Topics:

- QQ-plot → data ~ dist (visual)
- confidence Intervals → [code]
- bootstrapping
- CLT
- hypothesis testing (2-3 classes) ✓

[Questions
Chat]

QQ-plot

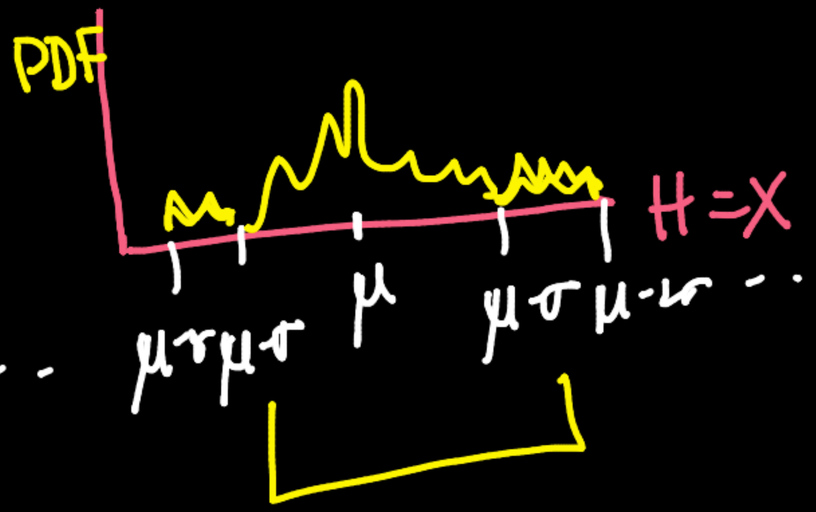
X : heights

$\underbrace{x_1, x_2, \ldots x_{100}}_{\text{observed}} \sim \text{Normal}(\mu = 160, \sigma = 10)$

Q how can you verify that 100 obs follow this dist

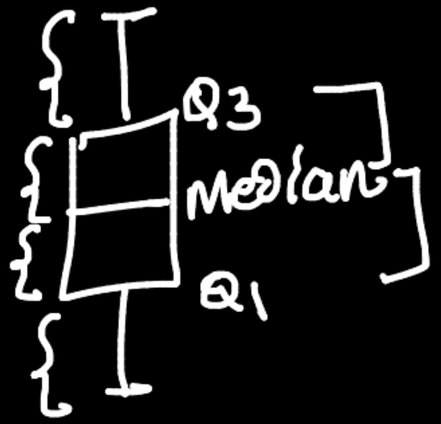
X → 67-95-99 rule

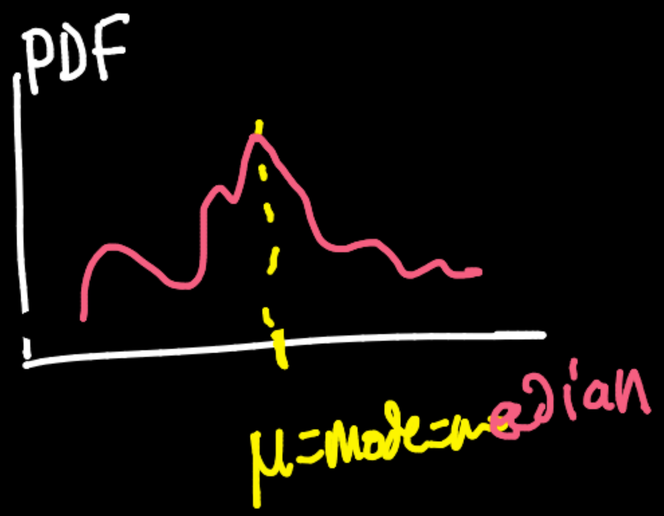
but not gaussian ..



X → median = mode = mean

→ boxplot





→ bell-shaped curve + 67-95-99

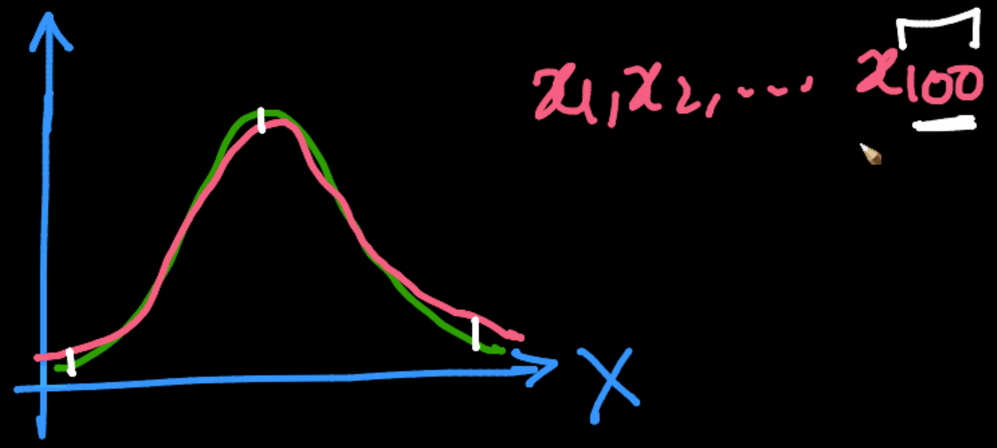
↳ binomial-distb: bell-shaped } like a

100 obs → empirical PDF ✓

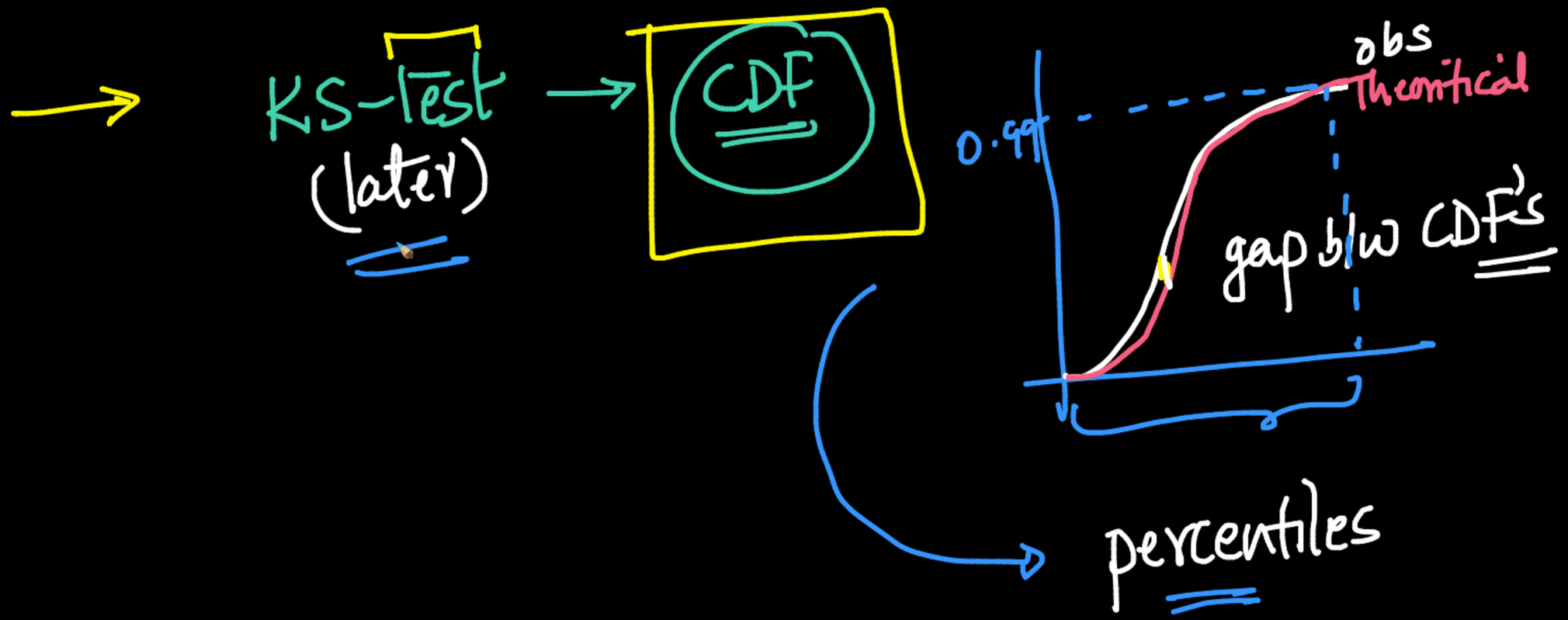
100 obs

(*) CDF → PDF

$P(X \leq x)$

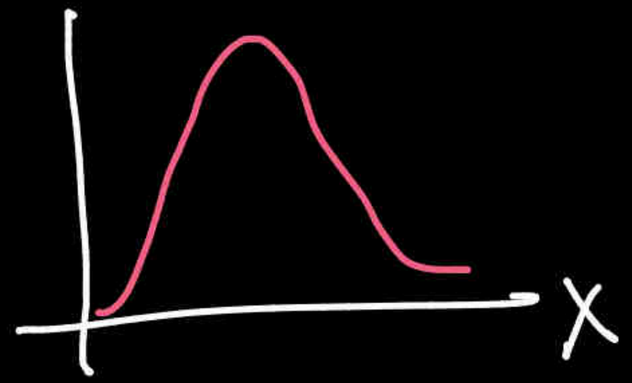


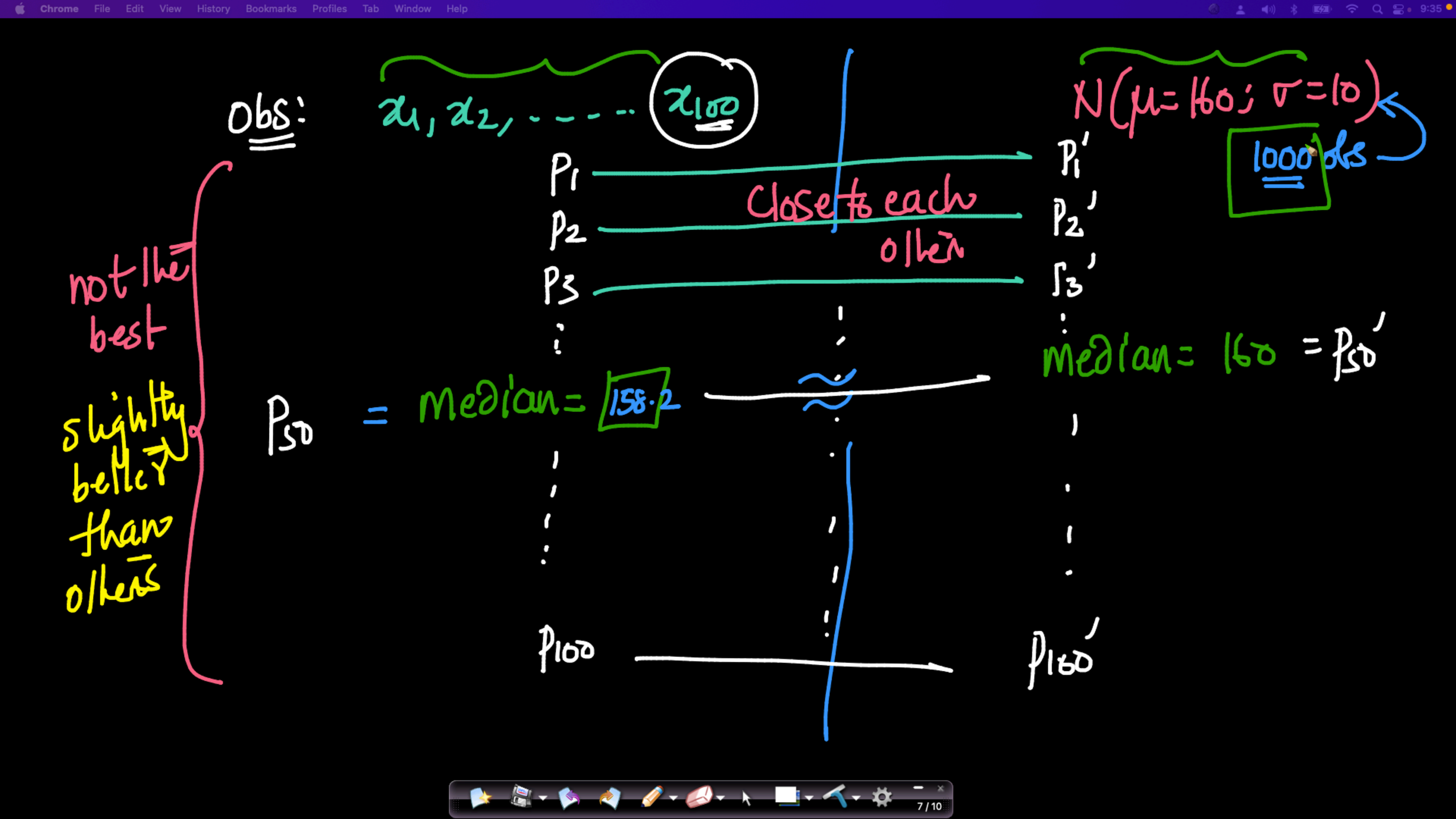
need not exactly overlap theoritical PDF

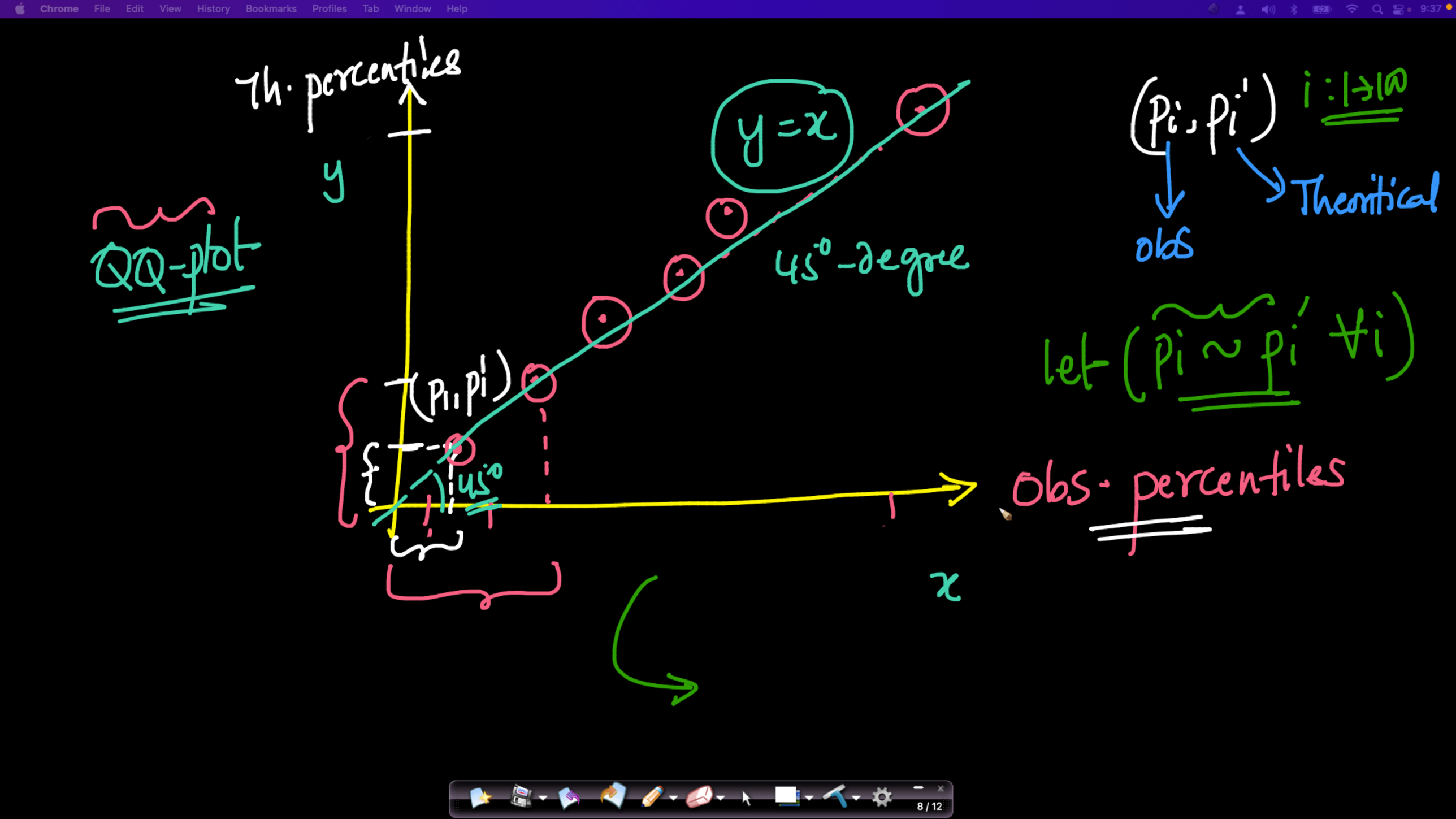


$$X \sim N(\mu = 160;\ \sigma = 10)$$

$$P(X = x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{\frac{-1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right\}$$




Obs:
$x_1, x_2, \ldots, x_{100}$
$N(\mu = 160; \sigma = 10)$
1000 obs
P_1
P_2
P_3
Close to each other
P_1'
P_2'
P_3'
not the best
slightly better than others
P_{50} = median = 158.2
median = 160 = P_{50}'
P_{100}
P_{160}'


Th. percentiles
y
y=x
45°-degree
(Pi, Pi')
45°
x
Obs. percentiles
QQ-plot
(Pi, Pi') i : 1 → n
Obs
Theoritical
let (Pi ~ Pi' ∀i)

Visual

(+) if all (p_i', p_i'') lie "close" to the $y=x$ (45°) line then obs-data follows the th. dist

Drawbacks

- how "close"? → subjective
- mathematical value on "confident" we are

$x_1, x_2, \ldots x_{100} \sim N(\mu = 160; \sigma = 10)$

100 is reasonable size sample

$m \approx \mu$ → for now

$s \approx \sigma$

(later) — better techniques

good enough approx

Rules of thumb

Classical stats

- not for ML & DL
- > 30 obs ✓
- ~100 obs → V. good
- ~1000 obs → great

QQ

comparing 2-dists

med1: #day rec. times

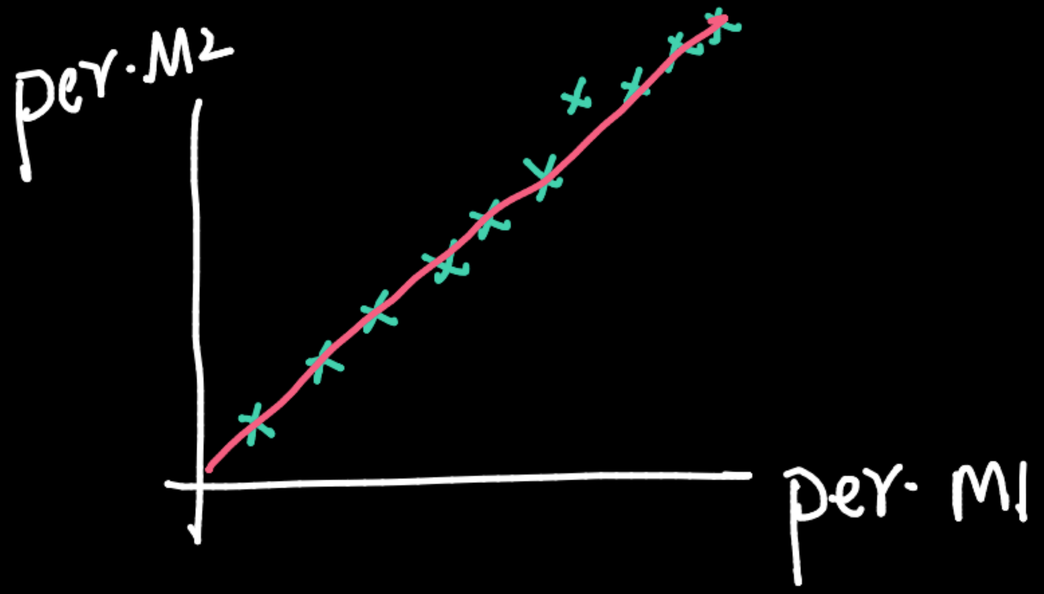
med2: #days to recover

100

120

need not be gaussian

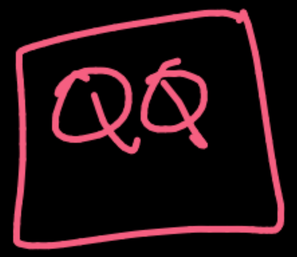
$p_1, p_2, \ldots p_{100}$



(p_i, p_i')

p_i → med1

p_i' → med2



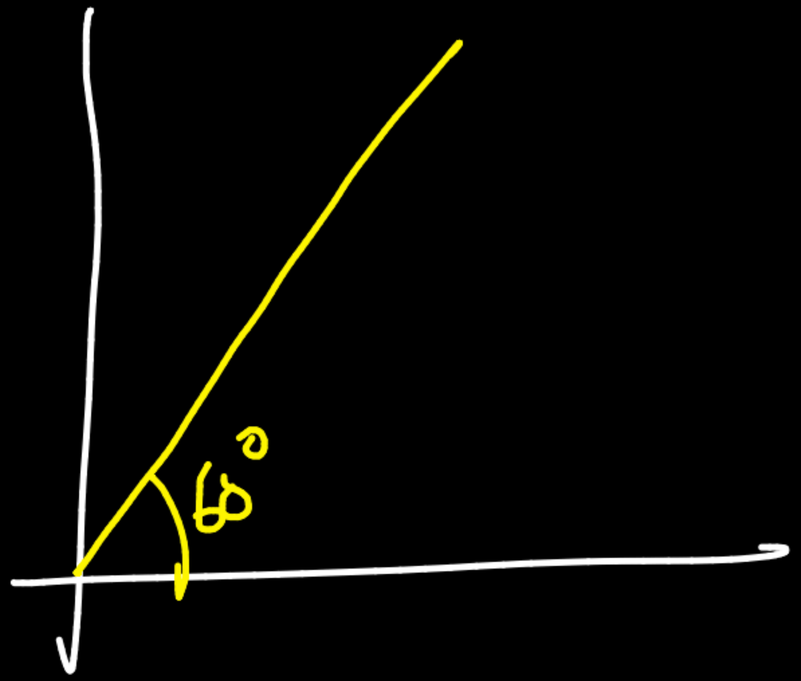
$X \sim Normal(\mu, \sigma)$

$Y \sim Binomial(n, p)$ ← QQ-plot

$y_1 \cdots y_{200}$ → percentiles

Binomial → percentiles

$x_1, x_2 \cdots x_{100} \sim N(\mu=160, \sigma=10)$

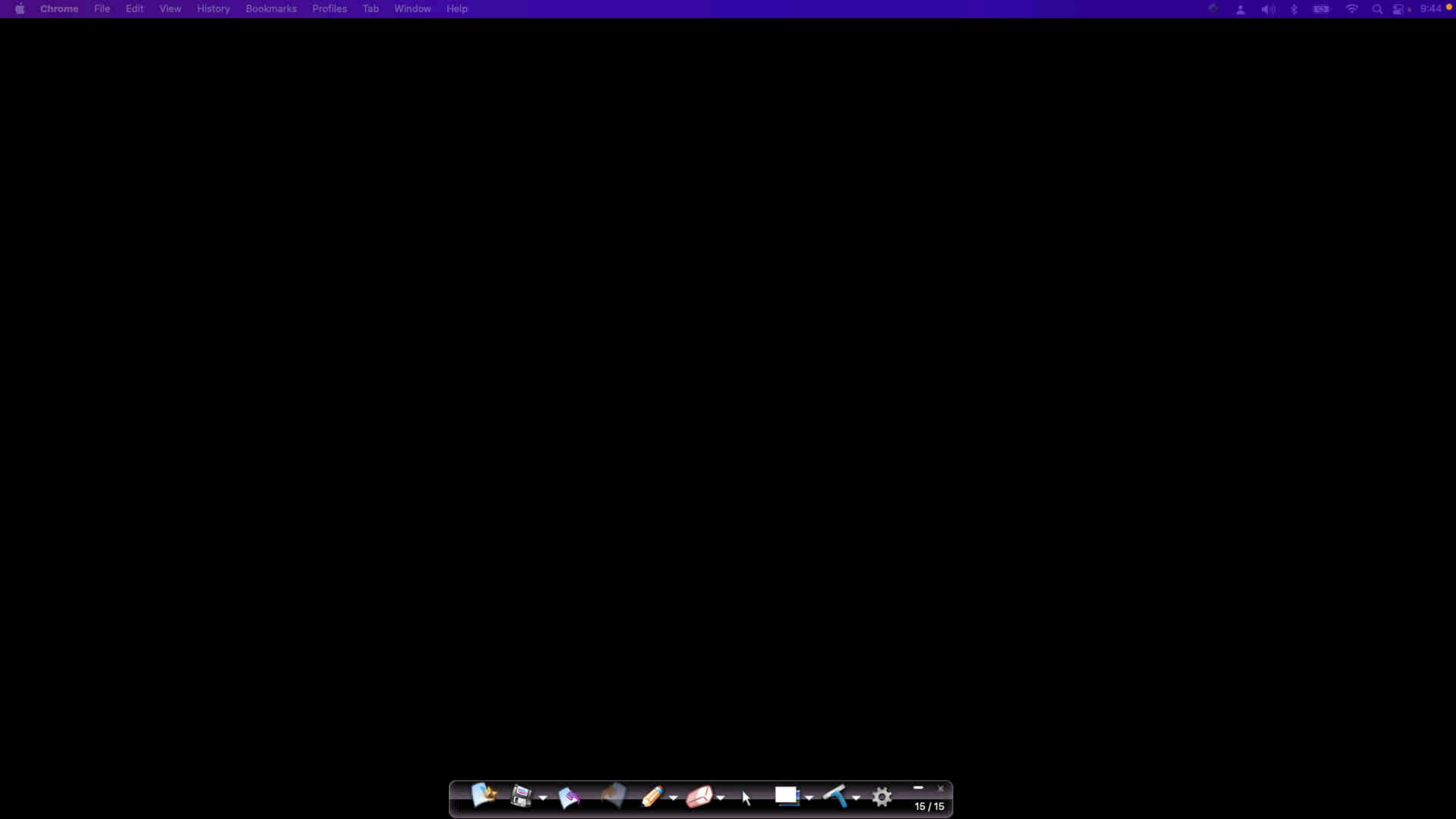


→ distributions Shape are similar

$x_1 \cdots x_{100} \sim$ Normal

but not

Normal ($\mu=160$ $\sigma=10$)


Chrome File Edit View History Bookmarks Profiles Tab Window Help
9:44
15 / 15

Chrome File Edit View History Bookmarks Profiles Tab Window Help 9:47

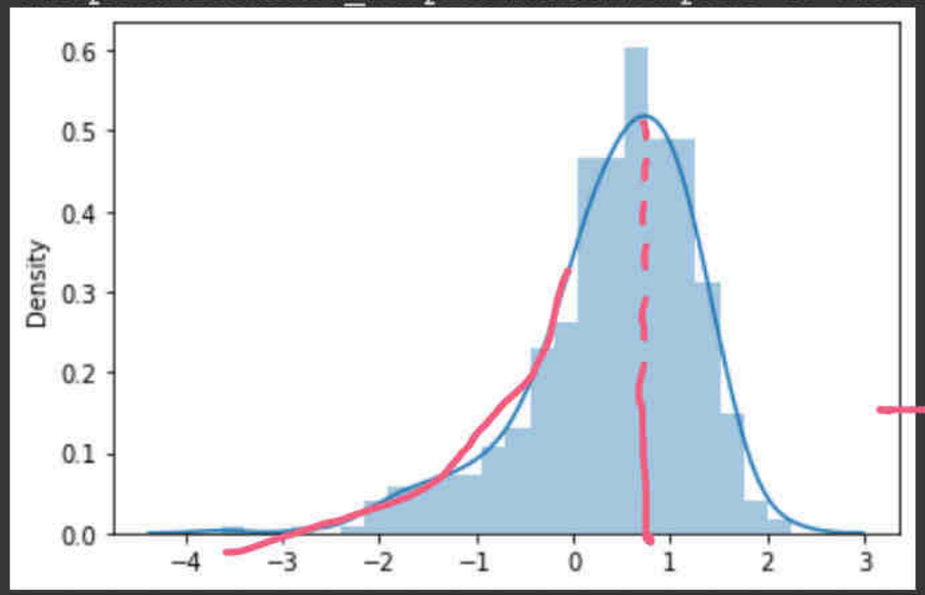
QQ-plot and CLT and Bootstra | New Tab

colab.research.google.com/drive/1m1Wa-7i9WknfRE9Y_aoi_5cbm5AAJBjW#scrollTo=1xlnUt3l1wcp

+ Code + Text RAM Disk

```
sns.distplot(x)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated func
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7fc92718db50>
```



n=500 obs

- Cannot be gaussian

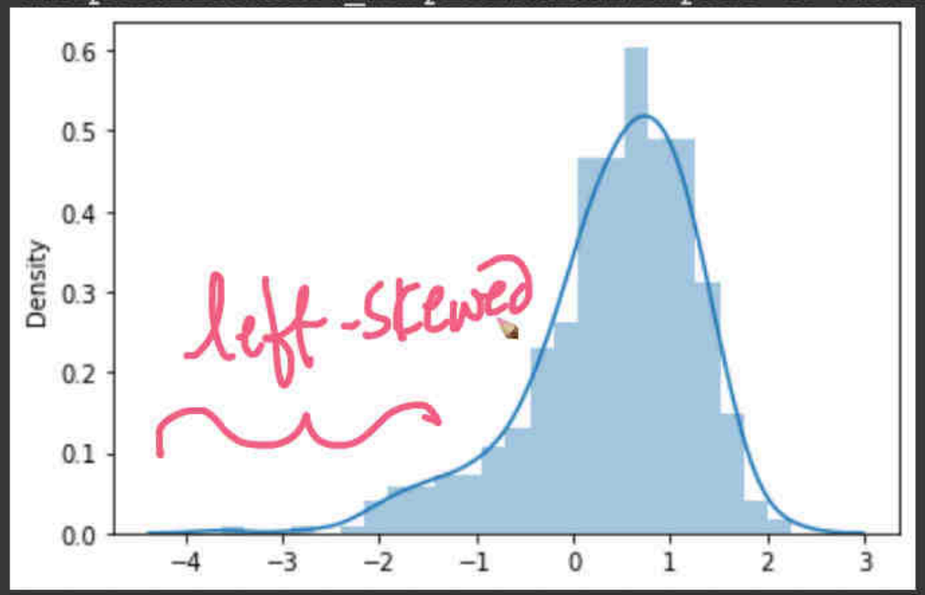
→ not symmetric

```
#QQ-Plot
fig, ax1 = plt.subplots()
plt.grid()
prob = stats.probplot(x, dist=stats.norm, plot=ax1)
```

16 / 16

```
sns.distplot(x)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated func
  warnings.warn(msg, FutureWarning)
<matplotlib.axes._subplots.AxesSubplot at 0x7fc92718db50>
```




```
#QQ-Plot
fig, ax1 = plt.subplots()
plt.grid()
prob = stats.probplot(x, dist=stats.norm, plot=ax1)
```

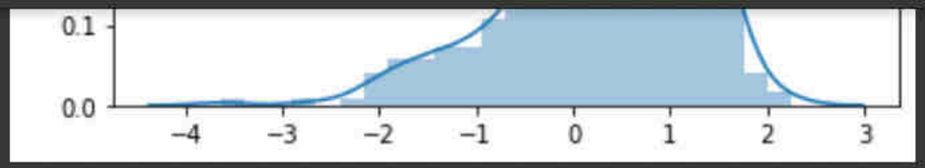
Chrome File Edit View History Bookmarks Profiles Tab Window Help 9:49

QQ-plot and CLT and Bootstra | New Tab

colab.research.google.com/drive/1m1Wa-7i9WknfRE9Y_aoi_5cbm5AAJBjW#scrollTo=1xlnUt3l1wcp

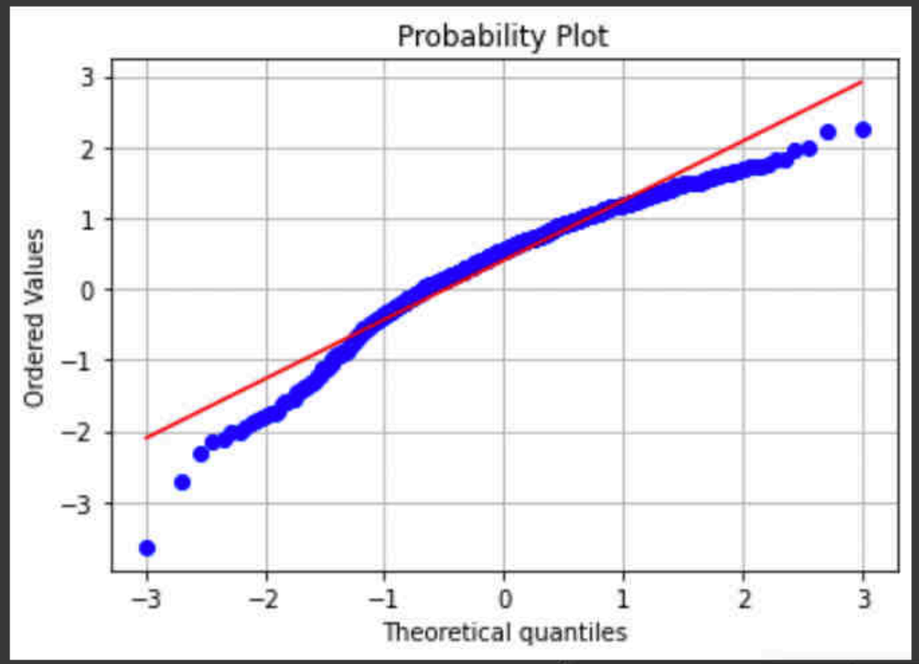
+ Code + Text

RAM
Disk




```
#QQ-Plot
fig, ax1 = plt.subplots()
plt.grid()
prob = stats.probplot(x, dist=stats.norm, plot=ax1)
```

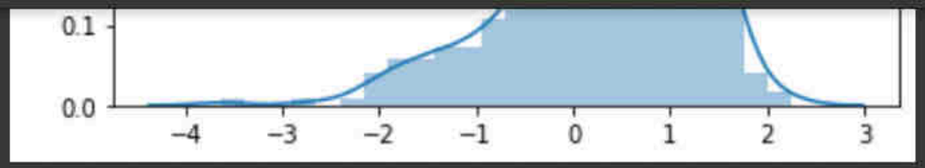
scipy.stats



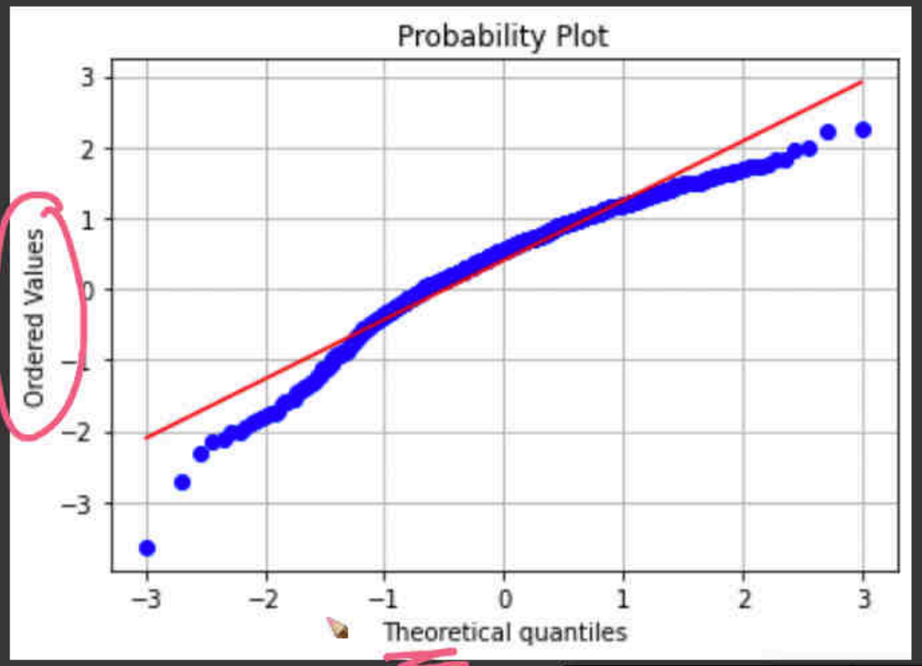
$x_1, x_2 \ldots x_{500}$

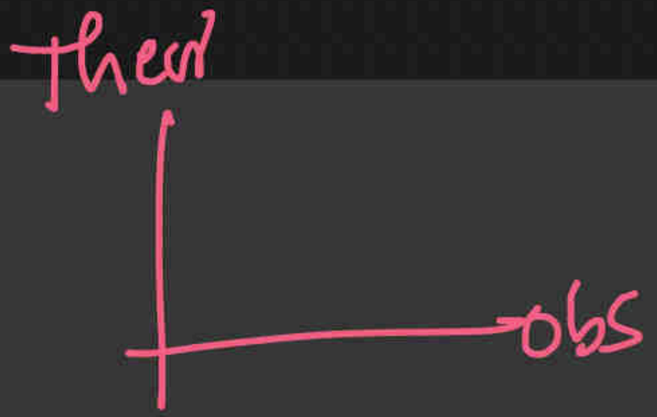
Normal (μ, σ)

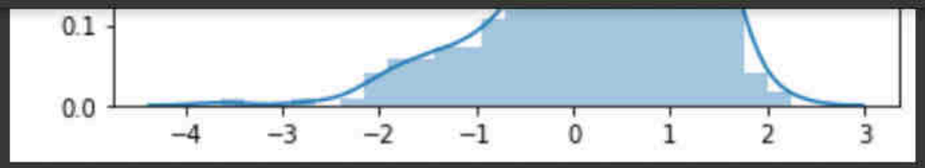
18 / 18



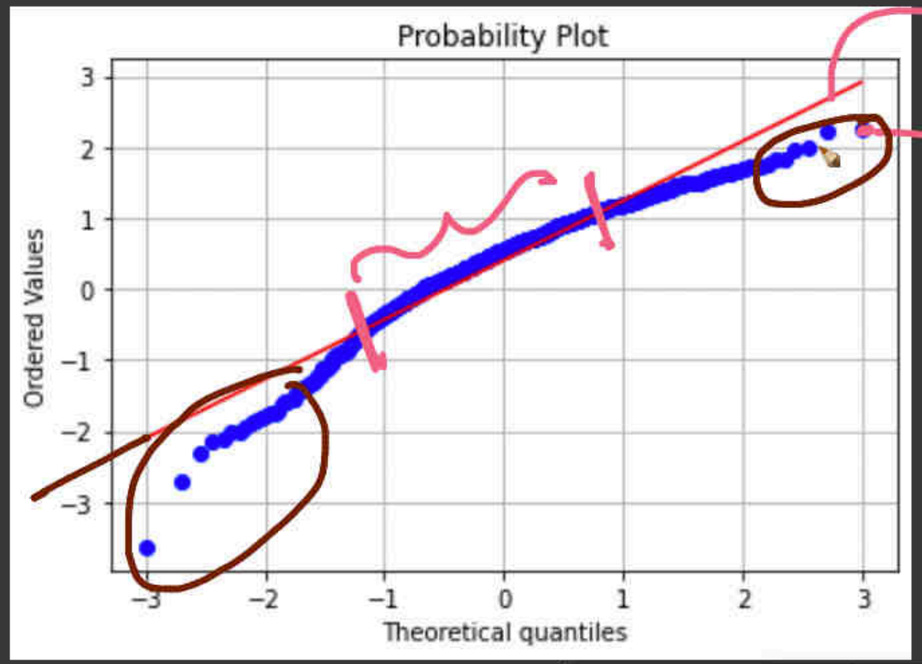

```
#QQ-Plot
fig, ax1 = plt.subplots()
plt.grid()
prob = stats.probplot(x, dist=stats.norm, plot=ax1)
```



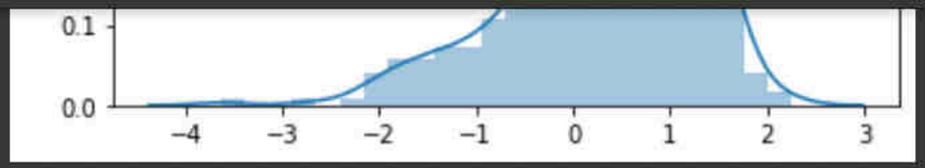





```
#QQ-Plot
fig, ax1 = plt.subplots()
plt.grid()
prob = stats.probplot(x, dist=stats.norm, plot=ax1)
```

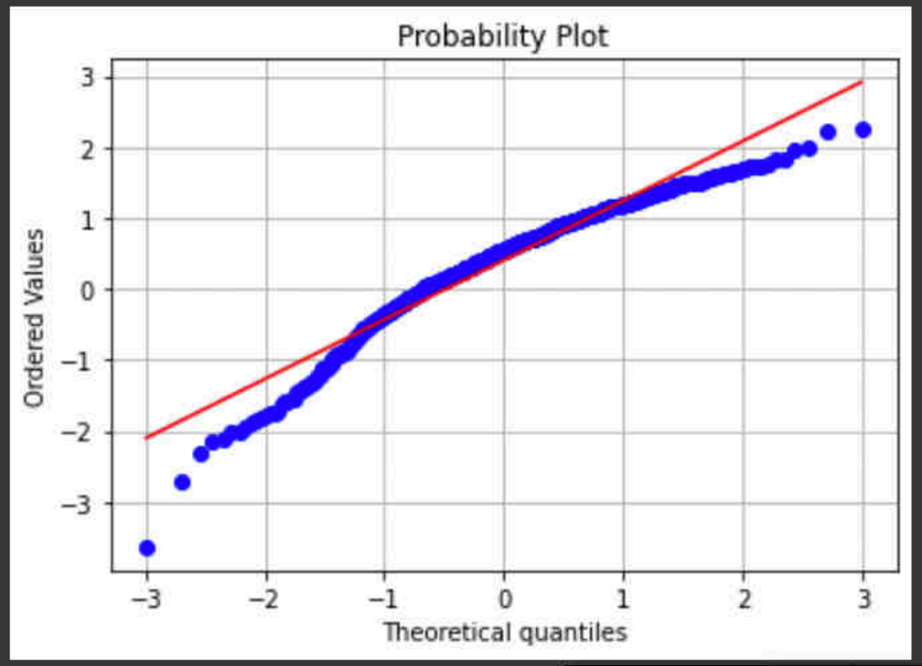


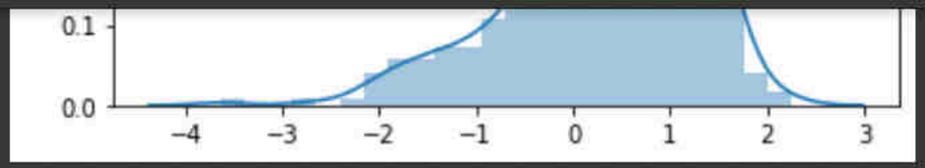
45° degree
(pi, pi')



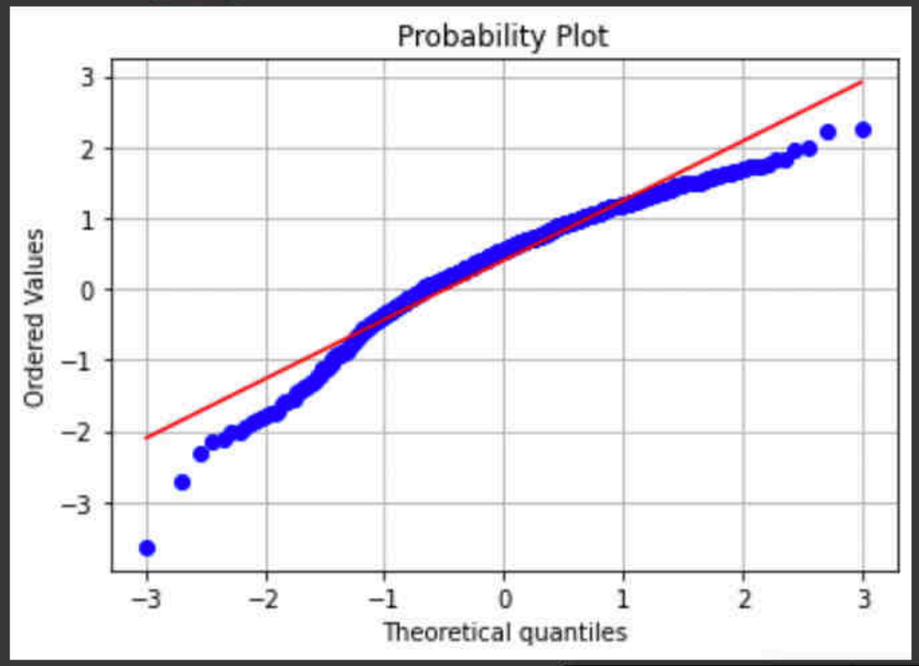

```
#QQ-Plot
fig, ax1 = plt.subplots()
plt.grid()
prob = stats.probplot(x, dist=stats.norm, plot=ax1)
```

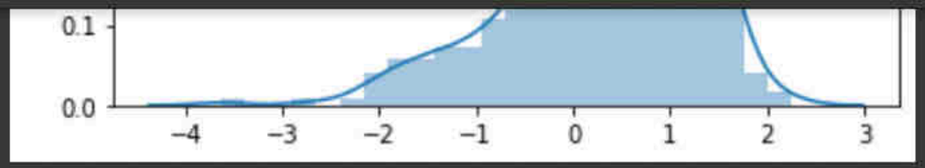
Gaussian — QQplot



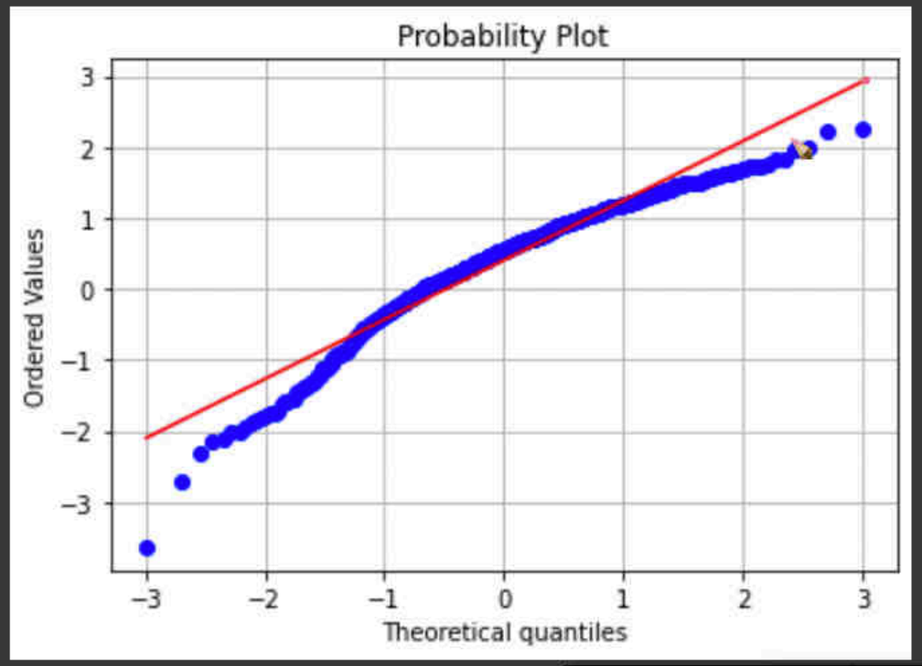



```
#QQ-Plot
fig, ax1 = plt.subplots()
plt.grid()
prob = stats.probplot(x, dist=stats.norm, plot=ax1)
```






```
[ ]  #QQ-Plot
     fig, ax1 = plt.subplots()
     plt.grid()
     prob = stats.probplot(x, dist=stats.norm, plot=ax1)
```



Search the docs ...

(scipy.sparse.csgraph)
Spatial algorithms and data structures
(scipy.spatial)
Distance computations
(scipy.spatial.distance)
Special functions (scipy.special)
Statistical functions (scipy.stats)
Result classes
Contingency table functions
(scipy.stats.contingency)
Statistical functions for masked arrays
(scipy.stats.mstats)
Quasi-Monte Carlo submodule
(scipy.stats.qmc)
Random Number Generators
(scipy.stats.sampling)
Low-level callback functions

squares fit. *plot* is an object that has to have methods "plot" and "text". The `matplotlib.pyplot` module or a Matplotlib Axes object can be used, or a custom object with the same methods. Default is None, which means that no plot is created.

Returns: **(osm, osr) : *tuple of ndarrays***

Tuple of theoretical quantiles (osm, or order statistic medians) and ordered responses (osr). *osr* is simply sorted input *x*. For details on how *osm* is calculated see the Notes section.

(slope, intercept, r) : *tuple of floats, optional*

Tuple containing the result of the least-squares fit, if that is performed by `probplot`. *r* is the square root of the coefficient of determination. If `fit=False` and `plot=None`, this tuple is not returned.

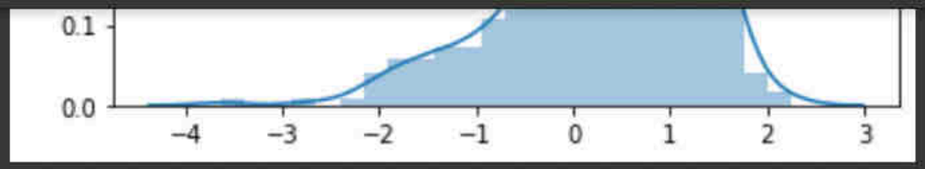
Notes

Even if *plot* is given, the figure is not shown or saved by `probplot`; `plt.show()` or `plt.savefig('figname.png')` should be used after calling `probplot`.

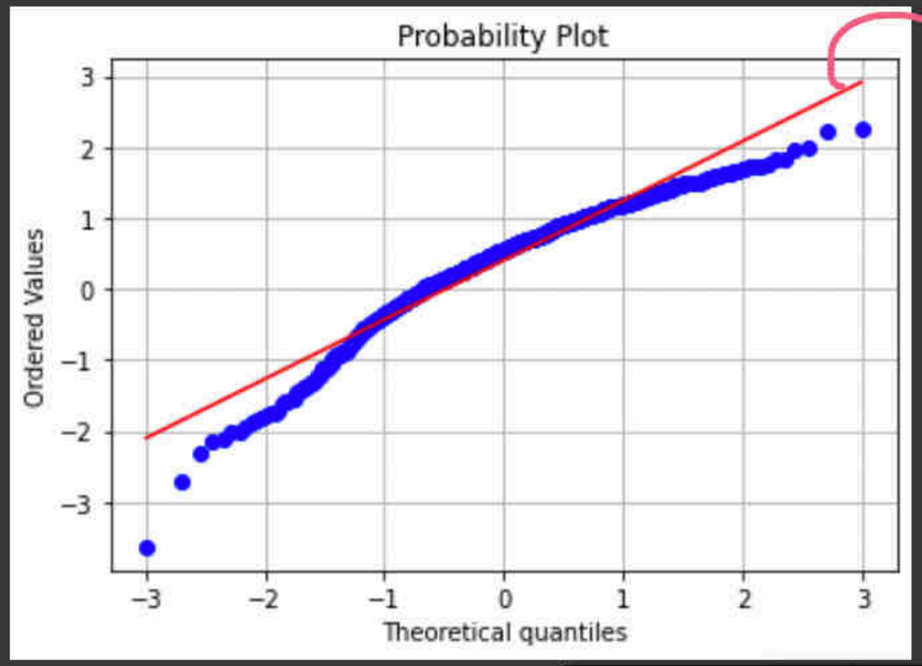
`probplot` generates a probability plot, which should not be confused with a Q-Q or a P-P plot. Statsmodels has more extensive functionality of this type, see `statsmodels.api.ProbPlot`.

The formula used for the theoretical quantiles (horizontal axis of the probability plot) is Filliben's

[5]




```
#QQ-Plot
fig, ax1 = plt.subplots()
plt.grid()
prob = stats.probplot(x, dist=stats.norm, plot=ax1)
```



not the 45°

squares fit. *plot* is an object that has to have methods "plot" and "text". The **matplotlib.pyplot** module or a Matplotlib Axes object can be used, or a custom object with the same methods. Default is None, which means that no plot is created.

Returns: **(osm, osr) : *tuple of ndarrays***

Tuple of theoretical quantiles (osm, or order statistic medians) and ordered responses (osr). *osr* is simply sorted input *x*. For details on how *osm* is calculated see the Notes section.

(slope, intercept, r) : *tuple of floats, optional*

Tuple containing the result of the least-squares fit, if that is performed by **probplot**. *r* is the square root of the coefficient of determination. If `fit=False` and `plot=None`, this tuple is not returned.

linear-regression (ML)

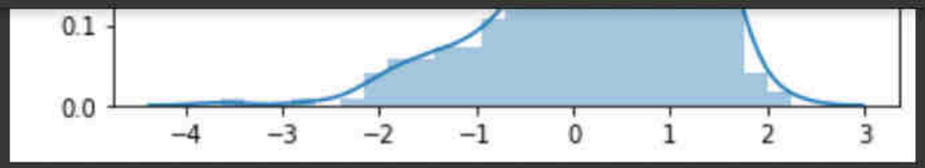
Notes

Even if *plot* is given, the figure is not shown or saved by **probplot**; `plt.show()` or `plt.savefig('figname.png')` should be used after calling **probplot**.

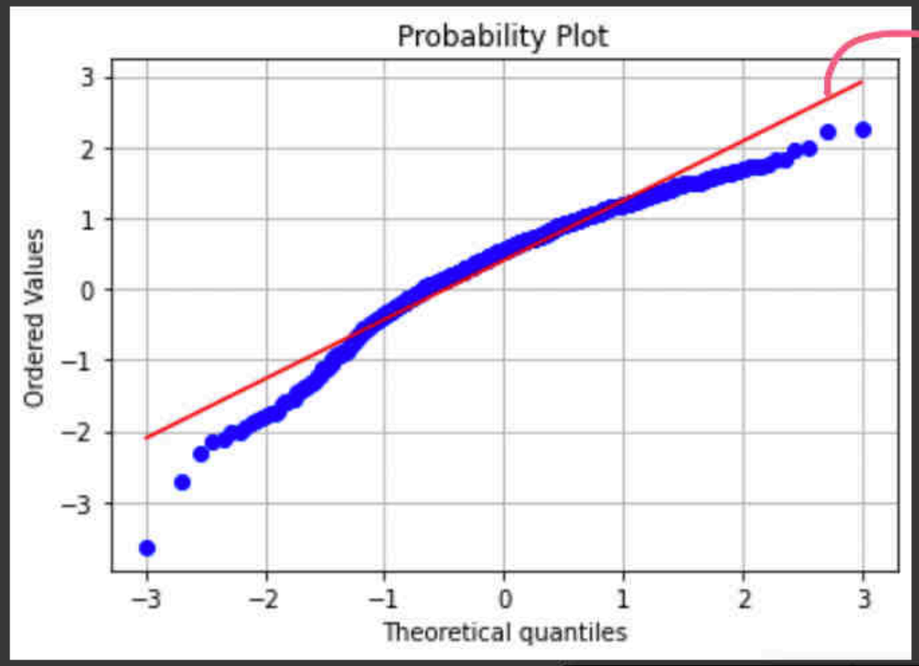
probplot generates a probability plot, which should not be confused with a Q-Q or a P-P plot. Statsmodels has more extensive functionality of this type, see `statsmodels.api.ProbPlot`.

The formula used for the theoretical quantiles (horizontal axis of the probability plot) is Filliben's

[5]

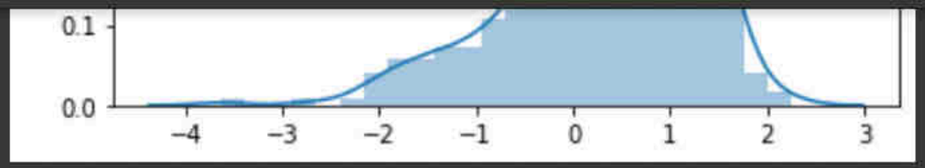



```
#QQ-Plot
fig, ax1 = plt.subplots()
plt.grid()
prob = stats.probplot(x, dist=stats.norm, plot=ax1)
```

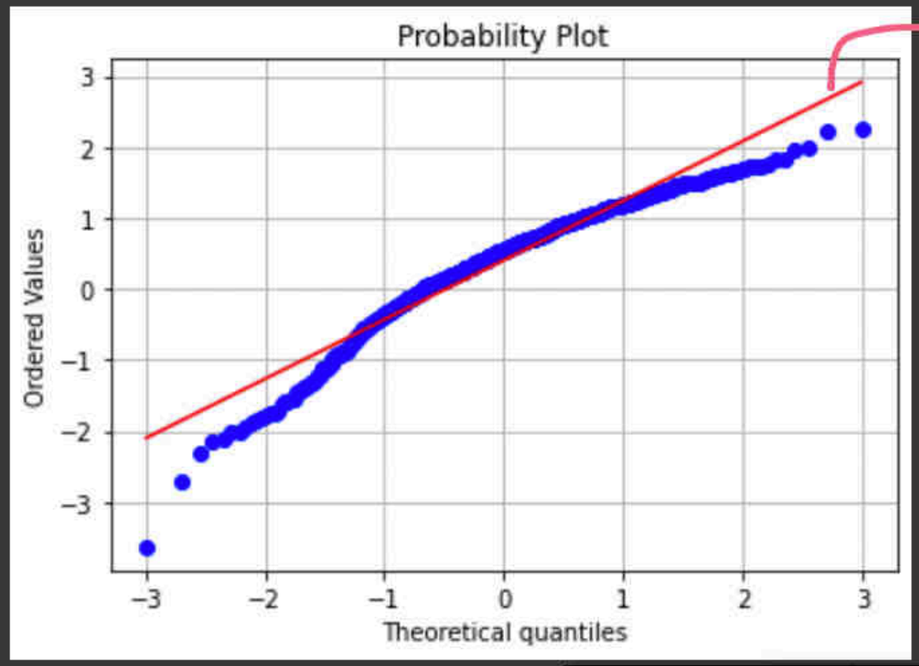


best line

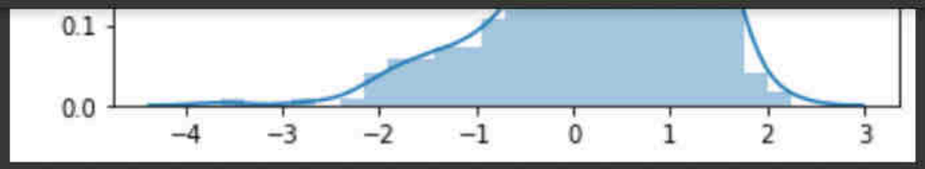
[5]




```
#QQ-Plot
fig, ax1 = plt.subplots()
plt.grid()
prob = stats.probplot(x, dist=stats.norm, plot=ax1)
```



[5]



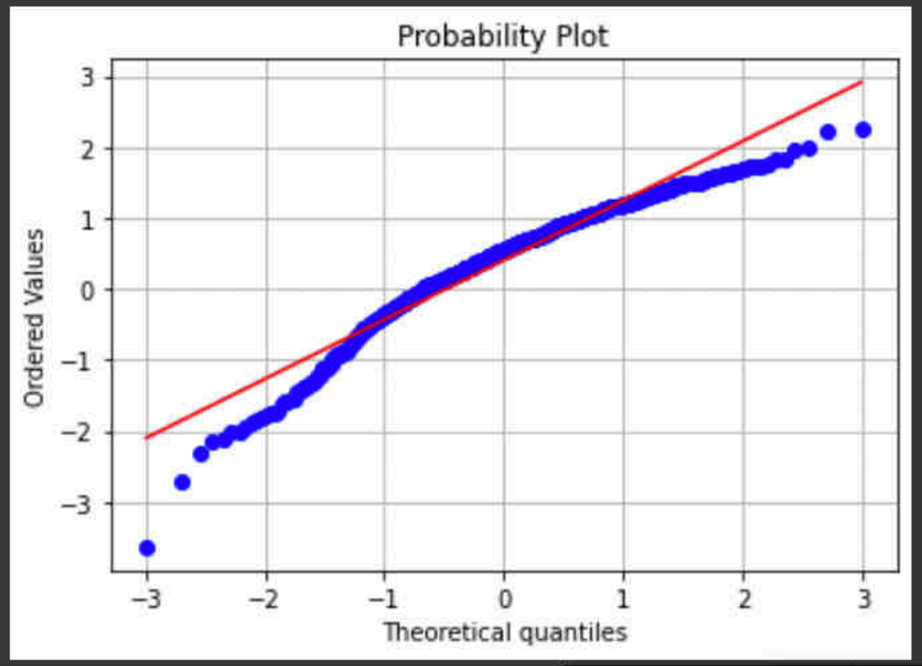

```
#QQ-Plot
fig, ax1 = plt.subplots()
plt.grid()
prob = stats.probplot(x, dist=stats.norm, plot=ax1)
```

$x_1, x_2 \cdots x_{100}$

P_1:

P_2:

P_1':



Chrome File Edit View History Bookmarks Profiles Tab Window Help 9:55

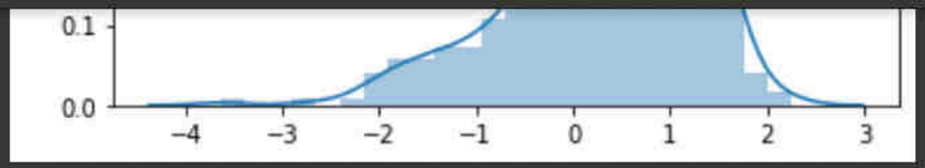
QQ-plot and CLT and Bootstra | New Tab | scipy.stats.probplot — SciPy

colab.research.google.com/drive/1m1Wa-7i9WknfRE9Y_aoi_5cbm5AAJBjW#scrollTo=ajaLzJL_JPre

+ Code + Text

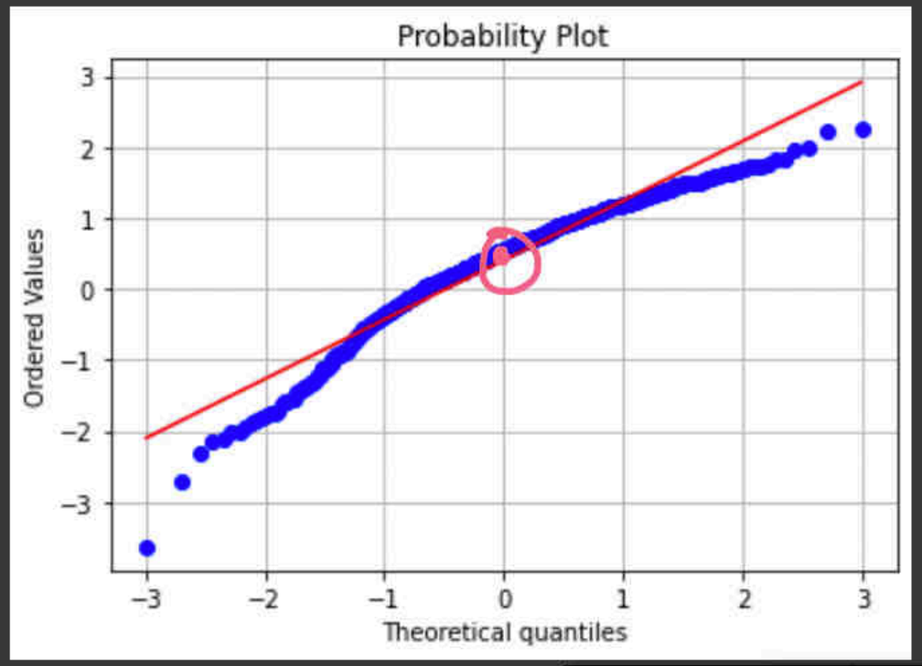
RAM Disk

[5]




```
#QQ-Plot
fig, ax1 = plt.subplots()
plt.grid()
prob = stats.probplot(x, dist=stats.norm, plot=ax1)
```

$x_1, x_2, \ldots, x_{500}$ → std → $\frac{x-\mu}{\sigma}$

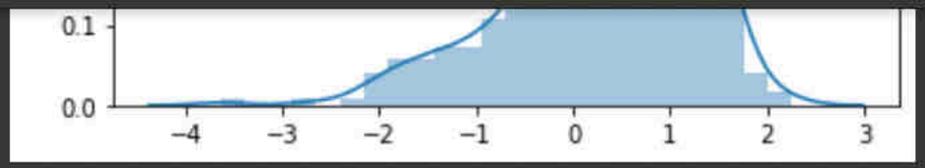


Normal $(\mu=160, \sigma=10)$

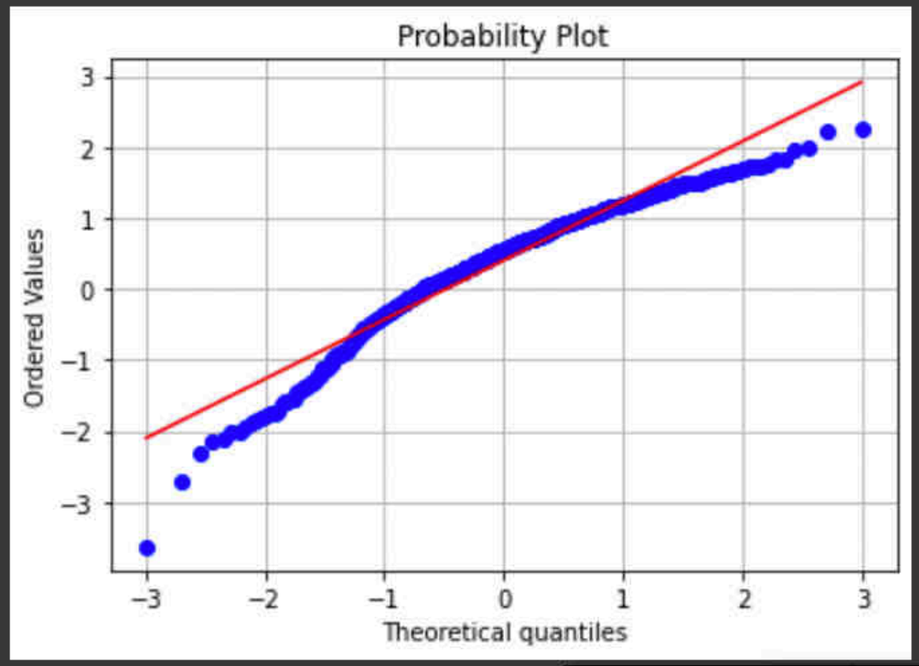
↳ $N(0,1)$

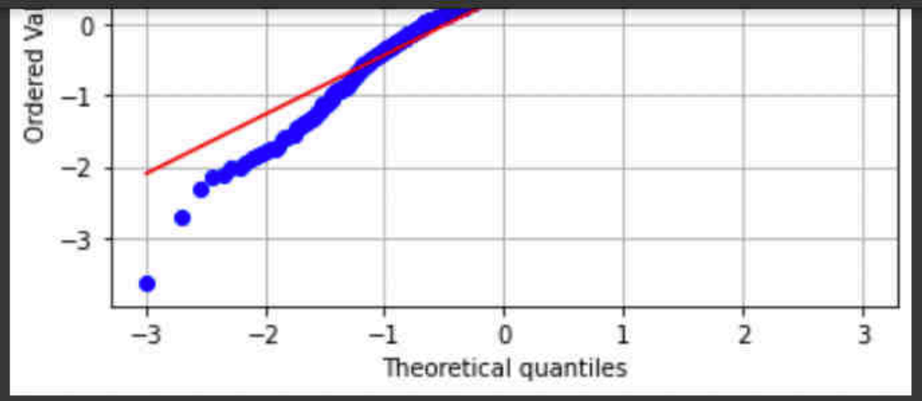
30 / 30

[5]



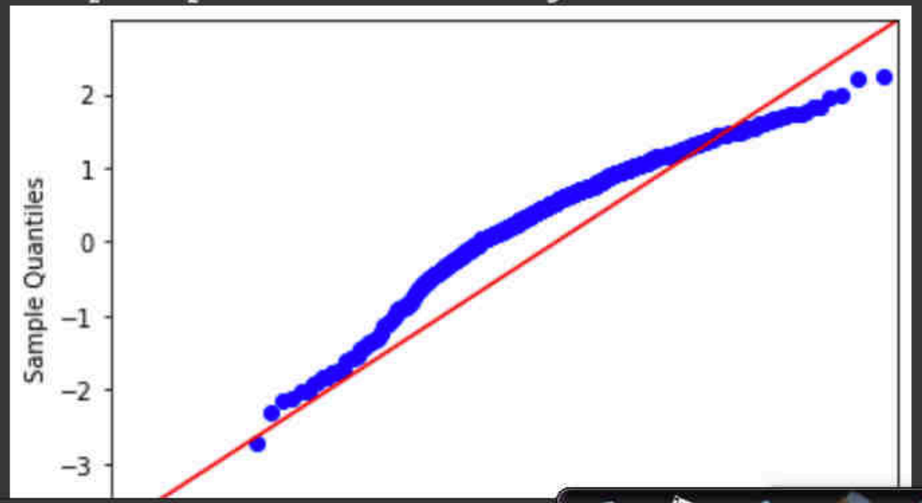

```
#QQ-Plot
fig, ax1 = plt.subplots()
plt.grid()
prob = stats.probplot(x, dist=stats.norm, plot=ax1)
```






```
import statsmodels.api as sm
fig = sm.qqplot(x, line='45')
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning: pandas.util.testing is depre
  import pandas.util.testing as tm
```



scipy.stats.probplot

```
scipy.stats.probplot(x, sparams=(), dist='norm', fit=True, plot=None,
rvalue=False)
```

[source]

Calculate quantiles for a probability plot, and optionally show the plot.

Generates a probability plot of sample data against the quantiles of a specified theoretical distribution (the normal distribution by default). `probplot` optionally calculates a best-fit line for the data and plots the results using Matplotlib or a given plot function.

Parameters: **x** : *array_like*

Sample/response data from which `probplot` creates the plot.

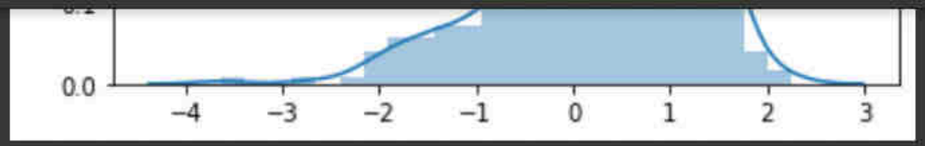
sparams : *tuple, optional*

Distribution-specific shape parameters (shape parameters plus location and scale).

dist : *str or stats.distributions instance, optional*

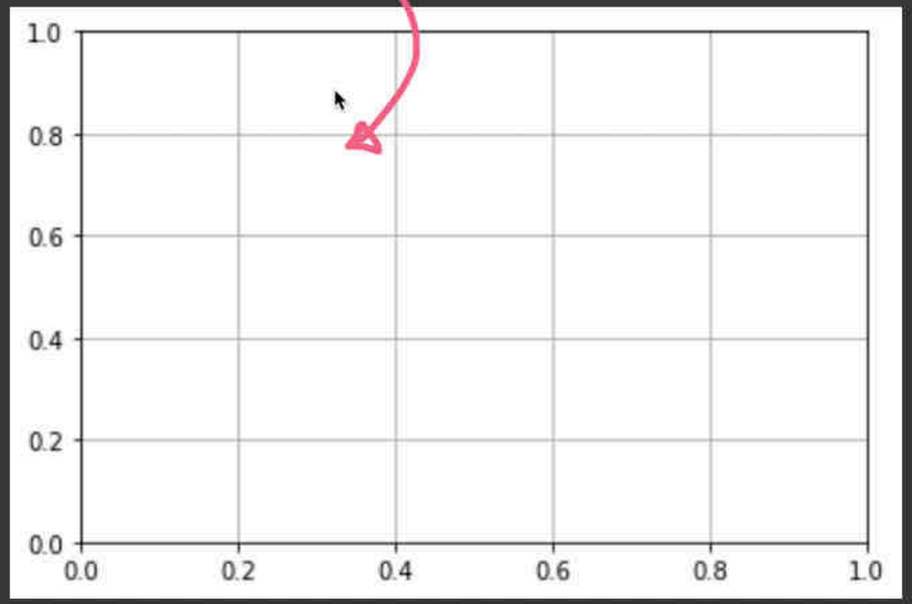
Distribution or distribution function name. The default is 'norm' for a normal probability plot. Objects that look enough like a stats.distributions instance (i.e. they have a ppf method) are also accepted.

[5]

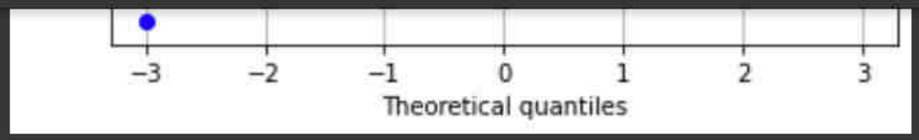



```
#QQ-Plot
fig, ax1 = plt.subplots()
plt.grid()
prob = stats.probplot(x, dist=stats.norm)
```

ax1

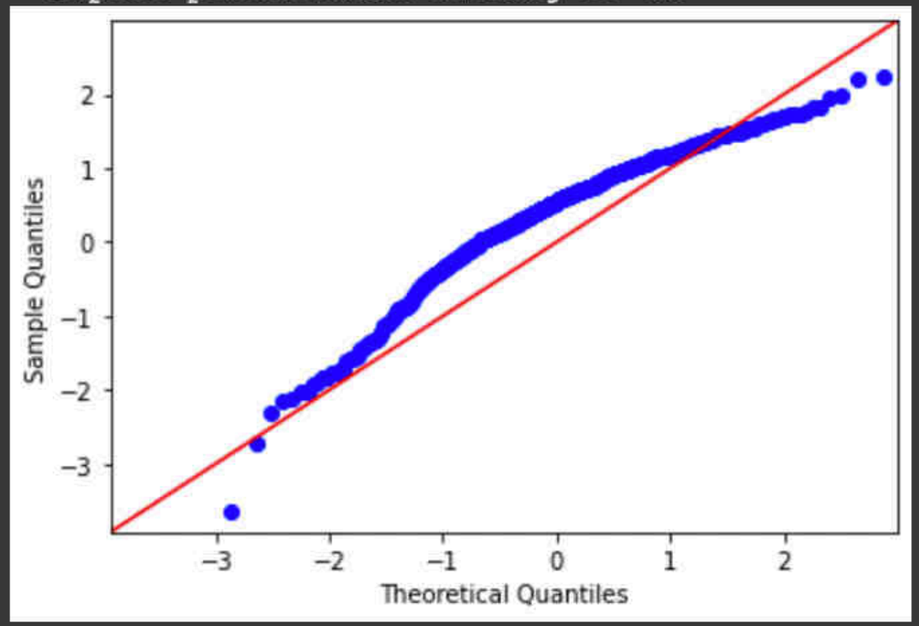


[6] import statsmodels.api as sm




```
[6] import statsmodels.api as sm
    fig = sm.qqplot(x, line='45')
```

```
/usr/local/lib/python3.7/dist-packages/statsmodels/tools/_testing.py:19: FutureWarning: pandas.util.testing is depre
  import pandas.util.testing as tm
```



$x_1, \ldots, x_{500}$

$\not\sim$ Normal

Chrome File Edit View History Bookmarks Profiles Tab Window Help 10:01

QQ-plot and CLT and Bootstra | scipy.stats.probplot — SciPy | statsmodels.graphics.gofplots

colab.research.google.com/drive/1m1Wa-7i9WknfRE9Y_aoi_5cbm5AAJBjW#scrollTo=gEeJVOFcH9WR

+ Code + Text RAM Disk

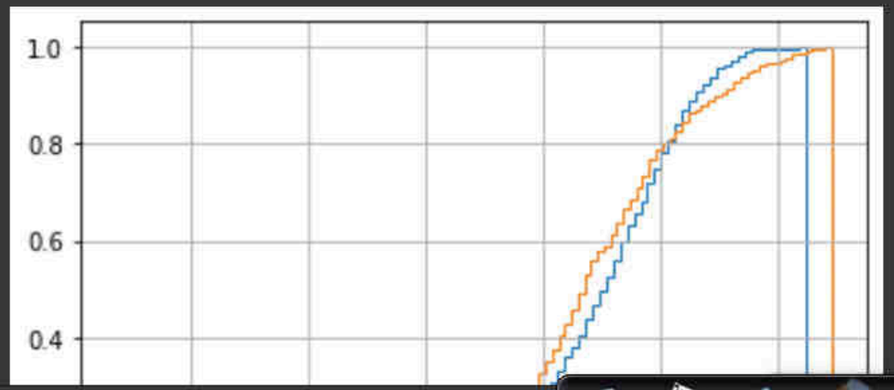
```
# Let us compare CDFs also
mu = np.mean(x)
s = np.std(x)
print(mu,s)
```

sample-mean ≈ μ

```
0.40812679178 0.8571215209422999
```

```
[ ] # normally distributed data with mean=mu, std-dev=s
    y = stats.norm.rvs(loc=mu, scale=s, size=500)
```

```
plt.grid()
a = plt.hist(x, bins=100, cumulative=True, label='CDF', density=True, histtype='step')
b = plt.hist(y, bins=100, cumulative=True, label='CDF', density=True, histtype='step')
plt.show()
```



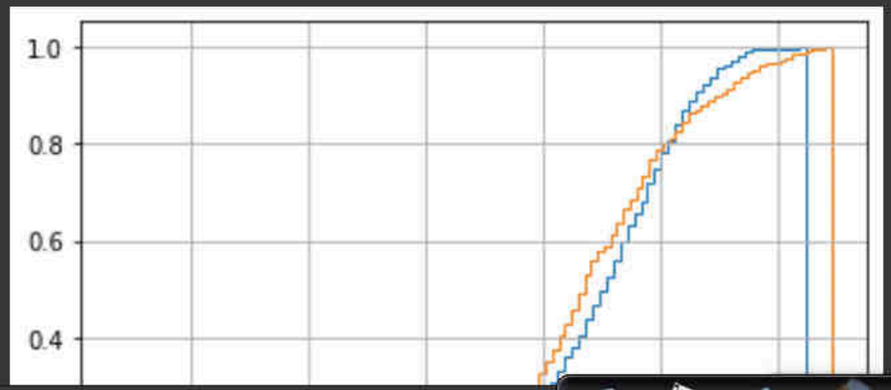
36 / 36

```
# Let us compare CDFs also
mu = np.mean(x)
s = np.std(x)
print(mu,s)
```

```
0.40812679178 0.8571215209422999
```

```
# normally distributed data with mean=mu, std-dev=s
y = stats.norm.rvs(loc=mu, scale=s, size=500)
```

```
plt.grid()
a = plt.hist(x, bins=100, cumulative=True, label='CDF', density=True, histtype='step')
b = plt.hist(y, bins=100, cumulative=True, label='CDF', density=True, histtype='step')
plt.show()
```



```
# Let us compare CDFs also
mu = np.mean(x)
s = np.std(x)
print(mu,s)
```

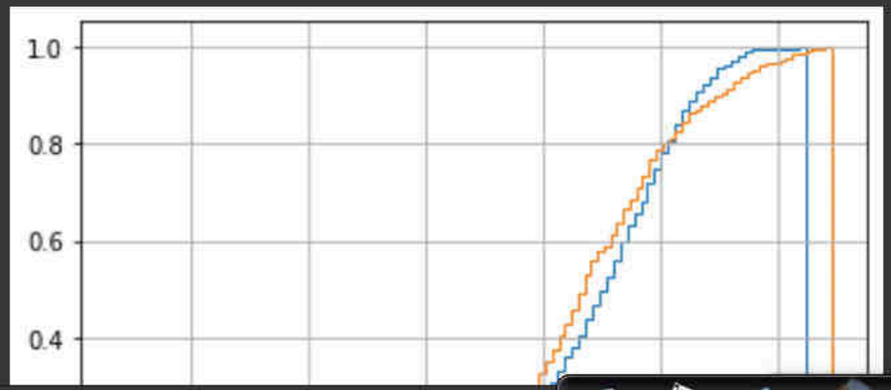
```
0.40812679178 0.8571215209422999
```

X: obs-data
$x_1, \ldots, x_{500}$
μ, σ : using sample data

```
# normally distributed data with mean=mu, std-dev=s
y = stats.norm.rvs(loc=mu, scale=s, size=500)
```

→ Th. dist

```
plt.grid()
a = plt.hist(x, bins=100, cumulative=True, label='CDF', density=True, histtype='step')
b = plt.hist(y, bins=100, cumulative=True, label='CDF', density=True, histtype='step')
plt.show()
```

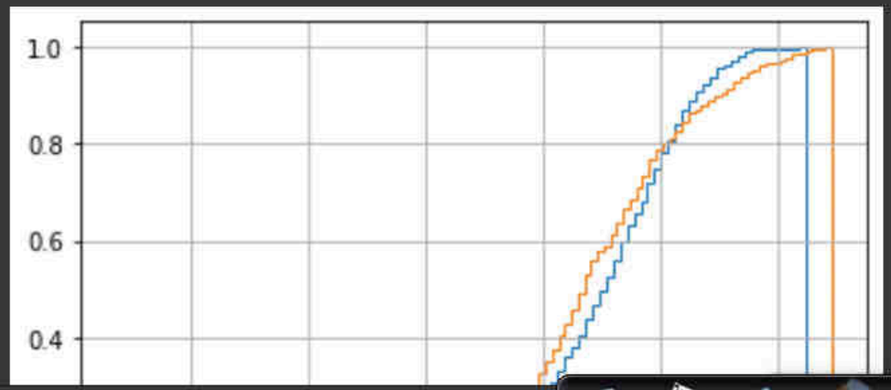


```
# Let us compare CDFs also
mu = np.mean(x)
s = np.std(x)
print(mu,s)
```

```
0.40812679178 0.8571215209422999
```

```
# normally distributed data with mean=mu, std-dev=s
y = stats.norm.rvs(loc=mu, scale=s, size=500)
```

```
plt.grid()
a = plt.hist(x, bins=100, cumulative=True, label='CDF', density=True, histtype='step')
b = plt.hist(y, bins=100, cumulative=True, label='CDF', density=True, histtype='step')
plt.show()
```



Chrome File Edit View History Bookmarks Profiles Tab Window Help 10:03

QQ-plot and CLT and Bootstra | scipy.stats.probplot — SciPy | statsmodels.graphics.gofplots

colab.research.google.com/drive/1m1Wa-7i9WknfRE9Y_aoi_5cbm5AAJBjW#scrollTo=gEeJVOFcH9WR

+ Code + Text

RAM Disk

```
# Let us compare CDFs also
mu = np.mean(x)
s = np.std(x)
print(mu,s)
```

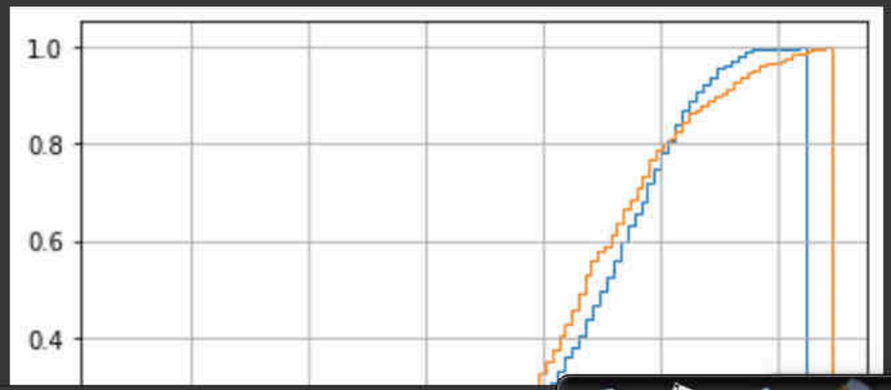
$x_1 \ldots x_{500}$

```
0.40812679178 0.8571215209422999
```

```
[ ] # normally distributed data with mean=mu, std-dev=s
    y = stats.norm.rvs(loc=mu, scale=s, size=500)
```

$y_1 \ldots y_{500}$

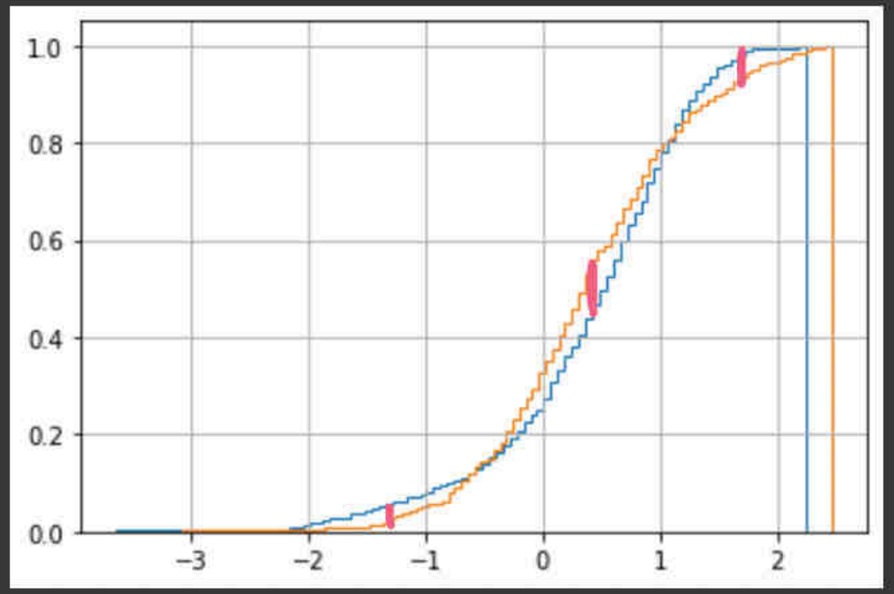
```
[ ] plt.grid()
    a = plt.hist(x, bins=100, cumulative=True, label='CDF', density=True, histtype='step')
    b = plt.hist(y, bins=100, cumulative=True, label='CDF', density=True, histtype='step')
    plt.show()
```



40 / 40

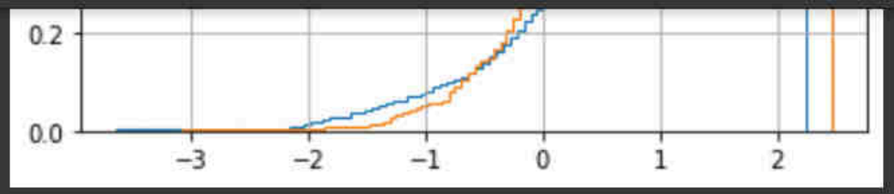
```
# normally distributed data with mean=mu, std-dev=s
y = stats.norm.rvs(loc=mu, scale=s, size=500)
```

```
plt.grid()
a = plt.hist(x, bins=100, cumulative=True, label='CDF', density=True, histtype='step')
b = plt.hist(y, bins=100, cumulative=True, label='CDF', density=True, histtype='step')
plt.show()
```



more gap → more likely the X ≠ Y

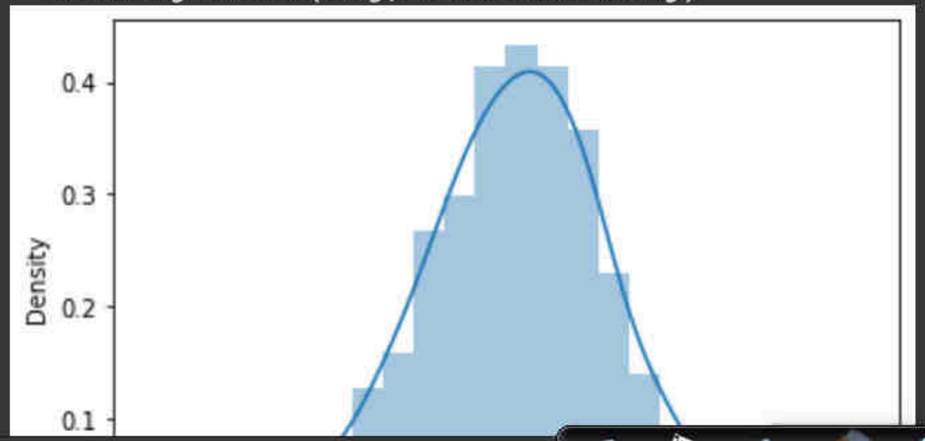
```
# case 2: QQ Plot for gaussian disb itself
x = stats.norm.rvs(loc=0, scale=1, size=500) # change 500 to 5000
sns.distplot(x)
```



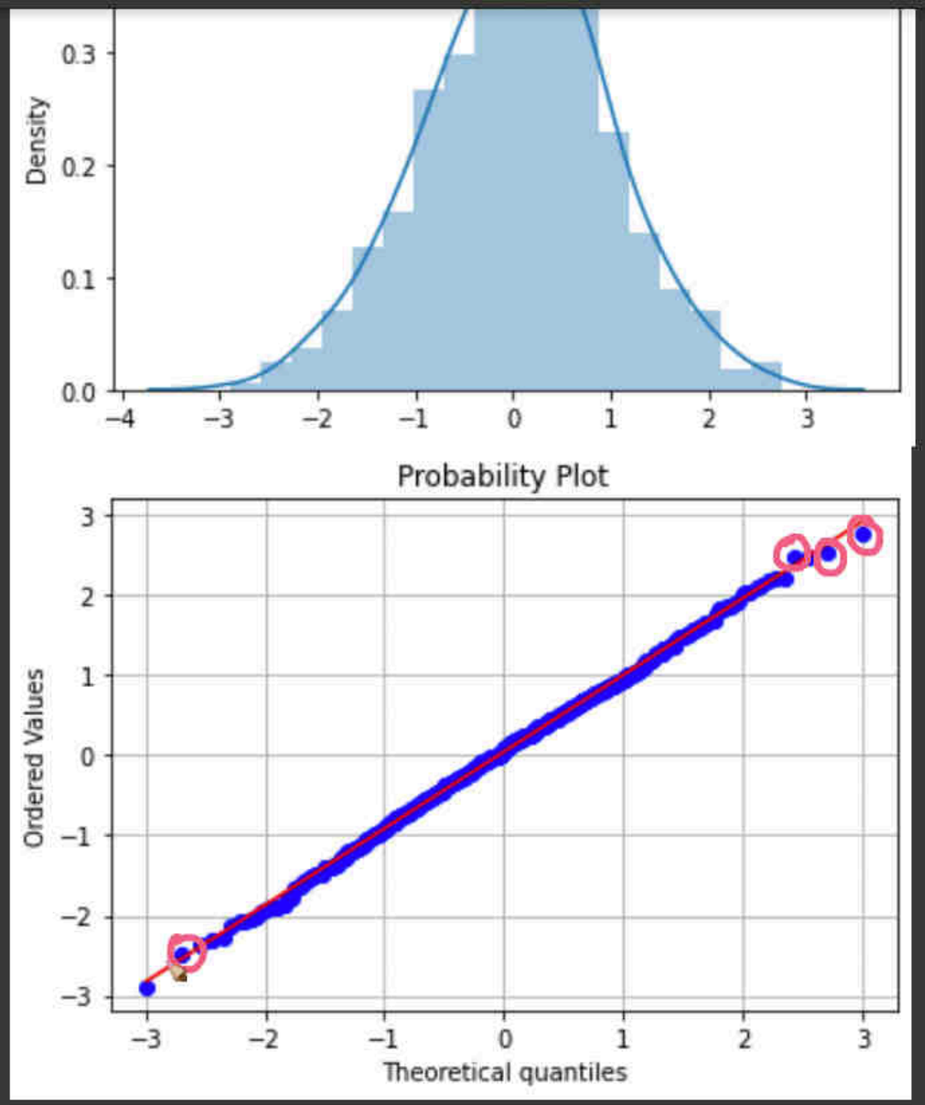

```
# case 2: QQ Plot for gaussian disb itself
x = stats.norm.rvs(loc=0, scale=1, size=500) # change 500 to 5000
sns.distplot(x)

#QQ-Plot
fig, ax1 = plt.subplots()
plt.grid()
prob = stats.probplot(x, dist=stats.norm, plot=ax1)
```

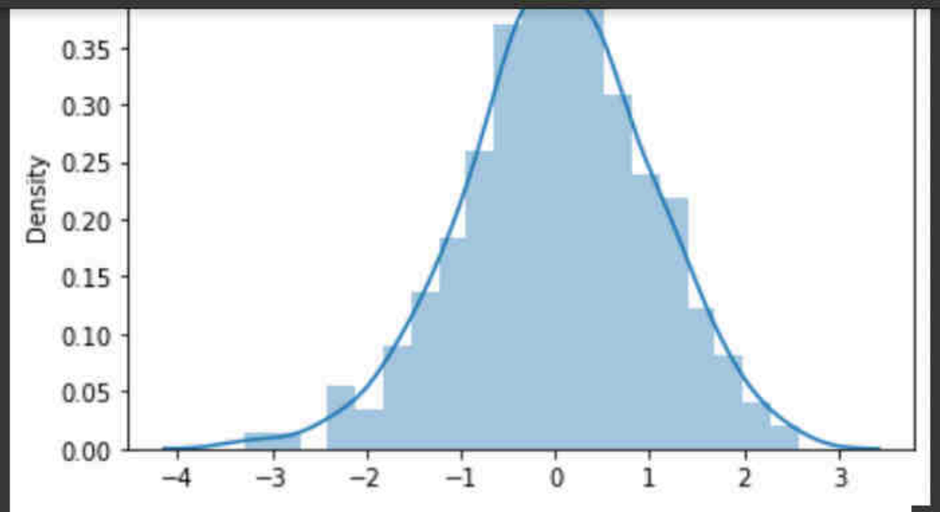
$x_1, \ldots x_{500}$

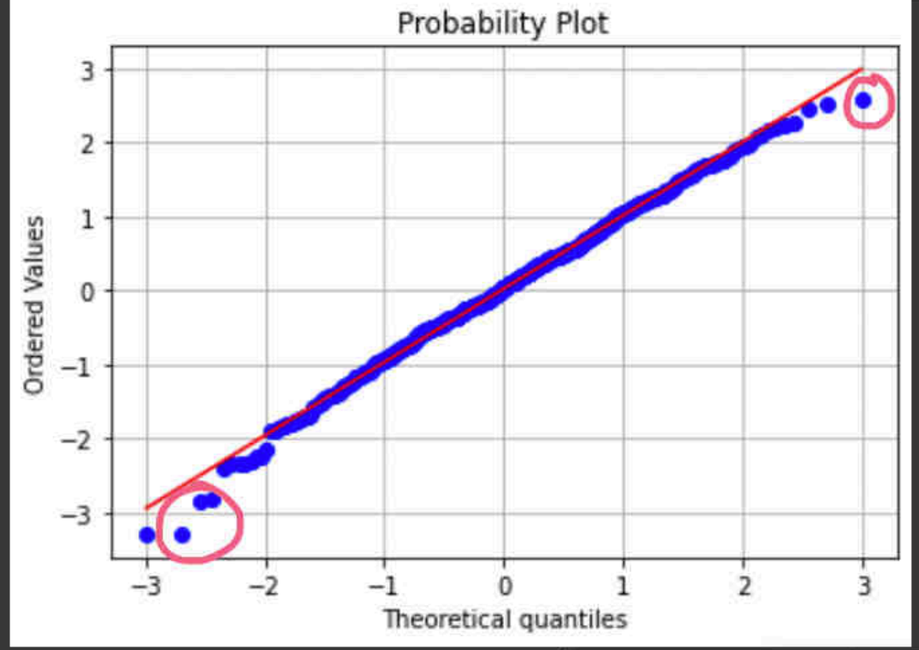
```
/usr/local/lib/python3.7/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated func
  warnings.warn(msg, FutureWarning)
```



QQ plot


Density
0.3
0.2
0.1
0.0
−4
−3
−2
−1
0
1
2
3
Probability Plot
Ordered Values
Theoretical quantiles



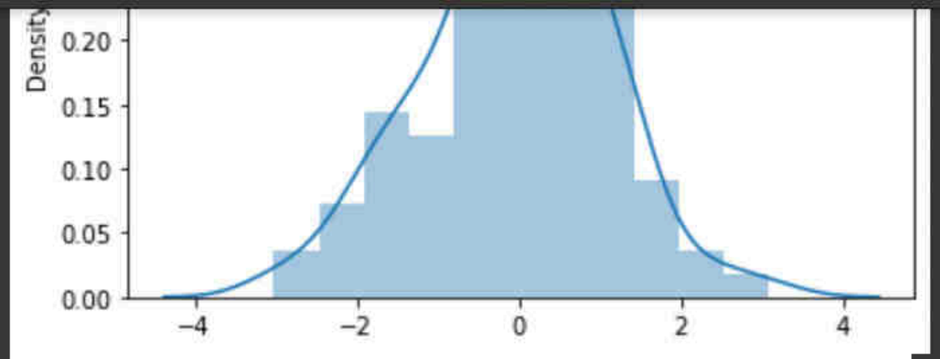


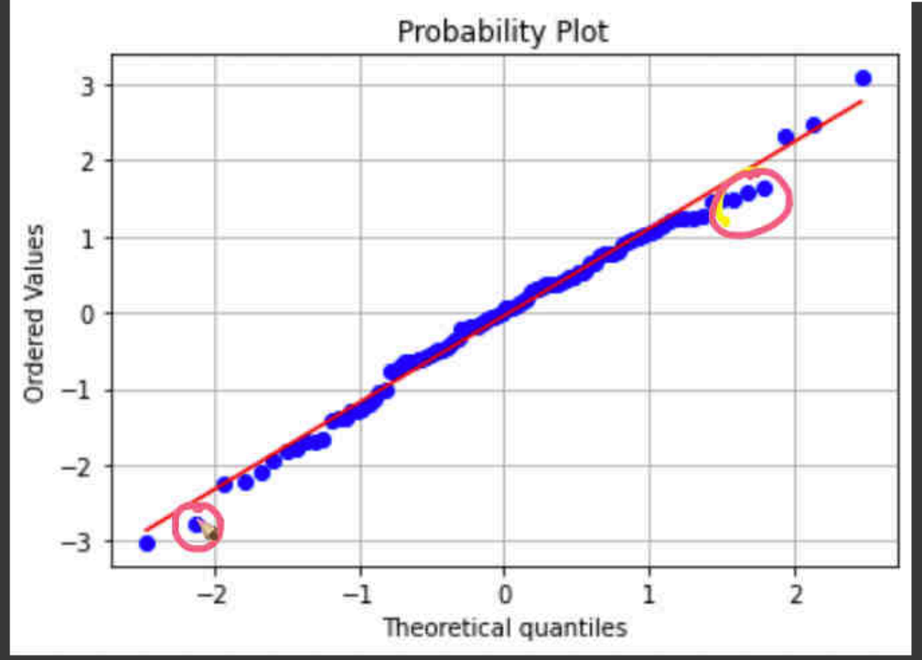
$n \downarrow$:

$n =$ 500 obs

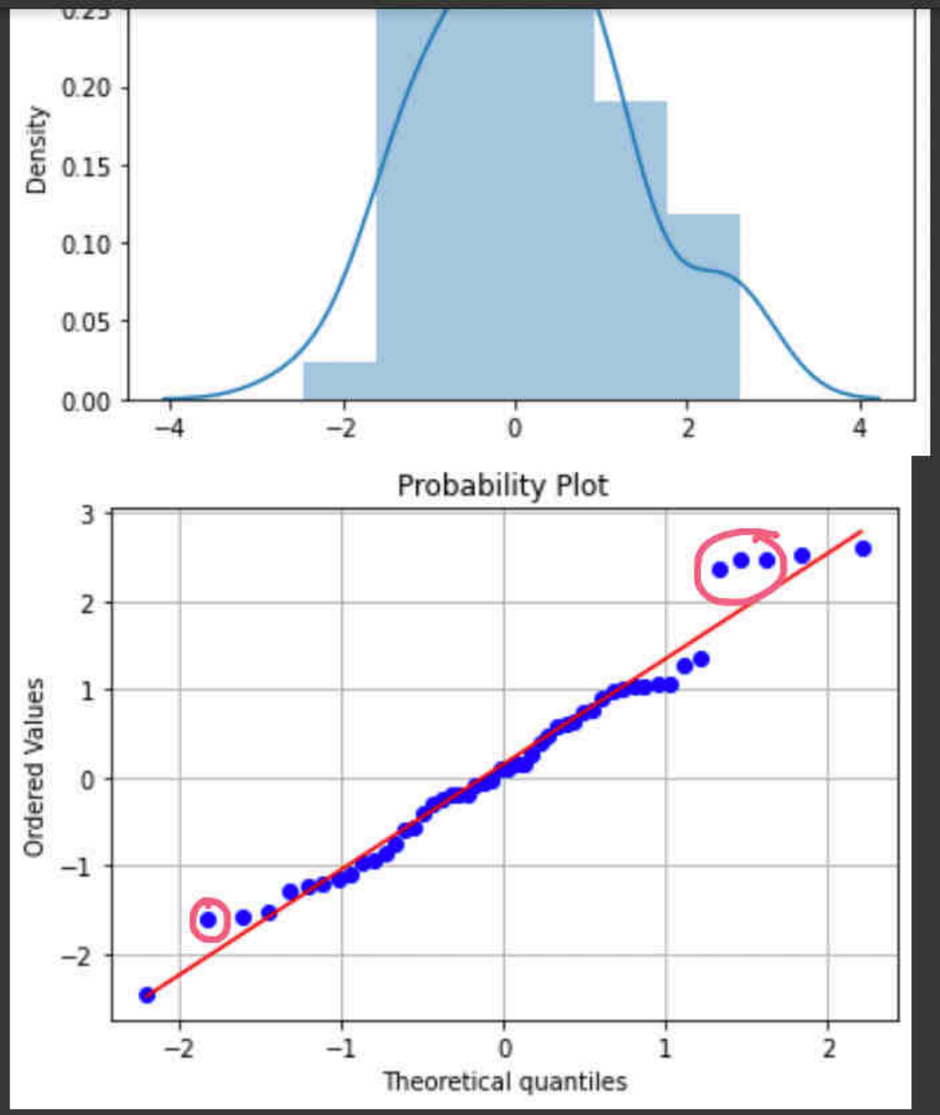
$n \uparrow$

as $n \to \infty$

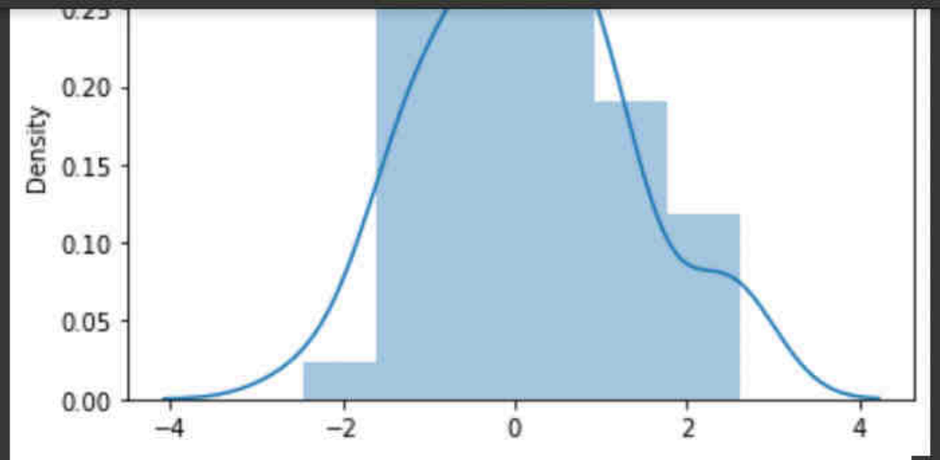


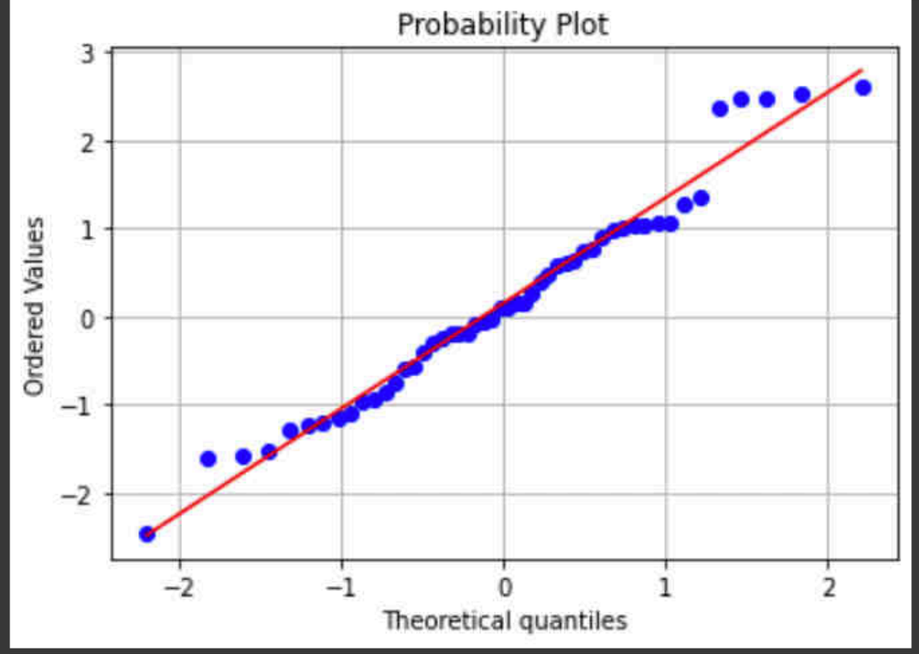


Uber Data

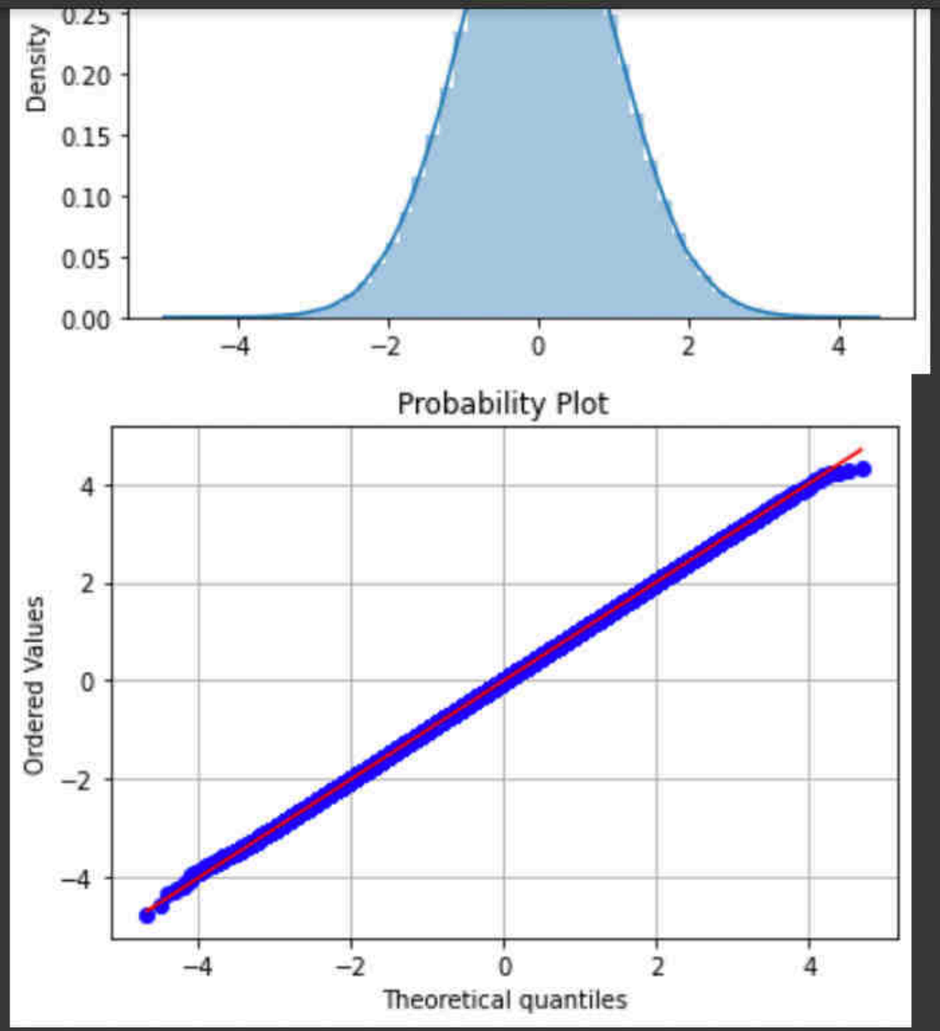


qqplot utility





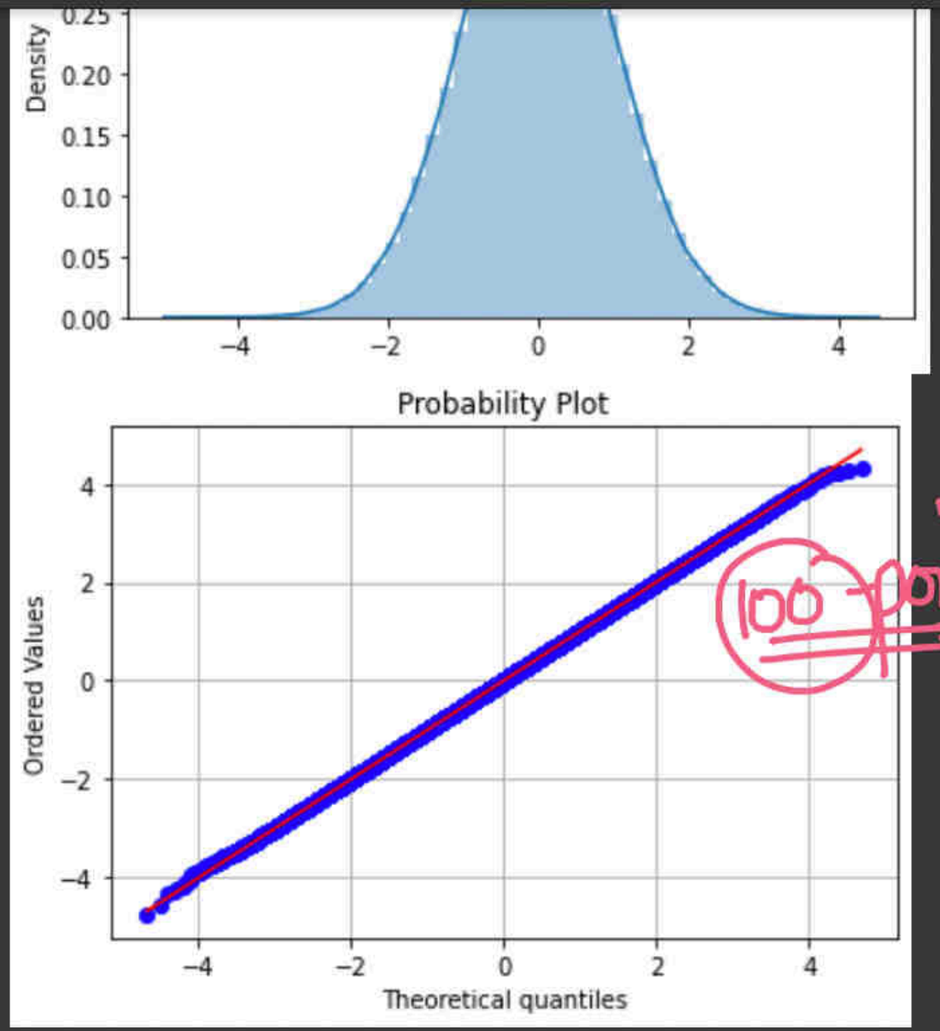
as $n \to \infty$ QQplot will work perfectly...



$n \uparrow$

$n \uparrow \Rightarrow$ QQ-plot will behave perfect

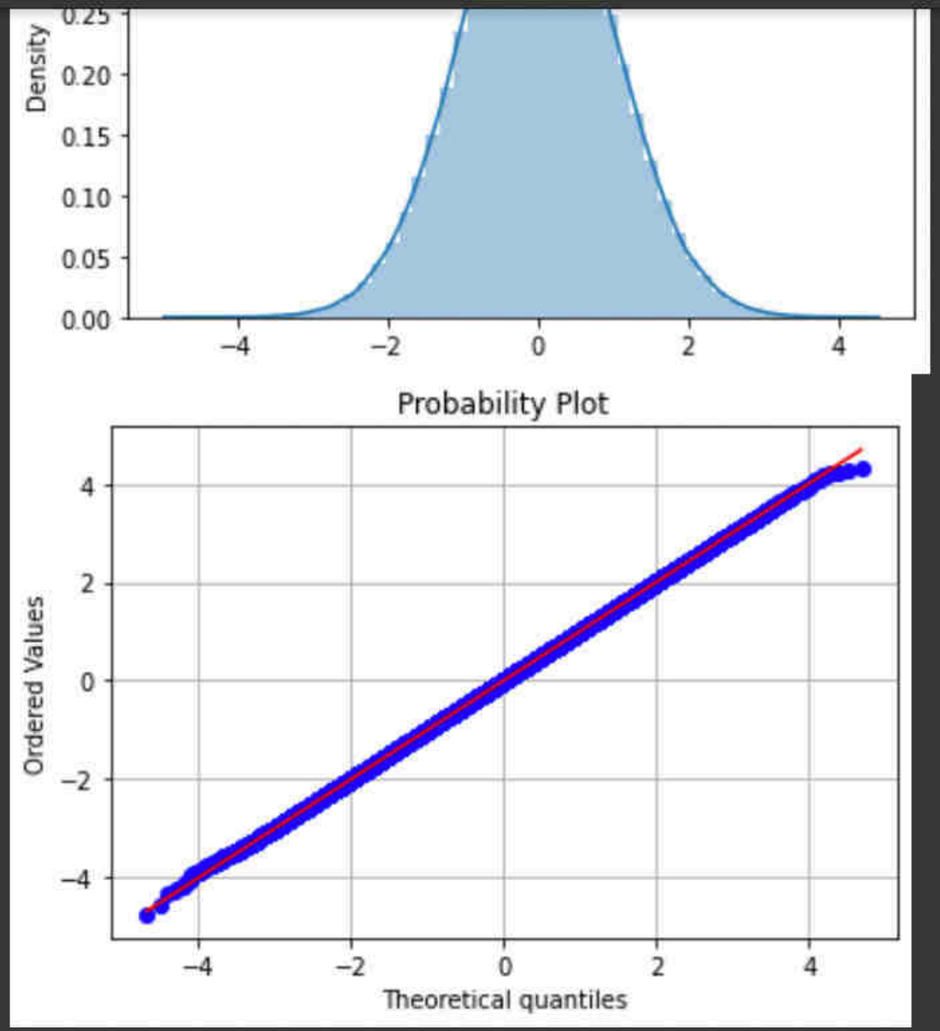
observation $\sim N(\cdot,\cdot)$



500 → 50

$n \downarrow$

Po·1
Po·2
⋮



n is small → KS-test
SW-test
AD-test

Programming zone

X [stats-scipy
stats models] ←

↳ WAF to plot QQ-plot using
scatter plot & line plot

20-25 min

[f(x, dist ...)
:

WAF to plot PDF from scratch

20 - min

✓ QQ-plot from scratch on your own

implement func or class in popular lib

→ JOIN pandas...

Confidence Intervals

Covid

were randomly assigned to receive either remdesivir (200 mg loading dose on day 1, followed by 100 mg daily for up to 9 additional days) or placebo for up to 10 days. The primary outcome was the time to recovery, defined by either discharge from the hospital or hospitalization for infection-control purposes only.

RESULTS

A total of 1062 patients underwent randomization (with 541 assigned to remdesivir and 521 to placebo). Those who received remdesivir had a median recovery time of 10 days (95% confidence interval [CI], 9 to 11), as compared with 15 days (95% CI, 13 to 18) among those who received placebo (rate ratio for recovery, 1.29; 95% CI, 1.12 to 1.49; P<0.001, by a log-rank test). In an analysis that used a proportional-odds model with an eight-category ordinal scale, the patients who received remdesivir were found to be more likely than those who received placebo to have clinical improvement at day 15 (odds ratio, 1.5; 95% CI, 1.2 to 1.9, after adjustment for actual disease severity). The Kaplan–Meier estimates of mortality were 6.7% with remdesivir and 11.9% with placebo by day 15 and 11.4% with remdesivir and 15.2% with placebo by day 29 (hazard ratio, 0.73; 95% CI, 0.52 to 1.03). Serious adverse events were reported in 131 of the 532 patients who received remdesivir (24.6%) and in 163 of the 516 patients who received placebo (31.6%).

CONCLUSIONS

Our data show that remdesivir was superior to placebo in shortening the time to recovery in adults who were hospitalized with Covid-19 and had evidence of lower respiratory tract infection. (Funded by the National Institute of Allergy and Infectious Diseases and others; ACTT-1 ClinicalTrials.gov number, NCT04280705.)

QUICK TAK
Remdesiv

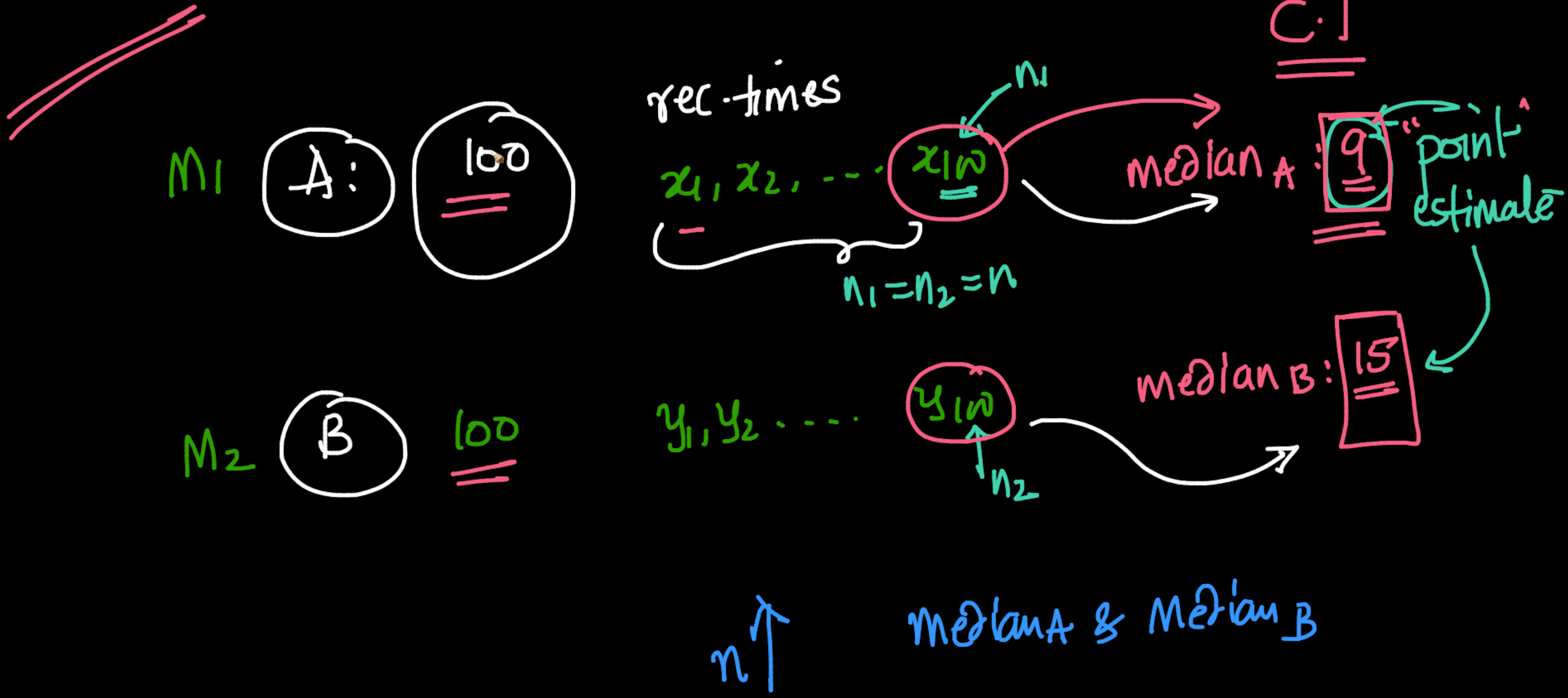
Related Articles

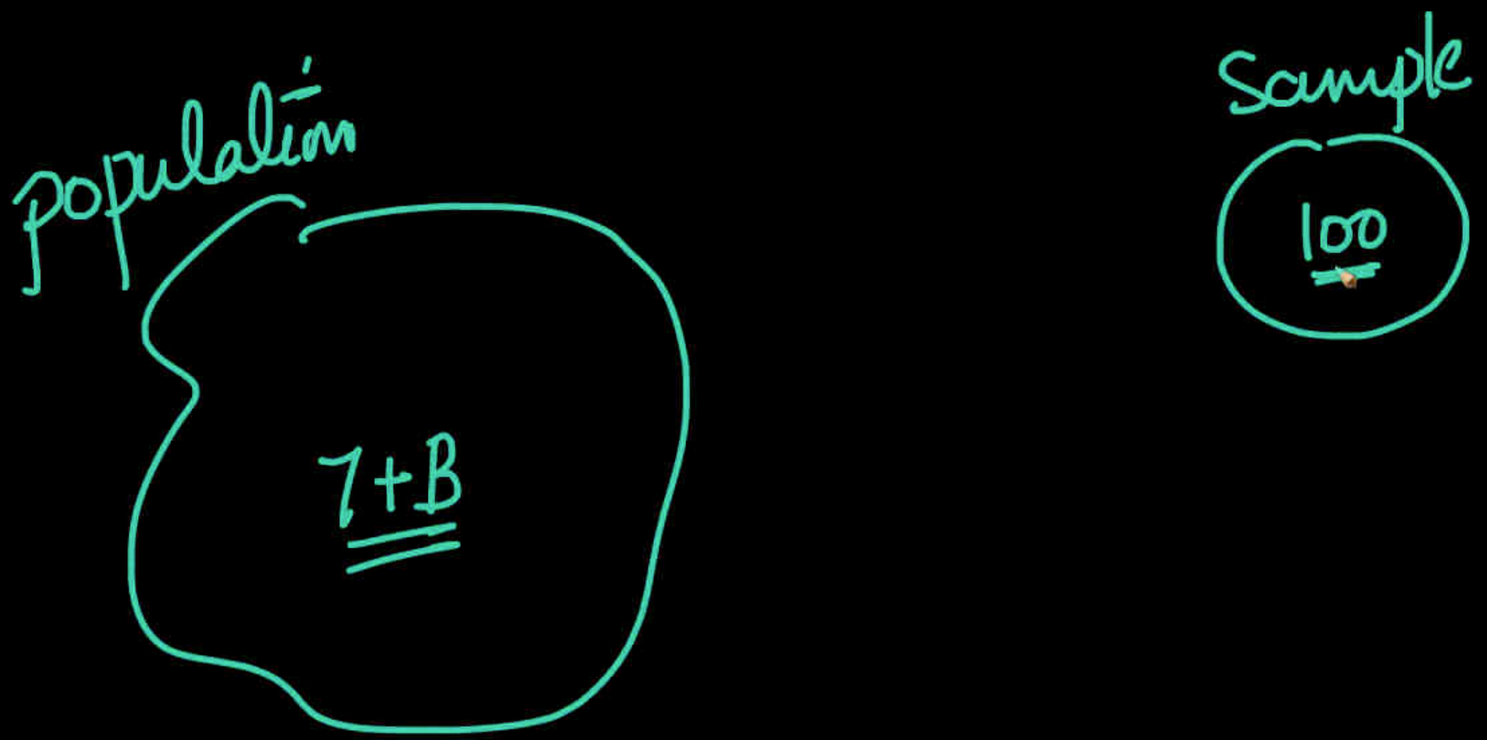
EDITORIAL NOV 5, 2020
Remdesivir — An Important First Step
C. Dolin and M.S. Hirsch

ORIGINAL ARTICLE NOV 5, 2020
Remdesivir for 5 or 10 Days in Patients with Severe Covid-19
J.D. Goldman and Others

CORRESPONDENCE SEP 3, 2020
Remdesivir for the Treatment of Covid-19 — Preliminary Report

C.1



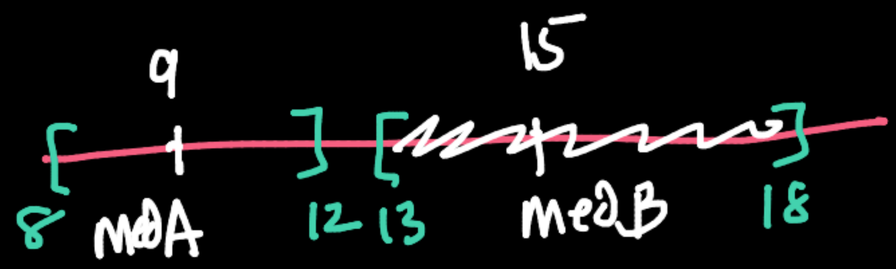


(A) sample changes ⟶ slightly diff from 9

(B) " " ⟶ " " is?

95% C. intervals

Code-idea



95% of times

✓ median A: [8 , 12]

median B: 95% [13, 18]

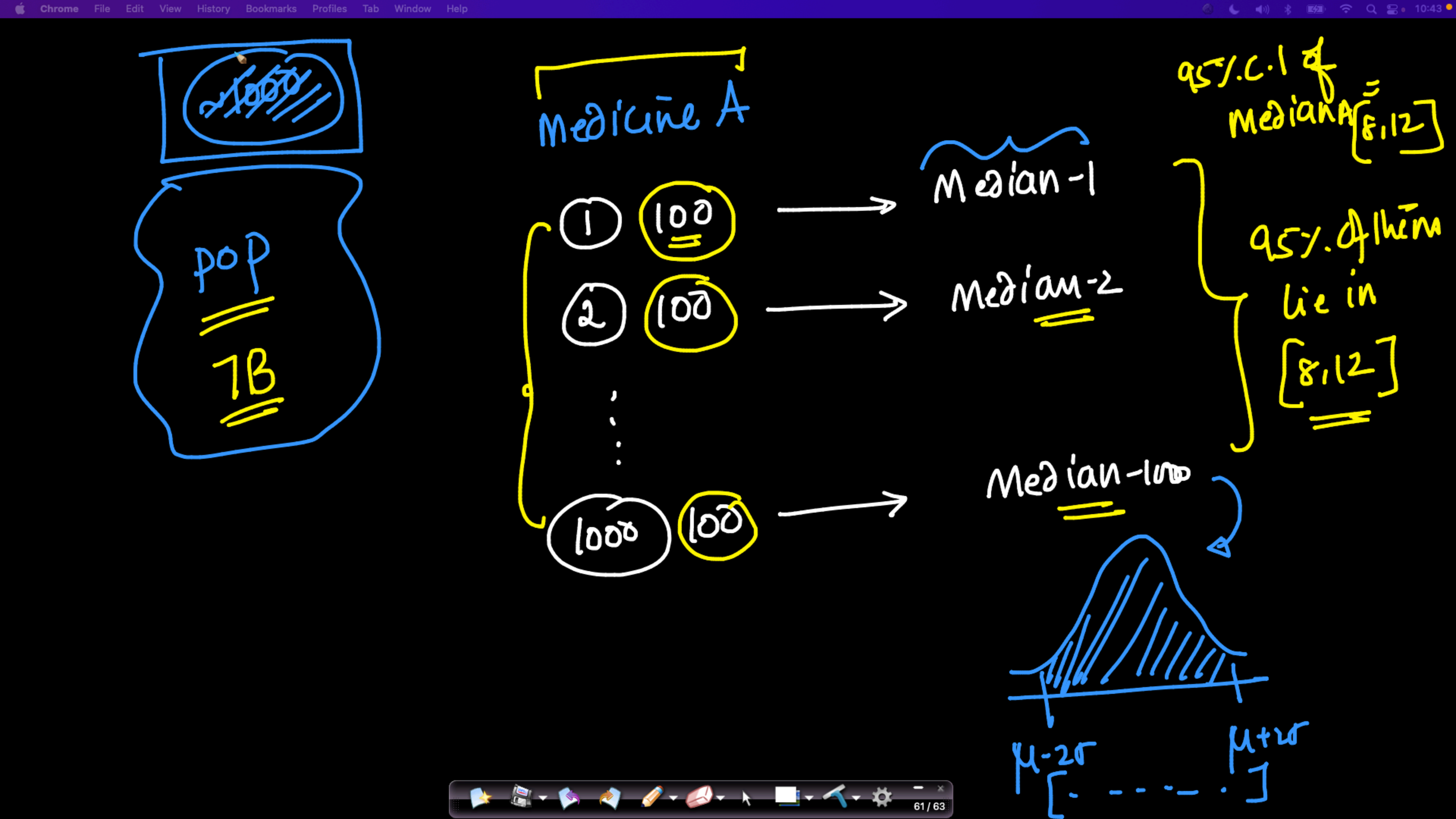
let

How to compute the C.I

- bootstrapping
- CLT

95% is commonly used

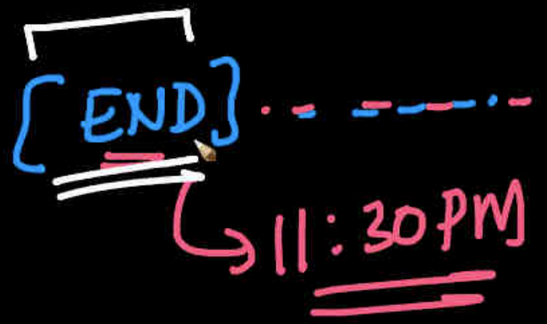
99% C.I ✓


~1000
pop
7B
Medicine A
1
100
2
100
1000
100
Median-1
Median-2
Median-1000
95% C.I of median = [8,12]
95% of them lie in [8,12]
μ-2σ
μ+2σ

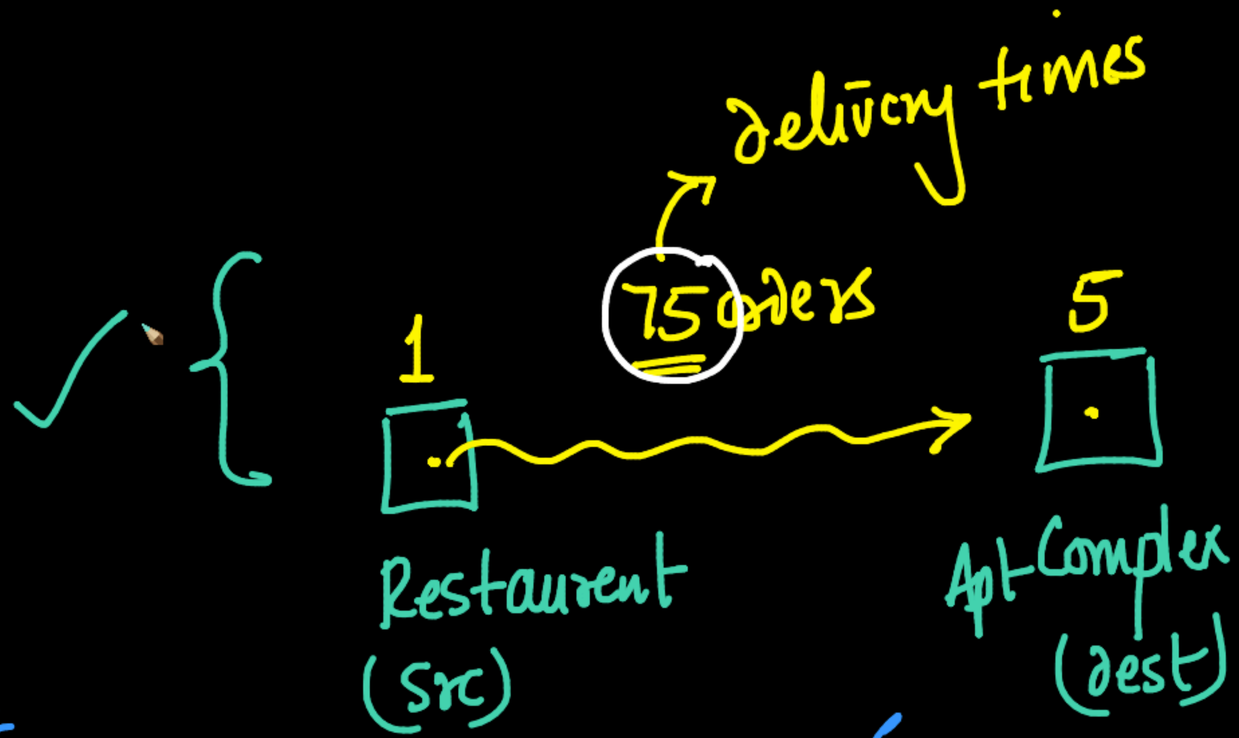
DS @ Amazon → population

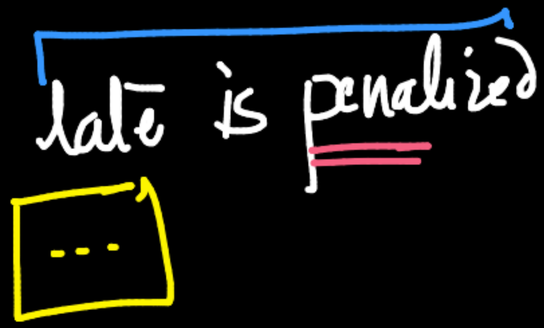
GoI → bank savings ...

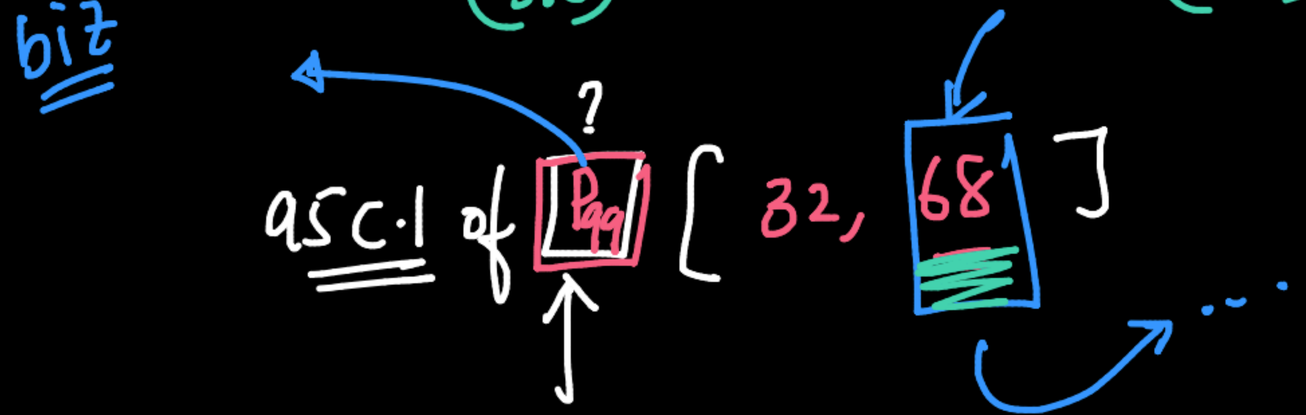
(Let) mediana

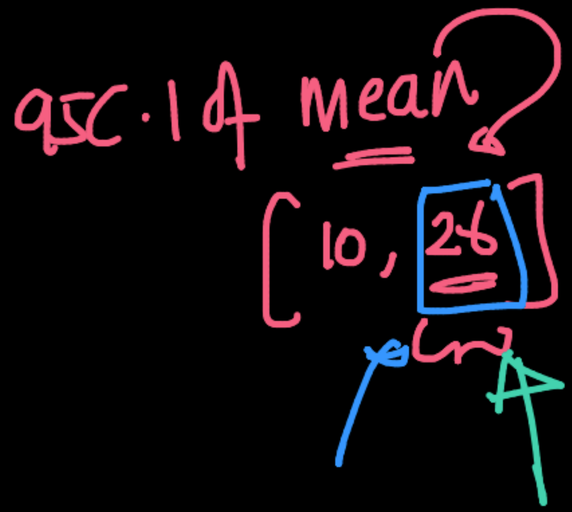


Uber Eats









95% C.I on means → CLT ; bootstrapping

(*) { medians, P99, P95 } — bootstrapping

Bootstrapping:

rec. times:

$x_1, x_2 \ldots x_{100}$; sample of $n = 100$ pts

Assumption: disb of Sample $\approx$ disb of the population

✓ $n \to \infty$

→ random-sample & n is not too small

Pseudo: code

medians = [ ]

$r = 1000$

for $i = 1$ to r

{

- randomly sample m points from $x_1, \ldots x_{100=n}$ with replacement $\times r$

- median of m points → medians[i]

}

- sort 1000 medians

- pick the $p_{2.5}$ & $p_{97.5}$ of the 1000 medians

q.s.c.i of median

$m = n$

$m \leq n = 100$

with "replacement"

1, 6, 6, 12, 3

bootstrapped samples

6, 8, 12, 6, 3

without replacement

1, 6, 8, 12, 3

n = 5

6, 8, 12, 3, 1

m = n = 5

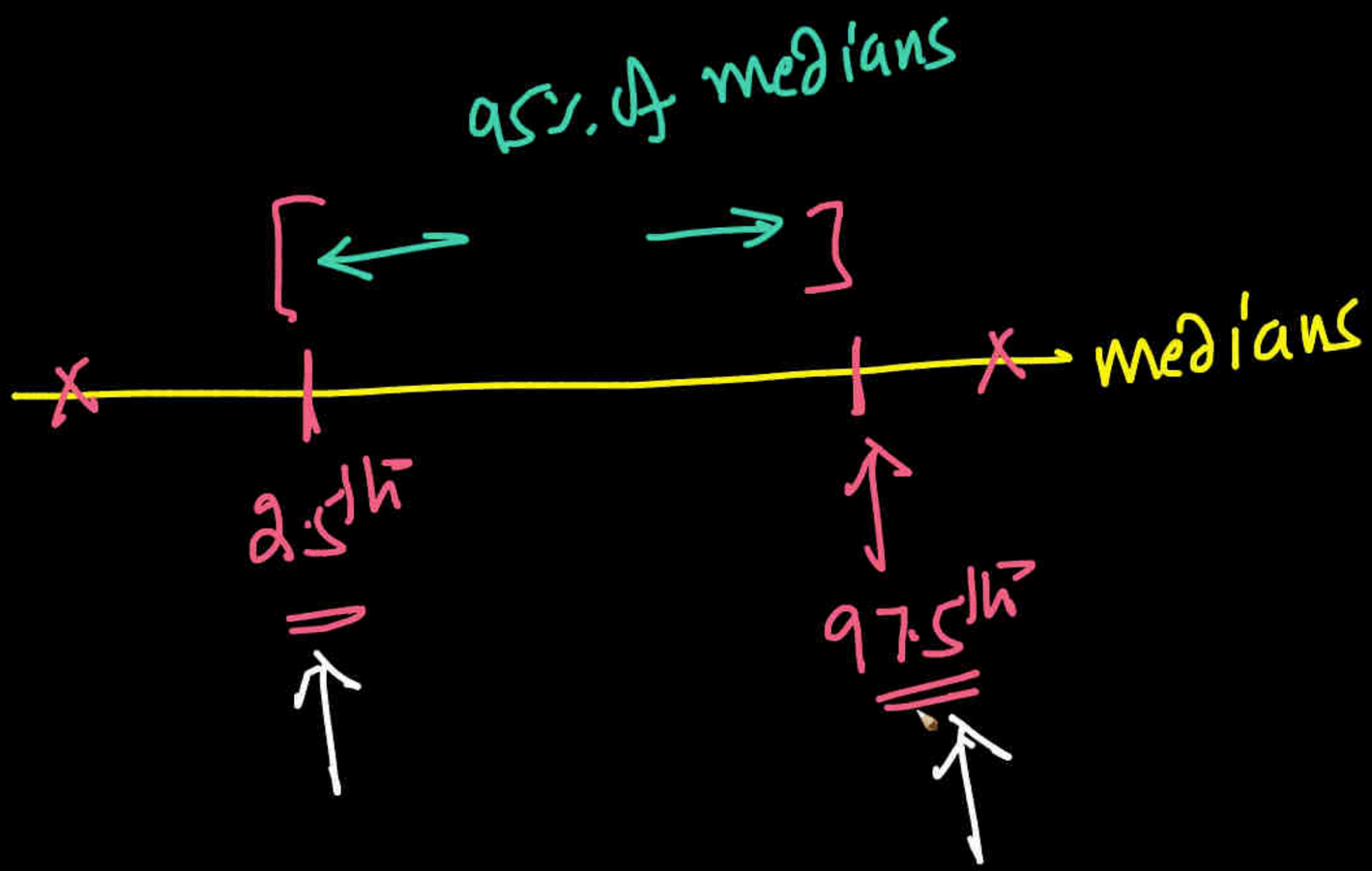
12, 6, 3, 1, 8

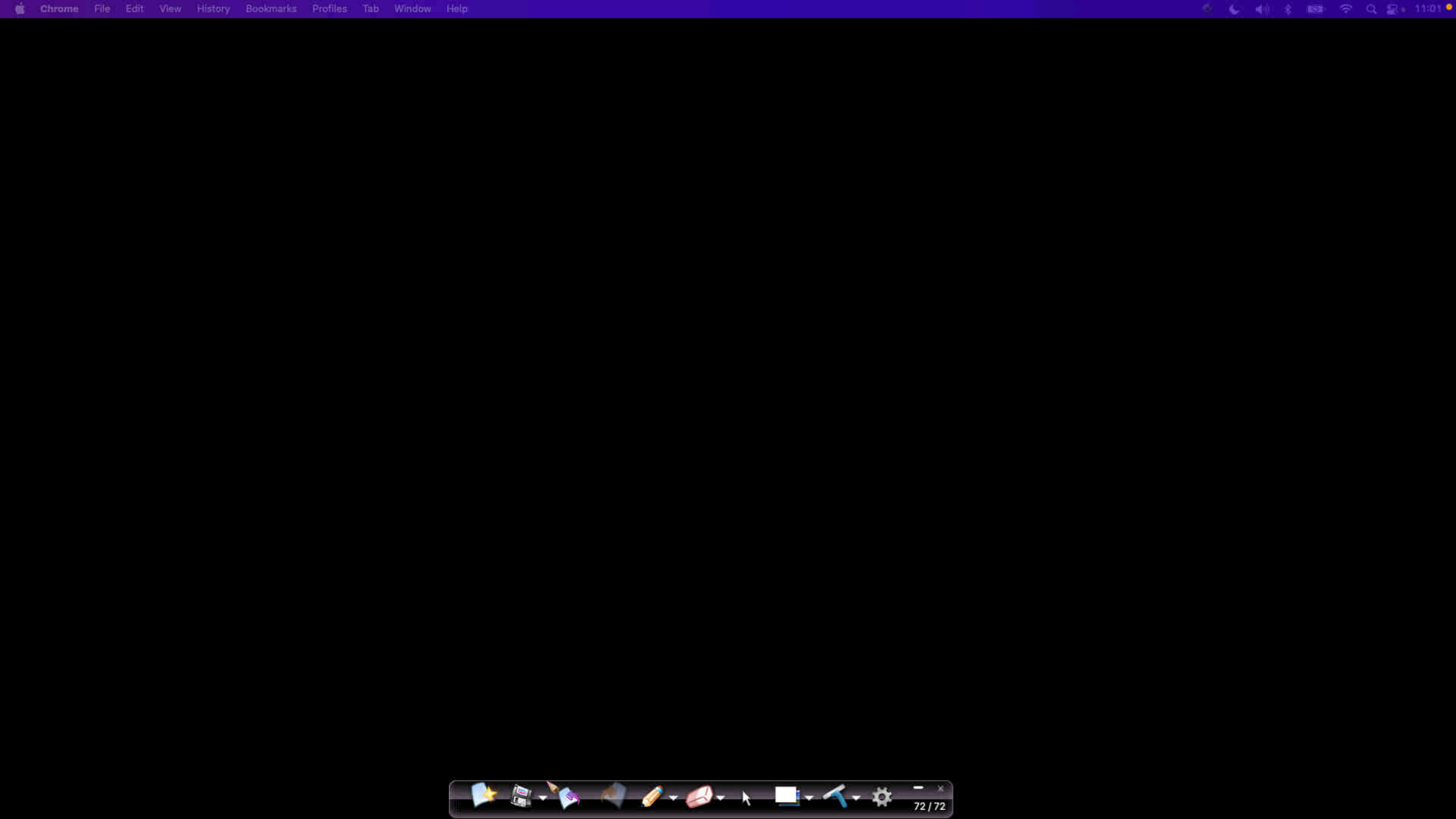
with replacement

can do this

$m > n$

$m < n$ or $m = n$




Chrome File Edit View History Bookmarks Profiles Tab Window Help
11:01
72 / 72

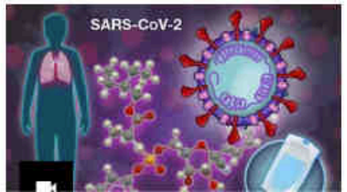
recovery, defined by either discharge from the hospital or hospitalization for infection-control purposes only.

RESULTS

A total of 1062 patients underwent randomization (with 541 assigned to remdesivir and 521 to placebo). Those who received remdesivir had a median recovery time of 10 days (95% confidence interval [CI], 9 to 11), as compared with 15 days (95% CI, 13 to 18) among those who received placebo (rate ratio for recovery, 1.29; 95% CI, 1.12 to 1.49; P<0.001, by a log-rank test). In an analysis that used a proportional-odds model with an eight-category ordinal scale, the patients who received remdesivir were found to be more likely than those who received placebo to have clinical improvement at day 15 (odds ratio, 1.5; 95% CI, 1.2 to 1.9, after adjustment for actual disease severity). The Kaplan–Meier estimates of mortality were 6.7% with remdesivir and 11.9% with placebo by day 15 and 11.4% with remdesivir and 15.2% with placebo by day 29 (hazard ratio, 0.73; 95% CI, 0.52 to 1.03). Serious adverse events were reported in 131 of the 532 patients who received remdesivir (24.6%) and in 163 of the 516 patients who received placebo (31.6%).

CONCLUSIONS

Our data show that remdesivir was superior to placebo in shortening the time to recovery in adults who were hospitalized with Covid-19 and had evidence of lower respiratory tract infection. (Funded by the National Institute of Allergy and Infectious Diseases and others; ACTT-1 ClinicalTrials.gov number, NCT04280705.)



QUICK TAKE

Remdesivir for the Treatment of Covid-19 — Final Report

01:45

73 / 73

Related Articles

EDITORIAL NOV 5, 2020
Remdesivir — An Important First Step
R. Dolin and M.S. Hirsch

ORIGINAL ARTICLE NOV 5, 2020
Remdesivir for 5 or 10 Days in Patients with Severe Covid-19
J.D. Goldman and Others

CORRESPONDENCE SEP 3, 2020
Remdesivir for the Treatment of Covid-19 — Preliminary Report

$\{ n \uparrow$
$m \approx n$

median

10

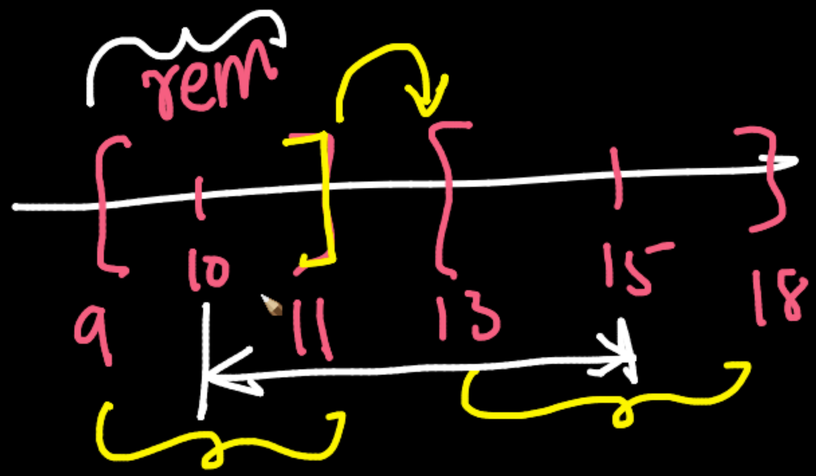
rem: [9, 11]

placeb: [13, 18]

500

15

[[ ]]



rem

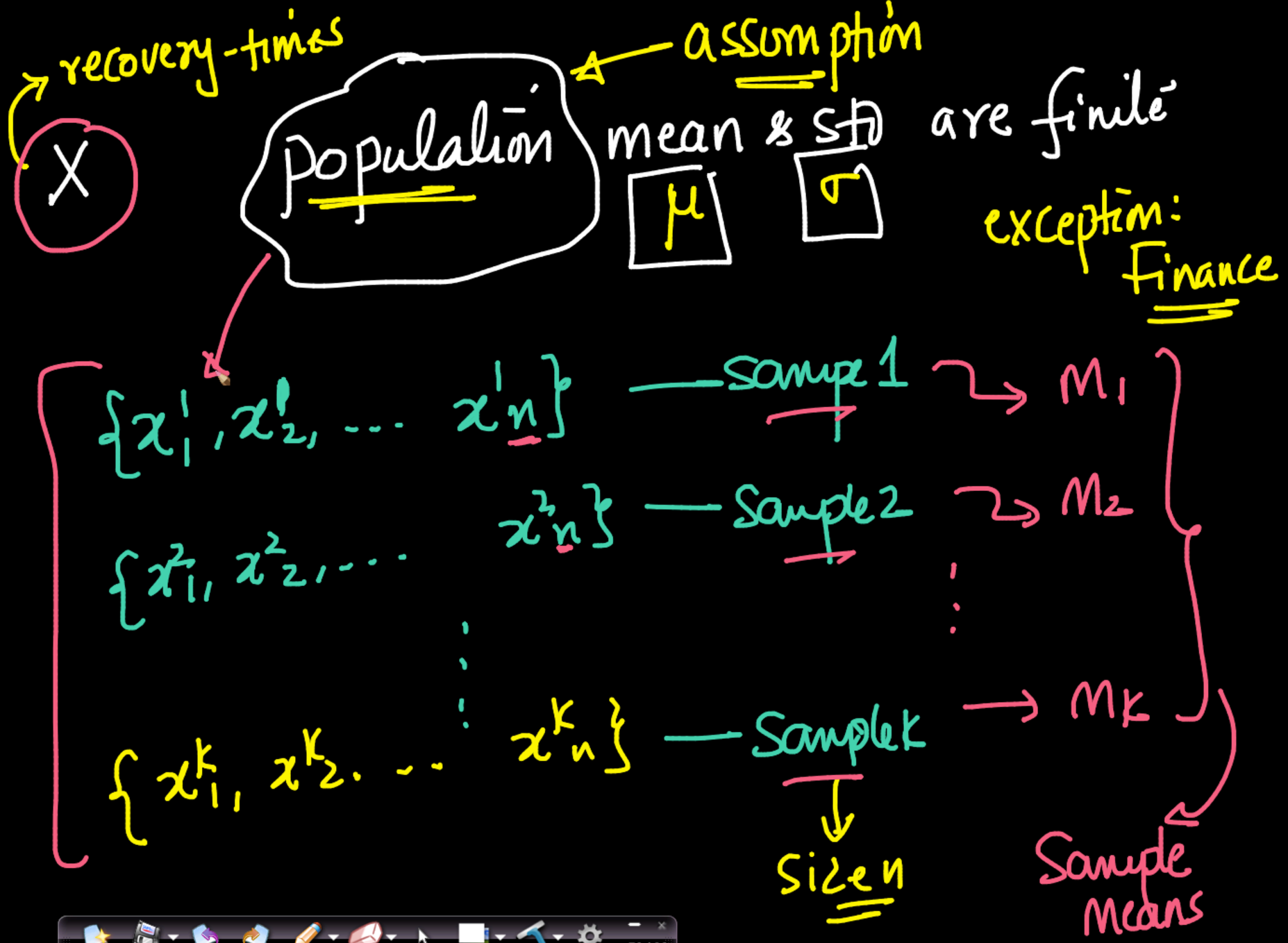
$n = 500 \rightarrow$ [9 10 11]

$m = n$

$n = 50{,}000 \rightarrow$ [9.8 10 10.2]

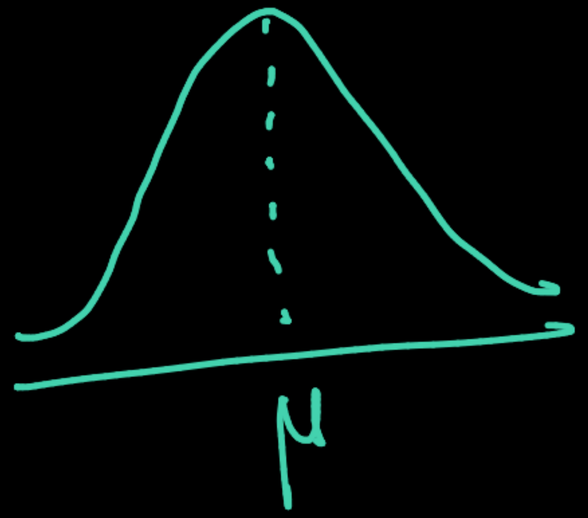
$m = n$

CLT



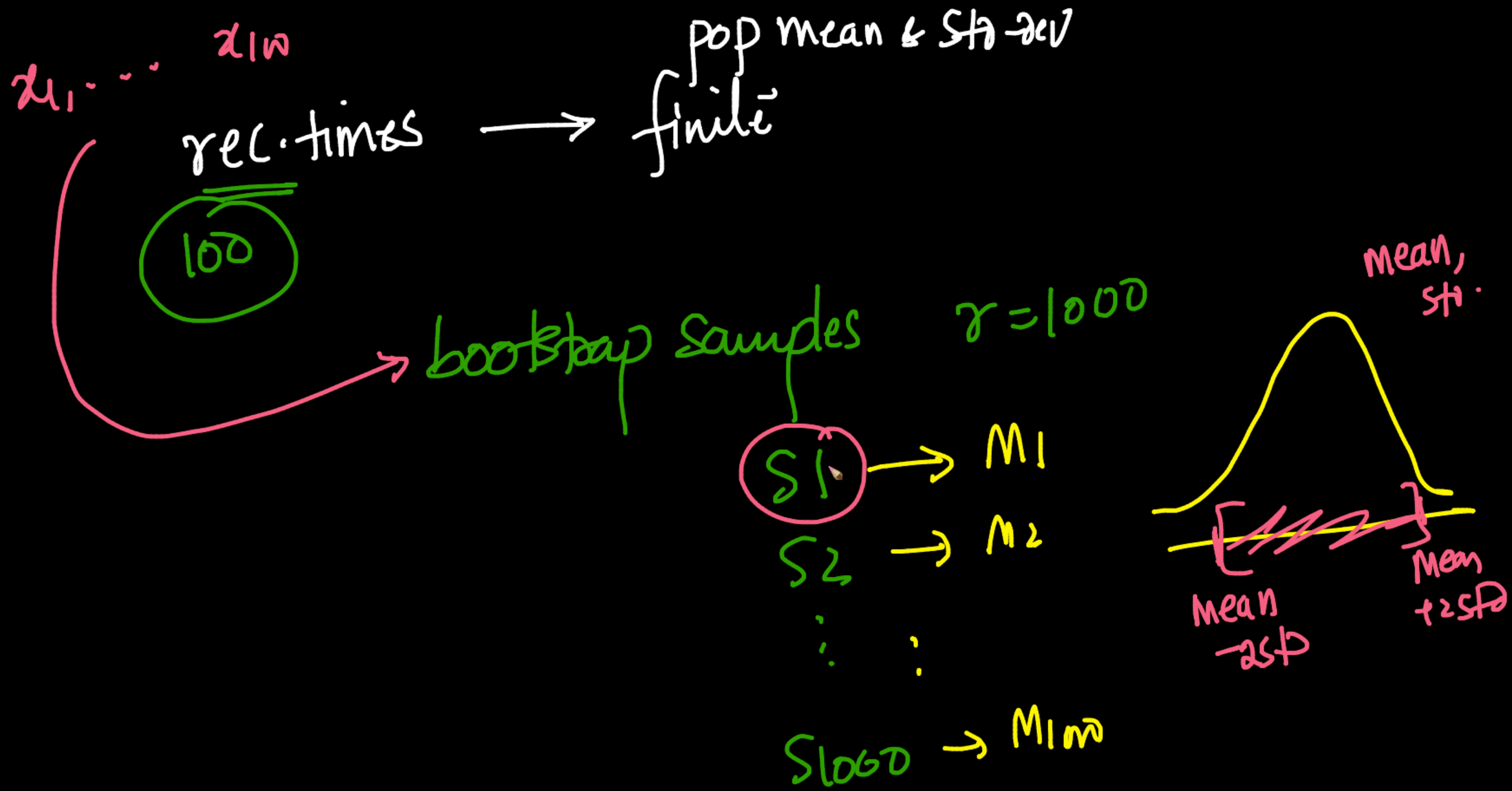
{ sample-means $M_1, M_2, \cdots M_K \sim$ Normal $\left(\mu, \frac{\sigma}{\sqrt{n}}\right)$

sample size



theoritically $n \to \infty$

Sample - medians ⟶ not guaranteed


pop mean & Std dev
$x_1, \ldots x_{100}$
rec. times
finite
100
bootstrap samples
$r = 1000$
S1
S2
S1000
M1
M2
M1000
mean, std.
mean -2sd
mean +2sd

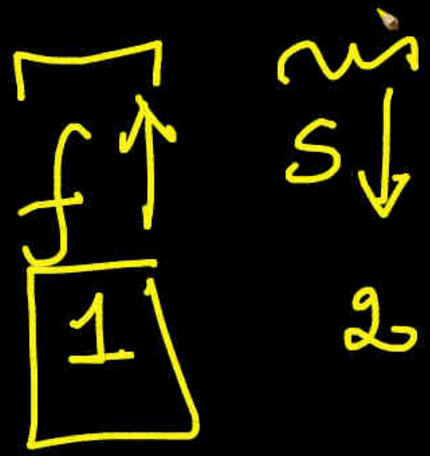
Lots of code

Simulation

} next-class

fast / slow: ↓

29
(good-speed)



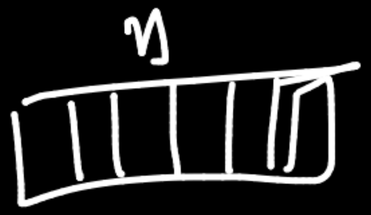
[END].

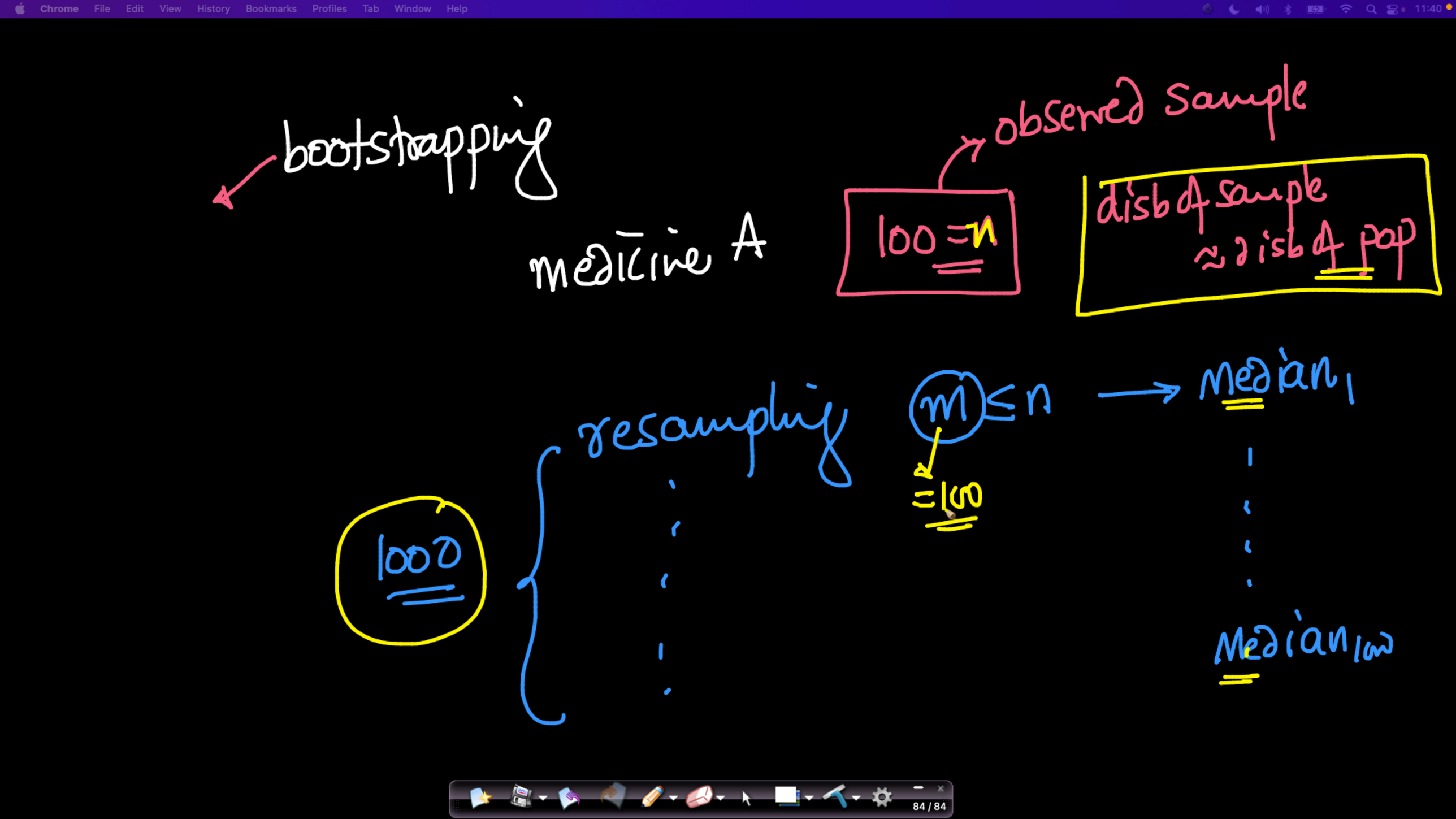
QQ-plot

TC: O(n lg n) + n + n

Sort + get percentiles ; plotting

SC: O(1)




bootstrapping
medicine A
observed sample
100 = n
disb of sample ≈ disb of pop
resampling
1000
m ≤ n
= 100
median 1
Median 1000

m , n & r Change → 95 C 1

[,]

QQ-plot → obs vs Binomial ✓

Bootstrapping → distribution-free method

'n' same for all samples

CLT → sample means are 'Gaussian' dist'b

↳ 'n' is sample

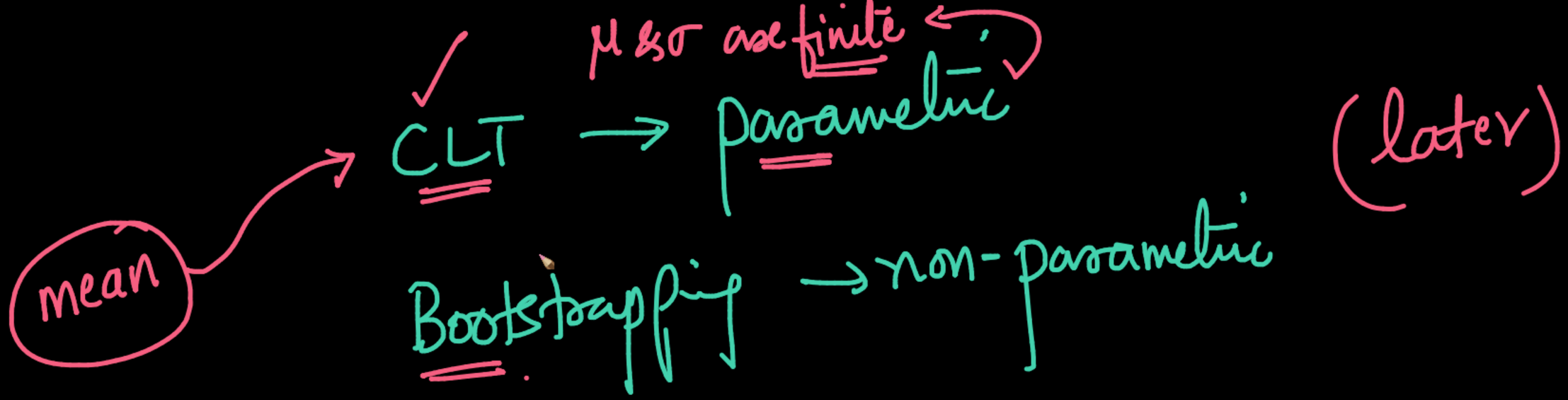
CLT → few outliers (1%)

↳ reasonably well ✓

Sample-1 → Small part of an outlier

↓

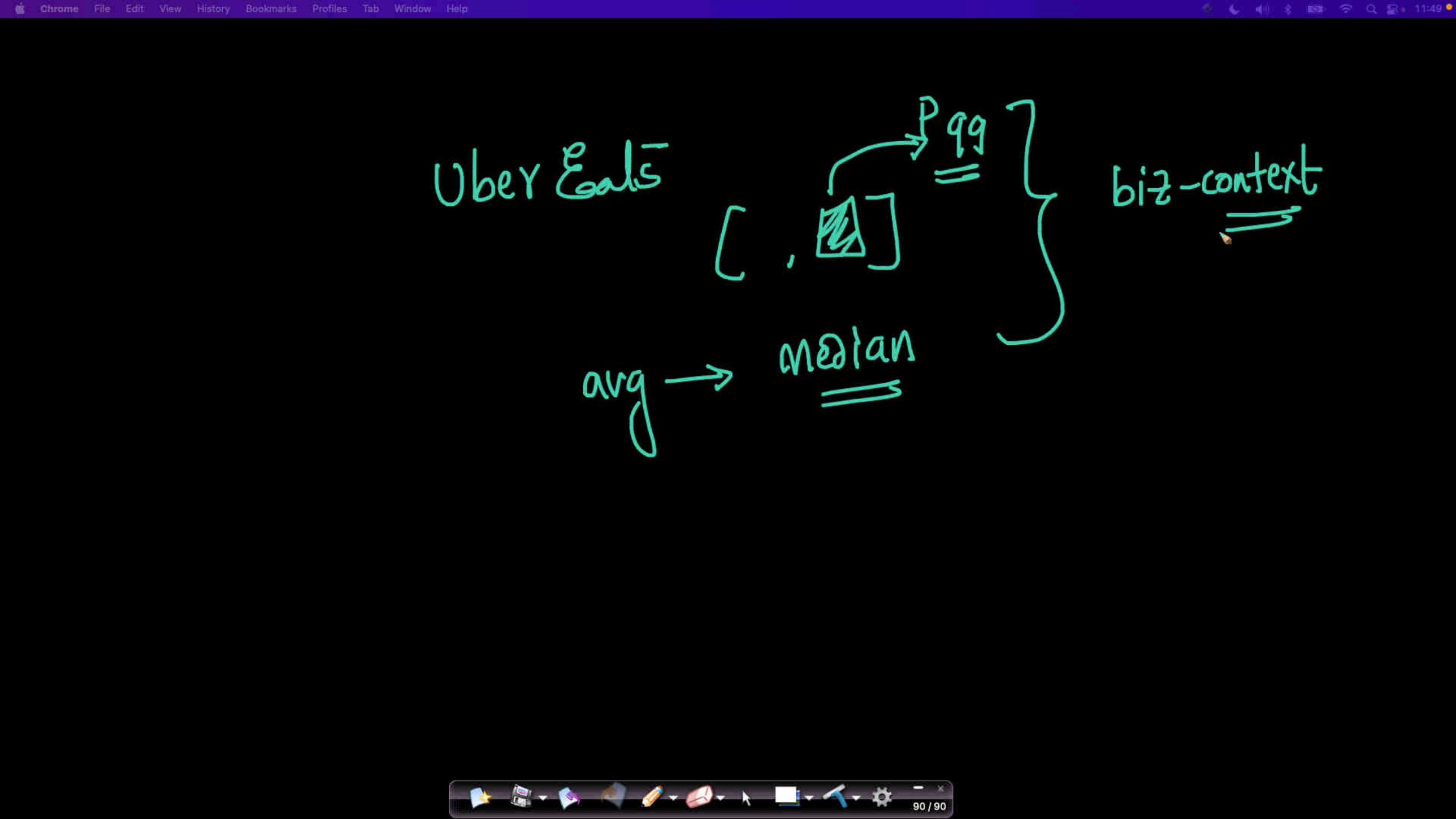
M_1

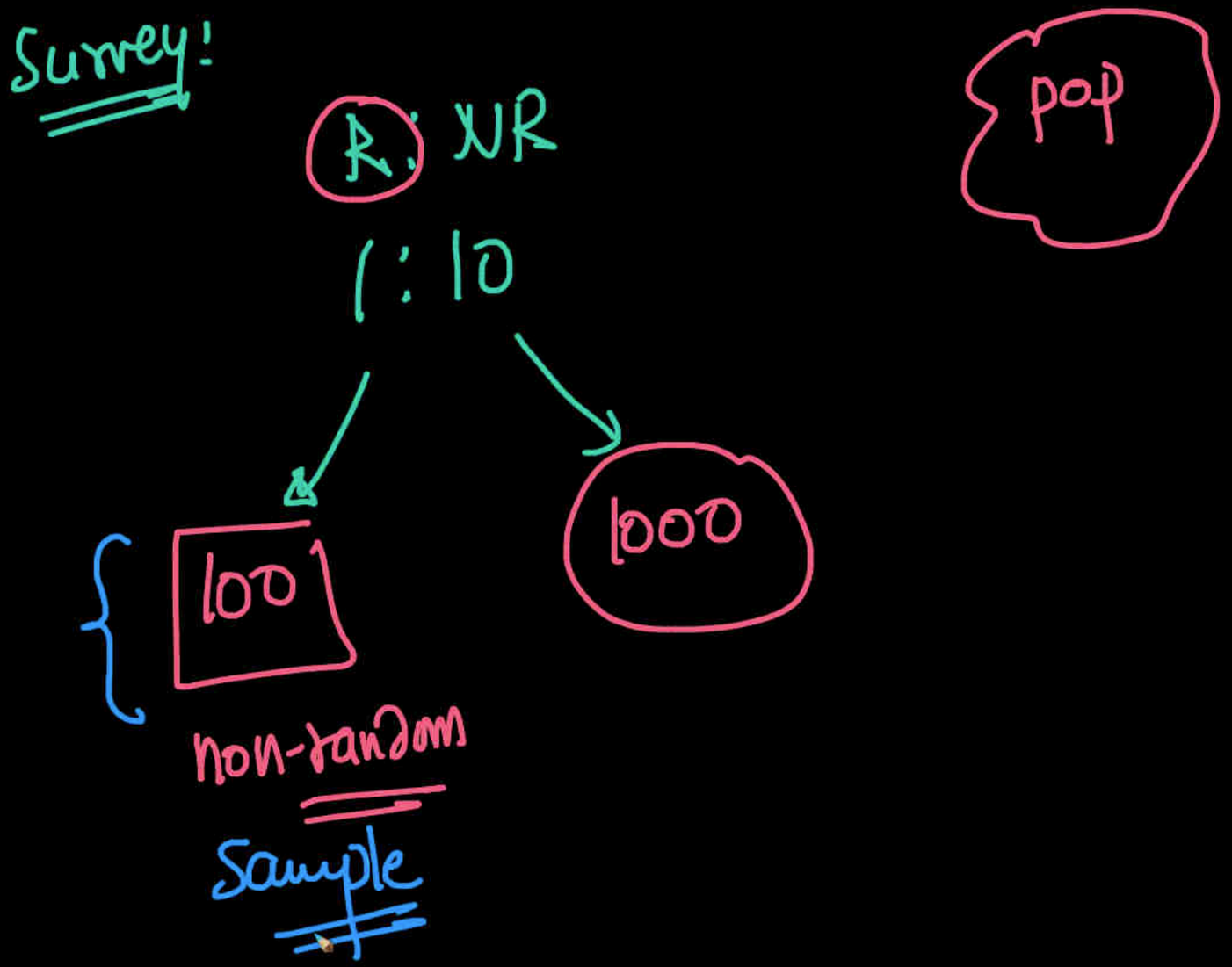

mean
CLT
μ & σ are finite
parametric
Bootstrapping
non-parametric
(later)

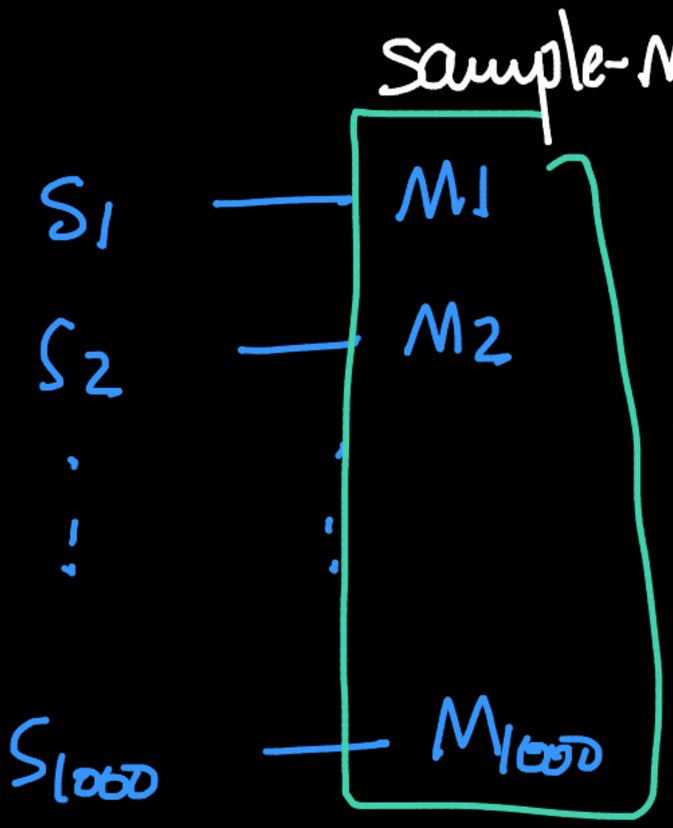
C.I of mean → CLT (μ & σ are finite)
→ sample means ~ Gaussian

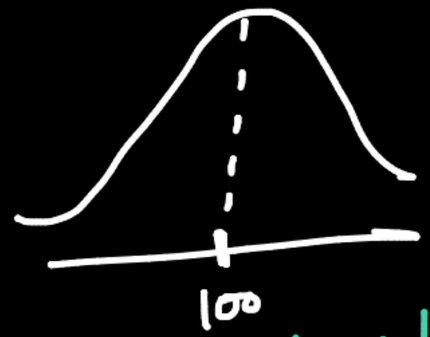
C.I of Median
P_{99}
⋮
C.I of mean
(μ & σ are not finite)

→ Bootstrapping...


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Uber Eats
P99
biz-context
avg → median
90 / 90







Mean (M_i's) = 100

Stdev (M_i's) = 2

$100 - 2 \times 2$ $100 + 2 \times 2$

95% CI of Mean [96, 104]

CLT: population μ & σ are finite

↳ finance & trading

$y_1, y_2 \ldots y_n$

$Y \sim$ log-normal

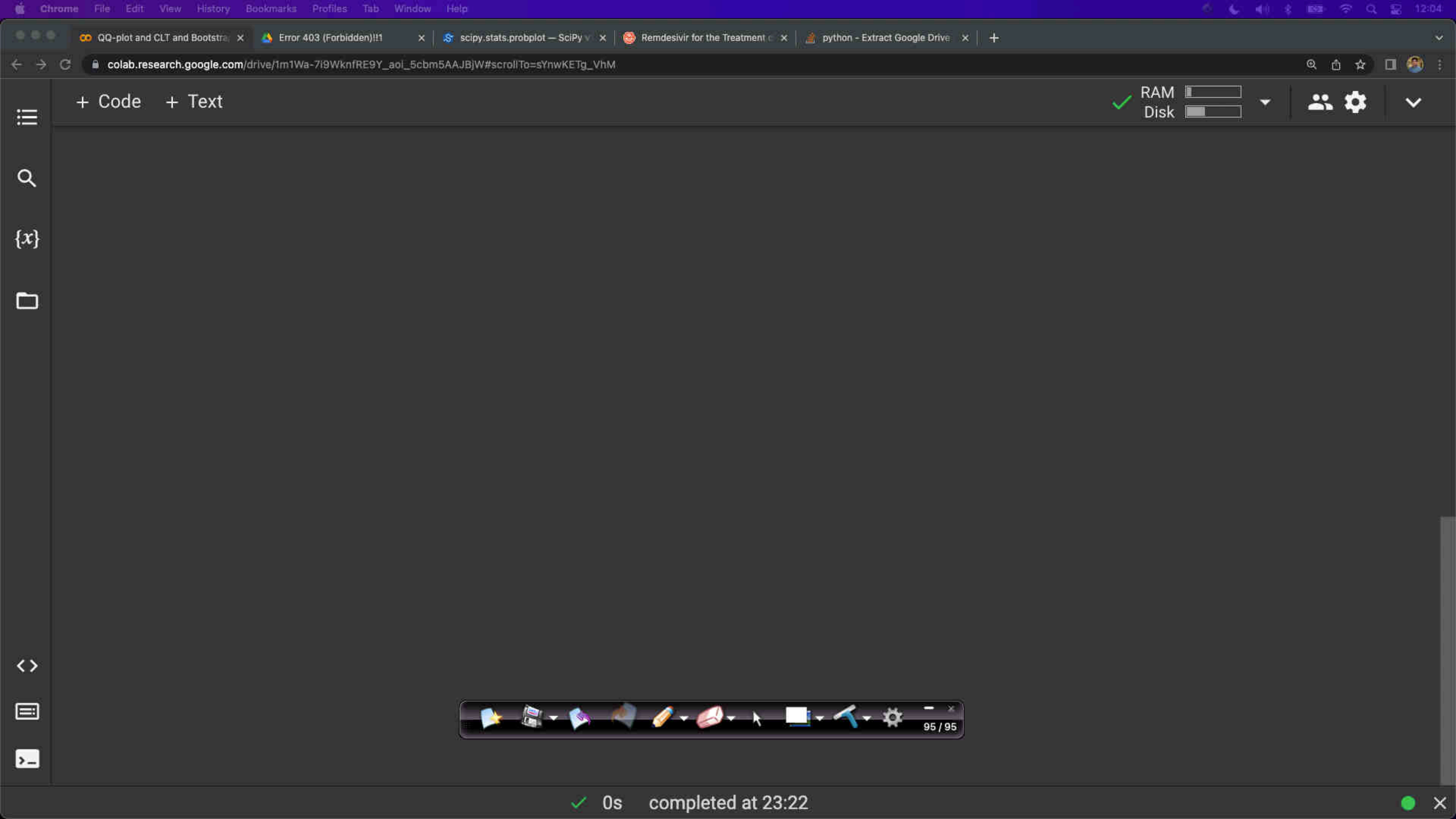
iff $\log(Y) \sim$ Normal

$\log(y_1)$ $\log(y_2) \ldots$ $\log(y_n)$

$\downarrow$ $\downarrow$

y_1' y_2' - - - - y_n' → Normal

QQ-plot


Chrome File Edit View History Bookmarks Profiles Tab Window Help
QQ-plot and CLT and Bootstra
Error 403 (Forbidden)!!1
scipy.stats.probplot — SciPy
Remdesivir for the Treatment
python - Extract Google Drive
colab.research.google.com/drive/1m1Wa-7i9WknfRE9Y_aoi_5cbm5AAJBjW#scrollTo=sYnwKETg_VhM
+ Code
+ Text
RAM
Disk
95 / 95
0s
completed at 23:22