arfima postestimation — Postestimation tools for arfima

Postestimation commands predict margins Remarks and examples
Methods and formulas References Also see

Postestimation commands

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

Command	Description
`estat acplot`	estimate autocorrelations and autocovariances
`irf`	create and analyze IRFs
`psdensity`	estimate the spectral density

The following standard postestimation commands are also available:

Command	Description
`contrast`	contrasts and ANOVA-style joint tests of estimates
* `estat ic`	Akaike's, consistent Akaike's, corrected Akaike's, and Schwarz's Bayesian information criteria (AIC, CAIC, AICc, and BIC)
`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 coefficients
`lrtest`	likelihood-ratio test
* `margins`	marginal means, predictive margins, marginal effects, and average marginal effects
* `marginsplot`	graph the results from margins (profile plots, interaction plots, etc.)
* `nlcom`	point estimates, standard errors, testing, and inference for nonlinear combinations of coefficients
`predict`	linear predictions, innovations, standardized innovations, etc.
* `predictnl`	point estimates, standard errors, testing, and inference for generalized predictions
`pwcompare`	pairwise comparisons of estimates
`test`	Wald tests of simple and composite linear hypotheses
`testnl`	Wald tests of nonlinear hypotheses

* `estat ic`, `margins`, `marginsplot`, `nlcom`, and `predictnl` are not appropriate after `arfima, mpl`.

predict

Description for predict

`predict` creates a new variable containing predictions such as expected values, fractionally differenced series, and innovations. All predictions are available as static one-step-ahead predictions, and the dependent variable is also available as a dynamic multistep prediction.

Menu for predict

Statistics > Postestimation

Syntax for predict

`predict` [*type*] *newvar* [*if*] [*in*] [, *statistic options*]

statistic	Description
Main	
`xb`	predicted values; the default
`residuals`	predicted innovations
`rstandard`	standardized innovations
`fdifference`	fractionally differenced series

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	
`rmse(`[*type*] *newvar*`)`	put the estimated root mean squared error of the predicted statistic in a new variable; only permitted with options `xb` and `residuals`
`dynamic(`*datetime*`)`	forecast the time series starting at *datetime*; only permitted with option `xb`

datetime is a # or a time literal, such as `td(1jan1995)` or `tq(1995q1)`; see [D] **Datetime**.

Options for predict

Main

`xb`, the default, calculates the predictions for the level of *depvar*.

`residuals` calculates the predicted innovations.

`rstandard` calculates the standardized innovations.

`fdifference` calculates the fractionally differenced predictions of *depvar*.

Options

`rmse(`[*type*] *newvar*`)` 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. `rmse()` is only permitted with the `xb` and `residuals` options.

`dynamic(`*datetime*`)` specifies when `predict` starts producing dynamic forecasts. The specified *datetime* must be in the scale of the time variable specified in `tsset`, and the *datetime* must be inside a sample for which observations on the dependent variables are available. For example, `dynamic(tq(2008q4))` causes dynamic predictions to begin in the fourth quarter of 2008, assuming that your time variable is quarterly; see [D] **Datetime**. If the model contains exogenous variables, they must be present for the whole predicted sample. `dynamic()` may only be specified with `xb`.

margins

Description for margins

`margins` estimates margins of response for expected values.

Menu for margins

Statistics > Postestimation

Syntax for margins

`margins` [*marginlist*] [, *options*]

`margins` [*marginlist*] , `predict(`*statistic* ...`)` [*options*]

statistic	Description
`xb`	predicted values; the default
`residuals`	not allowed with `margins`
`rstandard`	not allowed with `margins`
`fdifference`	not allowed with `margins`

Statistics not allowed with `margins` are functions of stochastic quantities other than `e(b)`.

For the full syntax, see [R] **margins**.

Remarks and examples

stata.com

Remarks are presented under the following headings:

Forecasting after ARFIMA
IRF results for ARFIMA

Forecasting after ARFIMA

We assume that you have already read [TS] **arfima**. In this section, we illustrate some of the features of `predict` after fitting an ARFIMA model using `arfima`.

▷ Example 1

We have monthly data on the one-year Treasury bill secondary market rate imported from the Federal Reserve Bank (FRED) database using `import fred`; see [D] **import fred**. Below we fit an ARFIMA model with two autoregressive terms and one moving-average term to the data.

```
. use https://www.stata-press.com/data/r18/tb1yr
(FRED, 1-year Treasury bill; secondary market rate, monthly 1959-2001)

. arfima tb1yr, ar(1/2) ma(1)
Iteration 0:  Log likelihood = -235.31856
Iteration 1:  Log likelihood = -235.26104  (backed up)
Iteration 2:  Log likelihood = -235.25974  (backed up)
Iteration 3:  Log likelihood =  -235.2544  (backed up)
Iteration 4:  Log likelihood = -235.13355
Iteration 5:  Log likelihood = -235.13064
Iteration 6:  Log likelihood = -235.12108
Iteration 7:  Log likelihood = -235.11917
Iteration 8:  Log likelihood = -235.11869
Iteration 9:  Log likelihood = -235.11868
Refining estimates:
Iteration 0:  Log likelihood = -235.11868
Iteration 1:  Log likelihood = -235.11868

ARFIMA regression

Sample: 1959m7 thru 2001m8                      Number of obs =     506
                                                Wald chi2(4)  = 1864.16
Log likelihood = -235.11868                     Prob > chi2   =  0.0000
```

tb1yr	Coefficient	Std. err.	z	P>|z|	[95% conf. interval]	
tb1yr						
_cons	5.496708	2.92038	1.88	0.060	-.2271321	11.22055
ARFIMA						
ar						
L1.	.2326101	.1136625	2.05	0.041	.0098357	.4553845
L2.	.3885209	.083565	4.65	0.000	.2247365	.5523054
ma						
L1.	.7755849	.0669559	11.58	0.000	.6443537	.906816
d	.4606495	.0646499	7.13	0.000	.3339381	.5873609
/sigma2	.1466495	.009232	15.88	0.000	.1285551	.1647438

```
Note: The test of the variance against zero is one sided, and the two-sided
      confidence interval is truncated at zero.
```

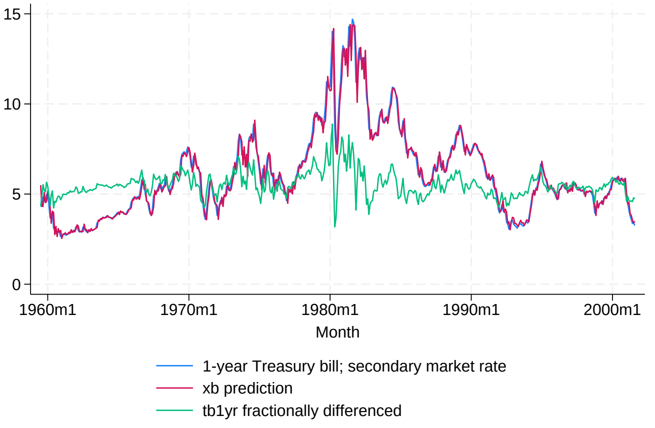
All the parameters are statistically significant at the 5% level, and they indicate a high degree of dependence in the series. In fact, the confidence interval for the fractional-difference parameter d indicates that the series may be nonstationary. We will proceed as if the series is stationary and suppose that it is fractionally integrated of order 0.46.

We begin our postestimation analysis by predicting the series in sample:

```
. predict ptb
(option xb assumed)
```

We continue by using the estimated fractional-difference parameter to fractionally difference the original series and by plotting the original series, the predicted series, and the fractionally differenced series. See [TS] **arfima** for a definition of the fractional-difference operator.

```
. predict fdtb, fdifference
. tsline tb1yr ptb fdtb, legend(cols(1))
```



The above graph shows that the in-sample predictions appear to track the original series well and that the fractionally differenced series looks much more like a stationary series than does the original.

◁

▷ Example 2

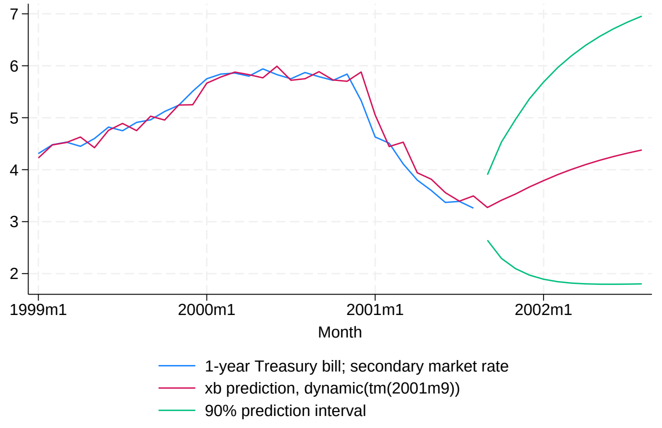
In this example, we use the above estimates to produce a dynamic forecast and a confidence interval for the forecast for the one-year treasury bill rate and plot them.

We begin by extending the dataset and using `predict` to put the dynamic forecast in the new `ftb` variable and the root mean squared error of the forecast in the new `rtb` variable. (As discussed in *Methods and formulas*, the root mean squared error of the forecast accounts for the idiosyncratic error but not for the estimation error.)

```
. tsappend, add(12)
. predict ftb, xb dynamic(tm(2001m9)) rmse(rtb)
```

Now we compute a 90% confidence interval around the dynamic forecast and plot the original series, the in-sample forecast, the dynamic forecast, and the confidence interval of the dynamic forecast.

```
. scalar z = invnormal(0.95)
. generate lb = ftb - z*rtb if month>=tm(2001m9)
(506 missing values generated)
. generate ub = ftb + z*rtb if month>=tm(2001m9)
(506 missing values generated)
. tsline tb1yr ftb if month>tm(1998m12) ||
> tsrline lb ub if month>=tm(2001m9),
> legend(cols(1) label(3 "90% prediction interval"))
```



◁

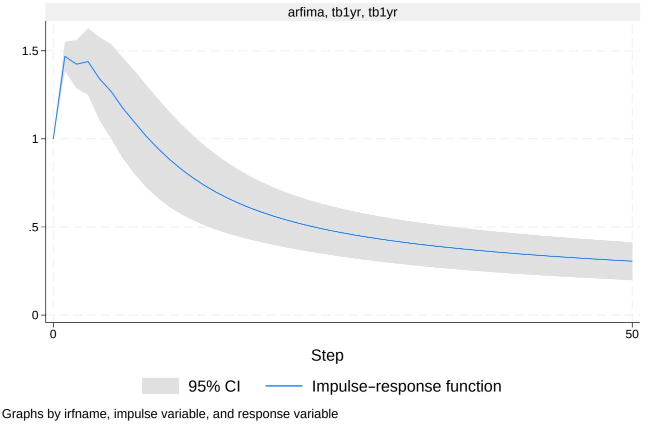
IRF results for ARFIMA

We assume that you have already read [TS] **irf** and [TS] **irf create**. In this section, we illustrate how to calculate the impulse–response function (IRF) of an ARFIMA model.

▷ Example 3

Here we use the estimates obtained in example 1 to calculate the IRF of the ARFIMA model; see [TS] **irf** and [TS] **irf create** for more details about IRFs.

```
. irf create arfima, step(50) set(myirf)
(file myirf.irf created)
(file myirf.irf now active)
(file myirf.irf updated)
. irf graph irf
```



The figure shows that a shock to `tb1yr` causes an initial spike in `tb1yr`, after which the impact of the shock starts decaying slowly. This behavior is characteristic of long-memory processes.

◁

Methods and formulas

Denote γ_h, $h = 1, \ldots, t$, to be the autocovariance function of the ARFIMA(p, d, q) process for two observations, y_t and y_{t-h}, h time periods apart. The covariance matrix $\mathbf{V}$ of the process of length T has a Toeplitz structure of

$$\mathbf{V} = \begin{pmatrix} \gamma_0 & \gamma_1 & \gamma_2 & \ldots & \gamma_{T-1} \\ \gamma_1 & \gamma_0 & \gamma_1 & \ldots & \gamma_{T-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \gamma_{T-1} & \gamma_{T-2} & \gamma_{T-3} & \ldots & \gamma_0 \end{pmatrix}$$

where the process variance is $\gamma_0 = \mathrm{Var}(y_t)$. We factor $\mathbf{V} = \mathbf{LDL}'$, where $\mathbf{L}$ is lower triangular and $\mathbf{D} = \mathrm{Diag}(\nu_t)$. The structure of $\mathbf{L}^{-1}$ is of importance.

$$\mathbf{L}^{-1} = \begin{pmatrix} 1 & 0 & 0 & \ldots & 0 & 0 \\ -\tau_{1,1} & 1 & 0 & \ldots & 0 & 0 \\ -\tau_{2,2} & -\tau_{2,1} & 1 & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ -\tau_{T-1,T-1} & -\tau_{T-1,T-2} & -\tau_{T-1,T-2} & \ldots & -\tau_{T-1,1} & 1 \end{pmatrix}$$

Let $z_t = y_t - \mathbf{x}_t\boldsymbol{\beta}$. The best linear predictor of z_{t+1} based on $z_1, z_2, \ldots, z_t$ is $\widehat{z}_{t+1} = \sum_{k=1}^{t} \tau_{t,k} z_{t-k+1}$. Define $-\boldsymbol{\tau}_t = (-\tau_{t,t}, -\tau_{t,t-1}, \ldots, -\tau_{t-1,1})$ to be the tth row of $\mathbf{L}^{-1}$ up to, but not including, the diagonal. Then $\boldsymbol{\tau}_t = \mathbf{V}_t^{-1}\boldsymbol{\gamma}_t$, where $\mathbf{V}_t$ is the $t \times t$ upper left submatrix of $\mathbf{V}$ and $\boldsymbol{\gamma}_t = (\gamma_1, \gamma_2, \ldots, \gamma_t)'$. Hence, the best linear predictor of the innovations is computed as $\widehat{\boldsymbol{\epsilon}} = \mathbf{L}^{-1}\mathbf{z}$, and the one-step predictions are $\widehat{\mathbf{y}} = \widehat{\boldsymbol{\epsilon}} + \mathbf{X}\widehat{\boldsymbol{\beta}}$. In practice, the computation is

$$\widehat{\mathbf{y}} = \widehat{\mathbf{L}}^{-1}\left(\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}}\right) + \mathbf{X}\widehat{\boldsymbol{\beta}}$$

where $\widehat{\mathbf{L}}$ and $\widehat{\mathbf{V}}$ are computed from the maximum likelihood estimates. We use the Durbin–Levinson algorithm (Palma 2007; Golub and Van Loan 2013) to factor $\widehat{\mathbf{V}}$, invert $\widehat{\mathbf{L}}$, and scale $\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}}$ using only the vector of estimated autocovariances $\widehat{\boldsymbol{\gamma}}$.

The prediction error variances of the one-step predictions are computed recursively in the Durbin–Levinson algorithm. They are the ν_t elements in the diagonal matrix $\mathbf{D}$ computed from the Cholesky factorization of $\mathbf{V}$. The recursive formula is $\nu_0 = \gamma_0$, and $\nu_t = \nu_{t-1}(1 - \tau_{t,t}^2)$.

Forecasting is carried out as described by Beran (1994, sec. 8.7), $\widehat{\mathbf{z}}_{T+k} = \widetilde{\gamma}_k' \widehat{\mathbf{V}}^{-1}\widehat{\mathbf{z}}$, where $\widetilde{\gamma}_k' = (\widehat{\gamma}_{T+k-1}, \widehat{\gamma}_{T+k-2}, \dots, \widehat{\gamma}_k)$. The forecast mean squared error is computed as $\text{MSE}(\widehat{\mathbf{z}}_{T+k}) = \widehat{\gamma}_0 - \widetilde{\gamma}_k' \widehat{\mathbf{V}}^{-1}\widetilde{\gamma}_k$. Computation of $\widehat{\mathbf{V}}^{-1}\widetilde{\gamma}_k$ is carried out efficiently using algorithm 4.7.2 of Golub and Van Loan (2013).

References

Beran, J. 1994. *Statistics for Long-Memory Processes*. Boca Raton, FL: Chapman and Hall/CRC.

Golub, G. H., and C. F. Van Loan. 2013. *Matrix Computations*. 4th ed. Baltimore: Johns Hopkins University Press.

Palma, W. 2007. *Long-Memory Time Series: Theory and Methods*. Hoboken, NJ: Wiley.

Also see

[TS] **arfima** — Autoregressive fractionally integrated moving-average models

[TS] **estat acplot** — Plot parametric autocorrelation and autocovariance functions

[TS] **irf** — Create and analyze IRFs, dynamic-multiplier functions, and FEVDs

[TS] **psdensity** — Parametric spectral density estimation after arima, arfima, and ucm

[U] 20 Estimation and postestimation commands