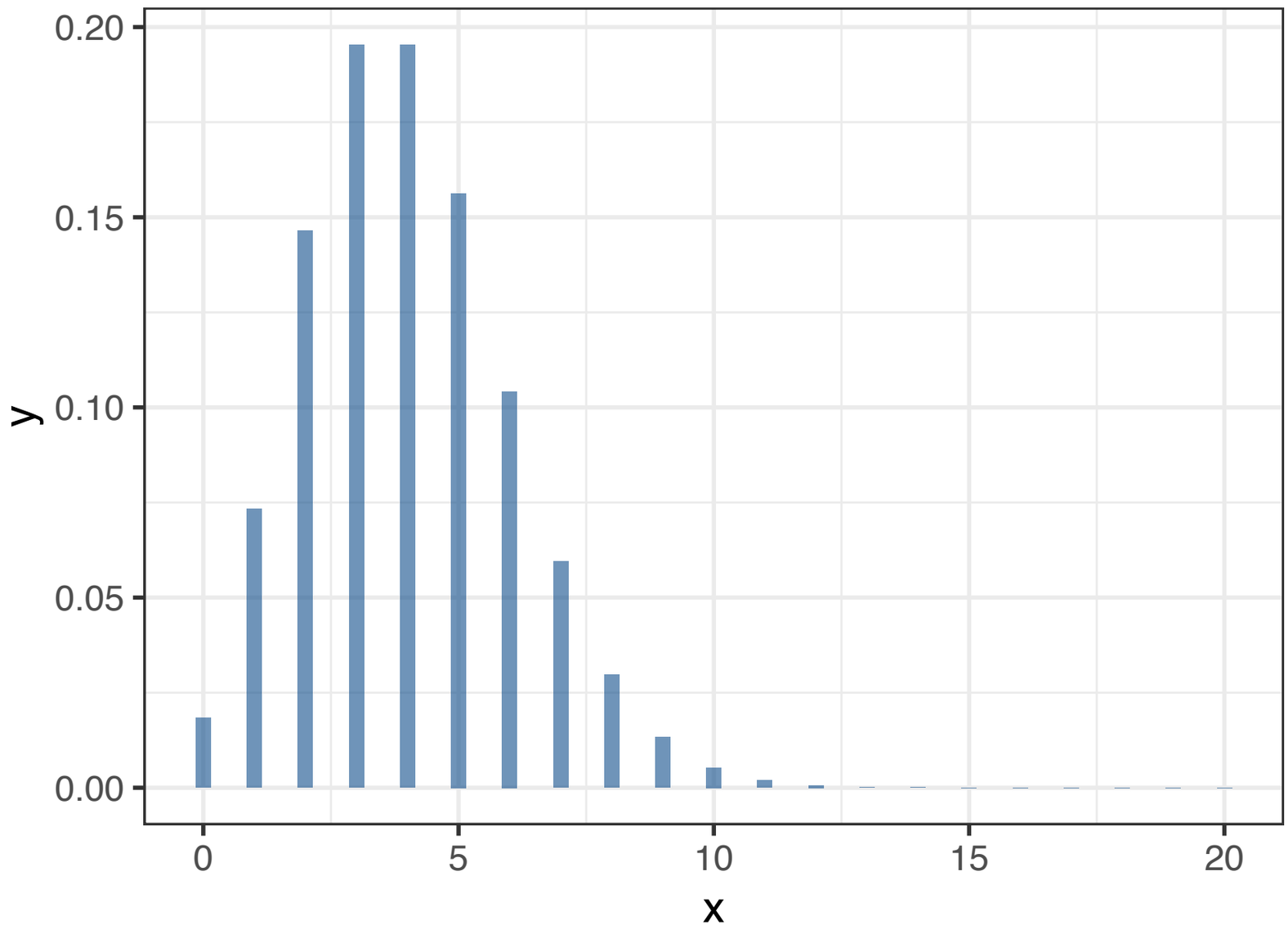
- Organize and plot the data

```
data = as.data.frame(cbind(x, y))

ggplot(data) +
   theme_bw() +
   geom_col(aes(x = x, y = y), fill = “dodgerblue4”, alpha = 0.6, width =
   0.3)
```

Probability Distributions

Poisson distribution



Probability Distributions

Poisson distribution

- In Stan, function is called `poisson`, requires:
  - `lambda (λ)` = shape value

```
poisson(lambda)
```

Probability Distributions

- See Stan manual for more, and for their proper usage

Questions?