ISFA2 — 2024-2025 — projet de simulation

Date limite : vendredi 20 décembre 2024, 23h59

Étude le l'impact de la dépendance dans un modèle d'assurance moto.

1 Modèle

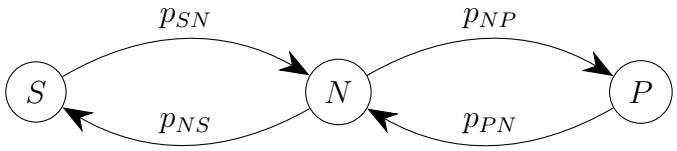
1.1 Contexte

On considère une assurance concernant les accidents de la route des motards. La pluie ayant un impact important sur les risques encourrus par les motards, on décide de modéliser la météo.

1.2 Météo

On décide de modéliser la météo des jours successifs d'une période donnée par une chaîne de Markov à trois états : S (*soleil*), N (*nuages*), P (*pluie*).

L'état initial de cette chaîne est toujours l'état *soleil*, et les probabilités de transition sont décrites par le graphe suivant.



1.3 Accidents

On suppose qu'un jour donné, un motard a un accident avec une probabilité qui dépend de la météo : si ce jour là la météo est au soleil, la probabilité qu'il ait un accident est p_S, et de même pour les deux autres météos possibles avec les probbailités p_N et p_P.

Un motard ne peut pas avoir plus d'un accident par jour.

1.4 Remboursements

On suppose que le montant (dans une unité arbitraire) d'un remboursement de sinistre lors de l'accident d'un motard suit la loi $\mathbb{P}_r$ dont la densité est proportionnelle à

$$f^*(x) = \mathbf{1}_{\mathbb{R}_+}(x) \exp(-\eta \ln(\alpha + |x - x_0|)) .$$

où $\alpha > 0$, $x_0 > 0$ et $\eta > 3$ sont des constantes.

Pour l'évaluation qui sera faite de votre code, on aura $x_0 < 3$ et $\alpha < 3$.

1.5 Remboursement total

On considère une période de 365 jours, sur laquelle on assure N motards.

On note R le remboursement total à effectuer sur toute cette période et pour tous les motards.

1.6 Indépendance

On considère que les montants de remboursements sont indépendants, et indépendants de la météo.

Consernant les motards, on fera deux hypothèses différentes :

- **Hypothèse A** : les motards circulent tous dans la même région, et ont donc une météo commune.
- **Hypothèse B** : les motards circulent dans des réhions différentes, et ont donc des météos indépendantes. On peut donc associer à chaque motard une chaîne de Markov décrivant la météo dans sa région, et les différentes chaînes de Markov sont indépendantes.

2 Résultat attendu

On cherche à approximer, par une méthode de Monte Carlo, l'espérance conditionnelle

$$m(s) = \mathrm{E}\left(R - s \mid R > s\right) ,$$

où s est un seuil donné.

Les simulations devront être effectuées avec le logiciel R, en faisant, dans un deuxième temps, appel à une partie de code sous C/C++ (cette technique sera expliquée dans un autre cours).

Vous choisirez, pour vos essais, des valeurs de paramètres qui vous semblent raisonnables.

2.1 Concours d'efficacité

Pour ce concours, chaque groupe devra proposer un code R pur (sans utiliser de code C/C++), qui effectue de la manière la plus efficace possible les opérations suivantes :

- Simulation de n valeurs suivant la loi $\mathbb{P}_r$.
- Approximation de $m(s)$ sous l'hypothèse A.
- Approximation de $m(s)$ sous l'hypothèse B.

Le programme `R` ne devra utiliser aucune bibliothèque spécifique (ne pas utiliser `library`).

Afin de comparer votre code à celui des autres groupes, vous pourrez déposer un fichier R sur `https://simulation.isfa.fr/`. Les codes déposés seront régulièrement évalués et les scores obtenus seront publiés. Plusieurs membres du groupe peuvent déposer un code, mais un seul code sera évalué chaque jour pour chaque étudiant.

Attention ! À la date limite, c'est le dernier code soumis par un membre du groupe sur la plate-forme qui sera retenu pour le classement final des groupes, et l'attribution d'une note. En plus de la mesure de son efficacité, la validité du code sera bien entendu vérifiée.

Afin d'être évalué, le code déposé doit suivre les consignes suivantes :

2.1.1 Nom

Le fichier déposé doit se nommer `simul.R`.

2.1.2 Syntaxe

Le fichier doit être syntaxiquement correct : on doit pouvoir appeler la commande R `source("simul.R")` sans erreur.

Le fichier doit être encodé en UTF8 (en cas de doute, n'utilisez aucun caractère accentué ou spécial dans votre code, même dans les commentaires).

Par ailleurs, le code ne doit faire aucun calcul long, ne doit produire aucun graphique : il doit uniquement définir les fonctions demandées (et éventuellement des fonctions intermédiaires).

2.1.3 Simulation de la loi $\mathbb{P}_r$

Le fichier doit déclarer une fonction `rremb` qui simule la loi $\mathbb{P}_r$, prenant les paramètres suivants :

- le nombre `n` de valeurs à simuler ;
- une liste `params` contenant les paramètres intervenant dans le modèle :
  - `params$alpha` $= \alpha$
  - `params$x0` $= x_0$
  - `params$eta` $= \eta$

Cette fonction doit renvoyer un vecteur de taille `n` contenant les valeurs simulées suivant le loi $\mathbb{P}_r$.

Lors de l'appel de votre fonction, la valeur de `n` utilisée sera ajustée pour obtenir un temps d'exécution proche de 5 s (ce n'est pas à vous de faire cet ajustement).

2.1.4 Approximation de $m(s)$

Le fichier doit déclarer deux fonctions `msa` et `msb`, qui calculent une approximation de $m(s)$ sous chacune des hypothèses A et B par une méthode de Monte Carlo. Elle doit prendre en compte les paramètres suivants :

- `n.simul` est le nombre de simulations à effectuer ;
- une liste `params` contenant les paramètres intervenant dans le modèle :
  - `params$alpha` $= \alpha$
  - `params$x0` $= x_0$
  - `params$eta` $= \eta$
  - `params$pacc` $= (p_S, p_N, p_P)$
  - `params$ptrans` $= (p_{SN}, p_{NS}, p_{NP}, p_{PN})$
  - `params$N` $= N$
  - `params$s` $= s$

Les fonctions `msa` et `msb` doivent renvoyer une liste contenant l'approximation de la valeur demandée, ainsi qu'un intervalle de confiance à 95 % associé à cette approximation, sous la forme

```
list(ms=...,demi.largeur=...)
```

La valeur de `n.simul` sera ajustée pour obtenir un temps d'exécution proche de 30 s (ce n'est pas à vous de faire cet ajustement).

2.1.5 Utilisation d'un code C à partir de R

Le fichier R doit également déclarer une fonction `msb.c`, qui fonctionne de la même façon que `msb`, mais qui fait appel à du code C en utilisant `Rcpp`.

Attention : la commande qui compile le code C avec `cppFunction` doit être appelée à l'intérieur de `msb.c` et non dans le code général.

Le but est d'améliorer la vitesse de simulation, du mieux que vous pouvez, sans forcément reprogrammer tout en C.

2.2 Influence des paramètres

Étudier l'influence des paramètres x_0, η, p_P, p_{NP} et s sur la valeur de $m(s)$. Le sens de cette influence était-il attendu ?

2.3 Rapport écrit et soutenance

Chaque groupe devra déposer sur `https://moodle.univ-lyon1.fr`, avant la date limite, un compte-rendu au format PDF exclusivement, sous le nom `GXX.pdf` (en remplaçant `XX` par le numéro du groupe). Le compte-rendu devra aborder les éléments suivants :

- Organisation du travail en groupe : quel a été le rôle de chaque étudiant, comment vous êtes-vous coordonnés ?
- Quelles sont les méthodes de simulation que vous avez implémentées ?
- Quelles sont les techniques de programmation qui vous semblent intéressantes dans votre code ?
- Quelle est la partie du code que vous avez codée en C/C++, et pourquoi ? Cela vous a-t-il fait gagner en efficacité ?
- Comment avez-vous vérifié la validité de votre code (tests avec certaines valeurs de paramètres, calculs intermédiaires, représentations graphiques, etc.) ?
- Détailler les résultats de votre analyse de l'influence des paramètres décrite en 2.2.
- Quelles études basées sur ce modèle vous paraîtraient pertinentes ? Quelles évolutions de ce modèle proposeriez-vous, afin de le rendre plus réaliste ?