Test for travel quarantine effect on SARS-CoV-2 imports

This script is to test whether the rate of new introductions from a country is effected by whether travellers from the country are under quarantine orders.

Date range for analysis is constrained to quarantine list data start/end dates. Phylogenetic import estimates are only for chains where travel context sequences clustered with them, and also only for one chain per polytomy. So these represent estimates for only a fraction of the total number of chains.

Q: why does it matters which transmission chain assumption we make if we take each polytomy as 1 import anyways? A: when polytomies are allowed to be Swiss, if the polytomy node is swiss then indeed, we cannot classify a location for the origin of the transmission chain This means should have more datapoints when s=F.

TODO: compare/contrast results when s=T and s=F

```r
workdir <- "../grapevine/jan-dec_-01_max_sampling_-5_context-sf"
s <- T

outdir <- paste(workdir, "output", sep = "/")
system(command = paste("mkdir -p", outdir))
db_connection <- open_database_connection(config_file = "../../../Repos/grapevine/workdir/input/config.
```

```
## [1] "config_file location: ../../../Repos/grapevine/workdir/input/config.yml"
## [1] "config file exists TRUE"
```

Load data: independent variables

```r
# indep variables: source country, whether travellers under quarantine order by day
travel_quarantine_by_country_day <- get_travel_quarantine_by_country_day(
  db_connection = db_connection
)
# indep variable: source country incidence per capita per day
country_incidence <- get_country_incidence(
  min_date = min(travel_quarantine_by_country_day$date),
  max_date = max(travel_quarantine_by_country_day$date),
  db_connection = db_connection
)
# indep variable: distance from country to Switzerland
country_distances <- get_country_distances(
  non_focal_countries = as.character(unique(travel_quarantine_by_country_day$iso_code)),
  focal_countries = "CHE",
  db_connection = db_connection
)
# indep variable: travel connection to Switzerland
country_arrivals <- read.delim(
  file = paste(
    workdir,
    "tmp/alignments/estimated_monthly_infectious_arrivals.txt",
    sep = "/"),
  stringsAsFactors = F) %>%
  select(date, iso_code, n_tourist_arrivals, n_commuter_permits)
```

```r
# clean data
warn_missing_incidence_data(
  travel_quarantine_by_country_day = travel_quarantine_by_country_day,
  country_incidence = country_incidence
)
```

```
## Warning in warn_missing_incidence_data(travel_quarantine_by_country_day = travel_quarantine_by_count:
##   iso_code           country_name
## 1      ABW                  Aruba
## 2      GUM                   Guam
## 3      PYF       French Polynesia
## 4      SXM           Sint Maarten
## 5      VGB British Virgin Islands
```

```r
travel_quarantine_by_country_day <- travel_quarantine_by_country_day %>%
  filter(iso_code %in% country_incidence$iso_code)

# merge data
indep_vars <- merge(
    x = travel_quarantine_by_country_day, y = country_incidence,
    by = c("iso_code", "date"),
    all = T)
indep_vars <- merge(
  x = indep_vars, y = country_distances,
  by = "iso_code",
  all.x = T)
indep_vars <- merge(
  x = indep_vars, y = country_arrivals,
  by = c("iso_code", "date"),
  all.x = T)
```

Load data: possible dependent variables

```r
# dep variable: number of FOPH recorded exposures from each country each day
recorded_exposures <- get_exposures_per_country_day(
  min_date = min(travel_quarantine_by_country_day$date),
  max_date = max(travel_quarantine_by_country_day$date),
  db_connection = db_connection
)
```

```
## [1] "Getting number of cases by exposure country and date of case confirmation from BAG meldeformula:

## Warning in get_exposures_per_country_day(min_date = min(travel_quarantine_by_country_day$date), : Re
## # A tibble: 2 x 4
## # Groups:   date [2]
##   english_name exp_land n_exposures date
##   <chr>        <chr>          <int> <date>
## 1 <NA>         ANDERES            1 2020-11-26
## 2 <NA>         Namibia            1 2021-02-07
```

```r
# dep variable: number of lineages departing a country each day
chains_asr <- load_chains_asr(s = s, workdir = workdir)
```

```
## Warning in scan(file = file, what = what, sep = sep, quote = quote, dec = dec, :
## EOF within quoted string

## Warning in scan(file = file, what = what, sep = sep, quote = quote, dec = dec, :
```

```
## EOF within quoted string

## Warning in read.table(file = file, header = header, sep = sep,
## quote = quote, : incomplete final line found by readTableHeader
## on '../grapevine/jan-dec_-01_max_sampling_-5_context-sf/tmp/asr/B.
## 1.1.74_m_3_p_1_s_T_tree_data_with_asr.txt'

## Warning in read.table(file = file, header = header, sep = sep,
## quote = quote, : incomplete final line found by readTableHeader
## on '../grapevine/jan-dec_-01_max_sampling_-5_context-sf/tmp/asr/B.
## 1.1.91_m_3_p_1_s_T_tree_data_with_asr.txt'
```

```r
chains_asr_representative <- pick_origin_representative_chains(chains_asr)
chains_asr_long <- pivot_chains_longer(chains_asr_representative)
asr_daily <- chains_asr_long %>%
  filter(foreign_tmrca >= min(travel_quarantine_by_country_day$date),
         foreign_tmrca <= max(travel_quarantine_by_country_day$date)) %>%
  group_by(foreign_tmrca, origin) %>%
  summarize(n_lineages = sum(asr_contribution, na.rm = T)) %>%
  ungroup() %>%
  mutate(iso_code = country_name_to_iso_code(country = origin)) %>%
  select(-origin) %>%
  rename("date" = "foreign_tmrca")

# merge data
dep_vars <- merge(
  x = recorded_exposures, y = asr_daily,
  by = c("iso_code", "date"),
  all = T)
```

Merge independent and dependent variables, aggregate timechunks

```r
model_data_daily <- merge(
  x = indep_vars, y = dep_vars,
  by = c("iso_code", "date"),
  all = T) %>%
  tidyr::replace_na(list(
    "quarantine_order" = F,  # not on the quarantine list = no quarantine order
    "n_exposures" = 0,  # no exposures in bag_meldeformular = 0 exposures
    "n_lineages" = 0,  # no foreign attachment points in the month have travel context descendents from
    "n_tourist_arrivals" = 0,
    "n_commuter_permits" = 0)) %>%  # no tourist arrivals to hotels recorded or commuter permits = 0 ar
  group_by(iso_code) %>%
  mutate(quarantine_y_and_n = length(unique(quarantine_order)) == 2) %>%
  filter(quarantine_y_and_n) %>%  # only countries on & off quarantine list at some point
  select(-quarantine_y_and_n) %>%
  ungroup()

model_data_weekly <- model_data_daily %>%
  mutate(date = format(date, "%y-%U")) %>%
  group_by(date, iso_code) %>%
  summarize(
    n_exposures = sum(n_exposures),
    n_lineages = sum(n_lineages),
    dist_to_CHE = dist_to_CHE[1],
    new_cases_per_million = mean(new_cases_per_million),
    quarantine_order = sum(quarantine_order) / n())  # indep var frac of days on list

model_data_monthly <- model_data_daily %>%
  mutate(date = format(date, "%y-%m-01")) %>%
  group_by(date, iso_code) %>%
  summarize(
    n_exposures = sum(n_exposures),
    n_lineages = sum(n_lineages),
    dist_to_CHE = dist_to_CHE[1],
    new_cases_per_million = mean(new_cases_per_million),
    quarantine_order = sum(quarantine_order) / n(),  # indep var frac of days on list
    n_tourist_arrivals = sum(n_tourist_arrivals, na.rm = T),  # a monthly variable only
    n_commuter_permits = sum(n_commuter_permits, na.rm = T))  # a monthly variable only
```

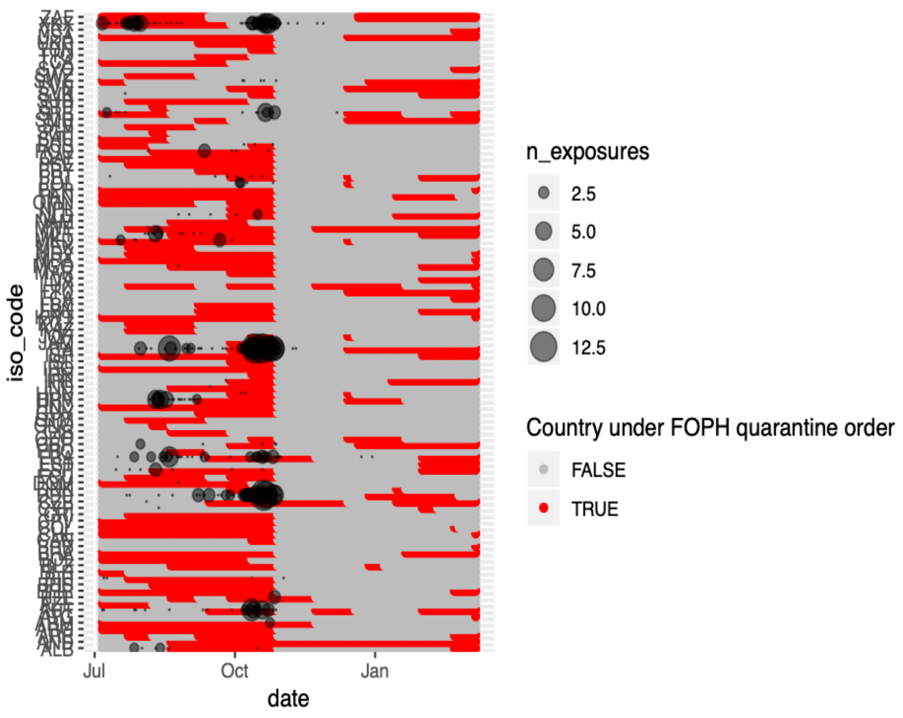
Plot data summary

```
plot_dep_var_by_quarantine_status(
  model_data = model_data_daily,
  outdir = outdir,
  dep_varname = "n_exposures")
```

```
## Saving 6.5 x 10 in image
```

```
## Warning: Removed 24292 rows containing missing values (geom_point).
```

```
## Warning: Removed 24292 rows containing missing values (geom_point).
```

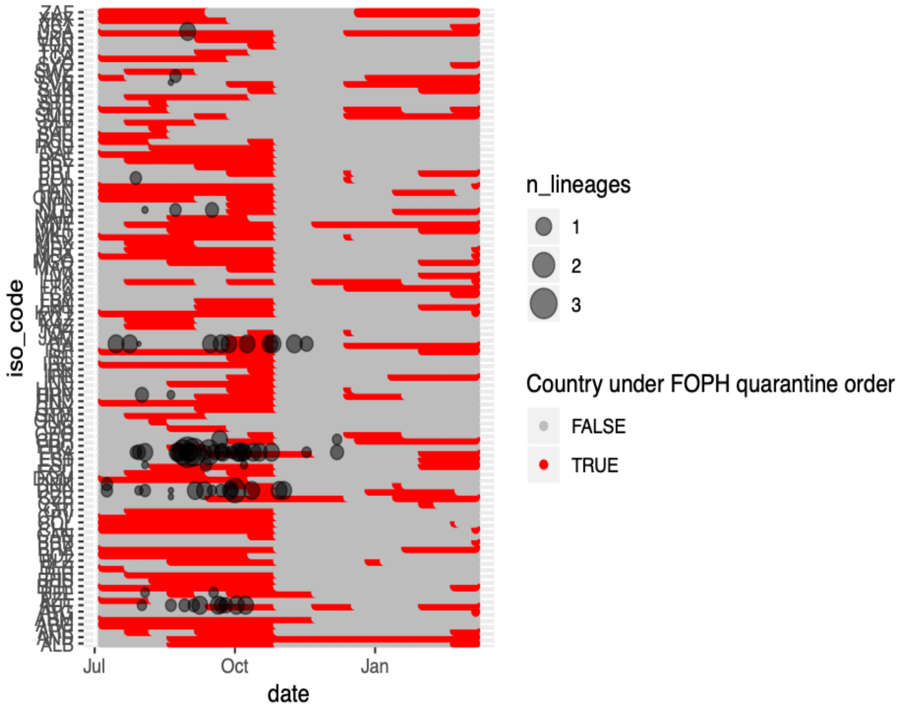


```r
plot_dep_var_by_quarantine_status(
  model_data = model_data_daily,
  outdir = outdir,
  dep_varname = "n_lineages")
```

```
## Saving 6.5 x 10 in image

## Warning: Removed 24520 rows containing missing values (geom_point).

## Warning: Removed 24520 rows containing missing values (geom_point).
```



Define model

```r
fit_model <- function(
  model_data, return_fitted_model = F, dep_varname, indep_varnames
) {
  my_formula <- paste(dep_varname, " ~ ", paste0(indep_varnames, collapse = " + "), sep = "")
  print(paste("Fitted model:", my_formula))
  fitted_model <- lm(
    data = model_data,
    formula = as.formula(my_formula))
  print(summary(fitted_model))
  if (return_fitted_model) {
    return(fitted_model)
  }
}
```

Fit model(s) Assumption: travel is constant year-round

```r
fit_model(
  model_data = model_data_monthly,
  dep_varname = "n_exposures",
  indep_varnames = c("new_cases_per_million", "quarantine_order", "dist_to_CHE",
                     "n_tourist_arrivals", "n_commuter_permits"))
```

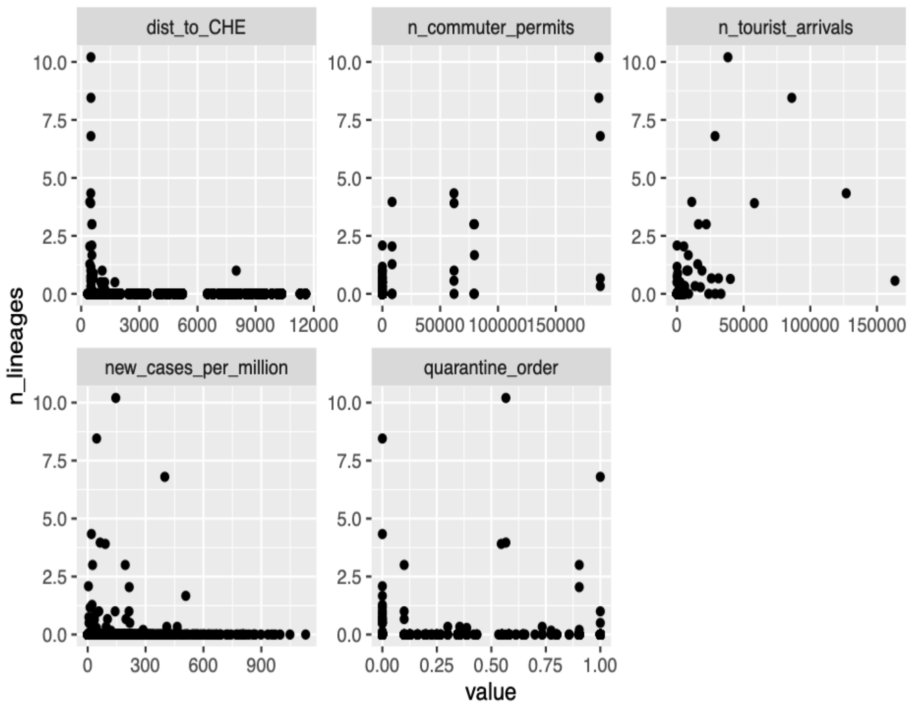
```
## [1] "Fitted model: n_exposures ~ new_cases_per_million + quarantine_order + dist_to_CHE + n_tourist_a
##
## Call:
## lm(formula = as.formula(my_formula), data = model_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -20.139  -0.590  -0.232   0.143  98.602
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)            7.992e-01  3.358e-01   2.380   0.0175 *
## new_cases_per_million -1.754e-03  9.124e-04  -1.923   0.0548 .
## quarantine_order       8.914e-01  3.800e-01   2.346   0.0192 *
## dist_to_CHE           -1.160e-04  4.802e-05  -2.416   0.0159 *
## n_tourist_arrivals     9.958e-05  2.213e-05   4.500 7.72e-06 ***
## n_commuter_permits     8.294e-05  1.170e-05   7.091 2.72e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.564 on 885 degrees of freedom
##   (9 observations deleted due to missingness)
## Multiple R-squared:  0.1672, Adjusted R-squared:  0.1625
## F-statistic: 35.54 on 5 and 885 DF,  p-value: < 2.2e-16
```

```r
fit_model(
  model_data = model_data_monthly,
  dep_varname = "n_lineages",
  indep_varnames = c("new_cases_per_million", "quarantine_order", "dist_to_CHE",
                     "n_tourist_arrivals", "n_commuter_permits"))
```

```
## [1] "Fitted model: n_lineages ~ new_cases_per_million + quarantine_order + dist_to_CHE + n_tourist_a
##
## Call:
## lm(formula = as.formula(my_formula), data = model_data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2066 -0.0356 -0.0111  0.0080  5.3123
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)            4.969e-02  2.828e-02   1.757   0.0793 .
## new_cases_per_million -1.640e-04  7.684e-05  -2.135   0.0331 *
## quarantine_order       4.507e-02  3.200e-02   1.408   0.1594
## dist_to_CHE           -6.040e-06  4.044e-06  -1.494   0.1356
## n_tourist_arrivals     1.354e-05  1.864e-06   7.267 8.07e-13 ***
## n_commuter_permits     2.310e-05  9.850e-07  23.449  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3844 on 885 degrees of freedom
##   (9 observations deleted due to missingness)
## Multiple R-squared:  0.5738, Adjusted R-squared:  0.5714
## F-statistic: 238.3 on 5 and 885 DF,  p-value: < 2.2e-16
```

```
model_data_monthly_gathered <- model_data_monthly %>%
  tidyr::gather(
    key = "variable", value = "value",
    dist_to_CHE, new_cases_per_million, quarantine_order, n_tourist_arrivals,
    n_commuter_permits)
ggplot(data = model_data_monthly_gathered,
       aes(x = value, y = n_lineages)) +
  geom_point() +
  facet_wrap(~variable, scales = "free")
```

```
## Warning: Removed 9 rows containing missing values (geom_point).
```


dist_to_CHE
n_commuter_permits
n_tourist_arrivals
new_cases_per_million
quarantine_order
n_lineages
value