A MATLAB LIBRARY OF TEMPORAL DISAGGREGATION AND INTERPOLATION METHODS

Enrique M. Quilis[1]

Macroeconomic Research Department
Ministry of Economy and Competitiveness
Paseo de la Castellana, 162
28046 – Madrid (SPAIN)
enrique.quilis@mineco.es

January, 2013

[1] I am indebted to Ana Abad, Juan Bógalo, Fátima Cardoso, Tommaso Di Fonzo, Leandro Navarro and Silvia Relloso for their help. I also thank J.osé Ramón Cancelo, Ángel Cuevas, Antoni Espasa, Rafael Frutos, Miguel Jerez, Agustín Maravall, Francisco Melis, Tommaso Proietti, José Miguel Ramos and Eduardo Salazar for stimulating discussions on temporal disaggregation and benchmarking.

INTRODUCTION

This release of the Matlab temporal disaggregation and interpolation library includes some new features[2]:

- Temporal disaggregation using a dynamic model, Autoregressive Dynamic Linear ADL(1,1), analyzed by Proietti (2006). This method encompasses Chow-Lin and Santos Silva-Cardoso.
- An additional utility to perform temporal aggregation: `temporal_agg_p()`
- Balancing according to the bi-proportional (RAS) method: `ras()`
- Improved grid search for Chow-Lin, Litterman and Santos Silva-Cardoso methods.
- A new function that encompasses the additive and proportional variants of the univariate Denton method. The new function also includes the Cholette (1984) treatment of initial conditions.
- A revised function that encompasses the additive and proportional variants of the multivariate Denton method.

The library includes a set of functions to perform temporal disaggregation, interpolation, extrapolation, and balancing, according to the following structure:

Univariate bechmarking without indicators:

- Boot-Feibes-Lisman: `bfl()`.
- Stram-Wei (an ARIMA model-based method): `sw()`.

  ➔ Served by `tduni_print()` and `tduni_plot()`.
  ➔ … or by `tdprint()` and `tdplot()`.

- Low-pass interpolation: `low_pass_interpolation()`.

Univariate bechmarking with indicators (QL optimization or model-based, BLUE methods):

- Denton: additive and proportional variants: `denton_uni()`.
- Chow-Lin, maximum likelihood and weighted least squares: `chowlin()`.
- Chow-Lin, Cochrane-Orcutt: `chowlin_co()`.
- Fernández: `fernandez()`.
- Litterman: `litterman()`.
- Santos Silva-Cardoso (dynamic extension of Chow-Lin): `ssc()`.
- Proietti's Autoregressive Dynamic Linear ADL(1,1) model: `proietti()`.

  ➔ Served by `td_print()` and `td_plot()`.
  ➔ … or by `tdprint()` and `tdplot()`.

- Guerrero (an ARIMA model-based method).

  ➔ Served by `td_print_G()` and `td_plot_G()`.

[2] New features and functions are highlighted using **red bold fonts**.

Multivariate bechmarking with indicators and transversal constraint:

- Multivariate Denton: additive and proportional variants: `denton()`.
- Rossi : `rossi()`.
- Di Fonzo : `difonzo()`.

  ➔ Served by `mtd_print()` and `mtd_plot()`.
  ➔ … or by `tdprint()` and `tdplot()`.

Multivariate balancing:

- Proportional adjustment: `bal()`.
- Van der Ploeg: `vdp()`.
- RAS Bi-proportional method: `ras()`.

Utilities:

- Backtesting temporal disaggregation: `backtest()`.
- Systematic sampler: `ssampler()`.
- Moving sum or average: `moving_acc()`.
- Temporal accumulation: `temporal_acc()`.
- Temporal aggregation preserving input dimension: `temporal_agg_p()`.
- Cast low-frequency data into high-frequency format: `copylow()`.

The presentation of the functions is self-contained: e.g. `help sw` shows the detailed structure of the `sw()` function: purpose, input, output, library and technical references.

A Visual Basic interface has been developed to apply most of the library using Excel spreadsheets. Basically, the interface consists of two main modules: a program which generates and manages a sequence of contextual menus and a linkage function that activates the temporal disaggregation library according to the user's choices, as expressed by means of the corresponding forms. The interface is described in Abad, A. y Quilis, E.M. (2005) "Software to perform temporal disaggregation", Eurostat, Working Paper and Studies, ISSN 1725-4825, p. 2-8.

As in previous releases, the library uses some functions contained in the *Econometric Toolbox* of Professor James LeSage. I strongly recommend the use of this library in combination with his toolbox, in order to achieve important economies of scale in the art and science of quantitative modeling. More information may be obtained in his website:

http://www.spatial-econometrics.com/

UNIVARIATE BENCHMARKING

<u>INFORMATION SET</u>:

LOW-FREQUENCY TIME SERIES (Benchmark)

- **BOOT-FEIBES-LISMAN:** `bfl()`
- **STRAM-WEI:** `sw()`
- **LOW-PASS INTERPOLATION:** `low_pass_interpolation()`

BOOT-FEIBES-LISMAN: Univariate benchmarking and interpolation

```
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
%           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.
```

STRAM-WEI: Univariate benchmarking and interpolation based on ARIMA modeling

```
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
%           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.
```

LOW-PASS INTERPOLATION: Univariate benchmarking and interpolation based on smoothing (via Hodrick-Prescott) and time-domain benchmarking (Denton)

```
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
%            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_uni
% ----------------------------------------------------------------------
% SEE ALSO: bfl, sw
% ----------------------------------------------------------------------
```

UNIVARIATE BENCHMARKING WITH INDICATORS

<u>INFORMATION SET</u>:

LOW-FREQUENCY TIME SERIES (Benchmark)
HIGH-FREQUENCY TIME SERIES (Indicators)

- **DENTON, additive and proportional variants:** `denton_uni()`
- **CHOW-LIN by Maximum Likelihood and Weighted Least Squares:** `chowlin()`
- **FERNANDEZ:** `fernandez()`
- **LITTERMAN by Maximum Likelihood and Weighted Least Squares:** `litterman()`
- **SANTOS SILVA-CARDOSO by Maximum Likelihood and Weighted Least Squares:** `ssc()`
- **PROIETTI ADL(1,1) by Maximum Likelihood and Weighted Least Squares:** `proietti()`
- **GUERRERO:** `guerrero()`

DENTON_UNI: Univariate benchmarking and interpolation

```
function res = denton_uni(Y,x,ta,d,sc,op1,op2)
% PURPOSE: Temporal disaggregation using the Denton method.
% ---------------------------------------------------------------------
% SYNTAX: res = denton_uni(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
%            sc= 4 ---> annual to quarterly
%            sc=12 ---> annual to monthly
%            sc= 3 ---> quarterly to monthly
%        op1: additive (1) or proportional (2) variant
%        op2: standard (1) or Cholette (2) solution
% ---------------------------------------------------------------------
% LIBRARY: aggreg
% ---------------------------------------------------------------------
% 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.
% Cholette, P. (1984) "Adjusting sub-annual series to yearly benchmarks",
% Survey Methodology, vol. 10, p. 35-49.
```

CHOW-LIN: Univariate benchmarking and interpolation based on indicators. High-frequency model: linear model + AR(1) disturbances. Estimation is performed by maximum likelihood or weighted least squares.

```
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.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
%            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], 100 points grid
%            rl: 1x1 ---> fixed value of rho parameter
%            rl: 1x2 ---> [r_min r_max] search is performed
%                on this range, using a 200 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.
```

FERNANDEZ: Univariate benchmarking and interpolation based on indicators. High-frequency model: linear model + I(1) disturbances (random walk). Estimation is performed by generalized least squares.

```
function res = fernandez(Y,x,ta,sc,opC)
% PURPOSE: Temporal disaggregation using the Fernandez method
% ------------------------------------------------------------
% SYNTAX: res = fernandez(Y,x,ta,sc,opC);
% ------------------------------------------------------------
% OUTPUT: res: a structure
%           res.meth    ='Fernandez';
%           res.ta      = type of disaggregation
%           res.type    = method of estimation
%           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.aic     = Information criterion: AIC
%           res.bic     = Information criterion: BIC
% ------------------------------------------------------------
% 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
%            sc= 4 ---> annual to quarterly
%            sc=12 ---> annual to monthly
%            sc= 3 ---> quarterly to monthly
%        opC: 1x1 option related to intercept
%            opc = -1 : pretest intercept significance
%            opc =  0 : no intercept in hf model
%            opc =  1 : intercept in hf model
% ------------------------------------------------------------
% LIBRARY: fernandez_W
% ------------------------------------------------------------
% SEE ALSO: chowlin, litterman, td_plot, td_print
% ------------------------------------------------------------
% REFERENCE: Fernandez, R.B.(1981)"Methodological note on the
% estimation of time series", Review of Economic and Statistics,
% vol. 63, n. 3, p. 471-478.
```

LITTERMAN: Univariate benchmarking and interpolation based on indicators. High-frequency model: linear model + ARI(1,1) disturbances (markovian random walk). Estimation is performed by maximum likelihood or weighted least squares.

```
function res = litterman(Y,x,ta,sc,type,opC,rl)
% PURPOSE: Temporal disaggregation using the Litterman method
% ------------------------------------------------------------
% SYNTAX: res = litterman(Y,x,ta,sc,type,opC,rl)
% ------------------------------------------------------------
% OUTPUT: res: a structure
%           res.meth    ='Litterman';
%           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.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
%            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], 100 points grid
%           rl: 1x1 ---> fixed value of rho parameter
%           rl: 1x2 ---> [r_min r_max] search is performed
%               on this range, using a 200 points grid
% ------------------------------------------------------------
% LIBRARY: litterman_W
% ------------------------------------------------------------
% SEE ALSO: chowlin, fernandez, td_plot, td_print
% ------------------------------------------------------------
% REFERENCE:  Litterman, R.B. (1983a) "A random walk, Markov model
% for the distribution of time series", Journal of Business and
% Economic Statistics, vol. 1, n. 2, p. 169-173.
```

SSC: Univariate benchmarking and interpolation based on indicators. High-frequency model: dynamic linear model. Estimation is performed by maximum likelihood or weighted least squares.

```
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.rho      = 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
%           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], 100 points grid
%           rl: 1x1 ---> fixed value of rho parameter
%           rl: 1x2 ---> [r_min r_max] search is performed
%               on this range, using a 200 points grid
% ---------------------------------------------------------
% LIBRARY: ssc_W
% ---------------------------------------------------------
% SEE ALSO: chowlin, litterman, fernandez, td_plot, td_print
% ---------------------------------------------------------
% REFERENCE: Santos, 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.
```

PROIETTI: Univariate benchmarking and interpolation based on indicators. High-frequency model: ADL(1,1). Estimation is performed by maximum likelihood or weighted least squares.

```
function res = proietti(Y,x,ta,sc,type,opC,rl)
% PURPOSE: Temporal disaggregation using the ADL(1,1) model
%          analyzed by Proietti (2006).
% ------------------------------------------------------------
% SYNTAX: res = proietti(Y,x,ta,sc,type,opC,rl);
% ------------------------------------------------------------
% OUTPUT: res: a structure
%           res.meth    ='Proietti';
%           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) and x(0))
%           res.gamma_sd = estimated model parameters: standard deviation
%           res.gamma_t  = estimated model parameters: t ratios
%           res.rho      = dynamic parameter phi
%           res.beta     = estimated model parameters (excluding y(0) and x(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: nx1 ---> matrix of high frequency indicator (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
%           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], 100 points grid
%           rl: 1x1 ---> fixed value of rho parameter
%           rl: 1x2 ---> [r_min r_max] search is performed
%               on this range, using a 200 points grid
% ---------------------------------------------------------
% LIBRARY: ar_order, arx, upredict, ssc
% ---------------------------------------------------------
% SEE ALSO: ssc, chowlin, litterman, fernandez, td_plot, td_print
% ---------------------------------------------------------
% REFERENCE: Proietti, T. (2006) "Temporal disaggregation by state space
% methods: dynamic regression methods revisited", Econometrics Journal,
% vol. 9, p. 357-372.
% ---------------------------------------------------------
% NOTE: This version only considers the case p=1 (without intercept).
% Initial condition for x is provided via backasting using an univariate
% AR(p) model. The order of the AR is determined via AIC. It can be changed
% to BIC. The functions ar_order(), arx() and upredict() belong to the BayARX library,
available at MATLAB Central File Exchange.
```

**GUERRERO: Univariate benchmarking and interpolation based on indicators.
High-frequency model: ARIMA-based benchmarking.**

```
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
%            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
%            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.
```

MULTIVARIATE BENCHMARKING WITH INDICATORS AND TRANSVERSAL CONSTRAINTS

INFORMATION SET:

LOW-FREQUENCY VECTOR TIME SERIES (Benchmarks)
HIGH-FREQUENCY VECTOR TIME SERIES (Indicators)
HIGH-FREQUENCY TIME SERIES (Transversal constraint)

- **DENTON: additive and proportional variants:** `denton()`
- **ROSSI:** `rossi()`
- **DI FONZO:** `difonzo()`

DENTON: Multivariate benchmarking and interpolation, with transversal constraint. Additive and proportional variants.

```
function res = denton(Y,x,z,ta,sc,d,op1)
% PURPOSE: Multivariate temporal disaggregation with transversal
% constraint. Denton method, additive or proportional variants.
% ------------------------------------------------------------------------
% SYNTAX: res = denton(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
%            sc= 4 ---> annual to quarterly
%            sc=12 ---> annual to monthly
%            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: difonzo, mtd_print, mtd_plot
% ---------------------------------------------------------------------
% 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
```

ROSSI: Multivariate benchmarking and interpolation with indicators, with transversal constraint.

```
function res = rossi(Y,x,z,ta,sc,opMethod,type)
% PURPOSE: Multivariate temporal disaggregation with transversal constraint
% ------------------------------------------------------------------------
% SYNTAX: res = rossi(Y,x_ini,z,ta,sc,opMethod);
% ------------------------------------------------------------------------
% 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
%           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)
%           Pretesting for intercept is made: 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.
```

DIFONZO: Multivariate benchmarking and interpolation with indicators, with transversal constraint.

```
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.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
%            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, mtd_print, mtd_plot
% ------------------------------------------------------------------------------
% 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.
```

MULTIVARIATE BALANCING

<u>INFORMATION SET</u>:

INITIAL ESTIMATES

- **PROPORTIONAL BALANCING:** `bal()`
- **VAN DER PLOEG:** `vdp()`
- **BI-PROPORTIONAL BALANCING: RAS METHOD:** `ras()`

BAL: Transversal balancing by proportional adjustment

```
function yb = bal(y,z)
% PURPOSE: Proportional adjustment of y to a given total z
% ---------------------------------------------------------------------------
% SYNTAX: yb = bal(y,z);
% ---------------------------------------------------------------------------
% OUTPUT: yb : nxM  --> balanced series
% ----------------------------------------------------------------------
% INPUT: y: nxM  --> unbalanced series
%        z: nx1 --> transversal constraint
% ----------------------------------------------------------------------
% LIBRARY: vec, desvec
% ----------------------------------------------------------------------
% SEE ALSO: denton
% ----------------------------------------------------------------------
% 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
```

van der PLOEG : Transversal balancing by means of QL optimization

```
function res = vdp(y,S,A,a)
% PURPOSE: Balancing by means of QL optimization (LS estimation)
% ------------------------------------------------------------------------
% SYNTAX: res = vdp(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:
% ------------------------------------------------------------------------
% 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.
```

RAS: Balancing by means of Bacharach bi-proportional method

```
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 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 -> graphical info about convergence (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.
```

UTILITIES

- **Backtesting:** `backtest()`
- **Systematic sampler:** `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()`

Backtest: recursive estimates and revisions for some procedures

```
function res = backtest(rex,Nh);
% PURPOSE: Backtesting of temporal disaggregation.
% ------------------------------------------------------------------------
% SYNTAX: res = backtest(rex,Nh);
% ------------------------------------------------------------------------
% OUTPUT: res: a structure with...
%         rex  : reference structure
%         yh   : recursive estimations from FROM T=N+Nh T=N
%         yrev : revision as %, compared with final (full info) estimation
% ------------------------------------------------------------------------
% INPUT: rex: a structure generated by chowlin(), fernandez(), litterman()
%             or ssc()
%        Nh: 1x1 --> number of low-freq. to compute revisions
% ------------------------------------------------------------------------
% LIBRARY: chowlin, fernandez, litterman, ssc
% ------------------------------------------------------------------------
% NOTE: Revision of estimates when adding low-frequency data from T=N-Nh to
% T=N
```

Systematic sampler

```
function [zs] = ssampler(z,op1,sc)
% PURPOSE: Systematic sampling of a high-frequency time series
% ------------------------------------------------------------
% SYNTAX: zs = ssampler(z,op1,sc)
% ------------------------------------------------------------
% OUTPUT: zs: nx1 sampled time series
% ------------------------------------------------------------
% INPUT:  z: nx1 ---> vector of high frequency data
%         op1: type of temporal aggregation
%         op1=1 ---> sum (flow)
%         op1=2 ---> average (index)
%         op1=3 ---> last element (stock) ---> interpolation
%         op1=4 ---> first element (stock) ---> interpolation
%         sc: number of high frequency data points
%           for each low frequency data points
% ------------------------------------------------------------
% LIBRARY: copylow, temporal_agg
% ------------------------------------------------------------
```

Temporal aggregation

```
function [y] = temporal_agg(z,op1,sc)
% PURPOSE: Temporal aggregation of a time series
% ------------------------------------------------------------
% SYNTAX: y = temporal_agg(z,op1,sc);
% ------------------------------------------------------------
% OUTPUT: y: Nx1 temporally aggregated series
% ------------------------------------------------------------
% INPUT:  z: nx1 ---> vector of high frequency data
%         op1: type of temporal aggregation
%         op1=1 ---> sum (flow)
%         op1=2 ---> average (index)
%         op1=3 ---> last element (stock) ---> interpolation
%         op1=4 ---> first element (stock) ---> interpolation
%         sc: number of high frequency data points
%            for each low frequency data points
%         Note: n = sc x N
% ------------------------------------------------------------
% LIBRARY: aggreg
% ------------------------------------------------------------
```

Temporal aggregation preserving input dimension

```
function [y] = temporal_agg_p(z,op1,sc)
% PURPOSE: Temporal aggregation of a time series preserving its dimension
% ------------------------------------------------------------
% SYNTAX: y = temporal_agg_p(z,op1,sc);
% ------------------------------------------------------------
% OUTPUT: y: nx1 temporally aggregated series, missing=0
% ------------------------------------------------------------
% INPUT:  z: nx1 ---> vector of high frequency data
%         op1: type of temporal aggregation
%         op1=1 ---> sum (flow)
%         op1=2 ---> average (index)
%         op1=3 ---> last element (stock) ---> interpolation
%         op1=4 ---> first element (stock) ---> interpolation
%         sc: number of high frequency data points
%           for each low frequency data points
%         Note: n = sc x N
% ------------------------------------------------------------
% LIBRARY: aggreg
% ------------------------------------------------------------
```

Temporal accumulation

```
function [za] = temporal_acc(z,op1,sc);
% PURPOSE: Accumulate within a period of sc observations
% ------------------------------------------------------------
% SYNTAX: za = temporal_acc(z,op1,sc);
% ------------------------------------------------------------
% OUTPUT: za: nx1 accumulated vector
% ------------------------------------------------------------
% INPUT:  op1: type of temporal aggregation
%         op1=1 ---> sum (flow)
%         op1=2 ---> average (index)
%         sc: number of high frequency data points
%            for each low frequency data points (freq. conversion)
% ------------------------------------------------------------
% LIBRARY: acc
% ------------------------------------------------------------
% SEE ALSO: temporal_agg
% ------------------------------------------------------------
```

Moving sum or average

```
function [zs] = moving_acc(z,op1,sc);
% PURPOSE: Moving sum (average) with accumulation=sc
% ------------------------------------------------------------
% SYNTAX: zs = moving_acc(z,op1,sc);
% ------------------------------------------------------------
% OUTPUT: zs: nx1 moving sum (average) vector with (sc-1) initial zeros
% ------------------------------------------------------------
% INPUT:  op1: type of temporal aggregation
%         op1=1 ---> sum (flow)
%         op1=2 ---> average (index)
%         sc: number of high frequency data points
%            for each low frequency data points (freq. conversion) = size
%            of sum filter
% ------------------------------------------------------------
% LIBRARY: movingsum
% ------------------------------------------------------------
% SEE ALSO: temporal_acc, temporal_agg
% ------------------------------------------------------------
```

Cast low-frequency data into high-frequency format

```
function [z] = copylow(Z,op1,sc)
% PURPOSE: Generates a high-frequency time series from a low-frequency one
% ---------------------------------------------------------------------------
% SYNTAX: z = copylow(Z,op1,sc);
% ---------------------------------------------------------------------------
% OUTPUT: z : nxk high frequency time series
% ---------------------------------------------------------------------------
% INPUT: Z     : an Nxk matrix of low frequency series, columnwise
%        op1   : type of temporal aggregation
%        op1=1 ---> copy sc times the lf data
%        op1=2 ---> copy sc times the mean lf data
%        op1=3 ---> last element (stock) ---> interpolation
%        op1=4 ---> first element (stock) ---> interpolation
%        sc: number of high frequency data points
%        for each low frequency data points (quarterly: sc=4, monthly: sc=12)
% ---------------------------------------------------------------------------
% LIBRARY:
% ---------------------------------------------------------------------------
% SEE ALSO: temporal_agg
% ---------------------------------------------------------------------------
```

EXTENSIONS

- **CHOW-LIN by Cochrane-Orcutt:** `chowlin_co()`
- **Extended interpolation:** `aggreg_v_X()`

CHOW-LIN _CO: Univariate benchmarking and interpolation based on indicators. High-frequency model: linear model + AR(1) disturbances. Estimation is performed by preliminary Cochrane-Orcutt estimation of the implied low-frequency model.

```
function res = chowlin_co(Y,x,ta,sc,opC)
% PURPOSE: Temporal disaggregation using the Chow-Lin method
%        (quarterly rho derived from Cochrane-Orcutt annual rho)
% ------------------------------------------------------------
% SYNTAX: res = chowlin_co(Y,x,ta,sc,opC);
% ------------------------------------------------------------
% OUTPUT: res: a structure
%           res.meth    ='Chow-Lin';
%           res.ta      = type of disaggregation
%           res.type    = method of estimation
%           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.aic     = Information criterion: AIC
%           res.bic     = Information criterion: BIC
%           res.co      = Cochrane-Orcutt regression (see LeSage
%           Econometric Toolbox)
% ------------------------------------------------------------
% 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)
%        sc: number of high frequency data points for each low frequency data points
%            sc= 3 ---> quarterly to monthly
%            sc= 4 ---> annual to quarterly
%        opC: 1x1 option related to intercept
%            opc = -1 : pretest intercept significance
%            opc =  0 : no intercept in hf model
%            opc =  1 : intercept in hf model
% ------------------------------------------------------------
% LIBRARY: chowlin_co
% ------------------------------------------------------------
% SEE ALSO: chowlin, litterman, fernandez, td_plot, td_print
% ------------------------------------------------------------
% 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. y Laroque, G. (1979) "Reflexions sur la methode d'elaboration
% des comptes trimestriels", Annales de l'INSEE, n. 36, p. 3-30.
```

Extended interpolation: generation of basic vector

```
function [c] = aggreg_v_X(op1,sc)
% PURPOSE: Generate a temporal aggregation vector (Extended version)
% ------------------------------------------------------------
% SYNTAX: c=aggreg_v_X(op1,sc);
% ------------------------------------------------------------
% OUTPUT: c: 1 x sc temporal aggregation vector
% ------------------------------------------------------------
% INPUT:  op1: type of temporal aggregation
%         op1 = -1 ---> average (index)
%         op1 =  0 ---> sum (flow)
%         op1 =  h ---> interpolates at h element
%               e.g. h=sc is stock at end of low-freq. period
%         sc: number of high frequency data points
%            for each low frequency data points (freq. conversion)
% ------------------------------------------------------------
% LIBRARY:
% ------------------------------------------------------------
% SEE ALSO: aggreg_v
```