# SCIPY

הספרייה scipy היא ספריית קוד פתוח של פייתון המשמשת לחישובים מתמטיים וטכניים.
הספרייה מכילה מודלים לאופטימיזציה לחישובים בתחומי האלגברה הלינארית, אינטגרציה, טורי פורייה עיבוד אותות ותמונות ועוד מלא.
הספרייה בנויה על בסיס מערכים מהספרייה numpy.

**התקנה:**

```
pip install scipy
```

**מה ההבדל העיקרי בין numpy ל-scipy ?**
באופן אידיאלי, numpy משמשת יותר לפעולות בסיסיות כמו מיון, אינדוקס, פונקציות אלמנטריות ועוד על מערכים ועל טיפוסי נתונים שונים.
לעומת זאת scipy מכילה את הפונקציות האלגבריות שחלקן אפשר אפילו למצוא ב-numpy במובן מסוים או עם מימוש לא מלא. בדר"כ אם מחפשים פיצרים חדשים, כמו חישוביים סטטיסטים מסובכים וכדו', סביר להניח שנמצא אותן ב-scipy.
scipy אומנם נבנת על בסיס numpy אך סביר להניח שאם נצטרך להשתמש בהן נייבא את שתי הספריות.

**תתי ספריות-**
ל-scipy יש כמה תתי ספריות (מרחבי שם) שימושיים בניהן נוכל למצוא ספריות כגון:
stats – לחישובים סטטיסטים והסתברויות של משתנים
cluster – ניתוחי אשכולות, כלומר קיבוץ אובייקטים לקבוצות קטנות (אשכולות), כך שכל הנתונים הדומים זה לזה(יותר משהם דומים לאחרים) יהיו באותה הקבוצה.
fftpack- לחישוב פונקציות פורייה.
ndimage- לניתוח תמונות עם n ממדים.
ועוד'.

**חישוב אינטגרל-**
אחת התתי ספריות של scipy היא integrate והיא מאפשרת לחשב אינטגרלים ממדים שונים.
הפונקציה הבסיסית ביותר לחישוב אינטגרל היא הפונקציה quad שמקבלת פונקציית חישוב, ותחום של האינטגרל, ומחזירה tuple עם התוצאה המשוערת של האינטגרל וחסם עליון על הטעות.
הפונקציה יוצאת מנקודת הנחה שיש רק נעלם אחד, אם נרצה לחשב אינטגרל של שני נעלמים נוכל להשתמש בפונקציה dblquad שמחשבת אינטגרל כפול, או tplquad לאינטגרל משולש:

```
import numpy as np
import matplotlib.pyplot as plt
from scipy import integrate

integrate.quad(lambda x: np.sin(x), 0,1)
```

```
(0.45969769413186023, 5.103669643922839e-15)
```

```
-np.cos(1) + np.cos(0)
```

```
0.45969769413186023
```

בדוגמא לעיל חישבנו את האינטגרל $\int_0^1 \sin(x)$ שידוע ששווה $-\cos(1)+\cos(0)$ , ובאמת התוצאות שוות.

אפשר גם להגדיר פונקציה עם פרמטרים נוספים שתהווה תבנית, ולהגדיר את הפרמטרים בזמן חישוב:

```
def integrand(x, a, b):
```

```
    return a*x**2 + b

integrate.quad(integrand, 0, 1, args=(2,1))
```

```
(1.6666666666666667, 1.8503717077085944e-14)
```

ניקח משל את האינטגרל הכפול הבא: $\int_0^\infty \int_1^\infty \frac{e^{-xt}}{t^n} dt\, dx$ , נוכל להשתמש בפונקציה dblquad כדי לחשב אותו:

```
def my_integral(n):
    return integrate.dblquad(lambda t, x: np.exp(-x*t)/t**n, 0, np.inf, lambda x: 1, lambda x: np.inf)

print(my_integral(4))
print(my_integral(3))
print(my_integral(2))
```

```
(0.2500000000043577, 1.298303346936809e-08)
(0.33333333325010883, 1.3888461883425516e-08)
(0.4999999999985751, 1.3894083651858995e-08)
```

שימו לב שהאינטגרל הראשון דורש גבולות ברורים ,ולאינטגרל הפנימי אנחנו מספקים גבולות לפי פונקציה.

**התמרת פורייה-**

התמרת פורייה היא כלי מתמטי המאפשר להמיר פונקציה נתונה כלשהי לסכום לינארי של פונקציות מחזוריות(פונקציות *sin* ו-*cos*).
ההתמרה היא בעצם טרנספורמציה ממרחב הזמן למרחב התדר, היות וכל הנושא עצום ומסובך לא נרחיב במילים, רק נגדי כי נוכל להשתמש בתת-ספרייה *fftpack* ובפרט במתודות *fft* לחישוב ההתמרה ו-*ifft* לחישוב ההתמרה ההופכית:

```
from scipy.fft import fft, ifft
x = np.array([1.0, 2.0, 1.0, -1.0, 1.5])
y = fft(x)
print(y)
yinv = ifft(y)
print(yinv)
```

```
[ 4.5        -0.j          2.08155948-1.65109876j -1.83155948+1.60822041j
 -1.83155948-1.60822041j  2.08155948+1.65109876j]

[ 1. +0.j  2. +0.j  1. +0.j -1. +0.j  1.5+0.j]
```

**חישובים עם פונקציית התפלגות ופונקציית צפיפות-**

פונקציית ההתפלגות המצטברת (cumulative distribution function ) היא פונקציה הסתברותית שערכיה קובעים את ההסתברות למאורעות מהצורה $X \le a$ לכל מאורע a ומשתנה מקרי X כלשהו.
למשל בקובייה הוגנת שההתפלגות שלה היא $p(a)=1/6$ עבור כל מספר טבעי a בין אחד לשש , אפשר להגדיר את פונקציית ההתפלגות המצטברת שלה כך:

$$F(a)=\begin{cases} 0: a<1 \\ 1/6: 1\le a<2 \\ 2/6: 2\le a<3 \\ 3/6: 3\le a<4 \\ 4/6: 4\le a<5 \\ 5/6: 5\le a<6 \\ 1: a\ge 6 \end{cases}$$

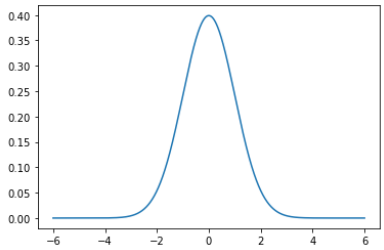
הסבר הסיכוי שנקבל מספר שקטן מאחד מהטלת קובייה הוא אפס ,כי אחד הוא המספר הקטן ביותר.
הסיכוי שנקבל מספר שהוא קטן מ-a כש-a בין 1 ל-2 ,או יותר מדויק מספר שהוא קטן מ-2 ,הוא $1/6$ כי רק אחד עונה על התנאי הזה, הסיכוי שנקבל מספר שהוא קטן משלוש הוא $2/6$ כי יש שני מספרים העונים על זה מתוך השש, 1 ו-2, וכו'.

פונקציית צפיפות(*probability density function*) היא פונקציה הסתברותית המתארת את הצפיפות של משתנה מקרי הכל נקודה במרחב המדגם.
ההסתברות שמשתנה מקרי יימצא בקטע מסוים היא האינטגרל של הצפיפות בקטע.

פונקציית הצפיפות של התפלגות נורמלית תקנית (בקטע [1,1-] (סטיית תקן אחד) ועם תוחלת 0) היא $f(x)=\frac{e^{\frac{-x^2}{2}}}{\sqrt{2\pi}}$
מה שאומר שהסיכוי שנקבל מספר בין 0 ל-1 ובין מינוס אחד לאפס שווה לחצי ושווה גם לאינטגרל בין אפס לאחד
על פונקציית הצפיפות: $p(x<0)=\int_{-1}^{0}\frac{e^{\frac{-x^2}{2}}}{\sqrt{2\pi}}=\int_{0}^{1}\frac{e^{\frac{-x^2}{2}}}{\sqrt{2\pi}}=¿\frac{1}{2}¿$

למה כל זה רלוונטי ל*scipy* ? כי ב-*scipy* יש אפשרות להשתמש בפונקציית הצפיפות ופונקציית ההתפלגות המצטברת של מרחב מסוים, למשל נניח יש לנו גרף של 10,000 נקודות פזורות במרחק שווה אחת מהשנייה, נרצה לראות את הסיכוי של כל אזור להיבחר באקראי לפי פונקציית הצפיפות של ההתפלגות הנורמלית:

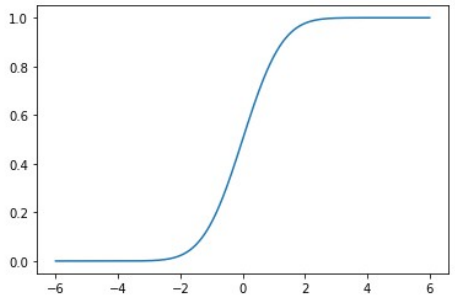
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm

space = np.linspace(-6,6,10000)
# loc = location, is the mean
# scale = standard deviation
fx = norm.pdf(space , loc=0 ,scale=1)
plt.plot(space , fx);
```




המשתנה *loc* מציין את התוחלת של הפונקצייה, והמשתנה *scale* את סטיית התקן (כמה הערכים 'פזורים' מהתוחלת).

עכשיו נראה את אותו מרחב לפי פונקציית ההסתברות המצטברת:

```
F_x = norm.cdf(space , loc=0 ,scale=1)
plt.plot(space , F_x);
```




הסבר על המהלך: לקחנו אינטרבל של מספרים בין -6 ל 6 שכל המספר במרחק שווה מהעוקב לו, ורצינו לבדוק מה ההסתברות לקבל כל מספר מהרחב הזה כאשר ההתפלגות היא נורמלית ומרחב המדגם שלנו הם כל אותם מספרים שלא רחוקים מהתוחלת (שהיא אפס) ביותר מאחד.
מה שיצא לנו הוא הגרף העליון, הגרף השני מתאר לנו את הסיכוי שנקבל מספר כלשהו שקטן ממספר אחר, למשל מה ההסתברות לקבל מספר שקטן ממינוס 0.5 בערך $\frac{1}{4}$ וכו'.