We have data for a total of 1261 shots (excluding the first shot of each game)

TOTAL			**1261**

642 of those shots were shots where the previous shot was missed. 619 were shots where the previous shot was made.

	Previous shot missed	Previous shot made	
TOTAL	**642**	**619**	**1261**

For convenience, call these “not shots” and “hot shots”

	“Not Shots” Previous shot missed	**“Hot Shots”** Previous shot made	
TOTAL	**642**	**619**	**1261**

Of the 642 shots where the previous shot was missed, he missed 313 and made 329.

	“Not Shots” Previous shot missed	**“Hot Shots”** Previous shot made	
Missed this shot	313		
Made this shot	329		
TOTAL	**642**	**619**	**1261**

Of the 619 shots where the previous shot was made, he missed 334 and made 285.

	“Not Shots” Previous shot missed	**“Hot Shots”** Previous shot made	
Missed this shot	313	334	
Made this shot	329	285	
TOTAL	**642**	**619**	**1261**

Overall, he missed 647 shots and made 614 shots.

	“Not Shots” Previous shot missed	**“Hot Shots”** Previous shot made	**TOTAL**
Missed this shot	313	334	**647**
Made this shot	329	285	**614**
TOTAL	**642**	**619**	**1261**

He made 51% of “not shots” and 46% of “hot shots”.

	“Not Shots” Previous shot missed	**“Hot Shots”** Previous shot made	**TOTAL**
Missed this shot	313	334	**647**
Made this shot	329	285	**614**
TOTAL	**642**	**619**	**1261**
	329/642 = **0.51**	285/619 = **0.46**	

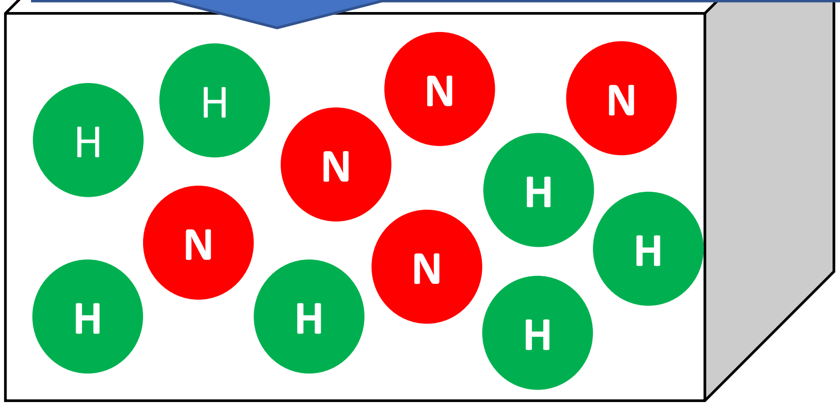
He made 5% fewer hot shots than not shots. Do we believe that he’s truly worse at hot shots? Or could the 5% difference just be due to random chance?

	“Not Shots” Previous shot missed	**“Hot Shots”** Previous shot made	**TOTAL**
Missed this shot	313	334	**647**
Made this shot	329	285	**614**
TOTAL	**642**	**619**	**1261**
	329/642 = **0.51**	285/619 = **0.46**	

What our simulation does in theory…

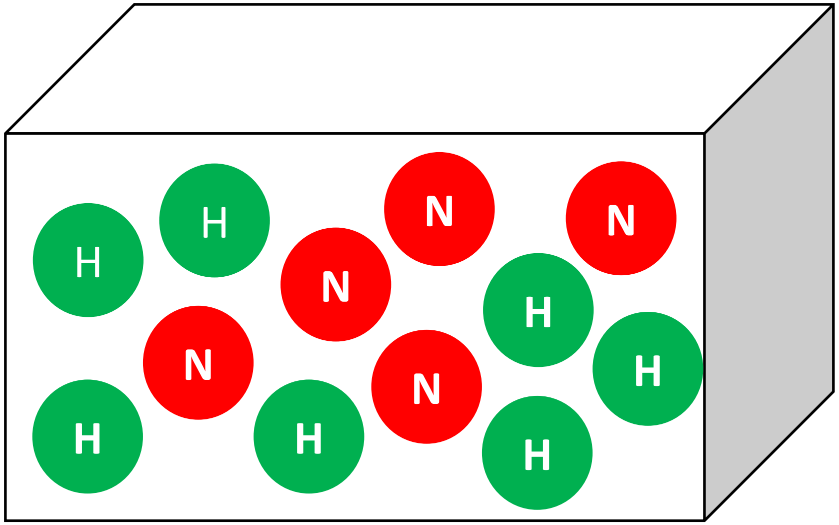
	“Not Shots” Previous shot missed	“Hot Shots” Previous shot made	TOTAL
Missed this shot	313	334	**647**
Made this shot	329	285	**614**
TOTAL	**642**	**619**	**1261**

Fill a box with 642 balls labeled N (not shots) and 619 ball labeled H (hot shots)

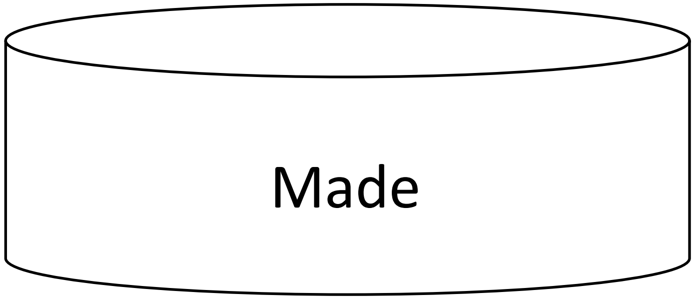


What our simulation does in theory…

	“Not Shots” Previous shot missed	“Hot Shots” Previous shot made	TOTAL
Missed this shot	313	334	**647**
Made this shot	329	285	**614**
TOTAL	642	619	**1261**



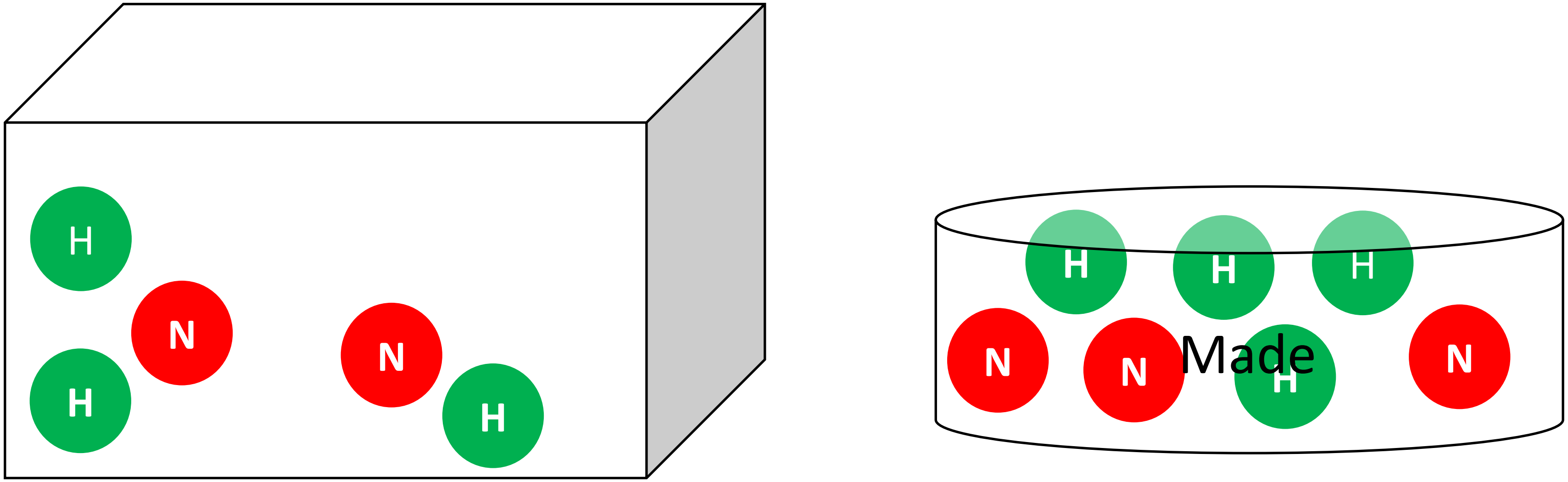
Get a bucket and label it “made”.
All the balls that end up in here will stand for shots made.



What our simulation does in theory…

	"Not Shots" Previous shot missed	"Hot Shots" Previous shot made	TOTAL
Missed this shot	313	334	**647**
Made this shot	329	285	**614**
TOTAL	642	619	**1261**

Randomly pick 614 balls out of the box and put them in the made bucket

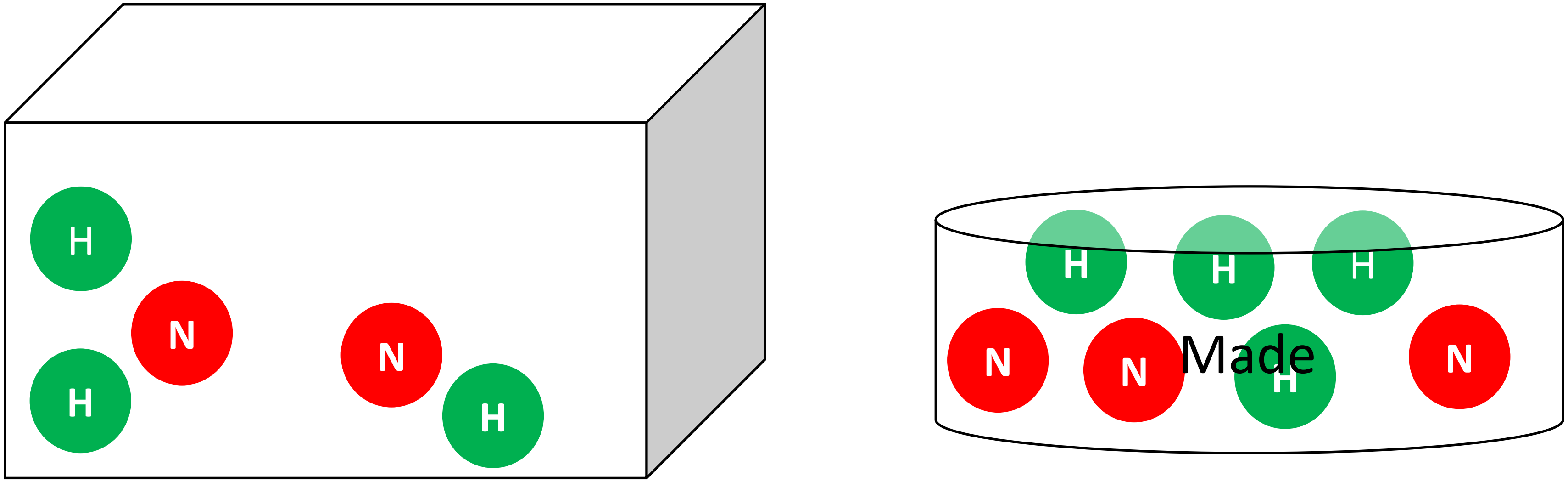


What our simula theory…

Stop and think.
Why are we picking 614 balls???

	"Not Shots" Previous shot missed	"Hot Shots" Previous shot made	TOTAL
Missed this shot	313	334	**647**
Made this shot	329	285	**614**
TOTAL	642	619	**1261**

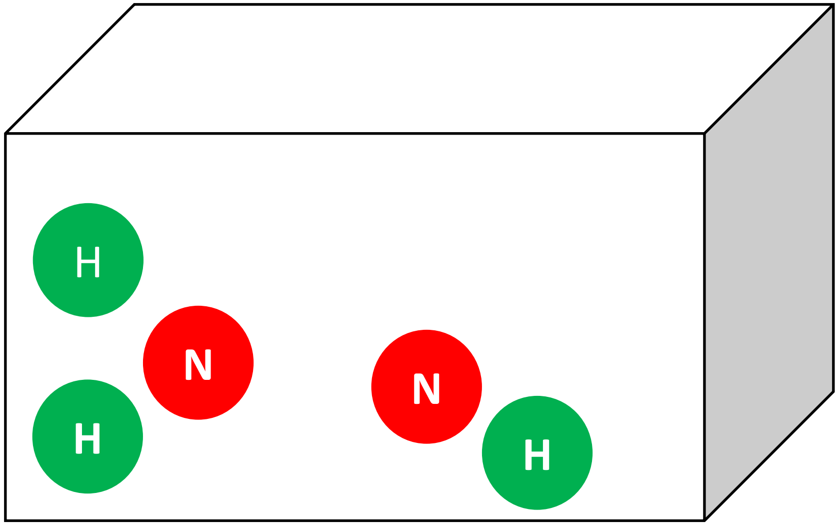
Randomly pick 614 balls out of the box and put them in the made bucket

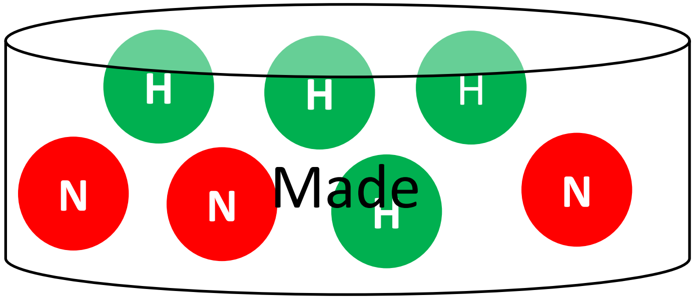


What our simulation does in theory…

	“Not Shots” Previous shot missed	“Hot Shots” Previous shot made	TOTAL
Missed this shot			647
Made this shot			614
TOTAL	642	619	1261

Count the number of H’s and N’s in the made bucket and record.



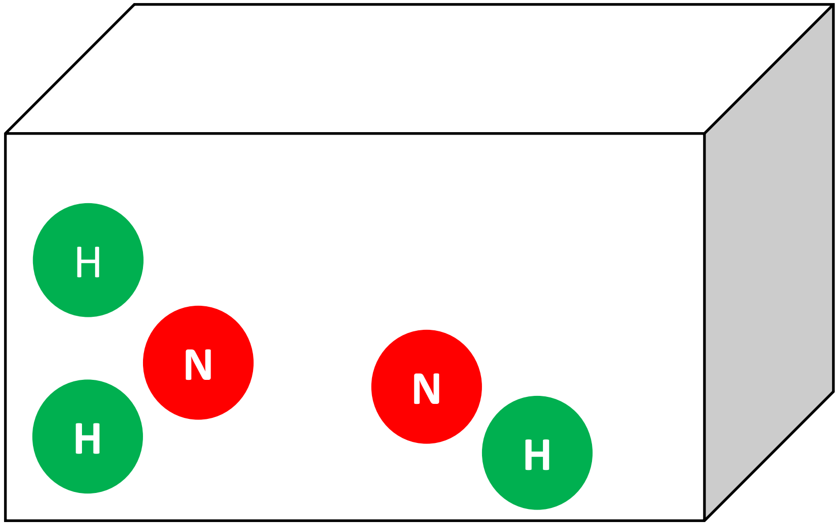


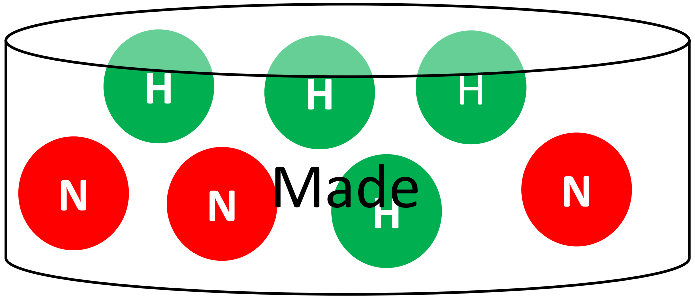
What our simulation does in theory…

	"Not Shots" Previous shot missed	"Hot Shots" Previous shot made	TOTAL
Missed this shot			647
Made this shot	300	314	614
TOTAL	642	619	1261

Count the number of H's and N's in the made bucket and record.

(Note: will likely be different every time, but imagine, for example, that we got 300 Ns and 314 Hs this time)

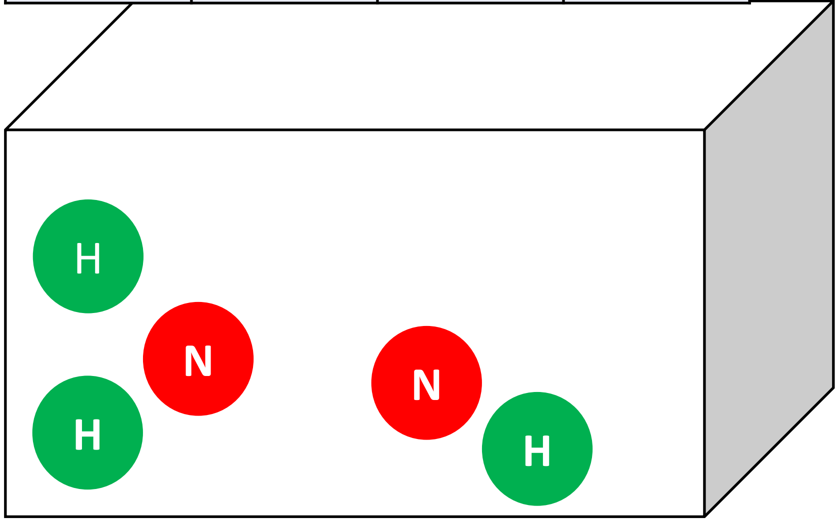


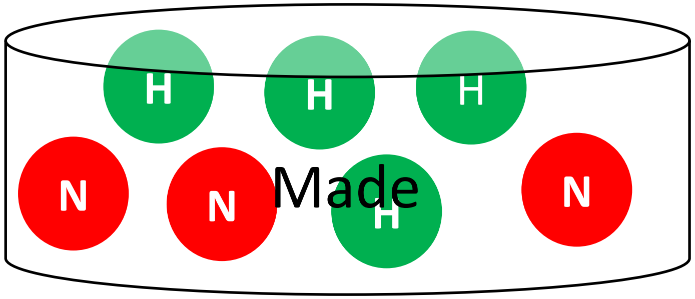


What our simulation does in theory…

	"Not Shots" Previous shot missed	"Hot Shots" Previous shot made	TOTAL
Missed this shot			647
Made this shot	300	314	614
TOTAL	642	619	1261
	300/642 = 47%	314/619= 51%	

Calculate the percentage of hot shots made and not shots made

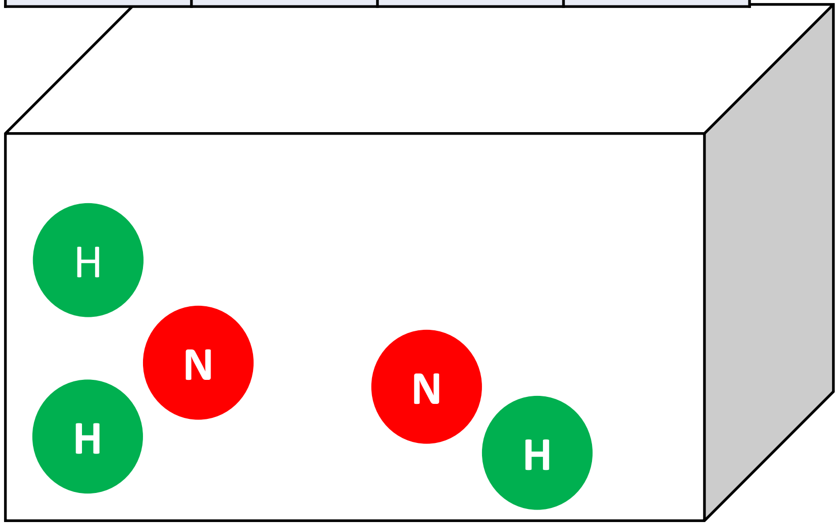


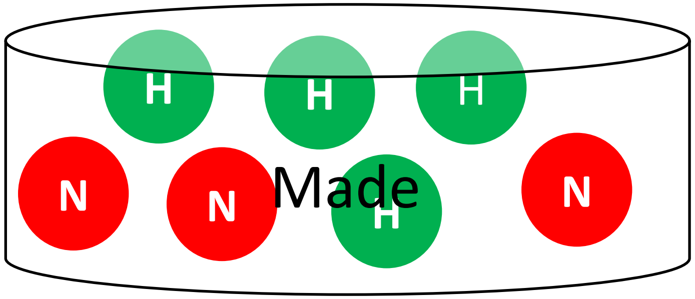


What our simulation does in theory…

	"Not Shots" Previous shot missed	"Hot Shots" Previous shot made	TOTAL
Missed this shot			647
Made this shot	300	314	614
TOTAL	642	619	1261
	300/642 = .47	314/619= .51	

Subtract (Hot Shots Percentage – Not Shots Percentage) to find the difference. (e.g. .51 - .47 = .04)



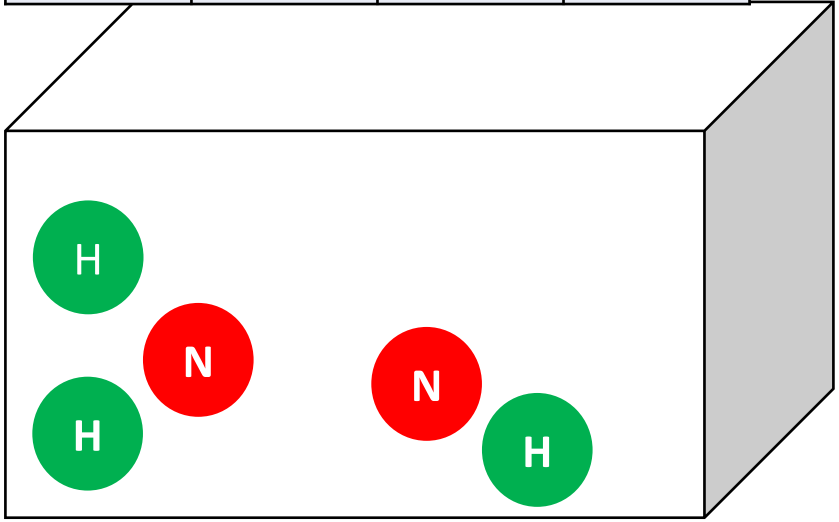


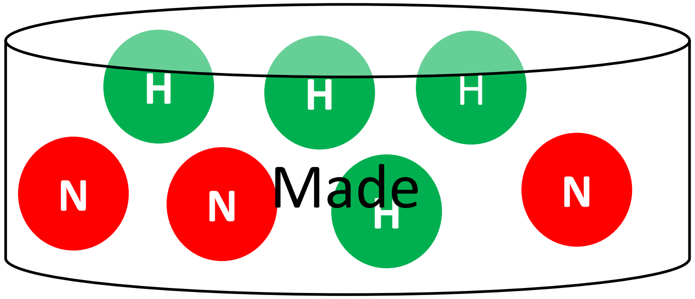
What our simulation does in theory…

	"Not Shots" Previous shot missed	"Hot Shots" Previous shot made	TOTAL
Missed this shot			647
Made this shot	300	314	614
TOTAL	642	619	1261
	300/642 = .47	314/619= .51	

Subtract (Hot Shots Percentage – Not Shots Percentage) to find the difference. (e.g. .51 - .47 = .04)

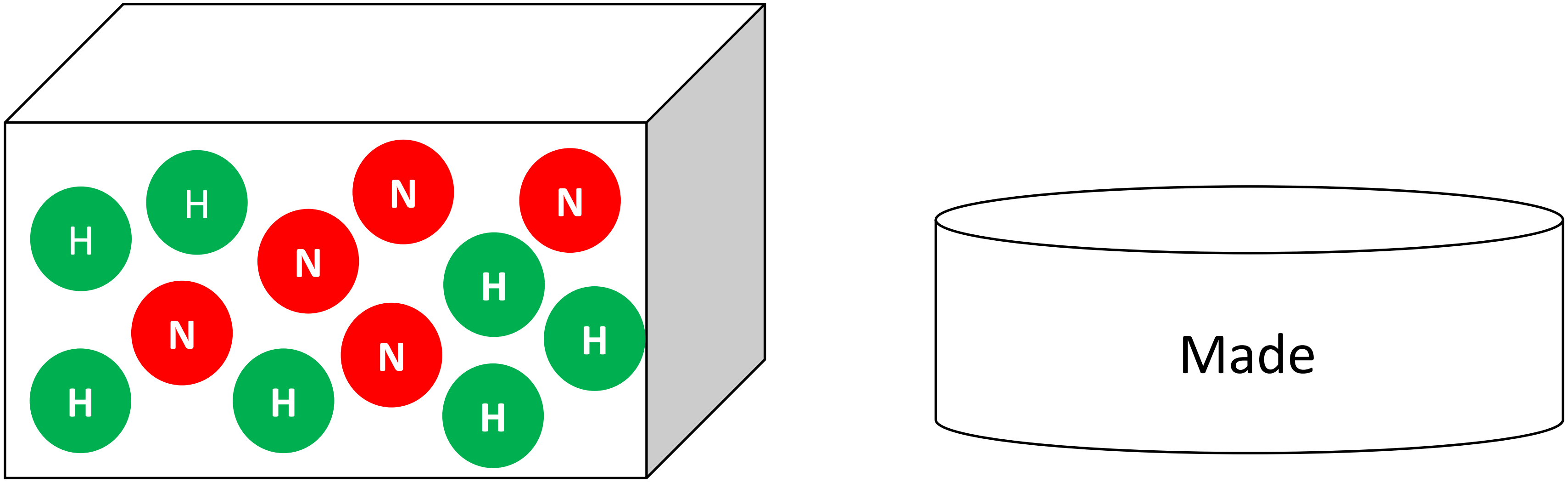
Record the percentage difference (e.g. .04). Then put all balls back in the box.





What our simulation does in theory…

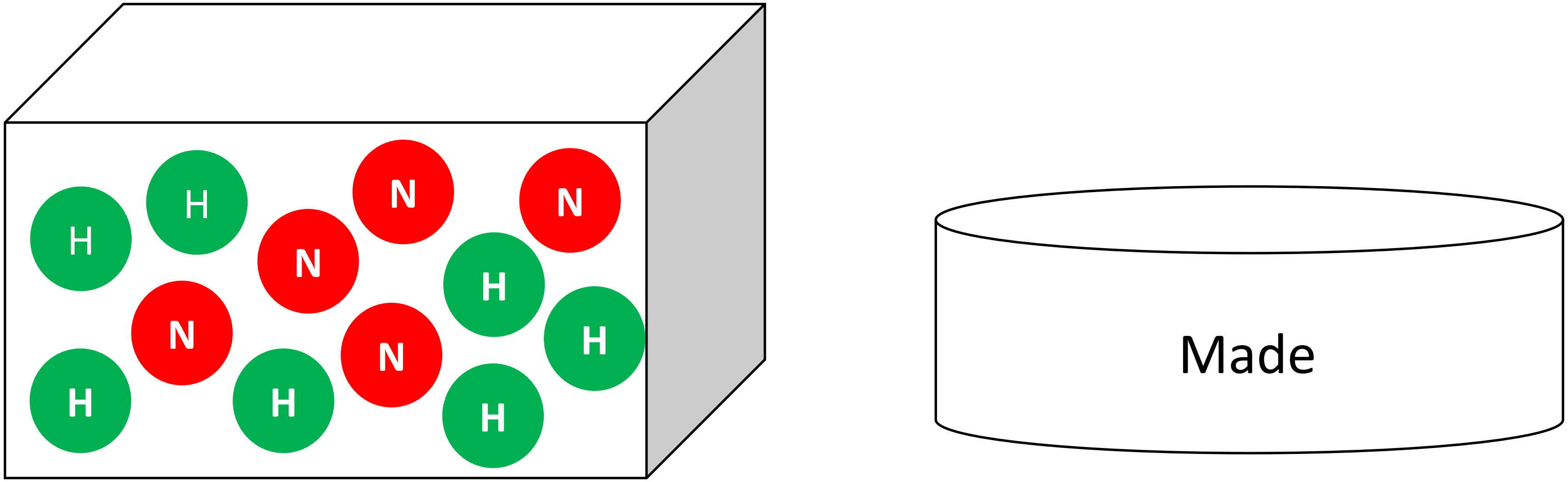
	"Not Shots" Previous shot missed	"Hot Shots" Previous shot made	TOTAL
Missed this shot			647
Made this shot			614
TOTAL	642	619	1261



What our simulation does in theory…

	"Not Shots" Previous shot missed	**"Hot Shots"** Previous shot made	**TOTAL**
Missed this shot			**647**
Made this shot			**614**
TOTAL	**642**	**619**	**1261**

Repeat 1,000 times. Each time record the difference between hot shots percent made and not shots percent made.



What the code actually does

What the code actually does

A few things to remember before we start:

- lag_data is a data frame that has all of the original Curry data, plus a new column we made called “lag_shot”
- The lag_shot column says “TRUE” if the previous shot was made and “FALSE” if the previous shot was missed

What the code actually does

A few things to remember before we start:

- lag_data is a data frame that has all of the original Curry data, plus a new column we made called “lag_shot”
- The lag_shot column says “TRUE” if the previous shot was made and “FALSE” if the previous shot was missed

```
# Number of shots taken after shots that were made
hot_shots <- lag_data %>%
     filter(lag_shot) %>%
     nrow()
```

This code says: take the data frame “lag_data, then filter it by giving me only the rows where the column lag_shot is “TRUE”, then count the number of rows you gave me. Finally, store that value in the variable “hot_shots”

What the code actually does

A few things tart:

- lag_data is f the original Curry data de called “lag_shot”
- The lag_sh he previous shot was made s shot was missed

	“Not Shots” Previous shot missed	“Hot Shots” Previous shot made	**TOTAL**
Missed this shot	313	334	**647**
Made this shot	329	285	**614**
TOTAL	**642**	**619**	**1261**

```
# Number of shots taken after shots that were made
hot_shots <- lag_data %>%
     filter(lag_shot) %>%
     nrow()
```

Check for understanding: After running this code, “hot_shots” should contain a single number. Specifically, it will be one of the numbers in the table above. Which number should it contain? Check if you’re right by typing “hot_shots” in the console and see what value it returns.

Thi lter
it b ot
is “ e.
Fin

What the code actually does

```
# Number of shots taken after shots that were made
hot_shots <- lag_data %>%
        filter(lag_shot) %>%
        nrow()

# Number of shots made after shots that were made
hot_made <- lag_data %>%
        filter(lag_shot & SHOT_MADE) %>%
        nrow()

# Number of shots taken after shots that were missed
not_shots <- lag_data %>%
        filter(!lag_shot) %>%
        nrow()

# Number of shots made after shots that were missed
not_made <- lag_data %>%
        filter(!lag_shot & SHOT_MADE) %>%
        nrow()
```

	“Not Shots” Previous shot missed	**“Hot Shots”** Previous shot made	**TOTAL**
Missed this shot	313	334	**647**
Made this shot	329	285	**614**
TOTAL	**642**	**619**	**1261**

Check for Understanding: The first block of code above is the one we just discussed. Look carefully at the next three blocks of code. Can you figure out what each one does? Which of the numbers from the table should be stored in the variables “hot_made”, “not_shots”, and “not_made”? Check if you’re right by typing these variable names into the console (or look for them in the environment window).

What the code actually does

```
# Number of shots taken after shots that were made
hot_shots <- lag_data %>%
        filter(lag_shot) %>%
        nrow()

# Number of shots made after shots that were made
hot_made <- lag_data %>%
        filter(lag_shot & SHOT_MADE) %>%
        nrow()

# Number of shots taken after shots that were missed
not_shots <- lag_data %>%
        filter(!lag_shot) %>%
        nrow()

# Number of shots made after shots that were missed
not_made <- lag_data %>%
        filter(!lag_shot & SHOT_MADE) %>%
        nrow()
```

	"Not Shots" Previous shot missed	**"Hot Shots"** Previous shot made	**TOTAL**
Missed this shot	313	334	**647**
Made this shot	329	285	**614**
TOTAL	**642**	**619**	**1261**

Tip: Write down what the four variables (hot_shots, hot_made, not_shots, and not_made) represent and what numbers they equal. It will make understanding the next block of code much easier.

Check for Understanding: The first block of code above is the one we just discussed. Look carefully at the next three blocks of code. Can you figure out what each one does? Which of the numbers from the table should be stored in the variables "hot_made", "not_shots", and "not_made"? Check if you're right by typing these variable names into the console (or look for them in the environment window).

What the code actually does

```
simulate_null <- function() {

# Make a list with the right number of shots of each type
shots <- c(rep("Hot", hot_shots), rep("Not", not_shots))
```

This says, make a list called “shots” that says “Hot” 619 times and then “Not” 642 times. Do you see how it does that? Tip: Type “shots” into the console to see what this looks like.

```
# randomly select the made shots from this list
made <- sample(shots, hot_made + not_made)
```

This says, create a new list called “made” and fill it by randomly picking 614 items from the list “shots”. Do you see how it does that?

What the code actually does

```
simulate_null <- function() {

# Make a list with the right number of shots of each type
shots <- c(rep("Hot", hot_shots), rep("Not", not_shots))
```

This says, make a list called “shots” that says “Hot” 619 times and then “Not” 642 times. Do you see how it does that? Tip: Type “shots” into the console to see what this looks like.

```
# randomly select the made shots from this list
made <- sample(shots, hot_made + not_made)
```

This says, create a new list called “made” and fill it by randomly picking 614 items from the list “shots”. Do you see how it does that?

Check for understanding: Earlier, we described what the simulation does “in theory” by imagining drawing balls from a box. What part of that theoretical description does the list “shots” correspond to? What part does the list “made” correspond to?

What the code actually does

(Note: They grayed out code was discussed on the previous slide)

```
simulate_null <- function() {

# Make a list with the right number of shots of each type
shots <- c(rep("Hot", hot_shots), rep("Not", not_shots))

# randomly select the made shots from this list
made <- sample(shots, hot_made + not_made)

# Compute the difference shot success between hot and not shots
random_hot_made <- sum(made == "Hot") / hot_shots
random_not_made <- sum(made == "Not") / not_shots
random_hot_made - random_not_made
}
```

Check for understanding: Can you figure out what this last block of code is doing? Hint: Think back to the theoretical description of the simulation. Given everything we’ve done so far, what’s left to do?

What the code actually does

(Note: They grayed out code was discussed on the previous slide)

```
simulate_null <- function() {

# Make a list with the right number of shots of each type
shots <- c(rep("Hot", hot_shots), rep("Not", not_shots))

# randomly select the made shots from this list
made <- sample(shots, hot_made + not_made)

# Compute the difference shot success between hot and not shots random_hot_made <- sum(made
== "Hot") / hot_shots random_not_made <- sum(made == "Not") / not_shots random_hot_made -
random_not_made
}
```

Notice that we've taken all of the above code and wrapped it in a function using {}. Basically, we're telling R to make a new function called "simulate_null". This means that from now on, every time I type "simulate_null", R does everything inside the {}. For example, try typing the following into a new chunk (after running the code above):

```
x <- simulate_null()
x
```

Run this code several times. You should see it spit out a different number each time. What does that number represent? Why is it different each time?

What the code actually does

```
null_samples <- data_frame(diff = replicate(1000,
simulate_null()))
```

This says to run the function “simulate_null” 1000 times and to store the results in a column called “diff” in a data frame called “null_samples”

```
empirical_diff <- hot_made/hot_shots - not_made/not_shots
```

This has nothing to do with the simulation. It’s based on the original data. Can you figure out what it does?

```
ggplot(null_samples, aes(x = diff)) +  geom_histogram(bins =
100) +   geom_vline(xintercept = empirical_diff, color =
"darkred", size = 2)
```

See if you can figure out what this code does on your own!