Dipl.-Phys.-Ing. Ralf Wirtz

Issue B3

2018

SimplexNumerica

Excel®-like SpreadSheet Module

with FormulaEngine

V18

Programming & Visualization

Excel-like SpreadSheet Module with FormulaEngine

This documentation is provided to familiarize you with the fundamentals of the *SimplexNumerica* SpreadSheet Module and its sophisticated FormulaEngine.

Info

→ This manual is an extension to the main SimplexNumerica manual.

→ The spreadsheet is based on Stingray's Objective Grid. Developed by Stefan Hoenig, meanwhile President & CEO, Syncfusion Inc.

→ I have extended Objective Grid over my 15 years Stingray and later Rogue Wave subscription for SimplexNumerica, only.

Please visit the last page for the license agreement!

SIMPLEXNUMERICA, *SIMPLEXNUMERICA*, SIMPLEXEDITOR AND Simplexety ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL THE AUTHOR BE LIABLE TO YOU FOR ANY DAMAGES, INCLUDING INCIDENTAL OR CONSEQUENTIAL DAMAGES, ARISING OUT OF THE USE OF THE PROGRAM, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Content

CONTENT 2

1 DEVELOPMENT 5

2 QUICK START 6

3 INTRODUCTION 9

3.1 Cell Values 9
3.1.1 Text Entries 9
3.1.2 Numeric Values 9
3.1.3 Dates and Times 9
3.1.4 Formulas 10

3.2 Multilevel Undo/Redo 10

3.3 Named Ranges 10

3.4 Copying and Moving Cells 11

3.5 Constraint Checking 11

3.6 Iterative Calculations 11

3.7 Precision 12

3.8 Embedded Tools 12

4 USER INTERFACE 13

4.1 Entrance 13

4.2 Ribbonbar 15

5 WORKSHEET FUNCTIONS 19

6 THE FORMULA ENGINE 21

6.1 Undo/Redo 21
6.1.1 Changes that cannot be undone 21

6.2 Formula Engine Limitations 21

6.3 Entering Numeric Values 22
6.3.1 Changing the Numeric Format 22

6.4 Select Date/Time Format 22

6.5 Entering Formulas 23
6.5.1 Built-in Functions 24

6.6 Ranges 24

6.6.1 Named Ranges 24

6.7 Copying and Moving Cells 25
6.7.1 Copying Formulas 25
6.7.2 Moving Data 26

6.8 Deleting Rows and Columns 26

6.9 Calculations 26
6.9.1 14.4.1 How SimplexNumerica Calculates 26
6.9.2 Mode of Recalculation 27
6.9.3 Methods of Recalculation 27
6.9.4 Iterative Calculations 27
6.9.5 Constraint Checking 28
6.9.6 Precision 28

6.10 SimplexNumerica Formulas 29
6.10.1 Formula Syntax 29
6.10.2 Formula Values 30
6.10.3 Formula Operators 30
6.10.4 Referencing Other Cells in Formulas 32
6.10.5 Cell Referencing in SimplexNumerica 32
6.10.6 Constraint Expressions 34
6.10.7 Explicit Dependency 34

6.11 Built-in Functions 35
6.11.1 Mathematical Functions 36
6.11.2 Statistical Functions 36
6.11.3 Conditional Statistical Functions 36
6.11.4 String Functions 36
6.11.5 Logic Functions 36
6.11.6 Digital Logic Functions 37
6.11.7 Financial Functions 37
6.11.8 Date and Time Functions 38
6.11.9 Miscellaneous Functions 39
6.11.10 Embedded Tools 39
6.11.11 Using SimplexNumerica Built-in Functions 39

6.12 Quick-Reference Guide to Built-in Functions 42
6.12.1 Mathematical Functions 42
6.12.2 Statistical Functions 44
6.12.3 Conditional Statistical Functions 46
6.12.4 String Functions 46
6.12.5 Logic Functions 48
6.12.6 Financial Functions 48
6.12.7 Date and Time Functions 51
6.12.8 Miscellaneous Functions 52
6.12.9 Embedded Tools 53

6.13 Error Messages 55
6.13.1 Types of Errors 55
6.13.2 Error Messages Reference 55

7 END-USER LICENSE AGREEMENT 64

1 Development

The *SimplexNumerica* spreadsheet module and its formula engine is based on Stingray's Objective Grid, developed by Stefan Hoenig. For *SimplexNumerica,* a lot of effort was taken into account to extend and present it as it meanwhile is - and the best remains to deal with it in the manner in which it has been designed with the goal to use it similar to Microsoft Excel®.

The module is fully integrated into *SimplexNumerica*, but still not a "have to" use. You can use the conventional GraphTable as before.

If you have any questions, please do not hesitate to contact me.

- Ralf Wirtz, Software Engineer
and Developer

Email: support*SimplexNumerica*.com
Web: www.*SimplexNumerica*.com

Hint

A formula starts always with the equal = sign, like = sin(0.1) or = cos(B12)

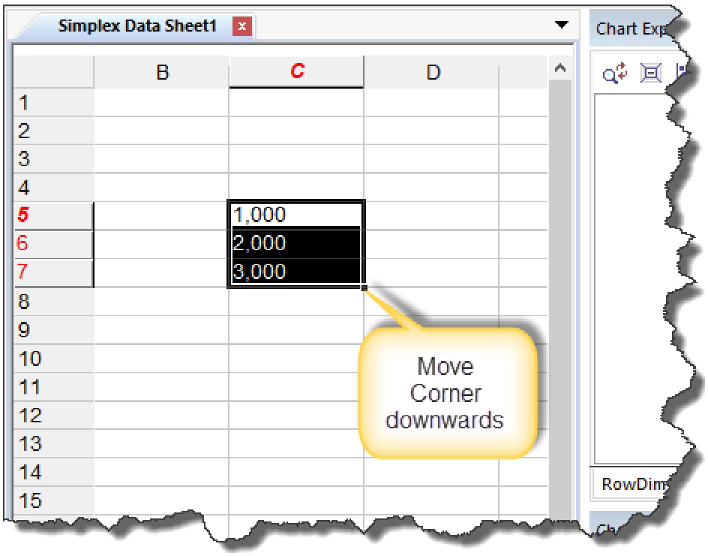
2 Quick Start

⇨ *See chapter 4 for the user interface and start-up via Start-up Dialog.*

The simplest way to start with the new spreadsheet module is via the Start-up Dialog. But here we will show the steps with a blanked background just after running the *SimplexNumerica* application.

Please follow the steps:

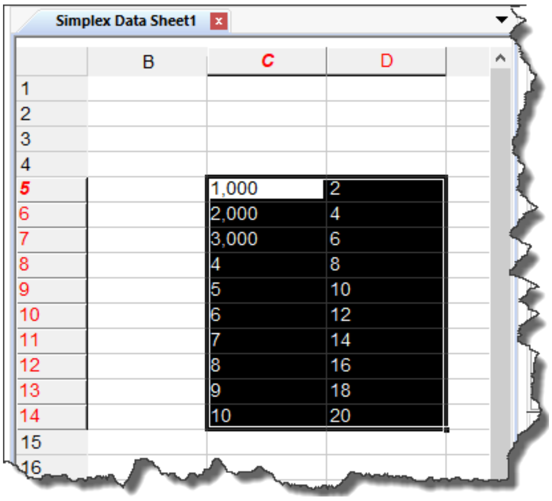
1. Double-click on the desktop icon and start *SimplexNumerica*.
2. For this quick start here, please close the Start-up dialog (if available).
3. Press key Ctrl + K and open an empty spreadsheet.
4. Click on a cell and input number 1, then click on cell underneath and input 2, then 3.



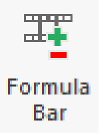
	B	C	D
1			
2			
3			
4			
5		1,000	
6		2,000	
7		3,000	
8			
9			
10			
11			
12			
13			
14			
15			

Move Corner downwards

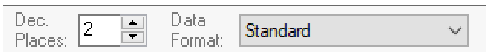
5. Select with the mouse these 3 cells and move the right bottom corner until reaching 10.
6. Place the cursor in cell next to 1 (in picture D5).
7. Write formula: = C5 * 2.
8. Move this cell cursor corner downwards to the left cell 10.
9. Select this data range.



	B	C	D
1			
2			
3			
4			
5		1,000	2
6		2,000	4
7		3,000	6
8		4	8
9		5	10
10		6	12
11		7	14
12		8	16
13		9	18
14		10	20
15			
16			

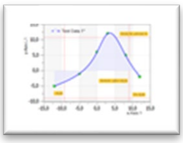


10. Show the Formula/Format bar and select Standard format and 2 decimal places.

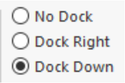


Dec. Places: 2 Data Format: Standard

11. Next, we want to generate a chart with that data range.

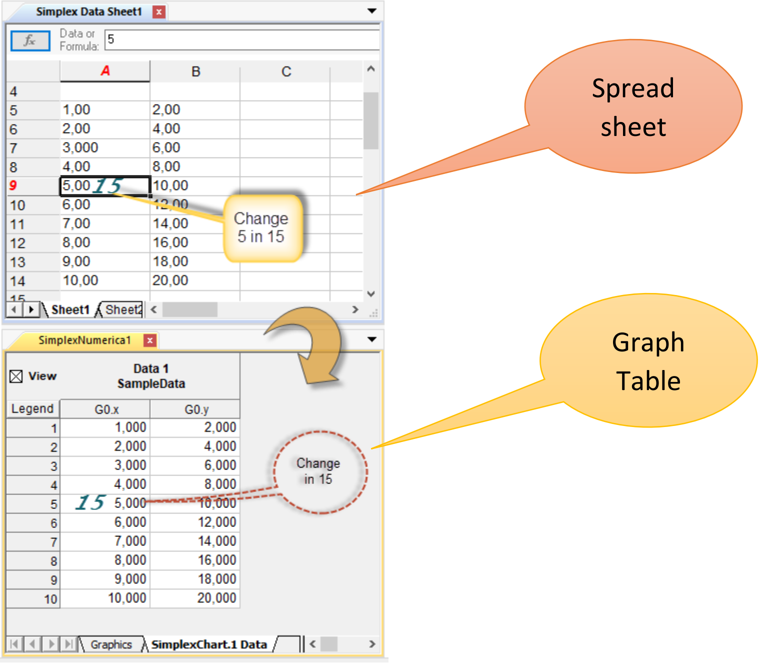


12. Click on this thumbnail symbol:



○ No Dock
○ Dock Right
● Dock Down

13. Because of this: → You should see the two MDI windows side-by-side (docked is it called).
14. Change value 5 in 15 to see the reaction in the *GraphTable* underneath.



Simplex Data Sheet1

Data or Formula: 5

	A	B	C
4			
5	1,00	2,00	
6	2,00	4,00	
7	3,000	6,00	
8	4,00	8,00	
9	5,00 15	10,00	
10	6,00	12,00	
11	7,00	14,00	
12	8,00	16,00	
13	9,00	18,00	
14	10,00	20,00	

SimplexNumerica1

☒ View — Data 1 SampleData

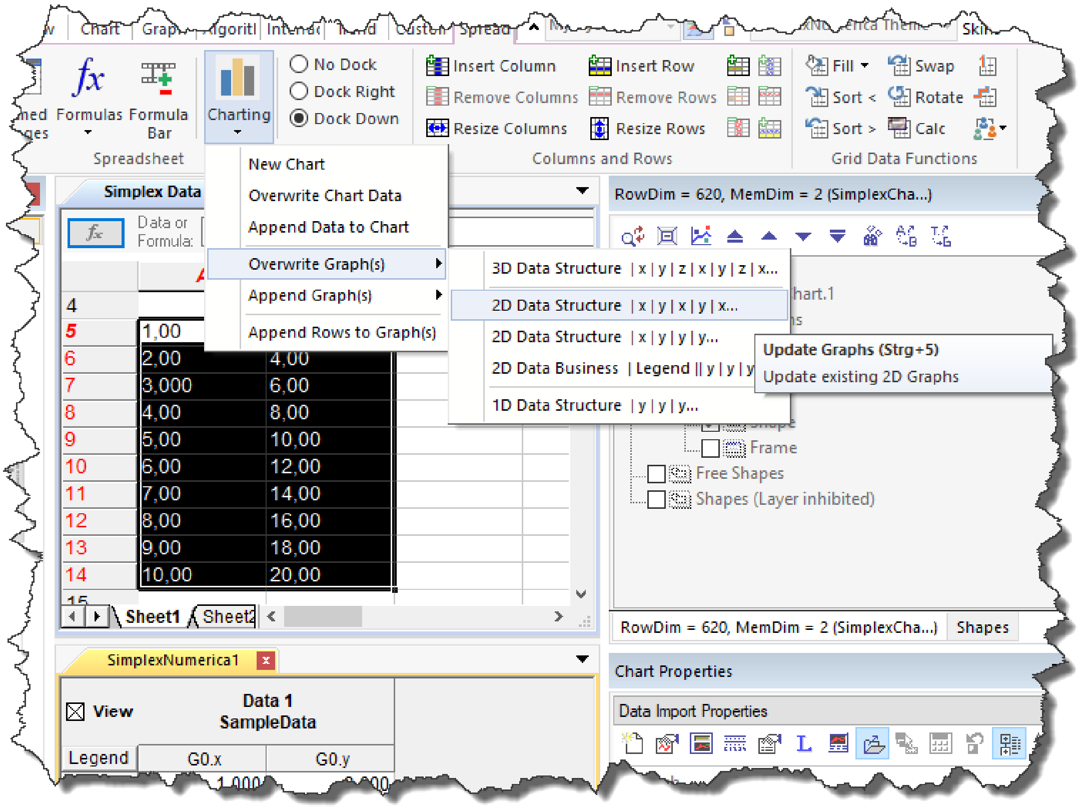
Legend	G0.x	G0.y
1	1,000	2,000
2	2,000	4,000
3	3,000	6,000
4	4,000	8,000
5	15 5,000	10,000
6	6,000	12,000
7	7,000	14,000
8	8,000	16,000
9	9,000	18,000
10	10,000	20,000

→ If you have changed a value in the spreadsheet, then this value should have been changed in the *GraphTable*, too.

```
Info
SimplexNumerica remembers on the last chart that was selected with the data range
from the spreadsheet. If you change data - because of this - it will change the
data in the GraphTable.
```

That are the main steps from a data range to a chart and updating its data. Later in this documentation you will see other possibilities for updating chart data.

Have a look to this:



4		
5	1,00	
6	2,00	4,00
7	3,000	6,00
8	4,00	8,00
9	5,00	10,00
10	6,00	12,00
11	7,00	14,00
12	8,00	16,00
13	9,00	18,00
14	10,00	20,00

3 Introduction

Hint

A formula starts always with the equal = sign in a cell or in the formula bar, like = sin(0.1) or = cos(B12)

Formulas are the backbone of a spreadsheet, establishing and calculating mathematical relationships between elements of the spreadsheet. Whereas numeric entries remain the same until you change them, cells defined by formulas are automatically changed to reflect changes in referenced cells - even where there are complex interdependencies among cells.

Info

With SimplexNumerica's Formula Engine you can have up to a million rows and 4096 columns.

Here is some overview about the formula engine[1]:

3.1 Cell Values

SimplexNumerica accepts several basic types of cell entries: text, constant numeric values, dates or times, formulas that calculate a value, and graphs. Calculated values may be single numbers or strings, or they can be arrays or tables of values.

3.1.1 Text Entries

Text entries are useful for labeling columns and rows, for including comments about data values being calculated, and for using *SimplexNumerica* to manage textual information, such as names, addresses or whatever your application may require.

3.1.2 Numeric Values

If a cell entry begins with a digit from 0 - 9, *SimplexNumerica* treats the entry as a numeric entry. *SimplexNumerica* also recognizes the following symbols as indicators of numeric entries: ``+", ``-", and ``.". You can format numeric values to be displayed in several ways, including fixed formats, scientific notation, currency, and hex.

3.1.3 Dates and Times

SimplexNumerica provides special, built-in features for displaying date entries in the format you choose. Example date and time formats include: 24-Oct-2017, 24-Oct, 10/24, Oct-17, 10/24/2017, 24.10.2017, and 2017-10-24 (ISO 8061). Time is displayed as: 12:00:05.

[1] You will find this text also in SimplexNumerica: Use Pulldownmenu File → New →Sample Excel-like Workbook

3.1.4 Formulas

Formulas establish and calculate mathematical relationships between elements of the spreadsheet. *SimplexNumerica* formulas can calculate with numbers, text, logical values, cell references, and other formulas. For example, you can easily calculate the sum of a series of cells, the total of values in a column, or the absolute value of another cell entry.

The real power of *SimplexNumerica* lies in its ability to calculate relationships among different cells in the spreadsheet by typing the row/column coordinates, or address, in the formula.

To reference a cell by address you should Type the row and column coordinates of the cell in the formula. For example, to reference Row 5 in Column D, type D5.

To reference a contiguous group of cells by address type the row and column coordinates of two cells in opposite corners of the block to be referenced, with two periods (..) between the coordinates. For example, to reference the first five columns and the first five rows of the spreadsheet, type A1..E5.

Hint

See the sample worksheets in SimplexNumerica for an overview of worksheet functions!

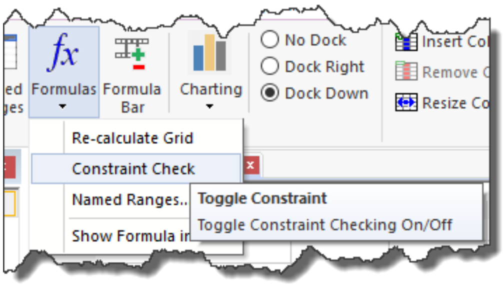
3.2 Multilevel Undo/Redo

The formula engine supports multilevel undo/redo, e.g. when cell references get update because of a move operation undo needs to take care of these changes. But, all these changes don't affect your existing applications and the API is 100% compatible.

3.3 Named Ranges

SimplexNumerica allows you to assign a name to a range of cells. These cells can then be referenced via the range name.

To edit, insert or rename ranges select Ribbonbar *Named Ranges*



You can assign a range via 'MYTEXT=A62..A63' or 'MYNUMBERS=A62..B62' in the dialog.

Later you can reference this range via =STRCAT(MYTEXT) or = AVG(MYNUMBERS)

3.4 Copying and Moving Cells

The Copy operation (either clipboard copy or ole drag & drop) duplicates a cell or range of cells (along with all formatting) and places these formulas in a new location, overwriting existing data in the destination range. *SimplexNumerica* automatically translates relative cell references in the copied formulas to reflect their new locations. For example, if cell A10 contains the formula =SUM(A1..A9) and is copied to cell B10, then B10 will contain the formula =SUM(B1..B9).

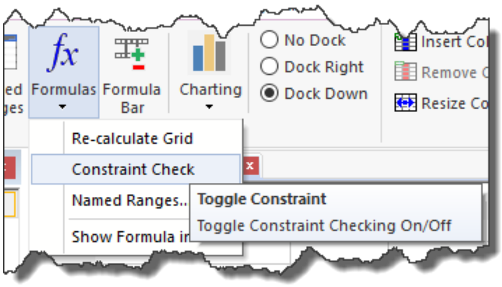
The Move operation moves a cell or range of cells to a new location, along with all formulas. *SimplexNumerica* clears the source cells and overwrites any existing data in the destination cells. Like the Copy Formulas operation, all cell references are updated to reflect the new cell/range location.

3.5 Constraint Checking

Constraints are limitations or conditions placed on the variables in your spreadsheet. They are expressed as algebraic statements appended to formulas. You can attach a constraint expression to any formula, by typing a semicolon (;) and the constraint conditions after the formula.

For example, the formula =A1 + A2 ; #>2 && #<=B5 || #==C7 means, ``the value of the current cell is the sum of cells A1 and A2, and that value must be either greater than 2 and less than or equal to the value of cell B5, or equal to the value of cell C7.

You can turn on/off constraint checking via the Constraint Checking menu item in the Ribbonbar.



3.6 Iterative Calculations

Normally, a formula in a given cell should not depend on that cell itself, either directly or indirectly. Such a condition is called a cyclic dependency. When cyclic dependencies exist, the rule for natural order recalculation as described above does not make sense. When you enter a formula, which creates a cyclic dependency, the message ``Cycle!'' is displayed in the cell.

In some cases, cyclic dependencies are useful in that they can represent iterative calculations, which *SimplexNumerica* supports. Iterative calculation is useful when two or more cells mutually depend on each other such that each time they are recalculated, their values become closer and closer to the desired answer.

3.7 Precision

SimplexNumerica performs all calculations in double-precision. Calculations with logical operators like

- ! (logical NOT)
- && (logical AND)
- || (logical OR)
- ? : (conditional)

consider a non-zero value to be **True** and a zero value to be **False**. Integer operators like

- ~ (complement)
- & (bitwise AND)
- | (bitwise OR)
- ^ (bitwise EXCLUSIVE-OR)
- % (modulus)

convert their operands to 32-bit integers before performing the operation.

3.8 Embedded Tools

Embedded Tools are a powerful feature in *SimplexNumerica*. Their power derives in part from their ability to return a set of data, not just a single value. This function makes non-scalar operations such as matrix multiplication and "live" recalculation as easy to use as an ordinary spreadsheet function.

Embedded tools store values in a group of adjacent cells. These adjacent cells are set to constant formulas with explicit dependencies on their neighboring cells. For example, an embedded tool in cell B2 might produce the formula =1.3459B2 in cell B3. This formula indicates that the cell currently contains the constant 1.3459 but that its value depends on the contents of cell B2 (the cell containing the embedded tool).

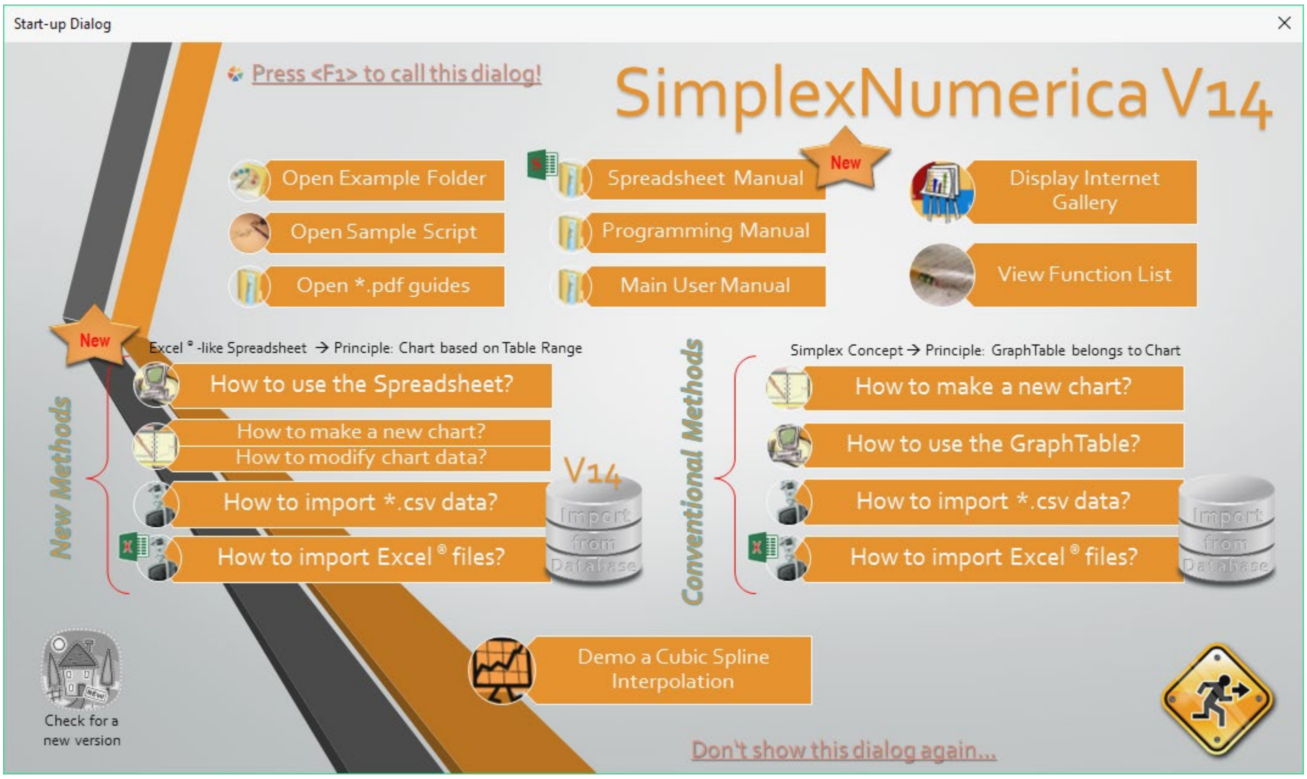
This notion of explicit dependencies is important for recalculation. It guarantees that any cell that references B3 will not be recalculated until after cell B2 is recalculated. This ensures that data generated by the embedded tool is always current.

Embedded tools look like normal functions, and they can be copied, moved and formatted just as any other formula in the spreadsheet. However, you must follow one important guideline: DO NOT combine embedded tools with other embedded tools in a single formula.

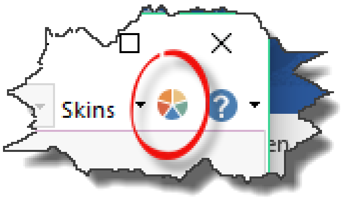
4 User Interface

4.1 Entrance

By far the simplest way to deep into the new *SimplexNumerica* Spreadsheet Module is after the start of the program, then when the Start-up dialog appears.

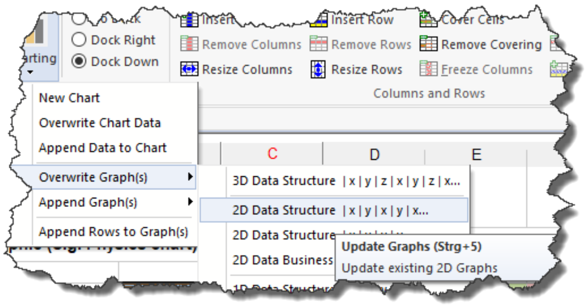

Start-up Dialog
Press <F1> to call this dialog!
SimplexNumerica V14
Open Example Folder
Open Sample Script
Open *.pdf guides
Spreadsheet Manual
New
Programming Manual
Main User Manual
Display Internet Gallery
View Function List
New
New Methods
Excel ® -like Spreadsheet → Principle: Chart based on Table Range
How to use the Spreadsheet?
How to make a new chart?
How to modify chart data?
How to import *.csv data?
How to import Excel ® files?
V14
Import from Database
Conventional Methods
Simplex Concept → Principle: GraphTable belongs to Chart
How to make a new chart?
How to use the GraphTable?
How to import *.csv data?
How to import Excel ® files?
Import from Database
Demo a Cubic Spline Interpolation
Check for a new version
Don't show this dialog again...

The New Methods on the left side will showing you in a simple and concise form - via balloon tips - the necessary steps to work around the Excel®-like Spreadsheet.



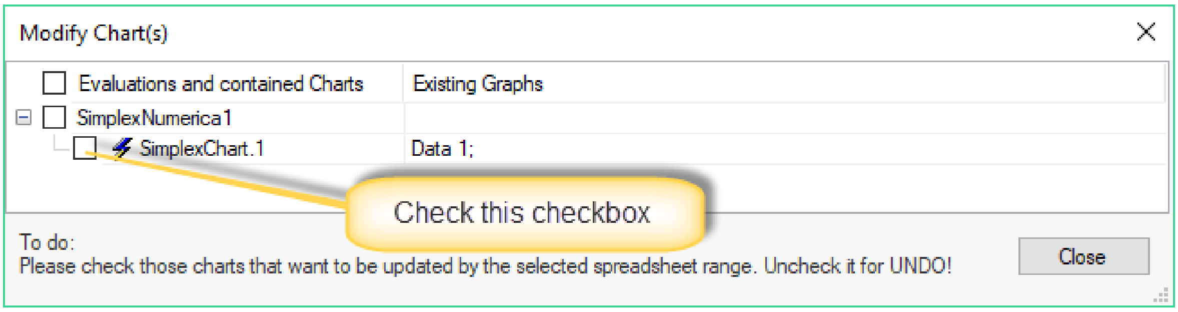
```
Hint
You can call the Start-up dialog again via key <F1> or
use this toolbar symbol.
```

→ Please click on the entry How to use the Spreadsheet? and follow the steps…



SimplexNumerica will open the sample sheet. The range around cell Graph1 is already selected. Now, you can select a thumbnail chart symbol and immediately the chart is filled by your selected data.

If there is already a preferred chart in any evaluation in the background, then you can use the Ribbonbar icon ***Charting*** and a matching popup menu entry to overwrite existing data from the chart. For that purpose, a dialog opens to select one or more (if available) charts whose data should be overwritten.

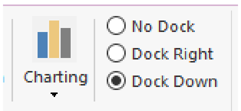


Modify Chart(s)

Evaluations and contained Charts	Existing Graphs
⊟ □ SimplexNumerica1	
□ SimplexChart.1	Data 1;

To do:
Please check those charts that want to be updated by the selected spreadsheet range. Uncheck it for UNDO!

Immediately the chart MDI window pops up from the background and goes underneath the spreadsheet MDI window (if intended so). You can set this docking behavior in the Ribbonbar:



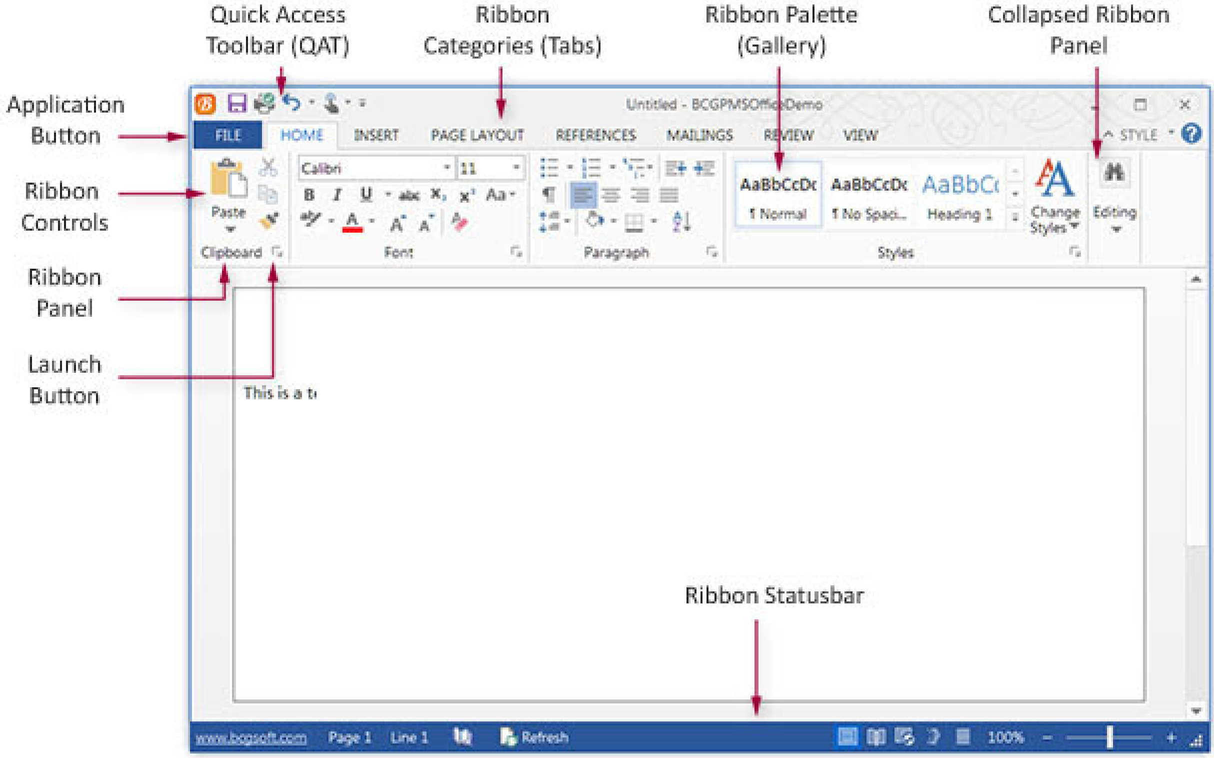
○ No Dock
○ Dock Right
● Dock Down

That is the way to show spreadsheet data based on one or more sheet ranges in a new or existing chart.

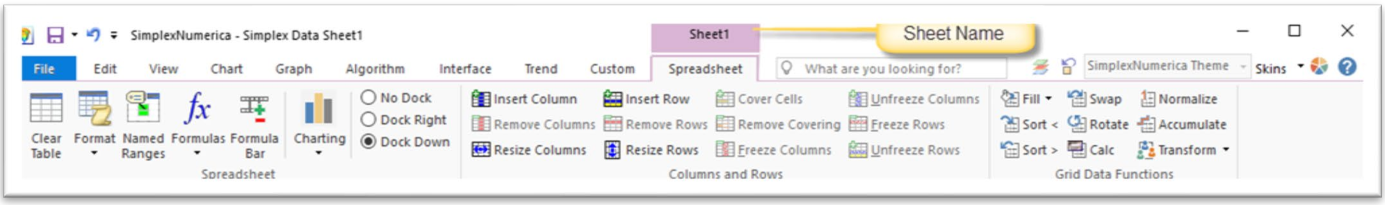
4.2 Ribbonbar

The Ribbonbar was introduced by Microsoft in Office 2007. It's not just a new control - it's a new user interface ideology. A Ribbonbar replaces traditional toolbars and menus with tabbed groups (Categories). Each tab is logically split into Panels and each panel may contain various controls and command buttons. In addition, a Ribbonbar provides smart layout maximally utilizing the available space. For example, if a Panel has been stretched and has no place to display all available controls, it becomes a menu button which can display sub-items on a popup menu.

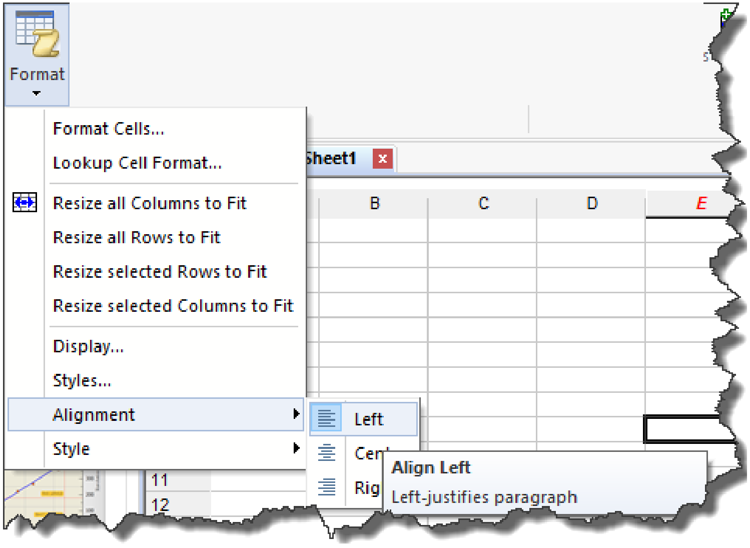
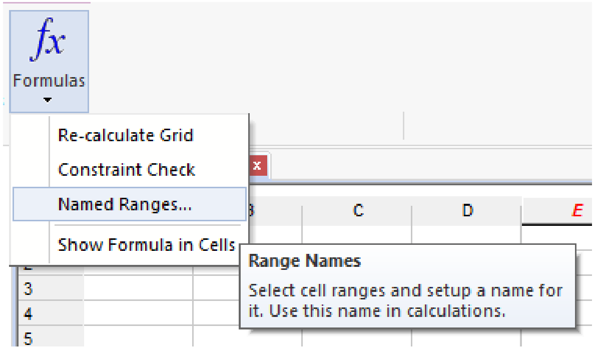
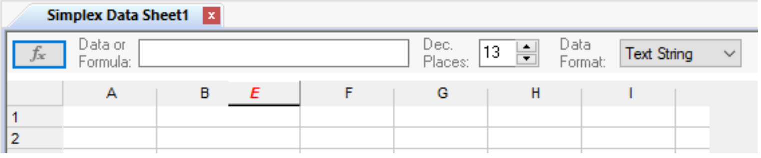
Here a common view of an Office-like Ribbonbar (similar to *SimplexNumerica*'s one).

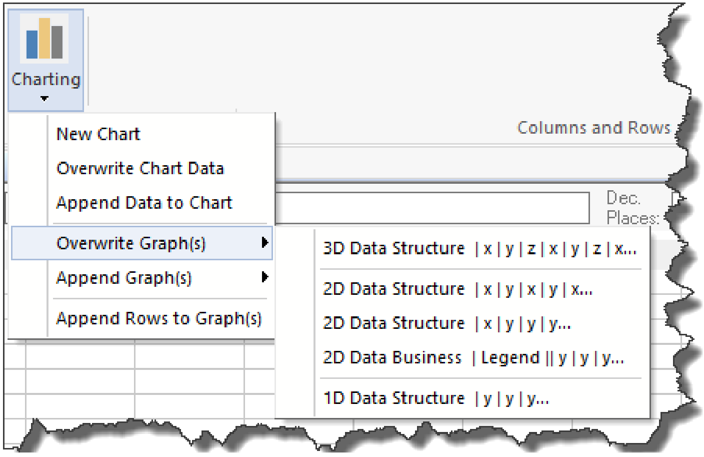


The spreadsheet module has its own Ribbonbar, that always appears when a sheet gets visible.



Here a summarizing overview about the Ribbonbar entries:

Symbol	Funktion
Clear Table	**Clear Table** Does what it says.
Format	**Format Table Range** Drop down the popupmenu:  Format Format Cells... Lookup Cell Format... Resize all Columns to Fit Resize all Rows to Fit Resize selected Rows to Fit Resize selected Columns to Fit Display... Styles... Alignment Style Left Cen Rig Sheet1 B C D E 11 12 Align Left Left-justifies paragraph
Named Ranges	**Named Ranges** SimplexNumerica allows you to assign a name to a range of cells. These cells can then be referenced via the range name.
Formulas	**Formula Procedures** Drop down the popupmenu:  Formulas Re-calculate Grid Constraint Check Named Ranges... Show Formula in Cells C D E 3 4 5 Range Names Select cell ranges and setup a name for it. Use this name in calculations.
Formula Bar	**Formula / Format Bar** Show/Hide the formula bar.  Simplex Data Sheet1 A B E F G H I 1 2 Data or Formula: Dec. Places: 13 Data Format: Text String

Charting	**Select an existing chart** Drop down the popupmenu:  Charting New Chart Overwrite Chart Data Append Data to Chart Overwrite Graph(s) Append Graph(s) Append Rows to Graph(s) Columns and Rows Dec. Places: 3D Data Structure | x | y | z | x | y | z | x... 2D Data Structure | x | y | x | y | x... 2D Data Structure | x | y | y | y... 2D Data Business | Legend || y | y | y... 1D Data Structure | y | y | y... If there is already a chart in a graphics view, then select a data range in any spreadsheet and update therewith chart's data.
○ No Dock ○ Dock Right ● Dock Down	**Dock MDI Windows automatically** Does it make sense to show the spreadsheet and graphics MDI window underneath after the above Charting menu? Yes, experience shows that such a window adjustment would appear necessary to provide for, so that you immediately can see and compare the results.
Insert Column	**Insert Column to the right** Hold Ctrl key to insert to the left.
Remove Columns	**Remove Cursor Column**
Resize Columns	**Resize Column**
Insert Row	**Insert Row to beneath** Hold Ctrl key to insert above.
Remove Rows	**Remove Cursor Row**
Resize Rows	**Resize Row**
Cover Cells	**Cover selected Cells (Range)** SimplexNumerica lets you cover cells. This means that one cell can span several other cells.
Remove Covering	**Remove Cover over Cursor placed Cell**
Freeze Columns	**Freeze Columns** Sets the selected column to the left side so that it stays fix and you can scroll only the other columns. Then during the horizontal scrolling, our selected column behaves the whole time visible.

	Unfreeze Columns Unfreeze the last column again.

5 Worksheet Functions

SimplexNumerica's functions are predefined formulas supplied with the program. They offer a shortcut approach to accomplishing the work of long, complex formulas. Mathematical and statistical functions are often used to sum a column of numbers, compute an average, determine a minimum or maximum value, or round the results of a formula.

Other functions are used for more specialized purposes such as computing the future value of an investment or the product of multiplying one cell range by another range. Some *SimplexNumerica* functions fall into the following categories:

- **Mathematical**
  Mathematical Functions perform calculations such as determining absolute value, finding the integer portion of a number, or establishing the value of a constant. Although you could accomplish these tasks with a formula, using a function saves time and trouble.
- **Statistical**
  Statistical Functions perform aggregation operations such as calculating means, minimums, maximums, and averages.
- **Conditional** Statistical
  Conditional Statistical Functions operate much like statistical aggregation functions, except that the last argument is a constraint expression that *SimplexNumerica* evaluates for each cell in the argument list. Only cells that meet constraint criteria are included in the calculation. The constraint expression may be any *SimplexNumerica* expression that evaluates to a numeric result.
- **Strings**
  String Functions manipulate and evaluate character strings. For example, string functions can return the length of a string, find the first occurrence of a string in a range, change a string from upper to lower-case and vice versa, or replace one string with another.
- **Logic**
  Logic Functions return one value if an argument meets certain criteria, another value if it does not. Logic functions are used as an adjunct to conditional statements.
- **Digital Logic**
  Digital Logic Functions perform digital logic operations such as AND, OR, NOT, etc.
- **Financial**
  Financial Functions perform common financial calculations, such as calculating the future value of an annuity at a given interest rate, straight-line depreciation, double-declining depreciation, or the payment term for a given investment. The financial functions in *SimplexNumerica* cover annuities, cash flows, assets. bonds, and Treasury Bills.
- **Date and Time**
  Date and Time Functions return values corresponding to the specified date, month, year, hour, minute or second. You can also use date/time functions to enter the current system time and date in a cell. These functions open up many possibilities for managing accounts receivable and calculating test times.
- **Miscellaneous**
  Miscellaneous Functions perform a variety of calculations, such as returning a reference to specific cells or ranges or returning the Nth argument from a list of arguments.

- **Embedded Tools**
  Embedded Tools are a powerful feature in *SimplexNumerica*. Their power derives in part from their ability to return a set of data, not just a single value. This function makes non-scalar operations such as matrix multiplication and "live" recalculation as easy to use as an ordinary spreadsheet function.

6 The Formula Engine

Formulas, the heart of any spreadsheet application, allow you to establish and calculate mathematical relationships among elements in the spreadsheet cells. While numeric entries remain the same until you change them, cells defined by formulas change automatically to reflect changes in referenced cells, even when there are complex interdependencies among cells.

SimplexNumerica formulas can calculate with numbers, text, logical values, cell references, and other formulas. For example, you can easily calculate the sum of a series of cells, the total of values in a column, a minimum or maximum value within a range, the rounded result of another formula, or the absolute value of a cell entry. Formulas can also express complex interdependencies among cells and they can define calculation constraints, such as limits on acceptable values or specific conditions under which a calculation should take place.

Once you enter a formula into a cell, the formula works behind the scene. The only trace of its existence is the result of the calculation. To view the formula in a cell, select the cell. You can edit the formula or the value in a cell at any time.

6.1 Undo/Redo

SimplexNumerica supports multi-level Undo/Redo for all operations in the engine. For example, if cell references are updated because of a move operation, Undo takes care of these changes.

6.1.1 Changes that cannot be undone

The only changes that cannot be undone are changes that are the result of an embedded tool or matrix operation because embedded tools store values in a group of adjacent cells. These adjacent cells are set to constant formulas with explicit dependencies on their neighboring cells. For example, if you have a matrix in A7..C9 and cell D10 contains the matrix function

"=TRANSPOSE(A7..C9)", the cells D10..F12 are the result set of the matrix operation.

If the user changes any cell in the range D10..F12, the change is stored as a value in the cell. If the user then changes a value in the range A7..C9, the cells D10..F12 are overwritten and no Undo information for the manually changed cells is generated.

6.2 Formula Engine Limitations

When using the formula engine, please note the following limitations:

- The maximum number of columns is 4096.
- The maximum number of rows is one million rows.
- The maximal text size for a cell is 512 bytes.
- Formulas can only refer to cells within the same sheet. There is no support for getting values from other sheets.

6.3 Entering Numeric Values

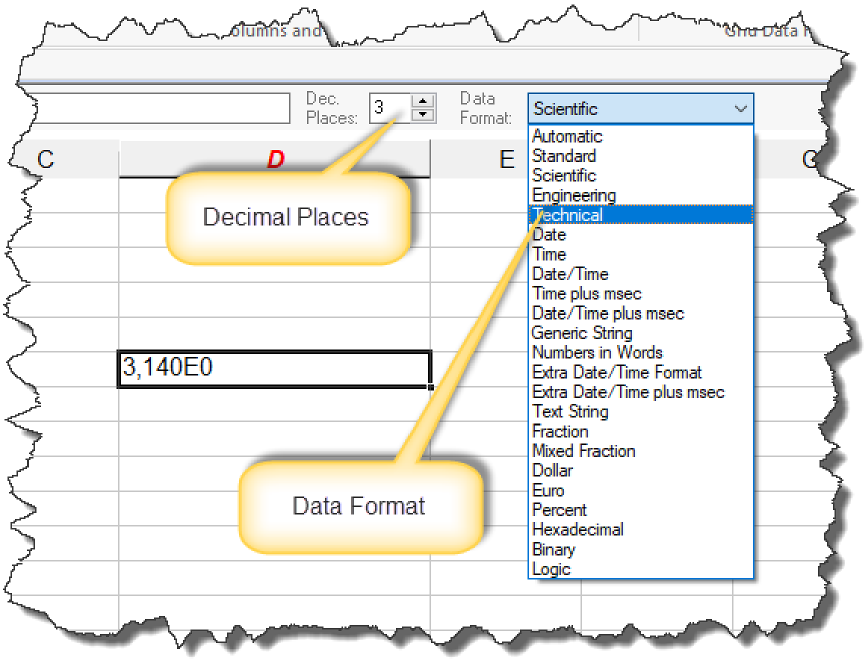
Follow these conventions for entering numeric values:

- To enter a positive number, use the number keys to type the number, with or without a + indicator. If you do not type a plus (+), *SimplexNumerica* assumes the number is positive.
- To enter a negative number, type a minus sign (-) and the number. Do not use parentheses to indicate negatives. However, if you change the numeric format to Dollars or Comma, *SimplexNumerica* displays negative numbers in parentheses.
- Do not use spaces or commas when entering numbers. You can display commas, if you want to, by changing the format.
- Be careful not to substitute a lower case L for the numeral 1 or the upper case O for the numeral 0.
- You can use scientific notation to enter a number, with the convention that the letter e separates the fraction from the base 10 exponent. 1.23E3 is equivalent to 1230.

6.3.1 Changing the Numeric Format

You can choose from a variety of formats for displaying numbers. The display formats do not change the number itself, the way the number is stored internally, or the way it is used in calculations. Formatting just changes the way *SimplexNumerica* displays numbers.

The format for a Numeric Value can be set in the Format and Formula bar:



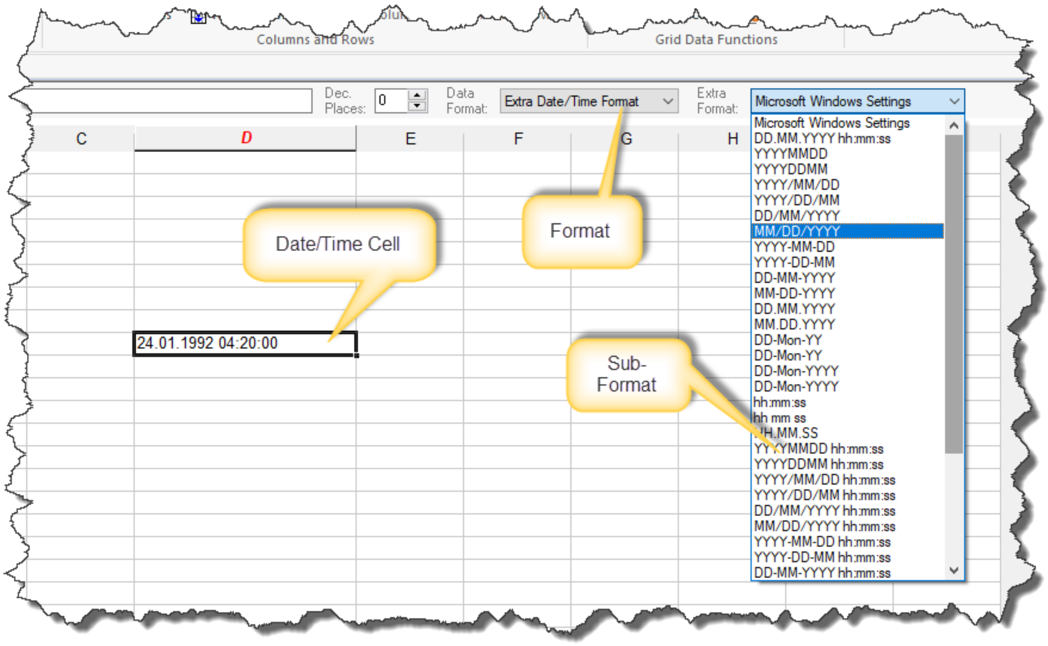
Dec. Places: 3

Data Format: Scientific

3,140E0

6.4 Select Date/Time Format

SimplexNumerica provides special, built-in features for displaying date entries in the format you choose. Date and time formats can be set in the Format and Formula bar:



Dec. Places: 0
Data Format: Extra Date/Time Format
Extra Format: Microsoft Windows Settings

24.01.1992 04:20:00

Use this Ribbonbar icon to show/hide the format bar: Formula Bar

Hint

SimplexNumerica provides the Date/Time internally as Julian Day
https://en.wikipedia.org/wiki/Julian_day

6.5 Entering Formulas

Formulas establish and calculate mathematical relationships between elements of the spreadsheet, e.g.:

= sin(A1) …calculates the sinus from cell A1.

Hint

Start always a formula input with the = sign!

SimplexNumerica formulas can calculate with numbers, text, logical values, cell references, and other formulas. For example, you can easily calculate the sum of a series of cells, the total of values in a column, or the absolute value of another cell entry.

Formulas are the heart of the *SimplexNumerica* spreadsheet, defining relationships between the values in other cells. For example, formulas can be used to sum a column of numbers, compute an average, determine a minimum or maximum value, or round the results of a calculation.

While constant entries remain the same until you change them, cells defined by formulas are automatically changed to reflect changes in referenced cells— even when there are complex interdependencies among cells.

Once entered in a cell, formulas are hidden behind the scenes, performing their work in the background and displaying only the result of their calculation. To view the formula in a cell, move the cell cursor to the cell and switch the cell into edit mode. You can edit the formula or values within the cell.

6.5.1 Built-in Functions

The *SimplexNumerica* Formula Engine comes with over 300 built-in worksheet functions. Functions can be used alone or in combination with formulas and other functions. *SimplexNumerica* was designed for demanding users, so it provides many highly specialized functions for business, engineering, scientific, and statistical applications.

6.6 Ranges

A range is a contiguous, rectangular block of cells that has been referenced as a group in a cell formula or selected to be affected by some editing action, such as Copy Formulas or Move.

Hint

To define more than one range in a data sheet, please hold down the Ctrl key and select ranges with the pressed left mouse button.

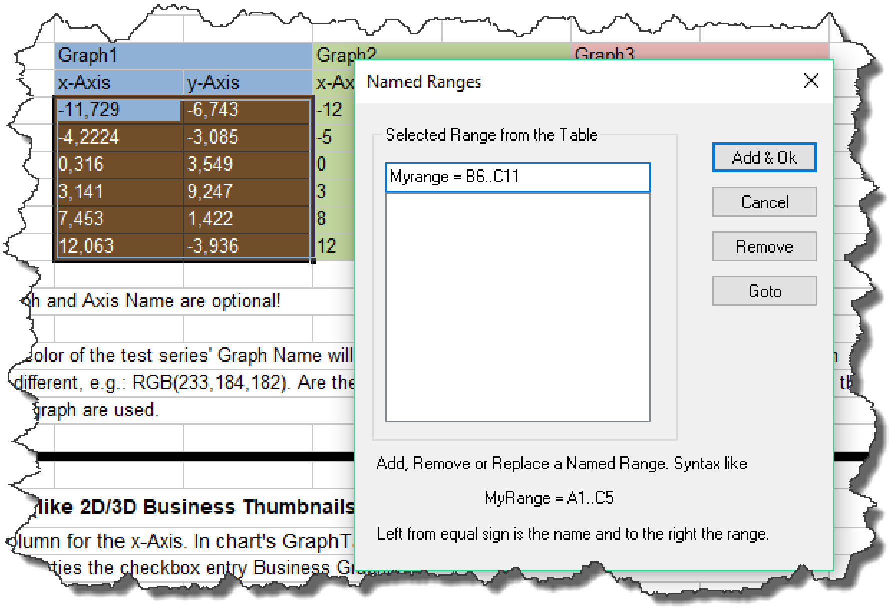
6.6.1 Named Ranges

Individual cells and cell ranges may be referred to by their standard address coordinates (A5..D25, for example) or by pre-assigned names. The Named Range option lets you assign a name to any cell or cell range.

Named Ranges

A named range is a range of cells to which a name has been assigned with the Named Range utility.

Use the Ribbonbar icon Named Ranges to call the following dialog



Graph1		Graph2	Graph3
x-Axis	y-Axis	x-Ax	
-11,729	-6,743	-12	
-4,2224	-3,085	-5	
0,316	3,549	0	
3,141	9,247	3	
7,453	1,422	8	
12,063	-3,936	12	

h and Axis Name are optional!

olor of the test series' Graph Name will
different, e.g.: RGB(233,184,182). Are the
graph are used.

like 2D/3D Business Thumbnails
lumn for the x-Axis. In chart's GraphT
ties the checkbox entry Business Gr

Named Ranges

Selected Range from the Table

Myrange = B6..C11

Add, Remove or Replace a Named Range. Syntax like

MyRange = A1..C5

Left from equal sign is the name and to the right the range.

A named cell is a cell to which a name has been assigned. Both named ranges and named cells can be referenced by their names or by their addresses.

Using names can help clarify the logic of your spreadsheet, making it easier to share among users, and easier to update long after it was originally designed. Once defined, names can be used anywhere standard cell or range references are used, including in cell formulas.

The following guidelines apply to named ranges:

- *SimplexNumerica* does not differentiate between uppercase and lowercase letters in range names. For example, *SimplexNumerica* would consider "Range1" and "range1" to be the same name.
- Names must begin with an uppercase or lowercase alphabetic character. The rest of the name may include any combination of alphabetic characters, numeric characters, and the following characters: $, ., or _.
- When an area containing a named range or cell is moved, *SimplexNumerica* automatically updates the definition of the name to reflect the new location.
- Named cells and ranges can have relative or absolute addresses. When cells containing references to named cells with absolute addresses are moved, the references are not updated.
- When cells containing a reference to a named cell or range are copied, *SimplexNumerica* converts the reference to an appropriately translated standard reference.
- If you redefine a name, all instances of that name in the spreadsheet are updated as soon as the spreadsheet is recalculated.
- If you remove a name definition, all references to that name in the spreadsheet are converted to appropriate standard references.
- To name a cell or range via script, call the SetRangeName() command:
  Example:
  SetRangeName(_T("MyNumber"), 6, 1, 10, 4);

6.7 Copying and Moving Cells

6.7.1 Copying Formulas

The Copy operations, clipboard copy and OLE Drag-and-Drop, duplicate a cell or range of cells in addition to all the formatting and then place these formulas in a new location, overwriting existing data in the destination range. *SimplexNumerica* automatically translates relative cell references in the copied formulas to reflect their new locations. For example, if cell A10 contained the formula = SUM(A1..A9) and you copied it to cell B10, then B10 would contain the formula = SUM(B1..B9).

To create an absolute cell reference, which is a value that *SimplexNumerica* doesn't translate if the cell contents are copied or moved, insert a dollar sign ($) before each component of the cell reference (e.g., A1).

Use the script function CopyCells() method to perform a copy operation programmatically.

SimplexNumerica overwrites any existing data in destination ranges, so plan carefully before carrying out Copy and Move operations. If you overwrite data by mistake during a Copy or Move, you can put it back like it was using Undo.

6.7.2 Moving Data

The Move operation moves a cell or range of cells to a new location, along with all formulas. *SimplexNumerica* clears the source cells and overwrites any existing data in the destination cells. Like the Copy Formulas operation, all cell references are updated to reflect the new cell/range location.

Use the script function MoveCells() method to perform a move operation programmatically.

The effects of moving cells include:

- When you move a cell that is referenced by a formula, *SimplexNumerica* automatically updates the formula for the new location, even if the cell reference is absolute.
- When you move a cell in a range that is referenced by a formula, the formula still references the original range. However, if you move a corner cell of the range, *SimplexNumerica* extends or contracts the range to match the new location.
- If you move a cell range that has been named or referenced in a formula, *SimplexNumerica* automatically updates the definition of the range name or the formula reference to reflect the new location.
- If you move part of a cell range that has been named or referenced in a formula, it can cause problems. *SimplexNumerica* updates range names and references only when you move one or more corner cells of the range. If you move cells in the interior of the range, *SimplexNumerica* does not change the range name or any references to it.

If you make a mistake when copying or moving data, you can use Undo to restore the spreadsheet to its state prior to the copy or move operation.

6.8 Deleting Rows and Columns

If you delete a row or column that contains a cell that is referenced by a formula in another cell, the formula reference will not be modified. Range references will be updated when rows or columns are deleted within a range specification.

6.9 Calculations

This chapter explains *SimplexNumerica* calculations, including:

- How *SimplexNumerica* calculates
- How to change calculation options such as automatic recalculation and iteration limit, and
- Elements of formulas.

6.9.1 14.4.1 How SimplexNumerica Calculates

SimplexNumerica uses natural order recalculation, which is a method of ordering the computations such that each cell's value is computed only after the values for all cells on which it depends have been computed. Natural order recalculation guarantees that cells are always computed correctly, regardless of the order in which the cell formulas were entered. Minimized grid windows are recalculated the same as visible sheets.

SimplexNumerica supports several recalculation options:

- Mode (Manual or Automatic)
- Method (Foreground or As Needed)
- Iteration Limit
- Constraint Checking
- Precision

6.9.2 Mode of Recalculation

SimplexNumerica supports two modes of recalculation: Manual and Automatic. When Manual is set, formulas are only recomputed when an explicit recalculation operation is requested. When Automatic is set, any action which causes the contents of a cell to change automatically triggers recalculation of any formulas depending on the changed cell.

6.9.3 Methods of Recalculation

Recalculation is the process whereby *SimplexNumerica* re-computes the mathematics of the worksheet after the user makes changes. *SimplexNumerica* lets you designate whether recalculation takes place on demand (manual mode) or after any change is made (automatic mode).

Natural recalculation is a method of calculation that follows a natural hierarchy of the importance of cell dependencies.

Automatic recalculation supports two methods of recalculations: Foreground and As Needed.

Foreground

When Foreground is set, *SimplexNumerica* enforces minimal recalculation, which means that only the cells in the spreadsheet that are potentially affected by an operation that you perform (such as editing a cell, or moving a range of cell) are recalculated. This feature minimizes the work performed during recalculation and thus speeds up your computations. Minimal recalculation is in effect whenever the recalculation is set to Automatic and the method is set to Foreground.

As Needed

When As Needed is set, *SimplexNumerica* enforces frugal recalculation, which further reduces the number of cells that need to be computed. The performance gain in recalculation can be significant, depending on the size and complexity of the sheet. Frugal recalculation is in effect whenever the recalculation mode in the Recalc Options dialog is set to Automatic and method is set to As Needed.

As Needed does not support Constraint Checking and Iterative Recalculation options. If all of this seems a bit confusing, refer to Figure 126 for clarification.

6.9.4 Iterative Calculations

Normally, a formula in a given cell should not depend on that cell itself, either directly or indirectly. A circular reference is when two or more cells are defined by each other. Such a condition is called a cyclic dependency. When cyclic dependencies exist, the rule for natural order recalculation as described above does not make sense. When you enter a formula that creates a cyclic dependency, the message "Cycle!" is displayed in the cell.

In some cases cyclic dependencies are useful in that they can represent iterative calculations, which *SimplexNumerica* supports. Iterative calculation is a method of calculation whereby *SimplexNumerica* passes through the worksheet more than once, addressing circular references or performing goalseeking. Iterative calculation is useful when two or more cells mutually depend on each other such that each time they are recalculated, their values become closer and closer to the desired answer.

When the Iteration Limit is set to a non-zero value and Method is set to Foreground, iterative calculation is enabled. In this mode, *SimplexNumerica* will make multiple recalculation passes, still preserving the natural order and minimal recalculation rules described above, until either the iteration limit has been reached or, if constraint checking is enabled, until all constraints are satisfied. The INIT function helps perform iterative calculations.

A forward reference is a cell whose value depends on values calculated later in a spreadsheet.

6.9.5 Constraint Checking

Constraint checking is a process whereby *SimplexNumerica* checks constraint expressions while recalculating the worksheet. If constraint-checking is disabled in the Default Sheet Characteristics dialog box, *SimplexNumerica* ignores constraint expressions.

A constraint expression is an expression appended to a formula that establishes conditions under which the formula operates or boundaries for valid results of the formula.

SimplexNumerica formulas may contain conditional expressions which establish conditions under which a formula operates or boundaries for valid results of the formula. When calculating the spreadsheet, *SimplexNumerica* ignores constraint expressions unless constraint checking is enabled and the recalculation method is set to Foreground for the spreadsheet.

6.9.6 Precision

SimplexNumerica performs all calculations in double-precision. Calculations with logical operators - ! (logical NOT), && (logical AND), || (logical OR), and ?: (conditional) - consider a non-zero value to be True and a zero value to be False. Integer operators - ~ (complement), & (bitwise AND), | (bitwise OR), ^ (bitwise EXCLUSIVE-OR), and % (modulus) convert their operands to 32-bit integers before performing the operation.

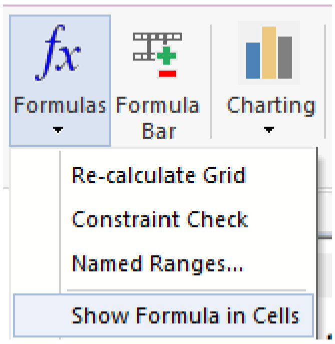
6.10 *SimplexNumerica* Formulas

Formulas establish and calculate mathematical relationships between elements of the spreadsheet. Whereas numeric entries remain the same until you change them, cells defined by formulas are automatically changed to reflect changes in referenced cells— even where there are complex interdependencies among cells.

SimplexNumerica formulas can calculate with numbers, text, logical values, cell references, and other formulas. For example, you can easily calculate the sum of a series of cells, the total of values in a column, a minimum or maximum value within a range, the rounded result of another formula, or the absolute value of a cell entry. Formulas can express complex interdependencies among cells, and they can define constraints on the calculation, such as limits on acceptable values or specific conditions under which a calculation should take place.

Once entered in a cell, formulas are hidden behind the scenes, perform their work in the background, and display only the result of their calculation. To view the formula in a cell, simply select the cell.

SimplexNumerica provides an option that lets you make all formula expression visible. Use Ribbonbar icon *Formulas* and its popup menu *Show Formula in Cells*:



SimplexNumerica also provides a wide array of functions that perform certain tasks. Functions can be used alone or in conjunction with formulas and other functions. *SimplexNumerica* provides many specialized functions in addition to those that are found in typical financial spreadsheets.

6.10.1 Formula Syntax

The general form of an *SimplexNumerica* formula is:

= expression; constraint expression // comment

In the above formula, expression defines the calculations needed to generate the cell's value, constraint expression places limits on acceptable values or the circumstances under which the calculation should take place, and comment is any text you want to attach to the cell.

The expression part of *SimplexNumerica* formulas looks just like an algebraic formula; it contains values and operators that define the relationships between values.

SimplexNumerica uses the following conventions for formulas:

- A formula must begin with an equal (=) sign. When you begin typing into a cell, *SimplexNumerica* automatically assumes that you are typing a formula if you start with one of the following characters: 0 1 2 3 4 5 6 7 8 9 . - =+
- Formulas can have as many as 511 characters. You can type spaces if you wish, but *SimplexNumerica* automatically removes them.

6.10.2 Formula Values

Formulas can contain any or all of the following types of values:

- Numbers, such as 123, -123, 12.3.
- Addresses of single cells, such as A1, D5, Z100.
- Addresses of cell ranges such as B12..G29, A1..D5.
- Absolute cell references denoted with dollar signs before the fixed coordinate (A1, $A1, or A$1), which will not be updated when the referencing cell is moved or copied.
- Functions such as SUM or RADIANS, with their arguments.
- Text surrounded by double quotation marks, such as "The sum is " or "Total".
- User-defined cell names or cell range names, such as TOTALS or PROJECT1.

6.10.3 Formula Operators

Operators are the characters that establish the relationships between values in a formula, such as +, -, *.

SimplexNumerica supports all the arithmetic, Boolean, and logical operators available in the C programming language. Arithmetic operators calculate numeric values; text operators act on strings of text, and logical operators evaluate true/false conditions.

SimplexNumerica also provides two operators—exponentiation (**) and percent (%)— that are not available in the C language. It does not support the C address operators or the operators that have side effects, such as ++.

SimplexNumerica formulas can contain the following operators to define relationship between values:

Table – Operator Precedence Definition

Operator	Definition
% 14	Unary percent
** 13	Exponentiation
+ 12	Unary plus
- 12	Unary minus
~ 12	Bitwise complement (integer)
! 12	Logical not
* 11	Multiplication
/ 11	Division
% 11	Remainder (integer)
+ 10	Addition

Operator	Definition
- 10	Subtraction
<< 9	Shift left (integer)
>> 9	Shift right (integer)
< 8	Less Than
> 8	Greater Than
<= 8	Less Than or Equal
>= 8	Greater Than or Equal
== 7	Equal
!= 7	Not Equal
& 6	Bitwise And, or String Concatenation
^ 5	Bitwise Exclusive-Or (integer)
| 4	Bitwise Or
&& 3	Logical And
|| 2	Logical Or
?: 1	Conditional

In formulas with more than one operator, *SimplexNumerica* evaluates operators in the order of precedence presented above, with highest precedence first. That is, AND/OR/NOT operators are evaluated after inequality operators in a logical expression, and multiplication/division operations are performed before subtraction/addition operations in an arithmetic expression. Operators at the same precedence level are evaluated from left to right.

The precedence of operators can be overridden by using parentheses to explicitly specify the order of evaluation.

Here are some special notes about *SimplexNumerica* operators:

- The operators marked ’(integer)’ in Table 19 automatically convert their operands to integers.
- The & operator performs double duty: as a bit-wise and if the operands are numbers or as a string concatenation operator joining two strings together if the operands are text.

Concatenation is linking two strings together. For example, if you concatenate (A1&A2) the string "John" in cell A1 with the string " Smith" in cell A2, the result is the value "John Smith."

- The % operator also performs double duty: as the percent operator when appended to a number or numeric expression, or as the C-style modulus operator when applied between two integer expressions.
- Operators that define equality/inequality relationships (such as == and <) can be used to compare text strings lexically (alphabetically).

In comparing mixed strings lexically, *SimplexNumerica* considers string operands to be lower than numeric operands.

- The conditional operator returns its second operand if its first operand evaluates True (non-zero) and returns its third operand if it evaluates False, (zero).
- In formulas with conditional operators, the second and third operands may be any type *SimplexNumerica* supports, including ranges. For example, the expression =SUM(A1 ? B1..C20 : C10..D15) returns the sum of B1..C20 if A1 evaluates to nonzero; otherwise it returns the sum of C10..D15.
- *SimplexNumerica* accepts most arithmetic operators used in other spreadsheets like MS Excel, but there are a few differences in syntax and precedence.

6.10.4 Referencing Other Cells in Formulas

The real power of *SimplexNumerica* lies in its ability to calculate relationships among different cells in the spreadsheet by typing the row/column coordinates, or address, in the formula.

To reference a cell by address:

Type the row and column coordinates of the cell in the formula. For example, to reference Row 5 in Column D, type D5.

To reference a contiguous group of cells by address:

Type the row and column coordinates of two cells in opposite corners of the block to be referenced, with two periods (..) between the coordinates. For example, to reference the first five columns and the first five rows of the spreadsheet, type A1..E5.

This differs from Microsoft Excel. The equivalent Excel syntax would be A1:A5.

6.10.5 Cell Referencing in SimplexNumerica

SimplexNumerica differentiates between relative, absolute, and indirect (or current cell) references. The latter is unique to *SimplexNumerica*.

Computed cell references are the result of a function that is itself a cell reference or range reference. See Section 14.6.11.3, “Computed Cell References,” for more information about these.

Relative Reference

Relative references are cell or range references that are interpreted relative to the current position of the cell containing the formula. Relative references are updated whenever a cell is copied or moved, to reflect the new position of the cell. By default, *SimplexNumerica* considers references to be relative.

SimplexNumerica tracks the referenced cell by considering its position relative to the formula cell, not by its address. For example, if the formula in cell A1 references cell B2, *SimplexNumerica* remembers that the referenced cell is one row down and one column right. If you copy the formula in cell A1 to another location (e.g., D17), the formula will reference the cell one row down and one column right of the new location (e.g., E18).

Absolute Reference

Absolute references are references to cells or ranges that remain fixed, no matter where the cell containing the formula is moved or copied. They are preceded with the dollar sign ($) before each coordinate to be fixed, such as $A1, A$1, or A1.

Absolute references remain the same, no matter where you move or copy the original formula. For example, if the formula in cell A1 references cell B2, and you copy the formula in cell A1 to another location (e.g. D17), the formula still references cell B2. To specify an absolute cell address, insert a dollar sign ($) before the address coordinate to be fixed, or before both coordinates if both the row and column coordinates are to be fixed. For example: B2.

To specify all or part of a cell address to be absolute, insert a dollar sign ($) before the address coordinate to remain fixed. For example:

- B5 makes the complete address absolute.
- $B5 makes the column coordinate (B) absolute, the row coordinate (5) relative. ⍰⍰B$5 makes the column coordinate (B) relative, the row coordinate (5) absolute.

Named Ranges

To assign a name to a cell or range of cells, use the script function SetRangeName() command. To reference a cell or range by name, type the pre-assigned name of the cell or cell block into the formula.

Cell ranges are also relative, so when you move a cell range, references in formulas within that range are updated to reflect their new location. To specify an absolute range reference, insert dollar signs ($) before the coordinates in the formula. For example, to make the range A1..D5 absolute, type the reference as A1..D5.

To specify part of a cell range to be absolute, insert dollar signs only before the coordinates to remain absolute. For example, $A1..$D5 will fix the column coordinates of cell references but adjust the row coordinates to reflect the new location.

Refer to Section 14.3.6, “Ranges,” for more information.

Indirect (or Current Cell Offset) Reference

An indirect reference is a unique cell referencing technique in *SimplexNumerica* that allows you to refer to cells by row and/or column offset values relative to the current cell. This can be used anywhere a cell reference is expected:

- function arguments
- formulas
- constraint expressions

Certain expressions within the context of *SimplexNumerica* require a means to express the current cell.

References to cells in the neighborhood of the current cell are made with offset values enclosed in braces ({}) following the pound sign # which identifies the current cell. The offsets tell *SimplexNumerica* where to look, in relation to the current cell, for the cell being referenced. A negative row offset indicates a row above the current row. A negative column offset indicates a column to the left of the current column. Likewise, positive offset numbers for the row and column indicate a row below and a column to the right of the current cell.

The format is as follows:

#{column offset, row offset}

- If you include only one value in the offset, *SimplexNumerica* assumes that it is a column offset. For example, the offset reference #{-1} tells *SimplexNumerica* to look to the column just left of the current cell.
- The offset values may be constants or expressions.

Examples:

- #{0,-1} refers to the cell above the current cell.
- #{-2} refers to the cell two columns to the left of the current cell.
- #{1} refers to the cell to the right of the current cell.
- #{0,1} refers to the cell below the current cell.
- CSUM(C4..C100, #{-1} == "Joe") calculates the sum of all the values in the range C4..C100 for which the cell in the column to the left contains the string "Joe".
- CCOUNT(C4..C100, # #{0,-1}) counts all the cells in the range C4..C100 whose value is greater than the contents of the cell immediately above.
- XVALUE("master.xs3", #) returns the value of the same cell reference in which this function is stored from the sheet indicated.
- /verb/#-1+2/ adds 2 to the cell value from the cell to the left.

6.10.6 Constraint Expressions

Constraints are limitations or conditions placed on the variables in your spreadsheet. They are expressed as algebraic statements appended to formulas. You can attach a constraint expression to any formula, by typing a semicolon (;) and the constraint conditions after the formula.

Constraint expressions establish conditions under which a formula operates or boundaries for valid results of the formula. Constraint expressions may be simple equality/inequality relationships, or they can be arbitrary formulas. Any valid *SimplexNumerica* expression that returns a numeric value is also a valid constraint expression. However, unlike the expression that defines a cell value, a constraint expression can reference the cell in which it resides, using the symbol #.

For example, the formula =A1 + A2 ; #>2 && #<=B5 || #==C7 means "the value of the current cell is the sum of cells A1 and A2, and that value must be either greater than 2 and less than or equal to the value of cell B5, or equal to the value of cell C7".

Constraint expressions are used, for example, in the conditional statistical functions. The benefit of constraint expressions is maximized when combined with current cell reference support (#) as indicated in the above example.

6.10.7 Explicit Dependency

There may be instances where you need to force a recalculation when certain cell values change, when there is no implicit dependency in the formula that would trigger an automatic recalculation. This explicit dependency option is indicated by appending a backslash (\) to the end of the dependent formula. For example, the formula =SUM(A1..A20)\D50 instructs *SimplexNumerica* to recalculate SUM(A1..A20) whenever the contents of D50 change.

This feature is particularly important when you have a constraint expression containing an offset reference that produces a cell reference outside the cell range referenced in a dependent formula. Under these circumstances, Automatic Recalculation would not necessarily be triggered. In the above example, CCOUNT(C4..C100, # #{0,-1}) counts all the cells in the range C4..C100 whose value is greater than the contents of the cell immediately above.

In order for C4 to be evaluated, it must be compared to C3 - which is not part of the explicit range, C4..C100. Without indicating an explicit dependency, C4 would never be evaluated properly. So, in this case, we would indicate the dependency as follows:

CCOUNT(C4..C100, # #{0,-1})\C3..C99 which tells *SimplexNumerica* to recalculate whenever any cell in the range C3..C99 changes.

For more information about explicit dependency and computed cell references, see

Section 14.6.11.3, “Computed Cell References.”

6.11 Built-in Functions

SimplexNumerica functions are predefined formulas supplied with the program that perform the work of many formulas or perform special functions that cannot be achieved by formulas, such as manipulating text strings. They offer a shortcut approach to accomplishing the work of long, complex formulas. Mathematical and statistical functions are often used to sum a column of numbers, compute an average, determine a minimum or maximum value, or round the results of a formula. Other functions are used for more specialized purposes such as computing the future value of an investment or the product of multiplying one cell range by another range. Some functions perform calculations that arithmetic operators cannot handle such as text-string manipulations.

An argument is a parameter appended to a function statement, specifying the values that *SimplexNumerica* should use in calculating the function. In the syntax statement for a function, the argument list is the list of arguments the function should use for its calculation.

SimplexNumerica functions fall into the following categories:

- Mathematical function—Performs calculations with numeric values as arguments.
- Statistical function—Performs aggregation and counting operations on a group of values expressed as a list of arguments.
- Conditional Statistical function—Operates much like statistical aggregation functions, except that the last argument is a constraint expression that *SimplexNumerica* evaluates for each cell in the argument list. Only cells that meet constraint criteria are included in the calculation.
- String functions—Act on strings of text, rather than on numeric values.
- Logical function—Evaluates conditions on a purely true/false basis, returning the value 0 if the condition is False and the value 1 if the condition is True.
- Digital Logical function—Returns the values 0, 1 or -1 (unknown) based on the value of its arguments. Digital logical functions evaluate the integer portion of a value. Any value not equal to 0 or 1 is considered unknown.
- Financial function—Performs a common financial calculation, such as future value of an annuity at a given interest rate.

- Date and Time function—Returns a value corresponding to the specified date, month, year, hour, minute or second. It can also be the current system time and date.
- Miscellaneous function—Performs a variety of calculations such as returning a reference to a specific cell or range.
- Embedded Tools—Have the ability to return data in a matrix, not just the resident cell.

6.11.1 Mathematical Functions

Mathematical functions perform calculations such as determining absolute value, finding the integer portion of a number, or establishing the value of a constant. Although you could accomplish these tasks with a formula, using a function saves time and trouble.

SimplexNumerica also provides a full range of trigonometric functions including sine, cosine, tangent, arc sine, hyperbolic sine, hyperbolic arc sine, as well as vector and matrix arithmetic and manipulation.

Mathematical functions perform calculations with numeric values as arguments, returning numeric values.

6.11.2 Statistical Functions

Statistical functions perform aggregation operations such as calculating means, minimums, maximums, and averages.

SimplexNumerica also provides more sophisticated statistical test functions that perform operations on a group of values expressed as a list of arguments. These include the F-test, t-tests, correlation coefficient, deviations, and all common averages.

Statistical functions return numeric values.

6.11.3 Conditional Statistical Functions

Conditional statistical functions operate much like statistical aggregation functions, except that the last argument is a constraint expression that *SimplexNumerica* evaluates for each cell in the argument list. Only cells that meet constraint criteria are included in the calculation. The constraint expression may be any *SimplexNumerica* expression that evaluates to a numeric result.

Conditional statistical functions return a numeric value.

6.11.4 String Functions

String functions manipulate and evaluate character strings. For example, string functions can return the length of a string, find the first occurrence of a string in a range, change a string from uppercase to lowercase and vice versa, or replace one string with another.

String functions return strings or numeric values.

6.11.5 Logic Functions

Logic functions return one value if an argument meets certain criteria, another value if it does not. Logic functions are used as an adjunct to conditional statements.

Logic functions return the value 1, 0, or a value.

6.11.6 Digital Logic Functions

Digital logic functions perform digital logic operations such as AND, OR, NOT, etc.

Digital logic functions return the values 0, 1, or -1 (unknown). Any value whose integer portion is not equal to 0 or 1 is considered unknown. Unknown input values may cause unknown output values.

6.11.7 Financial Functions

Financial functions perform common financial calculations, such as calculating the future value of an annuity at a given interest rate, straight-line depreciation, double-declining depreciation, or the payment term for a given investment. The financial functions in *SimplexNumerica* cover annuities, cash flows, assets, bonds, and Treasury Bills.

Financial functions are most useful for solving cash flow calculations where you know all but one variable. For example, if you know the present value of an investment, interest rate, and periodic payment, you can use the FV function to calculate the future value of the investment. If you know the future value and other variables, but need to know the present value, you can use the PV function.

Many financial functions require specifying a Day Count Basis. A Day Count Basis indicates the way in which the days in a month and the days in a year are to be counted. Most of the financial functions in securities involve four different Day Count Bases: 30/360, actual/actual, actual/360, and actual/365.

30/360 Day Count Basis assumes 30-day months and 360-day years (12 months x 30 days). *SimplexNumerica* also follows the "End-of-Month" rule which assumes that a security pays interest on the last day of the month and will always make its interest on the last day of the month. Special rules are followed when calculating the days between two dates on 30/360 Day Count Basis.

For example, let Start_Date = D1/M1/Y1, End_Date = D2/M2/Y2.

- If D1=31, *SimplexNumerica* uses 30 for D1.
- If D2=31, *SimplexNumerica* uses 31, unless D1=30 or D1=31. In this case, *SimplexNumerica* uses 30.
- If D1 is the last day of February (D1=28 or 29 in a leap year), *SimplexNumerica* uses 30 for D1.
- If D2 is the last day of February (D2=28 or 29 in a leap year) and D1 is also the last day of February, *SimplexNumerica* uses 30 for D2.

The special arguments used by *SimplexNumerica* financial functions are defined in the next Table.

Function	Purpose
interest rate	The interest rate to be used in the calculations. The rate may be specified as annual, monthly, or quarterly, but it must agree with the increment you use for periods. By default, the interest rate is an annual rate.
present value	The present value of an investment, representing the amount already received from or committed to an investment.

Function	Purpose
period	The number of periods over which the loan, investment, or depreciation is to be calculated. The periods may be defined in months, quarters, or years, but must agree with the increment used to define interest rate.
future value	The future value of an investment, given a certain present value, interest rate, and number of periods.
cost	The original cost of a depreciable capital asset.
salvage value	The remaining value of a capital asset after the depreciation period has expired.
allowable life	The allowable life of a depreciable item.
yield	The interest rate that will make the present value of the expected future cash flows equal to the price of the financial instrument.
price	The present value of the expected future cash flows where the discount rate is equal to the yield of the financial instrument.
coupon rate	The annual coupon rate of a security.
frequency	The number of coupon payments in a year.
basis	The day count basis to be used in calculation.

Functions related to fixed income securities usually require special dates as arguments: issue date, settlement date, first coupon date, last coupon date, maturity date of a security. When specified, the following constraints should be followed:

- issue settlement maturity
- issue first coupon maturity
- issue last coupon maturity

6.11.8 Date and Time Functions

Date and time functions return values corresponding to the specified date, month, year, hour, minute, or second. You can also use date/time functions to enter the current system time and date in a cell. These functions open up many possibilities for managing accounts receivable and calculating test times.

SimplexNumerica internally stores date and time information using not the same convention as other spreadsheet programs, because the Julian Day has a longer period of time.

- Dates and Times are represented as a double value equal to the Julian Day (see https://en.wikipedia.org/wiki/Julian_day).

6.11.9 Miscellaneous Functions

Miscellaneous functions perform a variety of calculations, such as returning a reference to specific cells or ranges or returning the Nth argument from a list of arguments.

6.11.10Embedded Tools

Embedded tools are a set of functions in *SimplexNumerica* that have the ability to return data in a matrix, not just the resident cell. These functions make non-scalar operations such as matrix multiplication and "live" recalculation as easy to use as an ordinary spreadsheet function.

Embedded tools are a powerful feature in *SimplexNumerica*. Their power derives in part from their ability to return a set of data, not just a single value. This function makes non-scalar operations such as matrix multiplication and "live" recalculation as easy to use as an ordinary spreadsheet function.

Embedded tools store values in a group of adjacent cells. These adjacent cells are set to constant formulas, with explicit dependencies on their neighboring cells. For example, an embedded tool in cell B2 might produce the formula =1.3459/B2 in cell B3. This formula indicates that the cell currently contains the constant 1.3459 but that its value depends on the contents of cell B2 (the cell containing the embedded tool).

This notion of explicit dependencies is important for recalculation. It guarantees that any cell that references B3 will not be recalculated until after cell B2 is recalculated. This ensures that data generated by the embedded tool is always current.

Embedded tools look like normal functions, and they can be copied, moved and formatted just as any other formula in the spreadsheet. However, you must follow one important guideline: DO NOT combine embedded tools with other embedded tools in a single formula. For example, the formula INVERT(MMUL(A1..C4,F1..I3)) is not allowed.

6.11.11Using SimplexNumerica Built-in Functions

You enter a function in a cell in the same way you enter a formula or any other entry, with a few additional guidelines.

- Type in the function name. *SimplexNumerica* recognizes the string as a function. Function names are abbreviations that indicate what the function does. For instance, ABS computes absolute value, ROUND rounds to the specified number of places, and AVG computes the average of a list of arguments. Function names may be preceded with an '' sign, but this is not required.
- After typing the function name, enter arguments in parentheses. Most functions use one or more arguments to define the task to be performed. For example, the AVG function averages the value of two or more arguments. The LENGTH function returns the length of an argument that is a character string.
- Use only the arguments required by the function, in the exact order specified in the function syntax. If you enter other arguments or enter them in the wrong order, *SimplexNumerica* will misinterpret their meaning or return an error message.

- All the function names in this chapter are typed in uppercase letters, but you can enter them in uppercase or lowercase for your entries.

Arguments

Arguments specify the values the function should use in its calculations. The number of arguments, their types, and their formats varies from one function to another. Arguments are usually numeric values, cell or range references, or string values. Most functions have at least one argument; a few have none.

The following table shows different types of arguments used in *SimplexNumerica* functions.

Argument	**Example**
Numeric Value	123
Address of a cell	A10
Address of a range	F9..F99
String Value	"Quarterly Report"

Using Operators with Functions

The result of a function depends on the order in which *SimplexNumerica* handles the calculations.

Computed Cell References

Computed cell references are the result of a function that is itself a cell reference or range reference.

Several *SimplexNumerica* functions such as CELLREF and RANGEREF return a result that is itself a cell reference or range reference. This is a powerful facility, but it must be used with caution because *SimplexNumerica* cannot take these indirect references into account when determining the order of recalculation. The same caution applies to constraint expressions used in conditional statistical functions. As a rule, cells that are indirectly referenced by a function are not automatically recalculated. *SimplexNumerica* provides a special construct to force a recalculation, referred to as an explicit dependency.

SimplexNumerica does not recalculate the spreadsheet unless explicit dependencies have been changed, so you may need to force recalculation if you change the value of a cell that is referenced only indirectly through a function.

For example, suppose you want to count the numeric values in the range C3..J100 that fall within the limits specified in cells A1 and A2. The *SimplexNumerica* formula to compute this is CCOUNT(C3..J100,#A1 && #<A2).

This formula will correctly count the numeric values in the range C3..J100. However, if you change the value in A1, *SimplexNumerica* will not automatically recalculate the result, because A1 is referenced only indirectly through the constraint expression.

- To force *SimplexNumerica* to recalculate the entire spreadsheet you should call the script function Recalc(). There is also a Recalculate menu in the Ribbonbar that calls Recalc().
- You can also force *SimplexNumerica* to do a partial recalculation with respect to that cell, edit the cell and append a blank and press the [Return] key on the cell containing the CCOUNT formula.

- You can also use explicit dependencies to circumvent the limitation described above, if you entered the formula below in the form CCOUNT(C3..J100,#A1 && #<A2)\A1\A2.

SimplexNumerica would take into account the dependencies on A1 and A2 and update the spreadsheet just as you expect.

Another approach is to construct the condition string with an expression that references the cells directly. For example, CCOUNT(C3..J100, STRCAT("#",A1,"&_<",A2)).

In this example, A1 and A2 are directly referenced and thus will properly trigger recalculation.

6.12 Quick-Reference Guide to Built-in Functions

6.12.1 Mathematical Functions

The following table lists the mathematical functions that are supported:

Function	Description
ABS(X)	The absolute value of X.
ACOS(X)	The arc cosine of X.
ASIN(X)	The arc sine of X.
ATAN(X)	The 2-quadrant arc tangent of X.
ATAN2(X, Y)	The 4-quadrant arc tangent of Y/X.
CEIL(X)	The smallest integer greater than or equal to X.
COS(X)	The cosine of X.
COSH(X)	The hyperbolic cosine of X.
DEGREES(X)	Converts the angle expressed in radians to degrees ( ).
DET(M)	The determinant of the matrix range M, which must be a square matrix.
DOT(R1, R2)	The dot product of the vectors R1 and R2.
EXP(X)	e raised to the X power.
FACT(N)	The value of N!.
FLOOR(X)	The largest integer less than or equal to X.
FRAC(X)	The fractional portion of X.
GAMMA(X)	The value of the gamma function evaluated at X.
GRAND	A 12th-degree binomial approximation to a Gaussian random number with zero mean and unit variance.
INT(X)	The integer portion of X.

LN(X)	The natural log (base e) of X.
LNGAMMA(X)	The log base e of the gamma function evaluated at X.

Function	Description
LOG(X)	The log base 10 of X.
LOG10(X)	The log base 10 of X.
LOG2(X)	The log base 2 of X.
MOD(X, Y)	The remainder of X/Y.
MODULUS(X, Y)	The modulus of X/Y.
PI	The value of pi.
POLY(X, ...)	The value of an Nth-degree polynomial in X.
PRODUCT(X, ...)	The product of all the numeric values in the argument list.
RADIANS(X)	Converts the angle expressed in degrees to radians ( ).
RAND	A uniform random number on the interval (0,1).
ROUND(X, n)	X rounded to n number of decimal places (0 to 15).
SIGMOID(X)	The value of the sigmoid function.
SIN(X)	The sine of X.
SINH(X)	The hyperbolic sine of X.
SQRT(X)	The positive square root of X.
SUMPRODUCT(R1, R2)	The dot product of the vectors R1 and R2, where R1 and R2 are of equal dimension.
TAN(X)	The tangent of X.
TANH(X)	The hyperbolic tangent of X.

TRANSPOSE(M)	The transpose of matrix M.
VECLEN(...)	The square root of the sum of squares of its arguments.

6.12.2 Statistical Functions

The following table lists statistical functions that are supported:

Function	Purpose
AVG(...)	The average (arithmetic mean) of its arguments.
CORR(R1, R2)	Pearson's product-moment correlation coefficient for the paired data in ranges R1 and R2.
COUNT(...)	A count of its non-blank arguments.
F(M, N, F)	The integral of Snedecor's F-distribution with M and N degrees of freedom from minus infinity to F.
ERF(L[, U])	Error function integrated between 0 and L; if U specified, between L and U.
ERFC(L)	Complementary error function integrated between L and infinity.
FORECAST(...)	Predicted Y values for given X.
FTEST(R1, R2)	The significance level ( ) of the two-sided F-test on the variances of the data specified by ranges R1 and R2.
GMEAN(...)	The geometric mean of its arguments.
HMEAN(...)	The harmonic mean of its arguments.
LARGE(R, N)	The Nth largest value in range R.
MAX(...)	The maximum of its arguments.
MEDIAN(...)	The median (middle value) of the range R1.
MIN(...)	The minimum of its arguments.
MODE(...)	The mode or most frequently occurring value.
MSQ(...)	The mean of the squares of its arguments.

PERCENTILE(R, N)	The value from the range R that is at the Nth percentile in R.
PERCENTRANK(R, N)	The percentile rank of the number N among the values in range R.
PERMUT(S, T)	The number of T objects that can be chosen from the set S, where order is significant.
PTTEST(R1, R2)	The significance level ( ) of the two-sided T-test for the paired samples contained in ranges R1 and R2.
QUARTILE(R, Q)	The quartile Q of the data in range R.
RANK(E, R[, O])	The rank of a numeric argument E in the range R.

Function	**Purpose**
SSQ(...)	The sum of squares of its arguments.
RMS(...)	The root of the mean of squares of its arguments.
SMALL(R, N)	The Nth smallest number in range R.
SSE(...)	The sum squared error of its arguments. It is equivalent to VAR(...) COUNT(...).
STD(...)	The population standard deviation (N weighting) of its arguments.
STDS(...)	The sample standard deviation (N-1 weighting) of its arguments.
SUM(...)	The sum of its arguments.
T(N, T)	The integral of Student's T-distribution with N degrees of freedom from minus infinity to T.
TTEST(R, X)	The significance level of the two-sided single population Ttest for the population samples contained in range R.
TTEST2EV(R1, R2)	The significance level ( ) of the two-sided dual population T-test for ranges R1 and R2, where the population variances are equal.
TTEST2UV(R1, R2)	The significance level ( ) of the two-sided dual population T-test for ranges R1 and R2, where the population variances are not equal.

VAR(...)	The sample variance (N weighting) of its arguments.
VARS(...)	The sample variance (N-1 weighting) of its arguments.
VSUM(...)	The visual sum of its arguments, using precision and rounding of formatted cell values.

6.12.3 *Conditional Statistical Functions*

The following table lists conditional statistical functions that are supported:

Function	Purpose
CAVG(..., C)	Conditional average.
CCOUN(..., C)	Conditional count.
CMAX(..., C)	Conditional maximum.
CMIN(..., C)	Conditional minimum.
CSTD(..., C)	Conditional sample standard deviation (N weighting).

Function	Purpose
CSTDS(..., C)	Conditional sample standard deviation (N-1 weighting).
CSUM(..., C)	Conditional sum.
CVAR(..., C)	Conditional population variance (N weighting).
CVARS(..., C)	Conditional population variance (N-1 weighting).

6.12.4 *String Functions*

The following table lists string functions that are supported:

Function	Purpose
CHAR(N)	The character represented by the code N.
CLEAN(S)	The string formed by removing all non-printing characters from the string S.
CODE(S)	The ASCII code for the first character in string S.
EXACT(S1, S2)	Returns true (1) if string S1 exactly matches string S2, otherwise returns 0.

FIND(S1, S2, N)	The index of the first occurrence of S1 in S2.
HEXTONUM(S)	The numeric value for the hexadecimal interpretation of S.
LEFT(S, N)	The string composed of the leftmost N characters of S.
LENGTH(S)	The number of characters in S.
LOWER(S)	S converted to lower case.
MID(S, N1, N2)	The string of length N2 that starts at position N1 in S.
NUMTOHEX(X)	The hexadecimal representation of the integer portion of X.
PROPER(S)	The string S with the first letter of each word capitalized.
REGEX(S1, S2)	Returns true (1) if string S1 exactly matches string S2; otherwise returns false (0). Allows "wildcard"' comparisons by interpreting S1 as a regular expression.
REPEAT(S, N)	The string S repeated N times.

Function	**Purpose**
RIGHT(S, N)	The string composed of the rightmost N characters of S.
STRCAT(...)	The concatenation of all its arguments.
STRING(X, N)	The string representing the numeric value of X, to N decimal places.
STRLEN(...)	The total length of all strings in its arguments.
TRIM(S)	The string formed by removing spaces from the string S.
UPPER(S)	The string S converted to upper case.
VALUE(S)	The numeric value represented by the string S; otherwise 0 if S does not represent a number.

6.12.5 Logic Functions

The following table lists the supported logic functions:

Function	Purpose
FALSE	The logical value 0.
FILEEXISTS(S)	I if file S can be opened for reading; otherwise 0.
IF(X, T, F)	The value of T if X evaluates to I, or F if X evaluates to 0.
ISERROR(X)	Returns I if X "contains" an error, otherwise 0.
ISNUMBER(X)	I if X is a numeric value; otherwise 0.
ISSTRING(X)	I if X is a string value; otherwise 0.
TRUE	The logical value I.
AND(...)	0 if any arguments are 0; I if all arguments are I; otherwise -I.
NAND(...)	0 if all arguments are I; I if any arguments are 0; otherwise -I.
NOR(...)	0 if any arguments are I; I if all arguments are 0; otherwise -I.
OR(...)	0 if all arguments are 0; I if any arguments are I; otherwise -I.
XOR(...)	-I if any of the arguments are not 0 or I; otherwise 0 if the total number of arguments with the value I is even; I if the total number of arguments with the value I is odd.

6.12.6 Financial Functions

The following table lists the supported financial functions:

Function	Description
COUPDAYBS(S, M, F[, B])	The number of days between the beginning of the coupon period to the settlement date.

ACCRINT(I, Ft, S, R, P, F[, B])	Accrued interest for a security that pays periodic interest.
ACCRINTM(I, S, R, P[, B])	Accrued interest for a security that pays interest at maturity.
COUPDAYS(S, M, F[, B])	The number of days in the coupon period that the settlement date is in.
COUPDAYSNC(S, M, F[, B])	The number of days between the settlement date and the next coupon date.
COUPNCD(S, M, F[, B])	The next coupon date after the settlement date.
COUPNUM(S, M, F[, B])	The number of coupon payments between the settlement date and maturity date.
COUPPCD(S, M, F[, B])	The previous (most recent) coupon date before the settlement date.
CTERM(R, FV, PV)	The number of compounding periods for an investment.
CUMIPMT(R, NP, PV, S, E, T)	The cumulative interest on a loan between start period S and end period E.
CUMPRINC(R, NP, PV, S, E, T)	The cumulative principal paid on a loan between start period S and end period E.
DB(C, S, L, P[, M])	Fixed-declining depreciation allowance.
DDB(C, S, L, N)	Double-declining depreciation allowance.
DISC(S, M, P, R[, B])	The discount rate for a security.

Function	**Description**
DOLLARDE(FD, F)	Converts a dollar amount expressed as a fraction form into a decimal form.
DOLLARFR(DD, F)	Converts a dollar amount expressed as a decimal form into a fraction form.
DURATION(S, M, R, Y, F[, B])	The Macauley duration of a security assuming $100 face value.
EFFECT(NR, NP)	Returns the effective annual interest rate.
FV(P, R, N)	Future value of an annuity.

FVSCHEDULE(P, S)	The future value of an initial investment after compounding a series of interest rates.
INTRATE(S, M, I, R[, B])	The interest rate for a fully invested security.
IPMT(R, P, NP, PV, FV[, T])	The interest payment for a specific period for an investment based on periodic, constant payments, and a constant interest rate.
IRR(G, F)	The internal rate of return on an investment. (See also XIRR and MIRR.)
MDURATION(S, M, R, Y, F[, B])	The modified Macauley duration of a security assuming $100 face value.
MIRR(CF, FR, RR)	The modified internal rate of return for a series of periodic cash flows.
NOMINAL(ER, NP)	The nominal annual interest rate.
ODDFPRICE(S, M, I, FC, R, Y, RD, F[, B])	The price per $100 face value of a security with an odd (short or long) first period.
ODDFYIELD(S, M, I, FC, R, PR, RD, F[, B])	The yield per of a security with an odd (short or long) first period.
PMT(PV, R, N)	The periodic payment for a loan.
PPMT(R, P, NP, PV, FV, T)	The payment on the principal for a specific period for an investment based on periodic, constant payments, and a constant interest rate.
PRICE(S, M, R, Y, RD, F[, B])	The price per $100 face value of a security that pays periodic interest.
PRICEDISC(S, M, D, RD[, B])	The price per $100 face value of a discounted security.
PRICEMAT(S, M, I, R, Y[, B])	The price per $100 face value of a security that pays interest at maturity.
PV(P, R, N)	The present value of an annuity

Function	**Description**
RATE(FV, PV, N)	The interest rate required to reach future value FV.

RECEIVED(S, M, I, D, [, B])	The amount received at maturity for a fully vested security.
SLN(C, S, L)	The straight-line depreciation allowance.
SYD(C, S, L, N)	The "sum-of-years-digits" depreciation allowance.
TBILLEQ(S, M, D)	The bond-equivalent yield (BEY) for a Treasury Bill.
TBILLYIELD(S, M, D)	The yield on a Treasury bill.
TERM(P, R, FV)	The number of payment periods for an investment.
VDB(C, S, L, S, E)	Fixed-declining depreciation allowance between two periods.
XIRR(G, V, D)	Internal rate of return for a series of cash flows with variable intervals.
XNPV(R, V, D)	Returns the net present value for a series of cash flows with variable intervals.
YIELD(S, M, R, PR, RD, F[, B])	Yield of a security that pays periodic interest.
YIELDMAT(S, M, I, R, PR[, B])	Annual yield of a security which pays interest at maturity.

6.12.7 Date and Time Functions

The following table lists the supported date and time functions:

Function	Description
DATE(Y, M, D)	The date value for year Y, month M, and day D.
DATEVALUE(S)	The corresponding date value for a given string S.
DAYS360(S, E)	The number of days between two dates, based on a 30/360 day count system.
DAY(DT)	The day number in the date/time value DT.

EOMONTH(S, M)	The date/time value representing the last day of the month M months after S, if M is positive, or M months before if M is negative.
HOUR(DT)	The hour value (0-23) of date/time value DT.
MINUTE(DT)	The minute value (0-59) of date/time value DT.
MONTH(DT)	The number of the month in date/time value DT.
NETWORKDAYS(S, E[, H])	The number of whole working days, starting at S and going to E, excluding weekends and holidays.
NOW	The date/time value of the current system date and time.
SECOND(DT)	The seconds value (0-59) of the date/time value DT.
TIME(H, M, S)	The time value for hour H, minute M, and second S.
TIMEVALUE(S)	The corresponding time value for a given string value S.
TODAY	The date value of the current system date.
WEEKDAY(D)	The integer representing the day of the week on which the day D falls. 1 is Sunday, 7 is Saturday.
WORKDAY(S, D[, H])	The day that is D working days after S, if D is positive, or before S, if D is negative, excluding weekends and all holidays specified as dates in range H.
YEAR(DT)	The year value of date/time value DT.
YEARFRAC(S, E[, B])	The portion of the year represented by the number of days between start date (S) and end date (E).

6.12.8 Miscellaneous Functions

The following table lists miscellaneous supported functions:

Function	**Purpose**
CELLREF(N1, N2)	A reference to the cell in column N1 and row N2.
CHOOSE(N, ...)	The Nth argument from the list.
COL(C)	The column address of the cell referenced by C.
COLS(R)	The number of columns in the specified range R.

HLOOKUP(X, S, R)	The value of the cell in range S that is R number of rows beneath X.
INIT(X1, X2)	The first argument on the first recalculation pass and the second argument on all subsequent recalculation passes when *SimplexNumerica* is performing iterative calculations.
INTERP2D(R1, R2, N)	The interpolation value for a 2-dimensional vector.
INTERP3D(R, X, Y)	The interpolation value for a 3-dimensional vector.
MATCH(V, R[, T])	The relative position in range R of value V based on positioning criteria T.
N(R)	The numeric value of the top left cell in range R.
RANGEREF(N1, N2, N3, N4)	A reference to the range defined by coordinates N1 through N4.
ROW(C)	The row address of the cell referenced by C.
ROWS(R)	The number of rows in the specified range R.
S(R)	The string value of the top left cell in range R.
VLOOKUP(X, S, C)	The value of the cell in range S that is C number of columns to the right of X.

Hint

Some SimplexNumerica functions return a result that is a range or cell reference. SimplexNumerica does not include these indirect references in determining the pattern of recalculation. Plan carefully before using these functions.

6.12.9 Embedded Tools

The following table lists supported embedded tools:

Function	**Description**
DFT(R)	The Discrete Fourier Transform of the range R.
EIGEN(M)	The eigenvalues of the matrix M.
FFT(R)	The Discrete Fourier Transform of the range R using a fast Fourier Transform algorithm.

Function	Description
FREQUENCY(R, B)	Returns a frequency distribution for values R with a set of intervals B.
INVDFT(R)	The inverse of the Discrete Fourier Transform of the range R.
INVERT(M)	The inverse of matrix M.
INVFFT(R)	The inverse of the Discrete Fourier Transform of the range R using a fast Fourier Transform algorithm.
LINFIT(X, Y)	The straight line least squares fit. This function is equivalent to POLYFIT(X, Y, 1).
LLS(A, Y)	The linear least squares solution X to the over-determined system of equations AX=Y.
MMUL(M1, M2)	The product of multiplying matrix M2 by matrix M1.
PLS(X, Y, d)	Analyzes the least squares polynomial model Y=P(X), where P is a polynomial of degree d.
POLYCOEF(X, Y, d)	The least squares coefficients for the polynomial fit Y=P(X), where P is a polynomial of degree d.
TRANSPOSE(M)	The transpose of matrix M.
TREND(NX, KX, KY)	The y values for new x values given existing x and y values.

Hint

Embedded tools should not be contained within other functions or arithmetic operations in a single formula. For example, the formula INVERT(MMUL(A1..C4,F1..I3)) is not allowed. You may, however, copy, move and format embedded tools just as any other function.

6.13 Error Messages

SimplexNumerica checks for a variety of errors. Depending on the error type, the most recent error message is displayed either inside the affected cell(s), on the Message Line, or inside the *SimplexNumerica* Message dialog box.

6.13.1 Types of Errors

Errors in Functions

Errors that occur inside functions are reported along with the name of the function in which the error occurred.

Formula Syntax Errors

These errors occur only when you are typing in a formula. When you finish entering the formula, *SimplexNumerica* will attempt to read the formula and convert it to an internal representation. If it is unable to do so, it continues to display the erroneous formula, switches into "edit mode", places the text cursor at the beginning of the text that it had difficulty parsing, and displays the error message.

The problem must be corrected before *SimplexNumerica* can continue.

Formula Evaluation Errors

Formula evaluation error occurs when *SimplexNumerica* reads in a formula and converts it into its internal formula representation, but is not able to evaluate the formula and produce a correct numeric or string formula. In some cases, the formula has been entered incorrectly, for example, an operand or parenthesis is missing. In other cases, an error has occurred as a result of computation that cannot be handled properly by the computer's floating point hardware, or there is an error condition in a cell or range that is referenced in the context of this formula. Errors can also occur in the evaluation of *SimplexNumerica* built-in functions.

6.13.2 Error Messages Reference

Use this handy reference table to look up the meaning of error messages.

Table – Alphabetized Summary of Error Messages

Error Message	Meaning
argument must be an integer	FACT has been passed a non-integer argument.
argument not a cell or range	Function has been passed an argument that is neither a cell nor a range.
argument out of range	An argument to a function is not within the correct range for the function and its other arguments.
arguments must be numeric	The function requires numeric arguments, which may be literal numbers, formulas which return numeric values, or references to cells containing numeric values.
arguments must be positive	The arguments in this function must be all positive values.
can not parse condition string	*SimplexNumerica* has encountered a malformed conditional expression.

cannot find interpolation	INTERP2D or INTERP3D is unsuccessful in finding interpolated values.
cash flow series must be a range	NPV and MIRR require that their cash flow series must be a range, which must represent a single column or row.
cash flow series must be single column or row	NPV and MIRR require that their cash flow series must be a range, which must represent a single column or row.
cell operand contains error condition	A cell which is referenced from the cell in which the error occurs contains an error condition.
cell reference out of range	A cell reference has been made that is outside the range A1..FAN32767
coefficient matrix has linearly dependent columns	The existence of a unique solution to a linear least squares (LLS) problem, Ax=b, requires that the columns of A are linearly independent.
column offset out of range	The third argument to the VLOOKUP function specifies an offset that is less than 0 or is greater than the width of the range specified in the second argument.
constraint check not supported with "As Needed"	Constraint checking is not supported when the recalculation is set to "As Needed".
contains an error indicator	A cell in one or more of the data ranges for a graph contains an error condition. The error condition must be resolved before *SimplexNumerica* can plot the graph.
could not find real root	IRR could not find a real root. This suggests that the data given to IRR is probably wrong.
count less than zero	User has passed a negative argument to a function that requires a count, for example, with LEFT, it is impossible to take the -2 leftmost characters of a string.
data set size must be = 3	LINFIT and LINCOEF require a data set of size 3 or larger.
data set size must be =	PLS, POLYFIT, and POLYCOEF require that the data set size be polynomial degree + 2
date series must be single column or row	XIRR and XNPV require the argument D (date series) to be a single column or single row.
decimal places out of range	STRING only takes a decimal place argument between 0 and 15.
degrees of freedom must be 0	F and T require degrees of freedom greater than zero, as "degrees of freedom"' is mathematically undefined for zero or less.
dimension must be power of 2	FFT and INVFFT require matrices whose dimensions are powers of two. The somewhat slower functions DFT and INVDFT, respectively, are equivalent functions which do not share this requirement.
divide by zero	An attempt has been made to divide by zero. Note that *SimplexNumerica* considers cells that are empty or contain text strings to have the value zero in the context of a numerical calculation.
does not accept arguments	Several *SimplexNumerica* functions, including PI, TRUE, FALSE, RAND, and GRAND, do not accept any arguments.
domain is -1 < x < 1	ATANH only takes arguments between -1 and 1, exclusive.
domain is -1 <= x <= 1	ACOS and ASIN only take arguments between -1 and 1, inclusive.
domain is 0 <= x <= 170	FACT only takes arguments between 0 and 170, inclusive. (Most platforms)
domain is x 0	LN, LOG2, LOG, GAMMA, and LNGAMMA only take arguments greater than zero.
domain is x = 1	ACOSH only takes arguments greater than or equal to 1.

"End Period" must be >=	
1	CUMIPMT and CUMPRINC require the argument E (end period) to be greater than or equal to 1.
"End Period'' must be >=	
"Start Period"	CUMIPMT, CUMPRINC and VDB require the argument E (end period) to be greater than or equal to S (start period).
ending line with a \	The \\ is an escape sequence introducer, which should be followed by another character for interpretation, but the string ended prematurely.
ending line with a superscript command	When displaying text in the context of graphics, a ^ is a superscript introducer. Like y^2 means "y squared". This message occurs when a ^ occurs at the end of the string.
ending line with subscript	
command	When displaying text in the context of graphics, an '_' is a subscript introducer. Like y_2 means "y subscript 2". This message occurs when an '_' occurs at the end of the string.
error in regular expression	An error occurred while parsing the regular expression used in a search or extract operation, or while executing REGEX or MATCH.
	expected the right hand
side of a range here	
expected to find [something] here	There was a parsing error. The cursor will be placed in the edit window in edit mode. Read the documentation for the function and correct the error.
expecting a function	There is something wrong with the formula you have entered on the edit line. The parser was expecting to find a function name at the point indicated by the cursor position.
expecting an operand	There is something wrong with the formula you have entered on the edit line. The parser was expecting to find an operand at the point indicated by the cursor position.
expecting an operator	There is something wrong with the formula you have entered on the edit line. The parser was expecting to find an operator at the point indicated by the cursor position.
extraneous operands	There is something wrong with the formula you have entered on the edit line. The parser finds an extraneous operand at the point indicated by the cursor position.
F must be >= 0	The third argument to F must be greater than or equal to 0.
first argument must be numeric	NPV and CHOOSE require that their first argument be numeric.
floating exception	A floating-point arithmetic hardware exception occurred during the computation of the function or expression. This means that the calculations resulted in a number out of the range that the computer hardware is able to represent.
found something unexpected here	*SimplexNumerica* has found something it doesn't understand in an expression.
"Fraction" must be > = 1	DOLLARDE and DOLLARFR require the argument F (fraction) to be greater than or equal to 1.
"Frequency" must be 1, 2 or 4	The argument Frequency (number of coupon payment per year) in financial functions is limited to one of the following choices: 1, 2 or 4

function not installed	This error occurs when *SimplexNumerica* encounters an "" followed by a function name which it does not recognize as one of its built-in functions, or one that has been installed by a connection program.
function stack overflow	This error occurs when functions are nested too deeply. *SimplexNumerica* supports nesting of functions up to 50 levels deep.
hex number greater than	*SimplexNumerica* cannot convert a hex string to a number if the hex string is 32 bits
IEEE Floating Exception	
(Infinity or NaN)	This error means that the formula caused a computation to occur which could not be calculated properly by the computer's IEEE standard floating point hardware. Most likely, this means that the computation would produce an intermediate or final result outside the range +/-1.8e308.
illegal cell or range reference	It happens when a copy or move operation results in a cell or range reference that is outside the range A1..FAN32767.
illegal operand of	
"operator"	This error occurs when one or both of the operands of the specified "operator" are not valid. Most likely, a range name was used as an operand in an arithmetic expression.
improper argument type	One or more arguments to the function are incompatible with the type of arguments required by the functions.
improper coefficient type	In the polynomial evaluation function, (POLY), one or more of the polynomial coefficients are non-numeric.
improper dimensions	Several *SimplexNumerica* matrix functions and embedded tools have certain requirements on the dimensions of their matrix arguments. Check the reference manual, if you are uncertain about those requirements.
incompatible matrix dimensions	In matrix multiplication (MMUL), the number of columns in the first matrix must equal the number of rows in the second matrix.
incompatible range dimensions	The *SimplexNumerica* dot product functions (DOT) requires vectors of equal size. It will also compute the sum-of-products of any two ranges with equal dimensions.
index column contains empty cell	The first column in the lookup table referenced by VLOOKUP must not contain empty cells.
index out of range	In FIND, the third argument may not be larger than the length of the second argument. In MID, the second argument may not be larger than the length of the first argument.
index row contains empty cell	The first row in the lookup table referenced by HLOOKUP must not contain empty cells.
integer parameter out of range	An integer parameter greater than 4294967296 or less than -2147483649 has been entered.
interest rate should be 0	EFFECT and NOMINAL require that argument R (interest rate) to be greater than 0.
interest schedule must be a single column or row	The argument R (array of interest rates) in FVSCHEDULE must be a single column or row.
invalid cell reference	User has tried to access a cell with a row that is negative, zero, or greater than 32767, or with a column that is negative or greater than FAN, or 4095.

invalid date	*SimplexNumerica* could not understand the date format. Date values must be in the range 1-73,050, representing the dates January 1, 1900, to December 31, 2099, respectively. This error can also occur when the year, month, and day values passed to DATE do not represent an actual date within this range (February 31, 1950, or January 1, 2589, for example).
invalid day count basis	The day count basis in financial functions should be one of the following choices: 0 (30/360), 1(actual/actual), 2 (actual/360) or 3 (actual/365)
invalid range reference	User has tried to make a range reference that references cells beyond the range of the spreadsheet; that is, a row which is negative, zero, or greater than 32767, or a column which is negative or greater than FAN, or 4095.
invalid table	The table of reference points in INTERP2D or INTERP3D contains non-numeric values or blank cells.
invalid time	*SimplexNumerica* cannot parse a time that the user has provided. Time values are fractional values from 0 to 1, representing fractions of a 24-hour period. When interpreting a number as a date/time value, *SimplexNumerica*interprets the integer portion of the number as the date and the fractional portion as the time on that date. A negative value is invalid. Also, the TIME function must have arguments in the range of 0-23 hours, 0-59 minutes, and 0-59 seconds. Any other values are invalid.
iterative calculation not	
supported with "As Needed"	To avoid infinite looping, iterative (self-referential) calculations are not supported when the recalculation method is "As Needed". To use iterative calculations, the user must choose manual recalculation.
less than 2 arguments	POLY requires 2 or more arguments.
"Life" and "Period" must be integers	DDB requires that "Life" and "Period", arguments 3 and 4, respectively, be integers.
"Life" must be > 0	SLN and SYD require that "Life" is greater than 0.
lookup failed to produce a match	HLOOKUP or VLOOKUP failed to produce a match. This should only happen with an alphabetic lookup.
"Lower limit" must be	
>=0	The argument L (lower limit) should be greater than or equal to 0 in ERF and ERFC.
magnitude too large	NUMTOHEX requires an argument between 2147483646 and 2147483647, inclusive.
matrix is singular	It is mathematically impossible to invert a singular matrix.
matrix must be square	It is impossible to invert, take the eigenvalue of, or take the determinant of a non-square matrix.
"Match Type" must be 0 for string match	The argument T (type of match) must be 0 if argument V (value to be matched) is text in MATCH.
matrix must be symmetric	EIGEN requires a symmetric matrix.
modula divide by zero	Mod 0 is an undefined operation.
must be -15 to +15 places	ROUND cannot round to greater than 15 places on either side of the decimal point.
must have "Cost" = "Sal-	DDB, SLN, SYD, DB, and VDB require that the "Cost"' arguvage" >= 0

must have issue < first coupon < maturity	The values of argument I (issue date), FC (first coupon date) and M (maturity date) must satisfy the following condition: I < FC < M
must have issue < last coupon < maturity	The values of argument I (issue date), LC (last coupon date) and M (maturity date) must satisfy the following condition: I < LC < M
must have "Life"' = "Period"' >= 1	DDB, DB, and VDB all require that the "Life"' argument be greater than or equal to the "Period" argument, which must be greater than or equal to 1.
must have N 0, K 0 and N < K	The arguments N (number of objects to choose from) and K (Number of objects to be chosen) in PERMUT must follow the following condition: N0, K0 and N<K.
need at least 2 cash flow values	A single data point does not make a cash flow series; it takes two to trend. Computing the internal rate of return (IRR) is undefined for only one value.
	operand equal to 0
operand larger than 32 bits	Integers in *SimplexNumerica* cannot take more than 32 bits to express. This restricts integers to the range 2147483647 to -2147483648, or 4294967295 to zero, depending on whether the operand is only positive or can be negative.
no duplicate number	The MODE cannot find the most frequently occurring number because found
no match was found	MATCH is unsuccessful in finding a match.
non hex digits in string	HEXTONUM requires that its argument be a string containing only hex digits, 0-9 and A-F.
non-numeric operand	An expression of some sort has a non-numeric operand where a numeric operand is required, making the result of the expression undefined.
non-numeric value in ...	Doing arithmetic on alphabetic entities is undefined.
not enough arguments to function	User has entered too few arguments to the function.
"Number" is not in the reference list	The number to be ranked is not in the reference list in RANK.
number is too	
[large|small]	The number is at or beyond the limit of the ability of the computer to express, and is treated as if it were slightly within the limit.
number of compounding periods should be >=1	EFFECT and NOMINAL require that argument C (number of compounding periods) to be greater than or equal to 1.
one argument must be non-zero	ATAN2 requires that one of its arguments be non-zero.
operand contains error Some cell referenced by the operand is in an error condition, or contains a condition reference to a cell which is in an error condition, etc.	
operand out of range	CHAR only takes integers between 1 and 255
operands of "&" must be same type	The "&" operator serves a dual purpose: if its operands are numeric, it performs a bitwise AND operation; if its operands are text strings, it concatenates the two strings. If the operands are neither numeric nor both strings, this error occurs.
operand less than or equal to 0	GMEAN does not take arguments that are 0 or negative.

operands of ".." must be	The .. operator can only join two cell references to create a range. It cancell reference
"Payment" and "FV" must have the same sign	TERM requires that Payment and Future Value have the same sign.
"Payment" must be nonzero	TERM requires that Payment be non-zero.
"Period" must be >= 0	SYD requires that Period be greater than or equal to 0.
"Period" must be an integer 0	FV, PMT, PV, and RATE require that Period be an integer greater than 0.
polynomial degree must be between 1 and 10	PLS, POLYFIT, and POLYCOEF require that the polynomial degree by between 1 and 10.
pooled sample size less than 3	TTEST2EV requires a pooled sample size greater than 2 to be mathematically defined.
population less than 1	CVAR, CSTD, SSE, VAR, and STD require a population greater than or equal to 1.
"PV" and "FV" must be non-zero	CTERM and RATE require that Present and Future Values be nonzero by definition.
"PV" and "FV" must have the same sign	CTERM and RATE require that Present and Future Values have the same sign.
ranges must be same dimensions	PTTEST and CORR require that both their arguments be ranges of equal dimensions, since they work with pairs of values, one value from each range.
"Rate" must be greater than -1	CTERM, FV, PMT, PV, TERM, NPV, XNPV, and XIRR require that their Rate argument be greater than -1.
"Rate" must be non-zero	CTERM requires that its Rate argument be non-zero.
rate found is less than -1	IRR has found a rate less than -1 after iterating the maximum number of times.
recursion too deep	This error will occur if *SimplexNumerica* encounters "a condition string within a condition string". For example, it happens with a conditional statistical formula whose condition string calls another conditional statistical function which in turn contains its own condition string.
result of expression is a range	Some *SimplexNumerica* functions, such as CELLREF and RANGEREF, return cell references or range references as a result. Cell and range references cannot be the final result of a formula.
resultant string too long	A string generated by a formula is too long (greater than 512 characters).
row offset out of range	The third argument to the HLOOKUP function specifies an offset that is less than 0 or is greater than the depth of the range specified in the second argument.
sample missing from pair	The two input ranges to the paired t-test (PTTEST) and Pearson product-moment correlation (CORR) functions contain paired values. If a value appears at a given position in the first range, there must also be a value in the corresponding position of the second range.
sample size less than 2	CVARS, CSTDS, VARS, STDS, TTEST, PTTEST, TTEST2UV, and FTEST require a sample size greater than 1.
searching NULL list	searching list with a NULL function.
selector out of range	The first argument to CHOOSE must be 0 or more and be less than or equal to the number of the rest of the arguments - 1.

settlement date should be < maturity date	Settlement date should be earlier than maturity date in financial functions.
settlement date should be = issue date	Settlement date should not be earlier than the issue date.
showing NULL list	showing list with a NULL function
"Start Period" must be = 1	CUMIPMT and CUMPRINC require the argument S (start period) to be greater than or equal to 1.
starting date should be at beginning of "Dates"	The number in argument D (dates) should not precede the starting date in
XIRR and XNPV.	
substring longer than string	FIND cannot find an instance of the pattern string within a shorter target string, since it is impossible to embed a string in a string shorter than itself.
	substring not found
token buffer overflow	This error can only occur when a formula is entered that is more complex than *SimplexNumerica* can accept. *SimplexNumerica* can accept up to 200 operators, numbers, function calls, and text strings in a single formula, which is more than any human can reasonably deal with.
	too few arguments
too many arguments to function	User has provided too many arguments to the function. No function can take more than 100 arguments.
too many arguments	NOT only takes one argument, unlike the rest of the digital logic functions. ROW and COL take 1 argument, ANNOTATE takes 3-5 arguments.
Treasury Bill should not be outstanding more than 1 year	The period between the settlement date and maturity date of a Treasury bill should not exceed one year.
unable to parse extract	
filter	Happens when you are doing an Extract operation and you specify an invalid boolean expression; e.g., #==/5.
unable to parse search condition	Happens when you are doing a numeric search and you specify an invalid boolean expression; e.g., #==/5
undefined symbolic name	This error occurs when *SimplexNumerica* encounters a symbolic range or cell reference that has not been defined. To use a symbolic name to refer to a cell or range, you must first define it using the SetRangeName command.
unexpected question mark	*SimplexNumerica* supports C-language compatible condition expressions, which use the operator pair "?" and ":". If one of these operators appears without the other, an error occurs.
unresolved name in expression	A name that is not a valid function or named range has been used in the expression.
"Upper limit" must be >=0	The argument U (Upper limit) should be greater than or equal to 0 in ERF.
"values" and "dates" series must have the same dimension	XIRR and XNPV require the argument V (cash flow series) and the argument D (date series) to have the same dimension.
"Values" must have at least one inflow and one outflow	MIRR requires the value range contains at least one income (positive value) and one payment (negative value)
wrong number of arguments	The number of arguments passed to the function is incorrect. Check Section

7 End-user License Agreement

Developer:

- Dipl-Phys.-Ing. Ralf Wirtz
  Mürlenbach/Germany
  Email: support*SimplexNumerica*.com
  Web: www.*SimplexNumerica*.com

All programs developed by Ralf Wirtz, like SIMPLEXNUMERICA, *SIMPLEXIPC*, SIMPLEXEDITOR AND Simplexety ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL THE AUTHOR BE LIABLE TO YOU FOR ANY DAMAGES, INCLUDING INCIDENTAL OR CONSEQUENTIAL DAMAGES, ARISING OUT OF THE USE OF THE PROGRAM, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This *SimplexNumerica* software or any *SimplexNumerica* software that is made available to download from a server is the copyrighted work of Ralf Wirtz and/or its suppliers. Use of the Software is governed by the terms of the end user license agreement, if any, which accompanies or is included with the Software ("License Agreement") . An end user will be unable to install any Software that is accompanied by or includes a License Agreement, unless he or she first agrees to the License Agreement terms.

The Software is made available for downloading solely for use by end users according to the License Agreement. Any reproduction or redistribution of the Software not in accordance with the License Agreement is expressly prohibited by law, and may result in severe civil and criminal penalties. Violators will be prosecuted to the maximum extent possible.

WITHOUT LIMITING THE FOREGOING, COPYING OR REPRODUCTION OF THE SOFTWARE TO ANY OTHER SERVER OR LOCATION FOR FURTHER REPRODUCTION OR REDISTRIBUTION IS EXPRESSLY PROHIBITED.

THE SOFTWARE IS WARRANTED, IF AT ALL, ONLY ACCORDING TO THE TERMS OF THE LICENSE AGREEMENT. EXCEPT AS WARRANTED IN THE LICENSE AGREEMENT, RALF WIRTZ HEREBY DISCLAIMS ALL WARRANTIES AND CONDITIONS WITH REGARD TO THE SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES AND CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT.

RESTRICTED RIGHTS LEGEND. Any Software which is downloaded from this Server for or on behalf of the United States of America, its agencies and/or instrumentalities ("U.S. Government"), is provided with Restricted Rights. Use , duplication, or disclosure by the U.S. Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 or subparagraphs (c)(1) and (2) of the Commercial Computer Software - Restricted Rights at 48 CFR 52.227-19, as applicable. Manufacturer is Ralf Wirtz, Kasterstr. 30, 52428 Jülich.

NOTICE SPECIFIC TO DOCUMENTS AVAILABLE ON THIS WEBSITE

Permission to use Documents (such as white papers, press releases, data sheets and FAQs) from this server ("Server") is granted, provided that (1) the below copyright notice appears in all copies and that both the copyright notice and this permission notice appear, (2) use of such Documents from this Server is for informational and non-commercial or personal use only and will not be copied or posted on any network computer or broadcast in any media, and (3) no modifications of any Documents are made. Use for any other purpose is expressly prohibited by law, and may result in severe civil and criminal penalties. Violators will be prosecuted to the maximum extent possible.

Documents specified above do not include the design or layout of the Ralf Wirtz website or any other Ralf Wirtz owned, operated, licensed or controlled site. Elements of Ralf Wirtz Web sites are protected by trade dress and other laws and may not be copied or imitated in whole or in part. No logo, graphic , sound or image from any Ralf Wirtz website may be copied or retransmitted unless expressly permitted by Ralf Wirtz.

RALF WIRTZ AND/OR ITS RESPECTIVE SUPPLIERS MAKE NO REPRESENTATIONS ABOUT THE SUITABILITY OF THE INFORMATION CONTAINED IN THE DOCUMENTS AND RELATED GRAPHICS PUBLISHED ON THIS SERVER FOR ANY PURPOSE. ALL SUCH DOCUMENTS AND RELATED GRAPHICS ARE PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND. RALF WIRTZ AND/OR ITS RESPECTIVE SUPPLIERS HEREBY DISCLAIM ALL WARRANTIES AND CONDITIONS WITH REGARD TO THIS INFORMATION, INCLUDING ALL IMPLIED WARRANTIES AND CONDITIONS OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL RALF WIRTZ AND/OR ITS RESPECTIVE SUPPLIERS BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF INFORMATION AVAILABLE FROM THIS SERVER.

THE DOCUMENTS AND RELATED GRAPHICS PUBLISHED ON THIS SERVER COULD INCLUDE TECHNICAL INACCURACIES OR TYPOGRAPHICAL ERRORS. CHANGES ARE PERIODICALLY ADDED TO THE INFORMATION HEREIN. RALF WIRTZ AND/OR ITS RESPECTIVE SUPPLIERS MAY MAKE IMPROVEMENTS AND/OR CHANGES IN THE PRODUCT(S) AND/OR THE PROGRAM(S) DESCRIBED HEREIN AT ANY TIME.

NOTICES REGARDING SOFTWARE, DOCUMENTS AND SERVICES AVAILABLE ON THIS WEBSITE.

IN NO EVENT SHALL RALF WIRTZ AND/OR ITS RESPECTIVE SUPPLIERS BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTUOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF SOFTWARE, DOCUMENTS, PROVISION OF OR FAILURE TO PROVIDE SERVICES, OR INFORMATION AVAILABLE FROM THIS SERVER.

LINKS TO THIRD PARTY SITES

THE LINKS IN THIS AREA WILL LET YOU LEAVE RALF WIRTZ'S SITE. THE LINKED SITES ARE NOT UNDER THE CONTROL OF RALF WIRTZ AND RALF WIRTZ IS NOT RESPONSIBLE FOR THE CONTENTS OF ANY LINKED SITE OR ANY LINK CONTAINED IN A LINKED SITE. RALF WIRTZ IS PROVIDING THESE LINKS TO YOU ONLY AS A CONVENIENCE, AND THE INCLUSION OF ANY LINK DOES NOT IMPLY ENDORSEMENT BY RALF WIRTZ OF THE SITE.

COPYRIGHT NOTICE.
Copyright © 1988-2017 Dipl.-Phys.-Ing. Ralf Wirtz, Hinter Herschenhaus, Mürlenbach/Eifel.
All rights reserved.
TRADEMARKS. Microsoft, Windows, Windows NT, MSN, The Microsoft Network and/or other Microsoft

products referenced herein are either trademarks or registered trademarks of Microsoft. Other product and company names mentioned herein may be the trademarks
of their respective owners.

The names of companies, products, people, characters and/or data mentioned herein are fictitious and are in no way intended to represent any real individual, company, product or event, unless otherwise noted.

Any rights not expressly granted herein are reserved.