



EViews[®] 13

Getting Started

S&P Global

EViews 13 Getting Started

EViews 13 Getting Started

Copyright © 1994–2022 S&P Global Inc.
All Rights Reserved

This software product, including program code and manual, is copyrighted, and all rights are reserved by S&P Global Inc. The distribution and sale of this product are intended for the use of the original purchaser only. Except as permitted under the United States Copyright Act of 1976, no part of this product may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written permission of IHS Global Inc.

Disclaimer

The authors and S&P Global Inc. assume no responsibility for any errors that may appear in this manual or the EViews program. The user assumes all responsibility for the selection of the program to achieve intended results, and for the installation, use, and results obtained from the program.

Trademarks

EViews® is a registered trademark of S&P Global Inc. Windows, Excel, PowerPoint, and Access are registered trademarks of Microsoft Corporation. PostScript is a trademark of Adobe Corporation. Bloomberg is a trademark of Bloomberg Finance L.P. All other product names mentioned in this manual may be trademarks or registered trademarks of their respective companies.

Third Party Licenses

This section contains third party notices or additional terms and conditions applicable to certain software technologies which may be used in one or more EViews products and/or services. Please be sure to consult the individual product files, about box, and/or install or manual documentation for specific copyright notices and author attributions. Notices on this page are current for EViews products released on or after October 1, 2017.

- diff template Library - Copyright © 2015 Tatsuhiko Kubo cubicdaiya@gmail.com. All rights reserved.
- FFmpeg Library - FFmpeg is a trademark of Fabrice Bellard, originator of the FFmpeg project.
- GZipHelper - Copyright © 1995-2002 Gao Dasheng dsgao@hotmail.com.
- jsonCPP Library - Copyright © 2007-2010 Baptiste Lepilleur and The JsonCPP Authors.
- openssl Library - Copyright © 1998-2016 The OpenSSL Project. All rights reserved.
- libcurl Library - Copyright © 1996-2013, Daniel Stenberg daniel@haxx.se.

-
- libharu Library - Copyright © 2000-2006 Takeshi Kanno, Copyright © 2007-2009 Antony Dovgal et all.
 - libssh2 Library - Copyright © 2004-2007 Sara Golemon sarag@libssh2.org, Copyright © 2005, 2006 Mikhail Gusarov dottedmag@dottedmag.net, Copyright © 2006-2007 The Written Word, Inc., Copyright © 2007 Eli Fant elifantu@mail.ru, Copyright © 2009 Daniel Stenberg, Copyright © 2008, 2009 Simon Josefsson. All rights reserved.
 - OpenXLSX Library Copyright © 2020, Kenneth Troidal Balslev All rights reserved.
 - rapidjson Library - Copyright © 2015 THL A29 Limited, a Tencent company, and Milo Yip.
 - shapelib Library - Copyright © 1998 Frank Warmerdam.
 - ssleay License - Copyright © 1995-1998 Eric Young (eay@cryptsoft.com) All rights reserved.
 - Tableau Data Extract API - Copyright © 2003-2017 Tableau and its licensors. All rights reserved.
 - Tramo/Seats - Copyright (c) 1996 Agustin Maravall and Victor Gomez. Windows version developed by G. Caporello and A. Maravall (Bank of Spain)
 - X11.2 and X12-ARIMA version 0.2.7 and X-13ARIMA-SEATS - Copyright (c) U.S. Census Bureau.
 - zlib Data Compression Library - Copyright © 1995-2017 Jean-loup Gailly and Mark Adler.

Notices, terms and conditions pertaining to third party software are located at <http://www.eviews.com/thirdparty> and incorporated by reference herein.

S&P Global Inc.
3030 Old Ranch Parkway
2nd Floor
Suite 260
Seal Beach, CA 90740
Telephone: (949) 856-3368
Fax: (949) 856-2044
e-mail: sales@eviews.com
web: www.eviews.com
August 23, 2022

Table of Contents

GETTING STARTED	1
Installing EViews	1
Installing the Program	1
Registering EViews	2
What is Registration?	2
How Do I Register?	2
Frequently Asked Questions about Registration	5
Updating Your Copy of EViews	6
Where to Go For Help	7
Online Help	7
The Help System	7
The EViews Manuals (PDF Files)	7
Tutorials	8
The EViews Forum	8
NEW FEATURES IN EIEWS 13	9
General EViews Interface	11
New Panes and Tabs User Interface	11
New Window Options	14
Program Debugging	15
Program Dependency Logging	17
Jupyter Notebook Support	18
Data Handling	19
Daily Data Seasonal Adjustment	19
New Excel File Writing Engine	21
Holiday Functions	21
Updated X-13 Engine	25
SDMX Databases	25
Trading Economics Databases	28
World Health Organization Databases	33
Simplified Matrix Data Access	37
New Matrix Import/Export Engine	40

Matrix Row and Column Labels	43
Graphs and Tables	45
Graph Line and Shade Transparency	45
Custom Graph Data Labels	49
Customizable Geomap Labels	52
High-Low-Median Colormap Preset	55
Table Search	58
Conditional Table Cells	60
Fixing Rows and Columns in Tables	61
Econometrics and Statistics	64
ARDL Estimation	64
Pool Mean Group (PMG) Estimation	66
Difference-in-Difference Estimation	68
Enhanced VEC Estimation	70
Bayesian Time-varying Coefficient Vector Autoregression	73
Cointegration Testing	76
Diagnostics in ARDL	77
Diagnostics in Panel ARDL/PMG	86
Enhanced Impulse Response Display	90
Forecast Sample Setting Commands	98
Command Language	100
Updated Command List	100
Updated Object List	100
Updated Function List	109
EViews 12 Compatibility Notes	110

Getting Started

Congratulations on your purchase of EViews 13, the premier forecasting and analysis package for Windows-based computers. This guide will lead you step-by-step through the installation and registration procedure for EViews.

(The following discussion describes the installation and registration process for single user copies of EViews and seat licenses purchased under a Volume License Program. Setting up machines to use concurrent use licenses will require a different procedure; for details, please check with your IT support department.)

Installing EViews

Installing the Program

To begin installation, simply click on the “EViews13Installer(64-bit).exe” executable program file.

- First, you will be prompted to read and accept the License Agreement, and to designate a directory into which you wish to install your copy of EViews. If you wish to change the default installation directory, click on **Browse** and navigate to the desired directory. Click on **Next** to continue.
- Next, you will be asked to enter a name and serial number. You should have been provided with a 24-character serial number as part of your purchase. Those of you who have obtained your copy of EViews as part of a Volume License agreement should obtain a serial number from your license administrator. Enter the serial number and your name as you wish it to appear in your copy of EViews, and click on **Next**.
- Select the components you wish to install and click on **Next**.
- Lastly, you will be asked about setting up a Start Menu folder containing shortcuts to the EViews example files folder and the EViews program executable. Clicking on **Next** starts the actual installation of files onto your computer.

You should note that as part of the installation procedure, EViews will prompt you to register files with the extensions “.WF1”, “.WF2”, “.PRG”, “.EDB”, “.AIPZ”, and “.UIPZ”.

If these extensions are already registered, possibly by an earlier version of EViews, you will be prompted to allow EViews 13 to override the existing registration. Registering the extensions is not required, but doing so will allow you to double-click on files with these extensions to launch EViews.

Once the installation procedure is completed, click on **Finish**. If you have elected to create an EViews shortcut, the EViews Start Menu folder will open. To launch EViews, double-click the EViews 12 icon. Subsequently, you may launch EViews using the shortcut on your desktop or by selecting EViews from the Start Menu shortcuts, if present, by double-clicking on EViews registered file types, or by navigating to the EViews installation directory and double-clicking on the EViews icon.

Registering EViews

What is Registration?

To use EViews 13 on a specific computer, you must first register the program using a serial number. EViews registration is the one-time process of assigning a serial number to a specific machine, sending a unique machine ID number to S&P Global Inc., and writing some information to your Windows registry or Mac application support directory. This is a simple process that can be performed in a few seconds.

Under the terms of the EViews Volume License agreement, “13C” (volume) license serial numbers may not be used to register multiple machines. Each volume licensed machine running EViews must be assigned a distinct serial number. Thus, licensing an office computer, home computer and laptop computer of a single user will require three distinct Volume License serial numbers.

The copy of EViews may be uninstalled and reinstalled on a registered machine, updated, or moved to a different directory without re-registering the copy for that machine. In the special case where a machine’s hard disk is wiped clean, but no other changes are made to the system, you may simply re-register your copy of EViews. Note that in this circumstance, reregistration on the machine will *not* count as an additional registration.

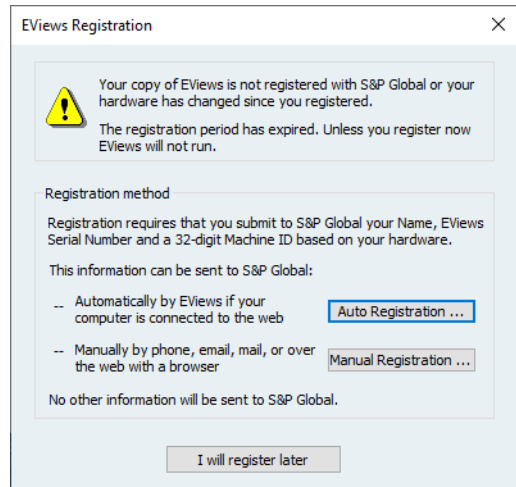
If an entire machine or a machine’s hard disk is replaced, you should contact our office to unregister your previous installation prior to re-registering.

How Do I Register?

Before starting the registration process, you should first locate your EViews serial number. You most likely will need to enter this number into EViews during the registration procedure.

If the copy of EViews is not registered, EViews will display a warning dialog. The dialog will inform you that EViews is not registered for this machine and, *if applicable*, will indicate the number of additional days the unregistered copy will continue to run.

On a Windows machine, if the copy of EViews is not registered, EViews will display a warning dialog. The dialog will inform you that EViews is not registered for this machine and, *if applicable*, will indicate the number of additional days the unregistered copy will continue to run.



You may choose to register in one of two ways: you may use the EViews auto registration features (by clicking on **Auto Registration...**), or you can manually register (by clicking on **Manual Registration...**). Selecting either of these two options will open a dialog prompting you for additional information.

Auto Registration

If your computer is connected to the Internet, auto registration makes registering EViews a snap. Simply click on the **Auto Registration...** button to display a dialog for entering your registration information.

EViews will fill out as many fields in this dialog as possible. If you wish to continue with the auto registration process, make sure that the entries in the **Serial #** and **Name** fields are filled in with the relevant information. When you click on the **Register now** button, EViews will attempt to contact one of our registration servers and, if successful, will transmit the information contained in the dialog to the server. The server will process the information and the machine will be registered to run EViews.

You should see a message indicating that registration was completed successfully, along with the number of machines that have been registered to the serial number.

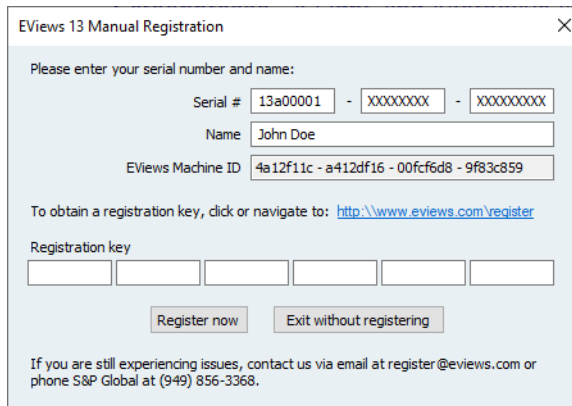
If you do not wish to continue with auto registration, click on the **Exit without registering** button and you will be returned to the main registration screen.

Note that there are some circumstances in which auto registration will fail. Obviously, auto registration will not work if the computer is not connected to the Internet. If registration fails, you should first verify that you have Internet access. Second, your computer may be behind a firewall which does not allow the required communication between your computer and our servers. Furthermore, while unlikely, it is possible that all of our registration servers are temporarily unresponsive.

If you continue to have problems with auto registration, you can choose to register manually as described in the next section, or you can contact us for assistance.

Manual Registration

If auto registration fails or if you prefer not to use the automatic registration features, you may elect to register manually. From the main registration page, click on **Manual Registration...** to display the manual registration portion of the dialog:



The screenshot shows a dialog box titled "EViews 13 Manual Registration" with a close button (X) in the top right corner. The dialog contains the following fields and text:

- Text: "Please enter your serial number and name:"
- Serial #: A text box containing "13a00001", followed by a hyphen, a text box containing "XXXXXXXX", another hyphen, and a text box containing "XXXXXXXX".
- Name: A text box containing "John Doe".
- EViews Machine ID: A text box containing "4a12f11c - a412df16 - 00fcf6d8 - 9f83c859".
- Text: "To obtain a registration key, click or navigate to: <http://www.eviews.com/register>"
- Registration key: A row of six empty text boxes.
- Buttons: "Register now" and "Exit without registering".
- Text: "If you are still experiencing issues, contact us via email at register@eviews.com or phone S&P Global at (949) 856-3368."

You must fill in the three fields in the dialog: the 24-character serial number, your name, and a 36-character registration key you must first obtain via online, by phone, or by email. EViews will help you by filling in as many fields as possible.

The easiest method of retrieving the registration key is online. Go to

<http://www.eviews.com/register/>

which will direct you to our registration servers. Follow the links to the registration page, and fill in the form. Enter your name, serial number, and the machine ID number as dis-

played in this registration dialog into the form. Click on the **Submit the form** button. You will be provided with the 36-character registration key.

Once you have obtained the key, enter it into the registration dialog. If necessary, select Help/EViews Registration... from the EViews main menu to display the registration page.

Make certain that the listed name, serial number, and registration key in the registration dialog matches exactly (this includes spaces and case) that of the registration confirmation page. Click **OK** to finish the registration process. Note that you should be able to copy-and-paste the registration key information from your browser into the dialog edit fields..

If all of the information is entered correctly, you will be informed that your registration is complete.

If you do not have access to a working web browser, you can contact our office via email or phone to obtain the key:

S&P Global Inc.
Attn: Registration
3030 Old Ranch Parkway, Suite 260
Seal Beach, CA 90740
Email: register@eviews.com
Phone: 949-856-3368

Please provide a registration name, full 24-character serial number, and the machine ID number. We will then provide you with the 36-character registration key.

If you receive the key via email, you should be able to copy-and-paste the key information into the dialog edit fields.

Contact Information

Once registration is completed, EViews will display an optional contact page form. You may submit this form to send name, address, phone number, and email information to S&P Global Inc. This information is for our records only and will not be redistributed to others.

Frequently Asked Questions about Registration

While the registration procedure should be straightforward, we understand that you may still have questions. The following are answers to the most frequently asked questions:

- *How do I find my serial number and other information about my copy of EViews?*

Your copy of EViews contains information about your registration status, as well as the product version and build date of the program. To obtain this information, simply select **Help/About EViews** from the main EViews menu.

- *I contacted you and received a key, but the key doesn't seem to work. What could be wrong?*

The most common registration problem results from entering a name or serial number which does not match the key. You should make certain that the name and serial number both match those provided when obtaining a key. Note that while the name is not case-sensitive, it should otherwise be entered *exactly* as originally provided. If you still experience problems, please contact our office.

- *My copy of EViews does not appear to have the features for the edition that I purchased. Do I need to purchase a new copy?*

No. Simply contact our office. Once we verify the edition of EViews that you have purchased, you should be able to re-register and upgrade your copy to enable the features.

- *I've replaced my computers and no longer have available registrations. What should I do?*

If there are special circumstances where you need to register an additional machine, please contact our office.

- *How do I change the name in which my copy is registered?*

Your copy of EViews contains the name in which it was first registered. If you wish to change the registration name, please contact our office.

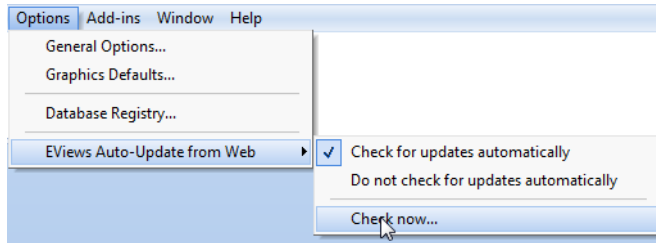
- *What if I have trouble registering?*

We do not anticipate that you will have problems registering your copy of EViews using one of the available methods (auto-registration, manual using our web servers, or manual using email or phone). Please feel free to contact our office if you encounter difficulties.

Updating Your Copy of EViews

EViews 12 offers an automatic updating feature that can check for new updates every day, and install an updated version if available. (The automatic update feature can be enabled or disabled from the **Options/EViews Auto-Update from Web** menu item.)

Alternately, you may manually check for updates from within EViews at any time by selecting **Check now...** under the **EViews Auto-Update from Web** menu item, or by selecting **EViews Update** from the **Help** menu.



You may also visit the EViews website to check for updates to the EViews program and other components (documentation, sample data, and sample programs). Go to:

<http://www.eviews.com>

and navigate to the downloads area. Downloading updates *will not* require re-registration of EViews on any previously registered computer. Simply download the update, run the installer, and you will have the latest shipping copy of your software.

Where to Go For Help

Your EViews installation includes documentation in the form of an interactive Help System and PDF versions of the manuals. In addition, online documentation and user-provided support are available.

Online Help

You may also access the EViews documentation online at

<https://help.eviews.com>

The Help System

All of the EViews documentation may be viewed from within EViews using the help system. To access the EViews help system, go to the main menu and select **Help/EViews Help Topics...** or click on **Help/Quick Help Reference** and select a topic to jump directly to relevant subsections.

The EViews Manuals (PDF Files)

Your EViews installation includes copies of the EViews manuals in Adobe Portable Document Format (.PDF) file format. You may access the PDF files from within EViews by clicking on **Help** in the main EViews menu and selecting the file of interest. Alternately, you may navigate to the “Docs” subdirectory of your EViews installation directory to access the files directly.

Tutorials

To get you started, we have provided a set of PowerPoint tutorials illustrating the basics of EViews. These tutorials are a great way to see EViews in action.

<http://www.eviews.com/Learning/index.html>

The EViews Forum

User-provided online support is available via the EViews Forum.

To supplement the information provided in the manuals and the help system, we encourage you to visit the EViews Online Forum, where you can find answers to common questions about installing, using, and getting the most out of EViews. The EViews Forum is an ideal place to ask questions of and share information with other EViews users.

The forum address is:

<https://forums.eviews.com>

New Features in EViews 13

EViews 13 features a number of exciting changes and improvements. The following is an overview of the most important new features in Version 13.

Note that in some cases, entries will appear in more than one section as they might otherwise be overlooked by those who may find them to be of interest.

General EViews Interface

- Alternative graphical user interface ([“New Panes and Tabs User Interface,” on page 11](#)).
- Dark mode and window positioning options ([“New Window Options,” on page 14](#)).
- Debugging tools for EViews programs ([“Program Debugging” on page 15](#)).
- Program dependency tracking ([“Program Dependency Logging” on page 17](#)).
- Jupyter Notebook Support ([“Jupyter Notebook Support” on page 18](#)).

Data Handling

Data Handling

- Daily data seasonal adjustment ([“Daily Data Seasonal Adjustment” on page 19](#)).
- New and improved Excel file writing engine ([“New Excel File Writing Engine” on page 21](#)).
- Enhanced holiday family of functions to return the proportion of an annual event associated with each observation ([“Holiday Functions” on page 21](#)).
- Updated X-13 Engine (featuring HTML output) ([“Updated X-13 Engine” on page 25](#)).

Data Sources and File Formats

- Australian Bureau of Statistics SDMX ([“SDMX Databases,” on page 25](#)).
- Deutsche Bundesbank SDMX ([“SDMX Databases,” on page 25](#)).
- Insee SDMX ([“SDMX Databases,” on page 25](#)).
- Trading Economics ([“Trading Economics Databases,” on page 28](#)).
- World Health Organization ([“World Health Organization Databases,” on page 33](#)).

Matrix Language

- Improved data access to matrix object data ([“Simplified Matrix Data Access,” on page 37](#)).

- Improved data import and export from matrix objects ([“New Matrix Import/Export Engine,”](#) on page 40).
- Expanded support for setting and using row and column labeling ([“Matrix Row and Column Labels,”](#) on page 43).

Graphs and Tables

- Line and shade transparency in graphs ([“Graph Line and Shade Transparency,”](#) on page 45).
- Custom graph data labels ([“Custom Graph Data Labels,”](#) on page 49).
- Customizable Geomap labels ([“Customizable Geomap Labels,”](#) on page 52).
- High-low-median colormap preset ([“High-Low-Median Colormap Preset,”](#) on page 55).
- Searching cells within tables ([“Table Search,”](#) on page 58).
- Conditional table cell expressions ([“Conditional Table Cells,”](#) on page 60).
- Fixed row and column display in tables ([“Fixing Rows and Columns in Tables,”](#) on page 61).

Econometrics and Statistics

Estimation and Analysis

- Enhanced Autoregressive Distributed Lag (ARDL) support featuring estimation of Nonlinear ARDL models and new diagnostics ([Chapter 29. “Linear and Nonlinear ARDL”](#)).
- Improved Pool Mean Group (PMG) estimation featuring expanded deterministic trend support, estimation with Nonlinear ARDL terms, and new diagnostics ([“Pool Mean Group \(PMG\) Estimation”](#) on page 66).
- Difference-in-Difference (DID) estimation and diagnostics ([“Difference-in-Difference Estimation”](#) on page 68).
- Enhanced Vector Error Correction (VEC) estimation, featuring improved support for deterministic regressors ([“Enhanced VEC Estimation”](#) on page 70).
- Bayesian Time-varying Coefficient Vector Autoregression (BTVCVAR) models ([“Bayesian Time-varying Coefficient Vector Autoregression”](#) on page 73).

Testing and Diagnostics

- Improved cointegration testing, featuring improved support for deterministic regressors ([“Cointegration Testing”](#) on page 76).
- New diagnostics in ARDL equations ([“Diagnostics in ARDL”](#) on page 77).

- New diagnostics in panel ARDL/PMG equations ([“Diagnostics in Panel ARDL/PMG” on page 86](#)).
- Extended VAR/VEC impulse response confidence interval calculation and display ([“Enhanced Impulse Response Display” on page 90](#)).

Command Language

- List of new and updated global commands ([“Updated Command List” on page 100](#)).
- List of new and updated object commands and data members ([“Updated Object List” on page 100](#)).
- List of new and updated functions ([“Updated Function List” on page 109](#)).

EViews 12 Compatibility Notes

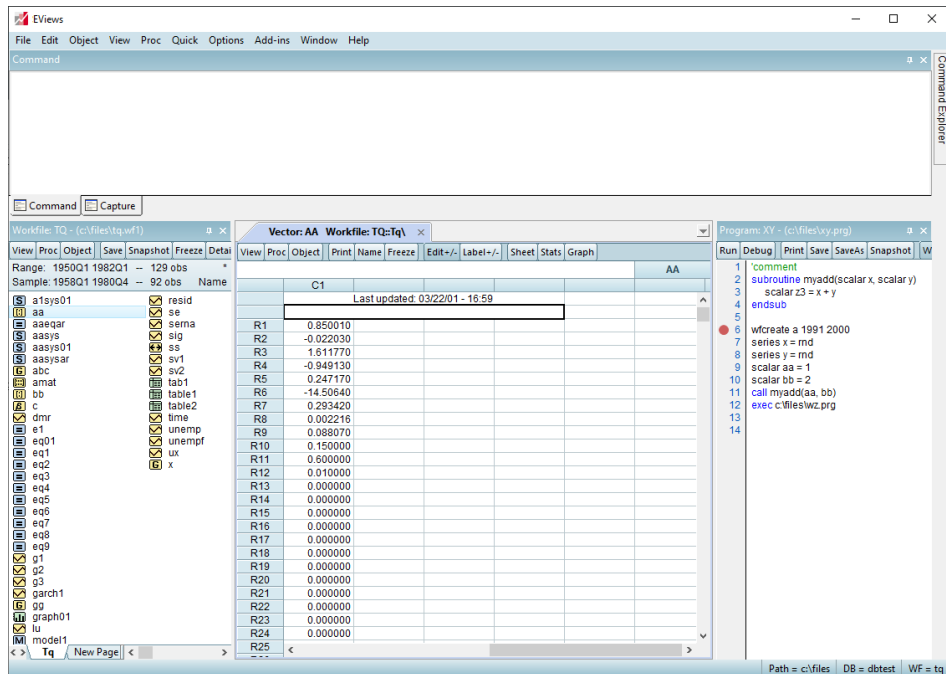
- Compatibility notes for users of EViews 12 and earlier ([“EViews 12 Compatibility Notes” on page 110](#)).

General EViews Interface

New Panes and Tabs User Interface

The familiar EViews multiple window interface offers users many advantages, especially on large computer screen displays. In some small screen settings, however, it can be more difficult to utilize fully the advantages of having multiple windows open at the same time.

EViews 13 offers an alternative user interface mode that employs panes and tabs in places of multiple windows. To maximize the use of space on smaller screens, the pane and tab mode offers a more organized way of displaying and grouping windows. The built-in organization properties of this interface may be ideally suited to smaller display environments.



In the pane and tab UI mode, different types of windows will appear in docked or undocked panes inside and outside the EViews frame:

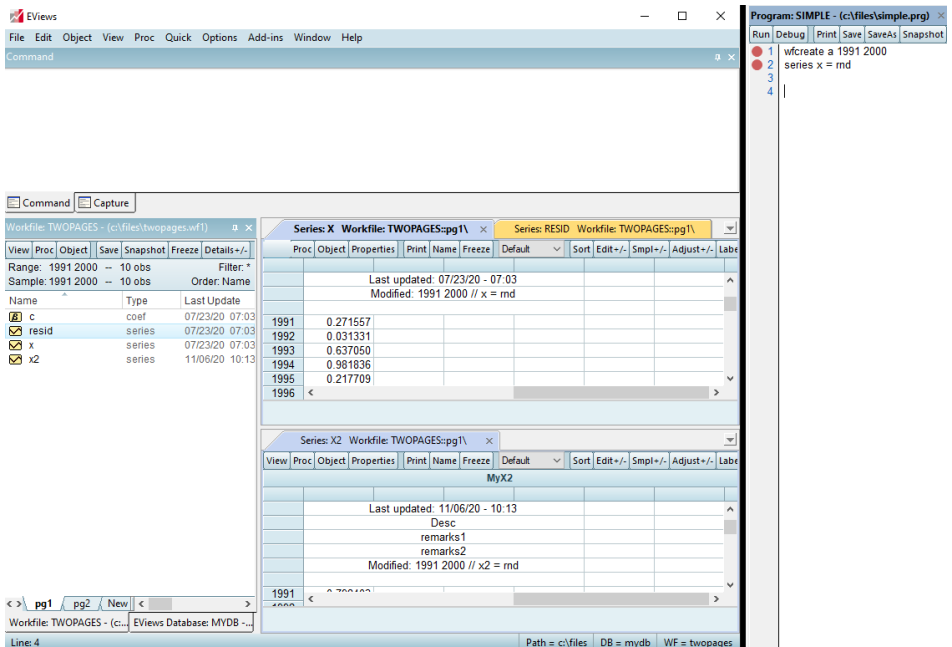
- Workfiles, programs, and database windows appear as panes that can be docked in different locations, grouped together, toggled to automatically hide or display, and float outside the EViews frame
- Object windows will appear as tabbed windows using all available client area inside the EViews frame. Multiple object windows appear as different tabs, and these tabs can be divided and stacked separately to view multiple objects simultaneously.

When you open multiple workfiles or multiple programs, previously opened windows will appear as tabs in the corresponding docks. You may bring focus to a specific window by clicking on the corresponding tab.

When you open multiple object windows, the previously opened windows will appear as tabs in the object pane. You may bring focus to an object by clicking on the down arrow at the upper right of the object dock and selecting the desired object, or by selecting **Window** from the main EViews menu and clicking on the object name in the list of opened objects.

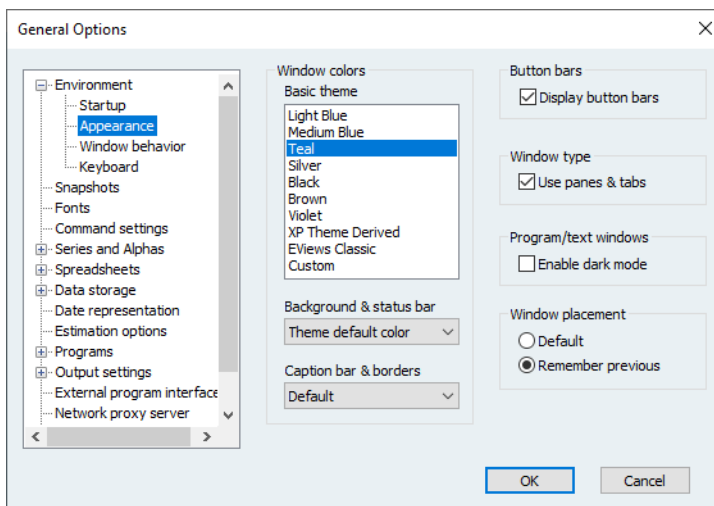
To further maximize screen real estate, you may place the workfile/database and the program panes in drawers. Placing a pane in a drawer temporarily hides the pane while retaining quick, on-demand access to the pane:

- Click on the pin icon at the top of a docked pane to hide it in a drawer on the side of the window. The pane window will close and be replaced by a drawer label on the side of the window.
- Hovering the cursor over the drawer label will open the drawer and display the pane window. Clicking away from the pane will close the drawer.
- Click again on the pin to remove the pane from the drawer and open the docked pane window.



In the example screenshot, a workfile window and a database window are grouped together into a single pane (docked to the left) and the program window is floating outside the EViews frame on the right. Three object windows are also open in the center, with the X and RESID object windows grouped together and the X2 window stacked below them.

To enable or disable the pane and tab mode, click on **General Options/Environment/Appearance** and select the **Use panes & tabs** checkbox to enable the new mode:



New Window Options

The **General Options/Environment/Appearance** dialog offers new settings to change the background of text and program windows, and to remember window placement between EViews sessions.

Dark mode

You may use the **Enable dark mode** checkbox to control the background in program and text windows.

```

Program: ROLLFCST1 - (c:\temp\rollfcst1.prg)
Run Debug Print Save SaveAs Snapshot Wrap+/- LineNum+/- Encrypt
1 'rolling forecast, re-estimating coeffs
2
3 'create workfile
4 wfccreate rollfcst q 1968 1982
5
6 'fill data
7 series y
8 y.fill
9 3.733,3.567,3.533,3.400,3.400,3.433,3.576,3.567,4.167,4.733,5.167,5.867,5.900,5.900,6.
10 033,5.967,5.767,5.633,5.600,5.333,4.933,4.867,4.833,4.800,5.033,5.133,5.633,6.667,8.1
11 33,8.767,8.600,8.433,7.633,7.467,7.833,7.900,7.467,7.100,6.900,6.633,6.200,6.000,5.96
12 7,5.833,5.733,5.767,5.800,5.867,6.133,7.500,7.633,7.500,7.333,7.400,7.333,8.367,8.767,
13 9.367,9.500,10.53
14
15
16 'set window size
17 lwindow = 20
18
19 'get size of workfile
20 llength = @obsrange
21
22 'declare equation for estimation
23 equation eq1

```

Remember window placement

The layout positions of primary windows (workfiles, programs, and database windows) can now be remembered between EViews sessions. This is especially useful in Pane & Tab mode (“[New Panes and Tabs User Interface](#)” on page 11) since primary windows may be positioned outside the EViews frame in that mode.

To remember positions, simply select the **Remember positions** radio button.

For each primary file, layout settings are stored in a hidden text file using the original name, but with the extension “.evsettings”. If you share both the primary file and the layout file with others, the original layout will be restored upon opening the primary file.

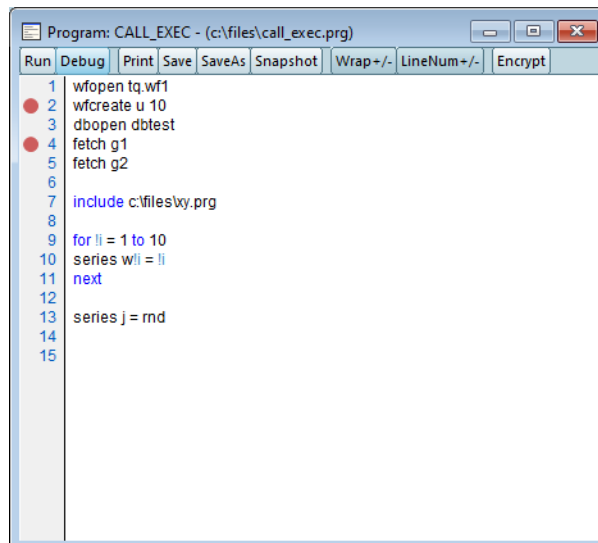
Note that when restoring window positions, complicated layouts such as panes grouped together may not appear exactly as before, especially if the layout is dependent on the existence of other panes.

Program Debugging

EViews offers tools for debugging to help you locate the source of problems. These tools allow you to run programs until they hit breakpoints on specific lines and then examine the state of your workfile at those breakpoints.

Setting and Clearing Breakpoints

Open the EViews program file and set the breakpoint by clicking to the left of the desired line. A red dot will appear:



Clicking on a red dot clears that breakpoint.

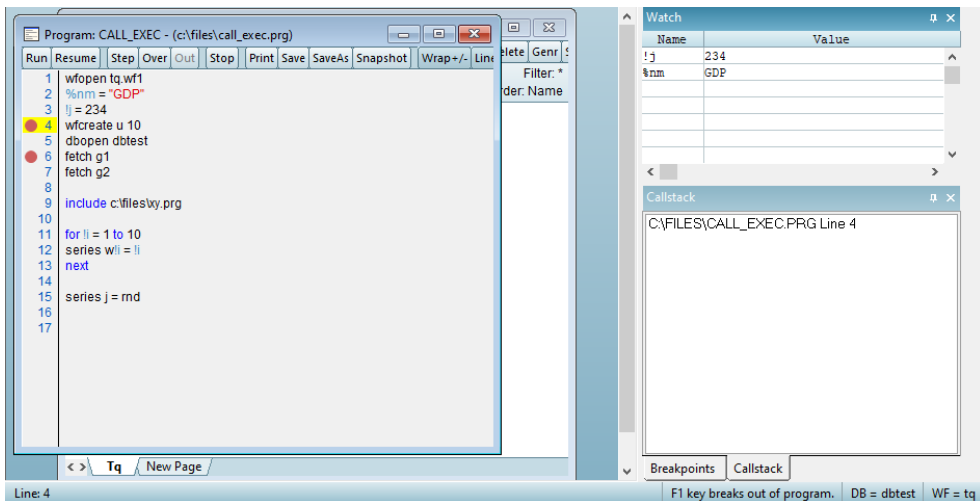
Starting a Debugging Session

To begin debugging the program, click the **Debug** button in the toolbar and enter any program arguments, choose whether to **Log dependencies**, and if desired change the **Maximum errors before halting**. Click on **OK**.

EViews will start program execution and open the debugging pane. There are three tabs in the pane: **Breakpoints**, **Watch**, and **Callstack**.

Stopping at a Breakpoint

When the program reaches an active breakpoint, the execution will pause at the red dot, now highlighted in yellow:



At this point you can look at the **Breakpoints**, **Callstack**, or **Watch** windows for more information.

- **Breakpoints** shows all defined breakpoints. Toggling the checkbox next to a breakpoint name inactivates or reactivates it.
- **Callstack** displays the current line of the active files and subroutines of the program.
- **Watch** presents the values of program and replacement variables.

Additionally, you may open EViews objects such as series or equations to examine their current states.

Button Bar

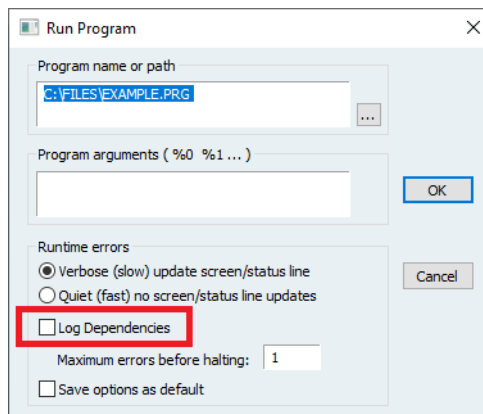
Resume continues program execution until the next breakpoint is reached. **Step** executes the current line. **Run** continues program execution to completion, ignoring any remaining breakpoints. **Stop** cancels program execution.

Restrictions during Debugging

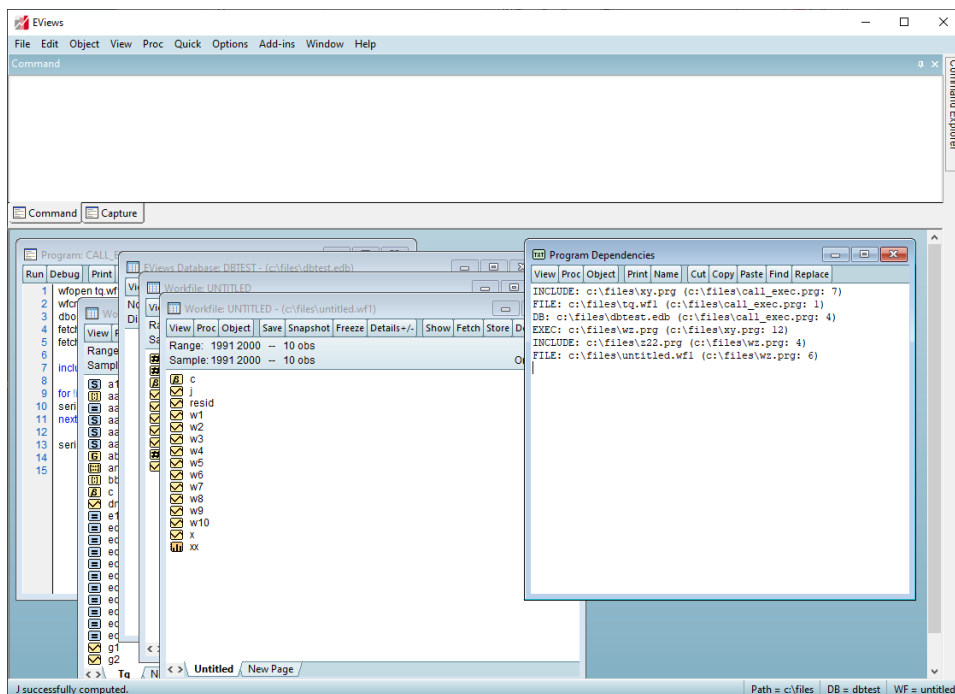
During a debugging session, you will not be able to close any windows opened by the program since this could negatively affect its execution.

Program Dependency Logging

EViews allows you to automatically log a program's external dependencies: files (as work-files, Excel files, CSV files, etc), databases, and other programs (as includes and execs). To use it, check the **Log Dependencies** checkbox in the **Run Program** dialog:



A new **Program Dependencies** window will appear showing the type, filename, and path of all the external dependencies detected during the run.



The above image shows the dependence on the external program files “xy.prg,” “wz.prg,” and “z22.prg,” the workfiles “tq.wf1” and “untitled.wf1,” and the database “dbtest.edb,” along with the line numbers producing these dependencies.

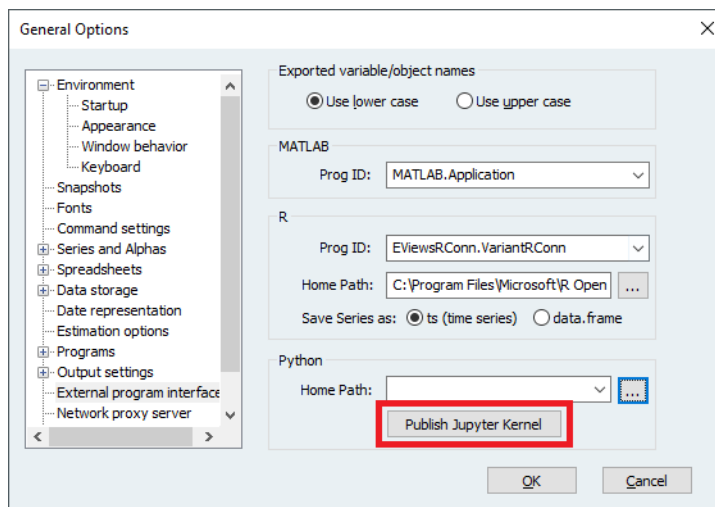
Jupyter Notebook Support

Jupyter is a popular free, open-source, web-based interactive development environment that allows users to create notebooks for documenting computational workflow. A notebook consists of an ordered series of input and output cells for the organization of code, explanatory text, and multimedia in a single document. The chronological, narrative record-keeping a notebook allows for are useful for analysis, reporting, teaching, documentation, and many other purposes.

The code in a notebook cell is passed to a programming language specific “kernel” on the backend that does the code execution. EViews, starting with version 13, can be used as a Jupyter kernel. Users can run an EViews program and display its results from within a notebook.

Using the EViews Jupyter Kernel

After Python and Jupyter have been installed (<https://jupyter.org/>), make the EViews kernel available to Jupyter by going to the main EViews menu and selecting **Options/General Options/External program interface**. Click the **Publish Jupyter Notebook** button.



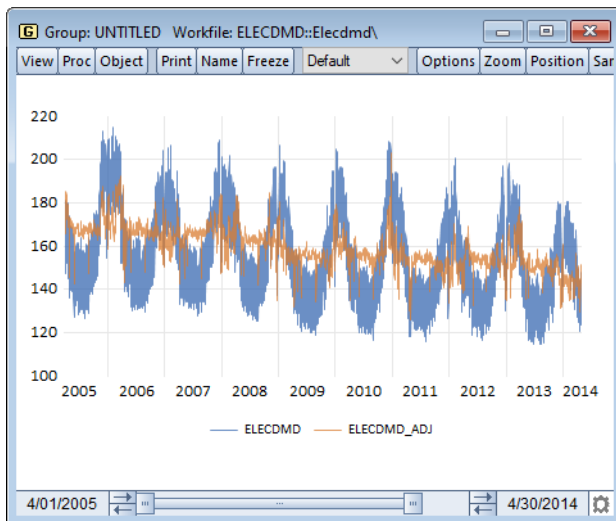
A new EViews-specific folder will be created in the Jupyter kernels folder location, usually found at “%AppData%/Roaming/jupyter/kernels”.

Next, run the notebook server. Depending on your installation, this can be done from a command prompt or the start menu in Windows or from Anaconda Navigator. Select the EViews kernel by choosing it from the “New” drop-down menu in the upper-right corner of the notebook dashboard. This will open the notebook editor, the interface where code and other input is entered and evaluated. Type EViews commands into the cells and run them with shift-enter (to run the current cell and select the following cell) or ctrl-enter (to run the current cell). More information on using Jupyter can be found in the Jupyter documentation (<https://jupyter-notebook.readthedocs.io/en/stable/notebook.html>).

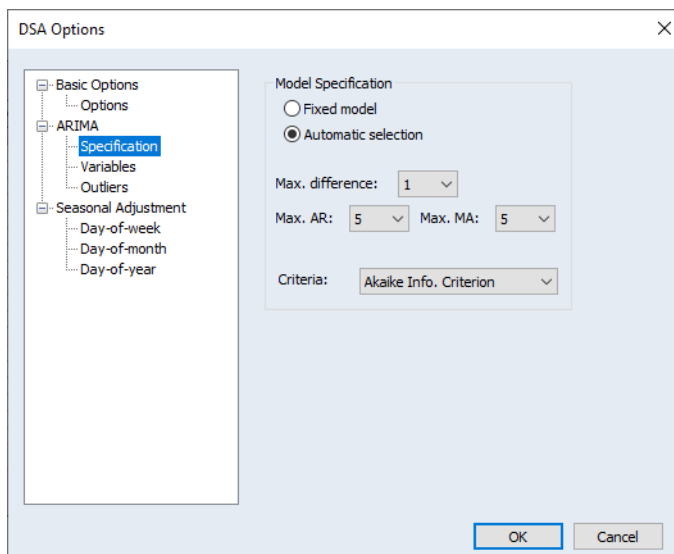
Data Handling

Daily Data Seasonal Adjustment

EViews 13 adds the ability to perform seasonal adjustment on daily data, by featuring an extended implementation of the seasonal adjustment of daily time series algorithm of Ollech (2021). Although the original Ollech (2021) algorithm is designed for 7-day week daily data, EViews’ implementation handles both 7 and 5-day week daily data.



To perform DSA seasonal adjustment in EViews, open the series and select **Proc/Seasonal Adjustment/DSA Daily Seasonal Adjustment...** EViews will then open a tree-structured DSA dialog to allow you to set the options for the DSA procedure:



The branches of the tree, on the left, allow you to specify the **Basic Options**, the **ARIMA** model, and the three STL **Seasonal Adjustment** components. Click on the node name in the left to select the node.

- For additional discussion of the new DSA procedure, see “[Daily Seasonal Adjustment](#),” on page 528 in *User’s Guide I*.
- See [Series::dsa](#) (p. 710) for command documentation in the *Object Reference*.

New Excel File Writing Engine

Previous versions of EViews used the Excel application itself to write Excel 2007 files (.XLSX) so that exporting data from an EViews workfile to an Excel 2007 file required that Excel be installed on your machine. Using Excel to write these files was convenient, but importantly came with performance and capability limitations.

EViews 13 introduces a new Excel 2007 writing engine that no longer requires installation of the Excel application. Notably, with Excel 2007 writing,

- exporting EViews data to Excel is now much more efficient, especially when performing a large number of writes,
- you may now write data into an existing Excel file (via command), an operation that was not permitted in earlier versions,
- you may now choose between using EViews cell formatting, retaining the pre-existing (in Excel) formats, or removing all formats (colors, font, number formatting, column widths and row heights) for the written range.
- For the most part, using the new engine will be transparent to users for previous versions (apart from the performance improvements). Simply perform your Excel 2007 writes in the usual fashion using the option “mode = update”. The command,

```
tab1.save(t=excelxml, cellfmt=EViews, mode=update) mytable
        range=Country!b5
```

will add the contents of table TAB1 to the preexisting “MyTable.XLSX” Excel file. The data will be written to the ‘Country’ sheet at cell B5. All preexisting cell formatting in the Excel file will be overwritten using the cell colors and fonts in TAB1.

- See [pagesave](#) (p. 483) and [wfsave](#) (p. 583) in the *Command and Programming Reference* for updated command documentation.
- See also the object specific save and export commands, [Table::save](#) (p. 992), [Matrix::export](#) (p. 517), [Vector::export](#) (p. 1123), [Sym::export](#) (p. 912), [Rowvector::export](#) (p. 648), and [Coef::export](#) (p. 25), in the *Object Reference*, for updated documentation.

Holiday Functions

The `@holiday` and `@holidayset` functions allow you to analyze and account for different behavior on holiday days, around the world. These functions may be used to create series that indicate the proportion of an annual event covered by each observation.

The `@holiday` function returns the proportion or identifier of an annual event covered by the observation, for each observation in the workfile.

The basic syntax for the holiday function is

```
Syntax:      @holiday(h[, b][, flag...])
             h:      string
             b:      string
             flag:   (optional) string
Return:      series
```

where *h* is the holiday event specification, *b* is a basis specification, and *flag* is a calculation scaling flag.

The `@holidayset` function returns the proportion or identifier of multiple annual events covered by the observation, for each observation in the workfile.

`@holiday` is similar to `@holidayset`, but allows for the specification of holidays using a range pair of dates, while disallowing multiple holiday event specifications.

The basic syntax for the holiday set function is

```
Syntax:      @holidayset(h[, b][, flag...])
             h:      string
             b:      string
             flag:   (optional) string
Return:      series
```

where *h* is one or more holiday event specifications, *b* is a basis specification, and *flag* is a calculation scaling flag.

- See “[Holiday Functions](#),” on page 669 of the *Command and Programming Reference* for complete syntax and discussion.

@holiday Examples

The command

```
series jan1 = @holiday("Jan1")
```

generates a series containing a non-zero value only for those observations associated with January 1st. For a daily or lower frequency workfile, only a single observation will cover January 1st each year and that observation will have a value of 1. For a higher frequency workfile, multiple observations will cover January 1st and thus have a value less than one. For example, in an intra-day workfile with a frequency of six hours, each of the four observations for January 1st every year will have the value 0.25.

```
series jan1early = @holiday("Jan1", "Mon-Sun,9AM-2PM")
```

generates a series that is non-zero only for those observations associated with January 1st and between the hours of 9AM (inclusive) and 3PM (exclusive). For an intra-day workfile with a frequency of six hours, the two observations for January 1st that begin at 6AM and noon each year will have the value 0.5. Note that the end time of the basis specification, e.g. 2PM, is considered to extend to the last moment of the specified time, thus 2PM is interpreted as 2:59:59.999PM (3PM rather than 2PM).

```
series mondays = @holiday("Nov1Mon(1)")
```

generates a series that is non-zero only for those observations that fall on the day after the first Monday in November, i.e. the Tuesday that is US federal Election Day.

```
series frenchlabor = @holiday("Labour.fr~")
```

Generates a series that is non-zero only for those observations associated with French Labor Day (May 1st). Should that day fall on a Saturday, the observation(s) for the preceding Friday will be non-zero instead, or if that day falls on a Sunday, the observation(s) for the following Monday will be non-zero. For example, in a daily workfile covering years 2020 through 2024, the observations for Friday May 1 2020, Friday April 30 2021, Monday May 2 2022, Monday May 1 2023, and Wednesday May 1 2024 would have the value 1.

```
series canadaday = @holiday("Canada!")
```

generates a series that is non-zero only for those observations associated with Canada Day (July 1st). Should that day fall on a Saturday or a Sunday, the observation(s) for the following Monday will be non-zero instead. For example, in a daily workfile covering years 2022 through 2024, the observations for Friday July 1 2022, Monday July 3 2023, and Monday July 1 2024 would have the value 1.

```
series lunar = @holiday("LNY~(-3)")
```

generates a series that is non-zero only for those observations occurring three days before the Lunar New Year, adjusted for weekends. For example, in a daily workfile covering year 2020, while the nominal date for the holiday is Saturday January 25, the observation for Tuesday January 21 would have the value 1 as a consequence of the weekend modifier and offset.

```
series newyears1 = @holiday("NewYears[1,2,0]")
```

generates a series that is non-zero only for those observations associated with New Year's Day (January 1st) and the preceding day (December 31st). In the common case where those two days will be covered by different observations, the observation for December 31st will have the value 0.33 and the observation for January 1st will have the value 0.67 given the relative weights specified.

```
series newyears2 = @holiday("NewYears[1,2,0]","denorm")
```

generates a series that is non-zero only for those observations associated with New Year's Day (January 1st) and the preceding day (December 31st). With the use of the “denorm”

option, the observation for December 31st will have the value 1 and the observation for January 1st will have the value 2.

```
series ukbank = @holiday("Bank.uk!")
```

generates a series that is non-zero only for those observations associated with United Kingdom bank holidays included in the “bank.uk” named group. The weekend modifier “!” is applied to each individual holiday in the group.

```
series postvets = @holiday("Veterans.us(7) Thanksgiving.us")
```

generates a series that is non-zero only for those observations between a week after US Veterans Day and US Thanksgiving. This range normally covers between five to eleven days, depending on the year. For example, in 2020 this range covers nine days (November 18 through November 26), thus in daily workfile the observations associated with those days would have the value 0.111.

@holidayset Examples

The command

```
series jan1set = @holidayset("Jan1")
```

generates a series containing a non-zero value only for those observations associated with January 1st. For a daily or lower frequency workfile, only a single observation will cover January 1st each year and that observation will have a value of 1. For a higher frequency workfile, multiple observations will cover January 1st and thus have a value less than one. For example, in an intra-day workfile with a frequency of six hours, each of the four observations for January 1st every year will have the value 0.25. Since only a single holiday is specified, this function behaves identically to `@holiday`.

```
series vetsset = @holidayset("Veterans.us(7) Thanksgiving.us")
```

generates a series that is non-zero only for those observations a week after US Veterans Day and on US Thanksgiving. For example, in 2020 the two holidays occur on November 18 and November 26, thus in daily workfile the observations associated with those days would each have the value 0.5. Note that unlike the `@holiday` function, the two holidays included are distinct and do not specify a range.

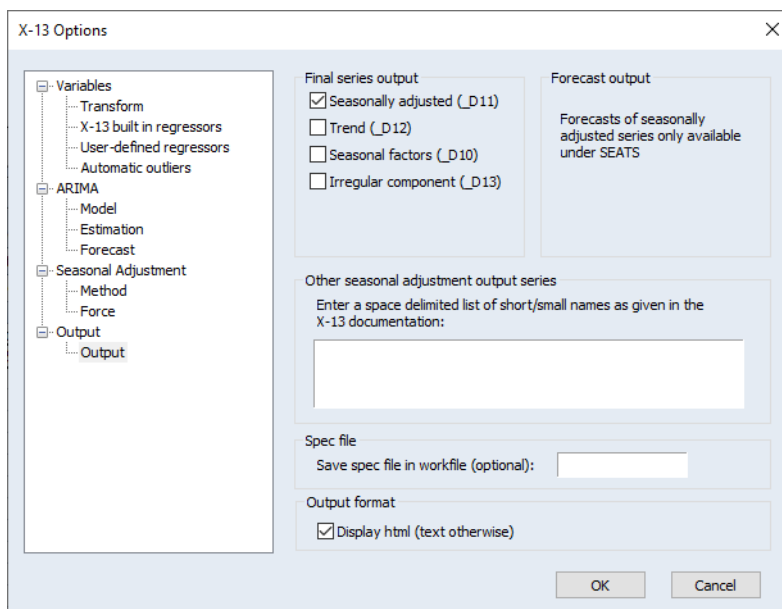
```
series easterset = @holidayset("Easter Unity.de[ramp(3)] Oct31~")
```

generates a series that is non-zero only for those observations associated with Easter, the five days on and around German Unity Day, and Halloween, adjusted for weekends. For example, in a daily workfile covering year 2020, the observation for Sunday April 12 will have the value 0.333, the observations for Thursday October 1 through Monday October 5 will have the values 0.037, 0.074, 0.111, 0.074, and 0.037, respectively, and finally the observation for Friday October 30 will have the value 0.333.

Updated X-13 Engine

EViews has updated the version of the Census X-13 executable to support HTML output. The current version will be newer than Version 1.1, Build 59.

The updated X-13 engine allows you to save your X-13 output directly to HTML. Click on the **Output** node in the tree, and make sure the **Display html** checkbox is selected:



Note that the Census Bureau has tested this version of X-13ARIMA-SEATS on Windows 10 and indicates that it may work on previous Windows versions.

SDMX Databases

As part of EViews support for SDMX Databases, EViews 13 now offers access to Australian Bureau of Statistics (ABS) SDMX, Deutsche Bundesbank (DB) SDMX, and Insee (L'Institut national de la statistique et des études économiques) SDMX data.

Please note that an internet connection is required to obtain SDMX online data. For more information on the ABS, DB, or Insee data, see:

<http://api.data.abs.gov.au>

<http://api.statistiken.bundesbank.de>

<http://www.insee.fr/en/information/2868055>

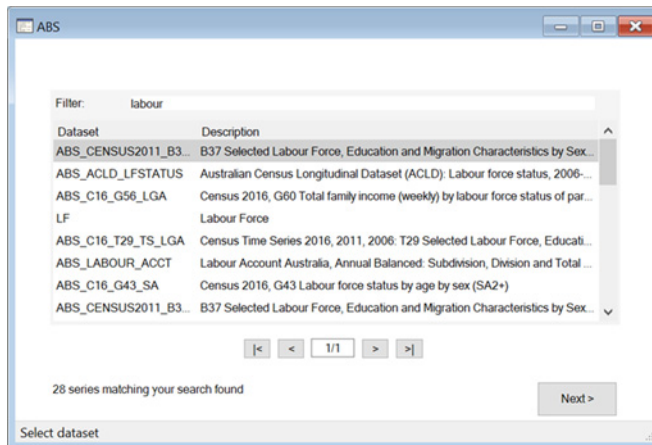
EViews offers a custom browser for navigation to and retrieval of available data from SDMX databases. To start, open a database window by selecting **File/Open Database...** from the

main EViews menus, then select **Eurostat SDMX Database**, **ECB SDMX Database**, **UN SDMX Database**, **IMF SDMX Database**, **OECD SDMX Database**, **ABS SDMX Database**, **Deutsche Bundesbank SDMX Database**, or **Insee SDMX Database** from the **Database/File type** dropdown menu. A dialog similar to the one below will be displayed:

Click **OK** to open the standard EViews database window. Click on **Browse** or **Browse-Append** to open a custom database.

For example, this is the interface to the Eurostat data:

The dialog allows data to be selected from within datasets; alternatively, the browser interface is a way to search through the datasets by typing a keyword in the **Filter** textbox:



Select a dataset and click **Next** to display a dialog for viewing and selecting series:

Command

Command Capture

ABS

Reset Filters

SDMX ABS
Dataset: Labour Force

MEASURE	SERIES NAME	MEASURE	SEX	AC
Employed - full-time (M1)			Females	Tot
Civilian population (M11)			Persons	Tot
Employed - part-time (M2)			Persons	Tot
Employed Persons - Monthly hours worked in all jobs (M18)			Males	Tot
Employed persons (M3)				
Employment to population ratio (M16)				
Full-time Employed - Monthly hours worked in all jobs (M19)				
Labour Force - Full-time (M7)				

788 series matching your se

< Back Export to workfile

Query complete Path = h:\temp DB = abs WF = none

The dialog contains a table with all of the series matching the search. On the left is an interface with a list of dropdown boxes containing additional search filter criteria. Click a box or boxes to select one or more options. If more filter criteria are selected the results on the right will be updated to match the new selections. On the bottom right is a graph preview of the series data selected.

Once the selection process is finished, drag-and-drop or click the **Export to workfile** button to export the series directly into a new or existing workfile.

- See “[Foreign Format Databases](#)” on page 362 in *User’s Guide I*.
- See also [dbopen](#) (p. 371) in the *Command and Programming Reference*.

Trading Economics Databases

Trading Economics provides access to a large range of historical and forecast data for economic indicators, stock markets, government bonds, exchange rates, and commodity prices.

Please note that an internet connection is required to obtain Trading Economics online data. For more information on Trading Economics data subscriptions, please see:

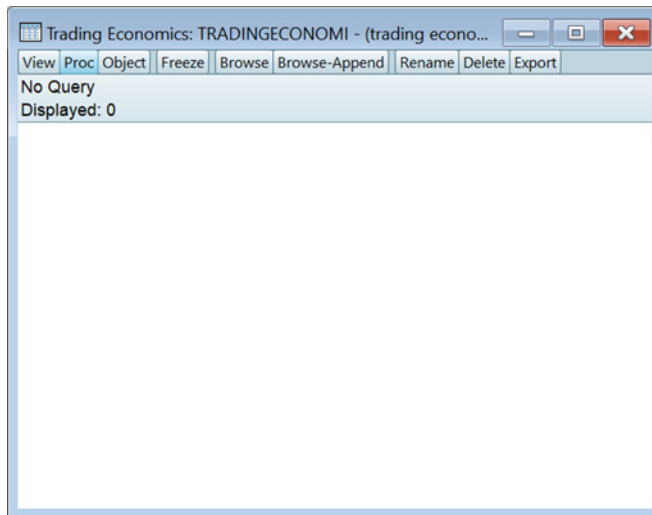
<https://tradingeconomics.com/analytics/features.aspx>

Additionally, direct access to Trading Economics online data is only offered in the EViews Enterprise Edition.

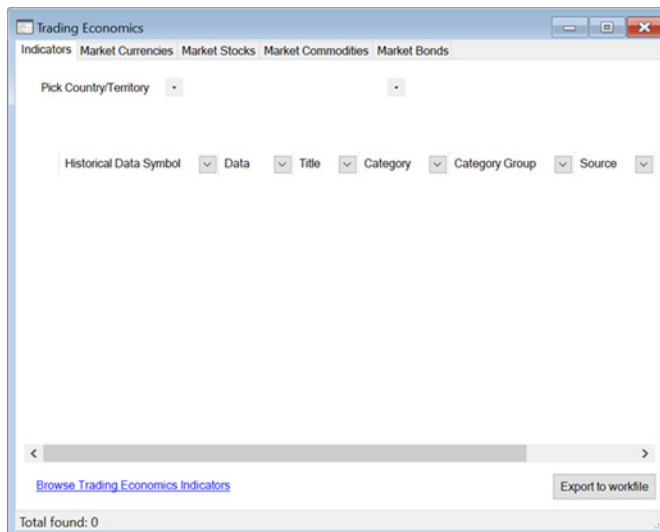
EViews offers a custom interface to Trading Economics data. To open the database, select **File/Open Database...** from the main EViews menu and then select **Trading Economics** from the **Database/File type** dropdown menu:

The screenshot shows a 'Database Specification' dialog box. It has a title bar with a close button. The main area is divided into two sections. The top section, 'Database specification', contains a dropdown menu for 'Database/File type' with 'Trading Economics' selected. Below it are text boxes for 'Server specification' (containing 'https://api.tradingeconomics.com'), 'User name', 'Password', and 'Database name' (containing 'Trading Economics'). There are three buttons: 'Browse', 'Browse Registry', and 'Add to Registry'. The bottom section, 'Open as', contains a text box for 'Database alias (optional short name)'. At the very bottom are 'OK' and 'Cancel' buttons.

Click **OK** to open the standard EViews database window:



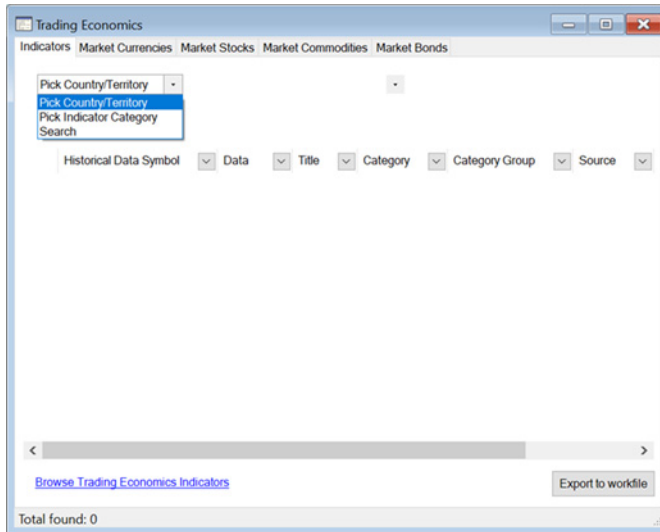
Click on **Browse** in the toolbar to open the custom Trading Economics window:



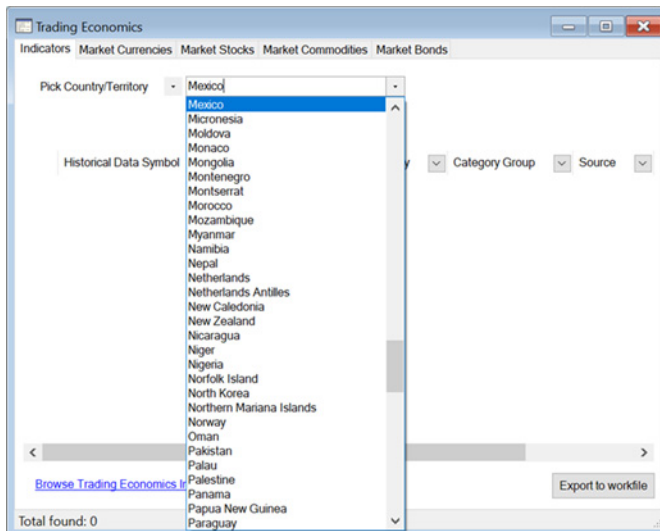
The dialog interface displays five tabs, or categories, of data: **Indicators**, **Market Currencies**, **Market Stocks**, **Market Commodities**, and **Market Bonds**.

Indicators

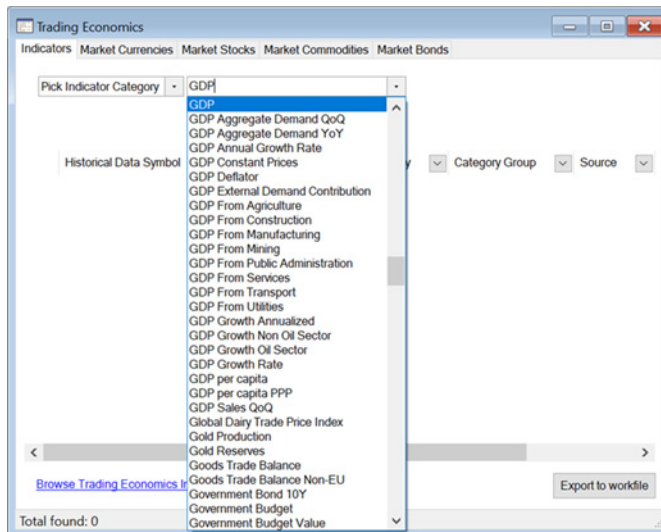
Under the **Indicators** tab, choose between picking a country/territory, an indicator category, or a search:



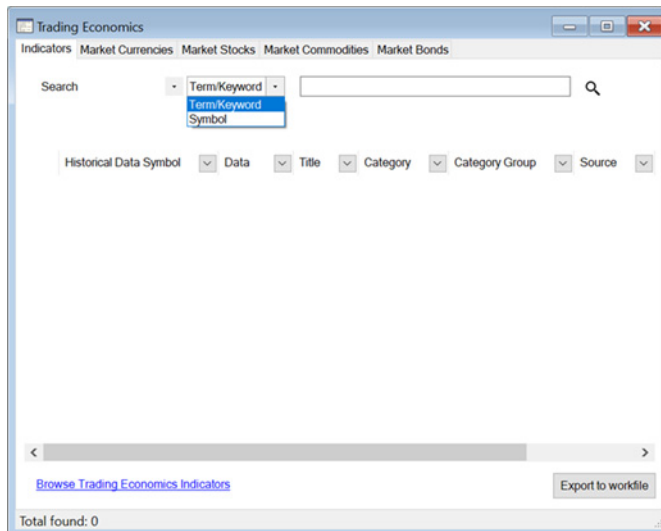
For **Pick Country/Territory**, select a country from the drop-down box to see historical and forecast indicators:



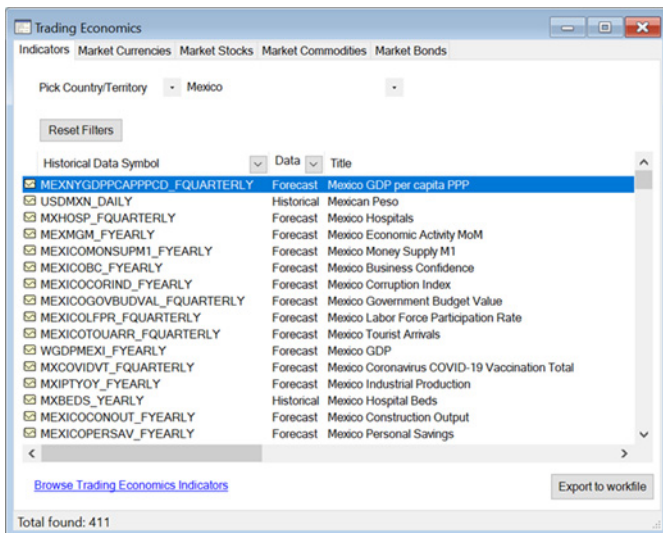
For **Pick Indicator Category**, select a category from the drop-down box to see historical and forecast indicators:



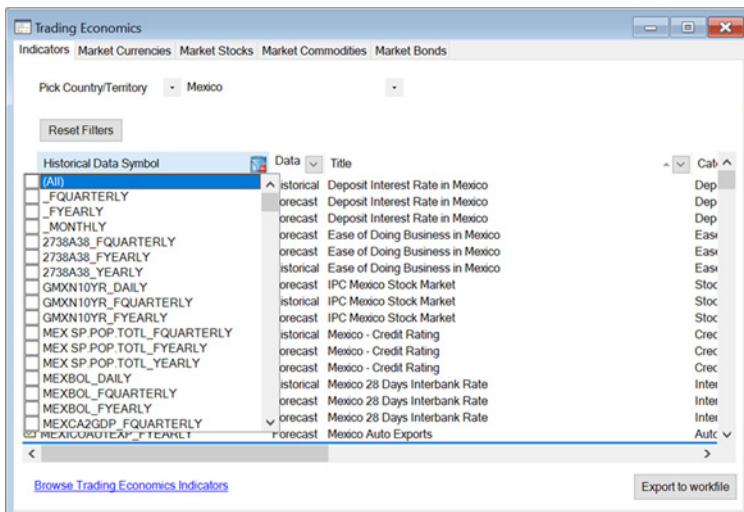
For **Search**, select **Term/Keyword** or **Symbol** and enter an expression in the text box:



The user interface provides a way to filter through the results regardless of the method chosen. After picking the country Mexico, for example, the dialog displays a table with the indicator results:



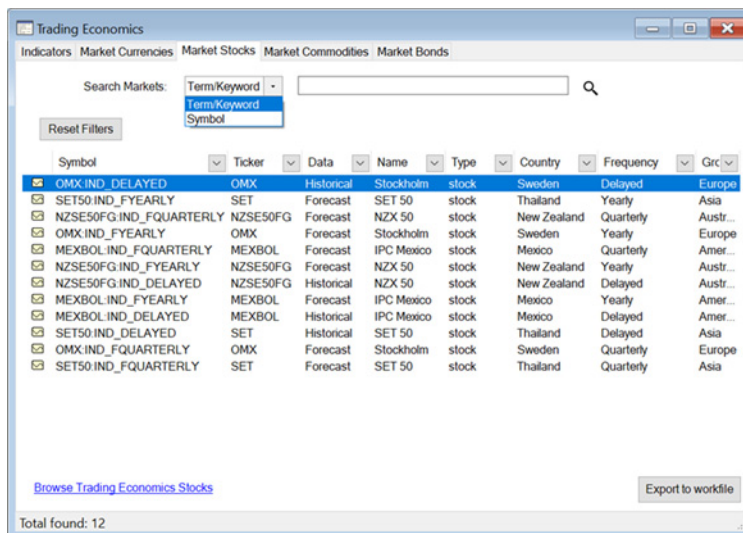
Click the drop-down box next to each column header to show a set of filter checkboxes. Select one or more of these to display selected values or select **All** to show all values. Click **Reset Filter** to clear all filters.



Markets

If any of the market tabs (**Market Currencies**, **Market Stocks**, **Market Commodities**, **Market Bonds**) are selected, the dialog will display a table with all the stock market symbol

results you have permission to see based on your subscription. For example, the below dialog shows the **Market Stocks** tab:



You may elect to search for a **Term/Keyword** or a **Symbol**. Once the series of interest is found and selected, drag-and-drop or click the **Export to workfile** button to export the series directly into a new or existing workfile.

For more information click the **Browse Trading Economics Stocks** link at the bottom of the dialog to open the Trading Economics webpage.

- See “Foreign Format Databases” on page 362 in *User’s Guide I*.
- See also `dbopen` (p. 371) in the *Command and Programming Reference*.

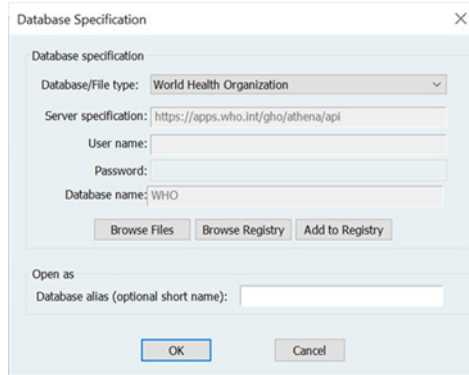
World Health Organization Databases

World Health Organization (WHO) provides access to a large range of health-related data and statistics. An internet connection is required to obtain WHO online data.

For more information on data subscriptions, please see:

<https://www.who.int/data/gho>

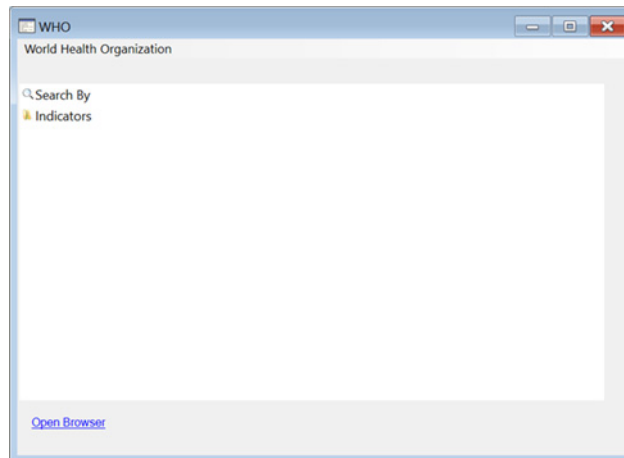
EViews offers a custom interface to World Health Organization data. To open the database, select **File/Open Database...** from the main EViews menu and select **World Health Organization** from the **Database/File type** dropdown menu:



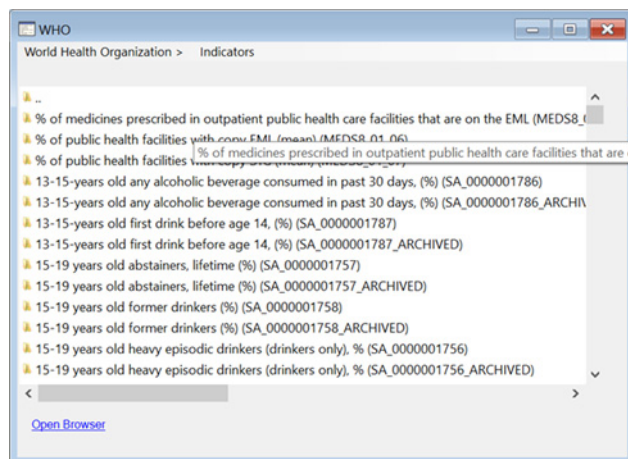
Click **OK** to open the standard EViews database window:



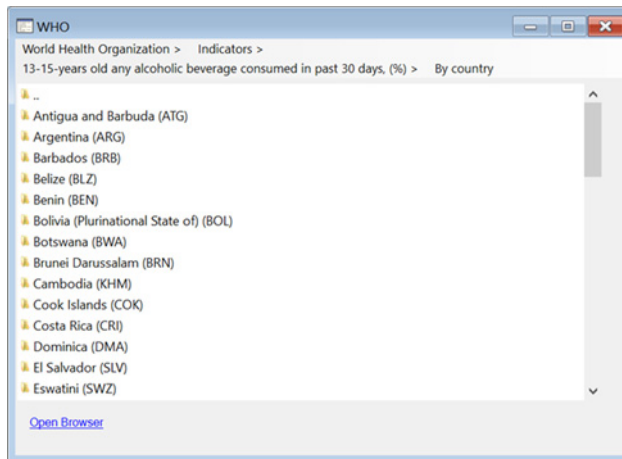
Click on **Browse** in the toolbar to open the custom WHO window:



Click on the **Indicators** folder and navigate through the set of nested folders:



Click on a folder to open its subtopics:



The full path of the active folder is shown in the header. Click on any folder within the path to return to it. For example, clicking on **By country** will move up a single level to show the country choices. Similarly, clicking on “..” in the window listing will move up a single level.

Alternately, click on **Search By** to search for a keyword:



Once the series of interest is selected, drag or copy it into a workfile. For more information, click the **Open Browser** link at the bottom of the dialog to open the WHO website.

- See “Foreign Format Databases” on page 362 in *User’s Guide I*.
- See also `dbopen` (p. 371) in the *Command and Programming Reference*.

Simplified Matrix Data Access

EViews 13 offers a useful set of tools for extracting parts of matrices for further use. Some of the functions described below are available in limited form in prior versions of EViews, but the new additions round out the set of functions for accessing data, allowing for easy-to-use operations that were previously difficult to perform.

Useful Matrix Utility Functions

There are five new or updated utility functions that are particularly useful for working with matrix data.

Briefly:

- `@fill(n1, n2, n3, ...)` – return a numeric vector with the specified values.
- `@range(n1, n2)` – return a numeric vector with the sequential integer values from *n1* to *n2*.
- `@seq(s, d, n)` – return a numeric vector with the arithmetic sequence of *n* elements beginning with *s* and incrementing by *d*.
- `@grid(n1, n2, n3)` – return a numeric vector containing a grid of *n3* values from *n1* to *n2*.
- `@sfill("str1", "str2", "str3", ...)` – return a svector using the specified double-quote enclosed strings.

Note that the first three functions create vector objects, while the latter creates an svector (string vector) object.

These functions are straightforward in intention. While `@fill` and `@seq` will work with arbitrary numeric values, we are interested here in their use in generating vectors of integer values.

```
vector x1 = @fill(1, 2, 4, 5, 7)
```

creates the vector with elements {1, 2, 4, 5, 7}, while

```
vector x2 = @range(1, 5)
```

creates the vector with elements {1, 2, 3, 4, 5}, while

```
vector x3 = @seq(1, 2, 4)
```

creates the vector with elements {1, 3, 5, 7}, while

```
vector x4 = @grid(1, 2, 6)
```

creates the vector with elements {1.0, 1.2, 1.4, 1.6, 1.8, 2.0}, and

```
svector sx1 = @sfill("apple", "pear", "orange")
```

creates the svector with the values {"apple", "pear", "orange"}.

Matrix Extraction Data Members

The best way to extract data from a matrix object is to use matrix object member functions.

Some of these member functions were available for selected objects in earlier versions and some are new in EViews 13. All of the functions have enhanced scope in EViews 13 along with new functionality for referencing data using matrix row and column labels.

The relevant functions for all matrix objects are:

- *obj.@col(arg)* – returns matrix object containing column(s) of *obj* associated with *arg*
- *obj.@row(arg)* – returns matrix object containing row(s) of *obj* associated with *arg*
- *obj.@sub(arg1, arg2)* – returns matrix object containing row(s) and col(s) of *obj* associated with *arg1* and *arg2*, respectively
- *obj.@dropcol(arg)* – returns matrix object containing column(s) of *obj* not associated with *arg*
- *obj.@droprow(arg)* – returns matrix object containing row(s) of *obj* not associated with *arg*
- *obj.@dropboth(arg1, arg2)* – returns matrix object containing row(s) and col(s) of *obj* not associated with *arg1* and *arg2*, respectively

For symmetric matrices, we also have the functions

- *obj.@sub(arg)* – returns sym object containing row(s) and col(s) of *obj* associated with *arg*, respectively
- *obj.@dropboth(arg)* – returns sym object containing row(s) and col(s) of *obj* not associated with *arg*, respectively

where

- *obj* is the name of a matrix object in the workfile
- *arg, arg1, arg2* are integers, scalar objects, vectors, strings, or svector

For cases where *args* are numeric (integers, scalars, vectors), the *arg* values act as row or column indices. Focusing on column functions, for example,

```
vector v1 = x.@col(3)
matrix m1 = x.@col(cid)
```

where *X* is a matrix and *CID* is a vector of column indices, and the two lines return the vector *V1* and matrix *M1* containing the 3rd column of *X*, and the columns of *X* referenced in

CID, respectively. Note that the elements of CID must be integers from 1 to the number of columns of X.

For cases where *args* are strings (string literal, string object, svector), the *arg* values are examined to find matches in the corresponding row or column labels (“[Matrix Row and Column Labels](#),” on page 43). For example, the commands

```
vector v2 = x.@col("apple")
matrix m2 = x.@col(scid)
```

where SCID is a svector of strings, produce the vector V2 and matrix M2 containing the 3rd column of X, and the columns of X with labels that match the elements of SCID, respectively. Note that all of the elements of SCID must be strings that match the column names previously assigned to X.

You may mix the types of *arg1* and *arg2* in data member functions that take two arguments so that, for example,

```
matrix m3 = x.@sub(3, "apple")
```

returns M3 containing the element of X in row 3 and column with label “apple”.

Similarly, the commands

```
matrix v1d = x.@dropcol(3)
matrix m1d = x.@dropcol(cid)
matrix v2d = x.@dropcol("apple")
matrix m2d = x.@dropcol(scid)
matrix m3d = x.@dropboth(5, "apple")
```

return the matrix objects

- V1D, the matrix X with column 3 dropped
- M1D, the matrix X with columns referenced by CID dropped
- V2D, the matrix X with the column with label “apple” dropped
- M2D, the matrix X with the columns with labels in SCID dropped
- M3D, the matrix with the row 5 dropped and the column labeled “apple” dropped

The special symmetric matrix versions of these functions return syms:

```
sym sym1 = symorig.@sub(cid)
sym sym2 = symorig.@dropcol(3)
sym sym3 = symorig.@dropcol(cid)
```

return the sym objects

- SYM1, the columns (and corresponding rows) of SYMORIG referenced by CID
- SYM2, the contents of SYMORIG after dropping column and row 3
- V2D, the contents of SYMORIG after dropping columns and rows referenced by CID

Matrix Extraction with Utility Functions

Combining matrix utility functions (“[Useful Matrix Utility Functions](#),” on page 37) with the matrix extraction data members offers flexible methods for obtaining data from matrices.

Consider, for example, the extraction of multiple columns from the matrix X using the `@col` data member function:

```
vector xid = @fill(1, 3, 5, 9)
matrix xsub = x.@col(xid)
```

extracts columns {1, 3, 5, 9} from the matrix X.

Since the *args* in the member data extraction functions may themselves be expressions, we may combine the two lines into a single expression:

```
matrix xsub1 = x.@col(@fill(1, 3, 5, 9))
```

Similarly, extracting or dropping the 7 through 9th columns of X may be done using

```
matrix xsub2 = x.@col(@range(7, 9))
matrix xsub3 = x.@dropcol(@range(7, 9))
```

We can perform the same compound extractions in 2-dimensions, as in

```
matrix xsub4 = x.@sub(@range(3, 6), @sfill("apple", "orange"))
matrix xsub5 = x.@dropboth(4, @sfill("apple", "orange"))
```

which create XSUB4 containing rows 3 to 6 and columns with labels matching “apple” and “orange” of X, and XSUB5 containing X after dropping row 4 and the columns with matching labels.

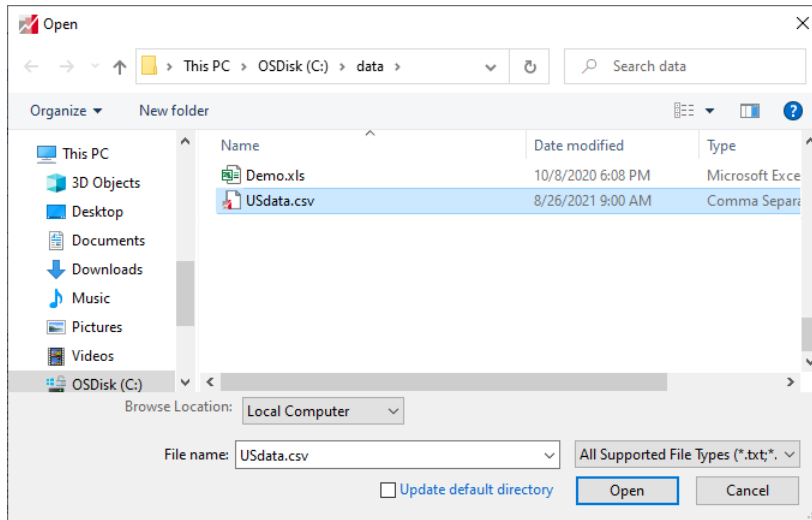
New Matrix Import/Export Engine

An all new matrix data engine makes it easier than ever to get data into and out of external data sources and offers improved support for different data formats.

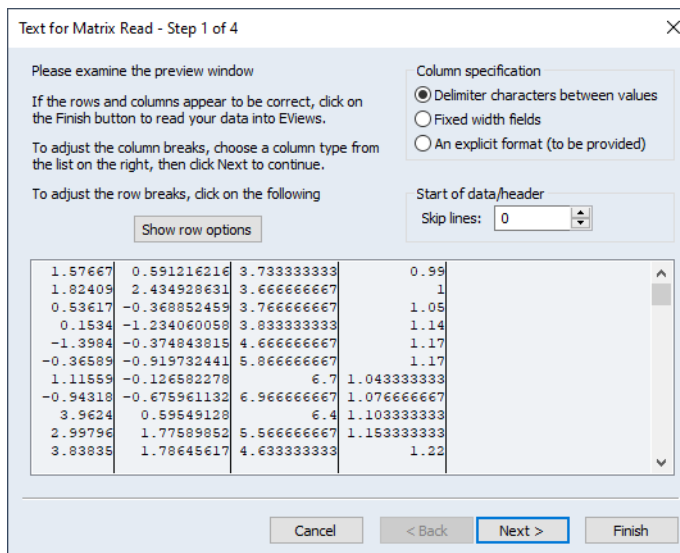
Import Data into Matrix objects

You may now read directly into an EViews matrix object (matrix, vector, sym, etc.) from text (both ASCII and binary), HTML, Excel XLSX, and Excel 97 XLS files. Excel reads includes support for named ranges and multiple pages. The new engine supports a number of different formats, and features EViews interactive data import wizard which walks you step-by-step through the data import, providing fine-tuned control of the pending import, previews of the final data, and command capture.

To read data into a matrix object open the matrix and select **Proc/Import Data...** EViews will open the standard file dialog prompting you to select a file:



Double-click to select the file, or highlight the file and click on **Open**. EViews will open a data import wizard:



Proceed through the steps of the wizard by filling out the desired values and clicking on **Next**, and click on **Finish** when ready to import the data. EViews will read the data into the matrix object, *resizing the object to match the source size, if possible*.

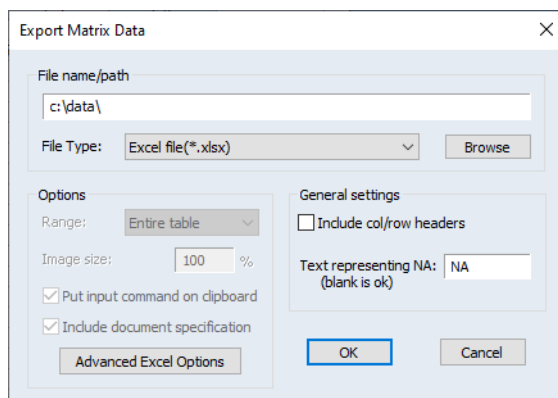
Note that some matrix objects offer some challenges in data import. If you have a vector import from multi-column data,

- See [Matrix::import \(p. 521\)](#) in the *Object Reference* for representative command documentation. Equivalent functions are available for vector, sym, rowvector, and coef objects.

Export Data from Matrix Objects

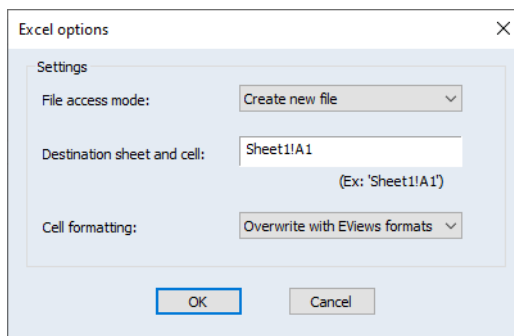
When you are ready to export data from a matrix object, EViews 13 allows you to write to a number of formats including the various ASCII, binary, HTML, RTF, and Excel formats, along with LaTeX, Markdown, and PDF files. Importantly, the Excel XLSX export allows you to write the matrix results into existing Excel files, beginning at a specified cell.

To write the contents of the matrix, select **Proc/Export Data...** from the matrix menu:



Enter a file name in the edit field, or **Browse** to select a file.

You may change the **Data order** to transpose the data prior to write, provide options related to the target file type, such as **Advanced Excel Options** for XLSX,



which permit writing into an specific sheet and cell of a new or existing file, with or without formatting.

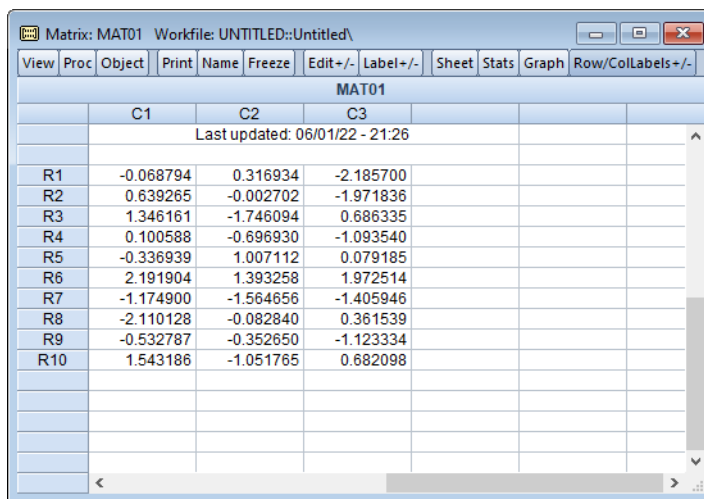
- See `Matrix::export` (p. 517) in the *Object Reference* for representative command documentation. Equivalent functions are available for vector, sym, rowvector, and coef objects.

Matrix Row and Column Labels

- labeling rows and columns in the spreadsheet
- referring to rows and columns of the matrix when accessing data

By default, there are no row and column labels in matrix objects. When spreadsheets show the contents of the matrix, the rows are labeled as “R1”, “R2”, *etc.*, and the columns are labeled as “C1”, “C2”, “C3”, *etc.*

To define row and column labels, you may open a matrix object display the spreadsheet view:



The screenshot shows a spreadsheet window titled "Matrix: MAT01" with a menu bar containing "View", "Proc", "Object", "Print", "Name", "Freeze", "Edit+/-", "Label+/-", "Sheet", "Stats", "Graph", and "Row/ColLabels+/-". The spreadsheet content is as follows:

MAT01			
	C1	C2	C3
	Last updated: 06/01/22 - 21:26		
R1	-0.068794	0.316934	-2.185700
R2	0.639265	-0.002702	-1.971836
R3	1.346161	-1.746094	0.686335
R4	0.100588	-0.696930	-1.093540
R5	-0.336939	1.007112	0.079185
R6	2.191904	1.393258	1.972514
R7	-1.174900	-1.564656	-1.405946
R8	-2.110128	-0.082840	0.361539
R9	-0.532787	-0.352650	-1.123334
R10	1.543186	-1.051765	0.682098

By default, there are no labels defined, but you may use the matrix procs `setrowlabels` and `setcollabels` to assign new values:

- See `Matrix::setrowlabels` (p. 541) and `Matrix::setcollabels` (p. 538) in the *Object Reference* for representative command documentation. Equivalent functions are available for vector, sym, rowvector, and coef objects,
- See `Matrix::clearrowlabels` (p. 509) and `Matrix::clearcollabels` (p. 508) in the *Object Reference* for representative command documentation. Equivalent functions are available for vector, sym, rowvector, and coef objects,

For example,

```
mat01.setcollabels "First" "Alternate"
```

sets column labels for the first two columns.

By default, these labels will be used in the spreadsheet display of the matrix:

The screenshot shows a window titled "Matrix: MAT01" with a menu bar containing "View", "Proc", "Object", "Print", "Name", "Freeze", "Edit +/-", "Label +/-", "Sheet", "Stats", "Graph", and "Row/ColLabels:". The matrix data is displayed as follows:

	First	Alternate	C3
	Last updated: 06/01/22 - 21:37		
R1	-0.068794	0.316934	-2.185700
R2	0.639265	-0.002702	-1.971836
R3	1.346161	-1.746094	0.686335
R4	0.100588	-0.696930	-1.093540
R5	-0.336939	1.007112	0.079185
R6	2.191904	1.393258	1.972514
R7	-1.174900	-1.564656	-1.405946
R8	-2.110128	-0.082840	0.361539
R9	-0.532787	-0.352650	-1.123334
R10	1.543186	-1.051765	0.682098

The **Row/Collabels** +/- toggles on and off the display of the labels.

EViews 13 adds the ability to extract data from the matrices using the row and column labels. For example, with the matrix object, we have matrix data members,

@col(arg) Returns the columns defined by *arg*.

@dropcol(arg) Returns the matrix with the columns defined by *arg* removed

@droprow(arg) Returns the matrix with rows defined by *arg* removed.

@drobsub(arg1, arg2) Returns the matrix with the rows defined by *arg1* and columns defined by *arg2* removed.

@row(arg) Returns the rows defined by *arg*.

@sub(arg1, arg2).. Returns the matrix with rows defined by *arg1* and columns with defined by *arg2*.

that all take *args* that may be integers, vectors of integers, string, or svectors of strings. Importantly, integer values will correspond to row and column indices. while string values will correspond to previously defined row and column labels.

Thus, in our example from above,

```
vector vec1 = mat01.@col("Alternate")
```

will create the vector VEC1 containing the column labeled with “Alternate” in MATRIX01.

This command is equivalent to

```
vector vec1 = mat01.@col(2)
```

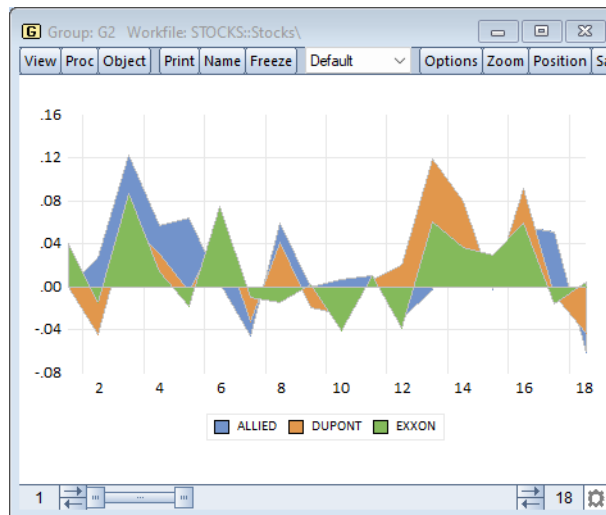
See “[Matrix Extraction Data Members](#),” on page 38 for additional discussion.

Graphs and Tables

Graph Line and Shade Transparency

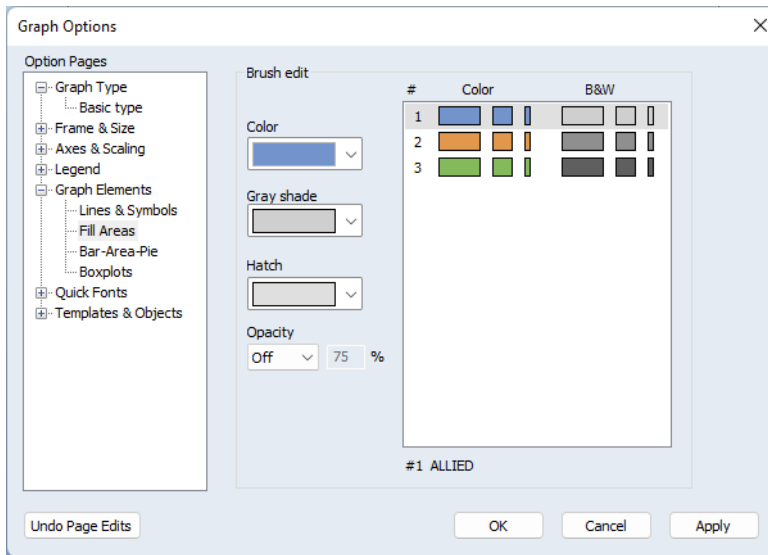
EViews 13 now allows you to customize the opacity (transparency) levels of individual lines and shades in a graph. By exercising fine control over the visibility of stacked graph elements, you can uncover previously hidden features of your data.

When plotting multiple data for series in prior versions of EViews, the graph for data from one series could obscure the data of another. For example, the following area graph shows time series graphs for three series in a group object (ALLIED, DUPONT, and EXXON):



In this graph, data for ALLIED is drawn first, followed by the data for DUPONT, and then by the data for EXXON. Notice that the areas for the later drawn series obscuring the areas for the earlier. Note in particular, that the values of EXXON for observations hide the corresponding values of DUPONT and ALLIED, particularly for observations from 9 to 16.

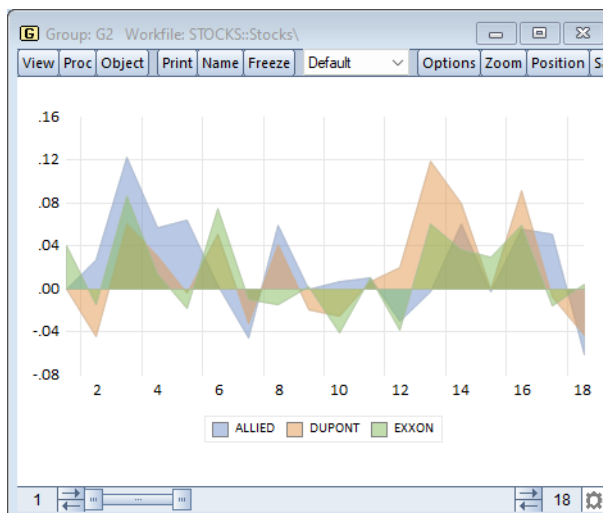
EViews 13 allows you to adjust the opacity of individual graph elements to improve the visibility of others. To set the opacity for one or more elements, double click on the graph to display the graph options, or select on the **Options** button on the button bar, then select the **Graph Elements** node.



Opacity levels may be set for individual **Lines** and **Fill Areas** by clicking on an entry in the right hand side of the graph to select a graph element and then entering a number from 0 to 100 in the **Opacity %** edit field:

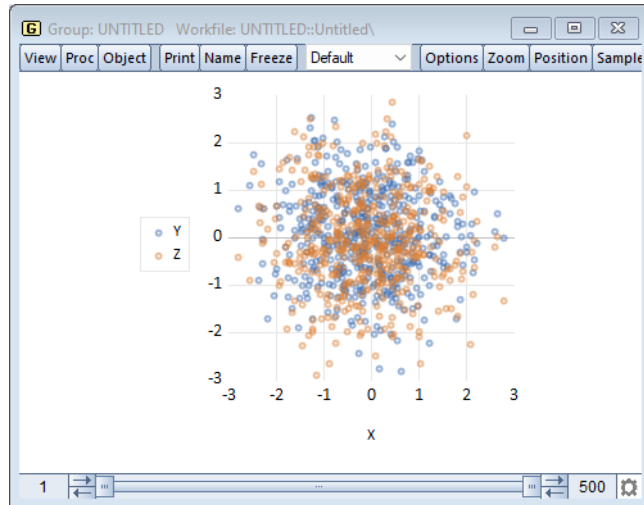
You may set levels for more than one element simultaneously by SHIFT or CTRL clicking to select multiple elements, and then applying the desired setting.

Here is the same graph after setting the **Opacity %** of all three of the fill elements to “50”:



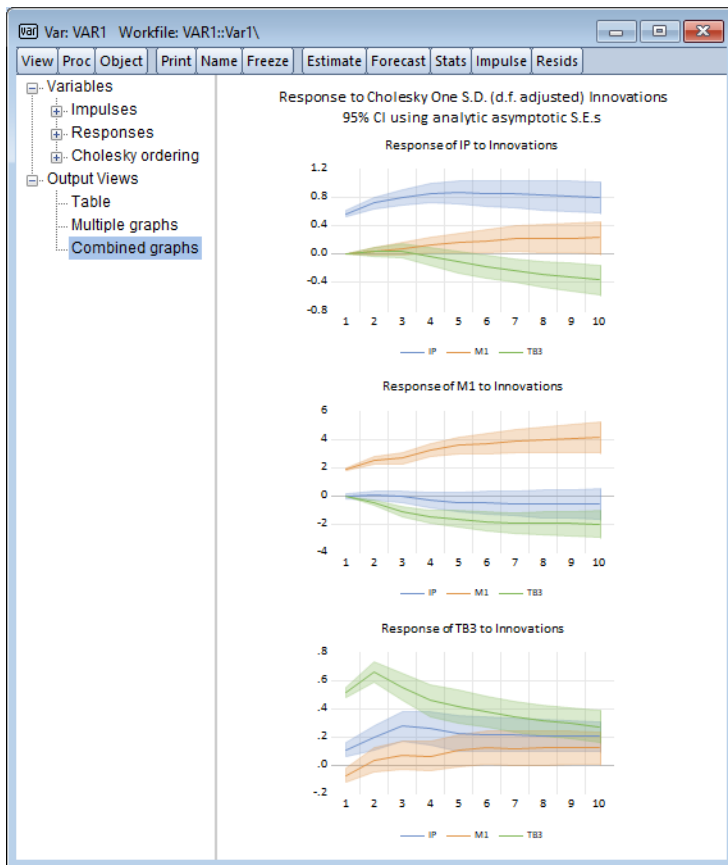
Note the additional visible detail in the values of the ALLIED and DUPONT series for observations 9 to 16. In particular, the values of ALLIED, the first drawn series were virtually invisible in the earlier graph, but now show an obvious sawtooth pattern.

Similarly, turning down the opacity in a dense scatterplot can show additional detail:



Judicious use of transparency settings makes possible the production of graphs that show data in ways that were not previously possible.

The new multiple shaded confidence intervals in combined impulse-response graphs ([“Enhanced Impulse Response Display”](#) on page 90) employ this feature and hint at the types of graphs that may be produced:



To set opacity levels by command use the `Graph::setelem` (p. 373) (in the *Object Reference*) command to select an element and add the `lineopacity` (for lines) and `fillopacity` (for fills) keywords to set the desired opacity values.

Setting the level to 0.0 (0%) will make the object completely transparent while 1.0 (100%) will make the object completely opaque.

For example,

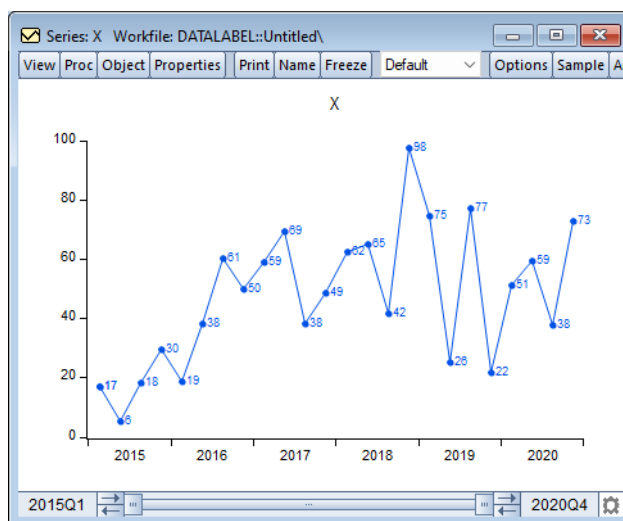
```
gr1.setelem(2) lineopacity(.5)
```

In our example, the second line in the graph was set to be 0.5 (50%) opaque.

Custom Graph Data Labels

Providing labels for data values can be an important tool for enhancing the information content of graphical presentation of data. EViews 13 offers new automatic tools which make it easy to augment your graphs with informative custom data and observation-based labels.

Recall that in EViews 12 and earlier, EViews offered option to use data values to label each of the observations in a graph.



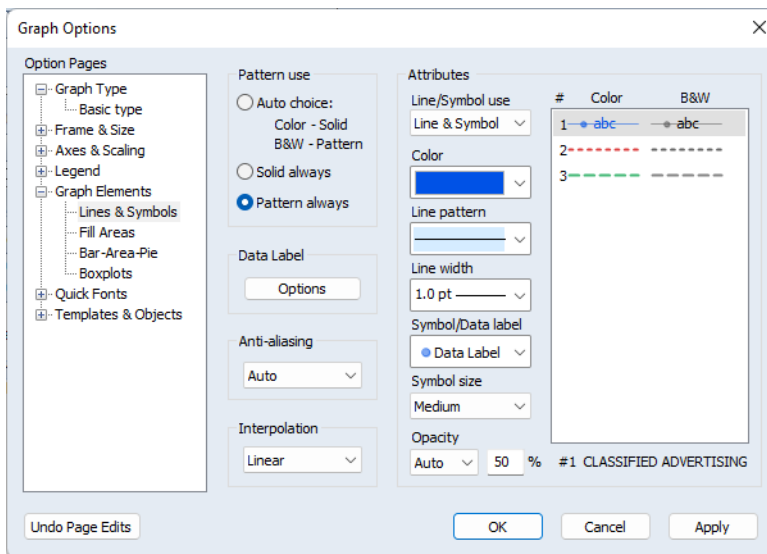
While often quite useful, this option had limitation. For one, labeling observations with data values was an all or nothing proposition; data values were either displayed for all observations or for none of the observations. Further, customization of the format of the data labels was limited to specifying a size and font of the data label, and showing or not showing an open or closed circle symbol alongside the label.

EViews 13 enhances the ability to label observations to provide you with greater control over your data labeling. You may now:

- Label all observations, no observations, the first observation, or the last observation.
- Specify the content of the data label by using the data value, the observation, the name of the series, or arbitrary text.
- Modify the font, font size and position of the data label.

Data labeling is controlled in the **Graph Options** dialog. Click on the **Options** button or double-click on the graph to display the dialog then select **Graph Elements** and **Lines & Symbols** and select one or more the elements on the right-hand side of the dialog The **Symbol/** Data label settings will be displayed when you choose **Symbols** or **Lines & Symbols** to dis-

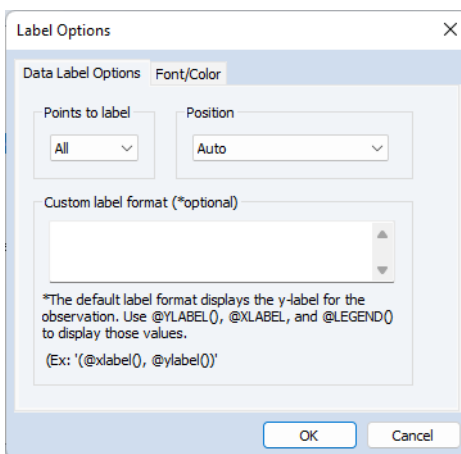
play. To show data labels, select one of the three **Data label** entries in the drop-down control (open circle and data label, closed circle and data label, data label):



Here, we have selected the **Data label** with a closed circle symbol.

By default, when you choose a data label setting, EViews will display labels for all observations.

EViews 13 introduces the ability to customize the data label display. Click on the large **Options** button under the **Data Label** entry in the middle of the dialog to display the **Label Options** dialog:



The first tab provides basic labeling options:

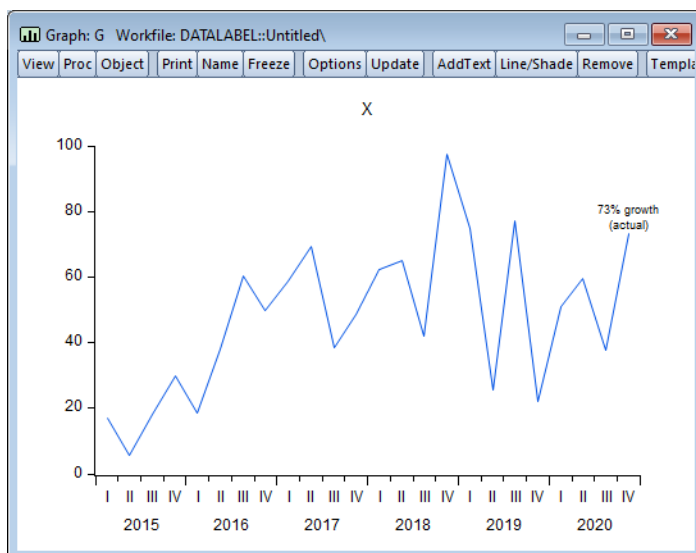
- The radio buttons in the bottom left allow you to choose between displaying labels for **All**, the **First**, or the **Last** observation.
- The **Position** drop down lets you choose to display the label using **Auto** positioning, or to the **Right of observation**, **Left of observation**, **Above observation**, **Below observation**, or **Center on observation**.
- The **Label Format** edit field allows you to specify the content of the label. You should enter text for the specification in the edit field. The special functions `@xlabel()` and `@ylabel()` instruct EViews to use the corresponding X-values and Y-values of the observations as the label; the function `@legend()` corresponds to the legend value for the data element. All other text will be used in the label as given. You may use the ENTER key format the label in multiple lines.

The second tab of the dialog specifies font family, size, and color settings, as well as special text effects like underlining and strikethrough.

Below, we use the **Last** observation setting to label the last data point using the Y-value to form a custom data string “% growth (actual)”, by entering

```
@ylabel()% growth
(actual)
```

in the edit field. Note that the carriage return in the edit field is used to create a multi-line label.

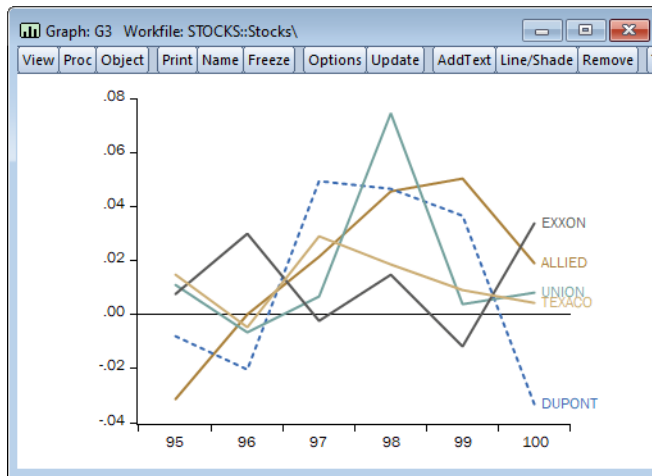


Importantly, if an observation were to be added to end of the graph, the data label would automatic change to reflect the new value as well as move to proper location above new observation.

Similarly, you may use custom data labels in place of a legend. In this example, the legend was disabled and we activate the custom data label for the last observation, using a

```
@legend()
```

specification in the edit field.



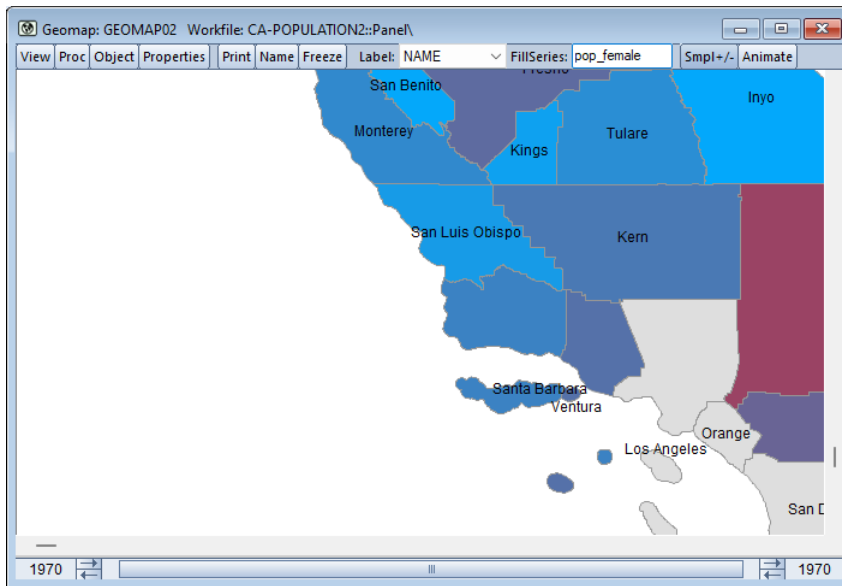
Note that a custom data label differs from a custom text label. Data labels are attached to an observation in the graph, while custom text labels may be placed anywhere in the graph. While custom text labels may be placed so that they appear related to specific data values, it takes a bit of work to tie them to the observation data values. In contrast, a custom data label is designed to be linked to the observations so it may easy to attach label text to the first, last, or all of the observations.

- See [Graph::datalabel \(p. 344\)](#) in the *Object Reference* for command documentation.

Customizable Geomap Labels

In EViews 13, you may now use more than one attribute, along with custom text and formatting, to label the shapes in your geomap.

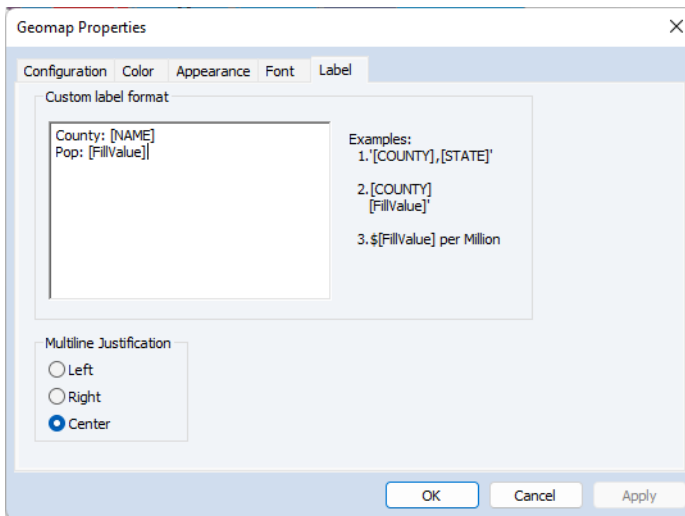
Note that the ability to attach labels to the shapes in a geomap is one of the most important tools for customizing the display of area data. In the simplest example, we may display the name of a geographic region in each area of a geomap display. For example, we may display the county names in our geomap display of the county areas in the State of California, USA.



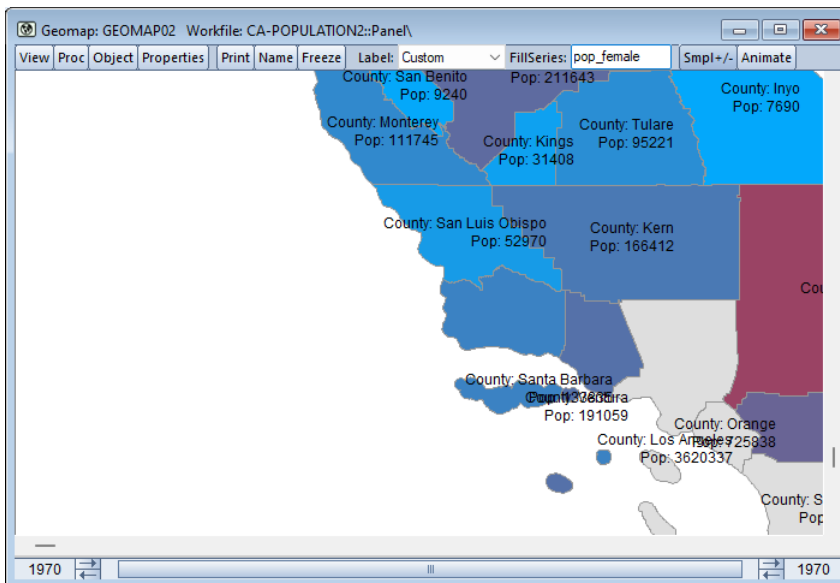
Here, the county names are one of several *attributes* of the geomap shapefile data, which pairs each shape with one or more pieces of information. Each shape in the file might have information on “Name”, “Population”, “Income”, “Population % change”, *etc.*

In EViews 12, you could display a geomap with a single label taken from a single attribute, as in the example above. So while you could instruct EViews to use the “Name” attribute to display county name, or the “Population” attribute to show population, you could not display labels showing both county name and population in each shape.

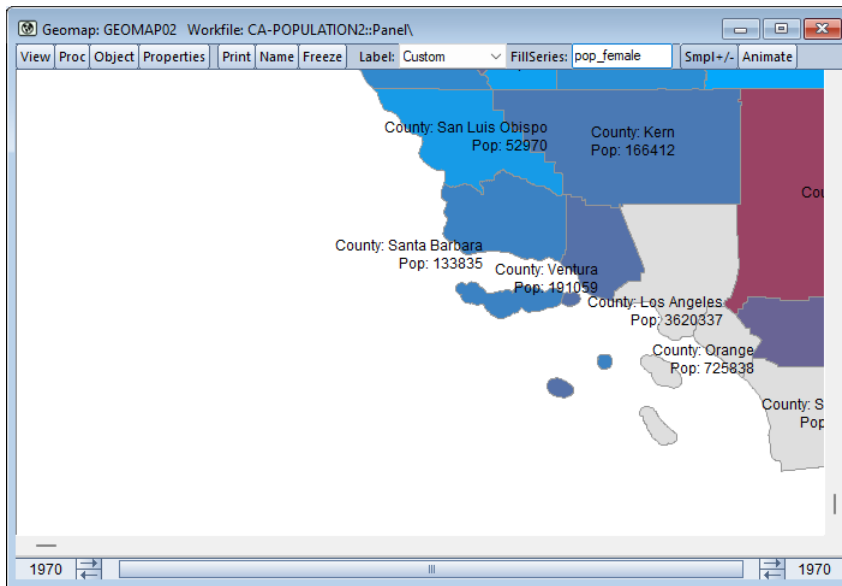
EViews 13 removes this single attribute restriction, allowing you to use multiple attributes to construct an area label. Furthermore, you may add custom text to your label, and apply custom formatting including, font family and size, and multi-line display.



Here we use the attribute “NAME” and the value of the workfile series POP_FEMALE to create a custom, multi-line label for each shape:



Should the default position of a label not be ideal, it may be adjusted manually. Simply click and drag a label to its desired location:



Note the improved labeling of the offshore areas in Santa Barbara, Ventura, and Los Angeles counties.

- See [Chapter 13. “Geomaps,”](#) on page 681 in *User’s Guide I* for updated documentation.

For command support, see:

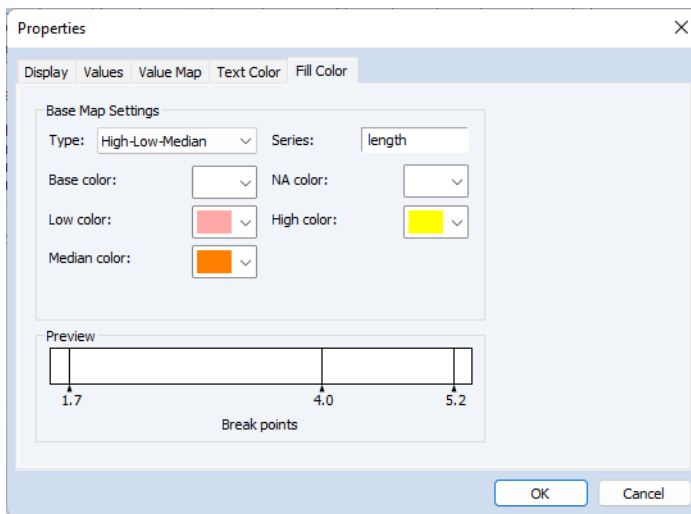
- [Geomap::setjust](#) (p. 318) in the *Object Reference* to set the display justification for multi-line area labels.
- [Geomap::setshapelabel](#) (p. 319) in the *Object Reference* to used attribute or value labels, or to create a custom label to use when labeling shapes.

High-Low-Median Colormap Preset

EViews supports colormaps where you can define sets of rules that translate numeric values into colors. These colormaps may then be used when setting the text or fill color for series or group spreadsheets, or the fill colors in geomaps.

EViews 12 offers a number of predefined colormaps for common settings, such as the **Positive-negative** colormap which negative values will be displayed as red and positive values will be displayed as black. In addition to EViews 12 pre-specified presets, EViews 13 includes a new preset which allows users to identify the high, low, and median observations.

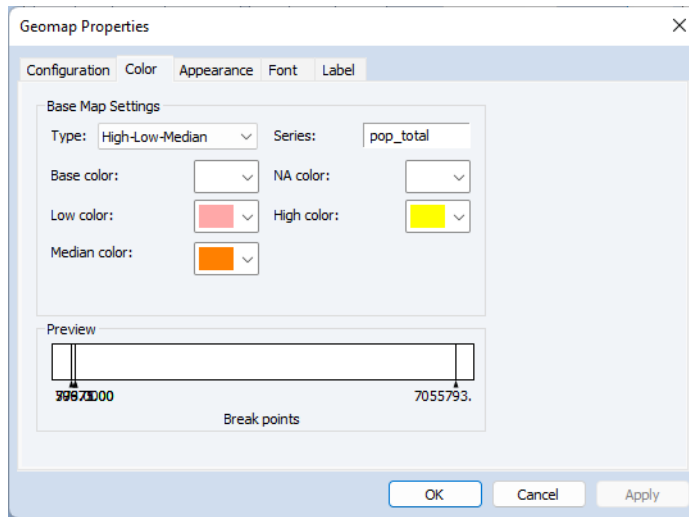
To apply this preset to a displayed series spreadsheet, click on **Properties** then select the **Text Color** or **Fill Color** tab as desired:



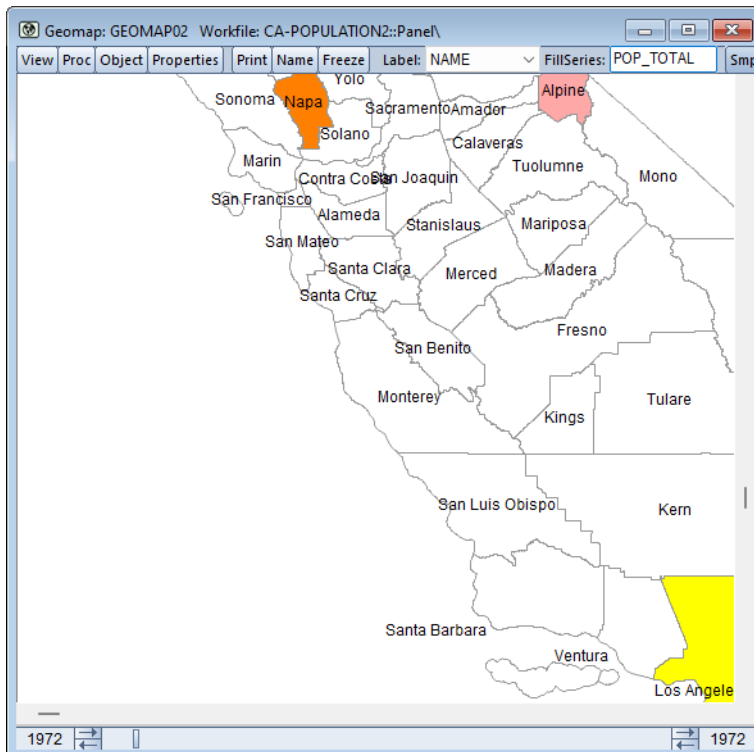
Choose **High-Low-Median** in the **Type** dropdown, and choose the colors appropriately. Click on **OK** to display the spreadsheet with the colormap applied:

View	Proc	Object	Properties	Print	Name	Freeze	Default	Sort	Edit+/-	Smp1+
205			1.7							
206			4.2							
207			4.3							
208			1.7							
209			4.4							
210			4.2							
211			2.2							
212			4.7							
213			4.0							
214			1.8							
215			4.7							
216			1.8							
217			4.5							
218			2.1							
219			4.2							
220			2.1							
221			5.2							
222			2.1							

For geomaps you will click on **Properties**, then select the **Color** tab and choose **High-Low-Median** in the **Type** dropdown,



Click on **OK** to display the geomap with the colormap applied:



Applying a high-low-median colormap is done using the existing `setfillcolor` proc and the “t=hilo” option:

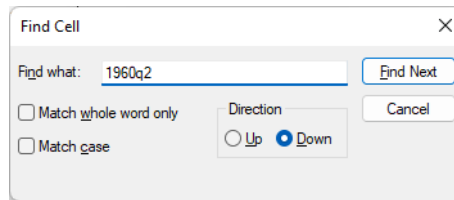
```
geomap01.setfillcolor(t=hilo) mapser(POP_TOTAL) min(-352258.35)
max(7408557.35) highclr(@RGB(255,168,168))
lowclr(@RGB(255,255,128)) medianclr(@RGB(255,190,96))
```

- See [Geomap::setfillcolor \(p. 311\)](#) in the *Object Reference* for command documentation.

Table Search

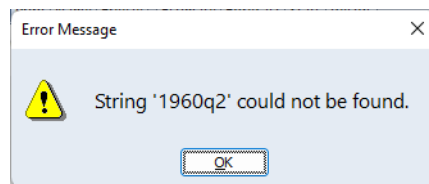
EViews 13 introduces the ability to search cells within a table, both interactively and in programs.

To locate a particular value or string in a table interactively, first ensure that the table window is active, then press `Ctrl + F` or select **Edit/Find** from the main menu to display the **Find** dialog.



When searching, you have the option to match the whole word, match case, and the specify the search direction. After making any selections, press the **Find Next** button to locate a matching cell.

If no match is found, EViews will produce an error message,



If a match is found, the first located corresponding cell within the table will be selected:

View	Proc	Object	Print	Name	Edit+/-	CellFmt	Grid+/-	Title	Comments+/-	
		A	B	C	D	E	F			
27		1958Q1	113.4750	503.1500	139.6330	0.225529				
28		1958Q2	114.6000	506.1750	139.6550	0.226404				
29		1958Q3	118.0167	518.0750	143.1710	0.227799				
30		1958Q4	121.2750	530.1500	144.1120	0.228756				
31		1959Q1	124.0750	541.2500	145.8600	0.229238				
32		1959Q2	127.3250	555.8250	146.1400	0.229074				
33		1959Q3	127.4000	555.3500	147.3960	0.229405				
34		1959Q4	128.4250	557.7500	145.4830	0.230256				
35		1960Q1	131.8250	569.8000	145.6990	0.231353				
36		1960Q2	131.5250	566.3750	145.5990	0.232223				
37		1960Q3	132.2250	567.0750	147.5250	0.233170				
38		1960Q4	130.9750	559.6500	146.5270	0.234030				
39		1961Q1	132.0250	562.9250	149.3190	0.234534				
40		—								

Subsequent **Find Next** searches will select the next matching cell. If the end of the table is reached, EVIEWS will resume the search from the beginning or end of the table, depending on the search direction.

Perhaps more importantly, EVIEWS 13 provides a new table data member `@find`, that provides a string containing cell identifiers which satisfy a boolean condition. The condition may be expressed as a mathematical (using numerically interpreted values of the cell contents) or string comparisons, using cell references.

The table data member,

```
@find("expression")
```

where expression can be a mathematical or string expression, returns a string containing the cell identifiers which meet the bool expression.

For example,

```
string s = tab1.@find("[b1:c15]>0.3")
```

returns the list of cells between cell B1 and C15 which have a value greater than 0.3.

```
string s = tab1.@find("[a1:e67]== \"1949q4\"")
```

returns the list of cells between A1 and E67 which match the string “1949q4”, ignoring case (note: string comparisons in `@find` ignore case).

```
string s = tab1.@find("[@all]==0.5")
```

returns the list of cells in the table that are equal to 0.5.

```
string s = tab1.@find("@instr([@all],\"in\")")
```

returns the list of cells in the table that contain the substring ‘in’.

```
string s = tab1.@find("[b3:c5]<[d5]")
```

returns the list of cells between B3 and C5 that have a value less than the value of cell D5

```
string s = tabl.@find("@left([@all], 2)=="cp"")
```

returns list of cells in the table that start with the string “cp”.

Conditional Table Cells

Many EViews table customization procedures allow you to specify ranges of cells. For example, `Table::setfillcolor` in the *Object Reference* allows you to specify the background color for the specified cells.

There are cases when you may wish to perform operations on cells in a table, but only under certain conditions. Suppose, for example, that you have a table with annual data on each row and you want to identify each row where there is a decrease in value in column 3 when compared to the value for the previous row (year). The goal is to apply custom text or cell formatting to these rows.

Fortunately, EViews allows you to identify the desired cells using a conditional target range specification comprised of a *target_range* and a *boolean_expression*. You may specify the conditional target cells using the syntax:

```
target_range if boolean_expression
```

where *target_range* gives the range of potential cells of interest, “if” is a keyword, and *boolean_expression* provides indicators for which of the *target_range* cells to use.

For example, the command

```
tabl.setfillcolor(A2:A7 if [A2:A7]<[A1:A6]) red
```

will only set the fill color to red for the target cell range, if the decreasing value condition holds.

- See “[Conditional Table Cells](#)” on page 68 in the *Command and Programming Reference* for detailed discussion.

Similarly, the table `@find` data member allows you to obtain a string list containing the cells in a table range that satisfy a condition. The syntax for `@find` requires a *find_expression*:

```
table_name.@find(find_expression)
```

where *find_expression* simultaneously specifies the range of cells to consider and the boolean expression for whether each cell should be used.

For example the command

```
string s = tabl.@find("[b1:c15]>0.3")
```

returns the list of cells between cell B1 and C15 which have a value greater than 0.3.

```
string s = tabl.@find("@instr([@all],\"in\")")
```

returns the list of cells in the table that contain the substring ‘in’.

```
string s = tab1.@find("[b3:c5]<[d5]")
```

returns the list of cells between B3 and C5 that have a value less than the value of cell D5

- See “Table Search” on page 58.

Fixing Rows and Columns in Tables

EViews 13 now allows you to fix rows and columns in the display of a table so that these rows and columns are always in view as you scroll the remaining cells of the table.

There are times when it is useful to lock one or more rows or columns in a table so that those rows and columns are always in view. Depending on the contents of the table, the first few rows or columns in your table may contain identifying information about the data cells within the table. When searching for a specific cell or set of cells, fixing the display of these identifiers makes this task much easier. This is especially useful for tables containing a large number of columns or rows.

This situation often occurs, for example, when working with tables where the first row contains column names and the first column contains observation IDs. In the case when the table is scrolled both vertically and horizontally such that the 20th column and the 100th row of a table is in view it may be helpful to see the column names and observation IDs to have provide context to the contents of the cell.

Fixing columns and rows in a table can be done by pressing the **Proc** button in a table object button bar. Then,

- To fix only the first row, select **Fix Row-Column/Fix 1 Row**. If more than 1 row is to be fixed, select a cell in the first row after the last desired fixed row and then from the **Proc** menu select **Fix Row-Column/Fix # Rows**. For example, to fix the first 3 rows of table, select a cell in the fourth row and then select **Fix Row-Column/Fix 3 Rows**.
- The process for fixing columns is the same as fixing rows with the obvious exception of selecting
- **Fix Row-Column/Fix #Columns** - Similarly, after navigating through the proc menu, the columns proceeding the current selected cell will be fixed. Note the fixed columns and rows will be denoted in the table by a black cell border.

In the image below which fixes the first row and column of the table, you can see the date column (column A) and the series names (row 1) are in view despite the table being scrolled vertically 123 rows and 13 columns where row 124 is the first row, and column J is the first column displayed.

The fixed columns are denoted by a black cell border that extends from the top to bottom of the table and the fixed rows are separated by a similar black border that extends from the left to right of the table.

With fixed headers and rows, we can readily identify the selected cell as being the value for LC for 1979q3.

View	Proc	Object	Print	Name	Edit+/-	CellFmt	Grid+/-	Title	Comments+/-
1	A	J	K	L	M	N			
		LC	LY	DLC	DLY				
124	1977Q2	753.3159	762.5400	0.283980	1.113716				
125	1977Q3	754.3273	764.3435	1.011454	1.803496				
126	1977Q4	755.7055	765.1025	1.378169	0.759015				
127	1978Q1	756.1642	766.3032	0.458671	1.200672				
128	1978Q2	758.1108	767.9113	1.946611	1.608089				
129	1978Q3	758.5941	768.5427	0.483326	0.631467				
130	1978Q4	759.5739	769.7121	0.979801	1.169395				
131	1979Q1	759.9652	770.3730	0.391255	0.660850				
132	1979Q2	759.7948	769.9163	-0.170359	-0.456698				
133	1979Q3	760.4845	770.2375	0.689660	0.321264				
134	1979Q4	760.9864	770.2375	0.501751	0.000000				
135	1980Q1	760.8573	770.7782	-0.128922	0.540639				
136	1980Q2	758.7868	768.9646	-2.070511	-1.813615				
137	1980Q3	759.9051	769.9435	1.118291	0.978905				
138	1980Q4	760.8672	771.3651	0.962141	1.421584				
139	1981Q1	761.2287	771.5525	0.361523	0.187431				
140	1981Q2	761.2040	771.1997	-0.024720	-0.352840				
141	1981Q3	761.6530	772.4402	0.448950	1.240591				
142	1981Q4	761.0853	772.0329	-0.567693	-0.407385				
143	1982Q1	761.6382	771.6773	0.552922	-0.355603				
144	1982Q2	762.1195	772.3518	0.481316	0.674567				
145									

In this second example, the fixed first row and column of the table makes it easy to identify the LINCOMEP values for Turkey from 1960 to 1978 despite the fact that the displayed values are for the 288th row and column F of the table.

Table: UNTITLED Workfile: GASOLINE:Gasoline\

View Proc Object Print Name Edit+/- CellFmt Grid+/- Title Comments+/-

7.801144247

	A	E	F	G	H	I
1		LGASPCAR	LINCOMEP	LRPMG		
287	SWITZERL - 78 -	4.05	-5.817	-1.032		
288	TURKEY - 60 -	6.13	-7.801	-0.2534		
289	TURKEY - 61 -	6.106	-7.787	-0.3425		
290	TURKEY - 62 -	6.085	-7.836	-0.4082		
291	TURKEY - 63 -	6.075	-7.631	-0.225		
292	TURKEY - 64 -	6.065	-7.627	-0.2522		
293	TURKEY - 65 -	5.823	-7.622	-0.2935		
294	TURKEY - 66 -	6.157	-7.511	-0.3564		
295	TURKEY - 67 -	6.044	-7.461	-0.3352		
296	TURKEY - 68 -	6.077	-7.421	-0.3651		
297	TURKEY - 69 -	5.721	-7.389	-0.2985		
298	TURKEY - 70 -	5.722	-7.324	-0.3988		
299	TURKEY - 71 -	5.67	-7.182	-0.3046		
300	TURKEY - 72 -	5.579	-7.084	-0.5464		
301	TURKEY - 73 -	5.686	-7.067	-0.6916		
302	TURKEY - 74 -	5.428	-7.013	-0.3397		
303	TURKEY - 75 -	5.427	-6.954	-0.5379		
304	TURKEY - 76 -	5.313	-6.91	-0.7514		
305	TURKEY - 77 -	5.313	-6.894	-0.9555		
306	TURKEY - 78 -	5.141	-6.889	-0.3529		
307	U.K. - 60 -	4.1	-6.187	-0.3911		
308	U.K. - 61 -	4.099	-6.169	-0.4519		
309						

To unfix any fixed rows or columns, simply press the **Proc** button and either select **Fix Row-Column/Remove Fixed Columns** or **Remove Fixed Rows**. Note: the cell selection is ignored when unfixing rows or columns.

Note that fixed rows and columns are saved with the table when the workfile is saved to disk.

- See [Table::fixcol \(p. 986\)](#), [Table::fixrow \(p. 987\)](#), and [Table::fixrowcol \(p. 987\)](#) in the *Object Reference*.

Fixing the beginning set of columns in table is accomplished via the table name followed by the `fixcol` proc and the number of columns to be fixed. This procedure force the number of specified columns to always be in view regardless of the horizontal scroll position. Once fixed, a black cell border will separate the fixed and unfixed rows and columns.

Similarly, to fix the beginning set of rows in the table use the `fixrow` keyword followed by the number of rows to be fixed.

For example, the commands

```
tab1.fixcol 2
tab1.fixrow 1
```

will fix the first two columns and the first row of the table.

Alternately you may fix both the row and column using the single command

```
tabl.fixrowcol 1 2
```

You may clear the fixed rows and columns using

```
tabl.fixcol 0
```

```
tabl.fixrow 0
```

or

```
tabl.fixrowcol 0 0
```

Econometrics and Statistics

EViews 13 offers a exciting new additions and improvements to its set of econometric and statistical features. The following is a brief outline of the most important new features, followed by additional discussion and pointers to full documentation.

ARDL Estimation

EViews 13 offers improvements to existing tools for analyzing data using Autoregressive Distributed Lag Models (ARDL), featuring estimation of Nonlinear ARDL (NARDL) models which allow for more complex dynamics, with explanatory variables having differing effects for positive and negative deviations from base values.

The classical ARDL framework assumes that the long-run cointegrating relationship is a symmetric linear combination of regressors. While this is a natural starting assumption, it does not match the behavioral finance and economics literature approach to modeling non-linearity and asymmetry (Kahneman, Tversky, and Shiller, 1979). In response, Shin (2014) proposes a nonlinear ARDL (NARDL) framework in which short-run and long-run nonlinearities are modeled as positive and negative partial sum decompositions of the explanatory variables.

From the main EViews menu, click on **Quick/Estimate Equation...** or type the command `equation` in the command line to open the equation dialog. Then select the **ARDL - Autoregressive Distributed Lag Models (including NARDL)** from the **Method** dropdown to display the ARDL dialog:

In the **Linear dynamic specification**, you should enter a list of variables consisting of the dependent variable followed by any symmetric ARDL distributed lag regressors. At a minimum, the edit field must contain the dependent variable.

Exogenous regressors, including deterministics, may be specified in the **Fixed regressors specifications** section. Trend regressors corresponding to the five deterministic cases discussed (**None**, **Restr. constant**, **Constant**, **Restr. trend**, **Trend**) may be specified using the **Trend specification** dropdown. All other exogenous regressors (those apart from the constant and the trend) should be specified in the **Fixed regressors** edit field.

Asymmetric distributed lag regressors may be listed under **Asymmetric dynamic specifications**. In particular, the **Long-run and short-run** edit field may be used to specify regressors which are asymmetric in both the long-run and short-run. Regressors which are asymmetric only in the long-run may be specified in the **Long-run only** edit field, while those which are asymmetric exclusively in the short-run are specified in the **Short-run only** edit field.

Dependent Variable: DLOG(REALCONS)
 Method: ARDL
 Date: 05/04/22 Time: 15:37
 Sample (adjusted): 1950Q3 2000Q4
 Included observations: 202 after adjustments
 Max. dependent lags: 1 (Fixed)
 Fixed-lag linear regressors: LOG(REALGDP)
 Fixed-lag dual non-linear regressors: LOG(REALGOVT)
 Deterministics: Restricted constant and no trend (Case 2)
 Selected model: ARDL(1,1,1)

Variable	Coefficient	Std. Error	t-Statistic	Prob.*
LOG(REALCONS(-1))	-0.126939	0.037213	-3.411168	0.0008
LOG(REALGDP(-1))	0.133061	0.040618	3.275890	0.0012
@CUMDP(LOG(REALGOVT(-1)))	-0.004936	0.008563	-0.576444	0.5650
@CUMDN(LOG(REALGOVT(-1)))	-0.018402	0.020006	-0.919856	0.3588
C	-0.100803	0.063927	-1.576846	0.1165
DLOG(REALGDP)	0.643050	0.050018	12.85647	0.0000
@DCUMDP(LOG(REALGOVT))	-0.149865	0.042682	-3.511242	0.0006
@DCUMDN(LOG(REALGOVT))	-0.114135	0.103098	-1.107051	0.2696
R-squared	0.473136	Mean dependent var	0.008782	
Adjusted R-squared	0.454125	S.D. dependent var	0.008864	
S.E. of regression	0.006549	Akaike info criterion	-7.180217	
Sum squared resid	0.008320	Schwarz criterion	-7.049196	
Log likelihood	733.2019	Hannan-Quinn criter.	-7.127206	
F-statistic	24.88805	Durbin-Watson stat	2.584413	
Prob(F-statistic)	0.000000			

*Note: p-values and any subsequent tests do not account for model selection.

- For full discussion, see [Chapter 29. “Linear and Nonlinear ARDL,”](#) on page 321, in *User’s Guide II*.
- See [Equation::ardl \(p. 61\)](#) in the *Object Reference* and [ardl \(p. 322\)](#) in the *Command and Programming Reference* for updated command documentation.

In addition to the support for estimating nonlinear ARDL specifications, EViews 13 offers a number of new ARDL diagnostics. These diagnostics are designed to help you examine the dynamic behavior of your variables, the long-run properties of your specification, and the appropriateness of your model.

- See the summary of new EViews 13 ARDL features in [“Diagnostics in ARDL”](#) on page 77.

Pool Mean Group (PMG) Estimation

In large sample panel datasets, a popular approach for analyzing dynamic data is the Pooled Mean Group (PMG) estimator of Pesaran, Shin and Smith (1999) (PSS). This model takes the cointegration form of the simple ARDL model and adapts it for a panel setting by allow-

ing the intercepts, short-run coefficients, and cointegrating terms, to differ across cross-sections.

EViews 13 extends the estimation of PMG models to support:

- a greater range of deterministic trend specifications (including those with fully restricted constant and trend terms)
- specifications with asymmetric regressors.

To estimate a Panel ARDL/PMG model in EViews, open the equation dialog by selecting **Quick/Estimate Equation...**, or by selecting **Object/New Object.../Equation** and selecting **PMG/ARDL** from the **Method** dropdown menu. EViews 13 will then display the new ARDL estimation dialog:

While much of the dialog is familiar from earlier versions of EViews, notably different in the EViews 13 PMG dialog are additional entries in the **Trend specification** corresponding to restricted intercept and trend specifications, and new edit fields for specifying **Asymmetric dynamic specifications**. The **Trend specification** upgrades allow for a wider range of long and short-term dynamics, while the latter fields support asymmetric variables in the PMG estimator:

- See “Pooled Mean Group ARDL Estimation” on page 1221 and “Pooled Mean Group Example” on page 1238 in *User’s Guide II* for discussion of EViews 13 panel PMG estimation.
- See `Equation::ardl` (p. 61) in the *Object Reference* and `ardl` (p. 322) in the *Command and Programming Reference* for updated command documentation.
- See also the discussion of non-panel ARDL estimation in “Background” on page 321 and “Estimating ARDL and NARDL in EViews” on page 326 in *User’s Guide II* for background detail.

EViews 13 also offers a number of new PMG diagnostics and tests to help you examine the dynamic behavior of your variables, the long-run properties of your specification, and the appropriateness of your model.

- See the summary of new EViews 13 PMG post-estimation tools in “Diagnostics in Panel ARDL/PMG” on page 86.

Difference-in-Difference Estimation

Difference-in-difference (DiD) estimation is a popular method of causal inference that allows estimation of the average impact of a treatment on individuals.

EViews 13 offers tools for estimation of the DiD model using the common two-way fixed-effects (TWFE) method, as well as post-estimation diagnostics of the TWFE model, such as those by Goodman-Bacon (2021), Callaway and Sant’Anna (2021), and Borusyak, Jaravel, and Spiess (2021).

To estimate a DiD model in EViews, bring up the equation dialog by clicking on **Object/New Object.../Equation** or **Quick/Estimate Equation...** from the main menu bar in your panel workfile. EViews will detect the presence of your panel structure and in place of the standard equation dialog will open the panel equation estimation dialog. Select **DiD – Difference-in-Difference** in the **Method** dropdown display the DiD dialog:

The screenshot shows the 'Equation Estimation' dialog box with the following details:

- Equation specification:** The text field contains 'l_homicide c'.
- Treatment variable:** The text field contains 'post'.
- Estimation settings:** The 'Method:' dropdown is set to 'DID - Difference-in-Difference'. The 'Sample:' text field contains '2000 2010'.

In the **Equation specification** edit field you should enter the dependent variable followed by any exogenous regressors *apart from the treatment variable*.

The treatment variable should be entered in the **Treatment Variable** edit field. The treatment series should be a binary variable indicating whether the individual has been treated (*i.e.*, is 1 if the observation in a treatment group which is post-treatment date for that group, and 0 otherwise).

The **Options** tab contains a single **Coefficient name** edit field that allows you to change the default coefficient vector.

Click on **OK** to perform the difference-in-difference estimation and display the output:

Dependent Variable: L_HOMICIDE
 Method: Difference-in-Difference
 Date: 05/13/22 Time: 10:56
 Periods included: 11
 Cross-sections included: 50
 Total panel (balanced) observations: 550

Variable	Coefficient	Std. Error	t-Statistic	Prob.
POST	0.081812	0.064117	1.275980	0.2026
R-squared	0.910576	Mean dependent var		1.405760
Adjusted R-squared	0.899604	S.D. dependent var		0.590154
S.E. of regression	0.186992	Akaike info criterion		-0.411237
Sum squared resid	17.09842	Schwarz criterion		0.066772
Log likelihood	174.0902	Hannan-Quinn criter.		-0.224439
F-statistic	82.98904	Durbin-Watson stat		1.469473
Prob(F-statistic)	0.000000	Parallel trend stat		0.820393
Prob(P. trend)	0.411992			

- See [Chapter 55. “Difference-in-Difference Estimation,”](#) on page 1275 in *User’s Guide II* for additional discussion.
- See [Equation::did \(p. 103\)](#), [Equation::didcs \(p. 104\)](#), [Equation::didgbdecomp \(p. 105\)](#), [Equation::didmakeeq \(p. 105\)](#), and [Equation::didtrends \(p. 106\)](#) in the *Object Reference* for command documentation.

Enhanced VEC Estimation

Estimation of Vector Error Correction (VEC) models using reduced rank regression (RRR) has been a staple of EViews for several versions. Previous versions of EViews supported a variety of built-in specifications for deterministic trends, and allowed users to add unrestricted exogenous regressors to account for short-term dynamics.

One important limitation of the existing tools for VEC estimation was the inability of users to add arbitrary exogenous variables to the cointegrating relation since the set of admissible restricted regressors was limited to the endogenous variables and potentially an intercept, and possibly a linear trend term.

EViews 13 removes this limitation on restricted regressors and improves on the prior treatment of exogenous variables in two distinct ways:

- by providing new deterministic trend assumption presets which allow for more flexible specification of restricted and unrestricted deterministic coefficients, and
- by allowing users to add exogenous variables that are restricted to the cointegrating relation, variables that are unrestricted, and variables that in both the short and

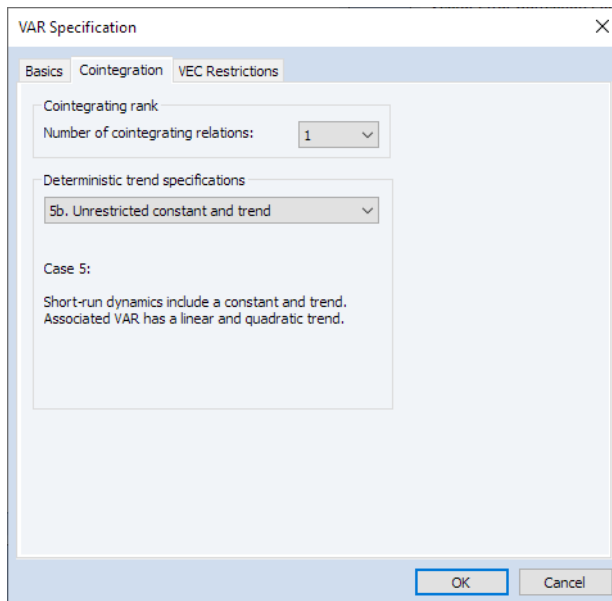
long-run equations, with an orthogonality assumption used to obtain the different coefficients.

To estimate a VEC using the new features, from the main application menu of an existing var object, click on the **Estimate** button to open the **VAR Specification** estimation dialog. Alternately, you may create a new VAR object by selecting **Object/New Object...** group, then selecting **VAR**. Once the dialog appears, select **Vector Error Correction** in the **Method** drop-down menu to display the VEC estimation dialog:

The screenshot shows the 'VAR Specification' dialog box with the 'VEC Restrictions' tab selected. The 'Method' dropdown is set to 'Vector Error Correction'. The 'Lag intervals for diff. endog' field contains '1 2'. The 'Estimation sample' field contains '1959m2 1995m04'. The 'Exogenous variables' section is divided into three categories: 'Short-run (outside cointegrating equation)' with 'ppl', 'Long-run (inside cointegrating equation)' with 'tb10y', and 'Both long-run and short-run' with 'urate'. The 'Endogenous variables' field contains 'm1 gdp tb3'. The 'OK' and 'Cancel' buttons are at the bottom right.

The **Exogenous variables** section of the main estimation dialog has fields for variables that are **Short-run (outside cointegrating equation)** only, variables that are **Long-run (inside the cointegrating equation)** only, and variables that are **Both long-run and short-run**.

Further, the **Deterministic trend specifications** dropdown on the **Cointegration** tab offers additional predefined deterministic trend specifications.



Selecting a specification in which both intercept and trend are short-run only and clicking on **OK** produces estimates using the new deterministics:

The screenshot shows the 'Vector Error Correction Estimates' window in EViews. The window title is 'Var: VEC3 Workfile: VAR1:=Var1'. The window contains the following text:

Vector Error Correction Estimates
Date: 05/30/22 Time: 13:03
Sample (adjusted): 1959M04 1982M03
Included observations: 276 after adjustments
Standard errors in () & t-statistics in []
Lags interval (in first differences): 1 to 2
Endogenous variables: M1 GDP TB3
Exogenous variables (short-run only): PPI
Exogenous variables (long-run only): TB10Y
Exogenous variables (short-run and long-run): URATE
Deterministic assumptions: Case 5: Constant and trend both belong to short-run regressors.

Cointegrating Eq:	CointEq1
M1(-1)	1.000000
GDP(-1)	-0.170876 (0.02142) [-7.97831]
TB3(-1)	-11.83218 (1.18748) [-9.96412]
TB10Y	11.01109 (1.63177) [6.74795]
URATE	-2.626517

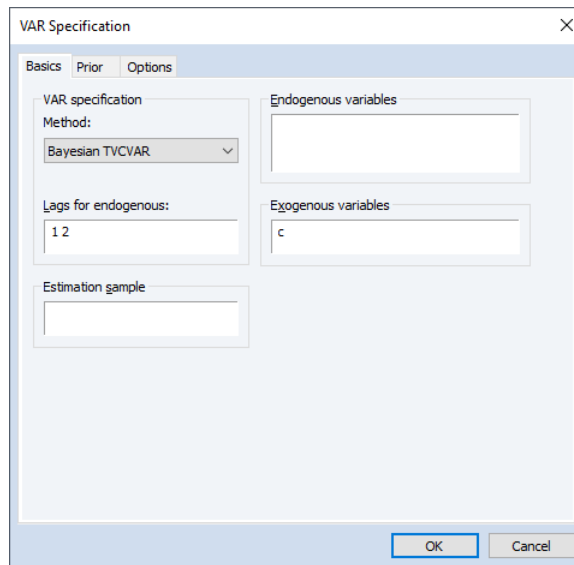
- See [Chapter 45. “Vector Error Correction Models \(VECMs\),”](#) on page 921 in *User’s Guide II* for additional discussion.

- See `var:ec` (p. 1065) in the *Object Reference* for updated command documentation.
- See also the summary of “Cointegration Testing” on page 76 for related improvements in EViews 13 cointegration testing.

Bayesian Time-varying Coefficient Vector Autoregression

The time-varying coefficients vector autoregression, or TVCVAR, is a nonlinear VAR model with coefficients that evolve smoothly over time. EViews 13 supports Bayesian TVCVAR, or BTVCVAR. The BTVCVAR is popular even among those who do not identify as Bayesian because the prior provides a convenient way to induce shrinkage in a model that needs it.

To estimate BTVCVAR in EViews, click on **Quick/Estimate VAR...** or run `var` in the command window. This will open the **VAR Specification** dialog. Select **Bayesian TVCVAR** from the **VAR type** dropdown menu. The dialog should now have the **Basics**, **Prior**, and **Options** tabs.



The screenshot shows the 'VAR Specification' dialog box with the 'Basics' tab selected. The 'Method' dropdown is set to 'Bayesian TVCVAR'. The 'Lags for endogenous:' field contains '1 2'. The 'Exogenous variables:' field contains 'c'. The 'Endogenous variables:' field is empty. The 'Estimation sample' field is empty. The 'OK' and 'Cancel' buttons are at the bottom right.

Endogenous variables, lags, exogenous variables, and the estimation sample are set in the **Basics** tab. Lags are required to be contiguous.

EViews gives users the option of setting aside a subset of the estimation sample for the purpose of specifying a prior distribution. The observations that go towards specifying the prior comprise the prior sample. The remaining observations make up the data sample, which is used to update the prior.

VAR Specification

Basics Prior Options

Hyperparameters

T0:	0	Prior sample size
tau0:	5.0	Prior scaling factor for b0
tau1:	1.0	Prior scaling factor for S
tau2:	0.01	Prior scaling factor for Q
nu1:	5.0	Prior dof for S
nu2:	5.0	Prior dof for Q

OK Cancel

Prior hyper-parameters are set inside the **Prior** tab. To help with setting the prior, EViews maps the BTVCVAR prior hyper-parameters to a set of six scalar quantities. Users set these scalars to specify a prior sample, control the variability of the time-varying coefficients, *etc.*

VAR Specification

Basics Prior Options

Sampler

Burn-in size: 5000

Posterior sample size: 5000

Thinning size: 1

Number of subchains: 11

RNG type: Knuth

Seed: Clear

Display

Point estimate type: Posterior median

Show credibility intervals

Credibility levels: 0.95

Use lines

Smoother

Smoothing method: Cholesky factor algorithm (CFA)

Stability

Stability method: None

Maximum number of attempts: 100

OK Cancel

There are four categories in the **Options** tab: **Sampler**, **Display**, **Smoother**, and **Stability**.

The **Sampler** options determine how the posterior sample is generated.

- The **burn-in size** is the number of initial draws to discard. It is specified as a count. The burn-in process gives the underlying Markov chain time to converge to the posterior distribution.
- The **posterior sample size** is used to determine how many posterior draws are used to carry out post-updating procedures (estimation, forecasting, impulse responses analysis, etc.).
- The **thinning size** is used to thin the Markov chain. A thinning size of r indicates that every r -th draw is stored. For example, no thinning occurs when r is set to 1, and every other draw is stored when r is set to 2. By definition, thinning does not apply to burn-in draws.
- The **seed** field is used to set the random seed for the posterior simulator. EViews will generate a seed automatically if the user does not specify one. Click **Clear** to clear the seed field.
- The **number of subchains** field determines how many subchains are used when the posterior sample gets regenerated. Regeneration is typically much faster than initial generation since subchains can be run in parallel.

Display options determine what to report as estimation results. Users can pick either posterior median or posterior mean as their point estimate. The point estimate type selected here will be applied to the coefficients, the observation covariance matrix, and the errors. Users can also display equal-tailed credibility intervals (bands) at one or more credibility levels for the coefficients. To do so, check the box next to **Show credibility intervals**. Bands use shading by default. To use lines instead, check the box next to **Use lines**.

A simulation smoother can be selected from the dropdown menu under **Smoother**. EViews currently supports three simulation smoothers: the **Cholesky factor algorithm** (CFA), the **Kalman filter and smoother** (KFS), and the method of **McCausland, Miller, & Pelletier** (MMP). For more information, see McCausland, Miller, & Pelletier (2011) and the references therein.

To enforce stable VAR coefficients at each date within the data sample, select **Cogley & Sargent** from the dropdown menu under **Stability**. The **maximum number of attempts** threshold ensures that the sampler does not get stuck in an infinite loop in an attempt to obtain stable draws.

Once the BTVCVAR model has been specified, click on **OK** to run the posterior simulator. Progress is displayed in the bottom left corner of the EViews window. Once posterior simulation is complete, estimates and other statistics based on the posterior sample are computed.

Estimation results are presented in a spool-like object. The nodes under **Output Sections** in the left pane are used for navigation. For example, clicking on the **Summary** node will bring

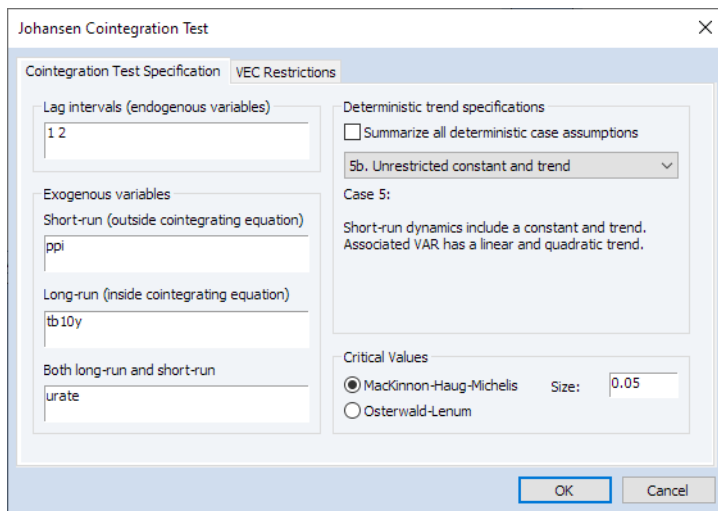
the summary table into focus. The checkboxes that appear below **Display Coefficients** are used to show/hide coefficient series that are associated to specific endogenous variables, lags, and exogenous variables. For example, unchecking the box next to **C** hides the coefficient series associated to the constant term in all graphs.

- See [Chapter 47. “Bayesian Time-varying Coefficients VAR Models,” beginning on page 987](#) in *User’s Guide II* for discussion.
- See [Var::btvcvar \(p. 1051\)](#) in the *Object Reference* for new command documentation.

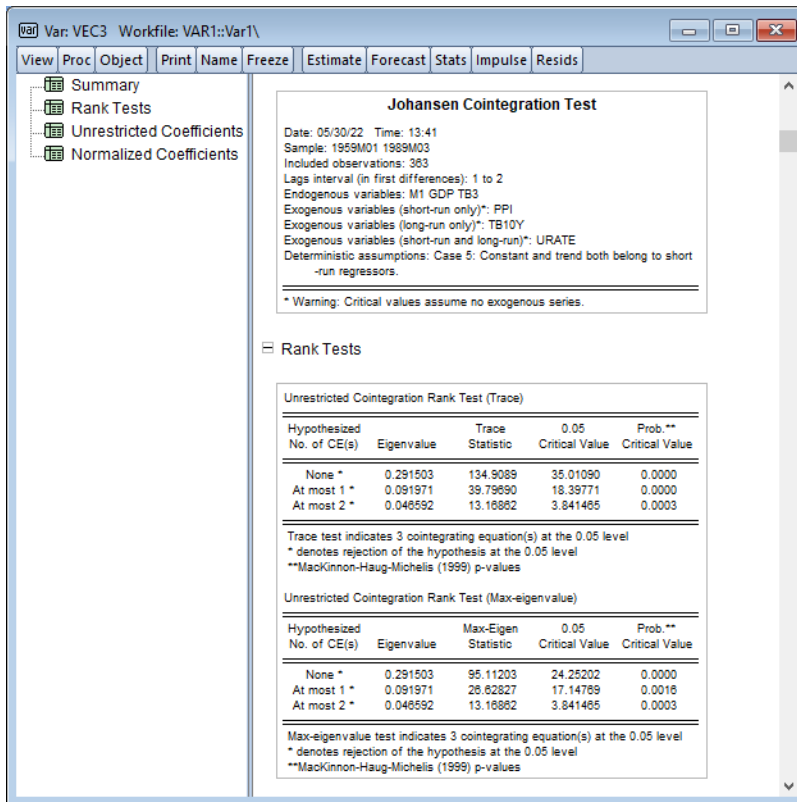
Cointegration Testing

The introduction of new deterministic trend settings and exogenous variable support in EViews 13 VEC estimation offers corresponding improvements in cointegration testing in both group and var object settings.

To perform a Johansen cointegration test to determine the rank that should be used in estimation of the VEC select **View/Cointegration Test/Johansen System Cointegration Test...**, from a group window, or **Views/Cointegration Test...** from an estimated VAR object window using. In the latter case, the test dialog will be pre-filled with the cointegration specification, if applicable:



All new settings allow you to specify new deterministic trend specifications with restricted constants and trends, and now allow you to specify exogenous variables that appear in the long-run cointegrating relation.



- See [Group::coint](#) (p. 397) for updated group object and [Var::coint](#) (p. 1058) in the *Object Reference* for updated var object command documentation.
- See also the summary “Enhanced VEC Estimation” on page 70 for related improvements in EViews 13 VEC estimation.

Diagnostics in ARDL

In addition to the support for nonlinear ARDL specifications, EViews 13 offers a number of new ARDL diagnostics. These diagnostics are designed to help you examine the dynamic behavior of your variables, the long-run properties of your specification, and the appropriateness of your model:

- Improved display tools make it easier to use the ARDL estimates to examine the short and long-run representations of the distributed lag regressors and cointegrating relationships. (“Enhanced Presentation of ARDL Results” on page 78.)

- EViews now produces *cumulative dynamic multipliers* (CDM) graphs showing cumulative marginal contribution of an explanatory variable to the dependent variable ([“Cumulative Dynamic Multiplier Graphs” on page 81](#)).
- Bounds tests for cointegration are now available as a view of your ARDL equation ([“Bounds Tests” on page 84](#)).
- You may evaluate symmetry restrictions on the coefficients of the asymmetric long and short-run regressors ([“Testing for Asymmetry” on page 85](#)).

Enhanced Presentation of ARDL Results

EViews 13 offers enhanced presentation of ARDL results, with an emphasis on highlighting the error correction results and the cointegrating relationship.

The new error correction spool output (**View/ARDL Diagnostics/Error-Correction Results**) shows two representations of the coefficients in the cointegrating relationship, allowing you to easily move between viewing the long-run relationship results in the Conditional Error Correction (CEC) and the Error Correction (EC) forms.

The CEC focuses on the natural division between long-run and short-run dynamics, with the former generated by regressors entering in levels/lags, and the latter by regressors in differences

Equation: EX2 Workfile: NARDL_EV13::Untitled

View Proc Object Print Name Freeze Estimate Forecast Stats Resids

Conditional Error Correction

Dependent Variable: DLOG(REALCONS)
Method: ARDL
Date: 05/04/22 Time: 15:37
Sample (adjusted): 1950Q4 2000Q4
Included observations: 201 after adjustments
Max. dependent lags: 3 (Fixed)
Fixed-lag linear regressors: LOG(REALGDP)
Static regressors: @EXPAND(@QUARTER, @DROPLAST)
Deterministics: Restricted constant and no trend (Case 2)
Selected model: ARDL(3,3)

Variable	Coefficient	Std. Error	t-Statistic	Prob.
LOG(REALCONS(-1))*	-0.118944	0.030474	-3.903191	0.0001
LOG(REALGDP(-1))	0.126497	0.032281	3.918624	0.0001
C	-0.109982	0.029236	-3.761208	0.0002
DLOG(REALCONS(-1))	-0.157714	0.069795	-2.259685	0.0250
DLOG(REALCONS(-2))	0.233853	0.068672	3.402444	0.0008
DLOG(REALGDP)	0.585088	0.051953	10.87699	0.0000
DLOG(REALGDP(-1))	0.047706	0.063725	0.748631	0.4550
DLOG(REALGDP(-2))	-0.190243	0.058922	-3.228753	0.0015
@QUARTER=1	-0.000259	0.001266	-0.204677	0.8380
@QUARTER=2	-0.000259	0.001259	-0.205412	0.8375
@QUARTER=3	0.000915	0.001256	0.728608	0.4671

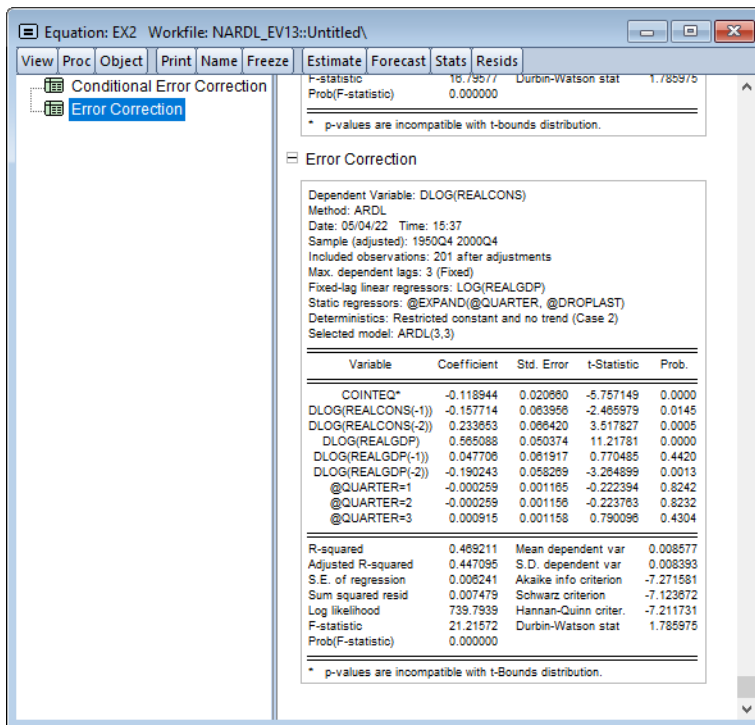
R-squared 0.485211 Mean dependent var 0.008577
Adjusted R-squared 0.441275 S.D. dependent var 0.008393
S.E. of regression 0.006274 Akaike info criterion -7.251681
Sum squared resid 0.007479 Schwarz criterion -7.070903
Log likelihood 739.7939 Hannan-Quinn criter. -7.178530
F-statistic 16.75577 Durbin-Watson stat 1.785975
Prob(F-statistic) 0.000000

* p-values are incompatible with t-bound distribution.

Error Correction

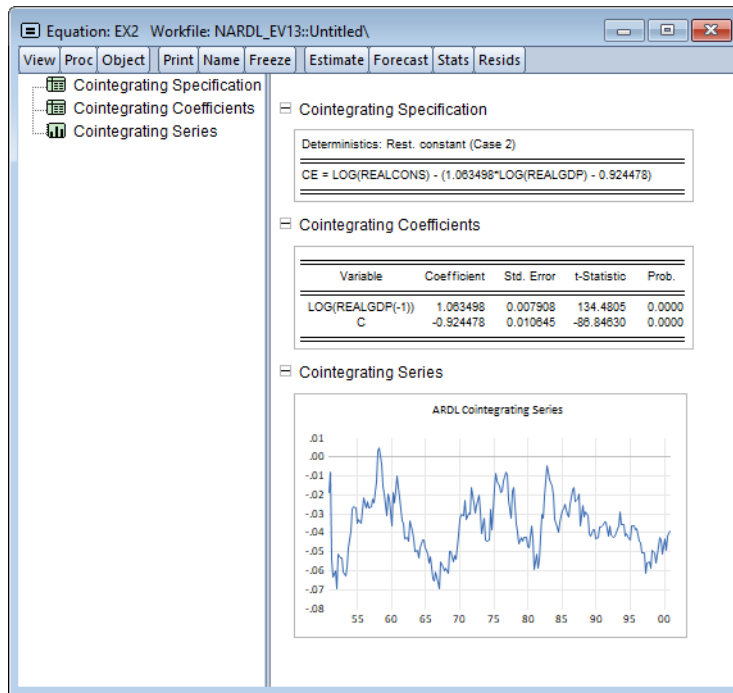
Dependent Variable: DLOG(REALCONS)
Method: ARDL

while the EC representation uses the cointegrating relation to highlight the speed of adjustment to long-run equilibrium,



- See “Error Correction Output View” on page 331 in *User’s Guide II*.
- See also `Equation::ecresults` (p. 109) in the *Object Reference* for command documentation.

The new cointegrating relation spool view (**View/ARDL Diagnostics/Cointegrating Relation**) lets you switch between viewing the specification, coefficient results, and a graphical display of the cointegrating relation.

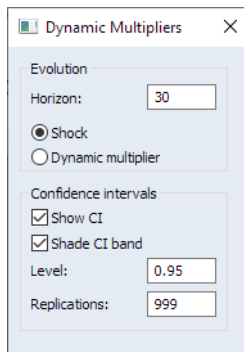


- See “Cointegrating Relation View” on page 333 in *User’s Guide II*.
- See also `Equation::cointrel` (p. 95) in the *Object Reference* for command documentation.

Cumulative Dynamic Multiplier Graphs

To display cumulative dynamic multiplier graphs for each of the explanatory variables in an EViews equation estimated using ARDL, click on **View/ARDL Diagnostics/Dynamic Multiplier Graph...**

EViews will open a dialog containing display and computation settings:

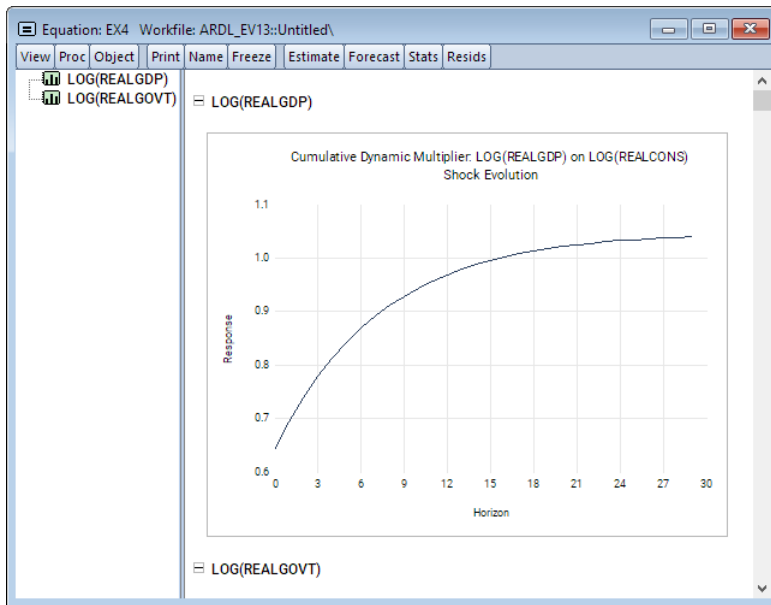


- You may enter the horizon length h (number of periods to compute the multipliers) in the **Horizon** edit field.
- For NARDL models, you will be offered the opportunity to display confidence intervals for the computed absolute difference between the positive and negative components for a given regressor. CIs are not available for linear ARDL specifications.

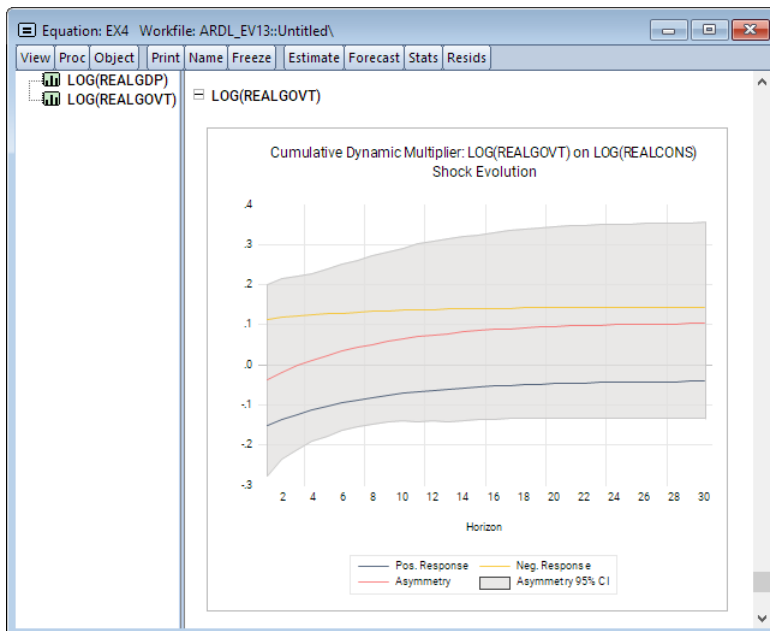
You may check the **Show CI** to display the CIs, and **Shade CI band** to display the CIs as bands instead of lines. The **Level** edit field controls the size of the CI, and the **Replications** governs how many replications to use in resampling for computing the CI.

Click on **OK** to continue. EViews will open a spool view, with each node in the spool containing the CDM graph corresponding to one of the explanatory variables.

For symmetric linear models, each graph contains a CDM along with a dashed horizontal line denoting the long-run value.



For asymmetric nonlinear models, each graph will show the positive and negative responses and limit values, along with a line showing the absolute value of the difference between the two, and if requested, a CI for the absolute difference:



- See “[Dynamic Multipliers View](#)” on page 338 in *User’s Guide II*.
- See `Equation::dynmult` (p. 108) in the *Object Reference* for command documentation.

Bounds Tests

EViews 13 now performs Pesaran’s (2001) bounds tests for cointegration that are robust to whether variables of interest are $I(0)$, $I(1)$, or mutually cointegrated. These tests are formulated as standard F -test or Wald tests of parameter significance in the cointegrating relationship of the Conditional Error Correction model for each cross-section.

To perform the bounds tests, click on **View/ARDL Diagnostics/Bounds Tests**. The results of the test performed are presented in the first table of a spool. Below the table of long run coefficient estimates are two additional tables, respectively titled as the F -Bounds Test and the t -Bounds Test.

Equation: EX3 Workfile: ARDL_EV13::Untitled\

View Proc Object Print Name Freeze Estimate Forecast Stats Resids

Bounds Test
Bounds Critical Values

Bounds Test

Null hypothesis: No levels relationship
Number of cointegrating variables: 1
Trend type: Rest. constant (Case 2)
Sample size: 203

Test Statistic	Value
F-statistic	17.247539

Bounds Critical Values

Sample Size	10%		5%		I
	I(0)	I(1)	I(0)	I(1)	
Asymptotic	3.020	3.510	3.620	4.160	4

* I(0) and I(1) are respectively the stationary and non-stationary bound

- See “Bounds Test View” on page 334 in *User’s Guide II*.
- See also `Equation::boundstest` (p. 68) in the *Object Reference* for command documentation.

Testing for Asymmetry

The NARDL specification is quite general and can accommodate asymmetries in different combinations of short and long-run dynamics. From an estimated NARDL, you may test long-run symmetry restrictions and/or short-run symmetry restrictions.

To perform the symmetry tests on the estimated NARDL, click on **View/ARDL Diagnostics/Symmetry Test** from an estimated equation:

Coefficient symmetry tests
 Null hypothesis: Coefficient is symmetric
 Degrees of freedom (simple tests): F(1,194), Chi-square(1)
 Degrees of freedom (joint tests): F(2,194), Chi-square(2)
 Equation: EX4

Variable	Statistic	Value	Probability
Long-run			
LOG(REALGOVT)	F-statistic	0.525997	0.4692
	Chi-square	0.525997	0.4683
Short-run			
LOG(REALGOVT)	F-statistic	0.077094	0.7816
	Chi-square	0.077094	0.7813
Joint (Long-Run and Short-Run)			
LOG(REALGOVT)	F-statistic	0.493551	0.6112
	Chi-square	0.987102	0.6105

- See [“Symmetry Test View” on page 337](#) in *User’s Guide II*.
- See also [Equation::symmtest \(p. 213\)](#) in the *Object Reference* for command documentation.

Diagnostics in Panel ARDL/PMG

EViews 13 features improved PMG diagnostics, including display of individual cross-section error correction equations coefficients and cointegrating series, cross-section specific bounds, Hausman tests for PMG coefficient similarity, and tests of symmetry restrictions:

- New views make it easier to use the PMG estimates to examine the short and long-run representations of the distributed lag regressors and cointegrating relationships. ([“Cross-sectional Cointegration Views” on page 87.](#))
- Cross-section bounds tests for cointegration are now available as a view of your ARDL equation ([“Cross-sectional Bounds Tests” on page 88.](#))
- You may evaluate symmetry restrictions on the coefficients of asymmetric long and short-run regressors ([“Testing for Symmetry” on page 89.](#))
- You may perform a Hausman-test of the similarity of the PMG estimator to the mean-group (MG) and dynamic fixed effects (DFE) estimators ([“Similarity Tests” on page 90.](#))

For general discussion, see [Chapter 29. “Linear and Nonlinear ARDL,”](#) on page 321 in *User’s Guide II* which discusses a variety of these features in non-panel settings

Cross-sectional Cointegration Views

The Error Correction Results view displays coefficient results for the error correction regressions for each cross-section. **Select View/ARDL Diagnostics/Cross-sectional Error-correction Results** to display a spool of tables with results for each cross-section:

Variable	Coefficient	Std. Error	t-Statistic	Prob.
COINTEQ	-0.100656	0.034958	-2.879321	0.0076
DLOG(INF)	0.055291	0.071792	0.742306	0.4641
DLOG(INC)	0.202197	0.060277	3.354458	0.0023
C	0.076552	0.021553	3.551845	0.0014

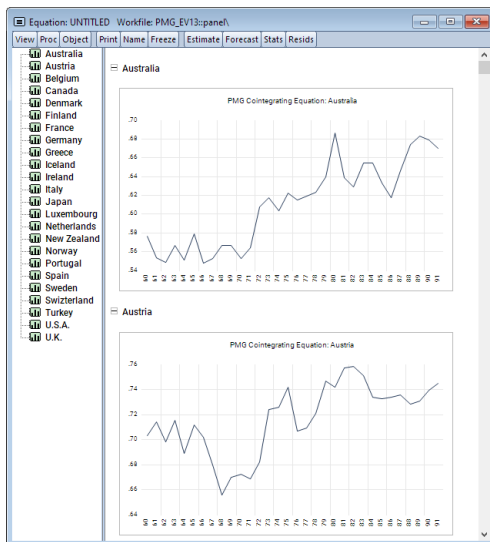
Variable	Coefficient	Std. Error	t-Statistic	Prob.
COINTEQ	-0.332225	0.089126	-3.727590	0.0009
DLOG(INF)	-0.293472	0.148905	-1.970866	0.0587
DLOG(INC)	0.093059	0.147533	0.630767	0.5333
C	0.254048	0.064862	3.916766	0.0005

Variable	Coefficient	Std. Error	t-Statistic	Prob.
COINTEQ	-0.172961	0.050977	-3.392897	0.0021
DLOG(INF)	0.050710	0.094686	0.533564	0.5966
DLOG(INC)	0.219710	0.105902	2.074655	0.0477
C	0.171224	0.046753	3.662275	0.0011

Variable	Coefficient	Std. Error	t-Statistic	Prob.
COINTEQ	-0.164322	0.056333	-2.916982	0.0069
DLOG(INF)	-0.075847	0.116010	-0.653795	0.5186
DLOG(INC)	0.381979	0.084555	4.517515	0.0001
C	0.099163	0.030920	3.207037	0.0033

- See [Equation::ecresults \(p. 109\)](#) in the *Object Reference* for command documentation.

The Cointegrating Relations Plots view displays information about the error correction term EC_t representing the cointegrating relation. **Select View/ARDL Diagnostics/Cross-sectional Cointegrating Relations Plots** to display a spool of plots for each cross-section:



- See [Equation::cointrel](#) (p. 95) in the *Object Reference* for command documentation.

Cross-sectional Bounds Tests

EViews 13 now performs Pesaran's (2001) bounds tests for cointegration that are robust to whether variables of interest are $I(0)$, $I(1)$, or mutually cointegrated. These tests are formulated as standard F -test or Wald tests of parameter significance in the cointegrating relationship of the Conditional Error Correction model for each cross-section.

To perform the bounds tests, click on **View/ARDL Diagnostics/Cross-sectional Bounds Tests**. The results of the test performed on each cross-section are presented in the first table of a spool. Below the table of long run coefficient estimates are two additional tables, respectively titled as the F -Bounds Test and the t -Bounds Test.

Equation: UNTITLED Workfile: PMG_EV13.pane1

View Proc Object Print Name Freeze Estimate Forecast Stats Resids

Bounds Test

Bounds Critical Values

Bounds Test

Null hypothesis: No levels relationship
 Number of cointegrating variables: 24
 Number of cointegrating variables: 2
 Trend type: Unrest. constant (Case 3)

Cross-Section	Obs.	F-Stat.	t-Stat.
Australia	32	3.676847	-2.976078
Austria	32	4.688066	-2.797571
Belgium	31	6.985687	-4.362671
Canada	32	7.906933	-4.593559
Denmark	32	1.054925	-1.257217
Finland	32	14.97449	-6.435964
France	32	4.689448	-1.512091
Germany	32	2.583601	-1.391053
Greece	32	3.437008	-3.149470
Iceland	32	2.729350	-2.597729
Ireland	32	4.793260	-3.659596
Italy	32	4.777133	-3.733318
Japan	32	16.82989	-3.954754
Luxembourg	32	3.130464	-2.100891
Netherlands	32	7.077504	-4.213263
New Zealand	32	9.213427	-4.718205
Norway	32	8.775988	-4.999296
Portugal	32	10.17183	-3.884162
Spain	32	4.395936	-3.467710
Sweden	32	3.605149	-3.178361
Switzerland	32	3.446830	-1.644373
Turkey	32	3.388064	-3.176190
U.S.A.	32	3.800081	-2.707570
U.K.	32	2.698197	-2.493495

Bounds Critical Values

Sample Size	10%		5%		1%	
	I(0)	I(1)	I(0)	I(1)	I(0)	I(1)
F-Statistic						
30	3.437	4.470	4.267	5.473	6.183	7.873
35	3.393	4.410	4.183	5.333	6.160	7.607
Asymptotic	3.170	4.140	3.790	4.850	5.150	6.360
t-Statistic						
Asymptotic	-2.570	-3.210	-2.860	-3.530	-3.430	-4.100

* I(0) and I(1) are respectively the stationary and non-stationary bounds.

- See [Equation::boundstest](#) (p. 68) in the *Object Reference* for command documentation.

Testing for Symmetry

One can test for symmetry formally by performing the usual t -test or F -test of parameter equality. For example, testing for symmetry for a specific long-run (LR) variable, say x_j , is equivalent to the following hypothesis:

$$H_0: \lambda_j^+ = \lambda_j^-$$

$$H_1: \lambda_j^+ \neq \lambda_j^- \quad (0.1)$$

To perform the symmetry test, select **View/ARDL Diagnostics/Symmetry Test** from the menu of a nonlinear asymmetric NARDL/PMG equation:

- See [Equation::symmtest](#) (p. 213) in the *Object Reference* for command documentation.

Similarity Tests

We may easily perform a Hausman test on the similarity of the PMG estimator to the mean-group (MG) and dynamic fixed effects (DFE) estimators. To conduct the test, click on **ARDL Diagnostics/PMG Similarity Test**:

Equation: UNTITLED Workfile: PMG_EV13:panel

View Proc Object Print Name Freeze Estimate Forecast Stats Resids

Hausman Test

Differences: Mean Group

Differences: Dynamic Fixed Effects

Estimation: Mean Group

Estimation: Dynamic Fixed Effects

Hausman Test

PMG Hausman Specification Test
Null hypothesis: Estimator is statistically similar to the PMG estimator

Estimator	Stat.	DOF	p-Value
Mean Group	1.596241	2	0.4502
Dynamic Fixed Effects	30.157643	2	0.0000

Differences: Mean Group

Coefficient Difference Overview: Mean Group

Variable	MG	PMG	Var(Diff.)	p-Value
LOG(INF)	-0.352900	-0.465846	0.010425	0.2686
LOG(INC)	0.918134	0.904433	0.000668	0.5961

Differences: Dynamic Fixed Effects

Coefficient Difference Overview: Dynamic Fixed Effects

Variable	DFE	PMG	Var(Diff.)	p-Value
LOG(INF)	-0.266343	-0.465846	0.001437	0.0000
LOG(INC)	0.912057	0.904433	0.000174	0.5631

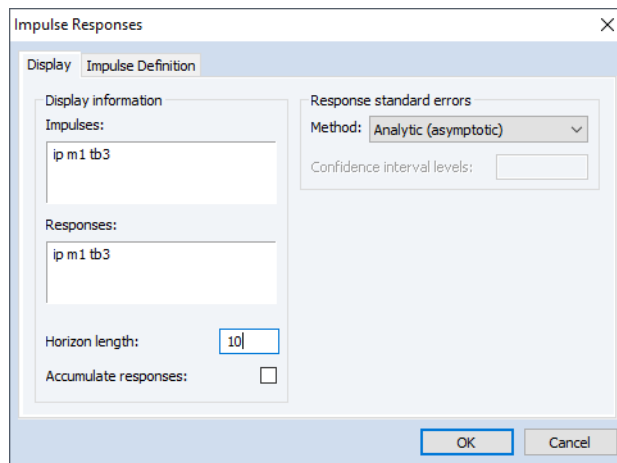
- See [Equation::similarity](#) (p. 207) in the *Object Reference* for command documentation.

Enhanced Impulse Response Display

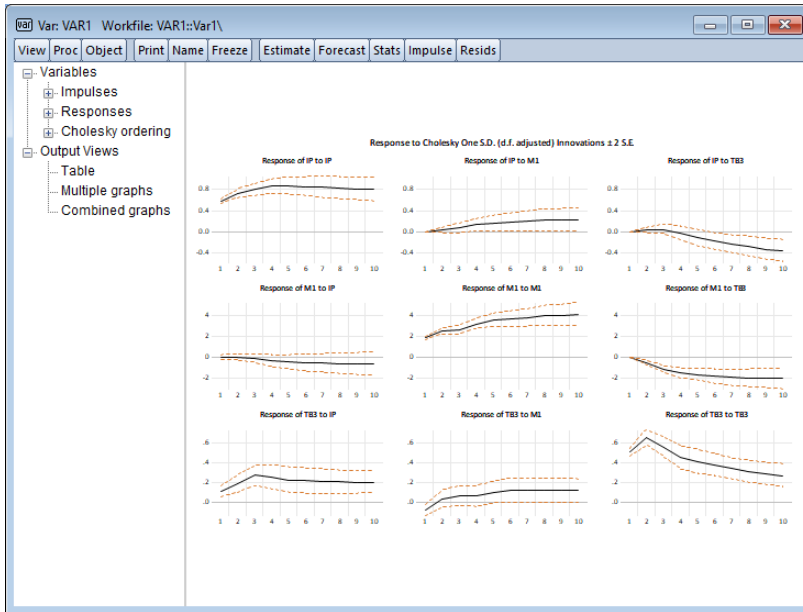
EViews 13 offers considerably more control over the computation and display of impulse response (and variance decomposition) confidence intervals. You may now display multiple confidence intervals with user-specified sizes in a single graph, and employ shading to improve the visibility of these intervals.

Recall that previous versions of EViews imposed important restrictions on the display of impulse response confidence intervals.

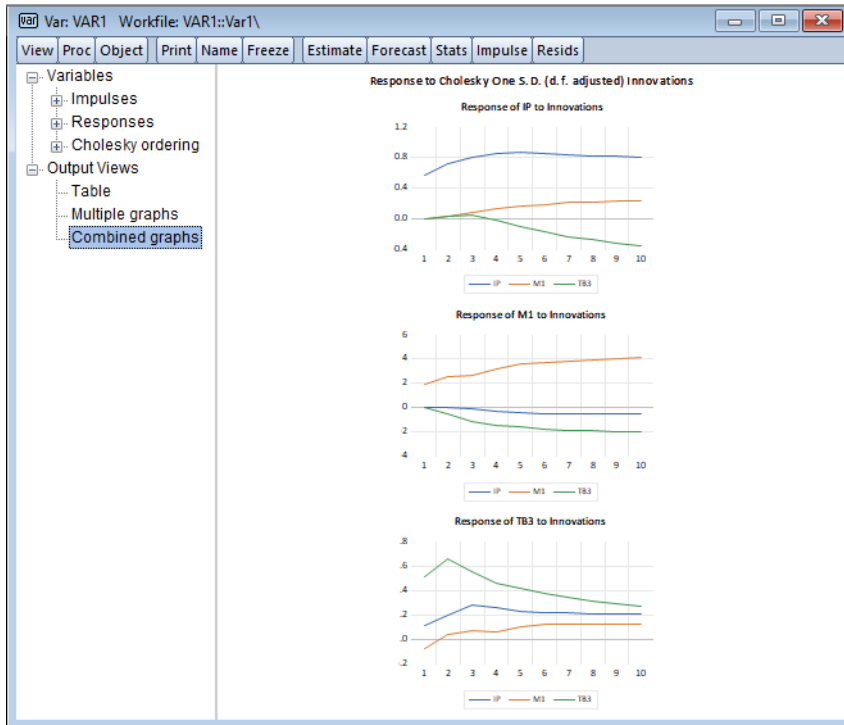
- First, for impulse-response confidence intervals computed using analytic or Monte Carlo standard errors, prior versions of EViews only allowed for the computation of ± 2 standard error bands. There were no options for displaying different sized intervals, let alone for displaying different intervals in the same graph:



- Second prior EViews only allowed you to use dotted lines to display the confidence bands around the impulse-responses:

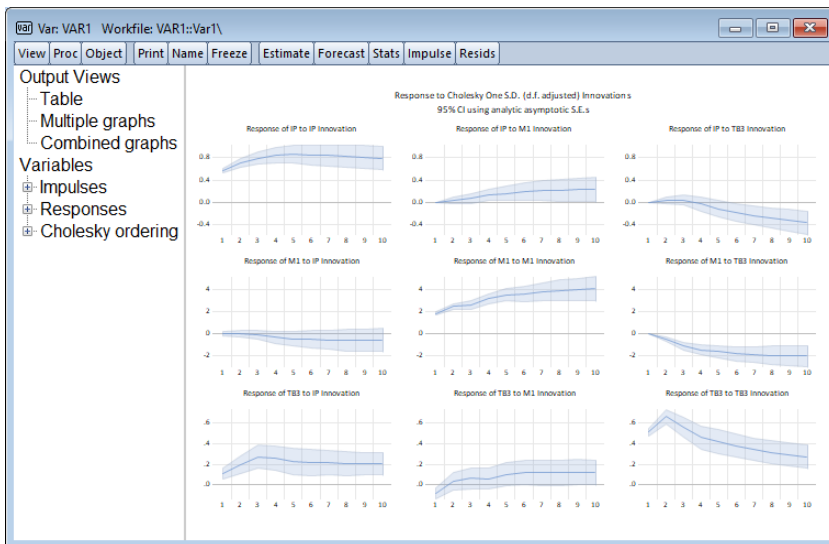


- Lastly, if you chose to display multiple impulses or responses in a combined graphs, confidence intervals were not available:



Fortunately, EViews 13 removes all three of these limitations.

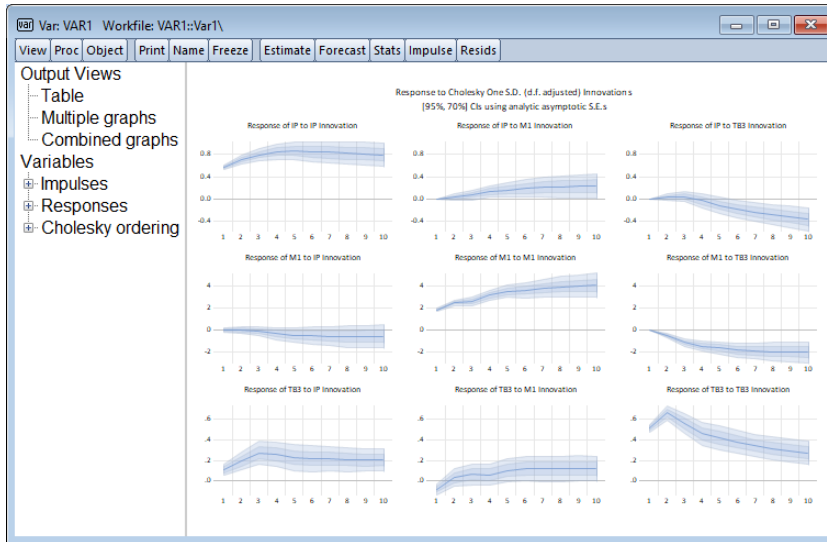
- First, by default, EViews 13 now uses shaded areas to depict confidence bands:



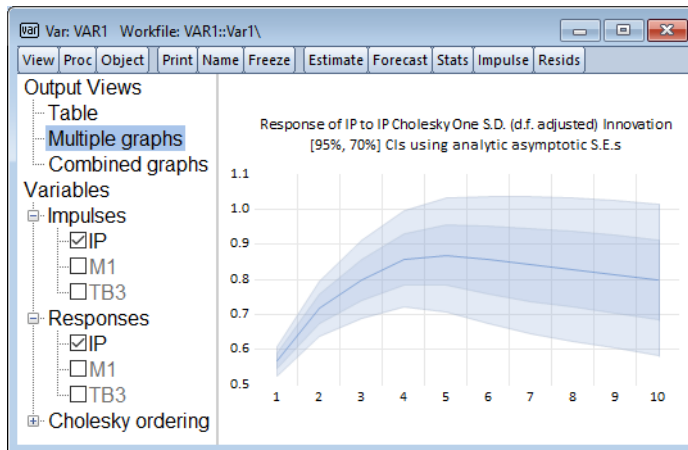
To display your graphs using the traditional lines, check the box next to **Display intervals using lines** in the impulse response dialog. If coming from the command line, you may use the `uselines` keyword in the impulse command options to display in the previous format.

- Second, EViews 13 now lets users display analytic and monte carlo confidence intervals with one or more user-specified sizes:

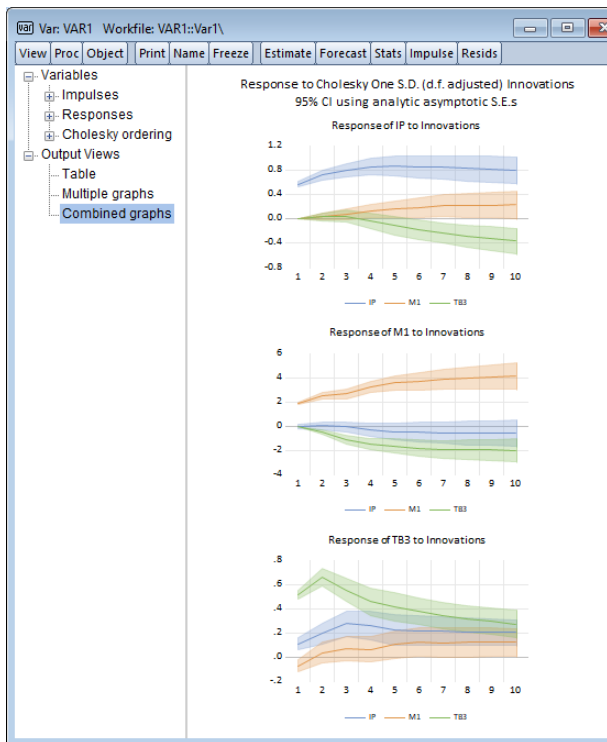
When multiple shaded bands are displayed, EViews automatically uses the new EViews 13 line and shade transparency engine (“[Graph Line and Shade Transparency](#)” on page 45) to display the bands using gradients:



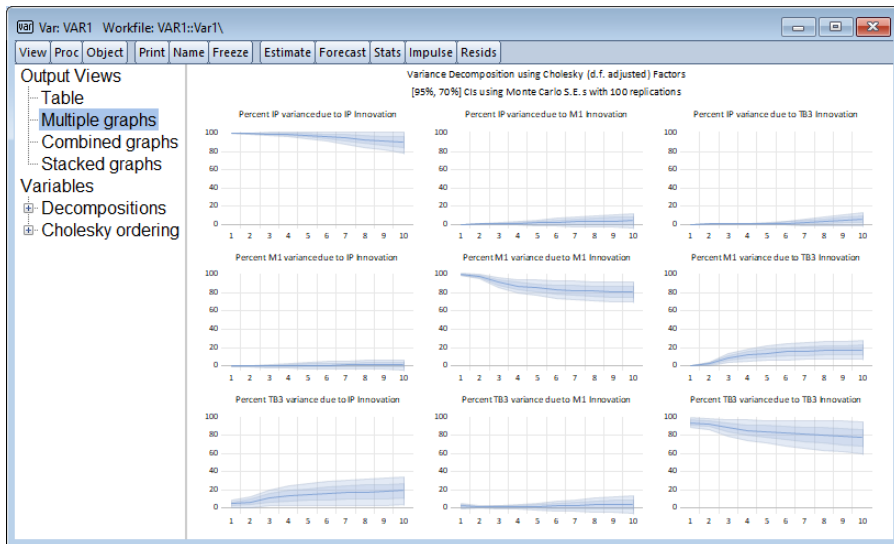
Using the selection interface to focus on the topmost left graph, better shows the shading:



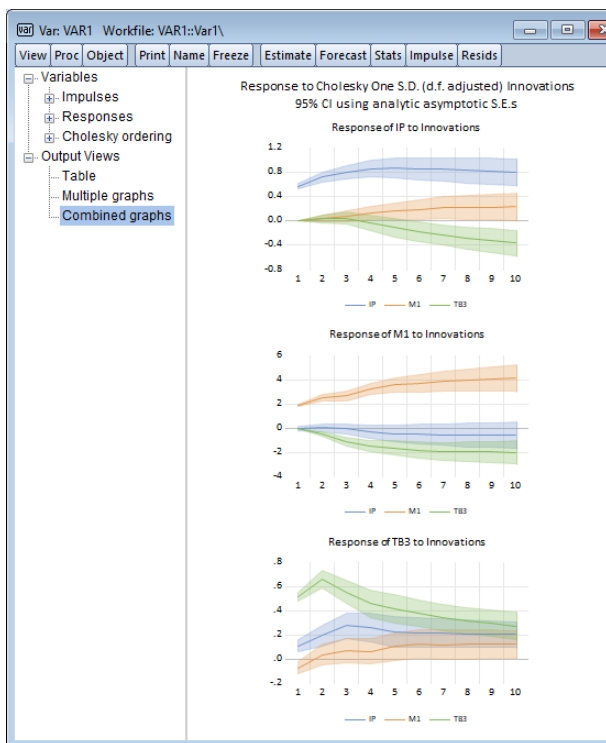
- Further, EViews 13 now allows you to display confidence intervals in the impulse-response combined graphs view:



All of the new shading and multiple band support is also provided for variance decomposition in multiple graph



and combined graph settings,



- See [Var::impulse](#) (p. 1078) and [Var::vdecomp](#) (p. 1111) in the *Object Reference*, for updated command documentation.

Forecast Sample Setting Commands

EViews 13 offers a small but surprisingly useful improvement in forecast sample setting.

Previously, forecasting via command generally required the user to first set the global `smpl` using a `smpl` statement prior to issuing the forecasting command.

For example, the following lines estimated the equation using one sample, and then forecast over two different samples,

```
smpl 1990q1 2010q4
equation eq1.ls y c x1 x2 x3
smpl 2011q1 2020q4
eq1.forecast fcst1
smpl 2008q3 2020q4
eq2.forecast fcst2
```

In this case, setting the global sample prior to equation forecasting was only mildly inconvenient. However, in some settings, particularly those involving non-equation forecasting where you are generating multiple forecasts and data, setting and resetting the global sample was inconvenient at best, and prone to error.

EViews 13 solves this problem by updating most forecast-generating commands to take a forecast sample information.

In this context, there are two types of forecasting procedures.

In the first set of procedures, the forecast sample is set independently of the remainder of the procedure. For these cases, EViews 13 offers an option to allow you to set the sample range pair directly using the option:

<code>forcsmpl =</code>	Fit sample (optional). If forecast sample is not provided, the workfile sample will be employed.
<code>smpl</code>	

This option is available interactively and in commands for:

- `Equation::fit` (p. 115) in the *Object Reference*.
- `Series::forcavg` (p. 720) in the *Object Reference*.
- `Equation::forecast` (p. 119) in the *Object Reference*.

The second type of forecasting procedure sets the forecast sample one-period beyond a previously specified estimation sample, and then forecasting continues from that period onward until a forecast endpoint.

In this case, EViews 13 allows you to set the forecast sample in two distinct ways: by specifying the number of periods to forecast, or by specifying the forecast end date:

<code>forclen = int</code>	Number of periods to forecast.
<code>forc = "date"</code>	Specify the date of the forecast end point.

These options are available interactively and in commands for:

- `Series::autoarma` (p. 694) in the *Object Reference*.
- `Series::ets` (p. 716) in the *Object Reference*.

Our earlier example changes from the commands

```
smp1 1990q1 2010q4
equation eq1.ls y c x1 x2 x3
smp1 2011q1 2020q4
eq1.forecast fcst1
smp1 2008q3 2020q4
eq2.forecast fcst2
```

to

```
smp1 1990q1 2010q4
equation eq1.ls y c x1 x2 x3
eq1.forecast(forcsmp1="2011q1 2020q4") fcst1
eq1.forecast(forcsmp1="2008q3 2020q4") fcst2
```

Command Language

Updated Command List

(Unless otherwise specified, all of the object views and procedures are in *Command and Programming Reference*.)

Commands

- dbopen** open a database (p. 371). (*updated*)
- deleteaddin** unregister a program file as an EViews Add-in (p. 380). (*new*)
- pageload** load one or more pages into a workfile from a workfile or a foreign data source (p. 481). (*updated*)
- pagesave** save page into a workfile or a foreign data source (p. 483). (*updated*)
- stom** Converts series, alpha, or group to vector, svector, or matrix after removing observations with missing values (p. 843). (*updated*)
- stomna** Converts series, alpha, or group to vector, svector, or matrix without removing observations with missing values (p. 844). (*updated*)
- wfdir** change the workfile view to a simple object directory listing (p. 566). (*new*)
- wffilter** change the workfile object filter for the current workfile window (p. 566). (*new*)
- wfopen** open workfile or foreign source data as a workfile (p. 567). (*updated*)
- wfsave** save workfile to disk as a workfile or a foreign data source (p. 583). (*updated*)
- xopen** open a connection to an external application (p. 599). (*updated*)

Updated Object List

(Unless otherwise specified, all of the object views and procedures are in *Object Reference*.)

Coef

Coef Procs

- export**save coef as Excel 2007 XLSX, CSV, tab-delimited ASCII text, RTF, HTML, Enhanced Metafile, PDF, TEX, or MD file on disk (p. 25).
(*new*)
- import**.....imports data from a foreign file into the coef object (p. 29). (*new*)

Coef Values

- @dropprow**(*arg*)Returns the coef with the rows defined by *arg* removed. *arg* may be an integer, vector of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*new*)
- @row**(*arg*)Returns the rows defined by *arg*. *arg* may be an integer, vector of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*new*)

Equations

Equation Methods

- ardl**.....least squares with autoregressive distributed lags (p. 61). (*updated*)
- did**.....estimate a panel equation using the difference-in-difference estimator (p. 103). (*new*)
- ls**equation using least squares or nonlinear least squares (p. 156). (*updated*)

Equation Views

- boundstest**perform the Pesaran, Shin and Smith (2001) bounds test of long-run relationships from an ARDL estimated equation (p. 68). (*updated*)
- cointrel**display information about the cointegrating relation specification and the coefficients in ARDL estimated equation (p. 95). (*new*)
- didcs**.....compute Callaway-Sant'Anna decomposition for difference-in-difference estimation (p. 104). (*new*)
- didgbdecomp**.....perform Goodman-Bacon decomposition for difference-in-difference estimation (p. 105). (*new*)
- didtrends**.....show difference-in-difference trends summary in graphical or tabular form (p. 106). (*new*)
- dynmult**.....compute dynamic multipliers for long-run regressors in ARDL equations (p. 108). (*new*)
- ecresults**.....display the conditional error correction (CEC) and error correction (EC) regression results (p. 109). (*new*)
- similarity**compute PMG Hausman test for similarity (p. 207). (*new*)

symmtest..... compute symmetry test for nonlinear distributed lag variables in nonlinear ARDL models (p. 213). (*new*)

Equation Procs

didmakeeq create an equation object with the underlying fixed-effects estimation of a difference-in-difference equation (p. 105). (*new*)

fit..... static forecast (p. 115). (*updated*)

forecast dynamic forecast (p. 119). (*updated*)

Equation Values

@varselkept space delimited list of variables kept by model selection. (*new*)

@varselrejected space delimited list of the variables dropped by model selection. (*new*)

Geomap

Geomap Procs

setfillcolor define the fill (background) color used in geomap shapes using values in a series (p. 311). (*updated*)

setjust set the display justification for multi-line area labels (p. 318). (*new*)

setshapelabel set which attribute to use or create a custom label to use when labeling shapes (p. 319). (*new*)

Geomap Values

@ids("attr", "val")space delimited string containing the ID numbers of all the areas which has the matching attribute value for the specified attribute name. (*new*)

Graph

Graph Procs

datalabel controls labeling of the observations/data in time plots (p. 344). (*new*)

datelabel controls labeling of the bottom date/time axis in time plots (p. 346). (*updated*)

setelem..... set individual line, symbol, bar and legend options for each series in the graph (p. 373). (*updated*)

Group

Group Views

coint test for cointegration between series in a group (p. 397). (*updated*)

Group Procs

setfillcolor set custom spreadsheet fill coloring for the group (p. 447). (*updated*)

settextcolorset custom spreadsheet text coloring for the group (p. 457).
(*updated*)

Matrix

Matrix Procs

clearcollabelsclear the column labels in a matrix object (p. 508). (*new*)

clearrowlabelsclear the row labels in a matrix object (p. 509). (*new*)

exportsave matrix as Excel 2007 XLSX, CSV, tab-delimited ASCII text, RTF, HTML, Enhanced Metafile, PDF, TEX, or MD file on disk (p. 517).
(*new*)

import.....imports data from a foreign file into the matrix object (p. 521).
(*updated*)

resize.....resize the matrix object (p. 537) (*new*).

setcollabelsset the column labels in a matrix object (p. 538). (*updated*)

setrowlabelsset the row labels in a matrix object (p. 541). (*updated*)

Matrix Values

@col(arg)Returns the columns defined by *arg*. *arg* may be an integer, vector of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined column labels. (*updated*)

@dropcol(arg)Returns the matrix with the columns defined by *arg* removed. *arg* may be an integer, vector of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined column labels. (*new*)

@droprow(arg)Returns the matrix with rows defined by *arg* removed. The *arg* may be integer, vectors of integers, strings, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*new*)

@dropboth(arg1, arg2) ... Returns the matrix with the rows defined by *arg1* and columns defined by *arg2* removed. The *args* may be integers, vectors of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*new*)

@row(arg)Returns the rows defined by *arg*. *arg* may be an integer, vector of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*updated*)

@sub(*arg1*, *arg2*).. Returns the matrix with rows defined by *arg1* and columns with defined by *arg2*. The *args* may be integers, vectors of integers, strings, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*updated*)

Rowvector

Rowvector Procs

clearcollabels..... clear the column labels in a rowvector object (p. 645). (*new*)
clearrowlabels clear the row labels in a rowvector object (p. 647). (*new*)
export export vector as Excel 2007 XLSX, CSV, tab-delimited ASCII text, RTF, HTML, Enhanced Metafile, PDF, TEX, or MD file on disk (p. 648). (*new*)
import..... imports data from a foreign file into the vector object (p. 653). (*updated*)
resize resize the rowvector object (p. 662). (*new*)
setcollabels..... set the column labels in a rowvector object (p. 663). (*new*)
setrowlabels set the row labels in a rowvector object (p. 666). (*new*)

Rowvector Values

@col(*arg*)..... Returns the columns defined by *arg*. *arg* may be an integer, vector of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined column labels. (*new*)
@dropcol(*arg*) Returns the matrix with the columns defined by *arg* removed. *arg* may be an integer, vector of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined column labels. (*new*)
@t Returns transpose. (*new*)

Series

Series Procs

autoarma..... forecast from a series using an ARIMA model with automatic determination of the specification (p. 694). (*updated*)
dsa..... seasonally adjust daily data using the DSA method (p. 710). (*new*)
ets..... perform Error-Trend-Season (ETS) estimation and exponential smoothing (p. 716). (*updated*)
forcavg..... average forecasts of a series (p. 720). (*updated*)
setfillcolor define the fill (background) color used in series spreadsheets (p. 757). (*updated*)

- settextcolor**set custom spreadsheet text coloring for the series (p. 766).
(*updated*)
- smooth**.....exponential smoothing (p. 773). (*updated*)

Svector

Svector Procs

- clearcollabels**clear the column labels in a svector object (p. 889). (*new*)
- clearrowlabels**clear the row labels in a svector object (p. 890). (*new*)
- fill**.....fill the svector with the specified values (p. 892). (*new*)
- resize**.....resize the svector object (p. 894). (*new*)
- setcollabels**set the column labels in a svector object (p. 895). (*new*)
- setrowlabels**set the row labels in a svector object (p. 895). (*new*)

Svector Values

- @droprw**(*arg*)Returns the svector with the rows defined by *arg* removed. *arg* may be an integer, vector of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*new*)
- @row**(*arg*)Returns the rows defined by *arg*. *arg* may be an integer, vector of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*new*)

Sym

Sym Procs

- clearcollabels**clear the column labels in a vector object (p. 900). (*new*)
- clearrowlabels**clear the row labels in a vector object (p. 902). (*new*)
- export**save sym matrix as Excel 2007 XLSX, CSV, tab-delimited ASCII text, RTF, HTML, Enhanced Metafile, PDF, TEX, or MD file on disk (p. 912). (*new*)
- import**.....imports data from a foreign file into the sym object (p. 916).
(*updated*)
- resize**.....resize the sym object (p. 926). (*new*)
- setcollabels**set the column labels in a sym object (p. 927). (*new*)
- setrowlabels**set the row labels in a sym object (p. 930). (*new*)

Sym Values

- @col**(*arg*)Returns the columns defined by *arg*. *arg* may be an integer, vector of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined column labels. (*new*)

- @dropcol(arg)** Returns the matrix with the columns defined by *arg* removed. *arg* may be an integer, vector of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined column labels. (*new*)
- @drowrow(arg)** Returns the matrix with rows defined by *arg1* and columns with rows defined by *arg2* removed. The *args* may be integer, vectors of integers, strings, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*new*)
- @dropboth(arg)** ... Returns the sym with the rows and columns defined by *arg* removed. *arg* may be an integer, vector of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*new*)
- @dropboth(arg1, arg2)** ...Returns the matrix with the rows defined by *arg1* and columns defined by *arg2* removed. The *args* may be integers, vectors of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*new*)
- @row(arg)** Returns the rows defined by *arg*. *arg* may be an integer, vector of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*new*)
- @sub(arg)** Returns the sym with rows defined by *arg1* and columns with rows defined by *arg2*. The *args* may be integers, vectors of integers, strings, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*new*)
- @sub(arg1, arg2)**.. Returns the matrix with rows defined by *arg1* and columns with defined by *arg2*. The *args* may be integers, vectors of integers, strings, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*updated*)
- @t** Returns transpose. (*new*)

Table

Table Procs

- deletecells** delete cells from a table (p. 983). (*updated*)
- deletecol**..... remove columns from a table (p. 984). (*updated*)
- deleterow** remove rows from a table (p. 984). (*updated*)
- fixcol**..... fixes a set of columns to left of the spreadsheet view so that the leading columns are always in view (p. 986). (*new*)

- fixrow**fixes a set of rows at the top of the spreadsheet view so that the leading rows are always in view (p. 987). *(new)*
- fixrowcol**fixes a set of rows at the top and a set of columns to left of a spreadsheet view so that the leading rows and columns are always in view (p. 987). *(new)*
- insertcells**insert cells into a table (p. 988). *(updated)*
- insertcol**insert additional columns into a table (p. 989). *(updated)*
- insertrow**insert additional rows into a table (p. 989). *(updated)*
- save**save table as Excel 2007 XLSX, CSV, tab-delimited ASCII text, RTF, HTML, Enhanced Metafile, PDF, TEX, or MD file on disk (p. 992). *(updated)*
- setfillcolor**set the fill (background) color of a set of table cells (p. 996). *(updated)*
- setfont**set the font for the text in a set of table cells (p. 998). *(updated)*
- setformat**set the display format of a set of table cells (p. 999). *(updated)*
- setheight**set the row height in a set of table cells (p. 1003). *(updated)*
- setjust**set the justification for a set of table cells (p. 1005). *(updated)*
- setlines**set the line characteristics and borders for a set of table cells (p. 1006). *(updated)*
- settextcolor**set the text color in a set of table cells (p. 1010). *(updated)*
- setwidth**set the column width for a set of table cells (p. 1011). *(updated)*

String values

- @find("arg")**string containing the cell identifiers meeting the boolean argument. This argument can be a mathematical or string expression. For example:

```
string s = tab1.@find("[b1:c15]>0.3")
```

returns a list of cells between B1 and C15 greater than 0.3;

```
string s = tab11.@find("[a1:e67]== \"1949q4\"")
```

returns a list of cells between A1 and E67 matching the string '1949q4' (string comparisons ignore case);

```
string s = tab1.@find("[@all]==0.5")
```

returns a list of cells in the table equal to 0.5;

```
string s = t.@find("@instr([@all],\"in\")")
```

returns a list of cells in the table containing the substring 'in';

```
string s = t.@find("[b3:c5]<[d5]")
```

returns a list of cells between B3 and C5 less than cell D5;

```
string s = t.@find("@left([@all],1)=\"cp\"")
```

VAR

Var Methods

- btvcvar** estimate a Bayesian time-varying coefficients VAR specification (p. 1051). (*new*)
- ec** estimate a vector error correction model (p. 1065). (*updated*)

Var View

- coint** Johansen cointegration test (p. 1058). (*updated*)
- impulse** impulse response functions (p. 1078). (*updated*)
- vdecomp** variance decomposition (p. 1111). (*updated*)

Var Procs

- fit** produce static forecasts from an estimated VAR (p. 1071). (*updated*)
- forecast** produce dynamic forecasts from an estimated VAR or VEC (p. 1073). (*updated*)

Var Values

- @cointadj** matrix containing cointegrating variable adjustment coefficients α (where applicable).
- @cointadjse** matrix containing standard errors of cointegrating variable adjustment coefficients (where applicable).
- @cointlr** matrix containing long-run equilibrium coefficients $\Pi = \alpha\beta'$ (where applicable)
- @cointlirse** matrix containing standard errors of long-run equilibrium coefficients (where applicable)
- @cointsr** matrix containing short-run adjustment coefficients Γ' (where applicable).
- @cointsrse** matrix containing standard errors of short-run adjustment coefficients (where applicable).

Vector

Vector Procs

- clearcollabels** clear the column labels in a vector object (p. 1118). (*new*)
- clearrowlabels** clear the row labels in a vector object (p. 1118). (*new*)
- export** export vector as Excel 2007 XLSX, CSV, tab-delimited ASCII text, RTF, HTML, Enhanced Metafile, PDF, TEX, or MD file on disk (p. 1123). (*new*)
- import** imports data from a foreign file into the vector object (p. 1127). (*updated*)
- resize** resize the vector object (p. 1137). (*new*)
- setcollabels** set the column label for the vector object (p. 1137). (*updated*)

[setrowlabels](#)set the row labels for the vector object (p. 1141). (*updated*)

Vector Values

[@dropprow](#)(*arg*)Returns the vector with the rows defined by *arg* removed. *arg* may be an integer, vector of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*new*)

[@row](#)(*arg*)Returns the rows defined by *arg*. *arg* may be an integer, vector of integers, string, or svector of strings. Integer values correspond to rows and string values correspond to previously defined row labels. (*updated*)

[@t](#)Returns transpose. (*new*)

Updated Function List

String Functions

[@sfill](#)Returns a svector containing the elements specified by the arguments to the function. (p. 724). (*new*)

[@str](#)returns a string representing the given number (p. 724). (*updated*)

[@val](#)returns the number that a string represents (p. 740). (*updated*)

[@wreplace](#)replaces parts of a string based on patterns (p. 756). (*updated*)

Matrix Functions

[@grid](#)Vector containing an equally spaced grid of elements (p. 808). (*new*)

[@hcat](#)Horizontally concatenate matrix objects (p. 809). (*updated*)

[@range](#)Returns a vector holding the sequential integers starting at *l* and ending at *h* (p. 827). (*new*)

[@vech](#)Vectorize (stack columns of) lower triangle of matrix object (p. 859). (*updated*)

[@zeros](#)Matrix or vector of zeros (p. 860). (*new*)

Support Functions

[@makevalidname](#)string containing an uppercased valid EViews name based on the input (p. 879). (*updated*)

[@xtype](#)returns the string describing the type of the active external application (p. 907). (*new*)

Workfile Functions

[@event](#)event identifier for observation (see “Event Function (@event),” on page 668). (*updated*)

@holiday returns the proportion of an annual event covered by each observation (see “[Holiday Functions](#),” on page 669). (*updated*)

@holidayset return the proportion of an annual set of events covered by each observation (see “[Holiday Functions](#),” on page 669). (*updated*)

Cumulative Statistics Functions

See “[Cumulative Statistic Functions](#),” on page 618.

@cumdn cumulative process of negative (below threshold) changes. (*new*)

@cumdp cumulative process of positive (above threshold) changes. (*new*)

@cumdz cumulative process of zero (at threshold) changes. (*new*)

@dcumdp difference of cumulative process of positive (above threshold) changes. (*new*)

@dcumdn difference of cumulative process of negative (below threshold) changes. (*new*)

@dcumdz..... difference of cumulative process of zero (at threshold) changes. (*new*)

EViews 12 Compatibility Notes

The following discussion describes EViews 13 compatibility issues for users of earlier versions.

Workfile Compatibility

With few exceptions, EViews 13 workfiles are backward compatible with EViews 12. Note that the following objects are new or have been modified in Version 12:

- Estimation objects estimated with methods that employ new features (nonlinear ARDL and PMG, Difference-in-difference equations, Bayesian Time-varying Coefficient VAR, VECs with general deterministics inside the cointegrating equation).

If you have saved workfiles containing any of the above objects and open them in earlier versions, EViews will delete the incompatible object and notify you that one or more objects were not read. If you then save the workfile, you will lose the objects. We recommend that you make a copy of any workfiles that contain these objects if you would like to use these workfiles in earlier versions of EViews.

Census X-13 Compatibility

EViews 13 uses a newer version of the Census X-13 executable. The current shipping version will be at least as new as Version 1.1, Build 59. There may be differences in results from previous versions due to Census Bureau changes in the routines.

Note that the Census Bureau has tested this version of X-13ARIMA-SEATS on Windows 10 and indicates that it may work on previous Windows versions.

