

XLINTERP USER'S MANUAL

VERSION 4.0

COPYRIGHT NOTICE

The **XLInterp** software and manual are copyrighted and licensed for use by one user per copy purchased. This manual and the software described in it are copyrighted with all rights reserved. Under the copyright laws, this manual or the software may not be copied, in whole or part without written consent of Techware Engineering Applications, Inc. Techware Engineering Applications, Inc. grants permission to the purchaser to make a limited number of copies of the add-in for backup purposes only, provided that the copies are not in use at the same time as the original. Additional reproduction of the add-in is a violation of copyright law. Violators will be prosecuted to the fullest extent of the law.

**Copyright© 1998 - 2012
Techware Engineering Applications, Inc.
All rights reserved**

TRADEMARKS

The following trademarks are used throughout this manual. They are registered trademarks of the companies shown.

Microsoft®, and Excel® are trademarks of Microsoft Corporation.

TABLE OF CONTENTS

1. INTRODUCTION	1
2. LOADING XLINTERP	2
3. USING THE INTERPOLATING FUNCTIONS	4
3.1 GENERAL APPROACH	4
3.2 DATA TABLE FORMAT	5
3.2.1 General Format	5
3.2.2 Tables with Descriptive Labels	6
3.3 GENERAL USAGE NOTES	7
3.3.1 Table Search Methods	7
3.3.2 Grid Spacing	7
3.3.3 Multiple Solutions	8
3.3.4 Sharp Changes in Data	8
3.3.5 Discontinuous Data	8
3.3.6 One-Dimensional Tables	9
3.4 GENERAL SYNTAX	9
3.5 VERSION AND SERIAL NUMBER	10
4. FUNCTION REFERENCE	11
5. EXAMPLES	21
5.1 INTERPRCT - EXAMPLE 1, ALL NUMERIC	22
5.2 INTERPRCT - EXAMPLE 2, NUMERIC AND LABEL DATA	22
5.3 INTERPCTR - EXAMPLE 3	23
5.4 INTERPRCDTDC - EXAMPLE 4	23
5.5 INTERPCTDTR - EXAMPLE 5	24

1. INTRODUCTION

XLInterp is an add-in for Microsoft Excel that provides a set of new functions, which perform linear and non-linear interpolation. The functions are useful for extracting data from user-defined tables of values. To maximize versatility, the functions include the following features:

- valid for one- or two-dimensional tables
- can use descriptive labels instead of numeric values to identify an entry in a table
- can perform forward and inverse interpolation using the same data
- can determine rates of change (partial derivatives)

These functions are useful in calculations that require material properties, or equipment performance as functions of one, or two variables. Of course, many other uses also exist. The user should note, however, that the functions perform interpolation not regression. That is, values returned by the functions are always calculated from straight lines or curves connecting points that are actually contained in the tables

XLInterp 4.0 supports 32-bit and 64-bit versions of Microsoft Excel. The installation instructions you received with the **XLInterp** package will guide you through installing the files you'll need for the applications you expect to use. Example files to help you get started will also be installed.

Please read the Licensing Agreement carefully. Understand that each user of your spreadsheets must have his own copy of **XLInterp**. You are not licensed to distribute any of the files you received with this **XLInterp** package.

2. LOADING XLINTERP

XLInterp 4.0 supports 32-bit and 64-bit versions of Microsoft Excel. There are two 32-bit versions of the **XLInterp** add-in for Excel, one for Excel 2007 and later and the other for earlier versions of Excel. Starting with Excel 2007, the allowable size of worksheets became much larger than in previous versions of Excel. The **XLInterp** version for Excel 2007 and later will allow you to use the functions on tables larger than the maximum worksheet size allowed in previous versions of Excel. The Setup program will determine which version of Excel you are using and install the proper add-in automatically.

If you followed the guidelines in the *Installation Instructions*, you should have the proper version of the Excel add-in installed. Before you can use the **XLInterp** functions in Excel, you have to activate the add-in using the Excel Add-in Manager. The procedure for activating the add-in is slightly different for 32-bit and 64-bit versions and so separate instructions are provided below.

Activating the XLInterp Functions for Excel (32-bit)

For versions of Excel prior to that in Office 2007, after starting Excel, use the Tools, Add-ins menu to start Excel's Add-In Manager. (In Office 2007, click the Office button, then click the "Excel Options" button, then select the Add-ins tab and click the "Go..." button at the bottom. In Office 2010, you can get to the Add-ins tab by clicking the File menu and then the Options item.) You should see an item "XLInterp for Excel" in the list box. If you did not find "XLInterp for Excel" in the Add-in Manager list box, click the 'Browse' button and look for the file XLInt32.xll. It should be in the C:\Program Files\Microsoft Office\OfficeXX\Library directory, where OfficeXX is the latest Office version installed. When you find it, click OK. "XLInterp for Excel" should now appear in the list box. If you are running Windows 7, look in the C:\Program Files (x86)\Microsoft Office\OfficeXX\Library directory for the file.

Click the check box next to "XLInterp for Excel" and press 'OK'. The **XLInterp** copyright notice should be displayed on the status bar at the bottom of Excel. The **XLInterp** add-in is now loaded into Excel and will reload every time you start Excel. If you do not want the Add-in to load each time you start Excel, go back to the Add-in Manager and uncheck the **XLInterp** box before closing Excel.

Activating the XLInterp Functions for Excel (64-bit)

Once you have started Excel, click the File Menu, then click the "Options" item, then select the Add-ins tab and click the "Go..." button at the bottom. Click the 'Browse' button and look for the file XLInt64.xll. It should be in the C:\Program Files\TechwareEng\XLInterp directory. When you find it, click OK. "XLInterp for Excel 2010" should now appear in the list box.

Click the check box next to "XLInterp for Excel 2010" and press 'OK'. The **XLInterp** copyright notice should be displayed on the status bar at the bottom of Excel. The **XLInterp** add-in is now loaded into Excel and will reload every time you start Excel. If you do not want the Add-in

to load each time you start Excel, go back to the Add-in Manager and uncheck the **XLInterp** box before closing Excel.

Once selected this way using the Add-in Manager, the functions will be loaded automatically, each time you start Excel. If you wish to unload the functions, use the Add-In Manager and uncheck the box labeled "XLInterp for Excel 2010". Thereafter, **XLInterp** for Excel will not load until you select it again using the Add-In Manager as described above.

Once the add-in is loaded, the interpolating functions are available in the same ways as Excel's built-in functions. That is, they can be typed into cell formulas or they can be inserted by pressing the Toolbar f_x button while creating a cell equation or using the Formula Paste Function menu selection. When using either of the two latter methods, the interpolating functions will be alphabetically sorted in a function category called "Engineering". The Engineering category is probably displayed at the bottom of the Category list. Generally, you can use any of the **XLInterp** functions in any cell formula. Be sure to prefix the function name with an "=" character if it is the first or only item in a formula.

After installation, the SAMPLES subdirectory to the **XLInterp** directory created on your hard disk includes a sample Excel spreadsheet which makes several typical calls to the interpolating functions.

3. USING THE INTERPOLATING FUNCTIONS

3.1 GENERAL APPROACH

XLInterp includes a total of nine functions -- three that perform interpolation and six that derive partial derivatives. A tenth function, `InterpVer()`, described in section 3.5 of this manual, provides the version and serial numbers of your copy of **XLInterp**.

To use the **XLInterp** functions, assemble data in tabular form anywhere on a spreadsheet (see section 3.2, Data Table Format for the format of these tables). The **XLInterp** functions will return values interpolated from within the table for use in cell equations anywhere else on the spreadsheet. The **XLInterp** functions can access any number of tables in a spreadsheet. In fact, the tables don't even have to be in the current workbook. You may find it convenient to create a library of workbooks containing tabular data that you use frequently. You can use the **XLInterp** functions to extract information from these tables without the need to copy the table to each workbook that requires the data. Note, however, that **XLInterp** requires any workbook which contains a table referenced by one of its functions, to be open in order for the functions to work.

The names of all functions begin with "Interp". The other letters making up the function names describe the functions' usage as follows:

- R row
- C column
- T table
- d change (delta)

The sequence of the letters in the function names implies the order of the input parameters and the type of value the functions return. The names of the three interpolating functions are listed below with brief descriptions of each function. The functions and their usage syntax are more fully documented in section 4, Function Reference.

Interpolating functions:

`InterpRCT` - returns a value interpolated from the data table determined by the input row value and column value.

`InterpRTC` - an inverse interpolation function, which returns a column value that would yield the input table value when interpolated from the data table along with the input row value.

`InterpCTR` - an inverse interpolation function, which returns a row value that would yield the input table value when interpolated from a table along with the input column value.

Partial derivative functions:

InterpRCdTdC - returns the partial derivative of the data in the table with respect to the column values at the point determined by interpolating the data in the table using the input row and column values.

InterpRCdTdR - returns the partial derivative of the data in the table with respect to the row values at the point determined by interpolating the data in the table using the input row and column values.

InterpRTdTdC - returns the partial derivative of the data in the table with respect to the column values at the point determined by inverse interpolation using the input row and table values.

InterpRTdTdR - returns the partial derivative of the data in the table with respect to the row values at the point determined by inverse interpolation using the input row and table values.

InterpCTdTdC - returns the partial derivative of the data in the table with respect to the column values at the point determined by inverse interpolation using the input column and table values.

InterpCTdTdR - returns the partial derivative of the data in the table with respect to the row values at the point determined by inverse interpolation using the input column and table values.

The functions provided by **XLInterp** can be used within equations just like Excel's built-in math functions. The functions can even be nested. As output, each function returns a single, floating point result. As inputs, each function requires a reference to the data table, two input arguments and an optional mode selection argument. Later sections of this chapter describe the calling syntax and other considerations in using the functions.

3.2 DATA TABLE FORMAT

3.2.1 General Format

The **XLInterp** functions accept a variety of data table formats. An understanding of these formats and the associated nomenclature is critical to comprehending this manual.

In general, data tables are arranged in contiguous rows and columns. The first row and the first column are special because they contain the 'row values' and 'column values,' respectively. Figure 1 provides a summary of this terminology.

The screenshot shows a Microsoft Excel spreadsheet titled "BigTest.xls". The spreadsheet contains a data table with 23 rows and 9 columns. The first column (A) contains values from 10 to 115 in increments of 5. The remaining columns (B-I) contain numerical values. Two callouts are present: one labeled "Column Values" pointing to the top row (row 1), and one labeled "Row Values" pointing to the first column (column A).

	A	B	C	D	E	F	G	H	I
1		300	350	400	450	500	550	600	650
2	10	1193.679	1217.116	1240.579	1264.12	1287.778	1311.561	1335.549	1359.7
3	15	1192.54	1216.243	1239.895	1263.571	1287.326	1311.201	1335.223	1359.416
4	20	1191.386	1215.361	1239.205	1263.018	1286.872	1310.82	1334.897	1359.13
5	25	1190.216	1214.469	1238.509	1262.462	1286.417	1310.437	1334.569	1358.845
6	30	1189.03	1213.568	1237.808	1261.903	1285.959	1310.054	1334.241	1358.559
7	35	1187.828	1212.657	1237.101	1261.34	1285.5	1309.67	1333.912	1358.272
8	40	1186.608	1211.736	1236.389	1260.774	1285.038	1309.284	1333.583	1357.986
9	45	1185.372	1210.806	1235.67	1260.204	1284.575	1308.897	1333.253	1357.698
10	50	1184.119	1209.865	1234.946	1259.631	1284.109	1308.509	1332.922	1357.411
11	55	1182.849	1208.914	1234.215	1259.055	1283.642	1308.12	1332.591	1357.123
12	60	1181.561	1207.953	1233.478	1258.474	1283.172	1307.73	1332.258	1356.834
13	65	1180.258	1206.991	1232.736	1257.891	1282.7	1307.338	1331.925	1356.545
14	70	269.7191	270.998	1231.986	1257.303	1282.226	1306.945	1331.592	1356.256
15	75	269.7283	270.998	1231.231	1256.711	1281.749	1306.55	1331.257	1355.966
16	80	269.7375	1204.001	1230.469	1256.116	1281.27	1306.155	1330.922	1355.676
17	85	269.7468	1202.987	1229.7	1255.517	1280.789	1305.758	1330.586	1355.385
18	90	269.756	1201.961	1228.926	1254.914	1280.306	1305.359	1330.249	1355.094
19	95	269.7652	1200.924	1228.144	1254.307	1279.821	1304.959	1329.911	1354.802
20	100	269.7744	1199.876	1227.356	1253.696	1279.333	1304.558	1329.572	1354.51
21	105	269.7836	1198.818	1226.561	1253.081	1278.842	1304.156	1329.233	1354.218
22	110	269.7929	1197.747	1225.759	1252.463	1278.35	1303.752	1328.893	1353.925
23	115	269.8021	1196.666	1224.951	1251.839	1277.854	1303.346	1328.552	1353.631

Figure 1 - **XLInterp** Terminology

Except for certain special circumstances, you should not skip lines or columns when creating data tables.

XLInterp only interpolates values within a table. It does not perform extrapolation beyond the bounds of the given table data.

In addition to the numerical data table format described here, the **XLInterp** functions can use tables with descriptive labels in the row and/or column values.

3.2.2 Tables with Descriptive Labels

Often, it is advantageous to use descriptive labels for row and/or column values. For example, it would be intuitive to use equipment model numbers as column values in a table of equipment performance versus temperature. The **XLInterp** functions accept tables that use such labels in either column values or row values or in both. Naturally, the functions cannot interpolate or find derivatives in the dimension using descriptive labels.

The **XLInterp** functions will return values only when there is a match (ignoring leading or trailing spaces and ignoring upper/lower case differences) between a label specified in the function call and a label in the data table. For example, when an **XLInterp** function is given a

descriptive label as a row argument, the function searches through the row values for a matching label. If the function finds a row value that matches, it will use data in that row to determine the return value. If the function searches through the entire data table without finding a match, the function returns a value of #VALUE to indicate an error.

When an **XLInterp** function is searching for a label, it ignores any row or column value that is numeric.

3.3 USAGE NOTES

All forward **XLInterp** functions return values directly from the specified data table when the input values match exactly with row and column values from the table or when using row and column labels. Similarly, the inverse **XLInterp** functions return row or column values directly when there is an exact match with data in the table. Otherwise, the functions use linear interpolation (or, at the user's option, 4-point non-linear interpolation) between values contained in the data table. The functions do not use any form of data regression.

3.3.1 Table Search Methods

In setting up data tables, you will find it helpful to understand how the **XLInterp** functions locate the data used for interpolating. The forward **XLInterp** functions search row and column values in the specified data table in order from first to last (i.e., top to bottom or left to right). The search stops when the function finds two adjacent values that bracket the function's argument (one value greater than the argument and the other smaller). It does not matter whether the row values are in ascending or descending order. The function then uses the values in these rows or columns for linear interpolation. For non-linear interpolation, the function uses four points – usually two on each side of the requested value.

If an **XLInterp** function searches through the entire range of row or column values finding neither an exact match nor two bracketing identifier values, the function returns a #VALUE error.

As a result of these search rules, it does not matter whether row or column values are in ascending or descending order. In fact, **XLInterp** will function properly even if the values change from ascending to descending order within the same table. If there is more than one set of adjacent row or column values that bracket the function argument, the function will use the first one that it finds.

When given a numeric argument, the **XLInterp** functions ignore row or column values that are non-numeric. If a search does encounter a non-numeric value, the search starts over beginning with the next numeric cell. Any numeric row or column values found before the non-numeric identifier are ignored.

If you wish to include non-numeric data in a table (e.g., units or other labels) put it either before or after the numeric data but not in between.

3.3.2 Grid Spacing

Row and column values do not have to be uniformly spaced. When working with data that is non-linear, the interpolation functions yield better results if the grid spacing is selected appropriately. You may use large differences where appropriate and change to smaller differences as required. For example, where a curve is fairly straight, a wide grid spacing may

be acceptable. Where curvature is greater, grid spacing should be tighter so that the interpolating approximations closely follow the actual curve. If the data you are working with is especially non-linear, use the non-linear interpolation option as it accounts for curvature in the data. Be sure to include a row or column value wherever there is a break or a knee in a curve.

3.3.3 Multiple Solutions

The **XLInterp** functions will easily find correct values for y as a function of x in situations like that shown in Figure 2. However, when using the inverse function (x as a function of y) there are two correct values for each y . In this case, the **XLInterp** functions will only find the first value of x . If this does not suit your needs, break the data into two tables at the function's inflection point.

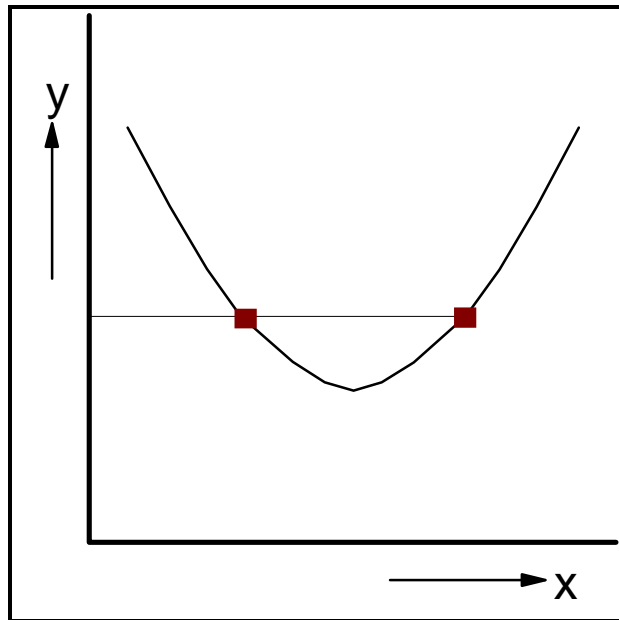


Figure 2 - Multiple Solutions

3.3.4 Sharp Changes in Data

You must be careful if the data in your table exhibits a sharp change in direction. These situations are handled better by use of the linear functions. Be sure to include a row or column value right at the point of change. Be aware that the non-linear functions will force the curve to fit the data which will likely yield poor results, especially with the derivative functions.

3.3.5 Discontinuous Data

To control interpolation for discontinuous data, you can take advantage of the fact that the **XLInterp** functions restart the search after blank cells or labels when searching for numeric row or column values. Whenever there is a discontinuity in data, provide a blank row or column within the data table. The **XLInterp** functions will search values on either side of the blank independently.

For example, if a function is defined for values between 100 and 200 and between 1000 and 2000 leave a blank column between the 200 and the 1000 columns. When the function searches across the columns, if the requested value is between 100 and 200 inclusive, the function will return a valid answer. If the requested value is between 200 and 1000, the function will encounter the non-numeric column identifier after the 200 column identifier and

begin the search with the next column which is 1000. It will not find a solution and will return #VALUE. If the requested value is between 1000 and 2000, the function will find a solution when it resumes its search after the blank column. This method works equally well using either linear or non-linear interpolating methods.

When non-linear interpolation is specified, the functions use a third-order curve fit using four values bracketing the requested point. Normally, the four values are selected two above and two below the selected values. However, when close to the edge of the data table, three of the four values may be on one side of the selected value. Therefore, you should be aware that discontinuities as far away as three columns from the requested point can affect the value calculated using non-linear interpolation.

3.3.6 One-Dimensional Tables

XLInterp provides functions for two-dimensional tables. If your data is one-dimensional (i.e., a function of only one variable), simply use the two-dimensional functions with a table that contains only two columns (or two rows). Use a label for the heading of the column (or row). When using the **XLInterp** functions, use that label as the second parameter in the function call.

For example, to use **XLInterp** to find the specific heat of a gas (e.g., helium) as a function of temperature create a data table as follows:

1. Place temperatures in the first column.
2. Place specific heat data in the second column.
3. Place a label at the top of the second column such as "He".
4. Create a range name for the table including the label row
5. An **XLInterp** function call would look like:

=InterpRCT(Table,125,"He","NL")

You could easily extend this table for additional gases by adding columns of data and labeling each column for its respective gas. When making an **XLInterp** function call, use the appropriate label for the desired gas.

3.4 GENERAL SYNTAX

The three interpolating functions are very similar in their usage and behavior. Therefore, for the sake of clarity, in this User's Manual we provide a general description. Please refer to the **XLInterp** on-line help for detailed function-specific descriptions.

All the **XLInterp** functions use a four-input parameter syntax similar to that shown here:

`Interpxxx(table-range,rowval,colval,mode)`

For all **XLInterp** functions, the first parameter is any valid range containing a table of data. This data is the source for all interpolated results. See Section 3.2 for a description of the layout of data tables.

The remaining parameters can include: a) row value; b) column value; c) table value; or d) linear/non-linear option switch. The functions require two of the first three types of inputs to find the third type. The linear/non-linear option can be either numerical or a label. A value of 0 or "L" calls for linear interpolation. A non-zero or "NL" value specifies non-linear interpolation.

The labels can use either upper or lower case. Chapter 4 provides a detailed description of the specific inputs for each function.

The row and column value parameters can be either numerical or labels depending on the corresponding contents of the data table. For example, if the first column of a data table contains numerical values, the row value should be numerical or the function will return #VALUE. (The function will also return #VALUE if the row value is outside the range of values in the first column.) Otherwise, if the first column of the data table contains labels, the row value should be a label or the function will return #VALUE. (The function will also return #VALUE if the row value does not match one of the labels in the first column of the table.)

Functions that return a table value are considered 'forward' functions since they use the data table in the conventional way. Functions that take a table value as input are considered 'inverse' functions.

3.5 VERSION AND SERIAL NUMBER

There may be new releases of **XLInterp** to add features or correct problems. Each new release will have a different version number. Every copy sold also has a unique serial number. You can identify the version and serial numbers of your copy by using the **InterpVer()** function. It takes no arguments -- but parentheses are needed -- and returns a floating point number containing the information (e.g., 4.012345). The first two digits are the **XLInterp** version number. The next five digits make up your copy's serial number.

4. FUNCTION REFERENCE

This section contains detailed descriptions of each of the **XLInterp** functions:

- **InterpRCT**
- **InterpRTC**
- **InterpCTR**
- **InterpRCdTdC**
- **InterpRCdTdR**
- **InterpRTdTdC**
- **InterpRTdTdR**
- **InterpCTdTdC**
- **InterpCTdTdR**

This section describes the detailed calling syntax for the functions. Chapter 3 addresses generally how the functions search through data tables to calculate return values.

InterpRCT

InterpRCT(table-range,row-value,col-value[,mode]) interpolates in two dimensions from a table.

Arguments	Description
<i>table-range</i>	is any valid range which represents the location of a table of data from which values will be interpolated. The first column in the table contains the values, which identify the data in each row. The first row in the table contains the values, which identify the data in each column. The cell in the upper left hand corner of the table is ignored.
<i>row-value</i>	can be either a numerical value or text. Operation of the function depends upon the contents of the first column of the table. If the first column of the table contains numerical values <i>row-value</i> should be a numerical value or the function will return a #VALUE error. The <i>row-value</i> should not be smaller than the minimum value in the first column nor larger than the maximum value in the first column or the function will return a #VALUE error. If the first column of the table contains text labels <i>row-value</i> should be a text value or the function will return a #VALUE error. If the <i>row-value</i> does not match the contents of a cell in the first column of the table, the function returns a #VALUE error.
<i>col-value</i>	can be either a numerical value or text. Operation of the function depends upon the contents of the first row of the table. If the first row of the table contains numerical values <i>col-value</i> should be a numerical value or the function will return a #VALUE error. The <i>col-value</i> should not be smaller than the minimum value in the first row nor larger than the maximum value in the first row or the function will return a #VALUE error. If the first row of the table contains text <i>col-value</i> should be a text value or the function will return a #VALUE error. If the <i>col-value</i> does not match the contents of a cell in the first row of the table, the function returns a #VALUE error.
<i>mode</i>	an optional parameter which determines whether the interpolation is linear or non-linear. A value of 0 or "L" causes linear interpolation, while a value of 1 or "NL" causes non-linear interpolation. If the <i>mode</i> argument is omitted, the function uses linear interpolation.

Return Value **Description**

If the *mode* is linear:

the function returns a value calculated from the table data using linear interpolation.

If the *mode* is non-linear:

the function returns a value calculated from the table data using non-linear interpolation.

InterpRTC

InterpRTC(table-range,row-value,table-value,[mode]) given a row value and a table value, this function finds the column value which would yield the given table value.

Arguments	Description
<i>table-range</i>	is any valid range which represents the location of a table of data from which values will be interpolated. The first column in the table contains the values, which identify the data in each row. The first row in the table contains the values, which identify the data in each column. The cell in the upper left hand corner of the table is ignored.
<i>row-value</i>	can be either a numerical value or text. Operation of the function depends upon the contents of the first column of the table. If the first column of the table contains numerical values <i>row-value</i> should be a numerical value or the function will return a #VALUE error. The <i>row-value</i> should not be smaller than the minimum value in the first column nor larger than the maximum value in the first column or the function will return a #VALUE error. If the first column of the table contains text labels <i>row-value</i> should be a text value or the function will return a #VALUE error. If the <i>row-value</i> does not match the contents of a cell in the first column of the table, the function returns a #VALUE error.
<i>table-value</i>	must be a numerical value within the range of values contained in the table-range.
<i>mode</i>	an optional parameter which determines whether the interpolation is linear or non-linear. A value of 0 or "L" causes linear interpolation, while a value of 1 or "NL" causes non-linear interpolation. If the <i>mode</i> argument is omitted, the function uses linear interpolation.

Return Value **Description**

If the *mode* is linear:

the function does inverse linear interpolation. It returns a value from within the range of column values that is the result of the inverse interpolation. This column value, if used together with the input row value, would return the input table-value if forward linear interpolation were performed on the table. If the given *table-value* is outside the range of table values, the function will return a #VALUE error. If the column identifiers are non-numeric, the function will return a #VALUE error unless the input table-value exactly equals a value along the given row in which case the function returns the label above that value.

If the *mode* is non-linear:

the function does inverse non-linear interpolation. It returns a value from within the range of column values that is the result of the inverse non-linear interpolation. This column value if used together with the input row value, would return the input table-value if non-linear interpolation were performed on the table. If the given *table-value* is outside the range of table values, the function will return a #VALUE error. If the column identifiers are non-numeric, the function will return a #VALUE error unless the input table-value exactly equals a value along the given row in which case the function returns the label above that value.

InterpCTR

InterpCTR(*table-range*,*column-value*,*table-value*,[*mode*]) given a column value and a table value, this function finds the row value which would yield the given table value.

Arguments	Description
<i>table-range</i>	is any valid range which represents the location of a table of data from which values will be interpolated. The first column in the table contains the values, which identify the data in each row. The first row in the table contains the values, which identify the data in each column. The cell in the upper left hand corner of the table is ignored.
<i>col-value</i>	can be either a numerical value or text. Operation of the function depends upon the contents of the first row of the table. If the first row of the table contains numerical values <i>col-value</i> should be a numerical value or the function will return a #VALUE error. The <i>col-value</i> should not be smaller than the minimum value in the first row nor larger than the maximum value in the first row or the function will return a #VALUE error. If the first row of the table contains text <i>col-value</i> should be a text value or the function will return a #VALUE error. If the <i>col-value</i> does not match the contents of a cell in the first row of the table, the function returns a #VALUE error.
<i>table-value</i>	must be a numerical value within the range of values contained in the <i>table-range</i> .
<i>mode</i>	an optional parameter which determines whether the interpolation is linear or non-linear. A value of 0 or "L" causes linear interpolation, while a value of 1 or "NL" causes non-linear interpolation. If the <i>mode</i> argument is omitted, the function uses linear interpolation.

Return Value **Description**

If the *mode* is linear:

the function does inverse linear interpolation. It returns a value from within the range of row values that is the result of the inverse interpolation. This row value, if used together with the input column value, would return the input table-value if forward linear interpolation were performed on the table. If the given *table-value* is outside the range of table values, the function will return a #VALUE error. If the row identifiers are non-numeric, the function will return a #VALUE error unless the input table-value exactly equals a value in the given column in which case the function returns the label to the left of that row.

If the *mode* is non-linear:

the function does inverse non-linear interpolation. It returns a value from within the range of row values that is the result of the inverse non-linear interpolation. This row value, if used together with the input column value, would return the input table-value if non-linear interpolation were performed on the table. If the given *table-value* is outside the range of table values, the function will return a #VALUE error. If the row identifiers are non-numeric, the function will return a #VALUE error unless the input table-value exactly equals a value in the given column in which case the function returns the label to the left of that row.

InterpRCdTdC

InterpRTC(table-range,row-value,col-value[,mode]) interpolates in two dimensions from a table and returns the partial derivative of the table data with respect to the column values.

Arguments	Description
<i>table-range</i>	is any valid range which represents the location of a table of data from which values will be interpolated. The first column in the table contains the values, which identify the data in each row. The first row in the table contains the values, which identify the data in each column. The cell in the upper left hand corner of the table is ignored.
<i>row-value</i>	can be either a numerical value or text. Operation of the function depends upon the contents of the first column of the table. If the first column of the table contains numerical values <i>row-value</i> should be a numerical value or the function will return a #VALUE error. The <i>row-value</i> should not be smaller than the minimum value in the first column nor larger than the maximum value in the first column or the function will return a #VALUE error. If the first column of the table contains text labels <i>row-value</i> should be a text value or the function will return a #VALUE error. If the <i>row-value</i> does not match the contents of a cell in the first column of the table, the function returns a #VALUE error.
<i>col-value</i>	can be either a numerical value or text. Operation of the function depends upon the contents of the first row of the table. If the first row of the table contains numerical values <i>col-value</i> should be a numerical value or the function will return a #VALUE error. The <i>col-value</i> should not be smaller than the minimum value in the first row nor larger than the maximum value in the first row or the function will return a #VALUE error. If the first row of the table contains text <i>col-value</i> should be a text value or the function will return a #VALUE error. If the <i>col-value</i> does not match the contents of a cell in the first row of the table, the function returns a #VALUE error.
<i>mode</i>	an optional parameter which determines whether the interpolation is linear or non-linear. A value of 0 or "L" causes linear interpolation, while a value of 1 or "NL" causes non-linear interpolation. If the <i>mode</i> argument is omitted, the function uses linear interpolation.

Return Value **Description**

If the *mode* is linear, then:

the function uses linear interpolation to find a point on each of the two columns bracketing the column-value corresponding to the input row-value. The function then returns the rate of change of table values with respect to column values (dTdC). This value is calculated by taking the difference in table values at those two points and dividing by the difference in the two column values.

If the *mode* is non-linear:

the function uses non-linear interpolation to find a point on each of the four columns bracketing the column-value corresponding to the input row value. The function then fits a third order polynomial curve through the four points and returns the rate of change of table values with respect to the column values (dTdC). This value is calculated as the derivative of the polynomial equation at the point equal to column-value.

InterpRCdTdR

InterpRCdTdR(table-range,row-value,col-value[,mode]) interpolates in two dimensions from a table and returns the partial derivative of the table data with respect to the row values.

Arguments	Description
<i>table-range</i>	is any valid range which represents the location of a table of data from which values will be interpolated. The first column in the table contains the values, which identify the data in each row. The first row in the table contains the values, which identify the data in each column. The cell in the upper left hand corner of the table is ignored.
<i>row-value</i>	<p>can be either a numerical value or text. Operation of the function depends upon the contents of the first column of the table. If the first column of the table contains numerical values <i>row-value</i> should be a numerical value or the function will return a #VALUE error. The <i>row-value</i> should not be smaller than the minimum value in the first column nor larger than the maximum value in the first column or the function will return a #VALUE error.</p> <p>If the first column of the table contains text labels <i>row-value</i> should be a text value or the function will return a #VALUE error. If the <i>row-value</i> does not match the contents of a cell in the first column of the table, the function returns a #VALUE error.</p>
<i>col-value</i>	<p>can be either a numerical value or text. Operation of the function depends upon the contents of the first row of the table. If the first row of the table contains numerical values <i>col-value</i> should be a numerical value or the function will return a #VALUE error. The <i>col-value</i> should not be smaller than the minimum value in the first row nor larger than the maximum value in the first row or the function will return a #VALUE error.</p> <p>If the first row of the table contains text <i>col-value</i> should be a text value or the function will return a #VALUE error. If the <i>col-value</i> does not match the contents of a cell in the first row of the table, the function returns a #VALUE error.</p>
<i>mode</i>	an optional parameter which determines whether the interpolation is linear or non-linear. A value of 0 or "L" causes linear interpolation, while a value of 1 or "NL" causes non-linear interpolation. If the <i>mode</i> argument is omitted, the function uses linear interpolation.

Return Value **Description**

If the *mode* is linear, then:

the function uses linear interpolation to find a point on each of the two rows bracketing the row-value corresponding to the input row value. The function then returns the rate of change of table values with respect to row values (dTdR). This value is calculated by taking the difference in table values at those two points and dividing by the difference in row values.

If the *mode* is non-linear:

the function uses non-linear interpolation to find a point on each of the four rows bracketing the row-value at a column value equal to col-value. The function then fits a third order polynomial curve through the four points and calculates the rate of change of table values with respect to row values (dTdR). This value is calculated as the derivative of the polynomial equation at the point equal to row-value.

InterpRTdTdC

InterpRTdTdC(*table-range*,*row-value*,*table-value*,*mode*) does inverse interpolation using a *row-value* and a value from the table data and returns the partial derivative of the table data with respect to the column values.

Arguments	Description
<i>table-range</i>	is any valid range which represents the location of a table of data from which values will be interpolated. The first column in the table contains the values, which identify the data in each row. The first row in the table contains the values, which identify the data in each column. The cell in the upper left hand corner of the table is ignored.
<i>row-value</i>	can be either a numerical value or text. Operation of the function depends upon the contents of the first column of the table. If the first column of the table contains numerical values <i>row-value</i> should be a numerical value or the function will return a #VALUE error. The <i>row-value</i> should not be smaller than the minimum value in the first column nor larger than the maximum value in the first column or the function will return a #VALUE error. If the first column of the table contains text labels <i>row-value</i> should be a text value or the function will return a #VALUE error. If the <i>row-value</i> does not match the contents of a cell in the first column of the table, the function returns a #VALUE error.
<i>table-value</i>	must be a numerical value within the range of values contained in the <i>table-range</i> .
<i>mode</i>	an optional parameter which determines whether the interpolation is linear or non-linear. A value of 0 or "L" causes linear interpolation, while a value of 1 or "NL" causes non-linear interpolation. If the <i>mode</i> argument is omitted, the function uses linear interpolation.

Return Value **Description**

If the *mode* is linear:

the function uses inverse linear interpolation to find the column value which corresponds to the input *row-value* and *table-value*. Next, the function finds a point on each of the two columns bracketing the calculated col-value. The function then returns the rate of change of table values with respect to column values (dTdC). This value is calculated by taking the difference in table values at those two points and dividing by the difference in column values.

If the *mode* is non-linear:

the function uses inverse non-linear interpolation to find the column value which corresponds to the input *row-value* and *table-value*. Next, the function finds a point corresponding to *row-value* on each of the four columns bracketing the determined *column-value*. The function fits a third order polynomial curve to the four points and calculates the rate of change of table values with respect to column values (dTdC) as the derivative of the polynomial equation at the point equal to *column-value*.

InterpRTdTdR

InterpRTdTdR(*table-range*,*row-value*,*table-value*[,*mode*]) does inverse interpolation using a *row-value* and a value from the table data and returns the partial derivative of the table data with respect to the row values.

Arguments

table-range

Description

is any valid range which represents the location of a table of data from which values will be interpolated. The first column in the table contains the values, which identify the data in each row. The first row in the table contains the values, which identify the data in each column. The cell in the upper left hand corner of the table is ignored.

row-value

can be either a numerical value or text. Operation of the function depends upon the contents of the first column of the table. If the first column of the table contains numerical values *row-value* should be a numerical value or the function will return a #VALUE error. The *row-value* should not be smaller than the minimum value in the first column nor larger than the maximum value in the first column or the function will return a #VALUE error.

If the first column of the table contains text labels *row-value* should be a text value or the function will return a #VALUE error. If the *row-value* does not match the contents of a cell in the first column of the table, the function returns a #VALUE error.

table-value

must be a numerical value within the range of values contained in the *table-range*.

mode

an optional parameter which determines whether the interpolation is linear or non-linear. A value of 0 or "L" causes linear interpolation, while a value of 1 or "NL" causes non-linear interpolation. If the *mode* argument is omitted, the function uses linear interpolation.

Return Value Description

If the *mode* is linear:

the function uses inverse linear interpolation to find the column value which corresponds to the input *row-value* and *table-value*. Next, the function finds a point on each of the two rows bracketing the *row-value* corresponding to the calculated *col-value*. The function then returns the rate of change of table values with respect to row values (dTdR). This value is calculated by taking the difference in table values at those two points and dividing by the difference in row values.

If the *mode* is non-linear:

the function uses inverse non-linear interpolation to find the column value which corresponds to the input *row-value* and *table-value*. Next, the function finds a point corresponding to the calculated *column-value* on each of the four rows bracketing the input *row-value*. The function fits a third order polynomial curve to the four points and calculates the rate of change of table values with respect to row values (dTdR) as the derivative of the polynomial equation at the point equal to *row-value*.

InterpCTdTdC

InterpCTdTdC(*table-range, col-value, table-value[, mode]*) does inverse interpolation using a *col-value* and a value from the table data and returns the partial derivative of the table data with respect to column values.

Arguments	Description
<i>table-range</i>	is any valid range which represents the location of a table of data from which values will be interpolated. The first column in the table contains the values, which identify the data in each row. The first row in the table contains the values, which identify the data in each column. The cell in the upper left hand corner of the table is ignored.
<i>col-value</i>	can be either a numerical value or text. Operation of the function depends upon the contents of the first row of the table. If the first row of the table contains numerical values <i>col-value</i> should be a numerical value or the function will return a #VALUE error. The <i>col-value</i> should not be smaller than the minimum value in the first row nor larger than the maximum value in the first row or the function will return a #VALUE error. If the first row of the table contains text <i>col-value</i> should be a text value or the function will return a #VALUE error. If the <i>col-value</i> does not match the contents of a cell in the first row of the table, the function returns a #VALUE error.
<i>table-value</i>	must be a numerical value within the range of values contained in the table-range.
<i>mode</i>	an optional parameter which determines whether the interpolation is linear or non-linear. A value of 0 or "L" causes linear interpolation, while a value of 1 or "NL" causes non-linear interpolation. If the <i>mode</i> argument is omitted, the function uses linear interpolation.

Return Value **Description**

If the *mode* is linear:

the function uses inverse linear interpolation to find the row value which corresponds to the input *col-value* and *table-value*. Next, the function finds a point on each of the two columns bracketing the *col-value* corresponding to the calculated *row-value*. The function then returns the rate of change of table values with respect to column values (dTdC). This value is calculated by taking the difference in table values at those two points and dividing by the difference in column values.

If the *mode* is non-linear:

the function uses inverse non-linear interpolation to find the row value which corresponds to the input *col-value* and *table-value*. Next, the function finds a point corresponding to the calculated *row-value* on each of the four columns bracketing the *col-value*. The function fits a third order polynomial curve to the four points and calculates the rate of change of table values with respect to column values (dTdC) as the derivative of the polynomial equation at the point equal to *col-value*.

InterpCTdTdR

InterpCTdTdR(*table-range, col-value, table-value[, mode]*) does inverse interpolation using a *col-value* and a value from the table data and returns the partial derivative of the table data with respect to the row values.

Arguments	Description
<i>table-range</i>	is any valid range which represents the location of a table of data from which values will be interpolated. The first column in the table contains the values, which identify the data in each row. The first row in the table contains the values, which identify the data in each column. The cell in the upper left hand corner of the table is ignored.
<i>col-value</i>	can be either a numerical value or text. Operation of the function depends upon the contents of the first row of the table. If the first row of the table contains numerical values <i>col-value</i> should be a numerical value or the function will return a #VALUE error. The <i>col-value</i> should not be smaller than the minimum value in the first row nor larger than the maximum value in the first row or the function will return a #VALUE error. If the first row of the table contains text <i>col-value</i> should be a text value or the function will return a #VALUE error. If the <i>col-value</i> does not match the contents of a cell in the first row of the table, the function returns a #VALUE error.
<i>table-value</i>	must be a numerical value within the range of values contained in the table-range.
<i>mode</i>	an optional parameter which determines whether the interpolation is linear or non-linear. A value of 0 or "L" causes linear interpolation, while a value of 1 or "NL" causes non-linear interpolation. If the <i>mode</i> argument is omitted, the function uses linear interpolation.

Return Value **Description**

If the *mode* is linear:

the function uses linear interpolation to find the row value which corresponds to the input *col-value* and *table-value*. Next, the function finds a point on each of the two rows bracketing the calculated *row-value* corresponding to the input *col-value* and returns the rate of change of table values with respect to row values (dTdR). This value is calculated by taking the difference in table values at those two points and dividing by the difference in row values.

If the *mode* is non-linear:

the function uses non-linear interpolation to find the row value which corresponds to the input *col-value* and *table-value*. Next, the function finds a point corresponding to the input *col-value* on each of the four rows bracketing the calculated *row-value*. The function fits a third order polynomial curve to the four points and calculates the rate of change of table values with respect to row values (dTdR) as the derivative of the polynomial equation at the point equal to the calculated *row-value*.

5. EXAMPLES

This section offers five examples that are representative of the operation of all the functions. The examples use the following functions:

- InterpRCT - Example 1, all numeric
- InterpRCT - Example 2, numeric and labels
- InterpCTR - Example 3
- InterpRCdTdC - Example 4
- InterpCTdTdR - Example 5

5.1 INTERPRCT - EXAMPLE 1, ALL NUMERIC

The table below, named **TABLE1**, is defined as cells (A1..E6). It represents a function of two variables. To make this example meaningful, the table actually represents the enthalpies of steam as a function of pressure and temperature. The numbers in row 1 represent temperatures in degrees F, while the numbers in column A represent pressures in psia. The cells in the area (B2..E5) hold the enthalpy values of steam in Btu/lb.

	A	B	C	D	E
1		400	500	600	700
2	10	1240.58	1287.78	1335.55	1384.05
3	50	1234.95	1284.11	1332.92	1382.02
4	100	1227.36	1279.33	1329.57	1379.46
5	150	1219.10	1274.32	1326.14	1375.88
6	200	1210.13	1269.04	1322.61	1374.25

The formula **=InterpRCT(TABLE1,60,440,0)** entered in any cell finds the enthalpy of steam at a pressure of 60 psia and a temperature of 440 deg F. In this example, the value returned is 1253.32, rounded to the nearest hundredth. If the last argument in the formula is changed from a 0 to either a 1 or an "NL", the function uses non-linear interpolation and returns a value of 1253.48. It is interesting to note that the ASME steam tables lists the enthalpy of steam at 60 psia and 440 deg F as 1253.5 Btu/lb, rounded to the nearest tenth. Techware's **WinSteam** software returns a more precise value of 1253.51. Even with the coarse table used in this example, the non-linear function returns an excellent result.

5.2 INTERPRCT - EXAMPLE 2, NUMERIC AND LABEL DATA

The table below, named **TABLE2**, is defined as cells (A1..D7). It represents the ideal gas enthalpies of several gases over a temperature range from 50 deg F to 500 deg F. The first variable, listed in column A, represents the temperatures in deg F. The second variable, listed in row 1, identifies the different gases, N2, O2, and CO2. The cells in each column below the gas names hold the values of the ideal gas enthalpies corresponding to the temperatures in column A.

	A	B	C	D
1		N2	O2	CO2
2	50	126.36	110.71	86.09
3	100	138.78	121.68	96.16
4	200	163.64	143.79	117.28
5	300	188.58	166.24	139.59
6	400	213.64	189.07	162.96
7	500	238.88	212.32	187.26

The formula **=InterpRCT(TABLE2,240,"CO2",0)** entered in any cell finds the ideal gas enthalpy of CO2 at 240 deg F by linear interpolation as 126.20. If the last argument in the formula is changed from a 0 to either a 1 or an "NL", the function uses non-linear interpolation and returns a value of 126.07. It is interesting to note that Techware's **@Gas** software, also returns a value of 126.07, which indicates that the non-linear interpolation yields excellent results.

5.3 INTERPCTR - EXAMPLE 3

The table below, named **TABLE1**, is the same table from Example 1. The numbers in row 1 represent temperatures in degrees F, while the numbers in column A represent pressures in psia. The cells in the area (B2..E6) hold the enthalpy values of steam in Btu/lb.

	A	B	C	D	E
1		400	500	600	700
2	10	1240.58	1287.78	1335.55	1384.05
3	50	1234.95	1284.11	1332.92	1382.02
4	100	1227.36	1279.33	1329.57	1379.46
5	150	1219.10	1274.32	1326.14	1375.88
6	200	1210.13	1269.04	1322.61	1374.25

If you have steam at a given temperature and enthalpy, you can use the InterpCTR function to find the corresponding pressure. For example, if the temperature of 440 deg F and the enthalpy is 1253.5 (see Example 1), the formula **=InterpCTR(TABLE1,440,1253.5,0)** finds the value of the pressure as defined in the table by using inverse linear interpolation. In this example, the value returned is 58.6, rounded to the nearest tenth. If the last argument in the formula is changed from a 0 to either a 1 or an "NL", the function uses non-linear inverse interpolation and returns a value of 59.8, rounded to the nearest tenth. Again, even with this coarse table, the inverse non-linear function returns a good result.

5.4 INTERPRCDTDC - EXAMPLE 4

The table below, named **TABLE1**, is the same table from Example 1. The numbers in row 1 represent temperatures in degrees F, while the numbers in column A represent pressures in psia. The cells in the area (B2..E6) hold the enthalpy values of steam in Btu/lb. From thermodynamics, we know that the specific heat at constant pressure is defined as the derivative of enthalpy with respect to temperature at constant pressure. So, we can use the data in **TABLE1** to calculate the specific heat.

	A	B	C	D	E
1		400	500	600	700
2	10	1240.58	1287.78	1335.55	1384.05
3	50	1234.95	1284.11	1332.92	1382.02
4	100	1227.36	1279.33	1329.57	1379.46
5	150	1219.10	1274.32	1326.14	1375.88
6	200	1210.13	1269.04	1322.61	1374.25

The formula **=InterpRCdTdC(TABLE1,60,440,0)** finds the value of the derivative of the table data (enthalpy) with respect to column values (temperature) at a constant row value (pressure). The function returns a numerical approximation to the specific heat using linear interpolation. In this example, the rounded return value is 0.4973 Btu/lb/deg F. If the *mode* is changed from a 0 to either a 1 or an "NL", the function uses non-linear interpolation and returns a value of 0.4976 Btu/lb/deg F. It is interesting to note that the ASME steam tables list the specific heat of steam at 60 psia and 440 deg F as 0.498 Btu/lb/deg F, and Techware's **WinSteam** function returns a more precise value of 0.4975 Btu/lb/deg F. Even with this coarse table, the non-linear function returns an excellent result.

5.5 INTERPCTDTR - EXAMPLE 5

The table below named, TABLE2, is the same table described in Example 2. The first variable, listed in column A, represents the temperatures in deg F. The second variable, listed in row 1, identifies the different gases, N2, O2, and CO2. The cells in each column below the gas names hold the values of the ideal gas enthalpies corresponding to the temperatures in column A. From thermodynamics, we know that the specific heat at constant pressure is defined as the derivative of enthalpy with respect to temperature at constant pressure. So, we can use the data in TABLE2 to calculate the specific heat.

	A	B	C	D
1		N2	O2	CO2
2	50	126.36	110.71	86.09
3	100	138.78	121.68	96.16
4	200	163.64	143.79	117.28
5	300	188.58	166.24	139.59
6	400	213.64	189.07	162.96
7	500	238.88	212.32	187.26

For example, if we know the gas and its ideal enthalpy, we can find its ideal gas specific heat using the formula **=InterpCTdTR(TABLE2,"CO2",126.07,0)**. "CO2" is the column value, 126.07 is the enthalpy in Btu/lb/deg F and the "0" instructs the function to use linear interpolation. The function returns a value of 0.223, which represents the specific heat in units of Btu/lb/deg F. If the last argument in the formula is changed from a 0 to either a 1 or an "NL", the function uses non-linear interpolation and returns a value of 0.222 Btu/lb/deg F.