SAS/ETS® 15.1 User's Guide

The SSM Procedure

This document is an individual chapter from *SAS/ETS® 15.1 User's Guide*.

The correct bibliographic citation for this manual is as follows: SAS Institute Inc. 2018. *SAS/ETS® 15.1 User's Guide*. Cary, NC: SAS Institute Inc.

SAS/ETS® 15.1 User's Guide

SAS Institute Inc., SAS Campus Drive, Cary, NC 27513-2414

November 2018

Chapter 33
The SSM Procedure

Contents

Overview: SSM Procedure

State space models (SSMs) are used for analyzing continuous response variables that are recorded sequentially according to a numeric indexing variable. In many cases, the indexing variable is time and the observations are collected at regular time intervals—for example, hourly, weekly, or monthly. In such cases, the resulting data are called time series data. In other cases, the indexing variable might not be time or the observations might not be equally spaced according to the indexing variable. These more general types of sequential data are called longitudinal data. Because of their sequential nature, these types of data exhibit some characteristic features. For example, chronologically closer measurements tend to be highly correlated while measurements farther apart are essentially uncorrelated. Data can be trending in a particular direction and can have seasonal or other periodic patterns. SSMs are specially designed to model such sequential data. They apply to both univariate and multivariate response situations and can easily incorporate predictor (independent variable) information when it is available.

The SSM procedure performs state space modeling of univariate and multivariate time series and longitudinal data. You can do the following with the SSM procedure:

- analyze quite general linear state space models
- use an expressive language to specify an SSM. An SSM specification consists of specifying a variety of matrices—for example, the state transition matrix and the covariance matrices of the state and observation disturbances. The SSM procedure provides language similar to a DATA step for specifying the elements of these matrices. The matrix elements can be user-defined functions of data variables and unknown parameters.
- easily specify several commonly needed univariate and multivariate SSMs by using only a few keywords. These SSMs include the principal univariate and multivariate structural models for regularly spaced data and a variety of trend and cycle models for the longitudinal data.
- estimate unknown model parameters by (restricted) maximum likelihood. The likelihood function is computed by using the (diffuse) Kalman filter algorithm.
- print, or output to a data set, the series forecasts, residuals, and the full-sample estimates of any linear combination of the underlying state variables. These estimates are obtained by using the (diffuse) Kalman filter and smoother algorithm.
- generate residual diagnostic plots and plots useful for detecting structural breaks

Background

State space models are widely used in a variety of fields such as engineering, statistics, econometrics, and agriculture. There are numerous references that deal with state space modeling, particularly with the state space modeling of time series data. State space modeling of longitudinal data has received a little less attention. The primary reference for the modeling techniques implemented in the SSM procedure is Harvey (1989). It contains treatment of both the time series and longitudinal data. Other useful books about this subject are Pelagatti (2015); Durbin and Koopman (2012); Jones (1993); Anderson and Moore (1979). In

addition, informative articles about state space modeling of longitudinal data include Wecker and Ansley (1983); Kohn and Ansley (1991); De Jong and Mazzi (2001); Eubank, Huang, and Wang (2003); Selukar (2015). For the implementation details of the diffuse Kalman filter and smoother (the main computational tool used by the SSM procedure), the main references are a series of articles (De Jong 1989, 1991; De Jong and Chu-Chun-Lin 2003) and the references therein.

Getting Started: SSM Procedure

This example illustrates how you can use the SSM procedure to analyze a panel of time series. The following data set, Cigar, contains information about yearly per capita cigarette sales for 46 geographic regions in the United States over the period 1963–1992. The variables lsales, lprice, lndi, and lpimin denote the per capita cigarette sales, price per pack of cigarettes, per capita disposable income, and minimum price in adjoining regions per pack of cigarettes, respectively (all in the natural log scale). The variable year contains the observation year, and the variable region contains an integer between 1 to 46 that serves as the unique identifier for the region. For additional data description see Baltagi and Levin (1992); Baltagi (1995). The data are sorted by year.

```
data cigar;
    input year region lsales lprice lndi lpimin;
    label lsales = 'Log cigarette sales in packs per capita';
    label lprice = 'Log price per pack of cigarettes';
    label lndi = 'Log per capita disposable income';
    label lpimin = 'Log minimum price in adjoining regions
                    per pack of cigarettes';
    year = intnx( 'year', '1jan63'd, year-63 );
    format year year.;
datalines;
63 1 4.54223 3.35341 7.3514 3.26194
63 2 4.82831 3.17388 7.5729 3.21487
63 3 4.63860 3.29584 7.3000 3.25037
63 4 4.95583 3.23080 7.9288 3.17388
63 5 5.05114 3.28840 7.9772 3.26576

   ... more lines ...
```

The goal of the analysis is to study the impact of the regressors on the smoking behavior and to understand the changes in the smoking patterns in different regions over the years. Consider the following model for lsales:

$$\text{lsales}_{i,t} = \boldsymbol{\mu}_{i,t} + \text{lprice}\,\boldsymbol{\beta}_1 + \text{lndi}\,\boldsymbol{\beta}_2 + \text{lpimin}\,\boldsymbol{\beta}_3 + \boldsymbol{\epsilon}_{i,t}$$

This model represents lsales in a region i and in a year t as a sum of region-specific trend components $\boldsymbol{\mu}_{i,t}$, the regression effects due to lprice, lndi, and lpimin, and the observation noise $\boldsymbol{\epsilon}_{i,t}$. Different variations of this model are obtained by considering different models for the trend component $\boldsymbol{\mu}_{i,t}$. Proper modeling of the trend component is important because it captures differences between the regions because of unrecorded factors such as demographic changes over time, results of anti-smoking campaigns, and so on. The following statements specify and fit one such model:

```
proc ssm data=Cigar plots=residual;
   id year interval=year;
   array RegionArray{46} region1-region46;
   do i=1 to 46;
      RegionArray[i] = (region=i);
   end;
   trend IrwTrend(ll) cross(matchparm)=(RegionArray) levelvar=0;
   irregular wn;
   model lsales = lprice lndi lpimin IrwTrend wn;
   eval TrendPlusReg = IrwTrend + lprice + lndi + lpimin;
   output out=forCigar pdv press;
run;
```

The PROC SSM statement specifies the input data set, Cigar, which contains analysis variables such as the response variable, lsales, and the predictor variables, lprice, lndi, and lpimin. The PLOTS=RESIDUAL option in the PROC SSM statement produces residual diagnostic plots. The optional ID statement specifies a numeric index variable (often a SAS date or datetime variable), which is year in this case. The INTERVAL=YEAR option in the ID statement indicates that the measurements are collected on a yearly basis. The next few statements define a 46-dimensional array of dummy variables, RegionArray, such that RegionArray[i] is 1 if region is i and is 0 otherwise. The next three statements, TREND, IRREGULAR, and MODEL, constitute the model specification part of the program:

- **`trend IrwTrend(ll) cross(matchparm)=(RegionArray) levelvar=0;`** defines a trend, named IrwTrend, of local linear type (which is signified by the keyword *ll* used within the parenthesis after the name). A local linear trend—a trend with time-varying level and time-varying slope—depends on two parameters: the disturbance variance of the level equation and the disturbance variance of the slope equation (see the section "Local Linear Trend" on page 2442 for more information). The LEVELVAR=0 specification fixes the disturbance variance of the level equation to 0, which results in a trend model called an *integrated random walk* (IRW). An IRW model tends to produce a smoother trend than a general local linear trend. In the limiting case, if the disturbance variance of the slope equation is also 0, the IRW trend reduces to a straight line (with a fixed intercept and slope). In addition, because of the use of the 46-dimensional array, RegionArray, in the CROSS= option (**`cross(matchparm)=(RegionArray)`**), this trend specification amounts to fitting a separate IRW trend for each region. This is because, as a result of the CROSS= option, IrwTrend is treated as a linear combination of 46 (the number of variables in RegionArray) stochastically independent, integrated random walks,

  $$\text{IrwTrend}_t = \sum_{i=1}^{46} \text{RegionArray}[i]\, \boldsymbol{\mu}_{i,t}$$

  where each $\boldsymbol{\mu}_{i,t}$ is an integrated random walk. Note that since RegionArray[i] is a binary variable, IrwTrend equals $\boldsymbol{\mu}_{i,t}$ when region is i. Lastly, the use of MATCHPARM option specifies that the different IRW trends $\boldsymbol{\mu}_{i,t}$ use the same disturbance variance parameter for their slope equation. This is done mainly for parsimony. Based on the model diagnostics shown later, this appears to be a reasonable model simplification.

- **`irregular wn;`** defines the observation noise $\boldsymbol{\epsilon}_{i,t}$, named wn, as a sequence of independent, identically distributed, zero-mean, Gaussian variables—a white noise sequence.

- **model lsales = lprice lndi lpimin IrwTrend wn;** defines the model for lsales as a sum of regression effects that involve lprice, lndi, and lpimin, a trend term, lrwTrend, and the observation noise wn.

The last two statements, EVAL and OUTPUT, control certain aspects of the procedure output. The following EVAL statement defines a linear combination, named TrendPlusReg, of selected terms in the MODEL statement:

```
eval TrendPlusReg = IrwTrend + lprice + lndi + lpimin;
```

This EVAL statement causes the SSM procedure to produce an estimate of TrendPlusReg (and its standard error), which can then be printed or output to a data set. TrendPlusReg contains all the terms in the model except for the observation noise and thus can be regarded as the *explanatory* part of the model. In the OUTPUT statement, you can specify an output data set that stores all the component estimates that are produced by the procedure. The following OUTPUT statement specifies forCigar as the output data set:

```
output out=forCigar pdv press;
```

The PDV option causes variables such as region1–region46, which are defined by the DATA step statements within the SSM procedure, also to be included in the output data set. The PRESS option causes the printing of fit measures that are based on the delete-one cross validation errors (see the section "Delete-One Cross Validation and Structural Breaks" on page 2439 for more information).

All the models that are specified in the SSM procedure possess a state space representation. For more information, see the section "State Space Model and Notation" on page 2426. The SSM procedure output begins with a table (not shown here) of the input data set that provides the name and other information. Next, the "Model Summary" table, shown in Figure 33.1, provides basic model information, such as the following:

- the dimension of the underlying state equation, 92 (because each of the 46 IRW trends $\boldsymbol{\mu}_{i,t}$ contributes two elements to the state)
- the diffuse dimension of the model, 95 (which is equal to the three regressors plus the 92 diffuse initial states of $\boldsymbol{\mu}_{i,t}$)
- the number of model parameters, 2 (which is the common disturbance variance of the slope equation in lrwTrend and the variance of the noise term wn)

This information is very useful in determining the computational complexity of the model (the larger state size, 92, explains the relatively long computing time—as much as two minutes on some desktops—for this example).

Figure 33.1 Summary of the Underlying State Space Model

The SSM Procedure

Model Summary	
Model Property	**Value**
Number of Model Equations	1
State Dimension	92
Dimension of the Diffuse Initial Condition	95
Number of Parameters	2

The index variable information is shown in Figure 33.2. Among other things, it categorizes the data to be of the type *Regular with Replication*, which implies that the data are regularly spaced with respect to the ID variable and at least some observations have the same ID value. This is clearly true in this example: the data are yearly without any gaps, and there are 46 observations in each year—one per region.

Figure 33.2 Index Variable Information

ID Variable Information

Name	Start	End	Max Delta	NDistinct	Type
year	1963	1992	1	30	Regular with Replication

Figure 33.3 provides simple summary information about the response variable. It shows that lsales has no missing values and no induced missing values because the predictors in the model, lprice, lndi, and lpimin, do not have any missing values either.

Figure 33.3 Response Variable Summary

Response Variable Information

Name	Number of Observations Total	Missing	Induced Missing	Minimum	Maximum	Mean	Std Deviation
lsales	1380	0	0	3.98	5.7	4.79	0.225

The regression coefficients of lprice, lndi, and lpimin are shown in Figure 33.4. As expected, the coefficient of lprice is negative and the coefficients of lndi and lpimin are positive, all being statistically significant. This is consistent with the expectation that the cigarette sales are adversely affected by the price and are positively correlated with the disposable income. The estimated effect of lpimin, called the bootlegging effect by Baltagi and Levin (1992), is statistically significant but smaller than the effects of lprice and lndi.

Figure 33.4 Estimated Regression Coefficients

Regression Parameter Estimates

Response Variable	Regression Variable	Estimate	Standard Error	t Value	Pr > |t|
lsales	lprice	-0.3480	0.0232	-15.01	<.0001
lsales	lndi	0.1425	0.0344	4.15	<.0001
lsales	lpimin	0.0619	0.0269	2.30	0.0214

Figure 33.5 Estimated Model Parameters

Model Parameter Estimates

Component	Type	Parameter	Estimate	Standard Error	t Value
lrwTrend	LL Trend	Slope Variance	0.000169	0.0000219	7.72
wn	Irregular	Variance	0.000592	0.0000342	17.29

Figure 33.5 shows the estimates of the disturbance variance of the slope equation in lrwTrend and the variance of the noise term wn.

Figure 33.6 shows a panel of residual normality diagnostic plots. These plots show that the residuals are symmetrically distributed but contain slightly larger than expected number of extreme residuals. Figure 33.7 shows the plot of residuals versus time. There the residuals do not exhibit any obvious pattern; however, the plot does show that more extreme residuals appear before 1970 and after 1989. On the whole, however, these plots do not exhibit serious violations of model assumptions.

Figure 33.6 Residual Normality Check

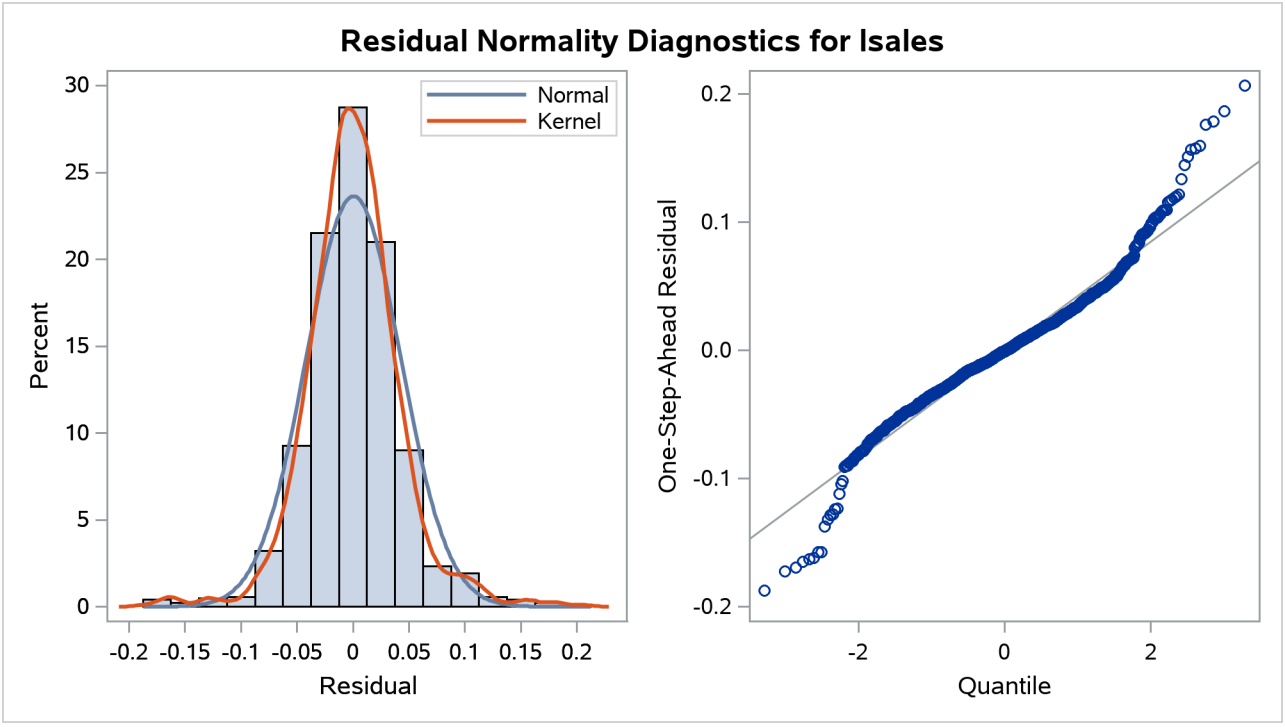


Figure 33.7 Standardized Residuals Plotted against Time

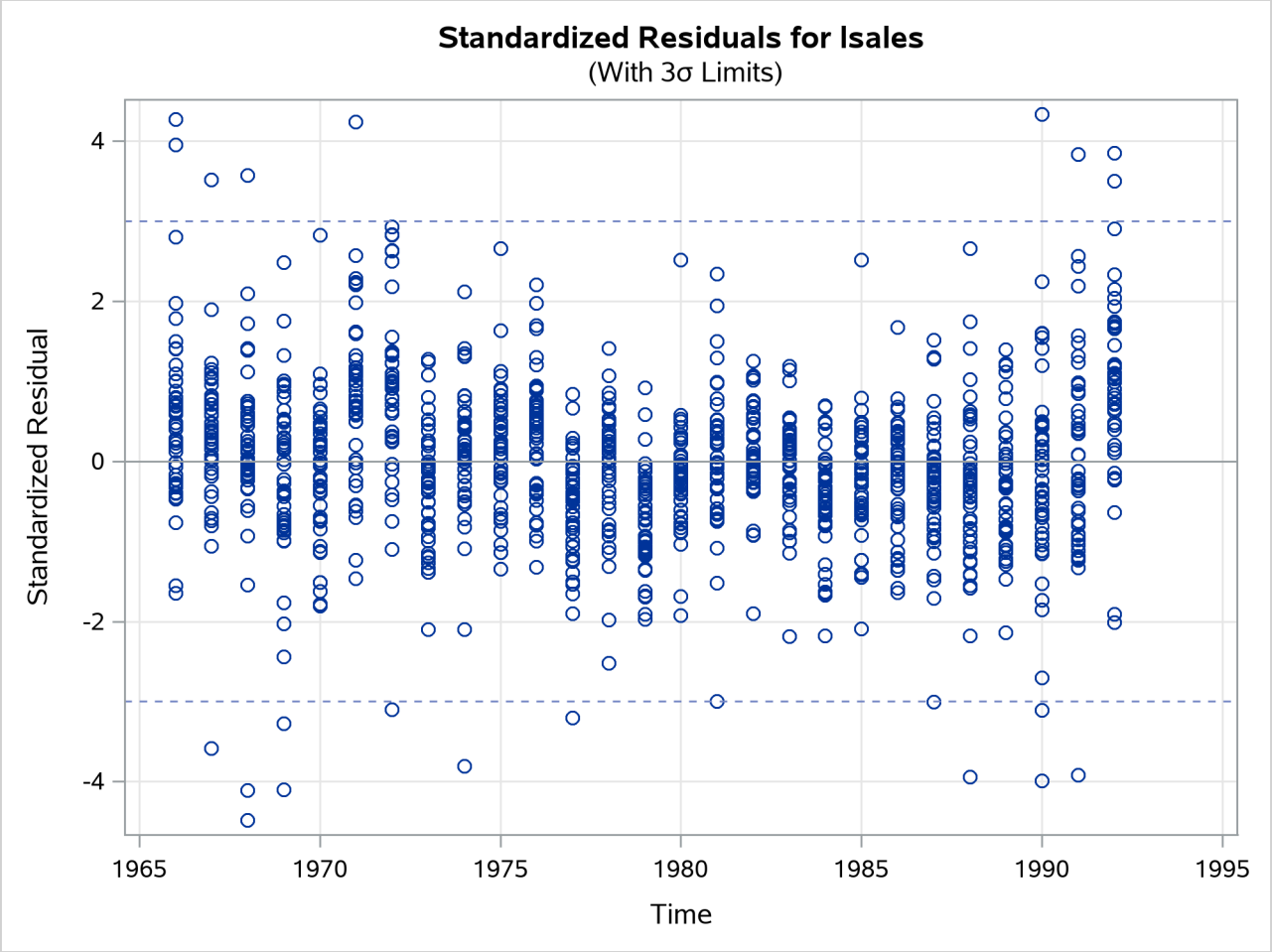


Figure 33.8 shows the details of the likelihood computations such as the number of nonmissing response values used and the likelihood of the fitted model. For more information, see the section "Likelihood Computation and Model-Fitting Phase" on page 2435. Figure 33.8 shows the likelihood-based information criteria in lower-is-better format, which are useful for model comparison.

Figure 33.8 Likelihood Computation Details

Likelihood Computation Summary	
Statistic	**Value**
Nonmissing Response Values Used	1380
Estimated Parameters	2
Initialized Diffuse State Elements	95
Normalized Residual Sum of Squares	1285.0002
Diffuse Log Likelihood	2246.0466
Profile Log Likelihood	2169.6232

Figure 33.9 Information Criteria

Information Criteria		
Statistic	Diffuse Likelihood Based	Profile Likelihood Based
AIC (lower is better)	-4488.093	-4145.246
BIC (lower is better)	-4477.776	-3637.952
AICC (lower is better)	-4488.084	-4130.417
HQIC (lower is better)	-4484.220	-3955.472
CAIC (lower is better)	-4475.776	-3540.952

In addition to the regression estimates, it is useful to analyze the estimates of different model components such as the trend component lrwTrend and the linear combination TrendPlusReg. These estimates can be printed by using the PRINT= option provided in the TREND and EVAL statements, or they can be output to a data set (as it is done in this illustration). This latter option is particularly useful for graphical exploration of these components by standard graphical procedures such as SGPLOT and SGPANEL procedures. The following statements produce a panel of plots that shows how well the proposed model fits the observed cigarette sales in the first three regions, which correspond to Alabama, Arizona, and Arkansas. The output data set, forCigar, contains all the needed information: Smoothed_TrendPlusReg contains the smoothed (full-sample) estimate of TrendPlusReg, and Smoothed_Lower_TrendPlusReg and Smoothed_Upper_TrendPlusReg contain its 95% lower and upper confidence limits. In addition, for easy readability, a user-defined format (RegionFormat), which is created by using the FORMAT procedure (not shown), is used to associate the region names to region values.

```
proc sgpanel data=forCigar noautolegend;
   where region <= 3;
   format region RegionFormat.;
   title 'Region-Specific Sales Patterns with 95% Confidence Band';
   panelby region / columns=3;
   band x=year lower=Smoothed_Lower_TrendPlusReg
   upper=Smoothed_Upper_TrendPlusReg;
   scatter x=year y=lsales;
   series x=year y= Smoothed_TrendPlusReg;
run;
```

Figure 33.10 Cigarette Sales Patterns for the First Three Regions

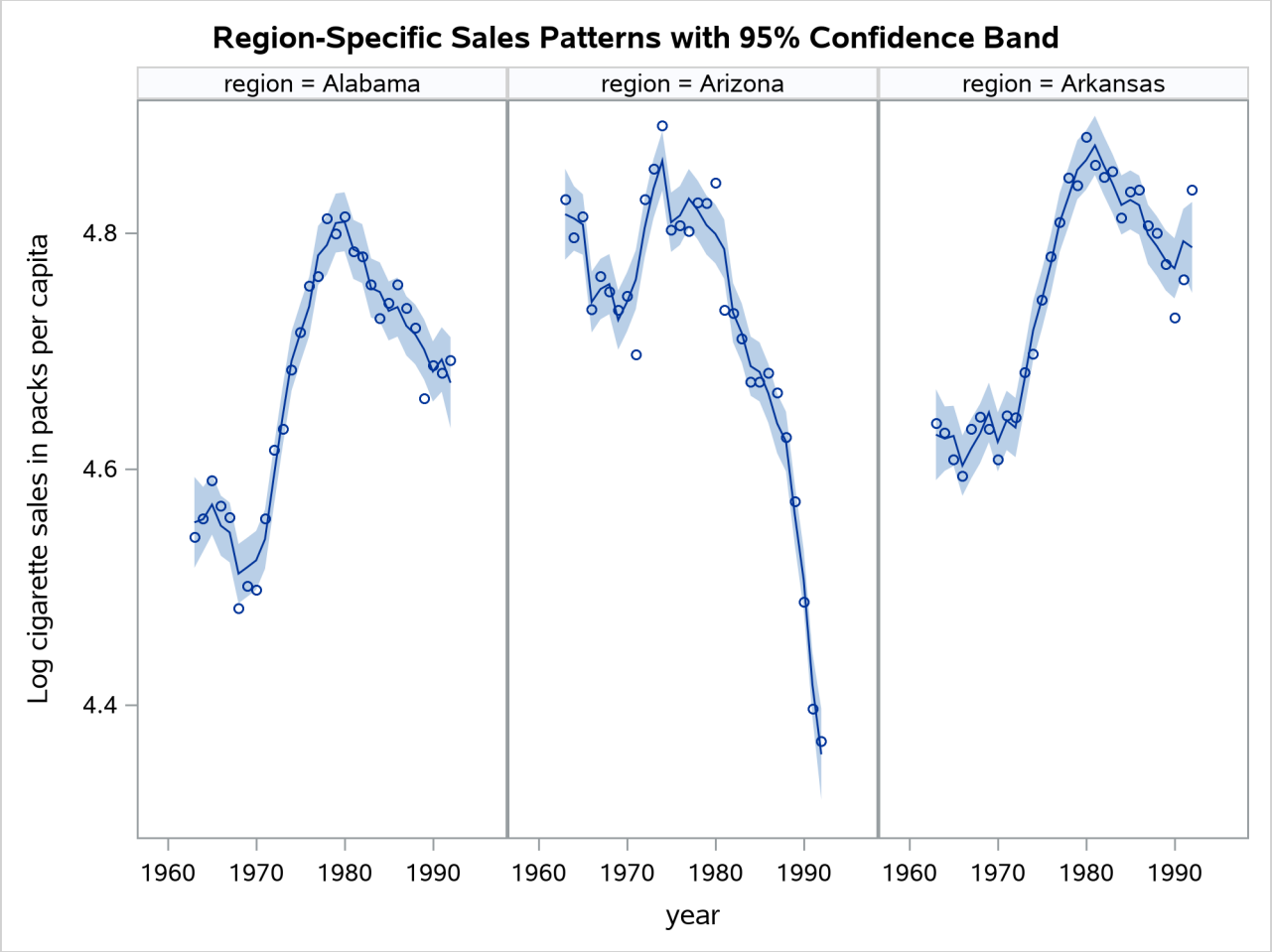
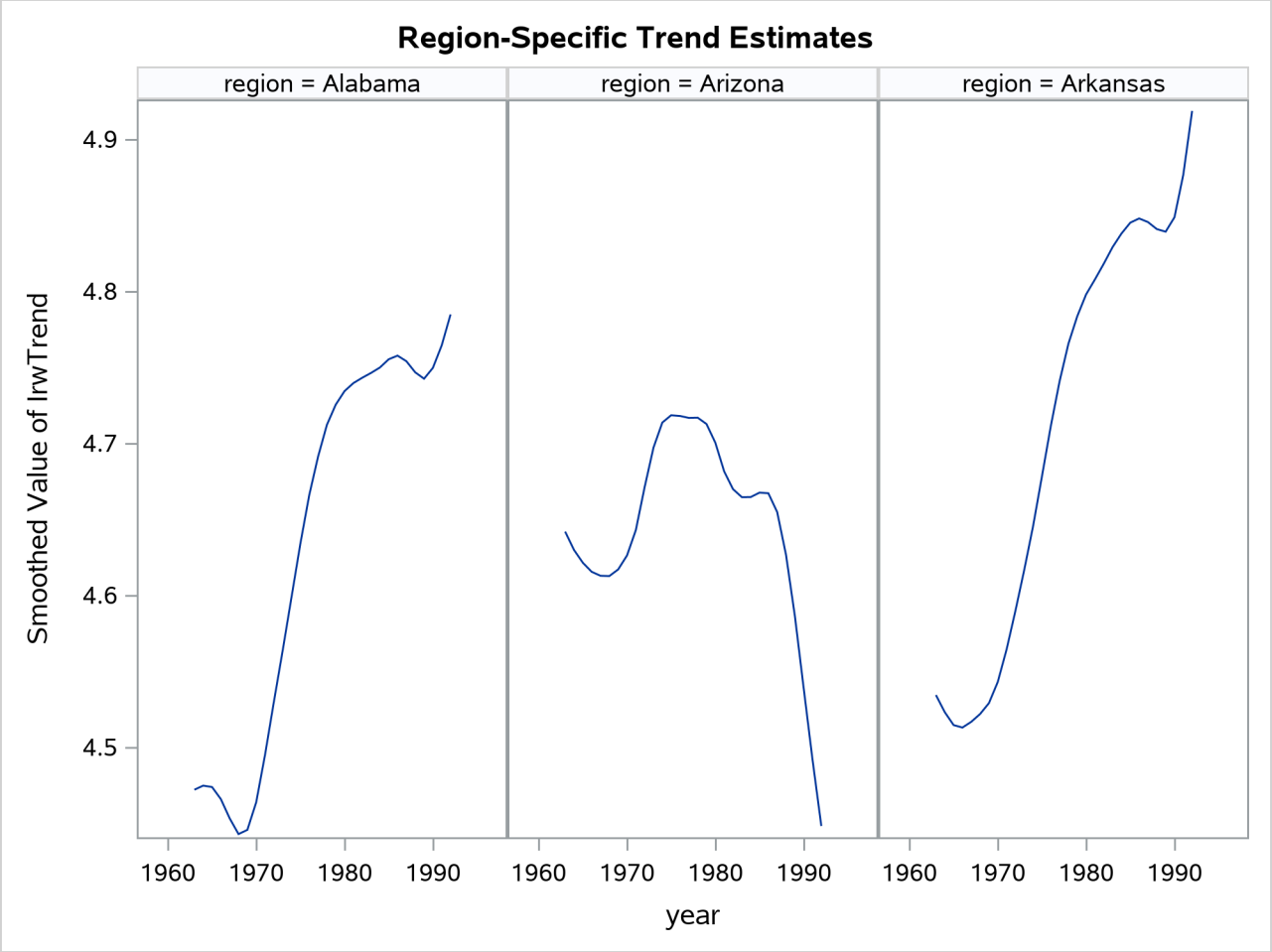


Figure 33.10 seems to indicate that the model fits the data reasonably well. It also shows that Arizona differs markedly from Alabama and Arkansas in its cigarette sales pattern over the years. The following statements produce a similar panel of plots that show the estimate of trend without the regression effects:

```
proc sgpanel data=forCigar noautolegend;
   where region <= 3;
   format region RegionFormat.;
   title 'Region-Specific Trend Estimates';
   panelby region / columns=3;
   series x=year y= smoothed_IrwTrend;
run;
```

Figure 33.11 Estimate of IrwTrend for the First Three Regions



The trend patterns, shown in Figure 33.11, seem to suggest that after accounting for the regression effects, per capita cigarette sales were on the rise in Alabama and Arkansas while they were declining in Arizona.

Syntax: SSM Procedure

The following statements are available in the SSM procedure:

PROC SSM < *options* > **;**
BY *variables* **;**
COMPONENT *name* **=** (*variables*)* *state* < / *options* > **;**
DEPLAG *name*(*response-variable*) *lag-term1* < *lag-term2 . . .* > **;**
EVAL *name* **=** *expression* < / *options* > **;**
ID *variable* < *option* > **;**
IRREGULAR *name* < *options* > **;**
MODEL *response* **=** *variables* < / *options* > **;**
OUTPUT < *options* > **;**
PARMS *variables* < / *options* > **;**
Programming statements **;**
STATE *name*(*dim*)< *options* > **;**
TREND *name*(*type*)< *options* > **;**

You can specify all statements except the BY, ID, and the OUTPUT statements multiple times. The PROC SSM statement and at least one MODEL statement are required. In addition to these statements, you can use most DATA step programming statements to define new variables that are needed for specifying different parts of the state space model.

NOTE: In the statement options described throughout this section, whenever you use a list to specify the elements of the system matrices, the list elements must all be of the same type: either all of them must be variables or all of them must be numbers. In addition, if the list contains more than one variable, then they cannot be of the array type. These are not serious restrictions. When the list contains mix of variables and numbers, you can redefine the numbers as constant variables. Similarly, you can reformulate a list that contains a mix of variables of array and non-array types as just one array by combining all its elements in a new array.

Functional Summary

Table 33.1 summarizes the statements and options that control the SSM procedure. Most commonly needed scenarios are listed; for more information, see the individual statements.

Table 33.1 Functional Summary

Description	Statement	Option
Data Set Options		
Specifies the input data set	PROC SSM	DATA=
Writes series and component forecasts to an output data set	OUTPUT	OUT=
Model Specification Options		
Specifies the index variable	ID	
Defines variables as model parameters	PARMS	
Specifies a response variable and the associated observation equation	MODEL	
Specifies a state subsection	STATE	

Table 33.1 *continued*

Description	Statement	Option
Specifies the transition matrix of a state subsection	STATE	T
Specifies the disturbance covariance matrix of a state subsection	STATE	COV
Specifies the size of the diffuse initial condition of a state subsection	STATE	A1
Specifies the initial covariance matrix of a state subsection	STATE	COV1
Specifies a state subsection for a predefined structural model	STATE	TYPE=
Specifies the regressors in a state equation	STATE	W
Specifies the input vector in a state equation	STATE	SINPUT=
Specifies a component	COMPONENT	
Specifies a predefined trend component	TREND	
Likelihood Optimization Process Control Options		
Specifies the optimization technique	PROC SSM	OPTIMIZER(TECH=)
Limits the number of iterations	PROC SSM	OPTIMIZER(MAXITER=)
Outlier and Structural Break Detection Options		
Turns on the search for additive outliers (AO)		Default
Turns on the search for structural breaks in a state subsection	STATE	CHECKBREAK
Turns on the search for structural breaks in a state subsection associated with a trend	TREND	CHECKBREAK
Specifies the significance level for additive outlier tests	OUTPUT	AO(ALPHA=)
Limits the reported number of additive outliers	OUTPUT	AO(MAXNUM=)
Limits the reported number of additive outliers to a percentage of the series length	OUTPUT	AO(MAXPCT=)
Specifies the significance level for structural break tests	OUTPUT	BREAK(ALPHA=)
Limits the reported number of structural breaks	OUTPUT	BREAK(MAXNUM=)
Limits the reported number of structural breaks to a percentage of the series length	OUTPUT	BREAK(MAXPCT=)
Turns on the search for maximal state shock	OUTPUT	MAXSHOCK
Graphical Residual and Outlier Analysis Options		
Creates a panel of plots that consists of residual normality plots	PROC SSM	PLOTS=RESIDUAL(NORMAL)
Creates the standardized residual plot against time	PROC SSM	PLOTS=RESIDUAL(STD)
Creates a panel of plots that consists of prediction error normality plots	PROC SSM	PLOTS=AO(NORMAL)

Table 33.1 *continued*

Description	Statement	Option
Creates the standardized prediction error plot against time	PROC SSM	PLOTS=AO(STD)
Creates the plot of maximal state shock chi-square statistics against time	PROC SSM	PLOTS=MAXSHOCK
Output Control Options		
Specifies the significance level of the forecast confidence limits	OUTPUT	ALPHA=
Prints the prediction error sum of squares table	OUTPUT	PRESS
Specifies a linear combination of components to be output	EVAL	
Global Printing and Plotting Options		
Turns off all printing for the procedure	PROC SSM	NOPRINT
Turns on all printing options for the procedure	PROC SSM	PRINTALL
Turns off all plotting for the procedure	PROC SSM	PLOTS=NONE
Turns on all plotting options for the procedure	PROC SSM	PLOTS=ALL
Printing State Equation System Matrix Options		
Prints the transition matrix that is associated with a state subsection	STATE	PRINT=T
Prints the disturbance covariance matrix that is associated with a state subsection	STATE	PRINT=COV
Prints the initial covariance matrix that is associated with a state subsection	STATE	PRINT=COV1
Prints the autoregressive coefficient matrix that is associated with a state subsection	STATE	PRINT=AR
Prints the moving average coefficient matrix that is associated with a state subsection	STATE	PRINT=MA
Printing Component, Series Forecast, and Smoothed Estimate Options		
Prints the series forecasts	MODEL	PRINT=FILTER
Prints the full-sample estimates of missing series values	MODEL	PRINT=SMOOTH
Prints the smoothed trend estimate	TREND	PRINT=SMOOTH
Prints the filtered trend estimate	TREND	PRINT=FILTER
Prints the smoothed component estimate	COMPONENT	PRINT=SMOOTH
Prints the filtered component estimate	COMPONENT	PRINT=FILTER
Prints the smoothed component estimate	EVAL	PRINT=SMOOTH
Prints the filtered component estimate	EVAL	PRINT=FILTER
BY Groups		
Specifies BY-group processing	BY	

PROC SSM Statement

PROC SSM < *options* > **;**

The PROC SSM statement is required. You can specify the following *options* in the PROC SSM statement:

BREAKPEAKS
prints an alternate form of the break summary tables when the CHECKBREAK option is used in the STATE or TREND statement or when the MAXSHOCK option is used in the OUTPUT statement. In this alternate form, the summary tables report the significant peaks of the shock statistics curves; see Example 33.8 for examples of these curves.

DATA=*SAS-data-set*
specifies the name of the SAS data set that contains the variables needed for the analysis. If you do not specify this option, PROC SSM uses the most recently created SAS data set.

LIKE=DIFFUSE | MARGINAL (Experimental)
specifies the type of likelihood to use for parameter estimation. You can specify the following values:

DIFFUSE	specifies diffuse likelihood.
MARGINAL	specifies marginal likelihood.

By default, LIKE=DIFFUSE. For more information about different likelihood types, see the section "Likelihood Computation and Model-Fitting Phase" on page 2435.

NOPRINT
turns off all the printing and plotting for the procedure. Any subsequent print options are ignored.

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

PLOTS< **(***global-plot-options***)** > **= (** *plot-request* < *(options)* > < **...** *plot-request* < *(options)* > > **)**
controls the plots produced with ODS Graphics. When you specify only one *plot-request*, you can omit the parentheses around it. Here are some examples:

```
plots=none
plots=all
plots=residual
plots=residual(normal)
plots=(maxshock residual(normal))
plots(unpack)=residual
```

If you do not specify any specific *plot-request*, then by default PROC SSM produces the plot of standardized residuals against time. For general information about ODS Graphics, see Chapter 21, "Statistical Graphics Using ODS" (*SAS/STAT User's Guide*).

Global Plot Options
The *global-plot-options* apply to all relevant plots generated by the SSM procedure. The following *global-plot-option* is supported:

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 *plot-requests* and their *options*:

ALL
produces all plots appropriate for the particular analysis.

AO< (*prediction-error-plot-options***) >**
produces the prediction error plots—one for each response variable. You can specify the following *prediction-error-plot-options*:

NORMAL
produces a summary panel of the prediction error diagnostics, which consist of the following:

- histogram of prediction errors
- normal quantile plot of prediction errors

STD
produces a scatter plot of standardized prediction errors against time.

MAXSHOCK
produces a scatter plot of maximal state shock statistics against time.

NONE
suppresses all plots.

RESIDUAL < (*residual-plot-options***) >**
produces the residuals plots—one for each response variable. You can specify the following *residual-plot-options*:

NORMAL
produces a summary panel of the residual diagnostics, which consist of the following:

- histogram of residuals
- normal quantile plot of residuals

STD
produces a scatter plot of standardized residuals against time.

For more information about the precise meaning of the terms *maximal state shock statistics* and *prediction errors*, see the section "Delete-One Cross Validation and Structural Breaks" on page 2439.

PRINTALL
turns on all the printing options for the procedure. All subsequent NOPRINT options in the procedure are ignored.

STATEINFO
prints two tables that provide information about the composition of the state vector in terms of the components specified in the model. One table describes the composition of state $\boldsymbol{\alpha}_t$, and the other table describes the diffuse vector $\boldsymbol{\delta}$ and the regressors, which are part of the initial condition specification $\boldsymbol{\alpha}_1$. For more information about the state space model notation, see the section "State Space Model and Notation" on page 2426.

OPTIMIZER(<TECHNIQUE=*technique***> <MAXITER=***integer***>)**
specifies options that are associated with the optimizer used in the maximum likelihood parameter estimation. The default settings of the optimization process are adequate in most problems. However, in some cases it might be useful to change the optimization technique or to change the maximum number of iterations. You can specify one of the following *techniques*:

ACTIVESET	corresponds to the active-set method.
DBLDOG	corresponds to the double-dogleg method.
INTERIORPOINT	corresponds to the primal-dual interior point method.
NEWRAP	corresponds to the Newton-Raphson method.
QUANEW	corresponds to the (dual) quasi-Newton method.
TRUREG	corresponds to the trust region method.

The default technique is TRUREG. The INTERIORPOINT and ACTIVESET techniques are documented in Chapter 11, "The Nonlinear Programming Solver" (*SAS/OR User's Guide: Mathematical Programming*), and the remaining techniques are documented in Chapter 6, "Nonlinear Optimization Methods." You can alter the maximum number of iterations setting in the nonlinear optimization search by specifying a nonnegative *integer* as the MAXITER= value.

ZSPARSE
enables the exploitation of the sparsity of the $\mathbf{Z}_t$ matrices in the observation equation during the modeling calculations (see the section "State Space Model and Notation" on page 2426 for further information). The use of this option can improve the computational efficiency of models that have a large state dimension and sparse $\mathbf{Z}_t$ matrices—that is, many of their elements are zero. You should use the ZSPARSE option only when the state dimension is sufficiently large (at least 30) and a good percentage (at least 50%) of $\mathbf{Z}_t$ entries are zero; otherwise, the computational efficiency can in fact degrade. For example, the illustration that is discussed in the section "Getting Started: SSM Procedure" on page 2394 is a good candidate for the use of the ZSPARSE option:

```
proc ssm data=Cigar plots=residual zsparse;
```

BY Statement

BY *variables* **;**

A BY statement can be used in the SSM procedure to process a data set in groups of observations that are defined by the BY variables. The model specified by using the MODEL and other statements is applied to all the groups defined by the BY variables. When a BY statement appears, the procedure expects the input data

set to be sorted in order of the BY variables. The BY variables are one or more variables in the input data set. The BY variables cannot be used in the model specification; in particular, they cannot be used as response variables or regressors in a MODEL statement.

COMPONENT Statement

COMPONENT *name* **=** (*var1 var2 . . . | number1 number2 . . .*) * *state* < / *options* > **;**

COMPONENT *name* **=** *state*[*integer*] < / *options* > **;**

COMPONENT *name* **=** (*Variable | Number*)* *state*[*integer*] < / *options* > **;**

The COMPONENT statement specifies a component (a linear combination of state elements), named *name*. You can use *name* later as a term in the right-hand side of the MODEL statement, which defines the observation equation. The estimate of *name* is output to the OUT= data set that is specified in the OUTPUT statement. In addition, you can print the component estimate by using the PRINT= option.

The first form of the COMPONENT statement defines a component as a dot product of a state subsection *state* and a row vector *(var1 var2 . . .)*. The value of *state* can be the name of a state subsection that is defined by using a STATE statement elsewhere in the program, or it can be the name of the state that is associated with a trend component defined by using a TREND statement elsewhere in the program (see the section "TREND Statement" on page 2421 for more information about the naming of the state that is associated with a trend component). The row vector *(var1 var2 . . .)*, which can be either a list of numbers or a list of variables, must be of the same dimension as the actual dimension of the state subsection. The dot product form—also called the explicit dot product form—of the component specification is unambiguous; however, it requires detailed knowledge of the state vector underlying the *state* specification. Suppose that mystate is a two-dimensional state defined by a STATE statement elsewhere in the program and that X1 and X2 are (numeric) predictor variables. The following are valid examples of the dot product form of the COMPONENT statement:

```
component c1 = (x1 x2) * mystate;
component c2 = (1 1) * mystate;
```

The second and the third forms of the COMPONENT statement are a shortened version of the first form. The second form defines the component as a particular element of state—for example, state[3] defines the component as the third element of *state*. The specified *integer* must lie between 1 and *dim*, the nominal dimension of state. The second form of component specification has another important use when the STATE statement that defines *state* uses the TYPE= option to set its type or when *state* is associated with a trend component. In these cases, the second form of the component specification assumes additional meaning when the nominal state dimension and the actual state dimensions differ (specifically the state types LL, SEASON, CYCLE, and VARMA and the states associated with all the trend types). For example, if *state* is a three-dimensional seasonal component, state[2] signifies an appropriate linear combination of *state* that results in the second of the three seasonals that constitute the three-dimensional seasonal. Similar interpretation holds for the CYCLE type. For more information, see the sections "Multivariate Season" on page 2448 and "Predefined Structural Models" on page 2445. The third form extends the second form by permitting multiplication by a variable or a number.

NOTE: A component that is based on a state associated with a trend component cannot be used as a right-hand side term in any MODEL statement. That is, it is defined purely for output purposes (either printed or output to a data set). However, it can be used as a term in the expression that is specified in an EVAL statement to build more complex linear combinations for output.

You can specify the following *options* to print the filtered or smoothed estimate of the component:

PRINT=FILTER | SMOOTH

PRINT=(< FILTER > < SMOOTH >)

requests printing of the filtered or smoothed estimate of the specified component.

DEPLAG Statement

DEPLAG *name*(*response-variable*) *lag-term1 < lag-term2 . . . >* **;**

The DEPLAG statement defines a term, named *name*, that consists of a linear combination of lagged response variables. You can use *name* later as a right-hand-side term in the MODEL statement for the response variable, as specified in *name(response-variable)*. For a multivariate model, a separate DEPLAG statement is needed for each MODEL statement that has a right-hand-side term that involves lagged response variables. The linear combination of lagged response variables is specified by using one or more *lag-terms*. Each *lag-term* specifies the lags that are associated with one of the response variables.

A *lag-term* is specified in one of the following forms:

lag-response-variable(**LAGS=***maximum-lag*)

lag-response-variable(**LAGS=(***integer1 integer2 . . .*)**)**

lag-response-variable(**LAGS=***maximum-lag* **COEFF=(***number1 number2 . . .*) | (*variable1 variable2 . . .*)**)**

lag-response-variable(**LAGS=(***integer1 integer2 . . .*) **COEFF=(***number1 number2 . . .*) | (*variable1 variable2 . . .*)**)**

The *lag-response-variable* in the lag term specification can be the same as the response variable that corresponds to the model equation (which is specified in *name(response-variable)*), or it can be a different response variable.

The first form of specification is useful when all lags up to the *maximum-lag*, which must be a positive integer, are present in the lag term. The second form is useful when only certain lags, which are specified as a list of positive integers in parentheses, are present. In these two cases, the lag coefficients are not specified and they are treated as unknown parameters to be estimated from the data.

The COEFF= option in the last two forms enable you to specify lag coefficients. The COEFF= option must follow the LAGS= option. You can use the COEFF=(*number1 number2 . . .*) option to specify the lag coefficients as known values. Similarly, you can use the COEFF=(*variable1 variable2 . . .*) option to specify user-defined variables as lag coefficients; the user-defined variables can be functions of parameters (which are defined by using the PARMS statement) and input variables. However, the lag coefficients cannot depend on any of the response variables. The number of coefficients specified in the COEFF= option must exactly equal the number of lags specified in the LAGS= option.

There can be at most one DEPLAG statement associated with a particular MODEL statement (you can specify all the needed lag terms in a single DEPLAG statement).

As an illustration, let lagsFORy1 and lagsFORy2 represent the following linear combinations of lagged response variables Y1, Y2, and Y3:

$$\text{lagsFORy1} = \theta_{11} Y1_{t-1} + \theta_{12} Y1_{t-2} + \theta_{22} Y2_{t-2} + \theta_{23} Y2_{t-3} + 1.2 Y3_{t-1} - 2.1 Y3_{t-2}$$

$$\text{lagsFORy2} = \text{Phi1}Y1_{t-1} + \text{Phi2}Y1_{t-2} + \theta_{21}Y2_{t-1}$$

where Phi1 and Phi2 denote user-defined variables and θ_{ij} denote generic parameters. You can specify lagsFORy1 (which is used in the model equation for Y1) and lagsFORy2 (which is used in the model equation for Y2) as follows:

```
deplag lagsFORy1(y1) y1(lags=2) y2(lags=(2 3)) y3(lags=2 coeff=(1.2 -2.1));
deplag lagsFORy2(y2) y1(lags=2 coeff=(phi1 phi2)) y2(lags=1);
... more statements ...;
model y1 = lagsFORy1 ...;
model y2 = lagsFORy2 ...;
model y3 = ...;
... more statements ...;
```

assuming that the right-hand side of the MODEL equation for Y3 does not have a term that involves lags of response variables.

The DEPLAG statement in PROC SSM has the same purpose as the DEPLAG statement in PROC UCM (see Chapter 41, "The UCM Procedure"). However, there are many differences in the syntax of the two statements, mainly because PROC SSM supports much more complex models. The syntax difference between the two DEPLAG statements can be illustrated by considering the differencing specification—$(1 - B)(1 - B^{12}) = (1 - B - B^{12} + B^{13})$—in the well-known airline model (ARIMA(0, 1, 1)(0, 1, 1)12 model). You can specify the lag-term that is implied by the differencing in the airline model in PROC UCM as follows:

```
deplag lags=(1)(12) phi=1 1 noest;
```

In PROC SSM the same specification has the following form:

```
deplag airLags(y) y(lags=(1 12 13) coeff=(1 1 -1));
```

Both these specifications define the same lag-term: $(y_{t-1} + y_{t-12} - y_{t-13})$.

For an example of the use of lagged response variables in a model specification, see Example 33.13. For more information about models that have dependent lags, see the section "Models with Dependent Lags" on page 2451.

NOTE: Models that have lagged response variables are permitted only if the data form a time series (either univariate or multivariate). The SSM procedure adds one more restriction on the models that use lagged response variables: the variables in the list that define a component in any of the COMPONENT statements must be free of unknown parameters. This restriction is artificial and is made primarily to reduce the overall complexity of the model. In future versions of the SSM procedure, this restriction might go away.

EVAL Statement

EVAL *name* **=** *number1*variable1 + number2*variable2 + . . .* **<** */ options* **>** **;**

The EVAL statement defines a linear combination, named *name*, of the terms used in the right-hand side of a MODEL statement. You can specify any variables (for example, predictor variables and names of components) in the expression of the EVAL statement; however, you cannot specify in this expression any observation disturbances that are specified by the IRREGULAR statement and any model terms that are specified by the DEPLAG statement. Suppose C1 and C2 are two components (defined by COMPONENT

statements elsewhere in the program), T1 is a trend component, and X1 is a regression variable used in a model. The following are valid examples of the EVAL statement:

```
eval e1 = c1 - c2;
eval e2 = t1 + c1 + x1;
eval e2 = t1 + 2*c1 - 1.5*x1;
```

The estimates of linear combinations defined by the EVAL statement (for example, E1, E2, and E3) are output to the OUT= data set that is specified in the OUTPUT statement.

The components used in a given EVAL expression must correspond to distinct state subsections. This requirement is imposed only to simplify the overall readability of the program and does not limit the type of linear combinations that can be specified; if two components in the right hand side of an EVAL expression share the same state subsection, a new component that combines the effect of these two components can always be defined.

In addition, you can print these estimates by using the following PRINT= options:

PRINT=FILTER | SMOOTH

PRINT=(< FILTER > < SMOOTH >)

requests printing of the filtered or smoothed estimate of the specified linear combination.

NOTE: The expression builder in the EVAL statement is primitive. For example, you cannot use parentheses to group terms.

ID Statement

ID *variable* < *option* > **;**

The ID statement names a numeric variable to associate a sequence value—usually related to a time stamp—to the observations in the input data set. The observations within a BY group must be ordered in ascending order by the ID variable. Often the ID variable's values are SAS date, time, or datetime values, and each observation within a BY group has a unique ID value. Generally, however, the ID variable can be any numeric variable, and there can be multiple observations with the same ID value. If the ID values are SAS date, time, or datetime values, you can specify the associated unit of time—for example, day, week or month—by using the INTERVAL= option. If an ID statement is not specified, the observation number, with respect to the BY group, is used as the time ID. Whenever an ID variable is specified, a variable, _ID_DELTA_, is automatically created that can be used as any input data set variable in the programming statements. _ID_DELTA_ contains the distance between two successive ID values. The first _ID_DELTA_ value is arbitrarily taken as one. If the INTERVAL= option is specified, the distance between the ID values is measured in terms of the number of intervals; therefore, for regularly spaced data, _ID_DELTA_ is identically equal to one. You can specify the following *option* in the ID statement:

INTERVAL=*value*

specifies the unit of time interval that is used for measuring the ID values. INTERVAL=*value* is used in conjunction with the ID variable to check that the input data are in the proper order. For a complete discussion of the supported intervals, see Chapter 4, "Date Intervals, Formats, and Functions."

IRREGULAR Statement

IRREGULAR *name* < *options* > **;**

The IRREGULAR statement specifies a one-dimensional white noise component, which can be used to specify the observation error in a MODEL statement. You can specify the following *options* in the IRREGULAR statement:

PRINT=SMOOTH
requests printing of the smoothed estimate of the specified irregular component.

VARIANCE=*variable* | *number*
specifies the variance of the white noise. Any nonnegative value, including 0, is permissible. If the *variable* contains unknown parameters, they are estimated from the data. Similarly, if the VARIANCE= option is not specified, the variance is estimated from the data.

MODEL Statement

MODEL *response* **=** *variables* < / *options* > **;**

A MODEL statement specifies an observation equation that describes a response variable as a sum of regression effects and components that are defined in the program. The response variable must be a numeric variable from the input data set. The variables used in the right-hand side of the model expression can be numeric variables from the input data set, numeric variables defined by using programming statements, or names of components that are specified in the COMPONENT, DEPLAG, TREND, or IRREGULAR statements.

For a multivariate model, a separate MODEL statement is needed for each of the response variables. In this case, the observation errors, which are specified in an IRREGULAR statement, must be different in each MODEL statement.

The components that are specified in a given MODEL statement must correspond to distinct state subsections. This requirement is imposed only to simplify the overall readability of the program and does not limit the type of models that can be specified; if two components on the right-hand side of a MODEL statement share the same state subsection, a new component that combines the effect of these two components can always be defined.

You can specify the following *options* in the MODEL statement; they must be separated from the list of terms in the right-hand side of the model equation by a slash (/):

AGGREGATE(START=*startFlag***)** (Experimental)
SUM(START=*startFlag***)**
produces a table of full-sample predictions of the temporally aggregated values of the response variable that is specified in the MODEL statement. The variable that you specify in the START= option, *startFlag*, must be a zero-one variable that flags the start of an aggregation interval—equal to 1 at the start of an interval and 0 otherwise. For example, you can use this option to obtain the forecasts of weekly (or monthly) totals from a daily series. In this case, the value of *startFlag* is 1 at the start of the week (or month) and 0 otherwise. For more information, see the section "Temporal Aggregation and Temporal Distribution (Experimental)" on page 2452. This option is valid only if the data form a time

series (either univariate or multivariate). If you use the AGGREGATE option in a MODEL statement, you cannot use the DISTRIBUTE option in the same statement or in another MODEL statement.

DISTRIBUTE(START=*startFlag*) (Experimental)
indicates that the response variable that is specified in the MODEL statement is a temporally aggregated version of an unobserved variable. The variable that you specify in the START= option, *startFlag*, must be a zero-one variable that flags the start of an aggregation interval—equal to 1 at the start of an interval and 0 otherwise. This option can be used only when the data form a time series (either univariate or multivariate) and when the overall model specification does not contain terms that involve lagged response variables (that is, the model specification does not involve the use of DEPLAG statements). If you use the DISTRIBUTE option in a MODEL statement, you cannot use the AGGREGATE option in the same statement or in another MODEL statement. For more information, see the section "Temporal Aggregation and Temporal Distribution (Experimental)" on page 2452.

PRINT=FILTER | SMOOTH

PRINT=(< FILTER > < SMOOTH >)
requests printing of the filtered or smoothed estimate of the specified response variable. The filtered estimate is produced during the filtering phase, and the smoothed estimate is produced by the smoothing phase of the Kalman filter and smoother algorithm. The filtered estimate is also called the one-step-ahead forecast of the response variable. The smoothed estimate corresponds to the full-sample prediction of the response variable. Since the full-sample prediction of a nonmissing response value is that value itself, full-sample predictions are printed only for the missing response values.

OUTPUT Statement

OUTPUT < *options* > **;**

The OUTPUT statement creates an optional output data set and also provides options to control certain aspects of the procedure output. If the OUT= option is specified, then an output data set is created to store estimates of the model components and series forecasts. If the OUT= option is omitted, then no data set is created by the OUTPUT statement. Other options in the OUTPUT statement produce additional information in the printed output generated by the procedure. For example, the AO and BREAK options control the search for additive outliers and structural breaks in the data, respectively.

AO(< ALPHA=*number* > < MAXNUM=*number* > < MAXPCT=*number* >)
controls the additive outlier search (see the section "Delete-One Cross Validation and the Additive Outlier Detection" on page 2439 for more information). The ALPHA= suboption specifies the significance level for reporting the outliers. The default is ALPHA=0.05. The MAXNUM= suboption limits the number of outliers to search. The default is MAXNUM=5. The MAXPCT= suboption is similar to the MAXNUM= suboption. In the MAXPCT= option you can limit the number of outliers to search for according to a percentage of the series length. The default is MAXPCT=1. When you specify both of these options, the lesser of the two search numbers is used.

ALPHA=*number*
specifies the significance level of the forecast confidence intervals. For example, ALPHA=0.05, which is the default, results in a 95% confidence interval.

BREAK(< ALPHA=*number* **> < MAXNUM=***number* **> < MAXPCT=***number* **>)**
controls the structural break search (for more information, see the section "Structural Breaks in the State Evolution" on page 2439). In order for this option to have any effect, the CHECKBREAK option in one of the STATE or TREND statements, or the MAXSHOCK option in the OUTPUT statement, must be turned on. The ALPHA= suboption specifies the significance level for reporting the breaks. The default is ALPHA=0.05. The MAXNUM= suboption limits the number of breaks to search. The default is MAXNUM=5. The MAXPCT= suboption is similar to the MAXNUM= suboption. In the MAXPCT= option, you can limit the number of breaks to search for according to a percentage of the number of distinct time points in the data. The default is MAXPCT=1. When you specify both of these options, the lesser of the two search numbers is used.

MAXSHOCK
causes the computation of the maximal state shock chi-square statistic at each distinct time point in the input data set. These statistics are output to the data set that is specified in the OUT= option. A time series plot of these statistics is produced if the PLOTS=MAXSHOCK option is specified in the PROC SSM statement. These statistics are useful for detecting structural breaks in the state evolution process. This option can be computationally expensive for a model with large state size. For more information, see the section "Structural Breaks in the State Evolution" on page 2439.

OUT=*SAS-data-set*
specifies an output data set for the forecasts. The output data set contains the ID variable (if specified), the response variables, the one-step-ahead and out-of-sample response variable forecasts, the forecast confidence intervals, the smoothed values of the response series, and the one-step-ahead and smoothed estimates of the model components—including expressions that are defined by using the EVAL statement. For more information, see the section "OUT= Data Set" on page 2461.

PDV
causes the inclusion of the variables (variables in the program data vector) that are defined by using the programming statements in the SSM procedure in the OUT= data set. The parameters defined by the PARMS statement are also included. The output data set contains the values of these variables evaluated for all the rows in the input data set that is specified in the DATA= option. The parameters in the PARMS statement contain their estimated values.

PRESS
prints the prediction error sum of squares (PRESS) and the generalized cross validation error sum of squares (GCV). The PRESS table also reports the number of summands that are used in these sums of squares. For more information, see the section "Delete-One Cross Validation and the Additive Outlier Detection" on page 2439.

PARMS Statement

PARMS *variable< =number> variable< =number> < / options>* **;**

The PARMS statement declares the parameters of a model and optionally sets their initial values. You can also specify the lower and upper limits of their validity range. The parameters declared by using the PARMS statement are called *named parameters* throughout this chapter. A model can have additional parameters: any unspecified quantity in the model specification becomes part of the parameter vector. You can specify the following *options*:

LOWER=(*number1 number2* **...)**

LOWER=(*number***)**

specifies the lower bounds for the specified parameters. The list can contain exactly one number, which is taken to be the lower bound for all the listed parameters in the statement, or it must contain as many values as the number of parameters specified. A missing value, denoted by ., is a permissible value, which signifies that the parameter has no lower bound.

UPPER=(*number1 number2* **...)**

UPPER=(*number***)**

specifies the upper bounds for the specified parameters. The list can contain exactly one number, which is taken to be the upper bound for all the listed parameters in the statement, or it must contain as many values as the number of parameters specified. A missing value, denoted by ., is a permissible value, which signifies that the parameter has no upper bound.

Programming Statements

To define the model, you can use most of the programming statements that are allowed in the SAS DATA step. For more information, see the *SAS Language Reference: Dictionary*. For the most part, the syntax of programming statements used in PROC SSM is identical to that used in the MODEL procedure (see Chapter 24, "The MODEL Procedure") and the NLMIXED procedure (see Chapter 86, "The NLMIXED Procedure" (*SAS/STAT User's Guide*)). However, there are some restrictions: the DATA step lagging and differencing functions are not allowed, and the use of character variables in the DATA step expressions is not permitted. These are not serious restrictions; usually you can overcome them by adding the variables that are created by such operations to the input data set before its use in the SSM procedure.

STATE Statement

STATE *name* **(***dim***)**< *options* > **;**

The STATE statement specifies a subsection of $\boldsymbol{\alpha}_t$, the overall state vector at time t (for more information, see the section "State Space Model and Notation" on page 2426). Consider the state equations that define the state space model:

$$\begin{aligned} \boldsymbol{\alpha}_{t+1} &= \mathbf{T}_t\boldsymbol{\alpha}_t + \mathbf{W}_{t+1}\boldsymbol{\gamma} + \mathbf{c}_{t+1} + \boldsymbol{\eta}_{t+1} \\ \boldsymbol{\alpha}_1 &= \mathbf{c}_1 + \mathbf{A}_1\boldsymbol{\delta} + \mathbf{W}_1\boldsymbol{\gamma} + \boldsymbol{\eta}_1 \end{aligned}$$

You can specify multiple STATE statements, each specifying a separate subsection. It is assumed that the subsections that are specified by using different STATE statements are mutually independent. This independence assumption implies a block-diagonal structure for the transition matrices $\mathbf{T}_t$ and the disturbance covariances $\mathbf{Q}_t$ for all $t \geq 1$. An appropriate block structure also applies to $\mathbf{W}_t$ and $\mathbf{A}_1$. The *options* in the STATE statement provide complete control over the description of the relevant blocks of $\mathbf{T}_t$, $\mathbf{Q}_t$, $\mathbf{W}_t$, and $\mathbf{A}_1$. The argument *dim* (a positive integer in *name* (*dim*)) specifies the nominal dimension of this subsection. In most situations, the nominal dimension and the actual dimension of the state subsection are the same. However, when you specify the TYPE= option, the actual dimension of the state subsection can be different

from the nominal dimension. The TYPE= option simplifies the state specification task for some commonly needed models.

NOTE: The T, COV, W, and COV1 options, described later in this section, specify the relevant blocks of $\mathbf{T}_t$, $\mathbf{Q}_t$, $\mathbf{W}_t$, and $\mathbf{Q}_1$, respectively. The structure of these matrix blocks is described in a similar way in the option descriptions. For example, the specification COV(I) corresponds to the identity form, COV(D) corresponds to the diagonal form, and COV(G) corresponds to the general form of the $\mathbf{Q}_t$ block.

You can use the following *options* in the STATE statement to specify the system matrices $\mathbf{T}_t$, $\mathbf{Q}_t$, $\mathbf{W}_t$, and $\mathbf{A}_1$ and to request printing of their estimates when they contain unknown parameters. You can also request the checking of unexpected changes—structural breaks—in the evolution of this state subsection by using the CHECKBREAK option.

A1(*nd***)**
treats the last *nd* elements of the state subsection as diffuse. This becomes the dimension of the relevant subsection of the diffuse vector $\boldsymbol{\delta}$. The $\mathbf{A_1}$ block is created by using appropriate columns of the identity matrix. The value of *nd* must lie between 1 and the nominal dimension, *dim*. The absence of this option signifies that this subsection of $\boldsymbol{\alpha}_t$ is nondiffuse. If both the COV1 and A1 options are specified, the last *nd* rows and columns of the matrix specified in the COV1 option are taken to be 0. This option cannot be used together with the RANK= option of the COV1 option.

CHECKBREAK< (ELEMENTWISE | OVERALL) >
turns on the checking of breaks for this state subsection. The ELEMENTWISE suboption requests the elementwise checking of any unexpected change in the state subsection as it evolves from one time point to the next. The OVERALL suboption requests a similar check for the entire state subsection—that is, in this case the change is measured as a multidimensional change. The ELEMENTWISE suboption is the default. Unless the PRINT=BREAKDETAIL option is specified, only a summary of the most significant breaks is produced. If the PRINT=BREAKDETAIL is specified, tables that contain the break significance statistics at every distinct time point are produced—one for the ELEMENTWISE suboption and one for the OVERALL suboption. For more information about the structural break detection process, see the section "Structural Breaks in the State Evolution" on page 2439. For an example of the use of the CHECKBREAK option, see Example 33.8.

COV(D) < = (*var1 var2 . . .***) | (***number1 number2 . . .***) >**
COV(G) < = (*var1 var2 . . .***) | (***number1 number2 . . .***) >**
COV(I) < = (*variable***) | (***number***) >**
COV(RANK=*integer* **)**
specifies the relevant block of the disturbance covariance Q_t (for $t \geq 2$) in the transition equation. As with the T option, the absence of this option signifies that this Q-block consists of only zeros. The structure of the Q-block is also similarly specified. However, the following differences exist:

- The list that is specified to form the covariance must result in a symmetric, positive semidefinite matrix. For an example, see Example 33.5.
- You can specify a rank constraint on the Q-block by specifying COV(RANK=*integer*), where the specified integer must lie between 1 and *dim*. A rank constraint is permissible only for the general form and only when its elements are not specified by using a list.
- The convention of treating unset variables as structural zeros, which is used in specifying sparsity of the T-block, is not used in the Q-block specification. Whenever you explicitly specify the entries of the Q-block by specifying a list of variables in parentheses, all variables in the list must evaluate to nonmissing values.

The following examples illustrate different ways of specifying a Q-block. It is assumed that *dim* = 2.

- COV(G) specifies a general-form Q-block, which contributes $(2 * (2 + 1))/2 = 3$ unspecified elements to the parameter vector $\boldsymbol{\theta}$.
- COV(RANK=1) specifies a rank-one Q-block.

COV1(D) < = (*var1 var2 . . .***) | (***number1 number2 . . .***) >**

COV1(G) < = (*var1 var2 . . .***) | (***number1 number2 . . .***) >**

COV1(I) < = (*variable***) | (***number***) >**

COV1(RANK=*integer* **)**

specifies the relevant block of the initial state covariance Q_1. The different options in this case have the same meaning as the options of the COV option. However, the following differences exist:

- If the elements of Q_1 are specified by a list of variables in parentheses, then these variables must evaluate to constant values. In particular, they can depend on parameters that are specified by the PARMS statements; however, they cannot depend on any of the input data columns.
- If the initial condition is partially diffuse (that is, the diffuse dimension *nd* specified in the A1 option is nonzero), the last *nd* rows and columns of the matrix specified in COV1 are taken to be zero. Moreover, if the elements of Q_1 are specified by a list, its number of elements must correspond to a matrix of dimension (*dim* – *nd*).

PRINT=AR | BREAKDETAIL | COV | COV1 | MA | T

PRINT=(< AR > < BREAKDETAIL > < COV > < COV1 > < MA > < T >)

requests printing of the respective system matrices and the printing of the break statistics at each distinct time point. You can specify PRINT=AR or PRINT=MA only if you specify the TYPE=VARMA option. If any of these matrices are time-varying, the matrix that corresponds to the first time instance is printed. For the BREAKDETAIL suboption to have any effect, the CHECKBREAK option must be turned on. If TYPE= option is used, the result of PRINT=COV can be different than the matrix supplied in the COV= option.

SINPUT = (*var1 var2 . . .***) | (***number1 number2 . . .***)**

specifies the relevant *dim*-dimensional block of the state input vector $\mathbf{c}_t$. The absence of this option signifies that this block of the $\mathbf{c}_t$ vector consists of only zeros. If the elements of $\mathbf{c}_t$ are specified by a list of variables in parentheses, then these variables must be independent of unknown parameters. In particular, they cannot be functions of parameters that are defined by the PARMS statements.

T(D) < = (*var1 var2 . . .***) | (***number1 number2 . . .***) >**

T(G) < = (*var1 var2 . . .***) | (***number1 number2 . . .***) >**

T(I) < = (*variable***) | (***number***) >**

specifies the relevant block of the transition matrix T_t. The absence of this option signifies that this block consists of only zeros. You can specify the structure of the T-block by specifying T(I) for the identity form, T(D) for the diagonal form, and T(G) for a general unstructured form. In addition, you can explicitly specify the entries of the T-block by specifying a list of numbers in parentheses, or by specifying in parentheses a list of variables that are defined by using the programming statements. The unspecified elements of the T-block are included in the list of parameters to be estimated from the data. If the elements of the T-block are supplied by a list in parentheses, the number of elements in the list depends on its structure. For the diagonal form, the list must contain exactly *dim* elements.

In the case of the identity form—T(I)—the block is already fully specified; however, a specification T(I)=(*variable*) is understood to mean that the identity block is scaled by the specified *variable* (or a *number*). In the general case—T(G)—the list must consist of *dim* ∗ *dim* elements, specified in a rowwise fashion. An inappropriate number of elements in the list results in a syntax error.

The following examples illustrate different ways of specifying the transition matrix. It is assumed that *dim* = 2.

- T(I) specifies that the T-block is a two-dimensional identity matrix.
- T(D) specifies that the T-block is a two-dimensional diagonal matrix. The two unspecified diagonal entries become part of the parameter vector $\boldsymbol{\theta}$.
- T(D)=(1.1 2) fully specifies the two-dimensional diagonal T-block.
- T(D)=(X_1 X_2) specifies a two-dimensional diagonal T-block where the diagonal elements are dynamically calculated based on the values of the variables X_1 and X_2. In this case the T-block can change with time if X_1 or X_2 changes with time.
- T(G) specifies a general form T-block (with $2^2 = 4$ unspecified elements).
- T(G)=(X_1 X_2 X_3 X_4) specifies a general form T-block where the first row is formed by X_1 and X_2, and the second row is formed by X_3 and X_4.

In practice the transition matrix is often sparse—that is, many of its elements are 0. The algorithms in the SSM procedure exploit this sparsity structure for computational efficiency. Whenever you explicitly specify the entries of the T-block by specifying a list of variables in parentheses, you can leave the variables that correspond to the zero elements *unset*. These unset variables are treated as structural zeros by the SSM procedure. The section "Sparse Transition Matrix Specification" on page 2431 further explains how to use this sparsity convention.

TYPE=WN

TYPE=RW

TYPE=LL < (SLOPECOV(I | D | G) < = (*var1, var2, . . .***) | (***number1, number2, . . .***) >) >**

TYPE=LL (SLOPECOV(RANK=*integer***))**

TYPE=SEASON (LENGTH=*integer* **< DROPH=***number-list* **> < KEEPH=***number-list* **>)**

TYPE=CYCLE < (< CT > < RHO=*variable* | *number* **> < PERIOD=***variable* | *number* **>) >**

TYPE=VARMA (< p < (I | D) > =*integer* **> < q< (D) >=***integer* **>)**

specifies a state subsection that corresponds to the specified type. You can specify either a number or a variable for the RHO= and PERIOD= suboptions. When TYPE=VARMA, the autoregressive and moving average orders can be at most 1 ($0 \leq p \leq 1$ and $0 \leq q \leq 1$). Moreover, by using the D and I flags with the order specification, you can impose additional structure on the autoregressive and moving average coefficient matrices—for example, specifying TYPE=VARMA(P=1) implies a VAR(1) model with general autoregressive coefficient matrix, whereas specifying TYPE=VARMA(P(D)=1) implies a VAR(1) model with diagonal autoregressive coefficient matrix. If you specify the TYPE= option, the T, COV1, SINPUT, and A1 options are not needed. In fact they are ignored, since the transition matrix $\mathbf{T}_t$ and the matrices in the initial condition ($\mathbf{Q}_1$ and $\mathbf{A}_1$) are implicitly defined by the choice of the type. However, the COV and W options can be useful. In fact, the specification of the COV option does play a key role in the eventual form of $\mathbf{Q}_t$—the covariance of the disturbance term in the transition equation. For the types LL, CYCLE, SEASON, and VARMA, the dimension of the resulting state subsection is a certain multiple of *dim*, the nominal dimension in the STATE statement. For example, the following specification results in a state subsection, named cycleState, of dimension 2**dim*:

```
state cycleState(dim) cov(g) type=cycle;
```

The name cycleState corresponds to the state underlying a *dim*-dimensional cycle component. All of these special state types require that the data be regular (replication is permissible); the only exception is TYPE=CYCLE(CT), which defines a continuous-time cycle and is applicable to any data type. Table 33.2 summarizes some of this information for easy reference. For more information about these state types, see the section "Predefined Structural Models" on page 2445.

The TYPE=LL specification results in a state that corresponds to a multivariate local linear trend. It is governed by two covariance matrices: the COV option specifies the covariance that corresponds to the level equation, and the SLOPECOV suboption specifies the covariance used in the slope equation. The omission of the SLOPECOV suboption signifies that the covariance used in the slope equation is zero. The form of the SLOPECOV suboption is exactly the same as that of the COV option.

The TYPE=CYCLE option results in a state that corresponds to a (stochastic) cycle. By default, this cycle is assumed to be for the regular data type. If TYPE=CYCLE(CT), the resulting cycle is applicable to any data type. The CT option is available only for *dim = 1*; that is, only a univariate cycle is available for the irregular data type. The cycle specification depends on a covariance matrix and two numbers: the damping factor RHO and the cycle period PERIOD. The covariance can be specified by the COV option. The damping factor is specified by the RHO= suboption; its value must lie between 0.0 and 1.0. The cycle period can be specified by the PERIOD= suboption. If the CT suboption is not included, the period value must be larger than 2.0. On the other hand, if the CT suboption is included, its value must be strictly positive. If these parameters are not specified, they are estimated from the data.

The TYPE=SEASON(LENGTH=*integer*) specifies a multivariate trigonometric season that contains the full set of harmonics (for more information, see "Multivariate Season" on page 2448). In some cases, you might want to drop some of the harmonics from this complete set to obtain a more parsimonious trigonometric season specification. You can use the DROPH= (to drop) or KEEPH= (to keep) suboption to control the harmonics that are included in the season specification as follows:

TYPE=SEASON(
LENGTH=*integer*
<DROPH=*number-list* | *n* **TO** *m* **BY** *p***>**
<KEEPH=*number-list* | *n* **TO** *m* **BY** *p***>**
)

The DROPH= and KEEPH= lists can include any integer between 1 and *LENGTH*/2 if the season length is even and any integer between 1 and (*LENGTH* – 1)/2 if the season length is odd. For example, the following specification results in a specification of a trigonometric season with a season length 12 that consists of only the first four harmonics $\boldsymbol{\zeta}_j$, $j = 1, 2, 3, 4$:

```
type=season(length=12  DROPH=5 6) ...;
```

The last two *high*-frequency harmonics, $\boldsymbol{\zeta}_5$ and $\boldsymbol{\zeta}_6$, are dropped. The DROPH= suboption cannot be used with the KEEPH= suboption.

Table 33.2 Summary of Predefined State Types

Type	Description	Parameters	State Dimension
WN	*dim*-variate white noise	COV	*dim*
RW	*dim*-variate random walk	COV	*dim*
LL	*dim*-variate local linear	COV, SLOPECOV	2**dim*
SEASON(LENGTH=*length*)	*dim*-variate season	COV	(*length*–1)**dim*
CYCLE	*dim*-variate cycle	COV, RHO, PERIOD	2**dim*
VARMA(P=*p* Q=*q*)	*dim*-variate VARMA(*p*, *q*)	COV, AR, MA	*dim**max(*p*, *q*+1)

W(D)= (*var1 var2 . . .***) | (***number1 number2 . . .***)**

W(G)= (*var1 var2 . . .***) | (***number1 number2 . . .***)**

W(I) < = (*variable***) | (***number***) >**

specifies the relevant block of the design matrix $\mathbf{W}_t$ in the transition equation. The W-block is of dimension $sdim \times sg$, where *sdim* denotes the actual dimension of the state subsection (which can be the same as *dim*, the nominal dimension, or different if the TYPE= option is used) and *sg* denotes the desired size of the subsection of the overall state regression vector $\boldsymbol{\gamma}$. The absence of this option signifies that the state equation does not contain any regression effects. The number of variables supplied in the W(G)= list option must be a multiple of *sdim*. For example, if *sdim* = 4 and the W(G)= list contains 8 variables, then the implied size of $\boldsymbol{\gamma}$ subsection is 2. If the W(D)= or W(I)= option is used, then the W-block is assumed to be an $sdim$-dimensional diagonal matrix and the W(D)= list must contain exactly *sdim* variables. For examples of the use of this option, see Example 33.8, Example 33.10, and Example 33.11.

TREND Statement

TREND *name* **(***type***)**< *options* > **;**

The TREND statement defines a term in the model that follows a stochastic pattern of a certain predefined type. The *options* in the TREND statement enable you to specify a wide variety of commonly used stochastic patterns. Each TREND statement in effect stands for a special pair of STATE and COMPONENT statements. You can specify more than one TREND statement. Each separate TREND statement defines a component that is assumed to be independent of all other component specifications in the model. Very often the TREND statement is used to specify a component that captures the time-varying level of the data. However, in many cases it is also used to define components of a more general nature; for example, it can be used to define a noise component that follows a stationary ARMA model.

You can refer to the state that is associated with a TREND statement by appending the string "_state_" to the end of its name. For example, name_state_ is the state that is associated with a trend named name. You can use name_state_ in a COMPONENT statement to define a linear combination of its elements. The estimate of this linear combination can then be printed or output to a data set. The nominal dimension of name_state_ is taken to be 1, or the number of variables in the list that is specified in the CROSS= option in the TREND statement that is used to define name (see Example 33.4 for an example of such use of the COMPONENT statement).

Some of these trend specifications are applicable to all the data types—that is, they can be used for both regular data types and irregular data types, whereas the others require that the data be regular or regular with replication. Of course, the trend specification is only part of the overall model specification. Therefore, the other parts of the model can imply additional constraints on the data type.

Table 33.3 lists the available trend models and their data requirements. The *type* column shows the admissible keywords that signify the particular trend type. For brevity, the Data Type column groups the data types regular and regular with replication into one category: regular. For more information about these trend models, see the section "Predefined Trend Models" on page 2442.

Table 33.3 Summary of Trend Types

type	**Data Type**	**Description**	**Parameters**
ARIMA(P=*integer* D=*integer* ...)	Regular	ARIMA model specification	AR and MA coefficients, and the error variance σ^2
DLL	Regular	Damped local linear	Level and slope σ_1^2, σ_2^2, damping factor ϕ
LL	Regular	Local linear	Level and slope σ_1^2, σ_2^2
RW	Regular	Random walk	Level σ^2
DECAY	Irregular	A type of decay pattern	Level σ^2, decay rate ϕ
DECAY(OU)	Irregular	Ornstein-Uhlenbeck decay pattern	Level σ^2, decay rate ϕ
GROWTH	Irregular	A type of growth pattern	Level σ^2, growth rate ϕ
GROWTH(OU)	Irregular	Ornstein-Uhlenbeck growth pattern	Level σ^2, growth rate ϕ
PS(*order*)	Irregular	Polynomial spline of a given order	Level σ^2

The keyword specification of different trend types, except possibly the ARIMA trend, is quite simple. For example, the following statement specifies polySpline as a trend of the type second-order polynomial spline:

```
trend polySpline(ps(2));
```

Similarly, the following statement defines dampedTrend as a damped local linear trend:

```
trend dampedTrend(dll) slopevar=x;
```

The variance parameter that governs the slope equation of this trend type is given by a variable x, which must be defined elsewhere in the program. The other parameters that define dampedTrend are left unspecified (and are estimated by using the data).

The ARIMA trend specification permits specification of trends that follow an ARIMA$(p,d,q)\times(P,D,Q)_s$ model. The specification of ARIMA models requires some notation, which is explained first.

Let B denote the backshift operator—that is, for any sequence ζ_t, $B\zeta_t = \zeta_{t-1}$. The higher powers of B represent larger shifts (for example, $B^3\zeta_t = \zeta_{t-3}$). A random sequence ζ_t follows an ARIMA$(p,d,q)\times(P,D,Q)_s$ model with nonseasonal autoregressive order p, seasonal autoregressive order P, nonseasonal differencing order d, seasonal differencing order D, nonseasonal moving average order q, and seasonal moving average order Q if it satisfies the following difference equation, which is specified in terms of the polynomials in the backshift operator, where a_t is a white noise sequence and s is the season length:

$$\phi(B)\Phi(B^s)(1-B)^d(1-B^s)^D\zeta_t = \theta(B)\Theta(B^s)a_t$$

The polynomials ϕ, Φ, θ, and Θ are of orders p, P, q, and Q, respectively, which can be any nonnegative integers. The season length s must be a positive integer. For example, ζ_t satisfies an ARIMA(1,0,1) model (that is, $p = 1, d = 0, q = 1, P = 0, D = 0$, and $Q = 0$) if

$$\zeta_t = \phi_1 \zeta_{t-1} + a_t - \theta_1 a_{t-1}$$

for some coefficients ϕ_1 and θ_1 and a white noise sequence a_t. Similarly, ζ_t satisfies an ARIMA(0,1,1)×(0,1,1)$_{12}$ model if

$$\zeta_t = \zeta_{t-1} + \zeta_{t-12} - \zeta_{t-13} + a_t - \theta_1 a_{t-1} - \Theta_1 a_{t-12} + \theta_1 \Theta_1 a_{t-13}$$

for some coefficients θ_1 and Θ_1 and a white noise sequence a_t. An ARIMA process is zero-mean, stationary, and invertible if $d = 0$, $D = 0$, and the defining polynomials ϕ, Φ, θ, and Θ have all their roots outside the unit circle—that is, their absolute values are strictly larger than 1.0. It is assumed that the coefficients of the polynomials ϕ, Φ, θ, and Θ are constrained so that the stationarity and invertibility conditions are satisfied. The unknown coefficients of these polynomials become part of the model parameter vector that is estimated by using the data. The general form of the ARIMA trend specification is as follows:

ARIMA(< P=*integer* **> < D=***integer* **> < Q=***integer* **> < SP=***integer* **> < SD=***integer* **> < SQ=***integer* **> < S=***integer* **>)**

By default, the different orders are equal to 0 and the season length is equal to 1. The following examples illustrate a few different ARIMA trend specifications. The following statement defines ima as an integrated moving average trend:

```
trend ima(arima(d=1 q=1));
```

The following statement defines airTrend as a trend that satisfies the well-known airline model (ARIMA(0,1,1)(0,1,1)12 model) for monthly seasonal data:

```
trend airTrend(arima(d=1 q=1 sd=1 sq=1 s=12));
```

The following statement defines arma11 as a zero-mean ARMA(1,1) trend with autoregressive parameter fixed to 0.1:

```
trend arma11(arima(p=1 q=1)) ar=0.1;
```

For an example of the use of the ARIMA trend specification, see Example 33.6.

You can use the following *options* in the TREND statement to specify the trend parameters and to request printing of the trend estimates. In addition, you can create a custom combination of a given trend type by specifying the CROSS= option to create a more general trend. For an example of using the CROSS= option, see the section "Getting Started: SSM Procedure" on page 2394 and the discussion of the second model in Example 33.4. You can also check for the unexpected changes in the trend component by using the CHECKBREAK option.

AR=ϕ_1 ϕ_2 **...** ϕ_p

lists the values of the coefficients of the nonseasonal autoregressive polynomial

$$\phi(B) = 1 - \phi_1 B - \cdots - \phi_p B^p$$

where the order p is specified in the ARIMA trend specification. The coefficients ϕ_i must define a stationary autoregressive polynomial.

CHECKBREAK<(ELEMENTWISE | OVERALL)>

turns on the checking of breaks for this trend component. The ELEMENTWISE suboption requests the elementwise checking of any unexpected change in the state subsection that is associated with the trend component. The OVERALL suboption requests a similar check for the entire state subsection—that is, in this case the change is measured as a multidimensional change. The ELEMENTWISE suboption is the default. Unless the PRINT=BREAKDETAIL option is specified, only a summary of the most significant breaks is produced. If the PRINT=BREAKDETAIL is specified, tables that contain the break significance statistics at every distinct time point are produced—one for the ELEMENTWISE suboption and one for the OVERALL suboption. If the CROSS= option is specified and the CROSS= list contains more than one variable, the OVERALL suboption considers subsections that are associated with each CROSS= variable separately. For more information about the structural break detection process, see the section "Structural Breaks in the State Evolution" on page 2439.

CROSS=(*var1, var2, ...***)**

CROSS(MATCHPARM)=(*var1, var2, ...***)**

creates a linear combination of one or more independent trend components that is based on the variables in the list. If the parameters of the trend are specified by options such as the LEVELVAR= option or the PHI= option, these parameters are shared by these constituent trends. For example, suppose that the CROSS= list contains two variables (X_1 and X_2) and the trend specification is of the type RW. The effect of CROSS=(X_1, X_2) is to create a component $\mu_t = X_1\mu_{1,t} + X_2\mu_{2,t}$, where $\mu_{1,t}$ and $\mu_{2,t}$ are two independent random walk trends. Moreover, if the random walk trend specification uses the LEVELVAR= option to specify the variance parameter, $\mu_{1,t}$ and $\mu_{2,t}$ share the same variance parameter; otherwise, two separate variance parameters are assigned to these random walks. If the second form of the CROSS option, CROSS(MATCHPARM)=, is used, then the constituent trends share all the relevant parameters no matter how they are specified. The CROSS= option is useful for a variety of situations. For example, suppose X is an indicator variable that is 1 before a certain time point t_0 and 0 thereafter. Then CROSS=(X) has the effect of turning off the trend component after time t_0. Similarly, suppose G_1 and G_2 are indicators for gender—for example, G_1 = (GENDER=1) and G_2 = (GENDER=0) for male and female cases, respectively. Then CROSS=(G_1, G_2) results in separate trends according to the gender. The variables in the CROSS= list must be free of unknown parameters.

The CROSS= option can be computationally expensive; computationally it is equivalent to specifying as many separate trends as the number of variables in the specified list.

LEVELVAR=*variable* | *number*

specifies the disturbance variance parameter for all the trend types. For trend types LL and DLL, this option specifies σ_1^2. Any nonnegative value, including 0, is permissible. If *variable* contains unknown parameters, they are estimated from the data. Similarly, if the LEVELVAR= option is not specified, σ^2 is estimated from the data.

MA=$\theta_1\ \theta_2\ \ldots\theta_q$

lists the values of the coefficients of the nonseasonal moving average polynomial,

$$\theta(B) = 1 - \theta_1 B - \cdots - \theta_q B^q$$

where the order q is specified in the ARIMA trend specification. The coefficients θ_i must define an invertible moving average polynomial.

NODIFFUSE

treats the diffuse elements in the initial state of the state subsection underlying the trend component as nondiffuse. This option is applicable to all trend types except ARIMA. For the ARIMA trend type, this option is ignored even if the nonseasonal or seasonal differencing orders are nonzero. The diffuse elements are assumed to be independent, zero-mean, Gaussian variables. Their variances become part of the parameter vector and are estimated by using the data. This option is useful for creating a trend component that can be interpreted as a deviation from an overall trend component (with diffuse initialization), which is defined separately.

PHI=*variable* | *number*

specifies the value of ϕ for trend types DLL, DECAY, DECAY(OU), GROWTH, and GROWTH(OU). For the type DLL, the specified value must be between 0.0 and 1.0. For types DECAY and DECAY(OU), ϕ must be strictly negative. For types GROWTH and GROWTH(OU), ϕ must be strictly positive. If *variable* contains unknown parameters, they are estimated from the data. Similarly, if the PHI= option is not specified, ϕ is estimated from the data.

PRINT=BREAKDETAIL | COV | COV1 | FILTER | SMOOTH | T

PRINT= (< BREAKDETAIL > < COV > < COV1 > < FILTER > < SMOOTH > < T >)

requests printing of the respective system matrices of the state equation that underlies the specified trend, the printing of its filtered and smoothed estimates, and the printing of the break statistics at each distinct time point. For the BREAKDETAIL suboption to have any effect, the CHECKBREAK option must be turned on. If any of these matrices are time-varying, the matrix that corresponds to the first time instance is printed.

SAR=$\Phi_1\ \Phi_2 \ldots \Phi_P$

lists the values of the coefficients of the seasonal autoregressive polynomial

$$\Phi(B^s) = 1 - \Phi_1 B^s - \cdots - \Phi_P B^{sP}$$

where the order P is specified by using the SP= option in the ARIMA trend specification and the season length s is specified in the S= option. The coefficients Φ_i must define a stationary autoregressive polynomial.

SMA=$\Theta_1\ \Theta_2 \ldots \Theta_Q$

lists the values of the coefficients of the seasonal moving average polynomial

$$\Theta(B^s) = 1 - \Theta_1 B^s - \cdots - \Theta_Q B^{sQ}$$

where the order Q is specified by using the SQ= option in the ARIMA trend specification and the season length s is specified in the S= option. The coefficients Θ_i must define an invertible moving average polynomial.

SLOPEVAR=*variable* | *number*

specifies the second disturbance variance parameter, σ_2^2, for trend types LL and DLL. Any nonnegative value, including 0, is permissible. If *variable* contains unknown parameters, they are estimated from the data. Similarly, if the SLOPEVAR= option is not specified, σ_2^2 is estimated from the data.

Details

Throughout this section, vectors and matrices are denoted by boldface letters. Generally, Greek letters (such as $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, and $\boldsymbol{\epsilon}$) denote unobserved or latent quantities—often estimated from the data—that represent model parameters, latent states, or noise variables. Capital letters such as X, Y, and Z are used to denote the observed data variables. Whenever there is no ambiguity, it is assumed that the matrices have appropriate dimensions when they are being multiplied—in particular, the vectors behave as column vectors or row vectors as the need arises. On many occasions, matrices are described inline—that is, they are described as parenthesized lists, in a rowwise fashion, with the rows separated by a comma. The term "dot product" is used to describe the scalar that results from the product of a row vector with a (conforming) column vector.

State Space Model and Notation

The (linear) state space model is described in the literature in a few different ways and with varying degree of generality. The description given in this section loosely follows the description given in Durbin and Koopman (2012, chap. 6, sec. 4). This formulation of SSM is quite general; in particular, it includes nonstationary SSMs with time-varying system matrices and state equations with a diffuse initial condition (these terms are defined later in this subsection).

Suppose that observations are collected in a sequential fashion (indexed by a numeric variable τ) on some variables: the vector $\mathbf{y} = (y_1, y_2, \ldots, y_q)$, which denotes the q-variate response values, and the k-dimensional vector $\mathbf{x}$, which denotes the predictors. Suppose that the observation instances are $\tau_1 < \tau_2 < \ldots < \tau_n$. The possibility that multiple observations are taken at a particular instance τ_i is not ruled out, and the successive observation instances do not need to be regularly spaced—that is, $(\tau_2 - \tau_1)$ does not need to equal $(\tau_3 - \tau_2)$. For $t = 1, 2, \ldots, n$, suppose p_t (≥ 1) denotes the number of observations recorded at instance τ_t. For notational simplicity, an integer-valued secondary index t is used to index the data so that $t = 1$ corresponds to $\tau = \tau_1$, $t = 2$ corresponds to $\tau = \tau_2$, and so on. Consider the following model:

$$\begin{aligned}
\mathbf{Y}_t &= \mathbf{Z}_t\boldsymbol{\alpha}_t + \mathbf{X}_t\boldsymbol{\beta} + \boldsymbol{\epsilon}_t && \text{Observation equation} \\
\boldsymbol{\alpha}_{t+1} &= \mathbf{T}_t\boldsymbol{\alpha}_t + \mathbf{W}_{t+1}\boldsymbol{\gamma} + \mathbf{c}_{t+1} + \boldsymbol{\eta}_{t+1} && \text{State transition equation} \\
\boldsymbol{\alpha}_1 &= \mathbf{c}_1 + \mathbf{A}_1\boldsymbol{\delta} + \mathbf{W}_1\boldsymbol{\gamma} + \boldsymbol{\eta}_1 && \text{Initial condition}
\end{aligned}$$

The following list describes these equations:

- The *observation equation* describes the relationship between the $(p_t * q)$-dimensional response vector $\mathbf{Y}_t$ and the unobserved vectors $\boldsymbol{\alpha}_t$, $\boldsymbol{\beta}$, and $\boldsymbol{\epsilon}_t$. The q-variate responses are vertically stacked in a column to form this $(p_t * q)$-dimensional response vector $\mathbf{Y}_t$. The m-dimensional vectors $\boldsymbol{\alpha}_t$ are called *states*, the k-dimensional vector $\boldsymbol{\beta}$ is the regression coefficient vector associated with predictors $\mathbf{x}$, and the $(p_t * q)$-dimensional vectors $\boldsymbol{\epsilon}_t$ are called the *observation disturbances*. The matrices $\mathbf{Z}_t$ (of dimension $(q * p_t) \times m$) and $\mathbf{X}_t$ (of dimension $(q * p_t) \times k$) correspond to the *state effect* and the *regression effect*, respectively. The elements of $\mathbf{X}_t$ are assumed to be fully known. The states $\boldsymbol{\alpha}_t$ and the disturbances $\boldsymbol{\epsilon}_t$ are *random* sequences. It is assumed that $\boldsymbol{\epsilon}_t$ is a sequence of independent, zero-mean, Gaussian random vectors with diagonal covariances, with the diagonal elements denoted by $\sigma^2_{t,i}, i = 1, 2, \ldots, q * p_t$.
- The state sequence $\boldsymbol{\alpha}_t$ is assumed to follow a Markovian structure described by the state transition equation and the associated initial condition.

- The *state transition equation* postulates that a new instance of the state, $\boldsymbol{\alpha}_{t+1}$, is obtained by multiplying its previous instance, $\boldsymbol{\alpha}_t$, by an m-dimensional square matrix $\mathbf{T}_t$ (called the state transition matrix) and by adding three more terms: a known nonrandom vector $\mathbf{c}_{t+1}$ (called the state input); a regression term $\mathbf{W}_{t+1}\boldsymbol{\gamma}$, where $\mathbf{W}_{t+1}$ is an $m \times g$-dimensional design matrix with fully known elements and γ is the g-dimensional regression vector; and a random disturbance vector $\boldsymbol{\eta}_{t+1}$. The m-dimensional state disturbance vectors $\boldsymbol{\eta}_t$ are assumed to be independent, zero-mean, Gaussian random vectors with covariances Q_t (not necessarily diagonal).

- The *initial condition* describes the starting condition of the state evolution equation. The starting state vector $\boldsymbol{\alpha}_1$ is assumed to be partially diffuse: it is the sum of a known nonrandom vector $\mathbf{c}_1$, a mean-zero Gaussian vector $\boldsymbol{\eta}_1$, and the terms $\mathbf{A}_1\boldsymbol{\delta}$ and $\mathbf{W}_1\boldsymbol{\gamma}$. $\mathbf{A}_1\boldsymbol{\delta}$ represents the contribution from a d-dimensional diffuse vector $\boldsymbol{\delta}$ (a diffuse vector is a Gaussian vector with infinite covariance). The observation and state regression vectors $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are also assumed to be diffuse. The $m \times d$ matrix $\mathbf{A}_1$ is assumed to be completely known.

- The observation disturbances $\boldsymbol{\epsilon}_t$ and the state disturbances $\boldsymbol{\eta}_t$ (for $t \geq 1$) are assumed to be mutually independent. Either the elements of the matrices $\mathbf{Z}_t$, $\mathbf{T}_t$, and Q_t and the diagonal elements of the observation disturbance covariances $\sigma^2_{t,i}$ are assumed to be completely known, or some of them can be functions of a small set of unknown parameters (to be estimated from the data). Suppose that this unknown set of parameters is denoted by $\boldsymbol{\theta}$.

- The d-dimensional diffuse vector $\boldsymbol{\delta}$ from the state initial condition together with the observation and state regression vectors $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ constitute the overall $(d + k + g)$-dimensional diffuse initial condition of the model. For more information, see the section "Filtering, Smoothing, Likelihood, and Structural Break Detection" on page 2433.

Although this description of the state space model might appear involved, it conveniently covers many variants of the SSMs that are encountered in practice and precisely describes the most general case that can be handled by the SSM procedure. An important restriction about the preceding description of the model formulation is that it assumes that the matrices $\mathbf{X}_t$ and $\mathbf{W}_t$ that appear in the observation equation and the state equation respectively are free of unknown parameters and that the covariances of the observation disturbances $\boldsymbol{\epsilon}_t$ are diagonal. In most practical situations, the model under consideration can be easily reformulated to a statistically equivalent form that conforms to this restriction.

NOTE: The transition matrix $\mathbf{T}_t$ in the state equation relates the state $\boldsymbol{\alpha}_t$ at time t with the state $\boldsymbol{\alpha}_{t+1}$ at time $t + 1$. In many situations, such as when the observations are taken at irregular time intervals, $\mathbf{T}_t$ depends on information at both t and $t + 1$. Therefore, it is more appropriate to denote the transition matrix as $\mathbf{T}_t^{t+1}$. However, for simplicity, the former notation is used throughout this chapter. The same comment applies to the covariance matrix $\mathbf{Q}_t$ of the disturbance term $\boldsymbol{\eta}_t$.

For easy reference, Table 33.4 summarizes the information contained in the SSM equations.

Table 33.4 State Space Model: Notation

Notation	Description
$\tau_1, \tau_2, \ldots, \tau_n$	Distinct index values at which the observations are recorded
n	Number of distinct index instances
p_t	Number of observations recorded at index τ_t, $t = 1, 2, \ldots, n$
q	Number of response variables in the model

Table 33.4 *continued*

Notation	Description
$\mathbf{Y}_t = (y_{t,1}, y_{t,2}, \ldots, y_{t,p_t*q})$	Vertically stacked vector of response values recorded at τ_t
$N = q * \sum_{t=1}^{n} p_t$	Total number of response values in the data set
k	Number of predictor (regressor) variables in the observation equation
$\mathbf{X}_t$	$(p_t * q) \times k$ matrix of predictor values recorded at τ_t
$\boldsymbol{\beta}$	k-dimensional regression vector that is associated with the predictors
$\boldsymbol{\epsilon}_t \sim N(0, (\sigma^2_{t,1}, \ldots))$	$(q * p_t)$-dimensional observation disturbance vector with diagonal covariance
m	Dimension of the state vectors $\boldsymbol{\alpha}_t$
$\boldsymbol{\alpha}_t$	m-dimensional state vector
$\mathbf{Z}_t$	$(q * p_t) \times m$ matrix that is associated with $\boldsymbol{\alpha}_t$ in the observation equation
$\mathbf{T}_t$	$m \times m$ state transition matrix
$\mathbf{c}_t$	m-dimensional state input vector
$\mathbf{W}_t$	$m \times g$ design matrix associated with $\boldsymbol{\gamma}$, the state regression vector
$\boldsymbol{\gamma}$	g-dimensional state regression vector
$\boldsymbol{\eta}_t \sim N(0, Q_t)$	m-dimensional state disturbance vector
d	Dimension of the diffuse vector $\boldsymbol{\delta}$ in the state initial condition
$\boldsymbol{\delta} \sim N(0, \kappa\Sigma), \kappa \to \infty$	Diffuse vector in the state initial condition
$\mathbf{A_1}$	$m \times d$ constant matrix associated with $\boldsymbol{\delta}$
$\boldsymbol{\eta}_1 \sim N(0, Q_1)$	m-dimensional state disturbance vector in the initial condition
$\boldsymbol{\theta}$	Parameter vector

Types of Sequence Data

The state space model specification in the SSM procedure requires proper understanding of both the data organization and the form of the model. The SSMs that are appropriate for time series data might not be appropriate for irregularly spaced longitudinal data. The SSM procedure distinguishes three types of data organization based on the way the observations are sequenced by the index variable. If an index variable is not specified, it is assumed that the observations are sequenced according to the observation number.

Regular: The observations are recorded at regularly spaced intervals; that is, $\tau_1, \tau_2, \ldots, \tau_n$ are regularly spaced. Moreover, at each observation instance τ_i a single observation is recorded; that is, $p_t = 1$ for all t. The standard time series data (both univariate and multivariate) fall in this category.

Regular with Replication: The observations are recorded at regularly spaced intervals, but $p_t > 1$ for at least one t. Here the word replication is used loosely—it does not mean that the multiple observations at τ_t are replications in the precise statistical sense. The panel or cross-sectional data types fall into this category. In the panel data case with p cross sections, $p_t = p$ for all t.

Irregular: The observations are not recorded at regular intervals, and the number of observations p_t at each index instance can be different. The longitudinal data fall into this category.

It is not always easy to decide whether the specified model is appropriate for the given data type. Whenever possible, the SSM procedure issues a note regarding the possible mismatch between the specified model and the data type being analyzed.

Overview of Model Specification Syntax

An SSM specification involves the description of the terms in the observation equation, the state transition equation, and the initial condition. For example, the response variables, the predictor variables, and the elements of the state transition matrix $\mathbf{T}_t$ must somehow be specified. The SSM procedure syntax is designed so that little effort is needed to specify the more commonly needed models, while a highly flexible language is available for specifying more complex models. Two syntax features help achieve this goal: the ability to build a complex specification by combining simpler subspecifications, and a programming language for creating lists of variables to be used later in the model specification.

The SSM procedure statements can be divided into two classes:

- programming statements, which are used to create lists of variables that can be used for a variety purposes (for example, as the elements of the model system matrices)
- statements specific to the SSM procedure that formulate the state space model and control its other aspects such as the input data specification and the resulting output

Since the matrices involved in the model specification can be specified as lists of variables, which you separately create by using the programming statements, you can finely control all aspects of the model specification. These programming statements permit the use of most DATA step language features such as the conditional logic (IF-THEN-ELSE), array type variables, and all the mathematical functions available in the DATA step. You can also use programming statements to define predictor variables on the fly.

Building a Complex Model Specification

In addition to being able to specify the system matrices in a flexible way, you can also build a complex model specification in a modular way by combining simpler subspecifications. Suppose that the state vector for the model to be specified is composed of subsections that are statistically independent, which is a common scenario in practical modeling situations. For example, suppose that $\boldsymbol{\alpha}_t$ can be divided into two disjoint subsections $\boldsymbol{\alpha}_t^a$ and $\boldsymbol{\alpha}_t^b$, which are statistically independent. This entails a corresponding block-diagonal structure to the system matrices $\mathbf{T}_t$, $\mathbf{W}_t$, and $\mathbf{Q}_t$ that govern the state equations. In this case the term $\mathbf{Z}_t\boldsymbol{\alpha}_t$ that appears in the observation equation also splits into the sum $\mathbf{Z}_t^a\boldsymbol{\alpha}_t^a + \mathbf{Z}_t^b\boldsymbol{\alpha}_t^b$ for appropriately partitioned matrices $\mathbf{Z}_t^a$ and $\mathbf{Z}_t^b$. The model specification syntax of the SSM procedure makes building an SSM from such smaller pieces easy. Throughout this chapter, the linear combinations of the state subsections (such as $\mathbf{Z}_t^a\boldsymbol{\alpha}_t^a$) that appear in the observation equation are called *components*. An SSM specification in the SSM procedure is created by combining separate component specifications. In general, you specify a component in two steps: first you define a state subsection $\boldsymbol{\alpha}_t^a$, and then you define a matching linear combination $\mathbf{Z}_t^a\boldsymbol{\alpha}_t^a$. For some special components, such as some commonly needed *trend* components, you can combine these two steps into one keyword specification.

The following list summarizes the (nonprogramming) SSM procedure syntax statements used for model specification:

- The ID statement specifies the index variable (τ). It is assumed that the data within each BY group are ordered (in ascending order) according to the ID variable. The SSM procedure automatically creates a variable, _ID_DELTA_, which contains the difference between the successive ID values. This variable is available for use by the programming statements to define time-varying system matrices. For example,

in the case of SSMs used for modeling the longitudinal data, the $\mathbf{T}_t$ and $\mathbf{Q}_t$ matrices often depend on _ID_DELTA_ (see Example 33.5).

- The PARMS statement specifies variables that serve as the parameters of the model. That is, it partially defines the model parameter vector $\boldsymbol{\theta}$. Other elements of $\boldsymbol{\theta}$ are implicitly defined if your specification of the system matrices is not fully complete.
- The STATE statement specifies a subsection of the model state vector. Multiple STATE statements can be used in the model specification; each one defines a statistically independent subsection of the model state vector. For full customization, $\mathbf{T}_t$, $\mathbf{W}_t$, and $\mathbf{Q}_t$ blocks that govern this subsection can be specified as lists of variables that are created by programming statements. However, you can obtain many commonly needed state subsection types simply by using the TYPE= option in this statement. For example, the use of TYPE=SEASON(LENGTH=12) results in a state subsection that can be used to define a monthly seasonal component.
- The COMPONENT statement specifies a linear combination that matches a state subsection that is previously defined in a STATE statement. Thus, a matching pair of STATE and COMPONENT statements define a component.
- The TREND statement is used for easy specification of some commonly needed components that follow stochastic patterns of certain predefined types.
- The IRREGULAR statement specifies the observation disturbance for a particular response variable.
- The DEPLAG statement specifies the terms in the model that involve lagged response variables.
- The MODEL statement specifies the observation equation for one of the response variables. A separate MODEL statement is needed for each response variable in the multivariate case. The MODEL statement specifies an equation in which the left-hand side is the response variable and the right-hand side is a list that contains components and regression variables.

Model Specification Steps

To illustrate the model specification steps, suppose y is a response variable and variables x1 and x2 are predictors. The following statements specify a model for y that includes two components named cycle and randomWalk, predictors x1 and x2, and an observation disturbance named whiteNoise:

```
parms lambda / lower=(1.e-6) upper=(3.14);
parms cycleVar / lower=(1.e-6);
array cycleT{4} c1-c4;
c1 = cos(lambda);
c2 = sin(lambda);
c3 = -c2;
c4 = c1;
state s_cycle(2) T(g)=(cycleT) cov(I)=(cycleVar) a1(2);
component cycle=(1 0)*s_cycle;
trend randomWalk(rw);
irregular whiteNoise;
model y = x1 x2 randomWalk cycle whiteNoise;
```

The specification begins with a PARMS statement that defines two parameters, lambda and cycleVar, along with their lower and upper bounds (essentially 0 and π for lambda, and 0 and infinity for cycleVar). Next, programming statements define an array of variables, cycleT, which contains four variables, c1–c4; these variables are used later for defining the elements of the transition matrix of a state subsection. The STATE statement specifies the two-dimensional subsection s_cycle; the dimension appears within the parentheses after the name (`s_cycle(2)`). The T= option specifies the transition matrix for the s_cycle (`T(g)=(cycleT)`); the `g` in `T(g)` signifies that the form of the T matrix is *general*. The COV= option (`cov(I)=(cycleVar)`) specifies the covariance of the state disturbance ($\mathbf{Q}_t$ for $t \geq 2$); because of the use of `I` in `cov(I)`, the covariance is of the form scaled identity, essentially a two-dimensional diagonal matrix with both diagonal elements equal to cycleVar. The initial condition for s_cycle is completely diffuse because the A1= option, which specifies $\mathbf{A}_1$, specifies that the dimension of the diffuse vector $\boldsymbol{\delta}$ is 2: `a1(2)`. In this case there is no need to specify the covariance $\mathbf{Q}_1$ in the initial condition. The COMPONENT statement specifies the component cycle. It specifies cycle as a dot product of two vectors—(1 0) and s_cycle, which merely selects the first element of s_cycle: `component cycle=(1 0)*s_cycle`. The TREND statement defines a component named randomWalk; its type is *rw*, which signifies random walk. The IRREGULAR statement defines an observation disturbance named whiteNoise. Both the randomWalk and whiteNoise specifications are only partially complete—for example, the disturbance variance of whiteNoise is not specified. These unspecified variances, trendVar, which corresponds to randomWalk, and wnVar, which corresponds to whiteNoise, are automatically included in the list of unknown parameters, $\boldsymbol{\theta}$, along with the parameters that are defined by the PARMS statements. Thus, the parameter vector for this model is $\boldsymbol{\theta}$ = (lambda cycleVar trendVar wnVar). Finally, the model specification is completed by the MODEL statement, which specifies the components of the observation equation: the response variable y, the predictors x1 and x2, the components randomWalk and cycle, and the irregular term whiteNoise.

The preceding statements result in an SSM with a three-dimensional state vector, which is the result of combining the two-dimensional state subsection, s_cycle, and a one-dimensional subsection underlying the trend, randomWalk. In this specification, the initial state is completely diffuse with $\mathbf{Q}_1$ a null matrix, and $\mathbf{A}_1$ equal to the three-dimensional identity. The other state system matrices $\mathbf{T}_t$ and $\mathbf{Q}_t$ are time-invariant:

$$\mathbf{T} = \begin{bmatrix} \cos(\text{lambda}) & \sin(\text{lambda}) & 0 \\ -\sin(\text{lambda}) & \cos(\text{lambda}) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \mathbf{Q} = \begin{bmatrix} \text{cycleVar} & 0 & 0 \\ 0 & \text{cycleVar} & 0 \\ 0 & 0 & \text{trendVar} \end{bmatrix}$$

The observation equation is obvious with $\mathbf{Z} = [1\ 0\ 1]$.

Sparse Transition Matrix Specification

It often happens that the transition matrix $\mathbf{T}$ (or $\mathbf{T}_t$ in the time-varying case) specified in a STATE statement is sparse—that is, many of its elements are zero. The algorithms in the SSM procedure exploit this sparsity for computational efficiency. In most cases the sparsity of a T-block can be inferred from the context. However, if the elements of the T-block are supplied by a list of variables in parentheses, it can be difficult to recognize elements that are structurally zero (this is because of the generality of the DATA step language used for defining the variables). To simplify the specification of such sparse transition matrix, SSM procedure has adopted a convention: the variables that correspond to structural zeros can (and should) be left *unset*—that is, these variables are declared, but no value is assigned to them. As an example, suppose that a three-dimensional state subsection has the following form of transition matrix for some variables X1, X2, and X3 defined elsewhere in the program:

$$\mathbf{T} = \begin{bmatrix} \text{X1} & 0 & 0 \\ \text{X2} & \text{X1} & 0 \\ \text{X3} & 0 & \text{X1} \end{bmatrix}$$

The following (incomplete) statements show how to specify such a T-block:

```
array tMat{3,3};
do i=1 to 3;
   tMat[i, i] = x1;
end;
tMat[2,1] = x2;
tMat[3,1] = x3;
state foo(3) T(g)=(tMat) ...;
```

In this specification only the nonzero elements of the tMat array, which contains $3 \times 3 = 9$ elements, are assigned a value. On the other hand, the following statements show an alternate way of specifying the same T-block. This specification explicitly sets the zeros in the T-block (the elements above the diagonal and tMat[3,2]) to 0.

```
array tMat{3,3};
do i=1 to 3;
   do j=1 to 3;
       if i=j then tMat[i, j] = x1;
       else if j > i then tMat[i, j] = 0;
   end;
end;
tMat[2,1] = x2;
tMat[3,1] = x3;
tMat[3,2] = 0;
state foo(3) T(g)=(tMat) ...;
```

The first specification is simpler, and is preferred. The second specification is mathematically equivalent (and generates the same output) but is computationally less efficient since its sparsity structure cannot always be reliably inferred due to the generality of the DATA step language. In the first specification, the unset elements are recognized to be *structural* zeros while the set elements are treated as nonzero for sparsity purposes. For a simple illustration, see Example 33.5. Proper sparsity specification can lead to significant computational savings for larger matrices.

Regression Variable Specification in Multivariate Models

Suppose that a regression variable in a multivariate model affects two or more response variables. For example, suppose that response variables y1 and y2 depend on a regression variable x. This dependence can be categorized as one of two types:

- In the more common case, the regression coefficient of x for y1 and the regression coefficient of x for y2 are different. The relationship can be described as follows:

$$\begin{aligned} y1 &= \beta_1 x + \text{other terms} \\ y2 &= \beta_2 x + \text{other terms} \end{aligned}$$

  In the SSM procedure you can specify this type of relationship in two equivalent ways:

  – You can specify the variable x in the MODEL statement for y1 and specify the variable x_copy (a copy of x) in the MODEL statement for y2 as follows:

```
x_copy = x;             /* create a copy of x */
model y1 = x ...;
model y2 = x_copy ...;
```

– You can specify the variable x in MODEL statements for both y1 and y2 as follows:

```
model y1 = x ...;
model y2 = x ...;
```

This specification avoids creating x_copy.

Of these two alternate ways, the first is preferred because x and x_copy can then be unambiguously used in an EVAL statement to refer to the terms $\beta_1 x$ and $\beta_2 x$, respectively.

- In the less common case, y1 and y2 share a common regression coefficient.The relationship can be described as follows:

$$\begin{aligned} y1 &= \beta x + \text{other terms} \\ y2 &= \beta x + \text{other terms} \end{aligned}$$

You can specify this type of relationship by placing the regression coefficient in the model state vector as follows:

```
state beta(1) T(I) A1(1) ;       /* beta is a constant state */
comp xeffect = beta*(x) ;
model y1 =  xeffect ...;
model y2 =  xeffect ...;
```

Here the STATE statement defines beta as a one-dimensional, time-invariant constant (because the transition matrix is identity, the disturbance covariance is 0 and the initial state is diffuse). Next, the COMP statement defines xeffect as the product between beta and the variable x. Subsequently, both y1 and y2 use xeffect in their respective MODEL statements.

Filtering, Smoothing, Likelihood, and Structural Break Detection

The Kalman filter and smoother (KFS) algorithm is the main computational tool for using SSM for data analysis. This subsection briefly describes the basic quantities generated by this algorithm and their relationship to the output generated by the SSM procedure. For proper treatment of SSMs with a diffuse initial condition or when regression variables are present, a modified version of the traditional KFS, called diffuse Kalman filter and smoother (DKFS), is needed. A good discussion of the different variants of the traditional and diffuse KFS can be found in Durbin and Koopman (2012). The DKFS implemented in the SSM procedure closely follows the treatments in De Jong and Chu-Chun-Lin (2003) and Francke, Koopman, and de Vos (2010). Additional details can be found in these references.

The state space model equations (see the section "State Space Model and Notation" on page 2426) imply that the combined response data vector $\mathbf{Y} = (\mathbf{Y}_1, \mathbf{Y}_2, \ldots, \mathbf{Y}_n)$ has a Gaussian probability distribution. This probability distribution is *proper* if d, the dimension of the diffuse vector $\boldsymbol{\delta}$ in the initial condition, is 0 and if $(k + g)$, the total number of regression variables in the observation and state equations, is also 0 (the

regression vectors $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ are also treated as a diffuse vectors). Otherwise, this probability distribution is *improper*. The KFS algorithm is a combination of two iterative phases: a forward pass through the data, called *filtering*, and a backward pass through the data, called *smoothing*, that uses the quantities generated during filtering. One of the advantages of using the SSM formulation to analyze the time series data is its ability to handle the missing values in the response variables. The KFS algorithm appropriately handles the missing values in $\mathbf{Y}$. For additional information about how PROC SSM handles missing values, see the section "Missing Values" on page 2456.

Filtering Pass

The filtering pass sequentially computes the quantities shown in Table 33.5 for $t = 1, 2, \ldots, n$ and $i = 1, 2, \ldots, q * p_t$.

Table 33.5 KFS: Filtering Phase

Quantity	Description
$\hat{y}_{t,i} = \mathrm{E}(y_{t,i} \mid y_{t,i-1}, \ldots y_{t,1}, \mathbf{Y}_{t-1}, \ldots, \mathbf{Y}_1)$	One-step-ahead prediction of the response values
$v_{t,i} = y_{t,i} - \hat{y}_{t,i}$	One-step-ahead prediction residuals
$F_{t,i} = \mathrm{Var}(y_{t,i} \mid y_{t,i-1}, \ldots y_{t,1}, \mathbf{Y}_{t-1}, \ldots, \mathbf{Y}_1)$	Variance of the one-step-ahead prediction
$\hat{\boldsymbol{\alpha}}_{t,i} = \mathrm{E}(\boldsymbol{\alpha}_t \mid y_{t,i-1}, \ldots y_{t,1}, \mathbf{Y}_{t-1}, \ldots, \mathbf{Y}_1)$	One-step-ahead prediction of the state vector
$\mathbf{P}_{t,i} = \mathrm{Cov}(\boldsymbol{\alpha}_t \mid y_{t,i-1}, \ldots y_{t,1}, \mathbf{Y}_{t-1}, \ldots, \mathbf{Y}_1)$	Covariance of $\hat{\boldsymbol{\alpha}}_{t,i}$
$\mathbf{b}_{t,i}$	$(d + k + g)$-dimensional vector
$\mathbf{S}_{t,i}$	$(d + k + g)$-dimensional symmetric matrix
$\left(\hat{\boldsymbol{\delta}} \;\; \hat{\boldsymbol{\beta}} \;\; \hat{\boldsymbol{\gamma}}\right)'_{t,i} = \mathbf{S}^{-1}_{t,i}\mathbf{b}_{t,i}$	Estimates of $\boldsymbol{\delta}$, $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$ by using the data up to (t, i)
$\mathbf{S}^{-1}_{t,i}$	Covariance of $\left(\hat{\boldsymbol{\delta}} \;\; \hat{\boldsymbol{\beta}} \;\; \hat{\boldsymbol{\gamma}}\right)'_{t,i}$
$\mathbf{S}^{*}_{t,i}$	$(d + k + g)$-dimensional symmetric matrix needed in the marginal likelihood computation

Here the notation $\mathrm{E}(y_{t,i} \mid y_{t,i-1}, \ldots, y_{t,1}, \mathbf{Y}_{t-1}, \ldots, \mathbf{Y}_1)$ denotes the *conditional expectation* of $y_{t,i}$ given the history up to the index $(t, i-1)$: $(y_{t,i-1}, \ldots, y_{t,1}, \mathbf{Y}_{t-1}, \ldots, \mathbf{Y}_1)$. Similarly $\mathrm{Var}(y_{t,i} \mid y_{t,i-1}, \ldots, y_{t,1}, \mathbf{Y}_{t-1}, \ldots, \mathbf{Y}_1)$ denotes the corresponding conditional variance. The quantity $v_{t,i} = y_{t,i} - \hat{y}_{t,i}$ is set to missing whenever $y_{t,i}$ is missing. Note that $\hat{y}_{t,i}$ are *one-step-ahead* forecasts only when the model has only one response variable and the data are a time series; in all other cases it is more appropriate to call them *one-measurement-ahead* forecasts (since the next measurement might be at the same time point). Despite this, $\hat{y}_{t,i}$ are called one-step-ahead predictions (and $v_{t,i}$ are called one-step-ahead residuals) throughout this document. In the diffuse case, the conditional expectations must be appropriately interpreted. The vector $\mathbf{b}_{t,i}$ and the matrix $\mathbf{S}_{t,i}$ contain some accumulated quantities that are needed for the estimation of $\boldsymbol{\delta}$, $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$. Of course, when $(d + k + g) = 0$ (the nondiffuse case), these quantities are not needed. In the diffuse case, because the matrix $\mathbf{S}_{t,i}$ is sequentially accumulated (starting at $t = 1, i = 1$), it might not be invertible until some $t = t_*, i = i_*$. The filtering process is called *initialized* after $t = t_*, i = i_*$. In some situations, this initialization might not happen even after the entire sample is processed—that is, the filtering process remains *uninitialized*. This can happen if the regression variables are collinear or if the data are not sufficient to estimate the initial condition $\boldsymbol{\delta}$ for some other reason. In

the diffuse case if the marginal likelihood is to be computed (see the section "Likelihood Computation and Model-Fitting Phase" on page 2435), an additional matrix ($\mathbf{S}^*_{t,i}$) is computed at each step by sequential accumulation.

The filtering process is used for a variety of purposes. One important use of filtering is to compute the likelihood of the data. In the model-fitting phase, the unknown model parameters $\boldsymbol{\theta}$ are estimated by maximum likelihood. This requires repeated evaluation of the likelihood at different trial values of $\boldsymbol{\theta}$. After $\boldsymbol{\theta}$ is estimated, it is treated as a known vector. The filtering process is used again with the fitted model in the forecasting phase, when the one-step-ahead forecasts and residuals based on the fitted model are provided. In addition, this filtering output is needed by the smoothing phase to produce the full-sample component estimates and for the structural break analysis.

Likelihood Computation and Model-Fitting Phase

Because of the Gaussian nature of the response vector, the likelihood of $\mathbf{Y}$ can be computed by using the prediction-error decomposition. The desired prediction-error decomposition is obtained by the filtering pass described in the previous section ("Filtering Pass" on page 2434). When the state space model under consideration has a nondiffuse initial condition and no regression effects are present in the observation and state equations, $\mathbf{Y}$ has a proper Gaussian distribution and its likelihood is defined unambiguously. Otherwise, the definition of the likelihood depends on the treatment of the diffuse quantities—$\boldsymbol{\delta}$, $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$. Francke, Koopman, and de Vos (2010) describe three variants of the likelihood function—*diffuse*-likelihood, *marginal*-likelihood, and *profile*-likelihood—that are commonly considered for state space models that have a diffuse initial condition. In the SSM procedure, you can either use diffuse likelihood ($\mathbf{L}_d(\mathbf{Y}, \boldsymbol{\theta})$) or marginal likelihood ($\mathbf{L}_m(\mathbf{Y}, \boldsymbol{\theta})$) for parameter estimation. Parameter estimation by marginal likelihood is a new experimental feature in this release. By default, diffuse likelihood is used for parameter estimation.

In terms of the quantities described in Table 33.5, diffuse, marginal, and profile likelihoods are defined as follows:

$$-2\log \mathbf{L}_d(\mathbf{Y}, \boldsymbol{\theta}) = N_0 \log 2\pi + \sum_{t=1}^{n}\sum_{i=1}^{q*p_t} (\log F_{t,i} + \frac{v_{t,i}^2}{F_{t,i}}) - \mathbf{b}_{n,p_n}^{'} \mathbf{S}_{n,p_n}^{-1} \mathbf{b}_{n,p_n} + \log(|\mathbf{S}_{n,p_n}|)$$

$$-2\log \mathbf{L}_m(\mathbf{Y}, \boldsymbol{\theta}) = N_0 \log 2\pi + \sum_{t=1}^{n}\sum_{i=1}^{q*p_t} (\log F_{t,i} + \frac{v_{t,i}^2}{F_{t,i}}) - \mathbf{b}_{n,p_n}^{'} \mathbf{S}_{n,p_n}^{-1} \mathbf{b}_{n,p_n} + \log(|\mathbf{S}_{n,p_n}|) - \log(|\mathbf{S}^*_{n,p_n}|)$$

$$-2\log \mathbf{L}_p(\mathbf{Y}, \boldsymbol{\theta}) = N \log 2\pi + \sum_{t=1}^{n}\sum_{i=1}^{q*p_t} (\log F_{t,i} + \frac{v_{t,i}^2}{F_{t,i}}) - \mathbf{b}_{n,p_n}^{'} \mathbf{S}_{n,p_n}^{-1} \mathbf{b}_{n,p_n}$$

In the preceding formulas, the terms that are associated with the missing response values $y_{t,i}$ are excluded and N denotes the total number of nonmissing response values in the sample. In addition, $N_0 = (N - k - g - d)$; $|\mathbf{S}_{n,p_n}|$ and $|\mathbf{S}^*_{n,p_n}|$ denote the determinants of $\mathbf{S}_{n,p_n}$ and $\mathbf{S}^*_{n,p_n}$, respectively; and $\mathbf{b}^{'}_{n,p_n}$ denotes the transpose of the column vector $\mathbf{b}_{n,p_n}$. If $\mathbf{S}_{n,p_n}$ is not invertible, then a generalized inverse is used in place of $\mathbf{S}^{-1}_{n,p_n}$, and $|\mathbf{S}_{n,p_n}|$ and $|\mathbf{S}^*_{n,p_n}|$ are computed based on the nonzero eigenvalues of $\mathbf{S}_{n,p_n}$ and $\mathbf{S}^*_{n,p_n}$, respectively. Moreover, in this case, $N_0 = N - \mathrm{Rank}(\mathbf{S}_{n,p_n})$. When $(d + k + g) = 0$, the terms that involve $\mathbf{S}_{n,p_n}$, $\mathbf{S}^*_{n,p_n}$, and $\mathbf{b}_{n,p_n}$ are absent.

The expression for marginal likelihood is derived by treating the diffuse quantities as fixed but unknown parameters. The expression can be shown to be based on a linear transformation of the N-dimensional response vector $\mathbf{Y}$. For a suitably chosen $N \times N$ matrix $\mathbf{H}$, let $\mathbf{U} = \mathbf{HY}$. $\mathbf{H}$ is chosen such that the

N-dimensional transformed vector $\mathbf{U}$ partitions into two uncorrelated (and independent because of their Gaussian nature) subvectors: an N_0-dimensional vector $\mathbf{U}_1$ and a $(d + k + g)$-dimensional vector $\mathbf{U}_2$. Furthermore, the distribution of $\mathbf{U}_1$ does not depend on the diffuse vectors ($\boldsymbol{\delta}$, $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$), and $\mathbf{U}_2$ stores the generalized least squares estimates of the diffuse vectors: $\mathbf{U}_2' = (\hat{\boldsymbol{\delta}}' \quad \hat{\boldsymbol{\beta}}' \quad \hat{\boldsymbol{\gamma}}')'$. It turns out that the N_0-dimensional vector, $\mathbf{U}_1$, has a proper Gaussian distribution and the marginal likelihood, $\log \mathbf{L}_m(\mathbf{Y}, \boldsymbol{\theta})$, is the proper likelihood of $\mathbf{U}_1$. The diffuse likelihood, $\log \mathbf{L}_d(\mathbf{Y}, \boldsymbol{\theta})$, is also based on $\mathbf{U}_1$. However, rather than assuming the diffuse quantities as unknown parameters, the expression of the diffuse likelihood is derived by assuming that $\boldsymbol{\delta}$, $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$ are random vectors with diffuse priors. Even though the marginal and diffuse likelihoods are based on different interpretations of the diffuse quantities, their expressions differ by only one term: the diffuse likelihood does not have the term $-\log(|\mathbf{S}^*_{n,p_n}|)$. Since the marginal likelihood is the proper likelihood of $\mathbf{U}_1$, the diffuse likelihood can be interpreted as a quasi-likelihood of $\mathbf{U}_1$. Apart from being essential to make it a proper likelihood, the extra term in the marginal likelihood plays another useful role: it makes the marginal likelihood invariant to linear rescaling of the diffuse effects, a desirable property in a likelihood. The diffuse likelihood is not invariant to linear rescaling of the diffuse effects. The profile likelihood, $\log \mathbf{L}_p(\mathbf{Y}, \boldsymbol{\theta})$, is the likelihood of the response vector $\mathbf{Y}$ evaluated at the generalized least squares estimates of the diffuse vectors: $(\boldsymbol{\delta}' \quad \boldsymbol{\beta}' \quad \boldsymbol{\gamma}')' = (\hat{\boldsymbol{\delta}}' \quad \hat{\boldsymbol{\beta}}' \quad \hat{\boldsymbol{\gamma}}')'$. It is derived by treating the diffuse quantities as fixed but unknown parameters and, like the marginal likelihood, is invariant to linear rescaling of the diffuse effects. For an illustration of this invariance property, see Example 33.18.

In the literature, the marginal likelihood (in addition to the diffuse likelihood) is also called the *restricted*-likelihood, and the estimate of the parameter vector $\boldsymbol{\theta}$ that is obtained by maximizing $\log \mathbf{L}_m(\mathbf{Y}, \boldsymbol{\theta})$ (or $\log \mathbf{L}_d(\mathbf{Y}, \boldsymbol{\theta})$) is called the restricted maximum likelihood estimate (REML). In this section, the REML estimate that is based on marginal likelihood is denoted by REML_M and the REML estimate that is based on diffuse likelihood is denoted by REML_D. In addition, the estimate of $\boldsymbol{\theta}$ that is obtained by maximizing the profile likelihood is called the maximum likelihood estimate (ML). In the absence of the diffuse quantities, all three likelihoods are the same and the REML_M, REML_D, and ML estimates coincide. When diffuse quantities are present, there is some evidence to prefer REML_M, the estimate of $\boldsymbol{\theta}$ that is based on the marginal likelihood. For more information, see Francke, Koopman, and de Vos (2010) and the references therein. By default, the SSM procedure uses diffuse likelihood for parameter estimation. You can switch to marginal likelihood by using the LIKE=MARGINAL option in the PROC SSM statement. In the current release of the SSM procedure, you cannot request parameter estimation by using profile likelihood.

Interestingly, for many types of state space models, REML_M and REML_D coincide even when diffuse effects are present. This is because for these models the extra term in the marginal likelihood, $-\log(|\mathbf{S}^*_{n,p_n}|)$, turns out to be independent of the parameter vector $\boldsymbol{\theta}$. Specifically, for models that are specified by using the PROC SSM syntax, REML_M and REML_D differ only if at least one of the following conditions hold:

- The transition matrix ($\mathbf{T}_t$) that is implied by a STATE statement depends on at least one unknown parameter and the diffuse dimension of the associated state subsection is nonzero.
- The list of variables that is specified in a COMPONENT statement depends on at least one unknown parameter and the diffuse dimension of the associated state subsection is nonzero.
- At least one lag term in a DEPLAG statement depends on an unknown parameter.
- A TREND statement with GROWTH or GROWTH(OU) type is present and the growth parameter, ϕ, is unknown.

In particular, if the parameter vector affects only the disturbance covariances (Q_t) in the state equation and the error variances ($\sigma^2_{t,i}$) in the observation equation (see Table 33.4), REML_M and REML_D coincide.

These observations also imply that REML_M and REML_D coincide for the most commonly used univariate and multivariate unobserved component models and for ARIMA models, with or without regression effects.

NOTE: For many examples in the section "Examples: SSM Procedure" on page 2462, one of the preceding conditions does hold and the REML_M and REML_D estimates do differ. However, in all these cases, it turns out that the differences in REML_M and REML_D are not large enough to change the overall conclusions of the analysis. As verification, you can rerun the analyses that are described in Example 33.10, Example 33.13, and Example 33.14 by using the LIKE=MARGINAL option in the PROC SSM statement. Of course, this will not be true in general.

The REML_D and REML_M estimates of the unknown parameter vector $\boldsymbol{\theta}$ (each denoted as $\hat{\boldsymbol{\theta}}$), are computed by maximizing the diffuse (or marginal) likelihood. This is done by using a nonlinear optimization process that involves repeated evaluations of $\mathbf{L}_d(\mathbf{Y}, \boldsymbol{\theta})$ (or $\mathbf{L}_m(\mathbf{Y}, \boldsymbol{\theta})$) at different values of $\boldsymbol{\theta}$. Approximate standard errors of $\hat{\boldsymbol{\theta}}$ are computed by taking the square root of the diagonal elements of its (approximate) covariance matrix. This covariance is computed as $-\mathbf{H}^{-1}$, where $\mathbf{H}$ is the Hessian (the matrix of the second-order partials) of $\log \mathbf{L}_d(\mathbf{Y}, \boldsymbol{\theta})$ (or $\log \mathbf{L}_m(\mathbf{Y}, \boldsymbol{\theta})$) evaluated at the optimum $\hat{\boldsymbol{\theta}}$. It is known that under mild regularity assumptions (as the number of distinct time points tends toward infinity), $\hat{\boldsymbol{\theta}}$ is consistent and efficient. For good discussions about REML_D, REML_M, and ML estimates, see Searle, Casella, and McCulloch (1992); Laird (2004); Francke, Koopman, and de Vos (2010).

If the marginal likelihood is used for parameter estimation, the SSM procedure reports the values of all three likelihoods at the parameter estimate $\hat{\boldsymbol{\theta}}$. Otherwise, PROC SSM reports the values of the diffuse and profile likelihoods that are calculated at the parameter estimate $\hat{\boldsymbol{\theta}}$. Let $\dim(\theta)$ denote the dimension of the parameter vector $\boldsymbol{\theta}$. After PROC SSM completes the parameter estimation, it prints the "Likelihood Computation Summary" table, which summarizes the likelihood calculations at $\hat{\boldsymbol{\theta}}$, as shown in Table 33.6.

Table 33.6 Likelihood Computation Summary

Quantity	Formula
Nonmissing response values used	N
Estimated parameters	$\dim(\theta)$
Initialized diffuse state elements	$\operatorname{rank}(\mathbf{S}_{n,p_n})$
Normalized residual sum of squares	$\sum_{t=1}^{n}\sum_{i=1}^{q*p_t}(\frac{v_{t,i}^2}{F_{t,i}}) - \mathbf{b}'_{n,p_n}\mathbf{S}^{-1}_{n,p_n}\mathbf{b}_{n,p_n}$
Diffuse log likelihood	$\log \mathbf{L}_d(\mathbf{Y}, \hat{\boldsymbol{\theta}})$
Marginal log likelihood	$\log \mathbf{L}_m(\mathbf{Y}, \hat{\boldsymbol{\theta}})$
Profile log likelihood	$\log \mathbf{L}_p(\mathbf{Y}, \hat{\boldsymbol{\theta}})$

In addition to the likelihood computation summary, PROC SSM also reports the information criteria that are based on the diffuse and profile likelihoods. It also reports the information criteria that are based on the marginal likelihood if marginal likelihood is used for parameter estimation. A variety of information criteria are reported. All these criteria are functions of twice the negative likelihood ($-2\log \mathbf{L}$, where the likelihood can be diffuse, marginal, or profile), N_* (the effective sample size), and $nparm$ (the effective number of model parameters). For information criteria that are based on the diffuse and marginal likelihoods, the effective sample size, N_*, is equal to N_0 and the effective number of model parameters, $nparm$, is equal to $\dim(\theta)$. For information criteria that are based on the profile likelihood, the effective sample size, N_*, is equal to N and the effective number of model parameters, $nparm$, is equal to $\dim(\theta) + d + k + g$. Table 33.7 summarizes the reported information criteria in smaller-is-better form.

Table 33.7 Information Criteria

Criterion	Formula	Reference
AIC	$-2\log \mathbf{L} + 2nparm$	Akaike (1974)
AICC	$-2\log \mathbf{L} + 2nparm N_*/(N_* - nparm - 1)$	Hurvich and Tsai (1989) Burnham and Anderson (1998)
HQIC	$-2\log \mathbf{L} + 2nparm \log\log(N_*)$	Hannan and Quinn (1979)
BIC	$-2\log \mathbf{L} + nparm \log(N_*)$	Schwarz (1978)
CAIC	$-2\log \mathbf{L} + nparm(\log(N_*) + 1)$	Bozdogan (1987)

Forecasting Phase

After the model-fitting phase, the filtering process is repeated again to produce the model-based one-step-ahead response variable forecasts ($\hat{y}_{t,i}$), residuals ($v_{t,i}$), and their standard errors ($\sqrt{F_{t,i}}$). In addition, one-step-ahead forecasts of the components that are specified in the MODEL statements, and any other user-defined linear combinations of $\boldsymbol{\alpha}_t$, are also produced. These forecasts are set to missing as long as the index $t < t_*$ (that is, until the filtering process is initialized). If the filtering process remains uninitialized, then all the quantities that are related to the one-step-ahead forecast (such as $\hat{y}_{t,i}$ and $v_{t,i}$) are reported as missing. When the fitted model is appropriate, the one-step-ahead residuals $v_{t,i}$ form a sequence of uncorrelated normal variates. This fact can be used during model diagnostic process.

Smoothing Phase

After the filtering phase of KFS produces the one-step-ahead predictions of the response variables and the underlying state vectors, the smoothing phase of KFS produces the full-sample versions of these quantities—that is, rather than using the history up to $(t, i-1)$, the entire sample $\mathbf{Y}$ is used. The smoothing phase of KFS is a backward algorithm, which begins at $t = n$ and $i = q * p_n$ and goes back toward $t = 1$ and $i = 1$. It produces the following quantities:

Table 33.8 KFS: Smoothing Phase

Quantity	Description
$\tilde{y}_{t,i} = \mathrm{E}(y_{t,i}\|\mathbf{Y})$	Interpolated response value
$\tilde{F}_{t,i} = \mathrm{Var}(y_{t,i}\|\mathbf{Y})$	Variance of the interpolated response value
$\tilde{\boldsymbol{\alpha}}_t = \mathrm{E}(\boldsymbol{\alpha}_t\|\mathbf{Y})$	Full-sample estimate of the state vector
$\tilde{\mathbf{P}}_t = \mathrm{Cov}(\boldsymbol{\alpha}_t\|\mathbf{Y})$	Covariance of $\tilde{\boldsymbol{\alpha}}_t$
$\left(\hat{\boldsymbol{\delta}}\ \hat{\boldsymbol{\beta}}\ \hat{\boldsymbol{\gamma}}\right)' = \mathbf{S}^{-1}_{n,p_n}\mathbf{b}_{n,p_n}$	Full-sample estimates of $\boldsymbol{\delta}$, $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$
$\mathbf{S}^{-1}_{n,p_n}$	Covariance of $\left(\hat{\boldsymbol{\delta}}\ \hat{\boldsymbol{\beta}}\ \hat{\boldsymbol{\gamma}}\right)'$

Note that if $y_{t,i}$ is not missing, then $\tilde{y}_{t,i} = \mathrm{E}(y_{t,i}|\mathbf{Y}) = y_{t,i}$ and $\tilde{F}_{t,i} = \mathrm{Var}(y_{t,i}|\mathbf{Y}) = 0$ because $y_{t,i}$ is completely known, given $\mathbf{Y}$. Therefore, $\tilde{y}_{t,i}$ provides nontrivial information only when $y_{t,i}$ is missing—in which case $\tilde{y}_{t,i}$ represents the best estimate of $y_{t,i}$ based on the available data. The full-sample estimates of components that are specified in the model equations are based on the corresponding linear combinations of $\tilde{\boldsymbol{\alpha}}_t$. Similarly, their standard errors are computed by using appropriate functions of $\tilde{\mathbf{P}}_t$.

If the filtering process remains uninitialized until the end of the sample (that is, if $\mathbf{S}_{n,p_n}$ is not invertible), some linear combinations of $\boldsymbol{\delta}$, $\boldsymbol{\beta}$, and $\boldsymbol{\gamma}$ are not estimable. This, in turn, implies that some linear combinations of $\boldsymbol{\alpha}_t$ are also inestimable. These inestimable quantities are reported as missing. For more information about the estimability of the state effects, see Selukar (2010).

Delete-One Cross Validation and Structural Breaks

In addition to the interpolation of missing response values and the full-sample estimation of components in the model, the smoothing phase can also produce several useful diagnostic measures that can indicate outlying observations and breaks in the state evolution process. The treatment of additive outliers and structural breaks that is described in this section is based on De Jong and Penzer (1998). Also see Selukar (2017) for illustrative examples.

Delete-One Cross Validation and the Additive Outlier Detection

Let $\mathrm{AO}_{t,i} = y_{t,i} - \mathrm{E}(y_{t,i}|\mathbf{Y}^{t,i})$ denote the difference between the observed response value $y_{t,i}$ and its estimate or prediction by using all the data except $y_{t,i}$, which is denoted by $\mathbf{Y}^{t,i}$. The smoothing phase of DKFS can generate $\mathrm{AO}_{t,i}$ (and its variance) at all (t,i). A large value of $\mathrm{AO}_{t,i}$ signifies that the observed response value $(y_{t,i})$ is unusual relative to the rest of the sample (according to the postulated model). Such values are called additive outliers. In the literature, $\mathrm{AO}_{t,i}$ are referred by a few different names. Sometimes they are called *delete-one cross validation errors* or simply *prediction errors*. In this chapter, these names are used interchangeably. Like the one-step-ahead residuals, $\nu_{t,i}$, the prediction errors can be used in checking the adequacy of the model. The prediction errors are normally distributed; however, unlike $\nu_{t,i}$, they are not serially uncorrelated. $\mathrm{AO}_{t,i}$ is set to missing when $y_{t,i}$ is missing. The SSM procedure prints a summary table of extreme additive outliers by default. In addition, you can request the plotting of the standardized prediction errors, and they can be output to a data set.

The prediction error sum of squares (PRESS)

$$\sum_{t,i} AO_{t,i}^2$$

can be a useful measure of fit to compare different models. It is also called the *cross validation error sum of squares*. An additional measure of fit based on the prediction errors is called the *generalized* cross validation error sum of squares (GCV). Denoting the variance of $\mathrm{AO}_{t,i}$ by $\mathrm{VAR\_AO}_{t,i}$, it is defined as

$$\frac{\sum_{t,i}(\mathrm{AO}_{t,i}^2/\mathrm{VAR\_AO}_{t,i}^2)}{[\,\sum_{t,i}(1/\mathrm{VAR\_AO}_{t,i})\,]^2}$$

You can request the printing of PRESS and GCV by specifying the PRESS option in the OUTPUT statement.

After inspecting the reported additive outliers, you can adjust the model to account for the effects of some of the extreme outlying observations. This can be done by including appropriate dummy variables in the observation equation.

Structural Breaks in the State Evolution

The additive outliers that are discussed in the preceding section are diagnostic measures associated with the measurement equation. The smoothing phase of DKFS can generate diagnostic measures that are also associated with the state equation.

For simplicity of notation and exposition, initially assume that the state equation has the following form:

$$\boldsymbol{\alpha}_{t+1} = \mathbf{T}_t\boldsymbol{\alpha}_t + \mathbf{c}_{t+1} + \boldsymbol{\eta}_{t+1}$$

That is, the state regression term $\mathbf{W}_{t+1}\boldsymbol{\gamma}$ is absent in the postulated model. Suppose that an unanticipated change of unknown size takes place in the i_0th element of the state at time $(t_0 + 1)$. The model can then be adjusted to account for this change by including a suitable dummy regressor in the state equation as follows:

$$\boldsymbol{\alpha}_{t+1} = \mathbf{T}_t\boldsymbol{\alpha}_t + \mathbf{W}_{t+1}\boldsymbol{\gamma} + \mathbf{c}_{t+1} + \boldsymbol{\eta}_{t+1}$$

Here $\mathbf{W}_t$ is a sequence of m-dimensional column vectors such that $\mathbf{W}_{t_0+1}[i_0] = 1$ and $\mathbf{W}_t[i] = 0$ for all other t and i. The estimate of the regression coefficient $\boldsymbol{\gamma}$ provides information about the size of the unanticipated change in the i_0th element of $\boldsymbol{\alpha}_t$ at time $t = t_0 + 1$. Similarly, an unanticipated change in a subsection of $\boldsymbol{\alpha}_t$ at a time $t = t_0 + 1$ can be estimated by using a set of appropriate dummies (the number of dummies equals the number of elements in the state subsection) in the state equation. The algorithm of De Jong and Penzer (1998) efficiently generates the estimates of such one-time changes in the state at all distinct time points in the sample in one smoothing pass. A statistically significant value of $\boldsymbol{\gamma}$ at a time point t_0 indicates an unanticipated change in the relevant element (or the subsection) of $\boldsymbol{\alpha}_{t_0}$. Note that the change associated with an additive outlier is temporary: the previous or the subsequent measurements are not affected. On the other hand, because of the evolutionary nature of the state equation, a one-time change in the state affects all the subsequent states, which in turn affect the subsequent observations. In this sense, a significant unanticipated change in the state is a *structural break*.

In the preceding discussion, the absence of the state regression variables in the postulated model was assumed only for notational simplicity. If the postulated model does contain some state regression variables, the dummy variable that is associated with the one-time state change is simply added to the existing set of state regression variables, and the interpretation of its regression coefficient as the measure of unanticipated change in the state remains unaffected.

In the SSM procedure, you can request the computation of significance statistics that are associated with one-time changes in the state subsections specified by using the STATE statement in addition to the state subsections that are associated with the components specified by using the TREND statements. This is done by using the CHECKBREAK option in these statements. In addition, you can request the computation of such statistics for the entire state $\boldsymbol{\alpha}_t$ by using the MAXSHOCK option in the OUTPUT statement. The significance statistics can be computed for both elementwise change and subsectionwise change. The computation of subsectionwise change statistics can be computationally expensive for large subsections (an inversion of a $p \times p$-dimensional matrix at each distinct time point in the sample is needed for the computation of significance statistics for a state subsection of size p). For an example of structural break analysis, see Example 33.8.

Estimation of User-Specified Linear Combination of State Elements

By default, the SSM procedure computes the estimates of all the components that are specified in the MODEL statements (you can print these estimates by using the PRINT= option in the respective TREND and COMPONENT statements, or you can output these estimates to a data set by specifying it in the OUT= option in the OUTPUT statement). However, in many cases it is desirable to obtain the estimates of additional linear combinations of the state elements and the regression effects. The SSM procedure provides two statements, the COMPONENT statement and the EVAL statement, that are useful for specifying virtually any desired linear combination of the elements of the state vector and the regression effects in the observation equation. After a desired linear combination is specified, you can print or output its estimate as you would for a component that is used in the MODEL statement. This feature of the SSM procedure is illustrated in many examples in the section "Examples: SSM Procedure" on page 2462. For example, in the second part of

Example 33.4, the COMPONENT and EVAL statements are used to define the contrasts between the growth profiles of cows that are receiving different treatments. Similarly, in Example 33.7, the EVAL statement is used to define the yield curve as a sum of the components that are used in the MODEL statement.

Contrasting PROC SSM with Other SAS Procedures

The SSM procedure complements several SAS/ETS procedures and the MIXED procedure in SAS/STAT software (see Chapter 81, "The MIXED Procedure" (*SAS/STAT User's Guide*)). The statistical models underlying all these procedures can be formulated as state space models; however, in many cases this formulation effort can be considerable. Generally speaking, when a problem can be formulated and satisfactorily solved either by using the SSM procedure or by using one of these other procedures, the other procedures are likely to be more efficient. However, in many instances, the SSM procedure can solve more general problems or offer more detailed analysis, or both. Throughout this discussion, it is assumed that the problem being solved can be modeled as a linear statistical model with Gaussian response variables. In particular, situations that require models such as autoregressive conditional heteroscedasticity (ARCH) models, and models with categorical response variables are not considered. The following list provides a more specific comparison of the SSM procedure with different procedures:

- All the SAS/ETS time series analysis procedures (the ARIMA, ESM, UCM, VARMAX, STATESPACE, and PANEL procedures) require time series data and are not applicable to the longitudinal data.
- For univariate time series analysis, the modeling facilities provided by the ARIMA, ESM, and UCM procedures are adequate in most cases. The SSM procedure can handle cases that do not fit neatly into one of these categories.
- For multivariate time series data analysis, you can use the VARMAX procedure for vector ARIMA modeling and the STATESPACE procedure for state space modeling. The capabilities of the SSM procedure are complementary to these procedures. In particular, the predefined multivariate structural models available in the SSM procedure cannot be specified by either of these procedures. In addition, you can formulate a much wider range of multivariate models—for example, models for series with different frequencies, by using the SSM procedure.
- When the $\mathbf{R}$ side effects are not too complicated (for example, if $\mathbf{R}$ is diagonal), the model considered by the MIXED procedure is a special case of the model considered by the SSM procedure. In the case of diagonal $\mathbf{R}$, it is easy to see that the state vector $\boldsymbol{\alpha}_t$ is equal to $\boldsymbol{\gamma}$, the MIXED random-effects vector, for all $t \geq 1$ (that is, $\boldsymbol{\alpha}_t$ is *time invariant*). Therefore, the random-effects MIXED model is obtained by setting $\mathbf{T} = \mathrm{Identity}$, $\mathbf{Q}_t = \mathbf{0}, t \geq 2$, $\mathbf{Q}_1 = \mathbf{G}$ (the MIXED $\mathbf{G}$ matrix), and $\mathbf{A}_1 = \mathbf{0}$.
- For the analysis of cross-sectional data, you can use the PANEL procedure. In this case, the SSM procedure capabilities are complementary. PROC SSM can provide alternate models, REML estimates, richer missing value support, and the estimates of the unobserved components (see the section "Getting Started: SSM Procedure" on page 2394 and the examples Example 33.2 and Example 33.11 for more information). In some situations the cross-sectional studies contain many panels but very few distinct time points. The PANEL procedure based analysis is better suited in such settings. In order for the analysis based on PROC SSM to be valid, the cross-sectional study must contain an adequate number of distinct time points.

Predefined Trend Models

The statistical models that govern the predefined trend components available in the SSM procedure are divided into two groups: models that are applicable to equally spaced data (possibly with replication), and models that are applicable more generally (the irregular data type). Each trend component can be described as a dot product $\mathbf{Z}\boldsymbol{\alpha}_t$ for some (time-invariant) vector $\mathbf{Z}$ and a state vector $\boldsymbol{\alpha}_t$. The component specification is complete after the vector $\mathbf{Z}$ is specified and the system matrices that govern the equations of $\boldsymbol{\alpha}_t$ are specified. For trend models for regular data, all the system matrices are time-invariant. For irregular data, $\mathbf{T}_t$ and $\mathbf{Q}_t$ depend on the spacing between the distinct time points: $(\tau_{t+1} - \tau_t)$.

Trend Models for Regular Data

These models are applicable when the data type is either regular or regular with replication. A good reference for these models is Harvey (1989).

Random Walk Trend

This model provides a trend pattern in which the level of the curve changes with time. The rapidity of this change is inversely proportional to the disturbance variance σ^2 that governs the underlying state. It can be described as $\mathbf{Z}\alpha_t$, where $Z = (1)$ and the (one-dimensional) state α_t follows a random walk:

$$\alpha_{t+1} = \alpha_t + \eta_{t+1}, \quad \eta_t \sim N(0, \sigma^2)$$

Here $\mathbf{T} = 1$ and $\mathbf{Q} = \sigma^2$. The initial condition is fully diffuse. Note that if $\sigma^2 = 0$, the resulting trend is a fixed constant.

Local Linear Trend

This model provides a trend pattern in which both the level and the slope of the curve change with time. This variation in the level and the slope is controlled by two parameters: σ_1^2 controls the level variation, and σ_2^2 controls the slope variation. If $\sigma_1^2 = 0$, the resulting trend is called an *integrated random walk*. If both $\sigma_1^2 = 0$ and $\sigma_2^2 = 0$, then the resulting model is the deterministic linear time trend. Here $\mathbf{Z} = (1\ 0)$, $\mathbf{T} = (1\ 1,\ 0\ 1)$, and $\mathbf{Q} = \mathrm{Diag}(\sigma_1^2,\ \sigma_2^2)$. The initial condition is fully diffuse.

Damped Local Linear Trend

This trend pattern is similar to the local linear trend pattern. However, in the DLL trend the slope follows a first-order autoregressive model, whereas in the LL trend the slope follows a random walk. The autoregressive parameter or the damping factor, ϕ, must lie between 0.0 and 1.0, which implies that the long-run forecast according to this pattern has a slope that tends to 0. Here $\mathbf{Z} = (1\ 0)$, $\mathbf{T} = (1\ 1,\ 0\ \phi)$, and $\mathbf{Q} = \mathrm{Diag}(\sigma_1^2,\ \sigma_2^2)$. The initial condition is partially diffuse with $\mathbf{Q_1} = \mathrm{Diag}(0,\ \sigma_2^2/(1 - \phi * \phi))$.

ARIMA Trend

This section describes the state space form for a component that follows an ARIMA$(p,d,q)\times(P,D,Q)_s$ model. The notation for ARIMA models is explained in the TREND statement.

First the state space form for the stationary case—that is, when $d = 0$ and $D = 0$, is explained. A number of alternate state space forms are possible in this case; the one described here is based on Jones (1980). With slight abuse of notation, let $p = p + s * P$ denote the effective autoregressive order, and let $q = q + sQ$ denote the effective moving average order of the model. Similarly, let ϕ be the effective autoregressive polynomial, and let θ be the effective moving average polynomial in the backshift operator with coefficients

$\phi_1, \ldots, \phi_p$ and $\theta_1, \ldots, \theta_q$, obtained by multiplying the respective nonseasonal and seasonal factors. Then, a random sequence ξ_t that follows an ARMA(p,q)×(P,Q)$_s$ model with a white noise sequence a_t has a state space form with state vector of size $m = \max(p, q+1)$. The system matrices are as follows: $\mathbf{Z} = [1\ 0\ \ldots\ 0]$, and the transition matrix $\mathbf{T}$, in a blocked form, is given by

$$\mathbf{T} = \begin{bmatrix} 0 & I_{m-1} \\ \phi_m\ \ldots & \phi_1 \end{bmatrix}$$

where $\phi_i = 0$ if $i > p$ and I_{m-1} is an $(m-1)$ dimensional identity matrix. The covariance of the state disturbance matrix $\mathbf{Q} = \sigma^2 \psi \psi'$, where σ^2 is the variance of the white noise sequence a_t and the vector $\psi = [\psi_0 \ldots \psi_{m-1}]'$ contains the first m values of the impulse response function—that is, the first m coefficients in the expansion of the ratio θ/ϕ. The covariance matrix of the initial state, $\mathbf{Q}_1$, is computed as

$$vec(\mathbf{Q}_1) = (\mathbf{I} - \mathbf{T} \bigotimes \mathbf{T})^{-1} vec(\mathbf{Q})$$

where $\bigotimes$ denotes the Kronecker product and the vec operation on a matrix creates a vector formed by vertically stacking the rows of that matrix.

A number of alternate state space forms are possible in the nonstationary case also. The form used by the SSM procedure utilizes the state space form for the stationary case as a building block. Suppose that a random sequence ξ_t follows an ARIMA(p,d,q)×(P,D,Q)$_s$ model with a white noise sequence a_t. As in the notation for the stationary case, with slight abuse of notation, let $d = d + s * D$ denote the effective differencing order, and let Δ be the effective differencing polynomial in the backshift operator with coefficients $\Delta_1, \ldots, \Delta_d$. It can be shown that ξ_t has a state space form with state vector size $m^\dagger = m + d$. In what follows, the system matrices and related quantities in the nonstationary case are described in terms of similar entities in the stationary case. A superscript dagger (†) has been added to distinguish the entities from the nonstationary case. $\mathbf{Z}^\dagger = [0\ 0\ \ldots\ 1\ \ldots\ 0]$ where the only nonzero value, 1, is at the index $m+1$, and the transition matrix, $\mathbf{T}^\dagger$, in a blocked form, is given by

$$\mathbf{T}^\dagger = \begin{bmatrix} \mathbf{T} & 0 & 0 \\ \mathbf{ZT} & \Delta_1 \ldots & \Delta_d \\ 0 & I_{d-1} & 0 \end{bmatrix}$$

The state disturbance matrix $\mathbf{Q}^\dagger$ is given by

$$\mathbf{Q}^\dagger = \begin{bmatrix} \mathbf{Q} & \mathbf{QZ}' & 0 \\ \mathbf{ZQ} & \mathbf{ZQZ}' & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Finally, the initial state is partially diffuse: the first m elements are nondiffuse and the last d elements are diffuse. The covariance matrix of the first m elements is $\mathbf{Q}_1$.

Trend Models for Irregular Data

A good reference for these models is De Jong and Mazzi (2001). Throughout this section $h_t = (\tau_{t+1} - \tau_t)$ denotes the difference between the successive time points. The system matrices $\mathbf{T}_t$ and $\mathbf{Q}_t$ that govern these models depend on h_t. However, whenever the notation is unambiguous, the subscript t is omitted.

Polynomial Spline Trend

This model is a general-purpose tool for extracting a smooth trend from the noisy data. The order of the spline governs the order of the local polynomial that defines the spline. The order-1 spline corresponds to Brownian motion (continuous-time random walk), the order-2 spline corresponds to integrated Brownian motion (continuous-time integrated random walk), and the order-3 spline provides a locally quadratic trend; the default order is 1. The dimension of the state underlying this component is the same as the order of the spline. The system matrices for the orders up to 3 are described as follows (in all the cases the initial condition is fully diffuse):

- order-1 spline: $\mathbf{Z} = (1)$, $\mathbf{T} = (1)$, and $\mathbf{Q} = \sigma^2(h)$
- order-2 spline: $\mathbf{Z} = (1\ 0)$, $\mathbf{T} = (1\ h,\ 0\ 1)$, and $\mathbf{Q} = \sigma^2 \begin{pmatrix} \frac{h^3}{3} & \frac{h^2}{2}, \frac{h^2}{2} & h \end{pmatrix}$
- order-3 spline: $\mathbf{Z} = (1\ 0\ 0)$, $\mathbf{T} = \begin{pmatrix} 1 & h & \frac{h^2}{2}, 0 & 1 & h, 0 & 0 & 1 \end{pmatrix}$, and

$$\mathbf{Q}[i,j] = \sigma^2 * \frac{h^{6-i-j+1}}{(6-i-j+1)(3-i)!(3-j)!} \quad 1 \le i,j \le 3$$

The system matrices for higher orders are similarly defined (for more information, see De Jong and Mazzi (2001)).

Note that, in addition to providing an estimate of the trend, this methodology can provide estimates of the higher-order derivatives of the trend. If $\boldsymbol{\alpha}$ denotes the k-dimensional subsection that is associated with a polynomial spline of order k, then its jth element ($1 \le j \le k$), $\boldsymbol{\alpha}[j]$, corresponds to the derivative of order $(j-1)$ of this polynomial spline. For an example of the estimation of the first derivative of a trend component, see Example 33.12. For additional information about using these types of trend patterns in data analysis, see Eubank, Huang, and Wang (2003); Kohn, Ansley, and Tharm (1991); Selukar (2015).

Decay and Growth Trends

There are two choices for the decay trend: DECAY and DECAY(OU). Similarly, there are two choices for the growth trend: GROWTH and GROWTH(OU). The "OU" stands for the Ornstein-Uhlenbeck form of these models. The decay trend is a sum of two correlated components: one component is a random walk, and the other component is a stationary autoregression. In its Ornstein-Uhlenbeck form, the random walk component is replaced by a constant. The growth trend (and its Ornstein-Uhlenbeck variant) has the same form as the decay trend except that the autoregression is nonstationary (in fact, it is explosive). For growth trend models, floating-point errors can result for even moderately long forecast horizons because of the explosive growth in the trend values.

The system matrices for the decay and the growth types in their respective cases are identical, except for the sign of the rate parameter ϕ: $\phi < 0.0$ for the decay type, and $\phi > 0.0$ for the growth type. In addition, the initial conditions for the growth models are fully diffuse; they are only partially diffuse for the decay models. The underlying state vector for all these models is two-dimensional.

The system matrices for the DECAY type are

$$\begin{aligned} \mathbf{Z} &= (1\ 1) \\ \mathbf{T} &= \mathrm{Diag}(1,\ \exp(h\phi)) \\ \mathbf{Q} &= \frac{\sigma^2}{\phi^3} \begin{pmatrix} h\phi & 1-\exp(h\phi), & 1-\exp(h\phi) & (\exp(2h\phi)-1)/2 \end{pmatrix} \end{aligned}$$

The initial condition is partially diffuse with $\mathbf{Q_1} = \text{Diag}(0,\ \frac{-\sigma^2}{2\phi^3})$. The system matrices for the GROWTH type are the same (with $\phi > 0.0$), except that the initial condition is fully diffuse; so $\mathbf{Q_1} = 0$.

For the DECAY(OU) type, $\mathbf{Z}$ and $\mathbf{T}$ are the same as DECAY, whereas

$$\mathbf{Q} = \text{Diag}\left(\ 0,\ \sigma^2\frac{(\exp(2h\phi)-1)}{2\phi}\ \right) \quad \text{and} \quad \mathbf{Q_1} = \text{Diag}(0,\ \frac{-\sigma^2}{2\phi})$$

The system matrices for the GROWTH(OU) type are the same (with $\phi > 0.0$), except that the initial condition is fully diffuse; so $\mathbf{Q_1} = 0$.

Predefined Structural Models

A set of predefined models is available in the SSM procedure for models called structural models in the time series literature. These predefined models can be used to model trend, seasonal, and cyclical patterns in the univariate and multivariate time series. For the most part, the multivariate models are straightforward generalizations of the corresponding univariate models—for example, the multivariate random walk trend described later in this section generalizes the univariate random walk trend that is described in the section "Random Walk Trend" on page 2442. All of these models, with the exception of the continuous-time cycle model, are applicable only to the regular data type. The continuous-time cycle model is applicable to all the data types; however, it is available for the univariate case only.

To specify these models, you must first use the STATE statement with the correct TYPE= option. When you specify the TYPE=option, you do not need to specify other options of the STATE statement (for example, the T option, the COV1 option, and the A1 option). However, you must specify the COV option, which describes the covariance of the disturbance term that drives the state equation. Throughout this section, the symmetric matrix specified by using the COV option is denoted by $\mathbf{\Sigma}$. For TYPE= LL, an additional matrix, specified by using the SLOPECOV suboption, also plays a role; it is denoted by $\mathbf{\Sigma}_{\text{slope}}$. Subsequently you must specify one or more COMPONENT statements to define the (univariate) components that are based on this state subsection for their inclusion in the MODEL statement. These univariate components exhibit interesting behavior based on the structure of $\mathbf{\Sigma}$ (and $\mathbf{\Sigma}_{\text{slope}}$, whenever applicable)—for example, imposing rank restrictions on $\mathbf{\Sigma}$ in the multivariate random walk results in these univariate trends moving together. For additional information about these models, see Harvey (1989).

The following example summarizes the steps needed to define a multivariate structural model by using a sequence of STATE and COMPONENT statements. For a full example, see Example 33.1. Suppose that a three-dimensional time series is being studied with response variables y1, y2, and y3. Suppose you want to specify the trivariate structural model

$$\mathbf{y}_t = \boldsymbol{\mu}_t + \boldsymbol{\psi}_t + \boldsymbol{\epsilon}_t$$

where $\mathbf{y}_t = (y_{1,t},\ y_{2,t},\ y_{3,t})$ denotes the response series, and $\boldsymbol{\mu}_t$, $\boldsymbol{\psi}_t$, and $\boldsymbol{\epsilon}_t$ denote the trivariate components, trend, cycle, and white noise, respectively. The three components of $\boldsymbol{\epsilon}_t$, the observation noise in the model, are not assumed to be independent. Therefore, you cannot specify them by using three IRREGULAR statements; you must include them in the state specification. The following (incomplete) statements show how to specify this model:

```
state whiteNoise(3)  type=wn ...;
component wn1 =  whiteNoise[1];
component wn2 =  whiteNoise[2];
component wn3 =  whiteNoise[3];

state randomWalk(3)  type=rw ...;
component rw1 =  randomWalk[1];
component rw2 =  randomWalk[2];
component rw3 =  randomWalk[3];

state cycleState(3)  type=cycle ...;
component c1 = cycleState[1];
component c2 = cycleState[2];
component c3 = cycleState[3];

model y1 = rw1 c1 wn1;
model y2 = rw2 c2 wn2;
model y3 = rw3 c3 wn3;
```

The first STATE statement defines whiteNoise, a state subsection that is needed for defining a three-dimensional white noise component. In turn, whiteNoise is used to define the three univariate white noise components: wn1, wn2, and wn3. The components wn1, wn2, and wn3 are correlated—their correlation structure is controlled by the covariance specification of whiteNoise. The second set of STATE and COMPONENT statements result in three correlated random walk trend components: rw1, rw2, and rw3. Finally, the last set of STATE and COMPONENT statements result in three correlated cycle components: c1, c2, and c3. In the end, the desired multivariate model is defined by including these (univariate) components in the appropriate MODEL statements.

In the preceding example, it is important to note the relationship between the nominal dimension (denoted by *dim* throughout this section) that is specified in the STATE statement and the actual dimension of the resulting state subsection. Note that the three state subsections, whiteNoise, randomWalk, and cycleState, are defined by using the same *dim* specification: 3. However, the actual dimensions of these state subsections depend on their type; they do not need to equal this specified dimension. Here, whiteNoise and randomWalk do have the same size, 3, as the specified *dim*. However, the size of cycleState, which is of TYPE=CYCLE, is $2 * dim = 6$. Another important point to note: no matter what the underlying size of the state subsection, the desired univariate components were obtained by using an identical specification scheme in the COMPONENT statement. This happens because the component specification style that is based on the element operator—[]—in the COMPONENT statement behaves differently when the TYPE= option is used to define the state subsection (for an illustration, see the section "Multivariate Season" on page 2448).

The system matrices for all these models are time-invariant, with the exception of the continuous-time cycle model. In this section, $\boldsymbol{\alpha}_t$ denotes the subsection of the overall model state $\boldsymbol{\alpha}_t$, and $\mathbf{T}$, $\mathbf{Q}$, and $\mathbf{A}_1$ denote the corresponding blocks of the larger system matrices.

For the multivariate cycle system matrices described in the section "Multivariate Cycle" on page 2448, the Kronecker product notation is useful: if $\mathbf{A}$ is an $m \times n$ matrix and $\mathbf{B}$ is a $p \times q$ matrix, then the Kronecker product $\mathbf{A} \bigotimes \mathbf{B}$ is an $mp \times nq$ block matrix:

$$\mathbf{A} \bigotimes \mathbf{B} = \begin{bmatrix} a_{11}B & \cdots & a_{1n}B \\ \vdots & \ddots & \vdots \\ a_{m1}B & \cdots & a_{mn}B \end{bmatrix}$$

Multivariate White Noise

The STATE statement option TYPE=WN specifies white noise of dimension *dim*—that is, a sequence of zero mean, independent, Gaussian vectors with covariance $\mathbf{\Sigma}$. The specification of the associated system matrices is trivial: $\mathbf{T}$ is zero, $\mathbf{Q} = \mathbf{\Sigma}$, and the initial condition is nondiffuse ($\mathbf{Q}_1 = \mathbf{\Sigma}$ and $\mathbf{A}_1 = 0$).

Multivariate white noise is needed to specify the observation equation noise term for the multivariate models for the time series data. Since the state space formulation for the SSM procedure requires the observation equation noise vector to have the diagonal form, you need to include the noise vector in the state. The noise term for the *i*th response variable is defined by a component that simply picks the *i*th element of this multivariate white noise. For example, the component wn_i defined as follows can be used as a noise term in the MODEL statement of the *i*th response variable:

```
state white(dim) type=wn ...;
component wn_3 = white[3];
```

Multivariate Random Walk Trend

The STATE statement option TYPE=RW specifies a *dim*-dimensional random walk

$$\boldsymbol{\alpha}_{t+1} = \boldsymbol{\alpha}_t + \boldsymbol{\eta}_{t+1}$$

where $\boldsymbol{\eta}_t$ is a sequence of zero mean, independent, Gaussian vectors with covariance $\mathbf{\Sigma}$. The specification of the associated system matrices is trivial: $\mathbf{T}$ is a *dim*-dimensional identity matrix, $\mathbf{I}_{dim}$, $\mathbf{Q} = \mathbf{\Sigma}$, and the initial condition is fully diffuse ($\mathbf{Q}_1 = 0$ and $\mathbf{A}_1 = \mathbf{I}_{dim}$).

The multivariate random walk is a useful trend model for multivariate time series data. The trend term for the *i*th response variable is defined by a component that simply picks the *i*th ($1 \leq i \leq dim$) element of $\boldsymbol{\alpha}_t$. For example, the component rw_i defined as follows can be used as a trend term in the MODEL statement of the *i*th response variable:

```
state randomWalk(3) type=rw ...;
component rw_2 = randomWalk[2];
```

Multivariate Local Linear Trend

The STATE statement option TYPE=LL specifies a *(2*dim)*-dimensional $\boldsymbol{\alpha}_t$, needed for defining a *dim*-dimensional local linear trend. The first *dim* elements of $\boldsymbol{\alpha}_t$ correspond to the needed multivariate trend, and the subsequent *dim* elements are needed to capture the slope vector of this trend. $\boldsymbol{\alpha}_t$ can be defined as

$$\boldsymbol{\alpha}_{t+1} = \mathbf{T}\boldsymbol{\alpha}_t + \boldsymbol{\eta}_{t+1}$$

where $\boldsymbol{\eta}_t$ is a sequence of zero mean, independent, Gaussian vectors with covariance $\mathrm{Diag}(\mathbf{\Sigma},\ \mathbf{\Sigma}_{\mathrm{slope}})$ and $\mathbf{T}$ is a 2**dim*-dimensional block matrix $\mathbf{T} = (\mathbf{I}_{dim}\ \mathbf{I}_{dim},\ \mathbf{0}\ \mathbf{I}_{dim})$. The initial condition is fully diffuse ($\mathbf{Q}_1 = 0$ and $\mathbf{A}_1 = \mathbf{I}_{2*dim}$). This is a multivariate generalization of the univariate local linear trend.

The multivariate local linear trend is a useful trend model for multivariate time series data. The trend term for the *i*th response variable is defined by a component that simply picks the *i*th element ($1 \leq i \leq dim$) of $\boldsymbol{\alpha}_t$. For example, the component ll_i defined as follows can be used as a trend term in the MODEL statement of the *i*th response variable:

```
state localLin(dim) type=ll(slopecov..) ...;
component ll_3 = localLin[3];
```

Multivariate Cycle

The STATE statement option TYPE=CYCLE specifies a *(2*dim)*-dimensional $\boldsymbol{\alpha}_t$, needed for defining a *dim*-dimensional cycle. As in the LL case, the first *dim* elements of $\boldsymbol{\alpha}_t$ correspond to the needed *dim*-dimensional cycle, and the remaining *dim* elements contain some auxiliary quantities. The cycle model defined in this subsection requires a regular data type—that is, the CT option is not included. Let ρ denote the damping factor, and let $\lambda = 2\pi/\mathit{period}$ be the frequency associated with the cycle. The admissible parameter ranges are $0 < \rho \leq 1$ and $\mathit{period} > 2$, which implies that $0 < \lambda < \pi$. Let $\mathbf{C} = \rho(\cos(\lambda)\ \sin(\lambda),\ -\sin(\lambda)\ \cos(\lambda))$, a 2×2 matrix, and let $\mathbf{T} = \mathbf{C} \bigotimes \mathbf{I}_{dim}$, a $2 * dim \times 2 * dim$ matrix. With this notation, the transition equation associated with $\boldsymbol{\alpha}_t$ is

$$\boldsymbol{\alpha}_{t+1} = \mathbf{T}\boldsymbol{\alpha}_t + \boldsymbol{\eta}_{t+1}$$

where $\boldsymbol{\eta}_t$ is a sequence of zero mean, independent, $(2 * dim)$-dimensional Gaussian vectors with covariance $\mathrm{Diag}(\boldsymbol{\Sigma},\ \boldsymbol{\Sigma})$. If $\rho = 1$, the initial condition is fully diffuse ($\mathbf{Q}_1 = 0$ and $\mathbf{A}_1 = \mathbf{I}_{2*dim}$). Otherwise, it is nondiffuse: $\mathbf{Q}_1 = \frac{1}{(1-\rho^2)}\mathrm{Diag}(\boldsymbol{\Sigma},\ \boldsymbol{\Sigma})$ and $\mathbf{A}_1 = 0$.

The multivariate cycle is useful for capturing periodic behavior for multivariate time series data. The cycle term for the *i*th response variable is defined by a component that simply picks the *i*th element of $\boldsymbol{\alpha}_t$. For example, the component cycle_i defined as follows can be used as a cycle term in the MODEL statement of the *i*th response variable:

```
state cycleState(dim) type=cycle  ...;
component cycle_2 = cycleState[2];
```

Multivariate Season

The STATE statement option TYPE=SEASON(LENGTH=*s*) specifies a *((s–1)*dim)*-dimensional $\boldsymbol{\alpha}_t$, needed for defining a *dim*-dimensional trigonometric season component with season length *s*. A (multivariate) trigonometric season component, $\boldsymbol{\zeta}$, is a sum of (multivariate) cycles of different frequencies,

$$\boldsymbol{\zeta} = \sum_{j=1}^{[s/2]} \boldsymbol{\zeta}_j$$

where the constituent cycles $\boldsymbol{\zeta}_j$, called harmonics, have frequencies $\lambda_j = 2\pi j/s$. All the harmonics are assumed to be statistically independent, have the same damping factor $\rho = 1$, and are governed by the disturbances with the same covariance matrix $\boldsymbol{\Sigma}$. The number of harmonics, $[s/2]$, equals $s/2$ if s is even and $(s-1)/2$ if it is odd. This means that specifying TYPE=SEASON(LENGTH=*s*) is equivalent to specifying $[s/2]$ cycle specifications with correct frequencies, damping factor $\rho = 1$, and the COV option restricted to the same covariance $\boldsymbol{\Sigma}$. The resulting $\boldsymbol{\alpha}_t$ is necessarily *((s–1)*dim)*-dimensional. When the season length *s* is even, the last harmonic cycle, $\boldsymbol{\zeta}_{s/2}$, has frequency π and requires special attention. It is of dimension *dim* rather than *2*dim* because its underlying state equation simplifies to a *dim*-variate autoregression with autoregression coefficient $-\mathbf{I}_{dim}$. As a result of this discussion, it is clear that the system matrices $\mathbf{T}$ and $\mathbf{Q}$ associated with the *((s–1)*dim)*-dimensional $\boldsymbol{\alpha}_t$ are block-diagonal with the blocks corresponding to the harmonics. The initial condition is fully diffuse.

For all the models discussed so far, the first *dim* elements of $\boldsymbol{\alpha}_t$ provided the needed (multivariate) component. This is not the case for the (multivariate) season component. Extracting the *i*th seasonal component from $\boldsymbol{\alpha}_t$ requires accumulating the contributions from the $[s/2]$ harmonics that are associated with this *i*th seasonal, which are not organized contiguously in $\boldsymbol{\alpha}_t$. For example, suppose that *dim* is 2 and the season length s is 4. In this case $[s/2]$ is 2, and the bivariate seasonal component is a sum of two independent bivariate cycles, $\boldsymbol{\zeta}_1$ and $\boldsymbol{\zeta}_2$. The cycle $\boldsymbol{\zeta}_1$ has frequency $\pi/2$ and its underlying state, say $\boldsymbol{\alpha}_t^a$, has dimension $2 * dim = 4$. The last harmonic, $\boldsymbol{\zeta}_2$, has frequency π, and therefore its underlying state, say $\boldsymbol{\alpha}_t^b$, has dimension 2. The combined state $\boldsymbol{\alpha}_t = (\boldsymbol{\alpha}_t^a, \boldsymbol{\alpha}_t^b)$ has dimension $6 = 4 + 2$. In order to extract the first bivariate seasonal component, you must extract the first components of bivariate cycles $\boldsymbol{\zeta}_1$ and $\boldsymbol{\zeta}_2$, which in turn implies the first elements of $\boldsymbol{\alpha}_t^a$ and $\boldsymbol{\alpha}_t^b$, respectively. Thus, obtaining the first bivariate seasonal component requires extracting the first and the fifth elements of the combined state $\boldsymbol{\alpha}_t$. Similarly, obtaining the second bivariate seasonal component requires extracting the second and the sixth elements of the combined state $\boldsymbol{\alpha}_t$. All this can be summarized by the dot product expressions

$$\begin{aligned} s_{1t} &= (1\,0\,0\,0\,1\,0)\,\boldsymbol{\alpha}_t \\ s_{2t} &= (0\,1\,0\,0\,0\,1)\,\boldsymbol{\alpha}_t \end{aligned}$$

where s_{1t} and s_{2t} denote the first and second components, respectively, of the bivariate seasonal component. Note that s_{1t} and s_{2t} are univariate seasonal components, each of season length 4, in their own right. They are correlated components; their correlation structure depends on $\boldsymbol{\Sigma}$.

Obtaining the desired components of the multivariate seasonal component is made easy by a special syntax convention of the COMPONENT statement. Continuing with the previous example, the following examples illustrate two equivalent ways of obtaining s_{1t} and s_{2t}. The first set of statements explicitly specify the linear combinations needed for defining s_{1t} and s_{2t}:

```
state seasonState(2) type=season(length=4)  ...;
component s_1 =( 1  0  0  0  1  0 ) * seasonState;
component s_2 =( 0  1  0  0  0  1 ) * seasonState;
```

The following simpler specification achieves the same result:

```
state seasonState(2) type=season(length=4)  ...;
component s_1 = seasonState[1];
component s_2 = seasonState[2];
```

In the latter specification, the meaning of the element operator [] changes if the state in question is defined by using the TYPE= option.

Multivariate ARMA

You can specify a state vector that follows a multivariate autoregressive, moving average (VARMA) model by using the STATE statement option TYPE=VARMA. The autoregressive and moving average orders can be either 0 or 1 ($0 \le p \le 1$ and $0 \le q \le 1$)—that is, only VAR(1), MA(1), and VARMA(1,1) models can be specified. The notation and the state space form of the VARMA model described here is taken from Reinsel (1997), which is a good reference for VARMA modeling.

A *dim*-dimensional vector process $\boldsymbol{\zeta}_t$ follows a zero-mean, autoregressive order *p*, moving average order *q* (VARMA(*p*, *q*)) model if it satisfies the following matrix difference equation:

$$\boldsymbol{\zeta}_t = \sum_{i=1}^{p} \boldsymbol{\Phi}_i \boldsymbol{\zeta}_{t-i} + \boldsymbol{\epsilon}_t - \sum_{j=1}^{q} \boldsymbol{\Theta}_j \boldsymbol{\epsilon}_{t-j}$$

Here $\mathbf{\Phi}_i$ and $\mathbf{\Theta}_j$ are *dim*-dimensional square matrices and $\boldsymbol{\epsilon}_t$ is a *dim*-dimensional, Gaussian, white noise sequence with covariance matrix $\mathbf{\Sigma}$. If autoregressive order p is 0, the term that involves $\mathbf{\Phi}_i$ is absent; similarly, if the moving average order q is 0, the term that involves $\mathbf{\Theta}_j$ is absent. Since AR and MA orders can be at most 1, the subscripts of $\mathbf{\Phi}_i$ and $\mathbf{\Theta}_j$ can be ignored in this discussion—when applicable, an AR coefficient matrix is denoted by $\mathbf{\Phi}$ and an MA coefficient matrix is denoted by $\mathbf{\Theta}$. The unknown elements of $\mathbf{\Phi}$, $\mathbf{\Theta}$, and $\mathbf{\Sigma}$ constitute the parameter vector that is associated with a VARMA state. The process $\boldsymbol{\zeta}_t$ defined by the VARMA difference equation is stationary and invertible (Reinsel 1997) if and only if the eigenvalues of $\mathbf{\Phi}$ and $\mathbf{\Theta}$ are strictly less than 1 in magnitude. By default, the SSM procedure imposes these stationarity and invertibility restrictions on $\mathbf{\Phi}$ and $\mathbf{\Theta}$. However, you can specify $\mathbf{\Phi}$ to be an identity matrix, in which case the resulting process is nonstationary.

A VARMA model can be cast into a state space form. The state space form used by the SSM procedure is described in Reinsel (1997, pp. 52–53). The system matrices for the supported VARMA models are as follows:

- The VAR(1) form is the simplest. In this case, the underlying state $\boldsymbol{\alpha}_t$ is the same as the VAR(1) process $\boldsymbol{\zeta}_t$. Therefore, $\mathbf{T} = \mathbf{\Phi}$ and $\mathbf{Q_t} = \mathbf{\Sigma}$.
- Taking $\mathbf{\Phi}$ equal to the zero matrix if $p = 0$, the VARMA(1,1) and MA(1) cases can be treated together. In this case, the underlying state $\boldsymbol{\alpha}_t$ is *2*dim* dimensional and the desired VARMA process $\boldsymbol{\zeta}_t$ corresponds to its first *dim* elements. Let $\mathbf{\Psi} = \mathbf{\Phi} - \mathbf{\Theta}$. Then, in the blocked form,

$$\mathbf{T} = \begin{bmatrix} \mathbf{0} & \mathbf{I}_{dim} \\ \mathbf{0} & \mathbf{\Phi} \end{bmatrix} \quad \text{and} \quad \mathbf{Q}_t = \mathbf{Q} = \begin{bmatrix} \mathbf{\Sigma} & \mathbf{\Sigma\Psi}' \\ \mathbf{\Psi\Sigma} & \mathbf{\Psi\Sigma\Psi}' \end{bmatrix}$$

Unless $\mathbf{\Phi}$ is restricted to be identity, the underlying state $\boldsymbol{\alpha}_t$ is stationary and the covariance of the initial condition is computed by

$$vec(\mathbf{Q}_1) = (\mathbf{I} - \mathbf{T} \bigotimes \mathbf{T})^{-1} vec(\mathbf{Q})$$

where $\bigotimes$ denotes the Kronecker product and the vec operation on a matrix creates a vector formed by vertically stacking the rows of that matrix. If $\mathbf{\Phi}$ is restricted to be identity, the initial condition is fully diffuse.

Continuous-Time Cycle

The STATE statement option TYPE=CYCLE(CT) specifies a two-dimensional $\boldsymbol{\alpha}_t$, needed for defining a univariate continuous time cycle. In this case the nominal dimension, *dim*, must be 1. In particular, $\mathbf{\Sigma}$ becomes one-dimensional, which is denoted by σ^2. This cycle can be used for any data type. As before, the parameters of the cycle are a damping factor ρ, $0 < \rho \leq 1$, and *period* > 0. Unlike in the discrete-time cycle described in the section "Multivariate Cycle" on page 2448, the *period* is not required to be larger than 2. Let $\lambda = 2\pi/$*period*, and let $h_t = (\tau_{t+1} - \tau_t)$ denote the difference between successive time points. In this case, the system matrices $\mathbf{T}$ and $\mathbf{Q}$ that govern $\boldsymbol{\alpha}_t$ depend on h_t. They are as follows:

$$\begin{aligned} \mathbf{T} &= \rho^h \left(\cos(\lambda h)\ \sin(\lambda h),\ -\sin(\lambda h)\ \cos(\lambda h)\right) \\ \mathbf{Q} &= \frac{\sigma^2(1-\rho^{2h})}{-2\ln(\rho)} * \mathbf{I}_2 \qquad \text{if } \rho < 1 \\ \mathbf{Q} &= \sigma^2 h \mathbf{I}_2 \qquad \text{if } \rho = 1 \end{aligned}$$

If $\rho < 1$, the initial condition is nondiffuse: $\mathbf{Q}_1 = \frac{\sigma^2}{-2\ln(\rho)}\mathbf{I}_2$. For $\rho = 1$, the initial condition is fully diffuse.

The first element of $\boldsymbol{\alpha}_t$ corresponds to the needed cycle, and the second element is an auxiliary quantity. You can define a cycle term based on this state as follows:

```
state cycleState(1) type=cycle(CT)  ...;
component cycle = cycleState[1];
```

The CT option must be included in the use of TYPE=CYCLE.

Models with Dependent Lags

Many useful time series models relate the present value of a response variable to its own lagged values and, in the multivariate case, the lagged values of other response variables in the model. In the SSM procedure, you can use the DEPLAG statement to specify the terms in the model that involve lagged response variables. These models apply only to the regular data type. This section describes the state space form of such models; for more information, see Harvey (1989, sec. 7.1.1). As an illustration, consider the following model, where the q-dimensional coefficient matrices $\boldsymbol{\Phi}_1$ and $\boldsymbol{\Phi}_2$ are either fully or partially known:

$$\begin{aligned}
\mathbf{Y}_t &= \boldsymbol{\Phi}_1\mathbf{Y}_{t-1} + \boldsymbol{\Phi}_2\mathbf{Y}_{t-2} + \mathbf{Z}_t\boldsymbol{\alpha}_t + \mathbf{X}_t\boldsymbol{\beta} + \boldsymbol{\epsilon}_t \\
\boldsymbol{\alpha}_{t+1} &= \mathbf{T}_t\boldsymbol{\alpha}_t + \mathbf{W}_{t+1}\boldsymbol{\gamma} + \mathbf{c}_{t+1} + \boldsymbol{\eta}_{t+1} \\
\boldsymbol{\alpha}_1 &= \mathbf{c}_1 + \mathbf{A}_1\boldsymbol{\delta} + \boldsymbol{\eta}_1
\end{aligned}$$

Except for the presence of the terms that involve lagged response vectors ($\boldsymbol{\Phi}_1\mathbf{Y}_{t-1}$ and $\boldsymbol{\Phi}_2\mathbf{Y}_{t-2}$) in the observation equation, the form of this model is the same as the standard state space form that is described in the section "State Space Model and Notation" on page 2426. It turns out that this model can be expressed in the standard state space form by suitably enlarging the latent vectors in the state equation and by appropriately reorganizing the system matrices. The enlarged latent vectors and the corresponding system matrices are distinguished by the presence of dagger (†) as a superscript in the following reformulated model,

$$\begin{aligned}
\mathbf{Y}_t &= \mathbf{Z}_t^{\dagger}\boldsymbol{\alpha}_t^{\dagger} \\
\boldsymbol{\alpha}_{t+1}^{\dagger} &= \mathbf{T}_t^{\dagger}\boldsymbol{\alpha}_t^{\dagger} + \mathbf{W}_{t+1}^{\dagger}\boldsymbol{\gamma}^{\dagger} + \mathbf{c}_{t+1}^{\dagger} + \boldsymbol{\eta}_{t+1}^{\dagger} \\
\boldsymbol{\alpha}_1^{\dagger} &= \mathbf{c}_1^{\dagger} + \mathbf{A}_1^{\dagger}\boldsymbol{\delta}^{\dagger} + \boldsymbol{\eta}_1^{\dagger}
\end{aligned}$$

where the following conditions are true (column vectors are displayed horizontally to save space):

- The enlarged state vector ($\boldsymbol{\alpha}_t^{\dagger}$) is formed by vertically stacking the old state vector ($\boldsymbol{\alpha}_t$), the observation disturbance vector ($\boldsymbol{\epsilon}_t$), and the present and lagged response vectors ($\mathbf{Y}_t$ and $\mathbf{Y}_{t-1}$, respectively). That is, $\boldsymbol{\alpha}_t^{\dagger} = [\boldsymbol{\alpha}_t \;\; \boldsymbol{\epsilon}_t \;\; \mathbf{Y}_t \;\; \mathbf{Y}_{t-1}]$. Because $\boldsymbol{\alpha}_t$ is m-dimensional and $\boldsymbol{\epsilon}_t$, $\mathbf{Y}_t$, and $\mathbf{Y}_{t-1}$ are q-dimensional, the dimension of $\boldsymbol{\alpha}_t^{\dagger}$ is $m^{\dagger} = (m + 3 * q)$.

- The new state regression vector ($\boldsymbol{\gamma}^{\dagger}$) is formed by vertically stacking the old state regression vector ($\boldsymbol{\gamma}$) and the observation equation regression vector ($\boldsymbol{\beta}$). That is, $\boldsymbol{\gamma}^{\dagger} = [\boldsymbol{\gamma} \;\; \boldsymbol{\beta}]$.

- The enlarged disturbance vector ($\boldsymbol{\eta}_t^{\dagger}$) is formed by vertically stacking the old state disturbance vector ($\boldsymbol{\eta}_t$), the observation disturbance vector ($\boldsymbol{\epsilon}_t$), the vector sum ($\mathbf{Z}_t\boldsymbol{\eta}_t + \boldsymbol{\epsilon}_t$), and filling the rest of the vector with zeros. That is, $\boldsymbol{\eta}_t^{\dagger} = [\boldsymbol{\eta}_t \;\; \boldsymbol{\epsilon}_t \;\; (\mathbf{Z}_t\boldsymbol{\eta}_t + \boldsymbol{\epsilon}_t) \;\; \mathbf{0}]$.

- The deterministic vector $\mathbf{c}^{\dagger}_{t+1} = [\mathbf{c}_{t+1} \;\; \mathbf{0} \;\; \mathbf{Z}_{t+1}\mathbf{c}_{t+1} \;\; \mathbf{0}]$.
- The last $2q$ elements of the initial state vector ($\boldsymbol{\alpha}^{\dagger}_1$), which correspond to $\mathbf{Y}_1$, and $\mathbf{Y}_0$, are taken to be diffuse (which means that the diffuse vector $\boldsymbol{\delta}^{\dagger}$ has $2q$ additional elements compared to $\boldsymbol{\delta}$).

The new system matrices can be described in blockwise form in terms of the old system matrices as follows:

- The $q \times (m + 3 * q)$-dimensional $\mathbf{Z}^{\dagger}_t = [\mathbf{0} \; \mathbf{0} \; \mathbf{I} \; \mathbf{0}]$, where $\mathbf{0}$ is either a $q \times m$-dimensional or $q \times q$-dimensional matrix of zeros and $\mathbf{I}$ is a q-dimensional identity matrix.
- The $m^{\dagger} \times m^{\dagger}$ matrices $\mathbf{T}^{\dagger}_t$ (transition matrix) and $\mathbf{Q}^{\dagger}_t$ (covariance of $\boldsymbol{\eta}^{\dagger}_{t+1}$) are

$$\mathbf{T}^{\dagger}_t = \begin{bmatrix} \mathbf{T}_t & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{Z}_{t+1}\mathbf{T}_t & \mathbf{0} & \boldsymbol{\Phi}_1 & \boldsymbol{\Phi}_2 \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \end{bmatrix} \quad \text{and} \quad \mathbf{Q}^{\dagger}_t = \begin{bmatrix} \mathbf{Q}_t & \mathbf{0} & \mathbf{Q}_t\mathbf{Z}'_{t+1} & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_{t+1} & \boldsymbol{\Sigma}_{t+1} & \mathbf{0} \\ \mathbf{Z}_{t+1}\mathbf{Q}_t & \boldsymbol{\Sigma}_{t+1} & (\mathbf{Z}_{t+1}\mathbf{Q}_t\mathbf{Z}'_{t+1} + \boldsymbol{\Sigma}_{t+1}) & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}$$

  where $\boldsymbol{\Sigma}_t$ denotes the covariance matrix (which is diagonal by design) of the observation error vector $\boldsymbol{\epsilon}_t$. Recall that the system matrices in the transition equation can depend on both t and $t + 1$ even if the subscripts of $\mathbf{T}$ and $\mathbf{Q}$ show dependence on t alone.
- The $m^{\dagger} \times (k + g)$ matrix $\mathbf{W}^{\dagger}_t$ is

$$\mathbf{W}^{\dagger}_{t+1} = \begin{bmatrix} \mathbf{W}_{t+1} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \\ \mathbf{Z}_{t+1}\mathbf{W}_{t+1} & \mathbf{X}_{t+1} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}$$

This state space form can be easily extended to account for higher-order lags.

Models that contain dependent lag terms must be used with care. Because the SSM procedure does not impose any special constraints on the lag coefficients (the elements of coefficient matrices $\boldsymbol{\Phi}_1, \boldsymbol{\Phi}_2$, and so on), the resulting models can often be explosive. For an example of a model with lagged response variables, see Example 33.13.

PROC SSM and PROC UCM (see Chapter 41, "The UCM Procedure") handle models that contain dependent lags in essentially the same way. However, there is one difference: if the model parameter vector contains unknown lag parameters, PROC UCM parameters are estimated by optimizing the nondiffuse part of the likelihood, whereas PROC SSM continues to use the full diffuse likelihood for parameter estimation.

Temporal Aggregation and Temporal Distribution (Experimental)

The response variables in time series analysis are often classified as either stock variables or flow variables. Stock variables, such as interest rates or temperatures, are measured at a particular point in time. Flow variables, such as monthly income or weekly sales, are defined with respect to an interval of time. Flow variables have the property that they remain meaningful under the operations of temporal aggregation and temporal distribution—for example, aggregation of daily sales to weekly sales and distribution (or disaggregation) of weekly sales to daily sales are quite natural, whereas the same cannot be said of temperature

readings. This section explains how you can use the SSM procedure to do model-based temporal aggregation and distribution of flow variables. State space models are often used to carry out model-based temporal aggregation and distribution. Two properties of state space models make them particularly suitable for this purpose:

- If a variable is modeled by a state space model at a particular time interval, its aggregated form—for example, daily to monthly—also follows a state space model. Moreover, the state space forms of these two models have a simple relationship.
- State space models can easily handle missing response values.

The discussion in this section, which is based on Harvey (1989, chap. 6, sec. 3), is limited to regular data types—that is, the data must be either univariate or multivariate time series.

Temporal Distribution

For the sake of simplicity, consider a simple case of distributing weekly observations of a flow variable, y, at a daily interval. Even though the values of y are observed weekly (suppose they are recorded each Sunday), in this case it is necessary to treat the observations $y_t, t \geq 1$, as a daily time series such that y_t equals the weekly total when t corresponds to the end of the week (Sunday), and y_t is missing on other days of the week. In addition, suppose that $y_t^\dagger$ denotes the unobserved time series of daily values of y. In other words, if t corresponds to a Sunday, then

$$y_t = \sum_{s=t-6}^{t} y_s^\dagger$$

Suppose that the unobserved daily series $y_t^\dagger$ can be modeled by a state space model. For example, suppose the model for $y_t^\dagger$ is

$$\begin{aligned} y_t^\dagger &= \mathbf{Z}_t \boldsymbol{\alpha}_t + \epsilon_t \\ \boldsymbol{\alpha}_{t+1} &= \mathbf{T}_t \boldsymbol{\alpha}_t + \boldsymbol{\eta}_{t+1} \end{aligned}$$

Then it is easy to see that the aggregated series y_t follows a state space model of the form

$$\begin{aligned} y_t &= \mathbf{Z}_t^\dagger \boldsymbol{\alpha}_t^\dagger \\ \boldsymbol{\alpha}_{t+1}^\dagger &= \mathbf{T}_t^\dagger \boldsymbol{\alpha}_t^\dagger + \boldsymbol{\eta}_{t+1}^\dagger \end{aligned}$$

where the following are true (both the row and column vectors are displayed horizontally to save space):

- The new state vector ($\boldsymbol{\alpha}_t^\dagger$) is formed by augmenting the old state vector ($\boldsymbol{\alpha}_t$) with a latent variable, y_t^f. That is, $\boldsymbol{\alpha}_t^\dagger = [\boldsymbol{\alpha}_t \; y_t^f]$. In fact, y_t^f represents the within-week running total of $y_t^\dagger$, so that when t corresponds to a Sunday, $y_t^f = y_t$.
- The new transition matrix $\mathbf{T}_t^\dagger$ is

$$\mathbf{T}_t^\dagger = \begin{bmatrix} \mathbf{T}_t & \mathbf{0} \\ \mathbf{Z}_{t+1}\mathbf{T}_t & \psi_{t+1} \end{bmatrix}$$

where ψ_t is a dummy variable that equals 1 when t is not the start of the week (not Monday) and equals 0 when t is the start of the week (Monday).

- The new disturbance vector ($\boldsymbol{\eta}_t^{\dagger}$) is formed by augmenting the old disturbance vector ($\boldsymbol{\eta}_t$) by ($\mathbf{Z}_t\boldsymbol{\eta}_t + \epsilon_t$). That is, $\boldsymbol{\eta}_t^{\dagger} = [\boldsymbol{\eta}_t \;\; \mathbf{Z}_t\boldsymbol{\eta}_t + \epsilon_t]$.
- The new design matrix for the state effect ($\mathbf{Z}_t^{\dagger}$) is $\mathbf{Z}_t^{\dagger} = [\mathbf{0}\ 1]$, where $\mathbf{0}$ is a zero vector of the same size as the old state vector $\boldsymbol{\alpha}_t$.

This shows that you can do model-based distribution of y values by carrying out the following steps:

1. Organize the y values as a daily time series.
2. Define a dummy variable, startWeek, that flags the start of the week—that is, startWeek is 1 when the day is Monday and 0 otherwise. Note that $\psi_t = 1 - \text{startWeek}_t$.
3. Specify a suitable state space model for the unobserved daily series $y_t^{\dagger}$. This specification in turn implies a state space model specification for y.
4. Carry out the analysis—model fitting, component estimation, and forecasting—of y in the usual fashion by using this implied model specification.
5. The smoothed values of y from the previous step provide the estimates of y_t^f. In addition, the estimates of $y_t^{\dagger}$ can be obtained as the smoothed estimates of appropriate linear combination of the elements of $\boldsymbol{\alpha}_t$ and ϵ_t.

The SSM procedure enables you to carry out the key steps—Step 3 to Step 5—in this process quite easily. The usual model specification syntax that uses the STATE, COMPONENT, and TREND statements to define the terms in a MODEL statement is used to define a model for the unobserved daily series $y_t^{\dagger}$ (the first part of Step 3). Then, the use of the DISTRIBUTE(START=*startWeek*) option in the MODEL statement causes the SSM procedure to use the implied model to analyze the observed y values. As a brief illustration, suppose that a data set Test contains two variables: date, a SAS date variable that indexes the daily observations, and y, the values of the weekly variable arranged as a daily series. Then the following PROC SSM statements show you how to distribute y at the daily interval:

```
proc ssm data=test;
    id date interval=day;
    startWeek = (weekday(date) = 2); /* indicator of Monday */
    state ...;
    comp term1 = ...;
    ...;
    state noise(1) type=wn ...;
    comp wnoise = noise[1];
    model y = term1 term2 ... wnoise / distribute(start=startWeek);
    /* daily_Y = sum of all terms in the MODEL statement */
    eval daily_Y = term1 + term2 + ...+ wnoise;
    output out=...;
 run;
```

Here are a few comments about this program:

- The terms in the MODEL statement correspond to the observation equation for the unobserved daily series $y_t^{\dagger}$. However, the DISTRIBUTE(START=*startWeek*) option causes the SSM procedure to use the implied model (with the augmented state vector) to analyze y—the weekly variable arranged as a daily series.

- Because wnoise—the white noise term (ϵ_t) in the observation equation of $y_t^{\dagger}$—is subsequently to be used in an EVAL statement, this program specifies it by using the STATE statement rather than by using the IRREGULAR statement.

- Because daily_Y (the component specified in the EVAL statement) is the sum of all the terms in the MODEL statement, it corresponds to the unobserved daily series $y_t^{\dagger}$. Therefore, the smoothed estimate of daily_Y (smoothed_daily_Y) provides the needed distribution of y at the daily interval.

In this release of the SSM procedure, the last element of the augmented state vector, y^f, is always initialized with diffuse distribution. A more flexible specification of the initial distribution of y^f might become possible in a future release.

To keep the explanation simple, the preceding discussion was confined to a single response variable. In fact, you can use the SSM procedure for temporal distribution in more general settings—for example, you can consider temporal distribution of one or more flow variables in a multivariate model that includes one or more response variables of stock type, one or more response variables of flow type, and one or more explanatory variables. An illustration of such modeling is shown in Example 33.16. The modeling of a response variable as a temporal aggregate of some unobserved latent variable is also needed in a process known as benchmarking; see Durbin and Koopman (2012, chap. 3, sec. 10.2) and Pelagatti (2015, chap. 9, sec. 2). You can use the SSM procedure in such benchmarking situations as well.

Temporal Aggregation

Temporal aggregation is the reverse of temporal distribution. In this case, the observations are available on a finer time scale, and you are interested in estimating the aggregated values on some coarser time scale—for example, estimating weekly totals from daily data. Of course, the aggregation is trivial in the historical region where the observations on the finer scale are known—in fact, in this case the estimation of the aggregate values is done with no estimation error. However, when the aggregate values are to be estimated in the region where the observations on the finer scale are missing—for example, in the forecast region—the problem becomes nontrivial. It is easier to explain the situation by using a simple example. Suppose $y_t, 1 \leq t \leq 100$, denote the daily observations of a response variable, y. Let $wy_t, t \geq 1$, denote the within-week daily running totals—that is, wy_t represents the total of y values up to the day t in the week that contains the day t. Clearly, given the daily values $y_t, 1 \leq t \leq 100$, the aggregate values $wy_t, 1 \leq t \leq 100$, are fully known. The question is, assuming that y follows a state space model, how do you estimate and obtain appropriate confidence intervals for wy_t in the forecast region ($t = 101, 102, \ldots$)? In this section you have already seen that when a variable is modeled by a state space model at a particular time interval, its aggregated form also follows a state space model. The AGGREGATE(START=) option in the MODEL statement of the SSM procedure enables you to perform temporal aggregation for a response variable. An illustration of such aggregation is shown in Example 33.17.

Covariance Parameterization

The covariance matrices specified by the COV and COV1 options in the STATE statement must be positive semidefinite. When these matrices are of general form and are not user-specified, they are internally parameterized by their Cholesky root. Suppose that Σ, an $m \times m$ positive semidefinite matrix of rank r, is such a covariance matrix. Then, Σ can always be written as

$$\Sigma = RR'$$

where the (generalized) Cholesky root, R, is an $m \times r$ lower triangular matrix with nonnegative diagonal elements (that is, $R[i,j] = 0$ if $j > i$ and $R[i,i] \geq 0,\ 1 \leq i \leq r$). The SSM procedure parameterizes Σ by the elements of its Cholesky root, which adds $r*(r+1)/2 + r*(m-r)$ elements to the parameter vector $\boldsymbol{\theta}$.

Missing Values

For a variety of reasons the data might contain missing response and predictor values. Before starting the analysis of a particular BY group, SSM procedure makes an internal copy of the data. The actual analysis is done by using this copy. The data in the copy are first examined for missing values in the response, predictor, and the ID variables. No missing values are permitted in the ID variable (if it is specified). If all the missing values are associated with only the response variables, then the internal copy of the data is not altered. However, if any of the predictors in the observation equation—the elements of $\mathbf{X}$ matrix—are found to contain missing values, the internal copy of the data is altered as follows: any missing predictor value is replaced by 0, and the response values that are dependent on that predictor in the corresponding row are set to missing. These missing response values are called the *induced missing values*. The reported analysis is based on the (possibly altered) internal copy of the BY group.

Missing values are not permitted in any of the other system matrices that define the state space model. In particular, missing values are not permitted in $\mathbf{Z}$, $\mathbf{T}$, $\mathbf{W}$, and $\mathbf{Q}$ matrices. In some cases the elements of these matrices depend on the data values. In such cases, care must be taken to ensure that these data values are not missing.

Computational Issues

A Well-Behaved Model

The model defined by the state space model equations (see the section "State Space Model and Notation" on page 2426) is very general. This generality is quite useful because it encompasses a wide variety of data generation processes. On the other hand, it also makes it easy to specify overly complex and numerically unstable models. If a suitable model is not already known and you are in the early phases of modeling, it is important to start with models that are relatively simple and well-behaved from the numerical standpoint. From the numerical and statistical considerations, two aspects of model formulation are particularly important: identifiability and numerical stability. A model is identifiable if the observed data has a distinct probability distribution for each admissible parameter vector. Unless proper care is taken, it is easy to specify an unidentifiable state space model. Similarly, predictions according to some types of state space models can display explosive growth or wild oscillations. This behavior is primarily governed by the transition matrix $\mathbf{T}$ (or $\mathbf{T}_t$ in the time-varying case). Unidentifiable models can run into difficulties during parameter estimation,

and explosive growth (and wild oscillation) causes numerical problems associated with finite-precision arithmetic. Unfortunately, no simple identifiability check is available for a general state space model, and it is difficult to decide at the outset whether a specified model might suffer from numerical instability. For a discussion of identifiability issues, see Harvey (1989, chap. 4, sec. 4). For a discussion of the stability properties of time-invariant state space models, see Harvey (1989, chap. 3, sec. 3). The following guidelines are likely to result in models that are identifiable and numerically stable:

- Build models by composing submodels that are known to be well-behaved. The predefined models provided by the SSM procedure are good submodel candidates (see the sections "Predefined Trend Models" on page 2442 and "Predefined Structural Models" on page 2445).
- Pay careful attention to the way the variety of system matrices are defined. The behavior of their elements, as functions of model parameters and other variables, must be well-understood. If these elements are defined by using DATA steps, you can validate their behavior by running these DATA steps outside of the SSM procedure. In particular, note the following:
  - The transition matrix $\mathbf{T}$ (or $\mathbf{T}_t$ in the time-varying case) determines the explosiveness characteristics of the model; it must be well-behaved for all parameters.
  - The disturbance covariances $\mathbf{Q}_t$ must be positive semidefinite for all parameters.
  - If the system matrices in the state equation, such as the transition matrix $\mathbf{T}_t$ or the disturbance covariance $\mathbf{Q}_t$, are time-varying and the data contain replicate observations (observations with the same ID value), check that the elements of these matrices do not vary during replicate observations. This follows from the fact that the underlying state does not vary during replications (see the state equation in the section "State Space Model and Notation" on page 2426 and the section "Types of Sequence Data" on page 2428).

Convergence Problems

As explained in the section "Likelihood Computation and Model-Fitting Phase" on page 2435, the model parameters are estimated by nonlinear optimization of the likelihood. This process is not guaranteed to succeed. For some data sets, the optimization algorithm can fail to converge. Nonconvergence can result from a number of causes, including flat or ridged likelihood surfaces and ill-conditioned data. It is also possible for the algorithm to converge to a point that is not the global optimum of the likelihood.

If you experience convergence problems, consider the following:

- Data that are extremely large or extremely small can adversely affect results because of the internal tolerances used during the filtering steps of the likelihood calculation. Rescaling the data can improve stability.
- Whenever possible, parameterize the disturbance variances in the model on the exponential scale. For illustrations of parameterizing disturbance variances in this manner, see Example 33.12 and Example 33.14.
- Examine your model for redundancies in the included components and regressors. The components or regressors that are nearly collinear to each other can cause the optimization process to become unstable.
- Lack of convergence can indicate model misspecification such as unidentifiable model or a violation of the normality assumption.

Computer Resource Requirements

The computing resources required for the SSM procedure depend on several factors. The memory requirement for the procedure is largely dependent on the number of observations to be processed and the size of the state vector underlying the specified model. If n denotes the sample size and m denotes the size of the state vector, the memory requirement for the smoothing phase of the Kalman filter is of the order of $6 \times 8 \times n \times m^2$ bytes, ignoring the lower-order terms. If the smoothed component estimates are not needed, then the memory requirement is of the order of $6 \times 8 \times (m^2 + n)$ bytes. Besides m and n, the computing time for the parameter estimation depends on the size of the parameter vector $\boldsymbol{\theta}$ and how many likelihood evaluations are needed to reach the optimum.

Displayed Output

The default printed output produced by the SSM procedure contains the following information:

- brief information about the input data set, including the data set name and label
- summary statistics for the response variables in the model, including the names of the variables, the total number of observations and the number of missing observations, the smallest and largest measurements, and the mean and standard deviation
- information about the index variable, including the index value of the first and the last observation, the maximum difference between the successive index values, the number of distinct index values, and the categorization of the data into regular, regular with replication, or irregular types
- estimates of the regression parameters if the model contains any predictors, including their standard errors, t statistics, and p-values
- convergence status of the likelihood optimization process if any parameters are estimated
- estimates of the free parameters at the end of the model-fitting phase, including the parameter estimates and their approximate standard errors
- the likelihood-based goodness-of-fit statistics, including the full likelihood, the sum of squares of residuals normalized by their standard errors, and the information criteria: AIC, AICC, HQIC, BIC, and CAIC
- summary of most significant additive outliers

ODS Table Names

The SSM procedure assigns a name to each table it creates. You can use these names to refer to the table when you use the Output Delivery System (ODS) to select tables and create output data sets. These names are listed in Table 33.9.

Table 33.9 ODS Tables Produced by PROC SSM

ODS Table Name	Description	Statement	Option
Tables That Summarize the Model Information			
ModelSummary	Summary information about the underlying state space model		Default
IdInformation	Summary information about the ID variable		Default
ResponseInfo	Summary information about the response variables		Default
StateSummary	Summary information about the model state vector	PROC SSM	STATEINFO
DiffuseStateSummary	Summary information about the diffuse initial state	PROC SSM	STATEINFO
Tables Related to Model Parameters and the Likelihood			
ConvergenceStatus	Convergence status of the estimation process		Default
RegressionEstimates	Estimates of the regression parameters	MODEL	Default
StateRegressionEstimates	Estimates of the state regression parameters	STATE	W
FixedStateEstimates	Estimates of time-invariant, non-stochastic state subsections		Default
NamedParameterEstimates	Estimates of the parameters specified in the PARMS statement	PARMS	Default
ParameterEstimates	Estimates of the unknown elements in the model system matrices		Default
DisturbanceCovariance	Estimate of the disturbance covariance	STATE	PRINT=COV
InitialCovariance	Estimate of the initial state covariance	STATE	PRINT=COV1
ARCoefficient	Estimate of the autoregressive coefficient matrix	STATE	PRINT=AR
MACoefficient	Estimate of the moving-average coefficient matrix	STATE	PRINT=MA
TransitionMatrix	Estimate of the state transition matrix	STATE	PRINT=T
FitSummary	Summary of the likelihood-based fit-statistics		Default

Table 33.9 *continued*

ODS Table Name	Description	Statement	Option
InformationCriteria	Likelihood-based information criteria		Default
Tables Related to Series and Component Forecasts			
Forecasts	Series forecasts	MODEL	PRINT=FILTER
SmoothedResponse	Smoothed series values	MODEL	PRINT=SMOOTH
FilteredComponent	Component forecasts	COMPONENT	PRINT=FILTER
SmoothedComponent	Smoothed component	COMPONENT	PRINT=SMOOTH
Tables Related to Outlier Detection and Model Quality			
AOSummary	Summary of additive outliers	Default	Default
ElementTrendBreakSummary	Elementwise trend break summary	TREND	CHECKBREAK
OverallTrendBreakSummary	Overall trend break summary	TREND	CHECKBREAK(OVERALL)
StateElementBreakSummary	Elementwise state break summary	STATE	CHECKBREAK
OverallStateBreakSummary	Overall state break summary	STATE	CHECKBREAK(OVERALL)
MaximalShockSummary	Summary of maximal state shocks	OUTPUT	MAXSHOCK
PRESS	Prediction error sum of squares	OUTPUT	PRESS

ODS Graph Names

You can refer to every graph produced through ODS Graphics with a name. The names of the graphs that PROC SSM generates are listed in Table 33.10, along with the required statements and options.

Table 33.10 ODS Graphs Produced by PROC SSM

ODS Graph Name	Description	Statement	Option
Graphs for One-Step-Ahead Residual Analysis			
ResidualNormalityPlot	Normality check	PROC SSM	PLOTS=RESIDUAL(NORMAL)
ResidualHistogram	Residual histogram	PROC SSM	PLOTS(UNPACK)=RESIDUAL
ResidualQQPlot	Residual Q-Q plot	PROC SSM	PLOTS(UNPACK)=RESIDUAL
StdResidualPlot	Time series plot of standardized residuals	PROC SSM	Default
Graphs Related to Outlier Detection and Structural Break			
PredErrorNormalityPlot	Normality check	PROC SSM	PLOTS=AO(NORMAL)
PredErrorHistogram	Prediction error histogram	PROC SSM	PLOTS(UNPACK)=AO
PredErrorQQPlot	Prediction error Q-Q plot	PROC SSM	PLOTS(UNPACK)=AO

Table 33.10 *continued*

ODS Graph Name	Description	Statement	Option
StdPredErrorPlot	Time series plot of standardized additive-outlier statistics	PROC SSM	PLOTS=AO(STD)
MaximalShockPlot	Time series plot of maximal state shock chi-square statistics	PROC SSM	PLOTS=MAXSHOCK

OUT= Data Set

You can use the OUT= option in the OUTPUT statement to store the series and component forecasts that are produced by PROC SSM. Which columns are included in the data set depends on the model specification. The model can have one or more response variables, a variety of components that appear in the MODEL statement, and components specified by the EVAL statement. The OUT= data set contains the one-step-ahead and full-sample estimates of the response variables, and all these components.

The following list describes the columns of the data set:

- the BY variables
- the ID variable, if specified by the ID statement
- Obs, a variable that contains the observation number
- the response series (more than one in the multivariate case)
- the following columns associated with the response series (the wildcard * is substituted by the name of one of the response variables):
  - FORECAST_* contains the one-step-ahead predicted values. and the multistep forecasts of the response series.
  - RESIDUAL_* contains the difference between the actual and forecast values.
  - StdErr_* contains the standard error of prediction.
  - Lower_* and Upper_* contain the lower and upper forecast confidence limits.
  - Smoothed_* contains the smoothed values of the response variable.
  - StdErr_Smoothed_* contains standard errors of the smoothed values of the response variable.
  - AO_* contains the additive outlier estimate.
  - StdErr_AO_* contains the standard error of the additive outlier estimate.
- the following columns associated with the components (the wildcard * is substituted by the name of one of the components):
  - FORECAST_* contains the one-step-ahead predicted values and the multistep forecasts of the component.

  - StdErr_* contains the standard error of prediction.
  - Smoothed_* contains the smoothed values of the component.
  - StdErr_Smoothed_* contains standard errors of the smoothed values of the component.
  - Smoothed_Lower_* and Smoothed_Upper_* contain the lower and upper confidence limits of the smoothed component.

- the maximal state shock chi-square statistics at distinct time points (this column is present only if the MAXSHOCK option is used in the OUTPUT statement)

Confidence limits are not produced for the smoothed series values or for the component forecasts; they are produced for the smoothed components.

Examples: SSM Procedure

Example 33.1: Bivariate Basic Structural Model

This example illustrates how you can use the SSM procedure to analyze a bivariate time series. The following data set contains two variables, f_KSI and r_KSI, which are measured quarterly, starting the first quarter of 1969. The variable f_KSI represents the quarterly average of the log of the monthly totals of the front-seat passengers killed or seriously injured during the car accidents, and r_KSI represents a similar number for the rear-seat passengers. The data set has been extended at the end with eight missing values, which represent four quarters, to cause the SSM procedure to produce model forecasts for this span.

```
data seatBelt;
input f_KSI r_KSI @@;
label f_KSI = "Front Seat Passengers Injured--log scale";
label r_KSI = "Rear Seat Passengers Injured--log scale";
date = intnx( 'quarter', '1jan1969'd, _n_-1 );
format date YYQS.;
datalines;
    6.72417 5.64654  6.81728 6.06123  6.92382 6.18190
    6.92375 6.07763  6.84975 5.78544  6.81836 6.04644
    7.00942 6.30167  7.09329 6.14476  6.78554 5.78212
    6.86323 6.09520  6.99369 6.29507  6.98344 6.06194
    6.81499 5.81249  6.92997 6.10534  6.96356 6.21298
    7.02296 6.15261  6.76466 5.77967  6.95563 6.18993
    7.02016 6.40524  6.87849 6.06308  6.55966 5.66084
    6.73627 6.02395  6.91553 6.25736  6.83576 6.03535
    6.52075 5.76028  6.59860 5.91208  6.70597 6.08029
    6.75110 5.98833  6.53117 5.67676  6.52718 5.90572
    6.65963 6.01003  6.76869 5.93226  6.44483 5.55616
    6.62063 5.82533  6.72938 6.04531  6.82182 5.98277
    6.64134 5.76540  6.66762 5.91378  6.83524 6.13387
    6.81594 5.97907  6.60761 5.66838  6.62985 5.88151
    6.76963 6.06895  6.79927 6.01991  6.52728 5.69113
    6.60666 5.92841  6.72242 6.03111  6.76228 5.93898
    6.54290 5.72538  6.62469 5.92028  6.73415 6.11880
```

```
    6.74094 5.98009  6.46418 5.63517  6.61537 5.96040
    6.76185 6.15613  6.79546 6.04152  6.21529 5.70139
    6.27565 5.92508  6.40771 6.13903  6.37293 5.96883
    6.16445 5.77021  6.31242 6.05267  6.44414 6.15806
    6.53678 6.13404 . . . . . .
run;
```

These data have been analyzed in Durbin and Koopman (2012, chap. 8, sec. 3). The analysis presented here is similar. To simplify the illustration, the monthly data have been converted to quarterly data and two predictors (the number of kilometers traveled and the real price of petrol) are excluded from the analysis. You can also use PROC SSM to carry out the more elaborate analysis in Durbin and Koopman (2012).

One of the original reasons for studying these data was to assess the effect on f_KSI of the enactment of a seat-belt law in February 1983 that compelled the front seat passengers to wear seat belts. A simple graphical inspection of the data (not shown here) reveals that f_KSI and r_KSI do not show a pronounced upward or downward trend but do show seasonal variation, and that these two series seem to move together. Additional inspection also shows that the seasonal effect is relatively stable throughout the data span. These considerations suggest the following model for $\mathbf{y}$ = (f_KSI, r_KSI):

$$\mathbf{y}_t = \begin{pmatrix} X_t \\ 0 \end{pmatrix} \beta + \boldsymbol{\mu}_t + \boldsymbol{\zeta}_t + \boldsymbol{\xi}_t$$

All the terms on the right-hand side of this equation are assumed to be statistically independent. These terms are as follows:

- The predictor X_t (defined as Q1_83_Shift later in the program) denotes a variable that is 0 before the first quarter of 1983, and 1 thereafter. X_t is supposed to affect only f_KSI (the first element of $\mathbf{y}$); it represents the enactment of the seat-belt law of 1983.
- $\boldsymbol{\mu}_t$ denotes a bivariate random walk. It is supposed to capture the slowly changing level of the vector $\mathbf{y}_t$. To capture the fact that f_KSI and r_KSI move together (that is, they are co-integrated), the covariance of the disturbance term of this random walk is assumed to be of lower than full rank.
- $\boldsymbol{\zeta}_t$ denotes a bivariate trigonometric seasonal term. In this model, it is taken to be fixed (that is, the seasonal effects do not change over time).
- $\boldsymbol{\xi}_t$ denotes a bivariate white noise term, which captures the residual variation that is unexplained by the other terms in the model.

The preceding model is an example of a (bivariate) basic structural model (BSM). The following statements specify and fit this model to f_KSI and r_KSI:

```
proc ssm data=seatBelt stateinfo;
   id date interval=quarter;
   Q1_83_Shift = (date >= '1jan1983'd);
   state error(2) type=WN cov(g) print=cov;
   component wn1 = error[1];
   component wn2 = error[2];
   state level(2) type=RW cov(rank=1) print=cov;
   component rw1 = level[1];
   component rw2 = level[2];
   state season(2) type=season(length=4);
```

```
    component s1 = season[1];
    component s2 = season[2];
    model f_KSI = Q1_83_Shift rw1 s1  wn1 / print=(smooth);
    model r_KSI = rw2 s2 wn2;
    eval f_KSI_sa = rw1 + Q1_83_Shift;
    output out=For1;
run;
```

The PROC SSM statement specifies the input data set, seatBelt. The use of the STATEINFO option in the PROC SSM statement produces additional information about the model state vector and its diffuse initial state. The optional ID statement specifies an index variable, date. The INTERVAL=QUARTER option in the ID statement indicates that the measurements were collected on a quarterly basis. Next, a programming statement defines Q1_83_Shift, the predictor that represents the enactment of the seat-belt law of 1983. It is used later in the MODEL statement for f_KSI. Separate STATE statements specify the terms $\boldsymbol{\mu}_t$, $\boldsymbol{\zeta}_t$, and $\boldsymbol{\xi}_t$ because they are statistically independent. Each model that governs them (white noise for $\boldsymbol{\xi}_t$, random walk for $\boldsymbol{\mu}_t$, and trigonometric seasonal for $\boldsymbol{\zeta}_t$) can be specified by using the TYPE= option of the STATE statement. When you use the TYPE= option, you can use the COV option to specify the information about the disturbance covariance in the state transition equation. The other details, such as the transition matrix specification and the specification of $\mathbf{A}_1$ in the initial condition, are inferred from the TYPE= option. The use of PRINT=COV in the STATE statement causes the estimated disturbance covariance to be printed. For $\boldsymbol{\xi}_t$ (a white noise), $\mathbf{A}_1$ is zero and $\mathbf{Q}_t = \mathbf{Q}$ for all $t \geq 1$, where $\mathbf{Q}$ is specified by the COV option. For $\boldsymbol{\mu}_t$ and $\boldsymbol{\zeta}_t$ the initial condition is fully diffuse—that is, $\mathbf{A}_1$ is an identity matrix of appropriate order and $\mathbf{Q}_1 = 0$. The total diffuse dimension of this model, $(d + k)$, is $9 = 8 + 1$ as a result of one predictor, Q1_83_Shift, and two fully diffuse state subsections, $\boldsymbol{\mu}_t$ and $\boldsymbol{\zeta}_t$. The components in the model are defined by suitable linear combinations of these different state subsections. The program statements define the model as follows:

- **state error(2) type=WN cov(g);** defines $\boldsymbol{\xi}_t$ as a two-dimensional white noise, named error, with the covariance of general form. Then two COMPONENT statements define wn1 and wn2 as the first and second elements of error, respectively.

- **state level(2) type=RW cov(rank=1);** defines $\boldsymbol{\mu}_t$ as a two-dimensional random walk, named level, with covariance of general form whose rank is restricted to 1. Then two COMPONENT statements define rw1 and rw2 as the first and second elements of level, respectively.

- **state season(2) type=season(length=4);** defines $\boldsymbol{\zeta}_t$ as a two-dimensional trigonometric seasonal of season length 4, named season, with zero covariance—signified by the absence of the COV option. Then two COMPONENT statements define s1 and s2 as appropriate linear combinations of season so that s1 represents the seasonal for f_KSI and s2 represents the seasonal for r_KSI. Because TYPE=SEASON in the STATE statement, the COMPONENT statement appropriately interprets **component s1 = season[1];** as s1 being a dot product: $(1\ 0\ 0\ 0\ 1\ 0) * \text{season}$. For more information, see the section "Multivariate Season" on page 2448.

- **model f_KSI = Q1_83_Shift rw1 s1 wn1;** defines the model for f_KSI, and **model r_KSI = rw2 s2 wn2;** defines the model for r_KSI.

The SSM procedure fits the model and reports the parameter estimates, their approximate standard errors, and the likelihood-based goodness-of-fit measures by default. In order to output the one-step-ahead and full-sample estimates of the components in the model, you can either use the PRINT= options in the MODEL statement and the respective COMPONENT statements or you can specify an output data set in the OUTPUT statement. In addition, you can use the EVAL statement to define specific linear combinations of the

underlying state that should also be estimated. The statement **`eval f_KSI_sa = rw1 + Q1_83_Shift;`** is an example of one such linear combination. It defines f_KSI_sa, a linear combination that represents the seasonal adjustment of f_KSI. The output data set, For1 (named in the OUTPUT statement) contains estimates of all the model components in addition to the estimate of f_KSI_sa.

The model summary table, shown in Output 33.1.1, provides basic model information, such as the dimension of the underlying state equation ($m = 10$), the diffuse dimension of the model ($(d + k) = 9$), and the number of parameters (5) in the model parameter vector $\boldsymbol{\theta}$.

Output 33.1.1 Bivariate Basic Structural Model

The SSM Procedure

Model Summary	
Model Property	**Value**
Number of Model Equations	2
State Dimension	10
Dimension of the Diffuse Initial Condition	9
Number of Parameters	5

Additional details about the role of different components in forming the model state and its diffuse initial condition are shown in Output 33.1.2 and Output 33.1.3. They show that the 10-dimensional model state vector is made up of subsections that are associated with error and level (each of dimension 2) and season (of dimension 6). Similarly, the nine-dimensional diffuse vector in the initial condition is made up of subsections that correspond to level, season, and the regression variable, Q1_83_Shift. Note that error does not contribute to the diffuse initial vector because it has a fully nondiffuse initial state.

Output 33.1.2 Bivariate Basic Structural Model State Vector Summary

State Vector Composition	
Subsection	**Dimension**
error	2
level	2
season	6

Output 33.1.3 Bivariate Basic Structural Model Initial Diffuse State Vector Summary

Diffuse Initial State Composition (Including Regressors)	
Subsection	**Dimension**
level	2
season	6
Q1_83_Shift	1

The index variable information is shown in Output 33.1.4.

Output 33.1.4 Index Variable Information

ID Variable Information

Name	Start	End	Max Delta	NDistinct	Type
date	1969:1	1985:4	1	68	Regular

Output 33.1.5 provides simple summary information about the response variables. It shows that f_KSI and r_KSI have four missing values each and no induced missing values because the predictor in the model, Q1_83_Shift, has no missing values.

Output 33.1.5 Response Variable Summary

Response Variable Information

Name	Number of Observations Total	Missing	Induced Missing	Minimum	Maximum	Mean	Std Deviation
f_KSI	68	4	0	6.16	7.09	6.71	0.206
r_KSI	68	4	0	5.56	6.41	5.97	0.186

The regression coefficient of Q1_83_Shift, shown in Output 33.1.6, is negative and is statistically significant. This is consistent with the expected drop in f_KSI after the enactment of the seat-belt law.

Output 33.1.6 Regression Coefficient of Q1_83_Shift

Regression Parameter Estimates

Response Variable	Regression Variable	Estimate	Standard Error	t Value	Pr > |t|
f_KSI	Q1_83_Shift	-0.408	0.0259	-15.74	<.0001

Output 33.1.7 shows the estimates of the elements of $\boldsymbol{\theta}$. The five parameters in $\boldsymbol{\theta}$ correspond to unknown elements that are associated with the covariance matrices in the specifications of error and level. Whenever a covariance specification is of a general form and is not defined by a user-specified variable list, it is internally parameterized as a product of its Cholesky root: $\mathrm{Cov} = \mathrm{Root}\ \mathrm{Root}'$. This ensures that the resulting covariance is positive semidefinite. The Cholesky root is constrained to be lower triangular, with positive diagonal elements. If rank constraints (such as the rank-one constraint on the covariance in the specification of level) are imposed, the number of free parameters in the Cholesky factor is reduced appropriately. For more information, see the section "Covariance Parameterization" on page 2456. In view of these considerations, the five parameters in $\boldsymbol{\theta}$ are a result of three parameters from the Cholesky root of error and two parameters that are associated with the Cholesky root of level.

Output 33.1.7 Parameter Estimates

Model Parameter Estimates					
Component	Type	Parameter	Estimate	Standard Error	t Value
error	Disturbance Covariance	RootCov[1, 1]	0.0361	0.00736	4.91
error	Disturbance Covariance	RootCov[2, 1]	0.0338	0.01131	2.99
error	Disturbance Covariance	RootCov[2, 2]	0.0462	0.00470	9.84
level	Disturbance Covariance	RootCov[1, 1]	0.0375	0.00843	4.45
level	Disturbance Covariance	RootCov[2, 1]	0.0223	0.00569	3.92

Output 33.1.8 shows the resulting covariance estimate of error after multiplying the Cholesky factors.

Output 33.1.8 White Noise Covariance Estimate

Disturbance Covariance for error		
	Col1	Col2
Row1	0.001307	0.001222
Row2	0.001222	0.003277

Similarly, Output 33.1.9 shows the covariance estimate of level disturbance. Note that because of the rank-one constraint, the determinant of this matrix is 0.

Output 33.1.9 Covariance Estimate of the Random Walk Disturbance

Disturbance Covariance for level		
	Col1	Col2
Row1	0.001408	0.000837
Row2	0.000837	0.000497

Output 33.1.10 shows the likelihood computation summary. This table is produced by using the fitted model to carry out the filtering operation on the data. For more information, see the section "Likelihood Computation and Model-Fitting Phase" on page 2435.

Output 33.1.10 Likelihood Computation Summary of the Fitted Model

Likelihood Computation Summary	
Statistic	Value
Nonmissing Response Values Used	128
Estimated Parameters	5
Initialized Diffuse State Elements	9
Normalized Residual Sum of Squares	119
Diffuse Log Likelihood	166.15755
Profile Log Likelihood	199.91165

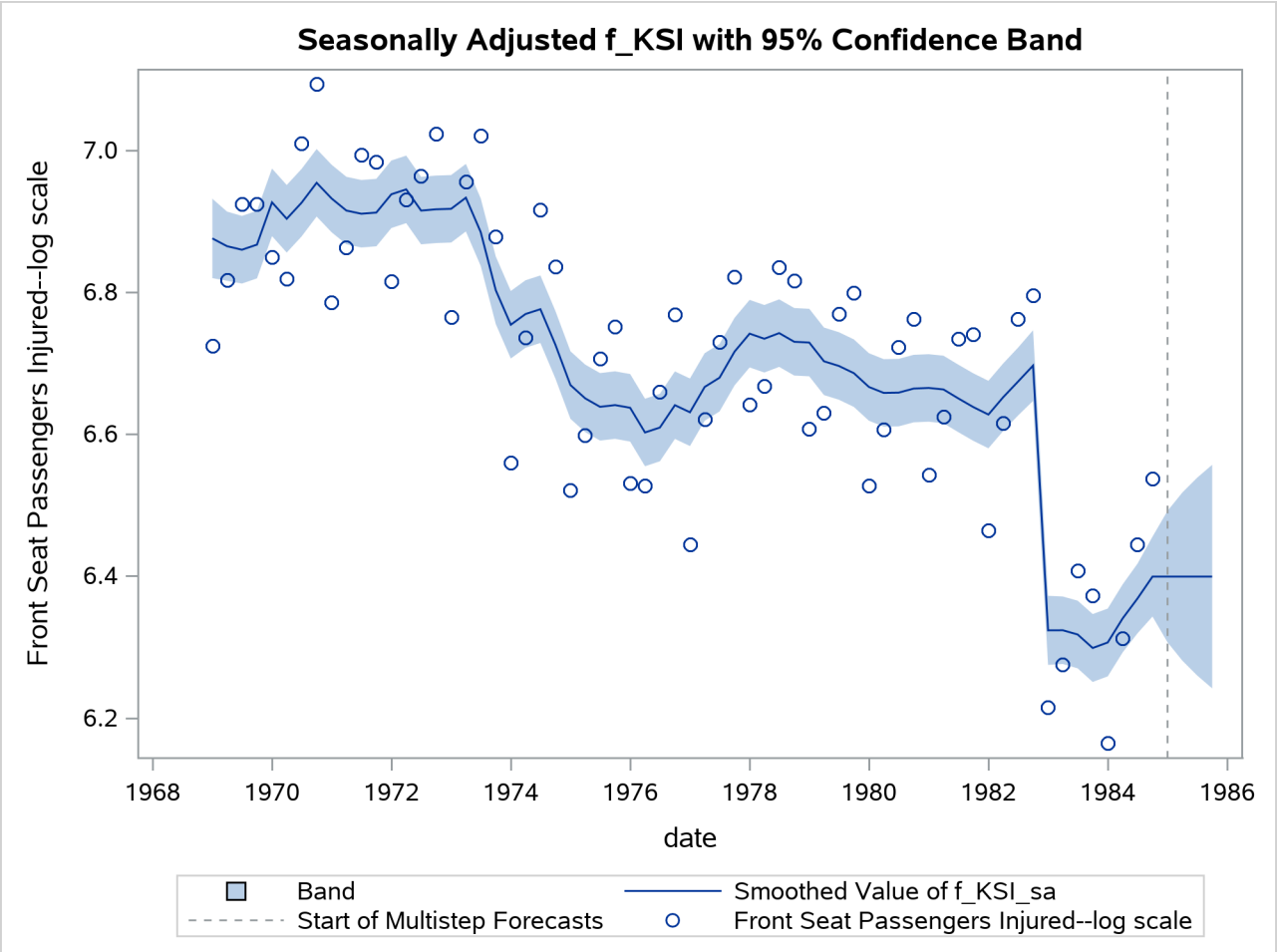
The output data set, For1, specified in the OUTPUT statement contains one-step-ahead and full-sample estimates of all the model components and the user-specified components (defined by the EVAL statement). Their standard errors and the upper and lower confidence limits (by default, 95%) are also produced.

The following statements use the For1 data set to produce a time series plot of the seasonally adjusted f_KSI:

```
proc sgplot data=For1;
   title "Seasonally Adjusted f_KSI with 95% Confidence Band";
   band x=date lower=smoothed_lower_f_KSI_sa
      upper=smoothed_upper_f_KSI_sa ;
   series x=date y=smoothed_f_KSI_sa;
   refline '1jan1985'd / axis=x lineattrs=(pattern=shortdash)
      LEGENDLABEL= "Start of Multistep Forecasts"
      name="Forecast Reference Line";
   scatter x=date y=f_KSI ;
run;
```

The generated plot is shown in Output 33.1.11.

Output 33.1.11 Plot of Seasonally Adjusted f_KSI



Example 33.2: Panel Data: Random-Effects and Autoregressive Models

This example shows how you can use the SSM procedure to specify and fit the two-way random-effects model and the autoregressive model to analyze a panel of time series. The fitting of dynamic panel model for such data is illustrated in Example 33.11. These (and a few other) model types can also be fitted by the PANEL

procedure, a SAS/ETS procedure that is specially designed to efficiently handle the cross-sectional time series data. However, because of the differences in their model fitting algorithms, generally the parameter estimates and other fit statistics produced by the SSM and PANEL procedures do not match. The SSM procedure always uses the (restricted) maximum likelihood for parameter estimation. The estimation method used by the PANEL procedure depends on the model type and the particular estimation options.

The cross-sectional data, Cigar, that are used in the section "Getting Started: SSM Procedure" on page 2394 are reused in this example. The output shown here is less extensive than the output shown in that section. The main emphasis of this example is how you can specify the two-way random effects model and the autoregressive model in the SSM procedure.

According to the two-way random effects model, the cigarette sales, lsales, can be described by the following equation:

$$\text{lsales}_{i,t} = \mu + \text{lprice}\,\beta_1 + \text{lndi}\,\beta_2 + \text{lpimin}\,\beta_3 + \zeta_i + \eta_t + \epsilon_{i,t}$$

This model represents lsales in region i and in year t as a sum of an overall intercept μ, the regression effects due to lprice, lndi, and lpimin, a zero-mean, random effect ζ_i associated with region i, a zero-mean, random effect η_t associated with year t, and the observation noise $\epsilon_{i,t}$. The region-specific random effects ζ_i and the year-specific random effects η_t are assumed to be independent, Gaussian sequences with variances σ_ζ^2 and σ_η^2, respectively. In addition, they are assumed to be independent of the observation noise, which is also assumed to be a sequence of independent, zero-mean, Gaussian variables with variance σ_ϵ^2.

You can specify and fit this model by using the following statements:

```
proc ssm data=Cigar;
   id year interval=year;
   parms s2g/ lower=(1.e-6);
   array RegionArray{46} region1-region46;
   do i=1 to 46;
      RegionArray[i] = (region=i);
   end;
   /* region-specific random effects */
   state zeta(46) T(I) cov1(I)=(s2g);
   component regionEffect = zeta * (RegionArray);
   /* year-specific random effect */
   state eta(1) type=wn cov(D);
   component timeEffect = eta[1];
   irregular wn;
   intercept = 1.0;
   model lsales = intercept lprice lndi lpimin
       timeEffect regionEffect wn;
run;
```

The PARMS statement defines s2g, a parameter that is restricted to be positive and is used later as the variance parameter for the region effect. Similarly the 46-dimensional array, RegionArray, of region-specific dummy variables is defined to be used later. The state subsection zeta corresponds to ζ, which is the 46-dimensional vector of region-specific, zero-mean, random effects. The component regionEffect extracts the proper element of ζ by using the array RegionArray. A constant column, intercept, is defined to be used later as an intercept term. The component timeEffect corresponds to η_t, and wn specifies the observation noise ϵ_{it}. Finally the MODEL statement defines the model. Some of the tables that are produced by running these statements are shown in Output 33.2.1 through Output 33.2.5.

The model summary, shown in Output 33.2.1, shows that the model is defined by one MODEL statement, the dimension of the underlying state vector is 47 (because ζ is 46-dimensional and η_t is one-dimensional), the diffuse dimension is 4 (because of the four predictors in the model), and there are three parameters to be estimated.

Output 33.2.1 Two-Way Random-Effects Model: Model Summary

The SSM Procedure

Model Summary	
Model Property	**Value**
Number of Model Equations	1
State Dimension	47
Dimension of the Diffuse Initial Condition	4
Number of Parameters	3

Output 33.2.2 provides the likelihood information about the fitted model.

Output 33.2.2 Two-Way Random-Effects Model: Likelihood Summary

Likelihood Computation Summary	
Statistic	**Value**
Nonmissing Response Values Used	1380
Estimated Parameters	3
Initialized Diffuse State Elements	4
Normalized Residual Sum of Squares	1376.0001
Diffuse Log Likelihood	1459.0277
Profile Log Likelihood	1470.8628

Output 33.2.3 shows the regression estimates.

Output 33.2.3 Two-Way Random-Effects Model: Regression Estimates

Regression Parameter Estimates					
Response Variable	**Regression Variable**	**Estimate**	**Standard Error**	**t Value**	**Pr > |t|**
lsales	**intercept**	2.798	0.1136	24.62	<.0001
lsales	**lprice**	-0.903	0.0365	-24.73	<.0001
lsales	**lndi**	0.592	0.0246	24.08	<.0001
lsales	**lpimin**	0.127	0.0398	3.18	0.0015

The ML estimate of s2g, a parameter specified in the PARMS statement, is shown in Output 33.2.4. It corresponds to σ_ξ^2, the variance of the region effect.

Output 33.2.4 Two-Way Random-Effects Model: Estimate of σ_ξ^2

Estimates of Named Parameters			
Parameter	**Estimate**	**Standard Error**	**t Value**
s2g	0.0241	0.00512	4.70

Output 33.2.5 Variance Estimates of η_t and ϵ_{it}

Model Parameter Estimates

Component	Type	Parameter	Estimate	Standard Error	t Value
eta	Disturbance Covariance	Cov[1, 1]	0.000681	0.000264	2.58
wn	Irregular	Variance	0.005698	0.000224	25.40

The estimates of the other unknown parameters in the model are shown in Output 33.2.5. It shows the estimate of the variance of the irregular component wn and the estimate of the variance of the time effect η_t.

The remainder of this example describes how you can specify and fit the following first-order vector autoregessive model to the cigarette data:

$$\begin{aligned} \text{lsales}_{i,t} &= \boldsymbol{\mu} + \text{lprice}\,\boldsymbol{\beta}_1 + \text{lndi}\,\boldsymbol{\beta}_2 + \text{lpimin}\,\boldsymbol{\beta}_3 + \boldsymbol{\zeta}_t[i] \\ \boldsymbol{\zeta}_t &= \boldsymbol{\Phi}\,\boldsymbol{\zeta}_{t-1} + \boldsymbol{\eta}_t \end{aligned}$$

This model represents lsales in region i and in year t as a sum of an overall intercept $\boldsymbol{\mu}$, the regression effects due to lprice, lndi, and lpimin, and the ith element of a vector error term $\boldsymbol{\zeta}_t[i]$. The multidimensional error sequence $\boldsymbol{\zeta}_t$ is assumed to follow a first-order autoregression with a diagonal autoregressive coefficient matrix $\boldsymbol{\Phi}$ and with a multivariate, white noise sequence $\boldsymbol{\eta}_t$ as its disturbance sequence. The covariance matrix of $\boldsymbol{\eta}_t$, $\boldsymbol{\Sigma}$, is assumed to be dense. Note that the dimension of the vectors $\boldsymbol{\zeta}_t$ is the same as the number of cross sections in the study (the number of regions in this example). Therefore, even for a relatively modest panel study, the total number of parameters to be estimated can get quite large. Therefore, in this example only the first three regions are considered in the analysis. The following statements specify and fit this model to the Cigar data set:

```
proc ssm data=Cigar;
   where region <= 3;
   id year interval=year;
   array RegionArray{3} region1-region3;
   do i=1 to 3;
      RegionArray[i] = (region=i);
   end;
   state zeta(3) type=varma(p(d)=1) cov(g) print=(ar cov);
   component eta = zeta*(RegionArray);
   intercept = 1.0;
   model lsales = intercept lprice lndi lpimin eta;
run;
```

The vectors $\boldsymbol{\zeta}_t$ are specified in the STATE statement. The TYPE= specification signifies that the three-dimensional state subsection, zeta, follows a vector AR(1) model with a diagonal transition matrix and a disturbance covariance of a general form. The PRINT=(AR COV) option causes the SSM procedure to print the estimated AR coefficient matrix, $\boldsymbol{\Phi}$, and the disturbance error covariance $\boldsymbol{\Sigma}$, respectively. The COMPONENT statement defines the appropriate error contribution (named eta), $\boldsymbol{\zeta}_t[i]$. Output 33.2.6 shows the estimated regression coefficients, Output 33.2.7 shows the estimate of $\boldsymbol{\Phi}$, and Output 33.2.8 shows the estimate of $\boldsymbol{\Sigma}$:

Output 33.2.6 Autoregressive Model: Regression Estimates

The SSM Procedure

Regression Parameter Estimates

Response Variable	Regression Variable	Estimate	Standard Error	t Value	Pr > |t|
lsales	intercept	3.6857	0.3961	9.31	<.0001
lsales	lprice	-0.2356	0.0833	-2.83	0.0047
lsales	ldi	0.1969	0.0774	2.54	0.0110
lsales	lpimin	0.0737	0.0995	0.74	0.4588

Output 33.2.7 Estimate of the AR Coefficient Φ

AR Coefficient Matrix for zeta

	Col1	Col2	Col3
Row1	0.925707	0	0
Row2	0	0.984015	0
Row3	0	0	0.960071

Output 33.2.8 Estimate of the Disturbance Covariance Σ

Disturbance Covariance for zeta

	Col1	Col2	Col3
Row1	0.000911	0.000342	0.000361
Row2	0.000342	0.002216	0.000172
Row3	0.000361	0.000172	0.000923

Example 33.3: Backcasting, Forecasting, and Interpolation

This example illustrates how you can do model-based extrapolation—backcasting, forecasting, or interpolation—of a response variable. All you need is to appropriately augment the input data set with the relevant ID and predictor information and assign missing values to the response variable in these places. The following DATA step creates one such augmented data set by using a well-known data set that contains recordings of the Nile River water level measured between the years 1871 and 1970. Suppose you want to backcast the Nile water level for two years before 1871, forecast it for two years after 1970, and interpolate its value for the year 1921—for illustration purposes, this value is assumed to be missing in the available data set.

```
data Nile;
   input level @@;
   year = intnx( 'year', '1jan1869'd, _n_-1 );
   format year year4.;
   if year = '1jan1921'd then level=.;
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
874   694   940  833   701   916   692  1020   1050  969
831   726   456  824   702   1120  1100 832    764   821
768   845   864  862   698   845   744  796    1040  759
781   865   845  944   984   897   822  1010   771   676
649   846   812  742   801   1040  860  874    848   890
744   749   838  1050  918   986   797  923    975   815
1020  906   901  1170  912   746   919  718    714   740
. .
;
```

It is also known that for this time span the Nile water level can be reasonably modeled as a sum of a random walk trend, a level shift in the year 1899, and the observation error. The following statements fit this model to the data:

```
proc ssm data=Nile;
    id year interval=year;
    shift1899 = ( year >= '1jan1899'd );
    trend rw(rw);
    irregular wn;
    model level = shift1899 RW wn  / print=smooth;
    output out=nileOut;
quit;
```

The model-based interpolated and extrapolated values of the Nile water level are shown in Output 33.3.1, which is produced by using the PRINT=SMOOTH option in the MODEL statement.

Output 33.3.1 Interpolated and Extrapolated Nile Water Level

The SSM Procedure

Full-Sample Prediction of Missing Values for level

Obs	ID	Estimate	Standard Error	95% Confidence Limits	
1	1869	1098	130	843	1353
2	1870	1098	130	843	1353
53	1921	851	129	599	1104
103	1971	851	129	599	1104
104	1972	851	129	599	1104

Example 33.4: Longitudinal Data: Smoothing of Repeated Measures

This example of a repeated measures study is taken from Diggle, Liang, and Zeger (1994, p. 100). The data consist of body weights of 27 cows, measured at 23 unequally spaced time points over a period of approximately 22 months. Following Diggle, Liang, and Zeger (1994), one animal is removed from the analysis, one observation is removed according to their Figure 5.7, and the time is shifted to start at 0 and is measured in 10-day increments. The design is a 2×2 factorial, and the factors are the infection of an animal with M. paratuberculosis and whether the animal is receiving iron dosing. The data set contains five variables: cow assigns a unique identification number—from 1 to 26—to each cow in the study, tpoint denotes the time of the growth measurement, weight denotes the growth measurement, iron is a dummy variable that indicates

whether the animal is receiving iron or not, and infection is a dummy variable that indicates whether the animal is infected or not. The goal of the study is to assess the effect of iron and infection—and their possible interaction—on weight. The following DATA steps create this data set:

```
data times;
  input time1-time23;
  datalines;
 122  150  166  179  219  247  276  296  324  354  380  445
 478  508  536  569  599  627  655  668  723  751  781
;

data Cows;
  if _n_ = 1 then merge times;
  array t{23} time1   - time23;
  array w{23} weight1 - weight23;
  input cow iron infection weight1-weight23 @@;
  do i=1 to 23;
     weight = w{i};
     tpoint = (t{i}-t{1})/10;
     output;
  end;
  keep cow iron infection tpoint weight;
  datalines;
  1 0 0  4.7    4.905  5.011  5.075  5.136  5.165  5.298  5.323
         5.416  5.438  5.541  5.652  5.687  5.737  5.814  5.799

    ... more lines ...
```

The following DATA step adds ironInf, a grouping variable that is used later during the plotting of the results. In the next step, the data are sorted by the index variable, tpoint.

```
data Cows;
   set Cows;
   ironInf = "No Iron and No Infection";
   if iron=1 and infection=1 then ironInf = "Iron and Infection";
   else if iron=1 and infection=0 then ironInf = "Iron and No Infection";
   else if iron=0 and infection=1 then ironInf = "No Iron and Infection";
   else ironInf = "No Iron and No Infection";
   run;

 proc sort data=Cows;
     by tpoint ;
 run;
```

To assess the effect of iron and infection on weight, the natural growth profile of the animals must also be accounted for. Here two alternate models for this problem are considered. The first model assumes that the observed weight of an animal is the sum of a common growth profile, which is modeled by a polynomial spline trend of order 2, the regression effects of iron and infection, and the observation error—modeled as white noise. An interaction term, for interaction between iron and infection, was found to be insignificant and is not included. In the second model, the common growth profile and the regression variables of the first model are replaced by four environment specific growth profiles.

The following statements fit the first model:

```
proc ssm data=Cows;
   id tpoint;
   trend growth(ps(2));
   irregular wn;
   model weight = iron infection growth wn;
   eval pattern = iron + infection + growth;
   output out=For;
quit;
```

Output 33.4.1 shows that the state dimension of this model is 2 (corresponding to the polynomial trend specification of order 2), the number of diffuse elements in the initial condition is 4 (corresponding to the trend and the two regressors iron and infection), and the number of unknown parameters is 2 (corresponding to the variance parameters of trend and irregular).

Output 33.4.1 Model1: Model Summary Information

The SSM Procedure

Model Summary

Model Property	Value
Number of Model Equations	1
State Dimension	2
Dimension of the Diffuse Initial Condition	4
Number of Parameters	2

Output 33.4.2 shows that the ID variable is irregularly spaced with replication.

Output 33.4.2 ID Variable Information

ID Variable Information

Name	Start	End	Max Delta	NDistinct	Type
tpoint	0	65.9	6.5	23	Irregular with Replication

The estimated regression coefficients of iron and infection, shown in Output 33.4.3, are significant and negative. This implies that both iron and infection adversely affect the response variable, weight.

Output 33.4.3 Model 1: Regression Estimates

Regression Parameter Estimates

Response Variable	Regression Variable	Estimate	Standard Error	t Value	Pr > |t|
weight	iron	-0.0748	0.00761	-9.82	<.0001
weight	infection	-0.1292	0.00859	-15.04	<.0001

The variance estimates of the trend component and the irregular component are shown in Output 33.4.4.

Output 33.4.4 Model 1: Estimates of Unnamed Parameters

Model Parameter Estimates

Component	Type	Parameter	Estimate	Standard Error	t Value
growth	PS(2) Trend	Level Variance	0.0000162	9.01E-06	1.80
wn	Irregular	Variance	0.0085849	5.03E-04	17.06

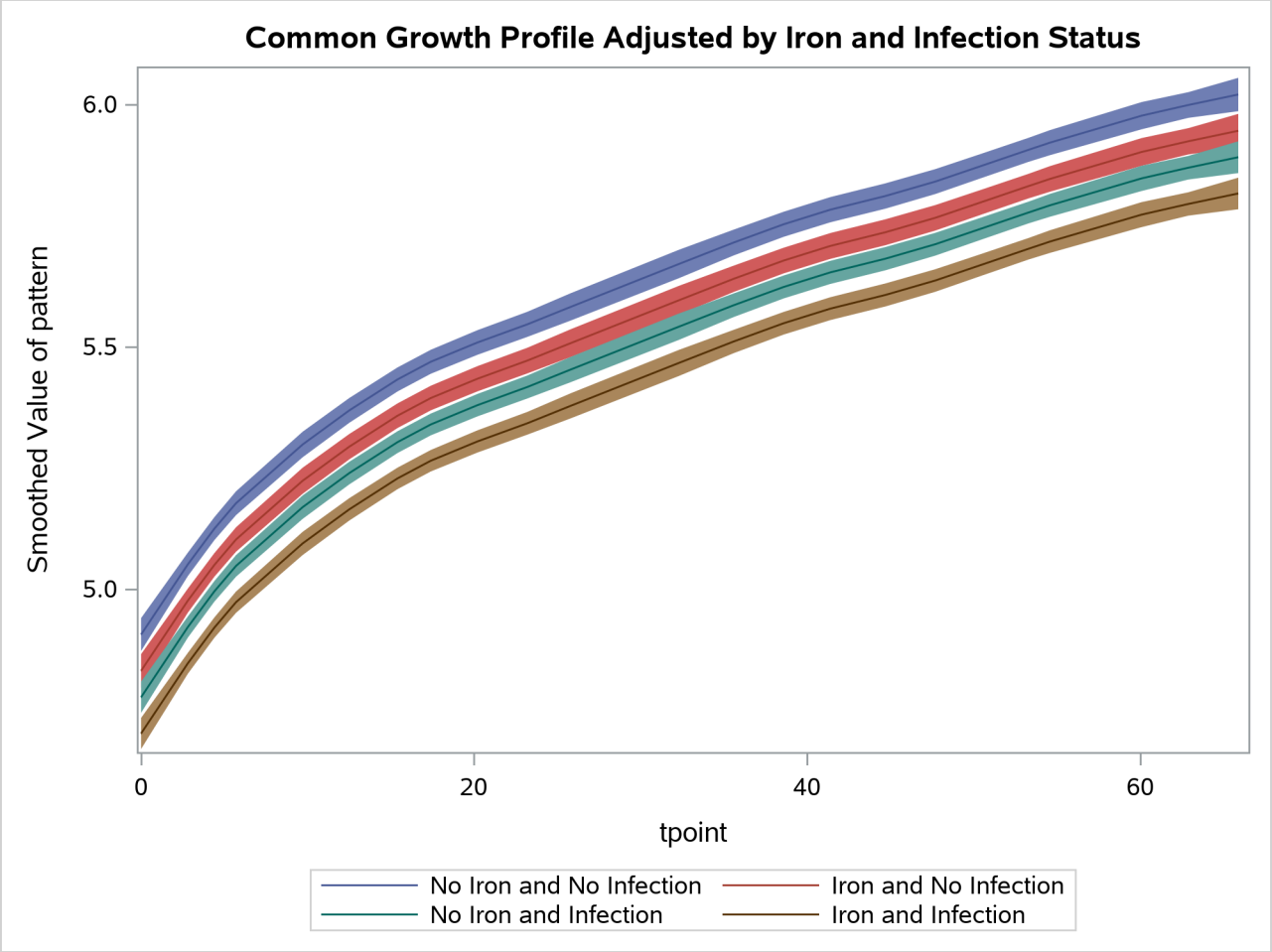
After examining the model fit, it is useful to study how well the patterns implied by the model follow the data. pattern, defined by the EVAL statement, is a sum of the trend component and the regression effects. A graphical examination of the smoothed estimate of pattern is done next. The following DATA step merges the output data set specified in the OUTPUT statement, For, with the input data set, Cows. In particular, this adds ironInf (a grouping variable from Cows) to For.

```
data For;
    merge for Cows;
    by tpoint;
run;
```

The following statements produce the graphs of smoothed_pattern, grouped according to the environment condition (see Output 33.4.5). The plot clearly shows that the control group "No Iron and No Infection" has the best growth profile, while the worst growth profile is for the group "Iron and Infection."

```
proc sgplot data=For noautolegend;
    title 'Common Growth Profile Adjusted by Iron and Infection Status';
    band x=tpoint lower=smoothed_lower_pattern
       upper=smoothed_upper_pattern / group=ironInf name="band";
    series  x=tpoint y=smoothed_pattern / group=ironInf name="series";
    keylegend "series";
run;
```

Output 33.4.5 Model 1: Growth Profile Comparison with 95% Confidence Bands

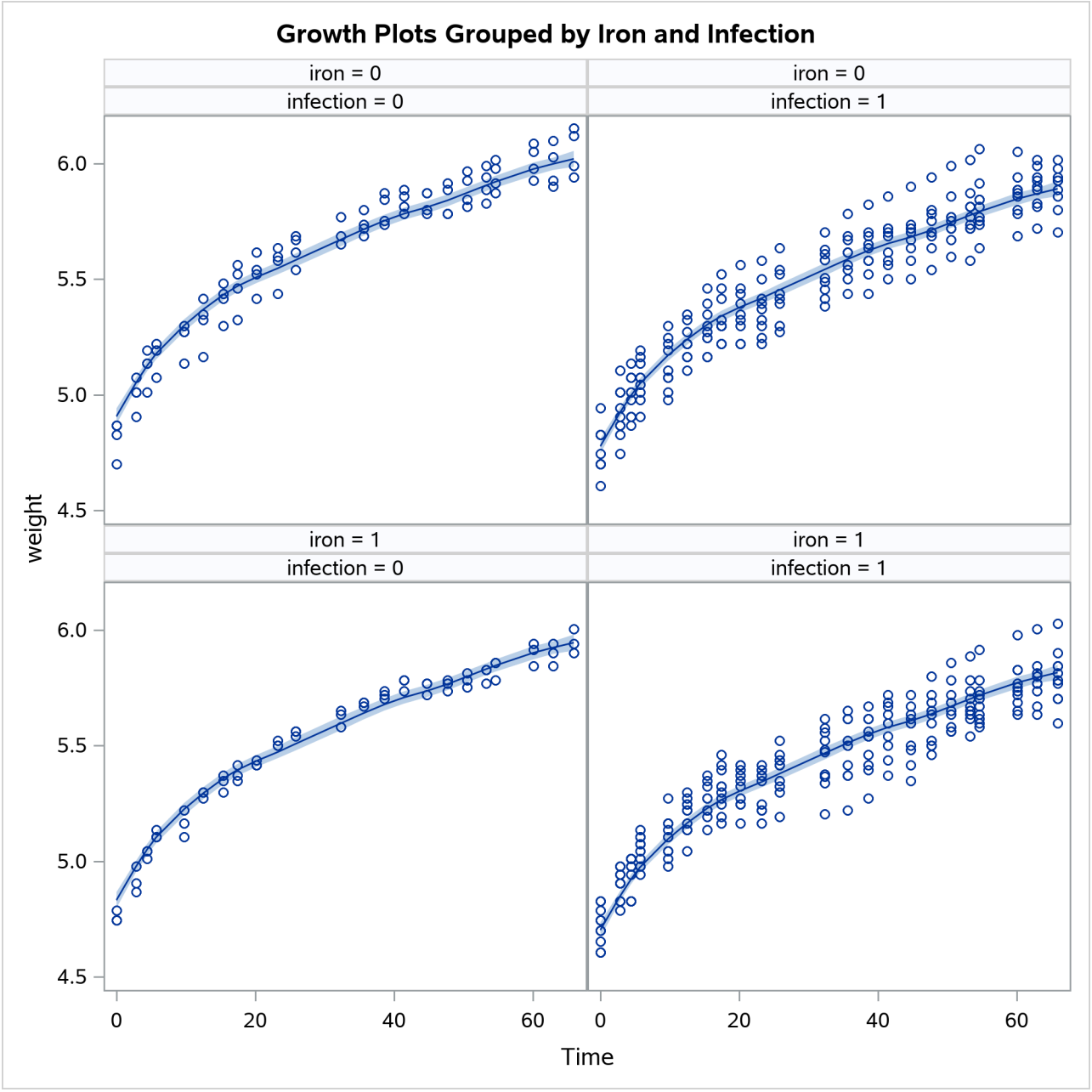


The following statements produce a panel of plots that show how well smoothed_pattern follows the observed data:

```
proc sgpanel data=For noautolegend;
  title 'Growth Plots Grouped by Iron and Infection';
   label tpoint='Time' ;
   panelby iron infection / columns=2;
   band x=tpoint lower=smoothed_lower_pattern
      upper=smoothed_upper_pattern ;
   scatter x=tpoint y=weight;
   series  x=tpoint y=smoothed_pattern ;
run;
```

Output 33.4.6 shows that the model fits the data reasonably well.

Output 33.4.6 Model 1: Smoothed Model Fit Lines



The following statements fit the second model. In this model separate polynomial trends are fit according to different settings of iron and infection by specifying an appropriate list of (dummy) variables in the CROSS= option of the trend specification.

```
proc ssm data=Cows;
    id tpoint;
    a1 = (iron=1 and infection=1);
    a2 = (iron=1 and infection=0);
```

```
    a3 = (iron=0 and infection=1);
    a4 = (iron=0 and infection=0);
    trend growth(ps(2)) cross=(a1-a4);
    irregular wn;
    model weight = growth wn;
    /* Define contrasts between a1 and other treatments  */
    comp a1Curve = growth_state_[1];
    comp a2Curve = growth_state_[2];
    comp a3Curve = growth_state_[3];
    comp a4Curve = growth_state_[4];
    eval contrast21 = a2Curve - a1Curve;
    eval contrast31 = a3Curve - a1Curve;
    eval contrast41 = a4Curve - a1Curve;
    output out=for1;
quit;
```

As a result of the CROSS= option, the trend component growth is actually a sum of four separate trends that correspond to the different iron-infection settings. Denoting growth by μ_t and the four independent trends by $\mu_{1,t}$, $\mu_{2,t}$, $\mu_{3,t}$, and $\mu_{4,t}$,

$$\mu_t = a1 * \mu_{1,t} + a2 * \mu_{2,t} + a3 * \mu_{3,t} + a4 * \mu_{4,t}$$

where a1, a2, a3, and a4 are the dummy variables specified in the CROSS= option. This shows that, for any given setting (say, the one for a4) μ_t is simply the corresponding trend $\mu_{4,t}$. In addition, note the form of the COMPONENT statements that define the components a1Curve, a2Curve, a3Curve, and a4Curve. This form of the COMPONENT statement treats the state that is associated with growth, named growth_state_ by convention, as a state of nominal dimension 4—the number of variables in the CROSS= list. This, in turn, implies that a1Curve, which is defined as growth_state_[1], refers to $\mu_{1,t}$. These components are subsequently used in the EVAL statements to define contrasts between the trends—for example, contrast21 corresponds to the difference between the trends $\mu_{2,t}$ and $\mu_{1,t}$. The estimates of these components (a1Curve, a2Curve, ..., contrast41) are output to the data set For1 named in the OUT= option of the OUTPUT data set.

The model summary, shown in Output 33.4.7, reflects the increased state dimension and the increased number of parameters.

Output 33.4.7 Model2: Model Summary Information

The SSM Procedure

Model Summary	
Model Property	**Value**
Number of Model Equations	1
State Dimension	8
Dimension of the Diffuse Initial Condition	8
Number of Parameters	5

Output 33.4.8 shows the parameter estimates for this model.

Output 33.4.8 Model2: Estimates of Unnamed Parameters (Partial Output)

Component	Parameter	Estimate	StdErr	tValue
growth(Cross = a1)	Level Variance	1.28E-05	6.83E-06	1.87
growth(Cross = a2)	Level Variance	8.72E-06	3.81E-06	2.29
growth(Cross = a3)	Level Variance	9.07E-06	4.23E-06	2.14
growth(Cross = a4)	Level Variance	8.45E-06	3.40E-06	2.49
wn	Variance	8.39E-03	4.98E-04	16.84

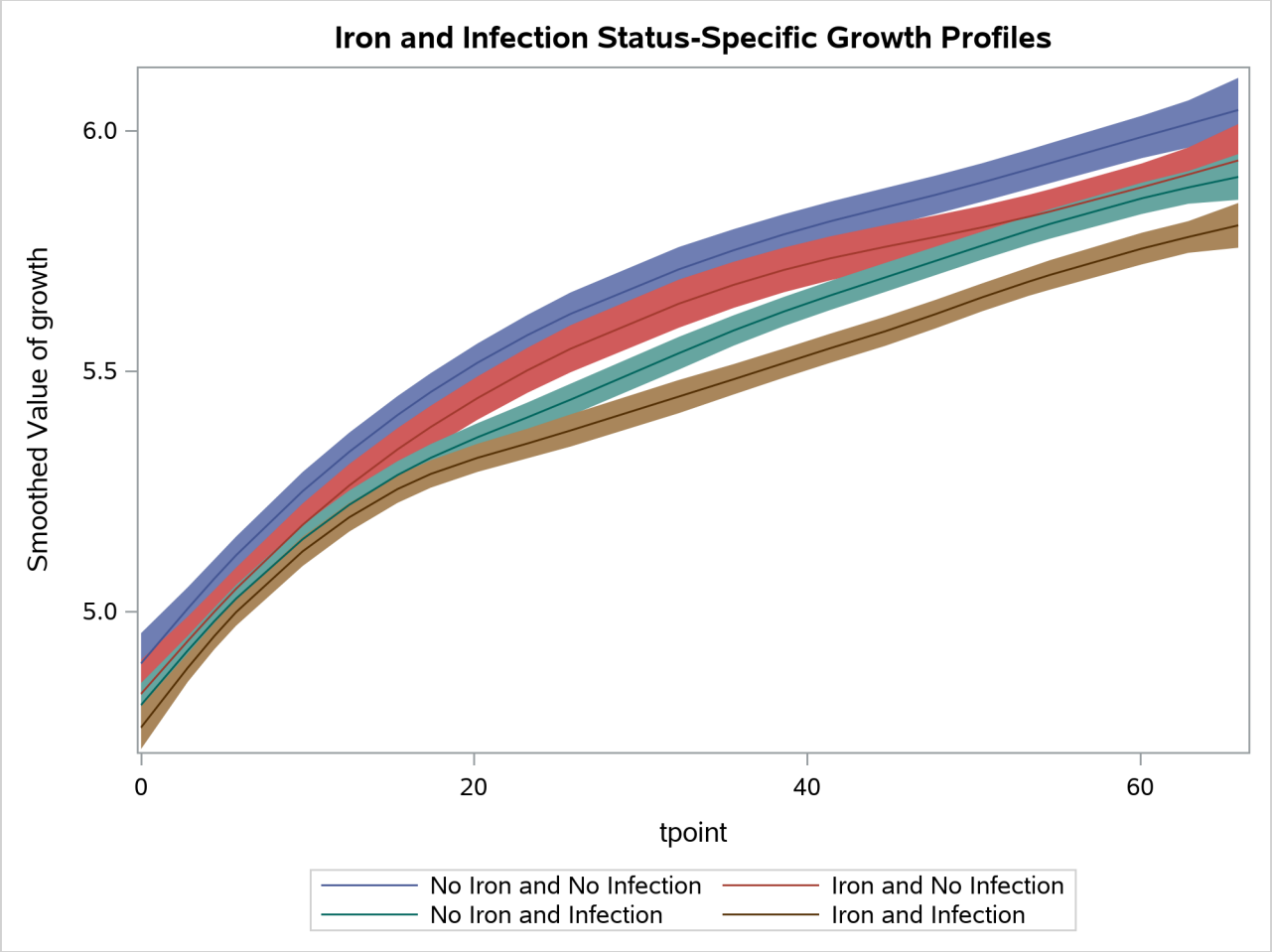
Next, the smoothed estimate of trend (growth) is graphically studied. The following DATA step prepares the data for the grouped plots of smoothed_growth by merging For1 with the input data set Cows. As before, the reason is merely to include ironInf (the grouping variable).

```
data For1;
    merge For1 Cows;
    by tpoint;
run;
```

The following statements produce the graphs of smoothed μ_t for the desired settings (since the grouping variable ironInf exactly corresponds to these settings). Once again, the plot in Output 33.4.9 clearly shows that the control group "No Iron and No Infection" has the best growth profile, while the worst growth profile is for the group "Iron and Infection." However, unlike the first model, the profile curves are not merely shifted versions of a common profile.

```
proc sgplot data=For1 noautolegend;
   title 'Iron and Infection Status-Specific Growth Profiles';
    band x=tpoint lower=smoothed_lower_growth
      upper=smoothed_upper_growth / group=ironInf name="band";
    series  x=tpoint y=smoothed_growth / group=ironInf name="series";
    keylegend "series";
run;
```

Output 33.4.9 Model 2: Growth Profile Comparison with 95% Confidence Bands

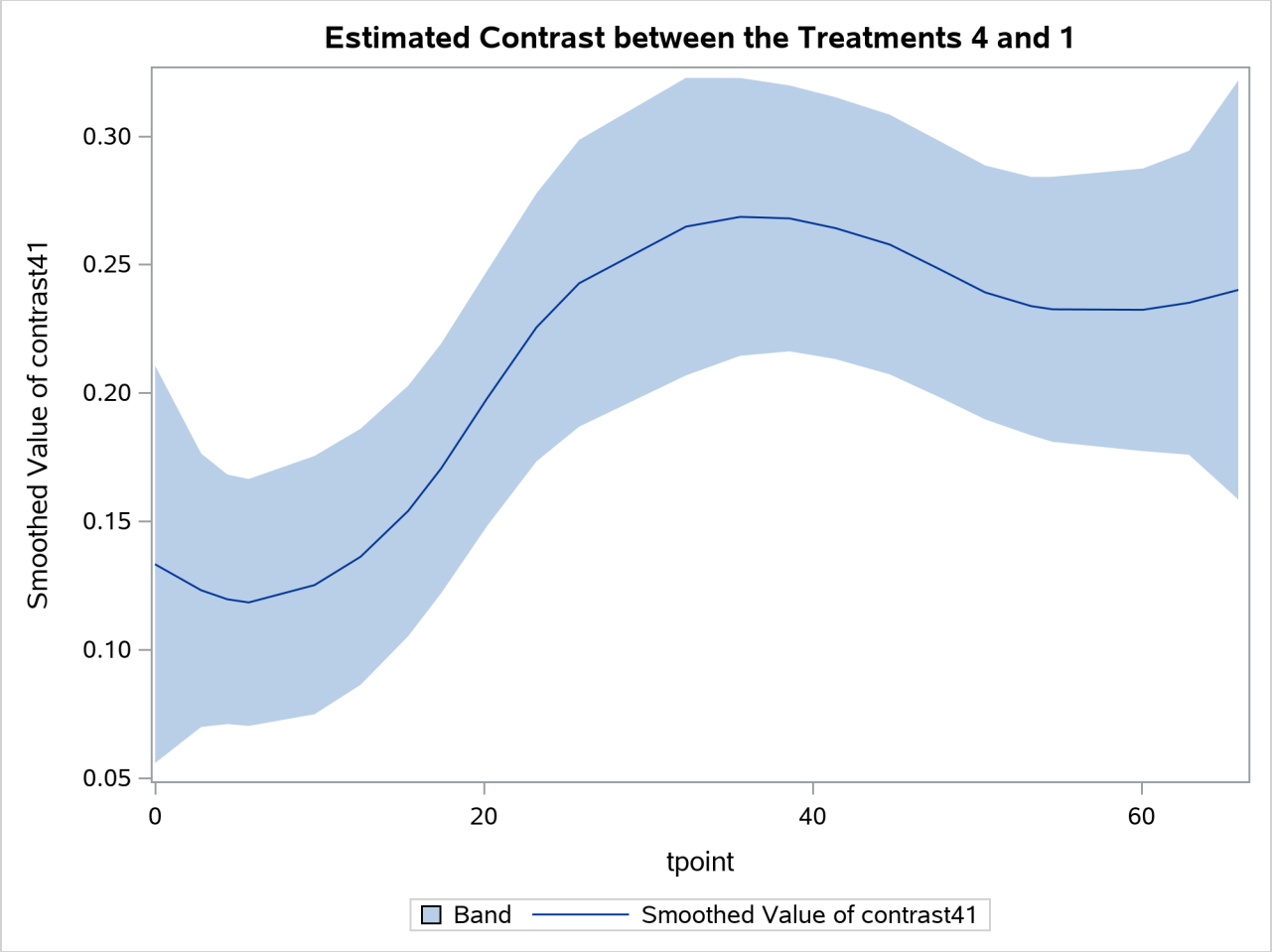


The following statements produce the plot of smoothed $(\mu_{4,t} - \mu_{1,t})$—contrast between the best and the worst growth profiles:

```
proc sgplot data=For1;
   title "Estimated Contrast between the Treatments 4 and 1 ";
    band x=tpoint lower=smoothed_lower_contrast41
       upper=smoothed_upper_contrast41;
    series x=tpoint y=smoothed_contrast41;
run;
```

Output 33.4.10 shows that the growth pattern of the control group "No Iron and No Infection" consistently remains above the growth pattern of the treatment group "Iron and Infection."

Output 33.4.10 Estimated Contrast between the Treatments 4 and 1 with 95% Confidence Bands



Example 33.5: A User-Defined Trend Model

This example shows how to specify a continuous-time trend model discussed in Harvey (1989, chap. 9, sec. 9.2.1). This model is not one of the predefined trend models in the SSM procedure. The system matrices that govern the two-dimensional state of this model are

$$\mathbf{T} = \begin{bmatrix} 1 & h \\ 0 & 1 \end{bmatrix}$$

$$\mathbf{Q} = \begin{bmatrix} h\sigma_1^2 + \frac{h^3\sigma_2^2}{3} & \frac{h^2\sigma_2^2}{2} \\ \frac{h^2\sigma_2^2}{2} & h\sigma_2^2 \end{bmatrix}$$

where $h = h_t = (\tau_{t+1} - \tau_t)$ denotes the difference between the successive time points, and the parameters σ_1^2 and σ_2^2 are called the level variance and the slope variance, respectively. The initial condition is fully diffuse. The trend component corresponds to the first element of this state vector. The second element of the state vector corresponds to the slope of this trend component. This model reduces to the polynomial spline model of order 2 if the level variance $\sigma_1^2 = 0$. (See the section "Polynomial Spline Trend" on page 2444.)

The following statements specify a trend-plus-noise model to model the growth of cows in the previous example (Example 33.4). The only cows that are considered are the ones that received iron and are infected.

```
proc ssm data=Cows;
   where iron=1 and infection=1;
   id tpoint;
   parms var1 var2 / lower=(1.e-8 1.e-8);
   array tMat{2,2};
   tMat[1,1] = 1;
   tMat[2,2] = 1;
   tMat[1,2] = _ID_DELTA_;
   array covMat{2,2};
   covMat[1,1] = var1*_ID_DELTA_ + var2*_ID_DELTA_**3/3;
   covMat[1,2] = var2*_ID_DELTA_**2/2;
   covMat[2,1] = covMat[1,2] ;
   covMat[2,2] = var2*_ID_DELTA_;
   state harveyLL(2) T(g)=(tMat) cov(g)=(covMat) a1(2);
   component trend = harveyLL[1];
   component slope = harveyLL[2];
   irregular wn;
   model  weight = trend  wn;
   output out=for;
run;
```

The program is easy to follow. The PARMS statement declares var1 and var2 as positive parameters, which correspond to σ_1^2 and σ_2^2, respectively. The programming statements define arrays tMat and covMat, which later become the matrices $\mathbf{T}$ and $\mathbf{Q}$, respectively. Note that the element tMat[2,1] is left unassigned, since it is a structural zero of $\mathbf{T}$ (see the section "Sparse Transition Matrix Specification" on page 2431 for more information). Recall that the predefined variable _ID_DELTA_ contains the value of h_t, which is needed for defining the elements of $\mathbf{T}$ and $\mathbf{Q}$ (see the section "ID Statement" on page 2412). The STATE statement defines the trend state vector, harveyLL, and the COMPONENT statement defines the trend component, trend, by selecting the first element of harveyLL. An additional COMPONENT statement defines the slope component, slope, as the second element of harveyLL. The slope component (which represents the cow's growth rate) is not part of the observation equation; it is specified so that its estimate is output to For (the OUT= data set specified in the OUTPUT statement). The IRREGULAR statement defines the observation noise, and the MODEL statement defines the trend-plus-noise model.

The estimates of var1 and var2 are shown in Output 33.5.1. It shows that the estimate of the level variance is nearly 0, implying that the fitted trend model is identical to the polynomial spline trend of order 2.

Output 33.5.1 Estimates of the Named Parameters

The SSM Procedure

Estimates of Named Parameters

Parameter	Estimate	Standard Error	t Value
var1	1.00E-08	0.000849	0.00
var2	1.24E-05	.	.

The estimate of the noise variance is shown in Output 33.5.2.

Output 33.5.2 Estimates of the Unnamed Parameters

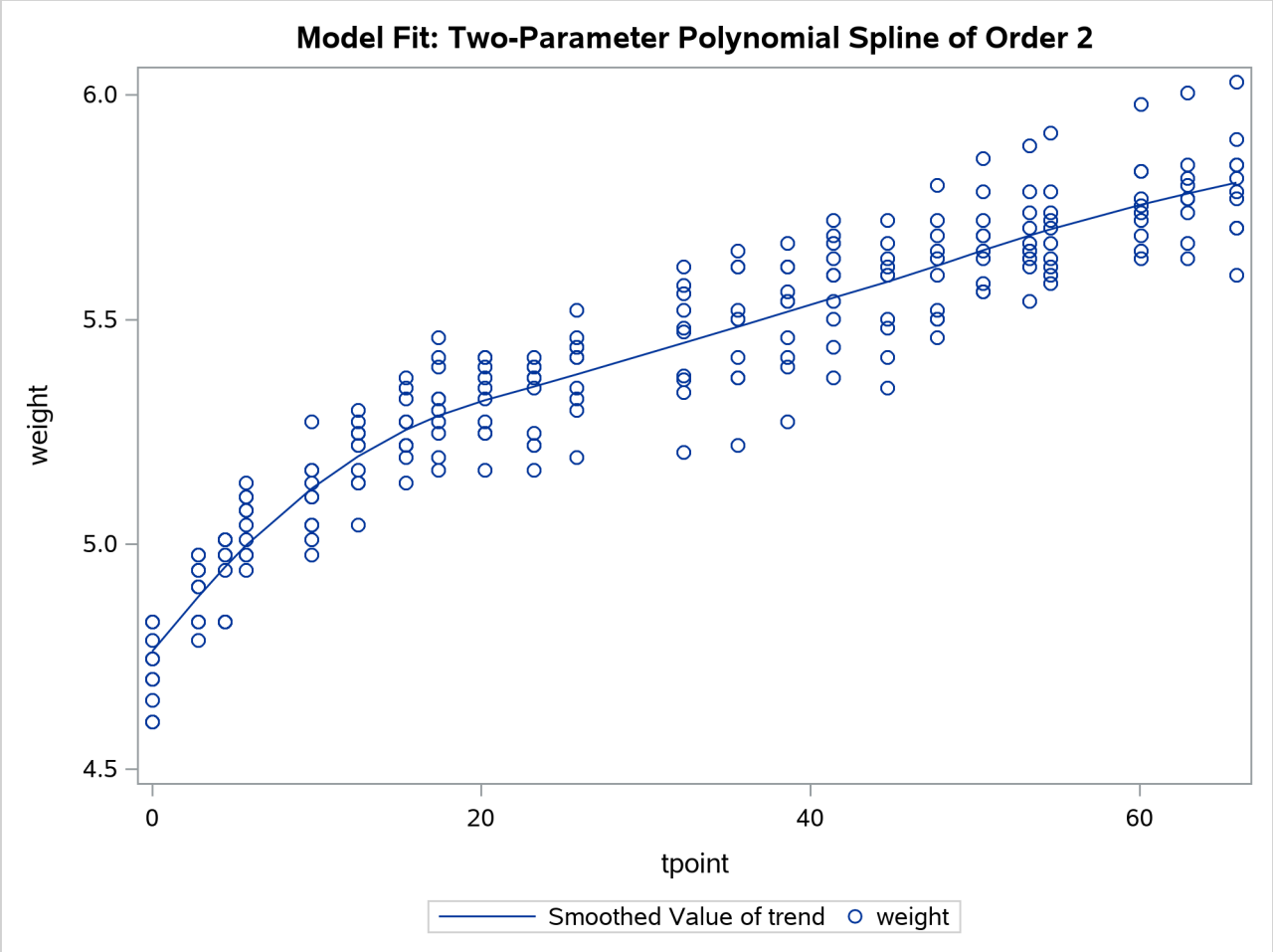
Model Parameter Estimates

Component	Type	Parameter	Estimate	Standard Error	t Value
wn	Irregular	Variance	0.00954	0.000909	10.49

The following statements produce the plot of the fit of this trend model (shown in Output 33.5.3):

```
proc sgplot data=For;
    title "Model Fit: Two-Parameter Polynomial Spline of Order 2";
    series x=tpoint y=smoothed_trend;
    scatter x=tpoint y=weight;
run;
```

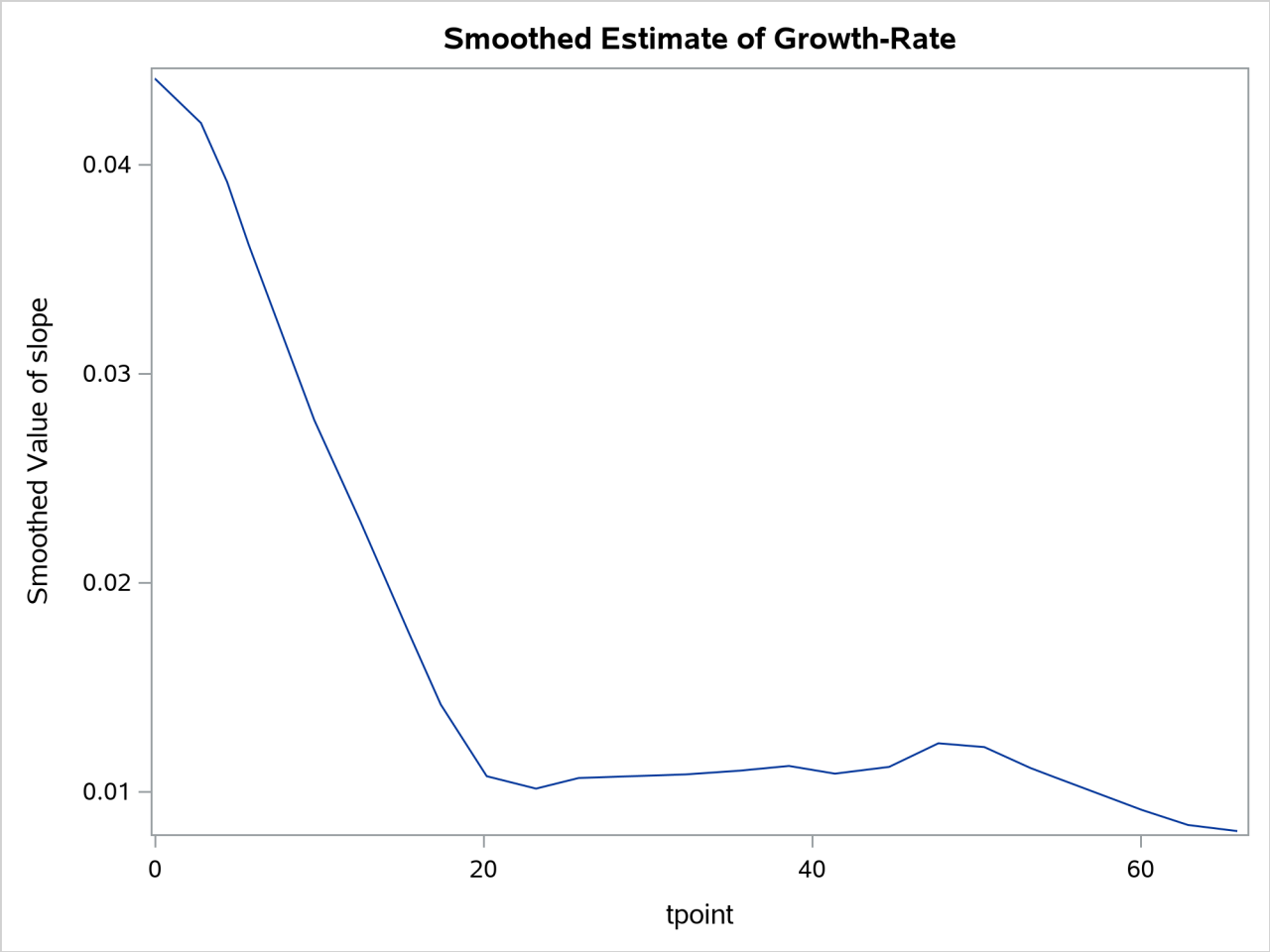
Output 33.5.3 A User-Defined Trend Model



The following statements produce the plot of the estimate of the slope component (shown in Output 33.5.4). This plot complements the preceding plot of trend; it shows the pattern of decline in the growth rate as the animals age.

```
proc sgplot data=For;
    title "Smoothed Estimate of Growth-Rate";
    series x=tpoint y=smoothed_slope;
run;
```

Output 33.5.4 Estimate of the slope Component



Example 33.6: Model with Multiple ARIMA Components

This example shows how you can fit the REGCOMPONENT models in Bell (2011) by using the SSM procedure. The following DATA step generates the data used in the last example of this article (Example 6: "Modeling a Time Series with a Sampling Error Component"). The variable y in this data set contains monthly values of the VIP series (value of construction put in place), a US Census Bureau publication that measures the value of construction installed or erected at construction sites during a given month. The values of y are known to be contaminated with heterogeneous sampling errors; the variable hwt in the data set is a proxy for this sampling error in the log scale. The variable hwt is treated as a weight variable for the noise component in the model.

```
data Test;
  input y hwt;
  date = intnx('month', '01jan1997'd, _n_-1 );
  format date date.;
  logy = log(y);
  label logy = 'Log value of construction put in place';
  datalines;
115.2    0.042
110.4    0.042
111.5    0.067
127.9    0.122
150.0    0.129
149.5    0.135
139.5    0.152
144.6    0.168
176.0    0.173

   ... more lines ...
```

The article proposes the following model for the log VIP series:

$$\log(y) = \mu_t + hwt * \eta_t$$

where μ_t follows an ARIMA$(0,1,1)\times(0,1,1)_{12}$ model and η_t is a zero-mean, AR(2) error process. In addition, the article fixes the values of some of the model parameters to known values in order to use the known background information. The following statements specify the model in the article:

```
proc ssm data=Test;
   id date interval=month;
   parm var1=0.016565 / lower=1.e-8;
   trend airlineTrend(arma(d=1 sd=1 q=1 sq=1 s=12)) variance=var1;
   trend ar2Noise(arma(p=2)) cross=(hwt) ar=0.600 0.246 variance=0.34488;
   model logy = airlineTrend ar2Noise;
   output outfor=For;
run;
```

Output 33.6.1 Estimates of the MA Parameters in the airlineTrend Model

The SSM Procedure

Model Parameter Estimates

Component	Type	Parameter	Estimate	Standard Error	t Value
airlineTrend	ARMA Trend	MA_1	0.421	0.301	1.40
airlineTrend	ARMA Trend	SMA_1	0.310	0.347	0.89

Output 33.6.2 Estimate of the Error Variance in the airlineTrend Model

Estimates of Named Parameters

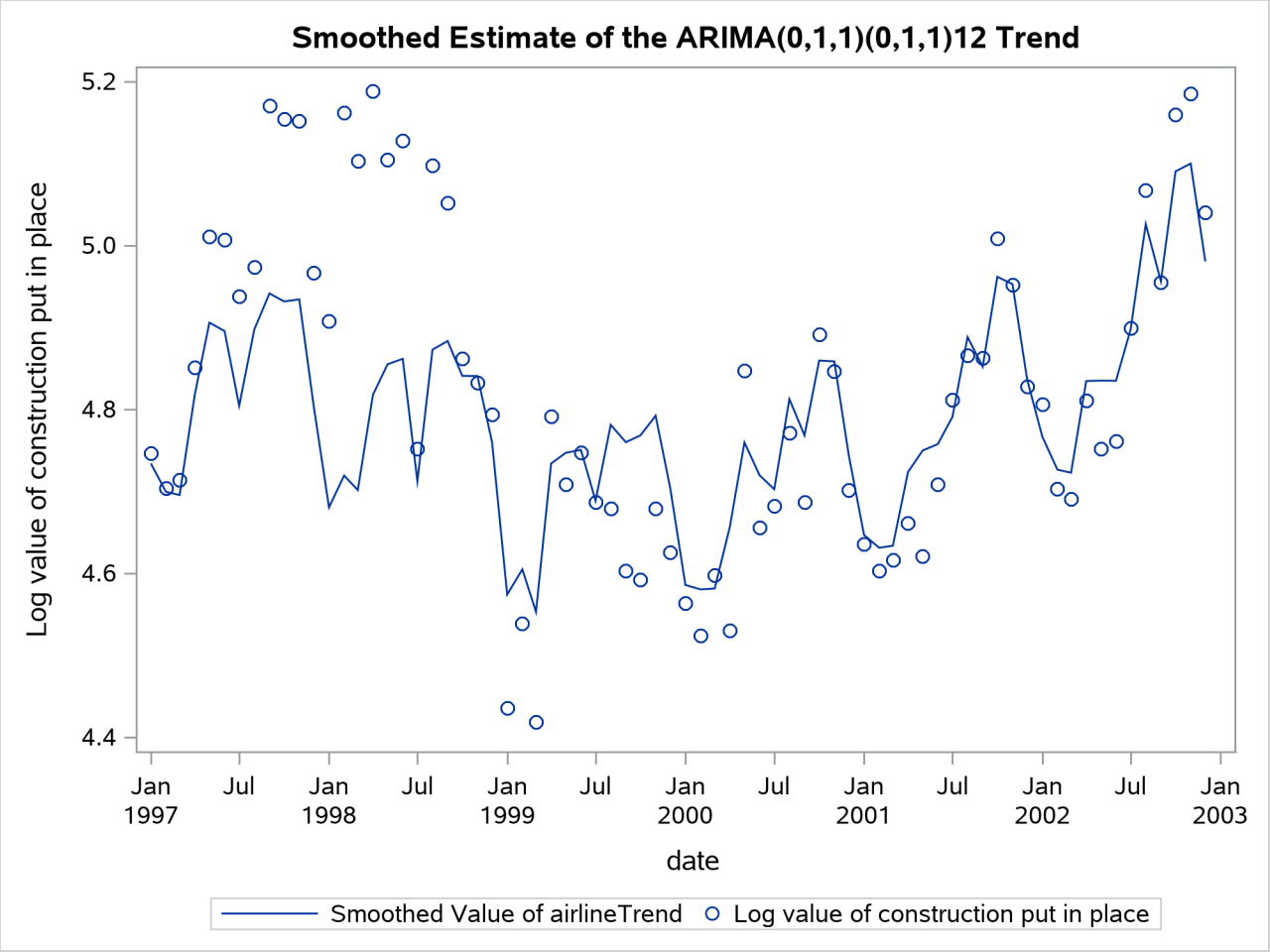
Parameter	Estimate	Standard Error	t Value
var1	0.004	0.00222	1.80

The ARIMA$(0,1,1)\times(0,1,1)_{12}$ trend μ_t is named airlineTrend and the zero-mean, AR(2) error process η_t is named ar2Noise. For more information about the ARIMA notation, see the TREND statement. The estimates of model parameters are shown in Output 33.6.1 and Output 33.6.2. These estimates are slightly different from the estimates given in the article; however, the estimated trend and noise series are qualitatively similar.

The following statements produce the plot of the estimate of the airlineTrend component (shown in Output 33.6.3). This plot is very similar to the trend plot shown in the article (the article plots are in the antilog scale).

```
proc sgplot data=For;
    title "Smoothed Estimate of the ARIMA(0,1,1)(0,1,1)12 Trend";
    series x= date y=smoothed_airlineTrend;
    scatter x= date y=logy;
run;
```

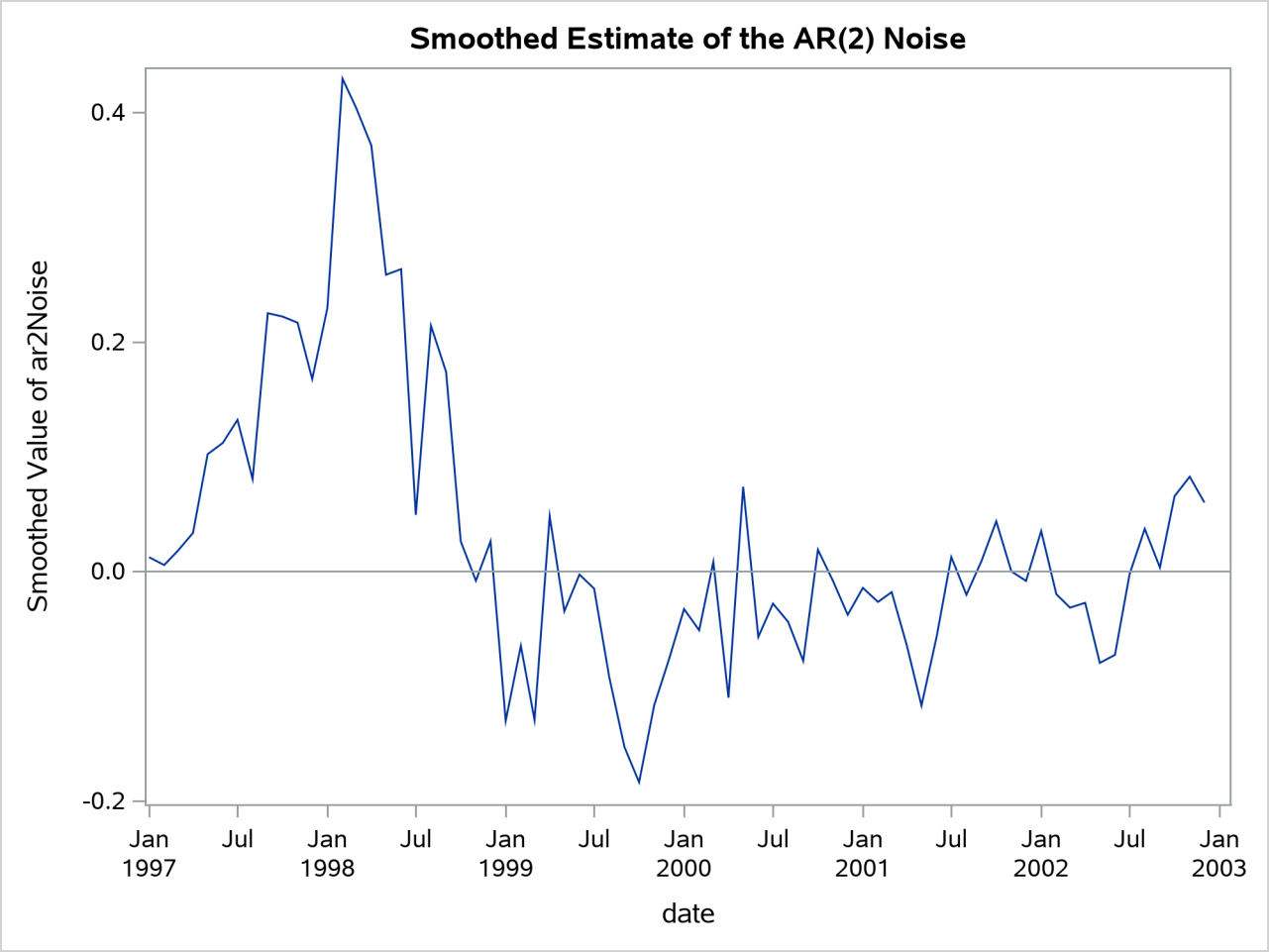
Output 33.6.3 Estimate of the airlineTrend Component



The following statements produce the plot of the estimate of the ar2Noise component (shown in Output 33.6.4). This plot is also very similar to the noise plot shown in the article (once again, the article plots are in the antilog scale).

```
proc sgplot data=For;
    title "Smoothed Estimate of the AR(2) Noise";
    series x= date y=smoothed_ar2Noise;
    refline 0;
run;
```

Output 33.6.4 Estimate of the ar2Noise Component



Example 33.7: A Dynamic Factor Model for the Yield Curve

This example shows how you can fit a variant of the dynamic Nelson-Siegel (DNS) factor model discussed in Koopman, Mallee, and van der Wel (2010). Also see the example in Durbin and Koopman (2012, chap. 8, sect. 6). The following DATA step creates the yield-curve data set, Dns, that is used in Koopman, Mallee, and van der Wel. The data are monthly bond yields that were recorded between the start of 1970 and the end of 2000 for 17 bonds of different maturities; the maturities range from three months to 10 years (120 months). The variable date contains the observation date, yield contains the bond yield, maturity contains the associated bond maturity, and mtype contains an index (ranging from 1 to 17) that sequentially labels bonds of increasing maturity. The data have been extended for two more years by adding missing yields for the years 2001 and 2002, which causes the SSM procedure to produce model forecasts for this span.

```
data Dns;
input date : date. yield maturity mtype;
format date date.;
datalines;
1-Jan-70    8.019    3    1
1-Jan-70    8.091    6    2
1-Jan-70    8.108    9    3

   ... more lines ...
```

In addition, suppose you are interested in extrapolating the fitted model to predict the yield of a hypothetical bond that has a maturity of 42 months and is not traded on the general exchange. The following DATA step creates the necessary missing values for this new bond, which is assigned the index of 18—that is, the value of mtype is 18:

```
data tmp1;
   set dns(keep=date);
   by date;
   if first.date then do;
       yield = .;
       maturity = 42;
       mtype = 18;
       output;
   end;
run;

 proc append data=tmp1 base=dns; run;
 proc sort data=dns;
 by date;
 run;
```

Suppose that $\theta_t(\tau)$ denotes the (idealized) yield at time t that is associated with a bond of maturity τ (in months). Even if time is not measured continuously and the bonds of only certain maturities are traded, $\theta_t(\tau)$ is treated as a smooth function of two continuous variables, time t and maturity τ. Koopman, Mallee, and van der Wel (2010) discuss a variety of models for $\theta_t(\tau)$, which is called the yield surface. One of these models depends on a positive, time-varying, scalar parameter λ_t and a time-varying three-dimensional vector parameter $\boldsymbol{\beta}_t$. This model can be described as follows:

$$\begin{aligned}\theta_t(\tau) &= \theta(\tau;\lambda_t,\boldsymbol{\beta}_t)\\ &= \beta_{1t} + \beta_{2t}\left(\frac{1-\exp(-\lambda_t\tau)}{\lambda_t\tau}\right) + \beta_{3t}\left(\frac{1-\exp(-\lambda_t\tau)}{\lambda_t\tau} - \exp(-\lambda_t\tau)\right)\end{aligned}$$

This model is a dynamic version of a static model discussed in Nelson and Siegel (1987), where λ_t and $\boldsymbol{\beta}_t$ are time invariant. For fixed time period t, the three terms in this model have relatively simple interpretation. The first term β_{1t} can be thought of as the overall yield level because it does not depend on τ, the bond maturity. It can also be thought of as the long term yield because as $\tau \uparrow \infty$ the other two terms vanish; the coefficients of both β_{2t} and β_{3t} converge to 0 as $\tau \uparrow \infty$ (recall that λ_t is positive). Next, note that as $\tau \downarrow 0$ the coefficient of β_{2t} in the second term converges to 1 while that of β_{3t} in the third term converges to 0; therefore the second term can be thought of as a correction to the overall yield that is associated with the short term bonds. Finally, note that the coefficient of β_{3t} in the third term is a unimodal function of τ that decays

monotonically to 0 as $\tau \downarrow 0$ and as $\tau \uparrow \infty$; therefore the third term is associated with the medium term bond yields. It is postulated that the observed yield, denoted by $y_t(\tau)$, is a noisy version of this unobserved (true) yield $\theta_t(\tau)$. The observed yield can be modeled as

$$\begin{aligned} y_t(\tau) &= \theta(\tau; \lambda_t, \boldsymbol{\beta}_t) + \epsilon_{t,\tau} \\ &= \beta_{1t} + \beta_{2t}\left(\frac{1-\exp(-\lambda_t\tau)}{\lambda_t\tau}\right) + \beta_{3t}\left(\frac{1-\exp(-\lambda_t\tau)}{\lambda_t\tau} - \exp(-\lambda_t\tau)\right) + \epsilon_{t,\tau} \\ (\boldsymbol{\beta}_t - \boldsymbol{\mu}) &= \boldsymbol{\Phi}(\boldsymbol{\beta}_{t-1} - \boldsymbol{\mu}) + \boldsymbol{\eta}_t \end{aligned}$$

where $\epsilon_{t,\tau}$ are zero-mean, independent, Gaussian variables with variance σ_τ^2, and $\boldsymbol{\eta}_t$ is a three-dimensional, Gaussian white noise. That is, $\boldsymbol{\beta}_t$ is a VAR(1) process with mean vector $\boldsymbol{\mu}$. The remainder of this example explains how to use the SSM procedure to fit this model to the yield data in the Dns data set.

Suppose that variables Z1, Z2, and Z3 are defined as the coefficients of β_{1t}, β_{2t}, and β_{3t}, respectively. That is,

$$\begin{aligned} Z1 &= 1 \\ Z2 &= \frac{1-\exp(-\lambda_t\tau)}{\lambda_t\tau} \\ Z3 &= \frac{1-\exp(-\lambda_t\tau)}{\lambda_t\tau} - \exp(-\lambda_t\tau) \end{aligned}$$

In this case,

$$\theta_t(\tau) = Z1 * \beta_{1t} + Z2 * \beta_{2t} + Z3 * \beta_{3t}$$

Let $\boldsymbol{\zeta}_t = \boldsymbol{\beta}_t - \boldsymbol{\mu}$. Then $\boldsymbol{\zeta}_t$ is a zero-mean VAR(1) process and $\boldsymbol{\beta}_t = \boldsymbol{\zeta}_t + \boldsymbol{\mu}$. In particular,

$$\begin{aligned} \theta_t(\tau) &= Z1 * \beta_{1t} + Z2 * \beta_{2t} + Z3 * \beta_{3t} \\ &= Z1 * \zeta_{1t} + Z2 * \zeta_{2t} + Z3 * \zeta_{3t} + Z1 * \mu_1 + Z2 * \mu_2 + Z3 * \mu_3 \end{aligned}$$

This shows that the model for $y_t(\tau)$ can be cast into a state space form with the following observation equation:

$$y_t(\tau) = \mathbf{Z}\boldsymbol{\zeta}_t + \mathbf{Z}\boldsymbol{\mu} + \epsilon_{t,\tau}$$

The underlying six-dimensional state vector $\boldsymbol{\alpha}_t$ is formed by joining the two independent subvectors, $\boldsymbol{\zeta}_t$ (which is a zero-mean, VAR(1) process) and the constant mean vector $\boldsymbol{\mu}$. That is, $\boldsymbol{\alpha}_t = (\zeta_{1t}\ \zeta_{2t}\ \zeta_{3t}\ \mu_1\ \mu_2\ \mu_3)'$.

Note that the variables Z2 and Z3 depend on the time varying parameter λ_t, which is unknown. λ_t is assumed to be a smooth and positive function of time t. In what follows λ_t is represented as an exponential of a cubic spline—a B-spline— in time with four evenly spaced interior knots between January 1970 and December 2002. A cubic spline with four interior knots can be represented as a sum of seven (number of knots + spline degree + 1) B-spline basis functions, $c_{1t}, c_{2t}, \ldots, c_{7t}$, for example. More specifically, λ_t can be expressed as

$$\lambda_t = \exp(v1 * c_{1t} + \ldots + v7 * c_{7t})$$

for some parameters $v1, v2, \ldots, v7$ and the B-spline basis functions (of time) $c_{1t}, c_{2t}, \ldots, c_{7t}$. Thus, the variables Z2 and Z3 become known functions of time, except for the parameters $v1, v2, \ldots, v7$, which are estimated from the data. The following statements augment the Dns data set with the B-spline basis columns in two steps. First a data set that contains the basis columns, c1–c7, is created by using the BSPLINE function in the IML procedure. This data set is then merged with the Dns data set.

```
proc iml;
   use dns;
   read all var {date} into x;
   bsp = bspline(x, 2, ., 4);
   create spline var{c1 c2 c3 c4 c5 c6 c7};
   append from bsp;
quit;
data dns;
   merge dns spline;
run;
```

The following statements use the SSM procedure to perform the model fitting and forecasting calculations. The variance of the observation equation disturbance for the hypothetical bond (mtype = 18) is taken to be the average of the neighboring bonds (mtype = 10 and 11), whose maturities are 36 and 48 months, respectively.

```
 proc ssm data=Dns optimizer(technique=dbldog maxiter=400);
      id date interval=month;

     /* Time-varying parameter lambda */
     parms v1-v7;
     lambda = exp(v1*c1 + v2*c2 + v3*c3 + v4*c4
          + v5*c5 + v6*c6 + v7*c7);

     /* Observation equation disturbance -- separate variance for each maturity */
     parms sigma1-sigma17 / lower=1.e-4;
     array s_array(17) sigma1-sigma17;
      do i=1 to 17;
         if (mtype=i) then sigma = s_array[i];
      end;
      if (mtype=18) then sigma = (sigma10+sigma11)/2;
      irregular wn variance=sigma;

      /* Variables Z1, Z2, Z3 needed in the observation equation */
      Z1= 1.0;
      tmp = lambda*maturity;
      Z2 = (1-exp(-tmp))/tmp;
      Z3 = ( 1-exp(-tmp)-tmp*exp(-tmp) )/tmp;

       /* Zero-mean VAR(1) factor zeta and the associated component */
      state zeta(3) type=VARMA(p(d)=1) cov(g) print=(cov ar);
      comp zetaComp = (Z1-Z3)*zeta;

      /* Constant mean vector mu and the associated component */
       state mu(3) type=rw;
       comp muComp = (Z1-Z3)*mu;

       /* Observation equation */
       model yield = muComp zetaComp wn;

       /* Various components defined only for output purposes  */
       eval yieldSurface = muComp + zetaComp;

       comp zeta1 = zeta[1];
```

```
    comp zeta2 = zeta[2];
    comp zeta3 = zeta[3];
    comp mu1 = mu[1];
    comp mu2 = mu[2];
    comp mu3 = mu[3];

    comp z2zeta = (Z2)*zeta[2];
    comp z3zeta = (Z3)*zeta[3];
    comp z2Mu =  (Z2)*mu[2];
    comp z3Mu =  (Z3)*mu[3];

    eval beta1 = mu1 + zeta1;
    eval beta2 = mu2 + zeta2;
    eval beta3 = mu3 + zeta3;

    eval shortTem = z2zeta + z2Mu;
    eval medTerm = z3zeta + z3Mu;

    /* output the component estimates and the forecasts */
    output out=dnsFor pdv;
run;
```

The DBLDOG optimization technique is used for parameter estimation since it is computationally more efficient in this example. The transition matrix, $\mathbf{\Phi}$, in the VAR(1) specification of $\boldsymbol{zeta}$ is taken to be diagonal (TYPE=VARMA(P(D)=1)) because the use of more general square matrix did not improve the model fit significantly. The mean vector $\boldsymbol{mu}$ (recall that $\boldsymbol{beta}_t = \boldsymbol{zeta}_t + \boldsymbol{mu}$) is specified as a three-dimensional random walk with zero disturbance covariance (signified by the absence of COV= option). The model specification part of the program ends with the MODEL statement; the subsequent COMP and EVAL statements define some useful linear combinations of the underlying state. Their estimates are computed after the model fit is completed and are output to the output data set dnsFor. The dnsFor data set also contains all the program variables and the parameters defined in the PARMS statement because the OUTPUT statement contains the PDV option.

Output 33.7.1 shows the estimated mean vector ($\boldsymbol{\mu}$). It shows that the mean long-term yield is 7.64. Output 33.7.2 shows the estimates of v1–v7 (used for defining time-varying λ_t) and the maturity specific observation variances. Output 33.7.3 shows the estimate of the VAR(1) transition matrix $\mathbf{\Phi}$, and Output 33.7.4 shows the associated disturbance covariance matrix $\mathbf{\Sigma}$. The model fit summary is shown in Output 33.7.5.

Output 33.7.1 Estimate of the Mean Vector ($\boldsymbol{\mu}$)

The SSM Procedure

Estimates of Fixed State Effects

State	Element Index	Estimate	Standard Error	t Value	Pr > |t|
mu	1	7.638	1.357	5.63	<.0001
mu	2	-1.319	0.777	-1.70	0.0897
mu	3	-0.309	0.268	-1.15	0.2481

Output 33.7.2 Estimates of v1–v7 and Observation Variances

Estimates of Named Parameters			
Parameter	Estimate	Standard Error	t Value
v1	-1.19585	0.303997	-3.93
v2	-2.93680	0.111439	-26.35
v3	-1.88701	0.068966	-27.36
v4	-2.31369	0.079110	-29.25
v5	-3.21865	0.105563	-30.49
v6	-1.66086	0.315651	-5.26
v7	-4.60100	1.547921	-2.97
sigma1	0.05405	0.004706	11.49
sigma2	0.00349	0.000865	4.03
sigma3	0.00869	0.000752	11.56
sigma4	0.01093	0.000900	12.14
sigma5	0.00865	0.000757	11.43
sigma6	0.00603	0.000571	10.56
sigma7	0.00519	0.000491	10.58
sigma8	0.00542	0.000497	10.90
sigma9	0.00562	0.000500	11.25
sigma10	0.00639	0.000559	11.43
sigma11	0.01032	0.000848	12.17
sigma12	0.00742	0.000676	10.98
sigma13	0.01106	0.000947	11.68
sigma14	0.01194	0.001052	11.36
sigma15	0.01244	0.001163	10.70
sigma16	0.02141	0.001843	11.62
sigma17	0.02747	0.002296	11.97

Output 33.7.3 Transition Matrix, $\mathbf{\Phi}$, Associated with $\boldsymbol{\zeta}$

AR Coefficient Matrix for zeta			
	Col1	Col2	Col3
Row1	0.989828	0	0
Row2	0	0.962479	0
Row3	0	0	0.803032

Output 33.7.4 Estimated Disturbance Covariance of $\boldsymbol{\zeta}$

Disturbance Covariance for zeta			
	Col1	Col2	Col3
Row1	0.108104	-0.02618	0.087117
Row2	-0.02618	0.360624	0.008945
Row3	0.087117	0.008945	1.072188

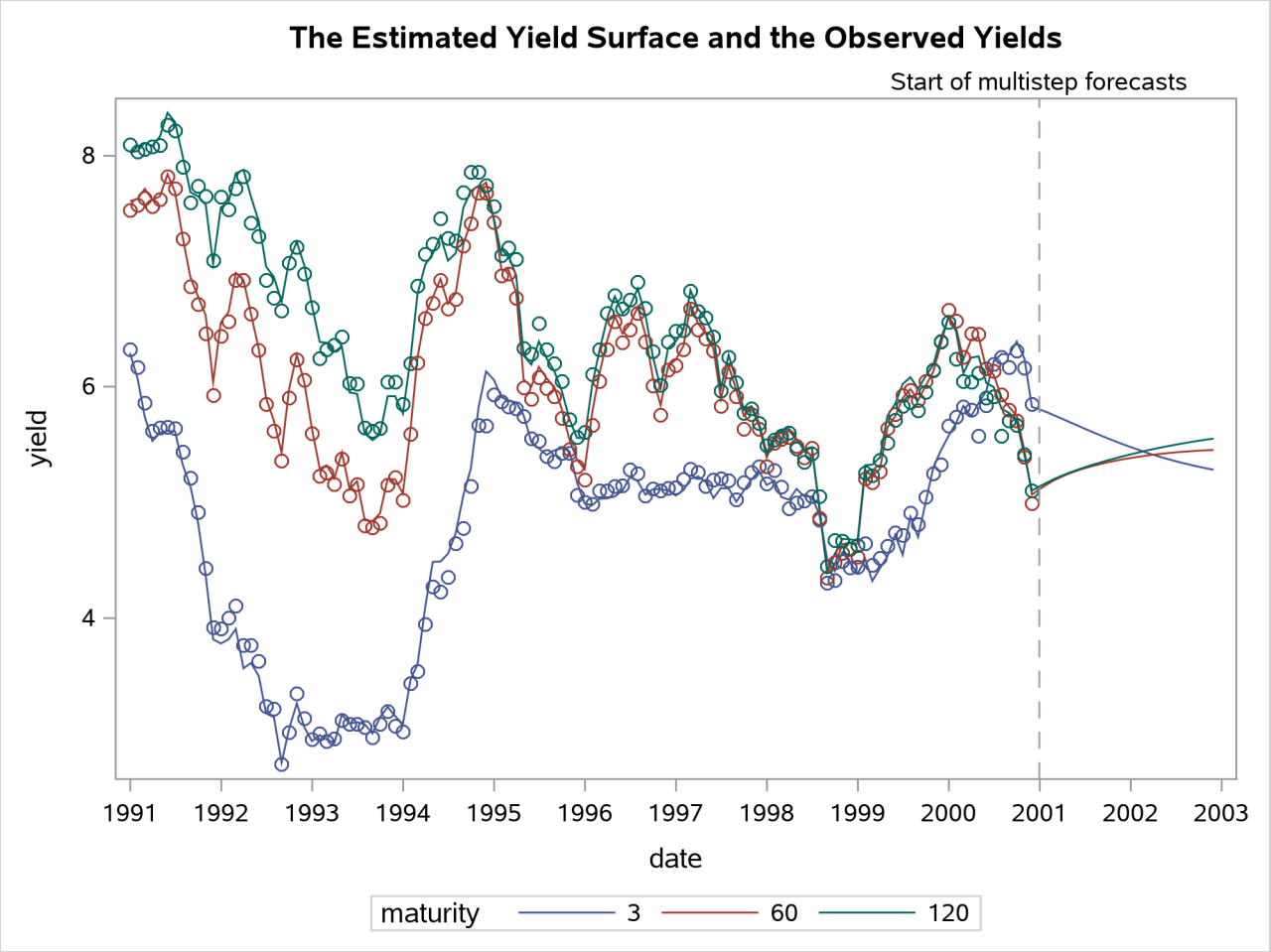
Output 33.7.5 Likelihood Computation Summary for the DNS Factor Model

Likelihood Computation Summary	
Statistic	**Value**
Nonmissing Response Values Used	6324
Estimated Parameters	33
Initialized Diffuse State Elements	3
Normalized Residual Sum of Squares	6321.0007
Diffuse Log Likelihood	3548.9546
Profile Log Likelihood	3547.494

The following statements produce the time series plots of the smoothed estimate of the idealized bond yield ($\theta_t(\tau)$) for bonds with maturities 30, 60, and 120 months (shown in Output 33.7.6). To simplify the display, the plots exclude the time span prior to 1991.

```
proc sgplot data= dnsFor;
    title "The Estimated Yield Surface and the Observed Yields ";
    where maturity in (3 60 120) and date >= '31dec1990'd;
    series x=date y=smoothed_yieldSurface / group=maturity;
    scatter x=date y=yield / group=maturity;
    refline '31dec2000'd / axis=x lineattrs=GraphReference(pattern = Dash)
        name="RefLine"   label="Start of multistep forecasts";
run;
```

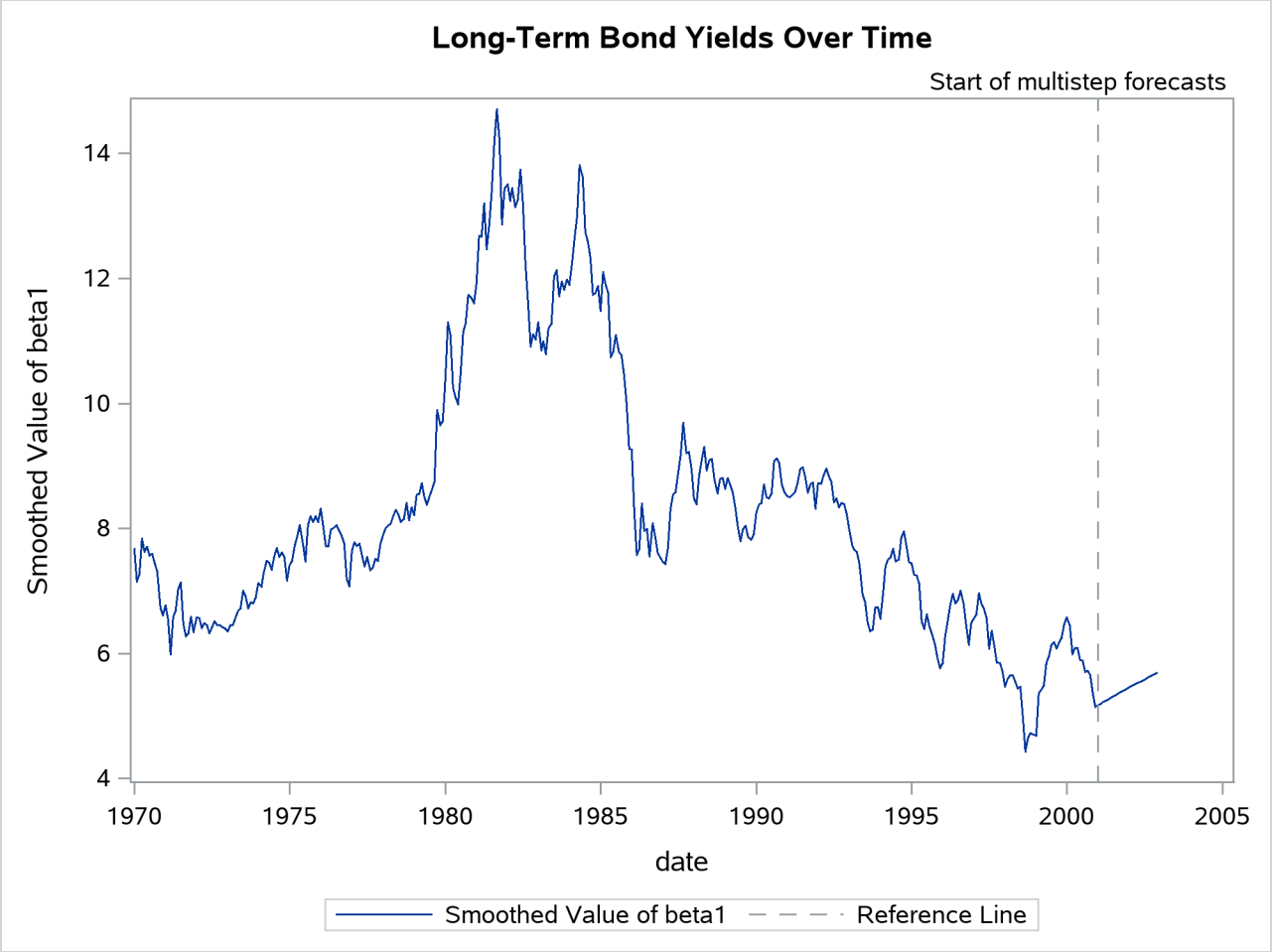
Output 33.7.6 Smoothed Estimate of $\theta_t(\tau)$ for $\tau = 3, 60, 120$



The plots indicate that the DNS model is a reasonable description of the yield data. Similar plots (not shown here) for other maturities also indicate the adequacy of the DNS model. The following statements produce the time series plot of the smoothed estimate of β_{1t}, the long-term bond yield (shown in Output 33.7.7) :

```
proc sgplot data=dnsFor;
   title "Long-Term Bond Yields Over Time ";
   series x=date y=smoothed_beta1 ;
   refline '31dec2000'd / axis=x lineattrs=GraphReference(pattern = Dash)
        name="RefLine"   label="Start of multistep forecasts";
run;
```

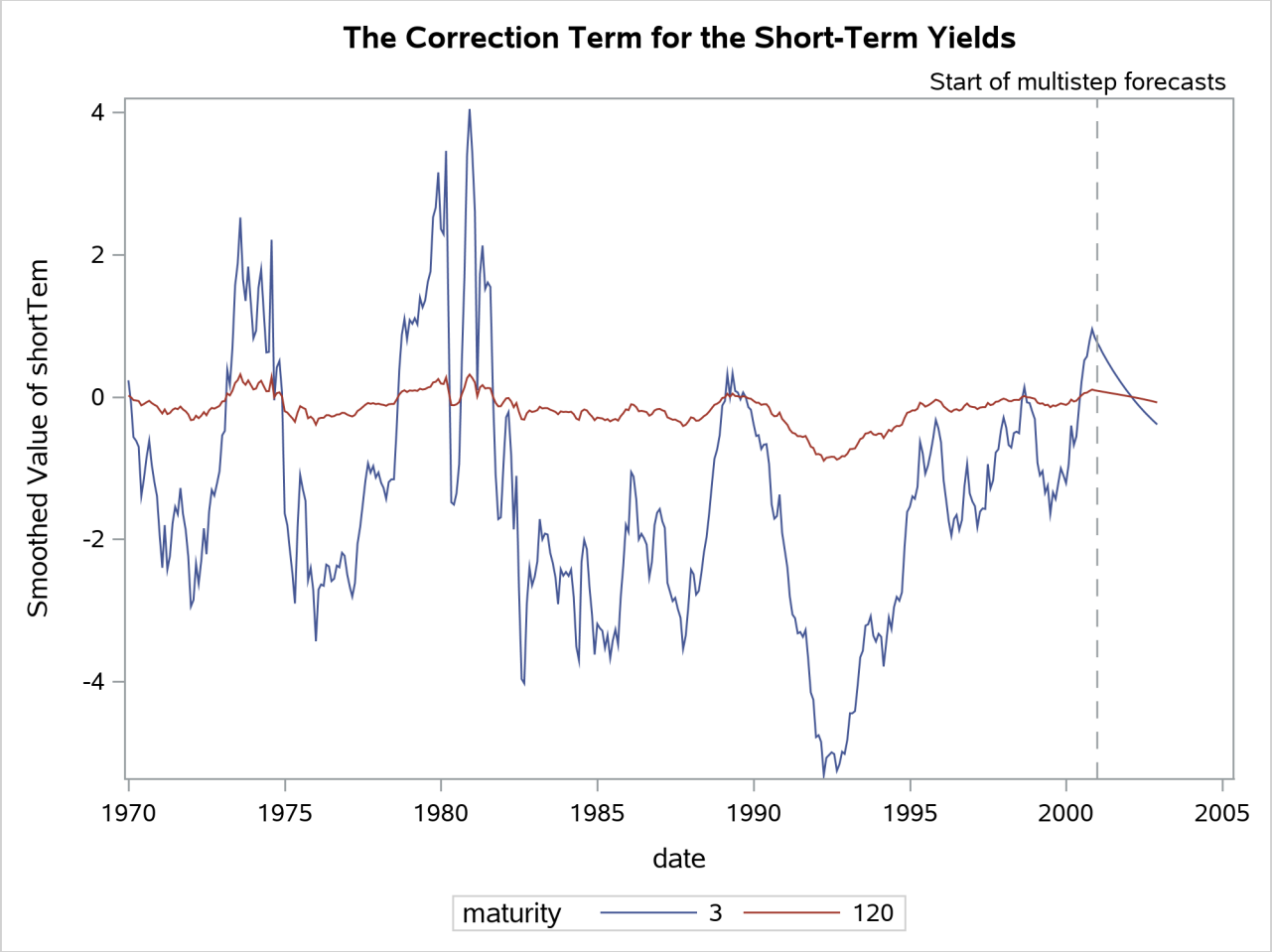
Output 33.7.7 Smoothed Estimate of β_{1t}, the Long-Term Yield



Similarly, Output 33.7.8, which is produced by the following statements, shows the smoothed estimate of the correction to the overall yield that is provided by the second term ($Z2 * \beta_{2t}$) for maturities of 3 months and 120 months. As expected, the correction for the (long-term) maturity of 120 months is negligible compared to the (short-term) maturity of 3 months.

```
proc sgplot data=dnsFor;
  title "The Correction Term for the Short-Term Yields ";
  where maturity in (3 120);
  series x=date y=smoothed_shortTem / group=maturity;
  refline '31dec2000'd / axis=x lineattrs=GraphReference(pattern = Dash)
        name="RefLine"   label="Start of multistep forecasts";
run;
```

Output 33.7.8 Smoothed Estimate of $Z2 * \beta_{2t}$, the Correction Term for the Short-Term Yields

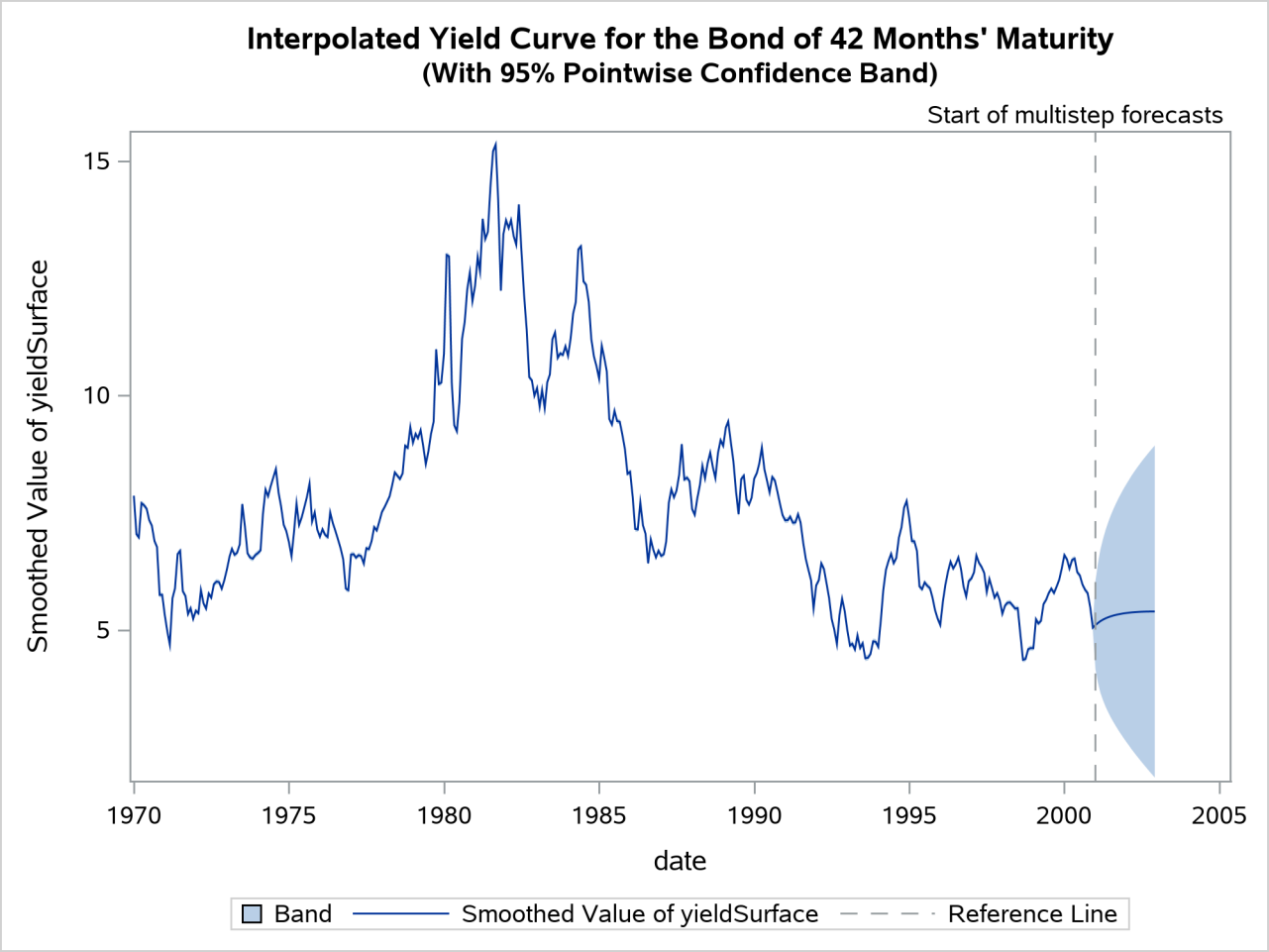


The following statements create plots that show the estimated yield for the hypothetical bond whose maturity is 42 months:

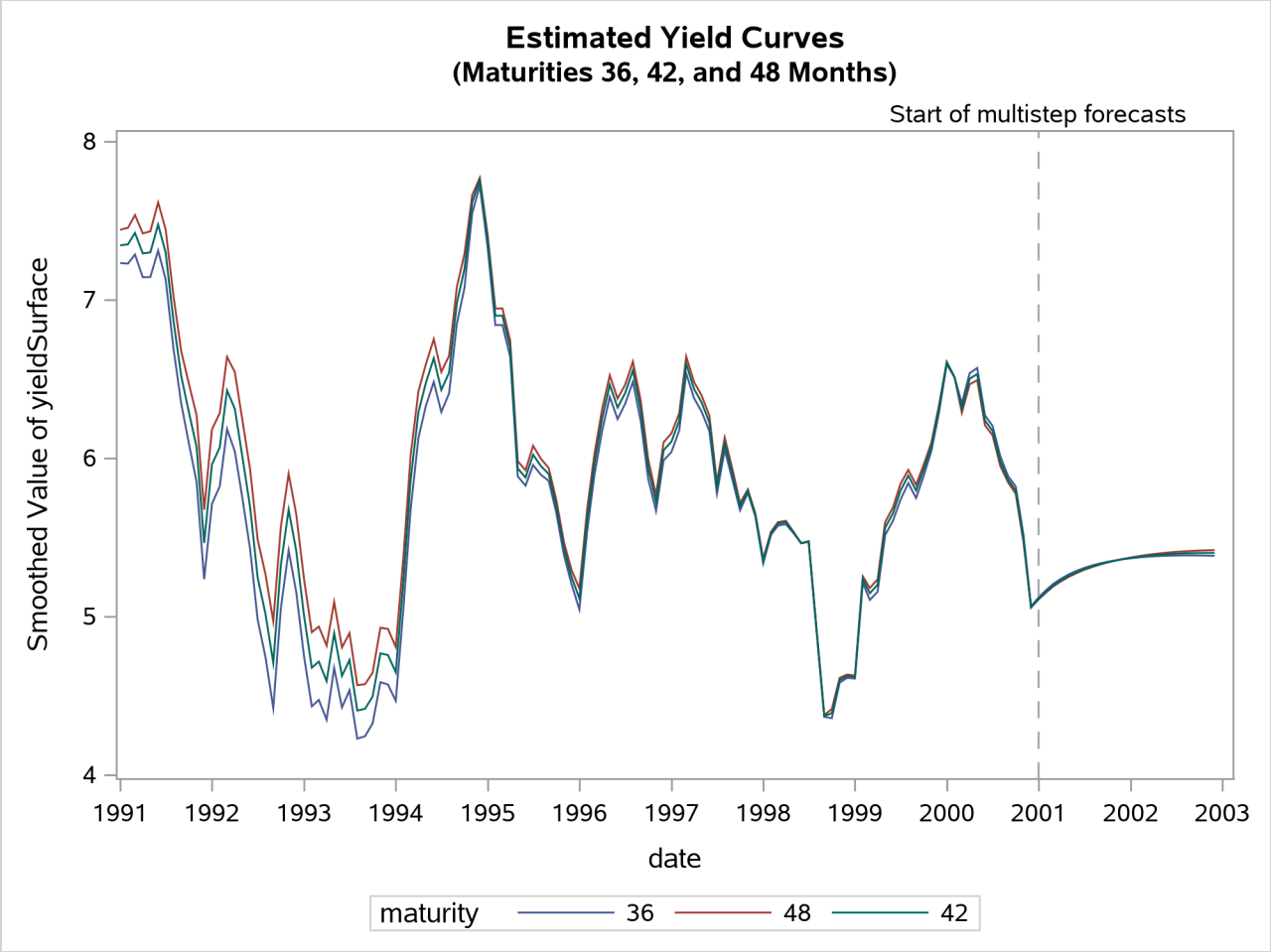
```
proc sgplot data=dnsFor;
   title "Interpolated Yield Curve for the Bond of 42 Months' Maturity";
   title2 "(With 95% Pointwise Confidence Band)";
   where maturity in (42);
   band x=date lower=smoothed_lower_yieldSurface upper=smoothed_upper_yieldSurface;
   series x=date y=smoothed_yieldSurface;
   refline '31dec2000'd / axis=x lineattrs=GraphReference(pattern = Dash)
      name="RefLine"   label="Start of multistep forecasts";
run;

proc sgplot data= dnsFor;
  title "Estimated Yield Curves";
  title2 "(Maturities 36, 42, and 48 Months)";
  where maturity in (36 42 48) and date >= '31dec1990'd;
  series x=date y=smoothed_yieldSurface / group=maturity;
  refline '31dec2000'd / axis=x lineattrs=GraphReference(pattern = Dash)
      name="RefLine"   label="Start of multistep forecasts";
run;
```

Output 33.7.9 shows the interpolated yield curve with a pointwise 95% confidence band. In the historical period, the confidence band appears too tight, mostly because of graphical scaling.

Output 33.7.9 Interpolated Yield Curve for 42 Months' Maturity



Output 33.7.10 Estimated Yield Curves



Output 33.7.10 shows the estimates of $\theta_t(\tau)$ for $\tau = 36$, 42, and 48 months. As expected, the estimated $\theta_t(42)$ lies between the estimates of $\theta_t(36)$ and $\theta_t(48)$.

Example 33.8: Diagnostic Plots and Structural Break Analysis

This example provides information about the diagnostic plots that the SSM procedure produces. In addition, a simple illustration of structural break analysis is also provided. For additional examples of structural break analysis, see Selukar (2017). The following plots are available in the SSM procedure:

- a panel of two plots—a histogram and a Q-Q plot—for the normality check of the one-step-ahead residuals $\nu_{t,i}$. A separate panel is produced for each response variable.
- a time series plot of standardized residuals, one per response variable
- a panel of two plots—a histogram and a Q-Q plot—for the normality check of the prediction errors $\mathrm{AO}_{t,i}$. A separate panel is produced for each response variable.
- a time series plot of standardized prediction errors, one per response variable
- a time series plot of maximal state shock chi-square statistics

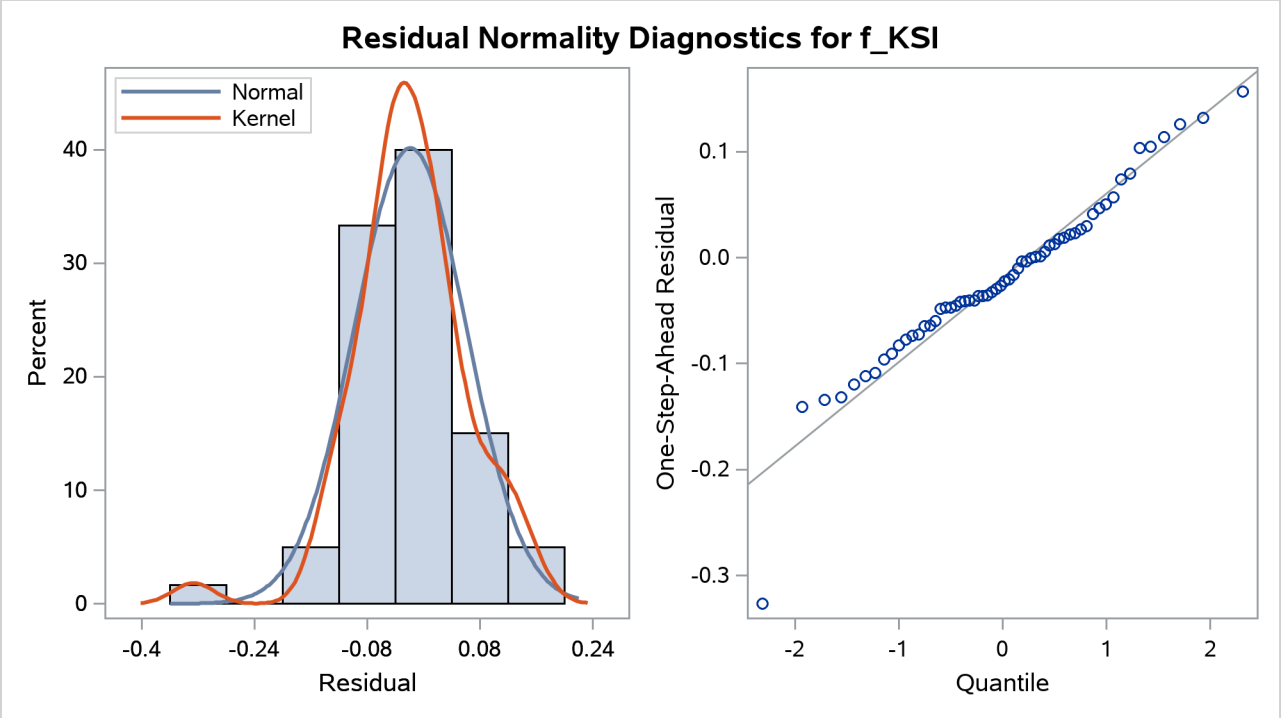
All these plots are used primarily for model diagnostics. In this example, the automobile seat-belt data that are discussed in Example 33.1 are revisited. In Example 33.1, the question under consideration is whether the data show evidence of the effectiveness of the seat-belt law that was introduced in the first quarter of 1983. An intervention variable, Q1_83_Shift, was used in the model to measure the effect of this law on the drivers and front-seat passengers who were killed or seriously injured in car accidents (f_KSI). In the current example, the analysis of these data begins without the knowledge of this seat-belt law. In effect, the same model is fitted without the use of the intervention variable Q1_83_Shift.

The following statements specify the model (without the intervention variable):

```
proc ssm data=seatBelt optimizer(tech=interiorpoint) plots=all;
   id date interval=quarter;
   state error(2) type=WN cov(g);
   component wn1 = error[1];
   component wn2 = error[2];
   state level(2) type=RW cov(rank=1) checkbreak;
   component rw1 = level[1];
   component rw2 = level[2];
   state season(2) type=season(length=4);
   component s1 = season[1];
   component s2 = season[2];
   model f_KSI = rw1 s1  wn1;
   model r_KSI = rw2 s2 wn2;
run;
```

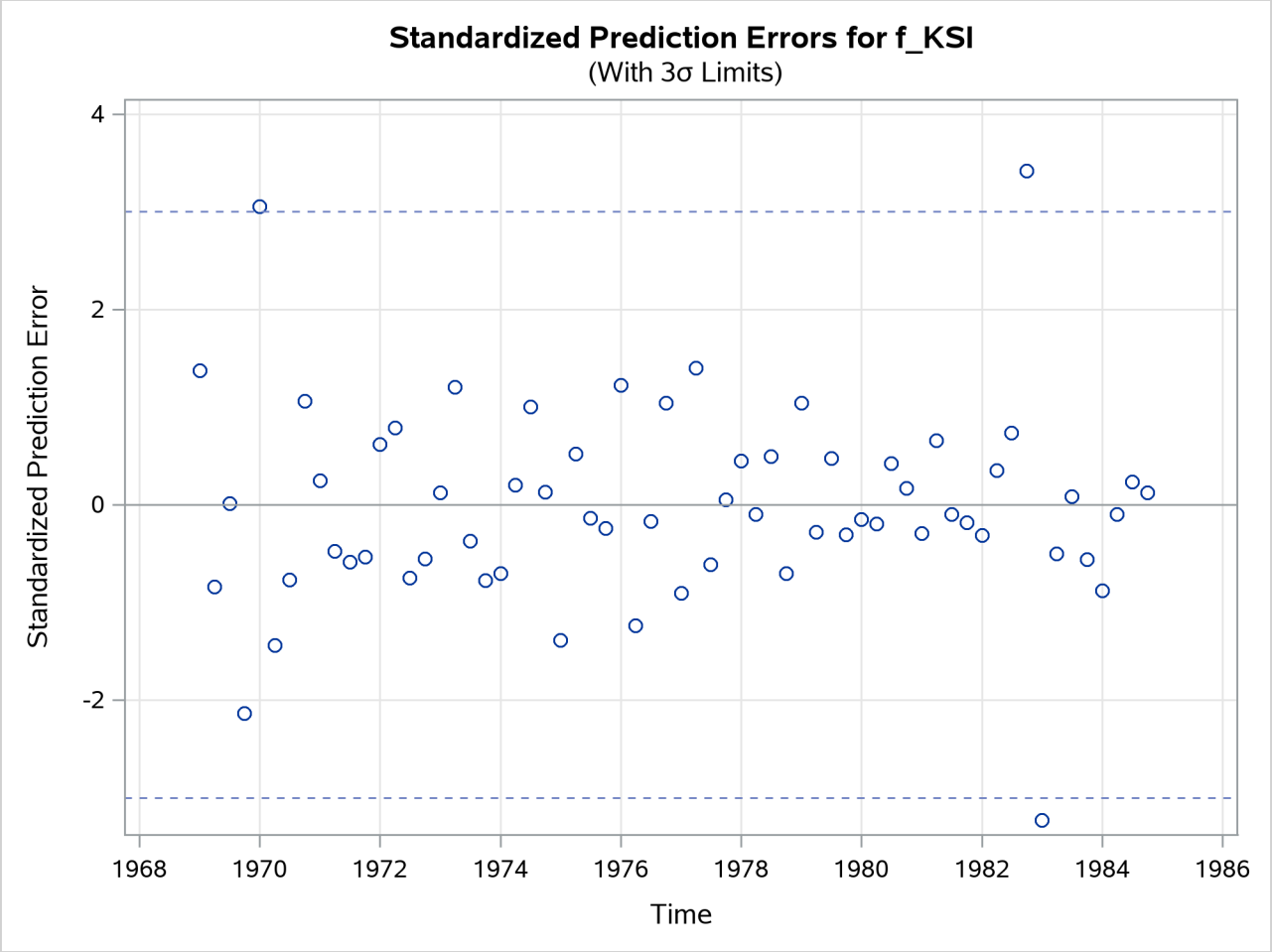
The PLOTS=ALL option in the PROC SSM statement turns on all the plotting options. Because there are two response variables, nine plots in total are produced: a separate set of four plots—two residual and two prediction error—is produced for f_KSI and r_KSI, and one maximal shock plot is produced. Only three of these plots are shown here. Output 33.8.1 shows the normality check for the one-step-ahead residuals for f_KSI. It shows some evidence of lack of normality.

Output 33.8.1 Normality Check of One-Step-Ahead Residuals for f_KSI



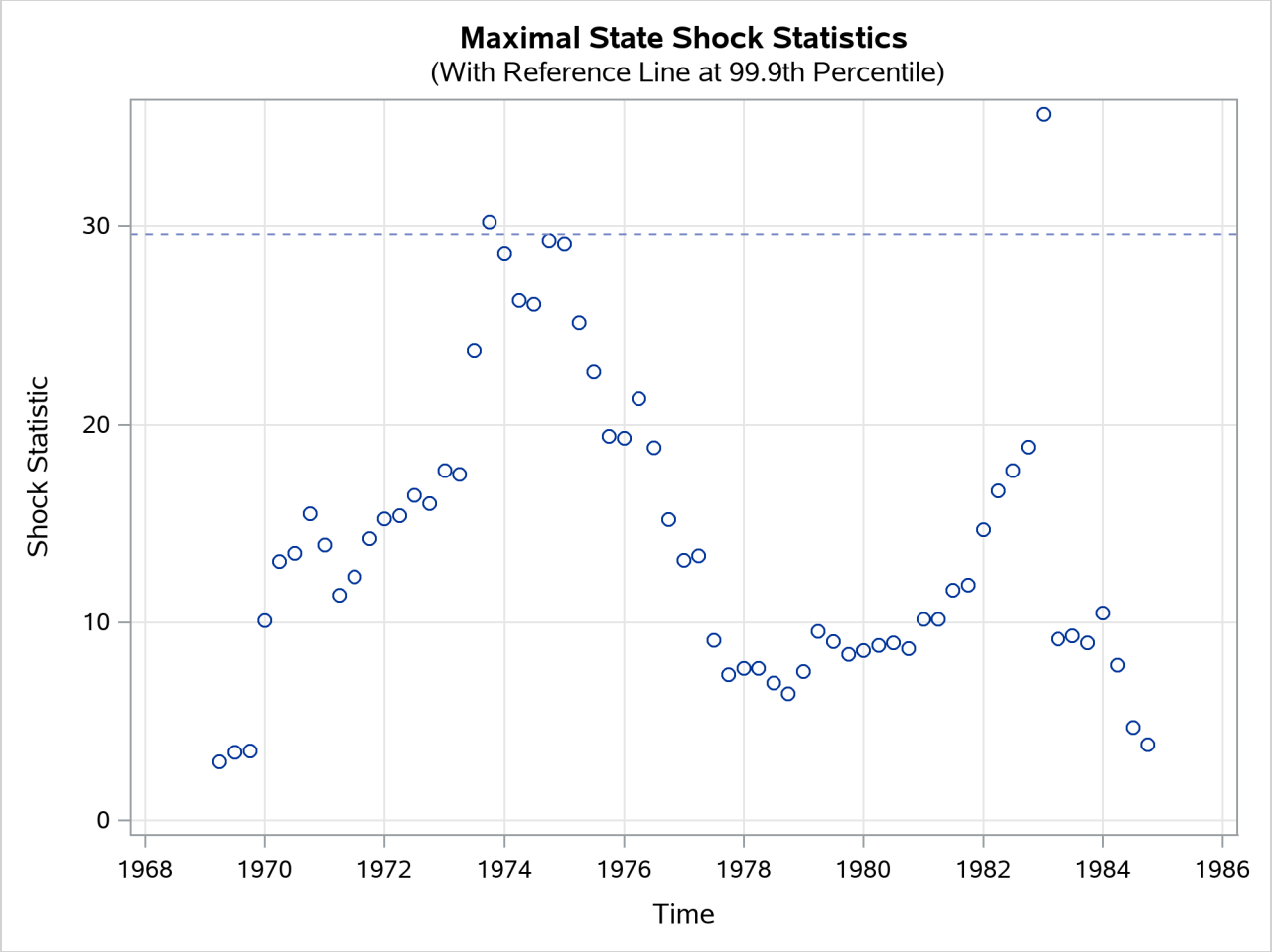
Output 33.8.2 shows the time series plot of standardized prediction errors for f_KSI. It identifies some extreme observations (additive outliers): two near 1983 and one near 1970.

Output 33.8.2 Time Series Plot of Standardized Prediction Errors for f_KSI



Output 33.8.3 shows the time series plot of maximal shock statistics. This plot can be very informative in showing the temporal locations of the structural changes in the overall observation-generation process (treating the fitted model as the reference). It can indicate locations of shifts in the process level or shifts in other characteristics, such as its slope. The precise nature of the shift (whether the shift occurs in the level or in some other aspects) can be determined by using the CHECKBREAK option in the appropriate STATE and TREND statements (as is done in the STATE statement in this example that defines the bivariate state level). In this example, the maximal shock statistics plot indicates two locations—the last quarter of 1973 and the first quarter of 1983—as likely locations for the structural breaks that are associated with the traffic accident process. These are indeed reasonable findings, because the last quarter of 1973 (beginning in October 1973) is associated with the start of the oil crisis that severely curtailed worldwide automobile traffic, and the first quarter of 1983 is associated with the introduction of the seat-belt law that might have improved the safety of drivers and front-seat passengers. In addition, Output 33.8.4 shows the summary of most likely break locations for the bivariate state level. It identifies a break in the first element of level (which corresponds to the drivers and front-seat passengers) in the first quarter of 1983.

Output 33.8.3 Time Series Plot of Maximal Shock Statistics



Output 33.8.4 Elementwise Break Summary for the Bivariate State: level

Elementwise Break Summary for level

Element ID	Index	Z Value	Pr > |z|
1983:1	1	-5.85	<.0001

The following statements fit a revised model that accounts for the break in the first element of level by introducing a dummy variable, Q1_83_Pulse, in the state equation:

```
ods output ElementStateBreakDetails=stateBreak;
proc ssm data=seatBelt optimizer(tech=interiorpoint) plots=all;
   id date interval=quarter;
   Q1_83_Pulse = (date = '1jan1983'd);
   zero = 0;
   state error(2) type=WN cov(g);
   component wn1 = error[1];
   component wn2 = error[2];
   state level(2) type=RW cov(rank=1) W(g)=(Q1_83_Pulse zero)
       checkbreak print=breakdetail;
   component rw1 = level[1];
   component rw2 = level[2];
   state season(2) type=season(length=4);
   component s1 = season[1];
   component s2 = season[2];
   model f_KSI =  rw1 s1  wn1;
   model r_KSI = rw2 s2 wn2;
run;
```

Note that using Q1_83_Pulse in the definition of level is equivalent to using Q1_83_Shift in the MODEL statement for f_KSI in Example 33.1. Output 33.8.5 shows the estimated change in the first element of the state level, which is the same as the estimated level shift shown in Output 33.1.6 (this is not surprising, because these two models are statistically equivalent).

Output 33.8.5 Estimate of the Regression Coefficient of Q1_83_Pulse

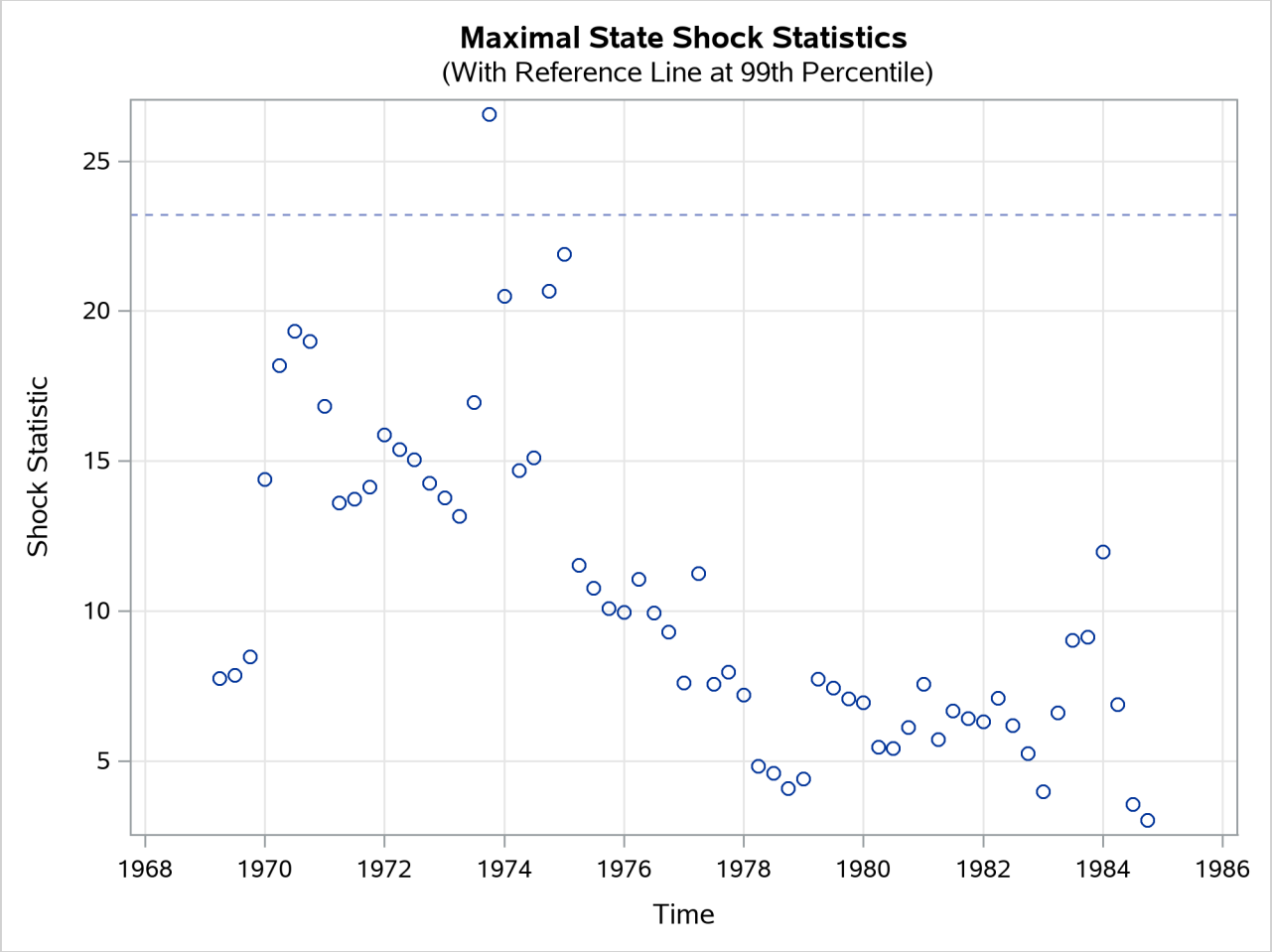
The SSM Procedure

Estimate of the State Equation Regression Vector

State	Element Index	Estimate	Standard Error	t Value	Pr > |t|
level	1	-0.408	0.0259	-15.74	<.0001

In the preceding SSM procedure statements, the CHECKBREAK option is used along with the PRINT=BREAKDETAIL option, which produces a table that contains the break statistics at every distinct time point (this table, in turn, is captured in the output data set stateBreak for later use). Output 33.8.6 shows the time series plot of maximal shock statistics for this revised model. As expected, the plot no longer shows the first quarter of 1983 as a structural break location. It continues to show the last quarter of 1973 as a structural break location, because the fitted model does not try to explicitly account for this shift.

Output 33.8.6 Time Series Plot of Maximal Shock Statistics for the Model with Q1_83_Pulse

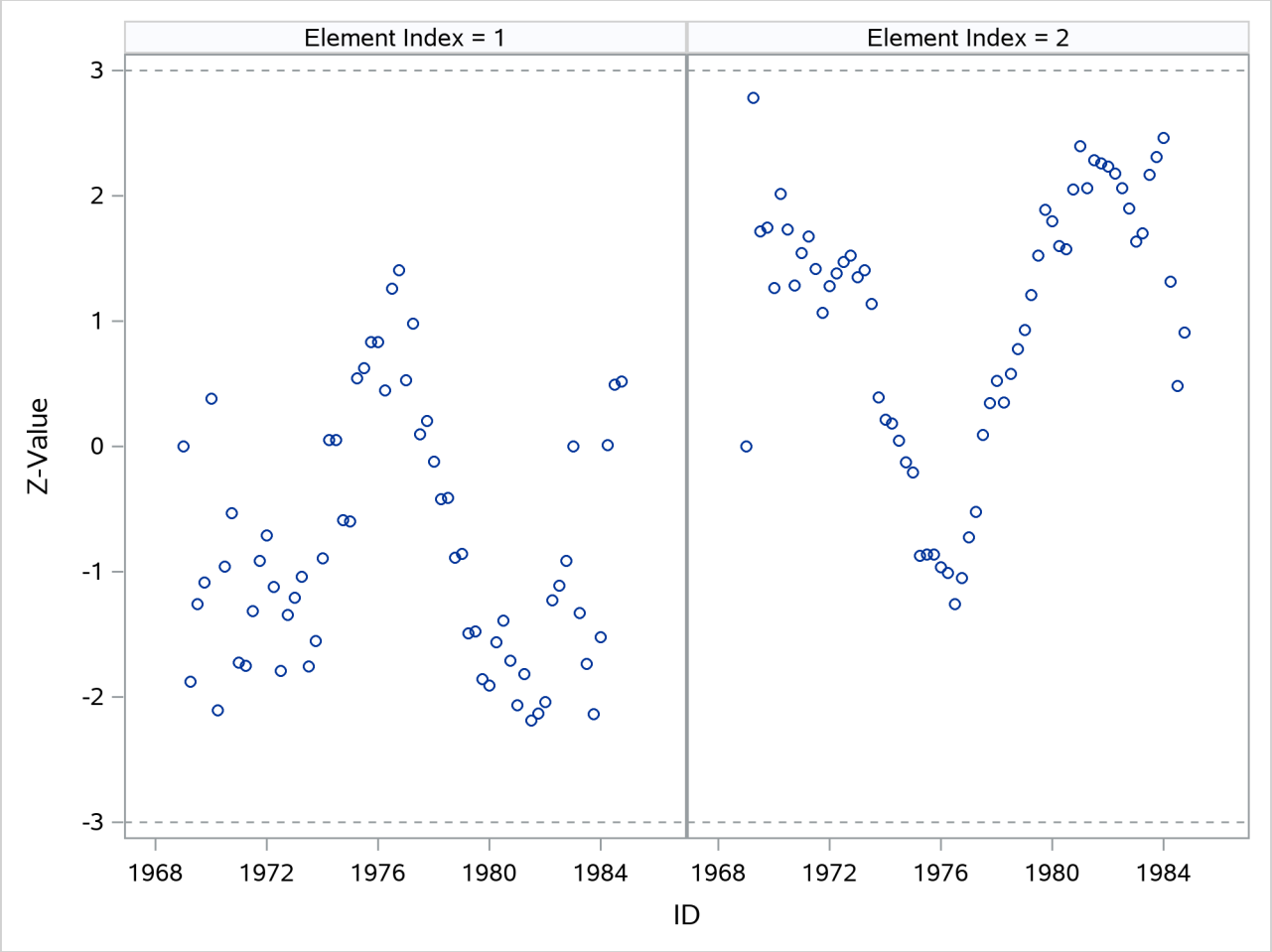


Note that the reference line in Output 33.8.3 is drawn at the 99.9th percentile, whereas the reference line in Output 33.8.6 is drawn at the 99th percentile. The reference line location in the maximal state shock chi-square statistics plot is based on the points in the plot. A reference line is drawn at percentile 80, 90, 99, or 99.9 based on the largest maximal shock statistic that is shown.

The detailed information in the data set stateBreak can be used to further investigate the possibility of significant breaks in the trend in and around 1973. The following statements produce scatter plots for the break statistics for both the drivers and front passengers and the rear passengers (reference lines are also drawn at –3 and 3 to check for extreme Z values):

```
proc sgpanel data=stateBreak;
    panelby elementIndex;
    scatter x=time y=zValue;
    refline 3 / axis=y lineattrs=(pattern=shortdash) noclip;
    refline -3 / axis=y lineattrs=(pattern=shortdash) noclip;
run;
```

Output 33.8.7 Elementwise Structural Break Statistics for level



The resulting graph, shown in Output 33.8.7, shows possible breaks in the second element—rear side passengers—around 1969. In general, however, the evidence of breaks in the elements of level is not very strong. This means that you must look elsewhere to explain the extreme point in Output 33.8.6.

Example 33.9: Longitudinal Data: Variable Bandwidth Smoothing

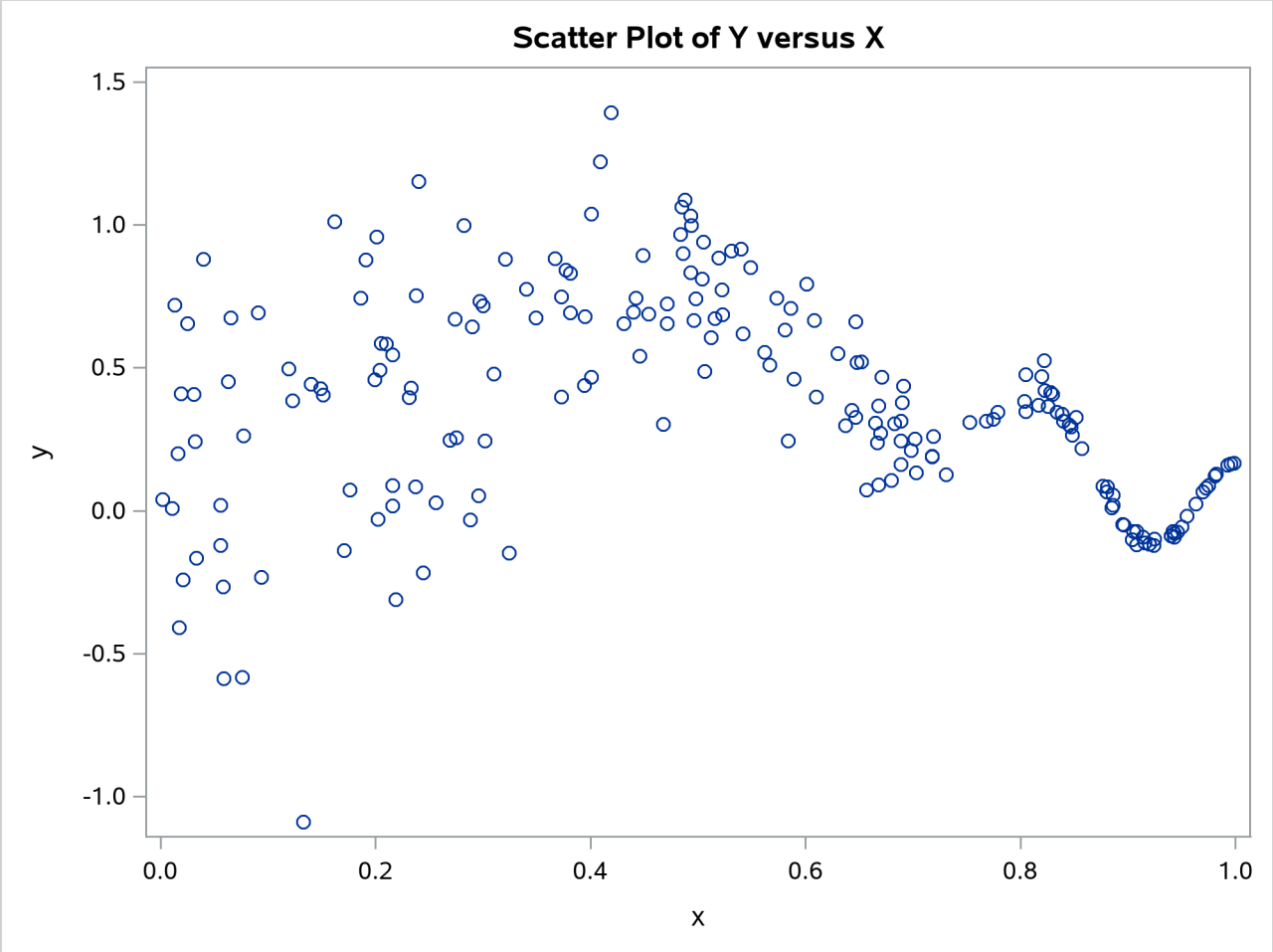
The data for this example, taken from Givens and Hoeting (2005, chap. 11, Example 11.8), contain two variables, x and y. The variable y represents noisy evaluation of an unknown smooth function at x. The data are sorted by x.

```
data Difficult;
input x y;
datalines;
0.002 0.040
0.011 0.009
0.013 0.719
0.016 0.199
0.017 -0.409

  ... more lines ...
```

Output 33.9.1 shows the scatter plot of y against x that is generated by the following statements:

```
proc sgplot data=Difficult;
   title "Scatter Plot of Y versus X";
   scatter x=x y=y ;
run;
```

Output 33.9.1 Scatter Plot of Y versus X



The plot clearly shows that the variance of y values varies considerably over the range of x values—the variance is larger for x values around 0.2 and gets increasingly smaller as the x values get closer to 1. Givens and Hoeting (2005) discuss the difficulties of extracting a smooth pattern from such data. Consider the following model for y:

$$\mathrm{y}(x) = \boldsymbol{\mu}_x + \boldsymbol{\epsilon}_x$$

where $\boldsymbol{\mu}_x$ is a smooth trend component and $\boldsymbol{\epsilon}_x$ is the observation noise with variance, $h(x)$, which changes with x: $\boldsymbol{\epsilon}_x \sim N(0, h(x))$. It is known (Durbin and Koopman 2012, chap. 3, sect. 9 and chap. 8, sect. 5) that modeling the trend $\boldsymbol{\mu}_x$ as a polynomial smoothing spline (for example, the way the growth curves are modeled in Example 33.4) and taking the variance function of the observation noise $\boldsymbol{\epsilon}_x$ a constant results in a trend estimate that can be termed a fixed-bandwidth-smoother. The optimal bandwidth turns out to be a

function of the signal-to-noise ratio: the ratio of the observation noise variance and the disturbance variance of the trend component. On the other hand, allowing the variance function of the observation noise to change with the x values results in a trend estimate that can be termed a variable-bandwidth-smoother. The rest of this example shows how to use the SSM procedure to create a data-dependent variance function $h(x)$ and to extract the associated (variable-bandwidth) smooth trend from such data. Suppose that the (unknown) variance function $h(x)$ can be approximated as

$$h(x) = \exp(\sum_{i=1}^{7} v_i \ \mathrm{SplineBasis}_i(x))$$

where $v_i, i = 1, 2, \ldots, 7$ are unknown parameters and $\mathrm{SplineBasis}_i(x), i = 1, 2, \ldots, 7$, are the full set of cubic spline basis functions (B-splines) with four evenly spaced internal knots between the range of x values—essentially, four equispaced points between 0.0 and 1.0. Note that the number of basis functions in the full set, 7, is the sum of the number of internal knots, 4, and the degree of the polynomial, 3. The following statements create a data set, Combined, that contains the variables x and y, along with the desired spline basis functions (col1–col7) that are created by using the BSPLINE function in PROC IML:

```
proc iml;
   use difficult;
   /* read x and y from difficult into temp */
   read all var _num_ into temp;
   x = temp[,1];
   /* generate B-spline basis for a cubic spline
      with 4 evenly spaced internal knots in the x-range */
   bsp = bspline(x, 2, ., 4);
   Combined = temp || bsp;
   /* create a merged data set with x, y, and
      spline basis columns                    */
   create Combined var {x y col1 col2 col3 col4 col5 col6 col7};
   append from Combined;
quit;
```

The following statements specify and fit the desired model to the data:

```
proc ssm data=Combined opt(tech=dbldog);
   id x;
   /* parameters needed to define h(x) */
   parms v1-v7;
   /* defining h(x) */
   var = exp(v1*col1 + v2*col2 + v3*col3 + v4*col4
      + v5*col5 + v6*col6 + v7*col7);
   /* defining the polynomial spline trend */
   trend trend(ps(2));
   /* defining the observation noise with variance h(x) */
   irregular wn variance=var;
   model y = trend wn;
   output out=For pdv;
run;
```

Output 33.9.2 shows the estimates of $v_i, i = 1, 2, \ldots, 7$, and Output 33.9.3 shows the estimate of the disturbance variance associated with the polynomial spline trend that is specified in the TREND statement.

Output 33.9.2 Estimates of v1–v7

The SSM Procedure

Estimates of Named Parameters

Parameter	Estimate	Standard Error	t Value
v1	-3.302	1.501	-2.20
v2	-0.826	0.619	-1.33
v3	-2.234	0.453	-4.93
v4	-3.130	0.412	-7.60
v5	-4.306	0.415	-10.38
v6	-6.901	0.588	-11.73
v7	-19.514	2.306	-8.46

Output 33.9.3 Estimate of the Disturbance Variance Associated with the Trend

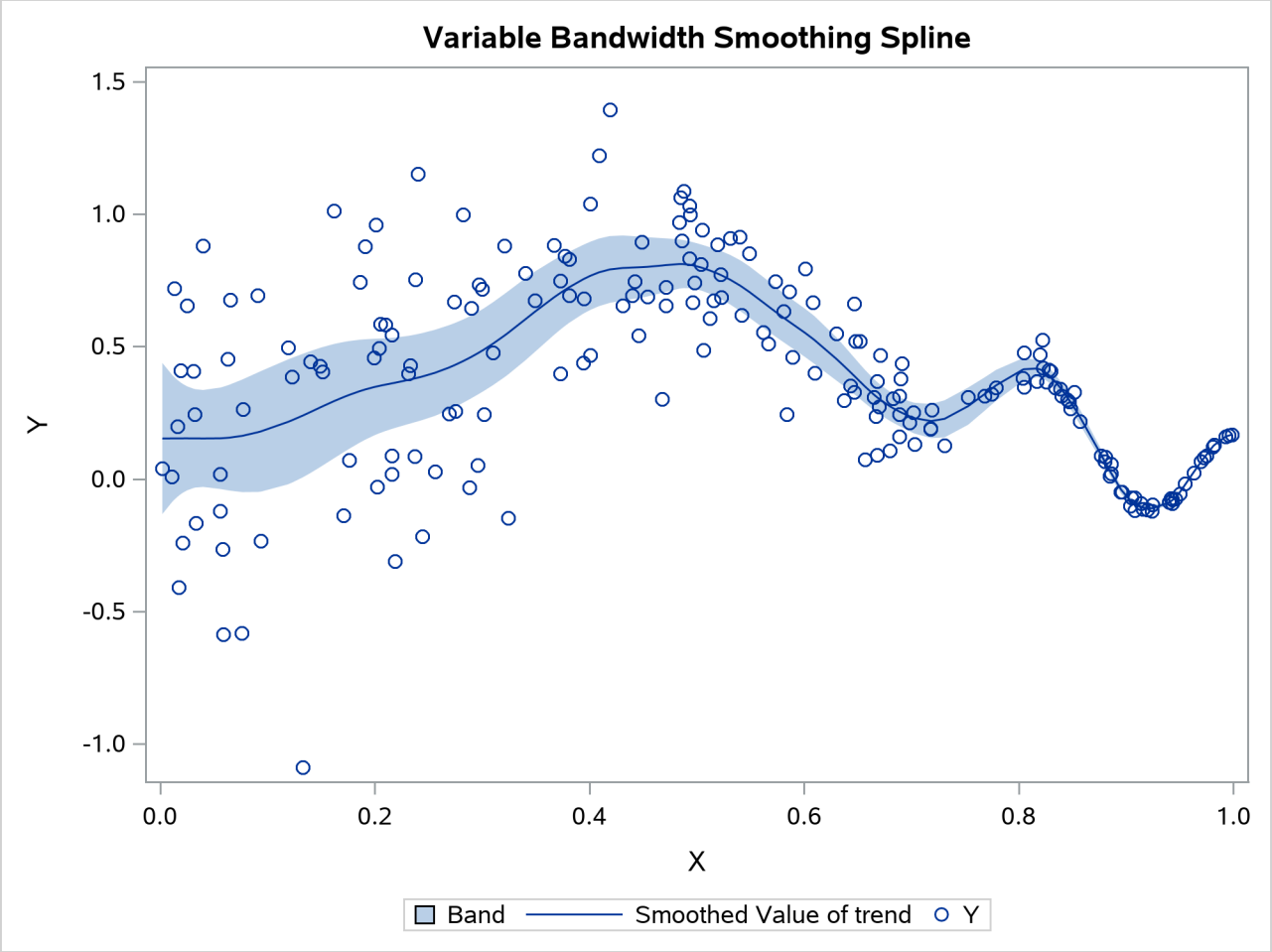
Model Parameter Estimates

Component	Type	Parameter	Estimate	Standard Error	t Value
trend	**PS(2) Trend**	**Level Variance**	339	110	3.07

The following statements produce a plot, shown in Output 33.9.4, of the fitted trend with 95% confidence band:

```
proc sgplot data=For;
   title "Variable Bandwidth Smoothing Spline";
   band x=x lower=smoothed_lower_trend
      upper=smoothed_upper_trend ;
   series x=x y=smoothed_trend;
   scatter x=x y=y;
run;
```

Output 33.9.4 Fitted Trend with 95% Confidence Band

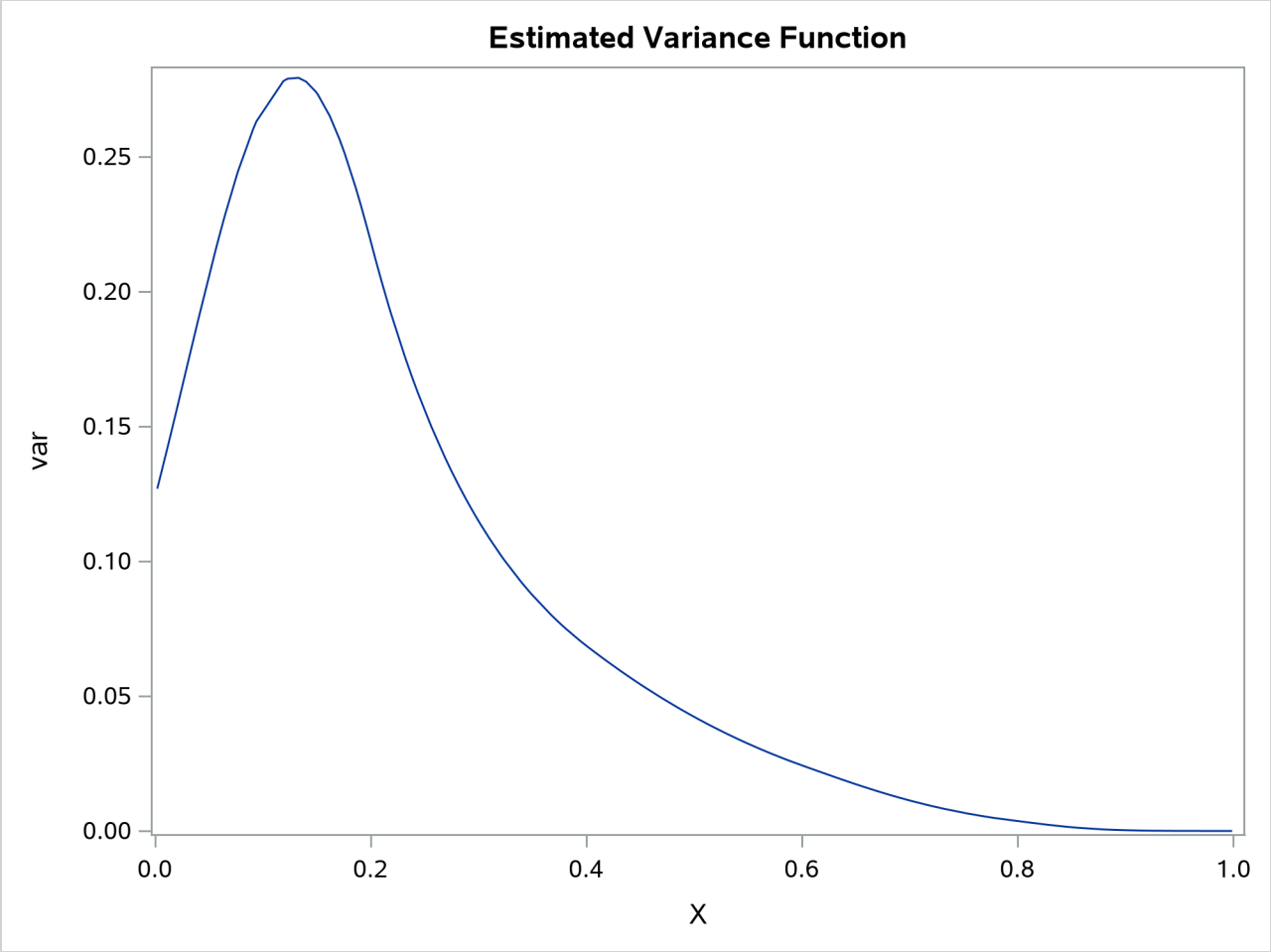


Clearly the fitted curve tracks the data quite well. Lastly, Output 33.9.5 (produced by using the following statements) shows the estimated variance function $h(x)$.

```
proc sgplot data=For;
   title "Estimated Variance Function";
   series x=x y=var;
run;
```

As expected, the curve attains its peak at an x value around 0.18 and decays to nearly 0 as x values reach 1.0.

Output 33.9.5 Estimated Variance Function $h(x) = \exp(\sum_{i=1}^{7} \hat{\nu}_i \ \mathrm{SplineBasis}_i(x))$



Example 33.10: A Transfer Function Model for the Gas Furnace Data

This example describes how you can include components in your model that follow a transfer function model. Transfer function models, a generalization of distributed lag models, are useful for capturing the contributions from lagged values of the predictor series. Box and Jenkins popularized ARIMA models with transfer function inputs in their famous book (Box and Jenkins 1976). This example shows how you can specify an ARIMA model that is suggested in that book to analyze the data collected in an experiment at a chemical factory. The data set, called Series J by Box and Jenkins, contains sequentially recorded measurements of two variables: x, the input gas rate, and y, the output CO_2. For the output CO_2, Box and Jenkins suggest the model

$$y_t = \mu + f_t + \zeta_t$$

where μ is the intercept, ζ_t is a zero-mean noise term that follows a second-order autoregressive model (that is, $\zeta_t \sim \mathrm{AR}(2)$), and f_t follows a transfer function model

$$f_t = \frac{(\gamma_1 B^3 + \gamma_2 B^4 + \gamma_3 B^5)}{(1 - \delta B)} x_t$$

The model for f_t is specified by using the ratio of two polynomials in the backshift operator B. Alternatively, this model can also be described as follows:

$$f_t = \delta f_{t-1} + \gamma_1 x_{t-3} + \gamma_2 x_{t-4} + \gamma_3 x_{t-5}$$

In this alternate form, it is easy to see that the equation for f_t can also be seen as a state evolution equation for a one-dimensional state with a (1×1) transition matrix δ and state regression variables x_{t-3}, x_{t-4}, and x_{t-5} (lagged values of x). This state equation has no disturbance term.

The following statements define the data set Seriesj. The variables x3, x4, and x5, which denote the appropriately lagged values of x, are also created. These variables are used later in the STATE statement that is used to specify f_t.

```
data Seriesj;
   input x y @@;
   label x = 'Input Gas Rate'
         y = 'Output CO2';
   x3 = lag3(x);
   x4 = lag4(x);
   x5 = lag5(x);
   obsIndex = _n_;
   label obsIndex = 'Observation Index';
datalines;
-0.109  53.8  0.000  53.6  0.178  53.5  0.339  53.5
 0.373  53.4  0.441  53.1  0.461  52.7  0.348  52.4
 0.127  52.2 -0.180  52.0 -0.588  52.0 -1.055  52.4

   ... more lines ...
```

The following SSM procedure statements carry out the modeling of y, the output CO_2, according to the preceding model:

```
proc ssm data=Seriesj(firstobs=6);
    id obsIndex;
    parms delta /lower=-0.9999 upper=0.9999;
    state tfstate(1) T(g)=(delta) W(g)=(x3 x4 x5) a1(1) checkbreak;
    comp tfinput = tfstate[1];
    trend ar2(arma(p=2)) ;
    intercept = 1;
    model y = intercept tfinput ar2 ;
    eval modelCurve = intercept + tfinput;
    forecast out=For;
run;
```

The coefficient of the denominator polynomial, δ, is specified in a PARMS statement. It is constrained to be less than 1 in magnitude, which ensures that the transfer function term does not have explosive growth. The transfer function model for f_t is specified in a STATE statement that defines a one-dimensional state named tfstate. In this statement, the transition matrix (which contains only one element, δ) is specified by using the T(g)= option; the state regression variables x3, x4, and x5 are specified by using the W(g)= option; and the CHECKBREAK option is used to turn on the search for unexpected changes in the behavior of f_t. The COV option is absent from this STATE statement because the disturbance term is absent from the state equation for f_t. Moreover, because nothing can be assumed about the initial condition of this state equation, it is taken to be diffuse (as signified by the A1 option). Note that the first five observations of the input data

set, Seriesj, are excluded from the analysis to ensure that the state regression variables x3, x4, and x5 do not contain any missing values. The component that is associated with tfstate, named tfinput, is specified in a COMPONENT statement that follows the STATE statement. A zero-mean, second-order autoregressive noise term, named ar2, is specified by using a TREND statement. Next, a constant regression variable, intercept, is defined to be used in the MODEL statement to capture the intercept term μ. Finally, the model specification is completed by specifying the response variable, y, and the three right-hand terms in the MODEL statement. Next, an EVAL statement is used to specify a component, modelCurve, which is the sum of the intercept and the transfer function input ($\mu + f_t$). The modelCurve component represents the structural part of the model and is defined only for output purposes: its estimate is output (along with the estimates of other components) to the data set that is specified in the OUT= option of the OUTPUT statement.

Note that the modeling of output CO_2 according to this model is also illustrated in Example 7.3 of the PROC ARIMA documentation (see Chapter 7, "The ARIMA Procedure"). The ARIMA procedure handles the computation of the transfer function f_t slightly differently than the way it is estimated by the SSM procedure. However, despite this algorithmic difference in the modeling procedures, for this example the estimated parameters agree quite closely (barring the sign conventions that are used to specify the model parameters).

Output 33.10.1 shows the estimate of μ, the intercept in the model. Output 33.10.2 shows the estimate of δ, the coefficient in the denominator polynomial of the transfer function. Output 33.10.3 shows the regression estimates of the state regression variables x3, x4, and x5 (which correspond to the coefficients of the numerator polynomial). Output 33.10.4 shows the estimates of the parameters of the AR(2) noise term.

Output 33.10.1 Estimate of μ

The SSM Procedure

Regression Parameter Estimates

Response Variable	Regression Variable	Estimate	Standard Error	t Value	Pr > |t|
y	intercept	53.4	0.145	368.20	<.0001

Output 33.10.2 Estimate of δ, the Coefficient of the Denominator Polynomial in the Transfer Function

Estimates of Named Parameters

Parameter	Estimate	Standard Error	t Value
delta	0.548	0.0398	13.78

Output 33.10.3 Regression Estimates of the State Regression Variables x3, x4, and x5

Estimate of the State Equation Regression Vector

State	Element Index	Estimate	Standard Error	t Value	Pr > |t|
tfstate	1	-0.530	0.0743	-7.13	<.0001
tfstate	2	-0.380	0.1022	-3.72	0.0002
tfstate	3	-0.519	0.0743	-6.99	<.0001

Output 33.10.4 Estimates of the Parameters of the Autoregressive Term

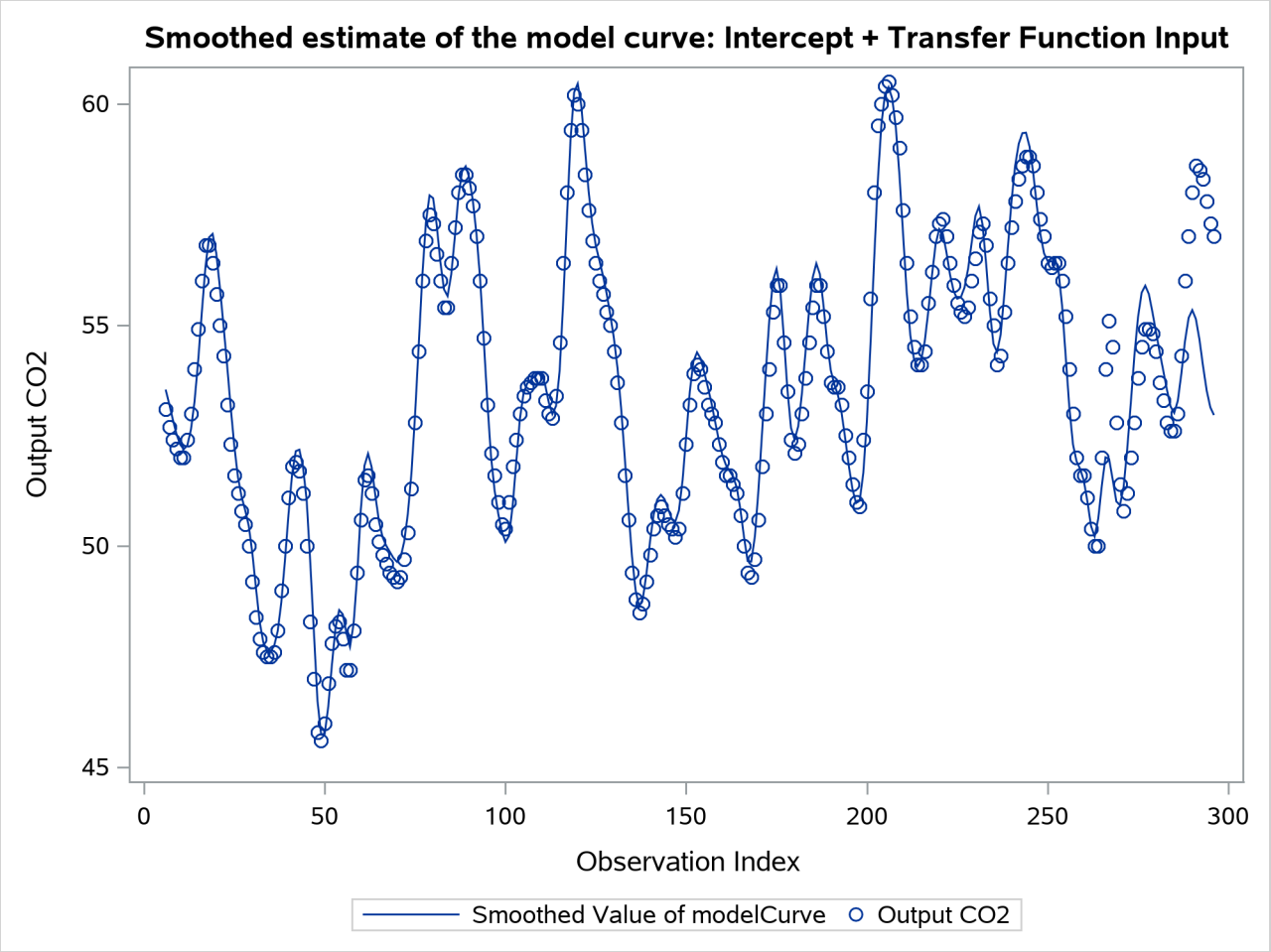
Model Parameter Estimates

Component	Type	Parameter	Estimate	Standard Error	t Value
ar2	ARMA Trend	Error Variance	0.0581	0.00486	11.96
ar2	ARMA Trend	AR_1	1.5319	0.04700	32.60
ar2	ARMA Trend	AR_2	-0.6291	0.04973	-12.65

The following statements produce a time series plot of the estimate of modelCurve—that is, the estimate of the structural part of the model ($\mu + f_t$)—along with the scatter plot of the observed values of the output CO_2. The plot, shown in Output 33.10.5, seems to indicate that the model captures the relationship between the input gas rate and the output CO_2 quite well, at least up to the observation index 250.

```
proc sgplot data=For;
   title "Smoothed estimate of the model curve: Intercept + Transfer Function Input";
   series x=obsIndex y=smoothed_modelCurve;
   scatter x=obsIndex y=y;
run;
```

Output 33.10.5 Smoothed Estimate of $\mu + f_t$

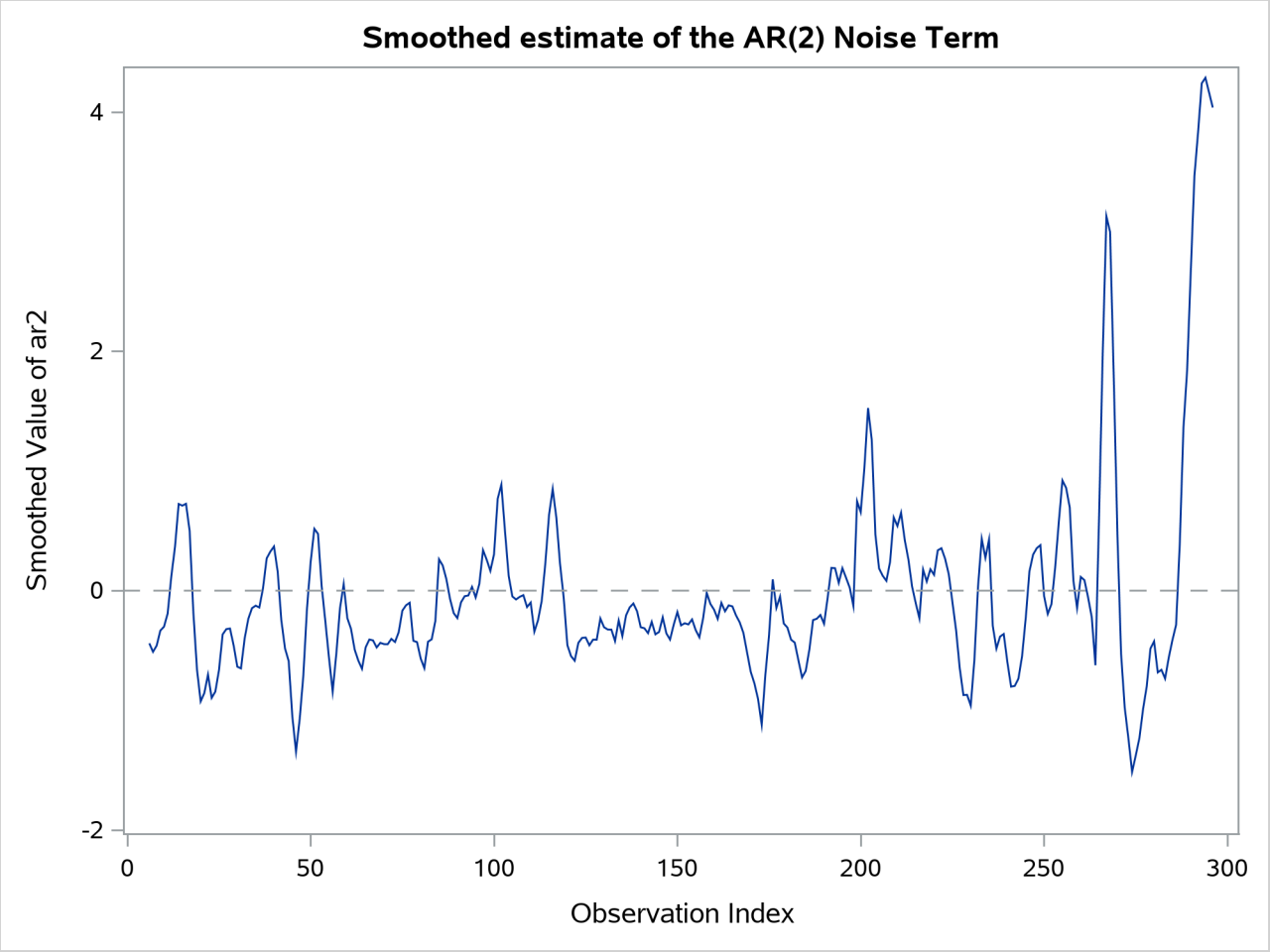


The plot shown in Output 33.10.6 shows the estimate of the noise term, ar2, which is produced by the following statements:

```
proc sgplot data=For;
   title "Smoothed estimate of the AR(2) Noise Term";
   series x=obsIndex y=smoothed_ar2;
   refline 0 / axis=y lineattrs=GraphReference(pattern = Dash);
run;
```

If you compare the scales of these two plots, it appears that the noise term is relatively small and that most of the variation in output CO_2 can be explained by the structural part of the model. It does, however, appear that the model fit deteriorates toward the latter part of the sample. The structural break analysis summary, shown in Output 33.10.7, indicates strong evidence of structural break in the behavior of f_t at or near the observation index 264. Obviously, this type of structural break analysis can be quite useful in industrial quality-control applications.

Output 33.10.6 Smoothed Estimate of the AR(2) Noise Term



Output 33.10.7 Summary of Breaks in f_t

Elementwise Break Summary for tfstate

ID	Element Index	Z Value	Pr > |z|
264	1	-5.03	<.0001
199	1	4.62	<.0001
198	1	-3.15	0.0016

Example 33.11: Panel Data: Dynamic Panel Model for the Cigar Data

This example shows how you can use the SSM procedure to specify and fit the so-called dynamic panel model, which is commonly used to analyze a panel of time series. Suppose that a panel of time series $y_{t,i}$ follows the model

$$y_{t,i} = \rho y_{(t-1),i} + \mu_i + \beta X_{t,i} + \zeta_t + \epsilon_{t,i}$$

where t denotes the time index (for example, $t = 1, \ldots, T$); i denotes the panel index (for example, $i = 1, \ldots, P$); ρ is the autoregression coefficient; μ_i denote the panel-specific intercepts; $X_{t,i}$ are observations on a regression variable with regression coefficient β (the same for all panels); ζ_t are unobserved, random time effects; and $\epsilon_{t,i}$ are the observation errors. The sequences ζ_t and $\epsilon_{t,i}$ are assumed to be independent, zero-mean Gaussian variables with variances σ_1^2 and σ_0^2, respectively. This is an example of a dynamic panel model that contains one regressor variable. It is easy to formulate this model equation as a state equation with state $\boldsymbol{\alpha}_t$ of size P—the number of panels. Taking $y_{t,i} = \boldsymbol{\alpha}_t[i]$, it is easy to see that the states $\boldsymbol{\alpha}_t$ evolve according to the equation

$$\boldsymbol{\alpha}_{t+1} = \mathbf{T}\boldsymbol{\alpha}_t + \mathbf{W}_{t+1}\boldsymbol{\beta} + \boldsymbol{\eta}_{t+1}$$

where $\mathbf{T} = \rho I_P$ (a P-dimensional, diagonal matrix with all its diagonal elements equal to ρ); $\mathbf{W}_t = (\mathbf{X}_t \;\; I_P)$ is a $P \times (1 + P)$-dimensional matrix (in a block form) of state regression variables, where the first block is a column that includes all the values $X_{t,i}$ that are associated with a given time index (t) and the second block is a P-dimensional identity matrix; $\boldsymbol{\beta} = (\beta \; \mu_1, \ldots, \mu_P)'$ is the $(1 + P)$-dimensional column vector of regression coefficients; and $\boldsymbol{\eta}_t = (\zeta_t + \epsilon_{t,1}, \ldots, \; \zeta_t + \epsilon_{t,P})'$ is a P-dimensional column vector of all the disturbances that are associated with time index t. Because ζ_t and $\epsilon_{t,i}$ are independent, the covariance matrix of $\boldsymbol{\eta}_t$—for example, $\mathbf{Q}_t$—is easy to calculate: $\mathbf{Q}_t[i,i] = \sigma_0^2 + \sigma_1^2$ and, for $i \neq j$, $\mathbf{Q}_t[i,j] = \sigma_1^2$. This formulation can be easily extended to multiple regression variables, such as r variables, by appropriately modifying the term that is associated with the state regression variables—$\mathbf{W}_t\boldsymbol{\beta}$: the new $\mathbf{W}_t$ matrix becomes $P \times (r + P)$-dimensional and the new regression vector $\boldsymbol{\beta}$ becomes $(r + P)$-dimensional.

The cross-sectional data, Cigar, that are used in the section "Getting Started: SSM Procedure" on page 2394 are reused in this example. In order to use the SSM procedure to perform the dynamic panel model–based analysis, the input data set must be reorganized so that it contains the variables that form the $P \times (r + P)$-dimensional matrix $\mathbf{W}_t$. For the Cigar data, the number of panels $P = 46$ (the number of regions considered in the study), and the number of regression variables $r = 3$. Therefore, the input data set needs to be augmented by $46 * (3 + 46) = 2{,}254$ variables that constitute the matrix $\mathbf{W}_t = (\mathbf{X}_t \quad I_{46})$—the first 46×3-dimensional block $\mathbf{X}_t$ contains the values of the three regression variables, lprice, lndi, and lpimin, at a given time index (a particular year in this case). The following DATA steps accomplish this task in two steps. In the first step, the raw data that form the rows of the Cigar data set are read into a temporary data set, Tmp, such that all 6*46 = 276 values that are associated with a given year (values of six variables—year, region, lsales, lprice, lndi, and lpimin for 46 panels in a given year) are read in a single row that consists of 276 columns. In the second step, the final input data set is formed by rearranging Tmp so that it contains the necessary variables in the proper order—year (the time index), region (the panel index), lsales (the response variable), and the variables that form the 46×49-dimensional $\mathbf{W}$ matrix (w1, ..., w2254).

```
data Tmp;
    input u1-u276;
datalines;
63 1 4.54223 3.35341 7.3514 3.26194
63 2 4.82831 3.17388 7.5729 3.21487
63 3 4.63860 3.29584 7.3000 3.25037

   ... more lines ...


data cigar(keep=year region lsales w1-w2254);
   array wmat{46, 49} w1-w2254;
   array ivar{46, 6} u1-u276;
   set tmp;
   year = intnx( 'year', '1jan63'd, u1-63 );
   format year year.;
   do i=1 to 46;
      region = ivar[i, 2];
      lsales = ivar[i, 3];
      do j=1 to 46;
          do k=1 to 49;
              wmat[j,k] = 0;
              if k = j+3 then wmat[j,k] = 1;
              if k=1 then wmat[j,k] = ivar[j, 4];
              if k=2 then wmat[j,k] = ivar[j, 5];
              if k=3 then wmat[j,k] = ivar[j, 6];
          end;
      end;
      output;
   end;
run;
```

The following statements specify and fit the dynamic panel model:

```
proc ssm data=Cigar opt(tech=dbldog maxiter=75);
    id year interval=year;
    parms rho / lower=-0.9999 upper=0.9999;
    parms sigma0 sigma1 / lower=1.e-8;
    array RegionArray{46} region1-region46;
    do i=1 to 46;
       RegionArray[i] = (region=i);
    end;
    array cov{46,46};
    do i=1 to 46;
        do j=1 to 46;
           if(i=j) then cov[i,j] = sigma0 + sigma1;
           else cov[i,j] = sigma1;
        end;
    end;
    state panelState(46) T(I)=(rho) W(g)=(w1-w2254)
      cov(g)=(cov) a1(46) checkbreak;
    comp dynPanel = (RegionArray)*panelState;
    model lsales = dynPanel;
    output out=for1 press;
run;
```

The estimates of the regression coefficients and the regional intercepts, which are all statistically significant, are shown in Output 33.11.1. In particular, the estimated coefficients of lprice, lndi, and lpimin, are –0.26, 0.13, and 0.07, respectively.

Output 33.11.1 Estimates of β_1, β_2, β_3 and the Regional Intercepts

The SSM Procedure

Estimate of the State Equation Regression Vector

State	Element Index	Estimate	Standard Error	t Value	Pr > |t|
panelState	1	-0.2627	0.0178	-14.79	<.0001
panelState	2	0.1340	0.0130	10.30	<.0001
panelState	3	0.0748	0.0198	3.78	0.0002
panelState	4	0.4265	0.0581	7.35	<.0001
panelState	5	0.3825	0.0605	6.32	<.0001
panelState	6	0.4425	0.0582	7.61	<.0001
panelState	7	0.3471	0.0631	5.50	<.0001
panelState	8	0.3686	0.0635	5.81	<.0001
panelState	9	0.4357	0.0614	7.10	<.0001
panelState	10	0.3753	0.0655	5.73	<.0001
panelState	11	0.4249	0.0606	7.01	<.0001
panelState	12	0.4185	0.0604	6.92	<.0001
panelState	13	0.3824	0.0602	6.35	<.0001
panelState	14	0.3942	0.0644	6.12	<.0001
panelState	15	0.4154	0.0626	6.64	<.0001
panelState	16	0.3961	0.0610	6.49	<.0001
panelState	17	0.3765	0.0618	6.10	<.0001
panelState	18	0.4528	0.0608	7.44	<.0001
panelState	19	0.4316	0.0586	7.36	<.0001
panelState	20	0.4357	0.0601	7.25	<.0001
panelState	21	0.3771	0.0639	5.90	<.0001
panelState	22	0.3939	0.0629	6.26	<.0001
panelState	23	0.4122	0.0621	6.64	<.0001
panelState	24	0.3949	0.0605	6.52	<.0001
panelState	25	0.4386	0.0565	7.77	<.0001
panelState	26	0.4118	0.0627	6.57	<.0001
panelState	27	0.3898	0.0604	6.45	<.0001
panelState	28	0.3818	0.0613	6.23	<.0001
panelState	29	0.4343	0.0632	6.87	<.0001
panelState	30	0.4619	0.0625	7.39	<.0001
panelState	31	0.3730	0.0636	5.86	<.0001
panelState	32	0.3784	0.0589	6.43	<.0001
panelState	33	0.3825	0.0625	6.12	<.0001
panelState	34	0.3784	0.0598	6.32	<.0001
panelState	35	0.4093	0.0628	6.52	<.0001
panelState	36	0.4155	0.0597	6.96	<.0001
panelState	37	0.3960	0.0615	6.44	<.0001
panelState	38	0.4075	0.0602	6.77	<.0001
panelState	39	0.4045	0.0586	6.91	<.0001
panelState	40	0.3918	0.0599	6.55	<.0001
panelState	41	0.4350	0.0608	7.15	<.0001
panelState	42	0.4007	0.0602	6.65	<.0001
panelState	43	0.3196	0.0597	5.36	<.0001
panelState	44	0.4337	0.0609	7.12	<.0001
panelState	45	0.3790	0.0634	5.98	<.0001

Output 33.11.1 *continued*

The SSM Procedure

Estimate of the State Equation Regression Vector

State	Element Index	Estimate	Standard Error	t Value	Pr > |t|
panelState	46	0.3767	0.0618	6.10	<.0001
panelState	47	0.4392	0.0597	7.36	<.0001
panelState	48	0.3932	0.0604	6.51	<.0001
panelState	49	0.3938	0.0616	6.40	<.0001

Output 33.11.2 shows the estimates of the autoregression coefficient ρ, the observation error variance σ_0^2, and the variance of the time effect (variance of ζ) σ_1^2.

Output 33.11.2 Estimates of ρ, σ_0^2, and σ_1^2

Estimates of Named Parameters

Parameter	Estimate	Standard Error	t Value
rho	0.831679	0.0124338	66.89
sigma0	0.001231	0.0000491	25.08
sigma1	0.000213	0.0000662	3.22

Finally, you can compare the fit of the dynamic panel model with the fit of the model that is discussed in the section "Getting Started: SSM Procedure" on page 2394. Output 33.11.3 shows the likelihood-based information criteria for the dynamic panel model, and Output 33.11.4 shows the same information for the other model.

Output 33.11.3 Likelihood-Based Information Criteria: Dynamic Panel Model

Information Criteria

Statistic	Diffuse Likelihood Based	Profile Likelihood Based
AIC (lower is better)	-4732.722	-4856.398
BIC (lower is better)	-4717.247	-4343.874
AICC (lower is better)	-4732.704	-4841.250
HQIC (lower is better)	-4726.913	-4664.667
CAIC (lower is better)	-4714.247	-4245.874

Output 33.11.4 Likelihood-Based Information Criteria: Getting Started Example

Information Criteria		
Statistic	Diffuse Likelihood Based	Profile Likelihood Based
AIC (lower is better)	-4488.093	-4145.246
BIC (lower is better)	-4477.776	-3637.952
AICC (lower is better)	-4488.084	-4130.417
HQIC (lower is better)	-4484.220	-3955.472
CAIC (lower is better)	-4475.776	-3540.952

Similarly, Output 33.11.5 shows fit criteria based on the delete-one cross validation error for the dynamic panel model, and Output 33.11.6 shows the same information for the other model.

Output 33.11.5 Delete-One Cross Validation Criteria: Dynamic Panel Model

Delete-One Cross Validation Error Criteria			
Variable	N	PRESS	Generalized Cross-Validation
lsales	1380	1.115309	5.62798E-7

Output 33.11.6 Delete-One Cross Validation Criteria: Getting Started Example

Delete-One Cross Validation Error Criteria			
Variable	N	PRESS	Generalized Cross-Validation
lsales	1380	1.290420	6.18144E-7

On the basis of both these considerations, the dynamic panel model appears to provide a better fit for the Cigar data than the model that is fit in the section "Getting Started: SSM Procedure" on page 2394.

Example 33.12: Multivariate Modeling: Long-Term Temperature Trends

In a presentation by Ansley and de Jong (2012), three monthly time series are jointly modeled to obtain long-term—several decades long—temperature predictions for certain regions of the northern hemisphere. This example shows how you can specify and fit the final model that this presentation proposed. The following data set, Temp, contains four variables: date dates the monthly observations; UAH contains monthly satellite global temperature readings, starting in December 1978; CRU contains monthly temperature data, starting in January 1850 (from a different source); and GISS contains monthly temperature data, starting in January 1880 (from yet another source). All these temperature data are scaled suitably so that the numbers represent temperature readings in centigrade.

```
data Temp;
input UAH CRU GISS @@;
date = intnx('month', '01jan1850'd, _n_-1);
format date date.;
datalines;
.    8.243    .
.    9.733    .

   ... more lines ...
```

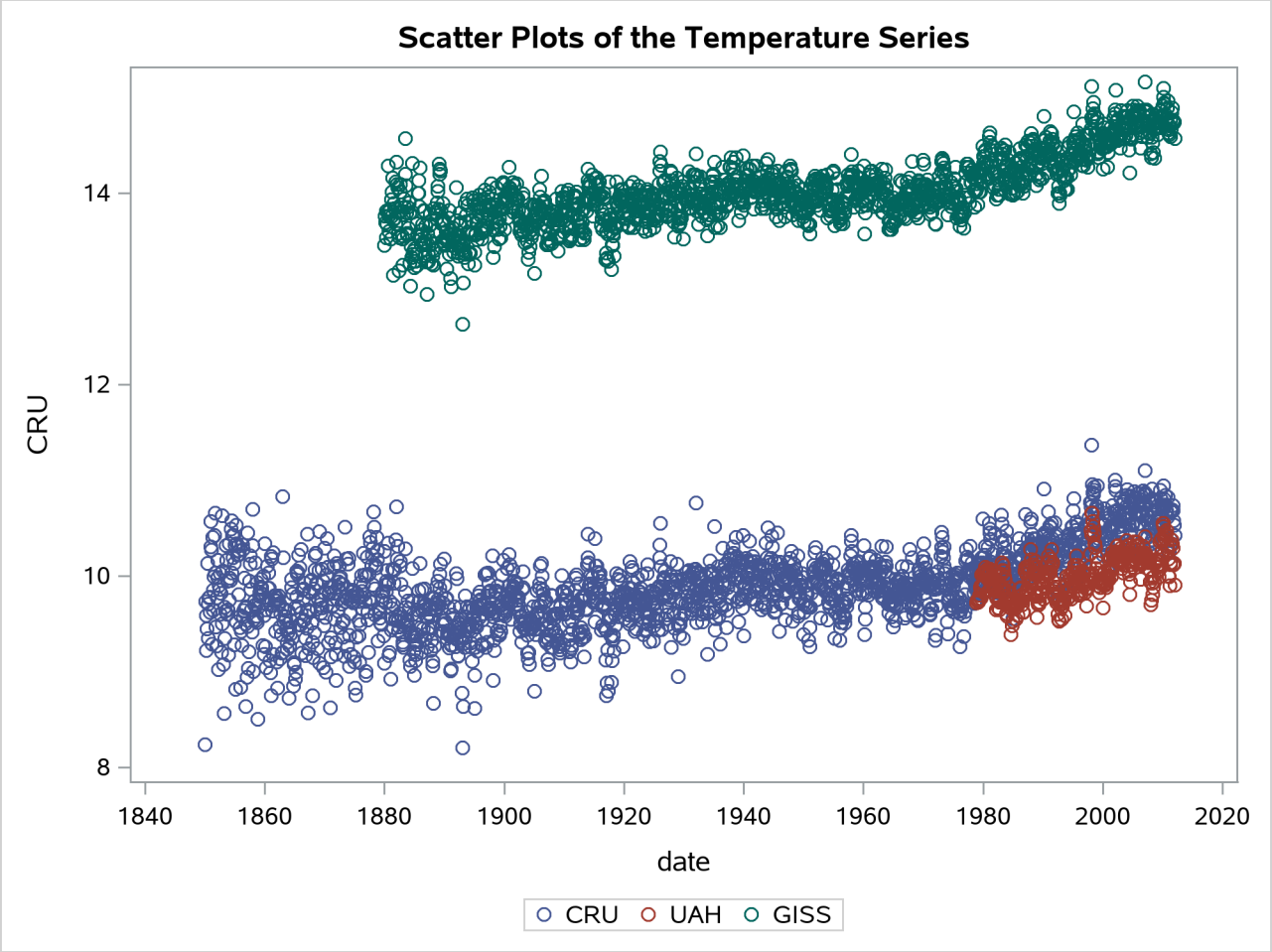
The following statements produce scatter plots of these three series in a single graph:

```
 proc sgplot data=Temp;
       title "Scatter Plots of the Temperature Series";
       scatter x=date y=cru;
       scatter x=date y=uah ;
       scatter x=date y=giss;
  run;
```

Output 33.12.1 shows the resulting graph. As already noted, these three series start at different points in the past. However, they all end at the same time: they all have measurements until January 2012, which is the last month in the data set. The mean levels of these series are different: the GISS measurements are generally larger than CRU and UAH by about 4 degrees. In addition, the variability in the CRU values seems to decrease with time (this is more apparent when the series is plotted by itself). The goal of the analysis is to use these data to make long-term predictions about future temperature levels. The following statements append 1,200 missing measurements to Temp, so that the model fitted by using the SSM procedure can be extrapolated to obtain temperature forecasts 100 years in the future:

```
data append(keep=date cru giss uah);
    do i=1 to 1200;
        cru = .; giss=.; uah=.;
        date = intnx('month','01jan2012'D, i);
    format date monyy7.;
    output;
    end;
run;

 proc append base=Temp data=Append; run;
```

Output 33.12.1 Scatter Plots of CRU, UAH, and GISS



Ansley and de Jong propose a parsimonious model that links these three time series. It can be described as

$$\begin{aligned} \mathrm{GISS}_t &= \mu_t + a\,\zeta_t + a\,r_1\,\epsilon_{1t} \\ \mathrm{CRU}_t &= \beta_{cru} + \mu_t + a\,\zeta_t + a\,\epsilon_{2t} \\ \mathrm{UAH}_t &= \beta_{uah} + \mu_t + a\,\zeta_t + a\,r_3\,\epsilon_{3t} \end{aligned}$$

where β_{cru} and β_{uah} are intercepts that are associated with CRU and UAH, respectively; μ_t is an integrated random walk trend; ζ_t is a zero-mean, autoregressive noise term (which is scaled by an unknown scaling factor a); and ϵ_{it} $(i = 1, 2, 3)$ are independent observation errors with different variances that are also scaled suitably. Note that the trend μ_t and the autoregressive noise term ζ_t are shared by the models of the three series, and, for identification purposes, the intercept for GISS is taken to be zero. In addition, the model parameters are assumed to be interrelated and are parameterized in a particular way (which leads to fewer parameters to estimate, and their relative scaling helps in parameter estimation). This special parameterization can be expressed as a function of seven basic parameters: *loga1*, *logr1*, *logr3*, *logsigma*, *b*, *c*, and *rhoParm* (this naming convention is different from that used by Ansley and de Jong (2012)).

Let $\sigma^2 = \exp(2\mathrm{logsigma})$, and let the scaling factors $a = \exp(\mathrm{loga1})$, $r_1 = \exp(\mathrm{logr1})$, and $r_3 = \exp(\mathrm{logr3})$. Then the model parameters can be described as follows:

- The parameters that are associated with the autoregression ζ_t: the damping factor $\rho = \frac{\exp(\text{rhoParm})-1}{\exp(\text{rhoParm})+1}$, the variance of the disturbance term = $a^2\sigma^2$, and the variance of the initial state = $a^2\sigma^2/(1-\rho^2)$
- The parameters that are associated with the integrated random walk trend: the variance of the disturbance term in the slope equation = σ^2
- The variance of the observation errors for GISS = $a^2 r_1^2 \sigma^2$
- The variance of the observation errors for UAH = $a^2 r_3^2 \sigma^2$
- The variance of the observation errors for CRU is taken to be time-varying with the following form: for $t \leq nobs$,

  $$\sigma_{2,t}^2 = a^2\sigma^2 \exp(2b + 2c\ t/nobs)$$

  where $nobs$ = 1,945 (the number of observations in the unappended data set). For $t >$ 1,945, it is fixed at its last value: $\sigma_{2,t}^2 = a^2\sigma^2 \exp(2b + 2c)$.

The following DATA step adds an observation index, tindex, to Temp, which is used in the SSM procedure to define the time-varying observation error variance for CRU:

```
data temp;
   set temp;
   tindex = _n_;
run;
```

The following statements fit the preceding model to the Temp data:

```
 ods output RegressionEstimates=regEst;
 proc ssm data=Temp;
     id date interval=month;
     parms loga1 logr1 logr3  logsigma;
     parms b=0 c=0;
     parms rhoParm;
     rho = (exp(rhoParm)-1)/(exp(rhoParm)+1);
     sigmaSq = exp(2*logsigma);
     initSigmaSq = sigmaSq/(1-rho*rho);
     a1 = exp(loga1);
     a1Sq = a1*a1;
     r1sq  = exp(2*logr1);
     r3sq  = exp(2*logr3);
     giss_var = a1Sq*r1sq*sigmaSq;
     nobs=1945;
     if tindex <= nobs then
         cru_var = a1Sq*exp(2*b + 2*c*tindex/nobs)*sigmaSq;
     else cru_var = a1Sq*exp(2*b + 2*c)*sigmaSq;
     uah_var = a1Sq*r3sq*sigmaSq;
     UAH_Intercept=1.0;
     CRU_Intercept=1.0;
     trend level(ll) variance=0 slopevar=sigmaSq;
     state auto(1) T(g)=(rho) cov(g)=(sigmaSq)  cov1(g)=(initSigmaSq);
     comp auto_common = auto*(a1);
     state wn(3) type=wn cov(d)=(giss_var cru_var uah_var);
     comp wn_giss = wn[1];
```

```
      comp wn_cru = wn[2];
      comp wn_uah = wn[3];
      model GISS = level auto_common  wn_giss;
      model CRU = CRU_Intercept level auto_common  wn_cru;
      model UAH = UAH_Intercept level auto_common  wn_uah;
      comp slope = level_state_*(0 1);
      output out=For pdv press;
  run;
```

Output 33.12.2 shows the estimated intercepts $\hat{\beta}_{cru}$ and $\hat{\beta}_{uah}$. As expected, they are quite close (see the scatter plots of CRU and UAH in Output 33.12.1).

Output 33.12.2 Estimated Intercepts for CRU and UAH

The SSM Procedure

Regression Parameter Estimates

Response Variable	Regression Variable	Estimate	Standard Error	t Value	Pr > |t|
CRU	CRU_Intercept	-4.10	0.00360	-1139.2	<.0001
UAH	UAH_Intercept	-4.48	0.00671	-666.48	<.0001

The estimates of the basic parameters that underlie the model parameters are shown in Output 33.12.3.

Output 33.12.3 Estimates of Basic Model Parameters

Estimates of Named Parameters

Parameter	Estimate	Standard Error	t Value
loga1	7.820	0.3819	20.48
logr1	-1.005	0.0884	-11.37
logr3	-0.231	0.0453	-5.09
logsigma	-9.682	0.3806	-25.44
b	0.737	0.0502	14.69
c	-1.403	0.0689	-20.35
rhoParm	1.432	0.0767	18.68

The following DATA steps add two variables (CRU_ADJ = CRU – $\hat{\beta}_{cru}$ and UAH_ADJ = UAH – $\hat{\beta}_{uah}$) to the output data set For. These adjusted versions of CRU and UAH have the same mean level as GISS—estimated μ_t.

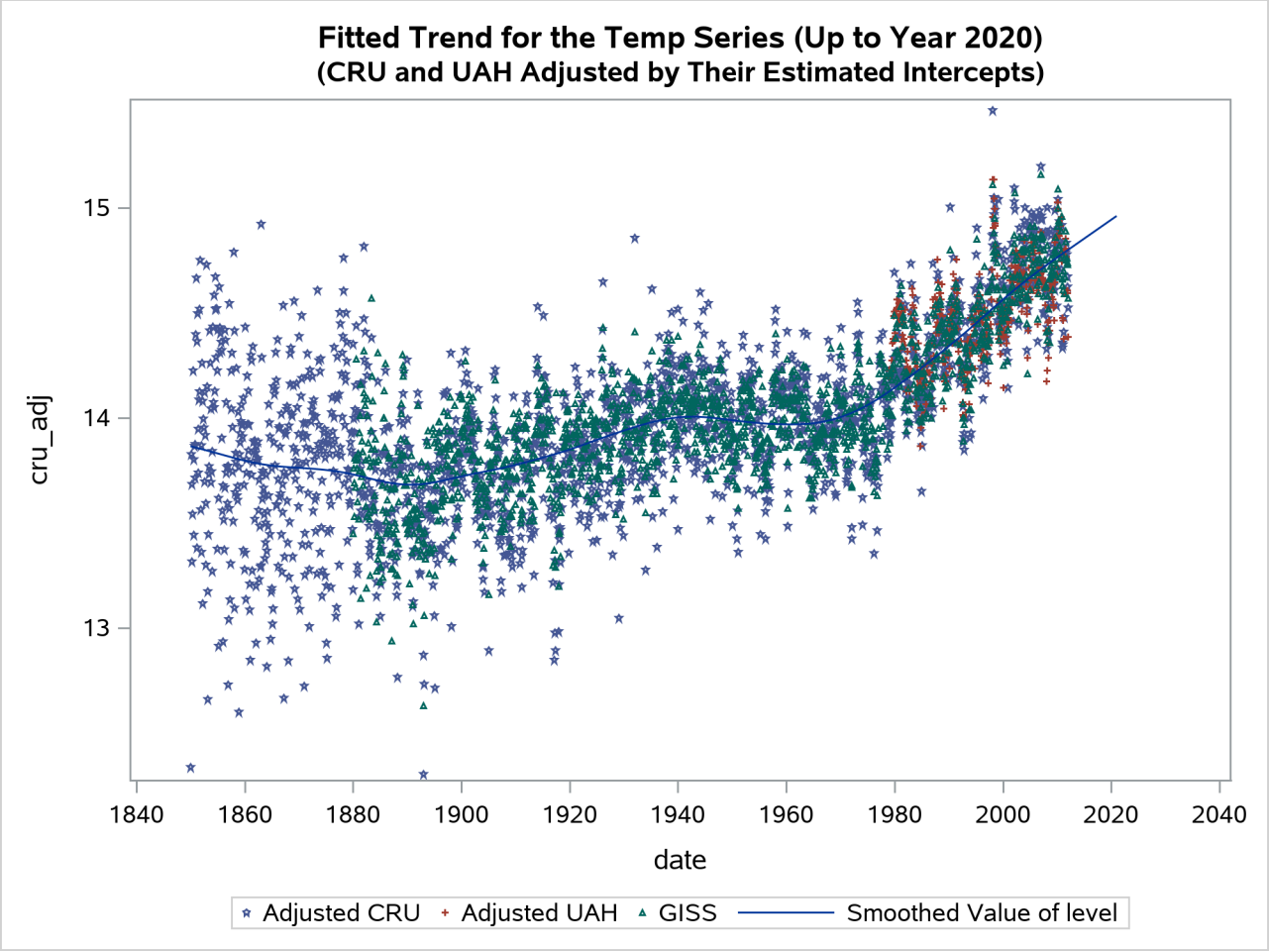
```
data _NULL_;
   set regEst;
   if _n_ = 1 then call symput('intercept1',trim(left(estimate)));
   else call symput('intercept2',trim(left(estimate)));
run;
data for;
    set For;
    cru_adj = cru - &intercept1;
    uah_adj = uah - &intercept2;
run;
```

The following statements produce a graph that contains four plots: scatter plots of GISS, CRU_ADJ, and UAH_ADJ and a series plot of the estimated μ_t.

```
proc sgplot data=For;
   where date < '01feb2021'd;
   title "Fitted Trend for the Temp Series (Up to Year 2020) ";
   title2 "(CRU and UAH Adjusted by Their Estimated Intercepts) ";
   scatter x=date y=cru_adj / LEGENDLABEL="Adjusted CRU"
       MARKERATTRS=GraphData1(symbol=star size=3);
   scatter x=date y=uah_adj / LEGENDLABEL="Adjusted UAH"
       MARKERATTRS=GraphData2(symbol=plus size=3);
   scatter x=date y=giss /
       MARKERATTRS=GraphData3(symbol=triangle size=3);
   series x=date y=smoothed_level / MARKERATTRS=GraphData4;
run;
```

Output 33.12.4 shows the resulting graph. It shows that the estimated mean level μ_t tracks the observed data quite well.

Output 33.12.4 Fitted Trend μ_t

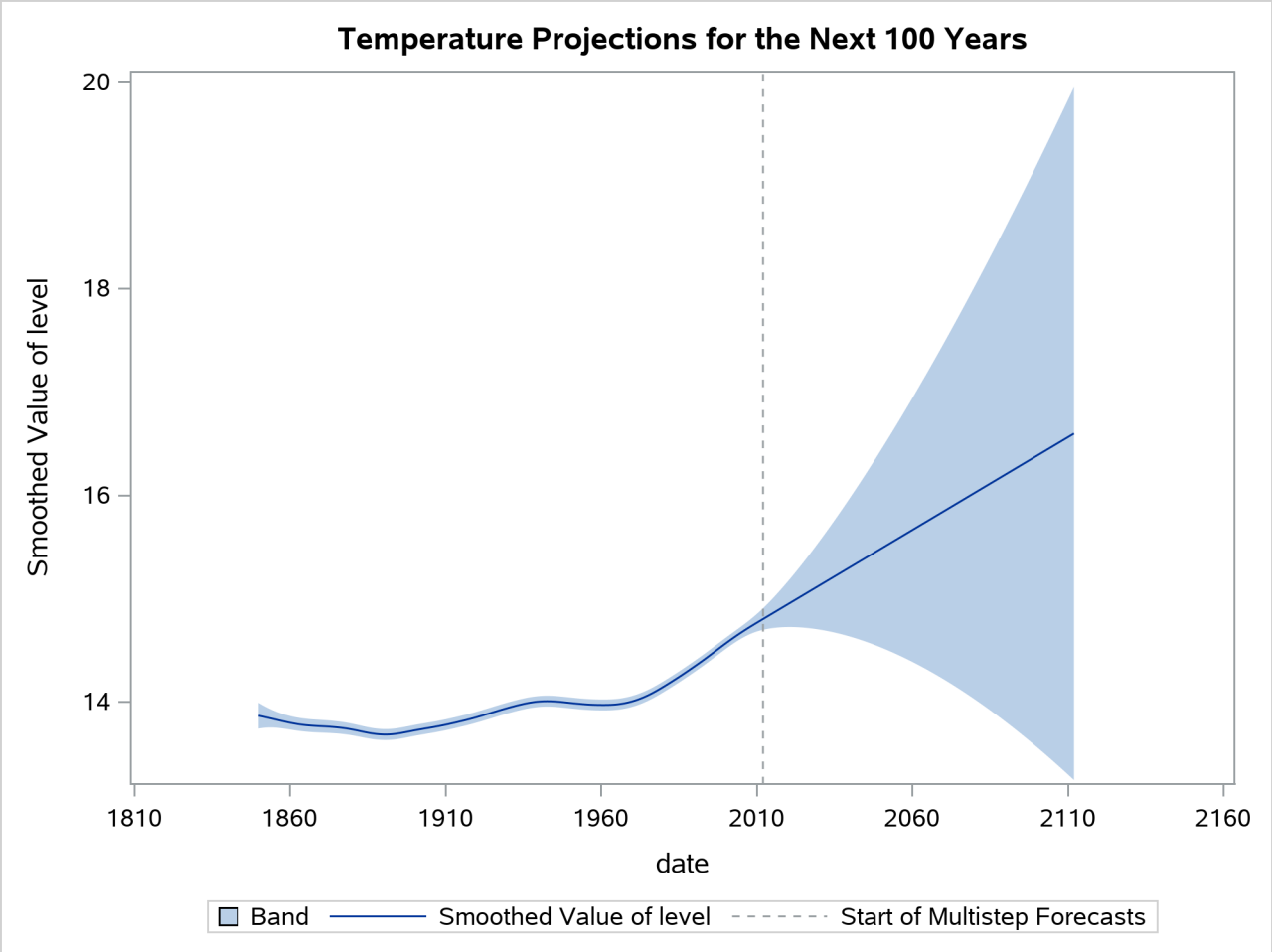


The following statements produce the time series plot of the estimated μ_t along with the 95% confidence band:

```
proc sgplot data=For;
   title "Temperature Projections for the Next 100 Years";
   band x=date lower=smoothed_lower_level upper=smoothed_upper_level;
   series x=date y=smoothed_level;
   refline '01feb2012'd / axis=x lineattrs=(pattern=shortdash)
      LEGENDLABEL= "Start of Multistep Forecasts"
      name="Forecast Reference Line";
run;
```

Output 33.12.5 shows the resulting graph.

Output 33.12.5 Long-Term Forecasts of μ_t



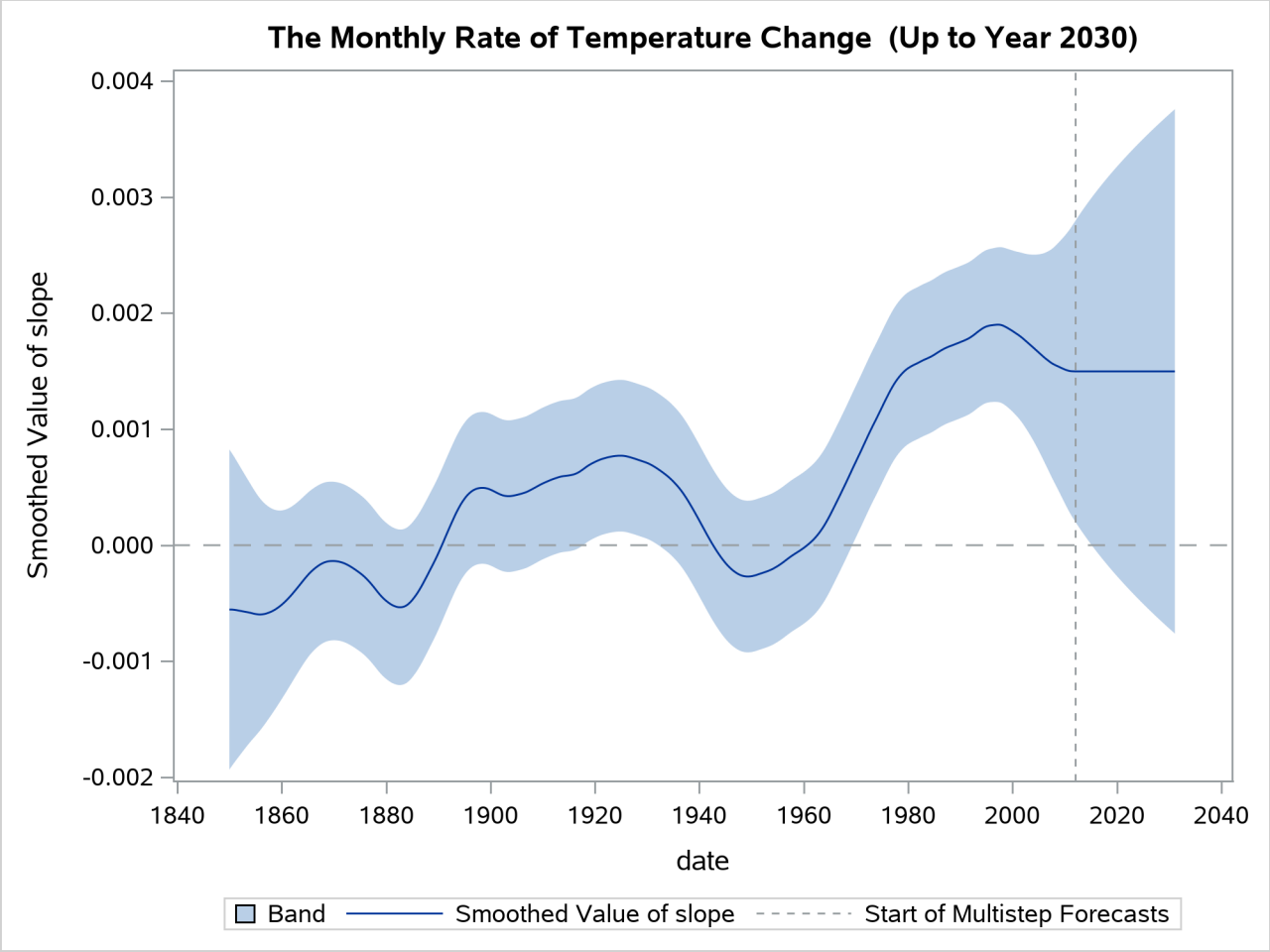
The following statements produce a similar graph for the estimated slope of μ_t:

```
proc sgplot data=For;
   where date <=  '01feb2031'd;
   title "The Monthly Rate of Temperature Change  (Up to Year 2030)";
   band x=date lower=smoothed_lower_slope upper=smoothed_upper_slope;
   series x=date y=smoothed_slope;
   refline '01feb2012'd / axis=x lineattrs=(pattern=shortdash)
      LEGENDLABEL= "Start of Multistep Forecasts"
```

```
        name="Forecast Reference Line";
    refline 0 / axis=y lineattrs=(pattern=dash);
run;
```

Output 33.12.6 shows the resulting plot of the estimated slope of μ_t.

Output 33.12.6 Forecasts of the Slope of μ_t



Based on the preceding analysis (see the plots of μ_t and its slope in Output 33.12.5 and Output 33.12.6), it appears that there has been statistically significant warming over the last 10 years, but the warming does not appear to be accelerating.

Example 33.13: Bivariate Model: Sales of Mink and Muskrat Furs

This example considers a bivariate time series of logarithms of the annual sales of mink and muskrat furs by the Hudson's Bay Company for the years 1850–1911. These data have been analyzed previously by many authors, including Chan and Wallis (1978); Harvey (1989); Reinsel (1997). There is known to be a predator-prey relationship between the mink and muskrat species: minks are principal predators of muskrats. Previous analyses for these data generally conclude the following:

- An increase in the muskrat population is followed by an increase in the mink population a year later, and an increase in the mink population is followed by a decrease in the muskrat population a year later.
- Because muskrats are not the only item in the mink diet and because both mink and muskrat populations are affected by many other factors, the model must include additional terms to explain the year-to-year variation.

The analysis in this example, which loosely follows the discussion in Harvey (1989, chap. 8, sec. 8), also leads to similar conclusions. It begins by taking Harvey's model 8.8.8 (a and b), with autoregressive order one, as the starting model—that is, it assumes that the bivariate (mink, muskrat) process $\mathbf{Y}_t$ satisfies the following relationship:

$$\boldsymbol{\mu}_t = \boldsymbol{\mu}_{t-1} + \boldsymbol{\beta} + \boldsymbol{\nu}_t$$
$$\mathbf{Y}_t = \boldsymbol{\mu}_t + \boldsymbol{\Phi}\mathbf{Y}_{t-1} + \boldsymbol{\xi}_t$$

This model postulates that $\mathbf{Y}_t$ can be expressed as a sum of three terms: $\boldsymbol{\mu}_t$, a bivariate trend that is modeled as a random walk with drift $\boldsymbol{\beta}$; $\boldsymbol{\Phi}\mathbf{Y}_{t-1}$, an AR(1) correction; and $\boldsymbol{\xi}_t$, a bivariate Gaussian white noise disturbance. It is assumed that the AR coefficient matrix $\boldsymbol{\Phi}$ is stable (that is, its eigenvalues are less than 1 in magnitude) and that the bivariate disturbances $\boldsymbol{\nu}_t$ (white noise associated with $\boldsymbol{\mu}_t$) and $\boldsymbol{\xi}_t$ are mutually independent.

The following statements show how you can specify this model in the SSM procedure:

```
proc ssm data=furs plots=residual;

   /* Specify the ID variable */
   id year interval=year;

   /* Define parameters */
   parms rho1 rho2/ lower=-0.9999 upper=0.9999;
   parms msd1 msd2 esd1 esd2 / lower=1.e-6;

   /* Specify the terms with lagged response variables */
   deplag LagsForMink(LogMink) LogMink(lags=1) LogMusk(lags=1);
   deplag LagsForMusk(LogMusk) LogMink(lags=1) LogMusk(lags=1);

   /* Specify the bivariate trend */
   array rwQ{2,2};
   rwQ[1,1] = msd1*msd1; rwQ[1,2] = msd1*msd2*rho1;
   rwQ[2,1] = rwQ[1,2]; rwQ[2,2]=msd2*msd2;
   state alpha(2) type=RW W(I) cov(g)=(rwQ);
   comp minkLevel = alpha[1];
   comp muskLevel = alpha[2];

   /* Specify the bivariate white noise */
   array wnQ{2,2};
   wnQ[1,1] = esd1*esd1; wnQ[1,2] = esd1*esd2*rho2;
   wnQ[2,1] = wnQ[1,2]; wnQ[2,2]=esd2*esd2;
   state error(2) type=WN cov(g)=(wnQ);
   comp minkWn = error[1];
   comp muskWn = error[2];

   /* Specify the observation equation */
```

```
   model LogMink = LagsForMink minkLevel minkWn;
   model LogMusk = LagsForMusk muskLevel muskWn;

   /* Specify an output data set to store component estimates */
   output out=salesFor press;
run;
```

The different parts of the program are explained as follows:

- The PARMS statements define parameters that are used to form the elements of Σ_1 (the covariance of $\boldsymbol{\nu}_t$, the disturbance term in the bivariate level equation) and Σ_2 (the covariance of $\boldsymbol{\xi}_t$, which is the bivariate white noise). Σ_1 is parameterized as (`msd1*msd1 msd1*msd2*rho1; msd1*msd2*rho1 msd2*msd2`). Σ_2 is similarly parameterized by using esd1, esd2, and rho2. In addition to ensuring that Σ_1 and Σ_2 are positive semidefinite, it turns out that this parameterization leads to an interpretable model at the end.
- The DEPLAG statements help define the terms that are associated with $\boldsymbol{\Phi}\mathbf{Y}_{t-1}$.
- The remaining statements are self-explanatory.

Output 33.13.1 shows the estimate of the drift vector $\boldsymbol{\beta}$ in the equation of $\boldsymbol{\mu}_t$ ($\boldsymbol{\mu}_t = \boldsymbol{\mu}_{t-1} + \boldsymbol{\beta} + \boldsymbol{\nu}_t$).

Output 33.13.1 Estimate of the Drift Vector $\boldsymbol{\beta}$

Estimate of the State Equation Regression Vector

State	Element Index	Estimate	Standard Error	t Value	Pr > |t|
alpha	1	-0.000817	0.0323	-0.03	0.9798
alpha	2	0.005953	0.0258	0.23	0.8175

Clearly, both elements of $\boldsymbol{\beta}$ are statistically insignificant, and the $\boldsymbol{\mu}_t$ equation can be simplified as $\boldsymbol{\mu}_t = \boldsymbol{\mu}_{t-1} + \boldsymbol{\nu}_t$. Next, Output 33.13.2 shows the estimates of the elements of Σ_1, and Σ_2, and Output 33.13.3 shows the estimates of the lag coefficients.

Output 33.13.2 Estimates of Σ_1, and Σ_2

Estimates of Named Parameters

Parameter	Estimate	Standard Error	t Value
rho1	0.8310	0.1377	6.03
rho2	-0.9999	1.6555	-0.60
msd1	0.2500	0.0354	7.06
msd2	0.1991	0.0592	3.36
esd1	0.0662	0.0597	1.11
esd2	0.1344	0.0527	2.55

Output 33.13.3 Estimates of Lag Coefficients (Elements of Φ)

Model Parameter Estimates

Component	Type	Parameter	Estimate	Standard Error	t Value
LagsForMink	Lag Coefficient Of LogMink	Lag[1]	-0.0011	0.173	-0.01
LagsForMink	Lag Coefficient Of LogMusk	Lag[1]	0.3349	0.137	2.45
LagsForMusk	Lag Coefficient Of LogMink	Lag[1]	-0.9905	0.142	-6.98
LagsForMusk	Lag Coefficient Of LogMusk	Lag[1]	0.6570	0.121	5.44

The main points of the output can be summarized as follows:

- phi11, the first element of Φ, which relates the current value of LogMink with its lagged value, is statistically insignificant. That is, lagged LogMink term could be dropped from the model equation for LogMink.
- rho2, the correlation coefficient between the elements of $\boldsymbol{\xi}_t$—the bivariate noise vector in the equation $\mathbf{Y}_t = \boldsymbol{\mu}_t + \boldsymbol{\Phi}\mathbf{Y}_{t-1} + \boldsymbol{\xi}_t$—is very near its lower boundary of –1 (in such cases the standard error of the parameter estimate is not reliable). This implies that the two elements of $\boldsymbol{\xi}_t$ are perfectly negatively correlated.

Taken together, these observations suggest the reduced model

$$\boldsymbol{\mu}_t = \boldsymbol{\mu}_{t-1} + \boldsymbol{\nu}_t$$
$$\mathbf{Y}_t = \boldsymbol{\mu}_t + \boldsymbol{\Phi}\mathbf{Y}_{t-1} + \boldsymbol{\xi}_t$$

where $\boldsymbol{\Phi} = (0 \quad \phi_{12};\ \phi_{21} \quad \phi_{22})$ and $\mathrm{Cov}(\boldsymbol{\xi}_t) = \Sigma_2$ is parameterized as (`esd1*esd1 -esd1*esd2; -esd1*esd2 esd2*esd2`). The program that produces the reduced model is a simple modification of the preceding program and is not shown.

Output 33.13.4 Estimates of Σ_1, and Σ_2 (Reduced Model)

Estimates of Named Parameters

Parameter	Estimate	Standard Error	t Value
rho1	0.8526	0.0978	8.71
msd1	0.2472	0.0282	8.78
msd2	0.1955	0.0365	5.36
esd1	0.0679	0.0385	1.76
esd2	0.1372	0.0328	4.18

Output 33.13.5 Estimates of Lag Coefficients (elements of Φ) (Reduced Model)

Model Parameter Estimates

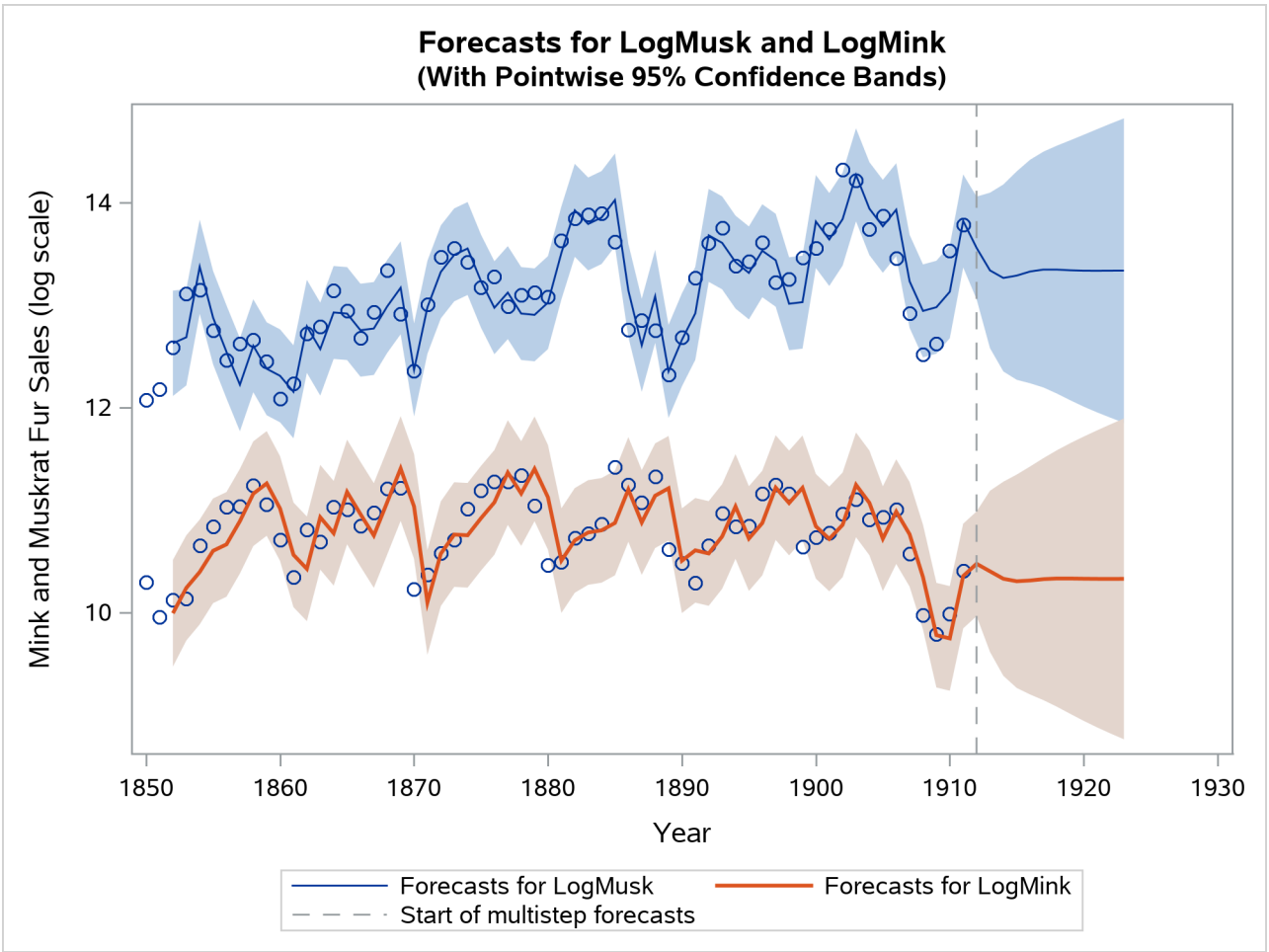
Component	Type	Parameter	Estimate	Standard Error	t Value
LagsForMink	Lag Coefficient Of LogMusk	Lag[1]	0.330	0.0986	3.35
LagsForMusk	Lag Coefficient Of LogMink	Lag[1]	-0.997	0.1168	-8.54
LagsForMusk	Lag Coefficient Of LogMusk	Lag[1]	0.668	0.1003	6.66

The tables in Output 33.13.4 and Output 33.13.5 show the new parameter estimates. By examining the parameter estimates, you can easily see that this model supports the general conclusions mentioned at the start of this example. In particular, note the following:

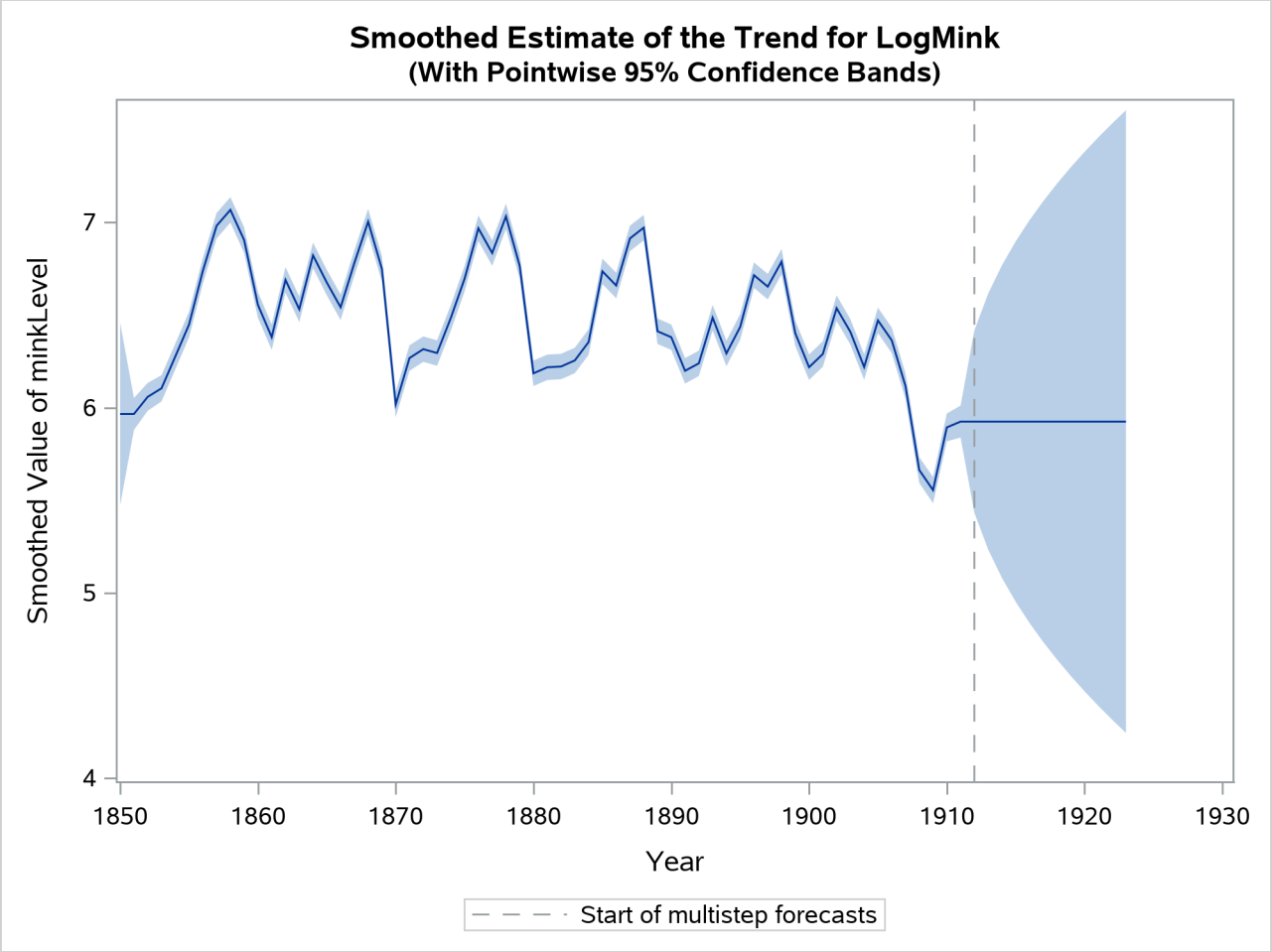
- $\hat{\phi}_{12} = 0.33$ implies that this year's mink abundance is positively correlated with last year's muskrat abundance.
- $\hat{\phi}_{21} = -0.99$ and $\hat{\phi}_{22} = 0.66$ imply that this year's muskrat abundance is negatively correlated with last year's mink abundance and positively correlated with last year's muskrat abundance.
- Even though the parameters were not restricted to ensure stability, the estimated $\mathbf{\Phi}$ turns out to be stable with a pair of complex eigenvalues, $0.317 \pm i\ 0.473$, and a modulus of 0.570 (these calculations are done separately by using the IML procedure).
- The fact that elements of $\boldsymbol{\xi}_t$ are perfectly negatively correlated further supports the predator-prey relationship.

Finally, Output 33.13.6 shows the plots of one-step-ahead and post-sample forecasts for LogMink and LogMusk, and Output 33.13.7 shows the plot of the smoothed (full-sample) estimate of the first element of $\boldsymbol{\mu}_t$: LogMink Trend.

Output 33.13.6 Forecasts for Mink and Muskrat Fur Sales in Logarithms



Output 33.13.7 Smoothed Estimate of LogMink Trend



Example 33.14: Factor Model: Now-Casting the US Economy

A well-known business conditions index, the Aruoba-Diebold-Scotti (ADS) business conditions index, is designed to track real business conditions at high frequency (for more information about this index, see http://www.philadelphiafed.org/research-and-data/real-time-center/business-conditions-index/). Its underlying (seasonally adjusted) economic indicators (weekly initial jobless claims, monthly payroll employment, industrial production, personal income less transfer payments, manufacturing and trade sales, and quarterly real GDP) blend high- and low-frequency information with stock and flow data. The ADS index is based on a rather elaborate state space model that takes into account the stock and flow nature of the underlying economic indicators. To simplify the illustration, this example uses the same economic indicators to develop a similar index by using a simpler factor model. You can also use PROC SSM to carry out the more elaborate modeling that underlies the ADS index. All these economic indicators are freely available from the Federal Reserve Economic Data (FRED). You can access these data by using the SASEFRED interface engine; see Chapter 48, "The SASEFRED Interface Engine." The names of analysis variables and the relevant information that is needed for using the SASEFRED engine to obtain these data are shown in Table 33.11. All variables are transformed versions of the original series: all, except l_icsa, are both logged and differenced; l_icsa is only logged. The input data set for the analysis,

named econ, is formed by merging these six series of different frequencies. This merging is done by treating them as daily series that have missing values on the days when the series values are not available—for example, ld_pinc is nonmissing only for the first day of the month. The econ data set contains one more variable, recession, which is an indicator variable for the recessionary periods. This time series is an interpretation of the US Business Cycle Expansions and Contractions data that are provided by the National Bureau of Economic Research (NBER) at http://www.nber.org/cycles/cyclesmain.html (also see http://research.stlouisfed.org/fred2/series/USRECDM). The variable recession is not used in modeling. It is used later to demonstrate the adequacy of the activity index that is created in this example.

Table 33.11 Analysis Variables and Their FRED IDs

Name	FRED ID	Frequency	Description
ld_payemp	PAYEMS	Monthly	Payroll employment
ld_pinc	W875RX1	Monthly	Real personal income excluding current transfer receipts
ld_mnfctr	CMRMTSPL	Monthly	Real manufacturing and trade industries sales
ld_indpro	INDPRO	Monthly	Industrial production index
ld_gdp	GDPC1	Quarterly	Real GDP
l_icsa	ICSA	Weekly	Initial jobless claims

For easier model description, the variables ld_payemp to ld_gdp are also denoted as y_{1t} to y_{5t}, and the variable l_icsa is denoted as y_{6t}. Using this notation, the following model is postulated for the six daily time series:

$$\mathbf{y}_{it} = \text{intercept}_i + \beta_i * \text{irw}_t + \epsilon_{it} \qquad 1 \le i \le 5$$
$$\mathbf{y}_{6t} = \mu_t + \beta_6 * \text{irw}_t + \epsilon_{6t}$$

A justification for this model is based on the following observations:

- The five time series y_{1t} to y_{5t} are logged and differenced versions of the underlying economic variables. Their plots (not shown here) show them to be hovering around a constant level, with some periods of deviation from this level. The plot of the sixth series, y_{6t}, which is logged but not differenced, shows a pronounced nonstationary pattern.

- All these series can be considered as proxies, possibly noisy, for the national economic activity. It is therefore reasonable to assume that a model for each of them will contain a common component, appropriately weighted, that is associated with the economic activity. In the current model this common component, named irw_t, is modeled as an integrated random walk. For y_{1t} to y_{5t}, the only other terms in the model are the respective intercepts, intercept_i, and the random disturbances, ϵ_{it}. Because y_{6t} shows a pronounced nonstationary pattern, its model has a time-varying level, μ_t, which is also modeled as an integrated random walk. For identifiability purposes, the initial condition for irw_t is taken to be 0. For the same reason, β_1, the coefficient of irw_t in the model for y_{1t} is taken to be 1.

- The underlying economic variables of the five time series y_{1t} to y_{5t} are positively correlated with the economic activity—for example, payroll employment is expected to increase with increased economic activity. On the other hand, y_{6t}, which is associated with the initial jobless claims, is negatively correlated with the economic activity. This means that, with β_1 taken to be 1, the estimates of $\beta_2, \ldots, \beta_5$ are expected to be positive and the estimate of β_6 is expected to be negative. In the factor modeling terminology, irw_t is called a factor and $\beta_1, \ldots, \beta_6$ are called the associated factor loadings.

The following statements show you how to specify this model in the SSM procedure:

```
ods output NamedParameterEstimates = named;
proc ssm data=econ opt(tech=activeset);
  id date interval=day;
  parms beta2-beta6;
  parms lv1-lv8;
  avar = exp(lv7);
  wnv1 = exp(lv1);  wnv2 = exp(lv2);
  wnv3 = exp(lv3);  wnv4 = exp(lv4);
  wnv5 = exp(lv5);  wnv6 = exp(lv6);
  tvar = exp(lv8);
  zero = 0;

   /* --- start of model spec ---*/
  state latent(2) t(g)=(1 1 0 1) cov(d)=(zero avar);
  comp c1 = latent[1];
  comp c2 = (beta2)*latent[1];
  comp c3 = (beta3)*latent[1];
  comp c4 = (beta4)*latent[1];
  comp c5 = (beta5)*latent[1];
  comp c6 = (beta6)*latent[1];

  irregular w1 variance=wnv1;
  int1 = 1;
  model ld_payemp = int1 c1 w1;

  irregular w2 variance=wnv2;
  int2 = 1;
  model ld_pinc = int2 c2 w2;

  irregular w3 variance=wnv3;
  int3 = 1;
  model ld_mnfctr = int3 c3 w3;

  irregular w4 variance=wnv4;
  int4 = 1;
  model ld_indpro = int4 c4 w4;

  irregular w5 variance=wnv5;
  int5 = 1;
  model ld_gdp = int5 c5 w5;

  irregular w6 variance=wnv6 ;
  trend t_icsa(ll) levelvar=0 slopevar=tvar;
  model l_icsa = c6 t_icsa w6;
  /* ---model spec done---*/

  eval icsaPattern = c6 + t_icsa;
  /*--index is a scaled version of the common factor--*/
  eval Index = 1000*c1;
  comp slope = latent[2];
  eval IndexSlope = 1000*slope;
  /*--just so recession is output to the output data set--*/
```

```
 rec = recession;
 output out=forecast1 press pdv;
run;
```

A few comments about the program:

- If the model and data are in reasonable accord, the default likelihood optimization settings work in most situations. However, in some cases the likelihood optimization process needs additional customization. Some experimentation with alternative optimization techniques and different parameterization of the model parameters can help. This example turns out to be one such case. The optimization technique ACTIVESET (`opt(tech=activeset)`) works better for WINDOWS and a few other platforms, whereas the default optimization technique works better for the AIX platform. In addition, the variances of all the disturbance terms in the model are parameterized in the exponential scale.

- The two-dimensional state that is associated with irw_t is named latent, and irw_t (the first element of latent) itself is named c1. Note that the second element of latent corresponds to the slope of irw_t. The components c2 to c6 correspond to $\beta_i * \text{irw}_t$ for $2 \leq i \leq 6$.

- The desired business index, named index, is a scaled version of irw_t (`eval Index = 1000*c1;`). This scaling is done purely for ease of display—the scaled values turn out to be in the range of –6.0 to 5.0. Another component, named IndexSlope, contains the slope of index, which is also a quantity of interest.

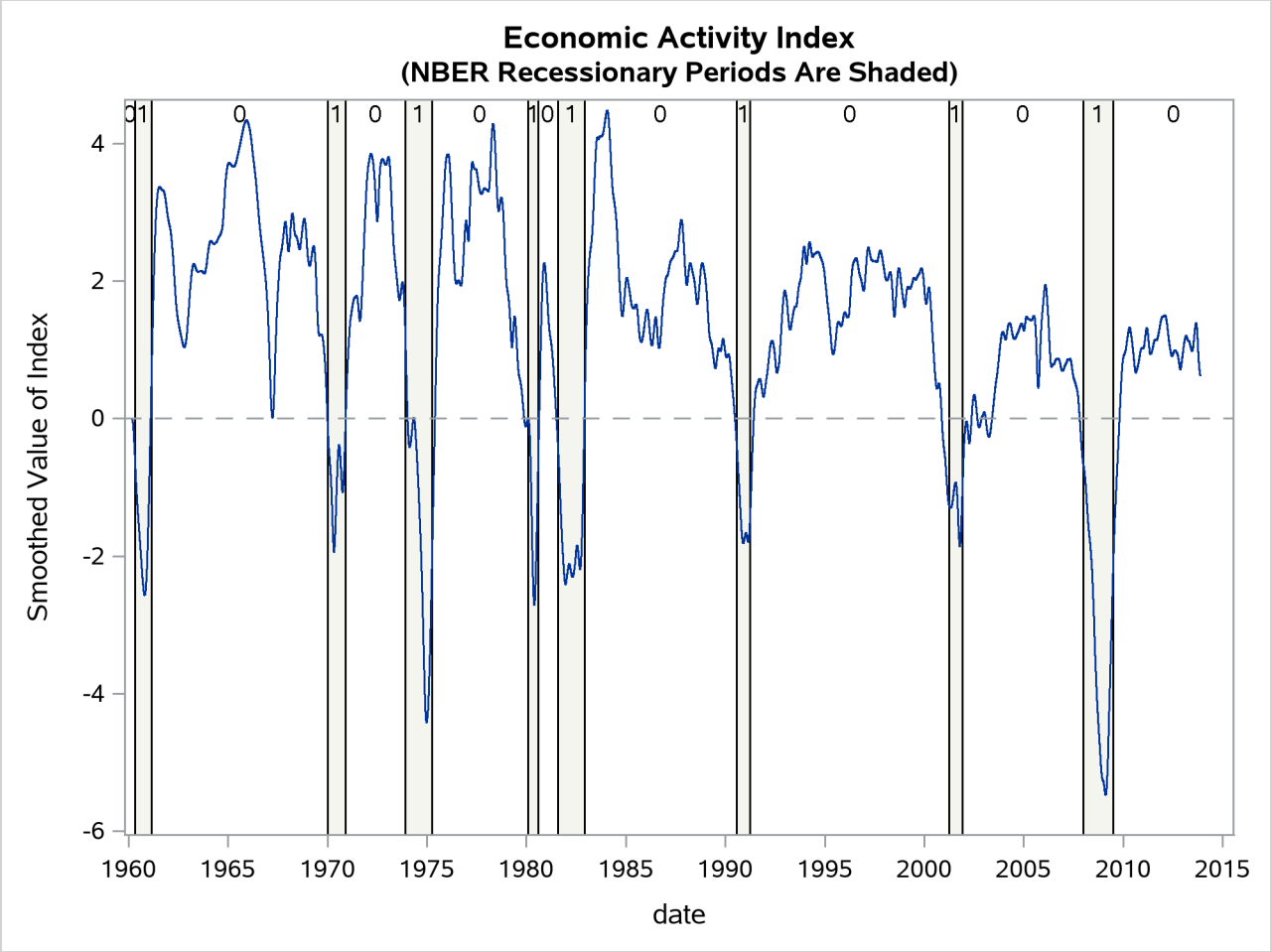
Output 33.14.1 Estimated Factor Loadings

The Estimated Loadings

Parameter	Estimate	StdErr	tValue
beta2	1.15	0.1276	8.98
beta3	1.96	0.2391	8.20
beta4	2.48	0.1646	15.08
beta5	3.27	0.2653	12.33
beta6	-96.46	9.6044	-10.04

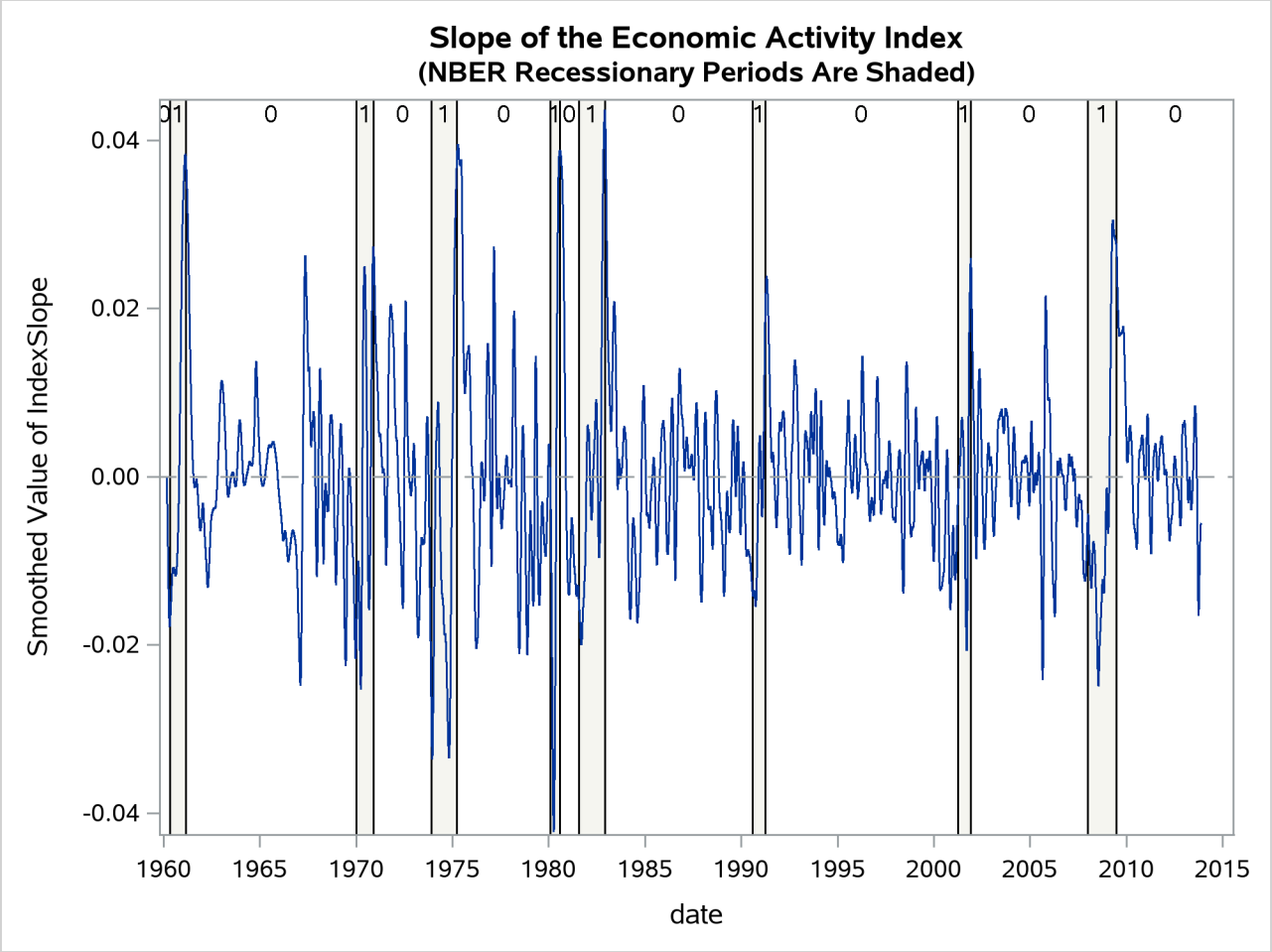
Output 33.14.1 shows the estimated factor loadings. They are statistically significant and their signs are consistent.

Output 33.14.2 The Estimate of Economic Activity Index



Output 33.14.2 shows the plot of the smoothed index. Note that it coheres quite well with the NBER recessionary periods. In Aruoba, Diebold, and Scotti (2009, sec. 4.4) the features of an earlier version of the ADS index are discussed in detail. Similar comments apply to this indicator also.

Output 33.14.3 Estimated Slope of the Economic Activity Index



Finally, Output 33.14.3 shows the plot of the slope of the index, which gives an idea of the direction of the economic activity.

Example 33.15: Longitudinal Data: Lung Function Analysis

The data for this example, which consist of 209 measurements of the lung function of an asthma patient, are analyzed in Wang (2013). The time series is measured mostly at two-hour time intervals but with irregular gaps. Wang (2013) fit a fourth-order continuous-time autoregressive model, CAR(4), to these data. The analysis results in a decomposition of the observed time series in three components:

- a slowly varying trend pattern, which appears to have a slight downward drift
- a diurnal component—a periodic pattern with a period of 24 hours
- a residual component

As shown in Wang (2013), the continuous-time autoregressive models can be formulated as state space models. However, in general, the form of such SSMs is quite complex. Consequently, specifying such a model by using the current SSM procedure syntax is impractical. On the other hand, you can analyze these types of longitudinal data by using continuous-time structural models, which are easy to specify in the SSM procedure. In this example, the lung function measurements, `y`, are modeled as

$$y_t = \text{intercept} + \beta * t + \zeta_t + \epsilon_t$$

where (intercept $+ \beta * t$) is a simple linear time trend, ζ_t is a continuous-time stochastic cycle, and ϵ_t is a Gaussian white noise sequence. Replacing the linear time trend with a more general time trend, such as a spline smoother, does not seem to change the fit, because the estimated smoothing spline turns out to be almost perfectly linear.

The following statements show you how to specify this model in the SSM procedure:

```
proc ssm data=asth;
   id time;
   state s1(1) type=cycle(CT) cov(g);
   comp c1 = s1[1];
   intercept = 1;
   irregular wn;
   model y= intercept time c1 wn;
   output out=for1 press;
   eval pattern=intercept+time+c1;
run;
```

The continuous-time stochastic cycle, named `c1`, is defined by a pair of STATE and COMPONENT statements. The STATE statement defines `s1` as a state subsection that is associated with a univariate, continuous-time cycle (signified by the use of **`type=cycle(CT)`**). The COMPONENT statement defines `c1` as its first element.

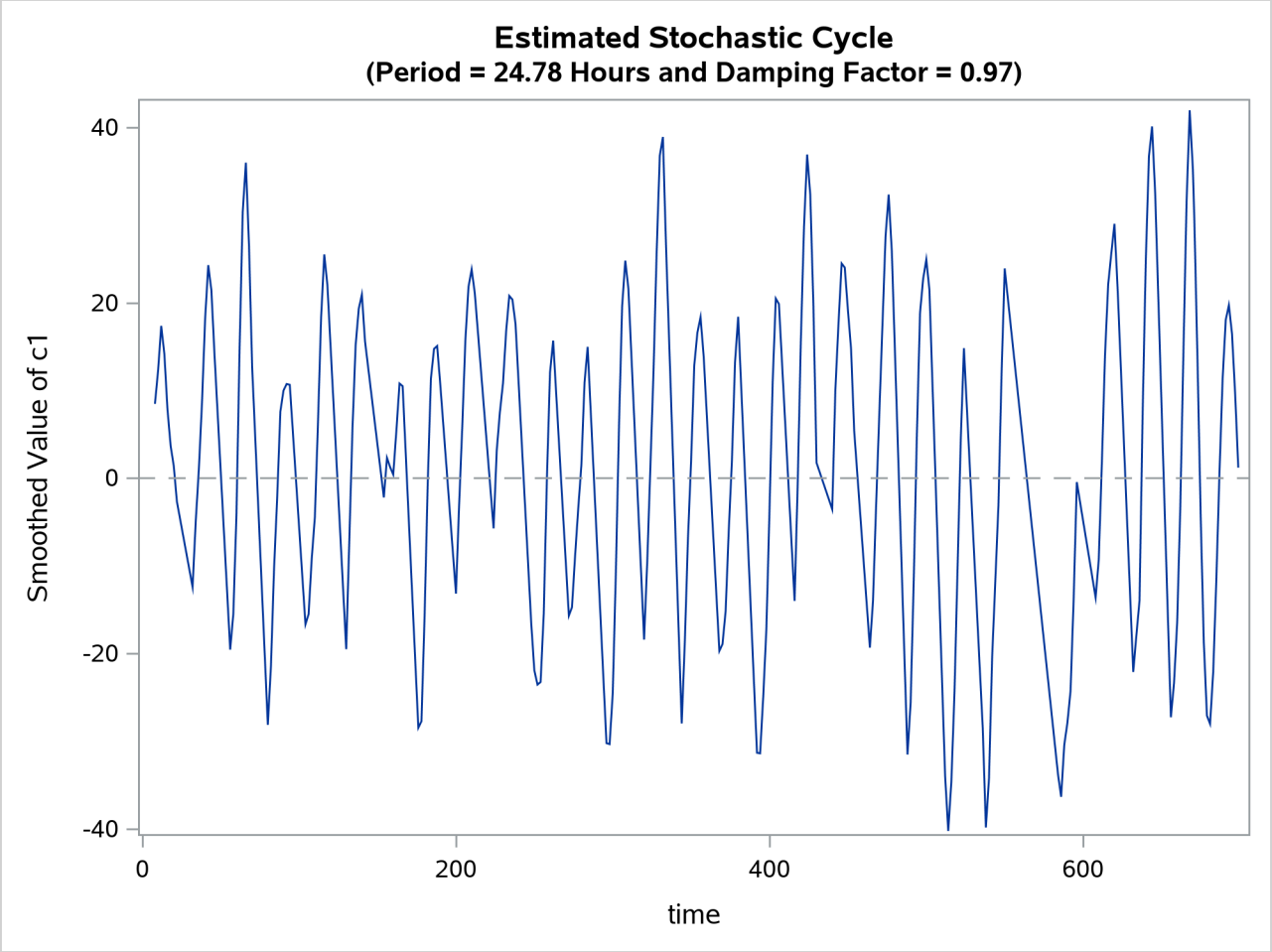
Output 33.15.1 Linear Time Trend: Estimates of Intercept and Slope

Regression Parameter Estimates

Response Variable	Regression Variable	Estimate	Standard Error	t Value	Pr > |t|
y	intercept	502.1637	3.50470	143.28	<.0001
y	time	-0.0201	0.00918	-2.19	0.0286

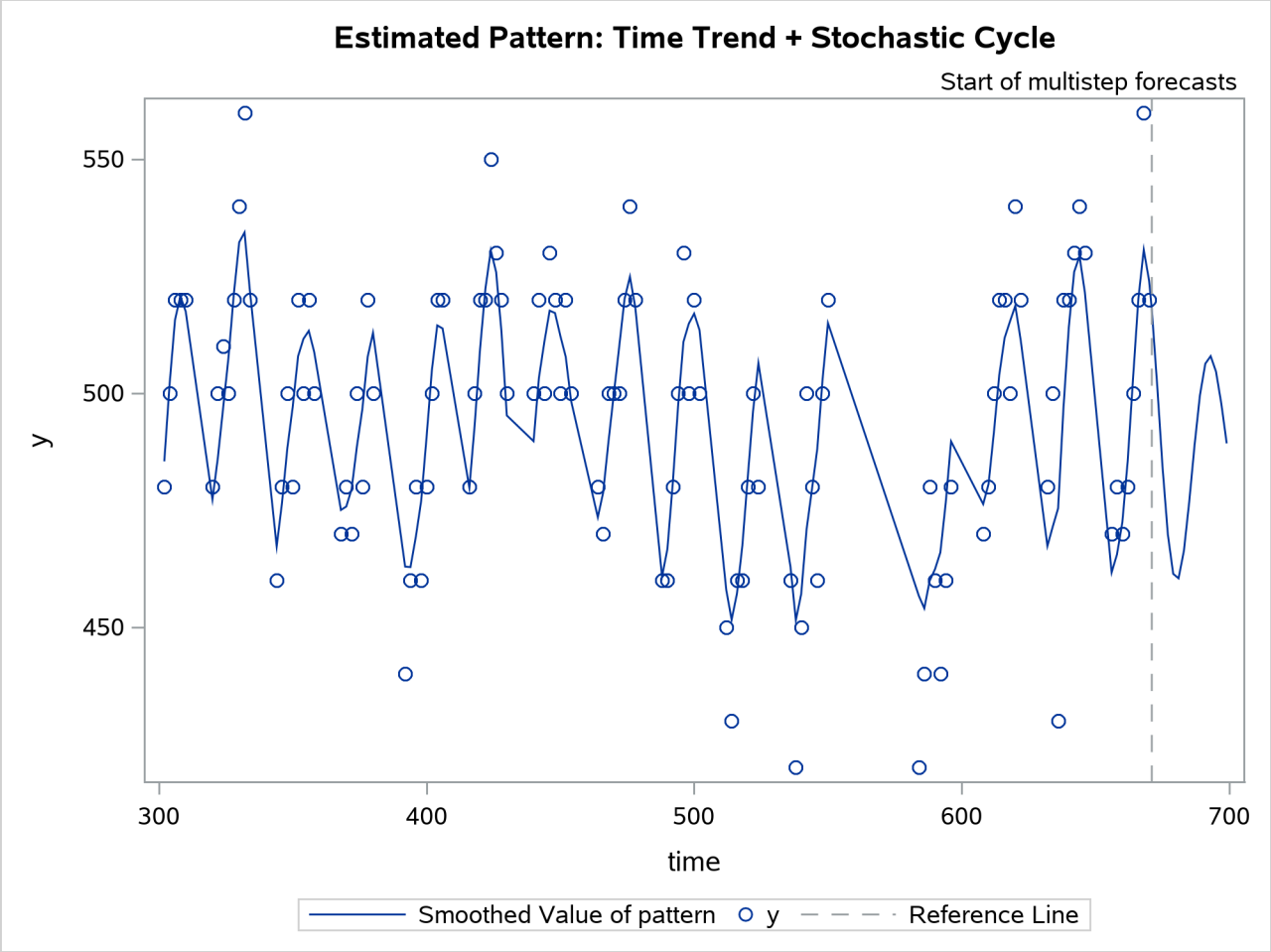
Output 33.15.1 shows the estimated intercept and slope of the time trend. The estimated slope (only marginally significant) is negative, which is consistent with the overall downward drift.

Output 33.15.2 Estimated Stochastic Cycle: ζ_t



Output 33.15.2 shows the plot of the estimated cycle component, which has a period of 24.78 hours and a damping factor of 0.97. That is, it is a nearly persistent diurnal cycle.

Output 33.15.3 Estimated Pattern: Intercept $+ \beta * t + \zeta_t$



Output 33.15.3 shows the fit of the de-noised y values (intercept $+ \beta * t + \zeta_t$). To reduce the clutter, only the second half of the data are plotted. The fit appears to be quite reasonable.

Example 33.16: Temporal Distribution: Estimating Monthly GDP (Experimental)

This example is based on a case study described in Pelagatti (2015, chap. 9, Example 9.2). The case study shows how you can estimate the monthly GDP (gross domestic product) for the United States by temporally distributing the quarterly GDP time series, which is readily available. The temporal distribution process is based on a bivariate model that relates two variables, the quarterly GDP and the monthly industrial production index (both for the United States). Assuming that t denotes the monthly time index, $indp_t$ denotes the monthly industrial production index series, and gdp_t denotes the quarterly GDP series that is organized as a monthly series (by setting the GDP numbers to missing for the months when they are not published), the case study uses the following model,

$$\begin{aligned} indp_t &= \mu_{1,t} + \psi_{1,t} + \epsilon_{1,t} \\ gdp_t^{\dagger} &= \mu_{2,t} + \psi_{2,t} + \epsilon_{2,t} \\ gdp_t &= gdp_t^{\dagger} + gdp_{t-1}^{\dagger} + gdp_{t-2}^{\dagger} \end{aligned}$$

where

- $gdp_t^{\dagger}$ is the unobserved monthly GDP series (which is to be estimated)
- $\boldsymbol{\mu}_t = (\mu_{1,t}\ \mu_{2,t})$ is a bivariate trend component that follows an integrated random walk
- $\boldsymbol{\psi}_t = (\psi_{1,t}\ \psi_{2,t})$ is a bivariate cycle component
- $\boldsymbol{\epsilon}_t = (\epsilon_{1,t}\ \epsilon_{2,t})$ is a bivariate white noise component

As explained in the section "Temporal Distribution" on page 2453, it is easy to fit this model by using the SSM procedure. As a first step, a data set, Useco, is created that organizes the monthly industrial production index and the quarterly GDP as monthly series. Specifically, Useco contains four variables: date dates the observations, indpro contains the monthly industrial production index, gdp contains the quarterly GDP, and startQ is a dummy variable that indicates the start of the quarter. This data set is essentially the same as the one that is used in the case study, except that, to improve the computational stability, gdp is scaled by 100. This data set has one peculiarity that is not mentioned in the case study: the GDP reporting pattern appears to have changed a few times over the years (October 1969, May 1992, and December 2014). The dummy variable, startQ, which indicates the start of the aggregation interval, is appropriately modified to take these changes into account. Output 33.16.1 shows the first few rows of Useco.

Output 33.16.1 First Few Rows of Useco

date	startQ	gdp	indpro
01JAN47	1	.	13.6351
01FEB47	0	.	13.7156
01MAR47	0	19.3447	13.7962
01APR47	1	.	13.6888
01MAY47	0	.	13.7425
01JUN47	0	19.3228	13.7425
01JUL47	1	.	13.6619
01AUG47	0	.	13.7425

The following statements show you how to specify the bivariate model for indpro and gdp:

```
proc ssm data=useco opt(maxiter=100);
   id date interval=month;
  /* Bivariate integrated random walk */
  state irwState(2) type=ll(slopecov(g));
  comp irwInd = irwState[1];
  comp irwGdp = irwState[2];
  /* Bivariate cycle */
  state cycle(2) type=cycle cov(g);
  comp cycInd = cycle[1];
  comp cycGdp = cycle[2];
  /* Bivariate white noise */
  state noise(2) type=wn cov(g);
  comp noiseInd = noise[1];
  comp noiseGdp = noise[2];
  /* Observation equations */
  model indpro = irwInd cycInd noiseInd;
  model gdp = irwGdp cycGdp noiseGdp / distribute(start=startQ);
  /* Components for output */
  eval trendCycGdp = irwGdp + cycGdp;
  eval trendCycInd = irwInd + cycInd;
  eval monthlyGdp = irwGdp + cycGdp + noiseGdp;
  /* Output data set */
  output out=forGdp pdv press;
run;
```

Here are a few comments about this program:

- The first STATE statement specifies irwState as a bivariate trend that follows an integrated random walk (which is a local linear trend without the disturbance term in the level equation); irwState corresponds to $\boldsymbol{\mu}_t$. The trend components in the models for indpro and gdp are specified in the two COMP statements that follow: irwInd and irwGdp correspond to $\mu_{1,t}$ and $\mu_{2,t}$, respectively.
- Similarly, the second STATE statement and the two COMP statements that follow it define cycInd and cycGdp as the two cycle components ($\psi_{1,t}$ and $\psi_{2,t}$) in the model.
- The noise components, noiseInd and noiseGdp, which correspond to $\epsilon_{1,t}$ and $\epsilon_{2,t}$, respectively, are also defined in the same way.

- The MODEL statement for indpro corresponds to the equation $indp_t = \mu_{1,t} + \psi_{1,t} + \epsilon_{1,t}$. On the other hand, the DISTRIBUTE(START=*startQ*) option in the MODEL statement of gdp causes gdp to be modeled as an aggregated version of the unobserved monthly GDP series ($gdp_t^{\dagger}$):

$$\begin{aligned} gdp_t^{\dagger} &= \mu_{2,t} + \psi_{2,t} + \epsilon_{2,t} \\ gdp_t &= gdp_t^{\dagger} + gdp_{t-1}^{\dagger} + gdp_{t-2}^{\dagger} \end{aligned}$$

- After the model specification is complete, EVAL statements are used to define some useful linear combinations of the components that are part of the model specification—for example, monthlyGdp (which is defined as a sum of irwGdp, cycGdp, and noiseGdp) corresponds to the unobserved monthly GDP ($gdp_t^{\dagger}$). The SSM procedure outputs the estimates of these components to the data set that is specified in the OUT= option in the OUTPUT statement.

The parameter estimates in Output 33.16.2 are similar to (but not the same as) the parameter estimates reported in the case study. In particular, the estimates of the parameters of the cycle component—for example, the damping factor (Rho = 0.99228) and the period (293.32178)—are reasonably close.

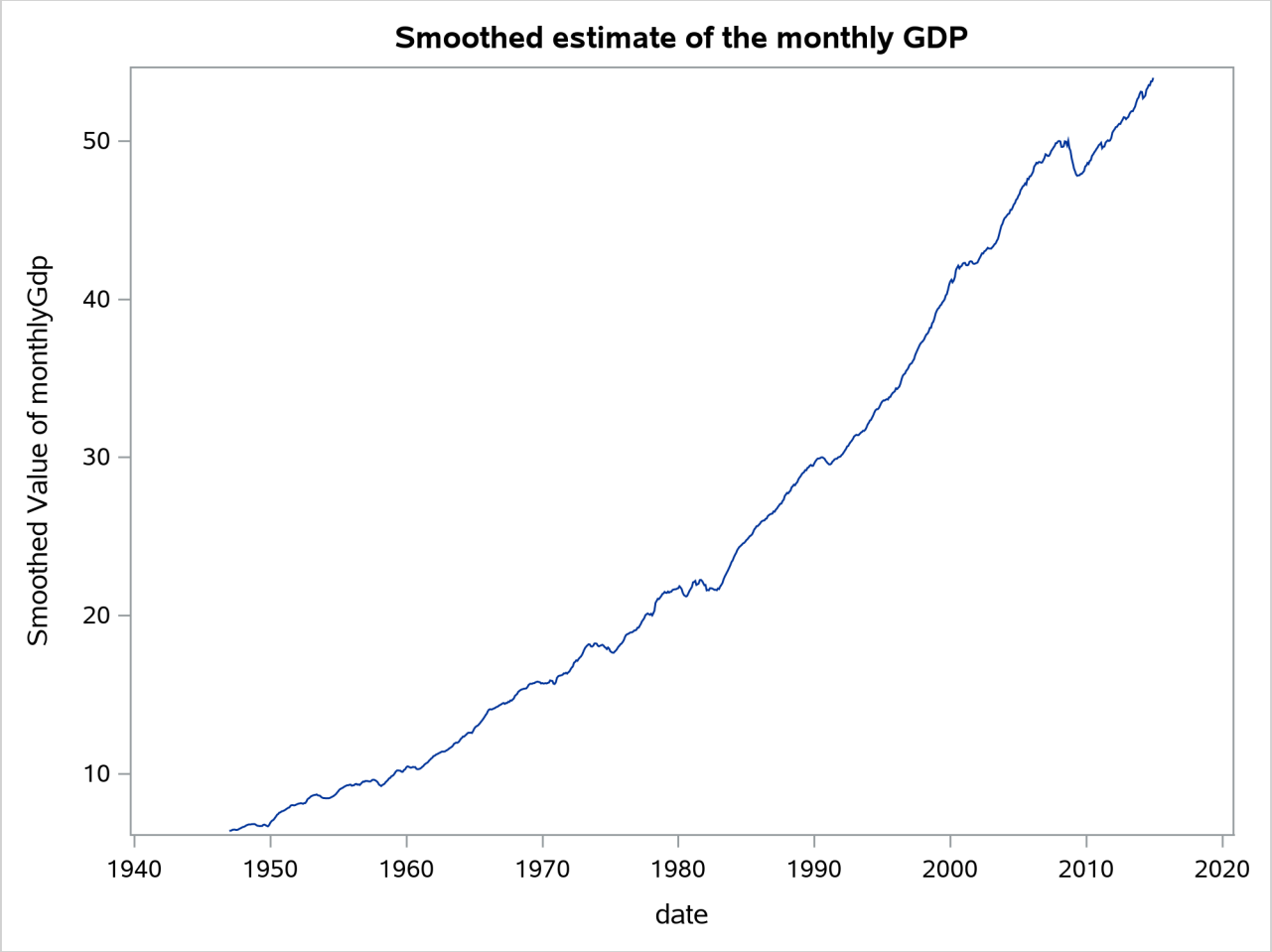
Output 33.16.2 Estimated Model Parameters

Model Parameter Estimates

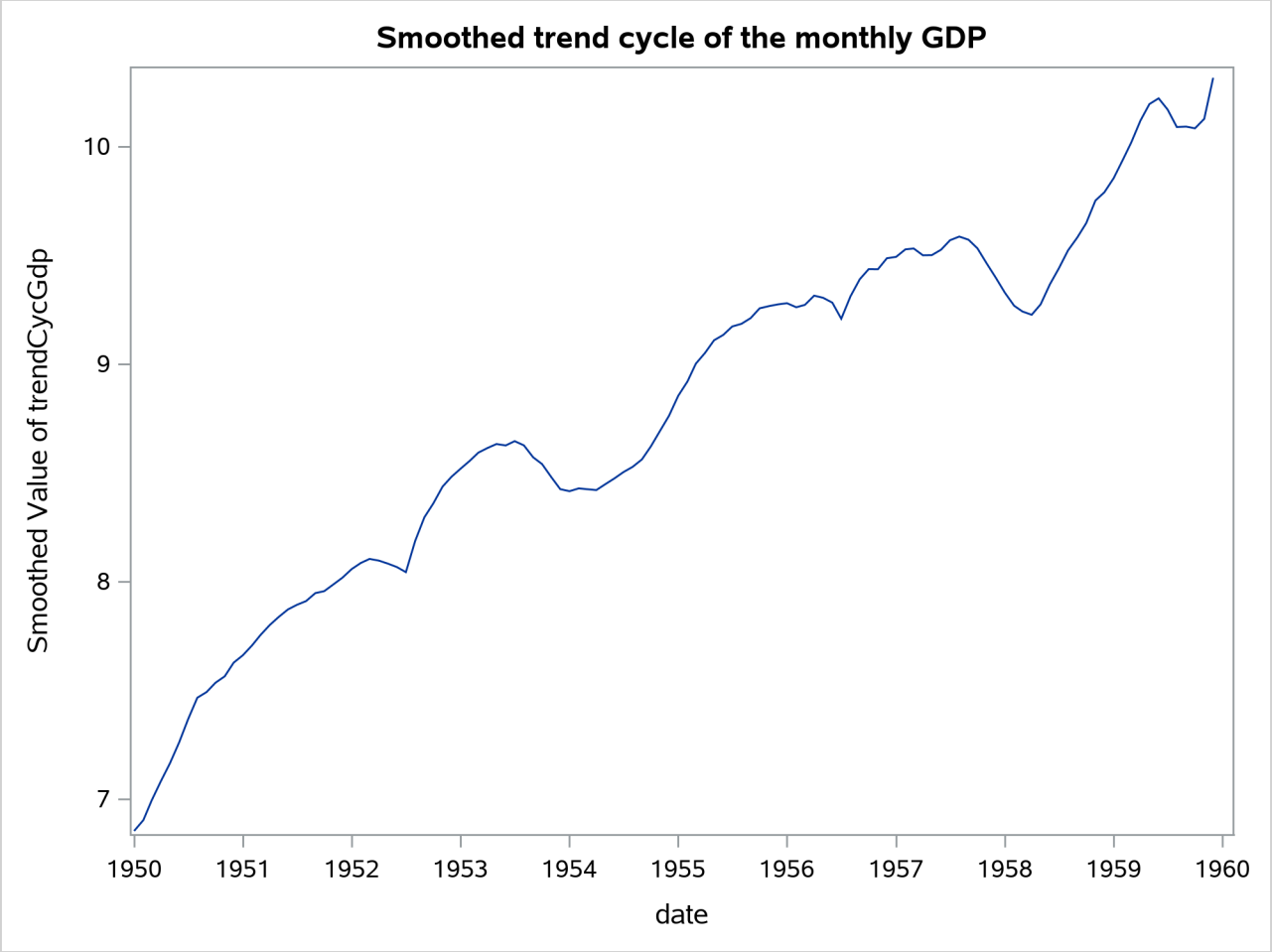
Component	Type	Parameter	Estimate	Standard Error	t Value
irwState	Slope Disturbance Covari	RootCov[1, 1]	0.10652	0.013572	7.85
irwState	Slope Disturbance Covari	RootCov[2, 1]	0.02060	0.002921	7.05
irwState	Slope Disturbance Covari	RootCov[2, 2]	0.00128	0.000507	2.52
cycle	Damping Factor	Rho	0.99228	0.004598	215.80
cycle	Cycle Period	Period	293.32178	94.269721	3.11
cycle	Disturbance Covariance	RootCov[1, 1]	0.32777	0.028788	11.39
cycle	Disturbance Covariance	RootCov[2, 1]	0.04196	0.013560	3.09
cycle	Disturbance Covariance	RootCov[2, 2]	0.06074	0.007665	7.92
noise	Disturbance Covariance	RootCov[1, 1]	0.08617	0.038876	2.22
noise	Disturbance Covariance	RootCov[2, 1]	-0.12117	0.094158	-1.29
noise	Disturbance Covariance	RootCov[2, 2]	0.03707	0.322549	0.11

Output 33.16.3 shows the plot of the estimated monthly GDP, and Output 33.16.4 shows the plot of the estimate of monthly trend-cycle estimate ($\mu_{2,t} + \psi_{2,t}$) for GDP.

Output 33.16.3 Estimate of Monthly GDP



Output 33.16.4 Smoothed Trend-Cycle Component of Monthly GDP (1950 to 1960)



Example 33.17: Temporal Aggregation: Triannual Nile River Level (Experimental)

This example illustrates how you can do model-based temporal aggregation of a response variable. The following DATA step creates a data set, Nile, by using a well-known data set that contains annual recordings of the Nile water level measured between the years 1871 and 1970. The Nile water level is clearly a stock variable, and temporal aggregation of such variables is usually meaningless. However, for illustration purposes, assume that you are interested in forecasting triannual totals of the water level.

```
data Nile;
   input level @@;
   year = intnx( 'year', '1jan1871'd, _n_-1 );
   format year year4.;
   startAggr = (mod(_n_, 3) = 1);
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
```

```
874   694   940  833   701   916   692  1020   1050  969
831   726   456  824   702   1120  1100 832    764   821
768   845   864  862   698   845   744  796    1040  759
781   865   845  944   984   897   822  1010   771   676
649   846   812  742   801   1040  860  874    848   890
744   749   838  1050  918   986   797  923    975   815
1020  906   901  1170  912   746   919  718    714   740
. . . . . . .
;
```

The Nile date set contains three variables: year indicates the observation year, level contains the yearly water level, and startAggr is a dummy variable that indicates the start of the triannual aggregation intervals. It is known that for the time span of the observations, the yearly water levels can be reasonably modeled as a sum of a random walk trend, a level shift in the year 1899, and the observation error. The following statements show you how to obtain forecasts of the triannual water level that are consistent with the model postulated for the yearly water levels:

```
proc ssm data=Nile;
  id year interval=year;
  shift1899 = ( year >= '1jan1899'd );
  trend rw(rw);
  irregular wn;
  model level = shift1899 RW wn  / aggregate(start=startAggr);
  output out=nileOut;
quit;
```

As a result of running this program, you get the usual output that is associated with fitting the specified model to the yearly water level. In addition (as explained in the section "Temporal Aggregation" on page 2455), the AGGREGATE option in the MODEL statement causes the estimation and printing of triannual aggregates of the water level. Output 33.17.1 shows the last few rows of this output. When the summands—the response values—in the aggregation are known, the aggregation can be done without error; that is, the standard error of the estimation is zero. However, when at least one summand in the aggregate is missing, the standard error of estimation is nonzero.

Output 33.17.1 Triannual Aggregate Values of the Nile Water Levels (Partial Output)

Time	Response	Start_Flag	Aggregate	StdErr	Lower	Upper
1967	919	1	919	0	919	919
1968	718	0	1637	0	1637	1637
1969	714	0	2351	0	2351	2351
1970	740	1	740	0	740	740
1971	.	0	1590	128	1338	1842
1972	.	0	2440	183	2081	2798
1973	.	1	850	128	598	1102
1974	.	0	1700	183	1341	2058
1975	.	0	2550	226	2108	2992
1976	.	1	850	128	598	1102
1977	.	0	1700	183	1341	2058

Example 33.18: Invariance of the Marginal Likelihood under Linear Rescaling of the Diffuse Effects (Experimental)

Consider the following alternate but equivalent specifications of a trend-plus-seasonal model (monthly seasonality):

$$y_t = \mu_t + \psi_t + \epsilon_t \qquad \text{Spec1}$$
$$y_t = \mu_t + m_{1,t} + \cdots + m_{11,t} + \epsilon_t \qquad \text{Spec2}$$

Here the trend (μ_t, a random walk with drift) and the irregular component (ϵ_t, white noise) are the same in both the specifications. However, the seasonal component is specified differently: in Spec1 the seasonality is modeled as a deterministic trigonometric seasonal component (ψ_t) whereas in Spec2 it is modeled using the seasonal dummies ($m_1 \cdots m_{11}$). Spec1 and Spec2 are statistically equivalent models from the perspective of the data generation process. This example uses these two specifications to demonstrate a useful invariance property of the marginal and profile likelihoods, which is described in the section "Likelihood Computation and Model-Fitting Phase" on page 2435. The airline passenger series, given as Series G in Box and Jenkins (1976), is used to illustrate the computations. The following DATA step prepares the log-transformed passenger series and the seasonal dummies that are needed for this example:

```
data seriesG;
   set sashelp.air;
   logair = log(air);
   array m{11} m1-m11;
   do i=1 to 11;
      m[i] = (month(date)=i);
   end;
run;
```

The following statements fit the two models to the log-transformed passenger series. The first PROC SSM call fits Spec1, and the second call fits Spec2.

```
proc ssm data=seriesG plots=none like=marginal;
   id date interval=month;
   trend rwDrift(ll) slopevar=0;
   irregular wn;
   state trigState(1) type=season(length=12);
   comp season = trigState[1];
   model logair = rwDrift season wn;
run;

proc ssm data=seriesG plots=none like=marginal;
   id date interval=month;
   trend rwDrift(ll) slopevar=0;
   irregular wn;
   model logair = rwDrift m1-m11 wn;
run;
```

For these two models, the parameter estimates that are based on the diffuse likelihood (REML_D) and the marginal likelihood ((REML_M) coincide because the extra term in the marginal likelihood ($-\log(|\mathbf{S}^*_{n,p_n}|)$) turns out to be independent of these parameters. Nevertheless, it is useful to use the LIKE=MARGINAL option in the PROC SSM statement so that both the likelihood computation summary and the information criteria tables display the likelihood values and the information criteria for all three likelihoods—diffuse,

marginal, and profile—at the estimated parameters. The parameter estimates for Spec1 and Spec2 are displayed in Output 33.18.1 and Output 33.18.2, respectively. As expected, the parameter estimates for the two specifications are the same because they are statistically equivalent models. The other aspects of the fit (such as model-based forecasts), which are not shown, also agree.

Output 33.18.1 Parameter Estimates For Spec1

Model Parameter Estimates

Component	Type	Parameter	Estimate	Standard Error	t Value
rwDrift	LL Trend	Level Variance	0.000766	0.000219	3.49
wn	Irregular	Variance	0.000368	0.000141	2.60

Output 33.18.2 Parameter Estimates For Spec2

Model Parameter Estimates

Component	Type	Parameter	Estimate	Standard Error	t Value
rwDrift	LL Trend	Level Variance	0.000766	0.000219	3.49
wn	Irregular	Variance	0.000368	0.000141	2.60

The fit summary tables shown in Output 33.18.3 (for Spec1) and Output 33.18.4 (for Spec2) show that the marginal and profile likelihoods (the last two lines in each table) for the two specifications also agree. However, you can see that the diffuse likelihood value for the two specifications differ (diffuse likelihood = 215.45 for Spec1 and diffuse likelihood = 226.89 for Spec2). This difference occurs because the diffuse likelihood is not invariant to the different (but equivalent) formulations of the seasonal effects. This also means that the information criteria that are based on the marginal and profile likelihoods, which are shown in Output 33.18.5 (for Spec1) and Output 33.18.6 (for Spec2), correctly conclude that the two specifications cannot be distinguished on the basis of these criteria, whereas the information criteria that are based on the diffuse likelihood erroneously suggest that Spec1 is inferior to Spec2.

Output 33.18.3 Likelihood Computation Summary For Spec1

Likelihood Computation Summary

Statistic	Value
Nonmissing Response Values Used	144
Estimated Parameters	2
Initialized Diffuse State Elements	13
Normalized Residual Sum of Squares	131
Diffuse Log Likelihood	215.4522
Profile Log Likelihood	265.63882
Marginal Log Likelihood	248.01412

Output 33.18.4 Likelihood Computation Summary For Spec2

Likelihood Computation Summary	
Statistic	**Value**
Nonmissing Response Values Used	144
Estimated Parameters	2
Initialized Diffuse State Elements	13
Normalized Residual Sum of Squares	131
Diffuse Log Likelihood	226.8959
Profile Log Likelihood	265.63882
Marginal Log Likelihood	248.01412

Output 33.18.5 Information Criteria For Spec1

Information Criteria			
Statistic	**Diffuse Likelihood Based**	**Profile Likelihood Based**	**Marginal Likelihood Based**
AIC (lower is better)	-426.9044	-501.2776	-492.0282
BIC (lower is better)	-421.1540	-456.7304	-486.2779
AICC (lower is better)	-426.8106	-497.5276	-491.9345
HQIC (lower is better)	-424.5678	-483.1762	-489.6916
CAIC (lower is better)	-419.1540	-441.7304	-484.2779

Output 33.18.6 Information Criteria For Spec2

Information Criteria			
Statistic	**Diffuse Likelihood Based**	**Profile Likelihood Based**	**Marginal Likelihood Based**
AIC (lower is better)	-449.7918	-501.2776	-492.0282
BIC (lower is better)	-444.0414	-456.7304	-486.2779
AICC (lower is better)	-449.6981	-497.5276	-491.9345
HQIC (lower is better)	-447.4552	-483.1762	-489.6916
CAIC (lower is better)	-442.0414	-441.7304	-484.2779

This example highlights the care that must be taken while doing model selection based on information criteria. It suggests that information criteria that are based on the marginal and profile likelihoods are preferred over the information criteria that are based on diffuse likelihood.

References

Akaike, H. (1974). "A New Look at the Statistical Model Identification." *IEEE Transactions on Automatic Control* AC-19:716–723.

Anderson, B. D., and Moore, J. B. (1979). *Optimal Filtering*. Englewood Cliffs, NJ: Prentice-Hall.

Ansley, C., and de Jong, P. (2012). "Inferring and Predicting Global Temperature Trends." Slide presentation at Time Series Econometrics: Conference in Honour of Andrew Harvey. http://www.oxford-man.ox.ac.uk/sites/default/files/events/Piet%20de%20Jong.pdf.

Aruoba, B. S., Diebold, F. X., and Scotti, C. (2009). "Real-Time Measurement of Business Conditions." *Journal of Business and Economic Statistics* 27:417–427.

Baltagi, B. H. (1995). *Econometric Analysis of Panel Data*. New York: John Wiley & Sons.

Baltagi, B. H., and Levin, D. (1992). "Cigarette Taxation: Raising Revenues and Reducing Consumption." *Structural Change and Economic Dynamics* 3:321–335.

Bell, W. R. (2011). "REGCMPNT—a Fortran Program for Regression Models with ARIMA Component Errors." *Journal of Statistical Software* 41:1–23. http://www.jstatsoft.org/v41/i07/.

Box, G. E. P., and Jenkins, G. M. (1976). *Time Series Analysis: Forecasting and Control*. Rev. ed. San Francisco: Holden-Day.

Bozdogan, H. (1987). "Model Selection and Akaike's Information Criterion (AIC): The General Theory and Its Analytical Extensions." *Psychometrika* 52:345–370.

Burnham, K. P., and Anderson, D. R. (1998). *Model Selection and Inference: A Practical Information-Theoretic Approach*. New York: Springer-Verlag.

Chan, W.-Y. T., and Wallis, K. F. (1978). "Multiple Time Series Modelling: Another Look at the Mink-Muskrat Interaction." *Journal of the Royal Statistical Society, Series C* 27:168–175.

De Jong, P. (1989). "Smoothing and Interpolation with the State-Space Model." *Journal of the American Statistical Association* 84:1085–1088.

De Jong, P. (1991). "The Diffuse Kalman Filter." *Annals of Statistics* 19:1073–1083.

De Jong, P., and Chu-Chun-Lin, S. (2003). "Smoothing with an Unknown Initial Condition." *Journal of Time Series Analysis* 24:141–148.

De Jong, P., and Mazzi, S. (2001). "Modeling and Smoothing Unequally Spaced Sequence Data." *Statistical Inference for Stochastic Processes* 4:53–71.

De Jong, P., and Penzer, J. (1998). "Diagnosing Shocks in Time Series." *Journal of the American Statistical Association* 93:796–806.

Diggle, P. J., Liang, K.-Y., and Zeger, S. L. (1994). *Analysis of Longitudinal Data*. Oxford: Clarendon Press.

Durbin, J., and Koopman, S. J. (2012). *Time Series Analysis by State Space Methods*. 2nd ed. Oxford: Oxford University Press.

Eubank, R. L., Huang, C., and Wang, S. (2003). "Adaptive Order Selection for Spline Smoothing." *Journal of Computational and Graphical Statistics* 12:382–397.

Francke, M. K., Koopman, S. J., and de Vos, A. F. (2010). "Likelihood Functions for State Space Models with Diffuse Initial Conditions." *Journal of Time Series Analysis* 31:407–414.

Givens, G. H., and Hoeting, J. A. (2005). *Computational Statistics*. Hoboken, NJ: John Wiley & Sons.

Hannan, E. J., and Quinn, B. G. (1979). "The Determination of the Order of an Autoregression." *Journal of the Royal Statistical Society, Series B* 41:190–195.

Harvey, A. C. (1989). *Forecasting, Structural Time Series Models, and the Kalman Filter*. Cambridge: Cambridge University Press.

Hurvich, C. M., and Tsai, C.-L. (1989). "Regression and Time Series Model Selection in Small Samples." *Biometrika* 76:297–307.

Jones, R. H. (1980). "Maximum Likelihood Fitting of ARMA Models to Time Series with Missing Observations." *Technometrics* 22:389–396.

Jones, R. H. (1993). *Longitudinal Data with Serial Correlation: A State Space Approach*. London: Chapman & Hall.

Kohn, R., Ansley, C., and Tharm, D. (1991). "The Performance of Cross-Validation and Maximum Likelihood Estimators of Spline Smoothing Parameters." *Journal of the American Statistical Association* 86:1042–1050.

Kohn, R., and Ansley, C. F. (1991). "A Signal Extraction Approach to the Estimation of Treatment and Control Curves." *Journal of the American Statistical Association* 86:1034–1041.

Koopman, S. J., Mallee, M. I. P., and van der Wel, M. (2010). "Analyzing the Term Structure of Interest Rates Using the Dynamic Nelson-Siegel Model with Time-Varying Parameters." *Journal of Business and Economic Statistics* 28:329–343.

Laird, N. (2004). *Analysis of Longitudinal and Cluster-Correlated Data.* Vol. 8 of NSF-CBMS Regional Conference Series in Probability and Statistics. Beachwood, OH: Institute of Mathematical Statistics. http://www.jstor.org/stable/4153193.

Nelson, C. R., and Siegel, A. F. (1987). "Parsimonious Modeling of Yield Curves." *Journal of Business* 60:473–489.

Pelagatti, M. M. (2015). *Time Series Modelling with Unobserved Components*. Boca Raton, FL: CRC Press.

Reinsel, G. C. (1997). *Elements of Multivariate Time Series Analysis*. 2nd ed. New York: Springer-Verlag.

Schwarz, G. (1978). "Estimating the Dimension of a Model." *Annals of Statistics* 6:461–464.

Searle, S. R., Casella, G., and McCulloch, C. E. (1992). *Variance Components*. New York: John Wiley & Sons.

Selukar, R. S. (2010). "Estimability of the Linear Effects in State Space Models with an Unknown Initial Condition." *Journal of Time Series Analysis* 31:167–168.

Selukar, R. S. (2015). "Functional Modeling of Longitudinal Data with the SSM Procedure." In *Proceedings of the SAS Global Forum 2015 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings15/SAS1580-2015.pdf.

Selukar, R. S. (2017). "Detecting and Adjusting Structural Breaks in Time Series and Panel Data Using the SSM Procedure." In *Proceedings of the SAS Global Forum 2017 Conference*. Cary, NC: SAS Institute Inc. http://support.sas.com/resources/papers/proceedings17/SAS0456-2017.pdf.

Wang, Z. (2013). "cts: An R Package for Continuous Time Autoregressive Models via Kalman Filter." *Journal of Statistical Software* 53:1–19.

Wecker, W. E., and Ansley, C. F. (1983). "The Signal Extraction Approach to Nonlinear Regression and Spline Smoothing." *Journal of the American Statistical Association* 78:81–89.

Subject Index

Syntax Index