НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

Звіти до комп’ютерних практикумів з кредитного модуля “Вступ до Data Science”

Виконав

Студент групи ІТ-02

Макаров І.С.

Перевірив:

Професор кафедри ОТ ФІОТ
Писарчук О.О.

Київ – 2022

Комп‘ютерний практикум No 1.

ДОСЛІДЖЕННЯ СТАТИСТИЧНИХ ХАРАКТЕРИСТИК ЕКСПЕРИМЕНТАЛЬНИХ ДАНИХ

Мета:

Виявити дослідити та узагальнити особливості застосування методів статистичного аналізу для задач визначення статистичних характеристик вхідного потоку експериментальних даних з використанням спеціалізованих пакетів мови програмування Python.

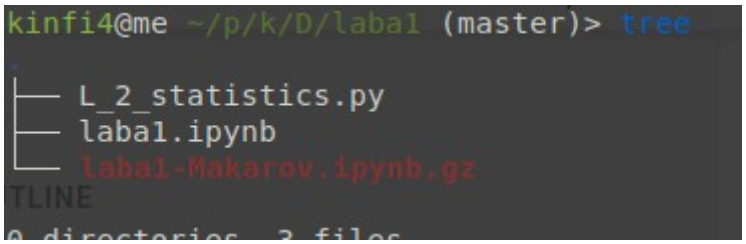
Варіант:

Я 10й у списку, той ось мій варіант, другий рівень складності

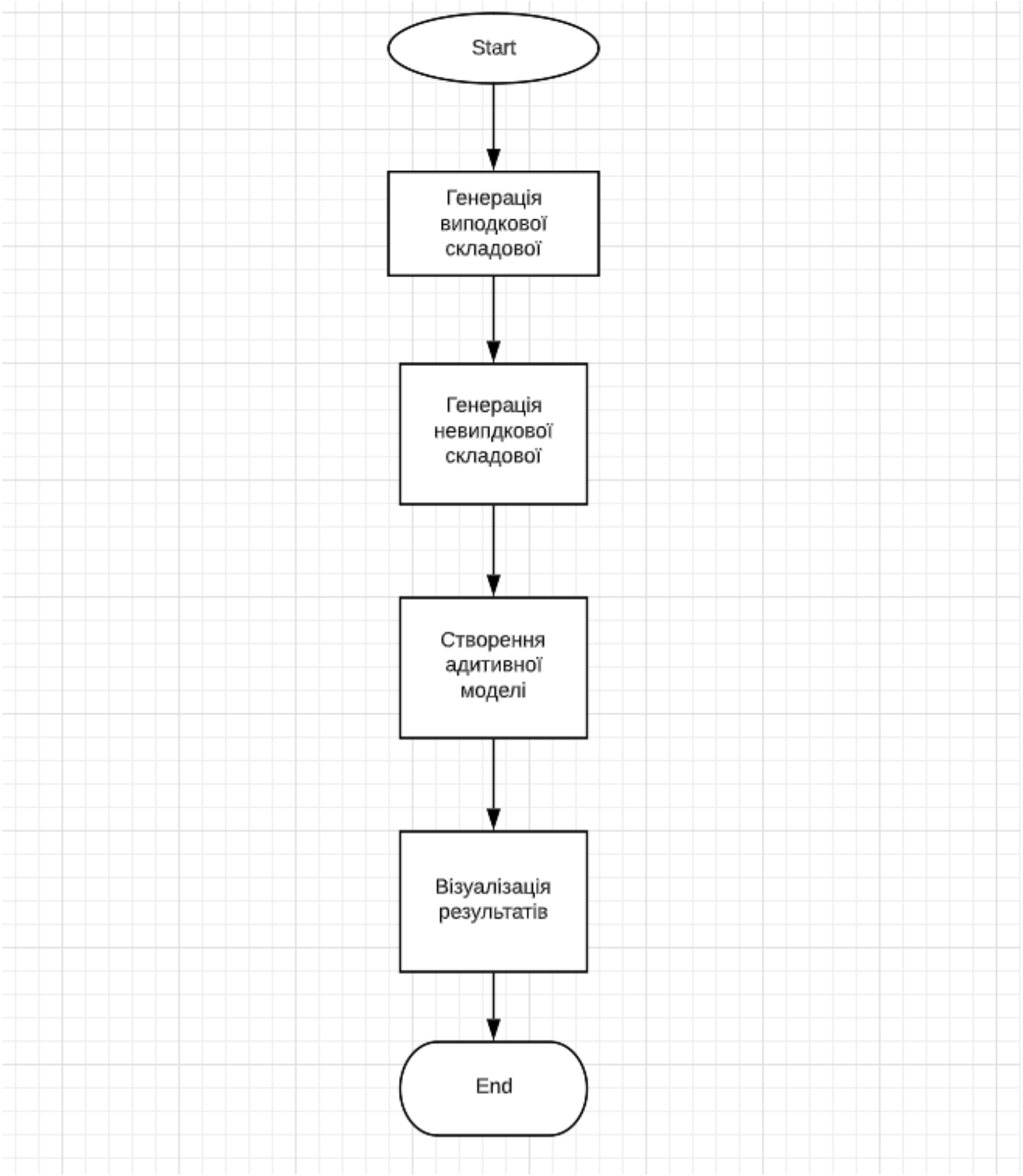
Закон зміни похибки – рівномірний, нормальний; Закон зміни досліджуваного процесу – кубічний, лінійний.

ВИКОНАННЯ

Стурктура проекту це один файл laba1.ipynb.



Діаграма:


Start
Генерація виподкової складової
Генерація невипдкової складової
Створення адитивної моделі
Візуалізація результатів
End

Перш за все вкажемо потрібні нам імпорти, та напишемо функцію, що буде виводити числові характеристики переданої вибірки. Також вкажемо default розмір наших вибірок.

In 1
```
import numpy as np
import matplotlib.pyplot as plt
```

In 2
```
def calculate_metrics(sample: np.ndarray) -> None:
    print(f'Mathematical expectation: {np.median(sample)}')
    print(f'Dispersion: {np.var(sample).round(3)}')
    print(f'Sigma: {np.std(sample).round(3)}')
    print('=' * 50)
```

Constants

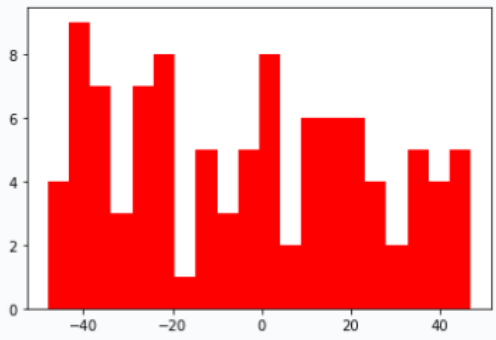
In 15
```
default_distribution_size = 10000
```

ЧАСТИНА 1

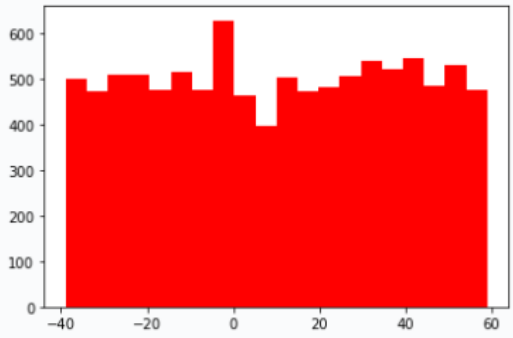
Спочатку розберемось з рівномірним розподілом, де невипадкова складова це кубічний процесс.

In 4
```
small_distribution_size = 100
uniform_distribution = np.random.uniform(-50, 50, small_distribution_size).astype(int)

plt.hist(uniform_distribution, bins=20, color='r')
plt.show()
calculate_metrics(uniform_distribution)

uniform_distribution = np.random.uniform(-50, 50, default_distribution_size).astype(int)

plt.hist(uniform_distribution, bins=20, color='r')
plt.show()
calculate_metrics(uniform_distribution)
```

В даному коді, я створив дві вибірки, першу зі 100 елементів, друга з default значенням розміру вибірки (10 000) елементів. Зробив я це, аби показати ЗВЧ на практиці.




```
Mathematical expectation: -1.5
Dispersion: 772.448
Sigma: 27.793
```




```
Mathematical expectation: 10.0
Dispersion: 809.659
Sigma: 28.455
```

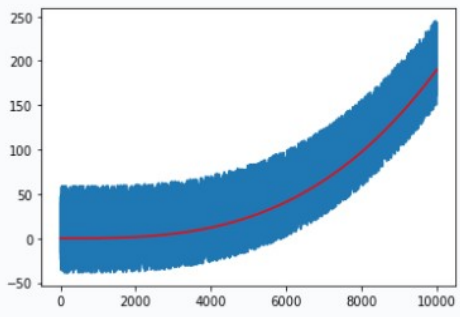
Як видно мат характеристики у двох вибірок майже однакові, однак в першій вгадати рівномірний розподіл навряд можно, все через малу кількість елементів вибірки.

Додамо невипадкову складову

Correcting the distribution

In 26

```
coefficient = 0.00000000019
correction_array = np.arange(uniform_distribution.shape[0])
correction_array = (correction_array ** 3) * coefficient

corrected_distribution = uniform_distribution + correction_array

plt.plot(corrected_distribution)
plt.plot(correction_array, color='red')
plt.show()

calculate_metrics(corrected_distribution)
```

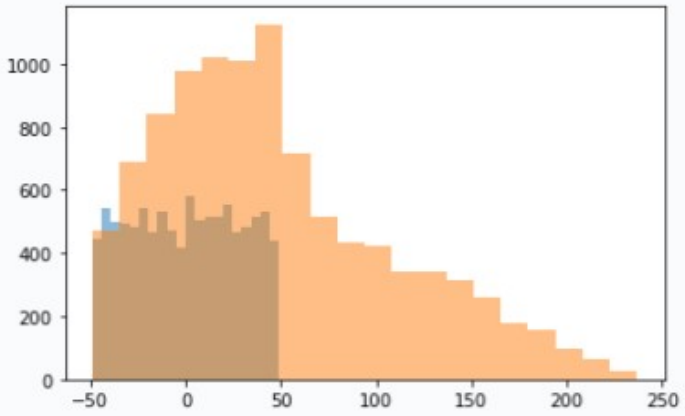



```
Mathematical expectation: 46.54511374946
Dispersion: 3711.168
Sigma: 60.919
```

Тепер подивимось на порівняння нашого розподілу, і окремо глянемо як виглядає наша адитивна модель.

Let's compare

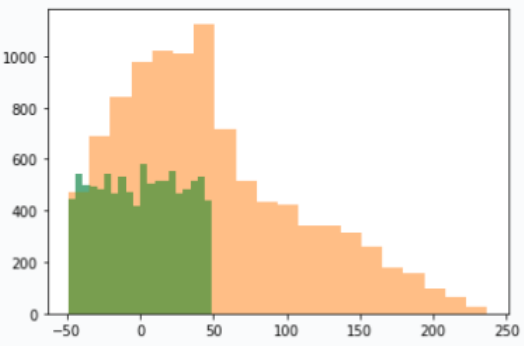
```
In 6
1 plt.hist(uniform_distribution, bins=20, alpha=0.5)
2 plt.hist(corrected_distribution, bins=20, alpha=0.5)
3
4 plt.show()
```



Як видно закон розпався, що свідчить про нелінійний невипадковий тренд
Тепер, оскільки ми знаємо як виглядає наш correction_array, то можемо його відняти

Take in consideration the dinamic of change

```
In 7
1 corrected_back_distribution = corrected_distribution - correction_array
2
3 plt.hist(uniform_distribution, bins=20, alpha=0.5)
4 plt.hist(corrected_distribution, bins=20, alpha=0.5)
5 plt.hist(corrected_back_distribution, bins=20, alpha=0.5)
6
7 plt.show()
```

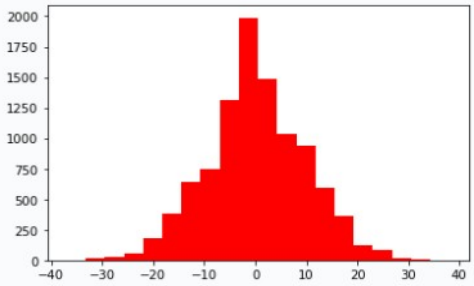


Частина 2

Тепер час сісти за нормальний закон, початок схожий.

Normal Distribution

```
In 12
mean, sigma = 0, 10
normal_distribution = np.random.normal(mean, sigma, size=default_distribution_size).astype(int)

plt.hist(normal_distribution, bins=20, color='r')
plt.show()
calculate_metrics(normal_distribution)
```

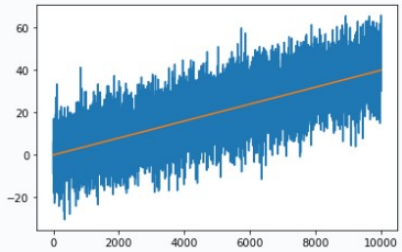



```
Mathematical expectation: 0.0
Dispersion: 91.928
Sigma: 9.588
==================================================
```

Побудуємо невипадкову складову.

Correcting the distribution

```
In 13
coefficient = 0.004
correction_norm_array = np.arange(normal_distribution.shape[0])
correction_norm_array = correction_norm_array * coefficient

corrected_norm_distribution = normal_distribution + correction_norm_array

plt.plot(corrected_norm_distribution)
plt.plot(correction_norm_array)
plt.show()

calculate_metrics(corrected_norm_distribution)
```

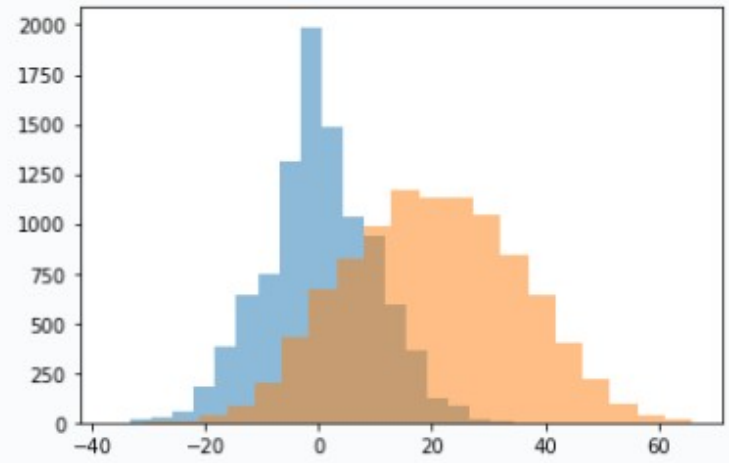



```
Mathematical expectation: 20.092
Dispersion: 226.245
Sigma: 15.041
==================================================
```

Але тут в нас лінійний закон невипадкової величини.
Ну що, візуалізуємо, що в нас вийшло

Let's compare

```
n 14  1  plt.hist(normal_distribution, bins=20, alpha=0.5)
      2  plt.hist(corrected_norm_distribution, bins=20, alpha=0.5)
      3
      4  plt.show()
```

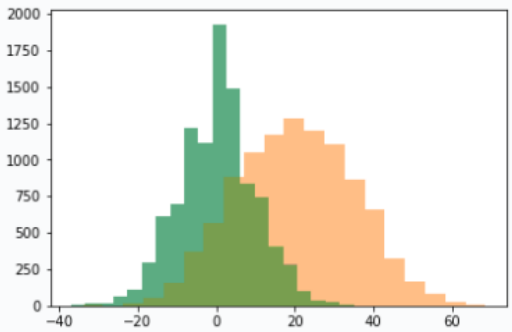


Як видно, закон не розпався, бо тренд лінійний.

Ну і нарешті, оскільки ми знаємо correction_norm_array, можемо відняти його

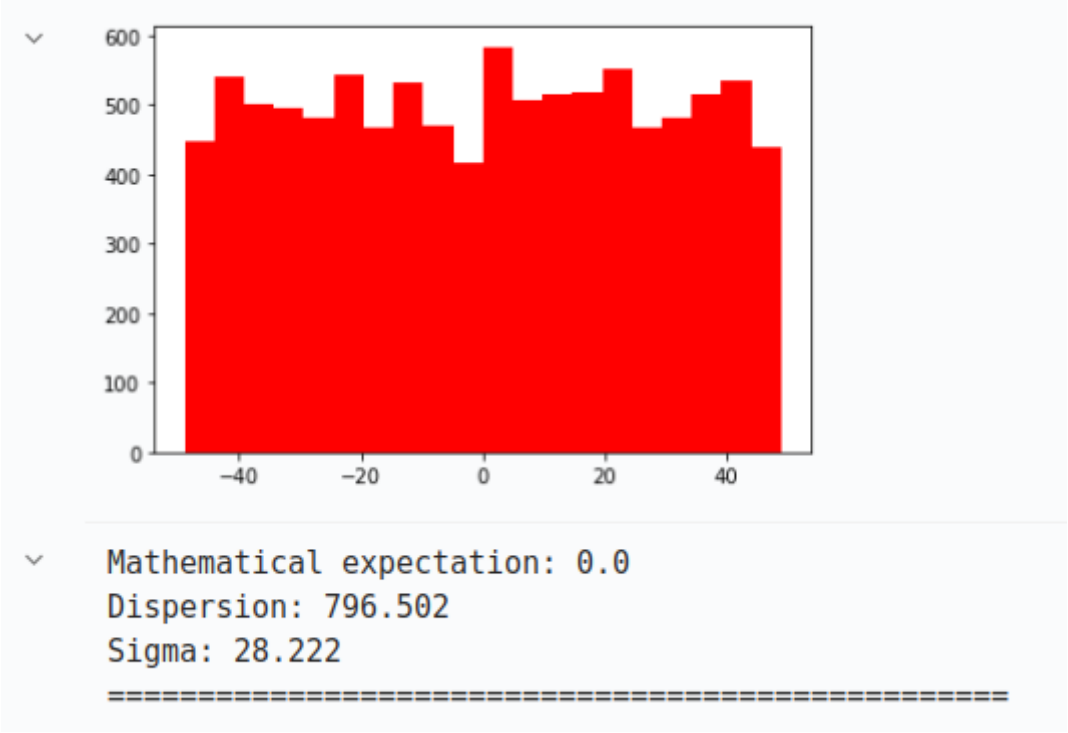
Take in consideration the dinamic of change

```
In 11  1  corrected_back_norm_distribution = corrected_norm_distribution - correction_norm_array
       2
       3  plt.hist(normal_distribution, bins=20, alpha=0.5)
       4  plt.hist(corrected_norm_distribution, bins=20, alpha=0.5)
       5  plt.hist(corrected_back_norm_distribution, bins=20, alpha=0.5)
       6
       7  plt.show()
```



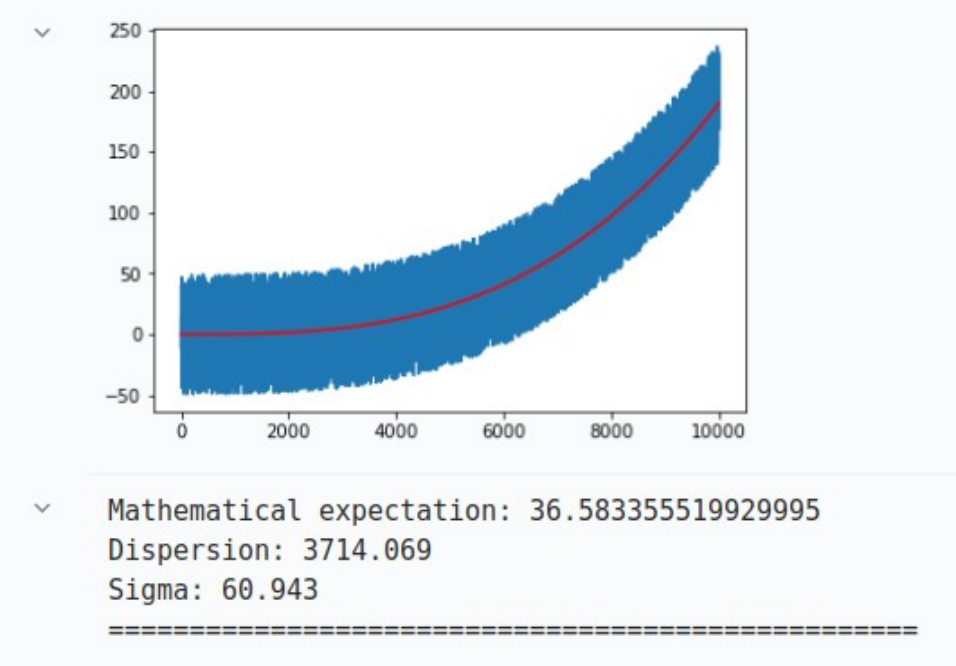
ДОВЕДЕННЯ ПРАЦЕЗДАТНОСТІ

Думаю з графіків доволі очевидно, що скрипт працює. В першій частині ми генерували рівномірно розподілену вибірку, від -50 до 50.

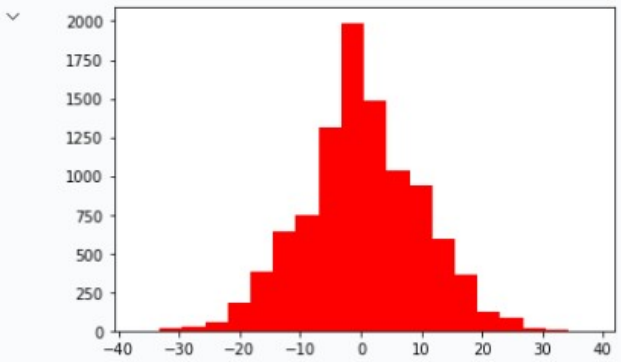


Доволі очевидно, що візуалзація, та мат. характеристики описують такий розподіл.

Так само адитивна модель, з кубічною невипадковою складовою очевидно справедлива

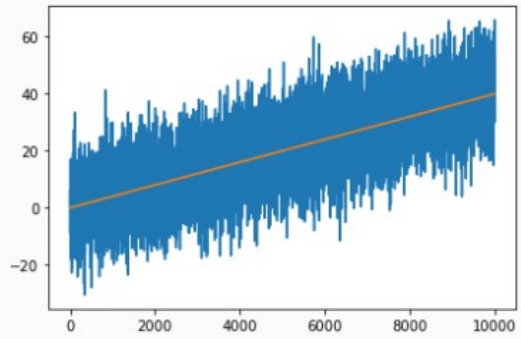


Щодо нормального розподілення.
Я вказував M = 0 та sigma = 10, що більш менш в нас і вийшло.



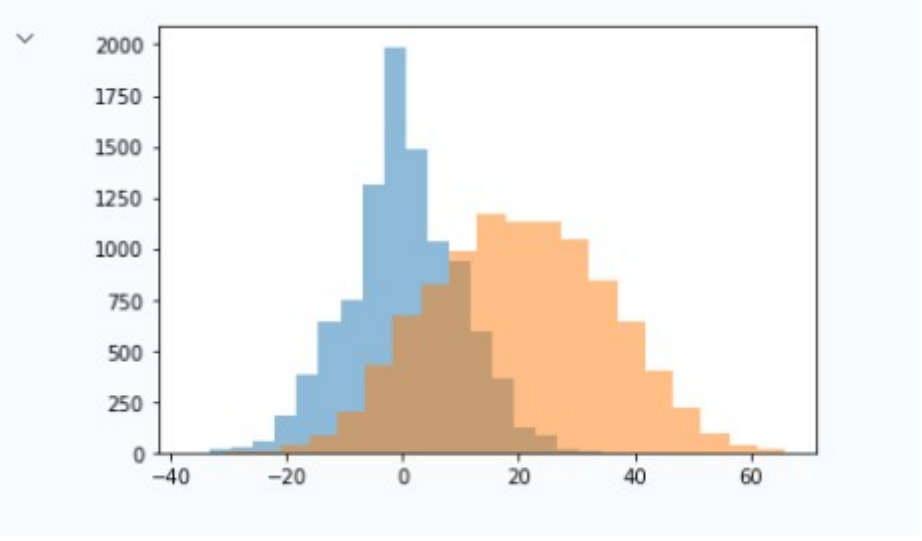

```
Mathematical expectation: 0.0
Dispersion: 91.928
Sigma: 9.588
==================================================
```

Але тут вже тренд лінійний




```
Mathematical expectation: 20.092
Dispersion: 226.245
Sigma: 15.041
==================================================
```

Що до речі, як видно не руйнує наш закон



ВИСНОВОК

В ході роботи я створив скрипт на мові програмування Python. Скрипт генерує дві випадкові величини, рівномірного та норм розподілу. А також дві невипадкові величини, лінійну та кубічну. Та створює адитивні моделі відповідних вип та невип величин. А також робить візуалізацію.