Chapitre 4 : TD de Statistiques sur l'échantillonnage

CORRECTION

I] Échantillonnage avec une calculatrice

Un sachet de *m&m's* est composé de bonbons de six couleurs également réparties entre : bleu, vert, jaune, rouge, orange, marron. Pablo ouvre un sachet de *m&m's*, et constate avec horreur que sur les 16 bonbons qu'il contient, aucun bonbon n'est bleu (sa couleur préférée) !!

a) Compléter le tableau qui donne les fréquences expérimentales (dans le sachet de Pablo) et théorique (en théorie, les couleurs sont également réparties).

Pour compléter le tableau, on calcule les rapports observés entre l'effectif d'une couleur et l'effectif total (fréquences exprimées en % en multipliant par 100 et en arrondissant à 0,01 près), par exemple 25,00% correspond au rapport 4÷16.

Couleur (code)	Effectifs	Fréquences expérimentales	Fréquences théoriques
Bleu (1)	0	0,00%	16,67%
Vert (2)	2	12,50%	16,67%
Jaune (3)	3	18,75%	16,67%
Rouge (4)	4	25,00%	16,67%
Orange (5)	5	31,25%	16,67%
Marron (6)	2	12,50%	16,67%

b) On décide de **simuler la production d'un sachet** de *m&m's* en programmant l'algorithme suivant :

- lire N (ici on peut directement écrire N=16, cela évite d'avoir à le saisir à chaque fois)
- X=0 (ce sera notre compteur de *m&m's* bleus)
- pour I allant de 1 à N : A=Nombre aléatoire de [0;1[; Si A<0,16666667 alors X=X+1
- affichage de X (nombre de *m&m's* bleus)
- affichage de Y=X×100÷N (calcul de la fréquence en %)

Pour générer un nombre pseudo-aléatoire de l'intervalle [0 ; 1[: **Ran#** (*Casio*), **Rand** ou **EntAléa** (*TI*)

Sur *Numworks*, dans l'attente d'une vraie fonction **random**, les instructions sont *g=rand*(*g*) ; *a=g*/32768 (la graine *g* est initialisé par *g*=xxxxxx) avec : *def rand*(*g*) : *return* (*g**1103515245+12345)/65536%32768

Il peut y avoir des variantes selon les modèles de calculatrices.

On peut écrire un programme très simple pour tester une fonction. Avec TI par exemple, si on veut être sûr de ce que fait *EntAleat*, on peut écrire le programme qui contient une seule ligne : Disp *EntAleat.* Cela devrait à chaque fois afficher un nombre différent de [0 ; 1[. Si ce n'est pas le cas ou s'il y a une erreur de syntaxe, il faut revenir au catalogue ou, mieux, au manuel de la calculatrice.

Les possibilités de programmation sont accentuées avec Python : depuis la version 1.2, on peut écrire des fonctions dans un même programme. Ici, j'ai utilisé la fonction *rand* déjà mise au point (pour le cours et pour le DM) et j'ai ajouté la fonction *simule* qui utilise rand. Lorsqu'on exécute le programme Python qui contient ces fonctions, on peut accéder aux fonctions avec la touche *var.* Je lance alors *simule*(123456). Ce qui est amusant avec ce GNPA, c'est qu'on obtient la même série de nombres aléatoires si on part de la même graine : j'obtiens x=2 et y=12,5 et tous ceux qui lanceront ce programme avec la même graine obtiendront la même chose. Avec une autre graine, par exemple 234567, j'obtiens x=1 et y=6,25. Pour les autres valeurs, je ne vous dit pas les graines choisies et nous obtenons (en principe) des valeurs différentes.

Casio	TI	Numworks

?→N
0→X
For 1→I TO N
Ran#→A
IF A<0,16666667
THEN X+1→ X
IfEnd
Next
X ◢
X×100÷N→ Y
Y ◢
	input N
	0→X
	For *(I,1,N)*
	EntAleat→A
	IF A<0,16666667
	THEN X+1→ X
	End
	End
	Disp X
	X×100÷N→ Y
	Disp Y
		def *rand*(g) :
		  return (g*1103515245+12345)/65536%32768
		
		def *simule*(g) :
		  n,x=16,0
		  for i in range(n) :
		  g=*rand*(g)
		  a=g/32768
		  if a<0,16666667 :
		  x=x+1
		  print("x= ",x,"y=",x*100/n)

>>Simuler la production de 10 sachets contenant 16 *m&m's* en utilisant 10 fois de suite ce programme.

N° du sachet	1	2	3	4	5	6	7	8	9	10
X=Nombre de *m&m's* rouges	2	1	3	2	5	1	3	2	2	4
Y=Fréquence de *m&m's* rouges	12,5	6,25	18,75	12,5	31,25	6,25	18,75	12,5	12,5	25

Combien de sachets contiennent 0 *m&m's* rouge?
Aucun, mais nous avons mis nos 10 premiers résultats, ils pourraient être différents...
Quelle est la fréquence moyenne des *m&m's* rouges? Nous avons obtenu 2+1+3+2+5+1+3+2+2+4=25 *m&m's* bleus sur 10×16=160 *m&m's*. Cela fait une fréquence moyenne de 25÷160=0,15625, soit 15,625%. Pas très loin de la fréquence théorique réelle : 16,6667%, mais nos résultats pourraient être différents...
Quel est leur nombre moyen par sachet? Nous avons obtenu 25 *m&m's* bleus sur 10 sachets. Cela fait un nombre moyen de 25÷10=2,5 par sachet. Pas très loin du nombre théorique réel : 16÷6=2,667 par sachet, mais nos résultats pourraient être différents...

c) On décide de **simuler la production de M=100 sachets** de ***m&m's***, sans avoir à relancer M fois le programme. Pour cela on *améliore* le programme précédent qui restera utilisable (il suffira d'entrer M=1).

- Ajouter une boucle « Pour J allant de 1 à M » qui contiendra l'autre boucle (Pour I allant de 1 à N)
- Ajouter un compteur Y, pour comptabiliser les sachets contenant 0 ***m&m's*** bleu.
- Ajouter un compteur Z qui comptabilise le nombre total de ***m&m's*** bleus pour les M sachets

Voici le nouvel algorithme qui réalise ce nouvel objectif :

- Lire N (ici N=16), Lire M (ici M=100)
- Y=0 (compteur des sachets contenant 0 *m&m's* bleu), Z=0 (compteur de *m&m's* bleus au total)
- Pour J allant de 1 à M
  X=0 (compteur de *m&m's* bleus par sachet)
  Pour I allant de 1 à N :
  A=Nombre aléatoire de [0;1[
  Si A<0,16666667 alors X=X+1
  Z=Z+X
  Si X=0 alors Y=Y+1
- affichage de Y (nombre de sachets contenant zéro *m&m's* bleu) ;
- affichage de Z (nombre de *m&m's* bleus au total)
- affichage de Z=Z×100÷(M×N) (la fréquence en % des *m&m's* bleus sur les M×N bonbons)

Programmons cet algorithme :

Casio	TI	Numworks

?→N
?→M
0→Y
0→Z
For 1→J TO M
0→X
For 1→I TO N
Ran#→A
IF A<0,16666667
THEN X+1→ X
IfEnd
Next
Z+X→Z
IF X=0
THEN Y+1→ Y
IfEnd
Next
Y ◢
Z×100÷N÷M→ Z
Z ◢
	input N
	input M
	0→Y
	0→Z
	For (J,1,M)
	0→X
	For (I,1,N)
	EntAleat→A
	IF A<0,16666667
	THEN X+1→ X
	End
	End
	Z+X→Z
	IF X=0
	THEN Y+1→ Y
	End
	End
	Disp Y
	Z×100÷N÷M→ Z
	Disp Z
		def rand(g) :
		  return (g*1103515245+12345)/65536%32768
		
		def simule(g) :
		  n,m,y,z=16,100,0,0
		  for j in range(m) :
		  x=0
		  for i in range(n) :
		  g=rand(g)
		  a=g/32768
		  if a<0,16666667 : x+=1
		  z+=x
		  if x=0 : y+=1
		  print("y= ",y,"z=",z,"freq z=",z*100/n/m)

>>Simuler 10 fois la production d'une série de 100 sachets contenant 16 ***m&m's*** en prenant M=100.

N° de la série de 100 sachets	1	2	3	4	5	6	7	8	9	10
Sachets contenant 0 ***m&m's*** bleu (Y)	4	7	4	4	8	3	1	5	3	5
Nombre ttotal de ***m&m's*** bleus (Z)	272	270	259	288	263	303	284	278	273	251
Fréquence de ***m&m's*** bleus (Z×100÷(MN))	17,0%	16,9%	16,2%	18,0%	16,4%	18,9%	17,8%	17,4%	17,1%	15,7%

Remarques : j'introduis dans ces programmes quelques notations commodes en Python comme par exemple l'**affectation multiple** *n,m,y,z*=16,100,0,0 qui remplace avantageusement 4 lignes pour *n*=16, *m*=100, *y*=0 et *z*=0. Certains d'entre vous se demanderont que fait une instruction comme *a,b*=*b,a*+*b*. Est-ce que le 2ème *a* est affecté par la 1ère affectation (de *b* dans *a*) ? Heureusement non, ce qui permet de calculer simplement les nombres de la suite de Fibonacci (si *a,b*=1,2 alors, après cette instruction on aura *a,b*=2,1+2 (et non 2+2), soit *a,b*=2,3 et, si on continue *a,b*=3,5 etc.).
Une autre notation intéressante est le **raccourci** y+=1 pour y=y+1 ou bien z+=x pour z=z+x. On peut aussi, lorsque cela se présente, écrire f*=n pour f=f*n ou q/=r pour q=q/r ou d-=c pour d=d-c. Cela paraît dérisoire mais cela fait gagner du temps à la longue.
Une dernière remarque concernant la graine g du GNPA : si on est lassé de devoir choisir soi-même une nouvelle graine à chaque tirage (sous peine d'obtenir les mêmes résultats lorsqu'on utilise la même), il faut déclarer g comme **variable globale**. On écrit pour cela, par exemple g=123456 dans le programme Python (en dehors des fonctions qui l'utilisent. Et, dans la fonction simule, on écrit *global g*. Tout simplement. De cette façon, on autorise la fonction *simule* à modifier la variable *g* qui restera modifiée pour les prochaines utilisations (tant qu'on ne ré-exécute pas le programme). Cela paraît compliqué car cela montre que les variables sont **locales** (elles appartiennent aux fonctions qui les définissent). Mais il peut y avoir des variables globales, utilisées sans précautions particulières tant qu'on ne les modifie pas. Ici, on modifie *g*, c'est pour cela qu'il faut écrire global *g*. Voici le programme Python avec la variable globale *g* :

```
def rand(n) :
  return (n*1103515245+12345)/65536%32768

g=123456

def simule() :
  n,m,y,z=16,100,0,0
  global g
  for j in range(m) :
    x=0
    for i in range(n) :
      g=rand(g)
      a=g/32768
      if a<0,16666667 : x+=1
    z+=x
    if x=0 : y+=1
  print("y= ",y,"z=",z,"freq z=",z*100/n/m)
```

Je renomme la graine *n* (au lieu de *g*) pour ne pas introduire de risque de conflit entre variable locale et globale

la variable globale *g* est définie à l'extérieur des fonctions qui l'utilisent et prend cette valeur initiale à l'exécution du programme

la fonction *simule* n'a plus d'argument (il faut enlever le *g* qui était contenu avant entre les parenthèses) ce qui fait qu'on la lancera et la relancera en écrivant juste *simule*() dans la console. La graine *g* ne sera pas réinitialisée à 123456 à chaque fois car elle continue d'exister en dehors de la fonction *simule*.

Dans la fonction simule, comme on modifie la valeur d'une variable globale, il faut la déclarer comme tel. Si on oublie l'instruction *global g*, il y aura une erreur qui dira que l'on a tenté d'utiliser une variable locale avant de l'avoir référencé (*rand*(*g*) n'existe pas si *g* n'a pas été initialisée)

Au vue de ces résultats que pouvez-vous conclure sur les sachets contenant 0 *m&m's* bleu? Sont-ils exceptionnels ou fréquents? S'agit-il d'une tricherie, d'un vol ou d'un phénomène dominé par le hasard?
Sur 100 sachets contenant 16 *m&m's*, il y a toujours quelques sachets ne contenant aucun *m&m's* bleu. Sur nos 10 séries de 100, il y en a entre 1 et 8, avec une moyenne de 4,4 sachets sans *m&m's* bleu pour 100 sachets, car (4+7+4+4+8+3+1+5+3+5)÷10=44÷10=4,4. Pablo est donc tombé sur un de ces sachets (il y en a environ 4,4%). Ils ne sont ni exceptionnels ni fréquents (1 pour 23 environ), on pourrait dire qu'ils sont assez rares.
On peut penser qu'il s'agit d'un phénomène dominé entièrement par le hasard. On ne pourrait pas comprendre l'intérêt de fabriquer des sachets ne contenant aucun *m&m's* bleu (il n'y a pas de mobile, si encore ils coûtaient plus cher à fabriquer on comprendrait mieux). La chaîne de fabrication doit fabriquer autant de bonbons de chaque couleur, ceux-ci sont mélangés et mis en sachet par des machines qui ne tiennent pas compte des couleurs. Il s'agit, en principe donc, du pur hasard.

d) Déterminer l'intervalle de fluctuation au seuil de 95% dans lequel doit se trouver la fréquence des *m&m's* bleu dans un sachet. Est-on dans le domaine d'application de la formule ?
Si on considère l'ensemble de nos 10 séries de 100 sachets de 16 bonbons, on est certainement dans le domaine d'application de la formule. La fréquence théorique étant de 1/6, soit 0,16666667 environ, et l'échantillon observé contenant 10×100×16=16 000 bonbons, l'intervalle de fluctuation au seuil de 95% est

$$\left[\ \frac{1}{6}-\frac{1}{\sqrt{16000}}\approx 0{,}15876\ \ ;\ \frac{1}{6}+\frac{1}{\sqrt{16000}}\approx 0{,}17457\ \right]$$

La fréquence expérimentale d'obtention des bonbons bleus est, en moyenne pour nos 10 fois 100 sachets de 16 bonbons, de 17,14% (j'ai additionné mes 10 fréquences puis divisé par 10 pour en obtenir la moyenne),

soit 0,1714.

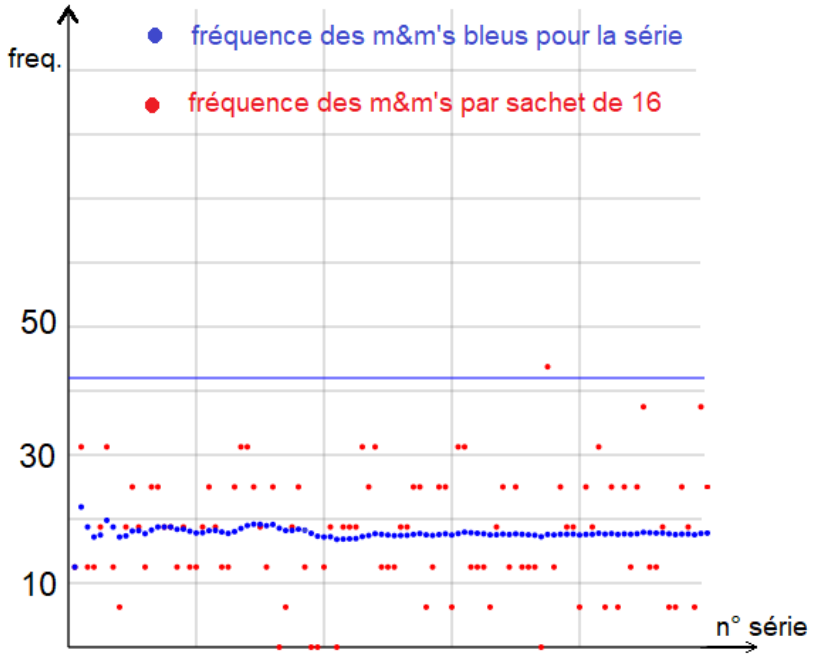
17,00%	16,90%	16,20%	18,00%	16,40%	18,90%	17,80%	17,40%	17,10%	15,70%	17,14%

Est-on dans l'IF ? Certes, oui $0{,}1714 \in [\ 0{,}15876\ ;\ 0{,}17457\ ]$. C'est encore heureux, puisque nous avons vraiment procédé par le seul effet du hasard. Cela aurait été étonnant d'être en dehors de l'IF (même si cela arrive, en principe dans 5% des cas), mais même alors, on n'aurait pas été très loin des bornes de l'IF.

La question, en fait, n'était pas celle-là : lorsque Pablo ouvre son paquet, il n'y a pas de bonbon bleu, la fréquence expérimentale d'obtention des bonbons bleus étant de 0/16=0 dans son paquet. L'IF pour cet échantillon de 16 bonbons n'est pas celui que l'on vient de calculer qui portait sur les 16000 bonbons générés par nos diverses simulations. L'IF qui répond à la question posée est un intervalle beaucoup plus étendu : $[\ \frac{1}{6}-\frac{1}{\sqrt{16}} \approx -0{,}0833\ ;\ \frac{1}{6}+\frac{1}{\sqrt{16}} \approx 0{,}416\ ]$. On peut prendre 0 comme valeur inférieure car une fréquence ne peut pas être négative.

Donc l'IF=$[\ 0\ ;\ 0{,}416\ ]$, et $0 \in IF$. La fréquence obtenue est compatible avec le tirage par le seul fait du hasard.

Est-on dans le domaine d'application de la formule ? Pas tout-à-fait ici car la taille de l'échantillon est trop petit (16 au lieu de 20 minimum) et la fréquence aussi est trop faible (0,16 au lieu de 0,2 minimum). Néanmoins, comme on n'est pas très loin du domaine d'application, on peut dire que le tirage est compatible avec l'hypothèse que seul le hasard prévaut dans sa constitution. En remplissant nos paquets avec des bonbons dont la couleur est choisie par le seul fait du hasard, on a vu que dans 1 paquet sur 23 environ, on obtient 0 bonbon bleu.



L'image ci-contre donne une idée de ce qui peut se passer pour 100 tirages (chaque point rouge est un tirage). On voit ici 5 paquets sur 100 (sur l'axe des abscisses) qui ne contiennent aucun bonbons bleus. Les points bleus montrent comment évolue la fréquence moyenne des bonbons bleus dans la série des 100 sachets : elle se stabilise de plus en plus sur une valeur proche de la fréquence expérimentale (0,1666667).

NB : on peut s'amuser à augmenter le nombre de bonbons par sachet pour voir à partir de quelle valeur l'IF ne contient plus 0. C'est à partir de n=37 bonbons dans un sachet que l'on doit commencer à s'étonner qu'il n'y ait pas de bonbons bleus dans un sachet. Car alors la fréquence 0 n'appartient pas à l'IF. Mais cela n'empêche que dans 5% des cas ce sera tout de même le cas...

n	16	20	25	30	35	36	37	38	39	40
$\frac{1}{6}-\frac{1}{\sqrt{n}}$	-0,0833	-0,0569	-0,0333	-0,0159	-0,0024	0,0000	0,0023	0,0044	0,0065	0,0086

e) <u>Prolongement</u> (la question n'était pas sur la feuille : elle n'est donc pas au programme du contrôle)

On décide de simuler la production de 100 sachets de *m&m's* à l'aide du tableur : ouvrir une page « classeur » du tableur « calc » d'*OpenOffice*. Pour simuler le tirage aléatoire d'une couleur codée par un chiffre allant de 1 à 6, on utilise les fonctions :

- ALEA() qui génère un nombre pseudo-aléatoire de l'intervalle [0 ; 1[
- ENT(*xy,zt*) qui donne la partie entière *xy* d'un nombre décimal *xy,zt*
- NB.SI(*plage;x*) qui calcule le nombre d'élément de la *plage* donnée contenant la valeur *x*

Par exemple, l'instruction =ENT(ALEA()*6)+1 permet de générer un nombre entier aléatoire allant de 1 à 6 (on peut aussi utiliser l'instruction =ALEA.ENTRE.BORNES(1;6)). L'instruction =NB.SI(B1:B20;1) permet de compter toutes les occurrences du nombre 1 (codant le *m&m's* bleu) dans la plage qui va de la cellule B1 à la cellule B20.

>>Simuler la production de 100 sachets contenant 16 *m&m's*.

N°	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Total PILE	f_exp
1				BLEU	BLEU									BLEU		BLEU	4	25,00%
2					BLEU			BLEU							BLEU		3	18,75%
3						BLEU											1	6,25%
4			BLEU	BLEU								BLEU					3	18,75%
5						BLEU									BLEU		2	12,50%
6														BLEU			1	6,25%
7														BLEU	BLEU		2	12,50%
8		BLEU					BLEU				BLEU	BLEU			BLEU		5	31,25%
9			BLEU									BLEU		BLEU	BLEU		4	25,00%
10	BLEU					BLEU		BLEU									3	18,75%

Voici le début de notre tableau qui donne une simulation de 10 sachets de 16 bonbons.
Pour afficher la couleur on a tapé : =SI(ALEA.ENTRE.BORNES(1;6)=1;"BLEU";"")

Pour faire la somme des bonbons bleus à droite, on a tapé : =NB.SI(B2:Q2;"BLEU")
En bas du tableau (après avoir étendu les formules des 19 colonnes jusqu'à N°=100 lignes), on a obtenu la somme des effectifs en tapant : =SOMME(R2:R101)
On a obtenu la fréquence moyenne en divisant par 16×100. La moyenne peut être obtenue en divisant par 100, sinon, le tableur donne ce résultat en faisant : =MOYENNE(R2:R101)

Pour l'écart-type, il suffit de faire :=ECARTYPEP(R2:R101)

Pour les valeurs extrêmes de la fréquence des bonbons bleus on tape : =MIN(S2:S101) et =MAX(S2:S101)

98											BLEU		BLEU				2	12,50%
99	BLEU			BLEU		BLEU		BLEU					BLEU				5	31,25%
100		BLEU						BLEU		BLEU					BLEU		4	25,00%

Total 100 échantillons	271
fréquence moyenne	0,16938
moyenne effectif	2,71
Ecart-type	1,52
f min	0,000
f max	0,375

Il faut noter que ces valeurs calculées, sur nos 100 échantillons aléatoires, vont varier d'une personne à une autre. On peut retirer (dans le sens de recommencer un tirage) les couleurs avec Ctrl+Maj+F9, on obtient alors d'autres résultats :

II] Une autre situation d'échantillonnage

Avec le même algorithme, en modifiant N, M et la fréquence théorique, étudier cette autre situation :
Avec le même algorithme, en modifiant N, M et la fréquence théorique, étudier cette autre situation :
Dans la classe de 2de4 il y a 18 garçons et 22 filles. Est-ce le résultat du hasard ou bien est-ce qu'il y a plus de filles en seconde au Lycée Henri IV? Pour tester cette hypothèse, on part d'une *fréquence théorique* de 50% de filles dans une classe d'âge et on génère des groupes de 5 classes de 40 élèves.
Faire les adaptations nécessaire et comptabiliser les classes d'au moins 22 filles. Discuter les résultats.
Approche expérimentale :

N° de la classe de 40 élèves	1	2	3	4	5	6	7	8	9	10
Nombre de classes contenant au moins 22 filles par groupe de 5 classes	3	1	1	1	1	1	1	1	2	1
Nombre de filles par groupe de 5 classes	105	105	99	94	95	101	98	93	98	95
Maximum du nombre de filles par classe	23	23	24	23	23	26	23	23	24	24
Fréquence des filles dans les classes	52,5%	52,5%	49,5%	47,0%	47,5%	50,5%	49,0%	46,5%	49,0%	47,5%

On génère 10 fois 5 classes de 40 élèves (M=5, N=40).
La mise au point du programme pour faire ce décompte exige, en plus de ces initialisations, deux petites modifications des tests :

- à la place de l'instruction « Si X=0 alors Y=Y+1 » on écrit « Si X ≥ 22 alors Y=Y+1 »
- à la place de l'instruction « Si A<0,16666667 alors X=X+1 » on écrit « Si A<0,5 alors X=X+1 »

Remarques :
Nous avons ajouté dans le programme un compteur pour enregistrer le nombre maximum de filles par classes : initialisé avec *maxi*=0 et renseigné après le « si x ≥ 22 alors y+=1 » avec un « si x > maxi alors maxi=x ». Par ailleurs, la condition x ≥ 22 se tape en Python x >= 22

Combien y a t-il, parmi les 50 classes générées, de classes contenant au moins 22 filles ?

On en a trouvé 13 sur 50, avec un maximum de 3 par groupe de 5 classes, soit 26% (un peu plus de 1 sur 4). Ces classes sont donc assez fréquentes et la notre en est un digne exemple.
Combien y a t-il, parmi les 10 groupes de 5 classes générées, de classes contenant au moins 22 filles ?
Dans les 10 groupes, il y a au moins une classe contenant au moins 22 filles.
Que penser de l'hypothèse testée ?
Il semble qu'un nombre égal de filles et de garçon entraîne assez souvent des classes ayant au moins 22 filles. On peut donc encore ici penser qu'il s'agit d'un phénomène dominé entièrement par le hasard avec une équiprobabilité des filles et des garçons.

Approche théorique :
Déterminons l'intervalle de fluctuation au seuil de 95% dans lequel doit se trouver la fréquence des filles dans une classe. Dans un échantillon de 40 individus, avec une fréquence théorique de 0,5, l'intervalle de fluctuation pour la fréquence des filles dans une classe est [0,3419 ; 0,6581] (nous utilisons la formule du cours : $f \pm \frac{1}{\sqrt{n}}$).
Que penser de l'hypothèse testée?
Dans notre échantillon (notre classe), la fréquence des filles est de $\frac{22}{40} \approx 0{,}55$, soit une fréquence compatible avec l'hypothèse d'équiprobabilité des filles et des garçons, au seuil de 5% (en prenant le risque de se tromper 5 fois sur 100). Nous ne devons donc pas chercher à expliquer ce nombre supérieur de filles par un autre facteur que l'effet du hasard qui suffit, à lui seul, à expliquer ce très léger surnombre.