COMPTE-RENDU PROBABILITES BATAILLE

GEOFFRE Yorick

Compte-rendu Probabilités bataille

# Préface

Ce compte-rendu détaille l’analyse probabiliste du jeu de Bataille. Cela comprend la programmation d’un simulateur du jeu pour en tirer des chiffres bruts, et l’analyse de ces chiffres pour valider ou contredire des hypothèses établies à partir des règles du jeu.

Table des matières

[Préface 1](#_Toc87113394)

[1) Introduction à la bataille et compréhension des règles de base 2](#_Toc87113395)

[2) Etude des règles de base et hypothèses 2](#_Toc87113396)

[3) Conception du simulateur 2](#_Toc87113397)

[4) Etude statistique 3](#_Toc87113398)

# Introduction à la bataille et compréhension des règles de base

Notre modèle de jeu de bataille se joue avec un paquet de 52 cartes (13 cartes de 4 couleurs différentes). Les couleurs ne sont pas importantes dans le jeu on conservera donc uniquement la valeur numérique de chaque carte.

Une fois les 52 cartes réparties de manière aléatoire entre les deux joueurs (26 cartes chacun), la partie peut commencer. Chaque joueur pose une carte, le joueur avec la carte la plus haute remporte la carte de l’adversaire.  
  
En cas d’égalité, la carte de chaque joueur + celle d’après sont mises en « bataille », la bataille continuera avec chaque égalité successive, et la personne qui aura une carte plus haute durant une bataille y mettra fin et remportera toutes les cartes mises en bataille.

# Etude des règles de base et hypothèses

Après une lecture compréhensive des règles, on peut dégager plusieurs détails sur ce jeu :

* Les joueurs n’ont aucune décision à prendre à travers le jeu, une fois que le « deck » est trié et que les joueurs ont reçu leurs cartes, la partie est décidée d’avance.
* Il existe plusieurs façons de gérer la bataille, notamment si on retrie les cartes collectées en fin de bataille ou non.

En termes d’hypothèses on pourrait se demander si :

* Avoir des cartes plus puissantes au début a une incidence directe sur le nombre de victoires d’un joueur (en corrélation avec le détail 1)
* Les différents types de bataille ont une incidence sur la durée de partie et/ou les victoires

# Conception du simulateur

Mon simulateur prend avantage des capacités de Scilab en tant que langage à paradigme fonctionnel. Notamment celle à passer des fonctions en tant qu’argument à d’autres fonctions. Cela me permet de créer des fonctions « génériques » telles que :

[jeu1,jeu2]=distribution(méthodeDistribution)

Qui distribue les cartes selon une méthode de distribution (fonction qui gère la randomisation des cartes du paquet) donnée en argument. Ainsi que :

[] = LanceBataille(nbParties, typeBataille, typeDistribution)

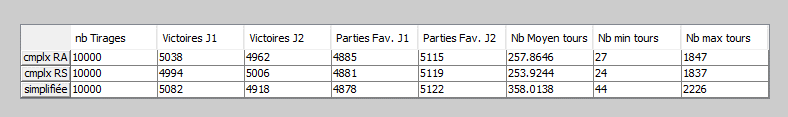
Qui prend en argument le nombre de parties (tirages) souhaités, le type de bataille souhaité (fonction : batailla complexe, simplifiée, ou complexe avec remise aléatoire) et enfin le type de distribution, qui sera passé à son tour à distribution ().

Ici le but du simulateur est de nous donner des chiffres à partir desquels il sera possible de faire une étude statistique. On l’utilisera aussi pour tester nos hypothèses.

# Etude statistique

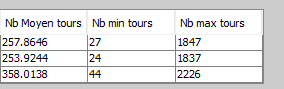
Etude générale

Pour commencer cette étude, on peut traiter les données sorties par le programme directement :



Sur 30 000 tirages en tout, on a testé la bataille complexe avec remise aléatoire, la bataille complexe avec remise standard et la bataille simplifiée.

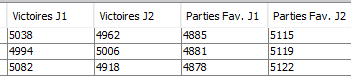
On a collecté la durée moyenne, minimale et maximale (en tours) des parties.



Cela nous montre qu’en moyenne, le nombre de tours pour une bataille complexe RA est à 258 tours, RS à 254 et simplifiée à 358. La différence entre RA et RS est assez négligeable mais ce n’est pas le cas de la bataille simplifiée. En moyenne elle dure 358 tours ce qui est largement supérieur aux batailles complexes.

Le nombre maximum et minimum de tours reflète également cette tendance, avec RS < RA < simplifiée. On peut dégager de ces données que le comportement de la bataille simplifiée sur le temps est différent de celui des batailles complexes.

On observe que le nombre de victoires pour la bataille complexe avec RA varie plus du 50/50 que celle sans RA, cela pourrait être dû au caractère aléatoire supplémentaire qu’offre la version avec RA.



Globalement les 3 versions sont assez proches l’une de l’autre en termes de distribution des victoires.

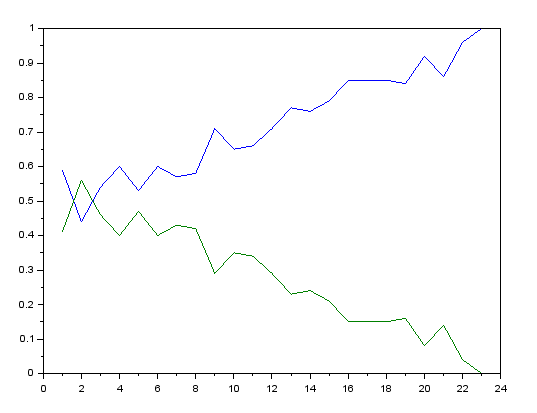
Ce qui peut aussi être remarqué est que les batailles ayant un caractère aléatoire supplémentaire (Complexe RA et simplifiée), ont également un rapport moindre à la « force » des cartes qui sont distribuées. Là où les victoires de la bataille complexe standard sont assez proches

Etude injuste

Dans cette étude, on utilise une fonction de distribution des cartes « injuste », c’est-à-dire que pour un chiffre donné (ici de 1 à 23), elle va vérifier qu’autant de cartes dans le paquet de J2 sont inférieures à celles de J1. Si elle trouve qu’une carte n’est pas inférieure, elle l’échange avec celle de J2 [exemple : J1 carte n°3 = 11 <-> J2 carte n°3 = 4 seront échangées en faveur de J1).

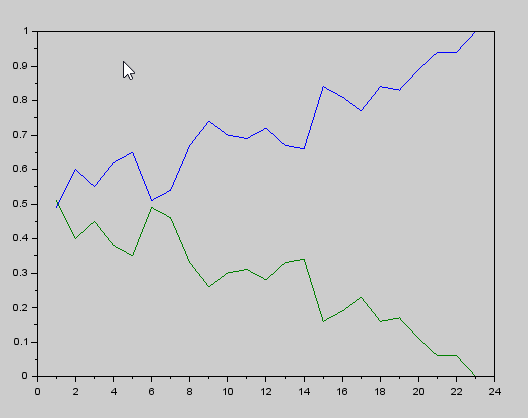
On a testé les 3 différentes batailles avec indice injuste de 1 à 23, sur 100 essais par cas :

**Bataille Complexe RA**



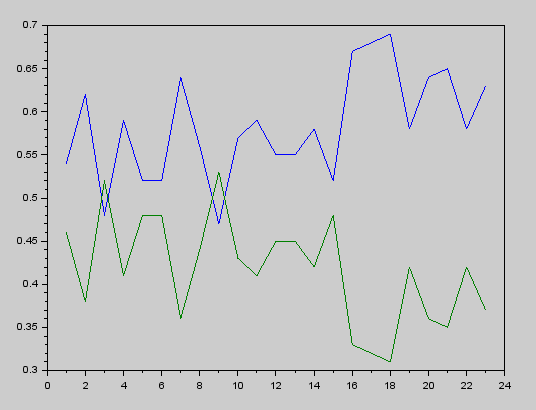
On voit ici les victoires de J1 (en bleu) et celles de J2 (en vert). Il est clair que la force des cartes a ici un impact majeur. Avec la force des cartes de J1 qui augmentent (plus de cartes sont échangées en sa faveur), ses victoires augmentent.

**Bataille Complexe RS**



On observe ici la même tendance qu’avec la bataille complexe RA.

**Bataille Simplifiée**

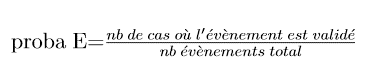


Dans le cas de la bataille simplifiée, les choses sont différentes, le graphe apparaît beaucoup plus chaotique et inconclusif sauf après que l’index injuste passe au-dessus de 16. Cela est clairement dû au caractère aléatoire de cette bataille qui donne une chance égale aux deux joueurs de remporter une bataille. Cette partie de l’étude met en évidence la différence de comportement de ces deux batailles.

Etude calculatoire – fonction de distribution

Les deux études précédentes étaient purement indicatives, et ne reposaient pas sur des calculs, uniquement sur les données directes.

Pour commencer, on va vérifier l’intégrité de la fonction de distribution « juste » et rapide. Pour se faire, il faut se rappeler que la probabilité qu’un évènement E soit valide sera

 (Créé via Latex)

Donc ici, la probabilité qu’une carte soit tirée (quelconque) sera de :



On considère donc la proba de tirer n’importe quelle carte de et de tirer une carte avec une valeur spécifique car les couleurs et symboles des cartes ne sont pas simulés (il y a 13x4 cartes dans ce jeu)

On va faire 2 000 tirages aléatoires de decks, et compter combien de fois J1 a eu chaque carte :



Prenons X la variable aléatoire que le joueur 1 ait reçu un roi (13), sur **200** tirages (200x52 = 10 400 cartes tirées en tout) (ce qui donne = **5 200** cartes par joueur) il l’a obtenu 388 fois.

Donc, en théorie, chaque carte devrait sortir = **400** (ce qui est déjà proche de nos résultats)

X suit ici une loi binomiale X ~ Bin(5200, p) [on a choisi 5200 car les 200 tirages sont faits entre 2 joueurs et 52\*200/2 = 5200, ça simplifie les calculs]

Nos valeurs dans ce cas :

Espérance : **m** = 5200\*p ⬄ m = 5200\* = **400** (trouvé précédemment)

Variance : **σ** = √(5200\*p(1 - p)) ⬄ = √(5200\* (1 - )) = **19,21537846** ≈ 19,21

P est la probabilité de tirer une carte (une des 13) dans une infinité de lancers, on a ici la loi *exacte*

Pr est la probabilité calculée si notre hypothèse (la fonction de distribution rapide est correcte) est valide.

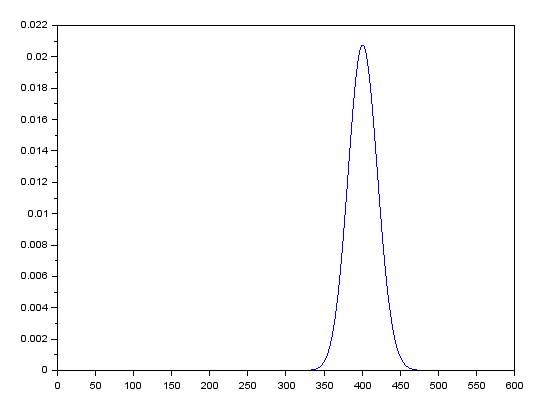
Pr a été partiellement calculé précédemment avec Pr = (pour un seul tirage)

On formule donc l’hypothèse H0, qui est celle où la fonction de distribution rapide est correcte. Ici, P=Pr=

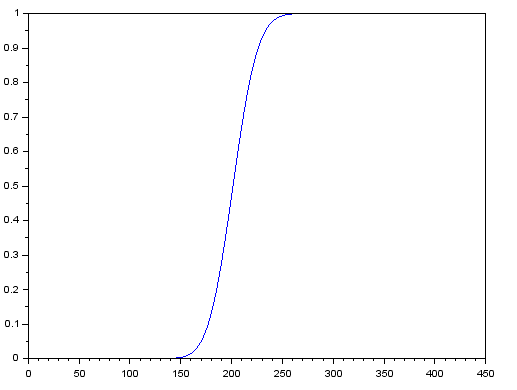
L’hypothèse inverse est H1, où la fonction est erronée et P !=

En supposant que H0 est vraie et que X suit X ~ N(**m**, **σ**) :

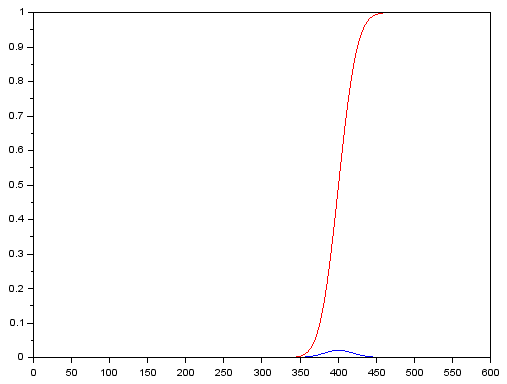
Scilab nous permet d’obtenir la courbe suivante (répartition de probabilités) [ligne 77 de projet.sce] [paramètres X ~ Bin(, 5200) ]:



La courbe de la somme cumulée est de :



Pour bien mettre les deux en relation d’échelle : voilà un graphe qui contient les 2 en même temps :



Pour être certain à 98% (pourcentage d’application standard en risque) que la fonction de distribution est correcte, on cherche dans le tableau de la fonction gaussienne Φ(z) = 0,98

On obtient donc **z** = **2,88**

Donc, on a : **d** = **z**\***σ** = 2,88\*19,21 = **55,32**

Ainsi, le seuil de rejet de H0 à 98% de confiance est le suivant :

**344,68 >= X <= 455,32** ⬄ ne rejette pas H0

**344,68 <= X || X >= 455,32** ⬄ rejette H0 en faveur de H1

Cela valide notre fonction de distribution de cartes, car le nombre de cartes pour chaque type de cartes par joueur n’a jamais dépassé les seuils de H0.

Etude calculatoire -