# Chapter 8
## The ARIMA Procedure

## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview: ARIMA Procedure</td>
<td>216</td>
</tr>
<tr>
<td>Getting Started: ARIMA Procedure</td>
<td>217</td>
</tr>
<tr>
<td>The Three Stages of ARIMA Modeling</td>
<td>217</td>
</tr>
<tr>
<td>Identification Stage</td>
<td>218</td>
</tr>
<tr>
<td>Estimation and Diagnostic Checking Stage</td>
<td>223</td>
</tr>
<tr>
<td>Forecasting Stage</td>
<td>229</td>
</tr>
<tr>
<td>Using ARIMA Procedure Statements</td>
<td>231</td>
</tr>
<tr>
<td>General Notation for ARIMA Models</td>
<td>231</td>
</tr>
<tr>
<td>Stationarity</td>
<td>234</td>
</tr>
<tr>
<td>Differencing</td>
<td>234</td>
</tr>
<tr>
<td>Subset, Seasonal, and Factored ARMA Models</td>
<td>236</td>
</tr>
<tr>
<td>Input Variables and Regression with ARMA Errors</td>
<td>237</td>
</tr>
<tr>
<td>Intervention Models and Interrupted Time Series</td>
<td>240</td>
</tr>
<tr>
<td>Rational Transfer Functions and Distributed Lag Models</td>
<td>241</td>
</tr>
<tr>
<td>Forecasting with Input Variables</td>
<td>243</td>
</tr>
<tr>
<td>Data Requirements</td>
<td>244</td>
</tr>
<tr>
<td>Syntax: ARIMA Procedure</td>
<td>245</td>
</tr>
<tr>
<td>Functional Summary</td>
<td>245</td>
</tr>
<tr>
<td>PROC ARIMA Statement</td>
<td>248</td>
</tr>
<tr>
<td>BY Statement</td>
<td>250</td>
</tr>
<tr>
<td>IDENTIFY Statement</td>
<td>251</td>
</tr>
<tr>
<td>ESTIMATE Statement</td>
<td>255</td>
</tr>
<tr>
<td>OUTLIER Statement</td>
<td>259</td>
</tr>
<tr>
<td>FORECAST Statement</td>
<td>260</td>
</tr>
<tr>
<td>Details: ARIMA Procedure</td>
<td>261</td>
</tr>
<tr>
<td>The Inverse Autocorrelation Function</td>
<td>261</td>
</tr>
<tr>
<td>The Partial Autocorrelation Function</td>
<td>262</td>
</tr>
<tr>
<td>The Cross-Correlation Function</td>
<td>262</td>
</tr>
<tr>
<td>The ESACF Method</td>
<td>263</td>
</tr>
<tr>
<td>The MINIC Method</td>
<td>265</td>
</tr>
<tr>
<td>The SCAN Method</td>
<td>266</td>
</tr>
<tr>
<td>Stationarity Tests</td>
<td>268</td>
</tr>
<tr>
<td>Prewhitening</td>
<td>268</td>
</tr>
<tr>
<td>Identifying Transfer Function Models</td>
<td>269</td>
</tr>
<tr>
<td>Missing Values and Autocorrelations</td>
<td>269</td>
</tr>
<tr>
<td>Estimation Details</td>
<td>270</td>
</tr>
</tbody>
</table>
Overview: ARIMA Procedure

The ARIMA procedure analyzes and forecasts equally spaced univariate time series data, transfer function data, and intervention data by using the autoregressive integrated moving-average (ARIMA) or autoregressive moving-average (ARMA) model. An ARIMA model predicts a value in a response time series as a linear combination of its own past values, past errors (also called shocks or innovations), and current and past values of other time series.

The ARIMA approach was first popularized by Box and Jenkins, and ARIMA models are often referred to as Box-Jenkins models. The general transfer function model employed by the ARIMA procedure was discussed by Box and Tiao (1975). When an ARIMA model includes other time series as input variables, the model is sometimes referred to as an ARIMAX model. Pankratz (1991) refers to the ARIMAX model as dynamic regression.

The ARIMA procedure provides a comprehensive set of tools for univariate time series model identification, parameter estimation, and forecasting, and it offers great flexibility in the kinds of ARIMA or ARIMAX...
models that can be analyzed. The ARIMA procedure supports seasonal, subset, and factored ARIMA models; intervention or interrupted time series models; multiple regression analysis with ARMA errors; and rational transfer function models of any complexity.

The design of PROC ARIMA closely follows the Box-Jenkins strategy for time series modeling with features for the identification, estimation and diagnostic checking, and forecasting steps of the Box-Jenkins method.

Before you use PROC ARIMA, you should be familiar with Box-Jenkins methods, and you should exercise care and judgment when you use the ARIMA procedure. The ARIMA class of time series models is complex and powerful, and some degree of expertise is needed to use them correctly.

---

**Getting Started: ARIMA Procedure**

This section outlines the use of the ARIMA procedure and gives a cursory description of the ARIMA modeling process for readers who are less familiar with these methods.

---

**The Three Stages of ARIMA Modeling**

The analysis performed by PROC ARIMA is divided into three stages, corresponding to the stages described by Box and Jenkins (1976).

1. **In the identification stage**, you use the IDENTIFY statement to specify the response series and identify candidate ARIMA models for it. The IDENTIFY statement reads time series that are to be used in later statements, possibly differencing them, and computes autocorrelations, inverse autocorrelations, partial autocorrelations, and cross-correlations. Stationarity tests can be performed to determine if differencing is necessary. The analysis of the IDENTIFY statement output usually suggests one or more ARIMA models that could be fit. Options enable you to test for stationarity and tentative ARMA order identification.

2. **In the estimation and diagnostic checking stage**, you use the ESTIMATE statement to specify the ARIMA model to fit to the variable specified in the previous IDENTIFY statement and to estimate the parameters of that model. The ESTIMATE statement also produces diagnostic statistics to help you judge the adequacy of the model.

   Significance tests for parameter estimates indicate whether some terms in the model might be unnecessary. Goodness-of-fit statistics aid in comparing this model to others. Tests for white noise residuals indicate whether the residual series contains additional information that might be used by a more complex model. The OUTLIER statement provides another useful tool to check whether the currently estimated model accounts for all the variation in the series. If the diagnostic tests indicate problems with the model, you try another model and then repeat the estimation and diagnostic checking stage.

3. **In the forecasting stage**, you use the FORECAST statement to forecast future values of the time series and to generate confidence intervals for these forecasts from the ARIMA model produced by the preceding ESTIMATE statement.

These three steps are explained further and illustrated through an extended example in the following sections.
Identification Stage

Suppose you have a variable called SALES that you want to forecast. The following example illustrates ARIMA modeling and forecasting by using a simulated data set TEST that contains a time series SALES generated by an ARIMA(1,1,1) model. The output produced by this example is explained in the following sections. The simulated SALES series is shown in Figure 8.1.

```
proc sgplot data=test;
   scatter y=sales x=date;
run;
```

**Figure 8.1** Simulated ARIMA(1,1,1) Series SALES
Using the IDENTIFY Statement

You first specify the input data set in the PROC ARIMA statement. Then, you use an IDENTIFY statement to read in the SALES series and analyze its correlation properties. You do this by using the following statements:

```plaintext
proc arima data=test ;
   identify var=sales nlag=24;
run;
```

Descriptive Statistics

The IDENTIFY statement first prints descriptive statistics for the SALES series. This part of the IDENTIFY statement output is shown in Figure 8.2.

**Figure 8.2 IDENTIFY Statement Descriptive Statistics Output**

**The ARIMA Procedure**

<table>
<thead>
<tr>
<th>Name of Variable = sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean of Working Series</td>
</tr>
<tr>
<td>Standard Deviation</td>
</tr>
<tr>
<td>Number of Observations</td>
</tr>
</tbody>
</table>

Autocorrelation Function Plots

The IDENTIFY statement next produces a panel of plots used for its autocorrelation and trend analysis. The panel contains the following plots:

- the time series plot of the series
- the sample autocorrelation function plot (ACF)
- the sample inverse autocorrelation function plot (IACF)
- the sample partial autocorrelation function plot (PACF)

This correlation analysis panel is shown in Figure 8.3.
These autocorrelation function plots show the degree of correlation with past values of the series as a function of the number of periods in the past (that is, the lag) at which the correlation is computed.

The NLAG= option controls the number of lags for which the autocorrelations are shown. By default, the autocorrelation functions are plotted to lag 24.

Most books on time series analysis explain how to interpret the autocorrelation and the partial autocorrelation plots. See the section “The Inverse Autocorrelation Function” on page 261 for a discussion of the inverse autocorrelation plots.

By examining these plots, you can judge whether the series is stationary or nonstationary. In this case, a visual inspection of the autocorrelation function plot indicates that the SALES series is nonstationary, since the ACF decays very slowly. For more formal stationarity tests, use the STATIONARITY= option. (See the section “Stationarity” on page 234.)

**White Noise Test**

The last part of the default IDENTIFY statement output is the check for white noise. This is an approximate statistical test of the hypothesis that none of the autocorrelations of the series up to a given lag are significantly different from 0. If this is true for all lags, then there is no information in the series to model, and no ARIMA model is needed for the series.
The autocorrelations are checked in groups of six, and the number of lags checked depends on the NLAG= option. The check for white noise output is shown in Figure 8.4.

**Figure 8.4 IDENTIFY Statement Check for White Noise**

<table>
<thead>
<tr>
<th>To Lag</th>
<th>Chi-Square</th>
<th>DF</th>
<th>Pr &gt; ChiSq</th>
<th>Autocorrelations</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>426.44</td>
<td>6</td>
<td>&lt;.0001</td>
<td>0.957 0.907 0.852 0.791 0.726 0.659</td>
</tr>
<tr>
<td>12</td>
<td>547.82</td>
<td>12</td>
<td>&lt;.0001</td>
<td>0.588 0.514 0.440 0.370 0.303 0.238</td>
</tr>
<tr>
<td>18</td>
<td>554.70</td>
<td>18</td>
<td>&lt;.0001</td>
<td>0.174 0.112 0.052 -0.004 -0.054 -0.098</td>
</tr>
<tr>
<td>24</td>
<td>585.73</td>
<td>24</td>
<td>&lt;.0001</td>
<td>-0.135 -0.167 -0.192 -0.211 -0.227 -0.240</td>
</tr>
</tbody>
</table>

In this case, the white noise hypothesis is rejected very strongly, which is expected since the series is nonstationary. The $p$-value for the test of the first six autocorrelations is printed as <0.0001, which means the $p$-value is less than 0.0001.

**Identification of the Differenced Series**

Since the series is nonstationary, the next step is to transform it to a stationary series by differencing. That is, instead of modeling the SALES series itself, you model the change in SALES from one period to the next. To difference the SALES series, use another IDENTIFY statement and specify that the first difference of SALES be analyzed, as shown in the following statements:

```plaintext
proc arima data=test;
   identify var=sales(1);
run;
```

The second IDENTIFY statement produces the same information as the first, but for the change in SALES from one period to the next rather than for the total SALES in each period. The summary statistics output from this IDENTIFY statement is shown in Figure 8.5. Note that the period of differencing is given as 1, and one observation was lost through the differencing operation.

**Figure 8.5 IDENTIFY Statement Output for Differenced Series**

<table>
<thead>
<tr>
<th>The ARIMA Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of Variable = sales</td>
</tr>
<tr>
<td>Period(s) of Differencing</td>
</tr>
<tr>
<td>Mean of Working Series</td>
</tr>
<tr>
<td>Standard Deviation</td>
</tr>
<tr>
<td>Number of Observations</td>
</tr>
<tr>
<td>Observation(s) eliminated by differencing</td>
</tr>
</tbody>
</table>

The autocorrelation plots for the differenced series are shown in Figure 8.6.
The autocorrelations decrease rapidly in this plot, indicating that the change in SALES is a stationary time series.

The next step in the Box-Jenkins methodology is to examine the patterns in the autocorrelation plot to choose candidate ARMA models to the series. The partial and inverse autocorrelation function plots are also useful aids in identifying appropriate ARMA models for the series.

In the usual Box-Jenkins approach to ARIMA modeling, the sample autocorrelation function, inverse autocorrelation function, and partial autocorrelation function are compared with the theoretical correlation functions expected from different kinds of ARMA models. This matching of theoretical autocorrelation functions of different ARMA models to the sample autocorrelation functions computed from the response series is the heart of the identification stage of Box-Jenkins modeling. Most textbooks on time series analysis, such as Pankratz (1983), discuss the theoretical autocorrelation functions for different kinds of ARMA models.

Since the input data are only a limited sample of the series, the sample autocorrelation functions computed from the input series only approximate the true autocorrelation function of the process that generates the series. This means that the sample autocorrelation functions do not exactly match the theoretical autocorrelation functions for any ARMA model and can have a pattern similar to that of several different ARMA models. If the series is white noise (a purely random process), then there is no need to fit a model. The check for
white noise, shown in Figure 8.7, indicates that the change in SALES is highly autocorrelated. Thus, an autocorrelation model, for example an AR(1) model, might be a good candidate model to fit to this process.

**Figure 8.7 IDENTIFY Statement Check for White Noise**

<table>
<thead>
<tr>
<th>To Lag</th>
<th>Autocorrelation Check for White Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Chi-Square</td>
</tr>
<tr>
<td>6</td>
<td>154.44 6</td>
</tr>
<tr>
<td>12</td>
<td>173.66 12</td>
</tr>
<tr>
<td>18</td>
<td>209.64 18</td>
</tr>
<tr>
<td>24</td>
<td>218.04 24</td>
</tr>
</tbody>
</table>

**Estimation and Diagnostic Checking Stage**

The autocorrelation plots for this series, as shown in the previous section, suggest an AR(1) model for the change in SALES. You should check the diagnostic statistics to see if the AR(1) model is adequate. Other candidate models include an MA(1) model and low-order mixed ARMA models. In this example, the AR(1) model is tried first.

**Estimating an AR(1) Model**

The following statements fit an AR(1) model (an autoregressive model of order 1), which predicts the change in SALES as an average change, plus some fraction of the previous change, plus a random error. To estimate an AR model, you specify the order of the autoregressive model with the P= option in an ESTIMATE statement:

```
estimate p=1;
run;
```

The ESTIMATE statement fits the model to the data and prints parameter estimates and various diagnostic statistics that indicate how well the model fits the data. The first part of the ESTIMATE statement output, the table of parameter estimates, is shown in Figure 8.8.

**Figure 8.8 Parameter Estimates for AR(1) Model**

| MU    | 0.90280 | 0.65984 | 1.37 | 0.1744 | 0 |
| AR1,1 | 0.86847 | 0.05485 | 15.83 | <0.001 | 1 |

The table of parameter estimates is titled “Conditional Least Squares Estimation,” which indicates the estimation method used. You can request different estimation methods with the METHOD= option.

The table of parameter estimates lists the parameters in the model; for each parameter, the table shows the estimated value and the standard error and $t$ value for the estimate. The table also indicates the lag at which the parameter appears in the model.
In this case, there are two parameters in the model. The mean term is labeled MU; its estimated value is 0.90280. The autoregressive parameter is labeled AR1,1; this is the coefficient of the lagged value of the change in SALES, and its estimate is 0.86847.

The $t$ values provide significance tests for the parameter estimates and indicate whether some terms in the model might be unnecessary. In this case, the $t$ value for the autoregressive parameter is 15.83, so this term is highly significant. The $t$ value for MU indicates that the mean term adds little to the model.

The standard error estimates are based on large sample theory. Thus, the standard errors are labeled as approximate, and the standard errors and $t$ values might not be reliable in small samples.

The next part of the ESTIMATE statement output is a table of goodness-of-fit statistics, which aid in comparing this model to other models. This output is shown in Figure 8.9.

**Figure 8.9 Goodness-of-Fit Statistics for AR(1) Model**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant Estimate</td>
<td>0.118749</td>
</tr>
<tr>
<td>Variance Estimate</td>
<td>1.15794</td>
</tr>
<tr>
<td>Std Error Estimate</td>
<td>1.076076</td>
</tr>
<tr>
<td>AIC</td>
<td>297.4469</td>
</tr>
<tr>
<td>SBC</td>
<td>302.6372</td>
</tr>
<tr>
<td>Number of Residuals</td>
<td>99</td>
</tr>
</tbody>
</table>

The “Constant Estimate” is a function of the mean term MU and the autoregressive parameters. This estimate is computed only for AR or ARMA models, but not for strictly MA models. See the section “General Notation for ARIMA Models” on page 231 for an explanation of the constant estimate.

The “Variance Estimate” is the variance of the residual series, which estimates the innovation variance. The item labeled “Std Error Estimate” is the square root of the variance estimate. In general, when you are comparing candidate models, smaller AIC and SBC statistics indicate the better fitting model. The section “Estimation Details” on page 270 explains the AIC and SBC statistics.

The ESTIMATE statement next prints a table of correlations of the parameter estimates, as shown in Figure 8.10. This table can help you assess the extent to which collinearity might have influenced the results. If two parameter estimates are very highly correlated, you might consider dropping one of them from the model.

**Figure 8.10 Correlations of the Estimates for AR(1) Model**

<table>
<thead>
<tr>
<th></th>
<th>MU</th>
<th>AR1,1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter</td>
<td>MU</td>
<td>AR1,1</td>
</tr>
<tr>
<td>MU</td>
<td>1.000</td>
<td>0.114</td>
</tr>
<tr>
<td>AR1,1</td>
<td>0.114</td>
<td>1.000</td>
</tr>
</tbody>
</table>

The next part of the ESTIMATE statement output is a check of the autocorrelations of the residuals. This output has the same form as the autocorrelation check for white noise that the IDENTIFY statement prints for the response series. The autocorrelation check of residuals is shown in Figure 8.11.
Figure 8.11 Check for White Noise Residuals for AR(1) Model

<table>
<thead>
<tr>
<th>To Lag</th>
<th>Chi-Square</th>
<th>DF</th>
<th>Pr &gt; ChiSq</th>
<th>Autocorrelations</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>19.09</td>
<td>5</td>
<td>0.0019</td>
<td>-0.327 -0.220 -0.128 0.068 -0.002 -0.096</td>
</tr>
<tr>
<td>12</td>
<td>22.90</td>
<td>11</td>
<td>0.0183</td>
<td>0.072 0.116 -0.042 -0.066 0.031 -0.091</td>
</tr>
<tr>
<td>18</td>
<td>31.63</td>
<td>17</td>
<td>0.0167</td>
<td>-0.233 -0.129 -0.024 0.056 -0.014 -0.008</td>
</tr>
<tr>
<td>24</td>
<td>32.83</td>
<td>23</td>
<td>0.0841</td>
<td>0.009 -0.057 -0.057 -0.001 0.049 -0.015</td>
</tr>
</tbody>
</table>

The $\chi^2$ test statistics for the residuals series indicate whether the residuals are uncorrelated (white noise) or contain additional information that might be used by a more complex model. In this case, the test statistics reject the no-autocorrelation hypothesis at a high level of significance ($p = 0.0019$ for the first six lags.) This means that the residuals are not white noise, and so the AR(1) model is not a fully adequate model for this series. The ESTIMATE statement output also includes graphical analysis of the residuals. It is not shown here. The graphical analysis also reveals the inadequacy of the AR(1) model.

The final part of the ESTIMATE statement output is a listing of the estimated model, using the backshift notation. This output is shown in Figure 8.12.

Figure 8.12 Estimated ARIMA(1, 1, 0) Model for SALES

<table>
<thead>
<tr>
<th>Model for variable sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimated Mean</td>
</tr>
<tr>
<td>Period(s) of Differencing</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Autoregressive Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor 1:</td>
</tr>
</tbody>
</table>

This listing combines the differencing specification given in the IDENTIFY statement with the parameter estimates of the model for the change in SALES. Since the AR(1) model is for the change in SALES, the final model for SALES is an ARIMA(1,1,0) model. Using $B$, the backshift operator, the mathematical form of the estimated model shown in this output is as follows:

$$(1 - B) sales_t = 0.902799 + \frac{1}{1 - 0.86847B} \alpha_t$$

See the section “General Notation for ARIMA Models” on page 231 for further explanation of this notation.

Estimating an ARMA(1,1) Model

The IDENTIFY statement plots suggest a mixed autoregressive and moving-average model, and the previous ESTIMATE statement check of residuals indicates that an AR(1) model is not sufficient. You now try estimating an ARMA(1,1) model for the change in SALES.

An ARMA(1,1) model predicts the change in SALES as an average change, plus some fraction of the previous change, plus a random error, plus some fraction of the random error in the preceding period. An ARMA(1,1) model for the change in SALES is the same as an ARIMA(1,1,1) model for the level of SALES.
To estimate a mixed autoregressive moving-average model, you specify the order of the moving-average part of the model with the Q= option in an ESTIMATE statement in addition to specifying the order of the autoregressive part with the P= option. The following statements fit an ARMA(1,1) model to the differenced SALES series:

```plaintext
estimate p=1 q=1;
run;
```

The parameter estimates table and goodness-of-fit statistics for this model are shown in Figure 8.13.

**Figure 8.13** Estimated ARMA(1, 1) Model for Change in SALES

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>t Value</th>
<th>Approx Pr &gt;</th>
<th>Lag</th>
</tr>
</thead>
<tbody>
<tr>
<td>MU</td>
<td>0.89288</td>
<td>0.49391</td>
<td>1.81</td>
<td>0.0738</td>
<td>0</td>
</tr>
<tr>
<td>MA1,1</td>
<td>-0.58935</td>
<td>0.08988</td>
<td>-6.56</td>
<td>&lt;.0001</td>
<td>1</td>
</tr>
<tr>
<td>AR1,1</td>
<td>0.74755</td>
<td>0.07785</td>
<td>9.60</td>
<td>&lt;.0001</td>
<td>1</td>
</tr>
</tbody>
</table>

The moving-average parameter estimate, labeled “MA1,1”, is –0.58935. Both the moving-average and the autoregressive parameters have significant t values. Note that the variance estimate, AIC, and SBC are all smaller than they were for the AR(1) model, indicating that the ARMA(1,1) model fits the data better without over-parameterizing.

The graphical check of the residuals from this model is shown in Figure 8.14 and Figure 8.15. The residual correlation and white noise test plots show that you cannot reject the hypothesis that the residuals are uncorrelated. The normality plots also show no departure from normality. Thus, you conclude that the ARMA(1,1) model is adequate for the change in SALES series, and there is no point in trying more complex models.
Figure 8.14 White Noise Check of Residuals for the ARMA(1,1) Model
The form of the estimated ARIMA(1,1,1) model for SALES is shown in Figure 8.16.

**Figure 8.16 Estimated ARIMA(1,1,1) Model for SALES**

<table>
<thead>
<tr>
<th>Model for variable sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimated Mean</td>
</tr>
<tr>
<td>Period(s) of Differencing</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Autoregressive Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor 1: 1 - 0.74755 B^<strong>(1)</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Moving Average Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor 1: 1 + 0.58935 B^<strong>(1)</strong></td>
</tr>
</tbody>
</table>

The estimated model shown in this output is

\[(1 - B)\text{sales}_t = 0.892875 + \frac{(1 + 0.58935B)}{(1 - 0.74755B)} \text{a}_t\]

In addition to the residual analysis of a model, it is often useful to check whether there are any changes in the time series that are not accounted for by the currently estimated model. The OUTLIER statement enables you to detect such changes. For a long series, this task can be computationally burdensome. Therefore, in general, it is better done after a model that fits the data reasonably well has been found. Figure 8.17 shows the output of the simplest form of the OUTLIER statement:
Two possible outliers have been found for the model in question. See the section “Detecting Outliers” on page 280, and the examples Example 8.6 and Example 8.7, for more details about modeling in the presence of outliers. In this illustration these outliers are not discussed any further.

Figure 8.17 Outliers for the ARIMA(1,1,1) Model for SALES

<table>
<thead>
<tr>
<th>The ARIMA Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outlier Detection Summary</td>
</tr>
<tr>
<td>Maximum number searched</td>
</tr>
<tr>
<td>Number found</td>
</tr>
<tr>
<td>Significance used</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Outlier Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obs Type</td>
</tr>
<tr>
<td>----------</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>67</td>
</tr>
</tbody>
</table>

Since the model diagnostic tests show that all the parameter estimates are significant and the residual series is white noise, the estimation and diagnostic checking stage is complete. You can now proceed to forecasting the SALES series with this ARIMA(1,1,1) model.

Forecasting Stage

To produce the forecast, use a FORECAST statement after the ESTIMATE statement for the model you decide is best. If the last model fit is not the best, then repeat the ESTIMATE statement for the best model before you use the FORECAST statement.

Suppose that the SALES series is monthly, that you want to forecast one year ahead from the most recently available SALES figure, and that the dates for the observations are given by a variable DATE in the input data set TEST. You use the following FORECAST statement:

```sas
forecast lead=12 interval=month id=date out=results;
run;
```

The LEAD= option specifies how many periods ahead to forecast (12 months, in this case). The ID= option specifies the ID variable, which is typically a SAS date, time, or datetime variable, used to date the observations of the SALES time series. The INTERVAL= option indicates that data are monthly and enables PROC ARIMA to extrapolate DATE values for forecast periods. The OUT= option writes the forecasts to the output data set RESULTS. See the section “OUT= Data Set” on page 282 for information about the contents of the output data set.

By default, the FORECAST statement also prints and plots the forecast values, as shown in Figure 8.18 and Figure 8.19. The forecast table shows for each forecast period the observation number, forecast value, standard error estimate for the forecast value, and lower and upper limits for a 95% confidence interval for the forecast.
### Figure 8.18 Forecasts for ARIMA(1,1,1) Model for SALES

#### The ARIMA Procedure

<table>
<thead>
<tr>
<th>Obs</th>
<th>Forecast</th>
<th>Std Error</th>
<th>Confidence Limits</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>171.0320</td>
<td>0.9508</td>
<td>169.1684 172.8955</td>
</tr>
<tr>
<td>102</td>
<td>174.7534</td>
<td>2.4168</td>
<td>170.0165 179.4903</td>
</tr>
<tr>
<td>103</td>
<td>177.7608</td>
<td>3.9879</td>
<td>169.9445 185.5770</td>
</tr>
<tr>
<td>104</td>
<td>180.2343</td>
<td>5.5658</td>
<td>169.3256 191.1430</td>
</tr>
<tr>
<td>105</td>
<td>182.3088</td>
<td>7.1033</td>
<td>168.3666 196.2310</td>
</tr>
<tr>
<td>106</td>
<td>184.0850</td>
<td>8.5789</td>
<td>167.2707 200.8993</td>
</tr>
<tr>
<td>107</td>
<td>185.6382</td>
<td>9.9841</td>
<td>166.0698 205.2066</td>
</tr>
<tr>
<td>108</td>
<td>187.0247</td>
<td>11.3173</td>
<td>164.8433 209.2061</td>
</tr>
<tr>
<td>109</td>
<td>188.2866</td>
<td>12.5807</td>
<td>163.6289 212.9443</td>
</tr>
<tr>
<td>110</td>
<td>189.4553</td>
<td>13.7784</td>
<td>162.4501 216.4605</td>
</tr>
<tr>
<td>111</td>
<td>190.5544</td>
<td>14.9153</td>
<td>161.3209 219.7879</td>
</tr>
<tr>
<td>112</td>
<td>191.6014</td>
<td>15.9964</td>
<td>160.2491 222.9538</td>
</tr>
</tbody>
</table>

### Figure 8.19 Forecasts for the ARMA(1,1,1) Model
Normally, you want the forecast values stored in an output data set, and you are not interested in seeing this printed list of the forecast. You can use the NOPRINT option in the FORECAST statement to suppress this output.

### Using ARIMA Procedure Statements

The IDENTIFY, ESTIMATE, and FORECAST statements are related in a hierarchy. An IDENTIFY statement brings in a time series to be modeled; several ESTIMATE statements can follow to estimate different ARIMA models for the series; for each model estimated, several FORECAST statements can be used. Thus, a FORECAST statement must be preceded at some point by an ESTIMATE statement, and an ESTIMATE statement must be preceded at some point by an IDENTIFY statement. Additional IDENTIFY statements can be used to switch to modeling a different response series or to change the degree of differencing used.

The ARIMA procedure can be used interactively in the sense that all ARIMA procedure statements can be executed any number of times without reinvokeing PROC ARIMA. You can execute ARIMA procedure statements singly or in groups by following the single statement or group of statements with a RUN statement. The output for each statement or group of statements is produced when the RUN statement is entered.

A RUN statement does not terminate the PROC ARIMA step but tells the procedure to execute the statements given so far. You can end PROC ARIMA by submitting a QUIT statement, a DATA step, another PROC step, or an ENDSAS statement.

The example in the preceding section illustrates the interactive use of ARIMA procedure statements. The complete PROC ARIMA program for that example is as follows:

``` SAS
proc arima data=test;
  identify var=sales nlag=24;
  run;
  identify var=sales(1);
  run;
  estimate p=1;
  run;
  estimate p=1 q=1;
  run;
  outlier;
  run;
  forecast lead=12 interval=month id=date out=results;
  run;
quit;
```

### General Notation for ARIMA Models

The order of an ARIMA (autoregressive integrated moving-average) model is usually denoted by the notation \(\text{ARIMA}(p,d,q)\), where

\[
\begin{align*}
  p & \quad \text{is the order of the autoregressive part} \\
  d & \quad \text{is the order of the differencing}
\end{align*}
\]
Chapter 8: The ARIMA Procedure

$q$ is the order of the moving-average process

If no differencing is done ($d = 0$), the models are usually referred to as $\text{ARMA}(p, q)$ models. The final model in the preceding example is an $\text{ARIMA}(1,1,1)$ model since the IDENTIFY statement specified $d = 1$, and the final ESTIMATE statement specified $p = 1$ and $q = 1$.

**Notation for Pure ARIMA Models**

Mathematically the pure ARIMA model is written as

$$W_t = \mu + \frac{\theta(B)}{\phi(B)} a_t$$

where

$t$ indexes time

$W_t$ is the response series $Y_t$ or a difference of the response series

$\mu$ is the mean term

$B$ is the backshift operator; that is, $BX_t = X_{t-1}$

$\phi(B)$ is the autoregressive operator, represented as a polynomial in the backshift operator: $\phi(B) = 1 - \phi_1 B - \ldots - \phi_p B^p$

$\theta(B)$ is the moving-average operator, represented as a polynomial in the backshift operator: $\theta(B) = 1 - \theta_1 B - \ldots - \theta_q B^q$

$a_t$ is the independent disturbance, also called the random error

The series $W_t$ is computed by the IDENTIFY statement and is the series processed by the ESTIMATE statement. Thus, $W_t$ is either the response series $Y_t$ or a difference of $Y_t$ specified by the differencing operators in the IDENTIFY statement.

For simple (nonseasonal) differencing, $W_t = (1-B)^d Y_t$. For seasonal differencing $W_t = (1-B)^d (1-B^s)^D Y_t$, where $d$ is the degree of nonseasonal differencing, $D$ is the degree of seasonal differencing, and $s$ is the length of the seasonal cycle.

For example, the mathematical form of the ARIMA(1,1,1) model estimated in the preceding example is

$$(1 - B)Y_t = \mu + \frac{(1 - \theta_1 B)}{(1 - \phi_1 B)} a_t$$

**Model Constant Term**

The ARIMA model can also be written as

$$\phi(B)(W_t - \mu) = \theta(B) a_t$$

or

$$\phi(B)W_t = \text{const} + \theta(B) a_t$$
where

\[ \text{const} = \phi(B) \mu = \mu - \phi_1 \mu - \phi_2 \mu - \ldots - \phi_p \mu \]

Thus, when an autoregressive operator and a mean term are both included in the model, the constant term for the model can be represented as \( \phi(B) \mu \). This value is printed with the label “Constant Estimate” in the ESTIMATE statement output.

### Notation for Transfer Function Models

The general ARIMA model with input series, also called the ARIMAX model, is written as

\[
W_t = \mu + \sum_i \frac{\omega_i(B)}{\delta_i(B)} B^{k_i} X_{i,t} + \frac{\theta(B)}{\phi(B)} a_t
\]

where

- \( X_{i,t} \) is the \( i \)th input time series or a difference of the \( i \)th input series at time \( t \)
- \( k_i \) is the pure time delay for the effect of the \( i \)th input series
- \( \omega_i(B) \) is the numerator polynomial of the transfer function for the \( i \)th input series
- \( \delta_i(B) \) is the denominator polynomial of the transfer function for the \( i \)th input series.

The model can also be written more compactly as

\[
W_t = \mu + \sum_i \Psi_i(B) X_{i,t} + n_t
\]

where

- \( \Psi_i(B) \) is the transfer function for the \( i \)th input series modeled as a ratio of the \( \omega \) and \( \delta \) polynomials: \( \Psi_i(B) = (\omega_i(B)/\delta_i(B)) B^{k_i} \)
- \( n_t \) is the noise series: \( n_t = (\theta(B)/\phi(B)) a_t \)

This model expresses the response series as a combination of past values of the random shocks and past values of other input series. The response series is also called the dependent series or output series. An input time series is also referred to as an independent series or a predictor series. Response variable, dependent variable, independent variable, or predictor variable are other terms often used.

### Notation for Factored Models

ARIMA models are sometimes expressed in a factored form. This means that the \( \phi \), \( \theta \), \( \omega \), or \( \delta \) polynomials are expressed as products of simpler polynomials. For example, you could express the pure ARIMA model as

\[
W_t = \mu + \frac{\theta_1(B)\theta_2(B)}{\phi_1(B)\phi_2(B)} a_t
\]

where \( \phi_1(B)\phi_2(B) = \phi(B) \) and \( \theta_1(B)\theta_2(B) = \theta(B) \).

When an ARIMA model is expressed in factored form, the order of the model is usually expressed by using a factored notation also. The order of an ARIMA model expressed as the product of two factors is denoted as \( \text{ARIMA}(p,d,q) \times (P,D,Q) \).
**Notation for Seasonal Models**

ARIMA models for time series with regular seasonal fluctuations often use differencing operators and autoregressive and moving-average parameters at lags that are multiples of the length of the seasonal cycle. When all the terms in an ARIMA model factor refer to lags that are a multiple of a constant $s$, the constant is factored out and suffixed to the ARIMA($p,d,q$) notation.

Thus, the general notation for the order of a seasonal ARIMA model with both seasonal and nonseasonal factors is ARIMA($p,d,q$)$\times$(P,D,Q)$_s$. The term ($p,d,q$) gives the order of the nonseasonal part of the ARIMA model; the term (P,D,Q)$_s$ gives the order of the seasonal part. The value of $s$ is the number of observations in a seasonal cycle: 12 for monthly series, 4 for quarterly series, 7 for daily series with day-of-week effects, and so forth.

For example, the notation ARIMA(0,1,2)$\times$(0,1,1)$_{12}$ describes a seasonal ARIMA model for monthly data with the following mathematical form:

$$(1 - B)(1 - B^{12})Y_t = \mu + (1 - \theta_{1,1}B - \theta_{1,2}B^2)(1 - \theta_{2,1}B^{12})a_t$$

**Stationarity**

The noise (or residual) series for an ARMA model must be *stationary*, which means that both the expected values of the series and its autocovariance function are independent of time.

The standard way to check for nonstationarity is to plot the series and its autocorrelation function. You can visually examine a graph of the series over time to see if it has a visible trend or if its variability changes noticeably over time. If the series is nonstationary, its autocorrelation function will usually decay slowly.

Another way of checking for stationarity is to use the stationarity tests described in the section “Stationarity Tests” on page 268.

Most time series are nonstationary and must be transformed to a stationary series before the ARIMA modeling process can proceed. If the series has a nonstationary variance, taking the log of the series can help. You can compute the log values in a DATA step and then analyze the log values with PROC ARIMA.

If the series has a trend over time, seasonality, or some other nonstationary pattern, the usual solution is to take the difference of the series from one period to the next and then analyze this differenced series. Sometimes a series might need to be differenced more than once or differenced at lags greater than one period. (If the trend or seasonal effects are very regular, the introduction of explanatory variables can be an appropriate alternative to differencing.)

**Differencing**

Differencing of the response series is specified with the VAR= option of the IDENTIFY statement by placing a list of differencing periods in parentheses after the variable name. For example, to take a simple first difference of the series SALES, use the statement

```
identify var=sales(1);
```
In this example, the change in SALES from one period to the next is analyzed.

A deterministic seasonal pattern also causes the series to be nonstationary, since the expected value of the series is not the same for all time periods but is higher or lower depending on the season. When the series has a seasonal pattern, you might want to difference the series at a lag that corresponds to the length of the seasonal cycle. For example, if SALES is a monthly series, the statement

```
identify var=sales(12);
```

takes a seasonal difference of SALES, so that the series analyzed is the change in SALES from its value in the same month one year ago.

To take a second difference, add another differencing period to the list. For example, the following statement takes the second difference of SALES:

```
identify var=sales(1,1);
```

That is, SALES is differenced once at lag 1 and then differenced again, also at lag 1. The statement

```
identify var=sales(2);
```

creates a 2-span difference—that is, current period SALES minus SALES from two periods ago. The statement

```
identify var=sales(1,12);
```

takes a second-order difference of SALES, so that the series analyzed is the difference between the current period-to-period change in SALES and the change 12 periods ago. You might want to do this if the series had both a trend over time and a seasonal pattern.

There is no limit to the order of differencing and the degree of lagging for each difference.

Differencing not only affects the series used for the IDENTIFY statement output but also applies to any following ESTIMATE and FORECAST statements. ESTIMATE statements fit ARMA models to the differenced series. FORECAST statements forecast the differences and automatically sum these differences back to undo the differencing operation specified by the IDENTIFY statement, thus producing the final forecast result.

Differencing of input series is specified by the CROSSCORR= option and works just like differencing of the response series. For example, the statement

```
identify var=y(1) crosscorr=(x1(1) x2(1));
```

takes the first difference of Y, the first difference of X1, and the first difference of X2. Whenever X1 and X2 are used in INPUT= options in following ESTIMATE statements, these names refer to the differenced series.
Subset, Seasonal, and Factored ARMA Models

The simplest way to specify an ARMA model is to give the order of the AR and MA parts with the P= and Q= options. When you do this, the model has parameters for the AR and MA parts for all lags through the order specified. However, you can control the form of the ARIMA model exactly as shown in the following section.

Subset Models

You can control which lags have parameters by specifying the P= or Q= option as a list of lags in parentheses. A model that includes parameters for only some lags is sometimes called a subset or additive model. For example, consider the following two ESTIMATE statements:

```plaintext
identify var=sales;
estimate p=4;
estimate p=(1 4);
```

Both specify AR(4) models, but the first has parameters for lags 1, 2, 3, and 4, while the second has parameters for lags 1 and 4, with the coefficients for lags 2 and 3 constrained to 0. The mathematical form of the autoregressive models produced by these two specifications is shown in Table 8.1.

<table>
<thead>
<tr>
<th>Option</th>
<th>Autoregressive Operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>P=4</td>
<td>(1 - \phi_1 B - \phi_2 B^2 - \phi_3 B^3 - \phi_4 B^4)</td>
</tr>
<tr>
<td>P=(1 4)</td>
<td>(1 - \phi_1 B - \phi_4 B^4)</td>
</tr>
</tbody>
</table>

Seasonal Models

One particularly useful kind of subset model is a seasonal model. When the response series has a seasonal pattern, the values of the series at the same time of year in previous years can be important for modeling the series. For example, if the series SALES is observed monthly, the statements

```plaintext
identify var=sales;
estimate p=(12);
```

model SALES as an average value plus some fraction of its deviation from this average value a year ago, plus a random error. Although this is an AR(12) model, it has only one autoregressive parameter.

Factored Models

A factored model (also referred to as a multiplicative model) represents the ARIMA model as a product of simpler ARIMA models. For example, you might model SALES as a combination of an AR(1) process that reflects short term dependencies and an AR(12) model that reflects the seasonal pattern.

It might seem that the way to do this is with the option P=(1 12), but the AR(1) process also operates in past years; you really need autoregressive parameters at lags 1, 12, and 13. You can specify a subset model
with separate parameters at these lags, or you can specify a factored model that represents the model as the product of an AR(1) model and an AR(12) model. Consider the following two ESTIMATE statements:

```
identify var=sales;
estimate p=(1 12 13);
estimate p=(1)(12);
```

The mathematical form of the autoregressive models produced by these two specifications are shown in Table 8.2.

<table>
<thead>
<tr>
<th>Option</th>
<th>Autoregressive Operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>P=(1 12 13)</td>
<td>((1 - \phi_1 B - \phi_{12} B^{12} - \phi_{13} B^{13}))</td>
</tr>
<tr>
<td>P=(1)(12)</td>
<td>((1 - \phi_1 B)(1 - \phi_{12} B^{12}))</td>
</tr>
</tbody>
</table>

Both models fit by these two ESTIMATE statements predict \(SALES\) from its values 1, 12, and 13 periods ago, but they use different parameterizations. The first model has three parameters, whose meanings may be hard to interpret.

The factored specification P=(1)(12) represents the model as the product of two different AR models. It has only two parameters: one that corresponds to recent effects and one that represents seasonal effects. Thus the factored model is more parsimonious, and its parameter estimates are more clearly interpretable.

### Input Variables and Regression with ARMA Errors

In addition to past values of the response series and past errors, you can also model the response series using the current and past values of other series, called input series.

Several different names are used to describe ARIMA models with input series. *Transfer function model*, *intervention model*, *interrupted time series model*, *regression model with ARMA errors*, *Box-Tiao model*, and *ARIMAX model* are all different names for ARIMA models with input series. Pankratz (1991) refers to these models as *dynamic regression* models.

#### Using Input Series

To use input series, list the input series in a CROSSCORR= option on the IDENTIFY statement and specify how they enter the model with an INPUT= option on the ESTIMATE statement. For example, you might use a series called \(PRICE\) to help model \(SALES\), as shown in the following statements:

```
proc arima data=a;
    identify var=sales crosscorr=price;
    estimate input=price;
run;
```

This example performs a simple linear regression of \(SALES\) on \(PRICE\); it produces the same results as PROC REG or another SAS regression procedure. The mathematical form of the model estimated by these statements is

\[ Y_t = \mu + \omega_0 X_t + a_t \]
The parameter estimates table for this example (using simulated data) is shown in Figure 8.20. The intercept parameter is labeled MU. The regression coefficient for PRICE is labeled NUM1. (See the section “Naming of Model Parameters” on page 277 for information about how parameters for input series are named.)

**Figure 8.20 Parameter Estimates Table for Regression Model**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>t Value</th>
<th>Approx Pr &gt;</th>
<th>Lag</th>
<th>Variable</th>
<th>Shift</th>
</tr>
</thead>
<tbody>
<tr>
<td>MU</td>
<td>199.83602</td>
<td>2.99463</td>
<td>66.73</td>
<td>&lt;.0001</td>
<td>0</td>
<td>sales</td>
<td>0</td>
</tr>
<tr>
<td>NUM1</td>
<td>-9.99299</td>
<td>0.02885</td>
<td>-346.38</td>
<td>&lt;.0001</td>
<td>0</td>
<td>price</td>
<td>0</td>
</tr>
</tbody>
</table>

Any number of input variables can be used in a model. For example, the following statements fit a multiple regression of SALES on PRICE and INCOME:

```r
proc arima data=a;
  identify var=sales crosscorr=(price income);
  estimate input=(price income); 
run;
```

The mathematical form of the regression model estimated by these statements is

\[ Y_t = \mu + \omega_1 X_{1,t} + \omega_2 X_{2,t} + \epsilon_t \]

**Lagging and Differencing Input Series**

You can also difference and lag the input series. For example, the following statements regress the change in SALES on the change in PRICE lagged by one period. The difference of PRICE is specified with the CROSSCORR= option and the lag of the change in PRICE is specified by the 1 $ in the INPUT= option.

```r
proc arima data=a;
  identify var=sales(1) crosscorr=price(1);
  estimate input=( 1 $ price );
run;
```

These statements estimate the model

\[ (1 - B) Y_t = \mu + \omega_0 (1 - B) X_{t-1} + \epsilon_t \]

**Regression with ARMA Errors**

You can combine input series with ARMA models for the errors. For example, the following statements regress SALES on INCOME and PRICE but with the error term of the regression model (called the noise series in ARIMA modeling terminology) assumed to be an ARMA(1,1) process.
proc arima data=a;
   identify var=sales crosscorr=(price income);
   estimate p=1 q=1 input=(price income);
run;

These statements estimate the model

\[ Y_t = \mu + \omega_1 X_{1,t} + \omega_2 X_{2,t} + \frac{(1 - \theta_1 B)}{1 - \phi_1 B} \alpha_t \]

**Stationarity and Input Series**

Note that the requirement of stationarity applies to the noise series. If there are no input variables, the response series (after differencing and minus the mean term) and the noise series are the same. However, if there are inputs, the noise series is the residual after the effect of the inputs is removed.

There is no requirement that the input series be stationary. If the inputs are nonstationary, the response series will be nonstationary, even though the noise process might be stationary.

When nonstationary input series are used, you can fit the input variables first with no ARMA model for the errors and then consider the stationarity of the residuals before identifying an ARMA model for the noise part.

**Identifying Regression Models with ARMA Errors**

Previous sections described the ARIMA modeling identification process that uses the autocorrelation function plots produced by the IDENTIFY statement. This identification process does not apply when the response series depends on input variables. This is because it is the noise process for which you need to identify an ARIMA model, and when input series are involved the response series adjusted for the mean is no longer an estimate of the noise series.

However, if the input series are independent of the noise series, you can use the residuals from the regression model as an estimate of the noise series, then apply the ARIMA modeling identification process to this residual series. This assumes that the noise process is stationary.

The PLOT option in the ESTIMATE statement produces similar plots for the model residuals as the IDENTIFY statement produces for the response series. The PLOT option prints an autocorrelation function plot, an inverse autocorrelation function plot, and a partial autocorrelation function plot for the residual series. Note that these residual correlation plots are produced by default.

The following statements show how the PLOT option is used to identify the ARMA(1,1) model for the noise process used in the preceding example of regression with ARMA errors:

proc arima data=a;
   identify var=sales crosscorr=(price income) noprint;
   estimate input=(price income) plot;
run;
   estimate p=1 q=1 input=(price income);
run;
In this example, the IDENTIFY statement includes the NOPRINT option since the autocorrelation plots for the response series are not useful when you know that the response series depends on input series.

The first ESTIMATE statement fits the regression model with no model for the noise process. The PLOT option produces plots of the autocorrelation function, inverse autocorrelation function, and partial autocorrelation function for the residual series of the regression on PRICE and INCOME.

By examining the PLOT option output for the residual series, you verify that the residual series is stationary and identify an ARMA(1,1) model for the noise process. The second ESTIMATE statement fits the final model.

Although this discussion addresses regression models, the same remarks apply to identifying an ARIMA model for the noise process in models that include input series with complex transfer functions.

---

**Intervention Models and Interrupted Time Series**

One special kind of ARIMA model with input series is called an *intervention model* or *interrupted time series* model. In an intervention model, the input series is an indicator variable that contains discrete values that flag the occurrence of an event affecting the response series. This event is an intervention in or an interruption of the normal evolution of the response time series, which, in the absence of the intervention, is usually assumed to be a pure ARIMA process.

Intervention models can be used both to model and forecast the response series and also to analyze the impact of the intervention. When the focus is on estimating the effect of the intervention, the process is often called *intervention analysis* or *interrupted time series analysis*.

**Impulse Interventions**

The intervention can be a one-time event. For example, you might want to study the effect of a short-term advertising campaign on the sales of a product. In this case, the input variable has the value of 1 for the period during which the advertising campaign took place and the value 0 for all other periods. Intervention variables of this kind are sometimes called *impulse functions* or *pulse functions*.

Suppose that SALES is a monthly series, and a special advertising effort was made during the month of March 1992. The following statements estimate the effect of this intervention by assuming an ARMA(1,1) model for SALES. The model is specified just like the regression model, but the intervention variable AD is constructed in the DATA step as a zero-one indicator for the month of the advertising effort.

```plaintext
data a;
  set a;
  ad = (date = '1mar1992'd);
run;

proc arima data=a;
  identify var=sales crosscorr=ad;
  estimate p=1 q=1 input=ad;
run;
```
Continuing Interventions

Other interventions can be continuing, in which case the input variable flags periods before and after the intervention. For example, you might want to study the effect of a change in tax rates on some economic measure. Another example is a study of the effect of a change in speed limits on the rate of traffic fatalities. In this case, the input variable has the value 1 after the new speed limit went into effect and the value 0 before. Intervention variables of this kind are called *step functions*.

Another example is the effect of news on product demand. Suppose it was reported in July 1996 that consumption of the product prevents heart disease (or causes cancer), and *SALES* is consistently higher (or lower) thereafter. The following statements model the effect of this news intervention:

```r
data a;
set a;
  news = (date >= '1jul1996'd);
run;
proc arima data=a;
  identify var=sales crosscorr=news;
  estimate p=1 q=1 input=news;
run;
```

Interaction Effects

You can include any number of intervention variables in the model. Intervention variables can have any pattern—impulse and continuing interventions are just two possible cases. You can mix discrete valued intervention variables and continuous regressor variables in the same model.

You can also form interaction effects by multiplying input variables and including the product variable as another input. Indeed, as long as the dependent measure is continuous and forms a regular time series, you can use PROC ARIMA to fit any general linear model in conjunction with an ARMA model for the error process by using input variables that correspond to the columns of the design matrix of the linear model.

Rational Transfer Functions and Distributed Lag Models

How an input series enters the model is called its *transfer function*. Thus, ARIMA models with input series are sometimes referred to as transfer function models.

In the preceding regression and intervention model examples, the transfer function is a single scale parameter. However, you can also specify complex transfer functions composed of numerator and denominator polynomials in the backshift operator. These transfer functions operate on the input series in the same way that the ARMA specification operates on the error term.

Numerator Factors

For example, suppose you want to model the effect of *PRICE* on *SALES* as taking place gradually with the impact distributed over several past lags of *PRICE*. This is illustrated by the following statements:
These statements estimate the model

\[ Y_t = \mu + (\omega_0 - \omega_1 B - \omega_2 B^2 - \omega_3 B^3) X_t + a_t \]

This example models the effect of \texttt{PRICE} on \texttt{SALES} as a linear function of the current and three most recent values of \texttt{PRICE}. It is equivalent to a multiple linear regression of \texttt{SALES} on \texttt{PRICE}, \texttt{LAG(PRICE)}, \texttt{LAG2(PRICE)}, and \texttt{LAG3(PRICE)}.

This is an example of a transfer function with one \textit{numerator factor}. The numerator factors for a transfer function for an input series are like the MA part of the ARMA model for the noise series.

**Denominator Factors**

You can also use transfer functions with \textit{denominator factors}. The denominator factors for a transfer function for an input series are like the AR part of the ARMA model for the noise series. Denominator factors introduce exponentially weighted, infinite distributed lags into the transfer function.

To specify transfer functions with denominator factors, place the denominator factors after a slash (/) in the \texttt{INPUT=} option. For example, the following statements estimate the \texttt{PRICE} effect as an infinite distributed lag model with exponentially declining weights:

```
proc arima data=a;
    identify var=sales crosscorr=price;
    estimate input=( / (1) price );
run;
```

The transfer function specified by these statements is as follows:

\[ \frac{\omega_0}{(1 - \delta_1 B)} X_t \]

This transfer function also can be written in the following equivalent form:

\[ \omega_0 \left( 1 + \sum_{i=1}^{\infty} \delta_1^i B^i \right) X_t \]

This transfer function can be used with intervention inputs. When it is used with a pulse function input, the result is an intervention effect that dies out gradually over time. When it is used with a step function input, the result is an intervention effect that increases gradually to a limiting value.
Rational Transfer Functions

By combining various numerator and denominator factors in the INPUT= option, you can specify rational transfer functions of any complexity. To specify an input with a general rational transfer function of the form

\[
\frac{\omega(B)}{\delta(B)} B^k X_t
\]

use an INPUT= option in the ESTIMATE statement of the form

\[
\text{input}=(k \delta-lags) / (\omega-lags) x)
\]

See the section “Specifying Inputs and Transfer Functions” on page 274 for more information.

Identifying Transfer Function Models

The CROSSCORR= option of the IDENTIFY statement prints sample cross-correlation functions that show the correlation between the response series and the input series at different lags. The sample cross-correlation function can be used to help identify the form of the transfer function appropriate for an input series. See textbooks on time series analysis for information about using cross-correlation functions to identify transfer function models.

For the cross-correlation function to be meaningful, the input and response series must be filtered with a prewhitening model for the input series. See the section “Prewhitening” on page 268 for more information about this issue.

Forecasting with Input Variables

To forecast a response series by using an ARIMA model with inputs, you need values of the input series for the forecast periods. You can supply values for the input variables for the forecast periods in the DATA= data set, or you can have PROC ARIMA forecast the input variables.

If you do not have future values of the input variables in the input data set used by the FORECAST statement, the input series must be forecast before the ARIMA procedure can forecast the response series. If you fit an ARIMA model to each of the input series for which you need forecasts before fitting the model for the response series, the FORECAST statement automatically uses the ARIMA models for the input series to generate the needed forecasts of the inputs.

For example, suppose you want to forecast SALES for the next 12 months. In this example, the change in SALES is predicted as a function of the change in PRICE, plus an ARMA(1,1) noise process. To forecast SALES by using PRICE as an input, you also need to fit an ARIMA model for PRICE.

The following statements fit an AR(2) model to the change in PRICE before fitting and forecasting the model for SALES. The FORECAST statement automatically forecasts PRICE using this AR(2) model to get the future inputs needed to produce the forecast of SALES.
proc arima data=a;
  identify var=price(1);
  estimate p=2;
  identify var=sales(1) crosscorr=price(1);
  estimate p=1 q=1 input=price;
  forecast lead=12 interval=month id=date out=results;
run;

Fitting a model to the input series is also important for identifying transfer functions. (See the section “Prewhitening” on page 268 for more information.)

Input values from the DATA= data set and input values forecast by PROC ARIMA can be combined. For example, a model for SALES might have three input series: PRICE, INCOME, and TAXRATE. For the forecast, you assume that the tax rate will be unchanged. You have a forecast for INCOME from another source but only for the first few periods of the SALES forecast you want to make. You have no future values for PRICE, which needs to be forecast as in the preceding example.

In this situation, you include observations in the input data set for all forecast periods, with SALES and PRICE set to a missing value, with TAXRATE set to its last actual value, and with INCOME set to forecast values for the periods you have forecasts for and set to missing values for later periods. In the PROC ARIMA step, you estimate ARIMA models for PRICE and INCOME before you estimate the model for SALES, as shown in the following statements:

proc arima data=a;
  identify var=price(1);
  estimate p=2;
  identify var=income(1);
  estimate p=2;
  identify var=sales(1) crosscorr=(price(1) income(1) taxrate);
  estimate p=1 q=1 input=(price income taxrate);
  forecast lead=12 interval=month id=date out=results;
run;

In forecasting SALES, the ARIMA procedure uses as inputs the value of PRICE forecast by its ARIMA model, the value of TAXRATE found in the DATA= data set, and the value of INCOME found in the DATA= data set, or, when the INCOME variable is missing, the value of INCOME forecast by its ARIMA model. (Because SALES is missing for future time periods, the estimation of model parameters is not affected by the forecast values for PRICE, INCOME, or TAXRATE.)

---

Data Requirements

PROC ARIMA can handle time series of moderate size; there should be at least 30 observations. With fewer than 30 observations, the parameter estimates might be poor. With thousands of observations, the method requires considerable computer time and memory.
Syntax: ARIMA Procedure

The ARIMA procedure uses the following statements:

```plaintext
PROC ARIMA options;
   BY variables;
   IDENTIFY VAR=variable options;
   ESTIMATE options;
   OUTLIER options;
   FORECAST options;
```

The PROC ARIMA and IDENTIFY statements are required.

Functional Summary

The statements and options that control the ARIMA procedure are summarized in Table 8.3.

<table>
<thead>
<tr>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Data Set Options</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>specify the input data set</td>
<td>PROC ARIMA</td>
<td>DATA=</td>
</tr>
<tr>
<td>specify the output data set</td>
<td>PROC ARIMA</td>
<td>OUT=</td>
</tr>
<tr>
<td>include only forecasts in the output data set</td>
<td>FORECAST</td>
<td>NOOUTALL</td>
</tr>
<tr>
<td>write autocovariances to output data set</td>
<td>IDENTIFY</td>
<td>OUTCOV=</td>
</tr>
<tr>
<td>write parameter estimates to an output data set</td>
<td>ESTIMATE</td>
<td>OUTSTAT=</td>
</tr>
<tr>
<td>write correlation of parameter estimates</td>
<td>ESTIMATE</td>
<td>OUTCORR</td>
</tr>
<tr>
<td>write covariance of parameter estimates</td>
<td>ESTIMATE</td>
<td>OUTCOV</td>
</tr>
<tr>
<td>write estimated model to an output data set</td>
<td>ESTIMATE</td>
<td>OUTMODEL=</td>
</tr>
<tr>
<td>write statistics of fit to an output data set</td>
<td>ESTIMATE</td>
<td>OUTSTAT=</td>
</tr>
</tbody>
</table>

**Options for Identifying the Series**

difference time series and plot autocorrelations IDENTIFY VAR=
specify response series and differencing IDENTIFY CROSSCORR=
specify and cross-correlate input series IDENTIFY CENTER
center data by subtracting the mean IDENTIFY NOMISS
delete previous models and start IDENTIFY CLEAR
specify the significance level for tests IDENTIFY ALPHA=
perform tentative ARMA order identification by using the ESACF method IDENTIFY ESACF
perform tentative ARMA order identification by using the MINIC method IDENTIFY MINIC
### Table 8.3  continued

<table>
<thead>
<tr>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>perform tentative ARMA order identification by using the SCAN method</td>
<td>IDENTIFY</td>
<td>SCAN</td>
</tr>
<tr>
<td>specify the range of autoregressive model orders for estimating the error series for the MINIC method</td>
<td>IDENTIFY</td>
<td>PERROR=</td>
</tr>
<tr>
<td>determine the AR dimension of the SCAN, ESACF, and MINIC tables</td>
<td>IDENTIFY</td>
<td>P=</td>
</tr>
<tr>
<td>determine the MA dimension of the SCAN, ESACF, and MINIC tables</td>
<td>IDENTIFY</td>
<td>Q=</td>
</tr>
<tr>
<td>perform stationarity tests</td>
<td>IDENTIFY</td>
<td>STATIONARITY=</td>
</tr>
<tr>
<td>selection of white noise test statistic in the presence of missing values</td>
<td>IDENTIFY</td>
<td>WHITENOISE=</td>
</tr>
</tbody>
</table>

#### Options for Defining and Estimating the Model

<table>
<thead>
<tr>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>specify and estimate ARIMA models</td>
<td>ESTIMATE</td>
<td></td>
</tr>
<tr>
<td>specify autoregressive part of model</td>
<td>ESTIMATE</td>
<td>P=</td>
</tr>
<tr>
<td>specify moving-average part of model</td>
<td>ESTIMATE</td>
<td>Q=</td>
</tr>
<tr>
<td>specify input variables and transfer functions</td>
<td>ESTIMATE</td>
<td>INPUT=</td>
</tr>
<tr>
<td>drop mean term from the model</td>
<td>ESTIMATE</td>
<td>NOINT</td>
</tr>
<tr>
<td>specify the estimation method</td>
<td>ESTIMATE</td>
<td>METHOD=</td>
</tr>
<tr>
<td>use alternative form for transfer functions</td>
<td>ESTIMATE</td>
<td>ALTPARM</td>
</tr>
<tr>
<td>suppress degrees-of-freedom correction in variance estimates</td>
<td>ESTIMATE</td>
<td>NODF</td>
</tr>
<tr>
<td>selection of white noise test statistic in the presence of missing values</td>
<td>ESTIMATE</td>
<td>WHITENOISE=</td>
</tr>
</tbody>
</table>

#### Options for Outlier Detection

<table>
<thead>
<tr>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>specify the significance level for tests</td>
<td>OUTLIER</td>
<td>ALPHA=</td>
</tr>
<tr>
<td>identify detected outliers with variable</td>
<td>OUTLIER</td>
<td>ID=</td>
</tr>
<tr>
<td>limit the number of outliers</td>
<td>OUTLIER</td>
<td>MAXNUM=</td>
</tr>
<tr>
<td>limit the number of outliers to a percentage of the series</td>
<td>OUTLIER</td>
<td>MAXPCT=</td>
</tr>
<tr>
<td>specify the variance estimator used for testing</td>
<td>OUTLIER</td>
<td>SIGMA=</td>
</tr>
<tr>
<td>specify the type of level shifts</td>
<td>OUTLIER</td>
<td>TYPE=</td>
</tr>
</tbody>
</table>

#### Printing Control Options

<table>
<thead>
<tr>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>limit number of lags shown in correlation plots</td>
<td>IDENTIFY</td>
<td>NLAG=</td>
</tr>
<tr>
<td>suppress printed output for identification</td>
<td>IDENTIFY</td>
<td>NOPRINT</td>
</tr>
<tr>
<td>plot autocorrelation functions of the residuals</td>
<td>ESTIMATE</td>
<td>PLOT</td>
</tr>
<tr>
<td>print log-likelihood around the estimates</td>
<td>ESTIMATE</td>
<td>GRID</td>
</tr>
<tr>
<td>control spacing for GRID option</td>
<td>ESTIMATE</td>
<td>GRIDVAL=</td>
</tr>
<tr>
<td>print details of the iterative estimation process</td>
<td>ESTIMATE</td>
<td>PRINTALL</td>
</tr>
<tr>
<td>suppress printed output for estimation</td>
<td>ESTIMATE</td>
<td>NOPRINT</td>
</tr>
</tbody>
</table>
### Table 8.3  continued

<table>
<thead>
<tr>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>suppress printing of the forecast values</td>
<td>FORECAST</td>
<td>NOPRINT</td>
</tr>
<tr>
<td>print the one-step forecasts and residuals</td>
<td>FORECAST</td>
<td>PRINTALL</td>
</tr>
</tbody>
</table>

#### Plotting Control Options
request plots associated with model identification, residual analysis, and forecasting

#### Options to Specify Parameter Values
- specify autoregressive starting values
  
- specify moving-average starting values
  
- specify a starting value for the mean parameter
  
- specify starting values for transfer functions

#### Options to Control the Iterative Estimation Process
- specify convergence criterion
- specify the maximum number of iterations
- specify criterion for checking for singularity
- omit initial observations from objective
- specify perturbation for numerical derivatives
- omit stationarity and invertibility checks
- use preliminary estimates as starting values for ML and ULS

#### Options for Forecasting
- forecast the response series
- specify how many periods to forecast
- specify the ID variable
- specify the periodicity of the series
- specify size of forecast confidence limits
- start forecasting before end of the input data
- specify the variance term used to compute forecast standard errors and confidence limits
- control the alignment of SAS date values

#### BY Groups
- specify BY group processing
**PROC ARIMA Statement**

PROC ARIMA options;

The following options can be used in the PROC ARIMA statement.

**DATA=SAS-data-set**

specifies the name of the SAS data set that contains the time series. If different DATA= specifications appear in the PROC ARIMA and IDENTIFY statements, the one in the IDENTIFY statement is used. If the DATA= option is not specified in either the PROC ARIMA or IDENTIFY statement, the most recently created SAS data set is used.

**PLOTS< (global-plot-options) > < = plot-request < (options) > >**

controls the plots produced through ODS Graphics. When you specify only one plot request, you can omit the parentheses around the plot request.

Here are some examples:

```plaintext
plots=none
plots=all
plots(unpack)=series(corr crosscorr)
plots(only)=(series(corr crosscorr) residual(normal smooth))
```

**Global Plot Options:**

The **global-plot-options** apply to all relevant plots generated by the ARIMA procedure. The following **global-plot-options** are supported:

**ONLY**

suppresses the default plots. Only the plots specifically requested are produced.

**UNPACK**

displays each graph separately. (By default, some graphs can appear together in a single panel.)

**Specific Plot Options**

The following list describes the specific plots and their options.

**ALL**

produces all plots appropriate for the particular analysis.

**NONE**

suppresses all plots.

**SERIES(< series-plot-options >)**

produces plots associated with the identification stage of the modeling. The panel plots corresponding to the CORR and CROSSCORR options are produced by default. The following **series-plot-options** are available:
ACF
produces the plot of autocorrelations.

ALL
produces all the plots associated with the identification stage.

CORR
produces a panel of plots that are useful in the trend and correlation analysis of the series. The panel consists of the following:

- the time series plot
- the series-autocorrelation plot
- the series-partial-autocorrelation plot
- the series-inverse-autocorrelation plot

CROSSCORR
produces panels of cross-correlation plots.

IACF
produces the plot of inverse-autocorrelations.

PACF
produces the plot of partial-autocorrelations.

RESIDUAL(< residual-plot-options >)
produces the residuals plots. The residual correlation and normality diagnostic panels are produced by default. The following residual-plot-options are available:

ACF
produces the plot of residual autocorrelations.

ALL
produces all the residual diagnostics plots appropriate for the particular analysis.

CORR
produces a summary panel of the residual correlation diagnostics that consists of the following:

- the residual-autocorrelation plot
- the residual-partial-autocorrelation plot
- the residual-inverse-autocorrelation plot
- a plot of Ljung-Box white-noise test p-values at different lags

HIST
produces the histogram of the residuals.
IACF
produces the plot of residual inverse-autocorrelations.

NORMAL
produces a summary panel of the residual normality diagnostics that consists of the following:
- histogram of the residuals
- normal quantile plot of the residuals

PACF
produces the plot of residual partial-autocorrelations.

QQ
produces the normal quantile plot of the residuals.

SMOOTH
produces a scatter plot of the residuals against time, which has an overlaid smooth fit.

WN
produces the plot of Ljung-Box white-noise test p-values at different lags.

FORECAST(<forecast-plot-options>)
produces the forecast plots in the forecasting stage. The forecast-only plot that shows the multistep forecasts in the forecast region is produced by default.

The following forecast-plot-options are available:

ALL
produces the forecast-only plot as well as the forecast plot.

FORECAST
produces a plot that shows the one-step-ahead forecasts as well as the multistep-ahead forecasts.

FORECASTONLY
produces a plot that shows only the multistep-ahead forecasts in the forecast region.

OUT=SAS-data-set
specifies a SAS data set to which the forecasts are output. If different OUT= specifications appear in the PROC ARIMA and FORECAST statements, the one in the FORECAST statement is used.

BY Statement

BY variables;

A BY statement can be used in the ARIMA procedure to process a data set in groups of observations defined by the BY variables. Note that all IDENTIFY, ESTIMATE, and FORECAST statements specified are applied to all BY groups.
Because of the need to make data-based model selections, BY-group processing is not usually done with PROC ARIMA. You usually want to use different models for the different series contained in different BY groups, and the PROC ARIMA BY statement does not let you do this.

Using a BY statement imposes certain restrictions. The BY statement must appear before the first RUN statement. If a BY statement is used, the input data must come from the data set specified in the PROC statement; that is, no input data sets can be specified in IDENTIFY statements.

When a BY statement is used with PROC ARIMA, interactive processing applies only to the first BY group. Once the end of the PROC ARIMA step is reached, all ARIMA statements specified are executed again for each of the remaining BY groups in the input data set.

### IDENTIFY Statement

```plaintext
IDENTIFY VAR=variable options ;
```

The IDENTIFY statement specifies the time series to be modeled, differences the series if desired, and computes statistics to help identify models to fit. Use an IDENTIFY statement for each time series that you want to model.

If other time series are to be used as inputs in a subsequent ESTIMATE statement, they must be listed in a CROSSCORR= list in the IDENTIFY statement.

The following options are used in the IDENTIFY statement. The VAR= option is required.

- **ALPHA=**significance-level
  
  The ALPHA= option specifies the significance level for tests in the IDENTIFY statement. The default is 0.05.

- **CENTER**
  
  centers each time series by subtracting its sample mean. The analysis is done on the centered data. Later, when forecasts are generated, the mean is added back. Note that centering is done after differencing. The CENTER option is normally used in conjunction with the NOCONSTANT option of the ESTIMATE statement.

- **CLEAR**
  
  deletes all old models. This option is useful when you want to delete old models so that the input variables are not prewhitened. (See the section “Prewhitening” on page 268 for more information.)

- **CROSSCORR=**variable (d11, d12, . . . , d1k )

- **CROSSCORR=** (variable (d11, d12, . . . , d1k )... variable (d21, d22, . . . , d2k ))

  names the variables cross-correlated with the response variable given by the VAR= specification.

  Each variable name can be followed by a list of differencing lags in parentheses, the same as for the VAR= specification. If differencing is specified for a variable in the CROSSCORR= list, the differenced series is cross-correlated with the VAR= option series, and the differenced series is used when the ESTIMATE statement INPUT= option refers to the variable.
**DATA=SAS-data-set**
specifies the input SAS data set that contains the time series. If the DATA= option is omitted, the DATA= data set specified in the PROC ARIMA statement is used; if the DATA= option is omitted from the PROC ARIMA statement as well, the most recently created data set is used.

**ESACF**
computes the extended sample autocorrelation function and uses these estimates to tentatively identify the autoregressive and moving-average orders of mixed models.

The ESACF option generates two tables. The first table displays extended sample autocorrelation estimates, and the second table displays probability values that can be used to test the significance of these estimates. The P=(p_{min} : p_{max}) and Q=(q_{min} : q_{max}) options determine the size of the table.

The autoregressive and moving-average orders are tentatively identified by finding a triangular pattern in which all values are insignificant. The ARIMA procedure finds these patterns based on the IDENTIFY statement ALPHA= option and displays possible recommendations for the orders.

The following code generates an ESACF table with dimensions of p=(0:7) and q=(0:8).

```proc arima data=test;
   identify var=x esacf p=(0:7) q=(0:8);
run;
```

See the section “The ESACF Method” on page 263 for more information.

**MINIC**
uses information criteria or penalty functions to provide tentative ARMA order identification. The MINIC option generates a table that contains the computed information criterion associated with various ARMA model orders. The PERROR=(p_{e,min} : p_{e,max}) option determines the range of the autoregressive model orders used to estimate the error series. The P=(p_{min} : p_{max}) and Q=(q_{min} : q_{max}) options determine the size of the table. The ARMA orders are tentatively identified by those orders that minimize the information criterion.

The following statements generate a MINIC table with default dimensions of p=(0:5) and q=(0:5) and with the error series estimated by an autoregressive model with an order, p_{e}, that minimizes the AIC in the range from 8 to 11.

```proc arima data=test;
   identify var=x minic perror=(8:11);
run;
```

See the section “The MINIC Method” on page 265 for more information.

**NLAG=number**
indicates the number of lags to consider in computing the autocorrelations and cross-correlations. To obtain preliminary estimates of an ARIMA(p, d, q) model, the NLAG= value must be at least p + q + d. The number of observations must be greater than or equal to the NLAG= value. The default value for NLAG= is 24 or one-fourth the number of observations, whichever is less. Even though the NLAG= value is specified, the NLAG= value can be changed according to the data set.
IDENTIFY Statement ◆ 253

NOMISS
uses only the first continuous sequence of data with no missing values. By default, all observations are used.

NOPRINT
suppresses the normal printout (including the correlation plots) generated by the IDENTIFY statement.

OUTCOV=SAS-data-set
writes the autocovariances, autocorrelations, inverse autocorrelations, partial autocorrelations, and cross covariances to an output SAS data set. If the OUTCOV= option is not specified, no covariance output data set is created. See the section “OUTCOV= Data Set” on page 283 for more information.

P=(p_{min} : p_{max})
see the ESACF, MINIC, and SCAN options for details.

PERROR=(p_{\epsilon,min} : p_{\epsilon,max})
determines the range of the autoregressive model orders used to estimate the error series in MINIC, a tentative ARMA order identification method. See the section “The MINIC Method” on page 265 for more information. By default p_{\epsilon,min} is set to p_{max} and p_{\epsilon,max} is set to p_{max} + q_{max}, where p_{max} and q_{max} are the maximum settings of the P= and Q= options on the IDENTIFY statement.

Q=(q_{min} : q_{max})
see the ESACF, MINIC, and SCAN options for details.

SCAN
computes estimates of the squared canonical correlations and uses these estimates to tentatively identify the autoregressive and moving-average orders of mixed models.

The SCAN option generates two tables. The first table displays squared canonical correlation estimates, and the second table displays probability values that can be used to test the significance of these estimates. The P=(p_{min} : p_{max}) and Q=(q_{min} : q_{max}) options determine the size of each table.

The autoregressive and moving-average orders are tentatively identified by finding a rectangular pattern in which all values are insignificant. The ARIMA procedure finds these patterns based on the IDENTIFY statement ALPHA= option and displays possible recommendations for the orders.

The following code generates a SCAN table with default dimensions of p=(0:5) and q=(0:5). The recommended orders are based on a significance level of 0.1.

```
proc arima data=test;
   identify var=x scan alpha=0.1;
run;
```

See the section “The SCAN Method” on page 266 for more information.

STATIONARITY=
performs stationarity tests. Stationarity tests can be used to determine whether differencing terms should be included in the model specification. In each stationarity test, the autoregressive orders can be specified by a range, test= ar_{max}, or as a list of values, test= (ar_1, ..., ar_n), where test is ADF, PP, or RW. The default is (0,1,2).

See the section “Stationarity Tests” on page 268 for more information.
STATIONARITY=(ADF= AR orders DLAG= s )

STATIONARITY=(DICKEY= AR orders DLAG= s )

performs augmented Dickey-Fuller tests. If the DLAG=s option is specified with s is greater than one, seasonal Dickey-Fuller tests are performed. The maximum allowable value of s is 12. The default value of s is 1. The following code performs augmented Dickey-Fuller tests with autoregressive orders 2 and 5.

```sql
proc arima data=test;
   identify var=x stationarity=(adf=(2,5));
run;
```

STATIONARITY=(PP= AR orders )

STATIONARITY=(PHILLIPS= AR orders )

performs Phillips-Perron tests. The following statements perform augmented Phillips-Perron tests with autoregressive orders ranging from 0 to 6.

```sql
proc arima data=test;
   identify var=x stationarity=(pp=6);
run;
```

STATIONARITY=(RW=AR orders )

STATIONARITY=(RANDOMWALK=AR orders )

performs random-walk-with-drift tests. The following statements perform random-walk-with-drift tests with autoregressive orders ranging from 0 to 2.

```sql
proc arima data=test;
   identify var=x stationarity=(rw);
run;
```

VAR=variable

names the variable that contains the time series to analyze. The VAR= option is required.

A list of differencing lags can be placed in parentheses after the variable name to request that the series be differenced at these lags. For example, VAR=X(1) takes the first differences of X. VAR=X(1,1) requests that X be differenced twice, both times with lag 1, producing a second difference series, which is 

\[(X_t - X_{t-1}) - (X_{t-1} - X_{t-2}) = X_t - 2X_{t-1} + X_{t-2}.\]

VAR=X(2) differences X once at lag two \((X_t - X_{t-2})\).

If differencing is specified, it is the differenced series that is processed by any subsequent ESTIMATE statement.

WHITENOISE=ST | IGNOREMISS

specifies the type of test statistic that is used in the white noise test of the series when the series contains missing values. If WHITENOISE=IGNOREMISS, the standard Ljung-Box test statistic is used. If WHITENOISE=ST, a modification of this statistic suggested by Stoffer and Toloi (1992) is used. The default is WHITENOISE=ST.
ESTIMATE Statement

< label: > ESTIMATE options ;

The ESTIMATE statement specifies an ARMA model or transfer function model for the response variable specified in the previous IDENTIFY statement, and produces estimates of its parameters. The ESTIMATE statement also prints diagnostic information by which to check the model. The label in the ESTIMATE statement is optional. Include an ESTIMATE statement for each model that you want to estimate.

Options used in the ESTIMATE statement are described in the following sections.

Options for Defining the Model and Controlling Diagnostic Statistics

The following options are used to define the model to be estimated and to control the output that is printed.

ALTPARM
specifies the alternative parameterization of the overall scale of transfer functions in the model. See the section “Alternative Model Parameterization” on page 275 for details.

INPUT= variable
INPUT= ( transfer-function variable ...) specifies input variables and their transfer functions.

The variables used on the INPUT= option must be included in the CROSSCORR= list in the previous IDENTIFY statement. If any differencing is specified in the CROSSCORR= list, then the differenced series is used as the input to the transfer function.

The transfer function specification for an input variable is optional. If no transfer function is specified, the input variable enters the model as a simple regressor. If specified, the transfer function specification has the following syntax:

$S(L_{1,1}, L_{1,2}, \ldots)(L_{2,1}, \ldots) / (L_{j,1}, \ldots) \ldots$

Here, $S$ is a shift or lag of the input variable, the terms before the slash (/) are numerator factors, and the terms after the slash (/) are denominator factors of the transfer function. All three parts are optional. See the section “Specifying Inputs and Transfer Functions” on page 274 for details.

METHOD= value
specifies the estimation method to use. METHOD=ML specifies the maximum likelihood method. METHOD=ULS specifies the unconditional least squares method. METHOD=CLS specifies the conditional least squares method. METHOD=CLS is the default. See the section “Estimation Details” on page 270 for more information.

NOCONSTANT
NOINT suppresses the fitting of a constant (or intercept) parameter in the model. (That is, the parameter $\mu$ is omitted.)

NODF estimates the variance by dividing the error sum of squares (SSE) by the number of residuals. The default is to divide the SSE by the number of residuals minus the number of free parameters in the model.
NOPRINT
suppresses the normal printout generated by the ESTIMATE statement. If the NOPRINT option is specified for the ESTIMATE statement, then any error and warning messages are printed to the SAS log.

\[ P=order \]
\[ P=(lag, \ldots, lag) \ldots (lag, \ldots, lag) \]
specifies the autoregressive part of the model. By default, no autoregressive parameters are fit.

\[ P=(l_1, l_2, \ldots, l_k) \] defines a model with autoregressive parameters at the specified lags. \( P= order \) is equivalent to \( P=(1, 2, \ldots, order) \).

A concatenation of parenthesized lists specifies a factored model. For example, \( P=(1,2,5)(6,12) \) specifies the autoregressive model

\[
(1 - \phi_{1,1}B - \phi_{1,2}B^2 - \phi_{1,3}B^5)(1 - \phi_{2,1}B^6 - \phi_{2,2}B^{12})
\]

PLOT
plots the residual autocorrelation functions. The sample autocorrelation, the sample inverse autocorrelation, and the sample partial autocorrelation functions of the model residuals are plotted.

\[ Q=order \]
\[ Q=(lag, \ldots, lag) \ldots (lag, \ldots, lag) \]
specifies the moving-average part of the model. By default, no moving-average part is included in the model.

\[ Q=(l_1, l_2, \ldots, l_k) \] defines a model with moving-average parameters at the specified lags. \( Q= order \) is equivalent to \( Q=(1, 2, \ldots, order) \). A concatenation of parenthesized lists specifies a factored model. The interpretation of factors and lags is the same as for the \( P= \) option.

WHITENOISE=ST | IGNOREMISS
specifies the type of test statistic that is used in the white noise test of the series when the series contains missing values. If \( \text{WHITENOISE=} \text{IGNOREMISS} \), the standard Ljung-Box test statistic is used. If \( \text{WHITENOISE=} \text{ST} \), a modification of this statistic suggested by Stoffer and Toloi (1992) is used. The default is \( \text{WHITENOISE=} \text{ST} \).

Options for Output Data Sets

The following options are used to store results in SAS data sets:

\textbf{OUTEST=SAS-data-set}
writes the parameter estimates to an output data set. If the OUTCORR or OUTCOV option is used, the correlations or covariances of the estimates are also written to the OUTEST= data set. See the section “OUTEST= Data Set” on page 284 for a description of the OUTEST= output data set.

\textbf{OUTCORR}
writes the correlations of the parameter estimates to the OUTEST= data set.
OUTCOV
writes the covariances of the parameter estimates to the OUTEST= data set.

OUTMODEL=SAS-data-set
writes the model and parameter estimates to an output data set. If OUTMODEL= is not specified, no model output data set is created. See the section “OUTMODEL= SAS Data Set” on page 287 for a description of the OUTMODEL= output data set.

OUTSTAT=SAS-data-set
writes the model diagnostic statistics to an output data set. If OUTSTAT= is not specified, no statistics output data set is created. See the section “OUTSTAT= Data Set” on page 288 for a description of the OUTSTAT= output data set.

Options to Specify Parameter Values

The following options enable you to specify values for the model parameters. These options can provide starting values for the estimation process, or you can specify fixed parameters for use in the FORECAST stage and suppress the estimation process with the NOEST option. By default, the ARIMA procedure finds initial parameter estimates and uses these estimates as starting values in the iterative estimation process.

If values for any parameters are specified, values for all parameters should be given. The number of values given must agree with the model specifications.

AR=value . . .
lists starting values for the autoregressive parameters. See the section “Initial Values” on page 275 for more information.

INITVAL=(initializer-spec variable . . . )
specifies starting values for the parameters in the transfer function parts of the model. See the section “Initial Values” on page 275 for more information.

MA=value . . .
lists starting values for the moving-average parameters. See the section “Initial Values” on page 275 for more information.

MU=value
specifies the MU parameter.

NOEST
uses the values specified with the AR=, MA=, INITVAL=, and MU= options as final parameter values. The estimation process is suppressed except for estimation of the residual variance. The specified parameter values are used directly by the next FORECAST statement. When NOEST is specified, standard errors, t values, and the correlations between estimates are displayed as 0 or missing. (The NOEST option is useful, for example, when you want to generate forecasts that correspond to a published model.)

Options to Control the Iterative Estimation Process

The following options can be used to control the iterative process of minimizing the error sum of squares or maximizing the log-likelihood function. These tuning options are not usually needed but can be useful if convergence problems arise.
BACKLIM=–n
omits the specified number of initial residuals from the sum of squares or likelihood function. Omitting values can be useful for suppressing transients in transfer function models that are sensitive to start-up values.

CONVERGE=value
specifies the convergence criterion. Convergence is assumed when the largest change in the estimate for any parameter is less than the CONVERGE= option value. If the absolute value of the parameter estimate is greater than 0.01, the relative change is used; otherwise, the absolute change in the estimate is used. The default is CONVERGE=0.001.

DELTA=value
specifies the perturbation value for computing numerical derivatives. The default is DELTA=0.001.

GRID
prints the error sum of squares (SSE) or concentrated log-likelihood surface in a small grid of the parameter space around the final estimates. For each pair of parameters, the SSE is printed for the nine parameter-value combinations formed by the grid, with a center at the final estimates and with spacing given by the GRIDVAL= specification. The GRID option can help you judge whether the estimates are truly at the optimum, since the estimation process does not always converge. For models with a large number of parameters, the GRID option produces voluminous output.

GRIDVAL=number
controls the spacing in the grid printed by the GRID option. The default is GRIDVAL=0.005.

MAXITER=n
specifies the maximum number of iterations allowed. The default is MAXITER=50.

NOLS
begins the maximum likelihood or unconditional least squares iterations from the preliminary estimates rather than from the conditional least squares estimates that are produced after four iterations. See the section “Estimation Details” on page 270 for more information.

NOSTABLE
specifies that the autoregressive and moving-average parameter estimates for the noise part of the model not be restricted to the stationary and invertible regions, respectively. See the section “Stationarity and Invertibility” on page 276 for more information.

PRINTALL
prints preliminary estimation results and the iterations in the final estimation process.

NOTFSTABLE
specifies that the parameter estimates for the denominator polynomial of the transfer function part of the model not be restricted to the stability region. See the section “Stationarity and Invertibility” on page 276 for more information.

SINGULAR=value
specifies the criterion for checking singularity. If a pivot of a sweep operation is less than the SINGULAR= value, the matrix is deemed singular. Sweep operations are performed on the Jacobian matrix during final estimation and on the covariance matrix when preliminary estimates are obtained. The default is SINGULAR=1E–7.
OUTLIER Statement

OUTLIER options;

The OUTLIER statement can be used to detect shifts in the level of the response series that are not accounted for by the previously estimated model. An ESTIMATE statement must precede the OUTLIER statement. The following options are used in the OUTLIER statement:

TYPE=ADDITIVE
TYPE=SHIFT
TYPE=TEMP ( d1, . . . , dk )
TYPE=([ADDITIVE] [SHIFT] [TEMP ( d1, . . . , dk )])

specifies the types of level shifts to search for. The default is TYPE=(ADDITIVE SHIFT), which requests searching for additive outliers and permanent level shifts. The option TEMP( d1, . . . , dk) requests searching for temporary changes in the level of durations d1, . . . , dk. These options can also be abbreviated as AO, LS, and TC.

ALPHA=significance-level

specifies the significance level for tests in the OUTLIER statement. The default is 0.05.

SIGMA=ROBUST | MSE

specifies the type of error variance estimate to use in the statistical tests performed during the outlier detection. SIGMA=MSE corresponds to the usual mean squared error (MSE) estimate, and SIGMA=ROBUST corresponds to a robust estimate of the error variance. The default is SIGMA=ROBUST.

MAXNUM=number

limits the number of outliers to search. The default is MAXNUM=5.

MAXPCT=number

limits the number of outliers to search for according to a percentage of the series length. The default is MAXPCT=2. When both the MAXNUM= and MAXPCT= options are specified, the minimum of the two search numbers is used.

ID=Date-Time ID variable

specifies a SAS date, time, or datetime identification variable to label the detected outliers. This variable must be present in the input data set.

The following examples illustrate a few possibilities for the OUTLIER statement.

The most basic usage, shown as follows, sets all the options to their default values.

    outlier;

That is, it is equivalent to

    outlier type=(ao ls) alpha=0.05 sigma=robust maxnum=5 maxpct=2;
The following statement requests a search for permanent level shifts and for temporary level changes of durations 6 and 12. The search is limited to at most three changes and the significance level of the underlying tests is 0.001. MSE is used as the estimate of error variance. It also requests labeling of the detected shifts using an ID variable `date`.

```
outlier type=(ls tc(6 12)) alpha=0.001 sigma=mse maxnum=3 ID=date;
```

**FORECAST Statement**

```
FORECAST options ;
```

The FORECAST statement generates forecast values for a time series by using the parameter estimates produced by the previous ESTIMATE statement. See the section “Forecasting Details” on page 278 for more information about calculating forecasts.

The following options can be used in the FORECAST statement:

- **ALIGN=** *option*
  
  controls the alignment of SAS dates used to identify output observations. The ALIGN= option allows the following values: `BEGINNING|BEG`, `MIDDLE|MID`, and `ENDING|END`. `BEGINNING` is the default.

- **ALPHA=** *n*
  
  sets the size of the forecast confidence limits. The ALPHA= value must be between 0 and 1. When you specify `ALPHA=α`, the upper and lower confidence limits have a $(1 - α)$ confidence level. The default is `ALPHA=0.05`, which produces 95% confidence intervals. ALPHA values are rounded to the nearest hundredth.

- **BACK=** *n*
  
  specifies the number of observations before the end of the data where the multistep forecasts are to begin. The BACK= option value must be less than or equal to the number of observations minus the number of parameters.

  The default is `BACK=0`, which means that the forecast starts at the end of the available data. The end of the data is the last observation for which a noise value can be calculated. If there are no input series, the end of the data is the last nonmissing value of the response time series. If there are input series, this observation can precede the last nonmissing value of the response variable, since there may be missing values for some of the input series.

- **ID=** *variable*
  
  names a variable in the input data set that identifies the time periods associated with the observations. The ID= variable is used in conjunction with the INTERVAL= option to extrapolate ID values from the end of the input data to identify forecast periods in the OUT= data set.

  If the INTERVAL= option specifies an interval type, the ID variable must be a SAS date or datetime variable with the spacing between observations indicated by the INTERVAL= value. If the INTERVAL= option is not used, the last input value of the ID= variable is incremented by one for each forecast period to extrapolate the ID values for forecast observations.
INTERVAL=\textit{interval}

\textbf{INTERVAL=}n

specifies the time interval between observations. See Chapter 5, “Date Intervals, Formats, and Functions,” for information about valid INTERVAL= values.

The value of the INTERVAL= option is used by PROC ARIMA to extrapolate the ID values for forecast observations and to check that the input data are in order with no missing periods. See the section “Specifying Series Periodicity” on page 280 for more details.

\textbf{LEAD=}n

specifies the number of multistep forecast values to compute. For example, if LEAD=10, PROC ARIMA forecasts for ten periods beginning with the end of the input series (or earlier if BACK= is specified). It is possible to obtain fewer than the requested number of forecasts if a transfer function model is specified and insufficient data are available to compute the forecast. The default is LEAD=24.

\textbf{NOOUTALL}

includes only the final forecast observations in the OUT= output data set, not the one-step forecasts for the data before the forecast period.

\textbf{NOPRINT}

suppresses the normal printout of the forecast and associated values.

\textbf{OUT=}SAS-data-set

writes the forecast (and other values) to an output data set. If OUT= is not specified, the OUT= data set specified in the PROC ARIMA statement is used. If OUT= is also not specified in the PROC ARIMA statement, no output data set is created. See the section “OUT= Data Set” on page 282 for more information.

\textbf{PRINTALL}

prints the FORECAST computation throughout the whole data set. The forecast values for the data before the forecast period (specified by the BACK= option) are one-step forecasts.

\textbf{SIGSQ=}value

specifies the variance term used in the formula for computing forecast standard errors and confidence limits. The default value is the variance estimate computed by the preceding ESTIMATE statement. This option is useful when you wish to generate forecast standard errors and confidence limits based on a published model. It would often be used in conjunction with the NOEST option in the preceding ESTIMATE statement.

---

Details: ARIMA Procedure

The Inverse Autocorrelation Function

The sample inverse autocorrelation function (SIACF) plays much the same role in ARIMA modeling as the sample partial autocorrelation function (SPACF), but it generally indicates subset and seasonal autoregressive models better than the SPACF.
Additionally, the SIACF can be useful for detecting over-differencing. If the data come from a nonstationary or nearly nonstationary model, the SIACF has the characteristics of a noninvertible moving-average. Likewise, if the data come from a model with a noninvertible moving average, then the SIACF has nonstationary characteristics and therefore decays slowly. In particular, if the data have been over-differenced, the SIACF looks like a SACF from a nonstationary process.

The inverse autocorrelation function is not often discussed in textbooks, so a brief description is given here. For more complete discussions, see: Cleveland (1972); Chatfield (1980); Priestley (1981).

Let \( W_t \) be generated by the ARMA\((p, q)\) process

\[
\phi(B)W_t = \theta(B)a_t
\]

where \( a_t \) is a white noise sequence. If \( \theta(B) \) is invertible (that is, if \( \theta \) considered as a polynomial in \( B \) has no roots less than or equal to 1 in magnitude), then the model

\[
\theta(B)Z_t = \phi(B)a_t
\]

is also a valid ARMA\((q,p)\) model. This model is sometimes referred to as the dual model. The autocorrelation function (ACF) of this dual model is called the inverse autocorrelation function (IACF) of the original model. Notice that if the original model is a pure autoregressive model, then the IACF is an ACF that corresponds to a pure moving-average model. Thus, it cuts off sharply when the lag is greater than \( p \); this behavior is similar to the behavior of the partial autocorrelation function (PACF).

The sample inverse autocorrelation function (SIACF) is estimated in the ARIMA procedure by the following steps. A high-order autoregressive model is fit to the data by means of the Yule-Walker equations. The order of the autoregressive model used to calculate the SIACF is the minimum of the NLAG= value and one-half the number of observations after differencing. The SIACF is then calculated as the autocorrelation function that corresponds to this autoregressive operator when treated as a moving-average operator. That is, the autoregressive coefficients are convolved with themselves and treated as autocovariances.

Under certain conditions, the sampling distribution of the SIACF can be approximated by the sampling distribution of the SACF of the dual model (Bhansali 1980). In the plots generated by ARIMA, the confidence limit marks (.) are located at \( \pm 2/\sqrt{m} \). These limits bound an approximate 95% confidence interval for the hypothesis that the data are from a white noise process.

---

**The Partial Autocorrelation Function**

The approximation for a standard error for the estimated partial autocorrelation function at lag \( k \) is based on a null hypothesis that a pure autoregressive Gaussian process of order \( k-1 \) generated the time series. This standard error is \( 1/\sqrt{m} \) and is used to produce the approximate 95% confidence intervals depicted by the dots in the plot.

---

**The Cross-Correlation Function**

The autocorrelation and partial and inverse autocorrelation functions described in the preceding sections help when you want to model a series as a function of its past values and past random errors. When you want to
include the effects of past and current values of other series in the model, the correlations of the response series and the other series must be considered.

The CROSSCORR= option in the IDENTIFY statement computes cross-correlations of the VAR= series with other series and makes these series available for use as inputs in models specified by later ESTIMATE statements.

When the CROSSCORR= option is used, PROC ARIMA prints a plot of the cross-correlation function for each variable in the CROSSCORR= list. This plot is similar in format to the other correlation plots, but it shows the correlation between the two series at both lags and leads. For example,

```
identify var=y crosscorr=x ...;
```

plots the cross-correlation function of Y and X, \( \text{Cor}(y_t, x_{t-s}) \), for \( s = -L \) to \( L \), where \( L \) is the value of the NLAG= option. Study of the cross-correlation functions can indicate the transfer functions through which the input series should enter the model for the response series.

The cross-correlation function is computed after any specified differencing has been done. If differencing is specified for the VAR= variable or for a variable in the CROSSCORR= list, it is the differenced series that is cross-correlated (and the differenced series is processed by any following ESTIMATE statement).

For example,

```
identify var=y(1) crosscorr=x(1);
```

computes the cross-correlations of the changes in Y with the changes in X. When differencing is specified, the subsequent ESTIMATE statement models changes in the variables rather than the variables themselves.

The ESACF Method

The extended sample autocorrelation function (ESACF) method can tentatively identify the orders of a stationary or nonstationary ARMA process based on iterated least squares estimates of the autoregressive parameters. Tsay and Tiao (1984) proposed the technique, and Choi (1992) provides useful descriptions of the algorithm.

Given a stationary or nonstationary time series \( \{z_t: 1 \leq t \leq n\} \) with mean corrected form \( \tilde{z}_t = z_t - \mu_z \) with a true autoregressive order of \( p + d \) and with a true moving-average order of \( q \), you can use the ESACF method to estimate the unknown orders \( p + d \) and \( q \) by analyzing the autocorrelation functions associated with filtered series of the form

\[
w_t^{(m,j)} = \hat{\Phi}_{(m,j)}(B) \tilde{z}_t = \tilde{z}_t - \sum_{i=1}^{m} \hat{\phi}_{i}^{(m,j)} \tilde{z}_{t-i}
\]

where \( B \) represents the backshift operator, where \( m = p_{\text{min}}, \ldots, p_{\text{max}} \) are the autoregressive test orders, where \( j = q_{\text{min}} + 1, \ldots, q_{\text{max}} + 1 \) are the moving-average test orders, and where \( \hat{\phi}_{i}^{(m,j)} \) are the autoregressive parameter estimates under the assumption that the series is an ARMA\((m, j)\) process.
For purely autoregressive models ($j = 0$), ordinary least squares (OLS) is used to consistently estimate $\hat{\phi}_j^{(m,0)}$. For ARMA models, consistent estimates are obtained by the iterated least squares recursion formula, which is initiated by the pure autoregressive estimates:

$$\hat{\phi}_j^{(m,j)} = \hat{\phi}_j^{(m+1,j-1)} - \hat{\phi}_1^{(m,j-1)} \hat{\phi}_j^{(m+1,j-1)}$$

The $j$th lag of the sample autocorrelation function of the filtered series $w_t^{(m,j)}$ is the extended sample autocorrelation function, and it is denoted as $r_j^{(m)} = r_j(w_t^{(m,j)})$.

The standard errors of $r_j^{(m)}$ are computed in the usual way by using Bartlett’s approximation of the variance of the sample autocorrelation function, $\text{var}(r_j^{(m)}) \approx (1 + \sum_{t=1}^{j-1} r^2_j(w_t^{(m,j)}))$.

If the true model is an ARMA ($p + d, q$) process, the filtered series $w_t^{(m,j)}$ follows an MA($q$) model for $j \geq q$ so that

$$r_j^{(p+d)} \approx 0 \quad j > q$$
$$r_j^{(p+d)} \neq 0 \quad j = q$$

Additionally, Tsay and Tiao (1984) show that the extended sample autocorrelation satisfies

$$r_j^{(m)} \approx 0 \quad j - q > m - p - d \leq 0$$
$$r_j^{(m)} \neq c(m - p - d, j - q) \quad 0 \leq j - q \leq m - p - d$$

where $c(m - p - d, j - q)$ is a nonzero constant or a continuous random variable bounded by −1 and 1.

An ESACF table is then constructed by using the $r_j^{(m)}$ for $m = p_{min}, \ldots, p_{max}$ and $j = q_{min} + 1, \ldots, q_{max} + 1$ to identify the ARMA orders (see Table 8.4). The orders are tentatively identified by finding a right (maximal) triangular pattern with vertices located at $(p + d, q)$ and $(p + d, q_{max})$ and in which all elements are insignificant (based on asymptotic normality of the autocorrelation function). The vertex $(p + d, q)$ identifies the order. Table 8.5 depicts the theoretical pattern associated with an ARMA(1,2) series.

**Table 8.4** ESACF Table

<table>
<thead>
<tr>
<th>AR</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$r_1^{(0)}$</td>
<td>$r_2^{(0)}$</td>
<td>$r_3^{(0)}$</td>
<td>$r_4^{(0)}$</td>
<td>$\cdot$</td>
</tr>
<tr>
<td>1</td>
<td>$r_1^{(1)}$</td>
<td>$r_2^{(1)}$</td>
<td>$r_3^{(1)}$</td>
<td>$r_4^{(1)}$</td>
<td>$\cdot$</td>
</tr>
<tr>
<td>2</td>
<td>$r_1^{(2)}$</td>
<td>$r_2^{(2)}$</td>
<td>$r_3^{(2)}$</td>
<td>$r_4^{(2)}$</td>
<td>$\cdot$</td>
</tr>
<tr>
<td>3</td>
<td>$r_1^{(3)}$</td>
<td>$r_2^{(3)}$</td>
<td>$r_3^{(3)}$</td>
<td>$r_4^{(3)}$</td>
<td>$\cdot$</td>
</tr>
<tr>
<td></td>
<td>$\cdot$</td>
<td>$\cdot$</td>
<td>$\cdot$</td>
<td>$\cdot$</td>
<td>$\cdot$</td>
</tr>
<tr>
<td></td>
<td>$\cdot$</td>
<td>$\cdot$</td>
<td>$\cdot$</td>
<td>$\cdot$</td>
<td>$\cdot$</td>
</tr>
</tbody>
</table>
The MINIC Method

The minimum information criterion (MINIC) method can tentatively identify the order of a stationary and invertible ARMA process. Note that Hannan and Rissanen (1982) proposed this method; for useful descriptions of the algorithm, see: Box, Jenkins, and Reinsel (1994); Choi (1992).

Given a stationary and invertible time series \( \{z_t : 1 \leq t \leq n\} \) with mean corrected form \( \tilde{z}_t = z_t - \mu_z \) with a true autoregressive order of \( p \) and with a true moving-average order of \( q \), you can use the MINIC method to compute information criteria (or penalty functions) for various autoregressive and moving average orders. The following paragraphs provide a brief description of the algorithm.

If the series is a stationary and invertible ARMA(\( p, q \)) process of the form

\[
\Phi_{(p,q)}(B) \tilde{z}_t = \Theta_{(p,q)}(B) \epsilon_t
\]

the error series can be approximated by a high-order AR process

\[
\hat{\epsilon}_t = \hat{\Phi}_{(p_e,q)}(B) \tilde{z}_t \approx \epsilon_t
\]

where the parameter estimates \( \hat{\Phi}_{(p_e,q)} \) are obtained from the Yule-Walker estimates. The choice of the autoregressive order \( p_e \) is determined by the order that minimizes the Akaike information criterion (AIC) in the range \( p_{e,\text{min}} \leq p_e \leq p_{e,\text{max}} \)

\[
AIC(p_e, 0) = \ln(\hat{\sigma}^2_{(p_e,0)}) + \frac{2(p_e + 0)}{n}
\]

where

\[
\hat{\sigma}^2_{(p_e,0)} = \frac{1}{n} \sum_{t=p_e+1}^{n} \hat{\epsilon}_t^2
\]

Note that Hannan and Rissanen (1982) use the Bayesian information criterion (BIC) to determine the autoregressive order used to estimate the error series while others recommend the AIC (Box, Jenkins, and Reinsel 1994; Choi 1992).
Once the error series has been estimated for autoregressive test order \( \mu = p_{\min}, \ldots, p_{\max} \) and for moving-average test order \( \jmath = q_{\min}, \ldots, q_{\max} \), the OLS estimates \( \hat{\Phi}_{(\mu, \jmath)} \) and \( \hat{\Theta}_{(\mu, \jmath)} \) are computed from the regression model

\[
\tilde{z}_t = \sum_{i=1}^{\mu} \phi_i^{(\mu, \jmath)} \tilde{z}_{t-i} + \sum_{k=1}^{\jmath} \theta_k^{(\mu, \jmath)} \tilde{\epsilon}_{t-k} + \text{error}
\]

From the preceding parameter estimates, the BIC is then computed

\[
BIC(\mu, \jmath) = \ln(\hat{\sigma}^{2}_{(\mu, \jmath)}) + 2(\mu + \jmath)\ln(n)/n
\]

where

\[
\hat{\sigma}^{2}_{(\mu, \jmath)} = \frac{1}{n-t_0} \sum_{t=t_0}^{n} \left( \tilde{z}_t - \sum_{i=1}^{\mu} \phi_i^{(\mu, \jmath)} \tilde{z}_{t-i} + \sum_{k=1}^{\jmath} \theta_k^{(\mu, \jmath)} \tilde{\epsilon}_{t-k} \right)
\]

where \( t_0 = p_{\max} + \max(\mu, \jmath) \).

A MINIC table is then constructed using \( BIC(\mu, \jmath) \); see Table 8.6. If \( p_{\max} > p_{\min} \), the preceding regression might fail due to linear dependence on the estimated error series and the mean-corrected series. Values of \( BIC(\mu, \jmath) \) that cannot be computed are set to missing. For large autoregressive and moving-average test orders with relatively few observations, a nearly perfect fit can result. This condition can be identified by a large negative \( BIC(\mu, \jmath) \) value.

Table 8.6  MINIC Table

<table>
<thead>
<tr>
<th>AR</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( BIC(0, 0) )</td>
<td>( BIC(0, 1) )</td>
<td>( BIC(0, 2) )</td>
<td>( BIC(0, 3) )</td>
<td>( \cdot )</td>
</tr>
<tr>
<td>1</td>
<td>( BIC(1, 0) )</td>
<td>( BIC(1, 1) )</td>
<td>( BIC(1, 2) )</td>
<td>( BIC(1, 3) )</td>
<td>( \cdot )</td>
</tr>
<tr>
<td>2</td>
<td>( BIC(2, 0) )</td>
<td>( BIC(2, 1) )</td>
<td>( BIC(2, 2) )</td>
<td>( BIC(2, 3) )</td>
<td>( \cdot )</td>
</tr>
<tr>
<td>3</td>
<td>( BIC(3, 0) )</td>
<td>( BIC(3, 1) )</td>
<td>( BIC(3, 2) )</td>
<td>( BIC(3, 3) )</td>
<td>( \cdot )</td>
</tr>
<tr>
<td>\cdot</td>
<td>\cdot</td>
<td>\cdot</td>
<td>\cdot</td>
<td>\cdot</td>
<td>\cdot</td>
</tr>
<tr>
<td>\cdot</td>
<td>\cdot</td>
<td>\cdot</td>
<td>\cdot</td>
<td>\cdot</td>
<td>\cdot</td>
</tr>
</tbody>
</table>

The SCAN Method

The smallest canonical (SCAN) correlation method can tentatively identify the orders of a stationary or nonstationary ARMA process. Tsay and Tiao (1985) proposed the technique, and for useful descriptions of the algorithm, see: Box, Jenkins, and Reinsel (1994); Choi (1992).

Given a stationary or nonstationary time series \( \{z_t \colon 1 \leq t \leq n\} \) with mean corrected form \( \tilde{z}_t = z_t - \mu_z \) with a true autoregressive order of \( p + d \) and with a true moving-average order of \( q \), you can use the SCAN method to analyze eigenvalues of the correlation matrix of the ARMA process. The following paragraphs provide a brief description of the algorithm.

For autoregressive test order \( \mu = p_{\min}, \ldots, p_{\max} \) and for moving-average test order \( \jmath = q_{\min}, \ldots, q_{\max} \), perform the following steps.
1. Let \( Y_{m,t} = (\hat{z}_t, \hat{z}_{t-1}, \ldots, \hat{z}_{t-m})' \). Compute the following \((m + 1) \times (m + 1)\) matrix

\[
\hat{\beta}(m, j + 1) = \left( \sum_t Y_{m,t-j-1}'Y_{m,t-j-1} \right)^{-1} \left( \sum_t Y_{m,t-j-1}'Y_{m,t} \right)
\]

\[
\hat{\beta}^*(m, j + 1) = \left( \sum_t Y_{m,t}'Y_{m,t} \right)^{-1} \left( \sum_t Y_{m,t}'Y_{m,t-j-1} \right)
\]

\[
\hat{A}^*(m, j) = \hat{\beta}^*(m, j + 1)\hat{\beta}(m, j + 1)
\]

where \( t \) ranges from \( j + m + 2 \) to \( n \).

2. Find the smallest eigenvalue, \( \lambda^*(m, j) \), of \( \hat{A}^*(m, j) \) and its corresponding normalized eigenvector, \( \Phi_{m,j} = (1, -\phi_1^{(m,j)}, -\phi_2^{(m,j)}, \ldots, -\phi_{m}^{(m,j)}) \). The squared canonical correlation estimate is \( \hat{\lambda}^*(m, j) \).

3. Using the \( \Phi_{m,j} \) as AR\((m)\) coefficients, obtain the residuals for \( t = j + m + 1 \) to \( n \), by following the formula:

\[
w_t^{(m,j)} = \hat{z}_t - \phi_1^{(m,j)}\hat{z}_{t-1} - \phi_2^{(m,j)}\hat{z}_{t-2} - \ldots - \phi_{m}^{(m,j)}\hat{z}_{t-m}.
\]

4. From the sample autocorrelations of the residuals, \( r_k(w) \), approximate the standard error of the squared canonical correlation estimate by

\[
\text{var}(\hat{\lambda}^*(m, j))^{1/2} \approx d(m, j)/(n - m - j)
\]

where \( d(m, j) = 1 + 2 \sum_{i=1}^{j-1} r_k(w^{(m,j)}) \).

The test statistic to be used as an identification criterion is

\[
c(m, j) = -(n - m - j)\ln(1 - \hat{\lambda}^*(m, j)/d(m, j))
\]

which is asymptotically \( \chi_1^2 \) if \( m = p + d \) and \( j \geq q \) or if \( m \geq p + d \) and \( j = q \). For \( m > p \) and \( j < q \), there is more than one theoretical zero canonical correlation between \( Y_{m,t} \) and \( Y_{m,t-j-1} \). Since the \( \hat{\lambda}^*(m, j) \) are the smallest canonical correlations for each \((m, j)\), the percentiles of \( c(m, j) \) are less than those of a \( \chi_1^2 \); therefore, Tsay and Tiao (1985) state that it is safe to assume a \( \chi_1^2 \). For \( m < p \) and \( j < q \), no conclusions about the distribution of \( c(m, j) \) are made.

A SCAN table is then constructed using \( c(m, j) \) to determine which of the \( \hat{\lambda}^*(m, j) \) are significantly different from zero (see Table 8.7). The ARMA orders are tentatively identified by finding a (maximal) rectangular pattern in which the \( \hat{\lambda}^*(m, j) \) are insignificant for all test orders \( m \geq p + d \) and \( j \geq q \). There may be more than one pair of values \((p + d, q)\) that permit such a rectangular pattern. In this case, parsimony and the number of insignificant items in the rectangular pattern should help determine the model order. Table 8.8 depicts the theoretical pattern associated with an ARMA(2,2) series.

### Table 8.7 SCAN Table

<table>
<thead>
<tr>
<th>AR</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>.</th>
<th>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>c(0,0)</td>
<td>c(0,1)</td>
<td>c(0,2)</td>
<td>c(0,3)</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>1</td>
<td>c(1,0)</td>
<td>c(1,1)</td>
<td>c(1,2)</td>
<td>c(1,3)</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>2</td>
<td>c(2,0)</td>
<td>c(2,1)</td>
<td>c(2,2)</td>
<td>c(2,3)</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>3</td>
<td>c(3,0)</td>
<td>c(3,1)</td>
<td>c(3,2)</td>
<td>c(3,3)</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
</tbody>
</table>
Table 8.8 Theoretical SCAN Table for an ARMA(2,2) Series

<table>
<thead>
<tr>
<th>AR</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>*</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>1</td>
<td>*</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>2</td>
<td>*</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>*</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>*</td>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

X = significant terms
0 = insignificant terms
* = no pattern

Stationarity Tests

When a time series has a unit root, the series is nonstationary and the ordinary least squares (OLS) estimator is not normally distributed. Dickey and Fuller studied the limiting distribution of the OLS estimator of autoregressive models for time series with a simple unit root (Dickey 1976; Dickey and Fuller 1979). Dickey, Hasza, and Fuller obtained the limiting distribution for time series with seasonal unit roots (Dickey, Hasza, and Fuller 1984). Hamilton (1994) discusses the various types of unit root testing.


The random-walk-with-drift test recommends whether or not an integrated times series has a drift term. Hamilton (1994) discusses this test.

Prewhitening

If, as is usually the case, an input series is autocorrelated, the direct cross-correlation function between the input and response series gives a misleading indication of the relation between the input and response series.

One solution to this problem is called prewhitening. You first fit an ARIMA model for the input series sufficient to reduce the residuals to white noise; then, filter the input series with this model to get the white noise residual series. You then filter the response series with the same model and cross-correlate the filtered response with the filtered input series.

The ARIMA procedure performs this prewhitening process automatically when you precede the IDENTIFY statement for the response series with IDENTIFY and ESTIMATE statements to fit a model for the input series. If a model with no inputs was previously fit to a variable specified by the CROSSCORR= option, then that model is used to prewhiten both the input series and the response series before the cross-correlations are computed for the input series.

For example,

```plaintext
proc arima data=in;
  identify var=x;
  estimate p=1 q=1;
  identify var=y crosscorr=x;
run;
```
Both X and Y are filtered by the ARMA(1,1) model fit to X before the cross-correlations are computed.

Note that prewhitening is done to estimate the cross-correlation function; the unfiltered series are used in any subsequent ESTIMATE or FORECAST statements, and the correlation functions of Y with its own lags are computed from the unfiltered Y series. But initial values in the ESTIMATE statement are obtained with prewhitened data; therefore, the result with prewhitening can be different from the result without prewhitening.

To suppress prewhitening for all input variables, use the CLEAR option in the IDENTIFY statement to make PROC ARIMA disregard all previous models.

**Prewhitening and Differencing**

If the VAR= and CROSSCORR= options specify differencing, the series are differenced before the prewhitening filter is applied. When the differencing lists specified in the VAR= option for an input and in the CROSSCORR= option for that input are not the same, PROC ARIMA combines the two lists so that the differencing operators used for prewhitening include all differences in either list (in the least common multiple sense).

**Identifying Transfer Function Models**

When identifying a transfer function model with multiple input variables, the cross-correlation functions can be misleading if the input series are correlated with each other. Any dependencies among two or more input series will confound their cross-correlations with the response series.

The prewhitening technique assumes that the input variables do not depend on past values of the response variable. If there is feedback from the response variable to an input variable, as evidenced by significant cross-correlation at negative lags, both the input and the response variables need to be prewhitened before meaningful cross-correlations can be computed.

PROC ARIMA cannot handle feedback models. The STATESPACE and VARMAX procedures are more appropriate for models with feedback.

**Missing Values and Autocorrelations**

To compute the sample autocorrelation function when missing values are present, PROC ARIMA uses only crossproducts that do not involve missing values and employs divisors that reflect the number of crossproducts used rather than the total length of the series. Sample partial autocorrelations and inverse autocorrelations are then computed by using the sample autocorrelation function. If necessary, a taper is employed to transform the sample autocorrelations into a positive definite sequence before calculating the partial autocorrelation and inverse correlation functions. The confidence intervals produced for these functions might not be valid when there are missing values. The distributional properties for sample correlation functions are not clear for finite samples. See Dunsmuir (1984) for some asymptotic properties of the sample correlation functions.
Estimation Details

The ARIMA procedure primarily uses the computational methods outlined by Box and Jenkins. Marquardt’s method is used for the nonlinear least squares iterations. Numerical approximations of the derivatives of the sum-of-squares function are taken by using a fixed delta (controlled by the DELTA= option).

The methods do not always converge successfully for a given set of data, particularly if the starting values for the parameters are not close to the least squares estimates.

Back-Forecasting

The unconditional sum of squares is computed exactly; thus, back-forecasting is not performed. Early versions of SAS/ETS software used the back-forecasting approximation and allowed a positive value of the BACKLIM= option to control the extent of the back-forecasting. In the current version, requesting a positive number of back-forecasting steps with the BACKLIM= option has no effect.

Preliminary Estimation

If an autoregressive or moving-average operator is specified with no missing lags, preliminary estimates of the parameters are computed by using the autocorrelations computed in the IDENTIFY stage. Otherwise, the preliminary estimates are arbitrarily set to values that produce stable polynomials.

When preliminary estimation is not performed by PROC ARIMA, then initial values of the coefficients for any given autoregressive or moving-average factor are set to 0.1 if the degree of the polynomial associated with the factor is 9 or less. Otherwise, the coefficients are determined by expanding the polynomial \((1 - 0.1 B)\) to an appropriate power by using a recursive algorithm.

These preliminary estimates are the starting values in an iterative algorithm to compute estimates of the parameters.

Estimation Methods

Maximum Likelihood

The METHOD= ML option produces maximum likelihood estimates. The likelihood function is maximized via nonlinear least squares using Marquardt’s method. Maximum likelihood estimates are more expensive to compute than the conditional least squares estimates; however, they may be preferable in some cases (Ansley and Newbold 1980; Davidson 1981).

The maximum likelihood estimates are computed as follows. Let the univariate ARMA model be

\[
\phi(B)(W_t - \mu_t) = \theta(B)\alpha_t
\]

where \(\alpha_t\) is an independent sequence of normally distributed innovations with mean 0 and variance \(\sigma^2\). Here \(\mu_t\) is the mean parameter \(\mu\) plus the transfer function inputs. The log-likelihood function can be written as follows:

\[
-\frac{1}{2\sigma^2}x'\Omega^{-1}x - \frac{1}{2}\ln(|\Omega|) - \frac{n}{2}\ln(\sigma^2)
\]

In this equation, \(n\) is the number of observations, \(\sigma^2\Omega\) is the variance of \(x\) as a function of the \(\phi\) and \(\theta\) parameters, and \(|\Omega|\) denotes the determinant. The vector \(x\) is the time series \(W_t\) minus the structural part of
the model \( \mu_t \), written as a column vector, as follows:

\[
\mathbf{x} = \begin{bmatrix}
W_1 \\
W_2 \\
\vdots \\
W_n
\end{bmatrix} - \begin{bmatrix}
\mu_1 \\
\mu_2 \\
\vdots \\
\mu_n
\end{bmatrix}
\]

The maximum likelihood estimate (MLE) of \( \sigma^2 \) is

\[
s^2 = \frac{1}{n} \mathbf{x}' \mathbf{\Omega}^{-1} \mathbf{x}
\]

Note that the default estimator of the variance divides by \( n - r \), where \( r \) is the number of parameters in the model, instead of by \( n \). Specifying the NODF option causes a divisor of \( n \) to be used.

The log-likelihood concentrated with respect to \( \sigma^2 \) can be taken up to additive constants as

\[
-\frac{n}{2} \ln(\mathbf{x}' \mathbf{\Omega}^{-1} \mathbf{x}) - \frac{1}{2} \ln(|\mathbf{\Omega}|)
\]

Let \( \mathbf{H} \) be the lower triangular matrix with positive elements on the diagonal such that \( \mathbf{HH}' = \mathbf{\Omega} \). Let \( \mathbf{e} \) be the vector \( \mathbf{H}^{-1} \mathbf{x} \). The concentrated log-likelihood with respect to \( \sigma^2 \) can now be written as

\[
-\frac{n}{2} \ln(\mathbf{e}' \mathbf{e}) - \ln(|\mathbf{H}|)
\]

or

\[
-\frac{n}{2} \ln(|\mathbf{H}|^{1/n} \mathbf{e}' \mathbf{H} |\mathbf{H}|^{1/n})
\]

The MLE is produced by using a Marquardt algorithm to minimize the following sum of squares:

\[
|\mathbf{H}|^{1/n} \mathbf{e}' \mathbf{H} |\mathbf{H}|^{1/n}
\]

The subsequent analysis of the residuals is done by using \( \mathbf{e} \) as the vector of residuals.

**Unconditional Least Squares**

The METHOD=ULS option produces unconditional least squares estimates. The ULS method is also referred to as the exact least squares (ELS) method. For METHOD=ULS, the estimates minimize

\[
\sum_{t=1}^{n} \hat{a}_t^2 = \sum_{t=1}^{n} (x_t - \mathbf{C}_t \mathbf{V}_t^{-1} (x_1, \ldots, x_{t-1})')^2
\]

where \( \mathbf{C}_t \) is the covariance matrix of \( x_t \) and \( (x_1, \ldots, x_{t-1}) \), and \( \mathbf{V}_t \) is the variance matrix of \( (x_1, \ldots, x_{t-1}) \). In fact, \( \sum_{t=1}^{n} \hat{a}_t^2 \) is the same as \( \mathbf{x}' \mathbf{\Omega}^{-1} \mathbf{x} \), and hence \( \mathbf{e}' \mathbf{e} \). Therefore, the unconditional least squares estimates are obtained by minimizing the sum of squared residuals rather than using the log-likelihood as the criterion function.
Conditional Least Squares

The METHOD=CLS option produces conditional least squares estimates. The CLS estimates are conditional on the assumption that the past unobserved errors are equal to 0. The series \( x_t \) can be represented in terms of the previous observations, as follows:

\[
x_t = a_t + \sum_{i=1}^{\infty} \pi_i x_{t-i}
\]

The \( \pi \) weights are computed from the ratio of the \( \phi \) and \( \theta \) polynomials, as follows:

\[
\frac{\phi(B)}{\theta(B)} = 1 - \sum_{i=1}^{\infty} \pi_i B^i
\]

The CLS method produces estimates minimizing

\[
\sum_{t=1}^{n} a_t^2 = \sum_{t=1}^{n} (x_t - \sum_{i=1}^{\infty} \hat{\pi}_i x_{t-i})^2
\]

where the unobserved past values of \( x_t \) are set to 0 and \( \hat{\pi}_t \) are computed from the estimates of \( \phi \) and \( \theta \) at each iteration.

For METHOD=ULS and METHOD=ML, initial estimates are computed using the METHOD=CLS algorithm.

Start-up for Transfer Functions

When computing the noise series for transfer function and intervention models, the start-up for the transferred variable is done by assuming that past values of the input series are equal to the first value of the series. The estimates are then obtained by applying least squares or maximum likelihood to the noise series. Thus, for transfer function models, the ML option does not generate the full (multivariate ARMA) maximum likelihood estimates, but it uses only the univariate likelihood function applied to the noise series.

Because PROC ARIMA uses all of the available data for the input series to generate the noise series, other start-up options for the transferred series can be implemented by prefixing an observation to the beginning of the real data. For example, if you fit a transfer function model to the variable Y with the single input X, then you can employ a start-up using 0 for the past values by prefixing to the actual data an observation with a missing value for Y and a value of 0 for X.

Information Criteria

PROC ARIMA computes and prints two information criteria, Akaike’s information criterion (AIC) (Akaike 1974; Harvey 1981) and Schwarz’s Bayesian criterion (SBC) (Schwarz 1978). The AIC and SBC are used to compare competing models fit to the same series. The model with the smaller information criteria is said to fit the data better. The AIC is computed as

\[-2\ln(L) + 2k\]

where \( L \) is the likelihood function and \( k \) is the number of free parameters. The SBC is computed as

\[-2\ln(L) + \ln(n)k\]
where \( n \) is the number of residuals that can be computed for the time series. Sometimes Schwarz’s Bayesian criterion is called the Bayesian information criterion (BIC).

If METHOD=CLS is used to do the estimation, an approximation value of \( L \) is used, where \( L \) is based on the conditional sum of squares instead of the exact sum of squares, and a Jacobian factor is left out.

**Tests of Residuals**

A table of test statistics for the hypothesis that the model residuals are white noise is printed as part of the ESTIMATE statement output. The chi-square statistics used in the test for lack of fit are computed using the Ljung-Box formula

\[
\chi^2_m = n(n+2) \sum_{k=1}^{m} \frac{r_k^2}{(n-k)}
\]

where

\[
r_k = \frac{\sum_{t=1}^{n-k} a_t a_{t+k}}{\sum_{t=1}^{n} a_t^2}
\]

and \( a_t \) is the residual series.

This formula has been suggested by Ljung and Box (1978) as yielding a better fit to the asymptotic chi-square distribution than the Box-Pierce Q statistic. Some simulation studies of the finite sample properties of this statistic are given by Davies, Triggs, and Newbold (1977); Ljung and Box (1978). When the time series has missing values, Stoffer and Toloi (1992) suggest a modification of this test statistic that has improved distributional properties over the standard Ljung-Box formula given above. When the series contains missing values, this modified test statistic is used by default.

Each chi-square statistic is computed for all lags up to the indicated lag value and is not independent of the preceding chi-square values. The null hypotheses tested is that the current set of autocorrelations is white noise.

**t-values**

The \( t \) values reported in the table of parameter estimates are approximations whose accuracy depends on the validity of the model, the nature of the model, and the length of the observed series. When the length of the observed series is short and the number of estimated parameters is large with respect to the series length, the \( t \) approximation is usually poor. Probability values that correspond to a \( t \) distribution should be interpreted carefully because they may be misleading.

**Cautions during Estimation**

The ARIMA procedure uses a general nonlinear least squares estimation method that can yield problematic results if your data do not fit the model. Output should be examined carefully. The GRID option can be used to ensure the validity and quality of the results. Problems you might encounter include the following:

- Preliminary moving-average estimates might not converge. If this occurs, preliminary estimates are derived as described previously in “Preliminary Estimation” on page 270. You can supply your own preliminary estimates with the ESTIMATE statement options.
• The estimates can lead to an unstable time series process, which can cause extreme forecast values or overflows in the forecast.

• The Jacobian matrix of partial derivatives might be singular; usually, this happens because not all the parameters are identifiable. Removing some of the parameters or using a longer time series might help.

• The iterative process might not converge. PROC ARIMA’s estimation method stops after \( n \) iterations, where \( n \) is the value of the MAXITER= option. If an iteration does not improve the SSE, the Marquardt parameter is increased by a factor of ten until parameters that have a smaller SSE are obtained or until the limit value of the Marquardt parameter is exceeded.

• For METHOD=CLS, the estimates might converge but not to least squares estimates. The estimates might converge to a local minimum, the numerical calculations might be distorted by data whose sum-of-squares surface is not smooth, or the minimum might lie outside the region of invertibility or stationarity.

• If the data are differenced and a moving-average model is fit, the parameter estimates might try to converge exactly on the invertibility boundary. In this case, the standard error estimates that are based on derivatives might be inaccurate.

---

### Specifying Inputs and Transfer Functions

Input variables and transfer functions for them can be specified using the INPUT= option in the ESTIMATE statement. The variables used in the INPUT= option must be included in the CROSSCORR= list in the previous IDENTIFY statement. If any differencing is specified in the CROSSCORR= list, then the differenced variable is used as the input to the transfer function.

**General Syntax of the INPUT= Option**

The general syntax of the INPUT= option is

\[
\text{ESTIMATE } \ldots \text{INPUT= ( transfer-function variable \ldots )}
\]

The transfer function for an input variable is optional. The name of a variable by itself can be used to specify a pure regression term for the variable.

If specified, the syntax of the transfer function is

\[
S \ \$ \ (L_{1,1}, L_{1,2}, \ldots)(L_{2,1}, \ldots)/(L_{1,1}, L_{1,2}, \ldots)(L_{i+1,1}, \ldots)
\]

\( S \) is the number of periods of time delay (lag) for this input series. Each term in parentheses specifies a polynomial factor with parameters at the lags specified by the \( L_{i,j} \) values. The terms before the slash (/) are numerator factors. The terms after the slash (/) are denominator factors. All three parts are optional.

Commas can optionally be used between input specifications to make the INPUT= option more readable. The \$ sign after the shift is also optional.

Except for the first numerator factor, each of the terms \( L_{i,1}, L_{i,2}, \ldots, L_{i,k} \) indicates a factor of the form

\[
(1 - \omega_{i,1} B^{L_{i,1}} - \omega_{i,2} B^{L_{i,2}} - \ldots - \omega_{i,k} B^{L_{i,k}})
\]

The form of the first numerator factor depends on the ALTPARM option. By default, the constant 1 in the first numerator factor is replaced with a free parameter \( \omega_0 \).
**Alternative Model Parameterization**

When the ALTPARM option is specified, the $\omega_0$ parameter is factored out so that it multiplies the entire transfer function, and the first numerator factor has the same form as the other factors.

The ALTPARM option does not materially affect the results; it just presents the results differently. Some people prefer to see the model written one way, while others prefer the alternative representation. Table 8.9 illustrates the effect of the ALTPARM option.

<table>
<thead>
<tr>
<th>INPUT= Option</th>
<th>ALTPARM</th>
<th>Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>INPUT=((1 2)(12)/(1)X);</td>
<td>No</td>
<td>$\frac{(\omega_0 - \omega_1 B - \omega_2 B^2)(1 - \omega_3 B^{12})}{(1 - \delta_1 B)} X_t$</td>
</tr>
<tr>
<td></td>
<td>Yes</td>
<td>$\omega_0 (1 - \omega_1 B - \omega_2 B^2)(1 - \omega_3 B^{12})/(1 - \delta_1 B) X_t$</td>
</tr>
</tbody>
</table>

**Differencing and Input Variables**

If you difference the response series and use input variables, take care that the differencing operations do not change the meaning of the model. For example, if you want to fit the model

$$Y_t = \frac{\omega_0}{(1 - \delta_1 B)} X_t + \frac{(1 - \theta_1 B)}{(1 - B)(1 - B^{12})} a_t$$

then the IDENTIFY statement must read

```plaintext
identify var=y(1,12) crosscorr=x(1,12);
estimate q=1 input=(/(1)x) noconstant;
```

If instead you specify the differencing as

```plaintext
identify var=y(1,12) crosscorr=x;
estimate q=1 input=(/(1)x) noconstant;
```

then the model being requested is

$$Y_t = \frac{\omega_0}{(1 - \delta_1 B)(1 - B)(1 - B^{12})} X_t + \frac{(1 - \theta_1 B)}{(1 - B)(1 - B^{12})} a_t$$

which is a very different model.

The point to remember is that a differencing operation requested for the response variable specified by the VAR= option is applied only to that variable and not to the noise term of the model.

---

**Initial Values**

The syntax for giving initial values to transfer function parameters in the INITVAL= option parallels the syntax of the INPUT= option. For each transfer function in the INPUT= option, the INITVAL= option should
give an initialization specification followed by the input series name. The initialization specification for each
transfer function has the form
\[ C \xi (V_{1,1}, V_{1,2}, \ldots)(V_{2,1}, \ldots)\ldots/(V_{i,1}, \ldots)\ldots \]
where \( C \) is the lag 0 term in the first numerator factor of the transfer function (or the overall scale factor if the
ALTPARM option is specified) and \( V_{i,j} \) is the coefficient of the \( L_{i,j} \) element in the transfer function.

To illustrate, suppose you want to fit the model
\[
Y_t = \mu + \frac{(\omega_0 - \omega_1 B - \omega_2 B^2)}{(1 - \delta_1 B - \delta_2 B^2 - \delta_3 B^3)}X_{t-3} + \frac{1}{(1 - \phi_1 B - \phi_2 B^3)}a_t
\]
and start the estimation process with the initial values \( \mu=10, \omega_0=1, \omega_1=0.5, \omega_2=0.03, \delta_1=0.8, \delta_2=-0.1, \delta_3=0.002, \phi_1=0.1, \phi_2=0.01. \) (These are arbitrary values for illustration only.) You would use the
following statements:

```plaintext
identify var=y crosscorr=x;
estimate p=(1,3) input=(3$(1,2)/(1,2,3)x)
    mu=10 ar=.1 .01
    initval=(1$(.5,.03)/(.8,-.1,.002)x);
```

Note that the lags specified for a particular factor are sorted, so initial values should be given in sorted order. For
example, if the P= option had been entered as P=(3,1) instead of P=(1,3), the model would be the same
and so would the AR= option. Sorting is done within all factors, including transfer function factors, so initial
values should always be given in order of increasing lags.

Here is another illustration, showing initialization for a factored model with multiple inputs. The model is
\[
Y_t = \mu + \frac{\omega_{1,0}}{(1 - \delta_{1,1} B)}W_t + (\omega_{2,0} - \omega_{2,1} B)X_{t-3}
+ \frac{1}{(1 - \phi_1 B)(1 - \phi_2 B^6 - \phi_3 B^{12})}a_t
\]
and the initial values are \( \mu=10, \omega_{1,0}=5, \delta_{1,1}=0.8, \omega_{2,0}=1, \omega_{2,1}=0.5, \phi_1=0.1, \phi_2=0.05, \) and \( \phi_3=0.01. \) You
would use the following statements:

```plaintext
identify var=y crosscorr=(w x);
estimate p=(1)(6,12) input=/(1)w 3$(1)x)
    mu=10 ar=.1 .05 .01
    initval=(5$//.8)w 1$(.5)x);
```

---

**Stationarity and Invertibility**

By default, PROC ARIMA requires that the parameter estimates for the AR and MA parts of the model
always remain in the stationary and invertible regions, respectively. The NOSTABLE option removes this
restriction and for high-order models can save some computer time. Note that using the NOSTABLE option
does not necessarily result in an unstable model being fit, since the estimates can leave the stable region for
some iterations but still ultimately converge to stable values. Similarly, by default, the parameter estimates
for the denominator polynomial of the transfer function part of the model are also restricted to be stable. The
NOTFSTABLE option can be used to remove this restriction.
**Naming of Model Parameters**

In the table of parameter estimates produced by the ESTIMATE statement, model parameters are referred to by using the naming convention described in this section.

The parameters in the noise part of the model are named as AR\(_i,j\) or MA\(_i,j\), where AR refers to autoregressive parameters and MA to moving-average parameters. The subscript \(i\) refers to the particular polynomial factor, and the subscript \(j\) refers to the \(j\)th term within the \(i\)th factor. These terms are sorted in order of increasing lag within factors, so the subscript \(j\) refers to the \(j\)th term after sorting.

When inputs are used in the model, the parameters of each transfer function are named NUM\(_i,j\) and DEN\(_i,j\). The \(j\)th term in the \(i\)th factor of a numerator polynomial is named NUM\(_i,j\). The \(j\)th term in the \(i\)th factor of a denominator polynomial is named DEN\(_i,j\).

This naming process is repeated for each input variable, so if there are multiple inputs, parameters in transfer functions for different input series have the same name. The table of parameter estimates shows in the “Variable” column the input with which each parameter is associated. The parameter name shown in the “Parameter” column and the input variable name shown in the “Variable” column must be combined to fully identify transfer function parameters.

The lag 0 parameter in the first numerator factor for the first input variable is named NUM\(_1\). For subsequent input variables, the lag 0 parameter in the first numerator factor is named NUM\(_k\), where \(k\) is the position of the input variable in the INPUT= option list. If the ALTPARM option is specified, the NUM\(_k\) parameter is replaced by an overall scale parameter named SCALE\(_k\).

For the mean and noise process parameters, the response series name is shown in the “Variable” column. The lag and shift for each parameter are also shown in the table of parameter estimates when inputs are used.

**Missing Values and Estimation and Forecasting**

Estimation and forecasting are carried out in the presence of missing values by forecasting the missing values with the current set of parameter estimates. The maximum likelihood algorithm employed was suggested by Jones (1980) and is used for both unconditional least squares (ULS) and maximum likelihood (ML) estimation.

The CLS algorithm simply fills in missing values with infinite memory forecast values, computed by forecasting ahead from the nonmissing past values as far as required by the structure of the missing values. These artificial values are then employed in the nonmissing value CLS algorithm. Artificial values are updated at each iteration along with parameter estimates.

For models with input variables, embedded missing values (that is, missing values other than at the beginning or end of the series) are not generally supported. Embedded missing values in input variables are supported for the special case of a multiple regression model that has ARIMA errors. A multiple regression model is specified by an INPUT= option that simply lists the input variables (possibly with lag shifts) without any numerator or denominator transfer function factors. One-step-ahead forecasts are not available for the response variable when one or more of the input variables have missing values.

When embedded missing values are present for a model with complex transfer functions, PROC ARIMA uses the first continuous nonmissing piece of each series to do the analysis. That is, PROC ARIMA skips observations at the beginning of each series until it encounters a nonmissing value and then uses the data from
there until it encounters another missing value or until the end of the data is reached. This makes the current version of PROC ARIMA compatible with earlier releases that did not allow embedded missing values.

**Forecasting Details**

If the model has input variables, a forecast beyond the end of the data for the input variables is possible only if univariate ARIMA models have previously been fit to the input variables or future values for the input variables are included in the DATA= data set.

If input variables are used, the forecast standard errors and confidence limits of the response depend on the estimated forecast error variance of the predicted inputs. If several input series are used, the forecast errors for the inputs should be independent; otherwise, the standard errors and confidence limits for the response series will not be accurate. If future values for the input variables are included in the DATA= data set, the standard errors of the forecasts will be underestimated since these values are assumed to be known with certainty.

The forecasts are generated using forecasting equations consistent with the method used to estimate the model parameters. Thus, the estimation method specified in the ESTIMATE statement also controls the way forecasts are produced by the FORECAST statement. If METHOD=CLS is used, the forecasts are *infinite memory forecasts*, also called *conditional forecasts*. If METHOD=ULS or METHOD=ML, the forecasts are *finite memory forecasts*, also called *unconditional forecasts*. A complete description of the steps to produce the series forecasts and their standard errors by using either of these methods is quite involved, and only a brief explanation of the algorithm is given in the next two sections. Additional details about the finite and infinite memory forecasts can be found in Brockwell and Davis (1991). The prediction of stationary ARMA processes is explained in Chapter 5, and the prediction of nonstationary ARMA processes is given in Chapter 9 of Brockwell and Davis (1991).

**Infinite Memory Forecasts**

If METHOD=CLS is used, the forecasts are *infinite memory forecasts*, also called *conditional forecasts*. The term *conditional* is used because the forecasts are computed by assuming that the unknown values of the response series before the start of the data are equal to the mean of the series. Thus, the forecasts are conditional on this assumption.

The series $x_t$ can be represented as

$$x_t = a_t + \sum_{i=1}^{\infty} \pi_i x_{t-i}$$

where $\phi(B)/\theta(B) = 1 - \sum_{i=1}^{\infty} \pi_i B^i$.

The $k$-step forecast of $x_{t+k}$ is computed as

$$\hat{x}_{t+k} = \sum_{i=1}^{k-1} \hat{\pi}_i \hat{x}_{t+k-i} + \sum_{i=k}^{\infty} \hat{\pi}_i x_{t+k-i}$$

where unobserved past values of $x_t$ are set to zero and $\hat{\pi}_i$ is obtained from the estimated parameters $\hat{\phi}$ and $\hat{\theta}$. 
Finite Memory Forecasts

For METHOD=ULS or METHOD=ML, the forecasts are finite memory forecasts, also called unconditional forecasts. For finite memory forecasts, the covariance function of the ARMA model is used to derive the best linear prediction equation.

That is, the $k$-step forecast of $x_{t+k}$, given $(x_1, \ldots, x_{t-1})$, is

$$\hat{x}_{t+k} = C_{k,t} V_t^{-1} (x_1, \ldots, x_{t-1})'$$

where $C_{k,t}$ is the covariance of $x_{t+k}$ and $(x_1, \ldots, x_{t-1})$ and $V_t$ is the covariance matrix of the vector $(x_1, \ldots, x_{t-1})$. $C_{k,t}$ and $V_t$ are derived from the estimated parameters.

Finite memory forecasts minimize the mean squared error of prediction if the parameters of the ARMA model are known exactly. (In most cases, the parameters of the ARMA model are estimated, so the predictors are not true best linear forecasts.)

If the response series is differenced, the final forecast is produced by summing the forecast of the differenced series. This summation and the forecast are conditional on the initial values of the series. Thus, when the response series is differenced, the final forecasts are not true finite memory forecasts because they are derived by assuming that the differenced series begins in a steady-state condition. Thus, they fall somewhere between finite memory and infinite memory forecasts. In practice, there is seldom any practical difference between these forecasts and true finite memory forecasts.

Forecasting Log Transformed Data

The log transformation is often used to convert time series that are nonstationary with respect to the innovation variance into stationary time series. The usual approach is to take the log of the series in a DATA step and then apply PROC ARIMA to the transformed data. A DATA step is then used to transform the forecasts of the logs back to the original units of measurement. The confidence limits are also transformed by using the exponential function.

As one alternative, you can simply exponentiate the forecast series. This procedure gives a forecast for the median of the series, but the antilog of the forecast log series underpredicts the mean of the original series. If you want to predict the expected value of the series, you need to take into account the standard error of the forecast, as shown in the following example, which uses an AR(2) model to forecast the log of a series $Y$:

```plaintext
data in;
  set in;
  ylog = log( y );
run;

proc arima data=in;
  identify var=ylog;
  estimate p=2;
  forecast lead=10 out=out;
run;

data out;
  set out;
  y = exp( ylog );
```
Specifying Series Periodicity

The INTERVAL= option is used together with the ID= variable to describe the observations that make up the time series. For example, INTERVAL=MONTH specifies a monthly time series in which each observation represents one month. See Chapter 5, “Date Intervals, Formats, and Functions,” for details about the interval values supported.

The variable specified by the ID= option in the PROC ARIMA statement identifies the time periods associated with the observations. Usually, SAS date, time, or datetime values are used for this variable. PROC ARIMA uses the ID= variable in the following ways:

- to validate the data periodicity. When the INTERVAL= option is specified, PROC ARIMA uses the ID variable to check the data and verify that successive observations have valid ID values that correspond to successive time intervals. When the INTERVAL= option is not used, PROC ARIMA verifies that the ID values are nonmissing and in ascending order.

- to check for gaps in the input observations. For example, if INTERVAL=MONTH and an input observation for April 1970 follows an observation for January 1970, there is a gap in the input data with two omitted observations (namely February and March 1970). A warning message is printed when a gap in the input data is found.

- to label the forecast observations in the output data set. PROC ARIMA extrapolates the values of the ID variable for the forecast observations from the ID value at the end of the input data according to the frequency specifications of the INTERVAL= option. If the INTERVAL= option is not specified, PROC ARIMA extrapolates the ID variable by incrementing the ID variable value for the last observation in the input data by 1 for each forecast period. Values of the ID variable over the range of the input data are copied to the output data set.

The ALIGN= option is used to align the ID variable to the beginning, middle, or end of the time ID interval specified by the INTERVAL= option.

Detecting Outliers

You can use the OUTLIER statement to detect changes in the level of the response series that are not accounted for by the estimated model. The types of changes considered are additive outliers (AO), level shifts (LS), and temporary changes (TC).

Let $\eta_t$ be a regression variable that describes some type of change in the mean response. In time series literature $\eta_t$ is called a shock signature. An additive outlier at some time point $s$ corresponds to a shock signature $\eta_s$ such that $\eta_s = 1.0$ and $\eta_t$ is 0.0 at all other points. Similarly a permanent level shift that
Detecting Outliers

originates at time $s$ has a shock signature such that $\eta_t = 0.0$ for $t < s$ and $1.0$ for $t \geq s$. A temporary level shift of duration $d$ that originates at time $s$ has $\eta_t$ equal to $1.0$ between $s$ and $s + d$ and $0.0$ otherwise.

Suppose that you are estimating the ARIMA model

$$D(B)Y_t = \mu_t + \frac{\theta(B)}{\phi(B)}a_t$$

where $Y_t$ is the response series, $D(B)$ is the differencing polynomial in the backward shift operator $B$ (possibly identity), $\mu_t$ is the transfer function input, $\phi(B)$ and $\theta(B)$ are the AR and MA polynomials, respectively, and $a_t$ is the Gaussian white noise series.

The problem of detection of level shifts in the OUTLIER statement is formulated as a problem of sequential selection of shock signatures that improve the model in the ESTIMATE statement. This is similar to the forward selection process in the stepwise regression procedure. The selection process starts with considering shock signatures of the type specified in the TYPE= option, originating at each nonmissing measurement. This involves testing $H_0: \beta = 0$ versus $H_a: \beta \neq 0$ in the model

$$D(B)(Y_t - \beta \eta_t) = \mu_t + \frac{\theta(B)}{\phi(B)}a_t$$

for each of these shock signatures. The most significant shock signature, if it also satisfies the significance criterion in ALPHA= option, is included in the model. If no significant shock signature is found, then the outlier detection process stops; otherwise this augmented model, which incorporates the selected shock signature in its transfer function input, becomes the null model for the subsequent selection process. This iterative process stops if at any stage no more significant shock signatures are found or if the number of iterations exceeds the maximum search number that results due to the MAXNUM= and MAXPCT= settings. In all these iterations, the parameters of the ARIMA model in the ESTIMATE statement are held fixed.

The precise details of the testing procedure for a given shock signature $\eta_t$ are as follows:

The preceding testing problem is equivalent to testing $H_0: \beta = 0$ versus $H_a: \beta \neq 0$ in the following “regression with ARMA errors” model

$$N_t = \beta \xi_t + \frac{\theta(B)}{\phi(B)}a_t$$

where $N_t = (D(B)Y_t - \mu_t)$ is the “noise” process and $\xi_t = D(B)\eta_t$ is the “effective” shock signature.

In this setting, under $H_0$, $N = (N_1, N_2, \ldots, N_n)^T$ is a mean zero Gaussian vector with variance covariance matrix $\sigma^2 \Omega$. Here $\sigma^2$ is the variance of the white noise process $a_t$ and $\Omega$ is the variance-covariance matrix associated with the ARMA model. Moreover, under $H_a$, $N$ has $\beta \xi$ as the mean vector where $\xi = (\xi_1, \xi_2, \ldots, \xi_n)^T$. Additionally, the generalized least squares estimate of $\beta$ and its variance is given by

$$\hat{\beta} = \frac{\delta}{\kappa}$$

$$\text{Var}(\hat{\beta}) = \frac{\sigma^2}{\kappa}$$

where $\delta = \xi^T \Omega^{-1}N$ and $\kappa = \xi^T \Omega^{-1} \xi$. The test statistic $\tau^2 = \delta^2/(\sigma^2 \kappa)$ is used to test the significance of $\beta$, which has an approximate chi-squared distribution with 1 degree of freedom under $H_0$. The type of estimate of $\sigma^2$ used in the calculation of $\tau^2$ can be specified by the SIGMA= option. The default setting is SIGMA=ROBUST, which corresponds to a robust estimate suggested in an outlier detection procedure in
X-12-ARIMA, the Census Bureau’s time series analysis program; see Findley et al. (1998) for additional information. The robust estimate of $\sigma^2$ is computed by the formula

$$\hat{\sigma}^2 = (1.49 \times \text{Median}(|\hat{\alpha}_t|))^2$$

where $\hat{\alpha}_t$ are the standardized residuals of the null ARIMA model. The setting SIGMA=MSE corresponds to the usual mean squared error estimate (MSE) computed the same way as in the ESTIMATE statement with the NODF option.

The quantities $\delta$ and $\kappa$ are efficiently computed by a method described in De Jong and Penzer (1998); see also Kohn and Ansley (1985).

**Modeling in the Presence of Outliers**

In practice, modeling and forecasting time series data in the presence of outliers is a difficult problem for several reasons. The presence of outliers can adversely affect the model identification and estimation steps. Their presence close to the end of the observation period can have a serious impact on the forecasting performance of the model. In some cases, level shifts are associated with changes in the mechanism that drives the observation process, and separate models might be appropriate to different sections of the data. In view of all these difficulties, diagnostic tools such as outlier detection and residual analysis are essential in any modeling process.

The following modeling strategy, which incorporates level shift detection in the familiar Box-Jenkins modeling methodology, seems to work in many cases:

1. Proceed with model identification and estimation as usual. Suppose this results in a tentative ARIMA model, say M.
2. Check for additive and permanent level shifts unaccounted for by the model M by using the OUTLIER statement. In this step, unless there is evidence to justify it, the number of level shifts searched should be kept small.
3. Augment the original dataset with the regression variables that correspond to the detected outliers.
4. Include the first few of these regression variables in M, and call this model M1. Reestimate all the parameters of M1. It is important not to include too many of these outlier variables in the model in order to avoid the danger of over-fitting.
5. Check the adequacy of M1 by examining the parameter estimates, residual analysis, and outlier detection. Refine it more if necessary.

**OUT= Data Set**

The output data set produced by the OUT= option of the PROC ARIMA or FORECAST statements contains the following:

- the BY variables
- the ID variable
• the variable specified by the VAR= option in the IDENTIFY statement, which contains the actual values of the response series

• FORECAST, a numeric variable that contains the one-step-ahead predicted values and the multistep forecasts

• STD, a numeric variable that contains the standard errors of the forecasts

• a numeric variable that contains the lower confidence limits of the forecast. This variable is named L95 by default but has a different name if the ALPHA= option specifies a different size for the confidence limits.

• RESIDUAL, a numeric variable that contains the differences between actual and forecast values

• a numeric variable that contains the upper confidence limits of the forecast. This variable is named U95 by default but has a different name if the ALPHA= option specifies a different size for the confidence limits.

The ID variable, the BY variables, and the response variable are the only ones copied from the input to the output data set. In particular, the input variables are not copied to the OUT= data set.

Unless the NOOUTALL option is specified, the data set contains the whole time series. The FORECAST variable has the one-step forecasts (predicted values) for the input periods, followed by \( n \) forecast values, where \( n \) is the LEAD= value. The actual and RESIDUAL values are missing beyond the end of the series.

If you specify the same OUT= data set in different FORECAST statements, the latter FORECAST statements overwrite the output from the previous FORECAST statements. If you want to combine the forecasts from different FORECAST statements in the same output data set, specify the OUT= option once in the PROC ARIMA statement and omit the OUT= option in the FORECAST statements.

When a global output data set is created by the OUT= option in the PROC ARIMA statement, the variables in the OUT= data set are defined by the first FORECAST statement that is executed. The results of subsequent FORECAST statements are vertically concatenated onto the OUT= data set. Thus, if no ID variable is specified in the first FORECAST statement that is executed, no ID variable appears in the output data set, even if one is specified in a later FORECAST statement. If an ID variable is specified in the first FORECAST statement that is executed but not in a later FORECAST statement, the value of the ID variable is the same as the last value processed for the ID variable for all observations created by the later FORECAST statement. Furthermore, even if the response variable changes in subsequent FORECAST statements, the response variable name in the output data set is that of the first response variable analyzed.

**OUTCOV= Data Set**

The output data set produced by the OUTCOV= option of the IDENTIFY statement contains the following variables:

• LAG, a numeric variable that contains the lags that correspond to the values of the covariance variables. The values of LAG range from 0 to N for covariance functions and from \(-N\) to \(N\) for cross-covariance functions, where \(N\) is the value of the NLAG= option.

• VAR, a character variable that contains the name of the variable specified by the VAR= option.
Chapter 8: The ARIMA Procedure

- CROSSVAR, a character variable that contains the name of the variable specified in the CROSSCORR= option, which labels the different cross-covariance functions. The CROSSVAR variable is blank for the autocovariance observations. When there is no CROSSCORR= option, this variable is not created.

- N, a numeric variable that contains the number of observations used to calculate the current value of the covariance or cross-covariance function.

- COV, a numeric variable that contains the autocovariance or cross-covariance function values. COV contains the autocovariances of the VAR= variable when the value of the CROSSVAR variable is blank. Otherwise COV contains the cross covariances between the VAR= variable and the variable named by the CROSSVAR variable.

- CORR, a numeric variable that contains the autocorrelation or cross-correlation function values. CORR contains the autocorrelations of the VAR= variable when the value of the CROSSVAR variable is blank. Otherwise CORR contains the cross-correlations between the VAR= variable and the variable named by the CROSSVAR variable.

- STDERR, a numeric variable that contains the standard errors of the autocorrelations. The standard error estimate is based on the hypothesis that the process that generates the time series is a pure moving-average process of order LAG–1. For the cross-correlations, STDERR contains the value $1/\sqrt{n}$, which approximates the standard error under the hypothesis that the two series are uncorrelated.

- INVCORR, a numeric variable that contains the inverse autocorrelation function values of the VAR= variable. For cross-correlation observations (that is, when the value of the CROSSVAR variable is not blank), INVCORR contains missing values.

- PARTCORR, a numeric variable that contains the partial autocorrelation function values of the VAR= variable. For cross-correlation observations (that is, when the value of the CROSSVAR variable is not blank), PARTCORR contains missing values.

OUTEST= Data Set

PROC ARIMA writes the parameter estimates for a model to an output data set when the OUTEST= option is specified in the ESTIMATE statement. The OUTEST= data set contains the following:

- the BY variables

- _MODLABEL_, a character variable that contains the model label, if it is provided by using the label option in the ESTIMATE statement (otherwise this variable is not created).

- _NAME_, a character variable that contains the name of the parameter for the covariance or correlation observations or is blank for the observations that contain the parameter estimates. (This variable is not created if neither OUTCOV nor OUTCORR is specified.)

- _TYPE_, a character variable that identifies the type of observation. A description of the _TYPE_ variable values is given below.

- variables for model parameters

The variables for the model parameters are named as follows:
<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERRORVAR</td>
<td>This numeric variable contains the variance estimate. The <em>TYPE</em>=EST observation for this variable contains the estimated error variance, and the remaining observations are missing.</td>
</tr>
<tr>
<td>MU</td>
<td>This numeric variable contains values for the mean parameter for the model. (This variable is not created if NOCONSTANT is specified.)</td>
</tr>
<tr>
<td>MAj_k</td>
<td>These numeric variables contain values for the moving-average parameters. The variables for moving-average parameters are named MAj_k, where j is the factor-number and k is the index of the parameter within a factor.</td>
</tr>
<tr>
<td>ARj_k</td>
<td>These numeric variables contain values for the autoregressive parameters. The variables for autoregressive parameters are named ARj_k, where j is the factor number and k is the index of the parameter within a factor.</td>
</tr>
<tr>
<td>Ij_k</td>
<td>These variables contain values for the transfer function parameters. Variables for transfer function parameters are named Ij_k, where j is the number of the INPUT variable associated with the transfer function component and k is the number of the parameter for the particular INPUT variable. INPUT variables are numbered according to the order in which they appear in the INPUT= list.</td>
</tr>
<tr>
<td><em>STATUS</em></td>
<td>This variable describes the convergence status of the model. A value of 0_CONVERGED indicates that the model converged.</td>
</tr>
</tbody>
</table>

The value of the _TYPE_ variable for each observation indicates the kind of value contained in the variables for model parameters for the observation. The OUTEST= data set contains observations with the following _TYPE_ values:

- **EST**: The observation contains parameter estimates.
- **STD**: The observation contains approximate standard errors of the estimates.
- **CORR**: The observation contains correlations of the estimates. OUTCORR must be specified to get these observations.
- **COV**: The observation contains covariances of the estimates. OUTCOV must be specified to get these observations.
- **FACTOR**: The observation contains values that identify for each parameter the factor that contains it. Negative values indicate denominator factors in transfer function models.
- **LAG**: The observation contains values that identify the lag associated with each parameter.
- **SHIFT**: The observation contains values that identify the shift associated with the input series for the parameter.

The values given for _TYPE_=FACTOR, _TYPE_=LAG, or _TYPE_=SHIFT observations enable you to reconstruct the model employed when provided with only the OUTEST= data set.

**OUTEST= Examples**

This section clarifies how model parameters are stored in the OUTEST= data set with two examples.

Consider the following example:
The model specified by these statements is

\[ Y_t = \mu + \omega_{1,0}X_{1,t} + \omega_{2,0}X_{2,t} + \frac{(1 - \theta_{11} B - \theta_{12} B^2)(1 - \theta_{21} B^2)}{(1 - \phi_{11} B)(1 - \phi_{21} B^6)} a_t \]

The OUTEST= data set contains the values shown in Table 8.10.

<table>
<thead>
<tr>
<th>Obs</th>
<th><em>TYPE</em></th>
<th>Y</th>
<th>MU</th>
<th>MA1_1</th>
<th>MA1_2</th>
<th>MA2_1</th>
<th>AR1_1</th>
<th>AR2_1</th>
<th>I1_1</th>
<th>I2_1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>EST</td>
<td>\sigma^2</td>
<td>\mu</td>
<td>\theta_{11}</td>
<td>\theta_{12}</td>
<td>\phi_{11}</td>
<td>\phi_{21}</td>
<td>\omega_{1,0}</td>
<td>\omega_{2,0}</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>STD</td>
<td>.</td>
<td>se \mu</td>
<td>se \theta_{11}</td>
<td>se \theta_{12}</td>
<td>se \phi_{11}</td>
<td>se \phi_{21}</td>
<td>se \omega_{1,0}</td>
<td>se \omega_{2,0}</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>FACTOR</td>
<td>.</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>LAG</td>
<td>.</td>
<td>0</td>
<td>1</td>
<td>3</td>
<td>12</td>
<td>1</td>
<td>6</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>SHIFT</td>
<td>.</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Note that the symbols in the rows for _TYPE_=EST and _TYPE_=STD in Table 8.10 would be numeric values in a real data set.

Next, consider the following example:

\[ Y_t = \mu + \frac{\omega_{10} - \omega_{11} B}{1 - \delta_{11} B - \delta_{12} B^2} X_{1,t-2} + \frac{\omega_{20} - \omega_{21} B}{1 - \delta_{21} B} X_{2,t-1} + \frac{(1 - \theta_1 B)}{(1 - \phi_1 B)} a_t \]

The OUTEST= data set contains the values shown in Table 8.11.

<table>
<thead>
<tr>
<th>Obs</th>
<th><em>TYPE</em></th>
<th>Y</th>
<th>MU</th>
<th>MA1_1</th>
<th>AR1_1</th>
<th>I1_1</th>
<th>I1_2</th>
<th>I1_3</th>
<th>I1_4</th>
<th>I2_1</th>
<th>I2_2</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>EST</td>
<td>\sigma^2</td>
<td>\mu</td>
<td>\theta_1</td>
<td>\phi_1</td>
<td>\omega_{10}</td>
<td>\omega_{11}</td>
<td>\delta_{11}</td>
<td>\delta_{12}</td>
<td>\omega_{20}</td>
<td>\delta_{21}</td>
</tr>
<tr>
<td>2</td>
<td>STD</td>
<td>.</td>
<td>se \mu</td>
<td>se \theta_1</td>
<td>se \phi_1</td>
<td>se \omega_{10}</td>
<td>se \omega_{11}</td>
<td>se \delta_{11}</td>
<td>se \delta_{12}</td>
<td>se \omega_{20}</td>
<td>se \delta_{21}</td>
</tr>
<tr>
<td>3</td>
<td>FACTOR</td>
<td>.</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>LAG</td>
<td>.</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>SHIFT</td>
<td>.</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
OUTMODEL= SAS Data Set

The OUTMODEL= option in the ESTIMATE statement writes an output data set that enables you to reconstruct the model. The OUTMODEL= data set contains much the same information as the OUTEST= data set but in a transposed form that might be more useful for some purposes. In addition, the OUTMODEL= data set includes the differencing operators.

The OUTMODEL data set contains the following:

- the BY variables
- _MODLABEL_, a character variable that contains the model label, if it is provided by using the label option in the ESTIMATE statement (otherwise this variable is not created).
- _NAME_, a character variable that contains the name of the response or input variable for the observation.
- _TYPE_, a character variable that contains the estimation method that was employed. The value of _TYPE_ can be CLS, ULS, or ML.
- _STATUS_, a character variable that describes the convergence status of the model. A value of 0_CONVERGED indicates that the model converged.
- _PARM_, a character variable that contains the name of the parameter given by the observation. _PARM_ takes on the values ERRORVAR, MU, AR, MA, NUM, DEN, and DIF.
- _VALUE_, a numeric variable that contains the value of the estimate defined by the _PARM_ variable.
- _STD_, a numeric variable that contains the standard error of the estimate.
- _FACTOR_, a numeric variable that indicates the number of the factor to which the parameter belongs.
- _LAG_, a numeric variable that contains the number of the term within the factor that contains the parameter.
- _SHIFT_, a numeric variable that contains the shift value for the input variable associated with the current parameter.

The values of _FACTOR_ and _LAG_ identify which particular MA, AR, NUM, or DEN parameter estimate is given by the _VALUE_ variable. The _NAME_ variable contains the response variable name for the MU, AR, or MA parameters. Otherwise, _NAME_ contains the input variable name associated with NUM or DEN parameter estimates. The _NAME_ variable contains the appropriate variable name associated with the current DIF observation as well. The _VALUE_ variable is 1 for all DIF observations, and the _LAG_ variable indicates the degree of differencing employed.

The observations contained in the OUTMODEL= data set are identified by the _PARM_ variable. A description of the values of the _PARM_ variable follows:

NUMRESID _VALUE_ contains the number of residuals.
NPARMS _VALUE_ contains the number of parameters in the model.
Chapter 8: The ARIMA Procedure

NDIFS  _VALUE_ contains the sum of the differencing lags employed for the response variable.
ERRORVAR  _VALUE_ contains the estimate of the innovation variance.
MU  _VALUE_ contains the estimate of the mean term.
AR  _VALUE_ contains the estimate of the autoregressive parameter indexed by the _FACTOR_ and _LAG_ variable values.
MA  _VALUE_ contains the estimate of a moving-average parameter indexed by the _FACTOR_ and _LAG_ variable values.
NUM  _VALUE_ contains the estimate of the parameter in the numerator factor of the transfer function of the input variable indexed by the _FACTOR_, _LAG_, and _SHIFT_ variable values.
DEN  _VALUE_ contains the estimate of the parameter in the denominator factor of the transfer function of the input variable indexed by the _FACTOR_, _LAG_, and _SHIFT_ variable values.
DIF  _VALUE_ contains the difference operator defined by the difference lag given by the value in the _LAG_ variable.

OUTSTAT= Data Set

PROC ARIMA writes the diagnostic statistics for a model to an output data set when the OUTSTAT= option is specified in the ESTIMATE statement. The OUTSTAT data set contains the following:

- the BY variables.
- _MODLABEL_, a character variable that contains the model label, if it is provided by using the label option in the ESTIMATE statement (otherwise this variable is not created).
- _TYPE_, a character variable that contains the estimation method used. _TYPE_ can have the value CLS, ULS, or ML.
- _STAT_, a character variable that contains the name of the statistic given by the _VALUE_ variable in this observation. _STAT_ takes on the values AIC, SBC, LOGLIK, SSE, NUMRESID, NPARMS, NDIFS, ERRORVAR, MU, CONV, and NITER.
- _VALUE_, a numeric variable that contains the value of the statistic named by the _STAT_ variable.

The observations contained in the OUTSTAT= data set are identified by the _STAT_ variable. A description of the values of the _STAT_ variable follows:

AIC  Akaike’s information criterion
SBC  Schwarz’s Bayesian criterion
LOGLIK  the log-likelihood, if METHOD=ML or METHOD=ULS is specified
SSE  the sum of the squared residuals
NUMRESID  the number of residuals
NPARMS the number of parameters in the model
NDIFS the sum of the differencing lags employed for the response variable
ERRORVAR the estimate of the innovation variance
MU the estimate of the mean term
CONV tells if the estimation converged. The value of 0 signifies that estimation converged. Nonzero values reflect convergence problems.
NITER the number of iterations

**Remark.** CONV takes an integer value that corresponds to the error condition of the parameter estimation process. The value of 0 signifies that estimation process has converged. The higher values signify convergence problems of increasing severity. Specifically:

- CONV= 0 indicates that the estimation process has converged.
- CONV= 1 or 2 indicates that the estimation process has run into numerical problems (such as encountering an unstable model or a ridge) during the iterations.
- CONV= 3 or greater indicates that the estimation process has failed to converge.

---

**Printed Output**

The ARIMA procedure produces printed output for each of the IDENTIFY, ESTIMATE, and FORECAST statements. The output produced by each ARIMA statement is described in the following sections.

**IDENTIFY Statement Printed Output**

The printed output of the IDENTIFY statement consists of the following:

- a table of summary statistics, including the name of the response variable, any specified periods of differencing, the mean and standard deviation of the response series after differencing, and the number of observations after differencing

- a plot of the sample autocorrelation function for lags up to and including the NLAG= option value. Standard errors of the autocorrelations also appear to the right of the autocorrelation plot if the value of LINESIZE= option is sufficiently large. The standard errors are derived using Bartlett’s approximation (Box and Jenkins 1976, p. 177). The approximation for a standard error for the estimated autocorrelation function at lag \( k \) is based on a null hypothesis that a pure moving-average Gaussian process of order \( k-1 \) generated the time series. The relative position of an approximate 95% confidence interval under this null hypothesis is indicated by the dots in the plot, while the asterisks represent the relative magnitude of the autocorrelation value.

- a plot of the sample inverse autocorrelation function. See the section “The Inverse Autocorrelation Function” on page 261 for more information about the inverse autocorrelation function.

- a plot of the sample partial autocorrelation function
• a table of test statistics for the hypothesis that the series is white noise. These test statistics are the same as the tests for white noise residuals produced by the ESTIMATE statement and are described in the section “Estimation Details” on page 270.

• a plot of the sample cross-correlation function for each series specified in the CROSSCORR= option. If a model was previously estimated for a variable in the CROSSCORR= list, the cross-correlations for that series are computed for the prewhitened input and response series. For each input variable with a prewhitening filter, the cross-correlation report for the input series includes the following:
  – a table of test statistics for the hypothesis of no cross-correlation between the input and response series
  – the prewhitening filter used for the prewhitening transformation of the predictor and response variables

• ESACF tables if the ESACF option is used

• MINIC table if the MINIC option is used

• SCAN table if the SCAN option is used

• STATIONARITY test results if the STATIONARITY option is used

**ESTIMATE Statement Printed Output**

The printed output of the ESTIMATE statement consists of the following:

• if the PRINTALL option is specified, the preliminary parameter estimates and an iteration history that shows the sequence of parameter estimates tried during the fitting process

• a table of parameter estimates that show the following for each parameter: the parameter name, the parameter estimate, the approximate standard error, $t$ value, approximate probability ($Pr > |t|$), the lag for the parameter, the input variable name for the parameter, and the lag or “Shift” for the input variable

• the estimates of the constant term, the innovation variance (variance estimate), the innovation standard deviation (Std Error Estimate), Akaike’s information criterion (AIC), Schwarz’s Bayesian criterion (SBC), and the number of residuals

• the correlation matrix of the parameter estimates

• a table of test statistics for hypothesis that the residuals of the model are white noise. The table is titled “Autocorrelation Check of Residuals.”

• if the PLOT option is specified, autocorrelation, inverse autocorrelation, and partial autocorrelation function plots of the residuals

• if an INPUT variable has been modeled in such a way that prewhitening is performed in the IDENTIFY step, a table of test statistics titled “Cross-correlation Check of Residuals.” The test statistic is based on the chi-square approximation suggested by Box and Jenkins (1976, pp. 395–396). The cross-correlation function is computed by using the residuals from the model as one series and the prewhitened input variable as the other series.
• if the GRID option is specified, the sum-of-squares or likelihood surface over a grid of parameter values near the final estimates

• a summary of the estimated model that shows the autoregressive factors, moving-average factors, and transfer function factors in backshift notation with the estimated parameter values.

**OUTLIER Statement Printed Output**

The printed output of the OUTLIER statement consists of the following:

• a summary that contains the information about the maximum number of outliers searched, the number of outliers actually detected, and the significance level used in the outlier detection.

• a table that contains the results of the outlier detection process. The outliers are listed in the order in which they are found. This table contains the following columns:
  
  – The Obs column contains the observation number of the start of the level shift.
  – If an ID= option is specified, then the Time ID column contains the time identification labels of the start of the outlier.
  – The Type column lists the type of the outlier.
  – The Estimate column contains $\hat{\beta}$, the estimate of the regression coefficient of the shock signature.
  – The Chi-Square column lists the value of the test statistic $\tau^2$.
  – The Approx Prob > ChiSq column lists the approximate $p$-value of the test statistic.

**FORECAST Statement Printed Output**

The printed output of the FORECAST statement consists of the following:

• a summary of the estimated model

• a table of forecasts with following columns:
  
  – The Obs column contains the observation number.
  – The Forecast column contains the forecast values.
  – The Std Error column contains the forecast standard errors.
  – The Lower and Uppers columns contain the approximate 95% confidence limits. The ALPHA= option can be used to change the confidence interval for forecasts.
  – If the PRINTALL option is specified, the forecast table also includes columns for the actual values of the response series (Actual) and the residual values (Residual).
PROC ARIMA assigns a name to each table it creates. You can use these names to reference the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 8.12.

### Table 8.12  ODS Tables Produced by PROC ARIMA

<table>
<thead>
<tr>
<th>ODS Table Name</th>
<th>Description</th>
<th>Statement</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>ChiSqAuto</td>
<td>chi-square statistics table for autocorrelation</td>
<td>IDENTIFY</td>
<td></td>
</tr>
<tr>
<td>ChiSqCross</td>
<td>chi-square statistics table for cross-correlations</td>
<td>IDENTIFY</td>
<td>CROSSCORR</td>
</tr>
<tr>
<td>AutoCorrGraph</td>
<td>Correlations graph</td>
<td>IDENTIFY</td>
<td></td>
</tr>
<tr>
<td>CrossCorrGraph</td>
<td>Cross-Correlations graph</td>
<td>IDENTIFY</td>
<td></td>
</tr>
<tr>
<td>DescStats</td>
<td>Descriptive statistics</td>
<td>IDENTIFY</td>
<td></td>
</tr>
<tr>
<td>ESACF</td>
<td>Extended sample autocorrelation function</td>
<td>IDENTIFY</td>
<td>ESACF</td>
</tr>
<tr>
<td>ESACFPValues</td>
<td>ESACF probability values</td>
<td>IDENTIFY</td>
<td>ESACF</td>
</tr>
<tr>
<td>IACFGraph</td>
<td>Inverse autocorrelations graph</td>
<td>IDENTIFY</td>
<td></td>
</tr>
<tr>
<td>InputDescStats</td>
<td>Input descriptive statistics</td>
<td>IDENTIFY</td>
<td></td>
</tr>
<tr>
<td>MINIC</td>
<td>Minimum information criterion</td>
<td>IDENTIFY</td>
<td>MINIC</td>
</tr>
<tr>
<td>PACFGraph</td>
<td>Partial autocorrelations graph</td>
<td>IDENTIFY</td>
<td>SCAN</td>
</tr>
<tr>
<td>SCAN</td>
<td>Squared canonical correlation estimates</td>
<td>IDENTIFY</td>
<td>SCAN</td>
</tr>
<tr>
<td>SCANPValues</td>
<td>SCAN chi-square probability values</td>
<td>IDENTIFY</td>
<td></td>
</tr>
<tr>
<td>StationarityTests</td>
<td>Stationarity tests</td>
<td>IDENTIFY</td>
<td>STATIONARITY</td>
</tr>
<tr>
<td>TentativeOrders</td>
<td>Tentative order selection</td>
<td>IDENTIFY</td>
<td>MINIC, ESACF, or SCAN</td>
</tr>
<tr>
<td>ARPolynomial</td>
<td>Filter equations</td>
<td>ESTIMATE</td>
<td></td>
</tr>
<tr>
<td>ChiSqAuto</td>
<td>chi-square statistics table for autocorrelation</td>
<td>ESTIMATE</td>
<td></td>
</tr>
<tr>
<td>ChiSqCross</td>
<td>chi-square statistics table for cross-correlations</td>
<td>ESTIMATE</td>
<td></td>
</tr>
<tr>
<td>CorrB</td>
<td>Correlations of the estimates</td>
<td>ESTIMATE</td>
<td></td>
</tr>
<tr>
<td>DenPolynomial</td>
<td>Filter equations</td>
<td>ESTIMATE</td>
<td></td>
</tr>
<tr>
<td>FitStatistics</td>
<td>Fit statistics</td>
<td>ESTIMATE</td>
<td></td>
</tr>
<tr>
<td>IterHistory</td>
<td>Iteration history</td>
<td>ESTIMATE</td>
<td>PRINTALL</td>
</tr>
<tr>
<td>InitialAREstimates</td>
<td>Initial autoregressive parameter estimates</td>
<td>ESTIMATE</td>
<td></td>
</tr>
<tr>
<td>InitialMAEstimates</td>
<td>Initial moving-average parameter estimates</td>
<td>ESTIMATE</td>
<td></td>
</tr>
<tr>
<td>InputDescription</td>
<td>Input description</td>
<td>ESTIMATE</td>
<td></td>
</tr>
<tr>
<td>MAPolynomial</td>
<td>Filter equations</td>
<td>ESTIMATE</td>
<td></td>
</tr>
</tbody>
</table>
 Statistical Graphics


Before you create graphs, ODS Graphics must be enabled (for example, with the ODS GRAPHICS ON statement). For more information about enabling and disabling ODS Graphics, see the section “Enabling and Disabling ODS Graphics” in that chapter.

The overall appearance of graphs is controlled by ODS styles. Styles and other aspects of using ODS Graphics are discussed in the section “A Primer on ODS Statistical Graphics” in that chapter.

This section provides information about the graphics produced by the ARIMA procedure. (See Chapter 21, “Statistical Graphics Using ODS” (SAS/STAT User’s Guide), for more information about ODS statistical graphics.) The main types of plots available are as follows:

- plots useful in the trend and correlation analysis of the dependent and input series
- plots useful for the residual analysis of an estimated model
- forecast plots

You can obtain most plots relevant to the specified model by default. For finer control of the graphics, you can use the PLOTS= option in the PROC ARIMA statement. The following example is a simple illustration of how to use the PLOTS= option.

**Airline Series: Illustration of ODS Graphics**

The series in this example, the monthly airline passenger series, is also discussed later, in Example 8.2.

The following statements specify an ARIMA(0,1,1)×(0,1,1)_{12} model without a mean term to the logarithms of the airline passengers series, xlog. Notice the use of the global plot option ONLY in the PLOTS= option.
of the PROC ARIMA statement. It suppresses the production of default graphics and produces only the plots specified by the subsequent RESIDUAL and FORECAST plot options. The `RESIDUAL(SMOOTH)` plot specification produces a time series plot of residuals that has an overlaid loess fit; see Figure 8.21. The `FORECAST(FORECAST)` option produces a plot that shows the one-step-ahead forecasts, as well as the multistep-ahead forecasts; see Figure 8.22.

```plaintext
proc arima data=seriesg
   plots(only)=(residual(smooth) forecast(forecasts));
   identify var=xlog(1,12);
   estimate q=(1)(12) noint method=ml;
   forecast id=date interval=month;
run;
```

**Figure 8.21** Residual Plot of the Airline Model

![Residual Plot of the Airline Model](image)
Figure 8.22 Forecast Plot of the Airline Model

ODS Graph Names

PROC ARIMA assigns a name to each graph it creates by using ODS. You can use these names to reference the graphs when you use ODS. The names are listed in Table 8.13.
<table>
<thead>
<tr>
<th>ODS Graph Name</th>
<th>Plot Description</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>SeriesPlot</td>
<td>Time series plot of the dependent series</td>
<td>PLOTS(UNPACK)</td>
</tr>
<tr>
<td>SeriesACFPlot</td>
<td>Autocorrelation plot of the dependent series</td>
<td>PLOTS(UNPACK)</td>
</tr>
<tr>
<td>SeriesPACFPlot</td>
<td>Partial-autocorrelation plot of the dependent series</td>
<td>PLOTS(UNPACK)</td>
</tr>
<tr>
<td>SeriesIACFPlot</td>
<td>Inverse-autocorrelation plot of the dependent series</td>
<td>PLOTS(UNPACK)</td>
</tr>
<tr>
<td>SeriesCorrPanel</td>
<td>Series trend and correlation analysis panel</td>
<td>Default</td>
</tr>
<tr>
<td>CrossCorrPanel</td>
<td>Cross-correlation plots, either individual or paneled. They are numbered 1, 2, and so on as needed.</td>
<td>Default</td>
</tr>
<tr>
<td>ResidualACFPlot</td>
<td>Residual-autocorrelation plot</td>
<td>PLOTS(UNPACK)</td>
</tr>
<tr>
<td>ResidualPACFPlot</td>
<td>Residual-partial-autocorrelation plot</td>
<td>PLOTS(UNPACK)</td>
</tr>
<tr>
<td>ResidualIACFPlot</td>
<td>Residual-inverse-autocorrelation plot</td>
<td>PLOTS(UNPACK)</td>
</tr>
<tr>
<td>ResidualWNPlot</td>
<td>Residual-white-noise-probability plot</td>
<td>PLOTS(UNPACK)</td>
</tr>
<tr>
<td>ResidualHistogram</td>
<td>Residual histogram</td>
<td>PLOTS(UNPACK)</td>
</tr>
<tr>
<td>ResidualQQPlot</td>
<td>Residual normal Q-Q Plot</td>
<td>PLOTS(UNPACK)</td>
</tr>
<tr>
<td>ResidualPlot</td>
<td>Time series plot of residuals with a superimposed smoother</td>
<td>PLOTS=RESIDUAL(SMOOTH)</td>
</tr>
<tr>
<td>ForecastsOnlyPlot</td>
<td>Time series plot of multistep forecasts</td>
<td>Default</td>
</tr>
<tr>
<td>ForecastsPlot</td>
<td>Time series plot of one-step-ahead as well as multistep forecasts</td>
<td>PLOTS=FORECAST(FORECAST)</td>
</tr>
</tbody>
</table>
**Examples: ARIMA Procedure**

### Example 8.1: Simulated IMA Model

This example illustrates the ARIMA procedure results for a case where the true model is known. An integrated moving-average model is used for this illustration.

The following DATA step generates a pseudo-random sample of 100 periods from the ARIMA(0,1,1) process $u_t = u_{t-1} + a_t - 0.8a_{t-1}, a_t \text{ iid } N(0, 1)$:

```plaintext
title1 'Simulated IMA(1,1) Series';
data a;
  u1 = 0.9; a1 = 0;
  do i = -50 to 100;
    a = rannor( 32565 );
    u = u1 + a - .8 * a1;
    if i > 0 then output;
    a1 = a;
    u1 = u;
  end;
run;
```

The following ARIMA procedure statements identify and estimate the model:

```plaintext
/*** Simulated IMA Model ---**/
proc arima data=a;
  identify var=u;
  run;
  identify var=u(1);
  run;
  estimate q=1 ;
  run;
quit;
```

The graphical series correlation analysis output of the first IDENTIFY statement is shown in **Output 8.1.1**. The output shows the behavior of the sample autocorrelation function when the process is nonstationary. Note that in this case the estimated autocorrelations are not very high, even at small lags. Nonstationarity is reflected in a pattern of significant autocorrelations that do not decline quickly with increasing lag, not in the size of the autocorrelations.
Output 8.1.1  Correlation Analysis from the First IDENTIFY Statement

The second IDENTIFY statement differences the series. The results of the second IDENTIFY statement are shown in Output 8.1.2. This output shows autocorrelation, inverse autocorrelation, and partial autocorrelation functions typical of MA(1) processes.
Output 8.1.2 Correlation Analysis from the Second IDENTIFY Statement

![Trend and Correlation Analysis for u(1)](image)

The ESTIMATE statement fits an ARIMA(0,1,1) model to the simulated data. Note that in this case the parameter estimates are reasonably close to the values used to generate the simulated data. \((\mu = 0, \hat{\mu} = 0.02; \theta_1 = 0.8, \hat{\theta}_1 = 0.79; \sigma^2 = 1, \hat{\sigma}^2 = 0.82)\). Moreover, the graphical analysis of the residuals shows no model inadequacies (see Output 8.1.4 and Output 8.1.5).

The ESTIMATE statement results are shown in Output 8.1.3.

### Output 8.1.3 Output from Fitting ARIMA(0,1,1) Model

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>t Value</th>
<th>Approx Pr &gt;</th>
<th>Lag</th>
</tr>
</thead>
<tbody>
<tr>
<td>MU</td>
<td>0.02056</td>
<td>0.01972</td>
<td>1.04</td>
<td>0.2997</td>
<td>0</td>
</tr>
<tr>
<td>MA1,1</td>
<td>0.79142</td>
<td>0.06474</td>
<td>12.22</td>
<td>&lt;.0001</td>
<td>1</td>
</tr>
</tbody>
</table>
Output 8.1.3 continued

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant Estimate</td>
<td>0.020558</td>
</tr>
<tr>
<td>Variance Estimate</td>
<td>0.819807</td>
</tr>
<tr>
<td>Std Error Estimate</td>
<td>0.905432</td>
</tr>
<tr>
<td>AIC</td>
<td>263.2594</td>
</tr>
<tr>
<td>SBC</td>
<td>268.4497</td>
</tr>
<tr>
<td>Number of Residuals</td>
<td>99</td>
</tr>
</tbody>
</table>

Model for variable u

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimated Mean</td>
<td>0.020558</td>
</tr>
<tr>
<td>Period(s) of Differencing</td>
<td>1</td>
</tr>
</tbody>
</table>

Moving Average Factors

Factor 1:  1 - 0.79142 B**(1)

Output 8.1.4 Residual Correlation Analysis of the ARIMA(0,1,1) Model

Residual Correlation Diagnostics for u(1)
Example 8.2: Seasonal Model for the Airline Series

The airline passenger data, given as Series G in Box and Jenkins (1976), have been used in time series analysis literature as an example of a nonstationary seasonal time series. This example uses PROC ARIMA to fit the airline model, ARIMA(0,1,1) \times (0,1,1)_{12}, to Box and Jenkins’ Series G. The following statements read the data and log-transform the series:

```sas
title1 'International Airline Passengers';
title2 '(Box and Jenkins Series-G)';
data seriesg;
   input x @@;
xlog = log( x );
date = intnx( 'month', '31dec1948'd, _n_ );
format date monyy.;
datalines;
112 118 132 129 121 135 148 148 136 119 104 118
... more lines ...
```

The following PROC TIMESERIES step plots the series, as shown in Output 8.2.1:

```sas
proc timeseries data=seriesg plot=series;
   id date interval=month;
   var x;
run;
```
Chapter 8: The ARIMA Procedure

Output 8.2.1 Time Series Plot of the Airline Passenger Series

The following statements specify an ARIMA(0,1,1)×(0,1,1)_{12} model without a mean term to the logarithms of the airline passengers series, \textit{xlog}. The model is forecast, and the results are stored in the data set \textit{B}.

```bash
/--- Seasonal Model for the Airline Series ---*/
proc arima data=seriesg;
   identify var=xlog(1,12);
   estimate q=(1)(12) noint method=ml;
   forecast id=date interval=month printall out=b;
run;
```

The output from the IDENTIFY statement is shown in Output 8.2.2. The autocorrelation plots shown are for the twice differenced series \((1 - B)(1 - B^{12})XLOG\). Note that the autocorrelation functions have the pattern characteristic of a first-order moving-average process combined with a seasonal moving-average process with lag 12.
Output 8.2.2 IDENTIFY Statement Output

International Airline Passengers
(Box and Jenkins Series-G)

The ARIMA Procedure

<table>
<thead>
<tr>
<th>Name of Variable = xlog</th>
</tr>
</thead>
<tbody>
<tr>
<td>Period(s) of Differencing</td>
</tr>
<tr>
<td>Mean of Working Series</td>
</tr>
<tr>
<td>Standard Deviation</td>
</tr>
<tr>
<td>Number of Observations</td>
</tr>
<tr>
<td>Observation(s) eliminated by differencing</td>
</tr>
</tbody>
</table>

Output 8.2.3 Trend and Correlation Analysis for the Twice Differenced Series

The results of the ESTIMATE statement are shown in Output 8.2.4, Output 8.2.5, and Output 8.2.6. The model appears to fit the data quite well.
Output 8.2.4  ESTIMATE Statement Output

| Parameter | Estimate | Standard Error | t Value | Pr > |t| | Lag |
|-----------|----------|----------------|---------|-------|---|-----|
| MA1,1     | 0.40194  | 0.07988        | 5.03    | <.0001|   | 1   |
| MA2,1     | 0.55686  | 0.08403        | 6.63    | <.0001|   | 12  |

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Variance Estimate</td>
<td>0.001369</td>
</tr>
<tr>
<td>Std Error Estimate</td>
<td>0.037</td>
</tr>
<tr>
<td>AIC</td>
<td>-485.393</td>
</tr>
<tr>
<td>SBC</td>
<td>-479.643</td>
</tr>
<tr>
<td>Number of Residuals</td>
<td>131</td>
</tr>
</tbody>
</table>

Model for variable xlog

| Period(s) of Differencing | 1.12 |

Moving Average Factors

| Factor 1: | 1 - 0.40194 B**(1) |
| Factor 2: | 1 - 0.55686 B**(12) |
Output 8.2.5 Residual Analysis of the Airline Model: Correlation

Residual Correlation Diagnostics for xlog(1 12)

ACF

PACF

IACF

White Noise Prob
The forecasts and their confidence limits for the transformed series are shown in Output 8.2.7.
The following statements retransform the forecast values to get forecasts in the original scales. See the section “Forecasting Log Transformed Data” on page 279 for more information.

```
data c;
  set b;
  x = exp( xlog );
  forecast = exp( forecast + std*std/2 );
  l95 = exp( l95 );
  u95 = exp( u95 );
run;
```

The forecasts and their confidence limits are plotted by using the following PROC SGPLOT step. The plot is shown in Output 8.2.8.

```
proc sgplot data=c;
  where date >= '1jan58'd;
  band Upper=u95 Lower=l95 x=date
    / LegendLabel="95% Confidence Limits";
  scatter x=date y=x;
  series x=date y=forecast;
run;
```
Example 8.3: Model for Series J Data from Box and Jenkins

This example uses the Series J data from Box and Jenkins (1976). First, the input series X is modeled with a univariate ARMA model. Next, the dependent series Y is cross-correlated with the input series. Since a model has been fit to X, both Y and X are prewhitened by this model before the sample cross-correlations are computed. Next, a transfer function model is fit with no structure on the noise term. The residuals from this model are analyzed; then, the full model, transfer function and noise, is fit to the data.

The following statements read 'Input Gas Rate' and 'Output CO'2 from a gas furnace. (Data values are not shown. The full example including data is in the SAS/ETS sample library.)

```
title1 'Gas Furnace Data';
title2 '(Box and Jenkins, Series J)';
data seriesj;
  input x y @@;
  label x = 'Input Gas Rate'
         y = 'Output CO2';
datalines;
-0.109 53.8 0.000 53.6 0.178 53.5
... more lines ...
```
The following statements produce Output 8.3.1 through Output 8.3.11:

```sas
proc arima data=seriesj;
    /*--- Look at the input process -----------------------------*/
    identify var=x;
    run;

    /*--- Fit a model for the input ----------------------------*/
    estimate p=3 plot;
    run;

    /*--- Cross-correlation of prewhitened series --------------*/
    identify var=y crosscorr=(x) nlag=12;
    run;

    /*--- Fit a simple transfer function - look at residuals ----*/
    estimate input=( 3 $ (1,2)/(1) x );
    run;

    /*--- Final Model - look at residuals ----------------------*/
    estimate p=2 input=( 3 $ (1,2)/(1) x );
    run;
quit;
```

The results of the first IDENTIFY statement for the input series X are shown in Output 8.3.1. The correlation analysis suggests an AR(3) model.

**Output 8.3.1 IDENTIFY Statement Results for X**

**Gas Furnace Data**  
**(Box and Jenkins, Series J)**

**The ARIMA Procedure**

<table>
<thead>
<tr>
<th>Name of Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean of Working Series</td>
<td>-0.05683</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>1.070952</td>
</tr>
<tr>
<td>Number of Observations</td>
<td>296</td>
</tr>
</tbody>
</table>
Output 8.3.2 IDENTIFY Statement Results for X: Trend and Correlation

The ESTIMATE statement results for the AR(3) model for the input series X are shown in Output 8.3.3.

Output 8.3.3 Estimates of the AR(3) Model for X

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>t Value</th>
<th>Approx Pr &gt;</th>
<th>Lag</th>
</tr>
</thead>
<tbody>
<tr>
<td>MU</td>
<td>-0.12280</td>
<td>0.10902</td>
<td>-1.13</td>
<td>0.2609</td>
<td>0</td>
</tr>
<tr>
<td>AR1,1</td>
<td>1.97607</td>
<td>0.05499</td>
<td>35.94</td>
<td>&lt;.0001</td>
<td>1</td>
</tr>
<tr>
<td>AR1,2</td>
<td>-1.37499</td>
<td>0.09967</td>
<td>-13.80</td>
<td>&lt;.0001</td>
<td>2</td>
</tr>
<tr>
<td>AR1,3</td>
<td>0.34336</td>
<td>0.05502</td>
<td>6.24</td>
<td>&lt;.0001</td>
<td>3</td>
</tr>
</tbody>
</table>

Constant Estimate: -0.00682
Variance Estimate: 0.035797
Std Error Estimate: 0.1892
AIC: -141.667
SBC: -126.906
Number of Residuals: 296
The IDENTIFY statement results for the dependent series Y cross-correlated with the input series X are shown in Output 8.3.4, Output 8.3.5, Output 8.3.6, and Output 8.3.7. Since a model has been fit to X, both Y and X are prewhitened by this model before the sample cross-correlations are computed.

**Output 8.3.3 continued**

<table>
<thead>
<tr>
<th>Model for variable x</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimated Mean</td>
</tr>
</tbody>
</table>

**Autoregressive Factors**

Factor 1: \(1 - 1.97607 B^{**(1)} + 1.37499 B^{**(2)} - 0.34336 B^{**(3)}\)

**Output 8.3.4 Summary Table: Y Cross-Correlated with X**

<table>
<thead>
<tr>
<th>Correlation of y and x</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Observations</td>
</tr>
<tr>
<td>Variance of transformed series y</td>
</tr>
<tr>
<td>Variance of transformed series x</td>
</tr>
</tbody>
</table>

Both series have been prewhitened.

**Output 8.3.5 Prewhitening Filter**

<table>
<thead>
<tr>
<th>Autoregressive Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor 1: (1 - 1.97607 B^{<strong>(1)} + 1.37499 B^{</strong>(2)} - 0.34336 B^{**(3)})</td>
</tr>
</tbody>
</table>
Output 8.3.6 IDENTIFY Statement Results for Y: Trend and Correlation

Trend and Correlation Analysis for y

- Time series plot of Y vs. observation.
- Autocorrelation function (ACF) for Y across lags.
- Partial autocorrelation function (PACF) for Y across lags.
- Inverse autocorrelation function (IACF) for Y across lags.
Output 8.3.7 IDENTIFY Statement for Y Cross-Correlated with X

The ESTIMATE statement results for the transfer function model with no structure on the noise term are shown in Output 8.3.8, Output 8.3.9, and Output 8.3.10.

Output 8.3.8 Estimation Output of the First Transfer Function Model

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>t Value</th>
<th>Pr &gt;</th>
<th>Lag</th>
<th>Variable</th>
<th>Shift</th>
</tr>
</thead>
<tbody>
<tr>
<td>MU</td>
<td>53.32256</td>
<td>0.04926</td>
<td>1082.51</td>
<td>&lt;.0001</td>
<td>0</td>
<td>y</td>
<td>0</td>
</tr>
<tr>
<td>NUM1</td>
<td>-0.56467</td>
<td>0.22405</td>
<td>-2.52</td>
<td>0.0123</td>
<td>0</td>
<td>x</td>
<td>3</td>
</tr>
<tr>
<td>NUM1,1</td>
<td>0.42623</td>
<td>0.46472</td>
<td>0.92</td>
<td>0.3598</td>
<td>1</td>
<td>x</td>
<td>3</td>
</tr>
<tr>
<td>NUM1,2</td>
<td>0.29914</td>
<td>0.35506</td>
<td>0.84</td>
<td>0.4002</td>
<td>2</td>
<td>x</td>
<td>3</td>
</tr>
<tr>
<td>DEN1,1</td>
<td>0.60073</td>
<td>0.04101</td>
<td>14.65</td>
<td>&lt;.0001</td>
<td>1</td>
<td>x</td>
<td>3</td>
</tr>
</tbody>
</table>

Constant Estimate: 53.32256
Variance Estimate: 0.702625
Std Error Estimate: 0.838227
AIC: 728.0754
SBC: 746.442
Number of Residuals: 291
Output 8.3.9 Model Summary: First Transfer Function Model

<table>
<thead>
<tr>
<th>Model for variable y</th>
</tr>
</thead>
<tbody>
<tr>
<td>Estimated Intercept  53.32256</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input Number 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Variable  x</td>
</tr>
<tr>
<td>Shift 3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Numerator Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor 1: -0.5647 - 0.42623 B**(1) - 0.29914 B**(2)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Denominator Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor 1: 1 - 0.60073 B**(1)</td>
</tr>
</tbody>
</table>

Output 8.3.10 Residual Analysis: First Transfer Function Model

The residual correlation analysis suggests an AR(2) model for the noise part of the model. The ESTIMATE statement results for the final transfer function model with AR(2) noise are shown in Output 8.3.11.
Output 8.3.11  Estimation Output of the Final Model

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>t Value</th>
<th>Approx Pr &gt;</th>
<th>Lag</th>
<th>Variable</th>
<th>Shift</th>
</tr>
</thead>
<tbody>
<tr>
<td>MU</td>
<td>53.26304</td>
<td>0.11929</td>
<td>446.48</td>
<td>&lt;.0001</td>
<td>y</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>AR1,1</td>
<td>1.53291</td>
<td>0.04754</td>
<td>32.25</td>
<td>&lt;.0001</td>
<td>1</td>
<td>y</td>
<td>0</td>
</tr>
<tr>
<td>AR1,2</td>
<td>-0.63297</td>
<td>0.05006</td>
<td>-12.64</td>
<td>&lt;.0001</td>
<td>2</td>
<td>y</td>
<td>0</td>
</tr>
<tr>
<td>NUM1</td>
<td>-0.53522</td>
<td>0.07482</td>
<td>-7.15</td>
<td>&lt;.0001</td>
<td>0</td>
<td>x</td>
<td>3</td>
</tr>
<tr>
<td>NUM1,1</td>
<td>0.37603</td>
<td>0.10287</td>
<td>3.66</td>
<td>0.0033</td>
<td>1</td>
<td>x</td>
<td>3</td>
</tr>
<tr>
<td>NUM1,2</td>
<td>0.51895</td>
<td>0.10783</td>
<td>4.81</td>
<td>&lt;.0001</td>
<td>2</td>
<td>x</td>
<td>3</td>
</tr>
<tr>
<td>DEN1,1</td>
<td>0.54841</td>
<td>0.03822</td>
<td>14.35</td>
<td>&lt;.0001</td>
<td>1</td>
<td>x</td>
<td>3</td>
</tr>
</tbody>
</table>

Constant Estimate 5.329425
Variance Estimate 0.058828
Std Error Estimate 0.242544
AIC 8.292809
SBC 34.00607
Number of Residuals 291

Output 8.3.12  Residual Analysis of the Final Model

Residual Correlation Diagnostics for y

[Graphs of ACF and PACF for y, IACF for y, White Noise Prob]
### Example 8.4: An Intervention Model for Ozone Data

This example fits an intervention model to ozone data as suggested by Box and Tiao (1975). Notice that the response variable, **OZONE**, and the innovation, **X1**, are seasonally differenced. The final model for the differenced data is a multiple regression model with a moving-average structure assumed for the residuals.

The model is fit by maximum likelihood. The seasonal moving-average parameter and its standard error are fairly sensitive to which method is chosen to fit the model (Ansley and Newbold 1980; Davidson 1981); thus, fitting the model by the unconditional or conditional least squares method produces somewhat different estimates for these parameters.

Some missing values are appended to the end of the input data to generate additional values for the independent variables. Since the independent variables are not modeled, values for them must be available for any times at which predicted values are desired. In this case, predicted values are requested for 12 periods beyond the end of the data. Thus, values for **X1**, **WINTER**, and **SUMMER** must be given for 12 periods ahead.

The following statements read in the data and compute dummy variables for use as intervention inputs:

```plaintext
title1 'Intervention Data for Ozone Concentration';
title2 '(Box and Tiao, JASA 1975 P.70)';
data air;
  input ozone @@;
  label ozone = 'Ozone Concentration'
    x1 = 'Intervention for post 1960 period'
    summer = 'Summer Months Intervention'
    winter = 'Winter Months Intervention';
  date = intnx( 'month', '31dec1954'd, _n_ );
  format date monyy.;
  month = month( date );
  year = year( date );
  x1 = year >= 1960;
  summer = ( 5 < month < 11 ) * ( year > 1965 );
  winter = ( year > 1965 ) - summer;
datalines;
```

---

**Output 8.3.13  Model Summary of the Final Model**

- **Model for variable y**
  - Estimated Intercept: 53.26304

<table>
<thead>
<tr>
<th>Autoregressive Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor 1: 1 - 1.53291 B**(1) + 0.63297 B**(2)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input Number 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input Variable: x</td>
</tr>
<tr>
<td>Shift: 3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Numerator Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor 1: -0.5352 - 0.37603 B**(1) - 0.51895 B**(2)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Denominator Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Factor 1: 1 - 0.54841 B**(1)</td>
</tr>
</tbody>
</table>
Example 8.4: An Intervention Model for Ozone Data

The following statements produce Output 8.4.1 through Output 8.4.3:

```sas
proc arima data=air;
    /* Identify and seasonally difference ozone series */
    identify var=ozone(12)
       crosscorr=( x1(12) summer winter ) noprint;
    /* Fit a multiple regression with a seasonal MA model */
    /* by the maximum likelihood method */
    estimate q=(1)(12) input=( x1 summer winter )
       noconstant method=ml;
    /* Forecast */
    forecast lead=12 id=date interval=month;
run;
```

The ESTIMATE statement results are shown in Output 8.4.1 and Output 8.4.2.

Output 8.4.1 Parameter Estimates

Intervention Data for Ozone Concentration
(Box and Tiao, JASA 1975 P.70)

The ARIMA Procedure

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Estimate</th>
<th>Standard Error</th>
<th>t Value</th>
<th>Pr &gt;</th>
<th>Lag</th>
<th>Variable</th>
<th>Shift</th>
</tr>
</thead>
<tbody>
<tr>
<td>MA1,1</td>
<td>-0.26684</td>
<td>0.06710</td>
<td>-3.98</td>
<td>&lt;.0001</td>
<td>1</td>
<td>ozone</td>
<td>0</td>
</tr>
<tr>
<td>MA2,1</td>
<td>0.76665</td>
<td>0.05973</td>
<td>12.83</td>
<td>&lt;.0001</td>
<td>12</td>
<td>ozone</td>
<td>0</td>
</tr>
<tr>
<td>NUM1</td>
<td>-1.33062</td>
<td>0.19236</td>
<td>-6.92</td>
<td>&lt;.0001</td>
<td>0</td>
<td>x1</td>
<td>0</td>
</tr>
<tr>
<td>NUM2</td>
<td>-0.23936</td>
<td>0.05952</td>
<td>-4.02</td>
<td>&lt;.0001</td>
<td>0</td>
<td>summer</td>
<td>0</td>
</tr>
<tr>
<td>NUM3</td>
<td>-0.08021</td>
<td>0.04978</td>
<td>-1.61</td>
<td>0.1071</td>
<td>0</td>
<td>winter</td>
<td>0</td>
</tr>
</tbody>
</table>

Variance Estimate 0.634506
Std Error Estimate 0.796559
AIC 501.7696
SBC 518.3602
Number of Residuals 204
Chapter 8: The ARIMA Procedure

Output 8.4.2 Model Summary

Model for variable ozone

Period(s) of Differencing 12

Moving Average Factors
Factor 1: 1 + 0.26684 B**(1)
Factor 2: 1 - 0.76665 B**(12)

Input Number 1
Input Variable x1
Period(s) of Differencing 12
Overall Regression Factor -1.33062

The FORECAST statement results are shown in Output 8.4.3.

Output 8.4.3 Forecasts

Forecasts for variable ozone

<table>
<thead>
<tr>
<th>Obs</th>
<th>Forecast</th>
<th>Std Error</th>
<th>95% Confidence Limits</th>
</tr>
</thead>
<tbody>
<tr>
<td>217</td>
<td>1.4205</td>
<td>0.7966</td>
<td>-0.1407 2.9817</td>
</tr>
<tr>
<td>218</td>
<td>1.8446</td>
<td>0.8244</td>
<td>0.2287 3.4604</td>
</tr>
<tr>
<td>219</td>
<td>2.4567</td>
<td>0.8244</td>
<td>0.8408 4.0725</td>
</tr>
<tr>
<td>220</td>
<td>2.8590</td>
<td>0.8244</td>
<td>1.2431 4.4748</td>
</tr>
<tr>
<td>221</td>
<td>3.1501</td>
<td>0.8244</td>
<td>1.5342 4.7659</td>
</tr>
<tr>
<td>222</td>
<td>2.7211</td>
<td>0.8244</td>
<td>1.1053 4.3370</td>
</tr>
<tr>
<td>223</td>
<td>3.3147</td>
<td>0.8244</td>
<td>1.6989 4.9306</td>
</tr>
<tr>
<td>224</td>
<td>3.4787</td>
<td>0.8244</td>
<td>1.8629 5.0946</td>
</tr>
<tr>
<td>225</td>
<td>2.9405</td>
<td>0.8244</td>
<td>1.3247 4.5564</td>
</tr>
<tr>
<td>226</td>
<td>2.3587</td>
<td>0.8244</td>
<td>0.7429 3.9746</td>
</tr>
<tr>
<td>227</td>
<td>1.8588</td>
<td>0.8244</td>
<td>0.2429 3.4746</td>
</tr>
<tr>
<td>228</td>
<td>1.2898</td>
<td>0.8244</td>
<td>-0.3260 2.9057</td>
</tr>
</tbody>
</table>

Example 8.5: Using Diagnostics to Identify ARIMA Models

Fitting ARIMA models is as much an art as it is a science. The ARIMA procedure has diagnostic options to help tentatively identify the orders of both stationary and nonstationary ARIMA processes.

Consider the Series A in Box, Jenkins, and Reinsel (1994), which consists of 197 concentration readings taken every two hours from a chemical process. Let Series A be a data set that contains these readings in a variable named X. The following SAS statements use the SCAN option of the IDENTIFY statement to generate Output 8.5.1 and Output 8.5.2. See “The SCAN Method” on page 266 for details of the SCAN method.

```sas
/*-- Order Identification Diagnostic with SCAN Method --*/
proc arima data=SeriesA;
  identify var=x scan;
run;
```
Output 8.5.1 Example of SCAN Tables

SERIES A: Chemical Process Concentration Readings

The ARIMA Procedure

<table>
<thead>
<tr>
<th>Squared Canonical Correlation Estimates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lags</td>
</tr>
<tr>
<td>AR 0</td>
</tr>
<tr>
<td>AR 1</td>
</tr>
<tr>
<td>AR 2</td>
</tr>
<tr>
<td>AR 3</td>
</tr>
<tr>
<td>AR 4</td>
</tr>
<tr>
<td>AR 5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SCAN Chi-Square[1] Probability Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lags</td>
</tr>
<tr>
<td>AR 0</td>
</tr>
<tr>
<td>AR 1</td>
</tr>
<tr>
<td>AR 2</td>
</tr>
<tr>
<td>AR 3</td>
</tr>
<tr>
<td>AR 4</td>
</tr>
<tr>
<td>AR 5</td>
</tr>
</tbody>
</table>

In Output 8.5.1, there is one (maximal) rectangular region in which all the elements are insignificant with 95% confidence. This region has a vertex at (1,1). Output 8.5.2 gives recommendations based on the significance level specified by the ALPHA=siglevel option.

Output 8.5.2 Example of SCAN Option Tentative Order Selection

<table>
<thead>
<tr>
<th>ARMA(p+d,q)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tentative Order Selection Tests</td>
</tr>
<tr>
<td>SCAN p+d q</td>
</tr>
<tr>
<td>1 1</td>
</tr>
</tbody>
</table>

(5% Significance Level)

Another order identification diagnostic is the extended sample autocorrelation function or ESACF method. See “The ESACF Method” on page 263 for details of the ESACF method.

The following statements generate Output 8.5.3 and Output 8.5.4:

```plaintext
/*--- Order Identification Diagnostic with ESACF Method ---*/
proc arima data=SeriesA;
   identify var=x esacf;
run;
```
In Output 8.5.3, there are three right-triangular regions in which all elements are insignificant at the 5% level. The triangles have vertices (1,1), (3,1), and (4,1). Since the triangle at (1,1) covers more insignificant terms, it is recommended first. Similarly, the remaining recommendations are ordered by the number of insignificant terms contained in the triangle. Output 8.5.4 gives recommendations based on the significance level specified by the ALPHA=siglevel option.

If you also specify the SCAN option in the same IDENTIFY statement, the two recommendations are printed side by side:

```plaintext
/*** Combination of SCAN and ESACF Methods ***/
proc arima data=SeriesA;
   identify var=x scan esacf;
run;
```
Example 8.5: Using Diagnostics to Identify ARIMA Models

Output 8.5.5 shows the results.

**Output 8.5.5** Example of SCAN and ESACF Option Combined

**SERIES A: Chemical Process Concentration** Readings

**The ARIMA Procedure**

<table>
<thead>
<tr>
<th>ARMA(p+d,q) Tentative Order Selection Tests</th>
<th>SCAN</th>
<th>ESACF</th>
</tr>
</thead>
<tbody>
<tr>
<td>p+d q p+d q</td>
<td>1 1  1 1</td>
<td>3 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 1</td>
</tr>
</tbody>
</table>

(5% Significance Level)

From **Output 8.5.5**, the autoregressive and moving-average orders are tentatively identified by both SCAN and ESACF tables to be \((p + d, q) = (1, 1)\). Because both the SCAN and ESACF indicate a \(p + d\) term of 1, a unit root test should be used to determine whether this autoregressive term is a unit root. Since a moving-average term appears to be present, a large autoregressive term is appropriate for the augmented Dickey-Fuller test for a unit root.

Submitting the following statements generates **Output 8.5.6**:

```latex
/*-- Augmented Dickey-Fuller Unit Root Tests --*/
proc arima data=SeriesA;
    identify var=x stationarity=(adf=(5,6,7,8));
run;
```

**Output 8.5.6** Example of STATIONARITY Option Output

**SERIES A: Chemical Process Concentration** Readings

**The ARIMA Procedure**

<table>
<thead>
<tr>
<th>Type</th>
<th>Lags</th>
<th>Rho Pr &lt; Rho</th>
<th>Tau Pr &lt; Tau</th>
<th>F Pr &gt; F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Zero Mean</td>
<td>5</td>
<td>0.0403 0.6913 0.42</td>
<td>0.8024</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>0.0479 0.6931 0.63</td>
<td>0.8508</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0.0376 0.6907 0.49</td>
<td>0.8200</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>0.0354 0.6901 0.48</td>
<td>0.8175</td>
<td></td>
</tr>
<tr>
<td>Single Mean</td>
<td>5</td>
<td>-18.4550 0.0267 0.0821 3.67 0.1367</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>-10.8939 0.1023 -2.02 0.2767 2.27 0.4931</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>-10.9224 0.1035 -1.93 0.3172 2.00 0.5605</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>-10.2992 0.1208 -1.83 0.3650 1.81 0.6108</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Trend</td>
<td>5</td>
<td>-18.4360 0.0871 -2.66 0.2561 3.54 0.4703</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>-10.8436 0.3710 -2.01 0.5939 2.04 0.7694</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>-10.7427 0.3773 -1.90 0.6519 1.91 0.7956</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>-10.0370 0.4236 -1.79 0.7081 1.74 0.8293</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The preceding test results show that a unit root is very likely given that none of the \( p \)-values are small enough to cause you to reject the null hypothesis that the series has a unit root. Based on this test and the previous results, the series should be differenced, and an ARIMA(0,1,1) would be a good choice for a tentative model for Series A.

Using the recommendation that the series be differenced, the following statements generate Output 8.5.7:

```sas
/*-- Minimum Information Criterion --*/
proc arima data=SeriesA;
   identify var=x(1) minic;
run;
```

**Output 8.5.7** Example of MINIC Table

**SERIES A: Chemical Process Concentration Readings**

The ARIMA Procedure

<table>
<thead>
<tr>
<th>Lags</th>
<th>MA 0</th>
<th>MA 1</th>
<th>MA 2</th>
<th>MA 3</th>
<th>MA 4</th>
<th>MA 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR 0</td>
<td>-2.05761</td>
<td>-2.3497</td>
<td>-2.32358</td>
<td>-2.31298</td>
<td>-2.30967</td>
<td>-2.28528</td>
</tr>
<tr>
<td>AR 3</td>
<td>-2.25092</td>
<td>-2.28088</td>
<td>-2.25567</td>
<td>-2.23455</td>
<td>-2.22997</td>
<td>-2.20769</td>
</tr>
<tr>
<td>AR 4</td>
<td>-2.25934</td>
<td>-2.2778</td>
<td>-2.25363</td>
<td>-2.22983</td>
<td>-2.20312</td>
<td>-2.19531</td>
</tr>
<tr>
<td>AR 5</td>
<td>-2.2751</td>
<td>-2.26805</td>
<td>-2.24249</td>
<td>-2.21789</td>
<td>-2.19667</td>
<td>-2.17426</td>
</tr>
</tbody>
</table>

The error series is estimated by using an AR(7) model, and the minimum of this MINIC table is \( BIC(0,1) \). This diagnostic confirms the previous result which indicates that an ARIMA(0,1,1) is a tentative model for Series A.

If you also specify the SCAN or MINIC option in the same IDENTIFY statement as follows, the BIC associated with the SCAN table and ESACF table recommendations is listed. Output 8.5.8 shows the results.

```sas
/*-- Combination of MINIC, SCAN and ESACF Options --*/
proc arima data=SeriesA;
   identify var=x(1) minic scan esacf;
run;
```

**Output 8.5.8** Example of SCAN, ESACF, MINIC Options Combined

**SERIES A: Chemical Process Concentration Readings**

The ARIMA Procedure

<table>
<thead>
<tr>
<th>ARMA((p+d,q))</th>
<th>Tentative Order Selection Tests</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCAN</td>
<td>ESACF</td>
</tr>
<tr>
<td>( p+d )</td>
<td>( q )</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

(5% Significance Level)
Example 8.6: Detection of Level Changes in the Nile River Data

This example shows how to use the OUTLIER statement to detect changes in the dynamics of the time series being modeled. The time series used here is discussed in De Jong and Penzer (1998). The data consist of readings of the annual flow volume of the Nile River at Aswan from 1871 to 1970. These data have also been studied by Cobb (1978). These studies indicate that river flow levels in the years 1877 and 1913 are strong candidates for additive outliers and that there was a shift in the flow levels starting from the year 1899. This shift in 1899 is attributed partly to the weather changes and partly to the start of construction work for a new dam at Aswan. The following DATA step statements create the input data set.

```sas
data nile;
  input level @@;
  year = intnx('year', '1jan1871'd, _n_-1);
  format year year4.;
datalines;
1120 1160 963 1210 1160 1160 813 1230 1370 1140
995 935 1110 994 1020 960 1180 799 958 1140
1100 1210 1150 1250 1260 1220 1030 1100 774 840
... more lines ...
```

The following program fits an ARIMA model, ARIMA(0,1,1), similar to the structural model suggested in De Jong and Penzer (1998). This model is also suggested by the usual correlation analysis of the series. By default, the OUTLIER statement requests detection of additive outliers and level shifts, assuming that the series follows the estimated model.

```sas
/*-- ARIMA(0, 1, 1) Model --*/
proc arima data=nile;
  identify var=level(1);
  estimate q=1 noint method=ml;
  outlier maxnum= 5 id=year;
run;
```

The outlier detection output is shown in Output 8.6.1.

--- Output 8.6.1 ARIMA(0, 1, 1) Model ---

The ARIMA Procedure

<table>
<thead>
<tr>
<th>Outlier Detection Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number searched</td>
</tr>
<tr>
<td>Number found</td>
</tr>
<tr>
<td>Significance used</td>
</tr>
</tbody>
</table>
Note that the first three outliers detected are indeed the ones discussed earlier. You can include the shock signatures that correspond to these three outliers in the Nile data set as follows:

```sas
data nile;
  set nile;
  AO1877 = ( year = '1jan1877'd );
  AO1913 = ( year = '1jan1913'd );
  LS1899 = ( year >= '1jan1899'd );
run;
```

Now you can refine the earlier model by including these outliers. After examining the parameter estimates and residuals (not shown) of the ARIMA(0,1,1) model with these regressors, the following stationary MA1 model (with regressors) appears to fit the data well:

```sas
/*-- MA1 Model with Outliers --*/
proc arima data=nile;
  identify var=level
    crosscorr=( AO1877 AO1913 LS1899 );
  estimate q=1
    input=( AO1877 AO1913 LS1899 )
    method=ml;
  outlier maxnum=5 alpha=0.01 id=year;
run;
```

The relevant outlier detection process output is shown in Output 8.6.2. No outliers, at significance level 0.01, were detected.

### Example 8.7: Iterative Outlier Detection

This example illustrates the iterative nature of the outlier detection process. This is done by using a simple test example where an additive outlier at observation number 50 and a level shift at observation number 100 are artificially introduced in the international airline passenger data used in Example 8.2. The following
Example 8.7: Iterative Outlier Detection

DATA step shows the modifications introduced in the data set:

```sas
data airline;
  set sashelp.air;
  logair = log(air);
  if _n_ = 50 then logair = logair - 0.25;
  if _n_ >= 100 then logair = logair + 0.5;
run;
```

In Example 8.2 the airline model, ARIMA(0, 1, 1) × (0, 1, 1)$_{12}$, was seen to be a good fit to the unmodified log-transformed airline passenger series. The preliminary identification steps (not shown) again suggest the airline model as a suitable initial model for the modified data. The following statements specify the airline model and request an outlier search:

```sas
/ *** Outlier Detection ---*/
proc arima data=airline;
  identify var=logair( 1, 12 ) noprint;
  estimate q= (1)(12) noint method= ml;
  outlier maxnum=3 alpha=0.01;
run;
```

The outlier detection output is shown in Output 8.7.1.

**Output 8.7.1** Initial Model

<table>
<thead>
<tr>
<th>The ARIMA Procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outlier Detection Summary</td>
</tr>
<tr>
<td>Maximum number searched</td>
</tr>
<tr>
<td>Number found</td>
</tr>
<tr>
<td>Significance used</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Outlier Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obs</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>100</td>
</tr>
<tr>
<td>50</td>
</tr>
<tr>
<td>135</td>
</tr>
</tbody>
</table>

Clearly the level shift at observation number 100 and the additive outlier at observation number 50 are the dominant outliers. Moreover, the corresponding regression coefficients seem to correctly estimate the size and sign of the change. You can augment the airline data with these two regressors, as follows:

```sas
data airline;
  set airline;
  if _n_ = 50 then AO = 1;
  else AO = 0.0;
  if _n_ >= 100 then LS = 1;
  else LS = 0.0;
run;
```

You can now refine the previous model by including these regressors, as follows. Note that the differencing order of the dependent series is matched to the differencing orders of the outlier regressors to get the correct “effective” outlier signatures.
Chapter 8: The ARIMA Procedure

/*-- Airline Model with Outliers --*/
proc arima data=airline;
  identify var=logair(1, 12)
    crosscorr=( AO(1, 12) LS(1, 12) )
    noprint;
  estimate q= (1)(12) noint
    input=( AO LS )
    method=ml plot;
  outlier maxnum=3 alpha=0.01;
run;

The outlier detection results are shown in Output 8.7.2.

### Output 8.7.2  Airline Model with Outliers

#### The ARIMA Procedure

<table>
<thead>
<tr>
<th>Outlier Detection Summary</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number searched</td>
</tr>
<tr>
<td>Number found</td>
</tr>
<tr>
<td>Significance used</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Obs Type</th>
<th>Estimate</th>
<th>Chi-Square</th>
<th>Approx Prob&gt;ChiSq</th>
</tr>
</thead>
<tbody>
<tr>
<td>135 Additive</td>
<td>-0.10310</td>
<td>12.63</td>
<td>0.0004</td>
</tr>
<tr>
<td>62 Additive</td>
<td>-0.08872</td>
<td>12.33</td>
<td>0.0004</td>
</tr>
<tr>
<td>29 Additive</td>
<td>0.08686</td>
<td>11.66</td>
<td>0.0006</td>
</tr>
</tbody>
</table>

The output shows that a few outliers still remain to be accounted for and that the model could be refined further.

### References


Subject Index

additive model
ARIMA model, 236

AIC, see Akaike information criterion

Akaike information criterion
AIC, 272
ARIMA procedure, 272

ARIMA model
additive model, 236
ARIMA procedure, 216
autoregressive integrated moving-average model, 216
Box-Jenkins model, 216
factored model, 236
multiplicative model, 236
notation for, 231
seasonal model, 236
subset model, 236

ARIMA procedure
Akaike information criterion, 272
ARIMA model, 216
ARIMAX model, 216, 237
ARMA model, 216
autocorrelations, 219
autoregressive parameters, 277
BY groups, 250
conditional forecasts, 278
certainty limits, 278
correlation plots, 219
cross-correlation function, 263
data requirements, 244
differencing, 234, 269, 275
factored model, 236
finite memory forecasts, 279
forecasting, 278, 280
Gauss-Marquardt method, 271
ID variables, 280
infinite memory forecasts, 278
input series, 237
interaction effects, 241
intervention model, 237, 240, 242, 316
inverse autocorrelation function, 261
invertibility, 276
Iterative Outlier Detection, 324
log transformations, 279
Marquardt method, 271
Model Identification, 318
moving-average parameters, 277
naming model parameters, 277
ODS graph names, 295
ODS Graphics, 248
Outlier Detection, 323
output data sets, 282–284, 287, 288
output table names, 292
predicted values, 278
prewhitening, 268, 269
printed output, 289
rational transfer functions, 243
regression model with ARMA errors, 237, 238
residuals, 278
Schwarz Bayesian criterion, 272
seasonal model, 236
stationarity, 220
subset model, 236	syntax, 245
time intervals, 280
transfer function model, 237, 241, 272
unconditional forecasts, 279

ARIMAX model
ARIMA procedure, 216, 237

ARIMAX models and
design matrix, 241

ARMA model
ARIMA procedure, 216
autoregressive moving-average model, 216
notation for, 232
augmented Dickey-Fuller tests, 254, 268
autocorrelations
ARIMA procedure, 219
plotting, 219
autoregressive integrated moving-average model, see ARIMA model
autoregressive moving-average model, see ARMA model
autoregressive parameters
ARIMA procedure, 277

Box-Jenkins model, see ARIMA model
BY groups
ARIMA procedure, 250
conditional forecasts
ARIMA procedure, 278
certainty limits
ARIMA procedure, 278
correlation plots
ARIMA procedure, 219
cross-correlation function
ARIMA procedure, 263

data requirements
   ARIMA procedure, 244
denominator factors
   transfer function model, 242
design matrix
   ARIMAX models and, 241
   Dickey-Fuller tests, 254
der differencing
   ARIMA procedure, 234, 269, 275
dynamic regression, 216, 237
ESACF (Extended Sample Autocorrelation Function method), 263
   Extended Sample Autocorrelation Function (ESACF) method, 263
factored model
   ARIMA model, 236
   ARIMA procedure, 236
finite memory forecasts
   ARIMA procedure, 279
forecasting
   ARIMA procedure, 278, 280
Gauss-Marquardt method
   ARIMA procedure, 271
ID variables
   ARIMA procedure, 280
impulse function
   intervention model and, 240
infinite memory forecasts
   ARIMA procedure, 278
input series
   ARIMA procedure, 237
interaction effects
   ARIMA procedure, 241
interrupted time series analysis, see intervention model
interrupted time series model, see intervention model
intervention analysis, see intervention model
intervention model
   ARIMA procedure, 237, 240, 242, 316
interrupted time series analysis, 240
interrupted time series model, 237
intervention analysis, 240
intervention model and
   impulse function, 240
   step function, 241
inverse autocorrelation function
   ARIMA procedure, 261
invertibility
   ARIMA procedure, 276
Iterative Outlier Detection
   ARIMA procedure, 324
log transformations
   ARIMA procedure, 279
Marquardt method
   ARIMA procedure, 271
MINIC (Minimum Information Criterion) method, 265
   Minimum Information Criterion (MINIC) method, 265
Model Identification
   ARIMA procedure, 318
   moving-average parameters
   ARIMA procedure, 277
multiplicative model
   ARIMA model, 236
naming model parameters
   ARIMA procedure, 277
nonstationarity, see stationarity
notation for
   ARIMA model, 231
   ARIMA model, 232
numerator factors
   transfer function model, 242
ODS graph names
   ARIMA procedure, 295
ODS Graphics
   ARIMA procedure, 248
of time series
   stationarity, 234
Outlier Detection
   ARIMA procedure, 323
output data sets
   ARIMA procedure, 282–284, 287, 288
output table names
   ARIMA procedure, 292
Phillips-Perron tests, 254
plotting
   autocorrelations, 219
predicted values
   ARIMA procedure, 278
prewhitening
   ARIMA procedure, 268, 269
printed output
   ARIMA procedure, 289
random-walk with drift tests, 254
rational transfer functions
   ARIMA procedure, 243
regression model with ARMA errors
   ARIMA procedure, 237, 238
residuals
   ARIMA procedure, 278
SBC, see Schwarz Bayesian criterion

SCAN (Smallest Canonical) correlation method, 266

Schwarz Bayesian criterion
  ARIMA procedure, 272
  SBC, 272

seasonal model
  ARIMA model, 236
  ARIMA procedure, 236

seasonal unit root test, 268

Smallest Canonical (SCAN) correlation method, 266

stationarity
  ARIMA procedure, 220
  nonstationarity, 220
  of time series, 234

stationarity tests, 253, 268

step function
  intervention model and, 241

subset model
  ARIMA model, 236
  ARIMA procedure, 236

time intervals
  ARIMA procedure, 280

transfer function model
  ARIMA procedure, 237, 241, 272
  denominator factors, 242
  numerator factors, 242

unconditional forecasts
  ARIMA procedure, 279

white noise test of the residuals, 256

white noise test of the series, 254
Syntax Index

ALIGN= option
  FORECAST statement (ARIMA), 260

ALPHA= option
  FORECAST statement (ARIMA), 260
  IDENTIFY statement (ARIMA), 251
  OUTLIER statement (ARIMA), 259

ALTPARM option
  IDENTIFY statement (ARIMA), 251

AR= option
  ESTIMATE statement (ARIMA), 255, 275

ARIMA procedure, 245
  syntax, 245

ARIMA procedure, PROC ARIMA statement
  PLOT option, 248

BACK= option
  FORECAST statement (ARIMA), 260
  BACKLIM= option
  ESTIMATE statement (ARIMA), 258

BY statement
  ARIMA procedure, 250

CENTER option
  IDENTIFY statement (ARIMA), 251

CLEAR option
  IDENTIFY statement (ARIMA), 251

CONVERGE= option
  ESTIMATE statement (ARIMA), 258

CROSSCORR= option
  IDENTIFY statement (ARIMA), 251

DATA= option
  IDENTIFY statement (ARIMA), 252
  PROC ARIMA statement, 248

DELTA= option
  ESTIMATE statement (ARIMA), 258

ESACF option
  IDENTIFY statement (ARIMA), 252

ESTIMATE statement
  ARIMA procedure, 255

FORECAST statement
  ARIMA procedure, 260

GRID option
  ESTIMATE statement (ARIMA), 258

GRIDVAL= option
  ESTIMATE statement (ARIMA), 258

ID= option
  FORECAST statement (ARIMA), 260
  OUTLIER statement (ARIMA), 259

IDENTIFY statement
  ARIMA procedure, 251, 259

INITVAL= option
  ESTIMATE statement (ARIMA), 257

INPUT= option
  ESTIMATE statement (ARIMA), 255, 274

INTERVAL= option
  FORECAST statement (ARIMA), 261, 280

LEAD= option
  FORECAST statement (ARIMA), 261

MA= option
  ESTIMATE statement (ARIMA), 257

MAXIT= option
  ESTIMATE statement (ARIMA), 258

MAXITER= option
  ESTIMATE statement (ARIMA), 258

MAXNUM= option
  OUTLIER statement (ARIMA), 259

MAXPCT= option
  OUTLIER statement (ARIMA), 259

METHOD= option
  ESTIMATE statement (ARIMA), 255

MINIC option
  IDENTIFY statement (ARIMA), 252

MU= option
  ESTIMATE statement (ARIMA), 257

NLAG= option
  IDENTIFY statement (ARIMA), 252

NOCONSTANT option
  ESTIMATE statement (ARIMA), 255

NODF option
  ESTIMATE statement (ARIMA), 255

NOEST option
  ESTIMATE statement (ARIMA), 257

NOINT option
  ESTIMATE statement (ARIMA), 255

NOLS option
  ESTIMATE statement (ARIMA), 258

NOMISS option
  IDENTIFY statement (ARIMA), 253

NOOUTALL option
  FORECAST statement (ARIMA), 261

NOPRINT option
ESTIMATE statement (ARIMA), 256
FORECAST statement (ARIMA), 261
IDENTIFY statement (ARIMA), 253
NOSTABLE option
   ESTIMATE statement (ARIMA), 258
NOTFSTABLE option
   ESTIMATE statement (ARIMA), 258

OUT= option
   FORECAST statement (ARIMA), 261, 282
   PROC ARIMA statement, 250
OUTCORR option
   ESTIMATE statement (ARIMA), 256
OUTCOV option
   ESTIMATE statement (ARIMA), 257
OUTCOV= option
   IDENTIFY statement (ARIMA), 253, 283
OUTTEST= option
   ESTIMATE statement (ARIMA), 256, 284
OUTMODEL= option
   ESTIMATE statement (ARIMA), 257, 287
OUTSTAT= option
   ESTIMATE statement (ARIMA), 257, 288

P= option
   ESTIMATE statement (ARIMA), 256
   IDENTIFY statement (ARIMA), 253
PERROR= option
   IDENTIFY statement (ARIMA), 253
PLOT option
   ESTIMATE statement (ARIMA), 256
   PROC ARIMA statement, 248
PLOTS option
   PROC ARIMA statement, 248
PRINTALL option
   ESTIMATE statement (ARIMA), 258
   FORECAST statement (ARIMA), 261
   PROC ARIMA statement, 248

Q= option
   ESTIMATE statement (ARIMA), 256
   IDENTIFY statement (ARIMA), 253

SCAN option
   IDENTIFY statement (ARIMA), 253
SIGMA= option
   OUTLIER statement (ARIMA), 259
SIGSQ= option
   FORECAST statement (ARIMA), 261
SINGULAR= option
   ESTIMATE statement (ARIMA), 258
STATIONARITY= option
   IDENTIFY statement (ARIMA), 253, 254

TYPE= option