Interroger Lexique avec R

Christophe Pallier

Contents

Introduction

Ce document montre comment interroger et manipuler la base Lexique avec la langage R. Merci de signaler d'éventuelles erreurs à christophe@pallier.org

Il fait parti du cours Programmation pour les Sciences Cognitives ou vous pourrez trouver d'autres documents potentiellement intéressants.

Prérequis

1. S'ils sont pas déjà installés sur votre ordinateur, vous devrez installer les logiciels R et R-Studio Desktop.
2. Téléchargez http://www.lexique.org/databases/Lexique382/Lexique382.zip
3. Créer un répertoire `Lexique` puis désippez-y 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; cliquez sur le bouton `create project`.

   Dans la fenêtre en bas à droite,parmis la liste de fichier sous l'onglet *Files*, vous devez voir `Lexique382.tsv`. Si ce 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 appuez sur 'Entrée'

   install.packages('tidyverse')

Vous êtes prêt !

Chargement de la table "Lexique.org"

Tout se déroule dans Rstudio. Soyez sûr de bien travailler dans le projet "lexique" créé dans la section "prérequis". Après un rdemarrage de rstsudio, utilisez le menu `File/Recent Projects`.

- 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 (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 deux lignes suivantes dans le bloc de code qui vient d'être créé; 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*)

```
require(tidyverse)  # you must have ran "install.packages('tidyverse')" earlier
```

```
## Loading required package: tidyverse
```

```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v ggplot2 3.1.0       v purrr   0.2.5
## v tibble  2.1.1       v dplyr   0.8.0.1
## v tidyr   0.8.3       v stringr 1.3.1
## v readr   1.3.1       v forcats 0.3.0
```

```
## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
lexique = read_delim("Lexique382.tsv", delim="\t")
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   ortho = col_character(),
##   phon = col_character(),
##   lemme = col_character(),
##   cgram = col_character(),
##   genre = col_character(),
##   nombre = col_character(),
##   infover = col_character(),
##   cvcv = col_character(),
##   p_cvcv = col_character(),
##   syll = col_character(),
##   `cv-cv` = col_character(),
##   orthrenv = col_character(),
##   phonrenv = col_character(),
##   orthosyll = col_character(),
##   cgramortho = col_character(),
##   morphoder = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

Puis insérer et executer la ligne suivante pour afficher les premières lignes de la table *lexique* :

```
head(lexique, 25)
```

```
## # A tibble: 25 x 35
##    ortho phon  lemme cgram genre nombre freqlemfilms2 freqlemlivres
##    <chr> <chr> <chr> <chr> <chr> <chr>          <dbl>         <dbl>
##  1 a     a     a     NOM   m     <NA>           81.4          58.6
##  2 a     a     avoir AUX   <NA>  <NA>        18559.        12801.
##  3 a     a     avoir VER   <NA>  <NA>        13572.         6426.
##  4 a ca~ akap~ a ca~ ADV   <NA>  <NA>            0.04          0.07
##  5 a ca~ akap~ a ca~ ADV   <NA>  <NA>            0.04          0.07
##  6 a co~ ak§t~ a co~ ADV   <NA>  <NA>            0             0.27
##  7 a fo~ afOR~ a fo~ ADV   <NA>  <NA>            0.04          0.88
##  8 a gi~ adZj~ a gi~ ADV   <NA>  <NA>            0             0.27
```

```
##  9 a je~ aZ1   à je~ ADV   <NA>  <NA>            1.45          3.85
## 10 a l'~ al5s~ a l'~ PRE   <NA>  <NA>            0.26          0
## # ... with 15 more rows, and 27 more variables: freqfilms2 <dbl>,
## #   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 de mots entiers

Supposons que vous vouliez extraire les lignes de lexiques 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
##   <chr> <chr> <chr> <chr> <chr> <chr>          <dbl>         <dbl>
## 1 arbre aRbR  arbre NOM   m     s               81.7         209.
## 2 avion avj§  avion NOM   m     s              128.           78.0
## 3 bate~ bato  bate~ NOM   m     s              125.           82.4
## 4 mais~ mEz§  mais~ NOM   <NA>  s              606.          575.
## # ... with 27 more variables: freqfilms2 <dbl>, 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 Environnment dans la fenetre en haut à droite dans rstudio.

Vous pouvez également sauvegarder cette novuelle table dans un fichier avec la ligne suivante:

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

Notez que les fichiers avec l'extension `csv` (tab-separated-values) peuvent être ouverts avec Excel.

Sélection par critères

Supposont que vous vouliez sélectionner les noms de 5 lettres, singuliers, de fréquence lexicale (films) comprisent entre 10 et 100.

```
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
##   <chr> <chr> <chr> <chr> <chr> <chr>          <dbl>         <dbl>
```

```
## 1 abîme abim  abîme NOM   m     s               6.01          20.6
## 2 achat aSa   achat NOM   m     s               9.75          17.0
## 3 acier asje  acier NOM   m     s              13.9           34.5
## 4 adieu adj2  adieu NOM   m     s              44.4           38.0
## 5 affût afy   affût NOM   m     s               1.42          11.4
## 6 agent aZ@   agent NOM   m     s             118.            39.3
## # ... with 27 more variables: freqfilms2 <dbl>, 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"

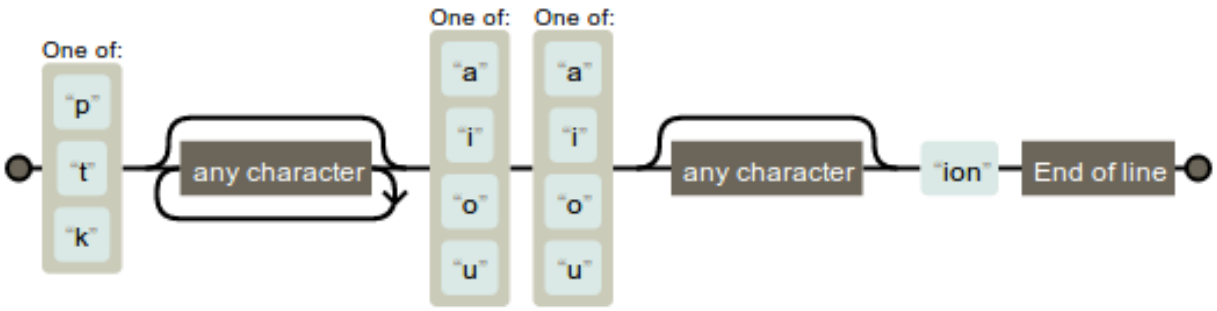
Les expressions régulières

Les expressions regulières, ou **regex**, sont des "patterns" qui permettent de rechercher des mots ayant certaines propriétés. Voici quelques exemples d'expressions régulières:

- `^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 beaucoup de 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 regex décrit un automate de transition à états finis. Le site https://regexper.com/ vous permet de visualiser l'automate associé à votre 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
##   <chr> <chr> <chr> <chr> <chr> <chr>          <dbl>         <dbl>
## 1 abdi~ abdi~ abdi~ NOM   f     s               0.05          1.96
## 2 abdu~ abdy~ abdu~ NOM   f     s               0.05          0
## 3 aber~ abER~ aber~ NOM   f     s               1.16          4.46
## 4 abje~ abZE~ abje~ NOM   f     s               0.51          2.3
## 5 abju~ abZy~ abju~ NOM   f     s               0             0.47
## 6 abla~ abla~ abla~ NOM   f     s               0.45          1.35
## # ... with 27 more variables: freqfilms2 <dbl>, 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>
```

Voous pouvez sauvegarder ces résultats avec la commande `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
##   <chr> <chr> <chr> <chr> <chr> <chr>          <dbl>         <dbl>
## 1 abdi~ abdi~ abdi~ NOM   f     s               0.05          1.96
## 2 abdi~ abdi~ abdi~ NOM   f     p               0.05          1.96
## 3 abdi~ abdi~ abdi~ VER   <NA>  <NA>            0.47          2.77
## 4 abdi~ abdi~ abdi~ VER   <NA>  <NA>            0.47          2.77
## 5 abdi~ abdi~ abdi~ VER   <NA>  <NA>            0.47          2.77
## 6 abdi~ abdi~ abdi~ VER   <NA>  <NA>            0.47          2.77
## # ... with 27 more variables: freqfilms2 <dbl>, 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 appeler plusieurs fois pour affiner progressivement la recherche. Par exemple, pour obtenir tous les mots de 8 lettres qui ne finissent pas 'ent':

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

```
## # A tibble: 6 x 35
##   ortho phon  lemme cgram genre nombre freqlemfilms2 freqlemlivres
##   <chr> <chr> <chr> <chr> <chr> <chr>          <dbl>         <dbl>
## 1 a gi~ adZj~ a gi~ ADV   <NA>  <NA>            0             0.27
## 2 a pr~ apRi~ a pr~ ADV   <NA>  <NA>            1.04          3.85
## 3 a pr~ apRi~ a pr~ NOM   m     <NA>            0.41          0.47
## 4 abai~ abEsE abai~ VER   <NA>  <NA>            4.93         18.0
## 5 abai~ abese abai~ VER   <NA>  <NA>            4.93         18.0
## 6 abai~ abEs  abai~ VER   <NA>  <NA>            4.93         18.0
## # ... with 27 more variables: freqfilms2 <dbl>, 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>
```