TEMPORAL DISAGGREGATION OF TIME SERIES: A MATLAB LIBRARY

Enrique M. Quilis[1] [2]

December 2018

Abstract

This library contains a complete set of MATLAB functions designed to perform temporal disaggregation of time series using a variety of techniques: methods without indicators (Boot-Feibes-Lisman, Stram-Wei, low-pass interpolation); methods with indicators using different approaches: optimization (Denton), static models (Chow-Lin, Fernandez and Litterman), dynamic models (Santos-Cardoso, Proietti), ARIMA models (Guerrero); and multivariate methods with indicators and transversal constraints (Denton, Rossi, Di Fonzo). The library contains also functions for balancing (proportional, RAS bi-proportional and Van der Ploeg) as well as an interface written in Visual Basic that allows its use in a spreadsheet environment. This library is intended for its use in production mode, easing the tasks of regular data compilation and short-term monitoring, and also for its use in research mode, allowing an in-depth exploration of the results and its internal mechanics.

---

[1] I am grateful to Ana Abad, Fátima Cardoso, José Ramón Cancelo, Ángel Cuevas, Tommaso Di Fonzo, Antoni Espasa, Rafael Frutos, Miguel Jerez, François Libeau, Agustín Maravall, Francisco Melis, Leandro Navarro, Julián Pérez, Tommaso Proietti, José Miguel Ramos, Silvia Relloso and Eduardo Salazar for their input at different stages of this project.

[2] E-mail address: equilis@gmail.com

INDEX

1. INTRODUCTION

Temporal disaggregation of economic time series has a predominantly practical orientation, from its beginning as a critical tool for the compilation of official statistics, mainly in the realm of the National Accounts. This orientation has eased its use as a general econometric tool for short-term economic monitoring.

As is the case in other quantitative fields, the availability of appropriate computer software has been instrumental for the spread of temporal disaggregation techniques among practitioners. In this way, this library is intended for its use in production mode, easing the tasks of regular data compilation and short-term monitoring, and also for its use in research mode, allowing an in-depth exploration of the estimation results and its internal mechanics.

The library contains a complete set of MATLAB functions designed to perform temporal disaggregation of economic time series using a variety of techniques: methods without indicators (Boot et al., 1967; Stram and Wei, 1986 and low-pass interpolation inspired by Sims, 1974); methods with indicators using different approaches: quadratic optimization (Denton, 1971), static models (Chow and Lin, 1971; Fernández, 1981 and Litterman, 1983), dynamic models (Santos-Cardoso, 2001 and Proietti, 2006), ARIMA models (Guerrero, 1990); and multivariate methods with indicators and transversal constraints (Denton, 1971; Rossi, 1982 and Di Fonzo, 1990). The library contains also functions for balancing (proportional, RAS bi-proportional and Van der Ploeg, 1982, 1985).

Apart from the specific papers above mentioned, the general theoretical background of the methods can be found in Di Fonzo (1987) and Dagum and Cholette (2006), among others. A comprehensive and updated analysis of temporal disaggregation, benchmarking and balancing can be found in Chen et al. (2018a) and the papers cited therein. Due to its close relationship with the procedures included in this library, we should mention Abad and Quilis (2005), Bisio and Moauro (2018), Chen et al. (2018b), Daalmans (2018), Guerrero and Corona (2018), Quilis (2018) and Temursho (2018).

2. TEMPORAL DISAGGREGATION WITHOUT INDICATORS

When the information set is composed only by the low-frequency benchmark, we have several methods to perform temporal disaggregation:

- Boot-Feibes-Lisman (BFL): `bfl()`, `bfl_v()`
- Stram-Wei (SW): `sw()`
- Low-Pass Interpolation: `low_pass_interpolation()`
- Chow-Lin method without indicator: `uni_chowlin()`

2.1. Boot-Feibes-Lisman (BFL) method

The method proposed by Boot et al. (1967) is a natural starting point due to its well-defined structure and ease of implementation. The next box presents its structure, defining inputs and outputs.

Box 2.1: Boot-Feibes-Lisman (BFL) function

```
function res = bfl(Y,ta,d,sc)
% PURPOSE: Temporal disaggregation using the Boot-Feibes-Lisman method
% ------------------------------------------------------------------------
% SYNTAX: res = bfl(Y,ta,d,sc);
% ------------------------------------------------------------------------
% OUTPUT: res: a structure
%         res.meth   = 'Boot-Feibes-Lisman'
%         res.N      = Number of low frequency data
%         res.ta     = Type of disaggregation
%         res.d      = Degree of differencing
%         res.sc     = Frequency conversion
%         res.y      = High frequency estimate
%         res.et     = Elapsed time
% ------------------------------------------------------------------------
% INPUT: Y: Nx1 ---> vector of low frequency data
%        ta: type of disaggregation
%            ta=1 ---> sum (flow)
%            ta=2 ---> average (index)
%            ta=3 ---> last element (stock) ---> interpolation
%            ta=4 ---> first element (stock) ---> interpolation
%        d: objective function to be minimized: volatility of ...
%            d=0 ---> levels
%            d=1 ---> first differences
%            d=2 ---> second differences
%        sc: number of high frequency data points for each low frequency data point
%            Some examples:
%            sc= 4 ---> annual to quarterly
%            sc=12 ---> annual to monthly
%            sc= 3 ---> quarterly to monthly
% ------------------------------------------------------------------------
% LIBRARY: sw
% ------------------------------------------------------------------------
% SEE ALSO: sw, tduni_print, tduni_plot
% ------------------------------------------------------------------------
% REFERENCE: Boot, J.C.G., Feibes, W. and Lisman, J.H.C. (1967)
% "Further methods of derivation of quarterly figures from annual data",
% Applied Statistics, vol. 16, n. 1, p. 65-75.
```

The next box presents a script for running `bfl()` and getting its output (text file and graphics). The procedure can be applied to a vector time series using `bfl_v()` if the input parameters (ta, d, sc) are the same for all the series.

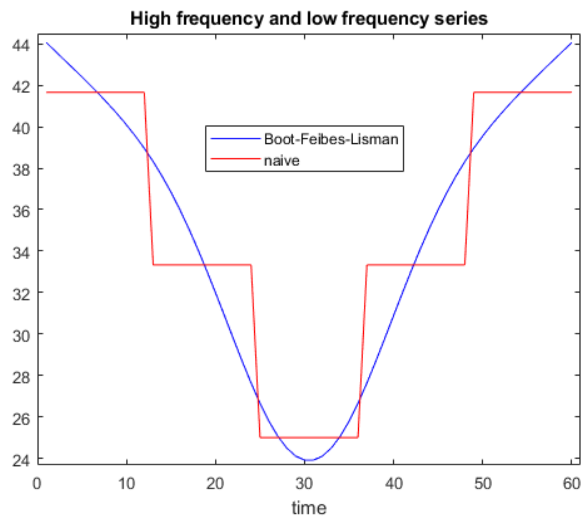
Box 2.2: Boot-Feibes-Lisman (BFL) script

```
% PURPOSE: demo of bfl()
%          Temporal disaggregation using the Boot-Feibes-Lisman method
%---------------------------------------------------
% USAGE: bfl_d
%---------------------------------------------------
close all; clear all; clc;
% Low-frequency data: Denton's benchmark Y
load denton;
% ---------------------------------------------
% Inputs
% Type of aggregation
ta = 1;
% Minimizing the volatility of d-differenced series
d = 2;
% Frequency conversion
sc = 12;
% ---------------------------------------------
% Calling the function: output is loaded in a structure called res
res = bfl(Y,ta,d,sc);
% ---------------------------------------------
% Outputs
% Printed output
file_out = 'bfl.out';
tdprint(res,file_out);
edit bfl.out;
% Graphics
tdplot(res);
```

The output of the `bfl()` procedure can be seen in the next box.

Box 2.3: Boot-Feibes-Lisman (BFL) output

```
****************************************************
TEMPORAL DISAGGREGATION METHOD: Boot-Feibes-Lisman
****************************************************
Number of low-frequency observations  :    5
Frequency conversion                  :   12
Number of high-frequency observations :   60
----------------------------------------------------
Degree of differencing                :    2
Type of disaggregation: sum (flow).
----------------------------------------------------
Elapsed time:    0.0160
----------------------------------------------------
```



2.2. Stram-Wei (SW) method

The method proposed by Stram and Wei (1986) is more general than the one proposed by Boot et al. (1967). In this way, BFL is implemented via SW as a special case. The price to pay for this generalization is that we have to feed the SW function with an estimate of the variance-covariance (VCV) matrix of the (unobserved) high-frequency time series. The next box presents its structure, defining inputs and outputs.

Box 2.4: Stram-Wei function

```
function res = sw(Y,ta,d,sc,v)
% PURPOSE: Temporal disaggregation using the Stram-Wei method.
% ------------------------------------------------------------------------
% SYNTAX: res = sw(Y,ta,d,sc,v);
% ------------------------------------------------------------------------
% OUTPUT: res: a structure
%         res.meth  = 'Stram-Wei'
%         res.N:    = Number of low frequency data
%         res.ta    = Type of disaggregation
%         res.d     = Degree of differencing
%         res.sc    = Frequency conversion
%         res.H     = nxN temporal disaggregation matrix
%         res.y     = High frequency estimate
%         res.et    = Elapsed time
% ------------------------------------------------------------------------
% INPUT: Y: Nx1 ---> vector of low frequency data
%        ta: type of disaggregation
%            ta=1 ---> sum (flow)
%            ta=2 ---> average (index)
%            ta=3 ---> last element (stock) ---> interpolation
%            ta=4 ---> first element (stock) ---> interpolation
%        d: number of unit roots
%        sc: number of high frequency data points for each low frequency data point
%            Some examples:
%            sc= 4 ---> annual to quarterly
%            sc=12 ---> annual to monthly
%            sc= 3 ---> quarterly to monthly
%        v: (n-d)x(n-d) VCV matrix of high frequency stationary series
% ------------------------------------------------------------------------
% LIBRARY: aggreg, aggreg_v, dif, movingsum
% ------------------------------------------------------------------------
% SEE ALSO: bfl, tduni_print, tduni_plot
% ------------------------------------------------------------------------
% REFERENCE: Stram, D.O. and Wei, W.W.S. (1986) "A methodological note on the
% disaggregation of time series totals", Journal of Time Series Analysis,
% vol. 7, n. 4, p. 293-302.
```

The next box presents a script for running `sw()` and getting its output (text file and graphics).

Box 2.5: Stram-Wei script

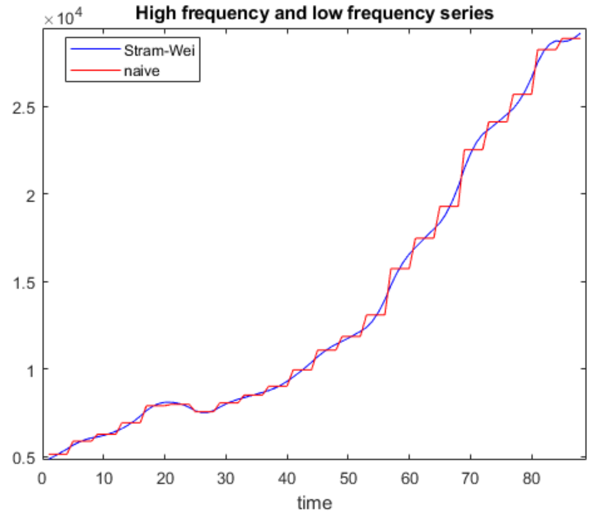
```
% PURPOSE: demo of sw()
%         Temporal disaggregation using the Stram-Wei method
%---------------------------------------------------
% USAGE: sw_d
%---------------------------------------------------
close all; clear all; clc;
% Low-frequency data: Spain's Exports of Goods. 1995 prices
Y =[ 20499
     23477
       …
       …
       …
```

```
      …
    115573 ];
% ---------------------------------------------
% Inputs for td library
% Type of aggregation
ta = 1;
% Minimizing the volatility of d-differenced series
d = 2;
% Frequency conversion
sc = 4;
% Number of observations of low-frequency input
N = length(Y);
% Number of observations of high-frequency output
n = sc*N;
% Defining the VCV matrix of stationary high-frequency time series
% Assumption of the example: IMA(d,2). For a comprehensive and more general analysis
% please consult Stram and Wei (1986)"Temporal aggregation in the ARIMA process",
% Journal of Time Series Analysis, vol. 7, núm. 4, p. 279-292.
% MA parameters
th1 =  0.9552;
th2 = -0.0015;
va = 0.87242 * ((223.5965)^2);
% ACF values
acf0 = va * (1+th1^2+th2^2);
acf1 = -va * th1 * (1-th2);
acf2 = -va * th2;
% Auxiliary vectors
a0(1:n-d) = acf0;
a1(1:n-d-1) = acf1;
a2(1:n-d-2) = acf2;
% High-frequency VCV matrix
v = 0.5*diag(a0) + diag(a1,-1) + diag(a2,-2);
v = v + tril(v)';
% Calling the function: output is loaded in a structure called res
res = sw(Y,ta,d,sc,v);
% ---------------------------------------------
% Outputs
% Printed output
file_sal = 'sw.out';
tdprint(res,file_sal);
edit sw.out;
% Graphics
tdplot(res);
```

The output of the `sw()` procedure is presented in the next box.

Box 2.6: Stram-Wei output

```
 ****************************************************
 TEMPORAL DISAGGREGATION METHOD: Stram-Wei
 ****************************************************
 Number of low-frequency observations  :   22
 Frequency conversion                  :    4
 Number of high-frequency observations :   88
 ----------------------------------------------------
 Degree of differencing                :    2
 Type of disaggregation: sum (flow).
 ----------------------------------------------------
 Elapsed time:   0.0160
 ----------------------------------------------------
```



2.3. Low-pass interpolation

Temporal disaggregation without the aid of high-frequency trackers can be considered as a sort of interpolation applied to a moving sum. This function is reminiscent affine of Sims (1974), combining non-informative interpolation with low-pass filtering, see also Wei (1990). The procedure has three steps:

- Raw interpolation: padding the low-frequency benchmark with zeros and scaling it.
- Low-pass smoothing by means of the Hodrick-Prescott filter.
- Enforcing consistency with the annual counterpart by means of benchmarking, using the Denton procedure (additive variant).

The inputs and outputs of the function are described in the next box:

Box 2.7: Low-pass interpolation function

```
function [y,w,x] = low_pass_interpolation(Y,ta,d,sc,lambda)
% PURPOSE: Low-pass interpolation using Hodrick-Prescott and Denton
% ------------------------------------------------------------------------
% SYNTAX: [y,w,x] = low_pass_interpolation(Y,ta,d,sc,lambda);
% ------------------------------------------------------------------------
% OUTPUT: y: nx1 ---> final interpolation
%         w: nx1 ---> intermediate interpolation (low-pass filtering of x)
%         x: nx1 ---> initial interpolation (padding Y with zeros)
% ------------------------------------------------------------------------
% INPUT: Y: Nx1 ---> vector of low frequency data
%        ta: 1x1 type of disaggregation
%            ta=1 ---> sum (flow)
%            ta=2 ---> average (index)
%            ta=3 ---> last element (stock) ---> interpolation
%            ta=4 ---> first element (stock) ---> interpolation
%        d: 1x1 objective function to be minimized: volatility of ...
%            d=0 ---> levels
%            d=1 ---> first differences
%            d=2 ---> second differences
%        sc: 1x1 number of high frequency data points for each low frequency data point
%            Some examples:
%            sc= 4 ---> annual to quarterly
%            sc=12 ---> annual to monthly
%            sc= 3 ---> quarterly to monthly
%        lambda: 1x1 --> balance between adjustment and smoothness (HP
```

```
%        low-pass filter)
% ------------------------------------------------------------------------
% LIBRARY: copylow, hp, denton
% ------------------------------------------------------------------------
% SEE ALSO: bfl, sw
% ------------------------------------------------------------------------
```

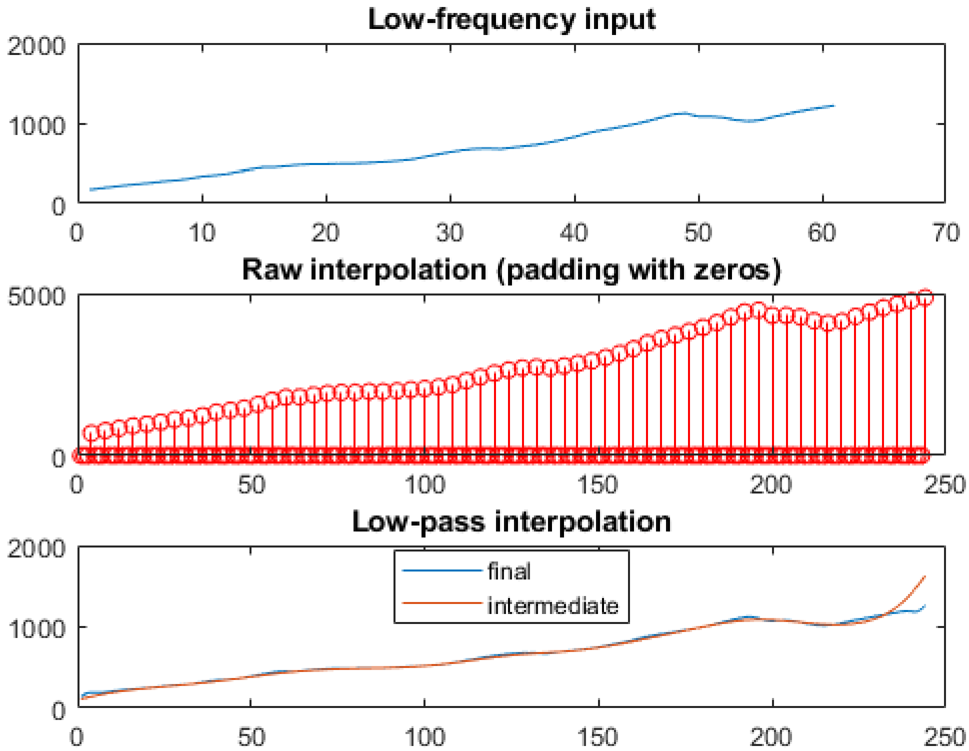
The implementation of the low-pass interpolation is described in the next box:

Box 2.8: Low-pass interpolation script

```
% PURPOSE: demo of low_pass_interpolation()
%          Temporal disaggregation using low-pass filtering
%          Low-pass filter = Hodrick-Prescott
% ----------------------------------------------------------------------
% USAGE: low_pass_interpolation_d
% ----------------------------------------------------------------------

clear all; close all; clc;
% ----------------------------------------------------------------------
% Annual GDP. Spain. 1960-2020. (AMECO Database).
load Spain_GDP;
Z = Y;
% ----------------------------------------------------------------------
% Sample conversion
sc = 4;
% ----------------------------------------------------------------------
% Hodrick-Prescott parameter
lambda = 1600;
% ----------------------------------------------------------------------
% Denton parameters:
% Type of aggregation
ta = 2;
% Minimizing the volatility of d-differenced series
d = 1;
% ----------------------------------------------------------------------
% Calling function
[z,w,x] = low_pass_interpolation(Z,ta,d,sc,lambda);
% ----------------------------------------------------------------------
% Plots
subplot(3,1,1);
plot(T,Z);
    title('LOW-FREQUENCY INPUT');
subplot(3,1,2);
stem(x,'r');
    title ('RAW INTERPOLATION (padding with zeros)');
subplot(3,1,3);
plot([z w]);
    legend('final','intermediate','Location','best');
    title('LOW-PASS INTERPOLATION')
```

The outputs of the procedure (raw interpolation and final interpolation) are depicted in the next graphs:

Figure 2.1: Low-pass interpolation output



2.4. Chow-Lin method without indicators

The Chow-Lin method, the workhorse of the model-based methods for temporal disaggregation with indicators, can also be used to perform temporal disaggregation without indicators. The trick consists of using an inert time index as the high-frequency tracker. The main advantage of using Chow-Lin in this way is the computation of confidence intervals for the temporally disaggregated time series. The use of the Chow-Lin procedure will be explained at length later, so we present here the script only:

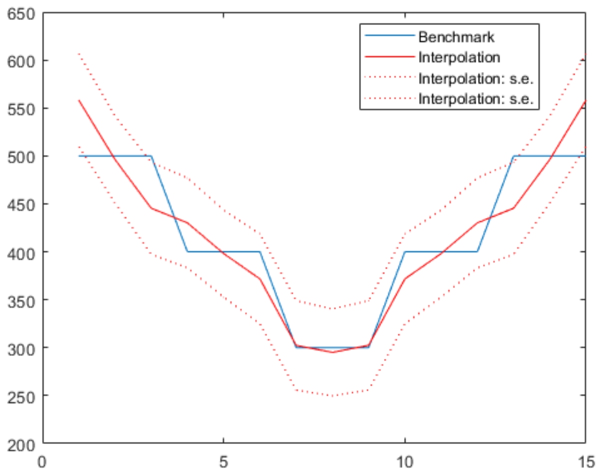
Box 2.9: Chow-Lin without indicators script

```
% PURPOSE: Demo of chowlin_uni()
%           Temporal disaggregation without indicator and Chow-Lin.
%---------------------------------------------------
% USAGE: chowlin_uni_d
%---------------------------------------------------

close all; clear all; clc;

% Low-frequency data: Denton's benchmark Y
load denton;

% ---------------------------------------------
% Inputs
% ---------------------------------------------
% Type of aggregation.
ta = 2;
% Frequency conversion.
sc = 3;
% Model for the indicator: 1 => linear trend, 2 => quadratic trend
opx = 2;
% Calling the function: output is loaded in a structure called res.
```

```
res = uni_chowlin(Y,ta,sc,opx);
% Calling printing function.
file_out = 'td.out';
tdprint(res,file_out);
edit td.out;
% Graphs
t = 1:res.n;
plot(t,copylow(Y,1,sc),t,res.y,'r-',t,res.y_lo,'r:',t,res.y_up,'r:');
    legend('Benchmark','Interpolation','Interpolation: s.e.','Interpolation: …
       s.e.','Location','best');
```

The graphic output of the procedure (temporal disaggregation and $\pm\sigma$ confidence interval) is depicted in the next graph:

Figure 2.2: Low-pass interpolation output



3. TEMPORAL DISAGGREGATION WITH HIGH-FREQUENCY INDICATORS

There are several methods to perform temporal disaggregation when the information set includes also a set of high-frequency trackers in addition to the low-frequency benchmark. This library includes the following methods:

- Denton method (additive and proportional variant): `denton()`
- Chow-Lin by Maximum Likelihood (ML) and Weighted Least Squares (WLS): `chowlin()`
- Fernández: `fernandez()`
- Litterman by ML and WLS: `litterman()`
- Santos-Silva and Cardoso by ML and WLS: `ssc()`
- ADL(1,1) model-based Proietti by ML and WLS: `proietti()`
- ARIMA model-based Guerrero: `guerrero()`

3.1 Quadratic optimization methods: Denton

The approach followed by BFL and SW can be easily extended to the case where the information set includes also a high-frequency tracker. This is precisely the approach followed by Denton (1971), see also Cholette (1984) and Di Fonzo and Marini (2012). The inputs and outputs of the Denton function are described in the next box:

Box 3.1: Denton method function

```
function res = denton(Y,x,ta,d,sc,op1)
% PURPOSE: Temporal disaggregation using the Denton method.
% -----------------------------------------------------------------------
% SYNTAX: res = denton(Y,x,ta,d,sc,op1,op2);
% -----------------------------------------------------------------------
% OUTPUT: res: a structure
%         res.meth  = 'Denton';
%         res.N     = Number of low frequency data
%         res.ta    = Type of disaggregation
%         res.sc    = Frequency conversion
%         res.d     = Degree of differencing
%         res.y     = High frequency estimate
%         res.x     = High frequency indicator
%         res.U     = Low frequency residuals
%         res.u     = High frequency residuals
%         res.et    = Elapsed time
% -----------------------------------------------------------------------
% INPUT: Y: Nx1 ---> vector of low frequency data
%        x: nx1 ---> vector of low frequency data
%        ta: type of disaggregation
%            ta=1 ---> sum (flow)
%            ta=2 ---> average (index)
%            ta=3 ---> last element (stock) ---> interpolation
%            ta=4 ---> first element (stock) ---> interpolation
%        d: objective function to be minimized: volatility of ...
```

```
%             d=1 ---> first differences
%             d=2 ---> second differences
%         sc: number of high frequency data points for each low frequency data point
%             Some examples:
%             sc= 4 ---> annual to quarterly
%             sc=12 ---> annual to monthly
%             sc= 3 ---> quarterly to monthly
%         op1: additive variant (1) or proportional variant(2)
% -----------------------------------------------------------------------
% LIBRARY: aggreg, dif
% -----------------------------------------------------------------------
% SEE ALSO: tdprint, tdplot
% -----------------------------------------------------------------------
% REFERENCE: Denton, F.T. (1971) "Adjustment of monthly or quarterly
% series to annual totals: an approach based on quadratic minimization",
% Journal of the American Statistical Society, vol. 66, n. 333, p. 99-102.
```

The next box shows the application of the Denton method by means of a script.

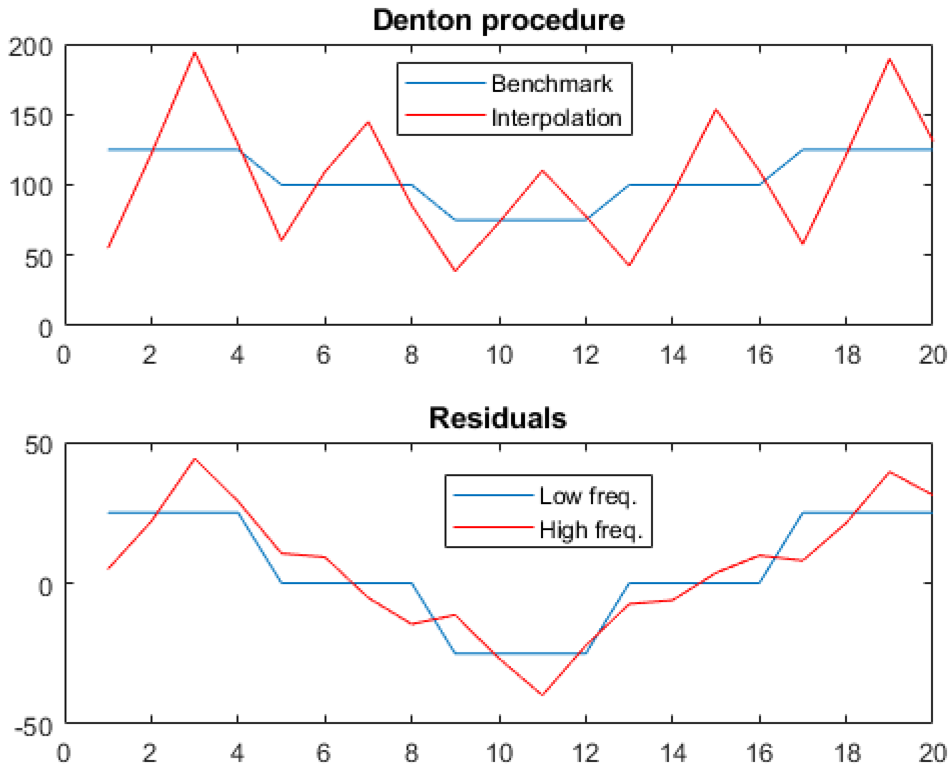
Box 3.2: Denton method scripts

```
% PURPOSE: Demo of denton()
%          Temporal disaggregation with indicator.
%          Denton method, additive and proportional variants.
%---------------------------------------------------
% USAGE: denton_d
%---------------------------------------------------

close all; clear all; clc;

% Loading data
load denton;

% ---------------------------------------------
% Inputs
% ---------------------------------------------
% Type of aggregation.
ta = 1;
% Minimizing the volatility of d-differenced series.
d = 2;
% Frequency conversion.
sc = 4;
% Variant: 1=additive, 2=proportional.
op1 = 2;
% Calling the function: output is loaded in a structure called res.
res = denton(Y,x,ta,d,sc,op1);
% Calling printing function.
% Name of ASCII file for output.
file_out = 'denton.out';
tdprint(res,file_out);
edit denton.out;
% Final plots
figure;
t = 1:length(res.y);
subplot(2,1,1)
plot(t,copylow(Y,2,sc),t,res.y,'r-');
    title('Low frequency input');
    legend('Benchmark','Interpolation');
    title('Denton procedure');
subplot(2,1,2)
plot(t,copylow(res.U,2,sc),t,res.u,'r-');
    legend('Low freq.','High freq.');
    title('Residuals');
```

The output of the procedure (log file and graphs) is depicted in the next box:

Box 3.3: Denton method output

```
****************************************************
 TEMPORAL DISAGGREGATION METHOD: Denton
 Type: Proportional variant
 ****************************************************
 Number of low-frequency observations  :    5
 Frequency conversion                  :    4
 Number of high-frequency observations :   20
 Number of extrapolations              :    0
 ----------------------------------------------------
 Degree of differencing                :    2
 Type of disaggregation: sum (flow).
----------------------------------------------------
Elapsed time:   0.0000
----------------------------------------------------
```



3.2 Model-based methods: Chow-Lin, Fernández and Litterman

We have written a set of functions that implement a model-based approach to temporal disaggregation using as inputs one or several high-frequency trackers and sharing a characterization that confines the dynamics to the innovation term: Chow-Lin (1971), Fernández (1981) and Litterman (1983). The Chow-Lin function considers both the Maximum Likelihood (ML) approach suggested by Bournay and Laroque (1979) and the Weighted Least Squares (WLS) proposal of Barbone et al. (1981).

The inputs and outputs of the Chow-Lin function are described in the next box:

Box 3.4: Chow-Lin function

```
function res = chowlin(Y,x,ta,sc,type,opC,rl)
% PURPOSE: Temporal disaggregation using the Chow-Lin method
% ------------------------------------------------------------
% SYNTAX: res = chowlin(Y,x,ta,sc,type,opC,rl);
% ------------------------------------------------------------
% OUTPUT: res: a structure
%           res.meth    ='Chow-Lin';
%           res.ta      = type of disaggregation
%           res.type    = method of estimation
%           res.opC     = option related to intercept
%           res.N       = nobs. of low frequency data
%           res.n       = nobs. of high-frequency data
%           res.pred    = number of extrapolations
%           res.sc      = frequency conversion between low and high freq.
%           res.p       = number of regressors (including intercept)
%           res.Y       = low frequency data
%           res.x       = high frequency indicators
%           res.y       = high frequency estimate
%           res.y_dt    = high frequency estimate: standard deviation
%           res.y_lo    = high frequency estimate: sd - sigma
%           res.y_up    = high frequency estimate: sd + sigma
%           res.u       = high frequency residuals
%           res.U       = low frequency residuals
%           res.beta    = estimated model parameters
%           res.beta_sd = estimated model parameters: standard deviation
%           res.beta_t  = estimated model parameters: t ratios
%           res.rho     = innovational parameter
%           res.sigma_a = variance of shocks
%           res.aic     = Information criterion: AIC
%           res.bic     = Information criterion: BIC
%           res.val     = Objective function used by the estimation method
%           res.wls     = Weighted least squares as a function of rho
%           res.loglik  = Log likelihood as a function of rho
%           res.r       = grid of innovational parameters used by the estimation method
% ------------------------------------------------------------
% INPUT: Y: Nx1 ---> vector of low frequency data
%        x: nxp ---> matrix of high frequency indicators (without intercept)
%        ta: type of disaggregation
%            ta=1 ---> sum (flow)
%            ta=2 ---> average (index)
%            ta=3 ---> last element (stock) ---> interpolation
%            ta=4 ---> first element (stock) ---> interpolation
%        sc: number of high frequency data points for each low frequency data points
%            Some examples:
%            sc= 4 ---> annual to quarterly
%            sc=12 ---> annual to monthly
%            sc= 3 ---> quarterly to monthly
%        type: estimation method:
%            type=0 ---> weighted least squares
%            type=1 ---> maximum likelihood
%        opC: 1x1 option related to intercept
%            opc = -1 : pretest intercept significance
%            opc =  0 : no intercept in hf model
%            opc =  1 : intercept in hf model
%        rl: innovational parameter
%            rl = []: 0x0 ---> rl=[0.05 0.99], 50 points grid
%            rl: 1x1 ---> fixed value of rho parameter
%            rl: 1x3 ---> [r_min r_max n_grid] search is performed
%                on this range, using a n_grid points grid
% ------------------------------------------------------------
% LIBRARY: chowlin_W
% ------------------------------------------------------------
% SEE ALSO: litterman, fernandez, td_plot, td_print, chowlin_co
% ------------------------------------------------------------
% REFERENCE: Chow, G. and Lin, A.L. (1971) "Best linear unbiased
% distribution and extrapolation of economic time series by related
% series", Review of Economic and Statistics, vol. 53, n. 4, p. 372-375.
% Bournay, J. and Laroque, G. (1979) "Reflexions sur la methode
% d'elaboration des comptes trimestriels", Annales de l'INSEE, n. 36, p. 3-30.
```

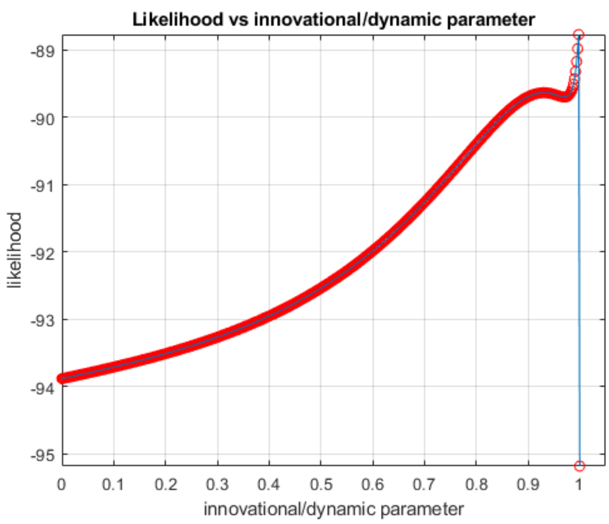
The next box shows the application of the Chow-Lin method.

Box 3.5: Chow-Lin script

```
% PURPOSE: demo of chowlin()
%          Temporal disaggregation with indicators.
%          Chow-Lin method
%---------------------------------------------------
% USAGE: chowlin_d
%---------------------------------------------------

close all; clear all; clc;

% Loading data
load bournay_laroque;

% --------------------------------------------
% Inputs
% Type of aggregation
ta = 1;
% Frequency conversion
sc = 4;
% Method of estimation
type = 1;
% Intercept
opC = -1;
% Interval of rho for grid search
% rl = []; %Default: search on [0.05 0.99] with 100 grid points
% rl = 0.57; %Fixed value
rl = [0.0 0.999999999 500];
% Calling the function: output is loaded in a structure called res
res = chowlin(Y,x,ta,sc,type,opC,rl);
% Printed output
file_out = 'chowlin.out';
tdprint(res,file_out);
edit chowlin.out;
% Graphs
tdplot(res);
```

As can be seen in the next graph, the likelihood profile changes abruptly near 1. For this reason, we have run again the previous script setting rl=[0.90 0. 999999999 500].

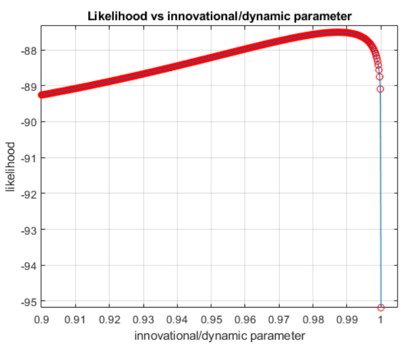
Figure 3.1: Chow-Lin method: likelihood profile

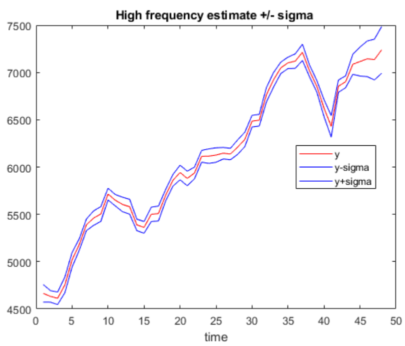


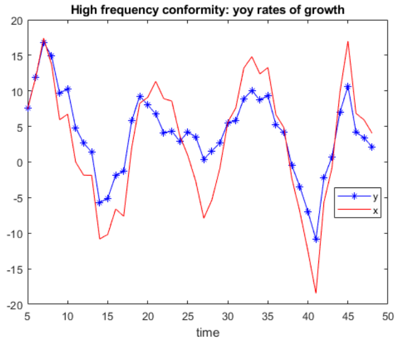
The output of the procedure (log file and graphs) is depicted in the next box:

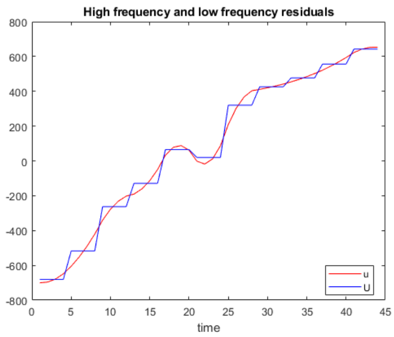
Box 3.6: Chow-Lin output

```
*****************************************************
TEMPORAL DISAGGREGATION METHOD: Chow-Lin
*****************************************************
Number of low-frequency observations :   11
Frequency conversion                 :    4
Number of high-frequency observations:   48
Number of extrapolations             :    4
Number of indicators                 :    1
-----------------------------------------------------
Type of disaggregation: sum (flow).
-----------------------------------------------------
Estimation method: Maximum likelihood.
-----------------------------------------------------
** High frequency model **
 Beta parameters (columnwise):
  * Estimate
  * Std. deviation
  * t-ratios
-----------------------------------------------------
  38.3774          6.3041          6.0877
-----------------------------------------------------
Innovational parameter:   0.9985
-----------------------------------------------------
AIC:   9.5853
BIC:   9.6576
-----------------------------------------------------
Low-frequency correlation (Y,X)
 - levels     : 0.9357
 - yoy rates  : 0.9153
-----------------------------------------------------
High-frequency correlation (y,x)
 - levels     : 0.9352
 - yoy rates  : 0.9320
-----------------------------------------------------
High-frequency volatility of yoy rates
 - estimate   : 5.6959
 - indicator  : 8.4992
 - ratio      : 0.6702
-----------------------------------------------------
High-frequency correlation (y,x*beta)
 - levels     : 0.9352
 - yoy rates  : 0.9320
-----------------------------------------------------
Elapsed time:   0.0780
```









The function that implements the method of and Litterman (1983) has the same inputs as the corresponding Chow-Lin function, including its ML and WLS estimation. The following box shows a script for running Litterman.

Box 3.7: Litterman script

```
% PURPOSE: demo of chowlin()
%          Temporal disaggregation with indicators.
%          Chow-Lin method
%---------------------------------------------------
% USAGE: chowlin_d
%---------------------------------------------------

close all; clear all; clc;

% Loading data
load bournay_laroque;

% ---------------------------------------------
% Inputs
% Type of aggregation
ta = 1;
% Frequency conversion
sc = 4;
% Method of estimation
type = 1;
% Intercept
opC = -1;
% Interval of rho for grid search
% rl = []; %Default: search on [0.05 0.99] with 100 grid points
% rl = 0.57; %Fixed value
rl = [0.00 0.99 500];
% Calling the function: output is loaded in a structure called res
res = litterman(Y,x,ta,sc,type,opC,rl);
% Printed output
file_out = 'litterman.out';
tdprint(res,file_out);
edit litterman.out;
% Graphs
tdplot(res);
```

Finally, the Fernández (1981) procedure can be applied using the next script:

Box 3.8: Fernández script

```
% PURPOSE: demo of chowlin()
%          Temporal disaggregation with indicators.
%          Chow-Lin method
%---------------------------------------------------
% USAGE: chowlin_d
%---------------------------------------------------

close all; clear all; clc;

% Loading data
load bournay_laroque;

% ---------------------------------------------
% Inputs
% Type of aggregation
ta = 1;
% Frequency conversion
sc = 4;
% Intercept
opC = -1;
% Calling the function: output is loaded in a structure called res
res = fernandez(Y,x,ta,sc,opC);
% Printed output
file_out = 'fernandez.out';
tdprint(res,file_out);
edit fernandez.out;
% Graphs
tdplot(res);
```

3.3 Model-based methods with explicit dynamics: Santos-Cardoso and Proietti

We present two functions now that implements an explicitly dynamic approach: Santos-Cardoso (2001) and Proietti (2006), see also Di Fonzo (2002).

The inputs and outputs of the Santos-Cardoso function are described in the next box:

Box 3.9: Santos-Cardoso function

```
function res = ssc(Y,x,ta,sc,type,opC,rl)
% PURPOSE: Temporal disaggregation using the dynamic Chow-Lin method
%          proposed by Santos-Silva & Cardoso (2001).
% ------------------------------------------------------------
% SYNTAX: res = ssc(Y,x,ta,sc,type,opC,rl);
% ------------------------------------------------------------
% OUTPUT: res: a structure
%           res.meth    ='Santos Silva-Cardoso';
%           res.ta      = type of disaggregation
%           res.type    = method of estimation
%           res.opC     = option related to intercept
%           res.N       = nobs. of low frequency data
%           res.n       = nobs. of high-frequency data
%           res.pred    = number of extrapolations
%           res.sc      = frequency conversion between low and high freq.
%           res.p       = number of regressors (including intercept)
%           res.Y       = low frequency data
%           res.x       = high frequency indicators
%           res.y       = high frequency estimate
%           res.y_dt    = high frequency estimate: standard deviation
%           res.y_lo    = high frequency estimate: sd - sigma
```

```
%            res.y_up    = high frequency estimate: sd + sigma
%            res.u       = high frequency residuals
%            res.U       = low frequency residuals
%            res.gamma    = estimated model parameters (including y(0))
%            res.gamma_sd = estimated model parameters: standard deviation
%            res.gamma_t  = estimated model parameters: t ratios
%            res.phi      = dynamic parameter phi
%            res.beta     = estimated model parameters (excluding y(0))
%            res.beta_sd = estimated model parameters: standard deviation
%            res.beta_t  = estimated model parameters: t ratios
%            res.phi     = innovational parameter
%            res.aic     = Information criterion: AIC
%            res.bic     = Information criterion: BIC
%            res.val     = Objective function used by the estimation method
%            res.wls     = Weighted least squares as a function of phi
%            res.loglik  = Log likelihood as a function of phi
%            res.r       = grid of innovational parameters used by the estimation method
%            res.et       = elapsed time
% ------------------------------------------------------------
% INPUT: Y: Nx1 ---> vector of low frequency data
%        x: nxp ---> matrix of high frequency indicators (without intercept)
%        ta: type of disaggregation
%            ta=1 ---> sum (flow)
%            ta=2 ---> average (index)
%            ta=3 ---> last element (stock) ---> interpolation
%            ta=4 ---> first element (stock) ---> interpolation
%        sc: number of high frequency data points for each low frequency data points
%            Some examples:
%            sc= 4 ---> annual to quarterly
%            sc=12 ---> annual to monthly
%            sc= 3 ---> quarterly to monthly
%        type: estimation method:
%            type=0 ---> weighted least squares
%            type=1 ---> maximum likelihood
%        opC: 1x1 option related to intercept
%            opc = -1 : pretest intercept significance
%            opc = 0 : no intercept in hf model
%            opc = 1 : intercept in hf model
%        rl: innovational parameter
%            rl = []: 0x0 ---> rl=[0.05 0.99], 50 points grid
%            rl: 1x1 ---> fixed value of rho parameter
%            rl: 1x3 ---> [r_min r_max n_grid] search is performed
%                on this range, using a n_grid points grid
% ------------------------------------------------------------
% LIBRARY: ssc_W
% ------------------------------------------------------------
% SEE ALSO: chowlin, litterman, fernandez, td_plot, td_print
% ------------------------------------------------------------
% REFERENCE: Santos-Silva, J.M.C. and Cardoso, F.(2001) "The Chow-Lin method
% using dynamic models", Economic Modelling, vol. 18, p. 269-280.
% Di Fonzo, T. (2002) "Temporal disaggregation of economic time series:
% towards a dynamic extension", Dipartimento di Scienze Statistiche,
% Universita di Padova, Working Paper n. 2002-17.
```

The next box shows the application of the Santos-Cardoso method.

Box 3.10: Santos-Cardoso script

```
% PURPOSE: demo of ssc()
%          Temporal disaggregation with indicators.
%          Santos-Silva & Cardoso method
%---------------------------------------------------
% USAGE: ssc_d
%---------------------------------------------------

close all; clear all; clc;

% Loading data
load ssc;

% ---------------------------------------------
% Inputs
```

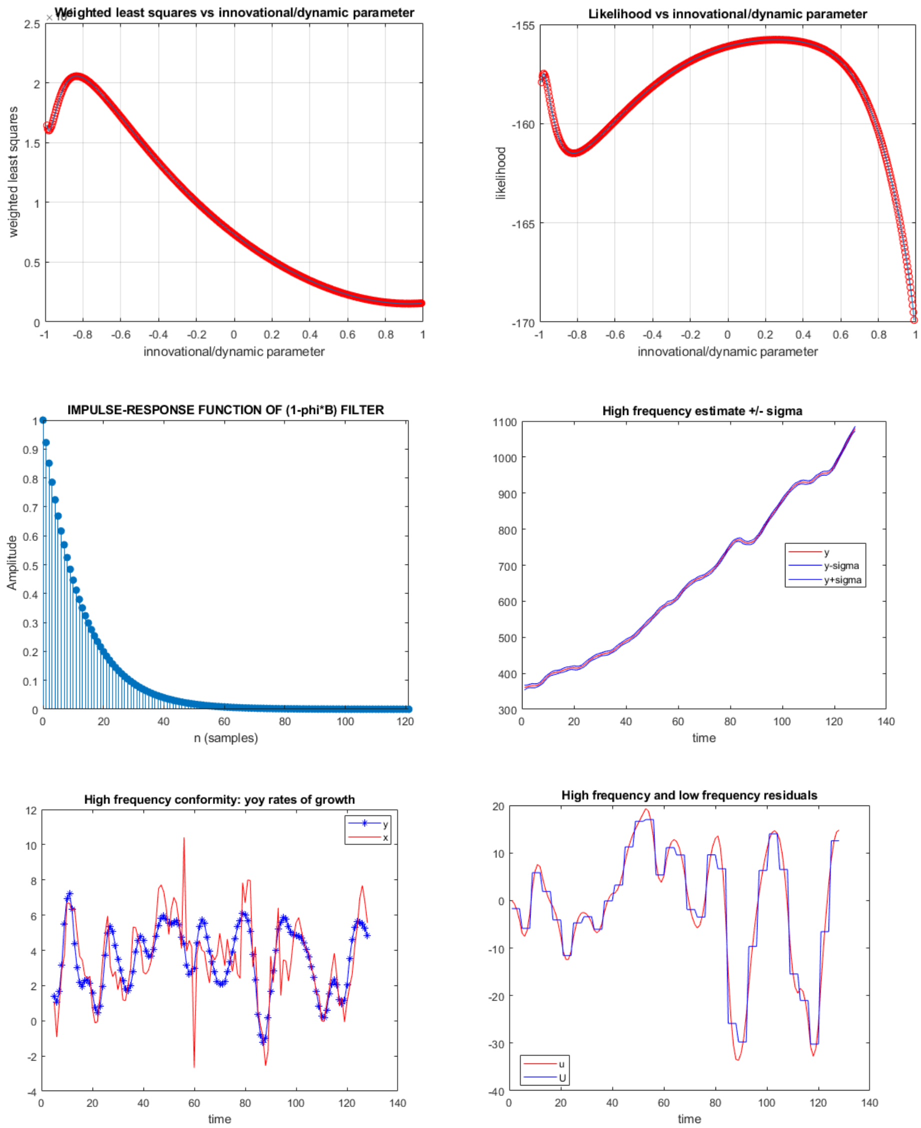
```
% Type of aggregation
ta=1;
% Frequency conversion
s=4;
% Method of estimation
type=0;
% Intercept
opC = 1;
% Interval of rho for grid search
% rl = [];
% rl = 0.57;
rl = [-0.99 0.99 500];

% Note: the grid search applied in the ssc procedure generates
% a warning when phi=0. This warning is muted.
warning off MATLAB:nearlySingularMatrix
% Calling the function: output is loaded in a structure called res
res = ssc(Y,x,ta,s,type,opC,rl);
warning on MATLAB:nearlySingularMatrix

% Printed output
file_out = 'ssc.out';
tdprint(res,file_out);
edit ssc.out;

% Graphs
tdplot(res);
```

The output of the procedure (log file and graphs) is depicted in the next box:

Box 3.11: Santos-Cardoso output

```
*******************************************************
TEMPORAL DISAGGREGATION METHOD: Santos-Silva & Cardoso
*******************************************************
Number of low-frequency observations :   32
Frequency conversion                 :    4
Number of high-frequency observations:  128
Number of extrapolations             :    0
Number of indicators                 :    2
----------------------------------------------------
Type of disaggregation: sum (flow).
----------------------------------------------------
Estimation method: Weighted least squares.
----------------------------------------------------
** High frequency model **
 Beta parameters (columnwise):
  * Estimate
  * Std. deviation
  * t-ratios
----------------------------------------------------
    0.6275          2.1575          0.2909
    0.0766          0.0028         27.1521
----------------------------------------------------
Dynamic parameter:   0.9225
----------------------------------------------------
Long-run beta parameters (columnwise):
    8.1016
    0.9884
----------------------------------------------------
Truncation remainder: expected y(0):
  * Estimate
  * Std. deviation
  * t-ratios
----------------------------------------------------
 117.2090          8.5049         13.7813
----------------------------------------------------
AIC:   4.1536
BIC:   4.2910
```

```
 ----------------------------------------------------
 Low-frequency correlation (Y,X)
  - levels     : 0.9994
  - yoy rates  : 0.8561
 ----------------------------------------------------
 High-frequency correlation (y,x)
  - levels     : 0.9990
  - yoy rates  : 0.7448
 ----------------------------------------------------
 High-frequency volatility of yoy rates
  - estimate   : 1.8542
  - indicator  : 2.3430
  - ratio      : 0.7914
 ----------------------------------------------------
 High-frequency correlation (y,x*beta)
  - levels     : 0.9990
  - yoy rates  : 0.7447
 ----------------------------------------------------
Elapsed time:   0.5630
```



Proietti (2006) extended the Santos-Cardoso method, using an first-order Autoregressive Dynamic Linear model, ADL(1,1), of an ADL(1,0) model. We have implemented the Proietti model using a shortcut that represents it in a matrix format like the one used for the Santos-Cardoso method. Hence, the script for running Proietti is quite similar to the one used for running Santos-Cardoso, as can be seen in the next box:

Box 3.12: Proietti script

```
% PURPOSE: demo of proietti()
%          Temporal disaggregation with indicators.
%          ADL(1,1) model.
%---------------------------------------------------
% USAGE: proietti_d
%---------------------------------------------------

close all; clear all; clc;

% Loading data
load ssc;

% ---------------------------------------------
% Inputs

% Type of aggregation
ta = 1;

% Frequency conversion
s = 4;

% Method of estimation
type = 1;

% Intercept
opC = 1;

% Interval of rho for grid search
% rl = [];
% rl = 0.57;
rl = [-0.99 0.99 500];

% Note: the grid search applied in the proietti procedure generates
% a warning when phi=0. This warning is muted.
warning off MATLAB:nearlySingularMatrix
% Calling the function: output is loaded in a structure called res
res = proietti(Y,x,ta,s,type,opC,rl);
warning on MATLAB:nearlySingularMatrix

% Printed output
file_out = 'proietti.out';
tdprint(res,file_out);
edit proietti.out;

% Graphs
tdplot(res);
```

3.4 ARIMA Model-based methods: Guerrero

The starting point of the method proposed by Guerrero (1990) is the assumption that the unobservable high-frequency counterparty y of a low-frequency benchmark can be represented by means of a general multiplicative ARIMA model.

Guerrero's method solves the benchmarking problem using also the information available in a set of k high-frequency indicators using also a BLUE approach. The method can be stated using the following algorithm:

- Estimation of the scaled indicator, by means of OLS on the low-frequency model.
- Preliminary estimator, based on the information provided by the ARIMA model for the scaled indicator.
- Final estimator, adding the information provided by the model for the high-frequency discrepancy.

The inputs and outputs of the Guerrero (1990) function are described in the next box[3]:

Box 3.13: Guerrero function

```
function res = guerrero(Y,x,ta,sc,rexw,rexd,opC)
% PURPOSE: ARIMA-based temporal disaggregation: Guerrero method
% ------------------------------------------------------------
% SYNTAX: res = guerrero(Y,x,ta,sc,rexw,rexd,opC);
% ------------------------------------------------------------
% OUTPUT: res: a structure
%         res.meth     ='Guerrero';
%         res.ta       = type of disaggregation
%         res.opC     = option related to intercept
%         res.N        = nobs. of low frequency data
%         res.n        = nobs. of high-frequency data
%         res.pred     = number of extrapolations
%         res.sc        = frequency conversion between low and high freq.
%         res.p        = number of regressors (+ intercept)
%         res.Y        = low frequency data
%         res.x        = high frequency indicators
%         res.w        = scaled indicator (preliminary hf estimate)
%         res.y1       = first stage high frequency estimate
%         res.y        = final high frequency estimate
%         res.y_dt     = high frequency estimate: standard deviation
%         res.y_lo     = high frequency estimate: sd - sigma
%         res.y_up     = high frequency estimate: sd + sigma
%         res.delta    = high frequency discrepancy (y1-w)
%         res.u        = high frequency residuals (y-w)
%         res.U        = low frequency residuals (Cu)
%         res.beta     = estimated parameters for scaling x
%         res.k        = statistic to test compatibility
%         res.et       = elapsed time
% ------------------------------------------------------------
% INPUT: Y: Nx1 ---> vector of low frequency data
%        x: nxp ---> matrix of high frequency indicators (without intercept)
%        ta: type of disaggregation
%            ta=1 ---> sum (flow)
%            ta=2 ---> average (index)
%            ta=3 ---> last element (stock) ---> interpolation
%            ta=4 ---> first element (stock) ---> interpolation
%        sc: number of high frequency data points for each low frequency data points
%            Some examples:
%            sc= 4 ---> annual to quarterly
%            sc=12 ---> annual to monthly
%            sc= 3 ---> quarterly to monthly
%        rexw, rexd ---> a structure containing the parameters of ARIMA model
%            for indicator and discrepancy, respectively (see calT function)
%        opC: 1x1 option related to intercept
%            opc = -1 : pretest intercept significance
```

[3] This function requires the `impz()` function from the Signal Processing Toolbox.

```
%            opc =  0 : no intercept in hf model
%            opc =  1 : intercept in hf model
% ------------------------------------------------------------
% LIBRARY: guerrero_W
% ------------------------------------------------------------
% SEE ALSO: chowlin, litterman, fernandez, td_print, td_plot
% ------------------------------------------------------------
% REFERENCE: Guerrero, V. (1990) "Temporal disaggregation of time
% series: an ARIMA-based approach", International Statistical
% Review, vol. 58, p. 29-46.
```

The next box shows the application of the Guerrero method.

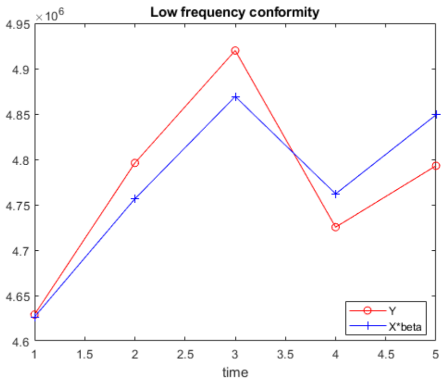
Box 3.14: Guerrero script

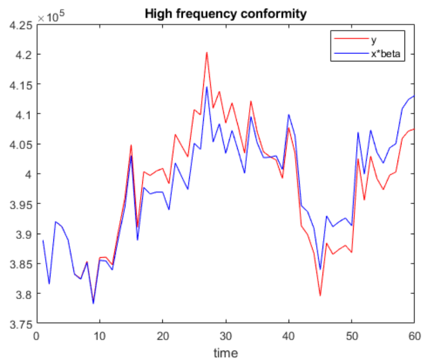
```
% PURPOSE: demo of guerrero()
%          Temporal disaggregation with indicators.
%          Guerrero ARIMA-based method
%---------------------------------------------------
% USAGE: guerrero_d
%---------------------------------------------------

clear all; clc; close all;

% ------------------------------------------------------------
% Loading data
load guerrero;

% ---------------------------------------------
% Inputs

% Type of aggregation
ta = 1;
% Frequency conversion
sc = 12;
% Intercept
opC = 1;

% Model for  w: (0,1,1)(1,0,1)
rexw.ar_reg = [1];
rexw.d  = 1;
rexw.ma_reg = [1 -0.40];

rexw.ar_sea = [1 0 0 0 0 0 0 0 0 0 0 0 -0.85];
rexw.bd = 0;
rexw.ma_sea = [1 0 0 0 0 0 0 0 0 0 0 0 -0.79];

rexw.sigma = 4968.716^2;

% Model for the discrepancy: (1,2,0)(1,0,0)
% See: Martinez and Guerrero, 1995, Test, 4(2), 359-76.

rexd.ar_reg = [1 -0.43];
rexd.d  = 2;
rexd.ma_reg = [1];

rexd.ar_sea = [1 0 0 0 0 0 0 0 0 0 0 0 0.62];
rexd.bd = 0;
rexd.ma_sea = [1];

rexd.sigma = 76.95^2;

% Calling the function: output is loaded in a structure called res
res = guerrero(Y,x,ta,sc,rexw,rexd,opC);

% Printed output
file_out = 'guerrero.out';
td_print(res,file_out);
edit guerrero.out;

% Graphs
td_plot(res);
```

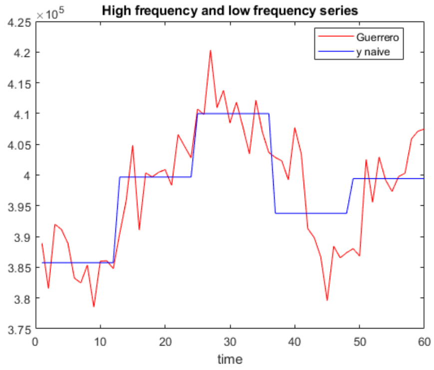
The output of the Guerrero procedure (log file and graphs) is depicted in the next box:

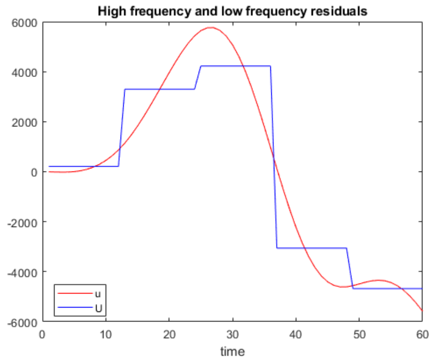
Box 3.15: Guerrero output

```
 ****************************************************
 TEMPORAL DISAGGREGATION METHOD: Guerrero
 ****************************************************
 Number of low-frequency observations :     5
 Frequency conversion                 :    12
 Number of high-frequency observations:    60
 Number of extrapolations             :     0
 Number of indicators                 :     2
 ----------------------------------------------------
 Type of disaggregation: sum (flow).
 ----------------------------------------------------
 Estimation method: BLUE.
----------------------------------------------------
 ** High frequency model **
  Beta parameters (columnwise):
   * Estimate
   * Std. deviation
   * t-ratios
 ----------------------------------------------------
219988.6766      49659.7689          4.4299
  1723.8723        481.2797          3.5819
----------------------------------------------------
 AIC:   7.5245
 BIC:   7.3683
 ----------------------------------------------------
Low-frequency correlation (Y,X)
  - levels      : 0.9003
  - yoy rates   : 0.9973
 ----------------------------------------------------
High-frequency correlation (y,x)
  - levels      : 0.9289
  - yoy rates   : 0.9835
 ----------------------------------------------------
High-frequency volatility of yoy rates
  - estimate    : 3.6623
  - indicator   : 6.2899
  - ratio       : 0.5823
 ----------------------------------------------------
High-frequency correlation (y,x*beta)
  - levels      : 0.9289
  - yoy rates   : 0.9832
 ----------------------------------------------------
Elapsed time:   0.0630
```






High frequency and low frequency series
Guerrero
y naive
time


High frequency and low frequency residuals
u
U
time

4. MULTIVARIATE METHODS

The evolution of temporal disaggregation is clearly related to its multivariate extension, including the incorporation of richer transversal constraints, see Rossi (1982), Di Fonzo (1990, 1994), Guerrero and Nieto (1999), Di Fonzo and Marini (2003, 2011) and Proietti (2011a) among others. In this library we have included several methods than consider explicitly several benchmarks and one cross section constraint. They are:

- Multivariate Denton: `denton_multi()`
- Rossi two-step multivariate procedure: `rossi()`
- Di Fonzo model-based BLUE approach: `difonzo()`

To illustrate the methods, we have used a simplified[4] example using regional data for Spain, see Cuevas et al. (2015) for additional details.

4.1. Quadratic optimization methods: Denton

The multivariate version of the Denton procedure can be considered as a straightforward extension of its univariate version. The corresponding function is:

Box 4.1: Multiple temporal disaggregation with a transversal constraint: multivariate Denton function

```
function res = denton_multi(Y,x,z,ta,sc,d,op1)
% PURPOSE: Multivariate temporal disaggregation with transversal
% constraint. Denton method, additive or proportional variants.
% ------------------------------------------------------------------------------
% SYNTAX: res = denton_multi(Y,x,z,ta,sc,d,op1);
% ------------------------------------------------------------------------------
% OUTPUT: res: a structure
%         res.meth  = 'Multivariate Denton';
%         res.N     = Number of low frequency data
%         res.n     = Number of high frequency data
%         res.pred  = Number of extrapolations (=0 in this case)
%         res.ta    = Type of disaggregation
%         res.sc     = Frequency conversion
%         res.d     = Degree of differencing
%         res.y     = High frequency estimate
%         res.z     = High frequency constraint
%         res.et    = Elapsed time
% -----------------------------------------------------------------------
% INPUT: Y: NxM ---> M series of low frequency data with N observations
%        x: nxM ---> M series of high frequency data with n observations
%        z: nx1 ---> high frequency transversal constraint
%        ta: type of disaggregation
%            ta=1 ---> sum (flow)
%            ta=2 ---> average (index)
%            ta=3 ---> last element (stock) ---> interpolation
%            ta=4 ---> first element (stock) ---> interpolation
%        sc: number of high frequency data points for each low frequency data points
%            Some examples:
%            sc= 4 ---> annual to quarterly
%            sc=12 ---> annual to monthly
```

[4] The simplification consists in considering 4 megaregions using a simple geographic criterion, instead of the real 17 Spanish regions and only one high-frequency tracker instead of several trackers.

```
%           sc= 3 ---> quarterly to monthly
%        d: objective function to be minimized: volatility of ...
%           d=0 ---> levels
%           d=1 ---> first differences
%           d=2 ---> second differences
%        op1: additive (1) or proportional (2) variant [optional, default=1]
% -----------------------------------------------------------------------
% LIBRARY: aggreg, aggreg_v, dif, vec, desvec
% -----------------------------------------------------------------------
% SEE ALSO: rossi, difonzo
% -----------------------------------------------------------------------
% REFERENCE: Di Fonzo, T. (1994) "Temporal disaggregation of a system of
% time series when the aggregate is known: optimal vs. adjustment methods",
% INSEE-Eurostat Workshop on Quarterly National Accounts, Paris, December.
```

The implementation of the method is described in the following script:

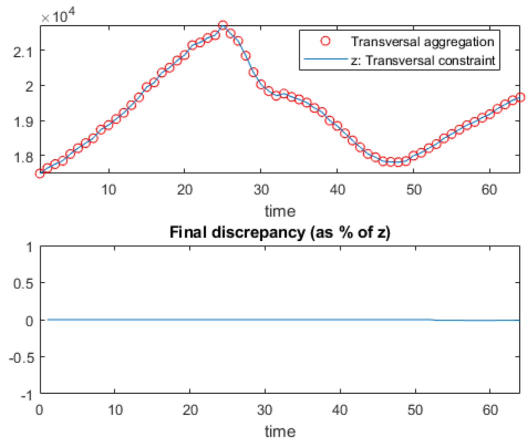
Box 4.2: Multiple temporal disaggregation with a transversal constraint: multivariate Denton script

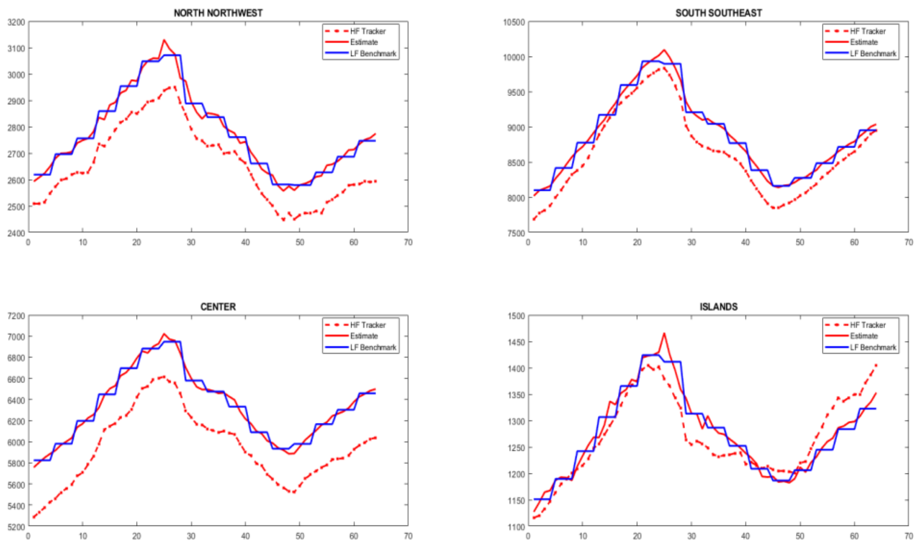
```
% PURPOSE: Demo of denton()
%          Temporal disaggregation with indicators.
%          Multivariate model with transversal constraint
%          Denton method, addititive or proportional variants.
%---------------------------------------------------
% USAGE: denton_d
%---------------------------------------------------

clc; close all; clear all;

% Loading data
load('Spain_employment_regional_4.mat')
x(end-2:end,:) = [];    %No free extrapolations
z(end-2:end) = [];      %No free extrapolations

% --------------------------------------------
% Inputs

% Type of aggregation
ta = 2;
% Frequency conversion
sc = 4;
% Minimizing the volatility of d-differenced series.
d = 2;
% Additive (1) or proportional (2) variant [optional, default=1]
op1 = 1;
% Multivariate temporal disaggregation
res = denton_multi(Y,x,z,ta,sc,d,op1);
% Printed output
file_out = 'denton_multi.out';
tdprint(res,file_out);
edit denton_multi.out;
% Graphs
tdplot(res);
figure
vnames = {'NORTH NORTHWEST', 'SOUTH SOUTHEAST', 'CENTER', 'ISLANDS'};
for j=1:size(x,2)
    subplot(2,2,j)
    plot([x(:,j) res.y(:,j) copylow(Y(:,j),1,sc)]);
    title(vnames(j));
    legend('HF Tracker','Estimate','LF Benchmark','Location','best');
end
```

The next box presents the printed and graphical output of the multivariate Denton:

Box 4.3: Multiple temporal disaggregation with a transversal constraint: multivariate Denton output

```
*******************************************************
TEMPORAL DISAGGREGATION METHOD: Multivariate Denton
 Type: Additive variant
*******************************************************
-------------------------------------------------------
Number of low-frequency observations  :   16
Frequency conversion                  :    4
Number of high-frequency observations :   64
Number of extrapolations           :    0
-------------------------------------------------------
Type of disaggregation: average (index).
-------------------------------------------------------
Degree of differencing                :    2
-------------------------------------------------------
Elapsed time:  21.1590
```





4.2. Two-step methods: Rossi

Rossi (1982) proposes a two-step approach that combines a first-step (preliminary) estimation by means of a model-based procedure (e.g., Chow-

Lin, Fernández or Litterman) and a second-step that incorporates the transversal constraint while preserving the temporal consistency achieved in the first step. The corresponding function is:

Box 4.4: Multiple temporal disaggregation with a transversal constraint: multivariate Rossi function

```
function res = rossi(Y,x,z,ta,sc,opMethod,type)
% PURPOSE: Multivariate temporal disaggregation with transversal constraint
% ---------------------------------------------------------------------------
% SYNTAX: res = rossi(Y,x,z,ta,sc,opMethod,type);
% ---------------------------------------------------------------------------
% OUTPUT: res: a structure
%         res.meth  = 'Multivariate Rossi';
%         res.N     = Number of low frequency data
%         res.n     = Number of high frequency data
%         res.pred  = Number of extrapolations (=0 in this case)
%         res.ta    = Type of disaggregation
%         res.sc     = Frequency conversion
%         res.y     = High frequency estimate
%         res.z     = High frequency constraint
%         res.et    = Elapsed time
% ---------------------------------------------------------------------
% INPUT: Y:     NxM ---> M series of low frequency data with N observations
%        x:     nxM ---> M series of high frequency data with n observations
%        z:     nx1 ---> high frequency transversal constraint
%        ta: type of disaggregation
%            ta=1 ---> sum (flow)
%            ta=2 ---> average (index)
%            ta=3 ---> last element (stock) ---> interpolation
%            ta=4 ---> first element (stock) ---> interpolation
%        sc: number of high frequency data points for each low frequency data points
%            Some examples:
%            sc= 4 ---> annual to quarterly
%            sc=12 ---> annual to monthly
%            sc= 3 ---> quarterly to monthly
%        opMethod: univariate temporal disaggregation procedure used to compute
%        preliminary estimates
%            opMethod = 1 -> Fernandez
%            opMethod = 2 -> Chow-Lin (optimized for rl=[], see chowlin)
%            opMethod = 3 -> Litterman (optimized for rl=[], see litterman)
%            Intercept is pretested: opC = -1
%        type: estimation method:
%            type=0 ---> weighted least squares
%            type=1 ---> maximum likelihood
% ---------------------------------------------------------------------
% LIBRARY: aggreg, vec, desvec, fernandez, chowlin, litterman
% ---------------------------------------------------------------------
% SEE ALSO: denton, difonzo, mtd_print, mtd_plot
% ---------------------------------------------------------------------
% REFERENCE: Rossi, N. (1982)"A note on the estimation of disaggregate
% time series when the aggregate is known", Review of Economics and Statistics,
% vol. 64, n. 4, p. 695-696.
% Di Fonzo, T. (1994) "Temporal disaggregation of a system of
% time series when the aggregate is known: optimal vs. adjustment methods",
% INSEE-Eurostat Workshop on Quarterly National Accounts, Paris, December.
```

The next box shows the script:

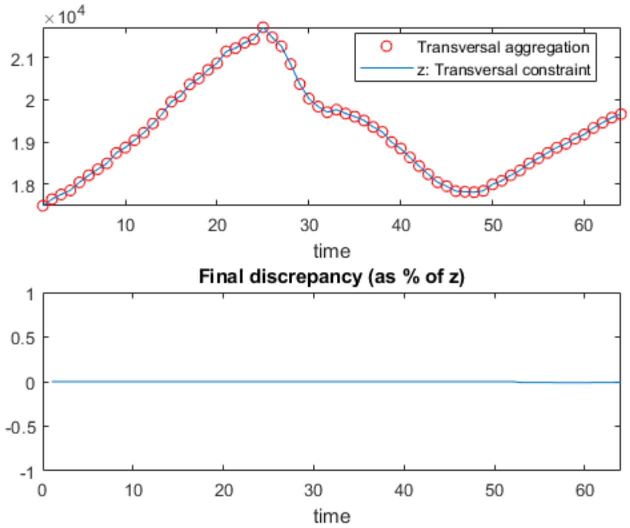
Box 4.5: Multiple temporal disaggregation with a transversal constraint: multivariate Rossi script

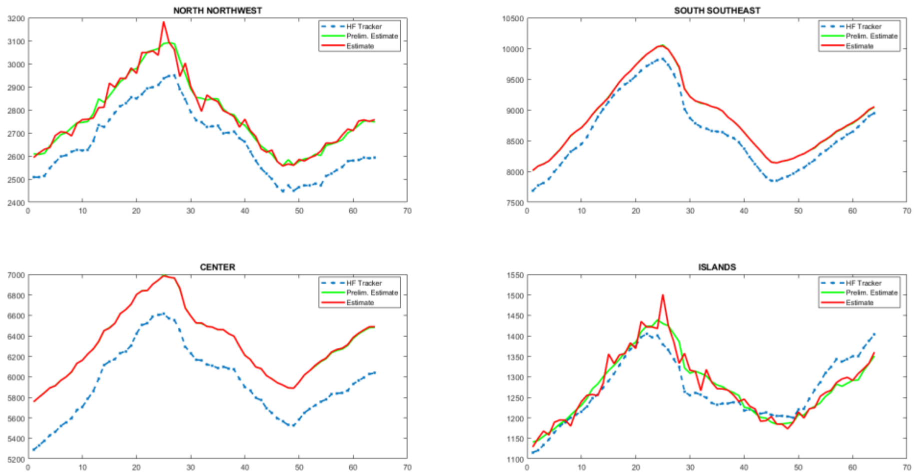
```
% PURPOSE: Demo of rossi()
%          Temporal disaggregation with indicators.
%          Multivariate model with transversal constraint
%             Rossi method
%---------------------------------------------------
% USAGE: rossi_d
%---------------------------------------------------

close all; clear all; clc;

% Loading data
load('Spain_employment_regional_4.mat')
x(end-2:end,:) = [];     %No free extrapolations
z(end-2:end) = [];       %No free extrapolations

% ----------------------------------------------
% Inputs

% Type of aggregation
ta = 2;
% Frequency conversion
sc = 4;
% Type of univariate disaggregation procedure
opMethod = 2;
% Type of univariate disaggregation procedure: estimation method
type = 1;
% Multivariate temporal disaggregation
res = rossi(Y,x,z,ta,sc,opMethod,type);
% Printed output
file_out = 'rossi.out';
tdprint(res,file_out);
edit rossi.out;
% Graphs
tdplot(res);
figure
vnames = {'NORTH NORTHWEST', 'SOUTH SOUTHEAST', 'CENTER', 'ISLANDS'};
for j=1:size(x,2)
    subplot(2,2,j)
    plot([x(:,j) res.y_prelim(:,j) res.y(:,j)]);
    title(vnames(j));
    legend('HF Tracker','Prelim. Estimate','Estimate','Location','best');
end
```

The printed and graphical output of the Rossi procedure is depicted in the next box:

Box 4.6: Multiple temporal disaggregation with a transversal constraint: multivariate Rossi output

```
*******************************************************
 TEMPORAL DISAGGREGATION METHOD: Multivariate Rossi
*******************************************************
-------------------------------------------------------
 Number of low-frequency observations  :   16
 Frequency conversion                  :    4
 Number of high-frequency observations :   64
 Number of extrapolations           :    0
-------------------------------------------------------
 Type of disaggregation: average (index).
-------------------------------------------------------
 Preliminary univariate disaggregation:  Chow-Lin
-------------------------------------------------------
Elapsed time:   0.1250
```

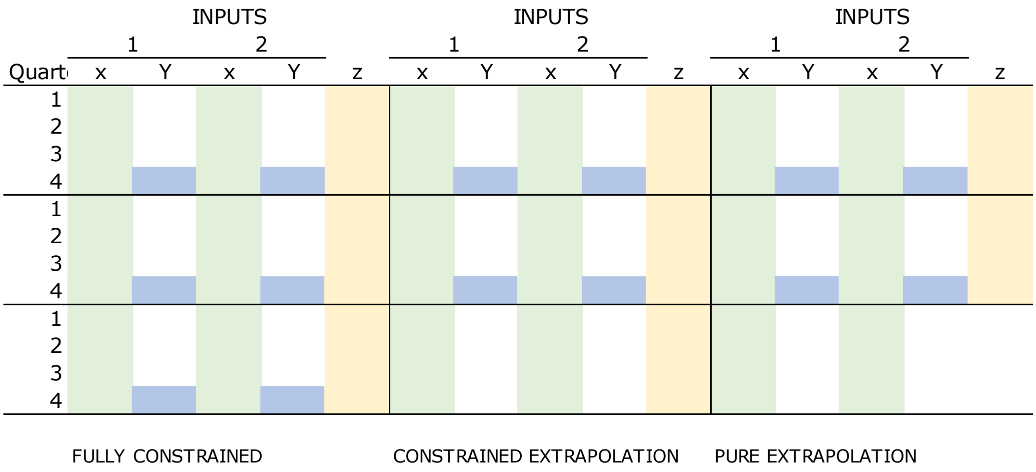




4.3. Model-based methods: Di Fonzo

Di Fonzo (1990) proposes a model-based method to perform multivariate temporal disaggregation with a transversal constraint. The method extends the framework provided by Chow-Lin to the multivariate case and assumes that the innovations are driven by a vector white noise or by a vector random walk. This procedure can handle doubly constrained estimation, transversally constrained estimation and free estimation. The next table illustrates the different cases:

Table 4.1: Di Fonzo method: alternative cases



In addition to this flexibility, the method provides standard errors of the estimates that are useful to gauge its uncertainty. The corresponding function is:

Box 4.7: Multiple temporal disaggregation with a transversal constraint: multivariate Di Fonzo function

```
function res = difonzo(Y,x,z,ta,sc,type,f)
% PURPOSE: Multivariate temporal disaggregation with transversal constraint
% ------------------------------------------------------------------------------------
% SYNTAX: res = difonzo(Y,x,z,ta,sc,type,f);
% ------------------------------------------------------------------------------------
% OUTPUT: res: a structure
%         res.meth  = 'Multivariate Di Fonzo';
%         res.meth1 = Model for shocks: white noise or random walk
%         res.N     = Number of low frequency data
%         res.n     = Number of high frequency data
%         res.pred  = Number of extrapolations
%         res.ta    = Type of disaggregation
%         res.sc     = Frequency conversion
%         res.type  = Model for high frequency innovations
%         res.beta  = Model parameters
%         res.y     = High frequency estimate
%         res.d_y   = High frequency estimate: std. deviation
%         res.z     = High frequency constraint
%         res.et    = Elapsed time
% ------------------------------------------------------------------------------------
% INPUT: Y: NxM  ---> M series of low frequency data with N observations
%        x: nxm  ---> m series of high frequency data with n observations, m>=M see (*)
%        z: nzx1 ---> high frequency transversal constraint with nz obs.
%        ta: type of disaggregation
%            ta=1 ---> sum (flow)
%            ta=2 ---> average (index)
%            ta=3 ---> last element (stock) ---> interpolation
%            ta=4 ---> first element (stock) ---> interpolation
%        sc: number of high frequency data points for each low frequency data points
%            Some examples:
%            sc= 4 ---> annual to quarterly
%            sc=12 ---> annual to monthly
%            sc= 3 ---> quarterly to monthly
%        type: model for the high frequency innvations
%            type=0 ---> multivariate white noise
%            type=1 ---> multivariate random walk
% (*) Optional:
%        f: 1xM ---> Set the number of high frequency indicators linked to
%                    each low frequency variable. If f is explicitly included,
%                    the high frequency indicators should be placed in
%                    consecutive columns
```

```
% ---------------------------------------------------------------------------------
% NOTE: Extrapolation is automatically performed when n>sN.
%       If n=nz>sN restricted extrapolation is applied.
%       Finally, if n>nz>sN extrapolation is perfomed in constrained
%       form in the first nz-sN observatons and in free form in
%       the last n-nz observations.
% ---------------------------------------------------------------------------------
% LIBRARY: aggreg, dif, vec, desvec
% ---------------------------------------------------------------------------------
% SEE ALSO: denton_multi, rossi
% ---------------------------------------------------------------------------------
% REFERENCE: Di Fonzo, T.(1990)"The estimation of M disaggregate time
% series when contemporaneous and temporal aggregates are known", Review
% of Economics and Statistics, vol. 72, n. 1, p. 178-182.
```

The script to run the Di Fonzo procedure is depicted in the next box:

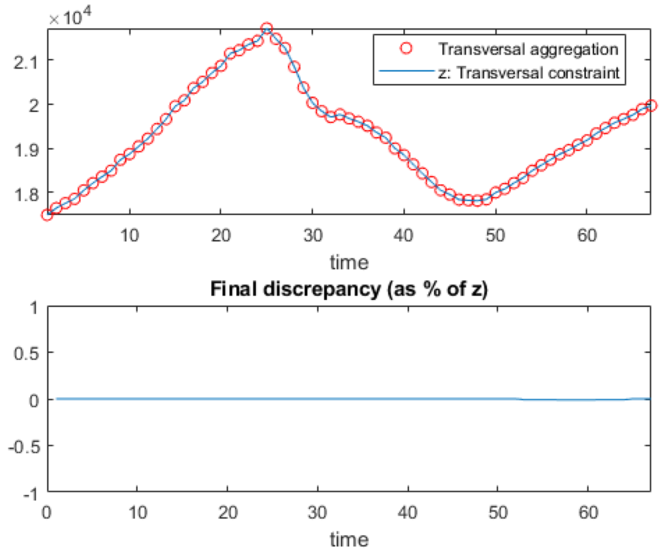
Box 4.8: Multiple temporal disaggregation with a transversal constraint: multivariate Di Fonzo script

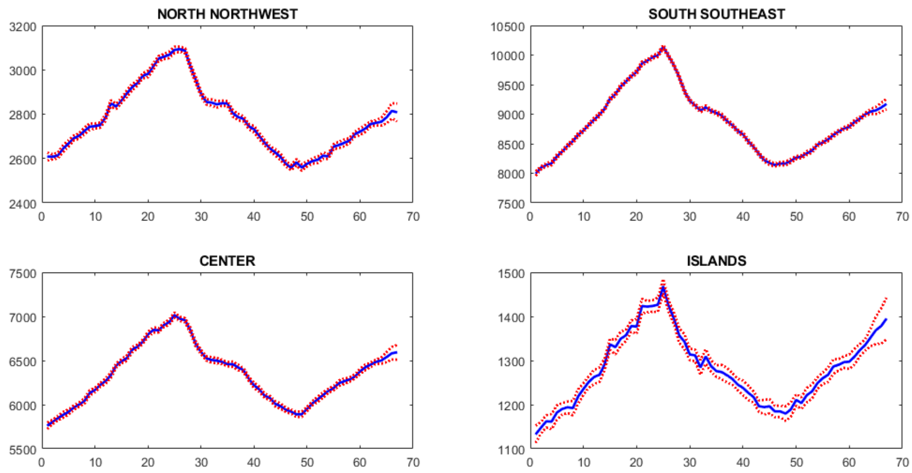
```
% PURPOSE: Demo of difonzo()
%          Temporal disaggregation with indicators.
%          Multivariate model with transversal constraint
%             di Fonzo method
%---------------------------------------------------
% USAGE: dinfonzo_d
%---------------------------------------------------

close all; clear all; clc;

% Loading data
load('Spain_employment_regional_4.mat')

% ---------------------------------------------
% Inputs

% Type of aggregation
ta = 2;
% Frequency conversion
sc = 4;
% Model for the innovations: white noise (0), random walk (1)
type = 1;
% Number of high frequency indicators linked to each low frequency
% aggregate
f = ones(1,size(x,2));
% Multivariate temporal disaggregation
res = difonzo(Y,x,z,ta,sc,type,f);
% Printed output
file_out = 'difonzo.out';
tdprint(res,file_out);
edit difonzo.out;
% Graphs
tdplot(res);

figure
vnames = {'NORTH NORTHWEST', 'SOUTH SOUTHEAST', 'CENTER', 'ISLANDS'};
for j=1:size(x,2)
    subplot(2,2,j)
    plot([res.y(:,j)],'-b');
    hold on
    plot([res.y(:,j)-res.d_y(:,j) res.y(:,j)+res.d_y(:,j)],':r');
    title(vnames(j));
end
```

The next box shows the printed and graphical output:

Box 4.9: Multiple temporal disaggregation with a transversal constraint: multivariate Di Fonzo output

```
*******************************************************
TEMPORAL DISAGGREGATION METHOD: Multivariate Di Fonzo
*******************************************************
-------------------------------------------------------
Number of low-frequency observations  :   16
Frequency conversion                  :    4
Number of high-frequency observations :   67
Number of extrapolations           :    3
-------------------------------------------------------
Type of disaggregation: average (index).
-------------------------------------------------------
Model for the innovations: Random Walk.
-------------------------------------------------------
Elapsed time:   0.0310
```





5. TRANSVERSAL BALANCING METHODS

Up to a certain point, balancing can be considered as a special case of multivariate temporal disaggregation with a transversal constraint, by means of removing from the latter the temporal consistency requirement.

Dropping this requirement, we can consider a simple method (proportional balancing) or adding a layer of complexity if we consider several transversal constraints (as in the case of the RAS method) or the incorporation of uncertainty in the estimation process (as in the van der Ploeg method). A detailed analysis of these issues can be found in Chen (2012), Bikker et al. (2010) and Chen et al. (2018b), among others.

The list of available procedures is:

- Proportional balancing: `bal()`
- Bi-proportional balancing, RAS method: `ras()`
- Optimization balancing, van der Ploeg: `vanderploeg()`

5.1. Proportional balancing

Proportional balancing can be applied by means of the `bal()` function:

Box 5.1: Transversal balancing function

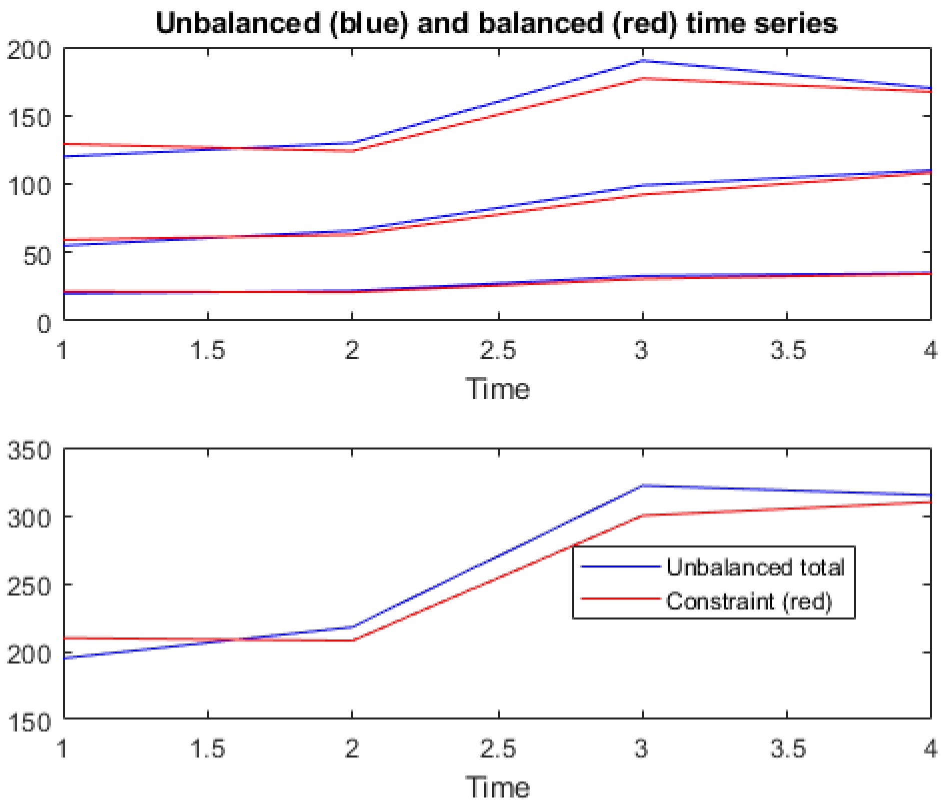
```
function yb = bal(y,z)
% PURPOSE: Proportional adjustment of y to a given total z
% ------------------------------------------------------------------------
% SYNTAX: yb = bal(y,z);
% ------------------------------------------------------------------------
% OUTPUT: yb : nxM  --> balanced time series
% ---------------------------------------------------------------------
% INPUT: y: nxM  --> unbalanced time series
%        z: nx1 --> transversal constraint
% ---------------------------------------------------------------------
% LIBRARY: vec, desvec
% ---------------------------------------------------------------------
% SEE ALSO: ras, vanderploeg
% ---------------------------------------------------------------------
% REFERENCE: di Fonzo, T. (1994) "Temporal disaggregation of a system of
% time series when the aggregate is known: optimal vs. adjustment methods",
% INSEE-Eurostat Workshop on Quarterly National Accounts, Paris, December.
```

The next script details how to implement the `bal()` function:

Box 5.2: Transversal balancing script

```
% PURPOSE: Demo of bal()
%          Proportional transversal balancing
%---------------------------------------------------
% USAGE: bal_d
%---------------------------------------------------

clc; clear all; close all;

% Unbalanced time series (read columnwise)
```

```
y = [120   20      55
     130   22      66
     190   33      99
     170   35      110 ];

% Transversal constraints
Z =[ 210
      208
      300
      310 ];

% Balancing
yb = bal(y,z);

% Graphs
t = (1:size(y,1));
subplot(2,1,1)
plot(t, y, '-b', t, yb, '-r')
    xlabel('Time')
    title('Unbalanced (blue) and balanced (red) time series')
subplot(2,1,2)
plot(t, sum(y')', '-b', t, z, '-r')
    xlabel('Time')
    legend('Unbalanced total', 'Constraint (red)','Location','best')
```



5.2. Bi-proportional (RAS) balancing

The so-called RAS or bi-proportional method provides a bidimensional extension of the proportional method, allowing its application to more complex data structures like Input-Outout (IO) tables, see Bacharach (1965). The `ras()` function is:

Box 5.3: Bi-proportional balancing (RAS) function

```
function F1 = ras(F0,x0,x1,v,u,opG)
% PURPOSE: Bi-proportional adjustment
% ------------------------------------------------------------
% SYNTAX: F1 = ras(F0,x0,x1,v,u,opG);
% ------------------------------------------------------------
% OUTPUT: F1: kxk -> updated (balanced) matrix
% ------------------------------------------------------------
% INPUT: F0: kxk -> benchmark matrix
%        x0: 1xk -> benchmark output (by cols)
%        x1: 1xk -> updated output (by cols)
%        v: 1xk -> updated F totals (by cols)
%        u: kx1 -> updated F totals (by rows)
%        opG: 1x1 -> convergence plot (optional, default=0)
% ------------------------------------------------------------
% LIBRARY:
% ------------------------------------------------------------
% SEE ALSO: bal, vdp
% ------------------------------------------------------------
% REFERENCE: Bacharach, M. (1965) "Estimating non-negative matrices from
% marginal data", International Economic Review, vol. 6, n. 3, p. 294-310.
```

The next box presents the script that implements the RAS procedure:

Box 5.4: Bi-proportional balancing (RAS) script

```
% PURPOSE: Demo of ras()
%          Bi-proportional transversal balancing
%---------------------------------------------------
% USAGE: ras_d
%---------------------------------------------------

clear all; close all; clc;

%-----------------------------------------------------------------------
% BENCHMARK
%-----------------------------------------------------------------------

% Unbalanced IO matrix
F0= [50 100 0
     30 50  20
     20 50  30  ];

% Total by column
x0 = [200 300 200];

%-----------------------------------------------------------------------
% UPDATE
%-----------------------------------------------------------------------

% Total output by row
u = [160 ; 150 ; 120];

% Total intermediante output by column
v = [100 250 80];

% Total output
x1 = [200 400 300];

% Graphics
opG = 1;

% RAS balancing
F1 = ras(F0, x0, x1, v, u, opG);
```

The inputs of the function are summarized in the next table:

Table 5.1: Bi-proportional balancing (RAS): inputs

BENCHMARK (T=0)

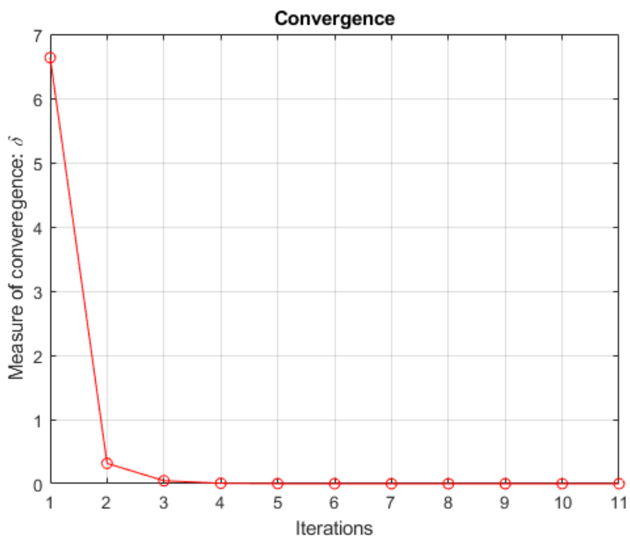
Product	Product A	Product B	Product C	Total	Discrepancy	Final Demand	Total Output
A	50	100	0	**150**	0	50	**200**
B	30	50	20	**100**	0	200	**300**
C	20	50	30	**100**	0	100	**200**
Total	**100**	**200**	**50**		**0**	350	700
Discrepancy	0	0	0				
Value Added	100	100	150		**F0**		
Total Output	**200**	**300**	**200**	700	**x0**		

UPDATE (T=1)

Product	Product A	Product B	Product C	Total	Discrepancy	Final Demand	Total Output
A				**160**		40	**200**
B				**150**		250	**400**
C				**120**		180	**300**
Total	**100**	**250**	**80**			470	900
Discrepancy							
Value Added	100	150	220		**u**		
Total Output	**200**	**400**	**300**	900	**v**		

The function provides information about the speed of convergence of the algorithm:

Figure 5.1: Bi-proportional balancing (RAS): convergence of the algorithm



Inputs and outputs of the example used to illustrate the RAS procedure are presented in the next box:

Table 5.2: Bi-proportional balancing (RAS): output

BALANCED UPDATE (T=1)

Product	A	B	C	Total	Discrepancy	Final Demand	Total Output
		Product					
A	45.25	114.75	0.00	**160**	0	40	**200**
B	36.23	76.56	37.21	**150**	0	250	**400**
C	18.52	58.69	42.79	**120**	0	180	**300**
Total	**100**	**250**	**80**		**0**	470	900
Discrepancy	0	0	0				
Value Added	100	150	220		F1		
Total Output	**200**	**400**	**300**	900			

5.3. Quadratic optimization methods: Van der Ploeg

The method proposed by Van der Ploeg (1985, 1986) combines a solid balancing procedure, based on a quadratic optimization problem, with the possibility of including a priori information about the reliability of the estimates. This combination yields a flexible procedure that can handle several constraints as well as different scenarios regarding the uncertainty of the inputs. The corresponding function is:

Box 5.5: Van der Ploeg function

```
function res = vanderploeg(y,S,A,a)
% PURPOSE: Balancing by means of QL optimization (LS estimation)
% ------------------------------------------------------------------------
% SYNTAX: res = vanderploeg(y,S,A,a);
% ------------------------------------------------------------------------
% OUTPUT: res: a structure with ...
%           z       : kx1 vector of balanced variables
%           Sz      : kxk VCV of final (balanced) estimates
%           lambda  : mx1 Lagrange multipliers
% ------------------------------------------------------------------------
% INPUT:   y      : kx1 vector of unbalanced variables (initial estimates)
%          S      : kxk VCV of initial estimates
%          A      : kxm matrix of linear constraints
%          a      : 1xm vector of autonomous terms related to linear constraints
% Note: a is optional. If it is not explicitly included, the function
% assumes a=0.
% ------------------------------------------------------------------------
% LIBRARY:
% ------------------------------------------------------------------------
% SEE ALSO: bal, ras
% ------------------------------------------------------------
% REFERENCE: Van der Ploeg, F.(1982)"Reliability and the adjustment
% of sequences of large economic accounting matrices", Journal of
% the Royal Statistical Society, series A, vol. 145, n. 2, p. 169-194.
```

The next script implements the Van der Ploeg method in a case in which eight estimates must satisfy two transversal constraints. The inputs include an initial (unbalanced) estimate of the variables (y) as well as a measure of it's a priori uncertainty (matrix C).

Box 5.6: Van der Ploeg script

```
% PURPOSE: Demo of vanderploeg()
%          Estimation subject to transversal constraints
%          by means of a quadratic optimization criterion
%-----------------------------------------------------------------
% USAGE: vanderploeg_d
```

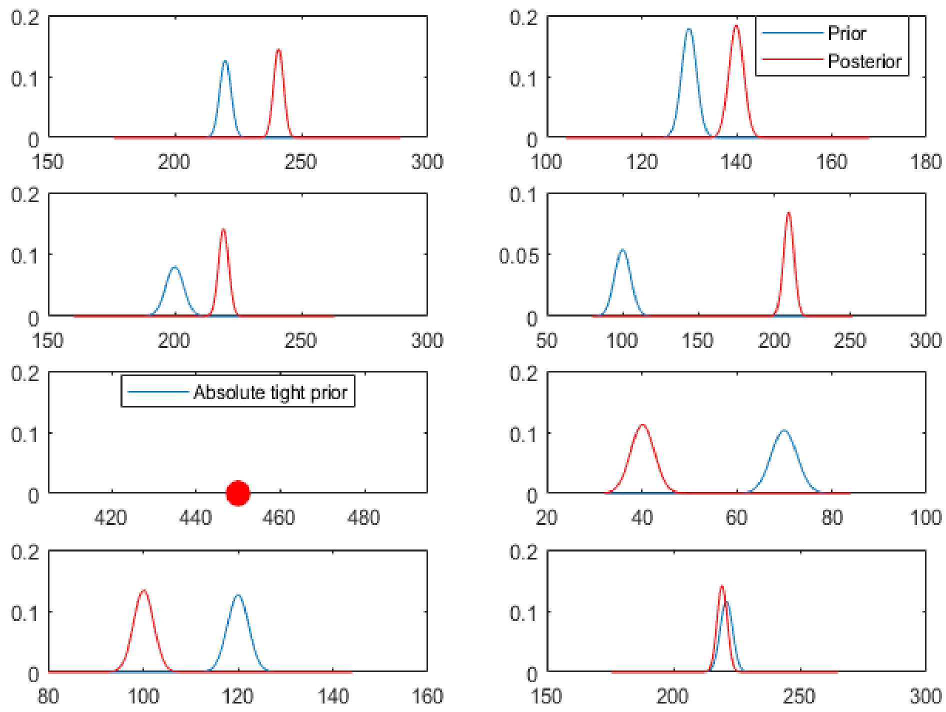
```
%-----------------------------------------------------------------

close all; clear all; clc;

%-----------------------------------------------------------------
% Unbalanced cross-section vector
y = [  220.00
       130.00
        200.00
        100.00
        450.00
         70.00
        120.00
        221.00 ];

[k, n]=size(y);

%-----------------------------------------------------------------
% Linear constraints
A =[     1.00              0
    1.00             0
    1.00          1.00
    1.00             0
   -1.00             0
   -1.00             0
   -1.00             0
   -1.00         -1.00   ];

[k, m] = size(A);

%-----------------------------------------------------------------
% VCV matrix of estimates

% Vector of variances
% Note: Fixed estimation: s(5)=0 --> z(5)=y(5)
s = [10 5 25 55 0 15 10 12];
Aux1 = (diag(sqrt(s)));

% Correlation matrix: C
C = zeros(k);
C(1,3) = 0.5;
Aux2 = tril(C');
C = C + Aux2 + diag(ones(1,k));

% VCV matrix: S
S = Aux1 * C * Aux1;

% van der Ploeg balancing
res = vanderploeg(y,S,A);

% Check
format bank
disp (''); disp ('*** INITIAL AND FINAL DISCREPANCIES ***'); disp('');
[ A' * y  A' * res.z]

% Revision (as %)
p = 100 * ((res.z - y) ./ y);

% Final results:
disp ('');
disp ('*** INITIAL ESTIMATE, FINAL ESTIMATE, REVISION AS %, INITIAL VARIANCES, FINAL
VARIANCES ***');
disp ('');
[y res.z p diag(S) diag(res.Sz)]
format short

% Graphs
sv = (diag(res.Sz));
s  = diag(S);
for j=1:k
    if (s(j) == 0)
        x = linspace(min(y(j),res.z(j))*0.9,max(y(j),res.z(j))*1.1,1000);
        subplot(4,2,j)
        plot(x,0)
        hold on
        plot(y(j),0,'or','LineWidth',6)
```

```
        axis([min(y(j),res.z(j))*0.9 max(y(j),res.z(j))*1.1 0 0.2])
        legend('Absolute tight prior','Location','best')
    else
        x = linspace(min(y(j),res.z(j))*0.8,max(y(j),res.z(j))*1.2,1000);
        y0 = (1/ (sqrt(2*pi*s(j)))) * exp(-((x-y(j)).^2 ./ s(j)));
        y1 = (1/ (sqrt(2*pi*sv(j)))) * exp(-((x-res.z(j)).^2 ./ sv(j)));
        subplot(4,2,j)
        plot(x, y0, x, y1,'-r')
        legend('Prior','Posterior','Location','best')
    end
end
```

The output of the is depicted in the next box, including a comparison of the prior distribution with the posterior distribution under the assumption of Gaussianity:

Box 5.7: Van der Ploeg output

```
*** INITIAL AND FINAL DISCREPANCIES ***

       -211.00          0.00
        -21.00             0

***     INITIAL         FINAL         REVISION      INITIAL VAR      FINAL VAR ***

        220.00         241.01          9.55          10.00           7.57
        130.00         139.94          7.65           5.00           4.73
        200.00         219.29          9.64          25.00           8.04
        100.00         209.35        109.35          55.00          22.58
        450.00         450.00             0              0              0
         70.00          40.18        -42.60          15.00          12.59
        120.00         100.12        -16.57          10.00           8.93
        221.00         219.29         -0.78          12.00           8.04
```



6. UTILITIES

This library contains a set of auxiliary functions that are used by many of the main functions but can also be used in an autonomous way. The following list presents the most important.

- Recursive estimates and revisions for some procedures: `backtest()`
- Systematic sampling: `ssampler()`
- Temporal aggregation: `temporal_agg()`
- Temporal aggregation preserving input dimension: `temporal_agg_p()`
- Temporal accumulation: `temporal_acc()`
- Moving sum (average): `moving_acc()`
- Cast low-frequency data into high-frequency format: `copylow()`

Finally, the library included a set of functions to generate printed and graphical output, (`tdprint()` and `tdplot()`, respectively) as well as some functions to identify, estimate and forecast univariate autoregressive (AR) models: `ar_order()`, `arx()` and `upredict()`.

REFERENCES

Abad, A. and E.M. Quilis (2005) "Software to Perform Temporal Disaggregation", Eurostat, Working Paper and Studies, ISSN 1725-4825.

Bacharach, M. (1965) "Estimating Non-Negative Matrices from Marginal Data", *International Economic Review*, 6(3): 294-310.

Barbone, L., G. Bodo and I. Visco (1981) "Costi e profitti nell'industria in senso stretto: Un'analisi su serie trimestrali, 1970–1980", *Bolletino della Banca d'Italia*, p. 467–510.

Bikker, R., J. Daalmans and N. Mushkudiani (2010) "A Multivariate Denton Method for Benchmarking Large Data Sets", Statistics Netherlands, Discussion Paper n. 10002.

Bisio, L. and Moauro, F. (2018) "Temporal Disaggregation by Dynamic Regressions: Recent Developments in Italian Quarterly National Accounts", *Statistica Neerlandica*, 72, 471-494.

Boot, J.C.G., W. Feibes and J.H.C. Lisman (1967) "Further Methods of Derivation of Quarterly Figures from Annual Data", *Applied Statistics*, 16, 65-75.

Bournay, J. and G. Laroque (1979) "Réflexions sur la méthode d'elaboration des comptes trimestriels", *Annales de l'INSEE*, 36, 3-30.

Chen, B. (2012) "A Balanced System of U.S. Industry Accounts and Distribution of the Aggregate Statistical Discrepancy by Industry", *Journal of Business and Economic Statistics*, 30, 202-211.

Chen, B., Di Fonzo, T. and Mushkudiani, N. (2018a), "Benchmarking, Temporal Disaggregation, and Reconciliation of Systems of Time Series", *Statistica Neerlandica*, 72, 402-405.

Chen, B., Di Fonzo, T. and Mushkudiani, N. (2018b), "The Statistical Reconciliation of Time Series of Accounts Between Two Benchmark Revisions", *Statistica Neerlandica*, 72, 533-552.

Cholette, P. A. (1984) "Adjusting Sub-Annual Series to Yearly Benchmarks," *Survey Methodology*, 10, 35–49.

Chow, G. and A.L. Lin (1971) "Best Linear Unbiased Distribution and Extrapolation of Economic Time Series by Related Series", *Review of Economic and Statistics*, 53, 372-375.

Cuevas, A., E.M. Quilis and A. Espasa (2015) "Quarterly Regional GDP Flash Estimates by Means of Benchmarking and Chain Linking", *Journal of Official Statistics*, 31, 627–647.

Dagum, E.B. and P.A. Cholette (2006) *Benchmarking, Temporal Distribution, and Reconciliation Methods for Time Series*, Springer.

Daalmans, J. (2018) “On the Sequential Benchmarking of Subannual Series to Annual Totals”, *Statistica Neerlandica*, 72, 406-420.

Denton, F.T. (1971) “Adjustment of Monthly or Quarterly Series to Annual Totals: An Approach Based on Quadratic Minimization”, *Journal of the American Statistical Society*, 66, 99-102.

Di Fonzo, T. (1987) *La stima indiretta di serie economiche trimestrali*, Cleup Editore.

Di Fonzo, T. (1990) “The Estimation of m Disaggregate Time Series when Contemporaneous and Temporal Aggregates are Known”, *Review of Economics and statistics*, 72, 178-182.

Di Fonzo, T. (1994) “Temporal Disaggregation of a System of Time Series when the Aggregate Is Known: Optimal vs. Adjustment Methods”, INSEE-Eurostat Workshop on National Accounts, Paris.

Di Fonzo, T. (2002) “Temporal Disaggregation of Economic Time Series: Towards a Dynamic Extension”, Dipartimento di Scienze Statistiche, Università di Padova, Working Paper n. 2002-17.

Di Fonzo, T. and M. Marini (2003) “Benchmarking Systems of Seasonally Adjusted Time Series According to Denton’s Movement Preservation Principle”, Dipartimento di Scienze Statistiche, Università di Padova, Working Paper no. 2003–09.

Di Fonzo, T. and M. Marini (2011) “Simultaneous and two-step reconciliation of systems of time series: methodological and practical issues”, *Journal of the Royal Statistical Society*, Series C, 60, 143–164.

Di Fonzo, T. and M. Marini (2012) “On the Extrapolation with the Denton Proportional Benchmarking Method”, International Monetary Fund, WP 12/169.

Fernández, R.B. (1981) “Methodological Note on the Estimation of Time Series”, *Review of Economic and Statistics*, 63, 471-478.

Guerrero, V. (1990) “Temporal Disaggregation of Time Series: an ARIMA-Based Approach”, *International Statistical Review*, 58, 29-46.

Guerrero, V. and F.H. Nieto (1999) “Temporal and Contemporaneous Disaggregation of Multiple Time Series”, *Test*, 8, 459-489.

Guerrero, V.M. and Corona, F. (2018) “Retropolating Some Relevant Series of Mexico's System of National Accounts at Constant Prices: The Case of Mexico City's GDP”, *Statistica Neerlandica*, 72, 495-519.

Litterman, R.B. (1983) “A Random Walk, Markov Model for the Distribution of Time Series”, *Journal of Business and Economic Statistics*, 1, 169-173.

Proietti, T. (2006) "Temporal Disaggregation by State Space Methods: Dynamic Regression Methods Revisited", *Econometrics Journal*, 9, 357–372.

Quilis, E.M. (2018) "Temporal Disaggregation of Economic Time Series: The View from the Trenches", *Statistica Neerlandica*, 72, 447-470.

Rossi, N. (1982) "A Note on the Estimation of Disaggregate Time Series when the Aggregate is Known", *Review of Economics and Statistics*, 64, 695-696.

Santos, J.M.C. and F. Cardoso (2001) "The Chow-Lin Method Using Dynamic Models", *Economic Modelling*, 18, 269-280.

Sims, C.A. (1974) "Seasonality in Regression", *Journal of the American Statistical Association*, 69: 618-626.

Stram, D.O. and W.W.S. Wei (1986) "A methodological note on the disaggregation of time series totals", *Journal of Time Series Analysis*, 7, 293-302.

Temursho, U. (2018) "Entropy-Based Benchmarking Methods", *Statistica Neerlandica*, 72, 421-446.

Van der Ploeg, F. (1982) "Reliability and the Adjustment of Large Economic Accounting Matrices", *Journal of the Royal Statistical Society*, series A, 145, 169-194.

Van der Ploeg, F. (1985) "Econometrics and Inconsistencies in the National Accounts", *Economic Modelling*, 2, 8-16.

Wei, W.W.S. (1990) *Time Series Analysis*, Addison Wesley.