Langages de programmation de haut niveau
Travaux Pratiques

Année 2020/2021

- R—Mots suivants

Pour aider le rédacteur d'un texte ou d'un programme, il est possible de lui proposer pour chaque mot qu'il écrit un candidat pour le mot suivant. Avec cet objectif, une étude préalable d'un corpus de textes adapté permet de déterminer les mots les plus susceptibles de suivre un mot, pour chaque mot envisageable.

Pour simplifier, dans ce travail, un « mot » désigne une succession contiguë de caractères alphabétiques — il est clair que cette définition est insuffisante dans le cas général, par exemple en français *aujourd'hui* et *tire-laine* devraient être comptés pour un seul mot, mais *jusqu'alors* et *était-on* pour deux.

1 - Apprentissage

La lecture d'un fichier texte (ne contenant pas une série de données :-) peut se faire en R par la fonction **`readLines(…)`** (tableau de lignes) ou par la fonction **`scan(file=…, what="character",skip=…,n=…)`** qui donne un tableau de token. Plus d'informations sur ces fonctions sont disponibles en particulier dans le livre d'E.Paradis (https://cran.r-project.org/doc/contrib/Paradis-rdebuts_fr.pdf). Changer au préalable de répertoire peut se faire par la commande `setwd()`.

Le corpus d'étude pour ce travail sera constitué d'un unique fichier (par exemple le code d'un de vos programme, une page web assez longue, ou le fichier `cyrano.txt` fourni).

a) Écrire le code permettant d'obtenir la liste des mots d'un fichier texte, dans leur ordre d'apparition.

Une fois obtenue la liste des mots du texte, il est facile en R d'obtenir certains indicateurs statistiques. La fonction **`table()`** par exemple détermine les fréquences des valeurs dans une liste. Quelques fonctions sur les tables pourront peut-être vous aider : `order, rev, which` (cherchez leurs usages).

b) Écrire le code permettant d'obtenir les 10 mots les plus fréquents du texte.

c) Tracer un (ou plusieurs) graphique montrant que peu de mots accaparent l'essentiel du texte.

De façon similaire peut être déterminé pour chaque couple de mots `(motavant,motaprès)` le nombre de fois où `motaprès` suit `motavant` dans le texte. Ces résultats forment donc une matrice carrée `FREQ`.

c) Écrire le code permettant d'obtenir la matrice `FREQ`

d) Puis écrire une fonction qui, à partir de `FREQ`*, donne une fonction fournissant le mot le plus vraisemblable après un mot donné (ou une chaîne vide si aucune proposition n'est possible).*

2 - Étude

Manipuler la matrice complète des fréquences peut être pénalisant. Peut-être la portion de la matrice correspondant aux mots les plus fréquents serait suffisante.

e) Effectuer des mesures de temps d'exécution de la fonction écrite en d) en tirant au hasard des mots.

f) Définir une nouvelle matrice `MINIFREQ` *en ne prenant dans* `FREQ` *que les lignes et les colonnes correspondant aux mots les plus fréquents (un paramètre est à définir ici :-).*

g) Comparer les temps d'exécutions pour une fonction (comme en d) utilisant `FREQ` *et une fonction similaire utilisant* `MINIFREQ`*. Effectuer les mesures sur les mêmes tirages aléatoires bien sûr, et tracer les graphiques correspondants.*

La matrice `FREQ` peut se lire aussi comme une matrice de probabilité : en divisant chaque terme (i,j) par la somme de la colonne j qui le contient, on obtient pour j fixé la probabilité que chaque mot d'indice i suive le mot d'indice j.

En divisant chaque terme par la somme de la ligne, on obtient à rebours pour le mot i la probabilité que chaque mot j l'ai précédé.

e) Comment obtenir la probabilité qu'a chaque mot de suivre deux mots donnés (qui se suivent) ?