arfima — Autoregressive fractionally integrated moving-average models

Description

`arfima` estimates the parameters of autoregressive fractionally integrated moving-average (ARFIMA) models.

Long-memory processes are stationary processes whose autocorrelation functions decay more slowly than short-memory processes. The ARFIMA model provides a parsimonious parameterization of long-memory processes that nests the autoregressive moving-average (ARMA) model, which is widely used for short-memory processes. By allowing for fractional degrees of integration, the ARFIMA model also generalizes the autoregressive integrated moving-average (ARIMA) model with integer degrees of integration. See [TS] **arima** for ARMA and ARIMA parameter estimation.

Quick start

Autoregressive fractionally integrated moving-average model for y with regressor x using `tsset` data

```
arfima y x
```

Add autoregressive components of orders 1 and 2 and a moving-average component of order 4

```
arfima y x, ar(1 2) ma(4)
```

ARIMA for y with autoregressive components of orders 1 and 2

```
arfima y, ar(1 2) smemory
```

Menu

Statistics > Time series > ARFIMA > ARFIMA models

Syntax

```
arfima depvar [indepvars] [if] [in] [, options]
```

options	Description
Model	
noconstant	suppress constant term
ar(*numlist*)	autoregressive terms
ma(*numlist*)	moving-average terms
smemory	estimate short-memory model without fractional integration
mle	maximum likelihood estimates; the default
mpl	maximum modified-profile-likelihood estimates
constraints(*numlist*)	apply specified linear constraints
SE/Robust	
vce(*vcetype*)	*vcetype* may be oim or robust
Reporting	
level(#)	set confidence level; default is level(95)
nocnsreport	do not display constraints
display_options	control columns and column formats, row spacing, line width, display of omitted variables and base and empty cells, and factor-variable labeling
Maximization	
maximize_options	control the maximization process; seldom used
collinear	keep collinear variables
coeflegend	display legend instead of statistics

You must tsset your data before using arfima; see [TS] **tsset**.
indepvars may contain factor variables; see **[U] 11.4.3 Factor variables**.
depvar and *indepvars* may contain time-series operators; see **[U] 11.4.4 Time-series varlists**.
by, collect, fp, rolling, and statsby are allowed; see **[U] 11.1.10 Prefix commands**.
collinear and coeflegend do not appear in the dialog box.
See **[U] 20 Estimation and postestimation commands** for more capabilities of estimation commands.

Options

Model

noconstant; see [R] **Estimation options**.

ar(*numlist*) specifies the autoregressive (AR) terms to be included in the model. An AR(p), $p \geq 1$, specification would be ar(1/p). This model includes all lags from 1 to p, but not all lags need to be included. For example, the specification ar(1 p) would specify an AR(p) with only lags 1 and p included, setting all the other AR lag parameters to 0.

ma(*numlist*) specifies the moving-average terms to be included in the model. These are the terms for the lagged innovations (white-noise disturbances). ma(1/q), $q \geq 1$, specifies an MA(q) model, but like the ar() option, not all lags need to be included.

`smemory` causes `arfima` to fit a short-memory model with $d = 0$. This option causes `arfima` to estimate the parameters of an ARMA model by a method that is asymptotically equivalent to that produced by `arima`; see [TS] **arima**.

`mle` causes `arfima` to estimate the parameters by maximum likelihood. This method is the default.

`mpl` causes `arfima` to estimate the parameters by maximum modified profile likelihood (MPL). The MPL estimator of the fractional-difference parameter has less small-sample bias than the maximum likelihood estimator when there are covariates in the model. `mpl` may only be specified when there is a constant term or *indepvars* in the model, and it may not be combined with the `mle` option.

`constraints(`*numlist*`)`; see [R] **Estimation options**.

SE/Robust

`vce(`*vcetype*`)` specifies the type of standard error reported, which includes types that are robust to some kinds of misspecification (`robust`) and that are derived from asymptotic theory (`oim`); see [R] ***vce_option***.

Options `vce(robust)` and `mpl` may not be combined.

Reporting

`level(#)`, `nocnsreport`; see [R] **Estimation options**.

display_options: `noci`, `nopvalues`, `noomitted`, `vsquish`, `noemptycells`, `baselevels`, `allbaselevels`, `nofvlabel`, `fvwrap(#)`, `fvwrapon(`*style*`)`, `cformat(%`*fmt*`)`, `pformat(%`*fmt*`)`, `sformat(%`*fmt*`)`, and `nolstretch`; see [R] **Estimation options**.

Maximization

maximize_options: `difficult`, `technique(`*algorithm_spec*`)`, `iterate(#)`, `[no]log`, `trace`, `gradient`, `showstep`, `hessian`, `showtolerance`, `tolerance(#)`, `ltolerance(#)`, `nrtolerance(#)`, `gtolerance(#)`, `nonrtolerance(#)`, and `from(`*init_specs*`)`; see [R] **Maximize** for all options.

Some special points for `arfima`'s *maximize_options* are listed below.

`technique(`*algorithm_spec*`)` sets the optimization algorithm. The default algorithm is `BFGS` and `BHHH` is not allowed. See [R] **Maximize** for a description of the available optimization algorithms.

You can specify multiple optimization methods. For example, `technique(bfgs 10 nr)` requests that the optimizer perform 10 BFGS iterations and then switch to Newton–Raphson until convergence.

`iterate(#)` sets the maximum number of iterations. When the maximization is not going well, set the maximum number of iterations to the point where the optimizer appears to be stuck and inspect the estimation results at that point.

`from(`*matname*`)` allows you to specify starting values for the model parameters in a row vector. We recommend that you use the `iterate(0)` option, retrieve the initial estimates from `e(b)`, and modify these elements.

The following options are available with `arfima` but are not shown in the dialog box:

`collinear`, `coeflegend`; see [R] **Estimation options**.

Remarks and examples

Long-memory processes are stationary processes whose autocorrelation functions decay more slowly than short-memory processes. Because the autocorrelations die out so slowly, long-memory processes display a type of long-run dependence. The autoregressive fractionally integrated moving-average (ARFIMA) model provides a parsimonious parameterization of long-memory processes. This parameterization nests the autoregressive moving-average (ARMA) model, which is widely used for short-memory processes.

The ARFIMA model also generalizes the autoregressive integrated moving-average (ARIMA) model with integer degrees of integration. ARFIMA models provide a solution for the tendency to overdifference stationary series that exhibit long-run dependence. In the ARIMA approach, a nonstationary time series is differenced d times until the differenced series is stationary, where d is an integer. Such series are said to be integrated of order d, denoted $I(d)$, with not differencing, $I(0)$, being the option for stationary series. Many series exhibit too much dependence to be $I(0)$ but are not $I(1)$, and ARFIMA models are designed to represent these series.

The ARFIMA model allows for a continuum of fractional differences, $-0.5 < d < 0.5$. The generalization to fractional differences allows the ARFIMA model to handle processes that are neither $I(0)$ nor $I(1)$, to test for overdifferencing, and to model long-run effects that only die out at long horizons.

❑ Technical note

An ARIMA model for the series y_t is given by

$$\rho(L)(1-L)^d y_t = \theta(L)\epsilon_t \tag{1}$$

where $\rho(L) = (1 - \rho_1 L - \rho_2 L^2 - \cdots - \rho_p L^p)$ is the autoregressive (AR) polynomial in the lag operator L; $Ly_t = y_{t-1}$; $\theta(L) = (1 + \theta_1 L + \theta_2 L^2 + \cdots + \theta_p L^p)$ is the moving-average (MA) lag polynomial; ϵ_t is the independent and identically distributed innovation term; and d is the integer number of differences required to make the y_t stationary. An ARFIMA model is also specified by (1) with the generalization that $-0.5 < d < 0.5$. Series with $d \geq 0.5$ are handled by differencing and subsequent ARFIMA modeling. ❑

Because long-memory processes are stationary, one might be tempted to approximate the processes with many terms in an ARMA model. But these approximate models are difficult to fit and to interpret because ARMA models with many terms are difficult to estimate and the ARMA parameterization has an inherent short-run nature. In contrast, the ARFIMA model has the d parameter for the long-run dependence and ARMA parameters for short-run dependence. Using different parameters for different types of dependence facilitates estimation and interpretation, as discussed by Sowell (1992a).

❑ Technical note

An ARFIMA model specifies a fractionally integrated ARMA process. Formally, the ARFIMA model specifies that

$$y_t = (1-L)^{-d}\{\rho(L)\}^{-1}\theta(L)\epsilon_t$$

The short-run ARMA process $\rho(L)^{-1}\theta(L)\epsilon_t$ captures the short-run effects, and the long-run effects are captured by fractionally integrating the short-run ARMA process.

Essentially, the fractional-integration parameter d captures the long-run effects, and the ARMA parameters capture the short-run effects. Having separate parameters for short-run and long-run effects makes the ARFIMA model more flexible and easier to interpret than the ARMA model. After estimating the ARFIMA parameters, the short-run effects are obtained by setting $d = 0$, whereas the long-run effects use the estimated value for d. The short-run effects describe the behavior of the fractionally differenced process $(1 - L)^d y_t$, whereas the long-run effects describe the behavior of the fractionally integrated y_t. ❑

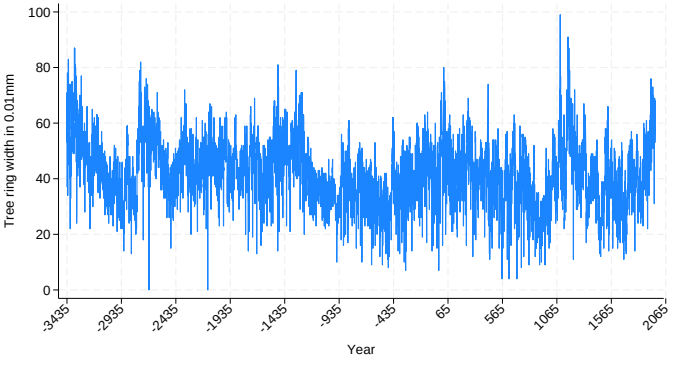
ARFIMA models have been useful in fields as diverse as hydrology and economics. Long-memory processes were first introduced in hydrology by Hurst (1951). Hosking (1981), in hydrology, and Granger and Joyeux (1980), in economics, independently discovered the ARFIMA representation of long-memory processes. Beran (1994), Baillie (1996), and Palma (2007) provide good introductions to long-memory processes and ARFIMA models.

Example 1: Mount Campito tree ring data

Baillie (1996) discusses a time series of measurements of the widths of the annual rings of a Mount Campito Bristlecone pine. The series contains measurements on rings formed in the tree from 3436 BC to 1969 AD. Essentially, larger widths were good years for the tree and narrower widths were harsh years.

We begin by plotting the time series.

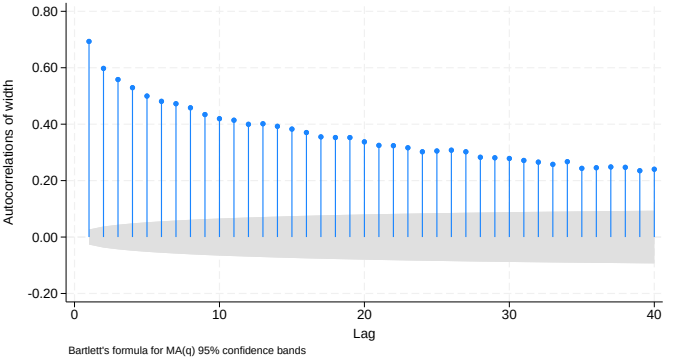
```
. use https://www.stata-press.com/data/r19/campito
(Campito Mnt. tree ring data from 3435BC to 1969AD)
. tsline width, tlabel(-3435(500)1969, angle(45)) ysize(2)
```



Good years and bad years seem to run together, causing the appearance of local trends. The local trends are evidence of dependence, but they are not as pronounced as those in a nonstationary series.

We plot the autocorrelations for another view:

```
. ac width, ysize(2)
```



The autocorrelations do not start below 1 but decay very slowly.

Granger and Joyeux (1980) show that the autocorrelations from an ARMA model decay exponentially, whereas the autocorrelations from an ARFIMA process decay at the much slower hyperbolic rate. Box et al. (2016) define short-memory processes as those whose autocorrelations decay exponentially fast and long-memory processes as those whose autocorrelations decay at the hyperbolic rate. The above plot of autocorrelations looks closer to hyperbolic than exponential.

Together, the above plots make us suspect that the series was generated by a long-memory process. We see evidence that the series is stationary but that the autocorrelations die out much slower than a short-memory process would predict.

Given that we believe the data were generated by a stationary process, we begin by fitting the data to an ARMA model. We begin by using a short-memory model because a comparison of the results highlights the advantages of using an ARFIMA model for a long-memory process.

```
. arima width, ar(1/2) ma(1) technique(bhhh 4 nr)
(setting optimization to BHHH)
Iteration 0:  Log likelihood = -18934.593
Iteration 1:  Log likelihood = -18914.337
Iteration 2:  Log likelihood = -18913.407
Iteration 3:  Log likelihood =  -18913.24
(switching optimization to Newton-Raphson)
Iteration 4:  Log likelihood = -18913.214
Iteration 5:  Log likelihood = -18913.208
Iteration 6:  Log likelihood = -18913.208
ARIMA regression
Sample: -3435 thru 1969                         Number of obs     =       5405
                                                Wald chi2(3)      =  133686.46
Log likelihood = -18913.21                      Prob > chi2       =     0.0000
```

width	Coefficient	OIM std. err.	z	P>|z|	[95% conf.	interval]
width						
_cons	42.45055	1.02142	41.56	0.000	40.44861	44.4525
ARMA						
ar						
L1.	1.264367	.0253199	49.94	0.000	1.214741	1.313993
L2.	-.2848827	.0227534	-12.52	0.000	-.3294785	-.240287
ma						
L1.	-.8066007	.0189699	-42.52	0.000	-.8437811	-.7694204
/sigma	8.005814	.0770004	103.97	0.000	7.854896	8.156732

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

The estimated coefficients seem high in magnitude. We use `estat aroots` to investigate further.

```
. estat aroots
```

Eigenvalue stability condition

Eigenvalue	Modulus
.9709661	.970966
.2934013	.293401

```
All the eigenvalues lie inside the unit circle.
AR parameters satisfy stability condition.
```

Eigenvalue stability condition

Eigenvalue	Modulus
.8066007	.806601

```
All the eigenvalues lie inside the unit circle.
MA parameters satisfy invertibility condition.
```

The roots of the AR polynomial are 0.971 and 0.293, and the root of the MA polynomial is 0.807; all of these are less than one in magnitude, indicating that the series is stationary and invertible but has a high level of persistence. See Hamilton (1994, 59) and [TS] **estat aroots** for details about computing and interpreting the roots of the polynomials from the estimated ARIMA coefficients.

Below we estimate the parameters of an ARFIMA model with only the fractional difference parameter and a constant.

```
. arfima width
Iteration 0:  Log likelihood = -18918.219
Iteration 1:  Log likelihood =  -18916.84
Iteration 2:  Log likelihood = -18908.508
Iteration 3:  Log likelihood = -18908.508  (backed up)
Iteration 4:  Log likelihood = -18907.302
Iteration 5:  Log likelihood = -18907.293
Iteration 6:  Log likelihood = -18907.279
Iteration 7:  Log likelihood = -18907.279
Refining estimates:
Iteration 0:  Log likelihood = -18907.279
Iteration 1:  Log likelihood = -18907.279

ARFIMA regression

Sample: -3435 thru 1969                         Number of obs =     5,405
                                                Wald chi2(1)  =   1864.43
Log likelihood = -18907.279                     Prob > chi2   =    0.0000
```

width	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
width						
_cons	44.01432	9.174318	4.80	0.000	26.03299	61.99565
ARFIMA						
d	.4468888	.0103497	43.18	0.000	.4266038	.4671737
/sigma2	63.92927	1.229754	51.99	0.000	61.519	66.33955

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

The estimate of d is large and statistically significant. The relative parsimony of the ARFIMA model is illustrated by the fact that the estimates of the standard deviation of the idiosyncratic errors are about the same in the five-parameter ARMA model and the three-parameter ARFIMA model.

Let's add an AR parameter to the above ARFIMA model:

```
. arfima width, ar(1)
Iteration 0:  Log likelihood = -18910.997
Iteration 1:  Log likelihood = -18910.949  (backed up)
Iteration 2:  Log likelihood = -18908.158  (backed up)
Iteration 3:  Log likelihood = -18907.248
Iteration 4:  Log likelihood = -18907.233
Iteration 5:  Log likelihood = -18907.233
Iteration 6:  Log likelihood = -18907.233
Refining estimates:
Iteration 0:  Log likelihood = -18907.233
Iteration 1:  Log likelihood = -18907.233

ARFIMA regression
Sample: -3435 thru 1969                         Number of obs =   5,405
                                                Wald chi2(2)  = 1875.34
Log likelihood = -18907.233                     Prob > chi2   =  0.0000
```

width	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
width						
_cons	43.98774	8.685174	5.06	0.000	26.96511	61.01037
ARFIMA						
ar						
L1.	.0063325	.020983	0.30	0.763	-.0347933	.0474584
d	.443247	.0158858	27.90	0.000	.4121114	.4743826
/sigma2	63.92915	1.229755	51.99	0.000	61.51887	66.33942

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

That the estimated AR term is tiny and statistically insignificant indicates that the d parameter has accounted for all the dependence in the series.

◁

As mentioned above, there is a sense in which the main advantages of an ARFIMA model over an ARMA model for long-memory processes are the relative parsimony of the ARFIMA parameterization and the ability of the ARFIMA parameterization to separate out the long-run effects from the short-run effects. If the true process was generated from an ARFIMA model, an ARMA model with many terms can approximate the process, but the terms make estimation difficult and the lack of separate long-run and short-run parameters complicates interpretation.

This example highlights the relative parsimony of the ARFIMA model. In the examples below, we illustrate the advantages of having separate parameters for long-run and short-run effects.

❑ Technical note

You may be wondering what long-run effects can be produced by a model for stationary processes. Because the autocorrelations of a long-memory process die out so slowly, the spectral density becomes infinite as the frequency goes to 0 and the impulse–response functions die out at a much slower rate.

The spectral density of a process describes the relative contributions of random components at different frequencies to the variance of the process, with the low-frequency components corresponding to long-run effects. See [TS] **psdensity** for an introduction to estimating and interpreting spectral densities implied by the estimated parameters of parametric models.

Granger and Joyeux (1980) motivate ARFIMA models by noting that their implied spectral densities are finite except at frequency 0 with $0 < d < 0.5$, whereas stationary ARMA models have finite spectral densities at all frequencies. Granger and Joyeux (1980) argue that the ability of ARFIMA models to capture this long-range dependence, which cannot be captured by stationary ARMA models, is an important advantage of ARFIMA models over ARMA models when modeling long-memory processes.

Impulse–response functions are the coefficients on the infinite-order MA representation of a process, and they describe how a shock feeds though the dynamic system. If the process is stationary, the coefficients decay to 0 and they sum to a finite constant. As expected, the coefficients from an ARFIMA model die out at a slower rate than those from an ARMA model. Because the ARMA terms model the short-run effects and the d parameter models the long-run effects, an ARFIMA model specifies both a short-run impulse–response function and a long-run impulse–response function. When an ARMA model is used to approximate a long-memory model, the ARMA impulse–response-function coefficients confound the two effects.

❑

▷ Example 2

In this example, we model the log of the monthly levels of carbon dioxide above Mauna Loa, Hawaii. To remove the seasonality, we model the twelfth seasonal difference of the log of the series. This example illustrates that the ARFIMA model parameterizes long-run and short-run effects, whereas the ARMA model confounds the two effects. (Sowell [1992a] discusses this point in greater depth.)

We begin by fitting the series to an ARMA model with an AR(1) term and an MA(2).

```
. use https://www.stata-press.com/data/r19/mloa, clear
. arima S12.log, ar(1) ma(2)
(setting optimization to BHHH)
Iteration 0:  Log likelihood =  2000.9262
Iteration 1:  Log likelihood =  2001.5484
Iteration 2:  Log likelihood =  2001.5637
Iteration 3:  Log likelihood =  2001.5641
Iteration 4:  Log likelihood =  2001.5641
ARIMA regression
Sample: 1960m1 thru 1990m12                     Number of obs     =        372
                                                Wald chi2(2)      =     500.41
Log likelihood = 2001.564                       Prob > chi2       =     0.0000
```

S12.log	Coefficient	OPG std. err.	z	P>|z|	[95% conf.	interval]
log						
_cons	.0036754	.0002475	14.85	0.000	.0031903	.0041605
ARMA						
ar						
L1.	.7354346	.0357715	20.56	0.000	.6653237	.8055456
ma						
L2.	.1353086	.0513156	2.64	0.008	.0347319	.2358853
/sigma	.0011129	.0000401	27.77	0.000	.0010344	.0011914

```
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, and they indicate a high degree of dependence.

Below we nest the previously fit ARMA model into an ARFIMA model.

```
. arfima S12.log, ar(1) ma(2)
Iteration 0:  Log likelihood =  2006.0757
Iteration 1:  Log likelihood =  2006.0774  (backed up)
Iteration 2:  Log likelihood =  2006.0775  (backed up)
Iteration 3:  Log likelihood =  2006.0804
Iteration 4:  Log likelihood =  2006.0805
Refining estimates:
Iteration 0:  Log likelihood =  2006.0805
Iteration 1:  Log likelihood =  2006.0805
ARFIMA regression
Sample: 1960m1 thru 1990m12                         Number of obs =    372
                                                    Wald chi2(3)  = 248.88
Log likelihood = 2006.0805                          Prob > chi2   = 0.0000
```

S12.log	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
S12.log						
_cons	.003616	.0012968	2.79	0.005	.0010743	.0061578
ARFIMA						
ar						
L1.	.2160894	.1015578	2.13	0.033	.0170397	.415139
ma						
L2.	.1633916	.0516909	3.16	0.002	.0620793	.2647038
d	.4042573	.0805435	5.02	0.000	.2463949	.5621197
/sigma2	1.20e-06	8.84e-08	13.63	0.000	1.03e-06	1.38e-06

```
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. That the confidence interval for the fractional-difference parameter d includes numbers greater than 0.5 is evidence that the series may be nonstationary. Alternatively, we proceed as if the series is stationary, and the wide confidence interval for d reflects the difficulty of fitting a complicated dynamic model with only 372 observations.

With the above caveat, we can now proceed to compare the interpretations of the ARMA and ARFIMA estimates. We compare these estimates in terms of their implied spectral densities. The spectral density of a stationary time series describes the relative importance of components at different frequencies. See [TS] **psdensity** for an introduction to spectral densities.

Below we quietly refit the ARMA model and use `psdensity` to estimate the parametric spectral density implied by the ARMA parameter estimates.

```
. quietly arima S12.log, ar(1) ma(2)
. psdensity d_arma omega1
```

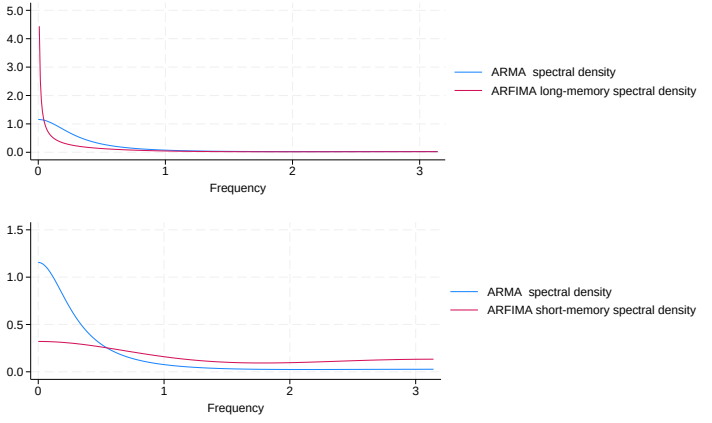
The `psdensity` command above put the estimated ARMA spectral density into the new variable `d_arma` at the frequencies stored in the new variable `omega1`.

Below we quietly refit the ARFIMA model and use `psdensity` to estimate the long-run parametric spectral density and then the short-run parametric spectral density implied by the ARFIMA parameter estimates. The long-run estimates use the estimated d, and the short-run estimates set d to 0 (as is implied by specifying the `smemory` option). The long-run estimates describe the fractionally integrated series, and the short-run estimates describe the fractionally differenced series.

```
. quietly arfima S12.log, ar(1) ma(2)
. psdensity d_arfima omega2
. psdensity ds_arfima omega3, smemory
```

Now that we have the ARMA estimates, the long-run ARFIMA estimates, and the short-run ARFIMA estimates, we graph them below.

```
. line d_arma d_arfima omega1, ylabel(, format(%3.1f)) name(lmem) nodraw
. line d_arma ds_arfima omega1, ylabel(, format(%3.1f)) name(smem) nodraw
. graph combine lmem smem, cols(1) xcommon
```



The top graph contains a plot of the spectral densities implied by the ARMA parameter estimates and by the long-run ARFIMA parameter estimates. As discussed by Granger and Joyeux (1980), the two models imply different spectral densities for frequencies close to 0 when $d > 0$. When $d > 0$, the spectral density implied by the ARFIMA estimates diverges to infinity, whereas the spectral density implied by the ARMA estimates remains finite at frequency 0 for stable ARMA processes. This difference reflects the ability of ARFIMA models to capture long-run effects that ARMA models only capture as the parameters approach those of an unstable model.

The bottom graph contains a plot of the spectral densities implied by the ARMA parameter estimates and by the short-run ARFIMA parameter estimates, which are the ARMA parameters for the fractionally differenced process. Comparing the two plots illustrates the ability of the short-run ARFIMA parameters to capture both low-frequency and high-frequency components in the fractionally differenced series. In contrast, the ARMA parameters captured only low-frequency components in the fractionally integrated series.

Comparing the ARFIMA and ARMA spectral densities in the two graphs illustrates that the additional fractional-difference parameter allows the ARFIMA model to identify both long-run and short-run effects, which the ARMA model confounds.

◁

❑ Technical note

As noted above, the spectral density of an ARFIMA process with $d > 0$ diverges to infinity as the frequency goes to 0. In contrast, the spectral density of an ARFIMA process with $d < 0$ is 0 at frequency 0.

The autocorrelation function of an ARFIMA process with $d < 0$ also decays at the slower hyperbolic rate. ARFIMA processes with $d < 0$ are sometimes called antipersistent because all the autocorrelations for lags greater than 0 are negative.

Hosking (1981), Baillie (1996), and others refer to ARFIMA processes with $d < 0$ as "intermediate memory" processes and ARFIMA processes with $d > 0$ as long-memory processes. Box, Jenkins, Reinsel, and Ljung (2016, 385) define long-memory processes as those with the slower hyperbolic rate of decay, which includes ARFIMA processes with $d < 0$. We follow Box et al. (2016) and thus call ARFIMA processes for $-0.5 < d < 0$ and $0 < d < 0.5$ long-memory processes.

Sowell (1992a) uses the properties of ARFIMA processes with $d < 0$ to derive tests for whether a series was generated by an $I(1)$ process or an $I(d)$ process with $d < 1$.

❑

▷ Example 3

In this example, we use `arfima` to test whether a series is nonstationary. More specifically, we test whether the series was generated by an $I(1)$ process by testing whether the first difference of the series is overdifferenced.

We have monthly data on the log of the number of reported cases of mumps in New York City between January 1928 and December 1972. We believe that the series is stationary, after accounting for the monthly seasonal effects. We use an ARFIMA model for differenced series to test the null hypothesis of nonstationarity. We use the confidence interval for the d parameter from an ARFIMA model for the first difference of the log of the series to perform the test. If the right-hand end of the 95% CI is less than 0, we conclude that the differenced series was overdifferenced, which implies that the original series was not nonstationary.

More formally, if y_t is $I(1)$, then $\Delta y_t = y_t - y_{t-1}$ must be $I(0)$. If Δy_t is $I(d)$ with $d < 0$, then Δy_t is overdifferenced and y_t is $I(d)$ with $d < 1$.

We use seasonal indicators to account for the seasonal effects. In the output below, we specify the `mpl` option to use the MPL estimator that is less biased in the presence of covariates.

`arfima` computes the maximum likelihood estimates (MLE) for the parameters of this stationary and invertible Gaussian process. Alternatively, the maximum MPL estimates may be computed. See *Methods and formulas* for a description of these two estimation techniques, but suffice it to say that the MLE estimates for d are biased in the presence of exogenous variables, even the constant term, for small samples. The MPL estimator reduces this bias; see Hauser (1999) and Doornik and Ooms (2004).

```
. use https://www.stata-press.com/data/r19/mumps2, clear
(Hipel and Mcleod (1994), http://robjhyndman.com/tsdldata/epi/mumps.dat)
. arfima D.log i.month, ma(1 2) mpl
Iteration 0:  Log modified profile likelihood =  53.766763
Iteration 1:  Log modified profile likelihood =  54.388641
Iteration 2:  Log modified profile likelihood =  54.934726  (backed up)
Iteration 3:  Log modified profile likelihood =  54.937524  (backed up)
Iteration 4:  Log modified profile likelihood =  55.002187
Iteration 5:  Log modified profile likelihood =   55.20462
Iteration 6:  Log modified profile likelihood =  55.205939
Iteration 7:  Log modified profile likelihood =  55.205949
Iteration 8:  Log modified profile likelihood =  55.205949
Refining estimates:
Iteration 0:  Log modified profile likelihood =  55.205949
Iteration 1:  Log modified profile likelihood =  55.205949
ARFIMA regression
Sample: 1928m2 thru 1972m6                      Number of obs =     533
                                                Wald chi2(14) = 1360.28
Log modified profile likelihood = 55.205949     Prob > chi2   =  0.0000
```

D.log	Coefficient	Std. err.	z	P>|z|	[95% conf.	interval]
D.log						
month						
February	-.220719	.0428112	-5.16	0.000	-.3046275	-.1368105
March	.0314683	.0424718	0.74	0.459	-.0517749	.1147115
April	-.2800296	.0460084	-6.09	0.000	-.3702043	-.1898548
May	-.3703179	.0449932	-8.23	0.000	-.4585029	-.2821329
June	-.4722035	.0446764	-10.57	0.000	-.5597676	-.3846394
July	-.9613239	.0448375	-21.44	0.000	-1.049204	-.873444
August	-1.063042	.0449272	-23.66	0.000	-1.151098	-.9749868
September	-.7577301	.0452529	-16.74	0.000	-.8464242	-.669036
October	-.3024251	.0462887	-6.53	0.000	-.3931494	-.2117009
November	-.0115317	.0426911	-0.27	0.787	-.0952046	.0721413
December	.0247135	.0430401	0.57	0.566	-.0596435	.1090705
_cons	.3656807	.0303215	12.06	0.000	.3062517	.4251096
ARFIMA						
ma						
L1.	.258056	.0684414	3.77	0.000	.1239133	.3921987
L2.	.1972011	.0506439	3.89	0.000	.0979409	.2964612
d	-.2329426	.067336	-3.46	0.001	-.3649188	-.1009663

We interpret the fact that the estimated 95% CI is strictly less than 0 to mean that the differenced series is overdifferenced, which implies that the original series is stationary.

◁

Stored results

arfima stores the following in e():

Scalars

e(N)	number of observations
e(k)	number of parameters
e(k_eq)	number of equations in e(b)
e(df_m)	model degrees of freedom
e(ll)	log likelihood
e(chi2)	χ^2
e(p)	p-value for model test
e(s2)	idiosyncratic error variance estimate, if e(method) = mpl
e(tmin)	minimum time
e(tmax)	maximum time
e(ar_max)	maximum AR lag
e(ma_max)	maximum MA lag
e(constant)	0 if noconstant, 1 otherwise
e(rank)	rank of e(V)
e(ic)	number of iterations
e(rc)	return code
e(converged)	1 if converged, 0 otherwise

Macros

e(cmd)	arfima
e(cmdline)	command as typed
e(depvar)	name of dependent variable
e(tvar)	time variable
e(covariates)	list of covariates
e(method)	mle or mpl
e(eqnames)	names of equations
e(title)	title in estimation output
e(tmins)	formatted minimum time
e(tmaxs)	formatted maximum time
e(chi2type)	Wald; type of model χ^2 test
e(vce)	*vcetype* specified in vce()
e(vcetype)	title used to label Std. err.
e(ma)	lags for MA terms
e(ar)	lags for AR terms
e(technique)	maximization technique
e(tech_steps)	number of iterations performed before switching techniques
e(properties)	b V
e(estat_cmd)	program used to implement estat
e(predict)	program used to implement predict
e(marginsok)	predictions allowed by margins
e(marginsnotok)	predictions disallowed by margins
e(asbalanced)	factor variables fvset as asbalanced
e(asobserved)	factor variables fvset as asobserved

Matrices

e(b)	coefficient vector
e(Cns)	constraints matrix
e(ilog)	iteration log (up to 20 iterations)
e(gradient)	gradient vector
e(V)	variance–covariance matrix of the estimators
e(V_modelbased)	model-based variance

Functions

e(sample)	marks estimation sample

In addition to the above, the following is stored in `r()`:

Matrices
`r(table)` matrix containing the coefficients with their standard errors, test statistics, p-values, and confidence intervals

Note that results stored in `r()` are updated when the command is replayed and will be replaced when any r-class command is run after the estimation command.

Methods and formulas

Methods and formulas are presented under the following headings:

Introduction

We model an observed second-order stationary time-series $y_t, t = 1, \ldots, T$, using the ARFIMA(p, d, q) model defined as

$$\boldsymbol{\rho}(L^p)(1-L)^d(y_t - \mathbf{x}_t\boldsymbol{\beta}) = \boldsymbol{\theta}(L^q)\epsilon_t$$

where

$$\begin{aligned}
\boldsymbol{\rho}(L^p) &= 1 - \rho_1 L - \rho_2 L^2 - \cdots - \rho_p L^p \\
\boldsymbol{\theta}(L^q) &= 1 + \theta_1 L + \theta_2 L^2 + \cdots + \theta_q L^q \\
(1-L)^d &= \sum_{j=0}^{\infty} (-1)^j \frac{\Gamma(j+d)}{\Gamma(j+1)\Gamma(d)} L^j
\end{aligned}$$

and the lag operator is defined as $L^j y_t = y_{t-j}, t = 1, \ldots, T$ and $j = 1, \ldots, t-1$; $\epsilon_t \sim N(0, \sigma^2)$; $\Gamma()$ is the gamma function; and $-0.5 < d < 0.5$, $d \neq 0$. The row vector $\mathbf{x}_t$ contains the exogenous variables specified as *indepvars* in the `arfima` syntax.

The process is stationary and invertible for $-0.5 < d < 0.5$; the roots of the AR polynomial, $\boldsymbol{\rho}(z) = 1 - \rho_1 z - \rho_2 z^2 - \cdots - \rho_p z^p = 0$, and the MA polynomial, $\boldsymbol{\theta}(z) = 1 + \theta_1 z + \theta_2 z^2 + \cdots + \theta_q z^q = 0$, lie outside the unit circle and there are no common roots. When $0 < d < 0.5$, the process has long memory in that the autocovariance function, γ_h, decays to 0 at a hyperbolic rate, such that $\sum_{h=-\infty}^{\infty} |\gamma_h| = \infty$. When $-0.5 < d < 0$, the process also has long memory in that the autocovariance function, γ_h, decays to 0 at a hyperbolic rate such that $\sum_{h=-\infty}^{\infty} |\gamma_h| < \infty$. (As discussed in the text, some authors refer to ARFIMA processes with $-0.5 < d < 0$ as having intermediate memory, but we follow Box et al. [2016] and refer to them as long-memory processes.)

Granger and Joyeux (1980), Hosking (1981), Sowell (1992b, 1992a), Baillie (1996), and Palma (2007) provide overviews of long-memory processes, fractional integration, and introductions to ARFIMA models.

The likelihood function

Estimation of the ARFIMA parameters $\boldsymbol{\rho}$, $\boldsymbol{\theta}$, d, $\boldsymbol{\beta}$ and σ^2 is done by the method of maximum likelihood. The log Gaussian likelihood of $\mathbf{y}$ given parameter estimates $\hat{\boldsymbol{\eta}} = (\hat{\boldsymbol{\rho}}', \hat{\boldsymbol{\theta}}', \hat{d}, \hat{\boldsymbol{\beta}}', \hat{\sigma}^2)$ is

$$\ell(\mathbf{y}|\hat{\boldsymbol{\eta}}) = -\frac{1}{2}\{T\log(2\pi) + \log|\widehat{\mathbf{V}}| + (\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}})'\widehat{\mathbf{V}}^{-1}(\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}})\} \tag{2}$$

where the covariance matrix $\mathbf{V}$ has a Toeplitz structure

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

$\mathrm{Var}(y_t) = \gamma_0$, $\mathrm{Cov}(y_t, y_{t-h}) = \gamma_h$ (for $h = 1, \ldots, t-1$), and $t = 1, \ldots, T$ (Sowell 1992b).

We use the Durbin–Levinson algorithm (Palma 2007; Golub and Van Loan 2013) to factor and invert $\mathbf{V}$. Using only the vector of autocovariances $\boldsymbol{\gamma}$, the Durbin–Levinson algorithm will compute $\hat{\boldsymbol{\epsilon}} = \widehat{\mathbf{D}}^{-0.5}\widehat{\mathbf{L}}^{-1}(\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}})$, where $\mathbf{L}$ is lower triangular and $\mathbf{V} = \mathbf{L}\mathbf{D}\mathbf{L}'$ and $\mathbf{D} = \mathrm{Diag}(\boldsymbol{\nu})$, $\nu_t = \mathrm{Var}(y_t)$. The algorithm performs these computations without generating the $T \times T$ matrix $\mathbf{L}^{-1}$.

During optimization, we restrict the fractional-integration parameter to $(-0.5, 0.5)$ using a logistic transform, $d^* = \log\{(x + 0.5)/(0.5 - x)\}$, so that the range of d^* encompasses the real line. During the "Refining estimates" step, the fractional-integration parameter is transformed back to the restricted space, where we obtain its standard error from the observed information matrix.

The autocovariance function

Computation of the autocovariances γ_h is given by Sowell (1992b) with numerical enhancements by Doornik and Ooms (2003) and is reviewed by Palma (2007, sec. 3.2.4). We reproduce it here. The autocovariance of an ARFIMA$(0, d, 0)$ process is

$$\gamma_h^* = \sigma^2 \frac{\Gamma(1-2d)}{\Gamma(1-d)\Gamma(d)} \frac{\Gamma(h+d)}{\Gamma(1+h-d)}$$

where $h = 0, 1, \ldots$. For ARFIMA(p, d, q), we have

$$\gamma_h = \sigma^2 \sum_{i=-q}^{q} \sum_{j=1}^{p} \psi(i)\xi_j C(d, p+i-h, \rho_j) \tag{3}$$

where

$$\psi(i) = \sum_{k=\max(0,i)}^{\min(q,q+i)} \theta_k \theta_{k-i}$$

$$\xi_j = \left\{ \rho_j \prod_{i=1}^{p} (1 - \rho_i \rho_j) \prod_{m \neq j} (\rho_j - \rho_m) \right\}^{-1}$$

and

$$C(d,h,\rho) = \frac{\gamma_h^*}{\sigma^2} \left\{ \rho^{2p} F(d+h, 1, 1-d+h, \rho) + F(d-h, 1, 1-d-h, \rho) - 1 \right\}$$

$F(\cdot)$ is the hypergeometric series (Zwillinger [Gradshteyn and Ryzhik] 2015)

$$F(a,b,c,x) = 1 + \frac{ab}{c \cdot 1} x + \frac{a(a+1)b(b+1)}{c(c+1) \cdot 1 \cdot 2} x^2 + \frac{a(a+1)(a+2)b(b+1)(b+2)}{c(c+1)(c+2) \cdot 1 \cdot 2 \cdot 3} x^3 + \cdots$$

The series recursions are evaluated backward as Doornik and Ooms (2003) emphasize. Doornik and Ooms (2003) also provide other computational enhancements, such as not dividing by ρ_j in (3).

The profile likelihood

Doornik and Ooms (2003) show that the parameters σ^2 and $\boldsymbol{\beta}$ can be concentrated out of the likelihood. Using (2), the MLE for σ^2 is

$$\hat{\sigma}^2 = \frac{1}{T} (\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}})' \widehat{\mathbf{R}}^{-1} (\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}}) \tag{4}$$

where $\mathbf{R} = \frac{1}{\sigma^2}\mathbf{V}$ and

$$\widehat{\boldsymbol{\beta}} = (\mathbf{X}'\hat{\mathbf{R}^{-1}}\mathbf{X})^{-1}\mathbf{X}'\hat{\mathbf{R}^{-1}}\mathbf{y} \tag{5}$$

is the weighted least-squares estimates for $\boldsymbol{\beta}$. Substituting (4) into (2) results in the profile likelihood

$$\ell_p(\mathbf{y}|\hat{\boldsymbol{\eta}}_r) = -\frac{T}{2} \left\{ 1 + \log(2\pi) + \frac{1}{T} \log|\widehat{\mathbf{R}}| + \log\hat{\sigma}^2 \right\}$$

We compute the MLEs using the profile likelihood for the reduced parameter set $\boldsymbol{\eta}_r = (\boldsymbol{\rho}', \boldsymbol{\theta}', d)$. Equations (4) and (5) provide MLEs for σ^2 and $\boldsymbol{\beta}$ to create the full parameter vector $\boldsymbol{\eta} = (\boldsymbol{\beta}', \boldsymbol{\rho}', \boldsymbol{\theta}', d, \sigma^2)$. We follow with the "Refining estimates" step, optimizing on the log likelihood (1). The refining step does not change the estimates; it produces the coefficient variance–covariance matrix from the observed information matrix.

Using this profile likelihood prevents the use of the BHHH optimization method because there are no observation-level scores.

The MPL

The small-sample MLE for d can be biased when there are exogenous variables in the model. The MPL reduces this bias (Hauser 1999; Doornik and Ooms 2004). The `mpl` option will direct `arfima` to use this optimization criterion. The MPL is expressed as

$$\ell_m(\mathbf{y}|\hat{\boldsymbol{\eta}}_r) = -\frac{T}{2} \{1 + \log(2\pi)\} - \left(\frac{1}{T} - \frac{1}{2}\right) \log|\widehat{\mathbf{R}}| - \left(\frac{T-k-2}{2}\right) \log\hat{\sigma}^2 - \frac{1}{2} \log|\mathbf{X}'\widehat{\mathbf{R}}^{-1}\mathbf{X}|$$

where $k = \text{rank}(\mathbf{X})$ (An and Bloomfield 1993).

There is no MPL estimator for σ^2, and you will notice its absence from the coefficient table. However, the unbiased estimate assuming ARFIMA$(0, 0, 0)$,

$$\tilde{\sigma}^2 = \frac{(\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}})' \widehat{\mathbf{R}}^{-1} (\mathbf{y} - \mathbf{X}\widehat{\boldsymbol{\beta}})}{T - k}$$

is stored in `e()` for postestimation computation of the forecast and residual root mean squared errors.

References

An, S., and P. Bloomfield. 1993. Cox and Reid's modification in regression models with correlated errors. Tech. rep., Department of Statistics, North Carolina State University, Raleigh, NC.

Baillie, R. T. 1996. Long memory processes and fractional integration in econometrics. *Journal of Econometrics* 73: 5–59. https://doi.org/10.1016/0304-4076(95)01732-1.

Baum, C. F., and S. Hurn. 2021. *Environmental Econometrics Using Stata*. College Station, TX: Stata Press.

Baum, C. F., S. Hurn, and K. Lindsay. 2020. Local Whittle estimation of the long-memory parameter. *Stata Journal* 20: 565–583.

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

Box, G. E. P., G. M. Jenkins, G. C. Reinsel, and G. M. Ljung. 2016. *Time Series Analysis: Forecasting and Control*. 5th ed. Hoboken, NJ: Wiley.

Doornik, J. A., and M. Ooms. 2003. Computational aspects of maximum likelihood estimation of autoregressive fractionally integrated moving average models. *Computational Statistics and Data Analysis* 42: 333–348. https://doi.org/10.1016/S0167-9473(02)00212-8.

———. 2004. Inference and forecasting for ARFIMA models with an application to US and UK inflation. *Studies in Nonlinear Dynamics and Econometrics* 8: art. 14. https://doi.org/10.2202/1558-3708.1218.

Golub, G. H., and C. F. Van Loan. 2013. *Matrix Computations*. 4th ed. Baltimore: Johns Hopkins University Press. https://doi.org/10.56021/9781421407944.

Granger, C. W. J., and R. Joyeux. 1980. An introduction to long-memory time series models and fractional differencing. *Journal of Time Series Analysis* 1: 15–29. https://doi.org/10.1111/j.1467-9892.1980.tb00297.x.

Hamilton, J. D. 1994. *Time Series Analysis*. Princeton, NJ: Princeton University Press. https://doi.org/10.2307/j.ctv14jx6sm.

Hauser, M. A. 1999. Maximum likelihood estimators for ARMA and ARFIMA models: A Monte Carlo study. *Journal of Statistical Planning and Inference* 80: 229–255. https://doi.org/10.1016/S0378-3758(98)00252-3.

Hosking, J. R. M. 1981. Fractional differencing. *Biometrika* 68: 165–176. https://doi.org/10.1093/biomet/68.1.165.

Hurst, H. E. 1951. Long-term storage capacity of reservoirs. *Transactions of the American Society of Civil Engineers* 116: 770–779.

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

Sowell, F. 1992a. Modeling long-run behavior with the fractional ARIMA model. *Journal of Monetary Economics* 29: 277–302. https://doi.org/10.1016/0304-3932(92)90016-U.

———. 1992b. Maximum likelihood estimation of stationary univariate fractionally integrated time series models. *Journal of Econometrics* 53: 165–188. https://doi.org/10.1016/0304-4076(92)90084-5.

, ed.[], 2015. *Table of Integrals, Series, and Products*. 8th ed. Waltham, MA: Elsevier.

Also see

[TS] **arfima postestimation** — Postestimation tools for arfima

[TS] **arfimasoc** — Obtain lag-order selection statistics for ARFIMAs

[TS] **arima** — ARIMA, ARMAX, and other dynamic regression models

[TS] **sspace** — State-space models

[TS] **tsset** — Declare data to be time-series data

[U] 20 Estimation and postestimation commands