Modèles d'urnes simulés avec MathsOntologie

La République de Venise était, lorsque Galilée et Pascal inventaient le calcul des probabilités, l'un des pays les plus puissants du monde. Mais on n'y faisait pas de mathématiques (on y faisait de l'argent, ce n'est pas pareil). Alors qu'est-ce qu'elle vient faire dans cette introduction, la Sérinissime[1] ? C'est qu'elle est intéressante à étudier du point de vue constitutionnel : Les doges étaient choisis par cooptation au sein d'un concile dont les membres étaient élus par des citoyens ***tirés au sort***[2] : On mettait dans un grand vase, autant de billes de métal que de candidats, toutes les billes étant indiscernables au toucher. Un enfant choisissait des billes ***au hasard*** dans le vase, et les distribuait aux candidats. N'avaient le droit de vote, que les candidats ayant reçu une bille en or, les autres n'avaient que des billes de cuivre.

Jacques Bernoulli, lorsqu'il a écrit sur les probabilités, a donc choisi pour exemple un modèle inspiré par les élections de doges vénitiens : Le vase, appelé ***urna***, et traduit dans cet article par ***urne***, contenant des boules de couleurs différentes, ou portant des numéros différents, d'où on extrait une ou plusieurs boules au hasard. Cet article explore, à l'aide de l'outil *MathsOntologie*, l'univers surprenamment vaste que Bernoulli a ouvert avec ses urnes au contenu caché. L'article est articulé en quatre parties, correspondant au niveau (de la Troisième à la Terminale) où les activités peuvent être explorées.

I/ En Troisième

1) Principe

Condorcet, s'inspirant du travail fondateur de Bernoulli, étudiait lui aussi des tirages au sort de boules noires et blanches. Mais n'ayant pas eu de vase à sa disposition, il mettait ses boules de billard dans un sac opaque :

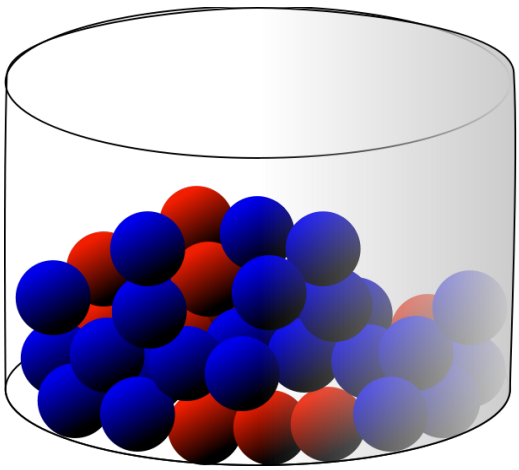
> *Supposons un sac dans lequel je sache qu'il existe quatre-vingt-dix boules blanches & dix noires, & qu'on me demande quelle est la probabilité d'en tirer une boule blanche ; ou que la boule étant déjà tirée, mais couverte d'un voile, on me demande quelle est la probabilité que l'on a tiré une boule blanche…*

(*extrait de « Essai sur l'application de l'analyse à la probabilité des décisions », 1785*)

1 Comme l'appelaient ses citoyens, fiers de leur version de la démocratie

2 Cette pratique rappelle la commune de Gênes dont les conseillers municipaux étaient tirés au sort parmi les citoyens, et, au-delà, les Saturnales romaines où le roi de la fête (roi d'un jour) était désigné par le hasard : était roi d'un jour celui qui mangeait une graine de haricot (« fève ») dans une expérience aléatoire s'apparentant plus au jeu de la roulette (une seule boule dans une urne atteinte au hasard). Nous fêtons tous les Saturnales chaque 6 janvier, la fève étant maintenant en plastique...

Pour mettre un peu de couleur, on va remplacer les boules noires et blanches par des boules rouges et bleues ; on va bien mettre les boules (ou autres objets) dans un sac[3], mais, comme c'est plus facile de dessiner un cylindre qu'un sac informe, on va représenter les urnes par des cylindres :



Ci-dessus, on a mis des boules de deux couleurs dans une boîte, et on sait que la boîte contient 70 % de boules bleues. On tire une seule boule au hasard de cette urne, et on demande la probabilité qu'elle soit rouge. La situation peut être déclinée à l'envi :

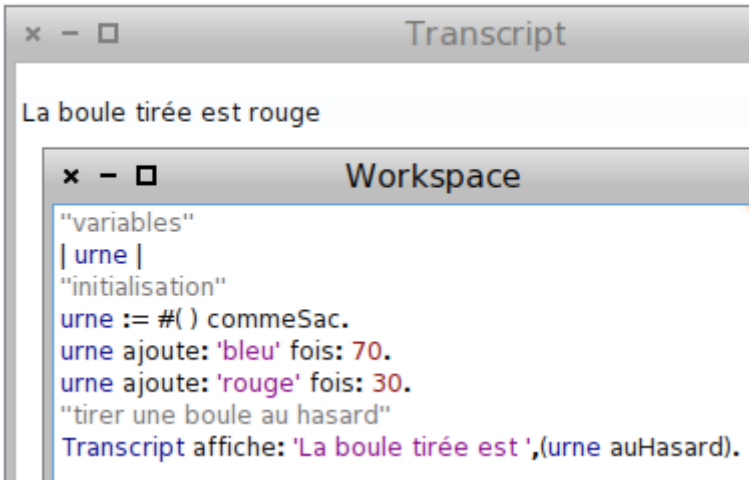
- ***Version pour enfants*** (ci-dessus) : Un distributeur de bonbons contient 70 bonbons à la menthe (en bleu) et 30 bonbons à la fraise (en rouge) ; en mettant une pièce d'un euro, on obtient un bonbon au hasard dans la boîte ; quelle est la probabilité qu'il soit à la fraise ?
- ***Version pour adultes*** : Une branche de fruitier[4] porte 100 fruits dont 70 sont déjà parvenus à maturité (en rouge ci-dessous), et 30 sont encore verts. Un « martin »[5] avale un des fruits au hasard. Quelle est la probabilité que ce fruit soit mûr ?

3 Un sac, ou multiensemble, est une structure de données de MathsOntologie. L'ordre d'introduction des éléments n'est pas répertorié, mais on peut mettre chacun en plusieurs exemplaires. Lorsque chaque objet n'apparaît qu'une fois au maximum dans un sac, on dit « ensemble » plutôt que sac. Les évènements en probabilités sont d'ailleurs représentés par des ensembles (axiomatique de Kolmogorov)

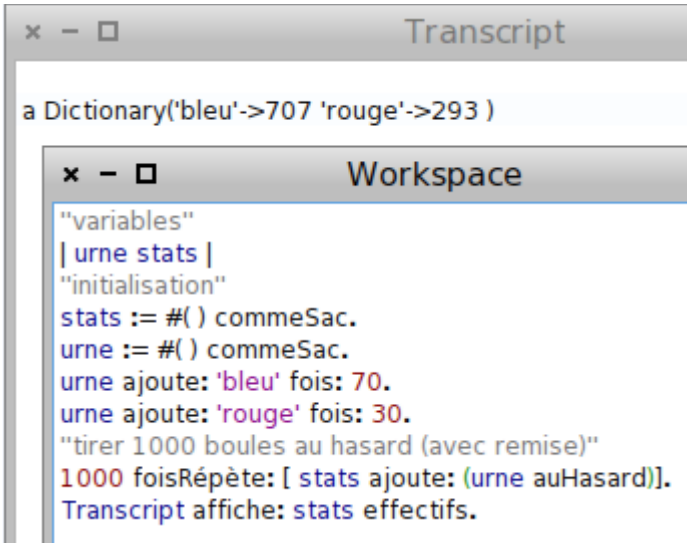
4 Dans la photo ci-dessus, un *coffea arabica*

5 *Acridotheres tristis*, un oiseau originaire d'Asie introduit par Pierre Poivre dans les Mascareignes pour lutter biologiquement contre les sauterelles, et qui fait des ravages dans les récoltes fruitières.

Pour simuler l'expérience avec MathsOntologie, on commence par créer un sac vide[6] qui s'appelle *urne*, on y place 70 boules bleues[7] et 30 boules rouges. Et on choisit une des boules au hasard dans l'urne :

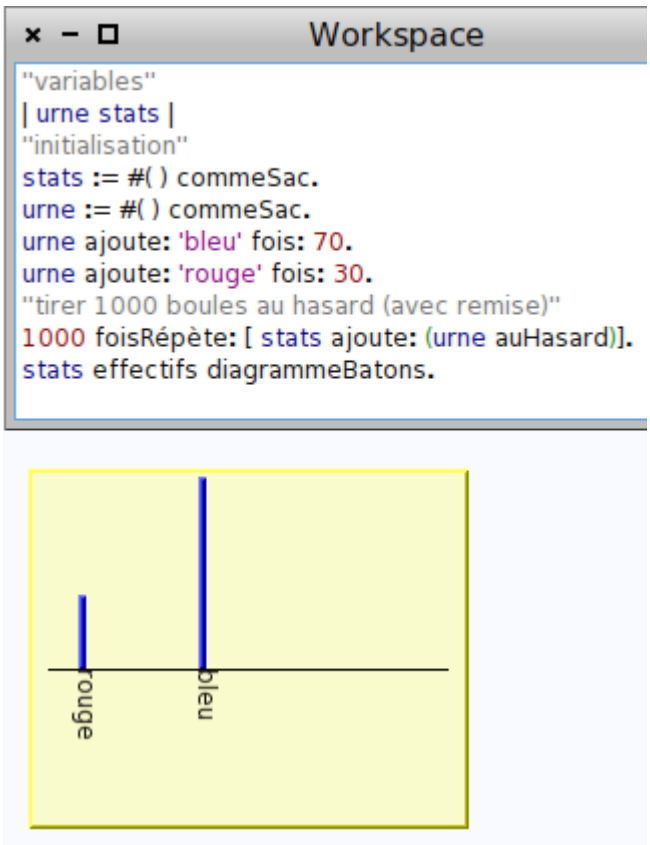


Maintenant qu'on peut simuler une expérience aléatoire, on peut répéter la simulation pour faire des statistiques[8] ; le tableau d'effectifs obtenu est alors un « dictionnaire »[9], qui associe à chaque résultat possible, le nombre de fois où on l'a obtenu :



On voit que les résultats statistiques sont eux-mêmes regroupés dans un autre sac, appelé *stats*. On peut représenter graphiquement les effectifs sous la forme d'un diagramme en bâtons :

6 En fait, on crée un tableau vide avant de le convertir en sac.
7 En fait, on y place 70 chaînes de caractères : Des mots « bleus » comme dirait Christophe...
8 C'est le but de Jacques Bernoulli : Illustrer la loi des grands nombres, qui permet de remplacer des probabilités parfois difficiles à calculer, par des fréquences mesurées « en vrai ».
9 Ou « application d'un ensemble dans un autre », l'ensemble de départ étant noté « keys » (les clés) et l'ensemble d'arrivée, « values » (les valeurs possibles) ; cette notion fondamentale en mathématiques décrit notamment les variables aléatoires (voir plus bas)

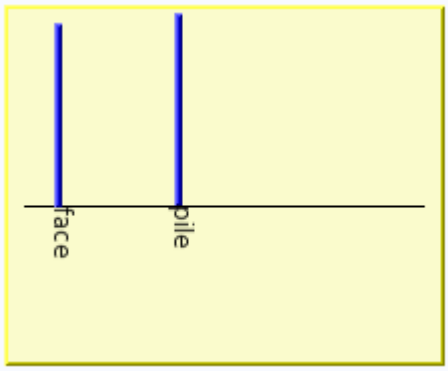


2) Pile ou face avec une urne

En fait, pour Bernoulli, le choix d'une boule dans une urne simule aussi le lancer d'une pièce truquée qui, dans le cas présent, a une probabilité 0,7 de tomber sur « pile » et une probabilité 0,3 de tomber sur « face ». Pour simuler une pièce non truquée, il suffit qu'il y ait autant de boules bleues que de boules rouges, et même une seule de chaque :

Du coup l'urne n'a plus besoin d'être un sac, on peut se contenter d'un tableau :

```
"variables"
| urne stats |
"initialisation"
stats := #( ) commeSac.
urne := #('pile' 'face').
"tirer 1000 boules au hasard (avec remise)"
1000 foisRépète: [ stats ajoute: (urne auHasard)].
stats effectifs diagrammeBatons.
```



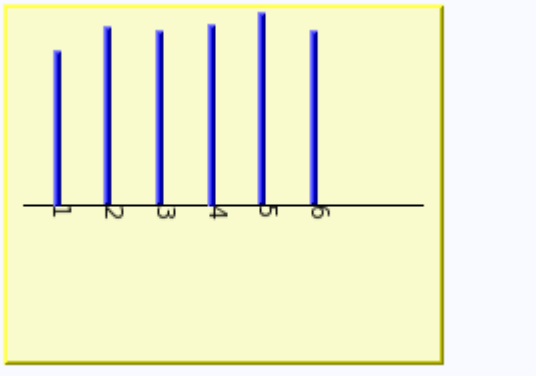
3) Lancer de dés avec une urne

a) Dé à 6 faces

Jacques Bernoulli s'intéressait surtout au cas « binomial » où il n'y a à chaque répétition, qu'un événement et son contraire ; mais on peut aussi envisager le cas « multinomial » où il y a plus de deux sortes de boules. Par exemple, le modèle d'urne de Bernoulli permet même de simuler le lancer d'un dé (ici un dé équilibré, si l'urne est assez grande pour permettre le mélange des boules) :

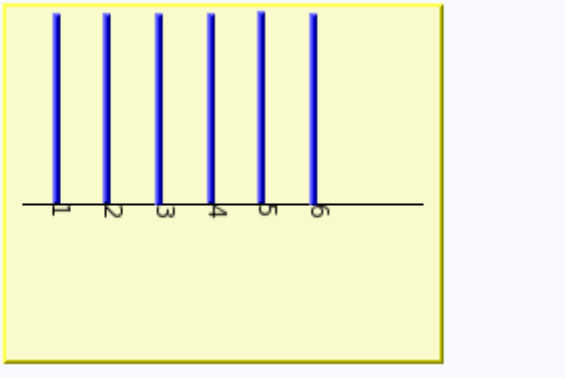
En MathsOntologie ça peut donner ceci :

```
"variables"
| urne stats |
"initialisation"
stats := #( ) commeSac.
urne := #(1 2 3 4 5 6).
"lancer le dé 1000 fois"
1000 foisRépète: [ stats ajoute: (urne auHasard)].
stats effectifs diagrammeBatonsTrié.
```



Mais dans ce cas précis, il suffit de choisir un nombre au hasard :

```
"variables"
| stats |
"initialisation"
stats := #( ) commeSac.
"lancer le dé 1 000 000 fois"
1000000 foisRépète: [ stats ajoute: (6 auHasard)].
stats effectifs diagrammeBatonsTrié.
```



Comme ça va plus vite avec un nombre qu'avec un tableau de nombres, on peut en quelques secondes lancer le dé un million de fois et voir que dans ce cas, il y a moins de fluctuation des fréquences autour des probabilités.

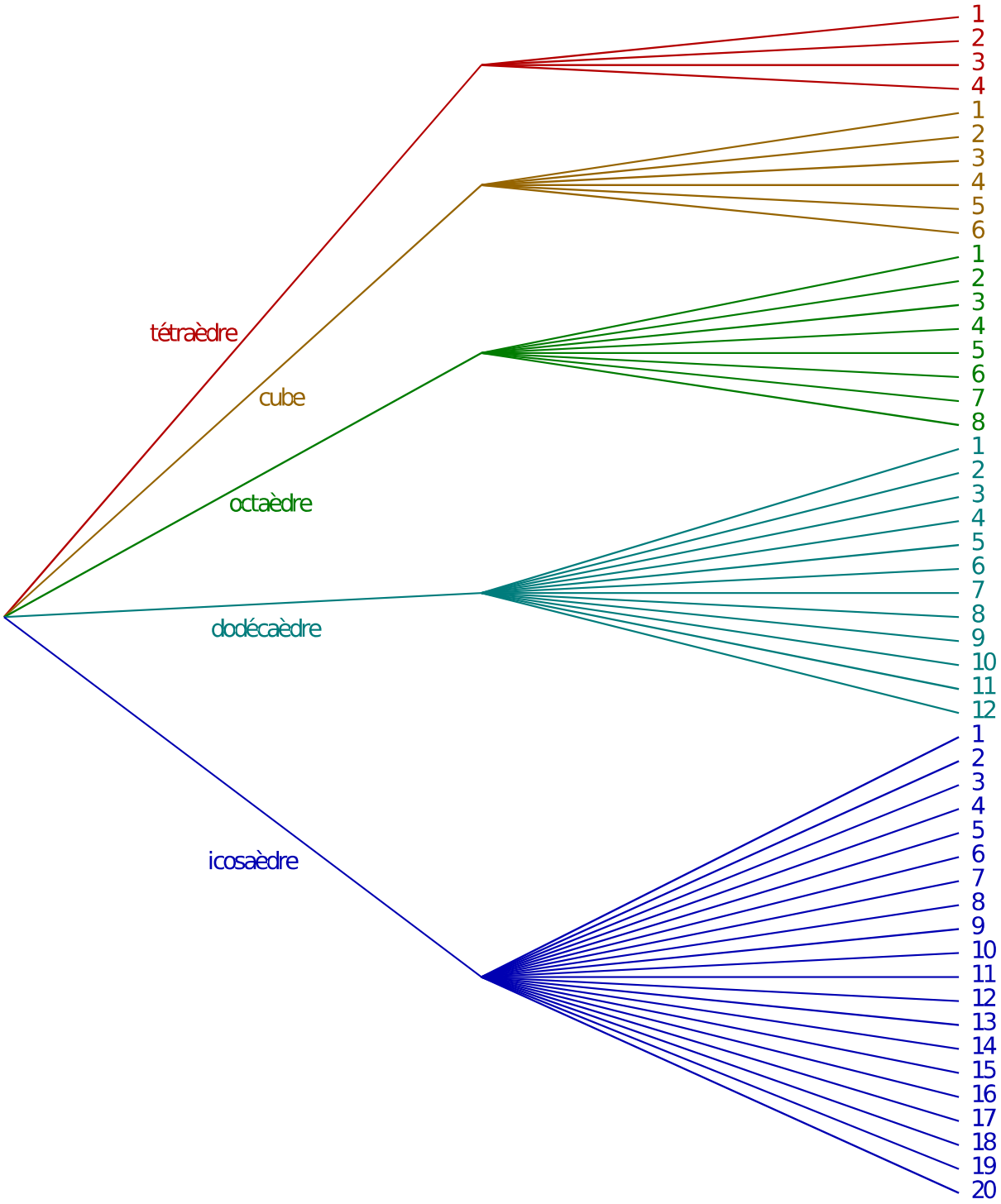
b) Plusieurs dés

La simplicité des simulations avec MathsOntologie permet alors de créer des problèmes en plusieurs étapes comme dans la situation suivante : ***Dans un sac, on a mis 5 dés, chacun ayant la forme d'un solide de Platon, numérotés à partir de 1*** :

On tire au sort un dé au hasard dans ce sac, et on le lance. Quelle est la probabilité d'avoir un 4 ?

Pour calculer la probabilité en question, on peut se servir de l'arbre suivant[10] :

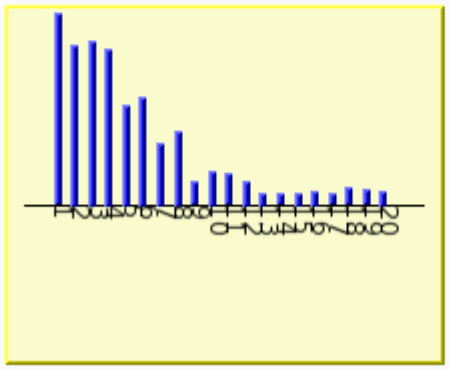
10 Les couleurs des branches de l'arbre correspondent aux couleurs des dés sur la photo



On compte 5 feuilles portant le numéro 4, et la probabilité voulue est donc 4/50=0,08. Mais c'est le tracé de l'arbre qui est long...

Une simulation avec MathsOntologie est très courte :

```
"variables"
| urne stats |
"initialisation"
stats := #( ) commeSac.
urne := #(4 6 8 12 20).
1000 foisRépète: [ stats ajoute: ((urne auHasard) auHasard)].
stats effectifs diagrammeBatonsTrié.
```



II/ En Seconde

1) Évènements

En troisième, un événement est décrit par une phrase comme « le dé est tombé sur un nombre pair ». Pour calculer la probabilité d'un événement, il est souhaitable de décrire celui-ci plutôt comme une liste d'éventualités (les cas favorables), que par une phrase. Euler écrivait :

> *La probabilité d'un événement quelconque est exprimée par une fraction, dont le numérateur est le nombre des cas où cet événement arrive, & le dénominateur est le nombre de tous les cas possibles*

(« *sur la probabilité des séquences dans la lotterie génoise* », page 193)

Ainsi, une probabilité est une fonction dont les antécédents sont des événements (des ensembles), et les images sont des réels compris entre 0 et 1. Avant de parler de probabilité, il est donc bon d'explorer un peu la notion d'ensemble et les opérations d'intersection et réunion[11] entre ensembles.

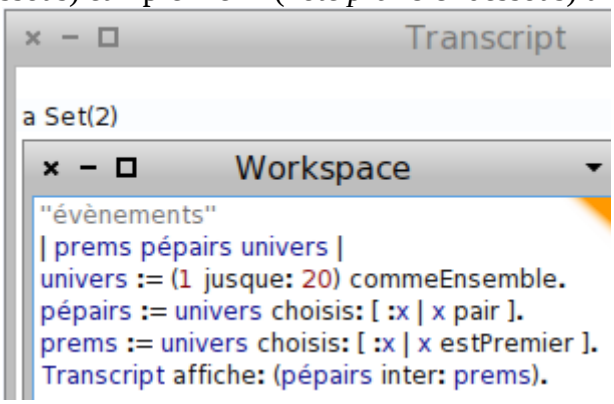
Par exemple, si on lance un dé icosaèdrique, voici la description ensembliste de quelques évènements :

- « le résultat est inférieur à 5 : {1, 2, 3, 4}
- « le résultat est négatif : { }
- « le résultat est pair : {2, 4, 6, 8, 10, 12, 14, 16, 18, 20}
- « le résultat est premier : {2, 3, 5, 7, 11, 13, 17, 19}

11 D'autant qu'en général, l'ensemble des solutions d'une inéquation est une réunion d'intervalles ; quant à l'intersection d'intervalles, elle est abordée en Terminales STI2D et STL où on explique que les événements (!) d'avril 2002 étaient prévisibles car les intervalles de fluctuation des candidats Jospin et Le Pen avaient une intersection non vide : Encore une histoire d'urne...

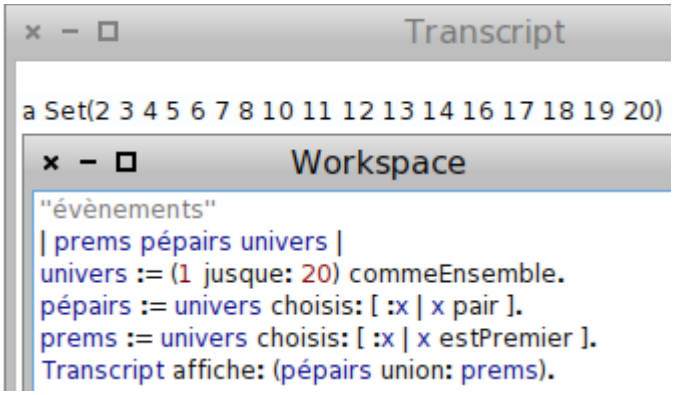
MathsOntologie possédant un objet « ensemble » (*Set* en anglais, qu'on voit lors de l'affichage), on peut calculer des événements en les déclarant comme ensembles au lieu de sacs (ce qui a pour effet de limiter le nombre d'occurrences de chaque éventualité à 1 maximum). Pour construire l'univers, on convertit l'intervalle d'entiers « *1 jusque: 20* » en un ensemble (ligne 3 ci-dessous) ; pour obtenir les résultats pairs, on extrait de l'univers les x tels que x est pair (ligne 4 ci-dessous) ; pour obtenir les nombres premiers, on extrait de l'univers les x tels que x est premier (ligne 5 ci-dessous) ; alternativement on aurait pu faire *prems:= #(2 3 5 7 11 13 17 19) commeEnsemble*

Pour décrire l'événement « le résultat est à la fois pair et premier », on réalise entre les événements « pair » (noté *pépair* ci-dessous) et « premier » (noté *prems* ci-dessous) un intersection notée *inter* :



On voit qu'il n'y a pas beaucoup de nombres à la fois premiers et pairs[12]...

De même, si on veut décrire en extension l'évènement « le résultat est ou bien premier, ou bien pair », on effectue entre les deux évènements de base, une réunion (on réunit les contenus de deux urnes en une seule urne résultante) notée *union* :



Cet exemple illustre que le « ou » en question est inclusif : Un nombre « pair ou premier » peut très bien être les deux à la fois.

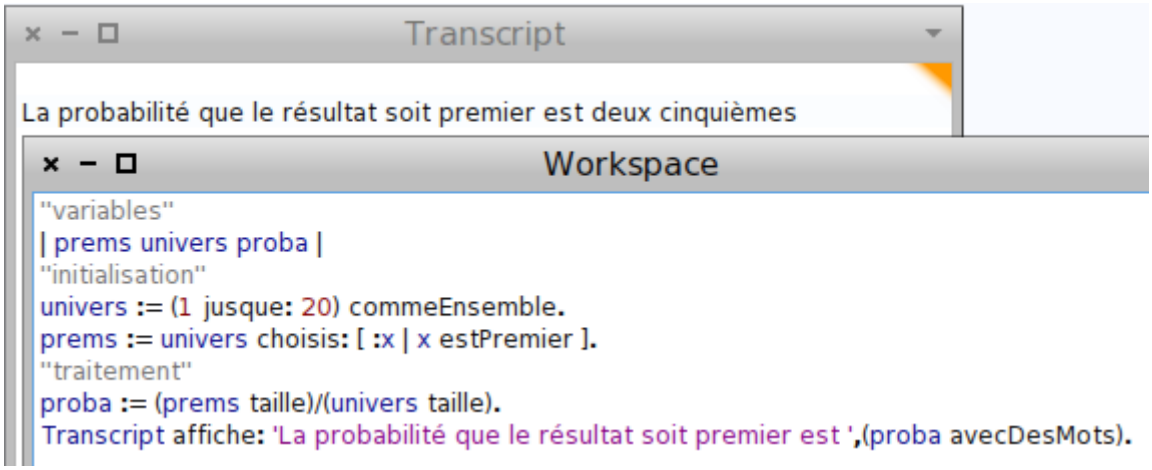
2) Probabilité d'un événement

Selon Euler, pour calculer la probabilité d'un événement, on doit diviser le nombre d'éventualités qui lui sont favorables, par le nombre d'éventualités de l'univers entier. On multiplie les exemples rapidement en laissant MathsOntologie compter plutôt que compter soi-même. Ainsi *univers compte : [:x | x estPremier]* renvoie 8. Mais une fois que l'événement prems a été défini, on peut aussi demander sa taille, qui est le nombre d'éventualités qu'il contient : *prems taille* renvoie aussi 8.

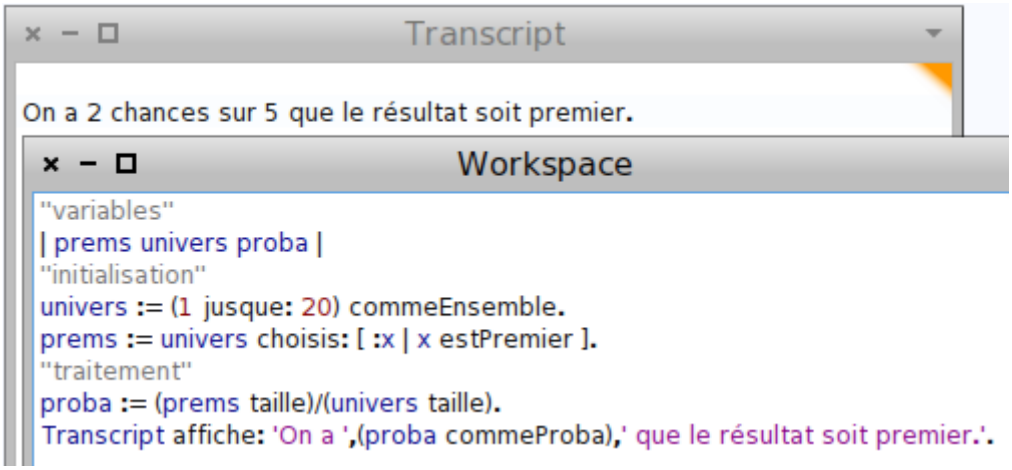
---

12 C'est un bon exemple de démonstration par l'absurde, assez courte pour se concentrer sur l'utilisation de la contraposée, donc abordable en Seconde : Si un nombre premier supérieur à 2 était pair, il serait divisible par 2 et ne saurait donc être premier.

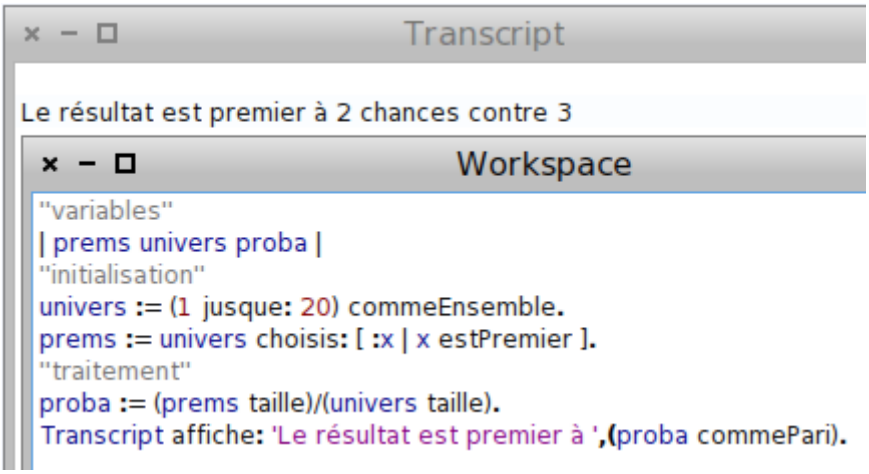
La définition donnée par Euler pour une probabilité insiste bien sur la nature de celle-ci : C'est une fraction. Or lorsqu'on ne précise pas le type d'un nombre dans MathsOntologie, le type par défaut est justement une fraction. Il suffit alors d'effectuer la division pour avoir la probabilité que le résultat soit premier (la fraction est automatiquement simplifiée) :



Il y a aussi une façon spéciale d'écrire une fraction si celle-ci est une probabilité :



Les bookmakers ont eux aussi leur langage bien particulier :



3) Dé à 12 faces

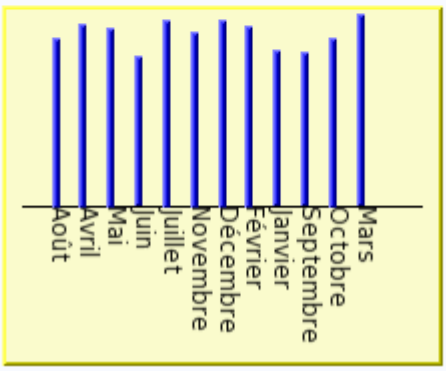
Une variante avec un dé dodécaédrique comme on en utilise dans les jeux de rôle : Au lieu de mettre des numéros sur les faces du dé, on peut mettre des noms de mois[13] ; on peut le faire à la main mais aussi utiliser un algorithme ad hoc pour construire la liste des noms de mois dans l'année. Le but du problème est de choisir un mois au hasard dans l'année (de façon équiprobable) et de calculer la probabilité qu'on puisse manger des huîtres ce mois-ci. Pour calculer l'événement « on peut manger des huîtres », on cherche la présence de la lettre 'r' dans le nom du mois, ce qui s'écrit simplement *mois contient: 'r'* ; mais au préalable, on doit calculer la liste des noms des 12 mois de l'année. L'algorithme est le suivant :

- On crée une variable de type « date » par DateAndTime new ;
- Cette date est alors initialisée au premier janvier 1900 ; on lui ajoute 15 jours pour être certain de toujours tomber au milieu du mois ;
- On initialise une urne vide comme précédemment ;
- On répète 12 fois (car il y a 12 mois dans l'année) l'opération de mettre le nom du mois courant dans l'urne et ajouter un mois.

On peut alors faire une simulation de 1000 choix de mois au hasard pour vérifier l'apparente équiprobabilité :

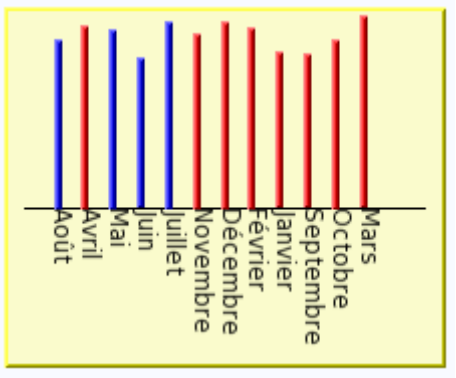
```
"variables"
| urne stats instant |
"initialisation"
instant := DateAndTime new + (15 jours).
stats := #( ) commeSac.
urne := #( ) commeSac.
12 foisRépète: [
    urne ajoute: instant nomDuMois.
    instant := instant+(30 jours).
    ].

1000 foisRépète: [ stats ajoute: (urne auHasard)].
stats effectifs diagrammeBatons.
```

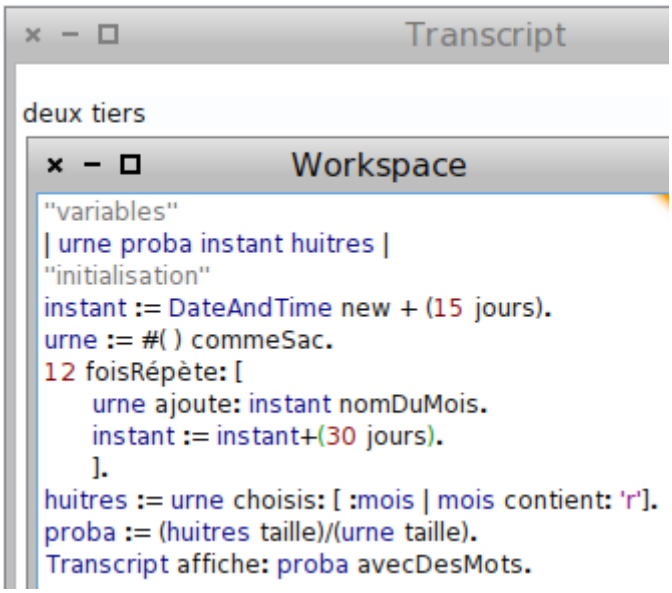


13 En Seconde ArMu, une variante est intéressante : mettre sur chaque face d'un dé dodécaédrique le nom d'une note de la gamme chromatique : do, do#, ré, mi bémol etc. En lançant 3 tels dés, on définit un accord ; on peut demander aux élèves de calculer la probabilité que l'accord soit dissonnant ; on peut même demander à un élève de jouer au piano l'accord après chaque lancer de trois dés. Ce faisant, on réinvente la musique stochastique chère à Yannis Xenakis.

Le diagramme en bâtons n'est pas trié, et les mois apparaissent dans le désordre. En coloriant en rouge ceux dont le nom comporte la lettre 'r', on peut calculer la probabilité de pouvoir manger des huîtres :



On compte 8 mois sur 12 où on peut manger des huîtres, la probabilité cherchée est donc le quotient de 8 par 12, soit deux tiers :



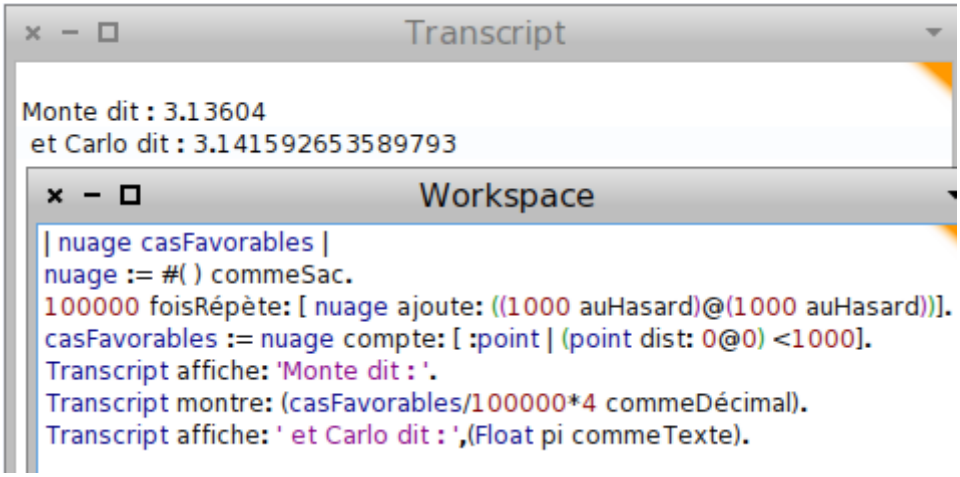
4) Estimation de π par l'algorithme de Monte-Carlo

Dans une urne, on ne peut pas seulement mettre des nombres ou des mots : On peut aussi mettre des couleurs, des dates, des points, des fonctions etc. Un sac plein de points aléatoires permet de calculer π par la méthode dite de Monte-Carlo : On lance une fléchette sur une cible carrée, et on compte combien de fois la fléchette est tombée dans un quart de cercle. En multipliant le quotient des aires par 4, on devrait obtenir un nombre voisin de π.

Pour cela, on remplit un sac avec 10 000 points dont l'abscisse et l'ordonnée sont aléatoires (entre 0 et 1 000).

Puis on compte parmi les 10 000 points, combien sont à une distance de (0;0) inférieure à 1 000.

Au bout de quelques secondes, on trouve une approximation de π plutôt bonne :



Ce genre d'activité peut être réinvesti en Terminale, à propos du calcul intégral (voir plus loin).

5) Théorie de l'échantillonnage

Selon Bernoulli, un échantillon s'obtient non en tirant une boule de l'urne, mais en en tirant plusieurs de suite (de façon non exhaustive pour garantir l'indépendance entre les tirages). Autrement dit, un échantillon est une sous-urne. Les jeux de hasard en sont pleins :

- une main au poker (ou dans tout autre jeu de cartes) est un échantillon prélevé dans une urne (ou paquet) de cartes
- une main au scrabble est un échantillon prélevé dans un sac de lettres
- une main au domino … etc
- lancer plusieurs fois un dé, c'est comme lancer plusieurs dés (en général depuis un gobelet) ; par exemple, le poker d'as, le 421 etc.
- le jeu de « quine »[14] est basé sur un tirage au sort dans une urne de nombres allant de 1 à 90

C'est sur ce dernier exemple qu'on va simuler des tirages, parce que le fait qu'il s'agisse de numéros simplifie la simulation. On peut aussi simuler, à l'aide de chaînes de caractères, les autres jeux cités ci-dessus.

Cependant, les tirages des jeux de hasard sont faits sans remise (à part lorsqu'on lance plusieurs fois un dé). Et il est plus facile (et plus conforme au programme) de s'intéresser à des tirages avec remise. D'ailleurs la probabilité que dans un tirage de 5 nombres au hasard, un même nombre arrive plusieurs fois, est assez faible : La probabilité de son contraire est 1 220 813 chances sur 1 366 875 soit environ 0,89 ; on peut l'assimiler à la probabilité de succès quand on remplace un tirage sans remise par un tirage avec remise.

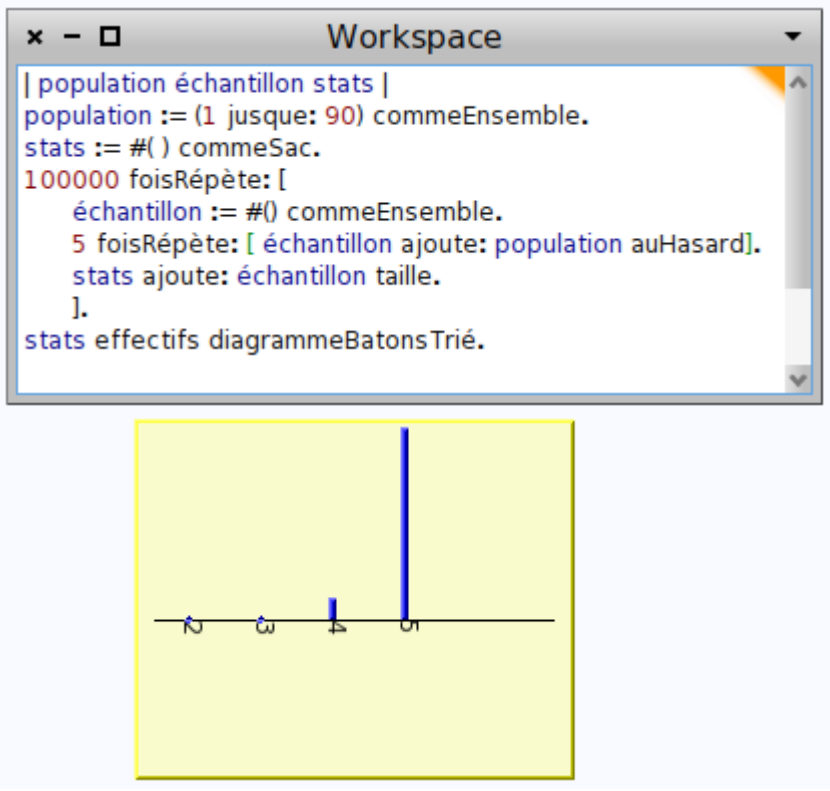
---

14 Très populaire sur l'île de La Réunion, ce jeu semble inspiré du fait qu'avant d'être une république avec un doge élu, la ville de Gènes tirait au sort ses 5 conseillers municipaux ; au XVIe siècle, le roi de France François Ier avait compris l'intérêt (!) fiscal qu'il y avait à lancer un jeu de hasard basé sur ce principe. Les 5 billets gagnants étaient noirs, les autres blancs. De là viendraient

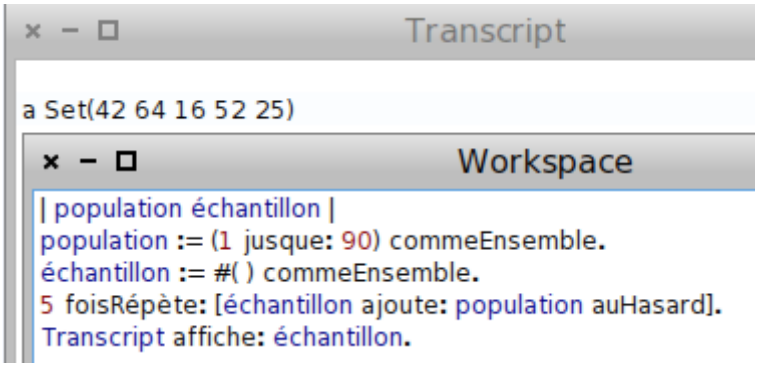
- l'expression « faire chou blanc », à cause de la couleur des billets perdants
- l'idée des boules noires et blanches chez Bernoulli (qui étaient effectivement remplacées par des billets chez Laplace)
- le nom de « quine » donné à cette ancienne forme de loto (il y avait 5 billets, puis 5 numéros, gagnants)
- le nom de « génoise » donné par Euler à la loterie citée ci-dessus.

Le jeu favorisé par François Ier a ensuite été interdit, jusqu'à ce que l'espion Casanova écrive au roi de France Louis XV une lettre exposant la valeur fiscale d'une loterie nationale (où seul l'État a le droit de jouer le rôle de la banque). L'idée continue à faire son chemin dans la France d'aujourd'hui...

La probabilité ci-dessus n'est pas calculable en Seconde, mais on peut l'estimer par simulation :



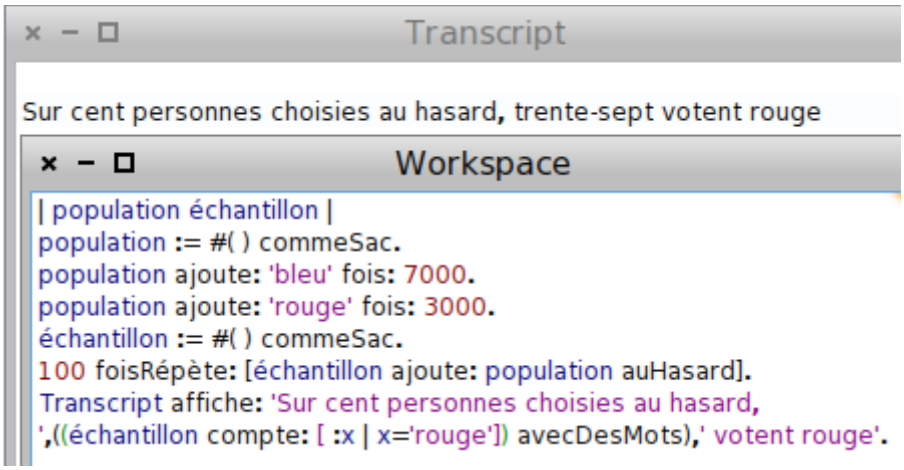
On voit ci-dessus comment on peut simuler un tirage avec remise : L'urne s'appelle maintenant *population* et on crée un second sac initialement vide appelé *échantillon*. On y place 5 fois de suite un élément de la population choisi au hasard :



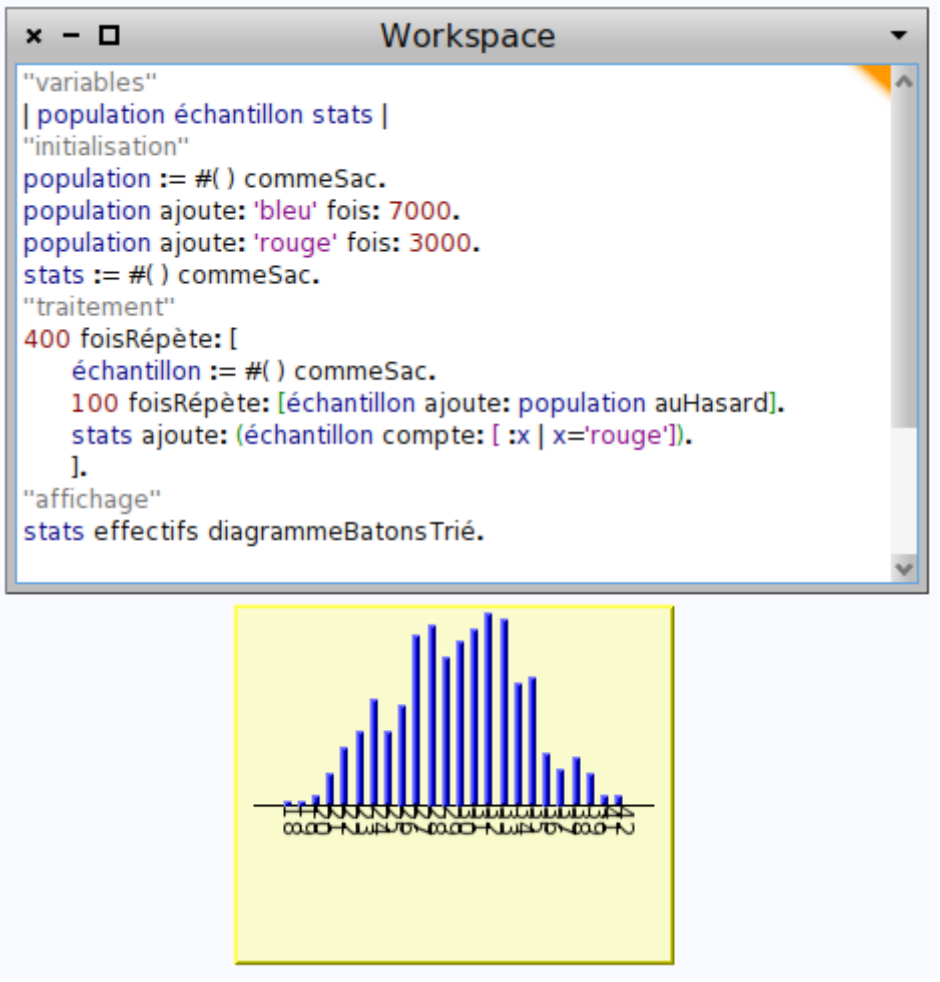
Le choix des noms *population* et *échantillon* n'est évidemment pas innocent : Après avoir dangereusement laissé les élèves de Seconde jouer au loto-quine, on profite de leur nouvelle connaissance des échantillons (et de leur fluctuation) pour parler de sondages d'opinion. Avec un exemple fictif[15] : Dans la commune de Hénon-Beaumint (dix mille électeurs), le maire sortant est membre du Parti Carmin, dont la couleur est le rouge, et son principal challenger est le jeune loup de l'Union Magenta-Phtalocyanine, dont la couleur est le bleu. Le maire du PC (rouge donc) a tellement déçu ses électeurs que seuls 3000 d'entre eux sont encore prêts à voter pour lui aux municipales. Le jeune loup de l'UMP (bleu donc) rassemblant les 7000 suffrages restants. Bien entendu, jusqu'aux élections municipales, ni le maire ni son futur successeur ne connaissent le résultat qui sortira des urnes. Alors le maire rouge va, sur les deniers de la commune, commanditer un sondage d'opinion, consistant à interroger 100 électeurs au hasard. Le résultat du sondage lui

15 Toute ressemblance avec des candidats aux dents longues est donc fictivement involontaire...

glace le sang : Seuls 37 d'entre eux voteraient pour lui :



Ignorant que l'échantillon lui était plutôt favorable, il va alors vider les finances de la commune en commanditant 400 nouveaux sondages, effectués chacun sur 100 personnes ; le diagramme en bâtons montre bien la fluctuation d'échantillonnage :



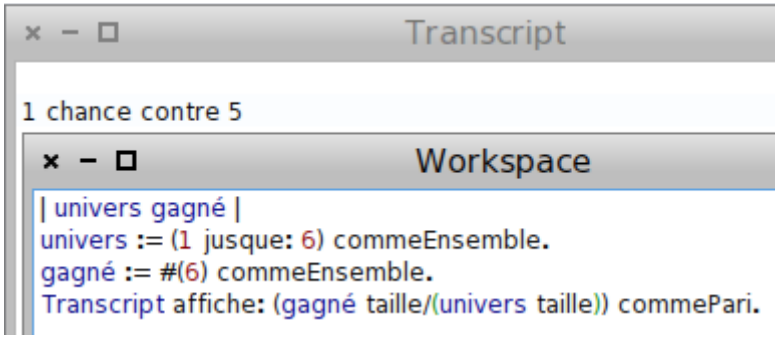
On voit là l'esquisse d'un intervalle de fluctuation : La plupart des échantillons se situent entre 20 et 40 % d'opinions favorables.

III/ En Première

1) Variable aléatoire

On rappelle qu'un événement est un ensemble (liste d'éventualités non triée). Si, à chaque éventualité, on associe un nombre[16], on définit un nombre aléatoire, qui s'appelle donc ***variable aléatoire***. Pour associer un nombre à chaque éventualité, MathsOntologie utilise un ***dictionnaire***, où des flèches représentent les associations. Un exemple a déjà été vu avec les tableaux d'effectifs, et la notion de loi de probabilité fait elle aussi appel à des dictionnaires. Pour créer un exemple de variable aléatoire, on peut chercher à inventer un jeu équilibré : Combien doit-on parier sur l'obtention d'un 6 en lançant un dé, pour que le pari soit équitable ?

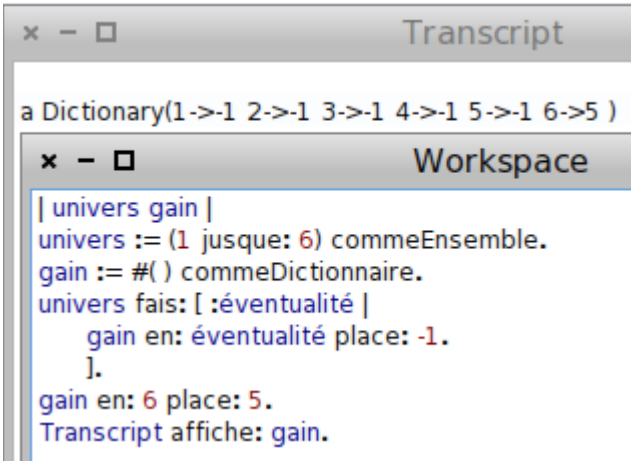
Une méthode un peu compliquée mais généralisable est celle-ci :



Une possibilité pour que le jeu soit équitable est qu'on gagne 5 euros si on a un 6 et qu'on perde 1 euro sinon. Ce qui se résume à ce tableau :

1. → -1 €
2. → -1 €
3. → -1 €
4. → -1 €
5. → -1 €
6. → 5 €

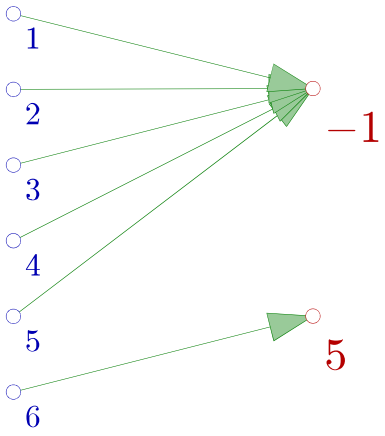
Le moyen le plus rapide de remplir le dictionnaire (une fois créé avec une liste vide convertie en dictionnaire) est de n'y mettre que des « -1 » puis remplacer le dernier « -1 » par un « 5 » :



Cette notion de dictionnaire, ou application d'un ensemble (ici {1, 2, 3 4, 5, 6}) dans un autre (ici,

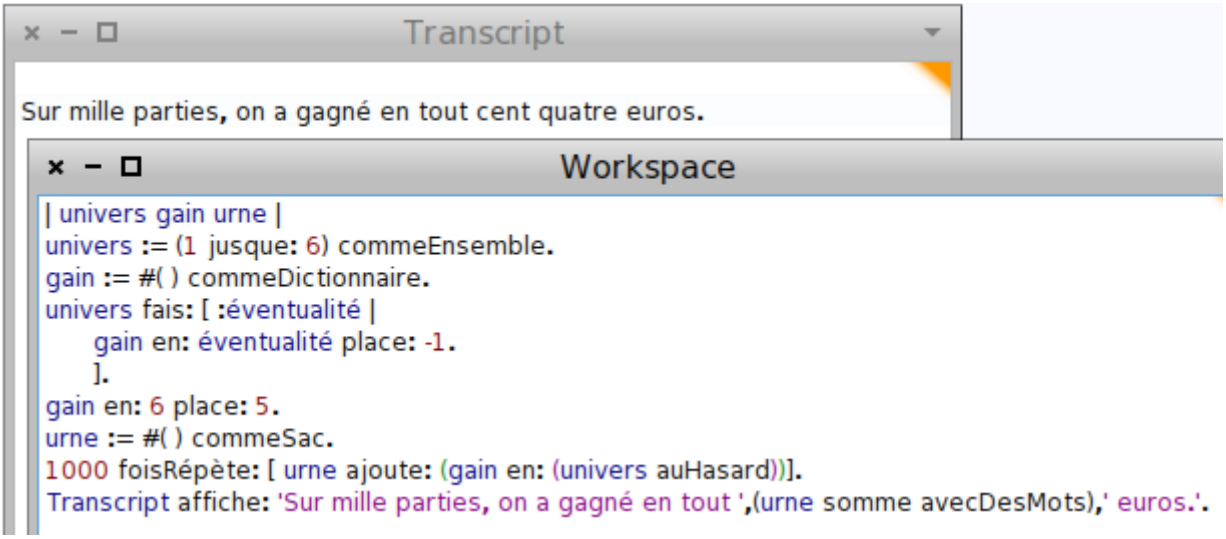
16 Pas nécessairement réel : On peut aussi, en Terminale, parler de variable aléatoire complexe

$\mathbb{Z}$) est fondamentale en mathématiques, parce que si l'ensemble de départ n'est plus nécessairement fini, on obtient une fonction numérique, objet d'une partie non négligeable du cours... On représente volontiers un dictionnaire par son graphe :

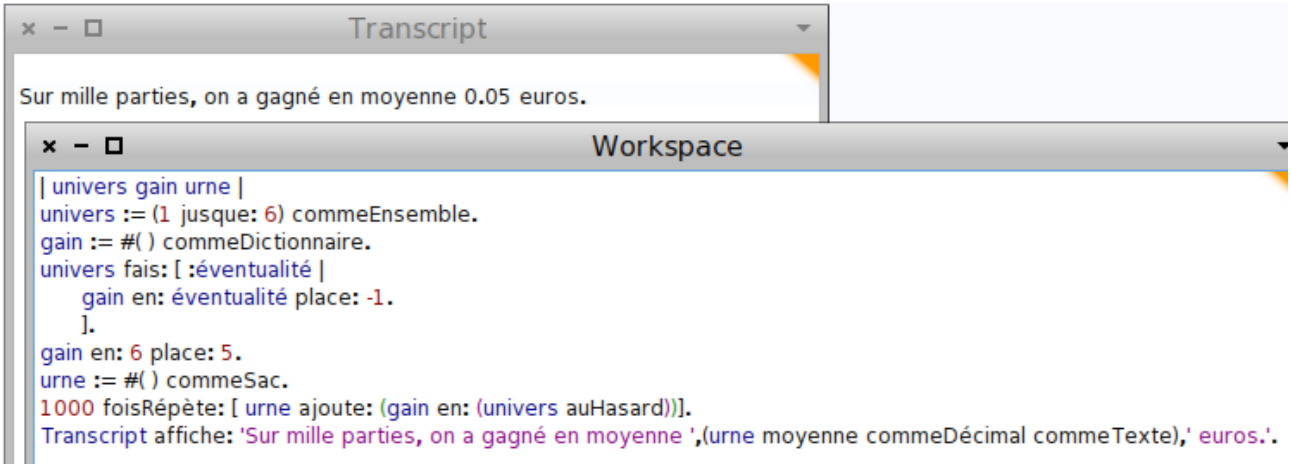


Lorsqu'on remonte les flèches, on traduit par des événements les valeurs que peut prendre la variable aléatoire : Ici, l'événement « le résultat du lancer de dé est inférieur à 6 » correspond à la valeur -1 €, et l'événement « le dé est tombé sur 6 » correspond à la valeur 5 €.

Pour simuler cette variable aléatoire, on choisit un nombre entre 1 et 6 au hasard, et on lit dans le dictionnaire la valeur que prend alors la variable aléatoire. Comme précédemment, on peut faire tout cela répétitivement, et la somme de la longue liste de nombres (mais tous égaux à -1 ou 5) donne le gain total :

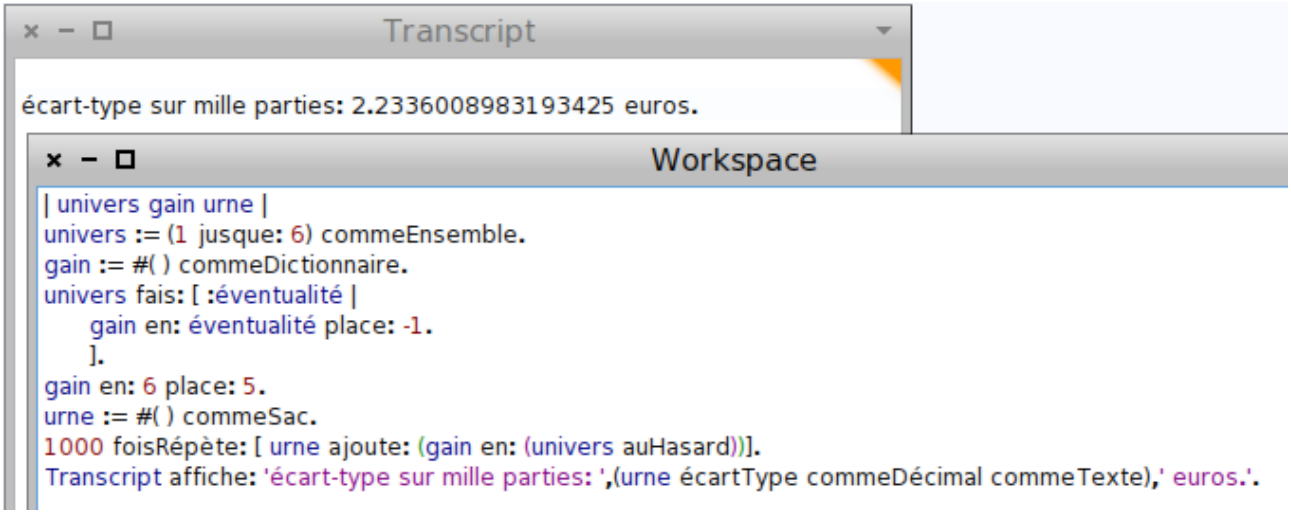


Le gain moyen est peut-être plus parlant, et surtout il mène à la notion d'espérance :



Il résulte en effet de la loi des grands nombres, que la moyenne sur un échantillon important est proche de l'espérance. En définissant la loi de la variable aléatoire par un dictionnaire *loi*, on peut définir l'espérance comme le nombre produit par cet algorithme : *(picore: [:éventualité | (gain en: éventualité)*(loi en: éventualité)]) somme*.

Mais on a tout fait pour que le résultat obtenu soit nul (le jeu devant être équilibré). De toute manière, des calculs d'espérance sont montrés plus bas. L'écart-type aussi approche d'une valeur limite, ce qui permet de définir l'écart-type d'une variable aléatoire :

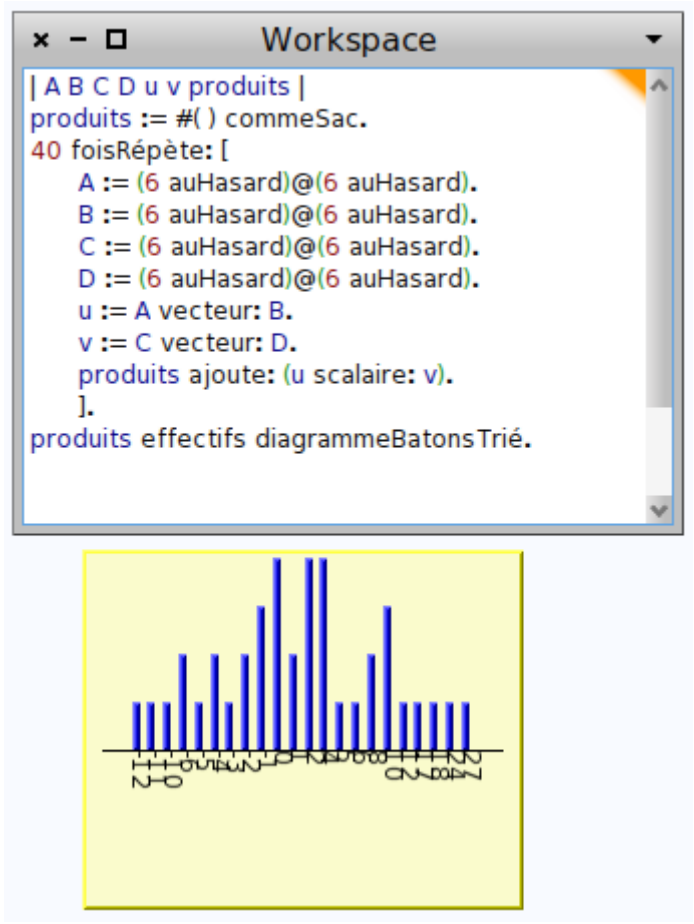


La valeur théorique est $\sqrt{5} \approx 2{,}236$.

2) Exemple de variable aléatoire discrète

On lance un dé 8 fois de suite : Les deux premiers résultats sont les coordonnées d'un point A (abscisse d'abord, ordonnée ensuite), les deux suivants sont les coordonnées d'un point B, les deux suivants, les coordonnées d'un point C et les deux derniers résultats sont les coordonnées d'un quatrième point D (toujours l'abscisse d'abord, l'ordonnée ensuite). Alors le produit scalaire des vecteurs $\overrightarrow{AB}$ et $\overrightarrow{CD}$ est un nombre dépendant des résultats des 8 lancers de dé : C'est une variable aléatoire. On demande sa loi, son espérance et sa variance.

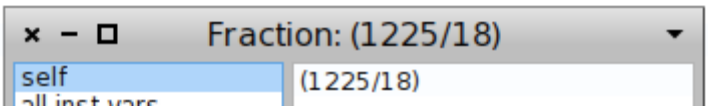
Pour commencer, une simulation montre que les produits scalaires sont assez souvent petits en valeur absolue :



Pour calculer les différentes probabilités, on fait comme pour les simulations, mais en parcourant toutes les possibilités dans des boucles imbriquées et une seule fois chaque possibilité. On découvre alors

- que les produits scalaires vont entre -50 et 50,
- que l'espérance est nulle,
- et que la variance est $\frac{1225}{18}$.

```
| A B C D u v produits |
produits := #( ) commeSac.
(1 jusque: 6) fais: [ :xA |
    (1 jusque: 6) fais: [ :yA |
        A := xA@yA.
        (1 jusque: 6) fais: [ :xB |
            (1 jusque: 6) fais: [ :yB |
                B := xB@yB.
                (1 jusque: 6) fais: [ :xC |
                    (1 jusque: 6) fais: [ :yC |
                        C := xC@yC.
                        (1 jusque: 6) fais: [ :xD |
                            (1 jusque: 6) fais: [ :yD |
                                D := xD@yD.
                                u := A vecteur: B.
                                v := C vecteur: D.
                                produits ajoute: (u scalaire: v).
                                ]
                            ]
                        ]
                    ]
                ]
            ]
        ]
    ].
Transcript affiche: (produits moyenne).
Transcript affiche: (produits écartType auCarré).
```



Si on trouve l'exemple trop compliqué, on peut ne lancer le dé que 6 fois, définir seulement trois points A, B et C et mettre dans l'urne un *A aireDuTriangleAvec: B et: C* et avoir la loi de l'aire du triangle ABC. On peut aussi étudier la variable aléatoire « distance entre A et B », les coefficients de l'équation réduite de la droite (AB), la distance de A à 0, etc.

3) Le jeu des petits chevaux

Le jeu de dada, ou des petits chevaux, est une sorte de jeu de l'oie simplifié : Chaque cheval (ou pion) avance d'un nombre de pas égal au résultat du lancer d'un dé, sauf la première fois, où il faut un 6 pour « sortir le cheval de l'écurie »[17]. La question se pose alors de savoir combien de fois en moyenne on doit lancer le dé avant de pouvoir sortir le cheval de l'écurie. La loi suivie par la variable aléatoire « nombre de lancers successifs du dé » est dite géométrique[18] ou de Pascal[19]

Un problème à la Blaise Pascal : Combien de fois faut-il lancer un dé au minimum, pour que la probabilité d'avoir un 6 dépasse 0,99 ?

En fait il est plus facile de chercher combien de fois on doit lancer le dé pour que la probabilité de ne pas avoir de 6 descende en-dessous de 0,01 : Cette dernière probabilité est une suite géométrique, par indépendance des répétitions des lancers. On découvre qu'il faut lancer le dé au moins 27 fois avec l'algorithme[20] suivant :

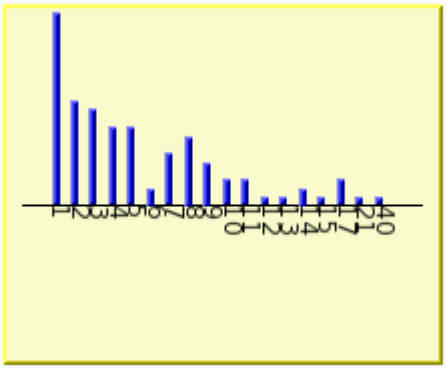
17 Le jeu « trivial poursuite » connaît une situation similaire mais en fin de jeu.
18 Parce que la suite des probabilités qu'elle soit égale à 1, à 2, à 3 etc. est une suite géométrique.
19 Bernoulli en attribuait la paternité à Blaise Pascal.
20 En fait c'est un pur exercice sur les suites géométriques (limite nulle à l'infini et décroissance), mais habillé avec des probabilités

```
| u n |
n := 0.
u := 1.
[ u < 0.01] jusqueVrai: [
    n := n augmenteDe1.
    u := u multipliePar: (5/6).
    Transcript affiche: 'En lançant le dé ',(n avecDesMots),' fois, la probabilité de gain est ',((1-u) commeProba).
    ].
```

```
En lançant le dé vingt-six fois, la probabilité de gain est
16909161206019344263 1 chances sur 17058172817957820825 6
```

Une simulation de 100 parties du jeu des petits chevaux se fait en lançant le dé jusqu'à ce qu'on ait un 6 (et ceci, 100 fois) :

```
| durées |
durées := #( ) commeSac.
100 foisRépète: [ | n |
    n := 1.
    [(6 auHasard) = 6] jusqueVrai: [ n := n suivant.].
    durées ajoute: n.
    ].
durées effectifs diagrammeBatonsTrié.
```



On voit le caractère géométrique de la suite. La répétition de 100 000 simulations du jeu suggère que l'espérance du nombre de lancers est 6 :

Transcript

```
6.0012
```

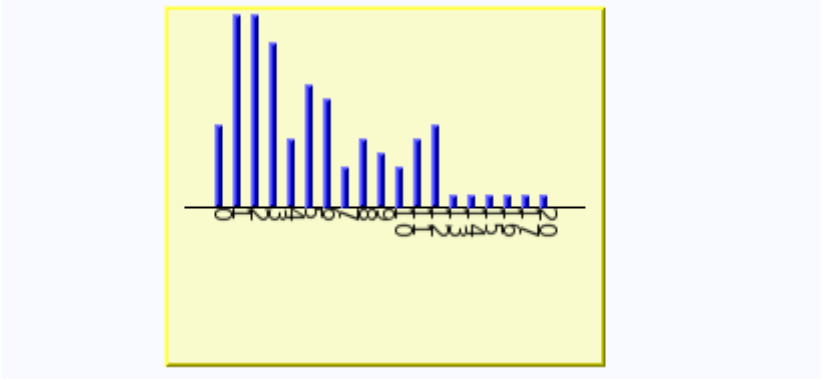
Workspace

```
| durées |
durées := #( ) commeSac.
100000 foisRépète: [ | n |
    n := 1.
    [(6 auHasard) = 6] jusqueVrai: [ n := n suivant.].
    durées ajoute: n.
    ].
Transcript affiche: (durées moyenne commeDécimal).
```

Pour transformer une loi géométrique (hors programme) en loi géométrique tronquée, il suffit de la tronquer ! Par exemple, si le poulain de Pauline tarde à sortir et que Pauline s'impatiente, elle peut abandonner le jeu au bout de 20 vaines tentatives ; autrement dit, on s'arrête non plus lorsqu'on a eu

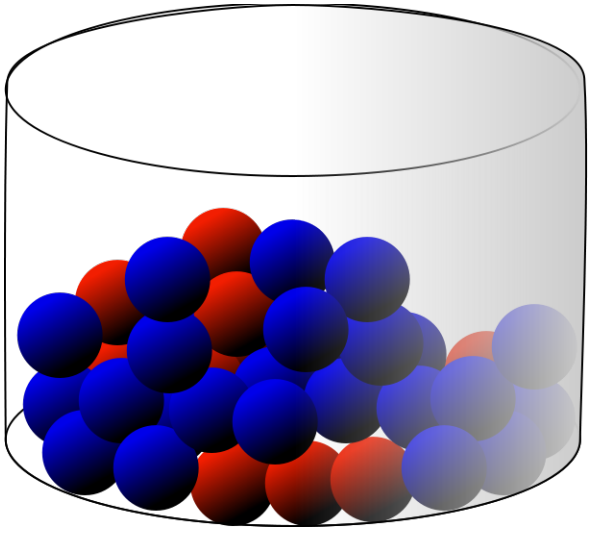
un 6, mais lorsque on a eu un 6 ou bien on a essayé 20 fois (mais dans ce cas on compte 0, par convention) :

```
| durées |
durées := #( ) commeSac.
100 foisRépète: [ | n |
    n := 1.
    [((6 auHasard) = 6) ou: (n > 20)] jusqueVrai: [ n := n suivant.].
    (n<=20) siVrai: [durées ajoute: n.] siFaux: [durées ajoute: 0].
    ].
durées effectifs diagrammeBatonsTrié.
```



4) La loi binomiale

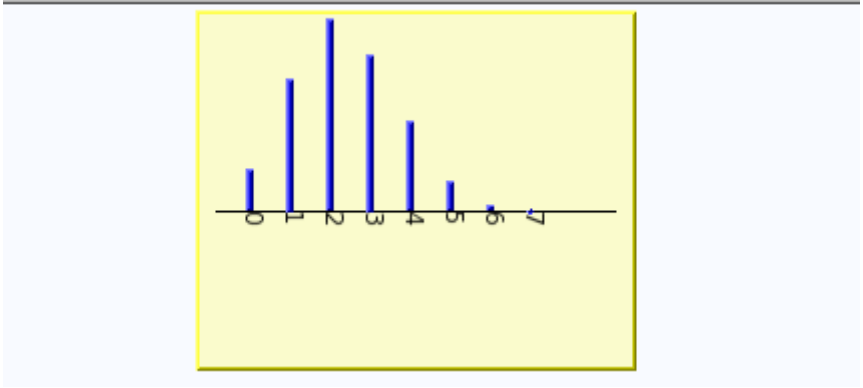
Retour à Jacques Bernoulli et à son urne de boules de deux couleurs ; mais cette fois-ci on tire 8 boules de l'urne et on compte combien d'entre elles sont rouges : Ce nombre est en effet une variable aléatoire, comprise entre 0 et 8.



La simulation est facile si on réinvestit ce qui a été vu précédemment : La phase d'initialisation consiste à créer et remplir l'urne comme on l'a déjà fait, et un sac vide destiné à recevoir les données sur lesquelles on va faire des *stats* (ici, un diagramme en bâtons). On boucle 1000 fois dans la phase « traitement » (parce qu'on veut faire 1000 simulations de tirages) ; dans cette boucle, on crée une nouvelle urne appelée *échantillon*, initialement vide, mais dans laquelle on va 8 fois de suite placer une boule au hasard choisie dans l'urne. Il ne reste alors plus qu'à compter le nombre de boules

rouges dans l'échantillon pour avoir une réalisation de la variable aléatoire :

```
"variables"
| urne stats échantillon |
"initialisation"
urne := #( ) commeSac.
urne ajoute: 'bleu' fois: 70.
urne ajoute: 'rouge' fois: 30.
stats := #( ) commeSac.
"traitement"
1000 foisRépète: [
    échantillon := #( ) commeSac.
    8 foisRépète: [échantillon ajoute: (urne auHasard)].
    stats ajoute: (échantillon compte: [ :couleur | couleur = 'rouge']).
    ].
"affichage"
stats effectifs diagrammeBatonsTrié.
```

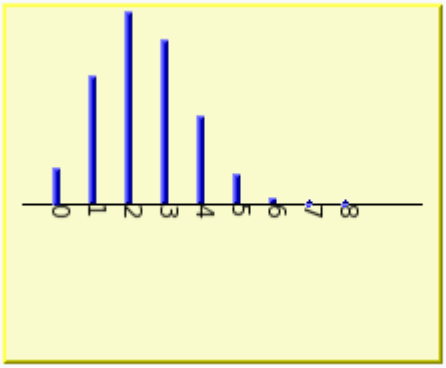


Bien que ce ne soit pas au programme, on peut donner un algorithme pour calculer la loi binomiale de paramètres 8 et 0,3 : La probabilité que le nombre de boules rouges dans l'échantillon soit égal à n est le produit de trois facteurs :

- le nombre de manières de choisir n objets parmi 8 : *n parmi: 8*
- la puissance nième de 0,3 : *0.3 puissance: n*
- la puissance (8-n)ième de 0,7 : *0.7 puissance: (8-n)*

Avec représentation graphique de la loi, cet algorithme donne ceci :

```
| binomiale N p q |
N := 8.
p := 0.3.
q := 1-p.
binomiale := #( ) commeDictionnaire.
(0 jusque: 8) fais: [ :n |
    | proba |
    proba := (n parmi: N)*(p puissance: n)*(q puissance: (N-n)).
    binomiale en: n place: proba.
    ].
binomiale diagrammeBatonsTrié.
```



On peut s'en servir pour calculer un intervalle de fluctuation :

5) Intervalle de fluctuation

Retour au village de Hénon-Beaumin : En supposant que 30 % des électeurs restent fidèles à leur maire rouge, un choix de 100 électeurs au hasard constitue un échantillon où le nombre d'électeurs rouges est une variable aléatoire binomiale de paramètres 100 et 0,3.

Transcript

```
Un intervalle de fluctuation à 95 % est [22 ; 38]
```

Workspace

```
"variables"
| binomiale N p q somme indice borneInf borneSup |
"initialisation"
N := 100.
q := 1-(p := 0.3).
binomiale := #( ) commeDictionnaire.
(0 jusque: N) fais: [ :n |
    | proba |
    proba := (n parmi: N)*(p puissance: n)*(q puissance: (N-n)).
    binomiale en: n place: proba.
    ].
"traitement"
indice := 0.
somme := 0.
[somme > 0.025] jusqueVrai: [
    somme := somme augmenteDe: (binomiale en: indice).
    indice := indice suivant.
    ].
borneInf := indice.
indice := 100.
somme := 0.
[somme > 0.025] jusqueVrai: [
    somme := somme augmenteDe: (binomiale en: indice).
    indice := indice précédent.
    ].
borneSup := indice.
"affichage"
Transcript affiche: 'Un intervalle de fluctuation à 95 % est ['.
Transcript montre: borneInf.
Transcript montre: ' ; '.
Transcript montre: borneSup.
Transcript montre: ']'.
```

Pour gagner du temps (de calcul), on précalcule les valeurs de la loi binomiale dans un tableau. Ensuite on initialise la variable indice à 0, puis on l'incrémente jusqu'à ce que la valeur cumulée de la loi binomiale dépasse 0,025 (soit 2,5 %, la moitié de 5%, en effet on veut un intervalle à 95 %). Le résultat est la borne inférieure de l'intervalle de fluctuation. Puis on recommence en réinitialisation la variable indice à 100, puis en la décrémentant jusqu'à ce que la valeur cumulée de la loi binomiale dépasse à nouveau 0,025 mais cette fois-ci, en descendant.

L'intervalle de fluctuation est [22 ; 38] à comparer avec celui vu en Seconde : [20;40].

IV/ En Terminale

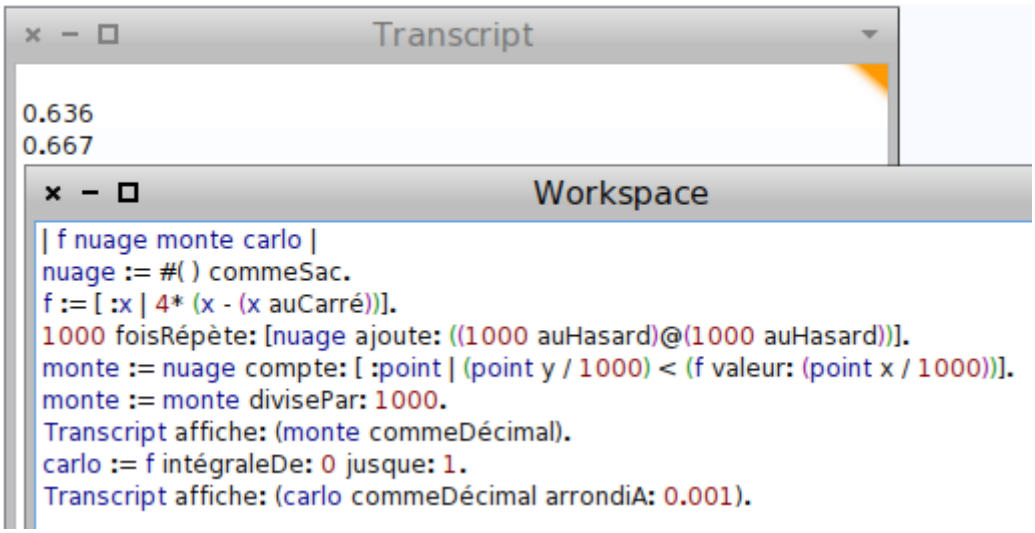
1) Intégrales

La méthode de Monte-Carlo, déjà vue en Seconde pour estimer l'aire d'un quart de cercle, peut être réinvestie en Terminale pour estimer des intégrales, ce qui permet d'aborder le passage des lois discrètes aux lois continues. Pour calculer l'intégrale sur [0;1] de $x \to 4x(1-x)$, on construit un nuage de points répartis uniformément dans le carré de côté 1, et on compte combien d'entre eux sont en-dessous de la parabole.

Comme seuls les nombres entiers peuvent être choisis au hasard[21], on divise un entier aléatoire de 3 chiffres par 1000 pour avoir un réel aléatoire entre 0 et 1.

Par souci de mnémotechnique, les deux variables numériques calculées s'appellent respectivement *monte* et *carlo* :

- *monte* contient l'estimation de l'intégrale par fréquence des points en-dessous de la parabole. Il faut convertir cette fraction en décimal pour bien la lire ;
- *carlo* contient l'intégrale calculée par la méthode des trapèzes (une fonctionnalité interne de MathsOntologie).



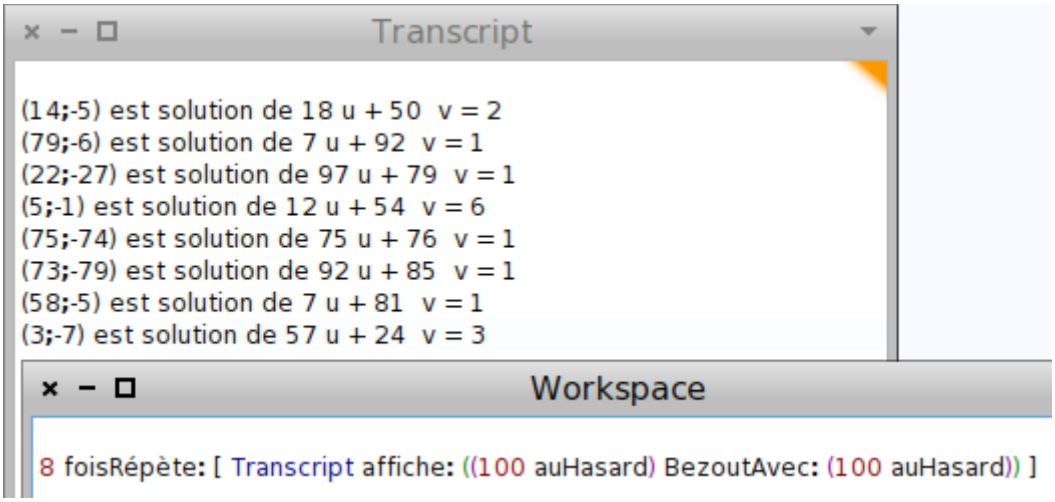
Remarque : Comme la méthode de Monte-Carlo est basée sur un comptage sur échantillon, on peut en déduire un intervalle de confiance pour l'intégrale ; ci-dessus on trouve [0,606 ; 0,666] qui ne contient pas la valeur exacte de l'intégrale...

21 Parmi les nombres ; sinon on peut remplir une urne avec des nombres réels comme par exemple avec *(0 jusque: 1 parPasDe: 0.001) commeSac*, et y choisir un nombre au hasard par la suite.

2) Arithmétique

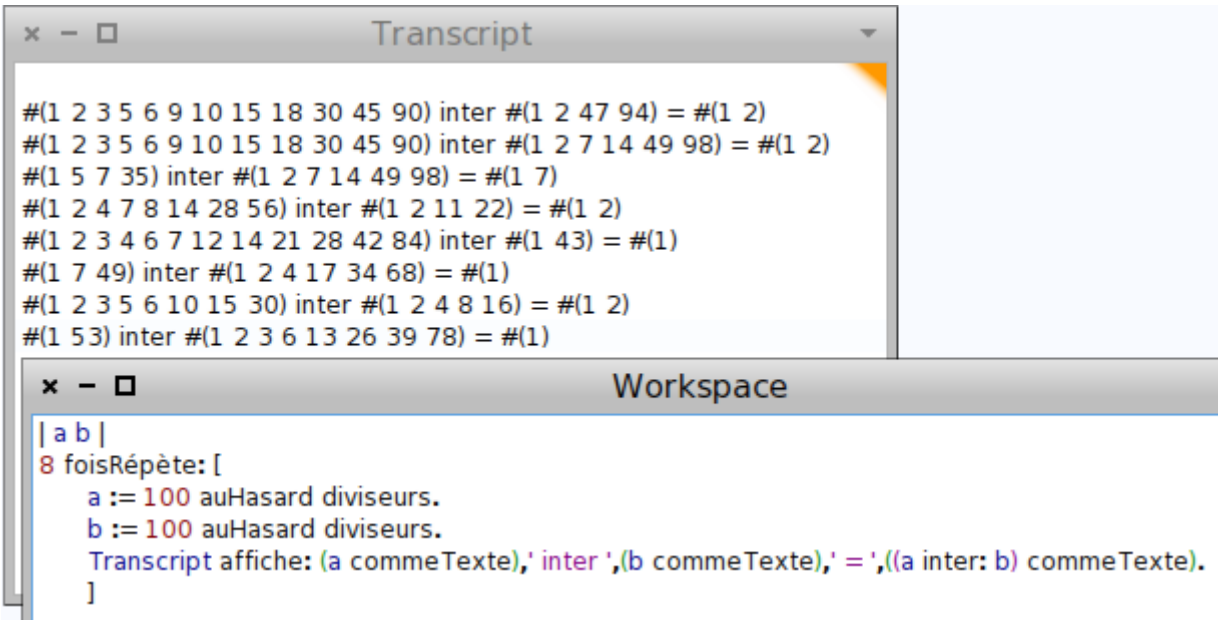
a) Utilisation du hasard pour expérimenter en arithmétique

Explorer des situations arithmétiques pour établir des conjectures se fait assez efficacement au hasard : On choisit des entiers au hasard pour voir si oui ou non ils vérifient une relation remarquable. Par exemple, la découverte expérimentale de l'existence d'une solution à l'équation de Bezout se fait avec le script suivant :

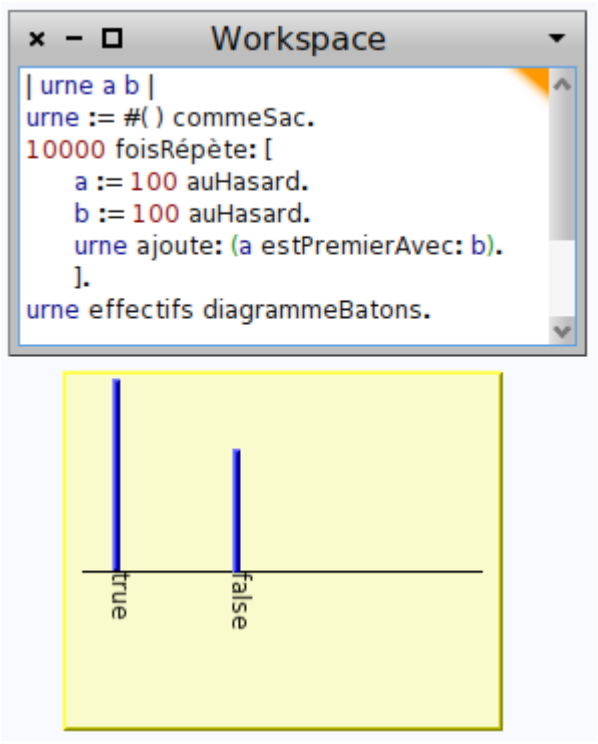


b)Phénomène stochastique en arithmétique

En examinant la liste des diviseurs de deux entiers choisis au hasard, et leur intersection, on constate que celle-ci est elle-même une liste de diviseurs : Celle de leur pgcd :

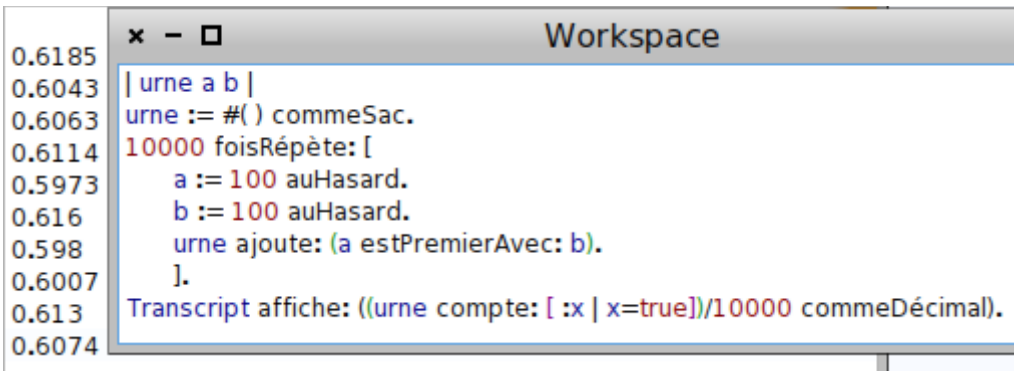


Cette expérience, destinée à faire découvrir expérimentalement la notion de nombres premiers entre eux, fait aussi constater que lorsqu'on choisit deux entiers au hasard, ils sont souvent premiers entre eux (ci-dessus, 3 fois sur 8). Une simulation sur 10 000 choix de couples d'entiers au hasard le confirme :

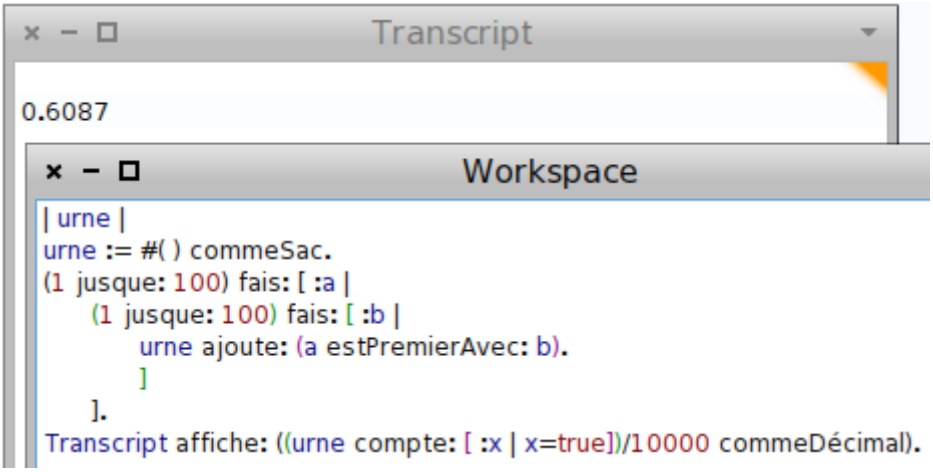


Le diagramme en bâtons montre que deux entiers choisis au hasard sont premiers entre eux dans plus de la moitié des cas. Pour en savoir plus, on peut afficher la fréquence et même plusieurs fois (pour éviter les errances de la fluctuation d'échantillonnage).

La probabilité que deux entiers choisis au hasard soient premiers entre eux semble voisine de 0,6 :



En fait, cette probabilité peut être calculée par une double boucle avec comptage :

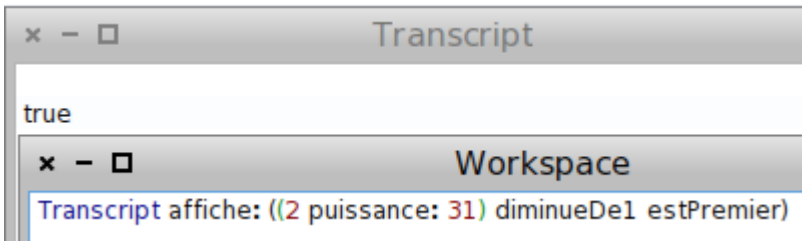


La probabilité que deux nombres compris entre 1 et 100 soient premiers entre eux, est donc 0,6087. On peut aisément (mais patiemment) modifier le script ci-dessus pour vérifier que la probabilité que deux entiers entre 1 et N soient premiers entre eux, tend vers une limite lorsque N tend vers l'infini[22]. De plus, on vérifie numériquement que cette limite est $\frac{6}{\pi} \approx 0{,}6079$.

c) Utilisation de l'arithmétique pour simuler le hasard

Une question sous-jacente à ce document depuis le début est « comment peut-on simuler le hasard avec un ordinateur qui est déterministe ? ». On est maintenant en mesure de créer son propre algorithme de génération de nombres pseudo-aléatoires ! La méthode choisie est celle des générateurs congruentiels de Lehmer, qui consiste en fait tout simplement à faire tourner une suite récurrente « arithmético-géométrique » dans un corps fini.

Par « suite arithmético-géométrique » on entend une suite récurrente obtenue par itération d'une fonction affine : $u_{n+1} = a \times u_n + b$. Très présentes au bac 2013, ces suites convergent si et seulement si -1<a<1. Sinon, elles tendent vers l'infini[23] … si on les considère comme définies sur $\mathbb{R}$. Pour peu qu'on les considère sur un corps fini, elles ne peuvent qu'être périodiques ou prépériodiques. L'arithmétique conduit à choisir le corps, a et b pour que la période soit la plus longue possible. Ici on va choisir un corps de la forme $\mathbb{Z}/p\mathbb{Z}$ où p est premier. Pour ne pas passer trop de temps à chercher un grand nombre premier, on choisit un nombre de Mersenne comme $2^{31}-1$ qui se trouve justement être premier :



Dans la suite on va donc effectuer tous les calculs modulo $2^{31}-1$. Pour déterminer a et b, on choisit l'algorithme suivant :

- On choisit un nombre a pas trop dur à fabriquer (date de naissance ou résultat d'un calcul

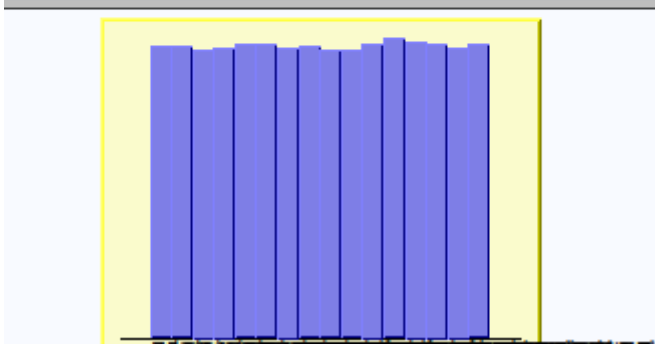
22 Mais le temps de calcul aussi tend vers l'infini...

23 Sauf si le premier terme est solution de x=ax+b, auquel cas la suite est constante. Ce cas est de probabilité nulle, on l'écarte donc en général (il est souvent indétectable numériquement)

assez simple) qui soit à la fois premier avec N (pas difficile : N est premier!) et assez grand pour que le caractère arithmético-géométrique de la suite ne soit pas trop évident (ici, $3^{15}+1$)

- On choisit b de façon similaire, sauf qu'il n'a pas besoin d'être premier avec N : Ici, 5^7 ;
- On fait une étude statistique pour voir si, pour un choix quelconque du premier terme, la suite semble bien aléatoire. Si c'est le cas, on a fini ; sinon on retourne à un des points précédents en essayant de deviner lequel parmi a et b, est responsable de la mauvaise qualité du générateur.

```
| N a b r urne |
N := (2 puissance: 31) diminueDe1.
a := (3 puissance: 15) augmenteDe1.
b := 5 puissance: 7.
r := 7.
urne := #( ) commeSac.
100000 foisRépète: [
    r  := (a * r + b) modulo: N.
    urne ajoute: r.
    ].
(urne effectifsEnClasses: 16) dessineHistogramme.
```



En fait, un histogramme sur 16 classes avec 100 000 itérations suggère que le générateur utilisé est bon. Il suffit alors de diviser le terme général par N pour simuler assez bien (et efficacement) un nombre suivant la loi uniforme sur [0;1[. Il peut être intéressant de comparer avec ce qui se fait dans certains logiciels libres, ne serait-ce que pour savoir si on a réellement inventé[24] un nouveau[25] générateur !

3) Modèle d'urnes d'Ehrenfest

Pour étudier la thermodynamique, les époux Ehrenfest (physiciens) ont imaginé que les boules de l'urne sont des atomes libres de se déplacer dans l'urne. Et ils ont fait communiquer (virtuellement) les moitiés de l'urne séparées par un diaphragme, un seul atome pouvant traverser le diaphragme à un moment donné. En bref, il s'agit de transférer une boule de loto portant un numéro choisi au hasard, de l'urne où elle est, vers l'autre urne. Au départ, toutes les boules sont dans une urne appelée *toi* et l'autre urne, appelée *tonFrère*, est initialement vide. Une fois qu'on a choisi un numéro de boule au hasard, on doit

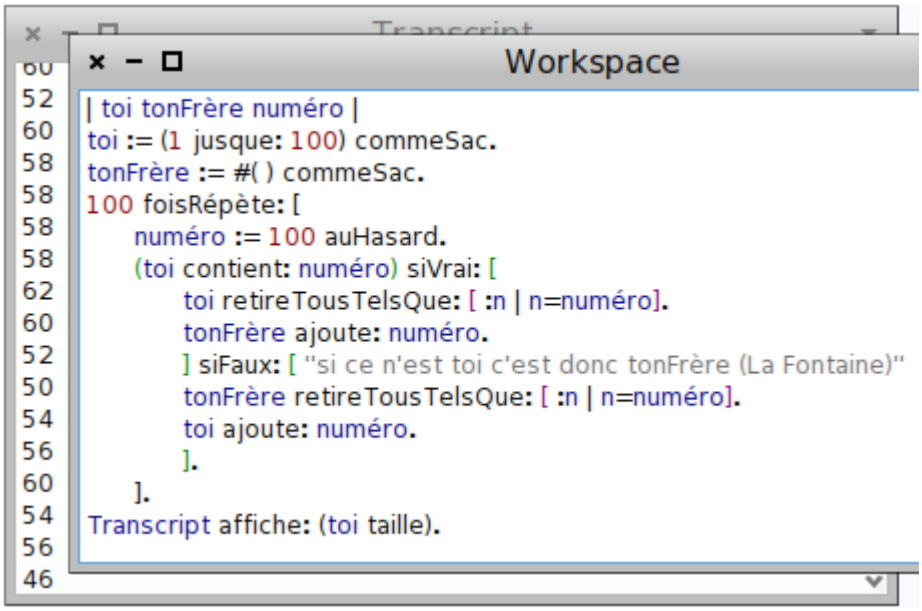
- enlever cette boule de l'urne qui la contient, avec *retireTousTelsQue* ;
- mettre ensuite la boule dans l'autre urne.

---

24 Avant de voir s'il mérite d'être appelé « générateur de Busser », on peut toujours le nommer « générateur de Lehmersenne » !

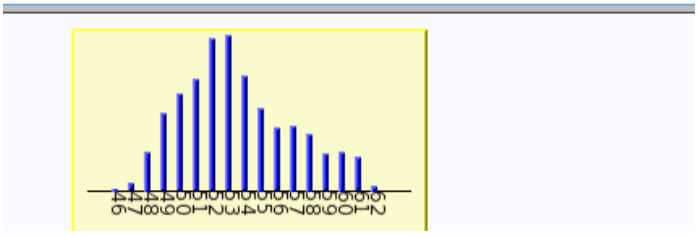
25 Python 3.2 par exemple, utilise le Mersenne twister, basé sur le nombre premier $2^{19\,937}-1$; mais ce n'est pas un générateur congruentiel, un « twist » y étant ajouté, d'où son nom.

Comme on ne sait pas avant de regarder dedans, quelle urne contient le numéro tiré, on doit effectuer un test et faire le transfert dans un sens ou l'autre, selon le résultat du test :



Le « control+D » qui lance le script a été effectué plusieurs fois ci-dessus, ce qui permet de voir que le nombre de boules encore présentes dans l'urne *toi* fluctue beaucoup. La répétition de l'expérience permet de le vérifier par un diagramme en bâtons :

```
| toi tonFrère numéro stats |
toi := (1 jusque: 100) commeSac.
tonFrère := #( ) commeSac.
stats := #( ) commeSac.
100 foisRépète: [
    numéro := 100 auHasard.
    (toi contient: numéro) siVrai: [
        toi retireTousTelsQue: [ :n | n=numéro].
        tonFrère ajoute: numéro.
        ] siFaux: [ "si ce n'est toi c'est donc tonFrère (La Fontaine)"
        tonFrère retireTousTelsQue: [ :n | n=numéro].
        toi ajoute: numéro.
        ].
    ].
400 foisRépète: [
    numéro := 100 auHasard.
    (toi contient: numéro) siVrai: [
        toi retireTousTelsQue: [ :n | n=numéro].
        tonFrère ajoute: numéro.
        ] siFaux: [ "si ce n'est toi c'est donc tonFrère (La Fontaine)"
        tonFrère retireTousTelsQue: [ :n | n=numéro].
        toi ajoute: numéro.
        ].
    stats ajoute: (toi taille).
    ].
stats effectifs diagrammeBatonsTrié.
```



L'idée peut être prolongée à d'autres activités comme l'urne de Polya (une urne où on rajoute à

chaque tirage une nouvelle boule avec *urne ajoute: (urne auHasard)* ; on peut donc parler de « tirage avec hyperremise »...) ou le problème des anniversaires, où on effectue un tirage sur des nombres allant de 1 à 365 – ou 366 pour les années bissextiles – et où on cherche d'éventuelles coïncidences ; voir la partie sur la théorie de l'échantillonnage en Seconde). On peut faire bien des choses rien qu'avec des boules dans une (ou plusieurs) urne !

4) Matrices

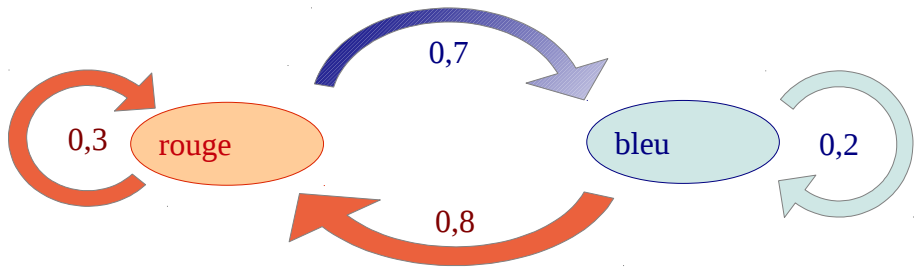
Pour finir en beauté, on va remplacer dans les urnes, les boules par des matrices. Le résultat est graphiquement parlant, vraiment intéressant[26]. Mais tout d'abord, on revient sur les urnes d'Ehrenfest, où on envisageait des changements d'état (le numéro de l'urne qui contient une boule est un état, à deux valeurs possibles) soumis au hasard. Dans un modèle d'urne d'Ehrenfest avec une seule boule, qui peut, ou non, changer d'urne à chaque top d'une horloge, on parle de chaînes de Markov. Cette notion a été utilisée dans plusieurs sujets de bac (ES spé maths puis S spé maths) en général pour étudier des mouvements de population entre deux régions, mais ici on va partir sur des considérations de probabilités conditionnelles, celles-ci pouvant très bien servir d'introduction à la multiplication de matrices :

a) Probabilités conditionnelles

Retour au village d'Hénon-Beaumin : Le maire rouge sortant a appris à la lumière des sondages que son voisin Monsieur Constant est particulièrement inconstant : Au début du sondage, il répond « rouge » ou « bleu » avec probabilité de 0,6 ou 0,4, et ensuite,

- si, à l'instant, Monsieur Constant a répondu « rouge », il change d'avis avec une probabilité de 0,8 et dans ce cas, ajoute « finalement je crois que je voterais plutôt bleu »
- si, à l'instant, Monsieur Constant vient de répondre qu'il vote bleu, il change d'avis avec une probabilité de 0,7 et, dans ce cas, affirme vouloir voter rouge quand même.

Monsieur Constant est inconstant, certes, mais à la longue, la probabilité qu'il vote rouge se stabilise à 8 chances sur 15, indépendamment de son choix initial. La situation peut être représentée par un graphe probabiliste :

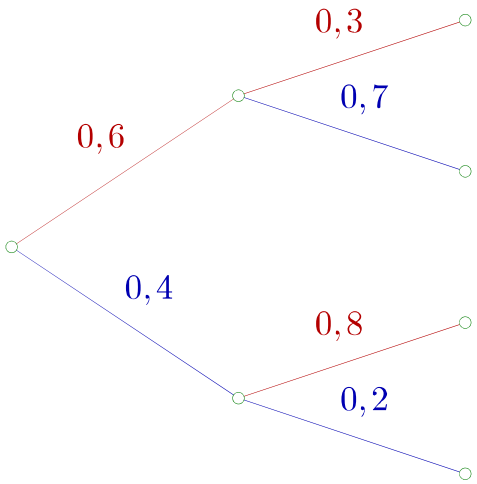


Mais aussi par une matrice de Markov, où on met les différentes probabilités conditionnelles ; ici c'est la matrice $M=\begin{pmatrix}0,3 & 0,8\\ 0,7 & 0,2\end{pmatrix}$. On crée une matrice dans MathsOntologie avec *M:= Matrice new*, puis on modifie ses coefficients en leur donnant une nouvelle valeur. Par exemple le 0,7 s'appelle *a21* (car il est à la deuxième ligne et première colonne) et se programme avec *M a21: 0.7*

Une matrice peut très bien être introduite comme commodité pour représenter plusieurs nombres

26 Mais pour l'instant il est difficile de dessiner une urne contenant des matrices, les boules étant plus faciles à dessiner. On peut imaginer par exemple des boules de billard avec des matrices à la place des nombres.

d'un coup. La multiplication d'une matrice par un vecteur étant introduite ensuite par l'arbre pondéré de cette situation :



En effet on y lit que la probabilité que Monsieur Constant vote rouge au bout de son premier changement d'avis est la somme des deux feuilles rouges, soit $0{,}6\times0{,}3+0{,}4\times0{,}8$. Si on représente les deux probabilités (vote rouge et vote bleu) dans un vecteur colonne $p=\begin{pmatrix}0{,}6\\0{,}4\end{pmatrix}$ (initialement, par la suite p va varier), alors la nouvelle valeur de ce vecteur colonne p sera, d'après ce qui précède, $\begin{pmatrix}0{,}3&0{,}8\\0{,}7&0{,}2\end{pmatrix}\begin{pmatrix}0{,}6\\0{,}4\end{pmatrix}=M\times p$. Et comme Monsieur Constant est vraiment inconstant, on recommence pour aboutir à des produits (en fait, ici, des puissances) de la matrice M multipliés par p. Dans MathsOntologie, pour multiplier M par p, on écrit *M transformeUnVecteur: p* :

```
(0.5000000119209289@0.4999999940395355)
(0.5500000107288361@0.4500000026822088)
(0.5250000172853471@0.47500000283122046)
(0.5375000193715097@0.46250000782310946)
(0.5312500239908697@0.4687500100955366)
(0.5343750271946195@0.4656250138767062)
(0.5328125311806803@0.46718751682899884)
(0.5335937847383327@0.4664062702329829)
```

Workspace

```
| M  p |
M := Matrice new a11: 0.3; a12: 0.8; a21: 0.7; a22: 0.2.
p := 0.6@0.4.
8 foisRépète: [
    p := M transformeUnVecteur: p.
    Transcript affiche: p.
    ].
```

Pour prouver la stabilité à long terme, on peut écrire *Transcript affiche: (M valeursPropresRéelles)* qui affiche l'ensemble { -0,5 ; 1 }. Ce qui veut dire que, diagonalisée, la matrice M devient $D=\begin{pmatrix}1&0\\0&-0{,}5\end{pmatrix}$ dont les puissances tendent vers $\begin{pmatrix}1&0\\0&0\end{pmatrix}$ lorsque l'exposant tend vers l'infini. Ainsi, si la matrice de passage est $P=\begin{pmatrix}8&1\\7&-1\end{pmatrix}$, on a $M=PDP^{-1}$, d'où $M^n=PD^nP^{-1}$ dont la

limite est $P\begin{pmatrix}1&0\\0&0\end{pmatrix}P^{-1}=\begin{pmatrix}\frac{8}{15}&\frac{8}{15}\\\frac{7}{15}&\frac{7}{15}\end{pmatrix}$: À la longue, Monsieur Constant vote rouge avec une probabilité de 8 chances sur 15, et bleu avec une probabilité de 7 chances sur 15.

Ces résultats peuvent être démontrés sans faire appel à la théorie des chaînes de Markov, en montrant par récurrence que la probabilité que Monsieur Constant vote rouge est une suite arithmético-géométrique et a donc une limite qu'on peut calculer. Voir à ce sujet l'exercice de spécialité du bac Pondichery 2013 (exercice 3).

b) Matrices aléatoires

L'exemple précédent aboutissait à une probabilité stable parce que la plus grande valeur propre de M (en valeur absolue) est égale à 1. On appelle ce nombre (la plus grande valeur propre, au signe près) le coefficient de Lipschitz de M. Donc ici, le coefficient de Lipschitz de M est 1. Si on met plusieurs matrices dans une urne, et qu'on les applique l'une après l'autre (après tirage au sort) répétitivement à un point P, on obtient une suite aléatoire de points dont chacun est l'image du précédent par la transformation associée à la matrice tirée au sort. Alors :

> **Théorème** (Barnsley) : *Si le produit des coefficients de Lipschitz de toutes les matrices qui sont dans l'urne est inférieur ou égal à 1, la suite des points obtenus se rapproche avec une probabilité 1 d'un ensemble du plan qui est stable par chacune des transformations.*

Cet ensemble s'appelle, on s'en doute, l'ensemble limite associé à l'urne (en plus, il est indépendant du choix initial du point définissant le nuage). Le théorème de Barnsley reste vrai si après avoir multiplié par une matrice M le vecteur colonne dont les coordonnées sont celles de M, on applique une translation[27]. Dans ce cas, les coordonnées du vecteur de cette translation sont stockées dans les coefficients a31 et a32 (cachés) de la matrice M, qui devient alors une matrice de 3 colonnes.

Alors, au lieu de mettre des matrices dans l'urne, on va carrément mettre des transformations[28], très précisément de 4 sortes :

- $\begin{pmatrix}x'\\y'\end{pmatrix}=\begin{pmatrix}0&0\\0&0{,}16\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix}$ de probabilité 0,1 (matrice A).
- $\begin{pmatrix}x'\\y'\end{pmatrix}=\begin{pmatrix}0{,}85&0{,}04\\-0{,}04&0{,}85\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix}+\begin{pmatrix}0\\1{,}6\end{pmatrix}$ de probabilité 0,76 (transformation B, sous forme d'une matrice 2 lignes 3 colonnes).
- $\begin{pmatrix}x'\\y'\end{pmatrix}=\begin{pmatrix}0{,}2&-0{,}26\\0{,}23&0{,}22\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix}+\begin{pmatrix}0\\1{,}6\end{pmatrix}$ de probabilité 0,07 (matrice C).
- $\begin{pmatrix}x'\\y'\end{pmatrix}=\begin{pmatrix}-0{,}15&0{,}28\\0{,}26&0{,}24\end{pmatrix}\begin{pmatrix}x\\y\end{pmatrix}+\begin{pmatrix}0\\0{,}44\end{pmatrix}$ de probabilité 0,07 (matrice D).

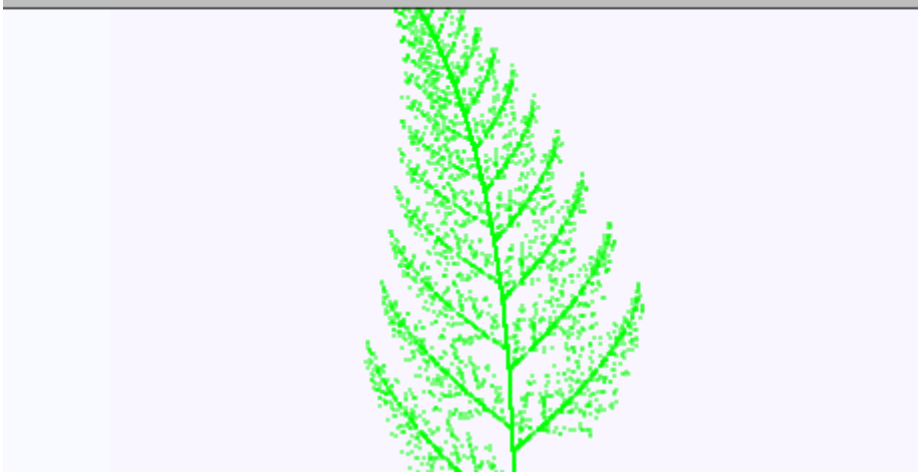
---

27 Un exemple célèbre est le dessin du triangle de Sierpinski par « IFS », les trois transformations utilisées étant des homothéties de rapport 0,5 et donc le produit des coefficients de Lipschitz est 0,125 qui est largement inférieur à 1 : Le dessin du triangle de Sierpinski est rapide.

28 Encore plus difficile à dessiner que des matrices dans une urne...

Le résultat obtenu est la célèbre fougère de Barnsley :

```
| A B C D p urne fougere pinceau |
A := Matrice new a22: 0.16.
B := Matrice new a11: 0.85; a12: 0.04; a21: -0.04; a22: 0.85; a23: 1.6.
C := Matrice new a11: 0.2; a12: -0.26; a21: 0.23; a22: 0.22; a23: 1.6.
D := Matrice new a11: -0.15; a12: 0.28; a21: 0.26; a22: 0.24; a23: 0.44.
urne := #() commeSac.
urne ajoute: A fois: 10.
urne ajoute: B fois: 76.
urne ajoute: C fois: 7.
urne ajoute: D fois: 7.
urne auHasard.
p := 1@1.
fougere := Morph new color: (Color magenta alpha: 0.02).
fougere extent: 600@400.
8000 foisRepete: [
    p := (urne auHasard) transformeUnPoint: p.
    pinceau := CircleMorph new borderColor: (Color green alpha: 0.5).
    pinceau borderColor: (Color green alpha: 0.5).
    pinceau extent: 2@2.
    pinceau position: (200@400)-(p*30).
    fougere addMorph: pinceau.
    ].
fougere openInWorld.
```



L'objet *fougere* est un dessin (ou « morph ») qui à la fin du script, est « ouvert dans le monde » (en gros, affiché). Ses dimensions sont 600 pixels et 400 pixels. L'objet *pinceau* est un disque vert (figurant un coup de pinceau) de rayon 2 pixels et dont la position dépend du point p. Pour transformer p par IFS, on fait juste *p:= (urne auHasard) transformeUnPoint: p* c'est-à-dire qu'on choisit une des transformations A, B, C et D dans l'urne et qu'on l'applique à p. On remarque que puisqu'il y a aussi des translations, la matrice ne transforme plus un vecteur mais un point.

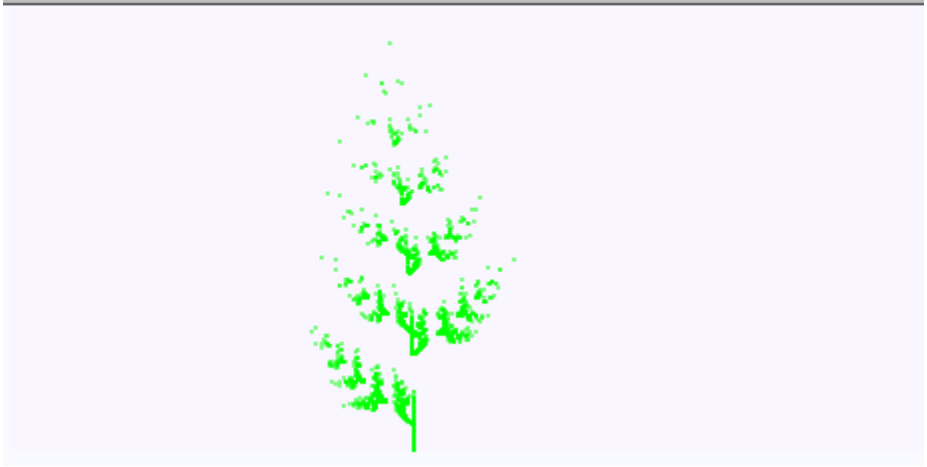
Les valeurs propres des matrices sont les suivantes :

matrices	**A**	**B**	**C**	**D**
Valeur propre 1	0	0,85-0,04i	0,21-0,244i	-0,288
Valeur propre 2	0,16	0,85+0,04i	0,21+0,244i	0,278

Les modules respectifs sont 0,16 0,85 0,322 et 0,288 (coefficients de Lipschitz). Leur produit vaut 0,0126 environ donc on est largement dans les conditions d'application du théorème de Barnsley.

Mais si les 4 transformations sont équiprobables, la fougère est un peu fantomatique en raison d'une accumulation des points près des extrémités :

```
| A B C D p urne fougere pinceau |
A := Matrice new a22: 0.16.
B := Matrice new a11: 0.85; a12: 0.04; a21: -0.04; a22: 0.85; a23: 1.6.
C := Matrice new a11: 0.2; a12: -0.26; a21: 0.23; a22: 0.22; a23: 1.6.
D := Matrice new a11: -0.15; a12: 0.28; a21: 0.26; a22: 0.24; a23: 0.44.
urne := #() commeSac.
urne ajoute: A; ajoute: B; ajoute: C; ajoute: D.
p := 1@1.
fougere := Morph new color: (Color magenta alpha: 0.02).
fougere extent: 600@400.
8000 foisRepete: [
    p := (urne auHasard) transformeUnPoint: p.
pinceau := CircleMorph new borderColor: (Color green alpha: 0.5).
pinceau borderColor: (Color green alpha: 0.5).
pinceau extent: 2@2.
    pinceau position: (200@400)-(p*30).
    fougere addMorph: pinceau.
    ].
fougere openInWorld.
```



Barnsley a donc amélioré son modèle de fougère en donnant des probabilités différentes aux 4 transformations, et pour cela, le modèle d'urne est encore une fois d'un grand secours.

Alain Busser
Lycée Roland-Garros
Le Tampon

liens :

- les lois binomiales et géométriques avec MathsOntologie
- les urnes de Polya et Ehrenfest avec MathsOntologie
- les dés dans l'urne
- le problème des anniversaires avec MathsOntologie
- un sujet de Brevet (onglet 2 pour les probabilités)
- le manuel de référence de MathsOntologie (il y a un chapitre complet sur les probabilités)
- le logiciel MathsOntologie