Automating update of the Smets and Wouters (2003) database

Published on October 15, 2015 by Thomas Brand (/author/thomas-brand.html), Nicolas Toulemonde (/author/nicolas-toulemonde.html)

database (/tag/database.html) , model (/tag/model.html) , estimation (/tag/estimation.html) , R (/tag/r.html)

Our purpose is to write a program that will automatically update the database used in the bayesian estimation of the DSGE model developped in Smets and Wouters (2003) for the Euro area. As no dataset for each variable fully covers the whole time period, we need to merge data from different sources.

Eight series are used in the original estimation of Smets and Wouters (2003) :

- GDP
- GDP Deflator
- Consumption
- Investment
- Employment
- Wage
- Working-age population (15-64)
- Interest rate

To those series we add 3 series :

- Hours worked
- Consumption Deflator
- Investment Deflator

In the original database of Smets and Wouters (2003), employment is used as a proxy of the hours worked whose series did not exist yet. Hours worked series is now available and we propose to build a long series as explained below.

One main difficulty is the multiplicity of the sources to obtain quarterly data for the Euro area since 1970. Of course, such an aggregation could seem a bit artificial to the extent that the Euro area was highly hypothetical at that time, but papers like Smets and Wouters (2003) show that it could be interesting to consider it nonetheless. Keeping in mind these limits, we try to obtain one single database by merging data from :

- the Area-Wide Model (AWM), originally constructed by Fagan et al. (2001),
- the Conference Board,
- the European Central Bank (ECB),
- Eurostat.

The first three sources are used only for historical data from 1970Q1 to the end of the 1990's. Updates will be fed only with Eurostat data for the eleven series from DBnomics (https://db.nomics.world/). The DBnomics API (https://api.db.nomics.world/) is used with the

rdbnomics (https://cran.r-project.org/web/packages/rdbnomics/index.html) package. All the code is written in R, thanks to the RCoreTeam (2016) and RStudioTeam (2016).

All data are seasonally and working days adjusted, except the interest rate and the population. We also choose to smooth the population.

As explained below, we use an updated version of the AWM database, which includes a Euro area composed of 19 countries. We keep this convention in the definition of the Euro area in the rest of the post.

Historical data (1970 - end of the 1990's)

Three sources are used to construct the database until the end of the 1990's : the main is the AWM database, but we also use the Conference Board and the ECB databases.

AWM database

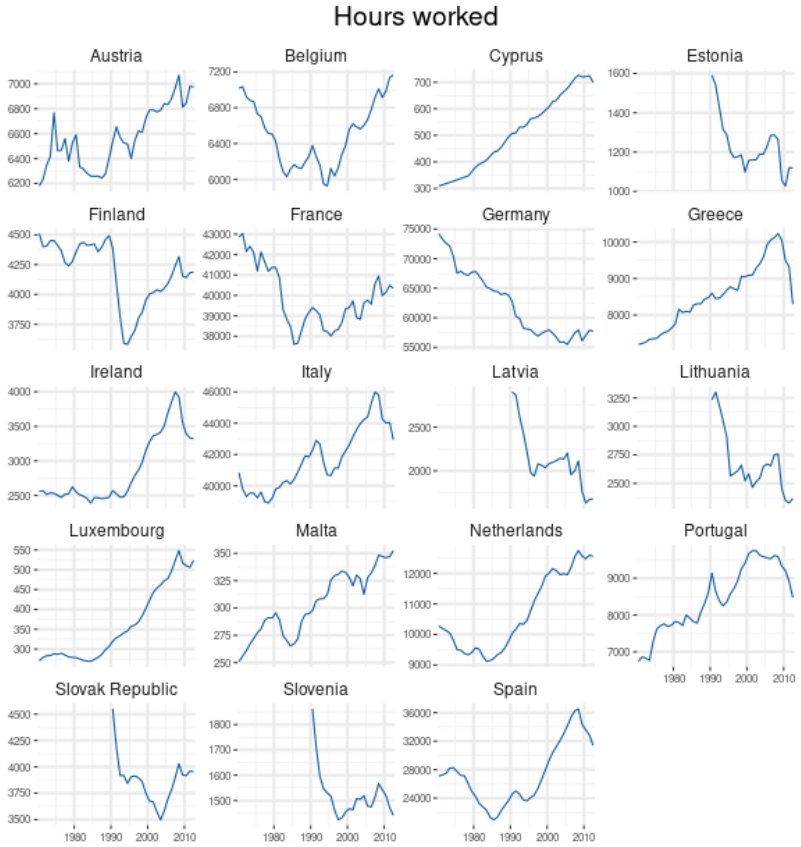
The AWM database was originally developped in Fagan et al. (2001). We use here an updated version of the database available on the EABCN website (http://www.eabcn.org/page/area-wide-model). We find here nine of the eleven series mentionned before. The exceptions are the hours worked and the population series, that the AWM database does not include. Those two series come from other sources and wil be treated separately.

```
link_to_awm <- "http://www.eabcn.org/sites/default/files/awm19up15.csv"

if (! "awm19up15.csv" %in% list.files()) {
  download.file(link_to_awm,
                destfile = "awm19up15.csv",
                method = "auto")
}

awm <- read.csv("awm19up15.csv", sep=",")

awm %<>%
  transmute(gdp            = YER, # GDP (Real)
            defgdp         = YED, # GDP Deflator
            conso          = PCR, # Private Consumption (Real)
            defconso       = PCD, # Consumption Deflator
            inves          = ITR, # Gross Investment (Real)
            definves       = ITD, # Gross Investment Deflator
            wage           = WIN, # Compensation to Employees (Nominal)
            shortrate      = STN, # Short-Term Interest Rate (Nominal)
            employ         = LNN, # Total Employment (Persons)
            period         = as.Date(as.yearqtr(X))) %>%
  gather(var, value, -period, convert = TRUE)
```

First special case : Hours worked

At the time Smets and Wouters (2003) wrote their paper, hours worked series didn't exist yet, so the authors used a formula linking employment to the hours worked in their model. Now, such a series is provided quarterly by Eurostat since 2000Q1. We propose here to build an historical hours worked series using data from the Conference Board until 1999Q4 and then data from Eurostat. More precisely, historical data come from the Total Economy Database (https://www.conference-board.org/data/economydatabase/index.cfm).

```
TED <- "TED---Output-Labor-and-Labor-Productivity-1950-2015.xlsx"
link_to_confboard <- paste0("https://www.conference-board.org/retrievefile.cfm?fil
ename=",TED,"&type=subsite")

if (! TED %in% list.files()) {
  download.file(link_to_confboard,
                destfile = TED,
                mode="wb")
}

# 19 countries in the Euro area (same as the AWM database)
EAtot_names <- c("Austria", "Belgium", "Cyprus", "Estonia", "Finland", "France",
                 "Germany", "Greece", "Ireland", "Italy", "Latvia", "Lithuania","L
uxembourg",
                 "Malta", "Netherlands", "Portugal", "Slovak Republic","Slovenia",
"Spain")

hours_confboard <-
  read_excel(TED, "Total Hours Worked", skip=2) %>%
  rename(country=Country)  %>%
  filter(country %in% EAtot_names) %>%
  select(-1) %>%
  gather(period, value, -country, na.rm=TRUE) %>%
  mutate(period = as.Date(paste0(period,"-07-01")),
         value = as.numeric(value)) %>%
  filter(period >= "1970-07-01" & period <= "2012-07-01")

ggplot(hours_confboard,aes(period,value)) +
  geom_line(colour=blueObsMacro) +
  facet_wrap(~country,ncol=4,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)) +
  ggtitle("Hours worked")
```



There are still two problems with such a series : first, the series does not cover all the 19 countries inside the Euro area for the whole period; second, data are annual.

Complete the hours worked series before 1990

Data from 19 countries of the Euro area are available in the Conference Board file only since 1990.

```
hours_confboard %>%
  group_by(period) %>%
  summarize(number_countries = length(country)) %>%
  tail(n=-12) %>%
  kable()
```

period	number_countries
1982-07-01	14
1983-07-01	14
1984-07-01	14
1985-07-01	14
1986-07-01	14
1987-07-01	14
1988-07-01	14
1989-07-01	14
1990-07-01	19
1991-07-01	19
1992-07-01	19
1993-07-01	19
1994-07-01	19
1995-07-01	19
1996-07-01	19
1997-07-01	19
1998-07-01	19
1999-07-01	19
2000-07-01	19
2001-07-01	19
2002-07-01	19
2003-07-01	19
2004-07-01	19
2005-07-01	19
2006-07-01	19
2007-07-01	19
2008-07-01	19
2009-07-01	19
2010-07-01	19
2011-07-01	19
2012-07-01	19

Indeed, between 1970 and 1990, the hours worked from five countries of Eastern Europe (Estonia, Latvia, Lithuania, Slovak Republic, Slovenia) are missing. So we choose to use the growth rates of the sum of hours worked series for the 14 countries available before 1990 to complete the series of the sum of hours worked over the 19 countries after 1990. It seems legitimate because since 1990, those 14 countries have represented more than 95% of the total hours worked. The `chain` function used here is detailed in the appendix.

```
# sum over the 14 countries
EA14_names <- c(filter(hours_confboard,period=="1970-07-01")$country)
hours_confboard_14 <-
  hours_confboard %>%
  filter(country %in% EA14_names) %>%
  group_by(period) %>%
  summarize(value = sum(value),
            var = "hours")

# sum over the whole countries
hours_confboard_tot <-
  hours_confboard %>%
  group_by(period) %>%
  summarize(value = sum(value),
            var = "hours")

hours_confboard_chained <-
  chain(to_rebase = hours_confboard_14,
        basis = hours_confboard_tot,
        date_chain = "1990-07-01")
```

Convert the annual hours worked series to quarterly data before 2000

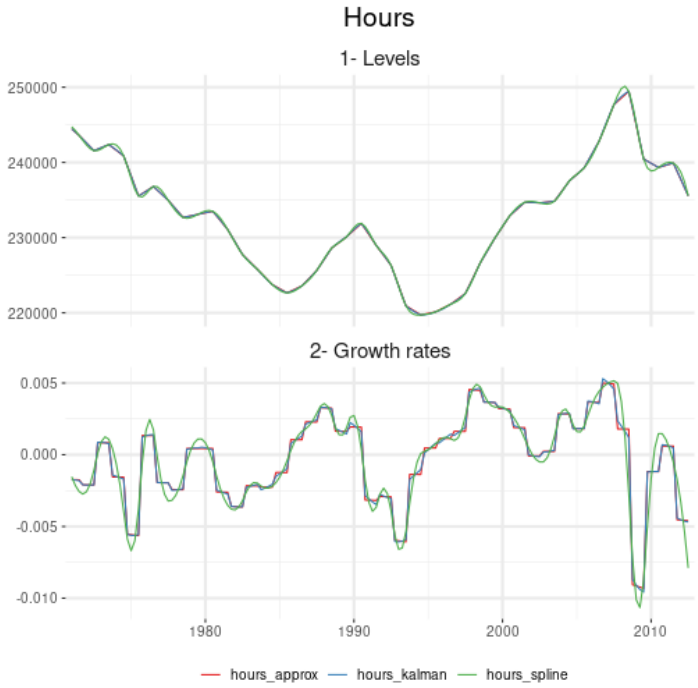
Once the annual data have been completed since 1970, they have to be turned into quarterly data.

```
hours_confboard_chained_q <-
  tibble(period=seq(as.Date("1970-07-01"),
                    as.Date("2012-07-01"),
                    by = "quarter"),
         value=NA) %>%
  left_join(hours_confboard_chained,by="period") %>%
  select(-value.x) %>%
  rename(value=value.y)
```

Several methods of interpolation are tested :

- constant quarterly growth rate over one year
- cubic spline
- Kalman filter

```
hours <- hours_confboard_chained_q

hours_approx <-
  hours %>%
  mutate(value=na.approx(value),
         var="hours_approx")

hours_spline <-
  hours %>%
  mutate(value=na.spline(value),
         var="hours_spline")

hoursts <- ts(hours$value,start=c(1970,4),f=4)
smoothed_hoursts <- tsSmooth(StructTS(hoursts,type="trend"))[,1]

hours_StructTS <-
  hours %>%
  mutate(value=smoothed_hoursts,
         var="hours_kalman")

hours_filtered <- bind_rows(hours_approx,hours_spline,hours_StructTS)

hours_filtered_levgr <-
  hours_filtered %>%
  mutate(value=log(value)-log(lag(value))) %>%
  data.frame(ind2="2- Growth rates") %>%
  bind_rows(data.frame(hours_filtered,ind2="1- Levels")) %>%
  filter(period>="1971-01-01")

ggplot(hours_filtered_levgr,aes(period,value,colour=var))+
  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()) +
  ggtitle("Hours")
```



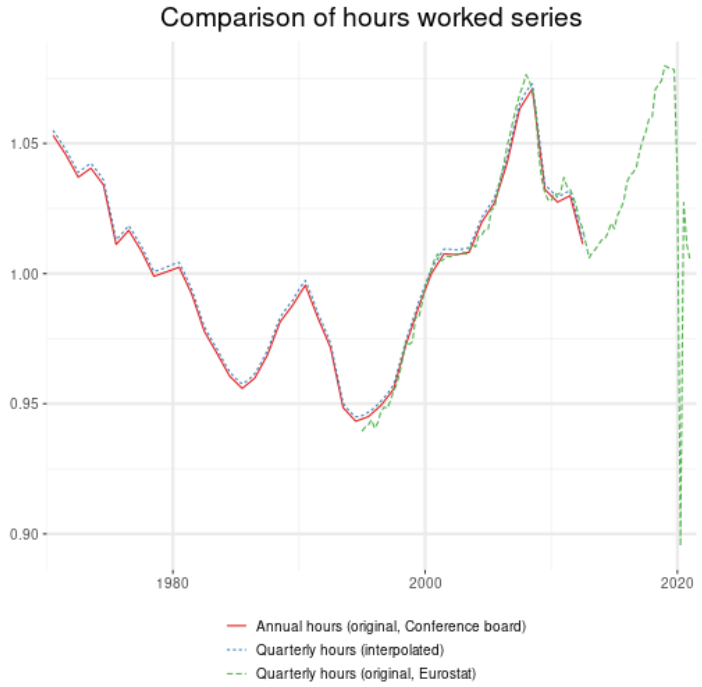
We retain the Kalman filter method of interpolation because we want to avoid the jump each first quarter in the growth rate implied by the first method, and the high volatility implied by the third filtering.

```
hours <- hours_StructTS %>%
  mutate(var="hours")
```

Compare the different series of hours worked

We want to check graphically that the interpolation of annual hours worked with the Kalman filter is consistent with raw data available in the most recent period. Remind that the interpolation is used only before 2000, we present interpolated data after this date in the plots only to check the consistency of our filtering, but will not use them after. To get recent Eurostat data, we use a plugin function (https://cran.r-project.org/web/packages/rdbnomics/index.html) from DBnomics (https://db.nomics.world/).

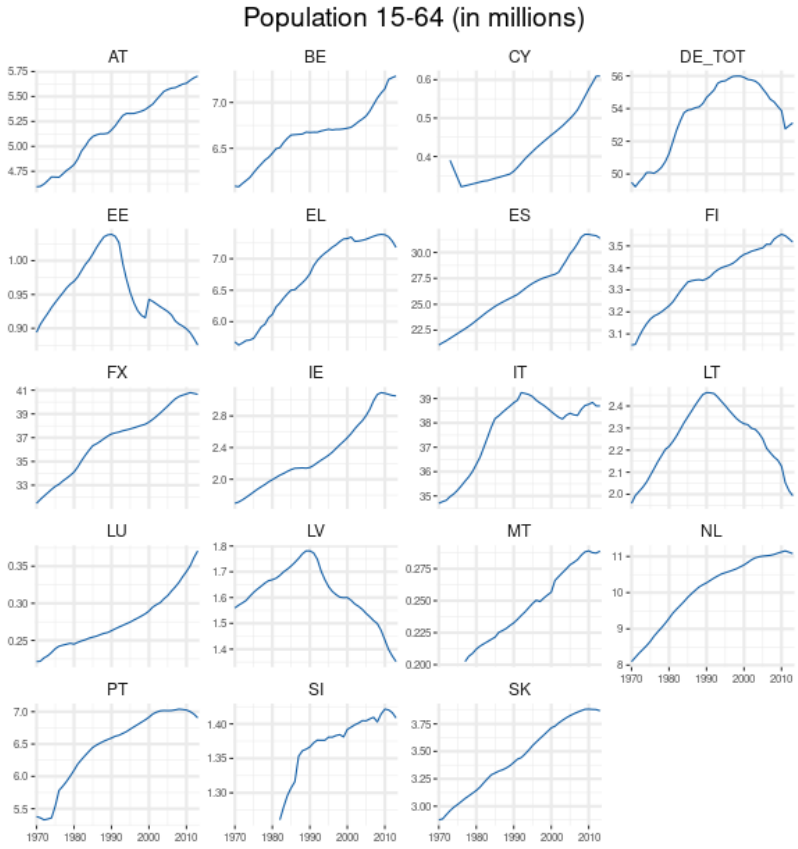
```
# convert Conference board annual hours worked series in 2000 basis index
valref <- filter(hours_confboard_chained,period=="2000-07-01")$value
hoursconfboard_ind <-
  hours_confboard_chained %>%
  transmute(period=period,
            var="Annual hours (original, Conference board)",
            value=value/valref)

# Quarterly hours worked series from Eurostat
df <- rdb(ids = "Eurostat/namq_10_a10_e/Q.THS_HW.TOTAL.SCA.EMP_DC.EA19")
eurostat_data <-
  df %>%
  select(period,value) %>%
  mutate(var = "Quarterly hours (original, Eurostat)")

valref <-
  eurostat_data %>%
  filter(year(period)==2000) %>%
  summarize(value=mean(value))

eurostat_data_ind <-
  eurostat_data %>%
   mutate(value=value/valref$value)

# convert interpolated hours worked series in 2000 basis index
valref <-
  hours %>%
  filter(year(period)==2000) %>%
  summarize(value=mean(value))

hours_ind <-
  hours %>%
  transmute(period,
            var="Quarterly hours (interpolated)",
            value=value/valref$value)


check <- bind_rows(hoursconfboard_ind,
                   eurostat_data_ind,
                   hours_ind)

ggplot(check, aes(period, value, group = var, linetype = var, colour = var)) +
  geom_line() +
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
  theme(legend.title=element_blank()) +
  guides(col=guide_legend(ncol=1),lty=guide_legend(ncol=1)) +
  ggtitle("Comparison of hours worked series")
```



Second special case : Population

The second special case is the population series. Quarterly population series only exist for the Euro Area after 2005. We propose here to construct an historical population series using annual data from Eurostat by country until 2005 and then original Euro area quarterly population series from Eurostat through DBnomics.

```
# We build the URL for the DBnomics API to get annual population series for the 19
countries of the Euro Area
EAtot_code <- c("AT", "BE", "CY", "DE_TOT", "EE", "IE",
                "EL", "ES","FX", "IT", "LT","LV", "LU",
                "NL", "PT", "SK", "FI", "MT", "SI")
url_country <- paste0("A.NR.Y15-64.T.",paste0(EAtot_code, collapse = "+"))
df <- rdb("Eurostat","demo_pjanbroad",mask = url_country)

pop_eurostat_bycountry <-
  df %>%
  select(geo, period, value) %>%
  rename(country = geo) %>%
   filter(period >= "1970-01-01",
          period <= "2013-01-01")

plot_pop_eurostat_bycountry <-
  pop_eurostat_bycountry %>%
  mutate(value = value/1000000)

ggplot(plot_pop_eurostat_bycountry,aes(period,value)) +
  geom_line(colour=blueObsMacro) +
  facet_wrap(~country,ncol=4,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)) +
  ggtitle("Population 15-64 (in millions)")
```



There are still two problems with such a series : first, the series does not cover all the 19 countries inside the Euro area for the whole period; second, data are annual.

Complete the population series before 1982

Between 1970 and 1981 (included), Eurostat only provides the population for 16 countries in the Euro area (not for Malta, Slovenia, and Cyprus).

```
pop_eurostat_bycountry %>%
  group_by(period) %>%
  summarize(number_countries = length(country)) %>%
  kable()
```

period	number_countries
1970-01-01	16
1971-01-01	16
1972-01-01	16
1973-01-01	17
1974-01-01	16
1975-01-01	16
1976-01-01	17
1977-01-01	17
1978-01-01	17
1979-01-01	17
1980-01-01	17
1981-01-01	17
1982-01-01	19
1983-01-01	18
1984-01-01	19
1985-01-01	19
1986-01-01	19
1987-01-01	19
1988-01-01	19
1989-01-01	19
1990-01-01	19

period	number_countries
1991-01-01	19
1992-01-01	19
1993-01-01	19
1994-01-01	19
1995-01-01	19
1996-01-01	19
1997-01-01	19
1998-01-01	19
1999-01-01	19
2000-01-01	19
2001-01-01	19
2002-01-01	19
2003-01-01	19
2004-01-01	19
2005-01-01	19
2006-01-01	19
2007-01-01	19
2008-01-01	19
2009-01-01	19
2010-01-01	19
2011-01-01	19
2012-01-01	19
2013-01-01	19

As those 16 countries represent more than 95% of the population of the EA19 in the last decades, we chain the series of the sum of the population for 19 countries to the series of the sum for 16 countries between 1970 and 1982. The method is the same as the one used with the series of hours worked: we use the `chain` function detailed in the appendix.

```
# We sum the annual population for 16 countries in the Euro area
EA16_code <- filter(pop_eurostat_bycountry,period=="1970-01-01")$country
pop_a_16 <-
  pop_eurostat_bycountry %>%
  filter(country %in% EA16_code) %>%
  group_by(period) %>%
  summarize(value = sum(value),
            var = "pop")

# We sum the annual population for all the available countries
pop_a_tot <-
  pop_eurostat_bycountry %>%
  group_by(period) %>%
  summarize(value = sum(value),
            var="pop")

# We use the chain function detailed in the appendix
pop_chained <-
  chain(to_rebase = pop_a_16,
        basis = pop_a_tot,
        date_chain = "1982-01-01")
```

Convert the annual population series to quarterly data before 2000

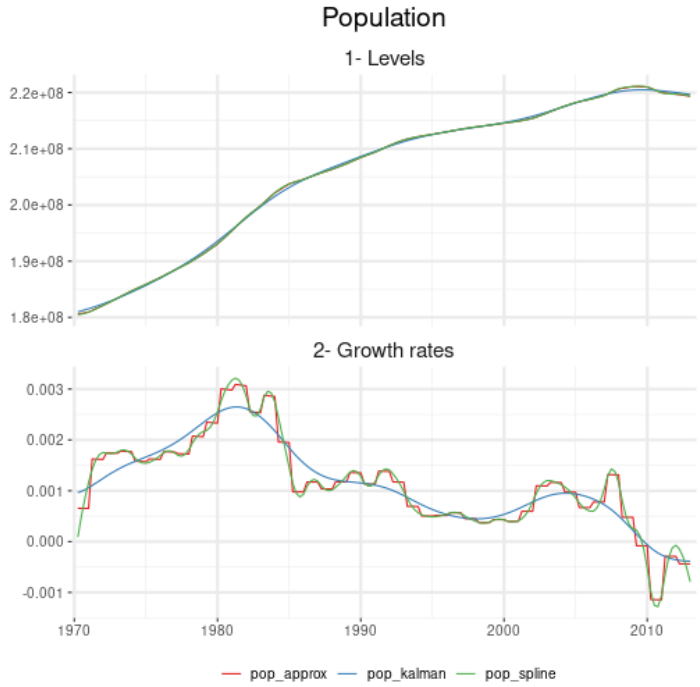
Once the annual data have been completed since 1970, they have to be turned into quarterly data.

```
pop_chained_q <-
  tibble(period=seq(as.Date("1970-01-01"),
                    as.Date("2013-01-01"),
                    by = "quarter"),
         value=NA) %>%
  left_join(pop_chained, by="period") %>%
  select(-value.x) %>%
  rename(value=value.y)
```

We test the same three methods of interpolation as for hours :

- constant quarterly growth rate over one year
- cubic spline
- Kalman filter

```
pop <- pop_chained_q

pop_approx <-
  pop %>%
  mutate(value=na.approx(value),
         var="pop_approx")

pop_spline <-
  pop %>%
  mutate(value=na.spline(value),
         var="pop_spline")

popts <- ts(pop$value,start=c(1970,1),f=4)
smoothed_popts <- tsSmooth(StructTS(popts,type="trend"))[,1]

pop_StructTS <-
  pop %>%
  mutate(value=smoothed_popts,
         var="pop_kalman")

pop_filtered <- bind_rows(pop_approx,pop_spline,pop_StructTS)

pop_filtered_levgr <- pop_filtered %>%
  mutate(value=log(value)-log(lag(value))) %>%
  data.frame(ind2="2- Growth rates") %>%
  bind_rows(data.frame(pop_filtered,ind2="1- Levels")) %>%
  filter(period>="1970-04-01")

ggplot(pop_filtered_levgr,aes(period,value,colour=var))+
  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()) +
  ggtitle("Population")
```



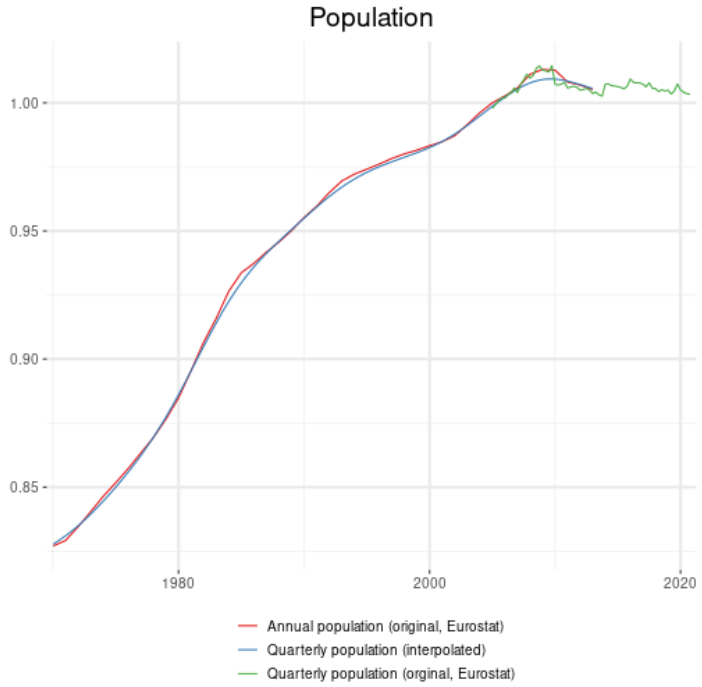
We retain the Kalman filter method of interpolation to avoid the jump each first quarter in the growth rate implied by the first method, and the high volatility implied by the third filtering.

```
pop <- pop_StructTS %>%
  mutate(var="pop")
```

Compare the different series of population

We want to check graphically that the interpolation of annual population with the Kalman filter is consistent with raw data available in the most recent period. Remind that the interpolation is used only before 2005, we present interpolated data after this date in the plots only to check the consistency of our filtering, but will not use them after.

```
# convert Conference board annual hours worked series in 2000 basis index
valref <- filter(pop_chained,period=="2005-01-01")$value
pop_a_ind <-
  pop_chained %>%
  transmute(period=period,
            var="Annual population (original, Eurostat)",
            value=value/valref)

# URL for quarterly population series
df <- rdb(ids="Eurostat/lfsq_pganws/Q.THS.T.TOTAL.Y15-64.POP.EA19")

eurostat_data <-
  df %>%
  select(period, geo, value) %>%
  rename(var= geo) %>%
  mutate(var= "Quarterly population (orginal, Eurostat)") %>%
  filter(period >= "2005-01-01")

valref <-
  eurostat_data%>%
  filter(year(period)==2005) %>%
  summarize(value=mean(value))

eurostat_data_ind <-
  eurostat_data %>%
  mutate(value=value/valref$value)

# convert interpolated population series in 2000 basis index
valref <-
  pop %>%
  filter(year(period)==2005) %>%
  summarize(value=mean(value))

pop_ind <-
  pop %>%
  transmute(period,
            var="Quarterly population (interpolated)",
            value=value/valref$value)

check <- bind_rows(pop_a_ind,
                   eurostat_data_ind,
                   pop_ind)

ggplot(check, aes(period, value,colour=var))+
  geom_line() +
  scale_x_date(expand = c(0.01,0.01)) +
  theme + xlab(NULL) + ylab(NULL) +
  theme(legend.title=element_blank()) +
  guides(col=guide_legend(ncol=1),lty=guide_legend(ncol=1)) +
  ggtitle("Population")
```



Recent data (since the end of the 1990's)

Once the historical database is created, all the variables can be found in Eurostat without transformation, through DBnomics.

```
old_data <- bind_rows(awm,
                      hours,
                      pop)

# URL for GDP/Consumption/Investment volumes and prices data
variable.list <- c("B1GQ","P31_S14_S15","P51G")
measure.list <- c("CLV10_MEUR","PD10_EUR")
url_var <- paste0(variable.list,collapse = "+")
url_meas <- paste0(measure.list,collapse = "+")
filter <- paste0("Q.",url_meas,".SCA.", url_var, ".EA19")
df <- rdb("Eurostat","namq_10_gdp",mask = filter)

d1 <-
  df %>%
  select(period, value,unit, na_item,series_name) %>%
  rename(var = na_item) %>%
  mutate( var = ifelse(var=="B1GQ"&unit=="CLV10_MEUR","gdp",
                       ifelse(var=="B1GQ","defgdp",
                              ifelse(var=="P31_S14_S15"&unit=="CLV10_MEUR","cons
o",
                                     ifelse(var=="P31_S14_S15","defconso",
                                            ifelse(var=="P51G"&unit=="CLV10_MEU
R","inves","definves")))))) %>%
  transmute(period,var,value,series_name)

# URL for wage series
df <- rdb(ids="Eurostat/namq_10_a10/Q.CP_MEUR.SCA.TOTAL.D1.EA19")

d2 <-
  df %>%
  select(period, unit, value, series_name) %>%
  rename(var=unit) %>%
  mutate(var="wage")

# URL for hours and employement
url_meas <- "THS_HW+THS_PER"
filter <- paste0("Q.",url_meas,".TOTAL.SCA.EMP_DC.EA19")
df <- rdb("Eurostat","namq_10_a10_e",mask = filter)

d3 <-
  df %>%
  select(period, unit, value, series_name) %>%
  rename(var= unit) %>%
  mutate(var=ifelse(var=="THS_HW","hours","employ")) %>%
  transmute(period,var,value, series_name)

# URL for quarterly 3-month rates
df <- rdb(ids="Eurostat/irt_st_q/Q.IRT_M3.EA")

d4 <-
  df %>%
  select(period, geo, value, series_name) %>%
  rename(var= geo) %>%
  mutate(var= "shortrate")

# URL for quarterly population series
df <- rdb(ids="Eurostat/lfsq_pganws/Q.THS.T.TOTAL.Y15-64.POP.EA19")

d5 <-
  df %>%
  select(period, geo, value, series_name) %>%
  rename(var= geo) %>%
  mutate(var= "pop") %>%
  filter(period >= "2005-01-01")
```

```
recent_data <- bind_rows(d1,d2,d3,d4,d5)
```

We can check the last date available for each variable.

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

var	maxdate
pop	2020-10-01
conso	2021-01-01
defconso	2021-01-01
defgdp	2021-01-01
definves	2021-01-01
employ	2021-01-01
gdp	2021-01-01
hours	2021-01-01
inves	2021-01-01
wage	2021-01-01
shortrate	2021-04-01

```
minmaxDate <- min(maxDate$maxdate)
recent_data %<>% filter(period <= minmaxDate)
```

Then we filter the recent database until 2020 Q4.

Final database

Now we can create the final database and chain the 11 historical series on the 11 recent series. To chain those series, we keep unchanged recent data from Eurostat and rebase the historical data.

We can check the first date available for each variable in the recent database.

```
minDate <-
  recent_data %>%
  group_by(var) %>%
  summarize(maxdate=min(period)) %>%
  arrange(maxdate)
kable(minDate)
```

var	maxdate
shortrate	1990-01-01
conso	1995-01-01
defconso	1995-01-01
defgdp	1995-01-01
definves	1995-01-01
employ	1995-01-01
gdp	1995-01-01
hours	1995-01-01
inves	1995-01-01
wage	1995-01-01
pop	2005-01-01

All the variables except the population (GDP, consumption, investment, their deflators, interest rates, hours, wage and employment) are chained at 1999Q1, official date of the creation of the Euro area.

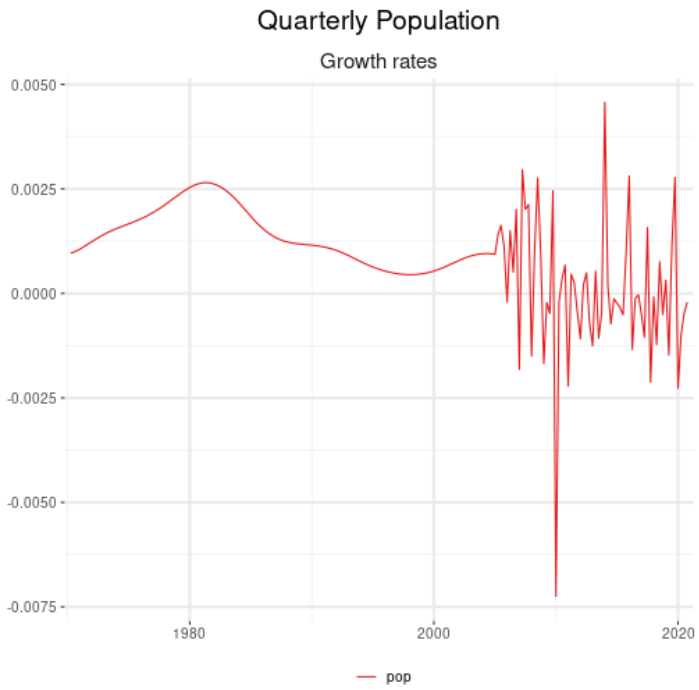
```
vars <- c("gdp","conso","inves","defgdp","defconso","definves","shortrate", "hour
s", "wage", "employ")
new_df <-
  recent_data %>%
  filter(var %in% vars)
old_df <-
  awm %>%
  filter(var %in% vars) %>%
  bind_rows(hours)
df1 <- chain(basis = new_df,
             to_rebase = old_df,
             date_chain = "1999-01-01")
```

Population special case

Chain and smooth the population series

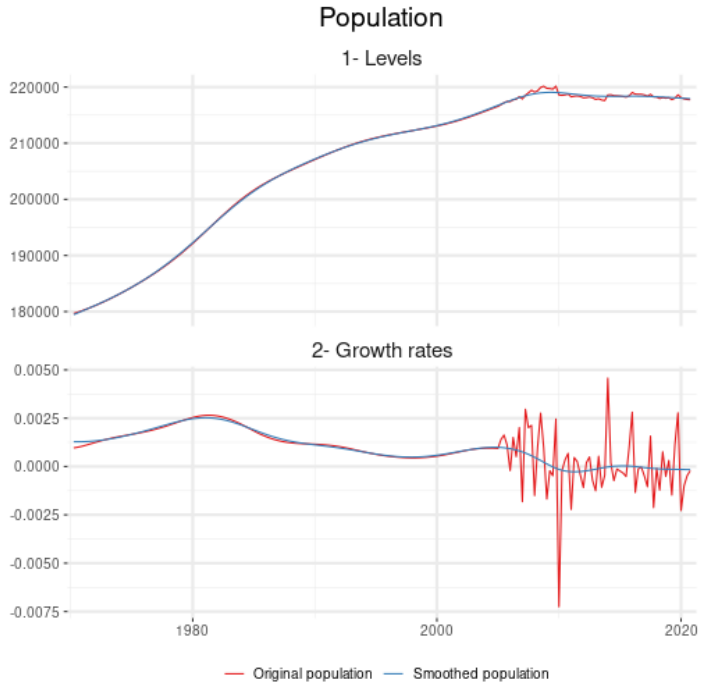
Population is a special case because we need to chain recent data with the historical series in 2005 (beginning of the population quarterly series) and we also need to make sure the series is as smooth as possible for normalization. First we chain the two series.

```
recent_pop_q <- filter(recent_data, var == "pop")

minDatePopQ <- min(recent_pop_q$period)

pop <- chain(basis = recent_pop_q,
             to_rebase= pop,
             date_chain=minDatePopQ)

plot_pop <- pop %>%
  mutate(value=log(value)-log(lag(value))) %>%
  data.frame(ind2="Growth rates") %>%
  filter(period>="1970-04-01")

ggplot(plot_pop, aes(period, value, colour = var)) +
  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()) +
  ggtitle("Quarterly Population")
```



The last years of the series exhibits a high level of volatility because they were not interpolated via a Kalman filter, we thus apply a Hodrick-Prescott filter to the series.

```
popts <- ts(pop$value,start=c(1970,1),f=4)
smoothed_popts <- hpfilter(popts, freq=1600)$trend

pop_StructTS <-
  pop %>%
  mutate(value=as.numeric(smoothed_popts),
         var="Smoothed population")
plot_pop <-
  pop %>%
  mutate(var="Original population")

pop_filtered <- bind_rows(plot_pop, pop_StructTS)

pop_filtered_levgr <- pop_filtered %>%
  mutate(value=log(value)-log(lag(value))) %>%
  data.frame(ind2="2- Growth rates") %>%
  bind_rows(data.frame(pop_filtered,ind2="1- Levels")) %>%
  filter(period>="1970-04-01")

ggplot(pop_filtered_levgr,aes(period,value,colour=var))+
  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()) +
  ggtitle("Population")
```



We retain the smoothed serie with the Hodrick-Prescott filter.

```
pop <- pop_StructTS %>%
  mutate(var = "pop")
```

So we can produce the final update of the Smets and Wouters (2003) database.

```
final_df <- bind_rows(df1, pop)

plot_df <- final_df
listVar <- list("Real GDP [1]" = "gdp",
                "Real consumption [2]" = "conso",
                "Real investment [3]" = "inves",
                "GDP deflator [4]" = "defgdp",
                "Consumption deflator [5]" = "defconso",
                "Investment deflator [6]" = "definves",
                "Real wage [7]" = "wage",
                "Hours worked [8]"= "hours",
                "Employment [9]" = "employ",
                "Interest rate [10]" = "shortrate",
                "Population [11]" = "pop")

plot_df$var <- factor(plot_df$var)
levels(plot_df$var)<-listVar

ggplot(plot_df,aes(period,value))+
  geom_line(colour=blueObsMacro)+
  facet_wrap(~var,scales = "free_y",ncol = 3)+
  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=7))
```


```
##  [1] "Quarterly - Chain linked volumes (2010), million euro - Seasonally and ca
lendar adjusted data - Gross domestic product at market prices - Euro area - 19 co
untries  (from 2015)"
##  [2] "Quarterly - Chain linked volumes (2010), million euro - Seasonally and ca
lendar adjusted data - Household and NPISH final consumption expenditure - Euro ar
ea - 19 countries  (from 2015)"
##  [3] "Quarterly - Chain linked volumes (2010), million euro - Seasonally and ca
lendar adjusted data - Gross fixed capital formation - Euro area - 19 countries
(from 2015)"
##  [4] "Quarterly - Price index (implicit deflator), 2010=100, euro - Seasonally
and calendar adjusted data - Gross domestic product at market prices - Euro area -
19 countries  (from 2015)"
##  [5] "Quarterly - Price index (implicit deflator), 2010=100, euro - Seasonally
and calendar adjusted data - Household and NPISH final consumption expenditure - E
uro area - 19 countries  (from 2015)"
##  [6] "Quarterly - Price index (implicit deflator), 2010=100, euro - Seasonally
and calendar adjusted data - Gross fixed capital formation - Euro area - 19 countr
ies  (from 2015)"
##  [7] "Quarterly - Current prices, million euro - Seasonally and calendar adjust
ed data - Total - all NACE activities - Compensation of employees - Euro area - 19
countries  (from 2015)"
##  [8] "Quarterly - Thousand hours worked - Total - all NACE activities - Seasona
lly and calendar adjusted data - Total employment domestic concept - Euro area - 1
9 countries  (from 2015)"
##  [9] "Quarterly - Thousand persons - Total - all NACE activities - Seasonally a
nd calendar adjusted data - Total employment domestic concept - Euro area - 19 cou
ntries  (from 2015)"
## [10] "Quarterly - 3-month rate - Euro area (EA11-1999, EA12-2001, EA13-2007, EA
15-2008, EA16-2009, EA17-2011, EA18-2014, EA19-2015)"
## [11] "Quarterly - Thousand - Total - Total - From 15 to 64 years - Population -
Euro area - 19 countries  (from 2015)"
```

You can download the 11 series directly here (http://shiny.cepremap.fr/data/EA_SW_rawdata.csv).

```
EA_SW_rawdata <-
  final_df %>%
  spread(key = var, value= value)

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

You can also download ready-to-use (normalized) data for the estimation here (http://shiny.cepremap.fr/data/EA_SW_data.csv).

```
EA_SW_data <-
  final_df %>%
  mutate(period=gsub(" ","",as.yearqtr(period))) %>%
  spread(key = var, value = value) %>%
  transmute(period = period,
            gdp_rpc=1e+6*gdp/(pop*1000),
            conso_rpc=1e+6*conso/(pop*1000),
            inves_rpc=1e+6*inves/(pop*1000),
            defgdp=defgdp,
            wage_rph=1e+6*wage/defgdp/(hours*1000),
            hours_pc=1000*hours/(pop*1000),
            pinves_defl=definves/defgdp,
            pconso_defl=defconso/defgdp,
            shortrate=shortrate/100,
            employ=1000*employ/(pop*1000))

EA_SW_data %>%
  na.omit() %>%
  write.csv("EA_SW_data.csv", row.names=FALSE)
```

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) {

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

  valref <- basis %>%
    filter(period == date_chain) %>%
    transmute(var, value_ref = value)

  res <- to_rebase %>%
    filter(period <= date_chain) %>%
    arrange(desc(period)) %>%
    group_by(var) %>%
    mutate(growth_rate = c(1, value[-1]/lag(value)[-1])) %>%
    full_join(valref, by = "var") %>%
    group_by(var) %>%
    transmute(period, value = cumprod(growth_rate)*value_ref)%>%
    ungroup() %>%
    bind_rows(filter(basis, period > date_chain)) %>%
    arrange(period)

  return(res)

}
```

Bibliography

G. Fagan, J. Henry, and R. Mestre. An area-wide model (awm) for the euro area. *ECB Working Paper Series*, 2001. ↩ 1 2

F. Smets and R. Wouters. An estimated dynamic stochastic general equilibrium model of the euro area. *Journal of the European Economic Association*, 2003. ↩ 1 2 3 4 5

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2016. URL: https://www.R-project.org (https://www.R-project.org). ↩

RStudio Team. *RStudio: Integrated Development Environment for R*. RStudio, Inc., Boston, MA, 2016. URL: http://www.rstudio.com/ (http://www.rstudio.com/). ↩

RELATED POSTS

Macroeconomic data for France, Germany, Italy, Spain & the Euro Area (/article/2021-02/five-countries-data/)

Description step by step to build automatic update of a macroeconomic database for France, Germany, Italy, Spain & the Euro Area.

▸ MORE (/ARTICLE/2021-02/FIVE-COUNTRIES-DATA/)

Automating update of an international database for the Euro Area (/article/2019-12/open-EA-data/)

Description step by step to build automatic update of an international database for the Euro Area.

▸ MORE (/ARTICLE/2019-12/OPEN-EA-DATA/)

Automating update of a fiscal database for the Euro Area (/article/2019-11/fipu-EA-data/)

Description step by step to build automatic update of a quarterly fiscal database for the Euro Area, similar to the Paredes, Pedregal...

▸ MORE (/ARTICLE/2019-11/FIPU-EA-DATA/)

Automating update of the Christiano, Motto and Rostagno (2014) database for the United States (/article/2016-06/cmr14-data/)

Description step by step to build automatic update of the Christiano, Motto and Rostagno (2014) database for the United States.

▸ MORE (/ARTICLE/2016-06/CMR14-DATA/)