SimOb Inject 11 - Descriptive analysis

## 1. Install packages and load libraries

# Load the required libraries into the current R session:  
pacman::p\_load(rio,   
 here,   
 tidyverse,   
 skimr,  
 plyr,  
 janitor,  
 lubridate,  
 gtsummary,   
 flextable,  
 officer,  
 epikit,   
 apyramid,   
 scales)  
  
# Load EpiStats from Github:  
# pacman::p\_load\_gh("Epiconcept-Paris/EpiStats")  
# pacman::p\_load\_gh("Epiconcept-Paris/EpiStats@release\_1.4-1\_2020-04-21")

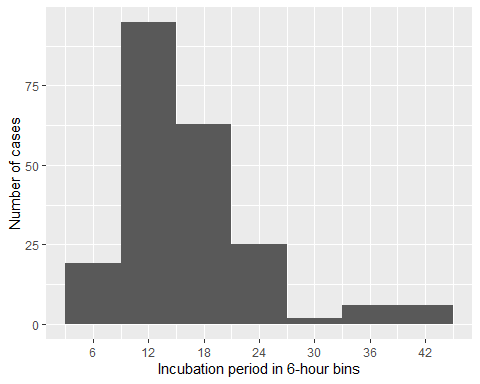
## 2. Import your data

# Import the raw data set:  
linelist <- rio::import(here::here("data", "Copenhagen\_clean\_inj10.xlsx"))%>%   
 # Convert character columns to factors:  
 mutate\_if(is.character,as.factor)  
  
# # Import the raw data set:  
# linelist <- rio::import(here::here("data", "Copenhagen\_clean\_inj10.xlsx"))%>%   
# # Convert character columns to factors:  
# mutate(across( c(sex, group, class,   
# tunaD, shrimpsD, greenD, vealD, pastaD, rocketD, sauceD,   
# breadD,champagneD, beerD, redwineD, whitewineD), ~as.factor(.)))

## 3. Time

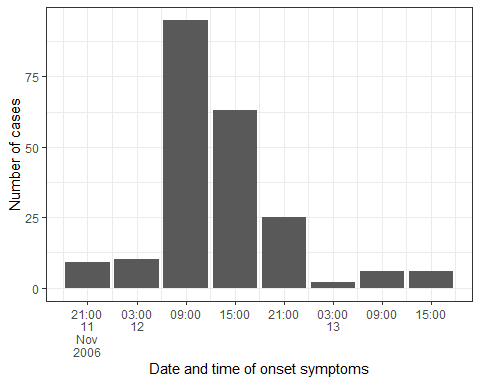
### a) Incubation period histogram

#| label: inc\_time  
  
incplot <- linelist %>%   
 #Keep cases only  
 filter(case == TRUE) %>%   
 # Create an empty ggplot frame:  
 ggplot() +  
 # Add a histogram of incubation:  
 geom\_histogram(aes(x = incubation),   
 # Set bin widths to 6 hours:  
 binwidth = 6) +  
 # Adapt scale to better fit data  
 scale\_x\_continuous(breaks = seq(0, 48, 6)) +   
 # Label x and y axes:  
 labs(x = "Incubation period in 6-hour bins",  
 y = "Number of cases")  
  
# Print plot:  
incplot

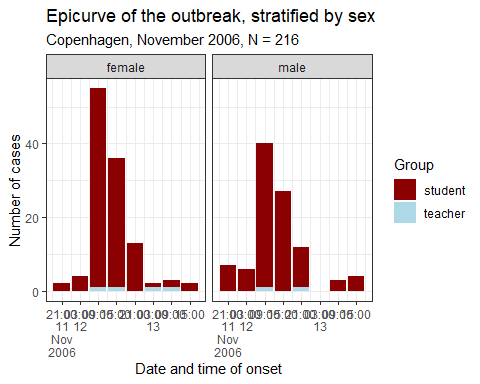


### b) Epicurve for date and time of onset

# Fetch data:  
epicurve\_datetime <- linelist %>%   
 # Filter for cases where dayonset is not missing:  
 filter(case == TRUE) %>% # & !is.na(dayonset)) %>%   
 # Add factor onset\_datetime to ggplot aesthetic:  
 ggplot(aes(x = onset\_datetime)) +   
 # Add geom\_bar:  
 geom\_bar() +  
 # Adapt scale to data and adjust axis label angle:  
 scale\_x\_datetime(  
 date\_breaks = "6 hours",  
 labels = label\_date\_short()) +  
 # Update x and y axis labels:  
 labs(x = "Date and time of onset symptoms",   
 y = "Number of cases") +  
 # Remove unnecessary grid lines:  
 theme\_bw()  
  
# Print epicurve:  
epicurve\_datetime



epicurve\_strata <- linelist %>%   
   
 # Filter for cases where dayonset is not missing:  
 filter(case == TRUE) %>%   
 # Add factor onset\_day to ggplot aesthetic:  
 ggplot(aes(x = onset\_datetime, fill = group)) +   
 # Add nicer fill colours:  
 scale\_fill\_manual(values = c("darkred", "lightblue")) +  
 # Add geom\_histogram:  
 geom\_bar() +  
 # Adjust x axis scales to a suitable unit:  
 scale\_x\_datetime(  
 date\_breaks = "6 hours",   
 labels = label\_date\_short()) +  
 # Update x and y axis labels:  
 labs(x = "Date and time of onset",   
 y = "Number of cases",   
 fill = "Group",   
 title = "Epicurve of the outbreak, stratified by sex",  
 subtitle = str\_glue("Copenhagen, November 2006, N = {sum(linelist$case)}")) +  
 # Stratify by sex:  
 facet\_wrap(facets = "sex",  
 ncol = 2) +  
 # Add theme:  
 theme\_bw()  
  
# Print epicurve:  
epicurve\_strata



## 4. Person

### a) Cross-tabulation of cases with group

linelist %>%   
   
 janitor::tabyl(case, group) %>%   
   
 adorn\_totals() %>%   
   
 adorn\_percentages() %>%   
   
 adorn\_pct\_formatting()

case student teacher  
 FALSE 94.4% 5.6%  
 TRUE 97.2% 2.8%  
 Total 96.0% 4.0%

### b) Cross-tabulation of cases with sex

linelist %>%   
   
 janitor::tabyl(case, sex) %>%   
   
 adorn\_totals() %>%   
   
 adorn\_percentages() %>%   
   
 adorn\_pct\_formatting()

case female male  
 FALSE 59.6% 40.4%  
 TRUE 54.2% 45.8%  
 Total 56.5% 43.5%

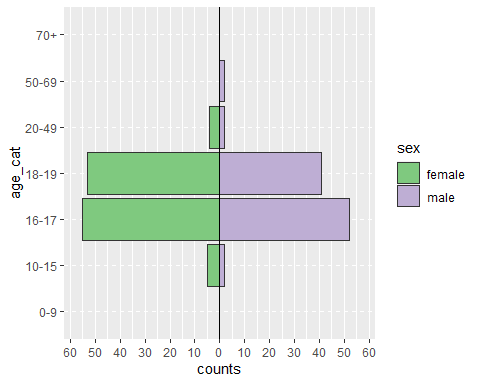
### c) Extra - Pyramid

Age-sex pyramid of cases

linelist <- linelist %>%   
 # Create age categories:  
 mutate(age\_cat = epikit::age\_categories(  
 # Name of age column:  
 x = age,   
 # Define the age categories:  
 breakers = c(0, 10, 16, 18, 20, 50, 70)  
 )  
 )  
  
  
# Check age categories:  
janitor::tabyl(linelist, age\_cat)

age\_cat n percent  
 0-9 0 0.00000000  
 10-15 11 0.02917772  
 16-17 201 0.53315650  
 18-19 147 0.38992042  
 20-49 11 0.02917772  
 50-69 7 0.01856764  
 70+ 0 0.00000000

# Pipe linelist:  
agesex <- linelist %>%   
 # Filter for cases only:  
 filter(case == TRUE) %>%   
 # Create age sex pyramid:  
 apyramid::age\_pyramid(  
 # Specify column containing age categories:  
 age\_group = "age\_cat",  
 # Specify column containing sex:  
 split\_by = "sex",   
 # Don't show midpoint on the graph:  
 show\_midpoint = FALSE  
 )  
  
# Print plot:  
agesex



(Hint: change show\_midpoint = FALSE to TRUE to see skewedness in the data patterns more easily).

## 5. Symptoms

1. Summary table of symptoms, stratified by case definition

# Create summary table:  
tabsymptoms <- linelist %>%   
 # Select person characteristics to summarise:  
 select(case, diarrhoea, bloody, vomiting,  
 abdo, nausea, fever,headache, jointpain) %>%   
 # transform clinical symptoms to factors, so NA can be accounted properly in the table  
 dplyr::mutate(  
 across(c(diarrhoea, bloody, vomiting,  
 abdo, nausea, fever,headache, jointpain),   
 ~as.factor(.))) %>%  
 # Make NA a explicit level of factor variables  
 dplyr::mutate(  
 across(c(diarrhoea, bloody, vomiting,  
 abdo, nausea, fever,headache, jointpain),   
 ~forcats::fct\_na\_value\_to\_level(.))) %>%   
 # Create the summary table:  
 gtsummary::tbl\_summary(  
 # Stratify by case:  
 by = case,   
 # Calculate row percentages:  
 percent = "column",  
 # Create nice labels:  
 label = list(  
 diarrhoea ~ "Diarrhoea",   
 bloody ~ "Dysentary",  
 vomiting ~ "Vomiting",  
 abdo ~ "Abdominal pain",  
 nausea ~ "Nausea",   
 fever ~ "Fever",   
 headache ~ "Headache",   
 jointpain ~ "Joint pain")  
   
 ) %>%   
 # Add totals:  
 add\_overall() %>%   
 # Make variable names bold and italics:  
 bold\_labels() %>%   
 italicize\_labels() %>%   
 # Modify header:  
 modify\_header(  
 label = "\*\*Characteristic\*\*",  
 stat\_0 = "\*\*Overall\*\*\n \*\*N\*\* = {N}",  
 stat\_1 = "\*\*Non-case\*\*\n \*\*N\*\* = {n}",  
 stat\_2 = "\*\*Case\*\*\n \*\*N\*\* = {n}",   
 ) %>%   
 # Convert to flextable for printing:  
 gtsummary::as\_flex\_table()  
  
# Print the table:  
tabsymptoms

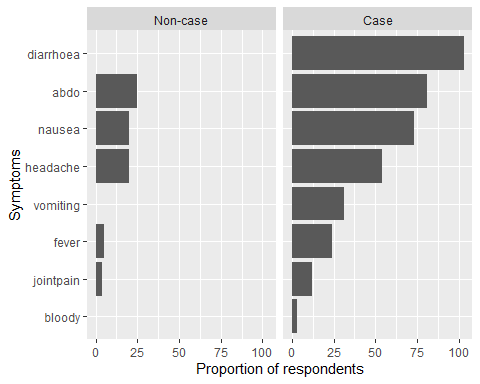
| **Characteristic** | **Overall** **N** = 3771 | **Non-case** **N** = 1611 | **Case** **N** = 2161 |
| --- | --- | --- | --- |
| ***Diarrhoea*** |  |  |  |
| FALSE | 46 (18%) | 40 (100%) | 6 (2.8%) |
| TRUE | 206 (82%) | 0 (0%) | 206 (97%) |
| ***Dysentary*** |  |  |  |
| FALSE | 189 (97%) | 42 (100%) | 147 (97%) |
| TRUE | 5 (2.6%) | 0 (0%) | 5 (3.3%) |
| ***Vomiting*** |  |  |  |
| FALSE | 149 (69%) | 42 (100%) | 107 (62%) |
| TRUE | 66 (31%) | 0 (0%) | 66 (38%) |
| ***Abdominal pain*** |  |  |  |
| FALSE | 35 (14%) | 6 (12%) | 29 (15%) |
| TRUE | 207 (86%) | 44 (88%) | 163 (85%) |
| ***Nausea*** |  |  |  |
| FALSE | 55 (25%) | 12 (26%) | 43 (24%) |
| TRUE | 169 (75%) | 34 (74%) | 135 (76%) |
| ***Fever*** |  |  |  |
| FALSE | 127 (74%) | 32 (80%) | 95 (73%) |
| TRUE | 44 (26%) | 8 (20%) | 36 (27%) |
| ***Headache*** |  |  |  |
| FALSE | 83 (38%) | 11 (25%) | 72 (41%) |
| TRUE | 137 (62%) | 33 (75%) | 104 (59%) |
| ***Joint pain*** |  |  |  |
| FALSE | 159 (85%) | 32 (84%) | 127 (85%) |
| TRUE | 29 (15%) | 6 (16%) | 23 (15%) |
| 1n (%) | | | |

1. Bar plot of symptoms stratified by case definition

# Create list of symptom variables:  
symptoms <- c("diarrhoea",   
 "bloody",   
 "vomiting",   
 "abdo",   
 "nausea",   
 "fever",   
 "headache",   
 "jointpain")  
  
# Create nice labels for case definition:  
caselabs <- ggplot2::as\_labeller(c(`FALSE` = "Non-case",   
 `TRUE` = "Case"))  
# Select variables and cases:  
symptom\_bar <- linelist %>%   
 # Select symptom columns:  
 select(case, c(all\_of(symptoms))) %>%  
 # Drop NAs:  
 drop\_na() %>%   
 # Reshape (pivot longer):  
 pivot\_longer(!case,   
 names\_to = "Symptoms",   
 values\_drop\_na = TRUE) %>%   
 # Keep only TRUE values:  
 filter(value == TRUE) %>%   
 # Group by symptoms and case:  
 group\_by(Symptoms, case) %>%   
 # Count for each symptom by case:  
 dplyr::summarise(count = n()) %>%   
 # Create plot:  
 ggplot(aes(  
 # Order symptom bars so most common ones are ontop:  
 x = reorder(Symptoms, desc(count), decreasing = TRUE),   
 y = count)) +  
 # Display bars as proportions  
 geom\_bar(stat = "identity") +  
 # Update x axis label:  
 xlab("Symptoms") +  
 # Update y axis label:  
 ylab("Proportion of respondents") +  
 # Flip plot on its side so symptom labels are clear:  
 coord\_flip() +  
 # Facet the plot by (labelled) case:  
 facet\_wrap(facets = "case",  
 labeller = caselabs,  
 ncol = 2)

`summarise()` has grouped output by 'Symptoms'. You can override using the  
`.groups` argument.

# Print plot:  
symptom\_bar



## 6. Attack proportions

1. Overall attack proportion

# Create table of case status:  
total\_ap <- tabyl(linelist, case) %>%   
 # Add row totals:  
 adorn\_totals(where = "row") %>%   
 # Add percentages with 1 digit after the decimal point:  
 adorn\_pct\_formatting(digits = 1) %>%   
 # Filter to rows where case is TRUE:  
 filter(case == TRUE) %>%   
 # Select the column percent:  
 select(percent) %>%   
 # Extract (pull) the value from this cell:  
 pull()  
  
# Print result:  
total\_ap

[1] "57.3%"

1. Attack proportions for class, group and sex by case status

# Table to calculate attack proportions:  
attack\_prop <- linelist %>%   
 # Select columns:  
 select (case, class, group, sex) %>%   
 # Create table:  
 tbl\_summary(  
 # Stratified by case  
 by = case,   
 # With row percentages  
 percent = "row") %>%  
 # Add totals:  
 add\_overall() %>%  
 # Add p values:  
 add\_p() %>%   
 # Make variable names bold and italics:  
 bold\_labels() %>%   
 italicize\_labels() %>%   
 # Modify header:  
 modify\_header(  
 label = "\*\*Characteristic\*\*",  
 stat\_0 = "\*\*Overall\*\* \*\*N\*\* = {N}",  
 stat\_1 = "\*\*Non-case\*\* \*\*N\*\* = {n}",  
 stat\_2 = "\*\*Case\*\* \*\*N\*\* = {n}",   
 p.value = "\*\*P value\*\*"  
 ) %>%   
 # Sort by p-value:  
 sort\_p() %>%   
 # Convert to flextable:  
 gtsummary::as\_flex\_table()  
  
# Print table:  
attack\_prop

| **Characteristic** | **Overall** **N** = 3771 | **Non-case** **N** = 1611 | **Case** **N** = 2161 | **P value**2 |
| --- | --- | --- | --- | --- |
| ***class*** |  |  |  | 0.090 |
| 1 | 131 (100%) | 63 (48%) | 68 (52%) |  |
| 2 | 101 (100%) | 44 (44%) | 57 (56%) |  |
| 3 | 111 (100%) | 38 (34%) | 73 (66%) |  |
| Unknown | 34 | 16 | 18 |  |
| ***group*** |  |  |  | 0.2 |
| student | 362 (100%) | 152 (42%) | 210 (58%) |  |
| teacher | 15 (100%) | 9 (60%) | 6 (40%) |  |
| ***sex*** |  |  |  | 0.3 |
| female | 213 (100%) | 96 (45%) | 117 (55%) |  |
| male | 164 (100%) | 65 (40%) | 99 (60%) |  |
| 1n (%) | | | | |
| 2Pearson's Chi-squared test | | | | |