Python – Lectures 1 & 2

(Basics)

Deepak Sharma
BSBE and DS&AI
IIT Roorkee

Historical background

- First truly modern computer was Mark I (1944).
- In 1936, mathematician Alan Turing described a hypothetical computing device called a universal Turing machine.
- The Church-Turing thesis states that if a function is computable, a Turing machine can be programmed to compute it.
- A programming language is said to be Turing complete if it can be used to simulate a universal Turing machine.
- All modern programming languages are Turing complete.

Programming languages

- Anything that can be programmed in one language (eg. Python) can be programmed in any other language (eg. C++).

- However, some things may be easier to program in a particular language:

  - MATLAB: for manipulating vectors and matrices
  - C: for programs that control data networks
  - PHP: for building Web sites
  - SQL: for building databases
  - Python: a general purpose language

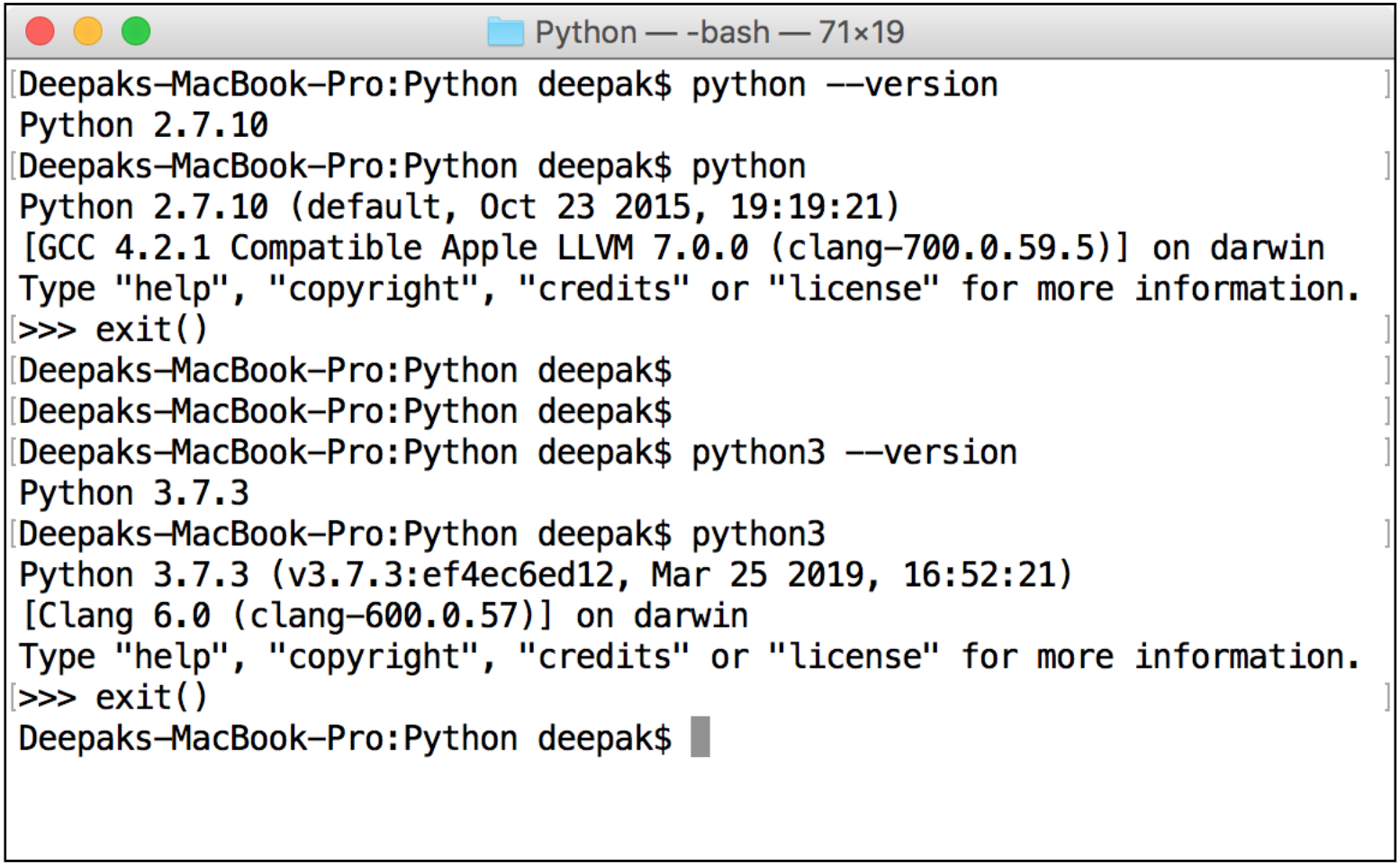
Programming rules

- Primitive constructs (words): literals (eg. number 3.2 or string abc) and infix operators (eg. + or /)
- Syntax: defines whether the strings of characters/symbols are well formed (eg. 2.3 2.3 is incorrect)
- Static semantics: defines which syntactically valid strings have meaning [eg 3.2/'abc' is syntactically well formed (<literal><operator><literal>) but incorrect]
- Semantics: associates meaning with syntactically correct strings with no static semantic errors

Python

- General-purpose programming language that can be used effectively to build any kind of program
- Introduced by Guido van Rossum in 1990; Python 2.0 in 2000 and Python 3.0 in 2008
- Easy to learn
- Designed to be interpreted (no compiler)
- Large number of libraries available

Python installation



Objects

Scalar:	**Single value**
int	integers
float	real numbers
bool	true/false
None	null
Non-Scalar:	**Many objects**
string	sequence of symbols (words)
tuple	ordered, immutatable
list	ordered, mutatable
dict	ordered, mutatable, key-value pair
set	unordered, mutatable, unique

Operators

Add/Sub	+, -	
Mult/Div	*, /	(// integer/floor division)
Remainder	%	
Power	**	
Equality	==, !=	
Relational	<, <=, >, >=	

First program

Python — vi circle_area.py — 58×15

```
#!/usr/bin/python

###################################
#calculate the area of a circle
###################################

pi = 3.14
radius = 11
area = pi*(radius**2
print area
~
~
~
~
"circle_area.py" 10L
```

Command interpretation

Comment

Python — -bash — 62×15

```
Deepaks-MacBook-Pro:Python deepak$ ls -lt
total 8
-rw-r--r--  1 deepak  admin  174 Jun 17 13:05 circle_area.py
Deepaks-MacBook-Pro:Python deepak$
Deepaks-MacBook-Pro:Python deepak$ python circle_area.py
379.94
Deepaks-MacBook-Pro:Python deepak$ chmod a+x circle_area.py
Deepaks-MacBook-Pro:Python deepak$ ls -lt
total 8
-rwxr-xr-x  1 deepak  admin  174 Jun 17 13:05 circle_area.py
Deepaks-MacBook-Pro:Python deepak$
Deepaks-MacBook-Pro:Python deepak$ ./circle_area.py
379.94
Deepaks-MacBook-Pro:Python deepak$
```

Python IDE's

- One can also do programming using integrated development environment (IDE)
- PyCharm, Anaconda, Jupyter, Canopy
- Provide text editor with syntax highlighting, auto completion and smart indentation
- Also provide an integrated debugger

Float (Jupyter notebook)

```
In [1]: a = 0.1
        b = 0.2
        c = a + b
        c

Out[1]: 0.30000000000000004

In [2]: c == 0.3

Out[2]: False

In [3]: from decimal import Decimal
        d = Decimal("0.1") + Decimal("0.2")
        d

Out[3]: Decimal('0.3')

In [4]: print(d)

        0.3
```

Float

In [1]:
```
from decimal import Decimal
d = Decimal("0.1") + Decimal("0.2")
print(d)
```

```
0.3
```

In [2]:
```
a = 0.3
d == a
```

Out[2]:
```
False
```

In [3]:
```
print(type(d))
```

```
<class 'decimal.Decimal'>
```

In [4]:
```
print(type(a))
```

```
<class 'float'>
```

In [5]:
```
d = float(d)
d == a
```

Out[5]:
```
True
```

Boolean

```
In [1]: a=True
        a

Out[1]: True

In [2]: type(a)

Out[2]: bool

In [3]: b=bool(0)
        b

Out[3]: False
```

Identity operators

- is, is not
- They compare the memory locations of two objects

```
In [1]: x = [1, 2, 3]
        y = [1, 2, 3]

In [2]: print(x == y)

        True

In [3]: print(x is y)   # different objects in memory

        False

In [4]: print(x is not y)

        True
```

None

- A special constant representing the absence of a value or a null value.

```
In [1]: a = None
        a is None

Out[1]: True
```

```
In [2]: b = 1
        b is not None

Out[2]: True
```

```
In [3]: print(type(None))

        <class 'NoneType'>
```

```
In [4]: variable = None

        if variable is None:
            print("variable has a value of None")
        else:
            print("variable has a value")

        variable has a value of None
```

Input, Type and Type conversions

Python — vi input_type.py — 87×21

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

###################################
#Input, Type and Type conversions
###################################

n = input('Enter an integer: ')
print(type(n), '\n')          #To find type of the object

x = n*4
print(x, '\n')

y=int(n)*4
print(y, '\n')

z=int(3.9)
print(z, '\n')
~
~
~
"input_type.py" 17L, 446C
```

Python — -bash — 59×12

```
Deepaks-MacBook-Pro:Python deepak$ ./input_type.py
Enter an integer: 3
<class 'str'>

3333

12

3

Deepaks-MacBook-Pro:Python deepak$
```

Multiple assignment

Python — vi multiple_assign.py — 65×15

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

################################
#multiple assignment
################################

x, y = 2, 3
x, y = y, x
print('x =', x)
print('y =', y)
~
~
~
~
```

Python — -bash — 65×9

```
Deepaks-MacBook-Pro:Python deepak$ ./multiple_assign.py
x = 3
y = 2
Deepaks-MacBook-Pro:Python deepak$ python multiple_assign.py
('x =', 3)
('y =', 2)
Deepaks-MacBook-Pro:Python deepak$
```

Not backward compatible

Overloaded operators

```
Python — -bash — 72×20
Deepaks-MacBook-Pro:Python deepak$ python3
Python 3.7.3 (v3.7.3:ef4ec6ed12, Mar 25 2019, 16:52:21)
[Clang 6.0 (clang-600.0.57)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>> 3+4
7
>>> 'Deepak'+'Sharma'
'DeepakSharma'
>>>
>>>
>>> 3*4
12
>>> 3*'Repeat'
'RepeatRepeatRepeat'
>>>
>>>
>>> exit()
Deepaks-MacBook-Pro:Pytho
```

```
AS2024 — IPython: BEC101/AS2024 — -bash — 68×21
(base) Deepaks-MacBook-Pro:AS2024 deepak$ ipython
Python 3.7.6 (default, Jan  8 2020, 13:42:34)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.12.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: 3+4
Out[1]: 7

In [2]: 'Deepak'+'Sharma'
Out[2]: 'DeepakSharma'

In [3]: 3*4
Out[3]: 12

In [4]: 3*'Repeat'
Out[4]: 'RepeatRepeatRepeat'

In [5]: exit()
(base) Deepaks-MacBook-Pro:AS2024 deepak$
```

String

```
In [1]: s1 = 'abc'
        s1

Out[1]: 'abc'

In [2]: s2 = '1'
        s2

Out[2]: '1'

In [3]: str(1)

Out[3]: '1'
```

String operations

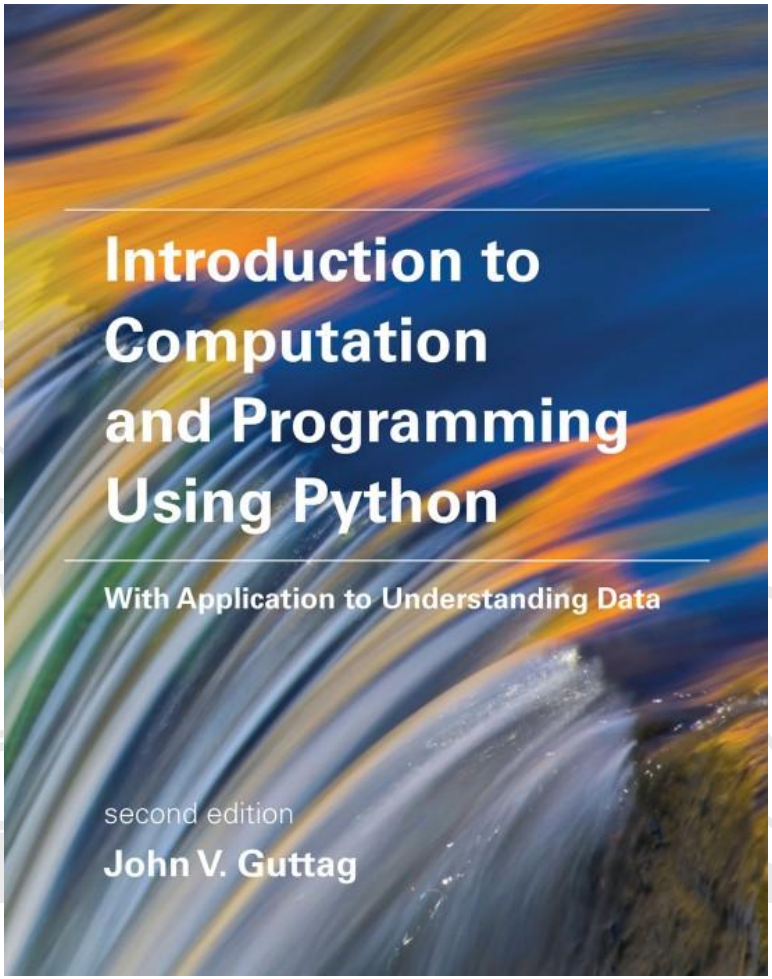
Python — vi string.py — 87×20

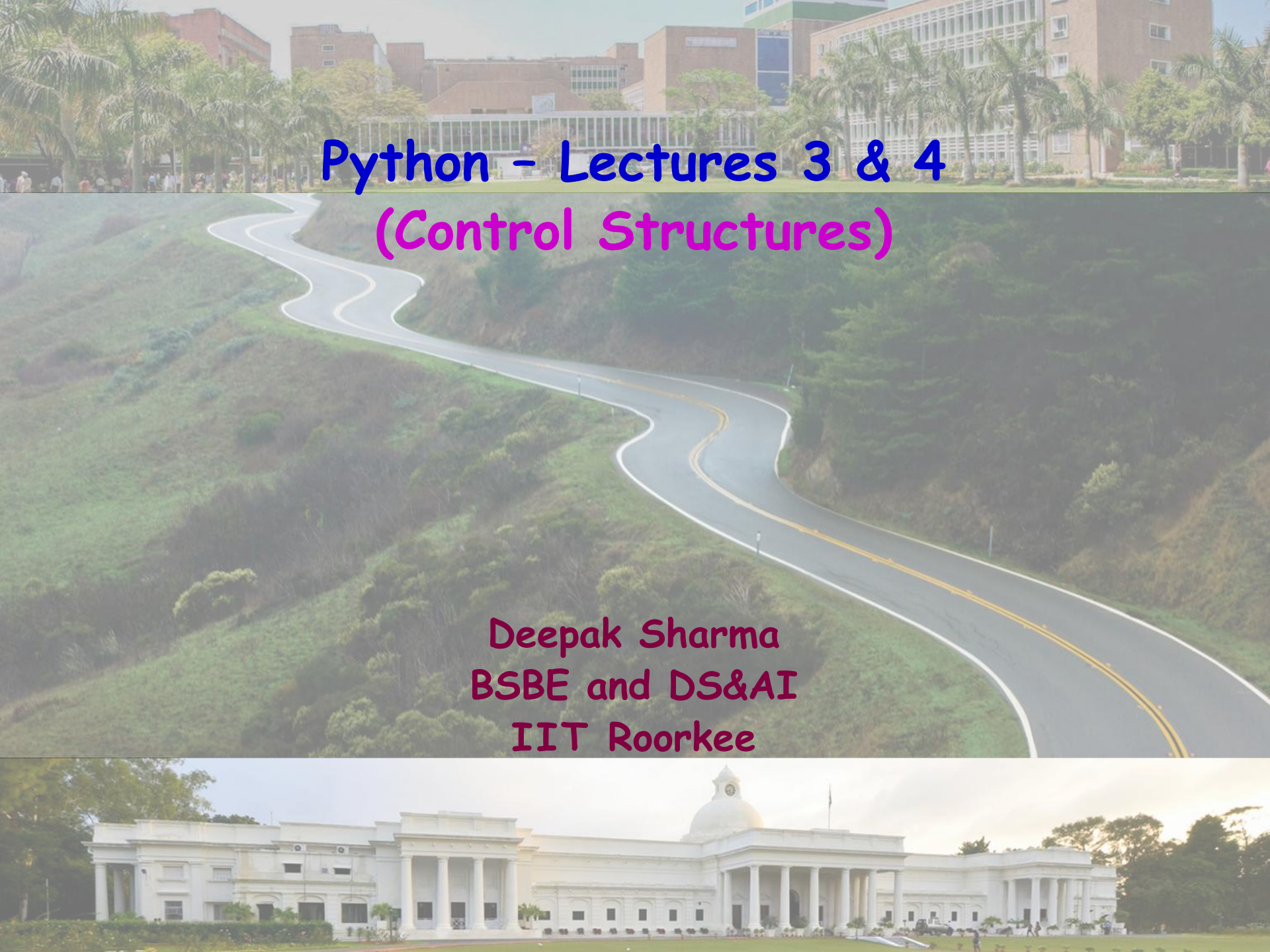
```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

################################
#String operations
################################

Length = len('abcd')          #Finds length of a string
print(Length, '\n')

Char_first = 'abcd'[0]        #Indexing is done to extract characters from string
Char_last = 'abcd'[-1]
print (Char_first, Char_last)
print (Char_first + Char_last, '\n')

Substring = 'abcd'[0:2]       #Slicing is used to extract substrings of arbitary lengths
print (Substring)
~
~
~
"string.py" 16L, 505C
```

Python — -bash — 49×8

```
Deepaks-MacBook-Pro:Python deepak$ ./string.py
4

a d
ad

ab
Deepaks-MacBook-Pro:Python deepak$
```

Further reading





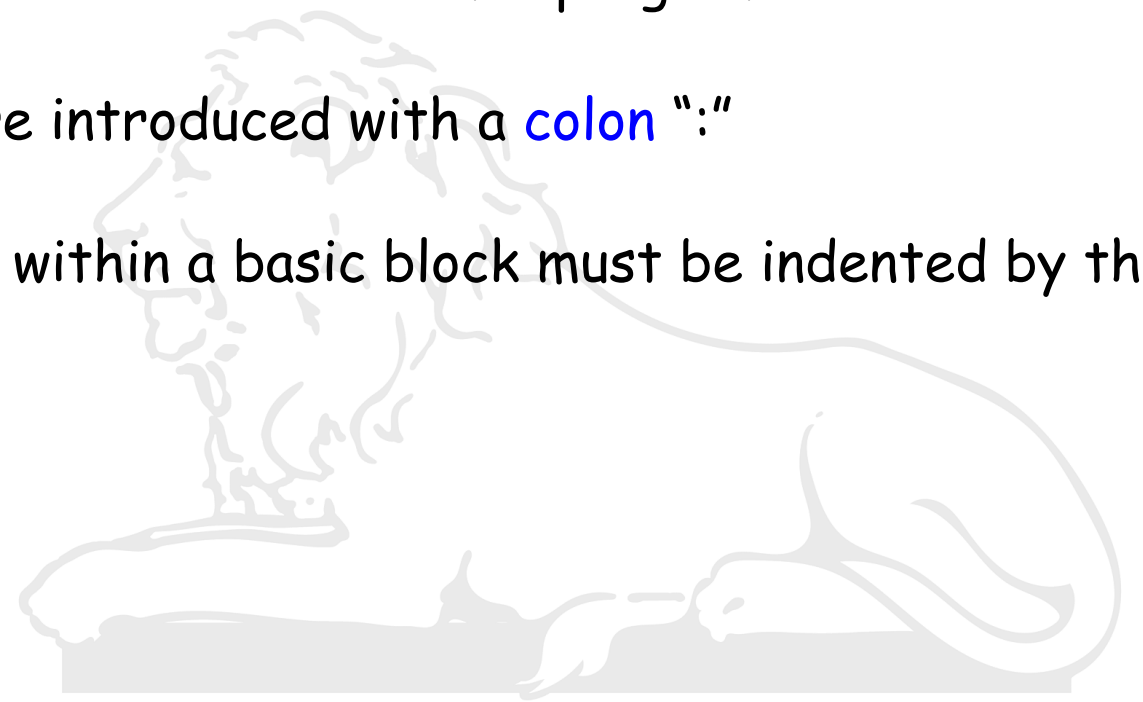
Python – Lectures 3 & 4

(Control Structures)

Deepak Sharma
BSBE and DS&AI
IIT Roorkee

Indentation

- In Python, indentation is significant
- Whitespace is used to delimit program blocks
- Blocks are introduced with a colon ":"
- Each line within a basic block must be indented by the same amount



if, else and elif statements

Python — vi if_else_elif.py — 70×21

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

###################################
#if, else and elif
###################################

x = input('Enter an integer: ')
x = int(x)

if x%2 == 0:
        if x%3 == 0:
                print ('Divisible by 2 and 3')
        else:
                print ('Divisible by 2 and not by 3')
elif x%3 == 0:
        print('Divisible by 3 and
        print('Indented')
print('Not indented')
~
~
"if_else_elif.py" 18L, 400C
```

Python — -bash — 53×10

```
Deepaks-MacBook-Pro:Python deepak$ ./if_else_elif.py
Enter an integer: 6
Divisible by 2 and 3
Not indented
Deepaks-MacBook-Pro:Python deepak$ ./if_else_elif.py
Enter an integer: 9
Divisible by 3 and not by 2
Indented
Not indented
Deepaks-MacBook-Pro:Python deepak$
```

Compound expressions

Python — vi compound_cond.py — 70×19

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

###################################
#if, elif and compound conditions
###################################

x = input('Enter an integer: ')
x = int(x)

if x%2 == 0 and x%3 == 0:
#if x%2 == 0 or x%3 == 0:
        print ('Divisible by 2 and 3')
elif x%2 == 0:
        print ('Divisible by 2 and not by 3')
elif x%3 == 0:
        print('Divisible b
~
~
"compound_cond.py" 16L, 40
```

Python — -bash — 56×7

```
Deepaks-MacBook-Pro:Python deepak$ ./compound_cond.py
Enter an integer: 6
Divisible by 2 and 3
Deepaks-MacBook-Pro:Python deepak$ ./compound_cond.py
Enter an integer: 9
Divisible by 3 and not by 2
Deepaks-MacBook-Pro:Python deepak$
```

Bitwise operators

- Bitwise AND (&): This operator compares corresponding bits of two integers. If both bits at a given position are 1, the resulting bit is 1; otherwise, it is 0.

```
In [1]: a = 5                # 0101 in binary
        b = 3                # 0011 in binary
        result = a & b       # 0001 (1 in decimal)
        print(result)
```

```
1
```

- Bitwise OR (|): This operator compares corresponding bits of two integers. If at least one of the bits at a given position is 1, the resulting bit is 1; otherwise, it is 0.

```
In [2]: a = 5                # 0101 in binary
        b = 3                # 0011 in binary
        result = a | b       # 0111 (7 in decimal)
        print(result)
```

```
7
```

Iteration/Looping (while loop)

Python — vi while.py — 70×19

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

###################################
#Square an integer
###################################

x = input('Enter an integer: ')
x = int(x)

ans = 0
itersLeft = x
while (itersLeft != 0):
        ans = ans + x
        itersLeft = itersLeft - 1
print (str(x) + '*' + str(x) + ' = ' + str(ans))
~
~
~
"while.py" 15L, 336C
```

Python — -bash — 54×7

```
Deepaks-MacBook-Pro:Python deepak$ ./while.py
Enter an integer: 7
7*7 = 49
Deepaks-MacBook-Pro:Python deepak$ ./while.py
Enter an integer: 25
25*25 = 625
Deepaks-MacBook-Pro:Python deepak$
```

abs function

Python — vi while_mod.py — 70×19

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

###################################
#Square an integer
###################################

x = input('Enter an integer: ')
x = int(x)

ans = 0
itersLeft = abs(x)
while (itersLeft != 0):
        ans = ans + abs(x)
        itersLeft = itersLeft - 1
print (str(x) + '*' + str(x) + ' = ' + str(ans))
~
~
~
"while_mod.py" 15L, 346C
```

Python — -bash — 54×7

```
Deepaks-MacBook-Pro:Python deepak$ ./while_mod.py
Enter an integer: -7
-7*-7 = 49
Deepaks-MacBook-Pro:Python deepak$ ./while_mod.py
Enter an integer: -25
-25*-25 = 625
Deepaks-MacBook-Pro:Python deepak$
```

break statement

Python — vi break.py — 70×17

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

########################################
#Find an integer divisible by 11 and 12
########################################

x = 1

while True:
        if x%11 == 0 and x%12 == 0:
                break
        x = x + 1
print(x, 'is divisible by 11 and 12')
~
~
~
"break.py" 13L, 291C
```

Python — -bash — 52×5

```
Deepaks-MacBook-Pro:Python deepak$ ./break.py
132 is divisible by 11 and 12
Deepaks-MacBook-Pro:Python deepak$
```

pass and continue statements

AS2023 — vi pass_continue.py — 70×20

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

###################################
#continue
###################################
for i in range(1,20):
        if i==1:
                pass
                print('This statement is after pass')
        elif i%2==0:
                continue
                print('This statement is after continue')
        elif i==11:
                break
        else:
                pri
~
~
"pass_continue.py"
```

AS2023 — -bash — 62×7

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./pass_continue.py
This statement is after pass
3  is odd
5  is odd
7  is odd
9  is odd
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

pass and continue statements

AS2023 — vi pass_continue2.py — 70×20

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

###################################
#continue
###################################
for i in range(1,20):
        if i==1:
                pass
                print('This statement is after pass')
        elif i%2==0:
                cont
                pri
        elif i==10:
                bre
        else:
                pri

~
~
"pass_continue2.py"
```

AS2023 — -bash — 63×12

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./pass_continue2.py
This statement is after pass
3  is odd
5  is odd
7  is odd
9  is odd
11  is odd
13  is odd
15  is odd
17  is odd
19  is odd
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

for loop

Python — vi for.py — 87×15

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

########################################
#for loop
########################################

x = 4

for i in range (0, x):  #arguments to range function are evaluated before 1st iteration
        print(i)
        x = 5
~
~
~
"for.py" 11L, 268C
```

Python — -bash — 52×9

```
Deepaks-MacBook-Pro:Python deepak$ ./for.py
0
1
2
3
Deepaks-MacBook-Pro:Python deepak$
```

for loop (take 2)

Python — vi for2.py — 72×16

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

########################################
#for loop (take2)
########################################

x = 4

for j in range (x):
        for i in range(x):
                print(i)
                x = 2
~
~
~
"for2.py" 12L, 230C
```

Python — -bash — 52×14

```
Deepaks-MacBook-Pro:Python deepak$ ./for2.py
0
1
2
3
0
1
0
1
0
1
Deepaks-MacBook-Pro:Python deepak$
```

for loop (take 3)

Python — vi for3.py — 74×16

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

########################################
#for loop (take3)
########################################

total = 0

for c in '123456789':       #for statement can be used with in operator
                               to iterate over characters of a string

        total = total + int(c)
print(total)
~
~
~
"for3.py" 11L, 356C
```

Python — -bash — 54×7

```
Deepaks-MacBook-Pro:Python deepak$ ./for3.py
45
Deepaks-MacBook-Pro:Python deepak$
```

Prime numbers

AS2023 — vi prime_number_range_counter.py — 70×22

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

###################################
#range
###################################
lower=int(input("Enter the lower interval: "))
upper=int(input("Enter the upper interval: "))

for num in range(lower,upper+1):
        if num>1:
                j=0
                for i in range(2,num):
                        if(num%i)==0:
~
~
"prime_n
```

AS2023 — -bash — 74×9

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./prime_number_range_counter.py
Enter the lower interval: 2
Enter the upper interval: 12
3
5
7
11
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

for else loop

AS2023 — vi for_else.py — 70×14

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

################################
#for else
################################

for x in range(6):
  print(x)
else:
  print("Finally finished!")
~
~
~
"for_else.py" 10L,
```

AS2023 — -bash — 58×9

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./for_else.py
0
1
2
3
4
5
Finally finished!
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

Prime numbers

AS2023 — vi prime_number_range_forelse.py — 70×18

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

#################################
#range
#################################
lower=int(input("Enter the lower interval: "))
upper=int(input("Enter the upper interval: "))

for num in range(lower,upper+1):
        if num>1:
                for i in range(2,num):
                        if(num%i)==0:
                                break
~
~
"prime_n
```

AS2023 — -bash — 74×9

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./prime_number_range_forelse.py
Enter the lower interval: 2
Enter the upper interval: 12
2
3
5
7
11
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

while else loop

AS2023 — vi while_else.py — 70×15

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

#################################
#while else
#################################

i = 1
while i < 6:
        print(i)
        i += 1
else:
        print("i is no longer less than 6")
~
~
"while_else.py"
```

AS2023 — -bash — 63×9

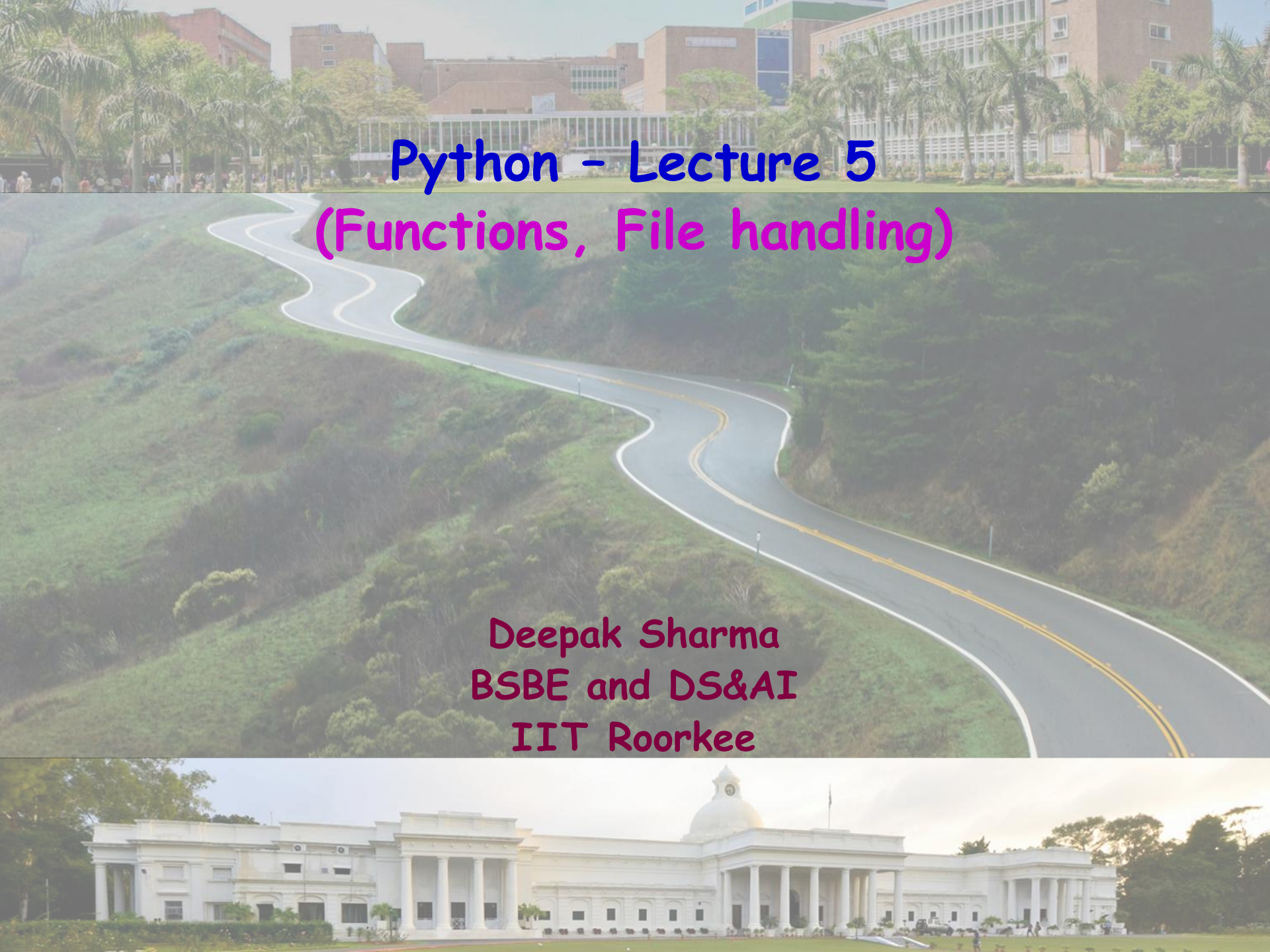
```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./while_else.py
1
2
3
4
5
i is no longer less than 6
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

Practice Programs

PP1: Make a program (without using compound expressions) to determine the grade of the marks entered by the user as per the scheme below:

Marks	Grade
$90 \leq x$	A
$80 \leq x < 90$	B
$70 \leq x < 80$	C
$60 \leq x < 70$	D
$x < 60$	F

PP2: Make a program to find the sum of even and odd digits of a number entered by the user.

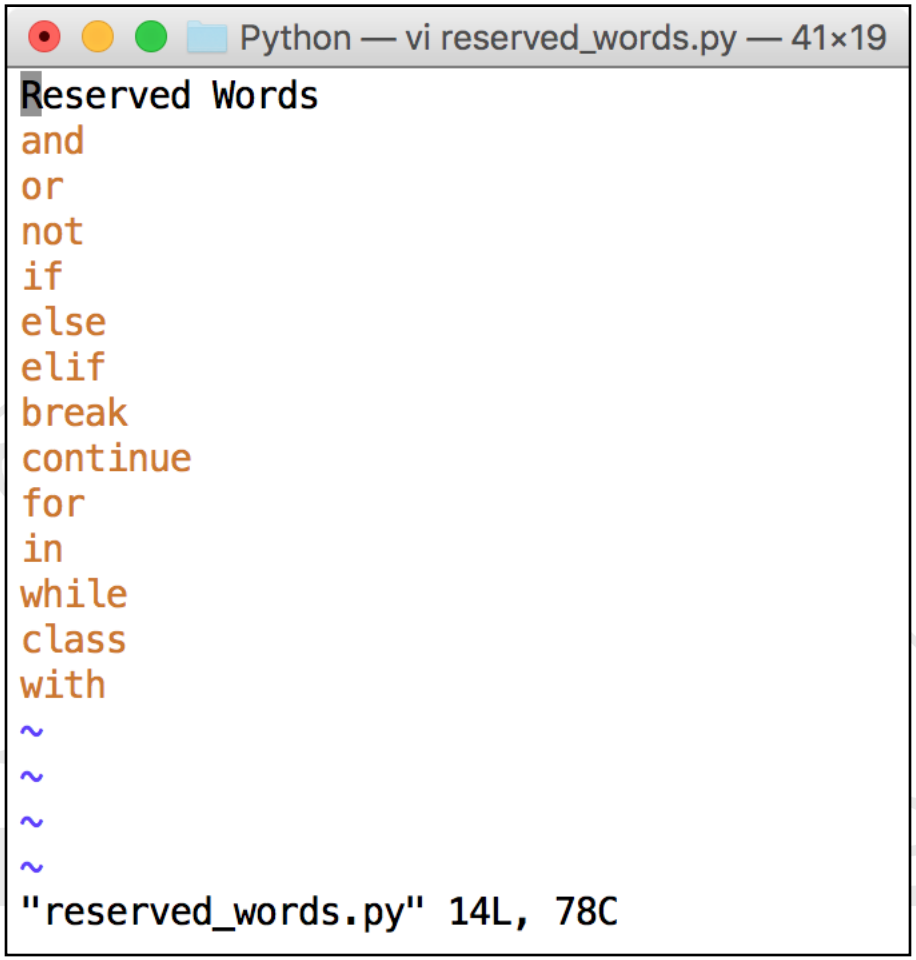


Python - Lecture 5

(Functions, File handling)

Deepak Sharma
BSBE and DS&AI
IIT Roorkee

Reserved words



Cube root (logic 1)

Python — vi cube_root.py — 72×22

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

########################################
#Find the cube root of a perfect cube
########################################

x = int(input('Enter an integer: '))
ans = 0

while ans**3 < abs(x):
        ans = ans + 1

if ans**3 != abs(x):
        print(x, 'is not a perfect c
else:
        if x < 0:
                ans = -ans
        print('Cube root of', x,'is'
~
~
~
"cube_root.py" 18L, 391C
```

Python — -bash — 50×10

```
Deepaks-MacBook-Pro:Python deepak$ ./cube_root.py
Enter an integer: 27
Cube root of 27 is 3
Deepaks-MacBook-Pro:Python deepak$ ./cube_root.py
Enter an integer: 9
9 is not a perfect cube
Deepaks-MacBook-Pro:Python deepak$ ./cube_root.py
Enter an integer: -8
Cube root of -8 is -2
Deepaks-MacBook-Pro:Python deepak$
```

Cube root (logic 2)

Python — vi cube_root2.py — 72×25

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

########################################
#Find the cube root of a perfect cube
########################################

x = int(input('Enter an integer: '))

#ans = 0
#while ans**3 < abs(x):
#       ans = ans + 1

for ans in range(0, abs(x)+1):
        if ans**3 >= abs(x):
                break

if ans**3 != abs(x):
        print(x, 'is not a perfect cube')
else:
        if x < 0:
                ans = -ans
        print('Cube root of', x,'is', ans
~
~
"cube_root2.py" 22L, 456C
```

Python — -bash — 50×10

```
Deepaks-MacBook-Pro:Python deepak$ ./cube_root.py
Enter an integer: 27
Cube root of 27 is 3
Deepaks-MacBook-Pro:Python deepak$ ./cube_root.py
Enter an integer: 9
9 is not a perfect cube
Deepaks-MacBook-Pro:Python deepak$ ./cube_root.py
Enter an integer: -8
Cube root of -8 is -2
Deepaks-MacBook-Pro:Python deepak$
```

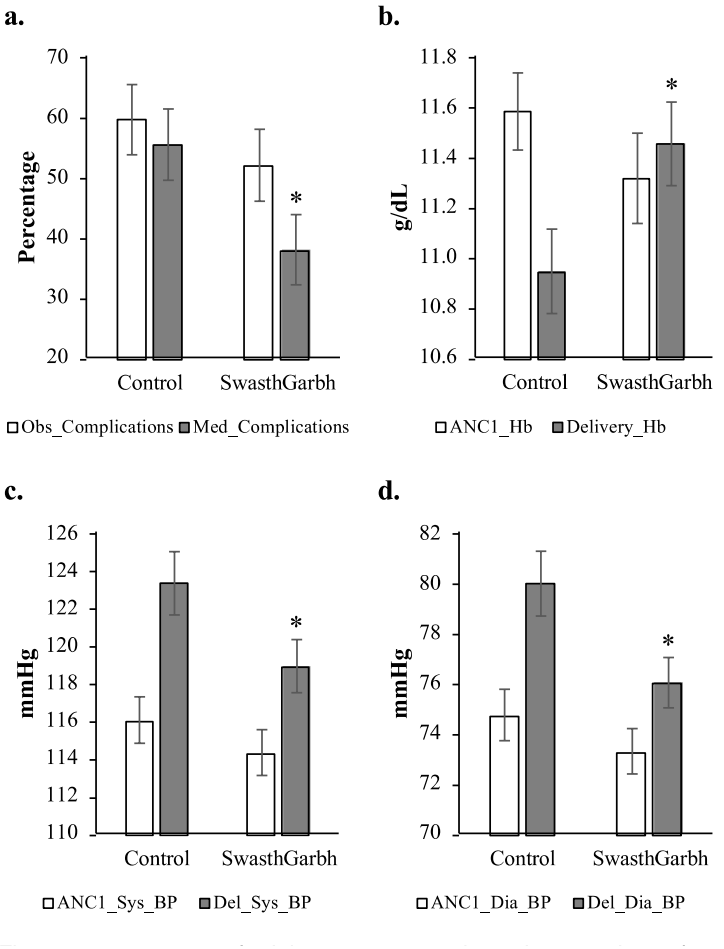
Statistical Analysis

(Basics)

Deepak Sharma
BSBE and DS&AI
IIT Roorkee

Need to study statistics

How will we prove that SwasthGarbh App is useful?



Baseline categorical and continuous variables were summarized as frequency (%) and mean ± SD, respectively. The continuous outcome variables that followed normal distribution were compared between the groups using unpaired t-test and those not following normal distribution were compared between the groups using the Wilcoxon rank sum test. The paired t-test/signed rank test was used to compare the change from baseline. Categorical variables were compared between the groups using Chi-square/Fisher's exact test. Linear regression analysis was done to find the effect of the treatment on the number of antenatal visits adjusting for socio-demographic variables. A two-sided p-value was considered statistically significant. All the statistical analysis was carried out using Stata 15.0 (StataCorp LP, Texas, USA).

Fig. 6. Improvement of clinical parameters in patients registered on SwasthGarbh App (test group; n=71) in comparison to Control patients (not registered on App; n=72). (a) Obstetric Complications (Obs_Complications) and Medical Complications (Med_Complications), (b) ANC1 hemoglobin (ANC1_Hb) and Delivery hemoglobin (Delivery_Hb), (c) ANC1 systolic BP (ANC1_Sys_BP) and Delivery systolic BP (Del_Sys_BP), and (d) ANC1 diastolic BP (ANC1_Dia_BP) and Delivery diastolic BP (Del_Dia_BP) [*indicates significant differences ($P<0.05$) between Medical Complications in SwasthGarbh and Control groups, or ANC1 and Delivery parameters; error bars represent standard error].

Sharma S, Soni S, … **Sharma D*** (2023) *IEEE JBHI*, 27, 2729-2738.

Need to study statistics

How will we prove any experimental result is significant?

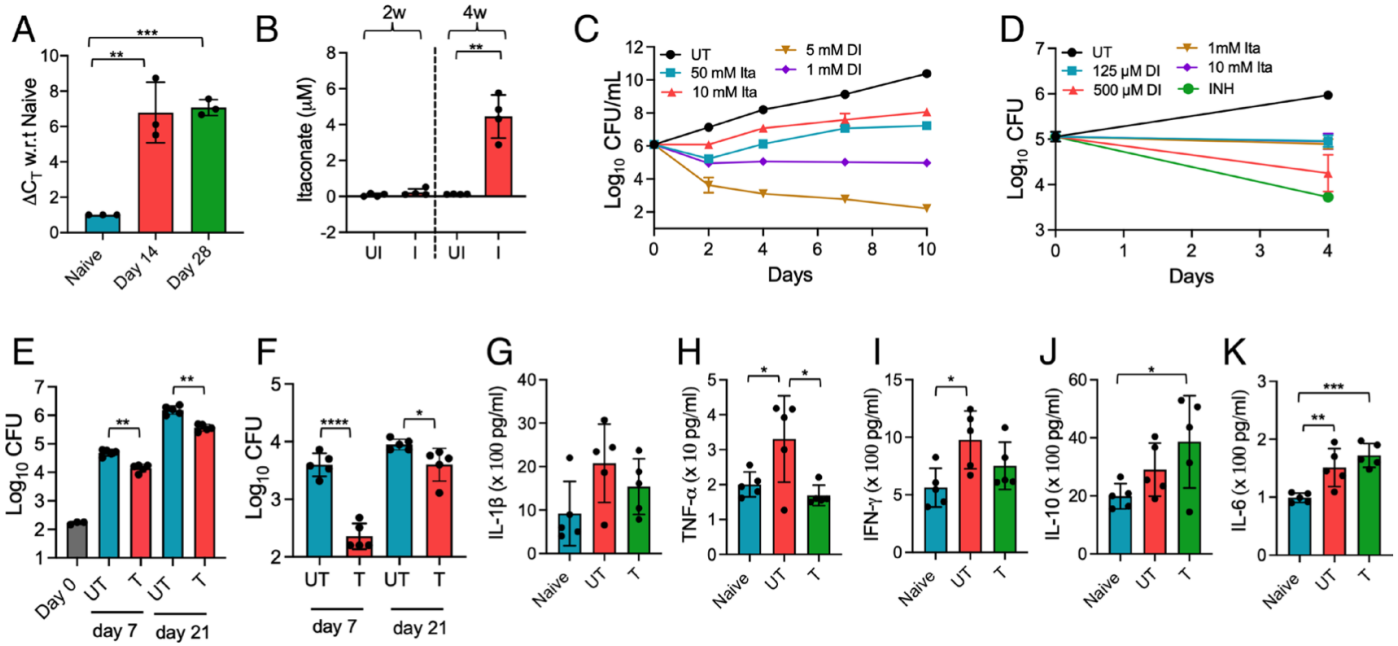


Fig. 1. Itaconate restricts the growth of *Mycobacterium tuberculosis* in vitro and in vivo. (*A* and *B*) The relative levels of *Irg1* (*A*) and itaconate (*B*) in the lung tissues of mice infected with *M. tuberculosis* at 2- and 4-wk postinfection were calculated as described in *Materials and Methods*. The data shown in panels (*A*) and (*B*) are obtained from three or four mice, respectively. (*C*) The growth of *M. tuberculosis* in MB7H9 medium was determined after exposure to various concentrations of sodium itaconate (ITA) or dimethyl itaconate (DI). (*D*) THP-1 macrophages were infected with *M. tuberculosis* strain at 1:10 MOI and treated with different concentrations of DI or ITA for 4 d. The data shown in panels C and D are the mean ± SD of $\log_{10}$ CFU obtained from two independent experiments performed in either duplicates or triplicates. (*E* and *F*) 6 to 8 wk old female BALB/c mice were infected with *M. tuberculosis* via aerosol route. The animals were administered intraperitoneally with 50 mg/kg DI for either 7 or 21 d. The data shown in these panels are mean ± SD of $\log_{10}$ CFU of lungs (*E*) and splenic (*F*) bacillary loads obtained from five animals per group. (*G–K*) The intracellular levels of IL-1β (*G*), TNF-α (*H*), IFN–γ (*I*), IL-10 (*J*), and IL-6 (*K*) were determined in naïve, untreated, and DI treated groups at day 21 posttreatment by ELISA. The data shown in these panels are mean ± SD obtained from five animals. The data were statistically analyzed using one-way ANOVA (*A*, *G*, *H*, *I*, *J*, and *K*) or a two-tailed "paired" *t* test (*B*, *E*, and *F*). $^{*}P < 0.05$, $^{**}P < 0.01$, $^{***}P < 0.001$ and $^{****}P < 0.0001$.

Priya M, … **Sharma D** … Singh R (2025) *PNAS*, 122, e2423114122.

Introduction

Statistics is the science of collection, presentation, analysis and interpretation of numerical data.

– Croxton & Cowden

Why Statistics?

- Helps us see patterns, trends and relationships in the data
- Assists in interpreting, predicting, pattern analysis and drawing conclusions
- Applicable across multiple fields including data science, medicine, biology, economics, social sciences, etc

Basic Concepts

- Descriptive statistics: Summarizing data (mean, median, mode, dispersion).
- Population: Entire group being studied.
- Sample: A subset of the population (the information the sample provides about the population is uncertain). Eg. avg CGPA of DAI101 based on my class sample (though the sample size is ~25%, will it be correct??)
- Inferential statistics: Making predictions or inferences about a population based on a sample. It includes using hypothesis testing to assess the validity of assumptions or claims about the population.
- Data Types:

  Quantitative (continuous, discrete) vs. Qualitative (nominal, ordinal).

General overview

- Question: Difference in BP recordings between machine and human measurements (Real/Chance) [Inferential Stats]
- Considerations:
  - How many machines should we test? [quality variation]
  - How many people should we test at each machine?

[Sample size estimation]

Mean blood pressures and differences between machine and human readings at four locations

		Systolic blood pressure (mm Hg)					
		Machine		Human		Difference	
Location	Number of people	Mean	Standard deviation	Mean	Standard deviation	Mean	Standard deviation
A	98	142.5	21.0	142.0	18.1	0.5	11.2
B	84	134.1	22.5	133.6	23.2	0.5	12.1
C	98	147.9	20.3	133.9	18.3	14.0	11.7
D	62	135.4	16.7	128.5	19.0	6.9	13.6

No real difference

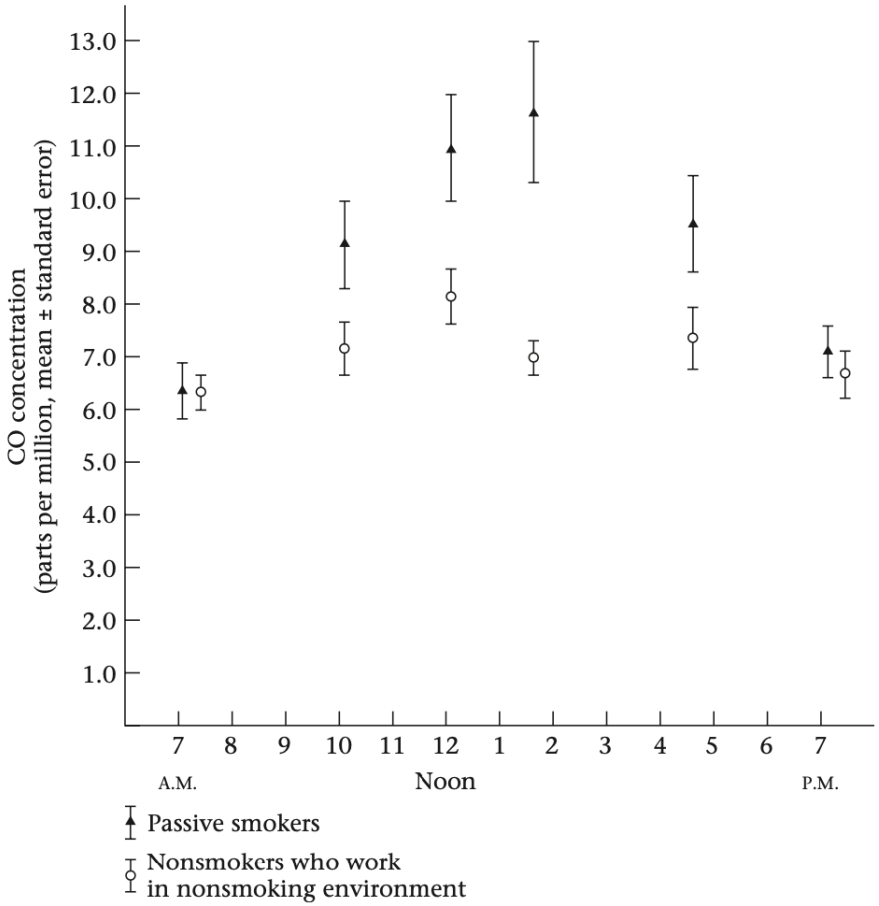
Data visualization

Graphic displays give researcher a clue to principal trends in data

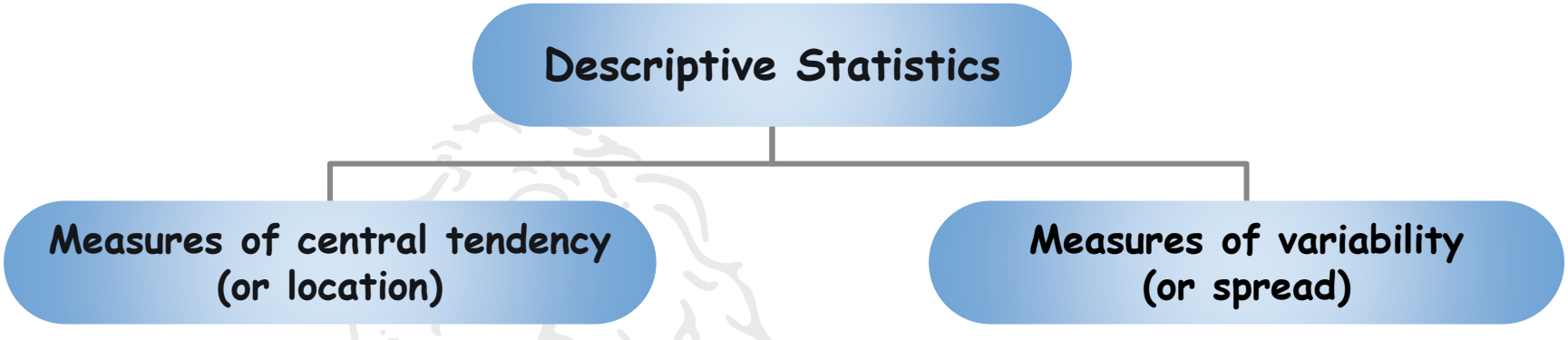
- Understandable without reading text
- Clear labeling

Converge-Diverge-Converge

Mean carbon-monoxide concentration (± standard error) by time of day as measured in the working environment of passive smokers and in nonsmokers who work in a nonsmoking environment



Descriptive Statistics



- Mean
- Median
- Mode

- Range
- Variance
- Standard Deviation
- IQR

Arithmatic Mean

- Sum of all the observations divided by the number of observations. It is written in statistical terms as

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

$$\sum_{i=1}^{n} x_i$$

is simply a short way of writing the quantity $(x_1 + x_2 + \cdots + x_n)$.

Arithmatic Mean

Sample of birthweights (g) of live-born infants born at a private hospital in San Diego, California, during a 1-week period

i	x_i	i	x_i	i	x_i	i	x_i
1	3265	6	3323	11	2581	16	2759
2	3260	7	3649	12	2841	17	3248
3	3245	8	3200	13	3609	18	3314
4	3484	9	3031	14	2838	19	3101
5	4146	10	2069	15	3541	20	2834

$$\bar{x} = (3265 + 3260 + \cdots + 2834)/20 = 3166.9 \text{ g}$$

- Very sensitive to extreme values; hence poor measure of central tendency
- 1st value 500g; Mean: 3028.7g
- Nonetheless, it is by far the most widely used measure of central tendency

Median

- If **n** observations are ordered from smallest to largest, then median is defined as

(1) The $\left(\frac{n+1}{2}\right)$th largest observation if n is odd

(2) The average of the $\left(\frac{n}{2}\right)$th and $\left(\frac{n}{2}+1\right)$th largest observations if n is even

Median of the previous question

First, arrange the sample in ascending order:

2069, 2581, 2759, 2834, 2838, 2841, 3031, 3101, 3200, 3245, 3248, 3260, 3265, 3314, 3323, 3484, 3541, 3609, 3649, 4146

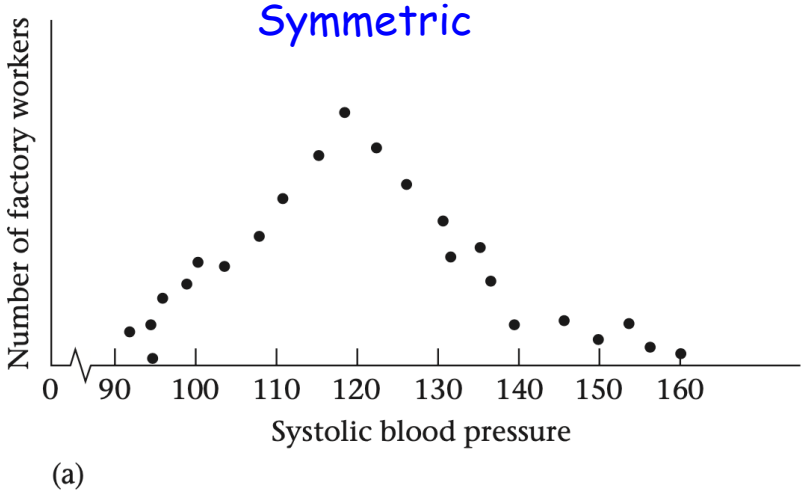
1st value 500g; Median: No change

Because n is even,

Sample median = average of the 10th and 11th largest observations
= (3245 + 3248)/2 = 3246.5 g

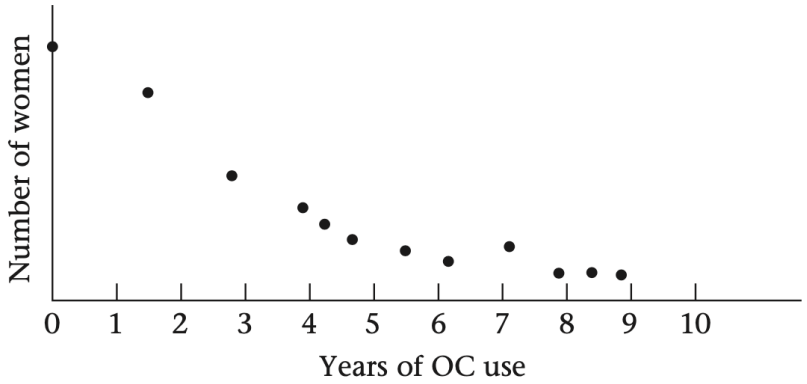
Symmetric and skewed distributions

Symmetric

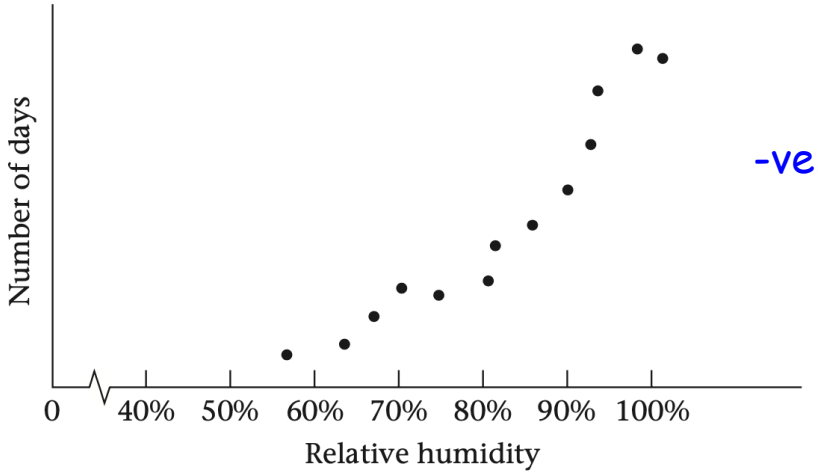


(a)

+vely skewed



(b)

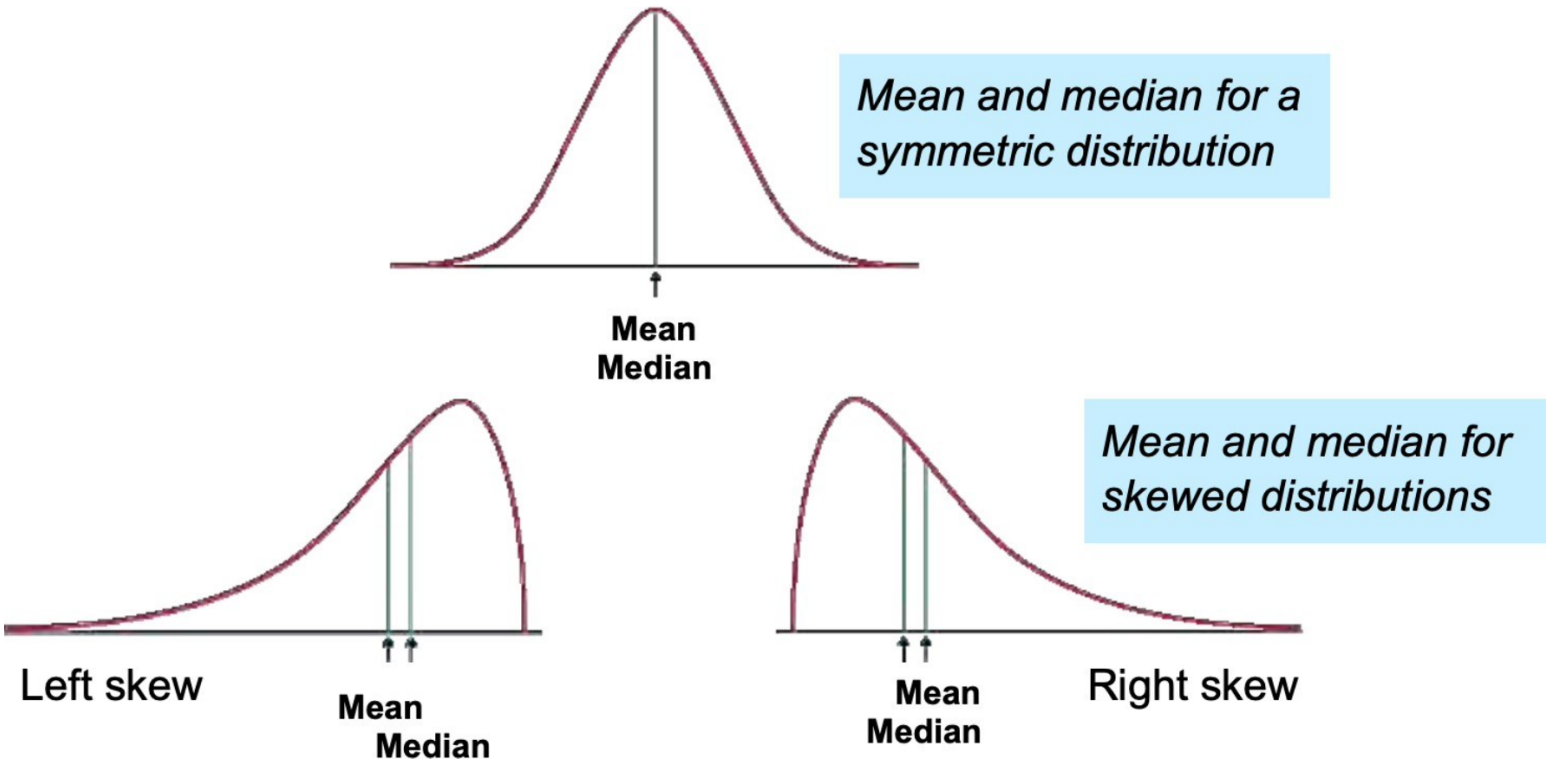


-vely skewed

(c)

Mean vs. Median

- In a symmetric distribution, mean ≈ median.
- If exactly symmetric, then mean = median.
- In a skewed distribution, the mean is pulled toward the longer tail.



Mode

- It is the most frequently occurring value among all the observations
- There may be more than one mode (Unimodal, Bimodal, Trimodal...)
- In the previous question:

  There is no mode because all values occur exactly once

Sample of admission white-blood counts (× 1000) for all patients entering a hospital in Allentown, PA, on a given day

i	x_i	i	x_i
1	7	6	3
2	35	7	10
3	5	8	12
4	9	9	8
5	8		

Mode: 8000

Mean, Median, Mode

Python — vi list_stats.py — 71×42

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

def mean(L):
        total=0
        for i in range(len(L)):
                total=total+L[i]
        mean=total/len(L)
        print("Mean: ",mean)
def median(L):
        L.sort()
        l=len(L)
        if l%2==0:
                median=(L[int(l/2)]+L[int((l-1)/2)])/2
        else:
                median=L[int((l-1)/2)]
        print("Median: ", median)
def mode(L):
        uniq=[]
        for i in range(len(L)):
                if L[i] not in uniq:
                        uniq.append(L[i])
                else:
                        pass
        uniq_count=[]
        for i in range(len(uniq)):
                count=0
                for j in range(len(L)):
                        if uniq[i]==L[j]:
                                count+=1
                uniq_count.append(count)
        max_num=max(uniq_count)
        k=uniq_count.index(max_num)
        mode=uniq[k]
        print("Mode: ", mode)

list1=[9,27,18,47,27,9,81,18,27]
list1.sort()
print(list1)
mean(list1)
median(list1)
mode(list1)
```

Python — -bash — 59×7

```
(base) Deepaks-MacBook-Pro:Python deepak$ ./list_stats.py
[9, 9, 18, 18, 27, 27, 27, 47, 81]
Mean:  29.22222222222222
Median:  27
Mode:  27
(base) Deepaks-MacBook-Pro:Python deepak$
```

Delete 27, Mode 9 (erroneous)

Mean, Median, Mode

Python — vi stats.py — 47×19

```
#!/Users/deepak/opt/anaconda3/bin/python3

####################################
#Statistics
####################################

import statistics

l=[9,27,18,47,27,9,81,18,27]
mean = statistics.mean(l)
median = statistics.median(l)
mode = statistics.mode(l)

print("The mean is: ",mean)
print("The median is: ",median)
print("The mode is: ",mode)
~
~
"stats.py" 16L, 352C
```

Python — -bash — 54×7

```
(base) Deepaks-MacBook-Pro:Python deepak$ ./stats.py
The mean is:  29.22222222222222
The median is:  27
The mode is:  27
(base) Deepaks-MacBook-Pro:Python deepak$
```

Delete 27

Error: No unique mode (3 equal values)

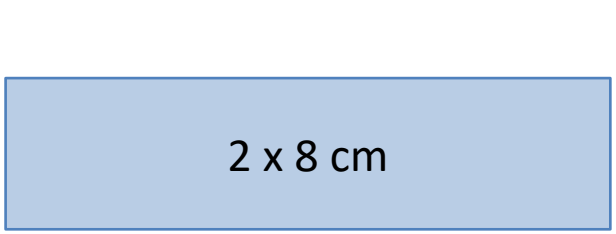
??

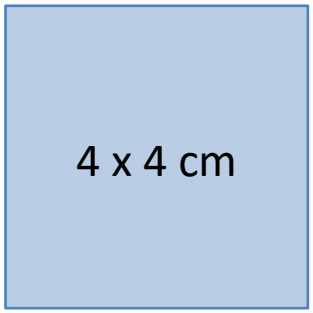
Geometric Mean

- Useful whenever the quantities to be averaged combine multiplicatively

The **geometric mean** is the antilogarithm of $\overline{\log x}$, where

$$\overline{\log x} = \frac{1}{n}\sum_{i=1}^{n} \log x_i$$





GM: 4 cm

Measures of variability

The **range** is the difference between the largest and smallest observations in a sample.

The **sample variance**, or **variance**, is defined as follows:

$$s^2 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}$$

The **sample standard deviation**, or **standard deviation**, is defined as follows:

$$s = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2}{n-1}} = \sqrt{\text{sample variance}}$$

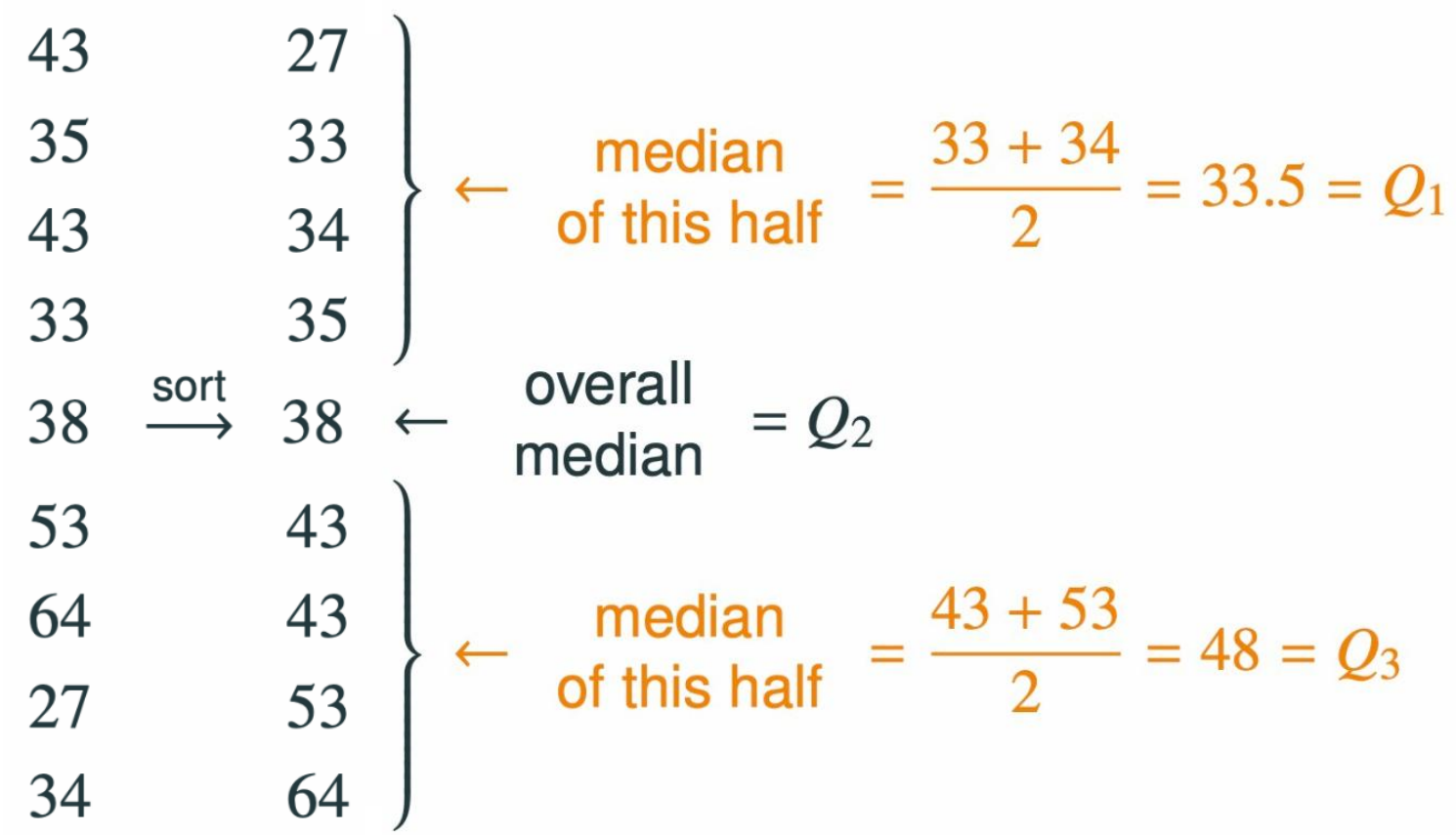
- In statistics, the number of degrees of freedom is the number of values in the final calculation of a statistic that are free to vary.
- For instance, in the above calculations, the degrees of freedom is equal to the number of independent scores (N) minus the number of parameters estimated as intermediate steps (one, namely, the sample mean) and is therefore equal to N–1.

Quartiles, IQR, Five-number summary

- Quartiles divide data into 4 even parts:
  - First quartile Q1 = 25th percentile
    - 25% of data fall below it and 75% above it
  - Second quartile Q2 = median = 50th percentile
  - Third quartile Q3 = 75th percentile
    - 75% of data fall below it and 25% above it
- Interquartile Range (IQR) = Q3 – Q1
- Five-Number Summary: min, Q1, Median, Q3, max
- Outlier: more than 1.5 × IQR below Q1 and above Q3

Example

- For the 9 numbers: 43, 35, 43, 33, 38, 53, 64, 27, 34



Example

- For the 9 numbers: 43, 35, 43, 33, 38, 53, 64, 27, 34

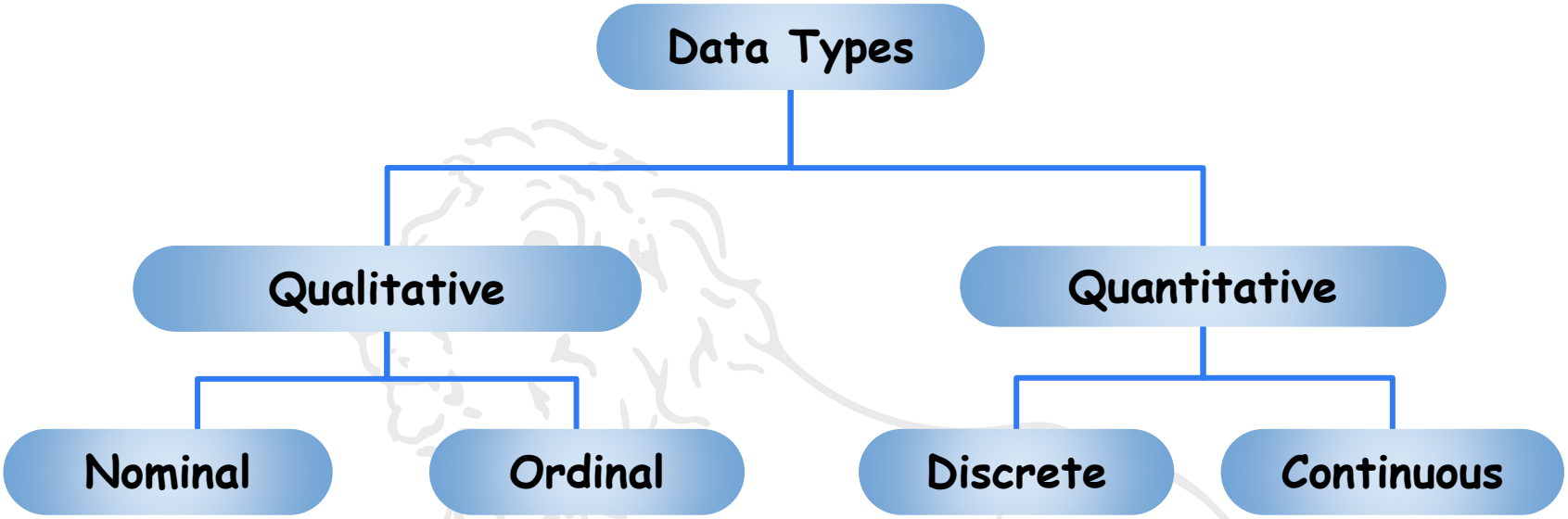
43		27
35		33
43		34
33		35
38	sort ⟶	38
53		43
64		43
27		53
34		64

IQR = Q3 − Q1 = 48 − 33.5 = 14.5

Five number summary: 27, 33.5, 38, 48, 64

Are there any outliers?

Data Types


Data Types
Qualitative
Quantitative
Nominal
Ordinal
Discrete
Continuous

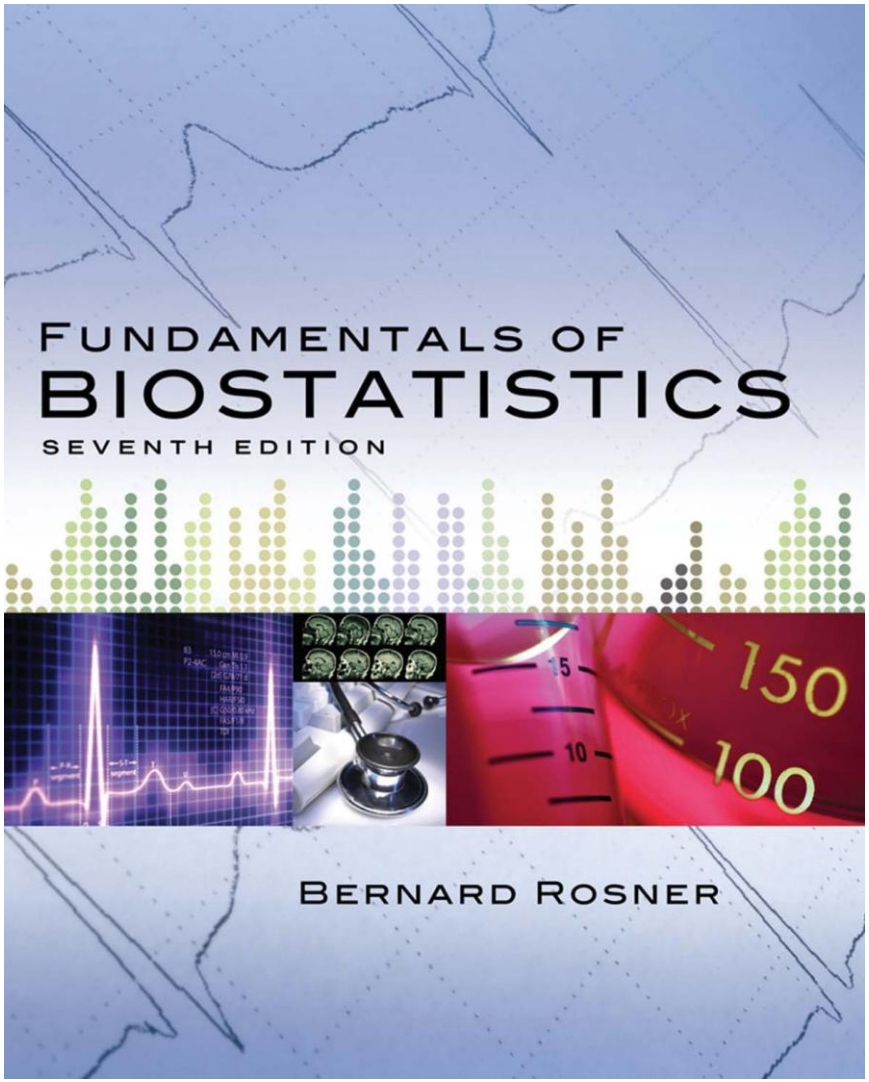
Qualitative data

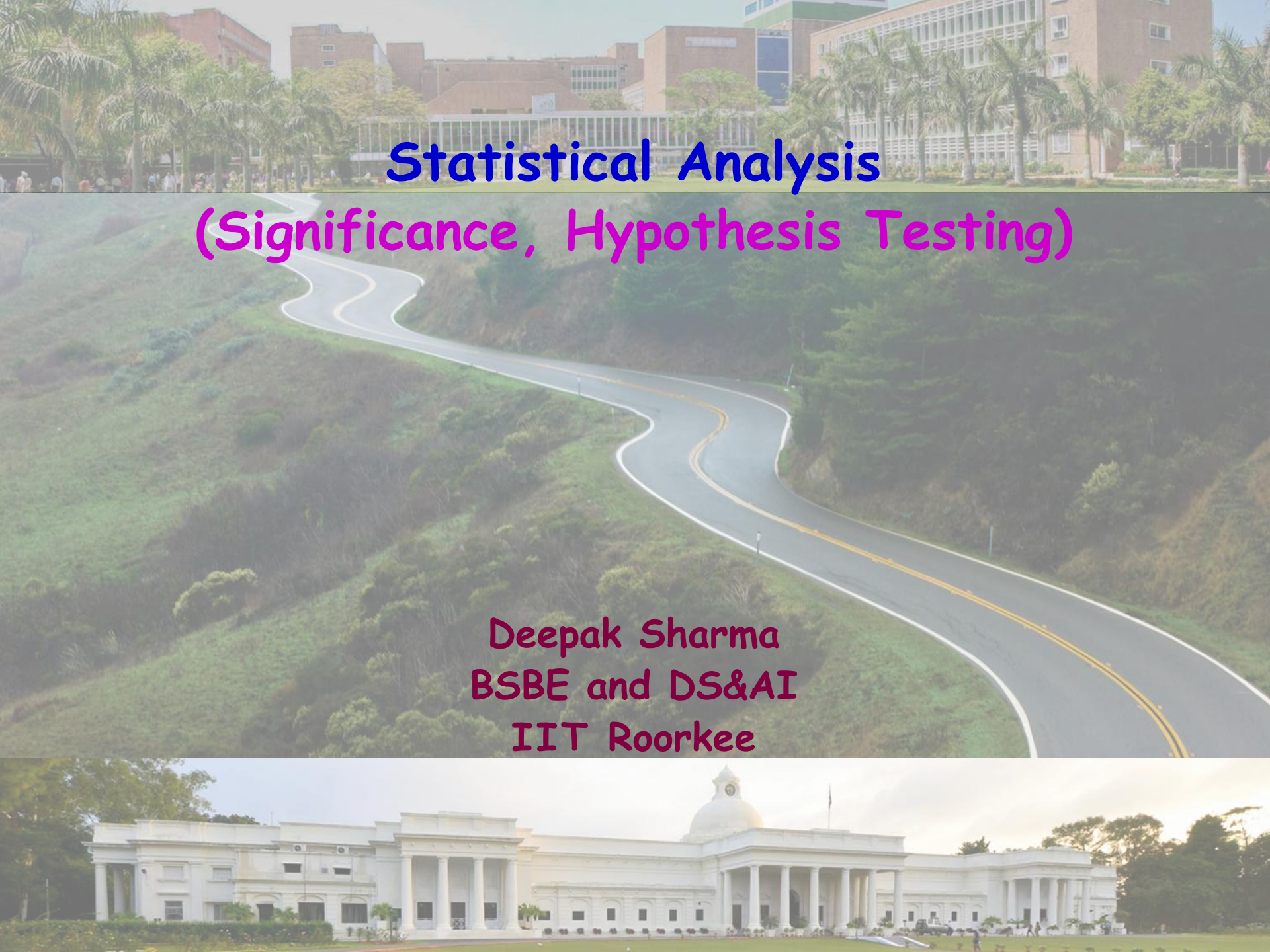
- Qualitative or categorical data: tells the features of the data and categorizes the data into various categories.
  - Nominal
    - Consists of categories or names that cannot be ordered or ranked.
    - Often used to categorize observations into groups, and the groups are not comparable.
    - Eg. gender (Male or female), race (White, Black, Asian), religion (Hinduism, Christianity, Islam, Jainism), blood type (A, B, AB, O), etc.
  - Ordinal
    - Consists of categories that can be ordered or ranked.
    - However, the distance between categories is not necessarily equal.
    - Eg. education level (Elementary, Middle, High School, College), job position (Manager, Supervisor, Employee), etc.

Quantitative data

- Quantitative data: has numerical values.
  - Discrete
    - Has discrete values (whole numbers, integers).
    - Eg. number of students, runs scored, etc.
  - Continuous
    - Represents the data in a continuous range.
    - Eg. temperature range, salary, etc.

Book


FUNDAMENTALS OF
BIOSTATISTICS
SEVENTH EDITION
150
100
BERNARD ROSNER



Statistical Analysis

(Significance, Hypothesis Testing)

Deepak Sharma
BSBE and DS&AI
IIT Roorkee

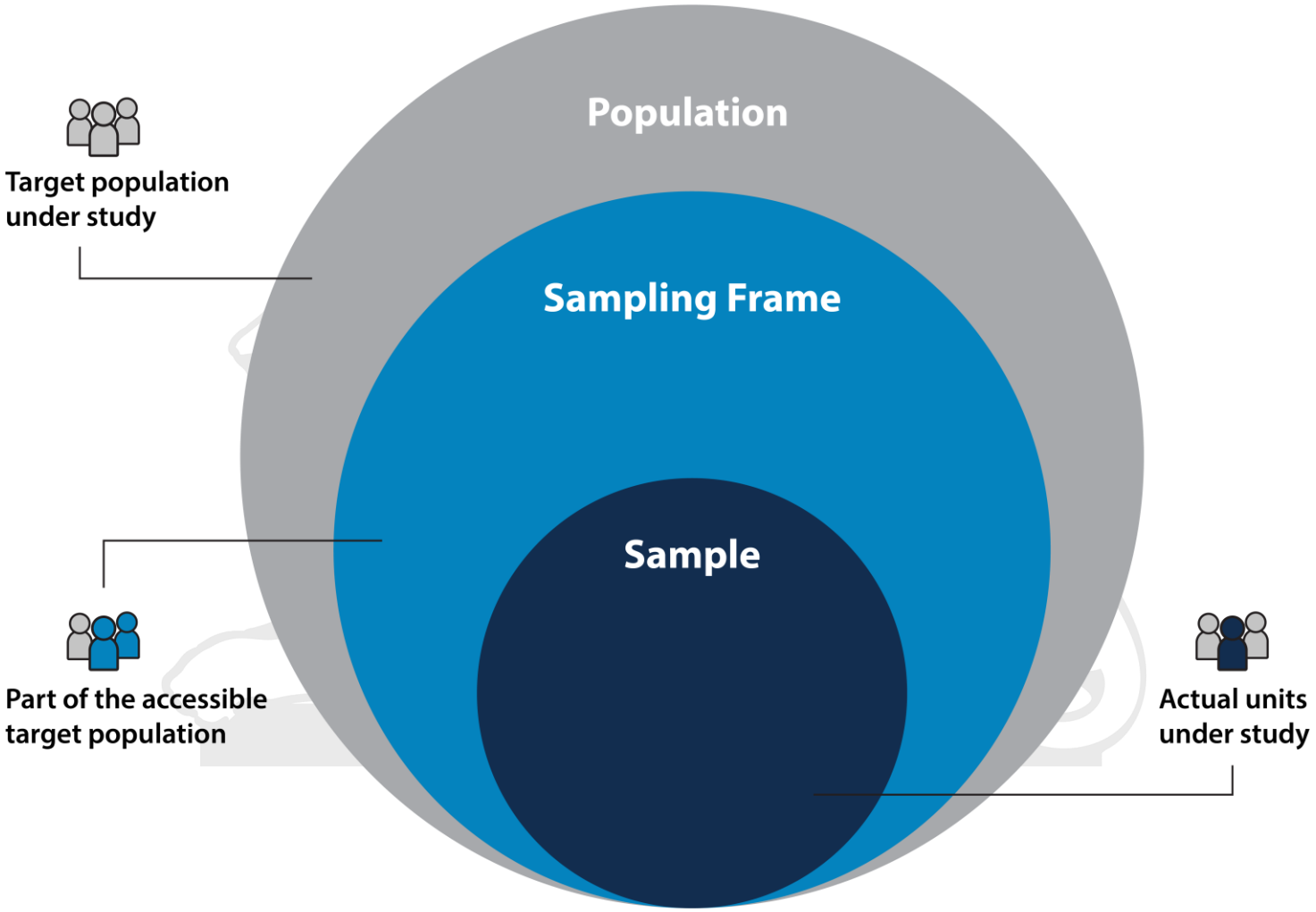
Sample, Population

A random sample is a selection of some members of the population such that each member is independently chosen and has a known nonzero probability of being selected.

A simple random sample is a random sample in which each group member has the same probability of being selected.

The **reference**, **target**, or **study population** is the group we want to study. The random sample is selected from the study population.

Sampling


Population
Target population under study
Sampling Frame
Sample
Part of the accessible target population
Actual units under study

Parameter vs Static

Parameter

- A numerical value summarizing a characteristic of a population (e.g., population mean).

Statistic

- A numerical value summarizing a characteristic of a sample (e.g., sample mean).

Note: Parameter is fixed; Statistic varies across samples.

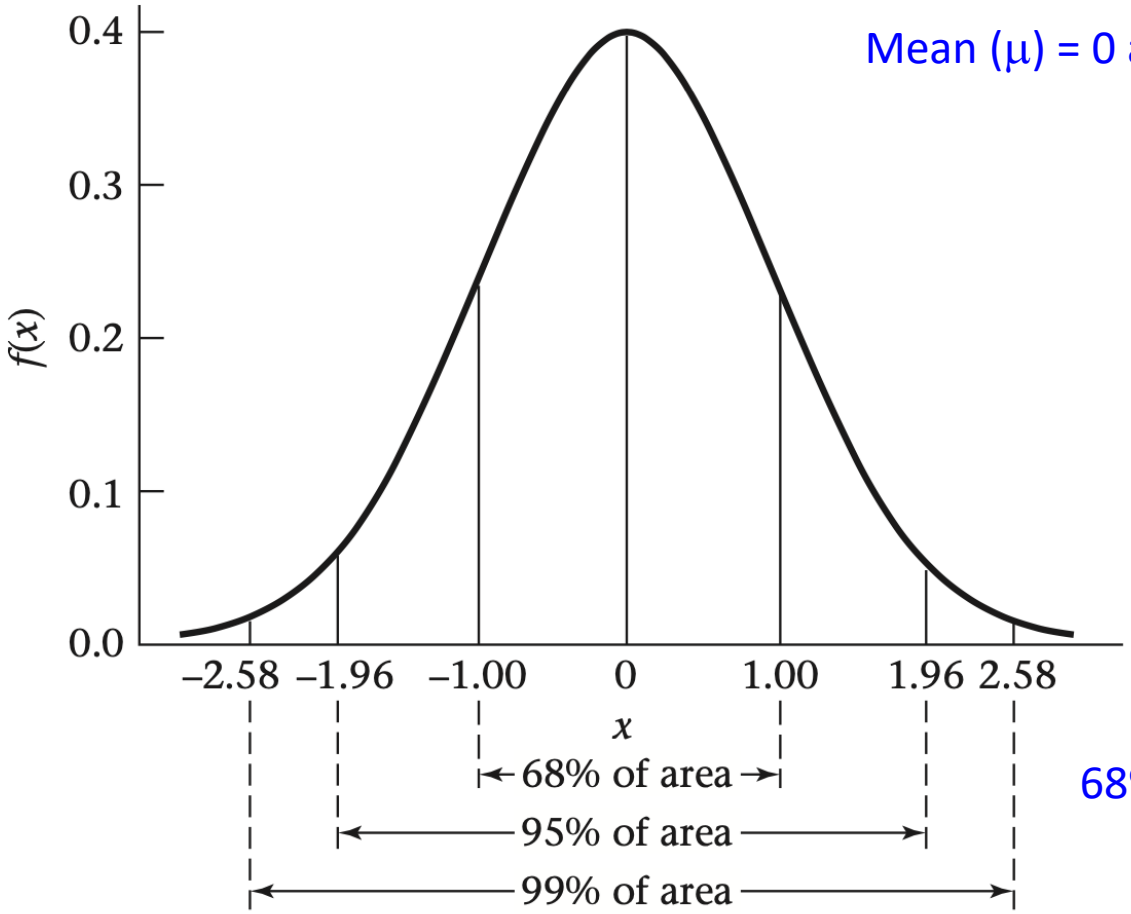
Example

- Population mean height of all adults = 170 cm (Parameter).
- Mean height from a sample of 100 adults = 168 cm (Statistic).

Statistical significance

- An important aspect of sampling theory is the study of tests of significance, which enables us to decide:
  - A result is statistically significant i.e. it is unlikely to have occurred by chance alone, given a predefined significance level.
  - Statistical significance depends on the p-value.
  - p value is probability of obtaining test results at least as extreme as the observed results, assuming the null hypothesis is true.
  - p-values are calculated using different probability distributions depending on the test. A significant result is when the p-value is less than the chosen level of significance (usually 0.05). Example: Evaluating whether a new algorithm performs better than the existing one.
- Null Hypothesis (H_0): No difference in performance.
- p-value = 0.02 indicates evidence to reject the null hypothesis.

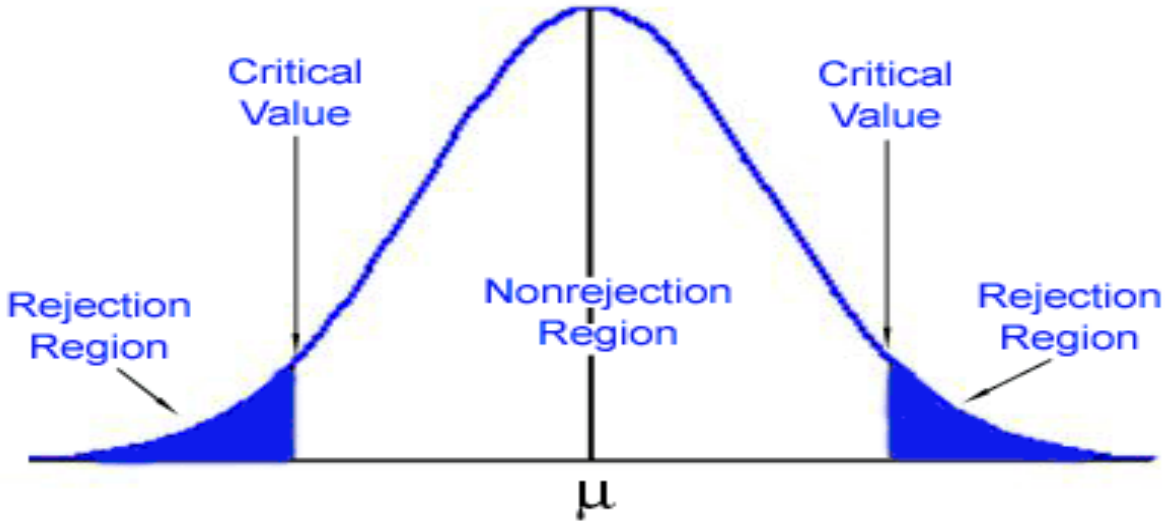
Standard normal distribution



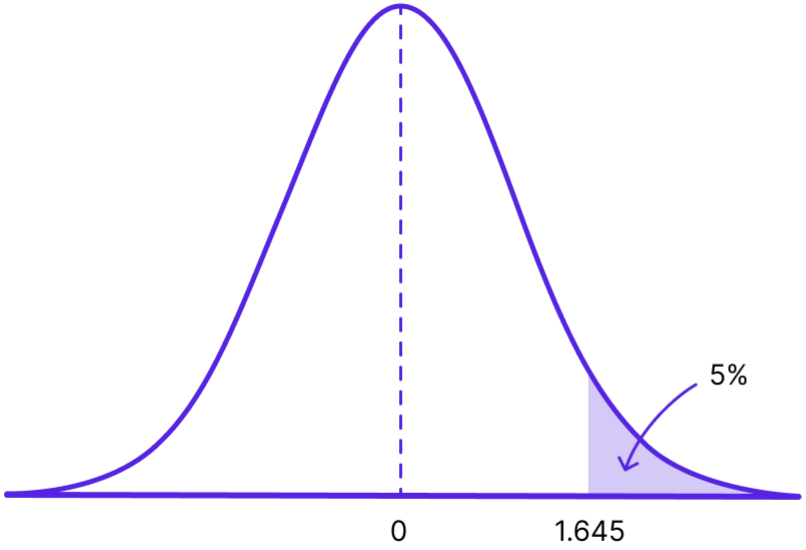
Mean (μ) = 0 and SD (σ) = 1

68% and 95% rule

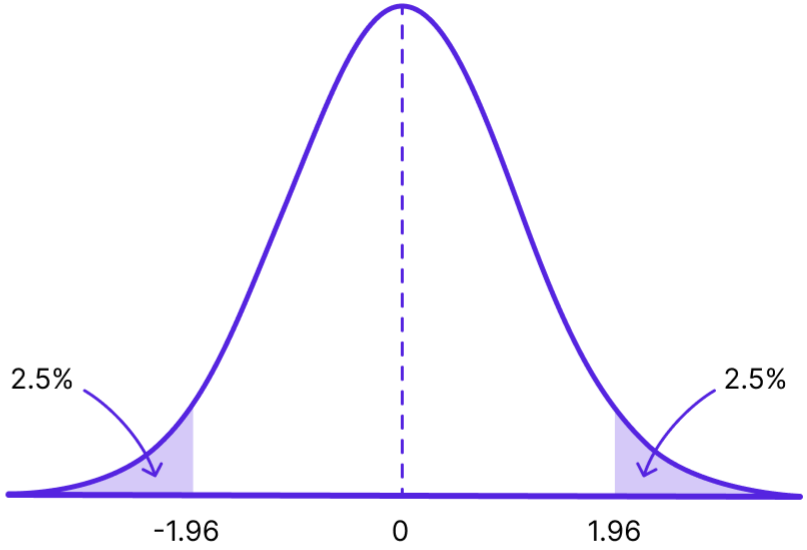
Hypothesis testing



Hypothesis testing



One-tailed test



Two-tailed test

Hypothesis testing

- Method of making decisions or inferences about a population based on sample data
- For applying the test of significance, we first set up a hypothesis:
  - Null Hypothesis (H_0): Assumes no effect or difference.
  - Alternative Hypothesis (H_1): Assumes an effect or difference exists.

Hypothesis testing, Error

Decision	Truth	
	H_0	H_1
Accept H_0	H_0 is true and H_0 is accepted	H_1 is true and H_0 is accepted
Reject H_0	H_0 is true and H_0 is rejected	H_1 is true and H_0 is rejected

Type II error Cancer

Type I error TB HIV

The probability of a **type I error** is the probability of rejecting the null hypothesis when H_0 is true.

The probability of a **type II error** is the probability of accepting the null hypothesis when H_1 is true.

Parametric vs. Non-parametric tests

Feature	Parametric test	Non-parametric test
Definition	Based on parameters (eg. Mean, variance, SD)	Not based on any parameter
Data distribution	Assumes normal/specific distribution	No assumption
Efficiency	More powerful if assumption holds	Less powerful but more flexible
Precision	More precise when assumption holds	Less precise but robust
Rule of thumb for use	**Mean/2 > SD**	**Mean/2 < SD**

Parametric tests

- Involves inference about the population parameters.

One sample test (One sided alternatives)

- Suppose we want to test the hypothesis that mothers with low socio- economic status (SES) deliver babies whose birthweights are lower than "normal."
- To test this hypothesis, a list is obtained of birthweights from 100 deliveries in a low-SES area. The mean birthweight ($\bar{x}$) is found to be 3.26 kg with a SD (s) of 0.68 kg. We know that the national mean birthweight is 3.4 kg (μ_0). Can we say the underlying mean birthweight from low-SES area is lower than the national average?

The **acceptance region** is the range of values of $\bar{x}$ for which H_0 is accepted.

The **rejection region** is the range of values of $\bar{x}$ for which H_0 is rejected.

A **one-tailed test** is a test in which the values of the parameter being studied (in this case μ) under the alternative hypothesis are allowed to be either greater than or less than the values of the parameter under the null hypothesis (μ_0), *but not both*.

One sample test (One sided alternatives)

- Suppose we want to test the hypothesis that mothers with low socio- economic status (SES) deliver babies whose birthweights are lower than "normal."
- To test this hypothesis, a list is obtained of birthweights from 100 deliveries in a low-SES area. The mean birthweight ($\bar{x}$) is found to be 3.26 kg with a SD (s) of 0.68 kg. We know that the national mean birthweight is 3.4 kg (μ_0). Can we say the underlying mean birthweight from low-SES area is lower than the national average?

One sample t-test:

$$t = \frac{\bar{x} - \mu_0}{s/\sqrt{n}} = \frac{3.26 - 3.4}{0.68/\sqrt{100}} = -2.05$$

$t_{99,\ 0.05}$ = -1.66 (number of freedom, n-1)

t < -1.66, Reject H_0 (Different, lower)

t distribution

Table 5 Percentage points of the t distribution ($t_{d,u}$)[a]

Degrees of freedom, d	u = .75	.80	.85	.90	.95	.975	.99	.995	.9995
1	1.000	1.376	1.963	3.078	6.314	12.706	31.821	63.657	636.619
2	0.816	1.061	1.386	1.886	2.920	4.303	6.965	9.925	31.598
3	0.765	0.978	1.250	1.638	2.353	3.182	4.541	5.841	12.924
4	0.741	0.941	1.190	1.533	2.132	2.776	3.747	4.604	8.610
5	0.727	0.920	1.156	1.476	2.015	2.571	3.365	4.032	6.869
6	0.718	0.906	1.134	1.440	1.943	2.447	3.143	3.707	5.959
7	0.711	0.896	1.119	1.415	1.895	2.365	2.998	3.499	5.408
8	0.706	0.889	1.108	1.397	1.860	2.306	2.896	3.355	5.041
9	0.703	0.883	1.100	1.383	1.833	2.262	2.821	3.250	4.781
10	0.700	0.879	1.093	1.372	1.812	2.228	2.764	3.169	4.587
11	0.697	0.876	1.088	1.363	1.796	2.201	2.718	3.106	4.437
12	0.695	0.873	1.083	1.356	1.782	2.179	2.681	3.055	4.318
13	0.694	0.870	1.079	1.350	1.771	2.160	2.650	3.012	4.221
14	0.692	0.868	1.076	1.345	1.761	2.145	2.624	2.977	4.140
15	0.691	0.866	1.074	1.341	1.753	2.131	2.602	2.947	4.073
16	0.690	0.865	1.071	1.337	1.746	2.120	2.583	2.921	4.015
17	0.689	0.863	1.069	1.333	1.740	2.110	2.567	2.898	3.965
18	0.688	0.862	1.067	1.330	1.734	2.101	2.552	2.878	3.922
19	0.688	0.861	1.066	1.328	1.729	2.093	2.539	2.861	3.883
20	0.687	0.860	1.064	1.325	1.725	2.086	2.528	2.845	3.850
21	0.686	0.859	1.063	1.323	1.721	2.080	2.518	2.831	3.819
22	0.686	0.858	1.061	1.321	1.717	2.074	2.508	2.819	3.792
23	0.685	0.858	1.060	1.319	1.714	2.069	2.500	2.807	3.767
24	0.685	0.857	1.059	1.318	1.711	2.064	2.492	2.797	3.745
25	0.684	0.856	1.058	1.316	1.708	2.060	2.485	2.787	3.725
26	0.684	0.856	1.058	1.315	1.706	2.056	2.479	2.779	3.707
27	0.684	0.855	1.057	1.314	1.703	2.052	2.473	2.771	3.690
28	0.683	0.855	1.056	1.313	1.701	2.048	2.467	2.763	3.674
29	0.683	0.854	1.055	1.311	1.699	2.045	2.462	2.756	3.659
30	0.683	0.854	1.055	1.310	1.697	2.042	2.457	2.750	3.646
40	0.681	0.851	1.050	1.303	1.684	2.021	2.423	2.704	3.551
60	0.679	0.848	1.046	1.296	1.671	2.000	2.390	2.660	3.460
120	0.677	0.845	1.041	1.289	1.658	1.980	2.358	2.617	3.373
∞	0.674	0.842	1.036	1.282	1.645	1.960	2.326	2.576	3.291

[a]The uth percentile of a t distribution with d degrees of freedom.

Source: Table 5 is taken from Table III of Fisher and Yates: "Statistical Tables for Biological, Agricultural and Medical Research," published by Longman Group Ltd., London (previously published by Oliver and Boyd Ltd., Edinburgh). Reprinted by permission of Pearson Education Ltd.

Non Parametric tests

- Not based on parameters (mean, variance).
- Sample size is small

Two sample Mann-Whitney test

- Compare observations from control (X) and treated (Y) groups
- H_0: X and Y have same distribution
- H_1: X and Y have different distribution

$$U = S_x - \frac{n_x (n_x + 1)}{2}$$

S_x : sum of ranks of x
n_x : number (sample size) of x

Mann Whitney test

The data shows the length of time (in days) that control mice (X) and inoculated mice (Y) live:

X:	20	23	28	31	30
Y:	20	29	23	48	41

H_0: X and Y have same distribution
H_1: X and Y have different distribution

	x	y	x	y	x	y	x	x	y	y
Arrange:	20	20	23	23	28	29	30	31	41	48
Ranks	1.5	1.5	3.5	3.5	5	6	7	8	9	10

$$U = 25 - \frac{5(5+1)}{2} = 10$$

Lower tail: 5; Upper tail:20

Since U = 10 does not fall in critical region, null hypothesis (H_o) accepted

Mann Whitney test

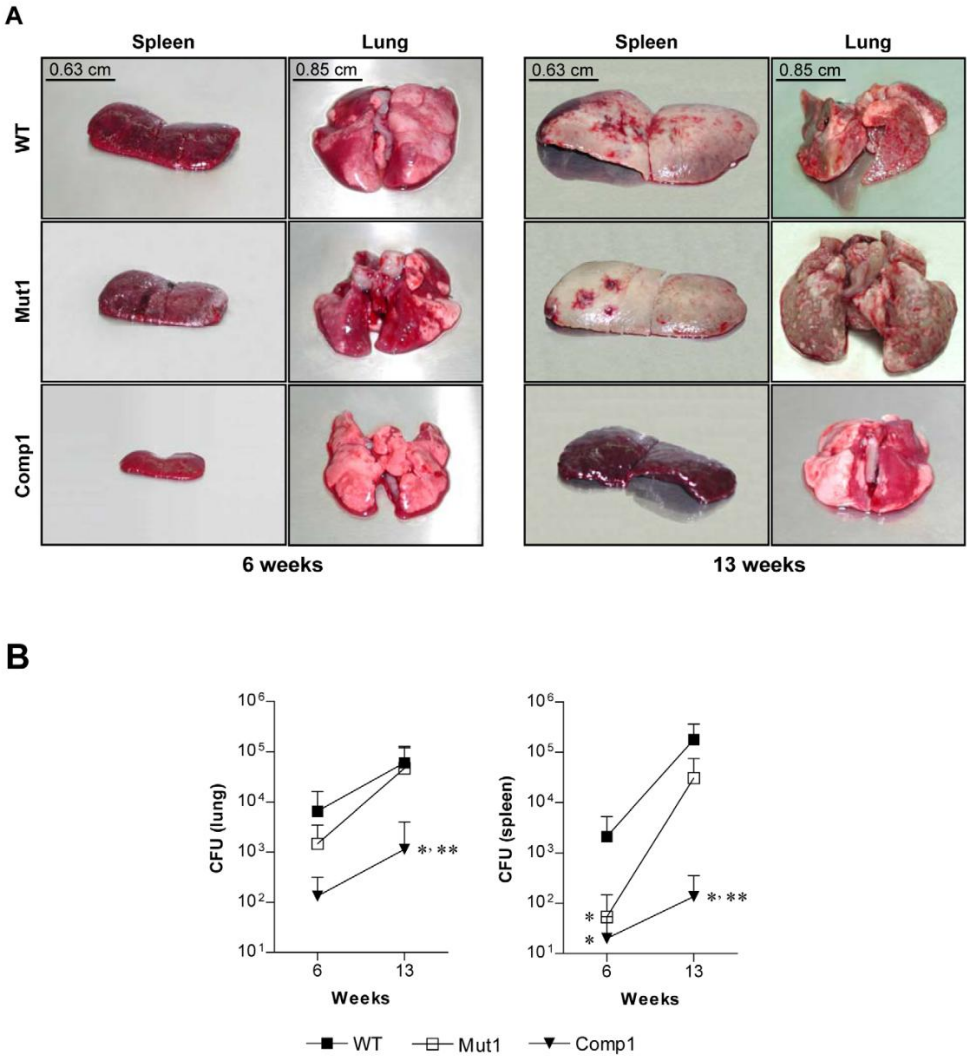


Figure 5. Virulence of passaged *M. tuberculosis* strains in guinea pigs. (**A**) Pictorial representation of lungs and spleens. (**B**) CFU in lungs and spleens are expressed as mean ± SD. *, ** represent P<0.05 in comparison to WT and Mut1, respectively.
doi:10.1371/journal.pone.0009448.g005

Majumdar SD, Sharma D *et al.* (2010) *PLoS One*, 5, e9448.

Confidence Interval

It describes the variability surrounding the sample point estimate (the wider the interval, the less confident we can be about the estimate of the population mean). In general, the larger the sample size the better (more precise) the estimate is.

- A confidence interval displays the probability that a parameter will fall between a pair of values around the mean.
- Statisticians often use p-values in conjunction with confidence intervals to gauge statistical significance.
- They are most often constructed using confidence levels of 95% or 99%.

Height of IITR students (100 samples; 5ft $\pm$ 1.5ft)
CGPA of IITR students (100 samples; 7 $\pm$ 2)

Confidence Interval

The equation for Confidence Interval for the population mean when the population standard deviation is unknown and the sample size is large (over 30) is

$$CI = \bar{x} \pm z\frac{s}{\sqrt{n}}$$

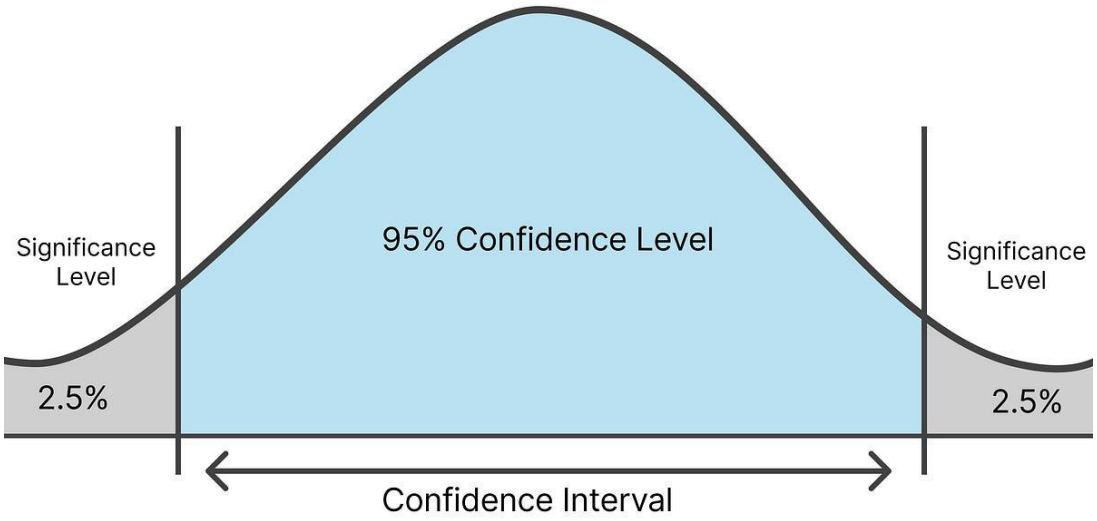
CI = confidence interval

$\bar{x}$ = sample mean

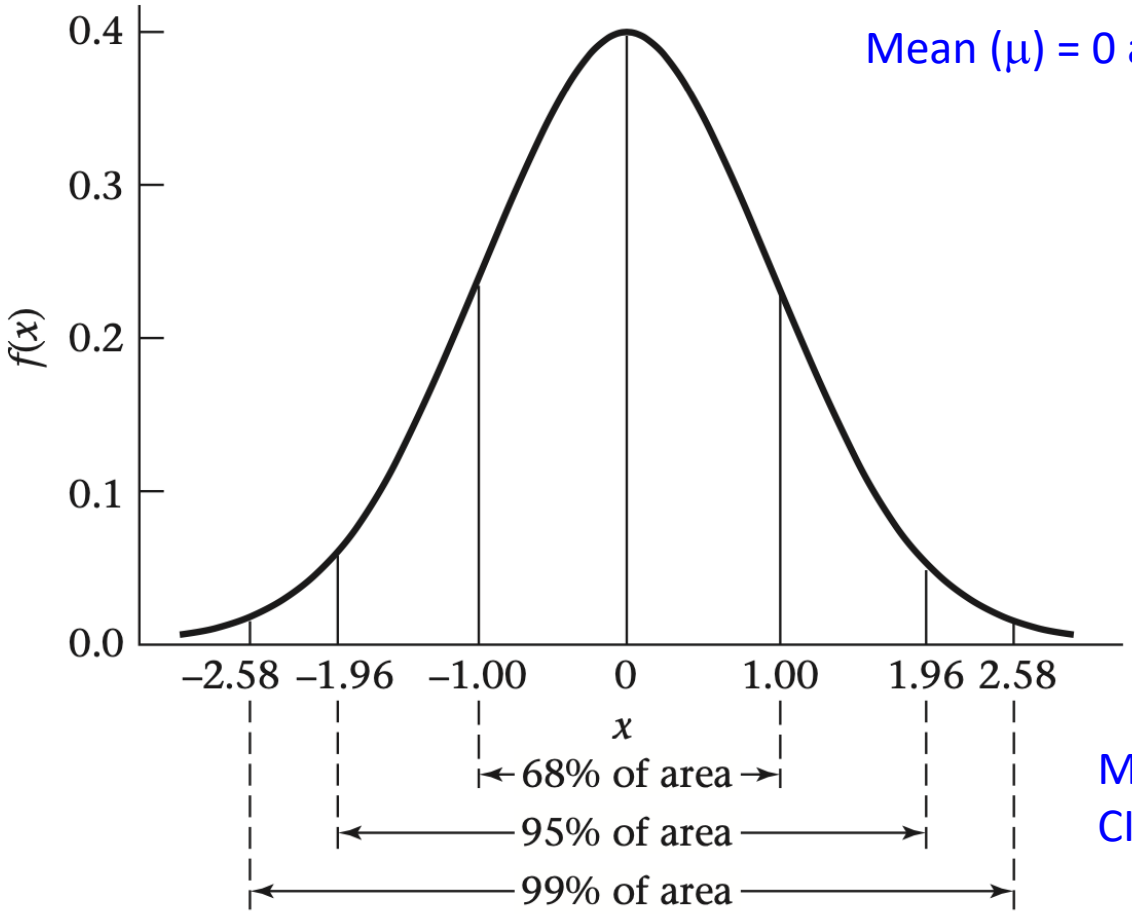
z = confidence level value

s = sample standard deviation

n = sample size



Standard normal distribution



Mean (μ) = 0 and SD (σ) = 1

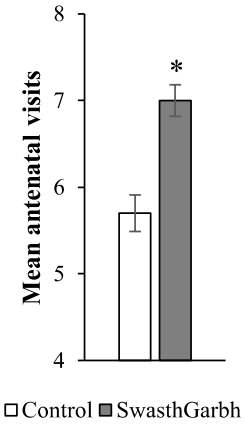
Mean ± SD

CI (~Mean ± 2SD)

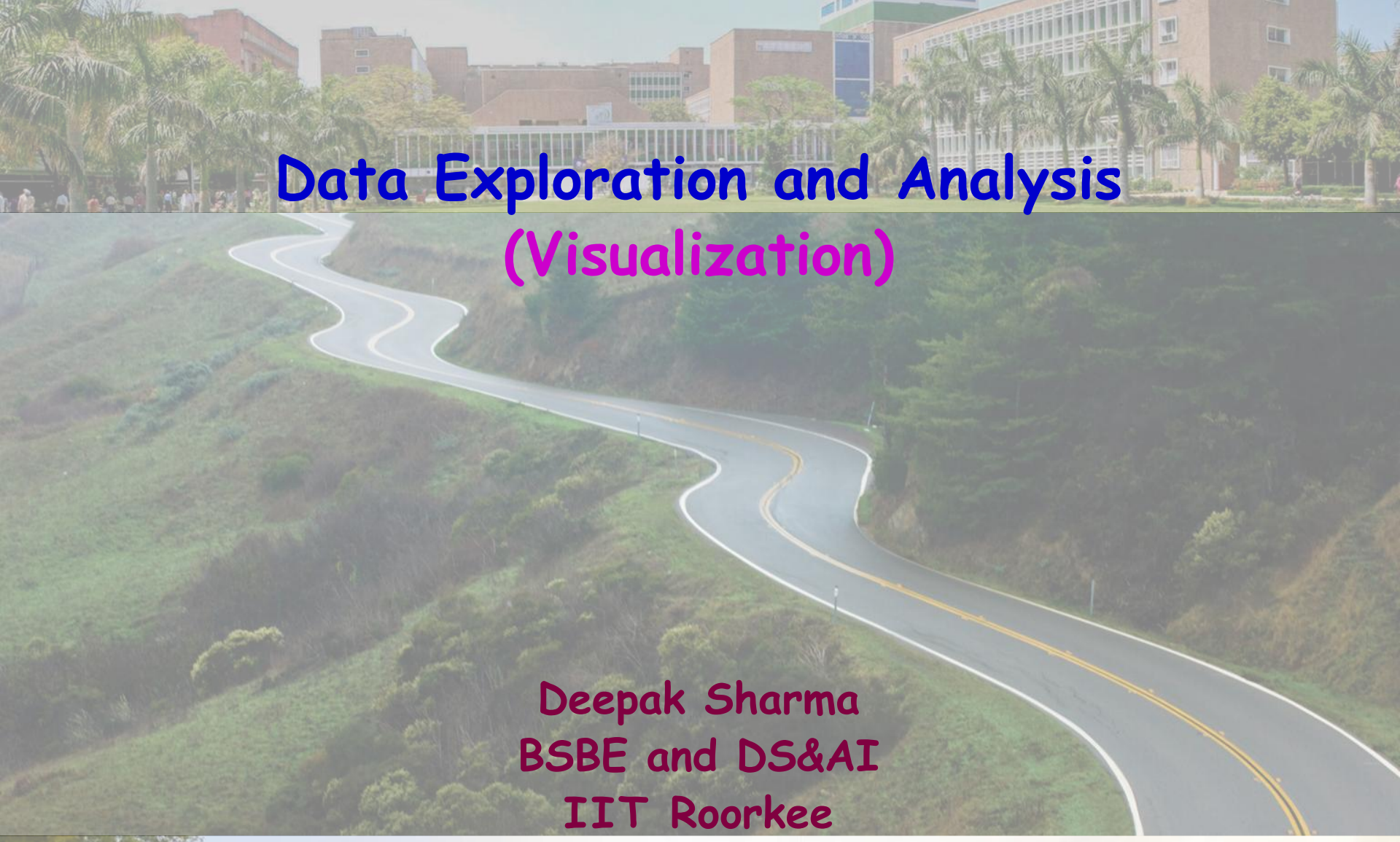
SwasthGarbh: Confidence Interval

Table S2: Relationship of treatment and socio-demographic variables on the number of ANC visits by regression analysis.

	Regression coefficient			
	Unadjusted (95% CI)	**P-value**	**Adjusted (95% CI)**	**P-value**
Mean difference in ANC visits	1.29 (0.74, 1.84)	<0.001	1.36 (0.82, 1.90)	<0.001
Age (years)	0.05 (-0.24, 0.12)	0.19		
POG at recruitment	-0.004 (0.13, 0.12)	0.95		
Higher Education	1.00 (0.43, 1.58)	0.001	0.35 (-0.31, 1.01)	
Lower Socio-economic Status	-1.06 (-1.71, -0.40)	0.002	-1.01 (-1.75, -0.27)	
Urban Area	-0.08 (-0.87, 0.71)	0.84		
Induction/IVF Conception	-0.22 (-0.92, 0.48)	0.54		
Gravidity >2	-0.23 (-0.86, 0.40)	0.48		
≥1 Abortions	-0.19 (-0.79, 0.41)	0.53		



Sharma S, Soni S *et al.* (2023) *IEEE JBHI*, 27, 2729-38.



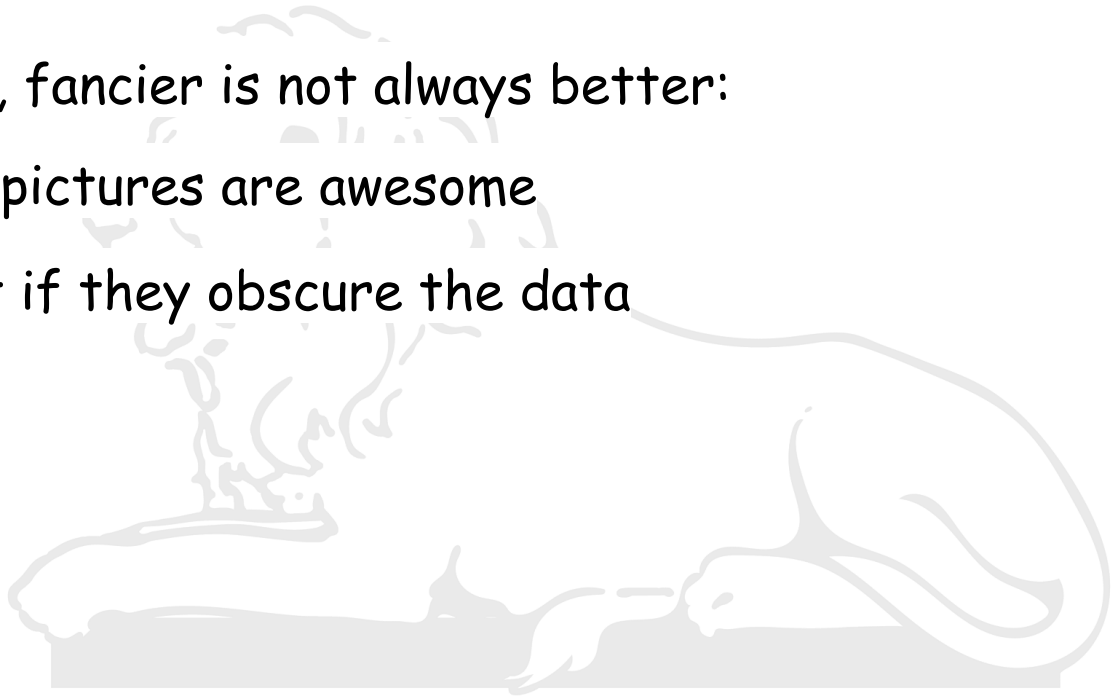
Data Exploration and Analysis

(Visualization)

Deepak Sharma
BSBE and DS&AI
IIT Roorkee

Why visualize?

- To look at the data and understand it.
- Don't just try to find differences, try to understand the factors.
- Moreover, fancier is not always better:
  - pretty pictures are awesome
  - but not if they obscure the data



Two trade-offs

- Informativeness vs. readability
  - Too little information can conceal data
  - But too much information can be overwhelming

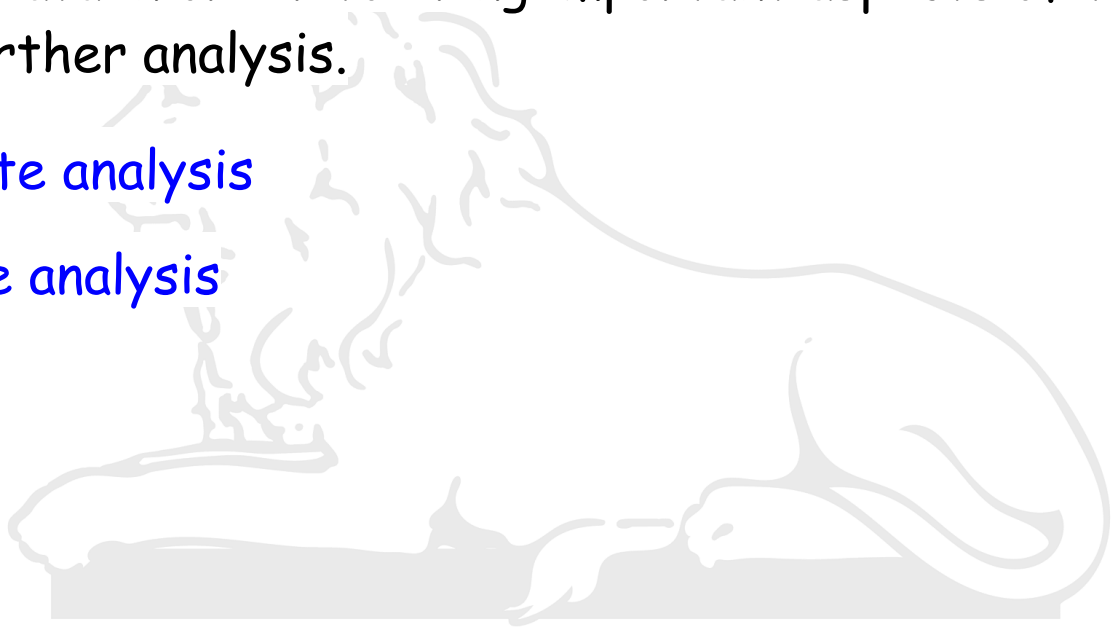
- Data-centric vs. viewer-centric
  - Viewers are accustomed to certain types of visualization
  - But novel visualizations can be truer to data

Data exploration

It is about describing the data by means of statistical and visualization techniques.

We explore data in order to bring important aspects of that data into focus for further analysis.

- Univariate analysis
- Bivariate analysis



Exploring categorical data

Single categorical variable

- Bar plots, pie charts
- Frequency tables

Relationship between two categorical variables

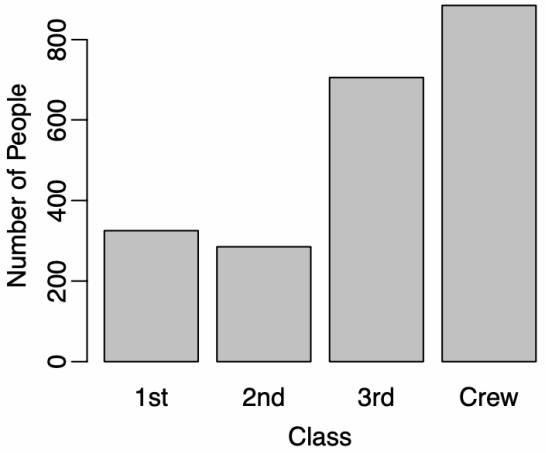
- Two-way contingency tables
- Segmented bar plots, Standardized segmented bar plots

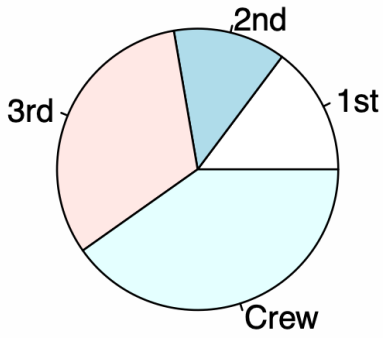
Categorical variables

A categorical variable is summarized by a table showing the count or the percentage of cases in each category, and is often displayed by a bar plot or a pie chart.

Ex: Passengers on Titanic

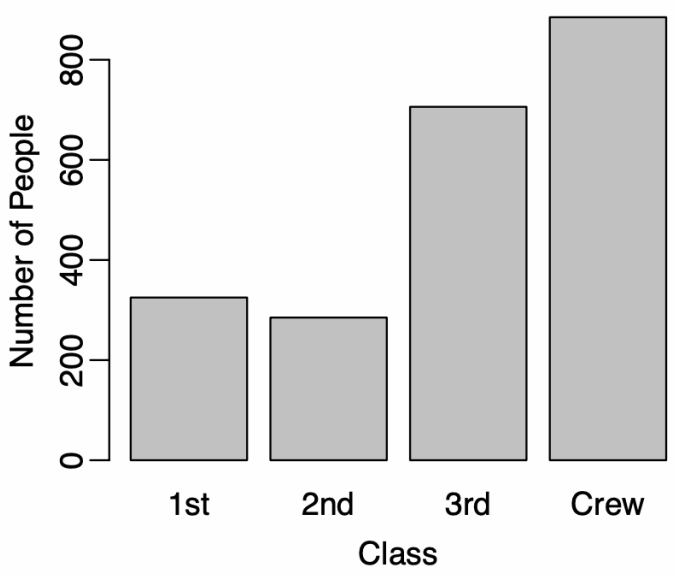
Class	Freq	Percent
1st	325	14.8%
2nd	285	12.9%
3rd	706	32.1%
Crew	885	40.2%
Total	2201	100%



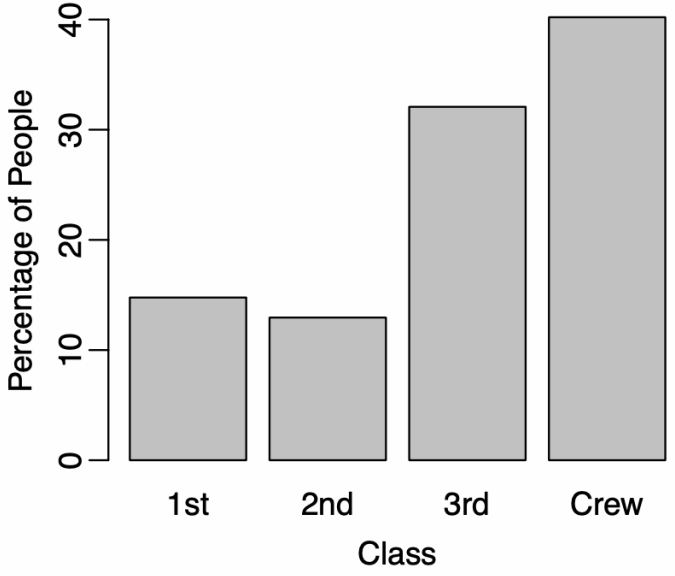


Relative frequency bar plot

A bar plot where proportions instead of frequencies are shown is called a relative frequency bar plot.



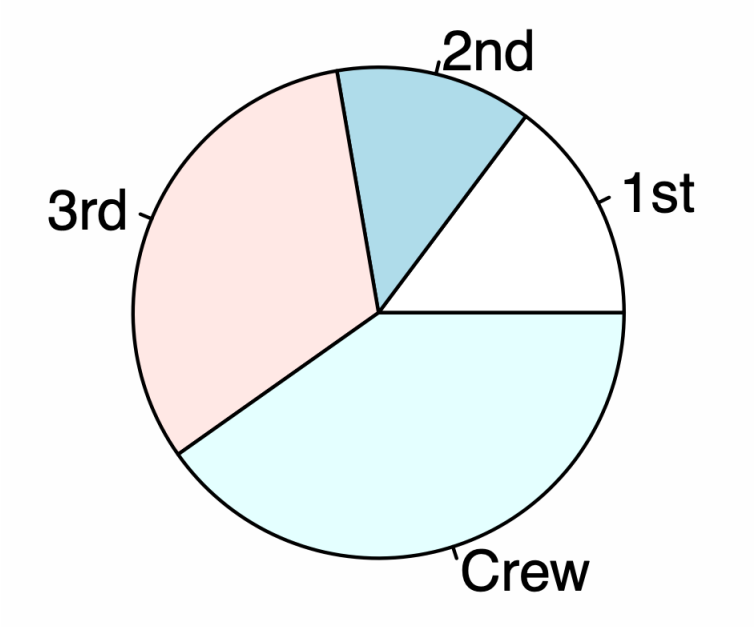
Bar Plot



Relative frequency bar plot

Disadvantage of pie charts

- In a pie chart, the areas of slices represents the percentages of categories.
- However, it is generally more difficult to compare group sizes in a pie chart than in a bar plot, especially when categories have nearly identical counts or proportions.



Two-way contingency table

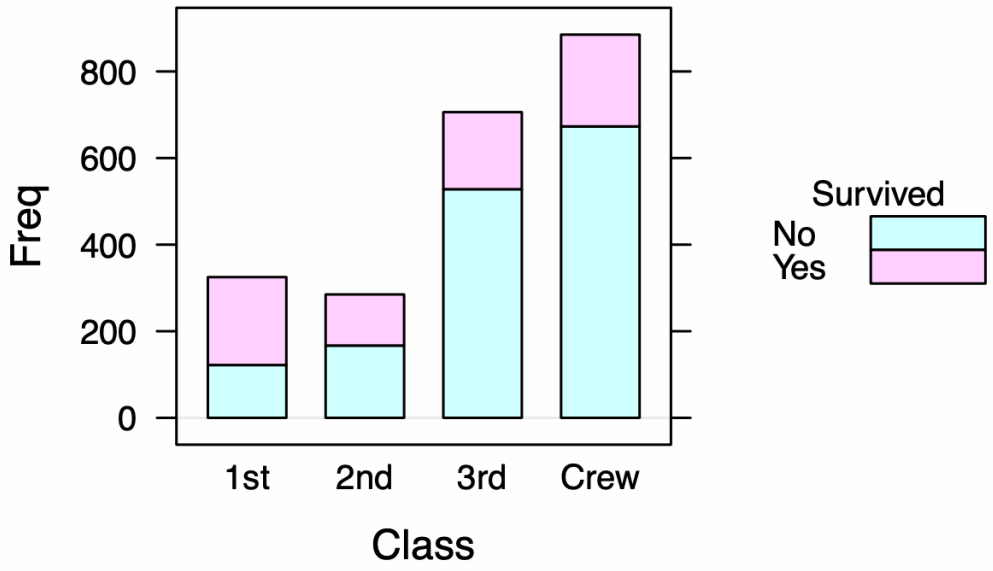
A table that summarizes data for two categorical variables is called a contingency table.

E.g., breakdown of people on Titanic by class and survival status

		Died	Survived	Total
Class	1st	122	203	325
	2nd	167	118	285
	3rd	528	178	706
	Crew	673	212	885
	Sum	1490	711	2201

Segmented bar plot

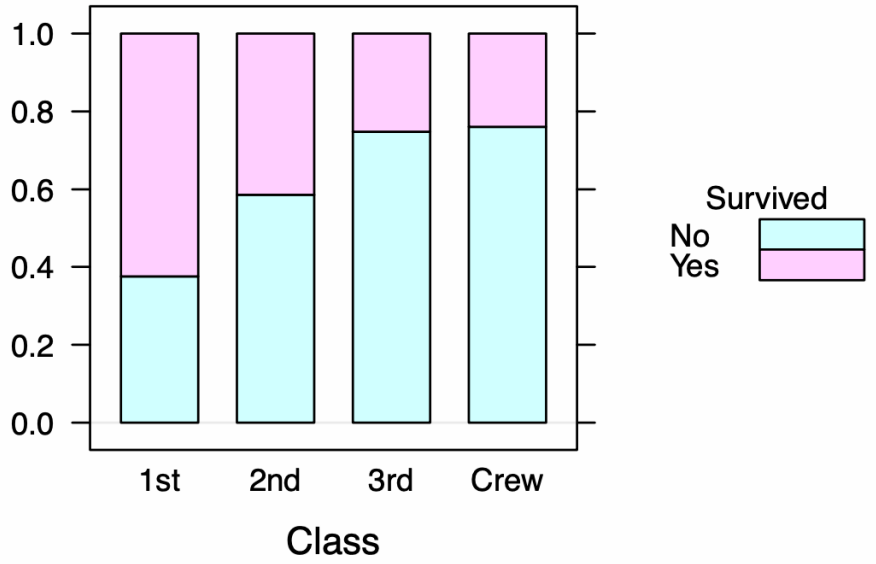
	Survived		
Class	No	Yes	Total
1st	122	203	325
2nd	167	118	285
3rd	528	178	706
Crew	673	212	885
Sum	1490	711	2201



Standardized segmented bar plot

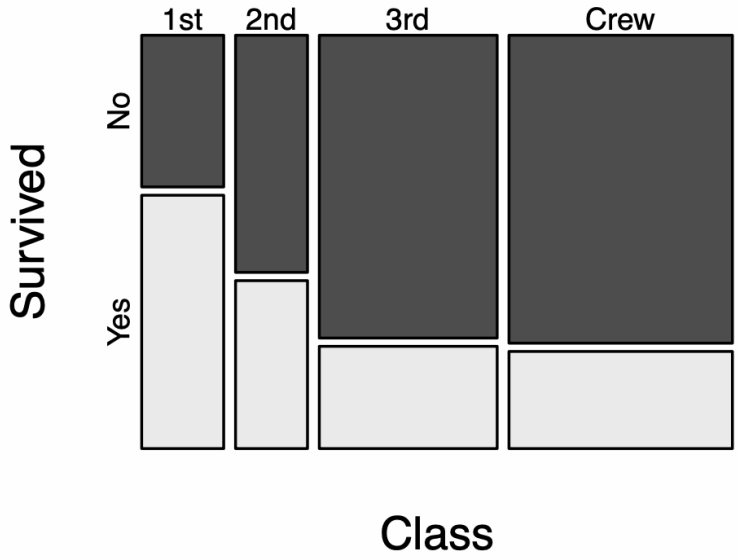
- Standardized segmented bar plots are convenient for comparing row proportions, and determining whether the two variables are independent.
- However, the information of row totals is lost after standardization.

		Survived		
		No	Yes	Total
Class	1st	0.38	0.62	1
	2nd	0.59	0.41	1
	3rd	0.75	0.25	1
	Crew	0.76	0.24	1



Mosaic Plots

- Bar widths = Row totals
- Segmented lengths within a bar = Row proportions



$$\boxed{\text{segment area}} = (\text{barwidth}) \times (\text{segment length})$$

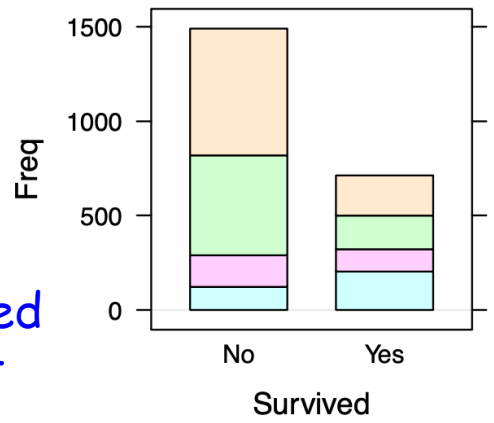
$$= \text{row total} \times (\text{row proportion})$$

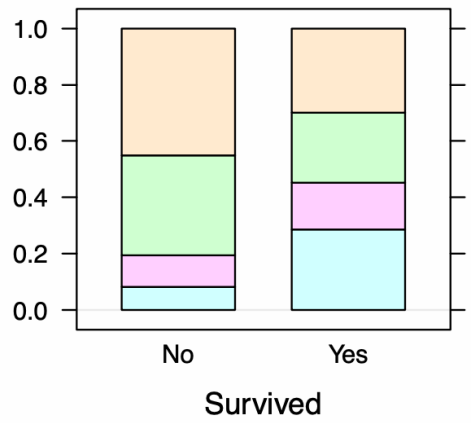
$$= \text{row total} \times \frac{\text{cell count}}{\text{row total}} = \boxed{\text{cell count}}$$

Comparative view of different plots

Instead of survival rates in the four classes, we can also plot as those who survived or died.

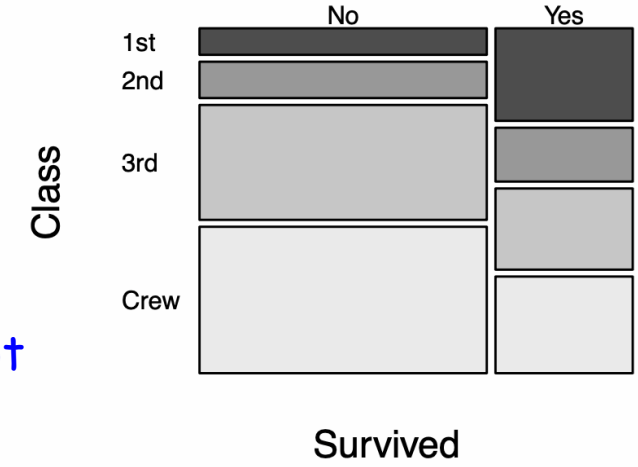
Segmented bar plot





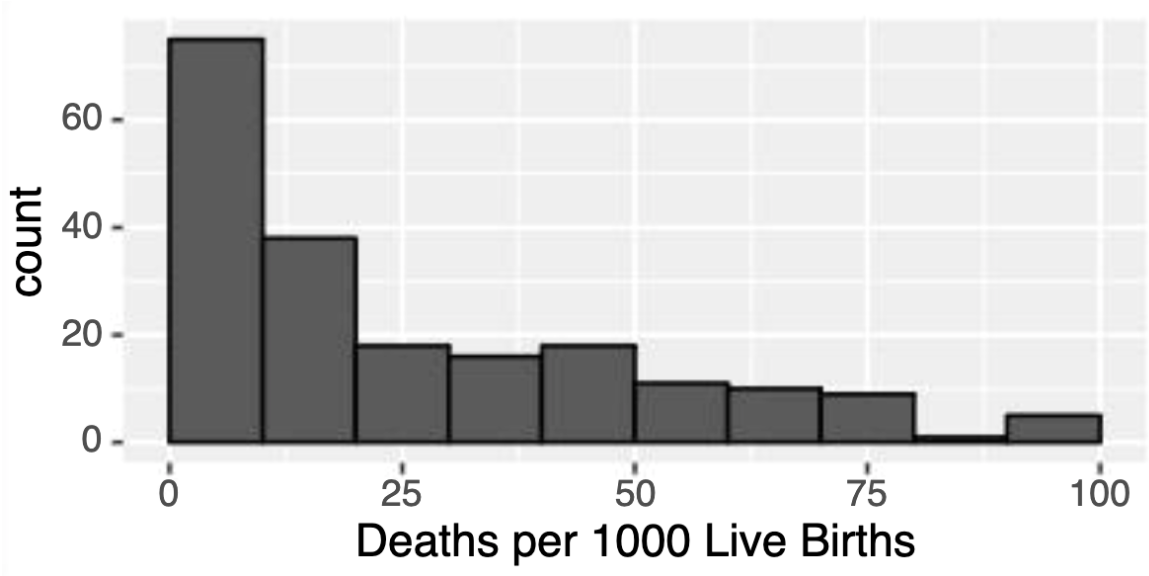
Standardized segmented bar plot

Mosaic plot

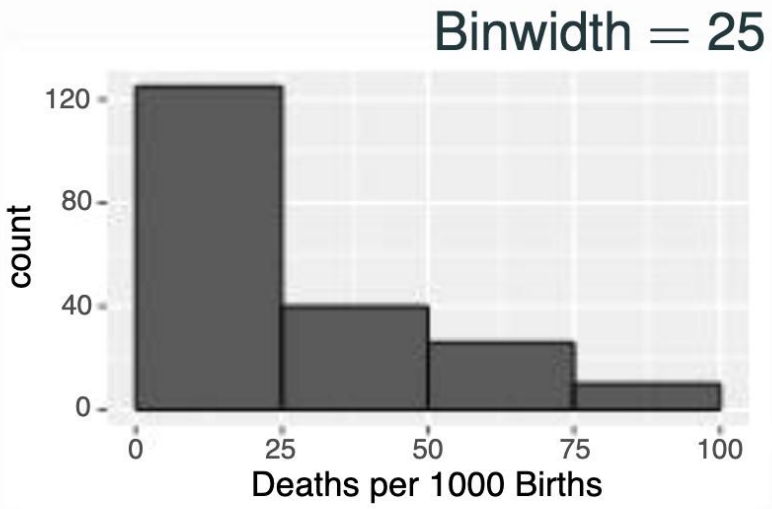


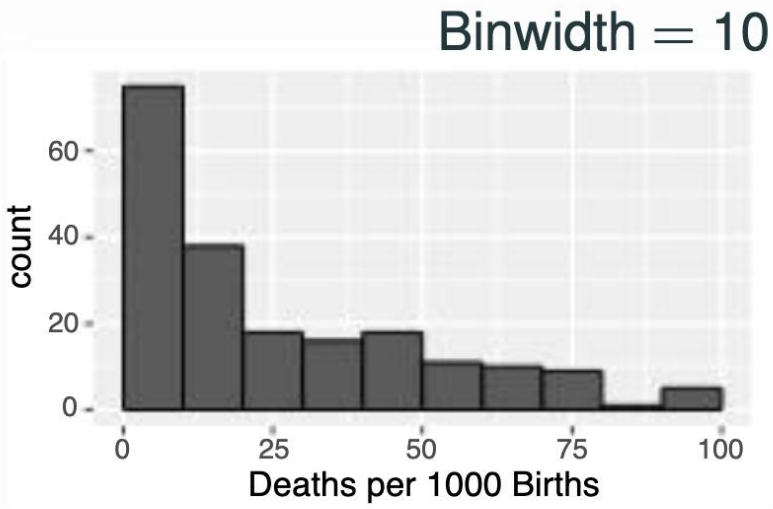
Numerical data: Histograms

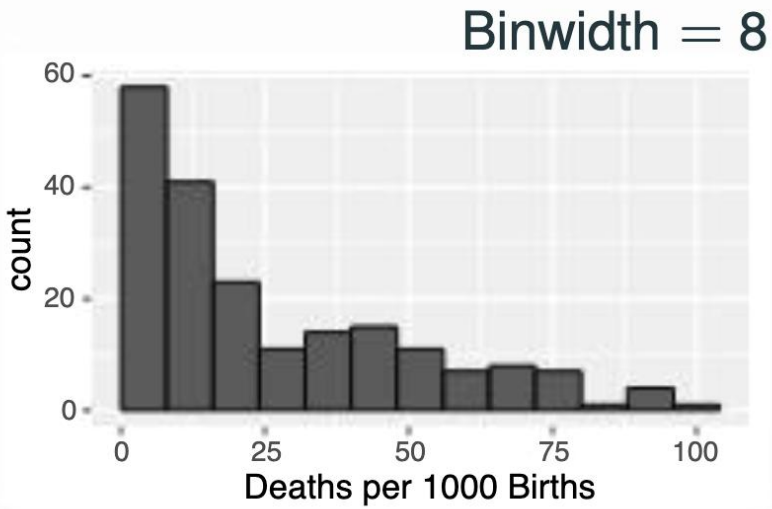
- Histograms are especially convenient for providing a view of the data density, and the shape of the data distribution.
- Higher bars indicate higher density region (more observations).
- The selection of bin width (width of the class intervals) can alter the shape of histogram.

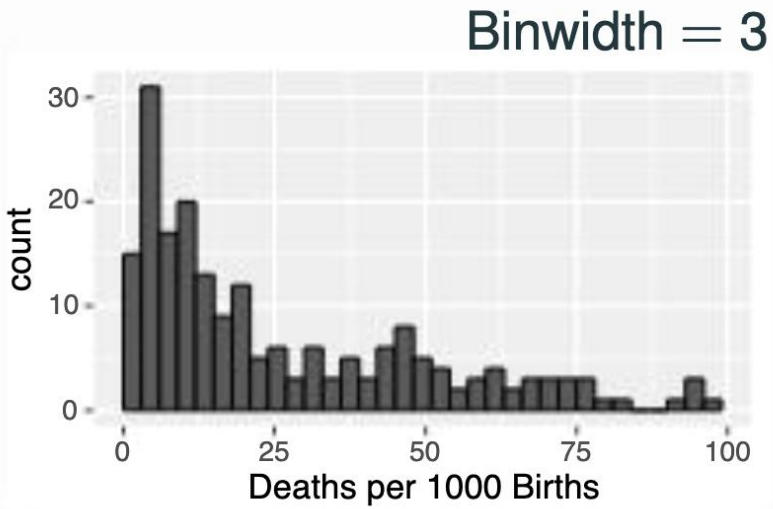


Different bin width





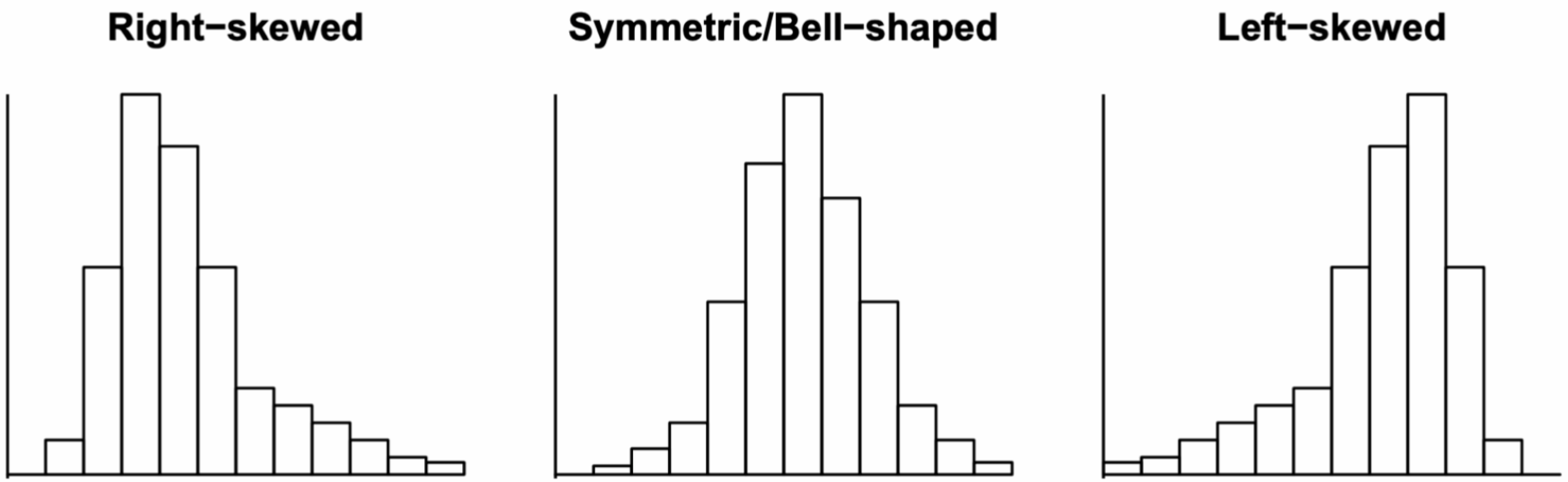




Choosing bin width

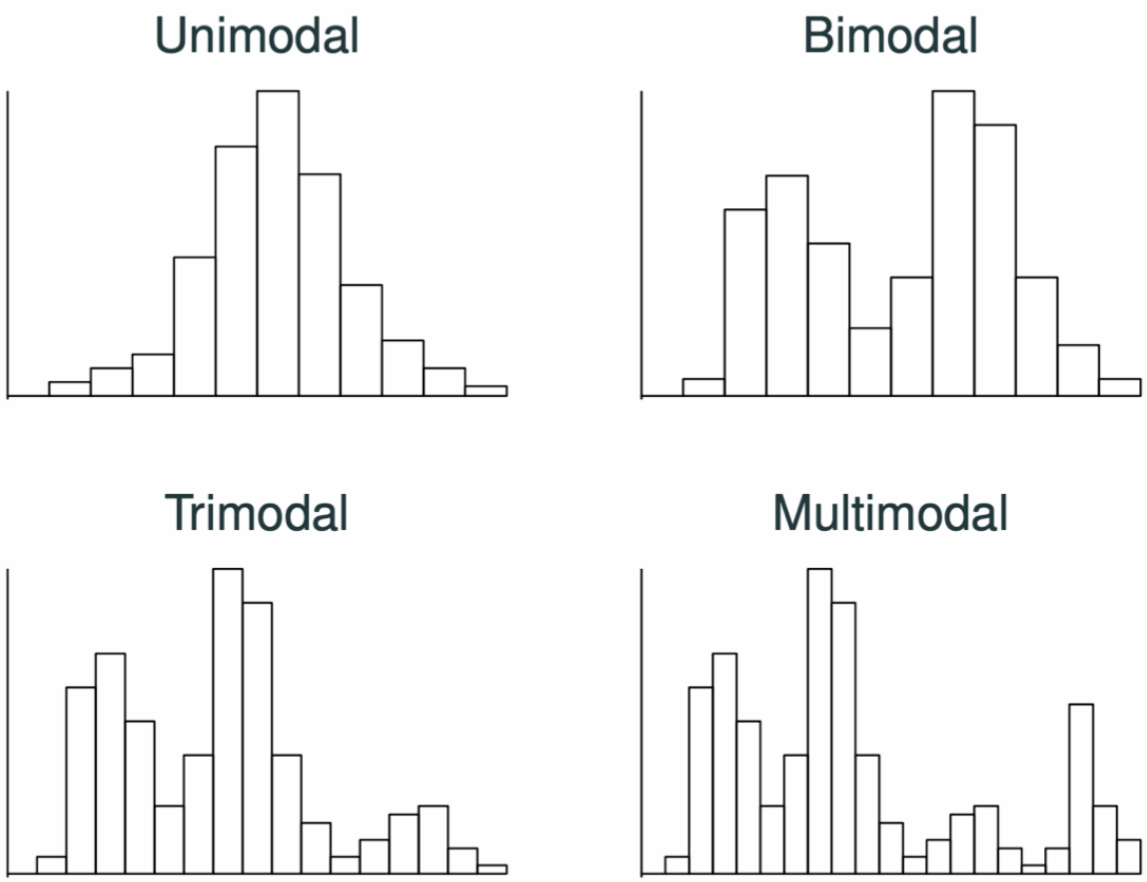
- Not too small that most bins have either 0 or 1 counts
- Not too big that you lose the details in a bin
- There may not be a unique (perfect) bin size
- More the observations, more the bins

Skewness of histograms

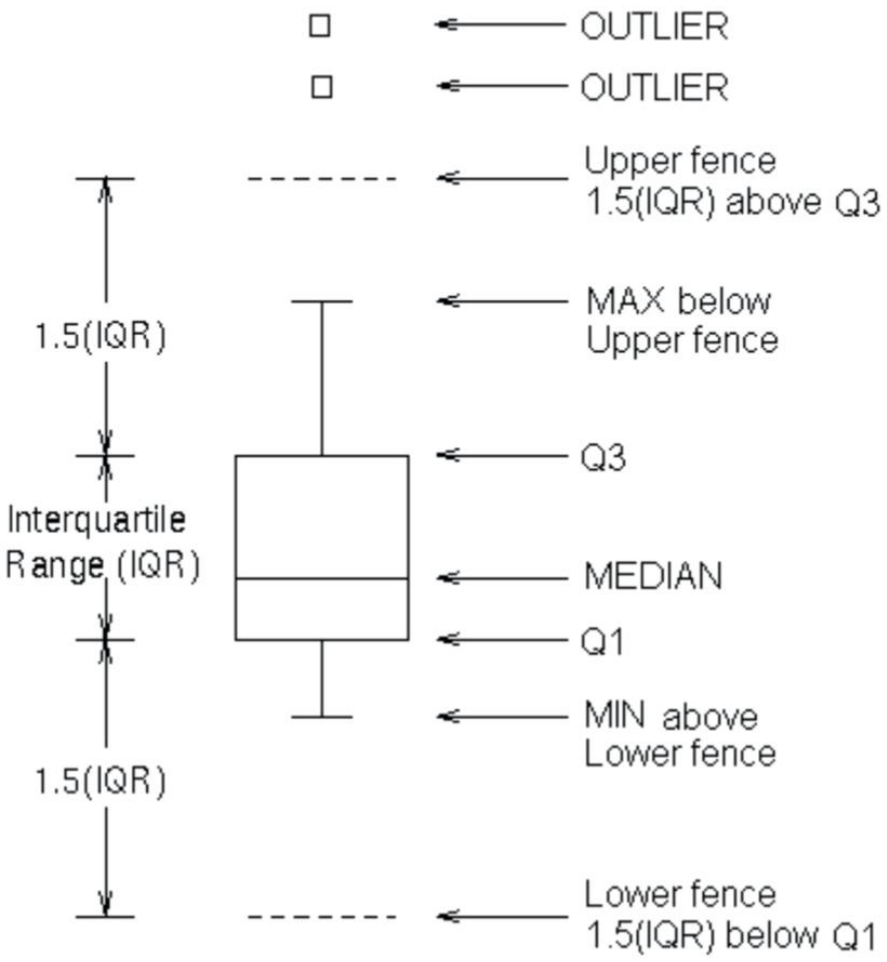


Mode of histograms

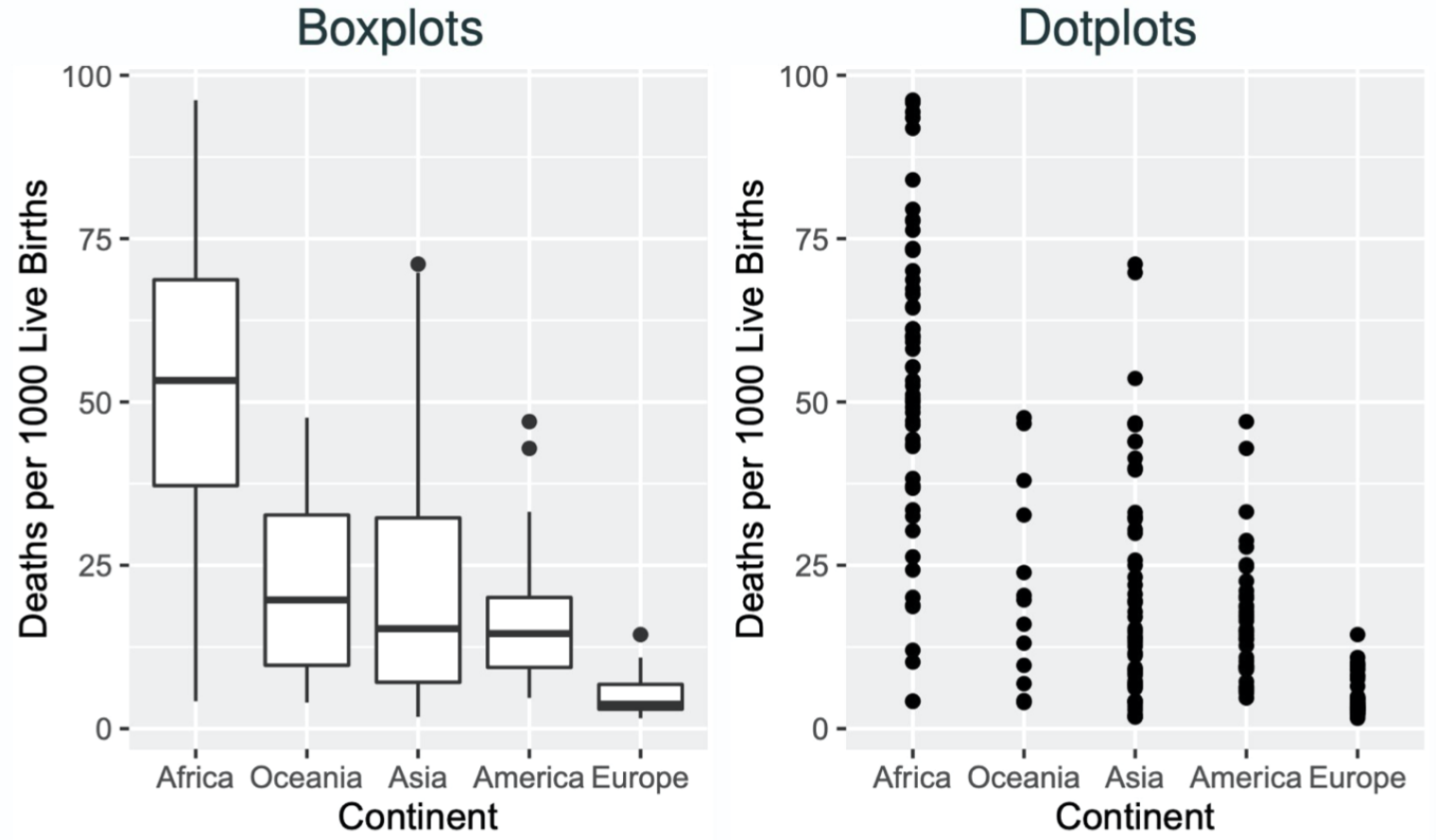
- A histogram with two or more modes may indicate a mixture of two or more distinct populations.



Box-and-whisker plot (Box plot)

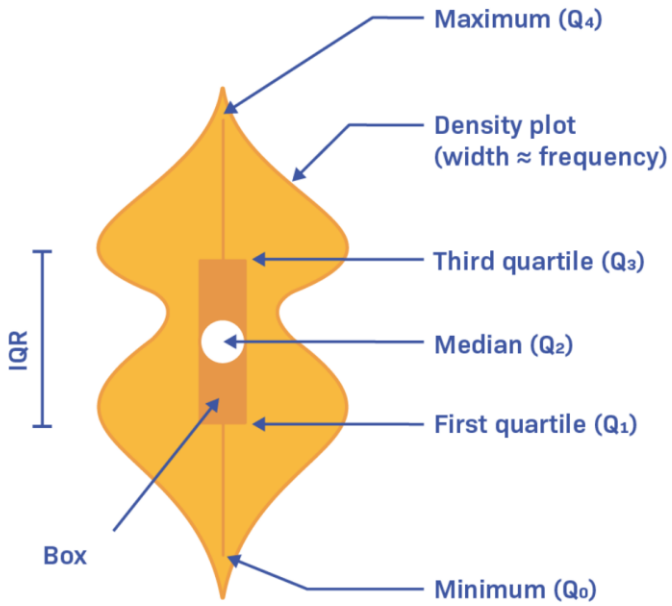


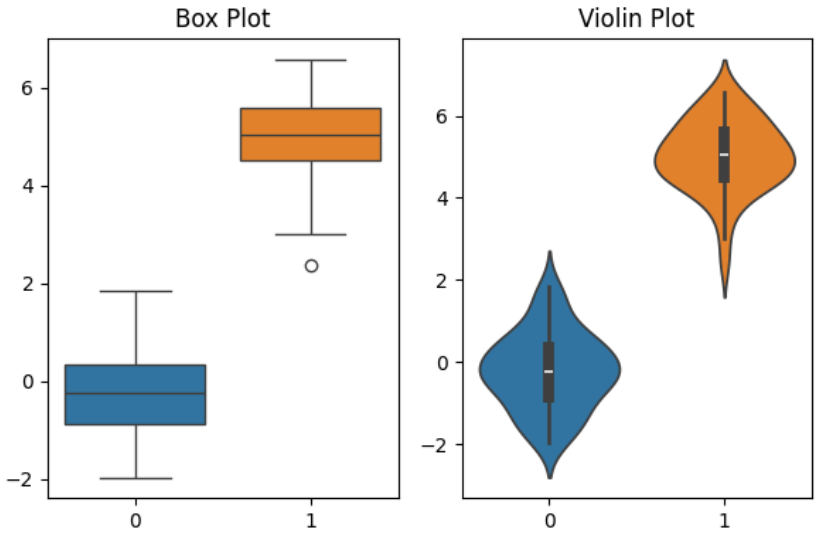
Box plot vs. Dot plot



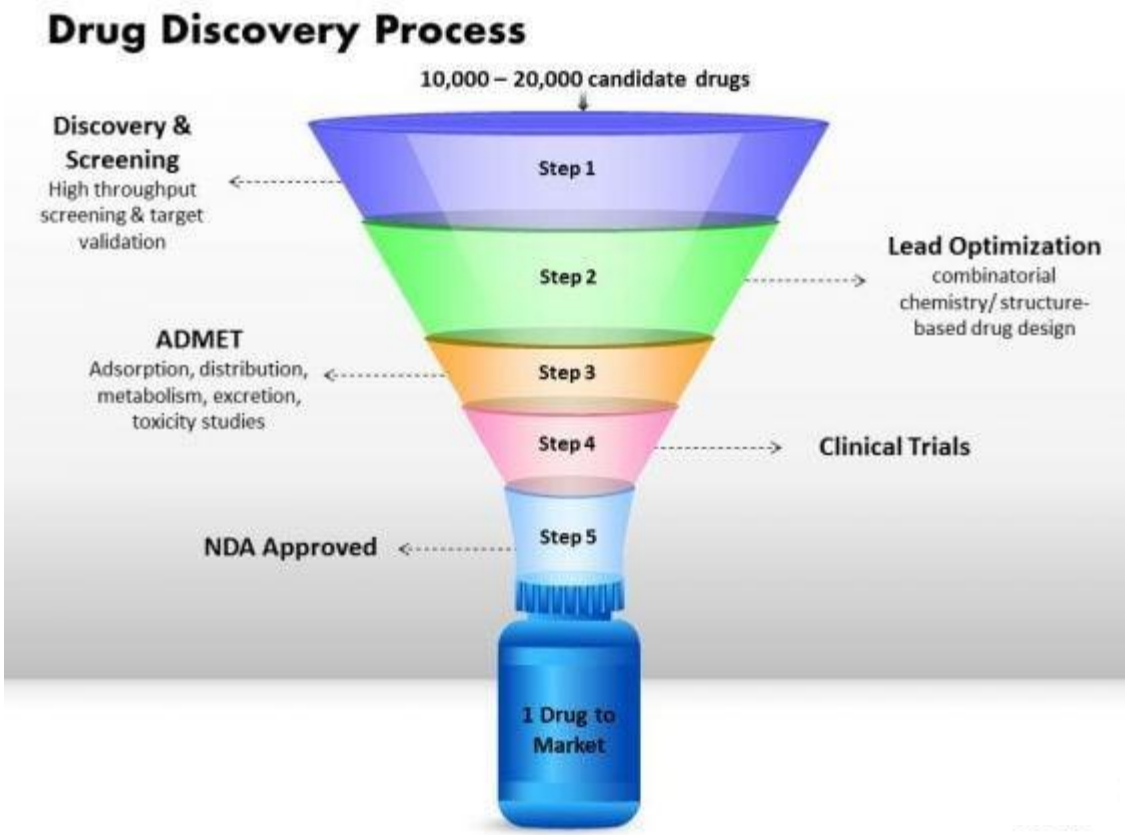
Violin plot

- Violin plots are similar to box plots, except that they also show the probability density of the data at different values, usually smoothed by a kernel density estimator.
- A violin plot will include all the data that is in a box plot: a marker for the median of the data; a box or marker indicating the interquartile range; and possibly all sample points, if the number of samples is not too high.





Funnel chart

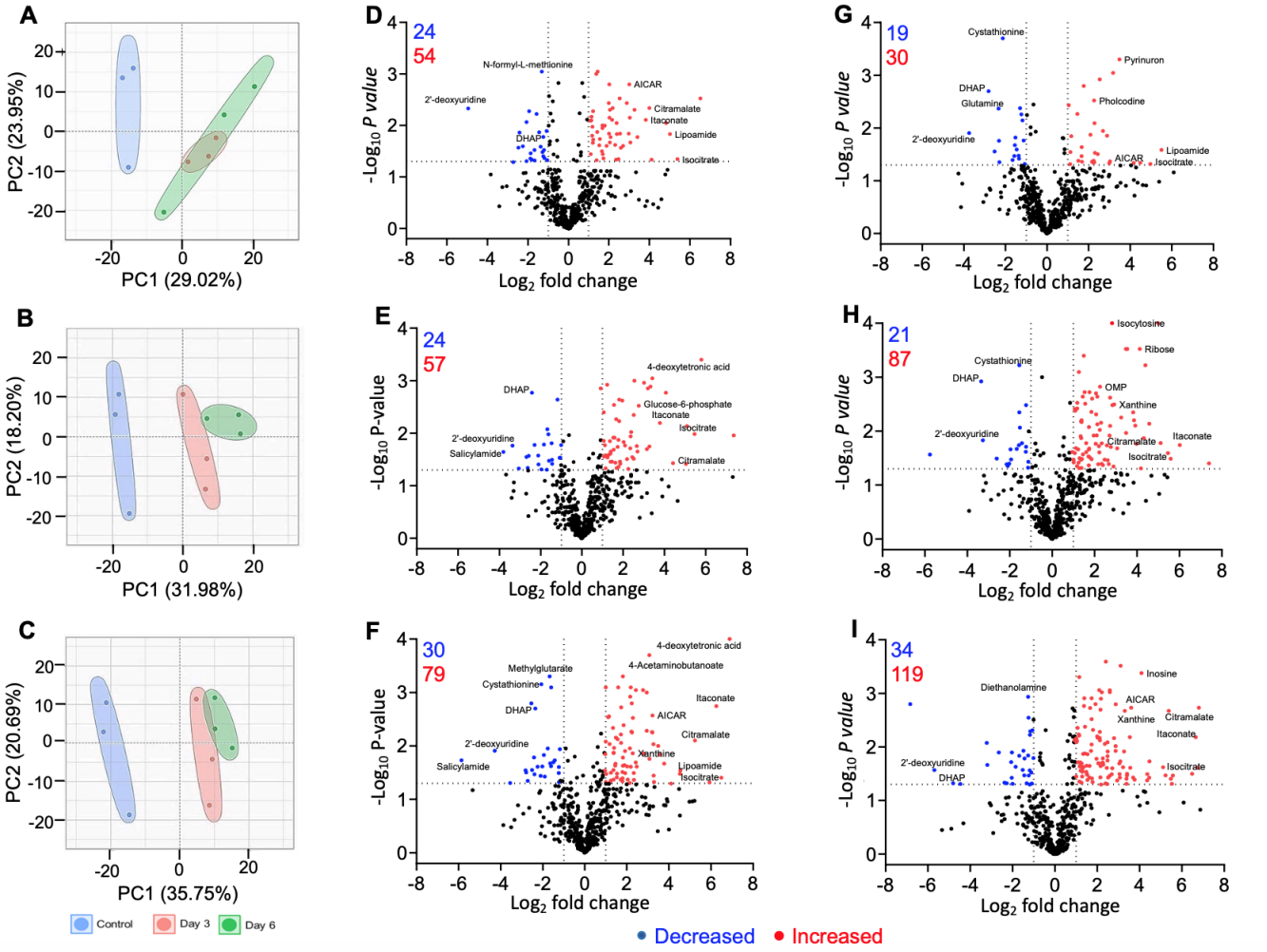


PCA and volcano plots

- Principal component analysis (PCA) is a linear dimensionality reduction technique with applications in exploratory data analysis, visualization and data preprocessing.
- When performing PCA, the first principal component of a set of p variables is the derived variable formed as a linear combination of the original variables that explains the most variance.
- The second principal component explains the most variance in what is left once the effect of the first component is removed, and we may proceed through p iterations until all the variance is explained.
- Volcano plot is a type of scatter-plot that is used to quickly identify changes in large data sets composed of replicate data. It plots significance versus fold-change on the y and x axes, respectively.
- A volcano plot is constructed by plotting the negative logarithm of the p value on the y axis (usually base 10).

PCA and volcano plots

Metabolite profiles of *M.tb* upon exposure to different concentration of itaconate [10 mM (*A*), 20 mM (B), 50 mM (*C*)].



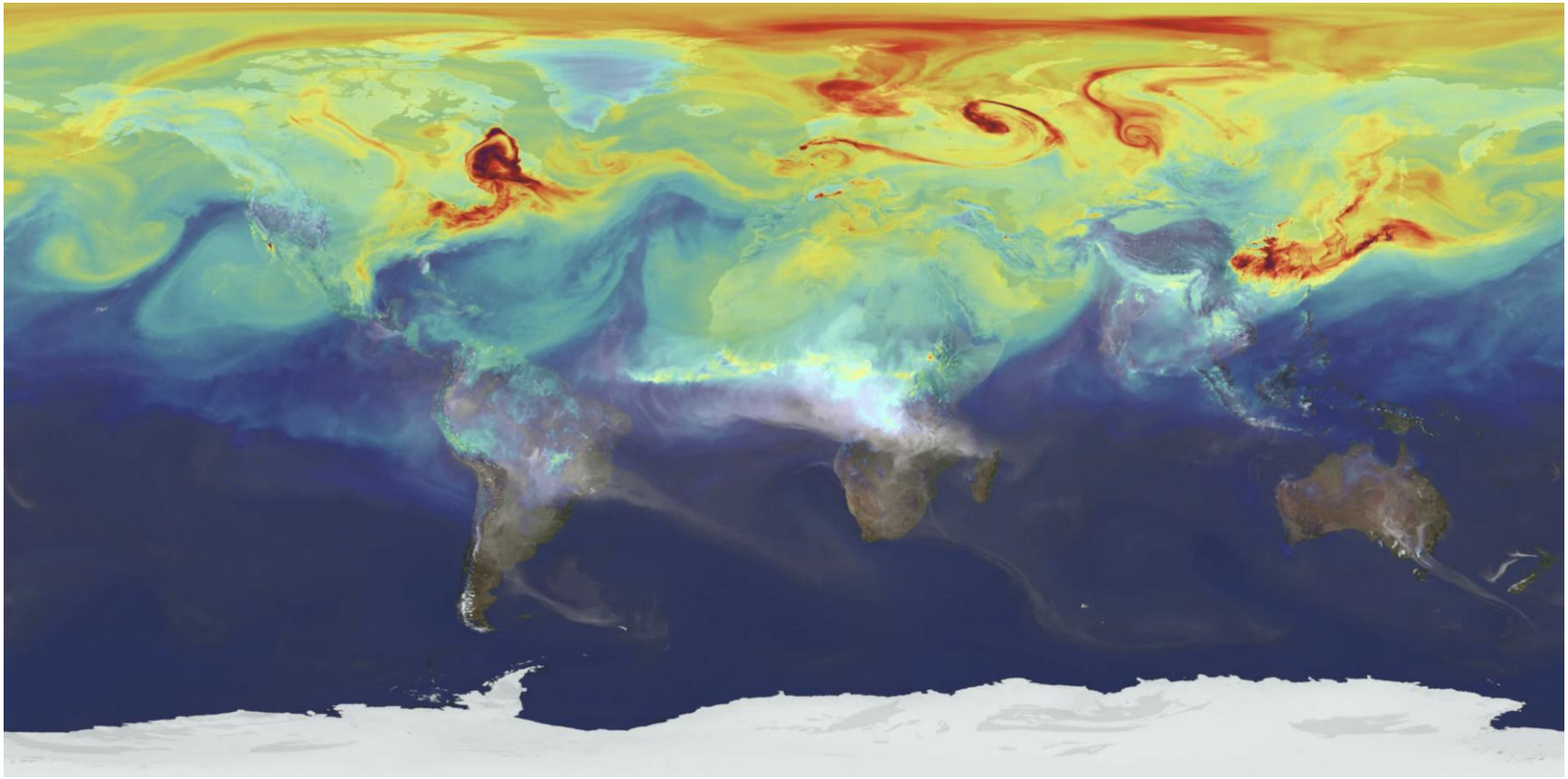
Priya M, Gupta SK, … **Sharma D…,** Singh R (2025) *PNAS*, 122, e2423114122.

Heat maps

- A **heat map** is a 2-dimensional data visualization technique that represents the magnitude of individual values within a dataset as a color.
- There are two main type of heat maps:
  - A **spatial heat map** displays the magnitude of a spatial phenomena as color, usually cast over a map.
  - A **grid heat map** displays magnitude as color in a two-dimensional matrix, with each dimension representing a category of trait and the color representing the magnitude of some measurement on the combined traits from each of the two categories.

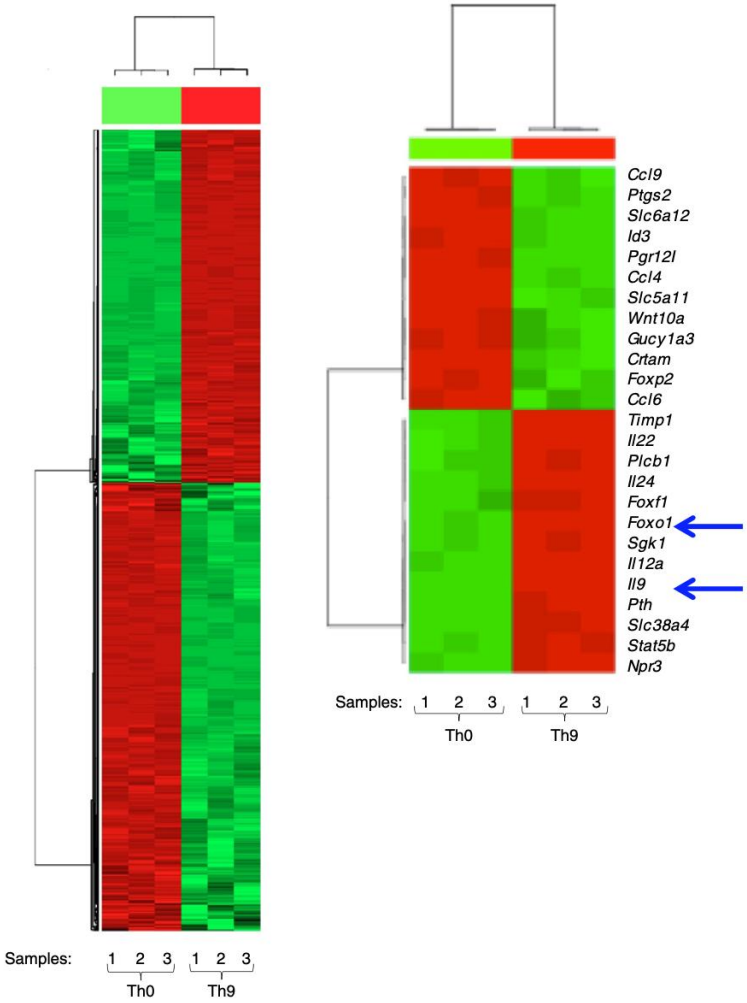
Spatial heat maps

Temperature across world map with red being the highest and blue being the lowest



Grid heat maps

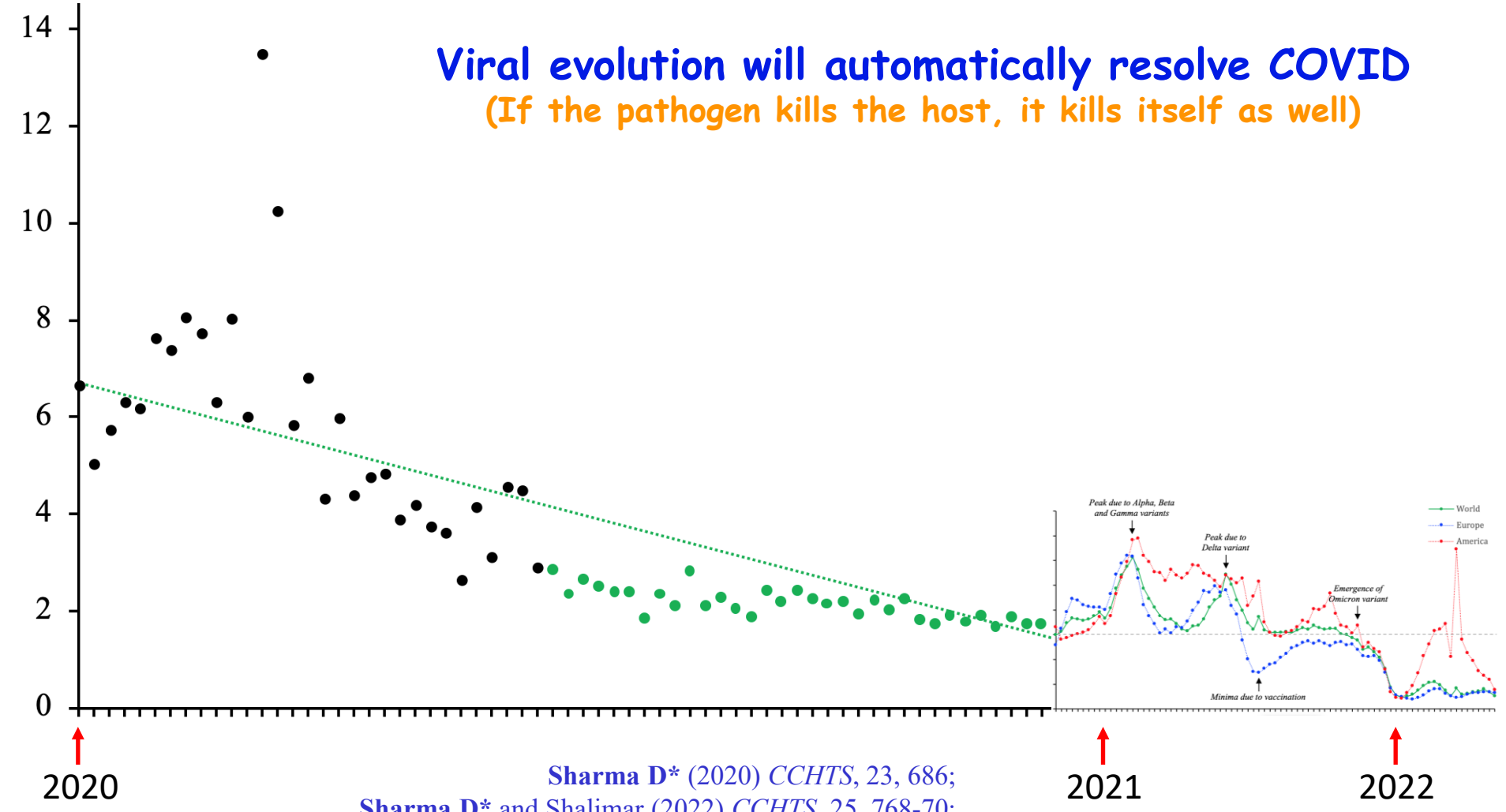
Significantly differentially expressed genes between Th9 and Th0



Malik S, Sadhu S, … **Sharma D…,** Awasthi A (2017) *Nature Commun*, 8, 815.

Real world data analysis: COVID

Lethality

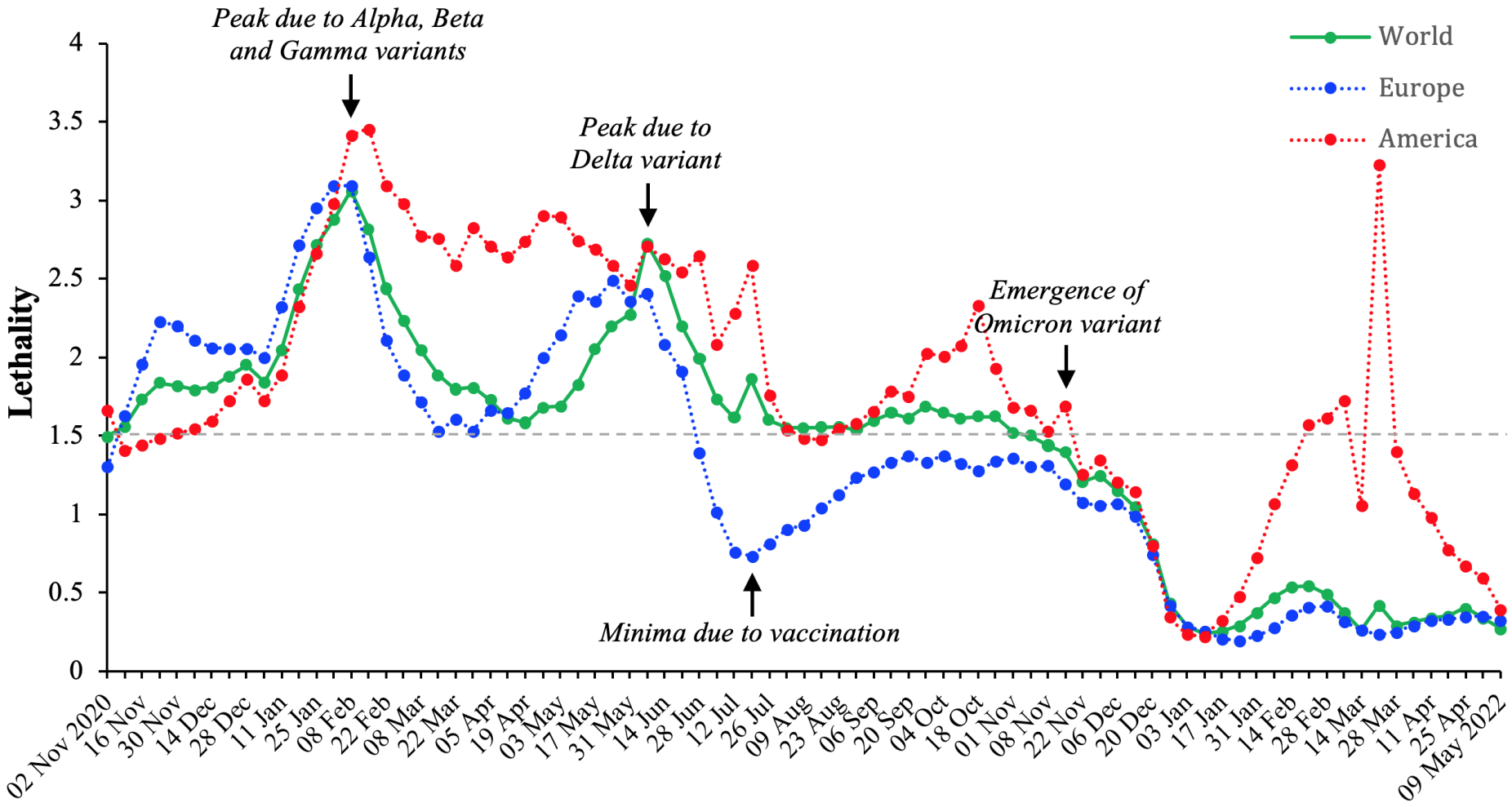


Viral evolution will automatically resolve COVID

(If the pathogen kills the host, it kills itself as well)

Sharma D* (2020) *CCHTS*, 23, 686;
Sharma D* and Shalimar (2022) *CCHTS*, 25, 768-70;
Sharma S, Sood V, **Sharma D*** (2023) *CCHTS*, 26, 2572-74.

COVID vaccines vs. Viral evolution

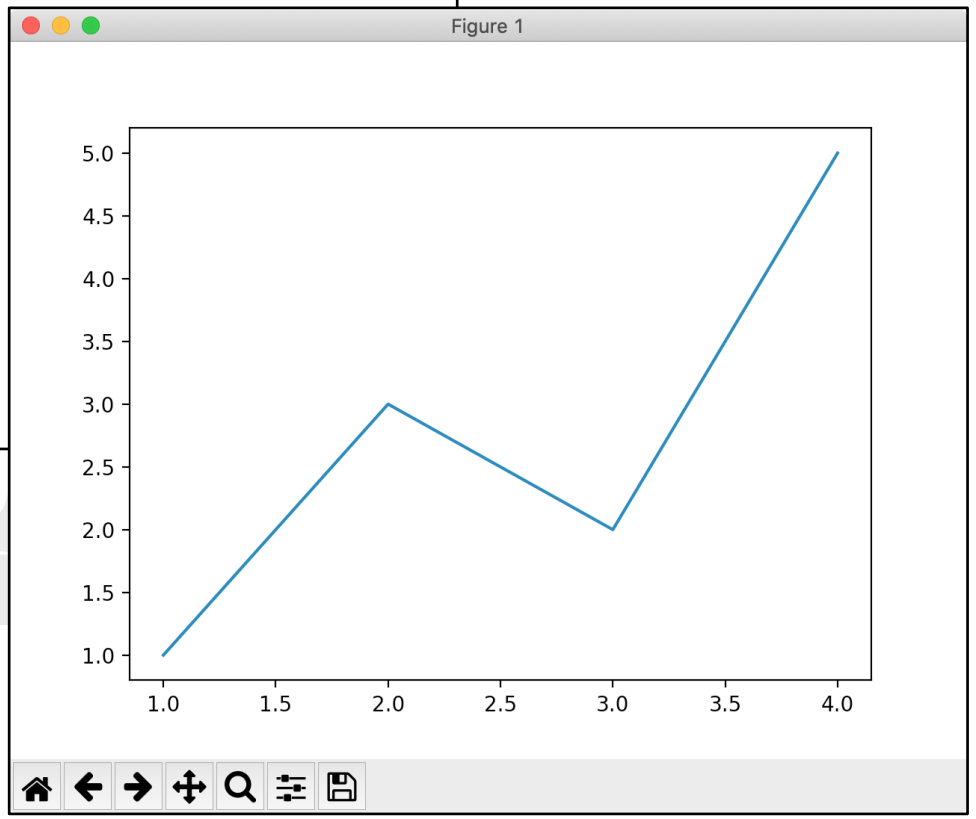


Sharma S, Sood V, **Sharma D*** (2023) *CCHTS*, 26, 2572-74.

Plotting graph

AS2023 — vi pylab_plot.py — 61×15

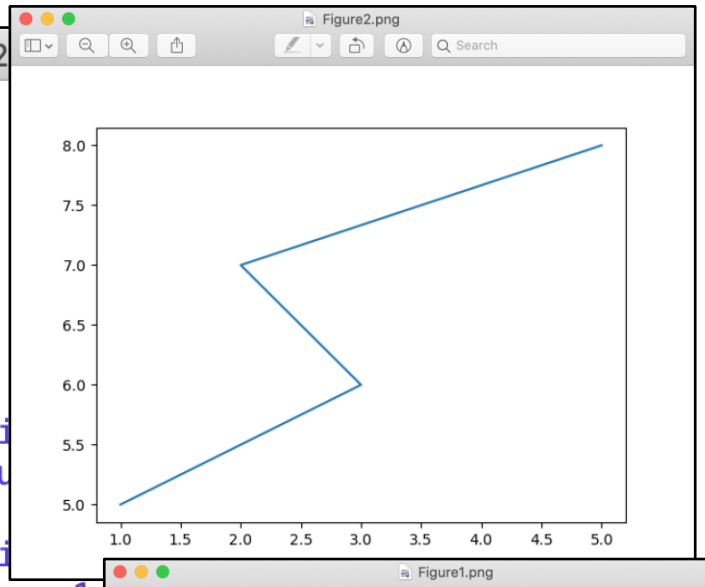
```
#!/Users/deepak/opt/anaconda3/bin/python3/

#####################################
#Plot a graph
#####################################

import pylab

pylab.figure(1)
pylab.plot([1,2,3,4],[1,3,2,5])
pylab.show()
~
~
~
"pylab_plot.py" 11L, 262C
```

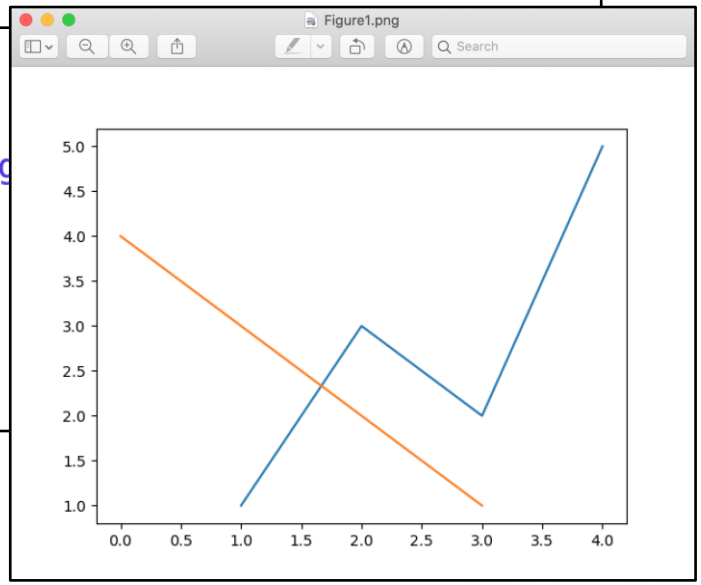


Plotting/Saving graphs

AS2023 — vi pylab_plot2_save.py — 66×2

```
#!/Users/deepak/opt/anaconda3/bin/python3/

####################################
#Plot/Save graphs
####################################

import pylab

pylab.figure(1)                          #create fi
pylab.plot([1,2,3,4],[1,3,2,5])          #draw figu

pylab.figure(2)                          #create fi
pylab.plot([1,3,2,5],[5,6,7,8])          #draw figure 1
pylab.savefig('Figure2')                 #save figure 2

pylab.figure(1)                          #go back to fig
pylab.plot([4,3,2,1])                    #draw again on
pylab.savefig('Figure1')                 #save figure 1
~
~
"pylab_plot2_save.py" 18L, 491C
```

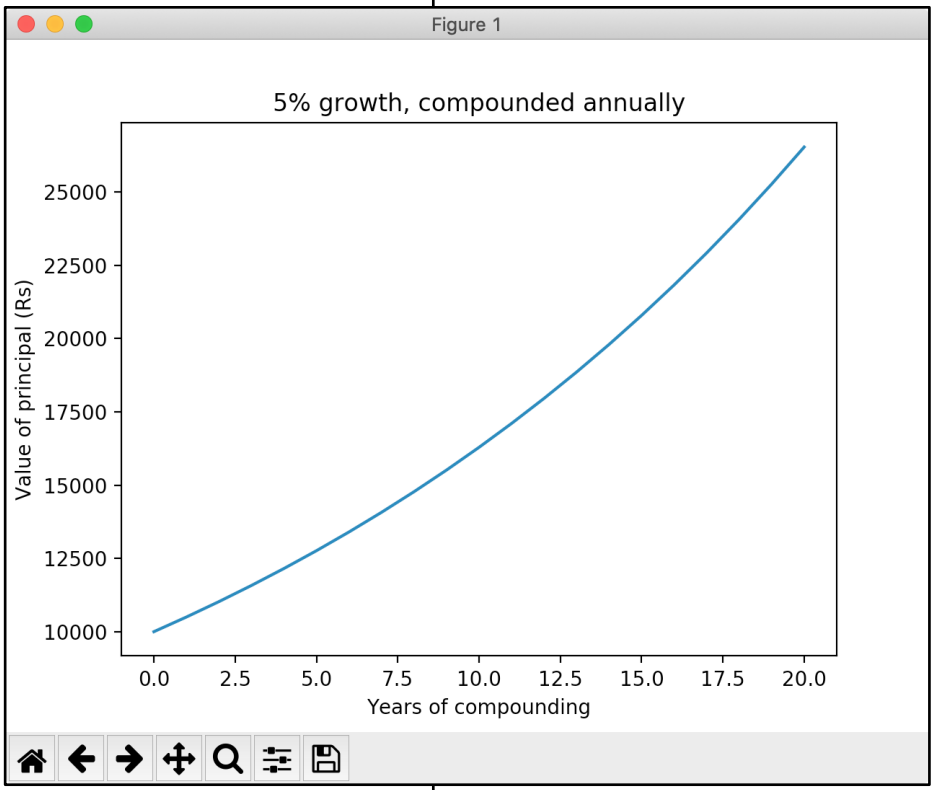




Plot compound interest

AS2023 — vi pylab_plot_compound.py — 69×26

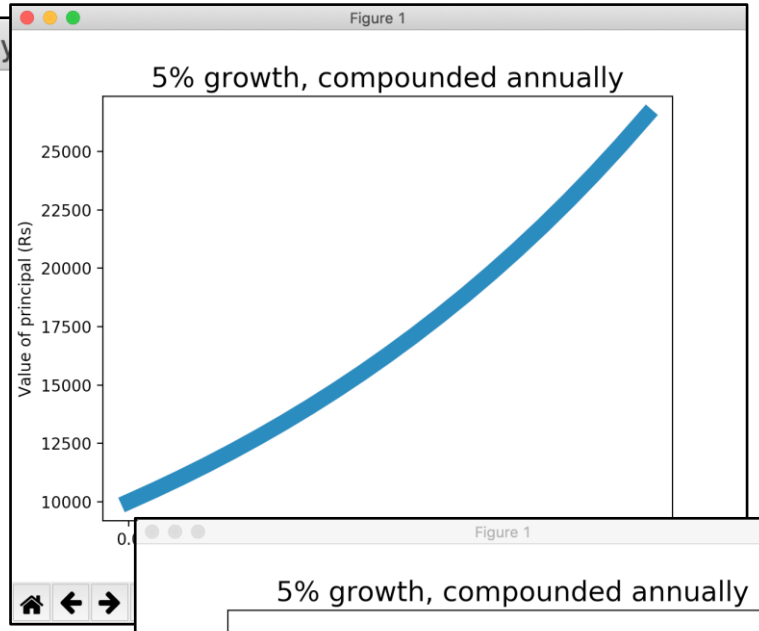
```
#!/Users/deepak/opt/anaconda3/bin/python3/

#####################################
#Plot compound interest
#####################################

import pylab

p = 10000
r = 0.05
years = 20

values=[]
for i in range(years+1):
    values.append(p)
    p = p + p*r

pylab.plot(values)

pylab.title('5% growth, compounded annually')
pylab.xlabel('Years of compounding')
pylab.ylabel('Value of principal (Rs)')
pylab.show()
~
~
"pylab_plot_compound.py" 23L, 417C
```

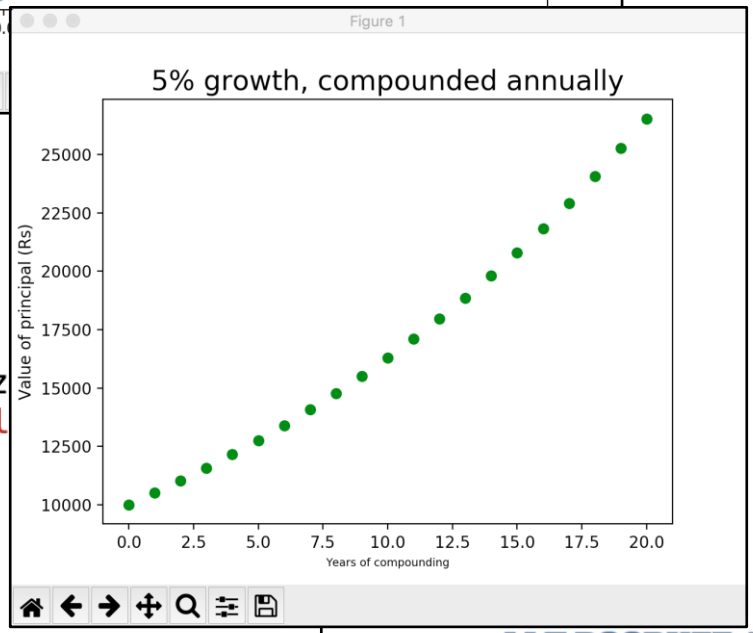


Plot compound interest

AS2023 — vi pylab_plot_compound2.py

```
#!/Users/deepak/opt/anaconda3/bin/python3/

###################################
#Plot compound interest
###################################

import pylab

p = 10000
r = 0.05
years = 20

values=[]
for i in range(years+1):
    values.append(p)
    p = p + p*r

pylab.plot(values, linewidth = 10)
#pylab.plot(values, 'go')

pylab.title('5% growth, compounded annually', fontsiz
pylab.xlabel('Years of compounding', fontsize='x-smal
pylab.ylabel('Value of principal (Rs)')
pylab.show()
~
"pylab_plot_compound2.py" 24L, 500C
```

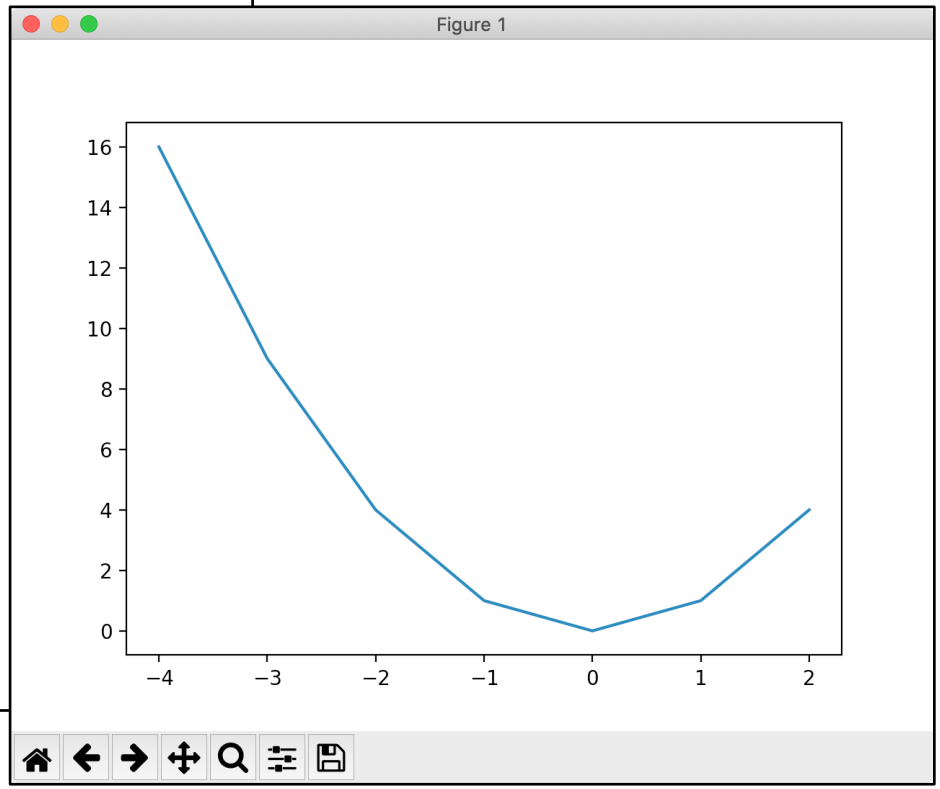




Plotting graph (take 2)

AS2023 — vi pylab_plot_range.py — 53×23

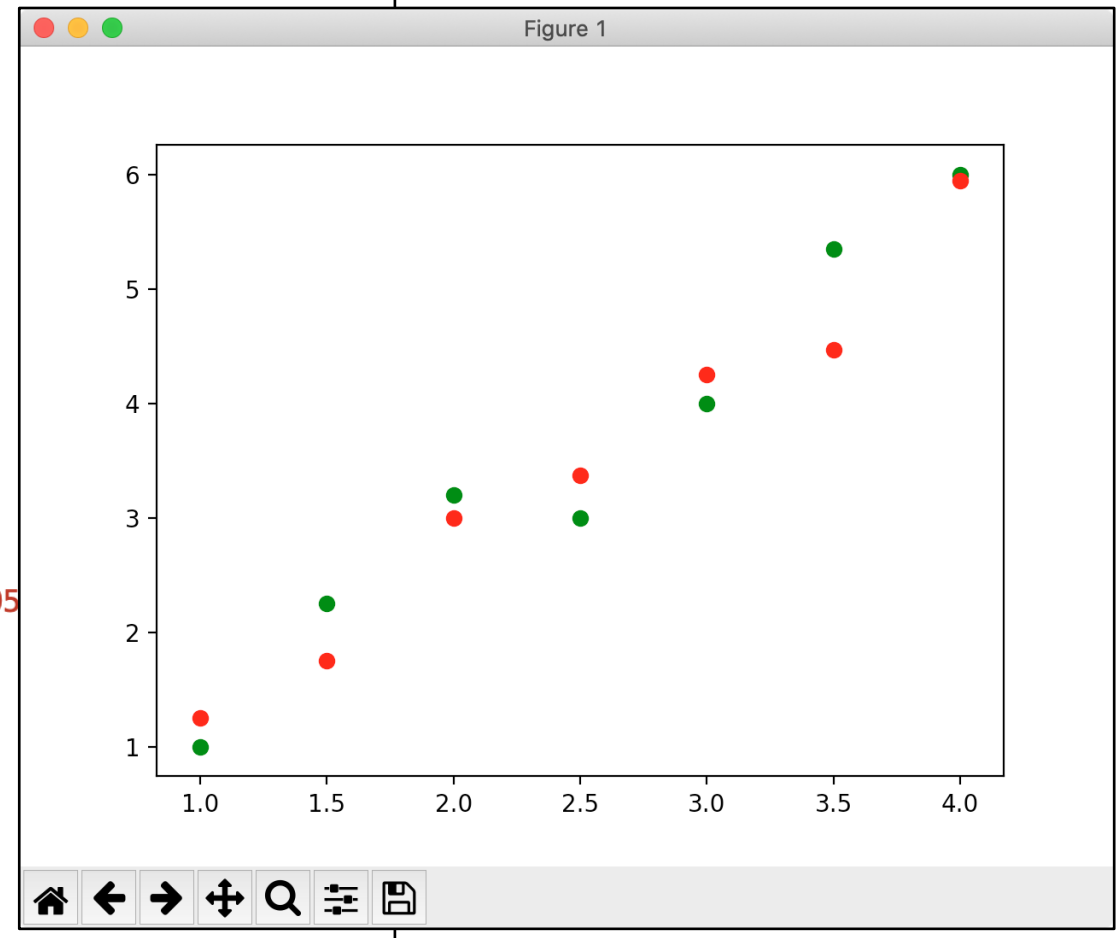
```
#!/Users/deepak/opt/anaconda3/bin/python3/

#####################################
#Plot a graph
#####################################

# Importing from Numpy
from numpy import *
# Importing from PyLab
from pylab import *

# X-axis of the curve
a = linspace(-4, 2, 7)
# Y-axis of the curve
b = a**2

# Plotting the curve with x and y-axis
plot(a, b)
# Showing curve in the output
show()
~
~
"pylab_plot_range.py" 20L, 406C
```



Scatter plot

Python — vi scatter_plot.py — 63×28

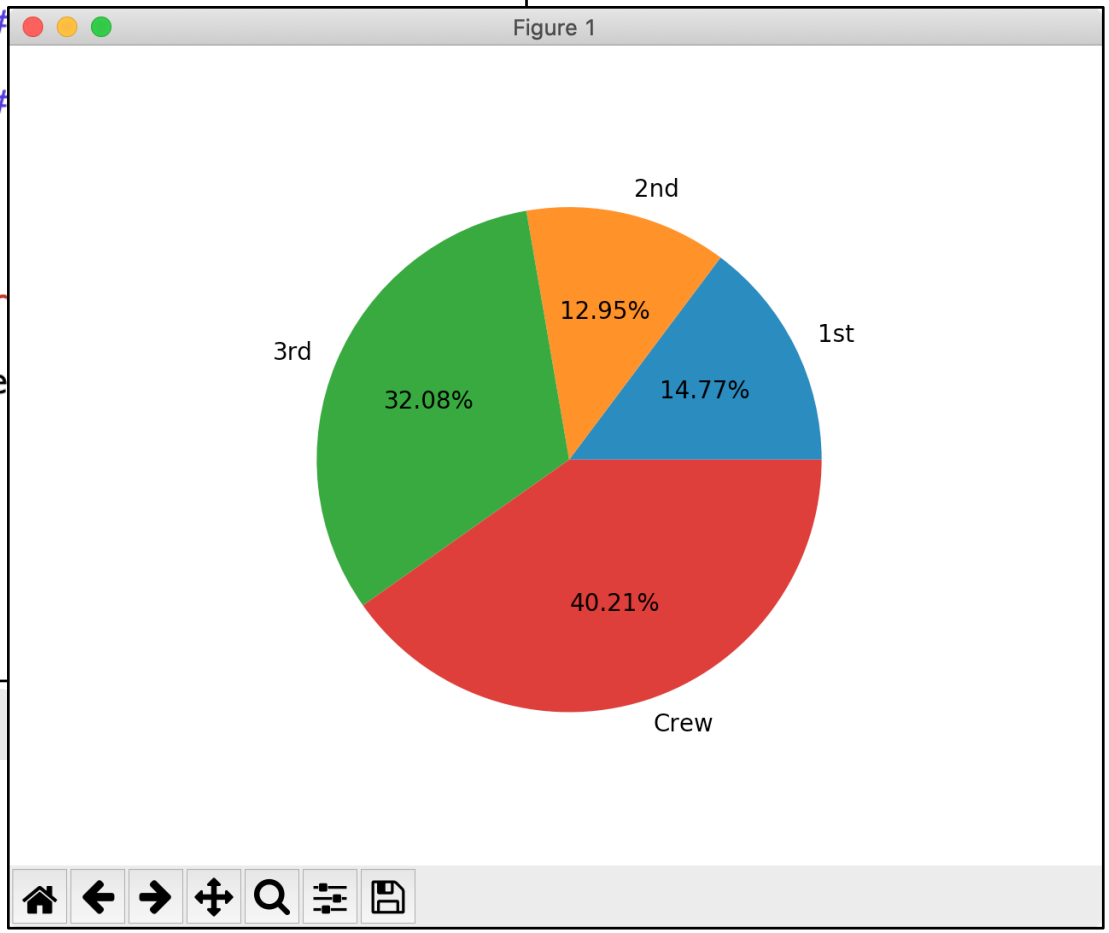
```
#!/Users/deepak/opt/anaconda3/bin/python3/

#####################################
#Scatter plot
#####################################

import pylab as plt
import numpy as np

# first data point
x = [1, 1.5, 2, 2.5, 3, 3.5, 4]
y = [1, 2.25, 3.2, 3, 4, 5.35, 6]

# depict first scatter plot
plt.scatter(x, y, c='green')

# second data point
x = [1, 1.5, 2, 2.5, 3, 3.5, 4]
y = [1.25, 1.75, 3, 3.37, 4.25, 4.47, 5.95

# depict second scatter plot
plt.scatter(x, y, c='red')

# depict illustration
plt.show()

~
"scatter_plot.py" 26L, 507C
```



Pie chart

Python — vi pie_chart.py — 63×18

```
#!/Users/deepak/opt/anaconda3/bin/python3/

####################################
#Pie chart
####################################

import pylab as plt
import numpy as np

mylabels = ["1st", "2nd", "3rd", "Cr
Data = [325,285,706,885]
plt.pie(Data,autopct='%1.2f%%', labe
plt.show()
~
~
~
~
"pie_chart.py" 13L, 300C
```



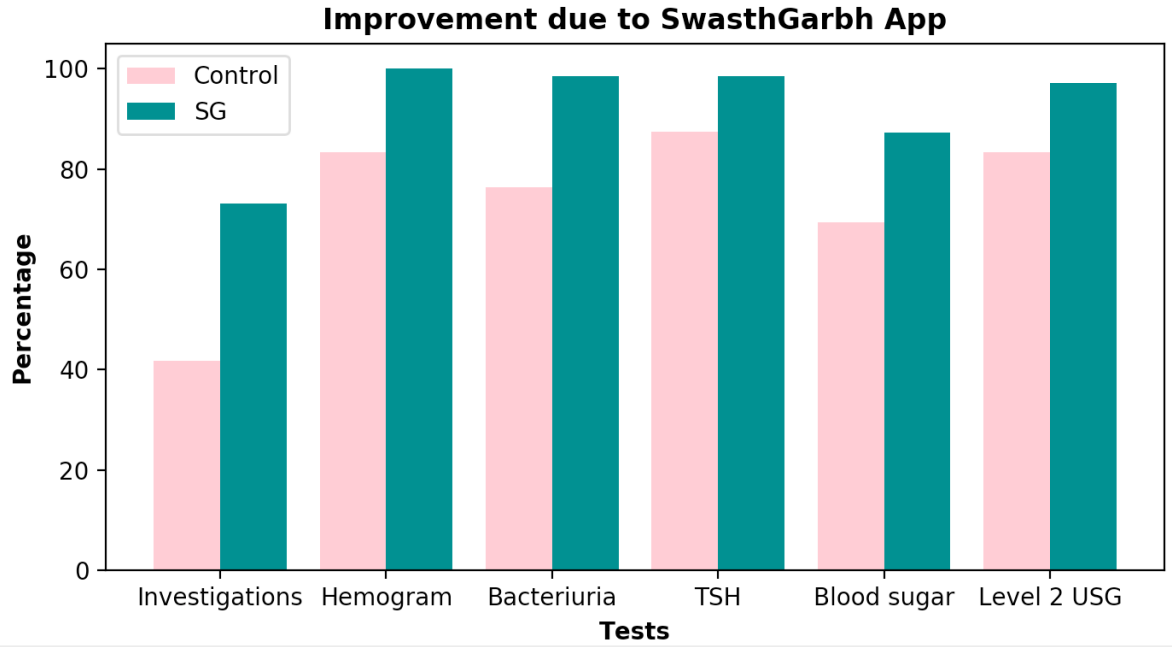
Bar plot

Python — vi bar_plot.py — 88×27

```
#!/Users/deepak/opt/anaconda3/bin/python3/

#####################################
#Bar plot
#####################
```

```
import pylab as plt
import numpy as np

Percent_C = [41.7,83.3
Percent_SG = [73.2,100
Tests = ['Investigatio

X_axis = np.arange(ler
plt.figure(figsize=(8,
plt.bar(X_axis - 0.2,
plt.bar(X_axis + 0.2,

plt.xticks(X_axis,Test
plt.title('Improvement
plt.xlabel('Tests', we
plt.ylabel('Percentage
plt.legend()
plt.show()
~
~
"bar_plot.py" 24L, 721C
```

Figure 1



Data Analysis
(Univariate, Bivariate)

Deepak Sharma
BSBE and DS&AI
IIT Roorkee

Univariate analysis

Univariate analysis focuses on analyzing a single variable at a time. It summarizes and finds patterns in the data, typically using statistical measures and visualizations.

Key Goals

- To understand the distribution, central tendency (central/typical value) and variability of the data.
- Identify outliers or anomalies.

Univariate analysis

- Variables could be either categorical or numerical.
- There are different statistical and visualization techniques of investigation for each type of variable.
- Numerical variables can be transformed into categorical counterparts by a process called binning or discretization.
- It is also possible to transform a categorical variable into its numerical counterpart by a process called encoding.

Univariate analysis

Visualization

- Barplots, Pie charts: For categorical data.
- Histograms: For distribution (numerical variables).
- Box Plots: For identifying outliers.

Descriptive Statistics

- Measures of Central Tendency: Mean, median, mode.
- Measures of Dispersion: Variance, SD, range, interquartile range.

Bivariate analysis

Bivariate analysis examines the relationship between two variables (attributes). It explores associations, dependencies, or correlations.

Key Goals

- To determine the nature and strength of the relationship between the two variables.
- Analyze patterns that may indicate causal or correlative relationships.

Bivariate analysis

Visualization

- Scatterplots: Show relationships between continuous variables.
- Segmented/Stacked/Grouped Bar Charts: For categorical relationships.
- Line plots: Show trends.

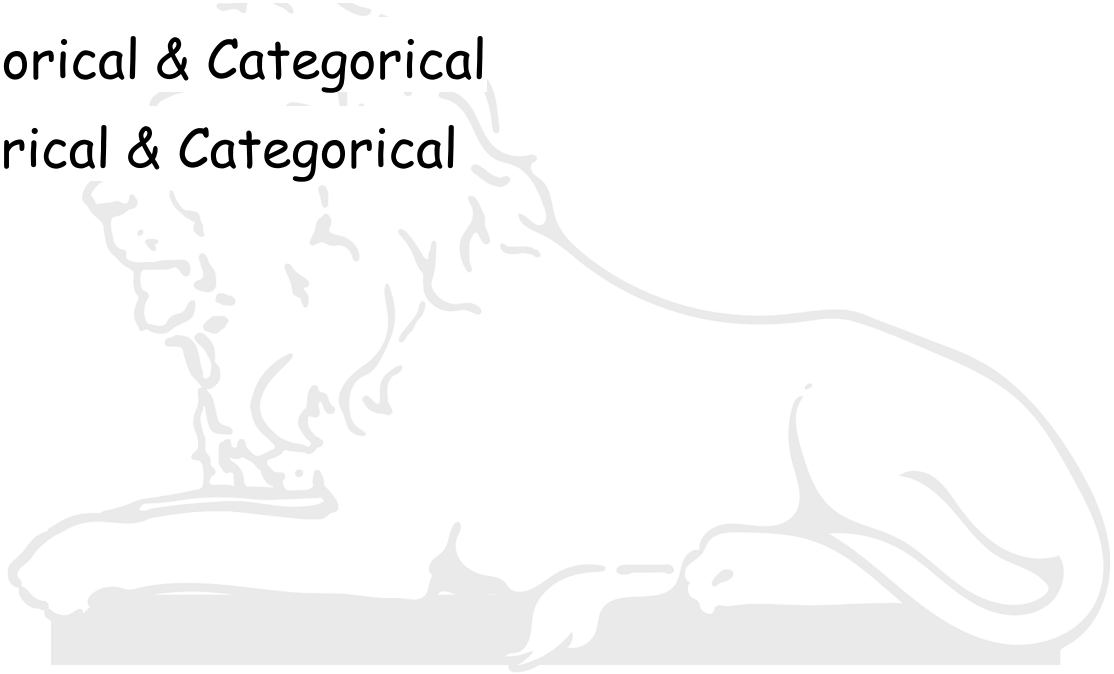
Statistical Tests

- Chi-square Test: For association between categorical variables.
- t-Test or ANOVA: For comparing means across groups.

Bivariate analysis

There are three types of bivariate analysis:

1. Numerical & Numerical
2. Categorical & Categorical
3. Numerical & Categorical

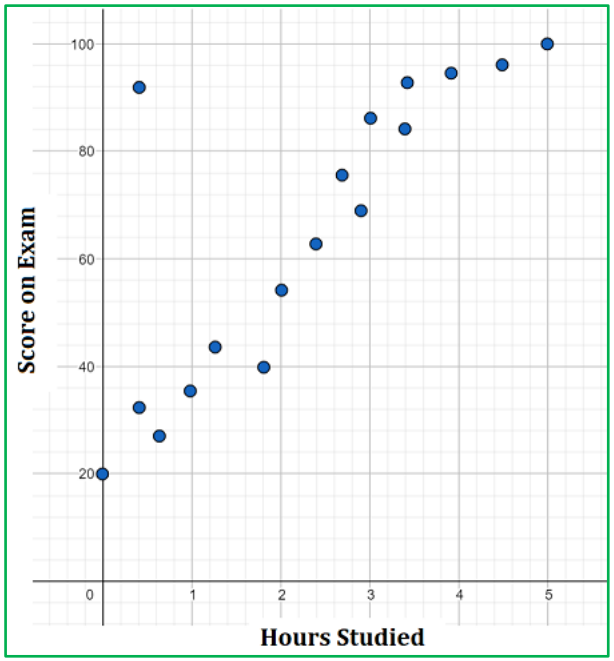


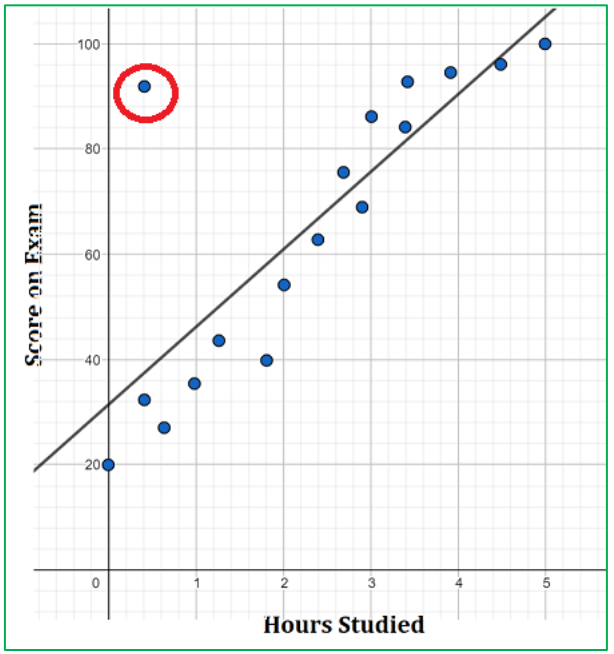
Numerical & Numerical

- A scatter plot is a useful visual representation of the relationship between two numerical variables (attributes).
- It is usually drawn before working out a linear correlation or fitting a regression line.
- The resulting pattern indicates the type (linear or non-linear) and strength of the relationship between two variables.
- Outliers can skew your analysis and conclusions. Scatterplots make it easy to spot outliers, helping you make informed decisions about whether to include or exclude them from your analysis.
- Scatterplots also offer a means to gain an understanding of your data's distribution. They enable you to assess whether data points are closely concentrated around a central line or if there is a notable dispersion, visually conveying valuable insights into the distribution of your numerical variables.

Scatter plot

- We can also draw a "Line of Best Fit" (also called "Trend Line"):
  - The line should be as close as possible to all points, and as many points above the line as below.





Regression analysis

- Models the relationship between a dependent variable (Y) and one or more independent variables (X).

  - Simple linear regression equation:

$$Y = \beta_0 + \beta_1 X + \epsilon \qquad (y = mx + c)$$

β_0: Intercept

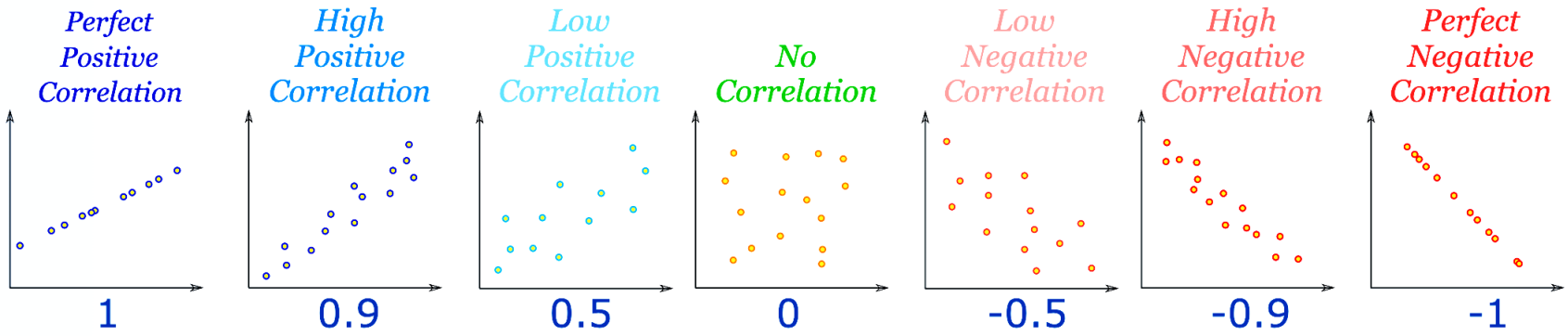
β_1: Slope (rate of change in Y per unit change in X)

ϵ: Error term

Correlation

- When the two sets of data are strongly linked together, they have a high correlation.
  - It is positive when the values increase together
  - It is negative when one value decreases as the other increases

- Correlation can have a value:
  - 1 is a perfect positive correlation
  - 0 is no correlation (the values don't seem linked at all)
  - -1 is a perfect negative correlation



Correlation

- Correlation Coefficient (r): Measures the strength and direction of a linear relationship between two variables.

$$r = \frac{Covar(x,y)}{\sqrt{Var(x)Var(y)}}$$

$$Var(x) = \frac{\sum(x-\bar{x})^2}{n}$$

$$Var(y) = \frac{\sum(y-\bar{y})^2}{n}$$

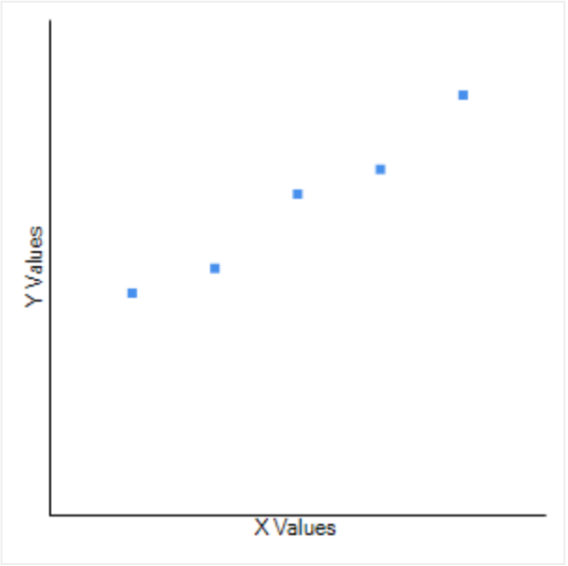
  - r ranges from -1 to 1:
    - r = 1: Perfect positive linear correlation.
    - r = –1: Perfect negative linear correlation.
    - r = 0: No linear correlation.

- Covariance: Measures how two variables vary together.

$$Covar(x,y) = \frac{\sum(x-\bar{x})(y-\bar{y})}{n}$$

Correlation Coefficient

Student	Hours studied (X)	Test score (Y)
1	2	45
2	4	50
3	6	65
4	8	70
5	10	85

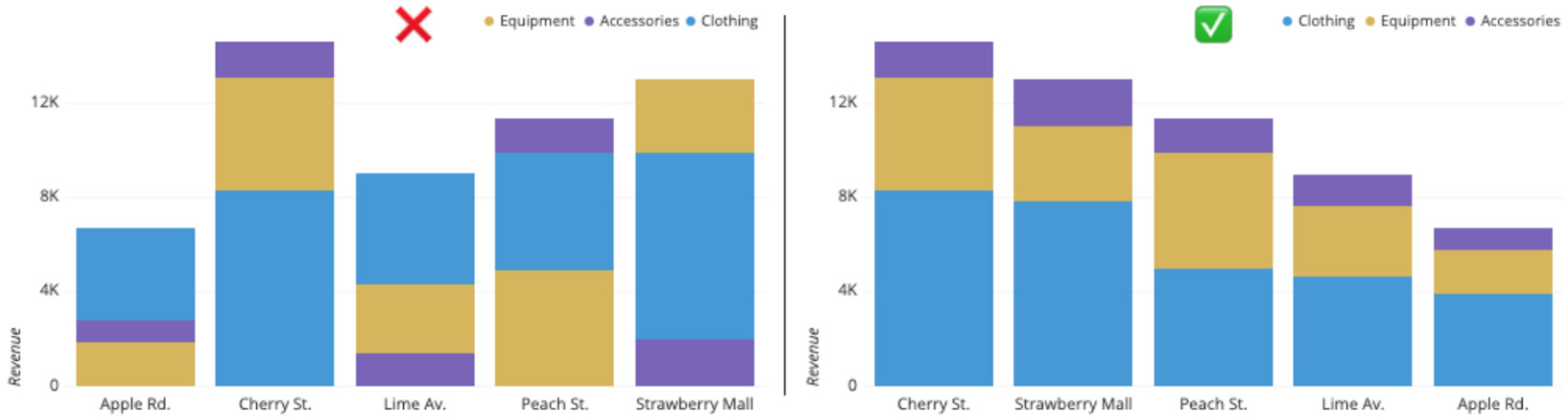


$X - M_x$	$Y - M_y$	$(X - M_x)^2$	$(Y - M_y)^2$	$(X - M_x)(Y - M_y)$
−4.000	−18.000	16.000	324.000	72.000
−2.000	−13.000	4.000	169.000	26.000
0.000	2.000	0.000	4.000	0.000
2.000	7.000	4.000	49.000	14.000
4.000	22.000	16.000	484.000	88.000
Mx: 6.000	My: 63.000	Sum: 40.000	Sum: 1030.000	Sum: 200.000

Covariance: 40; Variance (X) = 8; Variance (Y) = 206; Correlation: 0.986

Categorical & Categorical

- The stacked bar chart extends the standard bar chart from looking at numeric values across one categorical variable to two.
- Each bar in a standard bar chart is divided into a number of sub-bars stacked end to end, each one corresponding to various categories of the second categorical variable.



Chi-square test

$$\chi^2 = \sum_{i=1}^{r}\sum_{j=1}^{c}\frac{(n_{ij} - e_{ij})^2}{e_{ij}}$$

$$e_{ij} = \frac{n_{i.}n_{.j}}{n}$$

$$df = (r-1)(c-1)$$

- The chi-square test can be used to determine the association between categorical variables.
- It is based on the difference between the expected frequencies (*e*) and the observed frequencies (*n*) in one or more categories in the frequency table.

Chi-square test

Risk of breast cancer in women increases with increasing age at 1st childbirth

Status	Age at first birth ≥30	Age at first birth ≤29	Total
Case	683	2537	3220
Control	1498	8747	10,245
Total	2181	11,284	13,465

WHO Bulletin, *43*, 209–221, 1970.

Computation of Expected Values for 2 × 2 Contingency Tables

The **expected number of units** in the **(*i*, *j*) cell**, which is usually denoted by E_{ij}, is the product of the ***i*th** row margin multiplied by the ***j*th** column margin, divided by the grand total.

Expected table

Case–control status	Age at first birth ≥30	Age at first birth ≤29	Total
Case	521.6	2698.4	3220
Control	1659.4	8585.6	10,245
Total	2181	11,284	13,465

E_{11} = expected number of units in the (1, 1) cell
= 3220(2181)/13,465 = 521.6

E_{12} = expected number of units in the (1, 2) cell
= 3220(11,284)/13,465 = 2698.4

E_{21} = expected number of units in the (2, 1) cell
= 10,245(2181)/13,465 = 1659.4

E_{22} = expected number of units in the (2, 2) cell
= 10,245(11,284)/13,465 = 8585.6

$X^2 = (683-521.6)^2/521.6 + (2537-2698.4)^2/2698.4 + (1498-1659.4)^2/1659.4 + (8747 - 8585.6)^2/8585.6$

$= 161.4^2/521.6 + 161.4^2/2698.4 + 161.4^2/1659.4 + 161.4^2/8585.6$

$= 49.942 + 9.653 + 15.698 + 3.034 = 78.327$

$\chi^2_{1,.999} = 10.83 < 78.327 = X^2$, $p < 1 - 0.999 = 0.001$

For a level α test, reject H_0 if $X^2 > \chi^2_{1,1-\alpha}$ and accept H_0 if $X^2 \le \chi^2_{1,1-\alpha}$

The results are extremely significant. Thus, breast cancer incidence is significantly associated with having a first child after the age 30.

Chi-square test

Table 6 Percentage points of the chi-square distribution ($\chi^2_{d,u}$)[a]

	u													
d	.005	.01	.025	.05	.10	.25	.50	.75	.90	.95	.975	.99	.995	.999
1	0.0^4393[b]	0.0^3157[c]	0.0^3982[d]	0.00393	0.02	0.10	0.45	1.32	2.71	3.84	5.02	6.63	7.88	10.83
2	0.0100	0.0201	0.0506	0.103	0.21	0.58	1.39	2.77	4.61	5.99	7.38	9.21	10.60	13.81
3	0.0717	0.115	0.216	0.352	0.58	1.21	2.37	4.11	6.25	7.81	9.35	11.34	12.84	16.27
4	0.207	0.297	0.484	0.711	1.06	1.92	3.36	5.39	7.78	9.49	11.14	13.28	14.86	18.47
5	0.412	0.554	0.831	1.15	1.61	2.67	4.35	6.63	9.24	11.07	12.83	15.09	16.75	20.52
6	0.676	0.872	1.24	1.64	2.20	3.45	5.35	7.84	10.64	12.59	14.45	16.81	18.55	22.46
7	0.989	1.24	1.69	2.17	2.83	4.25	6.35	9.04	12.02	14.07	16.01	18.48	20.28	24.32
8	1.34	1.65	2.18	2.73	3.49	5.07	7.34	10.22	13.36	15.51	17.53	20.09	21.95	26.12
9	1.73	2.09	2.70	3.33	4.17	5.90	8.34	11.39	14.68	16.92	19.02	21.67	23.59	27.88
10	2.16	2.56	3.25	3.94	4.87	6.74	9.34	12.55	15.99	18.31	20.48	23.21	25.19	29.59
11	2.60	3.05	3.82	4.57	5.58	7.58	10.34	13.70	17.28	19.68	21.92	24.72	26.76	31.26
12	3.07	3.57	4.40	5.23	6.30	8.44	11.34	14.85	18.55	21.03	23.34	26.22	28.30	32.91
13	3.57	4.11	5.01	5.89	7.04	9.30	12.34	15.98	19.81	22.36	24.74	27.69	29.82	34.53
14	4.07	4.66	5.63	6.57	7.79	10.17	13.34	17.12	21.06	23.68	26.12	29.14	31.32	36.12
15	4.60	5.23	6.27	7.26	8.55	11.04	14.34	18.25	22.31	25.00	27.49	30.58	32.80	37.70
16	5.14	5.81	6.91	7.96	9.31	11.91	15.34	19.37	23.54	26.30	28.85	32.00	34.27	39.25
17	5.70	6.41	7.56	8.67	10.09	12.79	16.34	20.49	24.77	27.59	30.19	33.41	35.72	40.79
18	6.26	7.01	8.23	9.39	10.86	13.68	17.34	21.60	25.99	28.87	31.53	34.81	37.16	42.31
19	6.84	7.63	8.91	10.12	11.65	14.56	18.34	22.72	27.20	30.14	32.85	36.19	38.58	43.82
20	7.43	8.26	9.59	10.85	12.44	15.45	19.34	23.83	28.41	31.41	34.17	37.57	40.00	45.32
21	8.03	8.90	10.28	11.59	13.24	16.34	20.34	24.93	29.62	32.67	35.48	38.93	41.40	46.80
22	8.64	9.54	10.98	12.34	14.04	17.24	21.34	26.04	30.81	33.92	36.78	40.29	42.80	48.27
23	9.26	10.20	11.69	13.09	14.85	18.14	22.34	27.14	32.01	35.17	38.08	41.64	44.18	49.73
24	9.89	10.86	12.40	13.85	15.66	19.04	23.34	28.24	33.20	36.42	39.36	42.98	45.56	51.18
25	10.52	11.52	13.12	14.61	16.47	19.94	24.34	29.34	34.38	37.65	40.65	44.31	46.93	52.62
26	11.16	12.20	13.84	15.38	17.29	20.84	25.34	30.43	35.56	38.89	41.92	45.64	48.29	54.05
27	11.81	12.88	14.57	16.15	18.11	21.75	26.34	31.53	36.74	40.11	43.19	46.96	49.64	55.48
28	12.46	13.56	15.31	16.93	18.94	22.66	27.34	32.62	37.92	41.34	44.46	48.28	50.99	56.89
29	13.12	14.26	16.05	17.71	19.77	23.57	28.34	33.71	39.09	42.56	45.72	49.59	52.34	58.30
30	13.79	14.95	16.79	18.49	20.60	24.48	29.34	34.80	40.26	43.77	46.98	50.89	53.67	59.70
40	20.71	22.16	24.43	26.51	29.05	33.66	39.34	45.62	51.81	55.76	59.34	63.69	66.77	73.40
50	27.99	29.71	32.36	34.76	37.69	42.94	49.33	56.33	63.17	67.50	71.42	76.15	79.49	86.66
60	35.53	37.48	40.48	43.19	46.46	52.29	59.33	66.98	74.40	79.08	83.30	88.38	91.95	99.61
70	43.28	45.44	48.76	51.74	55.33	61.70	69.33	77.58	85.53	90.53	95.02	100.42	104.22	112.32
80	51.17	53.54	57.15	60.39	64.28	71.14	79.33	88.13	96.58	101.88	106.63	112.33	116.32	124.84
90	59.20	61.75	65.65	69.13	73.29	80.62	89.33	98.64	107.56	113.14	118.14	124.12	128.30	137.21
100	67.33	70.06	74.22	77.93	82.36	90.13	99.33	109.14	118.50	124.34	129.56	135.81	140.17	149.45

[a] $\chi^2_{d,u}$ = *u*th percentile of a χ^2 distribution with *d* degrees of freedom.
[b] = 0.0000393
[c] = 0.000157
[d] = 0.000982

Source: Reproduced in part with permission of the Biometrika Trustees, from Table 3 of *Biometrika Tables for Statisticians,* Volume 2, edited by E. S. Pearson and H. O. Hartley, published for the Biometrika Trustees, Cambridge University Press, Cambridge, England, 1972.

Categorical & Numerical

- A line chart with error bars displays information as a series of data points connected by straight line segments.
- Each data point is average of the numerical data for the corresponding category of the categorical variable with error bar showing standard error.

$$\mathrm{SE} = \frac{\sigma}{\sqrt{n}}$$

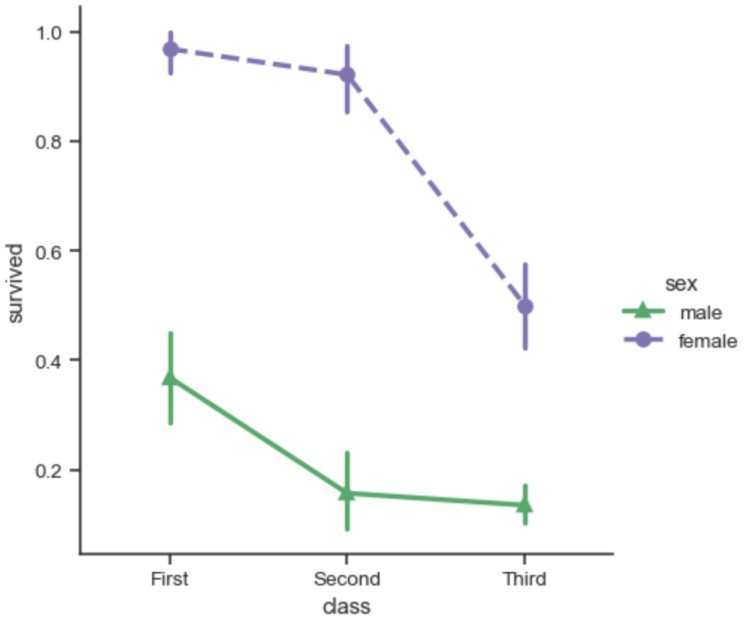
SE = standard error of the sample

σ = sample standard deviation

n = number of samples

Line chart

- A line chart with error bars displays information as a series of data points connected by straight line segments.
- Each data point is average of the numerical data for the corresponding category of the categorical variable with error bar showing SD/SE.



t-test

Two-Sample *t* Test for Independent Samples with Equal Variances

Suppose we wish to test the hypothesis H_0: $\mu_1 = \mu_2$ vs. H_1: $\mu_1 \neq \mu_2$ with a significance level of α for two normally distributed populations, where σ^2 is assumed to be the same for each population.

Compute the test statistic:

$$t = \frac{\bar{x}_1 - \bar{x}_2}{s\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

where $s = \sqrt{\left[(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2\right] / (n_1 + n_2 - 2)}$

If $t > t_{n_1 + n_2 - 2, 1-\alpha/2}$ or $t < -t_{n_1 + n_2 - 2, 1-\alpha/2}$

then H_0 is rejected.

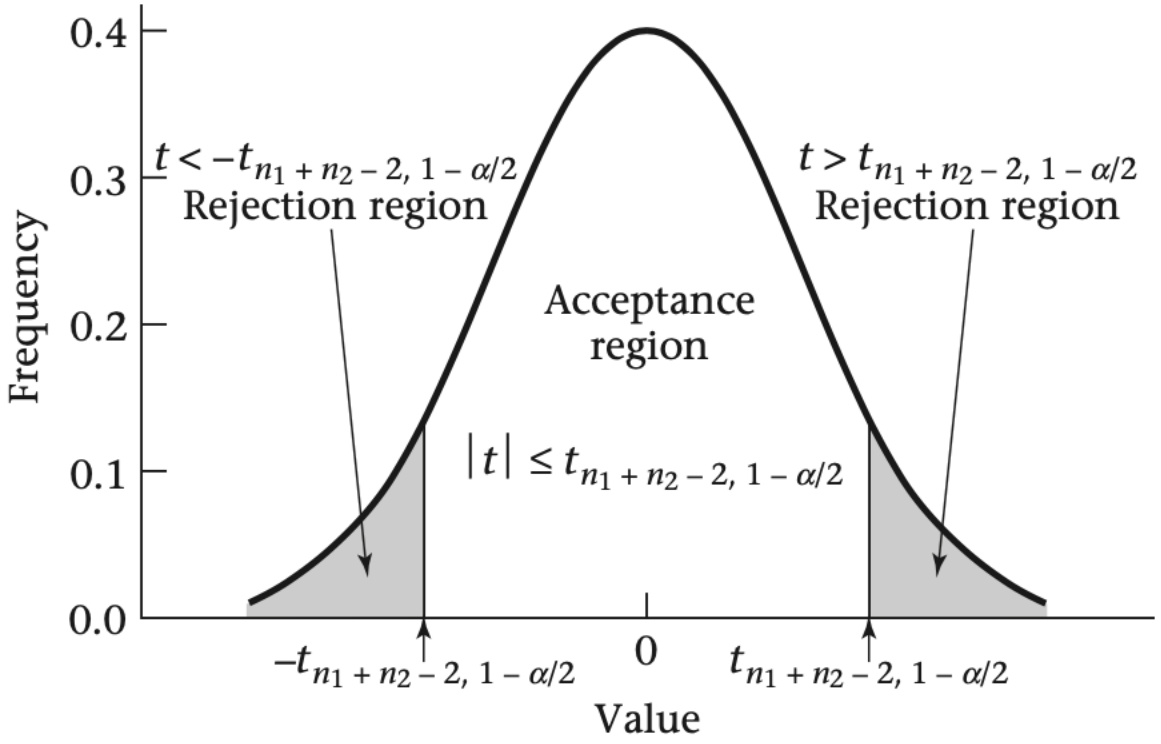
If $-t_{n_1+n_2-2,1-\alpha/2} \leq t \leq t_{n_1+n_2-2,1-\alpha/2}$

then H_0 is accepted.

The **pooled estimate of the variance** from two independent samples is given by

$$s^2 = \frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}$$

t-test


0.4
0.3
0.2
0.1
0.0
Frequency
$t < -t_{n_1 + n_2 - 2,\ 1 - \alpha/2}$
Rejection region
$t > t_{n_1 + n_2 - 2,\ 1 - \alpha/2}$
Rejection region
Acceptance region
$|t| \leq t_{n_1 + n_2 - 2,\ 1 - \alpha/2}$
$-t_{n_1 + n_2 - 2,\ 1 - \alpha/2}$
0
$t_{n_1 + n_2 - 2,\ 1 - \alpha/2}$
Value

t-test

Hypertension Suppose a sample of eight 35- to 39-year-old nonprenant, premenopausal OC users who work in a company and have a mean systolic blood pressure (SBP) of 132.86 mm Hg and sample standard deviation of 15.34 mm Hg are identified. A sample of 21 nonpregnant, premenopausal, non-OC users in the same age group are similarly identified who have mean SBP of 127.44 mm Hg and sample standard deviation of 18.23 mm Hg. What can be said about the underlying mean difference in blood pressure between the two groups?

$$t = \frac{\bar{x}_1 - \bar{x}_2}{s\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}$$

where $s = \sqrt{\left[(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2\right] / (n_1 + n_2 - 2)}$

The common variance is first estimated:

$$s^2 = \frac{7(15.34)^2 + 20(18.23)^2}{27} = \frac{8293.9}{27} = 307.18$$

or $s = 17.527$. The following test statistic is then computed:

$$t = \frac{132.86 - 127.44}{17.527\sqrt{1/8 + 1/21}} = \frac{5.42}{17.527 \times 0.415} = \frac{5.42}{7.282} = 0.74$$

$$t_{27,.975} = 2.052$$

$-2.052 \le 0.74 \le 2.052$, it follows that H_0 is accepted using a two-sided test at the 5% level

Analysis of Variance (ANOVA)

- The ANOVA test assesses whether the averages of more than two groups are statistically different from each other.

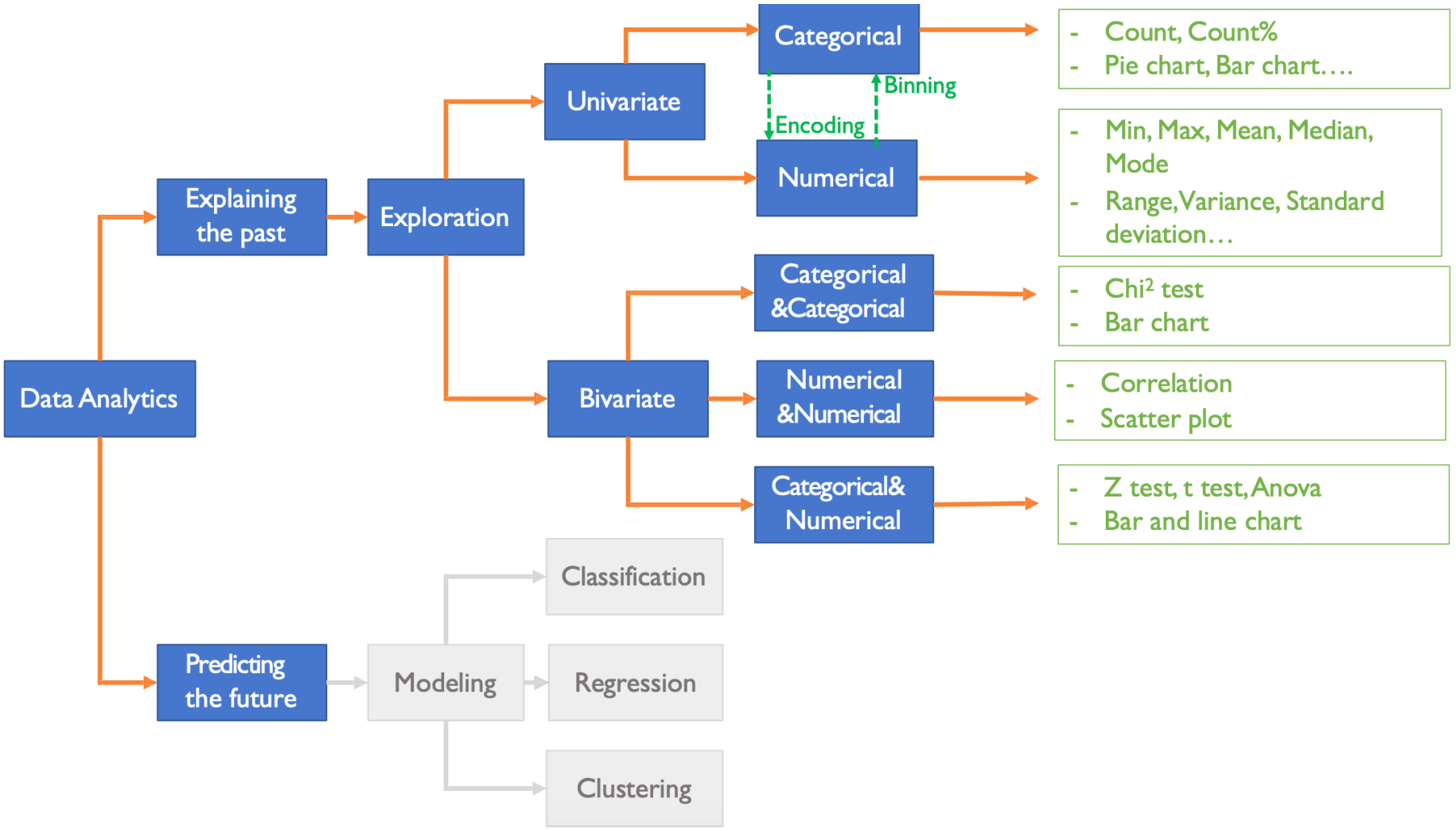
Short Computational Form for the Between SS and Within SS

$$\text{Between SS} = \sum_{i=1}^{k} n_i \bar{y}_i^2 - \frac{\left(\sum_{i=1}^{k} n_i \bar{y}_i\right)^2}{n} = \sum_{i=1}^{k} n_i \bar{y}_i^2 - \frac{y_{..}^2}{n}$$

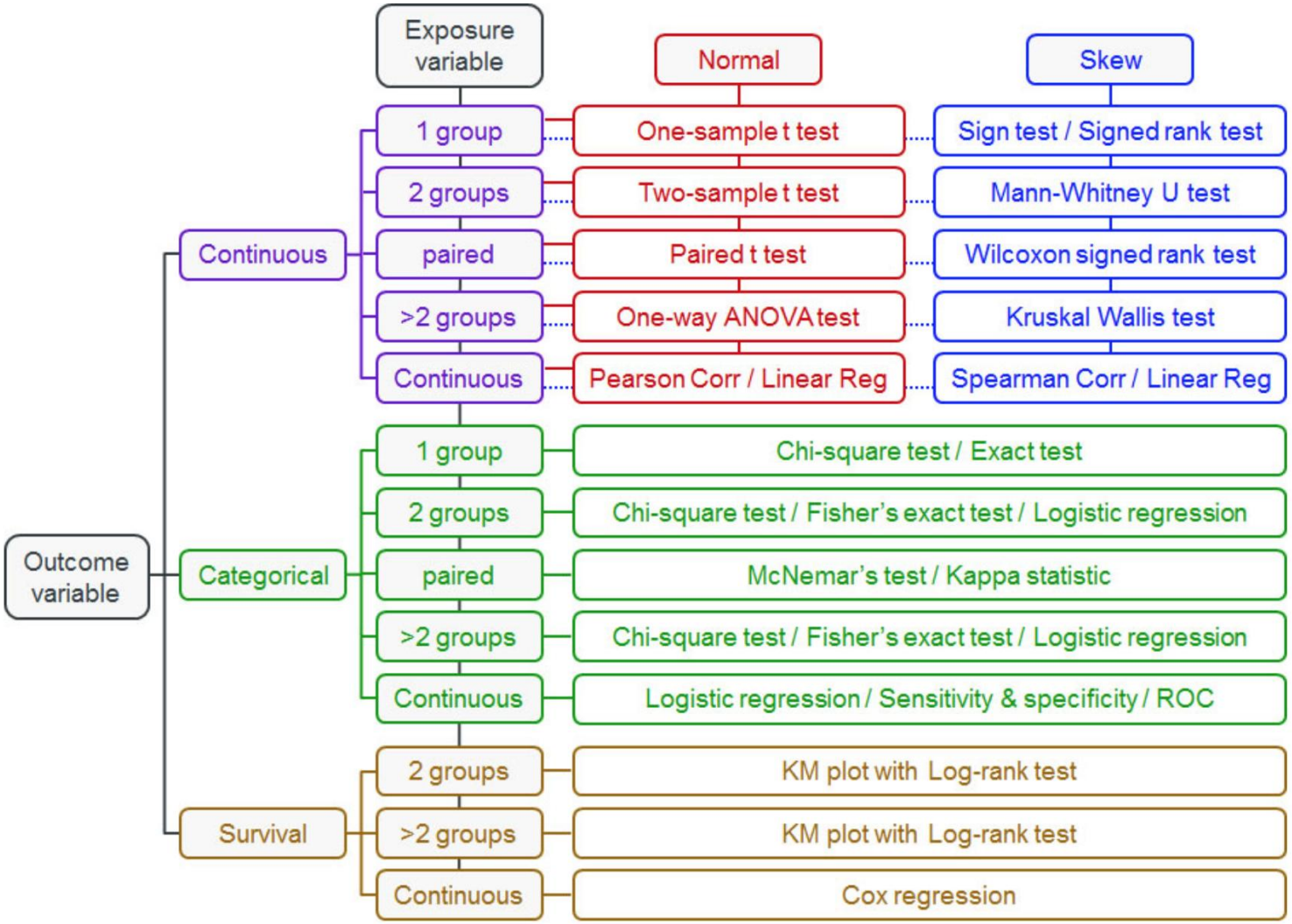
$$\text{Within SS} = \sum_{i=1}^{k} (n_i - 1) s_i^2$$

where $y..$ = sum of the observations across all groups—i.e., the grand total of all observations over all groups—and n = total number of observations over all groups.

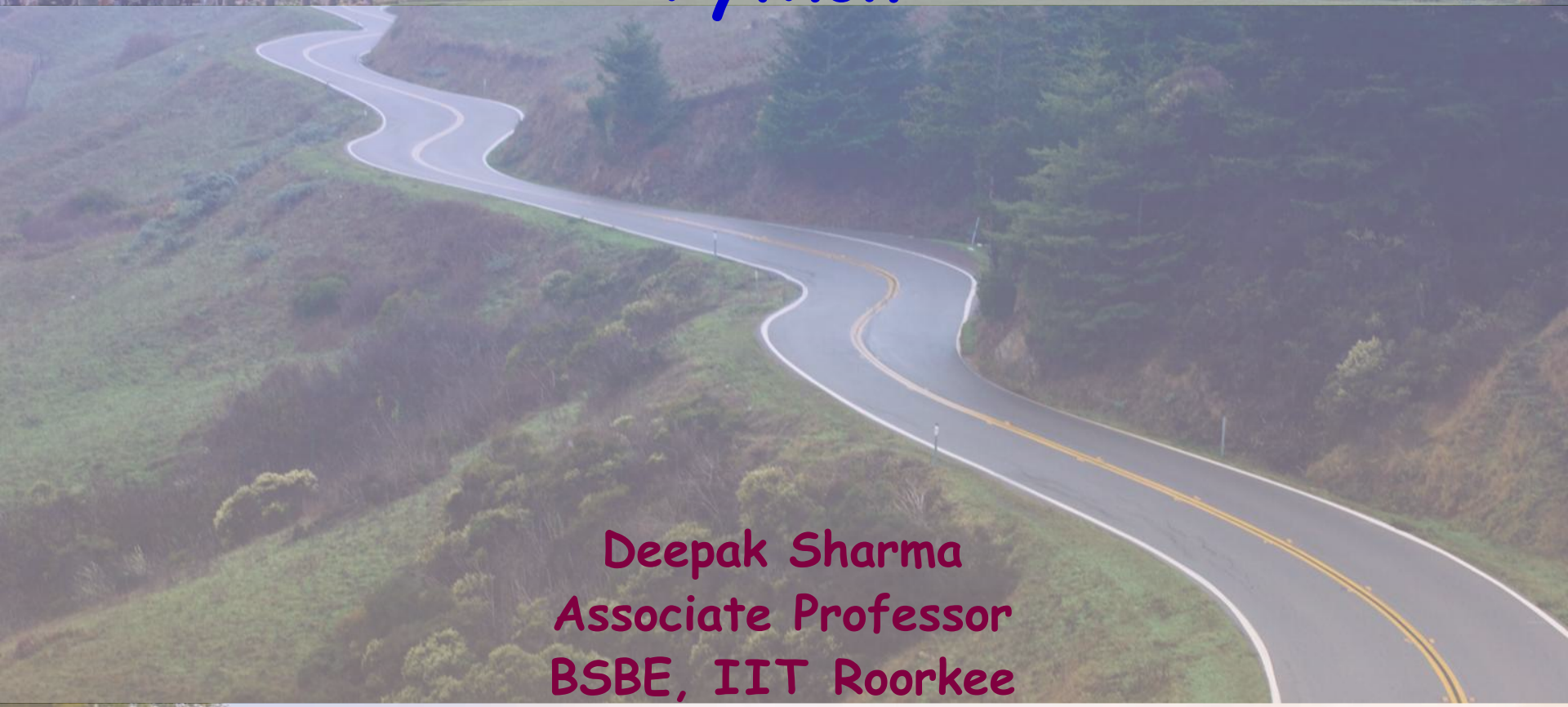
Summary


Data Analytics
Explaining the past
Exploration
Univariate
Categorical
Numerical
Encoding
Binning
Bivariate
Categorical &Categorical
Numerical &Numerical
Categorical& Numerical
Predicting the future
Modeling
Classification
Regression
Clustering
- Count, Count%
- Pie chart, Bar chart….
- Min, Max, Mean, Median, Mode
- Range,Variance, Standard deviation…
- Chi² test
- Bar chart
- Correlation
- Scatter plot
- Z test, t test,Anova
- Bar and line chart

Flow chart of statistical tests



Python



Deepak Sharma
Associate Professor
BSBE, IIT Roorkee

Scoping

Python — vi scoping.py — 70×22

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

########################################
#User-defined function and scoping
########################################

def f(x):                    #name x used as a formal parameter
        y = 1
        x = x + y
        print('x =', x)
        return x

x = 2
y = 7

z = f(x)                     #value of x used as actual parameter
print('z =', z)
print('x =', x)
print('y =', y)
~
~
"scoping.py" 19L, 408C
```

The formal parameter x and the local variable y exist only within the scope of the definition of f

Python — -bash — 48×6

```
Deepaks-MacBook-Pro:Python deepak$ ./scoping.py
x = 3
z = 3
x = 2
y = 7
Deepaks-MacBook-Pro:Python deepak$
```

for else loop

AS2023 — vi for_else.py — 70×14

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

#################################
#for else
#################################

for x in range(6):
  print(x)
else:
  print("Finally finished!")
~
~
~
"for_else.py" 10L,
```

AS2023 — -bash — 58×9

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./for_else.py
0
1
2
3
4
5
Finally finished!
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

Prime numbers

AS2023 — vi prime_number_range_forelse.py — 70×18

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

################################
#range
################################
lower=int(input("Enter the lower interval: "))
upper=int(input("Enter the upper interval: "))

for num in range(lower,upper+1):
        if num>1:
                for i in range(2,num):
                        if(num%i)==0:
                                break
                else:
~
~
"prime_nu
```

AS2023 — -bash — 74×9

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./prime_number_range_forelse.py
Enter the lower interval: 2
Enter the upper interval: 12
2
3
5
7
11
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

while else loop

AS2023 — vi while_else.py — 70×15

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

################################
#while else
################################

i = 1
while i < 6:
        print(i)
        i += 1
else:
        print("i is no longer less than 6")
~
~
"while_else.py"
```

AS2023 — -bash — 63×9

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./while_else.py
1
2
3
4
5
i is no longer less than 6
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

split

AS2023 — vi split.py — 72×14

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

####################################
#Split
####################################

str1 = "Welcome to IIT Roorkee"

x = str1.split()

print(x)
~
~
"split.py" 11L, 207C
```

AS2023 — -bash — 63×6

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./split.py
['Welcome', 'to', 'IIT', 'Roorkee']
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

split

AS2023 — vi split2.py — 72×14

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

####################################
#Split
####################################

str1 = "IIT-R,BSBE,BEC101,Deepak"

x = str1.split(",")

print(x)
~
~
"split2.py" 11L, 212C
```

AS2023 — -bash — 63×6

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./split2.py
['IIT-R', 'BSBE', 'BEC101', 'Deepak']
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

Lists to Dictionary

AS2023 — vi list_to_dict.py — 70×15

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

###################################
#List to Dictionary
###################################

index = [1, 2, 3]
languages = ['python', 'c', 'c++']

dictionary = dict(zip(index, languages))
print(dictionary)
~
~
~
"list_to_dict.py
```

AS2023 — -bash — 63×7

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./list_to_dict.py
{1: 'python', 2: 'c', 3: 'c++'}
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

Unordered Dictionary

AS2023 — vi dictionary_unordered.py — 66×19

```
#################################
#dictionary unordered
#################################

my_dict = {'a':1, 'b':2, 'c':3, 'd':4}
print(my_dict)

my_dict.pop('c')

print(my_dict)

my_dict['c'] = 3

print(my_dict)
~
~
~
~
"diction
```

AS2023 — -bash — 75×7

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ python2 dictionary_unordered.py
{'a': 1, 'c': 3, 'b': 2, 'd': 4}
{'a': 1, 'b': 2, 'd': 4}
{'a': 1, 'c': 3, 'b': 2, 'd': 4}
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

Ordered Dictionary

AS2023 — vi dictionary_ordered.py — 66×19

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

################################
#dictionary ordered
################################

my_dict = {'a':1, 'b':2, 'c':3, 'd':4}
print(my_dict)

my_dict.pop('c')

print(my_dict)

my_dict['c'] = 3

print(my_dic
~
~
"dictionary_
```

AS2023 — -bash — 70×7

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./dictionary_ordered.py
{'a': 1, 'b': 2, 'c': 3, 'd': 4}
{'a': 1, 'b': 2, 'd': 4}
{'a': 1, 'b': 2, 'd': 4, 'c': 3}
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

update Dictionary and max value

AS2024 — vi update_max_dict2.py — 70×19

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

#####################################
#Add elements in  Dictionary
#####################################

myDict = {'deepak':1, 'anamika':2, 'manavi':3}

myDict.update({'pranavi':4})
print(myDict)

freq = zip(myDict.values(),myDict.keys())
print(tuple(freq))

freq = max(zip(myDict.values(),myDict.keys()))
print(freq[0])
~
~
"update_max_dict2
```

AS2024 — -bash — 65×6

```
(base) Deepaks-MacBook-Pro:AS2024 deepak$ ./update_max_dict2.py
{'deepak': 1, 'anamika': 2, 'manavi': 3, 'pranavi': 4}
((1, 'deepak'), (2, 'anamika'), (3, 'manavi'), (4, 'pranavi'))
4
(base) Deepaks-MacBook-Pro:AS2024 deepak$
```

Merge Dictionaries

AS2023 — vi merge_dict.py — 70×17

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

###################################
#Merge Dictionaries
###################################

dict_1 = {1: 'a', 2: 'b'}
dict_2 = {2: 'c', 4: 'd'}

dict_3 = dict_2
#dict_3 = dict_2.copy()
dict_3.update(dict_1)

print(dict_3)
~
~
"merge_dict.py"
```

AS2023 — -bash — 63×7

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./merge_dict.py
{2: 'b', 4: 'd', 1: 'a'}
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

Sort Dictionary

AS2023 — vi sort_dict.py — 70×21

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

#####################################
#Sort Keys in  Dictionary
#####################################

myDict = {'deepak':1, 'anamika':2, 'pranavi':4, 'manavi':3}

myKeys = list(myDict.keys())
myKeys.sort()

sorted_dict = {}
for i in myKeys:
        sorted_dict.update({i: myDict[i]})
#sorted_dict = {i: myDict[i] for i in myKeys}

print(myDict)
print(sorted_dict)
~
~
"sort_dict.py" 18L,
```

AS2023 — -bash — 61×5

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./sort_dict.py
{'deepak': 1, 'anamika': 2, 'pranavi': 4, 'manavi': 3}
{'anamika': 2, 'deepak': 1, 'manavi': 3, 'pranavi': 4}
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

Time calculation

AS2023 — vi time.py — 70×17

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

###################################
#Time
###################################

import time

start = time.time()

print(123456789*123456789)

end = time.time()
print(end - start)
~
~
"time.py" 14L,
```

AS2023 — -bash — 63×7

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./time.py
15241578750190521
4.6014785766601 56e-05
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

Classes

- An abstract data type is a set of objects and the operations on those objects
- These are bound together so that one can pass an object from one part of a program to another and in doing so provide access not only to data attributes of the objects but also to operations – easy to manipulate that data
- Implements data abstractions using classes

Classes

Python — vi classes1.py — 104×15

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

class Person:    #name of the class and may be followed by some information about the class
  def __init__(self, name, age):  # function to initiate class;
                                    self- formal parameter to which the actual parameter p1 is bound
    self.name = name
    self.age = age

p1 = Person('Deepak', 43)

print(p1.name)
print(p1.age)
~
~
"classes1.py" 11L, 459C
```

Python — -bash — 54×5

```
Deepaks-MacBook-Pro:Python deepak$ ./classes1.py
Deepak
43
Deepaks-MacBook-Pro:Python deepak$
```

Object Methods

Python — vi class_methods.py — 89×15

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age

  def myfunc(self):  #methods in objects are the functions that belong to the object
    print("Hello my name is " + self.name)

p1 = Person('Deepak', 43)
p1.myfunc()
~
~
"class_methods.py" 12L, 320C
```

Python — -bash — 56×5

```
Deepaks-MacBook-Pro:Python deepak$ ./class_methods.py
Hello my name is Deepak
Deepaks-MacBook-Pro:Python deepak$
```

Modify Objects

Python — vi class_modify.py — 89×18

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

class Person:
  def __init__(self, name, age):
    self.name = name
    self.age = age

  def myfunc(self):  #methods in objects are the functions that belong to the object
    print('Hello my name is', self.name, 'and age is', self.age)

p1 = Person('Deepak', 43)

p1.age = 42  #modify object
p1.myfunc()
del p1  #delete object
~
~
"class_modify.py" 15L,
```

Python — -bash — 56×5

```
Deepaks-MacBook-Pro:Python deepak$ ./class_modify.py
Hello my name is Deepak and age is 42
Deepaks-MacBook-Pro:Python deepak$
```

Classes (IntegerSet)

Python — vi class_integerset.py — 106×42

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3

import sys
sys.tracebacklimit = 0

class IntSet(object):
        """An IntSet is a set of objects"""  #docstring - describe the abstraction provided by the class
        #Information about the implementation
        #Value of the set is represented by a list of ints, self.vals
        #Each int in the set occurs in self.vals exactly once

        def __init__(self):  #class instantiation
                """Create an empty set of integers"""
                self.vals = []

        def insert(self, e):
                """Assumes e is an integer and inserts e into self"""
                if e not in self.vals:
                        self.vals.append(e)

        def member(self, e):
                """Assumes e is an integer
                Returns True if e is in self, and False otherwise"""
                return e in self.vals

        def remove(self, e):
                """Assumes e is an integer and removes e into self
                Raises ValueError if e is not in self"""
                try:
                        self.vals.remove(e)
                except:
                        raise ValueError(str(e) +' not found')

        def __str__(self):
                """Returns a string representation of self"""
                self.vals.sort()  #sorts the values
                result = ''
                for e in self.vals:
                        result = result + str(e) + ','
                return '{' + result[:-1] + '}'  #-1 omits trailing comma

"class_integerset.py" 48L, 1307C
```

Python — vi class_integerset.py — 62×9

```
s = IntSet()  #creates a new instance (object) of IntSet
s.insert(3)
s.insert(2)
s.insert(1)
print(s)
print(s.member(3))
s.remove(4)
```

Python — -bash — 69×10

```
Deepaks-MacBook-Pro:Python deepak$ ./class_integerset.py
{1,2,3}
True
ValueError: list.remove(x): x not in list

During handling of the above exception, another exception occurred:

ValueError: 4 not found
Deepaks-MacBook-Pro:Python deepak$
```

Classes (Persons)

Python — vi class_persons.py — 74×38

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3
import datetime

class Person(object):
        def __init__(self, name):
                """Create a person"""
                self.name = name
                try:
                        lastBlank = name.rindex(' ')
                        self.lastName = name[lastBlank+1:]
                except:
                        self.lastname = name
                self.birthday = None

        def getName(self):
                """Return's self's full name"""
                return self.name

        def getLastName(self):
                """Return's self's last name"""
                return self.lastName

        def setBirthday(self, birthdate):
                """Assumes birthdate si of the type datetime.date
                Sets self's birthday to birthdate"""
                self.birthday = birthdate

        def getAge(self):
                """Returns self's current age in days"""
                if self.birthday == None:
                        raise ValueError
                return (datetime.date.today() - self.birthday).days

        def __str__(self):
                """Returns self's name"""
                return self.name

"class_persons.py" 43L, 1057C
```

Python — vi class_persons.py — 60×8

```
p1 = Person('Deepak Sharma')
p2 = Person('Manavi Sharma')
p3 = Person('Pranavi Sharma')
p3.setBirthday(datetime.date(2013, 9, 8))
print(p3.getLastName())
print(p3.getName(), 'is', p3.getAge(), 'days old')
```

Python — -bash — 55×5

```
Deepaks-MacBook-Pro:Python deepak$ ./class_persons.py
Sharma
Pranavi Sharma is 2211 days old
Deepaks-MacBook-Pro:Python deepak$
```

Count/find program

AS2023 — vi count_char_line2.py — 65×44

```
#!/Library/Frameworks/Python.framework/Versions/3.7/bin/python3
import re

file_name = input("Enter the file name: ")
name_handle = open(file_name,"r")

i=0
j=0
for char in name_handle.read():
    if char!="\n":
        i=i+1
    if char==" ":
        j= j+1
print("No. of characters (with spaces):",i)
print("No. of characters (without spaces):",i-j)
name_handle.close()

name_handle = open(file_name,"r")
text=name_handle.read()
#print(text)
words=re.split(" |\n", text)
words.pop()
print(words)
print("No. of words:", len(words))
name_handle.close()

name_handle = open(file_name,"r")
lines=name_handle.readlines()
#print(lines)
lines_count=len(lines)
print("No. of lines:",lines_count)
name_handle.close()

str_name = input("Enter a search word/character: ")
if len(str_name)>1:
   matches=words.count(str_name)
else:
   char=[]
   char.extend(text)
   matches=char.count(str_name)
print("No. of matches:",matches)
name_handle.close()
~
"count_char_line2.py" 42L, 939C
```

AS2023 — vi file_for_count.txt —...

```
Deepak Sharma
BSBE
IIT Roorkee
~
~
"file_for_count.txt" 3L, 31C
```

AS2023 — -bash — 66×12

```
(base) Deepaks-MacBook-Pro:AS2023 deepak$ ./count_char_line2.py
Enter the file name: file_for_count.txt
No. of characters (with spaces): 28
No. of characters (without spaces): 26
['Deepak', 'Sharma', 'BSBE', 'IIT', 'Roorkee']
No. of words: 5
No. of lines: 3
Enter a search word/character: e
No. of matches: 4
(base) Deepaks-MacBook-Pro:AS2023 deepak$
```

Thank You !