Nonlinear Relationships

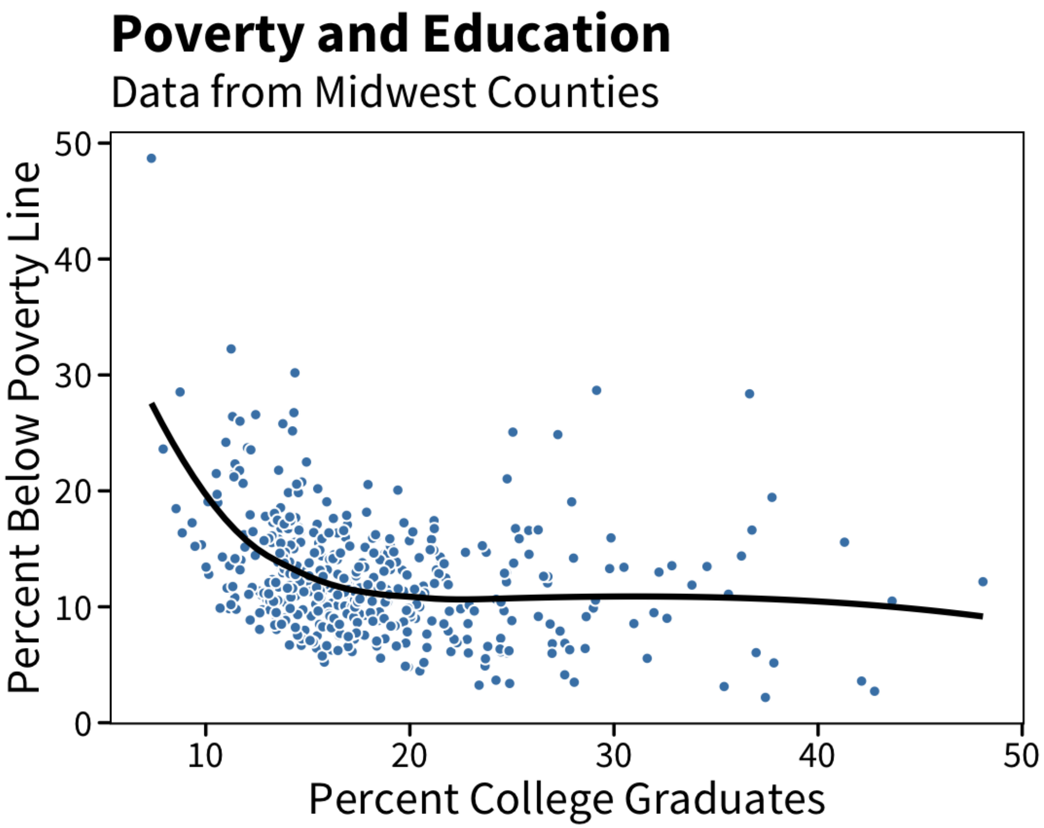
(Grab “Essay 2” Assignment Sheet)

Understanding Political Numbers

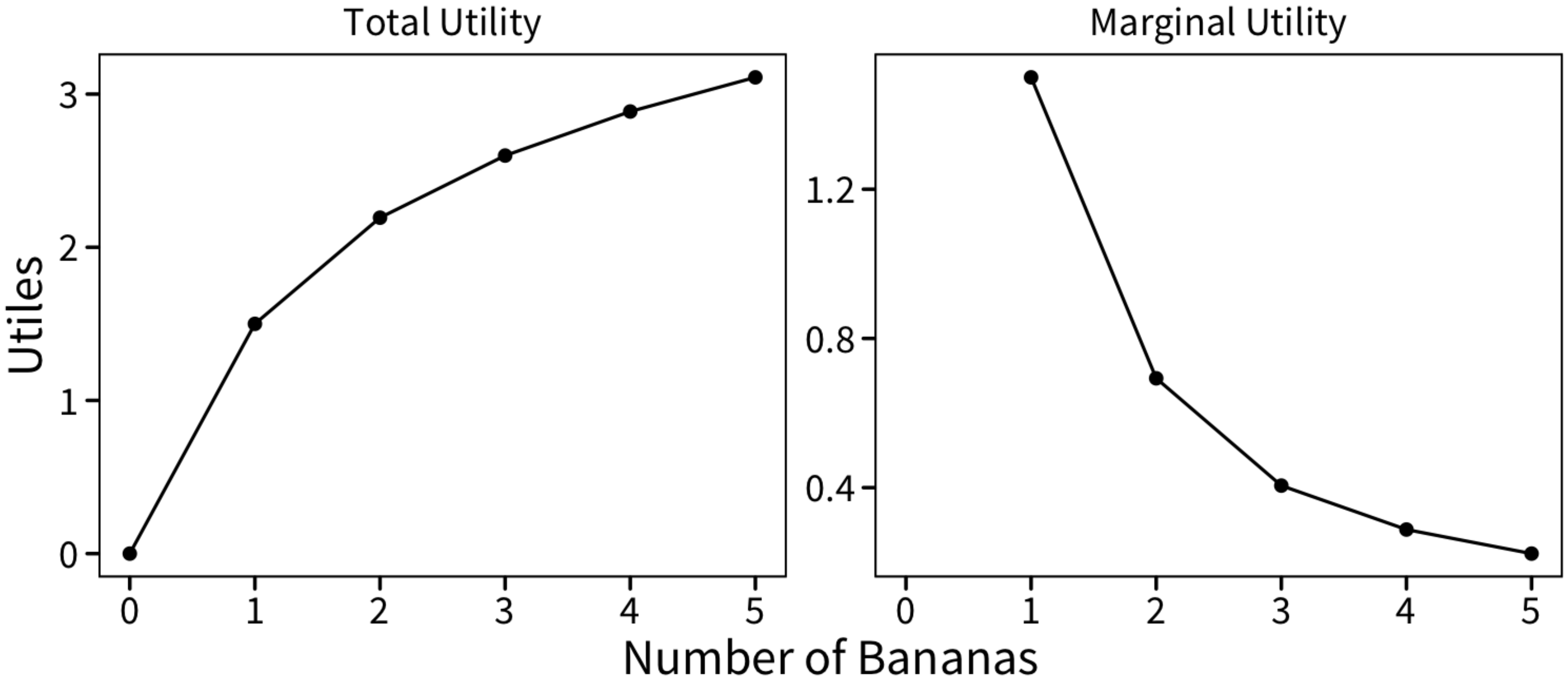
March 27, 2019

Where do you find nonlinear relationships?

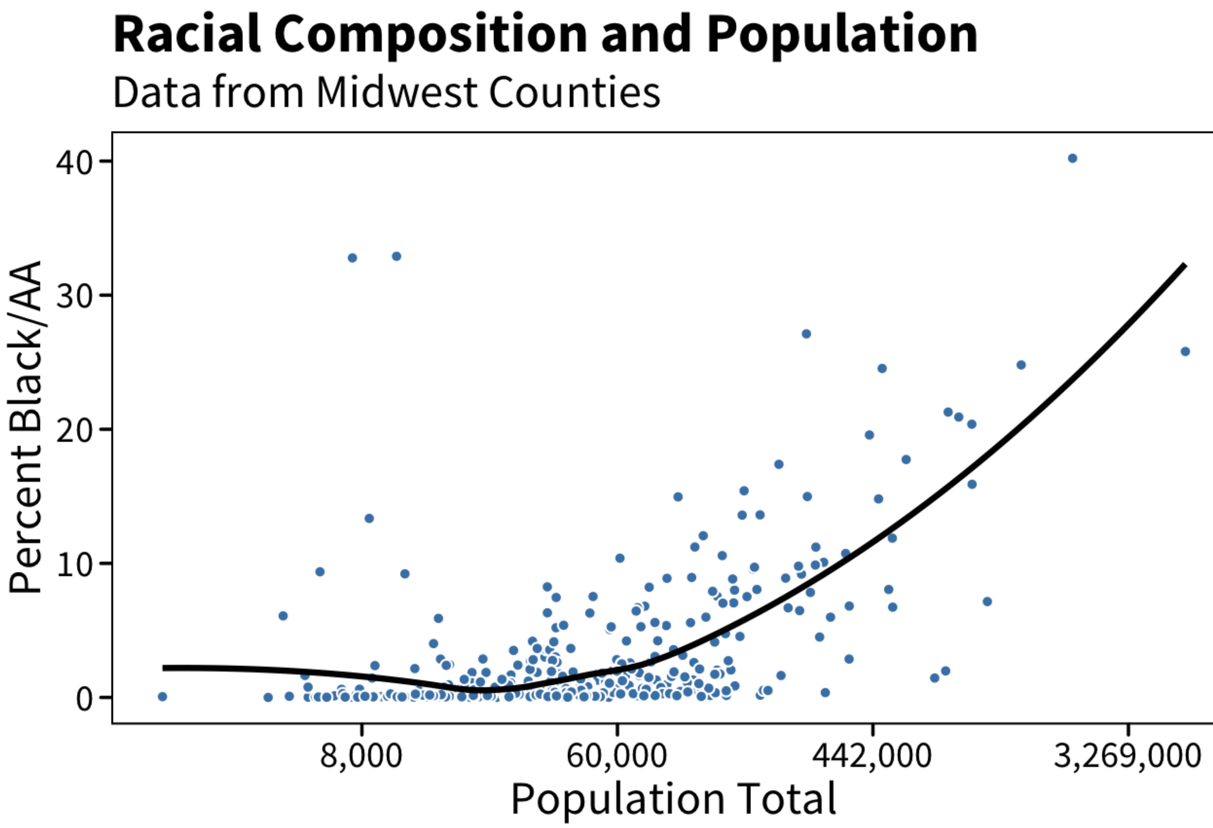
Diminishing Effects



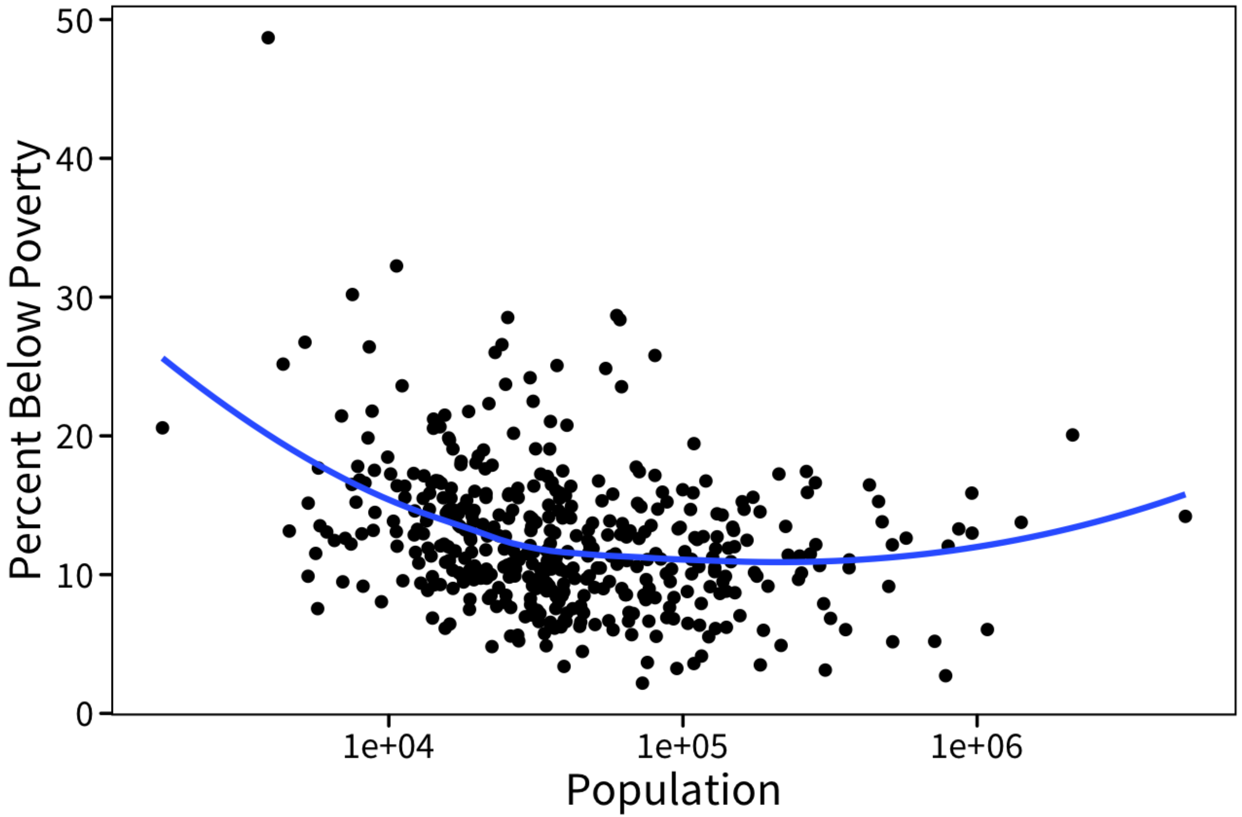
Value (Utility) of Eating Bananas



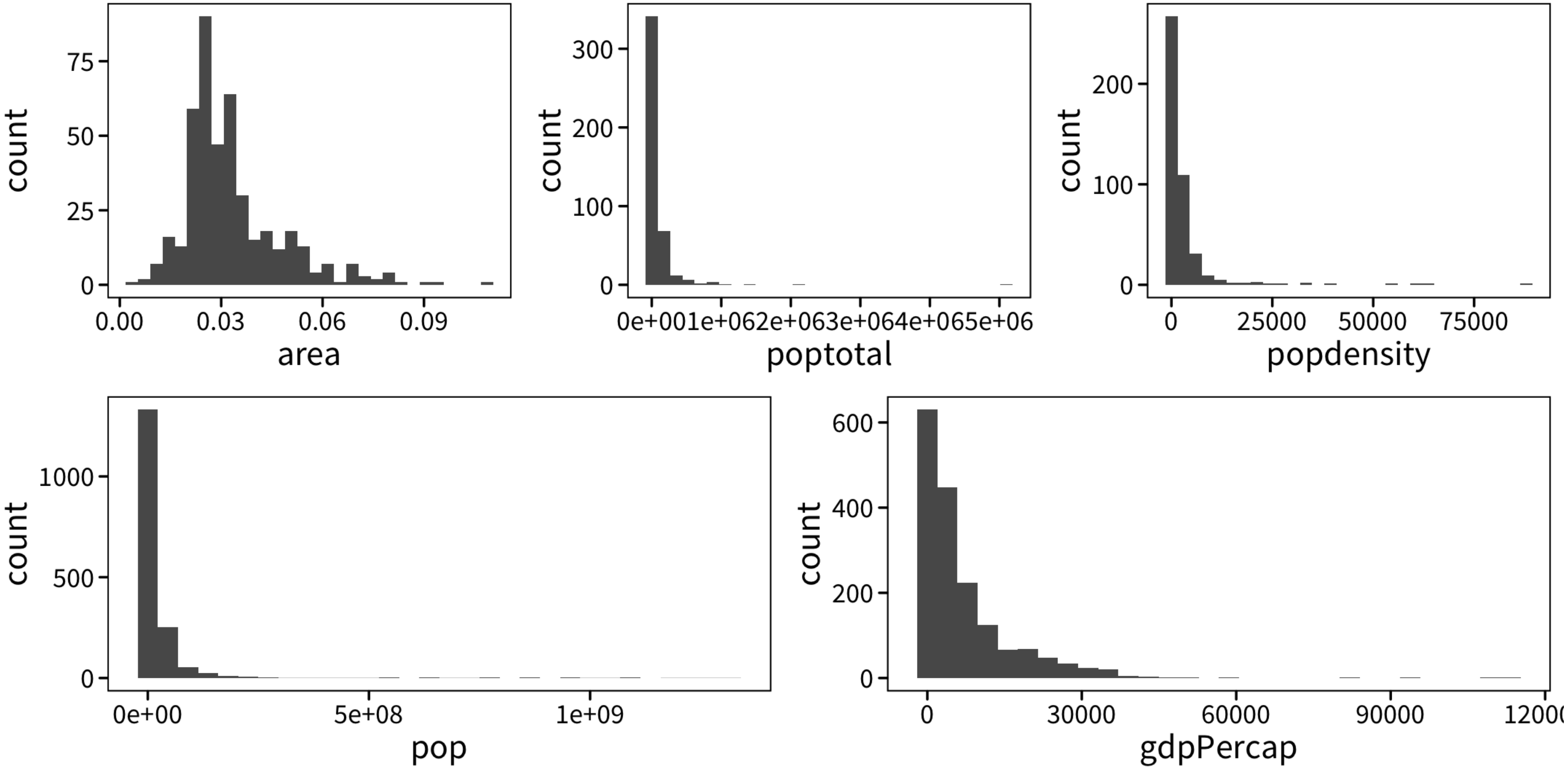
Natural Boundaries



"Multiplicative data"



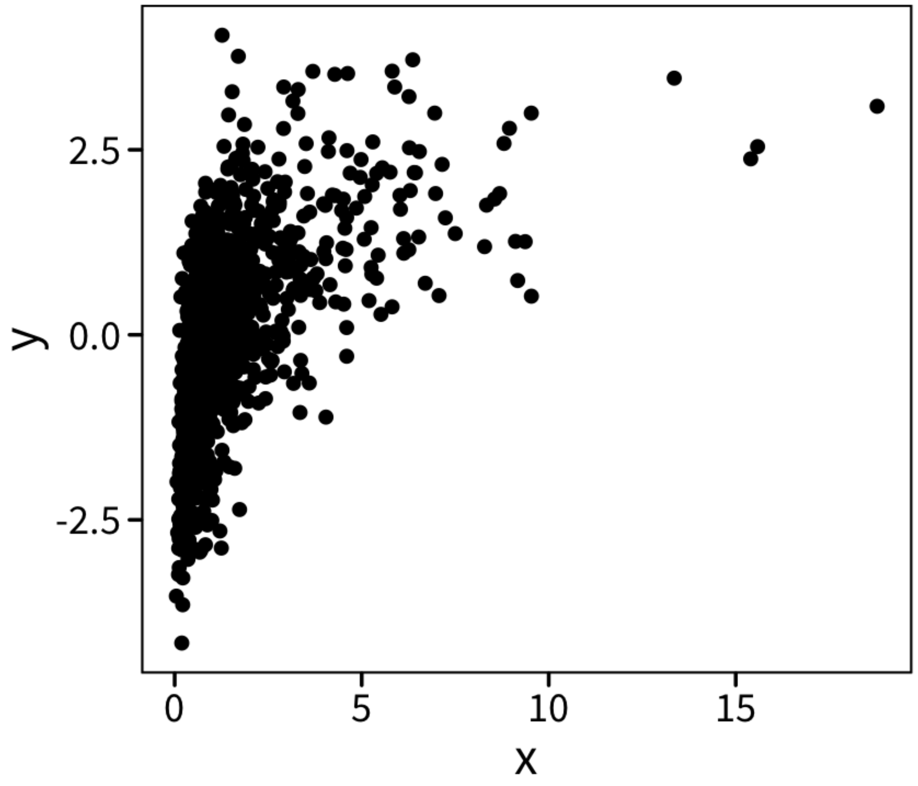
Long right tails (imperfect indicator)



Logarithmic transformations

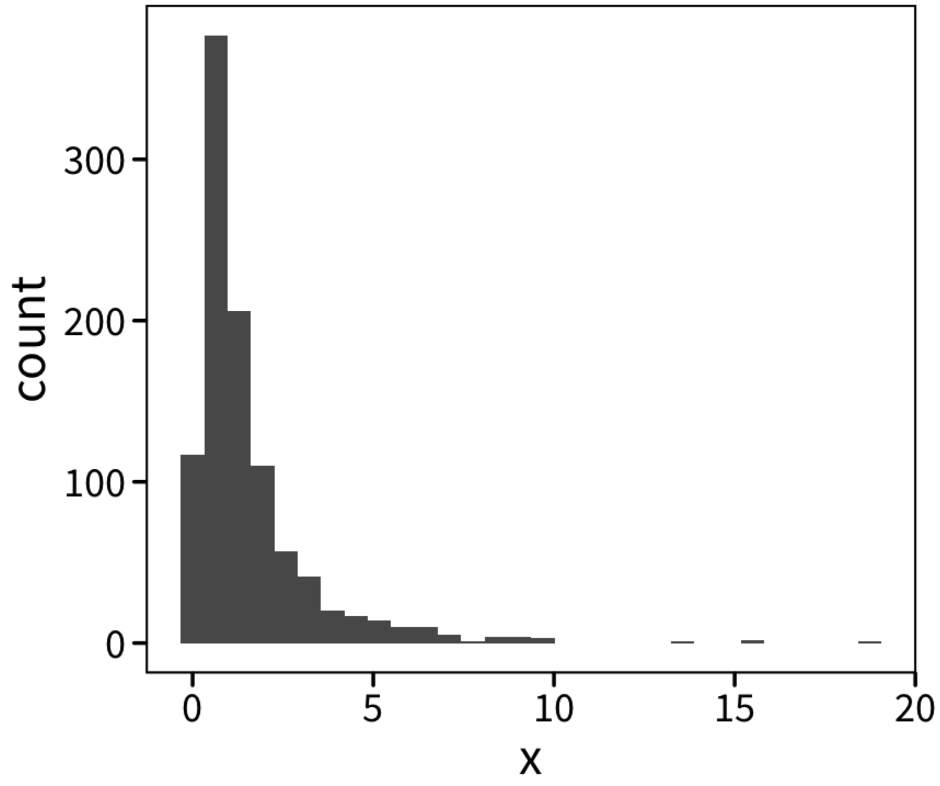
Logarithmic relationship

Sudden increase that tapers off



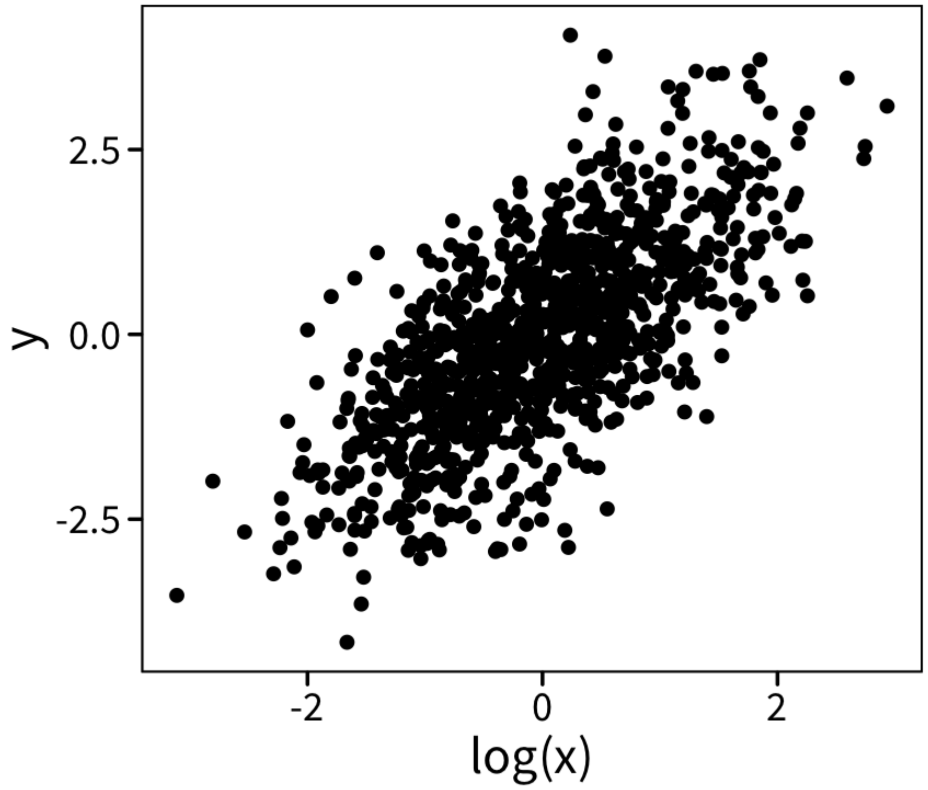
Skewed histogram

Long right tail



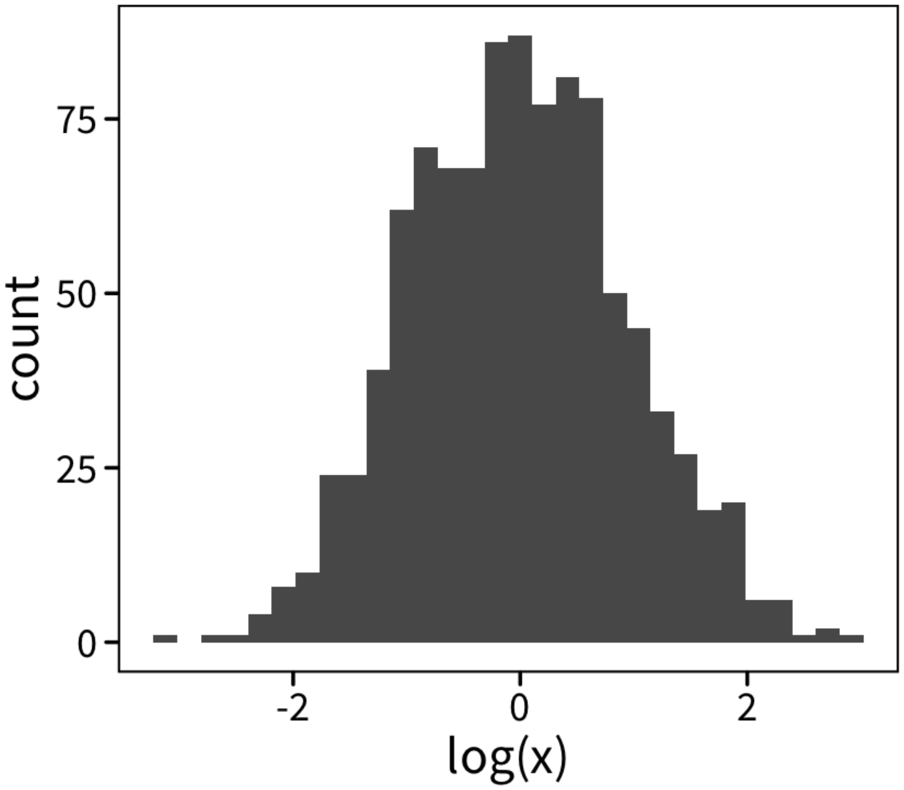
Logarithmic relationship

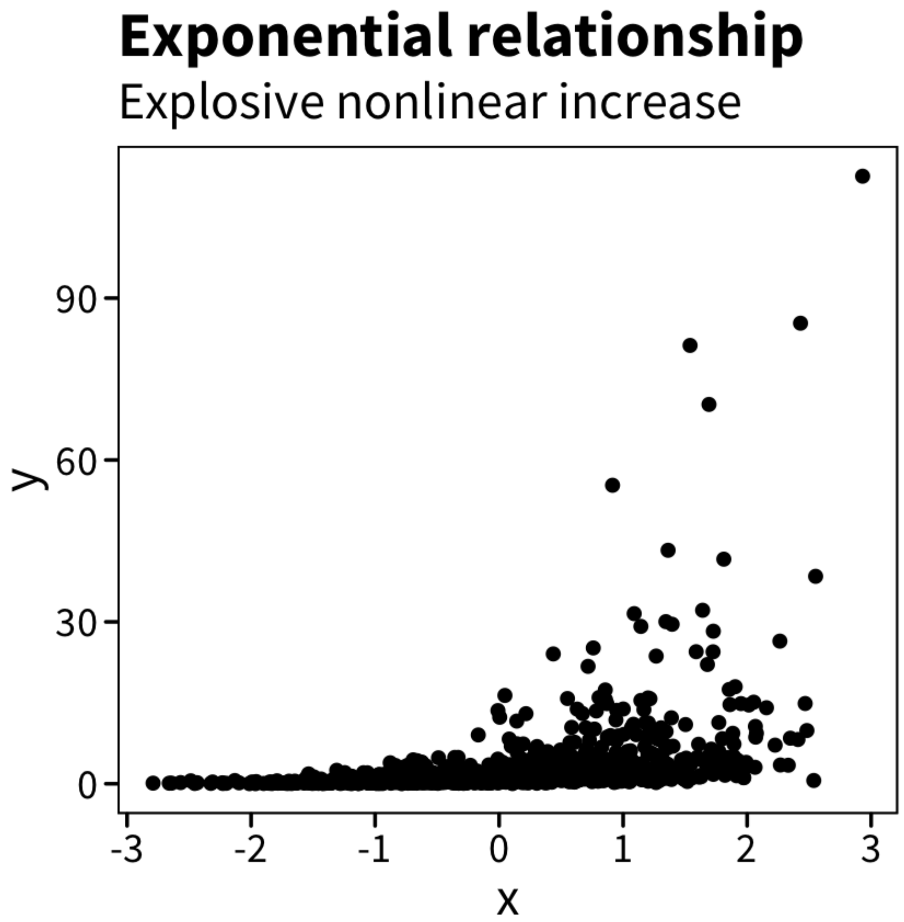
Appears LINEAR after logging

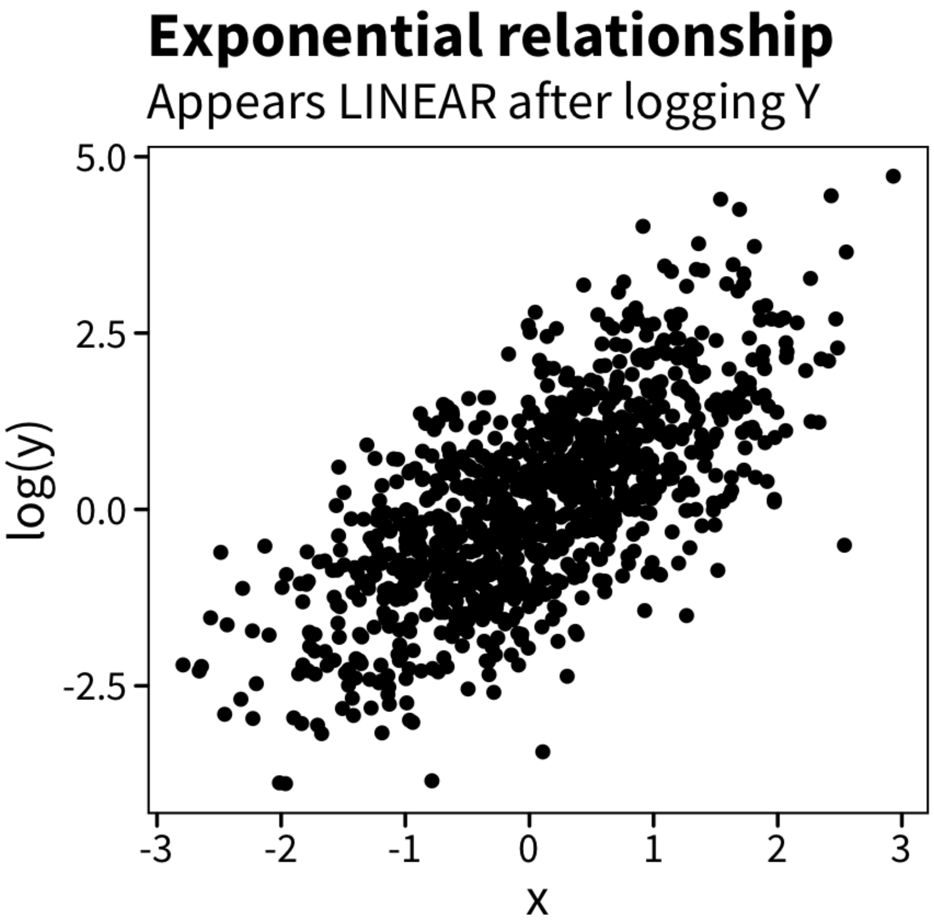


Skewed histogram

Appears NORMAL after logging

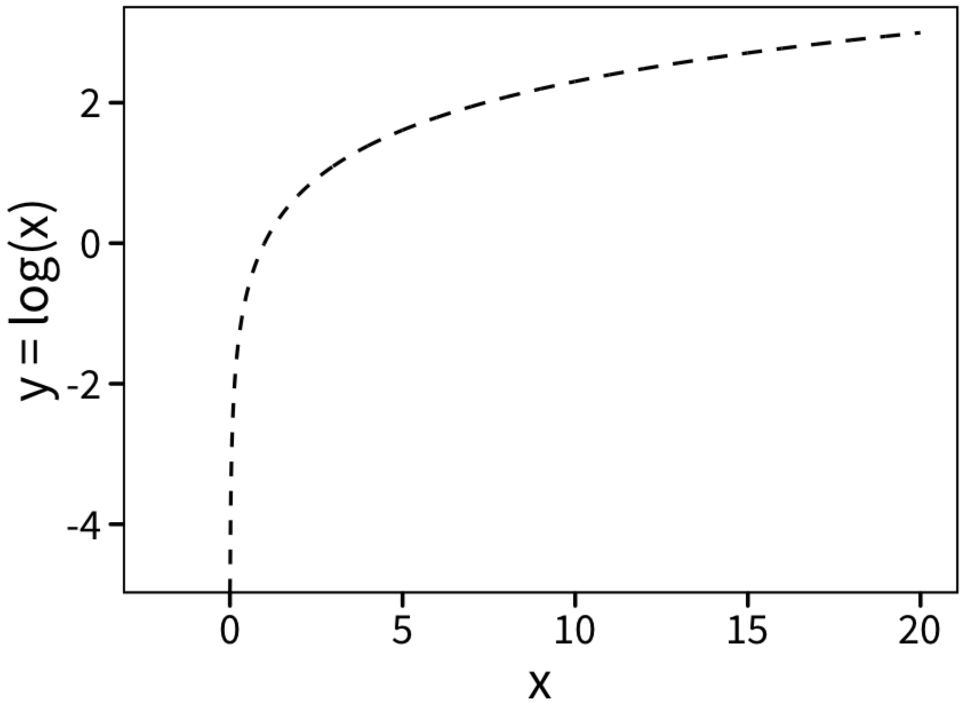



Exponential relationship
Explosive nonlinear increase
y
90
60
30
0
-3
-2
-1
0
1
2
3
x


Exponential relationship
Appears LINEAR after logging Y
log(y)
5.0
2.5
0.0
-2.5
-3
-2
-1
0
1
2
3
x

What is log? (Baby don't hurt me)

- Steep increase that diminishes
- *Never* fully "levels off"
- Defined over *positive* values only
- Commonly appears with count data, money, population



Logarithms "undo" exponentials

If $b^x = y$, then $\log_b(y) = x$

Logarithms "undo" exponentials

If $b^x = y$, then $\log_b(y) = x$

We usually only care about "base e" ($e = 2.7182818\ldots$)

Logarithms "undo" exponentials

If $b^x = y$, then $\log_b(y) = x$

We usually only care about "base e" ($e = 2.7182818\ldots$)

$$e^x = y$$

$$\ln(y) = x$$
$$\log(y) = x$$

Logarithms "undo" exponentials

If $b^x = y$, then $\log_b(y) = x$

We usually only care about "base e" ($e = 2.7182818\ldots$)

$$e^x = y$$

$$\ln(y) = x$$
$$\log(y) = x$$

Never worry about solving by hand

```
# natural log (base e) of 8
log(8)
```

```
## [1] 2.079442
```

```
# exponentials e^(2.079...)
exp(2.079442)
```

```
## [1] 8.000004
```

Logs and exponentials are *inverse functions*

Inverse functions: if $y = f(x)$, then $f^{-1}(y) = x$

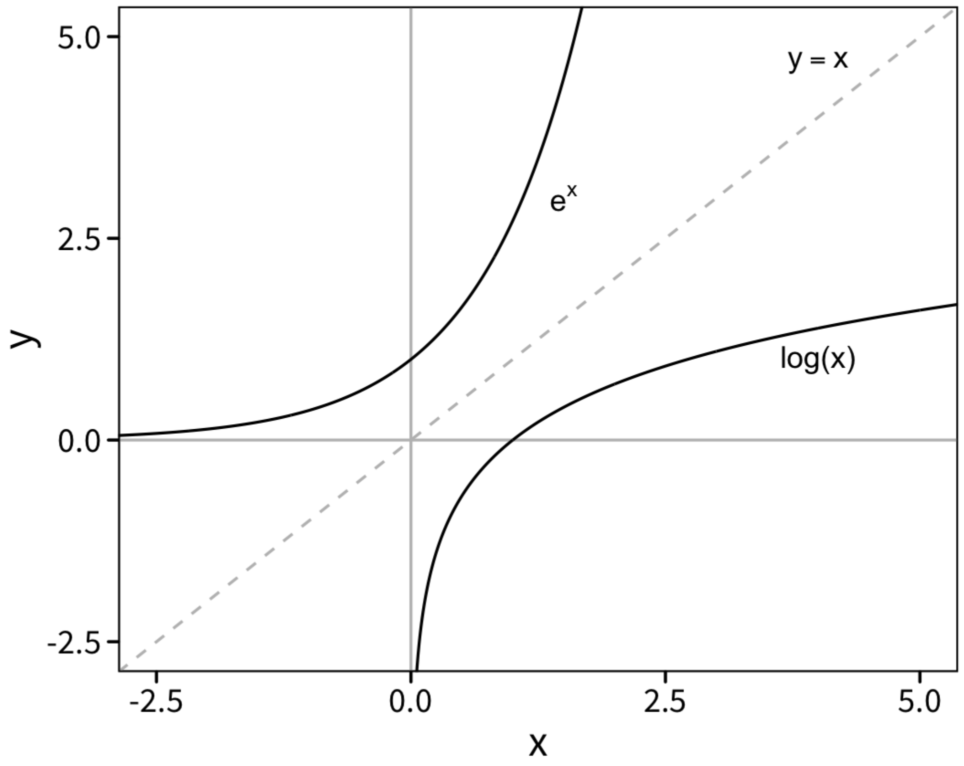
$$f^{-1}(f(x)) = x$$

$$\log(e^x) = x$$

$$e^{\log(x)} = x$$

If you need to log a variable: `log(var)`

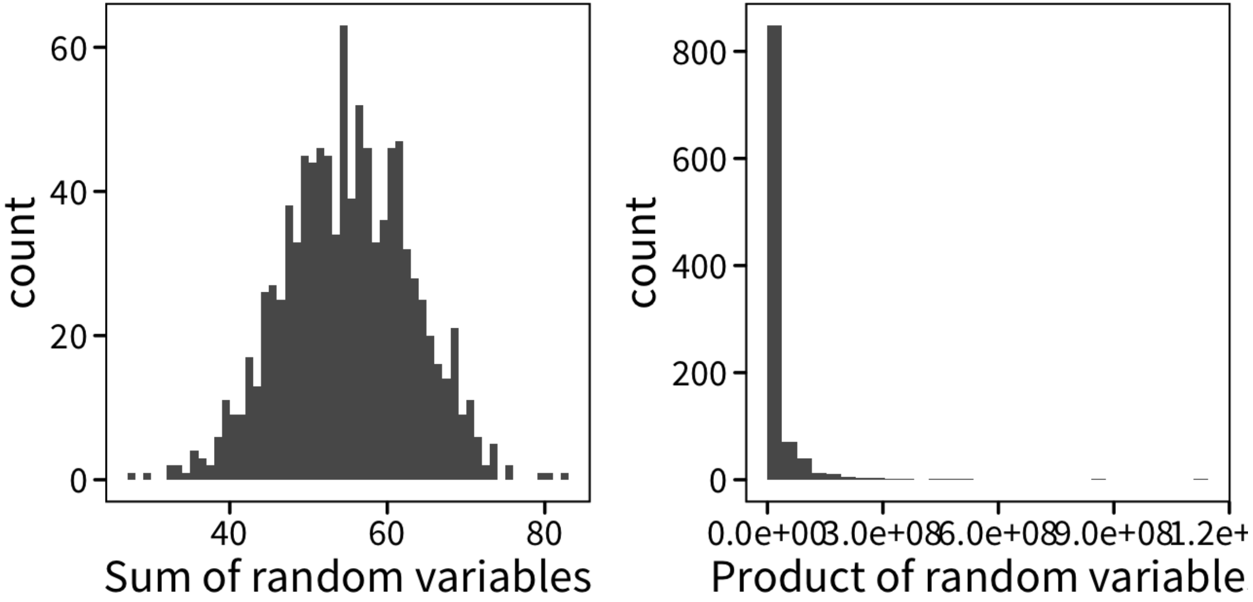
If you need to "unlog" a variable: `exp(var)`



Why we log for "multiplicative" data

Sum of random fluctuations -> Normal

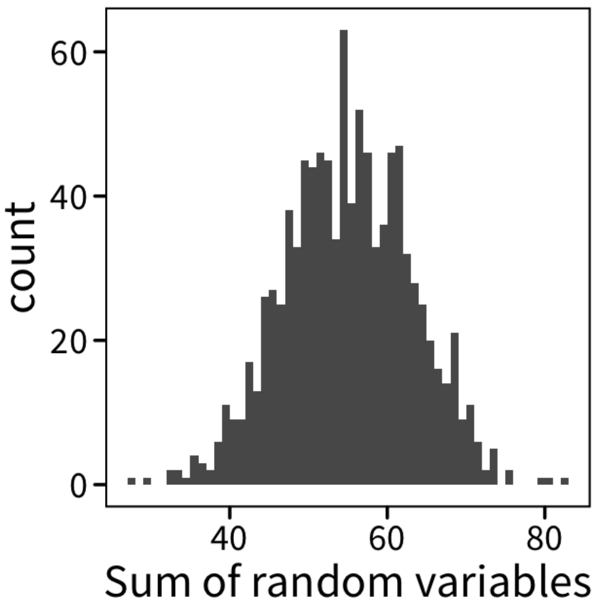
Product of random fluctuations -> "Log Normal"

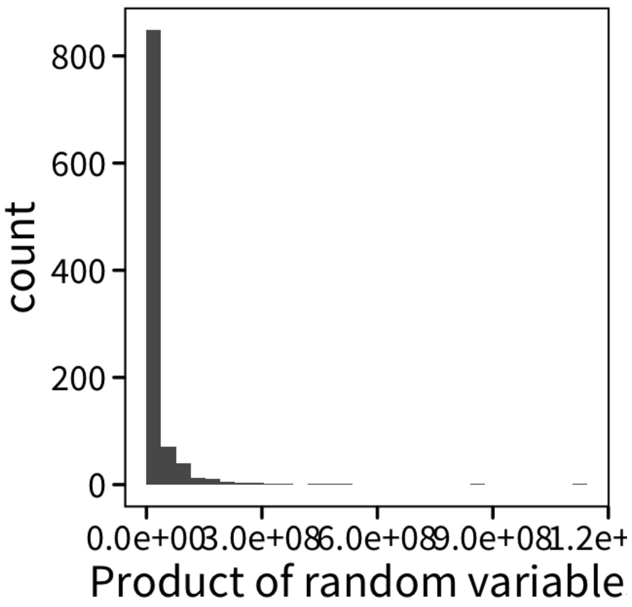


Why we log for "multiplicative" data

Sum of random fluctuations -> Normal

Product of random fluctuations -> "Log Normal"

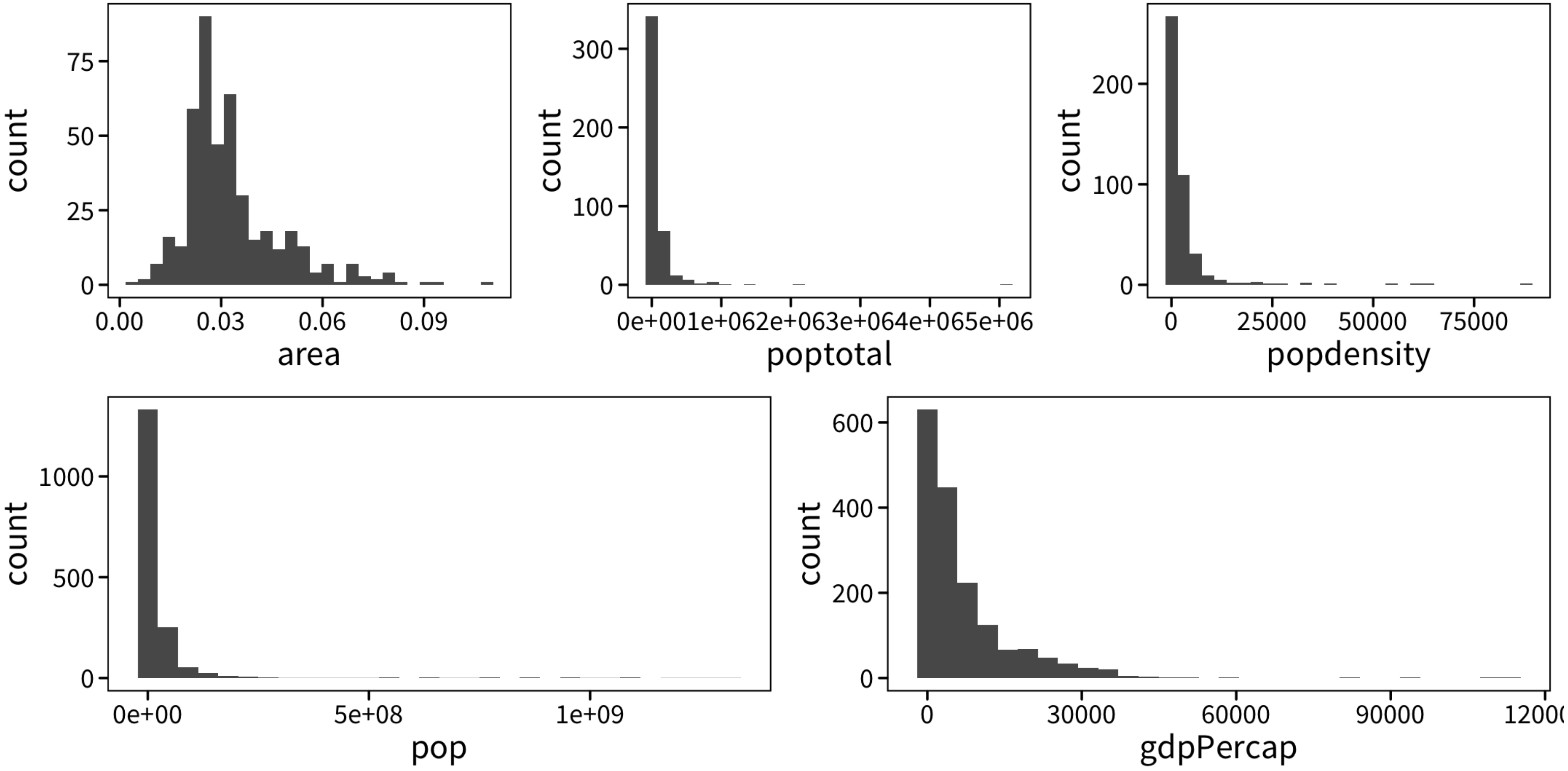




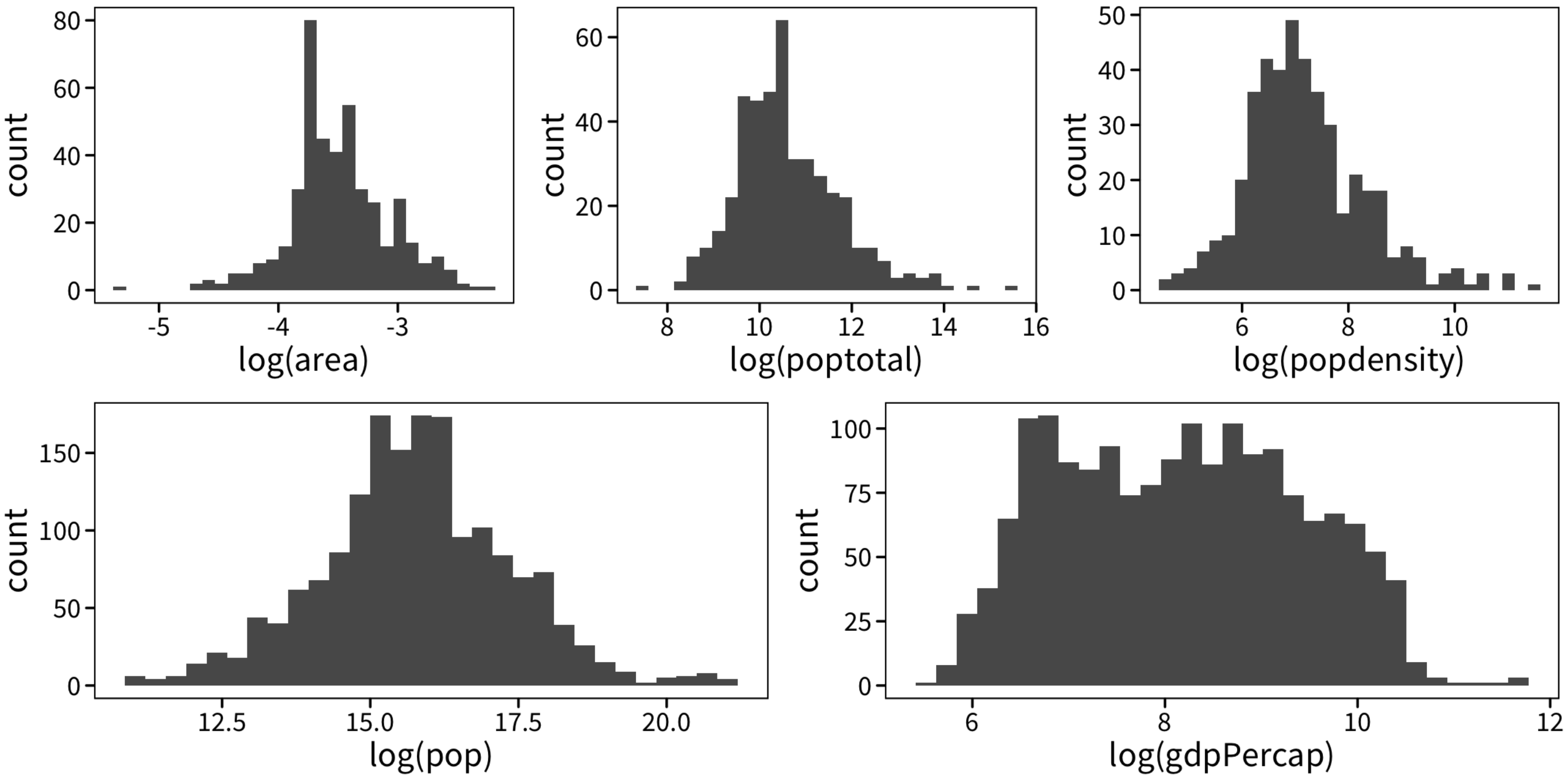
Logs turn *multiplicative* operations into *additive* (linear) operations

$$\log(a \times b) = \log(a) + \log(b)$$

Remember those long tails?

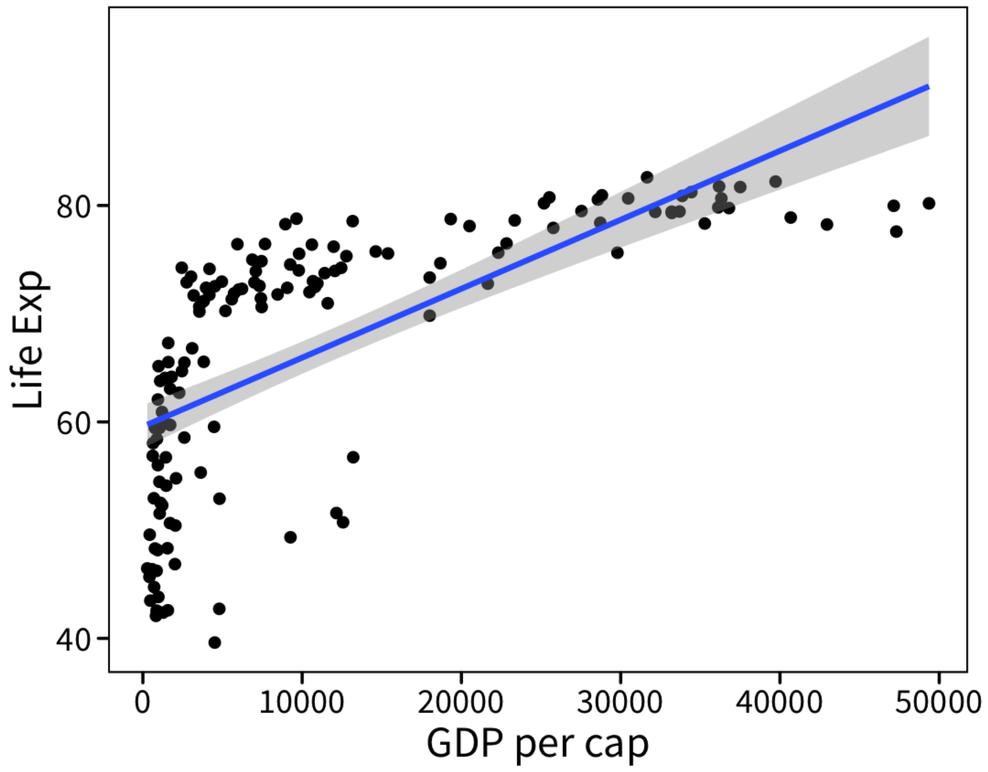


Take the log, long tails look normal



Logs in practice

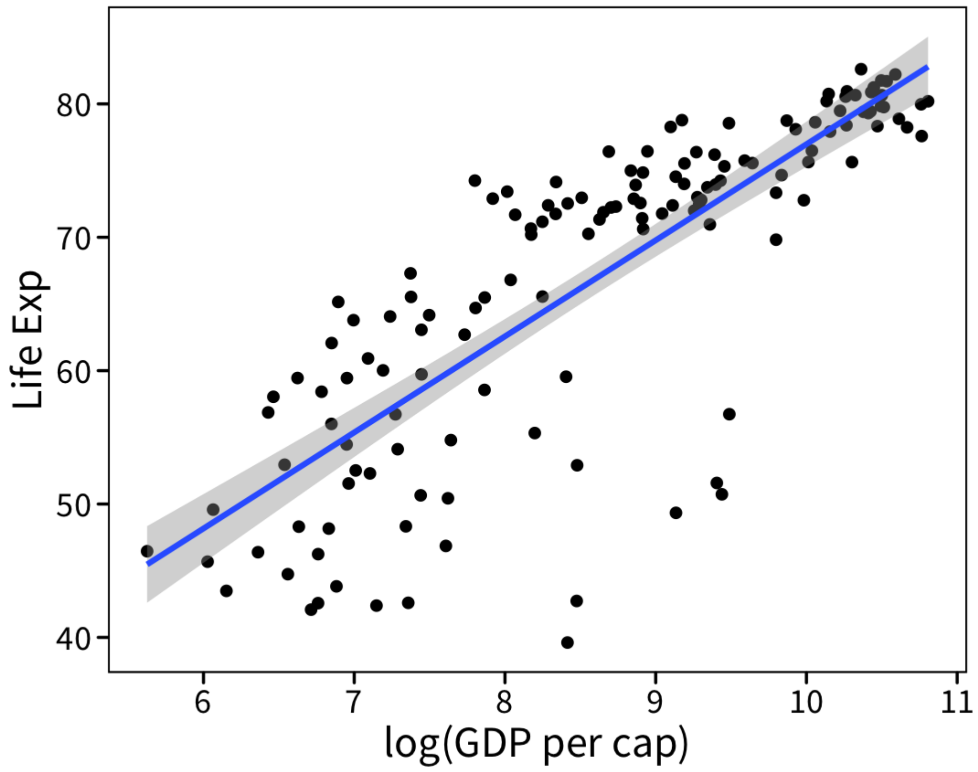
```r
library("gapminder")

# most recent gapminder year
gap_07 <- gapminder %>%
  filter(year == max(year))

# linear relationship looks bad
ggplot(gap_07, aes(x = gdpPercap, y = lifeExp)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(x = "GDP per cap", y = "Life Exp")
```



Logs in practice

Plot $y = f(log(x))$

```
# log(x) is... better
ggplot(gap_07,
       aes(x = log(gdpPercap), y = lifeExp)) +
  geom_point() +
  geom_smooth(method = "lm") +
  labs(x = "log(GDP per cap)", y = "Life Exp")
```



Logs in practice

Estimate the model

```
# new variable is log(x)
gap_07 <- gap_07 %>%
  mutate(log_gdp = log(gdpPercap))

log_model <- lm(lifeExp ~ log_gdp, data = gap_07)

tidy(log_model)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)     4.95     3.86       1.28 2.02e- 1
## 2 log_gdp         7.20     0.442     16.3  4.12e-34
```

Logs in practice

Estimate the model

```
# new variable is log(x)
gap_07 <- gap_07 %>%
  mutate(log_gdp = log(gdpPercap))

log_model <- lm(lifeExp ~ log_gdp, data = gap_07)

tidy(log_model)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)     4.95     3.86       1.28 2.02e- 1
## 2 log_gdp         7.20     0.442     16.3  4.12e-34
```

$$\widehat{\text{life}} = 4.95 + 7.2\log(\text{gdp.pc})$$

As *log GDP per capita* increases by one unit...

Logs in practice

Interpret graphically

```
# get predicted values for log(x)
# calculate unlogged x
# create upper & lower conf interval bounds
log_preds <- augment(log_model) %>%
  mutate(gdp_pc = exp(log_gdp),
         MOE = 1.96 * .se.fit,
         conf.low = .fitted - MOE,
         conf.high = .fitted + MOE) %>%
  print()
```

```
## # A tibble: 142 x 13
##    lifeExp log_gdp .fitted .se.fit  .resid    .hat .sigma .cooksd
##      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>  <dbl>   <dbl>
##  1    43.8    6.88    54.5   0.972 -10.7   0.0186    7.09 2.18e-2
##  2    76.4    8.69    67.5   0.599   8.89  0.00706   7.11 5.58e-3
##  3    72.3    8.74    67.9   0.600   4.43  0.00710   7.14 1.39e-3
##  4    42.7    8.48    66.0   0.601 -23.3   0.00712   6.87 3.85e-2
##  5    75.3    9.46    73.1   0.704   2.26  0.00976   7.15 5.03e-4
##  6    81.2   10.4     80.2   1.01    1.04  0.0200    7.15 2.21e-4
##  7    79.8   10.5     80.5   1.02   -0.712 0.0207    7.15 1.08e-4
##  8    75.6   10.3     79.2   0.956  -3.52  0.0180    7.14 2.28e-3
```

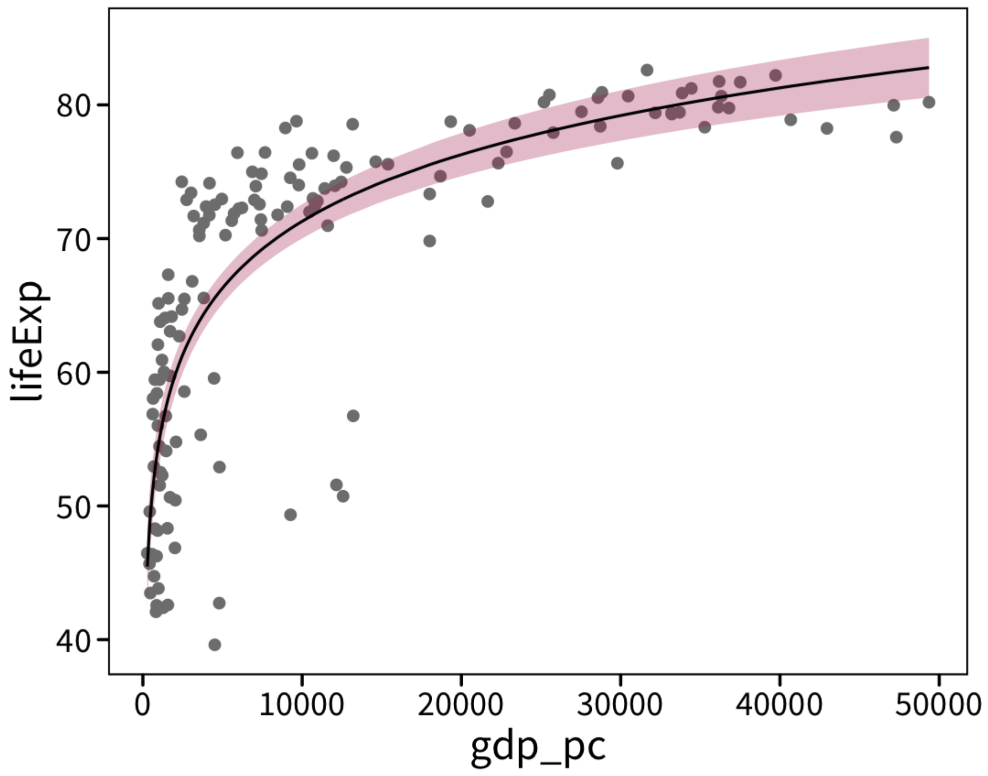
Logs in practice

Interpret graphically

```r
# get predicted values for log(x)
# calculate unlogged x
# create upper & lower conf interval bounds
log_preds <- augment(log_model) %>%
  mutate(gdp_pc = exp(log_gdp),
         MOE = 1.96 * .se.fit,
         conf.low = .fitted - MOE,
         conf.high = .fitted + MOE) %>%
  print()
```

```
## # A tibble: 142 x 13
##    lifeExp log_gdp .fitted .se.fit  .resid    .hat .s
##      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl> <d
## 1     43.8    6.88    54.5   0.972 -10.7   0.0186   7
## 2     76.4    8.69    67.5   0.599   8.89  0.00706  7
## 3     72.3    8.74    67.9   0.600   4.43  0.00710  7
## 4     42.7    8.48    66.0   0.601 -23.3   0.00712  6
## 5     75.3    9.46    73.1   0.704   2.26  0.00976  7
## 6     81.2   10.4     80.2   1.01    1.04  0.0200   7
## 7     79.8   10.5     80.5   1.02   -0.712 0.0207   7
## 8     75.6   10.3     79.2   0.956  -3.52  0.0180    7.14 2.28e-3
```

```r
# plot y over unlogged x, add yhat line
ggplot(log_preds, aes(x = gdp_pc, y = lifeExp)) +
  geom_point(color = "gray50") +
  geom_ribbon(
    aes(ymin = conf.low, ymax = conf.high),
    fill = "maroon", alpha = .3
  ) +
  geom_line(aes(y = .fitted))
```



Other log rules

$$\log(a \times b) = \log(a) + \log(b)$$

$$\log\left(\frac{a}{b}\right) = \log(a) - \log(b)$$

$$\log\left(a^b\right) = b \times \log(a)$$

$$\log(1) = 0$$

$$\log(0) = ?$$

Remember...

Diminishing effects, natural boundaries

Data generated from "multiplicative" process (populations, dollars)

Don't solve logs yourself; only need to `log(x)`

Undoing logs: `exp(log_x)`