Living the sample life

SAMPLING IN R

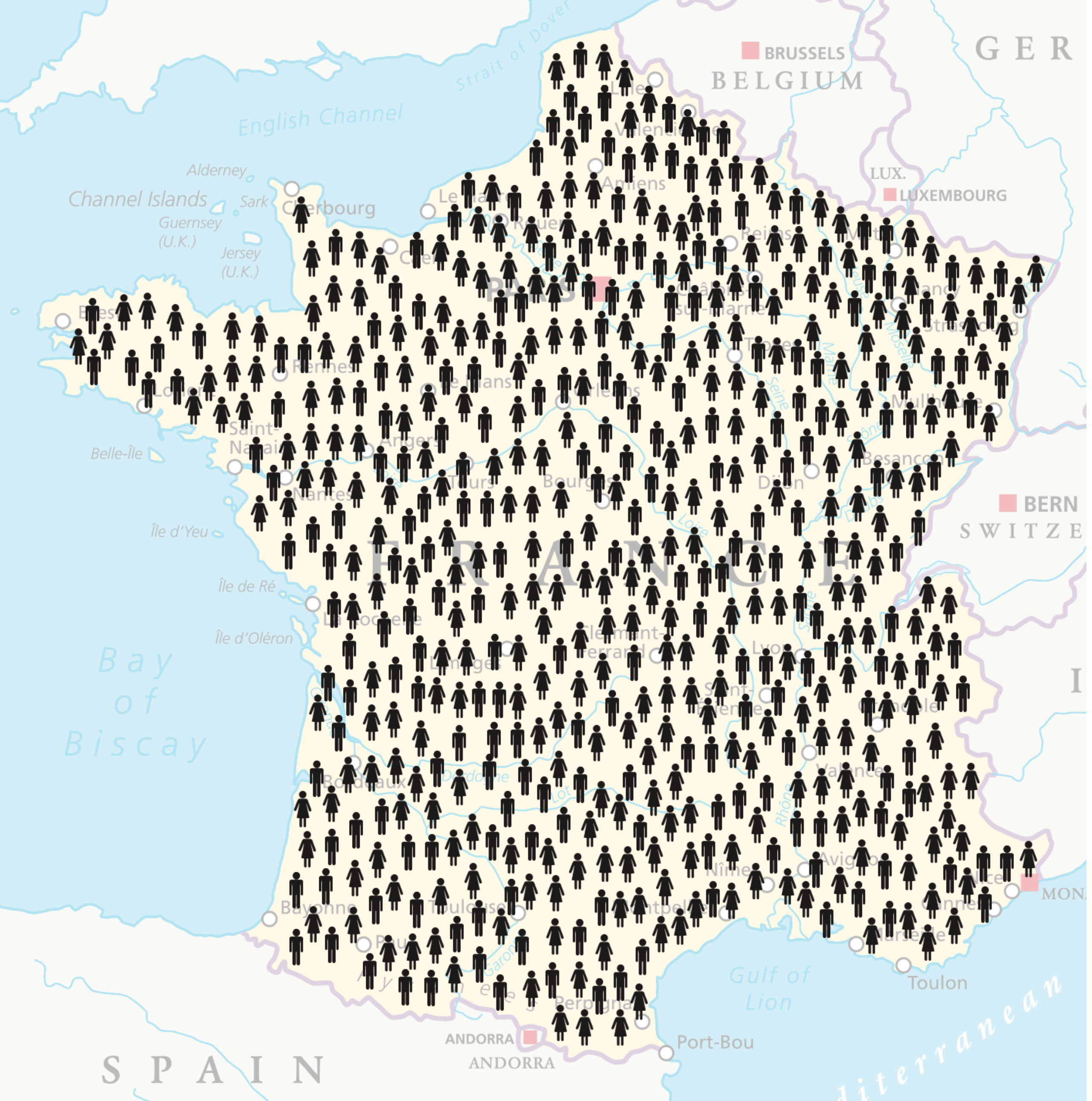
Richie Cotton

Data Evangelist at DataCamp

Estimating the population of France

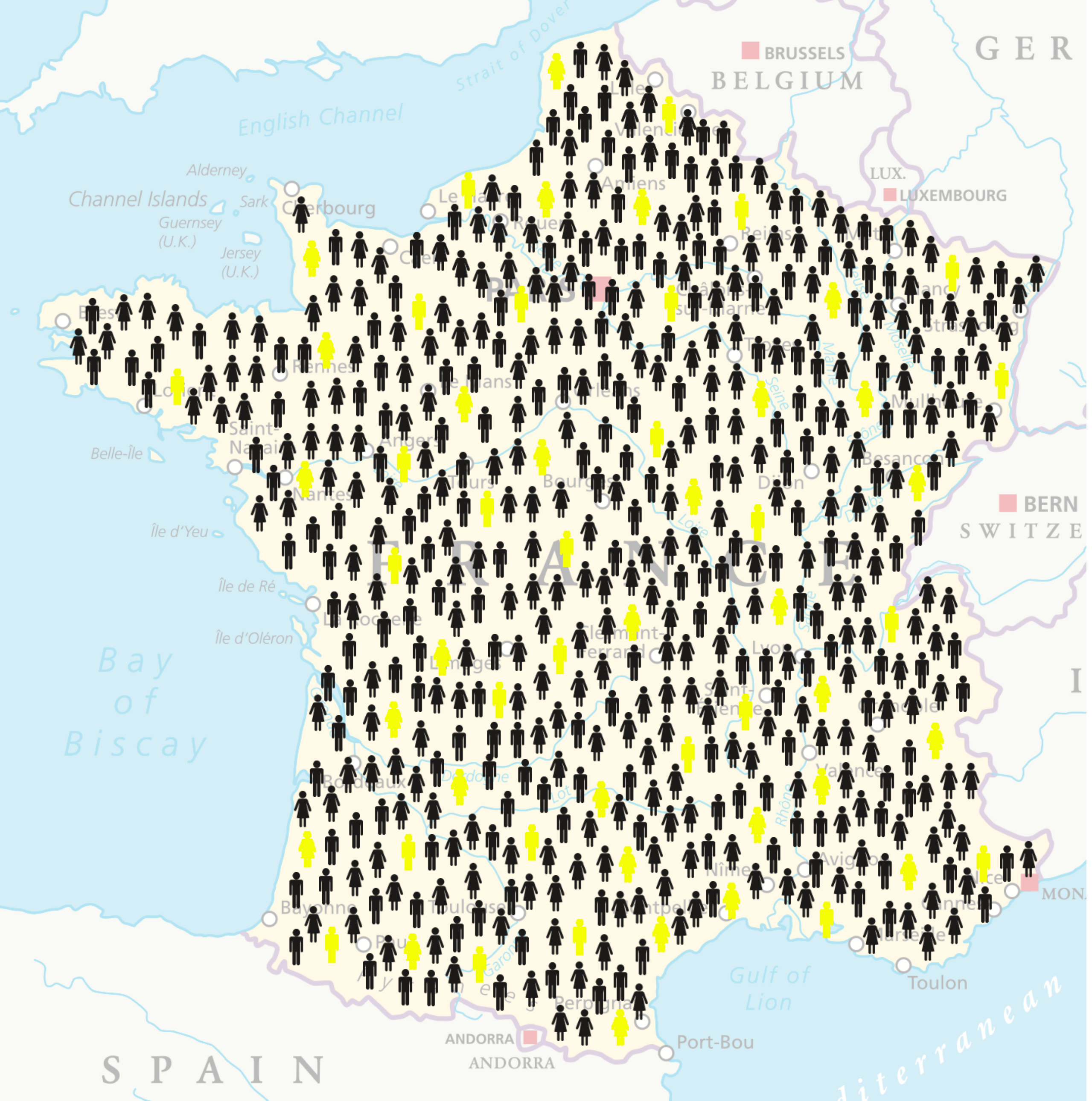
A census asks every household how many people live there.

There are lots of people in France



Censuses are really expensive!

Sampling households



It's cheaper to ask a small number of households and use statistics to make estimates about the whole population.

Working with a subset of the whole population is called *sampling*.

Population vs. sample

The *population* is the complete dataset.

- It doesn't have to refer to people.
- You typically don't know what the whole population is.

The *sample* is the subset of data you calculate on.

Coffee rating dataset

total_cup_points	variety	country_of_origin	aroma	flavor	aftertaste	body	balance
90.58	NA	Ethiopia	8.67	8.83	8.67	8.50	8.42
89.92	Other	Ethiopia	8.75	8.67	8.50	8.42	8.42
...	...	...	...	...	...	...	...
73.75	NA	Vietnam	6.75	6.67	6.5	6.92	6.83

- Each row represents 1 coffee.
- 1138 rows.
- We'll treat this as the population.

Points vs. flavor: population

```
pts_vs_flavor_pop <- coffee_ratings %>%
  select(total_cup_points, flavor)
```

```
dim(pts_vs_flavor_pop)
```

```
1338    2
```

```
      total_cup_points  flavor
1                90.58    8.83
2                89.92    8.67
3                89.75    8.50
4                89.00    8.58
...                ...     ...
1335             78.08    7.67
1336             77.17    7.33
1337             75.08    6.83
1338             73.75    6.67
```

Points vs. flavor: 10 row sample

```
pts_vs_flavor_samp <- coffee_ratings %>%
  select(total_cup_points, flavor) %>%
  slice_sample(n = 10)
```

```
dim(pts_vs_flavor_samp)
```

```
10  2
```

```
   total_cup_points flavor
1             82.25   7.58
2             83.50   7.67
3             80.50   7.17
4             79.33   7.17
5             83.83   7.58
6             84.17   7.75
7             83.67   8.17
8             81.92   7.50
9             82.67   7.58
10            83.42   7.67
```

Base-R sampling

Use `slice_sample()` for data frames, and `sample()` for vectors.

```r
cup_points_samp <- sample(coffee_ratings$total_cup_points, size = 10)
```

```
88.25 83.83 83.17 82.67 84.67 83.42 73.67 86.00 81.58 80.92
```

Population parameters & point estimates

A *population parameter* is a calculation made on the population dataset.

```
mean(pts_vs_flavor_pop$total_cup_points)
```

```
82.15
```

A *point estimate* or *sample statistic* is a calculation made on the sample dataset.

```
mean(cup_points_samp)
```

```
82.82
```

Point estimates with dplyr

```
pts_vs_flavor_pop %>%
  summarize(mean_flavor = mean(flavor))
```

```
  mean_flavor
1       7.526
```

```
pts_vs_flavor_samp %>%
  summarize(mean_flavor = mean(flavor))
```

```
  mean_flavor
1       7.716
```

Let's practice!

SAMPLING IN R

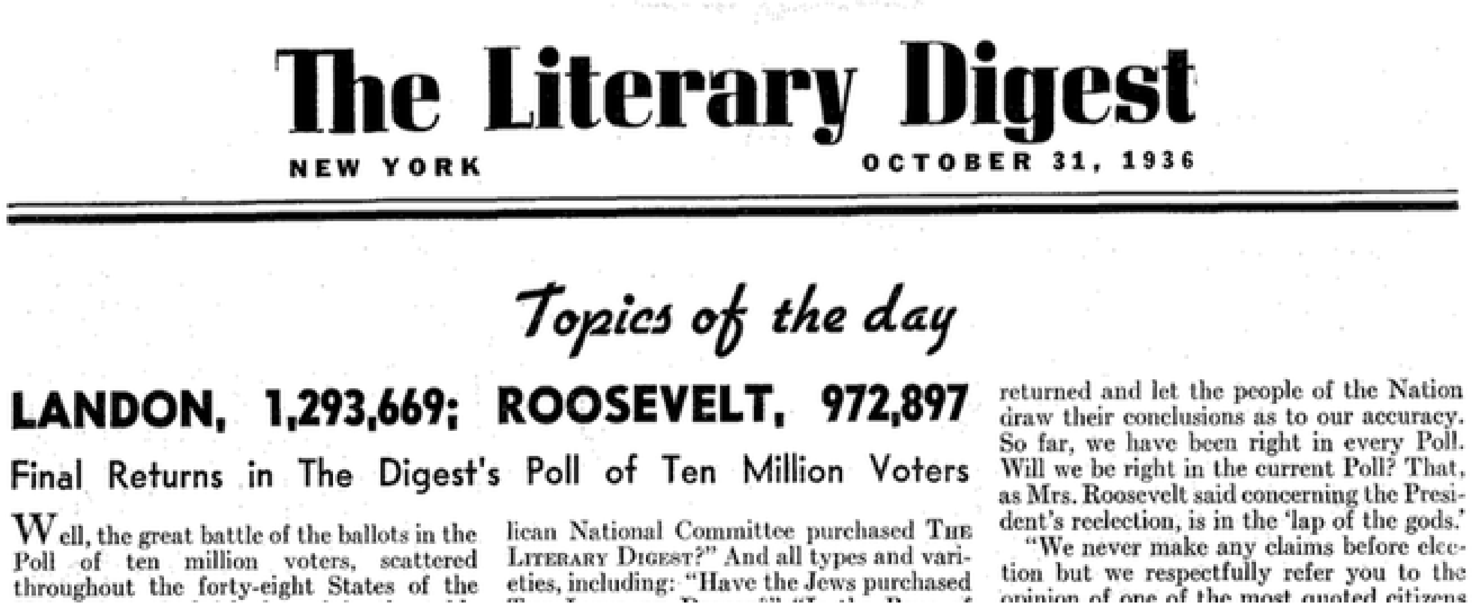
A little too convenient

SAMPLING IN R

Richie Cotton

Data Evangelist at DataCamp

The Literary Digest election prediction



The Literary Digest

NEW YORK OCTOBER 31, 1936

Topics of the day

LANDON, 1,293,669; ROOSEVELT, 972,897

Final Returns in The Digest's Poll of Ten Million Voters

Well, the great battle of the ballots in the Poll of ten million voters, scattered throughout the forty-eight States of the

lican National Committee purchased The Literary Digest?" And all types and varieties, including: "Have the Jews purchased

returned and let the people of the Nation draw their conclusions as to our accuracy. So far, we have been right in every Poll. Will we be right in the current Poll? That, as Mrs. Roosevelt said concerning the President's reelection, is in the 'lap of the gods.'

"We never make any claims before election but we respectfully refer you to the

- Prediction: Landon gets 57%; Roosevelt gets 43%
- Actual results: Landon got 38%; Roosevelt got 62%
- Sample not representative of population, causing *sample bias*.
- Collecting data by the easiest method is called *convenience sampling*.

Finding the mean age of French people

- Survey 10 people at Disneyland Paris.
- Their mean age is 24.6 years.
- Will this be a good estimate for all of France?

[1] Image by Sean MacEntee

How accurate was the survey?

Year	Average French Age
1975	31.6
1985	33.6
1995	36.2
2005	38.9
2015	41.2

- 24.6 years is a poor estimate.
- People who visit Disneyland aren't representative of the whole population.

Convenience sampling coffee ratings

```
coffee_ratings %>%
  summarize(mean_cup_points = mean(total_cup_points))
```

```
  mean_cup_points
1           82.09
```

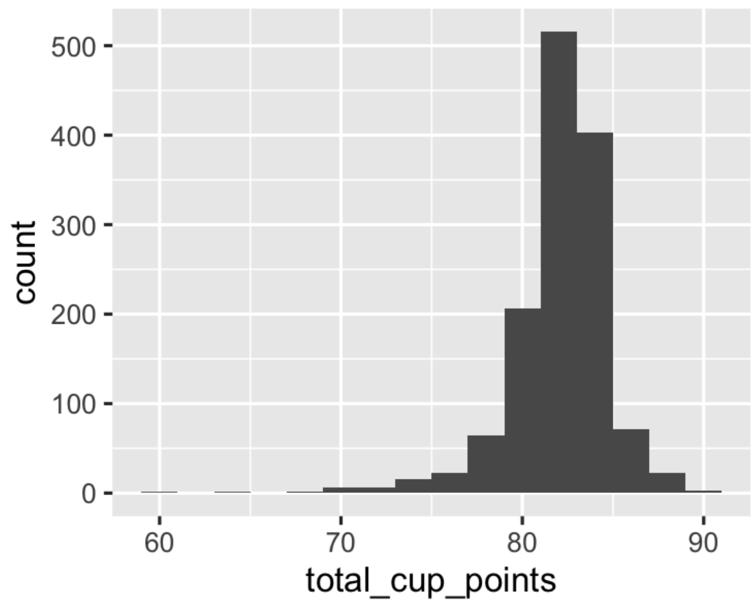
```
coffee_ratings_first10 <- coffee_ratings %>%
  slice_head(n = 10)
```

```
coffee_ratings_first10 %>%
  summarize(mean_cup_points = mean(total_cup_points))
```

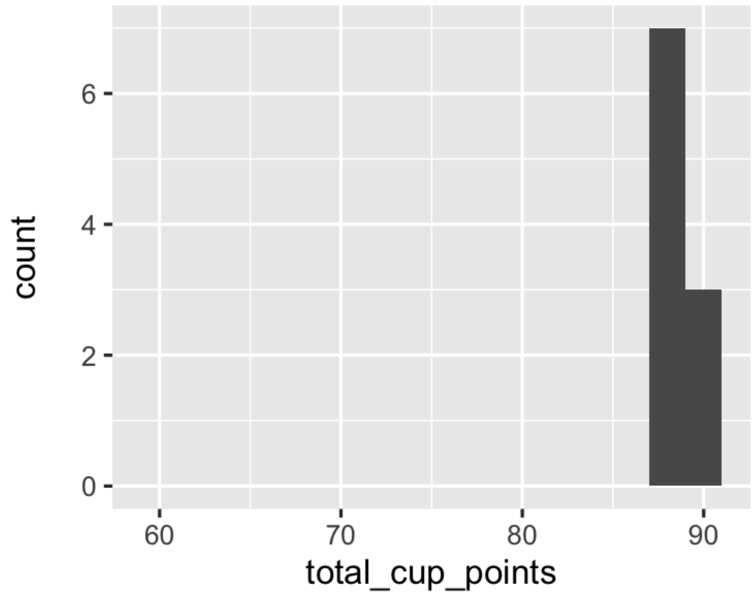
```
  mean_cup_points
1            89.1
```

Visualizing selection bias

```
coffee_ratings %>%
  ggplot(aes(x = total_cup_points)) +
  geom_histogram(binwidth = 2)
```

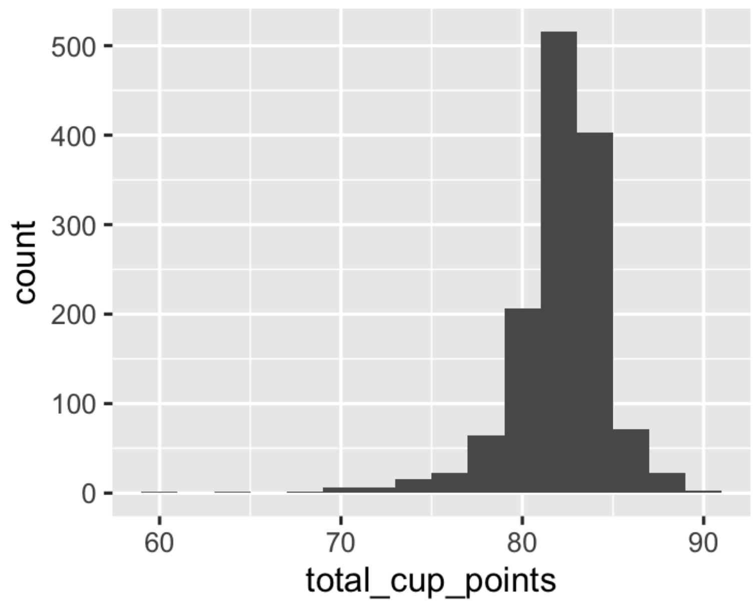


```
coffee_ratings_first10 %>%
  ggplot(aes(x = total_cup_points)) +
  geom_histogram(binwidth = 2) +
  xlim(59, 91)
```

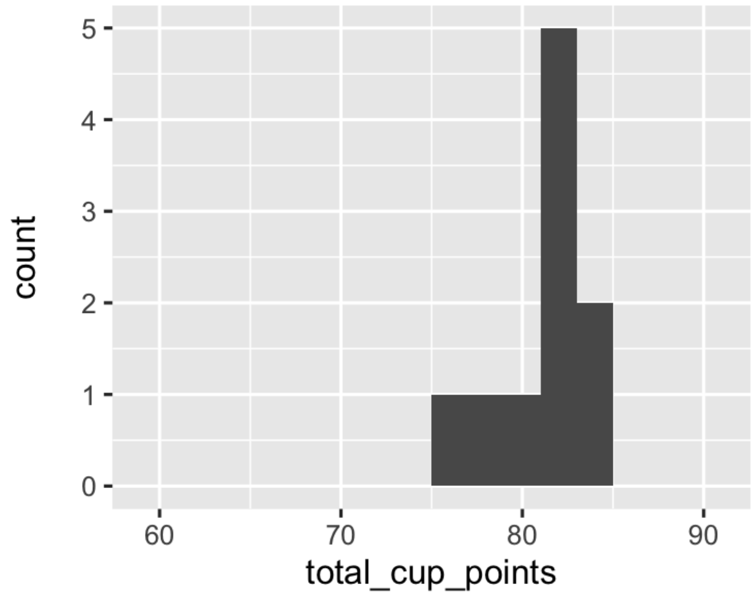


Visualizing selection bias 2

```
coffee_ratings %>%
  ggplot(aes(x = total_cup_points)) +
  geom_histogram(binwidth = 2)
```



```
coffee_ratings %>%
  slice_sample(n = 10) %>%
  ggplot(aes(x = total_cup_points)) +
  geom_histogram(binwidth = 2) +
  xlim(59, 91)
```



Let's practice!

SAMPLING IN R

How does Sue do sampling?

SAMPLING IN R

Richie Cotton

Data Evangelist at DataCamp

What does random mean?

> *{adjective}* made, done, happening, or chosen without method or conscious decision.

- Oxford Languages

True random numbers

- Generated from physical processes, like flipping coins.
- Hotbits uses radioactive decay.
- RANDOM.ORG uses atmospheric noise.
  - Available in R via the `random` package.
- True randomness is expensive.

[1] https://www.fourmilab.ch/hotbits [2] https://www.random.org

Pseudo-random number generation

- Next "random" number calculated from previous "random" number.
- The first "random" number calculated from a *seed*.
- If you start from the same seed value, all future random numbers will be the same.

```
seed <- 1
calc_next_random(seed)
```

```
3
```

```
calc_next_random(3)
```

```
2
```

```
calc_next_random(2)
```

```
6
```

Random number generating functions

function	distribution	function	distribution	function	distribution
rbeta	Beta	rgeom	Geometric	rsignrank	Wilcoxon signed rank
rbinom	Binomial	rhyper	Hypergeometric	rt	t
rcauchy	Cauchy	rlnorm	Lognormal	runif	Uniform
rchisq	Chi-squared	rlogis	Logistic	rweibull	Weibull
rexp	Exponential	rnbinom	Negative binomial	rwilcox	Wilcoxon rank sum
rf	F	rnorm	Normal		
rgamma	Gamma	rpois	Poisson		

Visualizing random numbers

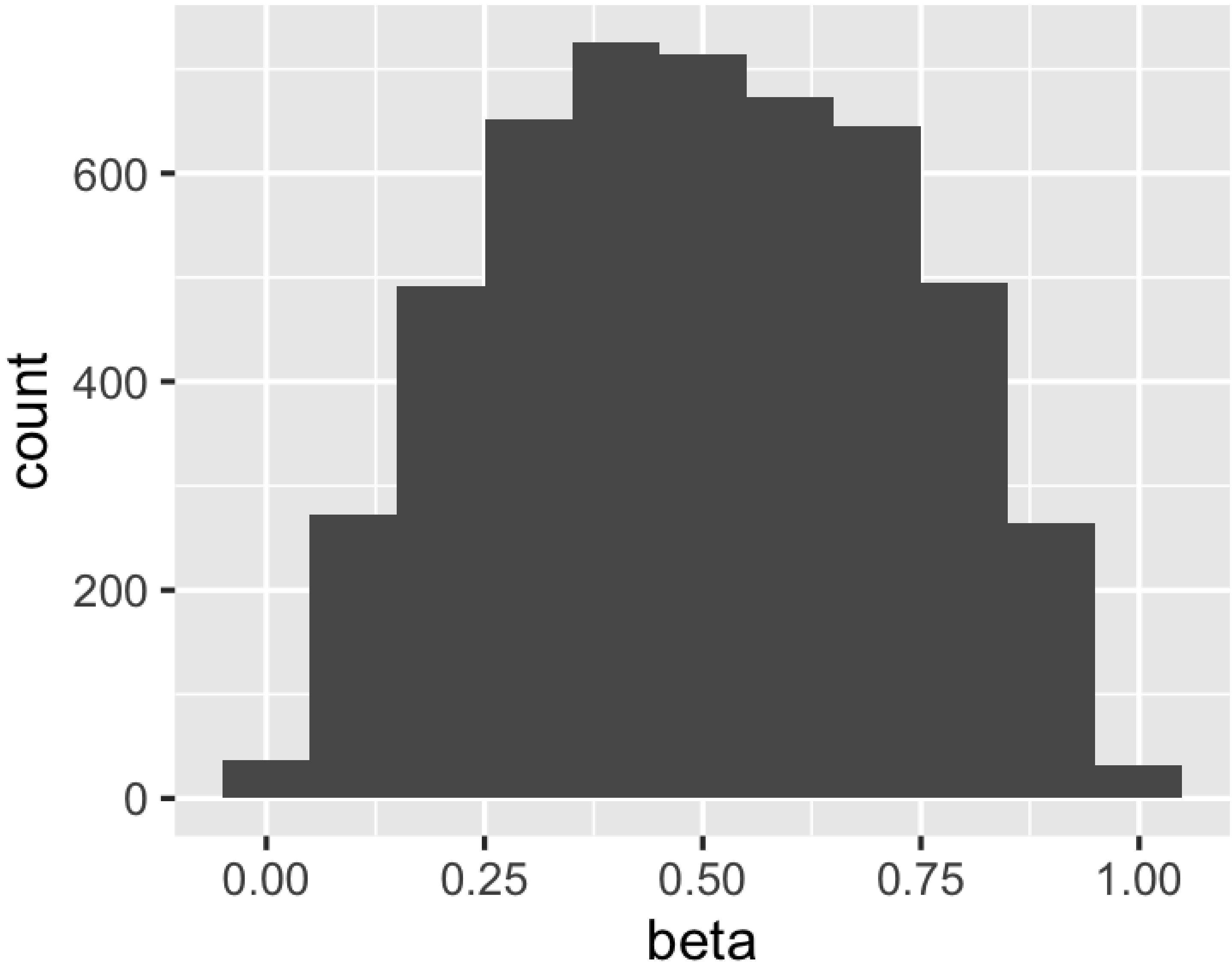
```
rbeta(5000, shape1 = 2, shape2 = 2)
```

```
[1] 0.2788 0.7495 0.6485 0.6665 0.6546 0.1575
...

[4996] 0.84719 0.35177 0.92796 0.67603 0.53960
```

```
randoms <- data.frame(
  beta = rbeta(5000, shape1 = 2, shape2 = 2)
)
```

```
ggplot(randoms, aes(beta)) +
  geom_histogram(binwidth = 0.1)
```



Random numbers seeds

```
set.seed(20000229)
```

```
rnorm(5)
```

```
-1.6538 -0.4028 -0.1654 -0.0734  0.5171
```

```
rnorm(5)
```

```
1.908  0.379 -1.499  1.625  0.693
```

```
set.seed(20000229)
```

```
rnorm(5)
```

```
-1.6538 -0.4028 -0.1654 -0.0734  0.5171
```

```
rnorm(5)
```

```
1.908  0.379 -1.499  1.625  0.693
```

Using a different seed

```
set.seed(20000229)
```

```
rnorm(5)
```

```
-1.6538 -0.4028 -0.1654 -0.0734  0.5171
```

```
rnorm(5)
```

```
1.908  0.379 -1.499  1.625  0.693
```

```
set.seed(20041004)
```

```
rnorm(5)
```

```
-0.6547 -0.7854 -0.0152  0.1514  0.5285
```

```
rnorm(5)
```

```
0.748  0.974  0.174 -0.781 -0.930
```

Let's practice!

SAMPLING IN R