A Beginner’s Benchmarking Demo Script

```
# Load G-Series in R (package gstest)
library(gstest)

# Set the working directory (for the PDF graph files)
iniwd <- getwd()
setwd("C:/Temp")




#### Example 1 (single quarterly series) ####
# Simple case with a single quarterly series to benchmark to annual values

# Quarterly indicator series
my_ts_qtr <- ts(c(1.9, 2.4, 3.1, 2.2, 2.0, 2.6, 3.4, 2.4, 2.3),
                start = c(2015, 1),
                frequency = 4)
my_series1 <- ts_to_tsDF(my_ts_qtr)

# Annual benchmarks for quarterly data
my_ts_ann <- ts(c(10.3, 10.2),
                start = 2015,
                frequency = 1)
my_benchmarks1 <- ts_to_bmkDF(my_ts_ann, ind_frequency = 4)

# Benchmarking using...
#   - recommended `rho` value for quarterly series (`rho = 0.729`)
#   - proportional model (`lambda = 1`)
#   - bias-corrected indicator series with the estimated bias (`biasOption = 3`)
out_bench1 <- benchmarking(my_series1,
                           my_benchmarks1,
                           rho = 0.729,
                           lambda = 1,
                           biasOption = 3)

# Outputs
View(out_bench1$series)
View(out_bench1$benchmarks)
View(out_bench1$graphTable)

# Graphics (default set of plots)
plot_graphTable(out_bench1$graphTable,
                "Ex1_graphs.pdf")
# With the G. R. Table
plot_graphTable(out_bench1$graphTable,
                "Ex1_graphs_with_GRTable.pdf",
                GR_table_flag = TRUE)


# Compare with package tempdisagg that allows Denton benchmarking
# (modified version by Cholette)

#install.packages("tempdisagg")
library(tempdisagg)
#install.packages("dplyr")
library(dplyr)


# Proportional Denton
mult_denton <- benchmarking(my_series1,
                            my_benchmarks1,
                            rho = 1,
                            lambda = 1)

td_mult_denton <- td(formula = my_ts_ann ~ 0 + my_ts_qtr,
                     method = "denton-cholette",
                     criterion = "proportional")

compare_mult <- ts.union(gstest = tsDF_to_ts(mult_denton$series, frequency = 4),
                         tempdisagg = predict(td_mult_denton),
                         dframe = TRUE) %>%
  cbind(., diff = .[, 1] - .[, 2])

# Same results
View(compare_mult)


# Additive Denton
add_denton <- benchmarking(my_series1,
                           my_benchmarks1,
                           rho = 1,
                           lambda = 0)

td_add_denton <- td(formula = my_ts_ann ~ 0 + my_ts_qtr,
                    method = "denton-cholette",
                    criterion = "additive")

compare_add <- ts.union(gstest = tsDF_to_ts(add_denton$series, frequency = 4),
                        tempdisagg = predict(td_add_denton),
                        dframe = TRUE) %>%
  cbind(., diff = .[, 1] - .[, 2])

# Same results
View(compare_add)




#### Example 2 (multiple quarterly series) ####
```

```r
# Two quarterly series to benchmark to annual values,
# with BY-groups and user-defined alterability coefficients

# Sales data (same sales for groups A and B; only alter coefs for van sales differ)
qtr_sales <- ts(matrix(c(# Car sales
  1851, 2436, 3115, 2205, 1987, 2635, 3435, 2361, 2183, 2822,
  3664, 2550, 2342, 3001, 3779, 2538, 2363, 3090, 3807, 2631,
  2601, 3063, 3961, 2774, 2476, 3083, 3864, 2773, 2489, 3082,
  # Van sales
  1900, 2200, 3000, 2000, 1900, 2500, 3800, 2500, 2100, 3100,
  3650, 2950, 3300, 4000, 3290, 2600, 2010, 3600, 3500, 2100,
  2050, 3500, 4290, 2800, 2770, 3080, 3100, 2800, 3100, 2860),
  ncol = 2),
  start = c(2011, 1),
  frequency = 4,
  names = c("car_sales", "van_sales"))

class(qtr_sales)

ann_sales <- ts(matrix(c(# Car sales
  10324, 10200, 10582, 11097, 11582, 11092,
  # Van sales
  12000, 10400, 11550, 11400, 14500, 16000),
  ncol = 2),
  start = 2011,
  frequency = 1,
  names = c("car_sales", "van_sales"))

# Quarterly indicator series (with default alter coefs for now)
my_series2 <- rbind(cbind(data.frame(group = rep("A", nrow(qtr_sales)),
                                     alt_van = rep(1, nrow(qtr_sales))),
                          ts_to_tsDF(qtr_sales)),
                    cbind(data.frame(group = rep("B", nrow(qtr_sales)),
                                     alt_van = rep(1, nrow(qtr_sales))),
                          ts_to_tsDF(qtr_sales)))

# Set binding van sales (alter coef = 0) for 2012 Q1 and Q2 in group A (rows 5 and 6)
my_series2$alt_van[c(5, 6)] <- 0

# Annual benchmarks for quarterly data (without alter coefs)
my_benchmarks2 <- rbind(cbind(data.frame(group = rep("A", nrow(ann_sales))),
                              ts_to_bmkDF(ann_sales, ind_frequency = 4)),
                        cbind(data.frame(group = rep("B", nrow(ann_sales))),
                              ts_to_bmkDF(ann_sales, ind_frequency = 4)))

# Benchmarking using...
#   - recommended RHO value for quarterly series (rho = 0.729)
#   - proportional model (lambda = 1)
#   - without bias correction (biasOption = 1 and bias not specified)
out_bench2 <- benchmarking(my_series2,
                           my_benchmarks2,
                           rho = 0.729,
                           lambda = 1,
                           biasOption = 1,
                           var = c("car_sales", "van_sales / alt_van"),
                           with = c("car_sales", "van_sales"),
                           by = "group")

# Outputs
View(out_bench2$series)
View(out_bench2$benchmarks)
View(out_bench2$graphTable)

# Graphics
plot_graphTable(out_bench2$graphTable,
                "Ex2_graphs.pdf")




#### Example 3 (multiple quarterly series) ####
# Same as example 2, but benchmarking all 4 series as BY-groups
# (4 BY-groups of 1 series instead of 2 BY-groups of 2 series)

my_series3 <- stack_tsDF(ts_to_tsDF(ts.union(A = qtr_sales, B = qtr_sales)))
my_series3$alter <- 1
my_series3$alter[my_series3$series == "A.van_sales" &
                   my_series3$year == 2012 & my_series3$period <= 2] <- 0
my_benchmarks3 <- stack_bmkDF(ts_to_bmkDF(ts.union(A = ann_sales, B = ann_sales),
                                          ind_frequency = 4))
out_bench3 <- benchmarking(my_series3,
                           my_benchmarks3,
                           rho = 0.729,
                           lambda = 1,
                           biasOption = 1,
                           var = "value / alter",
                           with = "value",
                           by = "series")

# Outputs
View(out_bench3$series)
View(out_bench3$benchmarks)
View(out_bench3$graphTable)

# Graphics
plot_graphTable(out_bench3$graphTable,
                "Ex3_graphs.pdf")

# Convert the benchmarked series as a "mts" object
my_out_series3 <- tsDF_to_ts(unstack_tsDF(out_bench3$series), frequency = 4)
class(my_out_series3)
plot(my_out_series3)
```

```
#### Monthly data (Box & Jenkins airline series) ####

data(AirPassengers)
my_AP_ind <- ts_to_tsDF(AirPassengers)

# Create annual benchmarks by changing the level (5 times larger), adding some random
# noise and dropping the last 2 benchmarks
set.seed(as.Date("2003-03-25"))  # for results reproducibility (select any date you want)
#set.seed(NULL)
ann_AP <- round(jitter(aggregate.ts(AirPassengers, nfrequency = 1, FUN = sum) * 5,
                       amount = 2500))
my_AP_bmk <- (ts_to_bmkDF(ann_AP, ind_frequency = 12))[1:10, ]


# With bias correction (estimated bias with `biasOption = 3`)
#   => everything looks good
out_bench_AP <- benchmarking(my_AP_ind,
                             my_AP_bmk,
                             rho = 0.9,
                             lambda = 1,
                             biasOption = 3)
View(out_bench_AP$series)
View(out_bench_AP$benchmarks)
View(out_bench_AP$graphTable)

plot_graphTable(out_bench_AP$graphTable,
                "AP_graphs.pdf")


# Without bias correction (`biasOption = 1`)
#   => issues with the projected adjustments at the end of the series for periods
#      not covered by a benchmark
out_bench_AP_noBias <- benchmarking(my_AP_ind,
                                    my_AP_bmk,
                                    rho = 0.9,
                                    lambda = 1,
                                    biasOption = 1)
View(out_bench_AP_noBias$series)
View(out_bench_AP_noBias$benchmarks)
View(out_bench_AP_noBias$graphTable)

plot_graphTable(out_bench_AP_noBias$graphTable,
                "AP_graphs_noBias.pdf")


# Denton benchmarking (`rho = 1`, bias correction is irrelevant)
#   => last adjustment repeated (forever) at the end of the series
#      (strong assumption, but not necessarily problematic)
out_bench_AP_denton <- benchmarking(my_AP_ind,
                                    my_AP_bmk,
                                    rho = 1,
                                    lambda = 1)
View(out_bench_AP_denton$series)
```

```r
View(out_bench_AP_denton$benchmarks)
View(out_bench_AP_denton$graphTable)

plot_graphTable(out_bench_AP_denton$graphTable,
                "AP_graphs_Denton.pdf")


# Denton benchmarking approximation (`rho = 0.999`, `biasOption = 3`)
#   => regression benchmarking model approximation of the "pure" Denton method
#   => last adjustment repeated at the end of the series (mild convergence to the bias
#      compared to the "pure" Denton method)
out_bench_AP_dentonApprox <- benchmarking(my_AP_ind,
                                          my_AP_bmk,
                                          rho = 0.999,
                                          lambda = 1,
                                          biasOption = 3)
View(out_bench_AP_dentonApprox$series)
View(out_bench_AP_dentonApprox$benchmarks)
View(out_bench_AP_dentonApprox$graphTable)

plot_graphTable(out_bench_AP_dentonApprox$graphTable,
                "AP_graphs_DentonApprox.pdf")


# Pro-rating (`lambda = 0.5` and `rho = 0`)
#   => no movement preservation: all adjustments are lumped into January every year
#      (with a disastrous impact on some of the initial December to January movements)
#   => immediate convergence to the bias (estimated with `biasOption = 3`) for the
#      projected adjustments at the end of the series
out_bench_AP_proRate <- benchmarking(my_AP_ind,
                                     my_AP_bmk,
                                     rho = 0,
                                     lambda = 0.5,
                                     biasOption = 3)
View(out_bench_AP_proRate$series)
View(out_bench_AP_proRate$benchmarks)
View(out_bench_AP_proRate$graphTable)

plot_graphTable(out_bench_AP_proRate$graphTable,
                "AP_graphs_proRating.pdf")




#### End of year stocks ("kinks" in the adjustments with `benchmarking()`) ####

# Quarterly indicator stock series (same pattern repeated every year)
my_stock_ind <- ts_to_tsDF(ts(rep(c(85, 95, 125, 95), 7),
                              start = c(2013, 1),
                              frequency = 4))

# Annual benchmarks (end-of-year stocks)
my_stock_bmk <- ts_to_bmkDF(ts(c(135, 125, 155, 145, 165),
                               start = 2013,
                               frequency = 1),
                            discrete_flag = TRUE,
                            alignment = "e",
                            ind_frequency = 4)

# With `benchmarking()` ("Proc Benchmarking" approach)
out_stock_PB <- benchmarking(my_stock_ind,
                             my_stock_bmk,
                             rho = 0.729,
                             lambda = 1,
                             biasOption = 3)

# With `stock_benchmarking()` ("Stock Benchmarking" approach)
out_stock_SB <- stock_benchmarking(my_stock_ind,
                                   my_stock_bmk,
                                   rho = 0.729,
                                   lambda = 1,
                                   biasOption = 3)

# Benchmarking adjustments of both approaches
plot_benchAdj(PB_graphTable = out_stock_PB$graphTable,
              SB_graphTable = out_stock_SB$graphTable)

# Have you noticed how smoother the `stock_benchmarking()` adjustments are compared
# to the `benchmarking()` ones?

# The gain in the quality of the resulting benchmarked stocks might not necessarily
# be obvious in this example.
plot(out_stock_SB$graphTable$t, out_stock_SB$graphTable$benchmarked,
     type = "b", col = "red", xlab = "t", ylab = "Benchmarked Stock")
lines(out_stock_PB$graphTable$t, out_stock_PB$graphTable$benchmarked,
      type = "b", col = "blue")
legend(x = "topleft", bty = "n", inset = 0.05, lty = 1, pch = 1,
       col = c("red", "blue"), legend = c("out_stock_SB", "out_stock_PB"))
title("Benchmarked Stock")

# PDF graphics
plot_graphTable(out_stock_PB$graphTable, "Stock_graphs_PB.pdf")
plot_graphTable(out_stock_SB$graphTable, "Stock_graphs_SB.pdf")

# What about cases where a flat indicator is used, which may happen in practice
# in absence of a good indicator of the quarterly (sub-annual) movement?
my_flat_ind <- my_stock_ind
my_flat_ind$value <- 1
out_stock_PB2 <- benchmarking(my_flat_ind,
                              my_stock_bmk,
                              rho = 0.729,
                              lambda = 1,
                              biasOption = 3)
out_stock_SB2 <- stock_benchmarking(my_flat_ind,
                                    my_stock_bmk,
                                    rho = 0.729,
                                    lambda = 1,
                                    biasOption = 3)

plot(out_stock_SB2$graphTable$t, out_stock_SB2$graphTable$benchmarked,
     type = "b", col = "red", xlab = "t", ylab = "Benchmarked Stock")
lines(out_stock_PB2$graphTable$t, out_stock_PB2$graphTable$benchmarked,
      type = "b", col = "blue")
legend(x = "bottomright", bty = "n", inset = 0.05, lty = 1, pch = 1,
       col = c("red", "blue"), legend = c("out_stock_SB2", "out_stock_PB2"))
title("Benchmarked Stock - Flat Indicator")

# The awkwardness of the benchmarked stocks produced by `benchmarking()` suddenly
# becomes obvious. That's because the benchmarked series corresponds to the
# benchmarking adjustments when using a flat indicator (e.g., a series of 1's
# with proportional benchmarking):
plot_benchAdj(PB_graphTable = out_stock_PB2$graphTable,
              SB_graphTable = out_stock_SB2$graphTable)

# The shortcomings of the "Proc Benchmarking" approach (function `benchmarking()`)
# with stocks is also quite noticeable in this case when looking at the resulting
# quarterly growth rates, which are conveniently produced by `plot_graphTable()`.
# Pay particular attention to the transition in the growth rates from Q4 to Q1
# every year in the generated PDF graphs.
plot_graphTable(out_stock_PB2$graphTable, "Stock_graphs_PB_flat_indicator.pdf")
plot_graphTable(out_stock_SB2$graphTable, "Stock_graphs_SB_flat_indicator.pdf")


# Reset the working directory to its initial location
setwd(iniwd)
```