Loi binomiale

Combinaisons

Le nombre de manières de choisir k objets parmi n objets au total s'appelle le "nombre de combinaisons" de k objets parmi n [1] ; un algorithme rapide mais pas très simple pour le calculer est le suivant (en CoffeeScript) :

```
combinaison = (n,k) ->
    if 0 <= k <= n
        if k > n/2
            combinaison n, n-k
        else
            if k is 0
                1
            else
                arrondi n/k * combinaison n-1, k-1
    else
        0
```

On peut le tester avec alcoffeethmique en entrant `affiche [combinaison 10, k for k in [0..10]]`. Un exemple : Pour savoir combien de mains (5 cartes) on peut avoir au poker (52 cartes) on fait `affiche combinaison 52, 5`. Cette fonction, hors programme même en première, ne sert qu'à définir la suivante :

loi binomiale

La formule donnant la probabilité binomiale n'est pas non plus au programme [2], mais on s'en servira dans la suite du TP.

```
binomiale = (N,p,k) ->
    q = 1-p
    combinaison(N,k)*puissance(p,k)*puissance(q,N-k)
```

Alors pour calculer la probabilité d'avoir 3 boules rouges dans un tirage (avec remise) de 10 boules dans une urne contenant 40 % de boules rouges, on affiche `binomiale 10, 0.4, 3` [3]. Ces fonctions ont été incorporées à `alcoffeethmique` pour permettre de les utiliser en CoffeeScript. Le TP commence donc maintenant :

Début du TP : Intervalle de fluctuation

On voudrait un intervalle de fluctuation à 95 %, c'est-à-dire que la probabilité que la variable aléatoire « le nombre de succès est dans cet intervalle » soit égale à 0,95 au moins. Pour cela, on choisit de partager les 5 % restants en parts égales, soit 2,5 % en-dessous de l'intervalle et 2,5 % au-dessus. L'algorithme est alors le suivant (au programme de Première) :

1. Sur MathsOntologie, `k parmi: n` ; sur la calculatrice Ti, `n nCr k`
2. à tout hasard, sur la Ti, on va dans le menu `distrib` et on choisit `binomFdP`, puis on entre successivement la taille de l'échantillon, la probabilité de succès et le nombre de succès dans l'échantillon
3. donc sur la Ti, `binomFdP(10,0.4,3)`.

√ **Variables :**
- $N \in \mathbb{N}$ est la taille de l'échantillon
- $p \in [0;1]$ est la probabilité de succès avec un tirage ; ces deux variables sont en entrée.
- $bI \in \mathbb{N}$ et $bS \in \mathbb{N}$ sont les bornes de l'intervalle (à calculer)
- *somme* est une variable intermédiaire pour additionner des probabilités binomiales

√ **Traitement**
- Initialiser bI à 0, bS à N et *somme* à 0
- **Jusqu'à ce que** $somme \geqslant 0,025$
  | Ajouter à *somme* la probabilité d'avoir bI succès
  | Incrémenter bI
- **Fin du "Jusqu'à"**
- Décrémenter bI
- Remettre *somme* à 0
- **Jusqu'à ce que** $somme \geqslant 0,025$
  | Ajouter à *somme* la probabilité d'avoir bS succès
  | Décrémenter bS
- **Fin du "Jusqu'à"**
- Incrémenter bS

√ **Sortie**
- Afficher bI et bS

L'objet de ce TP est donc de programmer cet algorithme de calcul d'intervalle de fluctuation basé sur une loi binomiale, par exemple sous la forme d'une fonction de deux variables N et p, dont le corps serait celui-ci (attention à l'indentation) :

```
IF = (N,p) ->
    [bI, bS] = [0, N]
```

Le nom IF de la fonction est une abréviation pour « Intervalle de Fluctuation ».