Chapter 19: Markov Switching estimation[1]

The empirical observation of the American economy has since a long time led economists to the conclusion that recession and expansion periods did not share the same dynamic and that these periods therefore should obey to two different economic *regimes*. This observation has given rise to the "Markov switching model" first proposed in econometrics by Goldfeld and Quandt and popularized by Hamilton[2] and whose estimation is the purpose of this chapter. Grocer allows the estimation of an extensive set of Markov switching models, through the translation from the Msvarlib package developed by Benoît Bellone for the gauss language (see Bellone (2005))[3]. The first part briefly presents the theoretical background behind Markov Switching models. The second one shows how such models can be estimated with grocer. And as usual the third part presents the functions and their specifications.

1. A brief theoretical overview

The Markov-switching model applies to data who are subject to regime shifts. More precisely, suppose that k endogenous variables y1,...,yk, observed at dates 1,...,T are taken from the following DGP:

$$y_t = x_t \beta_i + z_t \gamma + u_t$$

where:

- $y_t = (y_{1t},...,y_{kt})$
- $x_t = (x_{1t},...,x_{pt})$ is a (1xp) vector of switching exogenous variables
- z_t is a (1 x q) vector of non-switching exogenous variables
- β_i is a (px1) vector of coefficients
- γ is a (qx1) vector of coefficients
- u_t is a (1xk) vector of residuals

i refers to one among M regimes in which the variables can pass by. Because of the existence of the residual, i is not observable. Some coefficients (the β_is) depend on the regime, the other ones (γ) not.

Denote by s_t the regime to which the variables may belong to at date t and p_{ij} the unconditional transition probability that $s_t = i$ when the state at date (i-1) is $s_{t-1} = j$. The Markov switching model assumes that the matrix P of the transition probabilities $[p_{ij}]$ is constant over time and that it sums up all the time dependency between the states. Let θ be the vector of all the parameters, that is the β_i for i=1,...,M, γ, the matrix of probabilities $P=[p_{ij}]$; the M variance matrices Ω_i of the residuals.

Under these conditions, the model can be estimated by a maximum likelihood method. The method used by Benoît Bellone, and therefore by Grocer, is based upon the Kittagawa-Hamilton

[1] Many thanks to Benoît Bellone for all his help and support on this implementation of his package MSVARlib and careful reading of this chapter.

[2] A New Approach to the Economic Analysis of Nonstationary Time Series and the Business Cycle, *Econometrica*, march.

[3] « Classical Estimation of Multivariate Markov-Switching Models using MSVARlib », available at http://bellone.ensae.net/MSVARlib-v2.0.pdf.

filter. It is presented in detail in Bellone (2005):

- give all the parameters starting values (these are calculated automatically by grocer function ms_estimate, either from an a priori datation of the regimes given by the user or by splitting the period into M parts); these include the matrix of transition probabilties $P=[p_{ij}]$;

maximise the log-likelihood logL calculated through the following steps:

- calculation of the vector $P_{1|0}$ probabilities of each state at date 1 conditional on the state 0; these probabilities are calculated from the matrix P as the unconditional -also called ergodic- probabilities (performed in grocer function MSVAR_Ergodic);
- at each date t=1,...,T calculate (performed in grocer function MSVAR_Filt):
  - ✔ the log-likelihood for each state:
    $f(y_t|s_t = i,I_{t-1},\theta) = (2\pi)^{(-p/2)}.\det(\Sigma_j^{(-1/2)}).\exp(-(y_t-m_{t,j})' \Sigma_j^{(-1)} (y_t-m_{t,j})/2)$
  - ✔ the vector of probabilities $P(S(t)=i,y_t|I_{t-1},\theta)$ calculated as $(P .* [P(S_{t-1}=j)])'$[4]
  - ✔ $f(y_t|I_{t-1},\theta) = \sum_{i=1}^{M} [f(y_t \mid s_t = i, I_{t-1},\theta) . P(s_t = i, I_{t-1},\theta)]$
  - ✔ $P(s_t = i|I_t,\theta) = P(S(t)=i,y_t|I_{t-1},\theta)/f(y_t|I_{t-1},\theta)$
  - ✔ $\log L= \Sigma f(y_t|I_{t-1},\theta)$

The estimation provides three different types of output:

- the vector of parameters θ;
- the filtered probabilities: $P(s_t = i/I_t,\theta)$ where I_t is the information available at date t and θ is the (estimated) vector of parameters
- the smoothed probabilities: $P(s_t = i/I_T,\theta)$ where I_T is the information available over the whole estimation period and θ is the (estimated) vector of parameters

2. Estimating a Markov switching model

There are 3 different high level functions that estimate a MS model in grocer: the most general one, called ms_reg and two particular cases, ms_mean and ms_var. There is also a low level function ms_estimate, that estimates a MS model but with restrictions on the data which can only be in a matrix form.

The most general function deals with any Markov-switching model presented in part 1.

A call to ms_reg therefore involves entering successively and in that order the list of endogenous variables (say endo), the list of exogenous non switching variables (say exo_com), the list of exogenous switching variables (say exo_idio) and the number of states (say MS_M). Two further options are also compulsory: the choice between a common variance matrix of residuals for all states or a different variance matrix of residuals for each state (a scalar, say MS_M_V) that takes the value 1 in the first case and the number of states in the other case; the choice between heteroskedasticity, homoskedasticity or no constraint for this variance matrix (a scalar, say MS_var_opt, that takes the value 1, 2 or 3).

The simplest call to ms_reg takes therefore the following form:

[4] Where .* represents the elements by elements product.

r=ms_reg(endo,exo_com,exo_idio,MS_M,MS_M_V,MS_var_opt)

Note that, because of the multiplicity of variable types (exogenous, switching or non switching exogenous variables), there are more constraints on their representation that is usual in Grocer. Precisely, endo, exo_com and exo_idio can have only 2 types: string vector or list. This still allows a great flexibility with respect to the data that can be entered in ms_reg: time series, real vectors or matrices, the string 'const' are still allowed; they only have to be entered between quotes in a string matrix or without or between quotes in a list. Note that only variables entered between quotes will have their names kept for the display of the estimation results.

There are also some options:

- one can transform the data within the functions. Two transformations are allowed: demeaning and studentizing. The corresponding option is 'transf=dem' or 'tansf=stu'. By default, no transformation is done.
- one can provide an a priori datation, that will be used to estimate starting values of the parameters. The corresponding option is 'datation=x' where x is a times series (or a vector) of 1,2,...,n where x(t) takes the supposed value of the state at date t.
- one can choose the value of the increment used for the calculation of the numerical derivatives such as the “Gradiant” or the “Hessian”: the corresponding options are 'gdelta=val' or 'hdelta=val' where val is the chosen value (default is 1E-5).
- one can choose what results will be printed, by the option 'prt=nothing' (or 'noprint'), 'prt=all' (or prt=['initial;final']) if the user does not want to print any result or if she wants to print all results, that is initial results at starting values as well as the final result with the estimated parameters.

As an example, take the estimation of a univariate MS-AR(3) provided in function ms_reg_d:

```
-->load('SCI\macros\grocer\db\us_revu.dat')
```

// load the inverse of the US unemployment rate in the use over the period 1960m1- 2004m3.

```
-->bounds('1967m4','2004m2')
```

// restricts the estimation period to 1967m4- 2004m2.

```
-->nb_states=2
```

// the number of states is set to 2

```
-->switch_var=2
```

// variances are switching

```
-->var_opt=3
```

// unrestricted var-cov matrix

```
-->r=ms_reg('100*(log(us_revu)-lagts(2,log(us_revu)))',...
-->['100*(lagts(1,log(us_revu))-lagts(3,log(us_revu)))';...
-->'100*(lagts(2,log(us_revu))-lagts(4,log(us_revu)))';...
-->'100*(lagts(3,log(us_revu))-lagts(5,log(us_revu)))'],'const',nb_states,...
--> switch_var,var_opt,'transf=stud','prt=initial;final')
```

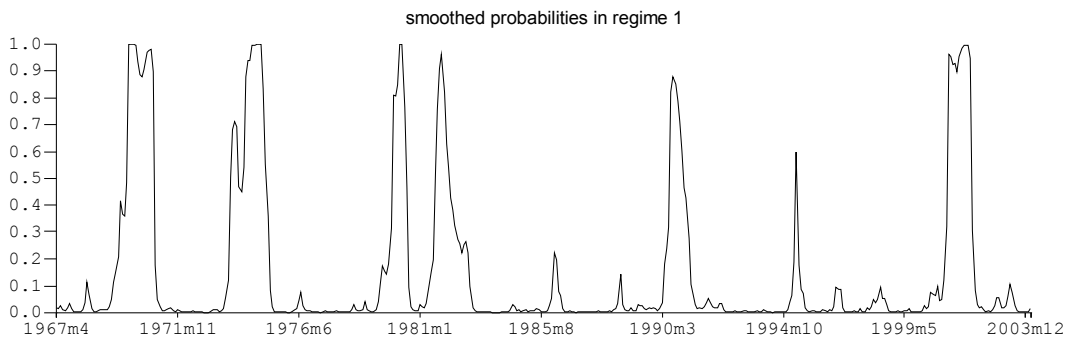
// provides estimation

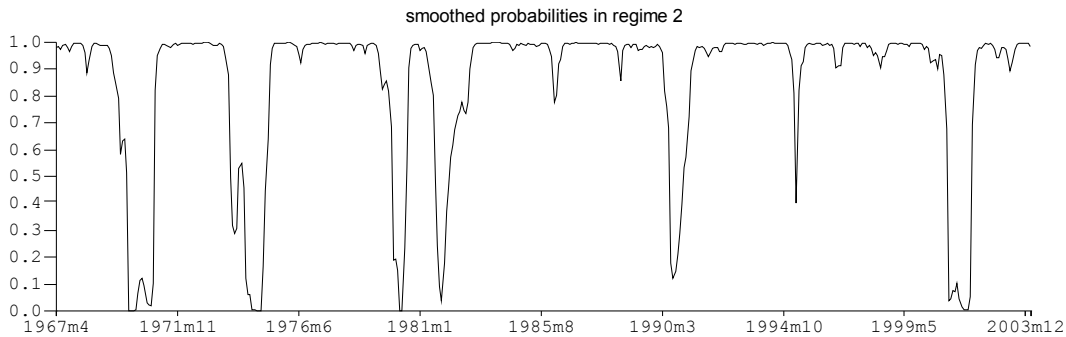
```
Initial statistical moments and parameters

 mu_out:

  - 0.7375131
    0.7341910



 cov1_switch (Regime in columns, vectorized covariance matrix):

    0.8380785    0.4596384



 cov3_switch (Regime in columns, vectorized covariance matrix):

    0.8380785    0.4596384



 cov1:

    0.9988707



 cov3:

    0.9988707

 ================Initial real and transformed parameters, para_m and param_init:==================

    2.1972246    0.9
    2.1972246    0.9
  - 0.5088768  - 0.5088768
    0.5065846    0.5065846
    0.6261278    0.3920360
    0.4454813    0.1984536
    0.5158985    0.5158985
  - 0.259515   - 0.259515
    0.2370234    0.2370234

 rows(param)

    9.



      ==================Initial      matrix      of      transition      markovian      probabilities,
ptrans_init:==================

    0.9    0.9
    0.1    0.1



 ==================Initial switching regressors (or intercepts), Beta :==================

  - 0.5088768    0.5065846



 ==================Initial ergodic state probabilities, prob_st_init:==================

    0.9
    0.1



 ==================Initial state covariance, var_init:========================
```

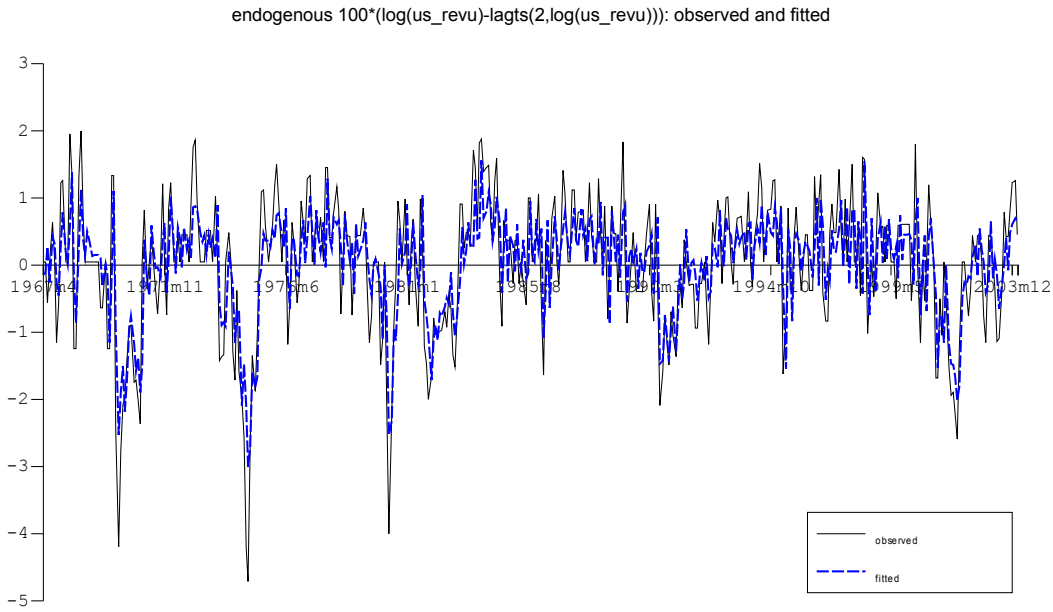
```
    0.3920360
    0.1984536


 ========================Det(inv(var_cov_init)):========================

    2.5507859    5.0389613


 ==================Initial non switching regressors (or intercepts), Delta :==================

    0.5158985
  - 0.259515
    0.2370234

  Optimization step, be patient ...



ms regression estimation results

estimation period: 1967m4-2004m2
number of observations: 443
number of variables: 9
Log likelihood: -451.83309
Degree of freedom: 434


=================Matrix of markovian transition probabilities P[i,j]: ==================
=================                [tstat - p-value]                     =================

 0.8515577         0.0261359
 [20.05 - 0]       [2.91 - 0.0038212]

 0.1484423         0.9738641
 [3.5 - 0.0005222] [108.37 - 0]


================= Ergodic probabilities :=================

0.1497089
0.8502911

================= Coefficients

============== All Regimes =============
exogenous                                                          coeff      t-statistic p value

*** for endogenous: 100*(log(us_revu)-lagts(2,log(us_revu))) ***
100*(lagts(1,log(us_revu))-lagts(3,log(us_revu)))                  0.6416687  8.3856322   6.661D-16
100*(lagts(2,log(us_revu))-lagts(4,log(us_revu)))                  -0.4550183 -7.7532367  6.439D-14
100*(lagts(3,log(us_revu))-lagts(5,log(us_revu)))                  0.3454198  8.0459549   8.216D-15


============== Regime 1 =============

exogenous                                                          coeff      t-statistic p value

*** for endogenous: 100*(log(us_revu)-lagts(2,log(us_revu))) ***
const                                                                -0.8031928 -7.1455419  3.816D-12

Variance-covariance matrix of residuals
           [tstat - p-value]

            0.7070971
            [6.25 - 9.921D-10]


============== Regime 2 =============
```

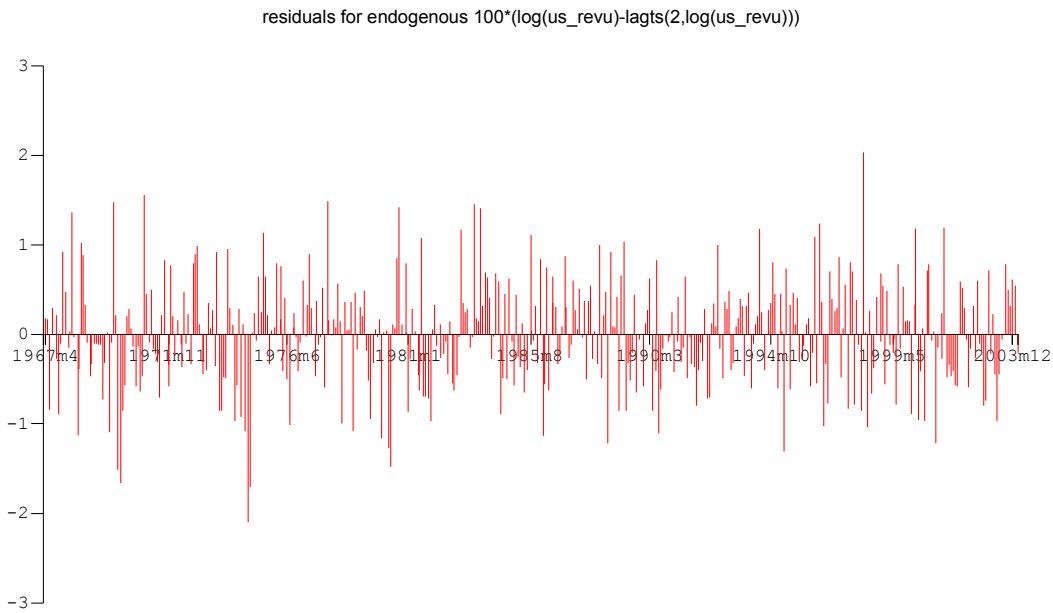
```
exogenous                                                          coeff     t-statistic p value

*** for endogenous: 100*(log(us_revu)-lagts(2,log(us_revu))) ***
const                                                               0.1445671 0.9378597   0.3488382

Variance-covariance matrix of residuals
           [tstat - p-value]

            0.3496791
            [9.11 - 0]
```

The function also provides 3 types of graphs (see below the results for the preceding example): graph(s) of the smoothed probabilities in each regime; graph(s) of observed and fitted values; graph(s) of residuals).







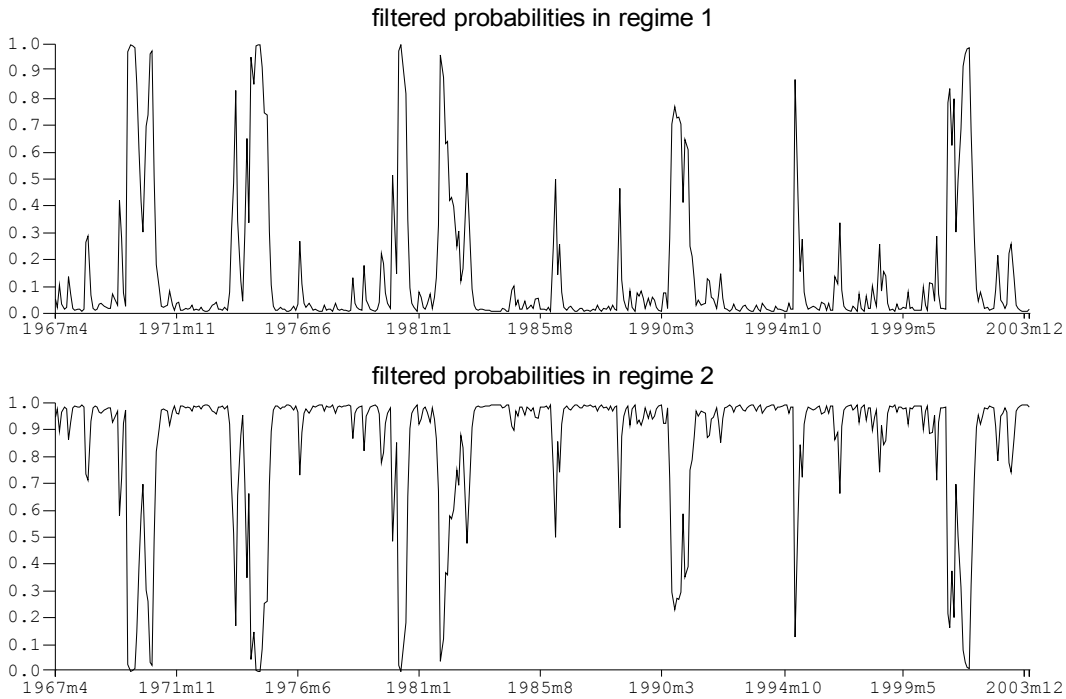


Note that the probabilities here graphed are the smoothed probabilities, that is the probabilities calculated using the estimated parameters and the information available over the whole period, and not the filtered probabilities. Filtered probabilities present the unappealing characteristics, neither to use the whole information set, not to be true real time estimates of the probabilities (contrary to what is often asserted), because the parameters used to calculate these probabilities have

been estimated over the whole period[5]. They can however be recovered (through the field 'filtered probs' of the results tlist -r in the above example) and graphed by the instruction:

```
-->pltms_prob(r,'filtered')
```

which, with the above example, gives the following graph:



The function ms_mean provides the estimation of a very particular case: a MS model where the only exogenous variable is the constant. In that case, there is one switching variable: the constant (if it were not switching, the model would not be a MS one anymore!) and no switching variable.

The syntax of the function ms_mean is therefore very similar to the one of the function ms_reg: except for the -switching and non switching- exogenous variables which are in this case useless, since they are predetermined ([] and 'const' respectively).

The simplest call to ms_mean is for instance:
ms_mean(endo,MS_M,MS_M_V,MS_var_opt)

and is equivalent to:
ms_reg(endo,[],'const',MS_M,MS_M_V,MS_var_opt)

[5] Strictly speaking, a true real tme estimation would require at each step to estimate a new set of parameter, which would be conditioned by the information set (I(t)).

The function ms_var provides the estimation of another peculiar case of a MS model: a MS model where the endogenous variables obey to a VAR model, which differs across states. Two natural cases are allowed: one where only the constant differs across states and the other one where all coefficients differ across states. The choice between these two cases is governed by the first argument of ms_var.

The simplest call to ms_var takes the following form:
ms_var(typvar,p,endo,MS_M,MS_M_V,MS_var_opt)

where:

- MS_M,MS_V MS_var_opt are, as previously, the # of states, 1 or MS_M if the variances are the same or differ across states and MS_var_opt indicates whether the variance matrix of residuals is heteroskedastic, homoskedastic or unconstrained;
- endo is the list of endogenous variables in the var (or a string vector whose elements are the names of the endogenous variables);
- p is the order of the VAR
- typvar is either 'const' if only the constant is allowed to differ across the states or 'all' of both the constant and the lags of the endogenous if variables are allowed to differ across the states.

Examples of functions ms_mean and ms_var are provided in functions ms_var_d and ms_var_d.

As in ms_reg, the following options are available in ms_mean and ms_var:

- one can transform the data within the functions. Two transformations are allowed: demeaning and studentizing. The corresponding option is 'transf=dem' or 'tansf=stu'. By default, no transformation is done.
- one can provide an a priori datation, that will be used to estimate starting values of the parameters. The corresponding option is 'datation=x' where x is a times series (or a vector) of 1,2,...,n where x(i) takes the supposed value of the state at date i.
- one can choose the value of the increment used for the calculation numerical derivative or hessian: the corresponding options are 'gdelta=val' or 'hdelta=val' where val is the chosen value (default is 1E-5).
- one can choose what results will be printed, by the option 'prt=nothing' (or 'noprint'), 'prt=all' (or prt=['initial;final']) if the user does not want to print any result or if she wants to print all results, that is the result at starting values as well as the result at the estimated parameters.

3. The functions and their specifications.

ms_estimate_____estimate a MS model

CALLING SEQUENCE

res=ms_estimate(y,x,z,transf,MS_typmod,MS_M,MS_M_V,MS_var_opt,MS_apriori,prt)

PARAMETERS

INPUT:
* y = (T x K) matrix of endogenous variables
* x = (T x n_x) matrix of switching exogenous variables
* z = (T x n_z) matrix of non switching exogenous variables
* transf = the way data are transformed for the estimation:
  - 'none' if there is no transformation
  - 'dem' if the data are demeaned
  - 'stu' if the data are studentized
* MS_typmod= type of MS model
  - 1: mean-variance switching model
  - 2: MS VAR regime dependent model
  - 3: MS VAR intercept regime dependent model
  - 4: partially regime dependent MS regression model
  - 5: regime dependent MS regression model
* MS_M = a scalar equal the # of states
* MS_V = a scalar:
  - 1 if the variance of the residuals is the same for all states
  - MS_M if the variance of the residuals differs among the states
* MS_var_opt = a scalar:
  - 1 if the variance of residuals is heteroskedastic
  - 2 if the variance of residuals is homoskedastic
  - 3 if the variance of residuals is unconstrained
* MS_apriori =
  - 0 if there is no a priori datation
  - 1 if there is an priori datation
* prt = %t if the initial values of the parameters are to be printed

---

OUTPUT:
r = a results tlist with:
* r('meth') = model literal type ('ms mean' 'ms var' or 'ms regression')
* r('typmod') = model numbered type
* r('y') = a (N x M) matrix of original endogenous variables
* r('x') = the (N x n_x) matrix of exogenous switching regressors
* r('z') = the (N x n_z) matrix of exogenous non switching regressors
* r('ymat') = the (N x M) matrix of -transformed- endogenous variables
* r('xmat') = the (N x M) matrix of -transformed- exogenous switching regressors
* r('zmat') = the (N x M) matrix of -transformed- non switching regressors
* r('switching V') = a scalar:
  - 1 if the variance does not switch with the states
  - M if the variance switches with the states
* r('var_opt') = a scalar:
  - 1 if the variance of residuals is heteroskedastic
  - 2 if the variance of residuals is homoskedastic
  - 3 if the variance of residuals is unconstrained
* r('nobs') = the # if observations

* r('nendo') = the # of endogenous variables
* r('nb_states') = the # of states
* r('coeff') = the (np x 1) vector of parameters
* r('llike') = the log-likekihood
* r('grad') = the gradient at the solution
* r('yhat') = the adjusted y
* r('resid') = the residuals of the regression
* r('dll') = the degrees of freedom
* r('prob_st') = the (M x 1) vector of ergodic state probabilities
* r('ptrans') = the (M x M) matrix of transition probabilities
* r('sigma') = the (M*M_V x M) variance-covariance matrix of the residuals
* r('beta_id') = the (1 x n_x*K*M) vector of switching parameters
* r('beta_co') = the (1 x n_z*K) vector of non switching parameters
* r('inv_sigma') = the (K x K) inverse of the variance matrix
* r('det_inv_sigma') = the determinant of the inverse of the variance matrix
* r('smoothed probs') = the (T x M) vector of smoothed probabilities
* r('stderr') = the (np x 1) vector of coefficients standard errors
* r('tstat') = the (np x 1) vector of associated t-stats
* r('pvalue') = the (np x 1) vector of associated p-values
* r('covbeta') = the (np x np) variance-covariance matrix of the parameters
* r('corbeta') = the (np x np) correlation matrix of the parameters
* r('ptrans_tstat') = the (M x 1) vector of t-stats for the transition probabilities
* r('beta_id_tstat') = the (1 x n_x*K*M) vector of t-stats for switching parameters
* r('beta_co_tstat') = the (1 x n_z*K) vector of t-stats for non switching parameters
* r('sigma_tstat') = the (M*M_V x M) matrix of t-stats for the variance-covariance matrix of the residuals
* r('ptrans_pvalue') = the (M x M) matrix of t-stats for transition probabilities
* r('beta_id_pvalue') = the (1 x n_x*K*M) vector of t-stats for switching parameters
* r('beta_co_pvalue') = the (1 x n_z*K) vector of t-stats for non switching parameters
* r('sigma_pvalue') = the (M*M_V x M) matrix of t-stats for the variance-covariance matrix of the residuals

---

DESCRIPTION

Estimates a MSVAR model by the maximum likelihood method. This is a low level function that is called from a high level function such as ms_mean or ms_var.

Example:

r=ms_estimate(grocer_y,grocer_x,grocer_z,grocer_transf,grocer_MS_typmod,grocer_MS_M,grocer_MS_M_V,grocer_MS_var_opt,grocer_MS_apriori,or(grocer_prt == 'initial'))

Example taken from function ms_mean.

ms_mean_____estimate a Markvov Switching (MS) mean-variance model

CALLING SEQUENCE

r=ms_mean(endo,MS_M,MS_M_V,MS_var_opt,arg1,...argn)

PARAMETERS

INPUT:
* endo = 'endo=y1;...;yn' where y1,..., yn are the names of endogenous variables
* MS_M = a scalar equal the # of states
* MS_V = a scalar:
  - 1 if the variance of the residuals is the same for all states
  - MS_M if the variance of the residuals differs among the states
* MS_var_opt = a scalar:
  - 1 if the variance of residuals is heteroskedastic
  - 2 if the variance of residuals is homoskedastic
  - 3 if the variance of residuals is unconstrained
* arg1,...,argn = optional arguments which can be:
  - 'datation=xx' where xx is the name of a series used as an a priori datation (default: no a priori datation)
  - 'transf=xx' where xx is either 'dem' if the user wants all series to be demeaned or 'stu' if the user wants all series to be studentized (default: no transformation)
  - 'gdelta=xx' where xx is a number used to calculate the numerical derivative of the log-likelihood (default 1e-4)
  - 'hdelta=xx' where xx is a number used to calculate the numerical hessian (default 1e-5)
  - 'prt=xx' where xx='nothing', 'final', 'all' or ['initial, final'] if the user wants to print nothing, only the final results or the final and the initial results
  - 'noprint' if the user wants to print nothing (equivalent to 'prt=nothing')

OUTPUT:
r = a results tlist with:
* r('meth') = 'ms mean'
* r('typmod') = 1
* r('y') = a (N x n_y) matrix of original endogenous variables
* r('x') = the (N x n_x) matrix of exogenous switching regressors = ones(T,n_x)
* r('z') = the [] matrix of exogenous non switching regressors
* r('ymat') = the (N x M) matrix of -transformed- endogenous variables
* r('xmat') = the (N x n_x) matrix of -transformed- exogenous switching regressors
* r('zmat') = the [] matrix of -transformed- non switching regressors
* r('switching V') = a scalar:
  - 1 if the variance does not switch with the states
  - M if the variance switches with the states
* r('var_opt') = a scalar:
  - 1 if the variance of residuals is heteroskedastic
  - 2 if the variance of residuals is homoskedastic
  - 3 if the variance of residuals is unconstrained

* r('nobs') = the # if observations
* r('nendo') = the # of endogenous variables
* r('nb_states') = the # of states
* r('coeff') = the (np x 1) vector of parameters
* r('llike') = the log-likekihood
* r('grad') = the gradient at the solution
* r('yhat') = the adjusted y
* r('resid') = the residuals of the regression
* r('dll') = the degrees of freedom
* r('prob_st') = the (M x 1) vector of ergodic state probabilities
* r('ptrans') = the (M x M) matrix of transition probabilities
* r('sigma') = the (M*M_V x M) variance-covariance matrix of the residuals
* r('beta_id') = the (1 x K*M) vector of switching parameters
* r('beta_co') = the [] vector of non switching parameters
* r('inv_sigma') = the (K x K) inverse of the variance matrix
* r('det_inv_sigma') = the determinant of the inverse of the variance matrix
* r('smoothed probs') = the (T x M) vector of smoothed probabilities
* r('stderr') = the (np x 1) vector of coefficients standard errors
* r('tstat') = the (np x 1) vector of associated t-stats
* r('pvalue') = the (np x 1) vector of associated p-values
* r('covbeta') = the (np x np) variance-covariance matrix of the parameters
* r('corbeta') = the (np x np) correlation matrix of the parameters
* r('ptrans_tstat') = the (M x 1) vector of t-stats for the transition probabilities
* r('beta_id_tstat') = the (1 x K*M) vector of t-stats for switching parameters
* r('beta_co_tstat') = the [] vector of t-stats for non switching parameters
* r('sigma_tstat') = the (M*M_V x M) matrix of t-stats for the variance-covariance matrix of the residuals
* r('ptrans_pvalue') = the (M x M) matrix of t-stats for transition probabilities
* r('beta_id_pvalue') = the (1 x n_x*K*M) vector of t-stats for switching parameters
* r('beta_co_pvalue') = the [] vector of t-stats for non switching parameters
* r('sigma_pvalue') = the (M*M_V x M) matrix of t-stats for the variance-covariance matrix of the residuals
* r('namey') = the (n_y x 1) vector of names of the endogenous variables
* r('namex_id') = the name of the switching exogenous variables = 'const'
* r('namex_co') = the [] vector of names of the non switching exogenous variables
* r('apriori') = a scalar
  - 0 if there is no a priori datation
  - 1 if there is an a priori datation
* r('prests') = a boolean indicating whether there is are ts in the regression
* r('prests') = a boolean indicating whether there is are ts in the regression
* r('datation') = the a priori datation if any
* r('namedat') = the name of the series used for an a priori datation if any
* r('bounds') = the bounds if there are ts in the regression

DESCRIPTION

Estimates a Markvov Switching (MS) mean-variance model by the maximum likelihood method.

Example:
nb_states=3; switch_var=1; var_opt=1 ; r=ms_mean(['delts(log(construc)), delts(log(ipi))';
'delts(log(helpwanted)), delts(log(revu))'],nb_states,switch_var,var_opt,'transf=stud',
'datation=datation_bb')

Example taken from function ms_mean_d. The endogenous variables are delts(log(construc)), delts(log(ipi)), delts(log(helpwanted)), delts(log(revu)). The number of states is set to 3, the variance of the residuals is not supposed to switch between states (switch_var=1), the residuals are supposed to be heteroskedastic (var_opt=1), endogenous variables are studentized ('transf=stud') and an a priori datation is provided ('datation=datation _bb').

ms_reg_____estimate a Markvov Switching (MS) regression model

CALLING SEQUENCE

r=ms_reg(endo,exo_com,exo_idio,MS_M,MS_M_V,MS_var_opt,arg1,...argn)

PARAMETERS

INPUT:
* endo =
 - (T x K) string matrix of endogenous variables
 or:
 - a list including all the endogenous variables in any of
 the following form:
  . a time series
  . a real matrix
  . a string representing such objects
  . the string 'const' (for the constant variable)
* exo_com =
 - (T x K) string matrix of non switching exogenous variables
 or:
 - a list including all the non switching exogenous variables in any of the following form:
  . a time series
  . a real matrix
  . a string representing such objects
  . the string 'const' (for the constant variable)
* exo_idio =
 - (T x K) string matrix of switching exogenous variables
 or:
 - a list containing all the switching exogenous variables in any of the following form:
  . a time series
  . a real matrix
  . a string representing such objects
  . the string 'const' (for the constant variable)

* MS_M = a scalar equal the # of states
* MS_V = a scalar:
  - 1 if the variance of the residuals is the same for all states
  - MS_M if the variance of the residuals differs among the states
* MS_var_opt = a scalar:
  - 1 if the variance of residuals is heteroskedastic
  - 2 if the variance of residuals is homoskedastic
  - 3 if the variance of residuals is unconstrained
* arg1,...argn = optional arguments which can be:
  - 'datation=xx' where xx is the name of a series used as an a priori datation (default: no a priori datation)
  - 'transf=xx' where xx is either 'dem' if the user wants all series to be demeaned or 'stu' if the user wants all series to be studentized (default: no transformation)
  - 'gdelta=xx' where xx is a number used to calculate the numerical derivative of the log-likelihood (default 1e-4)
  - 'hdelta=xx' where xx is a number used to calculate the numerical hessian (default 1e-5)
  - 'prt=xx' where xx='nothing', 'final', 'all' or ['initial';'final'] if the user wants to print nothing, only the final results or the final and the initial results
  - 'noprint' if the user wants to print nothing (equivalent to 'prt=nothing')

OUTPUT:
r = a results tlist with:
* r('meth') = model literal type ('ms mean' 'ms var' or 'ms regression')
* r('typmod') = model numbered type
* r('y') = a (N x M) matrix of original endogenous variables
* r('x') = the (N x n_x) matrix of exogenous switching regressors
* r('z') = the (N x n_z) matrix of exogenous non switching regressors
* r('ymat') = the (N x M) matrix of -transformed- endogenous variables
* r('xmat') = the (N x M) matrix of -transformed- exogenous switching regressors
* r('zmat') = the (N x M) matrix of -transformed- non switching regressors
* r('switching V') = a scalar:
  - 1 if the variance does not switch with the states
  - M if the variance switches with the states
* r('var_opt') = a scalar:
  - 1 if the variance of residuals is heteroskedastic
  - 2 if the variance of residuals is homoskedastic
  - 3 if the variance of residuals is unconstrained
* r('nobs') = the # if observations
* r('nendo') = the # of endogenous variables
* r('nb_states') = the # of states
* r('coeff') = the (np x 1) vector of parameters
* r('llike') = the log-likekihood
* r('grad') = the gradient at the solution
* r('yhat') = the adjusted y
* r('resid') = the residuals of the regression
* r('dll') = the degrees of freedom
* r('prob_st') = the (M x 1) vector of ergodic state probabilities

* r('ptrans') = the (M x M) matrix of transition probabilities
* r('sigma') = the (M*M_V x M) variance-covariance matrix of the residuals
* r('beta_id') = the (1 x n_x*K*M) vector of switching parameters
* r('beta_co') = the (1 x n_z*K) vector of non switching parameters
* r('inv_sigma') = the (K x K) inverse of the variance matrix
* r('det_inv_sigma') = the determinant of the inverse of the variance matrix
* r('smoothed probs') = the (T x M) vector of smoothed probabilities
* r('stderr') = the (np x 1) vector of coefficients standard errors
* r('tstat') = the (np x 1) vector of associated t-stats
* r('pvalue') = the (np x 1) vector of associated p-values
* r('covbeta') = the (np x np) variance-covariance matrix of the parameters
* r('corbeta') = the (np x np) correlation matrix of the parameters
* r('ptrans_tstat') = the (M x 1) vector of t-stats for the transition probabilities
* r('beta_id_tstat') = the (1 x n_x*K*M) vector of t-stats for switching parameters
* r('beta_co_tstat') = the (1 x n_z*K) vector of t-stats for non switching parameters
* r('sigma_tstat') = the (M*M_V x M) matrix of t-stats for the variance-covariance matrix of the residuals
* r('ptrans_pvalue') = the (M x M) matrix of t-stats for transition probabilities
* r('beta_id_pvalue') = the (1 x n_x*K*M) vector of t-stats for switching parameters
* r('beta_co_pvalue') = the (1 x n_z*K) vector of t-stats for non switching parameters
* r('sigma_pvalue') = the (M*M_V x M) matrix of t-stats for the variance-covariance matrix of the residuals
* r('namey') = the (ny x 1) vector of names of the endogenous variables
* r('namex_id') = the (n_x x 1) vector of names of the switching exogenous variables
* r('namex_co') = the (n_x x 1) vector of names of the non switching exogenous variables
* r('apriori') = a scalar
  - 0 if there is no a priori datation
  - 1 if there is an a priori datation
* r('prests') = a boolean indicating whether there are ts in the regression
* r('prests') = a boolean indicating whether there are ts in the regression
* r('datation') = the a priori datation if any
* r('namedat') = the name of the series used for an a priori datation if any
* r('bounds') = the bounds if there are ts in the regression

DESCRIPTION
Estimates a Markvov Switching (MS) regression model by the maximum likelihood method.

Example:
load('SCI\macros\grocer\db\us_revu.dat');
bounds('1967m4','2004m2'); nb_states=2; switch_var=2;
var_opt=3;
r=ms_reg('100*(log(us_revu)-lagts(2,log(us_revu)))',['100*(lagts(1,log(us_revu))-lagts(3,log(us_revu)))';'100*(lagts(2,log(us_revu))-lagts(4,log(us_revu)))';'100*(lagts(3,log(us_revu))-lagts(5,log(us_revu)))'],'const',nb_states,switch_var,var_opt,'transf=stud','prt=initial;final')

Example taken from function ms_reg_d. The endogenous variables is 100*(log(us_revu)-lagts(2,log(us_revu))). Exogenous non switching variables are 100*(lagts(1,log(us_revu))-lagts(3,log(us_revu))), 100*(lagts(2,log(us_revu))-lagts(4,log(us_revu))) and 100*(lagts(3,log(us_revu))-lagts(5,log(us_revu))). The constant is the only exogenous switching variable. The number of states is set to 2, the variance of the residuals is supposed to switch between states (switch_var=2), the variance of the residuals is unconstrained (var_opt=3), endogenous variables are studentized ('transf=stud') and both initial and final results are printed (this is equivalent to the estimation of the VAR model presented as an example in function ms_var_d.

ms_var_____estimate a Markvov Switching (MS) VAR model

CALLING SEQUENCE

r=ms_var(MS_typvar,MS_p,endo,MS_M,MS_M_V,MS_var_opt,arg1,...argn)

PARAMETERS

INPUT:
* typvar =
  - 'all' if the constant AND the endogenous variables are switching exogenous variables
  - 'const' if only the constant is a switching exogenous variable
* MS_p = the VAR lag order
* endo =
  - (T x K) string matrix of endogenous variables
  or:
  - a list including all the endogenous variables in any of the following form:
    . a time series
    . a real matrix
    . a string representing such objects
    . the string 'const' (for the constant variable)
* MS_M = a scalar equal the # of states
* MS_V = a scalar:
  - 1 if the variance of the residuals is the same for all states
  - MS_M if the variance of the residuals differs among the states
* MS_var_opt = a scalar:
  - 1 if the variance of residuals is heteroskedastic
  - 2 if the variance of residuals is homoskedastic
  - 3 if the variance of residuals is unconstrained
* arg1,...argn = optional arguments which can be:
  - 'datation=xx' where xx is the name of a series used as an a priori datation (default: no a priori datation)
  - 'transf=xx' where xx is either 'dem' if the user wants all series to be demeaned or 'stu' if the user wants all series to be studentized (default: no transformation)
  - 'gdelta=xx' where xx is a number used to calculate the numerical derivative of the log-likelihood (default 1e-4)

- 'hdelta=xx' where xx is a number used to calculate the numerical hessian (default 1e-5)
- 'prt=xx' where xx='nothing', 'final', 'all' or ['initial';'final'] if the user wants to print nothing, only the final results or the final and the initial results
- 'noprint' if the user wants to print nothing (equivalent to 'prt=nothing')

OUTPUT:
r = a results tlist with:
* r('meth') = model literal type: 'ms var'
* r('typmod') = model numbered type
* r('y') = a (N x M) matrix of original endogenous variables
* r('x') = the (N x n_x) matrix of exogenous switching regressors
* r('z') = the (N x n_z) matrix of exogenous non switching regressors
* r('ymat') = the (N x M) matrix of -transformed- endogenous variables
* r('xmat') = the (N x M) matrix of -transformed- exogenous switching regressors
* r('zmat') = the (N x M) matrix of -transformed- non switching regressors
* r('switching V') = a scalar:
  - 1 if the variance does not switch with the states
  - M if the variance switches with the states
* r('var_opt') = a scalar:
  - 1 if the variance of residuals is heteroskedastic
  - 2 if the variance of residuals is homoskedastic
  - 3 if the variance of residuals is unconstrained
* r('nobs') = the # of observations
* r('nendo') = the # of endogenous variables
* r('nb_states') = the # of states
* r('coeff') = the (np x 1) vector of parameters
* r('llike') = the log-likekihood
* r('grad') = the gradient at the solution
* r('yhat') = the adjusted y
* r('resid') = the residuals of the regression
* r('dll') = the degrees of freedom
* r('prob_st') = the (M x 1) vector of ergodic state probabilities
* r('ptrans') = the (M x M) matrix of transition probabilities
* r('sigma') = the (M*M_V x M) variance-covariance matrix of the residuals
* r('beta_id') = the (1 x n_x*K*M) vector of switching parameters
* r('beta_co') = the (1 x n_z*K) vector of non switching parameters
* r('inv_sigma') = the (K x K) inverse of the variance matrix
* r('det_inv_sigma') = the determinant of the inverse of the variance matrix
* r('smoothed probs') = the (T x M) vector of smoothed probabilities
* r('stderr') = the (np x 1) vector of coefficients standard errors
* r('tstat') = the (np x 1) vector of associated t-stats
* r('pvalue') = the (np x 1) vector of associated p-values
* r('covbeta') = the (np x np) variance-covariance matrix of the parameters
* r('corbeta') = the (np x np) correlation matrix of the parameters
* r('ptrans_tstat') = the (M x 1) vector of t-stats for the transition probabilities
* r('beta_id_tstat') = the (1 x n_x*K*M) vector of t-stats for switching parameters
* r('beta_co_tstat') = the (1 x n_z*K) vector of t-stats for non switching parameters

* r('sigma_tstat') = the (M*M_V x M) matrix of t-stats for the variance-covariance matrix of the residuals
* r('ptrans_pvalue') = the (M x M) matrix of t-stats for transition probabilities
* r('beta_id_pvalue') = the (1 x n_x*K*M) vector of t-stats for switching parameters
* r('beta_co_pvalue') = the (1 x n_z*K) vector of t-stats for non switching parameters
* r('sigma_pvalue') = the (M*M_V x M) matrix of t-stats for the variance-covariance matrix of the residuals
* r('namey') = the (ny x 1) vector of names of the endogenous variables
* r('namex_id') = the (n_x x 1) vector of names of the switching exogenous variables
* r('namex_co') = the (n_x x 1) vector of names of the non switching exogenous variables
* r('apriori') = a scalar
  - 0 if there is no a priori datation
  - 1 if there is an a priori datation
* r('prests') = a boolean indicating whether there is are ts in the regression
* r('prests') = a boolean indicating whether there is are ts in the regression
* r('datation') = the a priori datation if any
* r('namedat') = the name of the series used for an a priori datation if any
* r('bounds') = the bounds if there are ts in the regression

DESCRIPTION
Estimates a Markvov Switching (MS) VAR model by the maximum likelihood method.

Example:
```
load('SCI\macros\grocer\db\us_revu.dat');
bounds('1967m4','2004m2');
nb_states=2; switch_var=2;
var_opt=3;
r=ms_var('const',3,'100*(log(us_revu)-lagts(2,log(us_revu)))',nb_states,switch_var,var_opt,
'prt=initial;final','transf=stud')
```

Example taken from function ms_var_d. The constant is the only exogenous variable allowed to switch. The number of states is set to 2, the variance of the residuals is supposed to switch between states (switch_var=2), the variance of the residuals is unconstrained (var_opt=3), endogenous variables are studentized ('transf=stud') and both initial and final results are printed.

MSVAR_stderr_____Markvov Switching 2nd order moments

CALLING SEQUENCE

res=MSVAR_stderr(res,hdelta)

PARAMETERS

INPUT:
* res = a results tlist from a Markov switching estimation

* hdelta = a scalar, the increment used to calculate the hessian matrix
---
OUTPUT:
res = the initial results tlist with the following fields
updated:
* res('stderr') = the (np x 1) vector of coefficients standard errors
* res('tstat') = the (np x 1) vector of associated t-stats
* res('pvalue') = the (np x 1) vector of associated p-values
* res('covbeta') = the (np x np) variance-covariance matrix of the parameters
* res('corbeta') = the (np x np) correlation matrix of the parameters
* res('ptrans_tstat') = the (M x 1) vector of t-stats for the transition probabilities
* res('beta_id_tstat') = the (1 x n_x*K*M) vector of t-stats for switching parameters
* res('beta_co_tstat') = the (1 x n_z*K) vector of t-stats for non switching parameters
* res('sigma_tstat') = the (M*M_V x M) matrix of t-stats for the variance-covariance matrix of the residuals
* res('ptrans_pvalue') = the (M x M) matrix of t-stats for transition probabilities
* res('beta_id_pvalue') = the (1 x n_x*K*M) vector of t-stats for switching parameters
* res('beta_co_pvalue') = the (1 x n_z*K) vector of t-stats for non switching parameters
* res('sigma_pvalue') = the (M*M_V x M) matrix of t-stats for the variance-covariance matrix of the residuals
---

DESCRIPTION
From a MSVAR estimation, computes hessian, gradians, stdev, tstat, pvalues and print parameters, adapted courtesy of Thierry Roncalli (1995)[6]. This is the function used in function ms_estimate to perform these calculations, but it can also be used if the results of a ms estimation does not provide satisfactory results.

Example:
```
load('SCI\macros\grocer\db\us_revu.dat');
bounds('1967m4','2004m2');
nb_states=2;
switch_var=2;
var_opt=3;
r=ms_var('const',3,'100*(log(us_revu)-lagts(2,log(us_revu)))',nb_states,switch_var,var_opt,
'prt=initial;final','transf=stud');
r=MSVAR_stderr(r,1e-3)
```

In this example, the second order moments are re-estimated using a smaller increment than the one used to estimate the ms_var model (1e-5).

[6]Roncalli, T. (1995), Introduction à la programmation sous GAUSS, Applications à la Finance et à l'Econométrie, vol 2, Ritme informatique ed.