# Multiprocessing NBB-customized Temporal Disaggregation of time series

Corentin Lemasson

**Abstract**

The vignette explains how to perform temporal disaggregation (TD) of an annual to quarterly or monthly set of time series using the package *nbbTD*, which relies on R packages from JDemetra+. Either the sum or the average consistency between the annual benchmarks and the quarterly or monthly indicators are handled. The package focuses on the (enhanced) Denton PFD method using state space methodology. State space methodolgy helps to significantly speed up the process and expresses Denton as a statistical model. Considering this approach, more flexibility can be added to the original model. This includes the possibility to incorporate outliers (level shifts) in the Benchmark-to-Indicator (BI) ratio and change the infra-annual BI ratio (and therefore also the disaggregated series) manually for some periods. Hence, the main advantage of movement preservation is kept while some important drawbacks of Denton PFD as TD method, such as the high sensitivity to outliers and the lack of flexibility, can now be taken care of. When enhanced Denton (eDenton) is used, a procedure was developed to automatically select the forecasting method of the annual BI ratio but the forecasts can also be changed manually if necessary. Although the package was intended for temporal disaggregation, (e)Denton PFD method could be used for benchmarking too. In addition to (e)Denton PDF method, the traditional Chow-Lin method and its variants (Fernandez and Litterman) are also available in the package. In addition to the numerical output, a Shiny app is provided inside the package to visualize the results and help the user to make the best decision concerning the specification of the model for each series.

## Contents

## Overview

The package was conceived as a tool for the production of temporal disaggregated series. It handles annual to quarterly or monthly disaggregation and supports either sum or average consistency between the annual benchmarks and the infra-annual indicator. As far as methods are concerned, the Denton proportional first differences (PFD) method as well as the traditional Chow-Lin method and its variants Fernandez and Litterman are currently included in the package. The Denton PFD method can be applied as such or it can be enhanced in extrapolation by forecasting the development of the annual Benchmark-to-Indicator (BI) ratio for the year with no benchmark. When Denton or eDenton method is used, the user also has de possibility to include outliers and modify the infra-annual BI ratio (therefore also indirectly the disaggregated series) manually.

The BI ratio is the ratio between the benchmark and the indicator series. Annually, it corresponds to the ratio between the annual benchmark and the sum (or mean) of the four quarters of the indicator (assuming indicator is quarterly). On an infra-annual basis, it is the ratio between the disaggregated series and the indicator. It is interesting to monitor it closely as the annual movements of the BI ratio reflect the fit between the benchmark and the indicator. A constant BI ratio means that the growth rates of the benchmark and the indicator are similar while large deviations indicates different patterns between the two series. A systematic analysis of the annual BI ratio is also useful to detect anomalies in the indicator or the benchmark series. Often, in normal times, it can be argued that the infra-annual BI ratio should not change drastically from one period to another. When this is true, the Denton PFD method justifies itself by definition.

By keeping the quarter-on-quarter growth rates of the indicator unchanged in extrapolation, the Denton PFD method makes an implicit forecast of the annual BI ratio solely based on the development of the last few annual BI ratio's. On the other hand, the enhanced Denton method requires to make an explicit forecast of the annual BI ratio. In the package, the user can either use the automatic forecast procedure or provide the forecasts manually. The automatic procedure works as such: given that the number of observations is often limited, time series models may be unstable. The choice was made to consider a random walk process (i.e. the last annual BI ratio) by default and to deviate from this assumption only when there is strong statistical evidence against it. A score is calculated as the difference in RMSE of cross-validation errors between the best alternative model and a random walk process. The score is then compared by default to a 95%-quantile critical value obtained by simulation. The alternative models considered are the average BI ratio of the last five years (three if the series is short), the geometric average of the growth rates of the BI ratio in the last five years (three if the series is short) and TRAMO when the benchmark series is long enough (>= 13 years). Note that a selection of model simply based on the comparison of the score is not advisable with small sample size as it is likely that an alternative model displays a better score only by chance and not because it fits the data better.

To translate the annual forecast of the BI ratio to the infra-annual extrapolated periods, we use the simplified algorithm from IMF (reference: *On the extrapolation with the Denton Proportional Benchmarking method*, p.8) when the extrapolation period does not cover the entire year yet. When the entire year is covered, an extra year is implicitly added to the benchmark series by using the forecast of the annual BI ratio and the Denton PFD method is run considering this extra year for the benchmark. A combination of the two approaches is used when the extrapolated period exceeds one year (with a maximum of two years of extrapolation).

Chow-Lin method and its variants is an all-in one solution largely documented in the literature and it won't be discussed in too much details here. Just to mention that the method is applied as such for both distribution and extrapolation and numerical results are returned in details in the output. As with Denton, graphical analysis are also available.

Traditional regression-based methods such as Chow-Lin/Fernandez have shown their limits during the Covid-19 crisis. In particular, they tend to provide disaggregated series which are excessively smoothed when the assumptions of strict exogeneity or homoscedasticity does not hold. Moreover, they can lead to large revisions of the entire series and to some issues of comparability as the smoothness of the disaggregated series depends on the fit between the benchmark and the indicator. Denton PFD method, however, preserves as much as possible the (quarter-on-quarter) movement of the indicator. Movement preservation should be the goal of any temporal disaggregation when the infra-annual indicator is the only information available and there is no relationship between the BI ratio and the indicator (or, in other words, when there is no reason to believe that the variability of the indicator is higher than that of the benchmark). However, when this is not the case, models including a constant such as Chow-Lin can be more appropriate, but the assumptions underlying the model should be scrutinized.

Denton PFD method tends to keep the BI ratio (i.e. the ratio between the benchmark and the indicator) as constant as possible given the annual constraints. It is usually expressed in mathematical terms as a constrained minimization problem:

$$min_{y_t} \sum_{t=2}^{n} \left[ \frac{y_t}{x_t} - \frac{y_{t-1}}{x_{t-1}} \right]^2$$

subject to

$$\sum_t y_t = Y_y$$

Equivalently, the Denton PFD method can also be expressed as a statistical model considering the following state space representation

$$y_t = \beta_t x_t$$
$$\beta_{t+1} = \beta_t + \varepsilon_t \qquad \varepsilon_t \sim \mathsf{NID}(0, \sigma^2_\varepsilon)$$

where the annual constraint are taken care of by replacing the original series $y_t$ by a cumulated series $y^c_t$. Hence, the value of the last quarter/month of the year is observed and corresponds to the benchmark. The value of the other quarters are initially defined as missing and estimated by maximum likelihood. From there, the uncumulated disaggregated series is easily derived. Let's illustrate this process with an example from the dataset *nbb_data* available in the package.

Table 1: Denton PFD in state space: belgian transport industry

| year | quarter | yc | yc_est | y_est |
|---|---|---:|---:|---:|
| 2017 | Q1 | NA | 5385.7 | 5385.7 |
| 2017 | Q2 | NA | 10980.6 | 5594.9 |
| 2017 | Q3 | NA | 16270.5 | 5290.0 |
| 2017 | Q4 | 22064.7 | 22064.7 | 5794.2 |
| 2018 | Q1 | NA | 5478.3 | 5478.3 |
| 2018 | Q2 | NA | 11148.1 | 5669.8 |
| 2018 | Q3 | NA | 16930.3 | 5782.2 |
| 2018 | Q4 | 23056.6 | 23056.6 | 6126.3 |
| 2019 | Q1 | NA | 5811.5 | 5811.5 |
| 2019 | Q2 | NA | 11701.6 | 5890.0 |
| 2019 | Q3 | NA | 17518.8 | 5817.3 |
| 2019 | Q4 | 23864.3 | 23864.3 | 6345.5 |
| 2020 | Q1 | NA | 5862.2 | 5862.2 |
| 2020 | Q2 | NA | 11065.2 | 5202.9 |
| 2020 | Q3 | NA | 16528.0 | 5462.8 |
| 2020 | Q4 | 22532.1 | 22532.1 | 6004.1 |

The state space representation of Denton PFD method makes it possible to enhance the original model by adding new features. An argument against the use of Denton PFD as temporal disaggregation method is its locally high sensibility to outliers inducing wave effects. From the state space representation above, one could increase some value of $\sigma^2_\varepsilon$ to consider a higher error of $\beta_t$, which means adding a level shift in the infra-annual BI ratio (as $\beta_t = \frac{y_t}{x_t} = BI_t$) at a given point of time. The intensity of the outlier can be modified by playing with the value of $\sigma^2_\varepsilon$.

The infra-annual BI ratio (thus, also, indirectly the disaggregated series) can be adapted manually. This offers flexibility to the user and could or not be used in combination with outliers. Implementing this possibility was straightforward given the state space representation. From the example 1 above, the missing values of the first quarters/months of the year are just filled manually before estimating the rest of the series. Two important points need to be mention here: 1) when manual BI ratio are provided, they must be provided for each quarter/month of the year. 2) the weighted average of the infra-annual BI ratio manually imported must match the annual BI ratio. When this not the case, the difference is prorated and a warning is returned to the user.

It is advised to add outliers and make manual adaptation of the infra-annual BI ratio with parsimony only when there are strong arguments for doing so.

## Data

The package contains a list of two data sets called **nbb_data**. The first data set 'B1G_Y_data' includes three benchmark series and the second data set 'TURN_Q_data' includes the corresponding quarterly indicators. The benchmark series are the Belgian annual value added on the period 2009-2020 in chemical industry (CE), construction (FF) and transport services (HH) as published by National Accounts. The corresponding indicator series are (modified) production indicators derived from VAT statistics and covering the period 2009Q1-2021Q4. Those data are used as examples to illustrate the various functions.

The package also contains a data set called **table_rw** which contains the critical values to consider for the selection of the forecasting model of the annual BI ratio when enhanced Denton is used as temporal disaggregation model.

## Quick start

Install the package using the "–no-multiarch" option:

```
install.packages("xxx/nbbTD_0.1.0.tar.gz", repos = NULL, type = "source", INSTALL_opts="--no-multiarch")
```

Once the package installed, there are five steps to follow to run the main function:

1. Load the package
2. Load the benchmark and the indicator data as *data.frame* or *ts* object. Keep the series in the same order, one indicator by benchmark. Use any constant or NA's when no indicator is available. **The same column names (the aggregate's names) must be defined in both tables**.
3. Set config. Configuration must be set either by using the function **setConfig_default** or **setConfig_FromXLSX**. **setConfig_default** has default values but some or all of them can be adapted. Config can also be imported directly from Excel (see section 'Configuation').
4. Apply main function and stock the results into a variable
5. Display the results

```
# Example of the procedure to follow

## Step 1
suppressMessages(suppressWarnings(library(nbbTD)))

## Step 2
data("nbb_data")
benchmarks <- nbb_data$B1G_Y_data
indicators <- nbb_data$TURN_Q_data
colnames(benchmarks) <- colnames(indicators) <- c("DATE", "CE", "FF", "HH")
```

```
## Step 3
myconfig <- setConfig_default() # config by default, use ?setConfig_default for more information
# or alternatively:
# myconfig <- setConfig_FromXLSX(file = "xxx.xlsx")

## Step 4
res <- multiTD(benchmarks, indicators, myconfig) # use ?multiTD for more information over the optional 

## Step 5
print(res) # overview of the results
# res$eDenton$bi.infra # use '$' sign to select the output you're interested in
runShiny(res) # Visualization of the results
```

## Description of the functions

To list all functions available in the package;

```
ls("package:nbbTD")
#>   [1] "%>%"                                        "%between%"                                  "%ch
#>   [4] "%flike%"                                    "%ilike%"                                    "%in
#>   [7] "%like%"                                     ":="                                         "a"
#>  [10] "absolutePanel"                              "accuracy"                                   "Acf
#>  [13] "actionButton"                               "actionLink"                                 "add
#>  [16] "add.equation"                               "add_f_manual"                               "add
#>  [19] "addFilter"                                  "addResourcePath"                            "add
#>  [22] "addStyle"                                   "addWorksheet"                               "agg
#>  [25] "aggregation"                                "alloc.col"                                  "ani
#>  [28] "appendTab"                                  "ar"                                         "ar2
#>  [31] "arfima"                                     "arima"                                      "Ari
#>  [34] "arima.errors"                               "arimaorder"                                 "arm
#>  [37] "as.data.table"                              "as.Date.yearmon"                            "as.
#>  [40] "as.ITime"                                   "as.shiny.appobj"                            "as.
#>  [43] "auto.arima"                                 "autocorrelations"                           "aut
#>  [46] "autocorrelations.partial"                   "autolayer"                                  "aut
#>  [49] "baggedETS"                                  "baggedModel"                                "bas
#>  [52] "bats"                                       "between"                                    "biz
#>  [55] "bld.mbb.bootstrap"                          "bookmarkButton"                             "boo
#>  [58] "bootstrapPage"                              "bowmanshenton"                              "Box
#>  [61] "BoxCox.lambda"                              "br"                                         "bro
#>  [64] "brushedPoints"                              "brushOpts"                                  "cal
#>  [67] "calc_cv_tramo"                              "calc_fit_growth"                            "Cal
#>  [70] "CalcCVRMSE"                                 "CalcCVRMSEOfSelectedModels"                 "Cal
#>  [73] "calculate_pred_score"                       "callModule"                                 "cap
#>  [76] "Ccf"                                        "cdfChi2"                                    "cdf
#>  [79] "cdfInverseGamma"                            "cdfInverseGaussian"                         "cdf
#>  [82] "check_compatibility_model"                  "check_df_structure"                         "che
#>  [85] "check_time_range"                           "checkboxGroupInput"                         "che
#>  [88] "checkresiduals"                             "chgroup"                                    "chm
#>  [91] "cholette"                                   "chorder"                                    "CJ"
#>  [94] "clickOpts"                                  "cloneWorksheet"                             "cod
#>  [97] "column"                       "conditionalFormat"                    "con
#> [100] "conditionalPanel"             "conditionStackTrace"                  "con
#> [103] "convertFromExcelRef"          "convertToDate"                        "con
#> [106] "copy"                         "copyWorkbook"                         "cre
#> [109] "createNamedRegion"            "createRenderFunction"                 "cre
#> [112] "createWebDependency"          "createWorkbook"                       "cro
#> [115] "cube"                         "cubicspline"                          "cub
#> [118] "cumul"                        "CV"                                   "CVa
#> [121] "cycle"                        "data.table"                           "dat
#> [124] "dataValidation"               "date_decimal"                         "dat
#> [127] "dateRangeInput"               "dblclickOpts"                         "dca
#> [130] "dcast.data.table"             "debounce"                             "dec
#> [133] "deleteData"                   "densityChi2"                          "den
#> [136] "densityInverseGamma"          "densityInverseGaussian"               "den
#> [139] "denton"                       "denton_customized"                    "den
#> [142] "diagnostics"                  "dialogViewer"                         "dic
#> [145] "diskCache"                    "div"                                  "dm.
#> [148] "doornikhansen"                "downloadButton"                       "dow
#> [151] "downloadLink"                 "dshw"                                 "eas
#> [154] "em"                           "enableBookmarking"                    "equ
#> [157] "estimate"                     "ets"                                  "eve
#> [160] "exportTestValues"             "exprToFunction"                       "Ext
#> [163] "extractStackTrace"            "extrapolate_infra_annual_BI_ratio"    "fas
#> [166] "fast.tramoseats"              "fcase"                                "fco
#> [169] "fifelse"                      "fileInput"                            "fil
#> [172] "fillPage"                     "fillRow"                              "fil
#> [175] "filteredstatesstdev"          "filteringstates"                      "fil
#> [178] "findfrequency"                "fintersect"                           "fir
#> [181] "fixedPage"                    "fixedPanel"                           "fix
#> [184] "flowLayout"                   "fluidPage"                            "flu
#> [187] "forecast"                     "forecast.ets"                         "For
#> [190] "ForecastSeries"               "formatStackTrace"                     "fou
#> [193] "fourierf"                     "foverlaps"                            "fra
#> [196] "frankv"                       "fread"                                "fre
#> [199] "freezeReactiveVal"            "freezeReactiveValue"                  "fro
#> [202] "frollmean"                    "frollsum"                             "fse
#> [205] "fsetequal"                    "fsort"                                "fun
#> [208] "fwrite"                       "geom_forecast"                        "Geo
#> [211] "get_perc_mean_in_CI"          "getBaseFont"                          "get
#> [214] "getCreators"                  "getCurrentOutputInfo"                 "get
#> [217] "getDefaultReactiveDomain"     "getDTthreads"                         "get
#> [220] "getNumericRounding"           "getQueryString"                       "get
#> [223] "getSheetNames"                "getShinyOption"                       "get
#> [226] "getTables"                    "getUrlHash"                           "ggA
#> [229] "ggCcf"                        "gghistogram"                          "ggl
#> [232] "gglagplot"                    "ggmonthplot"                          "ggP
#> [235] "ggseasonplot"                 "ggsubseriesplot"                      "ggt
#> [238] "ggtaperedpacf"                "ggtsdisplay"                          "gme
#> [241] "groupColumns"                 "groupingsets"                         "gro
#> [244] "grp"                          "h1"                                   "h2"
#> [247] "h3"                           "h4"                                   "h5"
#> [250] "h6"                           "haskey"                               "hea
#> [253] "helpText"                     "hideTab"                              "hol
#> [256] "hour"                          "hoverOpts"                              "hr"
#> [259] "HTML"                          "htmlOutput"                             "htm
#> [262] "hw"                            "icon"                                   "IDa
#> [265] "imageOutput"                   "img"                                    "inc
#> [268] "includeHTML"                   "includeMarkdown"                        "inc
#> [271] "includeText"                   "incProgress"                            "ind
#> [274] "inputPanel"                    "inrange"                                "ins
#> [277] "insertPlot"                    "insertTab"                              "ins
#> [280] "installExprFunction"           "int2col"                                "inv
#> [283] "InvBoxCox"                     "is.acf"                                 "is.
#> [286] "is.baggedModel"                "is.bats"                                "is.
#> [289] "is.convertible.to.date"        "is.data.table"                          "is.
#> [292] "is.forecast"                   "is.key_missing"                         "is.
#> [295] "is.modelAR"                    "is.nnetar"                              "is.
#> [298] "is.reactive"                   "is.reactivevalues"                      "is.
#> [301] "is.singleton"                  "is.splineforecast"                      "is.
#> [304] "is_colnames_identical"         "isolate"                                "iso
#> [307] "isRunning"                     "isTruthy"                               "jar
#> [310] "key"                           "key_missing"                            "key
#> [313] "last"                          "library.dynam.unload"                   "lik
#> [316] "likelihood"                    "ljungbox"                               "loa
#> [319] "loading_cyclical"              "loading_periodic"                       "loa
#> [322] "loadSupport"                   "loadWorkbook"                           "loc
#> [325] "locallineartrend"              "ma"                                     "mai
#> [328] "makeHyperlinkString"           "makeReactiveBinding"                    "mar
#> [331] "maskReactiveContext"           "mday"                                   "mea
#> [334] "melt"                          "melt.data.table"                        "mem
#> [337] "merge.data.table"              "mergeCells"                             "min
#> [340] "modalButton"                   "modalDialog"                            "mod
#> [343] "modelAR"                       "modifyBaseFont"                         "mon
#> [346] "monthdays"                     "msae"                                   "msa
#> [349] "msae3"                         "msignal"                                "mst
#> [352] "msts"                          "multiTD"                                "mul
#> [355] "multiTDDenton"                 "multivariatecholette"                   "na.
#> [358] "nafill"                        "naive"                                  "nav
#> [361] "navbarPage"                    "navlistPanel"                           "nbb
#> [364] "ndiffs"                        "nearPoints"                             "nee
#> [367] "next_period"                   "nnetar"                                 "noi
#> [370] "NS"                            "ns.sep"                                 "nsd
#> [373] "numericInput"                  "observe"                                "obs
#> [376] "ocsb.test"                     "onBookmark"                             "onB
#> [379] "onFlush"                       "onFlushed"                              "onR
#> [382] "onRestore"                     "onRestored"                             "onS
#> [385] "onStop"                        "openXL"                                 "out
#> [388] "p"                             "Pacf"                                   "pag
#> [391] "pageSetup"                     "pageWithSidebar"                        "pan
#> [394] "parseQueryString"              "passwordInput"                          "per
#> [397] "plot.nbb.dsc.td.multiproc.output" "plot.nbb.dsc.td.multiproc.output.clvar" "plo
#> [400] "plot_bi_compare_manual"        "plot_bi_infra"                          "plo
#> [403] "plot_td_compare_manual"        "plot_td_series"                         "Plo
#> [406] "plotOutput"                    "plotPNG"                                "pre
#> [409] "prependTab"                    "print.nbb.dsc.td.multiproc.output"      "pri
#> [412] "printStackTrace"               "Progress"                               "pro
#> [415] "protectWorksheet"                              "quarter"                                     "rad
#> [418] "randomsChi2"                                   "randomsGamma"                                "ran
#> [421] "randomsInverseGaussian"                        "randomsT"                                    "rbi
#> [424] "reactive"                                      "reactiveFileReader"                          "rea
#> [427] "reactivePoll"                                  "reactivePrint"                               "rea
#> [430] "reactiveText"                                  "reactiveTimer"                               "rea
#> [433] "reactiveVal"                                   "reactiveValues"                              "rea
#> [436] "reactlog"                                      "reactlogReset"                               "rea
#> [439] "read.xlsx"                                     "readWorkbook"                                "reg
#> [442] "registerInputHandler"                          "remainder"                                   "rem
#> [445] "removeColWidths"                               "removeComment"                               "rem
#> [448] "removeInputHandler"                            "removeModal"                                 "rem
#> [451] "removeResourcePath"                            "removeRowHeights"                            "rem
#> [454] "removeTable"                                   "removeUI"                                    "rem
#> [457] "renameWorksheet"                               "renderCachedPlot"                            "ren
#> [460] "renderImage"                                   "renderPlot"                                  "ren
#> [463] "renderTable"                                   "renderText"                                  "ren
#> [466] "repeatable"                                    "ReplaceNAColByCst"                           "rep
#> [469] "req"                                           "resourcePaths"                               "res
#> [472] "result"                                        "rleid"                                       "rle
#> [475] "rollup"                                        "rowid"                                       "row
#> [478] "runApp"                                        "runExample"                                  "run
#> [481] "runGist"                                       "runGitHub"                                   "run
#> [484] "runUrl"                                        "rwf"                                         "sae
#> [487] "safeError"                                     "sarima"                                      "sav
#> [490] "seasadj"                                       "seasonal"                                    "sea
#> [493] "seasonaldummy"                                 "seasonaldummyf"                              "sea
#> [496] "seats.decompose"                               "second"                                      "sel
#> [499] "selectizeInput"                                "serverInfo"                                  "ses
#> [502] "set"                                           "setalloccol"                                 "set
#> [505] "setBookmarkExclude"                            "setcolorder"                                 "set
#> [508] "setConfig_default"                             "setConfig_FromXLSX"                          "set
#> [511] "setDT"                                         "setDTthreads"                                "set
#> [514] "setHeader"                                     "setHeaderFooter"                             "set
#> [517] "setindexv"                                     "setkey"                                      "set
#> [520] "setLastModifiedBy"                             "setnafill"                                   "set
#> [523] "setNumericRounding"                            "setorder"                                    "set
#> [526] "setProgress"                                   "setRowHeights"                               "set
#> [529] "sheets"                                        "sheetVisibility"                             "she
#> [532] "sheetVisible"                                  "sheetVisible<-"                              "shi
#> [535] "shinyApp"                                      "shinyAppDir"                                 "shi
#> [538] "shinyOptions"                                  "shinyServer"                                 "shi
#> [541] "shouldPrint"                                   "showBookmarkUrlModal"                        "sho
#> [544] "showModal"                                     "showNotification"                            "sho
#> [547] "showTab"                                       "sidebarLayout"                               "sid
#> [550] "signal"                                        "sim_rw"                                      "sim
#> [553] "sindexf"                                       "singleton"                                   "siz
#> [556] "SJ"                                            "sliderInput"                                 "smo
#> [559] "smoothedstatesstdev"                           "snaive"                                      "sna
#> [562] "snapshotPreprocessInput"                       "snapshotPreprocessOutput"                    "spa
#> [565] "spec_tramo_default"                            "spec_tramoseats_default"                     "spl
#> [568] "splitLayout"                                   "ssf"                                         "Sta
#> [571] "statisticaltest"                               "stlf"                                        "stl
#> [574] "stopApp"                                    "strong"                                     "sts
#> [577] "sts.forecast"                               "sts.outliers"                               "sts
#> [580] "submitButton"                               "summary.nbb.dsc.td.multiproc.output"        "sum
#> [583] "summary.nbb.dsc.td.multiproc.output.denton" "suppressDependencies"                       "sys
#> [586] "table_rw"                                   "tableOutput"                                "tab
#> [589] "tabPanel"                                   "tabsetPanel"                                "tag
#> [592] "tagAppendAttributes"                        "tagAppendChild"                             "tag
#> [595] "tagGetAttribute"                            "tagHasAttribute"                            "tag
#> [598] "tags"                                       "tagSetChildren"                             "tap
#> [601] "taperedpacf"                                "tbats"                                      "tba
#> [604] "td"                                         "temporaldisaggregation"                     "tem
#> [607] "terror"                                     "test.data.table"                            "tes
#> [610] "testofupdownruns"                           "textAreaInput"                              "tex
#> [613] "textOutput"                                 "thetaf"                                     "thr
#> [616] "timetaken"                                  "titlePanel"                                 "tra
#> [619] "tramo.forecast"                             "tramo.outliers"                             "tra
#> [622] "tramoseats"                                 "tramoseats.refresh"                         "tra
#> [625] "trendcycle"                                 "truelength"                                 "tsc
#> [628] "tsCV"                                       "tsdisplay"                                  "tsl
#> [631] "tsoutliers"                                 "tstrsplit"                                  "uiO
#> [634] "ungroupColumns"                             "ungroupRows"                                "uni
#> [637] "update.dev.pkg"                             "updateActionButton"                         "upd
#> [640] "updateCheckboxInput"                        "updateDateInput"                            "upd
#> [643] "updateNavbarPage"                           "updateNavlistPanel"                         "upd
#> [646] "updateQueryString"                          "updateRadioButtons"                         "upd
#> [649] "updateSelectizeInput"                       "updateSliderInput"                          "upd
#> [652] "updateTextAreaInput"                        "updateTextInput"                            "upd
#> [655] "updateVarSelectizeInput"                    "urlModal"                                   "val
#> [658] "ValidateAndFormatBIqManual"                 "ValidateAndFormatBIqOutliers"               "val
#> [661] "varloading"                                 "varlocallevel"                              "var
#> [664] "varnoise"                                   "varreg"                                     "var
#> [667] "varSelectInput"                             "varSelectizeInput"                          "ver
#> [670] "verticalLayout"                             "warn_fit_bi"                                "wda
#> [673] "week"                                       "wellPanel"                                  "wit
#> [676] "withMathJax"                                "withProgress"                               "wit
#> [679] "withTags"                                   "worksheetOrder"                             "wor
#> [682] "write.xlsx"                                 "writeComment"                               "wri
#> [685] "writeDataTable"                             "writeFormula"                               "yda
#> [688] "year"
```

Use help('name of the functions') or ¿name of the functions' for more information and examples over the function.

## Configuration

An object of class 'nbb.dsc.td.multiproc.config' is required as input parameter in functions **multiTD**, **multiTDDenton** and **multiTDCLvar**. It can be created by using either the function **setConfig_default** or **setConfig_FromXLSX**. See ?setConfig_default for detailed information about the structure of the input. Use **setConfig_FromXLSX** to import input directly from an Excel file. The latter should be created beforehand and includes four distrinct worksheets named 'model', 'f_bi_manual', 'bi_q_outliers' and 'bi_q_manual'. See ?setConfig_default (or read below) to know how to fill each worksheet.

The config object aims at gathering four important input to be submitted to **multiTD** function:

- *model*: the model to use for each series. Currently, 'edenton', 'denton', 'chow-lin', 'fernandez', 'litterman' are handled. ‘edenton’ is used by default. The input data should be composed of two columns called ‘series_name’ and ‘model’. At least one row must be filled. You can use the keyword ‘*ALL*’ if you intend to use the same model for all the series.

- *f_bi_manual*: the manual forecasts of the annual BI ratio for the years T+1 and T+2. The input data should be composed of three columns called ‘series_name’, ‘f_T1’ and ‘f_T2’. The table may be kept empty or filled only for some series when the user wants to keep the automatic forecasts for the other series.
- *bi_q_outliers*: the outlier(s) (level shift) to consider in the infra-annual BI ratio. The input data should be composed of four columns called ‘series_name’, ‘year’, ‘quarter_month’ and ‘intensity’. The table may remain empty if no outlier needs to be defined. The column ‘quarter_month’ must be filled with values from 1 to 4 for quarterly data and with values from 1 to 12 for monthly data. The column ‘intensity’ allows the user to set up the intensity of the outlier. The default value is 10 and a value of 1 means a normal data point. Note that the scale between 1 to 10 is not linear but exponential.
- *bi_q_manual*: the manual infra-annual BI ratio to consider. The input data should be composed of 14 columns called ‘series_name’, ‘year’, ‘q1_m1’, . . . , ‘q4_m4’, ‘m5’, . . . ‘m12’. The table may remain empty if no manual BI ratio needs to be defined. The columns ‘m5’ to ‘m12’ must remain empty in case of quarterly data. Keep in mind that 1) when manual BI ratio are provided, they must be provided for each quarter/month of the year. 2) the weighted average of the manual infra-annual BI ratio must match the value of the annual BI ratio. When this not the case, the difference is prorated and a warning is returned to the user.

```r
## Example
model <- data.frame (series_name  = c("a","b","c"),
                     model = c("edenton","chow-lin","denton"))
f_bi_manual <- data.frame (series_name  = c("a","b","c"),
                           f_T1 = c(0.8,0.5,10),
                           f_T2 = c(0.82,0.5,9.9))
bi_q_outliers <- data.frame (series_name  = c("a","b","c"),
                             year = c(2020,2020,2016),
                             quarter_month = c(2,2,4),
                             intensity = c(10,10,10)) # keep intensity=10 by default
bi_q_manual <- data.frame(series_name  = c("a","b","c"),
                          year = c(2020,2020,2016),
                          q1_m1 = c(0.8,0.5,10),
                          q2_m2 = c(0.7,0.4,8),
                          q3_m3 = c(0.84,0.55,11),
                          q4_m4 = c(0.85,0.54,11), # all quarters/months of the year must be provided
                          m5  = c(NA,NA,NA), # fill with NA from here on for quarterly data
                          m6  = c(NA,NA,NA),
                          m7  = c(NA,NA,NA),
                          m8  = c(NA,NA,NA),
                          m9  = c(NA,NA,NA),
                          m10  = c(NA,NA,NA),
                          m11  = c(NA,NA,NA),
                          m12  = c(NA,NA,NA))

myconfig <- setConfig_default(model, f_bi_manual, bi_q_outliers, bi_q_manual)
myconfig
#> $model
#>   series_name    model
#> 1           a  edenton
#> 2           b chow-lin
#> 3           c   denton
#>
```

```
#> $f_bi_manual
#>   series_name f_T1 f_T2
#> 1           a  0.8 0.82
#> 2           b  0.5 0.50
#> 3           c 10.0 9.90
#>
#> $bi_q_outliers
#>   series_name year quarter_month intensity
#> 1           a 2020             2        10
#> 2           b 2020             2        10
#> 3           c 2016             4        10
#>
#> $bi_q_manual
#>   series_name year q1_m1 q2_m2 q3_m3 q4_m4 m5 m6 m7 m8 m9 m10 m11 m12
#> 1           a 2020   0.8   0.7  0.84  0.85 NA NA NA NA NA  NA  NA  NA
#> 2           b 2020   0.5   0.4  0.55  0.54 NA NA NA NA NA  NA  NA  NA
#> 3           c 2016  10.0   8.0 11.00 11.00 NA NA NA NA NA  NA  NA  NA
#>
#> attr(,"class")
#> [1] "nbb.dsc.td.multiproc.config"
```

## Main functions

### Description

The function **multiTD** performs multiprocessing (or single processing if there is only one series) temporal disaggregation of time series of an annual to quarterly or monthly set of time series. When only one type of methods must be applied, the user also has the possibility to call the functions **multiTDDenton** and **multiTDCLvar** (CLvar as Chow-Lin and variants). There is little differences between the general and the specific functions which are explained in the next two sections. For the three functions, either the sum or the average consistency between the annual benchmarks and the quarterly or monthly indicators are handled.

During the execution of the code, a (non-exhaustive) number of tests are performed and warnings are returned to assist the user in the analysis of the results and for the specification of the models. It is strongly recommended to read carefully each warning message.

### Call

### multiTD

```
multiTD(
  benchmarks,
  indicators,
  config,
  conversion = c("Sum", "Average"),
  f.score.quantile = c("q.95", "q.90", "q.80", "q.99", "q.999"),
  simpl.e.T1 = FALSE,
  path.xlsx = ""
)
```

The datasets containing the benchmarks and the indicators series must respect some rules described in the documentation of the function (see ?multiTD). In particular, only one indicator maximum can be defined by series (use NA's or any constant when there is no indicator) and the columns of the two datasets must have the same name (the aggregate's names) and be in the same order. This is for the sake of security to avoid associating the wrong indicator to a benchmark series. Benchmark and indicator series can be imported into

RStudio from anywhere. If the data are located in an Excel file, they could be imported for example using the function *read_excel* from the package 'readxl'.

Currently, the **multiTD** function allows for five TD models from two types of methods to define for each series of the dataset: 'edenton' (enhanced Denton), 'denton', 'chow-lin', 'fernandez' and 'litterman'. The selection of the model for each series must be specified in the config object (see section 'Configuration'). Note that 'edenton' is the method by default.

For series where 'edenton' model is choosen, the forecast of the annual BI ratio can be changed manually for extrapolation when the user wants to deviate from the automatic forecast. In the output, an alternative forecasts, which come from the best (in terms of cross validation errors) alternative models automatically tested, is returned for comparison. The user also has the possibility to go further and compare the forecasts coming from more forecasting models by using the functions **ForecastSeries**, **CalcCVRMSEOfSelectedModels** and **CalcCVRMSE** (see section 'Other functions' for more info).

For series where 'denton' or 'edenton' model is defined, it is also possible to add outliers in the infra-annual BI ratio and/or change it manually. Those must be specified in the config object. It is advised to use those possibilities with parsimony only when there are strong arguments for doing so. Currently, outliers are not handled with Chow-Lin model and variants. When a specification cannot be handled by the model (e.g. if we define an outlier for a series treated with Chow-Lin), it is ignored and a warning is returned.

The parameter 'conversion' is the type of consistency to consider between the annual benchmarks and the infra-annual indicators. 'Sum'/'Average' means that the annual sum/average of the infra-annual disaggregated series must match the benchmark.

The parameter 'f.score.quantile' is only relevant when 'edenton' model is used. It tells how much evidence is required to switch from a random walk process to the best of TRAMO / mean of the last 5 years / geometric mean of growth rates of the last 5 years in the automatic selection of the forecasting model for the annual benchmark. For instance, when the quantile 'q.80' is specified instead of the default 'q.95', less evidence is required to switch to the alternative model.

The parameter 'simpl.e.T1' is very specific. It is only relevant when 'edenton' model is used and when the extrapolation period matches one year exactly. In this case, when simpl.e.T1=TRUE, the approximation from IMF continues to be applied, meaning no revision in the past of the series. When simpl.e.T1=FALSE (or anyway when the period of extrapolation exceeds one year), an extra year is implicitly added to the benchmark (by using the forecast of the annual BI ratio) before running the function, implying revision of the past of the series.

Finally, 'path.xlsx' allows the user to export the main results directly into an Excel file.

### multiTDDenton

```
multiTDDenton(
  benchmarks_ts,
  indicators_ts,
  enhanced = FALSE,
  f_bi_ts = NULL,
  bi_q_outliers_form = NULL,
  bi_q_manual_form = NULL,
  conversion = c("Sum", "Average")
)
```

In addition to the rules mentioned in the previous section, the datasets containing the benchmarks and the indicators must be defined as a ts object. The conversion of a data.frame to a ts object is simple using the function *ts* (see example in ?multiTDDenton).

When enhanced=TRUE, 'edenton' model is used, otherwise 'denton' model is used. There is an important difference between **multiTDDenton** and **multiTD** function when the extrapolation period equals or exceeds one year. In this case, **multiTD** automatically extent the benchmark series implicitly (when it is equal to one year, this depends on the value of the parameter *simpl.e.T1*) before running the model. **This is not the case with multiTDDenton and the user must previously call the function ExtendBenchmarksSeries if he wants to extend the benchmark series.**

The parameter 'f_bi_ts' is only relevant when enhanced=TRUE. The forecast of the annual BI ratio can be generated using the function **ForecastAnnualBIRatios** (see section 'Other functions').

The parameter 'bi_q_outliers_form' and 'bi_q_manual_form' allows the user to add outliers and define the infra-annual BI ratio manually for some year(s). Those can be added directly in R or imported from Excel (or other providers) by using the input structure defined in the configuration section. Then, the function **ValidateAndFormatBIqOutliers** and **ValidateFormatBIqManual** can be used respectively to get the appropriate format to use in **multiTDDenton**.

The parameter 'conversion' is the type of consistency to consider between the annual benchmarks and the infra-annual indicators. 'Sum'/'Average' means that the annual sum/average of the infra-annual disaggregated series must match the benchmark.

### multiTDCLvar

```
multiTDCLvar(
  benchmarks_ts,
  indicators_ts,
  models,
  conversion = c("Sum", "Average")
)
```

In addition to the respect of the same rules as in **multiTD**, the datasets containing the benchmarks and the indicators must be defined as a ts object. The conversion of a data.frame to a ts object is simple using the *ts* function (see example in ?multiTDCLvar).

The parameter 'models' allows the user to choose between 'chow-lin', 'fernandez' and 'litterman' for each series. A character vector with the name of the model to use for each series in the same order as the columns in 'benchmarks_ts' must be submitted.

The parameter 'conversion' is the type of consistency to consider between the annual benchmarks and the infra-annual indicators. 'Sum'/'Average' means that the annual sum/average of the infra-annual disaggregated series must match the benchmark.

### Output

```
# Example of output

## Step 1
suppressMessages(suppressWarnings(library(nbbTD)))

## Step 2
data("nbb_data")
benchmarks <- nbb_data$B1G_Y_data
indicators <- nbb_data$TURN_Q_data
colnames(benchmarks) <- colnames(indicators) <- c("DATE", "CE", "FF", "HH")

## Step 3
mymodel <- data.frame (series_name  = c("CE","FF","HH"), model = c("edenton","chow-lin","edenton"))
myconfig <- setConfig_default(model = mymodel) # use ?setConfig_default for more information

## Step 4
```

```
res <- multiTD(benchmarks, indicators, config = myconfig) # use ?multiTD for more information over the
#> Warning: Significant relationship was found between the bi ratio and the indicator in series HH
#> This could indicate the presence of an outlier or the necessity to use a model including a constant
#> Further analysis should be perform to find the reason and adapt the model if necessary.

## Step 5
print(res) # overview of all the results
#> Call:
#> multiTD(benchmarks = benchmarks, indicators = indicators, config = myconfig)
#>
#> Models:
#> chow-lin  edenton
#>        1        2
#>
#> Composition of the first-level output:
#>  [1] "call"           "model"          "Denton"         "eDenton"        "CLvar"          "benchmark
#> [10] "bi.annual"      "bi.annual.f"    "bi.annual.falt" "overall.fit"    "warnings"
#>
#> Use summary() for more details.

# summaryt(res) # overview of all the results
# plot(res, series_name = "FF") # plot td series and bi ratio
# print(res$eDenton) # specific print function for series treated with 'edenton'
# summary(res$Denton) # specific summary function for series treated with 'denton'
# plot(res$CLvar) # specific plot function for series treated with 'chow-lin', 'fernandez' or 'litterma

# res$eDenton$bi.infra # use '$' sign to navigate through the output
# res$CLvar$details$FF

# runShiny(res) # Visualization of the output (see section 'Shiny App')
```

The output of the function **multiTD** is an object of class ‘nbb.dsc.td.multiproc.output’ which contain many information intended to help the user evaluate the quality of the results and take informed decision concerning the specifications of the models (see also section ‘*nbbTD* in production’) for each series. Numeric results will be discussed in this section. Graphical analysis is also of particular interest. In addition to the specific **plot** functions, the function **runShiny** displays informative charts in a convenient manner and we therefore recommend the user to use it. See section ‘Shiny App’ for more details. Note that the function **runShiny** requires an object of class ‘nbb.dsc.td.multiproc.output’ as parameter and therefore only works with output from **multiTD**. The output from **multiTDDenton** and **multiTDCLvar**, which are object of class ‘nbb.dsc.td.multiproc.edenton.output’ and ‘nbb.dsc.td.multiproc.clvar.output’ respectively, are not handled by **runShiny**.

*First-level output* Those are output that are common to all models. First-level means that they are accessible using only one ‘$’ sign after the variable name containing the results (see example). They are listed in the **print** or **summary** function.

- *call*: call of the function
- *model*: the model used for each series
- *Denton*: path to the second-level output specific to the series where ‘denton’ model is used. Empty if ‘denton’ is not used for any series.
- *eDenton*: path to the second-level output specific to the series where ‘edenton’ model is used. Empty if ‘edenton’ is not used for any series. The disposition of the second-level output is similar to this of ‘denton’. This second-level output is the same as the first-level output of the function **multiTDDenton**.
- *CLvar*: path to the second-level output specific to the series where ‘chow-lin’, ‘fernandez’ or ‘litterman’ is used. Empty if ‘chow-lin’, ‘fernandez’ or ‘litterman’ is not used for any series. This second-level

output is the same as the first-level output of the function **multiTDCLvar**.

- *benchmarks*: the benchmarks series
- *indicators*: the indicators series
- *td.series*: the disaggregated series
- *bi.infra*: the resulting infra-annual BI ratio ($bi.infra = \frac{ts.series}{indicator}$)
- *bi.annual*: the annual BI ratio
- *bi.annual.f*: forecasts of the annual BI ratio for years T+1 and T+2. While this is defined explicitly when 'edenton' model is used, this is not the case with the other models. In those cases, the function **CalcImplicitAnnualBIForecast** is used to calculate the implicit forecast of the annual BI ratio's for year T+1 (only possible when the extrapolation period is greater or equal to one year).
- *bi.annual.falt*: alternative forecast of the annual BI ratio for years T+1 and T+2. When the selected forecasting model is random walk, the alternative forecast comes from the best alternative model tested (pm: mean of last 5 years, geometric mean of growth rates of last 5 years or TRAMO). When the selected forecasting model is not random walk, the alternative forecasts come from random walk. *bi.annual.falt* is define as 'NA' when another model than 'edenton' is used.
- *overall.fit*: it is not recommended to use an infra-annual indicator which fits the benchmark poorly. To measure the fit, a linear regression between the growth rates of the annual benchmark and the growth rates of the annualized indicator is performed. The t-stat and the p-value of the parameter associated with the indicator are returned here. The higher(lower) the t-stat(p-value), the higher the fit between the benchmark and the indicator. Note that if the |t-stat| < 2, there is no statistical evidence of relationship between the benchmark and the indicator. A negative t-stat means that there is an inverse relationship between the benchmark and the indicator. In this case, it should be verified that this makes sense!
- *warnings*: collect all the warnings automatically generated from a (non-exhaustive) number of tests performed during the execution of the code. Those should be read carefully as they intend to help the user in the analysis of the results and for the specification of the models.

*Second-level output* The second-level output of the function **multiTD** has the same structure for 'denton' and 'edenton' and corresponds to the first-level output of the function **multiTDDenton**. It focuses on the BI ratio and concerns only the series where the corresponding model is used.

- *td.series*: the disaggregated series
- *bi.infra*: the resulting infra-annual BI ratio ($bi.infra = \frac{ts.series}{indicator}$)
- *bi.infra.f*: the forecast of the next few infra-annual BI ratio. Could be used directly instead of re-running the entire process when there is no revision of neither the benchmark (also implicitly, rerun the process when an entire year of extrapolation is covered) or the indicator in the past as well as no extra input (such as outlier).

- *bi.infra.SD*: standard error of the BI ratio (i.e. the coefficient $\beta_t$). Can be used to draw a confidence interval around the BI ratio and therefore also around the disaggregated series.

- *bi.infra.NO*: when outlier(s) are specified, return the results without outlier for comparison. Otherwise, return the same results are *bi.infra.* Useful to compare the disaggregated series with and without outlier. (see Shiny App)
- *bi.infra.NM*: when some infra-annual BI ratio are changed manually, return the orginal results without those manual adaptation for comparison. Otherwise, return the same results are *bi.infra.* Useful to compare the disaggregated series with and without manual infra-annual BI ratio. (see Shiny App).
- *bi.infra.NONM*: when outlier(s) are specified and/or some infra-annual BI ratio are changed manually, return the orginal results without those adaptations. Otherwise, return the same results are *bi.infra.* Useful to compare the disaggregated series with and without outlier and manual infra-annual BI ratio. (see Shiny App).
- *bi.steadiness*: measure of the steadiness of the BI ratio. For each series, it returns the percentage of times that the mean of the BI ratio fits inside the confidence interval (CI). When the percentage is high enough (say > 0.95), the assumption of a steady BI ratio is plausible and a model like Fernandez where the BI ratio is expected to remain constant over time might be appropriate to use.

The second-level output ‘CLvar’ of the function **multiTD** includes the results of the series where one of the Chow-Lin variants is used. It also corresponds to the first-level output of the function **multiTDCLvar**.

- *td.series*: the disaggregated series
- *bi.infra*: the resulting infra-annual BI ratio ($bi.infra = \frac{ts.series}{indicator}$)
- *details*: path to a third-level output where the numeric output of the model is available by series. Among them, we retrieve the value of ‘rho’ which should always be checked for Chow-Lin and Litterman. A value of rho<0 leads to spurious results and should be avoided. The closer the value is from 1, the more ‘memory’ and the slower it converges to the long-term mean in extrapolation. On the other hand, a value of rho close to 0 means that we are close to a simple OLS model where the errors are simply divided by 4 (quarterly indicator) with a potential step effect resulting from this. The other output are the estimates of the coefficients of the model (the value of the t-stat is of particular interest to evaluate the fit between the benchmark and the indicator), the variance-covariance matrix, the part of the results that comes from the regression part of the equation, the smoothing percentage and the log-likelihood.

## Other functions

Most of the other functions are made available to format or improve the input of the main functions.

- **ValidateAndFormatBIqOutliers**: validate and format initial data set including the infra-annual BI outliers (or BI shifts) in a way that can be handled by the function **multiTDDenton**. See ?ValidateAndFormatBIqOutliers for more info.

- **ValidateFormatBIqManual**: validate and format the initial data set including the manual infra-annual BI ratios in a way that can be handled by the function **multiTDDenton**. See ?ValidateAndFormatBIqManual for more info.

- **ForecastAnnualBIRatios**: automatically forecast annual BI ratio’s. The function generates a list of two elements containing the primary and alternative forecasts. The elements are formatted to be compatible with the function **multiTDDenton** for which one of the two elements of the list can be used as input. See ?ForecastAnnualBIRatios for more info.

- **ForecastSeries**: forecast annual time series according to a selection of models. The choice of models is even larger than what is automatically tested. Thus, the function allows the user to refine the forecast of the annual BI ratio if necessary. See ?ForecastSeries for more info. See also ?CalcCVRMSEOfSelectedModels and ?CalcCVRMSE.

- **CalcCVRMSEOfSelectedModels**: calculate RMSE of cross validation errors for a selection of forecasting models (those used in the automatic procedure). By showing the magnitude of the differences between the models, It can be used as additional information to refine the forecast of the annual BI ratio if necessary. See ?CalcCVRMSEOfSelectedModels for more info.

- **CalcCVRMSE**: calculate RMSE of cross-validation errors for a given model. The choice of models is large and allows the user to situate the RMSE of cross-validation errors for a model which is not part of the automatic selection procedure. Can be used as complement of **ForecastSeries** to potentially decide to change the automatic forecast of the annual BI ratio (adapt it manually in the config in this case). See ?CalcCVRMSE for more info.

- **CalculBIratio**: calculate annual BI ratio which is the ratio between the annual benchmark and the annualized indicator. Extrapolation period is ignored. See ?CalculBIratio for more info.

- **ExtendBenchmarksSeries**: extend benchmarks series of one year based on the forecast of the annual BI ratio and the indicator if the extrapolation period is greater or equal to one year (only relevant with ‘edenton’). It is done automatically in **multiTD** when the extrapolation period exceeds one year or when the extrapolation period is equal to one year and the parameter simpl.e.T1=FALSE. This isn’t done automatically in **multiTDDenton** and the function **ExtendBenchmarksSeries** should be preliminary run on the benchmark series if relevant. See ?ExtendBenchmarksSeries for more info.

- **CalcImplicitAnnualBIForecast**: when 'edenton' is used, the forecast of the annual BI ratio is explicit. This is not the case with the other models. When another model is used, the function **CalcImplicitAnnualBIForecast** can be used to calculate the implicit forecast of the annual BI ratio (only possible when the extrapolation period is greater or equal to one year). The implicit forecast is also returned automatically in the output of the **multiTD** function in *bi.annual.f* as well as graphically in the corresponding tab of the Shiny app. See ?CalcImplicitAnnualBIForecast for more info.
- **PlotAnnualBIratio**: plot annual BI ratio and its forecast in a formatted chart. See ?PlotAnnualBIratio for more info.

## Shiny App

In addition to numeric output, the package includes a Shiny App to help visualize the results. The UI is accessible directly by calling the function **runShiny**. This function requires a single argument of class 'nbb.dsc.td.multiproc.output' which is the results of the function **multiTD**. The output from **multiTDDenton** and **multiTDCLvar** are not handled by **runShiny**.

```
runShiny(res)
```

The app is divided in three parts.

The first part on the top left side of the window is composed of a single drop down list where the user select the series to analyse.

The second part on the top right side of the window includes informative charts **on an annual basis**. There are four clickable tabs:

- *Growth chart*: by comparing the annual development of the benchmark and the indicator, it gives an insight of the global fit but also of local fit on the last few years for instance. It also helps to identify potential problematic years displaying a large mismatch.
- *Scaled annual BI ratio*: it shows the development of the scaled annual BI ratio. The forecasts are also displayed for the years T+1 and T+2 when 'edenton' model is used. When another model is used, the implicit forecast of the year T+1 is displayed as long as the extrapolation period is long enough. Graphical analysis of the annual BI ratio is a good way to evaluate the fit between the benchmark and the indicator through time. A stable BI ratio between two consecutive periods means that the benchmark and the indicator evolve in parallel. Conversely, an upward/downward shift in the BI ratio translates a positive/negative difference in the development of the annual benchmark and the annualized indicator. Scaling the BI ratio around an average value of 100 allows the user to interpret (approximately) the difference in terms of percentage.
- *Annual BI ratio*: same as *Scaled annual BI ratio* but the BI ratio is not scaled. Compared to the previous tab, the level of the BI ratio can be seen. This can be useful for instance if the user wants to adapt the forecast manually.
- *Annual BI ratio AF*: "AF" means "Alternative Forecast. The difference with the previous tab is the value of the forecasts when the model 'edenton' is used. When the primary forecasts come from a random walk process, the AF shows the forecasts from the best alternative model (from those selected in the automatic procedure). No forecast are displayed here when another model than 'edenton' is used.

The third part on the bottom right of the window includes informative charts **on an quarterly basis**. There are six clickable tabs:

- *TD series*: the disaggregated series
- *Infra BI ratio*: the resulting infra-annual BI ratio ($bi.infra = \frac{ts.series}{indicator}$) toegether with the forecasted values. The latter is only available with 'edenton' and 'denton' models and could be used directly in production instead of re-running the entire process when there is no revision of neither the benchmark (also implicitly, rerun the process when an entire year of extrapolation is covered) or the indicator in the past as well as no extra input (such as outlier).

- *(e)Denton CI*: it shows the disaggregated series together with the confidence interval. Only measurable with 'edenton' and 'denton'. Nothing is returned with Chow-Lin and its variants. With 'edenton', as the extrapolation of an incomplete year is done using an adhoc procedure, no confidence interval cannot be computed for this part of the series.
- *(e)Denton BI analysis*: another informative chart only relevant with 'edenton' and 'denton' models (no chart displayed with another model). It compares the infra annual BI ratio of the series with/without outlier(s) in the infra-annual BI ratio and with/without infra-annual BI ratio imported manually. Of course, when no outlier or manual BI ratio are specified, only one series, the infra-annual BI ratio, is displayed.
- *(e)Denton TD analysis*: same idea as the previous tab but here, the disaggregated series is compared instead of the BI ratio. It shows the impact of the outlier(s) or manual infra-annual BI ratio on the results.
- *Chow-Lin analysis*: only relevant with Chow-Lin method and variants. Nothing is returned with (e)Denton models. It decomposes the disaggregated series between the regression part and the smoothing part. It gives an insight of the part of the disaggregated series coming from the indicator and the part being smoothed.

## ***nbbTD*** in production

The processing time of **multiTD** is limited from a few seconds to a few minutes depending on the number of treated series.

When the time comes to review the models (or some elements of it), the suggestion is to proceed by trial and error by running the function as many time as necessary. For example, whether to include an outlier or not and set its intensity is something to test. The warnings are also here to help even though they are not exhaustive and both the numeric and graphical output should always be thoroughly analyzed.

A typical step-by-step approach could be:

1. Run **multiTD** a first time using default, previous or any specifications you think the best for each series
2. Check the annual fit between the benchmark and the indicator numerically and graphically. Is the indicator relevant? Are there years where the BI ratio changes suddenly? Why? Although there can be economical or structural reasons for that, inspection of the BI ratio can also be a good way to rapidly detect errors in either the indicator or the benchmark. If the reason is economical or structural and you want to use (e)Denton model, should the change(s) be smoothed (potentially causing wave effect) or is it preferable to manually add a shift in the BI ratio by defining outlier/manual change (or both)? When eDenton model is used, is the automatic forecast of the annual BI ratio good enough for extrapolation? You can adapt it manually if you have other information or (with caution) after checking the alternative forecasts or even forecasts from other models using the other functions available in the package.

3. Read the warning messages and check the infra-annual results (both numeric and graphic). Is there irrelevant input? Is there evidence to adapt the model for some series? While (e)Denton PFD method preserves the quarter-on-quarter movement of the indicator as much as possible, Chow-Lin method and its variants tend to smooth the discrepancies between the benchmark and the indicator. Both types of method rely on a number of assumptions which should be scrutinized. Chow-Lin method and its variants assumes homoscedasticity but also strict exogeneity which can be a poor assumption in the context of National Accounts for example where many changes in methodology, sources, population and even relationship between benchmark and indicator occur through time. With Denton, the relationship between benchmark and indicator is not fixed but is assumed to evolve smoothly over time. Large discrepancies can significantly affect Denton results locally inducing potential wave effects. This makes

Denton PFD method even more sensitive to outliers (locally) than Chow-Lin and variants where outliers rather have a global impact on the coefficients. Finally, it should be noted that with Denton, the impact of revision in the latest benchmark/indicator values of the series decrease quickly as we go backwards in time. This is not the case with Chow-Lin and its variants.

4. Change input if necessary and re-run the function **multiTD**
5. Repeat steps 2 and 3 until the results for all series are satisfying

## Further readings

- *IMF (2017) 'Quarterly National Accounts Manual', 2017 edition (p.86-126)*
- *Eurostat (2018) 'ESS guidelines on temporal disaggregation, benchmarking and reconciliation', 2018 edition.*
- *Chamberlin G. (2010) 'Temporal disaggregation', Office for National Statistics, Economic & Labour Market Review, November.*
- *Chow, G., & Lin, A.L. (1971) 'Best linear unbiased distribution and extrapolation of economic time series by related series', Review of Economic and Statistics, 53, 372-375.*
- *Quilis E. (2018) 'Temporal disaggregation of economic time series: The view from the trenches', Wiley, June.*