Méthode de Monte-Carlo & Application aux processus aléatoires

Examen final

É. N. S. des Mines de Nancy / Génie Industriel & Mathématiques Appliquées

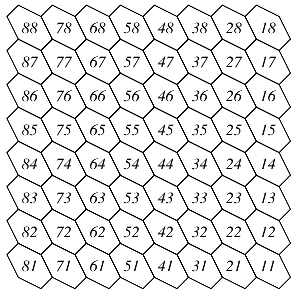
31 mars 2016

Ce sujet comporte 7 pages.

Problème I – MoHex

Ce problème s'intéresse à un jeu aux règles très simples, appelé Hex. *Les règles de ce jeu (que vous n'avez, en réalité, pas besoin de comprendre en détail) sont expliquées ci-après :*

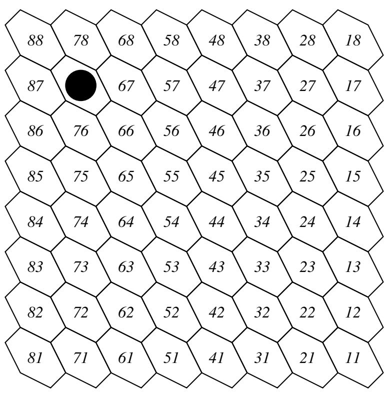
Le jeu se joue sur un plateau *de* 8×8 *cases ayant la forme suivante :*



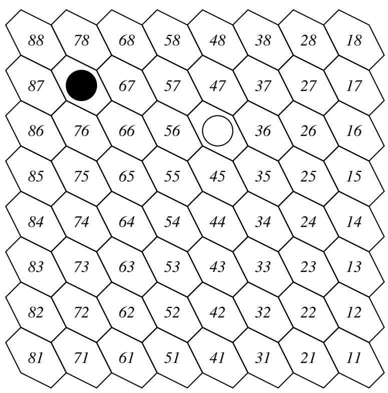
Plateau vide (`HexAAAA`*)*

(Les numéros qui figurent sur chaque case servent uniquement à identifier de quelle case on parle dans la suite du texte).

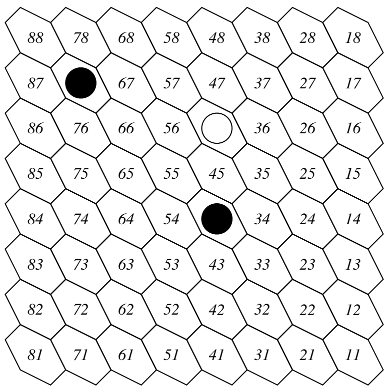
Deux joueurs s'affrontent, appelés resp. noir *et* blanc. *Par convention, on associe la valeur* 0 *à la couleur noire, et* 1 *à la couleur blanche. Au début du jeu, le plateau est vide. Puis, chacun son tour en commençant par* noir, *les joueurs vont poser un pion de leur propre couleur (dont ils disposent d'un stock illimité) sur une des cases libres de leur choix. Par exemple, les premiers coups de la partie pourraient être les suivants :*



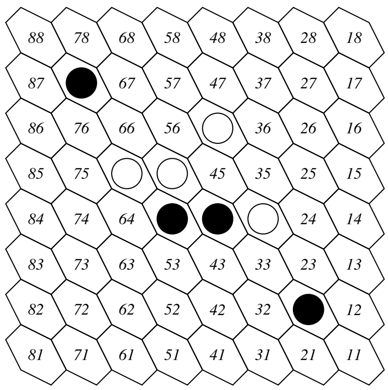
Premier coup : noir *joue en* 22
(`HexIYH1`*)*



Second coup : blanc *joue en* 35
(`HexIYH2`*)*



Troisième coup : noir *joue en* 55
(`HexIYH3`*)*



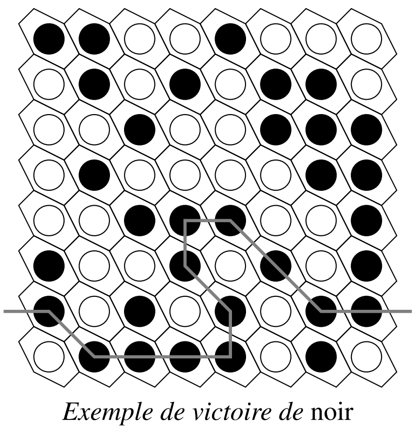
Après 8 coups
(`HexIYH8`*)*

Lorsque le plateau est entièrement rempli, la partie est terminée. On peut alors montrer qu'il se produit une et une seule des deux situations suivantes :

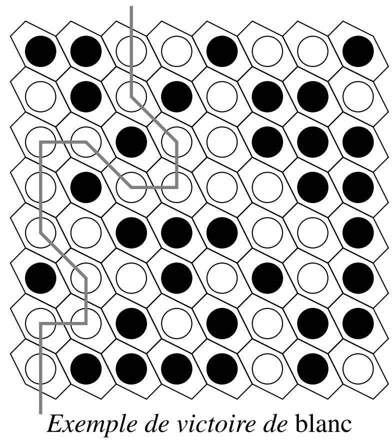
— *Ou bien il est possible de trouver un chemin de cases adjacentes "noires" (càd. de cases sur lesquelles sont posées des pions noirs) qui relie le bord de gauche du plateau au bord de droite du plateau ;*

— *Ou bien il est possible de trouver un chemin de cases blanches qui relie le bord du haut du plateau au bord du bas.*

Dans le premier cas, c'est noir *qui a gagné ; et dans le second cas, c'est* blanc.



Exemple de victoire de noir
(`HexOTMS`*)*



Exemple de victoire de blanc
(`HexCNCO`*)*

Dans ce problème, nous nous mettons dans la peau d'une équipe de développeurs en intelligence artificielle qui souhaite apprendre à l'ordinateur à jouer de façon pertinente à Hex. *Il s'avère qu'une stratégie relativement astucieuse, et simple à expliquer à un ordinateur* [∗]*, est la suivante :*

— *Une* configuration *étant donnée (c.-à-d. la donnée d'un plateau sont certaines cases sont noires, d'autres blanches, et d'autres encore vides), ainsi que l'identité du joueur dont c'est le* tour *de jouer* [†]*, la* valeur *de cette situation est définie comme la probabilité, si les joueurs finissent leur partie à partir de cette situation en jouant de façon complètement aléatoire parmi les coups autorisés, que le vainqueur final soit* blanc.

[∗]. Ce qu'on appelle une *heuristique*.

[†]. Nous parlerons de *situation* pour désigner la donnée conjointe d'une configuration du plateau et d'un tour de jeu.

— *Parmi tous les coups qu'il peut envisager, l'ordinateur choisira ensuite celui qui conduit à la situation dont la valeur est la plus faible possible s'il joue en tant que* noir, *resp. la plus élevée possible s'il joue en tant que* blanc.

Ainsi, le cœur du logiciel consistera à être capable de déterminer la valeur d'une situation donnée, càd. d'estimer la probabilité qu'une partie qui se terminerait aléatoirement à partir ce cette situation conduise à une victoire de blanc. *Comme la loi de probabilité correspondante est tout de même très compliquée, il n'est pas possible de calculer la valeur d'une situation exactement ; on est donc contraint de recourir à la méthode de Monte-Carlo.*

*Avec cet énoncé vous ont été envoyés des codes qui font une grande partie du travail spécifique au jeu d'*Hex. *Dans ces codes, "noir" est représenté par le nombre* 0*, resp. "blanc" par le nombre* 1*, resp. "vide" par le nombre* -1 *; les cases sont représentées à l'aide des numéros figurant sur le diagramme* `HexAAAA`*, et une configuration donnée (càd. savoir quelle case est remplie ou non avec quelle couleur) est représentée par un vecteur-colonne de taille 64, selon un procédé dont vous n'avez pas besoin de connaitre le détail. J'ai notamment implémenté les six fonctions suivantes, donc vous n'avez pas à comprendre le fonctionnement interne :*

— `PoserPion` *prend trois arguments : une configuration* F *(codée donc par un 64-vecteur colonne), un numéro de case* n*, et une couleur de pion* c*. Elle renvoie la configuration* F' *dans laquelle, par rapport à* F*, on a posé un pion de couleur* c *sur la case numéro* n*. Si la case* n *n'était pas libre dans la configuration* F*, la fonction affiche un message d'erreur.*

— `OterPion` *prend deux arguments : une configuration* F *et un numéro de case* n*. Elle renvoie la configuration* F' *déduite de* F *en ôtant le pion posé sur la case numéro* n*. (Si la case était vide, la fonction affiche un message d'erreur).*

— `QuiAGagne` *prend en argument une configuration dans laquelle le plateau doit complètement rempli, et renvoie l'identité du joueur qui a gagné. Si le plateau n'était pas complètement rempli, la fonction affiche un message d'erreur.*

— `CasesVides` *prend en argument une configuration* F *et renvoie un vecteur-ligne contenant l'identité des cases vides dans la configuration* F*. (Par conséquent,* `length(CasesVides(F))`*) renvoie le nombre de cases vides dans la configuration* F*.*

— `QuelleCouleur` *prend en argument une configuration* F *et un numéro de case* n *; et renvoie* 0 *si la case numéro* n *est occupée par un pion noir,* 1 *si elle est occupée par un pion blanc, et* -1 *si elle est vide.*

— `PlateauVide` *ne prend aucun argument et renvoie (le 64-vecteur colonne correspondant à) la configuration de départ, càd. la configuration dans laquelle toutes les cases sont vides. Ainsi,* `CasesVides(PlateauVide)` *renvoie le 64-vecteur de tous les numéros de cases existants.*

En chargeant le fichier `Hex.mat`*, vous disposerez aussi de variables dont chacune correspond à (la 64-vecteur-colonne qui code) la position du plateau décrite sur les différents schémas qui illustrent l'énoncé.*

8 pt **Question 1 –** Écrire une fonction `valeur` prenant trois paramètres : une configuration du plateau, un second argument disant de quel joueur c'est le tour, ainsi qu'un nombre de simulations pour la méthode de Monte-Carlo. Cette fonction affichera la probabilité que *blanc* gagne si l'on termine la partie aléatoirement, assortie de son intervalle de confiance à 95 %.

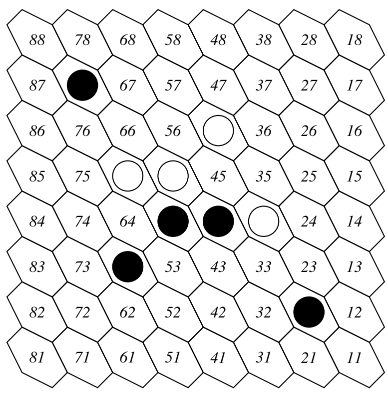
5 pt **Question 2 –** Expliquer pourquoi ce qui est important, ce n'est pas tant d'être capable

d'estimer les valeurs, que les *différences* entre les valeurs correspondant à deux situations différentes.

9 pt **Question 3 –** Écrire une fonction `valeur_coupl` prenant en argument deux configurations du plateau, l'identité du joueur dont c'est le tour, et un nombre de simulations pour la méthode de Monte-Carlo, et qui estime les valeurs respectives pour chaque configuration, en faisant en sorte que l'incertitude sur la *différence* des valeurs soit aussi faible que possible (mais on ne demande pas d'afficher l'intervalle de confiance de la différence entre les deux valeurs).

Variante – Vous pouvez aussi essayer de répondre à cette question en écrivant une fonction dans laquelle le premier argument est une matrice à 64 lignes de nombre de colonnes arbitraires, dont chacune des colonnes représente une certaine configuration, puis le second et le troisième argument sont resp. le joueur dont c'est le tour de jouer et le nombre simulations demandés, et qui effectue un couplage commun entre *toutes* les situations données pour estimer leurs valeurs respectives, de façon à ce que les diverses différences entre valeurs aient une incertitude aussi faible que possible. Pour cette variante, le barême de la question sera augmenté de moitié.

On cherche maintenant à estimer précisément la valeur de la situation où c'est à blanc *de jouer dans la configuration* `HexNXWW` *dessinée ci-dessous. Pour améliorer la précision de notre estimation, nous allons utiliser la variable de contrôle $A/8$, où A est le nombre de cases parmi les cases numéros* 52*,* 62*,* 66 *et* 76 *que le joueur* blanc *aura remplies au cours de la fin de partie aléatoire simulée.*



Configuration `HexNXWW`

3 pt **Question 4 –** Démontrer que l'espérance de A vaut exactement $112/55$.

5 pt **Question 5 –** Écrire une fonction `valeur_ctrl` estimant la valeur de la situation "configuration `HexNXWW`, au tour de *blanc*" à l'aide de la variable de contrôle $A/8$.

3 pt **(Question 11) –** À quoi pourrait ressembler une technique de conditionnement pour la détermination de la valeur d'une situation ? (On ne demande pas d'implémenter de code correspondant).

Problème II – Prévisions météo

Ce problème s'intéresse à des questions autour de la simulation et de la méthode de Monte-Carlo qui peuvent émerger lorsqu'on cherche à établir des prévisions météorologiques. Bien entendu, les véritables modèles servant à élaborer les prévisions météo sont *d'une complexité effroyable ; aussi, dans ce cadre de ce problème, nous ne considèrerons qu'un modèle très simplifié* [‡] *dans lequel l'état de la météo n'est décrit que par* 3 *variables. Ces variables sont notées respectivement* X*,* Y *et* Z*, et évoluent au cours du temps : on note ainsi* X_t *la valeur de la première variable au temps t, etc. Les variables* X, Y, Z *sont exprimées selon des unités arbitraires sans dimension physique, susceptibles de prendre n'importe quelles valeurs positives ou négatives ; précisons cependant que :*

— X *est liée à la vitesse du vent ; dans les situations concrètes, elle est généralement comprise entre* -20 *(vent faible) et* $+20$ *(vent fort).*
— Y *est liée à la température ; dans les situations concrètes, elle est généralement comprise entre* -25 *(chaleur) et* $+25$ *(froid).*
— Z *est liée à la quantité de nuages dans le ciel ; dans les situations concrètes, elle est généralement comprise entre* $+5$ *(ciel dégagé) et* $+50$ *(ciel couvert).*

Dans notre modèle, le système évolue selon le système d'équations différentielles stochastiques couplées suivant :

$$\begin{cases} dX_t = \lambda s(Y_t - X_t)dt + dW_t^X \\ dY_t = \lambda(rX_t - Y_t - X_tZ_t)dt + dW_t^Y \\ dZ_t = \lambda(X_tY_t - bZ_t)dt + dW_t^Z, \end{cases}$$

où $(W_t^X, W_t^Y, W_t^Z)_{t\geqslant 0}$ *est un mouvement brownien tridimensionnel non standard dont la matrice de covariance par unité de temps est une certaine matrice* $\mathbf{\Gamma}$*, càd. que* (W^X, W^Y, W^Z) *est un processus aléatoire markovien à valeurs dans* $\mathbf{R}^3$ *tel que, pour tous* $t < u$*,* $(W_u^X - W_t^X, W_u^Y - W_t^Y, W_u^Z - W_t^Z)$ *suit une loi normale centrée de matrice de covariance* $(u-t)\mathbf{\Gamma}$*.*

Les paramètres sont les suivants, où 'j*' est le symbole du jour (qui sera notre unité de durée temporelle de référence) :* $\lambda = 0{,}1\ \mathrm{j}^{-1}$*,* $s = 10$*,* $r = 28$*,* $b = 2{,}7$*, et*

$$\mathbf{\Gamma} = \begin{pmatrix} \gamma & -\alpha\gamma & 0 \\ -\alpha\gamma & \gamma & 0 \\ 0 & 0 & \gamma \end{pmatrix},$$

avec $\gamma = 50\ \mathrm{j}^{-1}$ *et* $\alpha = 0{,}25$*. Tous ces paramètres peuvent être chargés comme variables globales grâce au fichier* `meteo.mat`*, les noms respectifs de* $\lambda, s, r, b, \Gamma, \gamma, \alpha$ *étant* `lambda_`, `s_`, `r_`, `b_`, `GAMMA_`, `gamma_`, `alpha_`.

2 pt **(Question 12) –** Justifier que, si $\mathbf{A}$ est une matrice déterministe réelle 3×3 et sont ξ_1, ξ_2, ξ_3 trois variables normales standard indépendantes, alors $\mathbf{A}(\xi_1, \xi_2, \xi_3)^\mathsf{T}$ est un vecteur gaussien de matrice de covariance $\mathbf{A}\mathbf{A}^\mathsf{T}$.

7 pt **Question 6 –** Écrire une fonction `simulation_meteo` simulant le processus stochastique décrit ci-dessus. Cette fonction prendra en argument, dans cet ordre, les valeurs initiales X_0, Y_0 et Z_0 des variables météorologiques, ainsi qu'un temps T. Elle simulera l'évolution du processus sur une durée T en partant des valeurs (X_0, Y_0, Z_0), et affichera les valeurs X_T, Y_T et Z_T prises par les variables à la fin de simulation. (La fonction en elle-même ne renvoyant aucune valeur). On utilisera un pas de temps de longueur $5 \cdot 10^{-4}$ j.

Variante – Vous pouvez aussi choisir de répondre à la variante suivante de la question : plutôt que le quatrième argument soit constitué d'un seul temps T, il s'agirait d'un vecteur $(T_1, \ldots, T_k)$ (la taille k du vecteur n'étant pas fixée, mais choisie par l'utilisateur de la fonction lorsqu'il l'appelle,

[‡]. Il s'agit cependant d'un vrai modèle, qui a un sens physique et qui a été étudié scrupuleusement par les scientifiques !

typiquement entre 1 et 10) avec $0 < T_1 < T_2 \ldots < T_k$. Dans ce cas, la fonction affiche successivement, pour chaque i, la valeur prise par les variables au temps T_i (les différents T_i correspondant à la même simulation, bien entendu), ce qui permet ainsi à l'utilisateur de voir les prédictions de la simulation pour différents instants futurs. Si vous choisissez de répondre à cette variante de la question, le barème sera augmenté de moitié.

INDICATION – *Pour vous aider à vérifier la validité de votre réponse, il peut vous être utile de savoir que les variations des valeurs de X, Y et Z entre deux instants séparés d'une demi-journée sont généralement comprises entre 0 et 10 unités pour X, resp. entre 0 et 15 unités pour Y et pour Z.*

J'ai expliqué précédemment que Z était lié à la quantité de nuages dans le ciel. Plus précisément, l'ensoleillement S reçu au sol est lié à Z par la formule suivante :

$$S = S_* \exp(-\kappa e^{Z/Z_*}),$$

avec $S_* = 1\,000 \text{ W} \cdot \text{m}^{-2}$, $Z^* = 15$ et $\kappa = 0{,}1$ (resp. `Set_`, `Zet_` *et* `kappa_` *à partir du fichier* `meteo.mat`*).*

4 pt **Question 7 –** On suppose qu'à l'heure actuelle, $(X_0, Y_0, Z_0) = (15, 10, 35)$. On note m l'espérance (exacte) de S_T à partir de ces conditions initiales au temps $T = 3$ j. Écrire un programme estimant m par la méthode de Monte-Carlo, en précisant l'intervalle de confiance à 95 %, à partir de 32 simulations. Ce programme, s'appellera `meteo_MC_S`, ne prendra ni ne renverra rien, et affichera les valeurs calculées.

5 pt **Question 8 –** Nonobstant le caractère simplifié du modèle d'évolution [§], expliquer pourquoi, dans le contexte de la prévision météorologique “de base”, il n'est pas intéressant de faire un nombre très grand de simulations.

On suppose maintenant que les conditions initiales sont $(X_0, Y_0, Z_0) = (15, 20, 10)$ et qu'on s'intéresse à l'évolution de X sur le jour à venir. Dans ce cas, les simulations montrent que, au cours des deux jours à venir, il se trouve que X_t est susceptible de prendre des valeurs particulièrement grandes, facilement supérieures à 25. Or, lorsque X dépasse la valeur 32, il s'agit d'une tempête*, qui est susceptible d'avoir des conséquences désastreuses, de sorte que l'office météorologique est tenu d'alerter les populations dès lors que le risque d'une tempête au cours des deux jours à venir dépasse 5 %. Mais nous supposons ici que les moyens de calcul informatiques de l'office météorologique sont tels [¶] qu'il lui est concrètement impossible de lancer plus de 32 simulations en un temps raisonnable. Mais avec un nombre de simulations aussi faible, l'estimation d'un risque aussi faible que 5 % par la méthode naïve devient extrêmement floue : il est donc nécessaire de réduire la variance.*

L'office météo souhaite appliquer une réduction de la variance par échantillonnage préférentiel. Pour ce faire, il considère la loi d'évolution modifiée suivante : entre l'instant actuel ($t = 0$) et l'instant $t = 0{,}5$ j $=: \tau$ (`tau_`*), on ajoute une terme dans l'équation*

[§]. Je veux dire par là que vous pourriez remarquer que, dans la vraie vie, les modèles d'évolution météorologique sont très compliqués et que leur calcul prend beaucoup de temps ; on peut être donc conduit à réaliser un arbitrage entre un modèle très couteux en temps de calcul qui donne un résultat précis, mais dont on ne peut faire qu'une ou deux simulations, et un modèle simplifié qu'on peut faire tourner un grand nombre de fois, mais dont les résultats sont moins précis. Ici, je vous demande de traiter la question comme si le modèle d'évolution donné par l'énoncé était parfaitement exact.

[¶]. Dans la vraie vie, le problème est que les simulations sont extrêmement compliquées à réaliser, de sorte que même avec d'énormes moyens informatiques, les capacités de simulation de l'office météorologique restent assez limitées…

d'évolution de X :

$$\begin{cases} dX_t = \lambda s(Y_t - X_t)dt + \delta dt + dW_t^X \\ dY_t = \lambda(rX_t - Y_t - X_tZ_t)dt + dW_t^Y \\ dZ_t = \lambda(X_tY_t - bZ_t)dt + dW_t^Z, \end{cases}.$$

avec $\delta = 10\,\mathrm{j}^{-1}$ (`delta_`*). Puis, après le temps τ, l'évolution du système reprend son cours normal* [‖].

Il est possible de démontrer que dans un tel cas, la loi du système modifié est à densité par rapport à la loi du système non modifié, avec une densité relative valant

$$\exp\left(\frac{\delta}{\gamma(1-\alpha^2)}(A + \alpha B - \delta\tau/2)\right),$$

où A et B sont les variables aléatoires suivantes :

$$A := X_\tau - X_0 - \lambda s \int_0^\tau (Y_t - X_t)dt,$$
$$B := Y_\tau - Y_0 - \lambda \int_0^\tau (rX_t - Y_t - X_tZ_t)dt.$$

4 pt **(Question 13) –** Démontrer la formule ci-dessus concernant la densité de la loi modifiée par rapport à la loi originelle.

8 pt **Question 9 –** Implémenter la méthode d'échantillonnage préférentiel suggérée ci-dessus pour estimer, à partir de 32 simulations, la probabilité qu'une tempête éclate dans les deux jours à venir.

4 pt **(Question 14) –** En fait, le choix d'échantillonnage préférentiel suggéré ci-dessus s'avère très peu pertinent... À quoi pouvons-nous le voir ? Quelles sont les raisons, selon vous, qui expliquent que ce choix soit mauvais ?

☛ *Pour les deux questions suivantes, on utilisera une valeur de γ (et donc de Γ) différente de celle du reste de l'énoncé, à savoir $\gamma = 0{,}2$ (variables* `gammabis_` *et* `GAMMABIS_` *dans* `meteo.mat`*).*

Oublions mainteant ces histoires d'échantillonnage préférentiel pour nous pencher sur un autre problème. En réalité, lorsqu'on cherche à prédire la météo, il existe aussi une incertitude due à la mesure des variables au temps initial. L'office météo cherche à mesurer l'impact systématique dû à une erreur de mesure de $0{,}01 =: \varepsilon$ dans la mesure initiale de Z a sur la valeur de Y au temps T ; c'est-à-dire que, notant m_T l'espérance de Y_T partant de la condition initiale (X_0, Y_0, Z_0) et m'_T l'espérance de Y_T partant de la condition initiale $(X_0, Y_0, Z_0 + \varepsilon)$, il cherche à déterminer $m'_T - m_T$.

7 pt **Question 10 –** Écrire, d'une part un programme `MCdiff_naif` pour estimer $m'_T - m_T$ sans utiliser de technique de réduction de la variance, d'autre part un programme `MCdiff_coupl` pour estimer $m'_T - m_T$ en utilisant la technique de couplage idoine. Dans les deux cas, les programmes prendront pour arguments, d'une part la valeur de T, et d'autre part le nombre de simulations demandées. On prendra $(X_0, Y_0, Z_0) = (15, 10, 35)$.

4 pt **(Question 15) –** À nombre de simulations égal, regarder la précision de l'estimation de $m'_T - m_T$ pour chacune des valeurs suivantes de T, avec et sans couplage : ¼, ½, $1, 2, 4, 8, 16, 32, 64, 128$. Commenter.

[‖]. Ainsi, la dynamique du nouveau système change selon qu'on est avant ou après l'instant τ : dans le formalisme mathématique, on dirait qu'il s'agit d'un système markovien *inhomogène en temps*.