

PSYC 193R:

Data Analysis & Visualization with R

Lecture 7: One-sample t-test (analysis)

Jonas S.-H. Lau [silau -et- ucsd.edu]
psyc193r.ucsd.edu



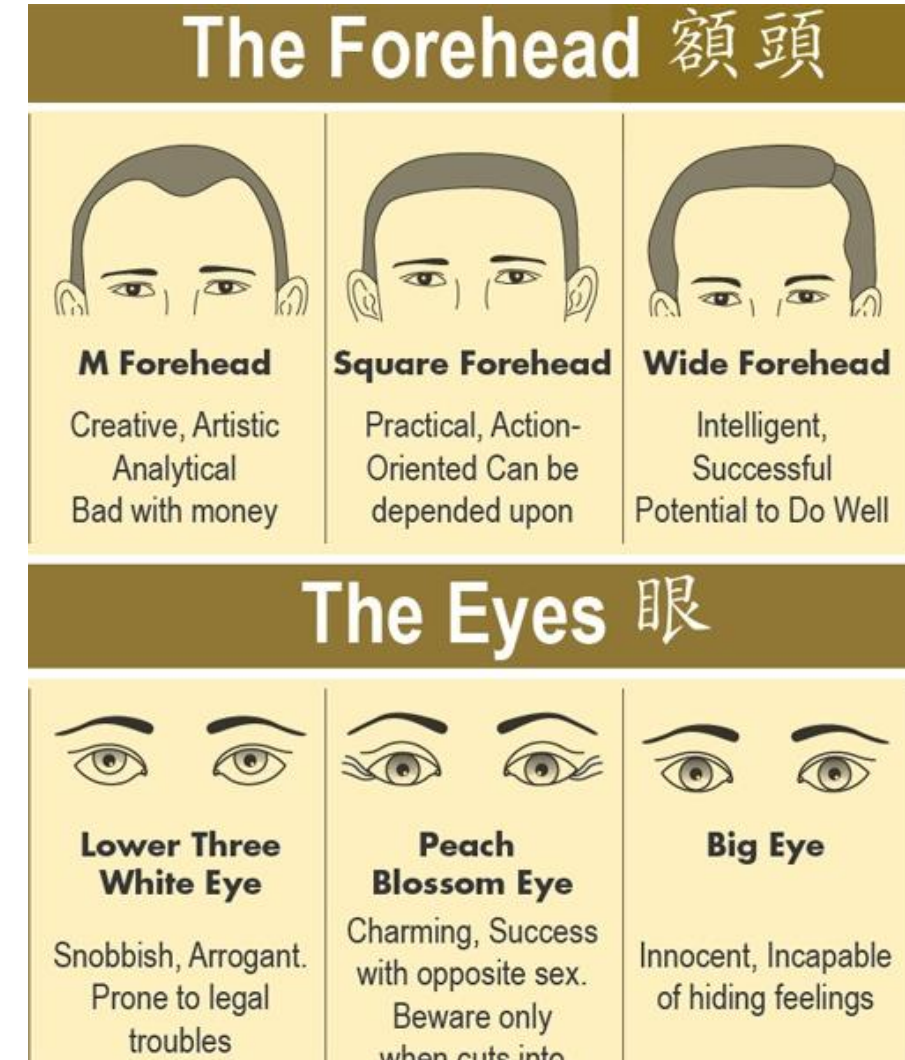
The Art of Face Reading

Jonas wants to know if the traditional art of face reading has a scientific basis.

Needless to say, there will be enormous benefits if we can read people's preferences reliably.

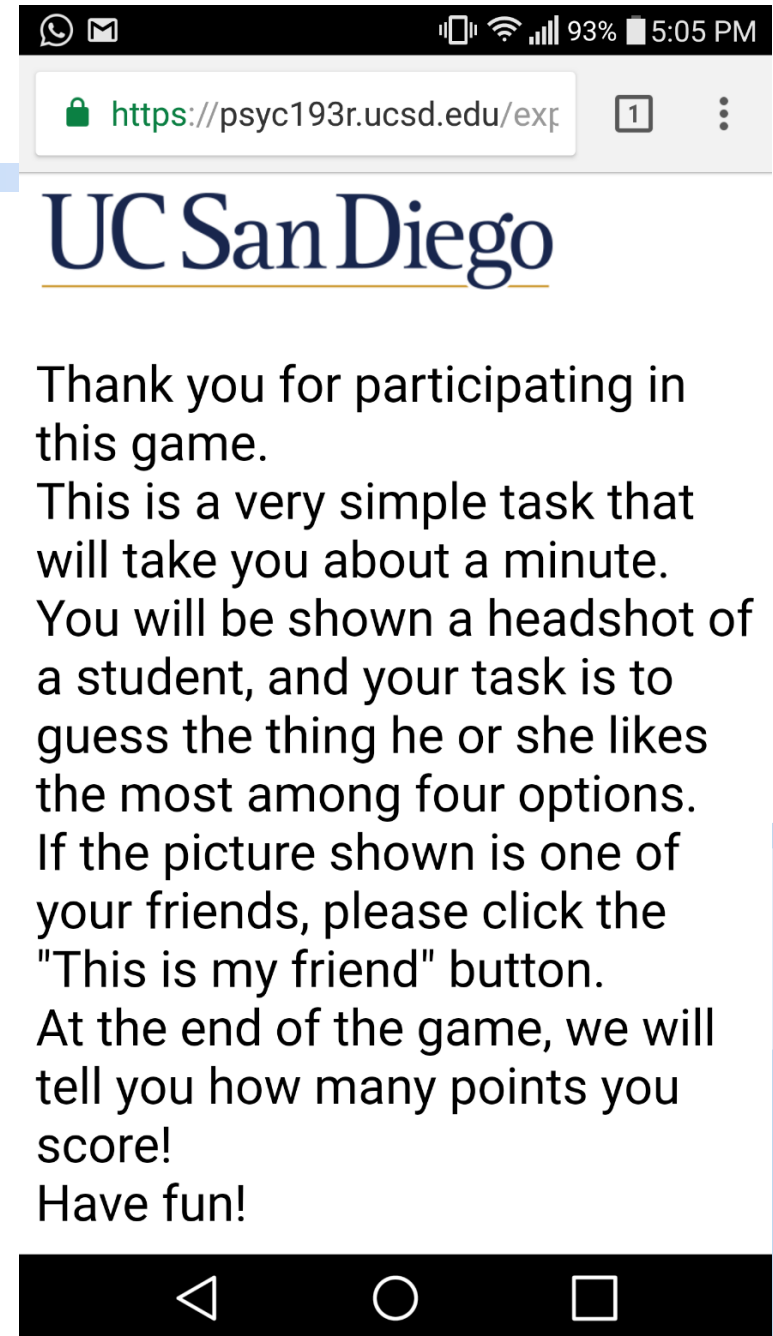
(e.g., first date, job interview, business meetings, ...)

Design a study to scientifically examine the issue.



Face Reading Experiment

- n participants performed 10 – 13 face-reading trials.
- In each trial, a headshot was shown together with four choices. Participants had to guess which the person's preference was out of the four options.
 - Question order was randomized
 - Question choices were randomized



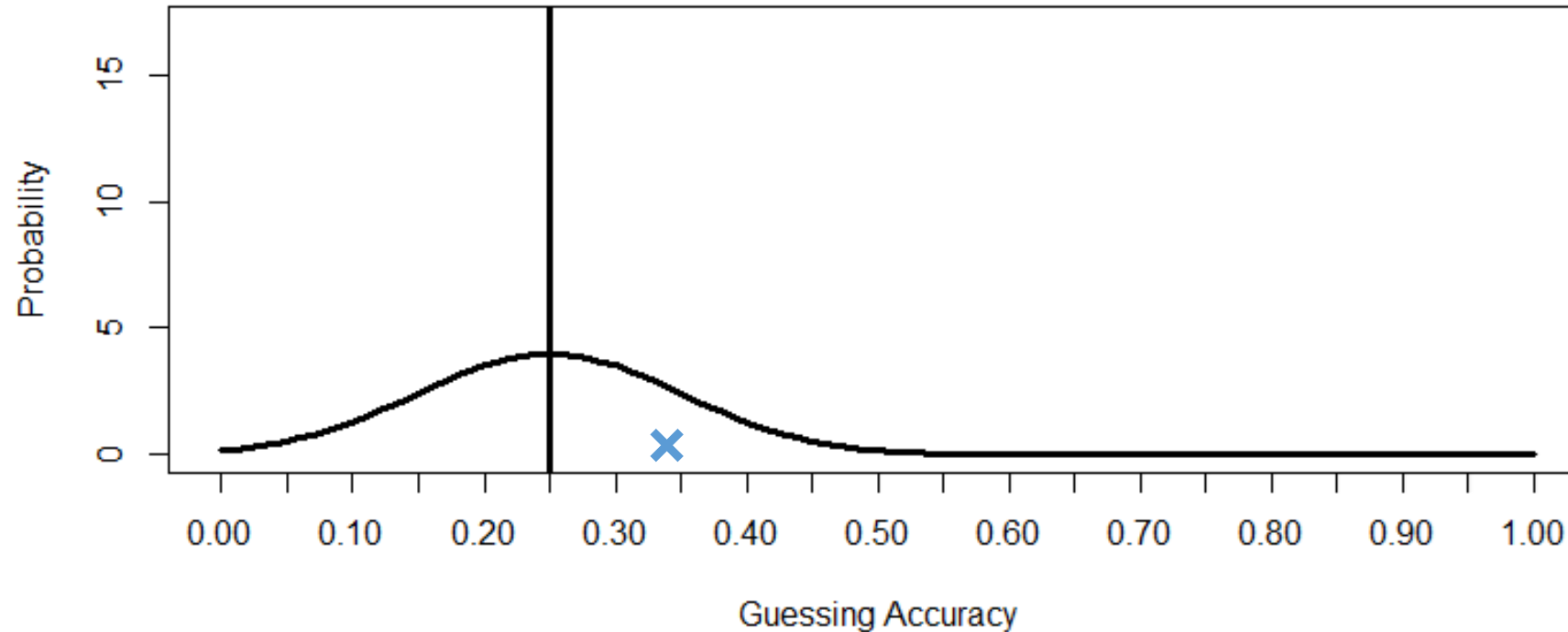
The Design

- What was the research question?
- What was the target population?
- What were the IV(s) and DV(s)?
- How were the IV(s) and DV(s) operationalized?
(also, scale of measurement?)
- Was it a between / within-subjects design?
- What are the expected results?
If the manipulation does not work:
If the manipulation works:



The basic structure

Baseline population distribution (no face / random guessing)



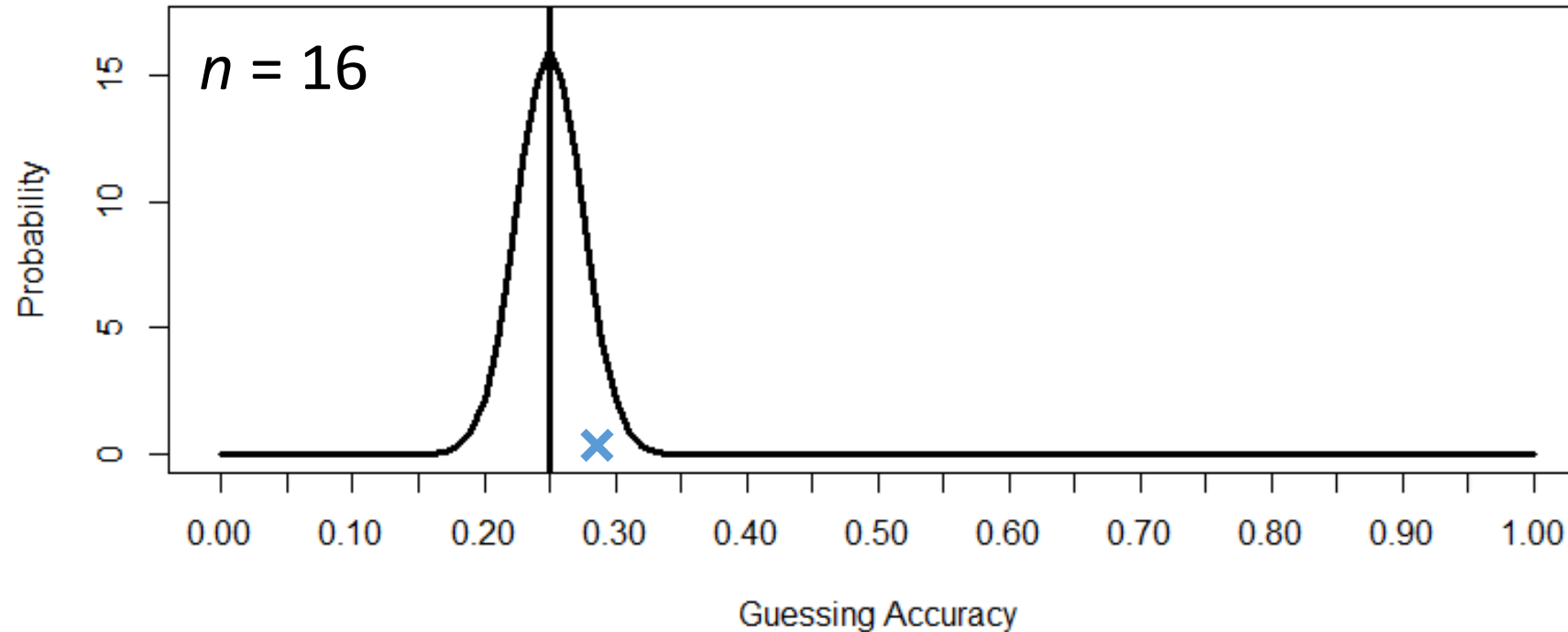
Sample Representing
the treated population
(guess with face)

$n = 16$



The basic structure

Baseline sampling distribution (no face / random guessing)



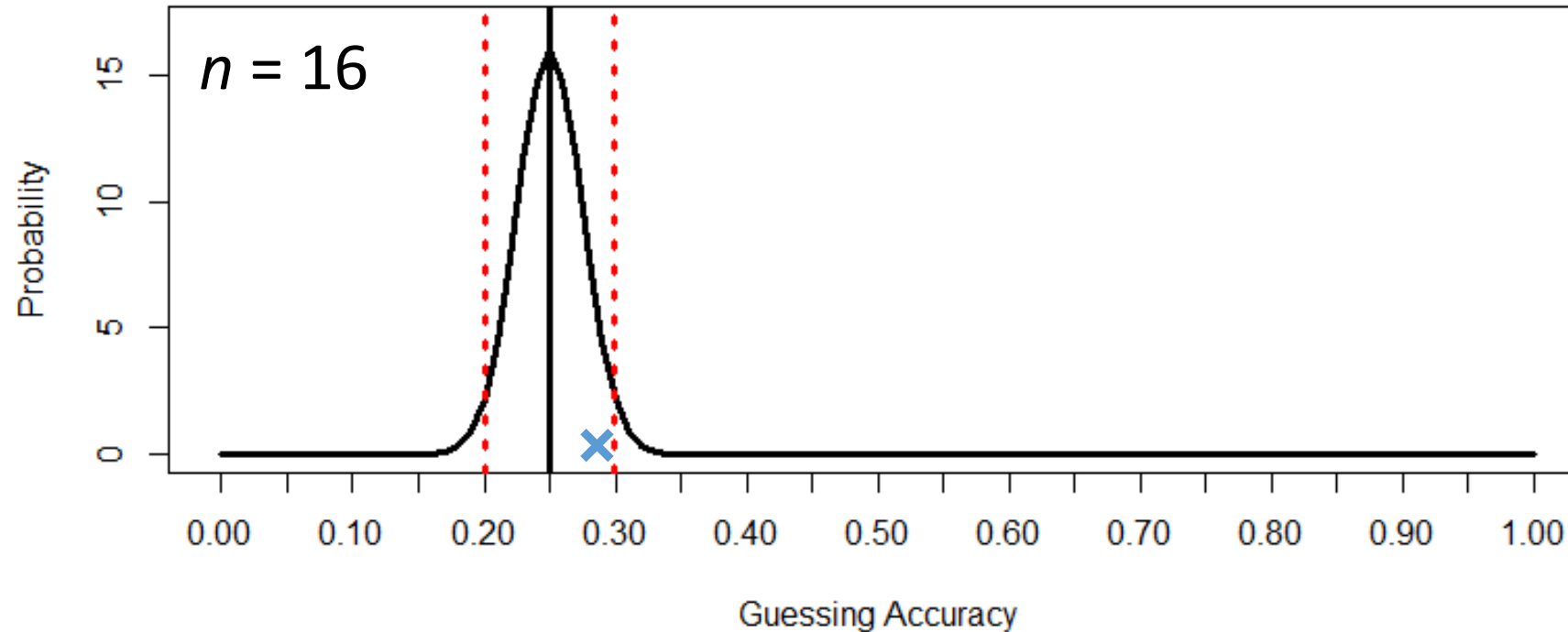
Sample Representing
the treated population
(guess with face)

$n = 16$



The basic structure

Baseline sampling distribution (no face / random guessing)



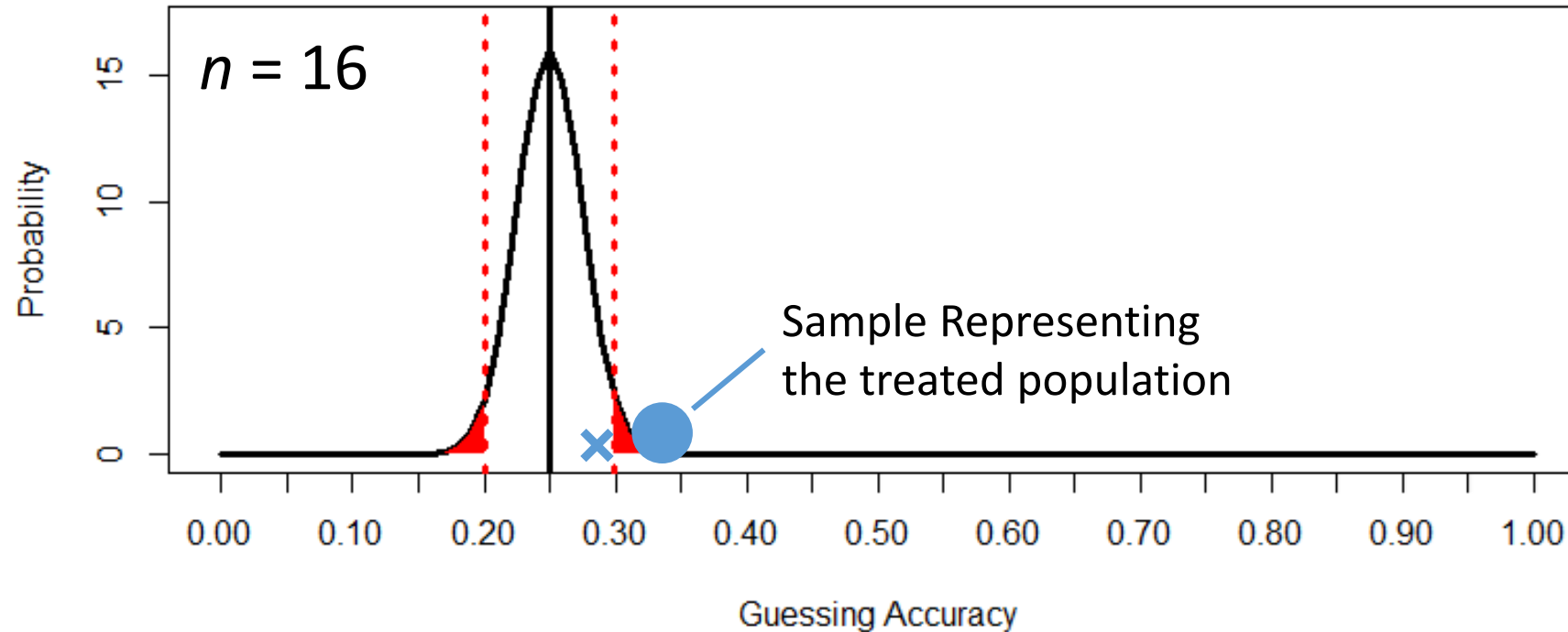
Sample Representing
the treated population
(guess with face)

$n = 16$



The basic structure

Baseline sampling distribution (no face / random guessing)



Z-test

Information needed

- From the baseline population:

$\mu_{baseline}$

σ

kurtosis: normality assumption

- From the sample:

Sample mean $M_{treatment}$

Sample Size n

Actual Information we have

- From the baseline population:

$\mu_{baseline}$ (theoretical)

- From the sample:

From the individual scores...

Sample mean $M_{treatment}$

Sample Size n



Z-test

Information needed

- From the baseline population:

μ_{baseline}

σ

kurtosis

- From the sample:

Sample mean M_{treatm}

Sample Size n

$n \geq 30$

Sampling
distribution
approaches normal

Z distribution

Actual Information we have

- From the baseline population:

μ_{baseline} (theoretical)

- From the sample:

From the individual scores...

Sample mean $M_{\text{treatment}}$

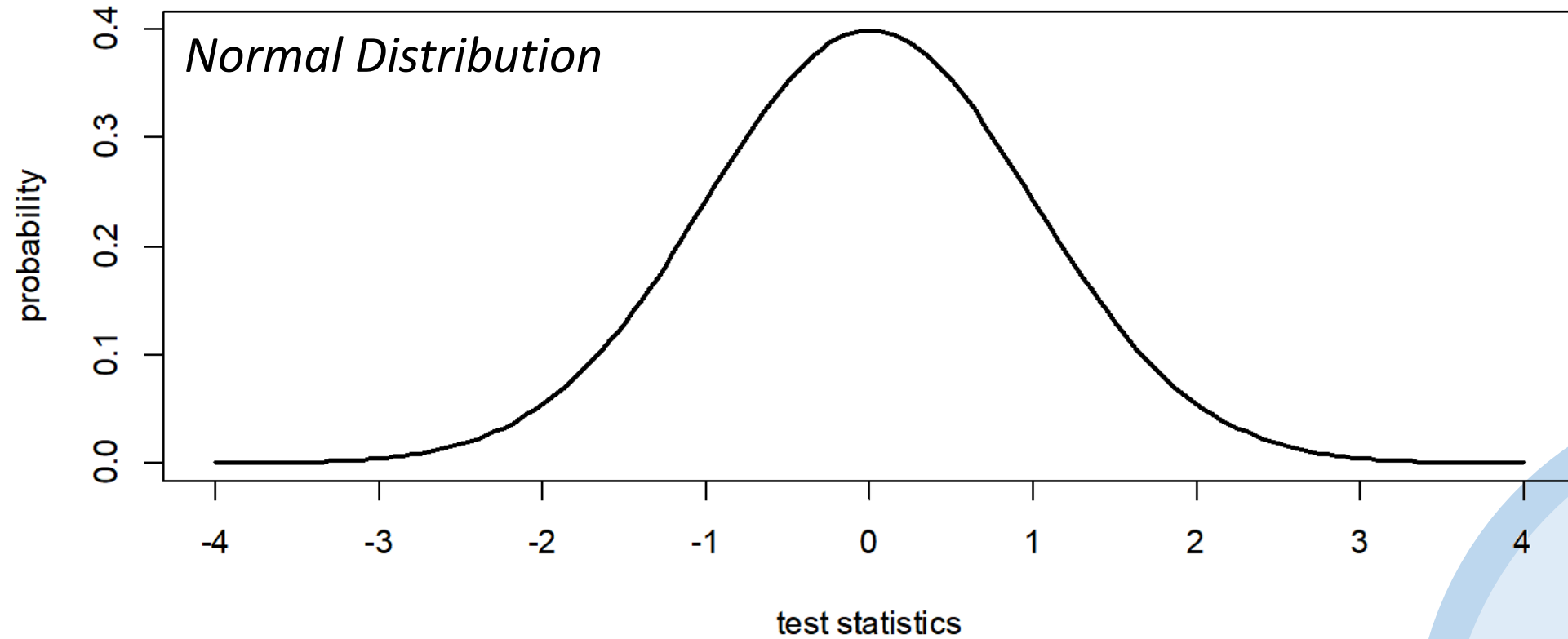
Sample Size n

degrees of freedom $(n - 1)$

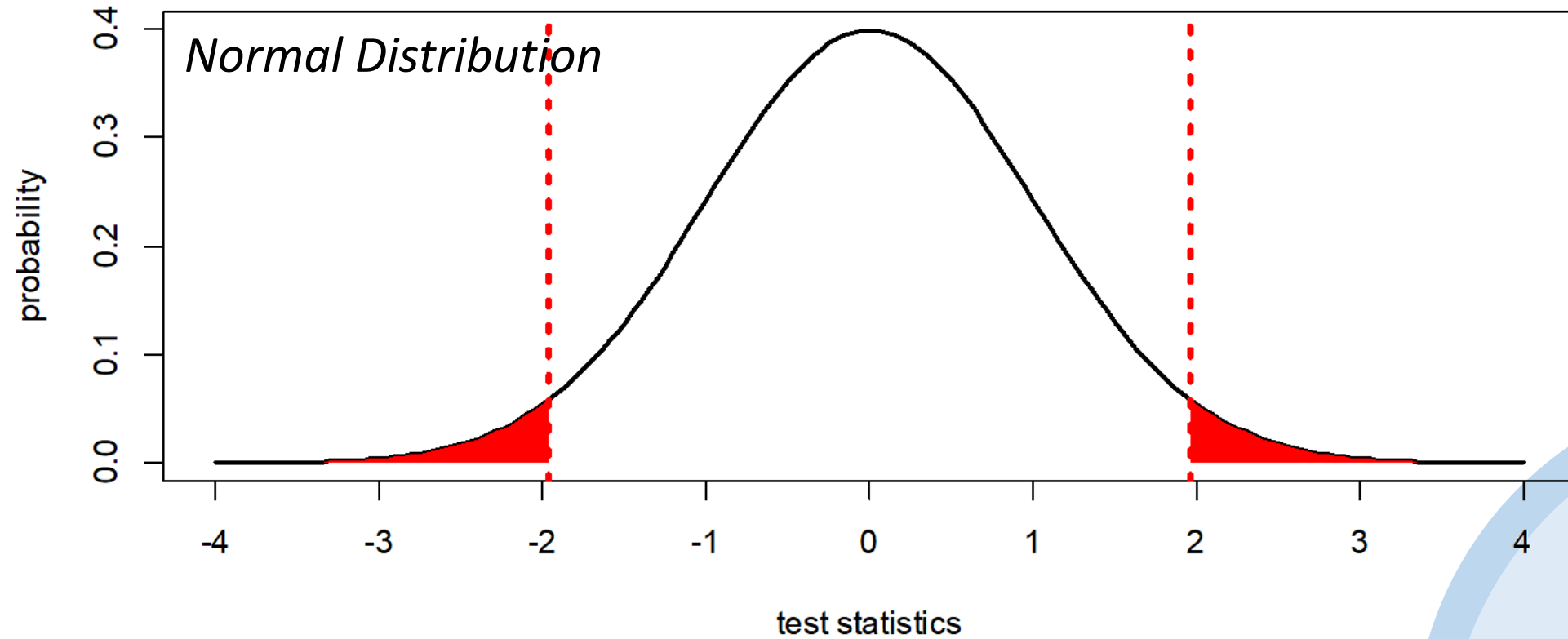
t distribution

$$S = \sqrt{\frac{\sum (X - M)^2}{n - 1}}$$

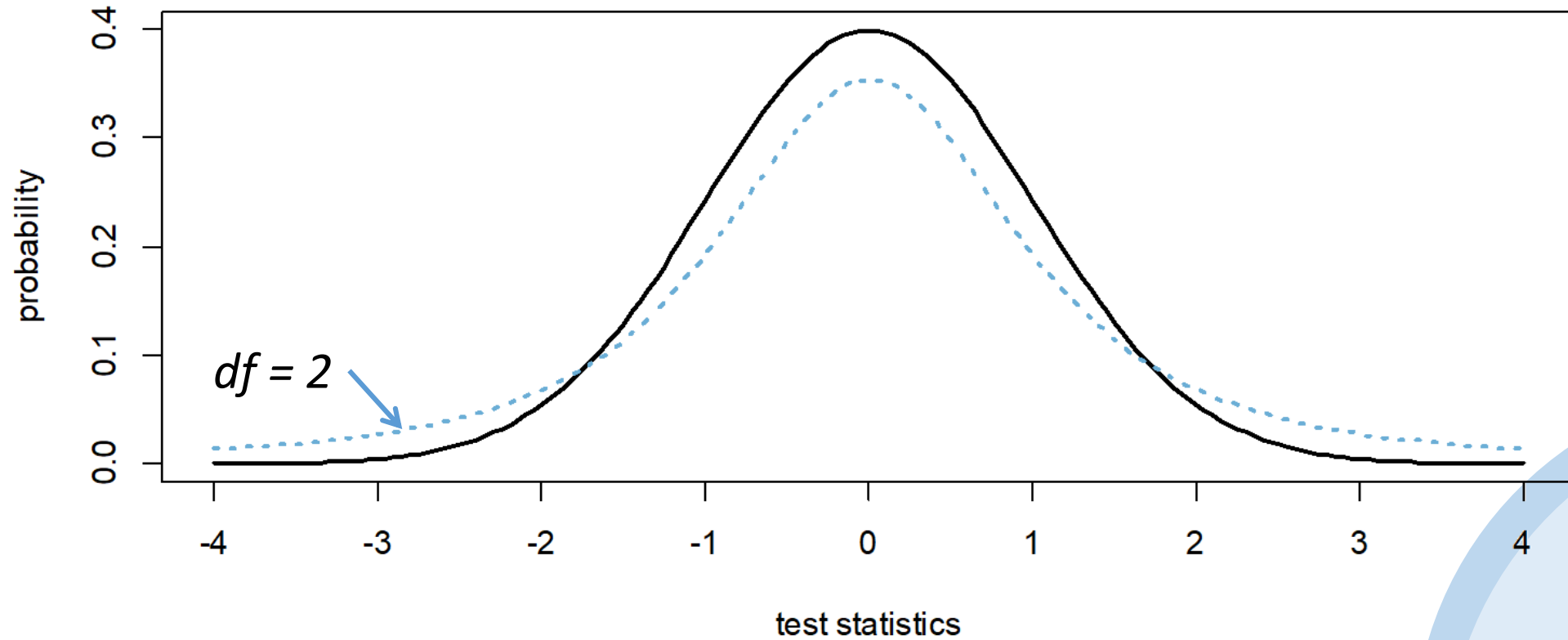
Normal (Z) distribution



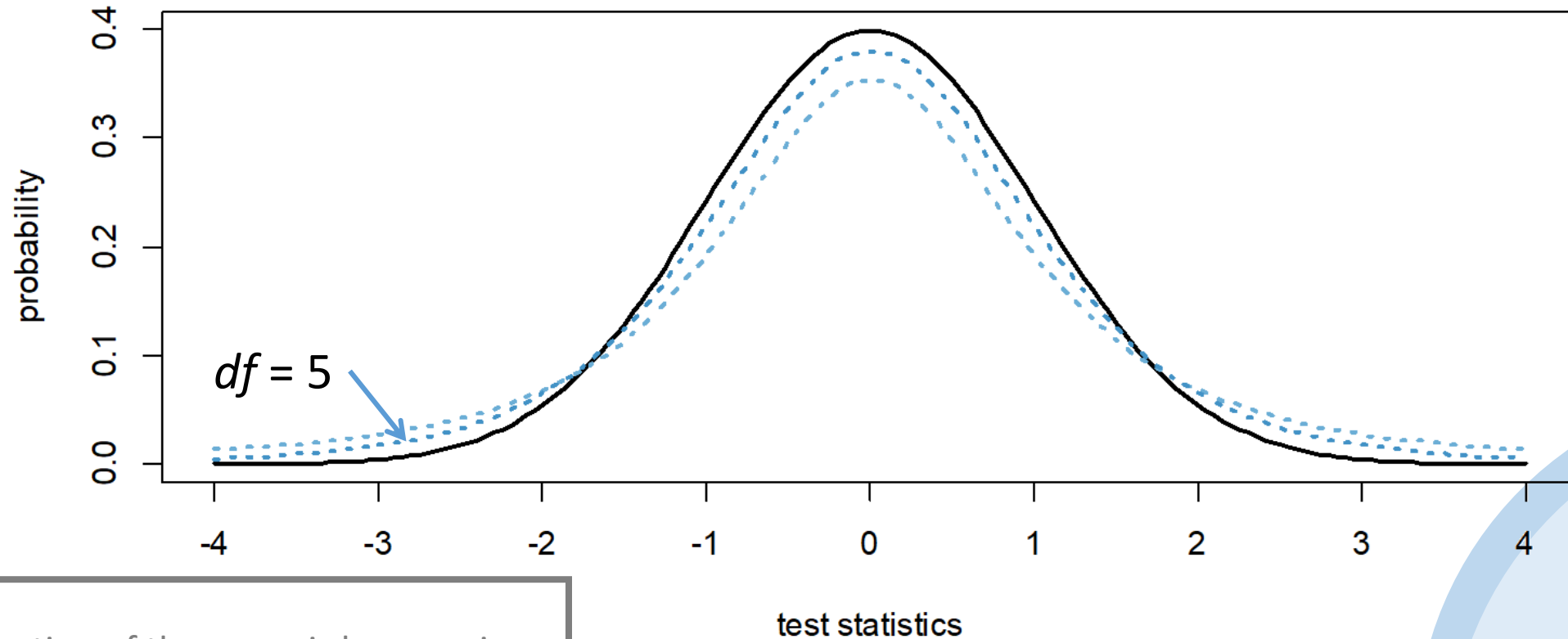
Normal (Z) distribution



t distribution ($df = 2$)

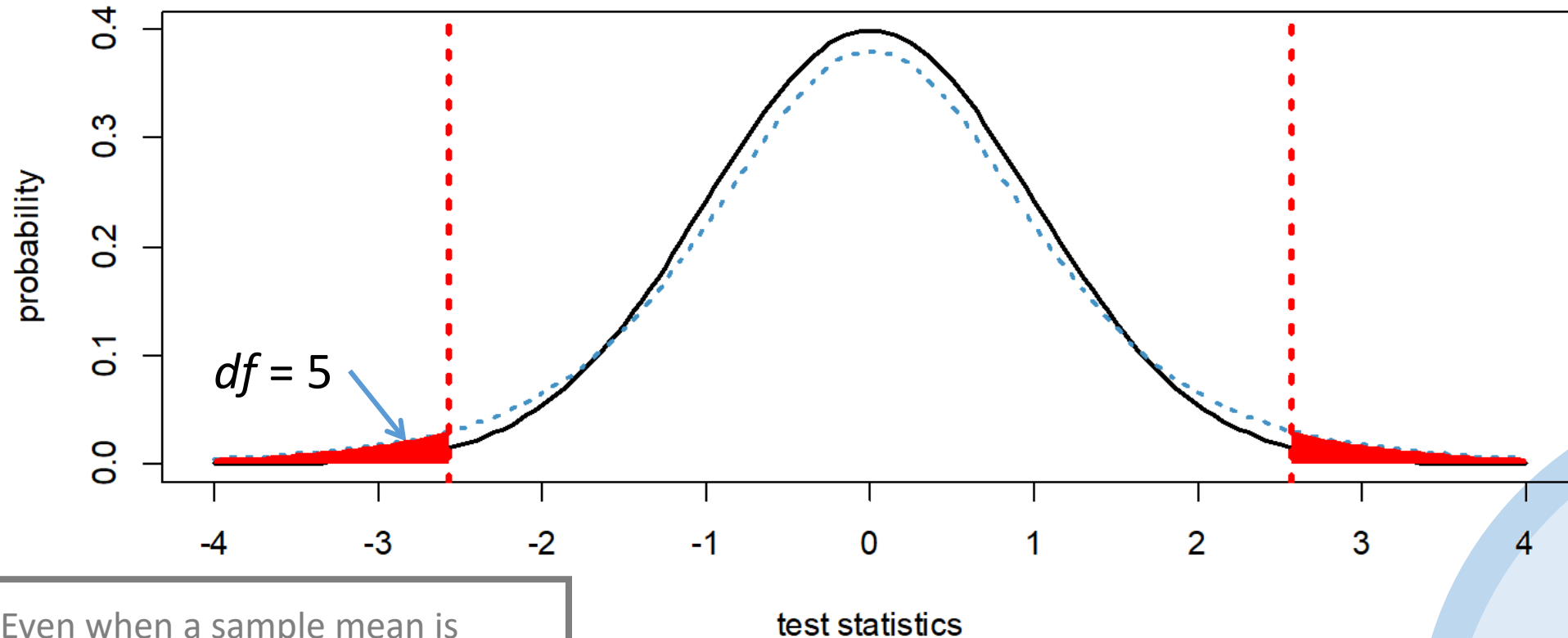


t distribution ($df = 5$)



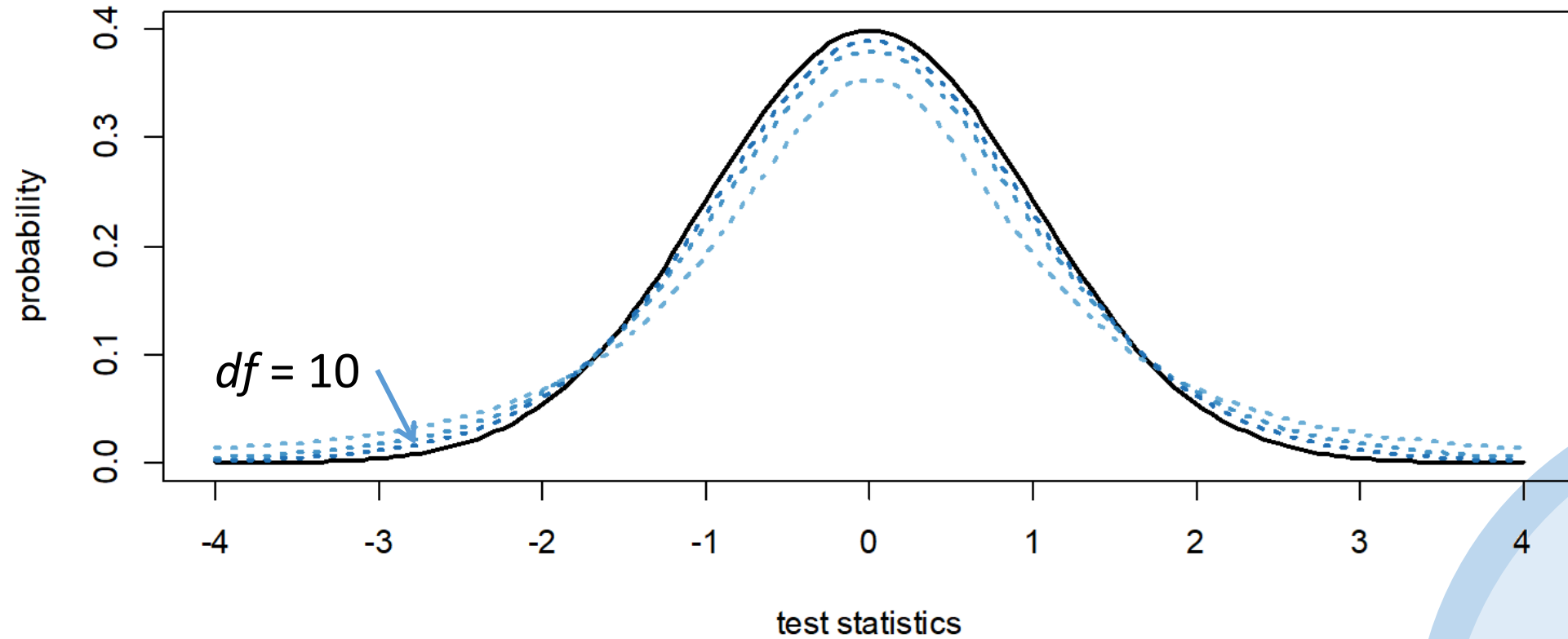
Estimation of the mean is less precise,
compared to the case when you have
the full distribution

t distribution ($df = 5$)



Even when a sample mean is 2.5 SE away from the null mean, you cannot be too confident that they are statistically different ($\alpha = 0.05$)

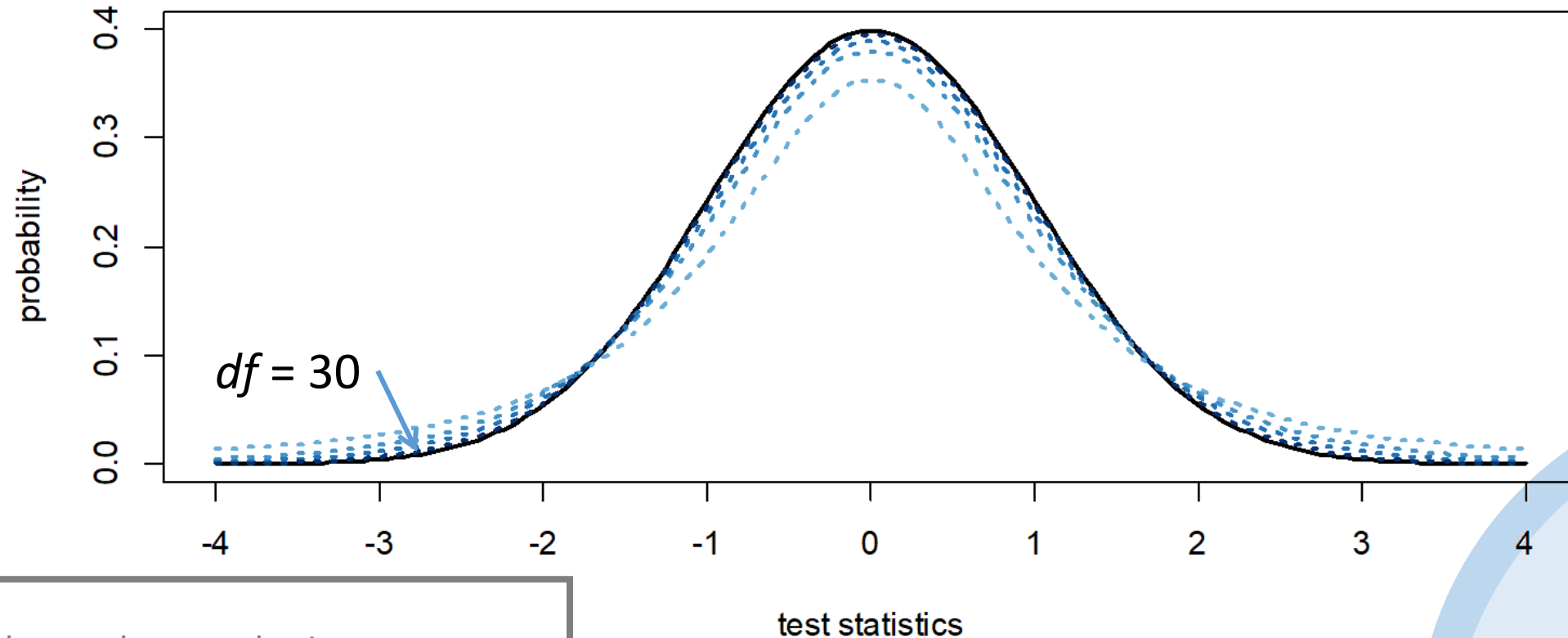
t distribution ($df = 10$)



t distribution ($df = 30$)

At $n = 30$:

- the sampling distribution is approaching normal (regardless of the population distribution)
- the t distribution ($df = 30$) is also approaching normal



The larger the sample size you use to estimate the population distribution, the closer it approaches normality

One-sample Z-test

When to use the test?	When population variance is known
Test Statistic & Formula	$Z_{Sample} = \frac{M_{treatment} - \mu_{baseline}}{\sigma_M}$
degrees of freedom (df)	---
Population Standard Deviation [^]	$\sigma = \sqrt{\frac{\sum (X - \mu)^2}{N}}$
Standard Error [#]	$\sigma_M = \frac{\sigma}{\sqrt{n}}$
Quantile ↑↓ Percentile	Quantile → Percentile: <code>pnorm(q, mean = 0, sd = 1)</code> Percentile → Quantile: <code>qnorm(p, mean = 0, sd = 1)</code>
Confidence Interval	$M_{treatment} \pm Z_{\frac{\alpha}{2}} \times \sigma_M$

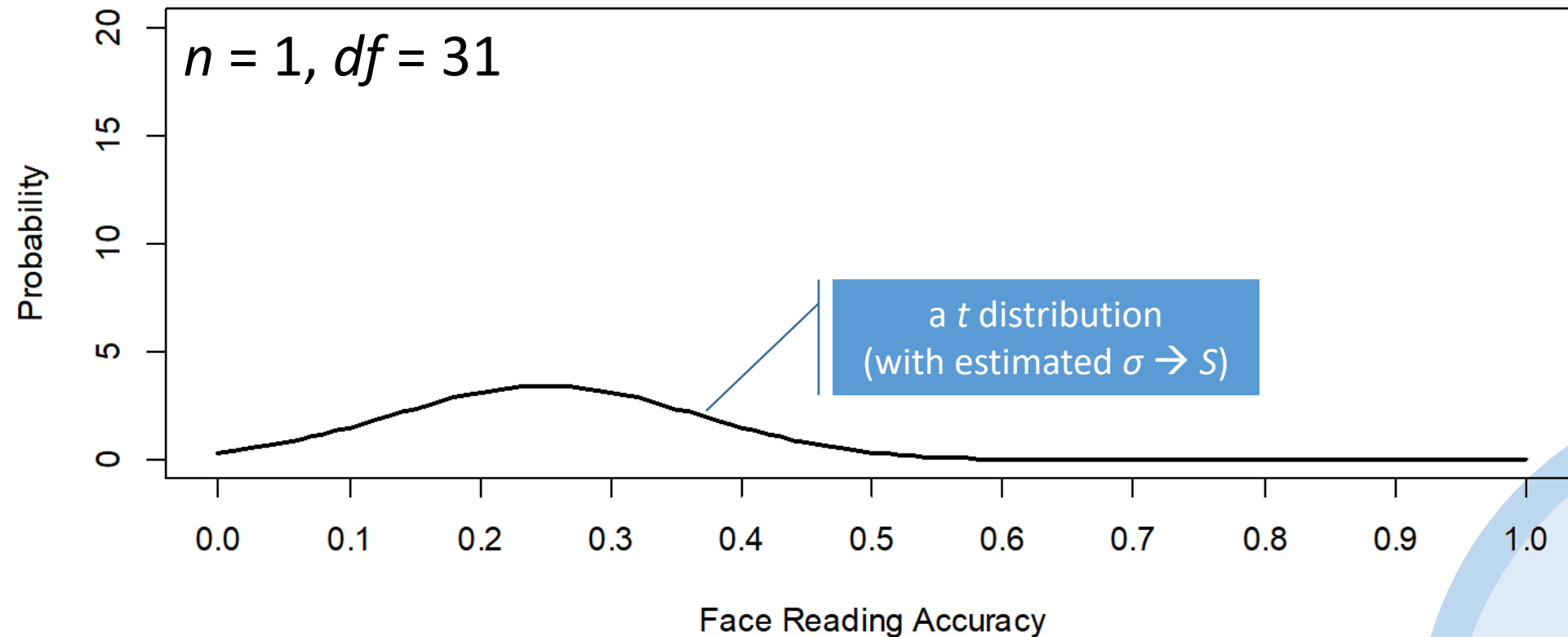
One-sample Student's t-test

When to use the test?	When population variance is unknown
Test Statistic & Formula	$t_{Sample} = \frac{M_{treatment} - \mu_{baseline}}{S_M}$
degrees of freedom (df)	$n - 1$
Population Standard Deviation [^]	Population standard deviation is estimated by: $S = \sqrt{\frac{\sum (X - M)^2}{n - 1}} = \sqrt{\frac{SS}{df}} \quad \text{sd}(X)$
Standard Error [#]	$S_M = \frac{S}{\sqrt{n}}$
Quantile ↑↓ Percentile	Quantile → Percentile: <code>pt(q, df)</code> Percentile → Quantile: <code>qt(p, df)</code>
Confidence Interval	$M_{treatment} \pm t_{\frac{\alpha}{2}, df} \times S_M$

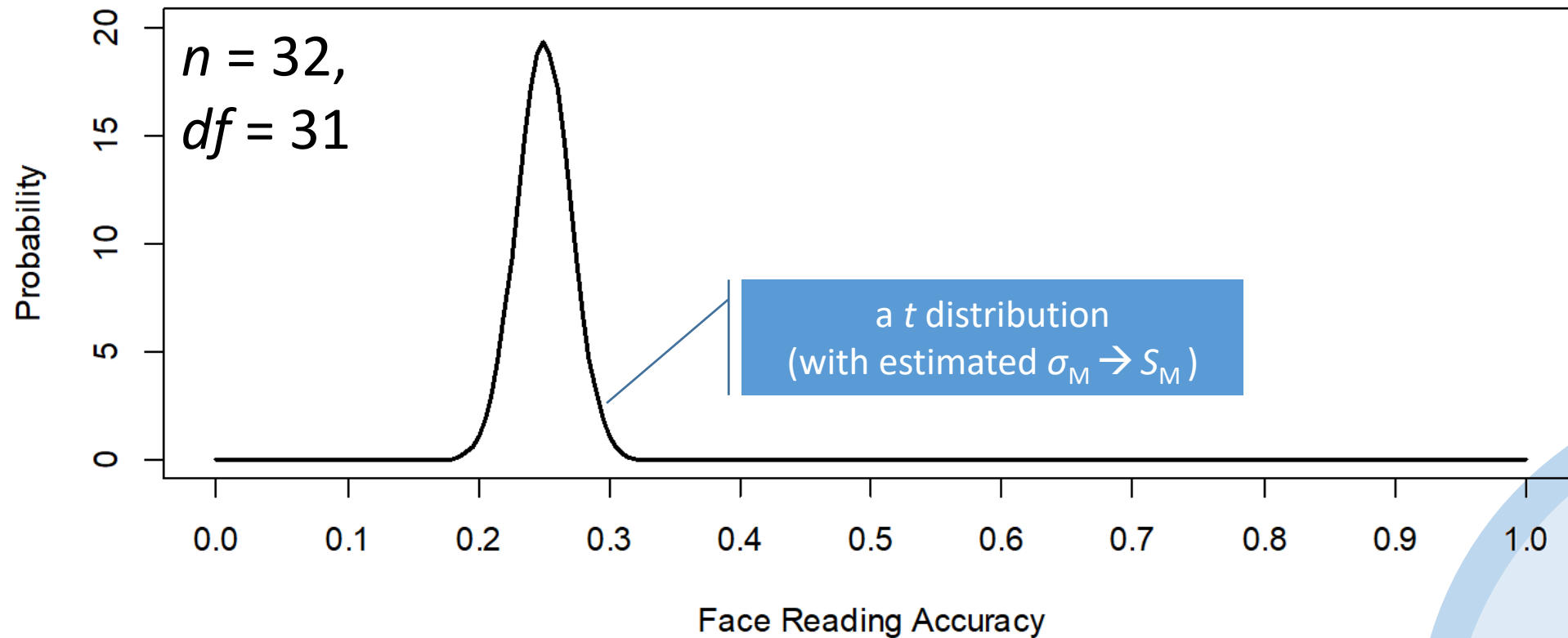
Note[^]: μ_M is the mean to be tested against the sample and M is the sample mean.

Note[#]: σ_M Standard Error is the Standard Deviation of *the distribution of sample means*.

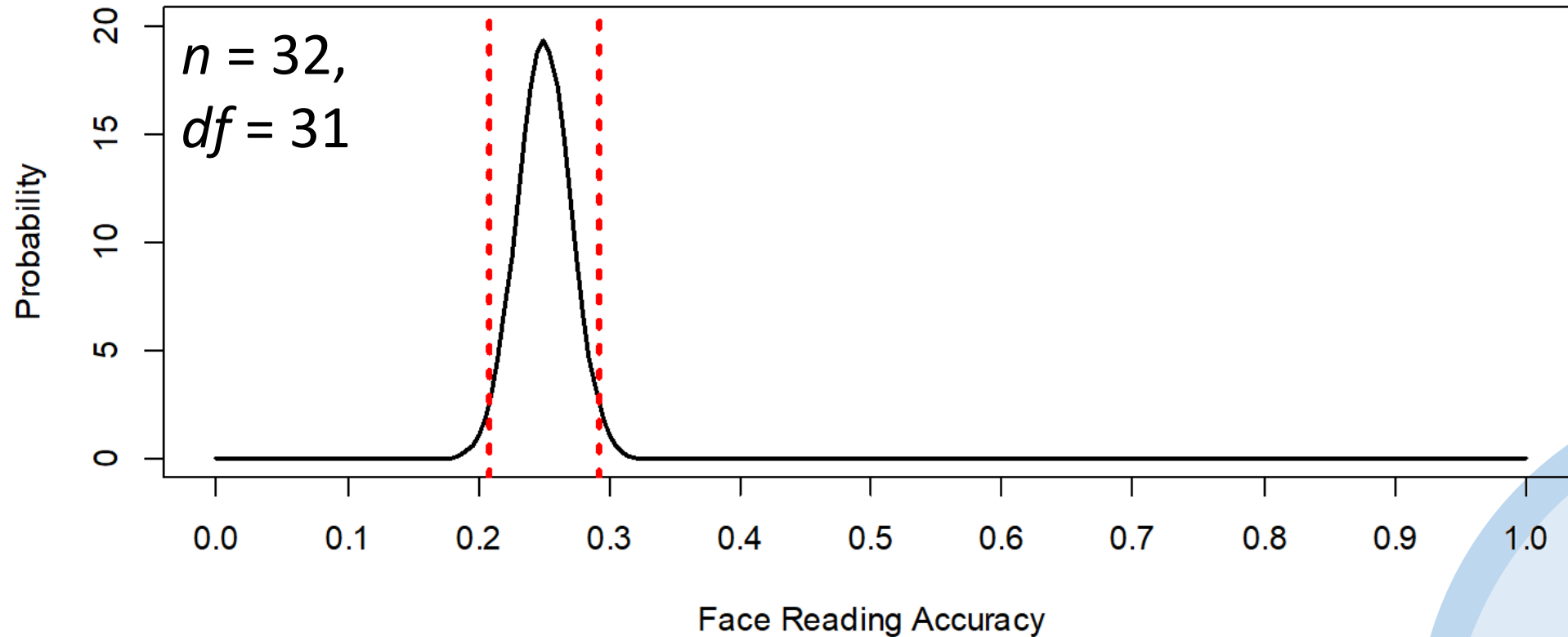
Population distribution: face reading experiment



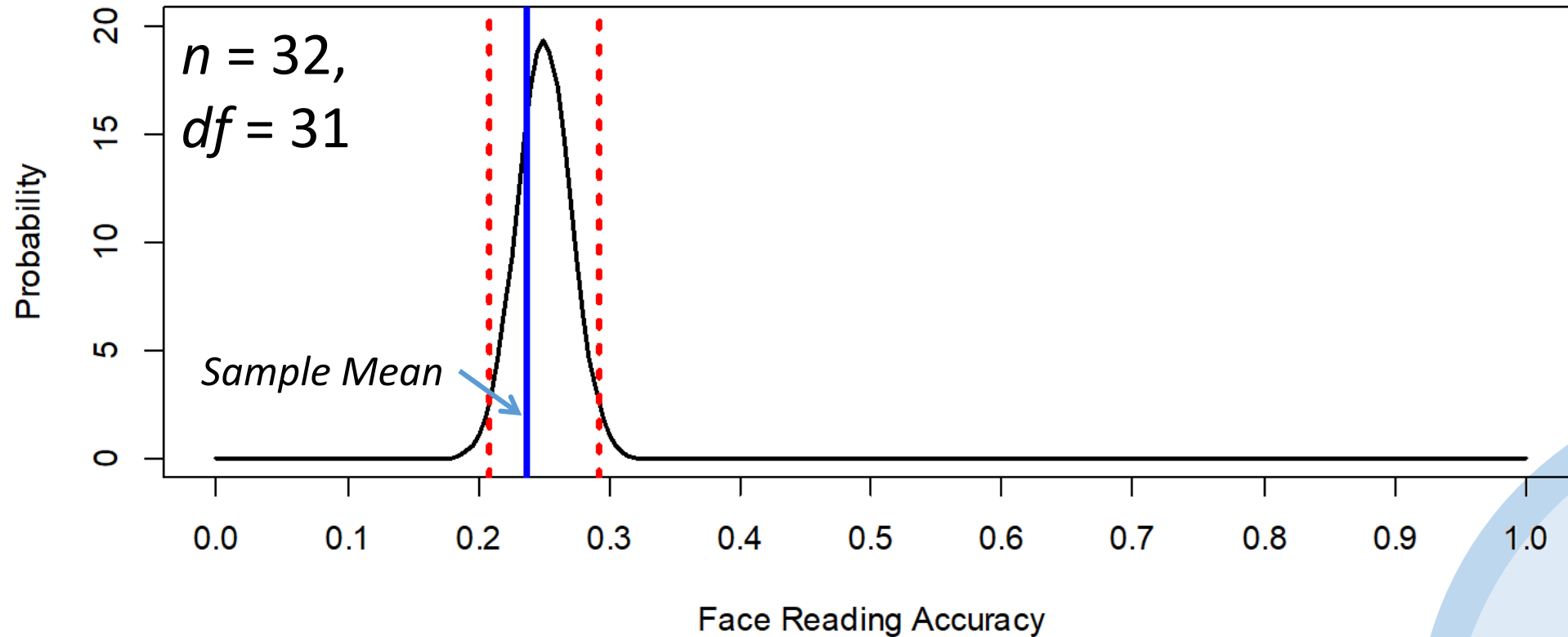
Sampling distribution



Sampling distribution



Sampling distribution



Hypothesis Testing – Step-by-step

- Step 0: Gather all the information you need, check the assumptions

- $\mu_{\text{baseline}} =$
 $\sigma = \text{unknown} \rightarrow \text{estimated with } S =$
 $M_{\text{treatment}} =$; $n =$; $df = n - 1$
kurtosis of population

- Step 1: Set up the hypotheses

- H_0 (null hypothesis)
- H_1 (Research hypothesis)

- $H_0:$
 $H_1:$

- Step 2: Calculate the test statistic

- $t_{\text{sample}} = \frac{M_{\text{treatment}} - \mu_{\text{baseline}}}{S_M}$

- Step 3: Find the probability associated with the test statistic
(i.e., the probability of getting such a score or a more extreme score x 2, given H_0 is true)

- $p(|t_{\text{sample}}| | H_0) =$

For a two-tailed test and t_{sample} on the left

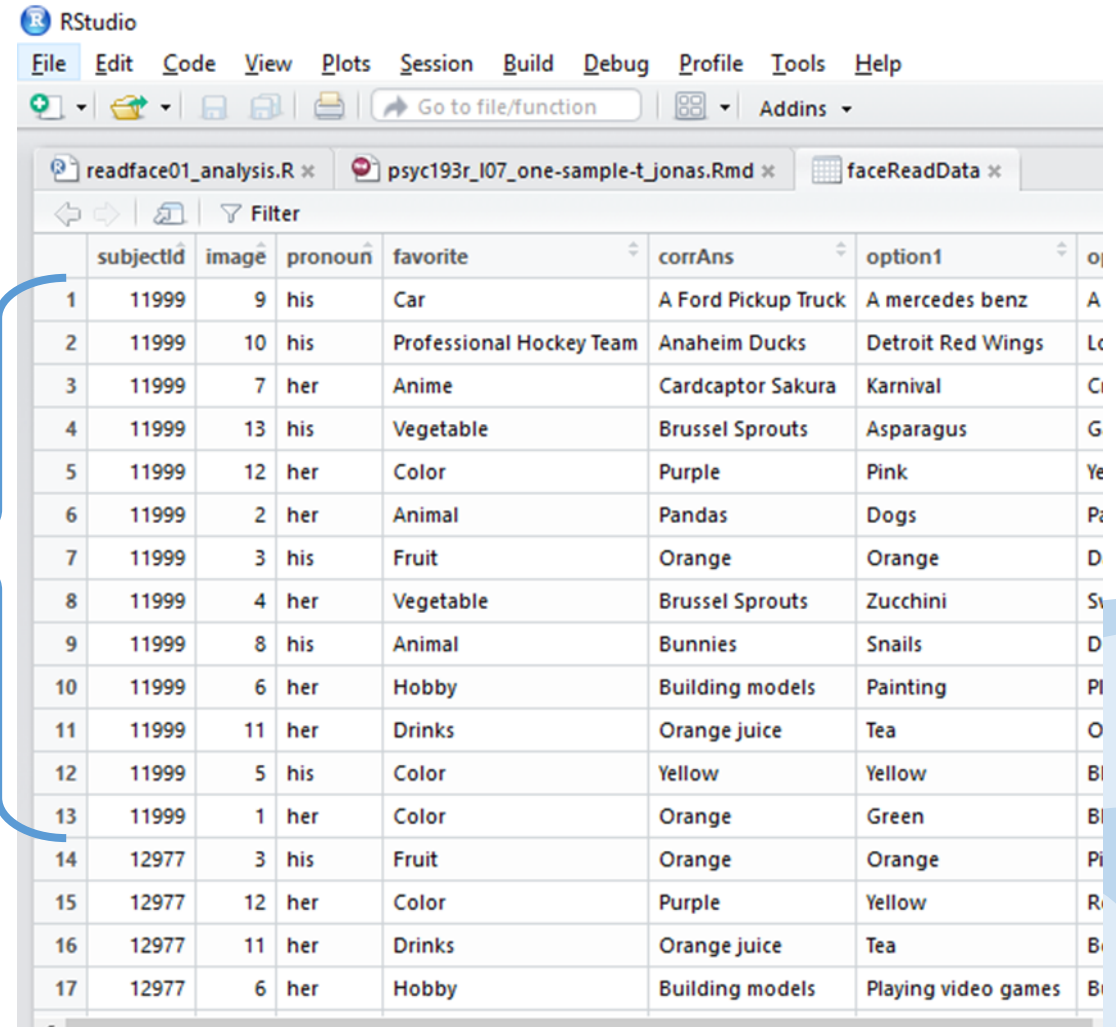
- Step 4: Decision & Conclusion
(compare p against α)

Import Data

- Download the data file:
https://psyc193r.ucsd.edu/data/psyc193r_readface_data.csv
- Tell R where your data file is:
Session → Set Working Directory → Choose Directory
`setwd()`
- Check the *comma separated values* (csv) file with MS Excel to ensure correct format
- Import the file into R as a data frame, saving it to “faceReadData” variable
`... = read.csv("file_name.csv")`

Data Structure

- The data have a "long format":
 - Each column represents a variable
 - Each row represents an entry (each trial)
- The variable we care the most are "subjectId", & "correct"
 - subjectId: a number that represents a participant (each should have 13 rows)
 - correct: denotes if a participant is correct on a particular trial



RStudio

File Edit Code View Plots Session Build Debug Profile Tools Help

Go to file/function

readface01_analysis.R x psyc193r_l07_one-sample-t_jonas.Rmd x faceReadData x

Filter

	subjectId	image	pronoun	favorite	corrAns	option1	option2
1	11999	9	his	Car	A Ford Pickup Truck	A mercedes benz	A
2	11999	10	his	Professional Hockey Team	Anaheim Ducks	Detroit Red Wings	Lo
3	11999	7	her	Anime	Cardcaptor Sakura	Karnival	Cl
4	11999	13	his	Vegetable	Brussel Sprouts	Asparagus	G
5	11999	12	her	Color	Purple	Pink	Ye
6	11999	2	her	Animal	Pandas	Dogs	Pi
7	11999	3	his	Fruit	Orange	Orange	D
8	11999	4	her	Vegetable	Brussel Sprouts	Zucchini	St
9	11999	8	his	Animal	Bunnies	Snails	D
10	11999	6	her	Hobby	Building models	Painting	Pl
11	11999	11	her	Drinks	Orange juice	Tea	O
12	11999	5	his	Color	Yellow	Yellow	Bl
13	11999	1	her	Color	Orange	Green	Bl
14	12977	3	his	Fruit	Orange	Orange	Pi
15	12977	12	her	Color	Purple	Yellow	R
16	12977	11	her	Drinks	Orange juice	Tea	B
17	12977	6	her	Hobby	Building models	Playing video games	B

Data Cleaning

- Convert `subjectId` into a factor
it means `subjectId` are nominal variables (they're just names)
Otherwise R would treat the variable as numerics
`faceReadData$subjectId = factor(faceReadData$subjectId)`
- Coerce / Make sure `correct` is a variable
so that we can do some computations with it
`faceReadData$correct = as.numeric(faceReadData$correct)`



Data Cleaning

- What we want is not exactly trial-level data, instead, we want to know the mean accuracy of each individual

We want to make a new data frame with two variables (columns), and save it to a new variable ("faceRead")

- subjectId
- accuracy

the data set

```
... = aggregate(correct ~ subjectId, data=faceReadData, FUN = mean)
```

group the scores "correct" by the same "subjectId"

use this function for the grouping

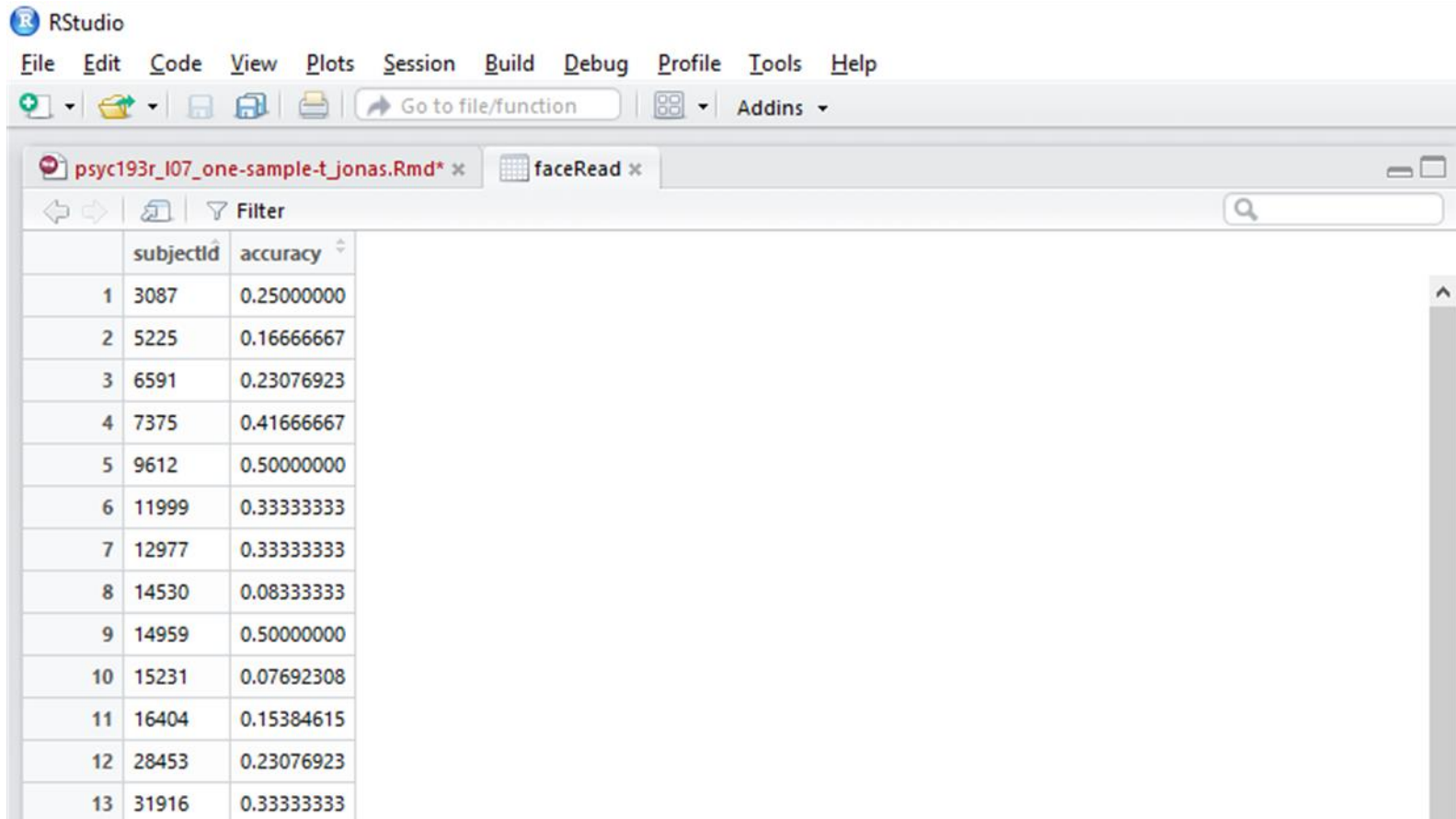
- Change the name of the aggregated variable to "accuracy"

```
names(faceRead)[2] = "accuracy"
```

name of the 2nd variable in "faceRead"

Data frame for t test

- Two columns: subjectId and accuracy
- Each row represents an entry from a participant



The screenshot shows the RStudio interface with a data frame loaded into the Environment pane. The data frame has two columns: 'subjectId' and 'accuracy'. The 'subjectId' column contains 13 unique values, and the 'accuracy' column contains corresponding numerical values. The data is as follows:

	subjectId	accuracy
1	3087	0.25000000
2	5225	0.16666667
3	6591	0.23076923
4	7375	0.41666667
5	9612	0.50000000
6	11999	0.33333333
7	12977	0.33333333
8	14530	0.08333333
9	14959	0.50000000
10	15231	0.07692308
11	16404	0.15384615
12	28453	0.23076923
13	31916	0.33333333

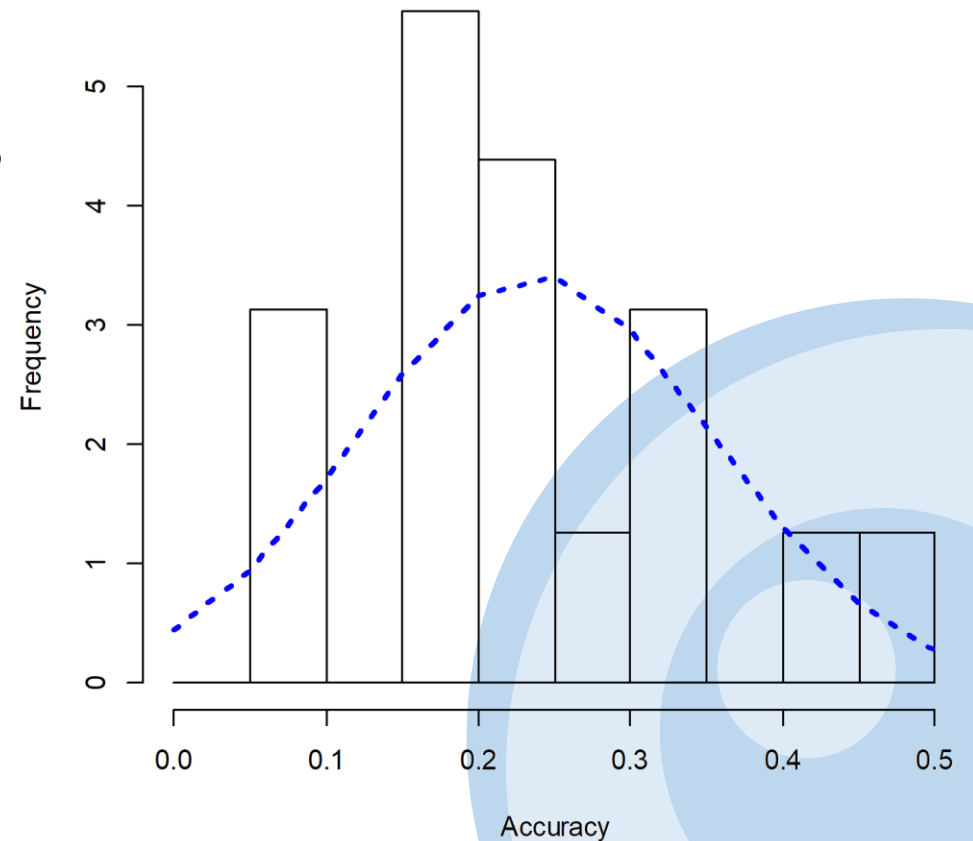
Examining the data / Checking assumptions

- Check the data structure for data analysis
 - One column of participant number ("subjectId")
 - One column of accuracy ("accuracy")
 - Each row represents an entry of one participant

head(...)

- Look at the data to see if there are outliers
 - Outlier tend to skew the sample mean
 - The easiest way is to make a histogram for visual assessment

hist(...)



Hypothesis Testing – Step-by-step

- Step 0: Gather all the information you need, check the assumptions

- $\mu_{\text{baseline}} =$
 $\sigma = \text{unknown} \rightarrow \text{estimated with } S =$
 $M_{\text{treatment}} =$; $n =$; $df = n - 1$
kurtosis of population

- Step 1: Set up the hypotheses

- H_0 (null hypothesis)
- H_1 (Research hypothesis)

- $H_0:$
 $H_1:$

- Step 2: Calculate the test statistic

- $t_{\text{sample}} = \frac{M_{\text{treatment}} - \mu_{\text{baseline}}}{S_M}$

- Step 3: Find the probability associated with the test statistic
(i.e., the probability of getting such a score or a more extreme score x 2, given H_0 is true)

- $p(|t_{\text{sample}}| | H_0) =$

For a two-tailed test and t_{sample} on the left

- Step 4: Decision & Conclusion
(compare p against α)

t.test() function for one-sample t-test

```
t.test(x, y = NULL,  
       alternative = c("two.sided", "less", "greater"),  
       mu = 0, paired = FALSE, var.equal = FALSE,  
       conf.level = 0.95, ...)
```

- x is an array: your sample scores
- mu is the baseline population mean
- Check ?t.test for usage. You can leave all the rest as default

```
t.test( ____, mu = ____, paired = FALSE)
```

Warning!!

Using built-in functions for statistical test is (almost too) easy,
make sure (1) you have the right data format,
(2) assumptions are met, (3) you are using the right test,
(4) you know how to interpret the results

The data analysis pipeline

Knowledge
of the field

(Cognitive, Sensation/Perception,
Developmental, Social, Neural)

Programming Skills

(generating / presenting stimulus,
automate data collection,
collecting reaction time data, etc)

Data Analysis

(data collection, data structure,
analysis and presentation)

+

Some Creativity

Define
research
questions

Design your
study
(to answer
the research
questions)

Come up
with a data
analysis plan

Get your data
(in the
correct
structure)

Structure
your data for
the analysis
(cleaning)

Analyze /
interpret
your data
(visualization)

The data analysis pipeline

Knowledge
of the field

(Cognitive, Sensation/Perception,
Developmental, Social, Neural)

Programming Skills

(generating / presenting stimulus,
automate data collection,
collecting reaction time data, etc)

Data Analysis

(data collection, data structure,
analysis and presentation)

+

Some Creativity

Define
research
questions

Design your
study
(to answer
the research
questions)

Come up
with a data
analysis plan

Get your data
(in the
correct
structure)

Structure
your data for
the analysis
(cleaning)

Analyze /
interpret
your data
(visualization)

t.test() function for one-sample t-test

```
t.test(x, y = NULL,  
       alternative = c("two.sided", "less", "greater"),  
       mu = 0, paired = FALSE, var.equal = FALSE,  
       conf.level = 0.95, ...)
```

- x is an array: your sample scores
- mu is the baseline population mean
- Check ?t.test for usage. You can leave all the rest as default

```
t.test(faceRead$accuracy, mu = 0.25, paired = FALSE, )
```

One Sample t-test

data: faceRead\$accuracy

t = -0.61023, df = 31, p-value = 0.5462

alternative hypothesis: true mean is not equal to 0.25

95 percent confidence interval:
0.1952799 0.2795162

sample estimates:

mean of x
0.237398

test statistics

Baseline population mean

Sample mean

Confidence Interval

Hypothesis Testing – Step-by-step

- Step 0: Gather all the information you need

- Check assumptions
- Data cleaning

- Step 1: Set up the hypotheses

- H_0 (null hypothesis)
- H_1 (Research hypothesis)

- Step 2: Calculate the test statistic

- Step 3: Find the probability associated with the test statistic

(i.e., the probability of getting such a score or a more extreme score x , given H_0 is true)

- Step 4: Decision & Conclusion
(compare p against α)

$\mu_{\text{baseline}} = 0.25$ (theoretical)

$S = \text{sd}()$

$M_{\text{treatment}} ; n ; df = n - 1 \rightarrow \text{t.test}()$
kurtosis of population ? *but $n \geq 30$ ✓*

$H_0:$

$H_1:$

See `t.test()` output

See `t.test()` output

Report baseline & sample means,
standard deviation, sample size,
decision/ conclusion
test statistics & p-value,

Effect Size Measure: Cohen's d

Magnitude of the effect in standard deviations

For comparison with known population standard deviation

$$\text{Cohen's } d = \frac{|\mu_{\text{treatment}} - \mu_{\text{baseline}}|}{\sigma}$$

For comparison with unknown population standard deviation

→ Estimate σ with S

$$\text{Cohen's } d = \frac{|\mu_{\text{treatment}} - \mu_{\text{baseline}}|}{S}$$

Cohen's d	Effect Size
0.2	Small
0.5	Medium
0.8	Large

Reliability Measure: Confidence Interval

When you repeat the same experiment (with the same condition and n), for ___% of the time, the sample mean will lie between [__ , __]

For comparison with known population standard deviation

$$M_{treatment} \pm Z_{\frac{\alpha}{2}} \times \sigma_M$$

Or, `pnorm( c(__, __), mean = __, sd = __ )`

For comparison with unknown population standard deviation
→ Estimate σ with S

$$M_{treatment} \pm t_{\frac{\alpha}{2}, df} \times S_M$$

Or, obtain in `t.test()` output

Statistical Power

Recall that:

- The main goal of power analysis is for sample size estimation
- The estimation is not exact because sample mean is generally unknown prior to an experiment

```
power = function(treatment_mean,
baseline_pop_mean, pop_sd, sample_size,
alpha = 0.05){
  # Calculate the standard error
  se = NA

  # Find the cutoff points in raw scores
  cutoff = NA

  # Find the probability of
  # getting a sample mean
  # more extreme than the cutoff scores
  # under H1
  stat_power = NA

  # save sample_size and power to an
array
  answer = NA

  # return the array
  return (answer)
}
```

Statistical Power

Recall that:

- When $n \geq 30$, the t distribution approaches Z distribution
- The power function we wrote can be used to estimate power in t tests ($n \geq 30$)
 - Note that the function tends to **overestimate** the actual power slightly
 - The sample size should be adjusted **upward** to compensate for that
- A new function can be built for t test, or you can use resources like:
pwr package in R
G*Power (<http://www.gpower.hhu.de/>)

```
power = function(treatment_mean,
baseline_pop_mean, pop_sd, sample_size,
alpha = 0.05){
  # Calculate the standard error
  se = NA

  # Find the cutoff points in raw scores
  cutoff = NA

  # Find the probability of
  # getting a sample mean
  # more extreme than the cutoff scores
  # under H1
  stat_power = NA

  # save sample_size and power to an
  array
  answer = NA

  # return the array
  return (answer)
}
```

pwr package in R

- Install a package in R

```
install.packages("pwr")
```

- Load the package each time after restarting R, if you need the package

```
library("pwr")
```

- Usage

```
?pwr.t.test
```

```
pwr.t.test(n = NULL, d = NULL, sig.level = 0.05,  
power = NULL, type = c("two.sample", "one.sample", "paired"),  
alternative = c("two.sided", "less", "greater"))
```

- Leave the argument that you are looking for as NULL

- E.g.,

```
pwr.t.test(n = 50, d = 0.5, sig.level = 0.05, power = NULL,  
type = "one.sample")
```