Closing the Health Gap in Brazil: Complete R code

Data cleansing

Mortality Information System (SIM)

```
# Import the datasets

for (i in c(2010, 2020)) {

  name <- paste0("sim_", i)
  tmp <- read.csv(paste0("DL/SIM_", i, ".csv"), header = TRUE) %>%
    as_tibble() %>%
    mutate_if(is.character, as.numeric) %>%
    mutate(year = i)
  assign(name, tmp)

}

rm(list = c("i", "name", "tmp"))

# ------------------------------------------------------------------------

# Combine the tibbles

sim <- sim_2010 %>%
  bind_rows(sim_2020) %>%
  select(municipality, year, everything()) %>%
  arrange(year, municipality)

# ------------------------------------------------------------------------

# Remove temporary files

rm(list = c("sim_2010", "sim_2020"))
```

Hospitalizations Information System (SIH)

```
# Import the datasets

for (i in c(2010, 2020)) {

  name <- paste0("sih_", i)
```

```
  tmp <- read.csv(paste0("DL/SIH_", i, ".csv"), header = TRUE) %>%
    as_tibble() %>%
    mutate_if(is.character, as.numeric) %>%
    mutate(year = i)
  assign(name, tmp)

}

rm(list = c("i", "name", "tmp"))

# ------------------------------------------------------------------------

# Combine the tibbles

sih <- sih_2010 %>%
  bind_rows(sih_2020) %>%
  select(municipality, year, everything()) %>%
  arrange(year, municipality)

# ------------------------------------------------------------------------

# Remove temporary files

rm(list = c("sih_2010", "sih_2020"))
```

Health Infrastructure Database (CNES)

```
# Import the datasets
# --- cnes_1: professionals, cnes_2: facilities, cnes_3: equipments

tmp_list <- c("professionals", "facilities", "equipments")

for (i in 1:3) {

  for (j in c(2010, 2020)) {

    name <- paste0("cnes_", i, "_", j)
    tmp <- read.csv(
      paste0("DL/health_", tmp_list[[i]], "_", j, ".csv"),
      header = TRUE
      ) %>%
      as_tibble() %>%
      mutate_if(is.character, as.numeric)
    assign(name, tmp)

  }

}

rm(list = c("tmp_list", "i", "j", "name", "tmp"))
```

```r
# ------------------------------------------------------------------------

# Make some modifications before combining all six datasets

# cnes_1_2010
# --- Add year column
  cnes_1_2010 <- cnes_1_2010 %>%
    mutate(year = 2010)

# cnes_1_2020
# --- Change the name of the first variable to "municipality"
  colnames(cnes_1_2020)[[1]] <- "municipality"
# --- Add year column
  cnes_1_2020 <- cnes_1_2020 %>%
    mutate(year = 2020)

# cnes_2_2010
# --- Add year column
  cnes_2_2010 <- cnes_2_2010 %>%
    mutate(year = 2010)

# cnes_2_2020
# --- Add year column
  cnes_2_2020 <- cnes_2_2020 %>%
    mutate(year = 2020)

# cnes_3_2010
# --- Add year column
  cnes_3_2010 <- cnes_3_2010 %>%
    mutate(year = 2010)

# cnes_3_2020
# --- Add year column
  cnes_3_2020 <- cnes_3_2020 %>%
    mutate(year = 2020)

# ------------------------------------------------------------------------

# Put everything together

for (i in 1:3) {

  name <- paste0("cnes_", i)
  tmp <- tibble() %>%
    bind_rows(get(paste0("cnes_", i, "_2010"))) %>%
    bind_rows(get(paste0("cnes_", i, "_2020"))) %>%
    select(municipality, year, everything()) %>%
    arrange(year, municipality)
  assign(name, tmp)

}

# Combining all three datasets yields a dataset with too many
```

```
# variables such that it would be difficult to browse on RStudio.
# So I comment out the following lines for now.
# cnes <- cnes_1 %>%
#   full_join(cnes_2, by = c("municipality", "year")) %>%
#   full_join(cnes_3, by = c("municipality", "year"))

# ------------------------------------------------------------------

# Remove temporary files

rm(list = c("cnes_1_2010", "cnes_1_2020",
            "cnes_2_2010", "cnes_2_2020",
            "cnes_3_2010", "cnes_3_2020",
            "i", "name", "tmp"))

# Use the following line if you have combined all three datasets
# rm(list = c("cnes_1", "cnes_1_2010", "cnes_1_2020",
#             "cnes_2", "cnes_2_2010", "cnes_2_2020",
#             "cnes_3", "cnes_3_2010", "cnes_3_2020",
#             "i", "name", "tmp"))
```

2010 Brazilian Census

```
# Import the datasets

census_households <- read.csv(
  "DL/census_households_2010.csv", header = TRUE
  ) %>%
    as_tibble()
# Add "hh_" before the variable names for the household level data
colnames(census_households)[3:17] <- paste0(
  "hh_", colnames(census_households)[3:17]
  )

census_people <- read.csv(
  "DL/census_people_2010.csv", header = TRUE
  ) %>%
  as_tibble()
# Add "ppl_" before the variable names for the people level data
colnames(census_people)[3:10] <- paste0(
  "ppl_", colnames(census_people)[3:10]
  )

municipality_codes <- read_xls("DL/municipality_codes.xls")
colnames(municipality_codes) <- c("state", "state_name",
                                  "municipality", "municipality_name")

# ------------------------------------------------------------------

# Combine the datasets
```

```
census <- municipality_codes %>%
  left_join(census_households, by = c("state", "municipality")) %>%
  left_join(census_people, by = c("state", "municipality")) %>%
  select(state, state_name, municipality, municipality_name,
         everything())

# ------------------------------------------------------------------------

# Remove temporary files

rm(list = c("census_households", "census_people"))
```

Save the datasets

SIM

```
sim <- municipality_codes %>%
  left_join(sim, by = c("municipality"))
save(sim, file = "sim_mortality.Rda")
```

SIH

```
sih <- municipality_codes %>%
  left_join(sih, by = c("municipality"))
save(sih, file = "sih_hospitalization.Rda")
```

CNES

```
# CNES_1: Health professionals

cnes_1 <- municipality_codes %>%
  left_join(cnes_1, by = c("municipality"))
save(cnes_1, file = "cnes_1_professionals.Rda")

# CNES_2: Facilities

cnes_2 <- municipality_codes %>%
  left_join(cnes_2, by = c("municipality"))
save(cnes_2, file = "cnes_2_facilities.Rda")

# CNES_3: Equipment

cnes_3 <- municipality_codes %>%
  left_join(cnes_3, by = c("municipality"))
save(cnes_3, file = "cnes_3_equipment.Rda")
```

Census

```
save(census, file = "census.Rda")
save(municipality_codes, file = "municipality_codes.Rda")
```

Memo

```
rm(list = ls())

Chart1_data <- read_csv("Chart1_data.csv")
```

```
## Rows: 31 Columns: 4
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (4): Year, Brazil, Latin_America, World
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# ------------------------------------------------------------------------

# Chart 1

Chart1 <- Chart1_data %>%
  ggplot(aes(x = Year)) +
    geom_line(aes(y = Brazil), color = "#C00000") +
    geom_line(aes(y = Latin_America), color = "#0070C0") +
    geom_line(aes(y = World), color = "#8BC63E") +
    scale_x_continuous(breaks = seq(1990, 2020, 5),
                       labels = c("90", "95", "00", "05", "10", "15", "20"),
                       limits = c(1990, 2020)) +
    scale_y_continuous(breaks = seq(60, 80, 5),
                       labels = seq(60, 80, 5),
                       limits = c(60, 80),
                       expand = c(0, 0, 0.025, 0)) +
    labs(title = "",
         subtitle = "years") +
    theme(
      plot.title = element_blank(),
      plot.subtitle = element_text(size = 9, hjust = 0),
      plot.caption = element_blank(),
      panel.background = element_rect(fill = "transparent"),
      plot.background = element_rect(fill = "transparent"),
      panel.grid.major.y =
        element_line(color = "grey", size = 0.25, linetype = "solid"),
      panel.grid.minor = element_blank(),
      panel.grid.major.x = element_blank(),
      axis.line.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.line.y = element_blank(),
```

```
      axis.title.x = element_blank(),
      axis.title.y = element_blank(),
      axis.text.x = element_text(color = "black", size = 9),
      axis.text.y = element_text(color = "black", size = 9),
      axis.ticks.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.ticks.y = element_blank(),
      axis.ticks.length.x = unit(5, "pt"),
      strip.background = element_rect(fill = NA),
      strip.text = element_text(color = "black", size = 9)) +
    annotate("text", x = 2020, y = 78.0, label = "Brazil", color = "#C00000",
             size = 9/.pt, hjust = 1) +
    annotate("text", x = 1991, y = 74.2, label = "Latin America", color = "#0070C0",
             size = 9/.pt, hjust = 0) +
    annotate("text", x = 2020, y = 70.8, label = "World", color = "#8BC63E",
             size = 9/.pt, hjust = 1)

ggsave("Chart1.pdf", Chart1,
       width = 5.0, height = 5.5, units = "cm",
       dpi = 300, device = cairo_pdf)
```

```
load("census.Rda")
load("sim_mortality.Rda")
load("sih_hospitalization.Rda")

# ------------------------------------------------------------------------

census_tmp <- census %>%
  select(municipality, ppl_pop)

# Calculate the # of deaths per capita

dpc_municipality_2020 <- sim %>%
  filter(year == 2020) %>%
  mutate(deaths = rowSums(.[c(6:24)], na.rm = TRUE)) %>%
  # Combine the population data from the 2010 Census
  left_join(census_tmp, by = "municipality") %>%
  mutate(dpc = deaths / ppl_pop)

# Calculate the # of hospitalizations per capita

hpc_municipality_2020 <- sih %>%
  filter(year == 2020) %>%
  mutate(hospitalizations = rowSums(.[c(6, 14, 15, 16, 24)],
                                    na.rm = TRUE)) %>%
  # Combine the population data from the 2010 Census
  left_join(census_tmp, by = "municipality") %>%
  mutate(hpc = hospitalizations / ppl_pop)

rm(census_tmp)

# ------------------------------------------------------------------------

# Chart 2
```

```r
Chart2_data <- census %>%
  select(1:4) %>%
  left_join(dpc_municipality_2020 %>%
              filter(year == 2020) %>%
              select(municipality, dpc),
            by = c("municipality")) %>%
  mutate(dpc_1000_2020 = dpc * 1000) %>% # Unit: per 1,000 ppl
  select(-dpc) %>%
  left_join(hpc_municipality_2020 %>%
              filter(year == 2020) %>%
              select(municipality, hpc),
            by = c("municipality")) %>%
  mutate(hpc_1000_2020 = hpc * 1000) %>% # Unit: per 1,000 ppl
  select(-hpc)

Chart2L <- Chart2_data %>%
  ggplot() +
    geom_histogram(aes(x = dpc_1000_2020, y = ..density.. * 100),
                   fill = "#C00000", bins = 50) +
    scale_x_continuous(breaks = seq(0, 20, 5),
                       labels = seq(0, 20, 5),
                       limits = c(0, 20)) +
    scale_y_continuous(breaks = seq(0, 25, 5),
                       labels = seq(0, 25, 5),
                       limits = c(0, 25),
                       expand = c(0, 0, 0.025, 0)) +
    labs(title = "",
         subtitle = "percent",
         x = "Deaths per 1,000 people") +
    theme(
      plot.title = element_blank(),
      plot.subtitle = element_text(size = 9, hjust = 0),
      plot.caption = element_blank(),
      panel.background = element_rect(fill = "transparent"),
      plot.background = element_rect(fill = "transparent"),
      panel.grid.major.y =
        element_line(color = "grey", size = 0.25, linetype = "solid"),
      panel.grid.minor = element_blank(),
      panel.grid.major.x = element_blank(),
      axis.line.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.line.y = element_blank(),
      axis.title.x = element_blank(),
      axis.title.y = element_blank(),
      axis.text.x = element_text(color = "black", size = 9),
      axis.text.y = element_text(color = "black", size = 9),
      axis.ticks.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.ticks.y = element_blank(),
      axis.ticks.length.x = unit(5, "pt"),
      strip.background = element_rect(fill = NA),
      strip.text = element_text(color = "black", size = 9)) +
    annotate("text", x = 10.5, y = 17.5, label = "2020", color = "#C00000",
             size = 9/.pt, hjust = 0)
```

```
ggsave("Chart2L.pdf", Chart2L,
       width = 5.0, height = 5.5, units = "cm",
       dpi = 300, device = cairo_pdf)

Chart2R <- Chart2_data %>%
  ggplot() +
    geom_histogram(aes(x = hpc_1000_2020, y = ..density.. * 100),
                   fill = "#C00000", bins = 50) +
    scale_x_continuous(breaks = seq(0, 100, 25),
                       labels = seq(0, 100, 25),
                       limits = c(0, 100)) +
    scale_y_continuous(breaks = seq(0, 5, 1),
                       labels = seq(0, 5, 1),
                       limits = c(0, 5),
                       expand = c(0, 0, 0.025, 0)) +
    labs(title = "",
         subtitle = "percent",
         x = "Hospitalizations per 1,000 people") +
    theme(
      plot.title = element_blank(),
      plot.subtitle = element_text(size = 9, hjust = 0),
      plot.caption = element_blank(),
      panel.background = element_rect(fill = "transparent"),
      plot.background = element_rect(fill = "transparent"),
      panel.grid.major.y =
        element_line(color = "grey", size = 0.25, linetype = "solid"),
      panel.grid.minor = element_blank(),
      panel.grid.major.x = element_blank(),
      axis.line.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.line.y = element_blank(),
      axis.title.x = element_blank(),
      axis.title.y = element_blank(),
      axis.text.x = element_text(color = "black", size = 9),
      axis.text.y = element_text(color = "black", size = 9),
      axis.ticks.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.ticks.y = element_blank(),
      axis.ticks.length.x = unit(5, "pt"),
      strip.background = element_rect(fill = NA),
      strip.text = element_text(color = "black", size = 9)) +
    annotate("text", x = 35, y = 3.5, label = "2020", color = "#C00000",
             size = 9/.pt, hjust = 0)

ggsave("Chart2R.pdf", Chart2R,
       width = 5.0, height = 5.5, units = "cm",
       dpi = 300, device = cairo_pdf)

# ------------------------------------------------------------------------

# Chart 3

pop_25 <- quantile(census$ppl_pop, 0.25, na.rm = TRUE)
pop_50 <- quantile(census$ppl_pop, 0.50, na.rm = TRUE)
pop_75 <- quantile(census$ppl_pop, 0.75, na.rm = TRUE)
```

```
Chart3L_data <- dpc_municipality_2020 %>%
  mutate(dpc_1000_2020 = dpc * 1000,
         flag = case_when(ppl_pop >= pop_75 ~ 1,
                          ppl_pop <= pop_75 & ppl_pop >= pop_50 ~ 2,
                          ppl_pop <= pop_50 & ppl_pop >= pop_25 ~ 3,
                          ppl_pop <= pop_25 ~ 4)) %>%
  mutate(Population =
           factor(flag,
                  labels = c("High", "Medium-high",
                             "Medium", "Low"))) %>%
  select(Population, dpc_1000_2020)

Chart3L <- Chart3L_data %>%
  ggplot() +
    geom_density(aes(x = dpc_1000_2020, y = ..density.. * 100,
                     color = Population)) +
    scale_x_continuous(breaks = seq(0, 20, 5),
                       labels = seq(0, 20, 5),
                       limits = c(0, 20)) +
    scale_y_continuous(breaks = seq(0, 30, 5),
                       labels = seq(0, 30, 5),
                       limits = c(0, 30),
                       expand = c(0, 0, 0.025, 0)) +
    scale_color_manual(
      values = c("#C00000", "#0070C0", "#8BC63E", "#AB937B")
      ) +
    labs(title = "",
         subtitle = "percent",
         x = "Deaths per 1,000 people") +
    theme(
      legend.position = "NONE",
      plot.title = element_blank(),
      plot.subtitle = element_text(size = 9, hjust = 0),
      plot.caption = element_blank(),
      panel.background = element_rect(fill = "transparent"),
      plot.background = element_rect(fill = "transparent"),
      panel.grid.major.y =
        element_line(color = "grey", size = 0.25, linetype = "solid"),
      panel.grid.minor = element_blank(),
      panel.grid.major.x = element_blank(),
      axis.line.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.line.y = element_blank(),
      axis.title.x = element_blank(),
      axis.title.y = element_blank(),
      axis.text.x = element_text(color = "black", size = 9),
      axis.text.y = element_text(color = "black", size = 9),
      axis.ticks.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.ticks.y = element_blank(),
      axis.ticks.length.x = unit(5, "pt"),
      strip.background = element_rect(fill = NA),
      strip.text = element_text(color = "black", size = 9)) +
    annotate("text", x = 9.5, y = 26.2, label = "Population:", color = "black",
             size = 9/.pt, hjust = 0) +
```

```
    annotate("text", x = 9.5, y = 23.7, label = "High", color = "#C00000",
             size = 9/.pt, hjust = 0) +
    annotate("text", x = 9.5, y = 21.2, label = "Medium-high", color = "#0070C0",
             size = 9/.pt, hjust = 0) +
    annotate("text", x = 9.5, y = 18.7, label = "Medium", color = "#8BC63E",
             size = 9/.pt, hjust = 0) +
    annotate("text", x = 9.5, y = 16.2, label = "Low", color = "#AB937B",
             size = 9/.pt, hjust = 0)


ggsave("Chart3L.pdf", Chart3L,
       width = 8.0, height = 5.3, units = "cm",
       dpi = 300, device = cairo_pdf)

Chart3R_data <- hpc_municipality_2020 %>%
  mutate(hpc_1000_2020 = hpc * 1000,
         flag = case_when(ppl_pop >= pop_75 ~ 1,
                          ppl_pop <= pop_75 & ppl_pop >= pop_50 ~ 2,
                          ppl_pop <= pop_50 & ppl_pop >= pop_25 ~ 3,
                          ppl_pop <= pop_25 ~ 4)) %>%
  mutate(Population =
           factor(flag,
                  labels = c("High", "Medium-high",
                             "Medium", "Low"))) %>%
  select(municipality, ppl_pop,Population, hpc_1000_2020)

Chart3R <- Chart3R_data %>%
  ggplot() +
    geom_density(aes(x = hpc_1000_2020, y = ..density.. * 100,
                     color = Population)) +
    scale_x_continuous(breaks = seq(0, 100, 20),
                       labels = seq(0, 100, 20),
                       limits = c(0, 100)) +
    scale_y_continuous(breaks = seq(0, 5, 1),
                       labels = seq(0, 5, 1),
                       limits = c(0, 5),
                       expand = c(0, 0, 0.025, 0)) +
    scale_color_manual(
      values = c("#C00000", "#0070C0", "#8BC63E", "#AB937B")
      ) +
    labs(title = "",
         subtitle = "percent",
         x = "Deaths per 1,000 people") +
    theme(
      legend.position = "NONE",
      plot.title = element_blank(),
      plot.subtitle = element_text(size = 9, hjust = 0),
      plot.caption = element_blank(),
      panel.background = element_rect(fill = "transparent"),
      plot.background = element_rect(fill = "transparent"),
      panel.grid.major.y =
        element_line(color = "grey", size = 0.25, linetype = "solid"),
      panel.grid.minor = element_blank(),
```

```
      panel.grid.major.x = element_blank(),
      axis.line.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.line.y = element_blank(),
      axis.title.x = element_blank(),
      axis.title.y = element_blank(),
      axis.text.x = element_text(color = "black", size = 9),
      axis.text.y = element_text(color = "black", size = 9),
      axis.ticks.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.ticks.y = element_blank(),
      axis.ticks.length.x = unit(5, "pt"),
      strip.background = element_rect(fill = NA),
      strip.text = element_text(color = "black", size = 9))

ggsave("Chart3R.pdf", Chart3R,
       width = 8.0, height = 5.3, units = "cm",
       dpi = 300, device = cairo_pdf)
```

```
pns_2019 <- readRDS("DL/PNS_2019.rds")

pns_subset <- pns_2019 %>%
  select(V0001, J001, J00101, V00281) %>%
  mutate(V0001 = as.numeric(V0001),
         J001 = as.numeric(J001),
         J00101 = as.numeric(J00101)) %>%
  mutate(J001_weighted = J001 * V00281,
         J00101_weighted = J00101 * V00281) %>%
  group_by(V0001) %>%
  summarise(J001_state = sum(J001_weighted) / sum(V00281),
            J00101_state = sum(J00101_weighted) / sum(V00281))
colnames(pns_subset) <- c("state", "J001", "J00101")

# Calculate the national average

pns_national <- pns_2019 %>%
  select(V0001, J001, J00101, V00281) %>%
  mutate(V0001 = as.numeric(V0001),
         # J001 = as.numeric(J001),
         J00101 = as.numeric(J00101)) %>%
  mutate(# J001_weighted = J001 * V00281,
         J00101_weighted = J00101 * V00281) %>%
  summarise(# J001_national = sum(J001_weighted) / sum(V00281),
            J00101_national = sum(J00101_weighted) / sum(V00281))
# J001_national <- pns_national %>%
#   select(J001_national) %>%
#   as.numeric()
J00101_national <- pns_national %>%
  select(J00101_national) %>%
  as.numeric()

census_state <- census %>%
  mutate(hh_room_density = hh_room_density * hh_households,
         hh_household_size = hh_household_size * hh_households,
         hh_wall_masonry = hh_wall_masonry * hh_households,
```

```
          hh_wall_rigged_wood = hh_wall_rigged_wood * hh_households,
          hh_wall_taipa = hh_wall_taipa * hh_households,
          hh_wall_used_wood = hh_wall_used_wood * hh_households,
          hh_wall_straw = hh_wall_straw * hh_households,
          hh_wall_other = hh_wall_other * hh_households,
          hh_no_wall = hh_no_wall * hh_households,
          hh_toilets = hh_toilets * hh_households,
          hh_sewage = hh_sewage * hh_households,
          hh_water = hh_water * hh_households,
          hh_garbage = hh_garbage * hh_households,
          hh_electricity = hh_electricity * hh_households,
          ppl_male = ppl_male * ppl_pop,
          ppl_age = ppl_age * ppl_pop,
          ppl_race_white = ppl_race_white * ppl_pop,
          ppl_race_black = ppl_race_black * ppl_pop,
          ppl_race_asian = ppl_race_asian * ppl_pop,
          ppl_race_mixed = ppl_race_mixed * ppl_pop,
          ppl_race_indigenous = ppl_race_indigenous * ppl_pop) %>%
  group_by(state) %>%
  summarise(hh_households = sum(hh_households, na.rm = TRUE),
            hh_room_density = sum(hh_room_density, na.rm = TRUE) / hh_households,
            hh_household_size = sum(hh_household_size, na.rm = TRUE) / hh_households,
            hh_wall_masonry = sum(hh_wall_masonry, na.rm = TRUE) / hh_households,
            hh_wall_rigged_wood = sum(hh_wall_rigged_wood, na.rm = TRUE) / hh_households,
            hh_wall_taipa = sum(hh_wall_taipa, na.rm = TRUE) / hh_households,
            hh_wall_used_wood = sum(hh_wall_used_wood, na.rm = TRUE) / hh_households,
            hh_wall_straw = sum(hh_wall_straw, na.rm = TRUE) / hh_households,
            hh_wall_other = sum(hh_wall_other, na.rm = TRUE) / hh_households,
            hh_no_wall = sum(hh_no_wall, na.rm = TRUE) / hh_households,
            hh_toilets = sum(hh_toilets, na.rm = TRUE) / hh_households,
            hh_sewage = sum(hh_sewage, na.rm = TRUE) / hh_households,
            hh_water = sum(hh_water, na.rm = TRUE) / hh_households,
            hh_garbage = sum(hh_garbage, na.rm = TRUE) / hh_households,
            hh_electricity = sum(hh_electricity, na.rm = TRUE) / hh_households,
            ppl_pop = sum(ppl_pop, na.rm = TRUE),
            ppl_male = sum(ppl_male, na.rm = TRUE) / ppl_pop,
            ppl_age = sum(ppl_age, na.rm = TRUE) / ppl_pop,
            ppl_race_white = sum(ppl_race_white, na.rm = TRUE) / ppl_pop,
            ppl_race_black = sum(ppl_race_black, na.rm = TRUE) / ppl_pop,
            ppl_race_asian = sum(ppl_race_asian, na.rm = TRUE) / ppl_pop,
            ppl_race_mixed = sum(ppl_race_mixed, na.rm = TRUE) / ppl_pop,
            ppl_race_indigenous = sum(ppl_race_indigenous, na.rm = TRUE) / ppl_pop)

# ------------------------------------------------------------------------

# Chart 4

census_tmp <- census %>%
  select(state, state_name) %>%
  distinct(state, .keep_all = TRUE)

vector_state <- unlist(census_tmp$state_name)
```

```r
Chart4_data <- pns_subset %>%
  left_join(census_tmp, by = c("state")) %>%
  mutate(state_name = factor(state_name,
                         levels = vector_state))

rm(list = c("census_tmp", "vector_state"))

Chart4 <- Chart4_data %>%
  ggplot() +
    geom_point(aes(x = state_name, y = J00101),
                   color = "#C00000", size = 1.5) +
    geom_hline(yintercept = J00101_national, color = "#0070C0",
               linetype = "dashed", size = 0.75) +
    scale_x_discrete(expand = c(0.025, 0)) +
    scale_y_continuous(breaks = seq(1.0, 2.4, 0.2),
                       labels = sprintf("%.1f", seq(1.0, 2.4, 0.2)),
                       limits = c(1.0, 2.4),
                       expand = c(0, 0, 0.025, 0)) +
    labs(title = "",
         subtitle =
           paste0("Subjective health condition ",
                  "(1: very good, 2: good, 3: regular, 4: bad, 5: very bad)"),
         x = "") +
    theme(
      plot.title = element_blank(),
      plot.subtitle = element_text(size = 9, hjust = 0),
      plot.caption = element_blank(),
      panel.background = element_rect(fill = "transparent"),
      plot.background = element_rect(fill = "transparent"),
      panel.grid.major.y =
        element_line(color = "grey", size = 0.25, linetype = "solid"),
      panel.grid.minor = element_blank(),
      panel.grid.major.x = element_blank(),
      axis.line.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.line.y = element_blank(),
      axis.title.x = element_blank(),
      axis.title.y = element_blank(),
      axis.text.x = element_blank(),
      axis.text.y = element_text(color = "black", size = 9),
      axis.ticks.x = element_blank(),
      axis.ticks.y = element_blank()
      ) +
    annotate("text", x = 27, y = 2.3, label = "Dotted line: National average",
             color = "#0070C0", size = 9/.pt, hjust = 1)

annotate_state_labels <- function(i){
    annotate("text", x = i, y = 1.015, label = Chart4_data$state_name[i], color = "black",
             angle = "90", size = 9/.pt, hjust = 0)
  }

state_labels <- lapply(c(1:27), annotate_state_labels)

Chart4 <- Chart4 + state_labels
```

```
ggsave("Chart4.pdf", Chart4,
       width = 16, height = 5.3, units = "cm",
       dpi = 300, device = cairo_pdf)

rm(annotate_state_labels)
```

```
load("cnes_1_professionals.Rda")
load("cnes_2_facilities.Rda")
load("cnes_3_equipment.Rda")

# ------------------------------------------------------------------

for (i in 1:3){

  vars <- ncol(get(paste0("cnes_", i)))

  tmp1 <- get(paste0("cnes_", i)) %>%
    select(c(3, 5:vars)) %>%
    filter(year == 2020) %>%
    select(-year)
  assign(paste0("cnes_", i, "_2020"), tmp1)

  vars <- ncol(get(paste0("cnes_", i, "_2020")))

  tmp2 <- get(paste0("cnes_", i, "_2020")) %>%
    summarise(n = n(),
              across(c(2:vars), ~ sum(!is.na(.x)))) %>%
    summarise(across(c(2:vars), ~ .x / n)) %>%
    pivot_longer(cols = everything()) %>%
    # Keep variables that more than half of the observations are missing
    filter(value > 0.5)
  assign(paste0("cnes_", i, "_vars"), tmp2)

  tmp3 <- get(paste0("cnes_", i, "_vars")) %>%
    pull(name)
  assign(paste0("cnes_", i, "_keep"), tmp3)

 rm(list = c("tmp1", "tmp2", "tmp3", paste0("cnes_", i, "_vars")))

}

# ------------------------------------------------------------------

# Chart 5

Chart5_data <- census %>%
  select(c(1:4, 20)) %>%
  left_join(cnes_1_2020, by = "municipality") %>%
  left_join(cnes_2_2020, by = "municipality") %>%
  left_join(cnes_3_2020, by = "municipality") %>%
  mutate(across(c(6:61), ~ .x / ppl_pop * 100000),
         flag = case_when(ppl_pop >= pop_75 ~ 1,
                          ppl_pop <= pop_75 & ppl_pop >= pop_50 ~ 2,
```

```
                          ppl_pop <= pop_50 & ppl_pop >= pop_25 ~ 3,
                          ppl_pop <= pop_25 ~ 4)) %>%
  mutate(Population =
           factor(flag,
                  labels = c("High", "Medium-high",
                             "Medium", "Low"))) %>%
  filter(!is.na(Population))

Chart5L <- Chart5_data %>%
  ggplot() +
    geom_density(aes(x = general_practitioner, y = ..density.. * 100,
                     color = Population)) +
    scale_x_continuous(breaks = seq(0, 300, 100),
                       labels = seq(0, 300, 100),
                       limits = c(0, 300)) +
    scale_y_continuous(breaks = seq(0, 2, 0.5),
                       labels = c("0.0", "0.5", "1.0", "1.5", "2.0"),
                       limits = c(0, 2),
                       expand = c(0, 0, 0.025, 0)) +
    scale_color_manual(
      values = c("#C00000", "#0070C0", "#8BC63E", "#AB937B")
      ) +
    labs(title = "",
         subtitle = "percent",
         x = "Deaths per 1,000 people") +
    theme(
      legend.position = "NONE",
      plot.title = element_blank(),
      plot.subtitle = element_text(size = 9, hjust = 0),
      plot.caption = element_blank(),
      panel.background = element_rect(fill = "transparent"),
      plot.background = element_rect(fill = "transparent"),
      panel.grid.major.y =
        element_line(color = "grey", size = 0.25, linetype = "solid"),
      panel.grid.minor = element_blank(),
      panel.grid.major.x = element_blank(),
      axis.line.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.line.y = element_blank(),
      axis.title.x = element_blank(),
      axis.title.y = element_blank(),
      axis.text.x = element_text(color = "black", size = 9),
      axis.text.y = element_text(color = "black", size = 9),
      axis.ticks.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.ticks.y = element_blank(),
      axis.ticks.length.x = unit(5, "pt"),
      strip.background = element_rect(fill = NA),
      strip.text = element_text(color = "black", size = 9)) +
    annotate("text", x = 100, y = 1.20, label = "Population:", color = "black",
             size = 9/.pt, hjust = 0) +
    annotate("text", x = 100, y = 1.05, label = "High", color = "#C00000",
             size = 9/.pt, hjust = 0) +
    annotate("text", x = 100, y = 0.90, label = "Medium-high", color = "#0070C0",
             size = 9/.pt, hjust = 0) +
```

```
    annotate("text", x = 100, y = 0.75, label = "Medium", color = "#8BC63E",
             size = 9/.pt, hjust = 0) +
    annotate("text", x = 100, y = 0.60, label = "Low", color = "#AB937B",
             size = 9/.pt, hjust = 0)

ggsave("Chart5L.pdf", Chart5L,
       width = 5.0, height = 5.5, units = "cm",
       dpi = 300, device = cairo_pdf)

Chart5R <- Chart5_data %>%
  ggplot() +
    geom_density(aes(x = GENERAL.HOSPITAL, y = ..density.. * 100,
                     color = Population)) +
    scale_x_continuous(breaks = seq(0, 60, 20),
                       labels = seq(0, 60, 20),
                       limits = c(0, 60)) +
    scale_y_continuous(breaks = seq(0, 30, 5),
                       labels = seq(0, 30, 5),
                       limits = c(0, 30),
                       expand = c(0, 0, 0.025, 0)) +
    scale_color_manual(
      values = c("#C00000", "#0070C0", "#8BC63E", "#AB937B")
      ) +
    labs(title = "",
         subtitle = "percent",
         x = "Deaths per 1,000 people") +
    theme(
      legend.position = "NONE",
      plot.title = element_blank(),
      plot.subtitle = element_text(size = 9, hjust = 0),
      plot.caption = element_blank(),
      panel.background = element_rect(fill = "transparent"),
      plot.background = element_rect(fill = "transparent"),
      panel.grid.major.y =
        element_line(color = "grey", size = 0.25, linetype = "solid"),
      panel.grid.minor = element_blank(),
      panel.grid.major.x = element_blank(),
      axis.line.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.line.y = element_blank(),
      axis.title.x = element_blank(),
      axis.title.y = element_blank(),
      axis.text.x = element_text(color = "black", size = 9),
      axis.text.y = element_text(color = "black", size = 9),
      axis.ticks.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.ticks.y = element_blank(),
      axis.ticks.length.x = unit(5, "pt"),
      strip.background = element_rect(fill = NA),
      strip.text = element_text(color = "black", size = 9))

ggsave("Chart5R.pdf", Chart5R,
       width = 5.0, height = 5.5, units = "cm",
       dpi = 300, device = cairo_pdf)
```

```r
# Now comes the estimation part

data_estimation <- census_state %>%
  left_join(pns_subset, by = c("state")) %>%
  select(state, J001, J00101, everything()) %>%
  mutate(y = J00101) # Change to J001 if necessary

# Split the dataset

set.seed(0812)
data_estimation_split <- initial_split(data_estimation, prop = 0.70)

data_estimation_train <- training(data_estimation_split)
data_estimation_test <- testing(data_estimation_split)

# Prepare the short model

f_short <- y ~ hh_room_density + hh_sewage + hh_water + ppl_age

# Prepare the long model

rm_vars <- c("state", "J001", "J00101", "municipality", "municipality_name",
             "hh_households", "ppl_pop",
             "hh_household_size", "hh_wall_masonry",
             "hh_wall_rigged_wood", "hh_wall_taipa", "hh_wall_used_wood",
             "hh_wall_straw", "hh_wall_other")

long_vars  <- setdiff(colnames(data_estimation), rm_vars)

f_long_rhs <- str_c(long_vars, collapse = " + ")

f_long   <- as.formula(str_c("y ~ ", f_long_rhs))

# Prepare the LASSO model

vector_y_train <- data_estimation_train$y
matrix_x_train <- as.matrix(data_estimation_train[, c(5:18, 20:26)])

# Training ----------------------------------------------------------

# Short model
fit_simple <- lm(f_short, data_estimation_train)

# Long model
fit_kitchen <- lm(f_long, data_estimation_train)

# LASSO model
fit_lasso <- cv.glmnet(x = matrix_x_train, y = vector_y_train)

# Evaluate the in-sample RMSEs

data_estimation_in <- data_estimation_train %>%
  mutate(
```

```
    # Short model
    pred_short = predict(fit_simple, newdata = data_estimation_train),
    # Long model
    pred_long  = predict(fit_kitchen, newdata = data_estimation_train),
    # LASSO
    pred_lasso = predict(fit_lasso, newx = matrix_x_train, s = "lambda.min")
  ) %>%
  select(y, pred_short, pred_long, pred_lasso)

rmse_in <- data_estimation_in %>%
  summarise(
    simple  = sqrt(mean((y - pred_short)^2)),
    kitchen = sqrt(mean((y -  pred_long)^2)),
    lasso   = sqrt(mean((y - pred_lasso)^2))
  ) %>%
  pivot_longer(cols = everything())
colnames(rmse_in) <- c("name", "rmse_in")

rmse_in %>%
  mutate(name = case_when(name == "simple" ~ "Short",
                          name == "kitchen" ~ "Long",
                          name == "lasso" ~ "LASSO"),
         rmse_in = round(rmse_in, digits = 3)) %>%
  gt() %>%
  cols_label(
    name = "Model",
    rmse_in = "In-sample RMSE",
  ) %>%
  cols_align(
    align = "center",
    columns = rmse_in
  )
```

Model	In-sample RMSE
Short	0.039
Long	0.016
LASSO	0.024

```
# Testing ------------------------------------------------------------

matrix_x_test <- as.matrix(data_estimation_test[, c(5:18, 20:26)])

data_estimation_out <- data_estimation_test %>%
  mutate(
    # Simple model
    pred_short = predict(fit_simple, newdata = data_estimation_test),
    # Kitchen-sink model
    pred_long = predict(fit_kitchen, newdata = data_estimation_test),
    # Lasso
    pred_lasso = predict(fit_lasso, newx = matrix_x_test, s = "lambda.min")
  ) %>%
  select(y, pred_short, pred_long, pred_lasso)
```

```r
rmse_out <- data_estimation_out %>%
  summarise(
    simple  = sqrt(mean((y - pred_short)^2)),
    kitchen = sqrt(mean((y -  pred_long)^2)),
    lasso   = sqrt(mean((y - pred_lasso)^2))
  ) %>%
  pivot_longer(cols = everything())
colnames(rmse_out) <- c("name", "rmse_out")

rmse_in %>%
  left_join(rmse_out, by = c("name")) %>%
  mutate(name = case_when(name == "simple" ~ "Short",
                          name == "kitchen" ~ "Long",
                          name == "lasso" ~ "LASSO"),
         rmse_in = round(rmse_in, digits = 3),
         rmse_out = round(rmse_out, digits = 3)) %>%
  gt() %>%
  cols_label(
    name = "Model",
    rmse_in = "In-sample RMSE",
    rmse_out = "Out-of-sample RMSE"
  ) %>%
  cols_align(
    align = "center",
    columns = rmse_in
  ) %>%
  cols_align(
    align = "center",
    columns = rmse_out
  )
```

Model	In-sample RMSE	Out-of-sample RMSE
Short	0.039	0.079
Long	0.016	0.092
LASSO	0.024	0.062

```r
# Predicting ----------------------------------------------------------

matrix_x_pred <- as.matrix(census[, c(6:19, 21:27)])

census_with_pred <- census %>%
  mutate(
    pred_lasso = predict(fit_lasso, newx = matrix_x_pred, s = "lambda.min")
  )
```

```r
load("ieps_municipality.Rda")

tmp <- ieps_municipality %>%
  mutate(municipality = codmun,
         year = ano,
         income = renda_dom_pc) %>%
```

```
  filter(year == 2020) %>%
  select(municipality, income)

# ------------------------------------------------------------------------

pred_lasso <- census_with_pred$pred_lasso[, 1]

mphi_deaths <- dpc_municipality_2020[, c(3, 27)]
mphi_hospitalization <- hpc_municipality_2020[, c(3, 29)]
mphi_pns <- tibble(census_with_pred[, 3], pred_lasso)

rm(pred_lasso)

mphi <- census[, c(1:5, 20)] %>%
  left_join(mphi_deaths, by = c("municipality")) %>%
  left_join(mphi_hospitalization, by = c("municipality")) %>%
  left_join(mphi_pns, by = c("municipality")) %>%
  left_join(tmp, by = c("municipality")) %>%
  filter(!is.na(hh_households) | !is.na(ppl_pop)) %>%
  filter(!is.na(pred_lasso)) %>%
  mutate(income = income * (-1)) %>%
  mutate(deaths =
           (dpc - mean(dpc, na.rm = TRUE)) / sd(dpc, na.rm = TRUE),
         hospitalization =
           (hpc - mean(hpc, na.rm = TRUE)) / sd(hpc, na.rm = TRUE),
         health_score =
           (pred_lasso - mean(pred_lasso, na.rm = TRUE)) / sd(pred_lasso, na.rm = TRUE),
         income =
           (income - mean(income, na.rm = TRUE)) / sd(income, na.rm = TRUE)) %>%
  mutate(deaths = deaths * 10 + 50,
         hospitalization = hospitalization * 10 + 50,
         health_score = health_score * 10 + 50,
         income = income * 10 + 50) %>%
  rowwise() %>%
  mutate(mphi = mean(c(deaths, hospitalization, health_score, income), na.rm = TRUE))
```

```
# Written in an extremely short period of time...

load("municipality_codes.Rda")

muni <- read_municipality(year = 2020, showProgress = FALSE)

tmp <- municipality_codes %>%
  arrange(municipality)
code <- tibble(name_muni = muni$name_muni,
               municipality_name = tmp$municipality_name)

tmp <- mphi[, c(4, 14)]

muni <- muni %>%
  left_join(code, by = c("name_muni")) %>%
  left_join(tmp, by = c("municipality_name")) %>%
  mutate(mphi2 = case_when(mphi <= 40 ~ 40,
```

```
                          mphi > 40 & mphi < 60 ~ mphi,
                          mphi >= 60 ~ 60))
colnames(muni)[11] <- "MPHI"
# ggplot(data = muni, (aes(x = muni$MPHI))) + geom_density()
Chart7 <- ggplot() +
  geom_sf(data = muni, aes(fill = MPHI), color = NA) +
  scale_fill_gradient2(low = "blue", mid = "white", high = "red",
                       midpoint = 50, na.value = NA) +
  theme(
      plot.title = element_blank(),
      plot.subtitle = element_blank(),
      plot.caption = element_blank(),
      panel.background = element_rect(fill = "transparent"),
      plot.background = element_rect(fill = "transparent"),
      panel.grid.major = element_blank(),
      panel.grid.minor = element_blank(),
      axis.line.x = element_blank(),
      axis.line.y = element_blank(),
      axis.title.x = element_blank(),
      axis.title.y = element_blank(),
      axis.text.x = element_blank(),
      axis.text.y = element_blank(),
      axis.ticks.x = element_blank(),
      axis.ticks.y = element_blank()
      )

ggsave("Chart7.pdf", Chart7,
       width = 15, height = 15, units = "cm",
       dpi = 300, device = cairo_pdf)
```

```
infra <- mphi[, c(1:4, 6, 14)] %>%
  left_join(cnes_1_2020, by = "municipality") %>%
  left_join(cnes_2_2020, by = "municipality") %>%
  left_join(cnes_3_2020, by = "municipality") %>%
  select(state, state_name, municipality, municipality_name,
         ppl_pop, mphi, all_of(cnes_1_keep), all_of(cnes_2_keep),
         all_of(cnes_3_keep)) %>%
  mutate(across(c(7:21), ~ .x / ppl_pop * 100000)) # Standardize
colnames(infra)[5] <- "population"

urban <- quantile(infra$population, 0.50)
infra_urban <- infra %>%
  filter(population >= urban)
infra_rural <- infra %>%
  filter(population < urban)

# Overall

overall_top <- quantile(infra$mphi, 0.25)
overall_bottom <- quantile(infra$mphi, 0.75)

infra_sum <- infra %>%
  mutate(flag = case_when(mphi <= overall_top ~ "top",
```

```
                          mphi >= overall_bottom ~ "bottom")) %>%
  filter(!is.na(flag)) %>%
  group_by(flag) %>%
  summarise(across(c(7:21), ~ median(.x, na.rm = TRUE)))

write_csv(infra_sum, "infra_sum.csv")

# Urban

urban_top <- quantile(infra_urban$mphi, 0.25)
urban_bottom <- quantile(infra_urban$mphi, 0.75)

infra_urban_sum <- infra_urban %>%
  mutate(flag = case_when(mphi <= urban_top ~ "top",
                          mphi >= urban_bottom ~ "bottom")) %>%
  filter(!is.na(flag)) %>%
  group_by(flag) %>%
  summarise(across(c(7:21), ~ median(.x, na.rm = TRUE)))

write_csv(infra_urban_sum, "infra_urban_sum.csv")

# # Rural
#
# rural_top <- quantile(infra_rural$mphi, 0.25)
# rural_bottom <- quantile(infra_rural$mphi, 0.75)
#
# infra_rural_sum <- infra_rural %>%
#   mutate(flag = case_when(mphi <= rural_top ~ "top",
#                           mphi >= rural_bottom ~ "bottom")) %>%
#   filter(!is.na(flag)) %>%
#   group_by(flag) %>%
#   summarise(across(c(7:21), ~ median(.x, na.rm = TRUE)))
```

```
mphi_top20 <- mphi %>%
  ungroup() %>%
  arrange(mphi) %>%
  slice(1:20) %>%
  select(state_name, municipality_name, mphi)

mphi_bottom20 <- mphi %>%
  ungroup() %>%
  arrange(desc(mphi)) %>%
  slice(1:20) %>%
  select(state_name, municipality_name, mphi)
```

```
# Prepare the data for plots

gdata_sih <- sih

gdata_sih[is.na(gdata_sih)] <- 0

gdata_sih <- gdata_sih %>%
  mutate(ch_21 = ifelse(year == 2020, NA_integer_, ch_21)) %>%
```

```
  mutate(total = rowSums(.[6:26], na.rm = TRUE)) %>%
  filter(total != 0)

for (i in 1:21) {

    num <- formatC(i, width = 2, flag = 0)
    col_name <- paste0("p_ch", num)

    gdata_sih <- gdata_sih %>%
      mutate(tmp = get(paste0("ch_", num)) / total * 100)

    colnames(gdata_sih)[which(names(gdata_sih) == "tmp")] <- col_name

}

gdata_sih_long <- gdata_sih[c(5, 28:48)] %>%
  gather(chapter, share, p_ch01:p_ch21, factor_key = FALSE) %>%
  mutate(chapter =
           as.factor(
             as.numeric(str_sub(chapter, start = 5, end = 6)))) %>%
  # Remove 0 percent at my discretion
  filter(share != 0)

# ------------------------------------------------------------------

# Create a plot

filter_chapter <- gdata_sih_long %>%
  filter(year == 2010) %>%
  group_by(chapter) %>%
  summarise(median = median(share)) %>%
  filter(median > 5) %>%
  select(chapter) %>%
  unlist()

ChartA4 <- gdata_sih_long %>%
  filter(year == 2010 & chapter %in% filter_chapter) %>%
  mutate(chapter =
           case_when(
             chapter == 1 ~ "Infectious diseases",
             chapter == 9 ~ "Circulatory system diseases",
             chapter == 10 ~ "Respiratory system diseases",
             chapter == 11 ~ "Digestive system diseases",
             chapter == 14 ~ "Genitourinary system diseases",
             chapter == 15 ~ "Pregnancy, Childbirth & Puerperium",
             chapter == 19 ~ "Injury, poisoning, etc."
             )
         ) %>%
  ggplot(aes(x = factor(chapter,
                        level = c("Pregnancy, Childbirth & Puerperium",
                                  "Respiratory system diseases",
                                  "Circulatory system diseases",
                                  "Digestive system diseases",
```

```
                                  "Infectious diseases",
                                  "Injury, poisoning, etc.",
                                  "Genitourinary system diseases")
                        ),
           y = share)) +
    stat_boxplot(geom = "errorbar", color = "#C00000", size = 0.5) +
    geom_boxplot(outlier.shape = NA, color = "#C00000", size = 0.5) +
    scale_x_discrete(expand = c(0.025, 0.5)) +
    scale_y_continuous(breaks = seq(0, 50, 10),
                       labels = seq(0, 50, 10),
                       limits = c(0, 50),
                       expand = c(0, 0, 0.025, 0)) +
    labs(title = "",
         subtitle = "",
         y = "Percentage share of hospitalization") +
    coord_flip() +
    theme(
      plot.title = element_blank(),
      plot.subtitle = element_blank(),
      plot.caption = element_blank(),
      panel.background = element_rect(fill = "transparent"),
      plot.background = element_rect(fill = "transparent"),
      panel.grid.major.y =
        element_blank(),
      panel.grid.minor = element_blank(),
      panel.grid.major.x =
        element_line(color = "grey", size = 0.25, linetype = "solid"),
      axis.line.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.line.y = element_blank(),
      axis.title.x = element_text(color = "black", size = 9),
      axis.title.y = element_blank(),
      axis.text.x = element_text(color = "black", size = 9),
      axis.text.y = element_text(color = "black", size = 9),
      axis.ticks.x = element_line(color = "black", size = 0.5, linetype = "solid"),
      axis.ticks.y = element_blank(),
      axis.ticks.length.x = unit(5, "pt"),
      strip.background = element_rect(fill = NA),
      strip.text = element_text(color = "black", size = 9))

ggsave("ChartA4.pdf", ChartA4,
       width = 16, height = 5.3, units = "cm",
       dpi = 300, device = cairo_pdf)
```