IFR Flows

Enrico Spinielli

August 19, 2016

Contents

1 Abstract

This document describes the assumptions, shortcuts and steps taken to prepare the data used the circular plots of the flow of flights entering/exiting and within the airspace of Eurocontrol's Member States for the years 2001-2015 as displayed on PRU's website [1], [2].

2 Introduction

The studies for the flow of flights entering/exiting and within the airspace of Eurocontrol's Member States is inspired by Sander, Abel and Bauer's website (see Sander, Abel, and Bauer 2014) and its companion working paper (see Sander et al. 2014).

The underlying dataset is coming from the PRU's Analytical database and takes into account all IFR flights in the period 2001-2016 as recorded by Eurocontrol's Network Manager (formely CFMU).

In the following chapters we will describe how we extracted the flows, what we filtered, the amount and impact of the filters, the compromises we have taken when mapping flights to countries and countries to regions and more.

This is a work in progress so throughout this document we will mark what needs to be tackled soon[1] with `TODO` and areas of further investigation[2] with `RSRC`.

3 The (nasty) details

3.1 Yearly city pair flows

The following query extracts city pair flows from the archived flight details in PRU's analytical database[3].

[1] The definition of *soon* is intentionally omitted to let us free to decide ;-)

[2] These are subjects that could potentially (typically because of our lack of knowledge) require a substantial amount of time/resources.

[3] We have access to flight details (and more) since 27^{th} Apr 1995 when the flight plan reception and processing for France, Germany and the Benelux States was handled by the then new Central Flowm Management Unit.

```
set worksheetname Flows;
SET SQLFORMAT csv;

SELECT
  TO_CHAR(F.LOBT, 'YYYY') AS YEAR
  , COUNT(FLT_UID) NB_OF_FLIGHT
  , A1.EC_ISO_CT_CODE DEP_ISO_COUNTRY_CODE
  , A2.EC_ISO_CT_CODE DES_ISO_COUNTRY_CODE
FROM
    SWH_FCT.FAC_FLIGHT F
  , SWH_FCT.DIM_AIRPORT A1
  , SWH_FCT.DIM_AIRPORT A2
  , SWH_FCT.DIM_AIRCRAFT_TYPE AC
WHERE
    A1.SK_AP_ID = F.SK_ADEP_ID
  AND
    A2.SK_AP_ID   = F.SK_ADES_ID
  AND
    F.LOBT       >= '01-JAN-2001'
  AND
    F.LOBT        < '01-JAN-2016'
  AND
    (A1.EC_ISO_CT_CODE != '##' AND A2.EC_ISO_CT_CODE != '##')
  AND
    F.AIRCRAFT_TYPE_ICAO_ID = AC.ICAO_TYPE_CODE
  AND
    (F.ICAO_FLT_TYPE != 'M')
  AND
    SUBSTR(AC.ICAO_DESC,1,1) != 'H'
GROUP BY
    A1.EC_ISO_CT_CODE
  , A1.ISO_CT_CODE
  , A2.EC_ISO_CT_CODE
  , TO_CHAR(F.LOBT,'YYYY');
```

The flollowing filters have been applied in the query above:

- flights without an ISO2 country code for either the departing or arriving airport have been excluded:

  ```
  ...
  AND (A1.EC_ISO_CT_CODE != '##' AND A2.EC_ISO_CT_CODE != '##')
  ...
  ```

- military and helicopter flights have been excluded:

  ```
  ...
  AND (F.ICAO_FLT_TYPE != 'M')
  AND SUBSTR(AC.ICAO_DESC,1,1) != 'H'
  ...
  ```

The following table summarises the amount of filtered out flights compared to the total per year.

Year	No ISO code	Military	Helicopter

- **TODO** fill the table above with the result of the queries we run in the past in order to justify the next paragraph.

Given the filtered out amounts are tiny compared with the relevant totals, it is justified to discard them because they are not going to distort the shape of the flows once we aggregate the flights per country and region.

3.2 Preprocessing

The raw data is preprocessed via the script `preprocessRawData.R`

We define the range of years we will process:

```
# year range (wef = with effect from; til = unTIL) [inclusive]
# note: changing wef (earlier) could get you into ISO code mapping nightmare
wef <- 2001
til <- 2015
```

and filter the dataset accordingly:

```
# keep only flows in year [wef, til] interval
flows %<>%
  filter(YEAR >= wef & YEAR <= til)
```

Then columns names are redefined:

```
flows %<>%
  rename(
    year = YEAR,
    flights = NB_OF_FLIGHT,
    origin_iso = DEP_ISO_COUNTRY_CODE,
    destination_iso = DES_ISO_COUNTRY_CODE)
```

Let's make sure the dataset is sound:

- `year` is within the 2001 - 2015 range
- origin and destinations are not undefined, (but "NA" is a valid value for Namibia)

```
library('assertr')
# basic checks: no NA's and right range for years
flows %<>%
  assert(not_na, destination_iso) %>%
  assert(not_na, origin_iso) %>%
  assert(in_set(seq(wef, til)), year)
```

Also due to our mapping of old ISO2 codes from our DB to more *modern* ones from the relevant year. This is the case for Serbia (`CS` to `RS`) and Martinique/Guadaloupe (`XF` to `MQ`)

```
flows %<>%
  # **Note**: the ISO code for Serbia is provided as CS,
  #           which was correct till 2005.
  #           Since 2006 it is RS.
  # Let's fix it:
  mutate(
    origin_iso      = ifelse(origin_iso      == "CS" & year >= 2006,
                             "RS", origin_iso),
    destination_iso = ifelse(destination_iso == "CS" & year >= 2006,
                             "RS", destination_iso)) %>%
  # Also `XF` has been used for Martinique/Guadaloupe.
  # Let's put `MQ` which will result in a proper UN region.
  mutate(
    origin_iso      = ifelse(origin_iso      == "XF", "MQ", origin_iso),
    destination_iso = ifelse(destination_iso == "XF", "MQ", destination_iso))
```

- `TODO` check ISO codes for Montenegro (ME since 2006), Serbia (RS since 2006), Congo (CD, since 1997), YU (Yugoslavia, till 2003-07), ZR (Zaire, till 1995-07)
- `TODO`: check what happens for Dutch Antilles, France's territories outside Europe (St Pierre's aiport, LFVP, in the oversea community of Saint-Pierre and Miquelon; or the ones in the French Guiana, ...)

- `RSRC` check how ISO 3166-2 codes evolved in time and how that is handled in the `EC_ISO_CT_CODE` column of the `SWH_FCT.DIM_AIRPORT` table. The relevant page on Wikipedia provides some info on the time when codes changed.
- `RSRC` if we are going to map the flows, it could be interesting to try to find shapefiles for how countries did split, reshaped, ...

Finally we save to output filename for further use.

```
data_dir = "data"
# input file
raw_flows_file = "raw/ifr_flows_2001-2015.csv"
# output file
city_pairs_file = "ifr_city_pairs.csv"

write_csv(flows, paste(data_dir, city_pairs_file, sep="/"), na="")
rm(flows)
```

3.3 Data transformation for circular plot

The circular plot clusters countries into regions. Luckily the `countrycode` R package converts country codes into one of eleven coding schemes including regional grouping. A recently added feature allows to provide a dictionary for custom mappings: we can hence use one for the our aviation data[4].

3.3.1 Preliminary fixes

We start by reading the previously processed data,

```
data_dir = "data"
city_pairs_file = "ifr_city_pairs.csv"

mydf <- read_csv(str_c(data_dir, city_pairs_file, sep = '/'), na = c(""))
flows <- tbl_df(mydf)
rm(mydf)
```

We then (arbitrarily) assign a UN region to countries that do not have one, i.e. Taiwan (ISO 3166-2: `TW`) and the British Indian Ocean Territory (ISO 3166-2: `IO`).

```
flows %<>%
  mutate(originregion_name = ifelse(origin_iso == "TW",
                                    "Eastern Asia",
                                    originregion_name)) %>%
  mutate(destinationregion_name = ifelse(destination_iso == "TW",
                                         "Eastern Asia",
                                         destinationregion_name)) %>%

  mutate(originregion_name = ifelse(origin_iso == "IO",
                                    "Eastern Africa",
                                    originregion_name)) %>%
  mutate(destinationregion_name = ifelse(destination_iso == "IO",
                                         "Eastern Africa",
                                         destinationregion_name))
```

3.3.2 Assign region to country

Assign regions as per Eurocontrol's PRU classification

[4] the README from the package repository describes this feature using a dataset we provided as a contribution to the package

PRU's region	Countries

- TODO: generate a table with the mapping, possibly using kable, see here

```
classification <- "eurocontrol_pru"
flows %<>%
  mutate(originregion_name = countrycode(origin_iso, "iso2c", classification,
                                         dictionary=custom_dict),
         destinationregion_name = countrycode(destination_iso, "iso2c", classification,
                                              dictionary=custom_dict),
         origin_name = countrycode(origin_iso, "iso2c", "country.name",
                                   dictionary=custom_dict),
         destination_name = countrycode(destination_iso, "iso2c", "country.name",
                                        dictionary=custom_dict))
```

3.3.3 Split dataset: intra and extra Eurocontrol

We have 3 classes of flows:

1. Flows within Eurocontrol area, i.e. flights where both departure and destination airports are in Eurocontrol Member States's territory.
2. Overfling flows, i.e. flights originating and landing in non-Member States
3. Flows in/out of Eurocontrol area, i.e. flights where either departure or destination airport is in a Member State.

Using the following tables (IN = inside Eurocontrol, OUT = outside Eurocontrol)

Departure AD	Destination AD	Flow type	Description
IN	IN	internal	Internal flows
IN	OUT	outbound	Outflying flows
OUT	IN	inbound	Inflying flows
OUT	OUT	overflying	Overflying flows

we can classify the flows

```
flowTypes <- c("internal", "outbound", "inbound", "overflying")
flowClass <- function(depRegion, desRegion) {
  isInternal = (depRegion == "Eurocontrol") & (desRegion == "Eurocontrol")
  if (isInternal) return("internal")

  isOutbound = (depRegion == "Eurocontrol") & !(desRegion == "Eurocontrol")
  if (isOutbound) return("outbound")

  isInbound = !(depRegion == "Eurocontrol") & (desRegion == "Eurocontrol")
  if (isInbound) return("inbound")

  return("overflying")
}

flows %<>%
  mutate(ftype =
           ifelse((originregion_name == "Eurocontrol") &
                    (destinationregion_name == "Eurocontrol"), "internal",
                  ifelse(!(originregion_name == "Eurocontrol") &
                           (destinationregion_name == "Eurocontrol"), "inbound",
                         ifelse((originregion_name == "Eurocontrol") &
```

```
                                    !(destinationregion_name == "Eurocontrol"), "outbound",
                                  "overflying"))))

#flows %<>%
#  mutate(ftype = flowClass(originregion_name, destinationregion_name))
```

We now split the dataset in two: `intra-flows` contains all `internal` flows; `extra-flows` all the rest (i.e. we will not separate the overflying flow from In/Outflying ones.)

```
intra_flows <- flows %>%
  filter(ftype == "internal")
```

Furthermore we map `intra-flows` regions according to ESRA08 classification from STATFOR (see pag 57 of STATFOR 2016):

```
# split intra-Eurocontrol flights in regions according to STATFOR
classification <- "eurocontrol_statfor"
intra_flows %<>%
  mutate(originregion_name = countrycode(origin_iso, "iso2c", classification,
                                         dictionary=custom_dict),
         destinationregion_name = countrycode(destination_iso, "iso2c", classification,
                                              dictionary=custom_dict),
         origin_name = countrycode(origin_iso, "iso2c", "country.name",
                                   dictionary=custom_dict),
         destination_name = countrycode(destination_iso, "iso2c", "country.name",
                                        dictionary=custom_dict))
```

Save the internal and external flows on disk:

```
intra_flows_file = "intra-flows.csv"
intra_flows %>%
  write.csv(., file = paste(data_dir, intra_flows_file, sep="/"), row.names = FALSE)
```

3.3.4 Refine Extra Flows

In order to reduce the number of regions in the final circular plot, we reassign the regions to some extra-European flows. So `Russia` and `China` go to `Asia` rather than having their own region. We also merge the whole Northern, Central and Souther America in one `America` region.

```
# Russia ->  Asia
extra_flows %<>%
  mutate(originregion_name = ifelse(originregion_name == "Russia",
                                    "Asia",
                                    originregion_name))
extra_flows %<>%
  mutate(destinationregion_name = ifelse(destinationregion_name == "Russia",
                                         "Asia",
                                         destinationregion_name))


# China ->  Asia
extra_flows %<>%
  mutate(originregion_name = ifelse(originregion_name == "China",
                                    "Asia",
                                    originregion_name))
extra_flows %<>%
  mutate(destinationregion_name = ifelse(destinationregion_name == "China",
                                         "Asia",
                                         destinationregion_name))
```

```
# Southern America ->  America
extra_flows %<>%
  mutate(originregion_name = ifelse(originregion_name == "Southern America",
                                    "America",
                                    originregion_name))

extra_flows %<>%
  mutate(destinationregion_name = ifelse(destinationregion_name == "Southern America",
                                         "America",
                                         destinationregion_name))

# Northern America -> N Ame
extra_flows %<>%
  mutate(originregion_name = ifelse(originregion_name == "Northern America",
                                    "America",
                                    originregion_name))
extra_flows %<>%
  mutate(destinationregion_name = ifelse(destinationregion_name == "Northern America",
                                         "America",
                                         destinationregion_name))
```

Save the external flows on disk:

```
extra_flows_file = "extra-flows.csv"
extra_flows %>%
  write.csv(., file = paste(data_dir, extra_flows_file, sep="/"), row.names = FALSE)
```

3.4 Analysis

- `RSRC`: analyse the flows time series. Summaries:

List of countries contributing the major movements:

3.5 Data Preparation for Visualisation

The circular plot library[5] we use for visualizing the flows of flights expects as input preprocessed data ready to plot.

In this section we explain which are the transformations and support files needed. These support files will be *compiled* by the code in another repository, `circflows`.

We will pick the extra flow dataset but the same set of files is produced for the intra flow one.

3.5.1 Countries to show

In order to avoid too small flows in the final visualization we filter out all country flows with less that 10 flights per day, i.e. $10 * 365$.

```
# threshold for filtering out small flows
fpd <- 5 # flights per day
threshold <- 365 * fpd
```

Then we will then generate a file that specifies which countries to show, something like:

```
iso,show (1 = yes; 2 = no)
US,1
IN,1
```

[5] This library is an extension of the chord diagram as made possible by Mike Bostock's D3.js.

```
CN,2
...
```

Now let's filter out the little flows...

```
# keep only the years as previously defined, used to generate the list of visible countries
# CHECK: shouldn't we filter on the sum of both directions?
extra_flows %<>%
  filter(flights >= threshold) %>%
  filter(year %in% extra_years) %>%
  spread(year, flights, fill=0)

# prepend "countryflow_"" to year columns, i.e. from "2007" to "countryflow_2007"
names(extra_flows) <- ifelse(names(extra_flows) %in% unlist(extra_years, use.names = FALSE),
                             paste("countryflow_", names(extra_flows), sep=""),
                             names(extra_flows))

extra_flows %>%
  write.csv(., file = paste(data_dir, "extra-flows-viz.csv", sep = "/"), row.names = FALSE)
```

3.5.2 Years in the dataset

We need a file listing the years in the dataset

```
extra_years <- extra_flows %>%
  select(year) %>%
  distinct() %>%
  arrange(year)

# names(years) <- c("year")

extra_years
# eventually keep only a subset of the time serie
extra_years <- extra_years[[1]] # get the array of _ALL_ year numbers
# extra_years <- c(extra_years[length(extra_years)]) # just take last one

extra_years %>%
  write.table(., sep=",", file = paste(data_dir, "extra-years-viz.csv", sep="/"),
              row.names = FALSE, col.names = c("year"), quote = TRUE)
```

- TODO: generate years file for intra-flow too.

3.5.3 Countries and regions in the datasets

We then need to create the files for the countries and the regions present in the (now filtered) dataset

```
extra_descountries <- extra_flows %>%
  select(destination_iso, destination_name) %>%
  rename(iso = destination_iso, name = destination_name) %>%
  distinct()

extra_depcountries <- extra_flows %>%
  select(origin_iso, origin_name) %>%
  rename(iso = origin_iso, name = origin_name) %>%
  distinct()

extra_countries <- union(extra_descountries, extra_depcountries) %>%
  distinct() %>%
  arrange(iso) %>%
```

```
  mutate(show = 1)

# set show=1 to cous0 rows matching cous iso
# CHECK: used?
#extra_countries %<>%
#  left_join(extra_count, by="iso") %>%
#  mutate(show=show.y,name=name.x) %>%
#  select(iso,name,show) %>%
#  mutate_each(funs(replace(., which(is.na(.)), 0)))

# quotes needed due to commas in some country names
extra_countries %>%
  write.csv(., file = paste(data_dir, "extra-countries-viz.csv", sep = "/"),
            row.names = FALSE, quote = TRUE)

# Regions in the (unfiltered) set
extra_desregions <- extra_flows %>%
  select(destinationregion_name) %>%
  rename(name = destinationregion_name) %>%
  distinct()

extra_depregions <- extra_flows %>%
  select(originregion_name) %>%
  rename(name = originregion_name) %>%
  distinct()

extra_regions <- union(extra_desregions, extra_depregions) %>%
  distinct() %>%
  arrange(name)

extra_regions %>%
  write.csv(., file = paste(data_dir, "extra-regions-viz.csv", sep="/"),
            row.names = FALSE, quote = TRUE)
```

References

Sander, Nikola, Guy J. Abel, and Ramon Bauer. 2014. "The Global Flow of People." http://www.global-migration.info/.

Sander, Nikola, Guy J. Abel, Ramon Bauer, and Johannes Schmidt. 2014. *Visualising Migration Flow Data with Circular Plots*. Wohllebengasse 12-14, A-1040 Vienna, Austria: Vienna Institute of Demography; Austrian Academy of Sciences. http://www.oeaw.ac.at/vid/download/WP2014_02.pdf.

STATFOR. 2016. "EUROCONTROL Seven-Year Forecast – February 2016." 15/12/17-52. EUROCONTROL/STATFOR. http://www.eurocontrol.int/sites/default/files/content/documents/official-documents/forecasts/seven-year-flights-service-units-forecast-2016-2022-Feb2016.pdf.