|  |
| --- |
|  |
| TP de Simulation |
| *Simulation stochastique d’une population de lapins* |
|  |
| **BARBESANGE Benjamin & GARÇON Benoît** |
| **novembre 2015** |

|  |
| --- |
|  |

Encadré par David Hill

Table des matières

[Présentation du TP 2](#_Toc435642940)

[Remerciements 2](#_Toc435642941)

[Objectif du TP 2](#_Toc435642942)

[Analyse des données 3](#_Toc435642943)

[Vie du lapin 3](#_Toc435642944)

[Naissance des lapins 3](#_Toc435642945)

[Solution retenue 4](#_Toc435642946)

[Organisation du code 4](#_Toc435642947)

[Simulation 4](#_Toc435642948)

[Vieillissement de population 4](#_Toc435642949)

[Naissance de lapereaux 5](#_Toc435642950)

[Récupération des données 6](#_Toc435642951)

[Résultats 7](#_Toc435642952)

[Test sur une période de 20 ans 7](#_Toc435642953)

[Conclusion 8](#_Toc435642954)

Présentation du TP

# Remerciements

Nous tenons à remercier M. Mazel pour les informations qu'il a pu nous apporter sur les différentes réflexions probabilistiques. Nous remercions également M. Hill pour son encadrement lors des séances de TP et l'aide qu'il a pu nous apporter.

# Objectif du TP

Ce TP s'inscrit dans le cadre des cours de simulation de 2ème Année. Le but est de créer un modèle de simulation de lapins un peu plus réaliste que les précédents modèles effectués par la fonction de Fibonacci.

L'objectif est également de produire une simulation dont les performances permettent d'obtenir un nombre de lapin de l'ordre du milliard en un temps raisonnable.

Analyse des données

Avant de débuter la simulation de la population de lapins, il faut se renseigner afin d'avoir une simulation aussi réaliste et performante que possible.

# Vie du lapin

L'espérance d'un lapin peut aller au maximum jusqu'à 15 années pour les plus coriaces. C'est pourquoi dans notre simulation nous ne gèrerons les lapins que jusqu'à leurs 15 ans ; année à laquelle ils mourront directement.

De plus, il est notable que les lapins n'ont pas le même taux de survie lorsqu'ils sont jeunes et lorsqu'ils sont plus âgés. Nous avons effectué la répartition suivante pour les taux de survie :



Figure 1 - Taux de mort des lapins en fonction de leur âge

# Naissance des lapins

Généralement, les femelles peuvent faire entre 4 et 8 portées par an avec plus de chance d'en effectuer entre 5 et 7. Chacune de ces portées peut donner naissance entre 3 et 6 lapereaux.

La période de gestation de la lapine se trouve entre 28 et 33 jours ; nous simplifierons et prendrons un mois comme période de gestation. En termes de maturité, celle-ci est atteinte entre 5 et 8 mois. Pour simplifier la simulation, nos lapins sont capables de se reproduire dès lors qu'ils entrent dans leur 10ème mois.

Solution retenue

# Organisation du code

Notre programme de simulation s'organise autour de 5 fichiers :

* main.cpp : code principal de la simulation
* ClasseLapins.cpp : représentation de lapins du même age
* LapinManager.cpp : code de gestion de toutes les classes de lapins et de la simulation

Les fichiers ClasseLapins.cpp et LapinManager.cpp sont accompagnés que leur header c++.

Vous pourrez trouver la documentation complète des fichiers en ouvrant le fichier *index.html*. Les fichiers du programme seront également trouvables et sont accompagnés d'un Makefile permettant de recompiler le programme sur votre machine.

# Simulation

Dans cette partie, nous détaillerons un peu plus les procédés utilisés pour effectuer la simulation. Afin d'obtenir de bonnes performances, nous avons utilisées certaines lois de probabilités étant adaptées pour les grandes populations.

## Vieillissement de population

En ce qui concerne le vieillissement de la population, si la population de la classe n'est pas trop importante (inférieure à 100), nous faisons un tirage selon une loi de Bernoulli ayant pour paramètre le taux de survie pour la classe. Nous le faisons pour chaque individu de la classe. Si ce tirage réussit, l'individu reste en vie, sinon il meut.

Dans le cas où nous avons beaucoup d'individus dans une classe (plus de 100), utilisons une loi normale d'espérance:

Et d'écart type :

Grace au cours de probabilité de première année, nous avons pu constater qu'une Loi Binomiale avec une grande population convergeait vers une loi Normale. C'est pourquoi ici, pour une grande population, nous avons décidé de choisir une loi Normale.

## Naissance de lapereaux

La naissance des lapereaux se répartie sur les 8 premiers mois de l'année. La probabilité de reproduction des femelles pour les 2 premiers mois ainsi que les 2 derniers mois de cette période est de 0.5. Sur les mois restant, la probabilité de reproduction est de 1 car le minimum de portées par ans est de 4.

Nous pouvons interpréter les mois sans reproduction comme étant logique puisque nous entrons dans une période hivernale, ce qui implique que nous aurons seulement des décès et aucune naissance.

En ce qui concerne les lois utilisées, nous avons toujours une loi Normale pour une classe dont l'effectif est supérieur à 100. L'espérance est :

Et l'écart type utilisé est :

Lorsque la population de la classe est inférieure à 100, nous faisons une épreuve de Bernoulli pour chaque femelle de la classe avec pour paramètre la probabilité de portée.

La détermination du sexe s'effectue en utilisant les mêmes lois dans le mêmes conditions (taille de la population de la classe) que dans le cas des naissances. L'espérance de la loi Normale dans ce case sont, pour l'espérance :

L'écart type :

Le paramètre de la loi de Bernoulli dans ce cas est 0.5 car nous estimons qu'il y a autant de chances d'obtenir un lapin mâle qu'un lapin femelle.

# Récupération des données

Pour lancer la simulation avec des paramètres personalisés pour le nombre d'années à simuler et un nombre de réplications, il faut lancer le programme dans le terminal et spécifier dans cet ordre : le nombre d'années à simuler, puis le nombre de réplications à effectuer. Dans le cas où rien n'est spécifie, nous simulons simplement 20 années en une seule fois.

Nous pouvons avoir également différents modes d'affichage. Pour lancer la simulation et obtenir les affichages à l'écran, il faut lancer le programme dans la console et ajouter l'option '-v'. On peut également utiliser l'option '-f' afin d'avoir les résultats dans le fichier *lap.out* qui sera créé dans le répertoire du projet. Ces deux options sont cumulables.

## Structure du fichier de sortie

En utilisant l'option –f du programme, un fichier de sortie *lap.out* est généré. Ce fichier contient les résultats de toutes les simulations lancées. Une ligne se compose des données suivantes :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| [ 9/2007 ] | 43526 | 4553 | 5277 | 21888 | 21638 |

Les champs contiennent :

* La date à laquelle on a recueillies les données. On va trouver le mois ainsi que l'année. Notons que l'année de début de simulation est 2000.
* La population actuelle à la date indiquée.
* Le nombre de naissances
* Le nombre de morts
* Le nombre de mâles
* Le nombre de femelles

Chaque donnée est séparée par une tabulation. Ainsi si on ouvre directement ce fichier avec un tableur, les données seront disposées correctement et on peut tracer des graphes.

Résultats

Le fait d'avoir implémenté la sauvegarde des données dans un fichier permet trace d'exécution du programme mois par mois. Ceci nous permet d'avoir des informations, comme le nombre de mâles, de femelles, la population courante ou encore le nombre de naissances. Il est possible.

Lors de ces simulations, étant donné que nous débutons avec un mâle et une femelle d'un an, il arrive que ceux deux lapins meurent rapidement sans avoir eu le temps de se reproduire. La population de lapins reste donc à 0.

# Simulation sur 20 ans et 50 réplications

Pour tester notre programme, nous procédons à une simulation sur 20 ans, avec 50 réplications.

La commande exécutée est la suivante :

./GameOfRabbitsI 20 50

Le programme nous retourne la population finale de chaque simulation. Ainsi, en entrant ces valeurs dans un tableur, nous pouvons calculer la moyenne et l'intervalle de confiance relatifs à ces données.

Figure - Intervalle de confiance sur 50 réplications

En lançant le programme avec la commande *time*, on peut obtenir le temps d'exécution du programme qui est ici de 1.640s.

Nous pouvons remarquer que l'intervalle de confiance est plutôt large. Nous allons donc effectuer plus de réplications.

# Simulation sur 20 ans et 5000 réplications

La commande exécutée est la suivante :

/GameOfRabbitsI 20 5000

Afin de teste les performances du programme, nous avons lancé une simulation de 20 ans avec 5000 réplications. Nous avons pu obtenir ces résultats en 2 minutes et 5 secondes. Nous pouvons donc établir une meilleure approximation des résultats.



Figure - Intervalle de confiance avec 5 000 réplications

Avec plus de réplications, nous observons que l'intervalle de confiance est largement réduit.

# Analyse des performances

Pour analyser les performances, nous avons exécutées la simulation sur un nombre d'années de plus en plus grand. Il en résulte le graphique suivant :



Figure - Temps d'exécution en fonction du nombre d'années

D'après ce graphique, on peut voir que le temps de calcul dans les premières années augmente de plus en plus et n'est pas linéaire. Ceci est dû au fait que la population n'est pas très élevée au début et que l'on simule individuellement chaque lapin.

Une fois que la population est assez importante et que l'on simule avec la loi normale, on observe que le temps d'exécution devient linéaire.

Conclusion

Grace à ce TP, nous avons pu avoir une meilleure approche de la simulation. Nous avons bien compris que l'analyse est la partie dominante qui va guider nos choix par rapport à l'orientation de la solution.

Dans notre cas, nous voulions pouvoir simuler une très longue période, c'est pourquoi nous avons simplifié certaines données comme le temps de gestation des femelles ou encore le temps de maturité des lapereaux.

Il en résulte donc ici un programme dont les performances sont bonnes, comme ont pu le montrer les résultats. Après avoir effectuées des comparaisons avec d'autres groupes ayant des simulations plus pointues, il est apparu que nos données sont proches des leurs.

De plus, ce TP a permis de mieux nous familiariser avec les fonctions de base en c++ permettant d'utiliser des lois ou encore le générateur de nombres aléatoires Mersenne Twister.