Introduction to the convergEU package

Federico M. Stefanini - Update: Berta Mizsei

2022-04-22 - rel 1.1.0 Index:

Introduction

The *convergEU* R package is a set of S3 functions and data objects suited for the analysis of economic and social convergence of Member States (**MS**) within the European Union (**EU**). The analyses performed by the package are not suitable for making causal inferences but they do allow the user to gain a wealth of insight into how convergence in certain indicators has evolved throughout time.

This vignette is intended to be a gentle introduction to the analysis of convergence performed with *convergEU* suite of functions. The package allows the user to access data from Eurofound (local) and Eurostat (download) with little effort. Furthermore, it is possible create or import custom datasets.

Since a dataset created or processed in this package should take the form of a **tibble**, at least some familiarity with the **dplyr** package (online dplyr site) is convenient. To implement convergence analysis with the *covergEU* package, the data must be in a tidy format - i.e. a rectangular table with time periods as the first column and units (e.g. Member Sates) in subsequent columns. Imported or downloaded data may not be in this tidy format, therefore the user will have to reformat their data to fit the requirements of the package. There are some cases where further elements of the **tidyverse** (https://www.tidyverse.org/) are used. For a general introduction to the **tidyverse**, see "R for Data Science" (online R4DS) by Wickham and Grolemund.

The most common data processed with the package are indicators downloaded from EU repositories, which are often tied to policy targets. It is important to note that the desired policy target for an indicator may be higher values (e.g. GDP/capita) or lower values (e.g. NEET rate). These are referred to as **highBest** and **lowBest** in the package: some functions require that the user provide an argument as a string chosen between the two possibilities "highBest" and "lowBest".

Keep in mind that functions in the *convergEU* package calculate summaries over sets of EU countries (e.g. the EU27 or the Euro area). Therefore, the dataset used must contain values for all countries within the selected grouping, even if the user is only interested in just two or three countries.

The *convergEU* package produces results as a list with metainformation. This list is made up of three components: *$res, $msg, $err*. The first list component, *$res*, is the actual result, if computed. The second component, *$msg* is a message (possibly a warning) accompanying the computed result. The third component, *$err*, is an error message or a list of errors if a result is not computed.

The **R** packages used in this vignette are:

```
require(convergEU)
require(ggplot2)
require(dplyr)
require(tidyverse)
require(eurostat)
require(purrr)
require(tibble)
require(tidyr)
```

```
require(ggplot2)
require(formattable)
require(caTools)
```

Loading and preparing data

This vignette presents how to work with two types of data sources: data produced by Eurofound that are available without an active internet connection, and Eurostat data that can be downloaded with an active internet connection.

Locally accessible datasets: Eurofound data

Some datasets are accessible from the *convergEU* package using the R function *data()*. The code below download a dataset with employment rates for the EU Member States.

```
library(convergEU)
data("emp_20_64_MS",package = "convergEU")
head(emp_20_64_MS)
```

The Eurofound datasets EWCS (Employment and Working Conditions Survey) and EQLS (European Quality of Life Survey) are locally available within *convergEU*, see:

```
data(package = "convergEU")
```

The object `dbEUF2018meta` contains a description of data that is available from within the *convergEU* package, i.e. it does not need to be downloaded from the Eurofound website.

```
print(dbEUF2018meta, n=200,width=200)
#> # A tibble: 13 x 10
#>    DIMENSION          SUBDIMENSION
#>    <chr>              <chr>
#>  1 Quality of life    Life satisfaction
#>  2 Quality of life    Health
#>  3 Quality of life    Health
#>  4 Quality of life    Quality of society
#>  5 Quality of life    Quality of society
#>  6 Quality of life    Quality of society
#>  7 Quality of life    Quality of society
#>  8 Quality of life    Quality of society
#>  9 Working conditions Working conditions
#> 10 Working conditions Working conditions
#> 11 Working conditions Working conditions
#> 12 Working conditions Working conditions
#> 13 Working conditions Working conditions
#>    INDICATOR                                      Code_in_database Official_code
#>    <chr>                                          <chr>            <chr>
#>  1 Mean life satisfaction                         lifesatisf       y16_q4
#>  2 Mean health status                             health           y16_q48
#>  3 Percentage of people with good or very good h~ goodhealth_p     y16_q48
#>  4 Mean level of trust in local government        trustlocal       y16_q35f
```

```
#>  5 Level of involvement in volunteering           volunt          y16_q29a
#>  6 Percentage of people involved in volunteering  volunt_p        y16_q29a
#>  7 Hours per week spent in informal care          caring_h        y16_q43a
#>  8 Social Exclusion Index                         socialexc_i     y16_socexind~
#>  9 JQI_Skills and discretion index                JQIskill_i      wq_slim -  J~
#> 10 JQI_Physical environment index                 JQIenviron_i    envsec_slim ~
#> 11 JQI_Intensity index                            JQIintensity_i  intens_slim ~
#> 12 JQI_Working time quality index                 JQItime_i       wlb_slim - J~
#> 13 Exposition to discrimination                   exposdiscr_p    disc_d -  Ha~
#>    Unit  Source_organisation Source_reference Disaggregation Bookmark_URL
#>    <chr> <chr>               <chr>            <chr>          <chr>
#>  1 --    Eurofound           EQLS             sex            https://www.eurofo~
#>  2 --    Eurofound           EQLS             sex            https://www.eurofo~
#>  3 %     Eurofound           EQLS             sex            https://www.eurofo~
#>  4 --    Eurofound           EQLS             sex            https://www.eurofo~
#>  5 --    Eurofound           EQLS             sex            https://www.eurofo~
#>  6 %     Eurofound           EQLS             sex            https://www.eurofo~
#>  7 hours Eurofound           EQLS             sex            https://www.eurofo~
#>  8 index Eurofound           EQLS             sex            https://www.eurofo~
#>  9 index Eurofound           EWCS             sex            https://www.eurofo~
#> 10 index Eurofound           EWCS             sex            https://www.eurofo~
#> 11 index Eurofound           EWCS             sex            https://www.eurofo~
#> 12 index Eurofound           EWCS             sex            https://www.eurofo~
#> 13 %     Eurofound           EWCS             sex            https://www.eurofo~
```

The raw local Eurofound database is accessed as follows:

```
require(convergEU)
data(dbEurofound)
head(dbEurofound)
#> # A tibble: 6 x 17
#>    time geo   geo_label sex     lifesatisf health goodhealth_p trustlocal volunt
#>   <dbl> <chr> <chr>     <chr>        <dbl>  <dbl>        <dbl>      <dbl>  <dbl>
#> 1  1960 AD    Andorra   Females         NA     NA           NA         NA     NA
#> 2  1960 AD    Andorra   Males           NA     NA           NA         NA     NA
#> 3  1960 AD    Andorra   Total           NA     NA           NA         NA     NA
#> 4  1960 AL    Albania   Females         NA     NA           NA         NA     NA
#> 5  1960 AL    Albania   Males           NA     NA           NA         NA     NA
#> 6  1960 AL    Albania   Total           NA     NA           NA         NA     NA
#> # ... with 8 more variables: volunt_p <dbl>, caring_h <dbl>, socialexc_i <dbl>,
#> #   JQIskill_i <dbl>, JQIenviron_i <dbl>, JQIintensity_i <dbl>,
#> #   JQItime_i <dbl>, exposdiscr_p <dbl>
```

. . . where variable names are:

```
names(dbEurofound)
#>  [1] "time"           "geo"            "geo_label"      "sex"
#>  [5] "lifesatisf"     "health"         "goodhealth_p"   "trustlocal"
#>  [9] "volunt"         "volunt_p"       "caring_h"       "socialexc_i"
#> [13] "JQIskill_i"     "JQIenviron_i"   "JQIintensity_i" "JQItime_i"
#> [17] "exposdiscr_p"
```

and the time ranges are:

```
c(min(dbEurofound$time), max(dbEurofound$time))
#> [1] 1960 2017
```

Remember that the databases will likely have missing data.

NOTE: Eurofound data are statically stored within *convergeEU* package. To have the most recent version of Eurofound data, please update the package.

The Eurofound dataset exploited during the development of this package: *emp_20_64_MS* is the tidy version of the dataset *emp_20_64*. It contains information on employment rates for 20 to 64 years old and can always be used for analysis without any need for further processing.

```
help(emp_20_64_MS)
```

All other locally accessible Eurofound indicators can be extracted from the Eurofound database as the first step of an analysis: this is the data preparation step. The user needs only to choose a time interval, an indicator and a set of countries (**MS**, Member States). For example:

```
convergEU_glb()$EU12
#> $dates
#> [1] "01-11-1993" "31/12/1994"
#>
#> $memberStates
#> # A tibble: 12 x 2
#>    MS             codeMS
#>    <chr>          <chr>
#>  1 Belgium        BE
#>  2 Denmark        DK
#>  3 France         FR
#>  4 Germany        DE
#>  5 Greece         EL
#>  6 Ireland        IE
#>  7 Italy          IT
#>  8 Luxembourg     LU
#>  9 Netherlands    NL
#> 10 Portugal       PT
#> 11 Spain          ES
#> 12 United-Kingdom UK
```

among those available:

```
names(convergEU_glb())[c(3:8)]
#> [1] "EA19"      "EU12"      "EU15"      "EU25"      "EU27_2007" "EU27_2019"
```

Remember that “EA” , “EA19” and “Euro area” are synonyms.

```
convergEU_glb()$EA
#> $dates
#> [1] NA NA
#>
#> $memberStates
#> # A tibble: 19 x 2
#>    MS          codeMS
```

```
#>    <chr>       <chr>
#>  1 Austria     AT
#>  2 Belgium     BE
#>  3 Cyprus      CY
#>  4 Estonia     EE
#>  5 Finland     FI
#>  6 France      FR
#>  7 Germany     DE
#>  8 Greece      EL
#>  9 Ireland     IE
#> 10 Italy       IT
#> 11 Latvia      LV
#> 12 Lithuania   LT
#> 13 Luxembourg  LU
#> 14 Malta       MT
#> 15 Netherlands NL
#> 16 Portugal    PT
#> 17 Slovakia    SK
#> 18 Slovenia    SI
#> 19 Spain       ES
```

As an example, here is how one would select the "lifesatisf" indicator taken from the column "Code_in_database" within the object *dbEUF2018meta* that contains the meta information:

```
head(dbEUF2018meta)
#> # A tibble: 6 x 10
#>   DIMENSION       SUBDIMENSION    INDICATOR Code_in_database Official_code Unit
#>   <chr>           <chr>           <chr>     <chr>            <chr>         <chr>
#> 1 Quality of life Life satisfact~ Mean lif~ lifesatisf       y16_q4        --
#> 2 Quality of life Health          Mean hea~ health           y16_q48       --
#> 3 Quality of life Health          Percenta~ goodhealth_p     y16_q48       %
#> 4 Quality of life Quality of soc~ Mean lev~ trustlocal       y16_q35f      --
#> 5 Quality of life Quality of soc~ Level of~ volunt           y16_q29a      --
#> 6 Quality of life Quality of soc~ Percenta~ volunt_p         y16_q29a      %
#> # ... with 4 more variables: Source_organisation <chr>, Source_reference <chr>,
#> #   Disaggregation <chr>, Bookmark_URL <chr>
myTB <- extract_indicator_EUF(
    indicator_code = "lifesatisf", #Code_in_database
    fromTime=2003,
    toTime=2016,
    gender= c("Total","Females","Males")[2],
    countries= convergEU_glb()$EU12$memberStates$codeMS
    )

myTB
#> $res
#> # A tibble: 4 x 14
#>    time sex       BE    DE    DK    EL    ES    FR    IE    IT    LU    NL    PT
#>   <dbl> <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1  2003 Femal~  7.38  7.48  8.37  6.75  7.43  6.97  7.89  7.12  7.77  7.54  5.86
#> 2  2007 Femal~  7.49  7.17  8.51  6.55  7.14  7.34  7.66  6.54  7.87  7.90  6.09
#> 3  2011 Femal~  7.47  7.27  8.30  6.13  7.51  7.17  7.41  6.86  7.72  7.74  6.72
#> 4  2016 Femal~  7.27  7.28  8.33  5.30  6.97  7.24  7.66  6.56  7.96  7.74  6.79
```

```
#> # ... with 1 more variable: UK <dbl>
#>
#> $msg
#> NULL
#>
#> $err
#> NULL
```

which results in three components: a dataset ready for further analyses ("$res"); possible messages for the user ("$msg") and, if an error occurs, a string (i.e. text) containing error messages ("$err").

Downloadable data: Eurostat repository

Eurostat data can be downloaded directly from the Eurostat via an active internet connection.

The heterogeneity in the structure of different indicators normalized into the database requires some attentions. Sometimes, a list of covariates for the indicator is available (gender, age, poverty status, etc.) and these values must be set to produce a tidy dataset (i.e. a table containing only time by countries).

First, raw data may be downloaded using the option *rawDump=T*:

```
ddTB1 <- download_indicator_EUS(
      indicator_code= convergEU_glb()$metaEUStat$selectorUser[1],
      fromTime = 2005,
      toTime = 2015,
      gender= c(NA,"T","F","M")[2],#c("Total","Females","Males")
      countries =  convergEU_glb()$EU28$memberStates$codeMS,
      rawDump=T )
ddTB1
```

```
#> # A tibble: 346,260 x 7
#>    unit  sex   age    isced11 geo    time values
#>    <fct> <fct> <fct>  <fct>   <fct> <dbl>  <dbl>
#>  1 PC    F     Y15-19 ED0-2   AT     2018   28.5
#>  2 PC    F     Y15-19 ED0-2   BE     2018    9.7
#>  3 PC    F     Y15-19 ED0-2   BG     2018   NA
#>  4 PC    F     Y15-19 ED0-2   CH     2018   50.5
#>  5 PC    F     Y15-19 ED0-2   CY     2018   NA
#>  6 PC    F     Y15-19 ED0-2   CZ     2018    2.5
#>  7 PC    F     Y15-19 ED0-2   DE     2018   21.7
#>  8 PC    F     Y15-19 ED0-2   DK     2018   53.9
#>  9 PC    F     Y15-19 ED0-2   EA19   2018   14
#> 10 PC    F     Y15-19 ED0-2   EE     2018   14.6
#> # ... with 346,250 more rows
```

This results in a dataset which is not in the tidy format.

Note that *unit* and *isced11* are auxilary variables specific to this indicator and that some filtering must be performed to obtain a tidy dataset years by countries. The argument *rawDump=F* filters and reshapes the bulk data as shown below, where *convergEU_glb()$EU28$memberStates$codeMS* is a vector of strings for the considered countries, *convergEU_glb()$metaEUStat$selectorUser[1]* contains the name of the indicator and *ageInterv* can be used to specify an age interval for a given indicator:

```r
ddTB2 <- download_indicator_EUS(
      indicator_code= convergEU_glb()$metaEUStat$selectorUser[1],
      fromTime = 2005,
      toTime = 2015,
      gender= c(NA,"T","F","M")[1],#c("Total","Females","Males")
      ageInterv = NA,
      countries =  convergEU_glb()$EU28$memberStates$codeMS,
      rawDump=F,
      uniqueIdentif = 1)

convergEU_glb()$metaEUStat$selectorUser[1]
ddTB2
```

```
#> [1] "lfsa_argaed"
#> $res
#> # A tibble: 11 x 31
#>    age   sex    time    BE    DK    FR    DE    EL    IE    IT    LU    NL    PT
#>    <fct> <fct> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
#>  1 Y15-~ T      2005   7.2  57.9  11.6  27.5   7    19.1  10.4   6.6  54.1  18
#>  2 Y15-~ T      2006   7.1  61.7  11.9  28.5   7.6  19.1   9     7.7  54.4  17.8
#>  3 Y15-~ T      2007   7    61.2  12.2  29.9   6.7  32.9   8.2   8.1  57.9  16.6
#>  4 Y15-~ T      2008   6.8  63.9  11.3  30.2   6.2  27.8   8.1   8.5  59.1  15.4
#>  5 Y15-~ T      2009   7    61.8  12    29     6    19.7   6.6  10.8  57.9  13.1
#>  6 Y15-~ T      2010   6.8  58.1  11.4  27.9   5.3  16.2   5.9   7.9  56.9  11.3
#>  7 Y15-~ T      2011   7.2  57.4  10.5  42.1   4.8  13.7   5.2   7.3  56.6  11.7
#>  8 Y15-~ T      2012   6.3  53.9  10    39.4   4.4  13.2   5.6   7.6  57.9  10.1
#>  9 Y15-~ T      2013   5.5  50.6   9.6  40     3.8  14.2   4.5  10.8  57.8   9
#> 10 Y15-~ T      2014   5.4  50     8.6  25.2   3    13.1   4    10.5  56.3   7.8
#> 11 Y15-~ T      2015   4.6  50.1   7.6  24.7   2.3  13.1   4.1  16.9  58.3   6.8
#> # ... with 18 more variables: ES <dbl>, UK <dbl>, AT <dbl>, FI <dbl>, SE <dbl>,
#> #   CY <dbl>, CZ <dbl>, EE <dbl>, HU <dbl>, LV <dbl>, LT <dbl>, MT <dbl>,
#> #   PL <dbl>, SK <dbl>, SI <dbl>, BG <dbl>, RO <dbl>, HR <dbl>
#>
#> $msg
#> $msg$gender
#> [1] "Gender automatically set to 'T'."
#>
#> $msg$Age
#> [1] "Age automatically set."
#>
#> $msg$Further_Conditioning
#> $msg$Further_Conditioning$current
#> [1] "Selected uniqueIdentif = 1 -> unit: PC; isced11: ED0-2"
#>
#> $msg$Further_Conditioning$available_seleTagLs
#>   uniqueIdentif                     tags
#> 1             1 unit: PC; isced11: ED0-2
#> 2             2 unit: PC; isced11: ED3_4
#> 3             3 unit: PC; isced11: ED5-8
#> 4             4   unit: PC; isced11: NRP
#> 5             5 unit: PC; isced11: TOTAL
#>
#>
```

```
#> $msg$Conditioning
#> $msg$Conditioning$indicator_code
#> [1] "lfsa_argaed"
#>
#> $msg$Conditioning$ageInterv
#> [1] Y15-19
#> 30 Levels: Y15-19 Y15-24 Y15-39 Y15-59 Y15-64 Y15-74 Y20-24 Y20-64 ... Y70-74
#>
#> $msg$Conditioning$gender
#> [1] "T"
#>
#>
#>
#> $err
#> NULL
```

The result is a list with the following components:

- $res contains the selected data as a tidy tibble (dataset);
- msggender is a component describing results;
- msggender states that gender was not selected (thus it was automatically set);
- msgAge states that age class was automatically set;
- msgFurther_Conditioning contains further variables to set in order to obtain a tidy dataset; in particular msgFurther_Conditioning$current states the value of conditioning specific variables taken from the list of possible ones msgFurther_Conditioning$available_seleTagLs;
- msgConditioning contains common conditioning variables, such as *ageInterv* and *gender*.

It is therefore possible to call the same function several times and specify the argument *uniqueIdentif* as an integer among those in the first column left of *msgFurther_Conditioning$available_seleTagLs* to obtain the same indicator under different scales and contexts. For example, here is how one would apply the fifth conditioning option that selects for males in the age interval "Y15-64":

```
ddTB3 <- download_indicator_EUS(
      indicator_code= convergEU_glb()$metaEUStat$selectorUser[1],
      fromTime = 2005,
      toTime = 2015,
      gender= "M",
      ageInterv = "Y15-64",
      countries =  convergEU_glb()$EU28$memberStates$codeMS,
      rawDump=F,
      uniqueIdentif = 5)
ddTB3
```

```
#> $res
#> # A tibble: 11 x 31
#>    age   sex    time    BE    DK    FR    DE    EL    IE    IT    LU    NL    PT
#>    <fct> <fct> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
#>  1 Y15-~ M      2005  73.9  83.6  75.2  80.6  78.4  80.6  74.6  76    82.3  78.9
#>  2 Y15-~ M      2006  73.4  84.1  74.9  81.3  78.5  81.7  74.5  75.3  82.2  79.2
```

```
#>  3 Y15-~ M      2007  73.6  83.7  74.7  81.7  78.4  84.7  74.3  75    83.2  79.2
#>  4 Y15-~ M      2008  73.3  84.3  74.7  82    78.4  83.5  74.3  74.7  84.2  79.2
#>  5 Y15-~ M      2009  72.8  83.6  75    82.2  78.5  80.7  73.5  76.6  84.1  78.2
#>  6 Y15-~ M      2010  73.4  82.6  74.9  82.4  78.3  78.7  73.1  76    83.3  77.8
#>  7 Y15-~ M      2011  72.3  82.3  74.6  82.7  77.2  78    72.8  75    83.2  78
#>  8 Y15-~ M      2012  72.5  81.4  75.3  82.6  76.9  77.8  73.7  75.9  83.9  77.3
#>  9 Y15-~ M      2013  72.7  80.6  75.5  82.6  76.9  78.3  73.3  76.3  84.3  76.5
#> 10 Y15-~ M      2014  72.4  81.1  75.1  82.5  76    78.6  73.6  77.2  84.2  76.7
#> 11 Y15-~ M      2015  72.2  81.6  75.3  82.1  75.9  79    74.1  76    84.6  76.7
#> # ... with 18 more variables: ES <dbl>, UK <dbl>, AT <dbl>, FI <dbl>, SE <dbl>,
#> #   CY <dbl>, CZ <dbl>, EE <dbl>, HU <dbl>, LV <dbl>, LT <dbl>, MT <dbl>,
#> #   PL <dbl>, SK <dbl>, SI <dbl>, BG <dbl>, RO <dbl>, HR <dbl>
#>
#> $msg
#> $msg$Further_Conditioning
#> $msg$Further_Conditioning$current
#> [1] "Selected uniqueIdentif = 5 -> unit: PC; isced11: TOTAL"
#>
#> $msg$Further_Conditioning$available_seleTagLs
#>   uniqueIdentif                     tags
#> 1             1 unit: PC; isced11: ED0-2
#> 2             2 unit: PC; isced11: ED3_4
#> 3             3 unit: PC; isced11: ED5-8
#> 4             4   unit: PC; isced11: NRP
#> 5             5 unit: PC; isced11: TOTAL
#>
#>
#> $msg$Conditioning
#> $msg$Conditioning$indicator_code
#> [1] "lfsa_argaed"
#>
#> $msg$Conditioning$ageInterv
#> [1] "Y15-64"
#>
#> $msg$Conditioning$gender
#> [1] "M"
#>
#>
#>
#> $err
#> NULL
```

Data preparation: from data structure to imputation of missing values

The analysis of convergence is performed on clean and imputed data that takes the shape of a tidy dataset with time by countries. The first step after downloading data is the description of the main features of such a dataset.

An illustrative example follows with the indicator *"JQIintensity_i"*:

```
# print(dbEUF2018meta[11,],n=20,width=100)
t(dbEUF2018meta[11,])
#>                     [,1]
#> DIMENSION           "Working conditions"
#> SUBDIMENSION        "Working conditions"
#> INDICATOR           "JQI_Intensity index"
#> Code_in_database    "JQIintensity_i"
#> Official_code       "intens_slim - JQI SLIM Intensity index"
#> Unit                "index"
#> Source_organisation "Eurofound"
#> Source_reference    "EWCS"
#> Disaggregation      "sex"
#> Bookmark_URL        "https://www.eurofound.europa.eu/surveys/about-eurofound-surveys/data-availabili
```

First the raw dataset is downloaded:

```
ddTB4 <- extract_indicator_EUF(
    indicator_code = "JQIintensity_i", #Code_in_database
    fromTime= 1965,
    toTime=2016,
    gender= c("Total","Females","Males")[1],
    countries= convergEU_glb()$EU28$memberStates$codeMS
    )
print(ddTB4$res,n=35,width=250)
```

```
#> # A tibble: 5 x 30
#>    time sex      AT    BE    BG    CY    CZ    DE    DK    EE    EL    ES    FI
#>   <dbl> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1  1995 Total  48.8  33.2  NA    NA    NA    40.8  39.0  NA    40.9  34.2  47.1
#> 2  2000 Total  42.9  37.3  43.7  53.4  42.6  40.9  37.6  38.2  43.5  36.2  46.7
#> 3  2005 Total  47.6  42.8  33.8  50.7  45.8  46.9  47.9  41.7  50.5  41.2  49.6
#> 4  2010 Total  42.1  40.2  31.2  52.5  41.9  44.9  39.1  41.9  48.6  38.0  45.9
#> 5  2015 Total  42.4  41.5  34.6  57.2  36.7  40.2  45.0  38.7  49.3  46.5  41.1
#>      FR    HR    HU    IE    IT    LT    LU    LV    MT    NL    PL    PT    RO
#>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1  38.4  NA    NA    39.0  34.1  NA    31.4  NA    NA    41.8  NA    36.2  NA
#> 2  39.5  NA    42.1  42.2  39.7  29.9  37.6  32.8  50.0  41.3  40.9  31.8  47.8
#> 3  40.5  31.6  47.2  36.9  41.9  37.3  40.6  34.3  48.4  40.3  35.7  40.1  45.9
#> 4  43.0  39.5  48.7  47.0  40.8  33.2  40.8  31.9  44.0  38.5  31.4  31.6  43.3
#> 5  42.7  38.4  44.7  42.8  38.1  37.8  42.4  31.5  44.8  38.7  35.0  36.8  54.2
#>      SE    SI    SK    UK
#>   <dbl> <dbl> <dbl> <dbl>
#> 1  43.3  NA    NA    47.3
#> 2  47.9  29.5  41.6  45.0
#> 3  48.1  49.2  39.6  42.5
#> 4  45.9  48.2  37.6  43.6
#> 5  46.1  43.0  35.9  45.9
```

. . . where error messages are not shown (the $err list component is empty).

The print output tells us that missing values are present. First, it is convenient to assign a meaningful name to the downloaded data:

```
JQIinte <- ddTB4$res
```

More features may be investigated as usual with common R functions, like the dimension of the dataset:

```
dim(JQIinte)
#> [1]  5 30
```

. . . that is, 5 rows and 30 columns. Here's how to check variable names:

```
names(JQIinte)
#>  [1] "time" "sex"  "AT"   "BE"   "BG"   "CY"   "CZ"   "DE"   "DK"   "EE"
#> [11] "EL"   "ES"   "FI"   "FR"   "HR"   "HU"   "IE"   "IT"   "LT"   "LU"
#> [21] "LV"   "MT"   "NL"   "PL"   "PT"   "RO"   "SE"   "SI"   "SK"   "UK"
```

A dataset can't have qualitative variables, vectors of strings, or missing values for computing convergence measures. A time variable should also be present, and if the name is not "time", than it must be passed during function calls as an argument to have proper data processing. The *check_data()* function checks for unsuited features that must be solved before starting the analysis. The object returned states if the dataset is ready for calculations, and if it is not, the error component states why checking failed: * "Error: one or more missing values in the dataframe."
* "Error: qualitative variables in the dataframe."
* "Error: string variables in the dataframe."
* "Error: timeName variable absent."
* "Error: time variable is not ordered."

For example, with the *JQIinte* data we have:

```
check_data(JQIinte,timeName="time")
#> $res
#> NULL
#>
#> $msg
#> NULL
#>
#> $err
#> [1] "Error: one or more missing values in the dataframe."
```

Missing values are present, thus imputation is required. This can be done by using the *impute_dataset* function:

```
JQIinteImp <- impute_dataset(JQIinte, timeName = "time",
                             countries=convergEU_glb()$EU28$memberStates$codeMS,
                             tailMiss = c("cut", "constant")[2],
                             headMiss = c("cut", "constant")[2])$res
print(JQIinteImp,n=35,width=250)
#> # A tibble: 5 x 30
#>    time sex      AT    BE    BG    CY    CZ    DE    DK    EE    EL    ES    FI
#>   <dbl> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1  1995 Total  48.8  33.2  43.7  53.4  42.6  40.8  39.0  38.2  40.9  34.2  47.1
#> 2  2000 Total  42.9  37.3  43.7  53.4  42.6  40.9  37.6  38.2  43.5  36.2  46.7
#> 3  2005 Total  47.6  42.8  33.8  50.7  45.8  46.9  47.9  41.7  50.5  41.2  49.6
#> 4  2010 Total  42.1  40.2  31.2  52.5  41.9  44.9  39.1  41.9  48.6  38.0  45.9
```

```
#> 5  2015 Total  42.4  41.5  34.6  57.2  36.7  40.2  45.0  38.7  49.3  46.5  41.1
#>      FR    HR    HU    IE    IT    LT    LU    LV    MT    NL    PL    PT    RO
#>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1  38.4  31.6  42.1  39.0  34.1  29.9  31.4  32.8  50.0  41.8  40.9  36.2  47.8
#> 2  39.5  31.6  42.1  42.2  39.7  29.9  37.6  32.8  50.0  41.3  40.9  31.8  47.8
#> 3  40.5  31.6  47.2  36.9  41.9  37.3  40.6  34.3  48.4  40.3  35.7  40.1  45.9
#> 4  43.0  39.5  48.7  47.0  40.8  33.2  40.8  31.9  44.0  38.5  31.4  31.6  43.3
#> 5  42.7  38.4  44.7  42.8  38.1  37.8  42.4  31.5  44.8  38.7  35.0  36.8  54.2
#>      SE    SI    SK    UK
#>   <dbl> <dbl> <dbl> <dbl>
#> 1  43.3  29.5  41.6  47.3
#> 2  47.9  29.5  41.6  45.0
#> 3  48.1  49.2  39.6  42.5
#> 4  45.9  48.2  37.6  43.6
#> 5  46.1  43.0  35.9  45.9
```

$res was added to the function call in order to use only the tibble containing years by countries. The imputation selected for the first (tail) and last (head) years is "constant", thus the first not missing value is propagated to missing years, but the alternative of cutting all years in which one or more missing are presents may be selected with the arguments:

```
tailMiss = "cut",headMiss = "cut"
```

The following code can be used to determine which years a country (here *HR*) is missing values for:

```
select(filter(JQIinte, is.na(HR)),time,HR)
#> # A tibble: 2 x 2
#>    time    HR
#>   <dbl> <dbl>
#> 1  1995    NA
#> 2  2000    NA
```

This code does the same thing by using pipes (and the *magrittr* package):

```
JQIinte %>%
  filter(is.na(HR)) %>%
  select(time,HR)
#> # A tibble: 2 x 2
#>    time    HR
#>   <dbl> <dbl>
#> 1  1995    NA
#> 2  2000    NA
```

The *check_data* function is called again but on imputed data:

```
check_data(JQIinteImp)
#> $res
#> NULL
#>
#> $msg
#> NULL
#>
#> $err
#> [1] "Error: string  variables in the dataframe."
```

. . . where the suspected string variable is *sex*:

```
JQIinteFin <- dplyr::select(JQIinteImp,-sex)
check_data(JQIinteFin)
#> $res
#> [1] TRUE
#>
#> $msg
#> NULL
#>
#> $err
#> NULL
```

. . . and thus *JQIinteFin* is the final object to start the data analysis.

The *tidyverse* functions *mutate, select, filter* are extremely useful for further selection and inspection, and can be used with or without the forward-pipe operator (%>%).

Note that before starting the analysis, the **number of digits may be selected**. For example, the above tibble can be rounded to integers by invoking the function *round()* where *digits = 0*:

```
JQIinteFin[, -1] <- round(select(JQIinteFin,- time), digits = 0)
JQIinteFin
#> # A tibble: 5 x 29
#>    time    AT    BE    BG    CY    CZ    DE    DK    EE    EL    ES    FI    FR
#>   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
#> 1  1995    49    33    44    53    43    41    39    38    41    34    47    38
#> 2  2000    43    37    44    53    43    41    38    38    43    36    47    40
#> 3  2005    48    43    34    51    46    47    48    42    50    41    50    41
#> 4  2010    42    40    31    52    42    45    39    42    49    38    46    43
#> 5  2015    42    41    35    57    37    40    45    39    49    46    41    43
#> # ... with 16 more variables: HR <dbl>, HU <dbl>, IE <dbl>, IT <dbl>, LT <dbl>,
#> #   LU <dbl>, LV <dbl>, MT <dbl>, NL <dbl>, PL <dbl>, PT <dbl>, RO <dbl>,
#> #   SE <dbl>, SI <dbl>, SK <dbl>, UK <dbl>
```

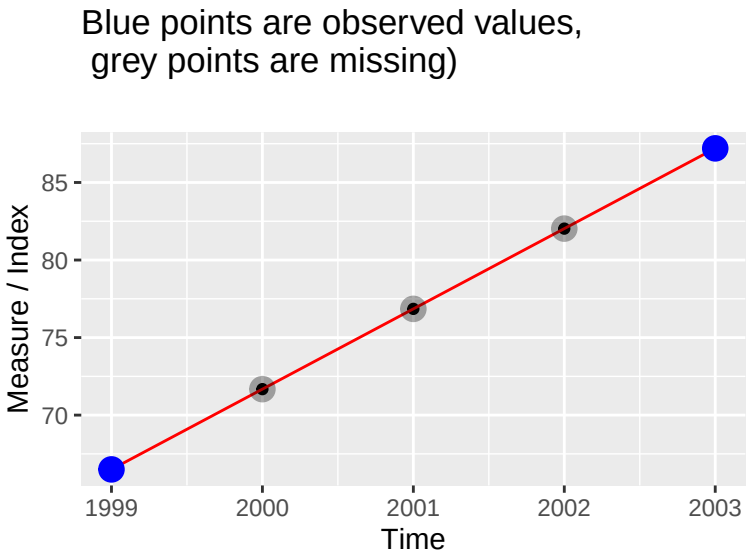
Imputing missing values using a straight line

The basic imputation method is deterministic, like in the average of two interval endpoints separated by just one year missing. If several missing values are present in a row, a linear change of an indicator is assumed over time between the two observed time points flanking a chunk of missing values.

```
intervalTime <-  c(1999,2000,2001,2002,2003)
intervalMeasure <- c( 66.5, NA,NA,NA,87.2)
currentData <- tibble(time= intervalTime, veval= intervalMeasure)
currentData
#> # A tibble: 5 x 2
#>    time veval
#>   <dbl> <dbl>
#> 1  1999  66.5
#> 2  2000  NA
#> 3  2001  NA
#> 4  2002  NA
#> 5  2003  87.2
```

```
resImputed <- impute_dataset(currentData,
                             countries = "veval",
                             timeName = "time",
                             tailMiss = c("cut", "constant")[2],
                             headMiss = c("cut", "constant")[2])
resImputed$res
#> # A tibble: 5 x 2
#>    time veval
#>   <dbl> <dbl>
#> 1  1999  66.5
#> 2  2000  71.7
#> 3  2001  76.9
#> 4  2002  82.0
#> 5  2003  87.2
```

In the figure below, grey points are imputed using observed values represented by solid blue points:



It is important to note that 10% of missing values for a country may already require substantive justification before interpreting results after imputation. It is up to the user whether they proceed with the analysis after checking the missingness of their data.

Weighted average smoothing of a complete dataset

This section focuses on smoothing values observed in the considered time interval. A smoothing procedure removes sudden large changes, showing a less variable time series than the original. Given that short time series (panel data) are considered here, a three-point weighted average is proposed. The smoothing procedure substitutes an original raw value $y_{m,i,t}$ of country m indicator i at time t with the weighted average

$$\check{y}_{m,i,t} = y_{m,i,t-1}\ (1-w)/2 + w\ y_{m,i,t} + y_{m,i,t+1}\ (1-w)/2$$

where $0 < w \leq 1$. The special case $w = 1$ corresponds to no smoothing. In case of missing values, an NA is returned. If the weight is outside the interval $(0, 1]$, an NA is returned. The first and last values are smoothed using weights w and $1 - w$.

After loading the data, imputation and smoothing should be performed. The example below with countries IT and DE illustrates how to do so. First, check if missing values are present:

```
workTB <- dplyr::select(emp_20_64_MS, time, IT,DE)
check_data(workTB)
#> $res
#> [1] TRUE
#>
#> $msg
#> NULL
#>
#> $err
#> NULL
```

If the dataframe passes the check, proceed to the smoothing step. Delete the time variable:

```
resSM <- smoo_dataset(select(workTB,-time), leadW = 0.149, timeTB= select(workTB,time))
resSM
#> # A tibble: 17 x 3
#>     time    IT    DE
#>    <dbl> <dbl> <dbl>
#>  1  2002  60.0  68.5
#>  2  2003  60.4  68.4
#>  3  2004  60.9  68.8
#>  4  2005  62.0  69.5
#>  5  2006  62.1  71.1
#>  6  2007  62.7  72.6
#>  7  2008  62.3  73.6
#>  8  2009  61.9  74.5
#>  9  2010  61.3  75.3
#> 10  2011  61.0  76.0
#> 11  2012  60.4  76.9
#> 12  2013  60.3  77.3
#> 13  2014  60.1  77.7
#> 14  2015  60.7  78.1
#> 15  2016  61.4  78.6
#> 16  2017  62.3  79.2
#> 17  2018  62.4  79.3
```

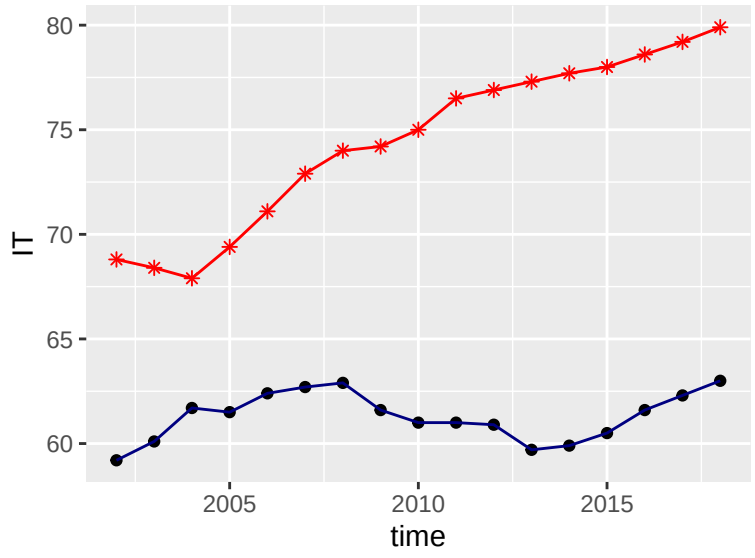
For a comparison:

```
compaTB<-bind_cols(workTB, select(resSM,-time))
#> New names:
#> * IT -> IT...2
#> * DE -> DE...3
#> * IT -> IT...4
#> * DE -> DE...5
names(compaTB)<-c("time", "IT","IT1","DE", "DE1")
compaTB
#> # A tibble: 17 x 5
#>     time    IT   IT1    DE   DE1
#>    <dbl> <dbl> <dbl> <dbl> <dbl>
#>  1  2002  59.2  68.8  60.0  68.5
#>  2  2003  60.1  68.4  60.4  68.4
#>  3  2004  61.7  67.9  60.9  68.8
#>  4  2005  61.5  69.4  62.0  69.5
```

```
#>  5  2006  62.4  71.1  62.1  71.1
#>  6  2007  62.7  72.9  62.7  72.6
#>  7  2008  62.9  74    62.3  73.6
#>  8  2009  61.6  74.2  61.9  74.5
#>  9  2010  61    75    61.3  75.3
#> 10  2011  61    76.5  61.0  76.0
#> 11  2012  60.9  76.9  60.4  76.9
#> 12  2013  59.7  77.3  60.3  77.3
#> 13  2014  59.9  77.7  60.1  77.7
#> 14  2015  60.5  78    60.7  78.1
#> 15  2016  61.6  78.6  61.4  78.6
#> 16  2017  62.3  79.2  62.3  79.2
#> 17  2018  63    79.9  62.4  79.3
```

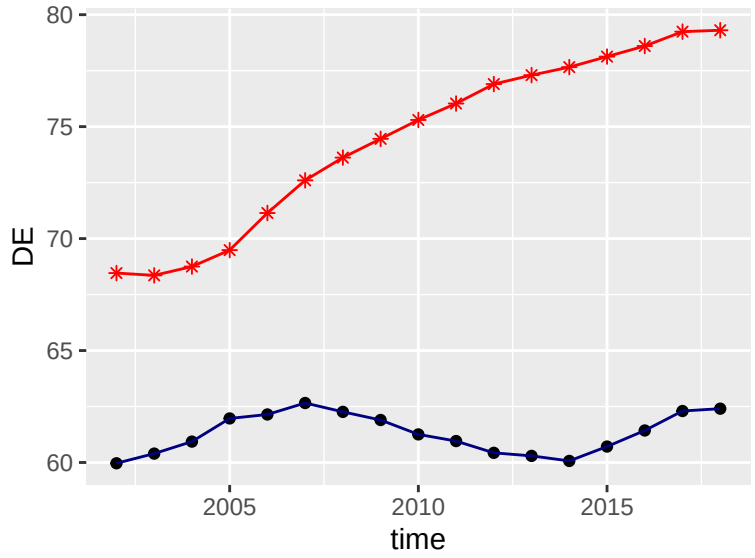
A graphical output shows changes for "IT", with the original index in blue and the smoothed index in red:

```
qplot(time,IT,data=compaTB) +
  geom_line(colour="navyblue") +
  geom_line(aes(x=time,y=IT1),colour="red") +
  geom_point(aes(x=time,y=IT1),colour="red",shape=8)
```



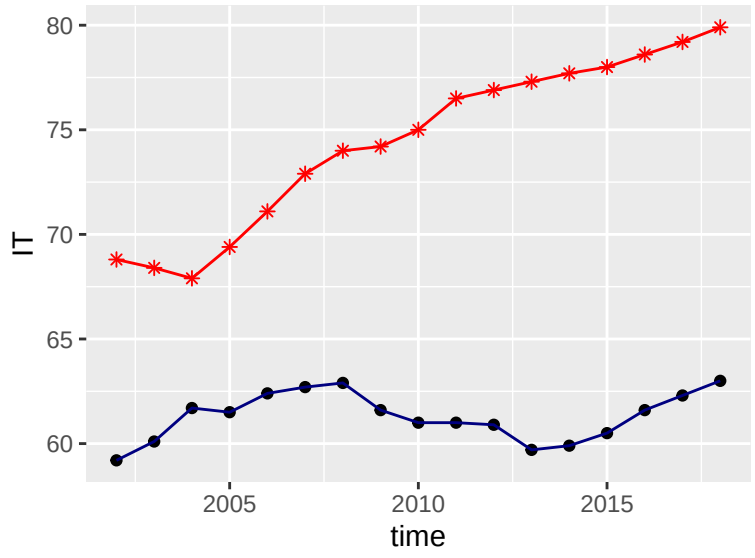
Similarly, for DE:

```
qplot(time,DE,data=compaTB) +
  geom_line(colour="navyblue") +
  geom_line(aes(x=time,y=DE1),colour="red") +
  geom_point(aes(x=time,y=DE1),colour="red",shape=8)
```



A weight equal to 1 leaves the data unchanged:

```
resSM <- smoo_dataset(select(workTB,-time), leadW = 1, timeTB= select(workTB,time))
compaTB <- bind_cols(workTB, select(resSM,-time))
#> New names:
#> * IT -> IT...2
#> * DE -> DE...3
#> * IT -> IT...4
#> * DE -> DE...5
names(compaTB)<-c("time", "IT","IT1","DE", "DE1")
qplot(time,IT,data=compaTB) +
  geom_line(colour="navyblue") +
  geom_line(aes(x=time,y=IT1),colour="red") +
  geom_point(aes(x=time,y=IT1),colour="red",shape=8)
```



A time window larger than 3 could be considered, but please consider that profound economic and social changes can occur in the EU over 3 years.

Moving average smoother

Several alternative smoothing algorithms are available in R. Classical *ma* smoothers are described in the *caTools* package.

The *emp_20_64_MS* dataset is now chosen with Italy as the country operations will be showcased upon.

```
data(emp_20_64_MS)
cuTB <- select(emp_20_64_MS,time)
cuTB <- mutate(cuTB,ITori =emp_20_64_MS$IT)
```

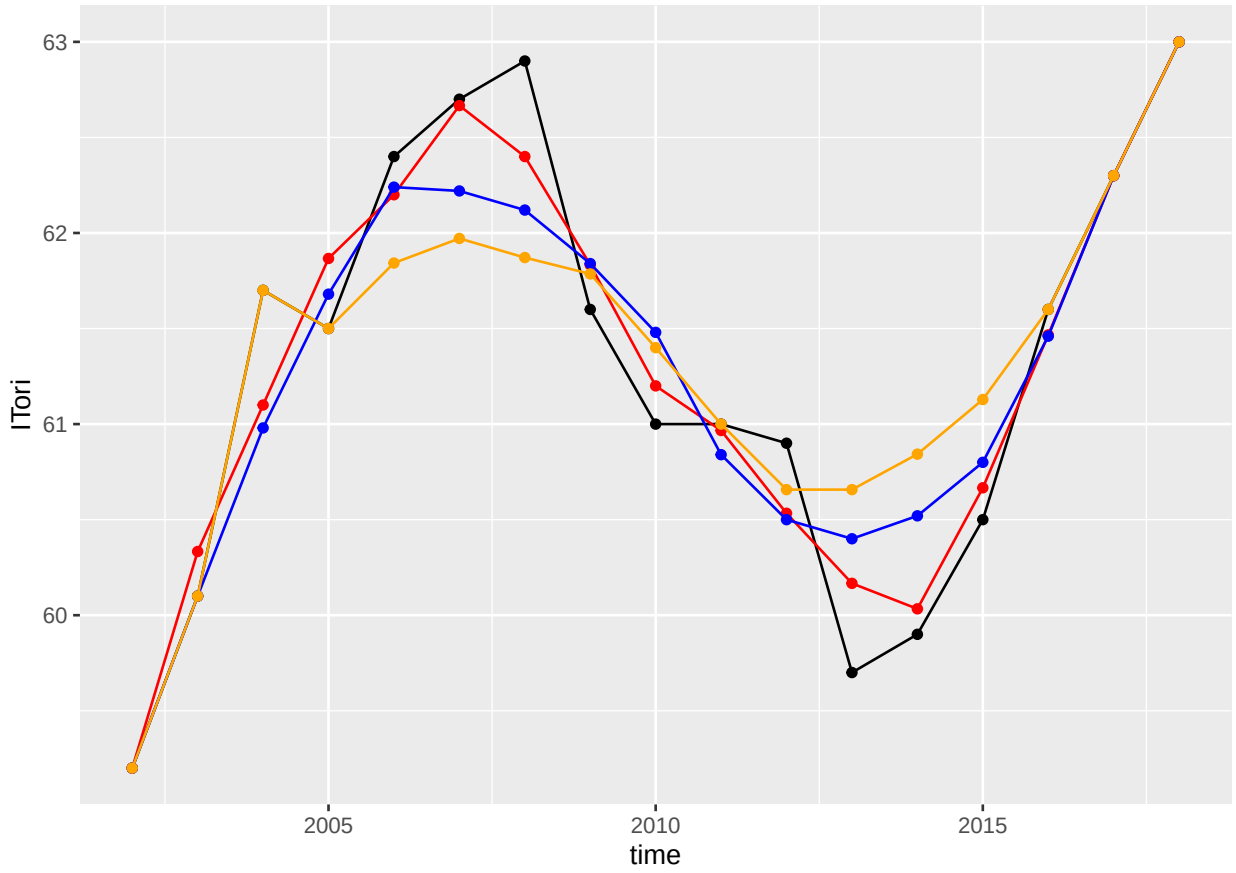
At the beginning and end of this series values are averages calculated on a smaller and smaller number of observations (tails):

```
cuTB <- mutate(cuTB, IT_k_3= runmean(emp_20_64_MS$IT, k=3,
        alg=c("C", "R", "fast", "exact")[4],
        endrule=c("mean", "NA", "trim", "keep", "constant", "func")[4],
        align = c("center", "left", "right")[1]))

cuTB <-  mutate(cuTB, IT_k_5= runmean(emp_20_64_MS$IT, k=5,
        alg=c("C", "R", "fast", "exact")[4],
        endrule=c("mean", "NA", "trim", "keep", "constant", "func")[4],
        align = c("center", "left", "right")[1]))

cuTB <-  mutate(cuTB, IT_k_7= runmean(emp_20_64_MS$IT, k=7,
        alg=c("C", "R", "fast", "exact")[4],
        endrule=c("mean", "NA", "trim", "keep", "constant", "func")[4],
        align = c("center", "left", "right")[1]))
```

The options *alg,endrule,align* in the *runmean* function are discussed in the *caTools* package.

The figure below shows results for different degrees of smoothing: original (black), k=3 (red), k=5 (blue), k=7 (orange).

```
myG <- ggplot(cuTB,aes(x=time,y=ITori))+geom_line()+geom_point()+
       geom_line(aes(x=time,y=IT_k_3),colour="red")+
       geom_point(aes(x=time,y=IT_k_3),colour="red")+
       #
       geom_line(aes(x=time,y=IT_k_5),colour="blue")+
       geom_point(aes(x=time,y=IT_k_5),colour="blue")+
       #
       geom_line(aes(x=time,y=IT_k_7),colour="orange")+
       geom_point(aes(x=time,y=IT_k_7),colour="orange")

myG
```



It is typically the case that the time series is so short that at $k = 7$ a lot of observations are smoothed with different number of observations (shorter at start and end).

The above calculations can be performed as shown below in the *convergEU* package:

```
cuTB <-  emp_20_64_MS[,c("time","IT","DE")]
ma_dataset(cuTB, kappa=3, timeName= "time")
#> $res
#> # A tibble: 17 x 3
#>     time    IT    DE
#>    <dbl> <dbl> <dbl>
#>  1  2002  59.2  68.8
#>  2  2003  60.3  68.4
#>  3  2004  61.1  68.6
#>  4  2005  61.9  69.5
#>  5  2006  62.2  71.1
#>  6  2007  62.7  72.7
#>  7  2008  62.4  73.7
#>  8  2009  61.8  74.4
#>  9  2010  61.2  75.2
#> 10  2011  61.0  76.1
#> 11  2012  60.5  76.9
#> 12  2013  60.2  77.3
#> 13  2014  60.0  77.7
#> 14  2015  60.7  78.1
#> 15  2016  61.5  78.6
```

```
#> 16  2017  62.3  79.2
#> 17  2018  63    79.9
#>
#> $msg
#> NULL
#>
#> $err
#> NULL
```

This approach is a bit less customizable, but it leads to a standard tidy dataset.

Absolute change

Absolute change for country m, indicator i at time t is defined as:

$$\Delta y_{m,i,t} = y_{m,i,t} - y_{m,i,t-1}$$

This can be calculated with the *convergEU* package with the function *abso_change*. In the *emp_20_64_MS* dataset, which is tidy and has no missing values:

```
data(emp_20_64_MS)
mySTB <- abso_change(emp_20_64_MS,
                         time_0 = 2005,
                         time_t = 2010,
                         all_within=TRUE,
                         timeName = "time")
names(mySTB$res)
#> [1] "abso_change"         "sum_abs_change"      "average_abs_change"
```

The equation above results in:

```
mySTB$res$abso_change
#>   time   AT   BE  BG   CY   CZ   DE   DK  EE  EL  ES   FI   FR  HR   HU   IE
#> 1 2003  0.4 -0.2 2.2  0.3 -0.7 -0.4 -0.9 0.8 1.0 1.1 -0.3  1.1 0.5  1.0 -0.4
#> 2 2004 -2.9  1.3 2.5  0.3 -0.9 -0.5  0.7 1.1 0.5 0.9 -0.4 -0.5 1.3 -0.4  0.6
#> 3 2005  2.0  0.7 0.7 -1.3  0.6  1.5 -0.1 1.8 0.1 2.3  0.5  0.2 0.3  0.2  1.6
#> 4 2006  1.2  0.0 3.2  1.4  0.5  1.7  1.4 3.9 1.2 1.5  0.9  0.0 0.6  0.4  0.8
#> 5 2007  1.2  1.2 3.3  1.0  0.8  1.8 -0.4 1.0 0.2 0.7  0.9  0.5 3.3 -0.3  1.7
#>     IT   LT   LU  LV   MT   NL   PL   PT   RO   SE   SI   SK  UK
#> 1  0.9  2.7 -1.2 1.1 -0.4 -0.5 -0.4 -1.1  0.5 -0.3 -1.9  1.8 0.4
#> 2  1.6 -1.1  0.5 0.0 -0.5 -0.4 -0.3 -0.4 -0.1 -0.7  2.9 -1.5 0.2
#> 3 -0.2  1.1  1.3 1.7  0.1 -2.2  1.3 -0.4 -1.1  0.3  0.1  1.0 0.3
#> 4  0.9  0.6  0.1 4.1  0.5  1.0  1.8  0.4  1.2  0.7  0.4  1.5 0.0
#> 5  0.3  1.4  0.5 2.0  0.7  1.8  2.6 -0.1 -0.4  1.3  0.9  1.2 0.0
```

If desired, less digits may be displayed. For instance, by rounding we get:

```
round(dplyr::select(mySTB$res$abso_change,AT:UK), 5)
#>     AT   BE  BG   CY   CZ   DE   DK  EE  EL  ES   FI   FR  HR   HU   IE   IT
#> 1  0.4 -0.2 2.2  0.3 -0.7 -0.4 -0.9 0.8 1.0 1.1 -0.3  1.1 0.5  1.0 -0.4  0.9
#> 2 -2.9  1.3 2.5  0.3 -0.9 -0.5  0.7 1.1 0.5 0.9 -0.4 -0.5 1.3 -0.4  0.6  1.6
```

```
#> 3  2.0  0.7 0.7 -1.3  0.6  1.5 -0.1 1.8 0.1 2.3  0.5  0.2 0.3  0.2  1.6 -0.2
#> 4  1.2  0.0 3.2  1.4  0.5  1.7  1.4 3.9 1.2 1.5  0.9  0.0 0.6  0.4  0.8  0.9
#> 5  1.2  1.2 3.3  1.0  0.8  1.8 -0.4 1.0 0.2 0.7  0.9  0.5 3.3 -0.3  1.7  0.3
#>     LT   LU  LV   MT   NL   PL   PT   RO   SE   SI   SK  UK
#> 1  2.7 -1.2 1.1 -0.4 -0.5 -0.4 -1.1  0.5 -0.3 -1.9  1.8 0.4
#> 2 -1.1  0.5 0.0 -0.5 -0.4 -0.3 -0.4 -0.1 -0.7  2.9 -1.5 0.2
#> 3  1.1  1.3 1.7  0.1 -2.2  1.3 -0.4 -1.1  0.3  0.1  1.0 0.3
#> 4  0.6  0.1 4.1  0.5  1.0  1.8  0.4  1.2  0.7  0.4  1.5 0.0
#> 5  1.4  0.5 2.0  0.7  1.8  2.6 -0.1 -0.4  1.3  0.9  1.2 0.0
```

The sum of absolute values

$$\sum_{t=t_0+1} |\Delta y_{m,i,t}|$$

is:

```
round(mySTB$res$sum_abs_change,4)
#>   AT   BE   BG   CY   CZ   DE   DK   EE   EL   ES   FI   FR   HR   HU   IE   IT
#>  7.7  3.4 11.9  4.3  3.5  5.9  3.5  8.6  3.0  6.5  3.0  2.3  6.0  2.3  5.1  3.9
#>   LT   LU   LV   MT   NL   PL   PT   RO   SE   SI   SK   UK
#>  6.9  3.6  8.9  2.2  5.9  6.4  2.4  3.3  3.3  6.2  7.0  0.9
```

Such sum can be divided by the number of pairs of years so that the result is an average per pair of years:

```
round(mySTB$res$average_abs_change,4)
#>   AT   BE   BG   CY   CZ   DE   DK   EE   EL   ES   FI   FR   HR   HU   IE   IT
#> 1.54 0.68 2.38 0.86 0.70 1.18 0.70 1.72 0.60 1.30 0.60 0.46 1.20 0.46 1.02 0.78
#>   LT   LU   LV   MT   NL   PL   PT   RO   SE   SI   SK   UK
#> 1.38 0.72 1.78 0.44 1.18 1.28 0.48 0.66 0.66 1.24 1.40 0.18
```

Summaries and clusters of countries

The unweighted average of country values is an important summary statistic. The possible sets of countries that can be specified are stored within the function generating global static objects and tables, called *convergEU_glb()*. Below we showcase how to use this function with the *emp_20_64_MS* dataset.

First note that the EU area includes the following MS:

```
convergEU_glb()$Eurozone
#> NULL
convergEU_glb()$EU19
#> NULL
```

The labels for the 28 MS are:

```
convergEU_glb()$EU28
#> $dates
#> [1] "01/06/2013" "01/02/2020"
#>
#> $memberStates
#> # A tibble: 28 x 2
#>    MS          codeMS
```

```
#>    <chr>       <chr>
#>  1 Belgium     BE
#>  2 Denmark     DK
#>  3 France      FR
#>  4 Germany     DE
#>  5 Greece      EL
#>  6 Ireland     IE
#>  7 Italy       IT
#>  8 Luxembourg  LU
#>  9 Netherlands NL
#> 10 Portugal    PT
#> # ... with 18 more rows
```

The list of known MS labels is:

```
names(convergEU_glb())[3:7]
#> [1] "EA19"      "EU12"      "EU15"      "EU25"      "EU27_2007"
```

For example, the unweighted average in the *emp_20_64_MS* dataset for the EU28 would be:

```
average_clust(emp_20_64_MS,
              timeName = "time",
              cluster = "EU28")$res[,c(1,30)]
#> # A tibble: 17 x 2
#>     time  EU28
#>    <dbl> <dbl>
#>  1  2002  67.5
#>  2  2003  67.8
#>  3  2004  67.9
#>  4  2005  68.4
#>  5  2006  69.6
#>  6  2007  70.6
#>  7  2008  71.0
#>  8  2009  69.0
#>  9  2010  68.1
#> 10  2011  68.1
#> 11  2012  68
#> 12  2013  68.0
#> 13  2014  69.0
#> 14  2015  70.0
#> 15  2016  71.0
#> 16  2017  72.5
#> 17  2018  73.8
```

while for EU12 it would be:

```
average_clust(emp_20_64_MS,
              timeName = "time",
              cluster = "EU12")$res[,c(1,30)]
#> # A tibble: 17 x 2
#>     time  EU12
#>    <dbl> <dbl>
#>  1  2002  69.1
```

```
#>  2  2003  69.1
#>  3  2004  69.4
#>  4  2005  69.9
#>  5  2006  70.6
#>  6  2007  71.3
#>  7  2008  71.4
#>  8  2009  69.9
#>  9  2010  69.2
#> 10  2011  68.6
#> 11  2012  68.0
#> 12  2013  67.8
#> 13  2014  68.4
#> 14  2015  69.2
#> 15  2016  70.1
#> 16  2017  71.2
#> 17  2018  72.3
```

An unknown label, like "EUspirit", would cause a computational error:

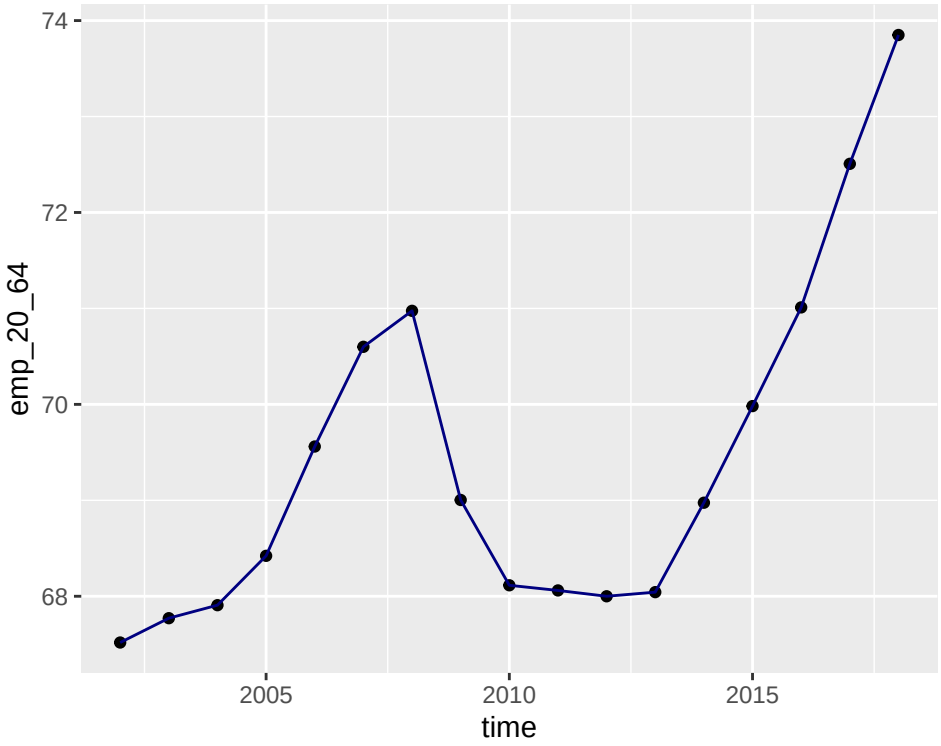
```
average_clust(emp_20_64_MS,timeName = "time",cluster = "EUspirit")
#> $res
#> NULL
#>
#> $msg
#> NULL
#>
#> $err
#> [1] "Error: badly specified countries."
```

. . . as would an incorrect time name:

```
average_clust(emp_20_64_MS,timeName = "TTime",cluster = "EA")
#> $res
#> NULL
#>
#> $msg
#> NULL
#>
#> $err
#> [1] "Error: Time variable not in the dataframe."
```

The time series can be plotted as shown:

```
wwTB <- average_clust(emp_20_64_MS,timeName = "time",cluster = "EU28")$res[,c(1,30)]
mini_EU <- min(wwTB$EU28)
maxi_EU <- max(wwTB$EU28)

qplot(time, EU28, data=wwTB,
      ylim=c(mini_EU,maxi_EU))+geom_line(colour="navy blue")+
      ylab("emp_20_64")
```



The analysis of convergence

Several measures of convergence have been recently proposed by Eurofound (Eurofound, 2018). In this section, each each measure is introduced and its usage showcased.

Beta-convergence

Let's assume that we have a tidy dataset (tibble) in the form years by countries. The calculations for beta convergence are performed according the following linear model:

$$\tau^{-1}(ln(y_{m,i,t+\tau}) - ln(y_{m,i,t})) = \beta_0 + \beta_1 ln(y_{m,i,t}) + \epsilon_{m,i,t}$$

where m represents an EU Member State (country), i refers to an indicator of interest, t is the reference time and $\tau \in \{1, 2, \ldots\}$ the length of the time window (typically 1 or more years).

The output of *beta_conv()* is a list in which transformed data, the point estimate of β_1 and a standard two-tails test (including the p-value and adjusted R squared) is reported . While it is not implemented in the package, a one-tail test $H_0 : \beta_1 \geq 0$ against $H_1 : \beta 1 < 0$ may also be used. In the implementation of the function *beta_conv()*, the same reference time is maintained across different years. The division of the left hand side by the amount of time elapsed can be skipped by passing the argument *useTau = FALSE*.

Below is an example on how to invoke the function:

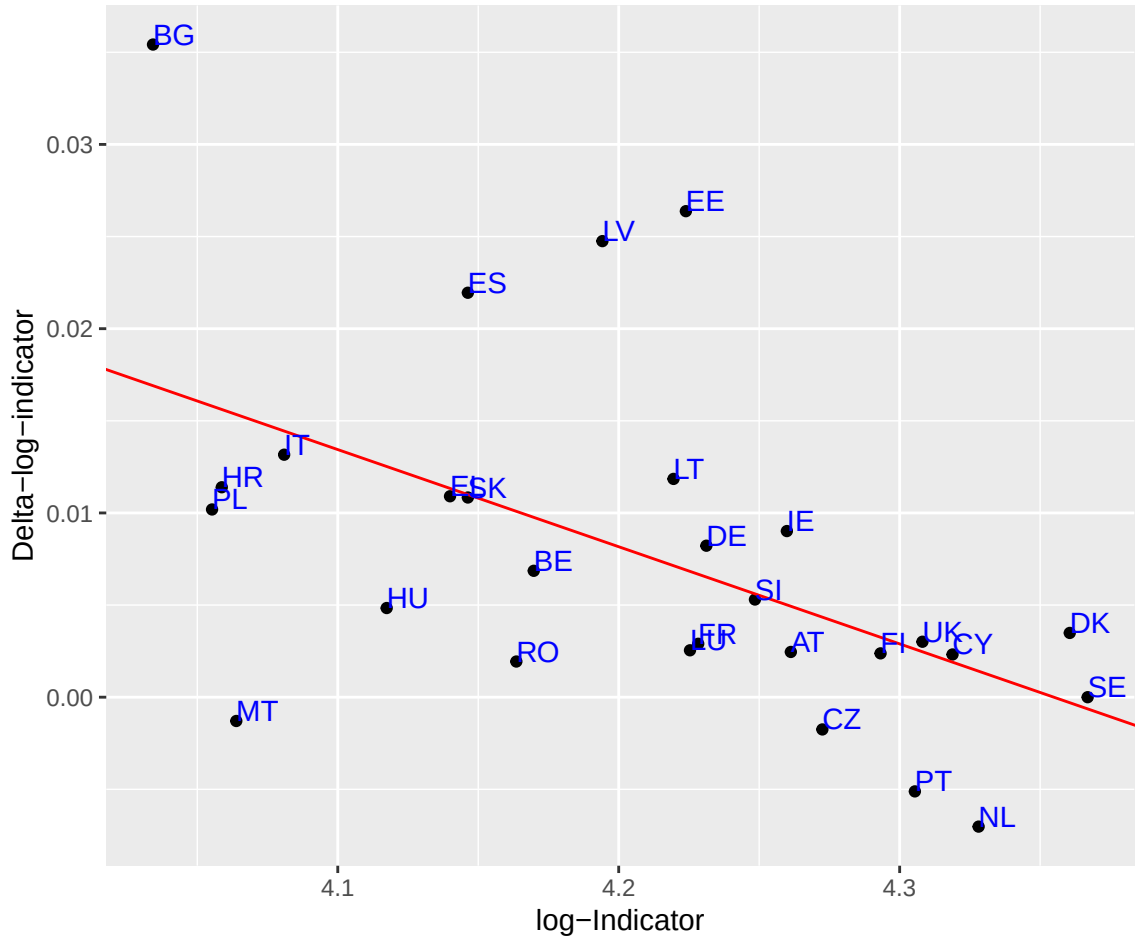
```
require(ggplot2)
require(dplyr)
require(tibble)

empBC <- beta_conv(emp_20_64_MS,
```

```
                time_0 = 2002,
                time_t = 2006,
                all_within = FALSE,
                timeName = "time")
empBC
#> $res
#> $res$workTB
#> # A tibble: 28 x 3
#>    deltaIndic indic countries
#>         <dbl> <dbl> <chr>
#>  1    0.00246  4.26 AT
#>  2    0.00686  4.17 BE
#>  3    0.0354   4.03 BG
#>  4    0.00232  4.32 CY
#>  5   -0.00175  4.27 CZ
#>  6    0.00822  4.23 DE
#>  7    0.00349  4.36 DK
#>  8    0.0264   4.22 EE
#>  9    0.0109   4.14 EL
#> 10    0.0220   4.15 ES
#> # ... with 18 more rows
#>
#> $res$model
#>
#> Call:
#> stats::lm(formula = deltaIndic ~ indic, data = workTB)
#>
#> Coefficients:
#> (Intercept)        indic
#>     0.22962     -0.05273
#>
#>
#> $res$summary
#> # A tibble: 2 x 5
#>   term        estimate std.error statistic p.value
#>   <chr>          <dbl>     <dbl>     <dbl>   <dbl>
#> 1 (Intercept)   0.230     0.0704      3.26 0.00310
#> 2 indic        -0.0527    0.0167     -3.15 0.00406
#>
#> $res$beta1
#> [1] -0.05272631
#>
#> $res$adj.r.squared
#> [1] 0.2485567
#>
#>
#> $msg
#> NULL
#>
#> $err
#> NULL
```

Note that *all_within = FALSE* is the default.

A plot of transformed data and the regression line can be obtained by running:

```
qplot(empBC$res$workTB$indic,
      empBC$res$workTB$deltaIndic,
      xlab="log-Indicator",
      ylab="Delta-log-indicator") +
  geom_abline(intercept = as.numeric(empBC$res$summary[1,2]),
              slope = as.numeric(empBC$res$summary[2,2]),
              colour = "red") +
  geom_text(aes(label=empBC$res$workTB$countries),
            hjust=0, vjust=0,colour="blue")
```



Labels are replicated as many times as the number of included years if *all_within=TRUE* was specified. Furthermore, note that if the value of the indicator at the start or end time were 0, calculating beta convergence would be impossible (since the log of 0 is not defined). To bypass this and allow the calculation of beta-convergence, a very small constant (equal to a hundredth of the smallest value in the dataset) is added to the indicator where it equals 0. This allows the calculation of beta convergence and should not affect the outcome of the analysis.

Sigma-convergence

The key concempt in sigma-convergence is variability with respect to the mean. Let $Y_{m,i,t}$ be the value of indicator i for member state m at time t, and $\overline{Y}_{A,i,t}$ the average over aggregation A. If $A = EU28$, then:

- the average is $\overline{Y}_{A,i,t} = n(A)^{-1} \sum_{m \in A} Y_{m,i,t}$, where $n(A)$ is the number of member states within aggregation A;
- the standard deviation is $s_{A,i,t} = \sqrt{(n(A)^{-1} \sum_{m \in A} (Y_{m,i,t} - \overline{Y}_{A,i,t})^2)}$;
- the coefficient of variation is $CV(A,i,t) = 100 \cdot \frac{s_{A,i,t}}{\overline{Y}_{A,i,t}}$.

For each year, the summaries above are calculated to let the user see if convergence (i.e., a reduction in heterogeneity) took place. Below is how to test for sigma convergence across the entire time interval in the dataset:

```
mySTB <- sigma_conv(emp_20_64_MS, timeName="time")
mySTB
#> $res
#> # A tibble: 17 x 5
#>     time stdDev     CV  mean devianceT
#>    <dbl>  <dbl>  <dbl> <dbl>     <dbl>
#>  1  2002   6.34 0.0939  67.5     1125.
#>  2  2003   5.95 0.0878  67.8      991.
#>  3  2004   5.70 0.0839  67.9      909.
#>  4  2005   5.54 0.0809  68.4      858.
#>  5  2006   5.57 0.0801  69.6      869.
#>  6  2007   5.47 0.0775  70.6      838.
#>  7  2008   5.36 0.0755  71.0      804.
#>  8  2009   5.03 0.0730  69.0      710.
#>  9  2010   5.24 0.0769  68.1      768.
#> 10  2011   5.59 0.0821  68.1      875.
#> 11  2012   5.98 0.0880  68       1002.
#> 12  2013   6.28 0.0922  68.0     1103.
#> 13  2014   5.98 0.0867  69.0     1000.
#> 14  2015   5.74 0.0820  70.0      922.
#> 15  2016   5.60 0.0789  71.0      879.
#> 16  2017   5.37 0.0741  72.5      808.
#> 17  2018   5.30 0.0717  73.8      786.
#>
#> $msg
#> NULL
#>
#> $err
#> NULL
```

It is also possible to specify a time interval:

```
sigma_conv(emp_20_64_MS, time_0 = 2002, time_t = 2004)
#> $res
#> # A tibble: 3 x 5
#>    time stdDev     CV  mean devianceT
#>   <dbl>  <dbl>  <dbl> <dbl>     <dbl>
#> 1  2002   6.34 0.0939  67.5     1125.
#> 2  2003   5.95 0.0878  67.8      991.
#> 3  2004   5.70 0.0839  67.9      909.
#>
#> $msg
```

```
#> NULL
#>
#> $err
#> NULL
```

The departure from the mean, where $-1, 0, 1$ indicates values respectively below -1 within the interval $(-1, 1)$ and above $+1$, can be characterized like so:

```
res <- departure_mean(oriTB = emp_20_64_MS, sigmaTB = mySTB$res)
names(res$res)
#> [1] "departures"      "squaredContrib"  "devianceContrib"
res$res$departures
#> # A tibble: 17 x 33
#>     time stdDev     CV  mean devianceT    AT    BE    BG    CY    CZ    DE    DK
#>    <dbl>  <dbl>  <dbl> <dbl>     <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
#>  1  2002   6.34 0.0939  67.5     1125.     0     0    -1     1     0     0     1
#>  2  2003   5.95 0.0878  67.8      991.     0     0    -1     1     0     0     1
#>  3  2004   5.70 0.0839  67.9      909.     0     0    -1     1     0     0     1
#>  4  2005   5.54 0.0809  68.4      858.     0     0    -1     1     0     0     1
#>  5  2006   5.57 0.0801  69.6      869.     0     0     0     1     0     0     1
#>  6  2007   5.47 0.0775  70.6      838.     0     0     0     1     0     0     1
#>  7  2008   5.36 0.0755  71.0      804.     0     0     0     1     0     0     1
#>  8  2009   5.03 0.0730  69.0      710.     0     0     0     1     0     1     1
#>  9  2010   5.24 0.0769  68.1      768.     1     0     0     1     0     1     1
#> 10  2011   5.59 0.0821  68.1      875.     1     0     0     0     0     1     1
#> 11  2012   5.98 0.0880  68       1002.     1     0     0     0     0     1     1
#> 12  2013   6.28 0.0922  68.0     1103.     1     0     0     0     0     1     0
#> 13  2014   5.98 0.0867  69.0     1000.     0     0     0     0     0     1     0
#> 14  2015   5.74 0.0820  70.0      922.     0     0     0     0     0     1     0
#> 15  2016   5.60 0.0789  71.0      879.     0     0     0     0     1     1     0
#> 16  2017   5.37 0.0741  72.5      808.     0     0     0     0     1     1     0
#> 17  2018   5.30 0.0717  73.8      786.     0     0     0     0     1     1     0
#> # ... with 21 more variables: EE <dbl>, EL <dbl>, ES <dbl>, FI <dbl>, FR <dbl>,
#> #   HR <dbl>, HU <dbl>, IE <dbl>, IT <dbl>, LT <dbl>, LU <dbl>, LV <dbl>,
#> #   MT <dbl>, NL <dbl>, PL <dbl>, PT <dbl>, RO <dbl>, SE <dbl>, SI <dbl>,
#> #   SK <dbl>, UK <dbl>
```

Details on the contribution of each MS to the variance at a given time t is evaluated by the square of the difference $(Y_{m,i,t} - \overline{Y}_{EU28,i,t})^2$ between the indicator i of country m at time t and the unweighted average of the member states. These can be can be obtained by running:

```
res$res$squaredContrib
#> # A tibble: 17 x 28
#>        AT    BE       BG       CY    CZ       DE    DK     EE    EL     ES    FI
#>     <dbl> <dbl>    <dbl>    <dbl> <dbl>   <dbl> <dbl>  <dbl> <dbl>  <dbl> <dbl>
#>  1 11.4    7.94  121.     57.5     17.5  1.64e+0 116.   0.612  22.3 18.6   32.3
#>  2 12.5   10.7    82.3    58.2     10.4  3.95e-1  92.7  1.77   15.8 12.1   26.3
#>  3  0.243  4.44   45.0    60.7      4.81 5.10e-5 104.   5.26   13.0  7.33  21.1
#>  4  3.91   3.69   42.5    35.7      5.19 9.58e-1  91.7 12.8    16.2  0.849 21.0
#>  5  4.16   9.37   19.9    38.9      2.69 2.37e+0  96.8 40.2    15.7  0.314 18.8
#>  6  4.84   8.41    4.84   38.4      1.96 5.29e+0  70.6 39.7    23.0  0.810 17.6
#>  7  7.98   8.85    0.0756 30.5      2.03 9.15e+0  59.7 37.5    21.9  6.13  23.3
#>  8 19.3    3.62    0.0414 39.6      3.60 2.70e+1  50.4  0.993  11.6 25.0   20.2
```

```
#>  9 33.5    0.264  11.7    47.4      5.22 4.74e+1  46.0  1.73   18.6 28.2   23.9
#> 10 37.7    0.579  26.6    28.5      8.06 7.12e+1  45.4  6.45   71.6 36.7   32.9
#> 11 41.0    0.640  25       4.84    12.2  7.92e+1  39.7 17.6   169   70.6   36
#> 12 43.0    0.710  20.6     0.710   19.9  8.57e+1  39.2 27.6   229.  89.2   27.6
#> 13 27.3    2.81   15.0     1.89    20.5  7.61e+1  32.8 28.4   246.  82.4   17.0
#> 14 18.6    7.74    8.31    4.34    23.2  6.43e+1  29.4 42.5   227.  63.7    8.51
#> 15 14.4   11.0    11.0     5.34    32.4  5.76e+1  24.9 31.2   219.  50.6    5.71
#> 16  8.37  16.1     1.46    2.91    35.9  4.48e+1  16.8 38.4   216.  49.1    2.87
#> 17  5.52  17.2     2.10    0.00250 36.6  3.66e+1  13.3 31.9   206.  46.9    6.00
#> # ... with 17 more variables: FR <dbl>, HR <dbl>, HU <dbl>, IE <dbl>, IT <dbl>,
#> #   LT <dbl>, LU <dbl>, LV <dbl>, MT <dbl>, NL <dbl>, PL <dbl>, PT <dbl>,
#> #   RO <dbl>, SE <dbl>, SI <dbl>, SK <dbl>, UK <dbl>
```

It is also possible to decompose the numerator of the variance, called deviance, at each time in order to calculate the percent contributed by each MS to the total deviance for indicator i of country m at time t.

$$100 \cdot \frac{(Y_{m,i,t} - \overline{Y}_{EU28,i,t})^2}{\sum_m (Y_{m,i,t} - \overline{Y}_{EU28,i,t})^2}$$

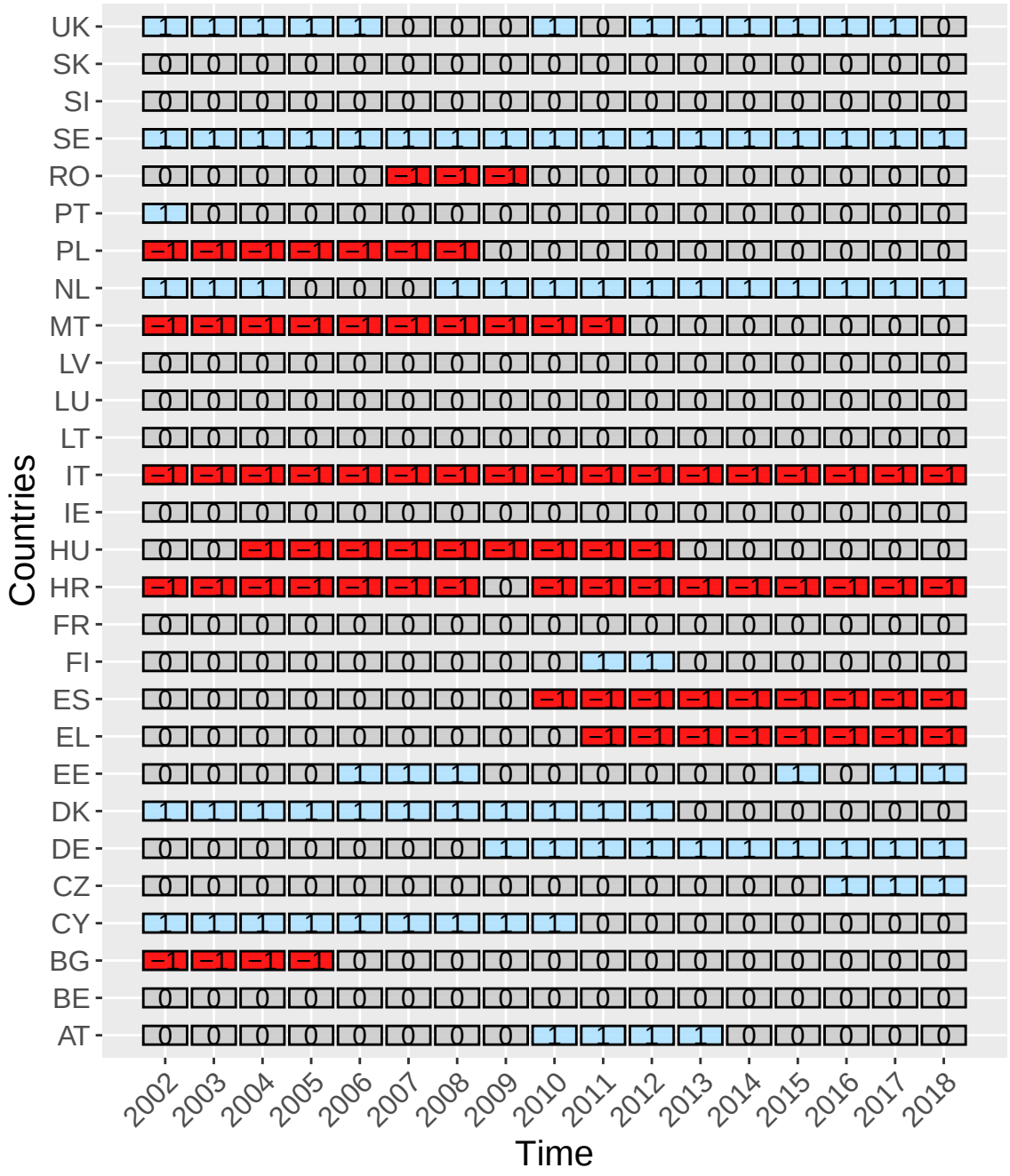
```
res$res$devianceContrib
#> # A tibble: 17 x 28
#>        AT     BE       BG       CY    CZ      DE    DK     EE    EL     ES    FI
#>     <dbl>  <dbl>    <dbl>    <dbl> <dbl>   <dbl> <dbl>  <dbl> <dbl>  <dbl> <dbl>
#>  1 1.02   0.706  10.8     5.11     1.56  1.46e-1 10.3  0.0544  1.98 1.66   2.87
#>  2 1.26   1.08    8.30    5.87     1.05  3.99e-2  9.35 0.178   1.59 1.22   2.65
#>  3 0.0267 0.488   4.95    6.68     0.529 5.61e-6 11.4  0.578   1.43 0.806  2.32
#>  4 0.456  0.430   4.95    4.16     0.605 1.12e-1 10.7  1.49    1.88 0.0989 2.44
#>  5 0.479  1.08    2.29    4.48     0.309 2.73e-1 11.1  4.63    1.81 0.0362 2.17
#>  6 0.578  1.00    0.578   4.59     0.234 6.31e-1  8.42 4.74    2.75 0.0967 2.11
#>  7 0.993  1.10    0.00941 3.80     0.253 1.14e+0  7.43 4.67    2.72 0.762  2.90
#>  8 2.72   0.511   0.00584 5.59     0.507 3.81e+0  7.10 0.140   1.63 3.53   2.85
#>  9 4.36   0.0344  1.52    6.17     0.680 6.17e+0  6.00 0.225   2.42 3.68   3.11
#> 10 4.31   0.0662  3.04    3.26     0.922 8.14e+0  5.19 0.737   8.18 4.20   3.77
#> 11 4.09   0.0639  2.50    0.483    1.22  7.91e+0  3.96 1.76   16.9  7.04   3.59
#> 12 3.90   0.0644  1.87    0.0644   1.80  7.77e+0  3.55 2.51   20.8  8.08   2.51
#> 13 2.73   0.280   1.50    0.189    2.05  7.61e+0  3.28 2.83   24.6  8.23   1.70
#> 14 2.02   0.839   0.901   0.470    2.52  6.97e+0  3.18 4.61   24.7  6.91   0.923
#> 15 1.63   1.25    1.25    0.608    3.68  6.55e+0  2.83 3.56   25.0  5.75   0.650
#> 16 1.04   1.99    0.180   0.360    4.44  5.54e+0  2.07 4.74   26.8  6.07   0.354
#> 17 0.703  2.19    0.268   0.000318 4.66  4.66e+0  1.70 4.06   26.2  5.97   0.764
#> # ... with 17 more variables: FR <dbl>, HR <dbl>, HU <dbl>, IE <dbl>, IT <dbl>,
#> #   LT <dbl>, LU <dbl>, LV <dbl>, MT <dbl>, NL <dbl>, PL <dbl>, PT <dbl>,
#> #   RO <dbl>, SE <dbl>, SI <dbl>, SK <dbl>, UK <dbl>
```

Notice that each row adds to 100.

It is possible to produce a graphical output about the main features of a country's time series, as shown below:

```
myGG <- graph_departure(res$res$departures,
                timeName = "time",
                displace = 0.25,
                displaceh = 0.45,
```

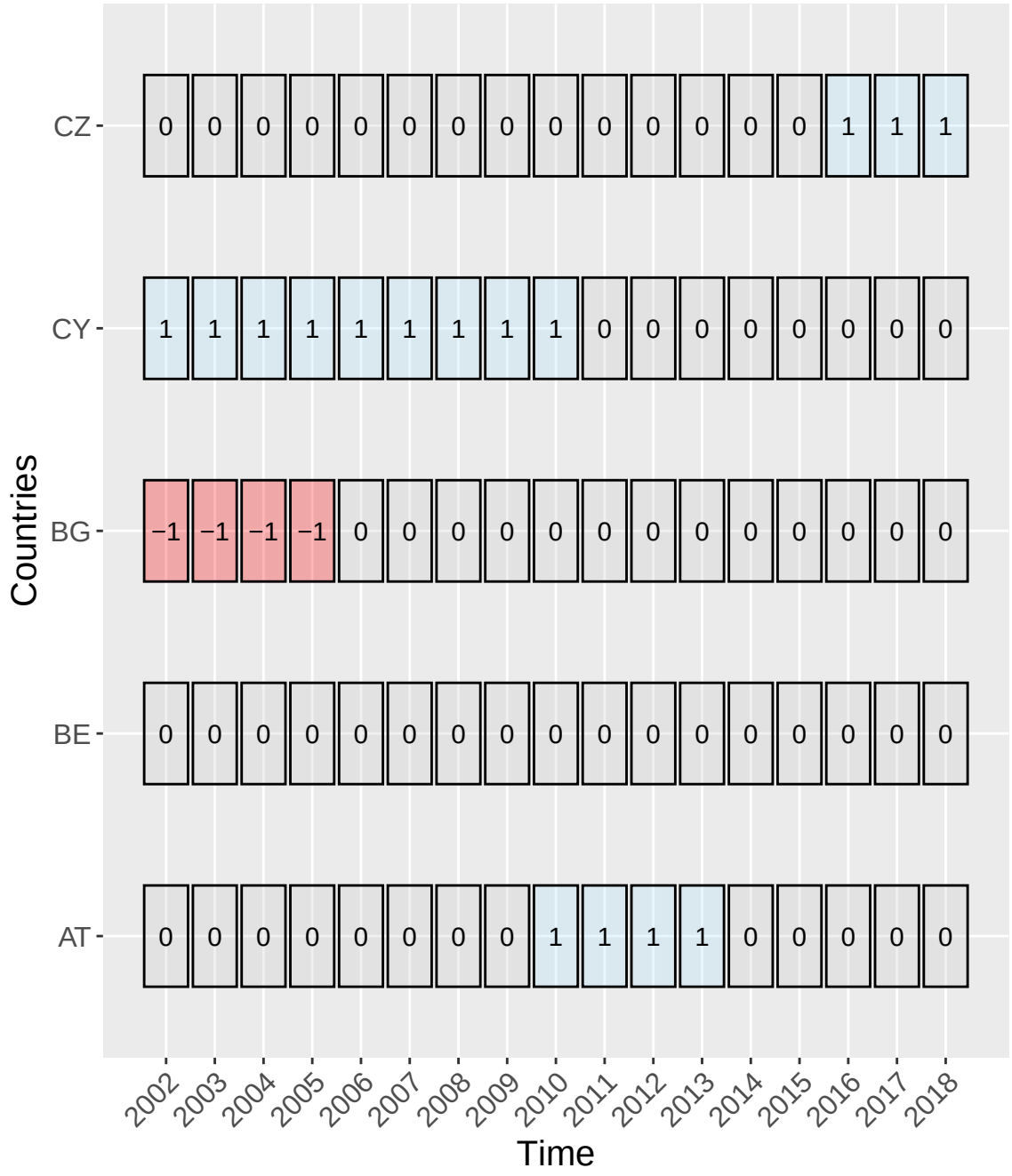
```
                dimeFontNum = 4,
                myfont_scale = 1.35,
                x_angle = 45,
                color_rect = c("-1"='red1', "0"='gray80',"1"='lightskyblue1'),
                axis_name_y = "Countries",
                axis_name_x = "Time",
                alpha_color = 0.9
                )
#> Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
#> "none")` instead.
myGG
#> $res
```




```
#>
#> $msg
#> NULL
#>
#> $err
#> NULL
```

Any selection of countries is feasible:

```r
#myWW1<- warnings()
myGG <- graph_departure(res$res$departures[,1:10],
                timeName = "time",
                displace = 0.25,
                displaceh = 0.45,
                dimeFontNum = 4,
                myfont_scale = 1.35,
                x_angle = 45,
                color_rect = c("-1"='red1', "0"='gray80',"1"='lightskyblue1'),
                axis_name_y = "Countries",
                axis_name_x = "Time",
                alpha_color = 0.29
                )
#> Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
#> "none")` instead.

myGG
#> $res
```




```
#>
#> $msg
#> NULL
#>
#> $err
#> NULL
```

Gamma-convergence

We now introduce gamma convergence by an index based on ranks.

Let $y_{m,i,t}$ be the value of indicator i for member state m at time $t = 0, 1, \ldots, T$, and $\{\tilde{y}_{m,i,t} : m \in A)$ the ranks for indicator i over member states in the reference set A, for example $A = EU28$, at a given time t. The sum of ranks within member state m is:

$$\tilde{y}_{m,i}^{(s)} = \sum_{t=0}^{T} \tilde{y}_{m,i,t}$$

The variance of the sum of ranks over the given interval

$$Var\left[\{\tilde{y}_{m,i}^{(s)} : m \in A\}\right]$$

may be compared to the variance of ranks in the reference time $t = 0$:

$$Var\left[\{\tilde{y}_{m,i,0} : m \in A\}\right]$$

The Kendall index KI, with respect to cluster A of member states for the indicator i over a given time interval is:

$$KI(A, i, T) = \frac{Var\left[\{\tilde{y}_{m,i}^{(s)} : m \in A\}\right]}{(T+1)^2 \;\; Var\left[\{\tilde{y}_{m,i,0} : m \in A\}\right]}$$

The measure of gamma-convergence is obtained with the following function:

```
gamma_conv(emp_20_64_MS,last=2016,ref=2002,timeName="time")
#> $res
#> [1] 0.7374964
#>
#> $msg
#> NULL
#>
#> $err
#> NULL
```

Delta-convergence

Let $y_{m,i,t}$ be the value of indicator i for MS m at time t, and $y_{i,t}^{(M)}$ the maximum value over member states in the reference set A (e.g., $A = EU28$):

$$y_{i,t}^{(M)} = max(\{y_{m,i,t} : m \in A\})$$

The distance of MS m from the top performer at time i is:

$$y_{i,t}^{(M)} - y_{m,i,t}$$

The overall distance at time t, called delta, is the sum of distances over the reference set A for the considered indicator i.

$$\delta_{i,t} = \sum_{m \in A} (y_{i,t}^{(M)} - y_{m,i,t})$$

Delta-convergence can be calculated as follows:

```
delta_conv(emp_20_64_MS,"time")
#> $res
#> # A tibble: 17 x 2
#>     time delta
#>    <dbl> <dbl>
#>  1  2002  316.
#>  2  2003  300.
#>  3  2004  285.
#>  4  2005  271.
#>  5  2006  276.
#>  6  2007  266.
#>  7  2008  264.
#>  8  2009  260.
#>  9  2010  280.
#> 10  2011  318.
#> 11  2012  319.
#> 12  2013  329.
#> 13  2014  309.
#> 14  2015  294.
#> 15  2016  285.
#> 16  2017  260.
#> 17  2018  245.
#>
#> $msg
#> NULL
#>
#> $err
#> NULL
```

Automated production of scoreboard and fiches

The *convergEU* package allows the user to produce scoreboards and fiches as HTML or pdf files in an automated way.

Scoreboards

Scoreboards showcase the raw values of an indicator (level, $y_{m,i,t}$) for MS m at time t for indicator i. The difference between years, i.e. the change:

$$y_{m,i,t} - y_{m,i,t-1}$$

can be calculated by following the example below.

We can produce the scoreboard for the dataset *emp_20_64_MS* with the following:

```
data(emp_20_64_MS)
resTB <- scoreb_yrs(emp_20_64_MS,timeName = "time")
resTB
#> $res
#> $res$sigma_conv
#> # A tibble: 17 x 9
#>     time stdDev     CV  mean devianceT elle1in elle1su elle2in elle2su
#>    <dbl>  <dbl>  <dbl> <dbl>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
```

```
#>  1  2002   6.34 0.0939  67.5     1125.    64.3    70.7    61.2    73.9
#>  2  2003   5.95 0.0878  67.8      991.    64.8    70.7    61.8    73.7
#>  3  2004   5.70 0.0839  67.9      909.    65.1    70.8    62.2    73.6
#>  4  2005   5.54 0.0809  68.4      858.    65.7    71.2    62.9    74.0
#>  5  2006   5.57 0.0801  69.6      869.    66.8    72.3    64.0    75.1
#>  6  2007   5.47 0.0775  70.6      838.    67.9    73.3    65.1    76.1
#>  7  2008   5.36 0.0755  71.0      804.    68.3    73.7    65.6    76.3
#>  8  2009   5.03 0.0730  69.0      710.    66.5    71.5    64.0    74.0
#>  9  2010   5.24 0.0769  68.1      768.    65.5    70.7    62.9    73.4
#> 10  2011   5.59 0.0821  68.1      875.    65.3    70.9    62.5    73.7
#> 11  2012   5.98 0.0880  68       1002.    65.0    71.0    62.0    74.0
#> 12  2013   6.28 0.0922  68.0     1103.    64.9    71.2    61.8    74.3
#> 13  2014   5.98 0.0867  69.0     1000.    66.0    72.0    63.0    75.0
#> 14  2015   5.74 0.0820  70.0      922.    67.1    72.9    64.2    75.7
#> 15  2016   5.60 0.0789  71.0      879.    68.2    73.8    65.4    76.6
#> 16  2017   5.37 0.0741  72.5      808.    69.8    75.2    67.1    77.9
#> 17  2018   5.30 0.0717  73.8      786.    71.2    76.5    68.6    79.1
#>
#> $res$sco_level
#> # A tibble: 17 x 29
#>     time    AT    BE    BG    CY    CZ    DE    DK    EE    EL    ES    FI    FR
#>    <dbl> <int> <int> <int> <int> <int> <int> <int> <int> <int> <int> <int> <int>
#>  1  2002     4     3     1     5     4     3     5     3     2     2     4     3
#>  2  2003     4     2     1     5     4     3     5     3     2     2     4     3
#>  3  2004     3     3     1     5     3     3     5     3     2     3     4     3
#>  4  2005     3     3     1     5     3     3     5     4     2     3     4     3
#>  5  2006     3     2     2     5     3     3     5     5     2     3     4     3
#>  6  2007     3     2     3     5     3     3     5     5     2     3     4     3
#>  7  2008     4     2     3     5     3     4     5     5     2     3     4     3
#>  8  2009     4     3     3     5     3     5     5     3     2     2     4     3
#>  9  2010     5     3     2     5     3     5     5     3     2     1     4     3
#> 10  2011     5     3     2     4     4     5     5     3     1     1     5     3
#> 11  2012     5     3     2     3     4     5     5     4     1     1     5     3
#> 12  2013     5     3     2     3     4     5     4     4     1     1     4     3
#> 13  2014     4     3     2     3     4     5     4     4     1     1     4     3
#> 14  2015     4     3     2     3     4     5     4     5     1     1     4     3
#> 15  2016     4     2     2     3     5     5     4     4     1     1     3     3
#> 16  2017     4     2     3     3     5     5     4     5     1     1     3     3
#> 17  2018     3     2     3     3     5     5     4     5     1     1     3     3
#> # ... with 16 more variables: HR <int>, HU <int>, IE <int>, IT <int>, LT <int>,
#> #   LU <int>, LV <int>, MT <int>, NL <int>, PL <int>, PT <int>, RO <int>,
#> #   SE <int>, SI <int>, SK <int>, UK <int>
#>
#> $res$sco_change
#> # A tibble: 17 x 29
#>     time    AT    BE    BG    CY    CZ    DE    DK    EE    EL    ES    FI    FR
#>    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
#>  1  2002    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA    NA
#>  2  2003     3     3     5     3     2     2     1     4     4     4     2     4
#>  3  2004     1     4     5     3     2     2     3     4     3     4     3     2
#>  4  2005     5     3     3     1     3     4     2     5     3     5     3     3
#>  5  2006     3     1     5     3     2     4     3     5     3     3     3     1
#>  6  2007     3     3     5     3     3     4     1     3     2     3     3     2
```

```
#>  7  2008     4     3     5     2     3     4     2     3     3     1     4     3
#>  8  2009     4     3     3     3     3     4     3     1     4     1     3     3
#>  9  2010     5     5     1     3     3     5     3     1     2     3     3     4
#> 10  2011     3     3     1     2     3     4     3     5     1     3     4     3
#> 11  2012     3     3     3     1     3     3     3     5     1     1     3     3
#> 12  2013     3     3     3     1     4     3     3     4     1     2     2     3
#> 13  2014     1     2     4     2     3     2     2     3     2     3     1     1
#> 14  2015     1     1     5     2     3     2     3     5     4     5     1     2
#> 15  2016     2     2     2     3     5     2     2     1     3     5     2     2
#> 16  2017     1     2     5     4     3     1     1     4     3     3     2     1
#> 17  2018     2     3     3     5     3     1     2     2     4     3     5     1
#> # ... with 16 more variables: HR <dbl>, HU <dbl>, IE <dbl>, IT <dbl>, LT <dbl>,
#> #   LU <dbl>, LV <dbl>, MT <dbl>, NL <dbl>, PL <dbl>, PT <dbl>, RO <dbl>,
#> #   SE <dbl>, SI <dbl>, SK <dbl>, UK <dbl>
#>
#> $res$sco_level_num
#> # A tibble: 17 x 29
#>     time    AT    BE    BG    CY    CZ    DE    DK    EE    EL    ES    FI    FR
#>    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
#>  1  2002   0.5   0    -1     1     0.5   0     1     0    -0.5  -0.5   0.5     0
#>  2  2003   0.5  -0.5  -1     1     0.5   0     1     0    -0.5  -0.5   0.5     0
#>  3  2004   0     0    -1     1     0     0     1     0    -0.5   0     0.5     0
#>  4  2005   0     0    -1     1     0     0     1     0.5  -0.5   0     0.5     0
#>  5  2006   0    -0.5  -0.5   1     0     0     1     1    -0.5   0     0.5     0
#>  6  2007   0    -0.5   0     1     0     0     1     1    -0.5   0     0.5     0
#>  7  2008   0.5  -0.5   0     1     0     0.5   1     1    -0.5   0     0.5     0
#>  8  2009   0.5   0     0     1     0     1     1     0    -0.5  -0.5   0.5     0
#>  9  2010   1     0    -0.5   1     0     1     1     0    -0.5  -1     0.5     0
#> 10  2011   1     0    -0.5   0.5   0.5   1     1     0    -1    -1     1       0
#> 11  2012   1     0    -0.5   0     0.5   1     1     0.5  -1    -1     1       0
#> 12  2013   1     0    -0.5   0     0.5   1     0.5   0.5  -1    -1     0.5     0
#> 13  2014   0.5   0    -0.5   0     0.5   1     0.5   0.5  -1    -1     0.5     0
#> 14  2015   0.5   0    -0.5   0     0.5   1     0.5   1    -1    -1     0.5     0
#> 15  2016   0.5  -0.5  -0.5   0     1     1     0.5   0.5  -1    -1     0       0
#> 16  2017   0.5  -0.5   0     0     1     1     0.5   1    -1    -1     0       0
#> 17  2018   0    -0.5   0     0     1     1     0.5   1    -1    -1     0       0
#> # ... with 16 more variables: HR <dbl>, HU <dbl>, IE <dbl>, IT <dbl>, LT <dbl>,
#> #   LU <dbl>, LV <dbl>, MT <dbl>, NL <dbl>, PL <dbl>, PT <dbl>, RO <dbl>,
#> #   SE <dbl>, SI <dbl>, SK <dbl>, UK <dbl>
#>
#>
#> $msg
#> NULL
#>
#> $err
#> NULL
```

The result is a list with three components: *the summary statistics,* the numerical labels indicating the interval of the partition a level belongs to, *the interval of the partition a change belongs to.

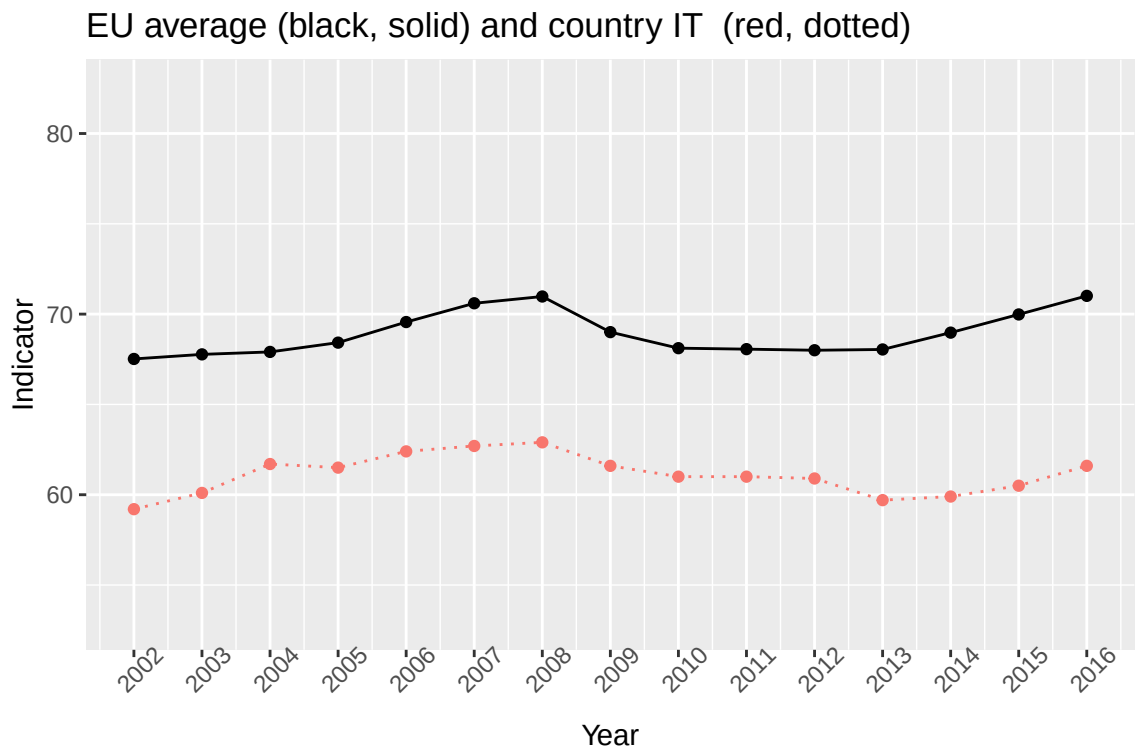
Numerical labels are assigned as follows, see (DRAFT JOINT EMPLOYMENT REPORT FROM THE COMMISSION AND THE COUNCIL 2019 :
* value 1 if a the original level or change is $y \leq m - 1 \cdot s$;
* value 2 if a the original level or change is $m - 1 \cdot s < y \leq m - 0.5 \cdot s$;

* value 3 if a the original level or change is $m - 0.5 \cdot s < y \leq m + 0.5 \cdot s$;
* value 4 if a the original level or change is $m + 0.5 \cdot s < y \leq m + 1 \cdot s$;
* value 5 if a the original level or change is $y > m + 1 \cdot s$.

We note that there is the possibility of representing the above summaries as coloured plots (TO DO) into scoreboards.

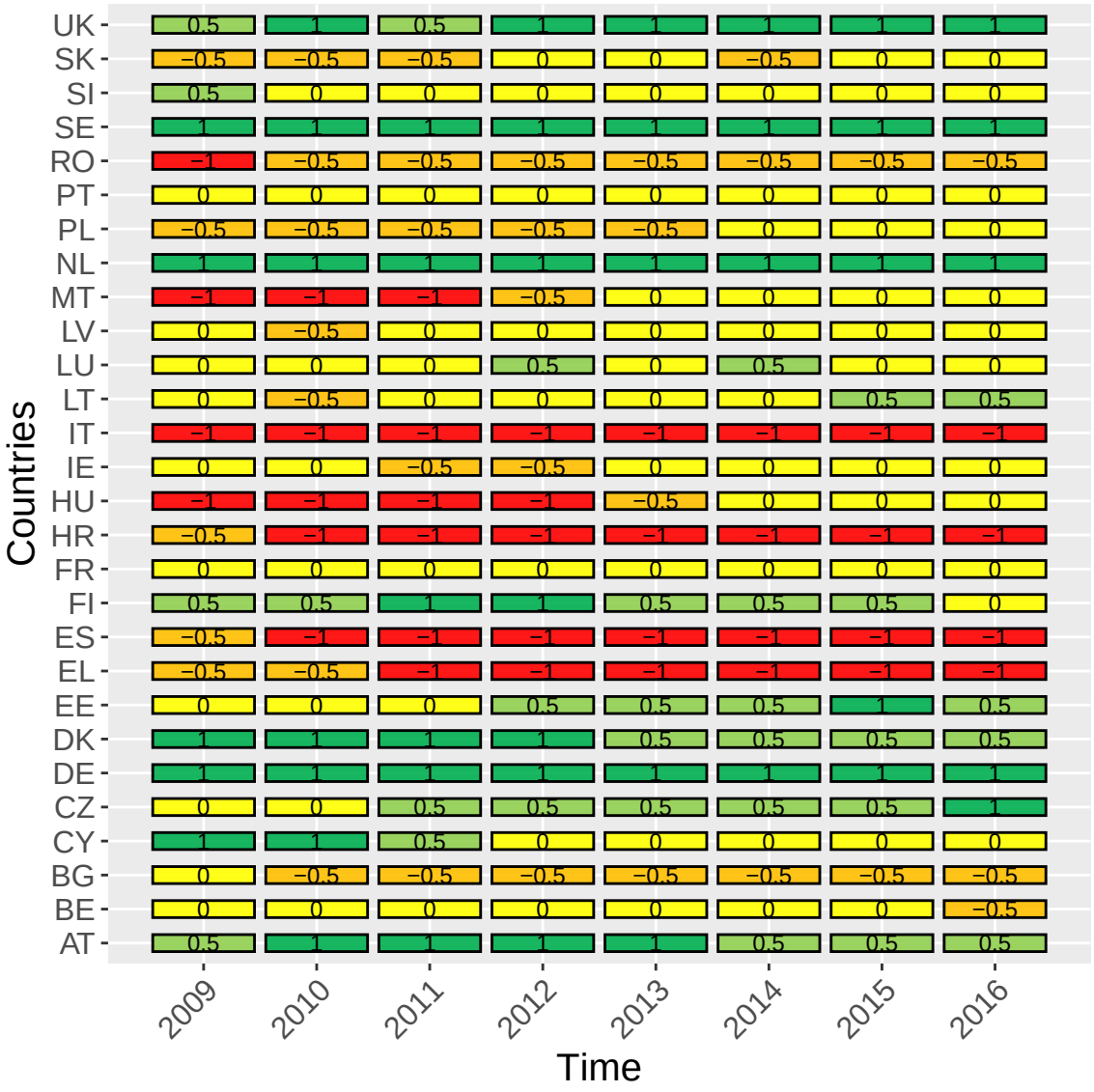
For the comparison of a country with the EU average, the following steps are recommended, from raw data:

```
# require(ggplot2)
# data(emp_20_64_MS)
selectedCountry <- "IT"
timeName <-  "time"
myx_angle <-  45

outSig <- sigma_conv(emp_20_64_MS, timeName = timeName,
           time_0=2002,time_t=2016)
miniY <- min(emp_20_64_MS[,- which(names(emp_20_64_MS) == timeName )])
maxiY <-  max(emp_20_64_MS[,- which(names(emp_20_64_MS) == timeName )])
estrattore<-  emp_20_64_MS[,timeName] >= 2002  &  emp_20_64_MS[,timeName] <= 2016
ttmp <- cbind(outSig$res, dplyr::select(emp_20_64_MS[estrattore,], -contains(timeName)))

myG2 <-
  ggplot(ttmp) + ggtitle(
  paste("EU average (black, solid) and country",selectedCountry ," (red, dotted)") )+
  geom_line(aes(x=ttmp[,timeName], y =ttmp[,"mean"]),colour="black") +
  geom_point(aes(x=ttmp[,timeName],y =ttmp[,"mean"]),colour="black") +
#        geom_line()+geom_point()+
    ylim(c(miniY,maxiY)) + xlab("Year") +ylab("Indicator") +
  theme(legend.position = "none")+
  # add countries
  geom_line( aes(x=ttmp[,timeName], y = ttmp[,"IT"],colour="red"),linetype="dotted") +
  geom_point( aes(x=ttmp[,timeName], y = ttmp[,"IT"],colour="red")) +
  ggplot2::scale_x_continuous(breaks = ttmp[,timeName],
                     labels = ttmp[,timeName]) +
   ggplot2::theme(
         axis.text.x=ggplot2::element_text(
         #size = ggplot2::rel(myfont_scale ),
         angle = myx_angle
         #vjust = 1,
         #hjust=1
         ))

myG2
```



It is also possible to graphically show departures in terms of the above defined partition:

```
obe_lvl <- scoreb_yrs(emp_20_64_MS,timeName = timeName)$res$sco_level_num
# select subset of time
estrattore <- obe_lvl[,timeName] >= 2009 & obe_lvl[,timeName] <= 2016
scobelvl <- obe_lvl[estrattore,]

my_MSstd <- ms_dynam(scobelvl,
                timeName = "time",
                displace = 0.25,
                displaceh = 0.45,
                dimeFontNum = 3,
                myfont_scale = 1.35,
                x_angle = 45,
                axis_name_y = "Countries",
                axis_name_x = "Time",
                alpha_color = 0.9
                )
#> Warning: `guides(<scale> = FALSE)` is deprecated. Please use `guides(<scale> =
#> "none")` instead.

my_MSstd
```



Country fiches

The **convergEU** package provides a function that automatically prepares one or more country fiches. The function allows the user to pass arguments to obtain information on various indicators and countries. The user specifies one key country and some other countries of interest can be listed to compare performances. Note that most arguments should be passed as strings instead of object names and that the dataset must be a complete (without missing values) tibble. Internet connection should be available when invoking the function to properly render the results.

Below is an example of a call to the function *go_ms_fi()* to illustrate the syntax. This command would create a country fiche for Germany comparing it with Italy, the UK and France for the timeframe 2002-2016. Note that most arguments are passed as strings instead of object names.

```
go_ms_fi(
    workDF ='myTB',
    countryRef ='DE',
```

```
    otherCountries = "c('IT','UK','FR')",
    time_0 = 2002,
    time_t = 2016,
    tName = 'time',
    indiType = "highBest",
    aggregation = 'EU27',
    x_angle = 45,
    dataNow = Sys.time(),
    author = 'A.Student',
    outFile = 'Germany-up2-2016',
    outDir = "/media/fred/STORE/PRJ/2018-TENDER-EU/STEP-1/bitbucketed/tt-fish",
    indiName = 'emp_20_64_MS',
    workTB = NULL
)
```

Here is a breakdown of the arguments passed in the function: * *workDF* is a string specifying the name of the working dataset that must be available in the global environment. * *countryRef* is a string determining the country (or unit) of main interest. This country will be shown in one-country plots. the short name of a member country that will be shown in one-country plots. * *otherCountries* specifies other countries that should be included in the analysis for comparison. * *time_0* specifies the starting time. * time_t specifies the end time. * *tName* is the name of the variable containing times. * *indiType* specifies whether the indicator is of type "lowBest" or "highBest" (i.e. if a low or high value is desirable for a country). * *aggregation* specifies the reference group of EU countries (e.g., 'EU27', 'EA'). If using a dataset with units other than EU Member States, then this should be 'custom'. * *x_angle* determines the axis orientation for time labels in graphs. * *dataNow* specifies the date of creation of the country fiche. *Sys.time()* will provide the exact date and time of when the function was run. * *author* specifies the author of the report, which will be shown in the fiche. * *outFile* is a string determining the name of the output file. This should not include a path. * *outDir* determines the output directory, eventually not existing (only one level allowed). * *indiName* is a string determining how the name of the considered indicator will appear in the fiche. * *workTB* is the name of a tibble containing data, optional, as an alternative to a global object.

Of particular importance the argument *outFile* that can be a string indicating the name of the output file. Similarly, *outDir* is the path (unit and folders) in which the final compiled html will be stored. The sintax of the path depend on the operating system; for example *outDir='F:/analysis/IT2018'* indicates that in the usb disk called 'F', within the folder 'analysis' is located folder 'IT2018' where R will write the country fiche. Note that a disk called 'F' must exist and also folder 'analysis' must exist in such unit, while on the contrary folder 'IT2018' is created by the function if it does not already exist.

Within the above mentioned output directory, besides the compiled HTML, a file called as specified in *outFile* is also stored, but with added the string '-workspace.RData' which contains data and plots produced during the compilation of the country fiche for further subsequent use in other technical reports.

Indicator fiches

The function *go_indica_fi()* allows the user to create indicator fiches in the form of an HTML or PDF file. Note that most arguments should be passed as strings instead of object names and that the dataset must be a complete (without missing values) tibble. Internet connection should be available when invoking the function to properly render the results.

An example of syntax to invoke the procedure is:

```
go_indica_fi(
    time_0 = 2005,
```

```
    time_t = 2010,
    timeName = 'time',
    workDF = 'emp_20_64_MS' ,
    indicaT = 'emp_20_64',
    indiType = c('highBest','lowBest')[1],
    seleMeasure = 'all',
    seleAggre = 'EU28',
    x_angle =  45,
    data_res_download =  FALSE,
    auth = 'A.Student',
    dataNow =  '2019/05/16',
    outFile = "test_IT-emp_20_64_MS",
    outDir = "/media/fred/STORE/PRJ/2018-TENDER-EU/STEP-1/bitbucketed/tt-fish",
    pdf_out = FALSE,
    workTB = NULL,
    selfContained = FALSE,
    eige_layout = FALSE,
    memStates = 'quintiles'# ('quintiles', 'default', 'custom')
)
  )
```

Here is a breakdown of the arguments passed in the function: * *time_0* specifies the starting time. * *time_t* specifies the end time. * *timeName* is the name of the variable containing times. * *workDF* is a string specifying the name of the working dataset that must be available in the global environment. * *indicaT* is a string determining how the name of the considered indicator will appear in the fiche. * *indiType* specifies whether the indicator is of type "lowBest" or "highBest" (i.e. if a low or high value is desirable for a country). * *seleMeasure* determines which measures of convergence will be calculated. This is a subset of the following collection of strings: "all", beta","delta","gamma","sigma". If uncertain, we recommend using"all", as it is a shortcut for the whole set. * *seleAggre* specifies the set of EU countries (e.g., 'EU27', 'EU19') for which the analysis should be run. If using a dataset with units other than EU Member States, then this should be 'custom'. * *x_angle* determines the axis orientation for time labels in graphs, the default is 45. * *data_res_download* determines whether the data and results should be downloaded, the default is FALSE. * *author* specifies the author of the report, which will be shown in the fiche. * *auth* specifies the author of the report, which will be shown in the fiche. The default is 'A.Student'. * *dataNow* specifies the date of creation of the indicator fiche. The default is the current date and time. * *outFile* is a string determining the name of the output file. This should not include a path. * *outDir* determines the output directory, eventually not existing (only one level allowed, in other words it cannot create a folder and a sub-folder, the folder should already exist and the subfolder will be created if specified in the path). * *pdf_out* lets the user choose whether to create the fiche as an HTML or a PDF. The default is FALSE. If passed as TRUE, then the fiche will be created also as a PDF file. * *workTB* is the name of a tibble containing data, optional, as alternative to a global object. * *selfContained* should be set to TRUE if just one file is desired. * *eige_layout* should be set to TRUE if the EIGE (European Institute for Gender Equality) layout is desired. * *memStates* determines what kinds of visualisations and analyses are included in the fiche when comparing units. There are three options:"default","quintiles", and"custom".

- "Default" includes an exploration of how countries' standard deviations changed throughout the timeframe.This is the original charts contained in version 0.5.1.
- "Quintiles", which is the selected option, sorts the Member States into quintiles based on the values of the indicator for each measured time in the timeframe. The evolution of the position of Member States can be tracked, and three maps are generated: the first depicting the quintile groupings at the start time, the second depicting the quintile groupings at the end time, and the third depicting the change between the start and the end time for each Member State's quintile group (e.g., if Hungary was in the first quintile in 2007 and the third in 2020, then that would be a change of +2 quintiles).

- "Custom" should be chosen if a custom dataset containing units, for example regions, that are not compatible with the first two kinds of analyses was used. If "custom" is chosen, a .csv file with quintiles groupings will be created in the output folder along with the fiche.

References

Below the main references are listed:

- Eurofound (2017), Towards a conceptual framework to monitor convergence in the European Union - evidence from desk research and expert workshop; by Massimiliano Mascherini, Martina Bisello, Hans Dubois, Franz Eiffe (Eurofound) in cooperation with ICF Consulting Services and Irene Rioboo Leston (URJC, Spain).

- Eurofound (2018), Upward convergence in the EU: Concepts, measurements and indicators, Publications Office of the European Union, Luxembourg; by: Massimiliano Mascherini, Martina Bisello, Hans Dubois and Franz Eiffe.