Loi Hypergéométrique

- ✓ MESRAR Hamza
- ✓ EL MAZOUTI Ilias
- ✓ IMARIREN Amine

Encadré Par : Pr.A.GHAZDALI

Plan

- Objectif
- Propriétés et Paramètres
- Exemple
- Simulation sous Python

Objectif

La loi **hypergéométrique** intervient dans le cas de plusieurs expériences aléatoires indépendantes auxquelles on associe un caractère étudié.

Explication :

Soit **N** l'effectif total de la population dans laquelle on prélève au hasard et sans remise **n** individus. La population est composée de **A** individus possédant le caractère étudié, et **m** individus ne possèdent pas le caractère étudié.

Différence entre Loi Binomiale et Hypergéométrique

La loi **hypergéométrique** et la loi **binomiale** indiquent toutes les deux le nombre d'occurrences d'un événement au cours d'un nombre fixe d'essais. Pour la loi **binomiale**, la probabilité est la même pour chaque essai. Dans le cadre de la loi **hypergéométrique**, chaque essai change la probabilité de chacun des essais suivants du fait de **l'absence de remise**.

Propriétés et Paramètres

Définition :

Soient n, N, A trois entiers positifs tels que $n \leq N$, $A < N$. On dit que X suit une loi hypergéométrique de paramètres N, n et A notée :

$$X \sim H(N,\ A,\ n)$$

Avec :

N : *Effectif total de la population*

A : Individus possédant le caractère étudié

n : *nombre des individus tirés*

Définition (suite)

Si :

$$- X(\Omega) \subset \{0, 1, \dots \dots \dots, n\}$$

La variable X *suit alors la loi Hypergeometrique définie par :*

$$- \quad P(X = k) = \frac{C_A^k \times C_{N-A}^{n-k}}{C_N^n}$$

Cas Particulier :

$$\mathrm{P}(\mathrm{X} \leq k) = \sum_{x \leq k} P(X = x)$$

Espérance et Variance

Espérance :

L'**espérance** d'une variable aléatoire X suivant une loi **hypergéométrique** de paramètres (N, A, n) est la même que celle d'une variable **binomiale** de paramètre (n, A) :

$$\mathbb{E}(X) = n \times \frac{A}{N}$$

Variance :

La **variance** d'une variable aléatoire suivant une loi **hypergéométrique** de paramètres (N, A, n) est :

$$V(X) = \frac{N-n}{N-1} \times \frac{nA}{N} \times \left(1 - \frac{A}{N}\right)$$

Exemple

On prend au **hasard**, en **même temps**, **trois** ampoules dans un lot de **15** dont **5** sont **défectueuses**. Calculer la probabilité des événements suivants :

A : au moins une ampoule est défectueuse.

B : les 3 ampoules sont défectueuses.

C : exactement une ampoule est défectueuse.

Correction

Pour A :

$$P(A) = P(X \geq 1)$$

$$= 1 - P(X = 0)$$

$$= 1 - \frac{C_5^0 \times C_{15-5}^{3-0}}{C_{15}^3}$$

$$= 0,736$$

Correction

Pour B :

$$P(B) = P(X = 3)$$

$$= \frac{C_5^3 \times C_{15-5}^{3-3}}{C_{15}^3}$$

$$= 0{,}021$$

Pour C :

$$P(B) = P(X = 1)$$

$$= \frac{C_5^1 \times C_{15-5}^{3-1}}{C_{15}^3}$$

$$= 0{,}49$$

Fonction Génératrice

Définition :

Soit ***X*** une variable aléatoire à valeurs dans ***N***. On appelle **fonction génératrice** de ***X*** la série entière suivante :

$$G_X(t) = \sum_{n=0}^{\infty} P(X = n)t^n$$

Pour la loi **Hypergéométrique** :

$$G_X(t) = \sum_{k=0}^{n} P(X = k)t^k$$

Avec :

$$P(X = k) = \frac{C_A^k \times C_{N-A}^{n-k}}{C_N^n}$$

Théorème :

Si X et Y sont deux variables aléatoires à valeurs dans $\mathbb{N}$ indépendantes, alors, pour tout $t \in]-1,1[$, $G_{X+Y}(t) = G_X(t) \times G_Y(t)$

Simulation sous Python

Loi Hypergeometrique P(X=x) :

```python
def hypergeom(N, A, n, x):  # P( X = x )

    """
    N : taille de la population
    A : nombre total d'elements succes dans N
    n : nombre de tirages effectues e partir de N
    x : nombre d'elements succes dans notre tirage de n elements
    renvoie : La probabilite en X = x ---> P( X = x )
    """
    ACx = comb(A, x)
    NCx = comb(N - A, n - x)
    NCn = comb(N, n)

    return (ACx) * NCx / NCn
```

Simulation (Suite)

Loi Hypergeometrique Cumulée :

```python
def hypergeom_C(N, A, n, t, min_value=None):  # P( X <= x )

    """
    : param N : taille de la population
    : param A : nombre total d'elements succes dans N
    : param n : nombre de tirages effectues a partir de N
    : param t : nombre d'elements succes dans notre tirage de n elements jusqu'a t
    : renvoie : proba cumulee calcule jusqu'a t
    """
    if min_value:
        return np.sum([hypergeom(N, A, n, x) for x in range(min_value, t + 1)])

    return np.sum([hypergeom(N, A, n, x) for x in range(t + 1)])
```

Simulation (Suite)

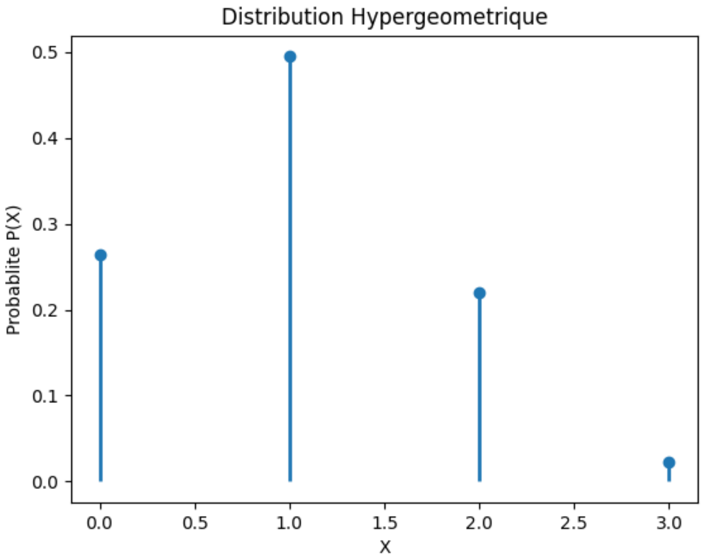
Distribution de loi Hypergeometrique:

```python
def hypergeom_plot(N, A, n):

    """
    : param N: taille de la population
    : param A: nombre total d'elements succes dans N
    : param n: nombre de tirages effectues a partir de N
    """

    x = np.arange(0, n + 1)
    y = [hypergeom(N, A, n, x) for x in range(n + 1)]
    plt.plot(x, y, "o")
    plt.vlines(x, 0, y, lw=2)
    plt.xlabel("X")
    plt.ylabel("Probablite P(X)")
    plt.title("Distribution Hypergeometrique")
    plt.show()
```

Output :

On prend le Cas de l’exemple précèdent.

```
hypergeom_plot(15, 5, 3);
```



Fin!