Our purpose is to write a program to automatically update a quarterly fiscal database for the Euro Area. The main difficulty of this exercise is to build long series that go as far as the 1980's.

We use two sources to build the database: the historical database developed in Paredes et al. (2014), which stops in 2013, and the latest Eurostat data. Throughout this article, we explain how we chained each series of PPP with the Eurostat data.

Both databases are taken without any seasonal adjustment. At the end of the post, chained data series are seasonally adjusted using the `seasonal` package developed by Sax (2016) using the X13 methodology.

To be automated, the recent points of the database are taken from DBnomics using the rdbnomics package. All the code is written in R, thanks to the RCoreTeam (2016) and RStudioTeam (2016).

The database will contain the following series:

- Direct taxes
- Indirect taxes
- Social security contributions by employees
- Social security contributions by employers
- Government consumption
- Government investment
- Government transfers
- Government subsidies
- Government compensation of employees
- Unemployment benefits
- Government debt
- Interest payments
- Total revenues
- Total expenditures

Historical data

First we get the historical series built by Paredes et al. (2014). Problem is, the series are not all directly usable: the series of social contribution by contributors do not exist before 1991.

```
url <- "PPP_raw.xls"

ppp <- read_excel(url, skip = 1)

ppp %<>%
  transmute(period      =   as.Date(as.yearqtr(`MILL. EURO, RAW DATA, NON-SEAS.
ADJUSTED, SMOOTHED ESTIMATES`, format="%YQ%q")),
            totexp      =   TOE,                # Total expenditures
            pubcons     =   GCN,                # General government consumption
expenditures
            pubinves    =   GIN,                # General government investment
            tfs         =   THN,                # Social payments
            unemp       =   `of which UNB`,     # Unemployment benefits (among
social payments)
            salaries    =   COE,                # Compensation of employees
            subs        =   SIN,                # Subsidies
            intpay      =   INP,                # General government interest
payments
            totrev      =   TOR,                # Total revenue
            indirtax    =   TIN,                # Total indirect taxes
            dirtax      =   DTX,                # Total direct taxes
            scr         =   as.numeric(SCR),    # Social contribution by
```

```
employers
            sce         =   as.numeric(SCE),    # Social contribution by
employees and self-employed
            sct         =   SCT,                # Total social security
contributions
            debt        =   MAL) %>%            # Euro area general government
debt
  filter(!is.na(period))
```

Assuming that the ratio of social contributions remains stable between employers and households before 1991, we can reconstruct the contribution of employees and employers using the series of total contribution. Using this technique we infer the series of social contribution by contributors before 1991.

```
# We calculate the ratio of social contribution by employers for the first point
in our series
prcent <-
  transmute(ppp, scr_sct=scr/sct) %>%
  na.omit() %>% first() %>% as.numeric()

# Using the ratio, we reconstruct earlier social contribution by contributor
scr_sce_before91 <-
  filter(ppp, is.na(scr)) %>%
  select(period, sct, scr, sce) %>%
  transmute(period,
            scr=prcent*sct,
            sce=sct-scr) %>%
  gather(var, value, -period)

# We reinject the constructed series in the ppp database
ppp %<>%
  select(-sct) %>%
  gather(var, value, -period, na.rm = TRUE) %>%
  bind_rows(scr_sce_before91) %>%
  arrange(var, period)

maxDate <-
  ppp %>%
  group_by(var) %>%
  summarize(maxdate=max(period)) %>%
  arrange(maxdate)
kable(maxDate)
```

var	maxdate
debt	2013-10-01
dirtax	2013-10-01
indirtax	2013-10-01
intpay	2013-10-01
pubcons	2013-10-01
pubinves	2013-10-01
salaries	2013-10-01
sce	2013-10-01
scr	2013-10-01
subs	2013-10-01
tfs	2013-10-01
totexp	2013-10-01
totrev	2013-10-01

var	maxdate
unemp	2013-10-01

Recent data

Historical data series stop in 2013. For latest points, we use DBnomics to get Eurostat's data. Eurostat's series on social contributions and on unemployment benefits present difficulties as well. We thus download the series from DBnomics in three steps:

1. We take the series on social contributions and we treat them in order to build quarterly series by contributors.
2. We take the series on unemployment benefits and we treat them in order to build quarterly series.
3. We take the series that do not present any problems

Special case: social contributions

Download annual data

```
var_taken <- c('D613', # Annual Households' actual social contributions (D613)
for general govt only (S13)
               'D612', # Annual Employers' imputed social contributions
               'D611') # Annual Employers' actual social contributions (D611)
for general govt only (S13)
url_variables <- paste0(var_taken, collapse="+")
filter <- paste0('A.MIO_EUR.S13.',url_variables,'.EA19')

df <- rdb("Eurostat","gov_10a_taxag",mask = filter)

data_1 <-
  df %>%
  select(period,var=na_item,value) %>%
  spread(var,value) %>%
  mutate(sce=D613+D612,
         scr=D611) %>%
  select(-D611,-D612,-D613) %>%
  gather(var,value,-period) %>%
  mutate(year=year(period))
```

The series of actual social contributions present 2 problems: (i) they are not quarterly; (ii) they are available only up to 2018. We fix this problem by using the two series of quarterly net total social contributions and quarterly employers contribution for the total economy.

From annual to quarterly data

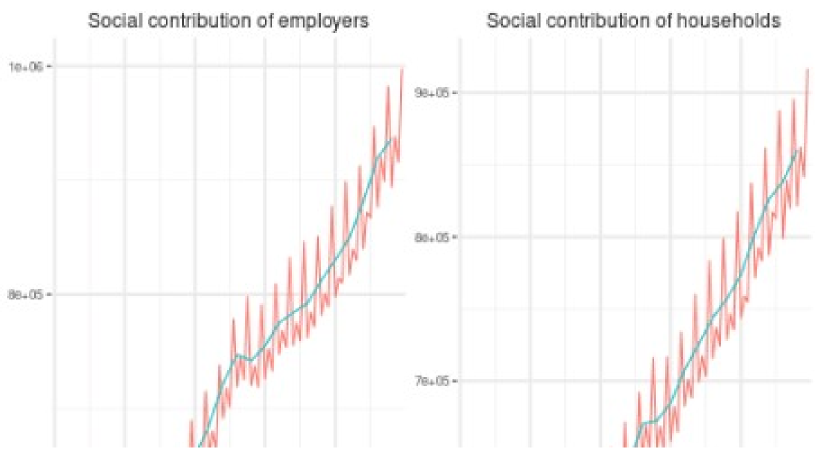
```
# Quarterly Net social contributions, receivable (D61REC) for general govt only
(S13)
df <- rdb("Eurostat","gov_10q_ggnfa",mask = "Q.MIO_EUR.NSA.S13.D61REC.EA19")
qsct <-
  df %>%
  transmute(period, var = 'sct', value) %>%
  mutate(year=year(period))

maxtime <- summarize(qsct,maxdate=max(period))
```

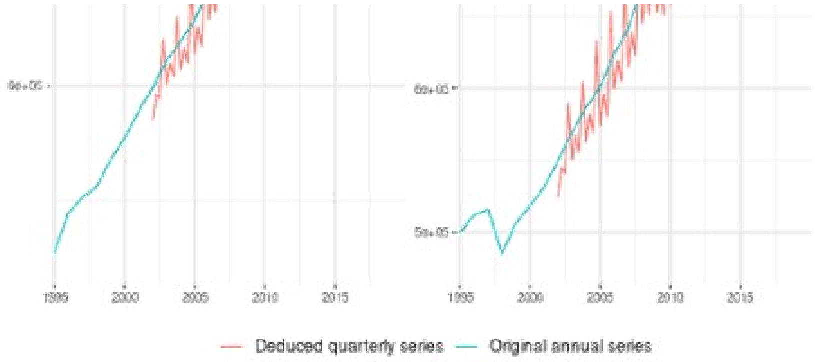
To turn the two annual series of social contributions into quarterly series, we use the series of quarterly total net social contributions to calculate the share of each contributor for each year. Using this share and the quarterly value of the total net social contributions, we can deduce the quarterly value of the net social contributions of each contributor.

```
# Calculate total amount of sct by year
qsct_a <-
  qsct %>%
  group_by(year) %>%
  summarise(value_a=sum(value))
qsct %<>% left_join(qsct_a, by="year")

# Convert data from annual to quarterly
qsce_uncomplete <-
  filter(data_1, var=="sce")  %>%
  full_join(qsct, by="year") %>%
  transmute(period=period.y,
            var=var.x,
            value=value.y*value.x/value_a) %>%
  filter(!is.na(value))

# Convert data from annual to quarterly
qscr_uncomplete <-
  filter(data_1, var=="scr")  %>%
  full_join(qsct, by="year") %>%
  transmute(period=period.y,
            var=var.x,
            value=value.y*value.x/value_a) %>%
  filter(!is.na(value))
```

We plot series to compare built quarterly series with annual series.

```
plot_treatment <-
  bind_rows(qscr_uncomplete, qsce_uncomplete) %>%
  mutate(Origin="Deduced quarterly series",
         value=4*value) %>%  # We multiply by 4 because on the plot we compare
quarterly level with annual levels
  bind_rows(mutate(data_1,Origin="Original annual series")) %>%
  mutate(var=ifelse(var=="sce","Social contribution of households","Social
contribution of employers")) %>%
  select(-year)

ggplot(plot_treatment,aes(period,value,colour=Origin)) +
  geom_line() +
  facet_wrap(~var,ncol=2,scales = "free_y") +
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
  theme(strip.text=element_text(size=12),
        axis.text=element_text(size=8)) +
  theme(legend.title=element_blank())
```





Most recent data

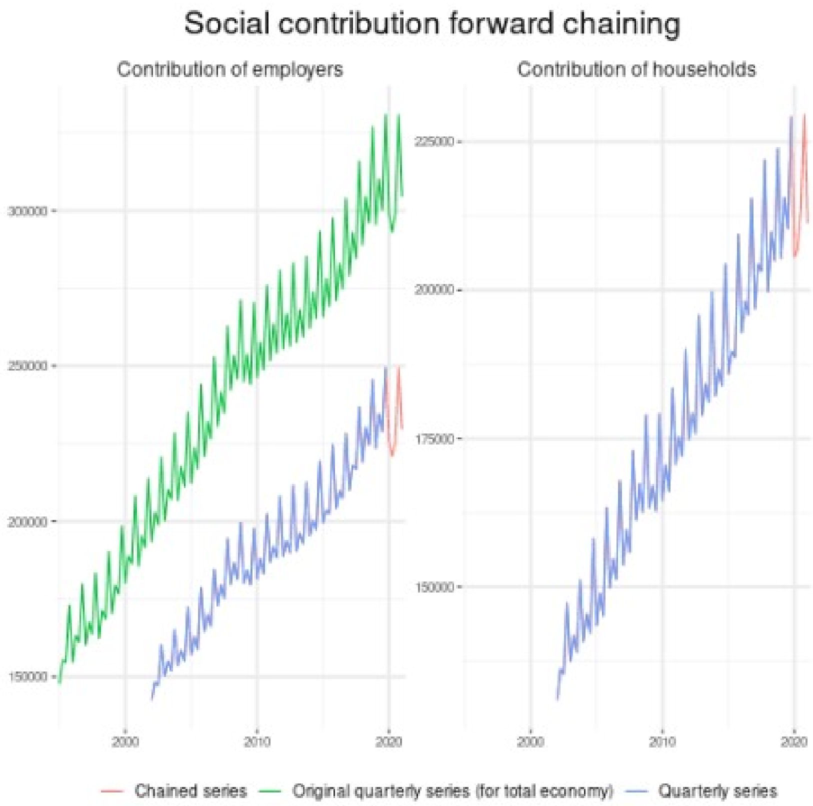
Now that we have the quarterly values, we use the series of total employers contribution for total economy along with the share of each contributors in total contributions to deduce latest points of contributions by households and employers.

```
# Quarterly Employers SSC for total economy
df <- rdb("Eurostat","namq_10_gdp",mask="Q.CP_MEUR.NSA.D12.EA19")
qscr_toteco <-
  df %>%
  transmute(period,var = 'scr',value) %>%
  mutate(year=year(period))

# Using recent data on employers total contribution we chain forward the social
contribution of employers
qscr <-
  chain(to_rebase = qscr_toteco,
        basis = qscr_uncomplete,
        date_chain = max(qscr_uncomplete$period),
        is_basis_the_recent_data=FALSE) %>%
  arrange(period)

# Assuming the ratio of social contribution by contributors remains constant
over time, we deduce social contribution of households
qsce <-
  bind_rows(qsce_uncomplete,
            select(qsct, period, value, var),
            qscr) %>%
  filter(period<=maxtime$maxdate) %>%
  spread(var,value, fill = 0) %>%
  transmute(period,
            sce=ifelse(period<=max(qsce_uncomplete$period),sce,sct-scr)) %>%
  gather(var, value, -period) %>%
  arrange(period)
```

Series of employers contribution are different in levels. Indeed, we are interested in social contributions of employers for general government only, and not for total economy. But the pattern of both series are very similar. So, by chaining them we take the variations from social contributions of employers for total economy and we apply them to the level of actual social contributions for general government only.

```
plot_treatment <-
  bind_rows(qscr_uncomplete,
            qsce_uncomplete) %>%
  mutate(Origin="Quarterly series") %>%
  bind_rows(mutate(qscr_toteco ,Origin="Original quarterly series (for total
economy)"),
            mutate(bind_rows(qsce,qscr), Origin="Chained series")) %>%
```

```
  mutate(var=ifelse(var=="sce","Contribution of households","Contribution of
employers")) %>%
  select(-year)

ggplot(plot_treatment,aes(period,value,colour=Origin)) +
  geom_line() +
  facet_wrap(~var,ncol=2,scales = "free_y") +
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
  theme(strip.text=element_text(size=12),
        axis.text=element_text(size=8)) +
  theme(legend.title=element_blank()) +
  ggtitle("Social contribution forward chaining")
```



Special case: unemployment benefits

We retrieve government social expenditures and compute their quaterly share for each year.

```
socialexp <-
  rdb("Eurostat","gov_10q_ggnfa",mask = "Q.MIO_EUR.NSA.S13.D62PAY.EA19") %>%
  mutate(year=year(period)) %>%
  select(period,value,year) %>%
  group_by(year) %>%
  mutate(sum=sum(value),
         ratio=value/sum) %>%
  ungroup() %>%
  select(-value,-year,-sum)
```

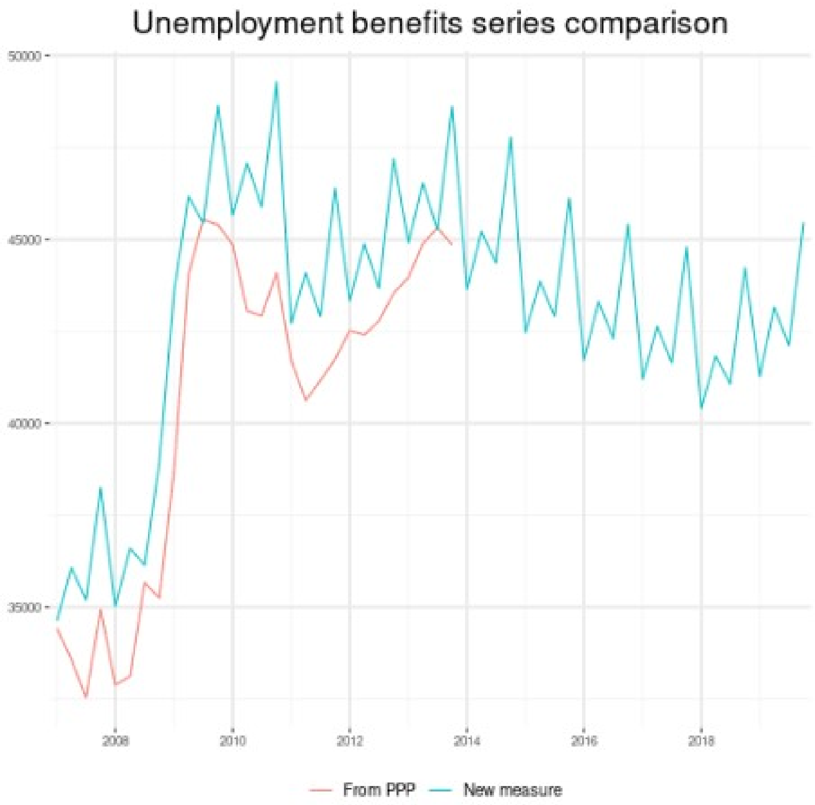
Then we retrieve the latest annual data on unemployment benefits, put them in a quarterly table and use the previous ratio of quarterly social expenditures to compute quarterly unemployment benefits.

```
df <- rdb("Eurostat","gov_10a_exp",mask = "A.MIO_EUR.S13.GF1005.TE.EA19")
recent_unemp <- df %>%
  mutate(year=year(period)) %>%
  select(period,value,year)
```

```
recent_unemp_q <-
  tibble(period=seq(min(recent_unemp$period),
                    length.out=nrow(recent_unemp)*4,
                    by = "quarter"),
         year=year(period)) %>%
  left_join(recent_unemp,by="year") %>%
  select(-period.y,-year) %>%
  rename(period=period.x)

unemp_q <-
  recent_unemp_q %>%
  inner_join(socialexp,by="period") %>%
  mutate(value=value*ratio,
         var="unemp") %>%
  select(-ratio)
```

We compare historical data with new quarterly series.

```
data_plot <-
  ppp %>%
  filter(var=="unemp") %>%
  mutate(var="From PPP") %>%
  bind_rows(mutate(unemp_q,var="New measure")) %>%
  filter(year(period)>=2007)

ggplot(data_plot,aes(period,value,colour=var))+
  geom_line()+
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
  theme(strip.text=element_text(size=12),
        axis.text=element_text(size=8)) +
  theme(legend.title=element_blank()) +
  ggtitle("Unemployment benefits series comparison")
```



Chaining recent data with historical

We now fetch the remaining series from DBnomics, none of the remaining series has to be treated before it can be used. We then include in the resulting dataframe the series of social contributions by contributors.

```
# List of var that can be taken on the first dataset
var_taken <- c('P3',                    # Public consumption
               'P51G',                  # Public gross fixed capital formation
               'D62PAY',                # Social transfers
               'D1PAY',                 # Compensation of employees
               'D3PAY',                 # Subsidies
               'D2REC',                 # Indirect taxes on production
               'D5REC',                 # Direct taxation on income and wealth
               'D41PAY',                # Interest payments
               'TE',                    # Total expenditures
               'TR')                    # Total revenue

# We build the URL, fetch the data and convert it to a data frame
url_variables <- paste0(var_taken, collapse="+")
filter <- paste0('Q.MIO_EUR.NSA.S13.', url_variables,'.EA19')
data_1 <- rdb("Eurostat","gov_10q_ggnfa",mask=filter)

# Government consolidated gross debt is in a different dataset so we make a
second call to DBnomics
data_2 <- rdb("Eurostat","gov_10q_ggdebt",mask="Q.GD.S13.MIO_EUR.EA19")

# We then bind the two data frame fetched on DBnomics
recent_data <-
  bind_rows(data_1, data_2) %>%
  transmute(value, period, var= as.factor(na_item))

# Used to harmonize DBnomics series var names with PPP
var_names <- c('pubcons', 'pubinves', 'tfs', 'salaries', 'subs', 'indirtax',
'dirtax','intpay','totexp','totrev','debt')
recent_data$var <- plyr::mapvalues(recent_data$var,c(var_taken,'GD'),var_names)

# We include the series of social contributions
var_names <- c(var_names, "sce", "scr","unemp")
recent_data %<>% bind_rows(qsce,qscr,unemp_q)
```

We can check the last available date for each series.

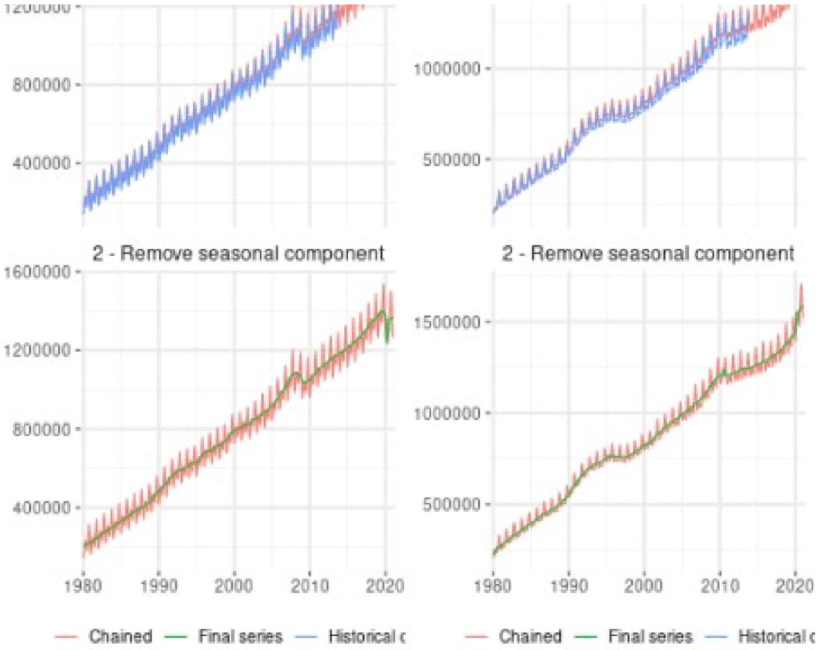
All that is left to do is to chain the dataframe of recent series with the historical database of Paredes et al. (2014). Once the data is chained we use the `seasonal` package to remove the seasonal component of each series. Hereafter, we will present the treatment on each variable to check graphically that what we obtain is consistent.

```
chained_data <-
  recent_data %>%
  chain(basis=.,
        to_rebase = filter(ppp, var %in% var_names),
        date_chain = "2007-01-01") %>%
  arrange(var, period) %>%
  select(period, var, value) %>%
  mutate(Origin = "Chained")

to_deseason <-
  chained_data %>%
  select(-Origin) %>%
```

```
  spread(var, value)

deseasoned <-
  bind_rows(lapply(unique(chained_data$var),
                   function(var) deseason(var_arrange = var,
                                          source_df = to_deseason))) %>%
  mutate(Origin = "Final series")

ppp %<>% mutate(Origin = "Historical data")

to_plot <- bind_rows(chained_data,
                     deseasoned,
                     ppp)
```

Total revenue and expenditures

```
plot_totrev <-
  to_plot %>%
  filter(var == "totrev",
         Origin != "Historical data") %>%
  mutate(ind2 = "2 - Remove seasonal component") %>%
  bind_rows(data.frame(filter(to_plot,var=="totrev",Origin !="Final series"),
ind2="1 - Chain series"))

plot_totexp <-
  to_plot %>%
  filter(var == "totexp",
         Origin != "Historical data") %>%
  mutate(ind2 = "2 - Remove seasonal component") %>%
  bind_rows(data.frame(filter(to_plot,var=="totexp",Origin !="Final series"),
ind2="1 - Chain series"))

p1 <- ggplot(plot_totrev,aes(period,value,colour=Origin))+
  geom_line()+
  facet_wrap(~ind2,scales = "free_y",ncol = 1)+
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
  ggtitle("Total revenue") +
  theme(legend.title=element_blank()) +
  theme(strip.text = element_text(size=12)) +
  theme(plot.title = element_text(size=16))

p2 <- ggplot(plot_totexp,aes(period,value,colour=Origin))+
  geom_line()+
  facet_wrap(~ind2,scales = "free_y",ncol = 1)+
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
  ggtitle("Total expenditures") +
  theme(legend.title=element_blank()) +
  theme(strip.text = element_text(size=12)) +
  theme(plot.title = element_text(size=16))

grid.arrange(arrangeGrob(p1,p2,ncol=2))
```



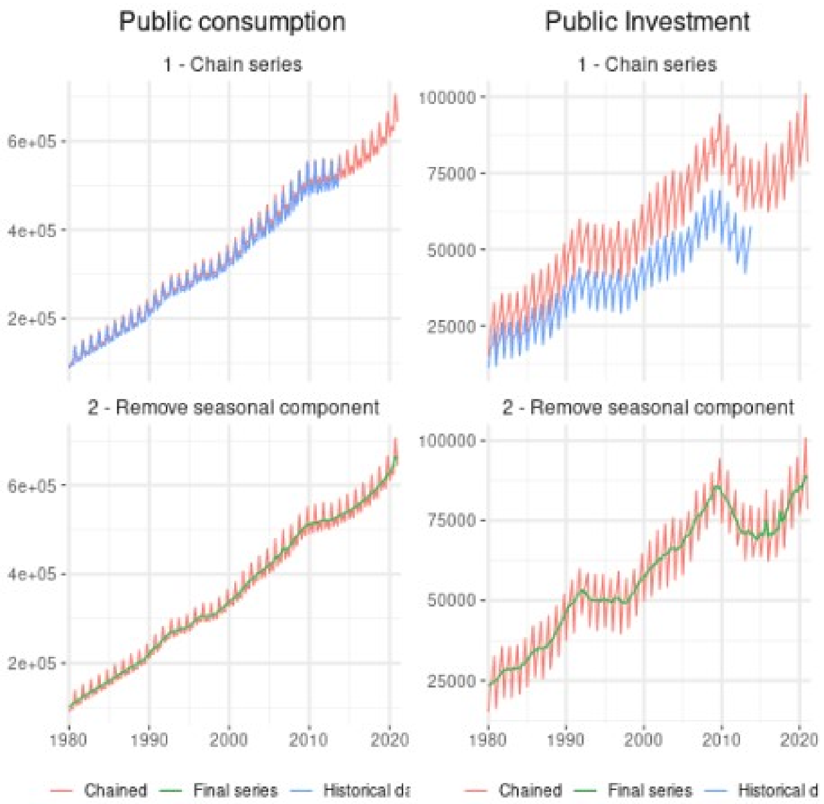
Public direct spending

The chained series of public consumption resembles strongly the historical series. Here our manipulation of the series allows us to create a long series without much loss.

There is on the chained series of investment a (visually) significant difference in level with the historical one. The method of chaining allows us to build a reasonable proxy for the series of public investment but at a certain loss.

```
plot_cons <-
  to_plot %>%
  filter(var == "pubcons",
         Origin != "Historical data") %>%
  mutate(ind2 = "2 - Remove seasonal component") %>%
  bind_rows(data.frame(filter(to_plot,var=="pubcons",Origin !="Final series"),
ind2="1 - Chain series"))

plot_inves <-
  to_plot %>%
  filter(var == "pubinves",
         Origin != "Historical data") %>%
  mutate(ind2 = "2 - Remove seasonal component") %>%
  bind_rows(data.frame(filter(to_plot,var=="pubinves",Origin !="Final series"),
ind2="1 - Chain series"))

p1 <- ggplot(plot_cons,aes(period,value,colour=Origin))+
  geom_line()+
  facet_wrap(~ind2,scales = "free_y",ncol = 1)+
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
  ggtitle("Public consumption") +
  theme(legend.title=element_blank()) +
  theme(strip.text = element_text(size=12)) +
  theme(plot.title = element_text(size=16))

p2 <- ggplot(plot_inves,aes(period,value,colour=Origin))+
  geom_line()+
  facet_wrap(~ind2,scales = "free_y",ncol = 1)+
  scale_x_date(expand = c(0.01,0.01)) +
```

```
  theme + xlab(NULL) + ylab(NULL) +
  ggtitle("Public Investment") +
  theme(legend.title=element_blank()) +
  theme(strip.text = element_text(size=12)) +
  theme(plot.title = element_text(size=16))

grid.arrange(arrangeGrob(p1,p2,ncol=2))
```

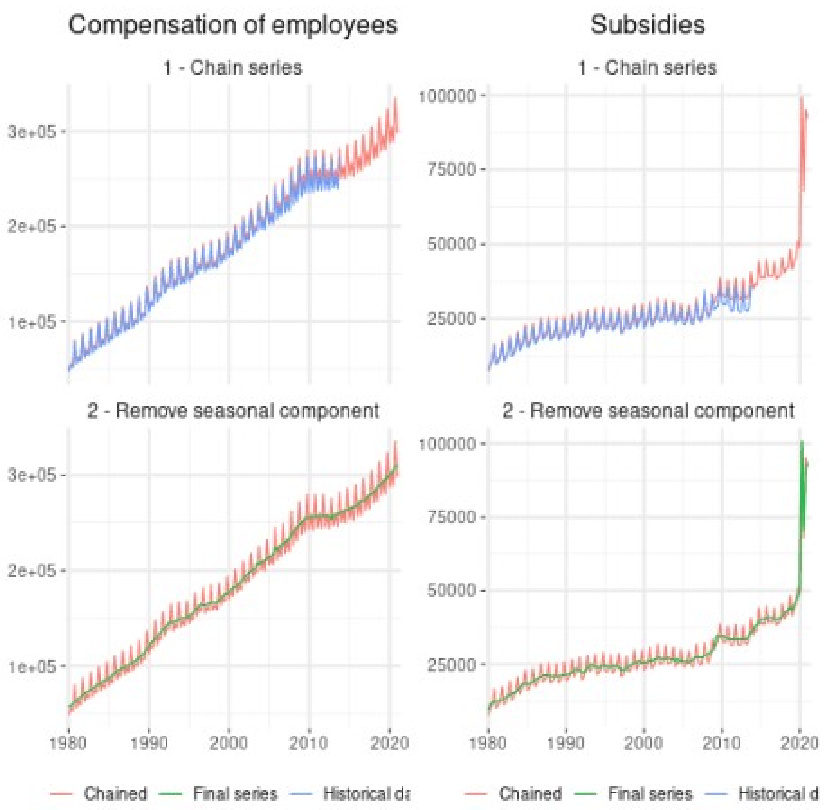


Specific spending

Both chaining seem to be consistent with the historical series. Here our manipulation does not entail much loss.

```
plot_salaries <-
  to_plot %>%
  filter(var == "salaries",
         Origin != "Historical data") %>%
  mutate(ind2 = "2 - Remove seasonal component") %>%
  bind_rows(data.frame(filter(to_plot,var=="salaries",Origin !="Final series"),
ind2="1 - Chain series"))

plot_subs <-
  to_plot %>%
  filter(var == "subs",
         Origin != "Historical data") %>%
  mutate(ind2 = "2 - Remove seasonal component") %>%
  bind_rows(data.frame(filter(to_plot,var=="subs",Origin !="Final series"),
ind2="1 - Chain series"))

p1 <- ggplot(plot_salaries,aes(period,value,colour=Origin))+
  geom_line()+
  facet_wrap(~ind2,scales = "free_y",ncol = 1)+
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
```

```
  ggtitle("Compensation of employees") +
  theme(legend.title=element_blank()) +
  theme(strip.text = element_text(size=12)) +
  theme(plot.title = element_text(size=16))

p2 <- ggplot(plot_subs,aes(period,value,colour=Origin))+
  geom_line()+
  facet_wrap(~ind2,scales = "free_y",ncol = 1)+
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
  ggtitle("Subsidies") +
  theme(legend.title=element_blank()) +
  theme(strip.text = element_text(size=12)) +
  theme(plot.title = element_text(size=16))

grid.arrange(arrangeGrob(p1,p2,ncol=2))
```

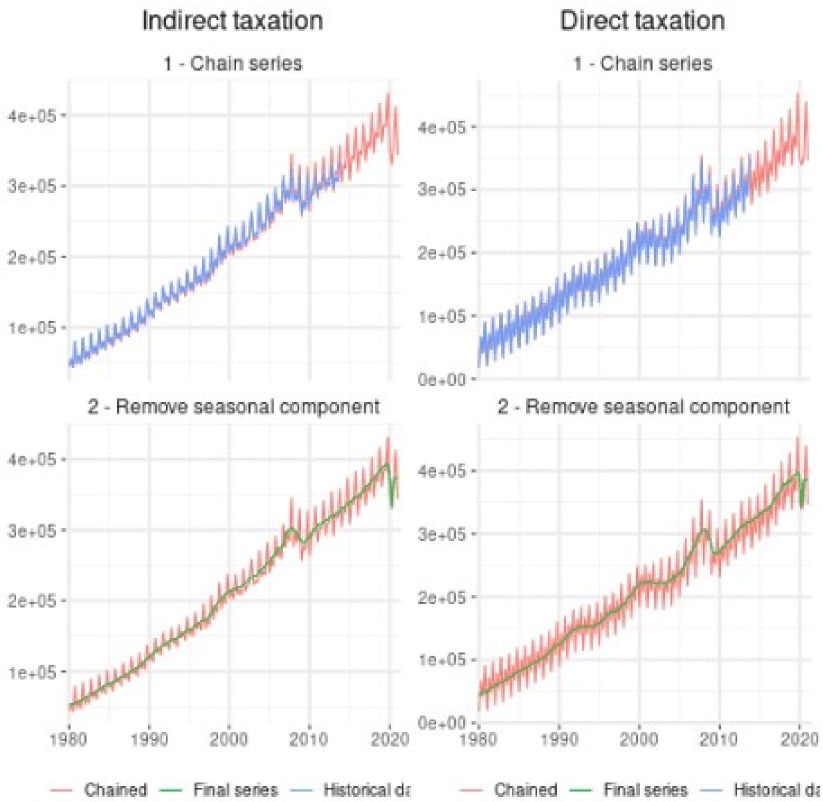


Taxes

Both chaining seem to be consistent with the historical series. Here our manipulation does not entail much loss.

```
plot_indir <-
  to_plot %>%
  filter(var == "indirtax",
         Origin != "Historical data") %>%
  mutate(ind2 = "2 - Remove seasonal component") %>%
  bind_rows(data.frame(filter(to_plot,var=="indirtax",Origin !="Final series"),
ind2="1 - Chain series"))

plot_dir <-
  to_plot %>%
  filter(var == "dirtax",
         Origin != "Historical data") %>%
```

```
  mutate(ind2 = "2 - Remove seasonal component") %>%
  bind_rows(data.frame(filter(to_plot,var=="dirtax",Origin !="Final series"),
ind2="1 - Chain series"))

p1 <- ggplot(plot_indir,aes(period,value,colour=Origin))+
  geom_line()+
  facet_wrap(~ind2,scales = "free_y",ncol = 1)+
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
  theme(legend.title=element_blank()) +
  theme(strip.text = element_text(size=12)) +
  theme(plot.title = element_text(size=16)) +
  ggtitle("Indirect taxation")

p2 <- ggplot(plot_dir,aes(period,value,colour=Origin))+
  geom_line()+
  facet_wrap(~ind2,scales = "free_y",ncol = 1)+
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
  theme(legend.title=element_blank()) +
  theme(strip.text = element_text(size=12)) +
  theme(plot.title = element_text(size=16)) +
  ggtitle("Direct taxation")

grid.arrange(arrangeGrob(p1,p2,ncol=2))
```

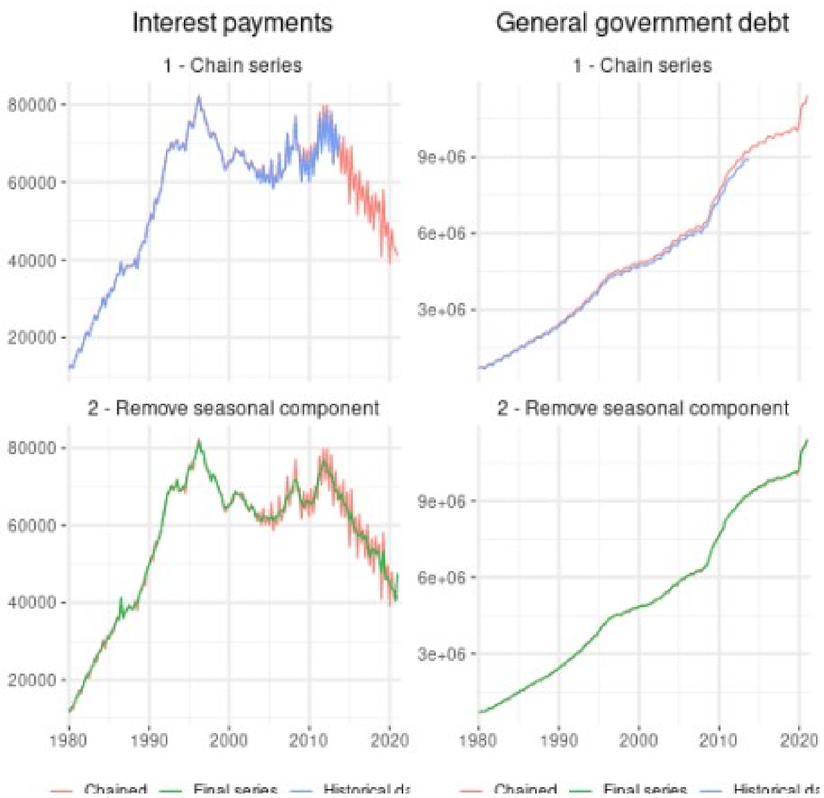


Debt and interest payments

The chained series of general government debt deviates slightly from the historical one, but the deviation is very thin and both chaining seem consistent. Here the seasonality of both series is weaker and the final series resemble strongly the chained ones.

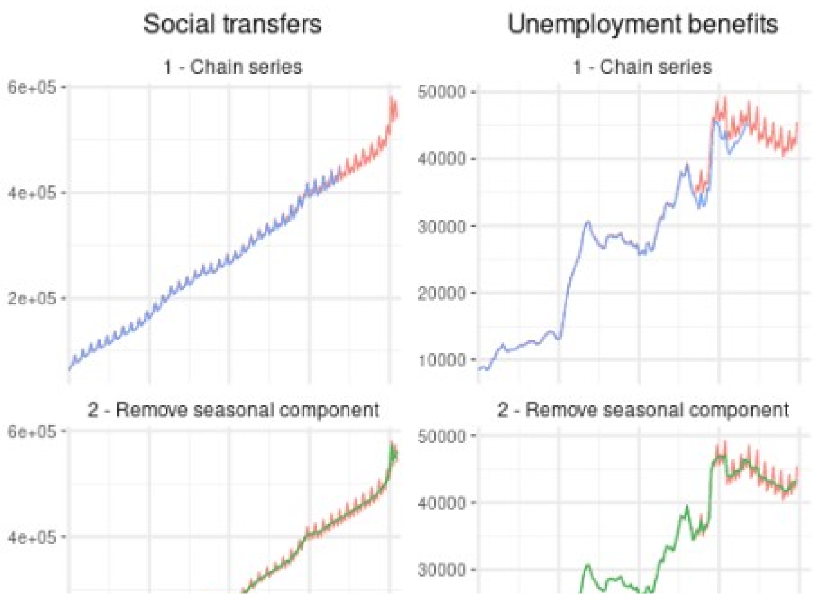
```
plot_debt <-
  to_plot %>%
```

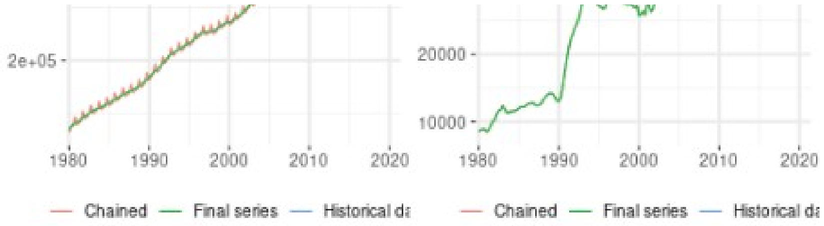
```
  filter(var == "debt",
         Origin != "Historical data") %>%
  mutate(ind2 = "2 - Remove seasonal component") %>%
  bind_rows(data.frame(filter(to_plot,var=="debt",Origin !="Final series"),
ind2="1 - Chain series"))

plot_intpay <-
  to_plot %>%
  filter(var == "intpay",
         Origin != "Historical data") %>%
  mutate(ind2 = "2 - Remove seasonal component") %>%
  bind_rows(data.frame(filter(to_plot,var=="intpay",Origin !="Final series"),
ind2="1 - Chain series"))

p1 <- ggplot(plot_intpay,aes(period,value,colour=Origin))+
  geom_line()+
  facet_wrap(~ind2,scales = "free_y",ncol = 1)+
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
  theme(legend.title=element_blank()) +
  theme(strip.text = element_text(size=12)) +
  theme(plot.title = element_text(size=16)) +
  ggtitle("Interest payments")

p2 <- ggplot(plot_debt,aes(period,value,colour=Origin))+
  geom_line()+
  facet_wrap(~ind2,scales = "free_y",ncol = 1)+
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
  theme(legend.title=element_blank()) +
  theme(strip.text = element_text(size=12)) +
  theme(plot.title = element_text(size=16)) +
  ggtitle("General government debt")

grid.arrange(arrangeGrob(p1,p2,ncol=2))
```



Total social transfers and unemployment benefits

```
plot_unemp <-
  to_plot %>%
  filter(var == "unemp",
         Origin != "Historical data") %>%
  mutate(ind2 = "2 - Remove seasonal component") %>%
  bind_rows(data.frame(filter(to_plot,var=="unemp",Origin !="Final series"),
ind2="1 - Chain series"))

plot_transf <-
  to_plot %>%
  filter(var == "tfs",
         Origin != "Historical data") %>%
  mutate(ind2 = "2 - Remove seasonal component") %>%
  bind_rows(data.frame(filter(to_plot,var=="tfs",Origin !="Final series"),
ind2="1 - Chain series"))

p1 <- ggplot(plot_transf,aes(period,value,colour=Origin))+
  geom_line()+
  facet_wrap(~ind2,scales = "free_y",ncol = 1)+
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
  theme(legend.title=element_blank()) +
  theme(strip.text = element_text(size=12)) +
  theme(plot.title = element_text(size=16)) +
  ggtitle("Social transfers")

p2 <- ggplot(plot_unemp,aes(period,value,colour=Origin))+
  geom_line()+
  facet_wrap(~ind2,scales = "free_y",ncol = 1)+
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
  theme(legend.title=element_blank()) +
  theme(strip.text = element_text(size=12)) +
  theme(plot.title = element_text(size=16)) +
  ggtitle("Unemployment benefits")

grid.arrange(arrangeGrob(p1,p2,ncol=2))
```





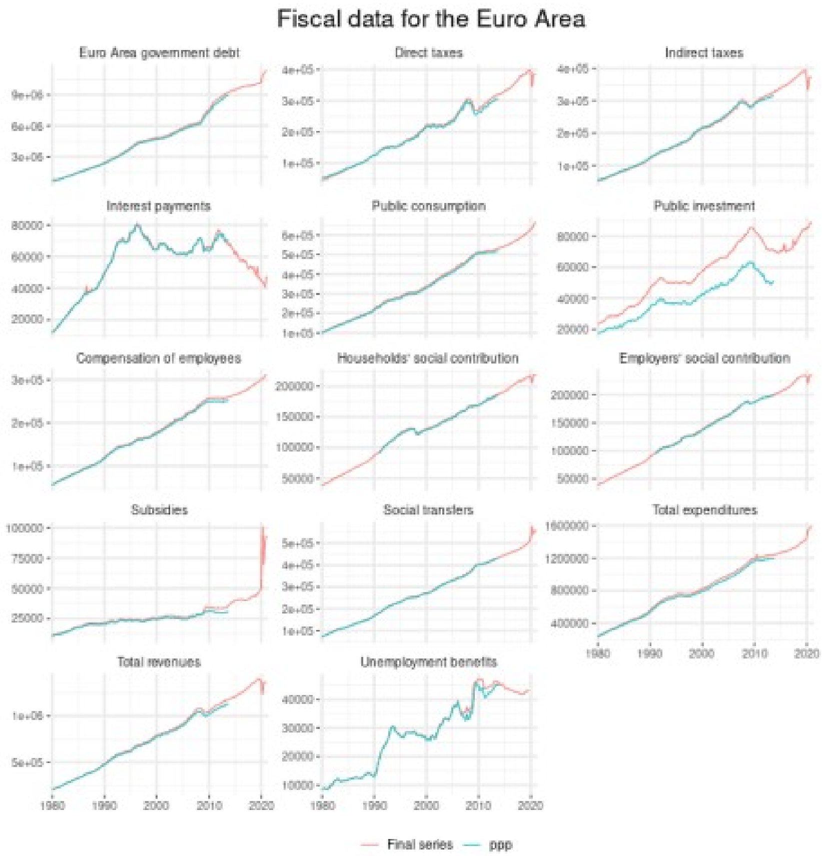
Building the final database

Comparing the obtained series with PPP

We want to check that the final database we created resembles the seasonally adjusted one of Paredes et al. (2014).

```
url <- "PPP_seasonal.xls"

pppSA <- read_excel(url, skip = 1)

pppSA %<>%
  transmute(period     =as.Date(as.yearqtr(`MILL. EURO, RAW DATA, SEASONALLY
ADJUSTED, SMOOTHED ESTIMATES`, format="%YQ%q")),
            totexp    =TOE,                 # Total expenditures
            pubcons   =GCN,                 # General government consumption
expenditure
            pubinves =GIN,                  # General government investment
            tfs       =THN,                 # Social payments
            salaries =COE,                  # Compensation of employees
            subs      =SIN,                 # Subsidies
            unemp     =`of whichUNB`,       # Unemployment benefits (among social
payments)
            intpay    =INP,                 # General government interest payments
            totrev    =TOR,                 # Total revenue
            indirtax =TIN,                  # Total indirect taxes
            dirtax    =DTX,                 # Total direct taxes
            scr       =as.numeric(SCR),     # Social contribution by employers
            sce       =as.numeric(SCE),     # Social contribution by employees and
self-employed
            debt      =MAL) %>%             # Euro area general government debt
  filter(!is.na(period))

plot_compare <-
  gather(pppSA, var, value, -period, convert= TRUE) %>%
  na.omit() %>%
  mutate(Origin="ppp") %>%
  bind_rows(deseasoned) %>%
  mutate(var=as.factor(var))

xlab_plot <- c('Euro Area government debt',
               'Direct taxes',
               'Indirect taxes',
               'Interest payments',
               'Public consumption',
               'Public investment',
               'Compensation of employees',
               "Households' social contribution",
               "Employers' social contribution",
               'Subsidies',
```

```
                'Social transfers',
                'Total expenditures',
                'Total revenues',
                'Unemployment benefits')

plot_compare$var <- plyr::mapvalues(plot_compare$var,levels(plot_compare$var),xl
ab_plot)

ggplot(plot_compare,aes(period,value,colour=Origin))+
  geom_line()+
  facet_wrap(~var,ncol=3,scales = "free_y")+
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL)+
  theme(legend.title=element_blank()) +
  theme(strip.text=element_text(size=10),
        axis.text=element_text(size=9))+
  ggtitle("Fiscal data for the Euro Area")
```



Final fiscal database for the Euro area

We eventually want to build a database close to Paredes et al. (2014). You can download all the raw series here.

```
EA_Fipu_rawdata <-
  deseasoned %>%
  select(-Origin) %>%
  spread(var,value)

EA_Fipu_rawdata %>%
  write.csv(file = "EA_Fipu_rawdata.csv",row.names = FALSE)
```

Then data are normalized by capita and price if needed, using data built to reproduce the Smets and Wouters (2003).

```
sw03 <-
  read.csv("http://shiny.nomics.world/data/EA_SW_rawdata.csv") %>%
  mutate(period=ymd(period)) %>%
  filter(period >="1980-01-01")

EA_Fipu_data <-
  EA_Fipu_rawdata %>%
  inner_join(sw03,by="period") %>%
  transmute(period=period,
            pubcons_rpc  = 100*1e+6*pubcons/(defgdp*pop*1000),
            pubinves_rpc = 100*1e+6*pubinves/(defgdp*pop*1000),
            salaries_rpc = 100*1e+6*salaries/(defgdp*pop*1000),
            subs_rpc     = 100*1e+6*subs/(defgdp*pop*1000),
            dirtax_rpc   = 100*1e+6*dirtax/(defgdp*pop*1000),
            indirtax_rpc = 100*1e+6*indirtax/(defgdp*pop*1000),
            tfs_rpc      = 100*1e+6*tfs/(defgdp*pop*1000),
            sce_rpc      = 100*1e+6*sce/(defgdp*pop*1000),
            scr_rpc      = 100*1e+6*scr/(defgdp*pop*1000),
            debt_rpc     = 100*1e+6*debt/(defgdp*pop*1000))

EA_Fipu_data %>%
  write.csv(file = "EA_Fipu_data.csv",row.names = FALSE)
```

You can download ready-to-use (normalized) data for the estimation here.

Appendix

Chaining function

To chain two datasets, we build a chain function whose input must be two dataframes with three standard columns (`period`, `var`, `value`). It returns a dataframe composed of chained values, ie the dataframe "to rebase" will be chained on the "basis" dataframe.

More specifically, the function :

- computes the growth rates from `value` in the dataframe of the 1st argument
- multiplies it with the value of reference chosen in `value` in the dataframe of the 2nd argument
- at the `date` specified in the 3rd argument.

```
chain <- function(to_rebase, basis, date_chain="2000-01-01",
is_basis_the_recent_data=TRUE) {

  date_chain <- as.Date(date_chain, "%Y-%m-%d")

  valref <- basis %>%
    filter(period == date_chain) %>%
    transmute(var, value_ref = value)
  # If chain is to update old values to match recent values
  if (is_basis_the_recent_data) {
    res <- to_rebase %>%
      filter(period <= date_chain) %>%
      arrange(desc(period)) %>%
      group_by(var) %>%
      mutate(growth_rate = c(1, value[-1]/lag(x = value)[-1])) %>%
      full_join(valref, by=c("var")) %>%
      group_by(var) %>%
      transmute(period, value=cumprod(growth_rate)*value_ref) %>%
      ungroup() %>%
```

```
      bind_rows(filter(basis, period>date_chain))
  } else {
    # If chain is to update recent values to match old values
    res <- to_rebase %>%
      filter(period >= date_chain) %>%
      arrange(period) %>%
      group_by(var) %>%
      mutate(growth_rate = c(1, value[-1]/lag(x = value, n = 1)[-1])) %>%
      full_join(valref, by=c("var")) %>%
      group_by(var) %>%
      transmute(period, value=cumprod(growth_rate)*value_ref) %>%
      ungroup() %>%
      bind_rows(filter(basis, period<date_chain))
  }

  return(res)
}
```

Seasonal adjustment

For the seasonal adjustment we just used a function to mutate the series into a time series, apply the function from the Sax (2016) package, mutate back into a dataframe.

```
deseason <- function(source_df=data_large, var_arrange="pubcons", ...){

local_data <- source_df

detrend <- local_data[var_arrange] %>%
  ts(start=c(1980,1), frequency = 4) %>%
  seas(na.action=na.exclude)

res <- as.numeric(final(detrend))

res <-  data_frame(value=res,
              period=local_data$period,
              var=var_arrange) %>%
    arrange(period) %>%
    transmute(period, var, value)
}
```