mswitch postestimation — Postestimation tools for mswitch

Postestimation commands

The following postestimation commands are of special interest after `mswitch`:

Command	Description
`estat transition`	display transition probabilities in a table
`estat duration`	display expected duration of states in a table

The following standard postestimation commands are also available:

Command	Description
`contrast`	contrasts and ANOVA-style joint tests of parameters
`estat ic`	Akaike's, consistent Akaike's, corrected Akaike's, and Schwarz's Bayesian information criteria (AIC, CAIC, AICc, and BIC, respectively)
`estat summarize`	summary statistics for the estimation sample
`estat vce`	variance–covariance matrix of the estimators (VCE)
`estimates`	cataloging estimation results
`etable`	table of estimation results
`forecast`	dynamic forecasts and simulations
`lincom`	point estimates, standard errors, testing, and inference for linear combinations of parameters
`lrtest`	likelihood-ratio test
`nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of parameters
`predict`	linear predictions, state probabilities, residuals, etc.
`predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`pwcompare`	pairwise comparisons of parameters
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

predict

Description for predict

predict creates new variables containing predictions such as predicted values, probabilities, residuals, and standardized residuals.

Menu for predict

Statistics > Postestimation

Syntax for predict

predict [*type*] {*stub** | *newvarlist*} [*if*] [*in*] [, *statistic options*]

statistic	Description
Main	
yhat	predicted values; the default
xb	equation-specific predicted values; default is predicted values for the first equation
pr	compute probabilities of being in a given state; default is one-step-ahead probabilities
residuals	residuals
rstandard	standardized residuals

These statistics are available both in and out of sample; type predict ... if e(sample) ... if wanted only for the estimation sample.

options	Description
Options	
smethod(*method*)	method for predicting unobserved states; specify one of onestep, filter, or smooth; default is smethod(onestep)
rmse(*stub** | *newvarlist*)	put estimated root mean squared errors of predicted statistics in new variables
dynamic(*time_constant*)	begin dynamic forecast at specified time
equation(*eqnames*)	names of equations for which predictions are to be made

method	Description
onestep	predict using past information
filter	predict using past and contemporaneous information
smooth	predict using all sample information

Options for predict

Main

yhat, xb, pr, residuals, and rstandard specify the statistic to be predicted.

yhat, the default, calculates the weighted and state-specific linear predictions of the observed variables.

xb calculates the equation-specific linear predictions of the observed variables.

pr calculates the probabilities of being in a given state.

residuals calculates the residuals in the equations for observable variables.

rstandard calculates the standardized residuals, which are the residuals normalized to have unit variances.

Options

smethod(*method*) specifies the method for predicting the unobserved states; smethod(onestep), smethod(filter), and smethod(smooth) allow different amounts of information on the dependent variables to be used in predicting the states at each time period. smethod() may not be specified with xb.

smethod(onestep), the default, causes predict to estimate the states at each time period using previous information on the dependent variables. The nonlinear filter is performed on previous periods, but only the one-step predictions are made for the current period.

smethod(filter) causes predict to estimate the states at each time period using previous and contemporaneous data by using the nonlinear filter. The filtering is performed on previous periods and the current period.

smethod(smooth) causes predict to estimate the states at each time period using all sample data by using the smoothing algorithm.

rmse(*stub** | *newvarlist*) puts the root mean squared errors of the predicted statistics into the specified new variables. The root mean squared errors measure the variances due to the disturbances but do not account for estimation error.

dynamic(*time_constant*) specifies when predict starts producing dynamic forecasts. The specified *time_constant* must be in the scale of the time variable specified in tsset, and the *time_constant* must be inside a sample for which observations on the dependent variables are available. For example, dynamic(tq(2014q4)) causes dynamic predictions to begin in the fourth quarter of 2014, assuming that the time variable is quarterly; see [D] **Datetime**. If the model contains exogenous variables, they must be present for the whole predicted sample. dynamic() may not be specified with xb, pr, residuals, or rstandard.

equation(*eqnames*) specifies the equations for which the predictions are to be calculated. If you do not specify equation() or *stub**, the results are the same as if you had specified the name of the first equation for the predicted statistic. equation() may be specified with xb only.

You specify a list of equation names, such as equation(income consumption) or equation(factor1 factor2), to identify the equations.

equation() may not be specified with *stub**.

estat

Description for estat

`estat transition` displays all of the transition probabilities in tabular form.

`estat duration` computes the expected duration that the process spends in each state and displays the results in a table.

Menu for estat

Statistics > Postestimation

Syntax for estat

Display transition probabilities in a table

`estat transition [ , level(#) ]`

Display expected duration of states in a table

`estat duration [ , level(#) ]`

`collect` is allowed with `estat transition` and `estat duration`; see **[U] 11.1.10 Prefix commands**.

Option for estat

`level(#)` specifies the confidence level, as a percentage, for confidence intervals. The default is `level(95)` or as set by `set level`; see **[U] 20.8 Specifying the width of confidence intervals**.

Remarks and examples

Remarks are presented under the following headings:

We assume that you have already read [TS] **mswitch**. In this entry, we illustrate some of the features of `predict` after using `mswitch` to estimate the parameters of a Markov-switching model.

All the predictions after `mswitch` depend on the unobserved states, which are estimated recursively using a nonlinear filter. Changing the sample can alter the state estimates, which can change all other predictions.

One-step predictions

One-step predictions in a Markov-switching model are the forecasted values of the dependent variable using one-step-ahead predicted probabilities.

▷ Example 1: One-step predictions for a series

In example 3 of [TS] **mswitch**, we estimated the parameters of a Markov-switching dynamic regression for the federal funds rate `fedfunds` as a function of its lag, the output gap `ogap`, and `inflation`.

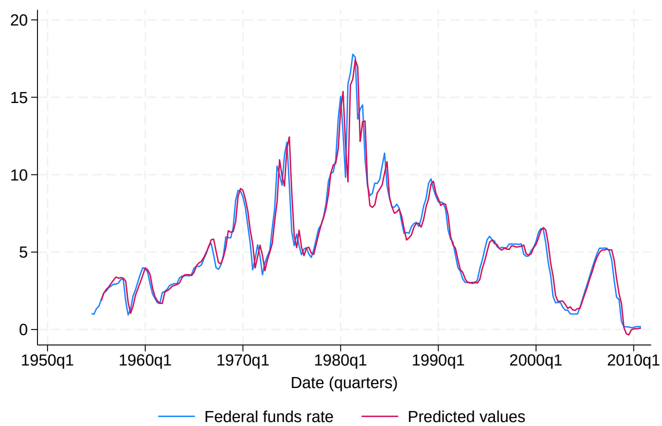
```
. use https://www.stata-press.com/data/r19/usmacro
(Federal Reserve Economic Data - St. Louis Fed)
. mswitch dr fedfunds, switch(L.fedfunds ogap inflation)
  (output omitted)
```

We obtain the one-step predictions for the dependent variable using the default settings for `predict`. The predictions are stored in the new variable `fedf`.

```
. predict fedf
(option yhat assumed; predicted values)
```

Next, we graph the actual values, `fedfunds`, and predicted values, `fedf`, using `tsline`. We change the label for `fedf` to "Predicted values"; see [TS] **tsline**.

```
. tsline fedfunds fedf, legend(label(2 "Predicted values"))
```



The graph shows that one-step-ahead predictions account for large swings in the federal funds rate.

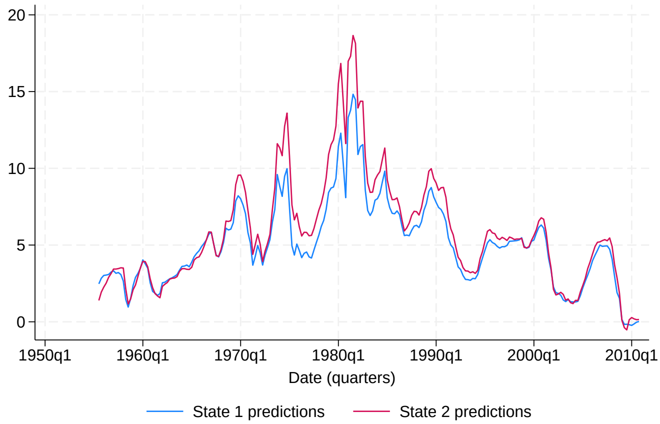
◁

▷ Example 2: State-specific one-step predictions

Continuing example 1, we may also wish to obtain state-specific predictions. This allows us to compare the predictions obtained for different states.

Note that this time, we specify `fedf*` rather than `fedf` so that `predict` generates two state-specific predictions with the prefix `fedf` instead of a single weighted prediction. Also note that the predicted values obtained in example 1 are the weighted average of the state-specific predictions, the weights being the one-step-ahead probabilities.

```
. predict fedf*
(option yhat assumed; predicted values)
. tsline fedf1 fedf2, legend(label(1 "State 1 predictions")
> label(2 "State 2 predictions"))
```



The graph shows that, as expected, the predicted values of `fedfunds` are higher in state 2, the high-interest rate state, than in state 1, the moderate-interest rate state.

Dynamic predictions

Dynamic predictions are out-of-sample forecasted values of the dependent variable using one-step-ahead probabilities.

◁

▷ Example 3: Dynamic predictions for Markov-switching autoregression

In example 6 of [TS] **mswitch**, we estimated the parameters of a Markov-switching autoregression for the US real gross national product as a function of its own lags.

```
. use https://www.stata-press.com/data/r19/rgnp, clear
(Data from Hamilton (1989))
. mswitch ar rgnp, ar(1/4)
  (output omitted)
```

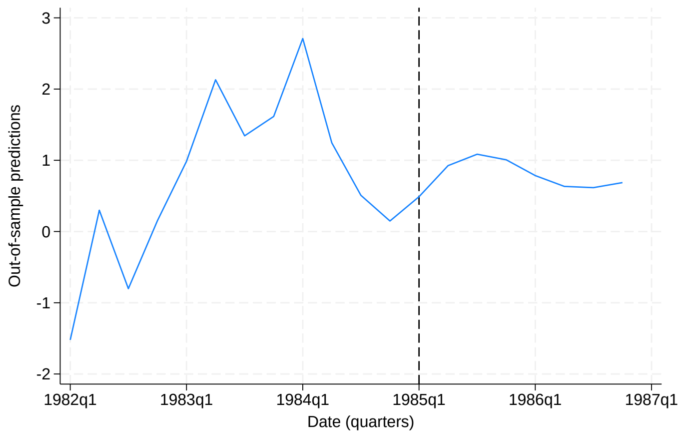
To obtain dynamic predictions, we use `predict` with the `dynamic()` option. The `dynamic()` option requires that all exogenous variables be present for the whole predicted sample. In this example, we have not specified any exogenous variables, so we do not check for that. However, we do need to have time values available for the predictions. So before submitting our `predict` command, we use `tsappend` to extend the dataset by eight periods.

Within `dynamic()`, we specify that dynamic predictions will begin in the first quarter of 1985, and we use the convenience function `tq()` to convert `1985q1` into a numeric date that Stata understands; see [FN] **Date and time functions**.

```
. tsappend, add(8)
. predict rgnp_f, dynamic(tq(1985q1))
(option yhat assumed; predicted values)
```

We again use `tsline` to plot the in- and out-of-sample predictions. We restrict the range to quarters 1982q1 to 1986q4 using function `tin()`.

```
. tsline rgnp_f if tin(1982q1,1986q4), ytitle("Out-of-sample predictions")
> tline(1985q1)
```



The vertical line shows where our out-of-sample prediction begins.

◁

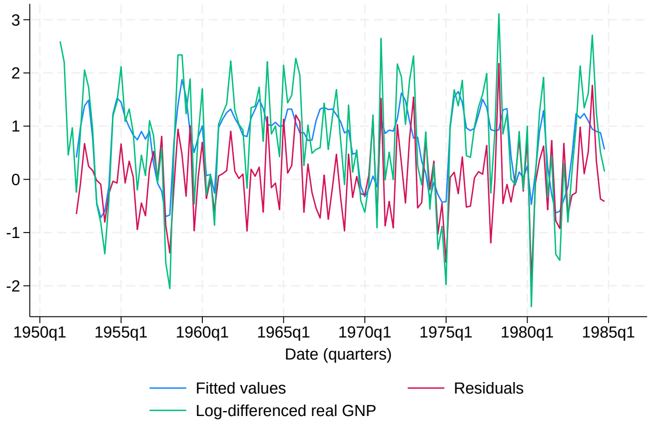
Model fit and state predictions

▷ Example 4: Assessing model fit

In this example, we examine the model fit by comparing the fitted values of US real gross national product and the residuals with the actual data. The fitted values are obtained using smoothed probabilities that consider all sample information.

```
. predict yhat, smethod(smooth)
(option yhat assumed; predicted values)
. predict res, residuals smethod(smooth)
```

```
. tsline yhat res rgnp, legend(label(1 "Fitted values") label(2 "Residuals"))
```



We see in the graph above that we do not obtain a good fit; the residuals account for much of the variation in the dependent variable.

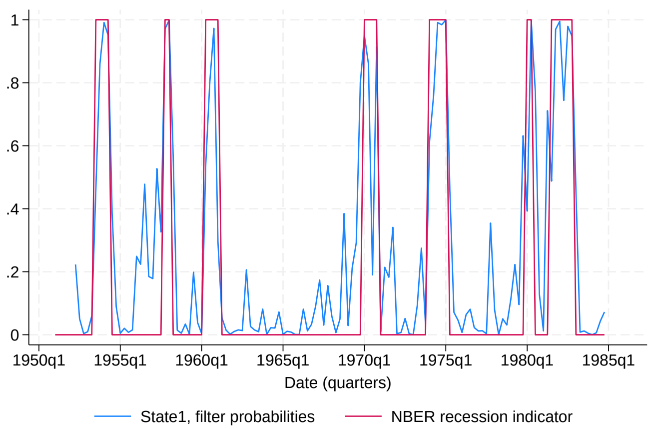
◁

▷ Example 5: Filtered probabilities

Continuing example 4, recall that the states in the model correspond to recession periods and expansion periods for the US economy. State 1 was the recession state. Here we compare the predicted probability of being in state 1 with the National Bureau of Economic Research recession periods stored in the indicator variable `recession`.

To obtain the filtered probabilities, typically used to predict state probabilities, we specify options `pr` and `smethod(filter)` with `predict`.

```
. predict fprob, pr smethod(filter)
. tsline fprob recession
```



The predictions of recession and expansion states fit well with the NBER dates. Thus, it appears that while our model does not have good fit, it does a good job of predicting the probability of being in a given state.

We could also have specified `smethod(smooth)` to obtain better estimates of the state probability using all sample information.

◁

▷ Example 6: Expected duration

Rather than predicting which state the series is in at a point in time, we may wish to know the average time it spends in a given state. We can compute the expected duration of the process being in a given state and show the result in a table using `estat duration`.

Continuing example 5, we can calculate the average length of recession periods and expansion periods for the US economy.

```
. estat duration
Number of obs = 131
```

Expected duration	Estimate	Std. err.	[95% conf.	interval]
State1	4.076159	1.603668	2.107284	9.545915
State2	10.42587	4.101872	5.017004	23.11771

The table indicates that state 1, the recession state, will typically persist for about 4 quarters and state 2, the expansion state, will persist for about 10 quarters.

◁

Stored results

`estat transition` stores the following in `r()`:

Scalars

`r(level)`	confidence level of confidence intervals

Macros

`r(label#)`	label for transition probability

Matrices

`r(prob)`	vector of transition probabilities
`r(se)`	vector of standard errors of transition probabilities
`r(ci#)`	vector of confidence interval (lower and upper) for transition probability

`estat duration` stores the following in `r()`:

Scalars

`r(d#)`	expected duration for state #
`r(se#)`	standard error of expected duration for state #
`r(level)`	confidence level of confidence intervals

Macros

`r(label#)`	label for state #

Matrices

`r(ci#)`	vector of confidence interval (lower and upper) for expected duration for state #

Methods and formulas

Forecasting a Markov-switching model requires estimating the probability of the process being at any given state at each time period. The forecasts are then computed by weighting the state-specific forecasts by the estimated probabilities. Refer to Hamilton (1993) and Davidson (2004) for more details on forecasting Markov-switching regression models.

By default and with the `smethod(filter)` option, `predict` estimates the probability of being at a state in each period by applying a nonlinear filter on all previous periods and the current period. (See *Methods and formulas* of [TS] **mswitch** for the filter equations.)

With the `smethod(smooth)` option, `predict` estimates the probabilities by applying a smoothing algorithm (Kim 1994) using all the sample information. With the `smethod(onestep)` option, `predict` estimates the probabilities using information from all previous periods to make one-step-ahead predictions.

The dependent variable is predicted by averaging the state-specific forecasts by the estimated probabilities. The residuals are computed as the difference between the predicted and the realized dependent variable. The standardized residuals are the residuals normalized to have unit variances.

Using an `if` or `in` qualifier to alter the prediction sample can change the estimate of the unobserved states in the period prior to beginning the dynamic predictions and hence alter the dynamic predictions. The initial values for the probabilities are obtained by calculating the ergodic probabilities from the transition matrix.

References

Davidson, J. 2004. Forecasting Markov-switching dynamic, conditionally heteroscedastic processes. *Statistics and Probability Letters* 68: 137–147. https://doi.org/10.1016/j.spl.2004.02.004.

Hamilton, J. D. 1993. "Estimation, inference and forecasting of time series subject to changes in regime". In *Handbook of Statistics*, edited by G. S. Maddala, C. R. Rao, and H. D. Vinod, vol. 11: 231–260. San Diego: Elseiver. https://doi.org/10.1016/S0169-7161(05)80044-6.

Kim, C.-J. 1994. Dynamic linear models with Markov-switching. *Journal of Econometrics* 60: 1–22. https://doi.org/10.1016/0304-4076(94)90036-1.

Also see

[TS] **mswitch** — Markov-switching regression models

[U] 20 Estimation and postestimation commands