The statistics package
Compute and typeset statistics table and graphics*

Julien "_FrnchFrgg_" Rivaud†

Released 2019/09/29

Contents

1 statistics documentation

The statistics package can compute and typeset statistics like frequency tables, cumulative distribution functions (increasing or decreasing, in frequency or absolute count domain), from the counts of individual values, or ranges, or even the raw value list with repetitions.

It can also compute and draw a bar diagram in case of individual values, or, when the data repartition is known from ranges, an histogram or the continuous cumulative distribution function.

You can ask statistics to display no result, selective results or all of them. Similarly statistics can draw only some parts of the graphs. Every part of the generated tables or graphics is customizable.

*This file describes v2.2, last revised 2019/09/29.

†E-mail: frnchfrgg@free.fr

1.1 Specifying and converting data

To compute and typeset things, statistics starts from what this documentation calls a ⟨*data source*⟩. Such a source can take two forms:

- A comma-separated list of ⟨*value*⟩ `[=` ⟨*count*⟩ `]`;
- A `\`⟨*macro*⟩ containing such a list.

If ⟨*count*⟩ is missing, it defaults to 1. *A priori* the ⟨*value*⟩s need not be unique nor sorted, but `\StatsTable` and `\StatsGraph` expect them to be. If you want your data to be in the form of a raw list of unsorted and repeated values, you can thus use the following command to convert the data to a form suitable for `\StatsTable` and `\StatsGraph`:

`\StatsSortData` `\StatsSortData \`⟨*destination*⟩ `= {`⟨*data source*⟩`}`

This command expect each ⟨*value*⟩ in the ⟨*data source*⟩ to be convertible to a floating point number (as understood by l3fp from the LaTeX3 kernel). It defines `\`⟨*destination*⟩ to hold an equivalent data source, where ⟨*value*⟩s are sorted in increasing order, and ⟨*count*⟩s are consolidated. As for all other statistics commands, ⟨*data source*⟩ can be either given directly between braces, or as a `\`⟨*macro*⟩ which contains the list.

```
\StatsSortData \mydata = { 2, 11=8, 6=3, 2=2, 11=1 }
\def \rawdata { 2=2, 11=9, 6, 2, 6, 6 }
\StatsSortData \yourdata = \rawdata
mydata contains [\mydata]\\
yourdata contains [\yourdata]
```

mydata contains [2=3,6=3,11=9]
yourdata contains [2=3,6=3,11=9]

The `\StatsTable` command will always assume that the ⟨*data source*⟩ is sorted and will not try to parse the ⟨*value*⟩s. On the contrary, `\StatsGraph` *will* parse each ⟨*value*⟩, and will act differently depending on whether every ⟨*value*⟩ is a ⟨*range*⟩ or the form `\IN` ⟨*[or]*⟩ ⟨*min*⟩ `;` ⟨*max*⟩ ⟨*[or]*⟩, or not.

If your ⟨*data source*⟩ is not given in ranges, but you want to count the values falling in each ⟨*range*⟩ of a list you can use:

`\StatsRangeData` `\StatsSortData \`⟨*destination*⟩ `= {`⟨*data source*⟩`} (`⟨*range list*⟩`)`

This command expect each ⟨*value*⟩ in the ⟨*data source*⟩ to be convertible to a floating point number (as understood by l3fp from the LaTeX3 kernel). It also expects ⟨*range list*⟩ to be a comma-separated list of ⟨*range*⟩s, and will define `\`⟨*destination*⟩ to a ⟨*data source*⟩ whose ⟨*value*⟩s are the said ⟨*range*⟩s and whose counts are, well… the number of floating point values that lie in those ⟨*range*⟩s.

`\StatsRangeData` does not need the ⟨*range*⟩s to be sorted, nor even disjoint, but in that case the behavior of `\StatsGraph` is unspecified.

Here is an example[1]:

```
\StatsRangeData \facebook = { 0, 1, 1.5, 1.5, 2, 3, 2.4, 2, 2.4=5,
                              3, 4=10, 5=6, 6=9, 6.5=5, 7, 7.1, 7.2,
                              7.3, 7.4, 7.5, 7.6, 7.7, 7, 7, 8, 8, 8,
                              9=5, 12=12}
                             (\IN[0;1;[, \IN[1;2;[, \IN[2;4;[,
                             \IN[4;7;[, \IN[7;10;[, \IN[10;14;[)
\tltostr \facebook
```

\IN [0;1;[=1,\IN [1;2;[=3,\IN [2;4;[=10,\IN [4;7;[=30,\IN [7;10;[=18,\IN [10;14;[=12

This data source will be used throughout the documentation.

[1] The `\tltostr` command is defined in this documentation to be an alias for the LaTeX3 command `\tl_to_str:N` which is equivalent to `\detokenize\expandafter{\`⟨*macro*⟩`}`.

1.2 Setting options

`\statisticssetup` `\statisticssetup [⟨module⟩] {⟨options⟩}`

This command lets you specify options for several tables or graphs. The options are set locally to the current group. Options for tables are in the `table` ⟨*module*⟩ and are the same as in the optional arguments of `\StatsTable`. Options for grapsh are in the `graph` ⟨*module*⟩ and are the same as in the optional arguments of `\StatsGraph`. You can also use `\statisticssetup` without a ⟨*module*⟩ and prefix all keys by the module name and a forward slash.

```
\statisticssetup{table/values=My values}
\statisticssetup[table]{counts=FooBar}
\StatsTable \facebook
```

My values	[0 ; 1[	[1 ; 2[	[2 ; 4[	[4 ; 7[	[7 ; 10[	[10 ; 14[
FooBar	1	3	10	30	18	12

1.3 Statistics tables

1.3.1 `\StatsTable` invocation

To typeset a table full of statistics values, you use the command:

`\StatsTable` `\StatsTable [⟨options$_1$⟩] {⟨data source⟩} [⟨options$_2$⟩]`

⟨*options*$_1$⟩ and ⟨*options*$_2$⟩ are both optional and taken into account. You will probably not use both at the same time even if `\StatsTable` will accept it (and apply ⟨*options*$_2$⟩ after ⟨*options*$_1$⟩, potentially overriding some settings). The idea is to let you decide where you feel the options should be. I find more logical to specify options after a `\macro` data source, but before an inline `{`⟨*data source*⟩`}`. Your mileage may vary.

If you do not use any option, you only get the line of values[2]:

```
\StatsTable \facebook
```

Values	[0 ; 1[	[1 ; 2[	[2 ; 4[	[4 ; 7[	[7 ; 10[	[10 ; 14[

OK, this is ugly. Let us add some reasonable amount of space (a better choice would be to use the cellprops package to control the spacing and a lot more):

```
\setlength\extrarowheight{1.5pt}
\StatsTable \facebook
```

Values	[0 ; 1[	[1 ; 2[	[2 ; 4[	[4 ; 7[	[7 ; 10[	[10 ; 14[

1.3.2 Choosing and naming rows

Let's add some rows to the table:

[2] The `\facebook` data source is defined on page 2.

values
counts
frequencies
icc
icf
dcc
dcf

`values [ = ⟨row header text⟩ ]`
`counts [ = ⟨row header text⟩ ]`
`frequencies [ = ⟨row header text⟩ ]`
`icc [ = ⟨row header text⟩ ]`
`icf [ = ⟨row header text⟩ ]`
`dcc [ = ⟨row header text⟩ ]`
`dcf [ = ⟨row header text⟩ ]`

These keys add the corresponding rows to the table. `icc` means increasing cumulative counts, `icf` is the same with frequencies, `dcc` is the row of decreasing cumulative counts and `dcf` for frequencies. If you omit ⟨*row header text*⟩ the key only activates the corresponding row; if you additionally use a value then the first cell of the row will use that value as text.

The initial header is `\valuename` for values, `\countname` for counts, `\freqname` for frequencies, `\iccname` for icc, `\icfname` for icf, `\dccname` for dcc and `\dcfname` for dcf.

```
\StatsTable \facebook[
                    values=Time in \si{h},
                    counts, frequencies, icc, dcc, icf, dcf
                ]
```

Time in h	[0 ; 1[	[1 ; 2[	[2 ; 4[	[4 ; 7[	[7 ; 10[	[10 ; 14[
Count	1	3	10	30	18	12
ICC	1	4	14	44	62	74
DCC	74	73	70	60	30	12
Frequency	1.4 %	4 %	13.5 %	40.6 %	24.3 %	16.2 %
ICF	1.4 %	5.4 %	18.9 %	59.5 %	83.8 %	100 %
DCF	100 %	98.6 %	94.6 %	81.1 %	40.5 %	16.2 %

novalues
nocounts
nofrequencies
noicc
nodcc
noicf
nodcf

`novalues, nocounts, nofrequencies, noicc, nodcc, noicf, nodcf`

If you want to *disable* a row you can use the `no`⟨*row*⟩ key. This is particularly useful for the `values` row, but you might need these keys to disable a row that you previously enabled with `\statisticssetup`.

```
\StatsTable \facebook [novalues, counts, icc]
```

Count	1	3	10	30	18	12
ICC	1	4	14	44	62	74

values/header
counts/header
frequencies/header
icc/header
icf/header
dcc/header
dcf/header

`values/header = ⟨row header text⟩`
`counts/header = ⟨row header text⟩`
`frequencies/header = ⟨row header text⟩`
`icc/header = ⟨row header text⟩`
`icf/header = ⟨row header text⟩`
`dcc/header = ⟨row header text⟩`
`dcf/header = ⟨row header text⟩`

These keys set the corresponding row header text, which will be used as the first cell of the row if the row is enabled. These keys does not enable their row by themselves, contrary to keys like `values` or `counts`.

The initial header is `\valuename` for values, `\countname` for counts, `\freqname` for frequencies, `\iccname` for icc, `\icfname` for icf, `\dccname` for dcc and `\dcfname` for dcf.

```
\statisticssetup{table/counts/header=People count}
\StatsTable \facebook[counts, frequencies, icc]
```

Values	[0 ; 1[	[1 ; 2[	[2 ; 4[	[4 ; 7[	[7 ; 10[	[10 ; 14[
People count	1	3	10	30	18	12
ICC	1	4	14	44	62	74
Frequency	1.4 %	4 %	13.5 %	40.6 %	24.3 %	16.2 %

1.3.3 Formatting cells

`values/format`
`counts/format`
`frequencies/format`
`icc/format`
`icf/format`
`dcc/format`
`dcf/format`

`values/format` = ⟨*formatting code*⟩
`counts/format` = ⟨*formatting code*⟩
`frequencies/format` = ⟨*formatting code*⟩
`icc/format` = ⟨*formatting code*⟩
`icf/format` = ⟨*formatting code*⟩
`dcc/format` = ⟨*formatting code*⟩
`dcf/format` = ⟨*formatting code*⟩

Each key in this list takes a value which will be used for each cell in the corresponding row. In this value, every occurrence of `#1` will be replaced by the content of the cell, which can be further configured by the `allcounts/format` key (for the rows `counts`, `icc` and `dcc`) or the `allfreqs/format` key (for the rows `frequencies`, `icf` and `dcf`). The idea is that the latter keys are intended for number formatting (decimal count, decimal separator, etc.) while the ⟨*row*⟩`/format` keys are intended for font/color changes. In this key, `\currentcolumn` expands to the data column number, starting from 1, to enable different formatting depending on the column. These keys are all initially equal to `#1` which means they pass-through the content unmodified.

```
\StatsTable \facebook[
  counts, icc,
  icc/format = \colorbox{blue!\currentcolumn 0!white}{#1}
]
```

Values	[0 ; 1[	[1 ; 2[	[2 ; 4[	[4 ; 7[	[7 ; 10[	[10 ; 14[
Count	1	3	10	30	18	12
ICC	1	4	14	44	62	74

`allcounts/format`

`allcounts/format` = ⟨*formatting code*⟩

This key take some formatting code, in which every occurrence of `#1` will be replaced by the integer count[3] in each cell of every row containing counts. The initial value is `\num{#1}`, using the siunitx package.

The result of this formatting code will then be passed to `counts/format`, `icc/format` or `dcc/format` depending on the row, for further parsing and formatting.

```
\StatsTable \facebook[
  counts, icc,
  icc/format = \colorbox{blue!\currentcolumn 0!white}{#1},
  allcounts/format = {\num[round-integer-to-decimal,
                           round-mode=figures]{#1}}
]
```

Values	[0 ; 1[	[1 ; 2[	[2 ; 4[	[4 ; 7[	[7 ; 10[	[10 ; 14[
Count	1.0	3.0	10	30	18	12
ICC	1.0	4.0	14	44	62	74

[3] As returned by `\fp_use:N` or `\fp_eval:n`.

`allfreqs/format` `allfreqs/format = ⟨formatting code⟩`

This key take some formatting code, in which every occurrence of `#1` will be replaced by the current frequency[4] in each cell of every row containing frequencies. The initial value is `\num{#1}`, using the siunitx package.

The result of this formatting code will then be passed to `freqs/format`, `icf/format` or `dcf/format` depending on the row, for further parsing and formatting.

The initial value is set by the `allfreqs/format/percent` key and typesets values in percentage (that is, multiplied by 100 with a trailing %).

```
\StatsTable \facebook[
  icc, frequencies, icf,
  allfreqs/format = {\num[round-mode=places,
                         round-integer-to-decimal,
                         round-precision=3]{#1}}
]
```

Values	[0 ; 1[	[1 ; 2[	[2 ; 4[	[4 ; 7[	[7 ; 10[	[10 ; 14[
ICC	1	4	14	44	62	74
Frequency	0.014	0.040	0.135	0.406	0.243	0.162
ICF	0.014	0.054	0.189	0.595	0.838	1.000

Note that if you use `allfreqs/format` to round the frequencies to an acceptable precision, your frequencies might not add up to 1 anymore, and summing the frequencies up to some value might not give the same result as computing the cumulative frequency from the cumulative count. If you want to avoid that, consider using the `digits` key of the `table` module, which rounds the cumulative frequencies *then* computes the individual frequencies as differences of consecutive cumulative ones. This essentially spreads the rounding errors so that they cancel each other, with a result not unlike that of the Bresenham algorithm.

`allfreqs/format/percent` `allfreqs/format/percent`

This key sets up `allfreqs/format` to display the frequencies as percentages, that is, multiplied by 100 with a trailing %. This is the initial setting.

TEXhackers note: This key is a shorthand for `allfreqs/format = \SI{\fp_eval:n{#1*100}}{\percent}`.

```
\StatsTable \facebook[ frequencies, icf, allfreqs/format/percent ]
```

Values	[0 ; 1[	[1 ; 2[	[2 ; 4[	[4 ; 7[	[7 ; 10[	[10 ; 14[
Frequency	1.4 %	4 %	13.5 %	40.6 %	24.3 %	16.2 %
ICF	1.4 %	5.4 %	18.9 %	59.5 %	83.8 %	100 %

`allfreqs/format/real` `allfreqs/format/real`

This key sets up `allfreqs/format` to `\num{#1}` which displays the frequencies as straight real numbers.

```
\StatsTable \facebook[ frequencies, icf, allfreqs/format/real ]
```

Values	[0 ; 1[	[1 ; 2[	[2 ; 4[	[4 ; 7[	[7 ; 10[	[10 ; 14[
Frequency	0.014	0.04	0.135	0.406	0.243	0.162
ICF	0.014	0.054	0.189	0.595	0.838	1

[4] As returned by `\fp_use:N` or `\fp_eval:n`.

digits `digits = ⟨integer⟩`

This key sets the number of digits after the decimal point to use for rounding cumulative frequencies. Point-wise frequencies are computed from these rounded cumulative frequencies to ensure consistency with the cumulative counts, and ensure the sum of frequencies equals 1. This essentially spreads the rounding errors so that they cancel each other, with a result not unlike that of the Bresenham algorithm.

The rounding takes place before any formatting by `allfreqs/format` or individual ⟨*row*⟩`/format`. The initial value is 3 (which means one digit after the decimal separator in percentage).

```
\StatsTable \facebook[ frequencies, icf, digits=2 ]
```

Values	[0 ; 1[	[1 ; 2[	[2 ; 4[	[4 ; 7[	[7 ; 10[	[10 ; 14[
Frequency	1 %	4 %	14 %	40 %	25 %	16 %
ICF	1 %	5 %	19 %	59 %	84 %	100 %

1.3.4 Hiding and showing column contents

In addition to ⟨*row*⟩`/format`, `allcounts/format` and `allfreqs/format` which can all use `\currentcolumn` to apply different formatting to different columns, you can also use the following keys:

showonly `showonly = ⟨integer and integer range list⟩`
showonly/hidden `showonly/hidden = ⟨formatting code⟩`
showonly/shown `showonly/shown = ⟨formatting code⟩`

The `showonly` key enables you to choose which columns you want *shown* — and thus which ones you want to have their contents hidden. It takes a comma-separated list of single numbers or ⟨*start*⟩`-`⟨*end*⟩ ranges of numbers. An empty value means *show everything*, and this is the initial value. To hide all contents, you can set `showonly` to a non-existent column number like 0.

Every column whose number is in the `showonly` list (of ranges) is deemed *shown*, which means all cells will be ultimately wrapped in the `showonly/shown` formatting code, where as usual `#1` is replaced by the contents. That key initially just passes through the contents as-is.

Every column whose number is *not* in the list is *hidden*, *i.e.* its cell contents are wrapped in the `showonly/hidden` formatting code. This key is initially empty which means the contents are ignored and the cell stays empty — which means its width will collapse and only the column separation will remain. You can decide to still typeset the contents in white, or even put them in a PDF "OCG layer" with the `ocgx2` package for instance.

```
\StatsTable \facebook[ counts, frequencies, showonly={2,4-6} ]
\StatsTable \facebook[ counts, frequencies, showonly={2,4-6},
                       showonly/hidden = \color{white}#1 ]
```

Values	[1 ; 2[	[4 ; 7[	[7 ; 10[	[10 ; 14[
Count	3	30	18	12
Frequency	4 %	40.6 %	24.3 %	16.2 %

Values	[1 ; 2[	[4 ; 7[	[7 ; 10[	[10 ; 14[
Count	3	30	18	12
Frequency	4 %	40.6 %	24.3 %	16.2 %

1.3.5 Formatting the table

maxcols

`maxcols` = ⟨*comma-separated list of integers*⟩

Setting this key to a positive integer n makes `\StatsTable` wrap after having added n columns to the current table. The table is closed, and a new one is created with the row headers typeset anew. Setting this key to a negative number or zero disables wrapping. If you set the key to a list of integers, each one is used as the value for the corresponding subtable, with the last number staying in effect for all remaining subtables. The initial value is 0.

TEXhackers note: If there is a non-positive integer in the list, all subsequent integers are ignored since there will be no further wrapping thus no other subtable.

tablesep

`tablesep` = ⟨*TEX content*⟩

This key holds some TEX content that will be inserted after each table when wrapping. It should probably contain something that creates a line return (either `\\` or `\par`), but can contain arbitrary code. The initial value is `\\`.

```
\StatsTable \facebook[ counts, maxcols=4,
                       tablesep=\par{\color{red}\hrule} ]
```

Values	[0 ; 1[	[1 ; 2[	[2 ; 4[	[4 ; 7[
Count	1	3	10	30

Values	[7 ; 10[	[10 ; 14[
Count	18	12

preline

`preline` = ⟨*array content*⟩

This key holds some TEX content that will be inserted first in the `array` environment, before any row content. It should probably be some kind of `\noalign` material, like a `\hline` or similar constructs. The initial value is `\firsthline`, with a fallback to `\hline` if the former doesn't exist.

postline

`postline` = ⟨*array content*⟩

This key holds some TEX content that will be inserted last in the `array` environment, after any row content. It should probably be some kind of `\noalign` material, like a `\hline` or similar constructs. The initial value is `\lasthline`, with a fallback to `\hline` if the former doesn't exist.

outline

`outline` = ⟨*array content*⟩

This key sets both `preline` and `postline` to the same value.

newline

`newline` = ⟨*array content*⟩

This key holds some TEX content that will be inserted at the end of each row, to separate it from the next. *It should contain some kind of* `\cr`, probably in the form of `\\`, but can also contain `\hline`s after the `\\`. The initial value is `\\` which creates tables without lines separating rows (as booktabs would recommend).

```
\setlength\extrarowheight{1ex}
\StatsTable \facebook[ counts, preline=\hline\hline,
                       postline=\hline\hline\hline,
                       newline=\\[1ex]\hline ]
```

Values	[0 ; 1[	[1 ; 2[	[2 ; 4[	[4 ; 7[	[7 ; 10[	[10 ; 14[
Count	1	3	10	30	18	12

`coltype` `coltype = ⟨preamble elements⟩`

This key sets the part of the array preamble that will be repeated for each content column in the table. It can contain any preamble content, like `|` for vertical lines, but should only countain a single column specifier. The initial value is `c`.

`headcoltype` `headcoltype = ⟨preamble elements⟩`

This key sets the part of the array preamble that will be used for the first column in the table, which contains the headers. It can contain any preamble content, like `|` for vertical lines, but should only countain a single column specifier. The initial value is `l`.

```
\StatsTable \facebook[ counts, coltype=@{}c, headcoltype=r ]
```

Values	[0 ; 1[	[1 ; 2[	[2 ; 4[	[4 ; 7[	[7 ; 10[	[10 ; 14[
Count	1	3	10	30	18	12

Note: these keys are here for convenience, but if you find yourself trying to do very clever things in them, you should consider using the cellprops package which is able to do much more complex border and background layouts with ease. In particular they probably shouldn't be used to workaround the very poor spacing of `array`: there are better solutions.

Several classic uses of these keys can be replaced by the following key:

`frame` `frame = none | clean | full`

The `frame` key selects a preset for `preline`, `postline`, `headcoltype` and `coltype`. The possible presets are:

- `none`: clears `preline` and `postline`, sets `headcoltype = l` and `coltype = c`. This removes all lines in the table and is useful if you use other means like cellprops to style the table.
- `clean`: sets `preline = \firsthline`, `postline = \lasthline`, `headcoltype = l` and `coltype = c`. This corresponds to the initial setting, and yields a layout similar to booktabs recommendations, especially if you set `\firsthline` and `\lasthline` to be a little thicker.
- `full`: sets `preline = \firsthline`, `postline = \lasthline`, `headcoltype = |l|` and `coltype = c|`. This separates all cells with rules.

```
\statisticssetup{table/showonly/hidden=\color{white}#1}
\StatsTable \facebook[ counts, frequencies, frame=none ]
\StatsTable \facebook[ counts, frequencies, frame=full, showonly=2-4 ]
```

Values	[0 ; 1[	[1 ; 2[	[2 ; 4[	[4 ; 7[	[7 ; 10[	[10 ; 14[
Count	1	3	10	30	18	12
Frequency	1.4 %	4 %	13.5 %	40.6 %	24.3 %	16.2 %

Values		[1 ; 2[	[2 ; 4[	[4 ; 7[		
Count		3	10	30		
Frequency		4 %	13.5 %	40.6 %		

`valign` `valign = t | c | b`

The value of this key is used for the optional argument of the `array` environment. This enables to align either the baseline of the first line, that of the last line, or the vertical center of the table with the surrounding baseline. The initial value is `t`.

1.4 Statistics graphs

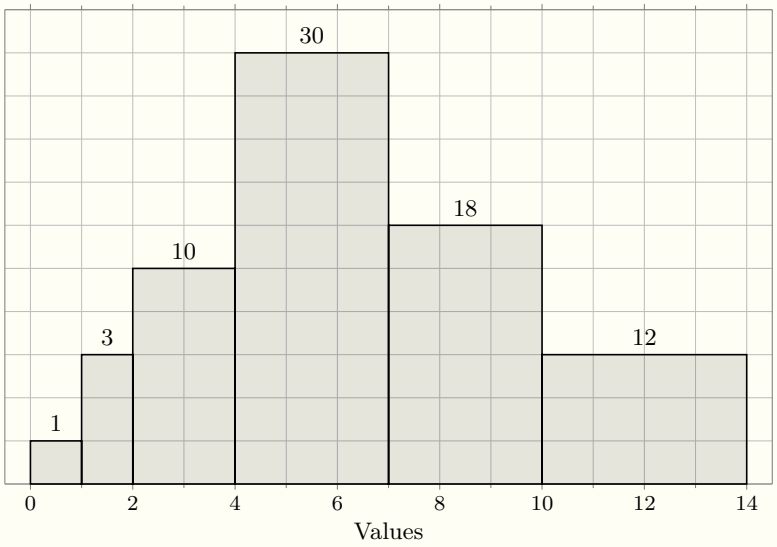
1.4.1 `\StatsGraph` invocation

To typeset a graphic from the statistics values, you use the command:

`\StatsGraph` `\StatsGraph [⟨options1⟩] {⟨data source⟩} [⟨options2⟩]`

⟨*options*$_1$⟩ and ⟨*options*$_2$⟩ are both optional and taken into account. You will probably not use both at the same time even if `\StatsGraph` will accept it (and apply ⟨*options*$_2$⟩ after ⟨*options*$_1$⟩, potentially overriding some settings). The idea is to let you decide where you feel the options should be. I find more logical to specify options after a `\macro` data source, but before an inline `{`⟨*data source*⟩`}`. Your mileage may vary.

```
\StatsGraph \facebook
```

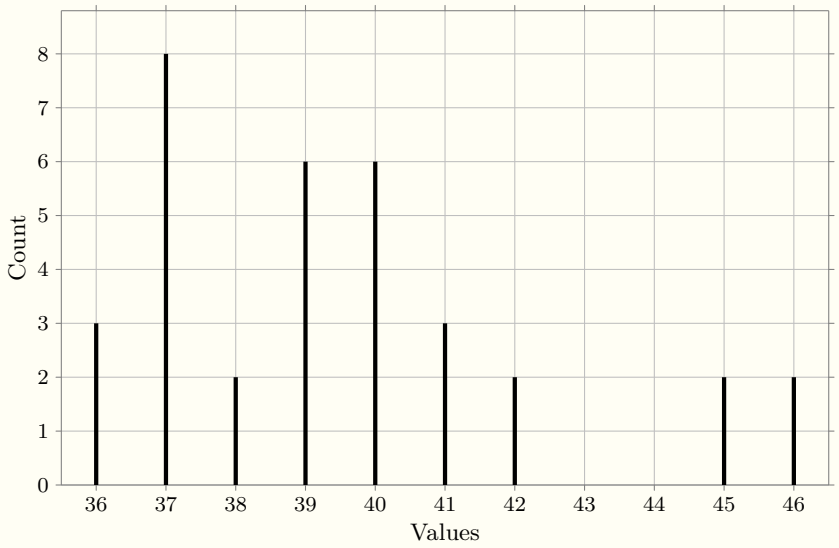


`\StatsGraph` will draw a different kind of graph depending on the ⟨*data source*⟩ itself, and the `cumulative` option key. A summary is shown in the table below:

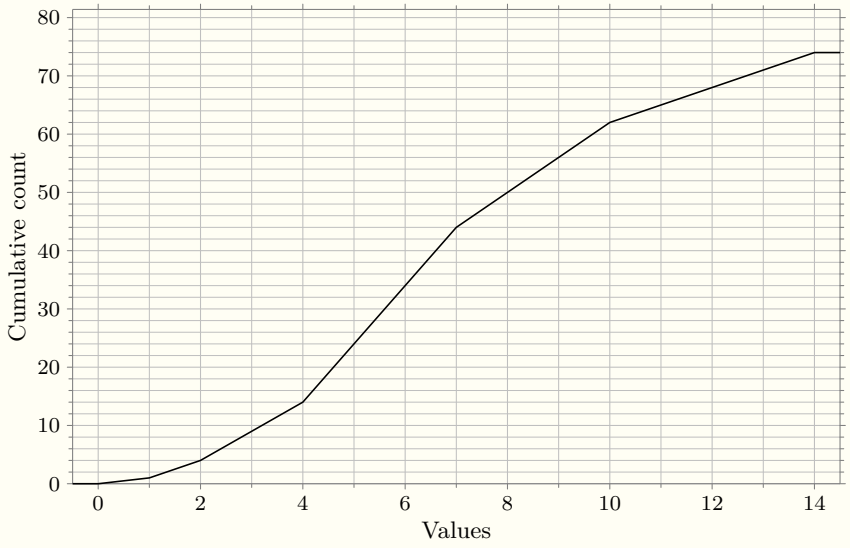
values are ranges	without `cumulative`	with `cumulative`
no	bar diagram[5]	*not implemented yet*
yes	histogram	cumulative distribution function

[5] In this documentation this is called a *comb graph*.

```
\def \combdata { 36=3, 37=8, 38=2, 39=6, 40=6, 41=3, 42=2, 45=2, 46=2 }
\StatsGraph \combdata
```




```
\StatsGraph \facebook [cumulative]
```



1.4.2 TikZ picture and datavisualization settings

`picture`
`picture/reset`

`picture` = ⟨*TikZ key options*⟩
`picture/reset`

The `picture` key *appends* content to the optional argument of the `tikzpicture` environment. It can contain any list of TikZ keys. The `picture/reset` key clears all content accumulated by the `picture` key, including the initial value.

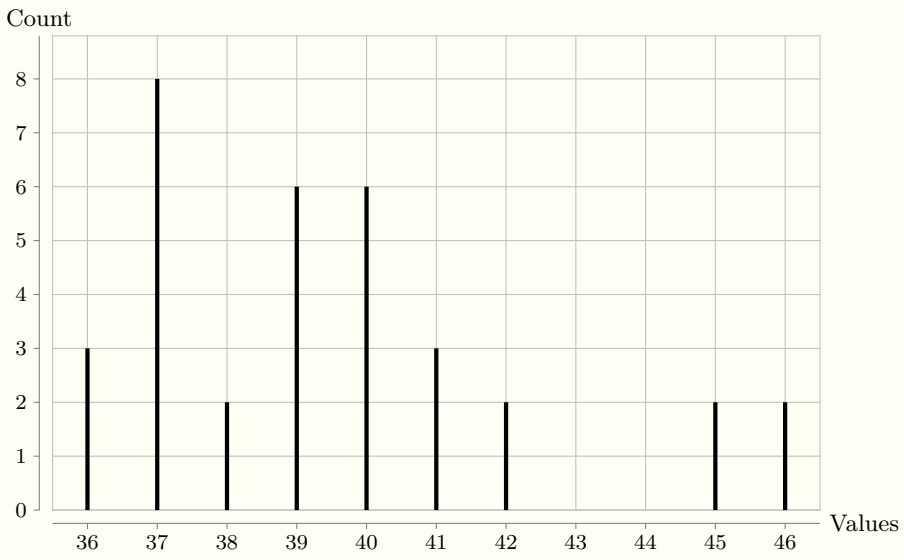
The initial value is:
`baseline = (current bounding box.center), label position = right`.

`axissystem`
`axissystem/reset`

`axissystem` = ⟨*TikZ cartesian axis system options*⟩
`axissystem/reset`

The `axissystem` key adds keys to the list of options passed to the `scientific axes` datavisualization key, The `axissystem/reset` key clears all content accumulated by the `axissystem` key, including the initial value, which is set by the initial value of the `width` key.

```
\StatsGraph \combdata [axissystem={end labels, clean}]
```



Two small helper keys are provided for a very common usage of `axissystem`:

`width` `width = ⟨TEX dimension expression⟩`

This key sets the width of the graphic to the given ⟨*TEX dimension expression*⟩, labels and padding excluded. The expression is evaluated at graph creation time. The initial value is `0.75\columnwidth`.

TEXhackers note: This key is a shortcut for `axissystem = { width =` ⟨*dimension*⟩ `}`

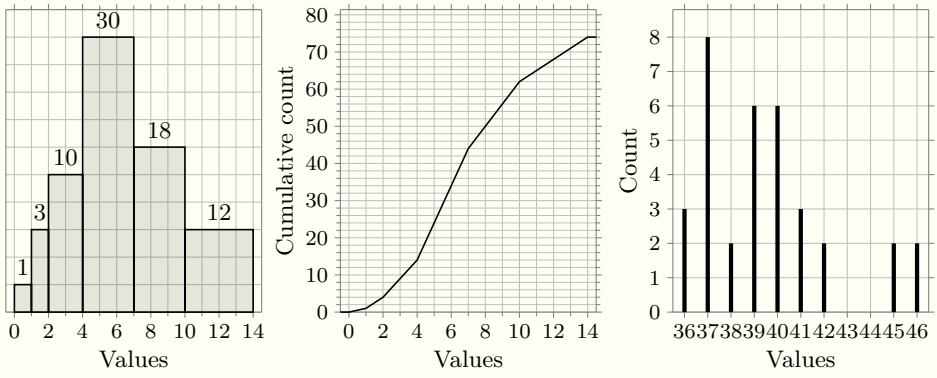
`height` `height = ⟨TEX dimension expression⟩`

This key sets the width of the graphic to the given ⟨*TEX dimension expression*⟩, labels and padding excluded. The expression is evaluated at graph creation time. Initially this is *unset*, which means the default of the cartesian axis system will be used, that is the choosen width divided by the golden ratio $\varphi = \frac{1+\sqrt{5}}{2}$.

TEXhackers note: This key is a shortcut for `axissystem = { height =` ⟨*dimension*⟩ `}`

To have more precise control over the scale of the graph, you can use the individual axis options provided by statistics to set an explicit scaling with TikZ DataVisualization keys like `unit length`. See the PGF/TikZ manual for more information.

```
\statisticssetup[graph]{ width = 0.25\columnwidth, height=4cm }
\centering
\StatsGraph \facebook
\StatsGraph \facebook [cumulative]
\StatsGraph \combdata
```



`tikzinfo'`
`tikzinfo'/reset`

`tikzinfo' =` ⟨*TikZ picture code*⟩
`tikzinfo'/reset`

This key *appends* content to be added in the `info'` section of the `\datavisualization` command. It can contain any TikZ code, and can use the `visualization cs` coordinate system. The result of this TikZ code is drawn *before* the data itself and will end up behind unless you play with TikZ layers. Some information might be unavailable or wrong since the data has not been drawn yet.

The `tikzinfo'/reset` key clears all content accumulated by the `tikzinfo'` key. The initial value is empty.

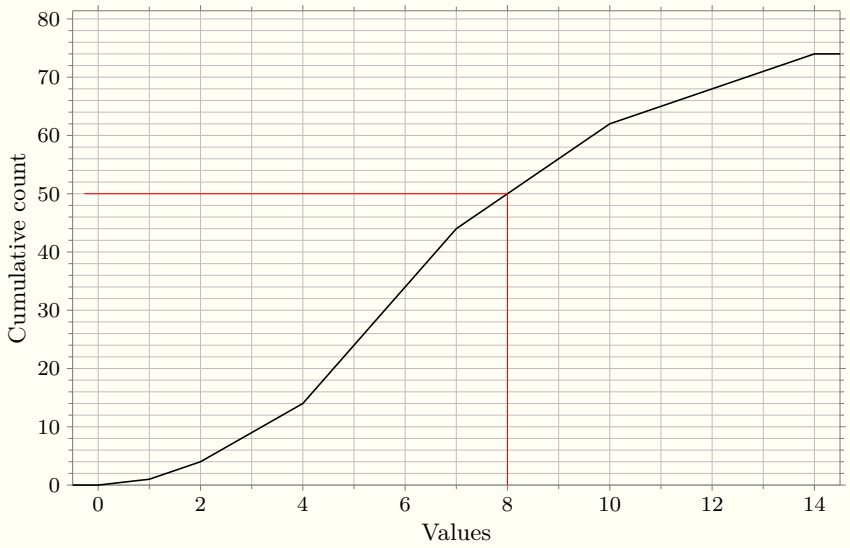
`tikzinfo`
`tikzinfo/reset`

`tikzinfo =` ⟨*TikZ picture code*⟩
`tikzinfo/reset`

This key *appends* content to be added in the `info` section of the `\datavisualization` command. It can contain any TikZ code, and can use the `visualization cs` coordinate system. The result of this TikZ code is drawn *after* the data itself and will end up in front of it unless you play with TikZ layers.

The `tikzinfo/reset` key clears all content accumulated by the `tikzinfo` key. The initial value is empty.

```
\StatsGraph \facebook [
  cumulative,
  tikzinfo = {
      \path (data bounding box.south west) coordinate (O);
      \path (visualization cs:x=8, y=50) coordinate (A);
      \draw[red] (O |- A) -- (A) -- (A |- O);
  }
]
```



1.4.3 Styling the graph

style
style/reset
comb/style
comb/style/reset
histogram/style
histogram/style/reset
cumulative/style
cumulative/style/reset

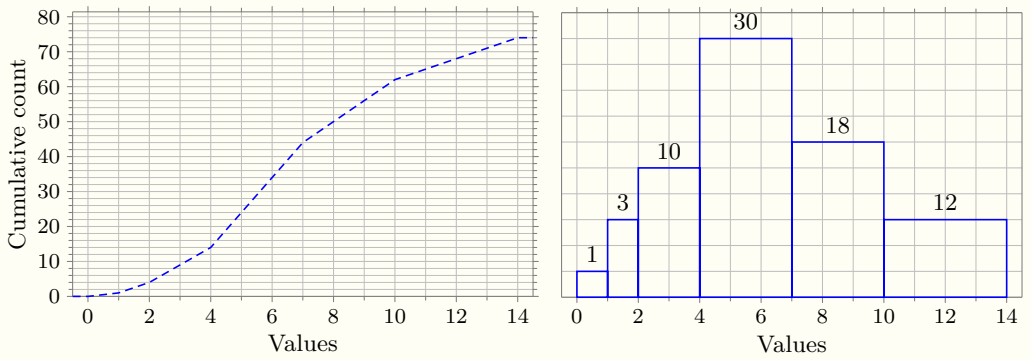
`style = ⟨TikZ path options⟩`
`⟨graph type⟩/style = ⟨TikZ path options⟩`
`style/reset, ⟨graph type⟩/style/reset`

The ⟨*graph type*⟩`/style` keys append options to the TikZ path created by the datavisualization when the corresponding graph type is used. You can clear these options with ⟨*graph type*⟩`/style/reset`. If you omit the graph type, this sets the label for all graph types simultaneously.

The initial values are:

```
comb/style = ultra-thick,
cumulative/style = %empty
histogram/style  = {
    every~path/.prefix~style=fill,
    semithick, black, fill=black, fill~opacity=0.1
},
```

```
\statisticssetup[graph]{width=0.45\linewidth,
                        style=blue, cumulative/style=densely dashed }
\StatsGraph \facebook [ cumulative ]
\hfill \StatsGraph \facebook[style={
  fill opacity=0, pattern=north west lines,
}]
```



1.4.4 Selecting which parts of the graph are shown

By default, the complete graph is shown; you can ask `\StatsGraph` to only show the parts corresponding to some of the input data:

showonly

`showonly = ⟨integer and integer range list⟩`

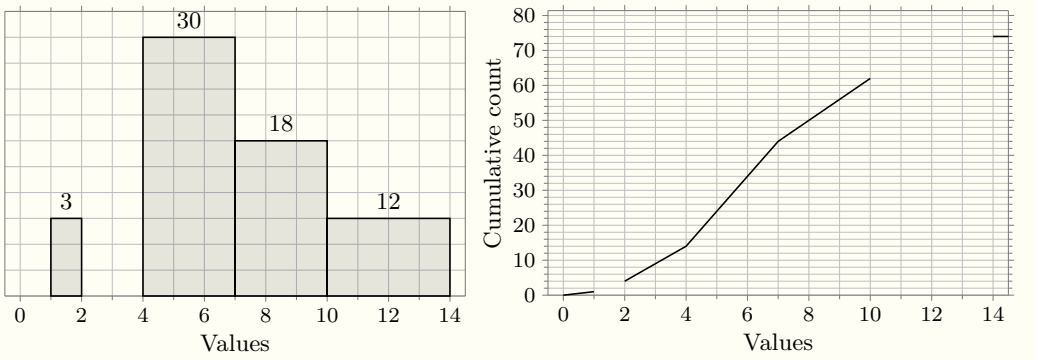
The `showonly` key enables you to set which parts of the graph you want *shown*. It takes a comma-separated list of single numbers or ⟨*start*⟩`-`⟨*end*⟩ ranges of numbers. An empty value means *show everything*, and this is the initial value. To hide all contents, you can set `showonly` to a non-existent part number like -1.

For comb graphs, the n-th part is the vertical bar corresponding to the n-th value in the data source. For histograms, this is the rectangle corresponding to the n-th range.

For cumulative distribution functions of data sources with ranges, this is the direct image of the n-th range by the function. The horizontal segment between $-\infty$ and the lower bound of the first range is assigned number 0, and the part right of the last range is selected by number $N+1$ where N is the total number of ranges.

Currently, the drawing of hidden parts is inhibited altogether, but in the future it is planned to have them drawn with another visualizer and a separate style.

```
\statisticssetup{ graph/width=0.45\columnwidth }
\StatsGraph \facebook [ showonly={2,4-6} ]
\StatsGraph \facebook [ cumulative, showonly={1,3-5,7} ]
```



1.4.5 Unit selection and vertical axis settings

`counts`
`frequencies`

`counts` [= ⟨*label*⟩]
`frequencies` [= ⟨*label*⟩]

These keys select the corresponding unit to use for the vertical axis of comb graphs and cumulative distribution graphs, and for the area display of histograms. Additionnally, if a ⟨*label*⟩ is provided, it is passed to the `counts/label` or the `frequencies/label` key.

The initially selected unit is `counts`.

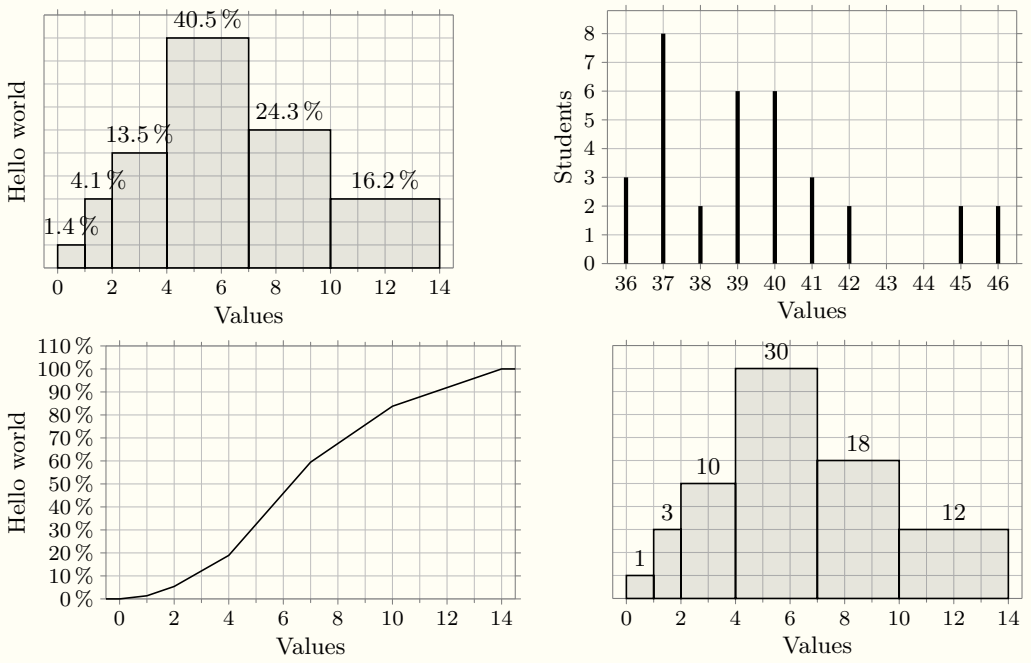
`comb/counts`
`comb/frequencies`
`histogram/counts`
`histogram/frequencies`
`cumulative/counts`
`cumulative/frequencies`

⟨*graph type*⟩`/counts` [= ⟨*label*⟩]
⟨*graph type*⟩`/frequencies` [= ⟨*label*⟩]

These keys select the unit to use for specific types of graphs separately. They can be used in the inline options of `\StatsGraph` too, but they probably only make sense in `\statisticssetup` to define different defaults for different graph types.

TEXhackers note: The `counts` key is actually a meta-key for `comb/counts, histogram/counts, cumulative/counts`, which applies the same value (or no value at all) to all three type-specific keys. The `frequencies` key is similar.

```
\statisticssetup[graph]{
  width=0.4\columnwidth,
  frequencies=Hello world, comb/counts=Students
}
\StatsGraph \facebook              \hfill \StatsGraph \combdata \\
\StatsGraph \facebook [cumulative] \hfill \StatsGraph \facebook[counts]
```



Note that setting a label for the vertical axis of histogram does not make much sense, even if your decision will be respected.

counts/label
frequencies/label
comb/counts/label
comb/frequencies/label
histogram/counts/label
histogram/frequencies/label
cumulative/counts/label
cumulative/frequencies/label

⟨*unit*⟩/label = ⟨*label*⟩
⟨*graph type*⟩/⟨*unit*⟩/label = ⟨*label*⟩

These keys set the label to use for the y axis of the graph when the corresponding unit is selected, *without* selecting it at that point. This is useful to provide your own defaults through `\statisticssetup`.

The keys `counts/label` and `frequencies/label` set the label for all three graph types, while the others are here to set individual defaults.

Initial values are as follows:

- `comb/counts/label = \countname`
- `comb/frequencies/label = \freqname`
- `cumulative/counts/label = \ccountname`
- `cumulative/frequencies/label = \cfreqname`
- `histogram/counts/label` and `histogram/frequencies/label` are unset

TEXhackers note: The ⟨*type*⟩/⟨*unit*⟩/label key is a shorthand for ⟨*type*⟩/⟨*unit*⟩/axis = { label = ⟨*label*⟩ }, which means that using ⟨*type*⟩/⟨*unit*⟩/axis/reset will also remove any defined label.

TEXhackers note: As before, ⟨*unit*⟩/label = ⟨*label*⟩ is equivalent to comb/⟨*unit*⟩/label = ⟨*label*⟩, histogram/⟨*unit*⟩/label = ⟨*label*⟩, cumulative/⟨*unit*⟩/label = ⟨*label*⟩

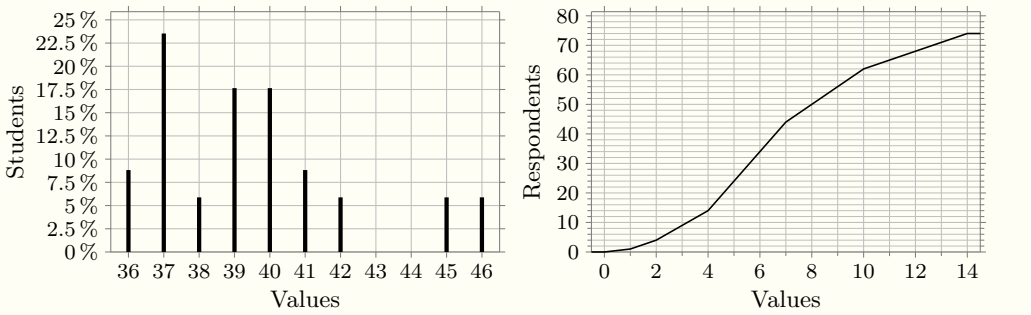
y/label
comb/y/label
histogram/y/label
cumulative/y/label

`y/label = ⟨label⟩`
`⟨graph type⟩/y/label = ⟨label⟩`

These keys set the label to use for the y axis of the graph for both units at the same time. `y/label` sets the label for all graph types and all units simultaneously, while ⟨*graph type*⟩`/y/label` can be used for individual graph types.

This can be useful to set the label in inline options without having to explicitely type the graph type or the selected unit:

```
\statisticssetup[graph]{
  width=0.38\columnwidth,
  comb/frequencies, cumulative/counts,
}
\StatsGraph \combdata [ y/label=Students ]
\StatsGraph \facebook [ cumulative, y/label=Respondents ]
```



counts/axis
frequencies/axis
comb/counts/axis
comb/frequencies/axis
histogram/counts/axis
histogram/frequencies/axis
cumulative/counts/axis
cumulative/frequencies/axis
counts/axis/reset
frequencies/axis/reset
comb/.../axis/reset
histogram/.../axis/reset
cumulative/.../axis/reset

⟨*unit*⟩`/axis = ⟨`*TikZ datavisualization axis options*`⟩`
⟨*unit*⟩`/axis/reset`
⟨*graph type*⟩`/⟨`*unit*`⟩/axis = ⟨`*TikZ datavisualization axis options*`⟩`
⟨*graph type*⟩`/⟨`*unit*`⟩/axis/reset`

The ⟨*unit*⟩`/axis` keys append options to the TikZ y axis when the corresponding unit is selected. You can clear these options with ⟨*unit*⟩`/axis/reset`. The ⟨*graph type*⟩`/⟨`*unit*`⟩/axis` and ⟨*graph type*⟩`/⟨`*unit*`⟩/axis/reset` keys do the same, but only for a specific graph type.

Initial values are as follows:

- `comb/counts/axis` and `cumulative/counts/axis` are equal to `ticks and grid={many, int about strategy, integer minor steps*}, label=`⟨*initial value of the label key*⟩
- `cumulative/counts/axis` and `cumulative/frequencies/axis` are equal to `ticks and grid=many, label=`⟨*initial value of the label key*⟩
- `histogram/counts/axis` and `histogram/frequencies/axis` are equal to `ticks=none, grid=`⟨*code to auto-compute the step*⟩ (see the `histogram/autostep` key below).

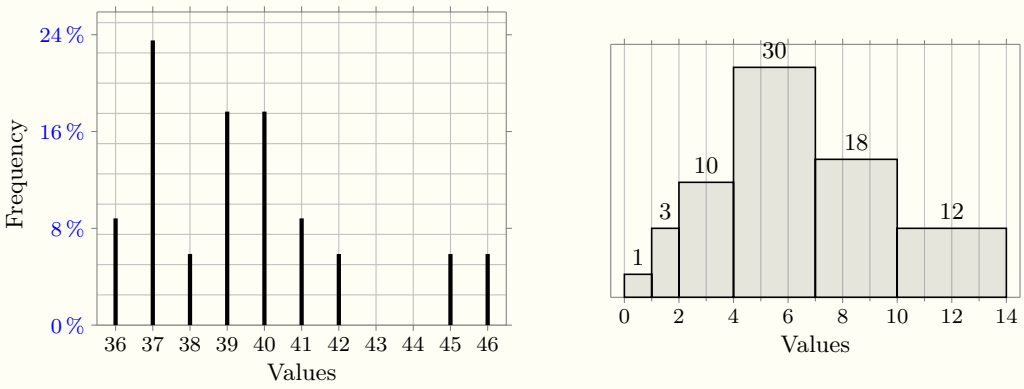
y/axis
y/axis/reset
comb/axis
comb/axis/reset
histogram/axis
histogram/axis/reset
cumulative/axis
cumulative/axis/reset

`y/axis = ⟨`*TikZ datavisualization axis options*`⟩`
`y/axis/reset`
⟨*graph type*⟩`/y/axis = ⟨`*TikZ datavisualization axis options*`⟩`
⟨*graph type*⟩`/y/axis/reset`

The `y/axis` keys append options to the TikZ y axis for all possible units and all graph types at the same time. The `y/axis/reset` key clears these options for all units and all types simultaneously.

The ⟨*graph type*⟩`/y/axis` and ⟨*graph type*⟩`/y/axis/reset` keys do the same, but only for a specific graph type.

```
\statisticssetup[graph]{
  width=0.4\columnwidth,
  comb/frequencies/axis = { ticks={step=0.08} },
  histogram/y/axis = { grid = none },
}
\StatsGraph \combdata [ frequencies, y/axis = {
  ticks={style=blue}, unit length=4cm per 0.25 units,
} ]
\hfill \StatsGraph \facebook
```



`/tikz/datavisualization/integer minor steps` `integer minor steps` [= ⟨*integer expression*⟩]
`/tikz/datavisualization/integer minor steps*` `integer minor steps*` [= ⟨*integer expression*⟩]

These are not keys in the `graph` module, but TikZ keys. They add code to automatically compute `minor steps between steps` after the axis step has been computed with the choosen strategy, so that the following constraints are respected:

- a minor step corresponds to an integer number;
- at most ⟨*integer expression*⟩ ticks are present on the axis (minor and major included, subminor not counted).

In addition, the starred version ensures that the major step is never below one, which makes sense for counts where sub-unit graduations are confusing at best.

If ommited, the ⟨*integer expression*⟩ defaults to 50.

These TikZ keys should not explode if the computed step is not an integer, but will probably not give a useful result, and in particular whether the minor step will be integer is not defined in that case.

TEXhackers note: The keys are independent of statistics and could be reused elsewhere.

counts/format
frequencies/format
y/format
comb/counts/format
comb/frequencies/format
comb/y/format
histogram/counts/format
histogram/frequencies/format
histogram/y/format
cumulative/counts/format
cumulative/frequencies/format
cumulative/y/format

⟨*unit*⟩/format = ⟨*formatting code*⟩
⟨*graph type*⟩/⟨*unit*⟩/format = ⟨*formatting code*⟩

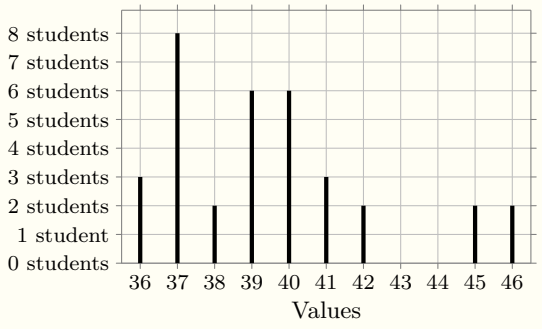
These keys set the format to use for all counts or frequenties that are typeset on the graphs. This includes the ticks on axes, and areas above histogram rectangles. The value should be TEX code to render the actual number, in which all occurrences of `#1` are replaced by the number to typeset.

Keys of the form `\meta{graph type}/\meta{unit}/format` are used to set the formatter of numbers in a specific unit when used in a specific graph. Keys of the form `\meta{unit}/format` set the formatter for all graph types at the same time, which is often desirable since it is rare that a frequency needs to be typeset differently in *e.g.* comb graphs and histograms.

You can use `\meta{graph type}/y/format` or `y/format` to set the formatter for both units at the same time, which is mainly useful for inline options to avoid repeating the selected unit for each key.

Initial settings are: `counts/format = \num{#1}` and `frequencies/format/percent` (see below for an exlpanation of that key).

```
\StatsGraph \combdata [
      y/label=, width=0.4\columnwidth,
      y/format=#1\text{ student\ifnum#1=1\else s\fi}
]
```



frequencies/format/real
comb/frequencies/format/real
histogram/frequencies/format/real
cumulative/frequencies/format/real

frequencies/format/real = ⟨*number of decimals*⟩
⟨*graph type*⟩/frequencies/format/real = ⟨*number of decimals*⟩

These keys make the corresponding format typeset its argument as a real number, using the `\num` command of the siunitx package.

TEXhackers note: This is equivalent to:

```
frequencies/format = \num[round-mode=places,round-precision=##1]{####1}
```

frequencies/format/percent
comb/frequencies/format/percent
histogram/frequencies/format/percent
cumulative/frequencies/format/percent

frequencies/format/percent = ⟨*number of decimals*⟩
⟨*graph type*⟩/frequencies/format/percent = ⟨*number of decimals*⟩

These keys make the corresponding format typeset its argument as a percentage, using the `\num` command of the siunitx package. This is the initial setting.

TEXhackers note: This is equivalent to:

```
frequencies/format = { \SI[round-mode=places,round-precision=##1]{
     \fp_eval:n{####1*100}
}{\percent}
```

counts/margin
frequencies/margin
y/margin
comb/counts/margin
comb/frequencies/margin
comb/y/margin
histogram/counts/margin
histogram/frequencies/margin
histogram/y/margin
cumulative/counts/margin
cumulative/frequencies/margin
cumulative/y/margin

⟨*unit*⟩/margin = ⟨*numeric expression*⟩
⟨*graph type*⟩/⟨*unit*⟩/margin = ⟨*numeric expression*⟩

These keys set the margin that will be used for the relevant axis in the corresponding graph type, that is the amount of space above the data that will be reserved by `\StatsGraph`. The ⟨*numeric expression*⟩ should compute a count or a frequency depending on the selected unit, and will correspond to the empty space reserved above the graph *in this very unit*.

In this expression, the following constants will be available: `\min` which is the minimum count or frequency where something is drawn in the graph (currently this is always zero); `\max` which is the maximum count or frequency in the graph; and `\range` which is `\max - \min`.

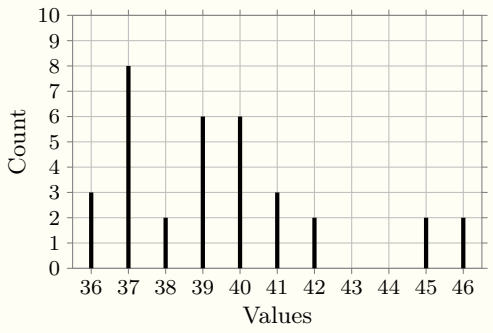
As usual, keys of the form `\meta{graph type}/\meta{unit}/margin` are used to define the margin in a specific unit when used in a specific graph, whereas keys of the form `\meta{unit}/margin` set the margin for all graph types at the same time.

You can use `\meta{graph type}/y/margin` or `y/margin` to set the margin for both units at the same time, which is mainly useful for inline options to avoid repeating the selected unit for each key.

The inital value is `y/margin = \range / 10`.

TEXhackers note: This expression will be evaluated with the rules of `\fp_eval:n` (with `\fp_gset:Nn` to be exact).

```
\StatsGraph \combdata [ width=0.4\columnwidth, y/margin=2 ]
```



1.4.6 Horizontal axis settings

values/label
x/label
comb/values/label
comb/x/label
histogram/values/label
histogram/x/label
cumulative/values/label
cumulative/x/label

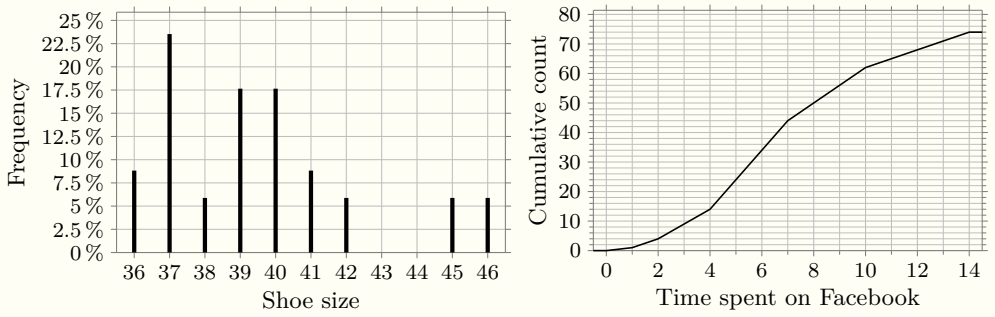
values/label = ⟨*label*⟩, x/label = ⟨*label*⟩
⟨*graph type*⟩/values/label = ⟨*label*⟩
⟨*graph type*⟩/x/label = ⟨*label*⟩

These keys set the label to use for the x axis of the graph when the corresponding graph type is used. The keys with `x` are aliases for the similar keys with `values`. If you omit the graph type, this sets the label for all graph types simultaneously.

The initial value is `values/label = \valuename`.

TEXhackers note: The ⟨*type*⟩`/values/label` key is a shorthand for ⟨*type*⟩`/values/axis = { label =` ⟨*label*⟩ `}`, which means that using ⟨*type*⟩`/values/axis/reset` will also remove any defined label.

```
\statisticssetup[graph]{
  width=0.38\columnwidth,
  comb/frequencies, cumulative/counts,
}
\StatsGraph \combdata [ values/label=Shoe size ]
\StatsGraph \facebook [ cumulative, x/label=Time spent on Facebook ]
```



---

values/axis
x/axis
comb/values/axis
comb/x/axis
histogram/values/axis
histogram/x/axis
cumulative/values/axis
cumulative/x/axis
values/axis/reset
x/axis/reset
comb/values/axis/reset
comb/x/axis/reset
histogram/values/axis/reset
histogram/x/axis/reset
cumulative/values/axis/reset
cumulative/x/axis/reset

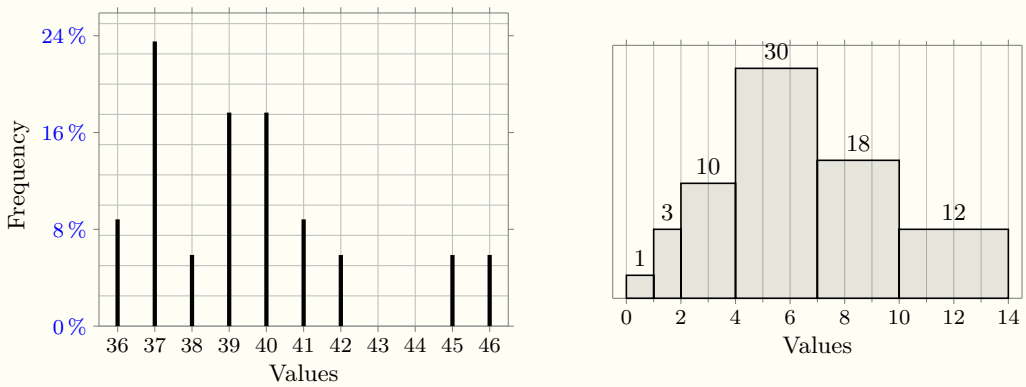
⟨*graph type*⟩/values/axis = ⟨*TikZ datavisualization axis options*⟩
⟨*graph type*⟩/x/axis = ⟨*TikZ datavisualization axis options*⟩
⟨*graph type*⟩/values/axis/reset, ⟨*graph type*⟩/x/axis/reset

The ⟨*graph type*⟩/values/axis keys append options to the TikZ x axis when the corresponding graph type is used. You can clear these options with ⟨*graph type*⟩/values/axis/reset. The keys with x are aliases for the similar keys with values. If you omit the graph type, this sets the label for all graph types simultaneously.

The initial value is:

```
values/axis = {
  label = \valuename,
  ticks and grid={many, integer minor steps}
}
```

```
\statisticssetup[graph]{
  width=0.4\columnwidth,
  comb/frequencies/axis = { ticks={step=0.08} },
  histogram/y/axis = { grid = none },
}
\StatsGraph \combdata [ frequencies, y/axis = {
  ticks={style=blue}, unit length=4cm per 0.25 units,
} ]
\hfill \StatsGraph \facebook
```



values/format
x/format
comb/values/format
comb/x/format
histogram/values/format
histogram/x/format
cumulative/values/format
cumulative/x/format

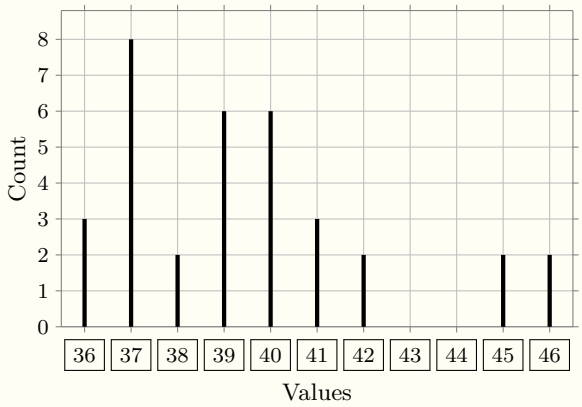
`values/format` = ⟨*formatting code*⟩, `x/format` = ⟨*formatting code*⟩
⟨*graph type*⟩`/values/format` = ⟨*formatting code*⟩
⟨*graph type*⟩`/x/format` = ⟨*formatting code*⟩

These keys set the format to use for all values that are typeset on the graphs, which currently means the values typeset alongside ticks on the x axis. The ⟨*formatting code*⟩ should be TEX code to render the actual number, in which all occurrences of `#1` are replaced by the value to typeset. The formatting code is typeset in math mode.

Keys of the form `\meta{graph type}/value/format` are used to set the formatter of values when used in a specific graph. The keys with `x` are aliases for the similar keys with `values`. If you omit the graph type, this sets the label for all graph types simultaneously.

The initial value is `values/format = \num{#1}`.

```
\StatsGraph \combdata [
      width=0.5\columnwidth,
      x/format=\fbox{$#1$}
]
```



values/margin
x/margin
comb/values/margin
comb/x/margin
histogram/values/margin
histogram/x/margin
cumulative/values/margin
cumulative/x/margin

`values/margin` = ⟨*numeric expression*⟩, `x/margin` = ⟨*numeric expression*⟩
⟨*graph type*⟩`/values/margin` = ⟨*numeric expression*⟩
⟨*graph type*⟩`/x/margin` = ⟨*numeric expression*⟩

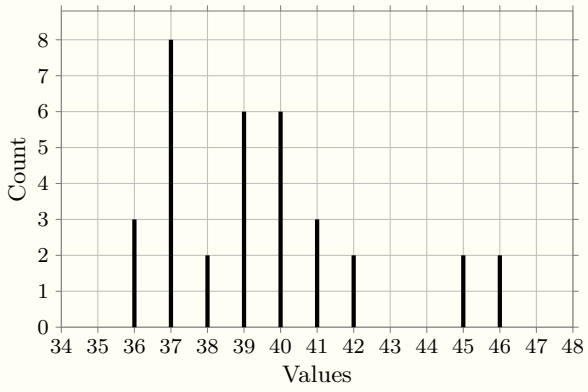
These keys set the margin that will be used for the x axis in the corresponding graph type, that is the amount of space left and right of the data that will be reserved by `\StatsGraph`. The ⟨*numeric expression*⟩, when evaluated, will correspond to the empty space reserved left of the smallest value and right of the biggest one, with the same scale as the values themselves.

In this expression, the following constants will be available: `\min` which is the minimum value in the graph; `\max` which is the maximum value; `\range` which is `\max - \min`; and `\xstep` which is the distance between two minor ticks in the graph (this is the axis step if `minor steps between steps` is empty).

The inital value is `x/margin = \xstep / 2`.

TEXhackers note: This expression will be evaluated with the rules of `\fp_eval:n` (with `\fp_gset:Nn` to be exact).

```
\StatsGraph \combdata [ width=0.5\columnwidth, x/margin=2 ]
```



1.4.7 Settings specific to cumulative graphs

`cumulative` `cumulative [ = ⟨truth value⟩ ]`

This key activates or deactivates the cumulative mode of `\StatsGraph`. The ⟨*truth value*⟩ must be either `true` or `false` or be ommited, in which case it defaults to `true`.

This mode is currently ignored if the counts are given for pointwise values, as opposed to value ranges. Support is planned but a suitable interface still needs to be devised for settings corresponding to the discontinuities.

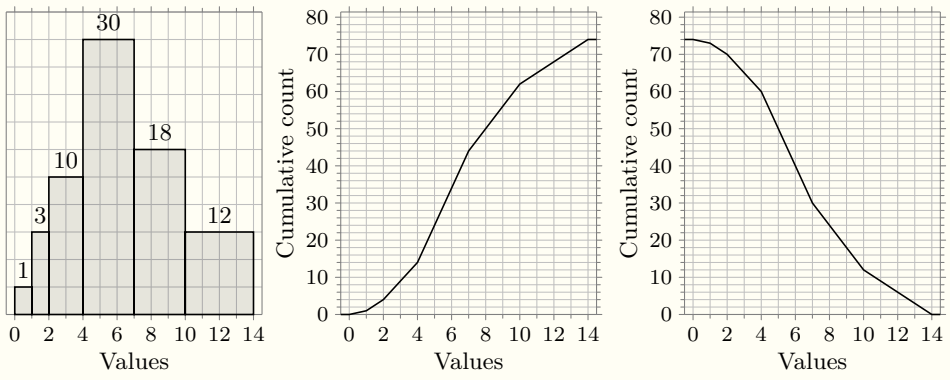
The initial value is `cumulative = false`.

`decreasing` `decreasing [ = ⟨truth value⟩ ]`

This key selects whether the cumulative mode of `\StatsGraph` plots the decreasing cumulative distribution function (that maps x to the frequency of $[x;+\infty[$) instead of the classical increasing one (mapping x to the frequency of $]-\infty;x]$). The ⟨*truth value*⟩ must be either `true` or `false` or be ommited, in which case it defaults to `true`.

The initial value is `decreasing = false`.

```
\statisticssetup[graph]{ width = 0.25\columnwidth, height=4cm }
\centering
\StatsGraph \facebook
\StatsGraph \facebook [cumulative]
\StatsGraph \facebook [cumulative, decreasing]
```



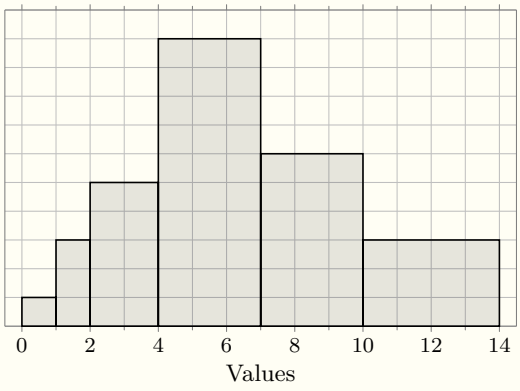
1.4.8 Settings specific to histograms

`histogram/areas` `histogram/areas [ = ⟨truth value⟩ ]`

This key activates or deactivates the typesetting of counts or frequencies above the rectangles in the histogram. They correspond to the area of the rectangle according to histogram rules, which explains the name of the key.

If ommited the ⟨*truth value*⟩ defaults to `true`, which is also the initial value.

```
\StatsGraph \facebook [width=0.5\columnwidth, histogram/areas = false]
```



`histogram/areas/style`
`histogram/areas/style/reset`

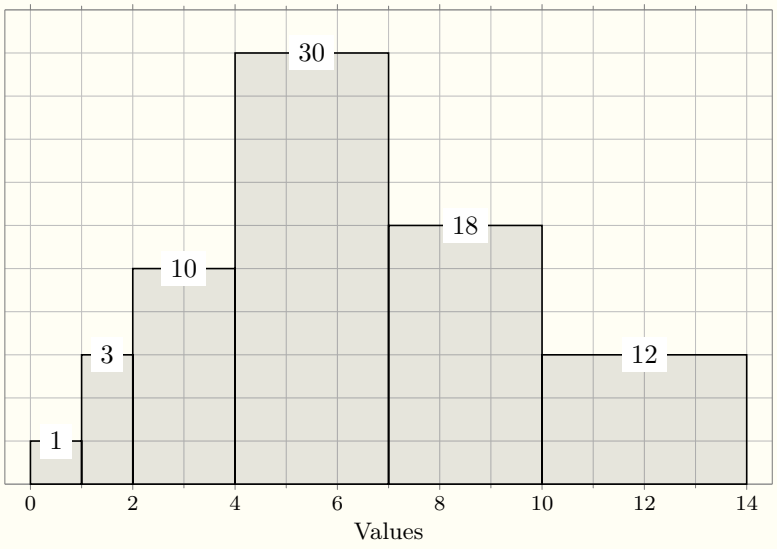
`histogram/areas/style =` ⟨*TikZ node options*⟩
`histogram/areas/style/reset`

This key appends options to the TikZ nodes containing the areas (counts or frequencies). Note that the typesetting of the areas will be controlled by the `histogram/`⟨*unit*⟩`/format` keys, which means that the `histogram/areas/style` is intended for common styling.

The initial value is `histogram/areas/style = { auto, font=\small }`.

TEXhackers note: The node is positionned in the middle of the top edge of the rectangle so if you do not want it there some style option like `auto` or `above` should be used.

```
\StatsGraph \facebook [ histogram/areas/style/reset,
                        histogram/areas/style = { fill=white } ]
```



histogram/counts/autostep
histogram/frequencies/autostep
histogram/y/autostep

`histogram/`⟨*unit*⟩`/autostep [ = `⟨*floating point expression*⟩]
`histogram/y/autostep [ = `⟨*floating point expression*⟩]

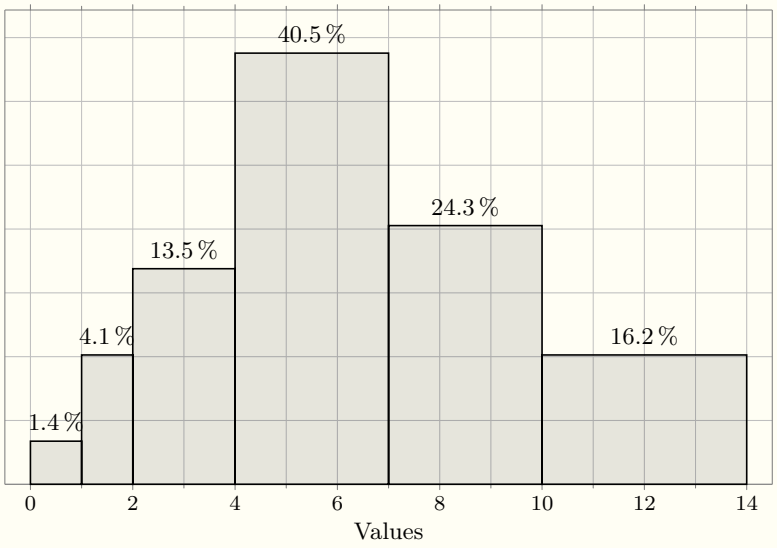
This key setups the y axis grid so that a grid tile corresponds to ⟨*floating point expression*⟩ items. This expression is interpreted as a count, but you can use the `\total` constant which is the total count. In particular, `\total/100` represents exactly 1 %.

This key essentially divides the ⟨*floating point expression*⟩ by the horizontal distance between minor steps of the values axis, then uses the result as the vertical step. As a convenience, `histogram/y/autostep` forwards its value to `histogram/legend/area` in addition to the `histogram/`⟨*unit*⟩`/autostep` keys.

If ommited the ⟨*floating point expression*⟩ defaults to 1. The initial value is `histogram/y/autostep = 1`.

TEXhackers note: `histogram/`⟨*unit*⟩`/autostep` uses `histogram/`⟨*unit*⟩`/axis` internally, so `histogram/`⟨*unit*⟩`/axis/reset` will neuter its effect.

```
\StatsGraph \facebook [frequencies, histogram/y/autostep=2*\total/100]
```



histogram/legend
histogram/legend/x
histogram/legend/w

`histogram/legend = { `⟨*legend keys*⟩ `}`
`histogram/legend/x = [ `⟨*floating point expression*⟩]
`histogram/legend/w = `⟨*floating point expression*⟩

If `histogram/legend/x` is set to an empty value, no legend will be typeset. Else, it should be a ⟨*floating point expression*⟩ which corresponds to the *value* at which the left side of the legend rectangle will lie. In that case `histogram/legend/w` should be a ⟨*floating point expression*⟩ representing the width (in value units) of the legend rectangle.

In both of these expressions, the following constants are available:

- `\min` which is the minimum value where data is present;
- `\max` which is the maximum value where data is present;
- `\range` which is `\max - \min`;
- `\xstep` which is the distance between two minor steps of the x axis.

In fact, you probably will not set these keys directly, but will use the `histogram/legend` key, which requires as value a comma-separated list of sub-keys that will be used under the `histogram/legend/` path. In particular, `histogram/legend = { x=2, y=3 }` is equivalent to `histogram/legend/x=2, histogram/legend/y=3`.

histogram/legend/y
histogram/legend/h
histogram/legend/area

`histogram/legend/y` = ⟨*floating point expression*⟩
`histogram/legend/h` = ⟨*floating point expression*⟩
`histogram/legend/area` = ⟨*floating point expression*⟩

If `histogram/legend/x` is not empty, `histogram/legend/y` and `histogram/legend/h` should be ⟨*floating point expression*⟩s which correspond to the y coordinate of the bottom side and the vertical dimension respectively of the legend rectangle, in count per value units.

In both of these expressions, the following constants are available:

- `\min` which is the y coordinate of the bottom of all histogram rectangles (this is always 0);
- `\max` which is the y coordinate of the tallest histogram rectangle;
- `\range` which is `\max - \min`;
- `\xstep` which is the distance between two minor steps of the x axis.
- `\width` which is the width of the legend rectangle as computed by evaluating `histogram/legend/w`;
- `\total` which is the total number of elements, useful when you want to size the legend using frequencies (the dimensions here always use counts).

Additionnally, when evaluating `histogram/legend/y` the `\height` constant will be available and equal to the just computed value of `histogram/legend/h`.

The key `histogram/legend/area` = ⟨*fp expression*⟩ is a shorthand for: `histogram/legend/h = (`⟨*fp expression*⟩`) / \width`.

Again, you probably will not set these keys directly but using the `histogram/legend` key.

histogram/legend/options
histogram/legend/options/reset
histogram/legend/label

`histogram/legend/options` = ⟨*TikZ node options*⟩
`histogram/legend/options/reset`
`histogram/legend/label` = ⟨*TikZ label value*⟩

The key `histogram/legend/options` appends the ⟨*TikZ node options*⟩ to the list of options that will be passed to the TikZ node responsible for the legend rectangle, *after* the options in `histogram/style`. You can use it to tweak the apparance of the legend.

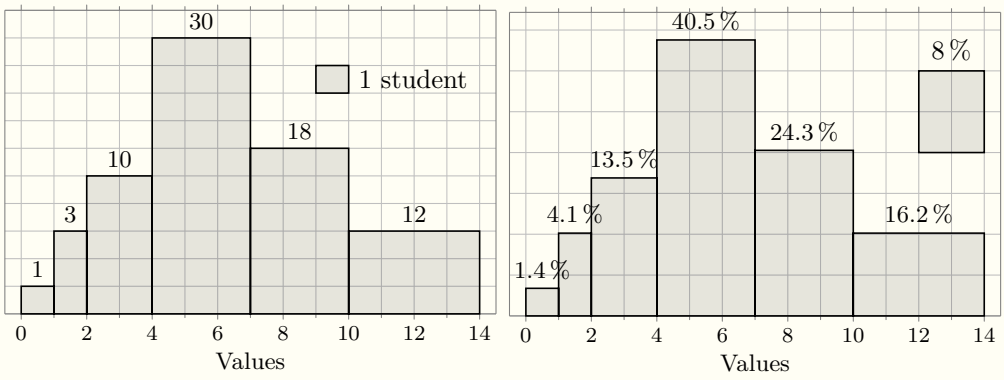
The key `histogram/legend/label` = ⟨*label*⟩ is a shorthand for: `histogram/legend/options = { label = {`⟨*fp expression*⟩`} }`, and thus uses the TikZ label syntax.

Again, you probably will not set these keys directly but using the `histogram/legend` key.

The initial value is `histogram/legend = { x=, y=0, w=\xstep, area=1 }` which means that no legend is typeset, and the legend options are empty.

TEXhackers note: `area=1` is actually set by the initial value of `histogram/y/autostep`.

```
\statisticssetup[graph]{ width = 0.48\columnwidth }
\StatsGraph \facebook [
  histogram/legend = { x=9, y=8, label=1 student }
]
\StatsGraph \facebook [
  frequencies, histogram/y/autostep=0.02*\total,
  histogram/legend = { x=12, y=2*\height, w=2, area=0.08*\total,
                       label=above:\SI{8}{\percent} }
]
```



2 statistics implementation

```
⟨*package⟩
⟨@@=statistics⟩

\ProvidesExplPackage
  {\ExplFileName}{\ExplFileDate}{\ExplFileVersion}{\ExplFileDescription}

\RequirePackage{xparse}
\RequirePackage{siunitx}
\RequirePackage{tikz}
\RequirePackage{etoolbox}

\ExplSyntaxOff
\usetikzlibrary{datavisualization, fit}
\ExplSyntaxOn
```

Translations

```
\tl_new:N \valuename
\tl_new:N \countname
\tl_new:N \freqname
\tl_new:N \ccountname
\tl_new:N \cfreqname
\tl_new:N \iccname
\tl_new:N \icfname
\tl_new:N \dccname
\tl_new:N \dcfname

\tl_set:Nn \valuename   { Values }
\tl_set:Nn \countname   { Count }
\tl_set:Nn \ccountname  { Cumulative~count }
\tl_set:Nn \freqname    { Frequency }
\tl_set:Nn \cfreqname   { Cumulative~frequency }
\tl_set:Nn \iccname     { ICC }
\tl_set:Nn \icfname     { ICF }
\tl_set:Nn \dccname     { DCC }
\tl_set:Nn \dcfname     { DCF }

\AtEndPreamble {
    \tl_if_exist:NT \captionsfrench {
        \tl_put_right:Nn \captionsfrench {
            \tl_set:Nn \valuename   { Modalit\'e }
            \tl_set:Nn \countname   { Effectif }
            \tl_set:Nn \ccountname  { Effectif~cumul\'e }
            \tl_set:Nn \freqname    { Fr\'equence }
            \tl_set:Nn \cfreqname   { Fr\'equence~cumul\'ee }
            \tl_set:Nn \iccname     { ECC }
            \tl_set:Nn \icfname     { FCC }
            \tl_set:Nn \dccname     { ECD }
            \tl_set:Nn \dcfname     { FCD }
        }
    }
}
```

2.1 Common facilities

```
\cs_new_protected:Nn \__statistics_keys_define:nn {
    \keys_define:nn { statistics / #1 } { #2 }
}
\cs_new_protected:Nn \__statistics_setup:nn {
    \keys_set:nn { statistics / #1 } { #2 }
}
\NewDocumentCommand \statisticssetup { o +m } {
    \IfNoValueTF { #1 } {
        \keys_set:nn { statistics } { #2 }
    }{
```

```
        \keys_set:nn { statistics / #1 } { #2 }
    }
}

\tl_new:N \l__statistics_data_tl
\seq_new:N \l__statistics_show_seq

\int_new:N \l__statistics_nbvals_int
\int_new:N \l__statistics_currange_int

\fp_new:N \l__statistics_total_fp
\fp_new:N \l__statistics_curtotal_fp

\fp_new:N \l__statistics_range_min_fp
\fp_new:N \l__statistics_range_max_fp
\tl_new:N \l__statistics_range_minrel_tl
\tl_new:N \l__statistics_range_maxrel_tl
\cs_new_protected_nopar:Npn
            \__statistics_parse_range:w \IN#1#2;#3;#4#5\q_stop {
```

- `#1` is the first `[` or `]`
- `#4` is the second `[` or `]` and `#5` eats all trailing tokens

```
    \fp_set:Nn \l__statistics_range_min_fp { #2 }
    \fp_set:Nn \l__statistics_range_max_fp { #3 }
}
\cs_new_protected_nopar:Npn
            \__statistics_parse_range_full:w \IN#1#2;#3;#4#5\q_stop {
    \fp_set:Nn \l__statistics_range_min_fp { #2 }
    \fp_set:Nn \l__statistics_range_max_fp { #3 }
    \tl_if_eq:nnTF { #1 } { [ } {
        \tl_set:Nn \l__statistics_range_minrel_tl { <=}
    }{
        \tl_set:Nn \l__statistics_range_minrel_tl { < }
    }
    \tl_if_eq:nnTF { #4 } { ] } {
        \tl_set:Nn \l__statistics_range_maxrel_tl { <= }
    }{
        \tl_set:Nn \l__statistics_range_maxrel_tl { < }
    }
    \exp_args:NNnx
    \prg_set_conditional:Nnn \__statistics_if_in_range:n { T } {
        \exp_not:N \fp_compare:nTF {
                \exp_not:N \l__statistics_range_min_fp
                \exp_not:V \l__statistics_range_minrel_tl
                \exp_not:n { ##1 }
                \exp_not:V \l__statistics_range_maxrel_tl
                \exp_not:N \l__statistics_range_max_fp }{
            \exp_not:N \prg_return_true:
        }{
            \exp_not:N \prg_return_false:
        }
    }
}
```

2.2 Compute and typeset statistics tables

```
\NewDocumentCommand \__statistics_IN:w { m u{;} u{;} m } {
    \ensuremath{ \left#1 \num{#2} \mathbin{;} \num{#3} \right#4 }
}

\cs_new_protected:Nn \__statistics_setshow:n {
    \seq_clear:N \l__statistics_show_seq
```

```
    \clist_map_inline:nn {#1} {
        \tl_if_in:nnTF {##1} {-} {
            \__statistics_setshow_aux:w ##1 \q_stop
        }{
            \seq_put_right:Nn \l__statistics_show_seq {##1}
        }
    }
}
\cs_new_protected:Npn \__statistics_setshow_aux:w #1 - #2 \q_stop {
    \int_step_inline:nnnn {#1} {1} {#2} {
        \seq_put_right:Nn \l__statistics_show_seq {##1}
    }
}
\cs_new_protected_nopar:Nn \__statistics_set_if_shown:N {
    \seq_if_empty:NTF \l__statistics_show_seq {
        \bool_set_true:N #1
    }{
        \seq_if_in:NVTF
                \l__statistics_show_seq
                \l__statistics_currange_int {
            \bool_set_true:N #1
        }{
            \bool_set_false:N #1
        }
    }
}

\int_new:N \l__statistics_table_maxcols_int
\int_set:Nn \l__statistics_table_maxcols_int {0}

\__statistics_keys_define:nn { table } {
    showonly    .value_required:n = true,
    showonly    .code:n = \__statistics_setshow:n{#1},

    showonly/hidden .value_required:n = true,
    showonly/hidden .code:n = {
                    \cs_set_protected:Nn
                            \__statistics_table_hidden_format:n
                            { #1 }
                },
    showonly/hidden .initial:n = ,

    showonly/shown  .value_required:n = true,
    showonly/shown  .code:n = {
                    \cs_set_protected:Nn
                            \__statistics_table_shown_format:n
                            { #1 }
                },
    showonly/shown  .initial:n = #1,

    maxcols     .clist_set:N  = \l__statistics_table_maxcols_clist,
    maxcols     .value_required:n = true,
    maxcols     .initial:n  = ,

    tablesep    .tl_set:N   = \l__statistics_table_sep_tl,
    tablesep    .value_required:n = true,
    tablesep    .initial:n  = \\,

    valign      .tl_set:N   = \l__statistics_table_valign_tl,
    valign      .value_required:n = true,
    valign      .initial:n  = t,

    coltype     .tl_set:N   = \l__statistics_table_coltype_tl,
```

```
    coltype     .value_required:n = true,

    headcoltype .tl_set:N   = \l__statistics_table_headcoltype_tl,
    headcoltype .value_required:n = true,

    newline     .tl_set:N   = \l__statistics_table_newline_tl,
    newline     .value_required:n = true,

    preline     .tl_set:N   = \l__statistics_table_preline_tl,
    preline     .value_required:n = true,

    postline    .tl_set:N   = \l__statistics_table_postline_tl,
    postline    .value_required:n = true,

    outline     .meta:n     = { preline={#1}, postline={#1} },
    outline     .value_required:n = true,

    frame       .choice:,
    frame/full  .meta:n     = { preline=\firsthline, postline=\lasthline,
                                newline=\\\hline,
                                headcoltype=|l|, coltype=c| },
    frame/full  .value_forbidden:n = true,

    frame/none  .meta:n     = { outline=, newline=\\,
                                headcoltype=l, coltype=c },
    frame/none  .value_forbidden:n = true,

    frame/clean .meta:n     = { preline=\firsthline, postline=\lasthline,
                                newline=\\,
                                headcoltype=l, coltype=c },
    frame/clean .initial:n  = ,
    frame/clean .value_forbidden:n = true,

    digits      .int_set:N  = \l__statistics_table_round_int,
    digits      .initial:n  = 3,

    allcounts/format    .code:n = {
                    \cs_set_protected:Nn
                            \__statistics_table_allcounts_format:n
                            { #1 }
                },
    allcounts/format    .value_required:n = true,
    allcounts/format    .initial:n = { \num{#1} },

    allfreqs/format     .code:n = {
                    \cs_set_protected:Nn
                            \__statistics_table_allfreqs_format:n
                            { #1 }
                },
    allfreqs/format     .value_required:n = true,

    allfreqs/format/real .meta:n = {
                    allfreqs/format = \num{##1}
                },
    allfreqs/format/real .value_forbidden:n = true,

    allfreqs/format/percent .meta:n = {
                    allfreqs/format = \SI{\fp_eval:n{##1*100}}{\percent}
                },
    allfreqs/format/percent .initial:n = ,
    allfreqs/format/percent .value_forbidden:n = true,

    allfreqs/format/scaled .meta:n = {
```

```
                    allfreqs/format = \num{\fp_eval:n{##1*#1}}
                },
    allfreqs/format/scaled .value_required:n = true,
}

\cs_new:Nn \__statistics_define_row:nnn {
```

- #1 (tl): row name;
- #2 (bool): enabled by default
- #3 (tl): default header;

```
    \tl_new:c { l__statistics_table_#1_name_tl }
    \bool_new:c { l__statistics_table_#1_bool }
    \__statistics_keys_define:nn { table } {
        #1          .code:n = {
                        \bool_set_true:c { l__statistics_table_#1_bool }
                        \quark_if_no_value:nF { ##1 } {
                            \__statistics_setup:nn { table } {
                                #1/header = { ##1 }
                            }
                        }
                    },
        #1          .default:n = \q_no_value,

        no#1        .code:n =
                        \bool_set_false:c { l__statistics_table_#1_bool },
        no#1        .value_forbidden:n = true,

        #1/header   .tl_set:c = { l__statistics_table_#1_name_tl },
        #1/header   .value_required:n = true,
        #1/header   .initial:n = { #3 },

        #1/format   .code:n = {
                        \cs_set_protected:cn
                                { __statistics_table_#1_format:n }
                                { ##1 }
                    },
        #1/format   .value_required:n = true,
        #1/format   .initial:n = { ##1 },
    }
    \bool_set:cn { l__statistics_table_#1_bool } { #2 }
}

\__statistics_define_row:nnn { values } \c_true_bool \valuename
\__statistics_define_row:nnn { counts } \c_false_bool \countname
\__statistics_define_row:nnn { frequencies }  \c_false_bool \freqname
\__statistics_define_row:nnn { icc }   \c_false_bool \iccname
\__statistics_define_row:nnn { icf }   \c_false_bool \icfname
\__statistics_define_row:nnn { dcc }   \c_false_bool \dccname
\__statistics_define_row:nnn { dcf }   \c_false_bool \dcfname

\__statistics_setup:nn { table } {
    values/format = \ensuremath{#1},
}

\cs_undefine:N \__statistics_define_row:nnn

\seq_new:N \l__statistics_table_contents_seq
\tl_new:N \l__statistics_table_preamble_tl

\tl_new:N \l__statistics_table_values_tl
\tl_new:N \l__statistics_table_counts_tl
\tl_new:N \l__statistics_table_frequencies_tl
```

```
\tl_new:N \l__statistics_table_icc_tl
\tl_new:N \l__statistics_table_icf_tl
\tl_new:N \l__statistics_table_dcc_tl
\tl_new:N \l__statistics_table_dcf_tl

\fp_new:N \l__statistics_table_curICF_fp
\fp_new:N \l__statistics_table_prevICF_fp

\bool_new:N \l__statistics_table_firstrow_bool

\seq_new:N \l__statistics_store_values_seq
\seq_new:N \l__statistics_store_counts_seq

\cs_generate_variant:Nn \keyval_parse:NNn { NNV }
\NewDocumentCommand \StatsTable { +O{} +m +O{} } {
    \group_begin:
```

Ensure some macros exist with sensible definitions

```
    \cs_if_exist:NF \firsthline {
        \cs_set_eq:NN \firsthline \hline
    }
    \cs_if_exist:NF \lasthline {
        \cs_set_eq:NN \lasthline \hline
    }
    \cs_if_exist:NF \IN {
        \cs_set_eq:NN \IN \__statistics_IN:w
    }
```

Handle optional settings

```
    \__statistics_setup:nn { table } { #1, #3 }
```

Get the data inline or from a variable

```
    \tl_if_single:nTF { #2 } {
```

Generate meaningful error by using the non-existent variable

```
        \cs_if_exist:NF #2 { #2 }
        \tl_set_eq:NN \l__statistics_data_tl #2
    }{
        \tl_set:Nn \l__statistics_data_tl { #2 }
    }
```

Define getters for some items of the table, to be used for instance to programmatically choose the formatting.

```
    \cs_set_nopar:Npn \getvalue {
        \seq_item:Nn \l__statistics_store_values_seq
    }
    \cs_set_nopar:Npn \getcount {
        \seq_item:Nn \l__statistics_store_count_seq
    }
```

Compute the total population count/frequency

```
    \fp_zero:N \l__statistics_total_fp
    \keyval_parse:NNV
            \__statistics_table_count:n
            \__statistics_table_count:nn
            \l__statistics_data_tl
```

Loop again and output the table

```
    \__statistics_table_start:
    \fp_zero:N \l__statistics_table_prevICF_fp
    \keyval_parse:NNV
            \__statistics_table_make:n
            \__statistics_table_make:nn
            \l__statistics_data_tl
    \__statistics_table_end:
```

Done

```
    \group_end:
}
```

table building functions

```
\cs_new_protected_nopar:Nn \__statistics_table_start: {
```

Init column count and fetch the next maxcols value (or keep the current one if we reached the end of the list).

```
    \int_zero:N \l__statistics_nbvals_int
    \clist_pop:NNT \l__statistics_table_maxcols_clist \l_tmpa_tl {
        \int_set:Nn \l__statistics_table_maxcols_int { \l_tmpa_tl }
    }
```

Start rows with headers

```
    \clist_map_inline:nn { values, counts, frequencies, icc, icf, dcc, dcf } {
        \tl_set:cx { l__statistics_table_##1_tl } {
            \exp_not:N \ensuremath { \exp_not:N \hbox {
                \exp_not:c { l__statistics_table_##1_name_tl }
            } }
        }
    }
}
\cs_new_protected_nopar:Nn \__statistics_table_end: {
```

Build-up the table preamble

```
    \tl_set:Nx \l__statistics_table_preamble_tl {
        \exp_not:n { \begin{array}[ }
            \exp_not:V \l__statistics_table_valign_tl
        \exp_not:n { ] }
                { \exp_not:V \l__statistics_table_headcoltype_tl
                  \prg_replicate:nn { \l__statistics_nbvals_int }
                    { \exp_not:V \l__statistics_table_coltype_tl } }
    }
```

Add each row if needed.

```
    \seq_clear:N \l__statistics_table_contents_seq
    \clist_map_inline:nn { values, counts, icc, dcc, frequencies, icf, dcf } {
        \bool_if:cT { l__statistics_table_##1_bool } {
            \seq_put_right:Nv
                    \l__statistics_table_contents_seq
                    { l__statistics_table_##1_tl }
        }
    }
    $\tl_use:N \l__statistics_table_preamble_tl
        \l__statistics_table_preline_tl
        \seq_use:Nn
                \l__statistics_table_contents_seq
                { \l__statistics_table_newline_tl }
        \\ \l__statistics_table_postline_tl
    \end{array}$
}
```

Counting auxiliaries

```
\cs_new_protected_nopar:Nn \__statistics_table_count:n {
    \__statistics_table_count:nn {} { 1 }
}
\cs_new_protected_nopar:Nn \__statistics_table_count:nn {
    \fp_add:Nn \l__statistics_total_fp { #2 }
}
```

Accumulating content

```
\cs_new_protected_nopar:Nn \__statistics_table_make:n {
    \__statistics_table_make:nn { #1 } { 1 }
}
\cs_new_protected_nopar:Nn \__statistics_table_make:nn {
```

Maybe close the table and create a new one

```
    \int_compare:nT
            { 0 < \l__statistics_table_maxcols_int
                = \l__statistics_nbvals_int } {
        \__statistics_table_end:
        \tl_use:N \l__statistics_table_sep_tl
        \__statistics_table_start:
    }
    \int_incr:N \l__statistics_nbvals_int
    \int_incr:N \l__statistics_currange_int
    \fp_add:Nn \l__statistics_curtotal_fp { #2 }
```

Hidden or not

```
    \__statistics_set_if_shown:N \l_tmpa_bool
    \tl_set:Nx \l_tmpa_tl {
        \exp_not:n { & \tl_set:Nn \currentcolumn } {
            \int_use:N \l__statistics_currange_int
        }
    }
    \bool_if:NTF \l_tmpa_bool {
        \tl_put_right:Nn \l_tmpa_tl
            {\__statistics_table_shown_format:n}
    }{
        \tl_put_right:Nn \l_tmpa_tl
            {\__statistics_table_hidden_format:n}
    }
```

Values

```
    \seq_put_right:Nn \l__statistics_store_values_seq { #1 }
    \bool_if:NT \l__statistics_table_values_bool {
        \tl_put_right:Nx \l__statistics_table_values_tl {
            \exp_not:V \l_tmpa_tl {
                \exp_not:n {
                    \__statistics_table_values_format:n { #1 }
                }
            }
        }
    }
```

Counts

```
    \seq_put_right:Nx \l__statistics_store_counts_seq { \fp_eval:n {#2} }
    \bool_if:NT \l__statistics_table_counts_bool {
        \tl_put_right:Nx \l__statistics_table_counts_tl {
            \exp_not:V \l_tmpa_tl {
                \exp_not:n {
                    \__statistics_table_counts_format:n {
                        { \__statistics_table_allcounts_format:n { #2 } }
                    }
                }
            }
        }
    }
```

ICC

```
    \bool_if:NT \l__statistics_table_icc_bool {
        \tl_put_right:Nx \l__statistics_table_icc_tl {
            \exp_not:V \l_tmpa_tl {
```

```
                \exp_not:n { \__statistics_table_icc_format:n }
                {
                    \exp_not:n{ \__statistics_table_allcounts_format:n }
                        { \fp_use:N \l__statistics_curtotal_fp }
                }
            }
        }
    }
```

DCC (= 1 - ICC + curcount)

```
    \bool_if:NT \l__statistics_table_dcc_bool {
        \tl_put_right:Nx \l__statistics_table_dcc_tl {
            \exp_not:V \l_tmpa_tl {
                \exp_not:n { \__statistics_table_dcc_format:n }
                {
                    \exp_not:n{ \__statistics_table_allcounts_format:n }
                    {
                        \fp_eval:n {
                            \l__statistics_total_fp
                                - \l__statistics_curtotal_fp
                                + #2
                        }
                    }
                }
            }
        }
    }
```

Frequencies (we compute them from the ICFs so that rounded freqs add up to 1)

```
    \fp_set:Nn \l__statistics_table_curICF_fp {
        round(\l__statistics_curtotal_fp
                / \l__statistics_total_fp,
              \l__statistics_table_round_int)
    }
    \bool_if:NT \l__statistics_table_frequencies_bool {
        \tl_put_right:Nx \l__statistics_table_frequencies_tl {
            \exp_not:V \l_tmpa_tl {
                \exp_not:n { \__statistics_table_frequencies_format:n }
                {
                    \exp_not:n{ \__statistics_table_allfreqs_format:n }
                    {
                        \fp_eval:n {
                            \l__statistics_table_curICF_fp
                                - \l__statistics_table_prevICF_fp
                        }
                    }
                }
            }
        }
    }
```

ICF

```
    \bool_if:NT \l__statistics_table_icf_bool {
        \tl_put_right:Nx \l__statistics_table_icf_tl {
            \exp_not:V \l_tmpa_tl {
                \exp_not:n { \__statistics_table_icf_format:n }
                {
                    \exp_not:n{ \__statistics_table_allfreqs_format:n }
                        { \fp_to_decimal:N \l__statistics_table_curICF_fp }
                }
            }
        }
    }
```

DCF (= 1 - ICF + curfreq = 1 - prevICF)

```
    \bool_if:NT \l__statistics_table_dcf_bool {
        \tl_put_right:Nx \l__statistics_table_dcf_tl {
            \exp_not:V \l_tmpa_tl {
                \exp_not:n { \__statistics_table_dcf_format:n }
                {
                    \exp_not:n{ \__statistics_table_allfreqs_format:n }
                    {
                        \fp_eval:n {
                            1 - \l__statistics_table_prevICF_fp
                        }
                    }
                }
            }
        }
    }
```

Prepare for next iteration

```
    \fp_set_eq:NN
        \l__statistics_table_prevICF_fp
        \l__statistics_table_curICF_fp
}
```

2.3 Compute and typeset statistics graphics

```
\cs_new_protected:Nn \__statistics_make_forwarded_key:nnnn {
```

- #1 (tl): common prefix
- #2 (tl): middle
- #3 (clist): replacements
- #4 (tl): common suffix

```
    \tl_clear:N \l_tmpa_tl
    \clist_map_inline:nn {#3} {
        \tl_put_right:Nx \l_tmpa_tl {
            \exp_not:n {#1}
            \tl_if_empty:nF {#1} { \tl_if_empty:nF {##1} {\exp_not:N /} }
            \exp_not:n {##1}
            \tl_if_empty:nF {#4} { \tl_if_empty:nF {##1} {\exp_not:N /} }
            \exp_not:n {#4,}
        }
    }
    \tl_set:Nx \l_tmpb_tl {
        \exp_not:n {#1}
        \tl_if_empty:nF {#1} { \tl_if_empty:nF {#2} {\exp_not:N /} }
        \exp_not:n {#2}
        \tl_if_empty:nF {#4} { \tl_if_empty:nF {#2} {\exp_not:N /} }
        \exp_not:n {#4}
    }
    \use:x {
        \exp_not:n { \__statistics_keys_define:nn { graph } }
        {
            \exp_not:V \l_tmpb_tl \exp_not:n { .default:n = \q_no_value, }
            \exp_not:V \l_tmpb_tl
                \exp_not:n { .code:n = \__statistics_forwarded_key:nn }
                    { \exp_not:V \l_tmpa_tl }
                    { \exp_not:n { ##1 } }
        }
    }
}
\cs_new_protected:Nn \__statistics_forwarded_key:nn {
```

```
    \quark_if_no_value:nTF { #2 } {
        \__statistics_setup:nn { graph } { #1 }
    }{
        \clist_set:Nn \l_tmpa_clist { #1,{} }
        \use:x {
            \exp_not:n { \__statistics_setup:nn { graph } } {
                \clist_use:Nn \l_tmpa_clist { = {#2}, }
            }
        }
    }
}
\cs_new_protected_nopar:Nn \__statistics_forward_keys:nn {
```

- #1 (clist): destination prefixes
- #2 (clist): keys

```
    \clist_map_inline:nn {#2} {
        \__statistics_make_forwarded_key:nnnn {} {} { #1 } { ##1 }
    }
}

\cs_new:Nn \__statistics_create_append_reset:nn {
```

- #1 (tl): key basename
- #2 (var): suffix of variable to store options into

```
    \tl_new:c { l__statistics_graph_#2_tl }
    \__statistics_keys_define:nn { graph } {
        #1          .value_required:n = true,
        #1          .code:n = \tl_put_right:cn
                              { l__statistics_graph_#2_tl }
                              { ##1, },

        #1/reset    .value_forbidden:n = true,
        #1/reset    .code:n = \tl_clear:c
                              { l__statistics_graph_#2_tl },
    }
}

\cs_new:Nn \__statistics_DO:nn { \__statistics_create_append_reset:nn {#1}{options_#2}

\cs_new:Nn \__statistics_define_unit:nn {
```

- #1 (tl): unit name (plural)
- #2 (tl): graph type

```
    \__statistics_DO:nn { #2/#1/axis }       { #2_#1axis }
    \__statistics_keys_define:nn { graph } {
        #2/#1       .code:n = {
                        \tl_set:cn {l__statistics_graph_#2_unit_tl} { #1 }
                        \quark_if_no_value:nF { ##1 } {
                            \__statistics_setup:nn { graph }{ #2/#1/label = { ##1 } }
                        }
                    },
        #2/#1       .default:n = \q_no_value,

        #2/#1/label .meta:n = { #2/#1/axis = { label = { ##1 } } },
        #2/#1/label .value_required:n = true,

        #2/#1/format .code:n = {
                        \cs_set_protected:cn
                              { __statistics_graph_#2_#1_format:n }
```

```
                                { ##1 }
                    },
        #2/#1/format .value_required:n = true,

        #2/#1/margin .tl_set:c = l__statistics_graph_#2_#1_vmargin_tl,
        #2/#1/margin .value_required:n = true,
    }
}

\__statistics_DO:nn { picture }                  { pic }
\__statistics_DO:nn { axissystem }               { system }

\__statistics_DO:nn { histogram/areas/style }    { areas }
\__statistics_DO:nn { histogram/legend/options } { legend }

\clist_map_inline:nn { histogram, cumulative, comb } {
    \__statistics_define_unit:nn                 { counts } { #1 }
    \__statistics_define_unit:nn                 { frequencies } { #1 }
    \__statistics_DO:nn { #1/style }             { #1 }
    \__statistics_DO:nn { #1/values/axis }       { #1_xaxis }
    \__statistics_keys_define:nn { graph/#1 } {
        values/margin .value_required:n = true,
        values/margin .tl_set:c = l__statistics_graph_#1_hmargin_tl,

        values/label .meta:n = { values/axis = { label = { ##1 } } },
        values/label .value_required:n = true,

        values/format .code:n = { \cs_set_protected:cn
                                    {__statistics_graph_#1_values_format:n} { ##1 }
        },
        values/format .value_required:n = true,

        frequencies/format/real .meta:n = {
            frequencies/format = {
                \num[round-mode=places,round-precision=##1]{####1}
            }
        },
        frequencies/format/real .default:n = 1,

        frequencies/format/percent .meta:n = {
            frequencies/format = {
                \SI[round-mode=places,round-precision=##1]{
                    \fp_eval:n{####1*100}
                }{\percent}
            }
        },
        frequencies/format/percent .default:n = 1,
    }
    \__statistics_make_forwarded_key:nnnn {#1/values}{}{label}{}
    \clist_map_inline:nn { axis, axis/reset, label, margin, format } {
        \__statistics_make_forwarded_key:nnnn {#1}{x}{values}{##1}
        \__statistics_make_forwarded_key:nnnn {#1}{y}{counts, frequencies}{##1}
    }
}

\cs_undefine:N \__statistics_DO:nn
\cs_undefine:N \__statistics_define_unit:nnn

\__statistics_forward_keys:nn { histogram, cumulative, comb } {
    values, values/label, values/margin, values/format,
    values/axis, values/axis/reset,
    x/label, x/axis, x/axis/reset, x/margin, x/format,
    counts, counts/label, counts/margin, counts/format,
```

```
    counts/axis, counts/axis/reset,
    frequencies, frequencies/label, frequencies/margin,
    frequencies/format, frequencies/format/real, frequencies/format/percent,
    frequencies/axis, frequencies/axis/reset,
    y/label, y/axis, y/axis/reset, y/margin, y/format,
    style, style/reset
}

\__statistics_create_append_reset:nn { tikzinfo' } { userpreinfo }
\__statistics_create_append_reset:nn { tikzinfo }  { userpostinfo }

\cs_undefine:N \__statistics_forward_keys:nn
\cs_undefine:N \__statistics_make_forwarded_key:nnnn
\cs_undefine:N \__statistics_create_append_reset:nn

\__statistics_keys_define:nn { graph } {
    showonly    .value_required:n = true,
    showonly    .code:n = \__statistics_setshow:n{#1},

    height      .value_required:n = true,
    height      .meta:n = { axissystem = { height = { #1 } } },

    width       .value_required:n = true,
    width       .meta:n = { axissystem = { width = { #1 } } },

    cumulative  .bool_set:N = \l__statistics_graph_cumulative_bool,
    cumulative  .default:n = true,

    decreasing  .bool_set:N = \l__statistics_graph_decreasing_bool,
    decreasing  .default:n = true,

    histogram/areas .bool_set:N = \l__statistics_graph_areas_bool,
    histogram/areas .default:n = true,

    histogram/legend/label .value_required:n = true,
    histogram/legend/label .meta:n = {
                histogram/legend/options = {label={#1}} },

    histogram/legend/area .value_required:n = true,
    histogram/legend/area .meta:n = {
                histogram/legend/h = (#1)/\width },

    histogram/legend    .value_required:n = true,
    histogram/legend    .code:n = {
                \__statistics_setup:nn { graph / histogram/legend  } {
                    #1
                }
            },

    histogram/y/autostep .value_required:n = true,
    histogram/y/autostep .meta:n = {
                histogram/counts/autostep = {#1},
                histogram/frequencies/autostep = {#1},
                histogram/legend/area = {#1},
            },
}
\tl_map_inline:nn {xywh} {
    \__statistics_keys_define:nn { graph / histogram / legend } {
        #1 .value_required:n = true,
        #1 .tl_set:c = {l__statistics_graph_legend_#1_tl},
    }
}
\clist_map_inline:nn { counts, frequencies } {
```

```
    \__statistics_keys_define:nn { graph/histogram/#1 } {
        autostep .default:n = 1,
        autostep .meta:n = { axis = {
                grid = { compute~step =
                    \group_begin:
                    \tl_set:Nx \total { \fp_to_decimal:N \l__statistics_total_fp }
                    \fp_gset:Nn \g_tmpa_fp { ##1 }
                    \group_end:
                    \tl_set:Nx \tikz@lib@dv@step {
                        \fp_eval:n {\g_tmpa_fp / \g__statistics_graph_xstep_fp }
                    }
                }
        }},
    }
}

\__statistics_setup:nn { graph }{
    width = 0.75\columnwidth,
    cumulative = false,
    decreasing = false,

    values/axis = {
        label = \valuename,
        ticks~and~grid={many, integer~minor~steps}
    },
    values/margin = \xstep / 2,
    values/format = \num{#1},

    y/margin = \range/10,

    counts/format = { \num{#1} },
    counts/axis = { ticks~and~grid={
        many, int~about~strategy, integer~minor~steps*,
        } },
    comb/counts/label = \countname,
    cumulative/counts/label = \ccountname,

    frequencies/format/percent,
    frequencies/axis = { ticks~and~grid=many },
    comb/frequencies/label = \freqname,
    cumulative/frequencies/label = \cfreqname,

    histogram/y/axis/reset,
    histogram/y/axis = {ticks = none},
    histogram/y/autostep = 1,
    histogram/legend = { x=, y=0, w=\xstep },
    histogram/style  = {
        every~path/.prefix~style=fill,
        semithick, black, fill=black, fill~opacity=0.1
    },
    histogram/areas,
    histogram/areas/style = { auto, font=\small },

    comb/style  = { ultra~thick },

    counts,

    picture = {
        baseline = (current~bounding~box.center),
        label~position = right,
    },
}
```

```
\tl_const:Nn \c__statistics_graph_savexstep_tl {
    grid = { compute~step/.append = {
        \cs_if_eq:NNF \tikz@lib@dv@step \relax {
            \pgfkeysgetvalue
                {/tikz/data~visualization/minor~steps~between~steps}
                \l_tmpa_tl
            \fp_gset:Nn \g__statistics_graph_xstep_fp {
                \tikz@lib@dv@step
                / (\fp_max:nn{0\l_tmpa_tl + 1}{1})
            }
        }
    }}
}
```

To detect that the user didn't set `minor steps between steps` himself after having used `integer minor steps` (which can be a default setting), we add a handler to the key that sets its value but also empties `\l__statistics_graph_maxminor_tl` so that we do not overwrite anything.

```
\tl_new:N \l__statistics_graph_maxminor_tl
\int_new:N \l__statistics_graph_minorsteps_int
\fp_new:N \l__statistics_graph_ims_step_fp
\fp_new:N \l__statistics_graph_ims_range_fp
\fp_new:N \l__statistics_graph_ims_threshold_fp
\tikzdatavisualizationset{
integer~minor~steps/.style={
    /utils/exec = \tl_set:Nn \l__statistics_graph_maxminor_tl {#1},
    minor~steps~between~steps/.code=
            \tl_clear:N \l__statistics_graph_maxminor_tl
            \pgfkeyssetvalue
                {/tikz/data~visualization/minor~steps~between~steps}
                {##1} ,
    compute~step/.append = {
        \tl_set_eq:NN \l_tmpa_tl \tikz@lib@dv@step
        \tl_if_empty:NT \l__statistics_graph_maxminor_tl {
            \tl_set_eq:NN \l_tmpa_tl \relax
        }
        \tl_if_eq:NNF \l_tmpa_tl \relax {
            \fp_set:Nn \l__statistics_graph_ims_step_fp { \l_tmpa_tl }
            \tikz@lib@dv@mapper.get~in~range~interval()
            \pgfdvinrangeinterval.get~min~and~max()
            \pgfdvmathexitbyscientificformat \l_tmpa_tl \pgfdvmin
            \pgfdvmathexitbyscientificformat \l_tmpb_tl \pgfdvmax
            \fp_set:Nn \l__statistics_graph_ims_range_fp { \l_tmpb_tl - \l_tmpa_tl }
            \fp_set:Nn \l__statistics_graph_ims_threshold_fp {
                \fp_max:nn {
                    \l__statistics_graph_ims_step_fp * (\l__statistics_graph_maxminor_t
                }{
                    \l__statistics_graph_ims_range_fp
                }
            }
            \int_set:Nn \l__statistics_graph_minorsteps_int
                    { \fp_to_int:N \l__statistics_graph_ims_step_fp }
            \bool_while_do:nn {
                \fp_compare_p:n {
                    \l__statistics_graph_minorsteps_int * \l__statistics_graph_ims_rang
                    > \l__statistics_graph_ims_threshold_fp
                }
            }{
                \tl_map_inline:nn {{2}{5}{10}} {
                    \fp_compare:nF {
                        \l__statistics_graph_minorsteps_int * \l__statistics_graph_ims_
                        > \l__statistics_graph_ims_threshold_fp * ##1
                    }{
```

```
                        \int_compare:nT {
                            \int_mod:nn{\l__statistics_graph_minorsteps_int}{##1} = 0
                        }{
                            \int_set:Nn
                                \l__statistics_graph_minorsteps_int
                                { \l__statistics_graph_minorsteps_int / ##1 }
                            \tl_map_break:
                        }
                    }
                }
                \fp_compare:nT {
                    \l__statistics_graph_minorsteps_int * \l__statistics_graph_ims_rang
                    > \l__statistics_graph_ims_threshold_fp
                }{
                    \tl_map_inline:nn {{3}{2}{5}{\l__statistics_graph_minorsteps_int}}
                        \int_compare:nT {
                            \int_mod:nn{\l__statistics_graph_minorsteps_int}{##1} = 0
                        }{
                            \int_set:Nn
                                \l__statistics_graph_minorsteps_int
                                { \l__statistics_graph_minorsteps_int / ##1 }
                            \tl_map_break:
                        }
                    }
                }
            }
            \int_compare:nNnTF \l__statistics_graph_minorsteps_int > 1 {
                \use:x { \exp_not:n {
                    \pgfkeyssetvalue
                    {/tikz/data~visualization/minor~steps~between~steps}
                    }
                    { \int_eval:n {\l__statistics_graph_minorsteps_int-1} }
                }
            }{
                \pgfkeyssetvalue
                    {/tikz/data~visualization/minor~steps~between~steps}
                    {}
            }
            \tl_clear:N \l__statistics_graph_maxminor_tl
        }
    }
},
integer~minor~steps/.default=50,
integer~minor~steps*/.style={
    compute~step/.append = {
        \tl_set_eq:NN \l_tmpa_tl \tikz@lib@dv@step
        \tl_if_eq:NNF \l_tmpa_tl \relax {
            \fp_compare:nT { \l_tmpa_tl < 1 } {
                \tl_set:Nx \tikz@lib@dv@step {1}
            }
        }
    },
    integer~minor~steps=#1,
},
integer~minor~steps*/.default=50,
}
```

First define a lot of variables:

```
\bool_new:N \l__statistics_graph_allranges_bool

\fp_new:N \l__statistics_graph_curvalue_fp
\fp_new:N \l__statistics_graph_curheight_fp
\fp_new:N \l__statistics_graph_prevheight_fp
\fp_new:N \l__statistics_graph_maxheight_fp
```

```
\fp_new:N \l__statistics_graph_minvalue_fp
\fp_new:N \l__statistics_graph_maxvalue_fp
\fp_new:N \g__statistics_graph_xstep_fp
\int_new:N \g__statistics_graph_last_int

\tl_new:N \l__statistics_graph_tikzdata_tl
\tl_new:N \l__statistics_graph_tikzinfo_tl
\clist_new:N \l__statistics_graph_tikzincludex_clist
\clist_new:N \l__statistics_graph_tikzincludey_clist
\tl_new:N \l__statistics_graph_tikzpicture_tl
```

No scale for counts, divide by total for freqs

```
\fp_new:N \l__statistics_graph_scale_fp
\fp_new:N \l__statistics_graph_counts_scale_fp
\fp_new:N \l__statistics_graph_frequencies_scale_fp
\fp_set:Nn \l__statistics_graph_counts_scale_fp { 1 }


\NewDocumentCommand \StatsGraph { +O{} +m +O{} } {
    \group_begin:
    \int_gincr:N \g__statistics_graph_last_int
```

Read saved x step, for automatic margin and histogram y step

```
    \tl_set:Nx \l_tmpa_tl {
        \exp_not:n { g__statistics_graph_xstep_ }
        \int_use:N \g__statistics_graph_last_int
        \exp_not:n { _tl }
    }
    \tl_if_exist:cTF { \l_tmpa_tl } {
        \fp_gset:Nn \g__statistics_graph_xstep_fp
            { \tl_use:c {\l_tmpa_tl} }
    }{
        \fp_gset:Nn \g__statistics_graph_xstep_fp { \c_one_int }
    }
```

Handle optional settings

```
    \__statistics_setup:nn { graph } { #1, #3 }
```

Get the data inline or from a variable

```
    \tl_if_single:nTF { #2 } {
```

Generate meaningful error by using the non-existent variable.

```
        \cs_if_exist:NF #2 { #2 }
        \tl_set_eq:NN \l__statistics_data_tl #2
    }{
        \tl_set:Nn \l__statistics_data_tl { #2 }
    }
```

Zero the maximum height in the graph, and setup min and max values.

```
    \fp_zero:N \l__statistics_graph_maxheight_fp
    \fp_set:Nn \l__statistics_graph_minvalue_fp {inf}
    \fp_set:Nn \l__statistics_graph_maxvalue_fp {-inf}
```

The following loop does 2 things:

- Counting the number of ranges and the total population count
- Detecting whether the ranges are intervals or single numbers

```
    \fp_zero:N \l__statistics_total_fp
    \int_zero:N \l__statistics_nbvals_int
    \bool_set_true:N \l__statistics_graph_allranges_bool
    \keyval_parse:NNV
            \__statistics_graph_prepare:n
            \__statistics_graph_prepare:nn
            \l__statistics_data_tl
```

The remainder is different whether we do histogram, cumulative, or comb

```
    \tl_clear:N \l__statistics_graph_tikzdata_tl
    \tl_clear:N \l__statistics_graph_tikzinfo_tl
    \int_zero:N \l__statistics_currange_int
    \bool_if:NTF \l__statistics_graph_allranges_bool {
        \bool_if:NTF \l__statistics_graph_cumulative_bool {
```

We draw a cumulative distribution function

```
            \__statistics_graph_dopicture_cumulative:
        }{
```

We draw an histogram

```
            \__statistics_graph_dopicture_hist:
        }
    }{
```

We draw a comb graph

```
        \__statistics_graph_dopicture_comb:
    }
```

Write xstep info to aux file

```
    \iow_now:Nx \@auxout {
        \exp_not:n {
            \ExplSyntaxOn
            \tl_gset:cn
        }
        {
            \exp_not:n {g__statistics_graph_xstep_}
            \int_use:N \g__statistics_graph_last_int
            \exp_not:n {_tl}
        }
        {
            \fp_to_decimal:N \g__statistics_graph_xstep_fp
        }
        \exp_not:n {
            \ExplSyntaxOff
        }
    }
    \group_end:
}
```

First pass

```
\cs_new_protected_nopar:Nn \__statistics_graph_prepare:n {
    \__statistics_graph_prepare:nn { #1 } { 1 }
}
\cs_new_protected_nopar:Nn \__statistics_graph_prepare:nn {
    \int_incr:N \l__statistics_nbvals_int
    \fp_add:Nn \l__statistics_total_fp { #2 }
    \exp_args:Nx \tl_if_eq:nnF { \tl_head:n {#1} }{ \IN } {
        \bool_set_false:N \l__statistics_graph_allranges_bool
    }
}
```

Shared utility functions

```
\cs_new_protected_nopar:Nn \__statistics_graph_addpoint:nnn {
    \tl_put_right:Nx \l__statistics_graph_tikzdata_tl {
        \exp_not:N \pgfkeys {
            \exp_not:n { /data~point/name = #1 }
                \int_use:N \l__statistics_currange_int
            \exp_not:n { ,/data~point/x = } \fp_eval:n { #2 }
            \exp_not:n { ,/data~point/y = } \fp_eval:n { #3 }
        }
        \exp_not:n { \pgfdatapoint }
```

```
    }
}
\cs_new_protected_nopar:Nn \__statistics_graph_outlier: {
    \tl_put_right:Nn \l__statistics_graph_tikzdata_tl {
        \pgfkeys{/data~point/outlier = true}
        \pgfdatapoint
        \pgfkeys{/data~point/outlier = }
    }
}
\cs_new_protected_nopar:Nn \__statistics_graph_setup:n {
    \fp_set_eq:Nc \l__statistics_graph_hmargin_tl {l__statistics_graph_#1_hmargin_tl}
    \tl_set_eq:Nc \l__statistics_graph_unit_tl { l__statistics_graph_#1_unit_tl }
    \tl_set_eq:Nc \l__statistics_graph_vmargin_tl
        {l__statistics_graph_#1_ \l__statistics_graph_unit_tl _vmargin_tl}
    \tl_set_eq:Nc
        \l__statistics_graph_options_yaxis_tl
        {l__statistics_graph_options_#1_ \l__statistics_graph_unit_tl axis_tl}
    \cs_set_eq:Nc
        \__statistics_graph_y_format:n
        {__statistics_graph_#1_ \l__statistics_graph_unit_tl _format:n}
    \cs_set_eq:Nc
        \__statistics_graph_values_format:n
        {__statistics_graph_#1_values_format:n}
    \fp_set_eq:NN
        \l__statistics_graph_frequencies_scale_fp
        \l__statistics_total_fp
    \fp_set_eq:Nc
        \l__statistics_graph_scale_fp
        {l__statistics_graph_ \l__statistics_graph_unit_tl _scale_fp}
}
\cs_new_protected_nopar:Nn \__statistics_graph_update_minmaxval:NN {
    \fp_set:Nn \l__statistics_graph_minvalue_fp {
        min( \l__statistics_graph_minvalue_fp, #1 )
    }
    \fp_set:Nn \l__statistics_graph_maxvalue_fp {
        max( \l__statistics_graph_maxvalue_fp, #2 )
    }
}
\cs_new_protected_nopar:Nn \__statistics_graph_update_maxheight: {
    \fp_set:Nn \l__statistics_graph_maxheight_fp {
        max( \l__statistics_graph_maxheight_fp , \l__statistics_graph_curheight_fp )
    }
}
\cs_new_protected_nopar:Nn \__statistics_graph_handle_hmargin: {
    \group_begin:
    \tl_set:Nx \min { \fp_to_decimal:N \l__statistics_graph_minvalue_fp }
    \tl_set:Nx \max { \fp_to_decimal:N \l__statistics_graph_maxvalue_fp }
    \tl_set:Nx \range {
        \fp_eval:n { \l__statistics_graph_maxvalue_fp - \l__statistics_graph_minvalue_f
    }
    \tl_set:Nx \xstep { \fp_to_decimal:N \g__statistics_graph_xstep_fp }
    \exp_args:NNV \fp_gset:Nn \g_tmpa_fp \l__statistics_graph_hmargin_tl
    \group_end:
    \clist_put_right:Nx \l__statistics_graph_tikzincludex_clist {
        \fp_eval:n { \l__statistics_graph_minvalue_fp - \g_tmpa_fp }
    }
    \clist_put_right:Nx \l__statistics_graph_tikzincludex_clist {
        \fp_eval:n { \l__statistics_graph_maxvalue_fp + \g_tmpa_fp }
    }
}
\cs_new_protected_nopar:Nn \__statistics_graph_handle_vmargin: {
    \group_begin:
    \tl_set:Nn \min { 0 }
```

```
    \tl_set:Nx \max { \fp_to_decimal:N \l__statistics_graph_maxheight_fp }
    \tl_set_eq:NN \range \max
    \exp_args:NNV \fp_gset:Nn \g_tmpa_fp \l__statistics_graph_vmargin_tl
    \group_end:
    \clist_put_right:Nx \l__statistics_graph_tikzincludey_clist {
        \fp_eval:n { \l__statistics_graph_maxheight_fp + \g_tmpa_fp }
    }
}
```

Second pass, histogram

```
\cs_new_protected_nopar:Nn \__statistics_graph_dopicture_hist: {
    \__statistics_graph_setup:n {histogram}
```

Loop through the list again to fill tikz data and labels

```
    \keyval_parse:NNV
            \__statistics_graph_make_hist:n
            \__statistics_graph_make_hist:nn
            \l__statistics_data_tl
```

Maybe add a legend

```
    \tl_if_empty:NF \l__statistics_graph_legend_x_tl {
        \group_begin:
        \tl_set:Nx \min { \fp_to_decimal:N \l__statistics_graph_minvalue_fp }
        \tl_set:Nx \max { \fp_to_decimal:N \l__statistics_graph_maxvalue_fp }
        \tl_set:Nx \range {
            \fp_eval:n { \l__statistics_graph_maxvalue_fp - \l__statistics_graph_minval
        }
        \tl_set:Nx \xstep { \fp_to_decimal:N \g__statistics_graph_xstep_fp }
        \exp_args:NNV \fp_gset:Nn \g_tmpa_fp \l__statistics_graph_legend_x_tl
        \exp_args:NNV \fp_gset:Nn \g_tmpb_fp \l__statistics_graph_legend_w_tl
        \group_end:
        \tl_set:Nx \l__statistics_graph_legend_x_tl { \fp_to_decimal:N \g_tmpa_fp }
        \tl_set:Nx \l__statistics_graph_legend_w_tl { \fp_to_decimal:N \g_tmpb_fp }

        \group_begin:
        \tl_set:Nn \min { 0 }
        \tl_set:Nx \max { \fp_to_decimal:N \l__statistics_graph_maxheight_fp }
        \tl_set_eq:NN \range \max
        \tl_set:Nx \xstep { \fp_to_decimal:N \g__statistics_graph_xstep_fp }
        \tl_set_eq:NN \width \l__statistics_graph_legend_w_tl
        \tl_set:Nx \total { \fp_to_decimal:N \l__statistics_total_fp }
        \exp_args:NNV \fp_gset:Nn \g_tmpb_fp \l__statistics_graph_legend_h_tl
        \tl_set:Nx \height { \fp_to_decimal:N \g_tmpb_fp }
        \exp_args:NNV \fp_gset:Nn \g_tmpa_fp \l__statistics_graph_legend_y_tl
        \group_end:

        \tl_put_right:Nx \l__statistics_graph_tikzinfo_tl {
            \exp_not:n { \path (visualization~cs }
            \token_to_str:N : \exp_not:n { x= }
            \exp_not:V \l__statistics_graph_legend_x_tl
            \exp_not:n { ,y= }
            \fp_to_decimal:N \g_tmpa_fp
            \exp_not:n { ) coordinate (LSW) (visualization~cs }
            \token_to_str:N : \exp_not:n { x= }
            \fp_eval:n {
                \l__statistics_graph_legend_x_tl +
                \l__statistics_graph_legend_w_tl
            }
            \exp_not:n { ,y= }
            \fp_eval:n { \g_tmpa_fp + \g_tmpb_fp }
            \exp_not:n { ) coordinate (LNE);
                \node[ fit=(LSW)~(LNE), draw, inner~sep=0pt,
            }
            \exp_not:V \l__statistics_graph_options_histogram_tl
```

```
            \exp_not:N ,
            \exp_not:V \l__statistics_graph_options_legend_tl
            \exp_not:n { ] {}; }
        }
    }
```

Create the picture itself

```
    \__statistics_graph_handle_hmargin:
    \__statistics_graph_handle_vmargin:
    \tl_set:Nx \l__statistics_graph_tikzpicture_tl {
        \exp_not:n { \begin{tikzpicture}[ }
            \exp_not:V \l__statistics_graph_options_pic_tl
        \exp_not:n { ] \datavisualization
                [scientific~axes = } {
                        \exp_not:V
                            \l__statistics_graph_options_system_tl
                    }
        \exp_not:n { , x~axis = } {
                    \exp_not:n { include~value/.list = } {
                        \exp_not:V \l__statistics_graph_tikzincludex_clist
                    }
                    \exp_not:n { , ticks = { tick~typesetter/.code = {
                        $\__statistics_graph_values_format:n { \fp_eval:n{####1} }$ }},
                    \exp_not:V
                        \l__statistics_graph_options_histogram_xaxis_tl
                    \exp_not:n { , }
                    \exp_not:V
                        \c__statistics_graph_savexstep_tl
                }
        \exp_not:n { , y~axis = } {
                    \exp_not:n { include~value/.list = } {
                        \exp_not:V \l__statistics_graph_tikzincludey_clist
                    }
                    \exp_not:n { , }
                    \exp_not:V
                        \l__statistics_graph_options_yaxis_tl
                }
        \exp_not:n { , visualize~as~line = histogram,
                       histogram = } {
            \exp_not:n { polygon, style = } {
                        \exp_not:V \l__statistics_graph_options_histogram_tl
                } }
        \exp_not:n { ] data [set = histogram, format = TeX~code] } {
            \exp_not:V \l__statistics_graph_tikzdata_tl
        }
        \exp_not:n { info' } {
            \exp_not:V \l__statistics_graph_userpreinfo_tl
        }
        \exp_not:n { info } {
            \exp_not:V \l__statistics_graph_tikzinfo_tl
            \exp_not:V \l__statistics_graph_userpostinfo_tl
        }
        \exp_not:n { ; \end{tikzpicture} }
    }
    \tl_use:N \l__statistics_graph_tikzpicture_tl
}
\cs_new_protected_nopar:Nn \__statistics_graph_make_hist:n {
    \__statistics_graph_make_hist:nn { #1 } { 1 }
}
\cs_new_protected_nopar:Nn \__statistics_graph_make_hist:nn {
    \int_incr:N \l__statistics_currange_int
```

Extract interval data

```
    \__statistics_parse_range:w #1 \q_stop
```

Compute rectangle height

```
    \fp_set:Nn \l__statistics_graph_curheight_fp {
        (#2) / ( \l__statistics_range_max_fp -
                    \l__statistics_range_min_fp)
    }
```

Add margins to axes

```
    \__statistics_graph_update_minmaxval:NN \l__statistics_range_min_fp \l__statistics_
    \__statistics_graph_update_maxheight:
```

Check if we want to show this element

```
    \__statistics_set_if_shown:N \l_tmpa_bool
```

Append the rectangle to the TikZ datavisualization content

```
    \__statistics_graph_addpoint:nnn { SW }
        { \l__statistics_range_min_fp }
        { 0 }
    \bool_if:NF \l_tmpa_bool {
```

Add an outlier point to inhibit the rectangle drawing

```
        \__statistics_graph_outlier:
    }
    \__statistics_graph_addpoint:nnn { NW }
        { \l__statistics_range_min_fp }
        { \l__statistics_graph_curheight_fp }
    \bool_if:NF \l_tmpa_bool { \__statistics_graph_outlier: }
    \__statistics_graph_addpoint:nnn { NE }
        { \l__statistics_range_max_fp }
        { \l__statistics_graph_curheight_fp }
    \bool_if:NF \l_tmpa_bool { \__statistics_graph_outlier: }
    \__statistics_graph_addpoint:nnn { SE }
        { \l__statistics_range_max_fp }
        { 0 }
    \bool_if:NT \l_tmpa_bool {
```

Maybe append a freq or count label on middle top of the rect

```
        \bool_if:NT \l__statistics_graph_areas_bool {
            \__statistics_graph_addlabel:nn
                \__statistics_graph_y_format:n
                { \fp_eval:n {#2 / \l__statistics_graph_scale_fp} }
        }
    }
    \__statistics_graph_outlier:
}
\cs_new_protected_nopar:Nn \__statistics_graph_addlabel:nn {
    \tl_put_right:Nx \l__statistics_graph_tikzinfo_tl {
        \exp_not:n { \path (NW }
        \int_use:N \l__statistics_currange_int
        \exp_not:n { ) -- node[ }
            \exp_not:V \l__statistics_graph_options_areas_tl
        \exp_not:N ] {
            \exp_not:n { $ #1 } { #2 } \exp_not:N $
        } \exp_not:n { (NE }
        \int_use:N \l__statistics_currange_int
        \exp_not:n { ); }
    }
}
```

second pass, comb

```
\cs_new_protected:Nn \__statistics_graph_dopicture_comb: {
    \__statistics_graph_setup:n {comb}
```

Loop through the list again to fill tikz data and labels

```
    \keyval_parse:NNV
            \__statistics_graph_make_comb:n
            \__statistics_graph_make_comb:nn
            \l__statistics_data_tl
```

Create the picture itself

```
    \__statistics_graph_handle_hmargin:
    \__statistics_graph_handle_vmargin:
    \tl_set:Nx \l__statistics_graph_tikzpicture_tl {
        \exp_not:n { \begin{tikzpicture}[ }
            \exp_not:V \l__statistics_graph_options_pic_tl
        \exp_not:n { ] \datavisualization
                [scientific~axes = } {
                        \exp_not:V
                            \l__statistics_graph_options_system_tl
                    }
        \exp_not:n { , x~axis = } {
                    \exp_not:n { include~value/.list = } {
                        \exp_not:V \l__statistics_graph_tikzincludex_clist
                    }
                    \exp_not:n { , ticks = { tick~typesetter/.code = {
                        $\__statistics_graph_values_format:n { \fp_eval:n{####1} }$ }},
                    \exp_not:V
                        \l__statistics_graph_options_comb_xaxis_tl
                    \exp_not:n { , }
                    \exp_not:V
                        \c__statistics_graph_savexstep_tl
                }
        \exp_not:n { , y~axis = } {
                    \exp_not:n { include~value/.list = } {
                        \exp_not:V \l__statistics_graph_tikzincludey_clist
                    }
                    \exp_not:n { , }
                    \exp_not:n { , ticks = { tick~typesetter/.code = {
                        $\__statistics_graph_y_format:n { \fp_eval:n{####1} }$ }}, }
                    \exp_not:V
                        \l__statistics_graph_options_yaxis_tl
                }
        \exp_not:n { , visualize~as~line = bar~graph,
                       bar~graph = } {
            \exp_not:n { style = } {
                        \exp_not:V \l__statistics_graph_options_comb_tl
                } }
        \exp_not:n { ] data [set = bar~graph, format = TeX~code] } {
            \exp_not:V \l__statistics_graph_tikzdata_tl
        }
        \exp_not:n { info' } {
            \exp_not:V \l__statistics_graph_userpreinfo_tl
        }
        \exp_not:n { info } {
            \exp_not:V \l__statistics_graph_tikzinfo_tl
            \exp_not:V \l__statistics_graph_userpostinfo_tl
        }
        \exp_not:n { ; \end{tikzpicture} }
    }
    \tl_use:N \l__statistics_graph_tikzpicture_tl
}
\cs_new_protected_nopar:Nn \__statistics_graph_make_comb:n {
    \__statistics_graph_make_comb:nn { #1 } { 1 }
}
\cs_new_protected_nopar:Nn \__statistics_graph_make_comb:nn {
    \int_incr:N \l__statistics_currange_int
```

Set value

```
    \fp_set:Nn \l__statistics_graph_curvalue_fp {
        #1
    }
```

Compute height

```
    \fp_set:Nn \l__statistics_graph_curheight_fp {
        (#2) / \l__statistics_graph_scale_fp
    }
```

Add margins to axes

```
    \__statistics_graph_update_minmaxval:NN
        \l__statistics_graph_curvalue_fp \l__statistics_graph_curvalue_fp
    \__statistics_graph_update_maxheight:
```

Check if we want to show this element

```
    \__statistics_set_if_shown:N \l_tmpa_bool
```

Append the bar to the TikZ datavisualization content

```
    \__statistics_graph_addpoint:nnn { S }
        { \l__statistics_graph_curvalue_fp }
        { 0 }
    \bool_if:NF \l_tmpa_bool {
```

add an outlier to inhibit the bar drawing

```
        \__statistics_graph_outlier:
    }
    \__statistics_graph_addpoint:nnn { N }
        { \l__statistics_graph_curvalue_fp }
        { \l__statistics_graph_curheight_fp }
    \__statistics_graph_outlier:
}
```

second pass, cumulative

```
\cs_new_protected_nopar:Nn \__statistics_graph_dopicture_cumulative: {
    \__statistics_graph_setup:n {cumulative}
```

Increasing or decreasing starting point

```
    \bool_if:NTF \l__statistics_graph_decreasing_bool {
        \fp_set_eq:NN \l__statistics_curtotal_fp
                      \l__statistics_total_fp
    }{
        \fp_zero:N \l__statistics_curtotal_fp
    }
    \fp_set:Nn \l__statistics_graph_curheight_fp {
        \l__statistics_curtotal_fp
            / \l__statistics_graph_scale_fp
    }
    \__statistics_graph_update_maxheight:
```

Loop through the list again to fill tikz data and labels

```
    \keyval_parse:NNV
            \__statistics_graph_make_cumulative:n
            \__statistics_graph_make_cumulative:nn
            \l__statistics_data_tl
```

After the last point we should be piecewise constant, which is the $N+1$-th item for `showonly` purposes. We call `\__statistics_graph_handle_hmargin:` even if we will add actual data in the margin, because that method computes the correct value for the margin from the options.

```
    \__statistics_graph_handle_hmargin:
    \int_incr:N \l__statistics_currange_int
    \__statistics_set_if_shown:N \l_tmpa_bool
```

```
    \bool_if:NF \l_tmpa_bool { \__statistics_graph_outlier: }
    \__statistics_graph_addpoint:nnn { E }
        { \l__statistics_graph_maxvalue_fp + \g_tmpa_fp }
        { \l__statistics_graph_curheight_fp }
```

Before the first point we should be piecewise constant. We stash the TikZ data away to prepend the first point and maybe an outlier if the segment should be hidden, then append the stashed data. The initial segment is numbered 0.

```
    \tl_set_eq:NN \l_tmpa_tl \l__statistics_graph_tikzdata_tl
    \tl_clear:N \l__statistics_graph_tikzdata_tl
    \int_zero:N \l__statistics_currange_int
    \__statistics_graph_addpoint:nnn { B }
        { \l__statistics_graph_minvalue_fp - \g_tmpa_fp }
        { \l__statistics_graph_maxheight_fp - \l__statistics_graph_curheight_fp }
    \__statistics_set_if_shown:N \l_tmpa_bool
    \bool_if:NF \l_tmpa_bool { \__statistics_graph_outlier: }
    \tl_put_right:NV \l__statistics_graph_tikzdata_tl \l_tmpa_tl
```

Create the picture itself

```
    \__statistics_graph_handle_vmargin:
    \tl_set:Nx \l__statistics_graph_tikzpicture_tl {
        \exp_not:n { \begin{tikzpicture}[ }
            \exp_not:V \l__statistics_graph_options_pic_tl
        \exp_not:n { ] \datavisualization
                [scientific~axes = } {
                        \exp_not:V
                            \l__statistics_graph_options_system_tl
                    }
        \exp_not:n { , x~axis = } {
                    \exp_not:n { include~value/.list = } {
                        \exp_not:V \l__statistics_graph_tikzincludex_clist
                    }
                    \exp_not:n { , ticks = { tick~typesetter/.code = {
                        $\__statistics_graph_values_format:n { \fp_eval:n{####1} }$ }},
                    \exp_not:V
                        \l__statistics_graph_options_cumulative_xaxis_tl
                    \exp_not:n { , }
                    \exp_not:V
                        \c__statistics_graph_savexstep_tl
                }
        \exp_not:n { , y~axis = } {
                    \exp_not:n { include~value/.list = } {
                        \exp_not:V \l__statistics_graph_tikzincludey_clist
                    }
                    \exp_not:n { , }
                    \exp_not:n { , ticks = { tick~typesetter/.code = {
                        $\__statistics_graph_y_format:n { \fp_eval:n{####1} }$ }}, }
                    \exp_not:V
                        \l__statistics_graph_options_yaxis_tl
                }
        \exp_not:n { , visualize~as~line = cumulative,
                       cumulative = } {
            \exp_not:n { style = } {
                        \exp_not:V \l__statistics_graph_options_cumulative_tl
                } }
        \exp_not:n { ] data [set = cumulative, format = TeX~code] } {
            \exp_not:V \l__statistics_graph_tikzdata_tl
        }
        \exp_not:n { info' } {
            \exp_not:V \l__statistics_graph_userpreinfo_tl
        }
        \exp_not:n { info } {
            \exp_not:V \l__statistics_graph_tikzinfo_tl
            \exp_not:V \l__statistics_graph_userpostinfo_tl
```

```
        }
        \exp_not:n { ; \end{tikzpicture} }
    }
    \tl_use:N \l__statistics_graph_tikzpicture_tl
}
\cs_new_protected_nopar:Nn \__statistics_graph_make_cumulative:n {
    \__statistics_graph_make_hist:nn { #1 } { 1 }
}
\cs_new_protected_nopar:Nn \__statistics_graph_make_cumulative:nn {
```

Extract interval data

```
    \__statistics_parse_range:w #1 \q_stop
```

Compute running total and new height

```
    \fp_set_eq:NN
        \l__statistics_graph_prevheight_fp
        \l__statistics_graph_curheight_fp
    \bool_if:NTF \l__statistics_graph_decreasing_bool {
        \fp_sub:Nn \l__statistics_curtotal_fp { #2 }
    }{
        \fp_add:Nn \l__statistics_curtotal_fp { #2 }
    }
    \fp_set:Nn \l__statistics_graph_curheight_fp {
        \l__statistics_curtotal_fp
            / \l__statistics_graph_scale_fp
    }
    \__statistics_graph_update_minmaxval:NN \l__statistics_range_min_fp \l__statistics_
    \__statistics_graph_update_maxheight:
```

Add points

```
    \int_incr:N \l__statistics_currange_int
    \__statistics_graph_addpoint:nnn { L }
        { \l__statistics_range_min_fp }
        { \l__statistics_graph_prevheight_fp }
```

If we don't want to show this segment, add an outlier so that the line is not drawn.

```
    \__statistics_set_if_shown:N \l_tmpa_bool
    \bool_if:NF \l_tmpa_bool { \__statistics_graph_outlier: }
    \__statistics_graph_addpoint:nnn { R }
        { \l__statistics_range_max_fp }
        { \l__statistics_graph_curheight_fp }
```

TODO: Median and co

```
}
```

2.4 Consolitate and sort values

```
\clist_new:N \l__statistics_compute_data_clist
\int_new:N \l__statistics_compute_count_int

\fp_new:N \l__statistics_compute_curvalue_fp
\seq_new:N \l__statistics_data_seq

\NewDocumentCommand \StatsSortData { +O{} u{=} m +O{} } {
    \group_begin:
```

Handle optional settings (there are none currently) `\@@_setup:nn { rangedata } { #1, #5`
Get the data inline or from a variable

```
    \tl_if_single:nTF { #3 } {
```

Generate meaningful error by using the non-existent variable.

```
        \cs_if_exist:NF #3 { #3 }
        \tl_set_eq:NN \l__statistics_data_tl #3
    }{
        \tl_set:Nn \l__statistics_data_tl { #3 }
```

```
    }
```

Sort the data according to values. We go through sequences because `\clist_sort:Nn` puts braces around the elements which prevents `\keyval_parse:NNn` to detect the equal sign.

```
    \seq_set_from_clist:NN \l__statistics_data_seq \l__statistics_data_tl
    \seq_sort:Nn \l__statistics_data_seq {
        \seq_set_split:Nnn \l_tmpa_seq {=} { ##1 }
        \seq_set_split:Nnn \l_tmpb_seq {=} { ##2 }
        \fp_compare:nNnTF
            { \seq_item:Nn \l_tmpa_seq {1} } > { \seq_item:Nn \l_tmpb_seq {1} }
        {
            \sort_return_swapped:
        }{
            \sort_return_same:
        }
    }
```

Append a sentinel NaN to ensure the last value is not trimmed. This value is particularly suitable because NaN is equal to no fp (even itself).

```
    \seq_put_right:Nn \l__statistics_data_seq { nan = 0 }
    \tl_set:Nx \l__statistics_data_tl { \seq_use:Nn \l__statistics_data_seq {,} }
```

Build the resulting clist while grouping equal values

```
    \clist_clear:N \l__statistics_compute_data_clist
    \int_zero:N \l__statistics_compute_count_int
    \fp_zero:N \l__statistics_compute_curvalue_fp
    \keyval_parse:NNV
            \__statistics_accumulate:n
            \__statistics_accumulate:nn
            \l__statistics_data_tl
    \exp_args:NNNV
    \group_end:
    \clist_set:Nn #2 \l__statistics_compute_data_clist
}
\cs_new_protected_nopar:Nn \__statistics_accumulate:n {
    \__statistics_accumulate:nn { #1 } { 1 }
}
\cs_new_protected_nopar:Nn \__statistics_accumulate:nn {
    \fp_compare:nNnTF { #1 } = { \l__statistics_compute_curvalue_fp } {
        \int_add:Nn \l__statistics_compute_count_int { #2 }
    }{
        \int_compare:nNnT { \l__statistics_compute_count_int } > { 0 } {
            \clist_put_right:Nx \l__statistics_compute_data_clist {
                \fp_to_decimal:N \l__statistics_compute_curvalue_fp
                \exp_not:n { = }
                \exp_not:V \l__statistics_compute_count_int
            }
        }
        \fp_set:Nn \l__statistics_compute_curvalue_fp { #1 }
        \int_set:Nn \l__statistics_compute_count_int { #2 }
    }
}
```

2.5 Count values in ranges to generate grouped counts

```
\NewDocumentCommand \StatsRangeData { +O{} u{=} m +r() +O{} } {
    \group_begin:
```

Handle optional settings (there are none currently) `\@@_setup:nn { rangedata } { #1, #5`
Get the data inline or from a variable

```
    \tl_if_single:nTF { #3 } {
```

Generate meaningful error by using the non-existent variable.

```
        \cs_if_exist:NF #3 { #3 }
        \tl_set_eq:NN \l__statistics_data_tl #3
```

```
    }{
        \tl_set:Nn \l__statistics_data_tl { #3 }
    }
```

Loop through the ranges and count values into them

```
    \clist_clear:N \l__statistics_compute_data_clist
    \clist_map_inline:nn { #4 } {
```

If not a range, bail out

```
        \exp_args:Nx \tl_if_eq:nnF { \tl_head:n {##1} }{ \IN } {
```

TODO: error message

```
            \clist_map_break:
        }
```

Extract interval data

```
        \__statistics_parse_range_full:w ##1 \q_stop
```

Loop through the point data and count matching values

```
        \int_zero:N \l__statistics_compute_count_int
        \keyval_parse:NNV
                \__statistics_range_count:n
                \__statistics_range_count:nn
                \l__statistics_data_tl
        \clist_put_right:Nx \l__statistics_compute_data_clist {
            \exp_not:n { ##1 = }
            \exp_not:V \l__statistics_compute_count_int
        }
    }
    \exp_args:NNNV
    \group_end:
    \clist_set:Nn #2 \l__statistics_compute_data_clist
}
\cs_new_protected_nopar:Nn \__statistics_range_count:n {
    \__statistics_range_count:nn { #1 } { 1 }
}
\cs_new_protected_nopar:Nn \__statistics_range_count:nn {
    \__statistics_if_in_range:nT { #1 } {
        \int_add:Nn \l__statistics_compute_count_int { #2 }
    }
}
```

⟨/package⟩