A/B Testing for Data Science

(with Python and R)

Carlos Afonso (Data Science Instructor)

NYC Data Science Academy

June 30, 2021

Outline & Learning Objectives

Outline

A/B Testing

- What? & Why?
- Examples
- How?
- Hypothesis Testing (p-value)

Permutation Test

- What? & Why?
- How?

Code

- From Scratch (in R & Python)
- (*Extra*) General Solution, R Shiny App

Q & A

Learning Objectives

Statistics

- A/B testing
- Hypothesis testing
- p-value
- Permutation test

Coding

- variables
- loops
- functions

A/B Test - What? & Why?

What is an A/B test?

An experiment to compare two competing **options** (A, B).

- **Options:** treatments (medical), designs (ad, web), products, prices, etc.

Why use an A/B test?

To determine if the options are **different**.

- **Different** in a statistical sense (hypothesis testing, permutation test).

To determine what's the **better** option.

- **Better** for the question/goal at hand (e.g., customer acquisition, profit).

A/B Test - Examples?

A/B Test - Examples? - General

Two soil treatments: which one promotes ***better seed germination***?

Two web headlines: which one produces ***more clicks***?

Two web ads: which one generates ***more conversions***?

Two prices:

- which one yields a ***higher net profit***?
- which one leads to ***more new costumers***?

Two therapies: which one is ***more effective at suppressing cancer***?

- **Control group:** subjects exposed to ***no treatment*** or ***standard treatment***.
- **Treatment group:** subjects exposed to the ***new treatment***.

A/B Test - Examples? - Specific

Microsoft (Bing):

- ***one A/B test:*** changing the way the Bing search engine displayed ad headlines
- led to a 12% increase in revenue;
- that's more than $100 million per year in the US alone

Amazon:

- moving credit card offers from its home page to the shopping cart page
- boosted profits by tens of millions of dollars annually

Google & Bing: only 10% to 20% of experiments generate positive results

Reference: The Surprising Power of Online Experiments (Harvard Business Review, 2017)

A/B Test - How? - Steps

(**0. Idea & Definition:** question, goal, data/subjects, options, test statistic.)

1. Subjects: Set of all subjects.

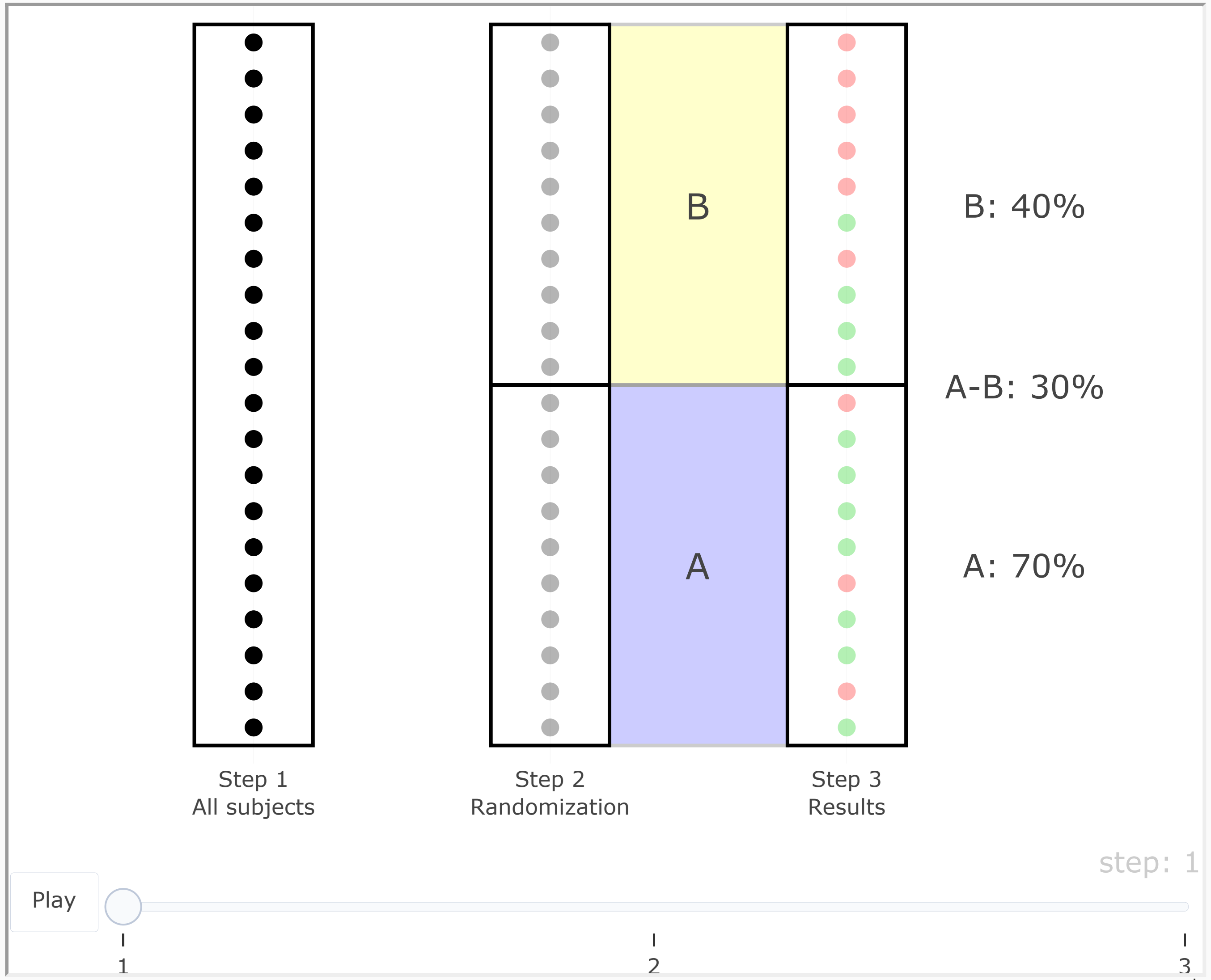
2. Randomization: Randomly assign subjects to the two groups (A, B).

3. Results: Expose subjects to options (A, B), measure results, and compute **test statistic**.

4. Hypothesis testing: determine if the observed difference is statistically significant.

- Can be done with a **permutation test**.

(**5. Action/Decision:** based on test results.)



A/B Test - How? - Didactic Example

A/B test observed results

	Option A	Option B
Total	10	10
No	3	6
Yes	7	4
Yes Rate (%)	70 %	40 %

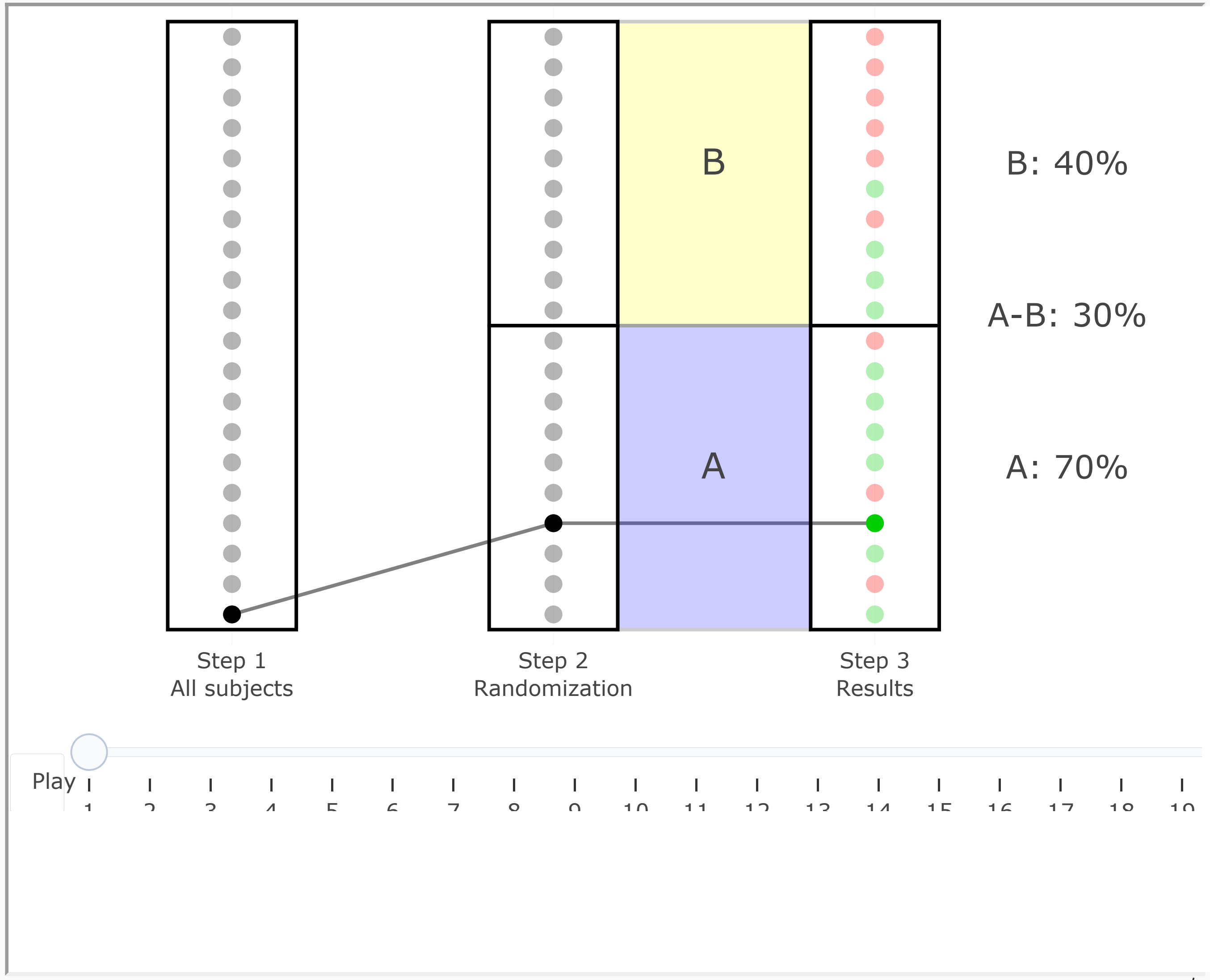
Test statistic: Yes Rate Difference

- (A - B) (%) = 70% - 40% = 30%

Is this a (statistically) significant difference?

- Is this difference just due to random change?
- Or, is it due the different options (A, B)?

Note: using small numbers for didactic purposes.



A/B Test - How? - Hypothesis Testing (step 4)

Is the observed difference (30%) statistically significant?

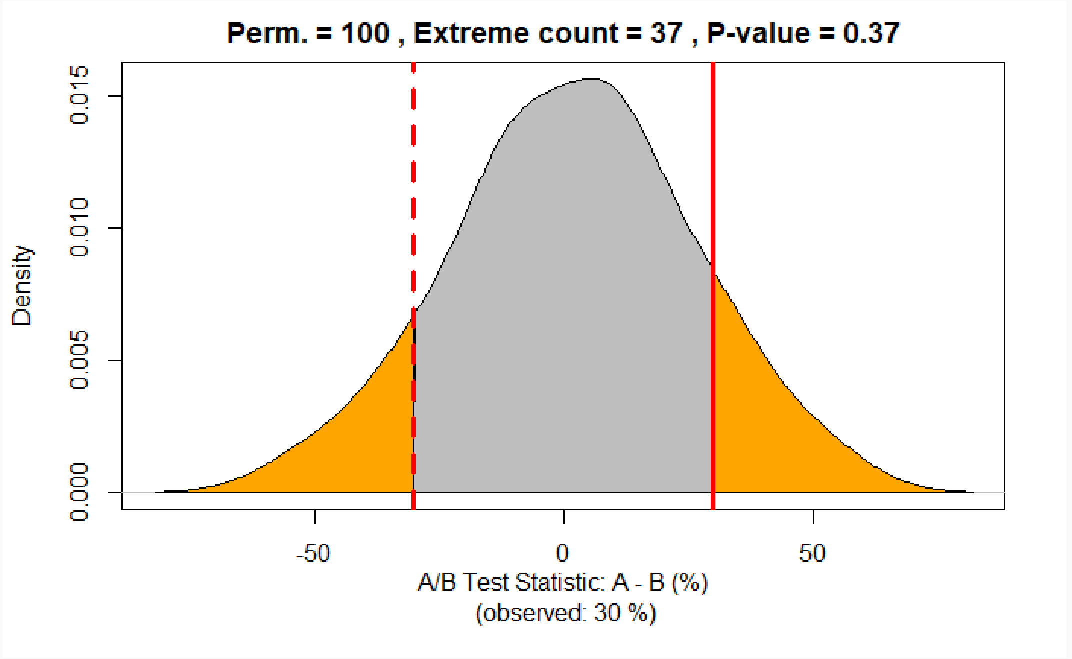
Thanks to **randomization** any observed difference between A and B must be due to either:

- *Null hypothesis*: **Random chance** (subject assignment)
- *Alternative hypothesis*: **Real difference** between A & B

Hypothesis testing: Is random chance (Null hypothesis) a reasonable explanation for the observed difference?

- Assumes that the Null hypothesis is true
- Creates corresponding Null model (probability model)
- Tests whether the observed difference is a reasonable outcome of that Null model
- Is the observed difference within the random variability of the Null model?

	Two-way test	**One-way test**
Null hypothesis	$A = B$ (same)	$A \leq B$ (A is not better)
Alternative hypothesis	$A \neq B$ (different)	$A > B$ (A is better)



A/B Test - How? - Hypothesis Testing (p-value)

P-value:

- given a random chance (probability) model that embodies the Null hypothesis,
- the p-value is the probability of obtaining results as unusual/extreme as the observed result.

Significance level (alpha):

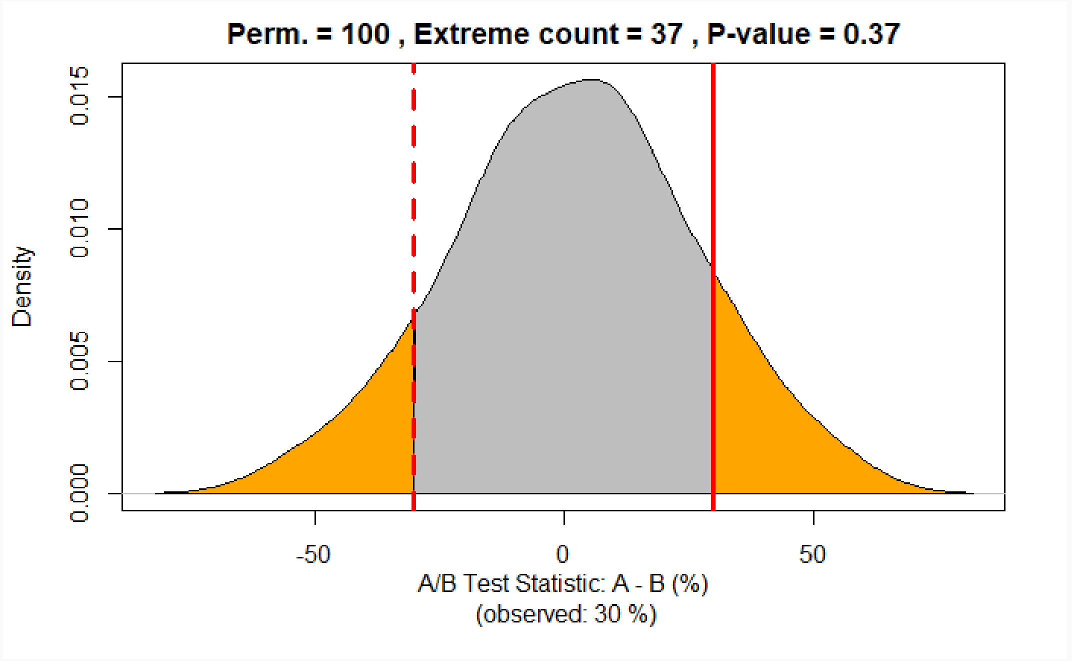
- the probability threshold of "unusualness" (e.g., 0.05)
- must be defined before the experiment
- the probability we accept for a type I error

Decision:

- $\text{p-value} \geq \text{alpha}$: retain the null hypothesis
  - observed difference due to random chance
- $\text{p-value} < \text{alpha}$: reject the null hypothesis
  - observed difference is real/significant

Type I error (false positive): Mistakenly concluding that an effect is real (when it is due to chance). Probability = alpha

Type II error (false negative): Mistakenly concluding that an effect is due to chance (when it is real).



Permutation Test - What? & Why?

What? & Why? Permutation test is a resampling procedure used for hypothesis testing.

Resampling: repeatedly sample values from the observed data to assess a statistic's random variability.

Two main types of resampling procedures:

- **Boostrap:** resampling **with** replacement, used to assess reliability of an estimate.
- **Permutation:** resampling **without** replacement, used for hypothesis testing.

Permutation Test:

- A resampling procedure used for hypothesis testing
- Process for combining two (or more) data samples together, and randomly (or exhaustively) reallocating the observations to resamples to assess the random variability of the test statistic
- A way to create the Null model, and compute the p-value
- Advantage: No assumptions (creates Null model from the data itself)

Permutation Test - How? - Steps (diagram 1)

Step 0: A/B test results.

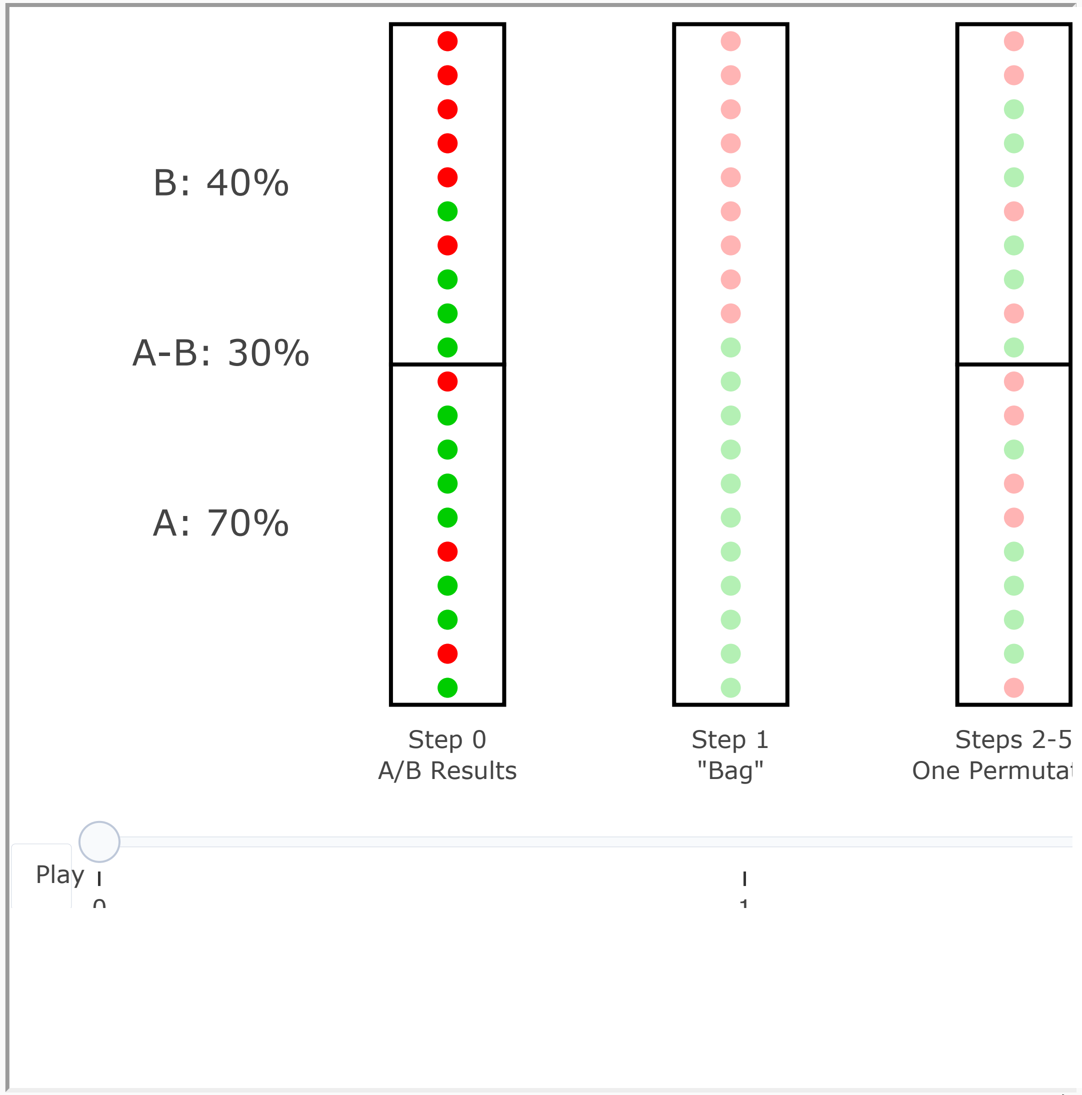
Step 1: Put all A/B test results in a single dataset ("bag").

Steps 2 to 5: Do one permutation:

- **Step 2:** Shuffle the "bag".
- **Step 3:** Randomly draw (without replacement) a resample of size of group A.
- **Step 4:** Randomly draw (without replacement) a resample of size of group B (the remainder).
- **Step 5:** Record the test statistic for resamples.

Step 6: Do many permutations: to yield a permutation distribution of the test statistic (Null model).

Hypothesis testing: Use permutation results to compute the p-value as the ratio of values that are as or more extreme than the observed test statistic.



Permutation Test - How? - Steps (diagram 2)

Step 0: A/B test results.

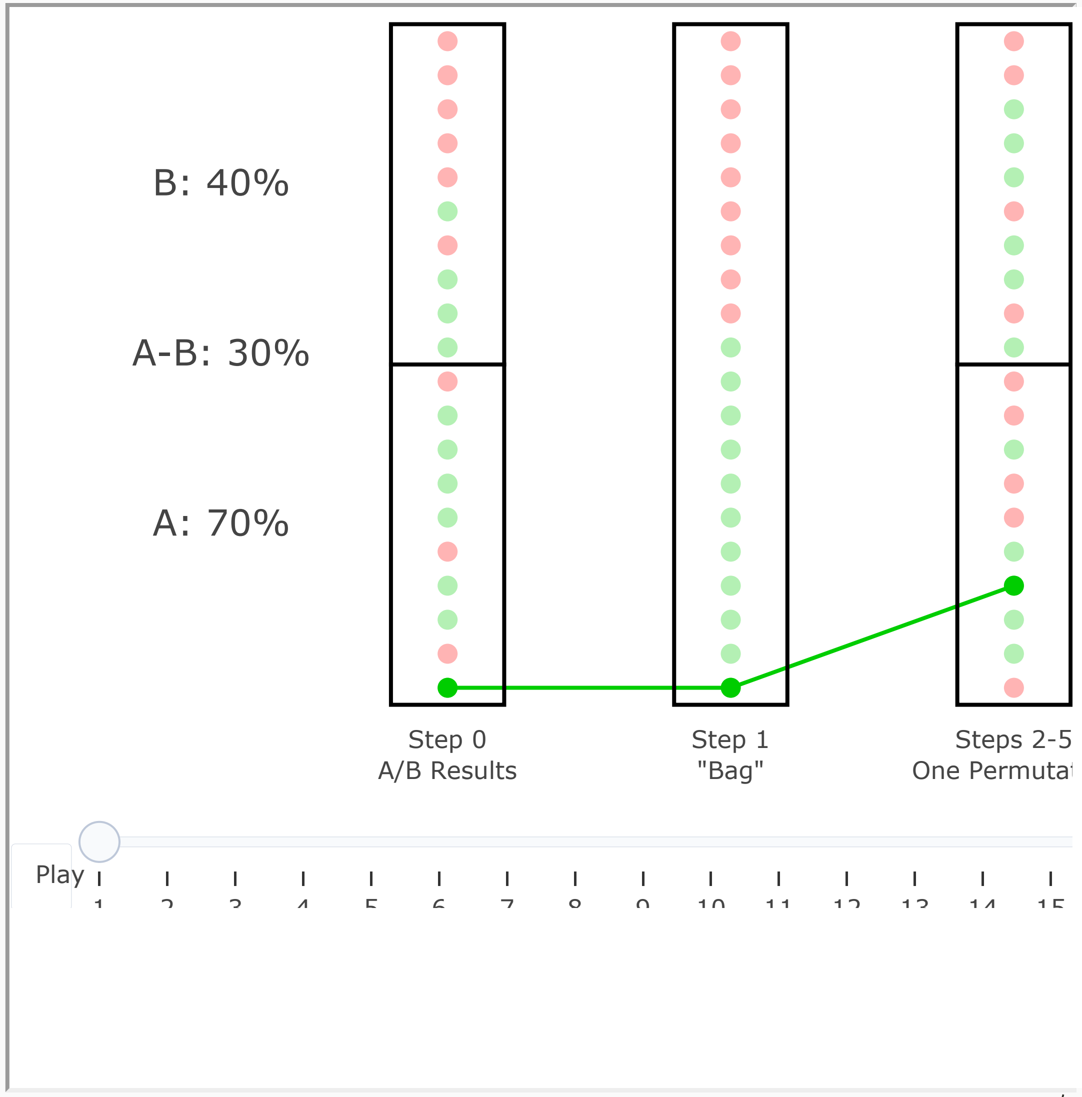
Step 1: Put all A/B test results in a single dataset ("bag").

Steps 2 to 5: Do one permutation:

- **Step 2:** Shuffle the "bag".
- **Step 3:** Randomly draw (without replacement) a resample of size of group A.
- **Step 4:** Randomly draw (without replacement) a resample of size of group B (the remainder).
- **Step 5:** Record the test statistic for resamples.

Step 6: Do many permutations: to yield a permutation distribution of the test statistic (Null model).

Hypothesis testing: Use permutation results to compute the p-value as the ratio of values that are as or more extreme than the observed test statistic.



Permutation Test - How? - Hypothesis Testing

A/B test observed difference: (A - B)(%) = 30%

Counts of results from **100 permutations**:

Result	**-50**	**-30**	**-10**	**10**	**30**	**50**
Counts	4	12	30	33	16	5

Two-way test (Null: $A = B$, Alternative: $A \neq B$)

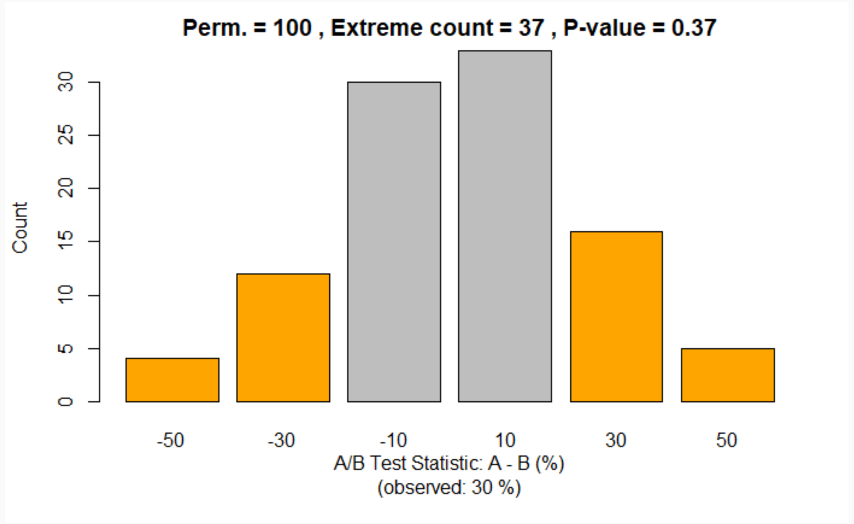
- Extreme values count: **37** = 4 + 12 + 16 + 5
- Extreme values ratio (p-value): **0.37** = 37 / 100

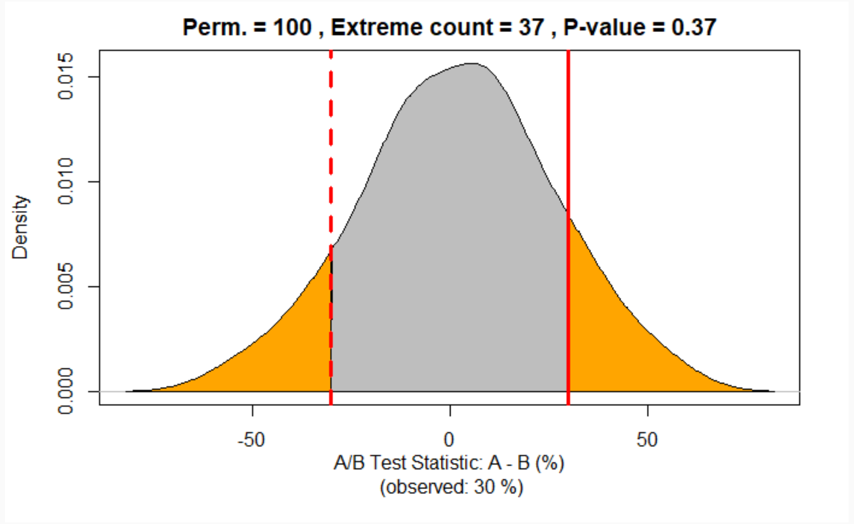
One-way test (Null: $A \leq B$, Alternative: $A > B$)

- Extreme (positive) values count: **21** = 16 + 5
- Extreme (positive) values ratio (p-value): **0.21** = 21 / 100

Decision: $\text{p-value} \geq \text{alpha}\ (0.05)$: retain null hypothesis

- observed difference due to random chance





Code (R & Python) - Libraries

```r
# === R ===

# No additional libraries
# (everything done with base R)

# R version used
version$version.string
```

```
## [1] "R version 4.1.0 (2021-05-18)"
```

```python
# === Python ===

import pandas as pd

from platform import python_version

from random import sample
from random import seed

print("Python version:", python_version())
```

```
## Python version: 3.9.4
```

Code (R & Python) - A/B Test - Didactic Example

```r
# ==== R ====

# Variables that fully define the A/B test results
t = 20     # Total number of subjects (A + B)
a = 10     # Number of subjects in group A
b = 10     # Number of subjects in group B
a_yes = 7  # Yes count in group A
b_yes = 4  # Yes count in group B

# Compute remaining A/B test results
t_yes = a_yes + b_yes       # Total Yes count
t_no  = t - t_yes           # Total No count
a_yes_pc = 100 * a_yes / a  # Yes percentage in A
b_yes_pc = 100 * b_yes / b  # Yes percentage in B

# A/B Test Statistic: Yes percentage difference (A-B)
ab_yes_pc = a_yes_pc - b_yes_pc

cat('Observed Yes Rate (%):  A:', a_yes_pc,
    ',  B:', b_yes_pc, ',  A-B:', ab_yes_pc,
    '\nTotal counts:  Yes:', t_yes, ',  No:', t_no)
```

```
## Observed Yes Rate (%):  A: 70 ,  B: 40 ,  A-B: 30
## Total counts:  Yes: 11 ,  No: 9
```

```python
# ==== Python ====

# Variables that fully define the A/B test results
t = 20     # Total number of subjects (A + B)
a = 10     # Number of subjects in group A
b = 10     # Number of subjects in group B
a_yes = 7  # Yes count in group A
b_yes = 4  # Yes count in group B

# Compute remaining A/B test results
t_yes = a_yes + b_yes       # Total Yes count
t_no  = t - t_yes           # Total No count
a_yes_pc = 100 * a_yes / a  # Yes percentage in A
b_yes_pc = 100 * b_yes / b  # Yes percentage in B

# A/B Test Statistic: Yes percentage difference (A-B)
ab_yes_pc = a_yes_pc - b_yes_pc

print('Observed Yes Rate (%):  A:', a_yes_pc,
      ',  B:', b_yes_pc, ',  A-B:', ab_yes_pc,
      '\nTotal counts:  Yes:', t_yes, ',  No:', t_no)
```

```
## Observed Yes Rate (%):  A: 70.0 ,  B: 40.0 ,  A-B: 30.0
## Total counts:  Yes: 11 ,  No: 9
```

Code (R & Python) - Perm. Test - One Permutation

```r
# === R ===

set.seed(0)  # for reproducible results
bag1 = c(rep(1, t_yes), rep(0, t_no)) # S1: All results bag
bag2 = sample(bag1)                   # S2: Shuffle bag
a_rs = bag2[1:a]                      # S3: Random sample A
b_rs = bag2[(a+1):(a+b)]              # S4: Random sample B

# Step 5: Compute the test statistic
a_yes_pc_rs = 100 * sum(a_rs) / a
b_yes_pc_rs = 100 * sum(b_rs) / b
ab_yes_pc_rs = a_yes_pc_rs - b_yes_pc_rs

cat('(1) Bag         :', bag1,
    '\n(2) Bag shuffled:', bag2,
    '\n(3) A resample  :', a_rs,
    '\n(4) B resample  :', rep(' ', a), b_rs,
    '\n(5) Resample Yes Rate (%):  A:', a_yes_pc_rs, ',  B:', b_yes_
    '\n(0) Observed Yes Rate (%):  A:', a_yes_pc, ',  B:', b_yes_pc,
```

```python
# === Python ===

seed(2)  # for reproducible results
bag1 = [1]*t_yes + [0]*t_no     # S1: All results bag
bag2 = sample(bag1, len(bag1))  # S2: Shuffle bag
a_rs = bag2[:a]                 # S3: Random sample A
b_rs = bag2[a:]                 # S4: Random sample B

# Step 5: Compute the test statistic
a_yes_pc_rs = 100 * sum(a_rs) / a
b_yes_pc_rs = 100 * sum(b_rs) / b
ab_yes_pc_rs = a_yes_pc_rs - b_yes_pc_rs

print('(1) Bag         :', str(bag1).replace(',', ''),
      '\n(2) Bag shuffled:', str(bag2).replace(',', ''),
      '\n(3) A resample  :', str(a_rs).replace(',', ''),
      '\n(4) B resample  :', ' '*(2*a-1), str(b_rs).replace(',', ''))
      '\n(5) Resample Yes Rate (%):  A:', a_yes_pc_rs, ',  B:', b_ye
      '\n(0) Observed Yes Rate (%):  A:', a_yes_pc, ',  B:', b_yes_p
```

```
## (1) Bag         : 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0
## (2) Bag shuffled: 0 1 1 1 1 0 0 1 0 0 1 0 1 1 0 1 1 1 0 0
## (3) A resample  : 0 1 1 1 1 0 0 1 0 0
## (4) B resample  :                     1 0 1 1 0 1 1 1 0 0
## (5) Resample Yes Rate (%):  A: 50 ,  B: 60 ,  A-B: -10
## (0) Observed Yes Rate (%):  A: 70 ,  B: 40 ,  A-B: 30
```

```
## (1) Bag         : [1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0]
## (2) Bag shuffled: [1 1 0 0 1 0 0 1 1 0 1 1 1 0 0 0 1 0 1 1]
## (3) A resample  : [1 1 0 0 1 0 0 1 1 0]
## (4) B resample  :                     [1 1 1 0 0 0 1 0 1 1]
## (5) Resample Yes Rate (%):  A: 50.0 ,  B: 60.0 ,  A-B: -10.0
## (0) Observed Yes Rate (%):  A: 70.0 ,  B: 40.0 ,  A-B: 30.0
```

Code (R & Python) - Perm. Test - Many Permutations

```r
# ═══ R ═══

set.seed(1) # for reproducible results
bag = c(rep(1, t_yes), rep(0, t_no))  # S1: All results bag

# Step 6: Repeat steps 2 to 5 a "large" number of times (p)
p = 100              # number of permutations
perm_res = rep(0, p) # vector for permutation results
for (i in 1:p) {
  bag = sample(bag)         # S2: Shuffle the bag
  a_rs = bag[1:a]           # S3: Random sample A
  b_rs = bag[(a+1):(a+b)]   # S4: Random sample B
  # Step 5: Compute the test statistic
  perm_res[i] = 100*sum(a_rs)/a - 100*sum(b_rs)/b
}

# Print results
options(width = 60); print(perm_res)
```

```
##   [1]  10 -10  10 -30 -10 -30  10 -10  10 -30  10  30 -10
##  [14]  10 -10  10 -30  10 -30  10  30 -10 -10 -50 -30 -10
##  [27]  10  10 -10 -10  10  10 -10  10 -10 -50 -30  30 -30
##  [40]  10 -10 -10  30 -50 -10 -10  30  30  10  30 -50 -10
##  [53]  10  10  10 -10  10 -10  50 -30  10 -10  10  50  30
##  [66] -10  10  30 -10 -10 -10  50  50  30  30  10  50 -30
##  [79] -10 -10  30  10 -10  30  10 -10  10  30  10 -30  10
##  [92]  10  10 -10  10 -30  30  30  10 -10
```

```python
# ═══ Python ═══

seed(2)  # for reproducible results
bag = [1]*t_yes + [0]*t_no  # S1: All results bag

# Step 6: Repeat steps 2 to 5 a "large" number of times (p)
p = 100           # number of permutations
perm_res = [0]*p  # list for permutation results
for i in range(p):
  bag = sample(bag, k=len(bag)) # S2: Shuffle the bag
  a_rs = bag[:a]                # S3: Random sample A
  b_rs = bag[a:]                # S4: Random sample B
  # Step 5: Compute the test statistic
  perm_res[i] = 100*sum(a_rs)/a - 100*sum(b_rs)/b

# Print results
n = 15
for i in range(0, len(perm_res), n):
  print(str(perm_res[i:i+n]).replace(",","").replace(".0",""))
```

```
## [-10 10 -10 10 -30 -10 -10 -10 -10 30 -30 10 10 10 -50]
## [-30 10 30 10 -30 10 10 -10 -10 50 -10 -30 -30 10 -10]
## [-30 10 10 -10 -30 30 10 -10 30 -10 -10 -10 10 -30 30]
## [-50 -30 -10 -10 30 30 30 -10 10 -10 -10 10 -30 10 10]
## [10 -10 -30 -10 30 -30 -10 10 10 10 -10 30 50 10 10]
## [10 10 -30 -10 -30 30 10 -10 -10 -10 -10 30 -10 -10 10]
## [10 -30 30 10 30 30 -30 10 10 -30]
```

Code (R & Python) - Perm. Test - Hypothesis Testing

```r
# === R ===

table(perm_res)
```

```
## perm_res
## -50 -30 -10  10  30  50
##   4  12  30  33  16   5
```

```r
# Two-way hypothesis test (Null: A = B, Alternative: A != B)
extreme_count = sum(abs(perm_res) >= abs(ab_yes_pc))

# One-way hypothesis test (Null: A <= B, Alternative: A > B)
pos_extreme_count = sum(perm_res >= ab_yes_pc)

cat("Number of permutations:", p,
    "\nTwo-way: Extreme count          :", extreme_count,
    "\nTwo-way: Extreme ratio (p-value):", extreme_count / p,
    "\nOne-way: Extreme count          :", pos_extreme_count,
    "\nOne-way: Extreme ratio (p-value):", pos_extreme_count / p)
```

```
## Number of permutations: 100
## Two-way: Extreme count          : 37
## Two-way: Extreme ratio (p-value): 0.37
## One-way: Extreme count          : 21
## One-way: Extreme ratio (p-value): 0.21
```

```python
# === Python ===
perm_res_s = pd.Series(perm_res)
print(pd.pivot_table(perm_res_s.value_counts().reset_index(),
                     values=0, columns="index").to_string(index=False))
```

```
##  -50.0  -30.0  -10.0   10.0   30.0   50.0
##      2     18     31     32     15      2
```

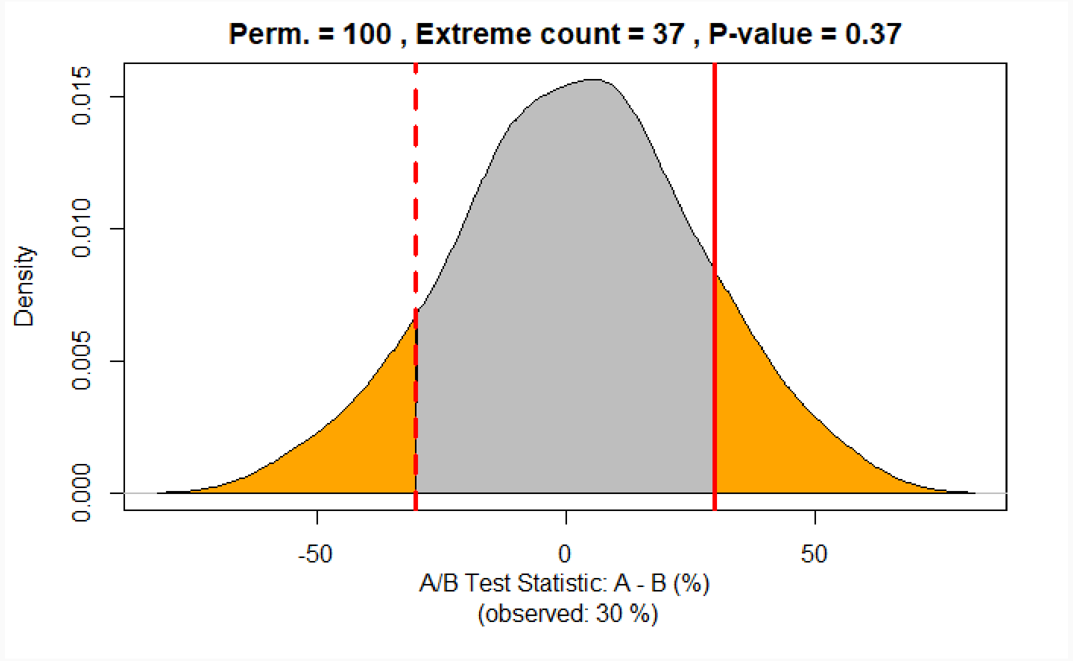
```python
# Two-way hypothesis test (Null: A = B, Alternative: A != B)
extreme_count = sum(perm_res_s.abs() >= abs(ab_yes_pc))

# One-way hypothesis test (Null: A <= B, Alternative: A > B)
pos_extreme_count = sum(perm_res_s >= ab_yes_pc)

print("Number of permutations:", p,
      "\nTwo-way: Extreme count          :", extreme_count,
      "\nTwo-way: Extreme ratio (p-value):", extreme_count / p,
      "\nOne-way: Extreme count          :", pos_extreme_count,
      "\nOne-way: Extreme ratio (p-value):", pos_extreme_count / p)
```

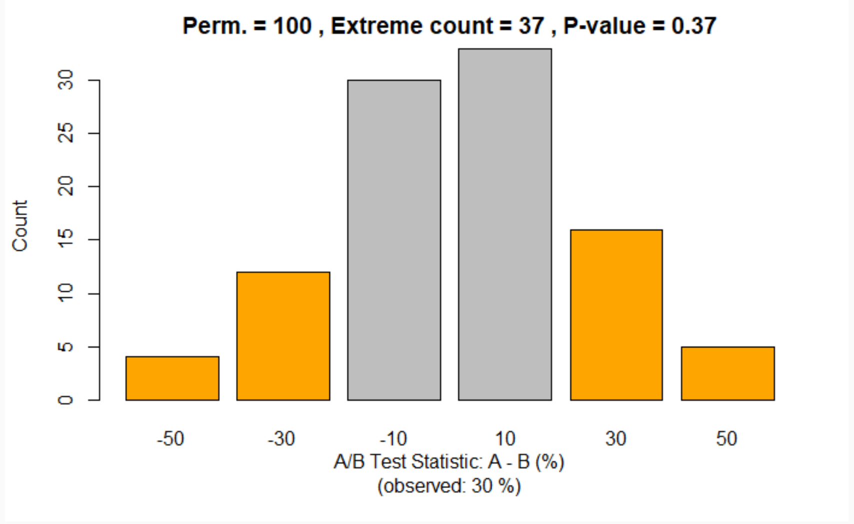
```
## Number of permutations: 100
## Two-way: Extreme count          : 37
## Two-way: Extreme ratio (p-value): 0.37
## One-way: Extreme count          : 17
## One-way: Extreme ratio (p-value): 0.17
```

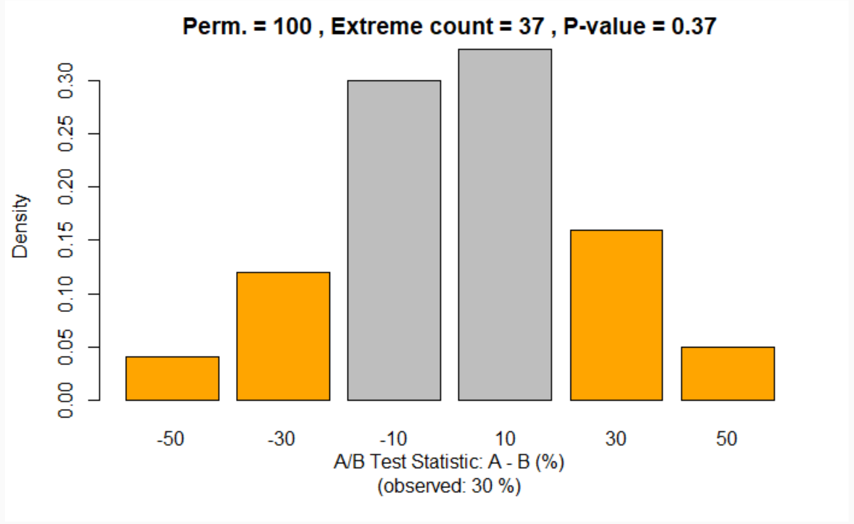
Code/Figs (R) - Perm. Test - Bar & Density Plots

```
table(perm_res)
```

```
## perm_res
## -50 -30 -10  10  30  50
##   4  12  30  33  16   5
```



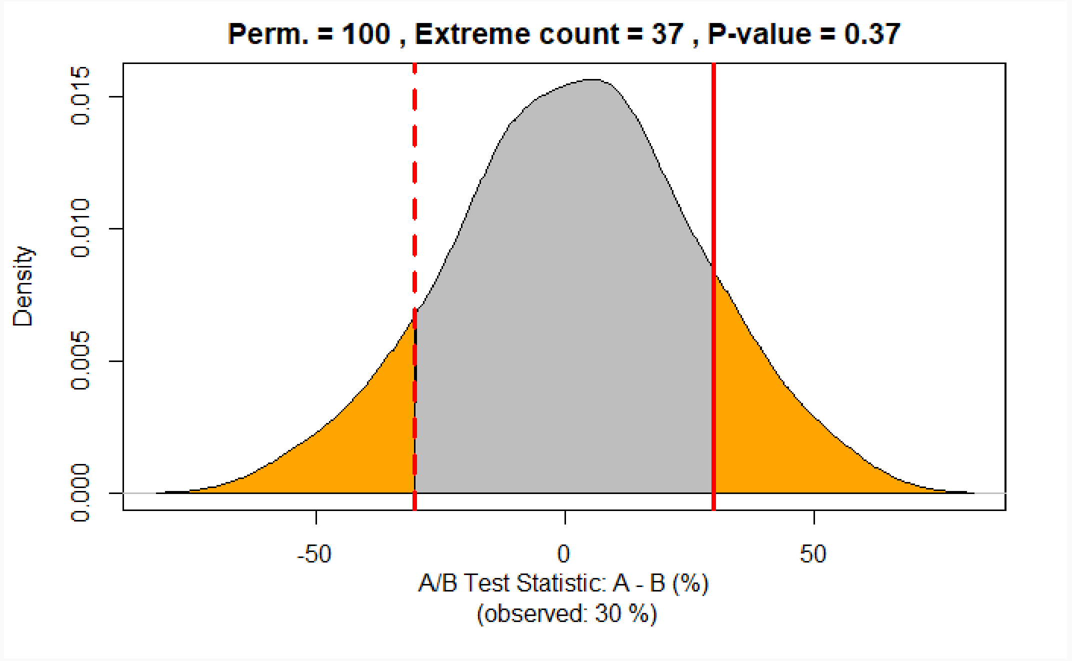


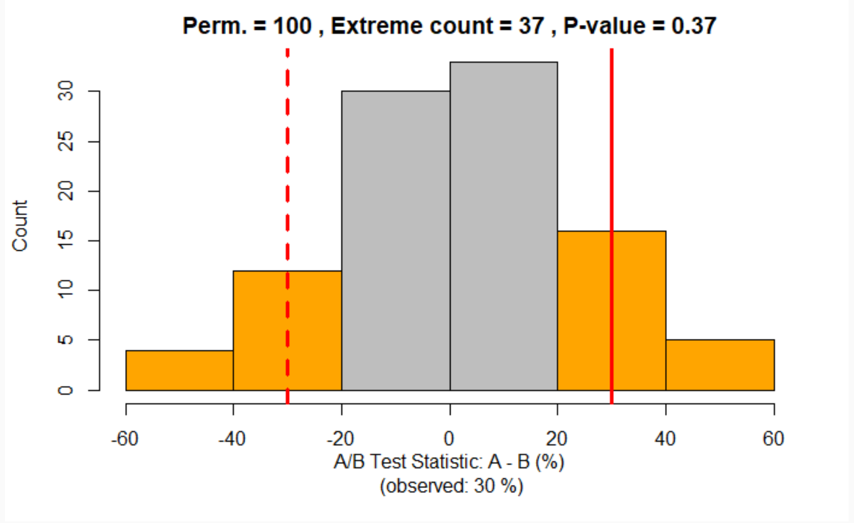


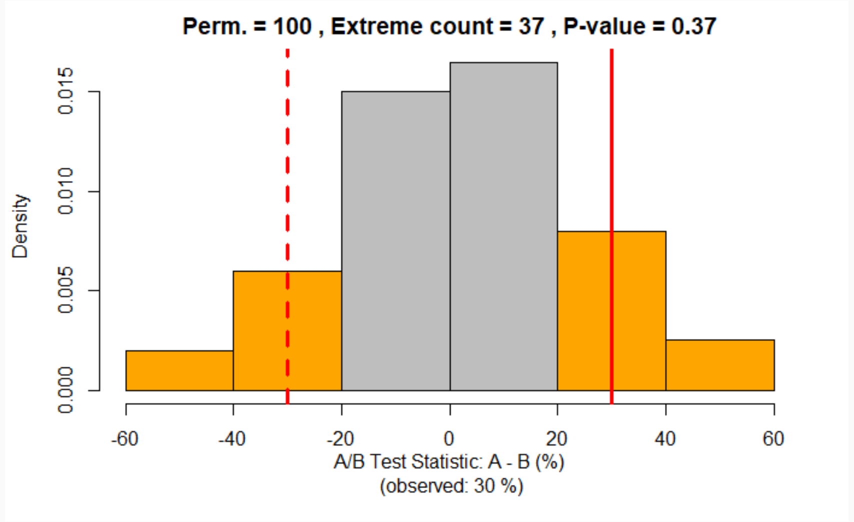
Code/Figs (R) - Perm. Test - Histogram & Density Plots

```r
table(perm_res)
```

```
## perm_res
## -50 -30 -10  10  30  50
##   4  12  30  33  16   5
```







Code (R) - General Solution

The previous code has been consolidated into functions in the `functions.R` script.

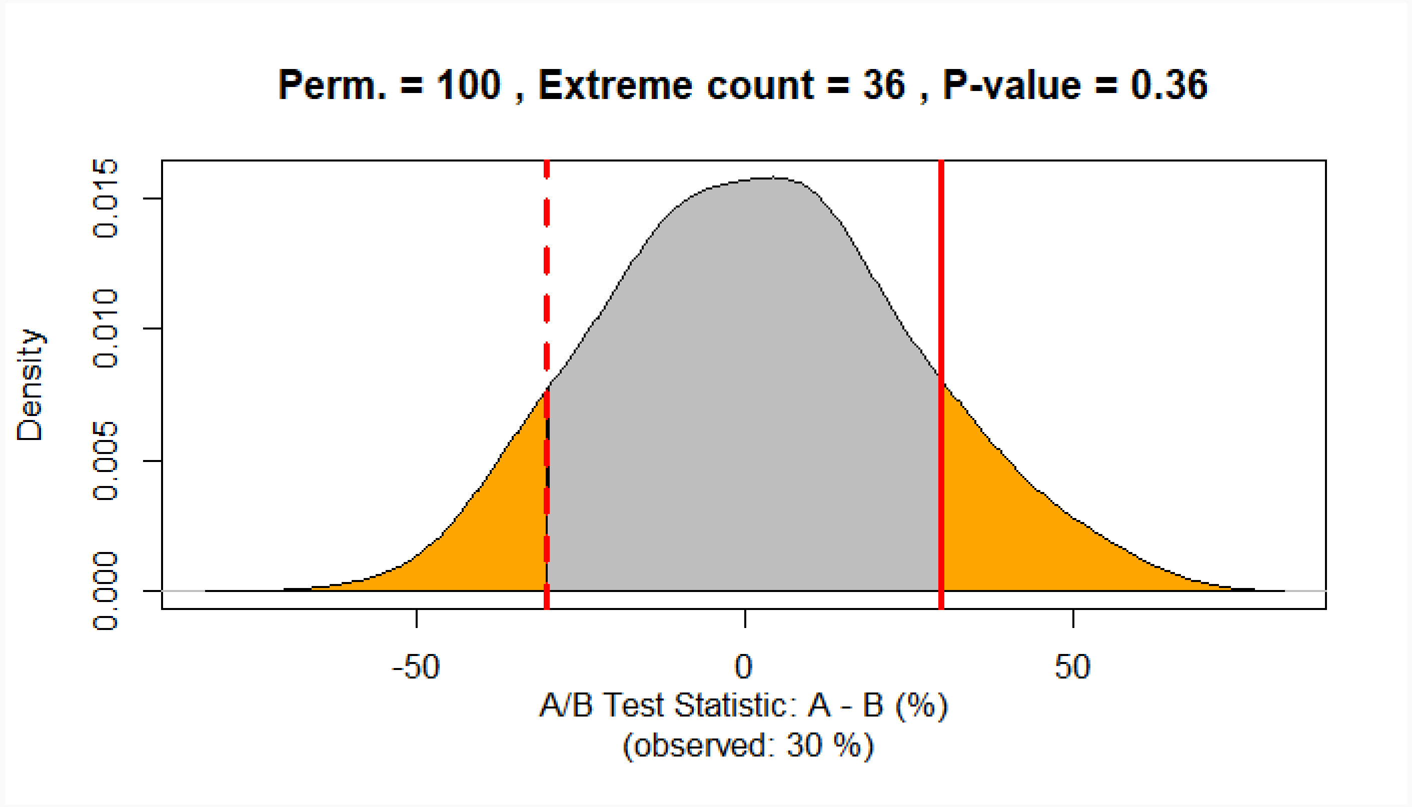
- The function `ab_permutation_test` can be used to perform the whole process of a Permutation Test for a given A/B Test (examples in the next slides).

```r
source("functions.R")
```

Code (R) - General Solution - Didactic Example

```r
ab_permutation_test(a_all=10, b_all=10, a_yes=7, b_yes=4, n_p=100, verbose=TRUE, random_seed=1)
```

```
## Observed Yes Rate (%):  A: 70 , B: 40 , A-B: 30
```

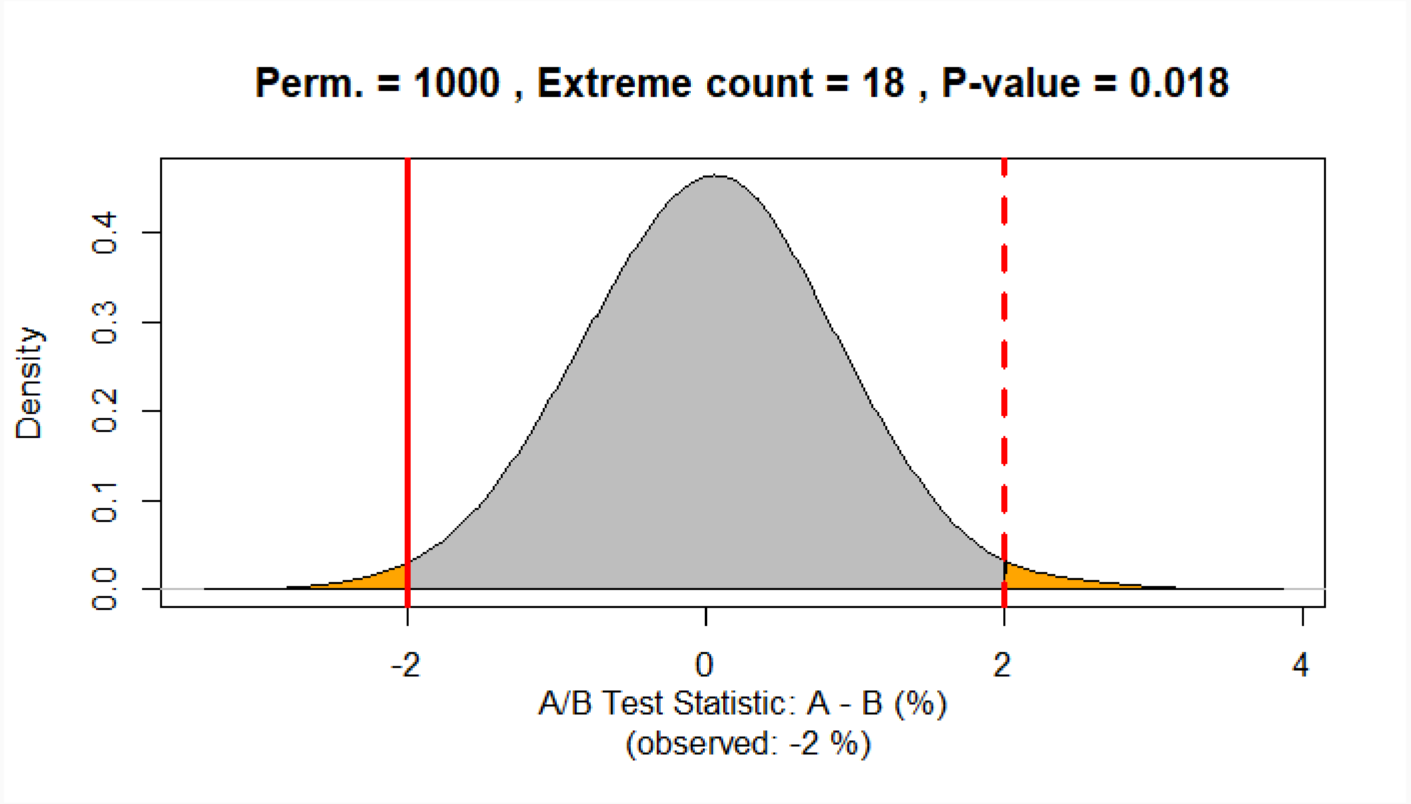


```
## [1] 0.36
```

Code (R) - General Solution - Other Example

```r
ab_permutation_test(a_all=1000, b_all=2000, a_yes=30, b_yes=100, n_p=1000, verbose=TRUE, random_seed=1)
```

```
## Observed Yes Rate (%):  A: 3 , B: 5 , A-B: -2
```



```
## [1] 0.018
```

Code (R) - General Solution - R Shiny App

You can use the R Shiny App defined in the `app.R` file to interactively explore A/B Testing with Permutation Test (i.e., the results of the `ab_permutation_test` function).

You can launch the app as usual in RStudio: open the `app.R` file in RStudio and press the `Run App` button.

Thanks!

Q & A

Ready to learn more?

Thank you all for joining this session. We hope you found it useful.

If you are interested in learning more with us, check out our offerings at the **NYC Data Science Academy**.

- **Bootcamps**
  - **Data Science Bootcamp**: In-person/Remote live instruction or Online instruction
  - **Data Analytics Bootcamp**: Online Instruction
- **Professional Development** programs
  - **Individual Courses**
  - **Course Bundles**

Applying is free!

You can also email admissions@nycdatascience.com for information about any of our programs.

Extra - p-value

Quotes from the American Statistical Association (ASA) Statement on Statistical Significance and P-Values

What is a p-Value?

> Informally, a p-value is the probability under a specified statistical model that a statistical summary of the data (e.g., the sample mean difference between two compared groups) would be equal to or more extreme than its observed value.

Principles

> 1. P-values can indicate how incompatible the data are with a specified statistical model.
> 2. P-values do not measure the probability that the studied hypothesis is true, or the probability that the data were produced by random chance alone.
> 3. Scientific conclusions and business or policy decisions should not be based only on whether a p-value passes a specific threshold.
> 4. Proper inference requires full reporting and transparency
> 5. A p-value, or statistical significance, does not measure the size of an effect or the importance of a result.
> 6. By itself, a p-value does not provide a good measure of evidence regarding a model or hypothesis.

Extra - Power and Sample Size

Power analysis is done in the initial stage of definition of an A/B test

Common task: determine how big of a **sample size** you will need.

Any of these can be determined once we define all the others:

- **Sample size:** the size of your sample data (number of observations)
- **Effect size:** minimum size of the effect you want to detect in a statistical test (e.g., 10% increase in conversation rates)
- **Power:** probability of detecting a give effect size when the effect is real (**True Positive**).
- **Significance level (alpha):** to be used in the statistical test (probability of a **False Positive**)

Power and Errors:

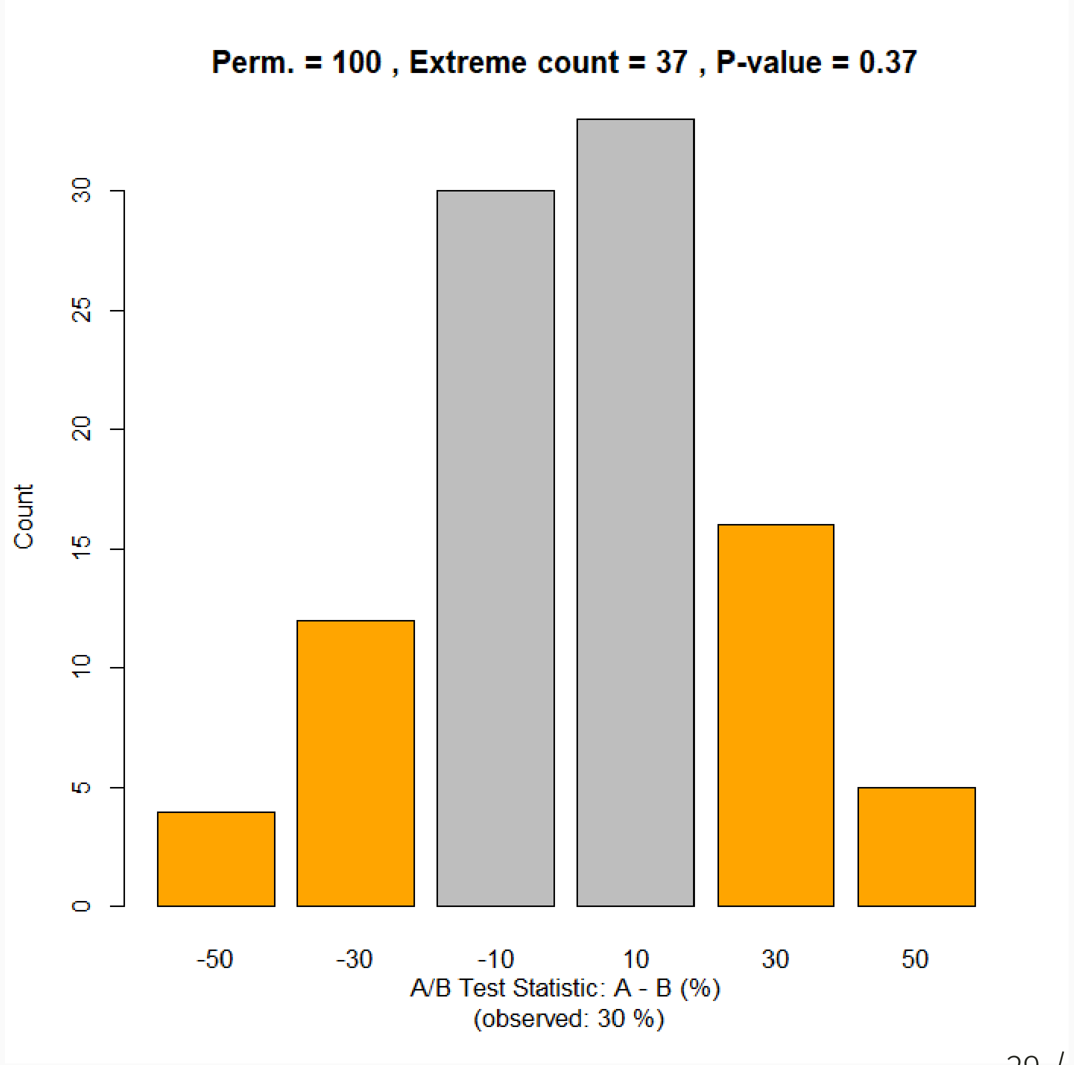
- **Power:** probability of detecting a give effect size when the effect is real (**True Positive**). Probability = $1 - \beta$
- **Type II error (False Negative):** Mistakenly concluding that an effect is due to chance (when it is real). Probability = β (beta)
- **Type I error (False Positive):** Mistakenly concluding that an effect is real (when it is due to chance). Probability = α (alpha)

Code:

- R: `pwr` package: https://rdrr.io/cran/pwr/
- Python: `statsmodels` package: https://www.statsmodels.org/

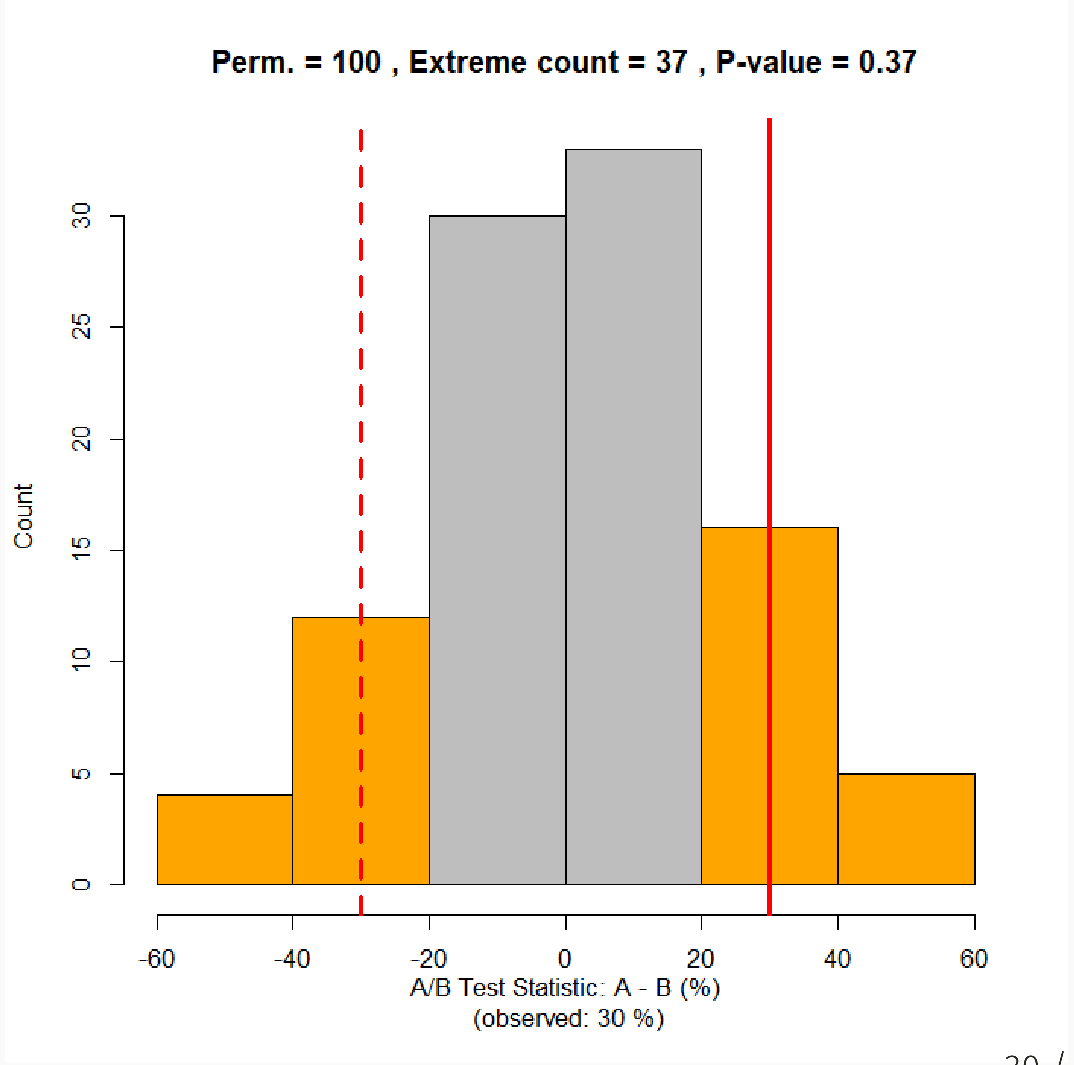
Extra - Bar Plot

```
# ══ R ══

counts = table(perm_res)

x = abs(as.numeric(names(counts)))

colors = ifelse(x < abs(ab_yes_pc), 'gray', 'orange')

title = paste("Perm. =", p,
              ", Extreme count =", extreme_count,
              ", P-value =", extreme_count / p)

x_lab = paste("A/B Test Statistic: A - B (%)\n",
              "(observed:", ab_yes_pc, "%)")

y_lab = "Count"

barplot(counts, col=colors, main=title,
        xlab=x_lab, ylab=y_lab)
```



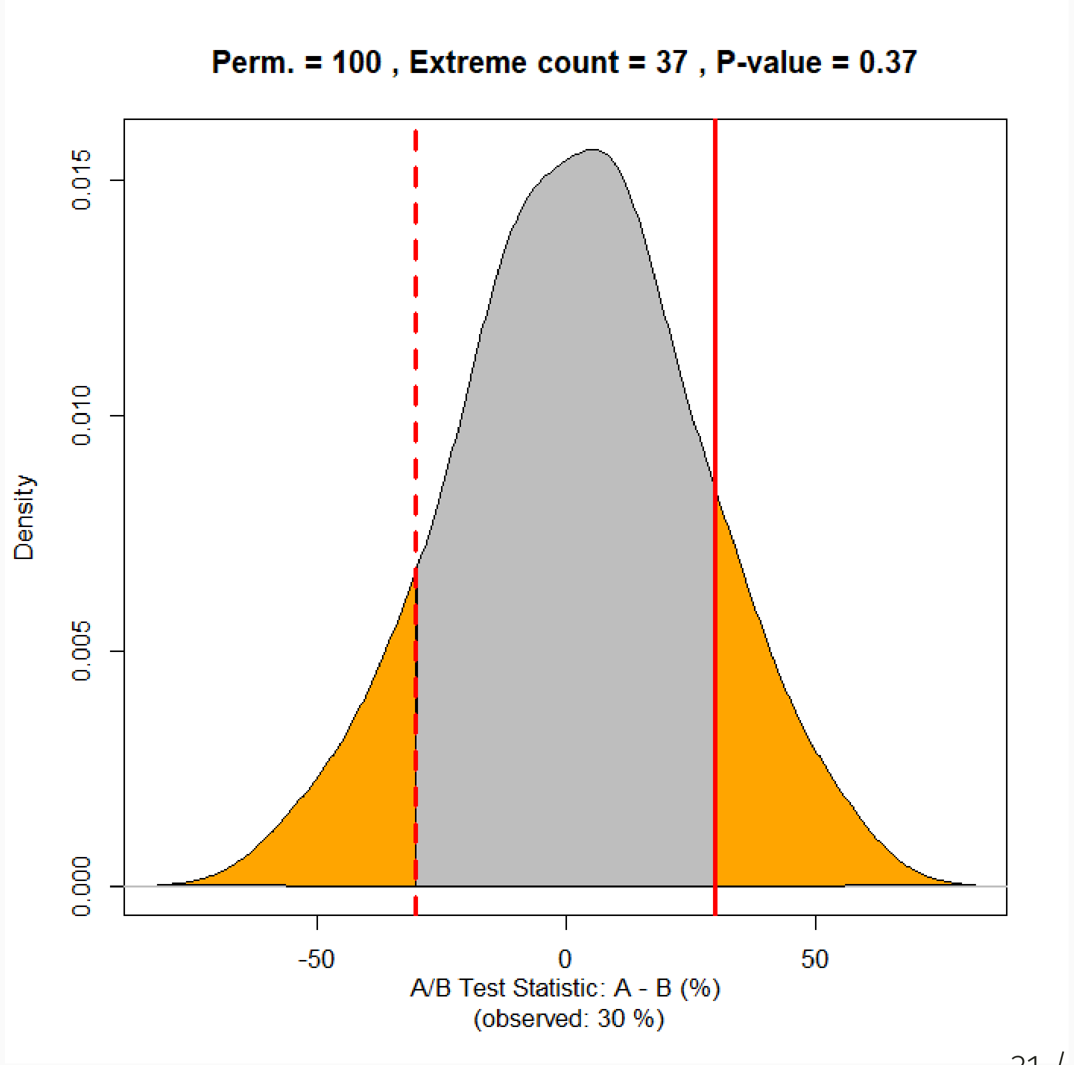
Extra - Histogram Plot

```r
# ═══ R ═══

h = hist(perm_res, breaks = "Scott", plot=FALSE)

b_left = abs(h$breaks[-length(h$breaks)])
b_right = abs(h$breaks[-1])
b_max = ifelse(b_left > b_right, b_left, b_right)

colors = ifelse(b_max < abs(ab_yes_pc), 'gray', 'orange')

title = paste("Perm. =", p,
              ", Extreme count =", extreme_count,
              ", P-value =", extreme_count / p)

x_lab = paste("A/B Test Statistic: A - B (%)\n",
              "(observed:", ab_yes_pc, "%)")

y_lab = "Count"

plot(h, col=colors, main=title, xlab=x_lab, ylab=y_lab)

abline(v = ab_yes_pc, col = 'red', lwd = 3)
abline(v = -ab_yes_pc, col = 'red', lty = 2, lwd = 3)
```



Extra - Density Plot

```
# ══ R ══

title = paste("Perm. =", p,
              ", Extreme count =", extreme_count,
              ", P-value =", extreme_count / p)

x_lab = paste("A/B Test Statistic: A - B (%)\n",
              "(observed:", ab_yes_pc, "%)")

d = density(perm_res, adjust=2)

plot(d, main=title, xlab=x_lab)

xc = abs(ab_yes_pc)
c1 = (d$x < -xc)     # extreme negative values (left)
c2 = (abs(d$x) < xc) # non-extreme values
c3 = (d$x > xc)      # extreme positive values (right)

polygon(c(d$x[c1], -xc), c(d$y[c1], 0), col='orange')
polygon(c(-xc, d$x[c2], xc), c(0, d$y[c2], 0), col='gray')
polygon(c(xc, d$x[c3]), c(0, d$y[c3]), col='orange')

abline(v = ab_yes_pc, col = 'red', lwd = 3)
abline(v = -ab_yes_pc, col = 'red', lty = 2, lwd = 3)
```


Perm. = 100 , Extreme count = 37 , P-value = 0.37
Density
A/B Test Statistic: A - B (%)
(observed: 30 %)