X-13ARIMA-SEATS Reference Manual
Accessible HTML Output Version

Version 1.1

Time Series Research Staff
Center for Statistical Research and Methodology
U.S. Census Bureau
Washington, DC 20233
email: x12@census.gov
WWW: https://www.census.gov/data/software/x13as.html

March 21, 2022

This page intentionally left blank.

Contents

List of Tables

1 Introduction

Contents

The `X-13ARIMA-SEATS` seasonal adjustment program is an enhanced version of the `X-11` Variant of the Census Method II seasonal adjustment program (Shiskin, Young, and Musgrave 1967). The enhancements include a more self-explanatory and versatile user interface and a variety of new diagnostics to help the user detect and remedy any inadequacies in the seasonal and calendar effect adjustments obtained under the program options selected. The program also includes a variety of new tools to overcome adjustment problems and thereby enlarge the range of economic time series that can be adequately seasonally adjusted. Examples of the use of these tools can be found in Findley and Hood (1999). Basic information on seasonal adjustment is given in Chapter 2 of Dagum and Cholette (2006) and in Chapter 1 of Ladiray and Quenneville (2001), where the `X-11` method is thoroughly documented. See also Bell and Hillmer (1984, 1985), den Butter and Fase (1991), and Klein (1991).

The chief source of these new tools is the extensive set of time series model building facilities built into the program for fitting what we call regARIMA models. These are regression models with ARIMA (autoregressive integrated moving average) errors. More precisely, they are models in which the mean function of the time series (or its logs) is described by a linear combination of regressors, and the covariance structure of the series is that of an ARIMA process. If no regressors are used, indicating that the mean is assumed to be zero, the regARIMA model reduces to an ARIMA model. There are built-in regressors for directly estimating various flow and stock trading day effects and holiday effects. There are also regressors for modeling certain kinds of disruptions in the series, or sudden changes in level, whose effects need to be temporarily removed from the data before the `X-11` methodology can adequately estimate seasonal adjustments. To address data problems not provided for, there is the capability of incorporating user-defined regression variables into the model fitted. The regARIMA modeling module of `X-13ARIMA-SEATS` was adapted from the regARIMA program developed by the Time Series Staff of U.S. Census Bureau's Statistical Research Division.

Whether or not special problems requiring the use of regressors are present in the series to be adjusted, a fundamentally important use of regARIMA models is to extend the series with forecasts (and backcasts) in order to improve the seasonal adjustments of the most recent (and the earliest) data. Doing this mitigates problems inherent in the trend estimation and asymmetric seasonal averaging processes of the type used by the `X-11` method near the ends of the series. The provision of this extension was the most important technical improvement offered by Statistics Canada's widely used `X-11` program. Its benefits, both theoretical and empirical, have been documented in many publications, including Geweke (1978), Dagum (1988), and Bobbitt and Otto (1990), and the articles referenced in these papers.

`X-13ARIMA-SEATS` is available as an executable program for PC microcomputers (386 or higher with a math coprocessor) running DOS (version 3.0 or higher), Sun 4 UNIX workstations, and VAX/VMS computers. Fortran source code is available for users to create executable programs on other computer systems. When it is released, the `X-13ARIMA-SEATS` program will be in the public domain, and may be copied or transferred. Computer files containing the current test version of the program (executables for various machines and source

code), this documentation, and examples, have been put on the Internet at `https://www.census.gov/data/software/x13as.html`. Limited program support is available via regular mail and email (the preferred mode of communication) at the addresses given on the title page. If problems are encountered running a particular input file, providing the input, data and resulting output files will facilitate our identification of the problem.

There are now two seasonal adjustment modules contained in the program. One uses the `X-11` seasonal adjustment method detailed in Shiskin, Young, and Musgrave (1967). The program has all the seasonal adjustment capabilities of the `X-11` and `X-11-ARIMA` programs. The same seasonal and trend moving averages are available, and the program still offers the `X-11` calendar and holiday adjustment routines.

The `X-11` seasonal adjustment module has also been enhanced by the addition of several new options, including

(a) the sliding spans diagnostic procedures, illustrated in Findley, Monsell, Shulman, and Pugh (1990)
(b) the ability to produce the revisions history of a given seasonal adjustment
(c) a new Henderson trend filter routine which allows the user to choose any odd number for the length of the Henderson filter
(d) new options for seasonal filters
(e) several new outlier detection options for the irregular component of the seasonal adjustment
(f) a table of the trading day factors by type of day
(g) a pseudo-additive seasonal adjustment mode.

The second seasonal adjustment module uses the ARIMA model-based seasonal adjustment procedure from the SEATS seasonal adjustment program developed by Victor Gómez and Agustin Maravall at the Bank of Spain. All the capabilities of SEATS are included in this version of `X-13ARIMA-SEATS`, which can generate stability and spectral diagnostics for SEATS seasonal adjustments in the same way as `X-11` seasonal adjustments.

For more details on the SEATS seasonal adjustment method, see Maravall (1995), Gómez and Maravall (1996), Gómez and Maravall (2001a), and Gómez and Maravall (2001b). Findley, Lytras, and Maravall (2016) provides a tutorial on the model-based seasonal adjustment method of the SEATS and its implementation in `X-13ARIMA-SEATS`.

The modeling module of `X-13ARIMA-SEATS` is designed for regARIMA model building with seasonal economic time series. To this end, several categories of predefined regression variables are available in `X-13ARIMA-SEATS`, including trend constants or overall means, fixed seasonal effects, trading-day effects, holiday effects, pulse effects (additive outliers), level shifts, temporary change outliers, and ramp effects. User-defined regression variables can also be easily read in and included in models. The program is designed around specific capabilities needed for regARIMA modeling, and is not intended as a general purpose statistical package. In particular, `X-13-ARIMA-SEATS` should be used in conjunction with other (graphics) software capable of producing high resolution plots of time series.

Observations (data) from a time series to be modeled and/or seasonally adjusted using `X-13ARIMA-SEATS` should be quantitative, as opposed to binary or categorical. Observations must be equally spaced in time, and missing values are not allowed. `X-13ARIMA-SEATS` handles only univariate time series models, i.e., it does not estimate relationships between different time series.

`X-13ARIMA-SEATS` uses the standard $(p\ d\ q)(P\ D\ Q)_s$ notation for seasonal ARIMA models. The $(p\ d\ q)$ refers to the orders of the *nonseasonal* autoregressive (AR), differencing, and moving average (MA) operators,

respectively. The $(P\ D\ Q)_s$ refers to the *seasonal* autoregressive, differencing, and moving average orders. The s subscript denotes the seasonal period, e.g., $s = 12$ for monthly data. Great flexibility is allowed in the specification of ARIMA structures: any number of AR, MA, and differencing operators may be used, missing lags are allowed in AR and MA operators, and AR and MA parameters can be fixed at user-specified values.

For the user who wishes to fit customized time series models, `X-13ARIMA-SEATS` provides capabilities for the three modeling stages of *identification*, *estimation*, and *diagnostic checking*. The specification of a regARIMA model requires specification both of the regression variables to be included in the model and also the type of ARIMA model for the regression errors (i.e., the orders $(p\ d\ q)(P\ D\ Q)_s$). Specification of the regression variables depends on user knowledge about the series being modeled. *Identification* of the ARIMA model for the regression errors follows well-established procedures based on examination of various sample autocorrelation and partial autocorrelation functions produced by the `X-13ARIMA-SEATS` program. Once a regARIMA model has been specified, `X-13ARIMA-SEATS` will *estimate* its parameters by maximum likelihood using an iterative generalized least squares (IGLS) algorithm. *Diagnostic checking* involves examination of residuals from the fitted model for signs of model inadequacy. `X-13ARIMA-SEATS` produces several standard residual diagnostics for model checking, as well as providing sophisticated methods for detecting additive outliers and level shifts. Finally, `X-13ARIMA-SEATS` can produce point forecasts, forecast standard errors, and prediction intervals from the fitted regARIMA model.

In addition to these modeling features, `X-13ARIMA-SEATS` has an automatic model selection procedure based largely on the automatic model selection procedure of TRAMO (Gómez and Maravall 1996, documented in Gómez and Maravall 2001a). There are also options that use AICC to determine if user-specified regression variables (such as trading day or Easter regressors) should be included into a particular series. Also, histories can be generated for likelihood statistics (such as AICC, a version of Akaike's AIC that adjusts for the length of the series being modeled) and forecasts to facilitate comparisons between competing models.

The next six chapters detail capabilities of the `X-13ARIMA-SEATS` program.

- **Chapter 2** provides an overview of running `X-13ARIMA-SEATS` and explains program limits that users can change.
- **Chapter 3** provides a general description of the required input file (specification file), and also discusses specification file syntax and related issues.
- **Chapter 4** discusses the general regARIMA model fit by the `X-13ARIMA-SEATS` program, summarizes the technical steps involved in regARIMA modeling and forecasting, and relates these steps to capabilities of the program.
- **Chapter 5** discusses some key points related to model estimation and inference that all users of the modeling features should be aware of, including some estimation problems that may arise and ways to address them.
- **Chapter 6** discusses some details of key seasonal adjustment diagnostics: spectrums, sliding spans, and revisions history.
- **Chapter 7** gives detailed documentation for each specification statement that can appear in the specification file. These statements function as commands that control the flow of `X-13ARIMA-SEATS`' execution and select among the various program options.

The focus in Chapters 2 through 6 is on giving an overview of the use and capabilities of the `X-13ARIMA-SEATS` program. In contrast, Chapter 7 is intended as the primary reference to be used when constructing specification files for running the `X-13ARIMA-SEATS` program.

1.1 Acknowledgements

We are indebted to Statistics Canada, particularly to Estela Dagum, providing us with the source code from `X-11-ARIMA` (Dagum 1980, Dagum 1988) to use as the starting point for the seasonal adjustment routines of `X-13ARIMA-SEATS` and giving us helpful advice.

We are grateful to Hirotugu Akaike and Makio Ishiguro of the Institute of Statistical Mathematics for permission to incorporate into `X-13ARIMA-SEATS` the source code of the autoregressive spectrum diagnostics of BAYSEA (Akaike and Ishiguro 1980).

`X-13ARIMA-SEATS` would not have been possible without the support of Agustín Maravall while at the Bank of Spain and Gianluca Caporello, who provided us with SEATS source code (including updates) and were generous with their advice and expertise.

Further, we are indebted to Victor Gómez for providing us with the Fortran code of TRAMO (Gómez and Maravall 2001a) to enable us to implement an automatic modeling procedure very similar to TRAMO's in `X-13ARIMA-SEATS`, and to Agustín Maravall, Gianluca Caporello, and contractors at the Bank of Spain for updates to the TRAMO and SEATS source code and advice.

1.2 License information and disclaimer

The construction, validity, performance, and effect of this Agreement for all purposes will be governed by Federal law of the United States.

User agrees to make a good faith effort to use the Software in a way that does not cause damage, harm, or embarrassment to the United States/Commerce. The United States/Commerce expressly reserve all rights and remedies.

2 Running X-13ARIMA-SEATS

Contents

Tables

Procedures for installing X-13ARIMA-SEATS are machine-specific; information about this is provided with the program, and is also available on the Internet at https://www.census.gov/data/software/x13as.html. Having installed the program on a microcomputer running a DOS operating system, a generic statement to run X-13ARIMA-SEATS is

path\x13asHTML *path*\filename

In this statement *path*\filename.spc is the main X-13ARIMA-SEATS input (specification) file. The program created a file named *path*\filename.html as an output file. The *path* to X-13ARIMA-SEATS is necessary if the file containing the X-13ARIMA-SEATS program is not in the current directory; this also holds for the *path* to the input file filename.spc.

The program creates an index to the program output file that appears on the right side of the output. This section contains links to the individual tables of the output file, and also contains links to the error file (see Section 2.3) and the log file (see Section 2.6) generated for this run.

Note that only the filename is specified, not the extension; the program will use the filename provided at runtime to form the filenames for all files generated by the program. For an X-13ARIMA-SEATS run using the spec file *filename*.spc, the output will be stored in the file *filename*.html, the error messages will be stored in the file *filename*_err.html, etc. Thus, if the spec file xuu1.spc is in a PC's current directory, typing

```
x13asHTML   xuu1
```

and pressing the <return> (or <enter> key) will cause the program to run and create files xuu1.html and xuu1_err.html in the current directory.

Program input and output are both discussed briefly below, and more extensively in the documentation that follows. To run the program under a UNIX (or Linux) operating system, substitute (forward) slashes for the backslashes in the generic statements above. To run X-13ARIMA-SEATS under other operating systems, specify paths, etc., using the syntax appropriate for the system. For the DOS and UNIX/Linux operating systems, a quick reference document is also available, giving more detailed instructions on the syntax for running X-13-ARIMA-SEATS in these operating systems.

2.1 Input

To apply X-13ARIMA-SEATS to any particular time series, a main input file, called a *specification file*, must be created. This ASCII (or "text") file contains a set of specifications or *specs* that X-13ARIMA-SEATS reads to obtain the information it needs about the time series data, the time series model to be used, the analysis to be performed, and the output desired. X-13ARIMA-SEATS assumes that the specification file has the extension .spc. Thus *path*\filename is sufficient in the above statements. If the specification file name has an extension other than .spc or no extension, use the entire file name. The only input files other than the spec file that X-13ARIMA-SEATS may need are optional files containing data for the time series being modeled, data for any user-defined regression variables, values for any user-defined prior-adjustment factors, and model types to try with the automatic model selection procedure from the **pickmdl** spec. The names of these files (including paths) are provided to X-13ARIMA-SEATS by listing them in appropriate specs in the spec file. The use of such additional input files is optional because the user can alternatively include the data values required in appropriate places in these specs, and a default set of models for the automatic modeling procedure is available. Chapter 7 explains how to write spec files.

2.2 Output

The usual output is written to the file *path*\filename.html. Individual specs control their contribution to this output using optional **print** arguments (discussed in Section 3.2). The **save** argument is used to create certain other output files for further analysis (for example, to save a time series of residuals for plotting using a graphics program). Cautionary note: When **save** is used, the program constructs the name of the file to which the specified output is written using naming conventions discussed in Section 3.2. If a file with this name already exists, it will be overwritten by X-13ARIMA-SEATS and the contents lost. Users should thus take suitable precautions when saving output. See Section 3.2 for more information.

2.3 Input errors

Input errors are reported as they are discovered by the program, which then prints appropriate error messages. These error messages are also stored in a file named *path*\filename_err.html. When the program can localize

the error, the line in the spec file containing the error will be printed out with a caret (^) positioned under the error. If the program cannot localize the error, then only the error message will be printed. If the error is fatal, then **ERROR:** will be displayed before the error message, sometimes with suggestions about what to change. For nonfatal errors, **WARNING:** will be printed before the message. **WARNING** messages are also used sometimes to call attention to a situation in which no error has been committed, but some caution is appropriate.

X-13ARIMA-SEATS first reads the whole spec file, reporting all input errors it finds. This way the user can try to correct more than one input error per run. Frequently, however, the only informative messages are those for the first one or two errors. These errors may result in other errors, especially if input errors occur in the **series** spec. The program will stop if any fatal errors are detected. Warnings will not stop the program, but should alert users to check both the input and output carefully to verify that the desired results are produced.

2.4 Specifying an alternate output filename

As was noted before, for an X-13ARIMA-SEATS run using the spec file *filename*.spc, the output will be stored in the file *filename*.html, the error messages will be stored in the file *filename*_err.html, etc. For the purpose of examining the effects of different adjustment and modeling options on a given series, it is sometimes desirable to use a different filename for the output than was used for the input. The general form for specifying an alternate filename for the output files is

path\x13asHTML *path*\filename *path*\outname (2.1)

This X-13ARIMA-SEATS run still uses the spec file *filename*.spc, but the output will be stored in the file *outname*.html, the error messages will be stored in the file *outname*_err.html, etc. All output files generated by this run will be stored using the path and filename given by the user, not the path and filename of the input specification file.

2.5 Running X-13ARIMA-SEATS on more than one series

In a production situation, it is essential to run more than one series in a given X-13ARIMA-SEATS run. X-13-ARIMA-SEATS allows for running multiple series in two modes:

(a) **multi-spec mode**, where there are input specification files for every series specified;
(b) **single spec mode**, where every series will be run with the options from a single input specification file.

Before X-13ARIMA-SEATS can be run in either mode, a *metafile* must be created. This is an ASCII file which contains the names of the files to be processed. Two types of metafiles are used by the X-13ARIMA-SEATS software: input metafiles (for multi-spec mode) and data metafiles (for single spec mode).

If an error occurs in one of the spec files in a metafile run, the program will print the appropriate error messages. Execution will stop for that series and the program will continue processing the remaining spec files. A listing of all the input files with errors is given in the X-13ARIMA-SEATS log file, described in Section 2.7.

2.5.1 Running X-13ARIMA-SEATS in multi-spec mode

Before X-13ARIMA-SEATS can be run in multi-spec mode, an *input metafile* must first be created. This is an ASCII file that contains the names of the files to be processed by X-13ARIMA-SEATS in sequence. An input metafile can have up to two entries per line: the filename (and path information, if necessary) of the input specification file for a given series, and an optional output filename for the output of that series. If an output filename is not given by the user, then the path and filename of the input specification file will be used to generate the output files. The input specification files are processed in the order in which they appear in the input metafile.

For example, to run the spec files xuu1.spc, xuu2.spc, and xuu3.spc, the input metafile should contain the following:

```
xuu1
xuu2
xuu3
```

This assumes that all these spec files are in the current directory. To run these files if they are stored in the c:\export\specs DOS directory, the metafile should read

```
c:\export\specs\xuu1
c:\export\specs\xuu2
c:\export\specs\xuu3
```

To run X-13ARIMA-SEATS with a input metafile, use the following syntax:

```
x13asHTML -m metafile
```

where metafile.mta is the metafile and -m is a flag informing X-13ARIMA-SEATS of the presence of a metafile.

For example, if the metafile defined above is stored in exports.mta, type

```
x13asHTML -m exports
```

and press the <return> key to run the corresponding spec files.

Note that when the name of the input metafile was given in the example above, only the filename was specified, not the extension; .mta is the required extension for the input metafile. Path information should be included with the input metafile name, if necessary.

The filenames used by X-13ARIMA-SEATS to generate output files are taken from the spec files listed in the metafile, not from the metafile itself. The example given above would generate output files named xuu1.html, xuu2.html, and xuu3.html corresponding to the individual spec files given in the metafile exports.mta, not a comprehensive output file named exports.html. To specify alternate output filenames for the example above, simply add the desired output filenames to each line of the input metafile, e.g.,

```
c:\export\specs\xuu1  c:\export\output\xuu1
c:\export\specs\xuu2  c:\export\output\xuu2
c:\export\specs\xuu3  c:\export\output\xuu3
```

In addition to the output files from the individual spec files, each metafile run creates a separate index file for all the output generated by the metafile. Links to the output, index, error, and log files are organized into a metafile index file. For input metafiles, the base filename of the metafile is used to generate the name of the metafile index; for the example given above, the metafile index file is stored in `exports_mta.html`.

2.5.2 Running X-13ARIMA-SEATS in single spec mode

To run `X-13ARIMA-SEATS` on many series using the same specification commands for each series, it is necessary to create a *data metafile*. A data metafile can have up to two entries per line: the complete filename (and path information, if necessary) of the data file for a given series and an optional output filename for the output of that series. If an output filename is not given by the user, then the path and filename of the data file will be used to generate the output files. **Note:** In a data metafile, no extension is assumed for the individual data files. The extensions must be specified, along with the path and filename, if the data files are not in the current directory.

The data files are processed in the order in which they appear in the data metafile. The options used to process each data file are provided by a single input specification file identified at runtime. This means that *all* the data files specified in the data metafile must be in the same format. Also, certain formats supported by `X-13ARIMA-SEATS` should be avoided; see the description of the **series** spec in Section 7.15 for more details.

For example, to process the data files`xuu1.dat`, `xuu2.dat`, and `xuu3.dat`, the data metafile should contain the following:

```
xuu1.dat
xuu2.dat
xuu3.dat
```

This assumes that all these data files are in the current directory. To run these files if they are stored in the `c:\export\data` DOS directory, the metafile should read

```
c:\export\data\xuu1.dat
c:\export\data\xuu2.dat
c:\export\data\xuu3.dat
```

To run `X-13ARIMA-SEATS` with a data metafile, use the following syntax:

```
x13asHTML specfile -d metafile,
```

where `metafile.dta` is the data metafile, `-d` is a flag that informs `X-13ARIMA-SEATS` of the presence of a data metafile, and `specfile.spc` is the single input specification file used for each of the series listed in the data metafile.

For example, if the data metafile with three series used for illustration above is named `exports.dta`, type

```
x13asHTML default -d exports
```

and press the <return> key to process the corresponding data files using the default.spc input specification file.

Note that when the name of the data metafile was given in the example above, only the filename was specified, not the extension; .dta is the required extension for the input metafile. Path information should be included with the data metafile name, if necessary.

The filenames used by X-13ARIMA-SEATS to generate output files are taken from the data files listed in the metafile, not by the metafile itself. The example given above would generate output files named xuu1.html, xuu2.html, and xuu3.html, corresponding to the individual data files given in the metafile exports.dta, not a comprehensive output file named exports.html. To specify alternate output filenames for the example above, simply add the desired output filenames to each line of the data metafile, e.g.,

```
c:\export\data\xuu1.dat  c:\export\output\xuu1
c:\export\data\xuu2.dat  c:\export\output\xuu2
c:\export\data\xuu3.dat  c:\export\output\xuu3
```

As with input metafiles, each data metafile run creates a separate index file for all the output generated by the data metafile. Links to the output, index, error, and log files are organized into a data metafile index file. The base filename of the data metafile is used to generate the name of the metafile index; for the example given above, the data metafile index file is stored in exports_dta.html.

2.5.3 Special case: file names containing spaces

In many current operating systems, it is permissible to have blank spaces in file names or paths – for example, c:\My Spec Files\test.spc. When specifying such a file in an input or data metafile, the user must enclose the entire filename with quotation marks ("). Otherwise, the program will assume that the first entry in the metafile is only the text up to the first space.

For example, if the spec files used in the second example in Section 2.5.1 were stored in a DOS directory named c:\export specs, then the input metafile should read:

```
"c:\export specs\xuu1"
"c:\export specs\xuu2"
"c:\export specs\xuu3"
```

Running X-13ARIMA-SEATS on the input metafile given above would generate output files named xuu1.html, xuu2.html, and xuu3.html in the c:\export specs directory.

This convention applies to data metafiles and alternate output filenames provided in metafiles as well. The following data metafile would read data files from the directory c:\export data and store the output files into the directory c:\export output:

```
"c:\export data\xuu1.dat"  "c:\export output\xuu1 a"
"c:\export data\xuu2.dat"  "c:\export output\xuu2 a"
"c:\export data\xuu3.dat"  "c:\export output\xuu3 a"
```

Running X-13ARIMA-SEATS on the data metafile given above would generate output files named xuu1 a.html, xuu2 a.html, and xuu3 a.html in the c:\export output directory.[1]

Be careful that the opening and closing quotation marks fully contain the filenames with no extra spaces, and that there are matching opening and closing quotation marks for each file. Also, there is one exception: you cannot have a space before the file extension. For example, the input specification file My Little Spec File .spc cannot be run by X-13ARIMA-SEATS either by itself or in a metafile. You will need to remove the space before the "." to run the file.

2.6 Log files

Every time X-13ARIMA-SEATS is run, a **log file** is produced where a summary of modeling and seasonal adjustment diagnostics can be stored for every series or spec file processed. When X-13ARIMA-SEATS is run in multi-spec or single spec model, as described in the previous section, the log file is stored with the same name and directory as the metafile (for multi-spec mode) or data metafile (for single spec mode), appending _log.html to the base metafile name. For example,

```
x13asHTML -m exports
```

runs each of the spec files stored in exports.mta and stores user-selected diagnostics into the log file exports_log.html.

If only one series is processed, _log.html is appended to the output directory and filename to form the name of the log file.

Users can specify which diagnostics are stored in the log file by using the **savelog** argument found in the **automdl**, **check**, **composite**, **estimate**, **history**, **pickmdl**, **regression**, **seats**, **spectrum**, **slidingspans**, **transform**, **x11**, and **x11regression** specs. The descriptions of the individual specs in Chapter 7 give more details on which diagnostics can be stored in the log file.

As mentioned in the previous section, if an error occurs in one of the spec files in a metafile run, a listing of all the input files with errors is given in the log file.

2.7 Flags

In the previous section, the flags **-m** and **-d** were required in the command line to obtain the desired run. There are several other input and output options that are specified on the command line. The general syntax for the command line can be given as

[1] Note that if you are trying to access the HTML output files in the address bar of a browser, you should replace the spaces with "%20".

path\x13asHTML *arg1* *arg2* ... *argN*

where the arguments given after x13asHTML can be either flags or filenames, depending on the situation.

Table 2.1 gives a summary of the flags available in X-13ARIMA-SEATS; the remainder of this section will describe what each flag means in more detail. These flags can be specified in any order on the command line. (Some must be followed by appropriate filenames).

flags	*description of flags*
-c	Suppress the indirect adjustment in a composite adjustment
-d *filename*	Filename (without extension) for data metafile
-g *dirname*	Directory where graphics metafile and related files for input to external graphics programs are stored
-i *filename*	Filename (without extension) for input specification file
-m *filename*	Filename (without extension) for input metafile
-n	(No tables) Only tables specifically requested in the input specification file will be printed out
-o *filename*	Filename (without extension) used for all output files generated during this run of the program
-q	Run X-13ARIMA-SEATS in quiet-mode (warning messages not sent to the console)
-r	Produce reduced X-13ARIMA-SEATS output (as in GiveWin version of X-13ARIMA-SEATS)
-s	Store seasonal adjustment and regARIMA model diagnostics in a file
-t	Store timing information in the diagnostics file (if **-s** or **-g** not specified, will generate diagnostic file)
-v	Only check input specification file(s) for errors; no other processing
-x	Produces XHTML accessible output

Table 2.1: X-13ARIMA-SEATS **Program Flags**

The **-m** and **-d** flags were described in the previous section. Note that one cannot specify both of these flags in the same run.

The **-i** flag indicates that the next argument is the path and filename of the input specification file. This flag does not need to be specified as long as the input specification file is the first argument; therefore, x13asHTML test and x13asHTML -i test are equivalent. The **-i** and **-m** flags cannot be specified in the same run.

Similar to **-i**, the **-o** flag indicates that the next argument is the path and filename for the output. The output extensions described earlier (.html and _err.html) as well as extensions associated with the **save** command will be appended to this filename. This flag also does not need to be specified as long as the input specification file is the first argument and the output filename is the second argument (as in Equation 2.1). So any of the following commands are equivalent:

```
x13asHTML test test2
x13asHTML -i test -o test2
x13asHTML -o test2 -i test
```

However, `x13asHTML -i test test2` will generate an error, since the first argument is the flag **-i**, not the spec file. The **-o** flag cannot be specified in the same run as the **-m** or **-d** flags. The **-o** and **-m** flags cannot be specified in the same run.

For operating systems that allow blank spaces in file names, the convention for specifying a file name as a flag is similar to that specified in Section 2.5.3. All filenames with at least one space in the filename or path should be enclosed in quotation marks (").

So any of the following commands should execute correctly:

```
x13asHTML "c:\My Spec Files\test" "c:\My Output\test2"
x13asHTML -i "c:\My Spec Files\test" -o "c:\My Output\test2"
x13asHTML -o "c:\My Output\test2" -i "c:\My Spec Files\test"
x13asHTML -m "c:\My Spec Files\alltest"
x13asHTML "c:\My Spec Files\testsrs" -d "c:\My Data Files\testsrs"
```

The **-s** flag specifies that certain seasonal adjustment and regARIMA modeling diagnostics that appear in the main output be saved in file(s) separate from the main output. These include tables in the main output file that are not tables of time series. Such tables cannot be stored in the format used for individual time series tables. When the **-s** flag is used, `X-13ARIMA-SEATS` automatically stores the most important of these diagnostics in a separate file that can be used to generate diagnostic summaries. This file (called the *diagnostics summary file*) will have the same path and filename as the main output, with the extension `.udg`. So for

```
x13asHTML test -s
```

the diagnostics summary file will be stored in `test.udg`, and for

```
x13asHTML test -s -o testout
```

the diagnostics summary file will be stored in `testout.udg`.

The diagnostics summary file is an ASCII database file. Within the diagnostic file, each diagnostic has a unique key to access it's value. The key is separated from the diagnostic value by a colon (':'), followed by white space. So in the entry below:

```
freq: 12
```

The key for this entry would be `freq`, and the value for the key would be `12`. Each record in the file provides a value for a unique key found at the beginning of the line.

User-defined metadata can be stored in the diagnostics summary file (for more details, see the description of the **metadata** spec in Section 7.10).

A program is available via the Internet at `https://www.census.gov/data/software/x13as.html` that reads the seasonal adjustment diagnostics file and produces a summary of the seasonal adjustment diagnostics. This program is written in the Icon programming language (see Griswold and Griswold 1997).

The **-g** flag indicates that the next argument is the complete path name of a directory into which output will be stored that is intended as input for a separate graphics program. This output consists of the following files:

(1) files of diagnostic data to be graphed, which are produced by the options specified in the `.spc` file;

(2) a *graphics metafile* containing the names of these files;

(3) a *diagnostics summary file* containing information about the time series being processed, about the regARIMA model fit to the series (if any), and about the seasonal adjustment requested (if any).

The graphics metafile carries the extension `.gmt` and the diagnostics summary file carries the extension `.udg`; these files carry the filename used for the main program output. For example, if a user enters

```
x13asHTML test -g c:\sagraph
```

the graphics metafile will be stored in `c:\sagraph\test.gmt` and the diagnostics summary file will be stored in `c:\sagraph\test.udg`. For

```
x13asHTML test -g c:\sagraph -o testout
```

the graphics metafile will be stored in `c:\sagraph\testout.gmt` and the diagnostics summary file will be stored in `c:\sagraph\testout.udg`. In both cases, related files needed to generate seasonal adjustment graphics will be also be stored in the `c:\sagraph` subdirectory. (NOTE: The directory entered after the **-g** flag must already have been created and should be different from the directory used for the output files; it can be a subdirectory of the latter.)

Two versions of a program named `X-13-Graph` (see Hood 2002a, Hood 2002c, and Lytras 2020b) that uses SAS/GRAPH (see SAS Institute Inc. 1990) to produce graphs from the graphics mode output are distributed with `X-13ARIMA-SEATS` on the Census Bureau website. In addition, `X-13-Graph Java` is an implementation of the `X-13-Graph` software in Java (Lytras 2020a). `X-13-Graph Java` generates graphics from the output of `X-13ARIMA-SEATS` and does not require SAS. It has almost all of the functionality of `X-13-Graph`. For examples of the use of `X-13-Graph` see Findley and Hood (1999). For a list of the files stored by `X-13ARIMA-SEATS` in graphics mode, along with the codes used in the graphics metafile to denote these files, see Appendix A.

If both the **-g** and **-s** options are used in the same `X-13ARIMA-SEATS` run, the complete version of the seasonal adjustment diagnostics file will be stored in the directory specified by the **-g** option (and not in the directory of the main output file). If a model diagnostics file is also generated, that file will be stored in the graphics directory as well. A warning message is written to the screen and to the log file telling the user that the seasonal adjustment diagnostics file (and the model diagnostics file, if it is produced) is in the graphics directory.

The **-n**, **-w**, **-p**, and **-r** flags all affect the format of program output. The **-n** option allows the user to restrict the number of tables appearing in the main output file. The `X-13ARIMA-SEATS` program produces a large number of tables in the main output file. While `X-13ARIMA-SEATS` is flexible in allowing users to determine

which tables are to be printed out, it is sometimes convenient to restrict the output to only a few tables. To facilitate this, the **-n** flag specifies that, as the default, no tables will be written to the main output file. Thus, only those tables explicitly specified by the user in the spec file are written.

The **-r** flag specifies that output tables and headers will be written in a format that will reduce the amount of output printed out to the main output file. The tables printed out are consolidated, and some blank lines in the printout are suppressed. This output option was first utilized in the version of X-13ARIMA-SEATS developed for use with the GiveWin econometrics package (see Doornik and Hendry 2001).

The **-x** flag specifies that the accessible program output should be generated in XHTML Version 1.0 Transitional rather than HTML Version 4.0. XHTML (Extensible HyperText Markup Language) is a family of XML markup languages that mirror or extend versions of the widely used Hypertext Markup Language (HTML). For a general discussion of XHTML, visit Wikipedia (2013).

The **-q** flag specifies that X-13ARIMA-SEATS will be run in "quiet mode." Warning messages that are normally printed to the console are suppressed, although error messages shall still be printed to the console. All warning messages not printed to the screen will be stored in the error file (see Section 2.3).

The **-c** flag is used to suppress the indirect adjustment in a composite adjustment. The spec files for each component will be processed as usual, and a direct adjustment will be calculated if the composite spec file contains either the **x11** or **seats** spec.

Finally, the **-v** flag specifies that X-13ARIMA-SEATS will be run in an input verification mode to enable the user to see if there are errors in one or more input spec files. This allows the user to check the program options for errors without doing complete X-13ARIMA-SEATS runs for all the series. The **-v** flag cannot be used with the **-s**, **-c**, or **-n** flags.

2.8 Program limits

The X-13ARIMA-SEATS Fortran source code contains limits on the maximum length of series, maximum number of regression variables in a model, etc. These limits are set at values believed to be sufficiently large for the great majority of applications, without being so large as to cause memory problems or to significantly lengthen program execution times.

Table 2.2 details those parameter variables in the model.prm, srslen.prm, and stdio.i files corresponding to X-13ARIMA-SEATS program limits that are subject to user modification.

The limits may be modified if required, but the Fortran source code of the program must then be recompiled and relinked to put the new limits into effect. The limits potentially requiring modification for this purpose occur in parameter statements in the files model.prm, srslen.prm, and stdio.i. We suggest keeping a backup copy of the original files, in case problems arise from an attempt to modify program limits.

parameter variable	*value (limit)*	*description of parameter*
`pobs`	780	maximum length of the series on input. The number, `pobs` + 2 × `pfcst` (see below), is the maximum length of input series of user-defined regression variables and user-defined prior adjustment factors — the additional 2 × `pfcst` values are allowed to accommodate values of regression variables or adjustment factors in a possible forecast (or backcast) period
`pyrs`	85	maximum number of years in the forecast and backcast extended series
`psp`	12	maximum seasonal period, i.e., observations more frequent than `psp` times per year are not allowed
`pfcst`	120	maximum number of forecasts
`pb`	80	maximum number of regression variables in a model (including predefined and user-defined regression variables specified, plus any regression variables generated by automatic outlier detection or an AIC test)
`pureg`	52	maximum number of user-defined regression variables
`porder`	36	maximum lag corresponding to any AR or MA parameter
`pdflg`	3	maximum number of differences in any ARIMA factor (nonseasonal or seasonal)
`psrs`	10000	maximum number of files that can be processed by a metafile
`pfilcr`	512	maximum number of characters in a file name including path (also maximum size for any argument read by the program at runtime)

Table 2.2: `X-13ARIMA-SEATS` **Program Limits**

3 The Specification File and Its Syntax

Contents

Tables

The main input to `X-13ARIMA-SEATS` comes from a special input file called an input specification file. This file contains a set of specifications or **specs** that give `X-13ARIMA-SEATS` various information about the data and the desired seasonal adjustment options and output, the time series model to be used, if any, etc. Table 3.1 describes the different **specs** that are currently available in the `X-13ARIMA-SEATS` program.

Each spec is defined in the spec file by its name, which is followed by braces { } containing *arguments* and their assigned *values*. The arguments and their value assignments take the form *argument = value*, or, if multiple values are required, *argument* = (*value*$_1$, *value*$_2$, ...). There are various types of values: titles, variable names, keywords, numerical values, and dates. These are defined and illustrated in the documentation of the individual specs in Chapter 7. Because of their occurrence in several specs, detailed discussions of the **print** and **save** arguments (Section 3.2), and **date** argument values (Section 3.3) are given below.

There are no required arguments for any spec other than either **series** or **composite** (see below). Most arguments have default values; these are given in the documentation of each spec. Default values for all arguments are used if no arguments are specified.

Typically, not all specs are included in any one spec file. In fact, for most `X-13ARIMA-SEATS` runs (any that is not a composite run) there is only one required spec in the specification file — the **series** spec. This spec must include either the **data** or **file** argument. (The only exception is when a data metafile is used; see Section 2.5.2 for more details.) Thus, `X-13ARIMA-SEATS` will accept the minimal spec file `series { data =` (*data values*) }. However, this spec file produces no useful output.

For `X-11` seasonal adjustment runs, the **x11** spec is needed, unless one or more of the **force**, **x11regression**, **slidingspans**, or **history** (with the **estimates** argument set to perform a seasonal adjustment history) specs are present. In this case, `X-13ARIMA-SEATS` behaves exactly as if the **x11** spec were present with default arguments, which is equivalent to including `x11{ }` in the spec file. A model-based seasonal adjustment can be specified with the **seats** spec; you cannot specify both **x11** and **seats** specs in the same spec file.

For model identification runs, the **identify** spec is needed. For model estimation, the **arima** and/or **regression** specs, and the **estimate** spec are ordinarily included. If the **estimate** spec is absent, but one or more of

series	a required spec except when composite adjustment is done. It specifies the time series data, start date, seasonal period, span to use in the analysis, and series title.
composite	specifies that both a direct and an indirect adjustment of a composite series be performed; it is used instead of the **series** spec.
transform	specifies a transformation and/or prior adjustment of the data.
x11	specifies `X-11` seasonal adjustment options, including mode of adjustment, seasonal and trend filters, and some seasonal adjustment diagnostics.
x11regression	specifies irregular regression options, including which regressors are used and what type of extreme value adjustments will be made to robustify the regression on the irregular component.
seats	specifies options to perform an ARIMA model-based seasonal adjustment as in SEATS, a seasonal adjustment methodology developed by Victor Gómez and Agustin Maravall (see Gómez and Maravall (1996)),
force	specifies options to force the totals of the seasonally adjusted series to be the same as the original series,
automdl	specifies an automatic model selection procedure based on TRAMO (see Gómez and Maravall 1996 and Gómez and Maravall 2001a,
pickmdl	specifies an automatic model selection procedure based on `X-11-ARIMA/88` (see Dagum 1988),
arima	specifies the ARIMA part of the regARIMA model,
regression	specifies regression variables used to form the regression part of the regARIMA model, and to determine the regression effects removed by the **identify** spec,
estimate	requests estimation or likelihood evaluation of the model specified by the **regression** and **arima** specs, and also specifies estimation options,
check	produces statistics useful for diagnostic checking of the estimated model,
forecast	specifies forecasting with the estimated model,
outlier	specifies automatic detection of additive outliers and/or level shifts using the estimated model. There is an optional test for temporary level shifts,
identify	produces autocorrelations and partial autocorrelations for specified orders of differencing of the data with regression effects (specified by the **regression** spec) removed for ARIMA model identification,
slidingspans	specifies that a sliding spans analysis of seasonal adjustment stability be performed,
history	requests the calculation of a historical record of seasonal adjustment revisions and/or regARIMA model performance statistics.
metadata	allows users to specify metadata keys and values for storage in the diagnostics summary file.
spectrum	allows users to specify options related to the spectral plots generated by the program.

Table 3.1: `X-13ARIMA-SEATS` **Specifications**

the **outlier**, **automdl**, **pickmdl**, **check**, **forecast**, **x11**, **slidingspans** and **history** specs is present, this forces estimation of the specified model. In this case, `X-13ARIMA-SEATS` behaves exactly as if the **estimate** spec were present with default arguments, which is equivalent to including `estimate{ }` in the spec file. If the **arima** spec is absent, estimation proceeds with the default ARIMA(0 0 0) model (white noise). This is equivalent to including `arima{ }` in the spec file.

The order of the specification statements in the spec file (with a couple of exceptions), and the order of arguments within the braces of any spec do not matter. The only requirements are that (*i*) one of **series**, **composite**, or **metadata** must be the first spec, and (*ii*) if **metadata** is the first spec, then either **series** or **composite** must be the second. The spec file is free format, and blank spaces, tabs, and blank lines may be used as desired to make the spec file more readable. Comments can also be included. The use of comments and other general rules governing input syntax are discussed in Section 3.4. **Important:** There must be a carriage return at the end of the last line of the spec file, otherwise, this line will not be read. This is a Fortran requirement.

3.1 Examples of input specification files

A very simple spec file producing a default `X-11` run is given in Example 3.1. The spectrum diagnostics in the output file of this run indicated the presence of a trading day component, and a message saying this was written in the output. A regARIMA model can be used to both estimate the trading effect and to extend the series by forecasts prior to seasonal adjustment.

Example 3.1: `X-13ARIMA-SEATS` **spec file for a default** `X-11` **run**

```
series { title = "Monthly Retail Sales of Household Appliance Stores"
      data = ( 530  529  526  532  568  785  543  510  554  523  540  599
               574  619  619  600  652  877  597  540  594  572  592  590
               632  644  621  604  613  828  578  533  582  605  660  677
               682  684  700  705  747 1065  692  654  719  690  706  759
               769  730  740  765  791 1114  695  680  788  778  780  805
               852  823  831  836  913 1265  726  711  823  780  844  870
               865  915  920  935 1030 1361  859  852  954  895  993 1109
              1094 1173 1120 1159 1189 1539 1022  987 1024 1005 1054 1098
              1191 1191 1161 1201 1294 1782 1154 1059 1178 1126 1120 1233
              1260 1311 1302 1365 1395 1899 1123 1087 1210 1157 1159 1260
              1357 1265 1231 1287 1452 2186 1309 1242 1388 1400 1397 1527
              1654 1650 1555 1560 1836 2762 1541 1480 1619 1455 1510 1698
              1651 1749 1783 1863 2074 3051 1836 1690 1856 1796 1904 1927
              1978 2055 1976 2204 2423 3502 1977 1767 1935 1900 2073 2143
              2299 2247 2162 2274 2529 3731 2184 1901 2058 1974 2018 2091
              2239 2253 2157 2190 2397 3659 2170 2086 2297 2251 2311 2520
   )
start = 2002.jan}
x11{ }
```

Examples 3.2 and 3.3 illustrate spec files that might be used to identify the ARIMA part of the model before the final seasonal and trading day adjustment is achieved in Example 3.4. Alternatively, the `X-11` trading day adjustment procedures described in Section 7.20 could be used.

It is customary to make at least two runs of `X-13ARIMA-SEATS` when modeling a time series. The first run is usually done to permit identification of the ARIMA part of the model; the second run is done to estimate and check the regARIMA model, and possibly to use it in forecasting the series. The spec file for the first run requires the **series** and **identify** specs, and may also include the **transform** and **regression** specs. The spec file for the second run includes the **series**, **arima**, and **estimate** specs; possibly the **transform** and **regression** specs; and the **outlier**, **check**, and **forecast** specs as desired. The two runs of `X-13ARIMA-SEATS` require two different spec files, or, more conveniently, the spec file from the first run can be modified for use in the second run. If diagnostic checking suggests changes need to be made to the estimated model, then the spec file can be modified again to change the model for a third run of the program.

The contents of a typical spec file for the model identification run might follow the same format as Example 3.2.

Example 3.2: `X-13ARIMA-SEATS` **spec file for regARIMA model identification**

```
series{title = "Monthly Retail Sales of Household Appliance Stores"
       data = ( 530  529  526  532  568  785  543  510  554  523  540  599
                574  619  619  600  652  877  597  540  594  572  592  590
                 .
                 .
                 .
               2239 2253 2157 2190 2397 3659 2170 2086 2297 2251 2311 2520)
       start = 2002.jan}
transform{function = log}
regression{variables = td}     # Comment:  Series has trading-day effects
identify{diff=(0, 1) sdiff = (0, 1)}
```

This spec file includes the **series**, **transform**, **regression**, and **identify** specs. It provides `X-13ARIMA-SEATS` with the data given in the **series** spec, takes the logarithm of the series (**transform** spec), and specifies regression variables (**regression** spec) known or suspected to affect the series. Here, `variables = td` includes the six trading-day contrast variables (`td6`) in the model and also adjusts the series for leap year effects. (See Section 4.3 and the documentation of the **regression** spec in Section 7.13.) The **identify** spec performs a regression of the differenced transformed series (also adjusted for length-of-month effects) on the differenced regression variables (the six trading-day variables). The regression uses the highest order of seasonal and nonseasonal differencing specified, $(1 - B)(1 - B^{12})$. The **identify** spec then computes a regression residual series for the undifferenced data from which it produces tables and line printer plots of the sample autocorrelation and partial autocorrelation functions for all combinations of seasonal and nonseasonal differencing specified (here, four sets of ACFs and PACFs).

After studying the output from the first run and identifying the ARIMA part of the model as, for example, $(0\ 1\ 1)(0\ 1\ 1)_{12}$, the **identify** spec is commented out and the **arima** and **estimate** specs are added to the spec file. The resulting spec file is given in Example 3.3 (the data are not reproduced in full).

Example 3.3: `X-13ARIMA-SEATS` spec file for regARIMA model estimation

```
series{title = "Monthly Retail Sales of Household Appliance Stores"
       data = ( 530  529  526  532  568  785  543  510  554  523  540  599
                574  619  619  600  652  877  597  540  594  572  592  590
                 .
                 .
                 .
               2239 2253 2157 2190 2397 3659 2170 2086 2297 2251 2311 2520)
       start = 2002.jan}
transform{function = log}
regression{variables = td}     # Comment:  Series has trading-day effects
# identify{diff=(0, 1) sdiff = (0, 1)}
arima{model = (0,1,1)(0,1,1)}
estimate{print = iterations}
```

This spec file includes the **series**, **transform**, **regression**, **arima**, and **estimate** specs. It specifies (**regression** and **arima** specs) and fits (**estimate** spec) the following model:

$$(1-B)(1-B^{12})\Big(y_t - \sum_{i=1}^{6} \beta_i T_{it}\Big) = (1-\theta B)(1-\Theta B^{12})a_t,$$

where the T_{it} are the six trading-day regression variables. The series y_t being modelled consists of the logarithms of the original data adjusted for leap-year effects. If diagnostic checking of residuals, outlier detection, or forecasting were desired, the appropriate specs would need to be added to the spec file.

Assuming this is a satisfactory model, a seasonal adjustment utilizing forecast extension can be performed by adding the **x11** and **forecast** to the input specification file. Such a spec file appears in Example 3.4 (the data are not reproduced in full).

The spec file now generates seasonal adjustments from 3×9 seasonal filters (**x11**) for the trading day pre-adjusted series. The pre-adjusted series is extended by 60 forecasts (**forecast**) prior to seasonal adjustment. The main output file will contain some diagnostics concerning the quality of the seasonal adjustment. Additional diagnostics can be specified by including the appropriate specs described in Chapter 7.

3.2 Print and save

Control of the output from `X-13ARIMA-SEATS` is achieved within individual specs by using the **print** and **save** arguments. The **print** argument controls the given spec's output to the main output file, while the **save** argument allows certain output tables to be written to files. For ease of reference we refer to all the individual parts of the output subject to control through **print** and **save** as "tables," even though some of this output (e.g., line printer plots of an ACF) is not in a form that is ordinarily thought of as a table. The tables subject to control through **print** and **save** are listed with their default print status and file extensions (for savable tables) under the documentation of the **print** and **save** arguments for each spec. Tables output to files using **save**

Example 3.4: `X-13ARIMA-SEATS` **spec file for seasonal adjustment**

```
series{title = "Monthly Retail Sales of Household Appliance Stores"
       data = ( 530  529  526  532  568  785  543  510  554  523  540  599
                574  619  619  600  652  877  597  540  594  572  592  590
                 .
                 .
                 .
               2239 2253 2157 2190 2397 3659 2170 2086 2297 2251 2311 2520)
       start = 2002.jan}
transform{function = log}
regression{variables = td}     # Comment:  Series has trading-day effects
# identify{diff=(0, 1) sdiff = (0, 1)}
arima{model = (0,1,1)(0,1,1)}
estimate{print = iterations}
forecast{maxlead = 60}
x11{seasonalma = s3x9}
```

are written in a format with high numerical precision and with minimal or no labelling information to facilitate their use for further analysis utilizing other software. Saved tables are also given a consistent format – a single tab separates fields.

Default output from a spec is written to the main output file if the **print** argument is absent, or if `print = default` or `print = ()` appears in the spec. To stop a spec from writing output to the main output file, set `print = none`. (Note: A few specs write some minor labelling information to the screen even with `print = none`.) To have all the available output tables and plots for a spec written to the main output file, set `print = all`. To have all the available output tables (no plots) for a spec written to the main output file, set `print = alltables`. To have a small subset of the available output tables for a spec written to the screen, set `print = brief`. Individual tables may be added to the `default`, `brief`, and `none` print levels by including their names as print argument values. These may (but need not) be preceded by a `+`. For example, in the **estimate** spec, `print = (+iterations +residuals)`, which is equivalent to `print = (default +iterations +residuals)`, requests printing of results from the estimation iterations and the residuals from the estimated model, in addition to the default output. Using `print = (none estimates)` requests printing of only the parameter estimates. Individual tables may be suppressed from the `default` and `all` print levels by including their names preceded by a `-` as print argument values, e.g., `print = (brief -acf)` or `print = (all -iterations)`.

If the user wishes to save any output tables to files, these must be specifically listed in the **save** arguments of the appropriate specs, e.g., `save = (mdl estimates)` in the **estimate** spec. Those tables that are savable may be specified in the **print** and **save** arguments using either a "long" name, the name listed in the spec's description, or a "short" 3-letter name, which is the same as the file extension used if the table is saved. For example, the optional table `regcmatrix` in the **estimate** spec can also be specified as `rcm`. The keywords `none`, `all`, `alltables`, `default`, and `brief` defined above are not available for use in the **save** argument. Also, names of tables to be saved should not be preceded with a `+` or `-`. Not all tables are savable, and not all specs produce savable tables.

The **save** argument writes the specified tables to individual files. A saved file will be placed in the same directory as the output, and will be given the filename of the main output file, but with a distinct 3-letter

extension. If a file with this name already exists, it will be overwritten. The extensions used are listed under the documentation of the **print** and **save** arguments for each spec. For example, suppose `X-13ARIMA-SEATS` is run (on a DOS machine) from the directory `C:\TSERIES` using as input a spec file stored in `SALES.SPC` in that directory. If the **estimate** spec contains `save = (mdl estimates)`, the resulting saved tables of the model and parameter estimates will be written to the files `C:\TSERIES\SALES.MDL` and `C:\TSERIES\SALES.EST`. If files with these names already exist, they will be overwritten. Although the extensions used by `X-13ARIMA-SEATS` have been chosen to avoid obvious conflicts (examples of extensions not used are `.dat`, `.exe`, `.com`, `.for`, `.spc`), users should still exercise caution to prevent unintended overwriting of files by `X-13ARIMA-SEATS` saves. A list of the files saved, with an `*` indicating those overwriting existing files, appears at the beginning of the program's output. If there are errors in the spec file or the program terminates prematurely for other reasons, some or all of the saved files may not be written.

3.3 Dates

Date arguments occur in several specs, and their values are always specified in the same format. Dates for monthly data are written *year.month*; this format generalizes to other seasonal periods (e.g., *year.quarter*). It is necessary to include all four digits when specifying a year. Thus, **67** means the year AD (or CE) 67, not AD 1967.

For monthly data the months can be denoted by either the integers 1–12 or by three-letter month abbreviations (**jan**, **feb**, **mar**, **apr**, **may**, **jun**, **jul**, **aug**, **sep**, **oct**, **nov**, and **dec**). Thus, **1967.12** and **1967.dec** are equivalent. For quarterly data or data with other seasonal periods, only integers are allowed, e.g., **1967.1** and **1967.4**.

For data of any periodicity, a zero can be placed in front of integers from 1 to 9 for padding (for example, **2002.02** is an acceptable date specification for February 2002).

Dates are used to define the starting time point of a series, and when defining a subset (**span**) of a time series to analyze. They are also used when defining outlier regression variables. For example, to specify regression variables for an additive outlier in April of 2008 and a level shift beginning in September of 2012, we use the following:

```
regression  {  variables = (ao2008.apr ls2012.sep)  }
```

The seasonality of the dates used must match the seasonality specified for the data in the **series** spec, e.g., **ao2002.jan** is valid for monthly data but is not permitted for quarterly data.

3.4 General rules of input syntax

Allowable input characters

The allowable input characters, excepting characters that appear within quotes, are letters, numbers, spaces, tabs, newline characters, and the following:

```
= . , { } ( ) [ ] + - #
```

The program will ignore any other ASCII characters in the spec file, but will flag them and generate a warning message. The following additional characters are allowed within quotes:

```
! % ' * / : ; < > ? @ \ _ / ~ ^
```

Also, double quotes are allowed within statements delimited by single quotes and vice-versa.

Braces, parentheses, and brackets
The { }, ( ), and [ ] characters serve different functions and cannot be used interchangeably. { } is used to contain arguments in a spec, ( ) is used to contain a list of multiple values for an argument, and [ ] is used (*i*) to contain values used in defining certain special arguments, such as the duration of an Easter holiday regression variable, e.g., `regression {variables = (td Easter[14])}`, and (*ii*) to enclose the lags present in an ARIMA model with missing lags, e.g., `arima {model = (0 1 [1,3])}`.

Case sensitivity
Spec names, arguments, dates, keywords (such as `none` and `all`), and predefined regression variable names (such as `td` and `seasonal`) are not case sensitive. Thus, `TD` and `td` are the same; both are recognized by the **variables** argument of the **regression** and **x11regression** specs.

Comments
Anything on a line after the # character, unless the # character is in quotes, is taken to be a comment. If parts of a spec are commented out, what remains must still have balanced parentheses, brackets, and braces.

Equals sign
The equals sign, =, is used when assigning values to arguments, e.g., `print = none`, or `title = "Monthly Retail Sales of Household Appliance Stores"`.

Line length in the spec file
Lines in the spec file are limited to 132 characters—any characters appearing beyond column 132 are ignored. In particular, note that if a data set with lines exceeding 132 characters is placed in a spec file this will result in data truncation on input. The 132 characters per line limitation does not apply, however, to data read from a separate file (not the spec file) using the **file** argument. (The latter would be governed by Fortran input line length restrictions, which may be system specific.)

Multiple argument values
Multiple argument values must be enclosed together in parentheses, e.g., `variables = (td seasonal const)`. If an argument accepts only a single value or it accepts multiple values but only one value is given, then parentheses are optional. For example, the following are all valid; `variables = td`, `variables = (td)`, `variables = (td seasonal)`, `start = 1967.4`, and `start = (1967.4)`.

Null list
A null list of arguments is allowed, e.g., `outlier{ }`. Any implied arguments in the null list then take on their default values.

Numerical values
Numerical values can be specified in free format, including the use of exponential notation (e.g., 400, 400.0, 400., and 4.e+2 all denote the same real value). Integer notation must be used when an integer is required (e.g., 2, not 2.0 or 2.e+0).

Ordering

The only restrictions on the ordering of specs are (*i*) one of **metadata**, **series**, and **composite** must be the first spec, and (*ii*) if **metadata** is the first spec, then either **series** or **composite** must be the second spec. Except for the **b** argument of the **regression** and **x11regression** specs, there are no restrictions on the ordering of *arguments* within specs (see Sections 7.13 and 7.20 for more details). The ordering of multiple *values* given to arguments matters for certain obvious cases, such as observations in **data** arguments (**series**, **transform**, **regression**, and **x11regression** specs), the ARIMA model specification in the **model** argument (**arima** spec), and dates in **span** arguments (**series** and **outlier** specs).

Separators

Blank spaces, tabs, and blank lines may be used as separators as desired. Within a list of multiple argument values, single commas may also be used as separators, e.g., `data = (0,1,2,3,4,5)`. Commas *must* be used to indicate missing argument values that are to be replaced by default values (for arguments that require a specific number of values). For example, the **span** argument requires two values. In the statement `span = (1967.4,)`, the presence of the comma after 1967.4 indicates that the second **span** argument value is missing, so it takes on its default value (the date of the last observation).

Titles and filenames

A title, such as the name of a time series, must consist of at least one allowable input character (see above), even if blank, and must be enclosed in either single or double quotes (`'title'` or `''title''`). Lower and upper case of characters is preserved within titles. When the # character appears within quotes, it is considered part of the title and does not denote the start of a comment. Titles must be completed on one line and contain no more than 79 characters. Filenames, including the path, must follow the same rules as titles.

4 RegARIMA Modeling Capabilities

Contents

Tables

Section 4.1 describes the general model handled by the `X-13ARIMA-SEATS` program. Sections 4.2 to 4.7 give summary descriptions of the capabilities of `X-13ARIMA-SEATS` for the various stages of regARIMA modeling and forecasting: data input and transformation, regression variable specification, ARIMA model identification and specification, model estimation and inference, diagnostic checking including outlier detection, and forecasting. These sections also mention which input specification statements (specs) are used to control the execution of the capabilities discussed. Detailed documentation of the specs is given in Chapter 7.

When building a regARIMA model, it is strongly recommended that one examine a high resolution plot of the time series. Such a plot gives valuable information about seasonal patterns, potential outliers, stochastic nonstationarity, etc. Additional plots may also be useful for examining the effects of possible transformations on the series or of various differencing operators being applied to the series. Since `X-13ARIMA-SEATS` does not possess such plotting capabilities, other software must be used for this purpose.

4.1 General model

ARIMA models, as discussed by Box and Jenkins (1976), are frequently used for seasonal time series. A general multiplicative seasonal ARIMA model for a time series z_t can be written

$$\phi(B)\Phi(B^s)(1-B)^d(1-B^s)^D z_t = \theta(B)\Theta(B^s)a_t, \tag{4.1}$$

where B is the backshift operator ($Bz_t = z_{t-1}$), s is the seasonal period, $\phi(B) = (1 - \phi_1 B - \cdots - \phi_p B^p)$ is the nonseasonal autoregressive (AR) operator, $\Phi(B^s) = (1 - \Phi_1 B^s - \cdots - \Phi_P B^{Ps})$ is the seasonal AR operator, $\theta(B) = (1 - \theta_1 B - \cdots - \theta_q B^q)$ is the nonseasonal moving average (MA) operator, $\Theta(B^s) = (1 - \Theta_1 B^s - \cdots - \Theta_Q B^{Qs})$ is the seasonal MA operator, and the a_ts are i.i.d. with mean zero and variance

σ^2 (white noise). The $(1-B)^d(1-B^s)^D$ implies nonseasonal differencing of order d and seasonal differencing of order D. If $d = D = 0$ (no differencing occurs), it is common to replace z_t in (4.1) by deviations from its mean, that is, by $z_t - \mu$ where $\mu = E[z_t]$.

A useful extension of ARIMA models results from the use of a time-varying mean function modeled via linear regression effects. More explicitly, suppose we write a linear regression equation for a time series y_t as

$$y_t = \sum_i \beta_i x_{it} + z_t \tag{4.2}$$

where y_t is the (dependent) time series, the x_{it} are regression variables observed concurrently with y_t, the β_i are regression parameters, and $z_t = y_t - \sum \beta_i x_{it}$, the time series of regression errors, is assumed to follow the ARIMA model in (4.1). Modeling z_t as ARIMA addresses the fundamental problem with applying standard regression methodology to time series data, which is that standard regression assumes that the regression errors (z_t in (4.2)) are uncorrelated over time. For time series data, however, the errors in (4.2) will usually be autocorrelated, and, moreover, they will often require differencing. Hence, assuming z_t is uncorrelated in such cases will typically lead to grossly invalid results.

The expressions (4.1)) and (4.2) taken together define the general regARIMA model allowed by the `X-13-ARIMA-SEATS` program. Combining (4.1)) and (4.2), the model can be written in a single equation as

$$\phi(B)\Phi(B^s)(1-B)^d(1-B^s)^D\left(y_t - \sum_i \beta_i x_{it}\right) = \theta(B)\Theta(B^s)a_t. \tag{4.3}$$

The regARIMA model (4.3) can be thought of either as generalizing the pure ARIMA model (4.1) to allow for a regression mean function ($\sum \beta_i x_{it}$) or as generalizing the regression model (4.2) to allow the errors z_t to follow the ARIMA model (4.1). In either case, the regARIMA model implies that first the regression effects are subtracted from y_t to get the zero mean series z_t, and then the error series z_t is differenced to get a stationary series w_t, with w_t assumed to follow the stationary ARMA model, $\phi(B)\Phi(B^s)w_t = \theta(B)\Theta(B^s)a_t$. Another way to write the regARIMA model (4.3) is

$$(1-B)^d(1-B^s)^D y_t = \sum_i \beta_i (1-B)^d(1-B^s)^D x_{it} + w_t. \tag{4.4}$$

where w_t follows the stationary ARMA model just given. Equation (4.4) emphasizes that the regression variables x_{it} in the regARIMA model, as well as the series y_t, are differenced by the ARIMA model differencing operator $(1-B)^d(1-B^s)^D$.

Notice that the regARIMA model as written in (4.3) assumes that the regression variables x_{it} affect the dependent series y_t only at concurrent time points, i.e., model (4.3) does not explicitly provide for lagged regression effects such as $\beta x_{i,t-1}$. Lagged effects, however, can be included by the `X-13ARIMA-SEATS` program by reading in appropriate user-defined lagged regression variables.

The `X-13ARIMA-SEATS` program provides additional flexibility in the specification of the ARIMA part of a regARIMA model by permitting (*i*) more than two multiplicative ARIMA factors, (*ii*) missing lags within the AR and MA polynomials, (*iii*) the fixing of individual AR and MA parameters at user-specified values when the model is estimated, and (*iv*) inclusion of a *trend constant*, which is a nonzero overall mean for the differenced series ($(1-B)^d(1-B^s)^D y_t$). These features of regARIMA model specification are discussed and illustrated in Section 4.6.

Detailed discussions of ARIMA modeling are given in the classic book by Box and Jenkins (1976), and also in several other time series texts, such as Abraham and Ledolter (1983) and Vandaele (1983).

4.2 Data input and transformation

Observations of the original time series to be analyzed are read into the program with the **series** spec. The data may either be included within the **series** spec or read from a file. The **span** and **modelspan** arguments of the **series** spec are used to restrict the analysis to a span of the data, omitting data from the beginning and/or end of the original time series. The **series** spec is also used to specify the starting date, seasonal period (if appropriate), and title for the time series.

The **transform** spec provides nonlinear transformations of the data and modification by prior-adjustment factors. The nonlinear transformations included are the (Box and Cox 1964) family of power transformations (such as the logarithm or square root), and the *logistic* transformation (useful for a time series of proportions greater than 0 and less than 1). A predefined prior adjustment may be specified that divides each observation in a monthly series by the corresponding *length of month* (or *length of quarter* for quarterly series) and then re-scales it by the average length of month (or quarter). Similarly, leap year adjustment factors for February are also available. Finally, a set of user-defined prior-adjustments may be supplied for division into or subtraction from the original time series. The result of the **series** and **transform** specs is the time series $y_t,\ t = 1, \ldots, n$, used in the regARIMA model (4.3).

4.3 Regression variable specification

Specification of a regARIMA model requires specification of both the regression variables (the x_{it}'s in (4.2)) and the ARIMA model (4.1) for the regression errors z_t. The former is done using the **regression** spec, and the latter using the **arima** spec (discussed in Section 4.4). Choosing which regression variables to include requires user knowledge relevant to the time series being modeled. Several regression variables that are frequently used in modeling seasonal economic time series are built into the `X-13ARIMA-SEATS` program and can be easily included in the model. These are discussed below, and the actual regression variables used are given in Table 4.1 in this section. Specification and use of these variables is described in the documentation of the **regression** spec in Section 4.6. In addition, users may input data for any other regression variables (called user-defined regression variables) that they wish to include in a model. As part of model estimation (see Section 4.5), `X-13ARIMA-SEATS` provides standard t-statistics to assess the statistical significance of individual regression parameters, and χ^2-statistics to assess the significance of groups of regression parameters corresponding to particular effects (such as trading-day effects).

The most basic regression variable is the *constant term*. If the ARIMA model does not involve differencing, this is the usual regression intercept, which, if there are no other regression variables in the model, represents the mean of the (stationary) series. If the ARIMA model does involve differencing, `X-13ARIMA-SEATS` uses a regression variable such that, when it is differenced according to the ARIMA model (see equation (4.4)), a column of ones is produced. The corresponding parameter is then called a *trend constant*, since it provides for a polynomial trend of the same degree as the number of differences in the model. For example, with nonseasonal differencing ($d > 0$) but no seasonal differencing ($D = 0$), the (undifferenced) trend constant regression variable is proportional to t^d. Note that the lower order polynomial terms, t^j for $0 \leq j < d$, are not included among the regression variables because they would be differenced to zero by $(1 - B)^d$, hence their coefficients cannot be estimated. With or without the trend constant, the model (4.3) (or (4.4)) implicitly allows for these lower order polynomial terms through the differencing. If seasonal differencing is requested ($D > 0$), the nature of the undifferenced trend constant regression variable is more complicated, though the trend constant can be thought

of as allowing for a polynomial of degree $d + D$. Without a trend constant, model (4.3) implicitly allows for a polynomial of degree $d + D - 1$.

Fixed seasonal effects in a monthly series can be modeled using 12 indicator variables, one for each calendar month. Since these 12 variables always sum to one, however, they are confounded with an overall level effect. This leads to one of two singularity problems: collinearity with the usual constant term in a model with no differencing; or a singularity in a model with differencing since the 12 variables, when differenced, always sum to 0. One appropriate reparameterization instead uses 11 contrasts in the 12 indicator variables. An alternative reparameterization uses 11 variables taken from the Fourier (trigonometric) series representation of a fixed monthly pattern. The variables used for both of these parameterizations are given in Table 4.1. `X-13ARIMA-SEATS` allows either of these options and also allows specifying the trigonometric terms only for selected frequencies. For quarterly series, or for series with other seasonal periods, `X-13ARIMA-SEATS` constructs the appropriate versions of these variables. Note that these variables cannot be used in a model with seasonal differencing, as they would all be differenced to zero.

Trading-day effects occur when a series is affected by the differing day-of-the-week compositions of the same calendar month in different years. Trading-day effects can be modeled with 7 variables that represent (*no. of Mondays*), ..., (*no. of Sundays*) in month t. Bell and Hillmer (1983) note, however, that a better parameterization of the same effects instead uses 6 contrast variables defined as (*no. of Mondays*) − (*no. of Sundays*), ..., (*no. of Saturdays*) − (*no. of Sundays*), along with a seventh variable for *length of month* (`lom`) or its deseasonalized version, the leap-year regressor (`lpyear`). In `X-13ARIMA-SEATS` the 6 contrast variables are called the `tdnolpyear` variables. Instead of using a seventh regressor, a simpler and often better way to handle multiplicative leap-year effects is to re-scale the February values Y_t of the original time series before transformation to $\bar{m}_{Feb}Y_t/m_t$, where Y_t is the original time series before transformation, m_t is the length of month t (28 or 29), and $\bar{m}_{Feb} = 28.25$ is the average length of February. (If the regARIMA model includes seasonal effects, these can account for the length of month effect except in Februaries, so the trading day model only has to deal with the leap year effect.) When this is done, only the `tdnolpyear` variables need be included in the model. `X-13ARIMA-SEATS` allows explicit choice of either approach, as well as an option (`td`) that makes a default choice of how to handle length-of-month effects – see the documentation of the **regression** spec in Section 7.13.

The previous paragraph assumes the time series being modeled represents the aggregation of some daily series (typically unobserved) over calendar months. Such series are called monthly *flow* series. If the series instead represents the value of some daily series at the end of the month, called a monthly *stock* series, then different regression variables are appropriate. Trading-day effects in end-of-month stock series can be modeled using 7 indicator variables for the day-of-the-week that the months end on. Since the sum of these variables is always one, this leads to a singularity problem, so 6 contrast variables are used instead. (See Table 4.1.) `X-13ARIMA-SEATS` also allows specification of regression variables appropriate for stock series defined as of some other day of the month, e.g., for beginning of the month stock series.

For a general discussion of stock and flow series, access Wikipedia (2019).

For quarterly flow time series, `X-13ARIMA-SEATS` allows the same trading-day options as in the monthly case. Trading-day effects in quarterly series are relatively rare, however, because the calendar composition of quarters does not vary as much over time, on a percentage basis, as that of months does. Trading-day variables are not provided for flow time series with seasonal periods other than monthly or quarterly, or for stock series other than monthly.

Table 4.1: **Predefined Regression Variables in `X-13ARIMA-SEATS`**

Regression effect[2]	*Variable definition(s)*
Trend Constant `const`	$(1-B)^{-d}(1-B^s)^{-D}I(t \geq 1)$, where $I(t \geq 1) = \begin{cases} 1 & \text{for } t \geq 1 \\ 0 & \text{for } t < 1 \end{cases}$
Fixed Seasonal[3] `seasonal`	$M_{1,t} = \begin{cases} 1 & \text{in January} \\ -1 & \text{in December} \\ 0 & \textit{otherwise} \end{cases}, \ldots, \quad M_{11,t} = \begin{cases} 1 & \text{in November} \\ -1 & \text{in December} \\ 0 & \text{otherwise} \end{cases}$
Fixed Seasonal[4] `sincos[ ]`	$\sin(\omega_j t)$, $\cos(\omega_j t)$, where $\omega_j = 2\pi j/12$, $1 \leq j \leq 6$ (Drop $\sin(\omega_6 t) \equiv 0$)
Trading Day (monthly or quarterly flow) `tdnolpyear`, `td`[5]	$T_{1,t}$ = (no. of Mondays) − (no. of Sundays), …, $T_{6,t}$ = (no. of Saturdays) − (no. of Sundays)
One Coefficient Trading Day (monthly or quarterly flow) `td1nolpyear`, `td1coef`[6]	(no. of weekdays) − $\frac{5}{2}$(no. of Saturdays and Sundays)
Length-of-Month (monthly flow) `lom`	$m_t - \bar{m}$, where m_t = length of month t (in days) and $\bar{m} = 30.4375$ (average length of month)
Length-of-Quarter (quarterly flow) `loq`	$q_t - \bar{q}$, where q_t = length of quarter t (in days) and $\bar{q} = 91.3125$ (average length of quarter)
Leap Year (monthly and quarterly flow) `lpyear`	$LY_t = \begin{cases} 0.75 & \text{in leap year February (first quarter)} \\ -0.25 & \text{in other Februaries (first quarter)} \\ 0.00 & \text{otherwise} \end{cases}$

[2]Restrictions, if any, are given in parentheses. Each entry also gives the name used to specify the regression effect in the **variables** argument of the **regression** spec, e.g., `regression { variables=const }`.

[3]The variables shown are for monthly series. Corresponding variables are available for any other seasonal period.

[4]See footnote 3.

[5]In addition to these 6 variables, the `td` option also includes the `lpyear` regression variable (for untransformed series), or it re-scales February values of Y_t to $\bar{m}_{Feb}Y_t/m_t$, where $\bar{m}_{Feb} = 28.25$ (average length of February) (for an original series Y_t that is transformed). Quarterly `td` is handled analogously.

[6]In addition to this variable, the `td1coef` option also includes the `lpyear` regression variable (for untransformed series), or it re-scales February values of Y_t to $\bar{m}_{Feb}Y_t/m_t$, where $\bar{m}_{Feb} = 28.25$ (average length of February) (for an original series Y_t that is transformed). Quarterly `td1coef` is handled analogously.

Table 4.1: **Predefined Regression Variables in `X-13ARIMA-SEATS`** (continued)

Regression effect	*Variable definition(s)*
Stock Trading Day (monthly stock) `tdstock[`w`]`	$D(w)_{1,t} = \begin{cases} 1 & \widetilde{w}^{\text{th}} \text{ day of month } t \text{ is a Monday} \\ -1 & \widetilde{w}^{\text{th}} \text{ day of month } t \text{ is a Sunday} \\ 0 & \text{otherwise} \end{cases},$ $\cdots, \quad D(w)_{6,t} = \begin{cases} 1 & \widetilde{w}^{\text{th}} \text{ day of month } t \text{ is a Saturday} \\ -1 & \widetilde{w}^{\text{th}} \text{ day of month } t \text{ is a Sunday} \\ 0 & \text{otherwise} \end{cases},$ where $\widetilde{w}$ is the smaller of w and the length of month t. For end-of-month stock series, set w to 31, i.e., specify `tdstock[31]`.
One Coefficient Stock Trading Day[7] (monthly stock) `tdstock1coef [`w`]`	$I(w)_t = -\frac{3}{5}D(w)_{1,t} - \frac{1}{5}D(w)_{2,t} + \frac{1}{5}D(w)_{3,t} + \frac{3}{5}D(w)_{4,t} + D(w)_{5,t}$ where $D_{i,t}$ is the i-th stock trading day regressor defined as above for stock day w. For end-of-month stock series, set w to 31, i.e., specify `tdstock1coef[31]`.
Easter Holiday[8] (monthly or quarterly flow) `easter[`w`]`	$E(w,t) = \frac{1}{w} \times$ [no. of the w days before Easter falling in month (or quarter) t]. (Note: This variable is 0 except in February, March, and April (or first and second quarter). It is nonzero in February only for $w > 22$.)
Easter Holiday[9] (monthly or quarterly flow) `easter[0]`	Extension of the `easter[`w`]` regressor: $E(0,t) = \begin{cases} 1 & \text{Easter in month or quarter t} \\ 0 & \text{otherwise} \end{cases}$ (Note: This variable is 0 except in March and April (or first and second quarter).)
End-of-Month Stock Easter[10] (monthly or quarterly stock) `easterstock[`w`]`	Let $E_f(w,t)$ denote the flow `easter[w]` regressor with long-term means removed. Then, in the monthly case: $E_s(w,t) = \begin{cases} E_f(w,t) & \text{in February} \\ E_f(w,t) + E_f(w,t-1) & \text{in March} \\ 0 & \text{otherwise} \end{cases}.$ (Note: This variable is 0 except in February and March (or first quarter). It is nonzero in February only for $w > 22$.)

[7] For details on the derivation of this regressor, see Findley and Monsell (2009).

[8] The actual variable used for monthly Easter effects is $E(w,t) - \bar{E}(w,t)$, where the $\bar{E}(w,t)$ are the "long-run" monthly means of $E(w,t)$ corresponding to a 500 year period of the Gregorian calendar, 1600-2099. This provides a close approximation to the average calculated over the much longer period of a complete cycle of the dates of Easter. For more details, see Bednarek (2019) and Montes (2001). (These means are nonzero only for February, March, and April). Analogous deseasonalized variables are used for Labor Day and Thanksgiving effects, and for quarterly Easter effects.

[9] See footnote 8.

[10] An analogous variable is available for quarterly stock series, with the formula documented in Section 4.2 of Findley (2009).

Table 4.1: **Predefined Regression Variables in `X-13ARIMA-SEATS`** (continued)

Regression effect	*Variable definition(s)*
Statistics Canada Easter (monthly or quarterly flow) `sceaster[`w`]`	If Easter falls before April w, let n_E be the number of the w days on or before Easter falling in March. Then: $$E(w,t) = \begin{cases} n_E/w & \text{in March} \\ -n_E/w & \text{in April} \\ 0 & \text{otherwise} \end{cases}.$$ If Easter falls on or after April w, then $E(w,t) = 0$. (Note: This variable is 0 except in March and April (or first and second quarter).)
Labor Day[11] (monthly flow) `labor[`w`]`	$L(w,t) = \frac{1}{w} \times$ [no. of the w days before Labor Day falling in month t]. (Note: This variable is 0 except in August and September.)
Thanksgiving[12] (monthly flow) `thank[`w`]`	$ThC(w,t) =$ proportion of days from w days before Thanksgiving through December 24 that fall in month t (negative values of w indicate days after Thanksgiving). (Note: This variable is 0 except in November and December.)
Additive Outlier at t_0 `ao`$date_0$	$$AO_t^{(t_0)} = \begin{cases} 1 & \text{for } t = t_0 \\ 0 & \text{for } t \neq t_0 \end{cases}$$ ($date_0$ is the date corresponding to time point t_0)
Level Shift at t_0 `ls`$date_0$	$$LS_t^{(t_0)} = \begin{cases} -1 & \text{for } t < t_0 \\ 0 & \text{for } t \geq t_0 \end{cases}$$
Temporary Change at t_0 `tc`$date_0$	$$TC_t^{(t_0)} = \begin{cases} 0 & \text{for } t < t_0 \\ \alpha^{t-t_0} & \text{for } t \geq t_0 \end{cases},$$ where α is the rate of decay back to the previous level ($0 < \alpha < 1$).
Seasonal Outlier at t_0 `so`$date_0$	$$SO_t^{(t_0)} = \begin{cases} 0 & \text{for } t \geq t_0 \\ -1 & \text{for } t < t_0,\ t \text{ same month/quarter as } t_0 \\ 1/(s-1) & \text{otherwise} \end{cases}$$

[11] See footnote 8.
[12] See footnote 8.

Table 4.1: **Predefined Regression Variables in** `X-13ARIMA-SEATS` (continued)

Regression effect	*Variable definition(s)*
Ramp, t_0 **to** t_1 `rp`$date_0$`-`$date_1$	$RP_t^{(t_0,t_1)} = \begin{cases} t_0 - t_1 & \text{for } t \le t_0 \\ t - t_1 & \text{for } t_0 < t < t_1 \\ 0 & \text{for } t \ge t_1 \end{cases}$
Quadratic Ramp (Increasing), t_0 **to** t_1 `qi`$date_0$`-`$date_1$	$QI_t^{(t_0,t_1)} = \begin{cases} -(t_1 - t_0)^2 & \text{for } t \le t_0 \\ (t - t_0)^2 - (t_1 - t_0)^2 & \text{for } t_0 < t < t_1 \\ 0 & \text{for } t \ge t_1 \end{cases}$
Quadratic Ramp (Decreasing), t_0 **to** t_1 `qd`$date_0$`-`$date_1$	$QD_t^{(t_0,t_1)} = \begin{cases} -(t_1 - t_0)^2 & \text{for } t \le t_0 \\ -(t_1 - t)^2 & \text{for } t_0 < t < t_1 \\ 0 & \text{for } t \ge t_1 \end{cases}$
Temporary Level Shift, t_0 **to** t_1 `tl`$date_0$`-`$date_1$	$TL_t^{(t_0,t_1)} = \begin{cases} 0 & \text{for } t < t_0 \\ 1 & \text{for } t_0 \le t \le t_1 \\ 0 & \text{for } t > t_1 \end{cases}$
Additive Outlier Sequence, t_0 **to** t_1 `aos`$date_0$`-`$date_1$	This adds a sequence of AO variables (as defined previously) beginning at t_0 and ending on t_1. That is, `aos1950.jan-1950.oct` is equivalent to listing `ao1950.jan ao1950.feb ao1950.mar ...ao1950.oct` individually. For convenience, `0.0` represents the end of the series; e.g., `aos2020.jan-0.0` would add a sequence of AO variables beginning at January 2020.
Level Shift Sequence, t_0 **to** t_1 `lss`$date_0$`-`$date_1$	The level shift counterpart to AOS, this adds a sequence of LS variables (as defined previously) beginning at t_0 and ending on t_1.

`X-13ARIMA-SEATS` also provides a simplified model for trading day variation of monthly or quarterly flow series that uses only one regressor, a weekday-weekend contrast variable:

$$T_t = (\text{no. of Weekdays}) - \frac{5}{2}(\text{no. of Saturdays and Sundays})$$

The underlying assumption for this model is that all weekdays (Monday through Friday) have identical effects, and Saturday and Sunday have identical effects. In `X-13ARIMA-SEATS` this model can be estimated in

two ways: by specifying the `td1coef` option if the user wishes the program to make the choice of how to handle length of month effects as with the `td` option mentioned above, or by specifying the `td1nolpyear` option in which case the length of month effects model must be specified by the user, as with `tdnolpyear`.

The daily constraints from the flow series given above can be applied to the case of stock trading day as well. The one-coefficient stock trading day variable for stock day w is given below:

$$I(w)_t = -\frac{3}{5}D(w)_{1,t} - \frac{1}{5}D(w)_{2,t} + \frac{1}{5}D(w)_{3,t} + \frac{3}{5}D(w)_{4,t} + D(w)_{5,t}$$

See Findley and Monsell (2009) for more details, along with an application using industrial inventory series.

Holiday effects (in a monthly flow series) arise from holidays whose dates vary over time if (i) the activity measured by the series regularly increases or decreases around the date of the holiday, and (ii) this differentially affects two (or more) months depending on the date the holiday occurs each year. (Effects of holidays with a fixed date, such as Christmas, are indistinguishable from fixed seasonal effects.) *Easter effects* are the most frequently found holiday effects in U.S. economic time series, since the date of Easter Sunday varies between March 22 and April 25. *Labor Day* and *Thanksgiving* also are potential, though less common, sources of holiday effects. The basic model used by `X-13ARIMA-SEATS` for Easter and Labor Day effects assumes that the level of activity changes on the w-th day before the holiday for a specified w, and remains at the new level through the day before the holiday. For Thanksgiving the model used assumes that the level of activity changes on the day that is a specified number of days before or after Thanksgiving and remains at the new level through December 24. The regression variable constructed for the holiday effect is, for a given month t, the proportion of the affected time period that falls in month t. In addition, `easter[0]` indicates an Easter effect in the month/quarter containing Easter, zero otherwise. (Actually, as noted in Table 4.1, these regressors are deseasonalized by subtracting off their long-run monthly means.) Essentially the same Easter effect variable applies also to quarterly flow time series, but Labor Day and Thanksgiving effects are not present in quarterly series.

For stock series, the perspective of stocks as accumulations of monthly flows can be used to derive holiday regressors for end of month stock series from cumulative sums of holiday regressors for flow series. A similar approach has been used to obtain useful stock series trading day regressors by Cleveland and Grupe (1983), Bell (1984, 1995), and Findley and Monsell (2009). While this could be done for any of the moving holidays in `X-13ARIMA-SEATS`, the only holiday currently with an end of month stock implementation is Easter; regressors are available for both monthly and quarterly series.

Let $E(w)^{flow}_{m,y}$ denote the deseasonalized `easter[w]` regressor for a monthly flow series derived for month m and year y. The end of month stock Easter regressor $E^{stock}_{m,y}$ is generated as follows:

$$E(w)^{stock}_{m,y} = \begin{cases} 0 & \text{for } m = 1 \\ E(w)^{flow}_{2,y} & \text{for } m = 2 \\ E(w)^{flow}_{3,y} + E(w)^{flow}_{2,y} & \text{for } m = 3 \\ 0 & \text{for } 4 \leq m \leq 12. \end{cases}$$

When $1 \leq w \leq 21$, $E(w)^{flow}_{m,y}$ is zero in February, so $E(w)^{stock}_{m,y}$ is nonzero only in March. See Findley (2009) for more details and an application for manufacturing inventory series, and Titova and Monsell (2009) and Chow and Moore (2009) for applications on U.S. and U.K. inventory series.

X-13ARIMA-SEATS provides several other types of regression variables to deal with abrupt changes in the level of a series of a temporary or permanent nature: *additive outliers* (AOs), *level shifts* (LSs), *temporary changes* (TCs), *seasonal outliers* (SOs), *ramps*, *quadratic ramps* (QDs and QIs), and *temporary level shifts* (TLs).

AOs affect only one observation in the time series.

LSs increase or decrease all observations from a certain time point onward by some constant amount.

TCs allow for an abrupt increase or decrease in the level of the series, with an exponentially rapid return to its previous level.

SOs allow for an abrupt increase or decrease in the level of the seasonal pattern that is compensated for in the other months or quarters.

Ramps allow for a linear increase or decrease (i.e., the rate of change is constant) in the level of the series over a specified time interval. Quadratic ramps are ramps where the rate of change in the level of the series is not constant – QDs have a decreasing (in magnitude) rate of change during the ramp phase, whereas QIs have an increasing rate of change.

Temporary level shifts increase or decrease all observations for a specific time span contained within the series by some constant amount.

There are also sequence outlier variables associated with AOs and LSs. AOSs add a sequence of AOs over a specified time interval – in function, it performs the equivalent of adding each AO in a time interval individually. LSSs are the level shift analogue to the AOSs – they add a sequence of LSs over a specified time interval to a regression.

The specific regression variables used to model these effects are given in Table 4.1. (LS regression variables are defined as -1 and then 0, in preference to an equivalent 0 and then 1 definition, to make the overall level of the regression mean function of any forecasts consistent with the most recent level of the time series. Similar considerations dictate the definition of ramp variables and seasonal outliers.)

The **regression** spec allows all of these variables to be specified for cases where prior knowledge suggests the presence of such effects at known time points. Often, however, large seasonal movements make it difficult to identify where such changes in level have occurred. Determination of the location and nature of potential outliers is the objective of the outlier detection methodology implemented by the **outlier** spec – see Section 4.6 and the **outlier** spec documentation in Section 7.11. This methodology can be used to detect AOs, TCs, and LSs (but not ramps, quadratic ramps, seasonal outliers, temporary level shifts, or the sequence version of AOs/LSs); any that are detected are automatically added to the model as regression variables.

Prespecified AOs (and AOSs), LSs (and LSSs), TCs, SOs, TLs, QDs, QIs, and ramps are actually simple forms of *interventions* as discussed by Box and Tiao (1975). While X-13ARIMA-SEATS does not provide the full range of dynamic intervention effects discussed by Box and Tiao, often a short sequence of suitably chosen AO, LS, TC, TL, QD, QI, and/or ramp variables can produce reasonable approximations to more complex dynamic intervention effects, albeit at the cost of an additional parameter or two. Analogous remarks apply to the relation between regARIMA models containing (user-defined) regression variables that are themselves stochastic time series, and the dynamic transfer function models discussed by Box and Jenkins (1976), chapters 10 and 11. Thus, regARIMA models can often be used to approximate more general dynamic transfer function models, although transfer function models require special treatment when forecasting, since future values of stochastic explanatory variables are generally unknown. (See Box and Jenkins 1976, Section 11.5).

4.4 Identification and specification of the ARIMA part of the model

The ARIMA part of a regARIMA model is determined by its orders and structure, e.g., $(p\ d\ q)$, $(P\ D\ Q)$, and s for model (4.1). If no regression variables are included in the model, then determination of the orders for the resulting pure ARIMA model (called ARIMA model identification) can be carried out by following well-established procedures that rely on examination of the sample autocorrelation function (ACF) and sample partial autocorrelation function (PACF) of the time series y_t and its differences. For regARIMA models, a modified approach is needed, since the presence of regression effects can distort the appearance of the ACF and PACF. Typically, the differencing orders can be identified by examining ACFs of the time series y_t and its differences. Then, one should obtain the residuals from a regression of the differenced data on the differenced regression variables. The ACF and PACF of these residuals can then be examined in the usual way to identify the AR and MA orders of the regression error term in the regARIMA model. This approach to regARIMA model identification is discussed and illustrated in Bell and Hillmer (1983).

The key spec used to implement this approach to regARIMA model identification is the **identify** spec. For illustration, consider a monthly seasonal time series. The usual ACFs examined to determine the differencing needed are those of y_t, $(1-B)y_t$, $(1-B^{12})y_t$, and $(1-B)(1-B^{12})y_t$. The **identify** spec can produce these ACFs in a single run. Once the differencing has been determined, another run of `X-13ARIMA-SEATS` can be made using the **identify** and **regression** specs together to (i) regress the differenced y_t series on the differenced regression variables, and (ii) produce the ACF and PACF of the regression residuals for use in identifying the AR and MA orders of the model. For example, if one nonseasonal and one seasonal difference are specified ($d = 1$ and $D = 1$), the **identify** and **regression** specs will fit the model

$$(1-B)(1-B^{12})y_t = \sum_i \beta_i (1-B)(1-B^{12})x_{it} + w_t \tag{4.5}$$

by ordinary least squares (OLS) and will produce the ACF and PACF of the regression residuals w_t in (4.5).

An alternative approach that does not require two runs of the `X-13ARIMA-SEATS` program can be used if the maximum differencing orders (nonseasonal and seasonal) that may be required are assumed to be known. For example, suppose that these maximum differencing orders are $d = 1$ and $D = 1$. Then the **identify** and **regression** specs can be used to (i) perform OLS regression on (4.5) to produce parameter estimates $\widetilde{\beta}_i$, (ii) compute an estimated (undifferenced) regression error series $\widetilde{z}_t = y_t - \sum_i \widetilde{\beta}_i x_{it}$, and (iii) produce ACFs and PACFs of $\widetilde{z}_t$, $(1-B)\widetilde{z}_t$, $(1-B^{12})\widetilde{z}_t$, and $(1-B)(1-B^{12})\widetilde{z}_t$. These ACFs and PACFs can be examined to determine the orders of differencing required, as well as the orders of the AR and MA operators for the model.

There is one exception to the above remarks. If a constant term is specified in the **regression** spec, then it will be included when the OLS regression is done on (4.5), but not when the regression effects are removed from the data. Thus, actually, $\widetilde{z}_t = y_t - \sum_{i\geq 2} \widetilde{\beta}_i x_{it}$ if $\widetilde{\beta}_1 x_{1t}$ is the trend constant term. To explain why this is done, we consider (4.5). From remarks in Section 4.3, a trend constant variable in model (4.5) allows for a polynomial of degree 2, although the constant and linear terms (for $t^0 \equiv 1$ and t) are implicitly allowed for through the differencing by $(1-B)(1-B^{12})$. Since the constant and linear coefficients cannot be estimated, the full polynomial effect cannot be subtracted from the undifferenced series y_t. Rather than subtract only the t^2 term of the polynomial, `X-13ARIMA-SEATS` ignores the estimated trend constant when creating the undifferenced regression error series $\widetilde{z}_t$. Similar remarks apply to the general model (4.4). The only effect that inclusion of a trend constant has on the computations of the **identify** spec is that its inclusion in (4.4) will affect the regression estimates $\widetilde{\beta}_i$ for $i \geq 2$.

4.5 Model estimation and inference

The **regression** and **arima** specs specify a regARIMA model. The **estimate** spec then estimates the model parameters by exact maximum likelihood, or by a variant known as conditional maximum likelihood (Box and Jenkins 1976, pp. 209–212), which is sometimes called conditional least squares. Users may specify maximization of the full exact likelihood, or of the likelihood conditional for the AR but exact for the MA parameters, or of the likelihood conditional for both the AR and MA parameters. Differences in AR parameter estimation between exact and conditional likelihood maximization are generally small, and there are situations where each approach is appropriate. (See Chapter 5.) Differences between exact and conditional likelihood for MA parameter estimation are more fundamental, with exact likelihood being the recommended approach. The option of choosing the conditional likelihood for MA parameters is provided in `X-13ARIMA-SEATS` mainly for comparison of results with other software, and for occasional use to produce initial estimates for exact maximum likelihood estimation when convergence problems arise. (See Section 5.1.) The default option is exact maximum likelihood estimation for both the AR and MA parameters.

Whichever choice of estimation method is made, the resulting log-likelihood for a pure ARIMA model is reduced to a sum of squares function that is then minimized by a nonlinear least squares routine (MINPACK, discussed by More, Garbow, and Hillstrom (1980). To maximize the likelihood for a full regARIMA model, an iterative generalized least squares (IGLS) algorithm (Otto, Bell, and Burman 1987) is used. This algorithm involves two general steps: (i) for given values of the AR and MA parameters, the regression parameters that maximize the likelihood are obtained by a generalized least squares (GLS) regression (using the covariance structure of the regression errors, which is determined by their ARIMA model), and (ii) for given values β_i of the regression parameters, the ARIMA model is fit by maximum likelihood to the time series of regression errors, $z_t = y_t - \sum \beta_i x_{it}$. IGLS iterates between these two general steps until convergence is achieved. (Output options in the **estimate** spec allow for display of intermediate results from the estimation iterations, if desired.) The likelihood function (exact or conditional) is evaluated using an approach derived from those suggested by Box and Jenkins (1976), chapter 7, Ljung and Box (1979), Hillmer and Tiao (1979), and Wilson (1983). Section 4.4 discusses certain problems that may arise in model estimation that all users should be aware of.

Statistical inferences about regARIMA model parameters may be made using asymptotic results for maximum likelihood estimation of ARIMA models (Box and Jenkins 1976, chapter 7; Brockwell and Davis 1991, chapter 8) and regARIMA models (Pierce 1971). These results state that, under suitable assumptions, the parameter estimates are approximately normally distributed with means equal to the true parameter values and with a certain covariance matrix that can be estimated. (The "suitable assumptions" include that the true model form is used, that the model's AR operators are all stationary and its MA operators are all invertible, and that the series is sufficiently long for the asymptotic results to apply.) Using these results, `X-13ARIMA-SEATS` provides standard errors for the ARMA and regression parameter estimates, and, optionally, correlation (or covariance) matrices for the estimates of both the ARMA and the regression parameters. (The regression parameter estimates are asymptotically uncorrelated with the ARMA parameter estimates.) These results may be used in the usual way to make normal theory inferences about model parameters, including, as mentioned in Section 4.3, use of t-statistics and χ^2-statistics produced by `X-13ARIMA-SEATS` to assess the statistical significance of individual regression parameters and of groups of regression parameters corresponding to particular regression effects. Also, since `X-13ARIMA-SEATS` prints out the value of the maximized log-likelihood function, various likelihood ratio tests are possible by making multiple runs of the program with different models.

`X-13ARIMA-SEATS` uses the maximum likelihood estimate of the residual variance σ^2, which is $\widehat{\sigma}^2 = \mathrm{SS}/(n - d - s \cdot D)$, where SS is the residual sum-of-squares and $n - d - s \cdot D$ is the effective number of observations after

differencing. (If the likelihood function that is conditional with respect to the AR parameters is used, replace $n-d-s\cdot D$ by $n-p-d-s\cdot P-s\cdot D$.) Notice there is no "degrees of freedom" adjustment for model parameters being estimated. For this reason, if `X-13ARIMA-SEATS` is used to fit a pure regression model – a model whose regression errors follow the ARIMA(0 0 0) model – $\widehat{\sigma}^2$ will differ from the usual unbiased regression variance estimate. Consequently, the resulting standard errors, t-statistics, and χ^2-statistics for the regression parameter estimates will also differ slightly from those that would be obtained from a standard regression program.

An alternative approach to inference is to use the likelihood-based model selection criteria produced by `X-13ARIMA-SEATS`: AIC, AICC (also known as the F-adjusted AIC), Hannan-Quinn, and BIC. For each of these statistics, the model producing the lower value is preferred. One advantage to these criteria over standard t-statistics, χ^2-statistics, and likelihood ratio tests is that they may be used to compare *non-nested* models – models that differ from each other in such a way that one model cannot be obtained simply by removing parameters from another model. (E.g., AR(1) versus MA(1) is a non-nested comparison.) Some caution must be exercised in use of the model selection criteria. Section 5.5 discusses certain situations that arise in regARIMA modeling for which the use of these criteria, as well as standard likelihood ratio tests, is invalid.

4.6 Diagnostic checking including outlier detection

Diagnostic checking of a regARIMA model is performed through analysis of the residuals from model estimation, the objective being to check if the true residuals (a_t in (4.3)) appear to be white noise – i.i.d. $N(0,\sigma^2)$. (Note: Normality of the a_ts is not needed for the large sample estimation and inference results; it is most important for validity of prediction intervals produced in forecasting.) The **check** spec is used to produce various diagnostic statistics using the residuals from the fitted model. To check for autocorrelation, `X-13ARIMA-SEATS` can produce ACFs and PACFs of the residuals (with standard errors), along with Ljung and Box (1978) summary Q-statistics. `X-13ARIMA-SEATS` can also produce basic descriptive statistics of the residuals and a histogram of the standardized residuals. The residuals can be written to a file for further analysis (such as high resolution plotting) by other software.

The residuals produced by the algorithm used for exact maximum likelihood estimation in `X-13ARIMA-SEATS` are the conditional expectations of the innovations a_t given the data $\{y_t; t=1,\ldots,n\}$, that is, $E[a_t|\{y_t\}]$. (See Box and Jenkins 1976, Appendix A7.4.) They are computed using the estimated values of the model parameters. For a nonseasonal ARIMA model of order (p,d,q), the residuals are computed for time points $t=d+1-q,\ldots,n$, which gives $n-d+q$ residuals. For a seasonal ARIMA model with seasonal period s, the first residual would be at $t=d+Ds-q-Qs+1$. For the airline model $d=q=1$ and $D=Q=1$, so there will be n residuals going from $t=1,\ldots,n$. Other algorithms that can be used to evaluate or approximate the likelihood of an ARIMA model – e.g., the conditional likelihood option in `X-13ARIMA-SEATS`, or the Kalman filter (used in other software) – can produce different residuals, even different numbers of residuals (even when the approaches yield the same likelihood value). If the estimated model is invertible, residuals computed in these different ways will eventually converge to each other as t increases, and so will typically be very close in the latter part of the time series, and may be very similar for most of the time series.

An important aspect of diagnostic checking of time series models is outlier detection. The **outlier** spec of `X-13ARIMA-SEATS` provides for automatic detection of additive outliers (AOs), temporary change outliers (TCs) and level shifts (LSs). These outlier types (referring to AOs, TCs, and LSs as "outliers") and their associated regression variables are defined in Section 4.3. `X-13ARIMA-SEATS`' approach to outlier detection is based on that of Chang and Tiao (1983) – see also Chang, Tiao, and Chen (1988) – with extensions and modifications as

discussed in Bell (1983) and Otto and Bell (1990). The general approach is similar to stepwise (GLS) regression, where the candidate regression variables are AO, LS, and/or TC variables for all time points at which outlier detection is being performed – $3n$ variables for detection of AOs, LSs, and TCs over an entire time series of length n. (Actually, slightly fewer than $3n$ variables are used in this case for reasons discussed in the DETAILS section of the **outlier** spec documentation in Section 7.11.) In brief, this approach involves computing t-statistics for the significance of each outlier type at each time point, searching through these t-statistics for significant outlier(s), and adding the corresponding AO, LS, or TC regression variable(s) to the model. Overly burdensome computation is avoided by holding the AR and MA parameters fixed as the outlier t-statistics are computed for each time point and outlier type. `X-13ARIMA-SEATS` provides two variations on this general theme. The `addone` method provides full model re-estimation after any outlier is added to the model, while the `addall` method re-estimates the model only after a set of detected outliers is added. A description of both these methods is given in the documentation of the **outlier** spec in Section 7.11, with more details in Appendix B of Findley, Monsell, Bell, Otto, and Chen (1998).

During outlier detection a robust estimate of the residual standard deviation, 1.48 $\times$ the median absolute deviation of the residuals (Hampel, Ronchetti, Rousseeuw, and Stahel 1986, p. 105), is used. Because outlier detection involves searching over all (or a specified set of) time points for the most significant outliers, the usual normal distribution critical values (e.g., 2.0) are too low for judging significance in outlier detection. The default critical value is determined by the number of observations in the interval searched for outliers (see Table 7.22), but this can be changed by the user.

When a model contains two or more level shifts, including those obtained from outlier detection as well as any level shifts specified in the **regression** spec, `X-13ARIMA-SEATS` will optionally produce t-statistics for testing null hypotheses that each run of two, three, etc. successive level shifts actually cancels to yield a net effect of zero beyond the last level shift in the run. The t-statistics produced are the sums of the estimated parameters for each run of successive level shifts divided by the appropriate standard error. An insignificant t-statistic (say, one less than 2 in magnitude) fails to reject the null hypothesis that the corresponding level shifts offset each other. Two successive level shifts cancel if the sum of the two corresponding regression parameters is zero. In this case, the two level shifts that cancel can be re-expressed as a temporary level shift starting at the time point of the first level shift and ending one time point before the second level shift. Similarly, three successive level shifts cancel if the sum of their three regression parameters is zero; these can be re-expressed as two adjacent temporary level shifts. There is a user-specified limit on the number of successive level shifts in the runs tested.

The tests for cancellation of level shifts are provided primarily to help users assess the impacts of level shifts in a model. If one or more of these t-statistics are insignificant, the user could consider re-specifying the model with the relevant level shift regression variables replaced by appropriate temporary level shift variables.

4.7 Forecasting

For a given regARIMA model with parameters estimated by the `X-13ARIMA-SEATS` program, the **forecast** spec will use the model to compute point forecasts, along with associated forecast standard errors and prediction intervals. The point forecasts are minimum mean squared error (MMSE) linear predictions of future y_ts based on the present and past y_ts assuming that the true model is used. That is, we assume that the regARIMA model form is correct, that the correct regression variables have been included, that no additive outliers or level shifts will occur in the forecast period, that the specified ARIMA orders are correct, and that the parameter values used (typically estimated parameters) are equal to the true values. These are standard assumptions, though

obviously unrealistic in practical applications. What is more realistically hoped is that the regARIMA model will be a close enough approximation to the true, unknown model for the results to be approximately valid. Two sets of forecast standard errors are produced. One assumes that all parameters are known. The other allows for additional forecast error that comes from estimating the regression parameters, while still assuming that the AR and MA parameters are known. For a reasonably long time series, (Box and Jenkins 1976, pp. 267–269) observe that the contribution to forecast error of the error in estimating the AR and MA parameters is generally small, thus providing a justification for ignoring this source of error when computing the forecast standard errors.

If the series has been transformed, then forecasting results are first obtained in the transformed scale, and then transformed back to the original scale. For example, if one specifies a model of form (4.3) for $y_t = \log(Y_t)$, where Y_t is the original time series, then y_t is forecasted first, and the resulting point forecasts and prediction interval limits are exponentiated to produce point and interval forecasts in the original (Y_t) scale. The resulting point forecasts are MMSE for $y_t = \log(Y_t)$, but not for Y_t under the "standard" assumptions mentioned above. Analogous procedures are followed for other allowable transformations. If any prior adjustments are made, these will also be inverted in the process of transforming the point forecasts and prediction interval limits back to the original scale.

Forecasting requires values for all regression variables through the forecast period. For the `X-13ARIMA-SEATS` predefined regression variables, the program will generate the future values required. For user-defined regressors, the program requires that the user supply the required future values appended to the values supplied for the time span of the observed series (via either the `data` or `file` arguments of the **regression** spec). The analogous requirement applies to backcasting, to which the following discussion also applies.

This requirement presents a problem for the situation where forecasting is being done only for forecast extension in seasonal adjustment (via `X-11` or `SEATS`) and the model includes user-defined regressors whose future values are unknown. If these future values were known and were supplied, for seasonal adjustment the program would (*i*) fit the model to the observed series, (*ii*) remove the estimated regression effects from the observed series, (*iii*) forecast extend the resulting residual series, (*iv*) perform seasonal adjustment on this residual series, and then (*v*) assign regression effects to the appropriate components (seasonal, trend, or irregular). Future values of any regression variables are not used in these calculations, and so are not really needed.

The workaround to address the requirement is to append arbitrary values to the end of any user-defined regressors with unknown future values to cover the forecast extension period, whether the values for these regressors are read from the `data` argument of the **regression** spec or from a file. Since these values will not be used in the seasonal adjustment calculations for the span over which the adjustment is performed, the seasonal adjustment results within this span are not affected by the arbitrariness of the appended values. The appended values will, however, affect the calculation of forecasts of the observed series and of the components, so that setting future values of the user-defined regressors to arbitrary values means that forecasts given in the `X-13ARIMA-SEATS` output will be meaningless. For this reason, it may make sense to set the arbitrary appended values to numbers that deviate greatly from reasonable values of the user-defined regressors, so it will be obvious that the forecasts produced are meaningless and should be disregarded.

If the user supplies actual forecasts of the user-defined regressors, rather than arbitrary values, then point forecast results in the `X-13ARIMA-SEATS` output will be meaningful, with their accuracy depending on the quality of the forecasts supplied for the user-defined regressors. Forecast uncertainty measures given in the program output will, however, understate the true forecast uncertainty, because the calculations will not account for the forecast error in the user-defined regressors.

5 Points Related to regARIMA Model Estimation

Contents

Tables

While the IGLS algorithm and nonlinear least squares routine used by the `X-13ARIMA-SEATS` program are quite reliable at finding maximum likelihood estimates for regARIMA models, problems in estimation occasionally do occur. Some problems that can arise in model estimation are discussed below, along with possible solutions. This is followed by important cautions regarding the use of the model selection criteria produced by the `X-13ARIMA-SEATS` program.

5.1 Initial values for parameters and dealing with convergence problems

Users may supply initial values for AR and MA parameters that are then used to start the iterative likelihood maximization. This is rarely necessary, however, and is not generally recommended. The default choice of initial parameter values in `X-13ARIMA-SEATS` is 0.1 for all AR and MA parameters. (Initial values are not needed for the regression parameters, which are determined in the GLS regressions.) This default choice of initial values appears to be adequate in the great majority of cases. Supplying better initial values (as might be obtained, e.g., by first fitting the model using conditional likelihood) does not seem to speed up convergence enough to make obtaining the initial estimates generally worth the effort. A possible exception to this occurs if initial estimates that are likely to be extremely accurate are already available, such as when one is re-estimating a model with a small amount of new data added to a time series. However, the main reason for specifying initial parameter values is to deal with convergence problems that may arise in difficult estimation situations.

When X-13ARIMA-SEATS' iterative estimation scheme fails to converge, several remedies are available. If the program stopped short of convergence because it reached the maximum number of iterations (indicated by a warning message to this effect and the printing of parameter values at the last iteration), then rerunning the program with initial parameter values set at the values obtained at the last iteration may produce convergence. An easier, though computationally slower, alternative is to simply increase the number of iterations allowed and rerun the program. If the program crashed before converging or reaching the maximum number of iterations, then it may help to first fit the model by conditional likelihood, and use the resulting parameter estimates as initial values for exact maximum likelihood estimation. On the other hand, it has been our experience that convergence problems are often due to the use of a model that is complicated (e.g., high order), or poorly conditioned. In such cases, the appropriate action is to examine the results and specify a simpler model. Sections 5.2 through 5.4 discuss some particular situations that can lead to estimation problems and that suggest specific model modifications.

5.2 Invertibility (of MA operators)

An MA polynomial, $\theta(B) = 1 - \theta_1 B - \cdots - \theta_q B^q$, is invertible if all the roots, $G_1, \ldots, G_q$, of $\theta(B) = 0$ lie outside the unit circle ($|G_j| > 1$ for all j). As shown in Brockwell and Davis (1991), pp. 123–125, for any invertible MA operator in an ARIMA model there are one or more corresponding noninvertible MA operators that produce the same autocovariance structure, and hence the same unconditional likelihood function. Although the data thus cannot discriminate between the invertible and corresponding noninvertible models, the preferred choice is the invertible model. This is essential for forecasting — grossly incorrect forecasting results can be obtained with noninvertible models. There is one important exception. MA polynomials with roots on the unit circle ($|G_j| = 1$), the boundary of the invertibility region, do not cause problems for forecasting when handled appropriately (by exact maximum likelihood for MA models).

Estimation in X-13ARIMA-SEATS enforces invertibility constraints on the MA parameters in the iterative nonlinear maximization of the likelihood function. Strictly speaking, then, models estimated by X-13ARIMA-SEATS are invertible. If the maximum likelihood estimates (MLEs) for a given model are actually on the boundary of the invertibility region, i.e., the model at the MLEs contains an MA operator with zeros exactly on the unit circle, then X-13ARIMA-SEATS' nonlinear search will approach the boundary of the invertibility region from within, and will generally get as close to the boundary as the convergence tolerance dictates or the maximum number of iterations allows. X-13ARIMA-SEATS can thus effectively produce estimated models on the boundary of the invertibility region. Convergence of the estimation iterations in such cases can be slow, since finding the maximum of the likelihood function on the boundary of the constrained parameter space is a difficult optimization problem. More importantly, convergence of the estimation to the invertibility boundary often indicates that the model is poorly conditioned and should alert users to examine the results (and possibly detailed output of the estimation iterations) for signs of this. Section 5.4 discusses the most important causes of poor conditioning – cancellation of factors and overdifferencing – and the appropriate remedies.

Estimation seems most likely to produce a noninvertible model when the model contains a seasonal difference and a seasonal MA polynomial, e.g., $1 - \Theta B^s$ when the MLE of Θ is 1. As such models are commonly used for seasonal economic time series, users should be alert to this possibility and be aware of the appropriate action to take as discussed in Section 5.4.

5.3 Stationarity (of AR operators)

An AR polynomial, $\phi(B) = 1 - \phi_1 B - \cdots - \phi_p B^p$, is stationary if all roots of $\phi(B) = 0$ lie outside the unit circle; otherwise, it is nonstationary. (More accurately, the series $w_t = (1-B)^d(1-B^s)^D z_t$ following the model $\phi(B)\Phi(B^s)w_t = \theta(B)\Theta(B^s)a_t$ (derived from equation (4.1)) is stationary if and only if the zeros of all the AR polynomials lie outside the unit circle.) The exact (for AR) likelihood function assumes all AR operators are stationary. Hence, the exact (for AR) likelihood can be evaluated, and estimation and other analysis (e.g., forecasting) performed, only if the AR parameters satisfy stationarity constraints. Thus, when the exact (for AR) likelihood function is used, `X-13ARIMA-SEATS` enforces stationarity constraints on the estimation. Unless cancellation of factors is present (see the next section), it is unlikely for `X-13ARIMA-SEATS`' nonlinear estimation to approach the boundary of the stationary region, since the log-likelihood approaches $-\infty$ as this boundary is approached.

If the likelihood is defined conditionally with respect to the AR parameters, stationarity is neither assumed nor enforced by the `X-13ARIMA-SEATS` software. Model estimation, forecasting, etc., are not compromised by parameter values outside the stationary region in this case. Inference results, however, are affected, as noted in Section 4.5. Special techniques (as in Fuller 1976, Section 8.5) are required for inference about AR parameters outside the stationary region.

5.4 Cancellation (of AR and MA factors) and overdifferencing

Cancellation of AR and MA factors is possible when a model with a mixed ARMA structure is estimated. A model as in (4.1) or (4.3) is said to have a mixed ARMA structure either if $p > 0$ and $q > 0$ or if $P > 0$ and $Q > 0$. (Technically, a model with $p > 0$ and $Q > 0$, or with $P > 0$ and $q > 0$, is also mixed, but such mixed models are unlikely to lead to cancellation problems.) The simplest example of cancellation occurs with the ARMA(1,1) model, $(1 - \phi B)z_t = (1 - \theta B)a_t$, when $\phi = \theta$. Cancelling the $(1 - \phi B)$ factor on both sides of the model $(1 - \phi B)z_t = (1 - \phi B)a_t$ leaves the simplified model, $z_t = a_t$. Because of this, the likelihood function will be nearly constant along the line $\phi = \theta$. This can lead to difficulties with convergence of the nonlinear estimation if the MLEs for the ARMA(1,1) model approximately satisfy $\widehat{\phi} = \widehat{\theta}$. Analogous problems occur in more complicated mixed models when an AR polynomial and an MA polynomial have a common zero (e.g., the ARIMA (2,1,2)(0,1,1) model that is used as a candidate model for the **automdl** spec). For a fuller discussion of this topic, see Box and Jenkins (1976), pp. 248–250.

If the `X-13ARIMA-SEATS` program has difficulty in converging when estimating a mixed model, cancellation of AR and MA factors may be responsible. In any case, possible cancellation can be checked by computing zeros of the AR and MA polynomials (setting `print = roots` in the **estimate** spec) and examining these for zeros common to an AR and an MA polynomial. If a common zero (or zeros) is found, then the model should be simplified by cancelling the common factor(s) (reducing the order of the corresponding AR and MA polynomials), and the model should be re-estimated. Cancellation need not be exact, but may be indicated by zeros of an AR and an MA polynomial that are approximately the same.

It is also possible for estimated MA polynomials to have factors that cancel with differencing operators. This occurs when a model has a nonseasonal difference and an estimated nonseasonal MA polynomial contains a $(1-B)$ factor or when the model has a seasonal difference and an estimated seasonal MA polynomial contains a $(1 - B^s)$ factor. For example, the model $(1 - B)(1 - B^s)z_t = (1 - \theta B)(1 - \Theta B^s)a_t$ involves such cancellation

if either θ or Θ is estimated to be one. Such cancellation is called "overdifferencing," since it implies that the series was differenced more times than necessary to achieve stationarity. When overdifferencing occurs the corresponding difference and MA factor may be canceled to simplify the model, but the user must then also add to the model regression term(s) to account for the deterministic function of time that was previously annihilated by the canceled differencing operator. This means that if a nonseasonal difference is canceled with a $(1 - \theta B)$ MA factor with $\widehat{\theta} = 1$, then the simplified model should include a trend constant (or overall mean, if the model had only this one difference). If a seasonal difference is canceled with a $(1 - \Theta B^s)$ seasonal MA factor with $\widehat{\Theta} = 1$, then the simplified model should include both a trend constant (or overall mean) and fixed seasonal effects. Overdifferencing is discussed by Abraham and Box (1978) and Bell (1987).

If estimation converges to an overdifferenced model, modifying the model by removing the differencing operator and MA factor that cancel as well as including the appropriate regression terms, and then re-estimating the model, is somewhat optional, because this cancellation does not necessarily lead to problems with model estimation and other results (assuming use of the likelihood function that is exact for the MA parameters). In particular, forecasting results should be the same for both the overdifferenced model and the corresponding modified model, and regression and ARMA parameter estimates and standard errors under the two models should be approximately the same. (However, log-likelihood values and the corresponding model selection criteria will be different for the two models – see the next section.) This contrasts with the situation regarding cancellation of AR and MA factors. Since cancellation of AR and MA factors is more likely to lead to convergence problems in estimation, common AR and MA factors should always be removed from the model, and the model re-estimated.

5.5 Use of model selection criteria

The `X-13ARIMA-SEATS` program provides the following model selection criteria: AIC (Akaike 1973, see also Findley 1985, 1999, and Findley and Wei 2002), AICC (Hurvich and Tsai 1989), a criterion due to Hannan and Quinn (1979), and BIC (Schwarz 1978). Suppose the number of estimated parameters in the model, including the white noise variance, is n_p. If after applying the model's differencing and seasonal differencing operations, there are N data, and if the estimated maximum value of the exact log likelihood function of the model for the untransformed data is denoted L_N, then the formulas for these criteria are:

$$\begin{aligned} AIC_N &= -2L_N + 2n_p \\ AICC_N &= -2L_N + 2n_p \left(1 - \frac{n_p + 1}{N}\right)^{-1} \\ HannanQuinn_N &= -2L_N + 2n_p \log\log N \\ BIC_N &= -2L_N + n_p \log N. \end{aligned}$$

If a function f of the **transform** spec is applied before regARIMA model estimation, then the maximized log likelihood L_N of the untransformed data Y_t in the formulas above is obtained as follows. Given the regARIMA model's differencing operator as $(1 - B)^d(1 - B^s)^D$, let the transformed data $y_t = f(Y_t)$ used for modeling be $y_{-(d+sD)+1}, \ldots, y_0, y_1, \ldots, y_N$. The number N is called the *effective number of observations*. Let

L_N^y denote the regARIMA model's maximized log likelihood for $y_1, \ldots, y_N$ conditional on $y_{-(d+sD)+1}, \ldots, y_0$, which is calculated as the maximized log likelihood of the regARIMA model for $(1-B)^d(1-B^s)^D y_t$, $1 \le t \le N$. Then

$$L_N = L_N^y + \sum_{t=1}^{N} \log \left| \frac{df\left(Y_t\right)}{dY_t} \right|.$$

The second term on the right is called the *transformation adjustment*. (It is the Jacobian of the data transformation $Y_t = f^{-1}\left(y_t\right)$, $1 \le t \le N$; see Chapter 6 of Mood, Graybill, and Boes, 1974.) In the case of the (natural) log transformation $f\left(Y_t\right) = \log Y_t$, for example, it is $-\sum_{t=1}^{N} \log Y_t$. Defining the model selection criteria in terms of the untransformed data Y_t makes it possible to compare competing transformations for this data, for example the log transformation and no transformation, see Section 7.18.

Akaike's Minimum AIC criterion (MAIC) states that, between any two models, the one with the smaller AIC is preferred; see Akaike (1973) and Findley (1999) for example. Similarly, for each of the other model selection criteria above, the model with the smaller value is preferred. This property is determined by the sign of the difference of the criterion values. Focusing on AIC, given two models, designated model 1 and model 2, with log maximum likelihood values and numbers of estimated parameters denoted by $L_N^{(1)}$ and $L_N^{(2)}$ and $n_p^{(1)}$ and $n_p^{(2)}$, respectively, we consider the AIC difference

$$AIC_N^{(1)} - AIC_N^{(2)} = -2\left\{L_N^{(1)} - L_N^{(2)}\right\} - 2\left(n_p^{(2)} - n_p^{(1)}\right). \tag{5.1}$$

When model 1 is of the correct type and is a special case of (is "nested in") model 2, then for long enough time series, the AIC difference (5.1) varies approximately like a chi-square variate with $n_p^{(2)} - n_p^{(1)}$ degrees of freedom, i.e. asymptotically

$$-2\left\{L_N^{(1)} - L_N^{(2)}\right\} \sim \chi^2_{n_p^{(2)} - n_p^{(1)}}. \tag{5.2}$$

This holds under standard assumptions, including the requirement that the true model is invertible, i.e. without unit magnitude roots in the MA polynomial (see Taniguchi and Kakizawa 2000, p. 61). The same result applies to AICC differences because $\left(n_p^{(1)} + 1\right)/N$ and $\left(n_p^{(2)} + 1\right)/N$ tend to zero as N increases.

Under (5.2), the asymptotic probability that model 2 will have a smaller AIC and thus incorrectly be preferred over model 1 by the MAIC criterion is, from (5.2),

$$P\left(AIC_N^{(1)} - AIC_N^{(2)} > 0\right) = P\left(\chi^2_{n_p^{(2)} - n_p^{(1)}} > 2(n_p^{(2)} - n_p^{(1)})\right). \tag{5.3}$$

Thus, the right hand side of (5.3) gives the asymptotic probability of a Type I error (rejecting model 1 in favor of model 2) by the Minimum AIC criterion. Some relevant values when the **aictest** argument of the **regression** spec (see Section 7.13) is used with trading day and holiday effect regression models of Table 4.1 are given in Table 5.1.

ν	$P(\chi^2_\nu > 2\nu)$	$P(\chi^2_\nu > 2\nu+1)$	$P(\chi^2_\nu > 2\nu+2)$	$P(\chi^2_\nu > 2\nu+3)$
1	.157	.083	.046	.025
2	.135	.082	.050	.030
5	.075	.051	.035	.025
6	.062	.043	.030	.020
7	.051	.036	.025	.017
∞	0	0	0	0

Table 5.1: **Probability that a Chi-Square Variate with ν Degrees of Freedom Exceeds $2\nu+\Delta_{AIC}$ for $\Delta_{AIC} = 0, 1, 2, 3$.**

Table 5.1 shows the effect on the asymptotic Type I error probability of using certain values Δ_{AIC} of the **aicdiff** argument in conjunction with the **aictest** argument to bias the decision toward the model without the regression effect tested. (The default is `aicdiff = 0`.) The degrees of freedom values ν for which probabilities $P(\chi^2_\nu > 2\nu + \Delta_{AIC})$ are given apply to certain trading day models defined in Table 4.1, e.g., `tdnolpyear` and `td1nolpyear`, with and without `lpyear`. However, with `td` and `td1coef` in the multiplicative adjustment case, when fixed leap year ratio preadjustment factors are used with `tdnolpyear` and `td1nolpyear` regressors, instead of estimating a coefficient of LY_t, the model with no trading day effects is not nested in the `td` and `td1coef` models (see the DETAILS section of Section 7.13). In these two cases, the use of **aictest** can be shown to have an asymptotic probability of a Type I error equal to zero, because the incorrect use of the fixed leap year ratio preadjustment factors cause the models with them to be asymptotically worse than the model with no trading day effects, so the discussion below leading to (5.4) applies.

When `aictest = easter` is used, the Type I error probabilities are slightly higher than those given in Table 5.1 because, instead of a single model, three different models, with `easter[1]`, `easter[8]` and `easter[15]` regressors, respectively, are being compared to a model with no Easter regressor.

Type I error probabilities may provide some helpful insights into properties of MAIC, but it must be kept in mind that they arise from a different modeling paradigm. The minimum AIC criterion is based on a deep approximation property rather than on conventional significance tests: under assumptions that encompass those used to calculate Type I error probabilities, an AIC difference is an asymptotically unbiased estimate of the difference between the Kullback-Leibler quasi-distances from the true model to the estimated models; see Akaike (1973), Findley (1999) and Findley and Wei (2002) for example. The Minimum AIC criterion seeks to indicate which model is closer to the truth in this sense. This property can justify the use of MAIC for some nonnested model comparisons where likelihood ratio tests based on a χ^2 distribution do not exist.

Also, regardless of whether the models are nested or nonnested, if model 2 is asymptotically worse than model 1 (specifically, farther from the true model in the Kullback-Leibler sense), then it can be shown that

$$\lim_{N\to\infty} \frac{1}{N}\left\{AIC_N^{(1)} - AIC_N^{(2)}\right\} = \lim_{N\to\infty} \frac{2}{N}\left\{L_N^{(2)} - L_N^{(1)}\right\} = C_{1,2} < 0, \tag{5.4}$$

(in probability) with the result that $AIC_N^{(1)} - AIC_N^{(2)}$ tends to $-\infty$ effectively linearly in N. Hence MAIC will strongly prefer model 1 for large enough N. The same result holds for $AICC_N^{(1)} - AICC_N^{(2)}$ (and also for the other criteria above). This property further helps to explain why AIC and AICC have often been found to be effective with nonnested model comparisons. For such comparisons, $\left|AICC_N^{(1)} - AICC_N^{(2)}\right|$ is often rather large

(e.g., greater than three), with series of average lengths unless the models being compared are quite close for the modeled series (as can happen with the `easter[w]` regressors of Table 4.1).

In situations in which multiple models are compared (more than two or three, perhaps substantially more), it is worthwhile to consider the model with the second smallest $AICC$ value as well as the minimum $AICC$ model, and perhaps other models whose $AICC$ value is close to the minimum value, especially when the model comparisons are nonnested. These alternative model sometimes have more desirable features, e.g. several fewer parameters, better interpretability, greater consistency with the model chosen for several closely related series, etc. Burnham and Anderson (2004) on page 271 offers rough rules of thumb for this situation, that we formulate with $AICC$ instead of AIC. With $AICC_{\min}$ denoting the minimum AICC value and $AICC_{\text{alt}}$ denoting the second smallest $AICC$ value or the $AICC$ value of some similarly competitive alternative model, set $\Delta = AICC_{\text{alt}} - AICC_{\min}$. If $\Delta \leq 2$, there is substantial support for the alternative model, considerably less if $4 \leq \Delta \leq 7$, and essentially no support if $\Delta > 10$.

We now turn to some situations that require special consideration.

5.5.1 Comparing models with different sets of outlier regressors

Critical values near 4.0 or larger are usually used to select outlier regressors with a given ARIMA model, see Table 7.22. (This is done to compensate for level of significance distortions and loss of power resulting from the large number of tests done by the automatic procedure of the **outlier** spec.) Outliers that enter the model with large critical values usually cause the maximum log likelihood to increase quite substantially and AIC and the other criteria to decrease correspondingly.

As a consequence, unless the models being compared have the same outliers (which often have similar effects on both log likelihoods and therefore have effects that almost cancel in differences of criterion values), the outliers can largely determine the model selection, rather than more relevant data properties. In particular, the model with the most outliers will often be the one with the smallest criterion value. Therefore, with automatic model selection using the model selection criteria, when the outlier sets are not automatically the same, it can be important to find out if differences in outlier sets have determined the outcome. This can be done by changing the specifications of the most competitive models so that all these models have same outlier regressors and then estimating the modified models and comparing their model selection criteria.

5.5.2 Comparing models with different transformations of data

Often a log transformation, or some other Box-Cox power transformation,

$$\begin{array}{ll} \lambda^2 + \left\{Y_t^{\lambda} - 1\right\}/\lambda, & \lambda \neq 0 \\ \log Y_t, & \lambda = 0 \end{array},$$

is applied to the original data Y_t prior to regARIMA modeling – see Section 7.18. (Note that the power $\lambda = 1$ yields Y_t, i.e., no transformation.) Frequently this transformation is preceded by division of the series Y_t by positive prior ratio-adjustment factors c_t.

For monthly data, an important example is the leap year preadjustment factor defined by

$$c_t = \begin{cases} \frac{28}{28.25}, & \text{28-day months} \\ \frac{29}{28.25}, & \text{29-day months} \\ 1, & \text{other months} \end{cases}, \tag{5.5}$$

(see Section 7.13). When both kinds of transformations are used, then

$$y_t = \begin{cases} \log Y_t - \log c_t, & \lambda = 0 \\ \lambda^2 + \left\{ (Y_t/c_t)^\lambda - 1 \right\} / \lambda, & \lambda \neq 0 \end{cases} \tag{5.6}$$

is the series for which a regARIMA model is sought.

With $\delta(B) = (1-B)^d(1-B^s)^D$, suppose the observed series is indexed as Y_t, $-(d+sD)+1 \leq t \leq N$, so the transformed series to which regARIMA model (4.3) is fit is y_t, $-(d+sD)+1 \leq t \leq N$. Thus $z_t = \delta(B)y_t$, $1 \leq t \leq N$, is the series from which regression and ARMA parameters are estimated by maximizing the regARMA model's Gaussian-form log likelihood function $L_N^z(\beta, \phi, \Phi, \theta, \Theta, \sigma_a^2; z_1, \ldots, z_N)$. Denoting its maximum value by L_N^y as above, a log likelihood for the untransformed data $Y_1, \ldots, Y_N$ (conditional[13] on the initial observations Y_t, $-(d+sD)+1 \leq t \leq 0$) is obtained by adding to the log likelihood the log Jacobian *transformation adjustment*[14] $\sum_{t=1}^N \log |dy_t/dY_t|$. This yields the maximum log likelihood value

$$L_N = L_N^y + \sum_{t=1}^N \log \left| \frac{dy_t}{dY_t} \right|$$

for the model for Y_t, whose definition now includes any data transformation (and/or preadjustments) as well as the regARIMA specification. N is called the effective number of observations.

For example, for y_t given by (5.6), the transformation adjustment is

$$\sum_{t=1}^N \log \left\{ c_t^{-1} \left(\frac{Y_t}{c_t} \right)^{\lambda - 1} \right\},$$

which reduces to $\sum_{t=1}^N \log Y_t^{-1} = -\sum_{t=1}^N \log Y_t$ when $\lambda = 0$. For the logistic transformation of data Y_t preadjusted so that $0 < Y_t/c_t < 1$ always holds, we have

$$y_t = \log \frac{Y_t/c_t}{1 - Y_t/c_t} = \log \frac{Y_t}{c_t - Y_t},$$

and the transformation adjustment is $-\sum_{t=1}^N \log \left\{ c_t^{-1}(c_t Y_t - Y_t^2) \right\}$.

[13] To provide this interpretation and other properties desirable for signal extraction, the initial $d+sD$ values of the series Y_t are assumed to be statistically independent of the $\delta(B)y_t$ – see Bell (1984) and Bell and Hillmer (1988). This is the only statistical assumption made for these initial variates.

[14] Because $\delta(B)y_t$ is a function of Y_s, $s \leq t$, the Jacobian matrix $[\partial z_t/\partial Y_s]_{1 \leq s,t \leq N}$ is a triangular matrix. Consequently, $\det [\partial z_t/\partial Y_s]_{1 \leq s,t \leq N} = \prod_{t=1}^N \partial z_t/\partial Y_t = \prod_{t=1}^N dy_t/dY_t$.

To compare different ratio preadjustments and/or different transformations (and perhaps different regression and ARMA specifications at the same time), we replace L_N^y by L_N in the criterion function formulas for AIC, AICC, Hannan-Quinn and BIC above, e.g. $AICC_N = -2L_N + 2n_p \left(1 - \frac{n_p+1}{N}\right)^{-1}$.

5.5.3 Comparing models with different differencing operators

The preceding discussion shows that a model with differencing operator $\delta^*(B) = (1-B)^{d^*}(1-B^s)^{D^*}$, such that $d^* + sD^* \neq d + sD$, yields a log likelihood function that is for a set of Y_t values different from $Y_1, \ldots Y_N$ (and that is conditional on a different set of initial values). Therefore its log likelihood function, and hence also the value of any of the model selection criteria, is not comparable[15] to the values of the same criterion obtained with the differencing operator $\delta(B) = (1-B)^d(1-B^s)^D$. To compare models with different differencing operators, the out-of-sample forecast error output of the **history** spec (Section 7.8) can be used, with the graphical diagnostics discussed in Sections 3 and 4 of Findley, Monsell, Bell, Otto, and Chen (1998) when the series is long enough that regARIMA models can be estimated reliably without the final two years of data, which are withheld for forecasting.

[15] Ozaki (1977) proposed a rescaling of AIC_N to compare different orders of differencings of nonseasonal ARIMA models. In the seasonal case, the analogue would be to multiply AIC_N by $(N + d + sD)/N$ and use the resulting value in all model comparisons. There is neither theoretical nor systematic empirical support for such a rescaling of any of the criteria, so rescaling this way is not an accepted practice for model selection.

6 Points Related to Seasonal Adjustment and Modeling Diagnostics

Contents

Tables

The `X-13ARIMA-SEATS` seasonal adjustment program contains several new diagnostics for modeling, model selection, adjustment stability, and for judging the quality of indirect as well as direct seasonal adjustments. This chapter deals specifically with three diagnostics that can be generated by the `X-13ARIMA-SEATS` program.

- Section 6.1 describes the spectral plots that `X-13ARIMA-SEATS` produces of the original series, the reg-ARIMA residuals, the final seasonal adjustment and the final irregular component. The plots are marked at frequencies commonly associated with seasonal and trading day variation, so the user can easily check for residual effects in the model residuals or seasonal adjustment. For more information, see Section 2.1 of Findley, Monsell, Bell, Otto, and Chen (1998) and Soukup and Findley (1999).
- Section 6.2 describes the sliding spans diagnostics, which compare seasonal adjustments from overlapping spans of a given time series. This provides an indication of the stability of the seasonal adjustment.
- Section 6.3 describes revisions history diagnostics, another stability diagnostic. The basic revision is the difference between the initial seasonal adjustment (often referred to as the *concurrent* adjustment) and the seasonal adjustment with all the data available at the time of the analysis (often referred to as the *final* adjustment).

6.1 Spectral plots

`X-13ARIMA-SEATS` provides spectral plots and associated interpretative messages to alert the user to the presence of seasonal and trading day effects. Spectral output is available for the original series and as many as three

series resulting from modeling or seasonal adjustment, namely the model residuals, when modeling is specified, and the adjusted series and irregulars series, when adjustment for seasonal (and possibly also trading day or holiday) effects is specified.

Currently, these plots are provided only for monthly series; a diagnostic to detect residual seasonality in time series of other periodicity, including quarterly, is the QS diagnostic from the TRAMO and SEATS programs. This diagnostic's output options can be specified in the **spectrum** spec; more information about it is given in the DETAILS of Section 7.17.

6.1.1 General information

For a stationary time series x_t with mean μ and autocovariances $\gamma_k = E(x_t - \mu)(x_{t+k} - \mu)$, $k = 0, \pm 1, \ldots$, the spectral density (spectrum for short) is a nonnegative function $g(\lambda)$, $0 \leq \lambda \leq 1/2$, which reformulates the content of the autocovariances in terms of amplitudes at frequencies of half a cycle per sampling period (month for our purposes) or less, in such a way that

$$\gamma_k = 2 \int_0^{1/2} g(\lambda) \cos(2\pi k \lambda) d\lambda, \; k = 0, \pm 1, \ldots \; .$$

When x_t is a stationary ARMA process with the backshift operator polynomial formula $\phi(B)(x_t - \mu) = \theta(B)a_t$, then its spectrum can be shown to be given by

$$g(\lambda) = \sigma^2 \frac{\left|\theta\left(e^{i2\pi\lambda}\right)\right|^2}{\left|\phi\left(e^{i2\pi\lambda}\right)\right|^2},$$

where σ^2 is the variance of the white noise a_t; see Priestley (1981). Here $e^{i2\pi\lambda} = \cos 2\pi\lambda + i \sin 2\pi\lambda$, and for a complex number $u + iv$ (u, v real), $|u + iv| = \sqrt{u^2 + v^2}$.

For the first-differenced original series of the **series** or **composite** spec (transformed in accord with the **transform** spec), the program's warning message about "visually significant" seasonal peaks, or the associated plot, can alert the user to the possibility that the series has a seasonal effect that is predictable (stable) enough for `X-13ARIMA-SEATS` to estimate with reasonable success. (If there are seasonal peaks in the spectrum but none that meet the criteria for visual significance, see below, then it is likely that any "seasonal" effects in the series change too rapidly from year to year or are too obscured by "noise" to be estimated reliably or stably.)

For the regARIMA model residuals (when a regARIMA model is estimated), and for the first-differenced, transformed seasonally adjusted series and the irregulars series (when the `x11` or `seats` spec is used), the messages indicate that the model or adjustment procedure for seasonal or trading day effects has either failed to capture such effects or, worse, has induced such effects in the series over the time interval used for spectrum estimation. Because seasonal and trading day patterns can change over time, and because adequate modeling or adjustment is usually most important for recent data, the time interval of the most recent 96 observations is the default interval for spectrum estimation (or the time interval specified by the applicable **modelspan** or **span** argument when the latter interval has length less than 96). In the case of trading day peaks, a peak (especially one at the lower of the two trading day frequencies) shows the need for trading day estimation if this was not done, and otherwise shows that the trading day regression model used is inadequate for the time interval used for spectrum estimation.

At seasonal frequencies, a peak in the model residuals indicates the need for a better fitting model for the time interval used for spectrum estimation. A peak in the spectrum from the seasonally adjusted series or irregulars reveals inadequacy of the seasonal adjustment filters for this interval, thereby indicating that different filters and/or a shorter data span should be considered. Usually, the spectrum estimator requires 72 data points to produce peaks sharply defined enough to trigger warning messages for seasonal or trading day effects.

6.1.2 AR spectrum

The default spectrum estimator used to detect seasonal and trading day effects is an autoregressive spectral estimator. For the series x_t (for example, the model residuals) whose spectrum is being estimated from data $x_1, \ldots, x_N$, autoregressive log-spectrum estimates (in decibel units) have the form

$$\widehat{s}\,(\lambda) = 10 \log_{10} \left\{ \frac{\widehat{\sigma}_m^2}{2\pi \left| 1 - \sum_{j=1}^{m} \widehat{\phi}_j e^{i2\pi j\lambda} \right|^2} \right\}, \; 0 \le \lambda \le 0.5, \tag{6.1}$$

where the coefficient estimates $\widehat{\phi}_j$ are those of the linear regression of $x_t - \bar{x}$ on $x_{t-j} - \bar{x}$, $1 \le j \le m$ for the data, with $\bar{x} = N^{-1} \sum_{t=1}^{N} x_t$, and where $\widehat{\sigma}_m^2$ is the sample variance of the resulting regression residuals. For large enough m (and N), a strong component with period $1/\lambda_0$ results in a near-zero value of the denominator of (6.1) at λ_0 and therefore in a peak at λ_0 in the graph of (6.1) – unless there is a stronger periodic component at a nearby frequency. For a discussion of this estimator, see pp. 600–612 of Priestley (1981). Application of the proof of Corollary 5.6.3 of Brillinger (1975) to the results of Theorem 6 of Berk (1974) shows that, under Berk's assumptions, which include invertibility of x_t, the log transformation in (6.1) stabilizes the large sample variance of $\widehat{s}(\lambda)$, i.e. the limiting value of $E\left(\widehat{s}(\lambda) - g(\lambda)\right)^2$, as $m \to \infty$ in case $g(\lambda)$ is not an AR process. However, the constant value for the end point frequencies $\lambda = 0, 1/2$ is twice as large as the constant value for the intermediate frequencies ($0 < \lambda < 1/2$).

`X-13ARIMA-SEATS` uses $m = 30$ for monthly series, which yields high resolution of strong components, meaning peaks that are sharply defined in the main output file's plot of $\widehat{s}(\lambda)$ (Recall that for the spectra providing information about the original series and the seasonal adjustment, the series x_t results from suppression of a trend component by differencing or detrending. Trends produce peaks at and near $\lambda = 0$ that are so dominant that they diminish the resolution of all other peaks.) The spectrum plots of `X-13ARIMA-SEATS` show values of $\widehat{s}(\lambda)$ at 61 frequencies that have the form $\lambda_k = k/120$, $0 \le k \le 60$, with two exceptions: for the values $k/120$ closest to the trading day frequencies (0.348, and 0.432 cycles per month for monthly series), λ_k is assigned the value of the trading day frequency instead of the value $k/120$. At trading day frequencies, values of $\widehat{s}(\lambda_k)$ are plotted with a column of T's. At seasonal frequencies (1/12, 2/12,. . . , 6/12 cycles per month for monthly series) values of $\widehat{s}(\lambda_k)$ are plotted with a column of S's. At all other frequencies, columns of asterisks ("stars") are used. These plots are very similar to those of the BAYSEA seasonal adjustment program (Akaike 1980 and Akaike and Ishiguro 1980) and are produced by a modified version of BAYSEA's Fortran code.

The monthly trading day frequency 0.348 can be derived by noting that a daily component which repeats every seven days goes through $4.348 \doteq 30.4375/7$ cycles in a month of average length (365.25/12 = 30.4375 days). It is therefore seen to advance 0.348 cycles per month when the data are obtained at twelve equally spaced times in a period of 365.25 days, the average length of a year. The connection of peaks at 0.432 cycles per month with trading day components is weaker – see Cleveland and Devlin (1980) – and not as reliable.

Note that the time series needs to have at least 60 observations for `X-13ARIMA-SEATS` to display the trading day frequencies in the plots and attempt to identify peaks for trading day frequencies.

Because of difficulties associated with tests of statistical significance for periodic components in autocorrelated data, see Chapter 8 of Priestley (1981), such tests are not used for the AR spectrum. The warning messages of `X-13ARIMA-SEATS` are based on an empirically obtained criterion of "visual significance" determined as follows from the range $\widehat{s}^{\max} - \widehat{s}^{\min}$ of the $\widehat{s}(\lambda_k)$ values, where $\widehat{s}^{\max} = \max_k \widehat{s}(\lambda_k)$ and $\widehat{s}^{\min} = \min_k \widehat{s}(\lambda_k)$. To be "visually significant," the value $\widehat{s}(\lambda_k)$ at a trading day or seasonal frequency λ_k (other than the seasonal frequency $\lambda_{60} = 0.5$) must be above the median of the plotted values of $\widehat{s}(\lambda)$ and must be larger than both neighboring values $\widehat{s}(\lambda_{k-1})$ and $\widehat{s}(\lambda_{k+1})$ by at least 6/52 times the range $\widehat{s}^{\max} - \widehat{s}^{\min}$. In the main output file's line printer plots of spectra, $\widehat{s}^{\max}$ is plotted 52 lines above $\widehat{s}^{\min}$, so a visually significant peak must be at least six lines (six "stars") high.

Peaks of any size at $\lambda_{60} = 1/2$ are ignored. The theoretical results from Berk (1974) and Brillinger (1975) mentioned above describe how $\widehat{s}(\lambda)$ will be more randomly variable at $\lambda_{60} = 1/2$ than at other seasonal frequencies. More erratic behavior and less reliable performance are to be expected when the spectral density $g(\lambda)$ of the series being estimated is zero or close to zero at some frequency, especially at $\lambda = 1/2$, which can happen, particularly with the series of irregulars (see Bell 2010). The empirical finding from practice is that visually significant peaks at $\lambda_{60} = 1/2$ occur too often in the spectra of seasonally adjusted and irregular series that have few or no other visually significant seasonal peaks.

Also, visually significant peaks at $\lambda_{50} = 5/12$ rather frequently occur when there are no other visually significant peaks. There is no economic explanation for such peaks.

6.1.3 Tukey spectrum

In addition to the AR spectrum, the program generates a non-parametric Tukey estimate of the spectrum. This estimate appears in the TRAMO-SEATS and TSW software and is part of that software's seasonality tests – see Maravall (2012) and Jenkins and Watts (1968).

For a series x_t with length N (e.g., the differenced SA series or the irregulars) and for specified $M < N$, set

$$\widehat{g}_x^T(\lambda) = c_0 + 2\sum_{k=1}^{M} w(k/M) c_k \cos(2\pi k\lambda),\ 0 \le \lambda \le 1/2$$

with

$$c_k = N^{-1} \sum_{t=1}^{N-k} (x_t - \bar{x})\,(x_{t-k} - \bar{x})\,,$$

where

$$\bar{x} = N^{-1} \sum_{t=1}^{N} x_t$$

and with

$$w(v) = \frac{1}{2} + \frac{1}{2}\cos(\pi v),\ 0 \le v \le 1.$$

For series that are monthly		
	tukey120=yes	**tukey120=no**
$N \geq 120$	120	112
$80 \leq N < 120$	79	79
For series that are not monthly		
	tukey120=yes	**tukey120=no**
$N \geq 60$	44	44

Table 6.1: **Values of M used in Tukey Spectrum Calcuations**

The M parameter is determined by the length and the seasonal frequency of the time series, as well as the value of the **tukey120** argument in the **spectrum** spec. Table 6.1 shows how the value of M is set internally.

As implied in Table 6.1, the Tukey spectrum can only be generated for monthly series with more than 79 observations and only for other series that have at least 60 observations. Note that this does not include the differencing done for some of the series (such as the original and seasonally adjusted series).

6.2 Sliding spans diagnostics

The sliding spans diagnostics are described in detail and compared with other quality diagnostics in the articles Findley, Monsell, Shulman, and Pugh (1990) and Findley and Monsell (1986). An abbreviated presentation will be given here. The basic diagnostics are descriptive statistics of how the seasonal adjustments and their month-to-month changes vary when the span of data used to calculate them is altered in a systematic way: any two neighboring spans differ to the extent that one starts and ends a year later than the other. The span length is determined by the length of the seasonal filter utilized for the adjustment. The ending date of the last span is usually the date of the most recent datum in the time series. Four spans are used if enough data are available. The index value $j = 1$ is assigned to the span with the earliest starting date, $j = 2$ to the span with the next earliest starting date, and so on.

For series whose seasonally adjusted values are all positive, the two most important sliding spans statistics, $A(\%)$ and $MM(\%)$, are calculated as follows. For a month t belonging to at least two spans, one of which is the j-th span, let A_t denote its seasonally (and, if applicable, trading day and holiday) adjusted value obtained from the complete series, and let A_t^j denote the adjusted value obtained when the seasonal adjustment procedure being considered (the procedure determined by the software options selected) is applied only to the data in the j-th span. The seasonal adjustment A_t is called (unacceptably) unstable if

$$\frac{\max_j A_t^j - \min_j A_t^j}{\min_j A_t^j} > 0.03. \tag{6.2}$$

Further, for months t such that both t and $t-1$ belong to at least two spans, the "seasonally adjusted month-to-month percent change" $100 \times (A_t - A_{t-1}) / A_{t-1}$ is called unstable if

$$\max_j \frac{A_t^j}{A_{t-1}^j} - \min_j \frac{A_t^j}{A_{t-1}^j} > 0.03. \tag{6.3}$$

In (6.2), the index j ranges over all spans containing month t, whereas in (6.3) the j-th span must contain month $t-1$ as well.

$A\,(\%)$ is used to denote the percent of months with unstable adjustments calculated with respect to the number of months for which the left hand side of (6.2) is defined (the number of months common to at least two spans). The analogous quantity for (6.3) is denoted $MM(\%)$. We recommend that, except in special circumstances of the sort discussed below, the seasonal adjustment produced by the procedure chosen should not be used if $A\,(\%) > 25$ (> 15 is considered problematic) or if $MM(\%) > 40$.

There is a similarly defined statistic $YY(\%)$ for year-to-year percent changes in the seasonally adjusted data, $100 \times (A_t - A_{t-12})\,/A_{t-12}$, based on the same threshold used to define unstable adjustments and month–to-month changes, usually the default 0.03 shown in (6.2) and (6.3). Because these year-to-year changes in the adjusted series can be misleading indicators of trend direction when turning points occur between months t and $t-12$, they are less important than the adjusted values themselves and the month-to-month changes in the adjusted values. Hence, the statistic $YY(\%)$ is less important than the others, but it is included in the output of `X-13ARIMA-SEATS` anyway because of the interest some data users have in year-to-year changes. The output text describes values of $YY(\%)$ greater than 10 as extreme, but this information is usually redundant in the sense that series with such a value have, in our experience, usually also had excessive values of $A(\%)$ or $MM(\%)$. In any case, we would not reject an adjustment based solely on the value of $YY(\%)$.

Sometimes, the causes of large values of $A(\%)$ or $MM(\%)$ can be identified and considered not overly problematic. For example, this could be the case when the months with unstable adjustments or changes are heavily concentrated in a known problem period several years back from the current year, or in one or two fixed calendar months each year that all data users can be expected to regard as quite problematic (such as winter months in series known to be very sensitive to differences in winter weather conditions). The sliding spans output makes it easy to identify such concentrations.

The output can show when a "mild" increase in the threshold beyond 0.03 will dramatically decrease the values of $A(\%)$ and $MM(\%)$ to acceptable levels: we have identified a few series for which increasing the threshold to 0.05 seemed justifiable, because most of the months for which the left hand sides of (6.2) and (6.3) were between 0.03 and 0.05 were months with very large seasonal movements, where users would be tolerant of more uncertainty, and not many months had values of these statistics substantially larger than 0.05.

This experience stimulated us to carry out a limited exploratory study with a variety of Census Bureau series focused on the goal of finding a statistical relationship between appropriate threshold values and seasonal factor size, which we could then use to adjust the threshold according to the size of the seasonal movements. However, within the set of series considered, we found no correlation between appropriate threshold values and the size of the seasonal movements. For example, there were a large number of series with rather sizable seasonal movements for which good values of $A(\%)$ and $MM(\%)$ were obtained with the 0.03 threshold, and there were other series with only moderately large seasonal movements for which the use of the 0.05 threshold did not lead to acceptable values of $A(\%)$ and $MM(\%)$. In fact, simulation experiments readily show that in a series with fixed seasonal effects (every January has the same seasonal factor, etc.), the values of the seasonal adjustment are quite sensitive to the variability of the irregulars component and quite insensitive to the size of the seasonal movements.

More often than not, when a choice of adjustment options for a series produces an adjustment that sliding spans diagnostics classify as unacceptable, there will be a different choice of options, perhaps with different seasonal filter lengths, or different trading day adjustment or forecast extension options, that will result in an adjustment that is classified as acceptable. When no choice of options produces an acceptable adjustment, the

issue is not whether the series is "seasonal" in some sense, but whether its seasonal behavior is repetitive enough, or revealed clearly enough in the available time series data, that it can be estimated with adequate reliability by `X-13ARIMA-SEATS` under any of the options considered.

6.3 Revisions history diagnostics

`X-13ARIMA-SEATS` generates revisions between the initial estimate and the most recent estimate for several quantities derived from seasonally adjusting a time series (see Table 7.17). `X-13ARIMA-SEATS` can also generate historical out-of-sample forecast errors and likelihood statistics derived from regARIMA model estimation. For some supporting theory for out-of-sample squared forecast error diagnostic output, see Findley (2005). These revisions and historical values are obtained as follows.

For a given series y_t where $t = 1, \ldots, T$, we define $A_{t|n}$ to be the seasonal adjustment of y_t calculated from the series $y_1, y_2, \ldots, y_n$, where $t \leq n \leq T$. The concurrent seasonal adjustment of observation t is $A_{t|t}$ and the most recent or "final" adjustment of observation t is $A_{t|T}$. The percent revision of the seasonally adjusted series is defined to be

$$R_t = \frac{A_{t|T} - A_{t|t}}{A_{t|t}},$$

and this is what the program reports. The revisions of the trend component and of seasonal factors derived from multiplicative or log-additive seasonal adjustment are also reported as percent revisions.

With additive seasonal adjustments, R_t is calculated the same way if all values $A_{t|t}$ have the same sign (the analogous statement holds for trends). Otherwise, differences are calculated:

$$R_t = A_{t|T} - A_{t|t}$$

In the additive adjustment case, revisions of seasonal factors are always calculated as differences, $S_{t|T} - S_{t|t}$, or, with projected seasonal factors $S_{t|T} - S_{t|t^*}$, where t^* represents the ending date of the series used to obtain the projected factor for month t.

Let $C_{t|n}$ denote the month-to-month (or quarter-to-quarter) change in the seasonally adjusted series at time t calculated from the series $y_1, y_2, \ldots, y_n$; then we can write

$$C_{t|n} = \frac{A_{t|n} - A_{t-1|n}}{A_{t-1|n}},$$

and the revision of these changes can be defined as

$$R_t = C_{t|T} - C_{t|t}.$$

Revisions for the month-to-month changes in the trend component are computed in the same manner.

The **sadjlags** and **trendlags** arguments produce an analysis of the revisions history for different lags past the concurrent observation. The target for this revisions analysis depends on the value of the **target** argument. Table 6.2 shows how the lagged revisions are calculated for for the different values of **target**.

Estimate	*Concurrent Target*	*Final Target*
Seasonally Adjusted Series	$(A_{t\|t+lag_i} - A_{t\|t})/A_{t\|t}$	$(A_{t\|T} - A_{t\|t+lag_i})/A_{t\|t+lag_i}$
Final Trend Component	$(T_{t\|t+lag_i} - T_{t\|t})/T_{t\|t}$	$(T_{t\|T} - T_{t\|t+lag_i})/T_{t\|t+lag_i}$
Change in Seasonally Adjusted Series (or Trend)	$C_{t\|t+lag_i} - C_{t\|t}$	$C_{t\|T} - C_{t\|t+lag_i}$

Estimate gives the estimate from the seasonal adjustment.
Concurrent Target gives the formula for the lagged revision history where the target is assumed to be the concurrent estimate.
Final Target gives the formula for the lagged revision history where the target is assumed to be the final estimate.
$A_{t|i}$ is the value of the seasonally adjusted series at time t calculated from the series up to time i.
$T_{t|i}$ is the value of the trend component at time t calculated from the series up to time i.
$C_{t|i}$ is the value of the change in the seasonally adjusted series at time t calculated for the series up to time i.

Table 6.2: **Revision Measure Calculated for Revision Lag Analysis**

If lags corresponding to one and two years (12 and 24 for monthly data, 4 and 8 for quarterly data) are included in **sadjlags**, then the revision between the seasonal adjustment calculated one year after time t and the adjustment 2 years after time t is also calculated: for monthly series, this is

$$RY_t = \frac{A_{t|t+24} - A_{t|t+12}}{A_{t|t+12}}.$$

This is done only for the seasonally adjusted series and the month-to-month (quarter-to-quarter) change of the seasonally adjusted series.

The analysis of the lagged revisions can give a useful picture of the behavior of the revisions over time. Using the concurrent estimate as the target shows how much a given adjustment changes as you add more data; using the final estimate as the target shows how quickly a given estimate converges to the final value.

Another motivation for the **sadjlags** and **trendlags** options is the fact that concurrent estimates are often based on preliminary data for the current month (or quarter). If the final data for the month are not available until two additional months have passed, then it would be appropriate to set `sadjlags = 2` in order to study the revisions to the adjustment based on the final datum for each month. For trends, there is the additional motivation that concurrent trend estimates are often unstable. For this reason, some analysts wait until several subsequent months of data are available for trend estimation before examining the `X-13ARIMA-SEATS` trend for a recent month. For an analyst who waits three months, `trendlags = 3` will provide the revisions of the trend estimates of interest.

7 Documentation for Individual Specs

Contents

Tables

The following pages provide detailed documentation on all of the specs, with discussions of the available arguments and their default values. Each spec's documentation also includes several examples illustrating its use. For the **series** and **transform** specs the examples are intended only to illustrate the capabilities of these specs. They do not show complete spec files in the sense that if these examples were used as input to the `X-13ARIMA-SEATS` program, they would produce no useful output. For the remaining specs (**composite**, **x11**, **identify**, **regression**, **arima**, **estimate**, **outlier**, **check**, **forecast**, **metadata**, **spectrum**, **slidingspans**, and **history**) the examples all show complete spec files that could be used, except that data sets (e.g., for the input series appearing in the **series** spec, or for a user-defined regression variable in the **regression** spec) are often abbreviated using the $\cdots$ notation.

Readers will notice that the examples for a given spec tend to vary, not only in content, but also in format. This is done deliberately to illustrate and emphasize the flexibility the user has in formatting the spec file.

The next few paragraphs will give the reader a summary of what specs to include in the input file when doing general tasks (such as a simple seasonal adjustment or modeling run). Except in certain default situations, arguments must be specified within each spec to accomplish these tasks. Information about these arguments can be formed within the sections of this chapter devoted to the individual specs.

For the reader who wants the shortest path to a seasonal adjustment, the essential specs are **series** and **x11**. These will yield a default `X-11` seasonal adjustment. If it is not clear whether the seasonal adjustment should be additive or multiplicative then the **transform** spec should be added. If an elementary approach to trading day and moving holiday effect estimation and adjustment is desired, then add **x11regression**. The **sliding spans** and **history** specs provide diagnostics for the stability of the adjustment when the span of data used to calculate the adjustment changes.

For the reader wanting the shortest path to modeling a time series, the essential specs are **series, automdl** (or **pickmdl**), and possibly **transform**. Add the **forecast** spec if forecasting is desired, add **outlier** if there are problematic data values or data movements, and add **regression** if trading day or holiday components may be present in the series. The **arima** spec replaces **automdl** if custom rather than automatic modeling is desired. It is supported by **identify**. The **check** spec provides standard model-fit diagnostics. The **history** spec provides forecasting diagnostics for comparing two models, and **estimate** offers estimation options and the ability to reuse stored models.

Time series models (obtained via **automdl/pickmdl/arima** and **transformation** can improve seasonal adjustment by extending the data with forecasts (via **forecast**), by providing a way of dealing with disruptions to the level of the series (via **outlier**) and by providing estimates of trading day and holiday effects (via **regression**) that are sometimes better than those obtained from **x11regression**.

The **composite** spec is required to obtain the indirect adjustment of an aggregate series from adjustments of its components and to compare this adjustment with its direct adjustment. of its components and to compare this adjustment with its direct adjustment. For indirect adjustment the **composite** spec replaces the **series** spec.

7.1 ARIMA

DESCRIPTION

Specifies the ARIMA part of the regARIMA model. This defines a pure ARIMA model if the **regression** spec is absent. The ARIMA part of the model may include multiplicative seasonal factors and operators with missing lags. Using the **ar** and **ma** arguments, initial values for the individual AR and MA parameters can be specified for the iterative estimation. Also, individual parameters can be held fixed at these initial values while the rest of the parameters are estimated.

USAGE

```
arima {   model = ([2 3] 1 1)(0 1 1)12
          title = "ARIMA Model"
          ar = (0.3f, -0.14)
          ma = (-0.7 0.85f) }
```

ARGUMENTS

ar Specifies initial values for nonseasonal and seasonal autoregressive parameters in the order that they appear in the **model** argument. If present, the **ar** argument must assign initial values to *all* AR parameters in the model. Initial values are assigned to parameters either by specifying the value in the argument list or by explicitly indicating that it is missing. Missing values take on their default value of 0.1. For example, for a model with two AR parameters, `ar = (0.7,)` is equivalent to `ar = (0.7, 0.1)`, but `ar = (0.7)` is not allowed. For a model with three AR parameters, `ar = (0.8, , -0.4)` is equivalent to `ar = (0.8, 0.1, -0.4)`. To hold a parameter fixed during estimation at its initial value, place an **'f'** immediately after the value in the **ar** list, e.g., `ar = (0.7f, 0.1)`.

ma Specifies initial values for all moving average parameters in the same way **ar** does for autoregressive parameters.

model Specifies the ARIMA part of the model. The format follows standard Box-Jenkins (1976) notation. In this notation a nonseasonal ARIMA model is specified as $(p\ d\ q)$, where p is the nonseasonal AR order, d is the number of nonseasonal differences, and q is the nonseasonal MA order. A multiplicative seasonal ARIMA model is specified as $(p\ d\ q)(P\ D\ Q)$, where p, d, and q are as before, P is the seasonal AR order, D is the number of seasonal differences, and Q is the seasonal MA order. Here, the first ARIMA factor, $(p\ d\ q)$, is assumed to be nonseasonal (i.e., its period is one) and the second ARIMA factor, $(P\ D\ Q)$, is assumed to be seasonal with the seasonal period set in the **series** spec. More than two ARIMA factors can be specified, and ARIMA factors can explicitly be given seasonal periods that differ from the default choices. See DETAILS for more information.

The operator orders $(p\ d\ q)$ in the ARIMA factors may be separated by spaces or commas, e.g., `(0 1 1)` is the same as `(0, 1, 1)`. Operators with missing lags are specified by

enclosing those lags present in brackets, with the lags in ascending order. For example, `model = ([2 3] 0 0)` specifies the model $(1 - \phi_2 B^2 - \phi_3 B^3)z_t = a_t$.

print and **save** No output tables are available for this spec.

title Specifies a title for the ARIMA model, in quotes. It must be less than 80 characters. The title appears above the ARIMA model description and the table of estimates. The default is to print `ARIMA Model`.

DETAILS

The **arima** spec **cannot** be used in the same spec file as the **pickmdl** or **automdl** specs; the **model**, **ma**, and **ar** arguments of the **arima** spec cannot be used when the **file** argument is specified in the **estimate** spec.

The model argument may include as many ARIMA factors as desired. However, there is a limit of 133 total AR, MA, and differencing coefficients in the model. Also, the maximum lag of any AR or MA parameter is 36, and the maximum number of differences in any ARIMA factor (nonseasonal or seasonal) is 3. (The latter two limits can be changed – see Section 2.8.)

In general, ARIMA factors are specified in the standard $(p\ d\ q)_s$ format, where s is the seasonal period of the operator. Thus, putting `(0 1 1)6` in the model argument includes differencing by $1 - B^6$ and a $1 - \Theta B^6$ MA term in the model. However, if the seasonal period s is not specified after an ARIMA factor, it is determined according to the following default rules. The first ARIMA factor without a specified seasonal period is assumed to be nonseasonal, i.e., its seasonal period is assumed to be one. The second ARIMA factor without a specified seasonal period is assumed to be a seasonal factor with the seasonal period set in the **series** spec. For example, if `period = 12` is specified in the **series** spec (or if the period is set to 12 because the start date there is given as *year.month*), then `model = (0 1 1)(0 1 1)` and `model = (0 1 1)1(0 1 1)12` are equivalent. If additional ARIMA factors are specified, these are assumed to be nonseasonal unless they are explicitly given a seasonal period. See Example 7.1 for an illustration of a model with three ARIMA factors. Note that if the seasonal period is one, then any ARIMA factors without a specified seasonal period have period one.

Users should not specify initial values for MA parameters that yield an MA polynomial with roots inside the unit circle. (See Section 5.4.) Doing so will cause the program to stop and print an error message asking the user to re-specify the initial parameters and rerun the program. Initial parameters that yield an MA polynomial with roots *on* the unit circle are allowed only if this non-invertible polynomial is not being estimated. That is, this is allowed if no estimation is being done, or if the parameters in this polynomial are specified as fixed during estimation. For example, if a model has a first order seasonal MA parameter as the only MA parameter, then `ma = (1.0f)` is always allowed, `ma = (1.0)` is allowed only if no estimation is done, and `ma = (1.1)` is never allowed.

If the likelihood function that is exact for AR polynomials is used (`exact = arma`, which is the default — see the **estimate** spec), users should not specify initial values for AR parameters that yield a non-stationary AR polynomial (one with roots on or inside the unit circle). Doing so will cause the program to stop and print an error message asking the user to re-specify the initial parameters and rerun the program.

The use of fixed coefficients in the ARIMA model can invalidate *AIC* and the other model selection statistics as well as some goodness-of-fit diagnostics – see the DETAILS sections of **estimate** and **check**.

EXAMPLES

The following examples show complete spec files.

Example 1 Specify and estimate a nonseasonal ARIMA model with a first difference and an MA parameter at lag 1, i.e., $(1 - B)y_t = (1 - \theta B)a_t$.

```
series { title  =  "Quarterly Grape Harvest" start = 2000.1
         period =  4
         data   = (8997 9401 ... 11346) }
arima { model = (0 1 1) }
estimate { }
```

Example 2 Specify and estimate the following seasonal ARIMA model for y_t, the logarithm of an original time series: $(1-\phi_1 B-\phi_2 B^2)(1-B)(1-B^{12})y_t = (1-\Theta_{12}B^{12})a_t$. Note that the start date in the **series** spec specifies a month, which sets the seasonal period to 12.

```
series { title = "Monthly sales" start = 1996.jan
          data = (138 128 ... 297) }
transform { function = log }
arima { model  = (2 1 0)(0 1 1) }
estimate { }
```

Example 3 Specify and estimate a regARIMA model with fixed seasonal effects, a trend constant, and the ARIMA (0 1 1) model for the regression errors. The model is then $(1-B)\,(y_t - \sum \beta_i M_{it} - c \cdot t) = (1 - \theta B)a_t$, where the M_{it} are the fixed seasonal effect regression variables.

```
Series { Title = "Monthly Sales"  Start = 1996.jan
         Data = (138 128 ... 297)  }
Transform  { Function = log  }
Regression  { Variables= (seasonal const) }
Arima  { Model = (0 1 1) }
Estimate  {  }
```

Example 4 Specify and estimate a model with one difference and an AR(2) operator with lag one missing; i.e., the model is $(1 - \phi_2 B^2)(1 - B)y_t = a_t$.

```
series{title = "Annual Olive Harvest" start = 2000
        data = (251 271 ... 240)  }
arima{model  = ([2] 1 0)}
estimate{ }
```

Example 5 Specify and estimate a model with a trend constant and with regression errors z_t following an ARIMA model with one seasonal difference and a first order seasonal moving average, but no nonseasonal factor, i.e., $(1 - B^{12})z_t = (1 - \Theta B^{12})a_t$. Note that the seasonal period of the ARIMA factor must be given explicitly in the **model** argument, because, as there is only one ARIMA factor, it would otherwise be assumed to be nonseasonal.

```
series { title = "Monthly sales"  start = 1996.jan
         data = (138 128 ... 297) }
transform { function = log }
regression { variables = const }
arima { model  = (0 1 1)12 }
estimate { }
```

Example 6 Specify and estimate a model including three ARIMA factors. The ARIMA model for the regression errors z_t is $(1-\phi_1 B)(1-\phi_3 B^3)(1-B)z_t = (1-\Theta B^{12})a_t$. The $1-\phi_3 B^3$ operator might be used to account for quarterly autocorrelation since each quarter is comprised of three months. Note that only the period of the quarterly factor needs to be given.

```
series { title = "Monthly sales"  start = 1996.jan
         data = (138 128 ... 297) }
transform { function = log }
regression { variables = (const seasonal)}
arima { model  = (1 1 0)(1 0 0)3(0 0 1)}
estimate { }
```

Example 7 Specify and estimate a model with regression errors z_t following the "airline model," ARIMA $(0\ 1\ 1)(0\ 1\ 1)_{12}$, with the seasonal MA parameter fixed at 1.0. The model used for z_t is $(1-B)(1-B^{12})z_t = (1-\theta B)(1-1.0B^{12})a_t$. The initial value of 0.1 used for θ is indicated by a missing value in the **ma** list. This model is actually equivalent to that used in Example 3, since it results from overdifferencing the model specified there by $1-B^{12}$. (See Section 5.4 for a discussion of overdifferencing.)

```
series { title = "Monthly sales"  start = 1996.jan
         data = (138 128 ...  297) }
transform{ function = log }
arima  { model = (0 1 1)(0 1 1)12
            ma = ( ,1.0f)}
estimate { }
```

7.2 AUTOMDL

DESCRIPTION

Specifies that the ARIMA part of the regARIMA model will be sought using an automatic model selection procedure derived from the one used by TRAMO (see Gómez and Maravall (2001a)). The user can specify the maximum ARMA and differencing orders to use in the model search, and can adjust thresholds for several of the selection criteria.

USAGE

```
automdl {   maxorder = (3 1)
            maxdiff = (1 1)  or  diff = (1 0)
            acceptdefault = no
            checkmu = yes
            ljungboxlimit = 0.99
            mixed = yes
            print = (none bestfivemdl autochoice)
            savelog = automodel
            seasonaloverdiff = yes
        }
```

ARGUMENTS

acceptdefault Controls whether the default model is chosen if the Ljung-Box Q statistic for its model residuals (checked at lag 24 if the series is monthly, 16 if the series is quarterly) is acceptable (`acceptdefault = yes`). If the default model is found to be acceptable, no further attempt will be made to identify a model or differencing order. The default for **acceptdefault** is `acceptdefault = no`.

checkmu Controls whether the automatic model selection procedure will check for the significance of a constant term (`checkmu = yes`) or will instead maintain the choice made by the user in the **regression** spec (`checkmu = no`). The default for **checkmu** is `checkmu = yes`.

diff Fixes the orders of differencing to be used in the automatic ARIMA model identification procedure. The **diff** argument has two input values, the regular differencing order and the seasonal differencing order. Both values must be specified; there is no default value. Acceptable values for the regular differencing orders are 0, 1 and 2; acceptable values for the seasonal differencing orders are 0 and 1. If the **diff** and **maxdiff** arguments are both specified in the same spec file, the values for the **diff** argument are ignored, and the program performs automatic identification of nonseasonal and seasonal differencing using the limits specified in **maxdiff**.

ljungboxlimit Acceptance criterion for confidence coefficient of the Ljung-Box Q statistic. If the Ljung-Box Q for the residuals of a final model (checked at lag 24 if the series is monthly, at

lag 16 if the series is quarterly) is greater than **ljungboxlimit**, the model is rejected, the outlier critical value is reduced, and model and outlier identification (if specified) is redone with a reduced value (see **reducecv** argument). The default for **ljungboxlimit** is `ljungboxlimit = 0.95`.

maxdiff Specifies the maximum orders of regular and seasonal differencing for the automatic identification of differencing orders. The **maxdiff** argument has two input values, the maximum regular differencing order and the maximum seasonal differencing order. Acceptable values for the maximum order of regular differencing are 1 or 2, and the acceptable value for the maximum order of seasonal differencing is 1. If the **diff** and **maxdiff** arguments are both specified in the same spec file, the values for the **diff** argument are ignored, and the program performs automatic identification of nonseasonal and seasonal differencing using the limits specified in **maxdiff**. The default is `maxdiff = (2 1)`.

maxorder Specifies the maximum orders of the regular and seasonal ARMA polynomials to be examined during the automatic ARIMA model identification procedure. The **maxorder** argument has two input values, the maximum order of regular ARMA model to be tested and the maximum order of seasonal ARMA model to be tested. The maximum order for the regular ARMA model must be greater than zero, and can be at most 4; the maximum order for the seasonal ARMA model can be either 1 or 2. The default is `maxorder = (2 1)`.

mixed Controls whether ARIMA models with nonseasonal AR and MA terms or seasonal AR and MA terms will be considered in the automatic model identification procedure (`mixed = yes`). If `mixed = no`, mixed models will not be considered. Note that a model with AR and MA terms in both the seasonal and nonseasonal parts of the model can be acceptable, as long as neither part include both AR and MA terms. For example, when `mixed = no`, an ARIMA (0 1 1)(1 1 0) model would be considered, but an ARIMA (1 1 1)(0 1 1) model would not, since there are AR and MA terms in the nonseasonal part of the model. The default for **mixed** is `mixed = yes`.

print The tables available for output are listed in Table 7.1. The save option is not available for this spec. The `header`, `autochoice`, and `unitroottest` tables are printed out by default. For a complete listing of the `brief` and `default` print levels for this spec, see Appendix B.

savelog The diagnostics available for output to the log file (see section 2.6) are listed on Table 7.2.

seasonaloverdiff Controls whether the automatic model selection procedure will check for seasonal overdifferencing in the final model (`seasonaloverdiff = yes`), or whether it only checks for nonseasonal overdifferencing (`seasonaloverdiff = no`). The default for **seasonaloverdiff** is `seasonaloverdiff = no`. For an explanation of how this test is done, see DETAILS.

name	*short*	*description of table*
autochoice	ach	model choice of automatic model procedure
autochoicemdl	amd	summary output for models estimated during choice of ARMA model orders
autodefaulttests	adt	tests performed on the default model (usually the airline model) of the automatic model identification procedure
autofinaltests	aft	final tests performed on the model identified by **automdl**
autoljungboxtest	alb	check of the residual Ljung-Box statistic
bestfivemdl	b5m	summary of best five models found during choice of ARMA model orders
header	hdr	header for the automatic modeling output
unitroottest	urt	choice of differencing
unitroottestmdl	urm	summary output for models estimated during difference order identification

Name gives the name of each table for use with the **print** argument.
Short gives a short name for these tables.

Table 7.1: **Available Output Tables for Automdl**

name	*short*	*description of diagnostic*
alldiagnostics	all	all modeling diagnostics listed in this table
autodiff	adf	choice of differencing by automatic model identification procedure
automodel	amd	choice of ARIMA model by automatic model identification procedure
bestfivemdl	b5m	summary for best five models found during choice of ARMA model orders
mean	mu	choice regarding use of constant term with automatically identified model

Name gives the name of each diagnostic for use with the **savelog** argument.
Short gives a short name for these diagnostics.

Table 7.2: **Available Log File Diagnostics for Automdl**

RARELY USED ARGUMENTS

armalimit Threshold value for t-statistics of ARMA coefficients used for final test of model parsimony. If the highest order ARMA coefficient has a t-value less than this value in magnitude, the program will reduce the order of the model. The value given for **armalimit** is also used for the final check of the constant term; if the constant term has a t-value less than **armalimit** in magnitude, the program will remove the constant term from the set of regressors. This value should be greater than zero. The default is `armalimit = 1.0`.

balanced Controls whether the automatic model procedure will have a preference for balanced models (where the order of the combined AR and differencing operator is equal to the order of the combined MA operator). Setting `balanced = yes` yields the same preference as the TRAMO program. The default is `balanced = no`.

exactdiff Controls if exact likelihood estimation is used when Hannen-Rissanen fails in automatic difference identification procedure (`exactdiff = yes`), or if conditional likelihood estimation is used (`exactdiff = no`). The default is to start with exact likelihood estimation and switch to conditional if the number of iterations for the exact likelihood procedure exceeds 200 iterations (`exactdiff = first`).

fcstlim Sets the acceptance threshold for the within-sample forecast error test of the final identified model. The absolute average percentage error of the extrapolated values within the last three years of data must be less than this value for forecasts to be generated with the final model. For example, `fcstlim = 20` sets this threshold to 20 percent. The value entered for this argument must not be less than zero or greater than 100. This option is only active when `rejectfcst = yes`. The default is `fcstlim = 15`.

hrinitial Controls whether Hannan-Rissanen estimation is done before exact maximum likelihood estimation to provide initial values when generating likelihood statistics for identifying the ARMA orders (`hrinitial = yes`). If `hrinitial = yes`, then models for which the Hannan-Rissanen estimation yields coefficients that are unacceptable initial values to the exact maximum likelihood estimation procedure will be rejected. The default is `hrinitial = no`.

reducecv The percentage by which the outlier critical value will be reduced when an identified model is found to have a Ljung-Box Q statistic with an unacceptable confidence coefficient. This value should be between 0 and 1, and will only be active when automatic outlier identification is selected. The reduced critical value will be set to $(1-\text{reducecv})\times CV$), where CV is the original critical value. The default is `reducecv = 0.14286`.

rejectfcst If `rejectfcst = yes`, then a test of the out-of-sample forecast error of the final three years of data will be generated with the identified model to determine if forecast extension will be applied. If the forecast error exceeds the value of `fcstlimit`, forecasts will not be generated with the final identified model, but the model will be used to generate preadjustment factors for calendar and outlier effects. The default is `rejectfcst = no`.

urfinal Threshold value for the final unit root test. If the magnitude of an AR root for the final model is less than this number, a unit root is assumed, the order of the AR polynomial is reduced by one, and the appropriate order of differencing (nonseasonal, seasonal) is increased. This value should be greater than one. The default is `urfinal = 1.05`.

DETAILS

The **automdl** spec *cannot* be used in the same spec file as the **pickmdl** or **arima** specs, or when the **file** argument is specified in the **estimate** spec.

The automatic ARIMA model selection procedure implemented into Version 0.3 is based on the procedure in the TRAMO time series modeling program developed by Victor Gómez and Agustin Maravall (Gómez and Maravall 1996). It is very similar to TRAMO's procedure but contains modifications to make use of `X-13ARIMA-SEATS`' different model estimation procedure, regARIMA model options, transformation and outlier identification procedures, and model diagnostics. Some additional tests have also been added. Consequently, the model selected can differ from the model TRAMO would select. Extensive testing has shown that the models selected are usually at least as good as those selected by TRAMO (preliminary results in Hood 2002a).

The TRAMO procedure is largely documented in Gómez and Maravall (2001a), but the actual implementation of the procedure in the current TRAMO program differs somewhat from the description that appears in the paper.

An overview of the ARIMA model selection procedure is given below, as given in Monsell (2002, 2006). The procedure can be summarized in five stages:

- **default model estimation:** a default model is estimated, initial outlier identification and regressor tests are performed, and residual diagnostics are generated;
- **identification of differencing orders:** empirical unit root tests are performed to determine the orders of differencing needed for the model;
- **identification of ARMA model orders:** an iterative procedure is applied to determine the order of ARMA parameters;
- **comparison of identified model with default model:** the identified model is compared to the default model; and
- **final model checks:** where the final model is checked for adequacy.

Note that the second stage is optional, as the user can specify the orders of regular and seasonal differencing using the **diff** argument.

Default model estimation

The first step of the automatic outlier procedure is to estimate a default model. For monthly and quarterly series, this is initially an "airline" model: ARIMA $(0\ 1\ 1)(0\ 1\ 1)_s$.

The default model is used to perform a number of tasks. If tests for trading day, Easter or user-defined regressors are requested by the user in the **regression** spec, an initial check for the significance of these effects is performed using the default model. The `X-13ARIMA-SEATS` program's **aictest** option is used to check the significance of the regressors using a small sample variant of AIC called AICC (otherwise known as the F-adjusted Akaike's Information Criterion, see Hurvich and Tsai 1989). For more details on how AIC tests for regressors are implemented within `X-13ARIMA-SEATS`, see the DETAILS section of the **regression** spec.

The procedure then checks the significance of including a constant term in the regARIMA model. A t-statistic for the mean of the model residuals is generated and is checked against a critical value of 1.96.

Once these tests are complete, the program performs automatic outlier identification (if specified by the user in the **outlier** spec). Details concerning `X-13ARIMA-SEATS` program's automatic outlier identification routine can be found in Appendix B of Findley, Monsell, Bell, Otto, and Chen (1998), or in the DETAILS section of the **outlier** spec.

After outlier identification, the trading day, Easter and constant regressors are checked to see if they are still significant. This test is simpler: t-tests are generated, and a critical value of 1.96 is used to determine if the regressors are significant (except for the constant regressor, which uses the same value specified in **armalimit**). For the trading day regressor, at least one of the regressors needs to have a critical value greater than 1.96. Note that this test is done for trading day and Easter regressors only if the **aictest** argument is given in the **regression** spec; the constant regressor is always tested.

After the regression part of the default model is determined, the program generates residual diagnostics for this model. These diagnostics are:

- the Ljung-Box Q statistic for the model residuals (at lag 24 if this is a monthly series, at lag 16 for a quarterly series),
- the confidence coefficient of this Ljung-Box Q statistic,
- a t-value for the mean of the regARIMA model residuals, and
- an estimate of the residual standard error.

The confidence coefficient is defined to be 1 minus the p-value of the Ljung-Box Q statistic, as in Lehman (1986). The TRAMO documentation (Gómez and Maravall 1996) refers to the confidence coefficient as the significance level.

These diagnostics will be compared later to those of the model selected by the automatic model identification procedure. The model identified by this procedure must show some improvement over the default model in these residual diagnostics; otherwise, the program will accept the default model.

Just before the model identification phase begins, the program removes the regression effects estimated by the default model from the original series. It is this series, rather than the original series, that is used in the model identification routines.

In this way, an attempt is made to robustify the model identification process, to ensure that the choice of differencing and model orders are not unduly affected by outliers, calendar effects, and other regression effects. This regression residual series is referred to as the **linearized series** in the TRAMO documentation.

Identification of differencing orders

Now the program will attempt to identify an appropriate order of differencing for the "linearized" series computed earlier. This is done by performing a series of unit root tests, fitting different ARMA models to the (sometimes differenced) linearized series. The estimation of these models is done using a technique called the

Hannan-Rissanen method (see Hannan and Rissanen 1982, Gómez and Maravall 2001a). This method computes the estimates of the ARMA parameters by setting up a linear regression using lagged values of the original series (to estimate the AR parameters) and lagged estimates of the innovations generated recursively from the autocovariances (to estimate the MA parameters). Biases in the MA parameters are corrected with a technique provided by Chen (1985), and the MA parameter estimates are improved when AR parameters are present by applying Chen's method to the series filtered by the AR filter (see Gómez 1998).

Step 1: The first stage of the procedure fits a $(2\ 0\ 0)(1\ 0\ 0)_s$ ARIMA model to the linear series using the Hannan-Rissanen method, and examines the real AR roots of the estimated model. The program considers such a root a unit root if the modulus of the root is less than 1.042, and the order of differencing that corresponds to the root (seasonal or nonseasonal) is increased by one.

If the Hannan-Rissanen procedure estimates a model with roots inside the unit circle, `X-13ARIMA-SEATS` re-estimates the model using exact maximum likelihood estimation, and the modulus test described above is applied to the resulting estimates.

Step 2: If differencing was found in Step 1, the linearized series is differenced at the start of Step 2. An ARMA $(1\ 1)(1\ 1)_s$ model is then fit to the resulting series, and the AR parameters are checked to see if they are close to one. The criterion for "close to one" depends on whether the program is examining the regular or seasonal AR coefficients.

If an AR coefficient is found that meets the criterion, the program checks to see if there is a common factor in the corresponding AR and MA polynomials of the ARMA model that can be canceled.

If there is no cancellation, the differencing order changes. The linearized series is differenced using this new set of differencing orders. The ARMA model is fit again, and the program checks to see if any additional differencing can be found. This process repeats until no more differencing is found.

Once the differencing orders are determined, a t-statistic for the mean term of the fully differenced series is generated based on either the sample mean (if no differencing is identified) or by adding a constant term to the regARIMA model. The critical value of the test is set based on the number of observations in the series.

This is a simplified overview of the actual process. Other tests may be performed if no differencing is found in Step 2, and the procedure has checks implemented to avoid going from no differencing after Step 1 to both regular and seasonal differencing after the first stage of Step 2. For more details, see Gómez and Maravall (2001a).

Identification of ARMA model orders

Once an appropriate set of differencing orders has been found, the program turns to the identification of the orders of the ARMA model. The basic procedure involves comparing values of the Bayesian Information Criterion (see Schwarz 1978) of a number of models, up to a maximum order for the regular and seasonal ARMA polynomial which can be specified by the user. As with Akaike's AIC criterion, the model with the lowest BIC is preferred.

The formula below is the classical formula for BIC that is printed out in the `X-13ARIMA-SEATS` output.

$$BIC_N = -2\widehat{L}_N + n_p \log N,$$

where $\widehat{L}_N$ is the maximized value of the log likelihood evaluated over N observations, n_p is the number of estimated parameters in the model, including the white noise variances, and N is the number of observations remaining after application of the model's differencing and seasonal differencing operations.

TRAMO uses a variant of this BIC formula in its automatic model identification procedure which divides the log likelihood and the penalty term by N. In order to be able to use TRAMO's final selection criteria, it is necessary that `X-13ARIMA-SEATS` have a comparable variant of BIC. So `X-13ARIMA-SEATS` generates the following BIC which is only used for the automatic modeling procedure:

$$BIC2_N = (-2\widehat{L}_N + n_p \log N)/N.$$

The identification procedure allows the user to specify the maximum order of regular AR and MA polynomial (m_r, can be as high as 3, with a default of 2) and seasonal AR and MA polynomial (m_s, can be as high as 2, wiith a default of 1) up to which the program estimates ARIMA models and generates values of BIC2. A three stage procedure is detailed in Gómez and Maravall (2000) that reduces the number of models estimated.

To get an initial estimate for the seasonal model orders, BIC2 is computed for all ARIMA models of the form $(3\ d\ 0)(P\ D\ Q)_s$, where d and D are the previously determined or specified regular and seasonal orders of differencing, respectively, and $0 \leq P, Q \leq m_s$. The program then chooses the pair of values P and Q that minimize BIC2.

Using these values of P and Q, the program now tries to identify the best model orders for the nonseasonal part of the ARIMA model. BIC2 is computed for all ARIMA $(p\ d\ q)(P\ D\ Q)_s$ models, where d and D are the regular and seasonal orders of differencing, respectively, and $0 \leq p, q \leq m_r$. The pair of values p and q are chosen that minimize BIC2.

Using these values of p and q, the selection of seasonal model orders is now refined. The program computes BIC2 for all ARIMA $(p\ d\ q)(P\ D\ Q)_s$ models, where d and D are the regular and seasonal orders of differencing, respectively, and $0 \leq P, Q \leq m_s$. The pair of values P and Q are chosen that minimize BIC2.

There is one exception for this third stage of the process: if no seasonal AR was found in the first stage of the process, and a seasonal differencing is present, then the program only computes BIC2 for ARIMA $(p\ d\ q)(0\ D\ Q)_s$ models, where d and D are the regular and seasonal orders of differencing, respectively, and $0 \leq Q \leq m_s$. The values of Q is chosen that minimizes BIC2.

During the ARMA order selection process, `X-13ARIMA-SEATS` keeps track of the models with the five smallest BIC2s. Once the identification phase is over, the program will compare the BIC2 for the best model with that of the other 4 models to see if there are models with BIC2s that are "close" enough that there is no "significant" difference between the models. The criteria for "close enough" depends on the length of the series, the magnitude of the difference between the BICs, and other criteria.

If the program finds a model that is "close" enough to the best model, the program also checks to see whether the model with the higher BIC is more parsimonious (especially in the seasonal operator) than the best model. If so, the program will accept the more parsimonious model.

The program also checks for model balance. A model is said to be more balanced than a competing model if the absolute difference between the total orders of the AR plus differencing and MA operators is smaller. While balanced models are useful for model-based seasonal adjustment, it is unclear whether this criterion is useful for the types of operations `X-13ARIMA-SEATS` does, as it induces a small bias toward mixed models, and

mixed ARMA models can be difficult to estimate due to near cancellation. Therefore, `X-13ARIMA-SEATS` makes checking for model balance at this stage optional; the default is not to test for model balance.

If the identified model is different from the default model, the program redoes many of the steps that determined the regressors of the default model. Outlier regressors identified for the default model are removed from the identified model. If the user has specified AIC testing of trading day, Easter, or user defined regressors, this testing will be redone for the identified model. Then outlier identification is redone for the identified model.

Comparison of identified model with default model

At this point, if the identified model is not the default model, the residual diagnostics from the automatically identified model are compared to those of the default model. Let Q_A be the confidence coefficient of the Ljung-Box Q statistic for the automatically identified model (at lag 24 for monthly series, at lag 16 for quarterly series), Q_D the confidence coefficient of the Ljung-Box Q statistic for the default model, RSE_A the residual standard error for the automatically identified model, and RSE_D the residual standard error for the default model.

The default model will be preferred over the automatically identified model if

- the number of outliers automatically identified for the default model is less than or equal to the number of automatically identified outliers for the automatically identified model, AND
- $Q_A < 0.95$ and $Q_D < 0.75$ and $RSE_D < RSE_A$, OR
- $Q_A > 0.95$ and $Q_D < 0.95$ (only on the first pass), OR
- $Q_A < 0.95$ and $Q_D < 0.75$ and $Q_D < Q_A$ and $RSE_D < RSE_A \times 1.013$, OR
- $Q_A \geq 0.95$ and $Q_D < 0.95$ and $RSE_D < RSE_A \times 1.013$, OR
- the automatic model is $(1\ 0\ 1)(0\ 1\ 1)_s$ or $(1\ 0\ 0)(0\ 1\ 1)_s$ and $\phi_1 \geq 0.82$, OR
- the automatic model is $(0\ 1\ 1)(1\ 0\ 1)_s$ or $(0\ 1\ 1)(1\ 0\ 0)_s$ and $\phi_s \geq 0.65$.

The program then tests to see if the preferred model is acceptable. The confidence coefficient of the Ljung-Box Q statistic is used as the criterion. If this value is greater than 0.975 (by default), the program will decrease the critical value of the automatic outlier identification based on the value of **reducecv**, given the formula below:

$$CV_r = (1 - \text{reducecv}) \times CV$$

where CV is the original critical value and CV_r is the reduced outlier critical value. The reduced critical value is not allowed to be smaller than 2.8.

The program will then attempt to redo the automatic modeling procedure and re-identify outliers with the new critical value. The re-identification of outliers will be done without another automatic model identification if no outliers were identified earlier.

Diagnostics are then generated for the revised model, and these diagnostics are compared to those of the previous preferred model. The Ljung-Box Q test is performed again; this time, the test fails if the confidence coefficient is greater than 0.99. If this does not result in a model with an acceptable Ljung-Box Q, the program sets the model to be $(3\ d\ 1)(0\ D\ 1)_s$ and attempts to identify outliers for this model.

Finally, t-statistics for the trading day, Easter, and constant regressors are checked, as they were after automatic model identification of the default model. Again, a critical value of 1.96 is used to determine if the regressors are significant (except for the constant regressor, which uses the same value specified in **armalimit**). For the trading day regressor, at least one of the regressors needs to have a critical value greater than 1.96.

Final model checks

Once a final model is selected, a final series of tests for model inadequacy is performed.

First, the model is checked for unit roots in the AR polynomial, to see if the order of regular or seasonal differencing should be corrected. The program detects a unit AR root if the modulus of a given AR root is less than or equal to 1.05. If a unit root is detected, the program then reduces the order of the appropriate AR polynomial, and increases the appropriate order of differencing. The program then estimates the updated model, and regenerates the model diagnostics.

Next, the model is checked for unit roots in the nonseasonal MA polynomial. Models with nonseasonal MA unit roots have led to inadmissible decompositions in model-based signal extraction procedures such as SEATS. The program takes the sum of the nonseasonal MA coefficient estimates and checks to see if this sum is within 0.001 of one. If so, the order of regular differencing is reduced by one, and the order of the corresponding MA polynomial is reduced by one. A constant term is added to the regARIMA model (if one is not already present), and this constant term is checked for significance.

In addition, there is an optional test for unit roots in the seasonal MA polynomial. The program takes the sum of the seasonal MA coefficient estimates and checks to see if this sum is within 0.001 of one. If so, the order of seasonal differencing is reduced by one, and the order of the corresponding MA polynomial is reduced by one. Seasonal regressors are added to the regARIMA model, and these regressors are checked for significance using an F-statistic.

After the test for regular (and seasonal, if specified) overdifferencing is finished, the program then estimates the updated model, redoes outlier identification (if specified), and regenerates model diagnostics.

Note: if a SEATS seasonal adjustment is specified, the program will not test for seasonal overdifferencing, even if the user requests the test.

If a constant term is not present in the regARIMA model, the program now checks if the t-statistic for the mean of the model residuals is significant (greater in magnitude than 2.5). If the t-statistic is significant, the program adds a constant term to the set of regressors.

A test for insignificant ARMA parameters is then performed in an attempt to simplify the identified model, with t-statistics for the ARMA coefficients generated. The highest order AR, MA, and seasonal AR and MA coefficients are tested for significance, using the following criteria:

- to avoid model order reduction the t-statistic of the largest order AR, MA, seasonal AR and seasonal MA coefficients has to be larger in magnitude than the value specified for **armalimit**, and

- the absolute value of the coefficient estimate itself must be greater than 0.15 (if there are at most 150 observations in the series) or 0.10 (if there are more than 150 observations).

If more than one insignificant coefficient is found for a given type of ARMA parameter (such as nonseasonal AR, or seasonal MA coefficients) and outlier identification has been specified, the program will reduce the outlier critical value using the value specified for **reducecv**. As noted before, the reduced outlier critical value cannot be less than 2.8. The program will then try to re-identify the model, with a reduced outlier critical value.

If outlier identification was not specified, if the critical value is already 2.8, or if only one insignificant coefficient is found, the program will reduce the order of the model by setting insignificant coefficients to zero and estimate the reduced model.

Note that if there is only one ARMA parameter in the model, the program will not remove it, even if it is insignificant. Also, no ARMA coefficients are eliminated from the model if a unit root is found (that is, if the magnitude of one of the roots is less than 1.053).

EXAMPLES

The following examples show complete spec files.

Example 1 Use the automatic ARIMA modeling procedure to select a model and use it to extend the series with one year of forecasts. Trading day and stable seasonal regression effects are to be included in the models. A default seasonal adjustment is to be performed.

```
series      { title = "Monthly sales"   start = 2001.jan
              file="ussales.dat"  }
regression  { variables = (td seasonal) }
automdl     {   }
estimate    {   }
x11         {   }
```

Example 2 Similar to Example 1, except that the differencing orders are preset to a regular and seasonal difference, and the maximum regular ARMA order to be examined will be 3.

```
series      { title = "Monthly sales"   start = 2001.jan
              file="ussales.dat" }
regression  { variables = td  }
automdl     { diff = ( 1 1 )
              maxorder = ( 3, ) }
outlier     {   }
estimate    {   }
x11         {   }
```

Example 3 The same as Example 1, except that the identified model will be saved in the log file, amd the program will use AIC to check if trading day regressors are needed.

```
series      { title = "Monthly sales"   start = 2001.jan
              file="ussales.dat" }
regression  { aictest = td   }
automdl     { savelog = amd  }
estimate    {   }
x11         {   }
```

7.3 CHECK

DESCRIPTION

Specification to produce statistics for diagnostic checking of residuals from the estimated model. Statistics available for diagnostic checking include the sample ACF and PACF of the residuals with associated standard errors, Ljung-Box Q-statistics and their p-values, summary statistics of the residuals, normality test statistics for the residuals, a spectral plot of the model residuals, and a histogram of the standardized residuals.

USAGE

```
check {   maxlag = 36
          print = (none +histogram +acf)
          qtype = bp
          save = (acf)
          savelog = normalitytest
      }
```

ARGUMENTS

maxlag The number of lags requested for the residual sample ACF and PACF for both tables and plots. The default is 24 for monthly series, 8 for quarterly series.

print and **save** Table 7.3 gives the available output tables for this spec. The `acf`, `acfplot`, `histogram`, and `normalitytest` tables are printed out by default. For a complete listing of the `brief` and `default` print levels for this spec, see Appendix B.

qtype The type of residual diagnostic to be displayed with the sample autocorrelation plots. If `qtype = ljungbox` or `qtype = lb`, the Ljung-Box Q-statistic will be the one produced. If `qtype = boxpierce` or `qtype = bp`, the Box-Pierce Q-statistic will be the one produced. The Ljung-Box statistic will be produced by default.

savelog The diagnostics available for output to the log file (see section 2.6) are listed on Table 7.4.

name	*short*	*save?*	*description of table*
acf	acf	+	autocorrelation function of residuals with standard errors and Ljung-Box Q-statistics computed through each lag
acfplot	acp	·	plot of residual autocorrelation function with ± 2 standard error limits
pacf	pcf	+	partial autocorrelation function of residuals with standard errors
pacfplot	pcp	·	plot of residual partial autocorrelation function with ± 2 standard error limits
acfsquared	ac2	+	autocorrelation function of squared residuals with standard errors and Ljung-Box Q-statistics computed through each lag
acfsquaredplot	ap2	·	plot of squared residual autocorrelation function with ± 2 standard error limits
normalitytest	nrm	·	Geary's a and kurtosis statistical tests for the normality of the model residuals, as well as a test for skewness of the residuals
durbinwatson	dw	·	Durbin-Watson statistic for model residuals
friedmantest	frt	·	Friedman non-parametric test for residual seasonality
histogram	hst	·	histogram of standardized residuals and the following summary statistics of the residuals: minimum, maximum, median, standard deviation, and robust estimate of residual standard deviation (1.48 × the median absolute deviation)

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.3: **Available Output Tables for Check**

RARELY USED ARGUMENTS

acflimit Limit for the t-statistic used to determine if residual sample ACFs and PACFs are flagged as significant in the diagnostic summary file (with the file extension **.udg**). The default is 1.6.

qlimit Limit for the p-value of the Q statistic used to determine if residual sample ACFs and PACFs are flagged as significant in the diagnostic summary file (with the file extension **.udg**) or the log output file (which ends with the text **.log**). The default is 0.05.

DETAILS

The **check** spec uses residuals from the estimated model. If the **estimate** spec is absent, the **check** spec forces estimation of the model (with default estimation options).

Under the null hypothesis that the model is correct, the Ljung-Box or Box-Pierce Q-statistics are asymptotically distributed as χ^2 with degrees of freedom equal to the number of lags used in computing them less the number of AR and MA parameters estimated. The degrees of freedom are shown on the output. Ignore the Q-statistics and p-values corresponding to zero degrees of freedom.

Another diagnostic included in the `X-13ARIMA-SEATS` software is a model-based F-statistic for determining if there is stable seasonality in the original series.

name	*short*	*description of diagnostic*
alldiagnostics	all	all modeling diagnostics listed in this table
normalitytest	nrm	test results from the normality tests on the regARIMA model residuals (kurtosis, skewness and Geary's a statistics)
ljungboxq	lbq	significant lags for the Ljung-Box Q statistic
boxpierceq	bpq	significant lags for the Box-Pierce Q statistic
durbinwatson	dw	Durbin-Watson statistic for regARIMA model residuals
friedmantest	frt	Friedman non-parametric test for residual seasonality
seasftest	sft	model-based F-statistic for seasonality from Lytras, Feldpausch, and Bell (2007)
tdftest	tft	model-based F-statistic for trading day from Pang and Monsell (2016)

Name gives the name of each diagnostic for use with the **savelog** argument.
Short gives a short name for these diagnostics.

Table 7.4: **Available Log File Diagnostics for Check**

This F-test is generated from the chi-square test of groups of regressors used to determine if a particular group of regression parameters in the regARIMA model are collectively zero. The chi-square test statistic is given below:

$$\widehat{\chi}^2 = \widehat{\beta}' \left[Var(\widehat{\beta})^{-1} \right] \widehat{\beta}. \tag{7.1}$$

One such group of predefined regressors is the fixed seasonal regressors. This type of regressor can be expressed in two ways – monthly (or quarterly) indicator variables or a trigonometric representation of a fixed monthly pattern.

When these regression terms are included in the regARIMA model, the chi-square test of the seasonal regressors is produced and serves as an indication of the stable seasonality in the series.

The chi-square test of the seasonal regressors can be corrected to account for the error in the estimation of the innovation variance by using the test statistic $\tilde{\chi}^2/k$, generated as follows:

$$\frac{\tilde{\chi}^2}{k} = \frac{\widehat{\chi}^2}{k_s} \times \frac{n-d-k}{n-d} \tag{7.2}$$

where $\widehat{\chi}^2$ is the chi-squared statistic from (7.1), n is the number of observations in the series, d is the degree of differencing, k is the total number of regressors estimated in the regARIMA model, and k_s is the number of regressors for the group of seasonal regressors being tested in (7.1). The test statistic in (7.2) follows an $F(k_s, n-d-k)$ distribution.

Lytras, Feldpausch, and Bell (2007) compared the performance of the $\tilde{\chi}^2/k$ statistic to several tests for stable seasonality that are commonly used by seasonal adjustment practitioners, but whose statistical properties are unknown. The simulation studies examined led the authors to recommend the use of the $\tilde{\chi}^2/k$ statistic over more traditional diagnostics.

In the same way, when trading day regression terms are included in the regARIMA model, the chi-square test of the trading day regressors is produced and serves as an indication of trading day variation in the series.

As before, the chi-square test of the trading day regressors can be corrected to account for the error in the estimation of the innovation variance by using the test statistic $\tilde{\chi}^2/k$, generated as follows:

$$\frac{\tilde{\chi}^2}{k} = \frac{\widehat{\chi}^2}{k_{td}} \times \frac{n-d-k}{n-d} \tag{7.3}$$

where $\widehat{\chi}^2$ is the chi-squared statistic from (7.1), n is the number of observations in the series, d is the degree of differencing, k is the total number of regressors estimated in the regARIMA model, and k_{td} is the number of regressors for the group of trading day regressors being tested in (7.1). The test statistic in (7.3) follows an $F(k_{td}, n-d-k)$ distribution.

Pang and Monsell (2016) compared the performance of the $\tilde{\chi}^2/k$ statistic to several tests for trading day that are commonly used by seasonal adjustment practitioners. The simulation studies examined led the authors to recommend the use of the $\tilde{\chi}^2/k$ statistic over the the chi square test of the trading day regressors and a set of spectral diagnostics.

`X-13ARIMA-SEATS` produces three statistics that test the regARIMA model residuals for deviations from normality; one tests for skewness using the statistic:

$$c = \frac{\sqrt{n}\sum_{i=1}^{n}\left(X_i - \bar{X}\right)^3}{\left(\sum_{i=1}^{n}\left(X_i - \bar{X}\right)^2\right)^{1.5}}$$

The remaining test statistics test for different concepts of kurtosis. Both require that there be no skewness in order to be able to detect their version of kurtosis. Both types of kurtosis rule out normality, as does skewness. The more reliable one for typical time series lengths is Geary's a statistic, whose definition in Geary (1936) and Gastwirth and Owens (1977) is:

$$a = \frac{\frac{1}{n}\sum_{i=1}^{n}\left|X_i - \bar{X}\right|}{\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(X_i - \bar{X}\right)^2}}$$

where $\bar{X}$ is the sample mean. The other kurtosis statistic, whose "significance" often signals the need for additional outlier regressors, but does not provide reliable kurtosis detection except with very long series, is the sample kurtosis:

$$b_2 = \frac{n\sum_{i=1}^{n}\left(X_i - \bar{X}\right)^4}{\left(\sum_{i=1}^{n}\left(X_i - \bar{X}\right)^2\right)^2}$$

Properties of both are discussed in Section 5.14 of Snedecor and Cochran (1980).

A significant value of one of these statistics indicates that the standardized residuals do not follow a standard normal distribution. `X-13ARIMA-SEATS` tests for significance at the one percent level, from values given in tables from Pearson (1938) and Pearson and Hartley (1954). If the regARIMA model fits the data well, such lack of normality ordinarily causes no problems.

However, a significant value can occur because certain data effects are not captured well by the model. Sometimes these effects can be captured by additional or different regressors (e.g. trading day, holiday or outlier regressors). Thus, significant values can be used as a stimulus to reconsider what regressors to use.

There are other important effects that can cause a significant value, such as random variation of the coefficients or time-varying conditional variances, which cannot be represented by regARIMA models. These other effects cause the test statistics and forecast coverage intervals of `X-13ARIMA-SEATS` to have reduced reliability. Their presence is often indicated by significant values of the Ljung-Box Q-statistics of the squared residuals.

The number of lags for the ACF of the squared residual is set to be equal to seasonal period of the series (12 for monthly series, 4 for quarterly series). This value cannot be changed by the **maxlag** argument.

The use of fixed coefficients in the ARIMA model can invalidate the *DF* (degrees of freedom) values and therefore also the associated chi-square p-values in the Ljung-Box or Box-Pierce Q-statistic output of **check**. This happens when the fixed values are actually estimated values from a previous model fitting. The p-values will have the expected (approximate) validity when a statistically insignificant coefficient has been fixed at the value zero.

EXAMPLES

The following examples show complete spec files.

Example 1 Print all available diagnostic checks of the residuals from the specified model. The sample autocorrelation and partial autocorrelation function of the residuals is computed through lag 36 (the default for monthly time series). The **check** spec forces model estimation to be performed (with default options) even though the **estimate** spec is not present.

```
series { title = "Monthly Retail Sales"
         start = 1994.jan
         file = "sales1.dat" }
regression { variables = (td ao2007.jun ls2011.jun easter[14]) }
arima { model = (0 1 1)(0 1 1) }
check { print = (all) }
```

Example 2 For the same series and model as in Example 1, produce all diagnostic checking statistics excluding the printed table and plot of the residual PACF. The residual ACF is computed through lag 36.

```
series { title = "Monthly Retail Sales"
         start = 1994.jan
         file = "sales1.dat" }
regression { variables = (td ao2007.jun ls2011.jun easter[14]) }
arima { model = (0 1 1)(0 1 1) }
check { print = (all -pacf -pacfplot) maxlag = 36 }
```

Example 3 Print all available diagnostics from the check spec. Save output from the Ljung-Box statistics, normality statistics for residuals, and the trading day regression F-test statistic to the log file using the **savelog** argument. Set the limits for significance of the Ljung Box Q-statistic and t-statistics generated from individual lags of the sample residual ACFs and PACFs.

```
series{
    file = "Warehouse clubs and supercenters.dat"
    period = 12  format = Datevalue
}
transform{  function = log  }
regression{
    variables = (  td AO2000.Mar TC2001.Feb  )
}
arima{  model =  (0 1 1)(0 1 1)  }
forecast{  maxlead = 24   print = none  }
estimate{ print = (roots regcmatrix acm)
    savelog = (aicc aic bic hq afc)
}
check{ print = all  savelog = (lbq nrm tft )
       acflimit = 2.0  qlimit=0.05
}
```

Example 4 Same as Example 3, except there are seasonal regressors specified in the regression spec, and the seasonal term has been removed from the ARIMA model. Save the seasonal regression F-test statistic to the log file, as well as the other diagnostics from the last example, using the **savelog** argument.

```
series{
    file = "Warehouse clubs and supercenters.dat"
    period = 12  format = Datevalue
}
transform{  function = log  }
regression{
    variables = (  td seasonal AO2000.Mar TC2001.Feb  )
}
arima{  model =  (0 1 1)  }
forecast{  maxlead = 24   print = none  }
estimate{ print = (roots regcmatrix acm)
    savelog = (aicc aic bic hq afc)
}
check{ print = all  savelog = (lbq nrm tft sft) }
```

7.4 COMPOSITE

DESCRIPTION

This spec is used as part of the procedure for obtaining both indirect and direct adjustments of a composite series. For obtaining composite adjustments, it is one of the required spec files referenced in a metafile. Previous spec files in the metafile must define the component series and how they are combined to form the composite (see the **comptype** and **compwt** arguments of the **series** spec). This spec is used in place of the **series** spec.

The user can specify a title for the composite adjustment, a name for the composite series, which tables are to be printed or stored, and which line-printer plots are to be produced from the indirect adjustment.

USAGE

```
composite {   title = "Total one family housing starts"
              name = "hs1ft"
              decimals = 2
              modelspan = (1985.Jan,)
              appendfcst = yes
              appendbcst = no
              type = stock
              print = (brief +indtest)
              save = (indseasonal)
              savelog = (indtest)
          }
```

ARGUMENTS

appendbcst Determines if backcasts will be included in certain tables selected for storage with the save option. If `appendbcst = yes`, then backcasted values will be stored with tables a16, b1, d10, and d16 of the **x11** spec, table s10 of the **seats** spec, tables a6, a7, a8, a8.tc, a9, and a10 of the **regression** spec, and tables c16 and c18 of the **x11regression** spec. If `appendbcst = no`, no backcasts will be stored. The default is to not include backcasts.

appendfcst Determines if forecasts will be included in certain tables selected for storage with the save option. If `appendfcst = yes`, then forecasted values will be stored with tables a16, b1, d10, and d16 of the **x11** spec, tables a6, a7, a8, a8.tc, a9, and a10 of the **regression** spec, and tables c16 and c18 of the **x11regression** spec. If `appendfcst = no`, no forecasts will be stored. The default is to not include forecasts.

decimals Specifies the number of decimals that will appear in the seasonal adjustment tables of the main output file. This value must be an integer between 0 and 5, inclusive (for example, `decimals = 3`). The default number of decimals is zero.

modelspan Specifies the span (data interval) of the composite time series that is to be used to determine all regARIMA model coefficients. This argument can be utilized when, for

example, the user does not want data early in the series to affect the forecasts, or, alternatively, data late in the series to affect regression estimates used for preadjustment before seasonal adjustment. The **modelspan** argument has two values, the start and end date of the desired span. A missing value defaults to the corresponding start or end date of the composite series being analyzed. For example, for monthly data, the statement `modelspan = (1968.1,)` causes whatever regARIMA model is specified in other specs to be estimated from the time series data starting in January, 1968 and ending at the end date of the analysis span. A comma is necessary if either the start or end date is missing. The start and end dates of the model span must both lie within the time span of the composite series, and the start date must precede the end date.

Another end date specification, with the form *0.per*, is available to set the ending date of **modelspan** always to be the most recent occurrence of a specific calendar month (quarter for quarterly data) in the span of data analyzed, where *per* denotes the calendar month (quarter). If the span of data considered ends in a month other than December, `modelspan = (,0.dec)` will cause the model parameters to stay fixed at the values obtained from data ending in the next-to-final calendar year of the span.

name The name of the composite time series. The name must be enclosed in quotes and may contain up to 8 characters. It will be printed as a label on every page of printed output.

print and **save** The default output tables available for the direct and indirect seasonal adjustments generated by this spec are given in Table 7.5; other output tables available are given in Table 7.6. For a complete listing of the `brief` and `default` print levels for this spec, see Appendix B.

Table 7.7 gives table names and abbreviations that can be used with the **save** argument to save certain tables as percentages rather than ratios. Specifying these table names in the **print** argument will not change the output of the program, and the percentages are only produced when multiplicative or log-additive seasonal adjustment is specified by the user in the **mode** argument of the **x11** spec; these quantities will be expressed as differences if `mode = add`.

savelog The diagnostics available for output to the log file (see section 2.6) are listed in Table 7.8.

title A title describing the composite time series. The title must be enclosed in quotes and may contain up to 79 characters – longer text strings will be truncated to the first 79 characters. It will be printed above the data in the output.

type Indicates the type of series being aggregated. If `type = flow`, the composite series is assumed to be a flow series; if `type = stock`, the composite series is assumed to be a stock series. The default is to not assign a type to the series.

RARELY USED ARGUMENTS

indoutlier If `indoutlier = yes`, the program will attempt to generate indirect point and level shift outliers from the components of the composite adjustment. If `indoutlier = no`, no in-

name	*short*	*save?*	*description of table*
adjcompositesrs	b1	+	aggregated time series data, prior adjusted, with associated dates
compositesrs	cms	+	aggregated time series data, with associated dates
header	hdr	·	header for indirect seasonal adjustment
indadjsatot	iaa	+	final indirect seasonally adjusted series, with yearly totals adjusted to match the original series
indadjustfac	iaf	+	final combined adjustment factors for the indirect seasonal adjustment
indadjustmentratio	i18	+	indirect total adjustment factors
indcalendar	ica	+	final calendar factors for the indirect seasonal adjustment
indcalendaradjchanges	ie8	+	percent changes (differences) in the original series adjusted for calendar effects
indforcefactor	iff	+	factors applied to get indirect seasonally adjusted series with forced yearly totals
indirregular	iir	+	final irregular component for the indirect adjustment
indreplacsi	id9	+	final replacement values for extreme SI-ratios (differences) for the indirect adjustment
indresidualseasf	irf	·	F-test for residual seasonality
indrevsachanges	i6a	+	percent changes for indirect seasonally adjusted series with revised yearly totals
indrndsachanges	i6r	+	percent changes (differences) in the indirect seasonally adjusted series
indsachanges	ie6	+	percent changes (differences) in the indirect seasonally adjusted series
indsadjround	irn	+	percent changes for rounded indirect seasonally adjusted series
indseasadj	isa	+	final indirect seasonally adjusted series
indseasonal	isf	+	final seasonal factors for the indirect seasonal adjustment
indseasonaldiff	isd	+	final seasonal difference for the indirect seasonal adjustment (only for pseudo-additive seasonal adjustment)
indtest	itt	·	test for adequacy of composite adjustment
indtrend	itn	+	final trend-cycle for the indirect adjustment
indtrendchanges	ie7	+	percent changes (differences) in the indirect final trend component
indunmodsi	id8	+	final unmodified SI-ratios (differences) for the indirect adjustment
origchanges	ie5	+	percent changes (differences) in the original series

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.5: **Default Output Tables for Composite**

name	*short*	*save?*	*description of table*
adjcompositesrsplot	b1p	·	plot of the prior adjusted aggregate series
calendaradjcomposite	cac	+	aggregated time series data, adjusted for regARIMA calendar effects.
compositeplot	cmp	·	plot of the prior adjusted aggregate series
indaoutlier	iao	+	final indirect AO outliers
indftestd8	idf	·	final unmodified SI-ratios (differences) for the indirect adjustment
indirregularplot	iip	·	plot of the final irregular component from the indirect seasonal adjustment
indlevelshift	ils	+	final indirect LS outliers
indmcdmovavg	if1	+	MCD moving average of the final indirect seasonally adjusted series
indmodoriginal	ie1	+	original series modified for extreme values from the indirect seasonal adjustment
indmodsadj	ie2	+	seasonally adjusted series modified for extreme values from the indirect seasonal adjustment
indmodirr	ie3	+	irregular component modified for extreme values from the indirect seasonal adjustment
indqstat	if3	·	quality control statistics for the indirect seasonal adjustment
indrobustsa	iee	+	final indirect seasonally adjusted series modified for extreme values
indseasadjplot	iap	·	plot of the final indirect seasonally adjusted series
indseasonalplot	isp	·	indirect seasonal factor plots, grouped by month or quarter
indtotaladjustment	ita	+	total indirect adjustment factors (only produced if the original series contains values that are ≤ 0)
indtrendplot	itp	·	plot of the final trend-cycle from the indirect seasonal adjustment
indx11diag	if2	·	summary of seasonal adjustment diagnostics for the indirect seasonal adjustment
indyrtotals	ie4	·	ratio of yearly totals of the original series and the indirect seasonally adjusted series
origwindsaplot	ie0	·	plot of the aggregate series with the indirect seasonally adjusted series
outlieradjcomposite	oac	+	aggregated time series data, adjusted for outliers.
prioradjcomposite	ia3	+	composite series adjusted for user-defined prior adjustments applied at the component level
ratioplotindsa	ir2	·	month-to-month (or quarter-to-quarter) ratio plots of the original series
ratioplotorig	ir1	·	month-to-month (or quarter-to-quarter) ratio plots of the indirect seasonally adjusted series

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.6: **Other Output Tables for Composite**

name	*short*	*description of table*
origchangespct	ip5	percent changes for composite series
indsachangespct	ip6	percent changes for indirect seasonally adjusted series
indrevsachangespct	ipa	percent changes for indirect seasonally adjusted series with forced yearly totals
indrndsachangespct	ipr	percent changes for rounded indirect seasonally adjusted series
indtrendchangespct	ip7	percent changes for indirect trend component
indcalendaradjchangespct	ip8	percent changes in original series adjusted for calendar effects
indseasonalpct	ips	indirect seasonal component expressed as percentages if appropriate
indirregularpct	ipi	indirect irregular component expressed as percentages if appropriate
indadjustfacpct	ipf	indirect combined adjustment factors expressed as percentages if appropriate

Name gives the name of each table for use with the **save** argument.
Short gives a short name for these tables.

Table 7.7: **Tables Saved as Percentages in the `save` Argument of Composite**

direct outliers will be used in generating components of the indirect seasonal adjustment. The default for the program is `indoutlier = yes`.

saveprecision The number of decimals stored when saving a table to a separate file with the **save** argument, e.g., `saveprecision = 10`. The default value of **saveprecision** is 15.

yr2000 If `yr2000 = yes`, a "century cutoff" for 2-digit years from data stored in "`X-11` formats" is set at 1945. Years 00–45 are interpreted as 20xx, and years 46–99 are interpreted as 19xx. If `yr2000 = no`, the program assumes all 2-digit years fall in the 20th century and will convert them to 4-digit years accordingly. The default for the program is `yr2000 = yes`.

Note: this option is set here to affect program behavior when files are read in other specs (such as the **transform** and **x11regression** specs).

DETAILS

An input specifications file with the **composite** spec can only be used in conjunction with spec files for component series which together define a composite series. The names of these other spec files must be listed in a metafile in which the name of this spec file appears last. The **comptype** argument of the **series** spec of each component series controls how the components are combined to form the final aggregate (composite) series. (See Section 2.5 for examples of how to run metafiles).

A composite adjustment run with this metafile produces an indirect seasonal adjustment of the composite series as well as a direct seasonal adjustment. The indirect adjustment is the combination specified by the

name	*short*	*description of diagnostic*
indtest	itt	test for adequacy of composite adjustment
indm1	im1	M1 Quality Control Statistic from indirect adjustment
indm2	im2	M2 Quality Control Statistic from indirect adjustment
indm3	im3	M3 Quality Control Statistic from indirect adjustment
indm4	im4	M4 Quality Control Statistic from indirect adjustment
indm5	im5	M5 Quality Control Statistic from indirect adjustment
indm6	im6	M6 Quality Control Statistic from indirect adjustment
indm7	im7	M7 Quality Control Statistic from indirect adjustment
indm8	im8	M8 Quality Control Statistic from indirect adjustment
indm9	im9	M9 Quality Control Statistic from indirect adjustment
indm10	imt	M10 Quality Control Statistic from indirect adjustment
indm11	ime	M11 Quality Control Statistic from indirect adjustment
indq	iq	overall index of the quality of the indirect seasonal adjustment
indq2	iq2	indirect Q statistic computed without the M2 Quality Control statistic
indmovingseasratio	isr	moving seasonality ratio from indirect adjustment
indicratio	iir	$\bar{I}/\bar{C}$ ratio from indirect adjustment
indfstabled8	id8	F-test for stable seasonality, performed on the final SI-ratios from indirect adjustment
indmovingseasf	isf	F-test for moving seasonality from indirect adjustment
indidseasonal	iid	identifiable seasonality test result for indirect adjustment
alldiagnostics	all	all seasonal adjustment diagnostics listed in this table

Name gives the name of each diagnostic for use with the **savelog** argument.
Short gives a short name for these diagnostics.

Table 7.8: **Available Log File Diagnostics for Composite**

comptype of the components, each adjusted or not adjusted according to the prescriptions of their spec files. The direct adjustment is done as requested in the spec file of the composite spec. To control the output for the direct seasonal adjustment, use the **print** and **save** arguments of the **x11** spec.

To include an unadjusted series as a component of the indirect seasonal adjustment of the aggregate series, specify the summary measures option by setting `type = summary` in the **x11** spec of this component.

Although none of the tables of seasonal adjustment diagnostics produced in this spec can be saved to its own file, specifying the diagnostic summary option with the **-s** flag at runtime allows the user to store information from the composite analysis into a diagnostic summary file (with the file extension **.udg**). In addition, the **savelog** argument can write selected diagnostics into the log file for a given run ((appending `_log.html` to the base output filename). For more information, see section 2.6.

If a sliding spans analysis of the direct and indirect adjustments is desired, the sliding spans analysis option must be specified for each of the component series. If the seasonal filter length is not the same for each component, then the user must use the **length** argument of the **slidingspans** spec to ensure that the spans stored for the component series are of the same length.

When a revisions history analysis of the seasonally adjusted series is specified for a composite seasonal adjustment, the revisions of both the direct and indirect seasonal adjustments of the composite series are produced. The revisions history analysis must be specified for each of the component series.

If a series is designated as a stock or a flow series by using the **type** argument, then trading day and Easter regressors specified in **regression** spec need to agree with this type – one cannot specify stock trading day regressors for a flow series. If a series type is not specified, then any trading day or holiday regressor may be used with the series.

EXAMPLES

The following examples illustrating all the steps of a composite adjustment show complete spec files.

Step 1 A spec file must be created for each of the component series. In this example, we process each of the components (Northeast, Midwest, South and West 1-family housing starts), using a simple sum to form the composite. An example of the spec file for the Northeast series (stored in `cne1hs.spc`), which is seasonally adjusted using 3×9 seasonal filters, is given below:

```
series { title="NORTHEAST ONE-FAMILY Housing Starts"
         file="cne1hs.ori"  name="CNE1HS"  format="2R"
         comptype=add  }
x11  {  seasonalma=(s3x9)
        title=(
             "Component for Composite Adjustment"
             "of Total U.S. 1-Family Housing Starts") }
```

The seasonal adjustment of CNE1HS produced by this spec file will be an addend in the calculation of the indirect seasonal adjustment of the composite series.

A spec file for a component series that is not seasonally adjusted is given below:

```
series { title="West ONE-FAMILY Housing Starts"
         file="cwt1hs.ori"  name="CWT1HS"  format="2R"
         comptype=add   }
x11  {   type=summary   }
```

This will cause the unadjusted series stored in `cwt1hs.ori` to be an addend in the calculation of the indirect seasonal adjustment of the composite series.

Step 2 Create a spec file for the indirect adjustment of total one-family housing starts, the sum of four regional series. The direct seasonal adjustment of the series will be multiplicative and will use a 3×9 seasonal moving average. Both the seasonal factors from the direct adjustment and the implied factors from the indirect adjustment will be saved. The spec file (stored in `c1fths.spc`) appears below:

```
composite {  title="TOTAL ONE-FAMILY Housing Starts"
             name="C1FTHS" save=(indseasonal) }
x11 { seasonalma=(s3x9)
      title="Composite adj. of 1-Family housing starts"
      save=(D10) }
```

Step 3 Create a metafile for the input specification files of the component and composite series. This metafile, stored in `hs1ftot.mta` appears below:

```
cne1hs
cmw1hs
cso1hs
cwt1hs
c1frhs
```

Note that the spec file for the composite series is listed last.

Step 4 To run `X-13ARIMA-SEATS` for this example, enter the following:

x13asHTML -m hs1ftot

and press the `<return>` (`<enter>`) key.

7.5 ESTIMATE

DESCRIPTION

Estimates the regARIMA model specified by the **regression** and **arima** specs. Allows the setting of various estimation options. Estimation output includes point estimates and standard errors for all estimated AR, MA, and regression parameters; the maximum likelihood estimate of the variance σ^2; t-statistics for individual regression parameters; χ^2-statistics for assessing the joint significance of the parameters associated with certain regression effects (if included in the model); and likelihood based model selection statistics (if the exact likelihood function is used). The regression effects for which χ^2-statistics are produced include stable seasonal effects, trading-day effects, and the set of user-defined regression effects.

USAGE

```
estimate {  tol = 1.0e-5
            maxiter = (500)
            exact = arma
            outofsample = yes
            print = (none +model +estimates +lkstats)
            save = (model)
            savelog = (aic bic) }
```

ARGUMENTS

exact Specifies use of exact or conditional likelihood for estimation, likelihood evaluation, and forecasting. The default is `exact = arma`, which uses the likelihood function that is exact for both AR and MA parameters. Other options are: `exact = ma`, which uses the likelihood function that is exact for MA, but conditional for AR parameters, and `exact = none`, which uses the likelihood function that is conditional for both AR and MA parameters.

maxiter The maximum number allowed of ARMA iterations (nonlinear iterations for estimating the AR and MA parameters). For models with regression variables, this limit applies to the total number of ARMA iterations over all IGLS iterations. For models without regression variables, this is the maximum number of iterations allowed for the single set of ARMA iterations. The default is `maxiter = 1500`.

outofsample Determines the kind of forecast error used in calculating the average magnitude of forecast errors over the last three years, a diagnostic statistic. If `outofsample = yes`, out-of-sample forecasts errors are used; these are obtained by removing the data in the forecast period from the data set used to estimate the model and produce one year of forecasts (for each of the last three years of data). If `outofsample = no`, within-sample forecasts errors are used. That is, the model parameter estimates for the full series are used to generate forecasts for each of the last three years of data. The default is `outofsample = no`.

print and **save** Table 7.9 gives the default output tables for this spec; the other output tables are given in Table 7.10. For a complete listing of the `brief` and `default` print levels for this spec, see Appendix B.

savelog The diagnostics available for output to the log file (see section 2.6) are listed in Table 7.11.

tol Convergence tolerance for the nonlinear estimation. Absolute changes in the log-likelihood are compared to **tol** to check convergence of the estimation iterations. For models with regression variables, **tol** is used to check convergence of the IGLS iterations (where the regression parameters are re-estimated for each new set of AR and MA parameters), see Otto, Bell, and Burman (1987). For models without regression variables there are no IGLS iterations, and **tol** is then used to check convergence of the nonlinear iterations used to estimate the AR and MA parameters. The default value is `tol = 1.0e-5`.

name	*short*	*save?*	*description of table*
options	opt	·	header for the estimation options
model	mdl	+	if used with the **print** argument, this controls printing of a short description of the model; if used with the **save** argument, this creates a file containing **regression** and **arima** specs corresponding to the model, with the estimation results used to specify initial values for the ARMA parameters
estimates	est	+	regression and ARMA parameter estimates, with standard errors
averagefcsterr	afc	·	average magnitude of forecast errors over each of the last three years of data
lkstats	lks	+	log-likelihood at final parameter estimates and, if `exact = arma` is used (default option), corresponding model selection criteria (AIC, AICC, Hannan-Quinn, BIC)

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.9: **Default Output Tables for Estimate**

RARELY USED ARGUMENTS

file Name of the file containing the model settings of a previous `X-13ARIMA-SEATS` run. Such a file is produced by setting `save = model` or `save = mdl` in this spec. The filename must be enclosed in quotes. If the file is not in the current directory, the path must also

name	*short*	*save?*	*description of table*
iterations	itr	+	detailed output for estimation iterations, including log-likelihood values and parameters, and counts of function evaluations and iterations
iterationerrors	ite	·	error messages for estimation iterations, including failure to converge
regcmatrix	rcm	+	correlation matrix of regression parameter estimates if used with the **print** argument; covariance matrix of same if used with the **save** argument
armacmatrix	acm	+	correlation matrix of ARMA parameter estimates if used with the **print** argument; covariance matrix of same if used with the **save** argument
lformulas	lkf	·	formulas for computing the log-likelihood and model selection criteria
roots	rts	+	roots of the autoregressive and moving average operators in the estimated model
regressioneffects	ref	+	$\mathbf{X}\widehat{\beta}$, matrix of regression variables multiplied by the vector of estimated regression coefficients
regressionresiduals	rrs	+	residuals from regression effects
residuals	rsd	+	model residuals with associated dates or observation numbers

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.10: **Other Output Tables for Estimate**

name	*short*	*description of diagnostic*
alldiagnostics	all	all modeling diagnostics listed in this table
aic	aic	Akaike's Information Criterion (AIC)
aicc	acc	Akaike's Information Criterion (AIC) adjusted for the length of the series
bic	bic	Baysean Information Criterion (BIC)
hannanquinn	hq	Hannan-Quinn Information Criterion
roots	rts	roots of the autoregressive and moving average operators in the estimated model
averagefcsterr	afc	average forecast error over the last three years of data

Name gives the name of each diagnostic for use with the **savelog** argument.
Short gives a short name for these diagnostics of the **savelog** argument.

Table 7.11: **Available Log File Diagnostics for Estimate**

be given. If the **file** argument is used, the **model**, **ma**, and **ar** arguments of the **arima** spec and the **variables**, **user**, and **b** arguments of the **regression** spec cannot be used. In addition to those, neither the **pickmdl** spec nor the **automdl** spec can be used.

fix Specifies whether certain coefficients found in the model file specified in the **file** argument are to be held fixed instead of being used as initial values for further estimation. If `fix = all`, both the regression and ARMA parameter estimates will be held fixed at their values in the model file. If `fix = arma`, only ARMA parameter estimates will be held fixed at their model file values. If `fix = reg`, only the regression parameter estimates will be held fixed at their model file values. If `fix = none`, none of the parameter estimates will be held fixed. The default is `fix = nochange`, which will preserve coefficient values specified as fixed in the model file and allow re-estimation of all other coefficients.

DETAILS

The inference results provided by `X-13ARIMA-SEATS` are asymptotically valid (approximately correct for sufficiently long time series) under "standard" assumptions—see Section 4.5. The likelihood based model selection statistics are provided only if the exact likelihood function is used. See Section 5.5 for comments on the use of model selection statistics.

If the estimation iterations converge, `X-13ARIMA-SEATS` prints a message to this effect and then displays the estimation results. If the iterations fail to converge, `X-13ARIMA-SEATS` prints a message indicating this and then displays the parameter values at the last iteration. These values should not be used as parameter estimates. Instead, the program should be rerun, possibly starting at the parameter values obtained when the iterations terminated. Potential causes of convergence problems and suggested remedies are discussed in Chapter 5.

The **tol** argument should not be set either "too large" or "too small." Setting **tol** too large can result in estimates far from the true MLEs, while setting **tol** too small can result in an unnecessarily large number of iterations or lead to a false impression of the precision of the results. What is too large or too small a value for

```
Roots of ARIMA Model
 Root                  Real  Imaginary    Modulus  Frequency
 ------------------------------------------------------------
 Nonseasonal AR
  Root  1           -0.6784     0.8817     1.1125     0.3544
  Root  2           -0.6784    -0.8817     1.1125    -0.3544
 Nonseasonal MA
  Root  1           -7.4107     0.0000     7.4107     0.5000
 Seasonal MA
  Root  1            1.5583     0.0000     1.5583     0.0000
```

Table 7.12: **Example of ARMA Roots Output**

tol depends on the problem; the default value of **tol**$= 10^{-5}$ is offered as a reasonable compromise. Setting **tol** to a number less than machine precision for a double precision number (approximately 10^{-14} for PCs and Sun4 computers) results in an error, but values for **tol** that even begin to approach machine precision are certainly too small.

For models with regression variables, a second convergence tolerance is needed to determine convergence of the ARMA iterations within each IGLS iteration. This tolerance is set by the program to $100 \times$ **tol** for the first two IGLS iterations, after which it is reset to **tol**. (Since relatively large changes can be made to the regression parameters in the initial IGLS iterations, it is not worth determining the ARMA parameters within **tol** at the start.) Thus, when **tol** takes on its default value of 10^{-5}, the ARMA convergence tolerance is 10^{-3} for the first two IGLS iterations, and thereafter it is 10^{-5} (= **tol**). Also, for models with regression variables, a limit is needed for the maximum number of ARMA iterations allowed within each IGLS iteration. This limit is set to 40.

If the ARMA iterations fail to converge on a particular IGLS iteration, this is generally not a problem. The program will continue with the next IGLS iteration, and its ARMA iterations may very well converge. In fact, all that is necessary for overall convergence is that the ARMA iterations of the *last* IGLS iteration converge, and that the IGLS iterations themselves converge to the tolerance **tol** within **maxiter** total ARMA iterations.

Setting `print = roots` produces a table of roots of all the AR and MA operators of the estimated model. In addition to the roots, the table provides the modulus (magnitude) and frequency (on $[-0.5, 0.5]$) of each root. Roots with modulus greater than one lie outside the unit circle, corresponding to stationary AR or invertible MA operators. (See Section 5.4.) AR roots on or inside the unit circle (modulus ≤ 1) should occur only when the likelihood function is defined conditionally for AR parameters (`exact = ma` or `exact = none`). MA roots inside the unit circle (modulus < 1) will never occur, since invertibility is enforced in the estimation. MA roots on the unit circle (modulus $= 1$) can be estimated within round-off error, or can occur in an MA operator all of whose parameters are specified as fixed during estimation.

In sample output shown in Table 7.12, the nonseasonal AR(2) polynomial has a pair of complex conjugate roots (zeros), $z = x \pm iy$, with $x = -0.6784$ and $y = 0.8817$, whose modulus (magnitude) is $r = \sqrt{(x^2 + y^2)} = 1.1125$. Because this number is close to unity (1.000), it is worthwhile to examine the nonnegative frequency of the root, i.e. the number $\lambda \geq 0$ such that $z = re^{\pm i2\pi\lambda}$ to determine if the series may contain a deterministic periodic component. The reasoning behind this is as follows. Whenever a modeled time series has a periodic

component $f(t)$ with period $1/\lambda$, i.e. $f(t+1/\lambda) = f(t)$, then an estimated AR polynomial of sufficiently high order is likely to have a root near $e^{\pm i2\pi\lambda}$ (unless the differencing operators have $e^{\pm i2\pi\lambda}$ as a root). There are theoretical results that help to explain why this happens, but a heuristic explanation is that for the simplest functions with this period,

$$f(t) = A\cos(2\pi\lambda t + c),$$

the AR(2) polynomial $\phi(B) = \left(1 - e^{-i2\pi\lambda}B\right)\left(1 - e^{i2\pi\lambda}B\right)$, whose roots are $e^{\pm i2\pi\lambda}$, has the property that

$$\phi(B)f(t) = 0.$$

Thus this AR(2) factor can perfectly predict $f(t)$ from $f(t-1)$ and $f(t-2)$. Fitting a model with an AR operator of order 2 or higher will tend to make the AR parameters take on values so that $\phi(B)f(t) = 0$. (An AR(1) polynomial suffices when $e^{i2\pi\lambda}$ is real, i.e. when $\lambda = 0, 1/2$.) . Hence the occurrence of an AR root with modulus $r \doteq 1$ suggests the presence of an approximately periodic component in the time series.

For monthly series, the frequencies of seasonal effects are $\lambda = 1/12, 2/12, 3/12, \ldots, 6/12$ (equivalent to 0.0833, 0.1666, 0.2500, ..., 0.5000, respectively). The frequency $\lambda = 0$ is associated with trend movements, and the frequency $\lambda = 0.3482$ with trading day effects. Note that the frequency 0.3544 of the nonseasonal AR roots in the table above is very close to the trading day frequency. In fact, the time series whose model produced the table has a strong trading day component, and the automatic modeling procedure added the AR(2) factor to account for it, because there were no trading day regressors in the **regression** spec.

In the MA polynomials, near unit roots with seasonal or trend frequencies usually indicate that the MA polynomials have one or more roots in common with the differencing or seasonal differencing polynomials. The presence of such a common factor $\kappa(B)$ indicates that the time series has deterministic trend or seasonal components. More specifically, in the notation of equation (4.3) of Section 4.1 (but ignoring regressors), it means that there is a function $f(t)$ satisfying $\kappa(B)f(t) = 0$ such that the time series y_t can be modeled as

$$\phi(B)\Phi(B^s)\left(\frac{(1-B)^d(1-B^s)}{\kappa(B)^D}\right)y_t = f(t) + \left(\frac{\theta(B)\Theta(B^s)}{\kappa(B)}\right)a_t.$$

In the example table above, the model's seasonal moving average polynomial is $\Theta\left(B^{12}\right) = 1 - \Theta B^{12}$ with $\Theta = 0.6417$ so the root is $1/\Theta = 1.5583$ (the root of $\Theta(z) = 1 - \Theta z$). Experience suggests that $1/\Theta$ generally needs to be 1.10 or less before it might be appropriate to replace the model with one having only fixed seasonal effects (i.e., a model with $D = 0$ and with `variables = seasonal` in the **regression** spec).

If the nonseasonal MA polynomial has a root close to the number 1 (i.e., modulus near 1, frequency near 0), it often means that there is *overdifferencing*. That is, one should consider an alternative model with differencing order d and nonseasonal MA order q both smaller by one, and a trend constant (i.e., $f(t) = C$ above with `variables = const` in the **regression** spec) should be included in the alternative model if it has a significant t-statistic.

The use of fixed coefficients in the ARIMA model or the regression model specified in either the **regression** or **x11regression** specs can invalidate the *AIC*, *AICC*, *Hannan Quinn*, and *BIC* model selection statistics in the output. This happens when the fixed values are actually estimated values from a previous model fitting. However, the p-values will have the expected (approximate) validity when a statistically insignificant coefficient has been fixed at the value zero.

EXAMPLES

The following examples show complete spec files.

Example 1 Estimate by generalized least squares the regression coefficients in the model $(1-B)(y_t - \sum_{i=1}^{11}\beta_i M_{it}) = (1-\theta B)a_t$, where the M_{it} are regression variables for monthly fixed seasonal effects. The MA parameter θ is held fixed at the value 0.25. Model residuals are saved in a file in the current directory with the same name as the spec file, but with the extension `.rsd`.

```
series { title = "Monthly Sales" start = 1996.1
          data = (138 128 ... 297) }
regression { variables = seasonal }
arima { model = (0,1,1)   ma = (0.25f) }
estimate { save = residuals }
```

Example 2 Estimate the seasonal model $(1-\phi B)(1-B)(1-B^{12})z_t = (1-\Theta B^{12})a_t$, with **tol** set to 10^{-4}, a looser convergence criterion than the default, and decrease the maximum number of iterations allowed to 100. Since there are no regression parameters in the model, both **tol** and **maxiter** apply to the single set of nonlinear ARMA iterations used to estimate ϕ and Θ. The likelihood function used in parameter estimation is exact for MA and conditional for AR parameters. The **print** argument specifies that the likelihood and parameter values are printed for each iteration and that the roots of the estimated AR and MA operators are printed following the last iteration. These same roots will also be printed to the log file. The **save** argument will save the final regARIMA model into a file.

```
series { title = "Monthly Inventory" start = 1998.12
         data = (1209 834 ... 1002) }
transform { function = log }
regression { variables = (td ao2009.01)  }
arima { model = (1,1,0)(0,1,1) }
estimate { tol = 1e-4  maxiter = 100  exact = ma  save = mdl
 print = (iterations roots)
 savelog = roots
}
```

Example 3 Same as Example 2, except the regARIMA model estimates saved in Example 2 are used in this run via the **file** argument. All parameter estimates are fixed to the values stored in the model file.

```
series { title = "Monthly Inventory" start = 1998.12
         data = (1209 834 ... 1002) }
transform { function = log }
estimate { file = "Inven.mdl"
           fix = all }
```

Example 4 Same as Example 3, except that three additional data values are available and we wish to have the program determine if any of them are outliers. The ending date of the data span in Examples 2 and 3 is December 2009. The regARIMA model parameters are to be kept fixed at the values obtained from this data span, which were stored by Example 2.

```
series { title = "Monthly Inventory" start = 1998.12
         data = (1209 834 ... 1002 1425 901 1375) }
transform { function = log }
estimate { file = "Inven.mdl"
           fix = all  }
outlier { span=(2010.01,)  }
```

7.6 FORCE

DESCRIPTION

An optional spec for invoking options that allow users to force yearly totals of the seasonally adjusted series to equal those of the original series for convenience. Two forcing methods are available, the original modified Denton method of `X-11-ARIMA` and earlier version of `X-13ARIMA-SEATS` described in Huot (1975) and Cholette (1978), and a newer method based on the regression benchmarking method of Cholette and Dagum (1994) as adapted by Quenneville, Cholette, Huot, Chiu, and DiFonzo (2004). See also Dagum and Cholette (2006).

USAGE

```
force {   lambda = 0.0
          mode = ratio
          rho = 0.85
          round = no
          start = oct
          target = calendaradj
          type = regress
          usefcst = no
          print = (none saa)
          save = saa
      }
```

ARGUMENTS

lambda Value of the parameter λ used to determine the weight matrix C for the regression method of forcing the totals of the seasonally adjusted series. For more details, see Section 2 of Quenneville et al. (2004). Permissible values of **lambda** range from -3.0 to 3.0. The most commonly used values are 1.0, 0.5 and 0.0, while cases could also be made for using either -2, -1, or 2; other values of **lambda** are extremely unlikely. The default is `lambda = 0.0`.

mode Determines whether the ratios (`mode = ratio`) or differences (`mode = diff`) in the annual totals of the series specified in the argument **target** and the seasonally adjusted series are stored, and on what basis the forcing adjustment factors are generated. The default is `mode = ratio`.

print and **save** Table 7.13 gives the available output tables for this spec. All these tables are included in the default printout. For a complete listing of the `brief` and `default` print levels for this spec, see Appendix B.

Table 7.14 gives table names and abbreviations that can be used with the **save** argument to save certain tables as percentages rather than ratios. Specifying these table names in the **print** argument will not change the output of the program, and the percentages are only produced when multiplicative or log-additive seasonal adjustment is specified by

the user in the **mode** argument of the **x11** spec; these quantities will be expressed as differences if `mode = add`.

name	*short*	*save?*	*description of table*
seasadjtot	saa	+	final seasonally adjusted series with constrained yearly totals (if **type = regress** or **type = denton**)
saround	rnd	+	rounded final seasonally adjusted series (if **round = yes**) or the rounded final seasonally adjusted series with constrained yearly totals (if **type = regress** or **type = denton**)
revsachanges	e6a	+	percent changes (differences) in seasonally adjusted series with revised yearly totals
rndsachanges	e6r	+	percent changes (differences) in rounded seasonally adjusted series
forcefactor	ffc	+	factors applied to get seasonally adjusted series with constrained yearly totals (if **type = regress** or **type = denton**)

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.13: **Default Output Tables for Force**

name	*short*	*description of table*
revsachangespct	p6a	percent changes in seasonally adjusted series with forced yearly totals
rndsachangespct	p6r	percent changes in rounded seasonally adjusted series

Name gives the name of each plot for use with the **save** argument.
Short gives a short name for these tables.

Table 7.14: **Tables Saved as Percentages in the `save` Argument of Force**

rho Value of the AR(1) parameter (ρ) used in the regression method of forcing the totals of the seasonally adjusted series. Admissible values of ρ must be between 0 and 1, inclusive. If $\rho = 1$, the modified Denton method is used. The default for this argument is 0.9 for monthly series, 0.729 ($= (0.9)^3$)for quarterly series. For more details, see Section 2 of Quenneville et al. (2004).

round When `round = yes`, the program will adjust the seasonally adjusted values for each calendar year so that the sum of the rounded seasonally adjusted series for any year will equal the rounded annual total; otherwise, the seasonally adjusted values will not be rounded. The default is `round = no`.

start This option sets the beginning of the yearly benchmark period over which the seasonally adjusted series will be forced to sum to the total. If **start** is not used, then the year is assumed to be the calendar year for the procedure invoked by setting `type = denton`

or `type = regress`. An alternate starting period can be specified for the year (such as the start of a fiscal year), however, by assigning to **start** the month (either the full name of the month or the abbreviations shown in Section 3.3) or quarter (**q1** for the first quarter, **q2** for the second quarter, etc.) of the beginning of the desired yearly benchmarking period. For example, to specify a fiscal year that starts in October and ends in September, set `start = october` or `start = oct`. To specify a fiscal year that starts in the third quarter of one year and ends in the second quarter of the next, set `start = q3`.

target Specifies which series is used as the target for forcing the totals of the seasonally adjusted series. The choices for this argument are given in Table 7.15. The default for this argument is `target = original`.

name	*description of option*
original	original series
calendaradj	calendar adjusted series
permprioradj	original series adjusted for permanent prior adjustment factors
both	original series adjusted for calendar and permanent prior adjustment factors

Table 7.15: **Choices for the `target` Argument of Force**

type Specifies options that allow the seasonally adjusted series be modified to force the yearly totals of the seasonally adjusted series and the series specified in the **target** argument to be the same. By default (`type = none`), the program will not modify the seasonally adjusted values.

When `type = denton`, the differences between the annual totals are distributed over the seasonally adjusted values in a way that tries to preserve the month-to-month (or quarter-to-quarter) movements of the original series for an additive seasonal adjustment, and tries to keep the ratio of the forced and unforced values constant for multiplicative adjustments. For more details see Huot (1975) and Cholette (1978).

When `type = regress`, a regression-based solution of Cholette and Dagum (1994) to the problem of benchmarking seasonally adjusted series is used. For more details see Quenneville et al. (2004).

These forcing procedures are not recommended if the seasonal pattern is changing or if trading day adjustment is performed; see DETAILS.

usefcst Determines if forecasts are appended to the series processed by the benchmarking routines used to force the yearly totals of the seasonally adjusted series. If `usefcst = yes`, then forecasts are used to extend the series in the forcing procedure; if `usefcst = no`, then forecasts are not used. The default is `usefcst = yes`.

RARELY USED ARGUMENTS

indforce Determines how the indirect seasonally adjusted series with forced yearly total is generated. If `indforce = yes`, the indirect seasonally adjusted series will be modified so that their yearly totals match those of the target series. If `indforce = no`, the seasonally adjusted series with forced yearly totals will be combined for each of the component series to form the indirect seasonally adjusted series with forced yearly totals. The default for this option is `indforce = yes`.

DETAILS

Let $X = (X_1, \ldots, X_T)'$ denote the vector of values X_t whose N annual totals within the time span $1 \leq t \leq T$ define the constraints, and let $A = (A_1, \ldots, A_T)'$ denote the vector of adjusted values A_t which are to be modified to have the same annual totals as the X_t. (For example, the X_t could be trading day adjusted values of an observed time series.) For the method `type = regress`, with specified values of **lambda** (λ) and **rho** (ρ), the vector of forced values $\tilde{A} = \left(\tilde{A}_1, \ldots, \tilde{A}_T\right)'$ satisfying the constraints is given by

$$\tilde{A} = A + CPCJ' \left(JCPCJ'\right)^{-1} (JX - JA), \tag{7.4}$$

where C is a $T \times T$ diagonal matrix whose diagonal is proportional to $\left(|A_1|^{\lambda}, \ldots, |A_T|^{\lambda}\right)$, where $P = \left[\rho^{|i-j|}\right]_{1 \leq i,j \leq T}$, and where J is an $N \times T$ matrix of zeros and ones such that JX and $J\tilde{A}$ are the vectors of annual totals defining the forcing constraint,

$$J\tilde{A} = JX, \tag{7.5}$$

see Quenneville et al. (2004), where it is shown that the right hand side of (7.4) minimizes

$$\left(\tilde{A} - A\right)' C^{-1} P^{-1} C^{-1} \left(\tilde{A} - A\right)$$

subject to (7.5). As this reference further explains, formulas for Denton's method are obtained from (7.4) by letting $\rho \to 1$.[16] When $\lambda = 0$, i.e. when C is the identity matrix with diagonal entries equal to one, this yields a vector $\tilde{A}$ whose entries minimize

$$\begin{aligned} f_{add}\left(\tilde{A}\right) &= \sum_{t=2}^{T} \left\{ \left(\tilde{A}_t - A_t\right) - \left(\tilde{A}_{t-1} - A_{t-1}\right) \right\}^2 \\ &= \sum_{t=2}^{T} \left\{ \left(\tilde{A}_t - \tilde{A}_{t-1}\right) - \left(A_t - A_{t-1}\right) \right\}^2, \end{aligned}$$

subject to (7.5). $f_{add}\left(\tilde{A}\right)$ is the objective function of Denton's additive method associated with `type = denton` and `mode = add`. The first expression on the right hand side shows that this method attempts to keep the changes

[16] When $\rho = 1$, (7.4) cannot be used because P becomes singular, and another equation given in Cholette (1984) is used instead. This equation involves the inversion of a $(T + N) \times (T + N)$ matrix whereas (7.4) involves the inversion of an $N \times N$ matrix. Consequently, users might observe an increase in computing time when using $\rho = 1$. An alternative to using $\rho = 1$ is to use $\rho = 0.9999$.

$\tilde{A}_t - A_t$ due to forcing constant over time, whereas the second offers the more appealing interpretation that the method attempts to have the forced values $\tilde{A}_t$ preserve the changes in the series A_t from one observation time to the next.

Similarly, when the diagonal entries of C coincide with the entries of A, corresponding to the case $\lambda = 1$ when $A_t \geq 0$, $1 \leq t \leq T$, then letting $\rho \to 1$ in (7.4) yields a vector $\tilde{A}$ whose entries minimize

$$f_{ratio}\left(\tilde{A}\right) = \sum_{t=2}^{T} \left(\frac{\tilde{A}_t}{A_t} - \frac{\tilde{A}_{t-1}}{A_{t-1}} \right)^2 \tag{7.6}$$
$$= \sum_{t=2}^{T} \left(\frac{\tilde{A}_{t-1}}{A_t} \right)^2 \left(\frac{\tilde{A}_t - \tilde{A}_{t-1}}{\tilde{A}_{t-1}} - \frac{A_t - A_{t-1}}{A_{t-1}} \right)^2,$$

subject to (7.5). $f_{ratio}\left(\tilde{A}\right)$ is the objective function of Denton's proportional method associated with `type = Denton` and `mode = ratio`. The first expression on the right shows that this method attempts to keep the ratios of forced to unforced values constant. However, the final expression shows that this method is not one which attempts to have the forced values $\tilde{A}_t$ preserve the percent changes $100\left(A_t - A_{t-1}\right)/A_{t-1}$ in the series A_t from one observation time to the next in any easily understood sense.[17] (These percent changes are often the most important product of multiplicative seasonal adjustment.) For all times t after the last complete year, minimization of (7.6) subject to (7.5) yields a "carry forward" factor c such that $\tilde{A}_t = cA_t$. As Quenneville et al. (2004) discuss, this can lead to large revisions in the $\tilde{A}_t$ at the end when another full year of data X_t becomes available to provide an additional forcing constraint. The recommended solution is, with $\lambda = 1$, to choose a value of ρ somewhat less than one. This causes the ratios $\tilde{A}_t/A_t$ in an incomplete year to decay effectively geometrically in ρ as t advances beyond the year of the last forcing constraint. This can lead to $\tilde{A}_t$ similar to those obtained from Denton's proportional method within years that have constraints but with smaller revisions to forced values in incomplete years as additional data become available. Hood (2005) presents comparison results for the regression method with various choices of λ and ρ and for other forcing methods.

Forcing causes

$$X_t + X_{t+1} + \cdots + X_{t+11} = A_t + A_{t+1} + \cdots + A_{t+11} \tag{7.7}$$

to hold when month t is the first month of a calendar year or specified fiscal year (for which data through X_{t+11} are available). The rationale usually given for forcing is the naive idea that "seasonal adjustment redistributes the seasonal effects throughout the year." To indicate the problematic character of this rationale for forcing, consider the situation in which different seasonal adjusters of a series can have any of the twelve calendar months as the starting months of their fiscal years, with the result that (7.7) is implicitly assumed to hold for all months t. (The widely used seasonal adjustment methods, including model-based methods that are mean square optimal if the model is correct, do not specially treat values of t associated with the beginning of the year.) We show that this assumption can hold for additive or multiplicative seasonal decompositions of the series X_t if and only if the series has an additive seasonal decomposition

$$X_t = S_t + A_t \tag{7.8}$$

[17] For the latter, the objective function $f_{CT}\left(\tilde{A}\right) = \sum_{t=2}^{T}\left(\tilde{A}_t/\tilde{A}_{t-1} - A_t/A_{t-1}\right)^2$ of Causey and Trager (1982) would be required. This function is not a quadratic in the $\tilde{A}_t$, so nonlinear methods are required for its minimization.

with perfectly repetitive seasonal effects, e.g. $S_t = S_{t+12}$ for monthly data. Indeed, in this situation, annual sums of the seasonal effects are constant,

$$S_t + S_{t+1} + \cdots + S_{t+11} = S_{t+1} + \cdots + S_{t+11} + S_{t+12} = \cdots , \tag{7.9}$$

and additive seasonal adjustment procedures produce values of S_t for which these sums are zero,

$$S_t + S_{t+1} + \cdots + S_{t+11} = 0, \tag{7.10}$$

for all t (because a nonzero constant component belongs to the level component of the series included in A_t). For additive decompositions (7.8), (7.10) is equivalent to (7.7). Conversely, when (7.10) holds for all t, it is clear from (7.9) that there is perfect repetition of the seasonal effects, i.e., $S_t = S_{t+12}$.

Now we show that if (7.7) holds for all t for a multiplicative decomposition

$$X_t = S_t^* A_t, \tag{7.11}$$

then X_t has an additive decomposition with perfectly repetitive seasonal effects and therefore is a series for which the multiplicative decomposition obscures the simplicity of the seasonality. Indeed, if we define $S_t = (S_t^* - 1)\, A_t$, then (7.11) can be rewritten as (7.8), so (7.7) implies that $S_t + S_{t+1} + \cdots + S_{t+11} = 0$, from which $S_t = S_{t+12}$ follows as before.

Forcing does not produce a perfectly stable seasonal pattern because only calendar or fiscal year totals are forced, not all twelve month sums. But the preceding discussion shows that the situation of perfectly stable additive seasonal patterns is one in which equality of annual totals of adjusted and unadjusted is to be expected. In particular, there is no conceptual justification for forcing when the seasonal pattern is evolving from one year to the next. (And since trading day patterns always change from one year to the next, there is no conceptual justification for forcing to unadjusted annual totals of series that have trading day effects.) There are practical justifications for forcing in certain contexts, such as the complex production situation of national accounts.

Forcing the seasonally adjusted totals to be the same as the original series annual totals can degrade the quality of the seasonal adjustment, especially when the seasonal pattern is undergoing change. It is not natural if trading day adjustment is performed because the aggregate trading day effect over a year is variable and moderately different from zero.

EXAMPLES

Example 1 A multiplicative monthly seasonal adjustment is to be performed with 3×9 seasonal moving averages for all months using ARIMA forecast extension of length 12 months, if one of the default model types is accepted. The fiscal yearly totals for the seasonally adjusted series will be forced to equal the totals of the original series for a fiscal year starting in October.

```
SERIES  { TITLE="EXPORTS OF TRUCK PARTS" START =1967.1
          FILE = "X21109.ORI"   }
PICKMDL {   }
X11  { SEASONALMA = S3X9    }
FORCE  {  START = OCTOBER   }
```

Example 2 The same as Example 1, except that the regression-based solution of Cholette and Dagum (1994) as adapted by Quenneville et al. (2004) is used.

```
SERIES  { TITLE="EXPORTS OF TRUCK PARTS" START =1967.1
          FILE = "X21109.ORI"   }
PICKMDL {   }
X11  { SEASONALMA = S3X9    }
FORCE  {  START = OCTOBER
          TYPE = REGRESS
          RHO = 0.8
}
```

Example 3 Revise the seasonally adjusted series so that the sum of the rounded seasonally adjusted series for any year will equal the rounded annual total; perform no other forcing of the series.

```
Series  { Title="Imports Of Truck Engines" Start =1967.1
          File = "I21110.Ori"   }
Pickmdl {   }
X11  { Seasonalma = S3X5    }
Force  {  Type = None
          Round = Yes
          Save = Rnd
}
```

7.7 FORECAST

DESCRIPTION

Specification to forecast and/or backcast the time series given in the **series** spec using the estimated model. The output contains point forecasts and forecast standard errors for the transformed series, and point forecasts and prediction intervals for the original series.

USAGE

```
forecast {  maxlead = 24
            maxback = 12
            probability = 0.95
            exclude = 10
            lognormal = yes
            print = (none +transformedbcast +transformed)
            save = (variances) }
```

ARGUMENTS

exclude Number of observations excluded from the end of the series (or from the end of the span specified by the **span** argument of the **series** spec, if present) before forecasting. The default is to start forecasting from the end of the series (or span), i.e., `exclude = 0`.

lognormal Determines if an adjustment is made to the forecasts when a log transformation is specified by the user to reflect that the forecasts are generated from a log-normal distribution (`lognormal = yes`). The default is not to make such an adjustment (`lognormal = no`).

maxback Number of backcasts produced. The default is 0 and 120 is the maximum. (The limit of 120 can be changed—see Section 2.8.) **Note:** Backcasts are not produced when SEATS seasonal adjustments are specified, or if the starting date specified in the **modelspan** argument of the **series** spec is not the same as the starting date of the analysis span specified in the **span** argument of the **series** spec.

maxlead Number of forecasts produced. The default is one year of forecasts (unless a SEATS seasonal adjustment is requested – then the default is three years of forecasts), and 120 is the maximum. (The limit of 120 can be changed—see Section 2.8.)

print and **save** The optional output tables are listed on Table 7.16. The `transformed` and `forecasts` tables are printed out by default. For a complete listing of the `default` and `brief` print levels for this table, see Appendix B.

probability Coverage probability for prediction intervals, assuming normality. The default is `probability = 0.95`, in which case prediction intervals on the transformed scale are *point forecast* $\pm$ 1.96 $\times$ *forecast standard error*.

name	*short*	*save?*	*description of table*
transformed	ftr	+	forecasts on the transformed scale, with corresponding forecast standard errors
variances	fvr	+	forecast error variances on the transformed scale, showing the contributions of the error assuming the model is completely known (stochastic variance) and the error due to estimating any regression parameters (error in estimating AR and MA parameters is ignored)
forecasts	fct	+	point forecasts on the original scale, along with upper and lower prediction interval limits
transformedbcst	btr	+	backcasts on the transformed scale, with corresponding forecast standard errors
backcasts	bct	+	point backcasts on the original scale, along with upper and lower prediction interval limits

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.16: **Available Output Tables for Forecast**

DETAILS

Forecasting is done with the estimated (or evaluated) model. If the **estimate** spec is not present, the **forecast** spec will force estimation (with default options) to be performed before forecasting. The model used for forecasting is the one specified by the **regression** and **arima** specs. If the **outlier** spec is present, the model is augmented by additional regression variables for any automatically identified outliers. Detected outliers can affect forecasts indirectly, through their effect on model parameter estimates, as well as directly, when outliers found near the end of the series affect the computation of the forecasts.

If the model includes one or more moving average operators then the forecasts will depend on the residuals from the estimated model. The **exact** argument of the **estimate** spec determines whether these are computed corresponding to exact likelihood (the default) or to a form of conditional likelihood.

Forecast standard errors include an adjustment for error arising from estimation of any regression parameters in the model, but do not include an adjustment for error arising from estimation of AR and MA parameters; see formula (12.42) of Bell (2004) and, for a general approach, Kohn and Ansley (1985).

If the model contains user-defined regression variables, values for these must be provided for all time points in the forecast period.[18]

Prediction intervals on the transformed scale are defined as

$$\textit{point forecast} \pm K \times \textit{forecast standard error},$$

[18]See the end of Section 4.7 for a discussion of what to do about forecast extension for seasonal adjustment of a series with a model that contains user-defined regressors whose future values are unknown.

where K denotes the standard error multiplier (from a table of the normal distribution) corresponding to the specified coverage probability. Point forecasts and prediction interval limits on the original scale are obtained by inverse transformation of those on the transformed scale, allowing for both transformation (Box-Cox or logistic) and prior adjustment factors (including the length-of-month or length-of-quarter adjustment implied if `variables = td` is included in the **regression** spec). If the **transform** spec includes user-defined prior adjustment factors, these must be provided through the forecast period for the results to be inverse transformed. If they are not provided through the forecast period, then they will be assumed to be 1 in the forecast period. In this case, effects of the user-defined adjustments on the forecasts will not be (and cannot be) undone.

A reason for using **exclude** > 0 is to produce forecasts for some time points whose data are withheld for purposes of evaluating the forecast performance of the model. `X-13ARIMA-SEATS` facilitates such comparisons by printing actual forecast errors (*observation − point forecast*) at all time points in the forecast period for which corresponding (transformed) observed data exist. Setting **exclude** > 0 produces *within-sample* comparisons, since the data that are withheld from forecasting are not withheld from model estimation. More realistic *out-of-sample* forecast comparisons are produced by withholding data from both model estimation and forecasting, which can be accomplished by using the **span** argument of the **series** spec. (See Example 4.)

Whenever forecasts and/or backcasts are generated in an `X-13ARIMA-SEATS` run in which seasonal adjustment is performed, they are appended to the original series, and the seasonal adjustment procedures are applied to the forecast and/or backcast extended series. If a seasonal adjustment is specified in a run in which a regARIMA model is used but the **forecast** spec is not, one year of forecasts are generated from the model. The only way to specify a seasonal adjustment without forecast extension is to set `maxlead = 0`.

If preadjustments for regARIMA estimated trading day, outlier, holiday or user-defined regression effects are prior adjusted from the original series, they are also adjusted out of the forecasts and backcasts.

Warning: if seasonal adjustment is specified by the **x11** spec, **exclude** cannot be used to exclude observations from the end of the series. In case it is used, **exclude** will be set to zero, and a warning message will be printed.

EXAMPLES

The following examples show complete spec files.

Example 1 Forecast up through 12 steps ahead from the end of a monthly time series, and produce 95 percent prediction intervals. These are all default options. Although the **estimate** spec is absent, the presence of the **forecast** spec forces model estimation with default estimation options. The point forecasts and prediction interval limits for the transformed series are exponentiated and then multiplied by $m_t/\bar{m}$ (to undo the length-of-month adjustment produced by `variables = td` in the **regression** spec) to convert them back to the original scale.

```
SERIES { TITLE = "Monthly sales"  START = 1996.JAN
         DATA = (138 128 ... 297) }
TRANSFORM { FUNCTION = LOG }
REGRESSION { VARIABLES = TD }
ARIMA {  MODEL = (0 1 1)(0 1 1)12 }
FORECAST { }
```

Example 2 Forecast up through 24 steps ahead from the end of the same series used in Example 1. Since the **outlier** spec is present, the estimated model used in forecasting will include any AO or LS outliers detected, in addition to the trading-day variables specified by the **regression** spec.

```
Series { Title = "Monthly Sales"  Start = 1996.jan
         Data = (138 128 ... 297) }
Transform { Function = Log}
Regression { Variables = Td }
Arima { Model = (0 1 1)(0 1 1)12 }
Estimate { }
Outlier { }
Forecast { Maxlead = 24 }
```

Example 3 Exclude 10 data points and forecast up through 15 steps ahead. The entire time series is used for parameter estimation, including the ten data points excluded at the end of the series when forecasting. For these last 10 data points, the *within-sample* forecast errors will be printed. At each forecast lead the prediction interval limits are obtained by exponentiating the point forecast on the log scale plus and minus 1.645 times the corresponding log forecast standard error, which corresponds to the requested 90 percent coverage probability.

```
series { title = "Monthly sales"  start = 1996.jan
          data = (138 128 ... 297) }
transform { function = log }
regression { variables = td }
arima { model = (0 1 1)(0 1 1)12 }
estimate { }
forecast { maxlead = 15
           probability = .90
           exclude = 10 }
```

Example 4 The series ends in March 2012, but the last 24 observations are excluded from model estimation by using a **span** argument in the **series** spec. Then, using the model with these parameter estimates, the last 24 observations are forecast from March 2010, the end of the span. The *out-of-sample* errors in forecasting the last 24 observations will be printed out. (Contrast this with Example 3.)

```
series { title = "Monthly sales"  start = 1996.jan
          data = (138 128 ... 297)
          span = ( ,2010.mar) }
transform { function = log}
regression { variables = td }
arima { model = (0 1 1)(0 1 1)12 }
estimate { }
forecast { maxlead = 24 }
```

Example 5 Forecast up through 12 months ahead from the end of a monthly time series, and produce 95 percent prediction intervals These are all default options. Also produce 12 backcasts of the series, and perform a default multiplicative seasonal adjustment of the forecast- and backcast-extended original series, prior adjusted for trading day effects.

```
series { title = "monthly sales"  start = 2000.jan
          file = "ussales.dat"     }
transform { function = log }
regression { variables = td }
arima { model = (0 1 1)(0 1 1)12 }
forecast {  maxback=12  }
x11{   }
```

Example 6 Same as Example 2, except that a log-normal correction will be applied to the forecasts.

```
Series { Title = "Monthly Sales"  Start = 1996.jan
         Data = (138 128 ... 297) }
Transform { Function = Log}
Regression { Variables = Td }
Arima { Model = (0 1 1)(0 1 1)12 }
Estimate { }
Outlier { }
Forecast { Maxlead = 24  Lognormal = Yes }
```

7.8 HISTORY

DESCRIPTION

Optional spec for requesting a sequence of runs from a sequence of truncated versions of the time series for the purpose of creating historical records of (*i*) revisions from initial (concurrent or projected) seasonal adjustments, (*ii*) out-of-sample forecast errors, and (*iii*) likelihood statistics. The user can specify the beginning date of the historical record and the choice of records (*i*) – (*iii*). If forecast errors are chosen, the user can specify a vector of forecast leads. **Warning:** Generating the history analysis can substantially increase the program's run time.

USAGE

```
history {   estimates = (sadj fcst trend)
            sadjlags = (1, 2, 3, 12)
            trendlags = (1, 2, 3)
            target = final
            start = 2005.jan
            fstep = (1 2)
            fixmdl = no
            fixreg = outlier
            endtable = 2017.Jan
            print = (all -revvalsa)
            save = (sar trr fcsterrors)
            savelog = (aveabsrevsa aveabsrevtrend)
        }
```

ARGUMENTS

endtable Specifies the final date of the output tables of the revisions history analysis of seasonal adjustment and trend estimates and their period-to-period changes. This can be used to ensure that the revisions history analysis summary statistics are based only on final (or nearly final) seasonal adjustments or trends. If **endtable** is not assigned a value, it is set to the date of the observation immediately before the end of the series or to a value one greater than the largest lag specified in **sadjlags** or **trendlags**. This option has no effect on the historical analysis of forecasts and likelihood estimates. Example: `endtable = 1990.jun`.

estimates Determines which estimates from the regARIMA modeling and/or the seasonal adjustment will be analyzed in the history analysis. Example: `estimates = (sadj aic)`. The default is the seasonally adjusted series (`sadj`). Table 7.17 gives a description of the available estimates.

fixmdl Specifies whether the regARIMA model will be re-estimated during the history analysis. If `fixmdl = yes`, the ARIMA parameters and regression coefficients of the regARIMA model will be fixed throughout the analysis at the values estimated from the entire series

name	*description of option*
sadj	final seasonally adjusted series (and indirect seasonally adjusted series, if composite seasonal adjustment is performed
sadjchng	month-to-month (or quarter-to-quarter) changes in the final seasonally adjusted series
trend	final Henderson trend component
trendchng	month-to-month (or quarter-to-quarter) changes in the final Henderson trend component
seasonal	final and projected seasonal factors
aic	AICCs and maximum log likelihoods for the regARIMA model
fcst	forecasts and evolving mean square forecast errors generated from the regARIMA model. **Warning:** This option can be used only when forecasts are produced, see the **forecast** spec in Section 7.7.
arma	estimated AR and MA coefficients from the regARIMA model
td	trading day regression coefficients from the regARIMA model. **Warning:** This option can be used only when trading day regressors are specified, see the **regression** spec in Section 7.13.

Table 7.17: **Choices for the `estimates` Argument of History**

(or model span, if one is specified via the **modelspan** argument). If `fixmdl = no`, the regARIMA model parameters will be re-estimated each time the end point of the data is changed. The default is `fixmdl = no`. This argument is ignored if no regARIMA model is fit to the series.

fixreg Specifies the fixing of the coefficients of a regressor group, either within a regARIMA model or an irregular component regression. These coefficients will be fixed at the values obtained from the model span (implicit or explicitly) indicated in the series or composite spec. All other coefficients will be re-estimated for each history span. Trading day (**td**), holiday (**holiday**), outlier (**outlier**), or other user-defined (**user**) regression effects can be fixed. This argument is ignored if neither a regARIMA model nor an irregular component regression model is fit to the series, or if `fixmdl = yes`.

fstep Specifies a vector of up to four (4) forecast leads that will be analyzed in the history analysis of forecast errors. Example: `fstep=(1 2 12)` will produce an error analysis for the 1-step, 2-step, and 12-step ahead forecasts. The default is (1 12) for monthly series or (1 4) for quarterly series. **Warning:** The values given in this vector cannot exceed the specified value of the **maxlead** argument of the **forecast** spec, or be less than one.

print and **save** The default output tables available for the direct and indirect seasonal adjustments generated by this spec are given in Table 7.18; other output tables available are given in Table 7.19. For a complete listing of the `brief` and `default` print levels for this spec, see Appendix B.

sadjlags Specifies a vector of up to 5 revision lags (each greater than zero) that will be analyzed in the revisions analysis of lagged seasonal adjustments. The calculated revisions for these revision lags will be those of the seasonal adjustments obtained using this many

name	*short*	*save?*	*description of table*
header	hdr	·	header for history analysis
outlierhistory	rot	+	record of outliers removed and kept for the revisions history (printed only if automatic outlier identification is used)
sarevisions	sar	+	revision from concurrent to most recent estimate of the seasonally adjusted data
sasummary	sas	·	summary statistics for seasonal adjustment revisions
chngrevisions	chr	+	revision from concurrent to most recent estimate of the month-to-month (or quarter-to-quarter) changes in the seasonally adjusted data
chngsummary	chs	·	summary statistics for revisions in the month-to-month (or quarter-to-quarter) changes in the seasonally adjusted data
indsarevisions	iar	+	revision from concurrent to most recent estimate of the indirect seasonally adjusted series
indsasummary	ias	·	summary statistics for indirect seasonal adjustment revisions
trendrevisions	trr	+	revision from concurrent to most recent estimate of the trend component
trendsummary	trs	·	summary statistics for trend component revisions
trendchngrevisions	tcr	+	revision from concurrent to most recent estimate of the month-to-month (or quarter-to-quarter) changes in the trend component
trendchngsummary	tcs	·	summary statistics for revisions in the month-to-month (or quarter-to-quarter) changes in the trend component
sfrevisions	sfr	+	revision from concurrent to most recent estimate of the seasonal factor, as well as projected seasonal factors
sfsummary	sfs	·	summary statistics for seasonal factor revisions
lkhdhistory	lkh	+	history of AICC and likelihood values
fcsterrors	fce	+	revision history of the accumulated sum of squared forecast errors
armahistory	amh	+	history of estimated AR and MA coefficients from the regARIMA model
tdhistory	tdh	+	history of estimated trading day regression coefficients from the regARIMA model
seatsmdlhistory	smh	+	SEATS ARIMA model history

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.18: **Default Output Tables for History**

name	*short*	*save?*	*description of table*
sfilterhistory	sfh	+	record of seasonal filter selection for each observation in the revisions history (printed only if automatic seasonal filter selection is used)
saestimates	sae	+	concurrent and most recent estimate of the seasonally adjusted data
chngestimates	che	+	concurrent and most recent estimate of the month-to-month (or quarter-to-quarter) changes in the seasonally adjusted data
indsaestimates	iae	+	concurrent and most recent estimate of the indirect seasonally adjusted data
trendestimates	tre	+	concurrent and most recent estimate of the trend component
trendchngestimates	tce	+	concurrent and most recent estimate of the month-to-month (or quarter-to-quarter) changes in the trend component
sfestimates	sfe	+	concurrent and most recent estimate of the seasonal factors and projected seasonal factors
fcsthistory	fch	+	listing of the forecast and forecast errors used to generate accumulated sum of squared forecast errors

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.19: **Other Output Tables for History**

observations beyond the time point of the adjustment. That is, for each value revisionlag$_i$ given in **sadjlags**, series values through time t + revisionlag$_i$ will be used to obtain the adjustment for time t whose revision will be calculated. For more information, see DETAILS.

This option is meaningful only if the revisions history of the seasonally adjusted series or month-to-month (quarter-to-quarter) changes in the seasonally adjusted series is specified in the **estimates** argument. The default is no analysis of revisions of lagged seasonal adjustments.

savelog The diagnostics available for output to the log file (see Section 2.6) are listed in Table 7.20.

start Specifies the starting date of the revisions history analysis. If this argument is not used, its default setting depends on the length of the longest seasonal filter used, provided that a seasonal adjustment is being performed (if there is no conflict with the requirement that 60 earlier observations be available when a regARIMA model is estimated and `fixmdl = no`, the default for **fixmdl**). The default starting date is six (6) years after the start of the series, if the longest filter is either a 3×3 or stable filter, eight (8) years for a

name	*short*	*description of diagnostic*
alldiagnostics	all	all revisions diagnostics listed in this table
aveabsrevsa	asa	average absolute revision of the seasonally adjusted series
aveabsrevchng	ach	average absolute revision of the month-to-month (or quarter-to-quarter) changes in the seasonally adjusted data
aveabsrevindsa	iaa	average absolute revision of the indirect seasonally adjusted series
aveabsrevtrend	atr	average absolute revision of the final trend component
aveabsrevtrendchng	atc	average absolute revision of the month-to-month (or quarter-to-quarter) changes in the trend component
aveabsrevsf	asf	average absolute revision of the final seasonal factors
aveabsrevsfproj	asp	average absolute revision of the projected seasonal factors
avesumsqfcsterr	afe	average sum of squared forecast error for each forecast lag

Name gives the name of each diagnostic for use with the **savelog** argument.
Short gives a short name for these diagnostics.

Table 7.20: **Available Log File Diagnostics for History**

3×5 filter, and twelve (12) years for a 3×9 filter. If no seasonal adjustment is done, the default is 8 years after the start of the series. Example: `start = 1990.jun`.

target Specifies whether the deviation from the concurrent estimate or the deviation from the final estimate defines the revisions of the seasonal adjustments and trends calculated at the lags specified in **sadjlags** or **trendlags**. The default is `target = final`; the alternative is `target = concurrent`.

trendlags Similar to **sadjlags**, this argument prescribes which lags will be used in the revisions history of the lagged trend components. Up to 5 integer lags greater than zero can be specified.

This option is meaningful only if the revisions history of the final trend component or month-to-month (quarter-to-quarter) changes in the final trend component is specified in the **estimates** argument. The default is no analysis of revisions lagged trend estimates.

RARELY USED ARGUMENTS

fixx11reg Specifies whether the irregular component regression model specified in the **x11regression** spec will be re-estimated during the history analysis. If `fixx11reg = yes`, the regression coefficients for the irregular component regression model are fixed throughout the analysis at the values estimated from the entire series. If `fixx11reg = no`, the irregular component regression model parameters will be re-estimated each time the end point of the history interval is advanced.

The default is `fixx11reg = no`. This argument is ignored if no irregular component regression model is specified.

outlier Specifies whether automatic outlier detection is to be performed whenever the regARIMA model is re-estimated during the revisions history analysis. This argument has no effect if the **outlier** spec is not used.

If `outlier = keep`, all outliers automatically identified using the full series are kept in the regARIMA model during the revisions history analysis. The coefficients estimating the effects of these outliers are re-estimated whenever the other regARIMA model parameters are re-estimated. No additional outliers are automatically identified and estimated. This is the default setting.

If `outlier = remove`, those outlier regressors that were added to the regression part of the regARIMA model when automatic outlier identification was performed on the full series are removed from the regARIMA model during the revisions history analysis. Consequently, their effects are not estimated and removed from the series. This option gives the user a way to investigate the consequences of not doing automatic outlier identification.

If `outlier = auto`, among outliers automatically identified for the full series, only those that fall in the time period up to **outlierwin** observations before the starting date of the revisions history analysis are automatically included in the regARIMA model. In each run of the estimation procedure with a truncated version of the original series, automatic outlier identification is performed only for the last **outlierwin**+1 observations. An outlier that is identified is used for the current run, but is only retained for the subsequent runs of the historical analysis if it is at least **outlierwin** observations from the end of the subsequent span of data being analyzed.

outlierwin Specifies how many observations before the end of each span will be used for outlier identification during the revisions history analysis. The default is 12 for monthly series or 4 for quarterly series. This argument has an effect only if the **outlier** spec is used, and if `outlier = auto` in the **history** spec.

refresh Specifies which of two sets of initial values is used for the regARIMA model parameter estimation. If `refresh = yes`, the parameter estimates from the last model evaluation are used as starting values for the current regARIMA model estimation done during the revisions history. If `refresh = no`, then the initial values of the regARIMA model parameters will be set to the estimates derived from the entire series. The default is `refresh = no`.

transformfcst Specifies whether the forecast error output of the history analysis is for forecasts of the original data (`transformfcst = no`) or for the forecasts of the transformed data specified by the **transform** spec (`transformfcst = yes`). See DETAILS. The default is `transformfcst = no`.

x11outlier Specifies whether the AO outlier identification will be performed during the history analysis for all irregular component regressions that result from the **x11regression** spec. If `x11outlier = yes`, AO outlier identification will be performed for each of the history runs. Those AO outlier regressors that were added to the irregular component regression model when automatic AO outlier identification was done for the full series are removed from the irregular component regression model prior to the history runs. If `x11outlier = no`, then the AO outlier regressors automatically identified are kept for each of the history runs. If the date of an outlier detected for the complete span of data does not occur in the data span of one of the history runs, the outlier will be dropped from the model for that run. The coefficients estimating the effects of these AO outliers are re-estimated whenever the other irregular component regression model parameters are re-estimated. However, no additional AO outliers are automatically identified and estimated. This option is ignored if the **x11regression** spec is not used, if the selection of the **aictest** argument in the **x11regression** spec results in the program not estimating an irregular component regression model, or if the **sigma** argument is used in the **x11regression** spec. The default is `x11outlier = yes`.

DETAILS

Section 6.3 gives technical details on revisions history analysis. For some supporting theory for out-of-sample squared forecast error diagnostic output, see Findley (2005).

When a revision history analysis of the seasonally adjusted series is specified for a composite seasonal adjustment, the revisions of both the direct and indirect seasonally adjusted series can be produced. The revision history analysis must be specified for each of the component series, even for those component series that are not seasonally adjusted, see the EXAMPLES section for the **composite** spec in Section 7.4.

The revision history of the indirect seasonally adjusted series (**sadj** in Table 7.17) is the only revision history available for indirect seasonal adjustments.

In each input specification file, a starting date for the history analysis must be specified using the **start** argument of this spec, and this starting date should be the same for each of the components and the composite series. If this is not the case, then only the history analysis of the direct seasonal adjustment will be performed.

If the automatic seasonal filter selection option is used, the program will redo the choice of seasonal filter each time the data span is changed in the revision history analysis. If the seasonal filter should change in the course of the analysis, a warning message will be printed out, and a table of the seasonal filter lengths chosen for each data span will be printed out.

The starting date for the forecast revisions depends on the values given for **fstep**. The starting date for a history of n-step-ahead forecast errors is n periods after the starting date of the history analysis. **Example:** if `fstep = (1 12)` and `start = 1992.jan`, the history for the 1-step and 12-step ahead forecasts will start in February of 1992 and January of 1993, respectively.

In some situations, the program automatically switches to using fixed model coefficients for the history analysis. This happens when the start of the revisions history analysis (which can be set by the user with the **start** argument) causes some truncated data span to have fewer than 60 observations for regARIMA model estimation, either because of the series length or a **span** or **modelspan** argument value (in the **series** or **composite** spec). In this case, the coefficients (ARIMA and regression) of the regARIMA model will be held fixed throughout the analysis at the values estimated from the entire series (or model span, if one is specified).

Fixing of the coefficients will also occur for every truncated data span that contains data later than the ending date specified in a **modelspan** argument. In particular, in the extreme case, when the ending date of the model span is earlier than the starting date of the history analysis, the coefficients of the regARIMA model will be fixed throughout the history analysis.

Regression models from the **x11regression** spec are treated similarly. For example, their coefficients are fixed if some truncated data span has fewer than 60 observations because of a date assigned to the **span** argument of the **x11regression** spec.

If an outlier specified by the user occurs in the period after the starting date of the revision history, that outlier will be dropped from the model at the start of the revision history analysis. It will be re-introduced into the regARIMA model when enough data have been added for the outlier variable to be defined. User-defined regressors are treated in the same way.

EXAMPLES

The following examples show complete spec files.

Example 1 A multiplicative monthly seasonal adjustment is to be performed with 3×9 seasonal moving averages for all months without using regARIMA model forecasts, backcasts, or regression outlier adjustments. A revision history of just the seasonally adjusted series will be performed (remember, this is the default history) for all data, after a startup period of twelve years (because 3×9 seasonal filters are used), with an additional analysis on the estimates made 2 periods after the concurrent observation.

```
Series {  Title = "Sales Of Livestock"  Start = 1997.1
          File = "cattle.ori"  }
X11 {  SeasonalMA = S3X9   }
History  {  sadjlags = 2  }
```

Example 2 Use a seasonal ARIMA model with regression variables for trading day and level shift preadjustment. The specified regression variables are a constant, trading day effects, and two level shifts, one in May 2002 and one in October 2006. The ARIMA part of the model is $(0\ 1\ 2)(1\ 1\ 0)_{12}$. Generate a history of the 1-step ahead forecast errors. Start the analysis in January of 2005; this means the first 1-step ahead forecast error in the analysis is for February of 2005.

```
series     {  title = "Exports of Leather goods"
              start = 1999.jul  file = "expleth.dat"  }
regression {  variables = (const td ls2002.may ls2006.oct)  }
arima      {  model = (0 1 2)(1 1 0)  }
estimate   {  }
history    {  estimates = fcst  fstep = 1  start=2005.jan  }
```

Example 3 Using the same regARIMA model and data as in Example 2, generate a history of the 1-step and 12-step ahead forecast errors as well as the ARMA coefficient estimates from the regARIMA model. Start the history in January of 2005. Save both histories to a file. In this file, zeros will be printed for the estimates where the 12-step ahead forecast errors are not defined (in this case, February to December of 2005) in order to maintain a uniform format for the file.

```
series     {  title = "Exports of Leather goods"
              start = 1999.jul
              file = "expleth.dat"  }
regression {  variables = (const td ls2002.may ls2006.oct)  }
arima      {  model = (0 1 2)(1 1 0)  }
estimate   {  }
history    {  estimates = (arma fcst)  start = 2005.jan
              save = (r6 amh)  }
```

Example 4 A multiplicative monthly seasonal adjustment is to be performed, with 3×3 seasonal moving averages, using regARIMA model forecasts to extend the series. The regARIMA model will be fit to the data up to the last December available to the series. A revision history of the seasonally adjusted series and the trend component will be calculated starting after the sixth year of the series, with the regARIMA model parameters re-estimated every December. Also, the history of the seasonal adjustment revisions of this series is integrated into the revision history calculation of the indirect seasonal adjustment of the composite series of which this series is a component. (The spec file for the composite series in the metafile must include an appropriate history spec, see Example 5.)

```
series {  title = "Housing Starts in the Midwest"
          start = 1967.1
          file = "hsmwtot.ori"
          modelspan = (,0.Dec)
          comptype=add
}
regression {  variables = td  }
arima { model = (0 1 2)(0 1 1)  }
x11 { seasonalMA = S3X3   }
history {  estimates = (sadj trend)  }
```

Example 5 A composite monthly seasonal adjustment is to be performed with 3×3 seasonal moving averages for all months using regARIMA model forecasts to extend the composite series. The regARIMA model will be fit to the data up to the last December available to the series. A revision history of both the direct and indirect seasonally adjusted series and the trend component from the direct seasonal adjustment will be performed, with the regARIMA model parameters re-estimated every December. The percent revisions for each of the estimates will be stored in separate files.

```
composite{ title = "Total Housing Starts in the US"
           modelspan = (,0.Dec)
  }
  regression {  variables = td  }
  arima { model = (0 1 1)(0 1 1)  }
  x11 {  seasonalMA = S3X3   }
  history  {  estimates = (sadj trend)
              save = (sar iar trr)  }
```

7.9 IDENTIFY

DESCRIPTION

Specification to produce tables and line printer plots of sample ACFs and PACFs for identifying the ARIMA part of a regARIMA model. Sample ACFs and PACFs are produced for all combinations of the nonseasonal and seasonal differences of the data specified by the **diff** and **sdiff** arguments. If the **regression** spec is present, the ACFs and PACFs are calculated for the specified differences of a series of regression residuals. If the **regression** spec is not present, the ACFs and PACFs are calculated for the specified differences of the original data.

USAGE

```
identify {  diff = (0, 1)
            sdiff = (0, 1)
            maxlag = 36
            print = (none +acf +acfplot +pacf +pacfplot)
            save = acf }
```

ARGUMENTS

diff Orders of nonseasonal differencing specified. The value 0 specifies no differencing, the value 1 specifies one nonseasonal difference $(1-B)$, the value 2 specifies two nonseasonal differences $(1-B)^2$, etc. The specified ACFs and PACFs will be produced for *all* orders of nonseasonal differencing specified, in combination with *all* orders of seasonal differencing specified in **sdiff**. The default is `diff = (0)`.

maxlag The number of lags specified for the ACFs and PACFs for both tables and plots. The default is 36 for monthly series, 12 for quarterly series.

print and **save** Table 7.21 gives the available output tables for this spec. All of these tables are included in the default printout, except for `regcoefficients`. For a complete listing of the `brief` and `default` print levels for this spec, see Appendix B.

sdiff Orders of seasonal differencing specified. The value 0 specifies no seasonal differencing, the value 1 specifies one seasonal difference $(1-B^s)$, etc. The specified ACFs and PACFs will be produced for *all* orders of seasonal differencing specified, in combination with *all* orders of nonseasonal differencing specified in **diff**. The default is `sdiff = (0)`.

DETAILS

If the **regression** spec is present, the program differences the series (after processing by the **transform** spec) and the regression variables using the maximum order of differencing specified by the **diff** and **sdiff** arguments. The differenced series is then regressed on the differenced regression variables. The resulting regression coefficients $(\tilde{\beta}_i)$ are then used to calculate *undifferenced* regression effects $(\sum_i \tilde{\beta}_i x_{it})$, which are then subtracted from the

name	*short*	*save?*	*description of table*
acf	iac	+	sample autocorrelation function(s), with standard errors and Ljung-Box Q-statistics for each lag
acfplot	acp	·	line printer plot of sample autocorrelation function(s) with ±2 standard error limits shown on the plot
pacf	ipc	+	sample partial autocorrelation function(s) with standard errors for each lag
pacfplot	pcp	·	line printer plot of sample partial autocorrelation function(s) with ±2 standard error limits shown on the plot
regcoefficients	rgc	·	regression coefficients removed from the transformed series before ACFs and PACFs were generated

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.21: **Available Output Tables for Identify**

undifferenced data (y_t) to produce a time series of undifferenced regression errors ($\tilde{z}_t = y_t - \sum_i \tilde{\beta}_i x_{it}$). This regression error time series and its differences, as specified by **diff** and **sdiff**, are then used to produce the ACFs and PACFs.

There is one exception to the above. If a constant term is specified in the **regression** spec (`variables = (const ...)`), it is included when the regression is done but not when the regression effects are subtracted from the series. See Section 4.4 for more discussion.

ACFs and PACFs are produced for all combinations of nonseasonal and seasonal differencing orders specified in **diff** and **sdiff**. For example, if `diff = (0, 1)` and `sdiff = 1` are specified, then ACFs and PACFs are computed for $(1-B^s)\tilde{z}_t$ and $(1-B)(1-B^s)\tilde{z}_t$, where $\tilde{z}_t$ is the series of regression errors, as discussed above, and s is the seasonal period specified in the **series** spec. If `diff = (0, 1, 2)` and `sdiff = (0, 1)` are specified, then ACFs and PACFs are computed for six series: $\tilde{z}_t$, $(1-B)\tilde{z}_t$, $(1-B)^2\tilde{z}_t$, $(1-B^s)\tilde{z}_t$, $(1-B)(1-B^s)\tilde{z}_t$, and $(1-B)^2(1-B^s)\tilde{z}_t$.

If both the **identify** and **estimate** specs are present, the **identify** spec is processed first. Note that the **identify** spec uses information from the **regression** spec, if present, but ignores the **arima** spec.

Users should make sure that differencing does not produce a singularity among the regression variables, including any user-defined regression variables, as singularities will cause a fatal error. One way this would arise is if **sdiff** was assigned a positive value (e.g., 1), while `variables = (seasonal)` was included in the **regression** spec.

If the number of lags requested for ACFs and PACFs equals or exceeds the length of the series (or the differenced series), the ACF and PACF will be computed only through the highest lag possible.

EXAMPLES

The following examples show complete spec files.

Example 1 Produce ACF tables useful for identifying the degree of differencing required for the monthly series $y_t = \log(Y_t)$, where Y_t is the original data input in the **series** spec. The ACFs are calculated for y_t, $(1-B)y_t$, $(1-B^{12})y_t$, and $(1-B)(1-B^{12})y_t$. The **regression** spec is absent so no regression effects are removed. ACFs are calculated through lag 36, the default for a monthly time series.

```
series {  title = "Monthly Sales"  start = 2006.jan
           data = (138 128 ... 297) }
transform { function = log }
identify { diff = (0, 1)
          sdiff = (0, 1)
          print = (none +acf) }
```

Example 2 Remove fixed seasonal effects before computing sample ACFs and sample PACFs. The **regression** spec includes a trend constant as well as the fixed seasonal variables. The **identify** spec removes the fixed seasonal effects by regressing $(1-B)y_t$ on the differenced regression variables $(1-B)x_{it}$, and computing undifferenced regression residuals $\tilde{z}_t = y_t - \sum_{i=2}^{12} \tilde{\beta}_i x_{it}$ (not subtracting out the trend constant term $\tilde{\beta}_1 x_{1t}$). It then computes ACFs and PACFs of $\tilde{z}_t$ and $(1-B)\tilde{z}_t$. The constant term allows for an overall nonzero mean in $(1-B)y_t$, so it is a linear trend constant, i.e., $x_{1t} = t$.

```
SERIES      { TITLE = "MONTHLY SALES"  START = 2006.JAN
              DATA = (138 128 ... 297) }
REGRESSION  { VARIABLES = (CONST SEASONAL) }
IDENTIFY    { DIFF = (0,1) }
```

Example 3 Produce ACF and PACF plots to identify the AR and MA parts of a regARIMA model. Do not print ACF and PACF tables. Suppose Y_t is the same series as in Example 1, that one nonseasonal and one seasonal difference are chosen, and that the model will include trading-day and Easter holiday effects. Because the **regression** spec is present, the **identify** spec first regresses $(1-B)(1-B^{12})y_t$ on $(1-B)(1-B^{12})x_{it}$, where the x_{it} are the regression variables for the trading-day and Easter holiday effects, and y_t consists of the logarithms of the original data Y_t adjusted for length-of-month effects. (See the description of `td` in the **regression** spec.) If $\tilde{\beta}_i$ denote the estimated regression coefficients, then this **identify** spec produces ACF and PACF plots for the regression residual series $(1-B)(1-B^{12})\left(y_t - \sum_i \tilde{\beta}_i x_{it}\right)$. The ACFs and PACFs are computed through lag 30.

```
Series { Title = "Monthly Sales"  Start = 2006.Jan
         Data = (138 128 ... 297) }
Transform { Function = Log }
Regression { Variables = (Td Easter[14])}
Identify { Diff = (1)  Sdiff = (1)  Maxlag = 30
           Print = (None +ACFplot +PACFplot)  }
```

Example 4 Produce ACFs and PACFs (through lag 16) for model identification, and also estimate a tentative model for a quarterly series. There is a known level shift in the first quarter of 2001. Its effect is estimated by regressing $(1-B)(1-B^4)y_t$ on the differenced level shift variable. This regression effect is then removed to produce the (undifferenced) regression residual series, $\tilde{z}_t = y_t - \tilde{\beta}\text{LS2001.1}_t$, and ACFs and PACFs are calculated for $\tilde{z}_t$, $(1-B)\tilde{z}_t$, $(1-B^4)\tilde{z}_t$, and $(1-B)(1-B^4)\tilde{z}_t$. The **identify** spec ignores the information in the **arima** spec.

The spec file below also specifies estimation and standard diagnostic checks of the regARIMA model, $(1-B)(1-B^4)\left(y_t - \beta\text{LS2001.1}_t\right) = (1-\theta B)(1-\Theta B^4)a_t$. Such an estimation of a tentative model on the same run that produces ACFs and PACFs for model identification is sometimes useful, if one has a prior idea what the appropriate ARIMA model might be. This could occur if the series had been modelled previously, but new data has since extended the series. If the diagnostic checks suggest that the tentative model is inadequate, the user will have information from both the diagnostic checks and the **identify** spec output to use in selecting a new model.

```
series { title = "Quarterly Sales"  start = 1993.1  period = 4
         data = (56.7 57.7 ... 68.0) }
regression { variables = (ls2001.1) }
arima { model = (0 1 1)(0 1 1) }
identify { diff = (0, 1)  sdiff = (0, 1)  maxlag = 16 }
estimate {  }
check {  }
```

7.10 METADATA

DESCRIPTION

Specification that allows users to insert metadata into the diagnostic summary file. Users can specify keys and corresponding values for those keys to insert additional information into the diagnostic summary file stored by `X-13ARIMA-SEATS` .

USAGE

```
metadata {   keys = (
                 "survey"
                 "analyst"
             )
             values = (
                 "United States retails sales"
                 "Dr.  Sigerson"
             )
           }
```

ARGUMENTS

keys A list of character strings used as keys for the metadata values specified in the **values** list. Up to 20 values can be specified - no single key can be more than 132 characters long, and all the keys taken together cannot be exceed 2000 characters.

An example with two keys is:

```
keys = ( "note1"
         "note2" )
```

If a list with more than one entry is used, each key must be on a separate line of the spec file. The keys should not contain spaces or colons (periods, commas and semicolons can be used), and should be unique values – each key must be different. Missing values and blank lines are not allowed.

values A list of character strings used as values associated with the keys provided in the **keys** argument. Up to 20 values can be specified – no single entry can be more than 132 characters long, and all the entries taken together cannot exceed 2000 characters.

An example with two arguments is:

```
values = ( "Special sale caused outlier in October 2005"
           "Analysis as of November 2006" )
```

If a list with more than one entry is used, each value must be on a separate line of the spec file. Missing values and blank lines are not allowed.

DETAILS

The **metadata** spec allows users to insert their own metadata into the summary diagnostics file. Users can specify unique keys and corresponding values for those keys, and these values are then entered as records into the summary diagnostics file. These records are formatted as

```
metadata.key: value
```

where `key` is a unique key specified by the user, and `value` is the corresponding value for that key. The text `"metadata."` signifies that this is user-defined metadata.

For example, when the user includes the following **metadata** spec into an input specification file:

```
metadata {
     keys = (
         "analyst"
         "date.reviewed"
         "units.of.measure"
         )
     values = (
         "Allen Smithee"
         "June 15, 2006"
         "Millions of Dollars"
         )
}
```

the following records will be written to the summary diagnostics file:

```
metadata.analyst: Allen Smithee
metadata.date.reviewed: June 15, 2006
metadata.units.of.measure: Millions of Dollars
```

In previous versions of `X-13ARIMA-SEATS` , the summary diagnostics file was generated only when the **-s** or **-g** runtime flags are used; now the summary diagnostics file will also be generated whenever the metadata spec is used.

If there are fewer keys than there are values, a warning message is produced, and the program will generate unique keys based on the position of the value in the array.

For example, the following **metadata** spec:

```
metadata {
     keys = (
         "analyst"
         "date.reviewed"
         )
     values = (
         "Allen Smithee"
         "June 15, 2006"
         "Millions of Dollars"
         )
}
```

produces the following records in the summary diagnostics file:

```
metadata.analyst: Allen Smithee
metadata.date.reviewed: June 15, 2006
metadata.key3: Millions of Dollars
```

Not specifying a key argument at all will force the program to generate unique keys for all the values specified.

```
metadata {
     values = (
         "Allen Smithee"
         "June 15, 2006"
         "Millions of Dollars"
         )
}
```

produces the following records in the summary diagnostics file:

```
metadata.key1: Allen Smithee
metadata.key2: June 15, 2006
metadata.key3: Millions of Dollars
```

If more keys are specified than values, execution will cease, and an error message will be produced.

Note that the **metadata** spec can appear in any order relative to the the other specs - it can be the first spec in the spec file, etc.

EXAMPLES

The following examples show complete spec files.

Example 1 Print all available diagnostic checks of the residuals from the specified model. The **check** spec forces model estimation to be performed (with default options) even though the **estimate** spec is not present. The **metadata** spec documents the analyst that developed the spec file.

```
series { title = "Monthly Retail Sales"
         start = 2004.jan  file = "sales1.dat" }
regression {  variables = td aictest = ( td easter ) }
arima { model = (0 1 1)(0 1 1) }
check { print = (all) }
outlier {  types = all  }
metadata {
   key = "analyst"
   value = "John J. J. Smith"
}
```

The record stored in the summary diagnostic file is

```
metadata.analyst: John J. J. Smith
```

Example 2 For the same series and model as in Example 1, produce all diagnostic checking statistics except the printed table and plot of the residual PACF. The residual ACF is computed through lag 24.

```
series { title = "Monthly Retail Sales"
         start = 2004.jan  file = "sales1.dat" }
regression {
  variables = ( td ao2007.jun ls2011.jun easter[8] )
  )
}
arima { model = (0 1 1)(0 1 1) }
check { print = (all -pacf -pacfplot) }
metadata {
   key = (
      "analyst"
      "spec.updated"
   )
   value = (
      "John J. J. Smith"
      "October 31, 2017"
   )
}
```

The record stored in the summary diagnostic file is

```
metadata.analyst: John J. J. Smith
metadata.spec.updated: October 31, 2017
```

Example 3 For the same series and model as in Example 2, add metadata text to describe the outliers found by the automatic outlier procedure.

```
series { title = "Monthly Retail Sales"
         start = 2004.jan  file = "sales1.dat" }
regression {
  variables = (
    td ao2007.jun ls2011.jun easter[15]
  )
}
arima { model = (0 1 1)(0 1 1) }
check { print = (all -pacf -pacfplot) }
x11 {   save = d11   }
metadata {
   key = (
      "analyst"
      "spec.final"
   )
   value = (
      "John J. J. Smith"
      "November 10, 2017"
      "AO caused by strike, LS caused by survey change"
   )
}
```

The record stored in the summary diagnostic file is

```
metadata.analyst: John J. J. Smith
metadata.spec.updated: November 10, 2017
metadata.key3: AO caused by strike, LS caused by survey change
```

7.11 OUTLIER

DESCRIPTION

Specification to perform automatic detection of additive (point) outliers, temporary change outliers, level shifts, or any combination of the three using the specified model. After outliers (referring to any of the outlier types mentioned above) have been identified, the appropriate regression variables are incorporated into the model as "Automatically Identified Outliers," and the model is re-estimated. This procedure is repeated until no additional outliers are found. If two or more level shifts are detected (or are present in the model due to the specification of level shift(s) in the **regression** spec), t-statistics can be computed to test null hypotheses that each run of two or more successive level shifts cancels to form a temporary level shift.

USAGE

```
outlier {   types = all
            critical = 3.75
            method = addall
            span = (1983.may, 1992.sep)
            lsrun = 0
            print = (none +header)
            save = tests
            savelog = id
        }
```

ARGUMENTS

critical Sets the value to which the absolute values of the outlier t-statistics are compared to detect outliers. The default critical value is determined by the number of observations in the interval searched for outliers (see the **span** argument below). It is obtained by a modification of the asymptotic formula of Ljung (1993) that interpolates critical values for numbers of observations between 3 and 99. Table 7.22 gives default critical values for various outlier span lengths.

If only one value is given for this argument (`critical = 3.5`), then this critical value is used for all types of outliers. If a list of up to three values is given (`critical = (3.5, 4.0, 4.0)`), then the critical value for additive outliers is set to the first list entry (3.5 in this case), the critical value for level shift outliers is set to the second list entry (4.0), and the critical value for temporary change outliers is set to the third list entry (4.0). A missing value, as in `critical = (3.25, , 3.25)`, is set to the default critical value. Raising the critical value decreases the sensitivity of the outlier detection routine, possibly decreasing the number of observations treated as outliers.

lsrun Compute t-statistics to test null hypotheses that each run of 2, ..., **lsrun** successive level shifts cancels so that their net effect after the last level shift in the run is zero. The t-statistics are computed as the sum of the estimated parameters for the level shifts in

Number of Observations Tested	Outlier Critical Value	Number of Observations Tested	Outlier Critical Value
1	1.96	48	3.63
2	2.24	72	3.73
3	2.44	96	3.80
4	2.62	120	3.85
5	2.74	144	3.89
6	2.84	168	3.92
7	2.92	192	3.95
8	2.99	216	3.97
9	3.04	240	3.99
10	3.09	264	4.01
11	3.13	288	4.03
12	3.16	312	4.04
24	3.42	336	4.05
36	3.55	360	4.07

Table 7.22: **Default Critical Values for Outlier Identification**

each run divided by the appropriate standard error. (See Otto and Bell 1993). Both automatically identified level shifts and level shifts specified in the **regression** spec are used in the tests. The **lsrun** argument can take values from 0 to 7; 0 and 1 request no computation of the *t*-statistics for cancellation of level shifts. If the value specified for **lsrun** exceeds the total number of level shifts in the model following outlier detection, then **lsrun** is reset to this total. The default value for **lsrun** is 0, i.e., no *t*-statistics for cancellation of level shifts are computed.

method Determines how the program successively adds detected outliers to the model. The choices are `method = addone` or `method = addall`. See DETAILS for a description of these two methods. The default is `method = addone`.

print and **save** Table 7.23 gives the available output tables for this spec. The **header** and **temporaryls** tables are printed out by default. For a complete listing of the `default` and `brief` print levels for this table, see Appendix B.

Note: The entry for an outlier *t*-statistic in the **finaltests** table is set to zero when testing for that outlier (regressor) causes the regression matrix to be singular, or when that outlier has been specified in the **variables** argument of the **regression** spec. In addition, when the **finaltests** table is saved, the *t*-statistics for all automatically identified outliers are also set to zero – that is, **finaltests** shows the *t*-statistics for time points that are not outliers. This table cannot be saved when automatic model selection is invoked using either the **automdl** or **pickmdl** specs.

savelog Setting `savelog = identified` or `savelog = id` causes a list of automatically identified outliers to be output to the log file (see section 2.6 for more information on the log file).

span Specifies start and end dates of a span of the time series to be searched for outliers. The

name	*short*	*save?*	*description of table*
header	hdr	·	options specified for outlier detection including critical value, outlier span, and types of outliers searched for
iterations	oit	+	detailed results for each iteration of outlier detection including outliers detected, outliers deleted, model parameter estimates, and robust and non-robust estimates of the residual standard deviation
tests	ots	·	t-statistics for every time point and outlier type on each outlier detection iteration
temporaryls	tls	·	summary of t-statistics for testing cancellation of level shifts
finaltests	fts	+	t-statistics for every time point and outlier type generated during the final outlier detection iteration (not saved when **automdl**/**pickmdl** is used)

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.23: **Available Output Tables for Outlier**

start and end dates of the span must both lie within the series and within the model span if one is specified by the **modelspan** argument of the **series** spec, and the start date must precede the end date. A missing value, e.g., `span = (1996.jan,)`, defaults to the start date or end date of the series, as appropriate. (If there is a **span** argument in the **series** spec, then, in the above remarks, replace the start and end dates of the series by the start and end dates of the span given in the **series** spec.)

(Note that a **span** argument in the **series** spec will supercede the **span** argument in the **outlier** spec.)

types Specifies the types of outliers to detect. The choices are: `types = ao` (detect additive outliers only), `types = ls` (detect level shifts only), `types = tc` (detect temporary change outliers only), `types = all` (detect all three of the above types simultaneously), and `types = none` (turn off outlier detection, but not t-statistics for temporary level shifts). The default is `types = (ao ls)`.

RARELY USED ARGUMENTS

almost Differential used to determine the critical value used for a set of "almost" outliers – outliers with t-statistics near the outlier critical value that are not incorporated into the regARIMA model. After outlier identification, any outlier with a t-statistic larger than

critical – **almost** is considered an "almost outlier" and is included in a separate table. The default is `almost = 0.5`; values for this argument must always be greater than zero.

tcrate Defines the rate of decay for the temporary change outlier regressor. This value must be a number greater than zero and less than one. The default value is `tcrate=0.7 ** (12/period)`, where period is the number of observations in one year (12 for monthly time series, 4 for quarterly time series). This formula for the default value of **tcrate** ensures the same rate of decay over an entire year for series of different periodicity. If the frequency of the time series is less than 4 (i.e., period < 4), then there is no default value, and the user will have to enter a value of **tcrate** if a temporary change outlier was specified in the **variables** argument of the **regression** spec, or if temporary change outliers are specified in the **types** argument of this spec. If this argument is specified in the **regression** spec, it is not necessary to include it in this spec.

DETAILS

A level shift (LS) at the first data point cannot be estimated since the level of the series prior to the given data is unknown. Therefore, no LS test statistic is calculated for the first data point. Also, an LS at the last data point cannot be distinguished from an AO there, and an LS at the second data point cannot be distinguished from an AO at the first data point. Thus, LS statistics are calculated for the second and last data points only if AOs are not also being detected. LS statistics that are not calculated are set to and printed out as 0.

Similarly, a temporary change (TC) outlier at the last data point cannot be distinguished from an AO there, so no TC statistic is calculated for the last data point if an AO is also being detected. TC statistics that are not calculated are set to and printed out as 0.

Users should be aware that certain combinations of AOs and LSs produce arithmetically equal effects. For example, these are equivalent: (*i*) an AO at time t_0 followed by an LS at $t_0 + 1$, (*ii*) LSs at both t_0 and $t_0 + 1$, and (*iii*) both an AO and an LS at t_0. Note that an LS at t_0 followed by an AO at $t_0 + 1$ is not equivalent to these other combinations, however. Because AOs are assigned to the irregular component and LSs to the trend-cycle, some users might prefer one equivalent combination of outliers over another.

In regard to the tests for cancellation of level shifts, it is worth noting that two LSs that cancel are equivalent to a temporary level shift that spans the interval from the time point of the first LS to the time point just before the second LS. See Table 4.1 in Section 4.3 for the definition of the temporary level shift variable TL.

The **addone** method works in the following way. The program calculates t-statistics for each type of outlier specified (AO, TC, and/or LS) at all time points for which outlier detection is being performed. If the maximum absolute outlier t-statistic exceeds the critical value, then an outlier has been detected, and the appropriate regression variable is added to the model. The program then estimates the new model (the old model with the detected outlier added) and looks for an additional outlier. This process is repeated until no additional outliers are found. At this point, a backward deletion process is used to delete "insignificant" outliers (those whose absolute t-statistics no longer exceed the critical value) from the model. This is done one at a time beginning with the least significant outlier, until all outliers remaining in the model are significant. During backward deletion the usual (non-robust) residual variance estimate is used, which can yield somewhat different outlier t-statistics than those obtained during outlier detection.

The **addall** method follows the same general steps as the **addone** method, except that on each outlier detection pass the **addall** method adds to the model *all* outliers with absolute t-statistics exceeding the critical value. Typically several of the outliers added this way will be found to be insignificant when the new model is estimated. The **addall** method thus depends heavily on the backward deletion process (much more than the **addone** method does) to remove unnecessary outliers added to the model in the detection phase.

The differences between the **addone** and **addall** schemes can produce different final sets of detected outliers. Two practical differences between the methods are worth noting. First, the **addone** method generally takes more computation time than **addall**. Second, the **addall** method may add so many outliers on a detection pass that it exceeds the maximum number of regression variables allowed in a model. In this case the program prints an error message to this effect and stops. Suggested remedies are to raise the cutoff value so fewer outliers are detected, or to switch to the **addone** method, for which this phenomenon is much less likely.

For either method, the outlier t-statistics for all possible time points on each detection pass can be printed by specifying `print = iterations`. Note that this option generates considerable output.

Choosing the critical value requires both judgement and experience. Based on a simulation study involving series of length up to 200 generated from low order nonseasonal ARIMA models, Chang, Tiao, and Chen (1988) recommended critical values of 3 for high sensitivity in detection of AO outliers, 3.5 for medium sensitivity, and 4 for low sensitivity.

Outlier detection begins with the model specified by the **regression** and **arima** specs and with estimated parameters. If the **estimate** spec is absent, the **outlier** spec forces estimation of the model (with default estimation options) prior to outlier detection.

If outliers are suspected at specific known time points, then they may be included in the model by adding the appropriate AO, TC, or LS regression variables directly to the model in the **regression** spec.

Outlier detection results can vary depending on the regARIMA model specified: observations are classified as outliers because the model fits them less well than most of the other observations. Therefore an inadequate regARIMA model can yield inappropriate outlier adjustments.

EXAMPLES

The following examples show complete spec files.

Example 1 Simultaneously search for both AO and LS outliers over the entire time series, using the **addone** method and a critical value that depends on the number of observations in the interval searched for outliers (default options). If the number of level shifts present in the model following outlier detection is two or more, compute t-statistics to test whether each run of 2, ..., 5 successive level shifts cancels to form a temporary level shift. Though the **estimate** spec is absent, the presence of the **outlier** spec forces model estimation with default estimation options.

```
series {  title = "Monthly sales"  start = 1996.jan
           data = (138 128 ... 297) }
arima {   model = (0 1 1)(0 1 1)12 }
outlier { lsrun = 5  types=(ao ls) }
```

Example 2 Search only for AO outliers using the **addall** method and a critical value of $t = 4.0$. Because the **span** argument is present in the **series** spec, only the time frame given there (January 2000 through December 2012) is used in model estimation and in outlier detection. The two level shifts specified in the **regression** spec are not tested for cancellation into a temporary level shift since **lsrun** takes its default value of 0.

```
Series { Title = "Monthly sales"  Start = 1996.jan
         Data = (138 128 ... 297)
         Span = (2000.Jan, 2012.Dec) }
Regression { Variables = (LS2001.Jun LS2010.Nov) }
Arima { Model = (0 1 1)(0 1 1)12 }
Estimate {  }
Outlier { Types = AO   Method = Addall  Critical = 4.0  }
```

Example 3 Estimate the model using the same span as in Example 2, but search only for LS outliers in 2011 and 2012. The default **addone** method is used, but with a critical value of $t = 3.0$. Each pair of successive LSs is tested for possible cancellation into a temporary LS.

```
series { title = "Monthly sales"  start = 1996.jan
           data = (138 128 ... 297)
           span = (2000.jan, 2012.dec) }
arima {  model = (0 1 1)(0 1 1)12 }
estimate { }
outlier { types = ls
          critical = 3.0
          lsrun = 2
          span = (2011.jan, 2012.dec) }
```

Example 4 Estimate the model using the same span as in Examples 2 and 3, but search for AO, TC, and LS outliers. The default **addone** method is used, but with a critical value of $t_{AO} = 3.0$ for AO outliers, $t_{LS} = 4.5$ for LS outliers, and $t_{TC} = 4.0$ for TC outliers.

```
series { title = "Monthly sales"  start = 1996.jan
           data = (138 128 ... 297)
           span = (2000.jan, 2012.dec) }
arima { model  = (0 1 1)(0 1 1)12 }
estimate { }
outlier { critical = (3.0, 4.5, 4.0)
          types = all
        }
```

7.12 PICKMDL

DESCRIPTION

Specifies that the ARIMA part of the regARIMA model will be sought using an automatic model selection procedure similar to the one used by `X-11-ARIMA/88` (see Dagum 1988). The user can specify which types of models are to be fitted to the time series in the procedure and can change the thresholds for the selection criteria.

USAGE

```
pickmdl {   mode = both
            method = best
            file = "my.mdl"
            fcstlim = 25.0
            bcstlim = 25.0
            qlim = 15.0
            overdiff = 0.99
            identity = all
            outofsample = yes
            print = (none autochoice)
            savelog = automodel
        }
```

ARGUMENTS

bcstlim Sets the acceptance threshold for the within-sample backcast error test when backcasts are specified by setting `mode = both`. The absolute average percentage error of the backcasted values is then tested against the threshold. For example, `bcstlim = 25` sets this threshold to 25 percent. The value entered for this argument must not be less than zero, or greater than 100. The default for **bcstlim** is 20 percent.

fcstlim Sets the acceptance threshold for the within-sample forecast error test. The absolute average percentage error of the extrapolated values within the last three years of data must be less than this value if a model is to be accepted by the **pickmdl** automatic modeling selection procedure. For example, `fcstlim = 20` sets this threshold to 20 percent. The value entered for this argument must not be less than zero, or greater than 100. The default for **fcstlim** is 15 percent.

file Valid path and filename of the file containing the models used in the **pickmdl** automatic model selection procedure. The models are specified using the same notation as in the **model** argument of the **arima** spec; see DETAILS below. This argument must be specified; there is no default.

identify Determines how automatic identification of outliers (via the **outlier** spec) and/or automatic trading day regressor identification (via the **aictest** argument of the **regression**

spec) are done within the **pickmdl** automatic model selection procedure. If `identify = all`, automatic trading day regressor and/or automatic outlier identification (done in that order if both are specified) are done for each model specified in the automatic model file. If `identify = first`, automatic trading day regressor and/or automatic outlier identification are done for the first model specified in the automatic model file. The decisions made for the first model specified are then used for the remaining models. The identification procedures are redone for the selected model, if the model selected is not the first. The default is `identify = first`.

method Specifies whether the **pickmdl** automatic model selection procedure will select the first model that satisfies the model selection criteria (`method = first`) or the estimated model with the lowest within-sample forecast error among all models that satisfy the model selection criteria (`method = best`). The default is `method = first`.

mode Specifies that the program will attempt to find a satisfactory model within the set of candidate model types specified by the user, using the criteria developed by Statistics Canada for the `X-11-ARIMA` program and documented in Dagum (1988); see DETAILS. The fitted model chosen will be used to produce a year of forecasts if `mode = fcst` or a year of forecasts and backcasts if `mode = both`. The default is `mode = fcst`. The **forecast** spec can be used to override the number of forecasts and backcasts used to extend the series. The model will be chosen from the types read in from a file named in the **file** argument (specified above). Do not use both **arima** and **pickmdl** in the same specification file. The same applies for **automdl** and **pickmdl**.

outofsample Determines which kind of forecast error is used for **pickmdl** automatic model evaluation and selection. If `outofsample = yes`, out-of-sample forecasts errors are used; these are obtained by removing the data in the forecast period from the data used to estimate the model and to produce one year of forecasts (for each of the last three years of data). If `outofsample = no`, within-sample forecasts errors are used. That is, the model parameter estimates for the full series are used to generate forecasts for each of the last three years of data. For conformity with `X-11-ARIMA`, outlier adjustments are made to the forecasted data that have been identified as outliers. The default is `outofsample = no`.

overdiff Sets the threshold for the sum of the MA parameter estimates in the overdifferencing test. The program computes the sum of the seasonal (for models with at least one seasonal difference) or nonseasonal (for models with at least one nonseasonal difference) MA parameter estimates. If the sum of the nonseasonal MA parameter estimates is greater than the limit set here, the **pickmdl** automatic model selection procedure will reject the model because of overdifferencing. If the sum of the seasonal MA parameter estimates is greater than the limit set here, the **pickmdl** automatic model selection procedure will print out a warning message suggesting the use of fixed seasonal effects in the **regression** spec, but will not reject the model. The default for this argument is 0.9; values entered for this argument should not be any lower than 0.9 and must not be greater than 1.

print The save option is not available for this spec. The tables available for output are listed in Table 7.24; all tables are included in the default printout. For a complete listing of the `brief` and `default` print levels for this spec, see Appendix B.

name	*short*	*description of table*
pickmdlchoice	pch	model choice of the **pickmdl** automatic model selection procedure
header	hdr	header for the **pickmdl** output
usermodels	umd	output for each model used in the **pickmdl** automatic model selection procedure

Name gives the name of each table for use with the **print** argument.
Short gives a short name for these tables.

Table 7.24: **Available Output Tables for Pickmdl**

qlim Sets the acceptance threshold for the p-value of the Ljung-Box Q-statistic for model adequacy. The p-value associated with the fitted model's Q must be greater than this value for a model to be accepted by the **pickmdl** automatic model selection procedure. For example, `qlim = 10` sets this threshold to 10 percent. The value entered for this argument must not be less than zero or greater than 100. The default for **qlim** is 5 percent.

savelog Setting `savelog = automodel` or `savelog = amd` causes the result of the model selection procedure to be output to the log file (see section 2.6 for more information on the log file).

DETAILS

The **pickmdl** spec **cannot** be used in the same spec file as the **automdl** or **arima** specs or when the **file** argument is specified in the **estimate** spec.

The default settings for the **pickmdl** automatic model selection procedure classify a model as acceptable if (1) the absolute average percentage error of the extrapolated values within the last three years of data is less than 15 percent, (2) the p-value associated with the fitted model's Ljung-Box Q-statistic test of the lack of correlation in the model's residuals must be greater than 5 percent, and (3) there are no signs of overdifferencing. There is an indication of overdifferencing if the sum of the nonseasonal MA parameter estimates (for models with at least one nonseasonal difference) is greater than 0.9. No model is selected when none of the models of the types in the model file is acceptable, unless one of the models is marked as a "default" model (more on this shortly). Any of these criteria can be changed using the **fcstlim**, **qlim**, and **overdiff** arguments.

Note that if there is a **regression** spec in the spec file, the regression terms specified there will be used with all the ARIMA models evaluated by the automatic model selection procedure. The original series is transformed as specified in the **transform** spec.

The `X-11-ARIMA` program developed by Statistics Canada uses the following model types in its automatic modeling procedure:

$$(0,1,1)(0,1,1)_s$$
$$(0,1,2)(0,1,1)_s$$
$$(2,1,0)(0,1,1)_s$$
$$(0,2,2)(0,1,1)_s$$
$$(2,1,2)(0,1,1)_s$$

where s denotes the seasonal period (see Dagum 1988). These model types cannot be used if a fixed seasonal effect is specified in the **regression** spec.

Each model type in the file designated by the **file** argument is listed on a separate line, with `"X"` at the end of each line except the last.

As mentioned above, no model is normally selected when none of the models of the types in the model file is acceptable. However, users can select one of the models to be a "default" model by placing an asterisk (`"*"`) instead an `"X"` at the end of the line. This will allow the program to use the default regARIMA model to generate preadjustment factors based on the regressors specified by the user in the **regression** spec if a model is not otherwise selected by the automatic modeling procedure. No forecasts (or backcasts) are generated if none of the models are selected by the procedure.

An example using the `X-11-ARIMA` default models is given below:

```
(0 1 1)(0 1 1) *
(0 1 2)(0 1 1) X
(2 1 0)(0 1 1) X
(0 2 2)(0 1 1) X
(2 1 2)(0 1 1)
```

EXAMPLES

The following examples show complete spec files.

Example 1 Use the automatic ARIMA modeling procedure to select a model and use it to extend the series with one year of forecasts. Trading day and stable seasonal regression effects are to be included in the models. A default seasonal adjustment is to be performed.

```
series      { title = "Monthly sales"   start = 2006.jan
              data = (138 128  ...  297) }
regression  { variables = (td seasonal) }
pickmdl     { mode = fcst    file = "nosdiff.mdl"  }
estimate    {  }
x11         {  }
```

The contents of `nosdiff.mdl` are given below:

```
(1 1 0) X
(2 1 0) X
(0 1 1) *
(0 1 2) X
(2 1 2)
```

Example 2 Similar to Example 1, except that the forecast acceptance threshold is changed to 20 percent, the chi-square acceptance threshold is set to 10 percent, and the overdifferencing acceptance threshold is changed to 0.99. Also, the first acceptable model will be selected, and automatic outlier identification will be done for all the models listed in `nosdiff.mdl`.

```
series      { title = "Monthly sales"   start = 2006.jan
              data = (138 128  ...  297) }
regression  { variables = td }
pickmdl     { mode = fcst    file = "nosdiff.mdl"
              method = first   fcstlim = 20   qlim = 10
              overdiff = 0.99  identify = all }
outlier     {  }
estimate    {  }
x11         {  }
```

Example 3 The same as Example 1, except that out-of-sample forecast errors are used in the model identification and selection process.

```
series      { title = "Monthly sales"   start = 2006.jan
              data = (138 128  ...  297) }
regression  { variables = td }
pickmdl     { mode = fcst    file = "nosdiff.mdl"
              outofsample=yes  }
estimate    {   }
x11         {   }
```

7.13 REGRESSION

DESCRIPTION

Specification for including regression variables in a regARIMA model or for specifying regression variables whose effects are to be removed by the **identify** spec to aid ARIMA model identification. Predefined regression variables are selected with the **variables** argument. The available predefined variables provide regressors modeling a constant effect, fixed seasonality, trading-day and holiday variation, additive outliers, level shifts, and temporary changes or ramps. Change of regime regression variables can be specified for seasonal and trading-day regressors. User-defined regression variables can be added to the model with the **user** argument. Data for any user-defined variables must be supplied, either in the **data** argument or in a file specified by the **file** argument (but not both). The **regression** spec can contain both predefined and user-defined regression variables.

USAGE

```
regression {  variables = (
                const seasonal or sincos[1, 2, 3]
                td or tdnolpyear or tdstock[31] or
                  td1coef or td1nolpyear or tdstock1coef[31]
                lom or loq lpyear
                easter[8] or sceaster[8] or easterstock[8]
                labor[8] thank[1]
                ao1997.apr ls1992.sep tc1999.sep
                aos1988.apr-1988.jul lss1982.sep-1982.nov
                rp1997.nov-1998.may qd2005.feb-2005.may qi2010.jul-2010.nov
                so1994.mar tl1969.mar-1969.may
              )
              print = (none) save = (rmx)
              savelog = aictest
              testalleaster = yes
              user = (cnybefore cnyafter IdulFitr strike)
              usertype = (holiday holiday holiday2 ao)
              start = 1995.jan
              data = (25 0.1 ···) or file = "weather.dat"
                                     format = "(2f5.1)"
              aictest = ( easter user
                          td or tdnolpyear or tdstock or
                            td1coef or td1nolpyear or tdstock1coef
                          lom or loq or lpyear )
              aicdiff = (2.0, ,3.0,) or pvaictest = 0.01
              tlimit = 2.0
              chi2test = yes
              chi2testcv = 0.005
            }
```

ARGUMENTS

aicdiff Defines the amount by which the AIC value (corrected for the length of the series, or AICC) of the model with the regressor(s) specified in the **aictest** argument must fall below the AICC of the model without these regressor(s) in order for the model with the regressors to be chosen. The default value is `aicdiff = 0.0`.

If only one value is given for this argument (`aicdiff = 3.5`), then this critical value is used for all types of regressors. If a list of up to four values is given (`aicdiff = (3.5, 4.0, 4.0, 5.5)`), then the AIC difference for trading day regressors is set to the first list entry (3.5 in this case), the AIC difference for length of month regressors is set to the second list entry (4.0), the AIC difference for Easter regressors is set to the third list entry (4.0), and the AIC difference for user-defined regressors is set to the fourth list entry (5.5). A missing value, as in `aicdiff = (3.25, ,3.25,)`, is set to the default critical value.

This argument cannot be used in the same spec file as the **pvaictest** argument.

For more information on how this option is used in conjunction with the **aictest** argument, see DETAILS.

aictest Specifies that an AIC-based selection will be used to determine if a given set of regression variables will be included with the regARIMA model specified. The only entries allowed for this variable are **td**, **tdnolpyear**, **tdstock**, **td1coef**, **td1nolpyear**, **tdstock1coef**, **lom**, **loq**, **lpyear**, **easter**, **easterstock**, and **user**. If a trading day model selection is specified, for example, then AIC values (with a correction for the length of the series, henceforth referred to as AICC) are derived for models with and without the specified trading day variable. By default, the model with smaller AICC is used to generate forecasts, identify outliers, etc. If more than one type of regressor is specified, the AIC tests are performed sequentially in this order: (1) trading day regressors, (2) length of month / length of quarter / leap year regressors, (3) Easter regressors, (4) user-defined regressors. If there are several variables of the same type (for example, several trading day regressors), then the **aictest** procedure is applied to them as a group. That is, either all variables of this type will be included in the final model or none. See DETAILS for more information on the testing procedure. If this option is not specified, no automatic AIC-based selection is performed.

chi2test Specifies that chi-squared statistics will be used to determine if groups of user-defined holiday regressors will be kept in the regARIMA model. When `chi2test = yes`, chi-squared statistics will be generated for all user-defined holiday regression groups, and those that are not significant (at the level of the argument **chi2testcv**) are removed from the regARIMA model. The default is `chi2test = no`, where no testing is done.

chi2testcv Sets the probability for the critical value used for the selection procedure in **chi2test**. The default is 0.01.

data Assigns data values to the user-defined regression variables. The time frame of the data values must cover the time frame of the series (or of the span specified by the **span** argument of the **series** spec, if present). It must also cover the time frame of forecasts

and backcasts requested in the **forecast** spec.[19]

The data values are read in free format. The numerical values given in this argument are assigned in the order in which the user-defined variables are named in the **user** argument. This assignment proceeds through all the user-defined variables for the first time point, then through all the variables for the second time point, etc. If the **data** argument is used, the **file** argument cannot be used.

file Name of the file containing data values for *all* user-defined regression variables. The filename must be enclosed in quotes. If the file is not in the current directory, the path must also be given. As with the **data** argument, the time frame of the data values must cover both the series and any forecasts and backcasts requested.[20] If the **file** argument is used, the **data** argument cannot be used.

format Denotes the format used when reading the data for the regression variables from the file named in the **file** argument. Six types of input are accepted:

a. free format, in which all numbers on a line will be read before continuing to the next line, and the numbers must be separated by one or more spaces (not by commas or tabs) (example: `format = "free"`);

b. a valid Fortran format, which must be enclosed in quotes and must include the initial and terminal parentheses (example: `format = "(6f12.0)"`);

c. "datevalue" format, in which the year, month or quarter, and the associated value for each of the user-defined regression variables for a given observation are given in this order in free format on individual lines in the data file. Thus, a line of the data file with three regressors having the values 0, 0, and 1 respectively for July of 1991 would have the form `1991 7 0 0 1`. All the user-defined regressors must be on the same record, and in the order of their appearance in the **user** argument (example: `format = "datevalue"`);

d. the "x13save" format `X-13ARIMA-SEATS` uses to save a table. This allows the user to read in a file saved from a previous `X-13ARIMA-SEATS` run (example: `format = "x13save"`);[21]

e. a variant of "free" format where the numbers must be separated by one or more spaces (not by commas or tabs), and decimal points are expressed as commas (a convention in some European countries). (example: `format = "freecomma"`);

f. a variant of "datevalue" format, where the year, month or quarter, and value of each observation are found in this order in free format on individual lines, where decimal points are expressed as commas. Thus, a line of the data file containing the value 1355.34 for July of 1991 would have the form `1991 7 1355,34`. The number of preceding blanks can vary (example: `format = "datevaluecomma"`).

If no **format** argument is given the data will be read in free format. The **format** argument cannot be used with the **data** argument, only with the **file** argument.

[19] See the end of Section 4.7 for a discussion of what to do about forecast extension for seasonal adjustment of a series with a model that contains user-defined regressors whose future values are unknown.

[20] See the end of Section 4.7 for a discussion of what to do about forecast extension for seasonal adjustment of a series with a model that contains user-defined regressors whose future values are unknown.

[21] Note that to maintain compatibility with previous versions of `X-12-ARIMA` the entry **x12save** will also be accepted.

print and **save** Table 7.25 gives the available output tables for this spec. All of these tables are included in the default printout, except `regressionmatrix` and `dailyweights`. Also, if there is only one type of outlier in the regARIMA model, then only the combined outlier table will print out, and the specific tables for the individual outlier effect (`ao`, `ls`, `tc`, `so`) will be suppressed. For a complete listing of the `brief` and `default` print levels for this spec, see Appendix B.

name	*short*	*save?*	*description of table*
regressionmatrix	rmx	+	values of regression variables with associated dates
aictest	ats	·	output from AIC-based test(s) for trading day, Easter, and user-defined regression variables
outlier	otl	+	combined regARIMA outlier factors (table A8)
aoutlier	ao	+	regARIMA additive (or point) outlier factors (table A8.AO)
levelshift	ls	+	regARIMA level shift, temporary level shift and ramp outlier factors (table A8.LS)
seasonaloutlier	so	+	regARIMA seasonal outlier factors (table A8.SO)
transitory	a13	+	regARIMA transitory component factors from user-defined regressors (table A13)
temporarychange	tc	+	regARIMA temporary change outlier factors (table A8.TC)
tradingday	td	+	regARIMA trading day factors (table A6)
holiday	hol	+	regARIMA holiday factors (table A7)
regseasonal	a10	+	regARIMA user-defined seasonal factors (table A10)
userdef	usr	+	factors from user-defined regression variables (table A9)
chi2test	cts	·	output from chi-squared based test for groups of user-defined regression variables
dailyweights	tdw	·	Daily weights from trading day regressors, normalized to sum to seven

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.25: **Available Output Tables for Regression**

pvaictest Probability used to generate a critical value for any AIC tests specified in this spec. This probablity must be > 0.0 and < 1.0. Table 7.26 shows the critical value generated for different values of `pvaictest` and different values of ν, the difference in the number of parameters between two models.

If this argument is not specified, the `aicdiff` argument is used to set the critical value for AIC testing. This argument cannot be used in the same spec file as the **aicdiff** argument.

savelog The diagnostics available for output to the log file (see section 2.6) are listed on Table

Value of pvaictest	$\nu = 1$	$\nu = 2$	$\nu = 3$	$\nu = 4$	$\nu = 5$	$\nu = 6$	$\nu = 7$
0.05	1.8415	1.9915	1.8147	1.4877	1.0705	0.5916	0.0671
0.01	4.6349	5.2103	5.3449	5.2767	5.0863	4.8119	4.4753
0.005	5.8794	6.5966	6.8382	6.8603	6.7496	6.5476	6.2777
0.001	8.8276	9.8155	10.2662	10.4668	10.5150	10.4577	10.3219

Table 7.26: **AIC Test Critical Values for Different Levels of `pvaictest` and ν**

7.27.

name	*short*	*description of diagnostic*
aictest	ats	test results from the AICC-based regressor selection procedure
chi2test	cts	test results from the Chi-squared based regressor selection procedure

Name gives the name of each diagnostic for use with the **savelog** argument.
Short gives a short name for these diagnostics.

Table 7.27: **Available Log File Diagnostics for Regression**

start The start date for the data values for the user-defined regression variables. The default is the start date of the series. Valid values are any date up to the start date of the series (or up to the start date of the span specified by the **span** argument of the **series** spec, if present).

testalleaster Specifies if an extra regression model is evaluated when more than one Easter regressor is specified in the **variables** argument. When `testalleaster = yes`, an additional regARIMA model is estimated that contains all Easter regressors specified by the user in the **variables** argument. An AICC diagnostic is generated from this model and used in the AIC-based testing procedure, as well as the AICCs for model with and without the individual Easter regressors. The default is `testalleaster = no` – only the individual Easter regressors specified by the user are used in the AIC testing procedure.

tlimit Sets the value to which the absolute values of the t-statistics of AO and LS sequence regressors are compared to retain those outliers in the regARIMA model. If this argument is not specified, AO and LS sequence regressors are not checked for significance.

user Specifies names for any user-defined regression variables. Names are required for all user-defined variables to be included in the model. The names given are used to label estimated coefficients in the program's output. Data values for the user-defined variables must be supplied, using either the **data** or **file** argument (not both). The maximum number of user-defined regression variables is 52. (This limit can be changed—see Section 2.8.)

usertype Assigns a type of model-estimated regression effect to each user-defined regression variable. It causes the variable and its estimated effects to be used and output in the same way as a predefined regressor of the same type. This option is useful when trying out alternatives to the regression effects provided by the program.

The type for a user-defined regression effect can be defined as a constant (**constant**), seasonal (**seasonal**), trading day (**td**), length-of-month (**lom**), length-of-quarter (**loq**), leap year (**lpyear**), outlier (**ao**, **ls**, or **so**), a user-defined transitory component for SEATS (**transitory**), or other user-defined (**user**) regression effects. In addition to the aforementioned types, users can also specify up to 5 different user-defined holidays (**holiday**, **holiday2**, **holiday3**, **holiday4**, and **holiday5**). This gives the user flexibility in specifying more than one holiday, and the chi-squared statistic is generated separately for these user-defined holidays.

One effect type can be specified for all the user-defined regression variables defined in the **regression** spec (`usertype = td`), or each user-defined regression variable can be given its own type (`usertype = (td td td td td td holiday user)`). Once a type other than **user** has been assigned to a user-defined variable, further specifications for the variable in other arguments, such as **aictest** or **noapply**, must use this type designation and not **user**. If this option is not specified, all user-defined variables have the type **user**. See DETAILS for more information on assigning types to user-defined regressors.

variables List of predefined regression variables to be included in the model. Data values for these variables are calculated by the program, mostly as functions of the calendar. See DETAILS for a discussion and a table of the available predefined variables. Also see Section 4.3 for additional information and a table defining the actual regression variables used.

RARELY USED ARGUMENTS

b Specifies initial values for regression parameters in the order that they appear in the **variables** and **user** arguments. Values may be specified for some or all of the regression coefficients. Values followed immediately by an 'f' will be held fixed in the model estimation; all other coefficients will be estimated in the GLS regression done for the model fitting. Thus, the sole reason for specifying any values of **b** is to hold those regression coefficients fixed when the model is fitted. E.g., if one specifies `b = (0.3,,0.7f)`, this is equivalent to specifying `b = (,,0.7f)` – the first and second coefficients will be estimated by GLS regression (so specifying the 0.3 is unnecessary), while the third coefficient is fixed at 0.7 throughout the model estimation, outlier detection, forecasting, etc.

centeruser Specifies the removal of the (sample) mean or the seasonal means from the user-defined regression variables. If `centeruser = mean`, the mean of each user-defined regressor is subtracted from the regressor. If `centeruser = seasonal`, means for each calendar month (or quarter) are subtracted from each of the user-defined regressors. If this option is not specified, the user-defined regressors are assumed to already be in an appropriately centered form and are not modified.

eastermeans Specifies whether the monthly means used to remove seasonality from the Easter regressor associated with the variable `easter[w]` are the long term (500 year) monthly means, as described in footnote 8 of Table 4.1 (`eastermeans = yes`), or the monthly means calculated from just the span of data used for calculating the coefficients of the Easter regressor (`eastermeans = no`). The default is `eastermeans = yes`. This argument is

ignored if no built-in Easter regressor is included in the regression model, or if the only Easter regressor is `sceaster[w]` (see DETAILS).

noapply List of the types of regression effects defined in the **regression** spec whose model-estimated values are *not* to be removed from the original series before the seasonal adjustment calculations specified by the **x11** spec are performed.

Applicable types are all modeled trading day effects (**td**), Easter, Labor Day, and Thanksgiving–Christmas holiday effects (**holiday**), point outliers (**ao**), level shifts and ramps (**ls**), temporary changes (**tc**), seasonal outliers (**so**), user-defined seasonal regression effects (**userseasonal**), and the set of user-defined regression effects (**user**).

tcrate Defines the rate of decay for the temporary change outlier regressor. This value must be a number greater than zero and less than one. The default value is `tcrate = 0.7 ** (12 / period)`, where period is the number of observations in one year (12 for monthly time series, 4 for quarterly time series. This formula for the default value of **tcrate** ensures the same rate of decay over an entire year for series of different periodicity. If the frequency of the time series is less than 4 (ie, period < 4), then there is no default value, and the user will have to enter a value of **tcrate** if a temporary change outlier was specified in the **variables** argument.

DETAILS

If forecasting is performed, `X-13ARIMA-SEATS` creates data values for the selected predefined regression variables for the entire forecast period. If there are any user-defined regression variables, then data values must also be supplied for them for the entire forecast period (similarly for the backcasts).[22] In addition to the limit of 52 user-defined regression variables, there is an overall limit of 80 regression variables in the model. (These limits can be changed—see Section 2.8.) The latter limit is on the total number of predefined and user-defined regression variables plus the number of regression variables added automatically by the outlier spec. The maximum length of the series of user-defined regression variables, not including the forecast period, is 780. (This limit can also be changed—see Section 2.8.)

Table 7.28: **Predefined Regression Variables**

Variable	Description
const	Trend constant regression variable to allow for a nonzero overall mean for the differenced data.
seasonal	Fixed seasonal effects parameterized via $s-1$ seasonal contrast variables (s = seasonal period). The resulting variables allow for month-to-month (or quarter-to-quarter, etc.) differences in level, but have no net effect on overall level. Cannot be used with `sincos` or in models with seasonal differencing except as a partial change of regime variable (see DETAILS where additional change of regime options are described, as in Table 7.29).

[22] See the end of Section 4.7 for a discussion of what to do about forecast extension for seasonal adjustment of a series with a model that contains user-defined regressors whose future values are unknown.

Table 7.28: **Predefined Regression Variables** (continued)

Variable	Description
sincos[ ]	Fixed seasonal effects (for s = seasonal period) parameterized via trigonometric regression variables of the form $\sin(\omega_j t)$ and $\cos(\omega_j t)$ at seasonal frequencies $\omega_j = (2\pi j/s)$ for $1 \leq j \leq s/2$ (dropping $\sin(\omega_j t) \equiv 0$ for $j = s/2$ for s even). Each frequency to be included must be specified, i.e., for monthly series `sincos[1, 2, 3, 4, 5, 6]` includes all seasonal frequencies while `sincos[1, 2, 3]` includes only the first three. Cannot be used with `seasonal` or in models with seasonal differencing.
td	Estimate monthly (or quarterly) flow trading-day effects by including the `tdnolpyear` variables (see below) in the model, and handle leap-year effects either by re-scaling (for transformed series) or by including the `lpyear` regression variable (for untransformed series). Can only be used for monthly or quarterly series, and cannot be used with `tdnolpyear`, `td1coef`, `td1nolpyear`, `lpyear`, `lom`, `loq`, `tdstock`, or `tdstock1coef`. If `td` is specified, do not specify `adjust = lpyear` or `adjust = lom` (`adjust = loq`) in the **transform** spec. Several change of regime options are described in DETAILS, as in Table 7.29.
tdnolpyear	Include the six day-of-week contrast variables (monthly and quarterly flow series only): (no. of *Mondays*) − (no. of *Sundays*), ..., (no. of *Saturdays*) − (no. of *Sundays*). Cannot be used with `td`, `td1coef`, `td1nolpyear`, `tdstock`, or `tdstock1coef`. Several change of regime options are described in DETAILS, as in Table 7.29.
td1coef	Estimate monthly (or quarterly) flow trading-day effects by including the `td1nolpyear` variable (see below) in the model, and handle leap-year effects either by re-scaling (for transformed series) or by including the `lpyear` regression variable (for untransformed series). Can only be used for monthly or quarterly series, and cannot be used with `td`, `tdnolpyear`, `td1nolpyear`, `lpyear`, `lom`, `loq`, `tdstock`, or `tdstock1coef`. If `td1coef` is specified, do not specify `adjust = lpyear` or `adjust = lom` (`adjust = loq`) in the **transform** spec. Several change of regime options are described in DETAILS, as in Table 7.29.
td1nolpyear	Include the weekday-weekend contrast variable (monthly and quarterly flow series only): (no. of *weekdays*) $-\frac{5}{2}$ (no. of *Saturdays and Sundays*). Cannot be used with `td`, `tdnolpyear`, `td1coef`, `tdstock`, or `tdstock1coef`. Several change of regime options are described in DETAILS, as in Table 7.29.
lpyear	Include a contrast variable for leap-year (monthly and quarterly flow series only): 0.75 for leap-year Februaries (first quarters), -0.25 for non-leap year Februaries (first quarters), and 0.0 otherwise. Cannot be used with `td`, `td1coef`, `tdstock`, or `tdstock1coef`. Several change of regime options are described in DETAILS, as in Table 7.29.
lom	Include length-of-month as a regression variable. If `lom` is requested for a quarterly series, `X-13ARIMA-SEATS` uses `loq` instead. Requesting `lom` when s is neither 12 nor 4 produces an error. Cannot be used with `td`, `td1coef`, `tdstock`, or `tdstock1coef`. Several change of regime options are described in DETAILS, as in Table 7.29.
loq	Include length-of-quarter as a regression variable. If `loq` is requested for a monthly series, `X-13ARIMA-SEATS` uses `lom` instead. The same restrictions that apply to `lom` apply to `loq`. Several change of regime options are described in DETAILS, as in Table 7.29.
tdstock[w]	Estimate day-of-week effects for inventories or other stocks reported for the w-th day of each month. The value w must be supplied and can range from 1 to 31. For any month of length less than the specified w, the `tdstock` variables are measured as of the end of the month. Use `tdstock[31]` for end-of-month stock series. Can only be used with monthly series and cannot be used with `tdstock1coef`, `td`, `tdnolpyear`, `td1coef`, `td1nolpyear`, `lom`, or `loq`.

Table 7.28: **Predefined Regression Variables** (continued)

Variable	Description
tdstock1coef[w]	Estimate a constrained stock trading day effect for inventories or other stocks reported for the w-th day of each month. The value w must be supplied and can range from 1 to 31. For any month of length less than the specified w, the `tdstock1coef` variables are measured as of the end of the month. Use `tdstock1coef[31]` for end-of-month stock series. Can only be used with monthly series and cannot be used with `tdstock`, `td`, `tdnolpyear`, `td1coef`, `td1nolpyear`, `lom`, or `loq`.
easter[w]	Easter holiday regression variable for monthly or quarterly flow data that assumes the level of daily activity changes on the w-th day before Easter and remains at the new level through the day before Easter. This value w must be supplied and can range from 1 to 25. A user can also specify an **easter**[0] regression variable, which assumes the daily level of activity level changes only on Easter Sunday. To estimate complex effects, several of these variables, differing in their choices of w, can be specified.
labor[w]	Labor Day holiday regression variable (monthly flow data only) that assumes the level of daily activity changes on the w-th day before Labor Day and remains at the new level until the day before Labor Day. The value w must be supplied and can range from 1 to 25.
thank[w]	Thanksgiving holiday regression variable (monthly flow data only) that assumes the level of daily activity changes on the w-th day before or after Thanksgiving and remains at the new level until December 24. The value w must be supplied and can range from -8 to 17. Values of $w < 0$ indicate a number of days after Thanksgiving; values of $w > 0$ indicate a number of days before Thanksgiving.
sceaster[w]	Statistics Canada Easter holiday regression variable (monthly or quarterly flow data only) assumes that the level of daily activity changes on the $(w-1)$-th day before Easter and remains at the new level through Easter day. The value w must be supplied and can range from 1 to 24. To estimate complex effects, several of these variables, differing in their choices of w, can be specified.
easterstock[w]	End of month stock Easter holiday regression variable for monthly or quarterly stock data. This regressor is generated from the `easter[w]` regressors. The value w must be supplied and can range from 1 to 25. To estimate complex effects, several of these variables, differing in their choices of w, can be specified.
ao*date*	Additive (point) outlier variable, AO, for the given date or observation number. For series with associated dates, AOs are specified as `ao`*date*. For monthly series this is `ao`*year.month* (e.g., `ao1985.jul` or `ao1985.7`), while for quarterly series this is `ao`*year.quarter* (e.g., `ao1985.1` for an AO in the first quarter of 1985), and for annual series this is `ao`*year* (e.g., `ao1922`). For series without associated dates, AOs are specified as `ao`*observation number*, e.g., `ao50` for an AO at observation 50. More than one AO may be specified. All specified outlier dates must occur within the series. (AOs with dates within the series but outside the span specified by the **span** argument of the **series** spec are ignored.)
aos*date-date*	Specifies a sequence of additive (point) outlier variables, AO, for the given range of dates or observation numbers. Sequence AO outlers begin and end on a given date, e.g., `aos2008.apr-2008.oct`. To have the sequence run through the end of the series, specify `0.0` as the end date. More than one AOS may be specified, although the spans should not overlap. All specified outlier dates must occur within the series. (AOSs with dates within the series but outside the span specified by the **span** argument of the **series** spec are ignored.)

Table 7.28: **Predefined Regression Variables** (continued)

Variable	Description
ls*date*	Regression variable for a constant level shift (in the transformed series) beginning on the given date, e.g., `ls1990.oct` for a level shift beginning in October 1990. More than one level shift may be specified. Dates are specified as for AOs and the same restrictions apply with one addition: level shifts cannot be specified to occur on the start date of the series (or of the span specified by the **span** argument of the **series** spec).
lss*date-date*	Specifies a sequence of level shift outlier variable, LS, for the given range of dates or observation numbers. Sequence LS outlers begin and end on a given date, e.g., `lss2008.jun-2008.nov`. To have the sequence run through the end of the series, specify `0.0` as the end date. More than one LSS may be specified, though the spans should not overlap. All specified outlier dates must occur within the series. (LSSs with dates within the series but outside the span specified by the **span** argument of the **series** spec are ignored.)
tc*date*	Regression variable for a temporary change (in the transformed series) beginning on the given date, e.g., `tc1990.oct` for a temporary change beginning in October 1990. More than one temporary change may be specified. Dates are specified as for AOs, and the same restrictions apply.
so*date*	Regression variable for a seasonal outlier (in the transformed series) beginning on the given date, e.g., `so1988.mar` for a seasonal outlier beginning in March 1988. More than one seasonal outlier may be specified. Dates are specified as for AOs, and the same restrictions apply with one addition: seasonal level shifts cannot be specified to occur on the start date of the series (or of the span specified by the `span` argument of the `series` spec).
rp*date-date*	Ramp effect that begins and ends on the given dates, e.g., `rp1988.apr-1990.oct`. The rate of change during the ramp for this regression variable is constant. More than one ramp effect may be specified. All dates of the ramps must occur within the series. (Ramps specified within the series but with both start and end dates outside the span specified by the **span** argument of the **series** spec are ignored.) Ramps can overlap other ramps, TLs, AOs, and level shifts.
qd*date-date*	Quadratic ramp effect that begins and ends on the given dates, e.g., `qd1998.may-2000.aug`. The rate of change during the ramp for this regression variable is decreasing in magnitude. More than one quadratic ramp effect may be specified. All dates of the ramps must occur within the series. (Ramps specified within the series but with both start and end dates outside the span specified by the **span** argument of the **series** spec are ignored.) Quadratic ramps can overlap other ramps, TLs, AOs, and level shifts.
qi*date-date*	Quadratic ramp effect that begins and ends on the given dates, e.g., `qi2010.apr-2011.oct`. The rate of change during the ramp for this regression variable is increasing in magnitude. More than one quadratic ramp effect may be specified. All dates of the ramps must occur within the series. (Ramps specified within the series but with both start and end dates outside the span specified by the **span** argument of the **series** spec are ignored.) Quadratic ramps can overlap other ramps, TLs, AOs, and level shifts.
tl*date-date*	Temporary level shift effect which begins and ends on the given dates, e.g., `tl1983.jul-1984.nov`. More than one temporary level shift effect may be specified. All dates of the temporary level shift regressor must occur within the series. (Temporary level shifts specified within the series but with start or end dates outside the span specified by the **span** argument of the **series** spec are ignored.) Temporary level shifts can overlap other TLs, ramps, AOs, and level shifts.

If **const** is specified in the **variables** argument, then the resulting regression variable allows for a constant

term in the series resulting from any differencing operations in the ARIMA model. If the ARIMA model involves no differencing, this is simply the usual regression constant term for a nonzero overall mean; if the ARIMA model does involve differencing, this regressor is called a trend constant. In the latter case the actual regression variable created is defined such that, after differencing, it yields a column of ones. See Section 4.3 for discussion.

We generally recommend specifying **td** in the **variables** argument when trading-day effects are thought to be present in a monthly *flow* time series – that is, a series whose values are monthly accumulations of daily values. In this case, how the program handles leap-year effects depends on information from the **transform** spec. If the series is transformed (Box-Cox or logistic transformation), then leap-year effects are removed by prior adjustment: the series is divided before transformation by a set of factors lp_t where $lp_t = 28.25/29$ if t is a leap year February, $lp_t = 28.25/28$ if t is a non-leap year February, and $lp_t = 1.00$ otherwise.

If the series is not transformed, then the leap-year regression variable **lpyear** is included in the model. Its values, denoted by LP_t are given by $LP_t = 29 - 28.25$ if t is a leap year February, $LP_t = 28 - 28.25$ if t is a non-leap year February, and $LP_t = 0.00$ otherwise. In both cases, the **tdnolpyear** regression variables, (no. of *Mondays*) − (no. of *Sundays*), ..., (no. of *Saturdays*) − (no. of *Sundays*), are also included in the model. Leap year effects are the nonseasonal component of length-of-month effects. When `type = trend` is used in the **x11** spec, with the result that there is no seasonal effect estimation and adjustment, then **td** handles length-of-month effects instead of leap-year effects. That is, with a transformation, there is prior adjustment by the length-of-month factors described in Table 7.28, and with no transformation, the **lom** regressor, whose value is the number of days in the month, is added to the regression with the **tdnolpyear** regressors.

In any situation in which the user prefers to model length-of-month effects in a transformed series, the leap year regressor is the nonseasonal component for the length-of-month (quarter) regressor. If the user prefers to model length-of-month effects in a transformed series through the **lom** regression variable, this can be done by specifying both **lom** and **tdnolpyear**, i.e., `variables = (lom tdnolpyear ...)`. If the user prefers to prior adjust an untransformed series for length-of-month effects, this can be done by specifying `variables = (tdnolpyear ...)` in the **regression** spec and `adjust = lom` in the **transform** spec.

If `adjust = lom` is specified in the **transform** spec, then including either **td** or **lom** in the **variables** list leads to a conflict. The conflict occurs either because two requests have been made to re-scale the series by dividing by length of month, or because both a length-of-month rescaling and the **lom** regression variable have been requested (which will generally lead to a singular system of equations for the regression coefficients). In this case, the user should either (*i*) remove `adjust = lom` from the **transform** spec, or (*ii*) in the **variables** list, replace **td** by **tdnolpyear**, or drop **lom**.

For quarterly flow time series, the same trading-day options are available, and the above comments apply with **lom** replaced by **loq**.

The values **lom** and **loq** are equivalent – if either is specified, the seasonal period specified in the **series** spec determines which is used. Thus, `period = 12` implies **lom** and `period = 4` implies **loq**. Also, note that **lom** or **loq** can be specified without **tdnolpyear**. This could be done to account for fixed seasonality due to length-of-month (or length-of-quarter) effects for a series with no day-of-week specific effects. Predefined length-of-period variables are available only for monthly or quarterly flow series.

For *stock* series, such as inventories, the program can estimate trading-day effects only for monthly series. **Tdstock**[w], where w can range from 1 to 31, creates six regression variables contrasting six days of the week with the seventh – see Section 4.3. The value w must be specified; it denotes the day of the month for which the stock is reported or the last day of the month, whichever is smaller. Therefore, **tdstock[31]** is used for end-of-month stocks.

The holiday effect regression variables (for Easter, Labor Day, and Thanksgiving) are for flow series. The Easter variable can be specified for either monthly or quarterly series. The Labor Day and Thanksgiving variables are only for monthly series.

If a series is designated as a stock or a flow series by using the **type** argument of the **series** or **composite** spec, then trading day and Easter regressors specified in **variables** argument need to agree with this type – one cannot specify stock trading day regressors for a flow series. If a series type is not specified, then any trading day or holiday regressor may be used with the series.

Change of regime regression variables can be specified for seasonal (**seasonal**), trigonometric seasonal (**sincos**), trading day (**td**, **tdnolpyear**, **tdstock**, **td1coef**, **td1nolpyear**, or **tdstock1coef**), leap year (**lpyear**), length-of-month (**lom**), and length-of-quarter (**loq**) regression variables. Two types of change of regime regressors are available: full and partial.

As Table 7.29 shows, change of regime regressors are specified by appending the change date, surrounded by one or two slashes, to the name of a regression variable in the **variables** argument of the **regression** spec. The date specified for the change of regime divides the series being modeled into two spans, an early span containing the data for times prior to the change date and a late span containing the data from on and after this date. Partial change of regime variables are restricted to one of these two spans, being zero in the complementary span. The full change of regime variables estimate both the basic regression of interest and the partial change of regime regression for the early span. For example, the full change of regime specification `variables = (td/1990.jan/)` is equivalent to the specification `variables = (td td/1990.jan//)`. It causes the program to output the coefficients estimated for **td** and for **td/1990.jan//** along with trading day factors for their combined effects.

Type	*Syntax*	*Example*
Full change of regime regressor	*reg/date/*	**td/1990.jan/**
Partial change of regime regressor, zero before change date	*reg//date/*	**td//1990.jan/**
Partial change of regime regressor, zero on and after change date	*reg/date//*	**td/1990.jan//**

Table 7.29: **Change of Regime Regressor Types and Syntax**

The coefficients resulting from use of a full change of regime regression have convenient interpretations. Let the basic regressors be denoted by X_{jt}, and let t_0 be the change point. Then the partial change of regime regressors for the early regime are

$$X^E_{jt} = \begin{cases} X_{jt} & \text{for } t < t_0 \\ 0 & \text{for } t \geq t_0 \end{cases}$$

and those for the late regime can be calculated as $X^L_{jt} = X_{jt} - X^E_{jt}$. For the data transformed as indicated in the **transform** spec, the effect estimated by the full change of regime regression has the form

$$\sum_j a_j X_{jt} + \sum_j b_j X^E_{jt} = \sum_j a_j X^L_{jt} + \sum_j (a_j + b_j) X^E_{jt}.$$

From the right-hand side formula, we observe that the coefficients a_j of the basic regressors X_{jt} can be interpreted as the coefficients of the late-span regressors X^L_{jt}, and the coefficients b_j of the X^E_{jt} can be interpreted

as measuring the change in the coefficients of the late-span regressors required to obtain coefficients for the early-span effects. Therefore, statistically significant b_j indicate the nature of the change of regime.

We illustrate two other natural uses for partial change of regime variables. First, the specification `variables = (td//1990.jan/)` can be used to estimate the trading day component of a series that has no statistically significant trading day effects prior to 1990, but possibly significant effects beginning in that year. Second, when an ARIMA model with seasonal differencing is specified in the **arima** spec, or in the models estimated by the **automdl** spec, then the specification `variables = (seasonal//1990.jan/)` can be used to estimate a fixed change in a somewhat variable seasonal pattern that takes place in January of 1990 and to test for the statistical significance of the estimated change.

The effect of the argument **aictest** can be to delete a regressor set named in the **variables** list from this list, or to add a regressor set to the **variables** list. The effect of a nonzero (positive) value of **aicdiff** is to make it more difficult for the **aictest** procedure to add the variable being tested to the current model. Let Δ_{AIC} denote the value associated with the **aicdiff** argument, which by default is zero. Let $AICC^{with}$ ($AICC^{without}$) denote the AICC value of the model with (without) a set of regressors specified in the **aictest** argument. If this set is not named in the **variables** list, it will be added to the regression model if

$$AICC^{with} + \Delta_{AIC} < AICC^{without}.$$

If this set is named in the variables list, it will be retained in the regARIMA model only if this inequality holds.

In the second case, if `aictest = (tdstock)`, then the end-of-month stock variables, specified by **tdstock**[**31**], are the variables being added, because 31 is the default value for w in **tdstock**[w].

There are more possibilities if `aictest = (easter)` and no Easter effect regressors appear in the **variables** list. Then three additional models are considered – the three models obtained by augmenting the specified regARIMA model with the regressor **easter**[w] for $w = 1, 8, 15$, respectively. The Easter regressor whose model has the smallest AICC is retained if its AICC is smaller than the model with no Easter regressors by at least the amount Δ_{AIC}; otherwise, the model without Easter regressors is selected.

Previous simulation experiments suggest that AICC does not distinguish with high reliability between **easter**[w] regressors whose w values differ by less than seven. The out-of-sample forecast diagnostics produced by the **history** spec can sometimes distinguish between such regressors by showing that one provides persistently more accurate forecasts, and therefore presumably better describes the Easter effect in the data.

Similar to the case for Easter, when `aictest = (td)` and no trading day regressors appear in the **variables** list, then additional models are considered. These are the models obtained by augmenting the specified regARIMA model with full and one coefficient trading day regressors, depending on the type of series (**td** and **td1coef** for flow series, **tdstock** and **tdstock1coef** for stock series).

The trading day regressor whose model has the smallest AICC is retained if its AICC is smaller than the model with no trading day regressors by at least the amount Δ_{AIC}; otherwise, the model without trading day regressors is selected.

When regressors appear in both the **aictest** and **variables** arguments, the regressors specified should have identical types. An exception for this is for trading day regressors. The entry `aictest = td` serves as a correct entry for any type of flow or stock trading day regressor. The sample day for stock trading day variables and the

date specified for change of regime regressors should *not* be included in the **aictest** argument; its value will be taken from the entry in the **variables** argument. For example, if `variables = (tdstock[15] ao1995.jan)`, then the entry for **aictest** can be `tdstock` or `td`.

Another exception is for Easter regressors. The entry `aictest = easter` serves as a correct entry for any type of flow or stock Easter regressor. The window length of the Easter regressor should *not* be included in the **aictest** argument; it will be determined by the entry in the **variables** argument. For example, if `variables = (easterstock[8] ao1992.aug)`, then either `easterstock` or `easter` is an acceptable entry for **aictest**.

Note that this is not affected by setting `type = stock` or `type = flow` in the **series** or **composite** specs; the entries `aictest = td` and `aictest = easter` can still be used for both stock and flow series. However, you cannot set `type = flow` in the **series** spec and then have `aictest = tdstock`.

Regressors specified by the **aictest** argument must also be able to be included with other regressors specified either in the **variables** and the **aictest** arguments. For example, the following **regression** spec is incorrectly specified, as the **td** and **lom** arguments cannot be specified together in the **variables** argument:

```
regression{
   variables = td
   aictest = lom
}
```

Using **tdnolpyear** instead would allow a model with the 6 trading day regressors and the length-of-month regressors.

In addition, users should not specify `aictest = lom` for series that are not monthly series, and `aictest = loq` for series that are not quarterly.

As mentioned above, trading day regressors are always tested before length-of-month (-quarter) or leap year regressors. If options specified in the **regression** spec lead to trading day and leap year regressors in the same regARIMA model, then the program will test the trading day and leap year regressors together if `aictest = td` is specified, but will test the sets of regressors separately if `aictest = (td lpyear)` is specified.

User-defined variables should be input to the program in deseasonalized form (unless they are seasonal regressors). The deseasonalization method described in Section C.1.3 is likely to be the appropriate one, because regressors are additive components of the regARIMA model. If deseasonalization is not done, then the seasonal factors will not include all estimated seasonal effects. Another problem is that regressors with seasonal components are likely to have estimated coefficients, and estimated effects, that are more correlated with one another and therefore more difficult to interpret.

If a type is assigned to a user-defined variable with the **usertype** argument, the factor derived from the user-defined regression variables of that type will be combined with the regression factor from variables of the same type specified in the **regression** spec. The resulting factor will be adjusted out of the series for the seasonal adjustment factor calculations determined by the **x11** or **seats** spec unless the type name appears in the **noapply** argument.

Setting `usertype = seasonal` will cause seasonal factors to be created from the user-defined regressors that will be adjusted out of the original series before the seasonal adjustment specified by the **x11** or **seats** spec is calculated. Combined seasonal factors are created from the `X-11` or `SEATS` and regression factors. In addition,

if `noapply = userseasonal` is specified, the user-defined seasonal regressors are treated exactly like seasonal regressors specified in the **variables** argument: the seasonal effect estimated from these regressors will not adjusted out of the series prior to seasonal adjustment. The effects estimated by Table 7.28 seasonal regressors specified in the **variables** argument are not available as output. If it is desired to remove these effects from the series prior to seasonal adjustment, this can be done by setting `save = rmx` to save the regressors in an output file. From this file, the regressors can be input to the program as user-defined regressors with `usertype = seasonal` to achieve the desired removal.

Note that if `format = "datevalue"` or `format = "x13save"`, the starting date of the user-defined regressor(s) is automatically read from the data file. Therefore, the starting date need not be specified with the **start** argument of the **regression** spec.

Trading day and/or holiday regressors may not be specified in the **regression** and **x11regression** specs simultaneously unless the **noapply** option is used to specify that the effects estimated by either the **regression** or **x11regression** spec not be used to adjust the series.

The two choices for the argument **eastermeans** yield noticeably different holiday factors. But the choice has no effect on forecasts (provided the regARIMA model used includes seasonal differencing or the fixed seasonal regressors) and usually has only negligible effects on the combined seasonal and holiday factors, because the seasonal factors change to compensate for the differences between the choices.

Table 7.30 gives the monthly means for February, March, and April that are used to obtain deseasonalized Easter regressors under `eastermeans = yes`; the means for other months are zero. These calendar means were generated from frequencies of the date of Easter for a 500 year period (1600–2099). These frequencies were computed from dates given in Bednarek (2019), which were checked using information from Montes (2001, 1997b, 1997a); the algorithm used by Montes to compute the date of Easter for the Gregorian calendar is given in Duffet-Smith (1981).

For quarterly series, the mean of the first quarter is equivalent to the sum of the February and March means from Table 7.30, the mean for the second quarter is equivalent to the April mean, and the means for other quarters are zero.

For a nonseasonal time series, an adjustment for trading day and holiday effects estimated by means of this spec can be obtained by setting `type = trend` in the **x11** spec.

Regarding the outlier regressors, users should be aware that several combinations of AOs and LSs produce arithmetically equal effects. For example, (*i*) an AO at time t_0 followed by an LS at $t_0 + 1$; (*ii*) LSs at both t_0 and $t_0 + 1$; (*iii*) both an AO and an LS at t_0. Note that an LS at t_0 followed by an AO at $t_0 + 1$ is not equivalent to these other combinations.

Because AOs are assigned to the irregular component and LSs to the trend-cycle, some users might prefer one equivalent combination over another.

When the **b** argument is used to fix coefficients, *AIC* and the other model selection statistics may become invalid – see the DETAILS section of **estimate**.

For more information concerning the modeling of holiday effects and the detection and modeling of trading day effects, see Findley and Soukup (2000) and Lin and Liu (2002).

Easter effect length (w)	February Mean	March Mean	April Mean
0	0.000000	0.232000	0.768000
1	0.000000	0.266000	0.734000
2	0.000000	0.281000	0.719000
3	0.000000	0.296667	0.703333
4	0.000000	0.312500	0.687500
5	0.000000	0.330400	0.669600
6	0.000000	0.348333	0.651667
7	0.000000	0.365429	0.634571
8	0.000000	0.382000	0.618000
9	0.000000	0.397556	0.602444
10	0.000000	0.413600	0.586400
11	0.000000	0.430546	0.569455
12	0.000000	0.447667	0.552333
13	0.000000	0.464308	0.535692
14	0.000000	0.480714	0.519286
15	0.000000	0.497333	0.502667
16	0.000000	0.514625	0.485375
17	0.000000	0.531882	0.468118
18	0.000000	0.549000	0.451000
19	0.000000	0.566105	0.433895
20	0.000000	0.583000	0.417000
21	0.000000	0.599905	0.400095
22	0.000273	0.616273	0.383455
23	0.001130	0.631130	0.367739
24	0.002083	0.645083	0.352833
25	0.003680	0.657600	0.338720

Table 7.30: **500 Year (1600–2099) Means for Easter Regressors with Window Length w.**

EXAMPLES

The following examples show complete spec files.

Example 1 Estimate a model with ARIMA (0 1 1) errors, fixed seasonal effects, and a trend constant.

```
SERIES      { TITLE = "Monthly sales"  START = 1996.JAN
              DATA = (138 128 ... 297) }
REGRESSION { VARIABLES = (CONST SEASONAL) }
ARIMA { MODEL = (0 1 1) }
ESTIMATE { }
```

Example 2 Specify a model to fit sine and cosine variables with the 4th and 5th seasonal frequency by ordinary least squares to the final irregular component of a series to test if "visually significant" spectrum peaks at these frequencies are statistically significant.

```
series { title = "Irregular Component of Monthly Sales"
         start = 1996.jan
         file = "sales.d13"
         format = "x13save"
       }
regression { variables = (const sincos[4,5]) }
estimate { }
spectrum { savelog=peaks }
```

Example 3 Specify regression variables for trading-day, Easter, Labor Day, and Thanksgiving effects in a monthly time series. The duration in number of days is specified for each holiday effect. Since `td` is specified and the series is log transformed, the original series (before transformation) is divided by the leap-year factors, and the `tdnolpyear` regression variables are fit to the transformed series. The regression coefficients are estimated by the **identify** spec through a regression of the maximally differenced series (after transformation and leap-year adjustment) on the correspondingly differenced regression variables. The **identify** spec then produces various sample ACFs and PACFs (of the regression residuals) to be used for identifying an ARIMA model for the regression errors.

```
Series { Title = "Monthly Sales"  Start = 1996.Jan
          Data = (138 128 ... 297) }
Transform { Function = Log }
Regression { Variables = (TD Easter[8] Labor[10] Thank[3]) }
Identify { Diff = (0 1) SDiff = (0 1) }
```

Example 4 Estimate a model including the same regressors as in Example 3, and also the `lom` regression variable in place of the division of the series by standard leap-year effects that the argument value `td` invokes. (Replacing the value of `td` with `tdnolpyear` prevents the division by the standard leap year effects.) Perform a test (using AICC) of the significance of the trading-day and Easter regressors. Note that the program will test the significance of the 6 trading-day regressors first, then the significance of the length-of-month regressor, and finally the significance of the Easter regressor. An ARIMA $(0\ 1\ 1)(0\ 1\ 1)_{12}$ model is used for the regression error series.

```
series      { title = "Monthly sales"  start = 1996.jan
              data = (138 128 ... 297) }
transform { function = log }
regression { variables = (tdnolpyear lom easter[8] labor[10] thank[3])
             aictest = (lom td easter) }
arima { model = (0 1 1)(0 1 1) }
estimate    { }
```

Example 5 Specified regression variables are a one coefficient stock trading day regressor and an end-of-month stock Easter regressor. Since the sample day specified in the trading day regressor is 31, it is an end-of-month stock regressor as well. Decide (using AICC) if the stock trading-day and Easter regressors should be kept in the model.

```
series     {  title = "Retail inventory of food products"
              start = 1990.jan  data = "foodri.dat"  type = stock
           }
regression {  variables = ( tdstock1coef[31]  easterstock[8] )
              aictest = ( td easter )
           }
arima      {  model = (0 1 1)(0 1 1)                }
x11        {                                        }
```

Example 6 Estimate a model with trading-day effects, two AOs, and a ramp outlier for a quarterly seasonal series. Accounting for these effects, the transformed series follows an ARIMA $(0\ 1\ 1)(0\ 1\ 1)_4$ model.

```
Series     { Title  = "Quarterly Sales"  Start = 1990.1  Period = 4
             Data  = (1039 1241 ...  2210)  }
Transform  { Function = Log  }
Regression { Variables = (AO2007.1 RP2005.2-2005.4 AO1998.1 TD) }
Arima      { Model = (0 1 1)(0 1 1) }
Estimate   { }
```

Example 7 Same as Example 6, but using an increasing quadratic ramp instead of the linear ramp regressor.

```
Series     { Title  = "Quarterly Sales"  Start = 1990.1  Period = 4
             Data  = (1039 1241 ...  2210)  }
Transform  { Function = Log  }
Regression { Variables = (AO2007.1 QI2005.2-2005.4 AO1998.1 TD) }
Arima      { Model = (0 1 1)(0 1 1) }
Estimate   { }
```

Example 8 Estimate a user-defined regression variable for a temporary level shift from the third quarter of 1985 through the first quarter of 1987. The effect of the temporary level shift is removed through the regression performed by the **identify** spec, prior to the computation of ACFs and PACFs for identification of the ARIMA part of the model.

```
series {title = "Quarterly sales"  start = 1981.1
         data = (301 294 ...  391)  period = 4  }
regression {user = tls
            data = (0 0 0 0 0 0 0 0 0 0 0 0 ...
                    0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 ... 0) }
identify   { diff = (0 1) sdiff = (0 1) }
arima   { model = (0 1 1)(0 1 1) }
estimate   { }
```

Example 9 Same as Example 8, except that the built-in temporary level shift regressor is used.

```
series {title = "Quarterly sales"  start = 1981.1
         data = (301 294 ...  391)  period = 4  }
regression {  variables = tl1985.03-1987.01  }
identify   { diff = (0 1) sdiff = (0 1) }
arima   { model = (0 1 1)(0 1 1) }
estimate   { }
```

Example 10 Estimate a model that involves a constant, fixed seasonal effects, and two user-defined regression variables. The data for the latter two variables is stored in the file `weather.dat` in the current directory. This file includes data on several other variables not being used in the model. The data for the two user-defined regression variables is extracted from this larger file using a Fortran format that skips the first 16 columns in the file. The start date is specified since the data set of user-defined regression variables begins before the data for the time series being modelled.

```
series { title = "Monthly Riverflow"      start = 1990.1
         data  = (8.234 8.209 ...  8.104) period = 12 }
regression { variables = (seasonal const)
             user = (temp precip)
             file = "weather.dat"
             format = "(t17,2f8.2)"
             start = 1980.1 }
arima { model = (3 0 0)(0 0 0) }
estimate { }
```

Example 11 Estimate a model for a monthly retail inventory series with end-of-month stock trading-day effects and one AO. The transformed series, minus the regression effects, follows an ARIMA $(0\ 1\ 0)(0\ 1\ 1)_{12}$ model. Decide (using AICC) if the stock trading-day regressors should be kept in the model.

```
series {title = "Retail Inventory - Family Apparel"
        start = 1997.1  period = 12
        data = (1893 1932 ... 3201 )
        type = stock }
transform {   function = log }
regression {  variables = (tdstock[31] ao2010.jul)
              aictest=tdstock }
arima { model = (0 1 0)(0 1 1) }
estimate    { }
```

Example 12 Estimate a model for a monthly retail sales series with stable seasonal and trading day regressors. Include regressors for a change of regime in both sets of regressors in December of 2008. The transformed series, minus the regression effects, follows an ARIMA (0 1 1) model.

```
series { title = "Retail Sales - Televisions"
```

```
          start = 1996.1  period = 12  type = flow
          file  = 'tvsales.ori'  }
transform { function = log }
regression {  variables = (td/2008.dec/ seasonal/2008.dec/) }
arima { model = (0 1 1) }
estimate { }
```

Example 13 Similar to example 12, only partial change of regime regressors are used in conjunction with the seasonal and trading day regressors so that the extra regressors are set to zero before December of 2008.

```
series {title = "Retail Sales - Televisions"
        start = 1996.1  period = 12  type = flow
        file  = 'tvsales.ori'   }
transform  { function = log }
regression { variables = (td td//2008.dec/
                          seasonal seasonal//2008.dec/) }
arima {  model = (0 1 1) }
estimate { }
```

Example 14 Estimate a model with two AOs, and two LSs for a quarterly seasonal series. Accounting for these effects, the transformed series follows an ARIMA $(0\ 1\ 1)(0\ 1\ 1)_4$ model.

```
Series     { Title  = "Quarterly Sales"  Start = 1993.1  Period = 4
             Data  = (1039 1241 ...  2210)  }
Transform  { Function = Log  }
Regression { Variables = (AO2001.3 LS2007.1 LS2007.3 AO2008.4) }
Arima      { Model = (0 1 1)(0 1 1) }
Estimate   { }
```

Example 15 Suppose the outliers included as regression variables were found via outlier detection performed in a previous run of the program. Suppose also that the t-test for cancellation of the two level shifts did not reject the null hypothesis of cancellation. The example below is the same as Example 14, except that we have replaced the two level shift outliers with a temporary level shift regressor that amounts to the second LS cancelling the first (see Section 7.11 for more details).

```
Series     { Title  = "Quarterly Sales"  Start = 1993.1  Period = 4
             Data  = (1039 1241 ...  2210)  }
Transform  { Function = Log  }
Regression { Variables = (AO2001.3 TL2007.1-2007.2 AO2008.4) }
Arima      { Model = (0 1 1)(0 1 1) }
Estimate   { }
```

Example 16 A variant of the last two examples uses a level shift sequence regressor, which inserts 3 level shift regressors for the span covering the first through third quarters of 2007.

```
Series     { Title  = "Quarterly Sales"  Start = 1993.1  Period = 4
             Data  = (1039 1241 ...  2210)  }
Transform  { Function = Log  }
Regression { Variables = (AO2001.3 LSS2007.1-2007.3 AO2008.4) }
Arima      { Model = (0 1 1)(0 1 1) }
Estimate   { }
```

Example 17 Specified regression variables are a trend constant and trading day effects. Use the automatic modeling procedure to select an ARIMA model. Additively seasonally adjust the series after preadjusting for the trading day regression effects.

```
series     {  title = "Exports of pasta products"
              start = 2000.jan  data = "pasta.dat" }
regression {  variables = (const td)               }
automdl    {                                       }
x11        {  mode = add                           }
```

Example 18 The regression effects selected are seasonal means, a constant, several outliers, trading day, and an Easter effect. There are user-defined regression variables for special sales promotions in 2008, 2009 and 2010, which are located in the file `promo.dat` in `3f12.0` format. The ARIMA part of the model is (2, 1, 0). Seasonally adjust the series after pre-adjusting for all the regression effects. Remove the Easter effects and trading day effects from the final seasonally adjusted series. Generate 24 forecasts.

```
series{ title = "Retail sales of children's apparel"
 file = "capprl.dat" start = 1995.1  }
transform{   function = log }
regression{
  variables = (const td ao1996.oct ls2011.dec easter[8]
  user = (sale2008 sale2009 sale2010)
  start = 1995.1   file = "promo.dat"   format = "(3f12.0)"
arima{   model = (2 1 0)   }
forecast{   maxlead = 24   }
x11{   save=seasonal  appendfcst=yes  }
```

Example 19 The same as Example 18, except that the user-defined regression effect will be handled the same way as additive outliers with regard to prior adjustments, final adjustments, print files, and save files.

```
series{ title = "Retail sales of children's apparel"
 file = "capprl.dat" start = 1995.1  }
transform{   function = log }
regression{
  variables = (const td ao1996.oct ls2011.dec easter[8]
               seasonal)
  user = (sale2008 sale2009 sale2010)
```

```
  start = 1995.1   file = "promo.dat"   format = "(3f12.0)"
  usertype = ao
}
arima{   model = (2 1 0)   }
forecast{   maxlead = 24   }
x11{   save=seasonal  appendfcst=yes   }
```

Example 20 Specify a regARIMA model with trading day and outlier terms. Specify starting values for the regression coefficients, and hold the coefficients of the outlier regressors fixed at these values. Use this model to generate 12 forecasts (by default, since an `x11` spec is present). Perform a default multiplicative seasonal adjustment, after prior adjustment for trading day and outlier factors.

```
series{
    format = "2L"
    title  = "Midwest Total Starts"
    file   = "mwtoths.dat"
    name   = "MWTOT "
}
transform{  function=log  }
arima{  model=(0 1 2 )(0 1 1 )  }
estimate{  save=mdl  }
regression{
    variables = (ao2007.jan ls2009.jan ls2009.mar ls2010.jan td)
    b = (  -0.7946F  -0.8739F  0.6773F  -0.6850F  0.0209
           ~0.0107   -0.0022   0.0018   ~0.0088  -0.0074  )
}
x11{  }
```

Example 21 Specified regression variables are a trend constant and trading day effects. As this is a non-seasonal series, generate a trend component after preadjusting for the trading day regression effects. Add the trading day adjusted original series as part of a composite adjustment.

```
series     {  title = "Meat, not seasonally adjusted, td adjusted"
              start = 2005.jan  data = "meat.dat"
              comptype = add                        }
regression {  variables = (const td)                }
arima      {  model = (0 1 1)                       }
x11        {  type = trend                          }
```

Example 22 Read in the data from a file using a predefined `X-11` data format. Note that the starting date is taken from the information provided in the data file, so it does not have to be specified. Specify a regARIMA model with trading day and holiday terms. Perform automatic outlier identification, and print out model diagnostics. Use this model to generate 12 forecasts. Perform a multiplicative seasonal adjustment, using a 3x3 seasonal moving average, after prior adjustment for trading day, outlier and holiday factors. Remove the holiday and trading

day factors from the final seasonally adjusted series. Save the trading day and holiday factors in individual output files.

```
Series {
  Format="1L"   File="bdptrs.dat"   Name="BDPTRS"
  Title="Department Store Sales"  }
Transform {  Function=Log  }
Regression {  Variables=( Td Easter[8] )
              Save = ( Td Holiday )  }
Arima {   Model=(0 1 1)(0 1 1)  }
Outlier {     }
Estimate {     }
Check {    }
Forecast {  }
X11 {
  Mode = Mult   Seasonalma = S3X3
  Title=( "Department Store Retail Sales Adjusted For"
          "Outlier, Trading Day, And Holiday Effects" )
}
```

Example 23 Same as previous example, except an `easter[0]` regressor is added to the `regression` spec.

```
Series {
  Format="1L"   File="bdptrs.dat"   Name="BDPTRS"
  Title="Department Store Sales"  }
Transform {  Function=Log  }
Regression {  Variables=( Td Easter[8] Easter[0] )
              Save = ( Td Holiday )  }
Arima {   Model=(0 1 1)(0 1 1)  }
Outlier {     }
Estimate {     }
Check {    }
Forecast {  }
X11 {
  Mode = Mult   Seasonalma = S3X3
  Title=( "Department Store Retail Sales Adjusted For"
          "Outlier, Trading Day, And Holiday Effects" )
}
```

Example 24 Same as previous example, except an `aictest` argument is added to the `regression` spec. Also, a `testalleaster` argument is added to ensure a model with both `easter[0]` and `easter[8]` regressors is included in the AIC-based testing.

```
Series {
  Format="1L"   File="bdptrs.dat"   Name="BDPTRS"
  Title="Department Store Sales"  }
```

```
Transform {  Function=Log  }
Regression {  Variables=( Td Easter[8] Easter[0] )
              Save = ( Td Holiday )
              aictest = (td easter)
              testalleaster = yes  }
Arima {   Model=(0 1 1)(0 1 1)  }
Outlier {     }
Estimate {     }
Check {    }
Forecast {  }
X11 {
  Mode = Mult   Seasonalma = S3X3
  Title=( "Department Store Retail Sales Adjusted For"
          "Outlier, Trading Day, And Holiday Effects" )
}
```

Example 25 This spec file reads in a set of seasonal regressors saved from a previous `X-13ARIMA-SEATS` run. The series adjusted for regression effects (including the user-defined seasonal effect) is saved.

```
series{ title = "US Total Housing Starts"
 file = "ustoths.dat" start = 1990.1
 period = 4  save = b1}
transform{   function = log }
regression{
  user = (s1 s2 s3)
  usertype = seasonal
  start = 1985.1   file = "seasreg.rmx"
  format = "x13save"
}
outlier{   }
arima{   model = (0 1 1)   }
forecast{   maxlead = 24   }
```

Example 26 This example shows how to specify a groups of user-defined holiday regressors for payments made to child care workers in Taiwan. Holiday regressors are specified for Chinese New Year, the Moon Festival, and the Mid Fall Festival. The `chi2test` option is used to determine which of the user-defined holiday regressors are significant.

```
series{
    file="serv.dat"  start=1991.jan  span=(1993.jan,)
    title = "Payment to family nanny, taiwan"
}
transform{    function=log   }
regression{
   variables = (  AO1995.Sep AO1997.Jan AO1997.Feb  )
```

```
   user=(    Beforecny      Betweencny      Aftercny
    Beforemoon     Betweenmoon     Aftermoon
    Beforemidfall  Betweenmidfall  Aftermidfall  )
   file="u1u2u3.dat"
   format="datevalue"
   start=1991.1
   usertype=(   holiday     holiday     holiday
    holiday2    holiday2    holiday2
    holiday3    holiday3    holiday3  )
   chi2test = yes
   savelog = chi2test
}
arima{  model=(0 1 1)(0 1 0)   }
check{  }
forecast{  maxlead=12  }
estimate{  savelog=(aic aicc bic)  }
```

7.14 SEATS

DESCRIPTION

An optional spec invoking the production of model-based signal extraction using SEATS, a seasonal adjustment program developed by Victor Gómez and Agustin Maravall at the Bank of Spain.

The user can set options that control ARIMA model estimation if done within the SEATS module (**epsiv** and **maxit**) and perform checks on the model submitted to the SEATS modules (**qmax**, **rmod**, and **xl**). The user can also choose options to decompose the trend-cycle into a long-term trend and a cycle component .

USAGE

```
seats {   appendfcst = yes
          finite = yes
          hpcycle = yes
          hplan = 1000
          hprmls = no
          hptarget = orig
          noadmiss = yes
          qmax = 20
          rmod = 0.85
          statseas = yes
          out = 2
          print = (s10 s11 s12 s1s s2s)
          printphtrf = 1
          save = (s10 s11)
          savelog = (normalitytest seatsmodel)
          tabtables = "xo,n,s,p"
      }
```

ARGUMENTS

appendfcst Determines if forecasts will be included in certain SEATS tables selected for storage with the **save** argument. If `appendfcst = yes`, then forecasted values will be stored with table `s10`. If `appendfcst = no`, no forecasts will be stored. The default is to not include forecasts.

finite The default (`finite = no`) produces filter and diagnostic output that are obtained from infinite (Wiener-Kolmogorov) filters, and signal extraction error and revisions statistics are associated with semi-infinite or bi-infinite data. With `finite = yes`, all of the filter output and most of the signal extraction error and revisions statistics are finite-sample quantities for the available data.

hpcycle If `hpcycle = yes`, then the program will decompose the trend-cycle into a long-term trend and a cycle component using the modified Hodrick-Prescott filter. If `hpcycle = no`, the program will not perform this decomposition. The default is to perform this decomposition (`hpcycle = yes`). For more information on the Hodrick-Prescott filter, see Kaiser and Maravall (2001), Wikipedia (2020), and McElroy (2008a).

hplan A parameter that is used to determine the modified Hodrick-Prescott filter. By default, the program will set this parameter automatically according to the seasonal period of the series.

hprmls If `hprmls = yes`, then the program will remove level shift or ramp outliers from the series the Hodrick-Prescott filter is applied to. If `hprmls = no`, the program will not perform this preadjustment. The default is to not adjust for level shifts (`hprmls = no`).

hptarget Allows the user to specify the target of the Hodrick-Prescott filter. If `hptarget = sadj`, the Hodrick-Prescott filter is applied to the final seasonally adjusted series. If `hptarget = trend`, the Hodrick-Prescott filter is applied to the final trend component. If `hptarget = orig`, the Hodrick-Prescott filter is applied to the original series. The default is to apply the Hodrick-Prescott filter to the final trend (`hptarget = trend`).

noadmiss When `noadmiss = yes`, if the model submitted to SEATS does not lead to an admissible decomposition, it will be replaced with a decomposable model. Otherwise when `noadmiss = no`, no approximation is done in this case. The default is `noadmiss = no`.

out Sets level of seasonal decomposition diagnostic output. The default (`out = 0`) produces the most complete output, while `out = 1` and `out = 2` produce more abbreviated output (with `out = 2` the most abbreviated).

If tables are specified in the **print** argument, **out** is set to 2; otherwise, the default is `out = 0`. Note that many tables are not available for saving when **out** is set to 2; it is recommended to set **out** to be 0 or 1 if you wish to save files from the **seats** spec.

print and **save** Table 7.31 gives the available tables that can be both printed out and saved for this spec. Choices here override the selection made with the out argument – if one or more of these tables is selected, no other SEATS output will be produced.

Note that in the descriptions for the tables named `diffseasonaladj` and `difftrend` given in Table 7.31, the term "fully differenced" means differenced to the order of the sum $d + D$ of the nonseasonal and seasonal differencing orders of the ARIMA model.

Table 7.32 gives a listing of tables that can only be saved by the program. Specifying one of the these tables in the **print** argument will have no effect on the printout – they should only be used with the **save** argument.

Note that many of the series specified in Table 7.32 are only produced in the finite filter calculations are used for the SEATS decomposition – you cannot save these tables if `finite = no` in the **seats** spec. Also, the component models cannot be saved when `out = 2`.

Table 7.33 gives table names and abbreviations that can be used with the **save** argument to save certain tables as percentages rather than ratios. Specifying these table names in the **print** argument will not change the output of the program, and the percentages are only produced when a log transformation is specified in the **transform** spec.

name	*short*	*description of table*
trend	s12	final SEATS trend component
seasonal	s10	final SEATS seasonal component
irregular	s13	final SEATS irregular component
seasonaladj	s11	final SEATS seasonal adjustment
transitory	s14	final SEATS transitory component
adjustfac	s16	final SEATS combined adjustment factors
adjustmentratio	s18	final SEATS adjustment ratio
trendfcstdecomp	tfd	forecast of the trend component
seasonalfcstdecomp	sfd	forecast of the seasonal component
seriesfcstdecomp	ofd	forecast of the series component
seasonaladjfcstdecomp	afd	forecast of the final SEATS seasonal adjustment
transitoryfcstdecomp	yfd	forecast of the transitory component
seasadjconst	sec	final SEATS seasonal adjustment with constant term included
trendconst	stc	final SEATS trend component with constant term included
totaladjustment	sta	total adjustment factors for SEATS seasonal adjustment
difforiginal	dor	fully differenced transformed original series
diffseasonaladj	dsa	fully differenced transformed SEATS seasonal adjustment
difftrend	dtr	fully differenced transformed SEATS trend
seasonalsum	ssm	seasonal-period-length sums of final SEATS seasonal component
cycle	cyc	cycle component
longtermtrend	ltt	long term trend
seasonaladjoutlieradj	se2	final SEATS seasonal adjustment, outlier adjusted
irregularoutlieradj	se3	final SEATS irregular component, outlier adjusted

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.

Table 7.31: **Available Output Tables in Both `print` and `save` Arguments for Seats**

printphtrf When `printphtrf = 1`, the program will produce output related to the transfer function and phase delay of the seasonal adjustment filter. Otherwise when `printphtrf = 0`, no such output is produced. The default is `printphtrf=0`.

qmax Sets a limit for the Ljung-Box Q statistic, which is used to determine if the model provided to the SEATS module is of acceptable quality. The default is `qmax = 50`.

When model coefficients are fixed in the **arima** or **regression** specs, it is often necessary to choose a larger value of **qmax** to keep SEATS from changing the model.

name	*short*	*description of table*
componentmodels	mdc	models for the components
filtersaconc	fac	concurrent finite seasonal adjustment filter
filtersasym	faf	symmetric finite seasonal adjustment filter
filtertrendconc	ftc	concurrent finite trend filter
filtertrendsym	ftf	symmetric finite trend filter
pseudoinnovtrend	pic	pseudo-innovations of the trend component
pseudoinnovseasonal	pis	pseudo-innovations of the seasonal component
psuedoinnovtransitory	pit	pseudo-innovations of the transitory component
psuedoinnovsadj	pia	pseudo-innovations of the final SEATS seasonal adjustment
squaredgainsaconc	gac	squared gain for finite concurrent seasonal adjustment filter
squaredgainsasym	gaf	squared gain for finite symmetric seasonal adjustment filter
squaredgaintrendconc	gtc	squared gain for finite concurrent trend filter
squaredgaintrendsym	gtf	squared gain for finite symmetric trend filter
timeshiftsaconc	tac	time shift for finite concurrent seasonal adjustment filter
timeshifttrendconc	ttc	time shift for finite concurrent trend filter
wkendfilter	wkf	end filters of the semi-infinite Wiener-Kolmogorov filter

Name gives the name of each table for use only with the **save** argument.
Short gives a short name for these tables.

Table 7.32: **Output Tables Available Only with `save` Argument for Seats**

name	*short*	*description of table*
seasonalpct	pss	final seasonal factors, expressed as percentages if appropriate
irregularpct	psi	final irregular component, expressed as percentages if appropriate
transitorypct	psc	final transitory component, expressed as percentages if appropriate
adjustfacpct	psa	combined adjustment factors, expressed as percentages if appropriate

Name gives the name of each table for use with the **save** argument.
Short gives a short name for these.

Table 7.33: **Tables Saved As Percentages in the `save` Argument for Seats**

savelog The diagnostics available for output to the log file (see section 2.6) are listed in Table 7.34.

name	*short*	*description of diagnostic*
seatsmodel	smd	model used by the SEATS module for signal extraction
x13model	xmd	model submitted to the SEATS module
normalitytest	nrm	normality test
overunderestimation	oue	over-under estimation diagnostics
totalsquarederror	tse	total mean squared error
componentvariance	cvr	component variances
concurrentesterror	cee	concurrent estimation error
percentreductionse	prs	percent reduction standard error
averageabsdiffannual	aad	annual Average absolute difference
seasonalsignif	ssg	test for seasonal significance
durbinwatson	dws	Durbin-Watson statistic for model residuals from SEATS output
friedman	frs	Friedman non-parametric test for residual seasonality from SEATS output

Name gives the name of each diagnostic for use with the **savelog** argument.
Short gives a short name for these diagnostics.

Table 7.34: **Available Log File Diagnostics for Seats**

statseas If `statseas = no`, the program will not accept a stationary seasonal model, and will change the seasonal part of the model to (0 1 1). If `statseas = yes`, the program will accept a stationary seasonal model. The default is `statseas = yes`.

tabtables A list of seasonal adjustment components and series to be stored in a separate file with the extension `_tbs.html`. The list is entered as a text string with codes listed in Table 7.35; individual entries can be separated by commas (`tabtables = "xo,n,s,p"`) or spaces (`tabtables = "xo n s p"`). Note that components can only be added – they cannot be removed as in the **print** argument. The default is `tabtables = "all"`.

RARELY USED ARGUMENTS

bias Corrects for the bias that may occur in multiplicative decomposition when the period-to-period changes are relatively large when compared to the overall mean. This argument should only be set when a log transformation is used.

If `bias = 1`, a correction is made for the overall bias for the full length of the series and for the forecasting period. This is the default value.

If `bias = -1`, a correction is made so that, for every year (including the forecasting period), the annual average of the original series equals the annual average of the seasonally adjusted series, and also (very approximately) equals the annual average of the trend.

If `bias = 0`, no bias correction is done. No other values are allowed.

code	*description of table*
all	all series
xo	original series
n	seasonally adjusted series
s	seasonal factors
p	trend-cycle
u	irregular
c	transitory
cal	calendar
pa	preadjustment factor
cy	cycle
ltp	long term trend
er	residuals
rg0	separate regression component
rgsa	regression component in seasonally adjusted series
stp	stochastic trend cycle
stn	stochastic seasonally adjusted series
rtp	real time trend cycle
rtsa	real time seasonally adjusted series

Code gives the code used to specify the series in the **tabtables** argument.

Table 7.35: **Components Savable in `_tbs.html` file**

epsiv Convergence criteria for ARIMA estimation within the SEATS module; this is used when the SEATS module determines that a model should be changed or re-estimated. This should be a small positive number; the default is 0.001.

epsphi When Phi(B) contains a complex root, it is allocated to the seasonal if its frequency differs from the seasonal frequencies by less than epsphi degrees. Otherwise, it goes to the cycle. The default is 2.

imean Indicates if the series is to be mean-corrected (`imean = yes`). The default is not to remove the mean from the series before signal extraction (`imean = no`).

maxit Number of iterations allowed for ARIMA estimation within the SEATS module; should be a positive integer. Default is 20.

rmod Limit for the modulus of an AR root. If the modulus of an AR root is larger than **rmod**, the root is assigned to the trend; if the modulus of an AR root is smaller than **rmod**, the root is assigned to the cycle. The default value of **rmod** is 0.80.

xl When the modulus of an estimated root falls in the range $(XL, 1)$, it is set to 1.00 if the root is in the AR polynomial. If the root is in the MA polynomial, it is set to **xl**. The default is 0.99.

SEATS file name	`X-13ARIMA-SEATS` *extension*	*Contents of file*
`rogtable.out`	*filename*`_rog.html`	selected statistics from the growth rate output
`summarys.txt`	*filename*`_sum.html`	summary information and diagnostics from SEATS adjustment
`table-s.out`	*filename*`_tbs.html`	annotated listing of the series, the seasonally adjusted series, and components of the model-based seasonal adjustment, saved in table format

SEATS file name gives the file name saved by the SEATS program.
`X-13ARIMA-SEATS` *extension* gives the file extension used to save the output from the corresponding SEATS output file.

Table 7.36: `X-13ARIMA-SEATS` **File Extensions for Special SEATS Saved Output**

DETAILS

A tutorial document Findley, Lytras, and Maravall (2016) is available for the ARIMA model-based seasonal adjustment method used by SEATS. For more details on the SEATS seasonal adjustment method and diagnostics, see Maravall (1995), Gómez and Maravall (1996) and Gómez and Maravall (2001b); for comparisons of SEATS and `X-11` seasonal adjustments and filters, see Hood, Ashley, and Findley (2000), Hood (2002b), Findley, Wills, Aston, Feldpausch, and Hood (2003), and Findley and Martin (2006).

Note that there are other output files that were saved by the SEATS program that are available when running the `X-13ARIMA-SEATS` program. These output files can contain forecasts, components or diagnostics generated from the SEATS model-based adjustment performed. Table 7.36 shows the file extensions that are used to save the corresponding special output file from SEATS, similar to how the short table names are used as file extensions in storing individual tables to separate files. These extensions do not have to be specified in the **save** argument – these files will be produced for every `X-13ARIMA-SEATS` run with a SEATS seasonal adjustment. Section 3.2 gives details on the naming conventions used for `X-13ARIMA-SEATS` saved output.

The matrix formulas of McElroy (2008b) for (nonstationary) ARIMA signal extraction substantially simplify those of Bell and Hillmer (1988). We will motivate them from re-expressions of the standard regression formulas for stationary (or more general) linear unobserved component decompositions of mean zero data $w_t, t = 1, \ldots, n$ into *uncorrelated* mean zero components

$$w_t = u_t + v_t.$$

Thus, with $u = (u_1, \ldots, u_n)'$, $v = (v_1, \ldots, v_n)'$, and $w = (w_1, \ldots, w_n)'$, we require $\Sigma_{uv} \equiv Euv' = 0_{n \times n}$, which yields the variance matrix decomposition

$$\Sigma_{ww} = \Sigma_{uu} + \Sigma_{vv}$$

and the covariance matrix formula

$$\Sigma_{uw} \equiv Eu\,(u + v)' = \Sigma_{uu}.$$

Hence, given Σ_{uu} and Σ_{vv}, the basic formula

$$\beta = \Sigma_{uw}\Sigma_{ww}^{-1}$$

for the coefficient matrix minimizing $E\left(u-\beta w\right)'\left(u-\beta w\right)$, which provides the mean square optimal linear estimate $\widehat{u}$ of u from w,

$$\widehat{u}=\beta w,$$

can be evaluated as

$$\begin{aligned}\beta &= \Sigma_{uu}\left(\Sigma_{uu}+\Sigma_{vv}\right)^{-1}\\ &= \left(\Sigma_{uu}^{-1}+\Sigma_{vv}^{-1}\right)^{-1}\Sigma_{vv}^{-1}.\end{aligned}\tag{7.12}$$

Similarly, the variance matrix of the error $e=u-\widehat{u}$ reduces to

$$\begin{aligned}\Sigma_{ee} &= \Sigma_{uu}-\Sigma_{uu}\Sigma_{ww}^{-1}\Sigma_{uu}\\ &= \left(\Sigma_{uu}^{-1}+\Sigma_{vv}^{-1}\right)^{-1}.\end{aligned}\tag{7.13}$$

The less familiar formulas (7.12) and (7.13) generalize to the ARIMA case.

Now consider ARIMA data $Y_1,\ldots,Y_n$ with differencing polynomial $\delta_Y\left(B\right)=1+\delta_1B^1+\cdots+\delta_dB^d$ $(d\geq 1)$ resulting in mean zero stationary $w_t=\delta_Y\left(B\right)Y_t$, $d+1\leq t\leq n$. We assume there is a *signal plus noise decomposition* of Y_t into difference-stationary components

$$Y_t=S_t+N_t$$

with differencing operators $\delta_S\left(B\right)$ and $\delta_N\left(B\right)$ of degrees d_S and d_N, respectively, with no common zeros and with $\delta_Y\left(B\right)=\delta_S\left(B\right)\delta_N\left(B\right)$. This gives us mean zero processes

$$u_t=\delta_S\left(B\right)S_t,\ v_t=\delta_N\left(B\right)N_t$$

that are *uncorrelated*, $Eu_tv_{t+h}=0$, for all t, h. For example, if $\delta_Y\left(B\right)=\left(1-B\right)\left(1-B^{12}\right)$ and S_t is the seasonal component for monthly data, then

$$\delta_S\left(B\right)=1+B+\cdots+B^{11},\ \delta_N\left(B\right)=\left(1-B\right)^2.$$

Stationary case formulas do not apply to estimate nonstationary S_t: variance matrices of ARIMA data cannot be estimated consistently. Consider the simplest difference stationary model, the *random walk*:

$$z_t=z_{t-1}+a_t\ \text{ or }\ (1-B)z_t=z_t-z_{t-1}=a_t,\tag{7.14}$$

with uncorrelated, zero-mean a_t with $Ea_t^2=\sigma_a^2$ for $t=2,\ldots,n$. The variance matrix of $a=\left(a_2,\ldots,a_n\right)'$, being $\Sigma_a=\sigma_a^2I_{n-1}$, can be consistently estimated from the available z_t-z_{t-1}. The variance matrix of $z_1,\ldots,z_n$ cannot: for $t\geq 2$, we have

$$z_t = z_{t-1} + a_t = z_{t-2} + a_{t-1} + a_t = \cdots = z_1 + \sum_{j=2}^{t} a_j.$$

Assuming z_1 is uncorrelated with all a_t (Assumption A of Bell, 1984), then for any $k \geq 0$,

$$\begin{aligned} Ez_t z_{t+k} &= Ez_t^2 + Ez_t \left(\sum_{j=t+1}^{t+k} a_j \right) \\ &= Ez_t^2 + 0 = Ez_1^2 + (t-1)\,\sigma_a^2. \end{aligned}$$

Clearly Ez_1^2 cannot be estimated consistently from one datum z_1.

In the ARIMA signal extraction case, assuming that $Y_1, \ldots, Y_d$ are uncorrelated with all w_t, McElroy (2008b) shows that the mean square optimal linear estimate of $S_t, 1 \leq t \leq n$ is

$$\widehat{S} = \beta Y \tag{7.15}$$

with

$$\beta = \left(\Delta_S' \Sigma_{uu}^{-1} \Delta_S + \Delta_N' \Sigma_{vv}^{-1} \Delta_N \right)^{-1} \Delta_N' \Sigma_{vv}^{-1} \Delta_N. \tag{7.16}$$

Here Δ_N in (7.16) implements the calculation of $\delta_N(B)\, Y_t$, $d_{S+1} \leq t \leq n$, and Σ_{vv} is the variance matrix of $v_{d_N+1}, \ldots, v_n$. Similarly, Δ_S implements $\delta_S(B)$. The variance matrix Σ_{ee} of the signal extraction error $e = S - \widehat{S}$ is given by

$$\Sigma_{ee} = \left(\Delta_S' \Sigma_{uu}^{-1} \Delta_S + \Delta_N' \Sigma_{vv}^{-1} \Delta_N \right)^{-1}. \tag{7.17}$$

SEATS, and its implementation in `X-13ARIMA-SEATS`, use the procedure of Hillmer and Tiao (1979) to derive ARIMA models for S_t and N_t from the ARIMA model for Y_t (assuming this ARIMA model has an "admissible" decomposition). Here S_t can denote any of the seasonal decomposition components (seasonal, trend, irregular, seasonally adjusted series, etc.). From the ARIMA models for S_t and N_t, the matrices Σ_{uu} and Σ_{vv} can be obtained, and therefore also the matrix of filters β for producing the component estimates $\widehat{S}_t, 1 \leq t \leq n$, as well as Σ_{ee}. From Σ_{ee}, standard errors and confidence intervals for $\widehat{S}_t$ can be obtained (which do not account for modeling error). When the log-transformation is used for modeling, $\widehat{S}_t$ and associated confidence intervals are exponentiated to obtain the estimates and confidence intervals for the observed data's seasonal decomposition components.

The program does not use the matrix formulas (7.15)–(7.16) to calculate the component estimates $\widehat{S}_t, 1 \leq t \leq n$. Instead, the original method of SEATS is used, which does not involve the time-consuming inversion of large matrices for long series. This "Wiener-Kolmogorov" method produces identical component estimates, but only bi-infinite-sample approximations to Σ_{ee} of (7.17) and to the associated standard errors and confidence

intervals, and similarly for the filter diagnostics. Setting `finite = yes` in the `seats` spec of `X-13ARIMA-SEATS` causes the matrix-based (finite-sample) versions of almost all diagnostics to be produced.

The finite-sample filter diagnostics (squared gain and time-shift functions) are illustrated and compared with infinite-filter diagnostics in Findley and Martin (2006). A derivation, analysis and comparison of one of the finite-sample over-/underestimation tests is given in Findley, McElroy, and Wills (2005). The general derivation of the finite-sample versions of these tests and their asymptotic distributions is given in McElroy (2008c). Finite-sample versions of other diagnostics have also been implemented, see McElroy and Gagnon (2008).

The tests are goodness-of-fit tests for the time series model chosen for the series; each of the tests evaluates the statistical properties of the models obtained for the seasonal factors, the seasonally adjusted series, the trend, and the irregular, as well as the properties these models predict for the variances and certain covariances of the estimates of these components. When the differencing operator for the ARIMA model for the series (usually in its log-transformed form) is $(1-B)^d(1-B^s)^D = (1-B)^{d+D}\left(1+B+\cdots+B^{s-1}\right)^D$ for $s = 4$ or 12, the basic component model assumptions are (i) that application of $\left(1+B+\cdots+B^{s-1}\right)^D$ to the seasonal component produces a stationary series whose ARMA model is known, and (ii) that application of $(1-B)^{d+D}$ to the seasonally adjusted series and trend, which yields what we call the *fully differenced* seasonally adjusted series and trend, does likewise for these components. Often, statistically significant values of the test statistics arise because application of $\left(1+B+\cdots+B^{s-1}\right)^D$ or $(1-B)^{d+D}$ to these components yields a series that is not stationary over the whole time interval of the observed series. This nonstationarity can often be detected in graphs of the outputs of these differencing operators applied to the estimated components. These outputs are available as the `seasonalsum` (`ssm`), `diffseasonaladj` (`dsa`), and `difftrend` (`dtr`) tables listed in Table 7.31, and the graphs can be obtained from `X-13-Graph` (see Hood 2002a, Hood 2002c, Lytras 2020a, and Lytras 2020b). Examination of the graphs when there is nonstationarity will frequently reveal shorter data intervals over which acceptable goodness-of-fit results can be obtained.

We recommend that series for which a stationary seasonal model is chosen (i.e., a model with seasonal AR or MA coefficients but no seasonal differencing) should not be seasonally adjusted. Adjustments of such series are susceptible to large revisions and are conceptually problematic, because seasonal factors of a given calendar month quickly change from indicating an increase to indicating a decrease, or vice versa. Thus, the repetitive quality inherent in the concept of seasonality is lacking. Changing the seasonal part of the model to (011), as `statseas = no` does, rarely produces more stable results and often imparts seasonality to the seasonally adjusted series.

The Hodrick-Prescott filter is applied to an estimated trend-cycle component, like that produced with the `X-13ARIMA-SEATS` seasonal-trend-irregular decomposition, with the goal of suppressing short-term economic cycle components. Its output is an estimate of the long-term trend. For more information on the Hodrick-Prescott filter, see Kaiser and Maravall (2001), Wikipedia (2020), and McElroy (2008a).

If the `hplan` argument is specified, the Hodrick-Precott filter is used, even if the user has set `hpcycle = no`. If `hpcycle = yes` and `hplan` is not set, the Hodrick-Prescott filter is only used when the series is long enough (120 observations for monthly series, 48 observations for quarterly series).

EXAMPLES

Example 1 A SEATS seasonal adjustment will be generated from the model determined by the automatic modeling procedure. The transformation will be selected by the automatic transfor-

mation selection procedure. Outlier identification will be performed for point, level shift, and temporary change outliers.

```
SERIES  { TITLE="EXPORTS OF TRUCK PARTS"
          START =1987.1
          FILE = "X21109.ORI"
          PERIOD = 12
}
TRANSFORM {  FUNCTION = AUTO  }
REGRESSION {  AICTEST = TD  }
AUTOMDL {   }
OUTLIER {  TYPES = (AO LS TC)  }
FORECAST {  MAXLEAD = 36  }
SEATS  { SAVE = S11  }
```

Example 2 A SEATS seasonal adjustment will be generated from the model specified by the user. Setting `finite = yes` in the **seats** spec will cause the finite sample output to be used, allowing the user to save finite sample filter diagnostics. A revision history of the seasonally adjusted series and the trend component will be performed, and the percent revisions of the seasonally adjusted series and the trend component will be saved in separate files.

```
Series  { Title="Quarterly Exports Of Mangos"
   Start =1990.1  File = "Xmango.Ori"  Period = 4 }
Transform {  Function = Log  }
Regression {  Aictest = Td   }
Arima {  Model = (0 1 1)(0 1 1)   }
Forecast {  Maxlead = 12  }
Seats {  Finite = yes
   Save = ( Squaredgainsaconc Timeshiftsaconc )
   Savelog = Overunderestimation
}
History { Estimates = (Sadj Trend)
   Save = ( Sarevisions Trendrevisions )
}
```

Example 3 A default SEATS seasonal adjustment will be generated from the model specified by the user for this bimonthly series. Outlier identification will be done on the entire series.

```
Series  { Title="Model based adjustment of Bimonthly exports"
   Start = 1995.1  File = "Xports6.Ori"  Period = 6 }
Transform {  Function = Log  }
Regression {  Variables = Td  }
Arima {  Model = (0 1 1)(0 1 1)   }
Outlier { types = (ao ls tc) }
Forecast {  Maxlead = 18  }
Seats {  save = (S11 S10 S12)  }
```

Example 4 Read in the data from a file using a predefined `X-11` data format. Note that the starting date will be taken from the information provided in the data file and so does not have to be specified. Specify a regARIMA model for the log transformed data with certain outlier terms. Use this model to generate 3 years of forecasts. Perform a SEATS model-based seasonal adjustment with a user-specified value for the parameter that generates the Hodrick-Prescott filter. Save the long term trend generated by this Hodrick-Prescott filter. The user specified value (125000) differs only slightly from the default value (133107), which is the recommended value for monthly series.

```
series { title = "NORTHEAST ONE FAMILY Housing Starts"
         file = "cne1hs.ori"   name="CNE1HS"   format="2R" }
transform {    function=log   }
regression {
    variables = (ao2006.feb ao2008.feb ls2010.feb
                 ls2012.nov ao2014.feb)
}
arima {  model = (0 1 2)(0 1 1)  }
forecast {  maxlead = 36  }
seats {  hplan = 125000   hpcycle = yes   save = ltt  }
```

7.15 SERIES

DESCRIPTION

Required spec that provides `X-13ARIMA-SEATS` with the time series data, a descriptive title for the series, the starting date of the series, the seasonal period (12 for monthly data, 4 for quarterly data,) and an optional restricted span (subset) within the time series to be used for the analysis. The data can either be included in the **series** spec by using the **data** argument, or they can be obtained from a file by using the **file** argument. Note that if `X-13ARIMA-SEATS` is run using a data metafile, the series should not be specified in this spec, since data files are specified in the data metafile (for more details, see Section 2.5).

USAGE

```
series {   title = "Example Series"
           start = 1997.1
           span = (2000.1,)
           modelspan = (2005.Jan, 0.Dec)
           name = "tstsrs"
           data = (480 ··· 1386)  or  file = "example.dat"
                                      format = "2r"
           decimals = 2
           precision = 1
           comptype = add
           compwt = 1.0
           print = (none +header)
           save = (spn)
           appendfcst = yes
           appendbcst = no
           type = stock
       }
```

ARGUMENTS

appendbcst Determines if backcasts will be included in certain tables selected for storage with the **save** option. If `appendbcst = yes`, then backcasted values will be stored with tables a16, b1, d10, and d16 of the **x11** spec, table s10 of the **seats** spec, tables a6, a7, a8, a8.tc, a9, and a10 of the **regression** spec, and tables c16 and c18 of the **x11regression** spec. If `appendbcst = no`, no backcasts will be stored. The default is to not include backcasts.

appendfcst Determines if forecasts will be included in certain tables selected for storage with the **save** option. If `appendfcst = yes`, then forecasted values will be stored with tables a16, b1, d9, d10, and d16 of the **x11** spec, tables a6, a7, a8, a8.tc, a9, and a10 of the **regression** spec, and tables c16 and c18 of the **x11regression** spec. If `appendfcst = no`, no forecasts will be stored. The default is to not include forecasts.

comptype Indicates how a component series of a composite (also called aggregate) series is incorporated into the composite. These component series can be *added into* the (partially formed) composite series (`comptype = add`), *subtracted from* the composite series (`comptype = sub`), *multiplied by* the composite series (`comptype = mult`), or *divided into* the composite series (`comptype = div`). The default is no aggregation (`comptype = none`).

Note that the composite series is initialized to zero, and each component is incorporated into the composite series sequentially. So when the desired composition is something like $comp = comp1 \times comp2$, the **comptype** argument for the first component should be set to `comptype = add`, so that the composite series is set to $0 + comp1$, and the **comptype** argument for the second component should be set to `comptype = mult`.

compwt Specifies that the series is to be multiplied by a constant before aggregation. This constant must be greater than zero (for example, `compwt = 0.5`). This argument can only be used in conjunction with **comptype**. The default composite weight is one.

data Vector containing the time series data. The data are read row-wise in the following format: there must be at least one blank space, comma, or carriage return separating each of the data values. The number of observations is automatically determined as the length of the data vector supplied. If the **data** argument is used, the **file** argument cannot be used.

decimals Specifies the number of decimals that will appear in the seasonal adjustment tables of the main output file. This value must be an integer between 0 and 5, inclusive (for example, `decimals = 5`). The default number of decimals is zero.

file Name of the file containing the time series data. The filename must be enclosed in quotes. If the file is not in the current directory, the complete filename including the path must be given. Valid path and filenames depend on the computer operating system. If the **file** argument is used, the **data** argument cannot be used.

format Denotes the format to be used in reading the time series data from the named file, when the data are not in free format. Several types of input can be used:

a. free format, in which all numbers on a line will be read before continuing to the next line, and the numbers must be separated by one or more spaces (not by commas or tabs) (example: `format = "free"`);

b. a valid Fortran format, which should be enclosed in quotes and must include the initial and terminal parentheses (example: `format = "(6f12.0)"`).

c. a two character code which corresponds to a set of data formats used in previous versions of `X-11` and `X-11-ARIMA` (example: `format = "1r"`);

d. "datevalue" format, where the year, month or quarter, and value of each observation are found in this order in free format on individual lines. Thus, a line of the data file containing the value 32531 for July 1991 would have the form `1991 7 32531`. The number of preceding blanks can vary (example: `format = "datevalue"`);

e. the format `X-13ARIMA-SEATS` uses to save a table. This allows the user to read in a file saved from a previous `X-13ARIMA-SEATS` run (example: `format = "x13save"`);[23]

[23]Note that to maintain compatibility with previous versions of `X-12-ARIMA` the entry **x12save** will also be accepted.

f. the format that the `TRAMO` and `SEATS` programs use to read in a series and its descriptors. This enables `X-13ARIMA-SEATS` to read in a data file formatted for the `TRAMO` modeling program or the `SEATS` seasonal adjustment program. (example: `format = "tramo"`);

g. a variant of "free" format where the numbers must be separated by one or more spaces (not by commas or tabs), and decimal points are expressed as commas (a convention in some European countries). (example: `format = "freecomma"`);

h. a variant of "datevalue" format, where the year, month or quarter, and value of each observation are found in this order in free format on individual lines, where decimal points are expressed as commas. Thus, a line of the data file containing the value 1355.34 for July 1991 would have the form `1991 7 1355,34`. The number of preceding blanks can vary (example: `format = "datevaluecomma"`).

In the predefined `X-11` data formats, the data is stored in 6 or 12 character fields, along with a year and series label associated with each year of data. For a complete list of these formats and how they are used, see DETAILS.

If no format argument is given, the data will be read in free format. The **format** argument cannot be used with **data**, only with **file**.

modelspan Specifies the span (data interval) of the data to be used to determine all regARIMA model coefficients. This argument can be utilized when, for example, the user does not want data early in the series to affect the forecasts, or, alternatively, data late in the series to affect regression estimates used for preadjustment before seasonal adjustment. As with the **span** argument, the **modelspan** argument has two values, the start and end date of the desired span. A missing value defaults to the corresponding start or end date of the span of the series being analyzed. For example, for monthly data, the statement `modelspan = (1968.1,)` causes whatever regARIMA model is specified in other specs to be estimated from the time series data starting in January 1968 and ending at the end date of the analysis span. A comma is necessary if either the start or end date is missing. The start and end dates of the model span must both lie within the time span of data specified for analysis in the **series** spec, and the start date must precede the end date.

Another end date specification, with the form *0.per*, is available to set the ending date of **modelspan** to always be the most recent occurrence of a specific calendar month (quarter for quarterly data) in the span of data analyzed, where *per* denotes the calendar month (quarter). Thus, if the span of data considered ends in a month other than December, `modelspan = (, 0.dec)` will cause the model parameters to stay fixed at the values obtained from data ending in the next-to-final calendar year of the span.

name The name of the time series. The name must be enclosed in quotes and may contain up to 64 characters. Up to the first 16 characters will be printed as a label on every page. When specified with the predefined formats of the **format** argument, the first six (or eight, if `format = "cs"`) characters of this name are also used to check if the program is reading the correct series or to find a particular series in a file where many series are stored.

period Seasonal period of the series. If `X-11` seasonal adjustments are generated, the only values currently accepted by the program are 12 for monthly series and 4 for quarterly series. If SEATS adjustments are generated, the values currently accepted by the program are 12 for monthly series, 6 for bimonthly series, 4 for quarterly series, 2 for biannual series. and 1 for annual series (primarily for trends). Otherwise, any seasonal period up to 12 can be specified. (This limit can be changed—see Section 2.8.) The default value for **period** is 12.

precision The number of decimal digits to be read from the time series. This option can only be used with the predefined formats of the **format** argument. This value must be an integer between 0 and 5, inclusive (for example, `precision = 5`). The default is zero. If **precision** is used in a **series** spec that does not use one of the predefined formats, the argument is ignored.

print and **save** Table 7.37 gives the available output tables for this spec. All these tables are included in the default printout, except `seriesplot` and `adjoriginalplot`. For a complete listing of the `brief` and `default` print levels for this spec, see Appendix B.

span Limits the data utilized for the calculations and analysis to a span (data interval) of the available time series. The **span** argument has two input values, the start and end date of the span. A missing value defaults to the corresponding start or end date of the input time series. For example, assuming monthly data, the statement `span = (1968.1,)` specifies a span starting in January 1968 and ending at the end date of the series input through the **data** or **file** argument. A comma is necessary if either the start or end date is missing. The start and end dates of the span must both lie within the series, and the start date must precede the end date.

start The start date of the time series in the format `start =` *year.seasonal period.* (See Section 3.3 and the examples below.) The default value of **start** is 1.1. (See DETAILS.)

title A title describing the time series. The title must be enclosed in quotes and may contain up to 79 characters. It will be printed on each page of the output (unless the **-p** option is invoked; see Section 2.7).

type Indicates the type of series being input. If `type = flow`, the series is assumed to be a flow series; if `type = stock`, the series is assumed to be a stock series. The default is to not assign a type to the series.

RARELY USED ARGUMENTS

divpower An integer value used to re-scale the input time series prior to analysis. The program divides the series by ten raised to the specified value. For example, setting `divpower = 2` will divide the original time series by 10^2, while `divpower = -4` will divide the series by 10^{-4}. Integers from -9 to 9 are acceptable values for **divpower**. If this option is not specified, the time series will not be re-scaled.

missingcode A numeric value in the input time series that the program will interpret as a missing value. This option can only be used in input specification files requiring a regARIMA

name	*short*	*save?*	*description of table*
header	hdr	·	summary of options selected for this run of `X-13ARIMA-SEATS`
span	a1	+	time series data, with associated dates (if the **span** argument is present, data are printed and/or saved only for the specified span)
seriesplot	a1p	·	plot of the original series
specfile	spc	·	contents of input specification file used for this run
savefile	sav	·	list of files to be produced by the `X-13-ARIMA-SEATS` run
seriesmvadj	mv	+	original series with missing values replaced by regARIMA estimates
calendaradjorig	a18	+	original series adjusted for regARIMA calendar effects
outlieradjorig	a19	+	original series adjusted for regARIMA outliers
adjoriginal	b1	+	original series, adjusted for prior effects and forecast extended
adjorigplot	b1p	·	plot of the prior adjusted original series augmented by prior-adjusted forecasts (if specified); if no prior factors or forecasts are used, the original series is plotted

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.37: **Available Output Tables for Series**

model to be estimated or identified automatically. The default value is −99999. Example: `missingcode = 0.0`.

missingval The initial replacement value for observations that have the value of **missingcode**. The subsequent replacement procedure is described in DETAILS. The default value of **missingval** is 1000000000. Example: `missingval = 10D10`.

saveprecision The number of decimals stored when saving a table to a separate file with the **save** argument. The default value of **saveprecision** is 15. Example: `saveprecision = 10`.

trimzero If `trimzero = no`, zeros at the beginning or end of a time series entered via the file argument are treated as series values. IF `trimzero = span`, leading and trailing zeros are ignored if they fall outside the span of data being analyzed (the **span** argument must be specified with both a starting date and an ending date). The default (`trimzero = yes`) causes leading and trailing zeros to be ignored. Note that when the **format** argument is set to either `free`, `datevalue`, `x13save`, or `tramo`, all values input are treated as series values, regardless of the value of **trimzero**.

DETAILS

The number of observations and the series end date are determined by the program after reading in the data. `X-13ARIMA-SEATS` accepts a maximum of 780 observations. (This limit can be changed—see Section 2.8.)

If spec files are copied from one directory to another or from one computer system to another, verify that the path and filenames in their **file** arguments remain valid.

The **series** spec cannot appear in a spec file with the **composite** spec. The latter signifies that a seasonal adjustment of a composite series is to be calculated.

Table 7.38 gives a description of the default formats for each of the valid two-character `X-11` format codes for the **format** argument, as well as the corresponding Fortran format. These formats can be modified by using the **precision** argument. If **precision** is used in a **series** spec that does not use an `X-11` format code, the argument is ignored.

Note that if one of the `X-11` format codes is specified (or if `format = "datevalue"`, `format = "datevaluecomma"`, `format = "tramo"`, or `format = "x13save"`), the start of the series is automatically read from the data file. Therefore, the starting date need not be specified with the **start** argument of the **series** spec.

If a data metafile is used to process a group of input files using a single input spec file, the `X-11` formats should be avoided. These formats require the name of the series (specified **name**) to verify that the data is in the file. This implies that all data files in the data metafile would be required to use the same series name. This is rarely desirable.

When doing a *formatted* read of a data *file*, `X-13ARIMA-SEATS` discards sequences of zeros at the ends of the series (unless `trimzero = no`). This convention is used to allow input of series stored in certain formats—Example 3 below gives an illustration. If the zeros at the ends of the series are true data values, `trimzero = no` will cause them to be treated as such. However, if the zeros at the beginning of a given series are real and the zeroes implied at the end of the series are not (due to blanks at the end of the line), then the file must be modified so that it can be read in free format. Example 4 below demonstrates this conversion.

Code	*Fortran Format for Monthly Data*	*Fortran Format for Quarterly Data*	*Description*
1r	(12f6.0,I2,A6)	(4(12x,f6.0),I2,A6)	year and identifier on the right, data in 6-digit fields
2r	(6f12.0,/,6f12.0,I2,A6)	(4f12.0,24x,I2,A6)	year and identifier on the right of the second line, data in 12-digit fields
1l	(A6,I2,12f6.0)	(A6,I2,4(12x,f6.0))	year and identifier on the left, data in 6-digit fields
2l	(A6,I2,6f12.0,/,8x,6f12.0)	(A6,I2,4f12.0)	year and identifier on the left of the first line, data in 12-digit fields
2l2	(A8,I4,6f11.0,2x,/,12x,6f11.0,2x)	(A8,I4,4f11.0,2x)	four digit year and identifier on the left of the first line, data in 11-digit fields
cs	(A8,I2,10X,12E16.10,18X)	(A8,I2,10X,12E16.10,18X)	data in CANSIM data base utility format, data in 16-digit fields
cs2	(A8,I4,12X,12E16.10,14X)	(A8,I4,12X,12E16.10,14X)	data in the new CANSIM data base utility format (called CANSIM2), data in 16-digit fields

Table 7.38: **Default Formats for Each `X-11` Format Code**

The **span** and **modelspan** arguments can be used with the **forecast** spec to generate *out-of-sample* forecast comparisons by excluding data at the end of the series. When either of these arguments is present, model estimation will use data only for the specified span. Forecasting then (by default) proceeds from the end of the span, producing comparisons of the withheld data with the forecasts. (See Example 4 of the **forecast** spec.)

Note that if the beginning date specified in the **modelspan** argument is not the same as the starting date in the **span** argument, backcasts cannot be generated by the program, regardless of the value of the **maxback** argument of the **forecast** spec.

When the program encounters a value equal to the value of **missingcode** in the original series, it inserts an additive outlier for that observation time into the set of regression variables of the model for the series and then replaces the missing value code with a value large enough to be considered an outlier during model estimation. After the regARIMA model is estimated, the program adjusts the original series using factors generated from these missing value outlier regressors. The adjusted values are estimates of the missing values.

If a series is designated as a stock or a flow series by using the **type** argument, then trading day and Easter regressors specified in **regression** spec need to agree with this type – one cannot specify stock trading day regressors for a flow series. If a series type is not specified, then any trading day or holiday regressor may be used with the series.

EXAMPLES

Note: The following examples do not show "complete" spec files in the sense that useful output is not produced unless additional specs (e.g., **x11** or **arima** and **estimate**) are also included.

Example 1 Specify a time series with the **data** argument.

```
series{
    title = "A Simple Example"
    start = 2007.jan    # period defaults to 12
     data = ( 480  467  514  505  534  546  539  541  551  537  584  854
              522  506  558  538  605  583  607  624  570  609  675  861
              .
              .
              .
             1684 1582 1512 1508 1574 2303 1425 1386) }
```

Example 2 Drop observations from both the beginning and end of a quarterly series that starts in 1964 and ends in 2017. The first six years of data are dropped to restrict the analysis to post-WWII data. The data held out for 2015–17 could be used to examine out-of-sample forecast performance.

```
series { data = (879 899 985 ...)   # There are 216 data values
         start = 1965.1             #       ending in 2017.4
        period = 4                # Quarterly series
          span = (1971.1, 2015.4)  }
```

Example 3 This example shows how the `X-13ARIMA-SEATS` program can read data from files stored in a format adopted from the `X-11-ARIMA` seasonal adjustment program. Here the data are available from July 1980 through February 1993 and are stored in the file `c:\data\sales1.dat` as follows:

```
                                  146.4 109.2 132.1 144.8 116.1 100.380SALES1
142.9 158.8 196.2 244.0 251.6 245.5 244.2 213.8 188.9 197.2 181.2 161.381SALES1
  .
  .
  .
148.8 177.2     0     0     0     0     0     0     0     0     0     093SALES1
```

The data are stored in `(12f6.1,i2,a6)` format, with the last eight columns in each line providing the year and series ID.

```
SERIES{ TITLE = "Monthly data in an X-11 format"
      PERIOD = 12
      FILE = "C:\DATA\SALES1.DAT"    # a DOS path and file
      PRECISION = 1
      FORMAT = "1r" }
```

Since Fortran formatted reads treat blanks as zeros, the input of the series obtains six zeros at the beginning. The input series also contains the ten zeros at the end. As noted in DETAILS, `X-13ARIMA-SEATS` discards the zeros read in from both the beginning and end of the series by default so that only the actual data are retained and assigned to the correct months (146.4 to July 1980, etc.). Also note that since the year is given on each line, the user does not have to enter a **start** argument.

Example 4 This example illustrates the rare case of a data file that must be modified for correct input to `X-13ARIMA-SEATS`. The original data file contains data for February 2000 through November 2015 stored in `(6f4.0,1x,i4)` format as follows.

```
        0  342 -256  491    0  0001
 -234  922 -111    2    0  199  0002
  .
  .
  .
  581 -987 -423   10    0       0022
```

This file cannot be read in free format because several of the data entries run together and because the file contains record counters (0001, 0002, ...) in columns 26–29. A free format read would treat the record counters as data. The file cannot be read with `(6f4.0)` format with a start date of February 2000 because `X-13ARIMA-SEATS` with the default `trimzero = yes` would incorrectly drop the zeros at the first and last observations and would then erroneously assign the value 342 to February 2000. Using `trimzero = no` would add extra zeros to the series, as the blank spaces at the beginning and end of the data set would be read as zero.

The solution is to reformat the data file so it can be read in free format. This requires removal of the record counters and separation of the data entries. The modified file, `example4.new`, is as follows:

```
        0  342 -256  491    0
 -234  922 -111    2    0  199
  .
  .
  .
  581 -987 -423   10    0
```

Then the following **series** spec will correctly read the data from the file `example4.new`.

```
series {title = "Data read correctly in with trimzero = no"
        start = 2000.2   period = 12
        file = "example4.new" }    # file is in current directory
```

Example 5 This example shows how the X-13ARIMA-SEATS program can read data in "date-value" format. The data are available from July 2000 through February 2013, and are stored in the file c:\data\sales1.edt as follows:

```
2000    7   14624
2000    8   10952
2000    9   13251
2000   10   14408
 .
 .
 .
2013    1  14838
2013    2  17762
```

Each data record contains the year, month, and value of a given observation of the time series.

```
SERIES{
  TITLE = "Monthly data in a datevalue format"
  PERIOD = 12
  FILE = "C:\DATA\SALES1.EDT"    # a DOS path and file
  FORMAT = "DATEVALUE"
  TYPE = FLOW
}
```

Note that as in the X-11-ARIMA format shown in Example 3 above, the starting date can be read directly from the input file, so the user does not have to include a **start** argument. Also, the **type** argument is used to specify that this is a flow series.

Example 6 The same as example 5, but this series will be used as a component in a composite adjustment. The number of decimals displayed in the output is set to be 2, and the span of data to be modeled will be set to be the start of the series through December 2012.

```
SERIES{
  TITLE = "Monthly data in a datevalue format"
  PERIOD = 12
  FILE = "C:\DATA\SALES1.EDT"   # a DOS path and file
  FORMAT = "DATEVALUE"
  TYPE = FLOW
  COMPTYPE = ADD
  DECIMALS = 2
  MODELSPAN = (,2012.DEC)
}
```

Example 7 This example shows how the X-13ARIMA-SEATS program handles missing data. The same data format is used as in the previous two examples, except a missing value code is inserted for January of 2010:

```
2000    7   14624
2000    8   10952
2000    9   13251
2000   10   14408
 .
 .
 .
2010    1  -99999.
 .
 .
 .
2013    1  14838
2013    2  17762
```

The textbfseries spec below will replace the missing value code for January 1990 with a number large enough to be considered an outlier, assuming a regARIMA model is estimated later in the input specification file.

```
SERIES{ TITLE = "Monthly data in a date-value format"
      PERIOD = 12
      FILE = "C:\DATA\SALES1.EDT"    # a DOS path and file
      FORMAT = "DATEVALUE"
}
```

Example 8 This example shows how the X-13ARIMA-SEATS program can read data from a file previously saved by X-13ARIMA-SEATS (in a previous run, the outlier adjusted original series was stored in the file c:\data\sales1.a11).

```
SERIES{ TITLE = "Monthly data in a file saved by \thisprogram\ "
      PERIOD = 12
      FILE = "C:\DATA\SALES1.A11"    # a DOS path and file
      FORMAT = "X13SAVE" }
```

Note that as in the X-11-ARIMA format shown in Example 3 and the "datevalue" format shown in Example 5 above, the starting date can be read directly from the input file, so a **start** argument is not included.

Example 9 This example shows how the `X-13ARIMA-SEATS` program can read data in the special "date-value" format that uses the convention of commas as decimal points. As in Example 5, the data are available from July 1990 through February 2013, and are stored in the file `c:\data\sales1c.edt` as follows:

```
1990    7   146,24
1990    8   109,52
1990    9   132,51
1990   10   144,08
 .
 .
 .
2013    1   148,38
2013    2   177,62
```

Each data record contains the year, month and value of a given observation of the time series.

```
SERIES{  TITLE = "Monthly data in the comma variant of datevalue format"
         PERIOD = 12
         FILE = "C:\DATA\SALES1C.EDT"    # a DOS path and file
         FORMAT = "DATEVALUECOMMA"  }
```

7.16 SLIDINGSPANS

DESCRIPTION

Optional spec providing sliding spans stability analysis. This compares different features of seasonal adjustment output from overlapping subspans of the time series data. The user can specify options to control the starting date for sliding spans comparisons (**start**), the length of the sliding spans (**length**), the threshold values determining sliding spans statistics (**cutseas**, **cuttd**, **cutchng**), how the values of the regARIMA model parameter estimates will be obtained during the sliding spans seasonal adjustment runs (**fixmdl**), and whether regARIMA automatic outlier identification is performed (**outlier**).

USAGE

```
slidingspans {   start = 1995.jan
                 length = 132
                 numspans = 3
                 cutchng = 3.0
                 cutseas = 3.0
                 cuttd = 2.0
                 outlier = yes
                 fixmdl = no
                 fixreg = outlier
                 print = (long -ssheader)
                 save = (sfspans)
                 savelog = (percent)
             }
```

ARGUMENTS

cutchng Threshold value for the month-to-month, quarter-to-quarter, or year-to-year percent changes in seasonally adjusted series. For a month (quarter) common to more than one span, if the maximum absolute difference of its period-to-period percent changes from the different spans exceeds the threshold value, then the month (quarter) is flagged as having an unreliable estimate for this period-to-period change. This value must be greater than 0; the default value is 3.0. Example: `cutchng = 5.0`

cutseas Threshold value for the seasonal factors and seasonally adjusted series. For a month (quarter) common to more than one span, if the maximum absolute percent change of its estimated seasonal factors or adjustments from the different spans exceeds the threshold value, then this month's (quarter's) seasonal factor or adjustment is flagged as unreliable. This value must be greater than 0; the default value is 3.0. Example: `cutseas = 5.0`

cuttd Threshold value for the trading day factors. For a month (quarter) common to more than one span, if the maximum absolute percent change of its estimated trading day factors from the different spans exceeds the threshold value, then this month's (quarter's) trading

day factor is flagged as unreliable. This value must be greater than 0; the default value is 2.0. Example: `cuttd = 1.0`

fixmdl Specifies how the initial values for parameters estimated in regARIMA models are to be reset before seasonally adjusting a sliding span. This argument is ignored if a regARIMA model is not fit to the series.

If `fixmdl = yes`, the values for the regARIMA model parameters for each span will be set to the parameter estimates taken from the original regARIMA model estimation. These parameters will be taken as fixed and not re-estimated. This is the default for **fixmdl**.

If `fixmdl = no`, the program will restore the initial values to what they were when the regARIMA model estimation was done for the complete series. If they were fixed in the **estimate** spec, they remain fixed at the same values.

If `fixmdl = clear`, initial values for each span will be set to be the defaults, namely 0.1 for all coefficients, and all model parameters will be re-estimated.

fixreg Specifies the fixing of the coefficients of a regressor group in either a regARIMA model or an irregular component regression. These coefficients will be fixed at the values obtained from the model span (implicit or explicitly) indicated in the **series** or **composite** spec. All other regression coefficients will be re-estimated for each sliding span. Trading day (**td**), holiday (**holiday**), outlier (**outlier**), or other user-defined (**user**) regression effects can be fixed. This argument is ignored if neither a regARIMA model nor an irregular component regression is fit to the series, or if `fixmdl = yes`.

length The length of each span, in months or quarters (in accordance with the sampling interval) of time series data used to generate output for comparisons. A length selected by the user must yield a span greater than 3 years long and less than or equal to 19 years long. If the length of the span is not specified by the user, the program will choose a span length based on the length of the seasonal filter selected by the user (or by the program if a seasonal filter was not specified by the user) when the seasonal adjustment is performed by the **x11** spec, or by the level of the seasonal MA parameter coefficient (Θ), when the seasonal adjustment is performed by the **seats** spec. For more information, see DETAILS. Monthly data example: `length = 96`

numspans Number of sliding spans used to generate output for comparisons. The number of spans selected by the user must be between 2 and 4, inclusive. If this argument is not specified by the user, the program will choose the maximum number of spans (up to 4) that can be formed based on the length of the sliding spans given by the user (or selected by the program if the **length** argument is not used). Example: `numspans = 4`

outlier Specifies whether automatic outlier detection is to be performed whenever the regARIMA model is re-estimated during the processing of each span. This argument has no effect if the **outlier** spec is not used.

If `outlier = keep`, the program carries over any outliers automatically identified in the original estimation of the regARIMA model for the complete time series, and does not perform automatic outlier identification when a regARIMA model is estimated for one of the sliding spans. If the date of an outlier detected for the complete span of data does not occur in one of the sliding spans, the outlier will be dropped from the model for that

span. This is the default setting.

If `outlier = remove`, those outlier regressors that were added to the regression part of the regARIMA model when automatic outlier identification was performed on the full series are removed from the regARIMA model during the sliding spans analysis. Consequently, their effects are not estimated and removed from the series. If outlier terms are included in the **regression** spec, these will be included in the model estimated for the spans. This option gives the user a way to investigate the consequences of not doing automatic outlier identification.

If `outlier = yes`, the program performs automatic outlier identification whenever a regARIMA model is estimated for a span of data.

print and **save** The default output tables available for the direct and indirect seasonal adjustments generated by this spec are given in Table 7.39; other output tables available are given in Table 7.40. For a complete listing of the `brief` and `default` print levels for this spec, see Appendix B.

savelog The only diagnostic available for output to the log file (see Section 2.6) is the percentage of observations flagged as unstable for each of the estimates from the seasonal adjustment estimates tested by the sliding spans analysis.

Specifying `savelog = percents` or `savelog = pct` will store this information into the log file.

start The starting date for sliding spans comparisons. The default is the beginning month of the second span. Example: `start = 1990.jan`

RARELY USED ARGUMENTS

additivesa Specifies whether the sliding spans analysis of an additive seasonal adjustment will be calculated from the maximum differences of the seasonally adjusted series (`additivesa = difference`) or from the maximum of an implied adjustment ratio of the original series to the final seasonally adjusted series (`additivesa = percent`). This option will also determine if differences (`additivesa = difference`) or percent changes (`additivesa = percent`) are generated in the analysis of the month-to-month, quarter-to-quarter, or year-to-year changes in seasonally adjusted series. The default is `additivesa = difference`. If the seasonally adjusted series for any of the spans contains values that are less than or equal to zero, the sliding spans analysis will be performed on the differences.

fixx11reg Specifies whether the irregular component regression model will be re-estimated during the sliding spans analysis, if one is specified in the **x11regression** spec. If `fixx11reg = yes`, the regression coefficients of the irregular component regression model are fixed throughout the analysis at the values estimated from the entire series. If `fixx11reg = no`, the irregular component regression model parameters will be re-estimated for each span. The default is `fixx11reg = yes`.

x11outlier Specifies whether the AO outlier identification will be performed during the sliding spans analysis for the irregular component regression specified in the **x11regression** spec. If `x11outlier = yes`, AO outlier identification will be done for each span. Those AO

name	*short*	*save?*	*description of table*
header	hdr	·	header text for the sliding spans analysis
factormeans	fmn	·	range analysis for each of the sliding spans
percent	pct	·	table showing the percent of observations flagged as unstable for the seasonal and/or trading day factors, final seasonally adjusted series (if necessary), and the month-to-month (or quarter-to-quarter) changes
summary	sum	·	tables, histograms and hinge values summarizing the percentage of observations flagged for unstable seasonal and/or trading day factors, final seasonally adjusted series (if necessary), and month-to-month (or quarter-to-quarter) changes
yysummary	suy	+	additional tables, histograms and hinge values summarizing the percentage of observations flagged for the year-to-year changes
indfactormeans	fmi	·	range analysis for the implicit adjustment factors of the indirectly seasonally adjusted series
indpercent	pci	·	tables of the percent of observations flagged as unstable for the seasonal factors and month-to-month (or quarter-to-quarter) changes of the indirect seasonal adjustment
indsummary	smi	·	tables, histograms and hinge values summarizing the percentage of observations flagged for unstable seasonal factors, month-to-month (or quarter-to-quarter) and year-to-year changes for the indirect adjustment

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.39: **Default Output Tables for Slidingspans**

name	*short*	*save?*	*description of table*
ssftest	ssf	·	F-tests for stable and moving seasonality estimated over each of the sliding spans
yypercent	pcy	·	additional entry for the percent of observations flagged as unstable for the year-to-year changes
sfspans	sfs	+	seasonal factors from all sliding spans
chngspans	chs	+	month-to-month (or quarter-to-quarter) changes from all sliding spans
saspans	ads	+	seasonally adjusted series from all sliding spans
ychngspans	ycs	+	year-to-year changes from all sliding spans
tdspans	tds	+	trading day factors from all sliding spans
indyypercent	piy	·	additional entry for the percent of observations flagged as unstable for the year-to-year (or quarter-to-quarter) changes of the indirect seasonal adjustment
indyysummary	siy	·	additional tables, histograms and hinge values summarizing the percentage of observations flagged for the year-to-year changes of the indirect seasonal adjustment
indsfspans	sis	+	indirect seasonal factors from all sliding spans
indchngspans	cis	+	indirect month-to-month (or quarter-to-quarter) changes from all sliding spans
indsaspans	ais	+	indirect seasonally adjusted series from all sliding spans
indychngspans	yis	+	indirect year-to-year changes from all sliding spans

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.40: **Other Output Tables for Slidingspans**

outlier regressors that were added to the irregular component regression model when automatic AO outlier identification was done for the full series are removed from the irregular component regression model prior to the sliding spans run. If `x11outlier = no`, then the automatically identified AO outlier regressors for the full series are kept for each sliding spans run. If the date of an AO outlier detected for the complete span of data does not occur in one of the sliding spans, the outlier will be dropped from the model for that span. The coefficients estimating the effects of these AO outliers are re-estimated whenever the other irregular component regression model parameters are re-estimated. However, no additional AO outliers are automatically identified and estimated. This option is ignored if the **x11regression** spec is not used, if the selection of the **aictest** argument results in the program not estimating an irregular component regression model,

or if the **sigma** argument is used in the **x11regression** spec. The default is `x11outlier = yes`.

DETAILS

This section provides some additional information about the arguments within the sliding spans spec. Section 6.2 contains a description of the sliding spans diagnostics and their interpretation. For more details on the sliding spans procedure, see Findley, Monsell, Shulman, and Pugh (1990).

Different adjustment quantities are examined in a sliding spans analysis, depending on the mode of the seasonal adjustment and whether trading day adjustment is done. For a multiplicative or log-additive seasonal adjustment, the seasonal factors and the month-to-month and year-to-year changes of the seasonally adjusted series are analyzed. For a multiplicative or log-additive seasonal and trading day adjustment, the trading day factors and seasonally adjusted series are analyzed as well. For an additive seasonal adjustment without trading day adjustment, the seasonally adjusted series and the month-to-month and year-to-year changes of the seasonally adjusted series are analyzed. If trading day adjustment is done, these analyses are performed for the seasonal and trading day adjusted series.

WARNING: In the additive adjustment case, the presence of relatively small values or negative values in the adjusted series can render unusable the percent change values which are the basis of almost all of the sliding spans statistics. In this situation, usually only a subjective analysis of the spans of adjusted series obtained by using **saspan** in the **print** or **save** arguments can be used to detect excessive instability. Further research is needed to develop more useful sliding spans statistics for additive adjustments.

One important choice that needs to be made in a sliding spans analysis is the length of the overlapping spans. When used with the **x11** spec, the length of the span is based on the length of the seasonal filter since here seasonal adjustment is performed with fixed length seasonal filters. Table 7.41 gives a listing of how long the sliding span is by default for different seasonal filters of the **x11** spec.

Seasonal filter	Length of Span (in years)
3-term	6
3 x 3	6
3 x 5	8
3 x 9	11
3 x 15	17

Table 7.41: **Default Sliding Span Lengths for `X-11` Seasonal Filters**

Note that additional observations are added to the length of the span by default so that each of the sliding spans begin in January (or the first quarter for quarterly series). If different seasonal filters are used for different months or quarters, the longest span length of the seasonal filters chosen will be used.

If a stable seasonal filter is selected, the program will determine the span by dividing the length of the series by the number of spans used (with the default number of spans set to 4). The length of the span can be no longer than 17 years and no shorter than 3 years.

For model-based seasonal adjustments, the ARIMA model-based seasonal adjustment filters generated in the **seats** are always as long as the data span being adjusted (when the ARIMA model specified has a moving average component). Findley, Wills, Aston, Feldpausch, and Hood (2003) develops an approach for determining span lengths that is based on an analysis of SEATS model-based adjustment filters associated with the airline model, the model chosen for about half the series adjusted by SEATS, see Gómez and Maravall (1996). Since values of θ and Θ are known for which the SEATS seasonal adjustment filters have gain and phase-shift properties very close to those of the `X-11` filters, as shown in Planas and Depoutot (2002) and Findley and Martin (2006), the sliding span lengths used for SEATS adjustments within `X-13ARIMA-SEATS` are calibrated to coincide with the span lengths used for the `X-11` filters when the two types of filters are close. In this way, the span length specifications used for SEATS adjustments are anchored to those of the `X-11` filters.

Table 7.42 gives the span length used by the program for a given value of Θ. Research of the type described in Feldpausch, Hood, and Wills (2004) showed that for simulated series with known components, using the sliding spans lengths based on the seasonal moving average parameter seemed to provide a more reliable indication of inaccuracy in the seasonal adjustment than other diagnostics commonly used with SEATS seasonal adjustments.

Seasonal MA	Length of Span (in years)
0.160	5
0.325	6
0.490	7
0.535	8
0.620	9
0.640	10
0.695	11
0.710	12
0.750	13
0.760	14
0.795	15
0.805	16
0.840	17
0.850	18
0.910	19

Table 7.42: **Minimum Values of Seasonal MA Parameter for Span Lengths.**

If automatic ARIMA modeling selected is selected by either the **automdl** or **pickmdl** spec, then the model selected by the procedure is used for all sliding spans. If no model is selected by the procedure, then no model will be estimated during the sliding spans analysis.

While many of the tables in this spec cannot be saved as individual files, specifying the seasonal adjustment diagnostic summary option with the **-s** flag at runtime allows the user to store information from the sliding spans analysis into a diagnostic summary file (with the file extension **.udg**). In addition, the **savelog** argument can write selected diagnostics into the log file for a given run. For more information, see Section 2.6.

If a sliding spans analysis of the direct and indirect adjustments of a composite seasonal adjustment is desired, the sliding spans analysis option must be specified for each of the component series. If the seasonal

filter length is not the same for each component, then the user will have to use the **length** argument defined above in each of the input files of the component series to ensure that the spans analyzed for these series are of the same length.

If the automatic seasonal filter selection option is used, the seasonal filters used to generate the original seasonal adjustment will be used for the seasonal adjustment of each of the spans.

If an outlier specified by the user does not occur in a given span, that outlier will be dropped from the model for that span, and will be re-introduced into the regARIMA model if it is defined in future spans. User-defined regressors are checked to see if they are not constant in each span (i.e., all values of the regressor equal to zero).

EXAMPLES

The following examples show complete spec files.

Example 1 Multiplicative monthly seasonal adjustment, 3×9 seasonal factors for all calendar months. Sliding spans analysis performed with default settings for all options.

```
SERIES { FILE = "TOURIST.DAT"   START = 1996.1   }
X11 {   SEASONALMA = S3X9    }
SLIDINGSPANS {    }
```

Example 2 Log-additive seasonal adjustment of quarterly data, 3×9 seasonal filters for the first two quarters, 3×5 seasonal filters for the last two quarters, 7-term Henderson trend filter. Sliding spans analysis performed with threshold values for selected tests set to 5.0.

```
Series       {
    File = "qstocks.dat"
    Start = 1967.1
    Title = "Quarterly stock prices on NASDAQ"
    Freq = 4
}
X11          {
    Seasonalma = (  S3x9 S3x9 S3x5 S3x5  )
    Trendma = 7
    Mode = Logadd
}
Slidingspans {
    cutseas = 5.0
    cutchng = 5.0
}
```

Example 3 Seasonal ARIMA model with regression variables used for trading day adjustment and for automatic outlier identification and adjustment. Specified regression variables are a constant, trading day effects, and a ramp between May 1982 and Sept. 1982. The ARIMA part

of the model is $(0\ 1\ 2)(0\ 1\ 1)_{12}$. Perform an additive seasonal adjustment on the series after preadjusting for outliers and trading day regression effects. Perform sliding spans analysis; incorporate any outliers found by the application of the automatic identification procedure to the full series into the regARIMA model re-estimated for each of the sliding spans. The length of the sliding spans is set so that the sliding spans statistics from this run can be compared to a SEATS seasonal adjustment to be done in the next example.

```
series {  title = "Number of employed machinists - X-11"
          start = 1980.jan  file = "machine.emp"
}
regression  { variables = (const td rp82.may-82.oct) }
arima       { model = (0 1 2)(0 1 1) }
outlier     { }
estimate    { }
check       { }
forecast    { }
x11 { mode = add   save = d11}
slidingspans { outlier = keep
               length = 144
               }
```

Example 4 Same as Example 3, except that a model-based seasonal adjustment is performed using the **seats** spec.

```
series {  title = "Number of employed machinists - SEATS"
          start = 1980.jan  file = "machine.emp"
}
regression  { variables = (const td rp82.may-82.oct) }
arima       { model = (0 1 2)(0 1 1) }
outlier     { }
estimate    { }
check       { }
forecast    { }
seats { save = s11 }
slidingspans { outlier = keep
               length = 144
               }
```

Example 5 The predefined regression effects to be estimated are a constant, trading day and a fixed seasonal. The ARIMA part of the model is $(3, 1, 0)$. Generate 60 forecasts. Seasonally adjust the series after pre-adjusting for the estimated trading day. Perform sliding spans analysis. Re-estimate the values of the regARIMA model parameters for each span.

```
series { title = "Cheese Sales in Wisconsin"
   file = "cheez.fil"   start = 2005.1   }
transform {  function = log  }
```

```
regression {  variables = (const seasonal tdnolpyear)  }
arima {  model = (3 1 0)  }
forecast {   maxlead = 60  }
x11 {  save = seasonal   appendfcst = yes  }
slidingspans {  fixmdl = no  }
```

Example 6 Sliding spans analysis will be performed on the multiplicative seasonal adjustment specified, using 3 sliding spans of length 40 quarters as specified. This would allow the user to get some indication of seasonal adjustment stability, even though the series is not long enough for a complete sliding spans analysis with spans of the length most appropriate for 3×9 seasonal filters (44 quarters).

```
Series       {
    File = "qstocks.dat"
    Start = 1987.1        Freq = 4
    Title = "Quarterly stock prices on NASDAQ"
}
X11 {  Seasonalma = S3x9  }
Slidingspans {
    Length = 40 Numspans = 3
}
```

7.17 SPECTRUM

DESCRIPTION

Optional spec that provides a choice between two spectrum diagnostics to detect seasonality or trading day effects in monthly series. Users can set the starting date of the span of data to be used to estimate the spectra (**start**) and the type of spectrum estimate to be generated (**type**). For more information on the spectrum diagnostic, see Section 6.1.

In addition, the alternative QS statistic for detecting seasonality, applicable also to quarterly series, is described here, and its output is illustrated. There is also an option for generating the QS statistic for the quarterly version of a monthly series.

USAGE

```
spectrum {   logqs = yes
             qcheck = yes
             print = (none +specsa +specirr)
             save = (sp0 sp1 sp2)
             savelog = peaks
             start = 2005.Jan
             tukey120 = no
         }
```

ARGUMENTS

logqs Determines whether the log of the original series or seasonally adjusted series will be taken before the QS statistic is computed `logqs = yes`. The default is `logqs = no`.

print and **save** Table 7.43 gives the available output tables for this spec. All these tables are included in the default printout, except `npsa` and `npsaind`. For a complete listing of the `brief` and `default` print levels for this spec, see Appendix B.

In addition to the tables in 7.43, there are a number of tables related to the Tukey spectrum that can be saved but are not part of the main output. These tables are noted in Table 7.44.

qcheck Determines if the QS diagnostic will be generated for the quarterly version of a monthly series `qcheck = yes` to check for quarterly seasonality. The default is `qcheck = no`. This argument only produces output for monthly time series.

savelog The diagnostics available for output to the log file (see section 2.6) are listed in Table 7.45.

start The starting date of the span of data to be used to estimate the spectra the original, seasonally adjusted, and modified irregular series. This date must be in the format `start = year.seasonal period`. This can be used to determine if there are residual trading day

name	*short*	*save?*	*description of table*
npsa	npa	·	non-parametric test of residual seasonality for seasonally adjusted series
npsaind	npi	·	non-parametric test of residual seasonality for indirect seasonal adjustment
qcheck	qch	·	QS diagnostic to detect seasonality in quarterly version of a monthly series
qs	qs	·	QS diagnostic to detect seasonality
qsind	qsi	·	QS diagnostic to detect seasonality (indirect adjustment)
specorig	sp0	+	spectral plot of the first-differenced original series
specsa	sp1	+	spectral plot of differenced, `X-11` seasonally adjusted series (or of the logged seasonally adjusted series if `mode = logadd` or `mode = mult`)
specirr	sp2	+	spectral plot of outlier-modified `X-11` irregular series
specseatssa	s1s	+	spectrum of the differenced final SEATS seasonal adjustment
specseatsirr	s2s	+	spectrum of the final SEATS irregular
specextresiduals	ser	+	spectrum of the extended residuals
specresidual	spr	+	spectral plot of the regARIMA model residuals
speccomposite	is0	+	spectral plot of first-differenced aggregate series
specindirr	is2	+	spectral plot of the first-differenced indirect seasonally adjusted series
specindsa	is1	+	spectral plot of outlier-modified irregular series from the indirect seasonal adjustment
tukeypeaks	tpk	·	peak probability of Tukey spectrum

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.43: **Available Output Tables for Spectrum**

or seasonal effects in the adjusted data from, say, the last seven years. Residual effects can occur when seasonal or trading day patterns are evolving. The default starting date for the spectral plots is set to be 96 observations (8 years of monthly data) from the end of the series. If the span of data to be analyzed is less than 96 observations long, it is set to the starting date of this span of data. Example: `start = 1987.Jan`.

tukey120 Determines whether the value of `m` used to generate the Tukey spectrum will be set to 120 if the length of the series is greater than or equal to 120 (`tukey120 = yes`). If `tukey120`

name	*short*	*description of table*
spectukeyorig	st0	Tukey spectrum of the first-differenced original series
spectukeysa	st1	Tukey spectrum of the differenced, `X-11` seasonally adjusted series (or of the logged seasonally adjusted series if `mode = logadd` or `mode = mult`)
spectukeyirr	st2	Tukey spectrum of the outlier-modified `X-11` irregular series
spectukeyseatssa	t1s	Tukey spectrum of the differenced final SEATS seasonal adjustment
spectukeyseatsirr	t2s	Tukey spectrum of the final SEATS irregular
spectukeyextresiduals	ter	Tukey spectrum of the extended residuals
spectukeyresidual	str	Tukey spectrum of the regARIMA model residuals
spectukeycomposite	it0	Tukey spectrum of the first-differenced aggregate series
spectukeyindsa	it1	Tukey spectrum of the first-differenced indirect seasonally adjusted series
spectukeyindirr	it2	Tukey spectrum of the outlier-modified irregular series from the indirect seasonal adjustment

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.

Table 7.44: **Output Tables Available Only with `save` Argument for Spectrum**

`= no`, the length of the series is used to determine if the value of `m` will be 112 or 79. The default is `tukey120 = yes`.

RARELY USED ARGUMENTS

decibel Determines whether spectral estimates will be expressed in terms of decibel units `decibel = yes`, as shown in equation (6.1). The estimates are plotted on the untransformed scale if `decibel = no`. The default is `decibel = yes`.

difference If `difference = no`, the spectrum of the (transformed) original series or seasonally adjusted series is calculated; if `difference = first`, the spectrum of the month-to-month differences of these series is calculated. The default (`difference = yes`) will apply a $\max(d + D - 1, 1)$ difference to the (transformed) original series and seasonally adjusted series before computing the spectrum, where d is the order of regular differencing and D is the order of seasonal differencing in the regARIMA model specified for the series. If no regARIMA model is specified, the default order of differencing is 1.

maxar An integer value used to set the maximum order of the AR spectrum used as the default type of spectrum plot. Integers from 1 to 30 are acceptable values for **maxar**. If this option is not specified, the maximum order will be set to 30.

name	*short*	*description of diagnostic*
alldiagnostics	all	all spectral diagnostics
dirnpsa	dnp	non-parametric test of residual seasonality for direct seasonal adjustment
dirpeaks	dpk	visually significant peaks in spectra for direct seasonal adjustment
dirqs	dqs	QS diagnostic to detect seasonality, direct seasonal adjustment
dirtukeypeaks	dtp	peak probabilities for Tukey spectra for direct seasonal adjustment
indnpsa	inp	non-parametric test of residual seasonality for indirect seasonal adjustment
indpeaks	ipk	visually significant peaks in spectra for indirect seasonal adjustment
indqs	iqs	QS diagnostic to detect seasonality, indirect seasonal adjustment
indtukeypeaks	itp	peak probabilities for Tukey spectra for inddirect seasonal adjustment
npsa	npa	non-parametric test of residual seasonality for seasonally adjusted series
peaks	spk	visually significant peaks in spectra
qcheck	qch	QS diagnostic to detect seasonality in quarterly version of a monthly series
qs	qs	QS diagnostic to detect seasonality
tukeypeaks	tpk	significant peaks of Tukey spectrum

Name gives the name of each diagnostic for use with the **savelog** argument.
Short gives a short name for these diagnostics.

Table 7.45: **Available Log File Diagnostics for Spectrum**

peakwidth Allows the user to set the width of the band used to determine spectral peaks. The default value is `peakwidth = 1`.

robustsa Allows the user to select the type of series used in the spectrum of the seasonally adjusted (or indirect seasonally adjusted) or series (Table G.1) or the irregular series (Table G.2). If `robustsa = yes`, then the extreme value adjusted (along with level shifts, if specified) versions of these series (Table E.2 and E.3 for `X-11` adjustments, outlier adjusted versions of SEATS components) are used in the spectrum for Tables G.1 and G.2. If `robustsa = no`, then the versions used (Table D.11 and D.13 for `X-11` adjustments, the SEATS seasonal adjustments and irregular) have not been extreme value adjusted. The default value is `robustsa = yes`.

series Allows the user to select the series used in the spectrum of the original (or composite) series (Table G.0). Table 7.46 shows the series that can be specified with this argument - the default is `series = adjoriginal` (or `b1`).

name	*short*	*description of table*
original	a1	original series
outlieradjoriginal	a19	original series, adjusted for regARIMA outliers
adjoriginal	b1	original series, adjusted for user specified and regARIMA prior effects
modoriginal	e1	original series modified for extremes

Name gives the name of each series which can be specified for use with the **series** argument.
Short gives a short name for the options of the **series** argument.

Table 7.46: **Choices for the `series` Argument of Spectrum**

Note that if the **x11** spec is not specified, the original series modified for extremes will not be generated; the setting `series = modoriginal` will be ignored, and the default setting will be used instead.

siglevel Sets the significance level for detecting a peak in the spectral plots. The default is `siglevel = 6`.

type The type of spectral estimate used in the spectral plots output by the program. If `type = periodogram`, the periodogram of the series is calculated and plotted. The default (`type = arspec`) produces an autoregressive model spectrum of the series.

DETAILS

A routine searches each of the spectra for peaks at the seasonal and trading day frequencies. A warning message is printed out if visually significant peaks are found, and the plot in which a peak was found is printed out. When the restricted output (the **-n** flag) option is used, the plot is not produced in the main output, but a message is printed suggesting that the user rerun the program without the **-n** flag.

For more information on the spectrum diagnostic, see Section 2.1 of Findley, Monsell, Bell, Otto, and Chen (1998) and Soukup and Findley (1999) as well as Section 6.1.

QS is a statistic that provides a test of the hypothesis of no seasonality. It is applied to appropriate series associated with the modeling and the seasonal adjustment of a given series. These include the original series (log transformed, regression and/or extreme-value adjusted as appropriate) and various output series, most importantly the seasonally adjusted series (log transformed, etc. as appropriate). See Table 7.47.

Let s be the seasonal period (12 for monthly data, 4 for quarterly data, for example). For an appropriately differenced version of each series, QS tests whether a positive autocorrelation at lag s is statistically significantly different from zero; or, when the lag s and lag $2s$ autocorrelations are both positive, whether at least one of the two is significantly different from zero.

For each of these series, to calculate its QS, first the order *ndif* of nonseasonal differencing is determined that will be applied to it. For the irregular component and for ARIMA model residuals, $ndif = 0$. For other

```
QS statistic for seasonality:
Original Series                                77.73  (P-Value =     0.0000)
Original Series (EV adj)                       77.73  (P-Value =     0.0000)
Residuals                                       0.01  (P-Value =     0.9953)
Seasonally Adjusted Series                      0.00  (P-Value =     1.0000)
Seasonally Adjusted Series (EV adj)             0.00  (P-Value =     1.0000)
Irregular Series                                0.00  (P-Value =     1.0000)
Irregular Series (EV adj)                       0.00  (P-Value =     1.0000)

QS statistic for seasonality (starting 1998.Jan):
Original Series                                43.70  (P-Value =     0.0000)
Original Series (EV adj)                       43.70  (P-Value =     0.0000)
Residuals                                       0.00  (P-Value =     0.9997)
Seasonally Adjusted Series                      0.00  (P-Value =     1.0000)
Seasonally Adjusted Series (EV adj)             0.00  (P-Value =     1.0000)
Irregular Series                                0.00  (P-Value =     1.0000)
Irregular Series (EV adj)                       0.00  (P-Value =     1.0000)
```

Table 7.47: **Example of QS Statistic Output**

series, when an ARIMA model is available that specifies an order d of nonseasonal differencing and/or an order D of seasonal differencing, then *ndif* is initially set as

$$ndif = \max(1, \min(d + D, 2))$$

If there is no ARIMA model, *ndif* is initially set to 1.

To calculate QS, sample autocorrelations are calculated in the usual way from the (always sample-mean centered) *ndif*-times nonseasonally differenced series. When $ndif = 1$ these autocorrelations are provisional: a check is done to see if they decay too slowly from a positive value at lag 1. The criterion depends on s. When $s = 6$ or $s = 12$, the program checks if the autocorrelations at lags 1, 2, 3, 4, and s are positive. When $s = 2$ or $s = 4$, the program checks if the autocorrelations at lags 1 to s are all greater than 0.2. If either of these conditions is satisfied, and $ndif = 1$, then *ndif* is reset to 2, and the autocorrelations used for QS are obtained from the twice-differenced series.

Let r_i denote the lag i autocorrelations for $i = s, 2s$, and let nz denote the length of the series. Set $n = nz - ndif$ and

$$R_i = \begin{cases} r_i, & \text{if } r_i > 0 \\ 0, & \text{if } r_i \leq 0. \end{cases}$$

If $R_s > 0$, set

$$QS = n(n+2)\left\{\frac{R_s^2}{n-s} + \frac{R_{2s}^2}{n-2s}\right\}.$$

If $R_s = 0$, set $QS = 0$. The statistic QS is assumed to be adequately approximated by a chi-squared distribution with two degrees of freedom. This is supported by simulations as illustrated in Maravall (2012).

The QS output only indicates that seasonality is present when QS is larger than a preset critical value for this distribution (usually one corresponding to a 0.01 significance level).

In the `X-13ARIMA-SEATS` sample output shown in Table 7.47, the output indicates that there is seasonality in the original series, no residual seasonality in the seasonally adjusted series, and no seasonality in the regARIMA model residuals.

For monthly series, setting `qcheck = yes` generates QS statistics for the quarterly versions of the original series, the original series adjusted for extreme values, the seasonally adjusted series, and the seasonally adjusted series adjusted for extreme values. If the starting date for the spectrum is different from the starting date of the series, QS statistics will be generated for the shortened series as well. An example of this output is given in Table 7.48.

```
 QS statistic for (quarterly) seasonality:
log(Original Series)                                98.36  (P-Value =     0.0000)
log(Original Series (EV adj))                      136.43  (P-Value =     0.0000)

 QS statistic for (quarterly) seasonality (starting 2006.1):
log(Original Series)                                38.56  (P-Value =     0.0000)
log(Original Series (EV adj))                       37.56  (P-Value =     0.0000)

 QS statistic for (quarterly) seasonality:
log(Seasonally Adjusted Series)                      0.00  (P-Value =     1.0000)
log(Seasonally Adj. Series (EV adj))                 1.44  (P-Value =     0.4872)

 QS statistic for (quarterly) seasonality (starting 2006.1):
log(Seasonally Adjusted Series)                      0.97  (P-Value =     0.6160)
log(Seasonally Adj. Series (EV adj))                 1.03  (P-Value =     0.5961)
```

Table 7.48: **Example of QS Statistic Output with `qcheck = yes`**

The quarterly series are only formed from complete quarters – quarters where all three months are available in the monthly series. Sometimes this means that the series will be truncated at either end.

EXAMPLES

Example 1 Multiplicative monthly seasonal adjustment, 3×9 seasonal filters for all months, 23-term Henderson moving average for the trend-cycle. Produce QS statistics with the log of the original series and the log of the seasonally adjusted series as well as the irregular component.

```
Series { File="klaatu.dat" Start = 2006.1 }
X11 { SeasonalMA =  s3x9  TrendMA = 23  }
Spectrum { savelog = all  logqs = yes  }
```

Example 2 This example shows how to obtain a spectrum plot of the first differences (month-to-month differences) of the logarithms of the series to check if the series has seasonal or trading day effects. This is a complete spec file.

```
series{ title = "Spectrum analysis of Building Permits Series"
        start = 1967.Jan
        file = "permits.dat"
        format = "(12f6.0)"
        print = none
      }
transform{ function = log
           print = none
         }
spectrum{ start = 1987.Jan
          print = (none +specorig)
          savelog = all
        }
```

Example 3 This spec file for a composite seasonal adjustment will generate the periodogram instead of the AR spectrum and will store information on whether there are peaks in the indirect seasonal adjustment and the results of the QS statistic for the indirect adjustments in the log file. The periodogram for the indirect seasonal adjustment will be saved. This is a complete spec file, but must be run with other spec files in a metafile; more information on composite seasonal adjustment is available in Section 7.4.

```
composite {  title="TOTAL ONE-FAMILY Housing Starts"
             name="C1FTHS" save=(indseasonal) }
x11 { seasonalma=(s3x9)
      title="Composite adj. of 1-Family housing starts"
      save=(D10) }
spectrum { savelog = (indpeaks indqs)
           type = periodogram
           save = is1 }
```

Example 4 Multiplicative default monthly seasonal adjustment. Produce QS statistics with the log of the original series, the log of the seasonally adjusted series, and the irregular component, as well as the quarterly version of the original series.

```
series { title = "Total U.S. Retail Sales"
         file = "uSretail.dat"  start = 1988.jan  }
transform { function = log }
regression { variables = ( td easter[8] labor[8] ) }
arima {  model =(0 1 1)(0 1 1)  }
forecast { maxlead = 60 }
spectrum { savelog = all  logqs = yes  qcheck=yes }
x11 { save=d11 }
```

7.18 TRANSFORM

DESCRIPTION

Specification used to transform or adjust the series prior to estimating a regARIMA model. With this spec the series can be Box-Cox (power) or logistically transformed, length-of-month adjusted, and divided by user-defined prior adjustment factors. Data for any user-defined prior adjustment factors must be supplied, either in the **data** argument or in a file specified by the **file** argument (but not both). For seasonal adjustment, a set of permanently removed factors can be specified as well as a set of factors that are temporarily removed until the seasonal factors are calculated.

USAGE

```
transform {   function = log  or  power = 0.0
              adjust = lom
              title = "prior adjustment factors"
              start = 2005.jan
              data = (1.25 ··· 1.90)  or  file = "prioradj.dat"
                                          format = "(6f12.3)"
              name = "Adjfac"
              aicdiff = 0.0
              mode = ratio
              type = temporary
              print = (none)
              save = (prior prioradjusted)
              savelog = atr
          }
```

ARGUMENTS

adjust Perform length-of-month adjustment on monthly data (`adjust = lom`), length-of-quarter adjustment on quarterly data (`adjust = loq`), or leap year adjustment of monthly or quarterly data (`adjust = lpyear`). (See DETAILS.)

Do not use the **adjust** argument if `td` or `td1coef` is specified in the **variables** argument of the **regression** or **x11regression** specs, or if additive or pseudo-additive seasonal adjustment is specified in the **mode** argument of the **x11** spec. Leap year adjustment (`adjust = lpyear`) is only allowed when a log transformation is specified in either the **power** or **function** arguments.

aicdiff Defines the difference in AICC needed to accept no transformation when the automatic transformation selection option is invoked (`function = auto`). The default value is `aicdiff = -2.0` for monthly and quarterly series, `aicdiff = 0.0` otherwise. For more information on how this option is used to select a transformation, see DETAILS.

data An array containing one or two series of preadjustment factors which, unless `mode = diff` (see below), must have positive values intended for division into the corresponding values of the input time series. The default value is a vector of ones (no prior adjustment). When **data** (or **file**) is used, an adjustment factor must be supplied for every observation in the series (or for the span specified by the **span** argument of the **series** spec, if present). Generally, an adjustment factor must also be supplied for each forecast (and backcast) desired. (See DETAILS.) The adjustment factors are read in free format. If a start date is supplied for the adjustment factors, then they may start before the beginning of the series. If the **data** argument is used, the **file** argument cannot be used. When `mode = diff`, the values in **data** are subtracted from the series, and they need not be positive.

Two series can be input via the **data** argument when both permanent and temporary prior adjustment factors are specified in **type** – see DETAILS for more information.

file Name of the file containing the user-defined prior adjustment factors. The filename must be enclosed in quotes. If the file is not in the current directory, the path must also be given. If the **file** argument is used, the **data** argument cannot be used. The value restrictions are the same as for **data**. If the data in the file are not in free format, the **format** argument must be used.

If both permanent and temporary prior adjustment factors are specified in **type**, the factors can be input from a single file or from two files – see DETAILS for more information.

format Denotes the format used to read the prior adjustment factors from a file. Eight types of input are accepted:

a. free format, in which all numbers on a line will be read before continuing to the next line, and the numbers must be separated by one or more spaces (not by commas or tabs) (example: `format = "free"`);

b. a valid Fortran format, which should be enclosed in quotes and must include the initial and terminal parentheses (example: `format = "(6f12.0)"`).

c. a two character code which corresponds to a set of data formats used in previous versions of `X-11` and `X-11-ARIMA` (example: `format = "1r"`);

d. "datevalue" format, where the year, month or quarter, and value of each observation are found in this order in free format on individual lines. Thus, a line of the data file containing the value 32531 for July 1991 would have the form `1991 7 32531`. The number of preceding blanks can vary (example: `format = "datevalue"`);

e. the format `X-13ARIMA-SEATS` uses to save a table. This allows the user to read in a file saved from a previous `X-13ARIMA-SEATS` run (example: `format = "x13save"`);[24]

f. the format that the `TRAMO` and `SEATS` programs use to read in a series and its descriptors. This enables `X-13ARIMA-SEATS` to read in a data file formatted for the `TRAMO` modeling program or the `SEATS` seasonal adjustment program. (example: `format = "tramo"`);

g. a variant of "free" format where the numbers must be separated by one or more spaces (not by commas or tabs), and decimal points are expressed as commas (a convention in some European countries). (example: `format = "freecomma"`);

[24]Note that to maintain compatibility with previous versions of `X-12-ARIMA` the entry **x12save** will also be accepted.

h. a variant of "datevalue" format, where the year, month or quarter, and value of each observation are found in this order in free format on individual lines, where decimal points are expressed as commas. Thus, a line of the data file containing the value 1355.34 for July 1991 would have the form `1991 7 1355,34`. The number of preceding blanks can vary (example: `format="datevaluecomma"`).

In the predefined `X-11` data formats, the data is stored in 6 or 12 character fields, with a year and series label associated with each year of data. For a complete list of these formats, see Table 7.38 in the DETAILS section of the **series** spec.

If no format argument is given, the data will be read in free format. The **format** argument can only be used with the **file** argument, not with **data**.

If permanent and temporary prior adjustment factors are input from two different files with distinct formats, then up to two formats can be specified – see DETAILS for more information.

function Transform the series Y_t input in the **series** spec using a log, square root, inverse, or logistic transformation. Alternatively, perform an AIC-based selection to decide between a log transformation and no transformation (`function = auto`) using either the regARIMA model specified in the **regression** and **arima** specs or the airline model (0 1 1)(0 1 1) (see DETAILS). The default is no transformation (`function = none`). Do not include both the **function** and **power** arguments. **Note:** there are restrictions on the values used in these arguments when preadjustment factors for seasonal adjustment are generated from a regARIMA model; see DETAILS.

value	*transformation*	*range for* Y_t	*equivalent* **power** *argument*
`none`	Y_t	*all values*	`power = 1`
`log`	$\log(Y_t)$	$Y_t > 0$ for all t	`power = 0`
`sqrt`	$0.25 + 2\left(\sqrt{Y_t} - 1\right)$	$Y_t \geq 0$ for all t	`power = 0.5`
`inverse`	$2 - (1/Y_t)$	$Y_t \neq 0$ for all t	`power = -1`
`logistic`	$\log(Y_t/(1 - Y_t))$	$0 < Y_t < 1$ for all t	*no equivalent*

Table 7.49: **Transformations Available Using the `function` Argument for Transform**

mode Specifies the way in which the user-defined prior adjustment factors will be applied to the time series. If prior adjustment factors to be divided into the series are not given as percents (e.g., (`100 100 50` $\cdots$)), but rather as ratios (e.g., (`1.0 1.0 0.5` $\cdots$)), set `mode = ratio`. If the prior adjustments are to be subtracted from the original series, set `mode = diff`. If `mode = diff` is used when the mode of the seasonal adjustment is set to be multiplicative or log additive in the **x11** spec, the factors are assumed to be on the log scale. The factors will be exponentiated to put them on the same basis as the original series. If this argument is not specified, then the prior adjustment factors are assumed to be percents (`mode = percent`).

If both permanent and temporary prior adjustment factors are specified in the **type** argument, then up to two values can be specified for this argument, provided they are compatible (e.g., `diff` cannot be specified along with `ratio` or `percent`). See DETAILS for more information.

name The name of the prior adjustment factors. The name must be enclosed in quotes and may contain up to 64 characters. Up to the first 16 characters will be printed as a label for the prior adjustment factors. When specified with the `X-11` formats of the **format** argument, the first six (or eight, if `format = "cs"`) characters of this name are also used with the predefined formats to check that the program is reading the correct series, or to find a particular series in a file where many series of factors are stored.

If both permanent and temporary prior adjustment factors are specified in **type**, then the user can specify series names for both sets of prior adjustment factors, or no name should be entered – see DETAILS for more information.

power Transform the input series Y_t using a Box-Cox power transformation,

$$Y_t \longrightarrow y_t = \begin{cases} \log(Y_t) & \lambda = 0; \\ \lambda^2 + (Y_t^\lambda - 1)/\lambda & \lambda \neq 0. \end{cases}$$

This formula for the Box-Cox power transformation is constructed so that its values will be close to Y_t when λ is near 1 and close to $\log Y_t$ when λ is near zero. It also has the property that the transformed value is positive when Y_t is greater than 1.

The power λ must be given (e.g., `power = 0.33`). The default is no transformation ($\lambda = 1$), i.e., `power = 1`. The log transformation (`power = 0`), square root transformation (`power = 0.5`), and the inverse transformation (`power = -1`) can alternatively be given using the **function** argument. Do not use both the **power** and the **function** arguments in the same spec file. **Note:** there are restrictions on the values used in these arguments when preadjustment factors for seasonal adjustment are generated from a regARIMA model; see DETAILS.

precision The number of decimal digits to be read from the file of prior adjustment factors. This option can only be used with the predefined formats of the **format** argument. This value must be an integer between 0 and 5, inclusive (for example, `precision = 5`). The default is zero. If **precision** is used in a **transform** spec that does not use one of the predefined formats, the argument is ignored.

If both permanent and temporary prior adjustment factors are specified in the **type** argument, then up to two values can be specified for this argument – see DETAILS for more information.

print and **save** Table 7.50 gives the available output tables for this spec. The `aictransform`, `prior`, and `prioradjusted` tables are printed out by default. For a complete listing of the `brief` and `default` print levels for this spec, see Appendix B.

savelog Setting `savelog = autotransform` or `savelog = atr` causes the result of the automatic transformation selection procedure to be output to the log file (see section 2.6 for more information on the log file).

name	*short*	*save?*	*description of table*
aictransform	tac	·	output from AIC-based test(s) for transformation
seriesconstant	a1c	+	original series with value from the constant argument added to the series
seriesconstantplot	acp	·	plot of original series with value from the constant argument added to the series
prior	a2	+	prior adjustment factors, with associated dates
permprior	a2p	+	permanent prior adjustment factors, with associated dates
tempprior	a2t	+	temporary prior adjustment factors, with associated dates
prioradjusted	a3	+	prior adjusted series, with associated dates
permprioradjusted	a3p	+	prior adjusted series using only permanent prior factors, with associated dates
prioradjustedptd	a4d	+	prior adjusted series (including prior trading day adjustments), with associated dates
permprioradjustedptd	a4p	+	prior adjusted series using only permanent prior factors and prior trading day adjustments, with associated dates
transformed	trn	+	prior adjusted and transformed data, with associated dates

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.50: **Available Output Tables for Transform**

start The start date of the user-defined prior adjustment factors. The default is the start date of the series. Valid values are any date up to the start date of the series (or up to the start date of the span specified by the **span** argument of the **series** spec, if present).

If both permanent and temporary prior adjustment factors are specified in the **type** argument, then up to two starting dates can be specified to read in the two sets of prior adjustment factors – see DETAILS for more information.

title A title for the set of user-defined prior adjustment factors. The title must be enclosed in quotes and may contain up to 79 characters.

type Specifies whether the user-defined prior adjustment factors are permanent factors (removed from the final seasonally adjusted series as well as the original series) or temporary factors (removed from the original series for the purposes of generating seasonal factors but not from the final seasonally adjusted series). If only one value is given for this argument (`type = temporary`), then only one set of user-defined prior adjustment factors will be expected. If both types of user-defined prior adjustment factors are given (`type = (temporary permanent)`), then two sets of prior adjustment factors will be expected; for more information see DETAILS. The default is `type = permanent`.

RARELY USED ARGUMENTS

constant Positive constant value that is added to the original series before the program models or seasonally adjusts the series. Once the program finishes modeling and/or seasonally adjusting the series with the constant value added, this constant is removed from the seasonally adjusted series as well as the trend component.

trimzero If `trimzero = no`, zeros at the beginning or end of a time series entered via the **file** argument are treated as series values. If `trimzero = span`, leading and trailing zeros outside the span of data being analyzed are ignored (the **span** argument of the **series** spec must be specified with both a starting date and an ending date). The default (`trimzero = yes`) causes leading and trailing zeros to be ignored. Note that when the **format** argument is set to either `datevalue`, `x13save`, or `tramo`, all values input are treated as series values, regardless of the value of **trimzero**.

DETAILS

If a Box-Cox or logistic transformation is specified in conjunction with a length-of-month (or leap year) adjustment and/or user-defined prior adjustment factors, the time series is first adjusted for length-of-month and/or prior factors, and then Box-Cox or logistically transformed. If both length-of-month and prior adjustment factors are specified, then combined adjustment factors (length-of-month $\times$ prior adjustment) are used. Length-of-quarter and leap year adjustments are handled in the same way.

If either `lom` or `loq` of the **adjust** argument is specified, the correct adjustment factor is determined by the **period** specified in the **series** spec. In the case of a monthly input series Y_t, each observation is first divided

by the number of days in that month (m_t) and then multiplied by the average length-of-month (30.4375), resulting in $(30.4375 \times Y_t)/m_t$. Length-of-quarter adjustments are performed in a similar manner, resulting in $(91.3125 \times Y_t)/q_t$, where q_t is the length in days of quarter t. Forecasts of the transformed and length-of-month adjusted data are transformed back to the original scale for output (see the documentation of the **forecast** spec).

If adjustment factors are supplied for the forecast period, then forecasts of the prior adjusted series will be inverse-transformed (multiplied or, if `mode = diff`, added to) with these factors. If adjustment factors are not supplied for the forecast period, then inverse-transformation of forecasts will only account for a Box-Cox or logistic transformation and for any length-of-month (or length-of-quarter) adjustment —this effectively assumes values of 1 for the user-defined prior adjustment factors throughout the forecast period (or 0 if `mode = diff`).

When seasonal adjustment is requested (using **x11** or **seats**), any value of **power** or **function** can be used for the purpose of forecasting the series with a regARIMA model. However, this is not the case when factors generated from the regression coefficients are used to adjust either the original series or the final seasonally adjusted series. In this case, the only accepted transformations are the log transformation (for multiplicative or log-additive seasonal adjustments) and no transformation, which can be specified as `power = 1` (for additive seasonal adjustments).

This restriction on the transformation is done because factors derived from the regression coefficients must be of the same type as factors generated by the seasonal adjustment procedure, so that combined adjustment factors can be derived and adjustment diagnostics generated. If the regARIMA model is applied to a log-transformed series, the regression factors are expressed in the form of ratios, which is the same form as the seasonal factors generated by the multiplicative (or log-additive) adjustment mode. Conversely, if the regARIMA model is fit to the original series, the regression factors are measured on the same scale as the original series, which matches the scale of the seasonal factors generated by the additive adjustment mode.

If no seasonal adjustment is done, any power transformation can be used.

When `function = auto` and the series being processed has all positive values, the program will choose between no transformation and a log transformation by fitting a regARIMA model to the untransformed and transformed series. The log transformation will be favored unless

$$AICC_{\text{nolog}} - AICC_{\log} < \Delta_{AICC} \quad \text{or} \quad AICC_{\log} + \Delta_{AICC} > AICC_{\text{nolog}}, \tag{7.18}$$

where $AICC_{\log}$ is the value of AICC from fitting the regARIMA model to the transformed series, $AICC_{\text{nolog}}$ is the value of AICC from fitting the regARIMA model to the untransformed series, and Δ_{AICC} is the value entered for the **aicdiff** argument, with a default of -2. Negative values of Δ_{AICC} bias the selection in favor of the log transformation. The default of -2 is used not for statistical reasons but for convenience. Multiplicative adjustment is appropriate for the vast majority of Census Bureau series, and we would prefer not to inconvenience users accustomed to multiplicative adjustments unless there is strong statistical support for additive adjustment. The AICC value for the log transformed series (or any transformed series) is obtained by applying an appropriate (Jacobian) adjustment to the log likelihood to make it compatible with the log likelihood of the estimated model of the untransformed series. (The adjustment is printed in the output if `print = lkf` is specified in the **estimate** spec.) If the series has a zero or negative value, no transformation is used.

If a regARIMA model has been specified in the **regression** and/or **arima** specs, then the procedure will use this model to generate the AICC statistics needed for the test. If no model is specified, or the automatic

model identification procedure is specified via the **automdl** or **pickmdl** spec, the program will use the airline model ((0 1 1)(0 1 1) in Box-Jenkins notation) to generate the AICC statistics.[25]

If seasonal adjustment is specified via the **x11** or **x11regression** spec, the program will set the seasonal adjustment mode to one that is appropriate for the transformation selected (multiplicative for a log transformation, additive for no transformation).

The program currently does not allow the use of user-defined prior adjustment factors with the automatic transformation selection option.

Users specifying both temporary and permanent user-defined prior adjustment factors must take advantage of some special features built into the **transform** spec. For the arguments related to data input, the user can specify an entry for each type of prior adjustment factor. The **type** argument tells the program which type of prior factor is being referred to by a given entry. For example, in the input specified below, the series of temporary prior-adjustment factors is read from `temp.fil` using a `(6F12.5)` format. These factors start in January 2000. The series of permanent prior adjustment factors, which starts in July 1995, is read from `perm.fil` using a `(F15.3)` format.

```
transform{
        type = (temporary permanent)
        file = ("temp.fil" "perm.fil")
        format = ("(6F12.5)" "(F15.3)")
        start = (2000.jan 1995.jul)
        mode = (ratio percent)
}
```

If two entries are given for the **file** argument, but only one entry for each of the **format**, **start**, **mode**, and **precision** arguments, then the values given are assumed to apply to both sets of factors. The number of values given for the **name** argument must match the number of prior adjustment factors implied by the **type** argument.

When the **data** argument is used to input two sets of prior adjustment factors, the data is assumed to be a matrix of two columns. The type assignment for the data columns is determined by the **type** argument. In the example below, the first column of data is interpreted to be a temporary prior adjustment factor (with values of 1.055, 0.990, and 1.025), and the second column of data is interpreted to be a permanent prior adjustment factor.

[25]Note that if only the **regression** spec is specified, the `X-13ARIMA-SEATS` default ARIMA model is the (0 0 0)(0 0 0) model (in Box-Jenkins notation). In other words, if the regression model includes trading day, but no ARIMA model is specified, then the program will use a (0 0 0)(0 0 0) ARIMA model and trading day regressors to generate the AICC statistics.

```
transform{
      type = (temporary permanent)
      data = ( 1.055     1.000
               0.990     1.000
               1.025     1.000
                .         .
                .         .
                .         .
               1.033     1.000 )
      start = 1980.jan
      mode = ratio
}
```

The same assumption is made when only one data file is given for two adjustment types, as in the input below.

```
transform{
      type = (temporary permanent)}
      file = "both.fil"}
      start = 1980.jan}
      mode = ratio}
}
```

X-13ARIMA-SEATS accepts a maximum of 780 user-defined prior adjustment factors of each type (temporary or permanent), not including the forecast period. (This limit can be changed—see Section 2.8.)

The **constant** argument is sometimes useful in the case where a series has a number of values close to zero such that neither multiplicative nor additive seasonal adjustment seems to be effective, or in the case where a series has zero or negative values and the series seems to behave in a manner that ordinarily calls for a multiplicative seasonal adjustment. Strategies for selecting the value of **constant**, as well as an application to Canadian air travel series, can be found in Chen and Durk (2005).

EXAMPLES

Note: The following examples do not show "complete" spec files. Useful output is not produced unless additional specs (e.g., **x11**, **identify**, or **arima** and **estimate**) are added.

Example 1 Specify a user-defined prior adjustment for a strike in March and April of 1967, as well as a length-of-month adjustment.

```
series { data = (879 899 462 670 985 973 ...)
         start = 1997.jan }
transform { data  = (1  1 .5 .75   1   1 ...)
            mode = ratio
            adjust = lom }
```

Example 2 Specify a constant to add to the series before modeling and seasonal adjustment. Use the automatic transformation selection procedure to determine if a log transformation should be used to transform the resulting series.

```
series { data = (6 79 98 42 4 73 85 26 ...)
         start = 1997.1  period=4 }
transform { constant=45  function = auto  }
```

Example 3 Specify a logarithmic transformation and also a user-defined adjustment by a price deflator that changes current dollars to constant (real) dollars. A start date is specified for the deflator series since it begins before the time series being modeled.

```
series {title = "Total U.S. Retail Sales --- Current Dollars"
        file = "retail.dat"
        start = 2000.jan }
transform {function = log
           title = "Consumer Price Index"
           start = 1990.jan  # adj. factors start January, 1990
           file = "cpi.dat"
           format  =  "(12f6.3)" }
```

Example 4 Same as Example 3, only a pre-defined format is used to read in the user-defined adjustment factors, and the factors are applied as temporary prior adjustment factors.

```
series {title = "Total U.S. Retail Sales --- Current Dollars"
        file = "retail.dat"
        start = 2000.jan }
transform {function = log
           title = "Consumer Price Index"
           start = 1990.jan  # adj. factors start January, 1990
           file = "cpi.dat"
           format  =  "1R"
           precision = 3
           name = "cpi"
           type = temporary
          }
```

Example 5 Specify a cube root transformation to stabilize the variation of a quarterly time series.

```
SERIES {TITLE="Annual Rainfall"
        FILE="RAIN.DAT"
        PERIOD=4
        START=1991.1}
TRANSFORM {POWER=.3333}
```

Example 6 This example uses two sets of user-defined prior adjustment factors: one for the Consumer Price Index that will be removed from the final seasonally adjusted series to convert the value of the series to current dollars (a permanent prior effect), and a set of strike effects (a temporary prior effect). Each set of factors is read from its own file. Since the files have the same format, single values are entered for **format** and **precision**.

```
series {title = "Retail Sales of computers --- Current Dollars"
        file = "rscomp.dat"
        start = 2000.jan
       }
transform { function = log
            title = "Consumer Price Index & Strike Effect"
            type = (permanent temporary)
            start = 1990.jan  # adj. factors start January, 1990
            file = ("cpi.dat" "strike.dat")
            format  = "1R"
            precision = 3
            name = ("cpi" "strike")
          }
```

Example 7 Use the automatic transformation selection procedure to determine if a log transformation should be used to transform the series. Since a regARIMA model is not specified, the program will use an airline model to generate the AICC values needed for the test. The AICC difference for the test has been reset to zero, so the program will choose the transformation based on which model estimation yields the smaller value of AICC.

```
series {title = "Total U.K. Retail Sales"
       file = "ukretail.dat"
       start = 1998.jan
       }
transform {function = auto
           aicdiff = 0.0
          }
```

7.19 X11

DESCRIPTION

An optional spec for invoking seasonal adjustment by an enhanced version of the methodology of the Census Bureau `X-11` and `X-11Q` programs. The user can control the type of seasonal adjustment decomposition calculated (**mode**), the seasonal and trend moving averages used (**seasonalma** and **trendma**), and the amount of extreme value adjustment performed during seasonal adjustment (**sigmalim**). The output options, specified by **print** and **save**, include final tables and diagnostics for the `X-11` seasonal adjustment method. In `X-13-ARIMA-SEATS`, additional specs can be used to diagnose data and adjustment problems, to develop compensating prior regression adjustments, and to extend the series by forecasts and backcasts. Such operations can result in a modified series from which the `X-11` procedures obtain better seasonal adjustment factors. For more details on the `X-11` seasonal adjustment diagnostics, see Shiskin, Young, and Musgrave (1967), Lothian and Morry (1978), and Ladiray and Quenneville (2001). Trading day effect adjustments and other holiday adjustments can be obtained from the **x11regression** spec.

USAGE

```
x11 {   mode = pseudoadd
        seasonalma = s3x9
        trendma = 13
        sigmalim = (1.25 2.75)
        title = "3x9 moving average, mad"
        appendfcst = yes
        appendbcst = no
        type = trend
        final = user
        print = ( brief +b2)
        save = (d10 d11)
        savelog = (m7 q)
    }
```

ARGUMENTS

appendbcst Determines if backcasts will be included in certain `X-11` tables selected for storage with the **save** option. If `appendbcst = yes`, then backcasted values will be stored with tables a16, b1, d10, d16, and h1. If `appendbcst = no`, no backcasts will be stored. The default is to not include backcasts.

appendfcst Determines if forecasts will be included in certain `X-11` tables selected for storage with the **save** option. If `appendfcst = yes`, then forecasted values will be stored with tables a16, b1, d10, and d16. If `appendfcst = no`, no forecasts will be stored. The default is to not include forecasts.

final List of the types of prior adjustment factors, obtained from the **regression** and **outlier** specs, that are to be removed from the final seasonally adjusted series. Additive outliers (`final = ao`), level shift and ramp outliers (`final = ls`), temporary change (`final = tc`), and factors derived from user-defined regressors (`final = user`) can be removed. If this option is not specified, the final seasonally adjusted series will contain these effects.

mode Determines the mode of the seasonal adjustment decomposition to be performed. There are four choices: (a) multiplicative (`mode = mult`), (b) additive (`mode = add`), (c) pseudo-additive (`mode = pseudoadd`), and (d) log-additive (`mode = logadd`) decomposition. The default mode is `mode = mult`, unless the automatic transformation selection procedure is invoked in the **transform** spec; in the latter case, the mode will match the transformation selected for the series (**mult** for the log transformation and **add** for no transformation).

print and **save** Table 7.51 gives the output tables that are available by default; Table 7.52 gives other tables that can be printed or saved using this argument, while Table 7.53 shows the line printer plots that can be specified using the **print** argument.

Table 7.54 gives table names and abbreviations that can be used with the **save** argument to save certain tables as percentages rather than ratios. Specifying these table names in the **print** argument will not change the output of the program, and the percentages are only produced when multiplicative or log-additive seasonal adjustment is specified by the user in the **mode** argument; these quantities will be expressed as differences if `mode = add`.

savelog The diagnostics available for output to the log file (see Section 2.6) are listed in Table 7.55.

seasonalma Specifies which seasonal moving average (also called seasonal "filter") will be used to estimate the seasonal factors. These seasonal moving averages are $n \times m$ *moving averages*, meaning that an n-term simple average is taken of a sequence of consecutive m-term simple averages.

The seasonal filters shown in Table 7.56 can be selected for the entire series, or for a particular month or quarter. If the same moving average is used for all calendar months or quarters, only a single value needs to be provided. If different seasonal moving averages are desired for some calendar months or quarters, a list of these must be entered, specifying the desired seasonal moving average for each month or quarter. An example for a quarterly series is the following: `seasonalma = (s3x3 s3x9 s3x9 s3x9)`.

If no seasonal moving average is specified, the program will choose the final seasonal filter automatically; this option can also be invoked by setting `seasonalma = msr`. This is done using the moving seasonality ratio procedure of `X-11-ARIMA/88`, see DETAILS. This is a change from previous versions of `X-11` and `X-11-ARIMA` where, when no seasonal moving average was specified, a 3×3 moving average was used to calculate the initial seasonal factors in each iteration, and a 3×5 moving average to calculate the final seasonal factors. This seasonal filtering sequence can be specified by entering `seasonalma = x11default`.

sigmalim Specifies the lower and upper sigma limits used to downweight extreme irregular values in the internal seasonal adjustment iterations. The **sigmalim** argument has two input values – the lower and upper sigma limits. Valid list values are any real numbers greater than zero with the lower sigma limit less than the upper sigma limit (example: `sigmalim`

name	*short*	*save?*	*description of table*
adjustdiff	fad	+	final adjustment difference (only for pseudo-additive seasonal adjustment)
adjustfac	d16	+	combined seasonal and trading day factors
adjustmentratio	e18	+	final adjustment ratios (original series/seasonally adjusted series)
calendar	d18	+	combined holiday and trading day factors
calendaradjchanges	e8	+	percent changes (differences) in original series adjusted for calendar effects
combholiday	chl	+	combined holiday prior adjustment factors, A16 table
irregular	d13	+	final irregular component
irrwt	c17	+	final weights for the irregular component
movseasrat	d9a	·	moving seasonality ratios for each period
origchanges	e5	+	percent changes (differences) in original series
replacsi	d9	+	final replacement values for extreme SI-ratios (differences), D iteration
sachanges	e6	+	percent changes (differences) in seasonally adjusted series
seasadj	d11	+	final seasonally adjusted series
seasonal	d10	+	final seasonal factors
seasonaladjregsea	ars	+	seasonal factors adjusted for user-defined seasonal regARIMA component
seasonaldiff	fsd	+	final seasonal difference (only for pseudo-additive seasonal adjustment)
tdaytype	tdy	·	trading day factors printed by type of month
totaladjustment	tad	+	total adjustment factors (only printed out if the original series contains values that are ≤ 0)
trend	d12	+	final trend-cycle
trendchanges	e7	+	percent changes (differences) in final trend component series
unmodsi	d8	+	final unmodified SI-ratios (differences)
unmodsiox	d8b	+	final unmodified SI-ratios, with labels for outliers and extreme values
yrtotals	e4	·	ratio of yearly totals of original and seasonally adjusted series

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.51: **Default Output Tables for X11**

name	*short*	*save?*	*description of table*
adjoriginalc	c1	+	original series modified for outliers, trading day and prior factors, C iteration
adjoriginald	d1	+	original series modified for outliers, trading day and prior factors, D iteration
autosf	asf	·	automatic seasonal factor selection
biasfactor	bcf	+	bias correction factors
extreme	c20	+	extreme values, C iteration
extremeb	b20	+	extreme values, B iteration
ftestb1	b1f	·	F-test for stable seasonality, B1 table
ftestd8	d8f	·	F-tests for stable and moving seasonality, D8
irregularadjao	ira	+	final irregular component adjusted for point outliers
irregularb	b13	+	irregular component, B iteration
irregularc	c13	+	irregular component, C iteration
irrwtb	b17	+	preliminary weights for the irregular component
mcdmovavg	f1	+	MCD moving average of the final seasonally adjusted series
modirregular	e3	+	irregular component modified for zero-weighted extreme values
modoriginal	e1	+	original series modified for zero-weighted extreme values
modseasadj	e2	+	seasonally adjusted series modified for zero-weighted extreme values
modsic4	c4	+	modified SI-ratios (differences), C iteration
modsid4	d4	+	modified SI-ratios (differences), D iteration
qstat	f3	·	quality control statistics
replacsib4	b4	·	preliminary replacement values for extreme SI-ratios (differences), B iteration
replacsib9	b9	·	replacement values for extreme SI-ratios (differences), B iteration
replacsic9	c9	+	modified SI-ratios (differences), C iteration
residualseasf	rsf	·	F-test for residual seasonality
robustsa	e11	+	robust final seasonally adjusted series
seasadjb11	b11	+	seasonally adjusted series, B iteration
seasadjb6	b6	+	preliminary seasonally adjusted series, B iteration
seasadjc11	c11	+	seasonally adjusted series, C iteration
seasadjc6	c6	+	preliminary seasonally adjusted series, C iteration
seasadjconst	sac	+	final seasonally adjusted series with constant from **transform** spec included
seasadjd6	d6	+	preliminary seasonally adjusted series, D iteration
seasonalb10	b10	+	seasonal factors, B iteration
seasonalb5	b5	+	preliminary seasonal factors, B iteration
seasonalc10	c10	+	preliminary seasonal factors, C iteration
seasonalc5	c5	+	preliminary seasonal factors, C iteration
seasonald5	d5	+	preliminary seasonal factors, D iteration
sib3	b3	+	preliminary unmodified SI-ratios (differences)
sib8	b8	+	unmodified SI-ratios (differences)
tdadjorig	c19	+	original series adjusted for final trading day
tdadjorigb	b19	+	original series adjusted for preliminary trading day
trendadjls	tal	+	final trend-cycle adjusted for level shift outliers
trendb2	b2	+	preliminary trend-cycle, B iteration
trendb7	b7	+	preliminary trend-cycle, B iteration
trendc2	c2	+	preliminary trend-cycle, C iteration
trendc7	c7	+	preliminary trend-cycle, C iteration
trendconst	tac	+	final trend component with constant from **transform** spec included
trendd2	d2	+	preliminary trend-cycle, D iteration
trendd7	d7	+	preliminary trend-cycle, D iteration
x11diag	f2	·	summary of seasonal adjustment diagnostics

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.52: **Other Output Tables for X11**

name	*description of plot*
irregularplot	plot of the final irregular component
origwsaplot	plot of the original series with the final seasonally adjusted series
ratioplotorig	month-to-month (or quarter-to-quarter) ratio plots of the original series
ratioplotsa	month-to-month (or quarter-to-quarter) ratio plots of the seasonally adjusted series
seasadjplot	plot of the final seasonally adjusted series
seasonalplot	seasonal factor plots, grouped by month or quarter
trendplot	plot of the final trend-cycle

Name gives the name of each plot for use with the **print** argument.

Table 7.53: **Plots Specified by the `print` Argument for X11**

name	*short*	*description of table*
adjustfacpct	paf	combined adjustment factors, expressed as percentages if appropriate
calendaradjchangespct	pe8	percent changes in original series adjusted for calendar factors
irregularpct	pir	final irregular component, expressed as percentages if appropriate
origchangespct	pe5	percent changes in the original series
sachangespct	pe6	percent changes in seasonally adjusted series
seasonalpct	psf	final seasonal factors, expressed as percentages if appropriate
trendchangespct	pe7	percent changes in final trend cycle

Name gives the name of each table for use with the **save** argument.
Short gives a short name for these tables.

Table 7.54: **Tables Saved As Percentages in the `save` Argument of X11**

name	*short*	*description of diagnostic*
alldiagnostics	all	all seasonal adjustment diagnostics listed in this table
fstableb1	fb1	F-test for stable seasonality, performed on the original series
fstabled8	fd8	F-test for stable seasonality, performed on the final SI-ratios
icratio	icr	$\bar{I}/\bar{C}$ ratio
idseasonal	ids	identifiable seasonality test result
m1	m1	M1 Quality Control Statistic
m10	m10	M10 Quality Control Statistic
m11	m11	M11 Quality Control Statistic
m2	m2	M2 Quality Control Statistic
m3	m3	M3 Quality Control Statistic
m4	m4	M4 Quality Control Statistic
m5	m5	M5 Quality Control Statistic
m6	m6	M6 Quality Control Statistic
m7	m7	M7 Quality Control Statistic
m8	m8	M8 Quality Control Statistic
m9	m9	M9 Quality Control Statistic
movingseasf	msf	F-test for moving seasonality
movingseasratio	msr	moving seasonality ratio
q	q	overall index of the quality of the seasonal adjustment
q2	q2	Q statistic computed without the M2 Quality Control statistic

Name gives the name of each diagnostic for use with the **savelog** argument.
Short gives a short name for these diagnostics.

Table 7.55: **Available Log File Diagnostics for X11**

name	*description of option*
s3x1	a 3×1 moving average
s3x3	a 3×3 moving average
s3x5	a 3×5 moving average
s3x9	a 3×9 moving average
s3x15	a 3×15 moving average
stable	stable seasonal filter: a single seasonal factor for each calendar month or quarter is generated by calculating the simple average of all the values for each month or quarter (taken after detrending and outlier adjustment).
x11default	uses a 3×3 moving average to calculate the initial seasonal factors in each iteration and a 3×5 moving average to calculate the final seasonal factors
msr	uses a 3×3 moving average to calculate the initial seasonal factors in each iteration, and a 3×5 moving average to calculate the final seasonal factors in the first two iterations; the seasonal filter used for the final seasonal factors is automatically selected using a moving seasonality ratio procedure

Table 7.56: `X-13ARIMA-SEATS` **Seasonal Filters Options and Descriptions**

`= (1.8 2.8)`). A missing value defaults to 1.5 for the lower sigma limit and 2.5 for the upper sigma limit. For example, the statement `sigmalim = (, 3.0)` specifies that the upper sigma limit will be set to 3.0, while the lower sigma limit will remain at the 1.5 default. A comma is necessary if either sigma limit is missing. For an explanation of how `X-13ARIMA-SEATS` uses these sigma limits to derive adjustments for extreme values, see DETAILS.

title Title of the seasonal adjustment, in quotes, for the convenience of the user. This can be a single title or a list of up to 8 titles; an example with two titles is

```
title = ("3x9, trading day adjustment"
         "for sales of sporting goods")
```

If a list is provided, each title must be on a separate line of the spec file. This list will be printed on the title page below the series title. There is no default seasonal adjustment title.

trendma Specifies which Henderson moving average will be used to estimate the final trend-cycle. Any odd number greater than one and less than or equal to 101 can be specified. Example: `trendma = 23`. If no selection is made, the program will select a trend moving average based on statistical characteristics of the data. For monthly series, either a 9-, 13-, or 23-term Henderson moving average will be selected. For quarterly series, the program will choose either a 5- or a 7-term Henderson moving average.

type When `type = summary`, the program develops estimates of the trend-cycle, irregular, and related diagnostics, along with residual seasonal factors and, optionally, residual trading day and holiday factors from an input series that is assumed to be either already seasonally adjusted or nonseasonal. These residual factors are not removed. The output series in the final seasonally adjusted series (table D11) will be the same as the original series (table A1).

When `type = trend`, the program develops estimates for the final trend-cycle and irregular components without attempting to estimate a seasonal component. The input series is assumed to be either already seasonally adjusted or nonseasonal. With this option, estimated trading day and holiday effects as well as permanent prior adjustment factors are removed to form the adjusted series (table D11) and to calculate the trend (table D12). When a metafile with a **composite** spec is used to obtain an indirect adjustment of an aggregate, these options are used for components of the aggregate that are not seasonally adjusted.

In the default setting, `type = sa` – the program calculates a seasonal decomposition of the series. With all three values of **type**, the final seasonally adjusted series (printed in the D11 table of the main output file) is used to form the indirect seasonal adjustment of the composite.

RARELY USED ARGUMENTS

calendarsigma Specifies if the standard errors used for extreme value detection and adjustment are computed separately for each calendar month (quarter), or separately for two complementary sets of calendar months (quarters). If `calendarsigma = all`, the standard errors will be computed separately for each month (quarter). If `calendarsigma = signif`, the standard errors will be computed separately for each month only if Cochran's hypothesis test determines that the irregular component is heteroskedastic by calendar month (quarter). If `calendarsigma = select`, the months (quarters) will be divided into two groups, and the standard error of each group will be computed. For the **select** option, the argument **sigmavec** must be used to define one of the two groups of months (quarters). If **calendarsigma** is not specified, the standard errors will be computed from 5 year spans of irregulars, in the manner described in Dagum (1988).

centerseasonal If `centerseasonal = yes`, the program will center the seasonal factors combined with user-defined seasonal regression effects. The default is `centerseasonal = no`.

excludefcst Specifies whether the `X-11` extreme value procedure is used on the forecasts and backcasts used to extend the series prior to seasonal adjustment. If `excludefcst = yes`, the extreme value procedure is used on the forecasts and backcasts. If `excludefcst = no`, extreme values are not identified and adjusted for in the forecasts and backcasts. The default is `excludefcst = yes`.

keepholiday Determines if holiday effects estimated by the program are to be kept in the final seasonally adjusted series. In the default setting, `keepholiday = no`, holiday adjustment factors derived from the program are removed from the final seasonally adjusted series. If `keepholiday = yes`, holiday adjustment factors derived from the program are kept in the final seasonally adjusted series. The default is used to produce a series adjusted for both seasonal and holiday effects.

print1stpass If `print1stpass = yes`, output from the seasonal adjustment needed to generate the irregular components used for the irregular regression adjustment procedures will be printed out. If `print1stpass = no`, this output will be suppressed, and only the tables associated with the irregular regression procedures are printed out. The default is `print1stpass = no`. When `print1stpass = yes`, the **print** argument controls which tables are actually printed.

sfshort Controls what seasonal filters are used to obtain the seasonal factors if the series is at most 5 years long. For the default case, `sfshort = no`, a stable seasonal filter will be used to calculate the seasonal factors, regardless of what is entered for the **seasonalma** argument. If `sfshort = yes`, `X-13ARIMA-SEATS` will use the central and one-sided seasonal filters associated with the choice given in the **seasonalma** argument wherever possible.

sigmavec Specifies one of the two groups of months (quarters) for whose irregulars a group standard error will be calculated under the `calendarsigma = select` option. The user enters the month(s) (either the full name of the month or the abbreviations shown in Section 3.3) or quarter(s) (**q1** for the first quarter, **q2** for the second quarter, etc.) that comprise one group; all remaining months or quarters comprise the second group. Example: `sigmavec=(jan feb dec)`. **Warning:** This argument can only be specified when

`calendarsigma = select`.

trendic Specifies the irregular-to-trend variance ratio that will be used to generate the end weights for the Henderson moving average. The procedure is taken from Doherty (2001). If this variable is not specified, the value of **trendic** will depend on the length of the Henderson trend filter. These default values closely reproduce the end weights for the set of Henderson trend filters that originally appeared in `X-11` and `X-11-ARIMA`.

true7term Specifies the end weights used for the 7 term Henderson filter. If `true7term = yes`, then the asymmetric ends weights for the 7 term Henderson filter are applied for observations at the end of the series where a central Henderson filter cannot be applied. If `true7term = no`, then central and asymmetric weights from a 5 term Henderson filter are applied, as in previous versions of the `X-11-ARIMA` program released by Statistics Canada. The default is `true7term = no`.

DETAILS

Modes of seasonal adjustment: In any `X-13ARIMA-SEATS` seasonal adjustment, the original time series (O) is decomposed into three basic components:

Trend-Cycle (C): The long-term and medium-to-long term movements of the series, including consequential turning points.

Seasonal (S): Within-year fluctuations about the trend that recur in a very similar way in the same month or quarter from year to year.

Irregular (I): The residual component that remains after seasonal and trend (and any trading day and holiday effects that have been identified) are removed from the series. It is characterized by movements of very short duration. These can be quite large if there are strikes or other unusual economic events of short duration.

Depending mainly on the nature of the seasonal movements of a given series, several different models are used to describe the way in which the components C, S, and I combine to form the original series O. `X-13-ARIMA-SEATS` provides modes of seasonal adjustment appropriate for four different decomposition models. Table 7.57 gives the four values of the **mode** arguments and the corresponding models, both for the original (O) and seasonally adjusted series (SA).

Entry for **mode**	*Name for* **mode**	*Model for O*	*Model for SA*
`mult`	Multiplicative	$O = C \times S \times I$	$SA = C \times I$
`add`	Additive	$O = C + S + I$	$SA = C + I$
`pseudoadd`	Pseudo-Additive	$O = C \times (S + I - 1)$	$SA = C \times I$
`logadd`	Log-Additive	$\log(O) = C + S + I$	$SA = exp(C + I)$

Table 7.57: **Modes of Seasonal Adjustment and Corresponding Models**

The default seasonal adjustment mode is multiplicative. This is because, for most seasonal economic time series, the magnitudes of the seasonal fluctuations appear to increase and decrease proportionally with increases

and decreases in the level of the series, in a way that is proportional to the level. A series with this type of seasonality is said to have *multiplicative seasonality*. To estimate the multiplicative components, the program uses a ratio-to-moving average method whose details are given in Shiskin, Young, and Musgrave (1967), Dagum (1988), Baxter (1994), and Ladiray and Quenneville (2001), among others. The pseudo-additive model is considered when some months (or quarters) have extremely small values (due to vacations or climate, for example), and the remaining months appear to have multiplicative seasonality. If the magnitude of the seasonal does not appear to be affected by the level of the series, then the series has *additive seasonality*, and the additive mode is appropriate.

The log-additive mode gives an alternative multiplicative decomposition which can be useful for certain econometric analysis, usually related to time series model considerations. For log-additive seasonal adjustment, the trend component is computed from an additive decomposition of the logged series ($\log(O)$), so the additive trend must be exponentiated in order to derive a trend with the same units as the original series. This results in a downwardly biased estimate of trend; this bias is adjusted in `X-13ARIMA-SEATS` using a bias-correction procedure described in Thomson and Ozaki (2002).

For multiplicative, pseudo-additive, and log-additive seasonal adjustment, the seasonal and irregular components are assumed to be ratios centered about 1. In the main output they are expressed as percentages so that they center about 100. For additive seasonal adjustment, the seasonal and irregular components are in the same units as the original time series and vary about 0.

When a regARIMA model is specified with the **regression** and **arima** specs, trading day, holiday, outlier, and other regression effects defined in the **regression** spec can be derived from the regression coefficients of a regARIMA model and used to adjust the original series prior to seasonal adjustment. In this case, these effects must be the same type as factors generated by the seasonal adjustment procedure, so that combined adjustment factors can be derived and adjustment diagnostics generated. If the regARIMA model is fit to a log-transformed series, the regression factors are expressed in the form of ratios, which is the same form as factors generated by the multiplicative or log-additive adjustment modes. Conversely, if the regARIMA model is fit to the original series, the regression factors are measured on the same scale as the original series, which matches the scale of the components generated by the additive adjustment mode. Therefore, users should be careful to ensure that the transformation specified by the **function** or **power** arguments of the **transform** spec is compatible with the seasonal adjustment mode specified by the **mode** argument of the **x11** spec. Furthermore, be aware that the default value for the **mode** argument is multiplicative seasonal adjustment, which conflicts with the default for the **function** and **power** arguments of the **transform** spec, which assume no transformation is done. Currently, you cannot use regression effects to pre-adjust the original series for a pseudo-additive seasonal adjustment.

Multiplicative and pseudo-additive seasonal adjustment give very similar results for most series with multiplicative seasonality, unless the seasonal amplitude of the series is large. If the smallest seasonal factor is 0.7 or less, there will be noticeable differences between the multiplicative and pseudo-additive seasonal adjustments. If the smallest seasonal factor is 0.5 or less, this difference is likely to be important. If a multiplicative seasonal adjustment produces extreme values (meaning values less than 100.0 in Table C17 and especially 0 values) that occur more frequently in months (or quarters) with small seasonal factors than in months with large seasonal factors, then pseudo-additive seasonal adjustment is likely to be better. For more details on when to use pseudo-additive seasonal adjustment, see Baxter (1994).

For simplicity, this discussion has ignored trading day and holiday effects. When these are estimated, they add additional factors to the decomposition and, depending on how they are defined, adjustment for them can

lead to larger differences between the annual totals of the adjusted series and the annual totals of the original time series.

Downweighting of extreme irregulars: Let μ_I be the assumed mean of the irregular component (1.0 for multiplicative seasonal adjustment, 0.0 for additive). Let σ_{X11} denote an estimate of the standard deviation of the irregular component for a month or quarter. If the absolute value of $I_t - \mu_I$ is less than the lower sigma limit multiplied by σ_{X11}, the irregular value I_t receives full weight. If the absolute value of $I_t - \mu_I$ is more than the upper sigma limit multiplied by σ_{X11}, the irregular value receives zero weight, meaning that I_t is replaced by μ_t for seasonal factor calculations. Otherwise, I_t is partially downweighted.

Automatic seasonal filter selection: This procedure is taken from `X-11-ARIMA/88`, see Dagum (1988). For the first two seasonal adjustment iterations, a 3×3 moving average is used to calculate the initial seasonal factors and a 3×5 moving average is used to calculate the final seasonal factor. In the third and final iteration, a 3×3 moving average is used to calculate the initial seasonal factors, but for the final iteration the program calculates the moving seasonality ratio ($\bar{I}/\bar{S}$, also called the global MSR). Then the program chooses whether to use a 3×3, 3×5, or 3×9 moving average based on the size of the global MSR. For more information on the moving seasonality ratio, see Lothian (1984).

Forecast extension: As mentioned in the introduction, an important use of regARIMA models is to extend the series with forecasts (and backcasts) to improve the seasonal adjustment of the most recent (and the earliest) observations. Therefore, `X-13ARIMA-SEATS` will extend the series with one year of forecasts prior to seasonal adjustment whenever a regARIMA model is specified with no **forecast** spec. To specify a seasonal adjustment without forecast extension, set `maxlead = 0` in the **forecast** spec.

Level shifts and the final Henderson trend: When level shifts are estimated and removed from the series prior to seasonal adjustment, they are put back into the final Henderson trend cycle (Table D12), so that this component will have the level of the observed data. A table of the trend cycle of the level shift adjusted time series can also be obtained by setting `print = trendadjls`.

Easter adjustment: The Easter adjustment options in this spec cannot be used when regARIMA model based holiday are specified in the **regression** spec, or if an Easter adjustment is specified within the **x11regression** spec.

Table of SI values with labels for extreme values: Table D8.B is designed to provide users with direct information about which of the unmodified Seasonal-Irregular values (the detrended series, henceforth called SI values) produced by the `X-11` seasonal adjustment program will be modified by extreme value adjustment (as shown by the irregular weights in Table C17) or are likely to have been affected by regARIMA outliers (either those specified by the user or those identified through the **outlier** spec).

Each SI value that has been identified as an `X-11` extreme value is printed with a "*" next to it. SI values at times at which a single regARIMA outlier occurs in the model are printed with a "#" next to them. Extreme SI values at times associated with at least one regARIMA outlier are printed with a "&" next to them; SIs at times with more than one regARIMA outlier will have a "@" next to them. All observations between (and including) the starting and ending points of a ramp outlier are marked as if they were outliers.

With multiplicative seasonal adjustments, SI values before and after level shift outliers that are most likely to have been affected by the level shift are marked with a "−" character next to the value. The number of observations flagged in this way depends on the magnitude of the level shift outlier (as determined by its regression coefficient estimate) and on the length of the Henderson filter used for the trend that generates the SI-ratios, in the manner described in Table 7.58.

Percent Change in Level (Δ_L)	**Length of Henderson Filter**				
	23	13	9	7	5
$\Delta_L \leq 1.1$	0	0	0	0	0
$1.1 < \Delta_L \leq 1.2$	1	1	0	0	0
$1.2 < \Delta_L \leq 1.3$	1	1	1	0	0
$1.3 < \Delta_L \leq 1.5$	2	1	1	0	0
$1.5 < \Delta_L \leq 1.8$	2	1	1	1	0
$1.8 < \Delta_L \leq 1.9$	2	2	1	1	0
$1.9 < \Delta_L \leq 2.0$	3	2	1	1	0
$2.0 < \Delta_L \leq 2.6$	3	2	1	1	1
$2.6 < \Delta_L \leq 2.9$	3	2	2	1	1
$2.9 < \Delta_L \leq 3.6$	4	2	2	1	1
$3.6 < \Delta_L \leq 5.5$	4	3	2	1	1
$5.5 < \Delta_L$	5	3	2	1	1

Table 7.58: **Number of Surrounding SI-ratios in Table D 8.B Assumed Affected by a Level Shift**

Treatment of nonseasonal series: A nonseasonal series can be decomposed into trend-cycle and irregular components using the `type = trend` option. This decomposition is obtained by a simplification of the `X-11` seasonal adjustment decomposition procedure that retains only the steps related to the Henderson trends and extreme value detection. Example 7 below shows how the `type = trend` option can also be applied to a seasonally adjusted series to obtain an alternative trend used by several national statistical offices in place of the final Henderson trend (D12) for a seasonal time series. The alternative is a slight update of the trend proposed by Dagum (1996).

EXAMPLES

Example 1 Multiplicative seasonal adjustment with all default options (so the program uses the moving seasonality ratio to select the seasonal filter length). The monthly series starts in January 2006 and is stored in free format in the file `klaatu.dat` in the current directory.

```
Series { File="klaatu.dat" Start = 2006.1 }
X11 {  }
```

Example 2 Multiplicative monthly seasonal adjustment, 3×9 seasonal filters for all months, 23-term Henderson moving average for the trend-cycle. Perform a test (using a version of AIC that adjusts for the length of the series) of the significance of the trading-day regressors in a regression of the irregular component.

```
Series { File="klaatu.dat" Start = 2006.1 }
X11 { SeasonalMA =  s3x9  TrendMA = 23  }
X11regression { variables = td   aictest=td  }
```

Example 3 Quarterly seasonal adjustment, 3×3 seasonal filters for the first two quarters, 3×5 seasonal filters for the other two quarters, 7-term Henderson trend moving average.

```
series {
        file="qhstarts.dat"
        start = 1967.1   period=4
       }
x11    {
        seasonalma = (s3x3 s3x3 s3x5 s3x5)
        trendma = 7
       }
```

Example 4 Seasonal ARIMA model with regression effects is used to obtain preadjustments of monthly data. No forecast extension will be done in this example. Specified regression variables are a constant, trading day effects, and two level shifts, one in May 2002 and one in September 2006. The ARIMA part of the model is $(0,1,2)(1,1,0)_{12}$. Additively seasonally adjust the series after preadjusting for the outlier, level-shift and trading day effects estimated using the regARIMA model. Use sigma limits set to 2.0 and 3.5 to search for extreme values in the irregular component of the seasonal decomposition. Use the `alltables` print level when printing out seasonal adjustment output.

```
SERIES { TITLE = "EXPORTS OF LEATHER GOODS"  START = 1999.JUL
   DATA = (815 866 926 ... 942)  }
REGRESSION { VARIABLES = (CONST TD LS2002.MAY LS2006.OCT)  }
ARIMA {  MODEL=(0 1 2)(1 1 0)   }
ESTIMATE {  }
FORECAST {  MAXLEAD=0   }
X11 {  MODE = ADD  PRINT = ALLTABLES  SIGMALIM = (2.0 3.5)  }
```

Example 5 The predefined regression effects used are trading day variables and a constant. User-defined regression variables are included to capture the effect of special sales promotions in 2008 and 2010. These variables are read from the file `special.dat`. The ARIMA part of the model is $(3,1,0)(0,1,1)_{12}$. The seasonal period, 12, is not specified since this is the default. Perform a multiplicative seasonal adjustment on the series after pre-adjusting for the regARIMA trading day and user-defined regression effects and extending the series with 12 forecasts and 12 backcasts. A two-line list of seasonal adjustment titles is specified.

```
series {  title = "Unit Auto Sales"  file = "autosal.dat"
         start = 2005.1  }
transform {  function = log  }
regression  { variables = (const td)~~~user = (sale08 sale10)
             file = "special.dat"     format = "(2f12.2)"  }
arima  { model = (3 1 0)(0 1 1)12  }
forecast  { maxlead=12    maxback=12  }
x11  {  title = ( "Unit Auto Sales"
                  "Adjusted for special sales in 2008, 2010" )
        }
```

Example 6 Read in the data from a file using a predefined `X-11` data format. Note that the starting date will be taken from the information provided in the data file and so does not have to be specified. Specify a regARIMA model for the log transformed data with certain outlier terms. Use this model to generate 5 years of forecasts. Perform a multiplicative seasonal adjustment using a 3×9 seasonal moving average for all months. Save the E2 table (Table D11 with D12 trend values substituted when the C17 value is zero) for use in the next example.

```
series { title="NORTHEAST ONE FAMILY Housing Starts"
         file="cne1hs.ori"   name="CNE1HS"    format="2R" }
transform {    function=log   }
regression {
    variables=(ao1996.feb ao1998.feb ls2000.feb
               ls2002.nov ao2004.feb)
}
arima {  model=(0 1 2)(0 1 1)  }
forecast {  maxlead=60  }
x11 {  seasonalma=(s3x9)
       title="Adjustment of 1 family housing starts"
       save = e2
}
```

Example 7 A continuation of Example 6. Use the `type = trend` option to obtain an alternative to D12 trend for seasonal time series proposed by Dagum (1996) and further evaluated and updated in Dagum and Luati (2009) and Darne and Dagum (2009). Read in the data from the seasonally adjusted series modified for outliers (AO and TC) and extreme values from the E2 file saved in Example 6. The starting date will be taken from the information provided in the E2 file and so does not have to be specified. The nonseasonal (0 1 1) model with be used to provide the six forecasts of the E2 series so that the symmetric 13 term Henderson filter can be applied. The AO outliers are commented out in the **regression** spec as these outliers have already been removed from the E2 table, and therefore do not have to be accounted for in the model. The default setting of forecast provides 12 forecasts, so it can be used. Note the very low values of the sigma limits used to smooth the E2 values. Note also that the log transformation is not used.

```
series {
  title="Trend for NORTHEAST ONE FAMILY Housing Starts"
  file="cne1hs.e2"
  format="x13save"
  }
regression {
  variables=(
#     ao1996.feb     ao1998.feb     ao2004.feb
     ls2010.feb     ls2012.nov
  )
}
```

```
arima { model=(0 1 1) }
forecast { }
x11{
   type=trend        trendma=13
   sigmalim=(0.7, 1.0)
   title="Updated Dagum (1996) trend of 1 family housing starts"
}
```

Example 8 The predefined regression effect is a constant. The user-defined regression variables are for strikes in 2002, 2005, 2010 and are located in the file `strikes.dat`. The ARIMA part of the model is $(0,1,1)(0,1,1)_{12}$. Since a model is specified in the spec, generate a year of forecasts by default. Seasonally adjust the series after pre-adjusting for the user-defined regression effects. Adjust the series for `X-11` trading day before estimating the regARIMA model.

```
series{ title="Automobile Sales"
        file = "carsales.dat"
        start = 2001.1 }
transform{ function = log }
regression{ variables = ( const )
            user = (strike02 strike05 strike10)
            file = "strike.dat"  format = "(3f12.0)"
            }
arima{ model = (0 1 1)(0 1 1)12 }
x11{  title = ("Car Sales in US - Adjust for strikes in 2002, 2005, 2010")
      save=seasonal appendfcst=yes
   }
x11regression { variables = td }
```

Example 9 Use the automatic transformation selection procedure to determine if a log transformation should be used to transform the series. Since a regARIMA model is not specified, the program will use an airline model to generate the AICC values needed for the test. The AICC difference for the test has been reset to zero, so the program will choose the transformation based on which model estimation yields the smaller value of AICC. The choice of transformation will determine if the seasonal adjustment will be a multiplicative or an additive seasonal adjustment.

```
series {title = "Total U.K. Retail Sales"
        file = "ukretail.dat"
        start = 1998.jan
        }
transform {function = auto
           aicdiff = 0.0
          }
x11 {save=d11
    }
```

7.20 X11REGRESSION

DESCRIPTION

An optional spec for use in conjunction with the **x11** spec for series without missing observations. This spec estimates calendar effects by regression modeling of the irregular component with predefined or user-defined regressors. The user can select predefined regression variables with the **variables** argument. The predefined variables are for calendar (trading-day and holiday) variation and additive outliers. A change of regime option is available with trading-day regressors. User-defined calendar effect regression variables can be included in the model via the **user** argument. Data for any user-defined variables must be supplied, either in the **data** argument or in a file named in the **file** argument (but not both). The regression model specified can contain both predefined and user-defined regression variables.

USAGE

```
x11regression {   variables = (td or td1coef or
                                   tdstock[31] or tdstock1coef[31]
                               easter[8] labor[8]
                               thank[1]
                               ao2007.apr )
                  user = (temperature precip)
                  start = 1985.jan
                      data = (25 0.1 ...)
                   or
                      file = "weather.dat"
                      format = "(2f5.1)"
                  tdprior = ( 1.4 1.4 1.4 1.4 1.4
                              0.0 0.0 )
                  aictest = ( easter user
                              td or td1coef or tdstock or tdstock1coef )
                  aicdiff = -2.0
                  span = (2000.jan, 2015.dec)
                      sigma = 2.75
                  or
                      critical = 3.5
                      outliermethod = addone
                      outlierspan = (2000.may, 2016.sep)
                  usertype = holiday
                  prior = yes
                  print = (brief +b15)
                  save = (c16 c18)
                  savelog = aictest
              }
```

ARGUMENTS

aicdiff Defines the difference in AICC needed to accept a regressor specified in the **aictest** argument. The default value is `aicdiff = 0.0`. For more information on how this option is used in conjunction with the **aictest** argument, see DETAILS.

aictest Specifies that an AIC-based comparison will be used to determine if a specified regression variable should be included in the user's irregular component regression model. The only entries allowed for this variable are **td**, **tdstock**, **td1coef**, **tdstock1coef**, **easter**, and **user**. If a trading day model selection is specified, for example, then AIC values (with a correction for the length of the series, henceforth referred to as AICC) are derived for models with and without the specified trading day variable. By default, the model with smaller AICC is used to generate forecasts, identify outliers, etc. If more than one type of regressor is specified, the AIC-tests are performed sequentially in this order: (a) trading day regressors, (b) easter regressors, (c) user-defined regressors. If there are several variables of the same type (for example, several td regressors), then the **aictest** procedure is applied to them as a group. That is, either all variables of this type will be included in the final model or none. See DETAILS for more information on the testing procedure. If this option is not specified, no automatic AIC-based selection will be performed.

critical Sets the critical value (threshold) against which the absolute values of the outlier t-statistics are compared to detect additive outliers (meaning extreme irregular values). This argument applies unless the **sigma** argument is used, or the only regressor(s) estimated is flow trading day. The assigned value must be a real number greater than 0. Example: `critical = 4.0`. The default critical value is determined by the number of observations in the interval searched for outliers (see the **outlierspan** argument below). Table 7.22 gives default critical values for a number of outlier span lengths. Larger (smaller) critical values predispose **x11regression** to treat fewer (more) irregulars as outliers. A large value of **critical** should be used if no protection is wanted against extreme irregular values.

data Assigns values to the user-defined regression variables. The time frame of the values must cover the time frame of the series (or of the span specified by the **span** argument of the **series** spec, if present). It must also cover the time frame of forecasts and backcasts requested in the **forecast** spec.[26] The data values are read in free format. The numerical values given in this argument should be in the order in which the user-defined variables are named in the **user** argument. This assignment should proceed through all the values of the user-defined variables for the first time point, then through all the values for the second time point, etc. If the **data** argument is used, the **file** argument cannot be used.

file Name of the file containing data values for *all* user-defined regression variables. The filename must be enclosed in quotes. If the file is not in the current directory, the path must also be given. As with the **data** argument, the time frame of the data values must cover both the series and any forecasts and backcasts.[27] If the **file** argument is used, the

[26] See the end of Section 4.7 for a discussion of what to do about forecast extension for seasonal adjustment of a series with a model that contains user-defined regressors whose future values are unknown.

[27] See previous footnote.

data argument cannot be used.

format Indicates the format used when reading the values for the regression variables in the file named in the **file** argument. Six types of input are accepted:

a. free format, in which all numbers on a line will be read before continuing to the next line, and the numbers must be separated by one or more spaces (not by commas or tabs) (example: `format = "free"`);

b. a valid Fortran format, which must be enclosed in quotes and must include the initial and terminal parentheses (example: `format = "(6f12.0)"`);

c. "datevalue" format, in which the year, month or quarter, and the associated values for each of the user-defined regression variables for a given observation are given, in this order, in free format on individual lines in the data file. Thus, a line of the data file with three regressors having the values 0, 0, and 1, respectively, for July 1991 would have the form `1991 7 0 0 1`. All the user-defined regressors must be on the same record, and in the order of their appearance in the **user** argument (example: `format = "datevalue"`);

d. the "x13save" format `X-13ARIMA-SEATS` uses to save a table. This allows the user to read in a file saved from a previous `X-13ARIMA-SEATS` run (example: `format = "x13save"`).[28]

e. a variant of "free" format where the numbers must be separated by one or more spaces (not by commas or tabs), and decimal points are expressed as commas (a convention in some European countries). (example: `format = "freecomma"`);

f. a variant of "datevalue" format, where the year, month or quarter, and value of each observation are found in this order in free format on individual lines, where decimal points are expressed as commas. Thus, a line of the data file with three regressors having the values 0.5, 0, and 1.25, respectively, for July 1991 would have the form `1991 7 0,5 0 1,25`. All the user-defined regressors must be on the same record, and in the order of their appearance in the **user** argument (example: `format = "datevaluecomma"`).

If no **format** argument is given the data will be read in free format. In free format, all numbers on a line will be read before continuing to the next line, and the numbers must be separated by one or more spaces (not by commas or tabs). The **format** argument cannot be used with the **data** argument, only with **file**.

outliermethod Determines how the program successively adds detected outliers to the model. The choices are `method = addone` or `method = addall`. See the DETAILS section of the **outlier** spec for a description of these two methods. The default is `method = addone`. This argument cannot be used if the **sigma** argument is used.

outlierspan Specifies start and end dates of the span of the irregular component to be searched for outliers. The start and end dates of the span must both lie within the series, and the start date must precede the end date. A missing value, e.g., `outlierspan = (2006.jan,)`,

[28] Note that to maintain compatibility with previous versions of `X-12-ARIMA` the entry **x13save** will also be accepted.

defaults to the start date or end date of the series, as appropriate. (If there is a **span** argument in the **series** spec, then, in the above remarks, replace the start and end dates of the series by the start and end dates of the span given in the **series** spec.) This argument cannot be used with the **sigma** argument.

print and **save** The default output tables available for the direct and indirect seasonal adjustments generated by this spec are given in Table 7.59; other output tables available are given in Table 7.60. For a complete listing of the `brief` and `default` print levels for this spec, see Appendix B.

name	*short*	*save?*	*description of table*
priortd	a4	+	prior trading day weights and factors
extremeval	c14	+	irregulars excluded from the irregular regression, C iteration
x11reg	c15	+	final irregular regression coefficients and diagnostics
tradingday	c16	+	final trading day factors and weights
combtradingday	c18	+	final trading day factors from combined daily weights
holiday	xhl	+	final holiday factors
calendar	xca	+	final calendar factors (trading day and holiday)
combcalendar	xcc	+	final calendar factors from combined daily weights
outlierhdr	xoh	·	options specified for outlier detection including critical value and outlier span
xaictest	xat	·	output from AIC-based tests for trading day and holiday

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.59: **Default Output Tables for X11regression**

prior Specifies whether calendar factors from the irregular component regression are computed in a preliminary run and applied as prior factors (`prior = yes`) or as a part of the seasonal adjustment process (`prior = no`). The default is `prior = no`. The **prior** argument has no effect when a regARIMA model is specified; in this case, the irregular component regression is always computed before seasonal adjustment.

savelog Setting `savelog = aictest` or `savelog = ats` causes the results of the AIC-based selection procedure specified by the **aictest** argument to be output to the log file (see Section 2.6 for more information on the log file).

name	*short*	*save?*	*description of table*
extremevalb	b14	+	irregulars excluded from the irregular regression, B iteration
x11regb	b15	+	preliminary irregular regression coefficients and diagnostics
tradingdayb	b16	+	preliminary trading day factors and weights
combtradingdayb	b18	+	preliminary trading day factors from combined daily weights
holidayb	bxh	+	preliminary holiday factors
calendarb	bxc	+	preliminary calendar factors
combcalendarb	bcc	+	preliminary calendar factors from combined daily weights
outlieriter	xoi	+	detailed results for each iteration of outlier detection including outliers detected, outliers deleted, model parameter estimates, and robust and non-robust estimates of the residual standard deviation
outliertests	xot	·	t-statistics for every time point of each outlier detection iteration
xregressionmatrix	xrm	+	values of irregular regression variables with associated dates
xregressioncmatrix	xrc	+	correlation matrix of irregular regression parameter estimates if used with the **print** argument; covariance matrix of same if used with the **save** argument

Name gives the name of each table for use with the **print** and **save** arguments.
Short gives a short name for these tables.
Save? indicates which tables can be saved (+) or not saved (·) into a separate file with the **save** argument.

Table 7.60: **Other Output Tables for X11regression**

sigma The sigma limit for excluding extreme values of the irregular components before trading day (only) regression is performed. Irregular values larger than this number of standard deviations from the mean (1.0 for multiplicative adjustments, 0.0 for additive adjustments) are excluded as extreme. Each irregular has a standard error determined by its month (or quarter) type. The month types are determined by the month length and by the day of the week on which the month starts. This argument cannot be used when regressors other than flow trading day are present in the model, or when the **critical** argument is used. The assigned value must be a real number greater than 0; the default is 2.5 (which is invoked only when the flow trading day variable(s) are the only regressor estimated). Example: `sigma = 3.0`.

span Specifies the span (data interval) of irregular component values to be used to estimate the regression model's coefficients. This argument can be utilized when, for example, the user does not want data early in the series to affect regression estimates used for preadjustment before seasonal adjustment. As with the **modelspan** spec detailed in the **series** spec, the **span** argument has two values, the start and end date of the desired span. A missing value defaults to the corresponding start or end date of the span of the input series. For example, for monthly data, the statement `span = (2008.1,)` causes whatever irregular regression model is specified to be estimated from the time series data starting in January 2008 and ending at the end date of the analysis span. A comma is necessary if either the start or the end date is missing. The start and end dates of the model span must both lie within the time span of data specified for analysis in the **series** spec, and the start date must precede the end date.

Another end date specification, with the form *0.per*, is available to set the ending date of **span** to always be the most recent occurrence of the specific calendar month (quarter for quarterly data) *per* in the span of data being analyzed. Thus, if the span of data considered ends in a month other than December, `span = (,0.dec)` will cause the regression coefficients to stay fixed at the values obtained from data ending in December of the next-to-final calendar year of the span.

start The start date for the values of the user-defined regression variables. The default is the start date of the series. Valid values are any date up to the start date of the series (or up to the start date of the span specified by the **span** argument of the **series** spec, if present).

tdprior User-input list of seven daily weights, starting with Monday's weight, which specify a desired `X-11` trading day adjustment prior to seasonal adjustment. These weights are adjusted to sum to 7.0 by the program. This option can be used only with multiplicative and log-additive seasonal adjustments. The values must be real numbers greater than or equal to zero. Example: `tdprior = (0.7 0.7 0.7 1.05 1.4 1.4 1.05)`.

user Specifies the list of names of user-defined regression variables. A name is required for each user-defined variable whose coefficients are to be estimated. The names given are used to label estimated coefficients in the program's output. Values for the user-defined variables must be supplied, using either the **data** or the **file** argument (but not both). The maximum number of user-defined regression variables is 52. (This limit can be changed—see Section 2.8.)

usertype Assigns a type to the user-defined regression variables. The user-defined regression effects can be defined as a trading day (**td**), holiday (**holiday**), or other user-defined (**user**) regression effects. A single effect type can be specified for all the user-defined regression variables defined in the **x11regression** spec (`usertype = td`), or each user-defined regression variable can be given its own type (`usertype = (td td td td td td holiday user)`). See DETAILS for more information on assigning types to user-defined regressors.

variables List of predefined regression variables to be included in the model. The values of these variables are calculated by the program, as functions of the calendar in most cases. See DETAILS for a discussion and a table of the available predefined variables.

RARELY USED ARGUMENTS

almost Differential used to determine the critical value used for a set of "almost" outliers – outliers with t-statistics near the outlier critical value that are not incorporated into the regARIMA model. After outlier identification, any outlier with a t-statistic larger than **critical** − **almost** is considered an "almost outlier," and is included in a separate table. The default is `almost = 0.5`; values for this argument must always be greater than zero.

b Specifies values for irregular component regression parameters in the order in which they appear in the **variables** and **user** arguments. Values may be specified for some or all of the regression coefficients. Values followed immediately by an 'f' will be held fixed in the model estimation; all other coefficients will be estimated in the OLS regression done for the model fitting. Thus, the sole reason for specifying any values of **b** is to hold those regression coefficients fixed when the model is fitted. E.g., if one specifies `b = (0.3,,0.7f)`, this is equivalent to specifying `b = (,,0.7f)` – the first and second coefficients will be estimated by OLS regression (so specifying the 0.3 is unnecessary), while the third coefficient is fixed at 0.7.

centeruser Specifies the removal of the (sample) mean or the seasonal means from the user-defined regression variables. If `centeruser = mean`, the mean of each user-defined regressor is subtracted from the regressor. If `centeruser = seasonal`, means for each calendar month (or quarter) are subtracted from each of the user-defined regressors. If this option is not specified, the user-defined regressors are assumed to already be in an appropriately centered form and are not modified.

eastermeans Specifies whether the means used to remove seasonality from the Easter regressor associated with the variable `easter[w]` are the long term (500 year) monthly means, as described in footnote 8 of Table 4.1 (`eastermeans = yes`), or the monthly means calculated from just the span of data used for calculating the coefficients of the Easter regressor (`eastermeans = no`). The default is `eastermeans = yes`. This argument is ignored if no built-in Easter regressor is included in the regression model, or if the only Easter regressor is `sceaster[w]` (see DETAILS).

forcecal Specifies whether the calendar adjustment factors are to be constrained to have the same value as the product (or sum, if additive seasonal adjustment is used) of the holiday and trading day factors (`forcecal = yes`), or not (`forcecal = no`). The default is `forcecal = no`. This argument is functional only when both holiday and trading day regressors are specified in the **variables** argument of this spec.

noapply List of the types of regression effects defined in the **x11regression** spec whose model-estimated values are not to be adjusted out of the original series or final seasonally adjusted series. Available effects include modeled trading day effects (**td**) and Easter, Labor Day, and Thanksgiving-Christmas holiday effects (**holiday**).

reweight Specifies whether the daily trading day weights are to be re-weighted when at least one of the daily weights in the C16 output table is less than zero (`reweight = yes`), or not (`reweight = no`). The default is `reweight = no`. This argument is functional only when trading day regressors are specified in the **variables** argument of this spec. **Note:** the default for previous versions of `X-11` and `X-11-ARIMA` corresponds to `reweight = yes`.

umdata An input array of mean-adjustment values, to be subtracted from the irregular series I_t (or $\log(I_t)$) before the coefficients of a model with a user-defined regressor are estimated. This argument, or **umfile**, is used when the mean function for predefined regressors described in DETAILS is incorrect for the model with user-defined regressors. The mean-adjustment function depends on the mode of adjustment. See DETAILS for more information.

The time frame of these values must cover the time frame of the series (or of the span specified by the **span** argument of the **series** spec, if present). It must also cover the time frame of forecasts and backcasts requested in the **forecast** spec. The data values are read in free format. If the **umdata** argument is used, the **umfile** argument cannot be used.

umfile Name of the file containing a series of mean-adjustment values to be subtracted from the irregular series I_t (or $\log(I_t)$) before the coefficients of a model with a user-defined regressor are estimated. This replaces the mean function that is subtracted from I_t when only predefined regressors are used, as described in DETAILS. The filename must be enclosed in quotes. If the file is not in the current directory, the path must also be given. As with the **umdata** argument, the time frame of the data values must cover both the series and any forecasts or backcasts. If the **file** argument is used, the **umdata** argument cannot be used.

umformat Denotes the format used when reading the data for the mean-adjustment values from the file named in the **umfile** argument. Eight types of input are accepted:

a. free format, in which all numbers on a line will be read before continuing to the next line, and the numbers must be separated by one or more spaces (not by commas or tabs) (example: `format = "free"`);

b. a valid Fortran format, which must be enclosed in quotes and must include the initial and terminal parentheses (`umformat = "(6f12.0)"`);

c. "datevalue" format, in which the year, month or quarter, and the associated value for the mean-adjustment for a given observation are given, in this order, in free

format on individual lines in the data file. Thus, a line of the data file with a mean adjustment of 1.01 for July of 1991 would have the form `1991 7 1.01` (`umformat = "datevalue"`);

d. the "x13save" format `X-13ARIMA-SEATS` uses to save a table. This allows the user to read in a file saved from a previous `X-13ARIMA-SEATS` run (`umformat = "x13save"`);[29]

e. a two character code which corresponds to a set of data formats used in previous versions of `X-11` and `X-11-ARIMA` (`umformat="1r"`);

f. the format that the `TRAMO` and `SEATS` programs use to read in a series and its descriptors. This enables `X-13ARIMA-SEATS` to read in a data file formatted for the `TRAMO` modeling program or the `SEATS` seasonal adjustment program. (`umformat = "tramo"`);

g. a variant of "free" format where the numbers must be separated by one or more spaces (not by commas or tabs), and decimal points are expressed as commas (a convention in some European countries). (`format = "freecomma"`);

h. a variant of "datevalue" format, where the year, month or quarter, and value of each observation are found in this order in free format on individual lines, where decimal points are expressed as commas. Thus, a line of the data file containing the value 355.398 for July of 1991 would have the form `1991 7 355,398`. The number of preceding blanks can vary (`format = "datevaluecomma"`).

In the predefined `X-11` data formats mentioned in (d), the data is stored in 6 or 12 character fields, with a year and series label associated with each year of data. For a complete list of these formats, see the DETAILS section of the **series** spec. If no **umformat** argument is given the data will be read in free format. The **umformat** argument cannot be used with the **umdata** argument, only with **umfile**.

umname The name of the series of values stored in the file named in **umfile**. The name must be enclosed in quotes and may contain up to 64 characters. Up to the first 16 characters will be printed as a label for the user-defined mean of the mean-adjustment values. When specified with the predefined formats of the **umformat** argument, the first six (or eight, if `umformat = "cs"`) characters of this name are also used with the predefined formats to check that the program is reading the correct series, or to find a particular series in a file where many series are stored.

umprecision The number of decimal digits to be read from the user-defined mean. This option can only be used with the predefined formats of the **umformat** argument. This value must be an integer between 0 and 5, inclusive (for example, `umprecision = 5`). The default is zero.

umstart The start date for the mean-adjustment values specified in **umdata** or **umfile**. The default is the start date of the series. Valid values are any date up to the start date of the series (or up to the start date of the span specified by the **span** argument of the **series** spec, if present).

[29]Note that to maintain compatibility with previous versions of `X-12-ARIMA` the entry **x12save** will also be accepted.

umtrimzero If `umtrimzero = no`, zeros at the beginning or end of the user mean time series entered via the umfile argument are treated as series values. If `umtrimzero = span`, causes leading and trailing zeros to be ignored outside the span of data being analyzed (the **span** argument must be specified with both a starting date and an ending date). The default (`umtrimzero = yes`) causes leading and trailing zeros to be ignored. Note that when the **format** argument is set to either `datevalue`, `x13save`, or `tramo`, all values input are treated as series values, regardless of the value of **umtrimzero**.

DETAILS

This spec is used to estimate a calendar effect, or other effect, from the irregular component I_t of a preliminary seasonal adjustment that did not adjust for the effect. The estimation is done by ordinary least squares (OLS) applied to a regression model for the effect.

In the simplest cases detailed below, the model has the form

$$I_t - 1.0 = \beta' X_t + e_t,$$

where X_t is a regression vector with variables that describe the basic effect of interest. In other cases, a more complicated linear transformation of I_t appears on the left of the model. In all cases, t-statistics, chi-square statistics, and AICs are calculated from the OLS estimates as though the regression errors e_t were independent and had constant variance. Unfortunately, the filtering operations used to produce I_t guarantee that both assumptions about e_t are not entirely correct, such that decisions made for the statistical significance of estimated effects from the statistics just mentioned are often less reliable than decisions made for effects estimated from a regARIMA model using the **regression** spec. That is, the statistics from **x11regression** are more likely than those from **regression** to suggest that a significant effect is present when it is not. For effects that are truly significant, the estimates from the **regression** and **x11regression** specs are usually quite close. When they differ appreciably, those from **regression** are usually more accurate than those from **x11regression**. (The forecast diagnostics of the **history** spec can be used to compare estimated effects for series of sufficient length, see Findley, Monsell, Bell, Otto, and Chen 1998 and Findley and Soukup 2000.) Thus use of **x11regression** should normally be restricted to series for which the user is unable to find a regARIMA model with good fit over the data span of interest.

Appendix C gives a detailed discussion of the irregular component regression models and their factors. Brief descriptions of the predefined regression variables that can be specified in the **x11regression** spec is given in Table 7.61 below.

Table 7.61: **Predefined Regression Variables for X11regression**

Variable	Description
td	Estimates monthly (or quarterly) flow trading-day effects by adding the `tdnolpyear` variables (see Table 7.28) to the model. The deviations of February from the average length of 28.25 are handled either by rescaling (for multiplicative adjustments) or by including the `lpyear` regression variable (for additive and log-additive adjustments). Cannot be used with `tdstock[ ]`, `tdstock1coef[ ]` or `td1coef`.
td1coef	Estimates monthly (or quarterly) flow trading-day effects by including the `td1nolpyear` variable (see below) in the model and handles leap-year effects either by rescaling (for transformed series) or by including the `lpyear` regression variable (for untransformed series). Can only be used for monthly or quarterly series, and cannot be used with `td`, `tdstock1coef[ ]` or `tdstock[ ]`.
tdstock[w]	Adds 6 stock trading-day variables to model the effect of the day of the week on a stock series estimated for the w-th day of each month. The value w must be supplied and can range from 1 to 31. For any month of length less than the specified w, the `tdstock` variables are measured as of the end of the month. Use `tdstock[31]` for end-of-month stock series. Can only be used with monthly series, and cannot be used with `td`, `tdstock1coef[ ]` or `td1coef`.
tdstock1coef[w]	Adds a constrained stock trading-day variable to model the effect of the day of the week on a stock series estimated for the w-th day of each month. The value w must be supplied and can range from 1 to 31. For any month of length less than the specified w, the `tdstock1coef` variables are measured as of the end of the month. Use `tdstock1coef[31]` for end-of-month stock series. Can only be used with monthly series, and cannot be used with `td`, `tdstock[ ]` or `td1coef`.
easter[w]	Easter holiday regression variable (monthly or quarterly flow data only) that assumes the level of daily activity changes on the w-th day before Easter and remains at the new level until the day before Easter. The value w must be supplied and can range from 1 to 25. A user can also specify an **easter**[0] regression variable, which assumes the daily level of activity level changes only on Easter Sunday. To estimate complex effects, several of these variables, differing in their choices of w, can be specified.
labor[w]	Labor Day holiday regression variable (monthly flow data only) that assumes the level of daily activity changes on the w-th day before Labor Day and remains at the new level until the day before Labor Day. The value w must be supplied and can range from 1 to 25.
thank[w]	Thanksgiving holiday regression variable (monthly flow data only) that assumes the level of daily activity changes on the w-th day before or after Thanksgiving and remains at the new level until December 24. The value w must be supplied and can range from -8 to 17. Values of $w < 0$ indicate a number of days after Thanksgiving; values of $w > 0$ indicate a number of days before Thanksgiving.
sceaster[w]	Statistics Canada Easter holiday regression variable (monthly or quarterly flow data only) assumes that the level of daily activity changes on the $(w-1)$-th day and remains at the new level through Easter day. The value w must be supplied and can range from 1 to 24. To estimate complex effects, several of these variables, differing in their choices of w, can be specified.

Table 7.61: **Predefined Regression Variables for X11regression** (continued)

Variable	Description
ao*date*	Adds an additive (point) outlier variable, AO, for the given date or observation number. For series with associated dates, AOs are specified as `ao`*date*. For monthly series the form is `ao`*year.month* (e.g., `ao1985.jul` or `ao1985.7`); for quarterly series it is `ao`*year.quarter* (e.g., `ao1985.1` for an AO in the first quarter of 1985). More than one AO may be specified. All specified outlier dates must occur within the series. (AOs with dates within the series but outside the span specified by the **span** argument of the **series** spec are ignored.)

The regression model specified by **x11regression** is estimated from the series of irregulars of the B and C iterations of the calculations of the **x11** spec. If the spec file also includes the **arima**, **automdl**, or **regression** spec, then the effects estimated via **x11regression** are obtained first, and they are removed from the data used for the estimations, or the forecast and backcast extensions, specified by these other specs. The series resulting from the calculations of these other specs is then decomposed by a second execution of the **x11** spec calculations in order to obtain the seasonal, trend, calendar-effect, and irregular components output by the program. Similarly, if the **x11** spec requests the Bateman-Mayes Easter-effect adjustment, this adjustment is calculated from a series that has been preadjusted for the effects estimated by **x11regression**.

If forecasting is performed, `X-13ARIMA-SEATS` creates data values for the selected predefined regression variables for the entire forecast period. If there are any user-defined regression variables, then data values must also be supplied for them for the entire forecast period.[30] In addition to the limit of 52 user-defined regression variables, there is an overall limit of 80 regression variables in the model. (These limits can be changed—see Section 2.8.) The latter limit is on the sum of the number of predefined and user-defined regression variables, plus the number of regression variables generated from automatic outlier detection. The maximum length of the series of user-defined regression variables, not including the forecast period, is 780. (This limit can also be changed—see Section 2.8.)

Trading day and/or holiday adjustments may be obtained either from regARIMA or from irregular regression models, but not from both. If these effects are estimated in both the **regression** and **x11regression** spec, then the **noapply** option must be used to ensure that only one set of factors is used in the adjustment.

The effect of the argument **aictest** can be to delete a regressor set named in the **variables** argument from this list or to add a regressor set to the model specified by the **variables** argument. The effect of a positive value of **aicdiff** is to make it more difficult for the **aictest** procedure to include the variable being tested in the model. Let Δ_{AIC} denote the value associated with the **aicdiff** argument, which by default is zero. Let $AICC^{with}$ (and $AICC^{without}$) denote the AICC value of the model with (or without) a set of regressors specified in the **aictest** argument. If this set is not named in the **variables** list, it will be added to the regression model if

$$AICC^{with} + \Delta_{AIC} < AICC^{without},$$

and similarly, if this set is already named in the **variables** list, it will be retained in the irregular component regression model only if this inequality holds.

[30] See the end of Section 4.7 for a discussion of what to do about forecast extension for seasonal adjustment of a series with a model that contains user-defined regressors whose future values are unknown.

If `aictest = (tdstock)`, then the end-of-month stock variables, specified by `tdstock[31]`, are the variables added, because 31 is the default value for w in **tdstock**[w].

There are more possibilities if `aictest = (easter)` and no Easter effect regressors appear in the **variables** argument. Then three additional models are considered, namely the models obtained by augmenting the specified irregular component regression model with the regressor **easter**[w] for $w = 1, 8, 15$, respectively. The Easter regressor whose model has the smallest AICC is retained if its AICC is smaller than the model with no Easter regressors by at least the amount $\Delta_{AIC} = 0$; otherwise, the model without Easter regressors is selected.

When trading day regressors appear in both the **aictest** and **variables** arguments, the type of regressors specified must be identical. The sample day for stock trading day variables and the date specified for change of regime regressors should *not* be included in the **aictest** argument; they will be assumed from the entry in the **variables** argument. For example, if `variables = (tdstock[15] ao1995.jan)`, then the entry for **aictest** should be `tdstock`.

If a trading day (**td** or **tdstock**) or holiday (**holiday**, or the specific U.S. holidays **easter**, **thanks**, and **labor**) regressor type is assigned in to a user-defined variable with the **usertype** argument, the factor derived from the user-defined regression variables of that type will be combined with the regression factor from variables of the same type specified in the **x11regression** spec. The resulting factor will be adjusted out of the series for the seasonal adjustment factor calculations determined by the **x11** spec unless the type name appears in the **noapply** argument.

If **x11regression** is used in a spec file without an **x11** spec, then the irregular component used for the regression is that obtained from the default specification `x11 { }`.

The two choices for the argument **eastermeans** yield noticeably different holiday factors. But the choice usually has negligible effects on the combined seasonal and holiday factors, because the seasonal factors change to compensate for the differences between the choices.

The monthly means used to obtain deseasonalized Easter regressors under `eastermeans = yes` were generated from frequencies of the date of Easter for a 500 year period (1600-2099). These frequencies can be computed by dates given in Bednarek (2019) which were checked using information from Montes (2001, 1997b, 1997a); the algorithm used to compute the date of Easter for the Gregorian calendar is given in Duffet-Smith (1981).

For a nonseasonal time series, an adjustment for trading day and holiday effects estimated by means of this spec can be obtained by setting `type = trend` in the **x11** spec.

When the **b** argument is used to fix coefficients, *AIC* and the other model selection statistics may become invalid — see the DETAILS section of **estimate**.

EXAMPLES

The following examples show complete spec files.

Example 1 Multiplicative seasonal adjustment with all default options (so the program uses the moving seasonality ratio to select the seasonal filter length). The monthly series starts in January 2006 and is stored in free format in the file `westus.dat` in the current directory. A trading day adjustment is done using a regression on the irregular component.

```
Series { File = "westus.dat"
         Start = 2006.1
       }
X11 {  }
X11Regression { Variables = td
              }
```

Example 2 Same as Example 1, only an AIC-based test will be performed to see if trading day and Easter regressors should be included in the regression on the irregular component.

```
Series { File = "westus.dat"
         Start = 2006.1
       }
X11 {  }
X11Regression { Variables = td
                Aictest = (td easter)
              }
```

Example 3 User-defined holiday regressors for the period both before and after Easter are included in the irregular regression along with trading day regressors. AO outlier identification will be performed during the irregular regression procedure.

```
series {
   file = "ukclothes.dat"
   start = 2005.Jan
}
x11 {  }
x11regression{
   variables = td             critical = 4.0
   user = (easter1 easter2)   file = "ukeaster.dat"
   usertype = holiday         start = 2000.Jan
}
```

Example 4 Prior trading day weights are provided with this spec file. The trading day weights calculated during the irregular regression will be combined with these weights for a combined trading day component.

```
series {
   file = "nzstarts.dat"  start = 2010.Jan
}
x11 {  }
x11regression{
   variables = td
   tdprior = (1.4 1.4 1.4 1.4 1.4 0.0 0.0)
}
```

Example 5 Perform a default seasonal adjustment. The trading day regressors in the **x11regression** spec will be fixed to their initial values; the Easter holiday regressor will be estimated.

```
series{
   format = '2R'
   title = 'MIDWEST ONE FAMILY Housing Starts'
   name = 'CMW1HS'
   file = 'cmw1hs.ori'
  span = (1994.01,2013.03)
}
x11{   }
x11regression{
        variables = (td easter[8])
        b = (  0.4453f  0.8550f -0.3012f  0.2717f
              -0.1705f  0.0983f -0.0082)
}
```

Example 6 Use an irregular component regression to estimate the trading day effect (with change of regime in January of 2009) and holiday effects.

```
series{
    title = 'Motor Home Sales'
    start = 1997.1
    span = (2002.1, )
    name = 'SB0562'
    file = 'C:\final.x12\T0B05601.TXT'
    format = '2L'
}
X11REGRESSION {  variables = ( td/2009.1/
   easter[8]  labor[10] thank[10] ) }
x11{
    seasonalma = x11default
    sigmalim = (1.8 2.8)
    appendfcst = YES
    save = (D11 D16)
}
```

Example 7 The predefined regression effects are trading day variables and a constant. The user-defined regression variables are for strikes in 2002, 2005, and 2010 and are located in the file `strikes.dat`. The ARIMA part of the model is $(0,1,1)(0,1,1)_{12}$. Since a model is specified in the spec, generate a year of forecasts by default. The seasonal period, 12, is not indicated since this is the default. Seasonally adjust the series after pre-adjusting for the user-defined regression effects. Before estimating the regARIMA model, do a prior pass to estimate a preliminary irregular and trading day and Easter effects, and remove the calendar effects from the series. A two-line seasonal adjustment title is specified.

```
series{ title = "Automobile Sales"
        file = "carsales.dat"
        start = 2001.Jan }
transform{ function = log }
regression{ variables = (const)
            user = (strike02 strike05 strike10)
            file = "strike.dat"
            format = "(3f12.0)" }
arima{ model = (0 1 1)(0 1 1)12 }
x11{  title = ("Car Sales in US"
               "Adjust for strikes in 2002, 2005, and 2010")
      save = seasonal appendfcst = yes
      }
x11regression{   variables = ( td easter[8] )  }
```

A Codes Associated With the X-13ARIMA-SEATS Graphics Metafile

As noted in section 2.7, the **-g** flag specifies a complete path name for a directory into which output will be stored that is intended as input for a separate graphics program. The program also stores a *graphics metafile* into this directory, which contains a list of the files stored by the program, along with codes that denote what table has been stored in the corresponding file.

Table A.1 below provides a list of all the tables that can be stored by `X-13ARIMA-SEATS` in graphics mode, along with the codes used in the graphics metafile to denote these files (in alphabetical order).

For example, if a record in the graphics metafile reads

```
sa g:\users\jones001\g2\StartsUS.d11
```

then the final seasonally adjusted series is stored in the file `g:\users\jones001\g2\StartsUS.d11`.

Table A.1: **Codes Associated with the `X-13ARIMA-SEATS` Graphics Metafile**

Code	Description
acf	residual autocorrelations
acf2	squared residual autocorrelations
adjcori	composite series (prior adjusted)
ador	original series (prior adjusted)
ahst	concurrent and revised seasonal adjustments and revisions
aichst	revision history of the likelihood statistics
ao	regARIMA AO outlier component
arat	final adjustment ratios
armahst	ARMA model coefficient history
bct	point backcasts and prediction intervals on the original scale
btr	point backcasts and standard errors for the transformed data
cad	regARIMA calendar adjusted original data
caf	combined adjustment factors
cal	combined calendar adjustment factors
ccal	final combined calendar factors from irregular component regression
cfchst	forecast and forecast error history
chol	combined holiday component
chss	sliding spans of the changes in the seasonally adjusted series
cmpcad	regARIMA calendar adjusted composite data
cmpoad	regARIMA outlier adjusted composite data
cmpori	composite time series data (for the span analyzed)

Table A.1: **Codes Associated with the X-13ARIMA-SEATS Graphics Metafile** (continued)

Code	Description
cmppadj	prior adjusted composite data
cmpspor	spectrum of the composite series
cmpsptukor	Tukey spectrum of the composite series
csahst	history of the period-to-period changes changes of the adjustments
ctd	final combined trading day factors from irregular component regression
ctrhst	history of the period-to-period changes of the trend-cycle values
fct	point forecasts and prediction intervals on the original scale
fcthst	revision history of the out-of-sample forecasts
fintst	final outlier test statistics
fltsac	concurrent seasonal adjustment filter
fltsaf	symmetric seasonal adjustment filter
flttrnc	concurrent trend filter
flttrnf	symmetric trend filter
frfc	factors applied to get adjusted series with forced yearly totals
ftr	point forecasts and standard errors for the transformed data
idacf	residual autocorrelations for different orders of differencing
idpacf	residual partial autocorrelations for different orders of differencing
indahst	concurrent and revised indirect seasonal adjustments and revisions
indao	indirect additive outlier adjustment factors
indarat	indirect final adjustment ratios
indcaf	indirect combined adjustment factors
indcal	indirect calendar component
indchss	sliding spans of the changes in the indirect seasonally adjusted series
indfrfc	factors applied to get indirect adjusted series with forced yearly totals
indirr	indirect irregular component
indls	indirect level shift adjustment factors
indmirr	irregular component modified for extremes from indirect adjustment
indmori	original data modified for extremes from indirect adjustment
indmsa	seasonally adjusted data modified for extremes from indirect adjustment
indrsi	final replacement values for SI component of indirect adjustment
indsa	indirect seasonally adjusted data
indsar	rounded indirect final seasonally adjusted series
indsass	sliding spans of the indirect seasonally adjusted series
indsat	final indirect seasonally adjusted series with forced yearly totals
indsf	indirect seasonal component
indsfss	sliding spans of the indirect seasonal factors
indsi	indirect unmodified SI component
indspir	spectrum of indirect modified irregular component
indspsa	spectrum of differenced indirect seasonally adjusted series
indsptukir	Tukey spectrum of indirect modified irregular component
indsptuksa	Tukey spectrum of differenced indirect seasonally adjusted series
indtadj	indirect total adjustment factors
indtrn	indirect trend cycle

Table A.1: **Codes Associated with the `X-13ARIMA-SEATS` Graphics Metafile** (continued)

Code	Description
indyyss	sliding spans of the year-to-year changes in the indirect seasonally adjusted series
irr	final irregular component
irrwt	final weights for irregular component
ls	regARIMA level shift outlier component
mdlest	regression and ARMA parameter estimates
mirr	modified irregular series
mori	original data modified for extremes
msa	modified seasonally adjusted series
mvadj	original series adjusted for missing value regressors
oad	regARIMA outlier adjusted original data
ori	time series data (for the span analyzed)
oricnt	time series data plus constant (for the span analyzed)
orifctd	series forecast decomposition (SEATS)
otl	regARIMA combined outlier component
pacf	residual partial autocorrelation
padj	prior-adjusted data
padjt	prior-adjusted data (including prior trading day adjustments)
ppradj	permanent prior-adjusted data
ppradjt	permanent prior-adjusted data (including prior trading day adjustments)
pprior	permanent prior-adjustment factors
prior	prior-adjustment factors
ptd	prior trading day factors
regrsd	residuals from the estimated regression effects
rgseas	regARIMA user-defined seasonal component
rhol	regARIMA holiday component
rsi	final replacement values for SI ratios
rtd	regARIMA trading day component
sa	final seasonally adjusted data
sac	final seasonally adjusted series with constant value added
safctd	final seasonally adjusted series forecast decomposition (SEATS)
sar	rounded final seasonally adjusted series
sass	sliding spans of the seasonally adjusted series
sat	final seasonally adjusted series with forced yearly totals
seataf	final combined adjustment factors (SEATS)
seatase	standard error of final seasonally adjusted series (SEATS)
seatcse	standard error of final transitory component (SEATS)
seatcyc	final cycle
seatdori	differenced original series after transformation, prior adjustment (SEATS)
seatdsa	differenced final seasonally adjusted series (SEATS)
seatdtr	differenced final trend (SEATS)
seatirr	final irregular component (SEATS)
seatirrotl	final irregular component outlier adjusted (SEATS)
seatltt	final long term trend

Table A.1: **Codes Associated with the `X-13ARIMA-SEATS` Graphics Metafile** (continued)

Code	Description
seatsa	final seasonally adjusted series (SEATS)
seatsaotl	final seasonally adjusted series adjusted for outliers (SEATS)
seatsf	final seasonal component (SEATS)
seatsse	standard error of final seasonal component (SEATS)
seatssm	sum of final seasonal component (SEATS)
seattrn	final trend component (SEATS)
seattse	standard error of final trend component (SEATS)
setarat	final adjustment ratios (SEATS)
setsac	final seasonally adjusted series with constant value added (SEATS)
settadj	total adjustment factors (SEATS)
settrc	final trend cycle with constant value added (SEATS)
settrns	final transitory component (SEATS)
sf	final seasonal factors
sffctd	final seasonal component forecast decomposition (SEATS)
sfhst	concurrent and projected seasonal component and their percent revisions
sfr	seasonal factors, adjusted for user-defined seasonal regARIMA component
sfss	sliding spans of the seasonal factors
sgsac	squared gain of the concurrent seasonal adjustment filter
sgsaf	squared gain of the symmetric seasonal adjustment filter
sgtrnc	squared gain of the concurrent trend filter
sgtrnf	squared gain of the symmetric trend filter
si	final unmodified SI ratios
siox	final unmodified SI ratios, with labels for outliers and extreme values
so	regARIMA seasonal outlier component
spcsir	spectrum of the irregular component (SEATS)
spcssa	spectrum of the seasonally adjusted series (SEATS)
spctuksir	Tukey spectrum of the irregular component (SEATS)
spctukssa	Tukey spectrum of the seasonally adjusted series (SEATS)
spexrsd	spectrum of the extended residuals (SEATS)
spir	spectrum of modified irregular series
spor	spectrum of the original series
sprsd	spectrum of the regARIMA model residuals
spsa	spectrum of differenced seasonally adjusted series
sptukexrsd	Tukey spectrum of the extended residuals (SEATS)
sptukir	Tukey spectrum of modified irregular series
sptukor	Tukey spectrum of the original series
sptukrsd	Tukey spectrum of the regARIMA model residuals
sptuksa	Tukey spectrum of differenced seasonally adjusted series
tadj	total adjustment factors
tc	regARIMA temporary change outlier component
tdhst	trading day coefficient history
tdss	sliding spans of the trading day factors
tprior	temporary prior-adjustment factors

Table A.1: **Codes Associated with the `X-13ARIMA-SEATS` Graphics Metafile** (continued)

Code	Description
trancmp	regARIMA transitory component
tranfcd	final transitory component forecast decomposition (SEATS)
trn	final trend cycle
trnfctd	final trend component forecast decomposition (SEATS)
trnhst	concurrent and revised Henderson trend-cycle values and revisions
tssac	time shift of the concurrent seasonal adjustment filter
tstrnc	time shift of the concurrent trend filter
usrdef	regARIMA user-defined regression component
xcal	final calendar factors from irregular component regression
xhol	final holiday factors from irregular component regression
xtd	final trading day factors from irregular component regression
xtrm	final extreme value adjustment factors
yyss	sliding spans of the year-to-year changes in the seasonally adjusted series

B Print and Save Tables

Contents

Tables

This appendix contains listings of the various output tables that can be printed or saved using the X-13-ARIMA-SEATS program.

Table B.1 contains a list of tables that are available for printing and saving using the **print** and **save** arguments of the individual specs.

A listing of special output tables associated with the **seats** spec is given in Appendix B.2, and a table with special output associated with the **spectrum** spec is given in Appendix B.3.

A listing of special tables that can be saved as percentages is given in Table B.6 in Appendix B.4.

B.1 Print and save tables available for X-13ARIMA-SEATS specs

Name gives the name of each table for use with the **print** and **save** arguments.

Short gives a short name for the tables of the **print** and **save** arguments. This name is also used as a file extension if the table is saved.

Save? indicates which tables can be saved (+) into a separate file with the **save** argument.

Brief indicates which tables are printed (+) when the brief print level is specified. See Section 3.2 for more information on print levels.

Default indicates which tables are printed (+) by default.

Spec indicates which spec the tables are defined for.

Table B.1: **Print and Save Tables**

Name	Short	Save?	Brief	Default	Spec
autochoice	ach	no	yes	yes	automdl
autochoicemdl	amd	no	no	no	automdl
autodefaulttests	adt	no	no	no	automdl
autofinaltests	aft	no	no	no	automdl
autoljungboxtest	alb	no	no	no	automdl
bestfivemdl	b5m	no	no	no	automdl
header	hdr	no	yes	yes	automdl
unitroottest	urt	no	yes	yes	automdl
unitroottestmdl	urm	no	no	no	automdl
acf	acf	yes	no	yes	check
acfplot	acp	no	no	yes	check
acfsquared	ac2	yes	no	yes	check
acfsquaredplot	ap2	no	no	yes	check
durbinwatson	dw	no	no	yes	check
friedmantest	frt	no	no	yes	check
histogram	hst	no	no	yes	check
normalitytest	nrm	no	no	yes	check
pacf	pcf	yes	no	no	check
pacfplot	pcp	no	no	no	check
adjcompositeplot	b1p	no	no	no	composite
adjcompositesrs	b1	yes	yes	yes	composite
calendaradjcomposite	cac	yes	no	no	composite
compositeplot	cmp	no	no	no	composite
compositesrs	cms	yes	yes	yes	composite
header	hdr	no	yes	yes	composite
indadjsatot	iaa	yes	yes	yes	composite
indadjustfac	iaf	yes	no	yes	composite
indadjustmentratio	i18	yes	no	no	composite
indaoutlier	iao	yes	no	yes	composite
indcalendar	ica	yes	no	yes	composite
indcalendaradjchanges	ie8	yes	no	yes	composite
indforcefactor	iff	yes	yes	yes	composite
indftestd8	idf	no	no	yes	composite
indirregular	iir	yes	no	yes	composite
indirregularplot	iip	no	no	no	composite
indlevelshift	ils	yes	no	yes	composite
indmcdmovavg	if1	yes	no	no	composite
indmodirr	ie3	yes	no	no	composite
indmodoriginal	ie1	yes	no	no	composite
indmodsadj	ie2	yes	no	no	composite
indqstat	if3	no	yes	yes	composite
indreplacsi	id9	yes	no	yes	composite

Table B.1: **Print and Save Tables** (continued)

Name	Short	Save?	Brief	Default	Spec
indresidualseasf	irf	no	no	yes	composite
indrevsachanges	i6a	yes	no	yes	composite
indrndsachanges	i6r	yes	no	yes	composite
indrobustsa	iee	yes	no	no	composite
indsachanges	ie6	yes	no	yes	composite
indsadjround	irn	yes	yes	yes	composite
indseasadj	isa	yes	yes	yes	composite
indseasadjplot	iap	no	no	no	composite
indseasonal	isf	yes	yes	yes	composite
indseasonaldiff	isd	yes	yes	yes	composite
indseasonalplot	isp	no	no	no	composite
indtest	itt	no	yes	yes	composite
indtotaladjustment	ita	yes	no	no	composite
indtrend	itn	yes	no	yes	composite
indtrendchanges	ie7	yes	no	yes	composite
indtrendplot	itp	no	no	no	composite
indunmodsi	id8	yes	no	yes	composite
indx11diag	if2	no	yes	yes	composite
indyrtotals	ie4	no	no	no	composite
origchanges	ie5	yes	no	yes	composite
origwindsaplot	ie0	no	no	no	composite
outlieradjcomposite	oac	yes	no	no	composite
prioradjcomposite	ia3	yes	no	no	composite
ratioplotindsa	ir2	no	no	no	composite
ratioplotorig	ir1	no	no	no	composite
armacmatrix	acm	yes	no	no	estimate
averagefcsterr	afc	no	no	yes	estimate
estimates	est	yes	yes	yes	estimate
iterationerrors	ite	no	no	no	estimate
iterations	itr	yes	no	no	estimate
lformulas	lkf	no	no	no	estimate
lkstats	lks	yes	yes	yes	estimate
model	mdl	yes	yes	yes	estimate
options	opt	no	no	yes	estimate
regcmatrix	rcm	yes	no	no	estimate
regressioneffects	ref	yes	no	no	estimate
regressionresiduals	rrs	no	no	no	estimate
residuals	rsd	yes	no	no	estimate
roots	rts	yes	no	no	estimate
forcefactor	ffc	yes	yes	yes	force
revsachanges	e6a	yes	no	yes	force
rndsachanges	e6r	yes	no	yes	force
saround	rnd	yes	yes	yes	force

Table B.1: **Print and Save Tables** (continued)

Name	Short	Save?	Brief	Default	Spec
seasadjtot	saa	yes	yes	yes	force
backcasts	bct	yes	no	no	forecast
forecasts	fct	yes	no	yes	forecast
transformed	ftr	yes	no	yes	forecast
transformedbcst	btr	yes	no	no	forecast
variances	fvr	yes	no	no	forecast
armahistory	amh	yes	yes	yes	history
chngestimates	che	yes	no	no	history
chngrevisions	chr	yes	no	yes	history
chngsummary	chs	no	yes	yes	history
fcsterrors	fce	yes	yes	yes	history
fcsthistory	fch	yes	no	no	history
header	hdr	no	yes	yes	history
indsaestimates	iae	yes	no	no	history
indsarevisions	iar	yes	no	yes	history
indsasummary	ias	no	yes	yes	history
lkhdhistory	lkh	yes	yes	yes	history
outlierhistory	rot	yes	yes	yes	history
saestimates	sae	yes	no	no	history
sarevisions	sar	yes	no	yes	history
sasummary	sas	no	yes	yes	history
seatsmdlhistory	smh	yes	yes	yes	history
sfestimates	sfe	yes	no	no	history
sfilterhistory	sfh	yes	no	no	history
sfrevisions	sfr	yes	no	yes	history
sfsummary	sfs	no	yes	yes	history
tdhistory	tdh	yes	yes	yes	history
trendchngestimates	tce	yes	no	no	history
trendchngrevisions	tcr	yes	no	yes	history
trendchngsummary	tcs	no	yes	yes	history
trendestimates	tre	yes	no	no	history
trendrevisions	trr	yes	no	yes	history
trendsummary	trs	no	yes	yes	history
acf	iac	yes	yes	yes	identify
acfplot	acp	no	no	yes	identify
pacf	ipc	yes	yes	yes	identify
pacfplot	pcp	no	no	yes	identify
regcoefficients	rgc	no	no	no	identify
finaltests	fts	yes	no	no	outlier
header	hdr	no	no	yes	outlier
iterations	oit	yes	no	no	outlier
temporaryls	tls	no	yes	yes	outlier
tests	ots	no	no	no	outlier

Table B.1: **Print and Save Tables** (continued)

Name	Short	Save?	Brief	Default	Spec
header	hdr	no	yes	yes	pickmdl
pickmdlchoice	pch	no	yes	yes	pickmdl
usermodels	umd	no	yes	yes	pickmdl
aictest	ats	no	yes	yes	regression
aoutlier	ao	yes	yes	yes	regression
chi2test	cts	no	yes	yes	regression
dailyweights	tdw	no	no	no	regression
holiday	hol	yes	yes	yes	regression
levelshift	ls	yes	yes	yes	regression
outlier	otl	yes	yes	yes	regression
regressionmatrix	rmx	yes	no	no	regression
regseasonal	a10	yes	yes	yes	regression
seasonaloutlier	so	yes	yes	yes	regression
temporarychange	tc	yes	yes	yes	regression
tradingday	td	yes	yes	yes	regression
transitory	a13	yes	yes	yes	regression
userdef	usr	yes	yes	yes	regression
adjustfac	s16	yes	no	no	seats
adjustmentratio	s18	yes	no	no	seats
cycle	cyc	yes	no	no	seats
difforiginal	dor	yes	no	no	seats
diffseasonaladj	dsa	yes	no	no	seats
difftrend	dtr	yes	no	no	seats
irregular	s13	yes	no	no	seats
irregularoutlieradj	se3	yes	no	no	seats
longtermtrend	ltt	yes	no	no	seats
seasadjconst	sec	yes	no	no	seats
seasonal	s10	yes	no	no	seats
seasonaladj	s11	yes	no	no	seats
seasonaladjfcstdecomp	afd	yes	no	no	seats
seasonaladjoutlieradj	se2	yes	no	no	seats
seasonalfcstdecomp	sfd	yes	no	no	seats
seasonalsum	ssm	yes	no	no	seats
seriesfcstdecomp	ofd	yes	no	no	seats
totaladjustment	sta	yes	no	no	seats
transitory	s14	yes	no	no	seats
transitoryfcstdecomp	yfd	yes	no	no	seats
trend	s12	yes	no	no	seats
trendconst	stc	yes	no	no	seats
trendfcstdecomp	tfd	yes	no	no	seats
adjoriginal	b1	yes	yes	yes	series
adjorigplot	b1p	no	no	no	series
calendaradjorig	a18	yes	no	no	series

Table B.1: **Print and Save Tables** (continued)

Name	Short	Save?	Brief	Default	Spec
header	hdr	no	yes	yes	series
outlieradjorig	a19	yes	no	no	series
savefile	sav	no	yes	yes	series
seriesmvadj	mv	yes	yes	yes	series
seriesplot	a1p	no	no	no	series
span	a1	yes	yes	yes	series
specfile	spc	yes	yes	yes	series
chngspans	chs	yes	no	no	slidingspans
factormeans	fmn	no	no	yes	slidingspans
header	hdr	no	yes	yes	slidingspans
indchngspans	cis	yes	no	no	slidingspans
indfactormeans	fmi	no	yes	yes	slidingspans
indpercent	pci	no	yes	yes	slidingspans
indsaspans	ais	yes	no	no	slidingspans
indsfspans	sis	yes	no	no	slidingspans
indsummary	smi	no	no	yes	slidingspans
indychngspans	yis	yes	no	no	slidingspans
indyypercent	piy	no	no	no	slidingspans
indyysummary	siy	no	no	no	slidingspans
percent	pct	no	yes	yes	slidingspans
saspans	ads	yes	no	no	slidingspans
sfspans	sfs	yes	no	no	slidingspans
ssftest	ssf	no	no	no	slidingspans
summary	sum	no	no	yes	slidingspans
tdspans	tds	yes	no	no	slidingspans
ychngspans	ycs	yes	no	no	slidingspans
yypercent	pcy	no	no	no	slidingspans
yysummary	suy	no	no	no	slidingspans
npsa	npa	no	no	no	spectrum
npsaind	npi	no	no	no	spectrum
qcheck	qch	no	yes	yes	spectrum
qs	qs	no	yes	yes	spectrum
qsind	qsi	no	yes	yes	spectrum
speccomposite	is0	yes	yes	yes	spectrum
specindirr	is2	yes	yes	yes	spectrum
specindsa	is1	yes	yes	yes	spectrum
specirr	sp2	yes	yes	yes	spectrum
specorig	sp0	yes	yes	yes	spectrum
specresidual	spr	yes	no	yes	spectrum
specsa	sp1	yes	yes	yes	spectrum
specextresiduals	ser	yes	yes	yes	spectrum
specseatsirr	s2s	yes	yes	yes	spectrum
specseatssa	s1s	yes	yes	yes	spectrum

Table B.1: **Print and Save Tables** (continued)

Name	Short	Save?	Brief	Default	Spec
tukeypeaks	tpk	no	yes	yes	spectrum
aictransform	tac	no	yes	yes	transform
permprior	a2p	yes	no	no	transform
permprioradjusted	a3p	yes	no	no	transform
permprioradjustedptd	a4p	yes	no	no	transform
prior	a2	yes	yes	yes	transform
prioradjusted	a3	yes	no	no	transform
prioradjustedptd	a4d	yes	no	no	transform
seriesconstant	a1c	yes	yes	yes	transform
seriesconstantplot	acp	no	no	no	transform
tempprior	a2t	yes	no	no	transform
transformed	trn	yes	no	no	transform
adjoriginalc	c1	yes	no	no	x11
adjoriginald	d1	yes	no	no	x11
adjustdiff	fad	yes	yes	yes	x11
adjustfac	d16	yes	yes	yes	x11
adjustmentratio	e18	yes	no	yes	x11
autosf	asf	no	no	no	x11
biasfactor	bcf	yes	no	no	x11
calendar	d18	yes	yes	yes	x11
calendaradjchanges	e8	yes	no	yes	x11
combholiday	chl	yes	yes	yes	x11
extreme	c20	yes	no	no	x11
extremeb	b20	yes	no	no	x11
ftestb1	b1f	no	no	no	x11
ftestd8	d8f	no	no	no	x11
irregular	d13	yes	no	yes	x11
irregularadjao	ira	yes	no	no	x11
irregularb	b13	yes	no	no	x11
irregularc	c13	yes	no	no	x11
irregularplot	irp	no	no	no	x11
irrwt	c17	yes	no	yes	x11
irrwtb	b17	yes	no	no	x11
mcdmovavg	f1	yes	no	no	x11
modirregular	e3	yes	no	no	x11
modoriginal	e1	yes	no	no	x11
modseasadj	e2	yes	no	no	x11
modsic4	c4	yes	no	no	x11
modsid4	d4	yes	no	no	x11
movseasrat	d9a	no	no	yes	x11
origchanges	e5	yes	no	yes	x11
origwsaplot	e0	no	no	no	x11
qstat	f3	no	no	no	x11

Table B.1: **Print and Save Tables** (continued)

Name	Short	Save?	Brief	Default	Spec
ratioplotorig	ra1	no	no	no	x11
ratioplotsa	ra2	no	no	no	x11
replacsi	d9	yes	no	yes	x11
replacsib4	b4	no	no	no	x11
replacsib9	b9	no	no	no	x11
replacsic9	c9	yes	no	no	x11
residualseasf	rsf	no	no	no	x11
robustsa	e11	yes	no	no	x11
sachanges	e6	yes	no	yes	x11
seasadj	d11	yes	yes	yes	x11
seasadjb11	b11	yes	no	no	x11
seasadjb6	b6	yes	no	no	x11
seasadjc11	c11	yes	no	no	x11
seasadjc6	c6	yes	no	no	x11
seasadjconst	sac	yes	yes	yes	x11
seasadjd6	d6	yes	no	no	x11
seasadjplot	sap	no	no	no	x11
seasonal	d10	yes	yes	yes	x11
seasonaladjregsea	ars	yes	yes	yes	x11
seasonalb10	b10	yes	no	no	x11
seasonalb5	b5	yes	no	no	x11
seasonalc10	c10	yes	no	no	x11
seasonalc5	c5	yes	no	no	x11
seasonald5	d5	yes	no	no	x11
seasonaldiff	fsd	yes	yes	yes	x11
seasonalplot	sfp	no	no	no	x11
sib3	b3	yes	no	no	x11
sib8	b8	yes	no	no	x11
tdadjorig	c19	yes	no	no	x11
tdadjorigb	b19	yes	no	no	x11
tdaytype	tdy	no	yes	yes	x11
totaladjustment	tad	yes	no	yes	x11
trend	d12	yes	no	yes	x11
trendadjls	tal	yes	no	no	x11
trendb2	b2	yes	no	no	x11
trendb7	b7	yes	no	no	x11
trendc2	c2	yes	no	no	x11
trendc7	c7	yes	no	no	x11
trendchanges	e7	yes	no	yes	x11
trendconst	tac	yes	no	no	x11
trendd2	d2	yes	no	no	x11
trendd7	d7	yes	no	no	x11
trendplot	trp	no	no	no	x11

Table B.1: **Print and Save Tables** (continued)

Name	Short	Save?	Brief	Default	Spec
unmodsi	d8	yes	no	yes	x11
unmodsiox	d8b	yes	no	yes	x11
x11diag	f2	no	no	no	x11
yrtotals	e4	yes	no	yes	x11
calendar	xca	yes	yes	yes	x11regression
calendarb	bxc	yes	no	no	x11regression
combcalendar	xcc	yes	yes	yes	x11regression
combcalendarb	bcc	yes	no	no	x11regression
combtradingday	c18	yes	yes	yes	x11regression
combtradingdayb	b18	yes	no	no	x11regression
extremeval	c14	yes	no	yes	x11regression
extremevalb	b14	yes	no	no	x11regression
holiday	xhl	yes	yes	yes	x11regression
holidayb	bxh	yes	no	no	x11regression
outlierfinaltests	xft	no	no	no	x11regression
outlierhdr	xoh	no	no	yes	x11regression
outlieriter	xoi	yes	no	no	x11regression
outliertests	xot	no	no	no	x11regression
priortd	a4	yes	yes	yes	x11regression
tradingday	c16	yes	yes	yes	x11regression
tradingdayb	b16	yes	no	no	x11regression
x11reg	c15	yes	no	yes	x11regression
x11regb	b15	yes	no	no	x11regression
xaictest	xat	no	yes	yes	x11regression
xregressioncmatrix	xrc	yes	no	no	x11regression
xregressionmatrix	xrm	yes	no	no	x11regression

B.2 Special output related to the seats spec

Tables B.2 and B.3 give examples of special types of output files that can be produced by the **seats** spec.

Table B.2 gives a listing of tables that can only be saved by the program by using the **save** argument within the **seats** spec.

name	*short*	*description of table*
componentmodels	mdc	models for the components
filtersaconc	fac	concurrent finite seasonal adjustment filter
filtersasym	faf	symmetric finite seasonal adjustment filter
filtertrendconc	ftc	concurrent finite trend filter
filtertrendsym	ftf	symmetric finite trend filter
pseudoinnovtrend	pic	pseudo-innovations of the trend component
pseudoinnovseasonal	pis	pseudo-innovations of the seasonal component
psuedoinnovtransitory	pit	pseudo-innovations of the transitory component
psuedoinnovsadj	pia	pseudo-innovations of the final SEATS seasonal adjustment
squaredgainsaconc	gac	squared gain for finite concurrent seasonal adjustment filter
squaredgainsasym	gaf	squared gain for finite symmetric seasonal adjustment filter
squaredgaintrendconc	gtc	squared gain for finite concurrent trend filter
squaredgaintrendsym	gtf	squared gain for finite symmetric trend filter
timeshiftsaconc	tac	time shift for finite concurrent seasonal adjustment filter
timeshifttrendconc	ttc	time shift for finite concurrent trend filter
wkendfilter	wkf	end filters of the semi-infinite Wiener-Kolmogorov filter

Name gives the name of each table for use with the **save** argument.
Short gives a short name for these tables.

Table B.2: **Output Tables Available Only with `save` Argument for Seats**

Note that the **out** argument can control whether one of these tables can be saved; Section 7.14 has more information.

`X-13ARIMA-SEATS` can also save other output files that were produced by the original SEATS program. These output files can contain forecasts, components or diagnostics generated from the SEATS model-based adjustment performed. Table B.3 shows the file extensions that are used to save the corresponding special output file from SEATS in the same way the short table names are used as file extensions in storing individual tables to separate files.

The **tabtables** argument of the **seats** spec gives a list of seasonal adjustment components and series to be stored in a separate file with the extension `_tbs.html`. The list is entered as a text string with codes listed in Table B.4; individual entries can be separated by commas (`tabtables = "xo,n,s,p"`) or spaces (`tabtables = "xo n s p"`). Note that components can only be added – they cannot be removed as in the **print** argument.

SEATS file name	*X-13ARIMA-SEATS extension*	*Contents of file*
`rogtable.out`	*filename*`_rog.html`	selected statistics from the growth rate output
`summarys.txt`	*filename*`_sum.html`	summary information and diagnostics from SEATS adjustment
`table-s.out`	*filename*`_tbs.html`	annotated listing of the series, the seasonally adjusted series, and components of the model-based seasonal adjustment, saved in table format

SEATS file name gives the file name saved by the SEATS program.
X-13ARIMA-SEATS extension gives the file extension used to save the output from the corresponding SEATS output file.

Table B.3: `X-13ARIMA-SEATS` **File Extensions for Special SEATS Saved Output**

Code	*Description of table*
all	all series
xo	original series
n	seasonally adjusted series
s	seasonal factors
p	trend-cycle
u	irregular
c	transitory
cal	calendar
pa	preadjustment factor
cy	cycle
ltp	long term trend
er	residuals
rg0	separate regression component
rgsa	regression component in seasonally adjusted series
stp	stochastic trend cycle
stn	stochastic seasonally adjusted series
rtp	real time trend cycle
rtsa	real time seasonally adjusted series

Code gives the code used to specify the series in the **tabtables** argument.

Table B.4: **Components Savable in** `_tbs.html` **File**

B.3 Special output related to the spectrum spec

Table B.5 gives a listing of tables that can only be saved by the program by using the **save** argument within the **spectrum** spec.

name	*short*	*description of table*
spectukeyorig	st0	Tukey spectral estimates of the first-differenced original series
spectukeysa	st1	Tukey spectral estimates of differenced, X-11 seasonally adjusted series (or of the logged seasonally adjusted series if **mode = logadd** or **mode = mult**)
spectukeyirr	st2	Tukey spectral estimates of outlier-modified X-11 irregular series
spectukeyseatssa	t1s	Tukey spectrum of the differenced final SEATS seasonal adjustment
spectukeyseatsirr	t2s	Tukey spectrum of the final SEATS irregular
spectukeyextresiduals	ter	Tukey spectrum of the extended residuals
spectukeyresidual	str	Tukey spectral estimates of the regARIMA model residuals
spectukeycomposite	it0	Tukey spectral estimates of first-differenced aggregate series
spectukeyindsa	it1	Tukey spectral estimates of the first-differenced indirect seasonally adjusted series
spectukeyindirr	it2	Tukey spectral estimates of outlier-modified irregular series from the indirect seasonal adjustment

Name gives the name of each table for use with the **save** argument.
Short gives a short name for these tables.

Table B.5: **Output Tables Available Only with save Argument for Spectrum**

B.4 Tables that save X-13ARIMA-SEATS output as percentages

Table B.6 gives table names and abbreviations that can be used with the **save** argument to save certain tables as percentages rather than ratios. The percentages are only produced when multiplicative or log-additive seasonal adjustment is specified by the user in the **mode** argument of the **x11** spec (or a log transformation is specified in the **transform** spec in the case of the tables from the **seats** spec).

name	*short*	*spec*	*description of table*
indadjustfacpct	ipf	composite	indirect combined adjustment factors expressed as percentages if appropriate
indcalendaradjchangespct	ip8	composite	percent changes in original series adjusted for calendar effects
indirregularpct	ipi	composite	indirect irregular component expressed as percentages if appropriate
indrevsachangespct	ipa	composite	percent changes for indirect seasonally adjusted series with forced yearly totals
indrndsachangespct	ipr	composite	percent changes for rounded indirect seasonally adjusted series
indsachangespct	ip6	composite	percent changes for indirect seasonally adjusted series
indseasonalpct	ips	composite	indirect seasonal component expressed as percentages if appropriate
indtrendchangespct	ip7	composite	percent changes for indirect trend component
origchangespct	ip5	composite	percent changes for composite series
revsachangespct	p6a	force	percent changes in seasonally adjusted series with forced yearly totals
rndsachangespct	p6r	force	percent changes in rounded seasonally adjusted series
adjustfacpct	psa	seats	combined adjustment factors, expressed as percentages if appropriate
irregularpct	psi	seats	final irregular component, expressed as percentages if appropriate
transitorypct	psc	seats	final transitory component, expressed as percentages if appropriate
seasonalpct	pss	seats	final seasonal factors, expressed as percentages if appropriate
adjustfacpct	paf	x11	combined adjustment factors, expressed as percentages if appropriate
calendaradjchangespct	pe8	x11	percent changes in original series adjusted for calendar factors
irregularpct	pir	x11	final irregular component, expressed as percentages if appropriate
origchangespct	pe5	x11	percent changes in the original series
sachangespct	pe6	x11	percent changes in seasonally adjusted series
seasonalpct	psf	x11	final seasonal factors, expressed as percentages if appropriate
trendchangespct	pe7	x11	percent changes in final trend cycle

Name gives the name of each table for use with the **save** argument.
Short gives a short name for these tables.
Spec indicates the corresponding spec for each table.

Table B.6: **Tables Savable as Percentages in the `save` Argument**

C Irregular-Component Regression Models in X-13ARIMA-SEATS

Contents

This appendix gives details of the models applied by the X-13ARIMA-SEATS **x11regression** spec to the irregular component to estimate calendar effects. The regression models detailed in this appendix are used to estimate a calendar effect, or other effect, from the irregular component I_t of a preliminary seasonal adjustment that did not adjust for the effect. The estimation is done by ordinary least squares (OLS) applied to a regression model for the effect. In the simplest cases detailed below, the model has the form

$$I_t - 1.0 = \beta' X_t + e_t,$$

where X_t is a regression vector with variables that describe the basic effect of interest.

C.1 Irregular regression models for multiplicative decompositions

The irregular component is presumed to have no seasonality or trend (beyond a constant level of 1.0, in the case of a multiplicative decomposition). Hence, the regressors that are used in regression models for the irregulars should usually not have a seasonal or trend component. For this reason, most trading day and Easter regression functions used in the **regression** spec (see Tables 4.1 and 7.28) are modified for use in the **x11regression** spec (see Table 7.61). The modifications for trading day variables for the various types of seasonal adjustment decompositions are derived in section 1.4 of Findley, Monsell, Bell, Otto, and Chen (1998). We will indicate the nature of this modification with a combined monthly flow trading day and holiday regression function of the form

$$\gamma_0 m_t + \sum_{j=1}^{6} \gamma_j (d_{j,t} - d_{7,t}) + \boldsymbol{\delta}' \mathbf{H}_t, \tag{C.1}$$

where $d_{j,t}$ = no. of weekdays of type j in month t (with $j = 1, \ldots, 7$ denoting Monday, ..., Sunday, respectively), $m_t = \sum_{j=1}^{7} d_{j,t}$ (the length on month t in days), and $\mathbf{H}_t$ denotes a (column) vector of holiday regressors.

Because of the definition of the calendar, over most time intervals of interest these variables are periodic, $m_{t+48} = m_t$, $d_{j,t+336} = d_{j,t}$, and $\mathbf{H}_{t+P} = \mathbf{H}_t$ with P depending on the holiday variables included in $\mathbf{H}_t$. (If all proposed corrections to the Gregorian calendar are used, the Easter calendar has a period of $38,000$ years, or $456,000$ months. For this reason it is often more practical to choose P so that approximate periodicity holds, $\mathbf{H}_{t+P} \simeq \mathbf{H}_t$).

If f_t is an approximately periodic function of period $12p$ months, $f_{t+12p} \simeq f_t$, then its (approximate) combined seasonal and level component is given by its calendar month means

$$f_t^* = \frac{1}{p}\sum_{j=1}^{p} f_{t+12j},$$

which is approximately periodic with period 12 months, $f_{t+12}^* \simeq f_t^*$. If seasonal and level effects are removed from f_t by division, the resulting deseasonalized, level-neutral component of f_t is f_t/f_t^*. To apply these ideas to the function (C.1) above, we note that if p is a multiple of 28, then $d_{j,t}^* = d_{7,t}^*, 1 \le j \le 6$, with the result that the seasonal and level component of this calendar effect function is

$$\gamma_0 m_t^* + \boldsymbol{\delta}'\mathbf{H}_t^*,$$

with

$$m_t^* = \begin{cases} m_t & \text{if } m_t = 30, 31 \\ 28.25 & \text{if } m_t = 28, 29 \end{cases}.$$

Therefore, if a time series contains a trading day and holiday component of the form (C.1), then its irregular component from multiplicative deseasonalization and detrending can be expected to have a trading day and holiday component close to

$$\frac{\gamma_0 m_t + \sum_{j=1}^{6}\gamma_j(d_{j,t} - d_{7,t}) + \boldsymbol{\delta}'\mathbf{H}_t}{\gamma_0 m_t^* + \boldsymbol{\delta}'\mathbf{H}_t^*} = \frac{\frac{m_t}{m_t^*} + \sum_{j=1}^{6}\alpha_j\left((d_{j,t} - d_{7,t})/m_t^*\right) + \boldsymbol{\beta}'\frac{\mathbf{H}_t}{m_t^*}}{1 + \boldsymbol{\beta}'\frac{\mathbf{H}_t^*}{m_t^*}}. \tag{C.2}$$

The expression on the right is a nonlinear function of $\alpha_j = \gamma_j/\gamma_0$ and $\boldsymbol{\beta} = \boldsymbol{\delta}/\gamma_0$. However, because trading day effects and holiday effects are usually in the range of a few percent, the approximation

$$\left(1 + \boldsymbol{\beta}'\frac{\mathbf{H}_t^*}{m_t^*}\right)^{-1} \simeq 1 - \boldsymbol{\beta}'\frac{\mathbf{H}_t^*}{m_t^*}$$

can be applied to (C.2). After multiplying the numerator on the right in (C.2) by this factor, the terms that involve products of coefficients are generally small enough to be ignored. This yields the following linear approximation to (C.2),

$$\frac{m_t}{m_t^*} + \sum_{j=1}^{6}\alpha_j\left(\frac{d_{j,t} - d_{7,t}}{m_t^*}\right) + \boldsymbol{\beta}'\left(\frac{\mathbf{H}_t - \mathbf{H}_t^*}{m_t^*}\right). \tag{C.3}$$

In obtaining this approximation, we have also made use of

$$\frac{m_t}{m_t^*} = 1 + \frac{1}{28.25}(m_t - m_t^*),$$

and have treated the term involving the leap year variable $LY_t = m_t - m_t^*$ as one whose product with $\boldsymbol{\beta}'(\mathbf{H}_t - \mathbf{H}_t^*)/m_t^*$ is negligible. The formula (C.3) suggests a linear regression model for the irregular component I_t of the form

$$I_t - \frac{m_t}{m_t^*} = \sum_{j=1}^{6} \alpha_j \left(\frac{d_{j,t} - d_{7,t}}{m_t^*} \right) + \boldsymbol{\beta}' \left(\frac{\mathbf{H}_t - \mathbf{H}_t^*}{m_t^*} \right) + \boldsymbol{\kappa}' \mathbf{AO}_t + e_t,$$

where $\mathbf{AO}_t$ denotes a regression vector containing any needed additive outlier variables. Instead of using this model, `X-13ARIMA-SEATS`, for conformity with the `X-11` and `X-11-ARIMA` trading day regression models, obtains the coefficients in (C.3) from ordinary least squares estimation (OLS) of the rescaled model

$$m_t^* I_t - m_t = \sum_{j=1}^{6} \alpha_j (d_{j,t} - d_{7,t}) + \boldsymbol{\beta}'(\mathbf{H}_t - \mathbf{H}_t^*) + \boldsymbol{\kappa}' \mathbf{AO}_t + \varepsilon_t \tag{C.4}$$

whenever `td` is specified in the **variables** argument of **x11regression**, with one or more of the holiday effect specifications `easter[w]`, `labor[w]`, and `thank[w]`. As explained in the footnote of Table 4.1, the regressors associated with these holiday variables also have the deseasonalized form $\mathbf{H}_t - \mathbf{H}_t^*$ when they are estimated from the **regression** spec. This is done so that seasonal effects occur only in the seasonal part of the model, and only in the seasonal factors of the decomposition. For conformity with `X-11-ARIMA/88`, the regressors associated with `sceaster[w]` are never deseasonalized. In effect, the entries of $\mathbf{H}_t^*$ in (C.4) associated with any specified `sceaster[w]` regressors are set to zero.

C.1.1 Obtaining separate trading day and holiday factors

The calendar factors (C.3) can be approximately factored as the product of holiday factors

$$1 + \boldsymbol{\beta}' \left(\frac{\mathbf{H}_t - \mathbf{H}_t^*}{m_t^*} \right) \tag{C.5}$$

and trading day factors

$$\frac{m_t}{m_t^*} + \sum_{j=1}^{6} \alpha_j \left(\frac{d_{j,t} - d_{7,t}}{m_t^*} \right) = \frac{\sum_{j=1}^{7} (1 + \alpha_j) d_{j,t}}{m_t^*}, \tag{C.6}$$

(with $\alpha_7 = -\sum_{j=1}^{6} \alpha_j$). The numbers $1 + \alpha_j$ are called the *daily weights*. The trading day factor formula (C.6) can also be written as

$$\frac{m_t}{m_t^*} + \frac{\sum_j^{(5)} \alpha_j}{m_t^*},$$

where $\sum_j^{(5)}$ denotes the sum of the j for which $d_{jt} = 5$. This formula shows that, apart from length of month effects, the trading day effects depend only on the effects of the days that occur five times in the month. When only trading day effects are estimated, the formulas above apply with $\boldsymbol{\beta} = \mathbf{0}$.

If at least one of the trading day "weights" $1+\alpha_j$ is negative and the option **reweight** has been specified, then, for the trading day factor calculation, all $\alpha_j < -1$ are replaced by $\alpha'_j = -1$ and all $\alpha_j \geq -1$ are replaced by $\alpha'_j = (1+\alpha_j)\, w - 1$, where

$$w = 7 \left\{ \sum_{\alpha_i \geq -1} (1+\alpha_i) \right\}^{-1},$$

assuming no $\alpha_j > -1$ has already been assigned a fixed value using the **b** argument. If there are fixed values, only unfixed $\alpha_j > -1$ are replaced, and in the replacement formula w is defined by

$$w = \left\{ 7 - \sum_{\alpha_i \text{ fixed}} (1+\alpha_i) \right\} \left\{ \sum_{\alpha_i \text{ not fixed}} (1+\alpha_i) \right\}^{-1},$$

for all $\alpha_i > -1$.

C.1.2 Estimating only holiday effects or stock trading day effects.

If only holiday effects, or stock trading day effects, are specified in the **variables** argument of **x11regression**, then X-13ARIMA-SEATS estimates these effects by OLS applied to models of the form

$$I_t - 1 = \boldsymbol{\beta}' (\mathbf{H}_t - \mathbf{H}^*_t) + \boldsymbol{\kappa}' \mathbf{AO}_t + e_t \tag{C.7}$$

$$I_t - 1 = \sum_{j=1}^{6} \alpha_j D_{j,t} + \boldsymbol{\kappa}' \mathbf{AO}_t + e_t \tag{C.8}$$

for holiday and stock trading day, respectively (where the $D_{j,t}$ in (C.8) are the regressors associated with the specified tdstock[w] in Table 4.1). These models lead to calendar effect adjustment factors of the form

$$1 + \boldsymbol{\beta}' (\mathbf{H}_t - \mathbf{H}^*_t) \tag{C.9}$$

$$1 + \sum_{j=1}^{6} \alpha_j D_{j,t} = 1 - \alpha_{j(t)} \tag{C.10}$$

for holiday and stock trading day, respectively (where the $\alpha_{j(t)}$ in (C.10) is the coefficient associated with the w-th day of month t).

C.1.3 Estimating user-defined flow trading day and holiday effects

The regression model (C.4) yields m_t/m^*_t as the component of the mean function for the irregulars I_t that is known independently of the estimated coefficients. This is also the default specification of the known component when user-defined variables are used. If this default is accepted, then the OLS regression model with at least one user-defined trading day or holiday variable has the form

$$m^*_t I_t - m_t = \boldsymbol{\alpha}' \mathbf{TD}_t + \boldsymbol{\beta}' \tilde{\mathbf{H}}_t + \boldsymbol{\kappa}' \mathbf{AO}_t + \varepsilon_t, \tag{C.11}$$

with $\mathbf{TD}_t$ and $\tilde{\mathbf{H}}_t$ denoting the vectors of trading day and holiday variables specified. User-defined variables are input by way of **file**or **data** arguments. The program does not deseasonalize user-defined variables. They should be input to the program in an appropriately deseasonalized form. `X-13ARIMA-SEATS` calculates calendar factors

$$\frac{m_t}{m_t^*} + \boldsymbol{\alpha}' \frac{\mathbf{TD}_t}{m_t^*} + \boldsymbol{\beta}' \frac{\tilde{\mathbf{H}}_t}{m_t^*}$$

that are approximately factored into holiday factors and trading day factors in analogy with (C.5), and (C.6). If only holiday effects are estimated, then the default known mean function component is the constant 1.0, and the model and resulting holiday factors are the analogues of (C.7) and (C.9). Similarly, if only stock trading day effects are estimated, then the default known mean function component is the constant 1.0, and the model and resulting holiday factors are the analogues of (C.8) and (C.10).

When the default known mean functions just described are not appropriate, the user can input a mean function μ_t by means of the **umfile** or **umdata** arguments. In this case, the regression model estimated is

$$I_t - \mu_t = \boldsymbol{\alpha}' \mathbf{TD}_t + \boldsymbol{\beta}' \tilde{\mathbf{H}}_t + \boldsymbol{\kappa}' \mathbf{AO}_t + e_t \tag{C.12}$$

and only the calendar factors

$$\mu_t + \boldsymbol{\alpha}' \mathbf{TD}_t + \boldsymbol{\beta}' \tilde{\mathbf{H}}_t$$

are produced. The coefficients $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, estimated from (C.12) are on a different scale from those obtained from (C.11), being smaller by the approximate factor

$$\frac{1}{48} \sum_{j=0}^{47} \frac{1}{m_t^*} \simeq 0.03288.$$

The same approximate scale difference holds for calendar coefficients calculated from **regression** instead of **x11regression**, or from (C.7) or (C.8) instead of (C.4).

C.2 Irregular regression models for other decomposition modes

We present below the models used with additive, pseudo-additive, and log-additive decompositions for the case of combined flow trading day and holiday effect estimation with predefined regressors. The appropriate modifications to these models for the case of user-defined, stock trading day or holiday regressors are analogous to those described above for multiplicative decompositions.

C.2.1 Additive decompositions

If `mode = add` in the **x11** spec, calendar effects are estimated by OLS from a model of the form

$$I_t = \alpha_0 LY_t + \sum_{j=1}^{6} \alpha_j (d_{j,t} - d_{7,t}) + \boldsymbol{\beta}' (\mathbf{H}_t - \mathbf{H}_t^*) + \boldsymbol{\kappa}' \mathbf{AO}_t + e_t.$$

The calendar effect is thus exactly the sum of the trading day effect $\alpha_0 LY_t + \sum_{j=1}^{6} \alpha_j (d_{j,t} - d_{7,t})$ and the holiday effect $\boldsymbol{\beta}' (\mathbf{H}_t - \mathbf{H}_t^*)$.

C.2.2 Pseudo-additive decompositions

If `mode = pseudoadd` in the **x11** spec, then, with $\bar{m} = 30.4375$ and $LY_t = m_t - m_t^*$, calendar effects are estimated by OLS from a model of the form

$$\bar{m}\,(I_t - 1) - LY_t = \sum_{j=1}^{6} \alpha_j (d_{j,t} - d_{7,t}) + \boldsymbol{\beta}'\,(\mathbf{H}_t - \mathbf{H}_t^*) + \boldsymbol{\kappa}'\mathbf{AO}_t + \varepsilon_t.$$

The calendar effect factors

$$1 + \frac{1}{\bar{m}} LY_t + \sum_{j=1}^{6} \alpha_j \left(\frac{d_{j,t} - d_{7,t}}{\bar{m}} \right) + \boldsymbol{\beta}' \left(\frac{\mathbf{H}_t - \mathbf{H}_t^*}{\bar{m}} \right)$$

can be approximately factored as

$$\left\{ 1 + \frac{1}{\bar{m}} LY_t + \sum_{j=1}^{6} \alpha_j \left(\frac{d_{j,t} - d_{7,t}}{\bar{m}} \right) \right\} \left\{ 1 + \boldsymbol{\beta}' \left(\frac{\mathbf{H}_t - \mathbf{H}_t^*}{\bar{m}} \right) \right\}$$

to obtain trading day and holiday factors.

C.2.3 Log-additive decompositions

If `mode = logadd` in the **x11** spec, calendar effects are estimated by OLS from a model of the form

$$m_t^* (\log I_t + 1) - m_t = \sum_{j=1}^{6} \alpha_j (d_{j,t} - d_{7,t}) + \boldsymbol{\beta}'(\mathbf{H}_t - \mathbf{H}_t^*) + \boldsymbol{\kappa}'\mathbf{AO}_t + \varepsilon_t. \tag{C.13}$$

These can be exactly factored into trading day and holiday factors,

$$\exp \left\{ -1 + \frac{m_t}{m_t^*} + \sum_{j=1}^{6} \alpha_j \left(\frac{d_{j,t} - d_{7,t}}{m_t^*} \right) \right\} \exp \left\{ \boldsymbol{\beta}' \left(\frac{\mathbf{H}_t - \mathbf{H}_t^*}{m_t^*} \right) \right\}.$$

Two other useful forms for the trading day factors can be obtained:

$$\exp \left\{ -1 + \frac{m_t}{m_t^*} + \sum_{j=1}^{6} \alpha_j \left(\frac{d_{j,t} - d_{7,t}}{m_t^*} \right) \right\} = \exp \left\{ -1 + \frac{m_t}{m_t^*} \right\} \exp \left\{ \sum_{j=1}^{6} \alpha_j \left(\frac{d_{j,t} - d_{7,t}}{m_t^*} \right) \right\} \tag{C.14}$$

$$= \exp \left\{ -1 + \sum_{j=1}^{7} (1 + \alpha_j) \frac{d_{j,t}}{m_t^*} \right\}. \tag{C.15}$$

(C.14) uses the fact that $e^x \simeq 1 + x$ for small values of x to emphasize the leap year factors $\exp\{-1 + m_t/m_t^*\} \simeq m_t/m_t^*$, while (C.15) uses the identity from (C.6) to emphasize the daily weights $(1 + \alpha_j)$.

C.3 When tdprior is used

Any of the coefficients in the models above can be assigned fixed values by an appropriate specification of the **b** argument. Sometimes users have prior information that suggests values for the seven daily weights associated with the trading day factors (C.6) of multiplicative, or (C.13) of log-additive, adjustment. When "prior" daily weights $1+\alpha_j^{(p)}$, $1 \le j \le 7$ are assigned values by means of the **tdprior** argument, the series is preadjusted by

$$\frac{\sum_{j=1}^{7}\left(1+\alpha_j^{(p)}\right)d_{j,t}}{m_t^*} = \frac{m_t}{m_t^*}\left\{1+\sum_{j=1}^{6}\alpha_j^{(p)}\left(\frac{d_{j,t}-d_{7,t}}{m_t}\right)\right\} \tag{C.16}$$

when `mode = mult` in **x11**, or by

$$\exp\left\{-1+\frac{m_t}{m_t^*}\right\}\exp\left\{\sum_{j=1}^{6}\alpha_j^{(p)}\left(\frac{d_{j,t}-d_{7,t}}{m_t^*}\right)\right\} \tag{C.17}$$

when `mode = logadd`. One advantage of using **tdprior** instead of **b** is that the user can also invoke **aictest** to automatically test whether significant trading day effects still occur in the irregular component of the preadjusted series and to calculate adjustment factors for removing any remaining effects. However, the fact that prior adjustment by (C.16) (or (C.17)) removes the leap year effect m_t/m_t^* (or $\exp\{-1+m_t/m_t^*\}$), makes it necessary, when **td** is specified in the **variables** argument, to modify the models (C.4) and (C.13) used by **x11regression** for estimating remaining effects. When `mode = mult`, the model

$$m_t I_t - m_t = \sum_{j=1}^{6}\alpha_j(d_{j,t}-d_{7,t}) + \boldsymbol{\beta}'\left(\mathbf{H}_t-\mathbf{H}_t^*\right) + \boldsymbol{\kappa}'\mathbf{AO}_t + \varepsilon_t$$

is used in place of (C.4), and, when `mode = logadd`,

$$m_t^* \log I_t = \sum_{j=1}^{6}\alpha_j(d_{j,t}-d_{7,t}) + \boldsymbol{\beta}'\left(\mathbf{H}_t-\mathbf{H}_t^*\right) + \boldsymbol{\kappa}'\mathbf{AO}_t + \varepsilon_t$$

instead of (C.13). The first model yields the calendar factors

$$1+\sum_{j=1}^{6}\alpha_j\left(\frac{d_{j,t}-d_{7,t}}{m_t}\right) + \boldsymbol{\beta}'\left(\frac{\mathbf{H}_t-\mathbf{H}_t^*}{m_t}\right),$$

from which combined calendar factors are formed by multiplication with (C.16). The result is approximately

$$\frac{m_t}{m_t^*}\left\{\sum_{j=1}^{7}\left(1+\alpha_j^{(p)}+\alpha_j\right)\frac{d_{j,t}}{m_t} + \boldsymbol{\beta}'\left(\frac{\mathbf{H}_t-\mathbf{H}_t^*}{m_t}\right)\right\}. \tag{C.18}$$

The second model yields the calendar factors

$$\exp\left\{\sum_{j=1}^{6}\alpha_j\left(\frac{d_{j,t}-d_{7,t}}{m_t^*}\right) + \boldsymbol{\beta}'\left(\frac{\mathbf{H}_t-\mathbf{H}_t^*}{m_t^*}\right)\right\},$$

and multiplication by (C.17) yields the combined factors

$$\exp\left\{-1+\sum_{j=1}^{6}\left(1+\alpha_j^{(p)}+\alpha_j\right)\frac{d_{j,t}}{m_t^*}\right\}. \tag{C.19}$$

The formulas (C.18) and (C.19) show that a statistically significant t-statistic for a coefficient α_j can be interpreted as meaning that the prior weight $1+\alpha_j^{(p)}$ needs significant revision.

Bibliography

Abraham, B. and G. Box (1978). Deterministic and forecast-adaptive time-dependent models. *Applied Statistics 27*, 120–130.

Abraham, B. and J. Ledolter (1983). *Statistical Methods for Forecasting*. New York: John Wiley & Sons.

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In B. Petrov and F. Csáki (Eds.), *Second International Symposium on Information Theory*, pp. 267–281. Budapest: Akadémiai Kiadó.

Akaike, H. (1980). Seasonal adjustment by a bayesian modeling. *Journal of Time Series Analysis 1*, 1–14.

Akaike, H. and M. Ishiguro (1980). BAYSEA, a bayesian seasonal adjustment program. Computer Science Monographs No. 13, Tokyo: The Institute for Statistical Mathematics.

Baxter, M. A. (1994). A guide to seasonal adjustment of monthly data with X-11 (third edition). Central Statistical Office, United Kingdom.

Bednarek, M. (2019). The dates of easter for 500 years. [Online]. Available: `http://mbednarek.byethost7.com/easter.php` [2019, August 1].

Bell, W. R. (1983). A computer program for detecting outliers in time series. *Proceedings of the American Statistical Association, Business and Economic Statistics Section*, 634–639.

Bell, W. R. (1984). Seasonal decomposition of deterministic effects. SRD Research Report No. 84/01, U. S. Census Bureau `http://www.census.gov/srd/papers/pdf/rr84-1.pdf`.

Bell, W. R. (1987). A note on overdifferencing and the equivalence of seasonal time series models with monthly means and models with (0,1,1)12 seasonal parts when $\theta = 1$. *Journal of Business and Economic Statistics 5*, 383–387. `http://www.census.gov/srd/papers/pdf/rr84-32.pdf`.

Bell, W. R. (1995). Correction to seasonal decomposition of deterministic effects. SRD Research Report No. 95/01, U. S. Census Bureau `http://www.census.gov/srd/papers/pdf/rr95-01.pdf`.

Bell, W. R. (2004). On RegComponent time series models and their applications. In A. Harvey, S. J. Koopman, and N. Shephard (Eds.), *State Space and Unobserved Component Models: Theory and Applications (Proceedings of a Conference in Honour of James Durbin)*, pp. 248–283. Cambridge: Cambridge University Press.

Bell, W. R. (2010). Unit root properties of seasonal adjustment and related filters. Research Report Statistics Number 2010-08, Research and Methodology Directorate, U. S. Census Bureau `http://www.census.gov/srd/papers/pdf/rrs2010-08.pdf`.

Bell, W. R. and S. Hillmer (1988). A matrix approach to likelihood evaluation and signal extraction for ARIMA component time series models. Research Report Number 1988/22, Statistical Research Division, U.S. Census Bureau `http://www.census.gov/srd/papers/pdf/rr88-22.pdf`.

Bell, W. R. and S. C. Hillmer (1983). Modeling time series with calendar variation. *Journal of the American Statistical Association 78*, 526–534.

Bell, W. R. and S. C. Hillmer (1984). Issues involved with the seasonal adjustment of economic time series. *Journal of Business and Economic Statistics 2*, 299–320. (invited paper with discussion).

Bell, W. R. and S. C. Hillmer (1985). A reply (to comments by Sims in Bell and Hillmer(1984)). *Journal of Business and Economic Statistics 3*, 95–97.

Berk, K. N. (1974). Consistent autoregressive spectral estimates. *Annals of Statistics 2*, 489–502.

Bobbitt, L. and M. C. Otto (1990). Effects of forcasts on the revisions of seasonally adjusted values using the X-11 seasonal adjustment procedure. *Proceedings of the American Statistical Association, Business and Economic Statistics Section*, 449–453. `http://www.census.gov/srd/papers/pdf/rr90-09.pdf`.

Box, G. and D. Cox (1964). An analysis of transformations. *Journal of Royal Statistical Society B 26*, 211–252.

Box, G. E. P. and G. M. Jenkins (1976). *Time Series Analysis: Forecasting and Control* (2nd ed.). San Francisco, CA: Holden-Day.

Box, G. E. P. and G. C. Tiao (1975). Intervention analysis with applications to economic and environmental problems. *Journal of the American Statistical Association 70*, 70–79.

Bozik, J. E. and M. C. Otto (1988). Benchmarking: Evaluating methods that preserve month-to-month changes. Research Report Number 88/07, Statistical Research Division, U.S. Census Bureau `http://www.census.gov/srd/papers/pdf/rr88-07.pdf`.

Brillinger, D. R. (1975). *Time Series–Data Analysis and Theory*. New York: Holt, Rinehart, and Winston.

Brockwell, P. J. and R. A. Davis (1991). *Time Series: Theory and Methods*. New York: Springer-Verlag.

Burnham, K. P. and D. R. Anderson (2004). Multimodal inference: Understanding AIC and BIC in model selection. *Sociological Methods & Research 33*, 261–304.

Causey, B. and M. L. Trager (1982). Derivation of solution to the benchmarking problem: Trend revision. Unpublished research notes, U.S. Census Bureau, reproduced in Appendix C of Bozik and Otto (1988) - `http://www.census.gov/srd/papers/pdf/rr88-07.pdf`.

Chang, I. and G. C. Tiao (1983). Estimation of time series parameters in the presence of outliers. Technical Report No. 8, Statistics Research Center, University of Chicago.

Chang, I., G. C. Tiao, and C. Chen (1988). Estimation of time series parameters in the presence of outliers. *Technometrics 30*, 193–204.

Chen, Z. (1985). The asymptotic efficiency of a linear procedure of estimation for ARMA models. *Journal of Time Series Analysis 6*, 52–62.

Chen, Z.-G. and P. Durk (2005). A therapy for ill-seasonality in X-11 seasonal adjustment. Statistics Canada Working Paper BSMD-2005-009E.

Cholette, P. and E. B. Dagum (1994). Benchmarking time series with autocorrelated survey errors. *International Statistical Review 62*, 367–377.

Cholette, P. A. (1978). A comparison and assessment of various adjustment methods of sub-annual series to yearly benchmarks. Research Paper, Seasonal Adjustment and Time Series Staff, Statistics Canada.

Cholette, P. A. (1984). Adjusting sub-annual series to benchmarks. *Survey Methodology 10*, 35–49.

Chow, J. and K. Moore (2009). Constructing an Easter regressor for a stock series in X-12-ARIMA. *Proceedings of the American Statistical Association, Business and Economic Statistics Section*. [CD-ROM].

Cleveland, W. P. and M. R. Grupe (1983). Modeling time series when calendar effects are present. In *Proceedings of the Conference on Applied Time Series Analysis of Economic Data*, pp. 57–73. Washington, DC: U. S. Census Bureau.

Cleveland, W. S. and S. J. Devlin (1980). Calendar effects in monthly time series: Detection by spectrum analysis and graphical methods. *Journal of the American Statistical Association 75*, 487–496.

Dagum, E. B. (1980). The X-11-ARIMA seasonal adjustment method. Statistics Canada.

Dagum, E. B. (1988). X-11-ARIMA/88 seasonal adjustment method - Foundations and users manual. Statistics Canada.

Dagum, E. B. (1996). A new method to reduce unwanted ripples and revisions in trend-cycle estimates from X-11-ARIMA. *Survey Methodology 22*, 77–83.

Dagum, E. B. and P. A. Cholette (2006). *Benchmarking, Temporal Distribution, and Reconciliation Methods for Time Series*. New York: Springer. Lecture Notes in Statistics, Vol. 186.

Dagum, E. B. and A. Luati (2009). A cascade linear filter to reduce revisions and false turning points for real time trend-cycle estimation. *Econometric Reviews 28*(1), 40–59.

Darne, O. and E. B. Dagum (2009). Performance of short-term trend predictors for current economic analysis. *Economics Bulletin*. [Online]. Available: `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.613.2813&rep=rep1&type=pdf` [2020, May 29].

den Butter, F. A. G. and M. M. G. Fase (1991). *Seasonal Adjustment as a Practical Problem*. Amsterdam: North Holland.

Doherty, M. (2001). Surrogate henderson filters in X-11. *Australian and New Zealand Journal of Statistics 43*, 385–392.

Doornik, J. A. and D. Hendry (2001). *GiveWin: An Interface to Empirical Modelling* (3rd ed.). London: Timberlake Consultants Press.

Duffet-Smith, P. (1981). *Practical Astronomy With Your Calculator* (2nd ed.). Cambridge University Press.

Feldpausch, R. M., C. C. H. Hood, and K. C. Wills (2004). Diagnostics for model-based seasonal adjustment. *Proceedings of the American Statistical Association, Business and Economic Statistics Section*. [CD-ROM] `http://www.census.gov/ts/papers/jsm2004rmf.pdf`.

Findley, D., T. S. McElroy, and K. C. Wills (2005). Modifications of SEATS' diagnostic for detecting over- and underestimation of seasonal adjustment components. `https://www.census.gov/ts/papers/findleymcelroywills2005.pdf`.

Findley, D. F. (1985). On the unbiasedness property of AIC for exact or approximating linear stochastic time series models. *Journal of Time Series Analysis 6*, 229–252.

Findley, D. F. (1999). Akaike's Information Criterion II. In S. Kotz, C. B. Read, and D. L. Banks (Eds.), *Encyclopedia of Statistical Science, Update Volume 3*, pp. 2–6. New York: John Wiley & Sons.

Findley, D. F. (2005). Asymptotic stationarity properties of out-of-sample forecast errors of misspecified regARIMA models and the optimality of GLS for one-step-ahead forecasting. *Statistica Sinica 15*(2), 447–476.

Findley, D. F. (2009). Stock series holiday regressors generated by flow series holiday regressors. SRD Research Report RRS 2009-4, U. S. Census Bureau http://www.census.gov/srd/papers/pdf/rrs2009-04.pdf.

Findley, D. F. and C. C. Hood (1999). X-12-ARIMA and its application to some Italian indicator series. In *Seasonal Adjustment Procedures – Experiences and Perspectives*, pp. 231–251. Rome: Istituto Nazionale di Statistica (ISTAT). http://www.census.gov/ts/papers/x12istat.pdf.

Findley, D. F., D. P. Lytras, and A. Maravall (2016). Illuminating ARIMA Model-Based Seasonal Adjustment with Three Fundamental Seasonal Models. pp. 1–42. http://dx.doi.org/10.1007/s13209-016-0139-4.

Findley, D. F. and D. E. K. Martin (2006). Frequency domain analysis of SEATS and X-11/X-12-ARIMA seasonal adjustment filters for short and moderate length time series. *Journal of Official Statistics 22*(1), 1–34.

Findley, D. F. and B. C. Monsell (1986). New techniques for determining if a time series can be seasonally adjusted reliably, and their application to U. S. foreign trade series. In M. Perryman (Ed.), *Regional Economic Modeling*, pp. 195–228. Amsterdam: Kluwer-Nijhoff.

Findley, D. F. and B. C. Monsell (2009). Modeling stock trading day effects under flow day-of-week constraints. *Journal of Official Statistics 25*(3), 415–430.

Findley, D. F., B. C. Monsell, W. R. Bell, M. C. Otto, and B. C. Chen (1998). New capabilities of the X-12-ARIMA seasonal adjustment program (with discussion). *Journal of Business and Economic Statistics 16*, 127–177. http://www.census.gov/ts/papers/jbes98.pdf.

Findley, D. F., B. C. Monsell, H. B. Shulman, and M. G. Pugh (1990). Sliding spans diagnostics for seasonal and related adjustments. *Journal of the American Statistical Association 85*, 345–355.

Findley, D. F. and R. J. Soukup (2000). Modeling and model selection for moving holidays. *Proceedings of the American Statistical Association, Business and Economic Statistics Section*, 102–107. http://www.census.gov/ts/papers/asa00_eas.pdf.

Findley, D. F. and C. Z. Wei (2002). AIC, overfitting principles, and the boundedness of moments of inverse matrices for autoregressions and related models. *Journal of Multivariate Analysis 83*, 415–450.

Findley, D. F., K. C. Wills, J. A. D. Aston, R. Feldpausch, and C. C. Hood (2003). Diagnostics for ARIMA model-based seasonal adjustment. *Proceedings of the American Statistical Association, Business and Economic Statistics Section*. [CD-ROM].

Fuller, W. A. (1976). *Introduction to Statistical Time Series*. New York: John Wiley & Sons.

Gastwirth, J. L. and M. G. B. Owens (1977). On classical test of normality. *Biometrika 64*, 135–139.

Geary, R. C. (1936). Moments of the ratio of mean deviation to the standard deviation as a test of normality. *Biometrika 28*, 295–305.

Geweke, J. (1978). Revision of seasonally adjusted time series. SSRI Report No. 7822, University of Wisconsin, Department of Statistics.

Gómez, V. (1998). Automatic model identification in the presence of missing observations and outliers. Working Paper D-98009, Ministerio de Economía y Hacienda, Dirección General de Análisis y Programación Presupuestaria, Madrid.

Gómez, V. and A. Maravall (1996). Programs TRAMO and SEATS : Instructions for the user (beta version: June 1997). Banco de España, Servicio de Estudios, DT 9628.

Gómez, V. and A. Maravall (2001a). Automatic modeling methods for univariate series. In D. Pena, G. C. Tiao, and R. S. Tsay (Eds.), *A Course in Time Series Analysis*. New York, NY: John Wiley & Sons.

Gómez, V. and A. Maravall (2001b). Seasonal adjustment and signal extraction in economic time series. In D. Pena, G. C. Tiao, and R. S. Tsay (Eds.), *A Course in Time Series Analysis*. New York, NY: John Wiley & Sons.

Griswold, R. E. and M. T. Griswold (1997). *The ICON Programming Language* (3rd ed.). San Jose: Peer-to-Peer Communications.

Hampel, F. R., E. M. Ronchetti, P. J. Rousseeuw, and W. A. Stahel (1986). *Robust Statistics: The Approach Based on Influence Functions*. New York, NY: John Wiley & Sons.

Hannan, E. J. and B. G. Quinn (1979). The determination of the order of an autoregression. *Journal of Royal Statistical Society B 41*, 190–195.

Hannan, E. J. and J. Rissanen (1982). Recursive estimation of mixed autoregressive-moving average models. *Biometrika 66*, 265–270.

Hillmer, S. C. and G. C. Tiao (1979). Likelihood function of stationary multiple autoregressive moving average models,. *Journal of the American Statistical Association 74*, 652–660.

Hood, C. C. (2002a). Comparing the automatic ARIMA model selection procedures of TRAMO and X-12-ARIMA Version 0.3 and the seasonal adjustments of SEATS and X-12-ARIMA. unpublished work presented at the Eurostat Working Group on Seasonal Adjustment Meeting, Luxembourg, April 2002.

Hood, C. C. (2002b). Comparison of time series characteristics for seasonal adjustments from SEATS and X-12-ARIMA. American Statistical Association, Proceedings of the Business and Economic Statistics Section [CD-ROM] `http://www.census.gov/ts/papers/choodasa2002.pdf`.

Hood, C. C. (2002c). X-12-Graph: A SAS/GRAPH program for X-12-ARIMA output, user's guide for the X-12-Graph batch for PC/Windows, version 1.2. U. S. Census Bureau, U. S. Department of Commerce.

Hood, C. C. (2005). An empirical comparison of methods for benchmarking seasonally adjusted series to annual totals. *Proceedings of the American Statistical Association, Business and Economic Statistics Section*. [CD-ROM] `https://www.census.gov/ts/papers/chood_asa2005.pdf`.

Hood, C. C., J. D. Ashley, and D. F. Findley (2000). An empirical evaluation of the performance of TRAMO/SEATS on simulated series. American Statistical Association, Proceedings of the Business and Economic Statistics Section `http://www.census.gov/ts/papers/asa00_ts.pdf`.

Huot, G. (1975). Quadratic minimization adjustment of monthly or quarterly series to annual totals. Research Paper, Seasonal Adjustment and Time Series Staff, Statistics Canada.

Hurvich, C. M. and C. Tsai (1989). Regression and time series model selection in small samples. *Biometrika 76*, 297–307.

Jenkins, G. M. and D. G. Watts (1968). *Spectral Analysis and Its Applications*. San Francisco, CA: Holden-Day.

Kaiser, R. and A. Maravall (2001). *Measuring Business-Cycles in Economic Time Series*. New York: Springer. Lecture Notes in Statistics, Vol. 154.

Klein, J. L. (1991). *Statistical Visions in Time – A History of Time Series Analysis 1662–1938*. Cambridge: Cambridge University Press.

Kohn, R. and C. F. Ansley (1985). Efficient estimation and prediction in time series regression models. *Biometrika 72*, 694–697.

Ladiray, D. and B. Quenneville (2001). *Seasonal Adjustment with the X-11 Method*. New York: Springer. Lecture Notes in Statistics, Vol. 158.

Lehman, E. L. (1986). *Testing Statistical Hypotheses* (2nd ed.). New York, NY: John Wiley & Sons.

Lin, J.-L. and T.-S. Liu (2002). Modeling lunar calendar holiday effects in Taiwan. *Taiwan Economic Forecast and Policy 33*(1), 1–37. `http://www.census.gov/ts/papers/lunar.pdf`.

Ljung, G. M. (1993). On outlier detection in time series. *Journal of Royal Statistical Society B 55*, 559–567.

Ljung, G. M. and G. E. P. Box (1978). On a measure of lack of fit in time series models. *Biometrika 65*, 297–304.

Ljung, G. M. and G. E. P. Box (1979). The likelihood function of stationary autoregressive-moving average models. *Biometrika 66*, 265–270.

Lothian, J. (1984). The identification and treatment of moving seasonality in X-11. *Proceedings of the Business and Economic Statistics Section of the American Statistical Association*, 166–171.

Lothian, J. and M. Morry (1978). A test of quality control statistics for the X-11-ARIMA seasonal adjustment program. Research Paper, Seasonal Adjustment and Time Series Staff, Statistics Canada.

Lytras, D. P. (2020a). X-13-Graph: A SAS/GRAPH Program for X-13ARIMA-SEATS Output, User's Guide for the Batch Program on the PC or Unix, Version 3.0. U. S. Census Bureau, U. S. Department of Commerce `https://www.census.gov/data/software/x13as/x13graph/x13gb.html`.

Lytras, D. P. (2020b). X-13-Graph Java documentation, Version 3.0. U. S. Census Bureau, U. S. Department of Commerce `https://www.census.gov/data/software/x13as/x13graph/x13gb-java.html`.

Lytras, D. P., R. M. Feldpausch, and W. R. Bell (2007). Determining seasonality: A comparison of diagnostics from X-12-ARIMA. *Proceedings of the International Conference on Establishment Surveys III*. (CD-ROM).

Maravall, A. (1995). Unobserved components in economic time series. In M. H. Pesaran and M. Wickens (Eds.), *The Handbook of Applied Econometrics*. Basil Blackwell.

Maravall, A. (2012). Seasonality tests and automatic model identification in TRAMO-SEATS (version: November 12, 2012). Banco de España, Servicio de Estudios.

McElroy, T. S. (2008a). Exact formulas for the Hodrick-Prescott filter. *Econometrics Journal 11*, 209–217. `http://www.census.gov/ts/papers/rrs2006-09.pdf`.

McElroy, T. S. (2008b). Matrix formulas for nonstationary signal extraction. *Econometric Theory 24*(4), 1–22.

McElroy, T. S. (2008c). Statistical properties of model-based signal extraction diagnostic tests. *Communications in Statistics: Theory and Methods 37*(4), 591–616.

McElroy, T. S. and R. Gagnon (2008). Finite sample revision variances for ARIMA model-based signal extraction. *Journal of Official Statistics 24*(3), 451–467.

Monsell, B. C. (2002). An update on the development of the X-12-ARIMA seasonal adjustment program. In *Proceedings of the 3rd International Symposium on Frontiers of Time Series Modeling*, pp. 1–11. Tokyo: Institute of Statistical Mathematics.

Monsell, B. C. (2006). Recent developments in seasonal adjustment software at the Census Bureau. Proceedings of the Eurostat Conference on Seasonality, Seasonal Adjustment and Their Implications for Short-term Analysis and Forecasting.

Montes, M. J. (1997a). Frequency of the date of Easter 1875 to 2124. `http://www.smart.net/~mmontes/freq3.html` [Online; accessed 8-December-1999].

Montes, M. J. (1997b). Frequency of the date of Easter over one 400 year Gregorian cycle. `http://www.smart.net/~mmontes/freq2.html` [Online; accessed 8-December-1999].

Montes, M. J. (2001). Calculation of the ecclesiastical calendar. `http://www.smart.net/~mmontes/ec-cal.html` [Online; accessed 31-July-2001].

Mood, A. M., F. A. Graybill, and D. C. Boes (1974). *Introduction to the Theory of Statistics, 3rd Edition*. New York: McGraw-Hill.

More, J. J., B. S. Garbow, and K. E. Hillstrom (1980). User guide for MINPACK-1. Report ANL-80-74, Argonne National Laboratory, Argonne, Illinois.

Otto, M. and W. R. Bell (1993). Detecting temporary changes in level in time series. *Proceedings of the American Statistical Association, Business and Economic Statistics Section*, 170–174.

Otto, M. C. and W. R. Bell (1990). Two issues in time series outlier detection using indicator variables. *Proceedings of the American Statistical Association, Business and Economic Statistics Section*, 182–187.

Otto, M. C., W. R. Bell, and J. P. Burman (1987). An iterative GLS approach to maximum likelihood estimation of regression models with ARIMA errors. *Proceedings of the American Statistical Association, Business and Economic Statistics Section*, 632–637. (available at `http://www.census.gov/srd/papers/pdf/rr87-34.pdf`).

Ozaki, T. (1977). On the order determination of ARIMA models. *Applied Statistics 26*, 290–301.

Pang, O. and B. C. Monsell (2016). Examining diagnostics for trading-day effects from X-13ARIMA-SEATS. Center for Statistical Research and Methodology Research Report Series Number 2016-02, Research and Methodology Directorate, U. S. Census Bureau `http://www.census.gov/srd/papers/pdf/RRS2016-02.pdf`.

Pearson, E. S. (1938). The probability integral transformation for testing goodness of fit and combining independent tests of significance. *Biometrika 30*, 134–148.

Pearson, E. S. and H. O. Hartley (1954). *Biometrika Tables for Statisticians, Volume 1*. Cambridge: Cambridge University Press.

Pierce, D. (1971). Least squares estimation in the regression model with autoregressive-moving average errors. *Biometrika 58*, 299–312.

Planas, C. and R. Depoutot (2002). Controlling revisions in ARIMA-model-based seasonal adjustment. *Journal of Time Series Analysis 23*, 193–214.

Priestley, M. (1981). *Spectral Analysis and Time Series*. London: Academic Press.

Quenneville, B., P. Cholette, G. Huot, K. Chiu, and T. DiFonzo (2004). Adjustment of seasonally adjusted series to annual totals. Research Report, Statistics Canada.

SAS Institute Inc. (1990). SAS/GRAPH Software: Reference, Version 6, First Edition, Volume 1. Cary, NC: SAS Institute.

Schwarz, G. (1978). Estimating the dimension of a model. *Annals of Statistics 6*, 461–464.

Shiskin, J., A. H. Young, and J. C. Musgrave (1967). The X-11 variant of the Census Method II seasonal adjustment program. Technical Paper No. 15, U.S. Department of Commerce, U. S. Census Bureau.

Snedecor, G. W. and W. G. Cochran (1980). *Statistical Methods* (7th ed.). Ames: The Iowa State University Press.

Soukup, R. J. and D. F. Findley (1999). On the spectrum diagnostics used by X-12-ARIMA to indicate the presence of trading day effects after modeling or adjustment. *Proceedings of the American Statistical Association, Business and Economic Statistics Section*, 144–149. `http://www.census.gov/ts/papers/rr9903s.pdf`.

Taniguchi, M. and Y. Kakizawa (2000). *Asymptotic Theory of Statistical Inference for Time Series*. New York: Springer-Verlag.

Thomson, P. and T. Ozaki (2002). Transformation and seasonal-trend decomposition. In *Proceedings of the 3rd International Symposium on Frontiers of Time Series Modeling*, pp. 197–212. Tokyo: Institute of Statistical Mathematics.

Titova, N. and B. C. Monsell (2009). Detecting stock calendar effects in U. S. Census Bureau inventory series. SRD Research Report RRS 2009-6, U. S. Census Bureau `http://www.census.gov/srd/papers/pdf/rrs2009-06.pdf`.

Vandaele, W. (1983). *Applied Time Series and Box-Jenkins Models*. New York: Academic Press.

Wikipedia (2013). Xhtml. `http://en.wikipedia.org/wiki/XHTML` [Online; accessed 2-April-2013].

Wikipedia (2019). Stock and flow. `http://en.wikipedia.org/wiki/Stock_and_flow` [Online; accessed 15-November-2019].

Wikipedia (2020). Hodrick-Prescott filter. `https://en.wikipedia.org/wiki/Hodrick%E2%80%93Prescott_filter` [Online; accessed 13-April-2020].

Wilson, G. T. (1983). The estimation for time series models, part I. Yet another algorithm for the exact likelihood of ARMA models. Technical Report No. 2528, Mathematics Research Center, University of Wisconsin-Madison.

Index