Multisystem inflammatory syndrome in children related to COVID-19: a systematic review

Extended methods and data analysis

Levi Hoste, Ruben Van Paemel, Filomeen Haerynck

24 August, 2020, 14:30:45

Contents

Introduction

In this RMarkdown file, the extended methods and data-analysis for the manuscript "Multisystem inflammatory syndrome in children related to COVID-19: a systematic review" is described. The complete data-analysis can be reproduced from the data collection sheet (in .xls format), provided in the supplementary files of the manuscript or on Github. The study protocol was published on the PROSPERO systematic review register, prior to conducting the review: CRD42020189248

```
knitr::opts_chunk$set(cache = FALSE, warning = FALSE, message = FALSE)
options(digits = 3)
options(width = 60)
library(tidyverse)
require(readxl)
require(httr)
require(reshape2)
require(broom)
require(RColorBrewer)
require(scales)
require(ggrepel)
require(gridExtra)
require(ggExtra)
library(ggbeeswarm)
require(ggpubr)
library(cowplot)
library(naniar)
require(DT)
require(zoo)
require(psych)
library(skimr)
```

```r
library(UpSetR)
library(see)
library(wesanderson)
library(padr)

options(scipen=999)

co_hb <- 12
co_neutrophilia <- 8000
co_CRP <- 10
co_lympho <- 1250
co_fibrino <- 400
co_Ddim <- 250
co_ferritin <- 300
co_albu <- 34
co_PCT <- 0.49
co_LDH <- 280
co_IL6 <- 16.4
co_ESR <- 22
co_BNP <- 100
co_NTproBNP <- 400
co_tropo <- 40
co_WBC <- 11000
co_platelet <- 150000
co_sodium <- 135
#input = df_cohort_controls
#find = "max"
#param = "CRP"

collapse_labvals_cohort <- function(input, find, param, verbose = FALSE){
  if (find == "max"){
    df <- input %>% select(contains(param) | contains("cohort_id") | contains("cohort_type") | contains("tot_cases_n"))
    if (verbose == TRUE){
      print("Column extracted from cohorts:")
      print(colnames(df))
    }
    df_med <- df %>% select(contains("med"))
    df_med <- type_convert(df_med)
    df_med <- df_med %>% mutate_all(funs(replace_na(., -999)))
    # colnames(df_med)[max.col(df_med,ties.method="first")]
    df_med <- df_med %>% mutate(med = as.numeric(apply(df_med, 1, max)))

    df_min <- df %>% select(contains("Q1"))
    df_min <- type_convert(df_min)
    df_min <- df_min %>% mutate_all(funs(replace_na(., -999)))
    #colnames(df_min)[max.col(df_min,ties.method="first")]
    df_min <- df_min %>% mutate(min = as.numeric(apply(df_min, 1, max)))

    df_max <- df %>% select(contains("Q3"))
    df_max <- type_convert(df_max)
    df_max <- df_max %>% mutate_all(funs(replace_na(., -999)))
    #colnames(df_max)[max.col(df_max,ties.method="first")]
    df_max <- df_max %>% mutate(max = as.numeric(apply(df_max, 1, max)))

    df_full <- cbind(df %>% select(cohort_id, cohort_type, tot_cases_n), df_med %>% select(med), df_min %>% select(min), df_max %>% select(max))
    df_full[df_full == -999] <- NA
    names(df_full)[names(df_full) == 'max'] <- paste0(param, "_max")
    names(df_full)[names(df_full) == 'min'] <- paste0(param, "_min")
    names(df_full)[names(df_full) == 'med'] <- paste0(param, "_med")
    df_full$data_descr <- "IQR"
    df_full$cohort_id <- paste0(df_full$cohort_id, " (n = ", as.character(df_full$tot_cases_n),")")
    write.csv(df_full, paste0("./data/cohort_", param, ".csv"))
    print(datatable(df_full, caption = paste0("overview of ", param)))
    return(df_full)
  }
  else if (find == "min"){
    df <- input %>% select(contains(param) | contains("cohort_id") | contains("cohort_type") | contains("tot_cases_n"))
    if (verbose == TRUE){
      print("Column extracted from cohorts:")
      print(colnames(df))
    }
    df_med <- df %>% select(contains("med"))
    df_med <- type_convert(df_med)
    df_med <- df_med %>% mutate_all(funs(replace_na(., 1e6)))
    # colnames(df_med)[max.col(df_med,ties.method="first")]
    df_med <- df_med %>% mutate(med = as.numeric(apply(df_med, 1, min)))

    df_min <- df %>% select(contains("Q1"))
    df_min <- type_convert(df_min)
    df_min <- df_min %>% mutate_all(funs(replace_na(., 1e6)))
    #colnames(df_min)[max.col(df_min,ties.method="first")]
    df_min <- df_min %>% mutate(min = as.numeric(apply(df_min, 1, min)))

    df_max <- df %>% select(contains("Q3"))
    df_max <- type_convert(df_max)
    df_max <- df_max %>% mutate_all(funs(replace_na(., 1e6)))
    #colnames(df_max)[max.col(df_max,ties.method="first")]
    df_max <- df_max %>% mutate(max = as.numeric(apply(df_max, 1, min)))

    df_full <- cbind(df %>% select(cohort_id, cohort_type, tot_cases_n), df_med %>% select(med), df_min %>% select(min), df_max %>% select(max))
    df_full[df_full == 1e6] <- NA
    names(df_full)[names(df_full) == 'max'] <- paste0(param, "_max")
    names(df_full)[names(df_full) == 'min'] <- paste0(param, "_min")
    names(df_full)[names(df_full) == 'med'] <- paste0(param, "_med")
    df_full$data_descr <- "IQR"
    df_full$cohort_id <- paste0(df_full$cohort_id, " (n = ", as.character(df_full$tot_cases_n),")")
    write.csv(df_full, paste0("./data/cohort_", param, ".csv"))
    print(datatable(df_full, caption = paste0("overview of ", param)))
    return(df_full)
  }
}

#input = df_singlecases
#find = "max"
#param = "CRP"

collapse_labvals_single <- function(input, find, param, verbose = FALSE){
```

```
  if (find == "max"){
    df <- input %>% select(contains(param) | contains("cohort_id"))
    if (verbose == TRUE){
      print("Column extracted from single cases:")
      print(colnames(df))
    }
    df_coll <- df %>% mutate_all(funs(replace_na(., -999)))
    df_coll <- type_convert(df_coll)
    # colnames(df_med)[max.col(df_med,ties.method="first")]
    df_coll <- df_coll %>% mutate(max = as.numeric(apply(df_coll, 1, max)))

    df_coll[df_coll == -999] <- NA
    names(df_coll)[names(df_coll) == 'max'] <- paste0(param, "_max")
    df_coll$data_descr <- "IQR"
    df_coll$cohort_id <- paste0("single cases (n = ", as.character(n_single_cases),")")
    write.csv(skim(df_coll), paste0("./data/singlecases_", param, ".csv"))
    return(df_coll)
  }
  else if (find == "min"){
    df <- input %>% select(contains(param) | contains("cohort_id"))
    if (verbose == TRUE){
      print("Column extracted from single cases:")
      print(colnames(df))
    }
    df_coll <- df %>% mutate_all(funs(replace_na(., 1e6)))
    # colnames(df_med)[max.col(df_med,ties.method="first")]
    df_coll <- df_coll %>% mutate(min = as.numeric(apply(df_coll, 1, min)))

    df_coll[df_coll == 1e6] <- NA
    names(df_coll)[names(df_coll) == 'min'] <- paste0(param, "_min")
    df_coll$cohort_id <- paste0("single cases (n = ", as.character(n_single_cases),")")
    write.csv(skim(df_coll), paste0("./data/singlecases_", param, ".csv"))
    return(df_coll)
  }
}


moveme <- function (df, movecommand) {
  invec <- names(df)

  movecommand <- lapply(strsplit(strsplit(movecommand, ";")[[1]],
                                 ",|\\s+"), function(x) x[x != ""])
  movelist <- lapply(movecommand, function(x) {
    Where <- x[which(x %in% c("before", "after", "first",
                              "last")):length(x)]
    ToMove <- setdiff(x, Where)
    list(ToMove, Where)
  })
  myVec <- invec
  for (i in seq_along(movelist)) {
    temp <- setdiff(myVec, movelist[[i]][[1]])
    A <- movelist[[i]][[2]][1]
    if (A %in% c("before", "after")) {
      ba <- movelist[[i]][[2]][2]
      if (A == "before") {
        after <- match(ba, temp) - 1
      }
      else if (A == "after") {
        after <- match(ba, temp)
      }
    }
    else if (A == "first") {
      after <- 0
    }
    else if (A == "last") {
      after <- length(myVec)
    }
    myVec <- append(temp, values = movelist[[i]][[1]], after = after)
  }

  df[,match(myVec, names(df))]
}

makeBarplot <- function(var_id_cohort, var_id_single, var_id){

  n_cohort <- df_cohort %>% select(tot_cases_n) %>% sum()#, outcome_death_n)
  var_cohort <- df_cohort[var_id_cohort] %>% sum(., na.rm = TRUE)#, outcome_death_n)

  n_single <- df_singlecases %>% nrow()

  var_single <- df_singlecases %>% filter(get(var_id_single) == TRUE) %>% nrow()

  n_all <- n_cohort + n_single
  var_all <- var_cohort + var_single

  bar_df_abs <- data.frame(x = c("cohort", "cohort", "single cases", "single cases", "all", "all"), col = c("total", var_id, "total", var_id, "total", var_
        id), vals = c(n_cohort, var_cohort, n_single, var_single, n_all, var_all) )

  bar_df_prct <- data.frame(x = c("cohort", "cohort", "single cases", "single cases", "all", "all"), col = c(paste0(var_id, " -"), paste0(var_id, " +"),
        paste0(var_id, " -"), paste0(var_id, " +"), paste0(var_id, " -"), paste0(var_id, " +")), vals = c(100-(var_cohort/n_cohort*100), var_cohort/n_
        cohort*100, 100-(var_single/n_single*100), var_single/n_single*100, 100-(var_all/n_all*100), var_all/n_all*100) )


  p_abs <- ggplot(bar_df_abs, aes(x = x, y =  vals, fill = col)) +
    geom_bar(stat = "identity", position = "dodge") +
    theme_bw() +
    labs(title = paste0("Total cases vs ", var_id), subtitle = "Absolute numbers", x = "group", y = "n", col = "") +
    scale_fill_manual(values = wes_palette("Royal1")) +
    theme(legend.title = element_blank())


  p_prct <- ggplot(bar_df_prct, aes(x = x, y =  vals, fill = col)) +
    geom_bar(stat = "identity", position = "fill") +
    theme_bw() +
    labs(title = paste0(var_id), subtitle = "Percent", x = "group", y = "%", col = "")  +
    scale_y_continuous(labels = scales::percent)+
    scale_fill_manual(values = wes_palette("Royal1")) +
```

```
    theme(legend.title = element_blank())

  ggarrange(p_abs, p_prct, legend = "bottom")

}

makeHeatmap_cohort <- function(param1, colname_single, exclude_single = NULL, plottitle, textsize = 3){
  var_cohort <- df_cohort %>% select(("cohort_id") | "tot_cases_n" | ( contains(param1) & contains("_n")))
  var_cohort$cohort_id <- paste0(var_cohort$cohort_id, " (n = ", as.character(var_cohort$tot_cases_n),")")
  var_cohort <- var_cohort %>%
    gather(variable, value, 3:ncol(var_cohort)) %>% group_by(cohort_id, variable) %>% summarize(prct = value/tot_cases_n*100)
  var_cohort$variable <- sub("_n", "", var_cohort$variable)

  if (!is.null(exclude_single)){
    var_single <- df_singlecases %>% select(-contains(exclude_single))
    var_single <- var_single %>% select(contains(colname_single))
  } else
  {
    var_single <- df_singlecases %>% select(contains(colname_single))
  }

  #%>% select(-contains("any"))
  cols <- sapply(var_single, is.logical)
  var_single[,cols] <- lapply(var_single[,cols], as.numeric)
  var_single <- colSums(var_single, na.rm = TRUE)
  var_single <- var_single/nrow(df_singlecases)*100
  var_single <- as.data.frame(var_single) %>% rownames_to_column()
  var_single$cohort_id <- paste0("single cases (n = ", n_single_cases,")")
  colnames(var_single) <- c("variable", "prct", "cohort_id")


  missing <- setdiff(var_single$variable, var_cohort$variable)
  if (length(missing) != 0 ){
    missing_df <- data.frame(variable = missing, prct = NA, cohort_id = unique(var_cohort$cohort_id))
    var_cohort <- bind_rows(var_cohort, as_tibble(missing_df))
  } else if (length(missing) == 0) {
    missing <- setdiff(var_cohort$variable, var_single$variable)
    if (length(missing) != 0){
      missing_df <- data.frame(variable = missing, prct = NA, cohort_id = unique(var_single$cohort_id))
      var_single <- bind_rows(var_single, as_tibble(missing_df))
    }
  }

  hm_cohort <- ggplot(var_cohort, aes(x = variable, y = cohort_id, fill = prct)) +
    geom_tile() + theme_classic() +
    theme(axis.text.x=element_blank(), axis.ticks.x=element_blank(), axis.line=element_blank())+
    scale_fill_gradient(low = "yellow", high="red", na.value = "lightgray", limits = c(0,100)) +
    labs(x = "", y = "cohort", title =plottitle) +
    geom_text(aes(label=round(prct, 2)), size = textsize, color = "black")

  hm_single <- ggplot(var_single, aes(x = variable, y = cohort_id, fill = prct)) +
    geom_tile() +  theme_classic() +
    theme(axis.text.x=element_text(angle=90, hjust=1), axis.line=element_blank())+
    scale_fill_gradient(low = "yellow", high = "red", na.value = "lightgray", limits = c(0,100))+ labs(y = "cohort") +
    geom_text(aes(label=round(prct, 2)), size = textsize, color = "black")

  plot_grid(hm_cohort, hm_single, align = "v", nrow = 2, rel_heights = c(2/3, 1/3))
}
```

Search strategy

Electronic bibliographical databases were searched, both indexed (PubMed, Embase) and preprint repositories (BioRxiv and MedRxiv). Additionally, COVID-19-specific research repositories were be searched (Cochrane COVID-19 Study Register, the World Health Organization (WHO) COVID-19 Global Research Database). Publications in English language between 31 December 2019 up to 30 June 2020, when the final search was carried out, were reviewed on eligibility. Both finished and ongoing studies were considered. The reference lists of the included studies were considered as an additional source.

Search strategy focused on keywords involving the hyperinflammatory presentation (PIMS-TS, MIS-C, hyperinflammation, HLH, toxic shock syndrome, vasculitis, Kawasaki disease), as well as the association with COVID-19 (SARS-CoV-2, COVID-19, novel coronavirus) and the pediatric population (children, adolescent, pediatric). Structured hierarchic keywords (MeSH, Emtree) and wildcards were used when applicable. Boolean operators were used to combine the various keywords of interest. Below, the full search terms are presented for the different databases).

PubMed

```
"
(PIMS*" OR "MIS*" OR "multisysteminflammat*" OR "hyperinflammat*" OR "inflammatory "disease OR "systemicinflammat*" OR "cytokine "release OR "Kawasaki*" OR
      ""vasculitis OR "toxic "shockOR ""shock OR (" pediatricmultisystem inflammatory disease, COVID-19 related" [Supplementary Concept]) OR "
       MucocutaneousLymph Node Syndrome"[Mesh] OR "Shock"[Mesh] OR "Vasculitis"[Mesh] OR ""inflammation[MeSH]) AND"(covid*" or "sars-cov"-2 or "2019-"nCov or
       "novel "coronavirus or "coronavirus "disease or"COVID-19" [Supplementary Concept] OR"severe acute respiratory syndrome coronavirus 2" [Supplementary
       Concept]) AND"(child*" or "adolescen*" or "teen*" or "pediatric*" or ""infant or ""newborn or"Child"[Mesh ] OR"Adolescent"[Mesh] OR"Pediatrics"[Mesh ]
      or "Infant ,Newborn"[Mesh ] or"Infant"[Mesh]) AND ("2019/12/31"[Date - Publication] : "3000"[Date - Publication])
```

Embase

```
('pims*' OR 'mis' OR 'mis-' cOR ' multisysteminflammat*' OR 'hyperinflammat*' OR 'inflammatory disease' OR ' systemicinflammat*' OR 'cytokine release' OR '
      kawasaki*' OR 'vasculitis' OR 'toxic shock' OR 'shock') AND ('covid*' OR 'sars-cov-2' OR '2019-ncov' OR 'novel coronavirus' OR 'coronavirus disease')
       AND ('child*' OR 'adolescen*' OR 'teen*' OR 'pediatric*' OR 'infant' OR 'newborn') AND [31-12-2019]/ sd
```

BioRxiv and MedRxiv

Literature search in biorxiv and medrxiv was done with the R by downloading the data from the dedicated COVID-19 SARS-CoV-2 preprints page in json format, and can be found on Github.

Cochrane COVID-19 study register

```
(pims* OR mis OR "mis-c" OR "multisystem inflammat*" OR hyperinflammat* OR "inflammatory disease" OR "systemic inflammat*" OR "cytokine release" OR kawasaki
      * OR vasculitis OR "toxic shock" OR shock) AND (child* OR adolescen* OR teen* OR pediatric* OR infant OR newborn)
```

WHO COVID-19 Global literature on coronavirus disease

```
("pims*" OR "mis" OR "mis-c" OR " multisysteminflammat*" OR "hyperinflammat*" OR "inflammatory disease" OR " systemicinflammat*" OR "cytokine release" OR "
      kawasaki*" OR "vasculitis" OR "toxic shock" OR "shock") AND ("child*" OR "adolescen*" OR "teen*" OR "pediatric*" OR "infant" OR "newborn")
```

Study selection and risk of bias assessment

Original studies were included with following designs: RCT, observational studies, case-control studies, cross-sectional studies, case reports and case series.

Records eligible for inclusion should present clinical cases fulfilling the following 3 criteria:

Inclusion criteria

1. Study population: hyperinflammatory syndrome meeting the case definitions of PIMS-TS or MIS(-C) in children (0-19 years of age) with a temporal association with confirmed or probable COVID-19
2. Outcome: clinical, epidemiological and immunological descriptions, therapeutic management and clinical effect, and prognosis of individuals or cohorts of patients.
3. Types of study designs: RCT, observational studies, case-control studies, cross-sectional studies, case reports and case series

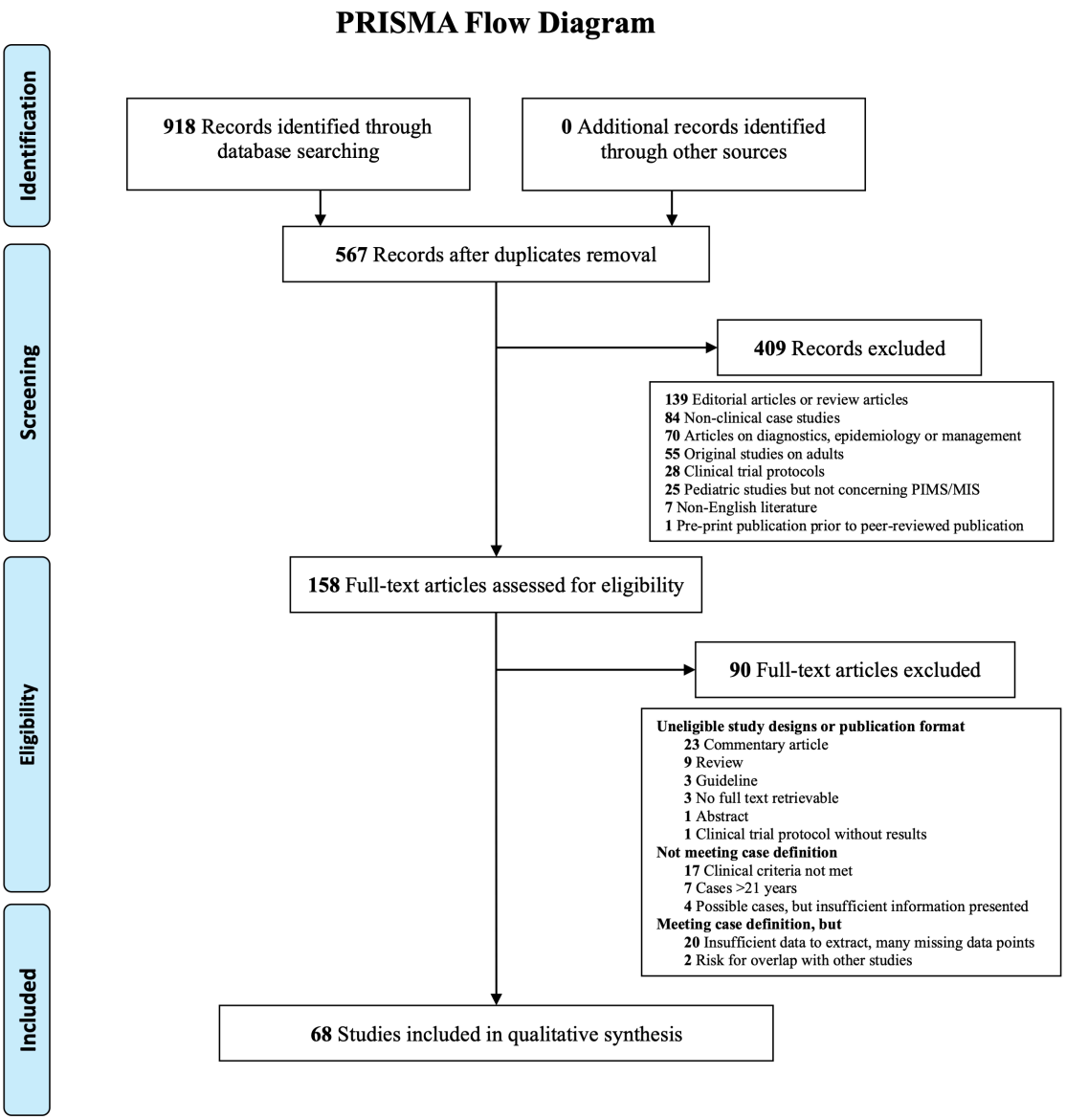
Exclusion criteria

1. Studies on adult patients with SARS-CoV-2 infection and/or SARS-CoV-2 associated hyperinflammatory syndromes
2. Studies on pediatric patients with other coronavirus infections (SARS-CoV-1 and Middle East Respiratory Syndrome Coronavirus (MERS-CoV) infection or other respiratory infections.
3. Studies with incomplete or lacking necessary data
4. Duplicate studies
5. Studies without accessible full text versions
6. Studies not in English language

Study selection was a two-stage process with, first, titles and abstracts of studies screened with retrieval using the search strategy and then, second, full text screening of potentially eligible studies assessed by two reviewers independently. Any disagreement over the eligibility of particular studies was resolved through discussion with a third reviewer.

The Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) checklist was used to guide the study selection and extraction process. Risk for bias on eligible observational studies were assessed by LH according to the NewCastle-Ottawa Scale (NOS), with full verification of all judgments by RVP. Level of evidence was rated according to Sackett.

PRISMA flow diagram



From: Moher D, Liberati A, Tetzlaff J, Altman DG, The PRISMA Group (2009). *P*referred *R*eporting *I*tems for *S*ystematic Reviews and *M*eta-*A*nalyses: The PRISMA Statement. PLoS Med 6(7): e1000097. doi:10.1371/journal.pmed1000097

Data extraction

All original studies describing clinical cases meeting the case definition of PIMS-TS, as defined by RCPCH, were eligible for inclusion. Primary outcome analysis focused on clinical, epidemiological and immunological manifestations, therapeutic management and prognosis.

The following data points were outlined to be extracted out of eligible records: patient characteristics (e.g. sex, age, ethnicity, anthropometry,…), comorbidities (e.g. cardiovascular disease, respiratory disease, diabetes mellitus, renal disease, malignancy/cancer, immunodeficiency,…), SARS-CoV-2 infection related data (e.g. close contacts with confirmed or suspected COVID-19, PCR and serology results,…), clinical symptoms (e.g. fever, respiratory, gastro-intestinal, neurological, dermatological, renal or cardiac manifestations, description of Kawasaki criteria,…), laboratory tests at various time points (e.g. haemoglobin, WBC, lymphocyte, neutrophil and platelet counts, sodium, ferritin, D-dimer, fibrinogen, albumin, creatinine, liver transaminases, CK, LDH, troponin, NT-proBNP, CRP, ESR, serum cytokines, complement, immunoglobulins,…), radiological results, hospital admission data (days of hospitalisation, days of ICU care,…), critical care interventions (invasive and non-invasive ventilation, inotropes/vasopressors use, ECMO,…) and therapeutics and their effect (corticosteroids, aspirin, IVIG, biotherapeutics, antibiotics,…). Fields with insufficient data will be excluded from final analysis. Additional parameters were included if relevant.

Cases were excluded if insufficient data suggesting a temporal association with SARS-CoV-2 was presented. A recent or current positive SARS-CoV-2 PCR (nasopharyngeal, fecal, other) or serology (IgA, IgM, IgG) results needed to be presented, or history of close contact (e.g. household) with a confirmed or highly suspect case of COVID-19 was required. Studies were excluded if data was inconsistently presented.

As a secondary outcome, it was outlined to make a qualitative assessment and propose an immunological mechanism underpinning this inflammatory syndrome based on the cases reported in literature and/or immunological investigations performed in these affected children. Nevertheless, up to the final search, insufficient data was available to conduct such assessments.

A single reviewer (LH) extracted data using a standardized form, while a second reviewer (RVP) cross checked all data for correctness and completeness. Any disagreement over study eligibility and conflict on data extraction were resolved by a third reviewer (FH).

Cohort studies and studies reporting on single cases were analysed separately, as we did not have access to the individual case characteristics of the cohort studies. For the cohort studies, proportions were calculated by summing only the studies which report on the variable, except for rare conditions such as death, comorbidities, use of ECMO or biopharmaceuticals.

Update between 2020-06-30 and 2020-08-13

Initially, the literature search and data-extraction was performed up to June 30, 2020. Afterwards, an update of the literature search, data-extraction and manuscript was done with studies published between July 1st and August 13th. In this second phase, conflicts during study selection were resolved by discussion between LH and RVP until a consensus was reached, instead of by the third independent third reviewer. For n = 7 studies, RVP extracted the data, while the second reviewer LH cross checked all data for correctness and completeness. No other changes to the literature seach methodology, data-extraction or analysis was done.

Data import and cleaning

Single cases

After data collection, we import the single cases from the general excel sheet and transform the excel sheet so that variables are columns and rows are cases. Columns without any values are also removed.

The single cases from Pouletty (10.1136/annrheumdis-2020-217960) are excluded (as they are included in the cohorts, and they only report IL6 data for single cases, which are added to the IL6 figure).

```
df_singlecases <-
  read_excel("/Users/rmvpaeme/Onedrive/UGent/PIMS-TS Systematic Review - General/Data extractie/data extractie.xlsx",
             sheet = "Single cases",
             skip = 1,
             col_names = FALSE)[,-c(1:2)]
df_singlecases <- df_singlecases %>% t()
df_singlecases <- as.data.frame(df_singlecases, stringsAsFactors = FALSE)
nms <- as.vector(df_singlecases[1,])
nms[is.na(nms)] <- 'tmp'
colnames(df_singlecases) <- make.unique(as.character(nms))
df_singlecases <- df_singlecases[-1,]
df_singlecases <- df_singlecases %>% select(-contains("tmp"))
df_singlecases <- df_singlecases %>% select(-variable_id)

df_singlecases <- df_singlecases %>%
  mutate_all(funs(str_replace(., "Yes", "yes")))
df_singlecases <- df_singlecases %>%
  mutate_all(funs(str_replace(., "No", "no")))
df_singlecases <- df_singlecases %>%
  mutate_all(funs(str_replace(., "pos", "yes")))
df_singlecases <- df_singlecases %>%
  mutate_all(funs(str_replace(., "neg", "no")))

df_singlecases <- df_singlecases %>%
  replace_with_na_all(condition = ~.x == "NA")

df_singlecases <- type_convert(df_singlecases)
df_singlecases_inclPouletty <- df_singlecases
df_singlecases <- df_singlecases %>% filter(doi != "https://10.1136/annrheumdis-2020-217960")  # these cases are excluded according to the data sheet

df_singlecases <- df_singlecases[colSums(!is.na(df_singlecases)) > 0]

df_singlecases$date_of_publication <- as.Date(df_singlecases$date_of_publication, origin = "1899-12-30")

n_single_cases <- nrow(df_singlecases)
```

Making summary statistics

In this section, data is summarized. For example, if there are any comorbidities present, a column "comorb_any" is added and annotated as TRUE. The same is done for COVID serology and symptoms of major organ (respiratory, cardiovascular etc).

```
df_singlecases <- df_singlecases %>% mutate(comorb_any = apply(df_singlecases %>% select(contains("comorb")), 1, any))
df_singlecases <- df_singlecases %>% moveme(., "comorb_any before comorb_cardiovasc")
```

If IgG, IgA, IgM or COVID serology is reported as positive, the column covid_sero_any is annotated as TRUE.

```
df_singlecases <- df_singlecases %>% mutate(covid_sero_any = apply(df_singlecases %>% select(covid_sero_pos, covid_IgA_pos, covid_IgM_pos, covid_IgG_pos),
      1, any))

df_singlecases <- df_singlecases %>% moveme(., "covid_sero_any before covid_sero_pos")
```

If PCR+, stool PCR+, IgG, IgA, IgM or COVID serology is reported as positive, the column covid_pos_any is annotated as TRUE.

```
df_singlecases <- df_singlecases %>% mutate(covid_pos_any = apply(df_singlecases %>% select(covid_PCR_pos, covid_PCR_stool_pos, covid_sero_pos, covid_IgA_
      pos, covid_IgM_pos, covid_IgG_pos), 1, any))

df_singlecases <- df_singlecases %>% moveme(., "covid_pos_any before covid_sero_any")
```

If any respiratory symptoms, symp_resp_any is annotated as TRUE.

```
df_singlecases <- df_singlecases %>% mutate(symp_resp_any = apply(df_singlecases %>% select(symp_resp_NS, symp_resp_URT, symp_resp_dyspnea, symp_resp_
      pneumonia, symp_resp_failure, symp_resp_chestpain), 1, any))

df_singlecases <- df_singlecases %>% moveme(., "symp_resp_any before symp_resp_NS")
```

If any GI symptoms, symp_GI_any is annotated as TRUE.

```
df_singlecases <- df_singlecases %>% mutate(symp_GI_any = apply(df_singlecases %>% select(contains("symp_GI")), 1, any))

df_singlecases <- df_singlecases %>% moveme(., "symp_GI_any before symp_GI_NS")
```

If any neurological symptoms, symp_neuro_any is annotated as TRUE.

```
df_singlecases <- df_singlecases %>% mutate(symp_neuro_any = apply(df_singlecases %>% select(symp_neuro_headache,symp_neuro_meningitis,symp_neuro_meningism,
      symp_neuro_asthenia,symp_neuro_irritab), 1, any))

df_singlecases <- df_singlecases %>% moveme(., "symp_neuro_any before symp_neuro_GCS")
```

If any renal symptoms, symp_renal_any is annotated as TRUE.

```
df_singlecases <- df_singlecases %>% mutate(symp_renal_any = apply(df_singlecases %>% select(symp_renal_AKI), 1, any))

df_singlecases <- df_singlecases %>% moveme(., "symp_renal_any before symp_renal_AKI")
```

If any cardiovascular symptoms, symp_cardiovasc_any is annotated as TRUE.

```
df_singlecases <- df_singlecases %>% mutate(symp_cardiovasc_any = apply(df_singlecases %>% select(symp_cardiovasc_myocard,
                                                                                                  symp_cardiovasc_pericard,
                                                                                                  symp_cardiovasc_cordilat,
                                                                                                  symp_cardiovasc_aneurysm,
                                                                                                  symp_cardiovasc_shock,
                                                                                                  symp_cardiovasc_tachycard,
                                                                                                  symp_cardiovasc_arrhyt), 1, any))

df_singlecases <- df_singlecases %>% moveme(., "symp_cardiovasc_any before symp_cardiovasc_myocard")

write.csv(df_singlecases, paste0("./data/df_singlecases.csv"))

#datatable(df_singlecases, caption = "Single cases dataframe")
```

Download single case data as .csv on Github

Cohorts

Afterwards, we do the same for the cohort sheet.

The papers by Grimaud et al. and Verdoni et al. are removed from the cohort dataframe, as most information is present in the single cases dataframe.

```
df_cohort <-
  read_excel("/Users/rmvpaeme/Onedrive/UGent/PIMS-TS Systematic Review - General/Data extractie/data extractie.xlsx",
             sheet = "Cohorts",
             skip = 1,
             col_names = FALSE)[,-c(1:3)]


df_cohort <- df_cohort %>% t()
df_cohort <- as.data.frame(df_cohort, stringsAsFactors = FALSE)
nms <- as.vector(df_cohort[1,])
nms[is.na(nms)] <- 'tmp'
colnames(df_cohort) <- make.unique(as.character(nms))
df_cohort <- df_cohort[-1,]
df_cohort <- df_cohort %>% select(-contains("tmp"))
df_cohort <- df_cohort %>% select(-variable_id)

df_cohort <- df_cohort %>%
  mutate_all(funs(str_replace(., "Yes", "yes")))
df_cohort <- df_cohort %>%
  mutate_all(funs(str_replace(., "No", "no")))
df_cohort <- df_cohort %>%
  mutate_all(funs(str_replace(., "pos", "yes")))
df_cohort <- df_cohort %>%
  mutate_all(funs(str_replace(., "neg", "no")))

df_cohort <- df_cohort %>%
  replace_with_na_all(condition = ~.x == "NA")

df_cohort <- type_convert(df_cohort)

df_cohort <- df_cohort[colSums(!is.na(df_cohort)) > 0]

df_cohort <- df_cohort %>% filter(doi != "https://doi.org/10.1186/s13613-020-00690-8") %>% filter(doi != "https://doi.org/10.1016/S0140-6736(20)31103-X")

df_cohort_controls <- df_cohort

df_cohort <- df_cohort %>% filter(cohort_type == "MIS-C")

df_cohort$date_of_publication <- as.Date(df_cohort$date_of_publication, origin = "1899-12-30")

write.csv(df_cohort, paste0("./data/df_cohort.csv"))

#datatable(df_cohort, caption = "Cohort dataframe")
```

Download cohort data as .csv on Github

Descriptive statistics

General

Click on the any of the tabs above to see descriptive statistics for every variable

Single cases

Download data as .csv on Github

```
#skim(df_singlecases)
write.csv(skim(df_singlecases), paste0("./data/singlecases_descriptivestats.csv"))
```

Cohorts

The "Prct_total" column is the percentage of e.g. death divided by the total cases in the cohort group. Only makes sense where n is reported e.g. therapy (not for lab values).

Download data as .csv on Github

```
skimsum <- skim_with(numeric = sfl(sum = ~ sum(., na.rm = TRUE), Prct_total = ~ sum(., na.rm = TRUE)/sum(df_cohort$tot_cases_n)*100), append = TRUE)
#skimsum(df_cohort)
write.csv(skimsum(df_cohort), paste0("./data/cohort_descriptivestats.csv"))
```

Historical controls

Download data as .csv on Github

```
df_cohort_controls_stats <- df_cohort_controls %>% filter(cohort_type != "MIS-C")
df_cohort_controls_stats <- df_cohort_controls_stats[colSums(!is.na(df_cohort_controls_stats)) > 0]
skimsum <- skim_with(numeric = sfl(sum = ~ sum(., na.rm = TRUE), Prct_total = ~ sum(., na.rm = TRUE)/sum(df_cohort_controls_stats$tot_cases_n)*100), append
       = TRUE)
#skimsum(df_cohort_controls_stats)

write.csv(skimsum(df_cohort_controls_stats), paste0("./data/historicalcontrols_descriptivestats.csv"))

write.csv(df_cohort_controls_stats, paste0("./data/df_cohort_controls_stats.csv"))
```

Data exploration

Important

For the cohorts, percentages describe the total (e.g. positive) cases, divided by the sum of the total cases **of studies reporting the variable**.

Cases in function of COVID pandemic

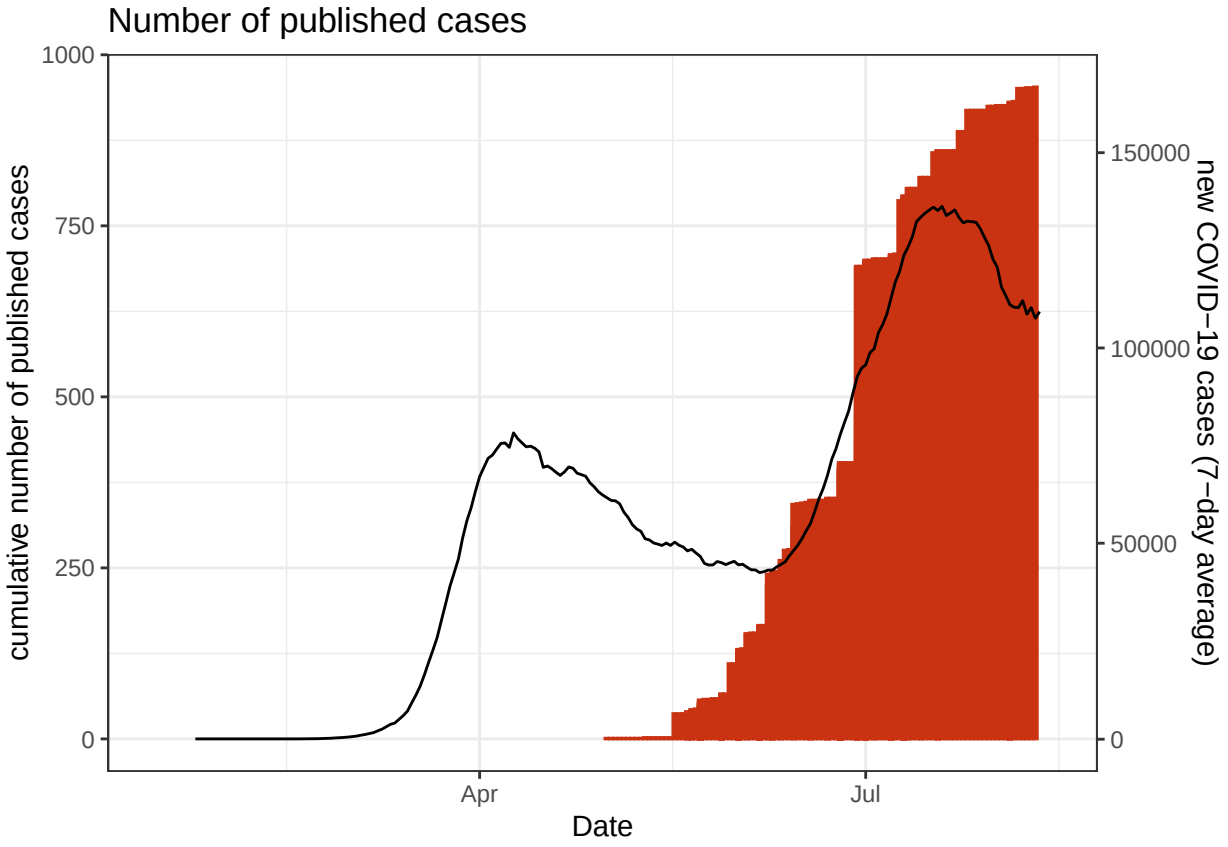
To investigate the relationship of the published PIMS cases with the ongoing COVID-19 pandemic, the case data from Johns Hopkins was downloaded (and added to this repository).

The list was filtered on the UK, US, Italy and France, as these country contribute the most to our dataset.

Caveat: this is a distored image of the PIMS cases: as the cases are published together, their true date of diagnosis is unknown.

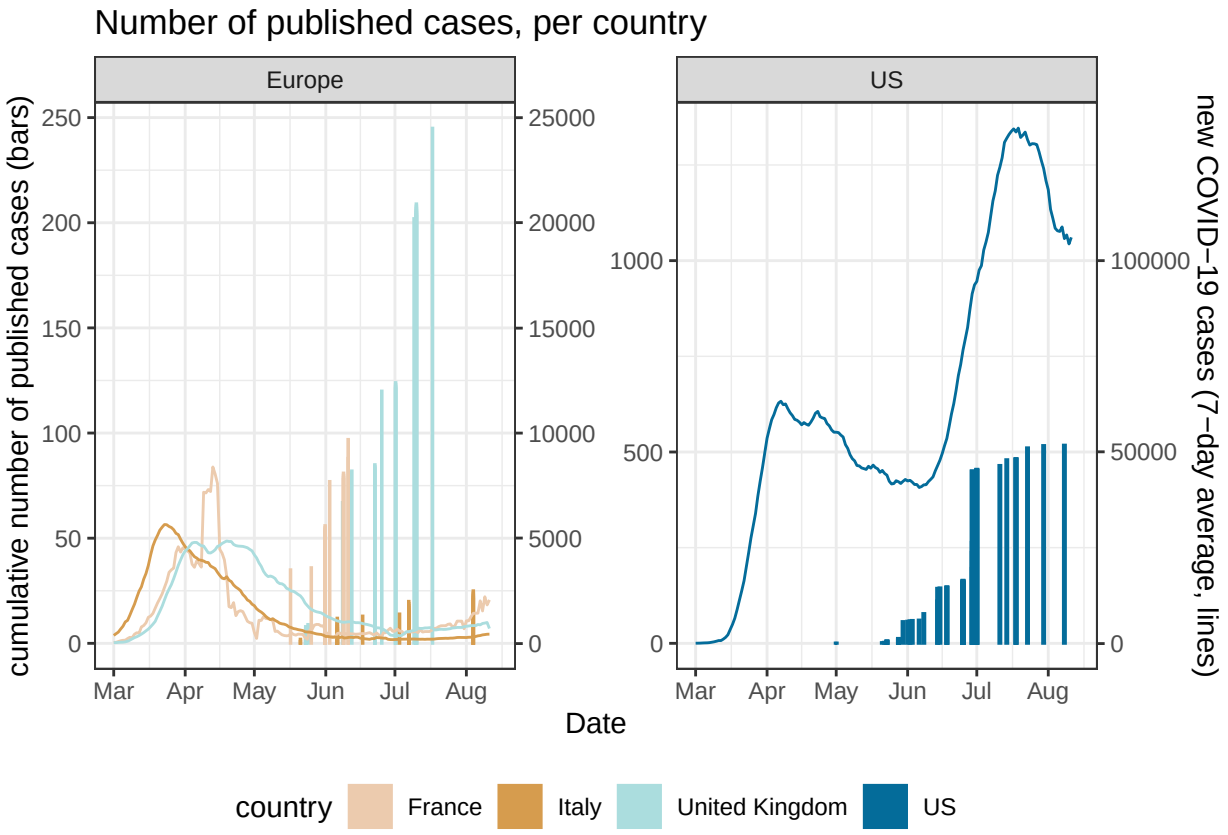
```
firstdiff <- function(x) {
  shifted <- c(0,x[1:(length(x)-1)])
  x-shifted
}

USA_cases <- read_csv("./data/time_series_covid19_confirmed_US.csv")
USA_cases <- USA_cases %>% select(-c(UID, iso2, iso3, code3, FIPS, Admin2, Province_State, Lat, Long_, Combined_Key))

names(USA_cases)[names(USA_cases) == 'Country_Region'] <- "Country/Region"

global_cases <- read_csv("./data/time_series_covid19_confirmed_global.csv")
global_cases <- global_cases %>% select(-c(`Province/State`, Lat, Long))

global_cases <- rbind(USA_cases, global_cases)

global_cases <- global_cases %>% melt()
global_cases$variable <- as.Date(global_cases$variable, format = "%m/%d/%y")
colnames(global_cases) <- c("country", "date_of_publication", "tot_cases_covid")
global_cases <- global_cases %>% filter(country == "United Kingdom"  | country == "Italy" | country == "France" | country == "US")
```

```r
all_glob_cases <- global_cases %>% group_by(date_of_publication) %>% summarise(total_cases = sum(tot_cases_covid))
all_glob_cases$newcases <- firstdiff(all_glob_cases$total_cases)
all_glob_cases$newcase_roll7 <- zoo::rollmean(all_glob_cases$newcases, k = 7, fill = NA)

evo_cases <- rbind(df_cohort %>% select(tot_cases_n, date_of_publication) %>% mutate(type = "cohort"),
                   df_singlecases %>% select(date_of_publication) %>% mutate(tot_cases_n = 1, type = "single"))

evo_cases <- pad(evo_cases)
evo_cases$tot_cases_n[is.na(evo_cases$tot_cases_n)] <- 0
evo_cases$cumplot <- cumsum(evo_cases$tot_cases_n)

full_data <- merge(evo_cases, all_glob_cases, all = TRUE)

p1 <- ggplot(full_data , aes(x = date_of_publication, y = cumplot)) +
  geom_col(position = "dodge", col = wes_palette("Royal1")[2], fill = wes_palette("Royal1")[2]) +
  theme_bw() + geom_line(aes(x = date_of_publication, y = newcase_roll7/175)) +
  labs(x = "Date", y = "cumulative number of cases", title = "Number of published cases") +  scale_x_date(limits = as.Date(c('2020-01-15','2020-08-14'))) +
        scale_y_continuous(
    "cumulative number of published cases",
    sec.axis = sec_axis(~ . * 175, name = "new COVID-19 cases (7-day average)")
  )

p1
```




```r
ggsave(p1, filename = "./plots/covid_evo_total.png", dpi = 300, height=7, width=10)
ggsave(p1, filename = "./plots/covid_evo_total.svg", dpi = 300, height=7, width=10)
ggsave(p1, filename = "./plots/covid_evo_total.pdf", dpi = 300, height=7, width=10)



all_glob_cases <- global_cases %>% group_by(date_of_publication, country) %>% summarise(total_cases = sum(tot_cases_covid)) %>% ungroup()


all_glob_cases <- all_glob_cases %>% group_by(country) %>%
  mutate(newcases = firstdiff(total_cases)) %>% ungroup()

#all_glob_cases$newcases <- firstdiff(all_glob_cases$total_cases)
all_glob_cases <- all_glob_cases %>% group_by(country)  %>% mutate(newcase_roll7 = zoo::rollmean(newcases, k = 7, fill = NA))

evo_cases <- rbind(df_cohort %>% select(tot_cases_n, date_of_publication, country) %>% mutate(type = "cohort"),
                   df_singlecases %>% select(date_of_publication, country) %>% mutate(tot_cases_n = 1, type = "single"))

country_barplot <- evo_cases

evo_cases <- pad(evo_cases)
#evo_cases$tot_cases_n[is.na(evo_cases$tot_cases_n)] <- 0
#evo_cases$tot_cases_n[is.na(evo_cases$tot_cases_n)] <- 0
evo_cases <- evo_cases %>% group_by(country) %>% mutate(cumplot = cumsum(tot_cases_n)) %>% ungroup()
evo_cases <- evo_cases %>% fill(country)

```

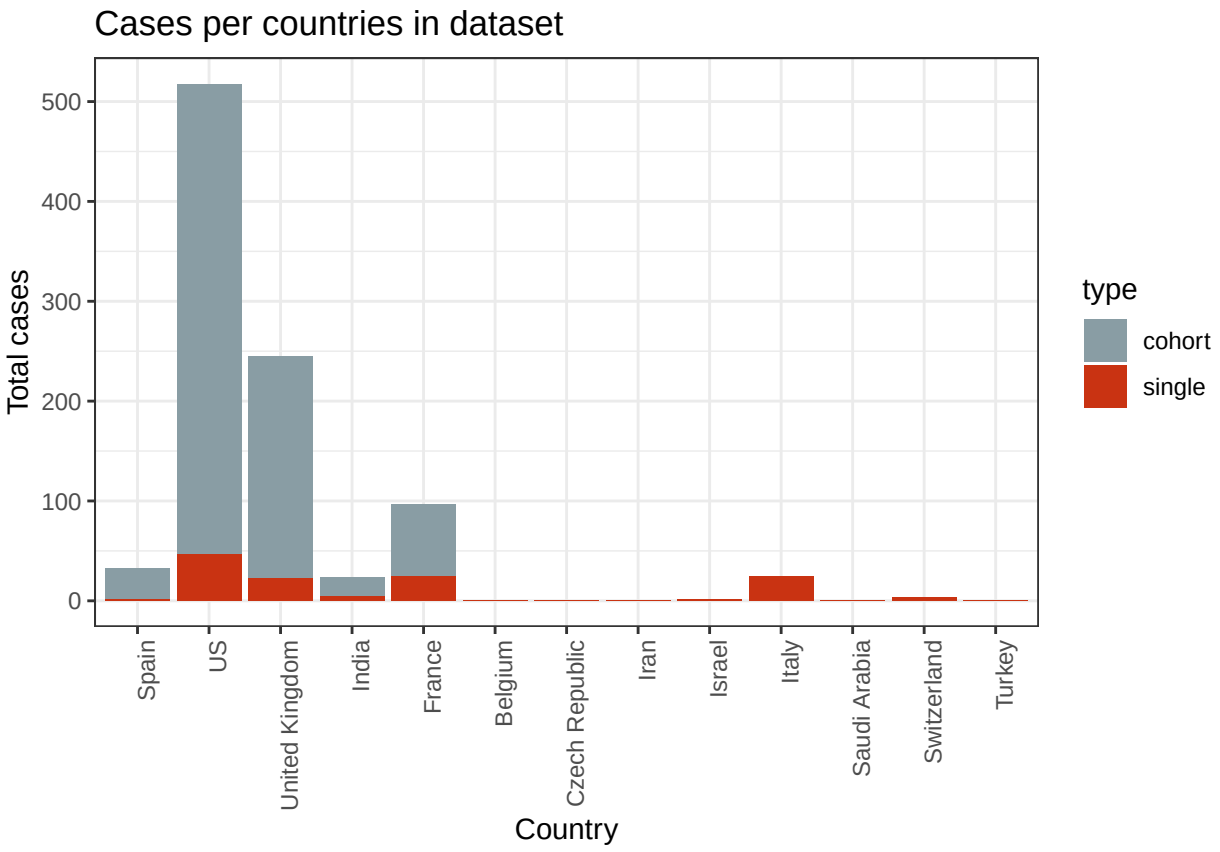
```r

full_data <- merge(evo_cases, all_glob_cases, all = TRUE)
full_data_filt <- full_data %>% filter(country == "United Kingdom"  | country == "Italy" | country == "France" | country == "US")

full_data_filt <- full_data_filt %>% mutate(continent = ifelse(country == "US", "US", "Europe"))

p1 <- ggplot(full_data_filt , aes(x = date_of_publication, y = cumplot)) +
  geom_col(position = "dodge", aes(fill = country, col = country)) +
  theme_bw() +
  scale_color_manual(values = wes_palette("Darjeeling2")[c(1,3,4,2)]) +
  scale_fill_manual(values = wes_palette("Darjeeling2")[c(1,3,4,2)]) +
  geom_line(aes(x = date_of_publication, y = newcase_roll7/100, col = country)) +
  labs(x = "Date", y = "cumulative number of cases (bars)", title = "Number of published cases, per country") +
  scale_x_date(limits = as.Date(c('2020-03-01','2020-08-14'))) + scale_y_continuous(    "cumulative number of published cases (bars)",
                                                                                        sec.axis = sec_axis(~ . * 100, name = "new COVID-19 cases (7-day
                                                                                              average, lines)")

  ) +
  theme(legend.position="bottom") +
  facet_wrap(~continent, scales = "free_y")

p1
```




```r
ggsave(p1, filename = "./plots/covid_evo_percountry.png", dpi = 300, height=7, width=10)
ggsave(p1, filename = "./plots/covid_evo_percountry.svg", dpi = 300, height=7, width=10)
ggsave(p1, filename = "./plots/covid_evo_percountry.pdf", dpi = 300, height=7, width=10)
```

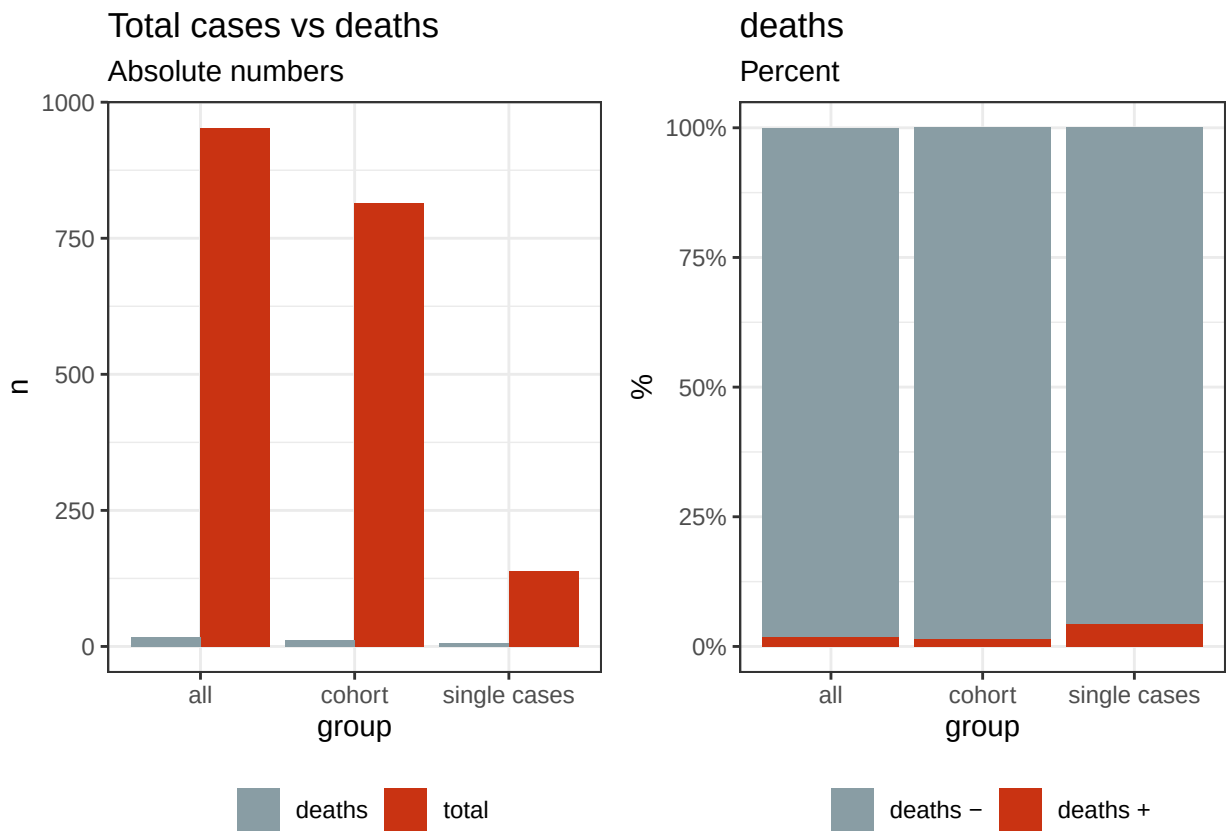
PIMS cases by country

```r
ggplot(country_barplot, aes(x = reorder(country, -tot_cases_n), y = tot_cases_n, fill = type)) +
  geom_bar(stat = 'identity') +
  theme_bw() +
  scale_fill_manual(values = wes_palette("Royal1")) +
  labs(x = "Country", y = "Total cases", title = "Cases per countries in dataset") +
  theme(axis.text.x=element_text(angle=90, hjust=1))
```



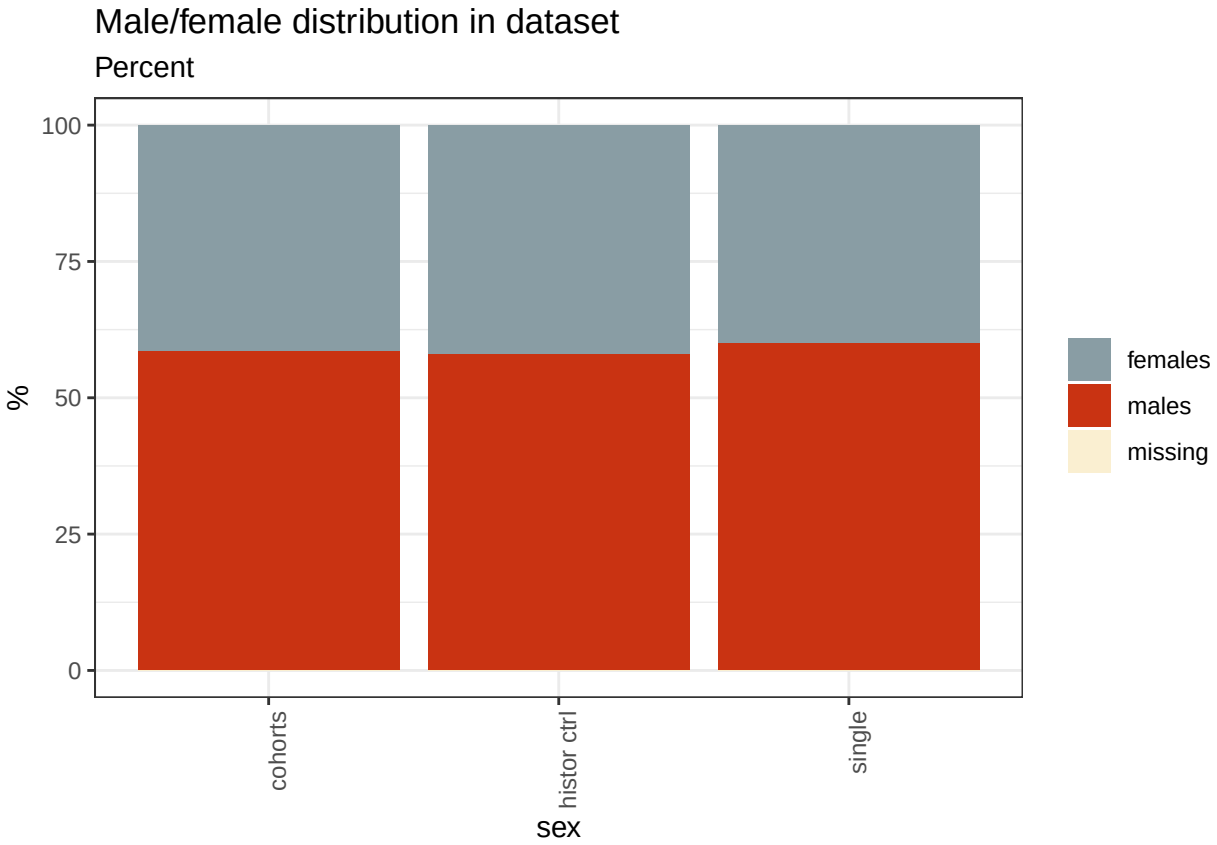
Total cases and deaths

```
#var_id_cohort = "outcome_death_n"
#var_id_single = "outcome_death"
#var_id = "deaths"
makeBarplot("outcome_death_n", "outcome_death", "deaths")
```



Sex and age distribution

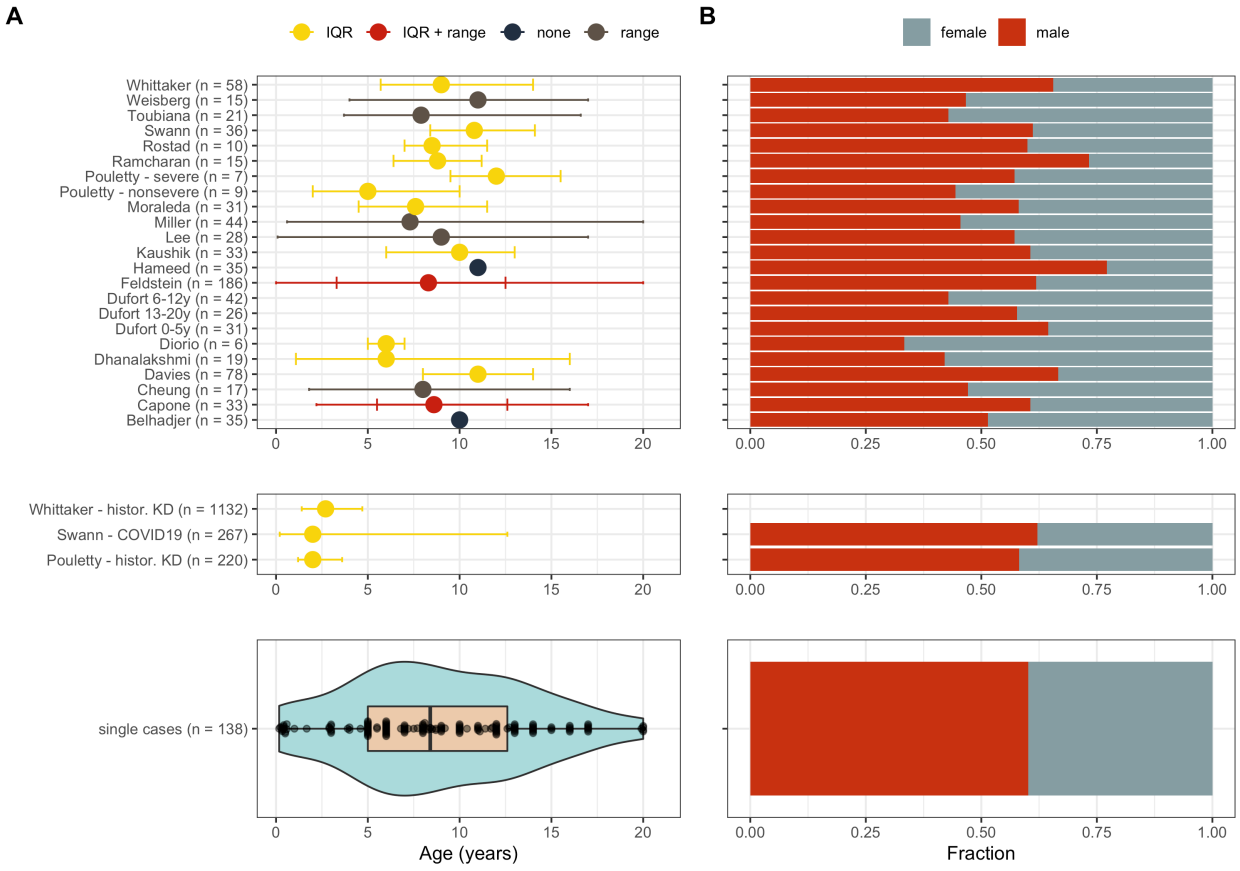
```r
n_cohort <- df_cohort %>% select(tot_cases_n) %>% sum()
var_cohort <- df_cohort %>% select(contains("sex"))
var_cohort <- colSums(var_cohort, na.rm = TRUE)
var_cohort <- var_cohort/sum(df_cohort$tot_cases_n)*100
var_cohort["sex_na"] <- (100 - var_cohort["sex_m"] - var_cohort["sex_f"])

var_control <- df_cohort_controls %>% filter(cohort_id == "Pouletty - histor. KD") %>% select(contains("sex"))
var_control <- colSums(var_control, na.rm = TRUE)
var_control <- var_control/sum(df_cohort_controls %>% filter(cohort_id == "Pouletty - histor. KD") %>% select(tot_cases_n))*100
var_control["sex_na"] <- (100 - var_control["sex_m"] - var_control["sex_f"])

n_single <- df_singlecases %>% nrow()
var_single <- df_singlecases %>% select(contains("sex"))
var_single$sex_m <- ifelse(var_single$sex == "M", TRUE, FALSE)
var_single$sex_f <- ifelse(var_single$sex == "F", TRUE, FALSE)
cols <- sapply(var_single, is.logical)
var_single[,cols] <- lapply(var_single[,cols], as.numeric)
var_single <- colSums(var_single %>% select(-sex), na.rm = TRUE)
var_single <- var_single/nrow(df_singlecases)*100
var_single["sex_na"] <- (100 - var_single["sex_m"] - var_single["sex_f"])

bar_df_prct <- data.frame(
  x = c("males", "females", "missing", "males", "females", "missing", "males", "females", "missing"),
  vals = c(var_single, var_cohort, var_control),
  col = c(rep("single", length(var_single)), rep("cohorts", length(var_cohort)), rep("histor ctrl", length(var_control))
  ))

p_prct <- ggplot(bar_df_prct, aes(x = col, y =  vals, fill = x)) +
  geom_bar(stat = "identity", position = "stack") +
  theme_bw() +
  labs(title = "Male/female distribution in dataset", subtitle = "Percent", x = "sex", y = "%", col = " ")  + lims(y = c(0,100)) + theme(axis.text.x=element_text(angle=90, hjust=1), legend.title = element_blank())+
  scale_fill_manual(values = wes_palette("Royal1"))
p_prct
```




```
var_cohort <- df_cohort %>% select(contains("sex") | ("cohort_id") | "tot_cases_n")
sex_f <- var_cohort %>% group_by(cohort_id) %>% summarize(prct = sex_f/tot_cases_n) %>%  mutate(sex = "female")
sex_m <- var_cohort %>% group_by(cohort_id) %>% summarize(prct = sex_m/tot_cases_n) %>% mutate(sex = "male")
sex_all <- rbind(sex_f, sex_m)

p_sex_cohort <- ggplot(sex_all, aes(y = cohort_id, x = prct, fill = sex)) +
  geom_bar(stat = "identity", position = "fill") +
  theme_bw() + labs(x = "") +
  scale_fill_manual(values = wes_palette("Royal1"))

var_controls <- df_cohort_controls %>% filter(cohort_id == "Pouletty - histor. KD") %>% select(contains("sex") | ("cohort_id") | "tot_cases_n")
sex_f <- var_controls %>% group_by(cohort_id) %>% summarize(prct = sex_f/tot_cases_n) %>% mutate(sex = "female")
sex_m <- var_controls %>% group_by(cohort_id) %>% summarize(prct = sex_m/tot_cases_n) %>% mutate(sex = "male")
sex_all <- rbind(sex_f, sex_m)

p_sex_controls <- ggplot(sex_all, aes(y = cohort_id, x = prct, fill = sex)) +
  geom_bar(stat = "identity", position = "fill") +
  theme_bw() + labs(x = "") +
  scale_fill_manual(values = wes_palette("Royal1"))

n_single <- df_singlecases %>% nrow()
var_single <- df_singlecases %>% select(contains("sex"))
var_single$sex_m <- ifelse(var_single$sex == "M", TRUE, FALSE)
var_single$sex_f <- ifelse(var_single$sex == "F", TRUE, FALSE)
cols <- sapply(var_single, is.logical)
var_single[,cols] <- lapply(var_single[,cols], as.numeric)
var_single <- colSums(var_single %>% select(-sex), na.rm = TRUE)
var_single <- var_single/nrow(df_singlecases)*100

sex_single <- data.frame(cohort_id = "single_cases", prct = c(var_single["sex_m"], var_single["sex_f"]), sex = c("male", "female"))

p_sex_single <- ggplot(sex_single, aes(y = cohort_id, x = prct, fill = sex)) +
  geom_bar(stat = "identity", position = "fill") +
  theme_bw() +
  scale_fill_manual(values = wes_palette("Royal1"))

a <- plot_grid(p_sex_cohort, p_sex_controls, p_sex_single, align = "v", nrow = 3, rel_heights = c(5/7, 1/7, 1/7))
```

```
cohort_age <- df_cohort_controls %>% select(contains("cohort_id") | contains("age") | contains("cohort_type")  | contains("tot_cases_n"))
cohort_age$cohort_id <- paste0(cohort_age$cohort_id, " (n = ", cohort_age$tot_cases_n,")")
cohort_age$age_med_yrs <- as.numeric(cohort_age$age_med_yrs )
cohort_age$age_Q1_yrs <- as.numeric(cohort_age$age_Q1_yrs)
cohort_age$age_Q3_yrs <- as.numeric(cohort_age$age_Q3_yrs)
cohort_age$age_min_yrs <- as.numeric(cohort_age$age_min_yrs)
cohort_age$age_max_yrs <- as.numeric(cohort_age$age_max_yrs)

cohort_age$data_descr <- ifelse(!is.na(cohort_age$age_Q1_yrs) & is.na(cohort_age$age_min_yrs) , "IQR",
                                ifelse(is.na(cohort_age$age_Q1_yrs) & !is.na(cohort_age$age_min_yrs), "range",
                                       ifelse(!is.na(cohort_age$age_Q1_yrs) & !is.na(cohort_age$age_min_yrs), "both", "none")))

p_age_cohort <- ggplot(cohort_age %>% filter(cohort_type == "MIS-C"), aes(y = cohort_id, x = age_med_yrs, col = data_descr)) +
```

```
  geom_point(size = 4) +
  geom_errorbar(aes(xmin=age_Q1_yrs, xmax=age_Q3_yrs), width=.8, position=position_dodge(.9)) +
  geom_errorbar(aes(xmin=age_min_yrs,  xmax=age_max_yrs), width=.2, position=position_dodge(.9)) +
  theme_bw() + lims(x = c(0,21)) +
  labs(y = "cohort", x = "", col = "bars") + theme(legend.position="top")+
  scale_color_manual(values = c(wes_palette("BottleRocket2")[1:3], wes_palette("BottleRocket1")[2]))

p_age_controls <- ggplot(cohort_age %>% filter(cohort_type != "MIS-C"), aes(y = cohort_id, x = age_med_yrs, col = data_descr)) +
  geom_point(size = 4) +
  geom_errorbar(aes(xmin=age_Q1_yrs, xmax=age_Q3_yrs), width=.2, position=position_dodge(.9)) +
  geom_errorbar(aes(xmin=age_min_yrs,  xmax=age_max_yrs), width=.2, position=position_dodge(.9)) +
  theme_bw() + lims(x = c(0,21)) +
  labs(y = "cohort", x = "", col = "bars") + theme(legend.position="none")+
  scale_color_manual(values = wes_palette("BottleRocket2")[2])

p_age_single <- ggplot(df_singlecases, aes(x = as.numeric(age), y = paste0("single cases (n = ", n_single,")"))) +
  geom_violin(fill = wes_palette("Darjeeling2")[4]) +
  geom_boxplot(width=.3, fill = wes_palette("Darjeeling2")[1]) +
  theme_bw() + geom_beeswarm(groupOnX=FALSE, alpha = 0.5) + lims(x = c(0,21)) +
  labs(y = "cohort", x = "Age (years)")

a <- plot_grid(p_age_cohort, p_age_controls, p_age_single, align = "v", nrow = 3, rel_heights = c(2/3, 1/5, 1/3))
```



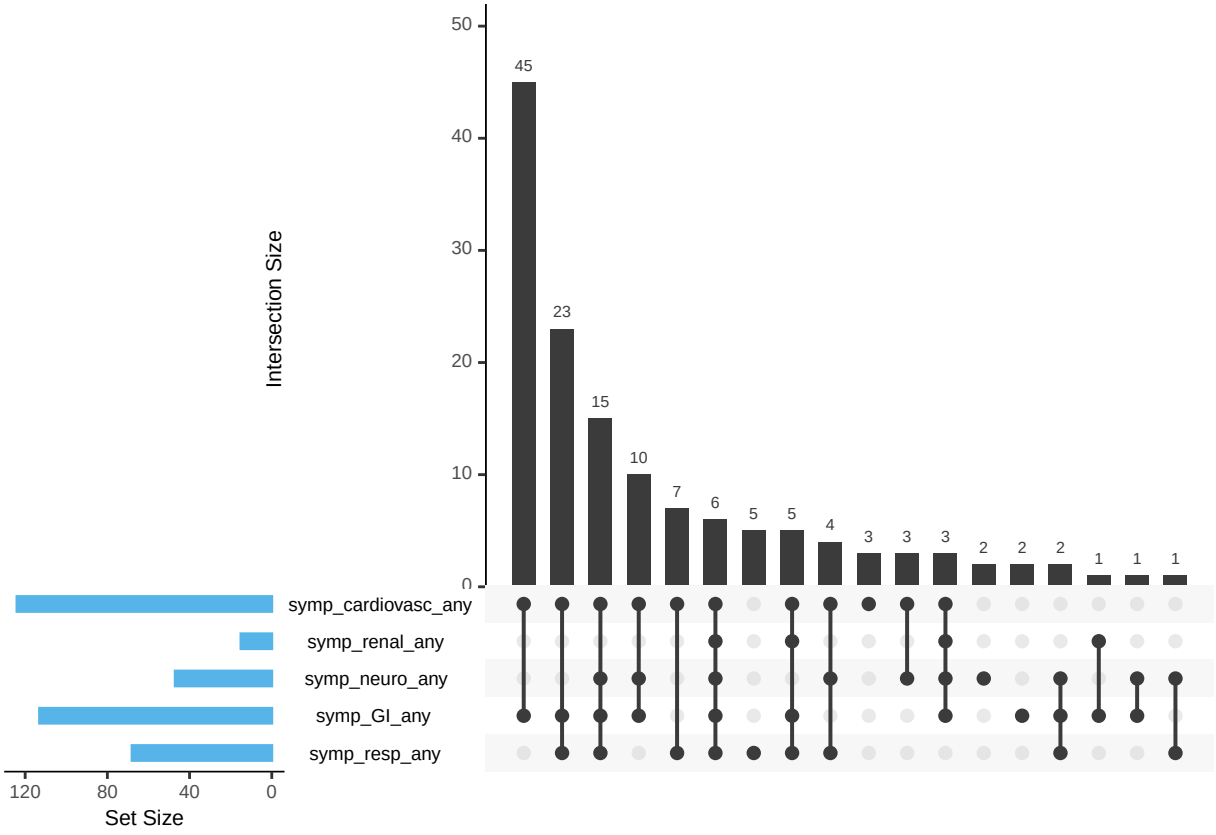
Symptoms

Single cases

All symptoms Where applicable, overlap of variable in the single case group was summarized with Upset plots (Lex & Gehlenborg, Nature Methods, 2014).

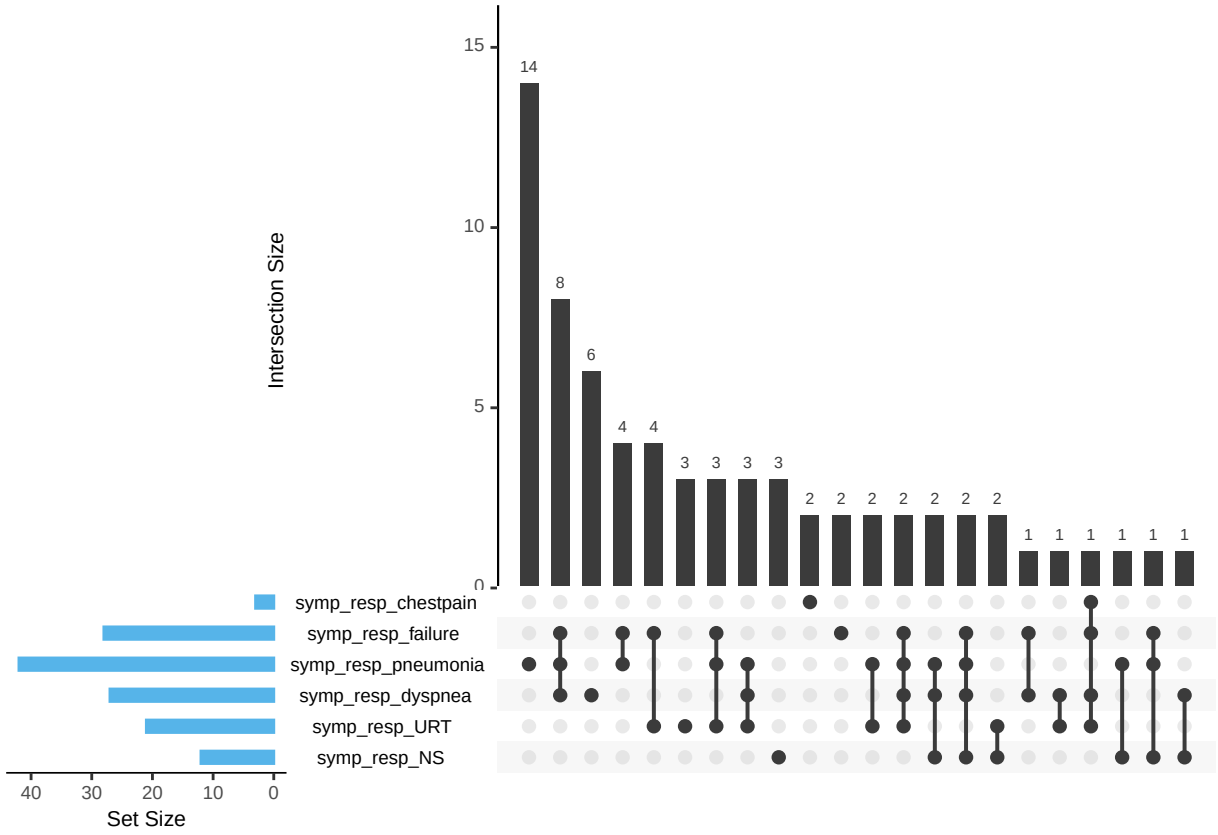
```
makeUpsetR <- function(input_df){
  var_single <- input_df
  cols <- sapply(var_single, is.logical)
  var_single[,cols] <- lapply(var_single[,cols], as.numeric)

  var_single_upsetr <- var_single
  var_single_upsetr[is.na(var_single_upsetr)] <- 0
  var_single_upsetr <- as.data.frame(var_single_upsetr)
  for(i in 1:ncol(var_single_upsetr)){ var_single_upsetr[ , i] <- as.integer(var_single_upsetr[ , i]) }
  upset(var_single_upsetr, sets = c(colnames(var_single_upsetr)), sets.bar.color = "#56B4E9",
        order.by = "freq", keep.order = TRUE)#, empty.intersections = "on", keep.order = FALSE)
```

```
}

makeUpsetR(df_singlecases %>% select(contains( "symp")) %>% select(contains("any")))
```



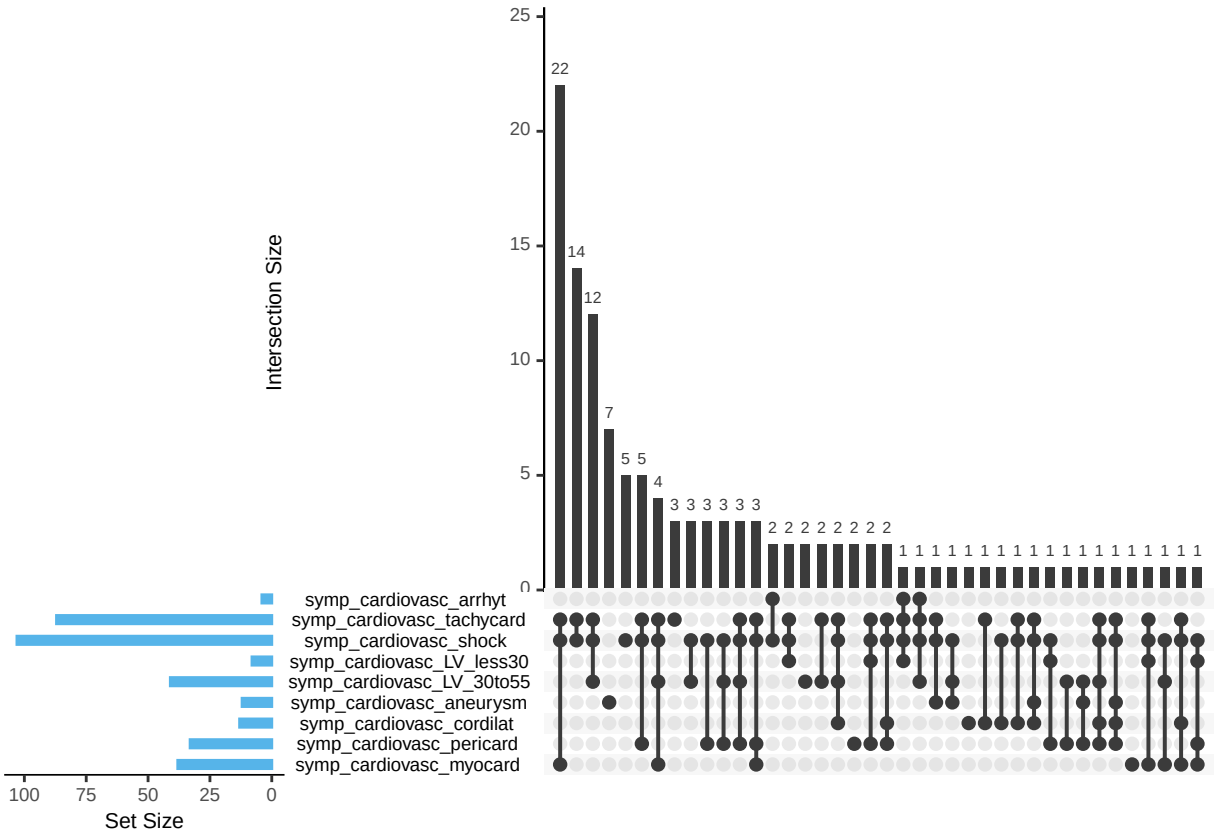
Respiratory

```
makeUpsetR(df_singlecases %>% select(contains("symp")) %>% select(contains("resp")) %>% select(-contains("any")))
```



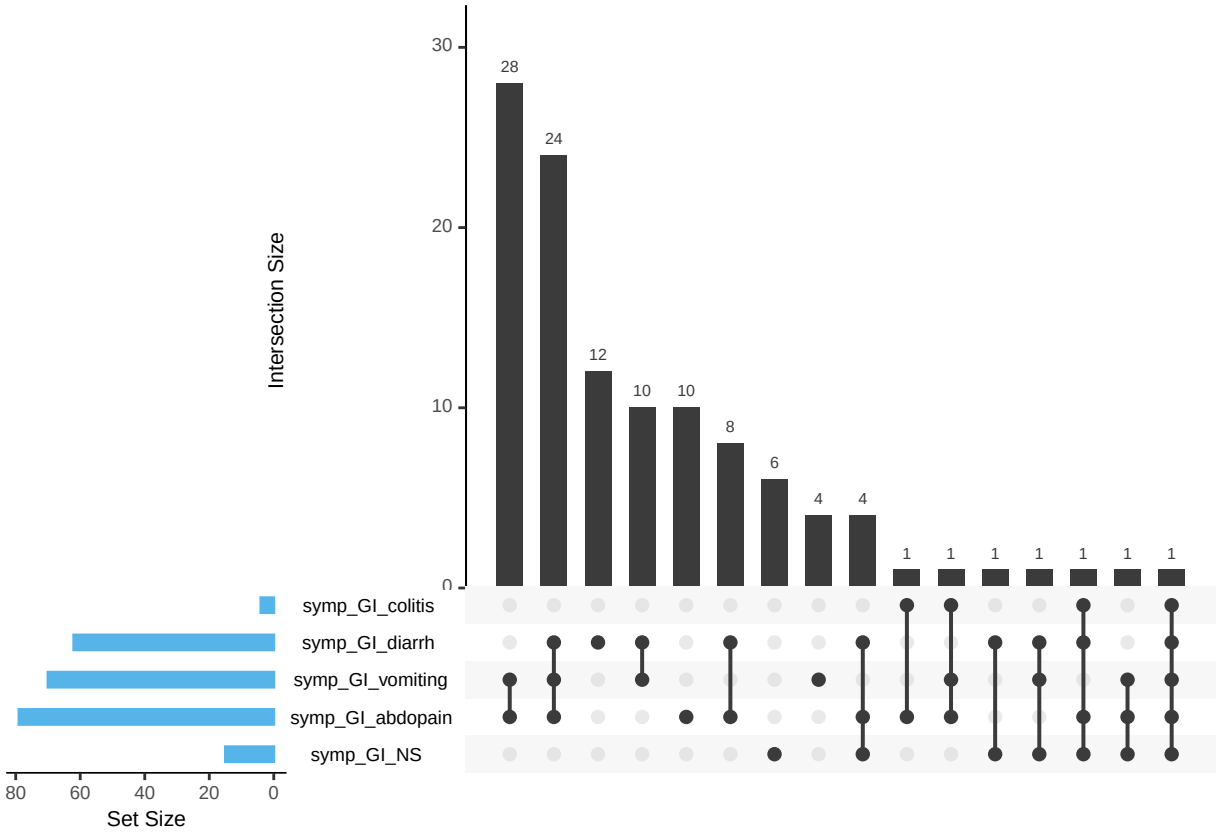
Cardiovascular

```
makeUpsetR(df_singlecases %>% select(contains("symp")) %>% select(contains("cardiovasc")) %>% select(-contains("LVEF")) %>% select(-contains("any")))
```



GI

```
makeUpsetR(df_singlecases %>% select(contains("symp")) %>% select(contains("GI")) %>% select(-contains("neuro")) %>% select(-contains("any")))
```

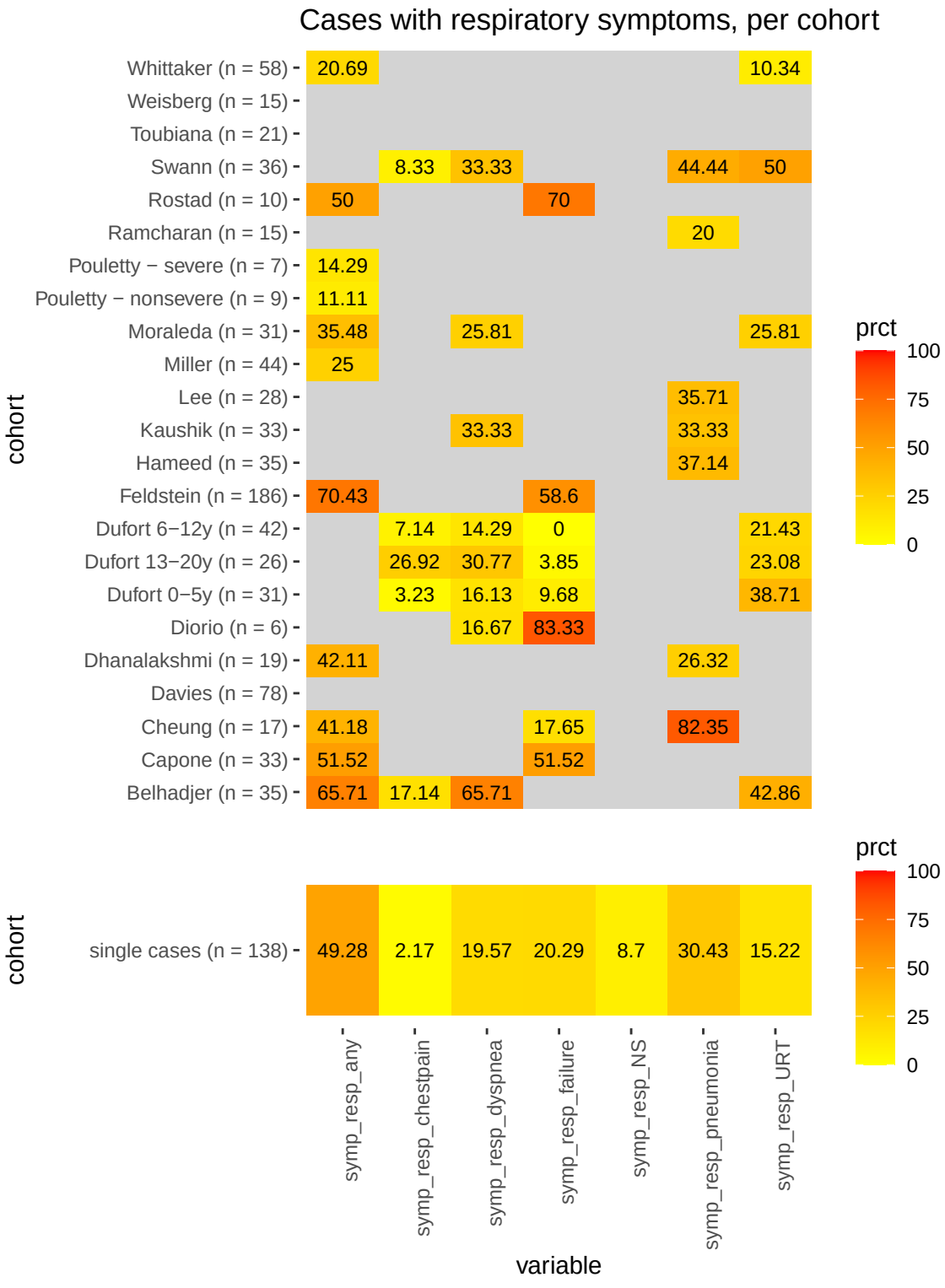


Single cases + cohort

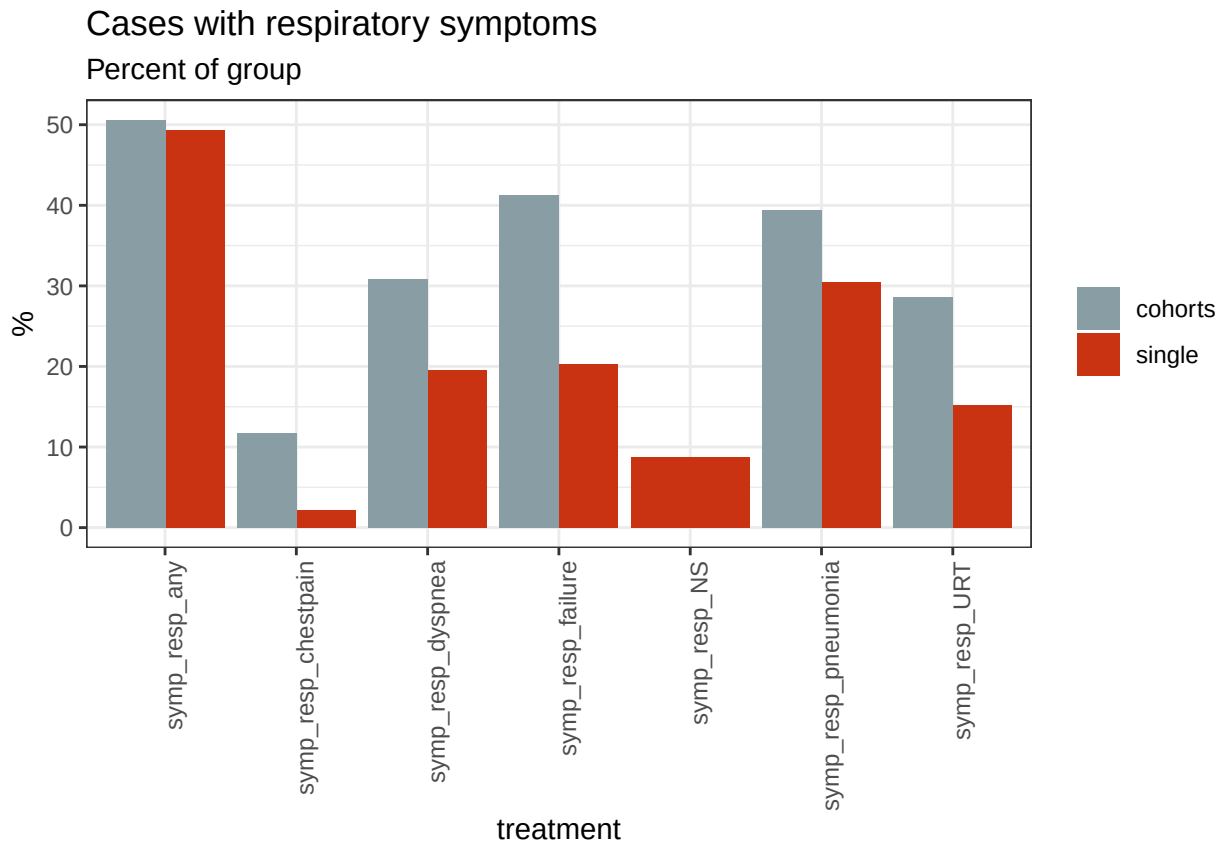
Respiratory

```r
barSymp <- function(colname_chort, colname_single, exclude_single = NULL, plottitle){

  var_cohort <- df_cohort %>%
    select(contains("cohort_id") | contains("tot_cases_n") | (contains(colname_chort) & contains("_n")))

  var_cohort <- var_cohort %>%
    gather(variable, value, 3:ncol(var_cohort)) %>%
    drop_na(value)  %>% group_by(variable) %>%
    summarize(prct = sum(value)/sum(tot_cases_n)*100)

  var_cohort <- setNames(var_cohort$prct, var_cohort$variable)
  names(var_cohort) <- sub("_n", "", names(var_cohort))

  n_single <- df_singlecases %>% nrow()

  if (!is.null(exclude_single)){
    var_single <- df_singlecases %>% select(-contains(exclude_single))
    var_single <- var_single %>% select(contains(colname_single))
  } else
  {
    var_single <- df_singlecases %>% select(contains(colname_single))
  }

  #%>% select(-contains("any"))
  cols <- sapply(var_single, is.logical)
  var_single[,cols] <- lapply(var_single[,cols], as.numeric)
  var_single <- colSums(var_single, na.rm = TRUE)
  var_single <- var_single/nrow(df_singlecases)*100

  bar_df_prct <- data.frame(
    x = c(names(var_single), names(var_cohort)),
    vals = c(var_single, var_cohort),
    col = c(rep("single", length(var_single)), rep("cohorts", length(var_cohort)))
  )

  p_prct <- ggplot(bar_df_prct, aes(x = x, y =  vals, fill = col)) +
    geom_bar(stat = "identity", position = "dodge") +
    theme_bw() +
    labs(title = plottitle,
         subtitle = "Percent of group", x = "treatment", y = "%", col = " ")  +
    theme(axis.text.x=element_text(angle=90, hjust=1), legend.title = element_blank())+
    scale_fill_manual(values = wes_palette("Royal1"))
  p_prct
}
```

```
makeHeatmap_cohort("symp_resp", "symp_resp", plottitle = "Cases with respiratory symptoms, per cohort")
```



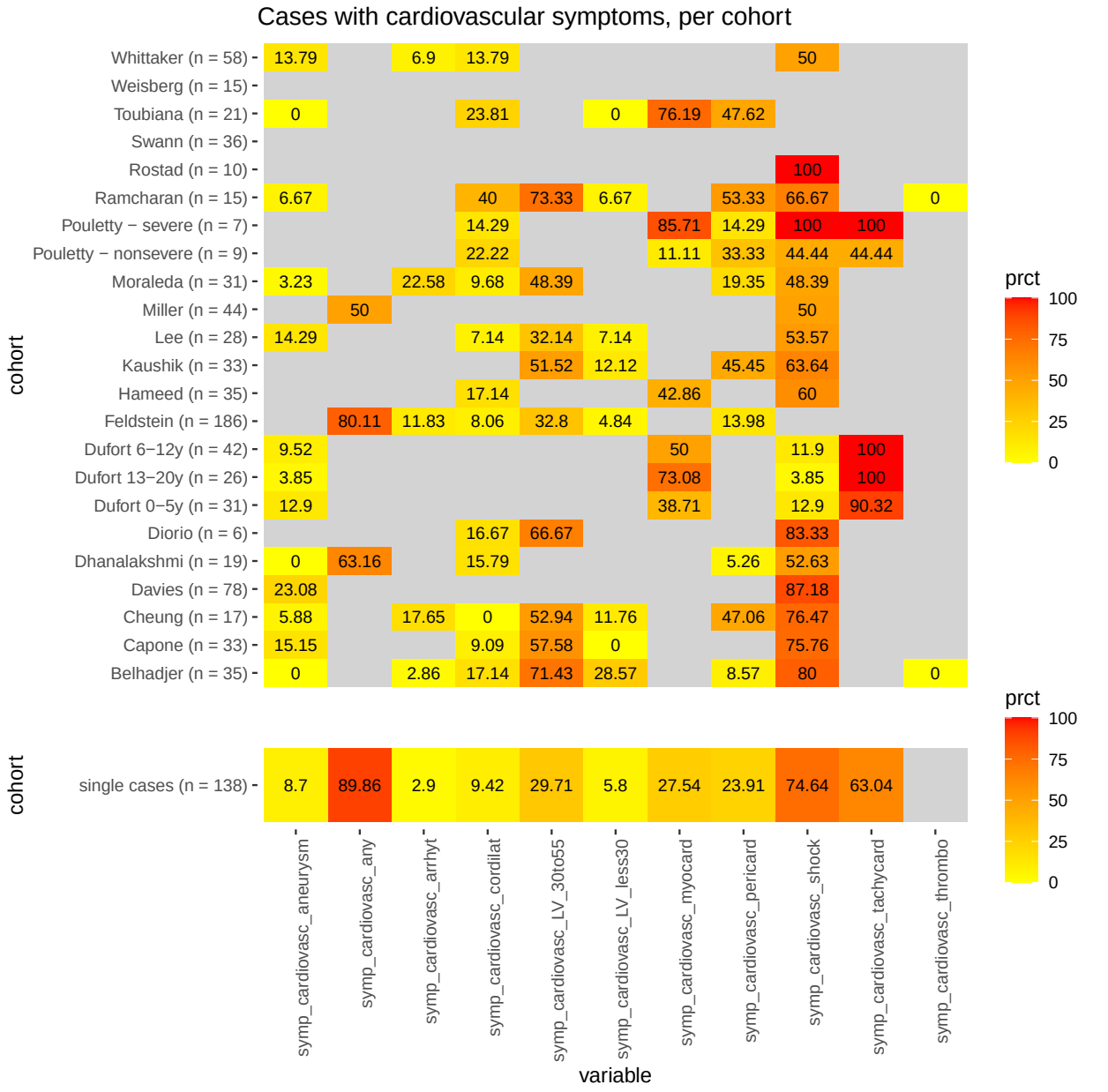

```
barSymp("symp_resp", "symp_resp", plottitle = "Cases with respiratory symptoms")
```



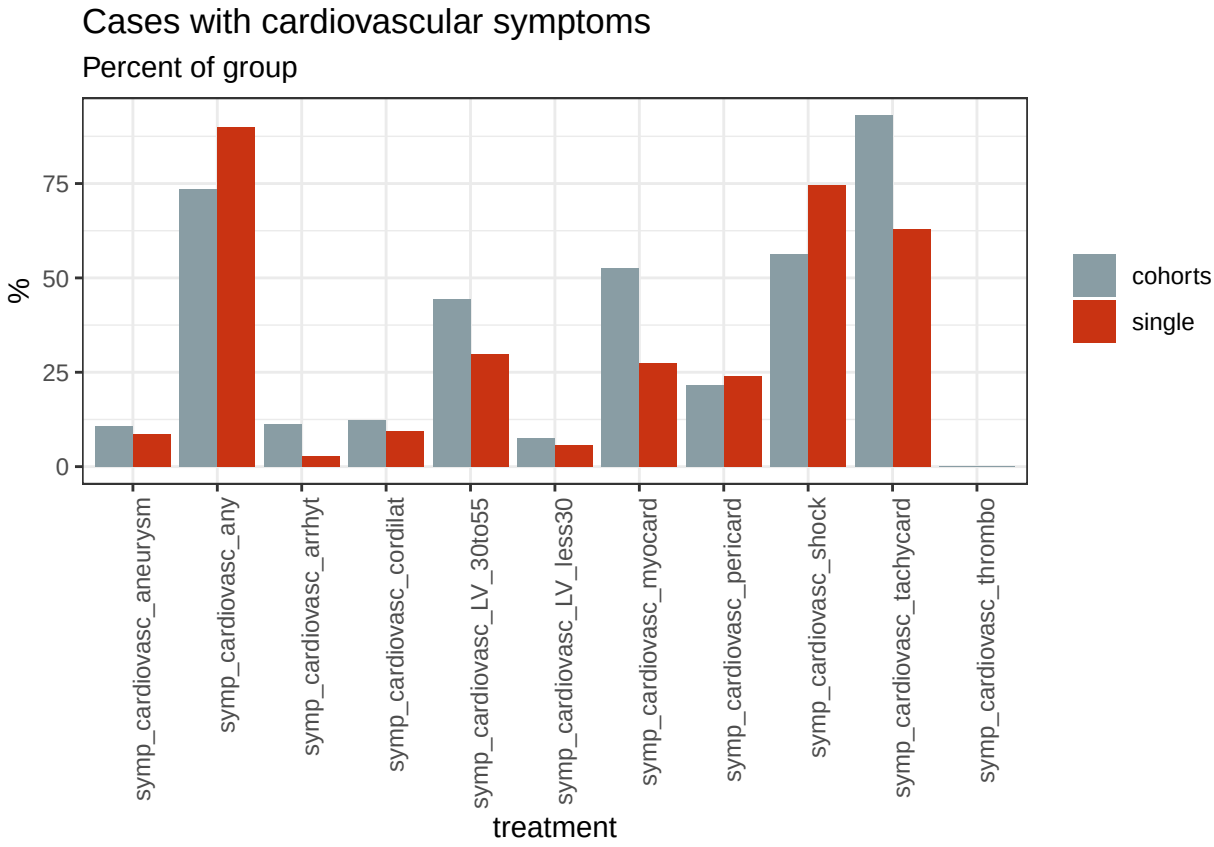

```
# var_cohort <- df_cohort %>% select(("cohort_id") | "tot_cases_n" |( contains("symp_resp") & contains("n")))
#
# resp_symp_cohort <- var_cohort %>%
#   gather(variable, value, 3:ncol(var_cohort)) %>% group_by(cohort_id, variable) %>% summarize(prct = value/tot_cases_n)
#
# ggplot(resp_symp_cohort, aes(x = prct, y = cohort_id, col = variable)) + geom_point()
```

Cardiovascular

```
makeHeatmap_cohort("symp_cardiovasc", "symp_cardiovasc", exclude_single = "symp_cardiovasc_LVEF", plottitle = "Cases with cardiovascular symptoms, per
    cohort")
```

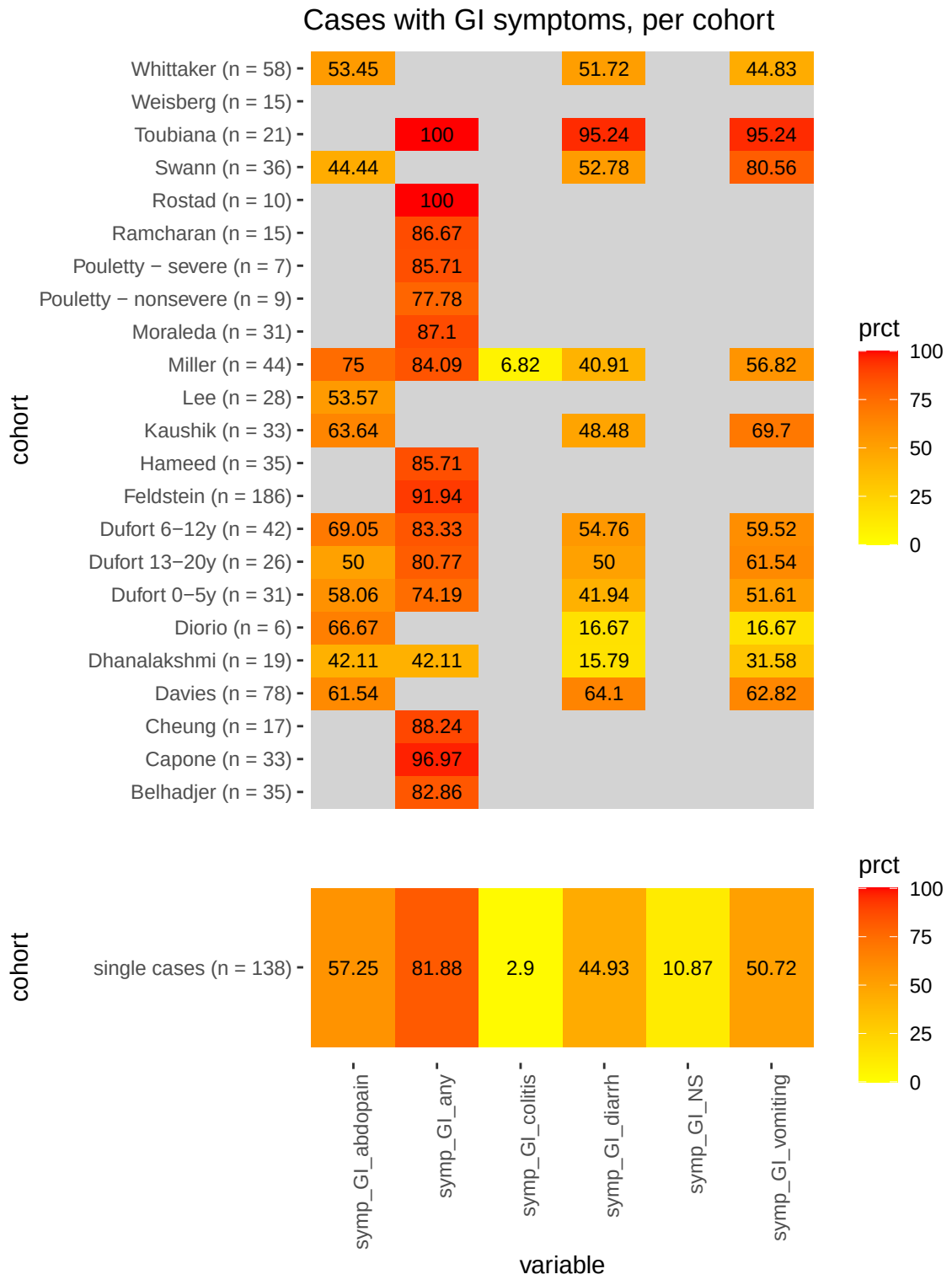



```
barSymp("symp_cardiovasc", "symp_cardiovasc", exclude_single = "symp_cardiovasc_LVEF", plottitle = "Cases with cardiovascular symptoms")
```

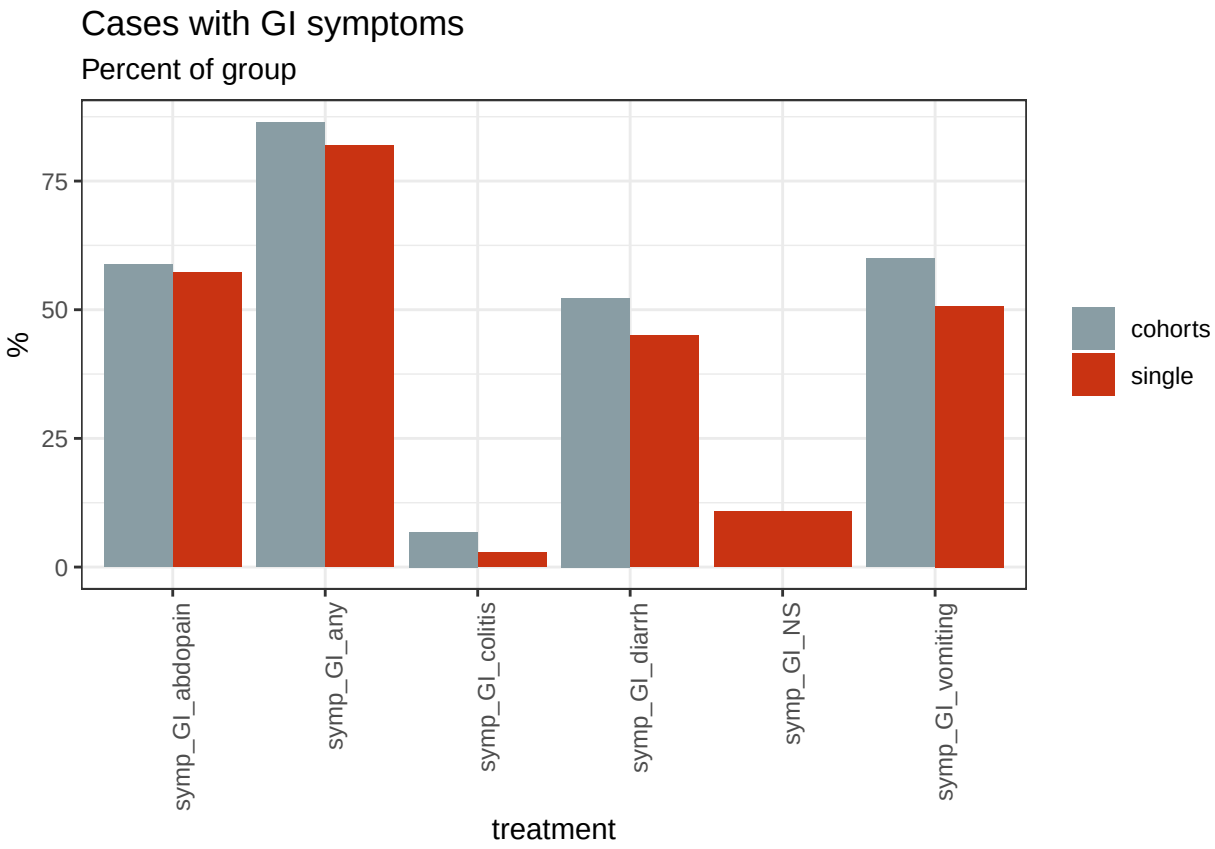


Gastro-intestinal

```
makeHeatmap_cohort("symp_GI", "symp_GI", plottitle = "Cases with GI symptoms, per cohort")
```



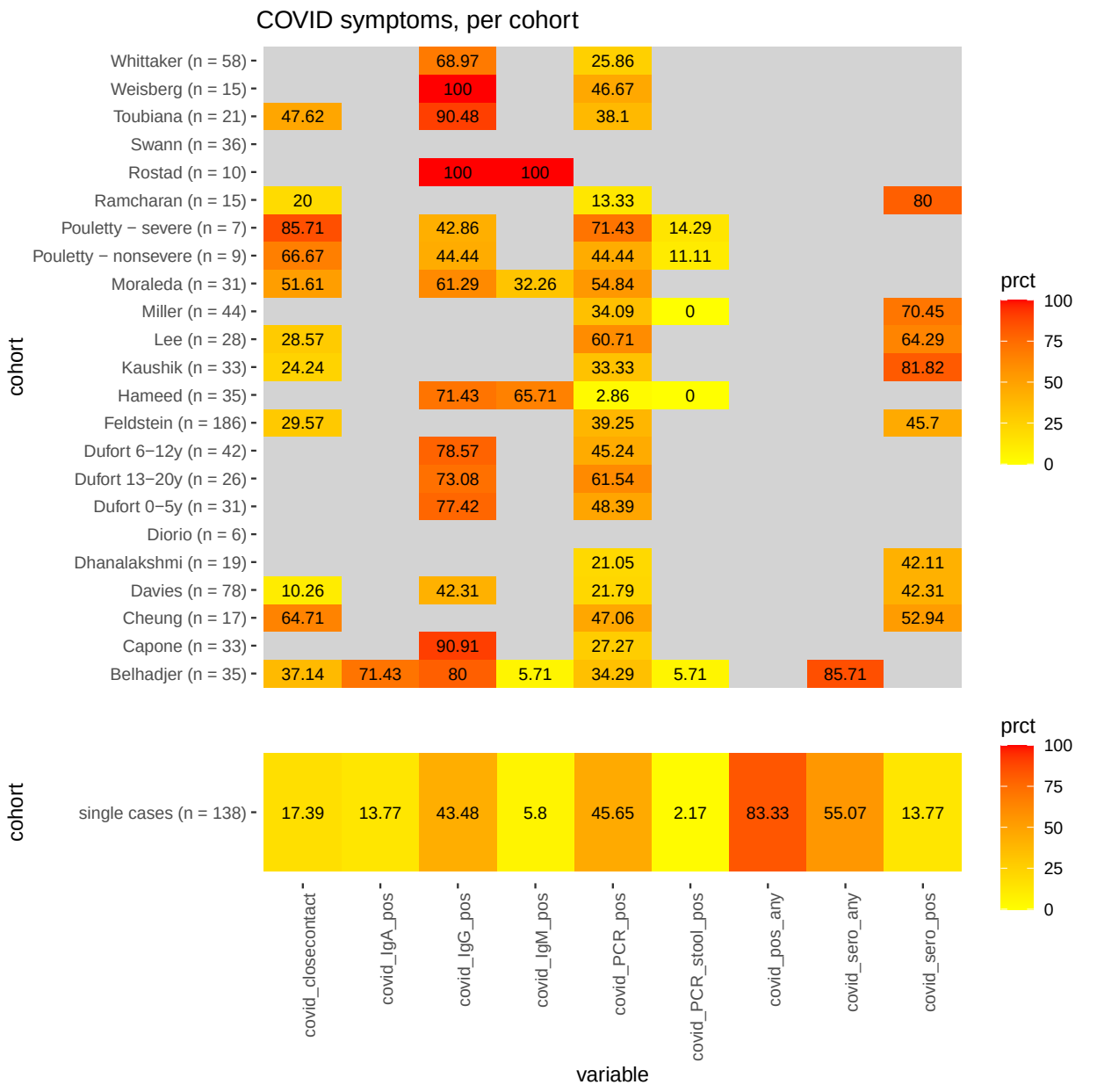

```
barSymp("symp_GI", "symp_GI", plottitle = "Cases with GI symptoms")
```



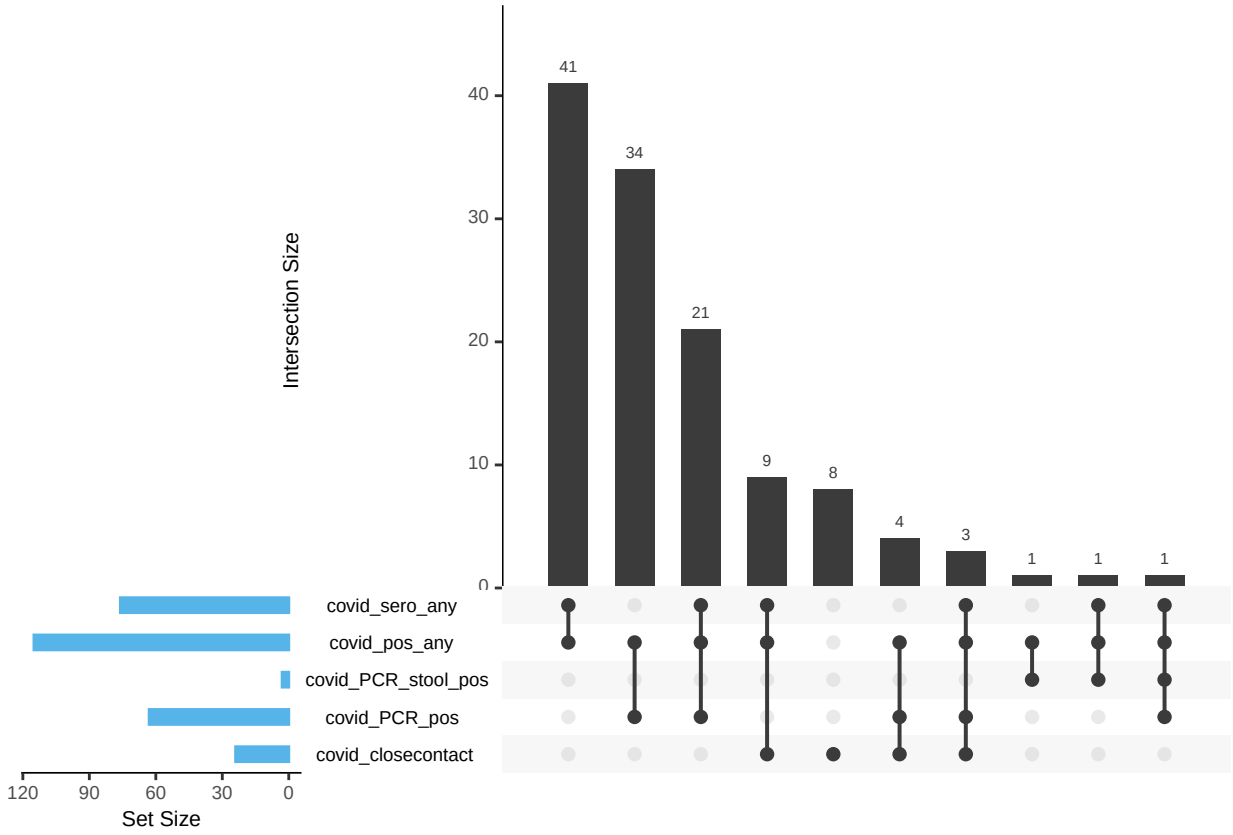
COVID contact

```
var_cohort <- df_cohort %>% select(("cohort_id" | "tot_cases_n") | ( contains("covid") & contains("_n") & (contains("pos") | contains("closecont")  |
      contains("any"))))
var_cohort$cohort_id <- paste0(var_cohort$cohort_id, " (n = ", as.character(var_cohort$tot_cases_n),")")

var_cohort <- var_cohort %>%
  gather(variable, value, 3:ncol(var_cohort)) %>% group_by(cohort_id, variable) %>% summarize(prct = value/tot_cases_n*100)

var_cohort$variable <- sub("n_", "", var_cohort$variable)

var_single <- df_singlecases %>% select(contains("covid"))
cols <- sapply(var_single, is.logical)
var_single[,cols] <- lapply(var_single[,cols], as.numeric)
var_single <- colSums(var_single, na.rm = TRUE)
var_single <- var_single/nrow(df_singlecases)*100
var_single <- as.data.frame(var_single) %>% rownames_to_column()
var_single$cohort_id <- paste0("single cases (n = ", n_single_cases,")")
colnames(var_single) <- c("variable", "prct", "cohort_id")


missing <- setdiff(var_single$variable, var_cohort$variable)
if (length(missing) != 0 ){
  missing_df <- data.frame(variable = missing, prct = rep(NA, length(missing)), cohort_id = rep(unique(var_cohort$cohort_id), length(missing)))
  var_cohort <- bind_rows(var_cohort, as_tibble(missing_df))
}

missing <- setdiff(var_cohort$variable, var_single$variable)

if (length(missing) != 0) {
  if (length(missing) != 0){
    data.frame(variable = missing, prct = rep(NA, length(missing)), cohort_id = rep(unique(var_single$cohort_id), length(missing)))
    var_single <- bind_rows(var_single, as_tibble(missing_df))
  }
}


hm_cohort <- ggplot(var_cohort, aes(x = variable, y = cohort_id, fill = prct)) +
  geom_tile() + theme_classic() +
  theme(axis.text.x=element_blank(), axis.ticks.x=element_blank(), axis.line=element_blank())+
  scale_fill_gradient(low = "yellow", high="red", na.value = "lightgray", limits = c(0,100)) +
  labs(x = "", y = "cohort", title = "COVID symptoms, per cohort") +
  geom_text(aes(label=round(prct, 2)), size = 3, color = "black")

hm_single <- ggplot(var_single, aes(x = variable, y = cohort_id, fill = prct)) +
  geom_tile() +  theme_classic() +
  theme(axis.text.x=element_text(angle=90, hjust=1), axis.line=element_blank())+
  scale_fill_gradient(low = "yellow", high = "red", na.value = "lightgray", limits = c(0,100))+ labs(y = "cohort") +
```

```
  geom_text(aes(label=round(prct, 2)), size = 3, color = "black")

plot_grid(hm_cohort, hm_single, align = "v", nrow = 2, rel_heights = c(2/3, 1/3))
```




```
var_cohort <- df_cohort %>%
  select(contains("cohort_id") | contains("tot_cases_n") | contains("covid") & contains("_n") & (contains("_pos") | contains("close")))

covid_cohort <- var_cohort %>%
  gather(variable, value, 3:ncol(var_cohort)) %>%
  drop_na(value)  %>% group_by(variable) %>%
  summarize(prct = sum(value)/sum(tot_cases_n)*100)

covid_cohort <- setNames(covid_cohort$prct, covid_cohort$variable)

n_single <- df_singlecases %>% nrow()
var_single <- df_singlecases %>% select(contains("covid"))
cols <- sapply(var_single, is.logical)
var_single[,cols] <- lapply(var_single[,cols], as.numeric)

makeUpsetR(df_singlecases %>% select(contains("covid")) %>% select(-contains("covid_IgM_pos")) %>% select(-contains("covid_IgA_pos"))  %>% select(-contains(
       "covid_IgG_pos"))  %>% select(-contains("covid_sero_pos")) )
```




```
var_single <- colSums(var_single, na.rm = TRUE)
var_single <- var_single/nrow(df_singlecases)*100

bar_df_prct <- data.frame(
  x = c("close contact reported", "PCR +", "stool +","PCR or stool or sero +", "any serology +", "sero + further NS", "IgA +", "IgM +", "IgG +", "close
        contact reported", "IgA +", "IgG +", "IgM +", "PCR +", "sero + further NS", "stool +"),
  vals = c(var_single, covid_cohort),
  col = c(rep("single", length(var_single)), rep("cohorts", length(covid_cohort)))
)

p_prct <- ggplot(bar_df_prct, aes(x = x, y =  vals, fill = col)) +
  geom_bar(stat = "identity", position = "dodge") +
  theme_bw() +
  labs(title = "SARS-CoV2 testing",
       subtitle = "Prct", x = "variable", y = "%", col = " ") +
  theme(axis.text.x=element_text(angle=90, hjust=1)) +
  scale_fill_manual(values = wes_palette("Royal1"))
#p_prct

neither_PCR_Ig <- nrow(df_singlecases %>% filter((covid_sero_any == FALSE | is.na(covid_sero_any)) & (covid_PCR_pos == FALSE | is.na(covid_PCR_pos)) & (
      covid_PCR_stool_pos == FALSE | is.na(covid_PCR_stool_pos))))

neither_PCR_Ig_closecontact <-
  nrow(df_singlecases %>% filter((covid_sero_any == FALSE |
                                    is.na(covid_sero_any)) &
                                   (covid_PCR_pos == FALSE |
                                      is.na(covid_PCR_pos)) &
                                   (covid_PCR_stool_pos == FALSE |
                                      is.na(covid_PCR_stool_pos)) &
                                   (covid_closecontact == FALSE | is.na(covid_closecontact))
  ))

print(paste0("Cases with neither PCR nor serology: ", neither_PCR_Ig))
```

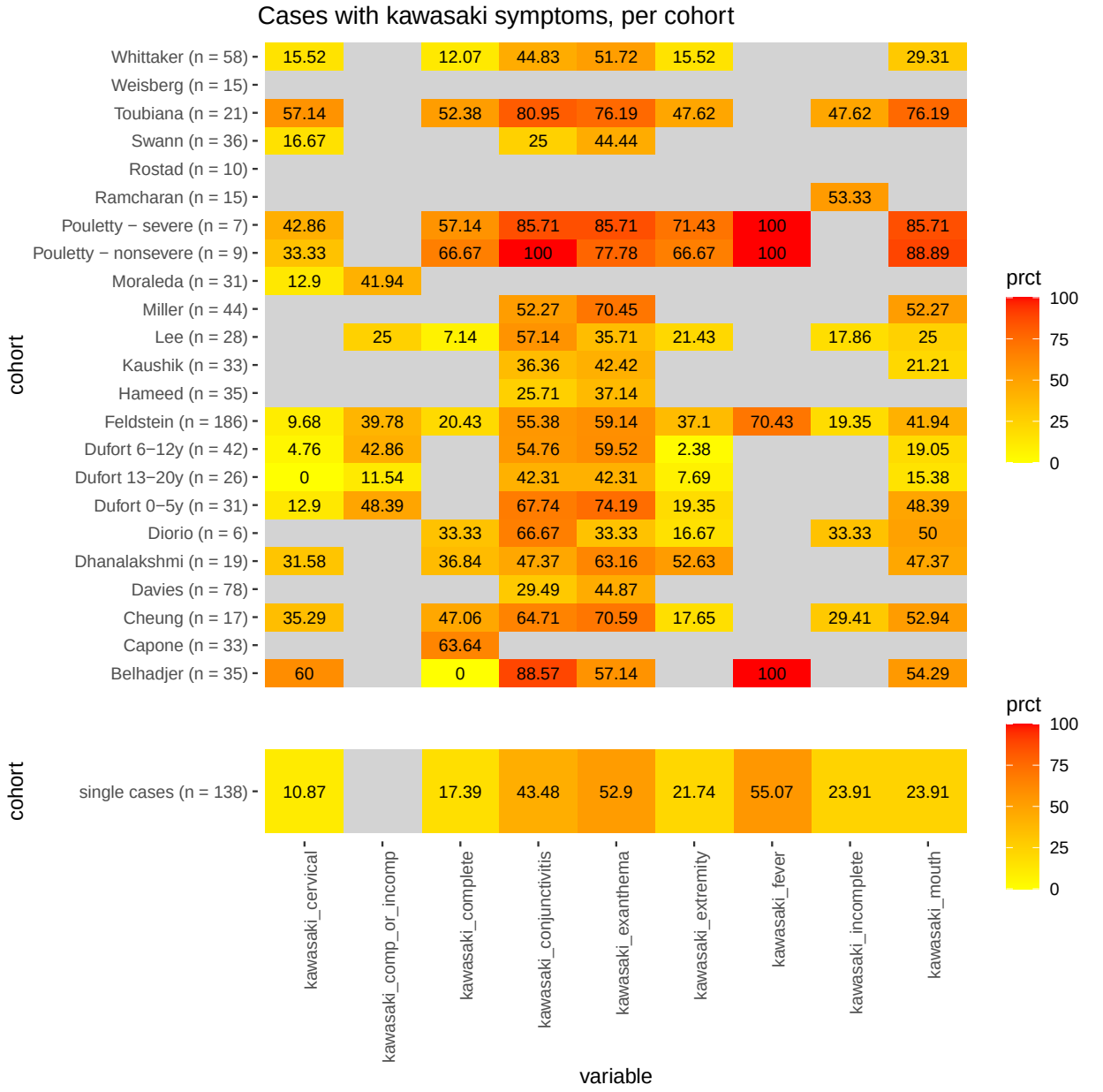
```
## [1] "Cases with neither PCR nor serology: 23"
```

```
print(paste0("Cases with neither PCR nor serology nor closecontact: ", neither_PCR_Ig_closecontact))
```

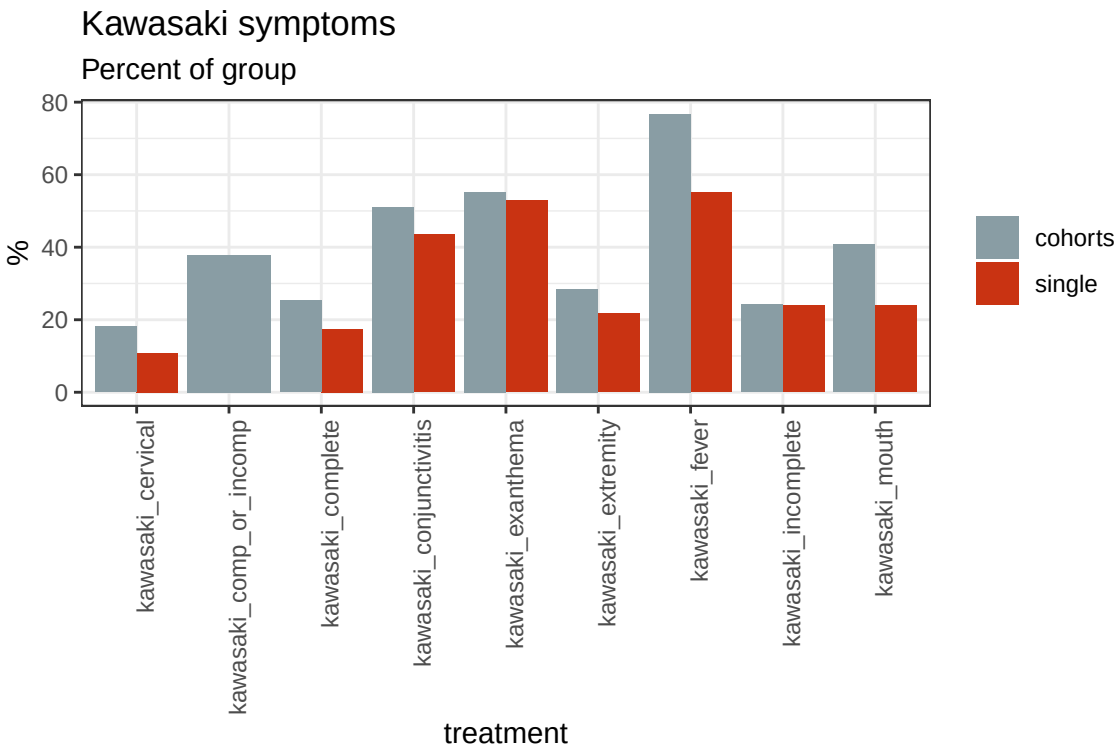
```
## [1] "Cases with neither PCR nor serology nor closecontact: 15"
```

Kawasaki criteria

```
makeHeatmap_cohort("kawasaki", "kawasaki",exclude_single = "koyobas", plottitle = "Cases with kawasaki symptoms, per cohort")
```

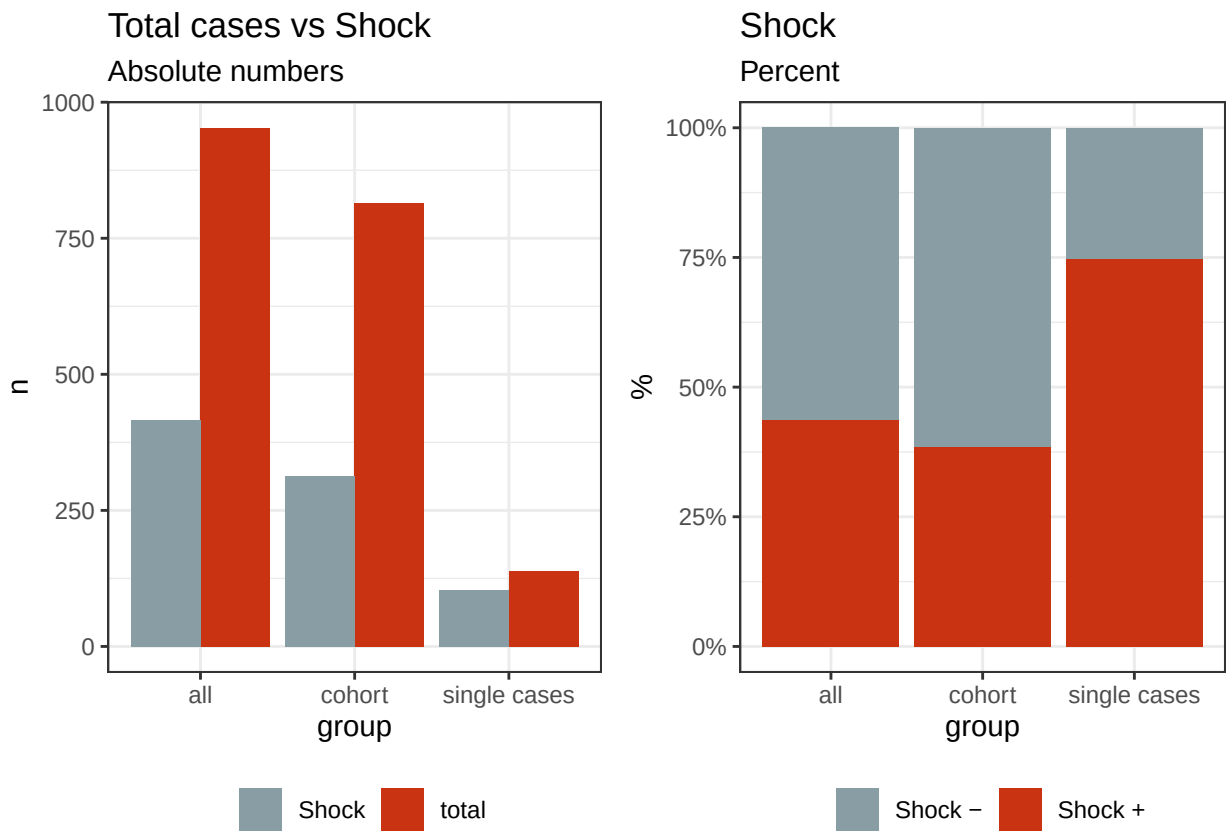



```
barSymp("kawasaki", "kawasaki", exclude_single = "koyobas", plottitle = "Kawasaki symptoms")
```



Shock

```
makeBarplot("symp_cardiovasc_shock_n", "symp_cardiovasc_shock", "Shock")
```

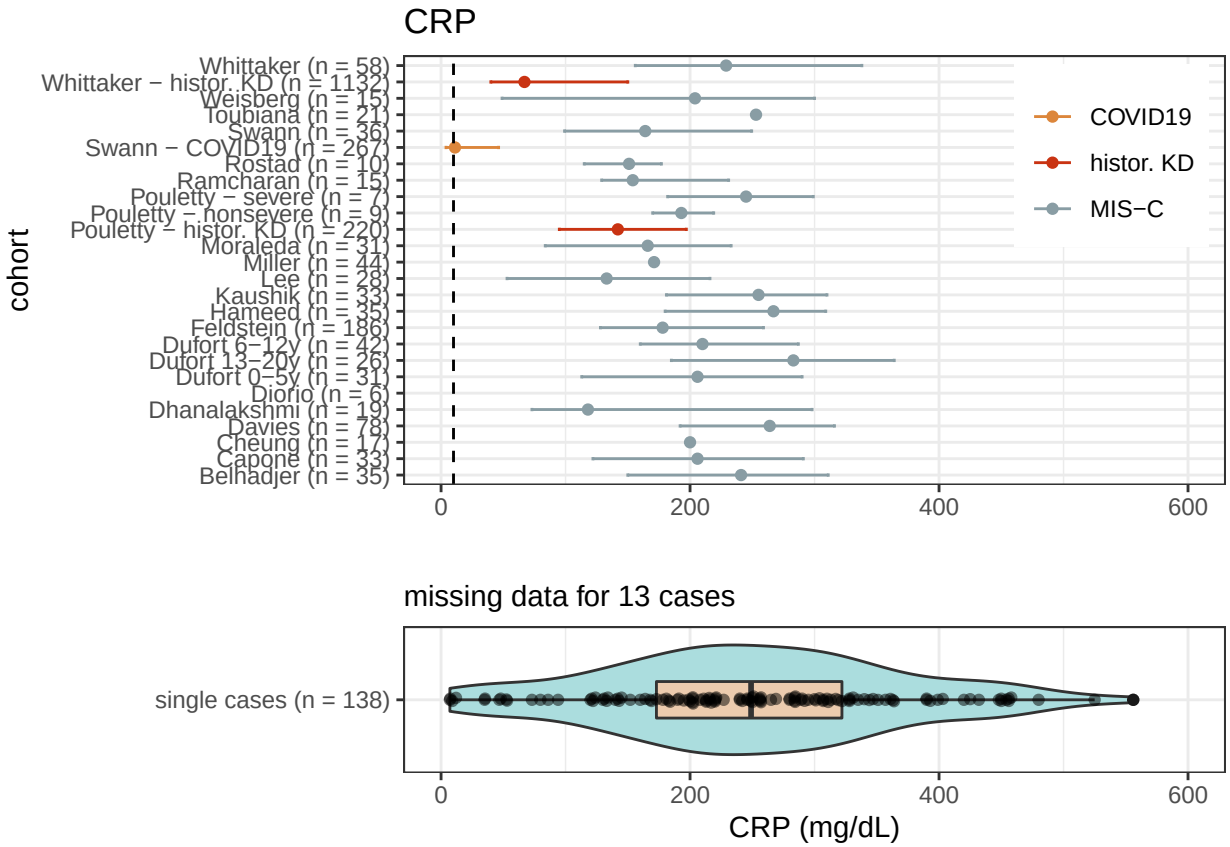


Lab values

For lab values, sometimes multiple values are reported (baseline, peak or not-specified). All lab values are collapsed based on the max (or the min for e.g. hemoglobin): so only the highest value of median, Q1 or Q3 is used. Dashed vertical line corresponds to the cutoff used in the study.

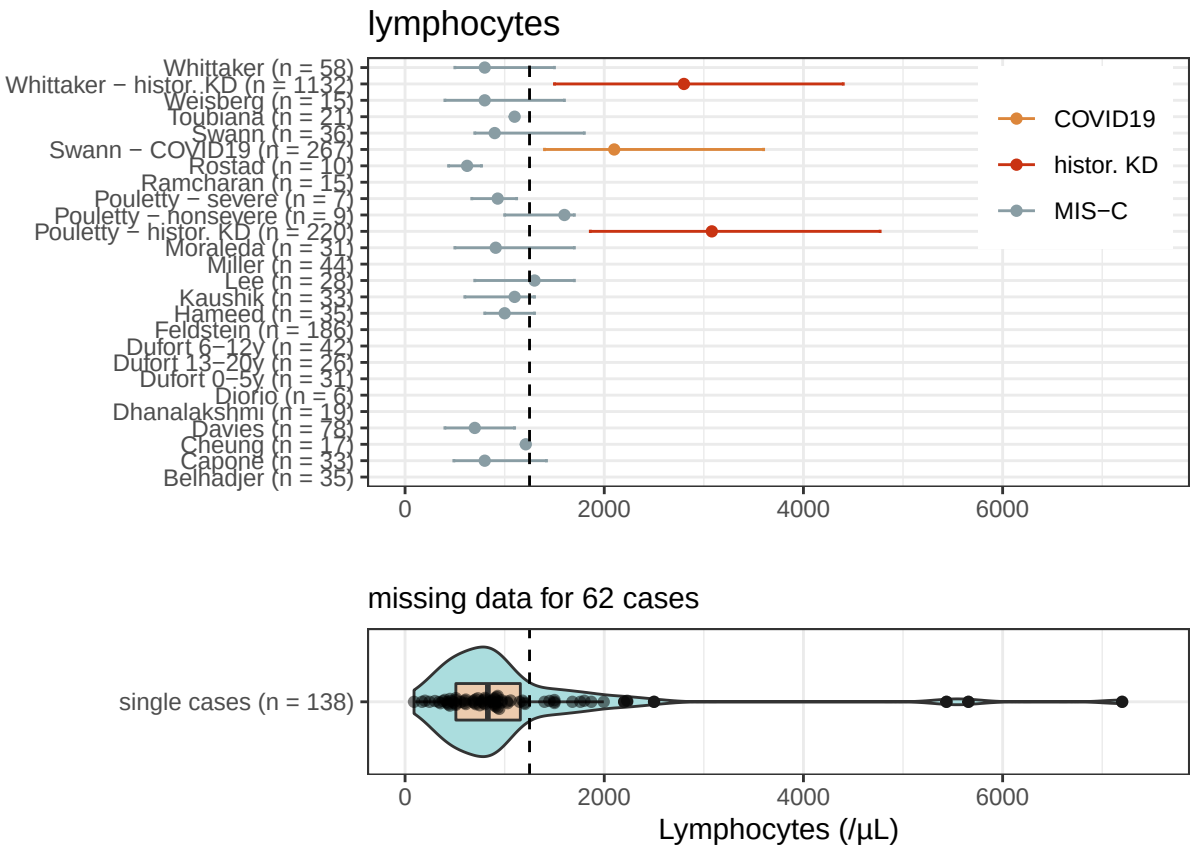
C-reactive protein

```
crp_collapse_cohort <- collapse_labvals_cohort(df_cohort_controls, "max", "CRP")
crp_collapse_single <- collapse_labvals_single(df_singlecases, "max", "CRP")
crp_missing <- sum(is.na(crp_collapse_single$CRP_max))

p_crp_cohort <- ggplot(crp_collapse_cohort, aes(y = cohort_id, x = CRP_med, col = cohort_type)) +
  geom_point() +
  geom_errorbar(aes(xmin=CRP_min, xmax=CRP_max), width=.2, position=position_dodge(.9)) + lims(x = c(0,600)) +
  theme_bw() + labs(title = "CRP", y = "cohort", x = "") +
  geom_vline(xintercept = co_CRP, linetype = "dashed", color = "black") + theme(legend.justification = c(1, 1), legend.position = c(0.98, 0.98), legend.
        title=element_blank()) +
  scale_color_manual(values = wes_palette("Royal1")[c(4,2,1)])

p_crp_single <- ggplot(crp_collapse_single, aes(x = as.numeric(CRP_max), y = cohort_id)) +
  geom_violin(fill = wes_palette("Darjeeling2")[4]) +
  geom_boxplot(width=.3, fill =  wes_palette("Darjeeling2")[1]) +
  theme_bw() + geom_beeswarm(groupOnX=FALSE, alpha = 0.5) + lims(x = c(0,600)) + labs(y = "", x = "CRP (mg/dL)", subtitle = paste0("missing data for ", crp_
        missing, " cases")) +
  geom_hline(yintercept = co_CRP, linetype = "dashed", color = "black")

CRP_grid <- plot_grid(p_crp_cohort, p_crp_single, align = "v", nrow = 2, rel_heights = c(2/3, 1/3))
CRP_grid
```



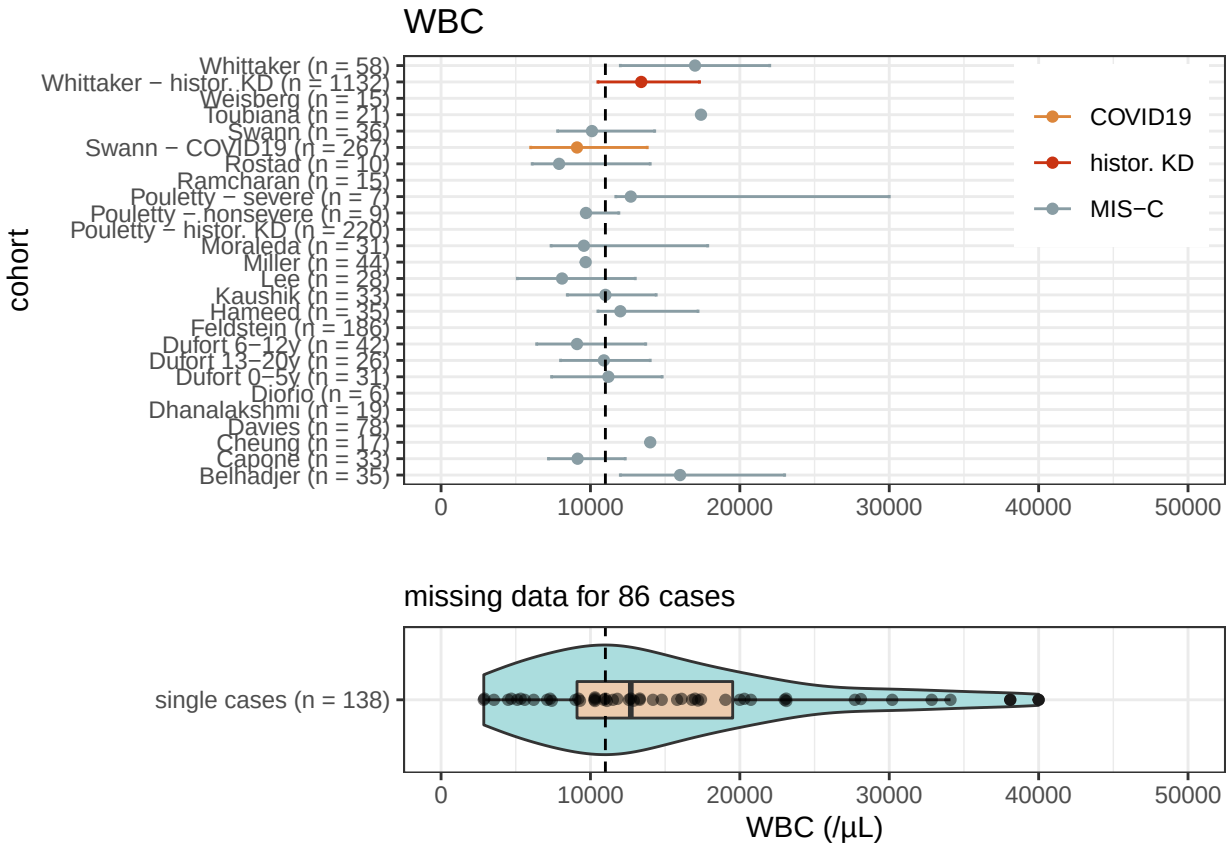
Lymphocytes

```
lympho_collapse_cohort <- collapse_labvals_cohort(df_cohort_controls, "min", "lympho")
lympho_collapse_single <- collapse_labvals_single(df_singlecases, "min", "lympho")
lympho_missing <- sum(is.na(lympho_collapse_single$lympho_min))

p_lympho_cohort <- ggplot(lympho_collapse_cohort, aes(y = cohort_id, x = lympho_med, col = cohort_type)) +
  geom_point() +
  geom_errorbar(aes(xmin=lympho_min, xmax=lympho_max), width=.2, position=position_dodge(.9)) +
  theme_bw() + labs(title = "lymphocytes", y = "", x = "") + lims(x = c(0,7500))  +
  geom_vline(xintercept = co_lympho, linetype = "dashed", color = "black") + theme(legend.justification = c(1, 1), legend.position = c(0.98, 0.98), legend.
        title=element_blank()) +
  scale_color_manual(values = wes_palette("Royal1")[c(4,2,1)])#+
#rremove("y.text")

p_lympho_single <- ggplot(lympho_collapse_single, aes(x = as.numeric(lympho_min), y = cohort_id)) +
  geom_violin(fill = wes_palette("Darjeeling2")[4]) +
  geom_boxplot(width=.3, fill = wes_palette("Darjeeling2")[1]) +
  lims(x = c(0,7500))+
  theme_bw() + geom_beeswarm(groupOnX=FALSE, alpha = 0.5)  + labs(y = "", x = "Lymphocytes (/µL)", subtitle = paste0("missing data for ", lympho_missing, "
        cases")) +
  geom_vline(xintercept = co_lympho, linetype = "dashed", color = "black") #+
#rremove("y.text")

lympho_grid <- plot_grid(p_lympho_cohort, p_lympho_single, align = "v", nrow = 2, rel_heights = c(2/3, 1/3))
lympho_grid
```



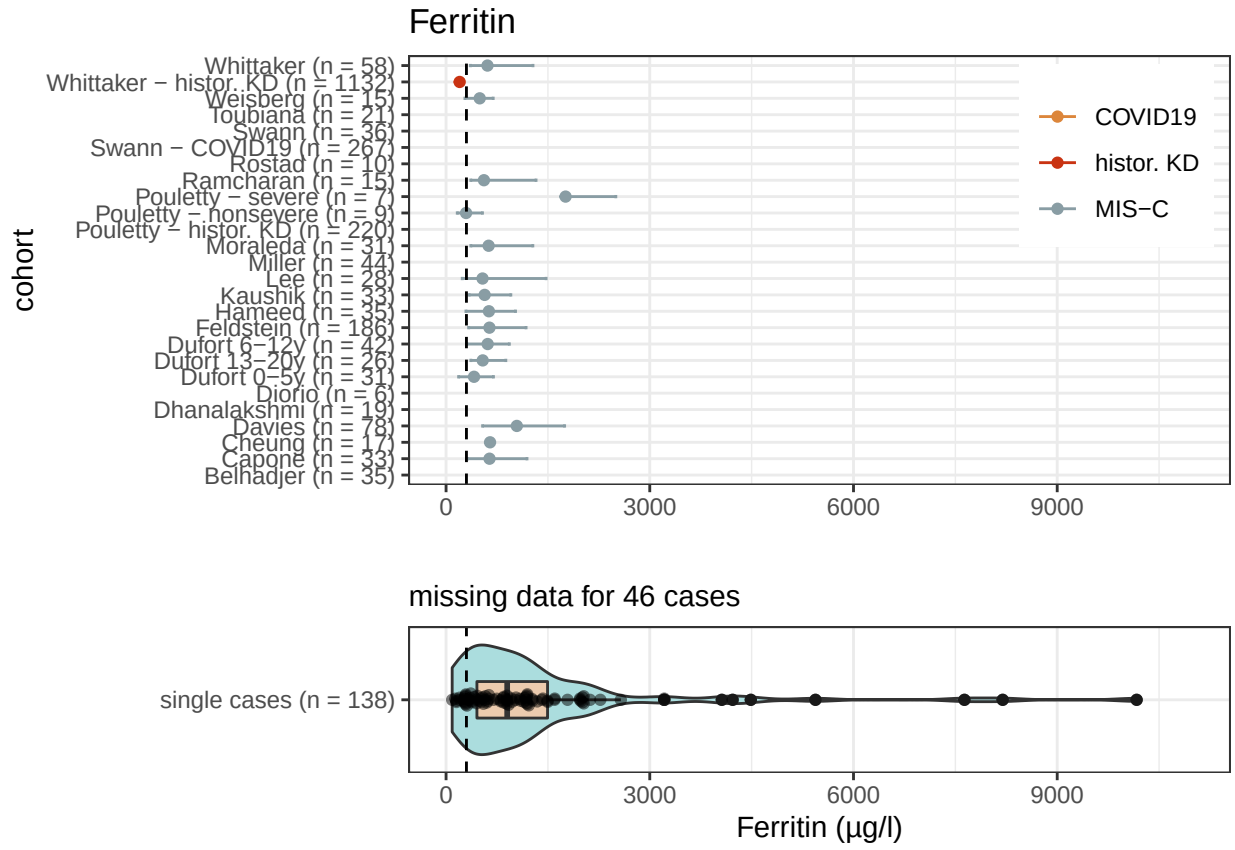
White blood cells

```
wbc_collapse_cohort <- collapse_labvals_cohort(df_cohort_controls, "max", "WBC")
wbc_collapse_single <- collapse_labvals_single(df_singlecases, "max", "WBC")
wbc_missing <- sum(is.na(wbc_collapse_single$WBC_max))

p_wbc_cohort <- ggplot(wbc_collapse_cohort, aes(y = cohort_id, x = WBC_med, col = cohort_type)) +
  geom_point() +
  geom_errorbar(aes(xmin=WBC_min, xmax=WBC_max), width=.2, position=position_dodge(.9)) + lims(x = c(0,50000)) +
  theme_bw() + labs(title = "WBC", y = "cohort", x = "")  +
  geom_vline(xintercept = co_WBC, linetype = "dashed", color = "black")  + theme(legend.justification = c(1, 1), legend.position = c(0.98, 0.98), legend.
        title=element_blank()) +
  scale_color_manual(values = wes_palette("Royal1")[c(4,2,1)])

p_wbc_single <- ggplot(wbc_collapse_single, aes(x = as.numeric(WBC_max), y = cohort_id)) +
  geom_violin(fill = wes_palette("Darjeeling2")[4]) +
  geom_boxplot(width=.3, fill = wes_palette("Darjeeling2")[1]) +
  theme_bw() + geom_beeswarm(groupOnX=FALSE, alpha = 0.5) + labs(y = "", x = "WBC (/µL)", subtitle = paste0("missing data for ", wbc_missing, " cases")) +
        lims(x = c(0,50000)) +
  geom_vline(xintercept = co_WBC, linetype = "dashed", color = "black")

WBC_grid <- plot_grid(p_wbc_cohort, p_wbc_single, align = "v", nrow = 2, rel_heights = c(2/3, 1/3))
WBC_grid
```



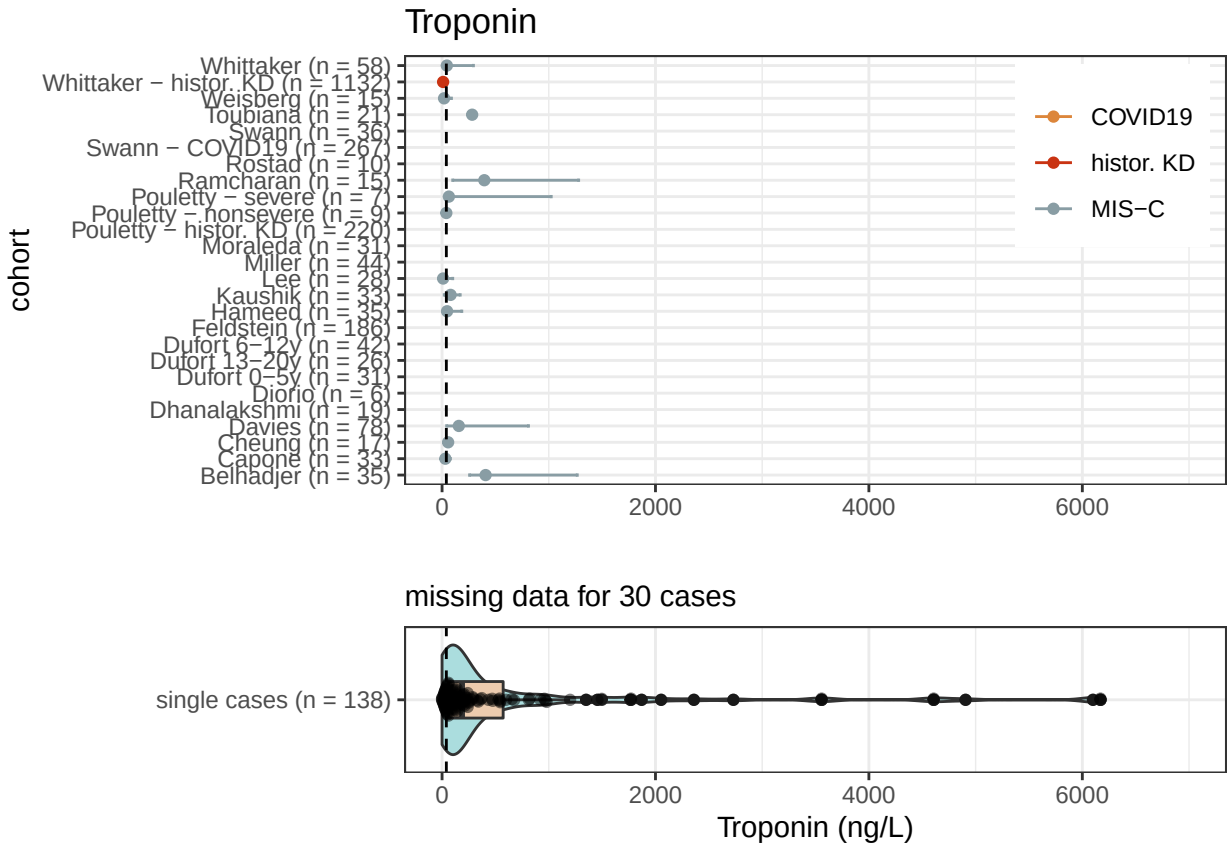
Ferritin

```
ferritin_collapse_cohort <- collapse_labvals_cohort(df_cohort_controls, "max", "ferrit")
ferritin_collapse_single <- collapse_labvals_single(df_singlecases, "max", "ferrit")
ferritin_missing <- sum(is.na(ferritin_collapse_single$ferrit_max))

p_ferritin_cohort <- ggplot(ferritin_collapse_cohort, aes(y = cohort_id, x = ferrit_med, col = cohort_type)) +
  geom_point() +
  geom_errorbar(aes(xmin=ferrit_min, xmax=ferrit_max), width=.2, position=position_dodge(.9)) + lims(x = c(0,11000)) +
  theme_bw() + labs(title = "Ferritin", y = "cohort", x = "") +
  geom_vline(xintercept = co_ferritin, linetype = "dashed", color = "black") + theme(legend.justification = c(1, 1), legend.position = c(0.98, 0.98), legend
      .title=element_blank()) +
  scale_color_manual(values = wes_palette("Royal1")[c(4,2,1)])

p_ferritin_single <- ggplot(ferritin_collapse_single, aes(x = as.numeric(ferrit_max), y = cohort_id)) +
  geom_violin(fill = wes_palette("Darjeeling2")[4]) +
  geom_boxplot(width=.3, fill = wes_palette("Darjeeling2")[1]) +
  theme_bw() + geom_beeswarm(groupOnX=FALSE, alpha = 0.5) + labs(y = "", x = "Ferritin (µg/l)", subtitle = paste0("missing data for ", ferritin_missing, "
      cases")) + lims(x = c(0,11000)) +
  geom_vline(xintercept = co_ferritin, linetype = "dashed", color = "black")

ferritin_grid <- plot_grid(p_ferritin_cohort, p_ferritin_single, align = "v", nrow = 2, rel_heights = c(2/3, 1/3))
ferritin_grid
```



Troponin

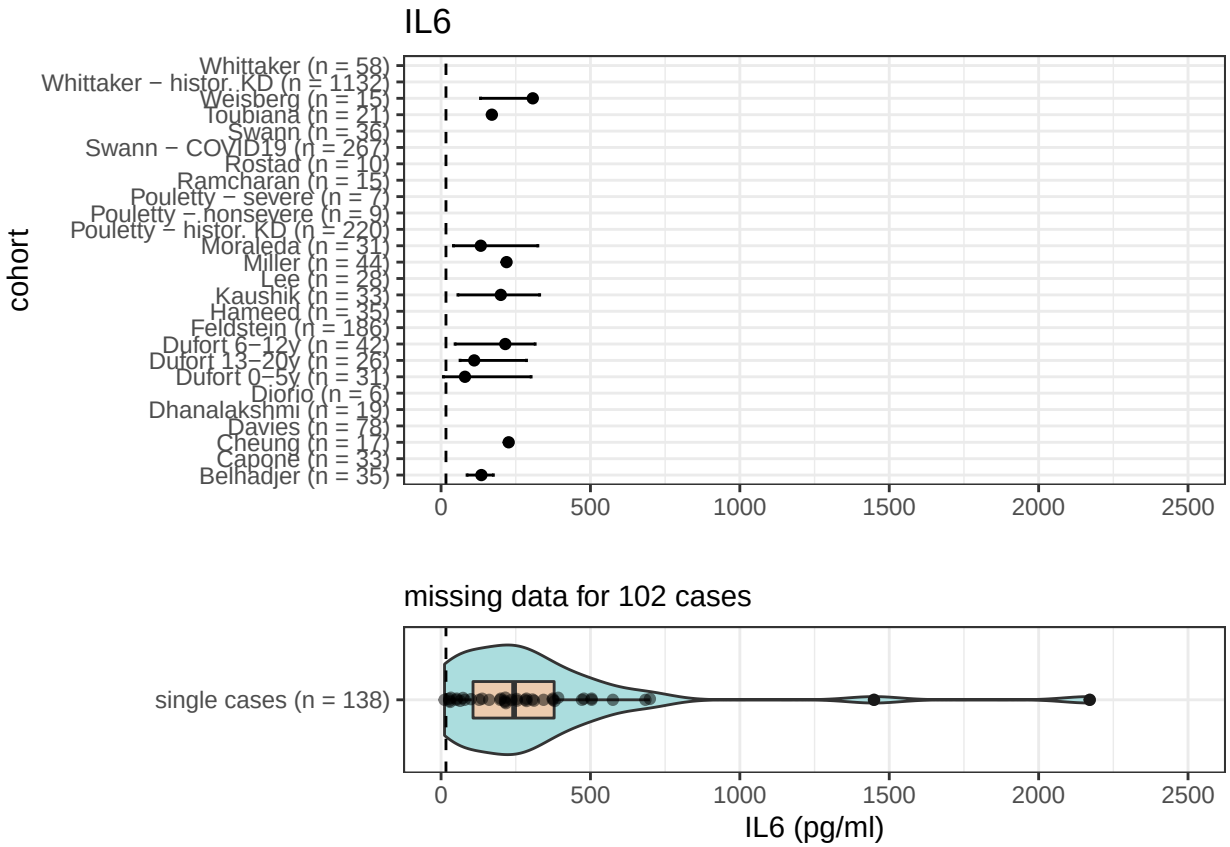
```
troponin_collapse_cohort <- collapse_labvals_cohort(df_cohort_controls, "max", "troponin")
troponin_collapse_single <- collapse_labvals_single(df_singlecases, "max", "troponin")
troponin_missing <- sum(is.na(troponin_collapse_single$troponin_max))

p_troponin_cohort <- ggplot(troponin_collapse_cohort, aes(y = cohort_id, x = troponin_med, col = cohort_type)) +
  geom_point() +
  geom_errorbar(aes(xmin=troponin_min, xmax=troponin_max), width=.2, position=position_dodge(.9)) + lims(x = c(0,7000)) +
  theme_bw() + labs(title = "Troponin", y = "cohort", x = "")  +
  geom_vline(xintercept = co_tropo, linetype = "dashed", color = "black")  + theme(legend.justification = c(1, 1), legend.position = c(0.98, 0.98), legend.
        title=element_blank()) +
  scale_color_manual(values = wes_palette("Royal1")[c(4,2,1)])

p_troponin_single <- ggplot(troponin_collapse_single, aes(x = as.numeric(troponin_max), y = cohort_id)) +
  geom_violin(fill = wes_palette("Darjeeling2")[4]) +
  geom_boxplot(width=.3, fill = wes_palette("Darjeeling2")[1]) +
  theme_bw() + geom_beeswarm(groupOnX=FALSE, alpha = 0.5) + labs(y = "", x = "Troponin (ng/L)", subtitle = paste0("missing data for ", troponin_missing, "
        cases")) + lims(x = c(0,7000)) +
  geom_vline(xintercept = co_tropo, linetype = "dashed", color = "black")

troponin_grid <- plot_grid(p_troponin_cohort, p_troponin_single, align = "v", nrow = 2, rel_heights = c(2/3, 1/3))
troponin_grid
```



IL-6

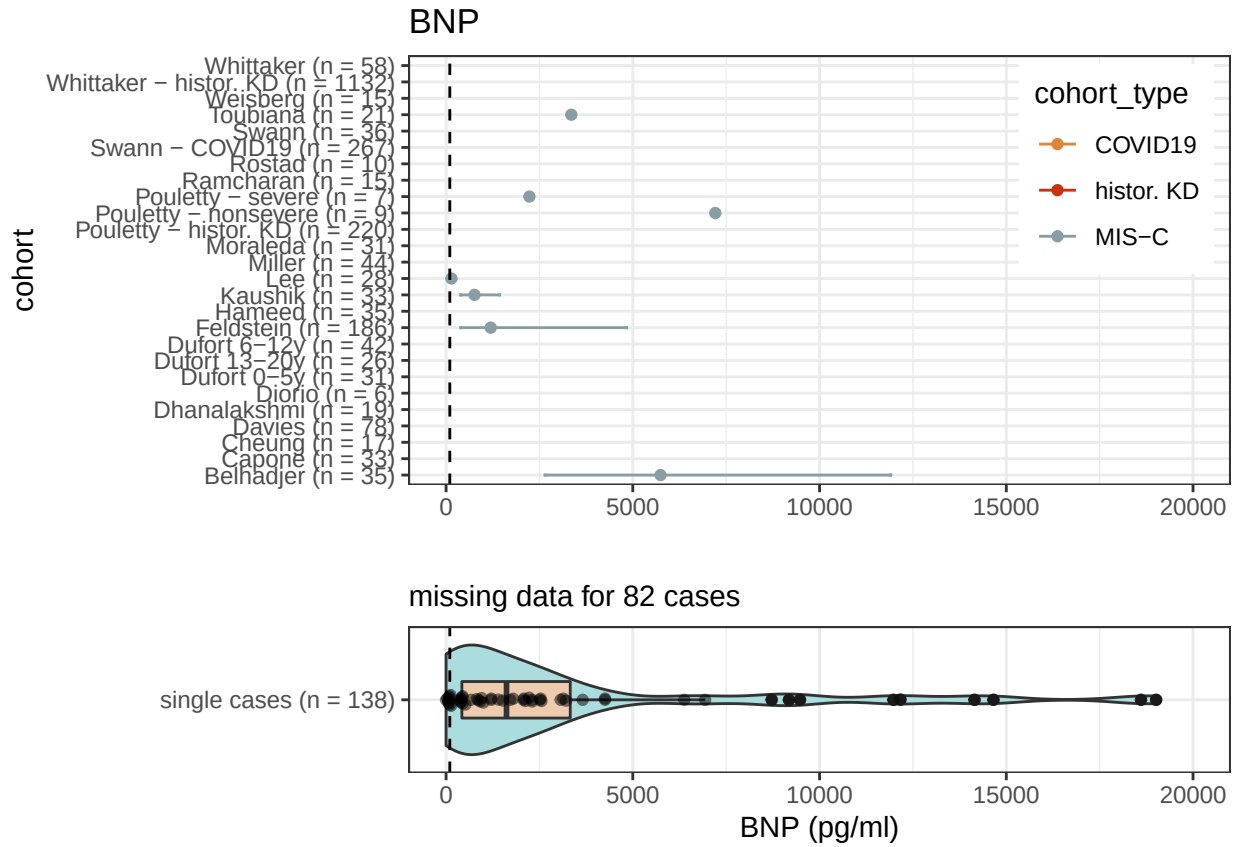
Note: The cases from Pouletty et al are added to the single cases as they report on IL6 values.

```
IL6_collapse_cohort <- collapse_labvals_cohort(df_cohort_controls, "max", "IL6")
IL6_collapse_single <- collapse_labvals_single(df_singlecases_inclPouletty, "max", "IL6")
IL6_missing <- sum(is.na(IL6_collapse_single$IL6_max))

p_IL6_cohort <- ggplot(IL6_collapse_cohort, aes(y = cohort_id, x = IL6_med)) +
  geom_point() +
  geom_errorbar(aes(xmin=IL6_min, xmax=IL6_max), width=.2, position=position_dodge(.9)) + lims(x = c(0,2500)) +
  theme_bw() + labs(title = "IL6", y = "cohort", x = "")  +
  geom_vline(xintercept = co_IL6, linetype = "dashed", color = "black")  +
  scale_color_manual(values = wes_palette("Royal1")[c(4,2,1)])

p_IL6_single <- ggplot(IL6_collapse_single, aes(x = as.numeric(IL6_max), y = cohort_id)) +
  geom_violin(fill = wes_palette("Darjeeling2")[4]) +
  geom_boxplot(width=.3, fill = wes_palette("Darjeeling2")[1]) +
  theme_bw() + geom_beeswarm(groupOnX=FALSE, alpha = 0.5) + labs(y = "", x = "IL6 (pg/ml)", subtitle = paste0("missing data for ", IL6_missing, " cases")) +
        lims(x = c(0,2500)) +
  geom_vline(xintercept = co_IL6, linetype = "dashed", color = "black")

IL6_grid <- plot_grid(p_IL6_cohort, p_IL6_single, align = "v", nrow = 2, rel_heights = c(2/3, 1/3))
IL6_grid
```



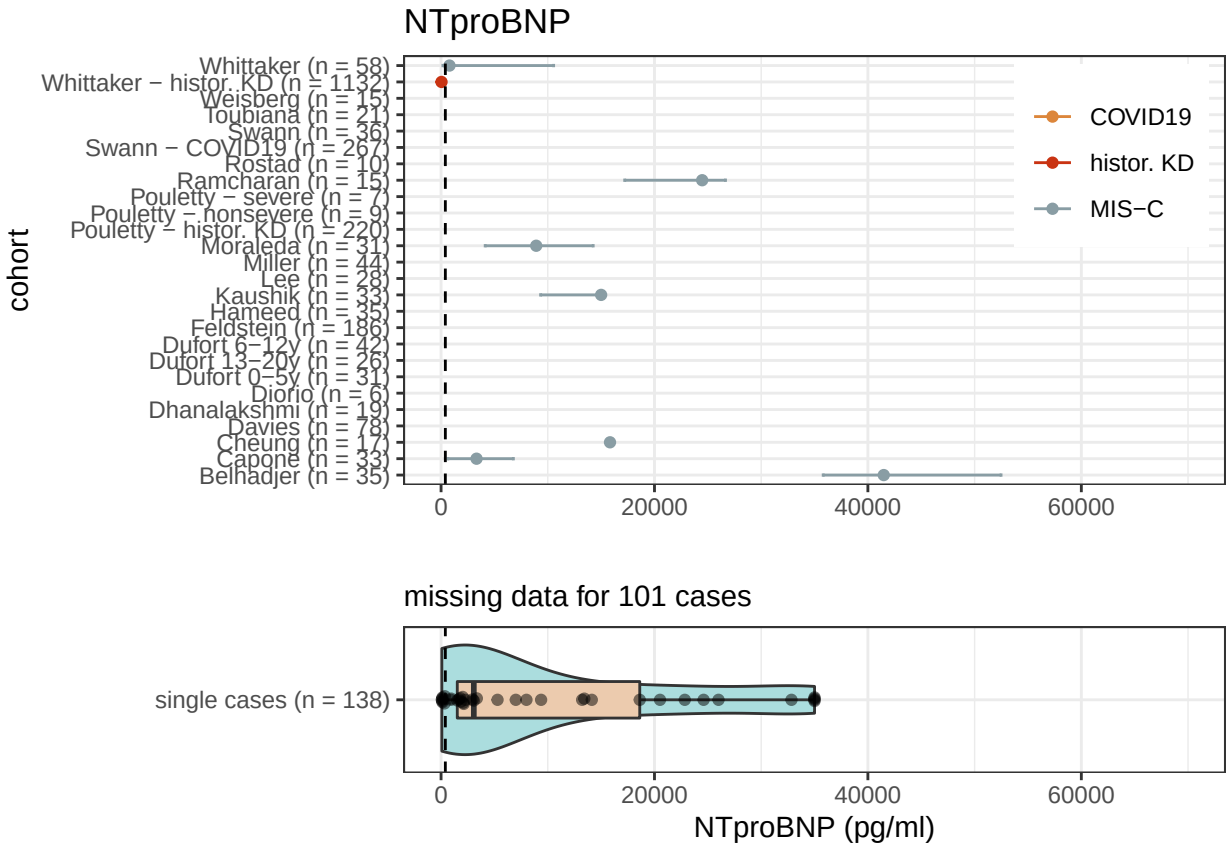
BNP

```
collapse_cohort <- collapse_labvals_cohort(df_cohort_controls, "max", "_BNP")
collapse_single <- collapse_labvals_single(df_singlecases, "max", "_BNP")
missing <- sum(is.na(collapse_single$`_BNP_max`))

p_BNP_cohort <- ggplot(collapse_cohort, aes(y = cohort_id, x = `_BNP_med`, col = cohort_type)) +
  geom_point() +
  geom_errorbar(aes(xmin=`_BNP_min`, xmax=`_BNP_max`), width=.2, position=position_dodge(.9)) + lims(x = c(0,20000)) +
  theme_bw() + labs(title = "BNP", y = "cohort", x = "")  +
  geom_vline(xintercept = co_BNP, linetype = "dashed", color = "black")  +theme(legend.justification = c(1, 1), legend.position = c(0.98, 0.98)) +
  scale_color_manual(values = wes_palette("Royal1")[c(4,2,1)])

p_BNP_single <- ggplot(collapse_single, aes(x = as.numeric(`_BNP_max`), y = cohort_id)) +
  geom_violin(fill = wes_palette("Darjeeling2")[4]) +
  geom_boxplot(width=.3, fill = wes_palette("Darjeeling2")[1]) +
  theme_bw() + geom_beeswarm(groupOnX=FALSE, alpha = 0.5) + labs(y = "", x = "BNP (pg/ml)", subtitle = paste0("missing data for ", missing, " cases")) +
        lims(x = c(0,20000)) +
  geom_vline(xintercept = co_BNP, linetype = "dashed", color = "black")

BNP_grid <- plot_grid(p_BNP_cohort, p_BNP_single, align = "v", nrow = 2, rel_heights = c(2/3, 1/3))
BNP_grid
```



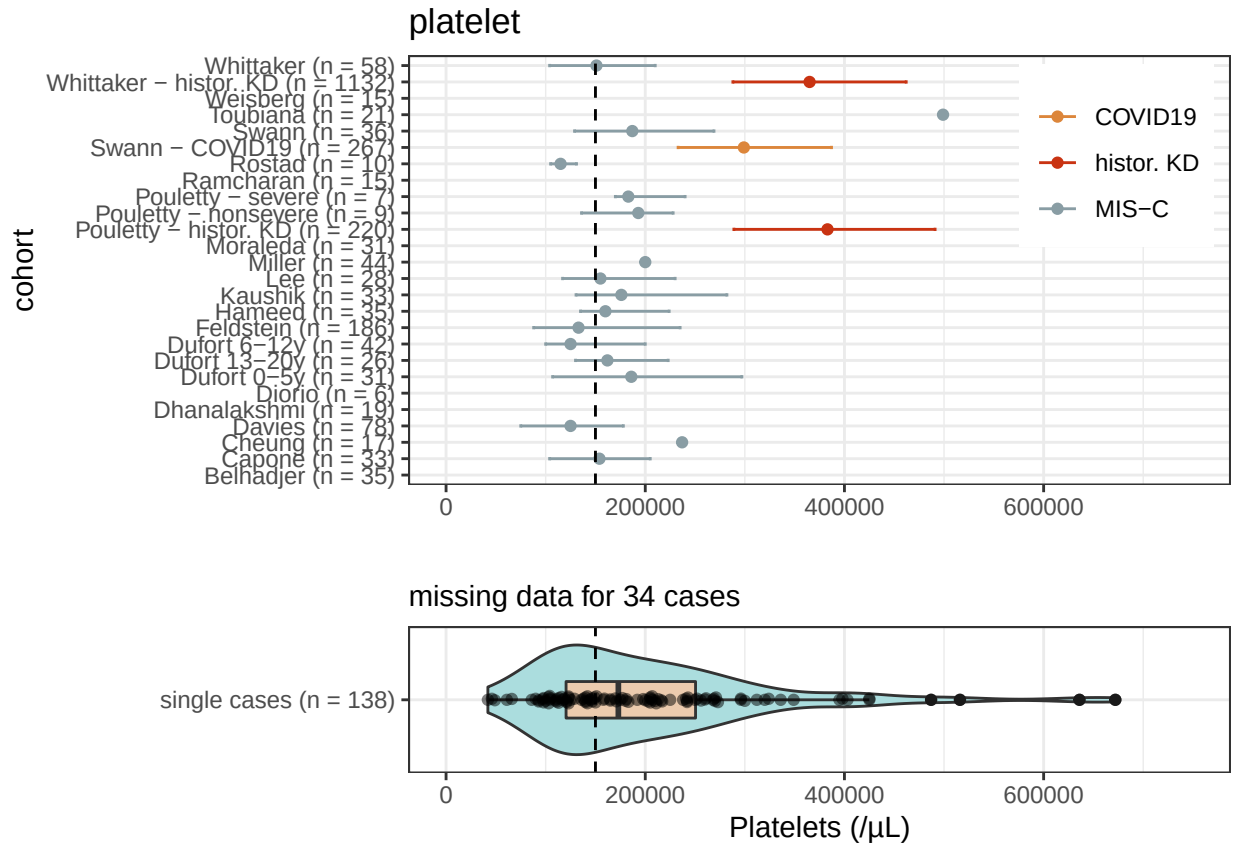
NTproBNP

```
collapse_cohort <- collapse_labvals_cohort(df_cohort_controls, "max", "NTproBNP")
collapse_single <- collapse_labvals_single(df_singlecases, "max", "NTproBNP")
missing <- sum(is.na(collapse_single$NTproBNP_max))

p_NTproBNP_cohort <- ggplot(collapse_cohort, aes(y = cohort_id, x = NTproBNP_med, col = cohort_type)) +
  geom_point() +
  geom_errorbar(aes(xmin=NTproBNP_min, xmax=NTproBNP_max), width=.2, position=position_dodge(.9)) + lims(x = c(0,70000)) +
  theme_bw() + labs(title = "NTproBNP", y = "cohort", x = "")  +
  geom_vline(xintercept = co_NTproBNP, linetype = "dashed", color = "black") + theme(legend.justification = c(1, 1), legend.position = c(0.98, 0.98), legend
        .title=element_blank()) +
  scale_color_manual(values = wes_palette("Royal1")[c(4,2,1)])

p_NTproBNP_single <- ggplot(collapse_single, aes(x = as.numeric(NTproBNP_max), y = cohort_id)) +
  geom_violin(fill = wes_palette("Darjeeling2")[4]) +
  geom_boxplot(width=.3, fill = wes_palette("Darjeeling2")[1]) +
  theme_bw() + geom_beeswarm(groupOnX=FALSE, alpha = 0.5) + labs(y = "", x = "NTproBNP (pg/ml)", subtitle = paste0("missing data for ", missing, " cases"))
        + lims(x = c(0,70000)) +
  geom_vline(xintercept = co_NTproBNP, linetype = "dashed", color = "black")

NTproBNP_grid <- plot_grid(p_NTproBNP_cohort, p_NTproBNP_single, align = "v", nrow = 2, rel_heights = c(2/3, 1/3))
NTproBNP_grid
```



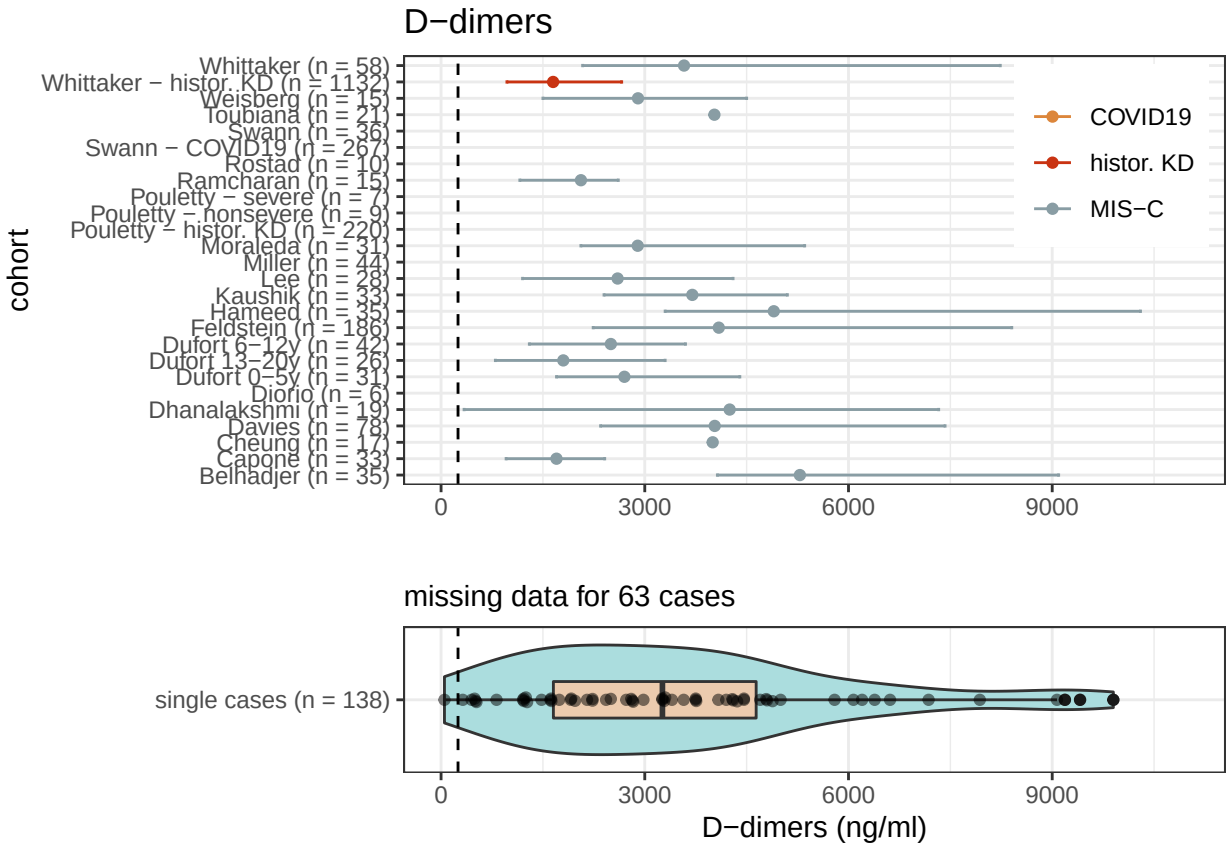
Platelets

```
collapse_cohort <- collapse_labvals_cohort(df_cohort_controls, "min", "platelet")
collapse_single <- collapse_labvals_single(df_singlecases, "min", "platelet")
missing <- sum(is.na(collapse_single$platelet_min))

p_platelet_cohort <- ggplot(collapse_cohort, aes(y = cohort_id, x = platelet_med, col = cohort_type)) +
  geom_point() +
  geom_errorbar(aes(xmin=platelet_min, xmax=platelet_max, col=cohort_type), width=.2, position=position_dodge(.9)) + lims(x = c(0,750000)) +
  theme_bw() + labs(title = "platelet", y = "cohort", x = "")  +
  geom_vline(xintercept = co_platelet, linetype = "dashed", color = "black")  + theme(legend.justification = c(1, 1), legend.position = c(0.98, 0.98),
        legend.title=element_blank()) +
  scale_color_manual(values = wes_palette("Royal1")[c(4,2,1)])

p_platelet_single <- ggplot(collapse_single, aes(x = as.numeric(platelet_min), y = cohort_id)) +
  geom_violin(fill = wes_palette("Darjeeling2")[4]) +
  geom_boxplot(width=.3, fill = wes_palette("Darjeeling2")[1]) +
  theme_bw() + geom_beeswarm(groupOnX=FALSE, alpha = 0.5) + labs(y = "", x = "Platelets (/µL)", subtitle = paste0("missing data for ", missing, " cases")) +
        lims(x = c(0,750000)) +
  geom_vline(xintercept = co_platelet, linetype = "dashed", color = "black")

platelet_grid <- plot_grid(p_platelet_cohort, p_platelet_single, align = "v", nrow = 2, rel_heights = c(2/3, 1/3))
platelet_grid
```



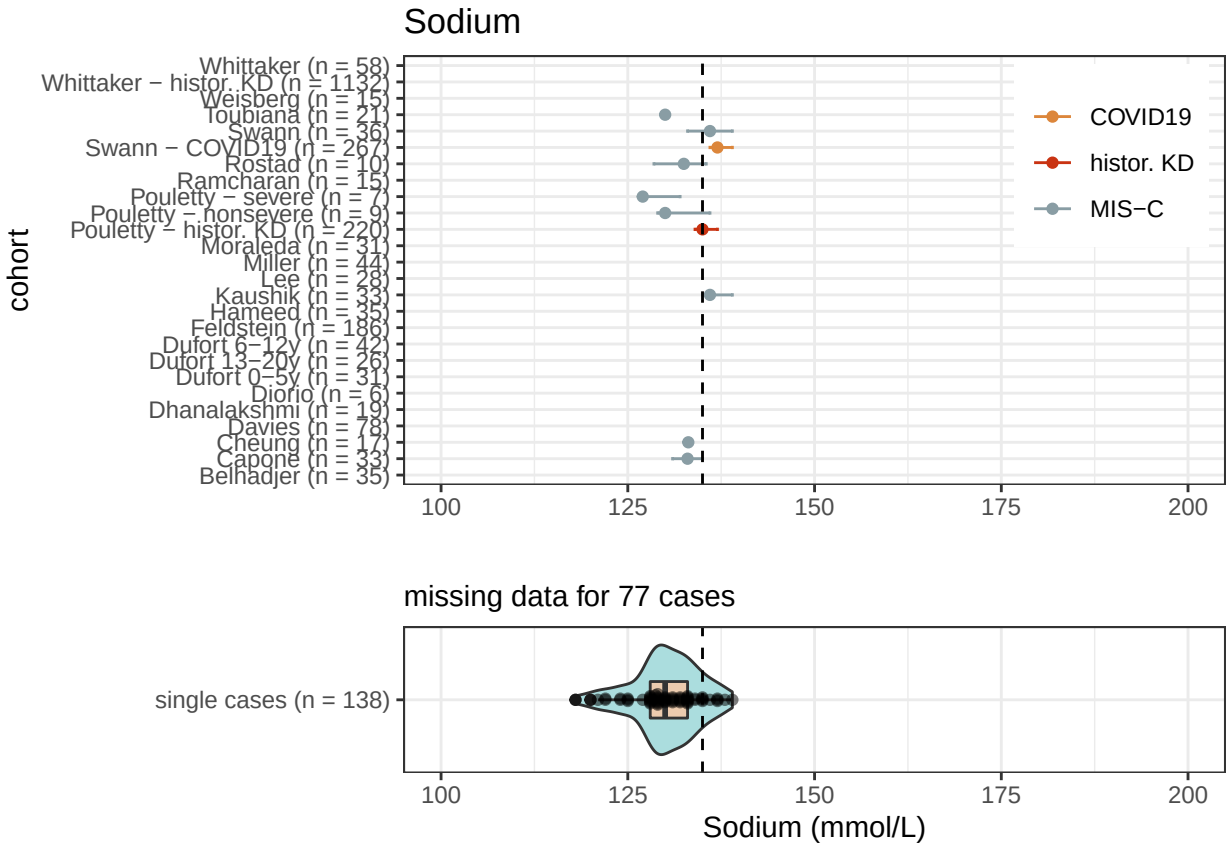
D-dimers

```
collapse_cohort <- collapse_labvals_cohort(df_cohort_controls, "max", "Ddim")
collapse_single <- collapse_labvals_single(df_singlecases, "max", "Ddim")
missing <- sum(is.na(collapse_single$Ddim_max))

p_Ddim_cohort <- ggplot(collapse_cohort, aes(y = cohort_id, x = Ddim_med, col = cohort_type)) +
  geom_point() +
  geom_errorbar(aes(xmin=Ddim_min, xmax=Ddim_max, col=cohort_type), width=.2, position=position_dodge(.9)) + lims(x = c(0,11000)) +
  theme_bw() + labs(title = "D-dimers", y = "cohort", x = "")  +
  geom_vline(xintercept = co_Ddim, linetype = "dashed", color = "black")  + theme(legend.justification = c(1, 1), legend.position = c(0.98, 0.98), legend.
        title=element_blank()) +
  scale_color_manual(values = wes_palette("Royal1")[c(4,2,1)])

p_Ddim_single <- ggplot(collapse_single, aes(x = as.numeric(Ddim_max), y = cohort_id)) +
  geom_violin(fill = wes_palette("Darjeeling2")[4]) +
  geom_boxplot(width=.3, fill = wes_palette("Darjeeling2")[1]) +
  theme_bw() + geom_beeswarm(groupOnX=FALSE, alpha = 0.5) + labs(y = "", x = "D-dimers (ng/ml)", subtitle = paste0("missing data for ", missing, " cases"))
        + lims(x = c(0,11000)) +
  geom_vline(xintercept = co_Ddim, linetype = "dashed", color = "black")

Ddim_grid <- plot_grid(p_Ddim_cohort, p_Ddim_single, align = "v", nrow = 2, rel_heights = c(2/3, 1/3))
Ddim_grid
```



Sodium

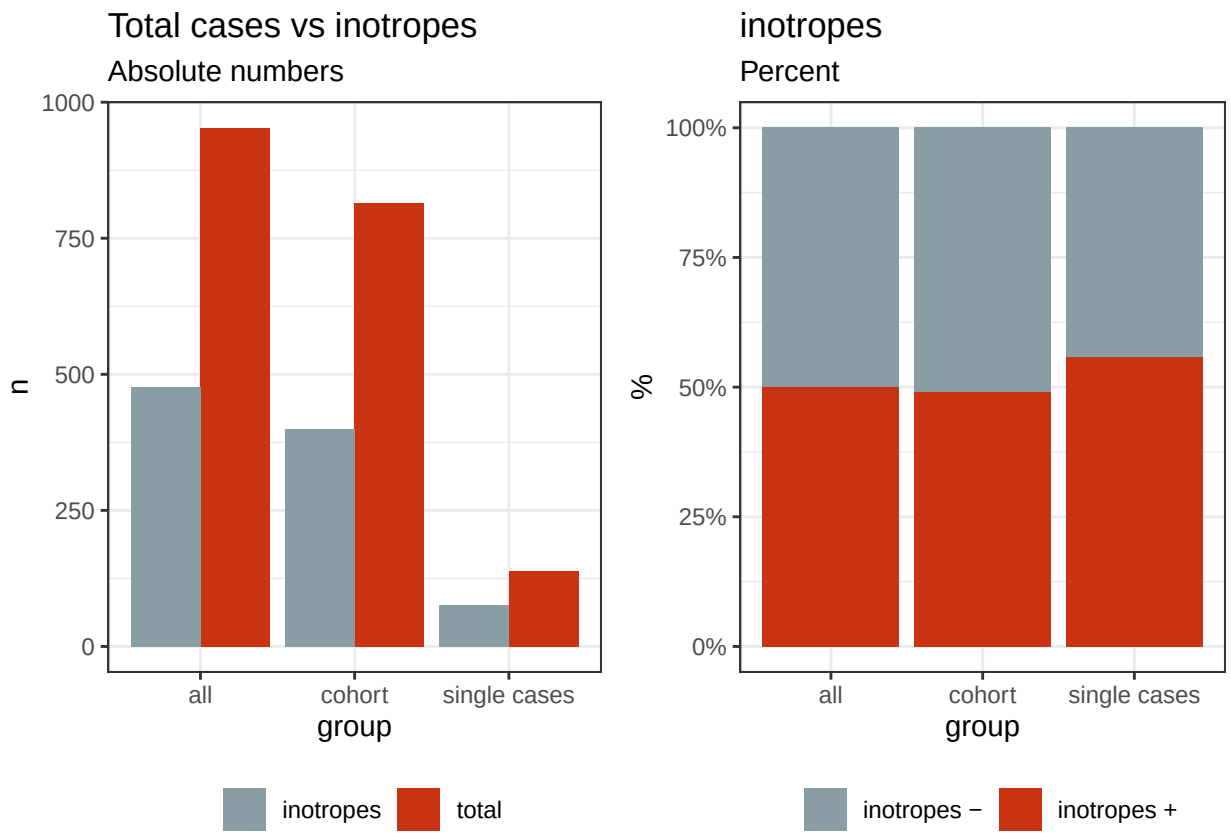
```r
collapse_cohort <- collapse_labvals_cohort(df_cohort_controls, "min", "sodium")
collapse_single <- collapse_labvals_single(df_singlecases, "min", "sodium")
missing <- sum(is.na(collapse_single$sodium_min))

p_sodium_cohort <- ggplot(collapse_cohort, aes(y = cohort_id, x = sodium_med, col = cohort_type)) +
  geom_point() +
  geom_errorbar(aes(xmin=sodium_min, xmax=sodium_max, col=cohort_type), width=.2, position=position_dodge(.9)) + lims(x = c(100,200)) +
  theme_bw() + labs(title = "Sodium", y = "cohort", x = "")  +
  geom_vline(xintercept = co_sodium, linetype = "dashed", color = "black")  + theme(legend.justification = c(1, 1), legend.position = c(0.98, 0.98), legend.
        title=element_blank()) +
  scale_color_manual(values = wes_palette("Royal1")[c(4,2,1)])

p_sodium_single <- ggplot(collapse_single, aes(x = as.numeric(sodium_min), y = cohort_id)) +
  geom_violin(fill = wes_palette("Darjeeling2")[4]) +
  geom_boxplot(width=.3, fill = wes_palette("Darjeeling2")[1]) +
  theme_bw() + geom_beeswarm(groupOnX=FALSE, alpha = 0.5) + labs(y = "", x = "Sodium (mmol/L)", subtitle = paste0("missing data for ", missing, " cases")) +
         lims(x = c(100,200)) +
  geom_vline(xintercept = co_sodium, linetype = "dashed", color = "black")

sodium_grid <- plot_grid(p_sodium_cohort, p_sodium_single, align = "v", nrow = 2, rel_heights = c(2/3, 1/3))
sodium_grid
```



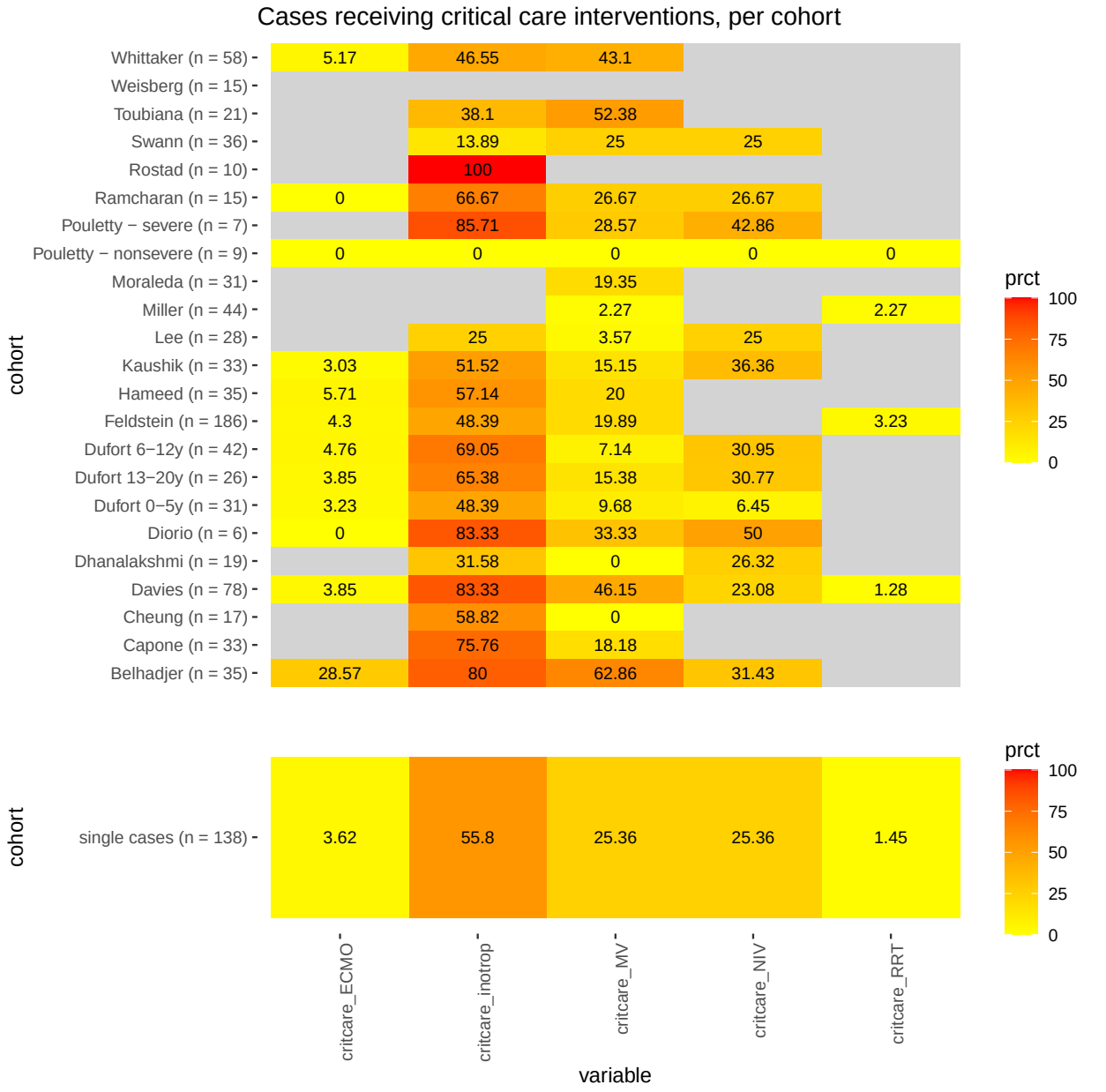
Critical care interventions

Inotropes

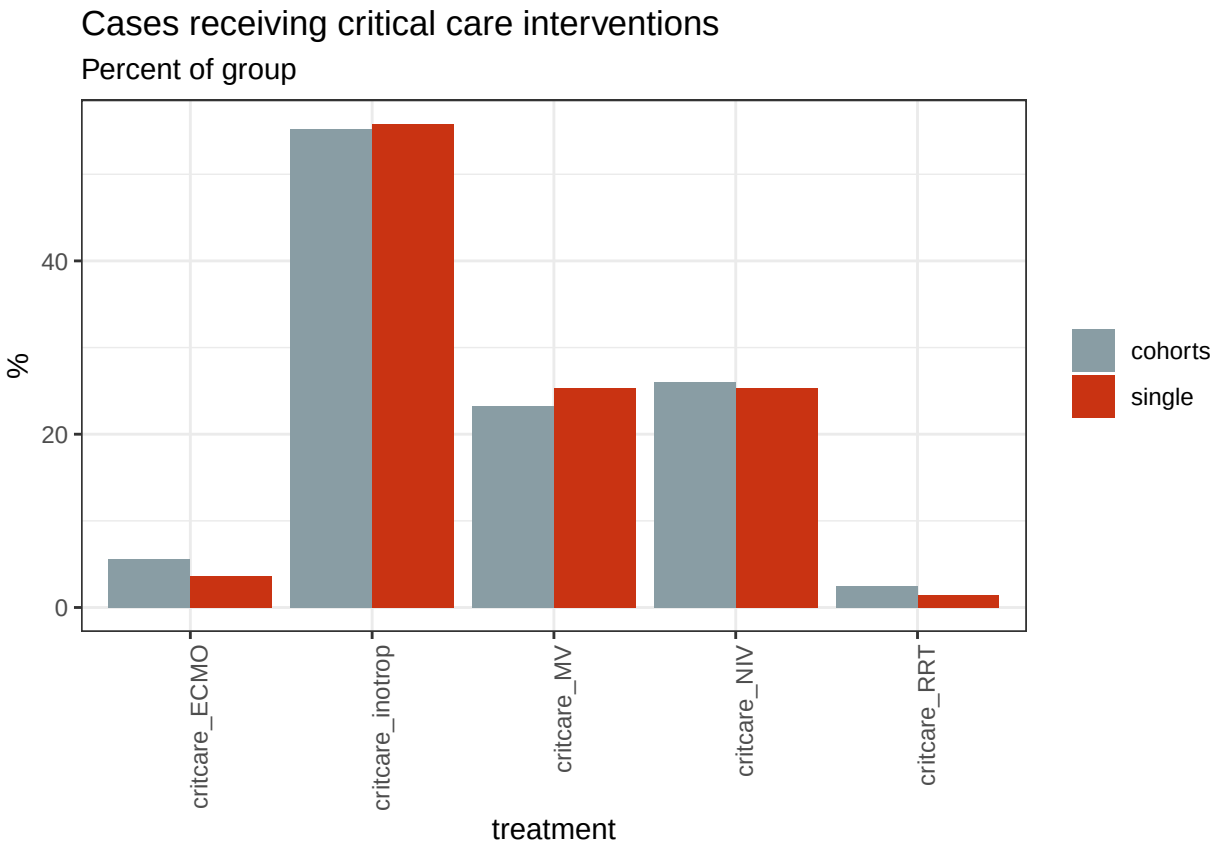
```
makeBarplot(var_id_cohort = "critcare_inotrop_n", var_id_single = "critcare_inotrop", var_id = "inotropes")
```




```
makeHeatmap_cohort("critcare", "critcare",exclude_single = "days", plottitle = "Cases receiving critical care interventions, per cohort")
```



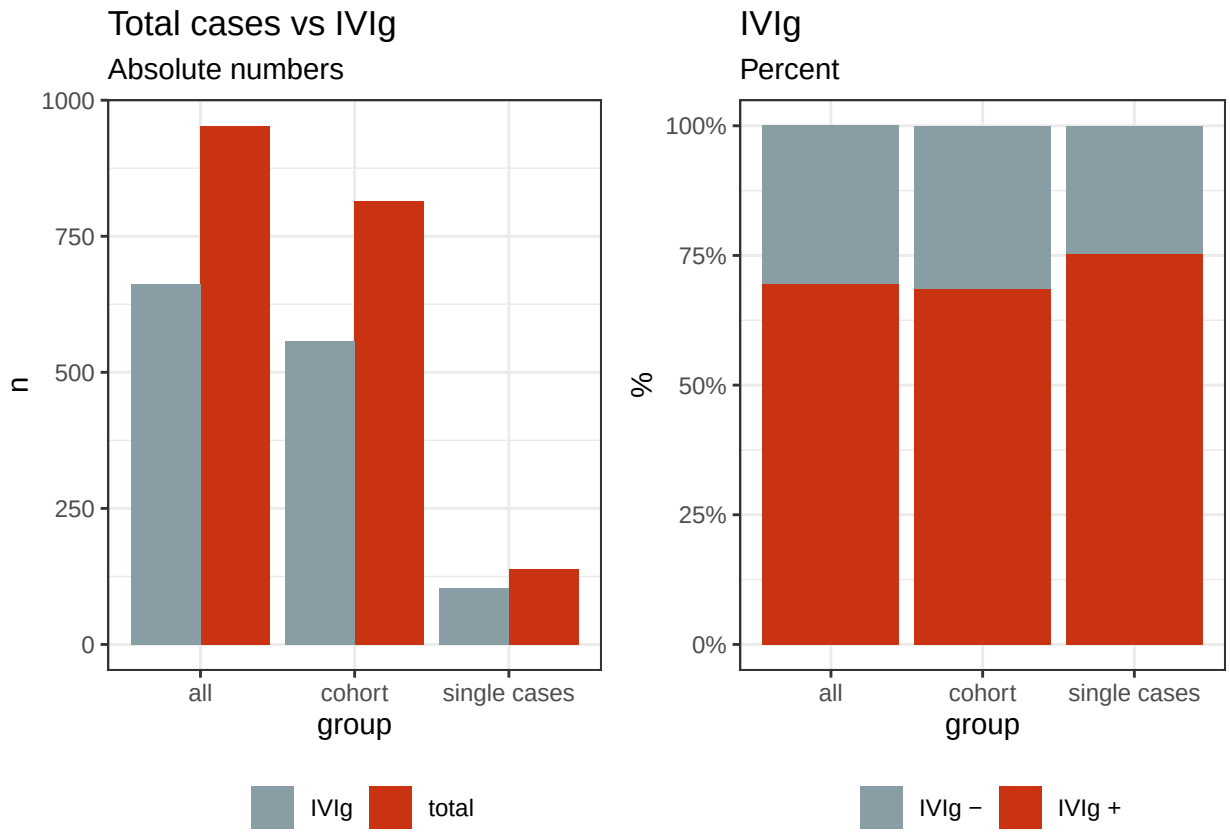

```
barSymp("critcare", "critcare", exclude_single = "days", plottitle = "Cases receiving critical care interventions")
```



Treatments

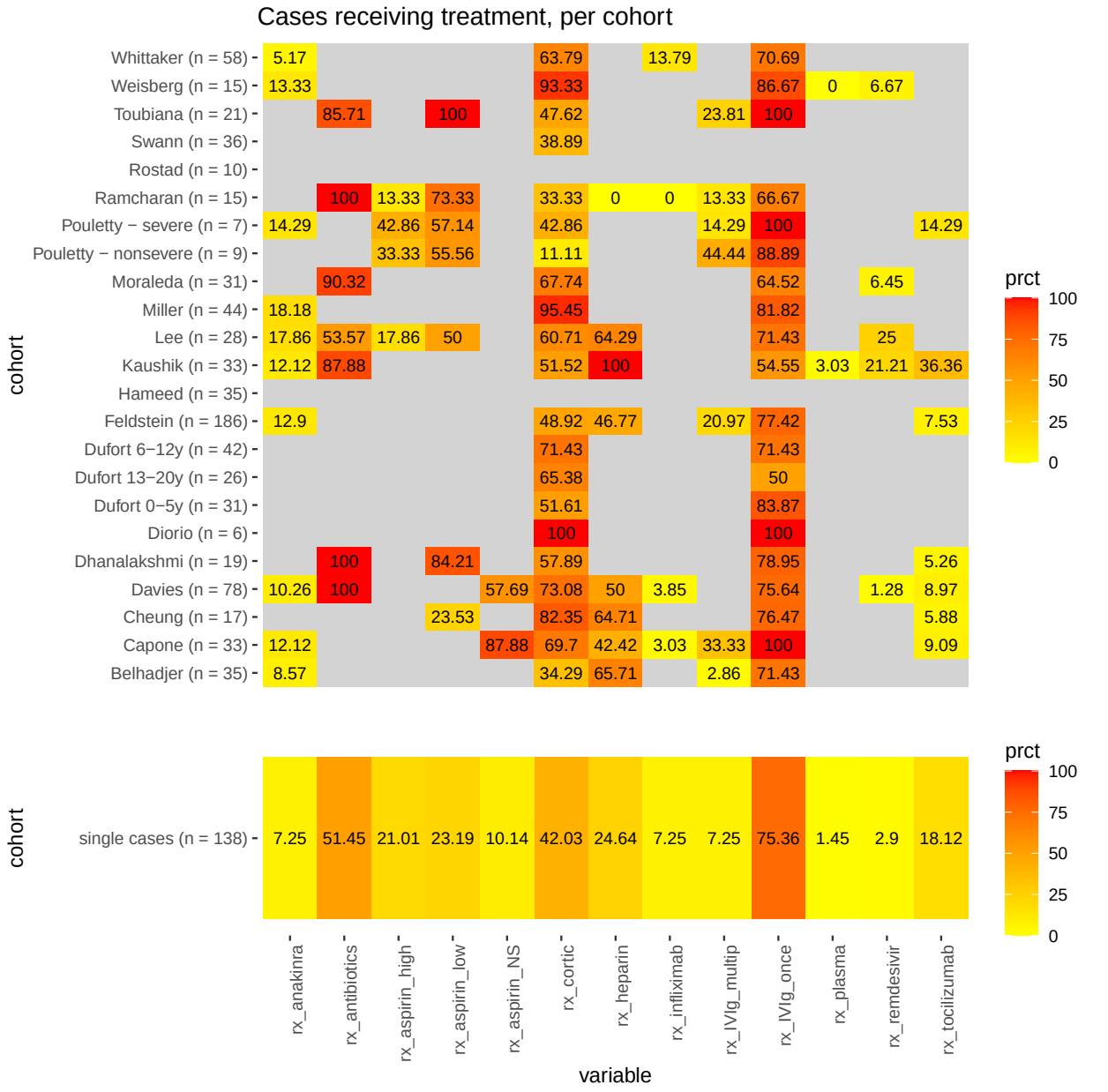
IVIg

```
makeBarplot(var_id_cohort = "rx_IVIg_once_n", var_id_single = "rx_IVIg_once", var_id = "IVIg")
```

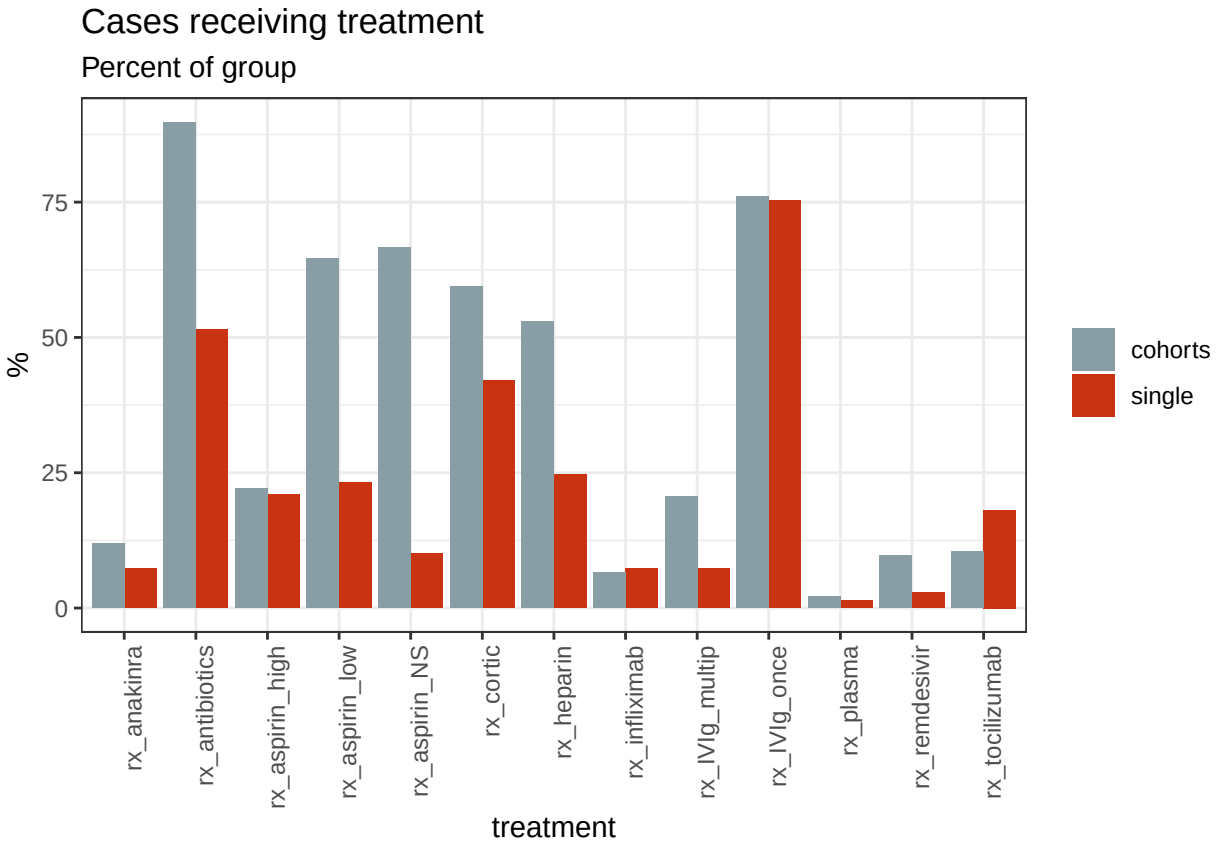


Overall therapy

```
makeHeatmap_cohort("rx", "rx",exclude_single = "days", plottitle = "Cases receiving treatment, per cohort")
```




```
barSymp("rx", "rx", exclude_single = "days", plottitle = "Cases receiving treatment")
```



Case definitions

Lab reference values

Cut-offs in this study:

- Neutrophilia > 8000/µL
- Elevated CRP > 10 mg/L
- Lymphopenia < 1250/µL
- WBC > 11000/µL
- Fibrinogen > 400 mg/dL
- D-dimers > 250 ng/mL
- Ferritin > 300 ng/mL
- Albumin < 34 g/L
- Procalcitonin > 0.49 ng/mL
- LDH > 280 U/L
- IL6 > 16.4 pg/mL
- ESR > 22 mm/
- BNP > 100 pg/mL
- NTproBNP > 400 pg/mL
- Troponin > 0.04 ng/mL

RCPCH, CDC and WHO

PIMS-TS

Source RCPCH

1. A child presenting with persistent fever, inflammation (neutrophilia, elevated CRP and lymphopaenia) and evidence of single or multi-organ dysfunction (shock, cardiac, respiratory, renal, gastrointestinal or neurological disorder) with additional features (see listed in Appendix 1). This may include children fulfilling full or partial criteria for Kawasaki disease.
2. Exclusion of any other microbial cause, including bacterial sepsis, staphylococcal or streptococcal shock syndromes, infections associated with myocarditis such as enterovirus (waiting for results of these investigations should not delay seeking expert advice).
3. SARS-CoV-2 PCR testing may be positive or negative

We are unable to evaluate criteria 2.

```
PIMS_TS_fulfilled <- apply(df_singlecases, 1, function(row) {
  # persistent fever, inflammation (neutrophilia, elevated CRP and lymphopaenia)
  pat_id <- row["patientID_int"]
  fever <- row["symp_fever"] == TRUE
  neutrophilia <- as.numeric(row["lab_neutrophils"]) > co_neutrophilia
  elevated_CRP <- (as.numeric(row["lab_CRP_admis"]) > co_CRP | as.numeric(row["lab_CRP_NS"]) > co_CRP | as.numeric(row["lab_CRP_peak"]) > co_CRP )
  lymphopenia <- as.numeric(row["lab_lymphocytes_lowest"]) < co_lympho
  inflamm <- any(fever, neutrophilia, elevated_CRP, lymphopenia)

  # lab values
  #fibrinogen <- row["lab_fibrino"] > co_fibrino
  #Ddimers <- row["lab_Ddim_peak"] > co_Ddim |  row["lab_Ddim_NS"] > co_Ddim
  #ferritin <- (row["lab_ferritin_NS"] > co_ferritin | row["lab_ferritin_admis"] > co_ferritin | row["lab_ferritin_peak"] > co_ferritin)
  #albumin <- row["lab_albumin_admis"] < co_albu | row["lab_albumin_lowest"] < co_albu | row["lab_albumin_NS"] < co_albu
  #lab_vals <- any(fibrinogen, Ddimers, ferritin, albumin)

  # single or multi-organ dysfunction (shock, cardiac, respiratory, renal, gastrointestinal or neurological disorder)
  pneumonia <- row["symp_resp_pneumonia"] == TRUE
  resp_failure <- row["symp_resp_failure"] == TRUE
  resp <- any(pneumonia, resp_failure)

  AKI <- row["symp_renal_AKI"] == TRUE
  RRT <- row["critcare_RRT"] == TRUE
  renal <- any(AKI, RRT)

  myocarditis <- row["symp_cardiovasc_myocard"] == TRUE
  pericarditis <- row["symp_cardiovasc_pericard"] == TRUE
  LVEF_under30 <- row["symp_cardiovasc_LV_less30"] == TRUE
  LVEF_30to55 <- row["symp_cardiovasc_LV_30to55"] == TRUE
  BNP <- (as.numeric(row["lab_BNP_admis"]) > co_BNP | as.numeric(row["lab_BNP_max"]) > co_BNP )
  NTproBNP <- as.numeric(row["lab_NTproBNP"]) > co_NTproBNP
  tropo <- as.numeric(row["lab_troponin_admis"]) > co_tropo
  shock <- row["symp_cardiovasc_shock"] == TRUE


  cardiovasc <- any(myocarditis, LVEF_under30, LVEF_30to55, NTproBNP, BNP, tropo, shock)

  rash <- row["kawasaki_exanthema"] == TRUE
  dermato <- any(rash)

  ddim <- as.numeric(row["lab_Ddim_NS"]) >= co_Ddim
  hemato <- any(ddim)

  organ_dysfunc <- sum(hemato, resp, renal, cardiovasc, dermato, na.rm = TRUE) >= 1

  criteria_fulfilled <- (inflamm) & organ_dysfunc #&lab_vals
  #return(c(pat_id, "criteria1_inflamm" = inflamm, "criteria2_labvals" = lab_vals, "criteria3_organdysfunc" = organ_dysfunc, "criteria_fulfilled" = criteria
         _fulfilled))
  return(c(pat_id, "criteria1_inflamm" = inflamm, "criteria2_organdysfunc" = organ_dysfunc, "criteria_fulfilled" = criteria_fulfilled))
})

PIMS_TS_fulfilled <- PIMS_TS_fulfilled %>% t() %>% as_tibble()
PIMS_TS_fulfilled <- type_convert(PIMS_TS_fulfilled)
PIMS_TS_fulfilled_heatmap <- PIMS_TS_fulfilled
cols <- sapply(PIMS_TS_fulfilled_heatmap, is.logical)
PIMS_TS_fulfilled_heatmap[,cols] <- lapply(PIMS_TS_fulfilled_heatmap[,cols], as.numeric)
PIMS_TS_fulfilled_heatmap_melt <- PIMS_TS_fulfilled_heatmap %>% melt()
PIMS_TS_fulfilled_heatmap_melt[is.na(PIMS_TS_fulfilled_heatmap_melt)] <- 2

skim(PIMS_TS_fulfilled)
```

Table 1: Data summary

Name	PIMS_TS_fulfilled
Number of rows	138
Number of columns	4

Table 1: Data summary

Column type frequency:	
character	1
logical	3
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
patientID_int	0	1	9	11	0	138	0

Variable type: logical

skim_variable	n_missing	complete_rate	mean	count
criteria1_inflamm	0	1	1	TRU: 138
criteria2_organdysfunc	0	1	1	TRU: 138
criteria_fulfilled	0	1	1	TRU: 138

```
#ggplot(PIMS_TS_fulfilled_heatmap_melt, aes(x = variable, y = as.character(patientID_int), fill = as.factor(value))) + geom_tile() + theme_classic() + theme
      (axis.line=element_blank()) + labs(y = "Patient ID", x = "criteria", fill = "criteria met", title = "Overview of which single cases fulfill PIMS-TS
      case definition") +  scale_fill_manual(labels = c("No", "Yes", "Missing"), values = c("pink2", "royalblue3", "darkgrey")) + theme(axis.text.x=element
      _text(angle=90, hjust=1))
```

CDC MIS-C

Source CDC and UpToDate

The case definition for MIS-C is:

1. Age <21 years
2. Clinical presentation consistent with MIS-C, including all of the following:

- Fever
- Documented fever >38.0°C (100.4°F) for 24 hours or
- Report of subjective fever lasting 24 hours
- Laboratory evidence of inflammation
- Severe illness requiring hospitalization
- Multisystem involvement
- 2 or more organ systems involved
- Cardiovascular (eg, shock, elevated troponin, elevated BNP, abnormal echocardiogram, arrhythmia)
- Respiratory (eg, pneumonia, ARDS, pulmonary embolism)
- Renal (eg, AKI, renal failure)
- Neurologic (eg, seizure, stroke, aseptic meningitis)
- Hematologic (eg, coagulopathy)
- Gastrointestinal (eg, elevated liver enzymes, diarrhea, ileus, gastrointestinal bleeding)
- Dermatologic (eg, erythroderma, mucositis, other rash)

3. No alternative plausible diagnoses

4. Recent or current SARS-CoV-2 infection or exposure

- Any of the following:
- Positive SARS-CoV-2 RT-PCR
- Positive serology
- Positive antigen test
- COVID-19 exposure within the 4 weeks prior to the onset of symptoms

```
CDC_fulfilled <- apply(df_singlecases, 1, function(row) {
  # criteria 1
  criteria1 = TRUE

  # criteria 2
  pat_id <- row["patientID_int"]

  # fever?
  fever <- row["symp_fever"] == TRUE | row["kawasaki_fever"] == TRUE

  inflamm <- any(fever)

  # lab values evidence for inflammation
  neutrophilia <- as.numeric(row["lab_neutrophils"]) > co_neutrophilia
  elevated_CRP <- (as.numeric(row["lab_CRP_admis"]) > co_CRP | as.numeric(row["lab_CRP_NS"]) > co_CRP | as.numeric(row["lab_CRP_peak"]) > co_CRP )
  lymphopenia <- as.numeric(row["lab_lymphocytes_lowest"]) < co_lympho
  fibrinogen <- as.numeric(row["lab_fibrino"]) > co_fibrino
  Ddimers <- as.numeric(row["lab_Ddim_peak"]) > co_Ddim |  as.numeric(row["lab_Ddim_NS"]) > co_Ddim
  ferritin <- (as.numeric(row["lab_ferritin_NS"]) > co_ferritin | as.numeric(row["lab_ferritin_admis"]) > co_ferritin | as.numeric(row["lab_ferritin_peak"])
        > co_ferritin)
  albumin <- as.numeric(row["lab_albumin_admis"]) < co_albu | as.numeric(row["lab_albumin_lowest"]) < co_albu | as.numeric(row["lab_albumin_NS"]) < co_albu
  PCT <- as.numeric(row["lab_PCT_admis"]) > co_PCT | as.numeric(row["lab_PCT_peak"]) > co_PCT | as.numeric(row["lab_PCT_NS"]) > co_PCT
  LDH <- as.numeric(row["lab_LDH"]) > co_LDH
  IL6 <- as.numeric(row["lab_IL6"]) > co_IL6
  ESR <- as.numeric(row["lab_ESR"]) > co_ESR

  lab_vals <- any(neutrophilia, elevated_CRP, lymphopenia, fibrinogen, Ddimers, ferritin, albumin, PCT, LDH, IL6, ESR)

  # Ilness requiring hospitalisation
  ## used surrogate parameters for hosp
  hosp_ICU <- row["admis_hosp_days"] > 1 | row["admis_ICU_days"] > 1 | row["admis_PICU_admis"] == TRUE
  NIV <- row["critcare_NIV"] == TRUE | row["critcare_NIV_days"] > 1
  MV <- row["critcare_MV"] == TRUE | row["critcare_MV_days"] > 1
  inotrop <- row["critcare_inotrop"] == TRUE | row["critcare_inotrop_days"] > 1
  ECMO <- row["critcare_ECMO"] == TRUE
  IVIg <- row["rx_IVIg_once"] == TRUE  |  row["rx_IVIg_multip"] == TRUE
  biologicals <- row["rx_anakinra"] == TRUE | row["rx_tocilizumab"] == TRUE | row["rx_infliximab"] == TRUE | row["rx_antibiotics"] == TRUE | row["rx_plasma"
        ] == TRUE | row["rx_remdesivir"] == TRUE
  heparin <- row["rx_heparin"] == TRUE


  req_hosp <- any(hosp_ICU, NIV, MV, inotrop, ECMO, IVIg, biologicals, heparin)

  ## multisystem involvement >= 2
  ## respiratory
  pneumonia <- row["symp_resp_pneumonia"] == TRUE
  resp_failure <- row["symp_resp_failure"] == TRUE
  resp <- any(pneumonia, resp_failure)

  AKI <- row["symp_renal_AKI"] == TRUE
  RRT <- row["critcare_RRT"] == TRUE
  renal <- any(AKI, RRT)

  myocarditis <- row["symp_cardiovasc_myocard"] == TRUE
  pericarditis <- row["symp_cardiovasc_pericard"] == TRUE
  LVEF_under30 <- row["symp_cardiovasc_LV_less30"] == TRUE
  LVEF_30to55 <- row["symp_cardiovasc_LV_30to55"] == TRUE
  BNP <- (as.numeric(row["lab_BNP_admis"]) > co_BNP | as.numeric(row["lab_BNP_max"]) > co_BNP )
  NTproBNP <- as.numeric(row["lab_NTproBNP"]) > co_NTproBNP
  tropo <- as.numeric(row["lab_troponin_admis"]) > co_tropo
  shock <- row["symp_cardiovasc_shock"] == TRUE

  cardiovasc <- any(myocarditis, LVEF_under30, LVEF_30to55, NTproBNP, BNP, tropo, shock)

  rash <- row["kawasaki_exanthema"] == TRUE
  dermato <- any(rash)

  organ_dysfunc <- sum(resp, renal, cardiovasc, dermato, na.rm = TRUE) >= 2

  criteria2 <- sum(inflamm, lab_vals, req_hosp, organ_dysfunc, na.rm = TRUE) == 4
  # criteria 3
  ## not evaluable
  criteria3 = TRUE
  # criteria 4
  # COVID pos?
  PCR_pos <- row["covid_PCR_pos"] == TRUE
  stool_pos <- row["covid_PCR_stool_pos"] == TRUE
  closecontact <- row["covid_closecontact"] == TRUE
  IgA <- row["covid_IgA_pos"] == TRUE
  IgM <- row["covid_IgM_pos"] == TRUE
  IgG <- row["covid_IgG_pos"] == TRUE
  any_sero <- row["covid_sero_pos"] == TRUE

  criteria4 <- any(PCR_pos, stool_pos, closecontact, IgA, IgM, IgG, any_sero)

  if (FALSE %in% c(criteria1, criteria2, criteria3, criteria4)){
    criteria_fulfilled <- FALSE
  } else if (NA %in% c(criteria1, criteria2, criteria3, criteria4)){
    criteria_fulfilled <- NA
  } else if (sum(criteria1, criteria2, criteria3, criteria4, na.rm = TRUE) == 4){
    criteria_fulfilled <- TRUE
```

```
  }

  #criteria_fulfilled <- sum(criteria1, criteria2, criteria3, criteria4, na.rm = TRUE) == 4
  return(c(pat_id, "criteria1_age" = criteria1, "criteria2_clinical" = criteria2, "criteria3_noAlt" = criteria3, "criteria4_recentExposure" = criteria4, "
        criteria_fulfilled" = criteria_fulfilled))
})

CDC_fulfilled <- CDC_fulfilled %>% t() %>% as_tibble()
CDC_fulfilled <- type_convert(CDC_fulfilled)
CDC_fulfilled_heatmap <- CDC_fulfilled
cols <- sapply(CDC_fulfilled_heatmap, is.logical)
CDC_fulfilled_heatmap[,cols] <- lapply(CDC_fulfilled_heatmap[,cols], as.numeric)
CDC_fulfilled_heatmap_melt <- CDC_fulfilled_heatmap %>% melt()
CDC_fulfilled_heatmap_melt[is.na(CDC_fulfilled_heatmap_melt)] <- 2

skim(CDC_fulfilled)
```

Table 4: Data summary

Name	CDC_fulfilled
Number of rows	138
Number of columns	6
Column type frequency:	
character	1
logical	5
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
patientID_int	0	1	9	11	0	138	0

Variable type: logical

skim_variable	n_missing	complete_rate	mean	count
criteria1_age	0	1.00	1.00	TRU: 138
criteria2_clinical	0	1.00	0.64	TRU: 89, FAL: 49
criteria3_noAlt	0	1.00	1.00	TRU: 138
criteria4_recentExposure	15	0.89	1.00	TRU: 123
criteria_fulfilled	8	0.94	0.62	TRU: 81, FAL: 49

```
#ggplot(CDC_fulfilled_heatmap_melt, aes(x = variable, y = as.character(patientID_int), fill = as.factor(value))) + geom_tile() + theme_classic() + theme(
      axis.line=element_blank()) + labs(y = "Patient ID", x = "criteria", fill = "criteria met", title = "Overview of which single cases fulfill CDC MIS-C
      case definition") +  scale_fill_manual(labels = c("No", "Yes", "Missing"), values = c("pink2", "royalblue3", "darkgrey")) + theme(axis.text.x=element
      _text(angle=90, hjust=1))
```

WHO case definition

Source UpToDate:

All 6 criteria must be met:

1. Age 0 to 19 years
2. Fever for 3 days
3. Clinical signs of multisystem involvement (at least 2 of the following):

- Rash, bilateral nonpurulent conjunctivitis, or mucocutaneous inflammation signs (oral, hands, or feet)
- Hypotension or shock
- Cardiac dysfunction, pericarditis, valvulitis, or coronary abnormalities (including echocardiographic findings or elevated troponin/BNP)
- Evidence of coagulopathy (prolonged PT or PTT; elevated D-dimer)
- Acute gastrointestinal symptoms (diarrhea, vomiting, or abdominal pain)

4. Elevated markers of inflammation (eg, ESR, CRP, or procalcitonin)
5. No other obvious microbial cause of inflammation, including bacterial sepsis and staphylococcal/streptococcal toxic shock syndromes
6. Evidence of SARS-CoV-2 infection

- Any of the following:
- Positive SARS-CoV-2 RT-PCR
- Positive serology
- Positive antigen test
- Contact with an individual with COVID-19

```r
#row <- df_singlecases[87, ]
WHO_fulfilled <- apply(df_singlecases, 1, function(row) {
  pat_id <- row["patientID_int"]

  # criteria 1
  criteria1 = TRUE

  # criteria 2: fever?
  fever <- row["symp_fever"] == TRUE | row["kawasaki_fever"] == TRUE

  criteria2 <- any(fever)

  # criteria 3: clinical signs of multisystem involvement (at least 2)
  ## Rash, bilateral nonpurulent conjunctivitis, or mucocutaneous inflammation signs (oral, hands, or feet)
  rash <- row["kawasaki_exanthema"] == TRUE
  conjunctivitis <- row["kawasaki_conjunctivitis"] == TRUE
  mucocutaneaous <- row["kawasaki_mouth"] == TRUE | row["kawasaki_extremity"] == TRUE

  criteria3_a <- any(rash, conjunctivitis, mucocutaneaous)

  ## hypotension or shock
  shock <- row["symp_cardiovasc_shock"] == TRUE
  criteria3_b <- any(shock)

  ## cardiac dysfunction
  myocarditis <- row["symp_cardiovasc_myocard"] == TRUE
  pericarditis <- row["symp_cardiovasc_pericard"] == TRUE
  LVEF_under30 <- row["symp_cardiovasc_LV_less30"] == TRUE
  LVEF_30to55 <- row["symp_cardiovasc_LV_30to55"] == TRUE
  BNP <- (as.numeric(row["lab_BNP_admis"]) > co_BNP | as.numeric(row["lab_BNP_max"]) > co_BNP )
  NTproBNP <- as.numeric(row["lab_NTproBNP"]) > co_NTproBNP
  tropo <- as.numeric(row["lab_troponin_admis"]) > co_tropo
  coronary <- row["symp_cardiovasc_cordilat"] == TRUE | row["symp_cardiovasc_aneurysm"] == TRUE

  criteria3_c <- any(myocarditis, LVEF_under30, LVEF_30to55, NTproBNP, BNP, tropo, coronary)

  ## coagulopathy
  fibrinogen <- as.numeric(row["lab_fibrino"]) > co_fibrino
  Ddimers <- as.numeric(row["lab_Ddim_peak"]) > co_Ddim |  as.numeric(row["lab_Ddim_NS"]) > co_Ddim

  criteria3_d <- any(fibrinogen, Ddimers)

  ## acute GI symptoms
  GIsymp <- row["symp_GI_NS"] == TRUE | row["symp_GI_abdopain"] == TRUE | row["symp_GI_vomiting"] == TRUE | row["symp_GI_diarrh"] == TRUE | row["symp_GI_
      colitis"] == TRUE

  criteria3_e <- any(GIsymp)

  criteria3 <- sum(criteria3_a, criteria3_b, criteria3_c, criteria3_d, criteria3_e, na.rm = TRUE) >= 2

  # criteria 4: Elevated markers of inflammation (eg, ESR, CRP, or procalcitonin)
  neutrophilia <- as.numeric(row["lab_neutrophils"]) > co_neutrophilia
  elevated_CRP <- (as.numeric(row["lab_CRP_admis"]) >= co_CRP) | (as.numeric(row["lab_CRP_NS"]) >= co_CRP) | (as.numeric(row["lab_CRP_peak"]) >= co_CRP )
  #  print(paste0(pat_id, elevated_CRP, row["lab_CRP_peak"]))
  lymphopenia <- as.numeric(row["lab_lymphocytes_lowest"]) < co_lympho

  ferritin <- (as.numeric(row["lab_ferritin_NS"]) > co_ferritin | as.numeric(row["lab_ferritin_admis"]) > co_ferritin | as.numeric(row["lab_ferritin_peak"])
      > co_ferritin)
  albumin <- as.numeric(row["lab_albumin_admis"]) < co_albu | as.numeric(row["lab_albumin_lowest"]) < co_albu | as.numeric(row["lab_albumin_NS"]) < co_albu
  PCT <- as.numeric(row["lab_PCT_admis"]) > co_PCT | as.numeric(row["lab_PCT_peak"]) > co_PCT | as.numeric(row["lab_PCT_NS"]) > co_PCT
  LDH <- as.numeric(row["lab_LDH"]) > co_LDH
  IL6 <- as.numeric(row["lab_IL6"]) > co_IL6
  ESR <- as.numeric(row["lab_ESR"]) > co_ESR

  criteria4 <- any(neutrophilia, elevated_CRP, lymphopenia, ferritin, albumin, PCT, LDH, IL6, ESR)

  # criteria 5: No other obvious microbial cause of inflammation
  criteria5 <- TRUE

  # criteria 6: COVID pos?
```

```
  PCR_pos <- row["covid_PCR_pos"] == TRUE
  stool_pos <- row["covid_PCR_stool_pos"] == TRUE
  closecontact <- row["covid_closecontact"] == TRUE
  IgA <- row["covid_IgA_pos"] == TRUE
  IgM <- row["covid_IgM_pos"] == TRUE
  IgG <- row["covid_IgG_pos"] == TRUE
  any_sero <- row["covid_sero_pos"] == TRUE

  criteria6 <- any(PCR_pos, stool_pos, closecontact, IgA, IgM, IgG, any_sero)

  if (NA %in% c(criteria1, criteria2, criteria3, criteria4, criteria5, criteria6)){
    criteria_fulfilled <- NA
  } else if (FALSE %in% c(criteria1, criteria2, criteria3, criteria4, criteria5, criteria6)){
    criteria_fulfilled <- FALSE
  } else if (sum(criteria1, criteria2, criteria3, criteria4, criteria5, criteria6, na.rm = TRUE) == 6){
    criteria_fulfilled <- TRUE
  } else {
    criteria_fulfilled <- FALSE
  }

  return(c(pat_id, "criteria1_age" = criteria1, "criteria2_fever" = criteria2, "criteria3_clinical" = criteria3, "criteria4_inflamm" = criteria4, "criteria5
        _noAlt" = criteria5, "criteria6_recentExposure" = criteria6, "criteria_fulfilled" = criteria_fulfilled))
})


WHO_fulfilled <- WHO_fulfilled %>% t() %>% as_tibble()
WHO_fulfilled <- type_convert(WHO_fulfilled)
WHO_fulfilled_heatmap <- WHO_fulfilled
cols <- sapply(WHO_fulfilled_heatmap, is.logical)
WHO_fulfilled_heatmap[,cols] <- lapply(WHO_fulfilled_heatmap[,cols], as.numeric)
WHO_fulfilled_heatmap_melt <- WHO_fulfilled_heatmap %>% melt()
WHO_fulfilled_heatmap_melt[is.na(WHO_fulfilled_heatmap_melt)] <- 2

skim(WHO_fulfilled)
```

Table 7: Data summary

Name	WHO_fulfilled
Number of rows	138
Number of columns	8
Column type frequency:	
character	1
logical	7
Group variables	None

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
patientID_int	0	1	9	11	0	138	0

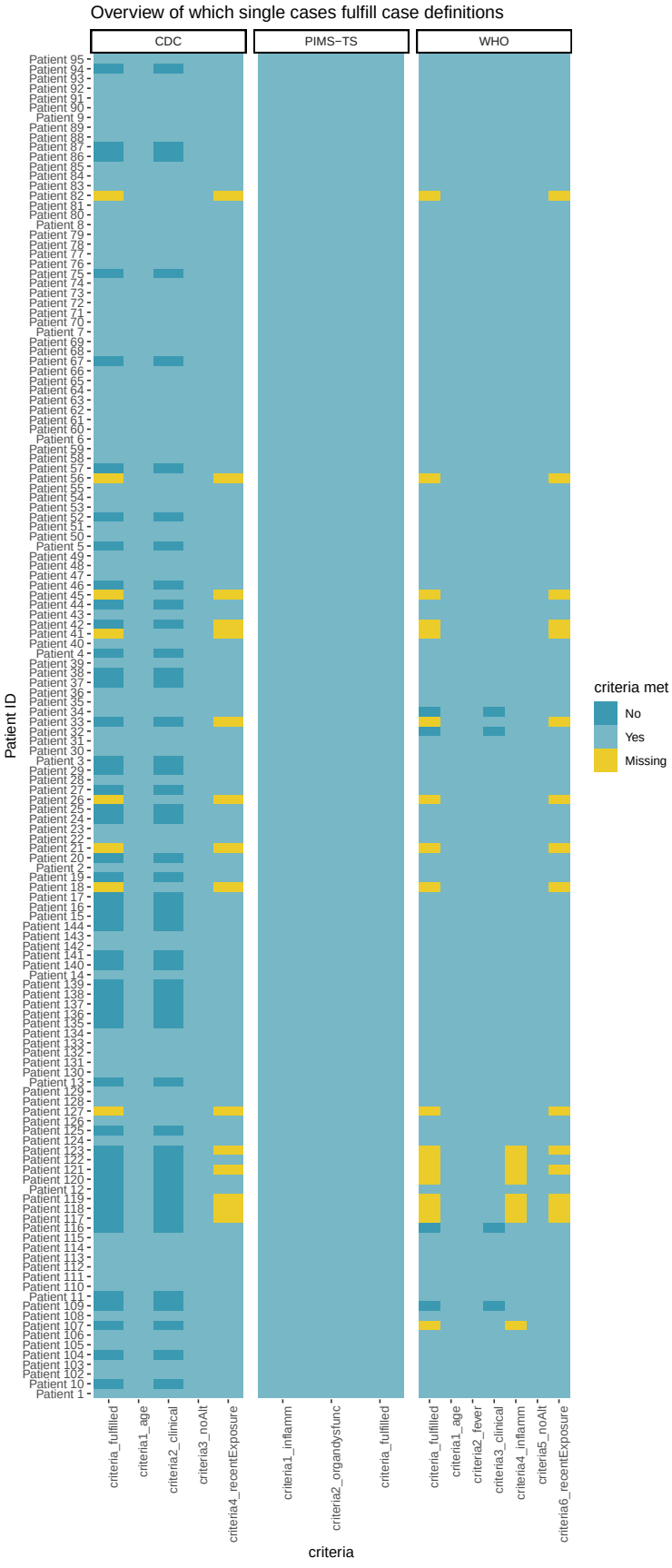
Variable type: logical

skim_variable	n_missing	complete_rate	mean	count
criteria1_age	0	1.00	1.00	TRU: 138
criteria2_fever	0	1.00	1.00	TRU: 138
criteria3_clinical	0	1.00	0.97	TRU: 134, FAL: 4
criteria4_inflamm	8	0.94	1.00	TRU: 130
criteria5_noAlt	0	1.00	1.00	TRU: 138
criteria6_recentExposure	15	0.89	1.00	TRU: 123
criteria_fulfilled	18	0.87	0.97	TRU: 116, FAL: 4

```
#ggplot(WHO_fulfilled_heatmap_melt, aes(x = variable, y = as.character(patientID_int), fill = as.factor(value))) + geom_tile() + theme_classic() + theme(
      axis.line=element_blank()) + labs(y = "Patient ID", x = "criteria", fill = "criteria met", title = "Overview of which single cases fulfill WHO MIS-C
      case definition") +  scale_fill_manual(labels = c("No", "Yes", "Missing"), values = c("pink2", "royalblue3", "darkgrey")) + theme(axis.text.x=element
      _text(angle=90, hjust=1))
```

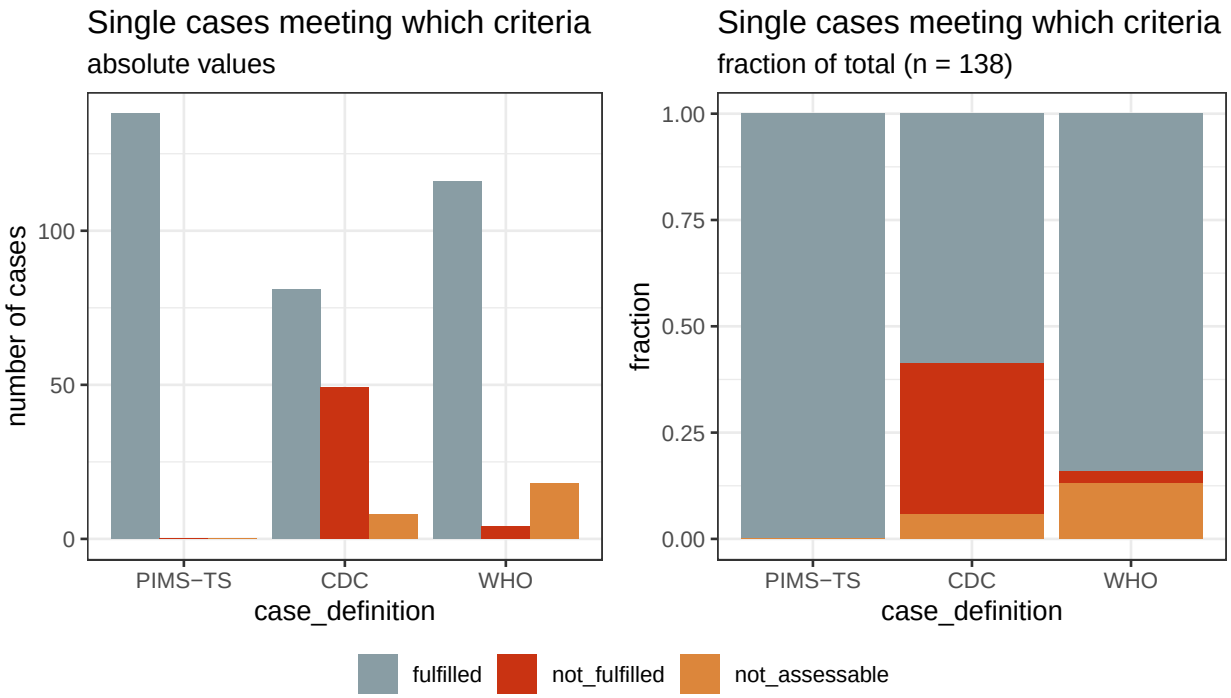
Per-case overview

```
PIMS_TS_fulfilled_heatmap_melt$criteria <- "PIMS-TS"
WHO_fulfilled_heatmap_melt$criteria <- "WHO"
CDC_fulfilled_heatmap_melt$criteria <- "CDC"

full_heatmap <- rbind(PIMS_TS_fulfilled_heatmap_melt, WHO_fulfilled_heatmap_melt, CDC_fulfilled_heatmap_melt)

ggplot(full_heatmap, aes(x = variable, y = as.character(patientID_int), fill = as.factor(value))) + geom_tile() + theme_classic() + theme(axis.line=element_
       blank()) + labs(y = "Patient ID", x = "criteria", fill = "criteria met", title = "Overview of which single cases fulfill case definitions") +  scale_
       fill_manual(labels = c("No", "Yes", "Missing"), values = wes_palette("Zissou1")) + theme(axis.text.x=element_text(angle=90, hjust=1)) + facet_wrap(~
       criteria, scales = "free_x")
```


Overview of which single cases fulfill case definitions
CDC
PIMS−TS
WHO
Patient ID
criteria
criteria met
No
Yes
Missing
criteria_fulfilled
criteria1_age
criteria2_clinical
criteria3_noAlt
criteria4_recentExposure
criteria1_inflamm
criteria2_organdysfunc
criteria_fulfilled
criteria_fulfilled
criteria1_age
criteria2_fever
criteria3_clinical
criteria4_inflamm
criteria5_noAlt
criteria6_recentExposure

Summary

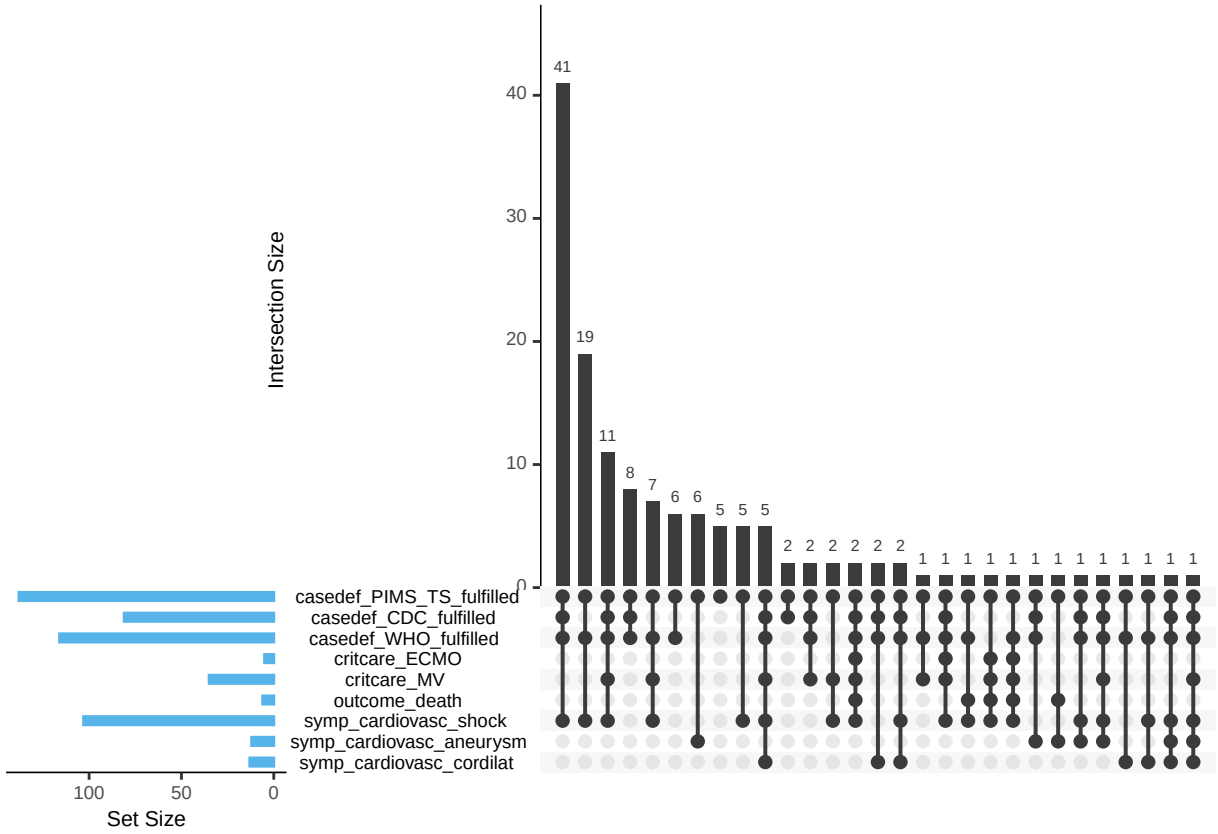
```
criteria_summary <- data.frame(PIMS_TS_fulfilled %>% select(criteria_fulfilled), CDC_fulfilled %>% select(criteria_fulfilled), WHO_fulfilled %>% select(
       criteria_fulfilled))
colnames(criteria_summary) <- c("PIMS-TS", "CDC", "WHO")

cols <- sapply(criteria_summary, is.logical)
criteria_summary[,cols] <- lapply(criteria_summary[,cols], as.numeric)

criteria_summary <- criteria_summary %>% melt() %>%
  group_by(variable) %>%
  summarise(fulfilled = sum(value == 1, na.rm = TRUE), not_fulfilled = sum(value == 0, na.rm = TRUE), not_assessable = sum(is.na(value)))
criteria_summary$sum <- rowSums(criteria_summary[,-1])

criteria_summary_melt <- criteria_summary %>% melt()
colnames(criteria_summary_melt) <- c("case_definition", "fulfilled", "count")

fill_bar <- ggplot(criteria_summary_melt %>% filter(fulfilled != 'sum'), aes(x = case_definition, y = count, fill = fulfilled)) +
  geom_bar(stat = "identity", position = "fill") + theme_bw() +
  labs(y = "fraction", title = "Single cases meeting which criteria", subtitle = paste0("fraction of total (n = ", max(criteria_summary_melt$count) ,")")) +
     theme(legend.title = element_blank()) +
  scale_fill_manual(values = wes_palette("Royal1")[c(1,2,4)])

dodge_bar <- ggplot(criteria_summary_melt %>% filter(fulfilled != 'sum'), aes(x = case_definition, y = count, fill = fulfilled)) +
  geom_bar(stat = "identity", position = "dodge") + theme_bw() +
  labs(y = "number of cases", title = "Single cases meeting which criteria", subtitle = "absolute values") +
   theme(legend.title = element_blank()) +
  scale_fill_manual(values = wes_palette("Royal1")[c(1,2,4)])

ggarrange(dodge_bar, fill_bar, legend = "bottom", common.legend = TRUE)
```



Association of case definition with outcome

```
WHO_outcome <- WHO_fulfilled_heatmap %>% select(contains("patientID_int") | contains("criteria_fulfilled"))
colnames(WHO_outcome) <- c("patientID_int", "casedef_WHO_fulfilled")

CDC_outcome <- CDC_fulfilled_heatmap %>% select(contains("patientID_int") | contains("criteria_fulfilled"))
colnames(CDC_outcome) <- c("patientID_int", "casedef_CDC_fulfilled")

PIMS_TS_outcome <- PIMS_TS_fulfilled_heatmap %>% select(contains("patientID_int") | contains("criteria_fulfilled"))
colnames(PIMS_TS_outcome) <- c("patientID_int", "casedef_PIMS_TS_fulfilled")

assoc_outcome <- merge(WHO_outcome, CDC_outcome, by = "patientID_int")
assoc_outcome <- merge(assoc_outcome, PIMS_TS_outcome)

#assoc_outcome <- assoc_outcome[complete.cases(assoc_outcome[ ,-1]),]

outcome_params <- df_singlecases %>% select(patientID_int | symp_cardiovasc_cordilat | symp_cardiovasc_aneurysm | symp_cardiovasc_shock | outcome_death |
       critcare_MV | critcare_ECMO)

assoc_outcome_full <- merge(outcome_params, assoc_outcome, by = "patientID_int", all = TRUE)

```

```
cols <- sapply(assoc_outcome_full, is.logical)
assoc_outcome_full[,cols] <- lapply(assoc_outcome_full[,cols], as.numeric)


makeUpsetR(assoc_outcome_full %>% select(-contains("patientID")))
```



Severe course

A new variable 'severe course' made, which contains the following:

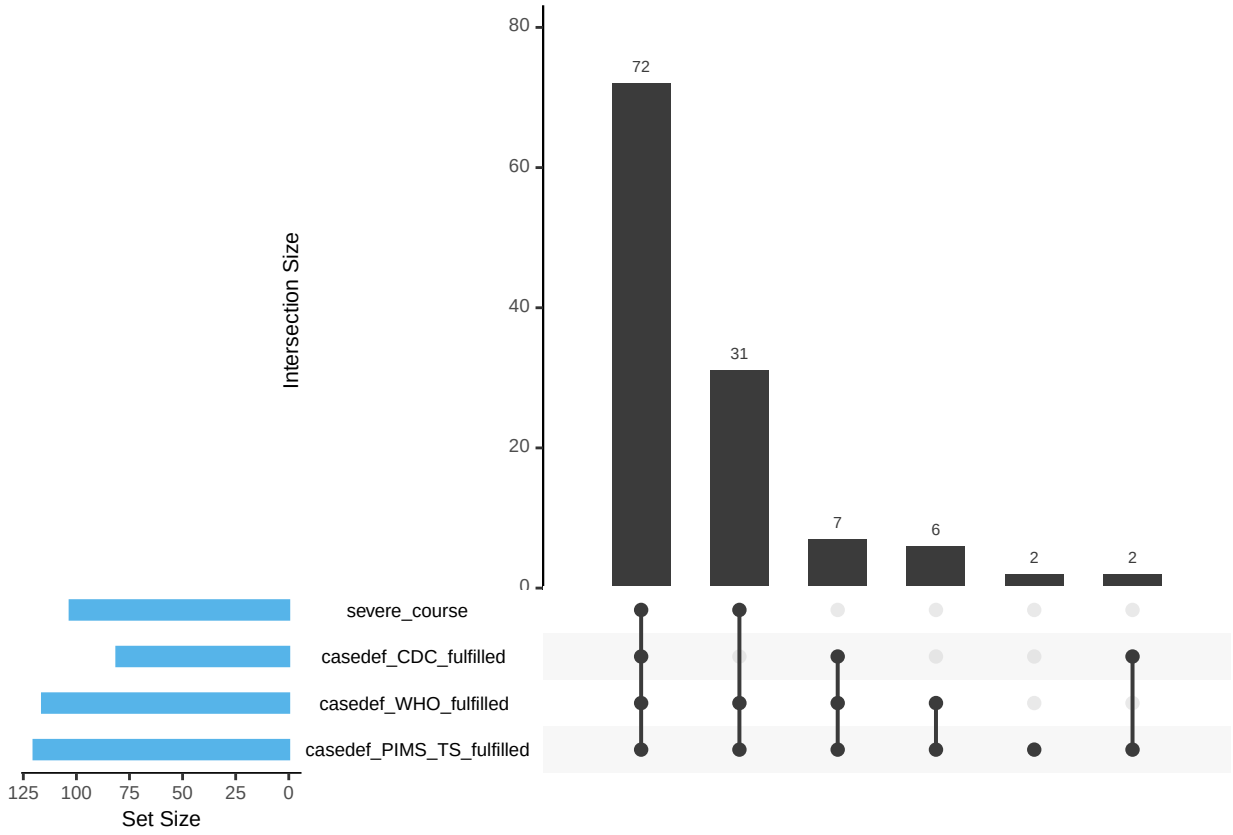
- symp_cardiovasc_cordilat
- symp_cardiovasc_aneurysm
- symp_cardiovasc_shock
- outcome_death
- critcare_MV
- critcare_ECMO
- critcare_RRT
- critcare_inotrop
- admis_PICU_admis

Mild presentation means all of the above are either 0 or NA.

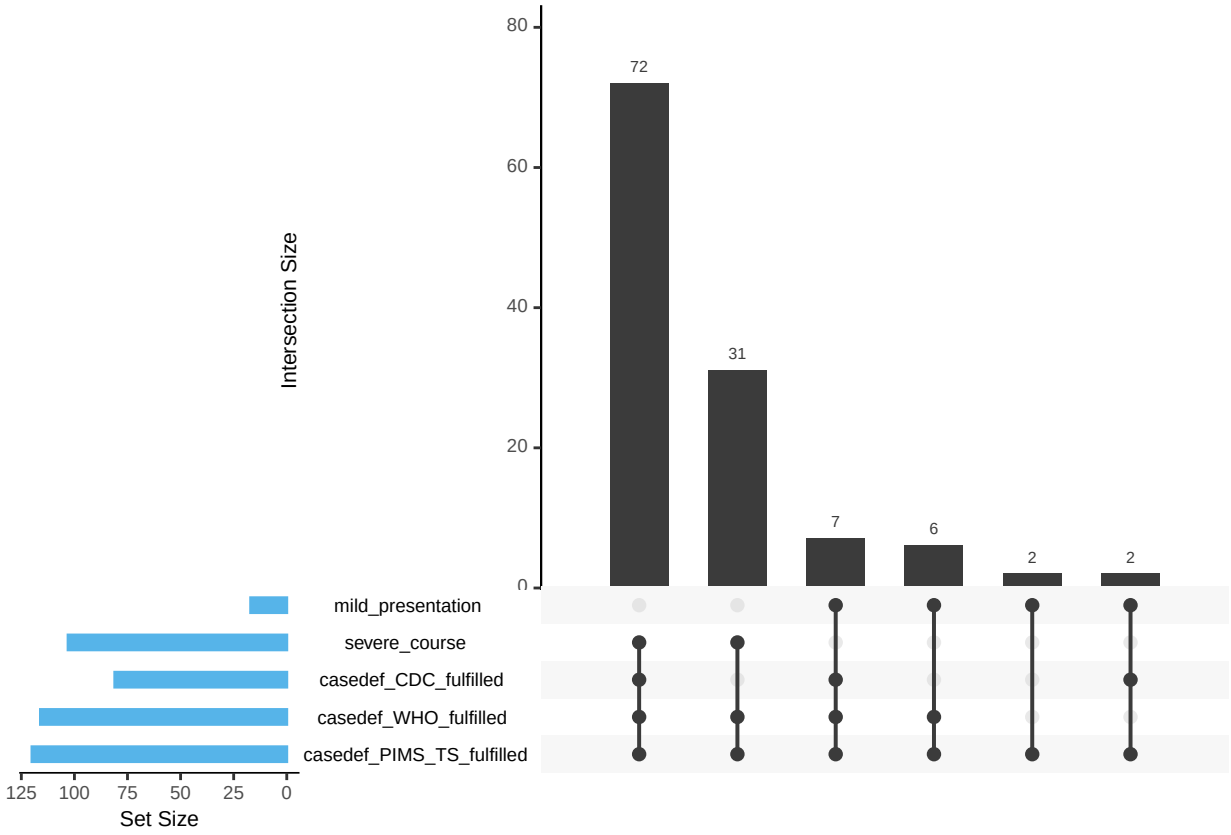
Cases with missing values in case defintions are removed.

```
assoc_outcome <- merge(WHO_outcome, CDC_outcome, by = "patientID_int")
assoc_outcome <- merge(assoc_outcome, PIMS_TS_outcome)
assoc_outcome <- merge(assoc_outcome, PIMS_TS_outcome)
assoc_outcome <- assoc_outcome[complete.cases(assoc_outcome[ ,-1]),]

```

```r
outcome_params <- df_singlecases %>% select(patientID_int | contains("critcare")  | contains("admis_PICU_admis") | contains("outcome_death")  |contains ("
       symp_cardiovasc_cordilat") | contains ("symp_cardiovasc_aneurysm")  |contains("symp_cardiovasc_shock"))

assoc_outcome_full <- merge(outcome_params, assoc_outcome, by = "patientID_int")

cols <- sapply(assoc_outcome_full, is.logical)
assoc_outcome_full[,cols] <- lapply(assoc_outcome_full[,cols], as.numeric)

assoc_outcome_full$severe_course <- ifelse(assoc_outcome_full$symp_cardiovasc_cordilat == 1 | assoc_outcome_full$symp_cardiovasc_aneurysm == 1 | assoc_
       outcome_full$symp_cardiovasc_shock == 1 | assoc_outcome_full$outcome_death == 1 | assoc_outcome_full$critcare_MV == 1 | assoc_outcome_full$critcare_
       ECMO == 1 | assoc_outcome_full$critcare_RRT == 1 | assoc_outcome_full$admis_PICU_admis == 1 | assoc_outcome_full$critcare_inotrop == 1 , 1, 0)

assoc_outcome_full$mild_presentation <- ifelse((assoc_outcome_full$severe_course == 0 | is.na(assoc_outcome_full$severe_course)), 1, 0)

makeUpsetR(assoc_outcome_full %>% select(contains("casedef") | contains("severe_course") ))
```




```r
makeUpsetR(assoc_outcome_full %>% select(contains("casedef") | contains("severe_course")  | contains("mild_pres") ))
```



Characteristics of severe course

Download data as .csv on Github

```
# in this step, do not remove NAs
assoc_outcome <- merge(WHO_outcome, CDC_outcome, by = "patientID_int")
assoc_outcome <- merge(assoc_outcome, PIMS_TS_outcome)
assoc_outcome <- merge(assoc_outcome, PIMS_TS_outcome)
#assoc_outcome <- assoc_outcome[complete.cases(assoc_outcome[ ,-1]),]

outcome_params <- df_singlecases %>% select(patientID_int | contains("critcare")  | contains("admis_PICU_admis") | contains("outcome_death")  |contains ("
      symp_cardiovasc_cordilat") | contains ("symp_cardiovasc_aneurysm")  |contains("symp_cardiovasc_shock"))

assoc_outcome_full <- merge(outcome_params, assoc_outcome, by = "patientID_int")

cols <- sapply(assoc_outcome_full, is.logical)
assoc_outcome_full[,cols] <- lapply(assoc_outcome_full[,cols], as.numeric)

assoc_outcome_full$severe_course <- ifelse(assoc_outcome_full$symp_cardiovasc_cordilat == 1 | assoc_outcome_full$symp_cardiovasc_aneurysm == 1 | assoc_
      outcome_full$symp_cardiovasc_shock == 1 | assoc_outcome_full$outcome_death == 1 | assoc_outcome_full$critcare_MV == 1 | assoc_outcome_full$critcare_
      ECMO == 1 | assoc_outcome_full$critcare_RRT == 1 | assoc_outcome_full$admis_PICU_admis == 1 | assoc_outcome_full$critcare_inotrop == 1 , 1, 0)


tab1 <- assoc_outcome_full %>% select(patientID_int,severe_course)
tab1$severe_course <- ifelse(is.na(tab1$severe_course), 0, 1)

tab2 <- df_singlecases %>% select(patientID_int,
                                  sex, age,
                                  symp_resp_any,
                                  symp_cardiovasc_any,
                                  symp_cardiovasc_myocard,
                                  symp_cardiovasc_pericard,
                                  symp_cardiovasc_cordilat,
                                  symp_cardiovasc_aneurysm,
                                  symp_cardiovasc_LV_30to55,
                                  symp_cardiovasc_LV_less30,
                                  symp_cardiovasc_shock,
                                  symp_cardiovasc_LVEF,
                                  symp_cardiovasc_tachycard,
                                  symp_GI_any,
                                  symp_neuro_any,
                                  kawasaki_complete,
                                  kawasaki_incomplete,
                                  kawasaki_fever,
                                  kawasaki_exanthema,
                                  kawasaki_extremity,
                                  kawasaki_mouth,
```

```
                                kawasaki_cervical,
                                kawasaki_conjunctivitis,
                                covid_PCR_pos,
                                covid_sero_any,
                                admis_PICU_admis,
                                critcare_NIV,
                                critcare_MV,
                                critcare_inotrop,
                                critcare_ECMO,
                                critcare_RRT,
                                rx_cortic,
                                rx_heparin,
                                rx_IVIg_once,
                                rx_IVIg_multip,
                                rx_anakinra,
                                rx_tocilizumab,
                                rx_infliximab,
                                rx_antibiotics,
                                rx_plasma,
                                rx_remdesivir)

tab2$sex <- as.factor(tab2$sex)

labvals <-
  data.frame(
    collapse_labvals_single(df_singlecases, "max", "CRP") %>% select(CRP_max),
    collapse_labvals_single(df_singlecases, "max", "ferritin") %>% select(ferritin_max),
    collapse_labvals_single(df_singlecases, "max", "IL") %>% select(IL_max),
    collapse_labvals_single(df_singlecases, "max", "WBC") %>% select(WBC_max),
    collapse_labvals_single(df_singlecases, "min", "lympho") %>% select(lympho_min),
    collapse_labvals_single(df_singlecases, "min", "platelet") %>% select(platelet_min),
    collapse_labvals_single(df_singlecases, "min", "sodium") %>% select(sodium_min),
    collapse_labvals_single(df_singlecases, "max", "Ddim") %>% select(Ddim_max),
    collapse_labvals_single(df_singlecases, "max", "trop") %>% select(trop_max)
  )
tab2 <- cbind(tab2, labvals)

tab <- merge(tab1, tab2)

skim <- tab %>% group_by(severe_course) %>% skim()

skim <- skim %>% select(skim_variable, severe_course, factor.top_counts, logical.count,  numeric.p25,  numeric.p50,  numeric.p75, n_missing)

skim$n_total <- rep(count(tab1, severe_course) %>% pull(n), nrow(skim)/2)
skim$n_valid <- skim$n_total - skim$n_missing

write.csv(skim, paste0("./data/unfavorable_course_descriptivestats.csv"))
```

SessionInfo

```
sessionInfo()
```

```
## R version 3.6.3 (2020-02-29)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Catalina 10.15.6
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets
## [6] methods   base
##
## other attached packages:
##  [1] gdtools_0.2.1       padr_0.5.2
##  [3] wesanderson_0.3.6   see_0.5.1
##  [5] UpSetR_1.4.0        skimr_2.1.1
##  [7] psych_1.9.12.31     zoo_1.8-7
##  [9] DT_0.13             naniar_0.5.2
## [11] cowplot_1.0.0       ggpubr_0.2.5
## [13] magrittr_1.5        ggbeeswarm_0.6.0
## [15] ggExtra_0.9         gridExtra_2.3
## [17] ggrepel_0.8.2       scales_1.1.1
## [19] RColorBrewer_1.1-2 broom_0.5.5
## [21] reshape2_1.4.3      httr_1.4.1
## [23] readxl_1.3.1        forcats_0.5.0
## [25] stringr_1.4.0       dplyr_0.8.5
## [27] purrr_0.3.4         readr_1.3.1
## [29] tidyr_1.0.2         tibble_3.0.1
## [31] ggplot2_3.3.1       tidyverse_1.3.0
##
## loaded via a namespace (and not attached):
##  [1] nlme_3.1-144       fs_1.4.0           lubridate_1.7.4
##  [4] insight_0.8.5      repr_1.1.0         tools_3.6.3
##  [7] backports_1.1.7    R6_2.4.1           vipor_0.4.5
## [10] DBI_1.1.0          colorspace_1.4-1   withr_2.2.0
## [13] tidyselect_1.1.0   mnormt_1.5-6       compiler_3.6.3
## [16] cli_2.0.2          rvest_0.3.5        xml2_1.3.0
## [19] labeling_0.3       bayestestR_0.7.0   ggridges_0.5.2
## [22] systemfonts_0.1.1 digest_0.6.25      svglite_1.2.3
## [25] rmarkdown_2.1      base64enc_0.1-3    pkgconfig_2.0.3
## [28] htmltools_0.4.0    highr_0.8          dbplyr_1.4.2
## [31] fastmap_1.0.1      htmlwidgets_1.5.1 rlang_0.4.6
## [34] rstudioapi_0.11    shiny_1.4.0.2      farver_2.0.3
```

```
## [37] generics_0.0.2    jsonlite_1.6.1    crosstalk_1.1.0.1
## [40] parameters_0.8.0  Rcpp_1.0.4        munsell_0.5.0
## [43] fansi_0.4.1       lifecycle_0.2.0   visdat_0.5.3
## [46] stringi_1.4.6     yaml_2.2.1        plyr_1.8.6
## [49] grid_3.6.3        parallel_3.6.3    promises_1.1.0
## [52] crayon_1.3.4      miniUI_0.1.1.1    lattice_0.20-38
## [55] haven_2.2.0       hms_0.5.3         knitr_1.28
## [58] pillar_1.4.4      ggsignif_0.6.0    effectsize_0.3.1
## [61] reprex_0.3.0      glue_1.4.1        evaluate_0.14
## [64] modelr_0.1.6      vctrs_0.3.0       httpuv_1.5.2
## [67] cellranger_1.1.0  gtable_0.3.0      assertthat_0.2.1
## [70] xfun_0.12         mime_0.9          xtable_1.8-4
## [73] later_1.0.0       beeswarm_0.2.3    ellipsis_0.3.1
```