How to count a wolf population?

- Bellevue Marie Rivier, Val de Durance
- Researcher, Laboratories : Yves PAPAGEGAY, INRIA Sophia Antipolis
- Students : Cindy
  Chloé
  Cassandra
  Léa
  Lou-Anne
  Alexis
  Dimitri
  Hugo

Co-funded by the Erasmus+ Programme of the European Union

Problem / 1st Solution

- Problem :
- There is an area, like a forest. We would like to know how many wolves are in this place without counting them one by one.
- **How can we do ?**

- 1st Solution :

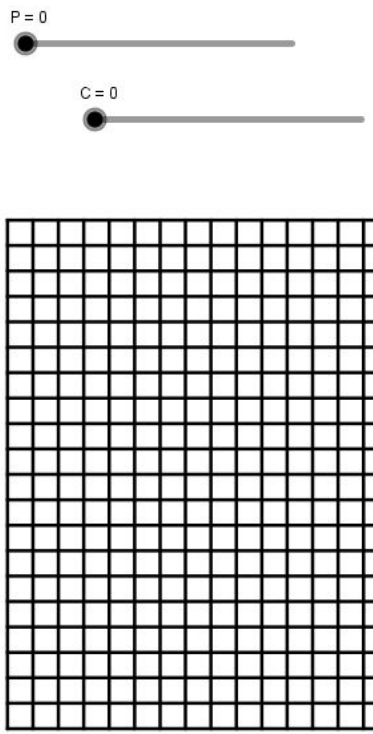
- Divide the area into multiple cells
- Choose a number of these cells
- Count the wolves in these cells
- Divide the number found by the number of cells you took
- Multiply this number by the total number of cells

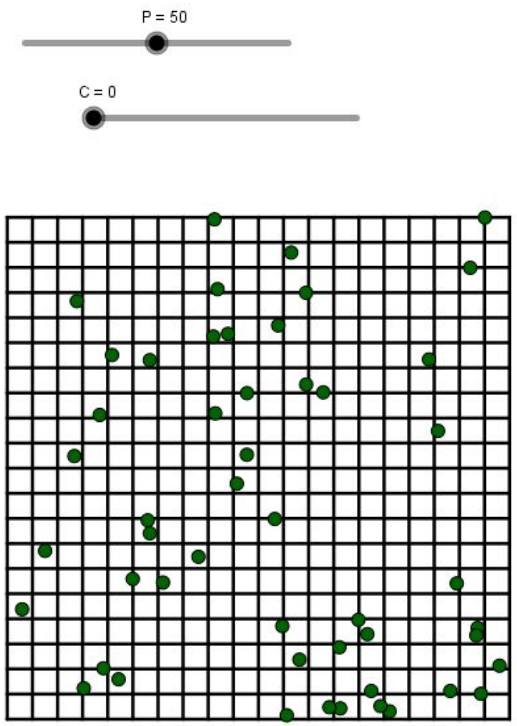
Estimating the abundance

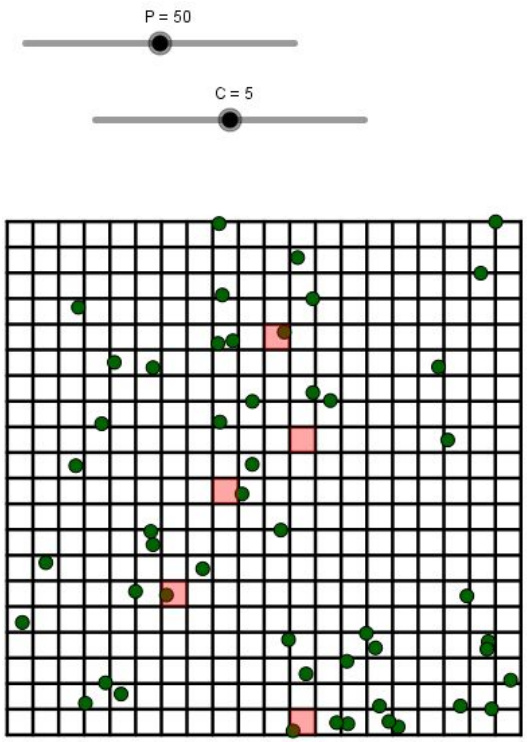
- P → Population
- C → Cells observed
- A → Total number of cells
- n → Wolves count in cells observed

A = 20 x 20 = 400

n = 3







Formula

The following equality is now obtained:

$$P = N/C * A$$

So, with: $n = 3$
$C = 5$
$A = 400$

$$P = 3/5 * 400 = 240$$

!!! We don't have a correct result because the number of cells observed is too small, it's just for example !!!

1st Program

(programme v2.py) × (Programme v3.py) ×

```python
import random as r
import numpy as np

y, x, z, Nbre_cases = int(input("How many lines in the array ?")), int(input("How many columns in the array ?")),int(input("How deep ?")), int(input("How many areas ?"))
Population_totale = r.randint(100,10000)
Population = np.zeros((y,x,z))

for a in range(Population_totale):
    Population[r.randint(0,y-1), r.randint(0,x-1), r.randint(0,z-1)] += 1

Population_estimée = (sum(np.random.choice(Population.ravel(), Nbre_cases))/Nbre_cases)*x*y*z
Population_total = sum(sum(sum(Population)))
print("\n Population : ",Population_total, "\n Estimated population : ", Population_estimée, "\n Error : ", round((abs(Population_total-Population_estimée)/Population_total)*100,2), " %")
```

Console 1/A ×

```
Python 3.8.5 (default, Sep  3 2020, 21:29:08) [MSC v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 7.22.0 -- An enhanced Interactive Python.

In [1]: runfile('D:/Documents/Maths en Jean/programme v2.py', wdir='D:/Documents/Maths en Jean')

How many lines in the array ?20

How many columns in the array ?20

How deep ?1

How many areas ?5

 Population :  9655.0
 Estimated population :  8320.0
 Error :  13.83  %

In [2]: |
```

2nd Program

(programme v2.py) × (Programme v3.py) ×

```python
import random as r
import numpy as np

y, x, z = int(input("How many lines in the array ?")), int(input("How many columns in the array ?")),int(input("How deep ?"))
Population_totale = r.randint(100,10000)

for i in range(1,51):
    error = 0
    for b in range(1,51):
        Population = np.zeros((y,x,z))

        for a in range(Population_totale):
            Population[r.randint(0,y-1), r.randint(0,x-1), r.randint(0,z-1)] += 1

        Population_total = sum(sum(sum(Population)))
        Population_estimée = (sum(np.random.choice(Population.ravel(), i))/i)*x*y*z
        error += (abs(Population_total-Population_estimée)/Population_total)*100

    print(f"\n Areas = {i}")

    print("\n Error : ", round(error/50,2), " %")
```

```
In [2]: runfile('D:/Documents/Maths en Jean/Programme v3.py', wdir='D:/Documents/Maths en Jean')
How many lines in the array ?20
How many columns in the array ?20
How deep ?1
 Areas = 1
 Error :  15.47  %
 Areas = 2
 Error :  12.0  %
 Areas = 3
 Error :  9.51  %
 Areas = 4
 Error :  7.64  %
 Areas = 5
 Error :  8.43  %
 Areas = 6
 Error :  6.97  %
 Areas = 7
 Error :  5.31  %
 Areas = 8
 Error :  5.65  %
 Areas = 9
 Error :  5.9  %
 Areas = 10
 Error :  6.62  %
 Areas = 11
 Error :  5.22  %
```

```
Areas = 10

Error :  6.62  %
```

2nd Solution

Method CMR (capture -marking –recapture)

- **Part 1 :**
  Modeling with rice

- **Part 2 :**
  Modeling with program

Part 1

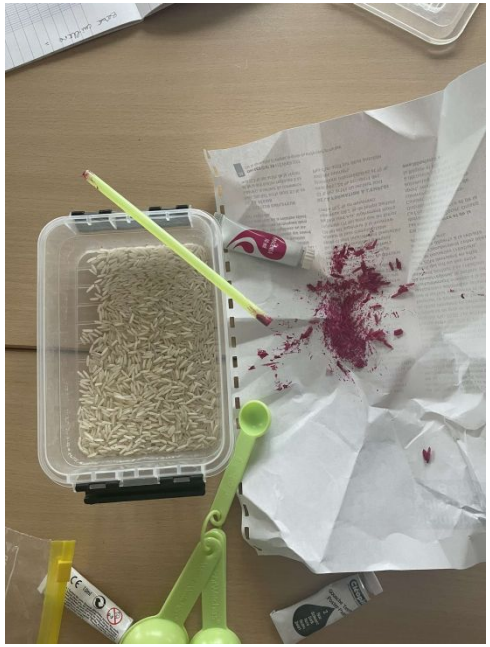
« Big sample »
= total population

= 1437 grains

Get a sample to color grains

2.5 ml measuring spoon

« Colored sample »
= marked wolves



= 134 grains

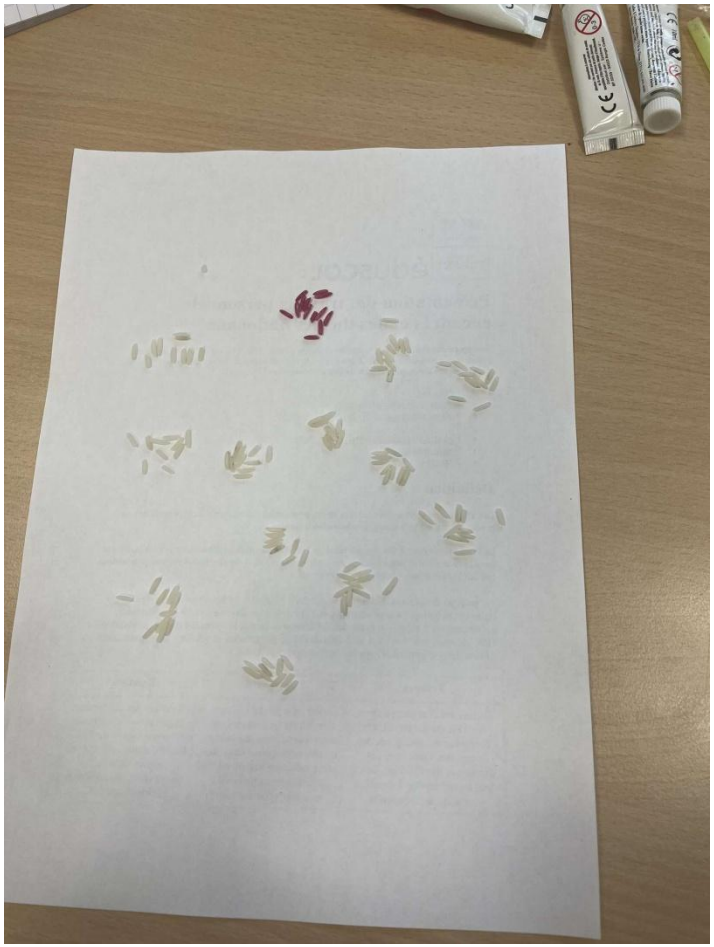
Putting pink grains in the big sample and mix

Experiment

Formula :

$$N = \frac{n1 \times n2}{r}$$

N = total population
n1 = colored sample (=134)
n2 = little sample
r = wolves marked in n2



30 times

Too large margin of error

Inequality of concentration

Formula :

$$P(|M_n - E(X)| \geq \delta) \leq \frac{V(X)}{n\,\delta^2}$$

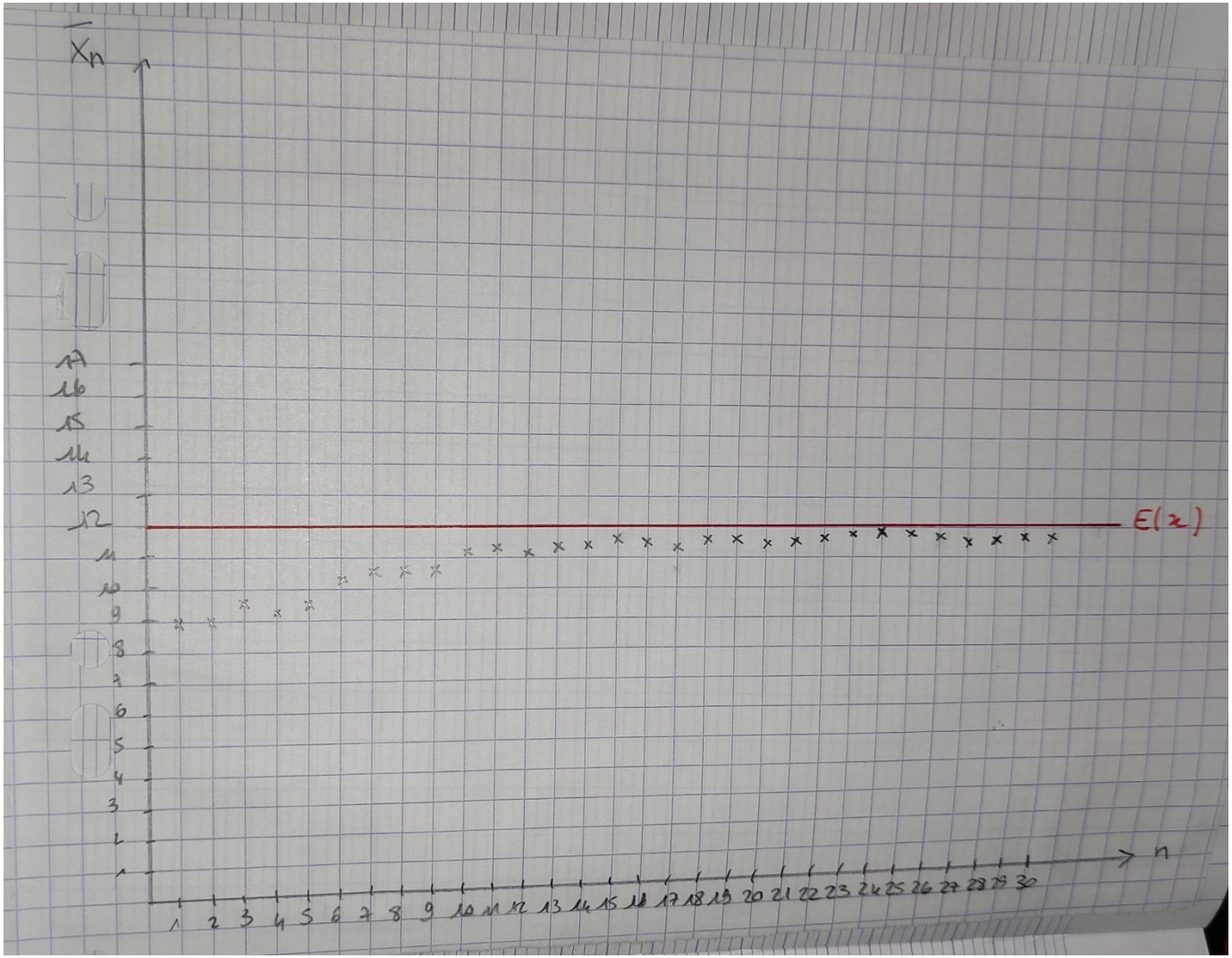
This is not a Bernouilli law

Comment :

The average of all values of n1 (colored rice) for the 30 little sample tend toward E(x) when n tend toward $+\infty$ (Graph 1).

Graph 1

Average of the 30 values of n1 ≃ 11.8 = 12



Confidence interval (1)

Frequency of occurrence (pink grains) = r / n2

<u>Example :</u>
In a little sample there are 122 white grains and 9 pink grains,
Frequency = 9/122 = 0.0737…

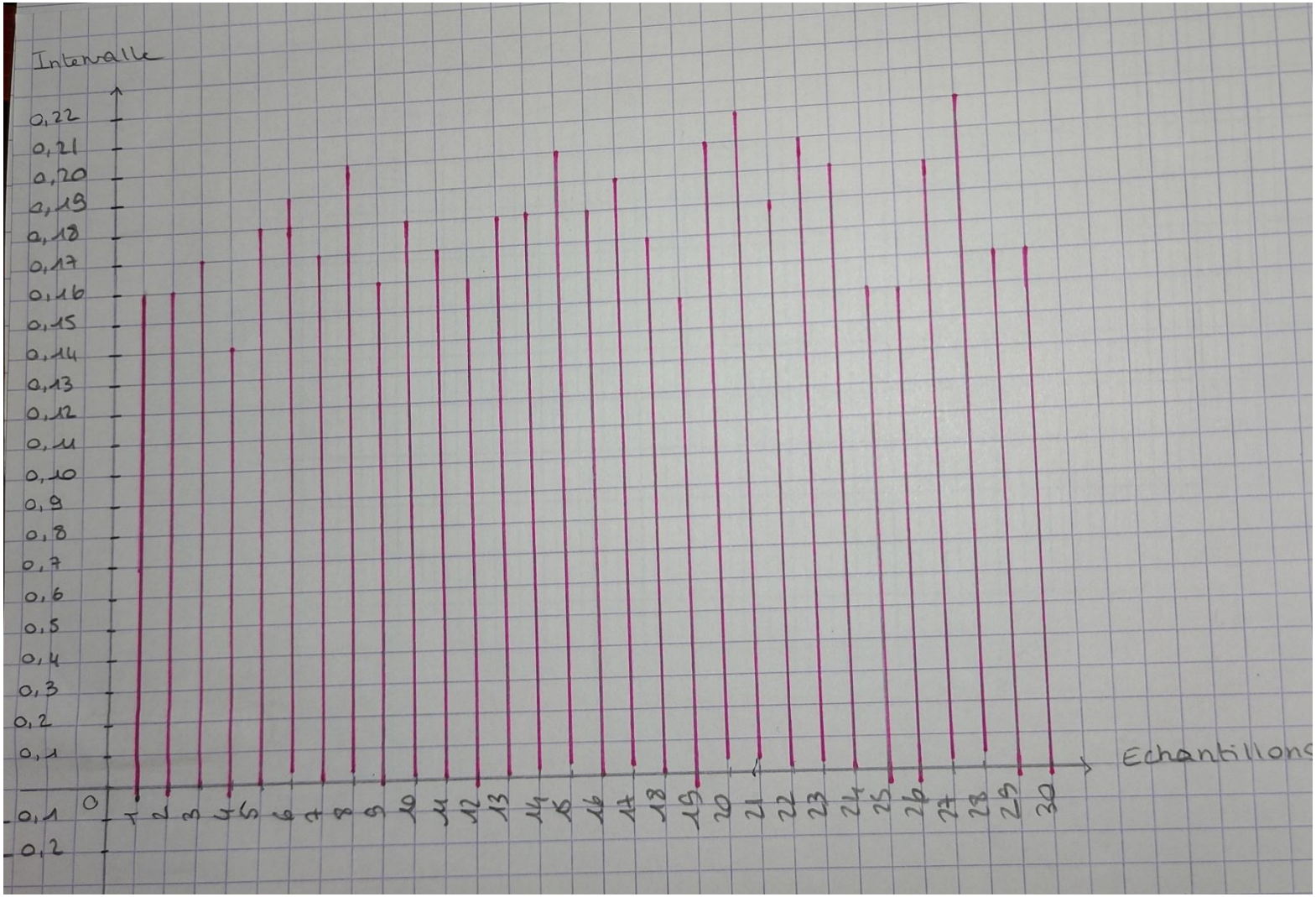
Theorem of Moivre-Laplace :

$$I = \left[f - \frac{1}{\sqrt{n}} ; f + \frac{1}{\sqrt{n}}\right]$$

I = confidence interval
f = frequency of occurrence
n = colored sample

Negative interval

Graph 2


Intervalle
0,22
0,21
0,20
0,19
0,18
0,17
0,16
0,15
0,14
0,13
0,12
0,11
0,10
0,9
0,8
0,7
0,6
0,5
0,4
0,3
0,2
0,1
0
-0,1
-0,2
Echantillons

Confidence interval (2)

Formula :

$$\left[p - 1.96\sqrt{\frac{p(1-p)}{n}} ; p + 1.96\sqrt{\frac{p(1-p)}{n}}\right]$$

M = marked wolves = 134

N = total population → ?

Let $p = 0,09$ (average of frequencies)

We admit that $\frac{X_1 + X_2 + \dots + X_n}{n} \to \mathcal{N}(0;1)$

Let $I = \left[p - 1,96\sqrt{\frac{p(1-p)}{n}} ; p + 1,96\sqrt{\frac{p(1-p)}{n}}\right]$

For $n = 50$,

$I = \left[0,09 - 1,96\sqrt{\frac{0,09(1-0,09)}{50}} ; 0,09 + 1,96\sqrt{\frac{0,09(1-0,09)}{50}}\right]$

$= [0,01 ; 0,17]$

We have $\frac{M}{N} \in I = [a ; b]$

$\Leftrightarrow 0,01 \leq \frac{134}{N} \leq 0,17$

$\Leftrightarrow \frac{134}{0,01} \geq N \geq \frac{134}{0,17}$

$\Leftrightarrow 13\,400 \geq N \geq 788$

Soit $N \in [788 ; 13\,400]$

↳ ($N = 1437$ in the experiment)

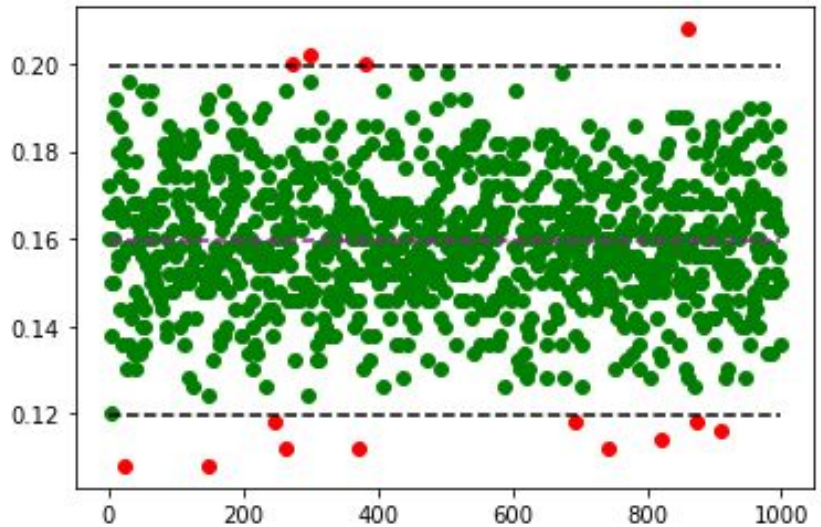
Comment :

If we increase the value of n,
the interval will become more refined

Example:
For n = 1000,
I = [0.07 ; 0.11]
And N ∈ [1218 ; 1914] → more precise

Program

```python
import random as r
from matplotlib import pyplot as plt
from math import sqrt

Population_totale = r.randint(400, 700)
Liste_Population = [0 for x in range(Population_totale)]
Vérif_liste = []
Nombre_animaux_marqués = int(input('How many animals to mark :'))

for x in range(Nombre_animaux_marqués):
    c = True

    while c == True:
        Random_animal = r.randint(0, (len(Liste_Population)-1))

        if Vérif_liste.count(Random_animal) == 0:
            Vérif_liste.append(Random_animal)
            Liste_Population[Random_animal] = 1
            c = False


Nombre_Observé = int(input('How many animals to capture :'))
Index_de_Base = r.randint(0, len(Liste_Population) - Nombre_Observé)

def moyenne(Liste_Population, Nombre_animaux_marqués, Nombre_Observé, Index_de_Base):
    M = []
    Liste_Resultat = []
    L = 0
    for a in range(10):
        marqués = sum(Liste_Population[Index_de_Base:(Index_de_Base + Nombre_Observé)])

        Index_de_Base = r.randint(0, (len(Liste_Population))-Nombre_Observé)

        Liste_Resultat.append((Nombre_animaux_marqués * Nombre_Observé) / marqués)

        M.append(marqués / Nombre_Observé)

        L += marqués

    J = L / 10

    return J


def estimation(Liste_Population, Nombre_animaux_marqués, Nombre_Observé, Index_de_Base):
    A = 0
    for i in range(1000):
        A += moyenne(Liste_Population, Nombre_animaux_marqués, Nombre_Observé, Index_de_Base)

    A = A / 1000

    B = (Nombre_animaux_marqués * Nombre_Observé) / A

    P = Nombre_animaux_marqués / B

    return P, B



p, B = estimation(Liste_Population, Nombre_animaux_marqués, Nombre_Observé, Index_de_Base)

mini, maxi = p-1/sqrt(B), p+1/sqrt(B)

for repet in range(10000):
    f = moyenne(Liste_Population, Nombre_animaux_marqués, Nombre_Observé, Index_de_Base) / Nombre_Observé
    if mini < f < maxi :
        plt.plot(repet, f, "go")
    else :
        plt.plot(repet, f, "ro")
plt.plot([0,repet],[p, p], "m--")
plt.plot([0,repet],[mini, mini], "k--")
plt.plot([0,repet],[maxi, maxi], "k--")
plt.show()

print(f'Estimated population : {B}')
print(f'Real population : {Population_totale}')
```




```
In [3]: runfile('D:/Documents/Maths en Jean/P

How many animals to mark :100

How many animals to capture :50
Estimated population : 495.64820872737334
Real population : 499
```

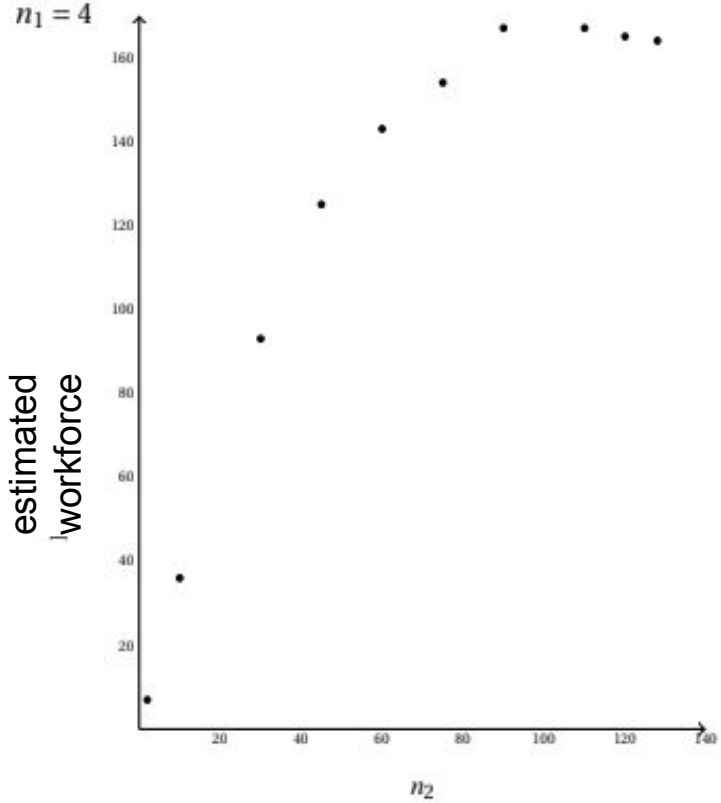
Capture N1 individuals in this population

Mark these N1, indiuals then release them in the population

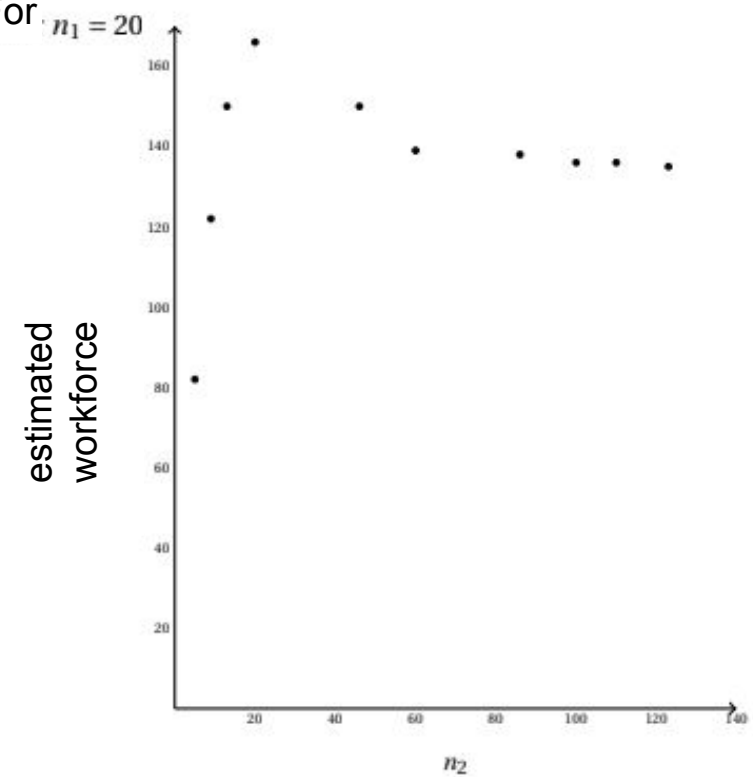
Capture N2 individuals in the population and count the number R of marked individuals

Part 2

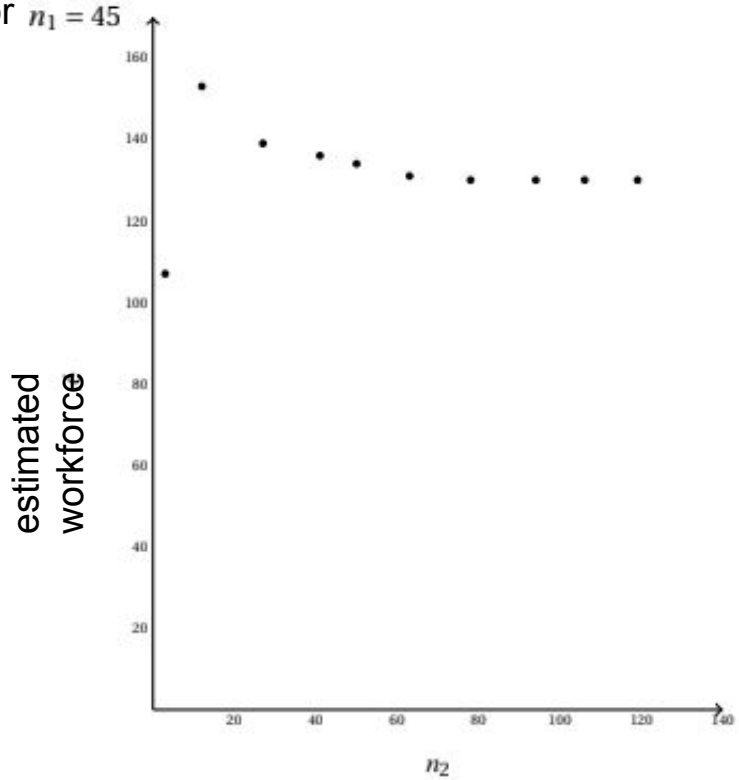
For $n_1 = 4$



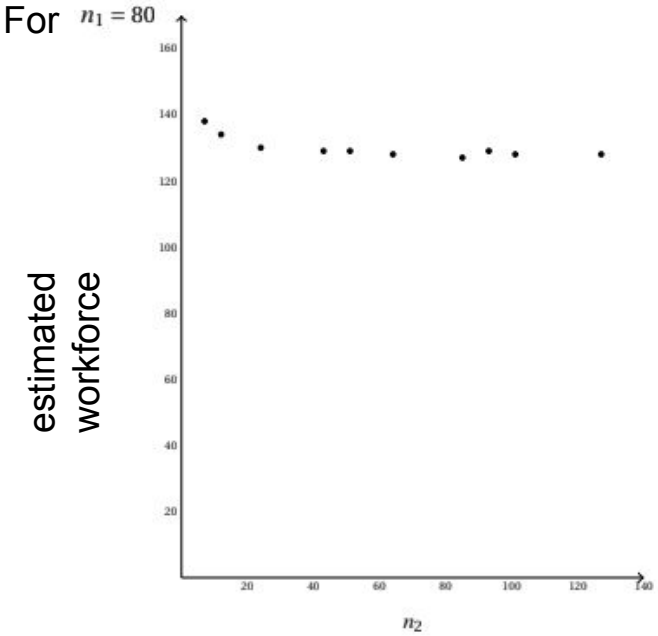
For $n_1 = 20$

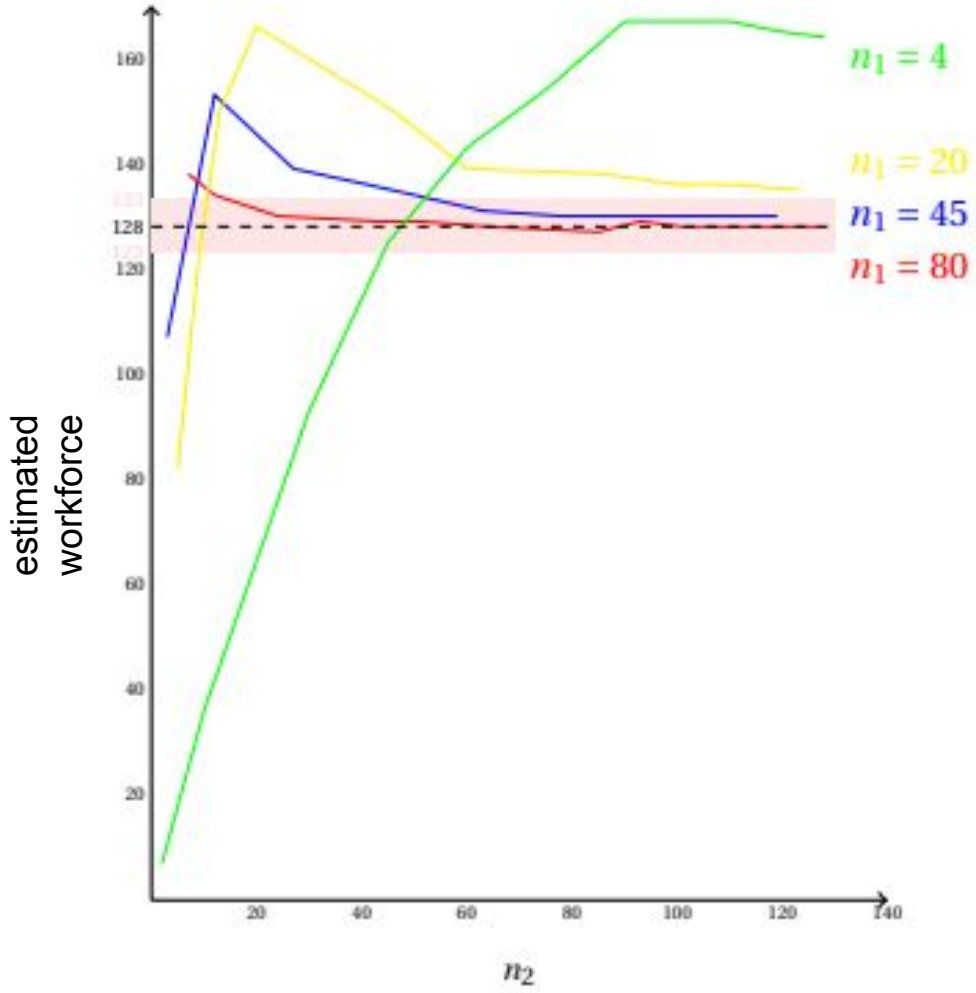


For $n_1 = 45$



For $n_1 = 80$




estimated workforce
160
140
128
120
100
80
60
40
20
20
40
60
80
100
120
140
n_2
$n_1 = 4$
$n_1 = 20$
$n_1 = 45$
$n_1 = 80$

THANK YOU FOR YOUR ATTENTION !

We are available to answer your questions

If you want to follow our work :
https://twinspace.etwinning.net/122026/home

Co-funded by the Erasmus+ Programme of the European Union