

# Automatic booktabs rules for tabular/tabular\*

Moritz R. Schäfer

2026/06/28 v1.1.2

## Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Example	1
1.2	Notes	2
<b>2</b>	<b>Interface and Implementation</b>	<b>2</b>
2.1	Environments	3
2.2	Public commands	4
2.3	Options	4
<b>3</b>	<b>Internal Helpers</b>	<b>5</b>

## 1 Introduction

The `booktabstabular` package provides wrapper environments around `tabular` and `tabular*` that insert `\toprule` at the beginning, `\bottomrule` at the end, and locally remap the legacy `\hline` and `\cline` commands to their booktabs equivalents `\midrule` and `\cmidrule`. It also provides commands and options to temporarily override the standard `tabular` and `tabular*` environments so that existing documents can opt into this behaviour without rewriting each table.

### 1.1 Example

A typical opt-in workflow is:

```
\usepackage[override=global]{booktabstabular}

\begin{tabular}{lll}
  First  & Second & Third \\
  \hline % mapped to \midrule
  Alpha  & Beta   & Gamma
\end{tabular}
```

Alternatively, the wrapper environments can be used directly:

```

\begin{booktabstabular}{lll}
  First & Second & Third \\
\hline
  Alpha & Beta & Gamma
\end{booktabstabular}

```

Both should yield the following output: 

First	Second	Third
Alpha	Beta	Gamma

 As you can

see, `\toprule` and `\bottomrule` are automatically inserted, and `\hline` is remapped to `\midrule`, so the table is typeset according to booktabs guidelines. The same applies to `tabular*` when using the `booktabstabular*` environment or the overridden `tabular*` name.

```

\begin{booktabstabular*}{\textwidth}{l@{\extracolsep{\fill}}l}
  First & Second \\
\hline
  Alpha & Beta
\end{booktabstabular*}

```

which produces the slightly wider table 

First	Second
Alpha	Beta

.

## 1.2 Notes

This package deliberately does not insert any intermediate rule such as `\midrule` automatically; that remains the author's responsibility.

The end wrapper inserts `\bottomrule` at the legal alignment boundary, so the final row may be written either with or without a trailing `\\`.

The package loads the standard `array` package. This provides the table row hooks used internally to decide whether the wrapper must insert a final `\crrc` before `\bottomrule`. As a result, all `tabular` and `tabular*` environments in the document use `array`'s extended table implementation. This is intended to be compatible with normal document-level table code, including packages such as `siunitx`; code that patches low-level tabular internals may observe `array`'s definitions rather than the kernel definitions.

The package does not attempt to remap the legacy `\hline` and `\cline` outside of the wrapper environments, so they will continue to work as normal in documents that only use the wrappers for a subset of tables.

The package affects only `tabular` and `tabular*`. It does not modify `array`, `longtable`, or any other alignment environment.

## 2 Interface and Implementation

Since the package is rather small, we will present the interface and implementation together in the following subsections.

```

1 <*package>
2 \RequirePackage{booktabs}
3 \RequirePackage{array}
4 \ProvidesExplPackage
5   {booktabstabular}

```

```

6 {2026-06-28}
7 {1.1.2}
8 {tabular/tabular* wrappers with booktabs rules}

```

## 2.1 Environments

`origtabular` (*env.*) The `origtabular` and `origtabular*` environments are copies of the original `tabular` and `tabular*` environments. They are used internally but also made available for users who want to call the original versions if they have overridden the standard names.

```

9 \NewEnvironmentCopy { origtabular } { tabular }
10 \NewEnvironmentCopy { origtabular* } { tabular* }

```

`booktabstabular` (*env.*) The `booktabstabular` environment is a drop-in wrapper for `tabular`. It accepts the same arguments as `tabular`: an optional position specifier [t] or [b], followed by the column specification.

It calls the copied begin macro directly rather than nesting a separate environment, which keeps LaTeX's environment stack balanced when the custom end code is used.

```

11 \NewDocumentEnvironment { booktabstabular } { o m }
12 {
13   \group_begin:
14   \bool_set_true:N \l__booktabstabular_active_bool
15   \cs_gset_eq:NN \__booktabstabular_maybe_crcr: \prg_do_nothing:
16   \cs_set_eq:NN \hline \midrule
17   \cs_set_eq:NN \cline \cmidrule
18   \IfNoValueTF { #1 }
19     { \origtabular { #2 } }
20     { \origtabular [ #1 ] { #2 } }
21   \toprule
22 }
23 {
24   \__booktabstabular_maybe_crcr:
25   \bottomrule
26   \endorigtabular
27   \cs_gset_eq:NN \__booktabstabular_maybe_crcr: \prg_do_nothing:
28   \group_end:
29 }

```

`booktabstabular*` (*env.*) The `booktabstabular*` environment is a drop-in wrapper for `tabular*`. It accepts the width, an optional position specifier, and the column specification.

```

30 \NewDocumentEnvironment { booktabstabular* } { m o m }
31 {
32   \group_begin:
33   \bool_set_true:N \l__booktabstabular_active_bool
34   \cs_gset_eq:NN \__booktabstabular_maybe_crcr: \prg_do_nothing:
35   \cs_set_eq:NN \hline \midrule
36   \cs_set_eq:NN \cline \cmidrule
37   \IfNoValueTF { #2 }
38     { \use:c { origtabular* } { #1 } { #3 } }
39     { \use:c { origtabular* } { #1 } [ #2 ] { #3 } }
40   \toprule
41 }
42 {
43   \__booktabstabular_maybe_crcr:

```

```

44 \bottomrule
45 \use:c { endorigtabular* }
46 \cs_gset_eq:NN \__booktabstabular_maybe_crcr: \prg_do_nothing:
47 \group_end:
48 }

```

## 2.2 Public commands

`\overridetabular` After calling `\overridetabular`, the public `tabular` and `tabular*` environments are rebound to the wrapper environments provided by this package. Note that many classes, including the standard classes, use a `tabular` in unexpected places, *i.e.*, to typeset the list of authors. If you don't want that to be affected, you should use the `override` package option.

```

49 \NewDocumentCommand \overridetabular {}
50 {
51   \RenewEnvironmentCopy { tabular } { booktabstabular }
52   \RenewEnvironmentCopy { tabular* } { booktabstabular* }
53 }

```

`\restoretabular` The command `\restoretabular` restores the original definitions of `tabular` and `tabular*`.

```

54 \NewDocumentCommand \restoretabular {}
55 {
56   \RenewEnvironmentCopy { tabular } { origtabular }
57   \RenewEnvironmentCopy { tabular* } { origtabular* }
58 }

```

## 2.3 Options

We offer an option to automatically override the standard `tabular` and `tabular*` environments.

`override` The `override` option selects how `tabular` and `tabular*` are rebound. The full syntax is `override=none|table|global`.

`none` Do not override the standard environments.

`global` This has almost the same effect as calling `\overridetabular` after loading the package. The difference is that the package option automatically disables the override around `\maketitle` to avoid affecting the title page, while the command does not.

`table` Override `tabular` and `tabular*` only inside `table` and `table*` environments. This is the safest and **recommended** option.

```

59 \str_new:N \l__booktabstabular_override_str
60 \DeclareKeys [ booktabstabular ]
61 {
62   override .choices:nn =
63     { none , table , global }
64     { \str_set_eq:NN \l__booktabstabular_override_str \l_keys_choice_str } ,
65   override .initial:n = none ,

```

`overridetabular`      The `overridetabular` option is a legacy alias for `override=global`, and is provided for backwards compatibility.

```

66       overridetabular .meta:n               = { override = global } ,
67     }
68
69 \ProcessKeyOptions [ booktabstabular ]

```

### 3 Internal Helpers

`\_booktabstabular_maybe_crrc:`      The token at the start of the end code must have the meaning of `\crrc` when the final row is still open, and expand to nothing when the last user token was a row separator. Keep that state through the table hooks rather than running a conditional at the alignment boundary.

```

70 \bool_new:N \l__booktabstabular_active_bool
71 \cs_new_eq:NN \__booktabstabular_maybe_crrc: \prg_do_nothing:
72 \AddToHook { tbl/row/begin } [ booktabstabular ]
73 {
74     \bool_if:NT \l__booktabstabular_active_bool
75     { \cs_gset_eq:NN \__booktabstabular_maybe_crrc: \crrc }
76 }
77 \AddToHook { tbl/row/init } [ booktabstabular ]
78 {
79     \bool_if:NT \l__booktabstabular_active_bool
80     { \cs_gset_eq:NN \__booktabstabular_maybe_crrc: \prg_do_nothing: }
81 }

```

*(End of definition for \\_\_booktabstabular\_maybe\_crrc:.)*

`\_booktabstabular_apply_override:`      Applies the override according to the value of the `override` option.

```

82 \cs_new_protected:Nn \__booktabstabular_apply_override:
83 {
84     \str_case:Vn \l__booktabstabular_override_str
85     {
86         { none }     { }
87         { global } { \__booktabstabular_activate_global: }
88         { table }  { \__booktabstabular_activate_float: }
89     }
90 }

```

*(End of definition for \\_\_booktabstabular\_apply\_override:.)*

`\_booktabstabular_activate_global:`      The function called if the `override` option is set to `global`.

```

91 \cs_new_protected:Nn \__booktabstabular_activate_global:
92 {
93     \hook_gput_code:nnn { cmd/maketitle/before } { . } { \restoretabular }
94     \hook_gput_code:nnn { cmd/maketitle/after } { . } { \overridetabular }
95     \overridetabular
96 }

```

*(End of definition for \\_\_booktabstabular\_activate\_global:.)*

`\_booktabstabular_activate_float:` The function called if the `override` option is set to `table`.

```

97 \cs_new_protected:Nn \_booktabstabular_activate_float:
98 {
99   \hook_gput_code:nnn { env/table/begin } { . } { \overridetabular }
100   \hook_gput_code:nnn { env/table*/begin } { . } { \overridetabular }
101 }

```

*(End of definition for \\_booktabstabular\_activate\_float:.)*

Finally, we call the function at the end of the package to apply the override if requested.

```

102 \_booktabstabular_apply_override:
103 </package>

```