Comparison of Comtrade and HMRC databases for selected commodities

Required libraries

```
library(RPostgreSQL)
library(tidyverse)
library(dbplyr)
library(rjson)
library(DBI)
library(lubridate)
library(tibble)
library(olsrr)
library(ggplot2)
library(ggExtra)
library(gridExtra)
```

Get the auxiliary data

```
source("get_HMRC_aux_data.R")
list1 <- get_HMRC_aux_data()
comcode <- data.frame(Reduce(rbind, list1[1]))
port    <- data.frame(Reduce(rbind, list1[2]))
country <- data.frame(Reduce(rbind, list1[3]))
eu_list <- c("BE","BG","CZ","DK","DE","EE","IE","EL","ES","FR","HR","IT",
             "CY","LV","LT","LU","HU","MT","NL","AT","PL","PT","RO","SI","SK","FI","SE","UK")
```

Use this line to search for commodity codes using a keyword

```
(comname      <- comcode[grep('CHICKEN',   toupper(comcode$description)),c(1,3)])
```

```
##     commoditycode
## 264      02071110
## 265      02071130
## 266      02071190
## 268      02071210
## 269      02071290
##
## 264                                                                                     Fresh or chil
## 265                                                      Fresh or chilled, plucked and drawn fowls of s
## 266 Fresh or chilled, plucked and drawn fowls of species Gallus domesticus, without heads, feet, necks, he
## 268                                                             Frozen fowls of species Gallus domesti
## 269                        Frozen fowls of species Gallus domesticus, plucked and drawn, without heads, 
```

Partners codes

Poland: 616

Spain : 724

Brazil: 76

UK : 826

Three commodities codes

Chicken: 02071

Cucumber: 070700

Beef: 160250

Set the partner country and the commodity: Leave the rest to the code: no worries about *arrivals* or *imports*

```r
#Define commodity and partner country
com_id  <- "160250"
part_id <- 76
#What am I searching for commodity-wise
(comcode[str_detect(comcode$commoditycode,paste('^',com_id,sep='')),3])
```

```
## [1] "Prepared or preserved meat or offal of bovine animals (excl. sausages and similar products, finely ho
## [2] "Prepared or preserved meat or offal of bovine animals, uncooked, incl. mixtures of cooked meat or offa
## [3] "Corned beef, in airtight containers"
## [4] "Meat or offal of bovine animals, prepared or preserved, cooked (excl. corned beef in airtight contain
```

GET COMTRADE DATA

```r
source("get_Comtrade_data.R")
#Comtrade SQL request
stime <- Sys.time()
df1   <- get_Comtrade_data(201401,201601,"default",com_id,as.character(part_id))
etime <- Sys.time()
print(etime-stime)
```

```
## Time difference of 1.568022 mins
```

Tidy Comtrade data

```r
#Group by commodity code for the same good if necessary (different cuts for chicken...)
print(unique(df1$commodity_code))
```

```
## [1] "160250"
```

```
df2  <- df1 %>% group_by(period,trade_flow,reporter,reporter_code,partner,partner_code) %>%
                summarize(net_weight_kg   = sum(netweight_kg),
                trade_value_usd = sum(trade_value_usd)) %>% ungroup()
#Compute the price in usd per kg
df2  <- df2  %>% mutate(price_usd_kg = trade_value_usd/net_weight_kg)
#Turn period into a proper date
df2  <- df2 %>% mutate(period_date = ymd(paste(period,"01",sep="")))
#Remove missing observations
df2 <- df2[complete.cases(df2),]
#Get the comtrade data for imports into the uk for the given commodity
comtrade_imports_into_uk <- df2 %>%
  filter(reporter=="United Kingdom") %>%
  filter(trade_flow=="Imports")
```

Get partner country alpha from the code

```
cname <- country[country$countryname==unique(df1$partner),2]
```

GET HMRC DATA

```
source("get_HMRC_data.R")
source("get_HMRC_data_imports.R")
stime <- Sys.time()
if(cname %in% eu_list){
  print('It belongs to EU')
HMRC_import_food_data <- get_HMRC_data(arrivals)
}else{
  print('It does not belong to EU')
HMRC_import_food_data <- get_HMRC_data_imports(imports)
}
```

```
## [1] "It does not belong to EU"
```

```
etime <- Sys.time()
print(etime-stime)
```

```
## Time difference of 10.08315 mins
```

```
#(col_names               <- t(as.data.frame(colnames(HMRC_import_food_data)))
```

Tidy the data depending on EU/non-EU (arrivals/imports)

```
if(cname %in% eu_list){
#Filter the data for the selected commodity_code
tmp  <- HMRC_import_food_data
#tmp1 <- tmp %>% filter(str_sub(smk_comcode,1,str_length(com_id)) == com_id)
tmp2 <- tmp[str_detect(tmp$smk_comcode,paste('^',com_id,sep='')),]
#Remove crazy year
current_year <- 2018
tmp2 <- tmp2 %>% filter(as.numeric(smk_period_reference)<100*(current_year+1))
```

```r
#Ignore some variables
tmp2 <- tmp2 %>%
select(-smk_coo_seq,-smk_coo_alpha) %>%
select(-smk_nature_of_transaction,-smk_mode_of_transport,-smk_no_of_consignments) %>%
select(-smk_suite_indicator,-smk_sitc,-smk_ip_comcode) %>% select(-smk_supp_unit,-smk_trade_ind,-smk_re
#Rename variables
tmp2 <- tmp2 %>% rename(commodity_code  = "smk_comcode")
tmp2 <- tmp2 %>% rename(partner_code    = "smk_cod_seq")
tmp2 <- tmp2 %>% rename(partner_id      = "smk_cod_alpha")
tmp2 <- tmp2 %>% rename(period          = "smk_period_reference")
tmp2 <- tmp2 %>% rename(trade_value_spd = "smk_stat_value")
tmp2 <- tmp2 %>% rename(netweight_kg    = "smk_nett_mass")
#Sterling pounds to US dollars
tmp2 <- tmp2 %>% mutate(trade_value_usd = trade_value_spd * 1.41) %>% select(-trade_value_spd)
}else{
#Filter the data for the selected commodity_code
tmp <- HMRC_import_food_data
tmp <- tmp %>% select(comcode,cod_sequence,cod_alpha,account_date,value,quantity_1)
tmp <- tmp %>% rename(commodity_code  = "comcode")
tmp <- tmp %>% rename(partner_code    = "cod_sequence")
tmp <- tmp %>% rename(partner_id      = "cod_alpha")
tmp <- tmp %>% rename(period_tmp      = "account_date")
tmp <- tmp %>% rename(trade_value_spd = "value")
tmp <- tmp %>% rename(netweight_kg    = "quantity_1")
#Filter the data for the selected commodity_code
tmp2 <- tmp[str_detect(tmp$commodity_code,paste('^',com_id,sep='')),]
tmp2 <- tmp2 %>% mutate(period = paste(str_sub(period_tmp,4,7),str_sub(period_tmp,1,2),sep='')) %>% sel
tmp2$partner_id <- gsub('GB', 'UK', tmp2$partner_id)
#Remove crazy year
current_year <- 2018
tmp2 <- tmp2 %>% filter(as.numeric(period)<100*(current_year+1))
#Sterling pounds to US dollars
tmp2 <- tmp2 %>% mutate(trade_value_usd = trade_value_spd * 1.41) %>% select(-trade_value_spd)
}
```

Keep going... No matter the HMRC table, the data is in *tmp2*

```r
#Group by commodity code for the same good if necessary (different cuts for chicken...)
print(unique(tmp2$commodity_code))
```

```
## [1] "16025031" "16025095" "16025010"
```

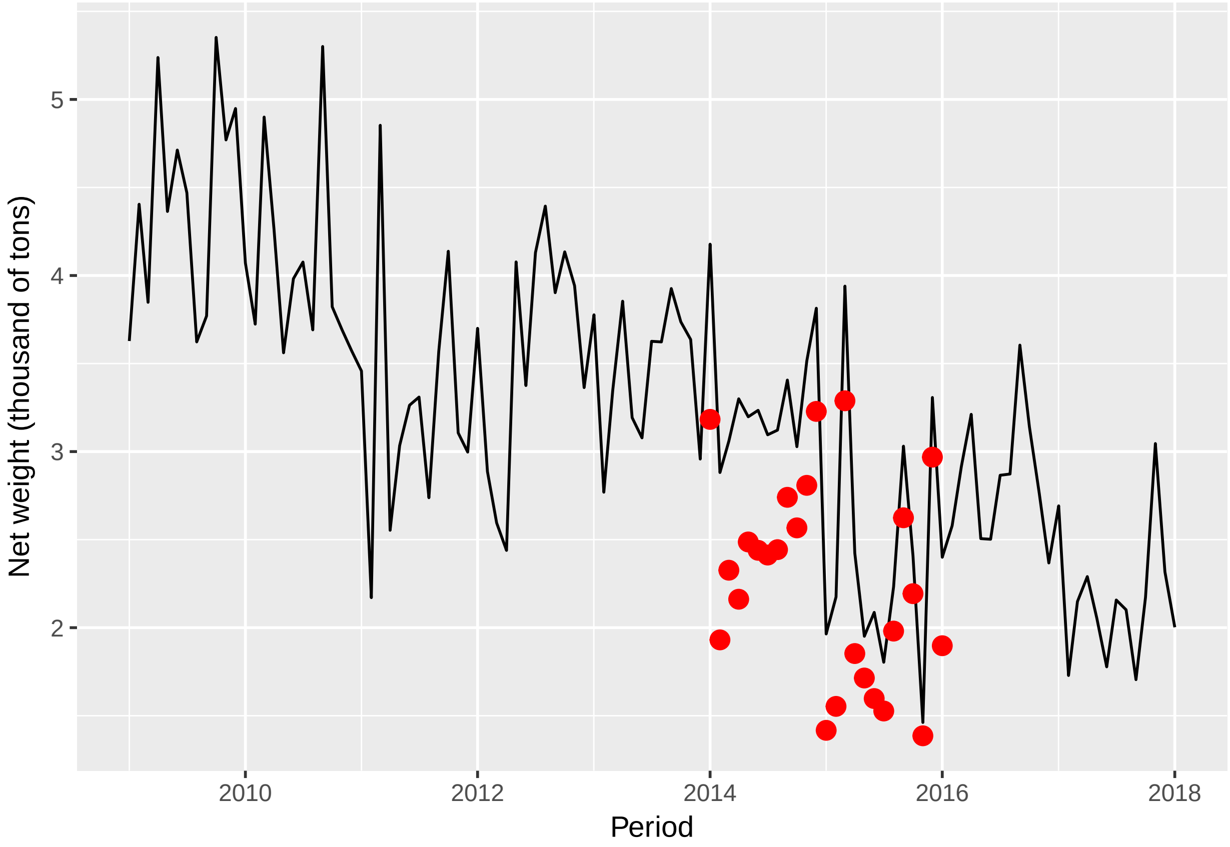
```r
tmp3 <- tmp2 %>% group_by(period,partner_id,partner_code) %>%
                 summarize(net_weight_kg   = sum(netweight_kg),
                 trade_value_usd = sum(trade_value_usd)) %>% ungroup()
#Compute the price in usd per kg
tmp3 <- tmp3 %>% mutate(price_usd_kg = trade_value_usd/net_weight_kg)
#Turn period into a proper date
tmp3 <- tmp3 %>% mutate(period_date = ymd(paste(as.character(as.numeric(period)),"01",sep="")))
#Remove missing observations
tmp4 <- tmp3[complete.cases(tmp3),]
tmp5 <- tmp4 %>% filter(trade_value_usd > 0 & net_weight_kg > 0)
#Get the comtrade data for imports into the uk for the given commodity
```

```
HMRC_imports_into_uk <- tmp5 %>% filter(partner_id ==cname)
```

Do the plots comparing both databases

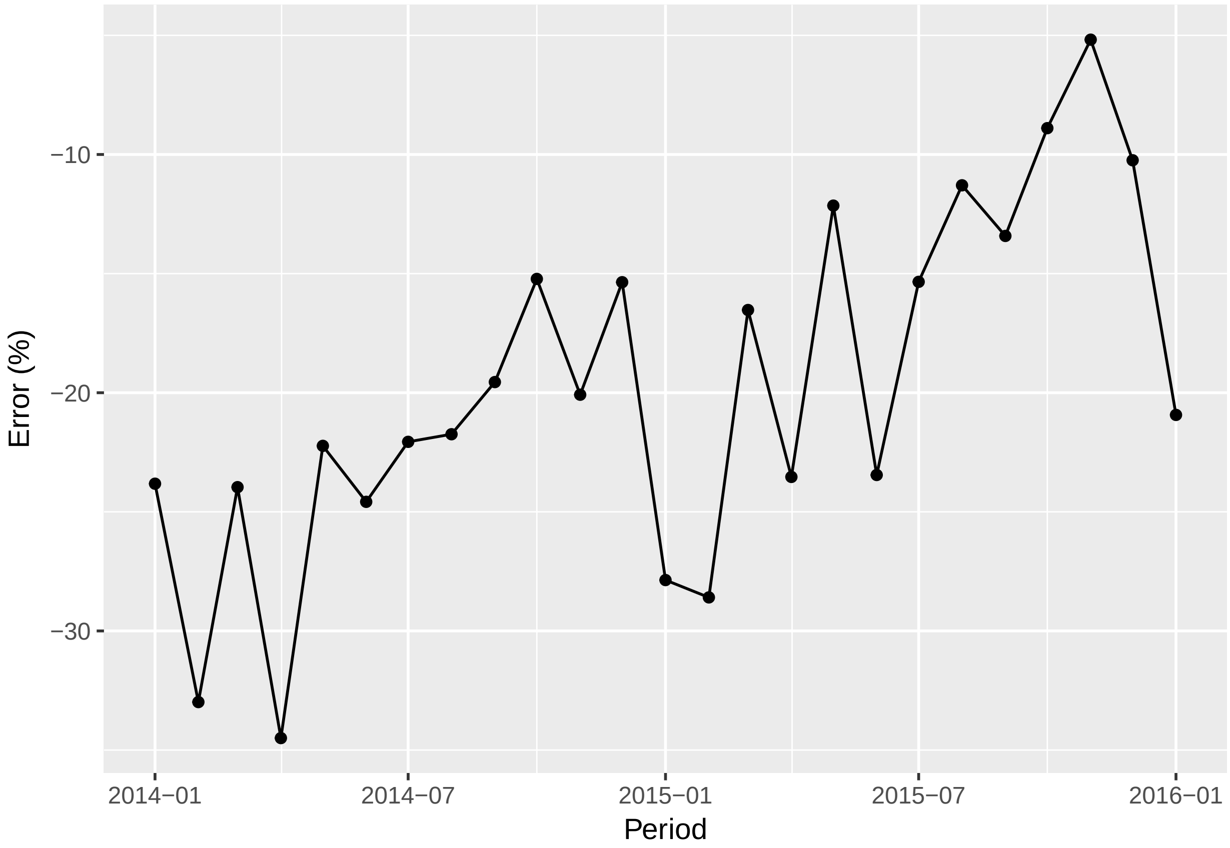
Net weight in kg

```
ggplot(NULL) + geom_line(data=HMRC_imports_into_uk,
                         mapping = aes(x=period_date,y=net_weight_kg/1e6)) +
               geom_point(data=comtrade_imports_into_uk,
                         mapping = aes(x=period_date,y=net_weight_kg/1e6),color="red",size=3) +
  labs(x="Period",y="Net weight (thousand of tons)")
```



Relative error

```
comb <- inner_join(comtrade_imports_into_uk,HMRC_imports_into_uk,by="period_date")
weight <- comb %>% select(starts_with("net_weight_kg"),period_date) %>%
  mutate(error = (100*(net_weight_kg.x-net_weight_kg.y)/net_weight_kg.y))
ggplot(data=weight,aes(x=period_date)) + geom_line(aes(y=error)) +
                                          geom_point(aes(y=error)) +
  labs(x="Period",y="Error (%)")
```


Error (%)
−10
−20
−30
2014−01
2014−07
2015−01
2015−07
2016−01
Period