RSV_PHIS_Data_Clean

Set up the workspace.

```
rm(list=ls(all.names=TRUE))
setwd("/Users/bpetros/Desktop/PHIS")
libs <- c("cowplot", "data.table", "ggplot2", "lubridate", "scales", "tidyverse")
invisible(lapply(libs, function(x) suppressPackageStartupMessages(library(x, character.only = TRUE))))
options(stringsAsFactors=FALSE, scipen = 999)
theme_set(theme_classic())
```

Read in all of the input files downloaded from the PHIS database as well as the lists of influenza and SARS-CoV-2 diagnostic codes.

```
# discharge diagnoses per patient encounter
dx <- read.csv("PHIS Data/RSV_All_Diagnosis_updated.csv",
               colClasses = c("Billing_Number" = "NULL", "dx_version" = "NULL"))

# patient & hospital metadata per patient encounter
pt <- read.csv("PHIS Data/RSV_All_PatientAbstract_updated.csv")

# readmissions occurring within 30 days of d/c from RSV-associated admission
readmit <- read.csv("PHIS Data/RSV_All_Readmit_updated.csv")

# pt-specific testing
tests = read.csv("PHIS Data/LabSummaryPackage_RSVonly.csv")

# ptest-specific testing
alltests = read.csv("PHIS Data/AllRSVTest_LabSummary.csv")

# ptest-specific testing (obs unit)
obstests = read.csv("PHIS Data/AllRSVTest_LabSummaryOBSonly.csv")

# pt data for all tested patients
ptest = read.csv("PHIS Data/AllRSVTest_PatientAbstract.csv")

# monthly flu IP and ED data
flu_ed <- read.csv("PHIS Data/Flu_ED.csv", nrow = 4196)
flu_ip <- read.csv("PHIS Data/Flu_IP.csv", nrow = 4540)

# monthly testing data
test_ed <- read.csv("PHIS Data/RSV_Testing_ED.csv", nrow = 3678)
test_ip <- read.csv("PHIS Data/RSV_Testing_IP.csv", nrow = 4246)

# monthly flu testing data
flutest_ed <- read.csv("PHIS Data/flu_testing_ed.csv", nrow = 3890)
flutest_ip <- read.csv("PHIS Data/flu_testing_ip.csv", nrow = 4267)
```

```r
# monthly non-multipanel testing data
test_ed_m <- read.csv("PHIS Data/RSV_Testing_ED_multi.csv", nrow = 917)
test_ip_m <- read.csv("PHIS Data/RSV_Testing_IP_multi.csv", nrow = 978)

# all ICD-9 & ICD-10 codes in data specific for current influenza infection
flu_codes <- c("4871", "4870", "48812", "48882", "4878", "48889", "48881", "48809",
               "48811", "48802", "48801", "J101", "J1001", "J111", "J102", "48819",
               "J1008", "J1100", "J09X1", "J1108", "J1089", "J1000", "J09X2", "J09X3",
               "J1083", "J1189", "J09X9", "J1183", "J1081", "J112", "J1181", "J1082", "J1182")

# all ICD-10 codes in data specific for current COVID-19 infection
covid_codes <- "U071"
```

Clean the patient data frame, which contains information on all patients diagnosed with RSV.

```r
# filter pt data for columns of interest
pt = pt[c("Hospital_Name", "Hospital_Number", "Medical_Record_Number", "Discharge_ID",
          "Patient_Type_Title", "Admit_Age_In_Days", "Admit_Age_In_Years", "DOB",
          "Gender_Title", "Ethnicity_Title", "Race_White", "Race_Black", "Race_Asian",
          "Race_Pacific_Islander", "Race_American_Indian","Race_Other", "Length_Of_Stay",
          "Disposition_Title", "ED_Charge_Flag", "NICU_Flag", "ICU_Flag", "Mechanical_Vent_Flag",
          "ECMO_Flag", "Admit_Date", "Discharge_Date", "Principal_Dx", "Principal_Dx_Title",
          "Census_Division", "Census_Region", "Complex_Chronic_Condition_Flag",
          "Premature_And_Neonatal_Flag", "Discharge_Mortality_Flag")]

# convert date columns to Date type
pt$Admit_Date <- as.Date(pt$Admit_Date, format = "%m/%d/%Y")
pt$Discharge_Date <- as.Date(pt$Discharge_Date, format = "%m/%d/%Y")

# remove entries with admission + discharge dates before 07-01-2013
pt <- pt %>% filter(Admit_Date >= as.Date("2013-07-01"))
pt <- pt %>% filter(Discharge_Date >= as.Date("2013-07-01"))

# remove entries with negative admit age or patients >= 18
pt <- pt %>% filter(Admit_Age_In_Days >= 0)
pt <- pt %>% filter(Admit_Age_In_Years <= 17)

# group the data by hospital
hospital_groups <- split(pt, pt$Hospital_Name)
# list to store hospitals reporting data every year
remove_hospitals <- c()
# all years in study period
full_years = unique(format(as.Date(pt$Discharge_Date), "%Y"))
# check each hospital for data from every year in study period
for (hospital in names(hospital_groups)) {
  hospital_data <- hospital_groups[[hospital]]
  # extract years from the Date column
  years <- format(as.Date(hospital_data$Admit_Date), "%Y")
  # check if data available every year
  all_years_present <- all(full_years %in% unique(years))
  # check if both "Inpatient" and "ED Visit" logged each year
  if (all_years_present) {
    years_with_inpatient <- unique(years[hospital_data$Patient_Type_Title == "Inpatient"])
```

```
    years_with_ed_visit <- unique(years[hospital_data$Patient_Type_Title == "ED Visit"])
    if (!all(full_years %in% years_with_inpatient) || !all(full_years %in% years_with_ed_visit)) {
      remove_hospitals <- c(remove_hospitals, hospital)}
  } else {
    remove_hospitals <- c(remove_hospitals, hospital)}}
rm(years_with_ed_visit, years_with_inpatient)

# list hospitals marked by PHIS docs as having incomplete data over study period
incomplete = c(1008,1014,2011,2026,2033,2036,3006,3009,3010,3012,3014,3015)

# remove hospitals with missing data
pt = pt %>%
  filter(!(Hospital_Name %in% remove_hospitals)) %>%
  filter(!(Hospital_Number %in% incomplete))

rm(all_years_present, full_years, hospital, incomplete, hospital_data, hospital_groups,
   remove_hospitals, years)

# create indicator variable for patients seen in ED
pt$ED_entry = ifelse(pt$ED_Charge_Flag == "Y" | pt$Patient_Type_Title == "ED Visit", 1, 0)
```

Append influenza and SC2 co-infection information to the patient data frame.

```
# require "Discharge_ID" to be in pt data
dx$flag <- ifelse(dx$Discharge_ID %in% pt$Discharge_ID, 1, 0)
dx <- dx[dx$flag == 1,]
dx$flag <- NULL

# flag pts who also had flu
flu_pts = unique(dx[dx$Dx_Code %in% flu_codes,]$Discharge_ID)
pt$flu <- ifelse(pt$Discharge_ID %in% flu_pts, 1, 0)
rm(flu_codes, flu_pts)

# flag pts who also had SC2
sc2_pts = unique(dx[dx$Dx_Code %in% covid_codes,]$Discharge_ID)
pt$sc2 <- ifelse(pt$Discharge_ID %in% sc2_pts, 1, 0)
rm(covid_codes, dx, sc2_pts)
```

Ensure that patients who transferred from one unit of the hospital to another are only counted as one encounter in the patient data frame.

```
# flag RSV-associated readmissions
readmit$flag <- ifelse(readmit$Discharge_ID %in% pt$Discharge_ID &
                         readmit$Readmit_Discharge_ID %in% pt$Discharge_ID, 1, 0)
readmit <- readmit[readmit$flag == 1,]
readmit$flag <- NULL

# only save most recent readmission as a readmission (e.g., not readmission if 2 admissions "away")
readmit <- readmit %>%
  arrange(Days_Between) %>%
  distinct(Readmit_Discharge_ID, .keep_all = TRUE)

# isolate true readmissions from same-day intrahospital transfers
```

```r
same_day_readmit = readmit[readmit$Days_Between == 0,]
same_day_readmit = merge(same_day_readmit, pt[c("Discharge_ID", "Medical_Record_Number",
                   "Admit_Date")], by = "Discharge_ID", all.x = TRUE)
true_readmit = readmit[readmit$Days_Between > 0,]
rm(readmit)

# function to combine entries into a single admission
create_single_entry <- function(df) {
  # copy entry with the most recent Discharge_Date
  recent_entry <- df[which.max(df$Discharge_Date), ]
  # replace Patient_Type_Title with "Inpatient" if any row has entry "Inpatient"
  recent_entry$Patient_Type_Title <- ifelse(("Inpatient" %in% df$Patient_Type_Title), "Inpatient", rece
  # replace Admit_Age_In_Days and Admit_Age_In_Years with the min number across rows
  recent_entry$Admit_Age_In_Days <- min(df$Admit_Age_In_Days)
  recent_entry$Admit_Age_In_Years <- min(df$Admit_Age_In_Years)
  # if any row contains "Y", make ED_Charge_Flag, NICU_Flag, ICU_Flag, Mechanical_Vent_Flag, and ECMO_F
  recent_entry$ED_Charge_Flag <- ifelse("Y" %in% df$ED_Charge_Flag, "Y", recent_entry$ED_Charge_Flag)
  recent_entry$NICU_Flag <- ifelse("Y" %in% df$NICU_Flag, "Y", recent_entry$NICU_Flag)
  recent_entry$ICU_Flag <- ifelse("Y" %in% df$ICU_Flag, "Y", recent_entry$ICU_Flag)
  recent_entry$Mechanical_Vent_Flag <- ifelse("Y" %in% df$Mechanical_Vent_Flag, "Y", recent_entry$Mecha
  recent_entry$ECMO_Flag <- ifelse("Y" %in% df$ECMO_Flag, "Y", recent_entry$ECMO_Flag)
  # make ED_entry 1 if any entry came from the ED
  recent_entry$ED_entry = ifelse(1 %in% df$ED_entry, 1, recent_entry$ED_entry)
  # make Admit_Date the earliest date
  recent_entry$Admit_Date <- min(df$Admit_Date)
  # replace LOS with the difference in dates
  recent_entry$Length_Of_Stay <- min(as.integer(recent_entry$Discharge_Date - recent_entry$Admit_Date),
  # if any row in sc2 or flu contains 1, make it 1
  recent_entry$sc2 <- ifelse(1 %in% df$sc2, 1, recent_entry$sc2)
  recent_entry$flu <- ifelse(1 %in% df$flu, 1, recent_entry$flu)
  return(recent_entry)}

# convert pt and same_day_readmit to data.table obj for speed
setDT(pt)
setDT(same_day_readmit)
result_list <- list()
# find unique combos of Medical_Record_Number and Admit_Date
pt_xfers <- unique(same_day_readmit[, c("Medical_Record_Number", "Admit_Date")])
# iterate over each unique pt-date and combine entries into single admission
for (i in 1:nrow(pt_xfers)){
  mrn <- pt_xfers$Medical_Record_Number[i]
  dt <- pt_xfers$Admit_Date[i]
  # subset the pt data.table based on MRN and admit date
  subset_dt <- pt[(Medical_Record_Number == mrn) & (Admit_Date == dt), ]
  # create single entry from the subset data.table
  entry <- create_single_entry(subset_dt)
  # append entry to the result list
  result_list[[i]] <- entry}
# combine the entries into single data.table
result_dt <- rbindlist(result_list)
rm(dt, entry, i, mrn, pt_xfers, subset_dt, result_list)

# remove rows associated with same-day readmissions from pt
```

```r
pt <- pt[!Discharge_ID %in% c(same_day_readmit$Discharge_ID, same_day_readmit$Readmit_Discharge_ID)]
# append the edited + combined entries to pt
pt <- rbind(pt, result_dt)
pt = data.frame(pt)
rm(same_day_readmit, result_dt)

# update LOS to include 0 days (< 1 day)
pt$Length_Of_Stay = as.integer(pt$Discharge_Date - pt$Admit_Date)

# determine freq of patient re-diagnoses outside of 7d window
pt_no_readmit = pt[!pt$Discharge_ID %in% true_readmit$Readmit_Discharge_ID,]
# id duplicate rows
dups <- pt_no_readmit[duplicated(pt_no_readmit$Medical_Record_Number) |
                        duplicated(pt_no_readmit$Medical_Record_Number, fromLast = TRUE), ]

# determine number of re-diagnoses
redx = table(dups$Medical_Record_Number)
cat("Number of patients with multiple RSV admissions:", length(unique(dups$Medical_Record_Number)), "\n
```

```
## Number of patients with multiple RSV admissions: 13203
```

```r
rm(dups, pt_no_readmit, redx, true_readmit)
```

Append testing information to the patient data frame.

```r
# total number of RSV tests conducted per pt encounter
total_tests <- tests %>%
  group_by(Discharge_ID) %>%
  summarise(Num_Tests = n())

# add to pt df
pt <- pt %>%
  left_join(total_tests, by = "Discharge_ID") %>%
  mutate(Num_Tests = ifelse(is.na(Num_Tests), 0, Num_Tests))

# isolate test types
unique_lab_test_titles <- unique(tests$Lab_Test_Title)

# add number of each type of RSV test to pt df
for (lab_test_title in unique_lab_test_titles) {
  pt <- pt %>%
    left_join(
      tests %>%
        filter(Lab_Test_Title == lab_test_title) %>%
        group_by(Discharge_ID) %>%
        summarise(Count = n()) %>%
        rename(!!lab_test_title := Count),
      by = "Discharge_ID")
  # replce NAs with zeroes
  pt[[lab_test_title]][is.na(pt[[lab_test_title]])] <- 0}

# rename pt columns
```

```
pt <- pt %>%
  rename_with(~ ifelse(. == "Respiratory syncytial virus culture", "culture",
                       ifelse(. == "Respiratory syncytial virus PCR", "PCR",
                              ifelse(. == "Respiratory syncytial virus RT-PCR", "RT_PCR",
                                     ifelse(. == "Combo SARS-CoV-2 and Multiple Respiratory Viral Organi
                                            ifelse(. == "Respiratory syncytial virus antigen", "antigen
                                                   ifelse(. == "Respiratory syncytial virus DNA probe",
                                                          ifelse(. == "Respiratory syncytial virus test
rm(lab_test_title, tests, total_tests, unique_lab_test_titles)
```

Clean the test data frame, which contains information on all patients tested for RSV.

```
# filter ptest data for columns of interest
ptest = ptest[c("Hospital_Name", "Hospital_Number", "Medical_Record_Number", "Discharge_ID",
          "Patient_Type_Title", "Admit_Age_In_Days", "Admit_Age_In_Years", "DOB",
          "Gender_Title", "Ethnicity_Title", "Race_White", "Race_Black", "Race_Asian",
          "Race_Pacific_Islander", "Race_American_Indian","Race_Other", "Length_Of_Stay",
          "Disposition_Title", "ED_Charge_Flag", "NICU_Flag", "ICU_Flag", "Mechanical_Vent_Flag",
          "ECMO_Flag", "Admit_Date", "Discharge_Date", "Principal_Dx", "Principal_Dx_Title",
          "Census_Division", "Census_Region", "Complex_Chronic_Condition_Flag",
          "Premature_And_Neonatal_Flag", "Discharge_Mortality_Flag")]

# convert date columns to Date type
ptest$Admit_Date <- as.Date(ptest$Admit_Date, format = "%m/%d/%Y")
ptest$Discharge_Date <- as.Date(ptest$Discharge_Date, format = "%m/%d/%Y")

# remove entries with admission + discharge dates before 07-01-2013
ptest <- ptest %>% filter(Admit_Date >= as.Date("2013-07-01"))
ptest <- ptest %>% filter(Discharge_Date >= as.Date("2013-07-01"))

# remove entries with negative admit age or patients >= 18
ptest <- ptest %>% filter(Admit_Age_In_Days >= 0)
ptest <- ptest %>% filter(Admit_Age_In_Years <= 17)

# remove entries in hospital missing from pt
ptest <- ptest %>% filter(Hospital_Name %in% pt$Hospital_Name)

# create indicator variable for patients seen in ED
ptest$ED_entry = ifelse(ptest$ED_Charge_Flag == "Y" | ptest$Patient_Type_Title == "ED Visit", 1, 0)

# update LOS to include 0 days (< 1 day)
ptest$Length_Of_Stay = as.integer(ptest$Discharge_Date - ptest$Admit_Date)

# add positivity indicator
ptest$pos = ifelse(ptest$Discharge_ID %in% pt$Discharge_ID, 1, 0)
```

Add the number and types of RSV tests conducted on each patient to the test data frame.

```
alltests = rbind(alltests, obstests)
rm(obstests)

# total number of RSV tests conducted per Discharge_ID
total_tests <- alltests %>%
```

```
  group_by(Discharge_ID) %>%
  summarise(Num_Tests = n())

# add to ptest df
ptest <- ptest %>%
  left_join(total_tests, by = "Discharge_ID") %>%
  mutate(Num_Tests = ifelse(is.na(Num_Tests), 0, Num_Tests))

alltests$Lab_Test_Title <- gsub(".*PCR.*", "PCR", alltests$Lab_Test_Title)
alltests$Lab_Test_Title <- gsub(".*Combo SARS-CoV-2.*", "SC2_multi", alltests$Lab_Test_Title)

# isolate test types
unique_lab_test_titles <- unique(alltests$Lab_Test_Title)

# add number of each type of RSV test to pt df
for (lab_test_title in unique_lab_test_titles) {
  ptest <- ptest %>%
    left_join(
      alltests %>%
        filter(Lab_Test_Title == lab_test_title) %>%
        group_by(Discharge_ID) %>%
        summarise(Count = n()) %>%
        rename(!!lab_test_title := Count),
      by = "Discharge_ID")
  # replace NAs with zeroes
  ptest[[lab_test_title]][is.na(ptest[[lab_test_title]])] <- 0}

# rename pt columns
ptest <- ptest %>%
  rename_with(~ ifelse(. == "Respiratory syncytial virus culture", "culture",
                  ifelse(. == "Respiratory syncytial virus antigen", "antigen",
                    ifelse(. == "Respiratory syncytial virus DNA probe", "DNA_probe",
                      ifelse(. == "Respiratory syncytial virus RNA", "RNA",
                        ifelse(. == "Respiratory syncytial virus test unspecified", "unspecified", .))))
rm(lab_test_title, alltests, total_tests, unique_lab_test_titles)
```

Bin emergency department dispositions into “admitted” and “other.”

```
ed_dispo_and_write_csv <- function(df, file_path) {

  # create dispo variable
  df <- df %>%
    mutate(ED_Dispo = ifelse(
      Patient_Type_Title == "Inpatient" & ED_entry == 1,
      "ED_Admission", ifelse(
        Disposition_Title == "Admitted as an Inpatient to this Hospital",
        "ED_Admission", "Other")))
  ed_other <- df[df$ED_Dispo == "Other" & df$ED_entry == 1, ]

  # relative frequencies of non-admission dispos
  freq <- round(sort(table(ed_other$Disposition_Title) / (nrow(ed_other)), decreasing = TRUE), 4)

  # write df to .csv
```

```
  write.csv(df, file_path, row.names = FALSE)}

ed_dispo_and_write_csv(pt, "cleaned/rsv_patient.csv")
ed_dispo_and_write_csv(ptest, "cleaned/tested_patient.csv")
```

Clean the flu counts data frame, which contains monthly counts of patients diagnosed with influenza virus.

```
# strip commas and convert d/c strings into integers
flu_ed$ED_dc = as.integer(gsub(",", "", flu_ed$ED_dc))

# join ED and IP data into single df
flu = right_join(flu_ed, flu_ip, by = c("hospital", "date"))
rm(flu_ed,flu_ip)

# convert date col to Date objects, with day set to 1
flu$date <- as.Date(paste0(flu$date, "01"), format = "%Y%m%d")

# filter for dates in study period & hospitals in study set
flu <- flu %>%
  filter(date >= as.Date("2013-07-01") & date <= as.Date("2023-06-30")) %>%
  filter(hospital %in% pt$Hospital_Number) %>%
  mutate_at(vars(ED_dc:death), ~ifelse(is.na(.), 0, .)) #replace NAs from join with zeroes

write.csv(flu, "cleaned/flu-per-hosp.csv", row.names = FALSE)

# sum across all hospitals
flu <- flu %>%
  group_by(date) %>%
  summarise_at(vars(ED_dc:death), sum, na.rm = TRUE)

write.csv(flu, "cleaned/flu.csv", row.names = FALSE)
```

Clean the flu test counts data frame, which contains monthly counts of patients tested for influenza virus.

```
# strip commas and convert test numbers into integers
flutest_ed$tests = as.integer(gsub(",", "", flutest_ed$tests))
flutest_ip$tests = as.integer(gsub(",", "", flutest_ip$tests))

# merge testing dfs
flutests <- full_join(flutest_ed, flutest_ip, by = c("hospital", "date")) %>%
  rename(ED_tests = tests.x, IP_tests = tests.y)
rm(flutest_ed, flutest_ip)

# convert date col to Date objects
flutests$date <- as.Date(paste0(flutests$date, "01"), format = "%Y%m%d")

# filter for dates in study period & hospitals in study set
flutests <- flutests %>%
  filter(date >= as.Date("2013-07-01") & date <= as.Date("2023-06-30")) %>%
  filter(hospital %in% pt$Hospital_Number) %>%
  mutate_at(vars(ED_tests:IP_tests), ~ifelse(is.na(.), 0, .)) %>% #replace NAs from join with zeroes
  group_by(date) %>%
  summarise_at(vars(ED_tests:IP_tests), sum, na.rm = TRUE)
```

```
write.csv(flutests, "cleaned/flu_tests.csv", row.names = FALSE)
rm(flutests)
```

Clean the RSV test counts data frame, which contains monthly counts of patients tested for RSV via either RSV-specific or multi-pathogen tests.

```
# strip commas and convert test numbers into integers
test_ed$tests = as.integer(gsub(",", "", test_ed$tests))
test_ip$tests = as.integer(gsub(",", "", test_ip$tests))

# merge testing dfs
tests <- full_join(test_ed, test_ip, by = c("hospital", "date")) %>%
  rename(ED_tests = tests.x, IP_tests = tests.y)
rm(test_ed, test_ip)

# convert date col to Date objects
tests$date <- as.Date(paste0(tests$date, "01"), format = "%Y%m%d")

# filter for dates in study period & hospitals in study set
tests <- tests %>%
  filter(date >= as.Date("2013-07-01") & date <= as.Date("2023-06-30")) %>%
  filter(hospital %in% pt$Hospital_Number) %>%
  mutate_at(vars(ED_tests:IP_tests), ~ifelse(is.na(.), 0, .)) #replace NAs from join with zeroes

# strip commas and convert test numbers into integers
test_ed_m$tests = as.integer(gsub(",", "", test_ed_m$tests))
test_ip_m$tests = as.integer(gsub(",", "", test_ip_m$tests))

# merge testing dfs
tests_m <- full_join(test_ed_m, test_ip_m, by = c("hospital", "date")) %>%
  rename(ED_tests = tests.x, IP_tests = tests.y)
rm(test_ed_m, test_ip_m)

# convert date col to Date objects
tests_m$date <- as.Date(paste0(tests_m$date, "01"), format = "%Y%m%d")

# filter for dates in study period & hospitals in study set
tests_m <- tests_m %>%
  filter(date >= as.Date("2013-07-01") & date <= as.Date("2023-06-30")) %>%
  filter(hospital %in% pt$Hospital_Number) %>%
  mutate_at(vars(ED_tests:IP_tests), ~ifelse(is.na(.), 0, .)) #replace NAs from join with zeroes

# merge testing dfs
tests <- full_join(tests, tests_m, by = c("hospital", "date"),
                   suffix = c("", "_multi"))
tests[is.na(tests)] <- 0
rm(tests_m)

write.csv(tests, "cleaned/tests-per-hosp.csv", row.names = FALSE)

# sum tests across all hospitals
tests <- tests %>%
  group_by(date) %>%
```

```r
  summarise_at(vars(ED_tests:IP_tests_multi), sum, na.rm = TRUE)

# write tests to df
write.csv(tests, "cleaned/tests.csv", row.names = FALSE)
```