

Time series observations, which are drawn sequentially, usually embody a structure where time is an important component. If you are unable to completely model this structure in the regression function itself, then the remainder spills over into the unobserved component of your model (its error) and this causes the errors of the statistical model to be correlated with one another.

In the following example consider is one with a lag in the error term. Below is a supply response for an agricultural crop modeled in log-log form: planted (acres) depends on price.

$$\ln(A_t) = \beta_1 + \beta_2 \ln(P_t) + e_t$$

The error of the model depends on the previous period's error and **white noise**. In time series analysis it is common to refer to independently distributed random errors as white noise.

$$e_t = \rho e_{t-1} + v_t$$

where ρ (rho) is a parameter that describes the dependence of e_t on e_{t-1} and v_t is a new random error term. Mathematically, the v_t behave according to

$$E(v_t) = 0 \quad \text{var}(v_t) = \sigma_v^2 \quad \text{cov}(v_t, v_s) = 0 \quad \text{for } t \neq s$$

For stability (and stationarity) of the model, we also require $-1 < \rho < 1$. As is demonstrated in your text, the e_t will be correlated with one another and are therefore said to be **autocorrelated**.

Hints

In order to use Stata, you'll have to create a time variable and declare the dataset to be a time-series. The simplest way is

```
gen time = _n
```

This generates a new variable that contains the observation number. This works because `_n` is understood by Stata to mean the observation number. Then, change the structure of your data to a time series using the `tsset` command

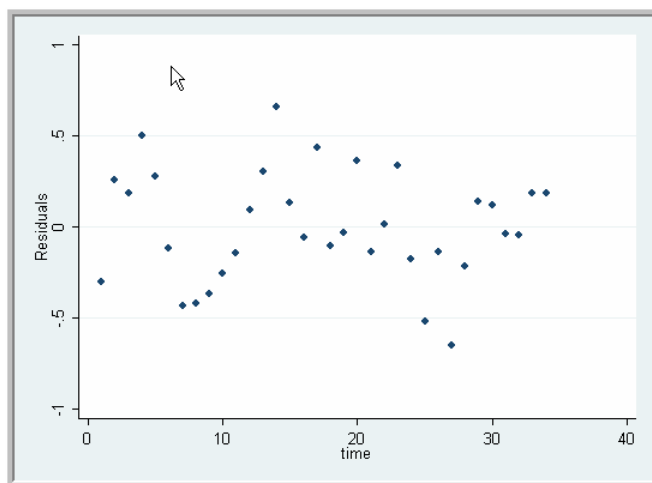
```
tsset time, yearly
```

Now, generate the logarithms of acreage and price, run the regression, and save the residuals

```
gen lp = log(p)  
gen la = log(a)  
  
regress la lp  
predict ehat, residual
```

Now, use the `twoway(scatter ehat time)` to generate a graph of the least squares residuals.

twoway (scatter ehat time)



From the graph there is some indication of positive autocorrelation; positive errors tend to follow other positive errors and negative errors follow other negative ones. Computing the correlation between \hat{e}_t and \hat{e}_{t-1} will give you numerical verification of what you see in the graph. First, generate lagged values of the residuals. This can be done in several ways. The first uses Stata's **subscripting** function.

Individual elements of variables and matrices may be referred to by **subscripting**. For instance, the 10th observation of the variable **x** would be referred to as **x[10]**. The subscript refers to the observation number and it is put inside of square brackets that follow the variable. To refer to lagged values of the variable **ehat**, you can use the subscript **_n-1** as in

```
gen ehat_1 = ehat[_n-1]
```

The subscript uses the observation number (**_n**) and subtracts one from it. So, **ehat[_n-1]** points to \hat{e}_{t-1} which it stores in the new variable called **ehat_1**.

The preferred method for generating lagged values uses Stata's built in lag function. The syntax for the lag function consists of a capital letter **L** followed by the desired lag number (e.g., 1, 2, or 3) and a period. This is used as prefix for the variable to be lagged (e.g., **L#.varname**). For instance, **L1.gnp** refers to the one period lagged value of variable **gnp**. To lag **gnp** two periods, you'd use **L2.gnp**. The lag function can be used only if the data are declared to be time series or as a panel.

The syntax to lag **ehat** one period is

```
gen ehat_1 = L1.ehat
```

For one period lags (**L1.varname**) the number **1** can be omitted (e.g., **L.varname**). There are other time series operators to take differences (**D.**), leads (**F.**), and seasonal differences (**S.**) that use the same syntax structure. The Stata documentation that describes how to use these is cleverly hidden in the online help system and can be revealed by typing **help tsvarlist** on the command line.

Finally, to obtain the correlation between \hat{e}_t and \hat{e}_{t-1} :

```
correlate ehat ehat_1
```

which yields

	ehat	ehat_1
ehat	1.0000	
ehat_1	0.4040	1.0000

ESTIMATING AN AR(1) MODEL

When the errors follow an AR(1) model $e_t = \rho e_{t-1} + v_t$, the least squares assumption MR4, $\text{cov}(e_t, e_s) = 0$ for $t \neq s$ is violated. Least squares is unbiased and consistent, but no longer efficient. Also, the usual standard errors are no longer correct, leading to statistically invalid hypothesis tests and confidence intervals.

Least squares and HAC standard errors

Although the usual least squares standard errors are not correct, we can compute consistent standard errors just as we did in heteroskedastic models using an estimator proposed by Newey and West. Newey-West standard errors (also known as HAC (heteroskedasticity and autocorrelation consistent) standard errors are analogous to the heteroskedasticity consistent standard errors introduced earlier in the course. They have the advantage of being consistent in models that have higher order autocorrelated errors. Also, they do not require explicit specification of the dynamic model of the errors that would otherwise be needed to estimate the parameters more precisely.

HAC is not quite as automatic as the heteroskedasticity robust standard error estimator. To be robust with respect to autocorrelation one has to specify how far away in time the autocorrelation is likely to be significant. Essentially, the autocorrelated errors over the chosen time window are averaged in the computation of HAC; you have to specify how many periods over which to average and how much weight to assign each residual in that average.

The weighted average is called a **kernel** and the number of errors to average is called **bandwidth**. To be quite honest, these terms shed no light on their meaning for the average user. Just think of the kernel as another name for weighted average and bandwidth as the term for number of terms to average. In Stata, you have no control over the kernel, but you can pick a bandwidth.

Implicitly there is a trade-off to consider. A larger bandwidth reduces bias (good), but increases variance (bad). A smaller bandwidth excludes more relevant autocorrelations. While this reduces variance, it increases bias. The tradeoff leads to a Goldilocks prescription to choose the bandwidth that is “just right!” Unfortunately, where this lies is generally not known and this makes the HAC less attractive in practice than its heteroskedasticity robust equivalent.

That is not to say that people haven’t tried to find an optimal bandwidth. Several methods have been proposed to compute one based on sample size. One methods uses $B = 0.75N^{1/3}$. The

larger your sample, N , the larger the bandwidth is. Another popular choice is $B=4(N/100)^{2/9}$. This one appears to be the default in other programs like EViews and it is the one used here to obtain the results in the text.

To compute this one in Stata use:

```
scalar B = round(4*(e(N)/100)^(2/9))
scalar list B
```

This returns the value 3. The result is rounded using the **round** function because you have to give Stata a whole number to specify the number of lags to use in the HAC's computation. Then, to estimate the model with least squares with Newey-West standard errors use the following command

```
newey la lp, lag(3)
```

The dialog to estimate this model is found by selecting **Statistics > Time series > Regression with Newey-West std. errors**. This brings up the **newey** dialog box shown below. Select the dependent and independent variables. Then click the radio button to select the bandwidth, choosing 3. Click **OK** and you'll obtain the results from your text. Note, Stata computes the 95% confidence intervals by default. If you wish to change α you can do so from the **Reporting** tab in the dialog.

Finally, if you choose the **No autocorrelation structure** in this dialog, you'll get the usual White's standard errors that are robust to heteroskedasticity only. Therefore, if you have autocorrelation, then you must tell Stata the maximum lag to consider.

Nonlinear Least Squares

As you can see, HAC standard errors suffer at least two disadvantages: 1) they are not automatic since they require specification of a bandwidth and 2) they are larger than standard errors of more efficient estimators. In this section, nonlinear least squares is used to efficiently estimate the parameters of the AR(1) model.

In your text book the authors start with the AR(1) regression model and, using a little algebra, arrive at

$$y_t = \beta_1(1 - \rho) + \beta_2 x_t + \rho y_{t-1} - \rho \beta_2 x_{t-1} + v_t$$

This model is nonlinear in the parameters, but has an additive white noise error. These features make the model suitable for nonlinear least squares estimation. Nonlinear least squares uses numerical methods to find the values of the parameters that minimize the sum of squared errors. Stata can do this easily using the **nl** command.

Unfortunately, the **nl** command cannot use Stata's built in time series operators (e.g., **L.var**, **D.var**) so you will have to use the **generate** command to take the lags of the dependent and independent variables, y and x . These lines use the **L.** operator to generate lags of $\ln(p)$ and $\ln(a)$.

```
gen la_1 = L1.la
gen lp_1 = L1.lp
```

Once these are defined, the **n1** command to estimate the model is

```
n1 (la = {b1}*(1-{rho}) + {b2}*lp_1+ {rho}*la_1 - {rho}*{b2}*(lp_1)),
    variables(lp la la_1 lp_1)
```

This should be typed on a single line, either in a do-file or in the **command** window. The syntax is fairly simple, but requires some explanation. The basic syntax is:

```
n1 (depvar=<sexp>) [if] [in] [weight] [, options]
```

Inside the first set of parentheses you type in the systematic portion of your model. Parameters must be enclosed in braces {}. You can use the **if**, **in**, and **weight** statements just as you do in a linear regression. However, because you are using variables that have been lagged, you will now have missing values for the lagged variables in the data set. For **n1** to work you must limit the sample to the complete observations, i.e., the ones for which nothing is missing. There are two ways to do this. First, you could use **(depvar=<sexp>) in 2/34**. Or, you can list the variables as we have done here using the option **variables(lp la la_1 lp_1)**.

A More General Model

A more general form of the model is considered

$$y_t = \delta + \delta_0 x_t + \delta_1 x_{t-1} + \theta_1 y_{t-1} + v_t$$

which is linear in the parameters and can be estimated by linear regression. This model is related to the previous model by the relationships

$$\delta = \beta_1(1 - \rho) \quad \delta_0 = \beta_2 \quad \delta_1 = -\rho\beta_2 \quad \theta_1 = \rho$$

The linear model can be estimated by (linear) least squares and a hypothesis test of the implied restriction can be conducted. The null hypothesis implied by the restriction is $H_0 : \delta_1 = -\theta_1\delta_0$ against the alternative that it is not equal.

DETECTING AUTOCORRELATION

Several methods are used to determine the presence or extent of autocorrelation. The first is to take a look at the residual **correlogram**. A residual correlogram is a graph that plots series of correlations between \hat{e}_t and \hat{e}_{t-j} against the time interval between the observations, $j=1,2,\dots, m$.

So, the first thing to do get the **residual correlogram** is to estimate the model using least squares and then save the residuals.

```
regress la lp
predict ehat, residual
```

Once you have the residuals, use the command

```
ac ehat
```

to graph the correlogram.

Each ‘dot’ on the correlogram represents the estimated correlation between observations j periods apart and the shaded area is the 95% confidence bounds. So, in this case the first autocorrelation lies outside of the boundary and is therefore significantly different from zero at the 5% level. The others lie inside the bounds and are not significant.

Another way to test for autocorrelation is to use the output from nonlinear least squares, which yields an estimate of ρ and its standard error. The estimator of ρ is approximately normally distributed which means that you can use its t-ratio in the usual way.

Finally, you can test whether residuals are correlated with one another using an **LM (Lagrange multiplier)** test. For autocorrelation, this test is based on an auxiliary regression where you regress least squares residuals on lagged least squares residuals and the original regressors. If the regressors, which include \hat{e}_{t-1} , explain sufficient variation in \hat{e}_t then there must be autocorrelation due to \hat{e}_{t-1} . For a regression model

$$y_t = \beta_1 + \beta_2 x_t + e_t,$$

estimate the parameters using least squares and save the residuals, \hat{e}_t . Lag the residuals to get \hat{e}_{t-1} . Then estimate a second ‘auxiliary’ regression with \hat{e}_t as the dependent variable and the lagged value \hat{e}_{t-1} as an independent variable. Include all of the other independent variables from the original regression as well. For a simple linear regression the auxiliary regression is

$$\hat{e}_t = \gamma_1 + \gamma_2 x_t + \rho \hat{e}_{t-1} + \text{residual}$$

NR^2 from this regression has a $\chi^2(1)$ distribution if the null hypothesis of no autocorrelation is true, where N is the number of observations in the auxiliary regression. Rejection leads you to conclude there is significant autocorrelation.

If you suspect higher order autocorrelation, include additional lags of the residuals. NR^2 from these regressions has a $\chi^2(p)$ distribution if the null hypothesis is true and p is the number of lagged residuals included in the model.

Interestingly, Stata has a built in command to do this test. It is a post-estimation command, which uses **estat bgodfrey**:

```
regress la lp
estat bgodfrey lags(1/3)
```

bgodfrey stands for “Breusch-Godfrey”, the last names of the two econometricians credited with this particular variant of the test. This command gives the same result as the manual method above. The `lags` option specifies which lagged residuals to include in the LM test.

AUTOREGRESSIVE MODELS

Autoregressive models include lags of the dependent variable as regressors. The AR(p) model is

$$y_t = \delta + \theta_1 y_{t-1} + \theta_2 y_{t-2} + \dots + \theta_p y_{t-p} + v_t$$

which has p lags of y_t as regressors. This model is simple to estimate in Stata since it is linear in the parameters. In this example we use the *inflation.dta* dataset and estimate an AR(3) model of the inflation rate. First, load the inflation data, clearing the memory of any existing data.

```
use inflation, clear
```

You’ll notice that in this dataset there are variables for year and month. These two variables can be combined to make a time variable that we will call **tt**. To do this use a function called **ym** (which stands for year/month) as shown below.

```
gen tt = ym(year,month)  
tsset tt, monthly
```

Since the data are recorded monthly, we used the **monthly** option in declaring **tt** to be a time series with **tsset**. Once the data have been declared time series you can use the built in time series operators (**L1.**, **L2.**, **L3.**, etc) to specify lags for variables in the model. The regression is

```
regress infln L1.infln L2.infln L3.infln
```

FINITE DISTRIBUTED LAG MODELS

Finite distributed lag models contain independent variables and their lags as regressors.

$$y_t = \alpha + \beta_0 x_t + \beta_1 x_{t-1} + \beta_2 x_{t-2} + \dots + \beta_q x_{t-q} + v_t, \quad t = q+1, \dots, T$$

In this example the y_t is the inflation rate and it is modeled as a function of the percentage change in wages, x_t , for the current and three periods lagged. In terms of the general model, $q=3$.

In Stata the model is linear and estimated using least squares

```
regress infln pcwage L1.pcwage L2.pcwage L3.pcwage
```

AUTOREGRESSIVE DISTRIBUTED LAG MODELS

Finally, we consider an autoregressive model that also contains a finite distributed lag. This is the so-called autoregressive distributed lag model (ARDL). The ARDL(p, q) model has the general form

$$y_t = \delta + \delta_0 x_t + \delta_1 x_{t-1} + \dots + \delta_q x_{t-q} + \theta_1 y_{t-1} + \dots + \theta_p y_{t-p} + v_t$$

It has p lags of the dependent variable, y_t , and q lags of the independent variable, x_t . The ARDL(2,3) model of inflation can be estimated using least squares

```
regress infln pcwage L1.pcwage L2.pcwage L3.pcwage L1.infln L2.infln
```

QUESTION 1

Use the `bangla.dta` dataset to estimate the area response model on page 1 using least squares with the wrong standard errors. Plot the residuals against time. Plot the correlogram and determine the likely extent of autocorrelation. Resestimate the model using least squares with HAC standard errors.

QUESTION 2

Estimate the model using assuming first order autocorrelation using nonlinear least squares. Test the null hypothesis of no autocorrelation against first order autocorrelation using the nonlinear least squares result. If there is higher order autocorrelation present, how would you detect it and what would you do to model it?

Use the `testnl` command to test the nonlinear restrictions implied by the NLS estimator relative to the “more general model” on page 5

QUESTION 3

Using the `inflation.dta` data estimate the finite distributed lag model and the ARDL model. Plot the correlogram of the residuals from each. Do the errors appear to be correlated? Using the appropriate Breusch-Godfrey test to test no autocorrelation against a suitable alternative. If further consideration of the model’s specification is warranted, how would you proceed?

