FSA_S2DS

Load the required libraries

```
library(rjson)
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyr)
library(knitr)
library(maptools)
```

```
## Loading required package: sp
```

```
## Checking rgeos availability: TRUE
```

```
library(stringr)
library(mapproj)
```

```
## Loading required package: maps
```

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------- ti
```

```
## ✓ tibble  1.4.1     ✓ purrr   0.2.4
## ✓ readr   1.1.1     ✓ forcats 0.2.0
```

```
## -- Conflicts ------------------------------------------------------------------------------- tidyvers
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::map()    masks maps::map()
```

```
library(rgdal)
```

```
## rgdal: version: 1.2-16, (SVN revision 701)
##  Geospatial Data Abstraction Library extensions to R successfully loaded
##  Loaded GDAL runtime: GDAL 1.10.1, released 2013/08/26
##  Path to GDAL shared files: /usr/share/gdal/1.10
##  GDAL binary built with GEOS: TRUE
##  Loaded PROJ.4 runtime: Rel. 4.8.0, 6 March 2012, [PJ_VERSION: 480]
##  Path to PROJ.4 shared files: (autodetected)
##  Linking to sp version: 1.2-5
```

```
library(ggmap)
library(maps)
library(mapdata)
```

```r
library(mapproj)
library(rworldmap)
```

```
## ### Welcome to rworldmap ###
## For a short introduction type :   vignette('rworldmap')
```

```r
library(measurements)
```

This function fetches data using the Comtrade API: nothing original, just copied the API usage for R in the example

```r
get.Comtrade <- function(url="http://comtrade.un.org/api/get?"
                         ,maxrec
                         ,type="C"
                         ,freq
                         ,px="HS"
                         ,ps
                         ,r
                         ,p
                         ,rg="all"
                         ,cc="ALL"
                         ,fmt="json"
                         ,token
)
{
  string<- paste(url
                 ,"max=",maxrec,"&" #maximum no. of records returned
                 ,"type=",type,"&" #type of trade (c=commodities)
                 ,"freq=",freq,"&" #frequency
                 ,"px=",px,"&" #classification
                 ,"ps=",ps,"&" #time period
                 ,"r=",r,"&" #reporting area
                 ,"p=",p,"&" #partner country
                 ,"rg=",rg,"&" #trade flow
                 ,"cc=",cc,"&" #classification code
                 ,"token=",token,"&" #token
                 ,"fmt=",fmt        #Format
                 ,sep = ""
  )

  if(fmt == "csv") {
    raw.data<- read.csv(string,header=TRUE)
    return(list(validation=NULL, data=raw.data))
  } else {
    if(fmt == "json" ) {
      raw.data<- fromJSON(file=string)
      data<- raw.data$dataset
      validation<- unlist(raw.data$validation, recursive=TRUE)
      ndata<- NULL
      if(length(data)> 0) {
        var.names<- names(data[[1]])
```

```
        data<- as.data.frame(t( sapply(data,rbind)))
        ndata<- NULL
        for(i in 1:ncol(data)){
          data[sapply(data[,i],is.null),i]<- NA
          ndata<- cbind(ndata, unlist(data[,i]))
        }
        ndata<- as.data.frame(ndata)
        colnames(ndata)<- var.names
      }
      return(list(validation=validation,data =ndata))
    }
  }
}
```

Using the function above, get the data according to the arguments.

Note that I used the token provided by Arthur

```
FSA_token <- "yGa9ysvivTWUUteZVeQUY4rMsCRBcxGTkDbcFbL773EMywrn6cLEDgIq7Wg3vfwZbYkXyhGsblu0wjZjbiwc2EZC0
s3 <- get.Comtrade(r="826", p="710,724,757,276", ps="201001,201101,201201,201301", freq="M",maxrec=1000
```

Convert the data downloaded using the API into a dataframe à-la-R

```
df <- data.frame(Reduce(rbind, s3))
#summary(df)
```

Fix the dataframe by removing the first empty row (IDK why this happens...)

```
df <- df[complete.cases(df$pfCode),]
#summary(df)
```

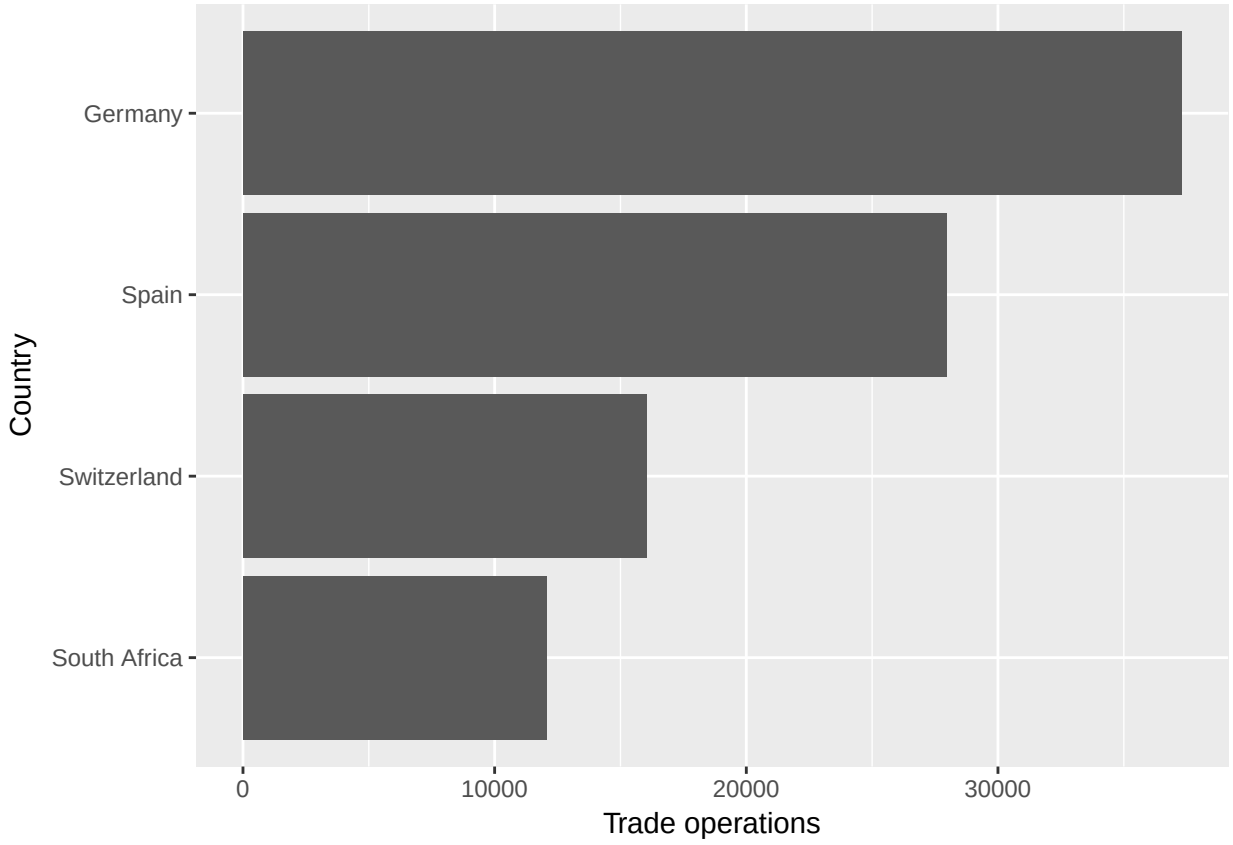
Add a new variable (use *mutate*) named *parent* that contains the two first digits of a commodity code

```
df <- df %>% mutate(parent = str_sub(cmdCode,1,2))
```

This plot shows the relevance of the trade with UK for some selected countries

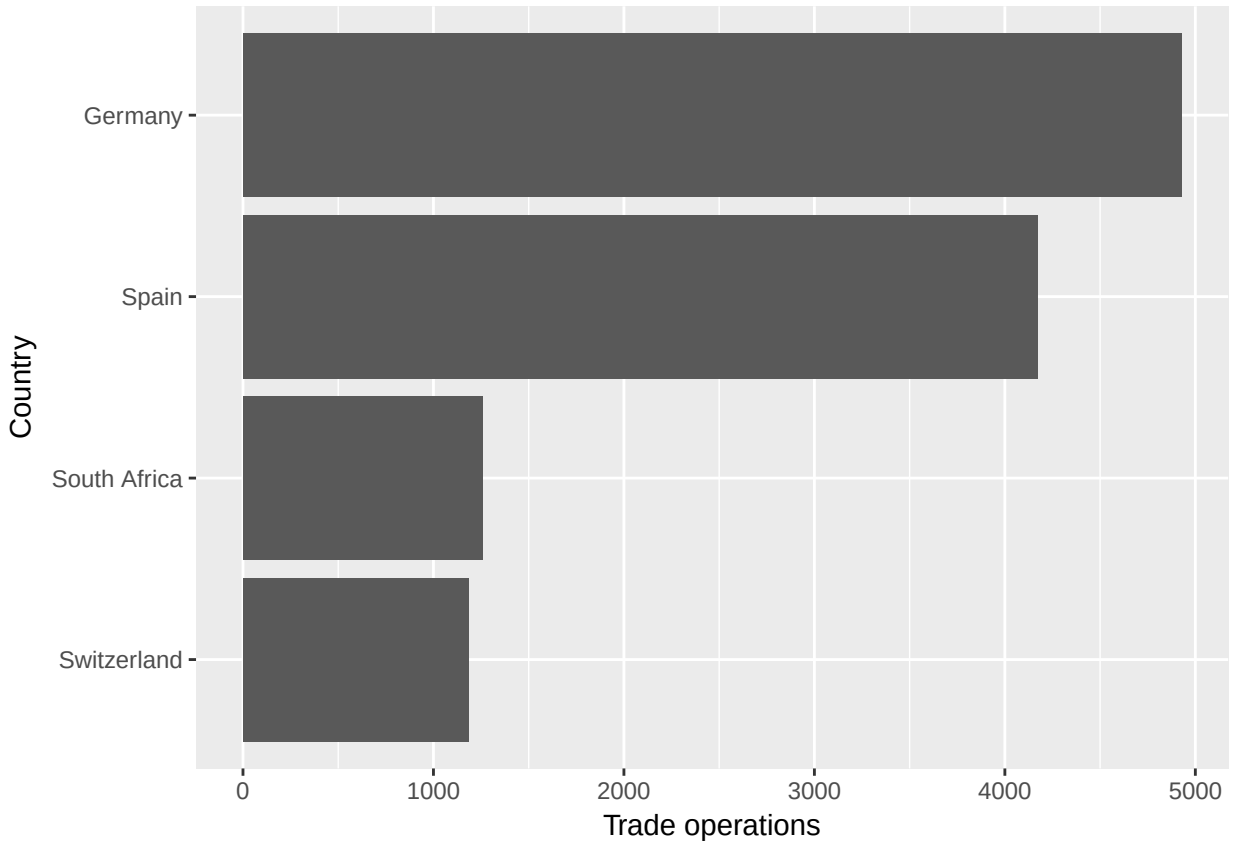
All comodities considered

```
vv <- df %>% group_by(ptTitle) %>% summarize(num = n())
ggplot(data=vv) + geom_bar(mapping = aes(x=reorder(ptTitle,num),y=num),stat='identity')+coord_flip() +
```



Assuming that the commodities corresponding to food have a parent code ranging from 01 to 24, this plot is exactly the same as above but just for food trade:

```
vvfood <- df %>% filter(parent<25) %>% group_by(ptTitle) %>% summarize(num = n())
ggplot(data=vvfood) + geom_bar(mapping = aes(x=reorder(ptTitle,num),y=num),stat='identity')+coord_flip()
```



The *class* for both variables (columns) *TradeValue* and *NetWeight* is 'factor'.

They should be 'numeric' since they are *numbers*, not *levels*.

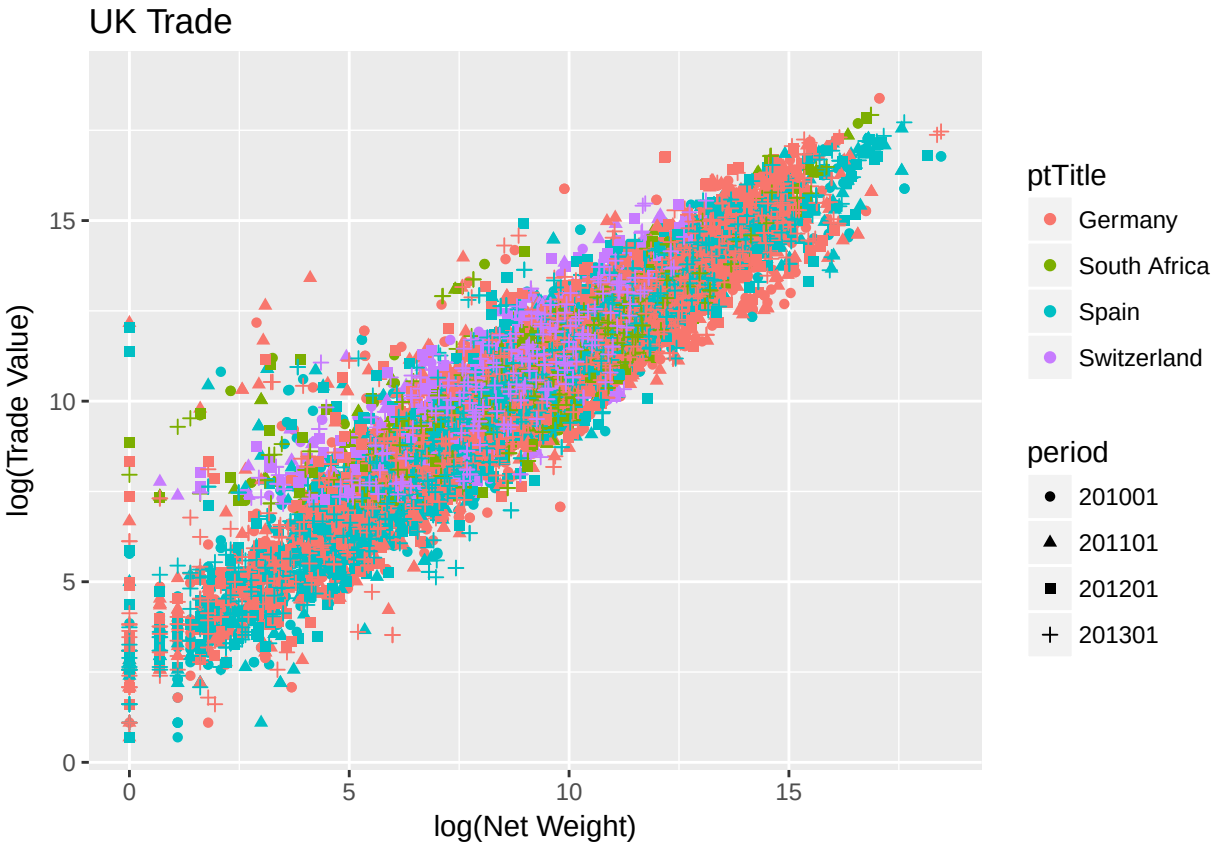
Fixing this here...

```
df$TradeValue <- as.numeric(as.character(df$TradeValue))
df$NetWeight  <- as.numeric(as.character(df$NetWeight))
```

Plot the Net Weight vs Trade Value for Germany, South Africa, Switzerland and Spain

Full time range considered.

```
ggplot(data=df %>% filter(parent<25)) +
  geom_point(mapping = aes(x=log(NetWeight),y=log(TradeValue),color=ptTitle,shape=period)) +
  labs(x="log(Net Weight)",y="log(Trade Value)",title="UK Trade")
```



Show some temporal evolution of the trade intensity with each country

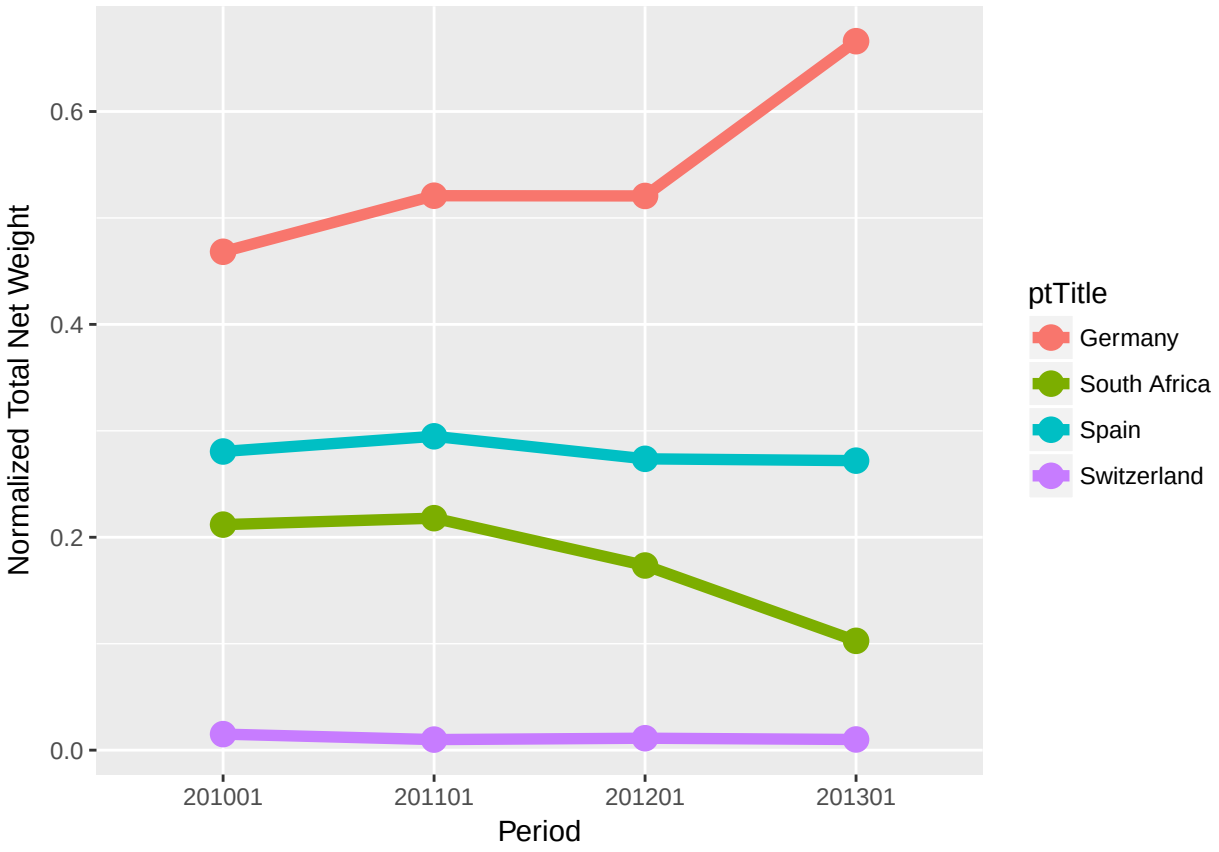
Simplify the dataframe by selecting only the relevant variables (columns) and remove all NA (not available) data entries that may mess the analysis up.

Note: since there are less data for 2012, it is necessary to normalize by the number of ocurrences

for each trade operation per country and per year.

That's why the total Weight (We) is divided by the the total number of trade operations (n)

```
tmp <- df %>% select(period,ptTitle,cmdCode,NetWeight,TradeValue,parent)
tmp <- tmp[complete.cases(tmp), ]
new <- tmp %>% group_by(period,ptTitle) %>% summarize(We = sum(NetWeight),Tr=sum(TradeValue),num=n())
ggplot(data = new) + geom_line(mapping =  aes(x=period,y=We/1e6/num,group=ptTitle,color=ptTitle),size=2
                   geom_point(mapping = aes(x=period,y=We/1e6/num,group=ptTitle,color=ptTitle),size=4
                   labs(x="Period",y="Normalized Total Net Weight")
```


Normalized Total Net Weight
0.6
0.4
0.2
0.0
201001
201101
201201
201301
Period
ptTitle
Germany
South Africa
Spain
Switzerland