Projet 1 - Chaînes de Markov

Eléments de processus stochastiques

Floriane Magera
Romain Mormont
Fabrice Servais
Troisième bachelier en sciences de l'ingénieur

Année académique 2013-2014

Table des matières

Chapitre 1

Question 1

1.1 Étude du modèle de base

Analyse du graphe A_1

1) Nous avons représenté le graphe à partir de la matrice d'adjacence A_1 sur la FIGURE 1.1

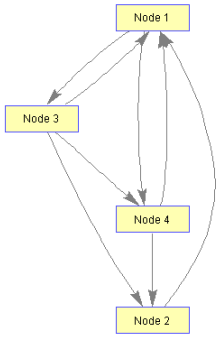


FIGURE 1.1 – Graphe de A_1

2) Avant de calculer la matrice de transition, il est nécessaire de caractériser la marche aléatoire. Autrement dit, il faut définir les poids/probabilités que nous appliquerons aux arêtes du graphe sur lequel le surfeur va évoluer. Nous avons déduit de l'énoncé du projet que les différentes possibilités de quitter un nœud devaient être **équiprobables** et nous utiliserons donc cette hypothèse dans la suite du rapport.

Suite à ce choix, la formation de la matrice de transition est très simple. Si l'on note A la matrice d'adjacence, alors il suffit d'appliquer la formule suivante pour calculer l'élément $Q(i,j)$:

$$Q(i,j) = A(i,j) \times \frac{1}{n} \sum_{j=1}^{n} A(i,j)$$

Cette formule permet de placer à 0 les éléments de Q représentant une transition impossible et de placer à une certaine probabilité les autres éléments de Q (toute probabilité non nulle d'une ligne de Q étant équiprobable comme attendu). Nous avons représenté le diagramme d'états à la FIGURE 1.1

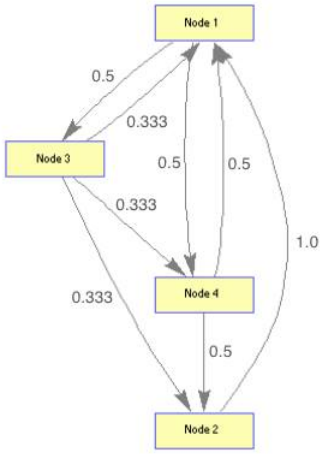


FIGURE 1.2 – Diagramme d'états à partir de A_1

3) Nous avons choisi un nombre de pas $t = 20$. Le cas où le surfeur démarre aléatoirement sur le graphe est représenté par une distribution initiale π_0 uniforme et le cas où il démarre d'une page fixe est représenté par une distribution initiale π_0 où toutes les probabilités sont nulles sauf celle située l'index correspondant au nœud de départ. L'évolution des probabilités dans les deux cas est donnée sur le FIGURE 1.3.

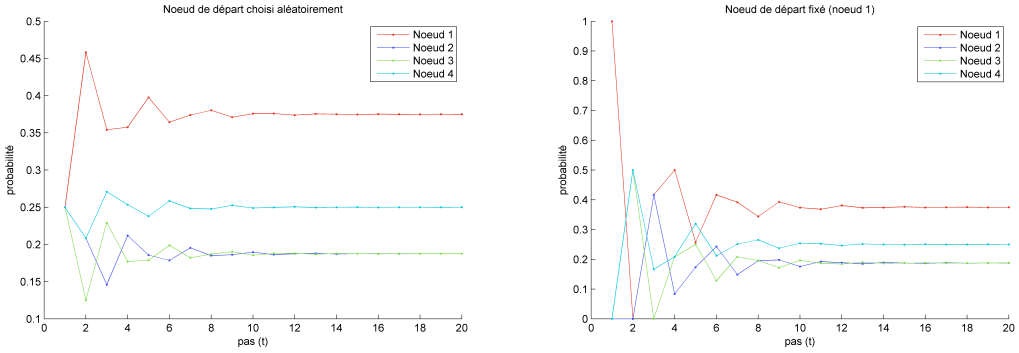


FIGURE 1.3 – Évolution de la distribution de probabilité

La matrice $Q^{(20)}$ obtenue est la suivante :

$$Q^{(20)} = \begin{pmatrix} 0.3751 & 0.1874 & 0.1875 & 0.2500 \\ 0.3751 & 0.1877 & 0.1874 & 0.2499 \\ 0.3749 & 0.1875 & 0.1875 & 0.2500 \\ 0.3749 & 0.1875 & 0.1875 & 0.2500 \end{pmatrix}$$

Si on observe les matrices $Q^{(k)}$ pour $k > 20$, on peut constater que les éléments se stabilisent et que les lignes s'égalisent.

4) La distribution stationnaire a été calculée par la méthode des puissances. Nous avons donc multiplié les distributions $\pi^{(k)}$ successives par Q jusqu'à ce que cette distribution se stabilise. Le critère de stabilisation choisi était le suivant :

$$\max\left(\left|\pi_j^{(k)} - \pi_j^{(k-1)}\pi\right|\right) < \varepsilon$$

où π_j est la $j^{\text{ième}}$ composante du vecteur π. La distribution stationnaire obtenue est donnée ci-dessous :

$$\pi_\infty = \begin{pmatrix} 0.3750 & 0.1875 & 0.1875 & 0.2500 \end{pmatrix}$$

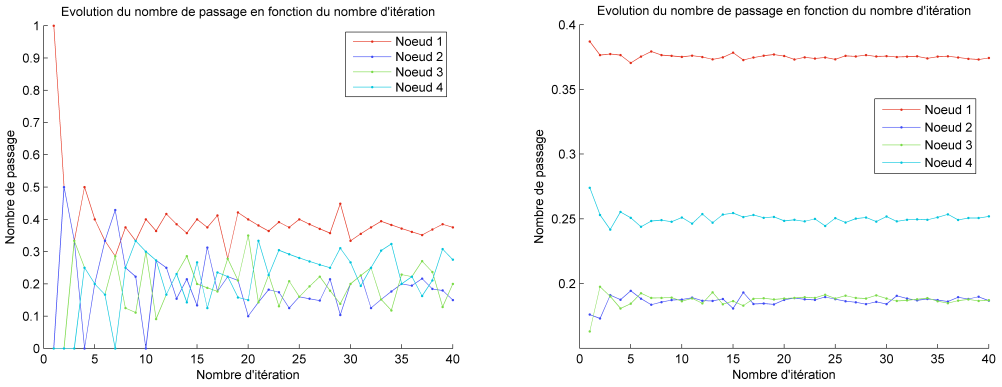
On constate que les lignes de la matrice sont très proches des valeurs observables sur les graphiques du point précédent.

5) On constate que le nœud 1 possède le meilleur PageRank, suivi des nœuds 2 et 3 à égalité et du nœud 4. On peut expliquer ce classement intuitivement :

- le **nœud 1** possède le plus d'arêtes entrantes donc ayant le plus de chance d'être visité
- le **nœud 3** possède le moins d'arêtes entrantes donc ayant le moins de chance d'être visité
- les **nœuds 2** et **4** possèdent le même nombre intermédiaire (par rapport aux deux autres) d'arêtes entrantes. Le PageRank du nœud 4 est néanmoins plus élevé que celui du nœud 2 puisque le nœud 4 possèdent une arête entrante venant du nœud 1 qui est le plus visité.
- malgré un nombre d'arêtes entrantes plus élevé que pour le nœud 3, le **nœud 2** possède une PageRank égal. Cela est dû au fait que, d'une part, le nœud 3 peut être visité depuis le nœud le plus visité (nœud 1) ce qui améliore son PageRank et, d'autre part, que le nœud 2 ne peut être accédé depuis des nœuds moins visités (nœud 3 et 4) ce qui abaisse son PageRank.

6) Dans un premier temps, nous avons généré une chaîne pour chaque longueur. Le résultat obtenu est donné sur la Figure 1.4(a). On peut déjà observer que les différentes courbes obtenues oscillent autour de leur probabilité stationnaire correspondante. Néanmoins, étant donné la présence d'oscillations, nous avons décidé de refaire l'expérience en générant cette fois-ci 1000 chaînes pour chaque longueur. Nous avons ensuite moyenné les différentes probabilités afin d'obtenir un résultat plus précis (voir Figure 1.4(b)). Les courbes obtenues nous permettent de confirmer les premières observations.

Remarquons aussi que, quelque soit la distribution de départ, la distribution converge vers la distribution stationnaire.



(a) 1 génération (b) 1000 générations

FIGURE 1.4 – Évolution du nombre de passage par un nœud

7) Nous pouvons relier ces résultats à la théorie car la chaîne est ***ergodique***. En effet :

- Premièrement, la chaîne est ***irréductible*** car il est possible de visiter tous les autres nœuds à partir de n'importe quel nœud donné.
- Deuxièmement, la chaîne est ***apériodique*** car le plus grand commun diviseur du nombre de pas nécessaires **N** pour aller d'un nœud vers lui même est inférieur à 2. Si on analyse le graphe, on observe que pour les nœuds 1, 3 et 4 le nombre de pas nécessaires vaut 2 alors que ce nombre vaut trois pour le nœud 2. **N** vaut donc 1.

Dans l'expérience du point 6, nous avons observé que la distribution de probabilité stationnaire correspond aux probabilités trouvées sur nos graphes lorsque le nombre d'itérations est assez élevé.

Analyse des graphes A_2 et A_3

1) Pour pouvoir calculer des éventuelles distributions stationnaires sur base des matrices A_2 et A_3, nous avons calculé les matrices de transition Q_2 et Q_3 :

$$Q_2 = \begin{pmatrix} 0 & 1.0000 & 0 & 0 \\ 0 & 0 & 0 & 1.0000 \\ 0.5000 & 0.5000 & 0 & 0 \\ 1.0000 & 0 & 0 & 0 \end{pmatrix} \quad Q_3 = \begin{pmatrix} 0 & 1.0000 & 0 & 0 & 0 \\ 1.0000 & 0 & 0 & 0 & 0 \\ 0 & 0.3333 & 0 & 0.3333 & 0.3333 \\ 0 & 0 & 0 & 0 & 1.0000 \\ 0 & 0 & 0 & 1.0000 & 0 \end{pmatrix}$$

Nous avons représenté les diagrammes d'états des deux chaînes de Markov sur la FIGURE 1.5. Nous avons ensuite appliqué la méthode des puissances pour obtenir des distributions $\pi^{(k)}$ successives. Le résultat est donné sur la FIGURE 1.6. On constate qu'aucune distribution stationnaire n'a pu être trouvée étant donné la présence d'**oscillations**.

On constate aussi que la **probabilité** d'atteindre une certain nœud **tombe à 0** ou est directement nulle dès la première itération dans les deux situations. Ce phénomène est dû à la présence de *dangling nodes* dans les deux graphes. On peut directement voir la présence de ce type de

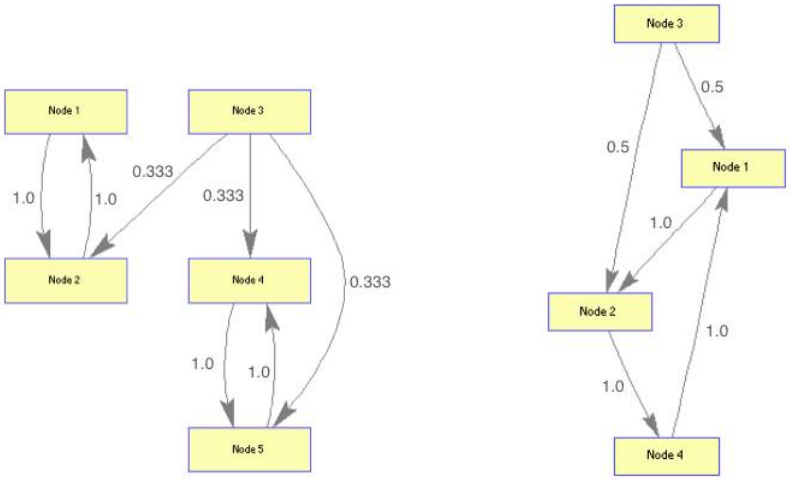


FIGURE 1.5 – Diagramme d'état de A_2 et A_3

nœud sur les matrices de transitions : elle se manifeste par une colonne composée uniquement de probabilités nulles et implique qu'il est impossible de passer d'un nœud quelconque au *dangling node*.

Ces deux phénomènes sont précisément des **limitations** du modèle du surfeur aléatoire simpliste présenté dans cette première partie. D'une part, les oscillations empêchent d'atteindre une distribution stationnaire. D'autre part, la probabilité nulle d'atteindre un nœud existant rend l'éventuelle distribution stationnaire (et donc le PageRank) peu représentative de l'organisation des nœuds puisqu'elle omet d'en prendre certains en compte.

Les **causes de ces phénomènes** sont d'une part la présence d'un cycle infini dans les graphes et la présence de *dangling nodes*. Ces problèmes vont être contournés en introduisant la téléportation dans la section suivante.

2) Tout d'abord, affichons les résultats obtenus à partir de la fonction `findStationnaryPi` (*cfr.* question 1.1.4 - méthode des puissances) appliquée aux différentes matrices de transition :

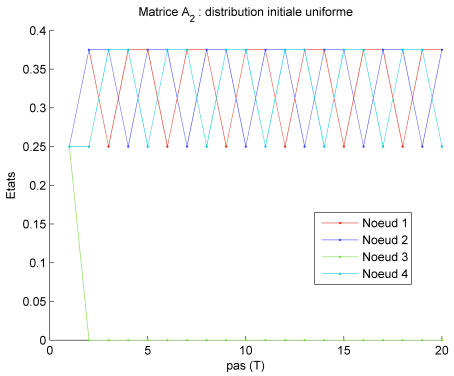
	Nœud 1	Nœud 2	Nœud 3	Nœud 4	Nœud 5
Q_1	0.3750	0.1875	0.1875	0.2500	—
Q_2	0.3750	0.3750	0	0.2500	—
Q_3	0.2667	0.2000	0	0.2667	0.2667

Notons que les distributions pour Q_2 et Q_3 ont été calculés à partir d'une distribution uniforme.

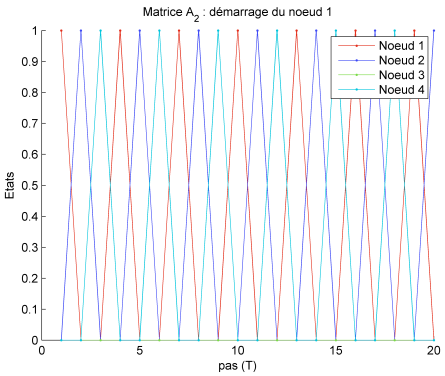
Dans cette section, deux méthodes de calcul ont été utilisées :

- **Résolution du système linéaire** : On a $\pi_\infty Q = \pi_\infty$ et $\sum_i \pi_{\infty,i} = 1$, que l'on peut transformer en un système :

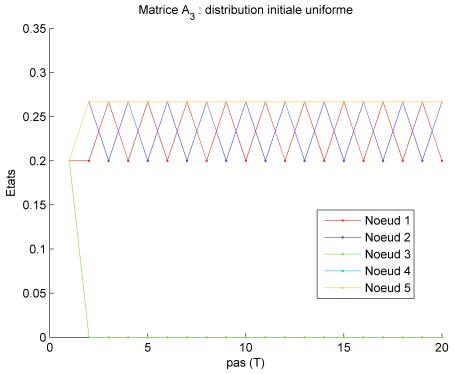
$$(Q' - \mathbb{1})\pi_\infty^T = b$$



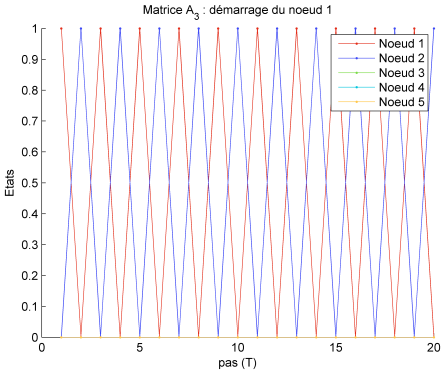
(a) Distribution initiale uniforme



(b) Depuis le noeud 1



(c) Distribution initiale uniforme



(d) Depuis le noeud 1

FIGURE 1.6 – Évolution des distributions de probabilités

où Q' est formé de la matrice Q^T à laquelle une ligne de '1' a été ajoutée (matrice de taille $(k+1) \times k$, k étant le nombre de nœuds) et b est un vecteur colonne (de taille $k+1$) composé de zéros sauf le dernier élément qui vaut '1'.

– **Résolution par vecteurs/valeurs propres** : On a le système suivant :

$$\pi_\infty Q = \pi_\infty$$
$$\Leftrightarrow Q^T \pi_\infty^T = \pi_\infty^T$$

Cette équation est une équation de type $Ax = \lambda x$, où A est de taille $n \times n$, x est $n \times 1$ et λ est un scalaire. Ce type d'équation est en fait un calcul de recherche de vecteurs et valeurs propres. Dans notre cas, la valeur propre λ vaut 1, il s'agit alors de calculer le vecteur propre correspondant et de le normaliser afin de respecter l'équation $\sum_i \pi_{\infty,i} = 1$, ce qui a été implémenté dans la fonction `getStationnaryPiBySystem`.

Ces deux méthodes donnent un résultat exactement identique pour chaque matrice de transition, à l'exception du troisième graphe pour lequel une distribution supplémentaire a été trouvée. En effet, la fonction `linsolve` émet un warning "*Rank deficient*" nous informant que le système est indéterminé, il n'y a pas suffisamment d'équations - rang de $Q' - \mathbb{1} <$ nombre d'inconnues,

mais ne renvoie qu'une seule solution. On peut en déduire qu'il existe une infinité de solutions.

On peut obtenir ce même résultat à partir des deux distributions obtenues par la méthode des vecteurs/valeurs propres : supposons les distributions stationnaires π_A et π_B, par définition :

$$\pi_A Q_3 = \pi_A$$

et

$$\pi_B Q_3 = \pi_B$$

On peut alors trouver une autre distribution (stationnaire) qui est combinaison des deux précédentes :

$$(\alpha\pi_A + \beta\pi_B)Q_3 = (\alpha\pi_A + \beta\pi_B)$$

On peut en déduire qu'il existe une infinité de distributions stationnaires pour la matrice Q_3.

Contrairement à la matrice A_1, les matrices A_2 et A_3 ne donnent pas le même résultats qu'avec la fonction `findStationnaryPi`. En effet, en exécutant celle-ci, le message "*Number of maximum iterations reached*" apparaît, signifiant que la sortie a une plus forte probabilité d'être un état non-stationnaire, c'est-à-dire que la fonction a été arrêtée avant d'atteindre un état stationnaire (s'il y en a un par la suite). Dans notre cas, en affichant les distributions, on peut voir que celles-ci sont périodiques, cela étant du aux cycles présents dans les graphes.

Concernant le deuxième graphe, le système d'équation est déterminé (rang de $Q' - \mathbb{1}$ = nombre d'inconnues), le vecteur que l'on obtient est bien unique. On peut observer que la probabilité de passer par le nœud 3 est nulle, nous sommes en effet en présence d'un graphe réductible. Au vu de la propriété des distributions stationnaires, nous continuerons dans l'état stationnaire si la distribution initiale est la distribution stationnaire. De même pour le troisième graphe, nous continuerons dans l'état stationnaire si la distribution initiale est une des distributions stationnaires.

1.2 Téléportation

1) La formule utilisée pour calculer la matrice de transition Q_t du modèle du surfeur avec téléportation est la suivante :

$$Q_t = (1 - \alpha)Q' + \alpha\tilde{Q}$$

où Q' et $\tilde{Q}$ sont des matrices de transition et α la probabilité de téléportation.

La première est la matrice de transition du graphe initial auquel on a rajouté des arêtes partant des *dangling nodes*. Elle a été calculée en remplaçant par $\frac{1}{n}$ tous les éléments de la matrice Q situés dans des lignes ne contenant que des 0. Cette valeur $\frac{1}{n}$ a été choisie en considérant une densité de probabilité uniforme entre les différentes arêtes partant des *dangling nodes*.

La seconde est la matrice de transition d'un graphe complet formé à partir des nœuds du graphe initial. Il s'agit, autrement dit, de la matrice de transition représentant la téléportation. Une combinaison linéaire de paramètre α est ensuite appliquée aux deux matrices pour trouver la matrice Q_t.

2) On sait qu'une distribution stationnaire π_s est unique si **la chaîne de Markov est irréductible**. Autrement dit, il faut que pour tout couple de nœuds (i_1, i_2), il existe une arête les reliant (un probabilité non-nulle de passer de i_1 à i_2). Cette propriété est vérifiée avec le modèle du surfeur modifié puisque la téléportation permet, depuis tout nœud, de se diriger vers un autre nœud tant que $\alpha > 0$.

A partir du moment où $\alpha = 0$, on n'est plus assuré que chaque paire de nœuds est reliée par une arête et donc que π_s est bien stationnaire.

3) Le classement des sites les plus visités, obtenus à l'aide de la distribution stationnaire, est donné dans la Table 1.1.

N°	PageRank	Page
1	0.0045	http ://purl.org/rss/1.0/modules/content
2	0.0027	http ://www.ulg.ac.be
3	0.0024	http ://ogp.me/ns♯
4	0.0024	http ://www.gre-liege.be
5	0.0023	http ://blog.intelliterwal.net
6	0.0023	http ://www.jalios.com
7	0.0022	http ://www.vmfnet.be
8	0.0022	http ://www.alinoa.be
9	0.0022	http ://www.ulb.ac.be
10	0.0022	http ://www.cedia.ulg.ac.be

TABLE 1.1 – Classement des sites ayant le meilleur PageRank ($\alpha = 0.15$)

4) Soit X_{t_j} la variable aléatoire qui prend la valeur p_i au temps t_j si le surfeur est sur la page i. On veut calculer :

$$P(X_{t_2} = p_2 | X_{t_3} = p_3, X_{t_1} = p_1)$$

En appliquant la formule de Bayes et en se rappelant de la propriété d'une chaîne de Markov, on obtient :

$$\frac{P(X_{t_2} = p_2 | X_{t_1} = p_1) \cdot P(X_{t_3} = p_3 | X_{t_2} = p_2,)}{P(X_{t_3} = p_3 | X_{t_1} = p_1)}$$

Lorsque l'on passe à la notation matricielle, on a :

$$\frac{Q^{t_3 - t_2}(p_3, p_2) \cdot Q^{t_2 - t_1}(p_2, p_1)}{Q^{t_3 - t_1}(p_3, p_1)}$$

Enfin en appliquant la formule pour le cas désigné dans l'énoncé, nous obtenons :

$$\frac{Q^{10}(p_3, p_2) \cdot Q^{9}(p_2, p_1)}{Q^{19}(p_3, p_1)} = 0.0308$$

1.3 Effet du α

1) Pour prouver que le score PageRank de toute page est au moins $\frac{\alpha}{n}$ (n est le nombre de pages), on peut développer une expression "*explicite*" des éléments de la matrice Q_t en utilisant

la formule donnée précédemment :

$$Q_t(i,j) = q_{ij}(1-\alpha) + \frac{1}{n}\alpha$$

où q_{ij} est un élément de la matrice Q'. Connaissant la relation qui lie $\pi^{(k)}$ et $\pi^{(k-1)}$, on a :

$$\begin{aligned}
\pi_j^{(k)} &= \sum_{i=1}^{n} Q_t(i,j)\pi_i^{(k-1)} \\
&= \sum_{i=1}^{n} \left(q_{ij}(1-\alpha) + \frac{\alpha}{n}\right)\pi_i^{(k-1)} \\
&= \sum_{i=1}^{n} q_{ij}(1-\alpha)\pi_i^{(k-1)} + \sum_{i=1}^{n} \frac{\alpha}{n}\pi_i^{(k-1)} \\
&= \frac{\alpha}{n} + \underbrace{(1-\alpha)\sum_{i=1}^{n} q_{ij}\pi_i^{(k-1)}}_{>0}
\end{aligned}$$

Le deuxième terme est inférieur à 1 (et même inférieur à $(1-\frac{\alpha}{n})$ afin de respecter le deuxième axiome de Kolmogorov) et surtout, positif. De ce fait, on peut affirmer que :

$$\pi_j^{(k)} \geq \frac{\alpha}{n}$$

On peut interpréter le cas où $\alpha = 0$ comme le cas où il n'y a pas de téléportation et le cas où $\alpha = 1$ comme le cas où il n'y a que téléportation (le surfeur n'utilise plus que les liens). Remarquons les valeurs prises par les distributions dans les deux cas :

$$\pi_{\alpha=0}^{(k)} = \pi^{(k-1)}Q'$$

$$\pi_{j,\alpha=1}^{(k)} = \tfrac{1}{n}, \forall j$$

Dans le deuxième cas, les PageRank de toutes les pages seront égaux et vaudront $\frac{1}{n}$. Afin de vérifier cette affirmation, nous avons calculé la distribution stationnaire pour $\alpha = 1$ et l'avons mise en parallèle avec la distribution stationnaire pour $\alpha = 0.15$. Le résultat est édifiant (voir Table 1.2), on constate en effet un PageRank uniforme dans le cas où $\alpha = 1$.

α	0.15	1
Moyenne	0.002	0.002
Ecart-type	1.3000e-04	**4.7753e-18**
Min.	0.0019	0.002
Max.	0.0045	0.002

TABLE 1.2 – Statistiques à propos des distributions stationnaires avec $\alpha = 0.15$ et $\alpha = 1$

2) Nous avons analysé l'évolution du PageRank de certaines pages lorsque alpha évolue. En regardant la Figure 1.7(a), on observe que plus alpha est grand, plus le PageRank des pages tend à s'uniformiser. En effet, un PageRank élevé au départ va diminuer avec l'augmentation de α, tandis qu'un PageRank assez bas va augmenter avec α vu que plus alpha est élevé, plus la probabilité d'arriver sur une telle page s'uniformise.

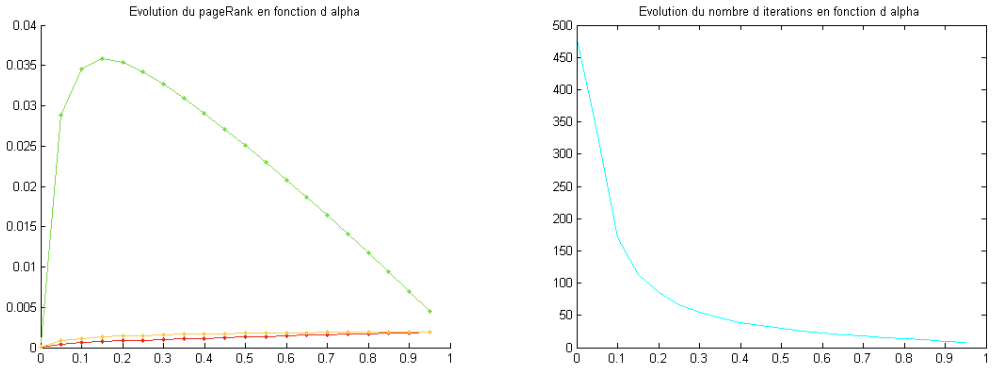


FIGURE 1.7 – Évolution du PageRank et du nombre d'itérations avec α

3) Nous avons déterminé l'évolution du nombre d'itérations nécessaires pour trouver une distribution stationnaire lorsque α varie. Les résultats sont présentés sur la Figure 1.7(b).

Les résultats observés sont attendus car il est logique de trouver une distribution stationnaire très rapidement lorsque α tend vers 1, étant donné que on ne tient compte que de la téléportation : la probabilité d'arriver sur une page ou une autre partant d'une page donnée est uniforme. Dans le cas contraire, lorsque alpha tend vers zéro, on ne repose plus sur le graphe de départ ce qui est plus compliqué à calculer qu'une répartition uniforme.

Chapitre 2

Question 2

2.1 Estimation d'une matrice de transition

2.1.1 Méthode d'estimation

Avant tout, voici les quelques notations que nous utiliserons :
- n, le nombre de noeuds dans le graphe
- X, la trace fournie (chaîne de Markov)
- Q_{est}, la matrice de transition recherchée

$$Q_{est} = \begin{pmatrix} \theta_{11} & \cdots & \theta_{1n} \\ \vdots & \ddots & \vdots \\ \theta_{n1} & \cdots & \theta_{nn} \end{pmatrix}$$

- N, la matrice dont l'élément β_{ij} est le nombre de transition de l'état i à j dans la trace X
- σ_i, la somme de la $i^{\text{ème}}$ ligne de la matrice N
- $\underline{\theta_i}$, le vecteur contenant les probabilités pour passer du nœud i à un autre nœud :

$$\underline{\theta_i} = [\theta_{i1} \cdots \theta_{in}]$$

- $P(X|\underline{\theta_i})$, la probabilité d'observer les départs du nœud i présents dans la trace connaissant $\underline{\theta_i}$:

$$P(X|\underline{\theta_i}) = \theta_{i1}^{\beta_{i1}} \times \cdots \times \theta_{i(n-1)}^{\beta_{i(n-1)}} \times \left(1 - \sum_{k=1}^{n-1} \theta_{ik}\right)^{\beta_{in}} = \left(1 - \sum_{k=1}^{n-1} \theta_{ik}\right)^{\beta_{in}} \times \prod_{k=1}^{n-1} \theta_{ik}^{\beta_{ik}}$$

- S_i, le système de $n-1$ équations à résoudre pour trouver la ligne i de la matrice de transition par la méthode du maximum de vraisemblance. On note S_{ij} la $j^{\text{ième}}$ équation de ce système :

$$S_i \equiv \begin{cases} \dfrac{\partial P(X|\underline{\theta_i})}{\partial \theta_{i1}} = 0 \\ \vdots \\ \dfrac{\partial P(X|\underline{\theta_i})}{\partial \theta_{i(n-1)}} = 0 \end{cases}$$

La résolution de ce système ne donne que les $n-1$ probabilités de la ligne i. Il suffit de les sommer pour obtenir la $n^{\text{ième}}$.

Développons l'équation trouvée ci-dessus (pour $j \in [2, n-2]$ mais que l'on peut facilement généraliser pour les cas où $j = 1$ ou $(n-1)$) :

$$\begin{aligned} S_{ij} &= \left[\beta_{ij} \theta_{ij}^{\beta_{ij}-1} \left(1 - \sum_{k=1}^{n-1} \theta_{ik}\right)^{\beta_{in}} - \beta_{in} \left(1 - \sum_{k=1}^{n-1} \theta_{ik}\right)^{\beta_{in}-1} \theta_{ij}^{\beta_{ij}} \right] \times \prod_{k=1, k\neq j}^{n-1} \theta_{ik}^{\beta_{ik}} \\ &= \left[\beta_{ij} \left(1 - \sum_{k=1}^{n-1} \theta_{ik}\right) - \beta_{in}\theta_{ij} \right] \times \left(1 - \sum_{k=1}^{n-1} \theta_{ik}\right)^{\beta_{in}-1} \times \theta_{ij}^{\beta_{ij}-1} \times \prod_{k=1, k\neq j}^{n-1} \theta_{ik}^{\beta_{ik}} = 0 \end{aligned}$$

Il nous reste à résoudre l'équation suivante :

$$\begin{aligned} & \beta_{ij} \left(1 - \sum_{k=1}^{n-1} \theta_{ik}\right) - \beta_{in}\theta_{ij} = 0 \\ \Leftrightarrow\ & \beta_{ij} \left(1 - \sum_{k=1}^{n-1} \theta_{ik}\right) = \beta_{in}\theta_{ij} \\ \Leftrightarrow\ & \frac{\beta_{in}}{\beta_{ij}}\theta_{ij} + \sum_{k=1}^{n-1} \theta_{ik} = 1 \\ \Leftrightarrow\ & \left(\frac{\beta_{in}}{\beta_{ij}} + 1\right)\theta_{ij} + \sum_{k=1, k\neq j}^{n-1} \theta_{ik} = 1 \quad (1) \end{aligned}$$

Le système S_i peut se réécrire sous forme matricielle de la manière suivante :

$$S_i \equiv \begin{pmatrix} \left(\frac{\beta_{in} + \beta_{i1}}{\beta_{i1}}\right) & 1 & \cdots & 1 \\ 1 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & 1 \\ 1 & \cdots & 1 & \left(\frac{\beta_{in} + \beta_{i(n-1)}}{\beta_{i(n-1)}}\right) \end{pmatrix} \begin{pmatrix} \theta_{i1} \\ \vdots \\ \theta_{i(n-1)} \end{pmatrix} = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$$

La résolution de ce système pour chaque ligne permet d'obtenir une estimation de la matrice de transition. Néanmoins, cette méthode est extrêmement inefficace puisqu'elle nécessite n résolutions du système. Bien qu'à notre échelle ($n = 50$), cette complexité élevée ne soit pas gênante, une simplification de la méthode d'estimation serait tout de même la bienvenue.

Repartons de l'équation (1) :

$$\begin{aligned}
&\left(\frac{\beta_{in}}{\beta_{ij}} + 1\right)\theta_{ij} + \sum_{k=1,k\neq j}^{n-1} \theta_{ik} = 1 \\
&\Leftrightarrow \frac{\beta_{in}}{\beta_{ij}}\theta_{ij} + \theta_{ij} = 1 - \sum_{k=1,k\neq j}^{n-1} \theta_{ik} \\
&\Leftrightarrow \frac{\beta_{in}}{\beta_{ij}}\theta_{ij} + \theta_{ij} = \theta_{ij} + \theta_{in} \\
&\Leftrightarrow \theta_{ij} = \frac{\theta_{in}}{\beta_{in}}\beta_{ij} \quad (2) \\
&\Leftrightarrow \sum_{i=0}^{n-1} \theta_{ij} = \frac{\theta_{in}}{\beta_{in}} \sum_{i=0}^{n-1} \beta_{ij} \\
&\Leftrightarrow 1 - \theta_{in} = \frac{\theta_{in}}{\beta_{in}} (\sigma_i - \beta_{in}) \\
&\Leftrightarrow \theta_{in} = \frac{\beta_{in}}{\sigma_i} \quad (3)
\end{aligned}$$

En injectant l'équation (3) dans l'équation (2), on obtient l'équation :

$$\theta_{ij} = \frac{\beta_{ij}}{\sigma_i}$$

La formule ci-dessus nous permet de simplifier l'estimation puisque qu'il suffit maintenant de **diviser chaque élément β_{ij} de la matrice N par la somme de la ligne dans laquelle il se trouve** pour trouver la matrice de transition. Il ne reste plus qu'à régler le problème où

$$\beta_{ij} = 0, \forall j \in [1, n]$$

En effet, cette situation mènerait à une division par zéro. Il faut donc appliquer un traitement à la matrice N afin de supprimer les valeurs nulles.

Smoothing

Un des problèmes liés à l'estimation par le maximum de vraisemblance se pose lorsque le set de données permettant d'estimer Q ne contient pas d'information sur certains nœuds, *i.e.* $\beta_{ij} = 0$. En effet, lorsque par la suite, nous souhaiterions estimer la probabilité qu'une certaine séquence, contenant un nœud qui n'a pas été compté pour estimer Q, corresponde à ce modèle, nous obtiendrons une probabilité nulle.

Plusieurs techniques de lissage existent : soient la transition du nœud ω_{i-1} vers le nœud ω_i observée, $c(x)$ le nombre de fois où le nœud x a été compté, $c(x, y)$ le nombre de fois la transition du nœud x vers le nœud y a été compté et n le nombre de nœuds :

- **Méthode de Laplace (*Laplace smoothing*) :** Cette méthode consiste à ajouter 1 à tous les éléments comptés. On obtient alors une probabilité de la forme :

$$P_{Laplace}(\omega_i|\omega_{i-1}) = \frac{c(\omega_{i-1}, \omega_i) + 1}{c(\omega_{i-1}) + n}$$

– ***Add-k smoothing*** **:** Cette méthode généralise la méthode de Laplace où, cette fois, k est ajouté au numérateur et $k.n$ est ajouté au dénominateur :

$$P_{Add-k}(\omega_i|\omega_{i-1}) = \frac{c(\omega_{i-1},\omega_i)+k}{c(\omega_{i-1})+k.n}$$

– ***Unigram prior smoothing*** **:** Il s'agit d'une variante de l'*add-k smoothing* où la probabilité d'obtenir le noeud (unigramme) ω_i intervient. Par rapport à la formule de l'*add-k smoothing*, on prend $k = m/n$ où m est un paramètre à déterminer. On obtient alors :

$$P_{Add-k-var}(\omega_i|\omega_{i-1}) = \frac{c(\omega_{i-1},\omega_i)+m.\left(\frac{1}{n}\right)}{c(\omega_{i-1})+m}$$

La "subtilité" de l'*unigram prior smoothing* consiste à remplacer la constante $\frac{1}{n}$ au numérateur par $P(\omega_i)$. On a au final :

$$\begin{aligned} P_{UnigramPrior}(\omega_i|\omega_{i-1}) &= \frac{c(\omega_{i-1},\omega_i)+m.P_{ML}(\omega_i)}{c(\omega_{i-1})+m} \\ &= \frac{c(\omega_{i-1},\omega_i)+m\frac{c(\omega_i)}{n}}{c(\omega_{i-1})+m} \end{aligned}$$

– ***Jelinek-Mercer smoothing*** **(interpolation) :** Il s'agit d'une interpolation linéaire entre le modèle de l'unigramme (noeud) et du bigramme (transitions) afin de pouvoir prendre en compte les probabilités de ces deux modèles :

$$\begin{aligned} P_{Jelinek-Mercer}(\omega_i|\omega_{i-1}) &= \lambda.P_{ML}(\omega_i|\omega_{i-1}) + (1-\lambda).P_{ML}(\omega_i) \\ &= \lambda\frac{c(\omega_i,\omega_{i-1})}{c(\omega_i)} + (1-\lambda)\frac{c(\omega_i)}{n} \end{aligned}$$

Il existe encore d'autres méthodes que nous n'avons pas implémentées :

– ***Good-Turing*** **:** Cette méthode agit sur les éléments non vus (transitions ne figurant pas dans la trace) et rectifie ce compte en fonction des éléments comptés 1 fois, 2 fois,... Les transitions effectuées r fois seront modifiés en r^* tel que $r^* = (r+1)\frac{n_{r+1}}{n_r}$, où n_r est le nombre de transitions comptés exactement r fois.

2.1.2 Utilisation des modèles estimés

Une méthode pour déterminer l'origine des traces d'un ensemble de traces consiste à calculer la probabilité $P(Q_i|tr_k)$ que tr_k provienne d'une certaine matrice de transition Q_i (la matrice de transition définissant le comportement d'un surfeur). Il suffira ensuite de prendre le surfeur dont la matrice de transition mène à la plus grande probabilité. Autrement dit :

$$arg\max_i P(Q_i|tr_k)$$

À l'aide de la formule de Bayes, on peut développer :

$$P(Q_i|tr_k) = \frac{P(tr_k|Q_i)P(Q_i)}{\sum_{j=1}^{n} P(tr_k|Q_j)P(Q_j)}$$

où le terme $P(tr_k|Q_i)$ est la vraisemblance de la trace connaissant la matrice Q_i et $P(Q_i)$ est la probabilité qu'une trace provienne du surfeur i.

Dans notre cas, la vraisemblance est un nombre tellement petit qu'il n'est pas représentable sur un ordinateur (à moins d'utiliser des bibliothèques spéciales). Nous avons donc décidé d'utiliser la **log-vraisemblance** de sorte que les nombres soient manipulables.

Étant donné que les traces tr_k ne peuvent appartenir qu'à deux surfeurs, nous pouvons aussi modifier notre règle de décision. Pour ce faire, nous définissons le *log-ratio* :

$$\begin{aligned} r &= \log \frac{P(Q_1|tr_k)}{P(Q_2|tr_k)} \\ &= \log \frac{P(tr_k|Q_1)P(Q_1)}{P(tr_k|Q_2)P(Q_2)} \\ &= \log P(tr_k|Q_1) - \log P(tr_k|Q_2) + \log \frac{P(Q_1)}{P(Q_2)} \end{aligned}$$

Si $r > 0$ alors la trace d'origine provient du **surfer 1** sinon elle provient du **surfer 2**. Notons que dans notre cas :

$$P(Q_i) = \frac{1}{2}, \forall i \in [1, 2]$$

En effet, nous savons que parmi les 20 traces à évaluer, 10 appartiennent au premier (soit la moitié) et 10 au deuxième (soit l'autre moitié).

Calcul de la vraisemblance

Pour une trace X, une matrice N contenant les éléments β_{ij} (nombre de transitions de l'état i à j dans X) et une matrice de transition Q, la vraisemblance est donnée par :

$$P(X|Q) = \prod_{i=1}^{n} \prod_{j=1}^{n} \theta_{ij}^{\beta_{ij}}$$

Comme expliqué plus haut, nous utiliserons la log-vraisemblance qui se calcule comme suit :

$$logP(X|Q) = \sum_{i=1}^{n} \sum_{j=1}^{n} \beta_{ij} \log \theta_{ij}$$

2.1.3 Analyse des 20 traces

En appliquant la méthode expliquée dans la Section 2.1.2, nous avons obtenu les résultats donnés dans la Table 2.1. Pour la raison expliquée précédemment, nous ne savons pas calculer explicitement la probabilité $P(Q_i|tr_k)$. Nous avons néanmoins donné le *log-ratio* r qui est, dans ce cas ci, la différence entre les log-vraisemblances. Nous observons bien 10 traces provenant du premier surfeur et 10 traces provenant du second.

2.1.4 Hypothèses, fiabilité et limites

Pour appliquer la méthode du maximum de vraisemblance, on doit uniquement connaître les probabilités qui nous sont données par le graphe. Nous avons testé les résultats de notre méthode.

Trace	r	Provient de
1	-0.4361	**2**
2	57.737	**1**
3	70.562	**1**
4	68.022	**1**
5	-153.35	**2**
6	127.45	**1**
7	-47.847	**2**
8	47.912	**1**
9	-13.472	**2**
10	-10.209	**2**
11	-183.48	**2**
12	-155.67	**2**
13	52.513	**1**
14	10.682	**1**
15	19.723	**1**
16	-104.7	**2**
17	-34.937	**2**
18	12.536	**1**
19	-53.645	**2**
20	126.89	**1**

TABLE 2.1 – Recherche des surfeur ayant engendré les traces

Test de la méthode

Pour tester cette méthode, nous avons procédé en plusieurs étapes :

1. Définition d'une matrice d'adjacence A de taille $k \times k$ (ici, $k = 5$)
2. Calcul de la matrice de transition Q et génération de n (ici, $n = 50$) chaînes de Markov X_i de taille m (ici, $m = 1000$)
3. Génération de n estimations $Q_{est,i}$ de la matrice de transition sur base des chaînes générées (méthode de lissage : *add-k* avec $k = 0.06$)
4. Calcul du module du biais E pour chaque θ_{ij}

En utilisant cette méthode nous avons obtenu des résultats intéressants. Ci-dessous sont donnés la matrice Q exacte, le biais moyen sur les 50 générations et Q_{est}, une des matrices estimées :

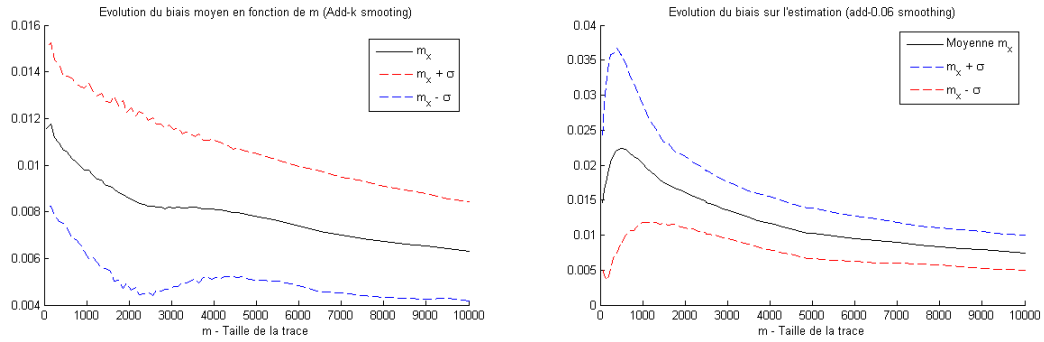
$$Q = \begin{pmatrix} 0.1 & 0.1 & 0.1 & 0.1 & 0.6 \\ 0.1 & 0.1 & 0.1 & 0.35 & 0.35 \\ 0.1 & 0.35 & 0.1 & 0.35 & 0.1 \\ 0.6 & 0.1 & 0.1 & 0.1 & 0.1 \\ 0.35 & 0.1 & 0.35 & 0.1 & 0.1 \end{pmatrix}$$

$$E = \begin{pmatrix} 0.0011 & 0.0006 & 0.0006 & 0.0005 & 0.0044 \\ 0.0021 & 0.0006 & 0.0009 & 0.0039 & 0.0020 \\ 0.0022 & 0.0035 & 0.0006 & 0.0027 & 0.0020 \\ 0.0067 & 0.0007 & 0.0007 & 0.0007 & 0.0017 \\ 0.0015 & 0.0006 & 0.0017 & 0.0005 & 0.0009 \end{pmatrix}$$

$$Q_{est} = \begin{pmatrix} 0.1309 & 0.1078 & 0.1106 & 0.1112 & 0.5396 \\ 0.1413 & 0.1072 & 0.1191 & 0.3047 & 0.3276 \\ 0.1431 & 0.2986 & 0.1140 & 0.3043 & 0.1400 \\ 0.5221 & 0.1076 & 0.1174 & 0.1176 & 0.1353 \\ 0.3301 & 0.1110 & 0.3157 & 0.1167 & 0.1265 \end{pmatrix}$$

Cette configuration nous a donné un biais moyen de 0.0318.

Étudions maintenant l'évolution du biais en fonction de m, la taille des traces. Pour ce faire, nous fixons k et n à 50. Ensuite, pour chaque valeur de m, nous appliquons la méthode de test expliquée ci-dessus et récupérons la moyenne et l'écart type du biais que nous portons sur le graphe. La FIGURE 2.1 montre les courbes obtenues. Nous constatons (et c'était attendu) que le biais induit par l'estimation diminue lorsque la taille de la trace augmente.



(a) *add-λ smoothing* : $\lambda = 1$ (b) *add-λ smoothing* : $\lambda = 0.06$

FIGURE 2.1 – Évolution du biais moyen en fonction de la taille des chaînes de Markov

Notons que la méthode de *smoothing* influe grandement sur cette courbe. En effet, on peut constater la présence d'un maximum dans l'intervalle $m \in [0, 1000]$ dans le cas du *add*-0.06 *smoothing*. La zone où se trouve ce maximum est particulière puisque elle correspond à une chaîne relativement courte comparée au nombre de transitions possibles (ici, 2500) sur le graphe. De ce fait, elle n'est pas assez représentative du surfeur.

2.1.5 Applications possibles de la méthode

Les chaînes de Markov ont de nombreuses applications possibles dans des domaines très variés tels que les sciences sociales, la biologie, les sciences économiques et pleins d'autres.

Pour citer quelques exemples, on utilise les modèles de Markov pour identifier les régions de l'ADN génomique qui codent les gènes, pour la reconnaissance et la synthèse vocale, pour prédire les éruptions d'un geyser et bien d'autres encore.

Dans notre parcours universitaire, nous avons déjà utilisé un modèle de Markov caché pour définir une caractéristique d'un texte (langue, auteur). L'algorithme se basait sur plusieurs échantillons de textes ayant des caractéristiques différentes. Nous établissions des modèles sur base de

ces échantillons et testions ensuite l'appartenance d'un texte à l'un des modèles. Grâce à cet algorithme, nous avons pu discerner la langue d'un texte, le langage d'un code, l'auteur d'un livre à partir d'échantillons pour chaque caractéristique. Malgré la simplicité de ce programme, nous pouvions déjà faire un certain nombre de tests différents.

2.2 Estimation de α

2.2.1 Méthodes d'estimation

Méthode algébrique

1) Le paramètre α est un paramètre de la matrice de transition représentant un surfeur aléatoire. Nous avons défini dans la Section 1.3, la formule définissant chaque élément de cette matrice de transition en fonction de α pour le modèle du surfeur aléatoire :

$$Q_t(i,j) = q_{ij}(1-\alpha) + \frac{\alpha}{n}$$

Un approche algébrique consiste à utiliser cette formule afin de définir α. En effet, en isolant le paramètre, on obtient :

$$\alpha = \frac{Q_t(i,j) - q_{ij}}{\frac{1}{n} - q_{ij}}$$

Regardons de plus près cette formule :

- q_{ij} : puisque nous connaissons le graphe parcouru par le surfeur, nous connaissons ce terme de manière exacte (voir Section 1.2)
- $Q_t(i,j)$: ce terme provient de la matrice de transition du surfeur. Celle-ci est inconnue puisque nous recherchons son paramètre α. Nous pouvons néanmoins l'estimer en utilisant la méthode du maximum de vraisemblance (voir Section 2.1). L'erreur d'estimation induite par cette estimation se propagera dans la valeur calculée de α.
- Si $q_{ij} = \frac{1}{n}$: notre formule nous donne alors un α infini. Ce cas correspond à la situation où le noeud i est un *dangling node*. Si ce cas se présente, il faut donc éliminer les résultats correspondants.

Finalement, nous disposons de n^2 équations qui nous fournissent un échantillon d'au plus n^2 estimations. L'estimateur de α peut être trouvé en prenant la moyenne de cet échantillon.

Nous avons implémenté cette méthode et l'avons testée d'abord sur notre propre jeu de données. En l'occurrence, nous avons utilisé une matrice d'adjacence d'ordre 7. Nous l'avons ensuite testée (en appliquant la méthode expliquée dans la Section **??**) sur les 22 traces fournies et avons trouvé des résultats cohérents.

Nous avons enfin voulu appliquer un test final qui consistait à évaluer la qualité des estimations pour des variations de paramètres et nous avons commencé par une variation du paramètre k, l'ordre du graphe. Nous avons alors remarqué un gros défaut de notre méthode qui est mis en évidence sur la Figure 2.2. L'erreur commise s'accroît très rapidement quand le paramètre k augmente ce qui rend inutilisable notre méthode pour des grands graphes. Après avoir effectué ces observations, nous avons cherché une autre méthode d'estimation.

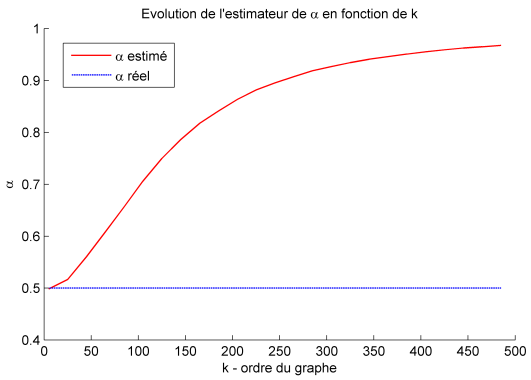


FIGURE 2.2 – Évolution de α en fonction de k - Méthode algébrique

Par le maximum de vraisemblance

Cette méthode consiste à générer la valeur de α qui maximise la vraisemblance d'une trace. Dans le cas où nous nous intéressons à n traces, nous procédons de la manière suivante :

- Nous générons un vecteur v contenant des valeurs de α couvrant l'intervalle $[0, 1]$ (de la manière suivante : `0.01:step:1`)
- Pour chaque α_i de v, nous générons la matrice de transition Q_i correspondante (prenant en compte la téléportation)
- Nous calculons la log-vraisemblance de chaque trace pour chaque matrice et nous sélectionnons le α_i qui a mené à la plus grande log-vraisemblance pour chaque trace.

Cette méthode, bien que satisfaisante du point de vue des résultats qu'elle fournit, est loin d'être efficace. En effet, nous devons générer une grand nombre de matrices de transition (ce nombre étant fonction du pas `step` appliqué) puis calculer la log-vraisemblance de toutes les traces pour toutes ces matrices. L'algorithme pourrait être amélioré en utilisant des méthodes d'optimisation numérique qui permettraient de ne pas devoir calculer toutes les matrices de transitions mais uniquement certaines d'entre elles. Nous pourrions aussi passer par développement analytique de la formule du maximum de vraisemblance. Ce développement pourrait mener, comme ce fût le cas dans la Section 2.1 pour l'estimation de Q, à une autre méthode plus efficace et plus simple.

2.3 Identification de l'origine d'une trace

Pour définir l'appartenance d'une trace à l'un des deux modèles, nous estimons la valeur des α_i des traces Tr_i des surfeurs grâce à la méthode décrite au premier point et nous considérons que une trace ayant un α estimé appartient au surfeur ayant un α_i le plus proche.

2.4 Application aux 20 traces

Nous avons calculé les valeurs de α pour chaque surfeur.

S_1	0.4010
S_2	0.6030

Trace	α	Méthode α
1	0.4120	1
2	0.4060	1
3	0.6320	2
4	0.3790	1
5	0.5870	2
6	0.3430	1
7	0.6350	2
8	0.4030	1
9	0.5610	2
10	0.5880	2
11	0.6450	2
12	0.6140	2
13	0.4250	1
14	0.3880	1
15	0.4120	1
16	0.6370	2
17	0.3860	1
18	0.5400	2
19	0.6070	2
20	0.4070	1

TABLE 2.2 – Identification de l'origine des traces

Si on compare avec le tableau 2.1, on remarque que 4 estimations diffèrent : celles pour les traces 1, 3, 17 et 18. Au vu des résultats des tests de nos méthodes, nous aurions plus tendance à choisir l'estimation à partir de α car l'erreur est plus faible.

2.5

Dans la section 2.1, nous utilisons uniquement la trace et l'ordre du graphe pour générer la matrice de transition Q. Une idée est donc de réutiliser cette méthode, mais seulement en remplaçant l'ordre du graphe par le nombre d'états distincts dans la trace. Nous créons donc un modèle de Markov caché car certains états peuvent être inconnus. Cette approximation devrait être correcte lorsque la taille de la trace est suffisamment grande de sorte que la plupart des états soient représentés dans celle-ci. Par contre si la trace est petite ou ne montre pas un nombre suffisant d'état, les résultats seraient très mauvais.

Annexe A

Comparaison de différentes méthodes de lissage

A.1 Estimation de Q

A.1.1 Évolution de l'erreur en fonction de la taille de la trace

On compare ici les précisions atteignables par les différentes méthodes, en comparant les erreurs quadratiques réalisées par rapport à Q lors de la modifications de paramètres relatifs à ces méthodes. Le graphe utilisé est le graphe donné dans l'énoncé.

La Table A.1 reprend les erreurs à paramètres fixés pour différentes tailles m_t de la trace. On peut remarquer que les méthodes "*add-k*" et "*unigram prior*" donnent des résultats fortement semblables, leurs précisions augmentent de la même façon. En comparaison avec l'interpolation, celle-ci possède une décroissance nettement plus faible, malgré le fait qu'elle possède une erreur quadratique plus grande pour des plus petites traces.

m_t	Add-k ($k = 0.06$)	Unigram prior ($m = 0.06 * 50$)	Interpolation ($\lambda = 0.3$)
100	1.3755×10^{-3}	1.5444×10^{-3}	4.2131×10^{-4}
300	1.4585×10^{-3}	1.4937×10^{-3}	2.8596×10^{-4}
500	1.1939×10^{-3}	1.2146×10^{-3}	2.5819×10^{-4}
800	9.0119×10^{-4}	9.1241×10^{-4}	2.4232×10^{-4}
1000	7.6995×10^{-4}	7.7718×10^{-4}	2.2381×10^{-4}
2000	4.3842×10^{-4}	4.3775×10^{-4}	2.2826×10^{-4}
5000	1.8841×10^{-4}	1.8972×10^{-4}	2.2237×10^{-4}
10000	9.6356×10^{-5}	9.6745×10^{-5}	2.2023×10^{-4}

Table A.1 – Précision de l'estimation de Q

A.1.2 Détermination des paramètres minimisant l'erreur

La Table A.2 reprend les différentes précisions obtenues en modifiant le paramètre k de la méthode "*add-k*", avec une taille de trace fixée à 5000. On remarque ici que la valeur minimale se situe en $\lambda = 1$.

λ	"*Add-k*"
0.02	$2.055\,09 \times 10^{-4}$
0.04	$2.060\,59 \times 10^{-4}$
0.06	$1.993\,54 \times 10^{-4}$
0.08	$1.929\,63 \times 10^{-4}$
0.1	$1.947\,27 \times 10^{-4}$
0.3	$1.584\,02 \times 10^{-4}$
0.5	$1.404\,48 \times 10^{-4}$
0.7	$1.283\,12 \times 10^{-4}$
0.9	1.1945×10^{-4}
1	$1.178\,69 \times 10^{-4}$

TABLE A.2 – Estimation du paramètre k pour la méthode *add-k*

La TABLE A.3 reprend les différentes précisions obtenues en modifiant le paramètre m de la méthode "*uniform prior*", avec une taille de trace fixée à 5000. On remarque ici que la valeur minimale se situe en $m = 100$.

k	$m = k.50$	Unigram prior
0.01	0.5	1.9824×10^{-4}
0.1	5	1.8813×10^{-4}
0.3	15	1.5445×10^{-4}
0.5	25	1.3759×10^{-4}
0.7	35	1.2522×10^{-4}
0.9	45	1.1752×10^{-4}
1	50	1.1499×10^{-4}
2	100	1.0787×10^{-4}
3	150	1.149×10^{-4}
5	250	$1.332\,35 \times 10^{-4}$

TABLE A.3 – Estimation du paramètre m pour la méthode "*uniform prior*"

La TABLE A.4 reprend les différentes précisions obtenues en modifiant le paramètre λ de la méthode par interpolation, avec une taille de trace fixée à 5000.

λ	Interpolation
0.1	2.2511×10^{-4}
0.2	2.2374×10^{-4}
0.3	2.2222×10^{-4}
0.4	2.2023×10^{-4}
0.5	2.19×10^{-4}
0.6	2.1351×10^{-4}
0.7	2.0713×10^{-4}
0.8	2.0226×10^{-4}
0.9	1.6776×10^{-4}
1	1.9933×10^{-4}

TABLE A.4 – Estimation du paramètre λ pour la méthode par interpolation

Bibliographie

[1] Bill MacCartney, *NLP Lunch Tutorial : Smoothing*. `http://nlp.stanford.edu/~wcmac/`, 21 Avril 2005.

[2] D. De Cao, R. Basili, *Probability Estimation*. 23 Avril 2009.

[3] David Smith, *Estimation - Maximum likelihood and smoothing* (Introduction to Natural Language Processing course). University of Massachusetts Amherst, Fall 2009.

[4] Dr. Mark Craven, *Markov Chain Models*

[5] `https://www.coursera.org/course/nlp`

[6] http ://stats.stackexchange.com/.../calculating-log-likelihood-for-given-mle-markov-chains