Interroger Lexique avec R

Christophe Pallier

Contents

Introduction

Ce document montre comment interroger et manipuler la base Lexique avec le logiciel R

Il fait parti du cours Programmation pour les Sciences Cognitives. Merci de signaler d'éventuelles erreurs à christophe@pallier.org

Préparation

1. S'ils ne sont pas déjà installés sur votre ordinateur, vous devrez installer les logiciels R et R-Studio Desktop.
2. Télécharger http://www.lexique.org/databases/Lexique382/Lexique382.zip
3. Créer un répertoire `Lexique` puis dézipper dans ce répertoire le fichier `Lexique382.zip` précédemment téléchargé.
4. Démarrer le programme **Rstudio**
   - Sélectionner le menu `File/New Project`;
   - Choisir “Existing directory” et naviguer jusqu'au répertoire `Lexique` que vous venez de créer; cliquer sur le bouton `create project`.
   - Dans la fenêtre en bas à droite, parmi la liste des fichiers listés sous l'onglet *Files*, il doit apparaitre `Lexique382.tsv`. Si tel n'est pas le cas, vérifiez que vous avez bien suivi les instructions.
5. Dans la fenêtre en bas à gauche de Rstudio, dans l'onglet *Console*, copier la ligne suivante puis appuyez sur ‘Entrée’

```
install.packages('tidyverse')
```

Laissez rstudio se débrouiller (mais vous devrez peut-être sélectionner un serveur). Quand il a fini, vous êtes prêt ! Vous n'aurez plus jamais à refaire ces étapes.

Chargement de la table “Lexique.org”

Tout se déroule dans Rstudio. Assurez vous de bien travailler dans le projet “lexique” créé dans la section “préparation”. Après un redémarrage de rstudio, utilisez le menu `File/Recent Projects` pour retrouver ce projet.

- Cliquez le menu “File / New File / R Notebook”. Un document “Untitled” apparait dans le fenetre en haut à gauche.
- C’est dans ce document que nous allons entrer du code R:

  Déplacez vous à la ligne 5 (juste après la ligne contenant ‘—’), et cliquez sur le bouton *Insert* puis choisisez *R* (vous pouvez aller plus vite en appuyant sur *Ctrl+Alt+I*).

  Copiez les trois lignes suivantes dans le bloc de code qui vient d’être créé.

```
require(tidyverse)  # you must have ran "install.packages('tidyverse')" earlier
lexique = read_delim("Lexique382.tsv", delim="\t")
head(lexique, 25)
```

```
## # A tibble: 25 x 35
##    ortho  phon   lemme cgram genre nombre freqlemfilms2 freqlemlivres freqfilms2
##    <chr>  <chr>  <chr> <chr> <chr> <chr>          <dbl>         <dbl>      <dbl>
##  1 a      a      a     NOM   m     <NA>           81.4          58.6       81.4
##  2 a      a      avoir AUX   <NA>  <NA>         18559.        12801.      6351.
##  3 a      a      avoir VER   <NA>  <NA>         13572.         6426.      5498.
##  4 a cap~ akapE~ a ca~ ADV   <NA>  <NA>             0.04          0.07       0.04
##  5 a cap~ akapE~ a ca~ ADV   <NA>  <NA>             0.04          0.07       0.04
##  6 a con~ ak§tR~ a co~ ADV   <NA>  <NA>             0             0.27       0
##  7 a for~ afORs~ a fo~ ADV   <NA>  <NA>             0.04          0.88       0.04
##  8 a gio~ adZjO~ a gi~ ADV   <NA>  <NA>             0             0.27       0
##  9 a jeun aZ1    à je~ ADV   <NA>  <NA>             1.45          3.85       0.18
## 10 a l'i~ al5st~ a l'~ PRE   <NA>  <NA>             0.26          0          0.26
## # ... with 15 more rows, and 26 more variables: freqlivres <dbl>,
## #   infover <chr>, nbhomogr <dbl>, nbhomoph <dbl>, islem <dbl>,
## #   nblettres <dbl>, nbphons <dbl>, cvcv <chr>, p_cvcv <chr>, voisorth <dbl>,
## #   voisphon <dbl>, puorth <dbl>, puphon <dbl>, syll <chr>, nbsyll <dbl>,
## #   cv-cv <chr>, orthrenv <chr>, phonrenv <chr>, orthosyll <chr>,
## #   cgramortho <chr>, deflem <dbl>, defobs <dbl>, old20 <dbl>, pld20 <dbl>,
## #   morphoder <chr>, nbmorph <dbl>
```

Puis, en faisant bien attention que le curseur soit à l’intérieur du bloc de code, cliquer sur *Run / Run current chunk* (ou bien appuyer sur la petite *fleche verte*, ou sur *Ctrl-Shift-Enter*). Vous devriez voir les premières lignes de la table lexique s’afficher. Celle doit également apparaitre dans l’onglet “Environment” en haut à droite.

Conseil: Lorsque vous quitterez rstudio, acceptez la proposition “sauvegarder le worskpace”. Ainsi, la prochaine fois que vous ouvrirez le projet ‘lexique’ dans rstudio, la table sera déjà présente dans l’environnement, et il ne sera donc pas nécessaire d’exécuter les lignes ci-dessus (`read_delim...`).

Sélection de mots entiers

Supposons que vous vouliez extraire les lignes de la table lexique correspondant, par exemple, aux mots ‘bateau’, ‘avion’, ‘maison’, ‘arbre’. Le code suivant fait précisemment cela:

```
items <- c('bateau', 'avion', 'maison', 'arbre')

selection <- subset(lexique, ortho %in% items)

head(selection)
```

```
## # A tibble: 4 x 35
##   ortho  phon  lemme  cgram genre nombre freqlemfilms2 freqlemlivres freqfilms2
##   <chr>  <chr> <chr>  <chr> <chr> <chr>          <dbl>         <dbl>      <dbl>
```

```
## 1 arbre  aRbR  arbre  NOM   m     s              81.7         209.        49.3
## 2 avion  avj§  avion  NOM   m     s             128.           78.0      106.
## 3 bateau bato  bateau NOM   m     s             125.           82.4      107.
## 4 maison mEz§  maison NOM   <NA>  s             606.          575.       570.
## # ... with 26 more variables: freqlivres <dbl>, infover <chr>, nbhomogr <dbl>,
## #   nbhomoph <dbl>, islem <dbl>, nblettres <dbl>, nbphons <dbl>, cvcv <chr>,
## #   p_cvcv <chr>, voisorth <dbl>, voisphon <dbl>, puorth <dbl>, puphon <dbl>,
## #   syll <chr>, nbsyll <dbl>, cv-cv <chr>, orthrenv <chr>, phonrenv <chr>,
## #   orthosyll <chr>, cgramortho <chr>, deflem <dbl>, defobs <dbl>, old20 <dbl>,
## #   pld20 <dbl>, morphoder <chr>, nbmorph <dbl>
```

Vous pouvez inspecter le contenu de la table `selection` en cliquant sur son nom dans l'onglet *Environment* situé dans la fenetre en haut à droite de rstudio.

Vous pouvez également sauvegarder les résultats obtenus dans un fichier, avec la commande suivante:

```
write_tsv(selection, 'selection.tsv')
```

Notez que les fichiers ayant l'extension `csv` (tab-separated-values) peuvent être ouverts avec Excel, ou OpenOffice Cal, ou même avec n'importe quel éditeur de texte. Note: le package `readr` de R fournit aussi des fonctions `write_excel_csv` et `write_excel_csv2` qui peuvent intéresser certains.

Si vous avez une liste de mots plus longues, il serait fastidieux d'écrire la ligne `items <- ...`. Plus simplement vous pouvez utiliser:

```
items = scan(what='characters')
```

Et copier la liste de mots. Entrer une ligne vide termine le processus.

La fonction `scan` permet aussi de lire la liste dans un fichier externe.

Sélection par critères

Supposons que vous vouliez sélectionner tous les noms de 5 lettres, singuliers, de fréquence lexicale (films) comprise entre 10 et 100. Voici la ligne magique:

```
selection = subset(lexique, cgram=='NOM' & nombre != 'p' & nblettres==5 & freqlivres > 10 & freqlivres <
head(selection)
```

```
## # A tibble: 6 x 35
##   ortho phon  lemme cgram genre nombre freqlemfilms2 freqlemlivres freqfilms2
##   <chr> <chr> <chr> <chr> <chr> <chr>          <dbl>         <dbl>      <dbl>
## 1 abîme abim  abîme NOM   m     s               6.01          20.6       5.03
## 2 achat aSa   achat NOM   m     s               9.75          17.0       5.2
## 3 acier asje  acier NOM   m     s              13.9           34.5      13.8
## 4 adieu adj2  adieu NOM   m     s              44.4           38.0      36.9
## 5 affût afy   affût NOM   m     s               1.42          11.4       1.36
## 6 agent aZ@   agent NOM   m     s             118.            39.3      92.4
## # ... with 26 more variables: freqlivres <dbl>, infover <chr>, nbhomogr <dbl>,
## #   nbhomoph <dbl>, islem <dbl>, nblettres <dbl>, nbphons <dbl>, cvcv <chr>,
## #   p_cvcv <chr>, voisorth <dbl>, voisphon <dbl>, puorth <dbl>, puphon <dbl>,
## #   syll <chr>, nbsyll <dbl>, cv-cv <chr>, orthrenv <chr>, phonrenv <chr>,
## #   orthosyll <chr>, cgramortho <chr>, deflem <dbl>, defobs <dbl>, old20 <dbl>,
## #   pld20 <dbl>, morphoder <chr>, nbmorph <dbl>
```

Sélection par “pattern”

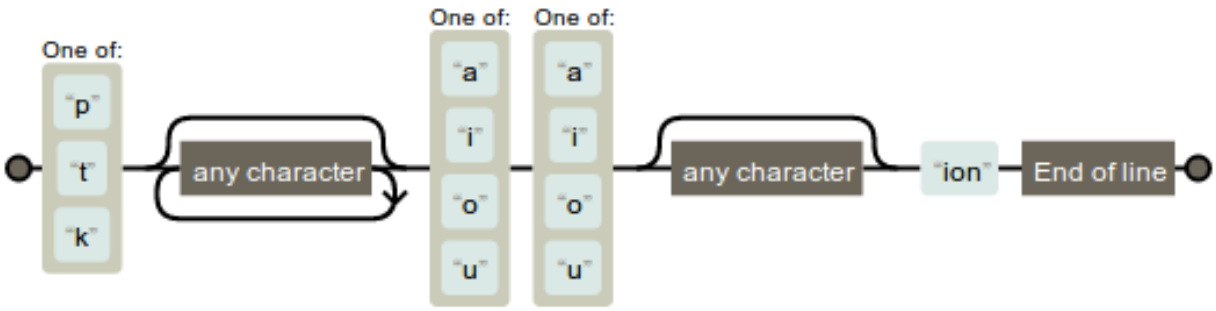
Les expressions régulières

Les expressions réulières, ou **regex**, sont des “patterns” qui permettent de rechercher des mots ayant certaines propriétés. Par exemple n’importe `a.b` désigne un mot contenant un `a` et un `b` séparés par une lettre quelonque. Voici d’autre exemples:

- `^maison$` : recherche le mot “maison” exactement
- `^anti` : recherche tous les mots commençant par “anti”
- `^jour$|^nuit$|^matin$|^soir$` : “jour” ou “nuit” ou “matin” ou “soir” (permet de rechercher une liste de mots)
- `ion` : recherche les mots qui contiennent la chaine “ion” dans n’importe quelle position
- `ion$` : mots se terminant par “ion”
- `^pr` : mots commençant par “pr”
- `^p..r$` : mots de quatre lettres commençant par “p”, finisant pas “r”
- `^p.*r$` : mots commencant par ‘”p’ et finissant par “r”
- `[aeiou][aeiou]` : mots contenant 2 voyelles successives
- `^[aeiou]` : mots commençant par une voyelle
- `^[^aeriou]` : mots ne commençant pas par une voyelle

Il existe de nombreux tutoriaux sur les regex sur le web, notamment celui-ci. La bible sur le sujet est le livre *Mastering Regular Expressions* de Jeff Friedl.

Une expression régulière décrit un automate de transitions à états finis. Le site https://regexper.com/ vous permet de visualiser l’automate associé à une regex. Par exemple `[ptk].*[aiou][aiou].?ion$` correspond à l’automate fini:



Recherches dans R avec grepl

R permet d’effectuer des recherches par pattern grâce à la fonction `grepl`. La syntaxe est `grepl(`*regex*, *variable*`)` pour rechercher les lignes où la variable “matche” la regex (Voir la doc R de grepl.

Cette fonction permet de localiser les lignes qui ‘matchent’ une expression, ou bien, en la niant avec le signe `!`, de supprimer des lignes qui matchent un pattern.

Voici quelques exemples:

- Pour obtenir tous les mots qui finissent par `tion` :

```
lexique %>% filter(grepl("tion$", ortho)) -> selection2
head(selection2)
```

```
## # A tibble: 6 x 35
##   ortho   phon  lemme  cgram genre nombre freqlemfilms2 freqlemlivres freqfilms2
##   <chr>   <chr> <chr>  <chr> <chr> <chr>          <dbl>         <dbl>      <dbl>
## 1 abdica~ abdi~ abdic~ NOM   f     s               0.05          1.96       0.05
```

```
## 2 abduct~ abdy~ abduc~ NOM   f     s               0.05          0          0.05
## 3 aberra~ abER~ aberr~ NOM   f     s               1.16          4.46       0.89
## 4 abject~ abZE~ abjec~ NOM   f     s               0.51          2.3        0.37
## 5 abjura~ abZy~ abjur~ NOM   f     s               0             0.47       0
## 6 ablati~ abla~ ablat~ NOM   f     s               0.45          1.35       0.43
## # ... with 26 more variables: freqlivres <dbl>, infover <chr>, nbhomogr <dbl>,
## #   nbhomoph <dbl>, islem <dbl>, nblettres <dbl>, nbphons <dbl>, cvcv <chr>,
## #   p_cvcv <chr>, voisorth <dbl>, voisphon <dbl>, puorth <dbl>, puphon <dbl>,
## #   syll <chr>, nbsyll <dbl>, cv-cv <chr>, orthrenv <chr>, phonrenv <chr>,
## #   orthosyll <chr>, cgramortho <chr>, deflem <dbl>, defobs <dbl>, old20 <dbl>,
## #   pld20 <dbl>, morphoder <chr>, nbmorph <dbl>
```

Encore une fois, vous pouvez sauvegarder ces résultats avec:

```
write_tsv(selection2, 'mots-en-tion.tsv')
```

- Pour lister tous les mots contenant un cluster de consonnes plosives, mais pas debut de mot:

```
lexique %>% filter(grepl('.[ptkbdg][ptkbdg]', phon)) -> selection3
head(selection3)
```

```
## # A tibble: 6 x 35
##   ortho   phon  lemme  cgram genre nombre freqlemfilms2 freqlemlivres freqfilms2
##   <chr>   <chr> <chr>  <chr> <chr> <chr>          <dbl>         <dbl>      <dbl>
## 1 abdica~ abdi~ abdic~ NOM   f     s               0.05          1.96       0.05
## 2 abdica~ abdi~ abdic~ NOM   f     p               0.05          1.96       0
## 3 abdiqua abdi~ abdiq~ VER   <NA>  <NA>            0.47          2.77       0
## 4 abdiqu~ abdi~ abdiq~ VER   <NA>  <NA>            0.47          2.77       0
## 5 abdiqu~ abdi~ abdiq~ VER   <NA>  <NA>            0.47          2.77       0
## 6 abdiqu~ abdi~ abdiq~ VER   <NA>  <NA>            0.47          2.77       0
## # ... with 26 more variables: freqlivres <dbl>, infover <chr>, nbhomogr <dbl>,
## #   nbhomoph <dbl>, islem <dbl>, nblettres <dbl>, nbphons <dbl>, cvcv <chr>,
## #   p_cvcv <chr>, voisorth <dbl>, voisphon <dbl>, puorth <dbl>, puphon <dbl>,
## #   syll <chr>, nbsyll <dbl>, cv-cv <chr>, orthrenv <chr>, phonrenv <chr>,
## #   orthosyll <chr>, cgramortho <chr>, deflem <dbl>, defobs <dbl>, old20 <dbl>,
## #   pld20 <dbl>, morphoder <chr>, nbmorph <dbl>
```

- L'opérateur `filter` peut être appelé plusieurs fois pour affiner progressivement la recherche.

  Par exemple, pour obtenir tous les mots de 8 lettres qui ne finissent pas `ent`:

```
lexique %>% filter(nblettres == 8) %>% filter(!grepl("ent$", ortho)) -> selection4
head(selection4)
```

```
## # A tibble: 6 x 35
##   ortho    phon  lemme cgram genre nombre freqlemfilms2 freqlemlivres freqfilms2
##   <chr>    <chr> <chr> <chr> <chr> <chr>          <dbl>         <dbl>      <dbl>
## 1 a giorno adZj~ a gi~ ADV   <NA>  <NA>            0             0.27       0
## 2 a priori apRi~ a pr~ ADV   <NA>  <NA>            1.04          3.85       0.63
## 3 a priori apRi~ a pr~ NOM   m     <NA>            0.41          0.47       0.41
## 4 abaissai abEsE abai~ VER   <NA>  <NA>            4.93         18.0        0.1
## 5 abaisser abese abai~ VER   <NA>  <NA>            4.93         18.0        1.09
## 6 abaisses abEs  abai~ VER   <NA>  <NA>            4.93         18.0        0.16
## # ... with 26 more variables: freqlivres <dbl>, infover <chr>, nbhomogr <dbl>,
## #   nbhomoph <dbl>, islem <dbl>, nblettres <dbl>, nbphons <dbl>, cvcv <chr>,
## #   p_cvcv <chr>, voisorth <dbl>, voisphon <dbl>, puorth <dbl>, puphon <dbl>,
## #   syll <chr>, nbsyll <dbl>, cv-cv <chr>, orthrenv <chr>, phonrenv <chr>,
## #   orthosyll <chr>, cgramortho <chr>, deflem <dbl>, defobs <dbl>, old20 <dbl>,
```

```
## #   pld20 <dbl>, morphoder <chr>, nbmorph <dbl>
```