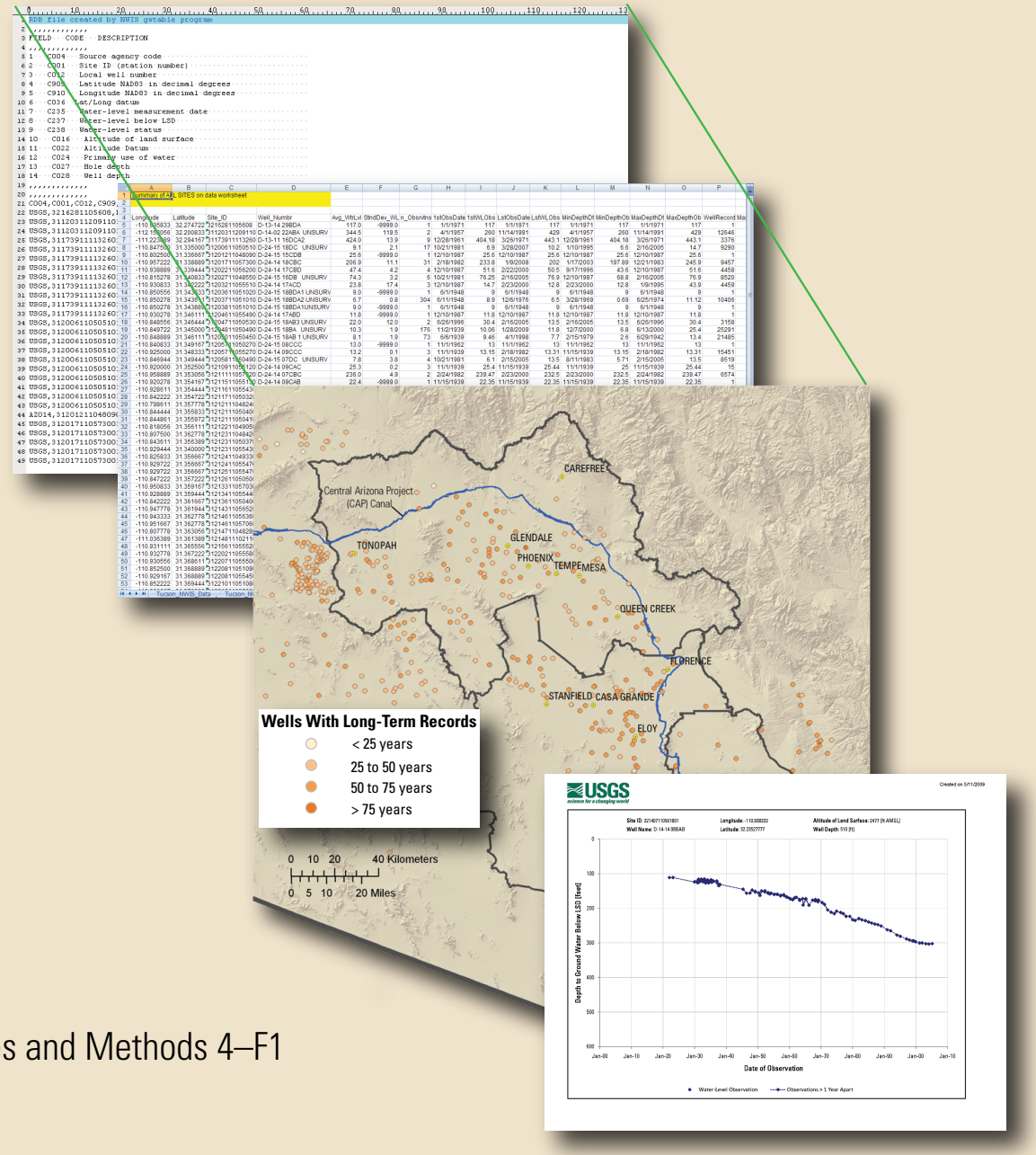
USGS
science for a changing world

National Water Availability and Use Pilot Program

Excel® Spreadsheet Tools for Analyzing Groundwater Level Records and Displaying Information in ArcMap™

Chapter 1 of
Section F, Groundwater, of **Book 4, Hydrologic Analysis and Interpretation**



Techniques and Methods 4–F1

U.S. Department of the Interior
U.S. Geological Survey

This page intentionally left blank

National Water Availability and Use Pilot Program

Excel® Spreadsheet Tools for Analyzing Groundwater Level Records and Displaying Information in ArcMap™

By Fred D Tillman

Chapter 1 of
Section F, Groundwater, of **Book 4, Hydrologic Analysis and Interpretation**

Techniques and Methods 4–F1

U.S. Department of the Interior
U.S. Geological Survey

U.S. Department of the Interior
KEN SALAZAR, Secretary

U.S. Geological Survey
Suzette M. Kimball, Acting Director

U.S. Geological Survey, Reston, Virginia: 2009

This report and any updates to it are available online at:
http://pubs.usgs.gov/tm/tm4f1/

For product and ordering information:
World Wide Web: http://www.usgs.gov/pubprod
Telephone: 1–888–ASK–USGS

For more information on the USGS—the Federal source for science about the Earth,
its natural and living resources, natural hazards, and the environment:
World Wide Web: http://www.usgs.gov
Telephone: 1–888–ASK–USGS

Suggested citation:

Tillman, Fred D, 2009, Excel® spreadsheet tools for analyzing groundwater level records and displaying information in ArcMap™: U.S. Geological Survey Techniques and Methods 4-F1, 59 p. [http://pubs.usgs.gov/tm/tm4f1/]

Cataloging-in-Publication data are on file with the Library of Congress (URL http://www.loc.gov/).

Produced in the Western Region, Menlo Park, California
Manuscript approved for publication, June 9, 2009
Text edited by Peter H. Stauffer
Layout and design by Stephen L. Scott

FRONT COVER
Illustration of the path from raw water-level data to useful groundwater information.

Contents

Figures

Tables

Conversion Factors and Datum

Multiply	By	To obtain
	Length	
foot (ft)	0.3048	meter (m)

Vertical coordinate information is referenced to the North American Vertical Datum of 1988 (NAVD 88).
Horizontal coordinate information is referenced to the North American Datum of 1927 (NAD 27).
Altitude, as used in this report, refers to distance above the vertical datum.

This page intentionally left blank

Excel® Spreadsheet Tools for Analyzing Groundwater Level Records and Displaying Information in ArcMap™

By Fred D Tillman

Abstract

When beginning hydrologic investigations, a first action is often to gather existing sources of well information, compile this information into a single dataset, and visualize this information in a geographic information system (GIS) environment. This report presents tools (macros) developed using Visual Basic for Applications (VBA) for Microsoft Excel® 2007 to assist in these tasks. One tool combines multiple datasets into a single worksheet and formats the resulting data for use by the other tools. A second tool produces summary information about the dataset, such as a list of unique site identification numbers, the number of water-level observations for each, and a table of the number of sites with a listed number of water-level observations. A third tool creates subsets of the original dataset based on user-specified options and produces a worksheet with water-level information for each well in the subset, including the average and standard deviation of water-level observations and maximum decline and rise in water levels between any two observations, among other information. This water-level information worksheet can be imported directly into ESRI® ArcMap™ as an "XY Data" file, and each of the fields of summary well information can be used for custom display. A separate set of VBA tools distributed in an additional Excel workbook creates hydrograph charts of each of the wells in the data subset produced by the aforementioned tools and produces portable document format (PDF) versions of the hydrograph charts. These PDF hydrographs can be hyperlinked to well locations in ArcMap™ or other GIS applications.

Introduction

The U.S. Geological Survey (USGS) Water Availability and Use Pilot Program Southwest Alluvial Basins project is developing new methods for analyzing and presenting groundwater conditions using existing datasets (Tillman and others, 2007; Tillman and others, 2008). An integral part of this project is to combine datasets from multiple sources, develop understanding of the scope and variability of the resulting dataset, produce subsets of the data based on criteria of interest, and present the data in a geographic information system (GIS) environment to aid in visualization of temporal water-level changes. Each of these steps involves the analysis of large amounts of water-level data, and Visual Basic for Applications (VBA) macros (programs) were written to aid in rapidly processing raw datasets into usable GIS information. Presenting well information in a GIS environment is useful in helping to understand the distribution of temporal water-level changes. The purpose of this report is to document the VBA macros distributed in the Well_Analyses.xlsm and Well_Hydrograph.xlsm workbooks. The report describes the data requirements and basic instructions for each tool. The tools described in this report are designed to be accessible by anyone with Excel® 2007, easy to use, and useful in producing GIS input files from raw data from the USGS National Water Information System (NWIS) and other databases. The capabilities of the tools are demonstrated using data from alluvial basins in Arizona.

Acknowledgments

Development of the tools described in this report was funded by the USGS National Water Availability and Use Pilot Program. The Arizona Department of Water Resources provided groundwater data for use in developing and testing these macros. Jeff Cordova of the USGS Arizona Water Science Center provided technical review at various stages in the development of this project.

Data Requirements

Macros created for analyzing water-level records and producing GIS files were written to maximize flexibility for

the user, but with certain requirements. All macros documented in this report require well data to be in a specifically formatted original data worksheet (fig. 1 and table 1). All data must begin on row 22 of the worksheet, and each row must contain a site identification number in the column specified. If multiple datasets are to be combined, the data from each dataset must be placed on the same worksheet, one directly after the other with no intervening blank rows. Some data are required by all VBA tools in this report, such as site identification number, longitude and latitude of the well location, and water-level observations with observation dates (table 1). Nonrequired data are used in the GIS attribute table or in the header of well hydrographs. If data are not available for any of the nonrequired fields, then the fields should be blank. Other data may be included on the worksheet after column P up to column CV (column 100) and will be copied automatically to all subsequent data subset worksheets. Column H may contain either depth to groundwater or water-level altitude data. Owing to limitations within Excel, no water-level observations are allowed prior to January 1, 1900. Allowable formats for water-level observation dates include the default output formats for NWIS-GWSI of yyyymmdd or yyyymm or any Excel-recognizable date format. To determine if a date is in an Excel-recognizable format, open the data in Excel, right-click on the date cell, select Format Cells and change the format to Number. If the date appears as an integer number, then it was previously in an Excel-recognizable date format. If the date does not change to an integer number, but instead appears in the same format as before the format change, then the date is in a text format not recognizable in Excel. To correct this, copy and paste all dates to a text editor, then copy them from the text editor and paste them back into Excel. All data must be sorted first by site identification number (smallest to largest), then by date of observation (earliest to latest). This and other formatting is performed in the combine_format_datasets tool described below. The VBA programs were written to run in Microsoft Excel 2007 and are thus distributed in the "*.xlsm" format.The expansion of Excel 2007 to allow as many as 1,048,576 rows of data permits the tools described in this report to be run on very large datasets.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	RDB file created by NWIS gwtable program														
2															
3	FIELD CODE DESCRIPTION														
4															
5	1 C004 Source agency code														
6	2 C001 Site ID (station number)														
7	3 C012 Local well number														
8	4 C909 Latitude NAD83 in decimal degrees														
9	5 C910 Longitude NAD83 in decimal degrees														
10	6 C235 Water-level measurement date														
11	7 C237 Water-level below LSD														
12	8 C238 Water-level status														
13	9 C016 Altitude of land surface														
14	10 C022 Altitude Datum														
15	11 C024 Primary use of water														
16	12 C027 Hole depth														
17	13 C028 Well depth														
18															
19															
20															
21	C004	C001	C012	C909	C910	C036	C235	C237	C238	C016	C022	C024	C027	C028	
22	USGS	3216281105608	D-13-14 29BDA	32.27479755	-110.9364808	NAD27	01/01/1971	117		2360	NGVD29	H	-9999	200	
23	USGS	311739111132601	D-13-11 16DCA2	32.2942416	-111.2245465	NAD27	12/28/1961	404.18		2290.09	NGVD29		-9999	-9999	
24	USGS	311739111132601	D-13-11 16DCA2	32.2942416	-111.2245465	NAD27	12/24/1962	409.3		2290.09	NGVD29		-9999	-9999	
25	USGS	311739111132601	D-13-11 16DCA2	32.2942416	-111.2245465	NAD27	12/20/1963	413.17		2290.09	NGVD29		-9999	-9999	
26	USGS	311739111132601	D-13-11 16DCA2	32.2942416	-111.2245465	NAD27	12/23/1964	417.33		2290.09	NGVD29		-9999	-9999	
27	USGS	311739111132601	D-13-11 16DCA2	32.2942416	-111.2245465	NAD27	12/23/1966	425.9		2290.09	NGVD29		-9999	-9999	
28	USGS	311739111132601	D-13-11 16DCA2	32.2942416	-111.2245465	NAD27	12/27/1967	427.98		2290.09	NGVD29		-9999	-9999	
29	USGS	311739111132601	D-13-11 16DCA2	32.2942416	-111.2245465	NAD27	12/28/1969	434.2		2290.09	NGVD29		-9999	-9999	
30	USGS	311739111132601	D-13-11 16DCA2	32.2942416	-111.2245465	NAD27	01/15/1970	440.8		2290.09	NGVD29		-9999	-9999	
31	USGS	311739111132601	D-13-11 16DCA2	32.2942416	-111.2245465	NAD27	03/26/1971	443.1		2290.09	NGVD29		-9999	-9999	
32	USGS	312544111212401	D-23-10 18ACA	31.42898229	-111.3573211	NAD27	05/19/1981	14.2		3670	NGVD29	S	80	-9999	
33	USGS	312547111211401	D-23-10 18AAC	31.42981562	-111.3545433	NAD27	05/19/1981	102.4		3725	NGVD29	H	160	160	
34	USGS	312547111211401	D-23-10 18AAC	31.42981562	-111.3545433	NAD27	12/12/1984	11.7		3725	NGVD29	H	160	160	
35	USGS	312600111200301	D-23-10 16BBB	31.43342678	-111.3348204	NAD27	05/19/1981	140.5		3820	NGVD29	S	200	200	
36	USGS	312656111142801	D-23-11 05CDB	31.44898255	-111.2417617	NAD27	05/20/1981	7.3		3998	NGVD29	U	10	10	
37	USGS	312743111201701	D-23-10 05AAB	31.46203698	-111.3387096	NAD27	05/19/1981	274.5		3950	NGVD29	S	300	300	
38	USGS	312823111213501	D-22-10 31BDB	31.47314759	-111.3603771	NAD27	05/19/1981	209.8		3950	NGVD29	S	-9999	-9999	
39	USGS	312823111213501	D-22-10 31BDB	31.47314759	-111.3603771	NAD27	12/12/1984	203.8		3950	NGVD29	S	-9999	-9999	
40	USGS	312826111305201	D-22-08 34BAD	31.47397982	-111.5151049	NAD27	03/24/1981	125.8		3433	NGVD29	S	-9999	430	
41	USGS	312826111305201	D-22-08 34BAD	31.47397982	-111.5151049	NAD27	11/27/1984	72.2		3433	NGVD29	S	-9999	430	
42	USGS	312827111303301	D-22-08 34ABD	31.47425761	-111.509827	NAD27	01/10/1964	195.2		3450	NGVD29	U	430	430	

Introduction | TucsonAMA_data | Example_SiteID List

Figure 1. Example of *original data worksheet* for use with Excel® tools for analyzing water-level records.

Table 1. Required location of data on worksheet for data used by Visual Basic for Applications tools for analyzing groundwater data and displaying in geographic information systems.

[Required information is shaded red. Data used in the hydrograph-creation tool described in this report, but not required, are shaded orange.]

Data	Worksheet Column	NWIS Code	Example
Source agency code	A	C004	USGS
Site ID (station number)	B	C001	312544111212401
Local well number	C	C012	D-23-10 18ACA
Latitude	D	C909	31.42898
Longitude	E	C910	-111.357
Lat/Long Datum	F	C036	NAD83
Water-level measurement date	G	C235	04/01/1957
Water-level below land surface OR water-level altitude	H	C237 OR computed	260.00 OR 2890.00
Water-level status	I	C238	P
Altitude of land surface	J	C016	3150.00
Altitude datum	K	C022	NGVD29
Primary use of water	L	C024	U
Hole depth	M	C027	358.00
Well depth	N	C028	320.00

Spreadsheet Tools

Two workbooks are described in this report, Well_Analyses.xlsm and Well_Hydrographs.xlsm. The Well_Analyses.xlsm workbook contains three tools—one to combine well data from multiple sources and format the resulting combined dataset, one to provide summary information on the dataset, and one to pull a user-selected subset of wells from the dataset and produce a file for use within ArcMap™ for projecting well locations. The Well-Hydrographs.xlsm workbook contains two tools: one that creates hydrographs and one that produces PDF-formatted versions of these hydrographs to link with well locations in ArcMap. The two workbooks are discussed separately below, with each tool described in detail.

To provide clarity, from this point forward in this report **workbooks** will be emphasized with bold text and *worksheets* will be formatted in italics. Three specific water-level data worksheets used frequently in following sections are: *original data worksheet* containing the original data from NWIS or other databases; *formatted data worksheet* containing a copy of the original data worksheet that has been combined (if multiple data sources were present in the original data worksheet), sorted, and formatted for use with other macros in this report; and *subset data worksheet* containing a subset of the formatted data worksheet based on user-defined choices (for example, observations that fall within a user-defined date range).

The macros described in this report are VBA macros, and therefore macros must be enabled in Excel for them to run (see Excel help for more information). To run the tools in Excel 2007, select the "View" menu tab, select "Macros," then "View Macros," or use keystrokes Alt + F8. This brings up a window with a list of the publicly visible macros in the spreadsheet (fig. 2). Select a macro from the list, then click "Run." The user may copy water-level data into a blank worksheet in the **Well_Analyses.xlsm** or **Well_Hydrographs.xlsm** workbooks, copy a data worksheet from another workbook into the **Well_Analyses.xlsm** or **Well_Hydrographs.xlsm** workbooks, or run the macros from a separate workbook as long as the workbook containing the macros remains open. If the data are kept in a separate workbook, then the macros should be run from within this workbook by either of the methods described above.

To optimize speed for large datasets, screen updating is turned off during execution of the macros in these workbooks. While running, only the status bar at the bottom of the Excel window will indicate progress. Upon

completion of the macro, a message box will appear that informs the user of the elapsed time of the macro run.

Well_Analyses.xlsm

The **Well_Analyses.xlsm** workbook contains three publicly visible macros (there are others that are not visible that work behind the scenes): combine_format_datasets, observation_stats, and create_xyz_worksheet. These tools are written to combine water-level datasets, summarize information from a water-level dataset, and create an input file of the summary information capable of being projected in ArcMap. Each macro is discussed in detail below.

Combine_format_datasets

This tool was designed to combine water-level data from multiple data sources and format the resulting dataset for use by other macros in the workbook. This functionality is particularly useful when there are multiple sources for water-level data in the area under investigation. For example, in the state of Arizona, much of the recent water-level information is collected and distributed by the Arizona Department of Water Resources, while the USGS NWIS-GWSI system contains most of the older data as well as recent data in areas where USGS projects are ongoing. Other site-specific water-level information may also be available from other entities, including cities, counties, and other government agencies. The first task when compiling data from multiple sources is often to combine the datasets for further use.

When using this tool, the user is first prompted for the worksheet containing the water-level data to be combined and formatted (fig. 3). The worksheet selected should be the one containing all of the original data. Data from all datasets should be present in this same *original data worksheet*. The location of data in this worksheet is specified by the rules described in the Data Requirements section, but the sequential order of the multiple datasets to be combined is irrelevant, as long as there are no blank rows between the multiple datasets.

The combine_format_datasets tool first copies this *original data worksheet* to a new worksheet with the

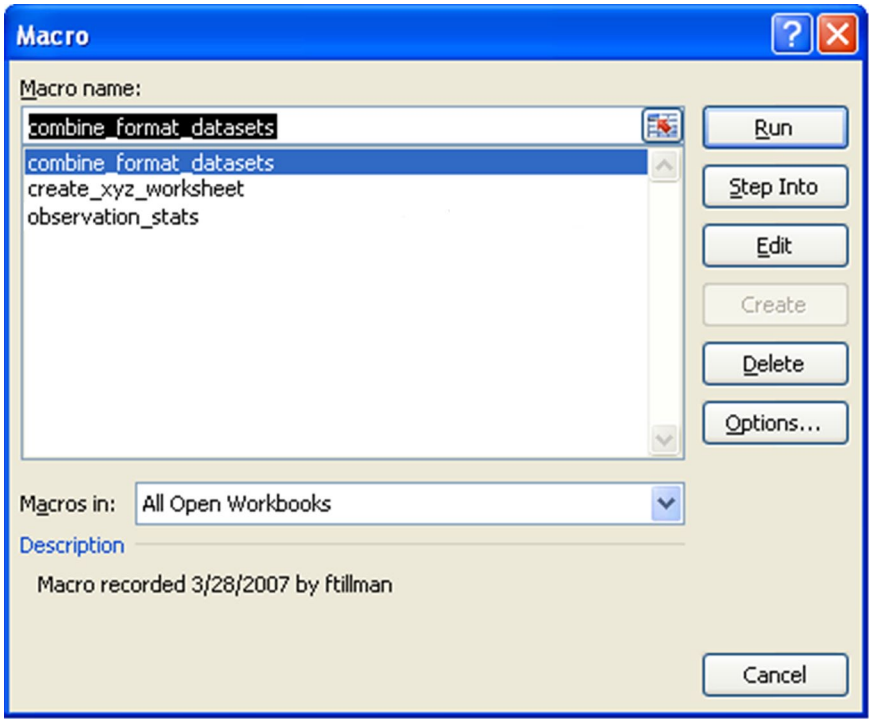


Figure 2. User-interface window in Microsoft Excel® for selecting tools for analyzing water-level records.

original data worksheet name plus "_FMT". The macro may fail if there are more than 31 total characters in the worksheet name. If this occurs, shorten the name of the *original data worksheet* and run the macro again. By working only on a copy of the original dataset, all original data are left unaltered. The macro searches through each row of sequential data, deleting rows that have no water-level observation or observation date. Columns that contain the site identification number, hole depth, well depth, and well altitude are all formatted as numbers. All blank cells within the hole depth, well depth, and well altitude fields are replaced by a no-data marker of -9999. Water-level observations are formatted to two decimal places, and the observation date is changed to mm/dd/yyyy format. The macro will detect if any water-level observation date is originally in yyyymm format and convert this date to mm/15/yyyy in the observation date column. The original yyyymm date is retained in the *formatted data worksheet* and placed in the same row in column CW (column 101). The user can look through this column to verify if dates in question were formatted properly. If a date is encountered that is not in any of the correct original formats, this row of data is deleted from the *formatted data worksheet*.

All data on the new *formatted data worksheet* are sorted first by site identification number, then by observation date. The macro then searches through the water-level observations and dates for each unique site identification number, deleting instances of identical date and observation. Longitude values are made negative for ease of use with ArcMap. If positive longitude values are required, they may be changed back using Excel functions. If any records for a site have a blank entry for latitude or longitude, then a nonblank value from a record for the same well is copied to the blank entry.

It is recommended that the combine_format_datasets macro be run on an *original data worksheet* even if there is only one source for the data. It is important for other macros described in this report that the data be formatted properly (for example, that site identification numbers

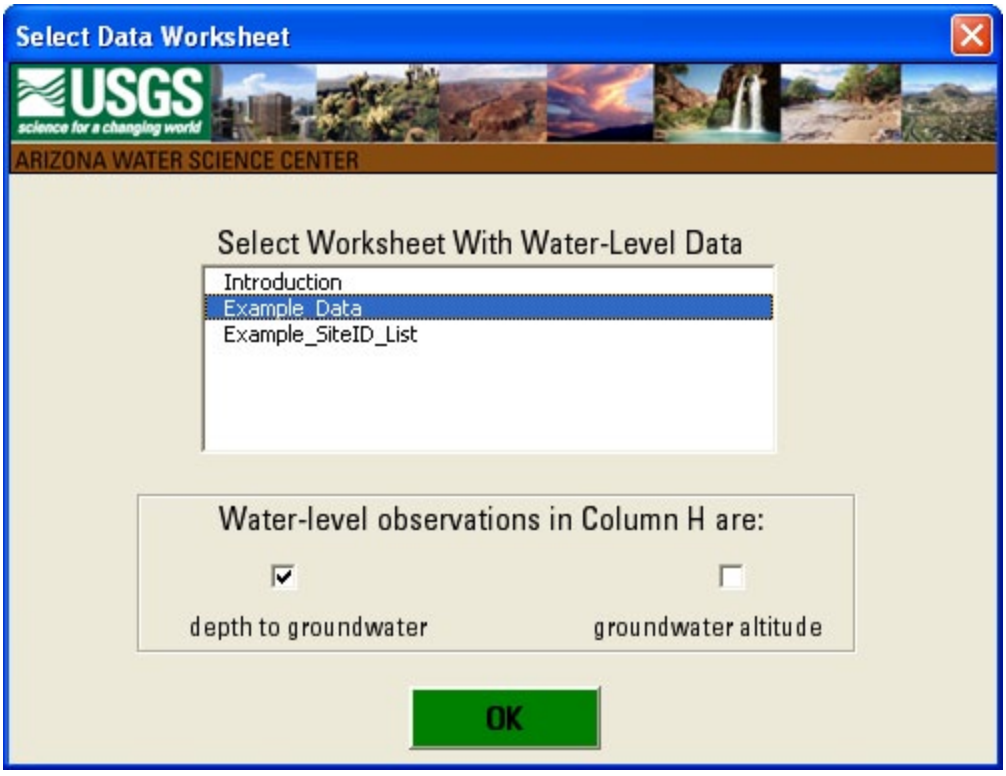


Figure 3. User-interface window in Microsoft Excel® for selecting the worksheet containing the water-level data (*original data worksheet*) and specifying whether the observations are depth to groundwater or water-level altitude.

be formatted as numbers and not text and that observation dates be recognizable in Excel as dates) and that no blanks exist in either the water-level observation or observation date cells. Because all combining and formatting is done on a copy of the *original dataset worksheet*, nothing is lost by running this macro on the original dataset.

observation_stats

This macro provides general information on water-level observations in the formatted dataset. Upon running this macro, the user is again prompted for the worksheet containing the dataset to be analyzed (fig. 3). The dataset selected should be the *formatted data worksheet* created by the preceding combine_format_datasets macro. A new worksheet titled *observation_stats* is created in the workbook to contain the summary information. The macro will fail if a worksheet with this name already exists in the active workbook. Therefore, it is recommended that the worksheet name be changed after completion of the macro. A table with each unique site identification number along with the number of water-level observations, the earliest observation date, and the last observation date is created on the worksheet, along with three additional summary tables. A Site Summary table is created that lists the number of unique site identification numbers in the dataset and the number of sites with water-level observations whose water-level status was flagged in the Water-level status (C238) column (column I) as either pumping, recently pumped, nearby pumping, nearby recently pumped, injecting, nearby injecting, or dry (fig. 4). A Water-Level Observations Summary table lists the number of sites with a minimum number of water-level observations ranging from 5 to 300 (fig. 4). A Summary of Water-Level Observations Time Span table presents information on the number of sites in the dataset whose first and last water-level observations are before and after specific dates (fig. 4).

create_xyz_worksheet

This macro creates two worksheets: one with a subset of the data from the *formatted data worksheet* based on user-specified options (*subset data worksheet*) and a second with summary information on water levels for each site in the *subset data worksheet* that can be imported into ArcMap as an "XY Data" file (*XY Data input file worksheet*). The user is first prompted for the worksheet containing the water-level data to be analyzed (fig. 3). As with the observation_stats macro, this should be the *formatted data worksheet*. A second window presents options for limiting data in the *formatted data worksheet* to be analyzed and included in the *XY Data input file worksheet* and *subset data worksheet* (fig. 5). Each of these four options is discussed separately below. Because the *subset data worksheet* and the *XY Data input file worksheet* are common to all possible subsets of the formatted data, these are also discussed in the sections below. After the user selects the method for limiting the data to be analyzed, the user is asked to keep or eliminate water-level observations whose status was flagged as either pumping, recently pumped, nearby pumping, nearby recently pumped, injecting, or nearby injecting (fig. 6). Flagged data chosen for elimination will not be included in the *subset data worksheet* or the *XY Data input file worksheet*. The macro will fail if worksheets in the active workbook exist with the same names as those the macro is attempting to create. Therefore, it is recommended that the newly created worksheets be renamed after completion of the macro.

All Data

The first data-limiting option is to include and analyze all data from the *formatted data worksheet* (fig. 5). The macro copies the *formatted data worksheet* to a *subset data worksheet* and eliminates only those flagged observations chosen by the user (fig. 6). The name of the *subset data worksheet* begins with "data_ALL_", with an extension indicating the flagged data eliminated (for example, data_ALL_no_PSI indicates no pumping, nearby pumping, or injecting flagged observations were included). A summary *XY Data input file worksheet* is created based on data in the *subset data worksheet* and is named "xyz_ALL_", with the same extension as the *subset data worksheet*.

List of Site Identification Numbers

The second data-limiting option is to only include and analyze a subset of the *formatted data worksheet* based on a user-supplied list of site identification numbers (fig. 5). The user is prompted for the worksheet that contains the list of site identification numbers (fig. 7). This list also has specific location and formatting requirements: the site identification numbers must be in Column A and begin on row 2, they must be formatted as numbers, and they must be sorted from lowest to highest values (fig. 8). The macro creates a *subset data worksheet* containing only data from sites on the site list, eliminating flagged observations chosen by the user. The name of the *subset data worksheet*

F	G	H	I	J
Site Summary			Water-Level Observations Summary	
Number of unique site id's:	3555		Min Observations	Number of Sites
Number of sites with observations flagged as			5	1593
Pumping (P)	0		15	963
Recently Pumped (R)	28		25	583
Nearby Pumping (S)	3		50	154
Nearby Recently Pumped (T)	0		100	46
Well Injecting (I)	0		150	14
Nearby Injecting (U)	0		200	6
Dry (D)	0		250	3
			300	2

L	M	N	O	P	Q	R	S	T	U
Summary of Water-Level Observations Time Span									
	And Continuing After -->								
Record Beginning Before	1/1/1991	1/1/1993	1/1/1995	1/1/1997	1/1/1999	1/1/2001	1/1/2003	1/1/2005	1/1/2007
1/1/1910	0	0	0	0	0	0	0	0	0
1/1/1920	0	0	0	0	0	0	0	0	0
1/1/1930	1	1	1	1	1	1	1	1	0
1/1/1940	21	21	19	16	16	12	12	9	0
1/1/1950	128	128	124	109	107	83	81	71	0
1/1/1960	415	412	403	362	354	317	313	254	0
1/1/1970	631	628	618	554	543	493	486	394	0
1/1/1980	752	748	738	642	631	572	565	460	0
1/1/1990	1332	1325	1296	1060	1043	942	932	794	0
				Number of Sites					

Figure 4. Example output of observation_stats program presenting summary information about water-level dataset.

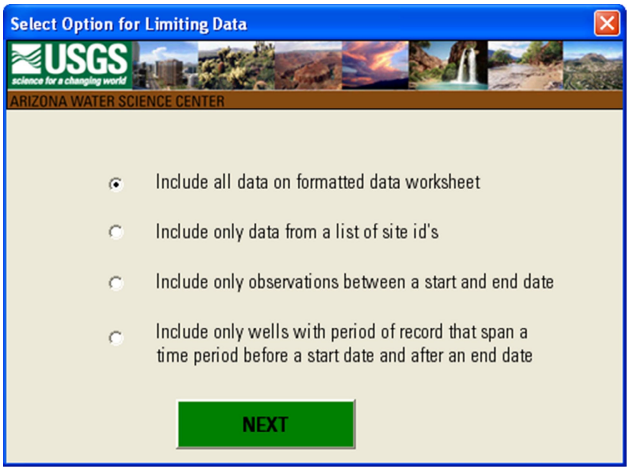


Figure 5. User-interface window in Microsoft Excel® for selecting data subsets for the water-level analysis tool.

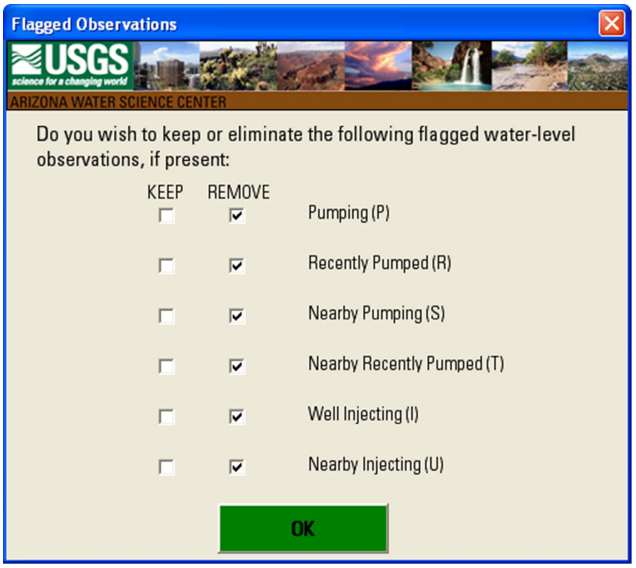


Figure 6. User-interface window in Microsoft Excel® for selecting options for retaining or removing water-level observations on the basis of water-level status.

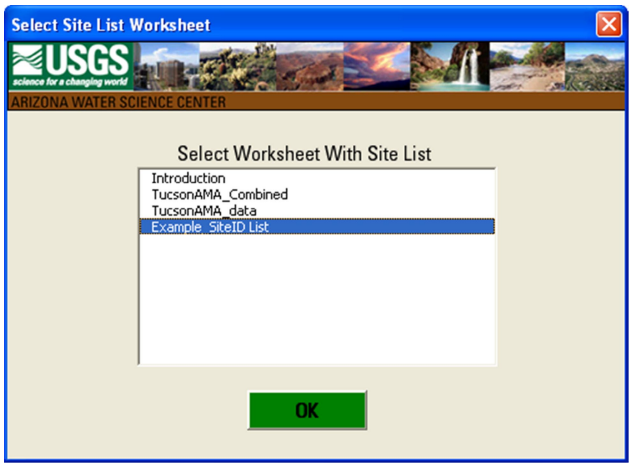
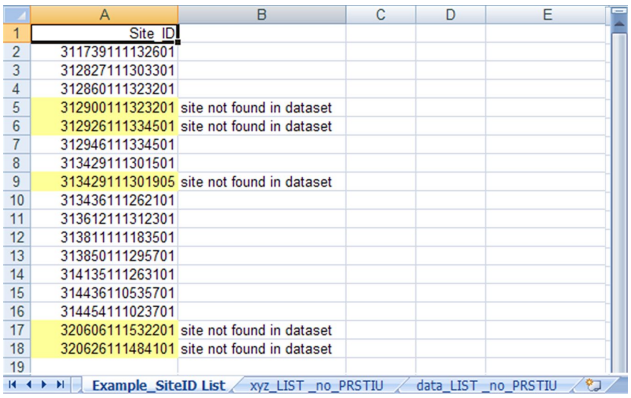


Figure 7. User-interface window in Microsoft Excel® for selecting worksheet containing a list of site identification numbers to be included in a subset of water-level data.



	A	B	C	D	E
1	Site_ID				
2	311739111132601				
3	312827111303301				
4	312860111323201				
5	312900111323201	site not found in dataset			
6	312926111334501	site not found in dataset			
7	312946111334501				
8	313429111301501				
9	313429111301905	site not found in dataset			
10	313436111262101				
11	313612111312301				
12	313811111183501				
13	313850111295701				
14	314135111263101				
15	314436110535701				
16	314454111023701				
17	320606111532201	site not found in dataset			
18	320626111484101	site not found in dataset			
19					

Example_SiteID List | xyz_LIST_no_PRSTIU | data_LIST_no_PRSTIU

Figure 8. Example of a list of site identification numbers used to create a subset of water-level data.

begins with "data_LIST_", with an extension indicating the flagged data eliminated. A summary *XY Data input file worksheet* is created based on data in the *subset data worksheet*, and is named "xyz_LIST_", with the same extension as the *subset data worksheet*. If no data are found for a site identification number on the list, then the site number is highlighted on the site list and the message "site not found in dataset" is displayed next to it (fig. 8).

Start and End Dates of Observation

The third data-limiting option is to only include and analyze a subset of the *formatted data worksheet* based on observations that fall between user-specified start and end dates (fig. 5). This option is useful for investigating changes in water levels during a time period of interest. The user is prompted for the earliest and latest observation dates that define the period of interest (fig. 9). The dates must be entered in mm/dd/yyyy format. The macro creates a *subset data worksheet* of only water-level observations that fall between the earliest and latest dates, eliminating flagged observations chosen by the user. The name of the *subset data worksheet* begins with "data_DATES_", with an extension indicating the flagged data eliminated. A summary *XY Data input file worksheet* is created based on data in the *subset data worksheet* and is named "xyz_DATES_", with the same extension as the *subset data worksheet*.

Wells with Records that Span a Time Period

The final data-limiting option is to only include and analyze a subset of the *formatted data worksheet* based on wells whose period of record begins before a user-specified beginning date and continues past a user-specified ending date (fig. 5). For example, it may be informative to obtain data and summary information for wells whose records begin during a predevelopment time period and continue until recent years. The user is prompted for the cutoff dates for the earliest and last observations (fig. 10) that define the time span of interest. The dates must be entered in mm/dd/yyyy format. Only wells that have at least one observation before the earliest date and one observation after the last date are accepted. All water-level data from these accepted wells, minus any flagged observations chosen by the user, are included in the *subset data worksheet*. The name of the *subset data worksheet* begins with "data_SPAN_", with an extension indicating the flagged data eliminated. A summary *XY Data input file worksheet* is created based on data in the *subset data*

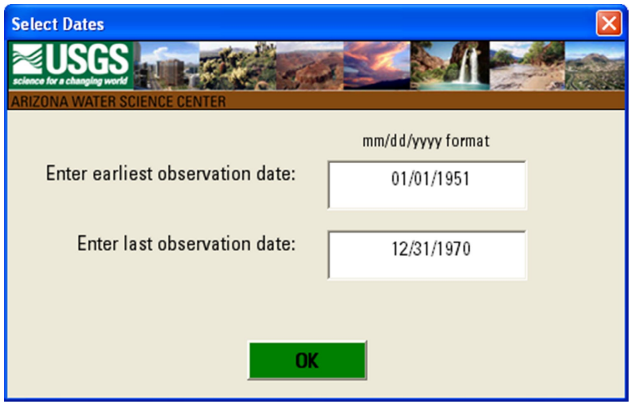


Figure 9. User-interface window in Microsoft Excel® for selecting earliest and latest observation dates for creating a subset of water-level data.

Table 2. Summary information tabulated by create_xyz_worksheet tool from data in a *subset data worksheet.*

Heading	Explanation
Longitude	longitude or y-coordinate of the site location as provided on the *subset data worksheet*
Latitude	latitude or x-coordinate of the site location as provided on the *subset data worksheet*
Site_ID	site identification number formatted as text
Well_Numbr	well number
Avg_WtrLvl	average water level for the site for observations on the *subset data worksheet*
StndDev_WL	standard deviation of water-level observations for the site for observations on the *subset data worksheet*
n_Obsrvtns	number of water-level observations for the site for observations on the *subset data worksheet*
1stObsDate	first water-level observation date for the site on the *subset data worksheet*
1stWLObs	first water-level observation for the site on the *subset data worksheet*
LstObsDate	last water-level observation date for the site on the *subset data worksheet*
LstWLObs	last water-level observation for the site on the *subset data worksheet*
MinDepthDt	date of the minimum depth to water for the site on the *subset data worksheet*
MinDepthOb	minimum depth to water for the site on the *subset data worksheet*
MaxDepthDt	date of the maximum depth to water for the site on the *subset data worksheet*
MaxDepthOb	maximum depth to water for the site on the *subset data worksheet*
WellRecord	number of days between the first and last water-level observation for the site on the *subset data worksheet*
MaxDecline	maximum water-level decline between any two observations for the site on the *subset data worksheet*
MaxRise	the maximum water-level rise between any two observations for the site on the *subset data worksheet*
Hole_Depth	hole depth for the site on the *subset data worksheet*
Well_Depth	well depth for the site on the *subset data worksheet*
LandSurfElv	land surface elevation for the site on the *subset data worksheet*
Hydrograph	a stem for creating a hyperlink to the well hydrograph in ArcGIS (see instructions for Well_Hydrographs.xlsm macros)

worksheet and is named "xyz_SPAN_", with the same extension as the *subset data worksheet*.

Summary XY Data Worksheet

The summary *XY Data input file worksheet* is created on the basis of water-level data in the *subset data worksheet*. It is important to recall that water-level information summarized in the *XY Data input file worksheet* is based solely on data contained on the *subset data worksheet* and will therefore be limited by user choices made when producing the subset worksheet. The first three rows of this worksheet list the user selections that were chosen to produce the *subset data worksheet* and then summarized in the *XY Data input file worksheet*. In this way, users can keep track of what data are being summarized. For example, the first row for an xyz_DATES worksheet

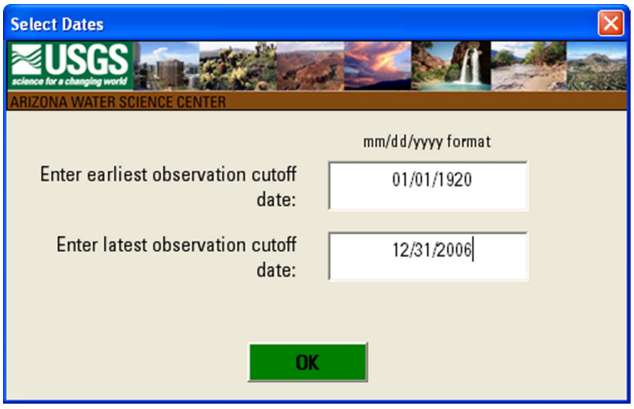


Figure 10. User-interface window in Microsoft Excel® for selecting earliest and latest water-level observation dates that define the time span of interest for creating a subset of water-level data.

might be: "Summary of OBSERVATIONS BETWEEN 1/1/1951 AND 12/31/1970"; and the second row might be: "_no_PRSTIU FLAGGED OBSERVATIONS." These first three rows of information must be deleted before the worksheet can be imported into ArcMap as an "XY Data" file. It may be useful to first save a copy of the *XY Data input file worksheet*, delete the first three rows, and then import this copy into ArcMap.

There is a row of summary information in the *XY Data input file worksheet* for each site identification number that includes spatial location information along with water-level summary information such as the average and standard deviation of water levels, first and last water-level observations and dates, and maximum water-level rise and decline, among others (table 2). Heading titles are chosen to fit within ArcMap's limit of 10 characters. Site identification numbers are formatted as 15-character text in order to avoid the imposition of scientific notation formatting in ArcMap. The Hydrograph column contains the characters "PATH\" followed by the site identification number followed by ".pdf". Macros contained in the **Well_Hydrograph.xlsm** workbook described in this report will produce PDF-formatted hydrographs that can be hyperlinked to the "XY Data" input file in ArcMap. The "PATH" string in the Hydrograph column must be replaced (using Excel's find and replace function, for example) with the path to the location of the hydrographs for the hyperlinking to work.

Well_Hydrographs.xlsm

The **Well_Hydrographs.xlsm** workbook contains macros that produce hydrograph charts from water-level data and make PDF-formatted versions of the hydrographs that can be hyperlinked to well locations in ArcMap. The workbook contains two publicly visible macros: makeExcelHydrographs and makePDFfromChart. These macros must be run from the workbook that contains the *subset data worksheet*, or the subset data worksheet must first be copied to the **Well_Hydrographs.xlsm** workbook. As the makePDFfromChart macro creates PDF-formatted versions of existing chart sheets, the makeExcelHydrographs macro must be run first, or other existing chart sheets must be present in the active workbook.

makeExcelHydrographs

This macro creates water-level hydrographs from the *subset data worksheet* produced by the **Well_Analyses.xlsm** macros (or a similarly formatted data worksheet). The data location and formatting requirements for the makeExcelHydrographs macro are the same as for the **Well_Analyses.xlsm** macros (fig.1 and table 1). Either depth-to-water or water-level-altitude data may be used. The user is first prompted for the worksheet containing the formatted water-level data for which hydrographs are to be constructed (fig. 3). This should be the *subset data worksheet* produced by any of the user-defined options in the previously described create_xyz_worksheet macro in the **Well_Analyses.xlsm** workbook. A second window presents options for the horizontal (date of observation) and vertical (observation depth or altitude) axes (fig. 11). Manually scaling the axes produces consistent hydrographs for all sites in the *subset data worksheet* and allows easy visual comparison of data among wells. If the automatic option is chosen for the observation axis, the minimum value for the axis will be the minimum observation for the site rounded down to the nearest ten, and the maximum axis value will be the maximum observation for the site rounded up to the nearest ten. For example, if a site has depth-to-water values that range from 18 to 51 feet below land surface, then the vertical axis will range from 10 to 60 feet. If the automatic option is chosen for the date of observation axis, then that axis will range from January 1st of the year of the earliest observation to January 1st of the year after the last observation.

The macro creates a new *chart worksheet* for each site identification number in the *subset data worksheet* and plots the water-level observation versus the obser-

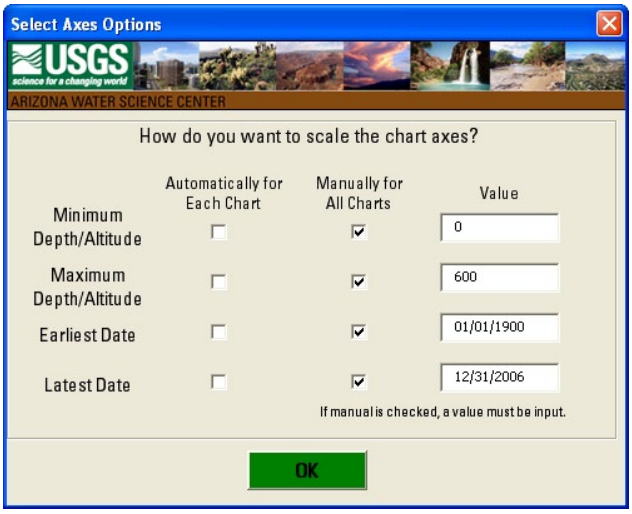


Figure 11. User-interface window in Microsoft Excel® for selecting axes options for well hydrographs.

vation date. Each *chart worksheet* is named by the site identification number. All charts are linked back to the original *subset data worksheet,* and any change in the worksheet will be reflected in the appropriate chart. The font for all text in the charts is set to Univers 57 Condensed. This font may either be changed manually within each chart, or the font used to create each chart may be changed by editing program lines 269, 318, 331, 339, 448, 463, 478, 492, 508, and 524. Well information, including the site identification number, well name, longitude and latitude of the well location, altitude of land surface, and well depth are included at the top of the hydrograph (fig. 12). When the hydrograph is printed or converted to a PDF, a "created on" date is placed in the header (fig. 12). An agency identifier logo may also be placed in the header by uncommenting program line 537 and entering the path to the graphic file. The water-level observations axis is adjusted depending on whether the data are depth to water (smallest value at the top, largest at the bottom) or water-level altitude (smallest value at the bottom, largest at the top). A solid line connects observations that are less than or equal to one year apart, with a dashed line connecting observations greater than one year apart. To facilitate plotting the solid and dashed lines, a worksheet entitled *SolidLineData* is created in each chart workbook. A new workbook is created to save each multiple of 50 charts. After each multiple of 50 charts, the user is prompted for a new workbook name to contain the next 50 charts. All charts in subsequent workbooks remain linked back to the original *subset data worksheet*. To maximize speed by reducing memory usage, only the workbook containing the original *subset data worksheet* remains open after reaching the 50 chart limit.

makePDFfromChart

This macro uses the free 2007 Microsoft Office® add-in "Microsoft Save as PDF or XPS" to export charts to the PDF format. The add-in is available through Microsoft's download site (http://www.microsoft.com/downloads). Once exported to the PDF format, the charts can then be hyperlinked to well locations in ArcMap. The macro prompts the user for the location of the folder where PDF charts will be saved. It is this path that should be used to replace the "PATH" string in the Hydrograph field of the *XY Data input file worksheet* described previously. The macro cycles through each chart in the active workbook and creates a PDF version of the chart using the site identification number plus ".pdf" as the file name. Only charts within the active workbook are exported to PDF. If multiple workbooks were created to hold more than 50 charts, then the macro will need to be run again from within each workbook that contains charts.

Displaying Results in ArcMap

Excel 2007 worksheets (.xlsx or .xlsm files) may be imported directly into ArcMap version 9.3. For earlier versions of ArcMap, the *XY Data input file worksheet* must first be saved either as an Excel 2003 file (.xls) or as a comma-separated value (.csv) file. The "PATH" string in the Hydrograph field of the *XY Data input file worksheet* must be replaced by the path to the PDF hydrographs to enable the hyperlinking feature. The first row must contain the field heading labels to be imported into ArcMap (the first three rows of user-selection information from the *XY Data input file worksheet* must first be deleted). To import the *XY Data input file worksheet* from the create_xyz_worksheet macro into ArcMap, with an ArcMap document open, select the "Tools" menu option, then "Add XY Data." Browse to the appropriate workbook, select the *XY Data input file worksheet*, and the longitude and latitude data should automatically be specified as the fields for the x and y coordinates. If not, use the pulldown menus to select these fields. Select the coordinate system of the well data, and the summary information for the wells is projected as a temporary point file in ArcMap.

To be able to use the hyperlinking feature, the temporary point file should be exported as a new feature class, such as a point shapefile (see ArcGIS Desktop™ help for more information). Once the well shapefile has been created, the hydrograph may be linked to the well location using field-based hyperlinks. To enable field-based hyperlinking, double-click the well shapefile in the table of contents, click the "Display" tab, check the "Support Hyperlinks using field" box, click the drop-down arrow and click the Hydrograph field. On the next row of information, select the "Document" radio button for hydrographs on your computer or the "URL" radio button for network-based hydrographs, then click "OK." To view the hydrographs, select the Hyperlink tool from the ArcMap toolbar (a lightning bolt), then click any of the well locations. A popup window will appear with the PDF hydrograph of the selected well.

The symbology properties in ArcMap can be used on the different fields of summary well information in the well shapefile. For example, using graduated colors on values in the MaxDecline field will highlight the

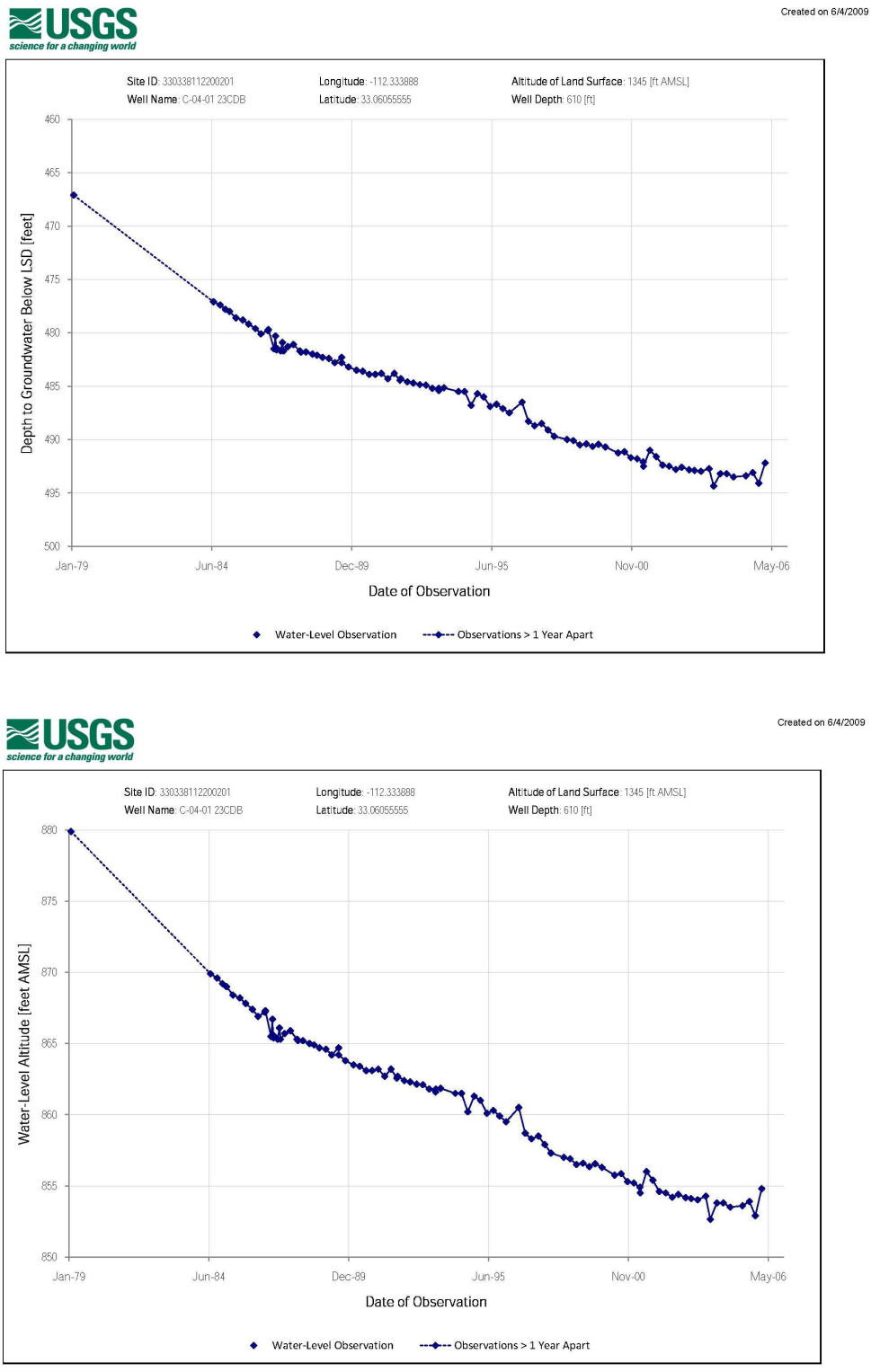


Figure 12. Example hydrographs produced by makeExcelHydrographs tool for water-level data given as depth to groundwater (top panel) and water-level altitude (bottom panel). LSD, land surface datum; AMSL, above mean sea level.

location and range of decline of water in wells that were observed to have falling water levels during the period of time represented by the *subset data worksheet*.

Summary and Conclusions

The macros described in this report provide an efficient method for processing water-level data from NWIS or other databases into usable information in a GIS environment. Macros are presented here for combining data from multiple sources and for summarizing general information on water-level observations in a dataset. Another macro creates an *XY Data input file worksheet* with water-level information for selected wells, such as first and last water-level observations, average water level, and maximum water-level decline and rise, among others. Wells and water-level observations may be included or excluded from this *XY Data input file worksheet* on the basis of user-specified selection criteria, such as including only wells on a specified list, only observations between specified dates, or only wells that span a given time period. This *XY Data input file worksheet* can be imported into ArcMap and projected as a point shapefile. An additional workbook contains macros that create PDF-formatted well hydrographs from the selected wells that can be hyperlinked to well locations in ArcMap.

Although the macros described in this report must all be run on a *formatted data worksheet*, they may be run in differing combinations to produce desired results. For example, a user might want to investigate wells with more than a given number of water-level observations. To accomplish this, run the observation_stats macro on the *formatted data worksheet*, sort the resulting site identification numbers by number of observations, delete all sites with fewer than the specified number of observations, sort the remaining wells by site identification number, then use this as the site list for the create_xyz_ worksheet macro. Another example might be to produce a *subset data worksheet* using the timespan method for limiting data to long-term monitored wells, then use the dates method on this subset of data to focus in on observations during a shorter time period of interest. Using these sets of tools, many options are available to refine the dataset.

Additionally, the data in the *original data worksheet* may be processed in Excel before running any of these macros. For example, all of the original data might first be sorted by water-use category. Then the data categorized as irrigation could be copied to a new worksheet (in the locations required by the tools). The combine_format_datasets macro would then be run on this irrigation-only data worksheet, and it would be ready for use by any of the XY Data-producing and hydrograph macros. Additional fields of data not mentioned in this report may also be used to first limit the original dataset in Excel, or selection choices may be made in NWIS or other database queries. One useful example may be to limit wells to a desired screened interval using NWIS codes C073 and C074.

Spatial representation of groundwater information in GIS systems has become nearly ubiquitous. The usefulness of GIS systems for groundwater investigations is evident both during the many stages of a hydrologic investigation, as well as in presentation of the investigation results in a meaningful way to policymakers and the public. Easy-to-use tools are required to bring within reach of all hydrologists the ability to efficiently process raw data into useful information.

References Cited

Tillman, F.D, Leake, S.A., Flynn, M.E., Cordova, J.T., and Schonauer, K.T., 2007, An online interactive map service for displaying ground-water conditions in Arizona: U.S. Geological Survey Open-File Report 2007-1436, 16 p. [http://pubs.usgs.gov/of/2007/1436/, last accessed 06/04/2009].

Tillman, F.D, Leake, S.A., Flynn, M.E., Cordova, J.T., Schonauer, K.T., and Dickinson, J.E., 2008, Methods and indicators for assessment of regional groundwater conditions in the southwestern United States: U.S. Geological Survey Scientific Investigations Report 2008-5209, 22 p. [http://pubs.usgs.gov/sir/2008/5209/, last accessed 06/04/2009].

This page intentionally left blank

Appendix: Program Listings

data_combine_and_format

```
Public DataWkbkName As String

Sub combine_format_datasets()
‘
 VBA updated 6/4/2009 by ftillman
‘
‘ This code combines water-level datasets from two or more sources into one worksheet.
‘ The combined-data worksheet is then formatted for use by other VBAs in this workbook.

Dim allDataSheet As String
Dim Counter As Long, siteRowStart As Long, siteRowEnd As Long, totalrows As Long
Dim onesRange As Range, siteIDRange As Range
Dim yr As Integer, mth As Integer, dy As Integer, dateStringLength As Long
Dim LatFind As String, LongFind As String
Dim startTime As Long, endTime As Long, elapsedTimeSeconds As Long
Dim elapsedTime As Variant, units As String

   DataWkbkName = ActiveWorkbook.Name

   lstSelectDataWorksheet.Show
   allDataSheet = DataWkshtName

startTime = Timer

siteRowStart = 22   ‘    first row of data
totalrows = 0

‘  First, copy the worksheet data to a new worksheet
   Sheets(allDataSheet).Select
   Sheets(allDataSheet).Copy After:=Sheets(1)
   ActiveSheet.Name = allDataSheet & “_FMT”
   Sheets(allDataSheet & “_FMT”).Move After:=Sheets(Worksheets.Count)

‘’ This loop is only to count the number of rows
   Counter = siteRowStart
   Do While (Cells(Counter, 2)) > 0
     totalrows = totalrows + 1
     Counter = Counter + 1
   Loop

   Application.ScreenUpdating = False    ‘ turns off screen updating
   Application.DisplayStatusBar = True   ‘ makes sure that the statusbar is visible
   Application.StatusBar = “Deleting rows that have no date or water-level observation”

‘   Delete rows of data that have no date or water level observation
   Counter = siteRowStart
   Do While (Cells(Counter, 2)) > 0
```

```
48      If (IsEmpty(Cells(Counter, 8)) Or IsEmpty(Cells(Counter, 7))) Then
49        Rows(Counter).Select
50        Selection.Delete Shift:=xlUp
51        Counter = Counter - 1
52        totalrows = totalrows - 1
53      End If
54      Counter = Counter + 1
55    Loop
56
57  ‘   Convert site ID to a number and format it
58  ‘     First, create a column of 1s in the last column of the spreadsheet
59        Cells(siteRowStart, 101).Select
60        Selection.Value = “1”
61        Selection.Copy
62        Range(Cells(siteRowStart, 101), Cells(siteRowStart + totalrows - 1, 101)).Select
63        ActiveSheet.Paste
64        Selection.Copy
65
66  ‘     Copy this column of 1s and paste special-multiply with the site ID
67        Range(Cells(siteRowStart, 2), Cells(siteRowStart + totalrows - 1, 2)).Select
68  ‘     Paste special
69        Selection.PasteSpecial Paste:=xlPasteAll, Operation:=xlMultiply, SkipBlanks:=False, Transpose:=False
70  ‘     Format site id
71        Selection.NumberFormat = “0”
72
73
74  ‘     Ensure Well and Hole Depth and Well Altitude are also numbers (not text)
75        Range(Cells(siteRowStart, 101), Cells(siteRowStart + totalrows - 1, 101)).Select
76        Selection.Copy
77        Range(Cells(siteRowStart, 13), Cells(siteRowStart + totalrows - 1, 14)).Select
78        Selection.PasteSpecial Paste:=xlPasteAll, Operation:=xlMultiply, SkipBlanks:=True, Transpose:=False
79        Selection.NumberFormat = “0.00”
80
81        Range(Cells(siteRowStart, 101), Cells(siteRowStart + totalrows - 1, 101)).Select
82        Selection.Copy
83        Range(Cells(siteRowStart, 10), Cells(siteRowStart + totalrows - 1, 10)).Select
84        Selection.PasteSpecial Paste:=xlPasteAll, Operation:=xlMultiply, SkipBlanks:=True, Transpose:=False
85        Selection.NumberFormat = “0.00”
86
87  ‘     Delete column of 1s
88        Columns(“CW:CW”).Select
89        Selection.Delete Shift:=xlToLeft
90
91      Application.StatusBar = “Formatting data”
92
93        Counter = siteRowStart
94        Do While Counter <= (siteRowStart + totalrows - 1)
95
96  ‘       Replace blank cells with -9999 to indicate no data
97            If (Cells(Counter, 10) = 0#) Then
98              Cells(Counter, 10) = -9999
```

```
99          End If
100         If (Cells(Counter, 13) = 0#) Then
101          Cells(Counter, 13) = -9999
102         End If
103         If (Cells(Counter, 14) = 0#) Then
104          Cells(Counter, 14) = -9999
105         End If
106
107 '       Format date as mm/dd/yyyy
108         Cells(Counter, 7).Select
109         If IsDate(Cells(Counter, 7)) Then
110          Selection.NumberFormat = "mm/dd/yyyy"
111         Else
112
113         Cells(Counter, 101) = Cells(Counter, 7)  'copy the existing date to a new column
114         dateStringLength = Len(Cells(Counter, 7))
115
116 '         If date is in format yyyymmdd
117         If dateStringLength = 8 Then          'if date is greater than 4 digits
118           yr = Left(Cells(Counter, 7), 4)        ' parse first 4 digits as year
119           If yr < "1900" Then                'if year is less than 1900, then Excel can't handle it, so delete the row
120             Range(Cells(Counter, 1), Cells(Counter, 20)).Delete Shift:=xlUp
121             Counter = Counter - 1
122             totalrows = totalrows - 1
123             GoTo 1000
124            End If
125           mth = Mid(Cells(Counter, 7), 5, 2)      ' parse next 2 digits as month
126           dy = Right(Cells(Counter, 7), 2)        ' parse last 2 digits as day
127           Cells(Counter, 7).Value = mth & "/" & dy & "/" & yr     ' concatenate date
128           Cells(Counter, 7).NumberFormat = "mm/dd/yyyy"
129         ElseIf dateStringLength = 6 Then      'if date is in format yyyymm
130           yr = Left(Cells(Counter, 7), 4)        ' parse first 4 digits as year
131           mth = Right(Cells(Counter, 7), 2)       ' parse next 2 digits as month
132           dy = 15        ' parse last 2 digits as day
133           Cells(Counter, 7).Value = mth & "/" & dy & "/" & yr     ' concatenate date
134           Cells(Counter, 7).NumberFormat = "mm/dd/yyyy"
135         Else
136
137 '       For all other possibilities, including yyyy, just delete the row
138           Range(Cells(Counter, 1), Cells(Counter, 20)).Delete Shift:=xlUp
139           Counter = Counter - 1
140           totalrows = totalrows - 1
141         End If    'end test of date length
142 1000    End If      'end test of IsDate
143
144 '       If not already, make longitude value negative for use in ArcMap xyz file
145         If Cells(Counter, 5).Value > 0 Then
146          Cells(Counter, 5).Value = (Cells(Counter, 5).Value) * (-1)
147         End If
148
149       Counter = Counter + 1
```

```
150       Loop
151
152
153
154 '   Format water level as a number with 2 decimal places
155       Range(Cells(siteRowStart, 8), Cells(siteRowStart + totalrows - 1, 8)).Select
156       Selection.NumberFormat = "0.00"
157
158 '   Sort worksheet by site ID, then by observation date
159       Range(Cells(siteRowStart, 1), Cells(siteRowStart + totalrows - 1, 101)).Sort _
160       Key1:=Range(Cells(21, 2), Cells(siteRowStart + totalrows - 1, 2)), Order1:=xlAscending, _
161       Key2:=Range(Cells(21, 7), Cells(siteRowStart + totalrows - 1, 7)), Order2:=xlAscending
162
163
164 '   Delete duplicate entries of water level and observation date
165 '    first, get the range of current site id
166    Application.StatusBar = "Deleting duplicate entries"
167
168    Row = siteRowStart
169    siteRowEnd = siteRowStart
170    Do While (Cells(Row, 2)) > 0
171       Do While (Cells(Row, 2) = Cells(Row + 1, 2))
172          Row = Row + 1
173          siteRowEnd = Row
174       Loop
175
176    If siteRowStart < siteRowEnd Then    'only need to check for duplicates, lat/long if more than 1 observation
177
178       TestSite = siteRowStart
179       Do While (TestSite <= siteRowEnd - 1)
180 '    compare water level date and observation within site ID and delete if duplicates
181        TestedSite = TestSite + 1
182        Do While (TestedSite <= siteRowEnd)
183         If (Cells(TestSite, 7) = Cells(TestedSite, 7)) And (Cells(TestSite, 8) = Cells(TestedSite, 8)) Then
184           Range(Cells(TestedSite, 1), Cells(TestedSite, 101)).Delete Shift:=xlUp
185           siteRowEnd = siteRowEnd - 1     '1 row deleted
186           TestedSite = TestSite + 1       'start check over again to ensure no other duplicates
187         Else
188           GoTo 1111       'data is sorted by date, so if next consecutive obs is not the same then no other will be
189         End If
190        Loop   'check rest of site ID data
191
192    1111     TestSite = TestSite + 1
193       Loop    'check next site ID against rest of site ID data
194
195 '-----------------------------------------------------------------may want to remove this-------------------------
196 '    if there are any water-level observations with no lat/long values for a given site id, copy them from one
       that has this info
197       LatFind = -9999
198       LongFind = -9999
199
```

```
200      If siteRowStart < siteRowEnd Then
201       Counter = siteRowStart
202       Do While (Counter < siteRowEnd)
203          If Cells(Counter, 4) >= 0 Or Not (IsEmpty(Cells(Counter, 4))) Then  'get value
204             LatFind = Cells(Counter, 4).Value    'latitude
205             LongFind = Cells(Counter, 5).Value    'longitude
206             Exit Do
207          End If
208          Counter = Counter + 1
209       Loop
210
211       Do While Counter <= siteRowEnd
212          If Cells(Counter, 4) <= 0 Or IsEmpty(Cells(Counter, 4)) Then  'find blank and copy
213             Cells(Counter, 4).Value = LatFind     'latitude
214             Cells(Counter, 5).Value = LongFind   'longitude
215          End If
216          Counter = Counter + 1
217       Loop
218
219       End If  'End lat/long 1 data point check
220
221    End If      'End big 1 data point check
222
223    siteRowStart = siteRowEnd + 1
224    Row = siteRowStart
225    siteRowEnd = siteRowStart
226    Loop    'loop through the next siteID block of data
227
228 '-------------------------------------------------------------------------------------------------------------------
229
230    Counter = 22
231
232 '-------------------------------------------------------------------------------------------------------------------
233    Application.StatusBar = False
234
235    Application.ScreenUpdating = True
236
237   Cells(1, 1).Select
238
239   endTime = Timer
240   elapsedTimeSeconds = endTime - startTime
241   If elapsedTimeSeconds < 0 Then
242   elapsedTimeSeconds = (86400 - startTime) + endTime
243   End If
244   If elapsedTimeSeconds >= 60 Then
245   elapsedTime = elapsedTimeSeconds / 60
246   units = " minutes"
247   Else
248   elapsedTime = elapsedTimeSeconds
249   units = " seconds"
250   End If
```

```
251  MsgBox “data_combine_and_format program ended.  Elapsed time “ & Format(expression:=elapsedTime,
       Format:=”Standard”) & units
252
253  End Sub
```

observation_stats

```
1    ‘ VBA updated by Fred Tillman on 6-4-2009
2    ‘ This VBA code summarizes information on water-level observations including number of
3    ‘ unique site id’s, number of observations per well, first and last observation dates per well,
4    ‘ information on number of sites with observations flagges as certain values (such as pumped),
5    ‘ and information on the time span of observations for all sites in the dataset.
6
7
8    Sub observation_stats()
9    Dim allDataSheet As String, StatsSheet As String
10   Dim siteID As String, nObs As Integer, Earliest_Obs As String, Last_Obs As String
11   Dim Counter As Long, allRowStart As Long, allRowEnd As Long, StatsRow As Integer
12   Dim nP As Integer, nR As Integer, nS As Integer, nT As Integer, nI As Integer, nU As Integer, nD As Integer
13   Dim nCount(1 To 9, 1 To 9) As Single, Before(1 To 9) As Date, After(1 To 9) As Date
14   Dim startTime As Long, endTime As Long, elapsedTimeSeconds As Long, elapsedTime As Variant
15   Dim units As String
16
17      nP = 0
18      nR = 0
19      nS = 0
20      nT = 0
21      nI = 0
22      nU = 0
23      nD = 0
24
25   DataWkbkName = ActiveWorkbook.Name
26
27      lstSelectDataWorksheet.Show
28      allDataSheet = DataWkshtName
29
30    StatsSheet = “Observation_Stats”
31    Set NewSheet1 = Worksheets.Add
32    NewSheet1.Name = StatsSheet
33    Sheets(StatsSheet).Move After:=Sheets(Worksheets.Count)
34
35   startTime = Timer
36
37    Worksheets(StatsSheet).Cells(1, 1).Value = “Site_ID”
38    Worksheets(StatsSheet).Cells(1, 2).Value = “n_Observations”
39    Worksheets(StatsSheet).Cells(1, 3).Value = “Earliest_Observation”
40    Worksheets(StatsSheet).Cells(1, 4).Value = “Last_Observation”
41
42    allRowStart = 22          ‘22
```

```
43  StatsRow = 2
44
45  Worksheets(allDataSheet).Activate
46
47   Application.ScreenUpdating = False    ‘ turns off screen updating
48   Application.DisplayStatusBar = True   ‘ makes sure that the statusbar is visible
49   Application.StatusBar = “Cataloging All Site IDs”
50
51  Do While (Worksheets(allDataSheet).Cells(allRowStart, 2)) > 0
52
53   ‘ first, get the range of current site id
54   Counter = allRowStart
55   allRowEnd = allRowStart
56   Do While (Cells(Counter, 2) = Cells(Counter + 1, 2))
57      Counter = Counter + 1
58      allRowEnd = Counter
59   Loop
60
61   ‘ check water-level status and update flag count
62   For Counter = allRowStart To allRowEnd
63      If Cells(allRowStart, 9) = “P” Then
64        nP = (nP + 1)
65        Exit For
66      ElseIf Cells(allRowStart, 9) = “R” Then
67        nR = (nR + 1)
68        Exit For
69      ElseIf Cells(allRowStart, 9) = “S” Then
70        nS = (nS + 1)
71        Exit For
72      ElseIf Cells(allRowStart, 9) = “T” Then
73        nT = (nT + 1)
74        Exit For
75      ElseIf Cells(allRowStart, 9) = “I” Then
76        nI = (nI + 1)
77        Exit For
78      ElseIf Cells(allRowStart, 9) = “U” Then
79        nU = (nU + 1)
80        Exit For
81      ElseIf Cells(allRowStart, 9) = “D” Then
82        nD = (nD + 1)
83        Exit For
84      End If
85   Next Counter
86
87   siteID = Cells(allRowStart, 2)          ‘ save siteID
88   nObs = allRowEnd - allRowStart + 1
89   Earliest_Obs = Cells(allRowStart, 7)
90   Last_Obs = Cells(allRowEnd, 7)
91
92   ‘ write information for each site id to stats worksheet
93   Worksheets(StatsSheet).Cells(StatsRow, 1).Value = siteID
```

```
94     Worksheets(StatsSheet).Cells(StatsRow, 2).Value = nObs
95     Worksheets(StatsSheet).Cells(StatsRow, 3).Value = Earliest_Obs
96     Worksheets(StatsSheet).Cells(StatsRow, 4).Value = Last_Obs
97    StatsRow = StatsRow + 1
98
99    allRowStart = allRowEnd + 1
100
101   Loop        ‘ big loop to next site id
102
103   Worksheets(StatsSheet).Activate
104
105   Worksheets(StatsSheet).Range(“A2:A” & allRowEnd + 2).NumberFormat = “0”         ‘site ID
106
107
108   ‘----------Summary Statistics----------’
109     Application.StatusBar = “Creating Table of Summary Pumping Status”
110
111     ‘Set up table for water-level observations summary
112     Worksheets(StatsSheet).Cells(1, 6).Value = “Site Summary”
113     Worksheets(StatsSheet).Cells(2, 6).Value = “Number of unique site id’s:”
114     Worksheets(StatsSheet).Cells(3, 6).Value = “Number of sites with observations flagged as”
115     Worksheets(StatsSheet).Cells(4, 6).Value = “Pumping (P)”
116     Worksheets(StatsSheet).Cells(5, 6).Value = “Recently Pumped (R)”
117     Worksheets(StatsSheet).Cells(6, 6).Value = “Nearby Pumping (S)”
118     Worksheets(StatsSheet).Cells(7, 6).Value = “Nearby Recently Pumped (T)”
119     Worksheets(StatsSheet).Cells(8, 6).Value = “Well Injecting (I)”
120     Worksheets(StatsSheet).Cells(9, 6).Value = “Nearby Injecting (U)”
121     Worksheets(StatsSheet).Cells(10, 6).Value = “Dry (D)”
122
123    ‘ Count number of unique site ids
124    nSites = 0
125    Counter = 2
126    Do While (Not IsEmpty(Cells(Counter, 2)))
127      nSites = nSites + 1
128      Counter = Counter + 1
129    Loop
130
131     Worksheets(StatsSheet).Cells(2, 7).Value = nSites
132     Worksheets(StatsSheet).Cells(4, 7).Value = nP
133     Worksheets(StatsSheet).Cells(5, 7).Value = nR
134     Worksheets(StatsSheet).Cells(6, 7).Value = nS
135     Worksheets(StatsSheet).Cells(7, 7).Value = nT
136     Worksheets(StatsSheet).Cells(8, 7).Value = nI
137     Worksheets(StatsSheet).Cells(9, 7).Value = nU
138     Worksheets(StatsSheet).Cells(10, 7).Value = nD
139
140     Application.StatusBar = “Creating Table of Summary Water-Level Observations”
141
142     ‘Set up table for water-level observations summary
143     Worksheets(StatsSheet).Cells(1, 9).Value = “Water-Level Observations Summary”
144     Worksheets(StatsSheet).Cells(2, 9).Value = “Min Observations”
```

```
145   Worksheets(StatsSheet).Cells(2, 10).Value = “Number of Sites”
146
147   Counter = 2
148   nCharts = 0
149
150   Do While Counter <= 10          ‘ counts sites min observations
151     Select Case Counter
152       Case 1
153         Worksheets(StatsSheet).Cells(Counter + 1, 9).Value = 1
154       Case 2
155         Worksheets(StatsSheet).Cells(Counter + 1, 9).Value = 5
156       Case 3
157         Worksheets(StatsSheet).Cells(Counter + 1, 9).Value = 15
158       Case 4
159         Worksheets(StatsSheet).Cells(Counter + 1, 9).Value = 25
160       Case 5
161         Worksheets(StatsSheet).Cells(Counter + 1, 9).Value = 50
162       Case 6
163         Worksheets(StatsSheet).Cells(Counter + 1, 9).Value = 100
164       Case 7
165         Worksheets(StatsSheet).Cells(Counter + 1, 9).Value = 150
166       Case 8
167         Worksheets(StatsSheet).Cells(Counter + 1, 9).Value = 200
168       Case 9
169         Worksheets(StatsSheet).Cells(Counter + 1, 9).Value = 250
170       Case 10
171         Worksheets(StatsSheet).Cells(Counter + 1, 9).Value = 300
172     End Select
173
174
175
176     Row = 1 ‘sort through summary data to see if sites have minimum number of observations
177     Do While (Worksheets(StatsSheet).Cells(Row, 2).Value) > 0
178      Row = Row + 1
179      If ((Cells(Row, 2).Value) >= Cells(Counter + 1, 9).Value) Then
180        nCharts = nCharts + 1
181      End If
182     Loop
183
184     Worksheets(StatsSheet).Cells(Counter + 1, 10).Value = nCharts
185
186     Counter = Counter + 1    ‘ Increment Counter
187     nCharts = 0
188
189   Loop
190
191
192
193   Application.StatusBar = “Creating Table of Summary Water-Level Observations Time Span”
194   n = 2
195   Worksheets(StatsSheet).Cells(n - 1, 12).Value = “Summary of Water-Level Observations Time Span”
```

```
196    Worksheets(StatsSheet).Cells(n + 1, 12).Value = “Record Beginning Before”
197    Worksheets(StatsSheet).Cells(n, 13).Value = “And Continuing After -->”
198    Worksheets(StatsSheet).Cells(n + 11, 17).Value = “Number of Sites”
199
200    Worksheets(StatsSheet).Cells(n + 2, 12).Value = “1/1/1910”
201    Worksheets(StatsSheet).Cells(n + 3, 12).Value = “1/1/1920”
202    Worksheets(StatsSheet).Cells(n + 4, 12).Value = “1/1/1930”
203    Worksheets(StatsSheet).Cells(n + 5, 12).Value = “1/1/1940”
204    Worksheets(StatsSheet).Cells(n + 6, 12).Value = “1/1/1950”
205    Worksheets(StatsSheet).Cells(n + 7, 12).Value = “1/1/1960”
206    Worksheets(StatsSheet).Cells(n + 8, 12).Value = “1/1/1970”
207    Worksheets(StatsSheet).Cells(n + 9, 12).Value = “1/1/1980”
208    Worksheets(StatsSheet).Cells(n + 10, 12).Value = “1/1/1990”
209
210    Worksheets(StatsSheet).Cells(n + 1, 13).Value = “1/1/1991”
211    Worksheets(StatsSheet).Cells(n + 1, 14).Value = “1/1/1993”
212    Worksheets(StatsSheet).Cells(n + 1, 15).Value = “1/1/1995”
213    Worksheets(StatsSheet).Cells(n + 1, 16).Value = “1/1/1997”
214    Worksheets(StatsSheet).Cells(n + 1, 17).Value = “1/1/1999”
215    Worksheets(StatsSheet).Cells(n + 1, 18).Value = “1/1/2001”
216    Worksheets(StatsSheet).Cells(n + 1, 19).Value = “1/1/2003”
217    Worksheets(StatsSheet).Cells(n + 1, 20).Value = “1/1/2005”
218    Worksheets(StatsSheet).Cells(n + 1, 21).Value = “1/1/2007”
219
220    Rvert = “L” & n + 2 & “:L” & n + 10
221    Range(Rvert).Select
222    With Selection.Borders(xlEdgeRight)
223       .LineStyle = xlContinuous
224       .Weight = xlMedium
225       .ColorIndex = xlAutomatic
226    End With
227
228    Rhoriz = “L” & n + 2 & “:U” & n + 2
229    Range(Rhoriz).Select
230    With Selection.Borders(xlEdgeTop)
231       .LineStyle = xlContinuous
232       .Weight = xlMedium
233       .ColorIndex = xlAutomatic
234    End With
235
236
237   ‘populate date arrays
238    For L = 1 To 9
239       Before(L) = Cells(L + 3, 12).Value
240       After(L) = Cells(3, L + 12).Value
241    Next L
242
243   ‘initialize nCount array
244    For L = 1 To 9
245      For j = 1 To 9
246       nCount(L, j) = 0
```

```
247     Next j
248    Next L
249
250
251    allRowStart = 2 ‘after first time through, this will be allRowEnd + 1
252
253    Do While Cells(allRowStart, 2) >= 1
254      For L = 1 To 9
255       For j = 1 To 9
256        If Cells(allRowStart, 3) <= Before(L) And Cells(allRowStart, 4) >= After(j) Then
257         nCount(L, j) = nCount(L, j) + 1
258        End If
259       Next j
260      Next L
261   allRowStart = allRowStart + 1
262   Loop ‘go to next site id
263
264   ‘write values to correct spot on table
265    For L = 1 To 9
266     For j = 1 To 9
267      Worksheets(StatsSheet).Cells(3 + L, j + 12).Value = nCount(L, j)
268     Next j
269    Next L
270
271
272   ‘-----FORMATTING--------------------------------------------------------------------
273
274   ‘widen first four columns
275    Columns(“A:D”).Select
276    Columns(“A:D”).EntireColumn.AutoFit
277
278   ‘format site summary table
279    Range(“F4:F10”).Select
280    With Selection
281        .HorizontalAlignment = xlRight
282    End With
283    Range(“G2:G10”).Select
284    With Selection
285        .HorizontalAlignment = xlCenter
286    End With
287
288    Range(“F1:G12”).Select
289    Selection.Borders(xlDiagonalDown).LineStyle = xlNone
290    Selection.Borders(xlDiagonalUp).LineStyle = xlNone
291    With Selection.Borders(xlEdgeLeft)
292        .LineStyle = xlContinuous
293        .Weight = xlMedium
294        .ColorIndex = xlAutomatic
295    End With
296    With Selection.Borders(xlEdgeTop)
297        .LineStyle = xlContinuous
```

```
298       .Weight = xlMedium
299       .ColorIndex = xlAutomatic
300     End With
301     With Selection.Borders(xlEdgeBottom)
302       .LineStyle = xlContinuous
303       .Weight = xlMedium
304       .ColorIndex = xlAutomatic
305     End With
306     With Selection.Borders(xlEdgeRight)
307       .LineStyle = xlContinuous
308       .Weight = xlMedium
309       .ColorIndex = xlAutomatic
310     End With
311     Selection.Borders(xlInsideVertical).LineStyle = xlNone
312     Selection.Borders(xlInsideHorizontal).LineStyle = xlNone
313
314     Columns("F:F").Select
315      Selection.columnwidth = 37#
316     Columns("G:G").Select
317      Selection.columnwidth = 14.7
318     Range("F1:G1").Select
319     With Selection.Interior
320       .Pattern = xlSolid
321       .PatternColorIndex = xlAutomatic
322       .ThemeColor = xlThemeColorDark2
323       .TintAndShade = -9.99786370433668E-02
324       .PatternTintAndShade = 0
325     End With
326
327   ' end of formatting site summary table
328
329
330
331   ' format Water-Level Observations Summary Table
332
333     Range("I1:J12").Select
334     Selection.Borders(xlDiagonalDown).LineStyle = xlNone
335     Selection.Borders(xlDiagonalUp).LineStyle = xlNone
336     With Selection.Borders(xlEdgeLeft)
337       .LineStyle = xlContinuous
338       .Weight = xlMedium
339       .ColorIndex = xlAutomatic
340     End With
341     With Selection.Borders(xlEdgeTop)
342       .LineStyle = xlContinuous
343       .Weight = xlMedium
344       .ColorIndex = xlAutomatic
345     End With
346     With Selection.Borders(xlEdgeBottom)
347       .LineStyle = xlContinuous
348       .Weight = xlMedium
```

```
    .ColorIndex = xlAutomatic
End With
With Selection.Borders(xlEdgeRight)
    .LineStyle = xlContinuous
    .Weight = xlMedium
    .ColorIndex = xlAutomatic
End With
Selection.Borders(xlInsideVertical).LineStyle = xlNone
Selection.Borders(xlInsideHorizontal).LineStyle = xlNone

    Columns("I:J").Select
Selection.columnwidth = 15
Range("I1:J1").Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark2
    .TintAndShade = -9.99786370433668E-02
    .PatternTintAndShade = 0
End With

Range("I2:J11").Select
With Selection
    .HorizontalAlignment = xlCenter
    .columnwidth = 17
End With

' end of formatting Water-Level Observations Summary Table

' format Water-Level Observations Time Span Summary Table

Range("L1:U1").Select
With Selection.Interior
    .Pattern = xlSolid
    .PatternColorIndex = xlAutomatic
    .ThemeColor = xlThemeColorDark2
    .TintAndShade = -9.99786370433668E-02
    .PatternTintAndShade = 0
End With

Range("M4:U12").Select
With Selection
    .HorizontalAlignment = xlCenter
End With

Columns("L:L").Select
Selection.columnwidth = 21

Range("Q13:Q13").Select
With Selection
    .HorizontalAlignment = xlCenter
```

```
400         .Font.Italic = True
401     End With
402
403 '-------------------------------------------------------------------
404
405     Range("A1:A1").Select
406
407     Application.StatusBar = False
408
409     Application.ScreenUpdating = True
410
411 endTime = Timer
412 elapsedTimeSeconds = endTime - startTime
413 If elapsedTimeSeconds < 0 Then
414   elapsedTimeSeconds = (86400 - startTime) + endTime
415 End If
416 If elapsedTimeSeconds >= 60 Then
417    elapsedTime = elapsedTimeSeconds / 60
418    units = " minutes"
419 Else
420    elapsedTime = elapsedTimeSeconds
421    units = " seconds"
422 End If
423 MsgBox "observation_stats program ended.  Elapsed time " & Format(expression:=elapsedTime,
Format:="Standard") & units
424
425
426 End Sub
```

create_xyz_worksheet

```
1    Public DataLimitSelection As String, P As Boolean, R As Boolean
2    Public S As Boolean, T As Boolean, I As Boolean
3    Public U As Boolean, makeDataWorksheet As Boolean, limited As String, noLimit As Boolean
4    Public isDepthtoWater As Boolean, isWaterElevation As Boolean
5    Public OutputDataSheet As String, DataWkshtName As String, OutputXYZSheet As String
6    Public ListWkshtName As String, callingSub As String
7    Public startDate As Date, endDate As Date
8
9
10
11   Sub create_xyz_worksheet()
12    Dim dataSheet As String, DataSelection As String
13
14   ' first, get name of worksheet with water-level data
15    lstSelectDataWorksheet.Show
16    dataSheet = DataWkshtName
17
18   ' next, present choices for ways of limiting subset including
```

```
'   - all data in worksheet [A]
'   - just data from a list of site id's [L]
'   - just observations between a start and end date [SE]
'   - just wells with records that span a time period before a start date and after an end date [TS]

 lstSelect_Data_Limit.Show
 DataSelection = DataLimitSelection

' ask user if they want to keep or exclude limit flagged data, and whether or not they want a data worksheet
' with water-level observations

 frmSelectFlags.Show

' create the extension for the worksheet name that will contain the xyz information

 Call getWorksheetExtension

' then select and run the correct sub

 If DataSelection = "A" Then
   Call xyz_From_All_Data
 ElseIf DataSelection = "L" Then
   Call xyz_From_List
 ElseIf DataSelection = "SE" Then
   Call xyz_From_Dates
 ElseIf DataSelection = "TS" Then
   Call xyz_From_POR_Span
 End If


End Sub

'-----------------------------------------------------------------------------------------
Private Sub getWorksheetExtension()

limited = " "
  If Not (P) Then
    limited = limited + "_no_P"
    If Not (R) Then
      limited = limited + "R"
    End If
    If Not (S) Then
      limited = limited + "S"
    End If
    If Not (T) Then
      limited = limited + "T"
    End If
    If Not (I) Then
      limited = limited + "I"
    End If
    If Not (U) Then
```

```
70        limited = limited + "U"
71      End If
72
73    ElseIf Not (R) Then
74      limited = limited + "_no_R"
75      If Not (S) Then
76        limited = limited + "S"
77      End If
78      If Not (T) Then
79        limited = limited + "T"
80      End If
81      If Not (I) Then
82        limited = limited + "I"
83      End If
84      If Not (U) Then
85        limited = limited + "U"
86      End If
87
88    ElseIf Not (S) Then
89      limited = limited + "_no_S"
90      If Not (T) Then
91        limited = limited + "T"
92      End If
93      If Not (I) Then
94        limited = limited + "I"
95      End If
96      If Not (U) Then
97        limited = limited + "U"
98      End If
99
100   ElseIf Not (T) Then
101     limited = limited + "_no_T"
102     If Not (I) Then
103       limited = limited + "I"
104     End If
105     If Not (U) Then
106       limited = limited + "U"
107     End If
108
109   ElseIf Not (I) Then
110     limited = limited + "_no_I"
111     If Not (U) Then
112       limited = limited + "U"
113     End If
114   ElseIf Not (U) Then
115     limited = limited + "_no_U"
116   End If
117
118 If P And R And S And T And I And U Then
119   noLimit = True
120 Else: noLimit = False
```

```
121  End If
122
123  End Sub
124
125  '-----------------------------------------------------------------------------------------
126  Private Sub delete_Flagged_Data(dataSheetToAnalyzeFlags As String)
127   Dim Counter As Long, TestedRow As Long
128
129
130  If Not (noLimit) Then       ' only go through the flagged data if a deletion was selected
131   Worksheets(dataSheetToAnalyzeFlags).Activate
132   TestedRow = 22
133
134   Application.ScreenUpdating = False    ' turns off screen updating
135   Application.DisplayStatusBar = True   ' makes sure that the statusbar is visible
136   Application.StatusBar = "Removing Selected Flagged Data"
137
138   Do While (Cells(TestedRow, 2)) > 0
139      Select Case Cells(TestedRow, 9).Value      'column containing flag
140
141      Case "P"
142         If Not (P) Then       'if the P flag is false, the user wishes to delete pumping data
143           Rows(TestedRow).Delete
144           TestedRow = TestedRow - 1     'if row deleted, will need to test same row number next loop
145         End If
146      Case "R"
147         If Not (R) Then       'if the R flag is false, the user wishes to delete recently-pumped data
148           Rows(TestedRow).Delete
149           TestedRow = TestedRow - 1     'if row deleted, will need to test same row number next loop
150         End If
151      Case "S"
152         If Not (S) Then       'if the S flag is false, the user wishes to delete nearby pumping data
153           Rows(TestedRow).Delete
154           TestedRow = TestedRow - 1     'if row deleted, will need to test same row number next loop
155         End If
156      Case "T"
157         If Not (T) Then       'if the T flag is false, the user wishes to delete nearby recently pumped data
158           Rows(TestedRow).Delete
159           TestedRow = TestedRow - 1     'if row deleted, will need to test same row number next loop
160         End If
161      Case "I"
162         If Not (I) Then       'if the I flag is false, the user wishes to delete injection data
163           Rows(TestedRow).Delete
164           TestedRow = TestedRow - 1     'if row deleted, will need to test same row number next loop
165         End If
166      Case "U"
167         If Not (U) Then       'if the U flag is false, the user wishes to delete nearby injecting data (ADWR
              only)
168           Rows(TestedRow).Delete
169           TestedRow = TestedRow - 1     'if row deleted, will need to test same row number next loop
170         End If
```

```
171       End Select
172
173       TestedRow = TestedRow + 1
174
175    Loop
176
177    Application.StatusBar = False
178    Application.ScreenUpdating = True
179
180   End If
181  End Sub
182
183  '---------------------------------------------------------------------------------------------
184  Private Sub make_xyz()
185
186  Dim dataRange As Range
187  Dim wlMaxDate As Date, wlMinDate As Date
188  Dim wlMinDepth As Double, wlMaxDepth As Double, wlMaxDecline As Double, wlMaxRise As Double
189  Dim wlMinDepth1 As Double, wlMaxDepth1 As Double, wlMinDepth2 As Double, wlMaxDepth2
     As  Double
190  Dim Counter As Long, outputRow As Long, wlMaxRow As Long, wlMinRow As Long
191  Dim siteRowStart As Long, siteRowEnd As Long, outputDataRow As Long
192  Dim wlAverage As Double, wlStDev As Double, totalrows As Double
193  Dim sitenumber As Integer
194  Dim siteID As String
195
196
197  'write column headings to xyz worksheet
198
199   Sheets(OutputXYZSheet).Cells(4, 1).Value = "Longitude"
200   Sheets(OutputXYZSheet).Cells(4, 2).Value = "Latitude"
201   Sheets(OutputXYZSheet).Cells(4, 3).Value = "Site_ID"
202   Sheets(OutputXYZSheet).Cells(4, 4).Value = "Well_Numbr"
203   Sheets(OutputXYZSheet).Cells(4, 5).Value = "Avg_WtrLvl"
204   Sheets(OutputXYZSheet).Cells(4, 6).Value = "StndDev_WL"
205   Sheets(OutputXYZSheet).Cells(4, 7).Value = "n_Obsrvtns"
206   Sheets(OutputXYZSheet).Cells(4, 8).Value = "1stObsDate"
207   Sheets(OutputXYZSheet).Cells(4, 9).Value = "1stWLObs"
208   Sheets(OutputXYZSheet).Cells(4, 10).Value = "LstObsDate"
209   Sheets(OutputXYZSheet).Cells(4, 11).Value = "LstWLObs"
210   Sheets(OutputXYZSheet).Cells(4, 12).Value = "MinDepthDt"
211   Sheets(OutputXYZSheet).Cells(4, 13).Value = "MinDepthOb"
212   Sheets(OutputXYZSheet).Cells(4, 14).Value = "MaxDepthDt"
213   Sheets(OutputXYZSheet).Cells(4, 15).Value = "MaxDepthOb"
214   Sheets(OutputXYZSheet).Cells(4, 16).Value = "WellRecord"
215   Sheets(OutputXYZSheet).Cells(4, 17).Value = "MaxDecline"
216   Sheets(OutputXYZSheet).Cells(4, 18).Value = "MaxRise"
217   Sheets(OutputXYZSheet).Cells(4, 19).Value = "Hole_Depth"
218   Sheets(OutputXYZSheet).Cells(4, 20).Value = "Well_Depth"
219   Sheets(OutputXYZSheet).Cells(4, 21).Value = "LandSurfElv"
220   Sheets(OutputXYZSheet).Cells(4, 22).Value = "Hydrograph"
```

```
221
222   If isDepthtoWater Then
223     Sheets(OutputXYZSheet).Cells(4, 12).Value = “MinDepthDt”
224     Sheets(OutputXYZSheet).Cells(4, 13).Value = “MinDepthOb”
225     Sheets(OutputXYZSheet).Cells(4, 14).Value = “MaxDepthDt”
226     Sheets(OutputXYZSheet).Cells(4, 15).Value = “MaxDepthOb”
227   ElseIf isWaterElevation Then
228     Sheets(OutputXYZSheet).Cells(4, 12).Value = “MinElevnDt”
229     Sheets(OutputXYZSheet).Cells(4, 13).Value = “MinElevnOb”
230     Sheets(OutputXYZSheet).Cells(4, 14).Value = “MaxElevnDt”
231     Sheets(OutputXYZSheet).Cells(4, 15).Value = “MaxElevnOb”
232   End If
233
234  siteRowStart = 22
235  outputRow = 5
236  outputDataRow = 22
237  Worksheets(OutputDataSheet).Activate  ‘do all the work on the subset data worksheet
238
239   Application.ScreenUpdating = False    ‘ turns off screen updating
240   Application.DisplayStatusBar = True   ‘ makes sure that the statusbar is visible
241   Application.StatusBar = “Creating XY Data Worksheet”
242
243
244  ‘Main loop back to here----------------------------------------------------------------------------------
245   Do While (Cells(siteRowStart, 2)) > 0
246   siteID = Cells(siteRowStart, 2)          ‘ save siteID
247   ‘ first, get the range of current site id
248      Counter = siteRowStart
249      Do While (Cells(Counter, 2) = Cells(Counter + 1, 2))
250         Counter = Counter + 1
251         siteRowEnd = Counter
252      Loop
253   ‘This is in case there is only 1 line of data
254      If siteRowEnd <= siteRowStart Then
255         siteRowEnd = siteRowStart
256      End If
257
258
259   ‘ Set the range of data for this site id, then do the math
260
261      Set dataRange = Range(Cells(siteRowStart, 8), Cells(siteRowEnd, 8))
262
263   ‘ Compute average and standard deviation of water level elevation, and count sample size
264      n = WorksheetFunction.Count(dataRange)
265      wlAverage = WorksheetFunction.Average(dataRange)
266      If n > 1 Then
267         wlStDev = WorksheetFunction.StDev(dataRange)
268      Else: wlStDev = -9999
269      End If
270
271   ‘ Find max and min water levels in data range
```

```
272     wlMinDepth = 99999   ‘ initialize max and min holders
273     wlMaxDepth = -99999  ‘
274     wlMinDepth1 = 99999
275     wlMinDepth2 = 99999
276     wlMaxDepth1 = -99999
277     wlMaxDepth2 = -99999
278
279     For Counter = siteRowStart To siteRowEnd
280     Set curCell = Cells(Counter, 8) ‘ row Counter, col 8 (depth to water level below land surface or
            water-level elevation)
281      If (curCell >= wlMaxDepth And IsDate(Cells(Counter, 7))) Then
282       wlMaxDepth = curCell
283       wlMaxDate = Cells(Counter, 7) ‘ set date of maximum water level depth, or maximum water-level
            elevation
284       wlMaxRow = Counter
285      End If
286     Next Counter
287
288     For Counter = siteRowStart To siteRowEnd
289     Set curCell = Cells(Counter, 8) ‘ row Counter, col 8 (depth to water level below land surface or
            water-level elevation)
290      If (curCell <= wlMinDepth And IsDate(Cells(Counter, 7))) Then
291       wlMinDepth = curCell
292       wlMinDate = Cells(Counter, 7) ‘ set date of minimum water level depth, or minimum water-level
            elevation
293       wlMinRow = Counter
294      End If
295     Next Counter
296
297     ‘ find the smallest WL value (minimum depth/elevation) prior to the larget value (maximum
          depth/elevation)
298     For Counter = siteRowStart To wlMaxRow
299      Set curCell = Cells(Counter, 8) ‘ row Counter, col 8 (depth to water level below land surface or
          water-level elevation))
300       If (curCell <= wlMinDepth1 And IsDate(Cells(Counter, 7))) Then
301        wlMinDepth1 = curCell
302        wlMinDate1 = Cells(Counter, 7) ‘ set date of early smallest WL value
303        wlMin1Row = Counter
304       End If
305     Next Counter
306
307     ‘ find the smallest WL value (minimum depth/elevation) after largest value (maximum depth/elevation)
308     For Counter = wlMaxRow To siteRowEnd
309        Set curCell = Cells(Counter, 8) ‘ row Counter, col 8 (depth to water level below land surface or
             water-level elevation))
310         If (curCell <= wlMinDepth2 And IsDate(Cells(Counter, 7))) Then
311           wlMinDepth2 = curCell
312           wlMinDate2 = Cells(Counter, 7) ‘ set date of late smallest WL
313           wlMin2Row = Counter
314         End If
315     Next Counter
```

```
316
317     ‘ find the largest WL value (maximum depth/elevation) prior to smallest (minimum depth/elevation)
318     For Counter = siteRowStart To wlMinRow
319       Set curCell = Cells(Counter, 8) ‘ row Counter, col 8 (depth to water level below land surface or
            water-level elevation))
320       If (curCell >= wlMaxDepth1 And IsDate(Cells(Counter, 7))) Then
321         wlMaxDepth1 = curCell
322         wlMaxDate1 = Cells(Counter, 7) ‘ set date of early largest WL value
323         wlMax1Row = Counter
324       End If
325     Next Counter
326
327     ‘ find the largest WL value (maximum depth/elevation) after smallest (minimum depth/elevation)
328     For Counter = wlMinRow To siteRowEnd
329       Set curCell = Cells(Counter, 8) ‘ row Counter, col 8 (depth to water level below land surface or
            water-level elevation))
330       If (curCell >= wlMaxDepth2 And IsDate(Cells(Counter, 7))) Then
331         wlMaxDepth2 = curCell
332         wlMaxDate2 = Cells(Counter, 7) ‘ set date of late largest WL value
333         wlMax2Row = Counter
334       End If
335     Next Counter
336
337     If wlMaxDepth = wlMinDepth1 Then
338       wlMaxDecline = 0
339     Else
340       wlMaxDecline = wlMaxDepth - wlMinDepth1   ‘this will be maximum rise for water-level elevations
341     End If
342     If wlMaxDepth = wlMinDepth2 Then
343       wlMaxRise = 0
344     Else
345       wlMaxRise = wlMaxDepth - wlMinDepth2      ‘this will be maximum decline for water-level elevations
346     End If
347
348     If (wlMaxDepth1 - wlMinDepth) > wlMaxRise Then
349       wlMaxRise = (wlMaxDepth1 - wlMinDepth)    ‘maximum decline for water-level elevations
350     End If
351     If (wlMaxDepth2 - wlMinDepth) > wlMaxDecline Then
352       wlMaxDecline = (wlMaxDepth2 - wlMinDepth) ‘maximum rise for water-level elevations
353     End If
354
355     ‘ Write summary output info to xyz worksheet
356     Sheets(OutputXYZSheet).Cells(outputRow, 1).Value = Worksheets(OutputDataSheet).
        Cells(siteRowStart, 5).
        Value ‘Longitude
357       Sheets(OutputXYZSheet).Cells(outputRow, 2).Value = Worksheets(OutputDataSheet).
        Cells(siteRowStart, 4).Value ‘Latitude
358     Sheets(OutputXYZSheet).Cells(outputRow, 3).Value = Worksheets(OutputDataSheet).Cells(siteRow
        Value ‘site ID
359     Sheets(OutputXYZSheet).Cells(outputRow, 4).Value = Worksheets(OutputDataSheet).
        Cells(siteRowStart, 3).
```

```
    Value 'well number
360 Sheets(OutputXYZSheet).Cells(outputRow, 5).Value = wlAverage
361 Sheets(OutputXYZSheet).Cells(outputRow, 6).Value = wlStDev
362 Sheets(OutputXYZSheet).Cells(outputRow, 7).Value = n
363 Sheets(OutputXYZSheet).Cells(outputRow, 8).Value = Worksheets(OutputDataSheet).Cells(siteRowStart, 7).
    Value 'first water-level observation date
364 Sheets(OutputXYZSheet).Cells(outputRow, 9).Value = Worksheets(OutputDataSheet).Cells(siteRowStart, 8).
    Value 'first water-level observation
365 Sheets(OutputXYZSheet).Cells(outputRow, 10).Value = Worksheets(OutputDataSheet).Cells(siteRowEnd, 7).
    Value     'last water-level observation date
366 Sheets(OutputXYZSheet).Cells(outputRow, 11).Value = Worksheets(OutputDataSheet).Cells(siteRowEnd, 8).
    Value     'last water-level observation
367 Sheets(OutputXYZSheet).Cells(outputRow, 12).Value = wlMinDate   'date of min depth water-level observa
       tion or min water-level elevation
368 Sheets(OutputXYZSheet).Cells(outputRow, 13).Value = wlMinDepth  'min depth water-level observation or
       min water-level elevation
369 Sheets(OutputXYZSheet).Cells(outputRow, 14).Value = wlMaxDate   'date of max depth water-level
       observation or max water-level elevation
370 Sheets(OutputXYZSheet).Cells(outputRow, 15).Value = wlMaxDepth  'max depth water-level
       observation or max water-level elevation
371 Sheets(OutputXYZSheet).Cells(outputRow, 16).Value = (Worksheets(OutputDataSheet).Cells(siteRowEnd, 7).
       Value - Worksheets(OutputDataSheet).Cells(siteRowStart, 7).Value) + 1 'length of site water-level record
       in days
372 Sheets(OutputXYZSheet).Cells(outputRow, 19).Value = Worksheets(OutputDataSheet).Cells(siteRowStart,
       13). Value   'hole depth
373 Sheets(OutputXYZSheet).Cells(outputRow, 20).Value = Worksheets(OutputDataSheet).Cells(siteRowStart,
       14). Value   'well depth
374 Sheets(OutputXYZSheet).Cells(outputRow, 21).Value = Worksheets(OutputDataSheet).Cells(siteRowStart,
    10). Value   'altitude of land surface
375 Sheets(OutputXYZSheet).Cells(outputRow, 22).Value = "PATH\" & siteID & ".pdf" 'filename for possible
       linked hydrograph
376
377 Sheets(OutputXYZSheet).Cells(outputRow, 16384).Formula = "=TEXT(C" & outputRow & ",
       ""###############"")"    'this formula necessary to make site ids appear as text in ArcMap
       attribute table
378
379 If isDepthtoWater Then
380 Sheets(OutputXYZSheet).Cells(outputRow, 17).Value = wlMaxDecline
381 Sheets(OutputXYZSheet).Cells(outputRow, 18).Value = wlMaxRise
382 ElseIf isWaterElevation Then
383 Sheets(OutputXYZSheet).Cells(outputRow, 17).Value = wlMaxRise
384 Sheets(OutputXYZSheet).Cells(outputRow, 18).Value = wlMaxDecline
385 End If
386
387 outputRow = outputRow + 1          'increment row for output worksheet
388 siteRowStart = siteRowEnd + 1      'increment to next site id on data worksheet
389
390 'Go back to data subset worksheet
391 Worksheets(OutputDataSheet).Activate
392
393 Loop
```

```
394  'Loop back to beginning of main Do While----------------------------------------------------------------------------
395
396  ' Once finished, do some formatting on xyz worksheet
397
398  Worksheets(OutputXYZSheet).Activate
399
400  Range("A4:B" & outputRow + 2).NumberFormat = "0.000000"  'lat/long
401  Range("E4:F" & outputRow + 2).NumberFormat = "0.0"  'water levels
402
403  Range("XFD5:XFD" & outputRow).Copy
404  Range("C5:C5").PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _
405     :=False, Transpose:=False
406  Columns("XFD:XFD").Delete Shift:=xlToLeft
407
408  Sheets(OutputXYZSheet).Cells.EntireColumn.AutoFit
409  Columns("A:A").columnwidth = 20
410
411
412  Application.StatusBar = False
413  Application.ScreenUpdating = True
414
415
416           End Sub
417
418  '-------------------------------------------------------------------------------------------
419  Private Sub xyz_From_All_Data()
420
421  Dim dataSheet As String
422  Dim startTime As Long, endTime As Long, elapsedTimeSeconds As Long, elapsedTime As Variant
423  Dim units As String424
425  dataSheet = DataWkshtName
426
427  ' Create worksheet to hold xyz information and name it
428
429  OutputXYZSheet = "xyz_ALL" + limited
430  Set NewSheet1 = Worksheets.Add
431  NewSheet1.Name = OutputXYZSheet
432  Sheets(OutputXYZSheet).Move After:=Sheets(Worksheets.Count)
433
434  startTime = Timer
435
436  Sheets(OutputXYZSheet).Cells(1, 1).Value = "Summary of ALL SITES on data worksheet"
437  If limited > " " Then
438    Sheets(OutputXYZSheet).Cells(2, 1).Value = limited + " FLAGGED OBSERVATIONS"
439  End If
440
441  Range("A1:D2").Interior.Color = 65535
442
443  OutputDataSheet = "data_ALL" + limited
444
445 '  Make a copy of original data worksheet
```

```
446
447   Sheets(dataSheet).Copy After:=Sheets(Worksheets.Count)
448   Sheets(Worksheets.Count).Name = OutputDataSheet
449
450   ‘ call sub that sorts through subset data worksheet, deleting appropriate flagged data
451   Call delete_Flagged_Data(OutputDataSheet)
452
453   ‘ call sub that creates xyz worksheet from subset data worksheet
454   Call make_xyz
455
456   Range(“A1”).Select
457
458   endTime = Timer
459   elapsedTimeSeconds = endTime - startTime
460   If elapsedTimeSeconds < 0 Then
461  elapsedTimeSeconds = (86400 - startTime) + endTime
462     End If
463   If elapsedTimeSeconds >= 60 Then
464   elapsedTime = elapsedTimeSeconds / 60
465   units = “ minutes”
466   Else
467   elapsedTime = elapsedTimeSeconds
468   units = “ seconds”
469   End If
470   MsgBox “xyz_From_All_Data program ended.  Elapsed time “ & Format(expression:=elapsedTime,
      Format:=”Standard”) & units
471
472   End Sub
473
474   ‘----------------------------------------------------------------------------------------
475   Private Sub xyz_From_List()
476
477   Dim dataSheet As String, listDataSheet As String
478   Dim SiteIDRow As Long, outputRow As Long, AllDataRow As Long, CopyRow As Long
479   Dim SiteIDFind As String, SiteFound As Boolean, allRowStart As Long, allRowEnd As Long
480   Dim copyRange As String
481   Dim startTime As Long, endTime As Long, elapsedTimeSeconds As Long, elapsedTime As Variant
482   Dim units As String
483
484   ‘  DATA WORKSHEET MUST FIRST BE SORTED BY SITE ID AND BY DATE
485   ‘  SITE LIST MUST ALSO BE SORTED BY SITE ID, WITH SITES IDS IN FIRST COLUMN
      BEGINNING IN SECOND ROW
486
487   lstSelectListWorksheet.Show
488   listDataSheet = ListWkshtName
489
490   dataSheet = DataWkshtName
491
492   ‘ open and name the xyz output worksheet
493
494   OutputXYZSheet = “xyz_LIST” + limited
```

```
    Set NewSheet1 = Worksheets.Add
    NewSheet1.Name = OutputXYZSheet
    Sheets(OutputXYZSheet).Move After:=Sheets(Worksheets.Count)

    startTime = Timer

    Sheets(OutputXYZSheet).Cells(1, 1).Value = "Summary of LIST OF SITES on data worksheet"
    If limited > " " Then
      Sheets(OutputXYZSheet).Cells(2, 1).Value = limited + " FLAGGED OBSERVATIONS"
    End If

    Range("A1:D2").Interior.Color = 65535

'   Open working copy of water-level data
    Sheets(dataSheet).Copy After:=Sheets(Worksheets.Count)
    Sheets(Worksheets.Count).Name = "tempDataSheet"

    Call delete_Flagged_Data("tempDataSheet")

'    Open subset data worksheet and copy header info

    OutputDataSheet = "data_LIST" + limited
    Set NewSheet1 = Worksheets.Add
    NewSheet1.Name = OutputDataSheet
    Sheets(OutputDataSheet).Move After:=Sheets(Worksheets.Count)

     Sheets(dataSheet).Select
     Rows("1:21").Select
     Selection.Copy
     Sheets(OutputDataSheet).Select
     ActiveSheet.Paste

    SiteIDRow = 2       'initialize counters
    AllDataRow = 22
    outputRow = 22

    Application.ScreenUpdating = False    ' turns off screen updating
    Application.DisplayStatusBar = True   ' makes sure that the statusbar is visible
    Application.StatusBar = "Copying Data for Selected Wells"

    ' Read in site ID from list--------------------------------------------------------------------------
    Do While (Sheets(listDataSheet).Cells(SiteIDRow, 1)) > 0
      SiteIDFind = Sheets(listDataSheet).Cells(SiteIDRow, 1).Value
      SiteFound = False

    ' Find data with this site ID, copy it to new data worksheet

    'first, find where siteID begins
    Worksheets("tempDataSheet").Activate
    Counter = AllDataRow            'because data is sorted by site id, only need to start looking where last search
stopped
```

```
545
        Do While (Cells(Counter, 2)) > 0
546     If Cells(Counter, 2) = SiteIDFind Then
547       allRowStart = Counter
548       SiteFound = True
549       Exit Do
550       End If
551       Counter = Counter + 1
552       Loop
553
554   If (Not SiteFound) Then            'if no data is found for site ID, reset starting point for search
555     Worksheets(listDataSheet).Cells(SiteIDRow, 1).Interior.ColorIndex = 36          'highlight site ID not
        found
556     Worksheets(listDataSheet).Cells(SiteIDRow, 2).Value = "site not found in dataset"
557     GoTo 333                         'and go to next site ID in list
558   End If
559
560   'then, find where this siteID ends
561   allRowEnd = Counter ' in case there is only 1 row of data
562   Do While (Cells(Counter, 2) = Cells(Counter + 1, 2))
563      Counter = Counter + 1
564      allRowEnd = Counter
565   Loop
566
567   'copy these rows over to new worksheet
568        copyRange = allRowStart & ":" & allRowEnd
569        Rows(copyRange).Select
570        Selection.Copy
571        Sheets(OutputDataSheet).Select
572        Rows(outputRow).Select
573        ActiveSheet.Paste
574
575        outputRow = outputRow + (allRowEnd - allRowStart + 1)     'location on new worksheet for next paste
576        AllDataRow = allRowEnd + 1                          'start location of search for next site ID
577
578   333       SiteIDRow = SiteIDRow + 1                          'location of next site ID
579     Loop    'back to read in next site ID ----------------------------------------------------------------------
580
581
582  ' call sub that creates XYZ worksheet and compiles attribute table information
583   Call make_xyz
584
585   Range("A1").Select
586
587  ' delete temporary data sheet
588   Application.DisplayAlerts = False
589   Sheets("tempDataSheet").Delete
590   Application.DisplayAlerts = True
591
592   endTime = Timer
593   elapsedTimeSeconds = endTime - startTime
```

```
If elapsedTimeSeconds < 0 Then
elapsedTimeSeconds = (86400 - startTime) + endTime
End If
If elapsedTimeSeconds >= 60 Then
elapsedTime = elapsedTimeSeconds / 60
units = " minutes"
Else
elapsedTime = elapsedTimeSeconds
units = " seconds"
End If
MsgBox "xyz_From_List program ended.  Elapsed time " & Format(expression:=elapsedTime, Format:="Standard") & units

End Sub

'-----------------------------------------------------------------------------------------
Private Sub xyz_From_Dates()
Dim dataSheet As String, Counter As Long
Dim startTime As Long, endTime As Long, elapsedTimeSeconds As Long, elapsedTime As Variant
Dim units As String


 dataSheet = DataWkshtName

' Select range of water levels based on input start, and end dates
   callingSub = "xyz_From_Dates"
   lstSelectDates.Show


' Create worksheet to hold xyz information and name it

 OutputXYZSheet = "xyz_DATES" + limited
 Set NewSheet1 = Worksheets.Add
 NewSheet1.Name = OutputXYZSheet
 Sheets(OutputXYZSheet).Move After:=Sheets(Worksheets.Count)
startTime = Timer

 Sheets(OutputXYZSheet).Cells(1, 1).Value = "Summary of OBSERVATIONS BETWEEN " & startDate & " AND " & endDate
 If limited > " " Then
   Sheets(OutputXYZSheet).Cells(2, 1).Value = limited + " FLAGGED OBSERVATIONS"
 End If

 Range("A1:D2").Interior.Color = 65535

 OutputDataSheet = "data_DATES" + limited

'  Make a copy of original data worksheet

 Sheets(dataSheet).Copy After:=Sheets(Worksheets.Count)
 Sheets(Worksheets.Count).Name = OutputDataSheet
```

```
643
644  ‘   Delete all rows of observations that fall outside of the specified user-input range
645
646  Application.ScreenUpdating = False    ‘ turns off screen updating
647  Application.DisplayStatusBar = True   ‘ makes sure that the statusbar is visible
648  Application.StatusBar = “Copying Data for Selected Date Range”
649
650   Worksheets(OutputDataSheet).Activate
651   Counter = 22
652   Do While (Cells(Counter, 2)) > 0
653     If Cells(Counter, 7) < startDate Or Cells(Counter, 7) > endDate Then
654       Rows(Counter).Delete
655       Counter = Counter - 1     ‘if row deleted, will need to test same row number next loop
656     End If
657     Counter = Counter + 1
658   Loop
659
660
661  ‘ call sub that sorts through subset data worksheet, deleting appropriate flagged data
662  Call delete_Flagged_Data(OutputDataSheet)
663
664  ‘ call sub that creates xyz worksheet from subset data worksheet
665  Call make_xyz
666
667   Range(“A1”).Select
668
669  endTime = Timer
670  elapsedTimeSeconds = endTime - startTime
671  If elapsedTimeSeconds < 0 Then
672  elapsedTimeSeconds = (86400 - startTime) + endTime
673  End If
674   If elapsedTimeSeconds >= 60 Then
675  elapsedTime = elapsedTimeSeconds / 60
676  units = “ minutes”
677  Else
678  elapsedTime = elapsedTimeSeconds
679  units = “ seconds”
680  End If
681  MsgBox “xyz_From_Dates program ended.  Elapsed time “ & Format(expression:=elapsedTime,
     Format:=”Standard”) & units
682
683  End Sub
684
685  ‘----------------------------------------------------------------------------------------
686  Private Sub xyz_From_POR_Span()
687  Dim recordStartDate As Date, recordEndDate As Date
688  Dim allRow As Long, siteRowStart As Long, siteRowEnd As Long, dataSheet As String
689  Dim startTime As Long, endTime As Long, elapsedTimeSeconds As Long, elapsedTime As Variant
690  Dim units As String
691
692  ‘ Get user-specified cutoff dates to define desired time span
```

```
693   callingSub = “xyz_From_POR_Span”
694   lstSelectDates.Show
695
696
697   dataSheet = DataWkshtName
698
699   ‘ Create worksheet to hold xyz information and name it
700
701   OutputXYZSheet = “xyz_SPAN” + limited
702   Set NewSheet1 = Worksheets.Add
703   NewSheet1.Name = OutputXYZSheet
704   Sheets(OutputXYZSheet).Move After:=Sheets(Worksheets.Count)
705
706   Sheets(OutputXYZSheet).Cells(1, 1).Value = “Summary of SITES WITH RECORDS BEGINNING BEFORE “
      & startDate & “ AND ENDING AFTER “ & endDate
707   If limited > “ “ Then
708   Sheets(OutputXYZSheet).Cells(2, 1).Value = limited + “ FLAGGED OBSERVATIONS”
709   End If
710
711   Range(“A1:G2”).Interior.Color = 65535
712
713   OutputDataSheet = “data_SPAN” + limited
714
715 ‘  Make a copy of original data worksheet
716
717   Sheets(dataSheet).Copy After:=Sheets(Worksheets.Count)
718   Sheets(Worksheets.Count).Name = OutputDataSheet
719   startTime = Timer
720
721   ‘ Call sub that sorts through subset data worksheet, deleting appropriate flagged data
722   Call delete_Flagged_Data(OutputDataSheet)
723
724
725   ‘ do all work on copy of data worksheet
726   Worksheets(OutputDataSheet).Activate
727
728   ‘ initialize counters
729   allRow = 22
730
731   Application.ScreenUpdating = False    ‘ turns off screen updating
732   Application.DisplayStatusBar = True   ‘ makes sure that the statusbar is visible
733   Application.StatusBar = “Copying Data for Wells with Specified Period of Record”
734
735   ‘  Loop through all site ids on copy of data worksheet
736   Do While (Cells(allRow, 2) > 0)
737
738   ‘ find first and last observation dates for site id
739   Counter = allRow
740   siteRowStart = allRow
741   siteRowEnd = siteRowStart
742   recordStartDate = Cells(siteRowStart, 7)
```

```
743       Do While (Cells(Counter, 2) = Cells(Counter + 1, 2))
744         Counter = Counter + 1
745         siteRowEnd = Counter
746       Loop
747       recordEndDate = Cells(siteRowEnd, 7)
748
749       ‘ if start of record is after user-specified start date or end of record is before user-specified end date,
        delete records for site id
750       If (recordStartDate > startDate) Or (recodEndDate < endDate) Then
751        Rows(siteRowStart & “:” & siteRowEnd).Delete
752        allRow = siteRowStart   ‘if site id data was deleted, will need to start search over at this row
753     Else
754        allRow = siteRowEnd + 1 ‘if site id data kept, go on to next site id
755     End If
756
757     Loop  ‘go back to next site id
758
759
760   ‘ Call sub that creates XYZ worksheet and compiles attribute table information
761       Call make_xyz
762
763       Range(“A1”).Select
764
765     endTime = Timer
766     elapsedTimeSeconds = endTime - startTime
767     If elapsedTimeSeconds < 0 Then
768       elapsedTimeSeconds = (86400 - startTime) + endTime
769     End If
770     If elapsedTimeSeconds >= 60 Then
771       elapsedTime = elapsedTimeSeconds / 60
772       units = “minutes”
773     Else
774       elapsedTime = elapsedTimeSeconds
775       units = “seconds”
776     End If
777
778     MsgBox “xyz_From_POR_Span program ended.  Elapsed time “ & Format(expression:=elapsedTime,
        Format:=”Standard”) & units
779
780     End Sub
```

makeExcelHydrographs

```
1   Public DataWkshtName As String, DataWkbkName As String, MinDateAxis As Date, MaxDateAxis As Date
2   Public MinDepthAxis As Long, MaxDepthAxis As Long, isMinDepth As Boolean, isMaxDepth As Boolean
3   Public isMinDate As Boolean, isMaxDate As Boolean, isDepthtoWater As Boolean, isWaterElevation As
      Boolean
4
5
6   Sub makeExcelHydrographs()
```

```
7
8   'This VBA program creates hydrograph charts from data supplied in a data worksheet.
9   'These hydrographs can be hyperlinked to well location in ArcGIS.
10
11  Dim Counter As Long, allRowStart As Long, allRowEnd As Long, nonUSGSRow As Integer
12  Dim added As Integer, RowCount As Integer, nObs As Integer, minObs As Integer, wkbookNum As Integer
13  Dim totalCharts As Integer
14  Dim SiteID As String, wellInfo As String, chartName As String, wellNameText As String
15  Dim wellLong As String, wellLat As String, wellAlt As String, wellDepth As String
16  Dim xRangeString As String, yRangeString As String
17  Dim xRangeAll As Range, yRangeAll As Range
18  Dim xRangeLegend As Range, yRangeLegend As Range
19  Dim allDataSheet As String
20  Dim xLegend As String, yLegend As String
21  Dim NewSeriesxRange(255) As String, NewSeriesyRange(255) As String
22  Dim startDate As String, endDate As String, currentWorkbookName As String, SolidLineData As  String
23  Dim minLate As Double, maxLate As Double
24  Dim dateStart As Date, dateEnd As Date
25  Dim BasinName As String
26  Dim startTime As Long, endTime As Long,  elapsedTimeSeconds As Long
27  Dim elapsedTime As Variant, units As String
28
29
30  DataWkbkName = ActiveWorkbook.Name
31
32    lstSelectWorksheet.Show            'get worksheet with data from user
33    lstSelectAxes.Show                 'get information on how water-level data is provided (depth or altitude)
    and how to handle axes
34    allDataSheet = DataWkshtName
35
36    startTime = Timer
37
38    currentWorkbookName = ActiveWorkbook.Name
39    workbookName = ActiveWorkbook.Name
40    dataWorkbookName = ActiveWorkbook.Name
41    RowCount = 2
42    wkbookNum = 1
43    totalCharts = 0
44
45    SolidLineData = "SolidLineData" & wkbookNum
46
47    ' open a new worksheet to contain <= 1-year observation data for plotting
48    Set NewSheet1 = Worksheets.Add
49    NewSheet1.Name = SolidLineData
50    Sheets(NewSheet1.Name).Move After:=Sheets(Worksheets.Count)
51    Worksheets(SolidLineData).Cells(1, 1).
      Value = "Date"
52    Worksheets(SolidLineData).Cells(1, 2).
      Value = "WL_Depth/Altitude"
53    Worksheets(SolidLineData).Cells(2, 1).Value = 33556        'dummy data
54    Worksheets(SolidLineData).Cells(2, 2).Value = 100
```

```
55
56    allRowStart = 22          ‘22
57    localRowCounter = 2                                    ‘this is the row on the Solidline data worksheet
58
59    Worksheets(allDataSheet).Activate
60
61    ‘-------------------------------------------------------------------------------------------------’
62    ‘Loop back to here after creating each chart for each site ID
63    ‘-----------------------------------
64    ‘-----------------------------------
65    Do While (Workbooks(dataWorkbookName).Worksheets(allDataSheet).Cells(allRowStart, 2)) > 0
66    ‘-----------------------------------
67    ‘-----------------------------------
68
69    Workbooks(dataWorkbookName).Worksheets(allDataSheet).Activate
70
71    ‘ first, get the range of current site id
72    Counter = allRowStart
73    allRowEnd = allRowStart
74    Do While (Cells(Counter, 2) = Cells(Counter + 1, 2))
75       Counter = Counter + 1
76       allRowEnd = Counter
77    Loop
78
79    ‘set range in between
80       xRangeString = “G” & allRowStart & “:G” & allRowEnd    ‘date of water measurement
81       yRangeString = “H” & allRowStart & “:H” & allRowEnd    ‘depth to water below land surface
82    Set xRangeAll=Workbooks(dataWorkbookName).Worksheets(allDataSheet).Range(xRangeString)
      ‘date of water measurement
83    Set yRangeAll =Workbooks(dataWorkbookName).Worksheets(allDataSheet).Range(yRangeString)
      ‘depth to water below land surface
84
85    ‘This is only for the legend
86    xLegend = “G” & allRowStart
87    yLegend = “H” & allRowStart
88    Set xRangeLegend = Workbooks(dataWorkbookName).Worksheets(allDataSheet).Range(xLegend)
89    Set yRangeLegend Workbooks(dataWorkbook Name).Worksheets(allDataSheet).Range(yLegend)
90
91    ‘save information for hydrograph header
92    SiteID = Cells(allRowStart, 2)          ‘ save siteID
93    wellNameText = Cells(allRowStart, 3)    ‘ save well name
94      If IsEmpty(Cells(allRowStart, 3)) Then
95       wellNameText = “unknown”
96      End If
97    wellLong = Cells(allRowStart, 5)        ‘ save well longitude
98    wellLat = Cells(allRowStart, 4)         ‘ save well latitude
99    wellAlt = Round(Cells(allRowStart, 10), 0)        ‘ save well altitude
100    If IsEmpty(Cells(allRowStart, 10)) Or wellAlt = “-9999” Then       ‘ in case data is missing or labeled -9999
101      wellAlt = “unknown”
102   End If
103   wellDepth = Cells(allRowStart, 14)      ‘ save well depth
```

```
104   If IsEmpty(Cells(allRowStart, 14)) Or wellDepth = "-9999" Then     ' in case data is missing or labeled -9999
105     wellDepth = "unknown"
106   End If
107
108  'get the minimum and max depths and dates for chart axes
109   minYear = Year(Workbooks(dataWorkbookName).Worksheets(allDataSheet).Cells(allRowStart, 7).Value)
110   maxYear = Year(Workbooks(dataWorkbookName).Worksheets(allDataSheet).Cells(allRowEnd, 7).Value)
111
112  minDepth = 9999
113  maxDepth = -9999
114  For Counter = allRowStart To allRowEnd
115    If Cells(Counter, 8) < minDepth Then
116      minDepth = Cells(Counter, 8)
117    End If
118    If Cells(Counter, 8) > maxDepth Then
119      maxDepth = Cells(Counter, 8)
120    End If
121  Next Counter
122
123  'find observations in range that are at most 1 year apart
124  nNewSeries = 0
125  Counter = allRowStart
126  Do While Counter <= allRowEnd
127     If Counter = allRowEnd Then               'Can't compare counter+1
128    GoTo 100
129   End If
130
131   If Cells(Counter + 1, 7) - Cells(Counter, 7) <= 365 Then    'if dates are within 1 year
132          nNewSeries = nNewSeries + 1                  'count number of blocks of data for later
133    localRowCounter = localRowCounter + 1
134    NewSeriesRowStart = localRowCounter
135    Workbooks(currentWorkbookName).Worksheets(SolidLineData).Cells(localRowCounter, 1).Value =
Workbooks(dataWorkbookName).Worksheets(allDataSheet).Cells(Counter, 7).Value    'date
136    Workbooks(currentWorkbookName).Worksheets(SolidLineData).Cells(localRowCounter, 2).Value =
Workbooks(dataWorkbookName).Worksheets(allDataSheet).Cells(Counter, 8).Value    'observation
137
138    For NewCounter = Counter + 1 To allRowEnd       'check subsequent sequential observation for end of series
139      If Cells(NewCounter + 1, 7) - Cells(NewCounter, 7) > 365 Then  'found end, so copy data, update counter
 and exit this for
140       localRowCounter = localRowCounter + 1
141       Workbooks(currentWorkbookName).Worksheets(SolidLineData).Cells(localRowCounter, 1).Value =
Workbooks(dataWorkbookName).Worksheets(allDataSheet).Cells(NewCounter, 7).Value    'date
142    Workbooks(currentWorkbookName).Worksheets(SolidLineData).Cells(localRowCounter, 2).Value =
Workbooks(dataWorkbookName).Worksheets(allDataSheet).Cells(NewCounter, 8).Value    'observation
143       NewSeriesxRange(nNewSeries) = "A" & NewSeriesRowStart & ":A" & localRow Counter 'date of
water measurement
144       NewSeriesyRange(nNewSeries) = "B"& NewSeriesRowStart & ":B" & localRowCounter 'depth to water
below land surface
145    Counter = NewCounter
146    Exit For
147    Else
```

```
148      localRowCounter = localRowCounter + 1
149      Workbooks(currentWorkbookName).Worksheets(SolidLineData).Cells(localRowCounter, 1).Value =
    Workbooks(dataWorkbookName).Worksheets(allDataSheet).Cells(NewCounter, 7).Value    'date
150      Workbooks(currentWorkbookName).Worksheets(SolidLineData).Cells(localRowCounter, 2).Value =
    Workbooks(dataWorkbookName).Worksheets(allDataSheet).Cells(NewCounter, 8).Value    'observation
151      If NewCounter >= allRowEnd Then              'Can't compare counter+1
152      NewSeriesxRange(nNewSeries) = "A" & NewSeriesRowStart & ":A" & localRowCounter   'date of
         water measurement
153      NewSeriesyRange(nNewSeries) = "B" & NewSeriesRowStart & ":B" & localRowCounter  'depth to water
    below land surface
154            Counter = NewCounter
155            GoTo 100
156           End If
157          Counter = NewCounter
158         End If
159       Next NewCounter
160     Else
161     End If
162   100    Counter = Counter + 1
163     Loop
164
165
166 ' ----------------------add chart ----------------------
167   Workbooks(currentWorkbookName).Worksheets(SolidLineData).Activate    'activate new workbook, if necessary
168    totalCharts = totalCharts + 1
169    chartName = SiteID
170
171
172   ActiveSheet.Shapes.AddChart.Select
173   ActiveChart.Location Where:=xlLocationAsNewSheet
174    ActiveChart.Name = chartName                    ' make worksheet name the site id
175   ActiveChart.Move After:=Sheets(Worksheets.Count + Charts.Count)
176
177    ActiveChart.ChartType = xlXYScatter
178    ActiveChart.SeriesCollection(1).Delete
179
180   'first put in dummy data point in order to make a single-point legend
181     ActiveChart.SeriesCollection.NewSeries
182    ActiveChart.SeriesCollection(1).XValues = xRangeLegend
183    ActiveChart.SeriesCollection(1).Values =  yRangeLegend
184    ActiveChart.SeriesCollection(1).Name = "Water-Level Observation"
185
186   'next put in all data with dashed line
187    ActiveChart.SeriesCollection.NewSeries
188    ActiveChart.SeriesCollection(2).XValues = xRangeAll
189    ActiveChart.SeriesCollection(2).Values =  yRangeAll
190    ActiveChart.SeriesCollection(2).Name = "Observations > 1 Year Apart"
191
192   Application.ScreenUpdating = False
193   Application.DisplayStatusBar = True    ' makes sure that the statusbar is visible
194   Application.StatusBar = "Creating Hydrograph Chart"
```

```
195
196  'last, put in series of observations that need solid line
197     If nNewSeries >= 0 Then
198     For i = 1 To nNewSeries
199    ActiveChart.SeriesCollection.NewSeries
200    ActiveChart.SeriesCollection(i+2).XValues = Workbooks(currentWorkbookNameWorksheets(SolidLineData).
     Range(NewSeriesxRange(i))
201    ActiveChart.SeriesCollection(i + 2).Values = Workbooks(currentWorkbookNameWorksheets(SolidLineData).
     Range(NewSeriesyRange(i))
202    ActiveChart.SeriesCollection(i + 2).Name = "delete" & nNewSeries
203    Next i
204   End If
205
206 ' insert a legend at the bottom of the chart
207   ActiveChart.HasLegend = True
208   ActiveChart.Legend.Select
209   Selection.Position = xlBottom
210
211 ' delete extra entries in legend
212   For i = 3 To ActiveChart.SeriesCollection.Count
213    ActiveChart.Legend.LegendEntries(3).Select    'the third legend entry will always be the one to delete
214    Selection.Delete
215   Next i
216
217
218
219 '   ----------FORMAT AXES----------'
220
221    'Chart Title and Axes Titles
222       With ActiveChart
223      .Axes(xlCategory, xlPrimary).HasTitle = True
224      .Axes(xlCategory, xlPrimary).AxisTitle.Characters.Text = "Date of Observation"
225      .Axes(xlValue, xlPrimary).HasTitle = True
226      If isDepthtoWater Then
227      .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Depth to Groundwater Below LSD [feet]"
228      ElseIf isWaterElevation Then
229      .Axes(xlValue, xlPrimary).AxisTitle.Characters.Text = "Water-Level Altitude [feet AMSL]"
230      End If
231   End With
232
233  'shrink plot area to allow room for well information in header
234   ActiveChart.PlotArea.Select
235   Selection.Height = 390      '368
236   Selection.Top = 40         '64
237
238  ' sets minimums and maximum for date and depth axes
239
240   ActiveChart.Axes(xlCategory).Select     'work on date axis
241   With ActiveChart.Axes(xlCategory)
242
243    If isMinDate Then
```

```
244         .MinimumScale = MinDateAxis
245       Else
246         minYear = "01/01/" & minYear
247         minYear = DateValue(minYear)
248         .MinimumScale = minYear
249       End If
250       If isMaxDate Then
251         .MaximumScale = MaxDateAxis
252       Else
253         maxYear = "01/01/" & (maxYear + 1)
254         maxYear = DateValue(maxYear)
255         .MaximumScale = maxYear
256       End If
257         .MajorUnitIsAuto = True
258
259     .MinorUnitIsAuto = True
260     .Crosses = xlAutomatic
261     .ReversePlotOrder = False
262     .ScaleType = xlLinear
263     .DisplayUnit = xlNone
264   End With
265
266 ' 10 point labels and formatted date
267     Selection.TickLabels.AutoScaleFont = False
268     With Selection.TickLabels.Font
269         .Name = "Univers 57 Condensed"  '"Arial"
270         .FontStyle = "Regular"
271         .Size = 10
272         .ColorIndex = xlAutomatic
273         .Background = xlAutomatic
274     End With
275     Selection.TickLabels.NumberFormat = "[$-409]mmm-yy;@"
276
277     With ActiveChart.Axes(xlValue)
278     ActiveChart.Axes(xlValue).Select    'work on depth axis
279
280       If isMinDepth Then
281         .MinimumScale = MinDepthAxis
282       Else
283         .MinimumScale = Application.WorksheetFunction.RoundDown(minDepth, -1)
284       End If
285       If isMaxDepth Then
286         .MaximumScale = MaxDepthAxis
287       Else
288         .MaximumScale = Application.WorksheetFunction.RoundUp(maxDepth, -1)
289       End If
290         .MajorUnitIsAuto = True
291
292     If isDepthtoWater Then
293
294     .MinorUnitIsAuto = True     'if the data is in depth to water, then reverse the order of the depth axis and have
```

```
    date axis cross at maximum
295   .Crosses = xlMaximum
296   .ReversePlotOrder = True
297   .ScaleType = xlLinear
298   .DisplayUnit = xlNone
299
300   ElseIf isWaterElevation Then
301
302   .MinorUnitIsAuto = True     'if the data is water altitude, keep order and have date axis cross automatically
303   .Crosses = xlAutomatic
304   .ReversePlotOrder = False
305   .ScaleType = xlLinear
306   .DisplayUnit = xlNone
307
308   End If
309
310
311   End With
312
313
314
315 ' 10 point labels and formatted depth
316   Selection.TickLabels.AutoScaleFont = False
317   With Selection.TickLabels.Font
318       .Name = "Univers 57 Condensed"  '"Arial"
319       .FontStyle = "Regular"
320       .Size = 10
321       .ColorIndex = xlAutomatic
322       .Background = xlAutomatic
323   End With
324   Selection.TickLabels.NumberFormat = "0"
325
326
327 '12 point axes titles
328   ActiveChart.Axes(xlValue).AxisTitle.Select
329   Selection.AutoScaleFont = True
330   With Selection.Font
331       .Name = "Univers 57 Condensed"  '"Arial"
332       .Size = 12
333       .ColorIndex = xlAutomatic
334       .Background = xlAutomatic
335   End With
336   ActiveChart.Axes(xlCategory).AxisTitle.Select
337   Selection.AutoScaleFont = True
338   With Selection.Font
339       .Name = "Univers 57 Condensed"  '"Arial"
340       .Size = 12
341       .ColorIndex = xlAutomatic
342       .Background = xlAutomatic
343   End With
344
```

```
345
346 '---------------------------- GRIDLINES --------------------------------------------------'
347 'sets gridlines of chart
348    With ActiveChart.Axes(xlCategory)
349       .HasMajorGridlines = True
350       .HasMinorGridlines = False
351    End With
352    With ActiveChart.Axes(xlValue)
353       .HasMajorGridlines = True
354       .HasMinorGridlines = False
355    End With
356
357 'make gridlines gray
358    ActiveChart.Axes(xlCategory).MajorGridlines.Select
359    With Selection.Border
360       .Color = RGB(192, 192, 192)
361 '      .ColorIndex = 15        '25% gray   24=less gray
362       .Weight = xlHairline
363       .LineStyle = xlContinuous
364    End With
365    ActiveChart.Axes(xlValue).MajorGridlines.Select
366    With Selection.Border
367       .Color = RGB(192, 192, 192)
368 '       .ColorIndex = 15    '25% gray       24=less gray
369       .Weight = xlHairline
370       .LineStyle = xlContinuous
371    End With
372
373 ' add border around chart
374    ActiveChart.ChartArea.Select
375    With Selection.Border
376       .ColorIndex = 1
377       .Weight = xlThin
378       .LineStyle = xlContinuous
379    End With
380
381
382 '---------------------------------------FORMAT DATA SERIES----------------------------------'
383 'sets properties of single-point data series (for legend)
384    ActiveChart.SeriesCollection(1).Select
385    With Selection
386       .MarkerBackgroundColorIndex = 11    ' markers are size 5 blue diamonds
387       .MarkerForegroundColorIndex = 11
388       .MarkerStyle = xlDiamond
389       .Smooth = False
390       .MarkerSize = 5
391       .Shadow = False
392    End With
393    With Selection.Border
394       .LineStyle = xlNone
395    End With
```

```
396
397  'sets properties of ALL-data series (make all data dashed)
398   ActiveChart.SeriesCollection(2).Select
399   With Selection.Border                ' ALL data has a dashed line between points
400      .ColorIndex = 11
401      .Weight = xlThin
402      .LineStyle = xlDot
403   End With
404
405   With Selection
406      .MarkerBackgroundColorIndex = 11    ' markers are size 5 blue diamonds
407      .MarkerForegroundColorIndex = 11
408      .MarkerStyle = xlDiamond
409      .Smooth = False
410      .MarkerSize = 5
411      .Shadow = False
412   End With
413
414  'sets properties of observations less than 1 year apart (solid line)
415   If nNewSeries > 0 Then
416   For i = 3 To (nNewSeries + 2)
417   ActiveChart.SeriesCollection(i).Select
418   With Selection.Border                ' observations < 1 year apart have a solid line over the dashed line
419      .ColorIndex = 11
420      .Weight = xlThin    'xlMedium for thicker line
421      .LineStyle = xlContinuous
422   End With
423
424   With Selection
425      .MarkerBackgroundColorIndex = 11    ' markers are size 5 blue diamonds
426      .MarkerForegroundColorIndex = 11
427      .MarkerStyle = xlDiamond
428      .Smooth = False
429      .MarkerSize = 5
430      .Shadow = False
431   End With
432   Next i
433   End If
434
435
436 '-------------------------------------------------------------------------------------------------'
437
438 '----------------------------ADD TEXT BOXES TO CHART-----------------------------------------------'
439
440  'add text box with site ID
441
442      ActiveChart.ChartArea.Select
443     ActiveChart.Shapes.AddLabel(msoTextOrientationHorizontal, 90#, 5, 150, 10).Select    '(Orientation, Left, Top,
     Width, Height)
444      Selection.ShapeRange(1).TextFrame.AutoSize = msoTrue
445      Selection.Characters.Text = "Site ID: " & SiteID
```

```
446   Selection.AutoScaleFont = False
447   With Selection.Characters.Font
448       .Name = "Univers 57 Condensed"  '"Arial"
449       .Size = 9
450       .ColorIndex = xlAutomatic
451   End With
452   With Selection.Characters(Start:=1, Length:=7).Font
453       .FontStyle = "Bold"
454   End With
455
456   'add text box for well name
457       ActiveChart.ChartArea.Select
458      ActiveChart.Shapes.AddLabel(msoTextOrientationHorizontal, 90#, 20, 150, 10).Select     '(Orientation, Left, Top,
      Width, Height)
459   Selection.ShapeRange(1).TextFrame.AutoSize = msoTrue
460   Selection.Characters.Text = "Well Name: " & wellNameText
461   Selection.AutoScaleFont = False
462   With Selection.Characters.Font
463       .Name = "Univers 57 Condensed"  '"Arial"
464       .Size = 9
465       .ColorIndex = xlAutomatic
466   End With
467   With Selection.Characters(Start:=1, Length:=9).Font
468       .FontStyle = "Bold"
469   End With
470
471   'add text box for longitude of well
472       ActiveChart.ChartArea.Select
473       ActiveChart.Shapes.AddLabel(msoTextOrientationHorizontal, 250#, 5, 150, 10).Select
474       Selection.ShapeRange(1).TextFrame.AutoSize = msoTrue
475       Selection.Characters.Text = "Longitude: " & Left(wellLong, 11)
476       Selection.AutoScaleFont = False
477       With Selection.Characters.Font
478       .Name = "Univers 57 Condensed"  '"Arial"
479       .Size = 9
480       .ColorIndex = xlAutomatic
481   End With
482   With Selection.Characters(Start:=1, Length:=9).Font
483       .FontStyle = "Bold"
484   End With
485
486   ' add text box for latitude
487   ActiveChart.Shapes.AddLabel(msoTextOrientationHorizontal, 250#, 20, 150, 10).Select
488   Selection.ShapeRange(1).TextFrame.AutoSize = msoTrue
489   Selection.Characters.Text = "Latitude: " & Left(wellLat, 11)
490   Selection.AutoScaleFont = False
491   With Selection.Characters.Font
492       .Name = "Univers 57 Condensed"  '"Arial"
493       .Size = 9
494       .ColorIndex = xlAutomatic
495   End With
```

```
496
497   With Selection.Characters(Start:=1, Length:=8).Font
498       .FontStyle = “Bold”
499   End With
500
501  ‘add text box for altitude of land surface
502   ActiveChart.ChartArea.Select
503   ActiveChart.Shapes.AddLabel(msoTextOrientationHorizontal, 410#, 5, 250, 10).Select
504   Selection.ShapeRange(1).TextFrame.AutoSize = msoTrue
505   Selection.Characters.Text = “Altitude of Land Surface: “ & wellAlt & “ [ft AMSL]”
506   Selection.AutoScaleFont = False
507   With Selection.Characters.Font
508       .Name = “Univers 57 Condensed”  ‘”Arial”
509       .Size = 9
510       .ColorIndex = xlAutomatic
511   End With
512
513   With Selection.Characters(Start:=1, Length:=24).Font
514       .FontStyle = “Bold”
515   End With
516
517
518  ‘add text box for well depth
519   ActiveChart.Shapes.AddLabel(msoTextOrientationHorizontal, 410#, 20, 150, 10).Select
520   Selection.ShapeRange(1).TextFrame.AutoSize = msoTrue
521   Selection.Characters.Text = “Well Depth: “ & wellDepth & “ [ft]”
522   Selection.AutoScaleFont = False
523   With Selection.Characters.Font
524       .Name = “Univers 57 Condensed”  ‘”Arial”
525       .Size = 9
526       .ColorIndex = xlAutomatic
527   End With
528
529   With Selection.Characters(Start:=1, Length:=10).Font
530       .FontStyle = “Bold”
531   End With
532
533  ‘----------ADD USGS LOGO AND “CREATED ON” DATE TO CHART HEADER----------
534
535  ‘Add header information
536
537  ‘ActiveSheet.PageSetup.LeftHeaderPicture.Filename = “D:\Shared\Graphics\USGS(green)_for_header.tif”
     ‘uncomment this line and insert path and filename for graphic file
538   With ActiveChart.PageSetup
539       .LeftHeader = “&G”
540       .CenterHeader = “”
541       .RightHeader = “&8Created on &D”
542       .LeftFooter = “&G”
543       .CenterFooter = “”
544       .RightFooter = “&G”
545       .LeftMargin = Application.InchesToPoints(0.5)
```

```
546       .RightMargin = Application.InchesToPoints(0.5)
547       .TopMargin = Application.InchesToPoints(0.8)
548       .BottomMargin = Application.InchesToPoints(0.5)
549       .HeaderMargin = Application.InchesToPoints(0.17)
550       .FooterMargin = Application.InchesToPoints(0.2)
551       .ChartSize = xlFullPage
552 '       .PrintQuality = 1200
553       .CenterHorizontally = False
554       .CenterVertically = False
555       .Orientation = xlLandscape
556       .Draft = False
557       .PaperSize = xlPaperLetter
558       .FirstPageNumber = xlAutomatic
559       .BlackAndWhite = False
560       .Zoom = 100
561     End With
562
563
564   '-------------------------------------------------------------------------------------------------------------
565   Application.StatusBar = False
566   Application.ScreenUpdating = True
567
568
569   Charts(chartName).Activate
570
571     xyzRow = xyzRow + 1
572
573     allRowStart = allRowEnd + 1
574
575   '-----------------------------------
576   '-----------------------------------
577   'After multiples of 50 charts.....
578
579   If xyzRow = 51 Then
580
581   ' save current workbook
582     ActiveWorkbook.Save
583
584   ' close the workbook unless it's the original data workbook
585     If wkbookNum > 1 Then
586     ActiveWorkbook.Close SaveChanges:=True    'ThisWorkbook.Close SaveChanges:=True
587   End If
588
589   ' open a new workbook for next 50 charts
590     wkbookNum = wkbookNum + 1
591     Set NewBook = Workbooks.Add
592     fileSaveName = Application.GetSaveAsFilename("MyFileName.xlsx", "Excel files,*.xlsx", 1,
        "Select the folder and workbook name for next set of charts")
593     ActiveWorkbook.SaveAs Filename:=fileSaveName
594
595   Application.SheetsInNewWorkbook = 1
```

```
596  Sheets("Sheet1").Select
597  Sheets("Sheet1").Name = "Information"
598  Worksheets("Information").Cells(1, 1).Value = "Chart data is linked to " & dataWorkbookName & " workbook"
599
600  currentWorkbookName = ActiveWorkbook.Name
601
602 ' open a new worksheet to contain <= 1-year observation data for plotting
603  SolidLineData = "SolidLineData" & wkbookNum
604  Set NewSheet1 = Worksheets.Add
605  NewSheet1.Name = SolidLineData
606  Sheets(NewSheet1.Name).Move After:=Sheets(Worksheets.Count)
607  Worksheets(SolidLineData).Cells(1, 1).Value = "Date"
608  Worksheets(SolidLineData).Cells(1, 2).Value = "WL_Depth/Altitude"
609  Worksheets(SolidLineData).Cells(2, 1).Value = 33556        'dummy data
610  Worksheets(SolidLineData).Cells(2, 2).Value = 100
611
612
613  waitTime = TimeSerial(Hour(Now()), Minute(Now()), Second(Now()) + 3)
614  Application.Wait waitTime
615
616  xyzRow = 1
617  localRowCounter = 2                                  'this is the row on the new worksheet
618
619 End If
620
621
622 Workbooks(dataWorkbookName).Worksheets(allDataSheet).Activate
623
624 Loop        ' big loop to do each chart
625 '-----------------------------------
626 '-----------------------------------
627
628 '-------------------------------------------------------------------------------
629 endTime = Timer
630 elapsedTimeSeconds = endTime - startTime
631 If elapsedTimeSeconds < 0 Then
632  elapsedTimeSeconds = (86400 - startTime) + endTime
633 End If
634 If elapsedTimeSeconds >= 60 Then
635   elapsedTime = elapsedTimeSeconds / 60
636   units = " minutes"
637 Else
638   elapsedTime = elapsedTimeSeconds
639   units = " seconds"
640 End If
641 MsgBox "makeExcelHydrographs program ended.  Elapsed time " & Format(expression:=elapsedTime,
    Format:="Standard") & units
642
643 End Sub
```

makePDFfromChart

```
Sub makePDFfromChart()
Dim PDFfileName As String, dataPath As String
Dim chartNum As Integer

If Dir(Environ("commonprogramfiles") & "\Microsoft Shared\OFFICE" _
    & Format(Val(Application.Version), "00") & "\EXP_PDF.DLL") <> "" Then    'first test to make sure Excel PDF creator is installed


  dataPath = BrowseForFolder("D:\Shared\Ground_Water_Availability\FY09")          'get path for location of hydrographs

 'loop through charts one at a time
  chartNum = 1

  Do While (chartNum) <= Charts.Count

   Charts(chartNum).Activate
   PDFfileName = dataPath & "\" & Charts(chartNum).Name & ".pdf"

   ActiveSheet.ExportAsFixedFormat Type:=xlTypePDF, Filename:= _
     PDFfileName, Quality:=xlQualityStandard, IncludeDocProperties:=True, IgnorePrintAreas _
     :=False, OpenAfterPublish:=False

   chartNum = chartNum + 1
  Loop

  MsgBox "Hydrographs exported to " & dataPath

Else
    MsgBox "PDF add-in Not Installed - Visit http://www.microsoft.com/downloads"
End If

End Sub

Function BrowseForFolder(Optional OpenAt As Variant) As Variant


  Dim ShellApp As Object

   'Create a file browser window at the default folder
  Set ShellApp = CreateObject("Shell.Application"). _
  BrowseForFolder(0, "Select a folder to save pdfs", 0, OpenAt)

   'Set the folder to that selected.
  On Error Resume Next
  BrowseForFolder = ShellApp.self.Path
  On Error GoTo 0

```

```
48     Set ShellApp = Nothing
49
50      'Valid selections can begin L: (where L is a letter) or \\ (as in \\servername\sharename.  All others are invalid
51     Select Case Mid(BrowseForFolder, 2, 1)
52     Case Is = ":"
53        If Left(BrowseForFolder, 1) = ":" Then GoTo Invalid
54     Case Is = "\"
55        If Not Left(BrowseForFolder, 1) = "\" Then GoTo Invalid
56     Case Else
57        GoTo Invalid
58     End Select
59
60     Exit Function
61
62  Invalid:
63      'If it was determined that the selection was invalid, set to False
64     BrowseForFolder = False
65
66  End Function
```