Scenario 2 - MPX in the EU

Dawson Coblin and Jose Lazo

2022-11-07

Monkeypox (MPX) Infection Rates in the European Union

This report aims to prepare data to analyze and assess the impact of MPV within
each country of the EU, and understand how case rates may differ by region and other various demographic factors. These findings will aid the implementation and appropriate response to the epidemic to local health departments.

Legend of Tasks and Milestones

Milestone	Description of Section
Milestone 1	Group Agreement
Milestone 2	Data-set Imported into R; Identify Key Data Elements and Types
Milestone 3	Clean Data-set Create Descriptive Statistics and Data Dictionary

Milestone 3 Sub-sections:
1.Sub-setting Rows and Columns as Needed- Clean individual Data Frames
2.Summarizing Data and Creating New Vectors, Organize Vectors for Use
3.Clean Variables and Combined Data-sets
4.Descriptive Stats of EU Data and Data Dictionary

1) Milestone 1: Group Agreement

2) Milestone 2: Data-set Imported into R; Identify Key Data Elements and Types

```
library(ggplot2)
library(tidyverse)
```

```
## Warning in system("timedatectl", intern = TRUE): running command ’timedatectl’
## had status 1

## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --

## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
## v purrr   0.3.4

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

```
library(readr)
library(lubridate)
```

```
##
## Attaching package: ’lubridate’

## The following objects are masked from ’package:base’:
##
##     date, intersect, setdiff, union
```

```
library(stringr)
library(readxl)
```

```
euro_mpx_cases <- read_csv("files/euro_mpx_cases.csv")
```

```
## Rows: 2987 Columns: 5
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): CountryExp, CountryCode, Source
## dbl  (1): ConfCases
## date (1): DateRep
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
euro_census_stats <- read_csv("files/euro_census_stats.csv")
```

```
## Rows: 152534 Columns: 10
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (8): COUNTRY_CODE, SEX, AGE, CAS, EDU, FLAGS, FOOTNOTES, RES_POP
## dbl (2): TIME, pop
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
euro_pop_denominators <- read_csv("files/euro_pop_denominators.csv")
```

```
## Rows: 603 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (6): DATAFLOW, LAST UPDATE, freq, indic_de, geo, OBS_FLAG
## dbl (2): TIME_PERIOD, OBS_VALUE
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
world_country_regions <- read_csv("files/world_country_regions.csv")
```

```
## Rows: 249 Columns: 11
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (7): name, alpha-2, alpha-3, iso_3166-2, region, sub-region, intermediat...
## dbl (4): country-code, region-code, sub-region-code, intermediate-region-code
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

Milestone 3: 1. Sub-setting Rows and Columns as Needed- Clean individual Data Frames

Data Frame Cleaning

```
#Cleaning the cases df
euro_mpx_cases <- euro_mpx_cases %>%
  select( DateRep, CountryExp, CountryCode  ,ConfCases)


#Creating a vector to use for country code/country name and regions in EU
euro_country_vector <- unique(euro_mpx_cases$CountryCode)

#Cleaning the denominators df
euro_pop_denominators <- euro_pop_denominators %>%
  filter( TIME_PERIOD == 2022) %>%
  rename( CountryCode = geo , total_pop_2022 = 'OBS_VALUE') %>%
  select( CountryCode, total_pop_2022)

euro_pop_denominators <- euro_pop_denominators[euro_pop_denominators$CountryCode
                                               %in% euro_country_vector, ]


#Clean census data to include only country code, res_pop. The date for the census
#dataframe is 2011, so the date is removed and not relevant.

euro_census_stats <- euro_census_stats %>%
  group_by(COUNTRY_CODE, RES_POP) %>%
  summarise(census_pop = sum(pop)) %>%
  rename(CountryCode = COUNTRY_CODE)
```

```
## ‘summarise()‘ has grouped output by ’COUNTRY_CODE’. You can override using the
## ‘.groups‘ argument.
```

```
#Widen census info to allow each country row to have pop information.
euro_census_stats <-
  pivot_wider(euro_census_stats, names_from = RES_POP, values_from = census_pop)

euro_census_stats$`0-1000`[euro_census_stats$`0-1000` == 0] <- NA
euro_census_stats$`1000-9999`[euro_census_stats$`1000-9999` == 0] <- NA
euro_census_stats$`10000-99999`[euro_census_stats$`10000-99999` == 0] <- NA
euro_census_stats$`100000-199999`[euro_census_stats$`100000-199999` == 0] <- NA
euro_census_stats$`200000-499999`[euro_census_stats$`200000-499999` == 0] <- NA
euro_census_stats$`500000-999999`[euro_census_stats$`500000-999999` == 0] <- NA
euro_census_stats$GE1000000[euro_census_stats$GE1000000 == 0] <- NA
```

Milestone 3: 2. Summarizing Data and Creating New Vectors, Organize Vectors for Use

```
#We then assign counties with a region code in a new column a list "EU_region,"
#and include the country name and code
sr_northern_e<- c(
  "Denmark" , "Estonia", "Finland", "Iceland", "Ireland", "Latvia", "Lithuania",
  "Norway", "Sweden" , "DK" , "EE", "FI", "IE", "IS", "LT", "LU", "NO", "SE"
)
sr_western_e<- c(
  "Austria", "Belgium", "France", "Germany", "Luxembourg", "Netherlands" ,
  "DE", "AT" , "BE" , "FR", "LV", "NL"
)
sr_eastern_e<- c(
  "Bulgaria", "Czechia", "Hungary", "Poland", "Romania", "Slovakia", "BG", "CZ",
  "HU", "PL" , "RO", "SK"
)
sr_southern_e<- c(
  "Croatia", "Greece", "Italy", "Malta", "Portugal", "Slovenia", "Spain", "EL",
  "ES", "HR", "IT", "MT", "PT", "SI"
)
sr_western_a<- c(
  "Cyprus" , "CY"
)

#Create a floor month for the date range to aggregate by month
euro_mpx_cases <- euro_mpx_cases %>%
  mutate(floor_month = floor_date( DateRep , "month"))

#Review the aggregate totals as a value of confirmed cases in Europe and list as
#"cases_grouped_monthly"

euro_mpx_cases <- euro_mpx_cases %>%
  group_by(floor_month, CountryCode) %>%
  summarize(cases_grouped_monthly = sum(ConfCases))
```

```
## ‘summarise()‘ has grouped output by ’floor_month’. You can override using the
## ‘.groups‘ argument.
```

```
#We now add the EU_Region vector based on the country name.
euro_mpx_cases <- euro_mpx_cases %>%
  mutate(EU_region = case_when(
    CountryCode %in% sr_eastern_e ~ "Eastern Europe",
    CountryCode %in% sr_northern_e ~ "Northern Europe",
    CountryCode %in% sr_southern_e ~ "Southern Europe",
    CountryCode %in% sr_western_e ~ "Western Europe",
    CountryCode %in% sr_western_a ~ "Western Asia"
    ))
```

Milestone 3: 3. Clean Variables and Combined Data-sets

```
#Combined the denominations data for 2022 pop with euro_mpx_cases dataset

euro_mpx_cases <-  inner_join( euro_pop_denominators, euro_mpx_cases,
                                by= "CountryCode")
#Join census data
euro_mpx_cases <- inner_join(euro_census_stats, euro_mpx_cases,
                              by= "CountryCode")
#Rearrange column order
euro_mpx_cases <- relocate(euro_mpx_cases, floor_month, EU_region,
                          total_pop_2022 , cases_grouped_monthly,
                          .before = CountryCode)

euro_mpx_cases<- relocate(euro_mpx_cases, CountryCode, .before = total_pop_2022)
```

Milestone 3: 4. Descriptive Stats of EU Data and Data Dictionary

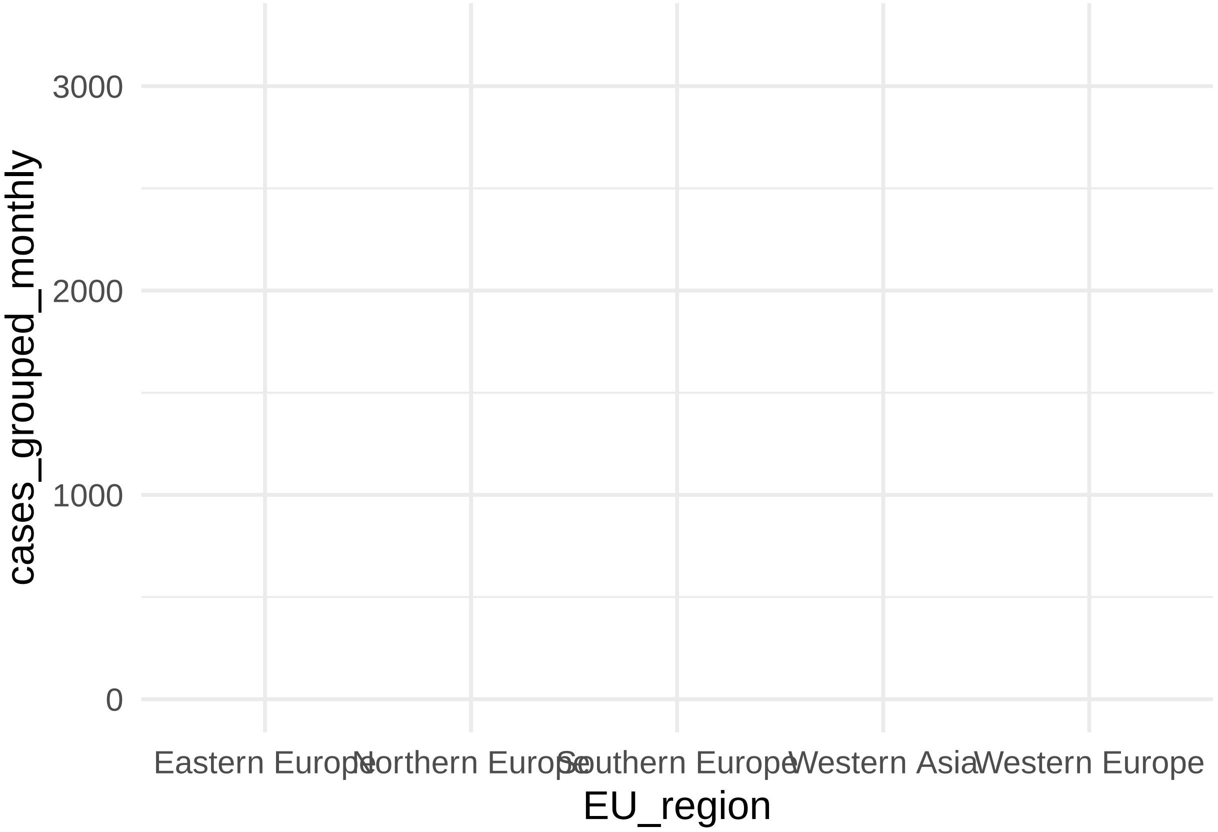
Descriptive stats: We find that the census information for 7 of the 29 countries in the EU are not available. These “0” values were replaced with “NA” to prevent graphing issues.

Region Data: We see that the regions in EU include various numbers of countries, thus having unequal populations.

“Eastern Europe” : 6 countries, total region population = 89171711
“Northern Europe”: 9 countries, total region population = 37518701
“Southern Europe”: 7 countries, total region population = 133879004
“Western Asia” : 1 countries, total region population = 904705
“Western Europe” : 6 countries, total region population = 190251495

```
visual <- ggplot(data = euro_mpx_cases,
                 aes(x = EU_region , y = cases_grouped_monthly))+
  theme_minimal(base_size = 15)

visual
```



```
#Upload the Data Dictionary for the central data-frame: "euro_mpx_cases"
#Data_Dictionary <- read_excel("~/PHW251_Fall2022/251/MS3_data_dictionary.xlsx")
#Data_Dictionary
```