# R Project - Aggregate Data in ADH

Ulrich Bergmann Lachlan Deer

We want to look for some aggregate facts about the US economy and its trade patterns to get a sense of how some macroeconomic indicators had evolved around China's WTO accession. In particular we will show that

- Expansion of Chinese Trade. Essentially all of US trade growth since the 1990s is from the expansion
- Fall in Real Interest Rates Around the time the Chinese trade expanded.
- Expansion of the Trade Deficit during this time period.

## Load Necessary Packages

Install the ones you do not have yet.

```
library("fredr")
library("purrr")
library("dplyr")
library("readr")
library("tidyr")
library("ggplot2")
library("magrittr")
library("lubridate")
library("PerformanceAnalytics")
```

## Get Data from Federal Reserve

We need the following variables from FRED:

```
codes = c("GDP", "IMP0015", "IMPCH", "EXP0015", "GS1", "CPILFESL")
```

We also need to tell FRED our API key to authenticate ourselves. For this course, you can use the following command:

```
api_key = "d89c49825894bf6e766a2e0afd8ba4f7"
fredr_set_key(api_key)
```

1. Use the `fredr_series_observations` command on a single variable to get the data for that variable
2. Now, find the option to pull only data starting at `1990-01-01`. Make sure you format this number as a date.
3. What happens if you use the previous command on `codes` instead of a single variable name?
4. Solve it by applying the correct map function which returns a dataframe by row-binding. Save your dataset as `df_raw`

## Data Transformations

For the rest of this part, take `df_raw` and save the transformed output as `df`:

5. Now split your data into columns
6. Rename your newly created columns as such:
   - `gdp` <- `GDP`,
   - `imp_ch` <- `IMPCH`,
   - `imp_all` <- `IMP0015`,
   - `exp_all` <- `EXP0015`,
   - `t_bill` <- `GS1`,
   - `cpi` <- `CPILFESL`
7. `cpi` is coded in billions of USD while exports and imports are in millions, multiply `cpi` by `1000` to have all values in millions
8. create additional variables for our date using the `lubridate` module. We want columns `year`, `quarter`, `month`, `day` that only contain this part from the `date` column. Find the correct functions in `lubridate` to achieve this
9. Sort your data by `date`

We see that gdp is coded quaterly while the imports and exports are per month. We need the data grouped annually and quarterly respectively for the next two parts:

## Data Grouping

10. Group `df` quarterly into a dataframe called `df_quarter`. In this dataset you want

- `start_date` as the minimum of date,
- `gdp`, `imp_all`, `imp_ch`, `exp_all` all aggregated as sums (how do you deal with NA's?)
- `cpi`, `t_bill` aggregated as averages

11. Group `df` annually into a dataframe called 'df_year. In this dataset you want

- `gdp`, `imp_all`, `imp_ch` aggregated as sums (how do you deal with NA's?)

## Fact 1: Increase of Imports from China and the Rest of the World

For this exercise, we are going to use `df_year`

1. drop data from 2020
2. We want to create the following three variables:

- `global_share = 100 * imp_all / gdp`
- `china_share = 100 * imp_ch / gdp`
- `nonchina_share = global_share - china_share`

3. Create the following graph

- years between `1991` and `2008`
- x-axis: years
- y-axis: `china_share` and `nonchina_share` (add to lines to your plot with different colors)
- a vertical line for `x == 2008`

## Fact 2: Increasing trade-deficit of the US

For this we will work with `df_year` again.

1. drop data from 2020
2. We want to create a variable `trade_deficit` (imports - exports)
3. And a variable `trade_deficit_share` for the share of the trade deficit compared to the gdp
4. Plot `trade_deficit_share` over time

5. Pick a nice color
6. Add a vertical line to the plot for `year == 2008`

## Fact 3: 400 basis point fall in real Interest rates leading into China Expansion

For this we will work with `df_quarter` again.

1. We need to calculate the inflation rate from the consumer price index (`cpi`) column
   - find out how to calculate this
   - look at the first lines of your dataset. What's the problem?
   - ungroup the dataset first before you repeat the previous step.
   - look at the first lines of your dataset again.
   - you now have the quaterly inflation rate
2. Try to apply the `Return.annualized` function from the `PerformanceAnalytics` package to the newly created `infl` column inside `mutate`
   - what's the problem?
3. We will solve this problem in two ways
   1. Use the function inside `map_dbl` instead (not in a `mutate`)
      - assing the output to a new column of `df_quarter` called `annum_return`
   2. Use `Vectorize(Return.annualized)` inside `mutate` instead an save the output to `annum_return2`
      - What does the function do?
4. Create a new variable called `real_r` as the difference between `t_bill` and one of your `annum_return` columns
5. Plot `real_r` against date
6. Add a vetical line at the date `2002-01-01`
   - Hint: You have to parse the date first as a `date` and then convert it into a `numerical` value