Code de réplication pour 'SAGE - Sectoral Assessment of Green Activities'

Table of contents

1 Préparer "*Identification Variables*"

1.1 Packages

```
install.packages("readxl")
library(readxl)
library(dplyr)
```

1.2 Données

- Variables "country", "iso2c", "iso3c", "region", "subregion" viennent d'un github. Nous avons reformulé les variables de cette source avec nos noms directement sur excel. Nous téléchargeons et nommons ce fichier "first_iv".

```
# Importation du fichier XLSX avec read_excel()
first_iv <- read_excel("/Users/.../first_iv.xlsx")
```

- Variables “developed_economies”, “developing_economies_excl_china”, “developing_economies_excl_LDCs”, “LLDCs”, “LIDE”, “MIDE”, “HIDE”, “BRICS”, “EU”, “G20”, “G77” viennent de UNCTAD. Nous avons un fichier .xlsx intitulé “UNCTAD_classification”. Nous reprenons les classifications des économies selon UNCTAD. Aller sur ce site : https://unctadstat.unctad.org/EN/Classifications.html –> Télécharger fichier .xlsx intitulé “All groups compositions”.

```
# Importation du fichier XLSX avec read_excel()
second_iv <- read_excel("/Users/.../UNCTAD_classification.xlsx")
```

1.3 Fusions

On crée des colonnes dans “first_iv” qui indiqueront si un pays appartient ou non à une des catégories de “second_iv”.

```
# Renomme les colonnes de second_iv
second_iv <- second_iv %>%
  rename(
    developed_economies = `Developed Economies`,
    developing_economies_excl_china = `Developing Economies (excl_China)`,
     ↪
    developing_economies_excl_LDCs = `Developing Economies (excl_LDCs)`,
     ↪
    LLDCs = `LLDCs (Landlocked developing countries)`,
    LIDE = `Low-income developing economies`,
    MIDE = `Middle-income developing economies`,
    HIDE = `High-income developing economies`,
    BRICS = BRICS,
    EU = `European Union (2020 …)`,
    G20 = G20,
    G77 = G77
  )

# Ajout de 11 variables booléennes à first_iv
first_iv <- first_iv %>%
  mutate(
    developed_economies = ifelse(country %in% second_iv$developed_economies,
     ↪  1, 0),
```

```
    developing_economies_excl_china = ifelse(country %in%
     ↪  second_iv$developing_economies_excl_china, 1, 0),
    developing_economies_excl_LDCs = ifelse(country %in%
     ↪  second_iv$developing_economies_excl_LDCs, 1, 0),
    LLDCs = ifelse(country %in% second_iv$LLDCs, 1, 0),
    LIDE = ifelse(country %in% second_iv$LIDE, 1, 0),
    MIDE = ifelse(country %in% second_iv$MIDE, 1, 0),
    HIDE = ifelse(country %in% second_iv$HIDE, 1, 0),
    BRICS = ifelse(country %in% second_iv$BRICS, 1, 0),
    EU = ifelse(country %in% second_iv$EU, 1, 0),
    G20 = ifelse(country %in% second_iv$G20, 1, 0),
    G77 = ifelse(country %in% second_iv$G77, 1, 0)
  )
```

On vérifie si la fusion est bonne en listant tous les pays pour lesquels toutes les nouvelles variables de classification contiennent uniquement des “0” (aucune appartenance à une classification).

```
# Filtre les pays où toutes les nouvelles variables de classification sont
 ↪  égales à 0
pays_sans_classification <- first_iv %>%
  filter(
    developed_economies == 0 &
    developing_economies_excl_china == 0 &
    developing_economies_excl_LDCs == 0 &
    LLDCs == 0 &
    LIDE == 0 &
    MIDE == 0 &
    HIDE == 0 &
    BRICS == 0 &
    EU == 0 &
    G20 == 0 &
    G77 == 0
  ) %>%
  select(country)

unique(pays_sans_classification$country)
```

Cela nous donne le tableau ci-après avec les noms problématiques et leurs noms selon le dataframe:

Country name second_iv	Country name first_iv	Corrected country name
Åland Islands	Aland Islands	Aland Islands
No classif.	Antarctica	Antarctica
Bolivia (Plurinational State of)	Bolivia Plurinational State of	Bolivia Plurinational State of
Bonaire, Sint Eustatius and Saba	Bonaire Sint Eustatius and Saba	Bonaire Sint Eustatius and Saba
Dem. Rep. of the Congo	Congo Democratic Republic of the	Congo Democratic Republic of the
Côte d'Ivoire	C√¥te d'Ivoire	Cote d'Ivoire
Curaçao	Cura√ßao	Curacao
China, Hong Kong SAR	Hong Kong	Hong Kong
Iran (Islamic Republic of)	Iran Islamic Republic of	Iran Islamic Republic of
Dem. People's Rep. of Korea	Korea Democratic People's Republic of	Korea Democratic People's Republic of
Republic of Korea	Korea Republic of	Korea Republic of
Lao People's Dem. Rep.	Lao People's Democratic Republic	Lao People's Democratic Republic
China, Macao SAR	Macao	Macao
Micronesia (Federated States of)	Micronesia Federated States of	Micronesia Federated States of
Republic of Moldova	Moldova Republic of	Moldova Republic of
Netherlands (Kingdom of the)	Netherlands Kingdom of the	Netherlands Kingdom of the
State of Palestine	Palestine State of	Palestine State of
Réunion	R√©union	Reunion
Saint Barthélemy	Saint Barth√©lemy	Saint Barthelemy
Saint Helena	Saint Helena Ascension and Tristan da Cunha	Saint Helena Ascension and Tristan da Cunha
Svalbard and Jan Mayen Islands	Svalbard and Jan Mayen	Svalbard and Jan Mayen
China, Taiwan Province of	Taiwan Province of China	Taiwan Province of China
United Republic of Tanzania	Tanzania United Republic of	Tanzania United Republic of
Türkiye	T√ºrkiye	Turkey
United Kingdom	United Kingdom of Great Britain and Northern Ireland	United Kingdom of Great Britain and Northern Ireland

Country name second_iv	Country name first_iv	Corrected country name
United States	United States of America	United States of America
Venezuela (Bolivarian Rep. of)	Venezuela Bolivarian Republic of	Venezuela Bolivarian Republic of
British Virgin Islands	Virgin Islands (British)	Virgin Islands (British)
United States Virgin Islands	Virgin Islands (U.S.)	Virgin Islands (U.S.)

Nous gardons les noms de la colonne “Corrected country name” (nous gardons au mieux les noms de first_iv comme référence):

```
# Remplacer pour "second_iv"
## Vecteur des noms de pays à remplacer dans second_iv
pays_a_remplacer <- c(
  "Åland Islands", "Bolivia (Plurinational State of)", "Bonaire, Sint
   ↪ Eustatius and Saba",
  "Dem. Rep. of the Congo", "Côte d'Ivoire", "Curaçao", "China, Hong Kong
   ↪ SAR",
  "Iran (Islamic Republic of)", "Dem. People's Rep. of Korea", "Republic of
   ↪ Korea",
  "Lao People's Dem. Rep.", "China, Macao SAR", "Micronesia (Federated States
   ↪ of)",
  "Republic of Moldova", "Netherlands (Kingdom of the)", "State of
   ↪ Palestine",
  "Réunion", "Saint Barthélemy", "Saint Helena", "Svalbard and Jan Mayen
   ↪ Islands",
  "China, Taiwan Province of", "United Republic of Tanzania", "Türkiye",
   ↪ "United Kingdom",
  "United States", "Venezuela (Bolivarian Rep. of)", "British Virgin
   ↪ Islands",
  "United States Virgin Islands", "Wallis and Futuna Islands"
)

## Vecteur des noms corrigés
noms_corriges <- c(
  "Aland Islands", "Bolivia Plurinational State of", "Bonaire Sint Eustatius
   ↪ and Saba",
  "Congo Democratic Republic of the", "Cote d'Ivoire", "Curacao", "Hong
   ↪ Kong",
```

```
  "Iran Islamic Republic of", "Korea Democratic People's Republic of", "Korea
   ↪ Republic of",
  "Lao People's Democratic Republic", "Macao", "Micronesia Federated States
   ↪ of",
  "Moldova Republic of", "Netherlands Kingdom of the", "Palestine State of",
  "Reunion", "Saint Barthelemy", "Saint Helena Ascension and Tristan da
   ↪ Cunha",
  "Svalbard and Jan Mayen", "Taiwan Province of China", "Tanzania United
   ↪ Republic of",
  "Turkey", "United Kingdom of Great Britain and Northern Ireland", "United
   ↪ States of America",
  "Venezuela Bolivarian Republic of", "Virgin Islands (British)", "Virgin
   ↪ Islands (U.S.)",       "Wallis and Futuna"
)

## Fonction pour remplacer les noms dans toutes les colonnes de second_iv
remplacer_modalites <- function(df) {
  ### Applique sur toutes les colonnes
  df[] <- lapply(df, function(col) {
    if (is.character(col)) {
      #### Remplace les noms des pays s'ils apparaissent comme modalités
      sapply(col, function(p) {
        if (p %in% pays_a_remplacer) {
          noms_corriges[which(pays_a_remplacer == p)]
        } else {
          p
        }
      })
    } else {
      col
    }
  })
  return(df)
}

## Applique la fonction pour remplacer les noms dans second_iv
second_iv <- remplacer_modalites(second_iv)

# Vecteur des noms de pays à remplacer dans first_iv
pays_a_remplacer_first_iv <- c(
  "Aland Islands", "Bolivia Plurinational State of", "Bonaire Sint Eustatius
   ↪ and Saba",
```

```
  "C√¥te d'Ivoire", "Cura√ßao", "Congo Democratic Republic of the", "Hong
   ↪ Kong",
  "Iran Islamic Republic of", "Korea Democratic People's Republic of",
   ↪ "Korea Republic of",
  "Lao People's Democratic Republic", "Macao", "Micronesia Federated States
   ↪ of",
  "Moldova Republic of", "Netherlands Kingdom of the", "Palestine State of",
   ↪ "R√©union",
  "Saint Barth√©lemy", "Saint Helena Ascension and Tristan da Cunha",
   ↪ "Svalbard and Jan Mayen",
  "Taiwan Province of China", "Tanzania United Republic of", "T√ºrkiye",
  "United Kingdom of Great Britain and Northern Ireland", "United States of
   ↪ America",
  "Venezuela Bolivarian Republic of", "Virgin Islands (British)", "Virgin
   ↪ Islands (U.S.)"
)
# Vecteur des noms corrigés
noms_corriges_first_iv <- c(
  "Aland Islands", "Bolivia Plurinational State of", "Bonaire Sint Eustatius
   ↪ and Saba",
  "Cote d'Ivoire", "Curacao", "Congo Democratic Republic of the", "Hong
   ↪ Kong",
  "Iran Islamic Republic of", "Korea Democratic People's Republic of",
   ↪ "Korea Republic of",
  "Lao People's Democratic Republic", "Macao", "Micronesia Federated States
   ↪ of",
  "Moldova Republic of", "Netherlands Kingdom of the", "Palestine State of",
   ↪ "Reunion",
  "Saint Barthelemy", "Saint Helena Ascension and Tristan da Cunha",
   ↪ "Svalbard and Jan Mayen",
  "Taiwan Province of China", "Tanzania United Republic of", "Turkey",
  "United Kingdom of Great Britain and Northern Ireland", "United States of
   ↪ America",
  "Venezuela Bolivarian Republic of", "Virgin Islands (British)", "Virgin
   ↪ Islands (U.S.)"
)
# Fonction pour remplacer les noms dans first_iv$country
first_iv$country <- sapply(first_iv$country, function(p) {
  if (p %in% pays_a_remplacer_first_iv) {
    noms_corriges_first_iv[which(pays_a_remplacer_first_iv == p)]
  } else {
    p
```

```
  }
})
```

Les noms sont homogénisés, on refait l'ajout des 11 variables:

```
# Ajout de 11 variables booléennes à first_iv
first_iv <- first_iv %>%
  mutate(
    developed_economies = ifelse(country %in% second_iv$developed_economies,
     ↪  1, 0),
    developing_economies_excl_china = ifelse(country %in%
     ↪  second_iv$developing_economies_excl_china, 1, 0),
    developing_economies_excl_LDCs = ifelse(country %in%
     ↪  second_iv$developing_economies_excl_LDCs, 1, 0),
    LLDCs = ifelse(country %in% second_iv$LLDCs, 1, 0),
    LIDE = ifelse(country %in% second_iv$LIDE, 1, 0),
    MIDE = ifelse(country %in% second_iv$MIDE, 1, 0),
    HIDE = ifelse(country %in% second_iv$HIDE, 1, 0),
    BRICS = ifelse(country %in% second_iv$BRICS, 1, 0),
    EU = ifelse(country %in% second_iv$EU, 1, 0),
    G20 = ifelse(country %in% second_iv$G20, 1, 0),
    G77 = ifelse(country %in% second_iv$G77, 1, 0)
  )
```

On teste de nouveaux s'il y a des pays sans classification:

```
# Filtre les pays où toutes les nouvelles variables de classification sont
 ↪  égales à 0
pays_sans_classification <- first_iv %>%
  filter(
    developed_economies == 0 &
    developing_economies_excl_china == 0 &
    developing_economies_excl_LDCs == 0 &
    LLDCs == 0 &
    LIDE == 0 &
    MIDE == 0 &
    HIDE == 0 &
    BRICS == 0 &
    EU == 0 &
    G20 == 0 &
    G77 == 0
```

```
  ) %>%
  select(country)
```

Il reste maintenant une variable à ajouter qui est “income_group” que l’on tire des données de World Development Indicators (WDI) de la Banque Mondiale. Nous avons téléchargé un fichier .xlsx intitulé “incomegroup_WDI.xlsx”

```
# Fichier Excel
third_iv <- read_excel("/Users/.../incomegroup_WDI.xlsx")

# Fusion de first_iv et third_iv en gardant toutes les observations de
 ↪  third_iv
first_iv <- first_iv %>%
  right_join(third_iv, by = "iso3c")
```

Le fichier “incomegroup_WDI.xlsx” ajoute 49 codes ISO, qui ne sont pas informés dans “first_-iv” :

```
unique(first_iv$iso3c)
```

Voici la liste de codes ISO supplémentaires: “AFE” “AFW” “ARB” “CEB” “CHI” “CSS” “EAP” “EAR” “EAS” “ECA” “ECS” “EMU” “EUU” “FCS” “HIC” “HPC” “IBD” “IBT” “IDA” “IDB” “IDX” “LAC” “LCN” “LDC” “LIC” “LMC” “LMY” “LTE” “MEA” “MIC” “MNA” “NAC” “OED” “OSS” “PRE” “PSS” “PST” “SAS” “SSA” “SSF” “SST” “TEA” “TEC” “TLA” “TMN” “TSA” “TSS” “UMC” “WLD” “XKX”.

Le seul que nous gardons est “XKX” qui désigne le Kosovo. Nous appliquons cet ajout à first_iv:

```
# On renomme la colonne "sub-region" en "sub_region"
first_iv <- first_iv %>%
  rename(subregion = `sub-region`)

# Modification des informations pour XKX
  first_iv <- first_iv %>%
    mutate(
      iso2c = ifelse(iso3c == "XKX", "XK", iso2c),
      income_group = ifelse(iso3c == "XKX", "Upper middle income",
       ↪  income_group),
      developed_economies = ifelse(iso3c == "XKX", 1, developed_economies),
```

```
      developing_economies_excl_china = ifelse(iso3c == "XKX", 0,
       ↪  developing_economies_excl_china),
      developing_economies_excl_LDCs = ifelse(iso3c == "XKX", 0,
       ↪  developing_economies_excl_LDCs),
      LLDCs = ifelse(iso3c == "XKX", 0, LLDCs),
      LIDE = ifelse(iso3c == "XKX", 0, LIDE),
      MIDE = ifelse(iso3c == "XKX", 0, MIDE),
      HIDE = ifelse(iso3c == "XKX", 0, HIDE),
      BRICS = ifelse(iso3c == "XKX", 0, BRICS),
      EU = ifelse(iso3c == "XKX", 0, EU),
      G20 = ifelse(iso3c == "XKX", 0, G20),
      G77 = ifelse(iso3c == "XKX", 0, G77),
      country = ifelse(iso3c == "XKX", "Kosovo", country),
      region = ifelse(iso3c == "XKX", "Europe", region),
      subregion = ifelse(iso3c == "XKX", "Eastern Europe", subregion)
  )
```

On supprime les codes iso3c en trop:

```
# Vecteur des codes iso3c à supprimer
codesiso3c_a_supprimer <- c("AFE", "AFW", "ARB", "CEB", "CHI", "CSS", "EAP",
 ↪  "EAR", "EAS", "ECA", "ECS", "EMU", "EUU", "FCS", "HIC", "HPC", "IBD",
 ↪  "IBT", "IDA", "IDB", "IDX", "LAC", "LCN", "LDC", "LIC", "LMC", "LMY",
 ↪  "LTE", "MEA", "MIC", "MNA", "NAC", "OED", "OSS", "PRE", "PSS", "PST",
 ↪  "SAS", "SSA","SSF", "SST", "TEA", "TEC", "TLA", "TMN", "TSA", "TSS",
 ↪  "UMC", "WLD")

# Supprime les lignes avec ces codes iso3c
first_iv <- first_iv %>%
  filter(!iso3c %in% codesiso3c_a_supprimer)
```

1.4 Enregistrement IV framework

```
# Téléchargement au format RDS
saveRDS(first_iv, file = "/Users/.../IV.rds")
```

2 Préparation des données relatives à l'activité verte

Les données d'activités économiques classées ISIC 2-digit sont disponibles sur ILOSTAT data : https://rshiny.ilo.org/dataexplorer45/

Pour faciliter le téléchargement nous opérons directement depuis R grâce au package "Rilostat" qui nous permet de télécharger et traiter les données plus facilement et rapidement.

2.1 Packages

```
install.packages("Rilostat")
library("Rilostat")
library(dplyr)
```

2.2 Liste des indicateurs d'intérêt

Nous consultons la liste des bases de données disponibles et sélectionnons celles qui nous intéressent, c'est-à-dire les données

```
list_indic <- get_ilostat_toc()
ISIC_list_indic <- get_ilostat_toc(search = 'ISIC')
```

2.3 Jeux de données sélectionnées

2.3.1 Avec "Sex" et "Economic activity"

	Variable	Description
1	EMP_TEMP_-SEX_EDU_-EC2_NB_A	Employment by sex, **education** and economic activity - ISIC level 2 (thousands)
2	EMP_TEMP_-SEX_STE_-EC2_NB_A	Employment by sex, **status in employment** and economic activity - ISIC level 2 (thousands)
3	EMP_TEMP_-SEX_IFL_-EC2_NB_A	Employment by sex, **informal/formal economy** and economic activity - ISIC level 2 (thousands)

	Variable	Description
4	EMP_TEMP_-SEX_EST_-EC2_NB_A	Employment by sex, **establishment size** and economic activity - ISIC level 2 (thousands)
5	EMP_NIFL_-SEX_EC2_-NB_A	**Informal employment** by sex and economic activity - ISIC level 2 (thousands)
6	EMP_NIFL_-SEX_EC2_-RT_A	**Informal employment rate** by sex and economic activity - ISIC level 2 (%)
7	EMP_PIFL_-SEX_EC2_-NB_A	**Employment outside the formal sector** by sex and economic activity - ISIC level 2 (thousands)
8	EMP_PIFL_-SEX_EC2_-RT_A	**Share of employment outside the formal sector** by sex and economic activity - ISIC level 2 (%)
9	EES_TEES_-SEX_EC2_-NB_A	**Employees** by sex and economic activity - ISIC level 2 (thousands)
10	HOW_TEMP_-SEX_EC2_-NB_A	**Mean weekly hours actually worked per employed person** by sex and economic activity - ISIC level 2
11	HOW_XEES_-SEX_EC2_-NB_A	**Mean weekly hours actually worked per employee** by sex and economic activity - ISIC level 2
12	EMP_CARE_-SEX_EC2_-NB_A	**Care employment** by sex and economic activity - ISIC level 2 (thousands)
13	EMP_STEM_-SEX_EC2_-NB_A	**Employment in STEM occupations** by sex and economic activity - ISIC level 2 (thousands)
14	EMP_TOUR_-SEX_EC2_-NB_A	**Tourism sector employment** by sex and economic activity - ISIC level 2 (thousands)
15	EMP_PUBL_-SEX_EC2_-NB_A	**Public sector employment** by sex and economic activity - ISIC level 2 (thousands)

2.3.2 Avec “Age” et “Economic activity”

	Variable Name	Description
1	EMP_-TEMP_-AGE_EC2_-NB_A	**Employment** by age and economic activity - ISIC level 2 (thousands)
2	EMP_NIFL_-AGE_EC2_-NB_A	**Informal employment** by age and economic activity - ISIC level 2 (thousands)
3	EMP_NIFL_-AGE_EC2_-RT_A	**Informal employment rate** by age and economic activity - ISIC level 2 (%)
4	EMP_PIFL_-AGE_EC2_-NB_A	**Employment outside the formal sector** by age and economic activity - ISIC level 2 (thousands)
5	EMP_PIFL_-AGE_EC2_-RT_A	**Share of employment outside the formal sector** by age and economic activity - ISIC level 2 (%)
6	EES_TEES_-AGE_EC2_-NB_A	**Employees** by age and economic activity - ISIC level 2 (thousands)
7	HOW_-TEMP_-AGE_EC2_-NB_A	**Mean weekly hours actually worked per employed person** by age and economic activity - ISIC level 2
8	HOW_-XEES_AGE_-EC2_NB_A	**Mean weekly hours actually worked per employee** by age and economic activity - ISIC level 2

2.4 Téléchargement

```
# Définition des identifiants des ensembles de données par âge
dataset_ids_age <- c(
  "EMP_TEMP_AGE_EC2_NB_A",
  "EMP_NIFL_AGE_EC2_NB_A",
  "EMP_NIFL_AGE_EC2_RT_A",
  "EMP_PIFL_AGE_EC2_NB_A",
  "EMP_PIFL_AGE_EC2_RT_A",
  "EES_TEES_AGE_EC2_NB_A",
  "HOW_TEMP_AGE_EC2_NB_A",
  "HOW_XEES_AGE_EC2_NB_A"
```

```
)

# Boucle pour télécharger et créer chaque cadre de données pour les
 ↪  indicateurs d'âge
for (id in dataset_ids_age) {
  df <- get_ilostat(id = id, segment = "indicateur")
  assign(id, df)
  cat("Chargement terminé pour", id, "\n")
}

# Définition des identifiants des ensembles de données par sexe
dataset_ids_sex <- c(
  "EMP_TEMP_SEX_EDU_EC2_NB_A",
  "EMP_TEMP_SEX_STE_EC2_NB_A",
  "EMP_TEMP_SEX_IFL_EC2_NB_A",
  "EMP_TEMP_SEX_EST_EC2_NB_A",
  "EMP_NIFL_SEX_EC2_NB_A",
  "EMP_NIFL_SEX_EC2_RT_A",
  "EMP_PIFL_SEX_EC2_NB_A",
  "EMP_PIFL_SEX_EC2_RT_A",
  "EES_TEES_SEX_EC2_NB_A",
  "HOW_TEMP_SEX_EC2_NB_A",
  "HOW_XEES_SEX_EC2_NB_A",
  "EMP_CARE_SEX_EC2_NB_A",
  "EMP_STEM_SEX_EC2_NB_A",
  "EMP_TOUR_SEX_EC2_NB_A",
  "EMP_PUBL_SEX_EC2_NB_A"
)

# Boucle pour télécharger et créer chaque cadre de données pour les
 ↪  indicateurs de genre
for (id in dataset_ids_sex) {
  df <- get_ilostat(id = id, segment = "indicateur")
  assign(id, df)
  cat("Chargement terminé pour", id, "\n")
}
```

2.5 Renommages

```
# Règles de changement de nom pour les indicateurs de "sexe
```

```
renaming_rules_sex <- list(
  EMP_TEMP_SEX_EDU_EC2_NB_A = list(time = "year",
                                   ref_area = "iso3c",
                                   obs_value = "value",
                                   classif1 = "edu",
                                   classif2 = "isic_code_2dg",
                                   indicator = "EMP_TEMP_SEX_EDU_EC2_NB_A"),
  EMP_TEMP_SEX_STE_EC2_NB_A = list(time = "year",
                                   ref_area = "iso3c",
                                   obs_value = "value",
                                   classif1 = "ste",
                                   classif2 = "isic_code_2dg",
                                   indicator = "EMP_TEMP_SEX_STE_EC2_NB_A"),
  EMP_TEMP_SEX_IFL_EC2_NB_A = list(time = "year",
                                   ref_area = "iso3c",
                                   obs_value = "value",
                                   classif1 = "ifl",
                                   classif2 = "isic_code_2dg",
                                   indicator = "EMP_TEMP_SEX_IFL_EC2_NB_A"),
  EMP_TEMP_SEX_EST_EC2_NB_A = list(time = "year",
                                   ref_area = "iso3c",
                                   obs_value = "value",
                                   classif1 = "est",
                                   classif2 = "isic_code_2dg",
                                   indicator = "EMP_TEMP_SEX_EST_EC2_NB_A"),
  EMP_NIFL_SEX_EC2_NB_A     = list(time = "year",
                                   ref_area = "iso3c",
                                   obs_value = "value",
                                   classif1 = "isic_code_2dg",
                                   indicator = "EMP_NIFL_SEX_EC2_NB_A"),
  EMP_NIFL_SEX_EC2_RT_A     = list(time = "year",
                                   ref_area = "iso3c",
                                   obs_value = "value",
                                   classif1 = "isic_code_2dg",
                                   indicator = "EMP_NIFL_SEX_EC2_RT_A"),
  EMP_PIFL_SEX_EC2_NB_A     = list(time = "year",
                                   ref_area = "iso3c",
                                   obs_value = "value",
                                   classif1 = "isic_code_2dg",
                                   indicator = "EMP_PIFL_SEX_EC2_NB_A"),
  EMP_PIFL_SEX_EC2_RT_A     = list(time = "year",
                                   ref_area = "iso3c",
                                   obs_value = "value",
```

```
                                  classif1 = "isic_code_2dg",
                                  indicator = "EMP_PIFL_SEX_EC2_RT_A"),
  EES_TEES_SEX_EC2_NB_A     = list(time = "year",
                                  ref_area = "iso3c",
                                  obs_value = "value",
                                  classif1 = "isic_code_2dg",
                                  indicator = "EES_TEES_SEX_EC2_NB_A"),
  HOW_TEMP_SEX_EC2_NB_A     = list(time = "year",
                                  ref_area = "iso3c",
                                  obs_value = "value",
                                  classif1 = "isic_code_2dg",
                                  indicator = "HOW_TEMP_SEX_EC2_NB_A"),
  HOW_XEES_SEX_EC2_NB_A     = list(time = "year",
                                  ref_area = "iso3c",
                                  obs_value = "value",
                                  classif1 = "isic_code_2dg",
                                  indicator = "HOW_XEES_SEX_EC2_NB_A"),
  EMP_CARE_SEX_EC2_NB_A     = list(time = "year",
                                  ref_area = "iso3c",
                                  obs_value = "value",
                                  classif1 = "isic_code_2dg",
                                  indicator = "EMP_CARE_SEX_EC2_NB_A"),
  EMP_STEM_SEX_EC2_NB_A     = list(time = "year",
                                  ref_area = "iso3c",
                                  obs_value = "value",
                                  classif1 = "isic_code_2dg",
                                  indicator = "EMP_STEM_SEX_EC2_NB_A"),
  EMP_TOUR_SEX_EC2_NB_A     = list(time = "year",
                                  ref_area = "iso3c",
                                  obs_value = "value",
                                  classif1 = "isic_code_2dg",
                                  indicator = "EMP_TOUR_SEX_EC2_NB_A"),
  EMP_PUBL_SEX_EC2_NB_A     = list(time = "year",
                                  ref_area = "iso3c",
                                  obs_value = "value",
                                  classif1 = "isic_code_2dg",
                                  indicator = "EMP_PUBL_SEX_EC2_NB_A")
)

# Règles de changement de nom pour les indicateurs d'âge
renaming_rules_age <- list(
  EMP_TEMP_AGE_EC2_NB_A = list(time = "year",
                               ref_area = "iso3c",
```

```
                            obs_value = "value",
                            classif1 = "age",
                            classif2 = "isic_code_2dg",
                            indicator = "EMP_TEMP_AGE_EC2_NB_A"),
EMP_NIFL_AGE_EC2_NB_A = list(time = "year",
                            ref_area = "iso3c",
                            obs_value = "value",
                            classif1 = "age",
                            classif2 = "isic_code_2dg",
                            indicator = "EMP_NIFL_AGE_EC2_NB_A"),
EMP_NIFL_AGE_EC2_RT_A = list(time = "year",
                            ref_area = "iso3c",
                            obs_value = "value",
                            classif1 = "age",
                            classif2 = "isic_code_2dg",
                            indicator = "EMP_NIFL_AGE_EC2_RT_A"),
EMP_PIFL_AGE_EC2_NB_A = list(time = "year",
                            ref_area = "iso3c",
                            obs_value = "value",
                            classif1 = "age",
                            classif2 = "isic_code_2dg",
                            indicator = "EMP_PIFL_AGE_EC2_NB_A"),
EMP_PIFL_AGE_EC2_RT_A = list(time = "year",
                            ref_area = "iso3c",
                            obs_value = "value",
                            classif1 = "age",
                            classif2 = "isic_code_2dg",
                            indicator = "EMP_PIFL_AGE_EC2_RT_A"),
EES_TEES_AGE_EC2_NB_A = list(time = "year",
                            ref_area = "iso3c",
                            obs_value = "value",
                            classif1 = "age",
                            classif2 = "isic_code_2dg",
                            indicator = "EES_TEES_AGE_EC2_NB_A"),
HOW_TEMP_AGE_EC2_NB_A = list(time = "year",
                            ref_area = "iso3c",
                            obs_value = "value",
                            classif1 = "age",
                            classif2 = "isic_code_2dg",
                            indicator = "HOW_TEMP_AGE_EC2_NB_A"),
HOW_XEES_AGE_EC2_NB_A = list(time = "year",
                            ref_area = "iso3c",
                            obs_value = "value",
```

```
                                classif1 = "age",
                                classif2 = "isic_code_2dg",
                                indicator = "HOW_XEES_AGE_EC2_NB_A")
)

rename_variables <- function(df, rules) {
  renamed_df <- df
  # Parcourt chaque nom de colonne dans les règles
  for (old_col in names(rules)) {
    new_col <- rules[[old_col]]
    # Renomme la colonne old_col en new_col
    renamed_df <- rename(renamed_df, !!new_col := !!sym(old_col))
  }
  return(renamed_df)
}

# Renomme les ensembles de données par sexe
for (id in dataset_ids_sex) {
  if (!is.null(renaming_rules_sex[[id]])) {
    df <- get(id)
    df_renamed <- rename_variables(df, renaming_rules_sex[[id]])
    assign(id, df_renamed)
    cat("Changement de nom demandé pour", id, "\n")
  }
}

# Renommer les ensembles de données par âge
for (id in dataset_ids_age) {
  if (!is.null(renaming_rules_age[[id]])) {
    df <- get(id)
    df_renamed <- rename_variables(df, renaming_rules_age[[id]])
    assign(id, df_renamed)
    cat("Changement de nom demandé pour", id, "\n")
  }
}
```

```
library(dplyr)

# Liste des colonnes à supprimer
columns_to_remove <- c("obs_status", "note_classif", "note_indicator",
 ↪  "note_source", "source")
```

```r
# Pour les ensembles de données basés sur l'âge
for (id in dataset_ids_age) {
  df <- get(id)
  df <- df %>% select(-any_of(columns_to_remove))
  assign(id, df)
  cat("Colonnes supprimées pour", id, "\n")
}

# Pour les ensembles de données basés sur le sexe
for (id in dataset_ids_sex) {
  df <- get(id)
  df <- df %>% select(-any_of(columns_to_remove))
  assign(id, df)
  cat("Colonnes supprimées pour", id, "\n")
}
```

```r
# Expression à supprimer
pattern_to_remove <- "EC2_ISIC4_"

# Liste combinée des identifiants de vos ensembles de données
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

# Boucle sur chaque image de données
for (id in all_datasets) {
  df <- get(id)
  # Vérifie si la colonne « isic_code_2dg » existe dans la base de données
  if ("isic_code_2dg" %in% names(df)) {
    df <- df %>%
      mutate(isic_code_2dg = gsub(pattern_to_remove, "", isic_code_2dg))
    assign(id, df)
    cat("La variable isic_code_2dg a été nettoyée pour", id, "\n")
  }
}
```

La variable « âge » est composée des modalités suivantes :

- AGE_YTHADULT_Y25-24 : Age (Youth, adults): 15-24
- AGE_YTHADULT_YGE25 : Age (Youth, adults): 25+
- AGE_YTHADULT_YGE15 : Age (Youth, adults): 15+
- NA

Toutes les modalités sont des niveaux agrégés, nous simplifierons donc leurs règles de dénomination en ne conservant que les éléments situés après « AGE_YTHADULT_ ».

```
# Expression à supprimer
pattern_to_remove_age <- "AGE_YTHADULT_"

# Combinez tous vos identifiants d'ensembles de données pour parcourir
 ↪  l'ensemble des données
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

# Boucle sur chaque image de données
for (id in all_datasets) {
  df <- get(id)
  # Si la colonne « âge » est présente, appliquer la transformation
  if ("age" %in% names(df)) {
    df <- df %>%
      mutate(age = gsub(pattern_to_remove_age, "", age))
    assign(id, df)
    cat("La variable âge a été nettoyée pour devenir", id, "\n")
  }
}
```

La variable « sexe » est composée des modalités suivantes :

- SEX_T : “Sex: Total”
- SEX_M : “Sex: Male”
- SEX_F : “Sex: Female”
- SEX_O : “Sex: Other”
- NA

Toutes les modalités sont dans des niveaux agrégés, nous simplifierons donc leurs règles de dénomination en ne conservant que les éléments après “SEX_”.

```
# Expression à supprimer
pattern_to_remove_sex <- "SEX_"

# Liste combinée des identifiants de vos ensembles de données
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

# Boucle sur chaque image de données
for (id in all_datasets) {
```

```
  df <- get(id)
  # Si la colonne « sexe » est présente, appliquer la transformation
  if ("sex" %in% names(df)) {
    df <- df %>%
      mutate(sex = gsub(pattern_to_remove_sex, "", sex))
    assign(id, df)
    cat("La variable « sexe » a été nettoyée pour devenir", id, "\n")
  }
}
```

Nous supprimons maintenant toutes les lignes contenant « NA » pour la variable « value » :

```
# Combiner tous les identifiants de datasets dans une seule liste
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

# Boucle sur chaque dataframe pour filtrer les lignes
for (id in all_datasets) {
  df <- get(id)
  # Conserver uniquement les lignes où 'value' n'est pas NA
  df <- df %>% filter(!is.na(value))
  assign(id, df)
  cat("Lignes avec une valeur manquante pour 'value' supprimées pour", id,
   ↪  "\n")
}
```

Nous réajustons les cadres de données EMP_TEMP_SEX_IFL_EC2_NB_A, EMP_-TEMP_SEX_STE_EC2_NB_A, EMP_TEMP_SEX_EST_EC2_NB_A, EMP_TEMP_-SEX_EDU_EC2_NB_A:

```
EMP_TEMP_SEX_IFL_EC2_NB_A <- EMP_TEMP_SEX_IFL_EC2_NB_A %>%
  mutate(ifl = gsub("IFL_NATURE_", "", ifl))

EMP_TEMP_SEX_STE_EC2_NB_A <- EMP_TEMP_SEX_STE_EC2_NB_A %>%
  mutate(ste = gsub("STE_AGGREGATE_", "", ste))

EMP_TEMP_SEX_EST_EC2_NB_A <- EMP_TEMP_SEX_EST_EC2_NB_A %>%
  mutate(est = gsub("EST_SIZEAGGREGATE_", "", est))

EMP_TEMP_SEX_EDU_EC2_NB_A <- EMP_TEMP_SEX_EDU_EC2_NB_A %>%
  mutate(edu = gsub("EDU_AGGREGATE_", "", edu))
```

Pour tous les cadres de données, si une cellule de la variable « isic_code_2dg » commence par « EC2_ISIC3_ », nous supprimons cette ligne :

```
# Expression régulière pour détecter les valeurs commençant par «
 ↪  EC2_ISIC3_ »
pattern <- "^EC2_ISIC3_"

# Liste combinée de tous les identifiants de nos ensembles de données
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

# Parcourir chaque cadre de données
for (id in all_datasets) {
  df <- get(id)
  # Si la colonne « isic_code_2dg » est présente, supprimer les lignes
   ↪  concernées
  if ("isic_code_2dg" %in% names(df)) {
    df <- df %>% filter(!grepl(pattern, isic_code_2dg))
    assign(id, df)
    cat("Lignes annulées pour", id, "\n")
  }
}
```

2.6 Fusionner avec le potentiel de verdissement (TVE)

Nous déterminons quelles activités économiques ont un potentiel d'écologisation : nous créons la variable « EU_ISIC4_2dg » et classons « 2 » si l'activité économique a un potentiel d'écologisation, « 1 » si elle n'est pas reconnue, et « 3 » si nous sommes confrontés à une activité économique indéterminée classée comme « TOTAL » ou "X" dans « isic_code_2dg ».

```
# Définir la liste de ISIC codes à fort potentiel de verdissement
codes_isic_verts <- c("A16", "A02", "C17", "C20", "C21", "C22", "C23", "C24",
 ↪
                      "C25", "C26", "C27", "C28", "C29", "C33", "D35", "E36",
                       ↪
                      "E37", "E38", "E39", "F41", "F42", "F43", "G46", "G47",
                       ↪
                      "H49", "H50", "H51", "H52", "H53", "I55", "J59", "J60",
                       ↪
                      "J61", "J62", "L68", "M71", "M72", "N77", "O84", "P85",
                       ↪
```

```
                    "Q86", "Q87", "Q88", "R90", "R91", "S95")

# Combine les identifiants de nos cadres de données (par ex. ceux des
 ↪  indicateurs par âge et par sexe)
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

# Pour chaque base de données, on crée la variable EU_ISIC4_2dg avec les
 ↪  conditions requises
for (df_name in all_datasets) {
  df <- get(df_name)
  if ("isic_code_2dg" %in% names(df)) {
    df <- df %>%
      mutate(EU_ISIC4_2dg = case_when(
        isic_code_2dg %in% c("TOTAL", "X") ~ "3",
        isic_code_2dg %in% codes_isic_verts ~ "2",
        TRUE ~ "1"
      ))
    assign(df_name, df)
    cat("Variable EU_ISIC4_2dg créée pour", df_name, "\n")
  }
}
```

2.7 Cartographie avec la classification des pays « IV.rds »

Nous ajoutons les variables suivantes de catégorisation des codes iso à 3 chiffres :

- "country": donne le nom des pays associés aux codes iso ;
- "region": indique le continent du pays ;
- "subregion": renvoie à des informations plus détaillées sur la région dans laquelle le pays est situé ;
- "income_group": renvoie à une classification UNCTAD ;
- "SEI": se réfère à la classification Alberti & Goujon (2019) des petites îles économiques ;
- "SIDS": se réfère à la classification des Nations Unies.

```
# Chargement du fichier "IV.rds
IV <- readRDS("/Users/.../IV.rds")

# Supposons que le cadre de données IV comporte une colonne « iso3c »
IV$iso3c <- toupper(trimws(IV$iso3c))
```

```r
# Combiner les noms des données dans un vecteur
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

# Pour chaque jeu de données, nous vérifions les codes iso3c (ici dans la
 ↪  variable « pays ») qui ne sont pas dans IV
for (df_name in all_datasets) {
  df <- get(df_name)

  # Normalise les codes des dataframes en utilisant toupper() et trimws()
  codes_df <- toupper(trimws(unique(df$iso3c)))

  differences <- setdiff(codes_df, IV$iso3c)

  cat("Pour le jeu de données", df_name, "les codes iso3c suivants ne
   ↪  correspondent pas à IV:\n")
  print(differences)
  cat("\n------------------------\n")
}

# Nous affirmons que les cadres de données sont référencés par les vecteurs
 ↪  dataset_ids_age et dataset_ids_sex
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

for (df_name in all_datasets) {
  df <- get(df_name)
  if ("iso3c" %in% names(df)) {
    df <- df %>%
      mutate(iso3c = if_else(iso3c == "KOS", "XKX", iso3c))
    assign(df_name, df)
    cat("Dans", df_name, "la modalité 'KOS' a été renommée 'XKX'.\n")
  }
}

# Crée un cadre de données avec les nouvelles lignes
new_entries <- data.frame(
  iso3c = c("COK", "NIU", "WLF"),
  country = c("cook_islands", "Niue", "Wallis_and_Futuna"),
  region = c("Oceania", "Oceania", "Oceania"),
  subregion = c("Polynesia", "Polynesia", "Polynesia"),
  income_group = c("High income", "Lower middle income", "High income"),
  SEI = c(1, 1, 1),
  SIDS = c(1, 1, 0),
```

```
  stringsAsFactors = FALSE
)

# Si IV et new_entries n'ont en commun que certaines colonnes
common_cols <- intersect(names(IV), names(new_entries))

# Sélection de ces colonnes dans chaque cadre de données
IV_subset <- IV[, common_cols, drop = FALSE]
new_entries_subset <- new_entries[, common_cols, drop = FALSE]

# Combine les deux cadres de données
IV_updated <- rbind(IV_subset, new_entries_subset)

# Résultats
print(head(IV_updated))

# Nous veillons à normaliser les codes dans IV_updated et dans nos dataframes
↪
IV_updated <- IV_updated %>%
  mutate(iso3c = toupper(trimws(iso3c)))

all_datasets <- c(dataset_ids_age, dataset_ids_sex)

for (df_name in all_datasets) {
  df <- get(df_name)
  if ("iso3c" %in% names(df)) {
    # Normalisation de la variable iso3c
    df <- df %>%
      mutate(iso3c = toupper(trimws(iso3c))) %>%
      left_join(IV_updated, by = c("iso3c" = "iso3c"))
    assign(df_name, df)
    cat("Fusion réalisée pour", df_name, "\n")
  }
}

# Combiner les noms des jeux de données
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

for (df_name in all_datasets) {
  df <- get(df_name)

  # Calcule le nombre de doublons (lignes dupliquées sur toutes les colonnes)
  n_doublons <- sum(duplicated(df))
```

```
  cat("Pour le jeu de données", df_name, ": Total doublons =", n_doublons,
   ↪  "\n")

  # Afficher les doublons s'il y en a
  if(n_doublons > 0) {
    cat("Doublons trouvés dans", df_name, ":\n")
    print(df[duplicated(df), ])
  }
  cat("-----------------------------------------------\n")
}
```

2.8 Transformation des variables en variables factorielles

2.8.1 “iso3c”

```
# Nous confirmons que nos cadres de données sont contenus dans ces deux
 ↪  vecteurs
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

# Initialise un vecteur vide pour stocker tous les codes iso3c uniques
all_iso3c <- character(0)

# Nous parcourons chaque base de données et extrayons la variable « iso3c »
for (df_name in all_datasets) {
  df <- get(df_name)
  if ("iso3c" %in% names(df)) {
    # Normaliser : mettre des majuscules et éliminer les espaces superflus
    codes <- unique(toupper(trimws(df$iso3c)))
    # Ajouter ces codes au vecteur global, en évitant les doublons
    all_iso3c <- union(all_iso3c, codes)
  }
}

# Pour plus de lisibilité, nous trions les codes
all_iso3c <- sort(all_iso3c)

# Afficher la liste des codes uniques
cat("Liste des codes iso3c uniques dans les cadres de données (à l'exclusion
 ↪  de IV) :\n")
```

```
print(all_iso3c)
```

Liste des codes iso3c codes dans tous les jeux de données : “AFG” “AGO” “ALB” “ARE” “ARG” “ARM” “AUS” “AUT” “BDI” “BEL” “BEN” “BFA” “BGD” “BGR” “BHS” “BIH” “BLR” “BOL” “BRA” “BRB” “BRN” “BTN” “BWA” “CHE” “CHL” [26] “CIV” “COD” “COG” “COK” “COL” “COM” “CPV” “CRI” “CYP” “CZE” “DEU” “DNK” “DOM” “ECU” “EGY” “ESP” “EST” “ETH” “FIN” “FJI” “FRA” “FSM” “GBR” “GEO” “GHA” “GMB” “GNB” “GRC” “GRD” “GTM” “GUY” “HND” “HRV” “HUN” “IDN” “IND” “IRL” “IRN” “IRQ” “ISL” “ISR” “ITA” “JOR” “JPN” “KEN” “KGZ” “KHM” “KIR” “KOR” “LAO” “LBN” “LBR” “LKA” “LSO” “LTU” “LUX” “LVA” “MDG” “MDV” “MEX” “MHL” “MKD” “MLI” “MLT” “MMR” “MNG” “MOZ” “MRT” “MUS” “NAM” “NCL” “NER” “NGA” “NIU” “NLD” “NOR” “NPL” “NRU” “PAK” “PAN” “PER” “PHL” “PLW” “PNG” “POL” “PRT” “PSE” “ROU” “RWA” “SDN” “SEN” “SGP” “SLB” “SLE” “SLV” “SOM” “SRB” “SUR” “SVK” “SVN” “SWE” “SWZ” “SYC” “TCD” “TGO” “THA” “TJK” “TLS” “TON” “TUN” “TUR” “TUV” “TZA” “UGA” “UKR” “URY” “USA” “VNM” “VUT” “WLF” “WSM” “XKX” “ZMB” “ZWE”.

```
# Créer une table de correspondance globale
global_iso3c <- c("AFG", "AGO", "ALB", "ARE", "ARG", "ARM", "AUS", "AUT",
                  "BDI", "BEL", "BEN", "BFA", "BGD", "BGR", "BHS", "BIH",
                  "BLR", "BOL", "BRA", "BRB", "BRN", "BTN", "BWA", "CHE",
                  "CHL", "CIV", "COD", "COG", "COK", "COL", "COM", "CPV",
                  "CRI", "CYP", "CZE", "DEU", "DNK", "DOM", "ECU", "EGY",
                  "ESP", "EST", "ETH", "FIN", "FJI", "FRA", "FSM", "GBR",
                  "GEO", "GHA", "GMB", "GNB", "GRC", "GRD", "GTM", "GUY",
                  "HND", "HRV", "HUN", "IDN", "IND", "IRL", "IRN", "IRQ",
                  "ISL", "ISR", "ITA", "JOR", "JPN", "KEN", "KGZ", "KHM",
                  "KIR", "KOR", "LAO", "LBN", "LBR", "LKA", "LSO", "LTU",
                  "LUX", "LVA", "MDG", "MDV", "MEX", "MHL", "MKD", "MLI",
                  "MLT", "MMR", "MNG", "MOZ", "MRT", "MUS", "NAM", "NCL",
                  "NER", "NGA", "NIU", "NLD", "NOR", "NPL", "NRU", "PAK",
                  "PAN", "PER", "PHL", "PLW", "PNG", "POL", "PRT", "PSE",
                  "ROU", "RWA", "SDN", "SEN", "SGP", "SLB", "SLE", "SLV",
                  "SOM", "SRB", "SUR", "SVK", "SVN", "SWE", "SWZ", "SYC",
                  "TCD", "TGO", "THA", "TJK", "TLS", "TON", "TUN", "TUR",
                  "TUV", "TZA", "UGA", "UKR", "URY", "USA", "VNM", "VUT",
                  "WLF", "WSM", "XKX", "ZMB", "ZWE")

# Créer un tableau de correspondance avec un code numérique pour chaque iso3c
global_mapping <- data.frame(
  iso3c = global_iso3c,
  global_code = seq_along(global_iso3c),
```

```
  stringsAsFactors = FALSE
)

# Pour chaque jeu de données, on transforme la variable "iso3c" en un facteur
 ↪  avec ces niveaux et on crée la variable "iso3c_code", qui utilise le code
 ↪  numérique prédéterminé.
# Les cadres de données se trouvent dans les vecteurs dataset_ids_age et
 ↪  dataset_ids_sex
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

for (df_name in all_datasets) {
  df <- get(df_name)
  if ("country" %in% names(df)) {
    df <- df %>%
      # Normalise la variable pays avec des lettres majuscules et sans
       ↪  espaces superflus
      mutate(country = toupper(trimws(iso3c))) %>%
      # Transforme la variable en facteur avec des niveaux fixes définis dans
       ↪  global_mapping
      mutate(country_factor = factor(iso3c, levels = global_mapping$iso3c),
             # Créer une nouvelle variable numérique basée sur le facteur
             iso3c_code = as.integer(country_factor))

    assign(df_name, df)
    cat("Dans", df_name, "la variable 'pays' a été transformée et codée avec
     ↪  'iso3c_code'.\n")
  }
}

# Combine les noms des jeux de données
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

for (df_name in all_datasets) {
  df <- get(df_name)
  if ("country_factor" %in% names(df)) {
    df <- df %>% select(-country_factor)
    assign(df_name, df)
    cat("Country_factor' variable supprimée pour", df_name, "\n")
  }
}
```

2.8.2 “sex”

Nous recodons « 1 » pour « Total », « 2 » pour « Homme », « 3 » pour « Femme » et « 4 » pour « Autre ».

```
# Combine les noms des jeux de données
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

for (df_name in all_datasets) {
  df <- get(df_name)
  if ("sex" %in% names(df)) {
    df <- df %>%
      mutate(sex = case_when(
        sex == "T" ~ 1,
        sex == "M" ~ 2,
        sex == "F" ~ 3,
        sex == "O" ~ 4,
        TRUE       ~ NA_real_
      ))
    assign(df_name, df)
    cat("La variable 'sex' a été recodée en", df_name, "\n")
  }
}
```

2.8.3 “age”

```
# Combine les noms des jeux de données
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

# Initialise un vecteur vide pour stocker les modalités uniques
all_age_modalities <- character(0)

# On parcout chaque cadre de données pour extraire les modalités uniques de
 ↪  "age".
for (df_name in all_datasets) {
  df <- get(df_name)
  if ("age" %in% names(df)) {
    # Nous convertissons en caractères pour éviter d'éventuelles conversions
     ↪  de facteurs
    modalities <- unique(as.character(df$age))
    all_age_modalities <- union(all_age_modalities, modalities)
```

```
  }
}

# Trie les modalités pour faciliter la lecture
all_age_modalities <- sort(all_age_modalities)

# Affiche la liste des modalités uniques pour la variable « âge ».
cat("Liste des modalités uniques pour la variable 'age' :\n")
print(all_age_modalities)
```

```
# Combine les noms des jeux de données
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

for (df_name in all_datasets) {
  df <- get(df_name)
  if ("age" %in% names(df)) {
    df <- df %>%
      mutate(age = case_when(
        age == "YGE15" ~ 1,  # Corresponds to "Y15" recodé comme 1
        age == "Y15-24"  ~ 2,  # "YGE15-24" recodé en 2
        age == "YGE25"  ~ 3,  # "YGE25" recodé en 3
        TRUE ~ NA_real_
      ))
    assign(df_name, df)
    cat("La variable 'age' a été enregistré dans", df_name, "\n")
  }
}
```

```
library(dplyr)

# Créer une table de correspondance globale

# Récupère des modalités uniques dans tous les jeux de données
all_datasets <- c(dataset_ids_age, dataset_ids_sex)
all_isic <- character(0)

for (df_name in all_datasets) {
  df <- get(df_name)
  if ("isic_code_2dg" %in% names(df)) {
    # Normalise en mettant des majuscules et en supprimant les espaces
    modalities <- unique(toupper(trimws(df$isic_code_2dg)))
    all_isic <- union(all_isic, modalities)
```

```
  }
}

all_isic <- sort(all_isic)

# Définition des conditions particulières
special_codes <- c("TOTAL", "X")
other_codes <- setdiff(all_isic, special_codes)

# Création de la table de correspondance
mapping_table <- data.frame(
  modality = c(special_codes, other_codes),
  code = c(1, 2, seq(from = 3, length.out = length(other_codes))),
  stringsAsFactors = FALSE
)

print(mapping_table)

# Recoder la variable isic_code_2dg directement dans chaque jeu de données

for (df_name in all_datasets) {
  df <- get(df_name)
  if ("isic_code_2dg" %in% names(df)) {
    df <- df %>%
      # Normalize and temporarily store in a new column
      mutate(tmp_isic = toupper(trimws(isic_code_2dg))) %>%
      # Join with correspondence table
      left_join(mapping_table, by = c("tmp_isic" = "modality")) %>%
      # Replace isic_code_2dg with the numeric code
      mutate(isic_code_2dg = code) %>%
      # Delete temporary columns
      select(-tmp_isic, -code)

    assign(df_name, df)
    cat("The variable 'isic_code_2dg' has been recoded for", df_name, "\n")
  }
}
```

2.8.4 “edu”, “ste”, “ifl”, “est”

```
# On combine les noms des jeux de données
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

# Fonction permettant d'extraire l'union des modalités d'une variable donnée
extract_unique_modalities <- function(variable_name, df_names) {
  modalities <- character(0)
  for (df_name in df_names) {
    df <- get(df_name)
    if (variable_name %in% names(df)) {
      # Convertir en caractères pour s'assurer que l'on travaille avec des
      ↪  valeurs textuelles
      modalities <- union(modalities,
 ↪  unique(as.character(df[[variable_name]])))
    }
  }
  return(sort(modalities))
}

# Extraction de modalités uniques pour les variables d'intérêt
edu_modalities <- extract_unique_modalities("edu", all_datasets)
ste_modalities <- extract_unique_modalities("ste", all_datasets)
ifl_modalities <- extract_unique_modalities("ifl", all_datasets)
est_modalities <- extract_unique_modalities("est", all_datasets)

# Affichage de la liste
cat("Modalités uniques pour la variable 'edu':\n")
print(edu_modalities)
cat("\nModalités uniques pour la variable 'ste':\n")
print(ste_modalities)
cat("\nModalités uniques pour la variable 'ifl':\n")
print(ifl_modalities)
cat("\nModalités uniques pour la variable 'est':\n")
print(est_modalities)
```

```
all_datasets <- c(dataset_ids_age, dataset_ids_sex)
for (df_name in all_datasets) {
  df <- get(df_name)

  # Recode la variable 'edu'
  if ("edu" %in% names(df)) {
```

```
  df <- df %>%
    mutate(edu = case_when(
    edu == "LTB" ~ 1,
    edu == "BAS" ~ 2,
    edu == "INT" ~ 3,
    edu == "ADV" ~ 4,
      edu == "TOTAL" ~ 5,
      edu == "X" ~ 6,
      TRUE ~ NA_real_
    ))
}

# Recode variable 'ste'
if ("ste" %in% names(df)) {
  df <- df %>%
    mutate(ste = case_when(
      ste == "TOTAL" ~ 1,
      ste == "EES" ~ 2,
      ste == "SLF" ~ 3,
      ste == "X" ~ 4,
      TRUE ~ NA_real_
    ))
}

# Recode variable 'ifl'
if ("ifl" %in% names(df)) {
  df <- df %>%
    mutate(ifl = case_when(
      ifl == "TOTAL" ~ 1,
      ifl == "INFORMAL" ~ 2,
      ifl == "FORMAL" ~ 3,
      TRUE ~ NA_real_
    ))
}

# Recode variable 'est'
if ("est" %in% names(df)) {
  df <- df %>%
    mutate(est = case_when(
      est == "TOTAL" ~ 1,
      est == "S1-4" ~ 2,
      est == "S5-49" ~ 3,
      est == "SGE50" ~ 4,
```

```
        est == "X" ~ 5,
        TRUE ~ NA_real_
      ))
  }

  assign(df_name, df)
  cat("Recodage complet pour", df_name, "\n")
}
```

2.8.5 “region”, “subregion”, “income_group”

```
# Fonction permettant d'extraire l'union des modalités d'une variable donnée
extract_unique_modalities <- function(variable_name, df_names) {
  modalities <- character(0)
  for (df_name in df_names) {
    df <- get(df_name)
    if (variable_name %in% names(df)) {
      # Nous normalisons en convertissant les caractères en majuscules et en
      ↪  supprimant les espaces
      modalities <- union(modalities,
↪  unique(toupper(trimws(as.character(df[[variable_name]])))))
    }
  }
  sort(modalities)
}

all_datasets <- c(dataset_ids_age, dataset_ids_sex)

# Extraire des modalités uniques pour chaque variable
unique_region <- extract_unique_modalities("region", all_datasets)
unique_subregion <- extract_unique_modalities("subregion", all_datasets)
unique_income <- extract_unique_modalities("income_group", all_datasets)

# Afficher les listes pour vérification
cat("Modalités uniques pour la variable 'region':\n")
print(unique_region)
cat("\nModalités uniques pour la variable 'subregion':\n")
print(unique_subregion)
cat("\nModalités uniques pour la variable 'income_group':\n")
print(unique_income)
```

```r
# Définit les niveaux de recodage pour chaque variable
region_levels    <- c("AFRICA", "AMERICAS", "ASIA", "EUROPE", "OCEANIA")
subregion_levels <- c("AUSTRALIA AND NEW ZEALAND", "CENTRAL ASIA", "EASTERN
 ↪  ASIA", "EASTERN EUROPE",
                      "LATIN AMERICA AND THE CARIBBEAN", "MELANESIA",
                       ↪  "MICRONESIA", "NORTHERN AFRICA",
                      "NORTHERN AMERICA", "NORTHERN EUROPE", "POLYNESIA",
                       ↪  "SOUTH-EASTERN ASIA",
                      "SOUTHERN ASIA", "SOUTHERN EUROPE", "SUB-SAHARAN
                       ↪  AFRICA", "WESTERN ASIA",
                      "WESTERN EUROPE")
income_levels    <- c("HIGH INCOME", "LOW INCOME", "LOWER MIDDLE INCOME",
 ↪  "UPPER MIDDLE INCOME")

all_datasets <- c(dataset_ids_age, dataset_ids_sex)

# Pour chaque jeu de données, si la variable existe, on recode directement
for (df_name in all_datasets) {
  df <- get(df_name)

  # Recode region
  if ("region" %in% names(df)) {
    df <- df %>%
      mutate(region = as.integer(factor(toupper(trimws(region)), levels =
       ↪  region_levels)))
  }

  # Recode subregion
  if ("subregion" %in% names(df)) {
    df <- df %>%
      mutate(subregion = as.integer(factor(toupper(trimws(subregion)), levels
       ↪  = subregion_levels)))
  }

  # Recode income_group
  if ("income_group" %in% names(df)) {
    df <- df %>%
      mutate(income_group = as.integer(factor(toupper(trimws(income_group)),
       ↪  levels = income_levels)))
  }

  assign(df_name, df)
```

```
  cat("Recodage de 'region', 'subregion' and 'income_group' réalisé pour",
   ↪  df_name, "\n")
}
```

2.8.6 “year”

```
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

for (df_name in all_datasets) {
  df <- get(df_name)
  if ("year" %in% names(df)) {
    df <- df %>%
      mutate(year = as.numeric(year))
    assign(df_name, df)
    cat("La variable 'year' a été convertie en valeur numérique pour les
     ↪  raisons suivantes", df_name, "\n")
  }
}
```

```
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

# On initialise un vecteur pour stocker les modalités uniques de 'year'
all_years <- numeric(0)

# Parcour chaque cadre de données et extrait les modalités uniques de 'year'
for (df_name in all_datasets) {
  df <- get(df_name)
  if ("year" %in% names(df)) {
    # On suppose que la variable 'year' est déjà numérique
    years <- unique(df$year)
    all_years <- union(all_years, years)
  }
}

# Trie les années obtenues pour une présentation ordonnée
all_years <- sort(all_years)

cat("Les modalités uniques pour la variable 'year' sont les suivantes:\n")
print(all_years)
```

2.8.7 “EU_ISIC4_2dg”

```
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

for (df_name in all_datasets) {
  df <- get(df_name)
  if ("EU_ISIC4_2dg" %in% names(df)) {
    df <- df %>%
      mutate(EU_ISIC4_2dg = as.numeric(EU_ISIC4_2dg))
    assign(df_name, df)
    cat("La variable 'EU_ISIC4_2dg' a été convertie en numérique pour les
     ↪  besoins suivants", df_name, "\n")
  }
}
```

2.8.8 “SEI” and “SIDS”

```
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

for (df_name in all_datasets) {
  df <- get(df_name)

  if ("SEI" %in% names(df)) {
    df <- df %>%
      mutate(SEI = as.numeric(SEI) + 1)
  }

  if ("SIDS" %in% names(df)) {
    df <- df %>%
      mutate(SIDS = as.numeric(SIDS) + 1)
  }

  assign(df_name, df)
  cat("Les variables SEI et SIDS ont été recodées pour les besoins de
   ↪  l'étude", df_name, "\n")
}
```

2.9 Recording

```
# Liste des identifiants de jeux de données
dataset_ids_age <- c(
  "EMP_TEMP_AGE_EC2_NB_A",
  "EMP_NIFL_AGE_EC2_NB_A",
  "EMP_NIFL_AGE_EC2_RT_A",
  "EMP_PIFL_AGE_EC2_NB_A",
  "EMP_PIFL_AGE_EC2_RT_A",
  "EES_TEES_AGE_EC2_NB_A",
  "HOW_TEMP_AGE_EC2_NB_A",
  "HOW_XEES_AGE_EC2_NB_A"
)

dataset_ids_sex <- c(
  "EMP_TEMP_SEX_EDU_EC2_NB_A",
  "EMP_TEMP_SEX_STE_EC2_NB_A",
  "EMP_TEMP_SEX_IFL_EC2_NB_A",
  "EMP_TEMP_SEX_EST_EC2_NB_A",
  "EMP_NIFL_SEX_EC2_NB_A",
  "EMP_NIFL_SEX_EC2_RT_A",
  "EMP_PIFL_SEX_EC2_NB_A",
  "EMP_PIFL_SEX_EC2_RT_A",
  "EES_TEES_SEX_EC2_NB_A",
  "HOW_TEMP_SEX_EC2_NB_A",
  "HOW_XEES_SEX_EC2_NB_A",
  "EMP_CARE_SEX_EC2_NB_A",
  "EMP_STEM_SEX_EC2_NB_A",
  "EMP_TOUR_SEX_EC2_NB_A",
  "EMP_PUBL_SEX_EC2_NB_A"
)

# Combine tous les noms dans un seul vecteur
all_datasets <- c(dataset_ids_age, dataset_ids_sex)

# Définit le répertoire de sauvegarde
save_directory <- "/Users/.../new_SAGE"

# Parcourt chaque jeu de données et enregistre au format RDS dans le dossier
 ↪  spécifié
for (df_name in all_datasets) {
  df_object <- get(df_name)
```

```
  file_path <- file.path(save_directory, paste0(df_name, ".rds"))
  saveRDS(df_object, file = file_path)
  cat("Le jeu de données", df_name, "a été sauvegardé dans", file_path, "\n")
}
```

3 “Sustainability” préparation des données

```
install.packages("dplyr")
install.packages("tidyverse")
install.packages("readxl")
install.packages("naniar")
library(dplyr)
library(tidyverse)
library(readxl)
library(naniar)
```

3.1 Liste des variables par modèles

3.1.1 Performance économique

Variable	Définition	Interprétation	Source	Variable retenue	Type de variable
PIB annuel			WDI (World Bank)	“GDP (constant 2015 US$) - ID:”NY.GDP.MKTP.KD”	Variable dépendante
Stock de capital (K)	Facteur de production classique. Quantité de ressources physiques et/ou immatérielles dont dispose le pays pour produire des B&S (infrastructures, machines, logiciels,…).	Reflète les investissements productifs cumulés (infrastructures, équipements).	PWT	“Capital stock at constant 2017 national prices (in mil. 2017 US$)”	Variable explicative

Variable	Rôle	Interprétation	Source	Variable retenue	Type de variable
Emploi total (L)	Facteur de production classique. Nombre de personnes en emplois.	Mesure la contribution du travail à la création de valeur.	WDI (World Bank)	"Labor force participation rate, total (% of total population ages 15+) (modeled ILO estimate) - ID:"SL.TLF.CACT.ZS"	Variable explicative
Part des secteurs verts	Intensité verte.	Variable principal d'intérêt : représente les potentialités de transformation structurelle vers une économie verte.	SAGE	GreenShare (cf ci-dessous un détail de la construction de l'indicateur GreenShare).	Variable principal d'intérêt.
Capital humain	Productivité et efficacité du travail.	Le capital humain améliore l'efficacité du facteur travail et renforce l'effet des emplois verts.	PWT	"Human capital index, based on years of schooling and returns to education".	Variable explicative
Investissements publics	Stimulent la demande et de la production	Les dépenses publiques d'infrastructure ou R&D soutiennent directement la productivité et l'emploi vert	WDI (World Bank)	"Expense (% of GDP)" - ID: "GC.XPN.TOTL.GD.ZS".	Variable de contrôle

Variable	Définition	Interprétation	Source	Variable retenue	Type de variable
Ouverture commerciale	(Exportationsi,t + Importationsi,t) / PIBi,t	Mesure standard de l'intégration internationale, appelée taux d'ouverture. L'ouverture commerciale favorise la diffusion technologique, notamment en matière de technologies vertes. Elle permet l'accès à de nouveaux marchés pour les secteurs innovants ou bas carbone. Mais elle peut aussi renforcer les spécialisations extractives, d'où la nécessité de la contrôler.	WDI (World Bank)	1- "Exports of goods and services (constant 2015 US$) - ID:"NE.EXP.GNFS.KD" 2- "Imports of goods and services (constant 2015 US$)" - ID: NE.IMP.GNFS.KD" 3- cf PIB annuel.	Variable de contrôle
Qualité de la régulation	Reflète la capacité de l'État à orienter l'activité économique via des incitations cohérentes, des normes stables et un cadre propice à l'innovation verte. C'est un facilitateur économique pour les emplois verts, sans être confondu avec la qualité des services publics ou la corruption.	Cible le lien public–privé : c'est le levier clé de la transition verte dans une économie de marché. Une bonne régulation attire les investissements, facilite l'émergence de nouveaux secteurs, et évite le verrouillage dans des modèles carbonés.	WGI (World Bank)	"Regulatory Quality"	Variable de contrôle

Variable	Définition	Interprétation	Source	Variable retenue	Type de variable
Taux d'électrification	Infrastructure de base indispensable pour toute forme de productivité - notamment secteurs verts (énergies renouvelables, industrie propre, services numériques…) ==> Proxy des infrastructures de base ici.	Dans les pays en développement, une faible électrification freine l'absorption des innovations, les investissements, et la création d'emplois stables. C'est aussi un indicateur synthétique du niveau d'infrastructure territoriale, sans être trop corrélé au PIB ou au capital physique. Il capte une barrière structurelle à la performance, indépendamment du capital ou du travail, notamment dans les économies à bas revenus.	WDI (World Bank)	"Access to electricity (% of population)" - ID: "EG.ELC.ACCS.ZS"	Variable de contrôle

Variable	Rôle	Interprétation	Source	Variable retenue	Type de variable
Structure sectorielle	Contrôle l'effet de la composition économique sur la performance. On se demande si le poids de certains secteurs (industrie, agriculture, services) fausse l'effet mesuré de l'intensité verte sur le PIB ?	Nous avons 4 variables : 3 pourcentages pour 3 grands secteurs, et un score entre 0 et 1, plus c'est élevé, plus le pays est dépendant d'un secteur. $$HHI_{i,t} = \left(\frac{VA_{\text{agr},i,t}}{100}\right)^2 + \left(\frac{VA_{\text{ind},i,t}}{100}\right)^2 + \left(\frac{VA_{\text{serv},i,t}}{100}\right)^2 \quad (1)$$	WDI (World Bank)	Nous obtenons 3 variables (% du PIB) : 1- "Agriculture, forestry and fishing, value added (constant 2015 US$)" - ID: "NV.AGR.TOTL.KD" 2- "Industry (including construction), value added (constant 2015 US$)" - ID: "NV.IND.TOTL.KD" 3- "Services, value added (constant 2015 US$)" - ID: "NV.SRV.TOTL.KD" Nous calculons ensuite l'indice de Herfindahl simplifié (3 secteurs) : HHIi,t = (	Variables de contrôle

3.1.2 Soutenabilité à long terme

Variable	Rôle	Interprétation	Source	Variable retenue	Type de variable
Épargne véritable (exprimée en part du PIB%), annuel			WDI (World Bank)	"Adjusted net savings, excluding particulate emission damage (% of GNI)" - ID: "NY.ADJ.SVNX.GN.ZS"	Variable dépendante
Part des secteurs verts	Intensité verte.	Variable principal d'intérêt : représente les potentialités de transformation structurelle vers une économie verte.	SAGE	GreenShare (cf ci-dessous un détail de la construction de l'indicateur GreenShare).	Variable principal d'intérêt.
Croissance du PIB/hab.	Dynamique économique immédiate.	Permet de distinguer les effets propres de l'emploi vert sur la soutenabilité de ceux liés à la conjoncture économique. Une croissance rapide peut améliorer l'ANS mécaniquement (via l'épargne), sans transformation structurelle.	WDI (World Bank)	"GDP per capita growth (annual %)" - ID: "NY.GDP.PCAP.KD.ZG"	

Variable	Rôle	Interprétation	Source	Variable retenue	Type de variable
PIB par habitant (log)	Niveau de développement.	Les pays plus riches ont plus de marge de manœuvre budgétaire et institutionnelle pour investir dans le capital humain, naturel ou vert. Il est essentiel de neutraliser cet effet de niveau dans l'analyse.	WDI (World Bank)	"GDP per capita (constant 2015 US$" - ID: "NY.GDP.PCAP.KD (log transformée)	
Capital humain (1)	Mesure indirecte de soutenabilité productive.	Un meilleur capital humain augmente la productivité, réduit les gaspillages et renforce la capacité d'absorption des innovations vertes. Il est aussi un déterminant direct de l'ANS à travers l'investissement éducatif.	WDI (World Bank) ou PWT	WDI : "Educational attainment, at least completed upper secondary, population 25+, total (%) (cumulative)" - ID: "SE.SEC.CUAT.UP.ZS"	
Capital humain (2)	Idem.	Idem.	PWT	"Human capital index, based on years of schooling and returns to education" PWT	

Variable	Rôle	Interprétation	Source	Variable retenue	Type de variable
Qualité institutionnelle / gouvernance (1)	Efficacité des politiques de transition	Reflète la capacité d'un pays à concevoir, appliquer et maintenir des politiques de long terme (notamment environnementales ou fiscales) — condition essentielle pour qu'une trajectoire d'emploi vert se traduise en gains soutenables.	WGI (World Bank)	"Government Effectiveness"	
Qualité institutionnelle / gouvernance (2)	Contrôle de corruption, stabilité de l'allocation des ressources publiques.	La corruption mine les capacités d'investissement dans les biens publics et dans la transition écologique. Elle peut affaiblir l'effet positif d'un emploi vert sur l'ANS via mauvaise gouvernance ou détournement.	WGI (World Bank)	"Control of Corruption"	
Political Rights			Freedom in the World 2025	"Political Rights"	
Civil Rights			Freedom in the World 2025	"Civil Rights"	

Variable	Rôle	Interprétation	Source	Variable retenue	Type de variable
Dépendance aux ressources naturelles (1)	Vulnérabilité à l'exploitation du capital naturel	Un pays très dépendant de ses ressources naturelles peut afficher une croissance forte tout en dégradant son ANS (via extraction non compensée). Ce contrôle capte une trajectoire de croissance non soutenable.	WDI (World Bank)	"Fuel exports (% of merchandise exports)" - ID: "TX.VAL.FUEL.ZS.UN"	
Dépendance aux ressources naturelles (2)	Idem.	Idem.	WDI (World Bank)	"Total naturel resources rents (% of GDP)" - ID: "NY.GDP.TOTL.RT.ZS"	
Taux d'inactivité	Pression latente sur le système productif et social. Correspond à la population en âge de travailler (généralement 15-64 ans) qui n'est ni en emploi, ni au chômage. C'est l'inverse du taux d'activité (Labour force participation rate, LFPR).	Un taux d'inactivité élevé indique une sous-utilisation du capital humain disponible. Cela réduit l'épargne, augmente les charges sociales, et fragilise les trajectoires de développement à long terme.	WDI (World Bank)	"Age dependency ratio (% of working-age population) - ID:"SP.POP.DPND"	
Niveau d'urbanisation	Structure spatiale de la transition.	L'urbanisation modifie la structure de consommation, d'investissement et les contraintes environnementales. Elle peut être source d'emplois verts (éco-construction, mobilité verte) ou de pressions écologiques supplémentaires.	WDI (World Bank)	"Urban population (% of total population) - ID: SP.URB.TOTL.IN.ZS""	

Variable	Rôle	Interprétation	Source	Variable retenue	Type de variable
Structure par âge	Pressions intergénérationnelles.	Une population jeune implique des besoins accrus en éducation, emploi et capital humain ; une population âgée implique des coûts sociaux plus élevés. Ces dynamiques affectent l'orientation des investissements publics et donc la soutenabilité.	WDI (World Bank)	"Population ages 0-14 (% of total population)" - ID: "SP.POP.0014.TO.ZS" et "Population ages 65 and above (% of total population" - ID: "SP.POP.65UP.TO.ZS"	
Indice de vulnérabilité climatique	Pour capturer les risques structurels liés à l'environnement.	Capture l'exposition aux risques climatiques (élévation des températures, sécheresse, catastrophes). Affecte directement la soutenabilité et la capacité à développer des emplois verts résilients.	Notre Dame Global Adaptation Index (ND-GAIN)	"ND-GAIN vulnerability score"	
Indice d'informalité	Pour contrôler l'effet d'un marché du travail fragmenté		ILOSTAT	"EMP_NIFL_SEX_AGE_EDU_NB_A" et on garde que les totaux de sex, age et edu.	

4 Préparation par sources

4.0.1 Tableau des variables et codes par source

Source	Variable	ID origin	SAGE name
World Development Index - WDI (World Bank)	PIB (annuel, constant 2015 US$)	NY.GDP.PCAP.KD	gdp_-const
	PIB par habitant (constant 2015 US$)	NY.GDP.MKTP.KD	gdppc_-const
	Croissance PIB/habitant (annuel, %)	NY.GDP.PCAP.KD.ZG	gdppc_-growth
	Dépenses publiques (% PIB)	GC.XPN.TOTL.GD.ZS	gov_-exp_-pct-gdp
	Taux de participation (LFPR, 15+)	SL.TLF.CACT.ZS	lf_-par-ticipa-tion
	Exportations de biens et services (valeurs constantes)	NE.EXP.GNFS.KD	exports_-const
	Importations de biens et services (valeurs constantes)	NE.IMP.GNFS.KD	imports_-const
	Ouverture économique (Exporta-tions/importations)/PIB	*Calcul des auteurs.*	open
	Pourcentage ouverture économique ((Exporta-tions/importations)/PIB)*100	*Calcul des auteurs.*	open_-pct

Source	Variable	ID origin	SAGE name
	Accès à l'électricité (% population)	EG.ELC.ACCS.ZS	access_-electricity
	VA - Agriculture (valeurs constantes)	NV.AGR.TOTL.KD	va_-agriculture
	VA - Industrie (valeurs constantes)	NV.IND.TOTL.KD	va_-industry
	VA - Services (valeurs constantes)	NV.SRV.TOTL.KD	va_-services
	Épargne véritable ajustée (% GNI)	NY.ADJ.SVNX.GN.ZS	adj_-savings
	Exportations de combustibles (% exportations)	TX.VAL.FUEL.ZS.UN	fuel_-exports_-pct_-exports
	Rentes naturelles totales (% PIB)	NY.GDP.TOTL.RT.ZS	nat_-resource_-rents_-pct-gdp
	Population 0–14 ans (% population)	SP.POP.0014.TO.ZS	pop_-0_-14_-pct
	Population 65+ (% population)	SP.POP.65UP.TO.ZS	pop_-65plus_-pct
	Taux d'urbanisation (% population urbaine)	SP.URB.TOTL.IN.ZS	urban_-pop_-pct

Source	Variable	ID origin	SAGE name
	Ratio de dépendance démographique	SP.POP.DPND	age_dependency_ratio
	Capital humain (éducation secondaire atteinte, %)	SE.SEC.CUAT.UP.ZS	educ_secondary_complete
	Gross capital formation (constant 2015 US$)	NE.GDI.TOTL.KD	gross_capital_const
	Gross capital formation (% of GDP)	NE.GDI.TOTL.ZS	gross_capital_pct_gdp
Penn World Table (PWT	Stock de capital net	rkna	rkna
	Capital humain (index basé sur les années d'études)	hc	hc
Freedom in the World		Deux indicateurs "Political Rights" et "Civil Liberties", scorés sur une échelle de 1 à 7 : plus le score est faible plus il a de political rights / ou de civil liberties. On fait la moyenne des deux pour créer la variable "democracy".	• democracy • PR • CL
SAGE	Part des emplois verts	GreenShare	
ND-GAIN	Indice de vulnérabilité climatique	ND-GAIN vulnerability score	vulnerability

Source	Variable	ID origin	SAGE name
Informal Economy Database	Part des emplois informels (% total)	EMP_NIFL_SEX_AGE_EDU_NB_A	• dge • mimic
Chrisendo et al. (2024)	Coefficient de Gini		gini

4.1 Réplication du jeu de données réunissant ces variables

4.1.1 Identification Variables

```
install.packages("readxl")
library(readxl)
```

4.1.2 Données

- Variables “country”, “iso2c”, “iso3c”, “region”, “subregion” viennent d’un github. On a reformulé les variables de cette source avec nos noms directement sur excel. Nous téléchargeons et nommons ce fichier “first_iv”.

```
first_iv <- read_excel("/Users/.../first_iv.xlsx")
```

- Variables “developed_economies”, “developing_economies_excl_china”, “developing_economies_excl_LDCs”, “LLDCs”, “LIDE”, “MIDE”, “HIDE”, “BRICS”, “EU”, “G20”, “G77” viennent de UNCTAD. Nous avons un fichier .xlsx intitulé “UNCTAD_classification”. Nous reprenons les classifications des économies selon UNCTAD. Aller sur ce site : https://unctadstat.unctad.org/EN/Classifications.html –> Télécharger fichier .xlsx intitulé “All groups compositions”.

```
second_iv <- read_excel("/Users/.../UNCTAD_classification.xlsx")
```

4.1.3 Fusions

On crée des colonnes dans “first_iv” qui indiqueront si un pays appartient ou non à une des catégories de “second_iv”.

```
# Renomme les colonnes de second_iv
second_iv <- second_iv %>%
  rename(
    developed_economies = `Developed Economies`,
    developing_economies_excl_china = `Developing Economies (excl_China)`,
     ↪
    developing_economies_excl_LDCs = `Developing Economies (excl_LDCs)`,
     ↪
    LLDCs = `LLDCs (Landlocked developing countries)`,
    LIDE = `Low-income developing economies`,
    MIDE = `Middle-income developing economies`,
    HIDE = `High-income developing economies`,
    BRICS = BRICS,
    EU = `European Union (2020 …)`,
    G20 = G20,
    G77 = G77
  )

# Ajout de 11 variables booléennes à first_iv
first_iv <- first_iv %>%
  mutate(
    developed_economies = ifelse(country %in% second_iv$developed_economies,
     ↪  1, 0),
    developing_economies_excl_china = ifelse(country %in%
     ↪  second_iv$developing_economies_excl_china, 1, 0),
    developing_economies_excl_LDCs = ifelse(country %in%
     ↪  second_iv$developing_economies_excl_LDCs, 1, 0),
    LLDCs = ifelse(country %in% second_iv$LLDCs, 1, 0),
    LIDE = ifelse(country %in% second_iv$LIDE, 1, 0),
    MIDE = ifelse(country %in% second_iv$MIDE, 1, 0),
    HIDE = ifelse(country %in% second_iv$HIDE, 1, 0),
    BRICS = ifelse(country %in% second_iv$BRICS, 1, 0),
    EU = ifelse(country %in% second_iv$EU, 1, 0),
    G20 = ifelse(country %in% second_iv$G20, 1, 0),
    G77 = ifelse(country %in% second_iv$G77, 1, 0)
  )
```

On vérifie si la fusion est bonne en listant tous les pays pour lesquels toutes les nouvelles

variables de classification contiennent uniquement des “0” (aucune appartenance à une classification).

```
# Filtre les pays où toutes les nouvelles variables de classification sont
 ↪  égales à 0
pays_sans_classification <- first_iv %>%
  filter(
    developed_economies == 0 &
    developing_economies_excl_china == 0 &
    developing_economies_excl_LDCs == 0 &
    LLDCs == 0 &
    LIDE == 0 &
    MIDE == 0 &
    HIDE == 0 &
    BRICS == 0 &
    EU == 0 &
    G20 == 0 &
    G77 == 0
  ) %>%
  select(country)

unique(pays_sans_classification$country)
```

Cela nous donne le tableau ci-après avec les noms problématiques et leurs noms selon le dataframe:

Country name second_iv	Country name first_iv	Corrected country name
Åland Islands	Aland Islands	Aland Islands
No classif.	Antarctica	Antarctica
Bolivia (Plurinational State of)	Bolivia Plurinational State of	Bolivia Plurinational State of
Bonaire, Sint Eustatius and Saba	Bonaire Sint Eustatius and Saba	Bonaire Sint Eustatius and Saba
Dem. Rep. of the Congo	Congo Democratic Republic of the	Congo Democratic Republic of the
Côte d’Ivoire	C√¥te d’Ivoire	Cote d’Ivoire
Curaçao	Cura√ßao	Curacao
China, Hong Kong SAR	Hong Kong	Hong Kong

Country name second_iv	Country name first_iv	Corrected country name
Iran (Islamic Republic of)	Iran Islamic Republic of	Iran Islamic Republic of
Dem. People's Rep. of Korea	Korea Democratic People's Republic of	Korea Democratic People's Republic of
Republic of Korea	Korea Republic of	Korea Republic of
Lao People's Dem. Rep.	Lao People's Democratic Republic	Lao People's Democratic Republic
China, Macao SAR	Macao	Macao
Micronesia (Federated States of)	Micronesia Federated States of	Micronesia Federated States of
Republic of Moldova	Moldova Republic of	Moldova Republic of
Netherlands (Kingdom of the)	Netherlands Kingdom of the	Netherlands Kingdom of the
State of Palestine	Palestine State of	Palestine State of
Réunion	R√©union	Reunion
Saint Barthélemy	Saint Barth√©lemy	Saint Barthelemy
Saint Helena	Saint Helena Ascension and Tristan da Cunha	Saint Helena Ascension and Tristan da Cunha
Svalbard and Jan Mayen Islands	Svalbard and Jan Mayen	Svalbard and Jan Mayen
China, Taiwan Province of	Taiwan Province of China	Taiwan Province of China
United Republic of Tanzania	Tanzania United Republic of	Tanzania United Republic of
Türkiye	T√ºrkiye	Turkey
United Kingdom	United Kingdom of Great Britain and Northern Ireland	United Kingdom of Great Britain and Northern Ireland
United States	United States of America	United States of America
Venezuela (Bolivarian Rep. of)	Venezuela Bolivarian Republic of	Venezuela Bolivarian Republic of
British Virgin Islands	Virgin Islands (British)	Virgin Islands (British)
United States Virgin Islands	Virgin Islands (U.S.)	Virgin Islands (U.S.)

Nous gardons les noms de la colonne "Corrected country name" (nous gardons au mieux les noms de first_iv comme référence):

```r
# Remplacer pour "second_iv"
## Vecteur des noms de pays à remplacer dans second_iv
pays_a_remplacer <- c(
  "Åland Islands", "Bolivia (Plurinational State of)", "Bonaire, Sint
   ↪ Eustatius and Saba",
  "Dem. Rep. of the Congo", "Côte d'Ivoire", "Curaçao", "China, Hong Kong
   ↪ SAR",
  "Iran (Islamic Republic of)", "Dem. People's Rep. of Korea", "Republic of
   ↪ Korea",
  "Lao People's Dem. Rep.", "China, Macao SAR", "Micronesia (Federated States
   ↪ of)",
  "Republic of Moldova", "Netherlands (Kingdom of the)", "State of
   ↪ Palestine",
  "Réunion", "Saint Barthélemy", "Saint Helena", "Svalbard and Jan Mayen
   ↪ Islands",
  "China, Taiwan Province of", "United Republic of Tanzania", "Türkiye",
   ↪ "United Kingdom",
  "United States", "Venezuela (Bolivarian Rep. of)", "British Virgin
   ↪ Islands",
  "United States Virgin Islands", "Wallis and Futuna Islands"
)

## Vecteur des noms corrigés
noms_corriges <- c(
  "Aland Islands", "Bolivia Plurinational State of", "Bonaire Sint Eustatius
   ↪ and Saba",
  "Congo Democratic Republic of the", "Cote d'Ivoire", "Curacao", "Hong
   ↪ Kong",
  "Iran Islamic Republic of", "Korea Democratic People's Republic of", "Korea
   ↪ Republic of",
  "Lao People's Democratic Republic", "Macao", "Micronesia Federated States
   ↪ of",
  "Moldova Republic of", "Netherlands Kingdom of the", "Palestine State of",
  "Reunion", "Saint Barthelemy", "Saint Helena Ascension and Tristan da
   ↪ Cunha",
  "Svalbard and Jan Mayen", "Taiwan Province of China", "Tanzania United
   ↪ Republic of",
  "Turkey", "United Kingdom of Great Britain and Northern Ireland", "United
   ↪ States of America",
  "Venezuela Bolivarian Republic of", "Virgin Islands (British)", "Virgin
   ↪ Islands (U.S.)",       "Wallis and Futuna"
)
```

```r
## Fonction pour remplacer les noms dans toutes les colonnes de second_iv
remplacer_modalites <- function(df) {
  ### Appliquer sur toutes les colonnes
  df[] <- lapply(df, function(col) {
    if (is.character(col)) {
      #### Remplacer les noms des pays s'ils apparaissent comme modalités
      sapply(col, function(p) {
        if (p %in% pays_a_remplacer) {
          noms_corriges[which(pays_a_remplacer == p)]
        } else {
          p
        }
      })
    } else {
      col
    }
  })
  return(df)
}

## Appliquer la fonction pour remplacer les noms dans second_iv
second_iv <- remplacer_modalites(second_iv)

# Vecteur des noms de pays à remplacer dans first_iv
pays_a_remplacer_first_iv <- c(
  "Aland Islands", "Bolivia Plurinational State of", "Bonaire Sint Eustatius
   ↪ and Saba",
  "C√¥te d'Ivoire", "Cura√ßao", "Congo Democratic Republic of the", "Hong
   ↪ Kong",
  "Iran Islamic Republic of", "Korea Democratic People's Republic of",
   ↪ "Korea Republic of",
  "Lao People's Democratic Republic", "Macao", "Micronesia Federated States
   ↪ of",
  "Moldova Republic of", "Netherlands Kingdom of the", "Palestine State of",
   ↪ "R√©union",
  "Saint Barth√©lemy", "Saint Helena Ascension and Tristan da Cunha",
   ↪ "Svalbard and Jan Mayen",
  "Taiwan Province of China", "Tanzania United Republic of", "T√ºrkiye",
  "United Kingdom of Great Britain and Northern Ireland", "United States of
   ↪ America",
  "Venezuela Bolivarian Republic of", "Virgin Islands (British)", "Virgin
   ↪ Islands (U.S.)"
```

```
)
# Vecteur des noms corrigés
noms_corriges_first_iv <- c(
  "Aland Islands", "Bolivia Plurinational State of", "Bonaire Sint Eustatius
   ↪  and Saba",
  "Cote d'Ivoire", "Curacao", "Congo Democratic Republic of the", "Hong
   ↪  Kong",
  "Iran Islamic Republic of", "Korea Democratic People's Republic of",
   ↪  "Korea Republic of",
  "Lao People's Democratic Republic", "Macao", "Micronesia Federated States
   ↪  of",
  "Moldova Republic of", "Netherlands Kingdom of the", "Palestine State of",
   ↪  "Reunion",
  "Saint Barthelemy", "Saint Helena Ascension and Tristan da Cunha",
   ↪  "Svalbard and Jan Mayen",
  "Taiwan Province of China", "Tanzania United Republic of", "Turkey",
  "United Kingdom of Great Britain and Northern Ireland", "United States of
   ↪  America",
  "Venezuela Bolivarian Republic of", "Virgin Islands (British)", "Virgin
   ↪  Islands (U.S.)"
)
# Fonction pour remplacer les noms dans first_iv$country
first_iv$country <- sapply(first_iv$country, function(p) {
  if (p %in% pays_a_remplacer_first_iv) {
    noms_corriges_first_iv[which(pays_a_remplacer_first_iv == p)]
  } else {
    p
  }
})
```

Les noms sont homogénisés au mieux, on refait l'ajout des 11 variables:

```
# Ajout de 11 variables booléennes à first_iv
first_iv <- first_iv %>%
  mutate(
    developed_economies = ifelse(country %in% second_iv$developed_economies,
     ↪  1, 0),
    developing_economies_excl_china = ifelse(country %in%
     ↪  second_iv$developing_economies_excl_china, 1, 0),
    developing_economies_excl_LDCs = ifelse(country %in%
     ↪  second_iv$developing_economies_excl_LDCs, 1, 0),
    LLDCs = ifelse(country %in% second_iv$LLDCs, 1, 0),
```

```
    LIDE = ifelse(country %in% second_iv$LIDE, 1, 0),
    MIDE = ifelse(country %in% second_iv$MIDE, 1, 0),
    HIDE = ifelse(country %in% second_iv$HIDE, 1, 0),
    BRICS = ifelse(country %in% second_iv$BRICS, 1, 0),
    EU = ifelse(country %in% second_iv$EU, 1, 0),
    G20 = ifelse(country %in% second_iv$G20, 1, 0),
    G77 = ifelse(country %in% second_iv$G77, 1, 0)
  )
```

On teste de nouveaux s'il y a des pays sans classification:

```
# Filtrer les pays où toutes les nouvelles variables de classification sont
 ↪  égales à 0
pays_sans_classification <- first_iv %>%
  filter(
    developed_economies == 0 &
    developing_economies_excl_china == 0 &
    developing_economies_excl_LDCs == 0 &
    LLDCs == 0 &
    LIDE == 0 &
    MIDE == 0 &
    HIDE == 0 &
    BRICS == 0 &
    EU == 0 &
    G20 == 0 &
    G77 == 0
  ) %>%
  select(country)
```

Il reste maintenant une variable à ajouter qui est "income_group" que l'on tire des données de World Development Indicators (WDI) de la Banque Mondiale. Nous avons téléchargé un fichier .xlsx intitulé "incomegroup_WDI.xlsx"

```
third_iv <- read_excel("/Users/.../incomegroup_WDI.xlsx")

# Fusion de first_iv et third_iv en gardant toutes les observations de
 ↪  third_iv
first_iv <- first_iv %>%
  right_join(third_iv, by = "iso3c")
```

Le fichier "incomegroup_WDI.xlsx" ajoute 49 codes ISO, qui ne sont pas informés dans "first_iv" :

```r
unique(first_iv$iso3c)
```

Voici la liste de codes ISO supplémentaires: “AFE” “AFW” “ARB” “CEB” “CHI” “CSS” “EAP” “EAR” “EAS” “ECA” “ECS” “EMU” “EUU” “FCS” “HIC” “HPC” “IBD” “IBT” “IDA” “IDB” “IDX” “LAC” “LCN” “LDC” “LIC” “LMC” “LMY” “LTE” “MEA” “MIC” “MNA” “NAC” “OED” “OSS” “PRE” “PSS” “PST” “SAS” “SSA” “SSF” “SST” “TEA” “TEC” “TLA” “TMN” “TSA” “TSS” “UMC” “WLD” “XKX”.

Le seul que nous gardons est “XKX” qui désigne le Kosovo. Nous appliquons cet ajout à first_iv:

```r
# Renomme la colonne "sub-region" en "sub_region"
first_iv <- first_iv %>%
  rename(subregion = `sub-region`)

# Modification des informations pour XKX
  first_iv <- first_iv %>%
    mutate(
      iso2c = ifelse(iso3c == "XKX", "XK", iso2c),
      income_group = ifelse(iso3c == "XKX", "Upper middle income",
       ↪  income_group),
      developed_economies = ifelse(iso3c == "XKX", 1, developed_economies),
      developing_economies_excl_china = ifelse(iso3c == "XKX", 0,
       ↪  developing_economies_excl_china),
      developing_economies_excl_LDCs = ifelse(iso3c == "XKX", 0,
       ↪  developing_economies_excl_LDCs),
      LLDCs = ifelse(iso3c == "XKX", 0, LLDCs),
      LIDE = ifelse(iso3c == "XKX", 0, LIDE),
      MIDE = ifelse(iso3c == "XKX", 0, MIDE),
      HIDE = ifelse(iso3c == "XKX", 0, HIDE),
      BRICS = ifelse(iso3c == "XKX", 0, BRICS),
      EU = ifelse(iso3c == "XKX", 0, EU),
      G20 = ifelse(iso3c == "XKX", 0, G20),
      G77 = ifelse(iso3c == "XKX", 0, G77),
      country = ifelse(iso3c == "XKX", "Kosovo", country),
      region = ifelse(iso3c == "XKX", "Europe", region),
      subregion = ifelse(iso3c == "XKX", "Eastern Europe", subregion)
  )
```

On supprime les codes iso3c en trop :

```r
# Vecteur des codes iso3c à supprimer
codesiso3c_a_supprimer <- c("AFE", "AFW", "ARB", "CEB", "CHI", "CSS", "EAP",
 ↪  "EAR", "EAS", "ECA", "ECS", "EMU", "EUU", "FCS", "HIC", "HPC", "IBD",
 ↪  "IBT", "IDA", "IDB", "IDX", "LAC", "LCN", "LDC", "LIC", "LMC", "LMY",
 ↪  "LTE", "MEA", "MIC", "MNA", "NAC", "OED", "OSS", "PRE", "PSS", "PST",
 ↪  "SAS", "SSA","SSF", "SST", "TEA", "TEC", "TLA", "TMN", "TSA", "TSS",
 ↪  "UMC", "WLD")

# Supprime les lignes avec ces codes iso3c
first_iv <- first_iv %>%
  filter(!iso3c %in% codesiso3c_a_supprimer)
```

```r
# Ajoute "Channel Islands"
first_IV <- first_IV %>%
  mutate(across(
    c(
      developed_economies, developing_economies_excl_china,
 ↪  developing_economies_excl_LDCs,
      LLDCs, LIDE, MIDE, HIDE, BRICS, EU, G20, G77, SEI, SIDS
    ),
    ~ as.numeric(as.character(.))
  ))

first_IV <- first_IV %>%
  add_row(
    country = "Channel Islands",
    iso3c = "CHI",
    region = "Europe",
    subregion = "Northern Europe",
    developed_economies = 1,
    developing_economies_excl_china = 0,
    developing_economies_excl_LDCs = 0,
    LLDCs = 0,
    LIDE = 0,
    MIDE = 0,
    HIDE = 1,
    BRICS = 0,
    EU = 0,
    G20 = 0,
    G77 = 0,
    income_group = "High income",
    SEI = 1,
```

```
    SIDS = 0
  )
```

5 Enregistrement IV framework

```
# Sauvegarde le fichier fusionné au format RDS
saveRDS(IV, file = "/Users/.../IV.rds")
```

```
## Chargement du fichier IV.rds
IV <- readRDS("/Users/.../IV.rds")
```

5.1 WDI

```
install.packages('WDI')
library("WDI")
```

```
# Définit les nouveaux indicateurs à télécharger
indicators2 <- c(
  # Performances économiques
  "NY.GDP.MKTP.KD",        # PIB (constant 2015 US$)
  "NY.GDP.PCAP.KD",        # PIB par habitant
  "GC.XPN.TOTL.GD.ZS",     # Dépenses publiques (% PIB)
  "SL.TLF.CACT.ZS",        # Taux de participation
  "NE.EXP.GNFS.KD",        # Exportations (val. constante)
  "NE.IMP.GNFS.KD",        # Importations (val. constante)
  "EG.ELC.ACCS.ZS",        # Accès à l'électricité
  "NV.AGR.TOTL.KD",        # VA - Agriculture
  "NV.IND.TOTL.KD",        # VA - Industrie
  "NV.SRV.TOTL.KD",        # VA - Services

  # Soutenabilité
  "NY.ADJ.SVNX.GN.ZS",     # Épargne nette ajustée
  "NY.GDP.PCAP.KD.ZG",     # Croissance du PIB/hab
  "SE.SEC.CUAT.UP.ZS",     # Éducation secondaire
  "SP.POP.0014.TO.ZS",     # Jeunes
  "SP.POP.65UP.TO.ZS",     # +65 ans
```

```
  "SP.POP.DPND",           # Ratio dépendance
  "SP.URB.TOTL.IN.ZS",     # Urbanisation
  "TX.VAL.FUEL.ZS.UN",     # Exportations combustibles
  "NY.GDP.TOTL.RT.ZS",     # Rentes naturelles

  # Nouveaux indicateurs capital
  "NE.GDI.TOTL.KD",        # Formation brute de capital fixe (constante 2015
   ↪ USD)
  "NE.GDI.TOTL.ZS"         # Formation brute de capital fixe (% PIB)
)
```

```
# Téléchargement des données pour tous les pays de 1990 à 2023
wdi_data3 <- WDI(country = "all", indicator = indicators2, start = 1990, end
 ↪ = 2023, extra = TRUE)
```

```
# 4. Nettoyage
wdi_data3 <- wdi_data3 %>%
  select(-region, -capital, -longitude, -latitude, -income, -lending, -iso2c,
   ↪ -lastupdated, -status) %>%
  mutate(
    country = as.factor(country),
    iso3c = as.factor(iso3c)
  ) %>%
  arrange(country, iso3c, year)

# On standardise les noms de colonnes
colnames(wdi_data3) <- tolower(trimws(colnames(wdi_data3)))
```

```
wdi_data3 <- wdi_data3 %>%
  rename(
    gdp_const = ny.gdp.mktp.kd,
    gdppc_const = ny.gdp.pcap.kd,
    gov_exp_pctgdp = gc.xpn.totl.gd.zs,
    lf_participation = sl.tlf.cact.zs,
    exports_const = ne.exp.gnfs.kd,
    imports_const = ne.imp.gnfs.kd,
    access_electricity = eg.elc.accs.zs,
    va_agriculture = nv.agr.totl.kd,
    va_industry = nv.ind.totl.kd,
    va_services = nv.srv.totl.kd,
    adj_savings = ny.adj.svnx.gn.zs,
```

```
    gdppc_growth = ny.gdp.pcap.kd.zg,
    educ_secondary_complete = se.sec.cuat.up.zs,
    pop_0_14_pct = sp.pop.0014.to.zs,
    pop_65plus_pct = sp.pop.65up.to.zs,
    age_dependency_ratio = sp.pop.dpnd,
    urban_pop_pct = sp.urb.totl.in.zs,
    fuel_exports_pct_exports = tx.val.fuel.zs.un,
    nat_resource_rents_pctgdp = ny.gdp.totl.rt.zs,
    gross_capital_const = ne.gdi.totl.kd,
    gross_capital_pctgdp = ne.gdi.totl.zs
  )
```

```
# Plusieurs années apparaissent plusieurs fois pour un même pays, on les
 ↪  agrège en une ligne unique
wdi_data3 <- wdi_data3 %>%
  group_by(country, iso3c, year) %>%
  summarise(across(everything(), ~ first(na.omit(.x))), .groups = "drop")
```

```
# Filtre les lignes où le code iso3c est manquant ou vide
countries_without_iso2 <- wdi_data3[is.na(wdi_data3$iso3c) | wdi_data3$iso3c
 ↪  == "", ]
# Affiche les pays sans code iso3c
unique_countries_without_iso2 <- unique(countries_without_iso2$country)
# Affiche la liste des pays
print(unique_countries_without_iso2)
```

```
# Liste des pays à supprimer
countries_to_remove2 <- c("High income", "Low income", "Lower middle income",
 ↪  "Not classified", "Upper middle income", "Not classified", "Lending
 ↪  category not classified", "Global Partnership for Education")
# Filtre le dataframe pour supprimer ces pays
wdi_data3 <- wdi_data3[!wdi_data3$country %in% countries_to_remove2, ]
```

```
# Identifie les colonnes des indicateurs WDI (exclure country, iso3c, year)
indicator_vars2 <- setdiff(names(wdi_data3), c("country", "iso3c", "year"))

# Créer une table avec pays et le nombre de valeurs non-NA
countries_empty2 <- wdi_data3 %>%
  group_by(country) %>%
  summarise(all_na = all(is.na(across(all_of(indicator_vars))))) %>%
```

```r
  filter(all_na) %>%
  pull(country)

# Affiche les pays sans aucune donnée sur aucune année
print(countries_empty2)
```

```r
wdi_data3 <- wdi_data3 %>%
  filter(!country %in% countries_empty2)
```

```r
sort(unique(wdi_data3$country))
```

```r
non_countries2 <- c(
  "Africa Eastern and Southern", "Africa Western and Central", "Arab World",
  "Caribbean small states", "Central Europe and the Baltics",
   ↪  "Early-demographic dividend",
  "East Asia & Pacific", "East Asia & Pacific (excluding high income)",
  "East Asia & Pacific (IDA & IBRD countries)", "Euro area", "Europe &
   ↪  Central Asia",
  "Europe & Central Asia (excluding high income)",
  "Europe & Central Asia (IDA & IBRD countries)", "European Union",
  "Fragile and conflict affected situations", "Global Partnership for
   ↪  Education",
  "Heavily indebted poor countries (HIPC)", "High income", "IBRD only",
  "IDA & IBRD total", "IDA blend", "IDA only", "IDA total",
  "Late-demographic dividend", "Latin America & Caribbean",
  "Latin America & Caribbean (excluding high income)",
  "Latin America & the Caribbean (IDA & IBRD countries)",
  "Least developed countries: UN classification", "Lending category not
   ↪  classified",
  "Low & middle income", "Low income", "Lower middle income",
  "Middle East & North Africa", "Middle East & North Africa (excluding high
   ↪  income)",
  "Middle East & North Africa (IDA & IBRD countries)", "Middle income",
  "North America", "Not classified", "OECD members", "Other small states",
  "Pacific island small states", "Post-demographic dividend",
  "Pre-demographic dividend", "Small states", "South Asia", "South Asia (IDA
   ↪  & IBRD)",
  "Sub-Saharan Africa", "Sub-Saharan Africa (excluding high income)",
  "Sub-Saharan Africa (IDA & IBRD countries)", "Upper middle income", "World"
)
```

```r
wdi_data3 <- wdi_data3 %>%
  filter(!country %in% non_countries2)
```

```r
sum(is.na(wdi_data3$country))  # Nombre de NA dans la colonne country
```

```r
sum(is.na(wdi_data3$iso3c))    # Nombre de NA dans la colonne iso3c
```

```r
## Vérifie les différences entre les deux dataframes pour détecter des
 ↪  incohérences
differences10 <- setdiff(wdi_data3$country, IV$country)
cat("Les pays dans wdi_data3 qui ne correspondent pas à IV:\n")
print(differences10)
```

```r
unique(IV$country)
```

```r
# Vecteur de correspondance des noms WDI -> IV
country_mapping2 <- c(
  "Bahamas, The" = "Bahamas",
  "Bolivia" = "Bolivia Plurinational State of",
  "British Virgin Islands" = "Virgin Islands (British)",
  "Congo, Dem. Rep." = "Congo Democratic Republic of the",
  "Congo, Rep." = "Congo",
  "Egypt, Arab Rep." = "Egypt",
  "Gambia, The" = "Gambia",
  "Hong Kong SAR, China" = "Hong Kong",
  "Iran, Islamic Rep." = "Iran Islamic Republic of",
  "Korea, Dem. People's Rep." = "Korea Democratic People's Republic of",
  "Korea, Rep." = "Korea Republic of",
  "Kyrgyz Republic" = "Kyrgyzstan",
  "Lao PDR" = "Lao People's Democratic Republic",
  "Macao SAR, China" = "Macao",
  "Micronesia, Fed. Sts." = "Micronesia Federated States of",
  "Moldova" = "Moldova Republic of",
  "Netherlands" = "Netherlands Kingdom of the",
  "Netherlands Antilles" = "Netherlands Antilles",
  "Slovak Republic" = "Slovakia",
  "St. Kitts and Nevis" = "Saint Kitts and Nevis",
  "St. Lucia" = "Saint Lucia",
  "St. Martin (French part)" = "Saint Martin (French part)",
  "St. Vincent and the Grenadines" = "Saint Vincent and the Grenadines",
```

```
  "Tanzania" = "Tanzania United Republic of",
  "Turkiye" = "Turkey",
  "United Kingdom" = "United Kingdom of Great Britain and Northern Ireland",
  "United States" = "United States of America",
  "Venezuela, RB" = "Venezuela Bolivarian Republic of",
  "Yemen, Rep." = "Yemen",
  "Channel Islands" = "Channel Islands",
  "Mayotte" = "Mayotte",
  "West Bank and Gaza" = "Palestine State of"
)
```

```
# Remplace les noms dans la colonne country selon la correspondance
wdi_data3 <- wdi_data3 %>%
  mutate(country = recode(country, !!!country_mapping2))
```

```
# Vérifie les différences entre les deux dataframes pour détecter des
 ↪  incohérences
differences_iso3c_2 <- setdiff(wdi_data3$iso3c, IV$iso3c)
cat("Les iso3c dans wdi_data3 qui ne correspondent pas à IV:\n")
print(differences_iso3c_2)
```

```
# Téléchargement
saveRDS(wdi_data3, "/Users/.../WDI.RDS")
write.csv(wdi_data3, "/Users/.../WDI.csv", row.names = FALSE)
```

5.2 Freedom in the World 2025

Nous téléchargeons un fichier excel intitulé "Country and Territory Ratings and Statuses, 1973-2024 (Excel Download)" au lien suivant : https://freedomhouse.org/report/freedom-world

Le capital institutionnel est représenté par l'indicateur de démocratie Freedom House's indicator of democracy, calculé comme la moyenne des "political rights" and "civil liberties."

Les deux indicateurs sont scorés entre 1 et 7. Nous téléchargeons le document d'abord au fichier .xlsx. Nous retenons le seuil onglet "Country Ratings, Statuses", que nous isolons dans un nouveau fichier excel.

Le fichier .xlsx est dans une forme à retravailler et à convertir au format "long".

```r
# Lire le fichier avec 2 lignes d'en-tête
df_freedom <- read_excel(
  path = "/Users/.../freedom.xlsx",
  sheet = "Country Ratings, Statuses ",
  col_names = FALSE
)

# Extraire les deux lignes d'en-tête
header1 <- as.character(df_freedom[1, ])
header2 <- as.character(df_freedom[2, ])

# Créer un vrai nom de pays
country_col <- "Country"

# Supprimer les 2 lignes d'en-tête
df_freedom <- df_freedom[-c(1,2), ]
colnames(df_freedom)[1] <- country_col

# Initialiser un tableau vide
long_data_freedom <- tibble()

# Parcourir les colonnes par groupes de 3 (PR / CL / Status)
for (i in seq(2, ncol(df_freedom), by = 3)) {
  pr_col <- i
  cl_col <- i + 1

  # Extraire l'année depuis la 2e ligne d'en-tête
  year_val <- header2[pr_col]
  if (is.na(year_val)) next  # si pas d'année on saute

  # Créer un petit bloc de données
  bloc <- tibble(
    Country = df_freedom[[country_col]],
    Year = year_val,
    PR = as.numeric(df_freedom[[pr_col]]),
    CL = as.numeric(df_freedom[[cl_col]])
  )

  # Empiler
  long_data_freedom <- bind_rows(long_data_freedom, bloc)
}

# Nettoyer : enlever NA pays
```

```r
long_data_freedom <- long_data_freedom %>% filter(!is.na(Country))
```

```r
# Ajoute une colonne "democracy" = moyenne de PR et CL
long_data_freedom <- long_data_freedom %>%
  mutate(
    democracy = rowMeans(across(c(PR, CL)), na.rm = TRUE)
  )

# Convertit tous les NaN en NA
long_data_freedom <- long_data_freedom %>%
  mutate(
    democracy = ifelse(is.nan(democracy), NA, democracy)
  )
```

```r
# Renomme "Country" en "country"
long_data_freedom <- long_data_freedom %>%
  rename(country = Country)
```

```r
sort(unique(long_data_freedom$country))
```

```r
sort(unique(IV$country))
```

Noms de pays non correspondants : Bolivia; Brunei; Congo (Brazzaville); Congo (Kinshasa); Czech Republic; Czechoslovakia; Germany, E.; Germany, W.; Iran; Laos; Micronesia; Moldova; Netherlands; North Korea; Russia; Serbia and Montenegro; South Korea; St. Kitts and Nevis; St. Lucia; St. Vincent and the Grenadines; Syria; Taiwan; Tanzania; The Gambia; USSR; United Kingdom; United States; Venezuela; Vietnam; Vietnam, N.; Vietnam, S.; Yemen, N.; Yemen, S.; Yugoslavia

Les pays suivants se répètent : Germany, E. ; Germany, W. ; Vietnam, N. ; Vietnam, S. ; Yemen, N. ; Yemen, S. Pour chacun :

- Germany : premières données en 1990, jusqu'alors était en NA. "Germany, E" et "Germany, W" deviennent "NA".
- Vietnam : premières données en 1976, jusqu'alors était en NA. "Vientman, N." et "Vietman, S." deviennent "NA".
- Yemen : premières données en 1990, jusqu'alors était en NA. "Yemen, N." et "Yemen, S." deviennent "NA".

Vu que dans nos analyses on ne va pas s'intéresser aux années avant 1990, on peut supprimer ces quatre répétitions : Germany, E. ; Germany, W. ; Vietnam, N. ; Vietnam, S. ; Yemen, N. ; Yemen, S.

```
# Supprimer les anciennes entités à ne pas conserver
long_data_freedom <- long_data_freedom %>%
  filter(!country %in% c(
    "Germany, E.",
    "Germany, W.",
    "Vietnam, N.",
    "Vietnam, S.",
    "Yemen, N.",
    "Yemen, S."
  ))
```

Nous réduisons le jeu de données entre 1990 et 2024 :

```
# Filtrer les années entre 1990 et 2024
long_data_freedom <- long_data_freedom %>%
  filter(as.numeric(Year) >= 1990, as.numeric(Year) <= 2024)
```

Nous gérons maintenant les pays, car ils n'ont pas le même statut dans le temps.

- pays doublon, "Czech Republic" et "Czechoslovakia". Les données pour "Czech Republic" apparaissent dès 1993, avant étaient NA. Les données de "Czechoslovakia" deviennent NA après 1993. "Slovakia" a des données qui apparaissent également dès 1993. On va donc supprimer "Czechoslovakia".

```
# Supprimer les anciennes entités à ne pas conserver
long_data_freedom <- long_data_freedom %>%
  filter(!country %in% c(
    "Czechoslovakia"
  ))
```

- "Serbia", "Serbia and Montenegro" et "Montenegro". Les données de "Serbia" apparaissent en 2006, les données de "Montenegro" apparaissent en 2006 également, et les données "Serbia and Montenegro" sont entre 2003 et 2005. Nous supprimons ces dernières :

```
# Supprimer les anciennes entités à ne pas conserver
long_data_freedom <- long_data_freedom %>%
  filter(!country %in% c(
    "Serbia and Montenegro"
  ))
```

- “USSR” et “Russia”. Ce dernier apparait avec des données en 1991, et inversement pour “USSR”. Nous supprimons “USSR”.

```
# Supprimer les anciennes entités à ne pas conserver
long_data_freedom <- long_data_freedom %>%
  filter(!country %in% c(
    "USSR"
  ))
```

```
sort(unique(long_data_freedom$country))
```

```
sort(unique(IV$country))
```

Noms à harmoniser : Bolivia ; Brunei ; Congo (Brazzaville) ; Congo (Kinshasa) ; Czech Republic ; Iran ; Laos ; Micronesia ; Moldova ; Netherlands ; North Korea ; Russia ; South Korea ; St. Kitts and Nevis ; St. Lucia ; St. Vincent and the Grenadines ; Syria ; Taiwan ; Tanzania ; The Gambia ; United Kingdom ; United States ; Venezuela ; Vietnam ; Yugoslavia.

```
long_data_freedom <- long_data_freedom %>%
  mutate(country = recode(country,
    "Bolivia" = "Bolivia Plurinational State of",
    "Brunei" = "Brunei Darussalam",
    "Congo (Brazzaville)" = "Congo",
    "Congo (Kinshasa)" = "Congo Democratic Republic of the",
    "Czech Republic" = "Czechia",
    "Iran" = "Iran Islamic Republic of",
    "Laos" = "Lao People's Democratic Republic",
    "Micronesia" = "Micronesia Federated States of",
    "Moldova" = "Moldova Republic of",
    "Netherlands" = "Netherlands Kingdom of the",
    "North Korea" = "Korea Democratic People's Republic of",
    "South Korea" = "Korea Republic of",
    "Russia" = "Russian Federation",
    "St. Kitts and Nevis" = "Saint Kitts and Nevis",
```

```
    "St. Lucia" = "Saint Lucia",
    "St. Vincent and the Grenadines" = "Saint Vincent and the Grenadines",
    "Syria" = "Syrian Arab Republic",
    "Tanzania" = "Tanzania United Republic of",
    "The Gambia" = "Gambia",
    "United Kingdom" = "United Kingdom of Great Britain and Northern
     ↪  Ireland",
    "United States" = "United States of America",
    "Venezuela" = "Venezuela Bolivarian Republic of",
    "Vietnam" = "Viet Nam"
    # "Taiwan" et "Yugoslavia" sont laissés tels quels volontairement
  ))
```

“Taiwan” et “Yugoslavia” ne sont pas présents dans wdi_data2, inutile de les garder ici :

```
# Supprimer les anciennes entités à ne pas conserver
long_data_freedom <- long_data_freedom %>%
  filter(!country %in% c(
    "Taiwan",
    "Yugoslavia"
  ))
```

Maintenant on souhaite ajouter une colonne de codes ISO en 3 lettres, afin de faciliter les fusions et gestions futures :

```
# Joindre les codes ISO3 à partir de IV
long_data_freedom <- long_data_freedom %>%
  left_join(IV %>% select(country, iso3c), by = "country")
```

```
any(is.na(long_data_freedom$iso3c))
```

On renomme “Year” en “year” :

```
long_data_freedom <- long_data_freedom %>%
  rename(year = Year)
```

On veut réorganiser en fonction des pays et années :

```
long_data_freedom <- long_data_freedom %>%
  arrange(country)
```

On réorganise l'ordre des colonnes :

```
long_data_freedom <- long_data_freedom %>%
  select(country, iso3c, year, democracy, PR, CL)
```

```
# Télécharge
write_csv(long_data_freedom, "/Users/.../freedom.csv")
saveRDS(long_data_freedom, "/Users/.../freedom.rds")
```

5.3 PWT

Nous téléchargeons d'abord le fichier Excel de pwt10.01 sur le site de *Groningen Growth and Development Center* : https://www.rug.nl/ggdc/productivity/pwt/

Le jeu de données a été conçu par Feenstra, Robert C., Robert Inklaar and Marcel P. Timmer (2015), "The Next Generation of the Penn World Table" *American Economic Review*, 105(10), 3150-3182.

Nous allons trier les variables qui nous intéressent (à savoir "hc" et "rnna"), les enregistrer au format .rds et les fusionner avec le jeu de données.

```
library(readxl)
pwt2 <- read_excel("/Users/.../pwt1001.xlsx", sheet = "Data")
```

```
library(dplyr)

pwt2_sub <- pwt2 %>%
  select(countrycode, year, hc, rnna) %>%
  rename(iso3c = countrycode) %>%
  mutate(year = as.integer(year)) %>%
  filter(year >= 1990 & year <= 2019) %>%
  arrange(iso3c, year)
```

```
# On test les bons codes iso3c pour s'assurer qu'on perd pas de pays en cours
 ↪  de route
IV <- readRDS("/Users/.../IV.rds")
```

```r
differences <- setdiff(pwt2_sub$iso3c, sustainability$iso3c)
cat("Les codes ISO3C présents dans pwt2_sub mais pas dans sustainability
 ↪  :\n")
print(differences)
```

Trois codes présents dans PWT mais pas dans sustainability qui désignent Taïwan, Montserrat et Anguilla. Ils ne sont pas présents dans les données WDI donc nous ne les retenons pas.

```r
# On ajoute les variables hc et rnna à sustainability
library(dplyr)

sustainability <- sustainability %>%
  select(-hc, -rkna) %>%  # Supprime les anciennes variables hc et rkna
  left_join(
    pwt2_sub %>% select(iso3c, year, hc, rnna),
    by = c("iso3c", "year")
  )
```

5.4 ND-GAIN vulnerability score

On télécharge tout ND-GAIN au lien suivant (ND-GAIN, 2025) : https://gain.nd.edu/our-work/country-index/download-data/

Concerne plus de 180 pays des Nations Unies et sur plus de 20 ans. Les données sont fournies sous forme de fichiers CSV séparés dans un seul fichier compressé.

On va dans le dossier “resources” puis “vulnerability” puis “vulnerability.csv”. Les variables: ISO3, Name, puis des colonnes allant de 1995 à 2022.

On va importer ici un fichier au format .csv : “vulnerability.csv”.

```r
vulnerability <- read_csv2("/Users/.../vulnerability.csv")
```

```r
# Renomme les colonnes
vulnerability <- vulnerability %>%
  rename(
    iso3c = ISO3,
    country = Name
  )
```

```r
# Transforme au format long
vulnerability <- vulnerability %>%
  pivot_longer(
    cols = matches("^\\d{4}$"),      # toutes les colonnes qui sont des
     ↪  années (ex: 1995, 1996, ...)
    names_to = "year",
    values_to = "vulnerability"
  ) %>%
  mutate(year = as.integer(year))    # pour s'assurer que year est bien
   ↪  numérique
```

```r
any(is.na(vulnerability$country))
any(is.na(vulnerability$iso3c))
any(is.na(vulnerability$year))
```

```r
# Vérifie les différences entre les deux dataframes (vulnerability et IV)
 ↪  pour détecter des incohérences
differences2 <- setdiff(vulnerability$country, IV$country)
cat("Les pays dans vulnerability qui ne correspondent pas à IV:\n")
print(differences2)
```

```r
sort(unique(IV$country))
```

```r
vulnerability <- vulnerability %>%
  mutate(country = recode(country,
    "Bolivia, Plurinational State of"     = "Bolivia Plurinational State of",
    "Cape Verde"                          = "Cabo Verde",
    "Congo, the Democratic Republic o"    = "Congo Democratic Republic of
     ↪  the",
    "Czech Republic"                      = "Czechia",
    "Iran, Islamic Republic of"           = "Iran Islamic Republic of",
    "Korea, Democratic People's Repub"    = "Korea Democratic People's
     ↪  Republic of",
    "Korea, Republic of"                  = "Korea Republic of",
    "Libyan Arab Jamahiriya"              = "Libya",
    "Macedonia"                           = "North Macedonia",
    "Micronesia, Federated States of"     = "Micronesia Federated States of",
    "Moldova, Republic of"                = "Moldova Republic of",
    "Netherlands"                         = "Netherlands Kingdom of the",
    "Swaziland"                           = "Eswatini",
```

```
    "Tanzania, United Republic of"        = "Tanzania United Republic of",
    "United Kingdom"                      = "United Kingdom of Great Britain
     ↪  and Northern Ireland",
    "United States"                       = "United States of America",
    "Venezuela, Bolivarian Republic o"    = "Venezuela Bolivarian Republic
     ↪  of"
  ))
```

```
# Vérifie les différences entre les deux dataframes (vulnerability et IV)
 ↪  pour détecter des incohérences
differences3 <- setdiff(vulnerability$iso3c, IV$iso3c)
cat("Les iso3c dans vulnerability qui ne correspondent pas à IV:\n")
print(differences3)
```

```
vulnerability %>%
  group_by(country) %>%
  summarise(all_na = all(is.na(vulnerability))) %>%
  filter(all_na)
```

```
vulnerability <- vulnerability %>%
  filter(!country %in% c(
    "Andorra",
    "Liechtenstein",
    "Monaco",
    "Saint Kitts and Nevis",
    "San Marino"
  ))
```

```
any(is.na(vulnerability$vulnerability))
```

```
# Chemins de sauvegarde
write_csv(vulnerability, "/Users/.../vulnerability.csv")     # Format CSV
saveRDS(vulnerability, "/Users/.../vulnerability.rds")       # Format RDS
```

5.5 Informal Economy Database

Part des emplois informels (% total)	EMP_NIFL_SEX_AGE_EDU_NB_A

La base de données sur l'économie informelle peut être téléchargée en format Excel. Le document à citer : Elgin, C., M. A. Kose, F. Ohnsorge, and S. Yu. 2021. "Understanding Informality." *CERP Discussion Paper 16497*, Centre for Economic Policy Research, London.

Fichier composé de 12 onglets, l'onglet "Read me" détaille tout. Selon Elgin et al. (2021), plusieurs études ont utilisé différentes méthodes pour la estimer la taille du secteur informel. Ici, pour la large quantité de pays et d'années couverts, nous retenons deux indicateurs :

- Le **modèle MIMIC (Multiple Indicators Multiple Causes)** est une approche économétrique fondée sur des équations structurelles permettant d'estimer la taille relative de l'économie informelle comme une variable latente. Il repose sur l'identification de plusieurs causes explicatives (comme la fiscalité, la réglementation ou la qualité institutionnelle) et d'indicateurs observables (comme la demande d'électricité ou la circulation monétaire) qui traduisent les effets de cette économie cachée. L'approche aboutit à un indice synthétique calibré pour obtenir une estimation de l'économie informelle en pourcentage du PIB. Cette méthode est appréciée pour sa cohérence entre pays et sa large couverture temporelle et géographique, mais elle souffre de certaines limites, notamment une faible sensibilité aux changements de court terme, une dépendance à des données de calibration externes et une sensibilité aux spécifications du modèle. Elle tend à capturer les tendances structurelles de l'informalité, plutôt que ses variations conjoncturelles.
- Le modèle **DGE (Dynamic General Equilibrium)**, quant à lui, s'appuie sur une modélisation théorique du comportement intertemporel des agents économiques, qui allouent leur travail entre secteurs formel et informel en fonction d'arbitrages liés à la productivité, aux incitations fiscales ou aux conditions de marché. Ce cadre dynamique permet de simuler les effets de chocs économiques et de politiques publiques sur la répartition du travail et la production entre les deux sphères. Le modèle, calibré sur la base de données macroéconomiques, fournit une estimation de l'économie informelle en part du PIB et permet une meilleure prise en compte des cycles économiques. Sa robustesse théorique et sa sensibilité temporelle en font un outil particulièrement utile pour les analyses dynamiques, bien qu'il repose sur des hypothèses fortes et sur la disponibilité de données fiables, ce qui peut en limiter l'usage dans certains contextes.

Nous séparons les deux onglets "DGE_p" et "MIMIC_p" et les importons.

```
dge <- read_excel("/Users/.../DGE.xlsx")
mimic <- read_excel("/Users/.../MIMIC.xlsx")
```

```
dge <- dge %>%
  rename(
    country = Economy,
    iso3c = Code
  )
```

```r
mimic <- mimic %>%
  rename(
    country = Economy,
    iso3c = Code
  )
```

```r
dge <- dge %>%
  pivot_longer(
    cols = matches("^\\d{4}$"),  # toutes les colonnes qui sont des années
     ↪  (ex : 1995, 2000, etc.)
    names_to = "year",
    values_to = "dge"
  ) %>%
  mutate(year = as.integer(year))
```

```r
mimic <- mimic %>%
  pivot_longer(
    cols = matches("^\\d{4}$"),
    names_to = "year",
    values_to = "mimic"
  ) %>%
  mutate(year = as.integer(year))
```

```r
# Vérifie les différences entre les deux dataframes (dge et IV) pour détecter
 ↪  des incohérences
differences4 <- setdiff(dge$country, IV$country)
cat("Les country dans dge qui ne correspondent pas à IV:\n")
print(differences4)
```

```r
sort(unique(IV$country))
```

```r
dge <- dge %>%
  mutate(country = recode(country,
    "Bahamas, The"                    = "Bahamas",
    "Bolivia"                         = "Bolivia Plurinational State of",
    "Congo, Dem. Rep."               = "Congo Democratic Republic of the",
    "Congo, Rep."                     = "Congo",
    "Czech Republic"                  = "Czechia",
    "Egypt, Arab Rep."               = "Egypt",
```

```
    "Gambia, The"                     = "Gambia",
    "Iran, Islamic Rep."             = "Iran Islamic Republic of",
    "Korea, Rep."                    = "Korea Republic of",
    "Kyrgyz Republic"                = "Kyrgyzstan",
    "Lao PDR"                         = "Lao People's Democratic Republic",
    "Moldova"                         = "Moldova Republic of",
    "Netherlands"                     = "Netherlands Kingdom of the",
    "Slovak Republic"                = "Slovakia",
    "St. Lucia"                       = "Saint Lucia",
    "St. Vincent and the Grenadines" = "Saint Vincent and the Grenadines",
    "Tanzania"                        = "Tanzania United Republic of",
    "Türkiye"                         = "Turkey",
    "United Kingdom"                 = "United Kingdom of Great Britain and
     ↪  Northern Ireland",
    "United States"                  = "United States of America",
    "Venezuela, RB"                  = "Venezuela Bolivarian Republic of",
    "Yemen, Rep."                    = "Yemen"
  ))
```

```
# Vérifie les différences entre les deux dataframes (mimic et IV) pour
 ↪  détecter des incohérences
differences5 <- setdiff(mimic$country, IV$country)
cat("Les country dans mimic qui ne correspondent pas à IV:\n")
print(differences5)
```

```
mimic <- mimic %>%
  mutate(country = recode(country,
    "Bahamas, The"         = "Bahamas",
    "Bolivia"              = "Bolivia Plurinational State of",
    "Congo, Dem. Rep."     = "Congo Democratic Republic of the",
    "Congo, Rep."          = "Congo",
    "Czech Republic"       = "Czechia",
    "Egypt, Arab Rep."     = "Egypt",
    "Gambia, The"          = "Gambia",
    "Iran, Islamic Rep."   = "Iran Islamic Republic of",
    "Korea, Rep."          = "Korea Republic of",
    "Kyrgyz Republic"      = "Kyrgyzstan",
    "Lao PDR"              = "Lao People's Democratic Republic",
    "Moldova"              = "Moldova Republic of",
    "Netherlands"          = "Netherlands Kingdom of the",
    "Slovak Republic"      = "Slovakia",
    "Tanzania"             = "Tanzania United Republic of",
```

```
    "Türkiye"              = "Turkey",
    "United Kingdom"       = "United Kingdom of Great Britain and Northern
     ↪  Ireland",
    "United States"        = "United States of America",
    "Venezuela, RB"        = "Venezuela Bolivarian Republic of",
    "Yemen, Rep."          = "Yemen"
  ))
```

```
# Vérifie les différences entre les deux dataframes (mimic et IV) pour
 ↪  détecter des incohérences
differences6 <- setdiff(mimic$iso3c, IV$iso3c)
cat("Les country dans mimic qui ne correspondent pas à IV:\n")
print(differences6)
```

```
# Vérifie les différences entre les deux dataframes (dge et IV) pour détecter
 ↪  des incohérences
differences7 <- setdiff(dge$iso3c, IV$iso3c)
cat("Les country dans dge qui ne correspondent pas à IV:\n")
print(differences7)
```

```
dge_mimic <- full_join(dge, mimic, by = c("country", "iso3c", "year"))
```

```
# Chemins de sauvegarde
write_csv(dge_mimic, "/Users/.../dge_mimic.csv")
saveRDS(dge_mimic, "/Users/.../dge_mimic.rds")
```

5.6 GINI

Issu du travail ci-après : Chrisendo D, Niva V, Hoffman R, Sayyar SM, Rocha J, Sandström V, Solt F, Kummu M. 2024. "Income inequality has increased for over two-thirds of the global population". Preprint. doi: https://doi.org/10.21203/rs.3.rs-5548291/v1

Déposés sur Zenodo, nous téléchargeons le fichier csv intitulé "tabulated_adm0_gini_disp.csv" qui propose des scores gini annuels, entre 1990 et 2021.

Directement sur excel nous :

1- Supprimons des cellules inutiles à la toute fin avec des "NA";

2- Supprimon lles colonnes "cntry_code" et "slope";

3- Renommons “iso3” en “iso3c” et “Country” en “country”.

Maintenant nous passons au format long :

```
gini <- read_csv2("/Users/.../gini.csv")
```

```
gini <- gini %>%
  pivot_longer(
    cols = matches("^\\d{4}$"),
    names_to = "year",
    values_to = "gini"
  ) %>%
  mutate(year = as.integer(year))
```

```
# Vérifie les différences entre les deux dataframes (gini et IV) pour
 ↪  détecter des incohérences
differences8 <- setdiff(gini$country, IV$country)
cat("Les country dans gini qui ne correspondent pas à IV:\n")
print(differences8)
```

```
gini <- gini %>%
  filter(!country %in% c("Anguilla", "Western Sahara", "Taiwan"))
```

```
gini <- gini %>%
  mutate(country = recode(country,
    "Bolivia"                                  = "Bolivia Plurinational
     ↪  State of",
    "Brunei"                                   = "Brunei Darussalam",
    "Ivory Coast"                              = "Cote d'Ivoire",
    "Congo, the Democratic Republic of the"    = "Congo Democratic Republic
     ↪  of the",
    "Cape Verde"                               = "Cabo Verde",
    "Czech Republic"                           = "Czechia",
    "Micronesia, Federated States of"          = "Micronesia Federated
     ↪  States of",
    "United Kingdom"                           = "United Kingdom of Great
     ↪  Britain and Northern Ireland",
    "Iran, Islamic Republic of"                = "Iran Islamic Republic
     ↪  of",
    "South Korea"                              = "Korea Republic of",
    "Moldova, Republic of"                     = "Moldova Republic of",
```

```
    "Macedonia, the former Yugoslav Republic of" = "North Macedonia",
    "Netherlands"                                = "Netherlands Kingdom of
     ↪  the",
    "Korea, Democratic People's Republic of"     = "Korea Democratic People's
     ↪  Republic of",
    "Palestinian Territory, Occupied"            = "Palestine State of",
    "Russia"                                     = "Russian Federation",
    "Swaziland"                                  = "Eswatini",
    "Tanzania, United Republic of"               = "Tanzania United Republic
     ↪  of",
    "United States"                              = "United States of
     ↪  America",
    "Venezuela"                                  = "Venezuela Bolivarian
     ↪  Republic of",
    "Vietnam"                                    = "Viet Nam"
  ))
```

```
# Vérifie les différences entre les deux dataframes (gini et IV) pour
 ↪  détecter des incohérences
differences9 <- setdiff(gini$iso3c, IV$iso3c)
cat("Les iso3c dans gini qui ne correspondent pas à IV:\n")
print(differences9)
```

```
gini <- gini %>%
  mutate(iso3c = ifelse(iso3c == "XKO", "XKX", iso3c))
```

```
# Téléchargement
write_csv(gini, "/Users/.../gini.csv")
saveRDS(gini, "/Users/.../gini.rds")
```

6 Fusion dataframes

Les dataframe que l'on fusionne : WDI.RDS, freedom.rds, pwt.rds, vulnerability.rds, dme_-mimic.rds, gini.rds.

Les variables clés sont "iso3c", "country" et "year".

Nous commencçons avec une première fusion des données en dehors des datasets SAGE. Nous fusionnerons ensuite ce dataset à chacun des 6 datasets de la base SAGE.

```r
wdi <- readRDS("/Users/.../WDI.RDS")
freedom <- readRDS("/Users/.../freedom.rds")
pwt <- readRDS("/Users/.../pwt.rds")
vulnerability <- readRDS("/Users/.../vulnerability.rds")
dge_mimic <- readRDS("/Users/.../dge_mimic.rds")
gini <- readRDS("/Users/.../gini.rds")
```

```r
# On s'assure que year est en 'integer' partout
wdi <- wdi %>% mutate(year = as.integer(year))
freedom <- freedom %>% mutate(year = as.integer(year))
pwt <- pwt %>% mutate(year = as.integer(year))
vulnerability <- vulnerability %>% mutate(year = as.integer(year))
dge_mimic <- dge_mimic %>% mutate(year = as.integer(year))
gini <- gini %>% mutate(year = as.integer(year))
```

Le dataset a des données jusqu'en 2024, mais pas les autres. Le dataset avec le plus de données "wdi" s'arrête en 2023, nous le prennons comme référence.

```r
sustainability <- wdi %>%
  full_join(freedom, by = c("iso3c", "country", "year")) %>%
  full_join(pwt, by = c("iso3c", "country", "year")) %>%
  full_join(vulnerability, by = c("iso3c", "country", "year")) %>%
  full_join(dge_mimic, by = c("iso3c", "country", "year")) %>%
  full_join(gini, by = c("iso3c", "country", "year"))
```

```r
sustainability <- sustainability %>%
  filter(year <= 2023)
```

6.1 Data cleaning de 'Sustainability'

6.1.1 Aperçu général

```r
glimpse(sustainability)
```

```r
summary(sustainability)
```

Les principales variables économiques telles que le PIB (gdp_const), le PIB par habitant (gdppc_const) ou encore les exportations et importations sont relativement bien renseignées, même si certaines comme les dépenses publiques en pourcentage du PIB (gov_exp_pctgdp) ou la formation brute de capital présentent une part importante de valeurs manquantes (plus de 50 % dans certains cas). À noter que la valeur ajoutée des services (va_services) est vide (0 constant), ce qui suggère une erreur ou une variable non exploitée. En revanche, les variables de valeur ajoutée agricole et industrielle semblent correctement renseignées.

Côté indicateurs sociaux et démographiques, les parts de population jeune (pop_0_14_pct) et âgée (pop_65plus_pct) sont bien remplies, contrairement au taux d'achèvement du secondaire (educ_secondary_complete), très incomplet (72 % de valeurs manquantes). L'accès à l'électricité est assez bien couvert, tout comme le ratio de dépendance démographique.

Les dimensions institutionnelles, issues du fichier Freedom House, sont présentes via les variables PR, CL et leur moyenne democracy, mais contiennent près de 1000 valeurs manquantes. Les données issues de Penn World Tables, comme le capital humain (hc) ou le stock de capital net (rkna), sont partiellement renseignées, avec notamment plus de 3000 valeurs manquantes pour le capital humain.

Enfin, plusieurs variables sont mal typées : gini et vulnerability sont au format caractère et devront être converties en numérique après traitement des séparateurs (par exemple, virgule à remplacer par point). Certains doublons sont également présents et devront être supprimés. Il faudra aussi évaluer la pertinence de conserver les variables les plus incomplètes ou les supprimer si leur utilité analytique est limitée.

```
sustainability %>% count(year, sort = TRUE)
```

Entre 1990 et 2023, chaque année contient 220 lignes, ce qui correspond au nombre de pays dans ta base (220 pays). Cela montre que la base est parfaitement équilibrée temporellement et géographiquement jusqu'en 2021. À partir de 2022 et 2023, il y a une seule observation manquante par an, avec 219 pays renseignés au lieu de 220.

```
sustainability %>%
  filter(year %in% c(2022, 2023)) %>%
  filter(is.na(country) | country == "") %>%
  select(iso3c, year, everything())
```

```
sustainability %>%
  filter(year %in% c(2022, 2023)) %>%
  filter(is.na(iso3c) | iso3c == "")
```

On se rend compte ici que Mayotte et Netherlands Antilles ont des données uniquement pour une variable à chaque fois, nous pouvons donc les supprimer :

```
sustainability <- sustainability %>%
  filter(!country %in% c("Mayotte", "Netherlands Antilles"))
```

Ensuite, nous voyons que “gini” et “vulnerability” sont au format caractère (chr) à cause des virgules ou d’autres caractères. On les convertit proprement en numeric :

```
library(stringr)

sustainability <- sustainability %>%
  mutate(
    gini = as.numeric(str_replace(gini, ",", ".")),
    vulnerability = as.numeric(str_replace(vulnerability, ",", "."))
  )
```

```
summary(sustainability$gini)
summary(sustainability$vulnerability)
```

Maintenant nous allons identifier les colonnes trop incomplètes. Nous calculons le taux de valeurs manquantes par variables et visualisons celles à risques (nous prenons un seuil >80% de NA, et envisageons de ne pas retenir la variable qui dépasserait ce seuil) :

```
na_ratio <- sustainability %>%
  summarise(across(everything(), ~mean(is.na(.)))) %>%
  pivot_longer(cols = everything(), names_to = "variable", values_to =
   ↪  "na_ratio") %>%
  arrange(desc(na_ratio))

print(na_ratio)
```

Nous observons ici qu’aucune variable ne dépasse le seuil de 80%.

Nous nous assurons qu’il n’y a pas de doublons :

```
sustainability %>%
  count(country, iso3c, year) %>%
  filter(n > 1)
```

```
# Nous avons doublons avec "Saint Vincent and the Grenadines" donc nous
 ↪  supprimons doublons:
sustainability <- sustainability %>%
  distinct(country, iso3c, year, .keep_all = TRUE)
```

```
str(sustainability)
```

Nous vérifions les doublons dans les noms des pays :

```
sustainability %>%
  count(country) %>%
  arrange(desc(n))
```

```
library(naniar)

sustainability %>%
  miss_var_summary() %>%
  filter(pct_miss > 90)
```

```
library(ggplot2)

ggplot(sustainability, aes(x = gdp_const)) +
  geom_boxplot() +
  coord_cartesian(ylim = c(0, quantile(sustainability$gdp_const, 0.95, na.rm
   ↪  = TRUE)))
```

```
ggplot(sustainability, aes(x = dge)) +
  geom_boxplot() +
  coord_cartesian(ylim = c(0, quantile(sustainability$dge, 0.95, na.rm =
   ↪  TRUE)))
```

```
ggplot(sustainability, aes(x = vulnerability)) +
  geom_boxplot() +
  coord_cartesian(ylim = c(0, quantile(sustainability$vulnerability, 0.95,
   ↪  na.rm = TRUE)))
```

Nous ajoutons une variable “economic_openness” à partir des données d’importations et d’exportations :

Ouverture économique : (exportations + importations) / PIB

```
sustainability <- sustainability %>%
  mutate(
    open = (exports_const + imports_const) / gdp_const
  )
```

```r
sustainability <- sustainability %>%
  mutate(
    open_pct = 100 * (exports_const + imports_const) / gdp_const
  )
```

```r
IV <- readRDS("/Users/.../IV.rds")
```

Nous fusionnons enfin avec le dataset IV pour obtenir des classifications de pays :

```r
sustainability <- sustainability %>%
  left_join(IV, by = c("iso3c", "country"))
```

Nous vérifions enfin la couverture temporelle pour nous assurer qu'il n'y a pas d'anomalie :

```r
range(sustainability$year)
table(sustainability$year)
```

Nous pouvons voir que la période 1990 à 2921 est parfaitement complète avec 218 observations par an. En 2022 et 2023 nous observons 217 lignes. Nous inspectons quelles variables sont vides (NA) pour les pays en 2022 et 2023.

```r
sustainability %>%
  filter(year %in% c(2022, 2023)) %>%
  mutate(nb_na = rowSums(is.na(across(-c(country, iso3c, year))))) %>%
  arrange(desc(nb_na)) %>%
  select(country, iso3c, year, nb_na) %>%
  head(40)
```

Nous testons de savoir si un sous-panel équilibré est possible où toutes les variables critiques sont non-NA :

```r
vars_critiques <- c(
  "gdp_const",            # PIB total
  "gdppc_const",          # PIB/hab
  "gov_exp_pctgdp",       # Dépenses publiques (% PIB)
  "hc",                   # Capital humain (PWT)
  "rkna",                 # Stock de capital
  "democracy",            # Moyenne PR/CL (Freedom House)
  "vulnerability",        # ND-Gain
```

```
  "gini",                 # Inégalités
  "dge",                  # Informalité (DGE)
  "mimic"                 # Informalité (MIMIC)
)
```

```
sustainability <- sustainability %>%
  mutate(nb_na_critiques = rowSums(is.na(across(all_of(vars_critiques)))))
```

```
sustainability %>%
  filter(nb_na_critiques > 0) %>%
  count(year) %>%
  arrange(year)
```

```
# Sauvegarde
library(readr)

write_csv(sustainability, "/Users/.../sustainability.csv")
saveRDS(sustainability, "/Users/.../sustainability.rds")
```