

LaTeXcel Package

Peter Brookes Chambers

Version 1.0, 2023/12/03

Contents

1	Introduction	2
2	Environments, Macros and Colours	3
2.1	The <code>\formula</code> Macro	3
3	Options	4
4	Examples	7
4.1	Default	7
4.2	Background Colours	8
4.3	Text Formatting	9
4.4	Padding and Cell Addresses	10
4.5	Cell Sizing	12
4.6	Borders	13
4.7	Highlighting a Cell	16
4.8	Formulae and the Formula Bar	17

Introduction

This package provides an environment which mimics the appearance of a spreadsheet. It is modeled after the Google Sheets interface, but is also very similar to that of Excel. It is best seen with an example (alignment in the source code is purely for readability):

```
1 \begin{latexcel}[
2   selected cell      = {1,4},
3   header cell style  = \color{white}\textbf,
4   header row color   = Sheets_Blue,
5   odd row color      = Sheets_PaleBlue,
6 ]
7   Name              & Age   & City              \\
8 % |-----|-----|-----|
9   John Smith        & 28    & New York          \\
10  Alice Johnson      & 35    & Los Angeles       \\
11  Bob Anderson       & 22    & Chicago           \\
12  Eva Williams       & 30    & San Francisco     \\
13  David Brown        & 25    & Miami
14 \end{latexcel}
```

B5		fx Bob Anderson			
	A	B	C	D	E
1					
2		Name	Age	City	
3		John Smith	28	New York	
4		Alice Johnson	35	Los Angeles	
5		Bob Anderson	22	Chicago	
6		Eva Williams	30	San Francisco	
7		David Brown	25	Miami	
8					

The package is loaded as normal with `\usepackage{latexcel}`. It relies on the following packages:

- `tikz` for all graphical elements
- `xcolor` for colours
- `environ` for the environment

- \LaTeX syntax, for all options and content processing

Note that, since the graphics are rendered by \LaTeX , the font used will significantly impact the appearance of the tables. Of particular note, the “ \LaTeX ” in the formula bar may appear strange in certain fonts. In this document, the **Fira Sans** and **Fira Mono** fonts are used, which are freely available on CTAN. The Helvetica font is also a very close match to the Arial font used by default in Google Sheets.

Environments, Macros and Colours

The package provides a single environment, `\latexcel`, which takes one optional argument; a comma-separated list of key-value pairs. These are detailed in the next section. The contents of the environment should follow the same syntax as a `\tabular` environment: lines delimited by a double backslash (`\`), and columns delimited by an ampersand (`&`). The final line should **not** be terminated with a double backslash.

The package also provides a single macro, `\formula`, which takes one argument; the formula to be parsed and coloured. This is detailed in the section below.

In addition to the `\latexcel` environment, the package provides a number of colours which mimic those used in Google Sheets. These are shown in Table 1.












Name	Example	Alias
SheetsOrange	 <code>A1:B3</code>	Sheets1
SheetsPurple	 <code>A1:B3</code>	Sheets2
SheetsCyan	 <code>A1:B3</code>	Sheets3
SheetsMaroon	 <code>A1:B3</code>	Sheets4
SheetsBlue	 <code>A1:B3</code>	Sheets5
SheetsYellow	 <code>A1:B3</code>	Sheets6
SheetsGreen	 <code>A1:B3</code>	Sheets7
SheetsBrown	 <code>A1:B3</code>	Sheets8
SheetsNumber	 <code>12, TRUE</code>	–
SheetsString	 <code>“str”</code>	–
SheetsText	 <code>SUM</code>	–

Table 1: The colours provided by the `\latexcel` package. These are in the same order in which Google sheets will apply them to cell ranges in formulae.

The `\formula` Macro

The `\formula` macro is provided to format formulae in the same way as Google Sheets. It takes one argument, which is the formula to be formatted. It applies colours to cell addresses, strings, and booleans in the same way as Google Sheets. The example below shows the output of the `\formula` macro for a number of different formulae. Unfortunately, due to the lack of lookahead and lookbehind capabilities of the regex parsing in \LaTeX , it is not possible to automatically colour numbers, so this will be visibly different to Google Sheets.

```

1 \notomono % Enable the noto mono font, a very close match to the font used in Google Sheets formulae
2 \formula{= LINEST($I$2:$I$12, J2:J12, TRUE, FALSE)} \\
3 \formula{= "$R^2 = " & L2 & "$"} \\
4 \formula{= IF(COUNTBLANK(A2) = 0, A2, B2)} \\
5 \formula{= ('Main Sheet'!A1 + Sheet2!A1) / (Sheet2!A1 - 'Main Sheet'!A1)}

```

```

= LINEST($I$2:$I$12, J2:J12, TRUE, FALSE)
= "$R^2 = " & L2 & "$"
= IF(COUNTBLANK(A2) = 0, A2, B2)
= ('Main Sheet'!A1 + Sheet2!A1) / (Sheet2!A1 - 'Main Sheet'!A1)

```

Cell addresses are coloured in order, repeating colours for repeated cell addresses (as in the third example for `A2`, and fourth for both cell addresses). Booleans are coloured with the colour `SheetsNumber`, which is the same colour as numbers in Google Sheets. Strings are coloured with the colour `SheetsString`, which is the same colour as strings in Google Sheets. Note that the formula passed to the `\formula` macro is not expanded in the same way as normal \LaTeX , and so dollar signs, ampersands, and most other special characters do not need to be escaped. However, including braces `{, }` may cause unexpected behaviour, and will very likely throw an (potentially cryptic) error. This also means that \LaTeX macros cannot be used in formulae.

As with everything else, whilst this is modelled on Google Sheets, it is also extremely similar to the appearance of Excel formulae, with the only major difference being the colours and their order. If it is important to emulate Excel specifically, simply redefine the colours in table 1 to match those used in Excel.

Options

The `\latexcel` environment takes a number of options, which are described below. These options are passed as a set of comma-separated key-value pairs to the `\latexcel` environment. The option is given on the left, below which is the default (the value used if the option is not specified), and the initial value (if any) is given in brackets; this is the value used if the option is specified but no value is given.

Note: all options which use colours use the US spelling “color” rather than the UK spelling “colour” which is used in the *text* of this documentation.

<code>row height</code> 1	Accepts any float, which is interpreted as a height in centimetres. This sets the height of all cells in the table, including headers. It is not possible to set the height of individual rows.
<code>padding rows</code> 1	Accepts any non-negative integer. This sets the number of additional rows rendered before and after the table. Be sure that this and the <code>starting row</code> option are set appropriately to avoid unexpected row numbering – they are not checked for consistency.
<code>padding row height</code> 0.65	Accepts any float, which is interpreted as a height in centimetres. This sets the height of the padding rows.

<code>starting row</code> 2	Accepts any positive integer. This sets the number of the first row in the table (the header row). This is absolute, so padding row numbering is calculated relative to this value. If the number of padding rows is greater than or equal to this value, some padding rows will be rendered with row numbers less than 1.
<code>column width</code> 4	Accepts any float, which is interpreted as a width in centimetres. This sets the width of all columns in the table. It is not possible to set the width of individual columns.
<code>padding columns</code> 1	Accepts any non-negative integer. This sets the number of additional columns rendered before and after the table. Be sure that this and the <code>starting column</code> option are set appropriately to avoid unexpected column numbering – they are not checked for consistency.
<code>padding column width</code> 0.65	Accepts any float, which is interpreted as a width in centimetres. This sets the width of the padding columns.
<code>starting column</code> 2	Accepts any positive integer. This sets the index of the first column in the table (the leftmost column). Column indices are rendered as letters (in the style of Excel or Google Sheets), with 1 corresponding to A. This is absolute, so padding column numbering is calculated relative to this value. If the number of padding columns is greater than or equal to this value, some padding columns will be rendered with column indices less than 1. This will result in unexpected characters being rendered as part of the table. It is vital to avoid this.
<code>cell style</code> {}	Accepts a comma-separated list of commands to apply to the non-header cell content, cell by cell. This can be empty, a single value, or multiple values. If the number of styles is less than the number of columns, the styles are repeated, allowing for (for example) alternating styles. The final token in each style may take the content of the cell as an argument, for example <code>\textbf</code> is acceptable.
<code>every cell style</code> {}	Accepts any commands (not a list of commands), to be applied to every non-header cell, after any cell-specific styles specified in <code>cell style</code> . The final token in <code>every cell style</code> may take the cell contents as an argument, after any cell-specific styles have been applied. In effect, the order is <code>\everycellstyle{\cellstyle{cell contents}}</code> . This allows for styling specified in <code>cell style</code> to overwrite the styling specified in <code>every cell style</code> .
<code>header cell style</code> {}	Accepts a comma-separated list of commands to apply to the header cell content, cell by cell. This can be empty, a single value, or multiple values. If the number of styles is less than the number of columns, the styles are repeated, allowing for (for example) alternating styles. The final token in each style may take the content of the header as an argument, for example <code>\textbf</code> is acceptable.
<code>every header cell style</code> {}	Accepts any commands (not a list of commands), to be applied to every header cell, after any cell-specific styles specified in <code>header cell style</code> . The final token in <code>every header cell style</code> may take the header cell contents as an argument, after any cell-specific styles have been applied. In effect, the order is <code>\everyheadercellstyle{\headercellstyle{header cell contents}}</code> . This allows for styling specified in <code>header cell style</code> to overwrite the styling specified in <code>every header cell style</code> .
<code>header row color</code> white	Accepts any colour, including colour combinations such as <code>blue!50!white</code> . This sets the background colour of the header cells.
<code>even row color</code> white	Accepts any colour, including colour combinations such as <code>blue!50!white</code> . This sets the background colour of the even-numbered rows.

<code>odd row color</code> <code>white</code>	Accepts any colour, including colour combinations such as <code>blue!50!white</code> . This sets the background colour of the odd-numbered rows.
<code>borders</code> <code>true (true)</code>	Boolean. If <code>true</code> , borders are rendered around the table and between columns and rows according to <code>row borders</code> and <code>column borders</code> . If <code>false</code> , no borders are rendered, overwriting the values of <code>row borders</code> and <code>column borders</code> .
<code>row borders</code> <code>false (true)</code>	Boolean. If <code>true</code> , borders are rendered between rows. If <code>false</code> , no borders are rendered between rows, except for the header row. Borders are always rendered on the outside of the table, unless <code>borders = false</code>
<code>column borders</code> <code>true (true)</code>	Boolean. If <code>true</code> , borders are rendered between columns. If <code>false</code> , no borders are rendered between columns. Borders are always rendered on the outside of the table, unless <code>borders = false</code>
<code>border style</code> <code>{}</code>	Accepts any valid tikz style, such as “ <code>dashed, draw = red</code> ”. This sets the style of the borders, appended to a default style of “ <code>line width = <width>, black</code> ”, where <code><width></code> is specified by the <code>border width</code> option.
<code>border width</code> <code>0.75pt</code>	Accepts any \LaTeX dimension, such as <code>1mm</code> or <code>0.5pt</code> . This sets the line width of the borders.
<code>selected cell</code> <code>{}</code>	Accepts a comma-separated pair of integers, which specify the row and column of the selected cell within the table; i.e. such that the first header is <code>{1,1}</code> . Padding cells can be selected, but setting this value to a cell which is not rendered (i.e. not within the table and not within the padding rows or columns) will throw an error. This is not checked internally. The selected cell will be highlighted, and its contents displayed in the formula bar if <code>formula</code> is not set.
<code>formula</code> <code>{}</code>	Accepts a token list (i.e. standard \LaTeX syntax) to be displayed in the formula bar. This is prepended by an equals sign, and rendered in the <code>\ttfamily</code> font via the <code>\formula</code> macro. If this is not set, the contents of the selected cell are displayed instead, without the use of the <code>\formula</code> macro.
<code>auto formula</code> <code>true (true)</code>	Boolean. If <code>true</code> , the formula will be passed to the <code>\formula</code> macro, which will format it in the same way as Google Sheets. If <code>false</code> , the formula will simply be rendered in the <code>\ttfamily</code> font, allowing for manual formatting.
<code>show formula bar</code> <code>true (true)</code>	Boolean. If <code>true</code> , the formula bar is rendered, including the formula. If <code>false</code> , the entire formula bar is not rendered, and the picture stops at the top of the column labels.
<code>scale</code> <code>1</code>	Accepts any float. This scales the entire picture by the given factor using a <code>scalebox</code> .

Examples

Default

With no options specified, the table is rendered as follows:

```
1 \begin{latexcel}
2   Name      & Age  & City      \\
3   John Smith & 28   & New York  \\
4   Alice Johnson & 35   & Los Angeles \\
5   Bob Anderson & 22   & Chicago   \\
6   Eva Williams & 30   & San Francisco \\
7   David Brown  & 25   & Miami     \\
8 \end{latexcel}
```

B2 | fx

	A	B	C	D	E
1					
2		Name	Age	City	
3		John Smith	28	New York	
4		Alice Johnson	35	Los Angeles	
5		Bob Anderson	22	Chicago	
6		Eva Williams	30	San Francisco	
7		David Brown	25	Miami	
8					

Background Colours

Background colours can be specified for the header row, and for odd and even rows separately. The example below uses the colours from the Rosé Pine Dawn palette, which can be found here: <https://rosepinetheme.com>. The option `header cell style` is also used to set the header text to an appropriate colour.

```
1 \begin{latexcel}[
2   header cell style = \color{BackgroundColour}\textbf,
3   cell style       = \color{ForegroundColour},
4   header row color = Blue,
5   even row color   = BackgroundColour,
6   odd row color    = Surfaces5
7 ]
8   Name           & Age   & City           \\
9 % |-----|-----|-----|
10  John Smith     & 28   & New York       \\
11  Alice Johnson  & 35   & Los Angeles    \\
12  Bob Anderson  & 22   & Chicago        \\
13  Eva Williams  & 30   & San Francisco  \\
14  David Brown   & 25   & Miami          \\
15 \end{latexcel}
```

B2

fx

	A	B	C	D	E
1					
2		Name	Age	City	
3		John Smith	28	New York	
4		Alice Johnson	35	Los Angeles	
5		Bob Anderson	22	Chicago	
6		Eva Williams	30	San Francisco	
7		David Brown	25	Miami	
8					

Text Formatting

The formatting for the text in cells can be specified using `cell style` and `every cell style` for non-header cells, and `header cell style` and `every header cell style` for header cells. The “every” variant is applied to each cell, while the non-“every” variant can be used to specify formatting for each column individually, and overrides the “every” variant. The example below makes use of the `siunitx` package to format the numbers in the table with units.

```
1 % Declare a new SI unit for year
2 \DeclareSIUnit{\year}{yr}
3
4 \newcommand{\formatAge}[1]{
5   \SI{#1}{\year}
6 }
7
8 \begin{latexcel}[
9   header cell style = \textbf,
10  every cell style  = \color{blue},
11  cell style        = {}, {\formatAge}, \ttfamily,
12 ]
13   Name           & Age   & City           \\
14 % |-----|-----|-----|
15   John Smith    & 28   & New York       \\
16   Alice Johnson & 35   & Los Angeles    \\
17   Bob Anderson  & 22   & Chicago        \\
18   Eva Williams  & 30   & San Francisco  \\
19   David Brown   & 25   & Miami          \\
20 \end{latexcel}
```

B2

fx

	A	B	C	D	E
1					
2		Name	Age	City	
3		John Smith	28yr	New York	
4		Alice Johnson	35yr	Los Angeles	
5		Bob Anderson	22yr	Chicago	
6		Eva Williams	30yr	San Francisco	
7		David Brown	25yr	Miami	
8					

Padding and Cell Addresses

The `padding rows` and `padding columns` options can be used to specify the number of additional rows and columns to render before and after the table. Setting this to at least 1 helps to sell the rendered picture as a genuine extract from Google Sheets, but it is not necessary. The example below includes no padding rows or columns, with the table starting at the cell `A1`.

```
1 \begin{latexcel}[
2   padding rows = 0,
3   padding columns = 0,
4   starting row = 1,
5   starting column = 1,
6 ]
7   Name           & Age   & City           \\
8 % |-----|-----|-----|
9   John Smith     & 28   & New York       \\
10  Alice Johnson  & 35   & Los Angeles    \\
11  Bob Anderson   & 22   & Chicago        \\
12  Eva Williams   & 30   & San Francisco  \\
13  David Brown    & 25   & Miami          \\
14 \end{latexcel}
```

A1	A	B	C
1	Name	Age	City
2	John Smith	28	New York
3	Alice Johnson	35	Los Angeles
4	Bob Anderson	22	Chicago
5	Eva Williams	30	San Francisco
6	David Brown	25	Miami

In the example below, the table starts at `F3`, with 2 padding columns and 1 padding row. Note that the row and column numbering is absolute, so the padding rows and columns are numbered relative to the starting row and column. This means that the labelled columns start at `D` and the labelled rows start at `2`.

```

1 \begin{latexcel}[
2   padding columns = 2,
3   starting row = 3,
4   starting column = 6
5 ]
6   Name          & Age   & City          \\
7 % |-----|-----|-----|
8   John Smith    & 28   & New York      \\
9   Alice Johnson & 35   & Los Angeles   \\
10  Bob Anderson  & 22   & Chicago       \\
11  Eva Williams  & 30   & San Francisco \\
12  David Brown   & 25   & Miami
13 \end{latexcel}

```

F3								
	D	E	F	G	H	I	J	
2								
3			Name	Age	City			
4			John Smith	28	New York			
5			Alice Johnson	35	Los Angeles			
6			Bob Anderson	22	Chicago			
7			Eva Williams	30	San Francisco			
8			David Brown	25	Miami			
9								

Cell Sizing

Cell sizes can be specified for the table cells and the padding cells separately, always in centimetres. The example below uses a smaller row height than the default for the table cells, and a wider column width for the padding cells. The `scale` option can also be used to scale the entire picture; here, it is used to shrink the picture to 80% of its original size.

```
1 \begin{latexcel}[
2   row height = 0.65,
3   padding column width = 1,
4   scale = 0.8,
5 ]
6   Name          & Age   & City          \\
7 % |-----|-----|-----|
8   John Smith    & 28   & New York      \\
9   Alice Johnson & 35   & Los Angeles   \\
10  Bob Anderson  & 22   & Chicago       \\
11  Eva Williams  & 30   & San Francisco \\
12  David Brown   & 25   & Miami
13 \end{latexcel}
```

B2					
fx					
	A	B	C	D	E
1					
2		Name	Age	City	
3		John Smith	28	New York	
4		Alice Johnson	35	Los Angeles	
5		Bob Anderson	22	Chicago	
6		Eva Williams	30	San Francisco	
7		David Brown	25	Miami	
8					

Borders

By default, the table will include a border around the entire table, as well as between the columns and to separate the header row from the table body. Passing `borders = false` will remove all borders, overriding all other border options. This is shown below, with some background colours.

```
1 \begin{latexcel}[
2   borders          = false,
3   header row color  = Blue,
4   header cell style = \color{BackgroundColour}\textbf,
5   cell style        = \color{ForegroundColour},
6   even row color    = BackgroundColour,
7   odd row color     = Surface5
8 ]
9   Name           & Age   & City           \\
10  % |-----|-----|-----|
11   John Smith    & 28   & New York       \\
12   Alice Johnson & 35   & Los Angeles    \\
13   Bob Anderson  & 22   & Chicago        \\
14   Eva Williams  & 30   & San Francisco  \\
15   David Brown   & 25   & Miami          \\
16 \end{latexcel}
```

B2					
	A	B	C	D	E
1					
2		Name	Age	City	
3		John Smith	28	New York	
4		Alice Johnson	35	Los Angeles	
5		Bob Anderson	22	Chicago	
6		Eva Williams	30	San Francisco	
7		David Brown	25	Miami	
8					

Row borders and column borders can be enabled or disabled separately. The example below shows both in their non-default state.

```
1 \begin{latexcel}[
2   column borders      = false,
3   row borders        = true,
4   header row color    = Blue,
5   header cell style   = \color{BackgroundColour}\textbf,
6   cell style          = \color{ForegroundColour},
7   even row color      = BackgroundColour,
8   odd row color       = Surface5
9 ]
10 Name           & Age   & City           \\
11 % |-----|-----|-----|
12 John Smith     & 28   & New York       \\
13 Alice Johnson  & 35   & Los Angeles    \\
14 Bob Anderson   & 22   & Chicago        \\
15 Eva Williams   & 30   & San Francisco  \\
16 David Brown    & 25   & Miami          \\
17 \end{latexcel}
```

B2					
	A	B	C	D	E
1					
2		Name	Age	City	
3		John Smith	28	New York	
4		Alice Johnson	35	Los Angeles	
5		Bob Anderson	22	Chicago	
6		Eva Williams	30	San Francisco	
7		David Brown	25	Miami	
8					

The border styling can be specified with the `border` style option. This option is passed to the `tikz draw` command, so any valid `tikz` styling can be used. It is appended to “`line width = <width>, black`” (where `<width>` is specified with the `border width` option), so it can override these options if desired. The example below uses a dashed line of a different colour. Additionally, the option `border width` is used to increase the width of the borders.

```

1 \begin{latexcel}[
2   header row color    = Blue,
3   header cell style   = \color{BackgroundColour}\textbf,
4   cell style          = \color{ForegroundColour},
5   even row color      = BackgroundColour,
6   odd row color       = Surface5,
7   border style        = {densely dashed, Red},
8   border width        = 1mm,
9 ]
10  Name                & Age  & City                \\
11  % |-----|-----|-----|
12  John Smith          & 28  & New York            \\
13  Alice Johnson        & 35  & Los Angeles         \\
14  Bob Anderson         & 22  & Chicago              \\
15  Eva Williams         & 30  & San Francisco       \\
16  David Brown          & 25  & Miami
17 \end{latexcel}

```

B2					
	A	B	C	D	E
1					
2		Name	Age	City	
3		John Smith	28	New York	
4		Alice Johnson	35	Los Angeles	
5		Bob Anderson	22	Chicago	
6		Eva Williams	30	San Francisco	
7		David Brown	25	Miami	
8					

Highlighting a Cell

The `selected cell` option can be used to highlight a cell, and display its contents in the formula bar. The example below highlights the cell **D3**, which is in the 3rd column and 2nd row of the table. The coordinates are 1-indexed, starting from the top-left cell of the table (i.e, the first header cell). Note that the cell contents are displayed in the formula bar, as well as the selected cell address in the top left.

```
1 \begin{latexcel}[
2   selected cell = {3, 2}
3 ]
4   Name           & Age   & City           \\
5 % |-----|-----|-----|
6   John Smith     & 28   & New York       \\
7   Alice Johnson  & 35   & Los Angeles    \\
8   Bob Anderson   & 22   & Chicago        \\
9   Eva Williams   & 30   & San Francisco  \\
10  David Brown    & 25   & Miami          \\
11 \end{latexcel}
```

D3		fx New York			
	A	B	C	D	E
1					
2		Name	Age	City	
3		John Smith	28	New York	
4		Alice Johnson	35	Los Angeles	
5		Bob Anderson	22	Chicago	
6		Eva Williams	30	San Francisco	
7		David Brown	25	Miami	
8					

Formulae and the Formula Bar

The entire formula bar can be hidden with the `show formula bar` option. This is shown below; a cell is still selected, but its contents and address are not displayed.

```
1 \begin{latexcel}[
2   selected cell      = {3, 4},
3   show formula bar   = false
4 ]
5   Name              & Age   & City          \\
6   John Smith        & 28   & New York      \\
7   Alice Johnson     & 35   & Los Angeles   \\
8   Bob Anderson      & 22   & Chicago       \\
9   Eva Williams      & 30   & San Francisco \\
10  David Brown       & 25   & Miami
11 \end{latexcel}
```

	A	B	C	D	E
1					
2		Name	Age	City	
3		John Smith	28	New York	
4		Alice Johnson	35	Los Angeles	
5		Bob Anderson	22	Chicago	
6		Eva Williams	30	San Francisco	
7		David Brown	25	Miami	
8					

By default, if a cell is selected then its contents will be shown in the formula bar. However, this can be overridden by providing the `formula` option with a value. This value will be displayed in the formula bar, preceded by an equals sign. The example below shows the same table as above, but with the formula `=AVERAGE(C3:C7)` displayed in the formula bar. The colouring is applied automatically via the `\formula` macro. To disable this, set `auto formula = false`.

```

1 \begin{latexcel}[
2   selected cell   = {2, 7},
3   formula         = {AVERAGE($C$3:$C$7)}
4 ]
5   Name           & Age   & City           \\
6   John Smith     & 28   & New York       \\
7   Alice Johnson  & 35   & Los Angeles    \\
8   Bob Anderson   & 22   & Chicago        \\
9   Eva Williams   & 30   & San Francisco  \\
10  David Brown     & 25   & Miami          \\
11  \bfseries Average Age & 26   & 
12 \end{latexcel}

```

C8

fx

= AVERAGE(\$C\$3:\$C\$7)

	A	B	C	D	E
1					
2		Name	Age	City	
3		John Smith	28	New York	
4		Alice Johnson	35	Los Angeles	
5		Bob Anderson	22	Chicago	
6		Eva Williams	30	San Francisco	
7		David Brown	25	Miami	
8		Average Age	26		
9					

With `auto formula = true`, the value passed to the `formula` option is accepted without expansion in the usual manner, and so most special characters such as dollar signs and ampersands do not need to be escaped. With `auto formula = false`, the value is treated as a normal token list (i.e., normal \LaTeX), and so special characters must be escaped. This does allow for the use of \LaTeX macros. The example below shows the same table as above, but with the formula formatted manually.

```

1 \begin{latexcel}[
2   selected cell   = {2, 7},
3   formula        = {AVERAGE(\textcolor{Sheets1}{\C\$3:\C\$7})},
4   auto formula    = false
5 ]
6   Name           & Age   & City           \\
7   John Smith     & 28   & New York       \\
8   Alice Johnson  & 35   & Los Angeles    \\
9   Bob Anderson   & 22   & Chicago        \\
10  Eva Williams   & 30   & San Francisco  \\
11  David Brown    & 25   & Miami          \\
12  \bfseries Average Age & 26   &
13 \end{latexcel}

```

C8

▼

fx

= AVERAGE(C\$3:C\$7)

	A	B	C	D	E
1					
2		Name	Age	City	
3		John Smith	28	New York	
4		Alice Johnson	35	Los Angeles	
5		Bob Anderson	22	Chicago	
6		Eva Williams	30	San Francisco	
7		David Brown	25	Miami	
8		Average Age	26		
9					