RÉPUBLIQUE DU SÉNÉGAL

Un Peuple - Un But - Une Foi

Ministère de l'Économie, du Plan et de la Coopération

Agence Nationale de la Statistique et de la Démographie (ANSD)

École Nationale de la Statistique et de l'Analyse Économique (ENSAE | Pierre Ndiaye)

TP4

Rédigé par :

FOGWOUNG DJOUFACK Sarah-Laure

NIASS Ahmadou

NGUEMFOUO NGOUMTSA Célina

SENE Malick

Élèves Ingénieurs Statisticiens Économistes

Sous la supervision de :

M. Aboubacar HEMA

Research Analyst

Année scolaire : 2024/2025

TABLE DE MATIERES

Contents

CONSIGNE DU TP4

Ce TP consiste à faire correspondre les codes **ADM3_PCODE** de chaque commune des pays **Burkina Faso** et **Niger** entre deux bases de données :

- **Base EHCVM** : Contient des informations détaillées sur les conditions de vie, les revenus, et d'autres aspects socio-économiques des ménages.
- **Base HDX** : Contient des données administratives avec les codes **ADM3_PCODE**, souvent utilisées pour le suivi humanitaire et les informations géographiques et administratives des communes.

Le **problème principal** est que les noms des communes ne sont pas toujours écrits de la même façon dans les deux bases. Pour résoudre cela, il faut regarder aux niveaux département et région afin de trouver les correspondances exactes.

L'objectif à atteindre est donc de **fusionner les deux bases de données en s'assurant que les codes ADM3_PCODE de chaque commune soient bien alignés**.

STEP 1: IMPORTATION DES PACKAGES

Dans cette étape, nous importons plusieurs packages qui nous permettront d'effectuer le travail demandé. Voici un aperçu de ce que chaque package apporte :

- **Le package haven** : est particulièrement utile pour importer des fichiers `.dta` (Stata) en R.
- **Le package readxl** : permet de lire les fichiers Excel (formats .xls et .xlsx).
- **Le package dplyr** :permet de filtrer, sélectionner, regrouper, trier, et effectuer d'autres opérations de transformation, facilitant ainsi la manipulation des données.
- **Le package stringr** : fournit des fonction pour la manipublation des chaines de caractères (traitement de textes, recherche/substitution/modification des chaines de caractères ...)
- **Le package sf** : fournit des outils pour lire, manipuler et analyser des données spatiales
- **Le package tidyverse** : collection de packages qui facilitent la manipulation, la visualisation et l'analyse des données. Il inclut des packages comme ggplot2 (pour la visualisation), dplyr (pour la manipulation des données), tidyr (pour le nettoyage des données), et plus encore.
- **Le package stringi** : une version plus complète du package stringr pour la manipulation des chaînes de caractères. Il propose une plus grande variété de fonctions pour le traitement de texte et est conçu pour être plus performant dans des cas plus complexes.

```
library(haven)     # Pour lire le fichier Stata
library(readxl)    # Pour lire le fichier Excel
library(dplyr)     # Pour la manipulation des données
library(stringr)   # Pour manipuler les chaînes de caractères
library(sf) # pour la gestion des données spatiales
library(tidyverse) #manipulation,visualisation et analyse
library(stringi) # Version plus complete de stringr
```

STEP 2: TRAITEMENT POUR LE NIGER

2-a CHARGEMENT DES BASES DE DONNÉES

Nous commençons par charger les bases de données en utilisant les packages appropriés. La première base est au format Stata (utilisée pour les données de l'EHCVM), et la deuxième est au format Excel (utilisée pour les données de la base HDX).

```
ehcvm_raw <- haven::read_dta(
  "../Donnees/s00_me_ner2021.dta"
  )  # Chargement de la base de données EHCVM (format Stata)
hdx_raw <- readxl::read_excel(
  "../Donnees/ner_admgz_ignn_20230720.xlsx"
  ) # Chargement de la base de données HDX (format Excel)
```

2-b TRAITEMENT DES COMMUNES DANS LA BASE EHCVM

Nous nettoyons les noms des communes dans la base EHCVM en appliquant plusieurs transformations : - Conversion des codes des communes en labels. - Mise en minuscules, suppression des accents, des apostrophes et des caractères non alphanumériques. - Suppression des espaces en début et en fin, ainsi que des espaces multiples. - Correction des noms de certaines communes.

```
ehcvm <- ehcvm_raw %>%
  dplyr::mutate(
    commune_label = as_factor(s00q03),
    commune_clean = commune_label |>
      stringr::str_to_lower() |>
      (\(x) iconv(x, to = "ASCII//TRANSLIT"))() |>
      stringr::str_replace_all("'", "") |>
      stringr::str_replace_all("[^[:alnum:] ]", " ") |>
      stringr::str_trim() |>
      stringr::str_replace_all("\\barrondissement\\b", "") |>
      stringr::str_replace_all("\\b1\\b", "i") |>
      stringr::str_replace_all("\\b2\\b", "ii") |>
```

```
      stringr::str_replace_all("\\b3\\b", "iii") |>
      stringr::str_replace_all("\\b4\\b", "iv") |>
      stringr::str_replace_all("\\b5\\b", "v") |>
      stringr::str_squish()
  )
```

2-c CORRECTION ET TRAITEMENT DES COMMUNES DANS LA BASE HDX

De même, dans la base HDX, nous corrigeons certains noms de communes en fonction des départements et régions correspondants. Une série de règles est appliquée pour certains cas spécifiques. Nous nettoyons également les noms de communes dans la base HDX de la même manière que pour EHCVM.

```
hdx <- hdx_raw %>%
  dplyr::mutate(
    ADM3_FR = dplyr::case_when(
      ADM3_FR ==
        "Tibiri" & ADM1_FR == "Dosso" ~ "Tibiri Doutchi",
      ADM3_FR ==
        "Tibiri" & ADM1_FR == "Maradi" ~ "Tibiri Maradi",
      ADM3_FR ==
        "Gangara" & ADM1_FR == "Maradi" ~ "Gangara Gazaoua",
      ADM3_FR ==
        "Gangara" & ADM1_FR == "Zinder" ~ "Gangara Tanout",
      TRUE ~ ADM3_FR
    ),

    # Création de la colonne corrigée commune_clean
    commune_clean = ADM3_FR |>
      stringr::str_to_lower() |>
      (\(x) iconv(x, to = "ASCII//TRANSLIT"))() |>
      stringr::str_replace_all("'", "") |>
      stringr::str_replace_all("[^[:alnum:] ]", " ") |>
      stringr::str_trim() |>
      stringr::str_squish()
  )
```

2-d JOINTURE DES BASES

Nous effectuons une jointure des deux bases (EHCVM et HDX) en utilisant la colonne “commune_clean” comme clé de correspondance. Ici, nous utilisons une jointure gauche (left_join) pour conserver toutes les communes de la base EHCVM.

```
ehcvm_hdx_merged <- dplyr::left_join(
  ehcvm, hdx, by = "commune_clean")
```

2-e VERIFICATION DES COMMUNES NON APPARIEES

Une fois la jointure effectuée, nous vérifions les communes qui ne se sont pas appariées. Nous filtrons et affichons les lignes où “ADM3_FR” est NA, ce qui signifie qu’il n’y a pas eu de correspondance.Et on verra qu’il y a eu des correspondances partout.

```
non_matchees <- dplyr::filter(
  ehcvm_hdx_merged, is.na(ADM3_FR)) %>%
  dplyr::distinct(s00q03, commune_clean)

non_matchees
```

```
## # A tibble: 0 x 2
## # i 2 variables: s00q03 <dbl+lbl>, commune_clean <chr>
```

On constate que toutes les communes sont matchées.

2-f Sauvegarde de la base obtenue

La base obtenue a été sauvegardée.

```
haven::write_dta(ehcvm_hdx_merged, "../Sorties/EHCVM_HDX_Niger.dta")
```

STEP 3: TRAITEMENT POUR LE BURKINA FASO

3-a CHARGEMENT DES BASES DE DONNÉES

Nous commençons par charger les bases de données en utilisant les packages appropriés. Comme précedement, la première base est au format Stata (utilisée pour les données de l'EHCVM), et la deuxième est au format Excel (utilisée pour les données de la base HDX).

```
ehcvm_raw <- haven::read_dta(
  "../Donnees/s00_me_bfa2021.dta")
shape_raw <- readxl::read_excel(
  "../Donnees/bfa_adminboundaries_tabulardata.xlsx",
  sheet = "ADM3")
```

3-b CORRECTION DES ERREURS D'ORTHOGRAPHE

L'étape suivant est la correction des erreurs des erreurs d'orthographe dans la base EHCVM.

Après avoir analysé les noms des communes, nous avons vu que dans la base ehcvm certaines communes n'ont pas de correspondance dans la base hdx, en vérfiant nous avons vu que cela étaient du à des différences orthographiques. Vu qu'elles n'étaient pas nombreuses on a voulu les corriger manuellement. Par ailleurs, pour les arrondissements numérotés de 1 à 12, nos recherches ont montré qu'ils appartiennent tous à la commune de Ouagadougou. Nous avons donc choisi de les uniformiser en remplaçant leur nom par "Ouagadougou" dans notre base de données. Pour ce faire le data frame correction_map a été créé:

Création du correction_map

```
correction_map <- data.frame(
  commune_label = c(
    "Zeguedeguin","Bondigui","Absouya", "Bondokuy",
    "Bomborokuy", "Bittou", "Bokin", "Boudry",
    "Bobo Dioulasso-Konsa", "Bobo Dioulasso-Dô", "Arbole",
    "Dapelogo", "Dissin", "Fada N'gourma", "Gounghin",
    "Kokoloko", "Gourcy", "La-Todin", "Karankasso-Vigue",
    "Sanga", "Meguet", "Soaw", "Samorogouan", "Sabce",
```

```
    "Tanghin Dassouri", "Oury", "Cassou",
    "Arrondissement 1", "Arrondissement 2",
    "Arrondissement 3", "Arrondissement 4",
    "Arrondissement 5", "Arrondissement 6",
    "Arrondissement 7", "Arrondissement 8",
    "Arrondissement 9", "Arrondissement 10",
    "Arrondissement 11", "Arrondissement 12",
    "Arrondissement N 1", "Arrondissement N 2",
    "Arrondissement N 3", "Arrondissement N 4",
    "Arrondissement N 5", "Arrondissement N 6",
    "Arrondissement N 7"),
  corrected_name = c(
    "Senguènèga","Bondokui","Ambsouya", "Bondokui",
    "Bomborokui", "Bitou", "Boken", "Boudri",
    "Bobo-Dioulasso", "Bobo-Dioulasso", "Arbollé",
    "Dapeolgo", "Dissihn", "Fada Ngourma", "Gounguen",
    "Kokologo", "Goursi", "La-Toden", "Karangasso-Vigué",
    "Sangha", "Mégué", "Soa", "Samôgôgouan", "Sabsé",
    "Tanguen-Dassouri", "Ouri", "Kassou", "Ouagadougou",
    "Ouagadougou", "Ouagadougou", "Ouagadougou",
    "Ouagadougou", "Ouagadougou", "Ouagadougou",
    "Ouagadougou", "Ouagadougou", "Ouagadougou",
    "Ouagadougou", "Ouagadougou", "Ouagadougou",
    "Ouagadougou", "Ouagadougou", "Ouagadougou")
)
```

- **Application de la correction à la base EHCVM**

```
ehcvm_corrected <- ehcvm_raw %>%
  dplyr::left_join(correction_map,
                   by = c("s00q03" = "commune_label")) %>%
  dplyr::mutate(s00q03 = ifelse(is.na(corrected_name),
                                s00q03, corrected_name)) %>%
  dplyr::select(-corrected_name)
```

3-c NETTOYAGE DES DONNES

Nous nettoyons ensuite les noms des communes et départements des bases EHCVM et HDX, notament en tenant compte de la casse, supprimant les accents, caractères spéciaux, apostrophes et espaces inutiles. Pour ce faire, la fonction *clean_names* a été crée.

- **Création de la fonction**

```
clean_names <- function(name) {
  name %>%
    stringr::str_to_lower() %>%
    iconv(to = "ASCII//TRANSLIT") %>%
    stringr::str_replace_all("'", "") %>%
    stringr::str_replace_all("[-]", " ") %>%
    stringr::str_replace_all("[^[:alnum:] ]", " ") %>%
    stringr::str_trim() %>%
    stringr::str_squish()
}
```

- **Traitement de la base EHCVM**

```
# EHCVM : convertion des codes en labels et nettoyage
ehcvm <- ehcvm_corrected %>%
  dplyr::mutate(
    commune_label = forcats::as_factor(s00q03),
    departement_label = forcats::as_factor(s00q02),
    commune_clean = clean_names(commune_label),
    departement_clean = clean_names(departement_label)
  )
```

- **Traitement de la base HDX**

```
# Shape Data : nettoyage des communes et départements
shape <- shape_raw %>%
  dplyr::mutate(
    commune_clean = clean_names(ADM3_FR),
    departement_clean = clean_names(ADM2_FR)
  )
```

3-d FUSION

Une fois toutes les données nettoyées, nous effectuons une jointure sur la base EHCVM afin d'obtenir les noms corrects des communes dans la base.

Toutefois, il existe des communes necessitant des vérifications par département : il s'agit des communes boussouma, namissiguima et thiou.

```
communes_ambigues <- c("boussouma", "namissiguima", "thiou")
```

Une fois ces communes identifiées, un premier merge sera effectué sur les autres communes.

```
# Premier merge
ehcvm_shape_merged <- ehcvm %>%
  dplyr::filter(!commune_clean %in% communes_ambigues) %>%
  dplyr::left_join(shape, by = "commune_clean")
```

Une fusion est à présent faite sur les communes ambigües en tenant compte des départements.

```
# Fusion pour les communes ambigües
ehcvm_shape_merged_ambigues <- ehcvm %>%
  dplyr::filter(commune_clean %in% communes_ambigues) %>%
  dplyr::left_join(shape, by = c(
    "commune_clean", "departement_clean"))

# Combiner les deux bases après fusion
ehcvm_shape_final <- dplyr::bind_rows(
  ehcvm_shape_merged, ehcvm_shape_merged_ambigues)
```

Une fois la fusion terminée, finalisons le travail en faisant des vérifications et en supprimant les colonnes inutiles.

3-e VERFICATION

Vérifions si toutes les communes ont été vusionnées

```r
non_matchees <- ehcvm_shape_final %>%
  dplyr::filter(is.na(ADM3_PCODE)) %>%
  dplyr::distinct(s00q03, departement_label, commune_clean)

head(non_matchees)
```

```
## # A tibble: 0 x 3
## # i 3 variables: s00q03 <chr>, departement_label <fct>, commune_clean <
```

Toutes les communes ont été matchées. La dernière étape, la suppression des colonnes temporaires peut enfin être éffectuée.

3-f SUPPRESSION DES COLONNES TEMPORAIRES

Les colonnes temporaires à supprimer sont: communes_clean, departement_clean et commune_label.

```r
ehcvm_final <- ehcvm_shape_final %>%
  dplyr::select(
    -commune_clean, -departement_clean.x, -departement_clean.y,
    -departement_clean, -commune_label, -departement_label)
```

Voici un aperçu de la base finale:

```r
print(head(ehcvm_final))
```

```
## # A tibble: 6 x 43
##    hhid grappe menage vague hhweight s00q00    s00q01  s00q02  s00q03 s0
##   <dbl>  <dbl>  <dbl> <dbl>   <dbl> <dbl+lbl>  <dbl+l> <dbl+lb> <chr>  <
## 1 586005   586     5    2     439 2 [Burkin~ 2 [Bou~ 13 [Kos~ Djiba~ 2 [Ru
## 2 586028   586    28    2     439 2 [Burkin~ 2 [Bou~ 13 [Kos~ Djiba~ 2 [Ru
## 3 586043   586    43    2     439 2 [Burkin~ 2 [Bou~ 13 [Kos~ Djiba~ 2 [Ru
## 4 586044   586    44    2     439 2 [Burkin~ 2 [Bou~ 13 [Kos~ Djiba~ 2 [Ru
## 5 586052   586    52    2     439 2 [Burkin~ 2 [Bou~ 13 [Kos~ Djiba~ 2 [Ru
## 6 586082   586    82    2     439 2 [Burkin~ 2 [Bou~ 13 [Kos~ Djiba~ 2 [Ru
## # i 33 more variables: s00q05 <chr>, s00q07a <dbl+lbl>, s00q07b <dbl+lb
## #   s00q07c <dbl+lbl>, s00q07d <dbl+lbl>, s00q07d2 <dbl+lbl>, s00q22 <d
## #   s00q23a <chr>, s00q24a <chr>, s00q25a <chr>, s00q23b <chr>, s00q24b
```

```
## #   s00q25b <chr>, s00q08 <dbl+lbl>, s00q27 <dbl+lbl>, s00q28 <dbl+lbl>
## #   GPS__Latitude <dbl>, GPS__Longitude <dbl>, ADM3_FR <chr>, ADM3_PCOD
## #   ADM3_REF <chr>, ADM3ALT1_FR <lgl>, ADM3ALT2_FR <lgl>, ADM2_FR <chr>
## #   ADM2_PCODE <chr>, ADM1_FR <chr>, ADM1_PCODE <chr>, ADM0_FR <chr>, ..
```

3-h SAUVEGARDE DE LA BASE OBTENUE

La base obtenue a ensuite été sauvegardée.

```
haven::write_dta(ehcvm_final, "../Sorties/EHCVM_HDX_Burkina.dta")
```