The package nicematrix*

F. Pantigny
fpantigny@wanadoo.fr

May 15, 2024

Abstract

The LaTeX package nicematrix provides new environments similar to the classical environments {tabular}, {array} and {matrix} of array and amsmath but with extended features.

$$\begin{array}{c|cccc|} & C_1 & C_2 & \cdots & C_n \\ \hline L_1 & a_{11} & a_{12} & \cdots & a_{1n} \\ L_2 & a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ L_n & a_{n1} & a_{n2} & \cdots & a_{nn} \end{array}$$

Product	dimensions (cm)			Price
	L	l	h	
small	3	5.5	1	30
standard	5.5	8	1.5	50.5
premium	8.5	10.5	2	80
extra	8.5	10	1.5	85.5
special	12	12	0.5	70

The package nicematrix is entirely contained in the file `nicematrix.sty`. This file may be put in the current directory or in a `texmf` tree. However, the best is to install nicematrix with a TeX distribution such as MiKTeX, TeX Live or MacTeX.

Remark
If you use LaTeX via Internet with, for example, Overleaf, you can upload the file `nicematrix.sty` in the repertory of your project in order to take full advantage of the latest version de nicematrix.[1]

This package can be used with `xelatex`, `lualatex`, `pdflatex` but also by the classical workflow `latex-dvips-ps2pdf` (or Adobe Distiller). *However, the file nicematrix.dtx of the present documentation should be compiled with XeLaTeX.*

This package requires and **loads** the packages l3keys2e, array, amsmath, pgfcore and the module shapes of PGF (tikz, which is a layer over PGF, is *not* loaded). The final user only has to load the package with `\usepackage{nicematrix}`.

The idea of nicematrix is to create PGF nodes under the cells and the positions of the rules of the tabular created by array and to use these nodes to develop new features. As usual with PGF, the coordinates of these nodes are written in the `aux` to be used on the next compilation and that's why nicematrix may need **several compilations**.[2]

Most features of nicematrix may be used without explicit use of PGF or TikZ (which, in fact, is not loaded by default).

A command `\NiceMatrixOptions` is provided to fix the options (the scope of the options fixed by this command is the current TeX group: they are semi-global).

*This document corresponds to the version 6.27x of nicematrix, at the date of 2024/05/06.

[1] The latest version of the file `nicematrix.sty` may be downloaded from the SVN server of TeXLive:
`https:www.tug.org/svn/texlive/trunk/Master/texmf-dist/tex/latex/nicematrix/nicematrix.sty`

[2] If you use Overleaf, Overleaf will do automatically a sufficient number of compilations.

1 The environments of this package

The package nicematrix defines the following new environments.

```
{NiceTabular}    {NiceArray}    {NiceMatrix}
{NiceTabular*}   {pNiceArray}   {pNiceMatrix}
{NiceTabularX}   {bNiceArray}   {bNiceMatrix}
                 {BNiceArray}   {BNiceMatrix}
                 {vNiceArray}   {vNiceMatrix}
                 {VNiceArray}   {VNiceMatrix}
```

The environments `{NiceArray}`, `{NiceTabular}` and `{NiceTabular*}` are similar to the environments `{array}`, `{tabular}` and `{tabular*}` of the package array (which is loaded by nicematrix).

The environments `{pNiceArray}`, `{bNiceArray}`, etc. have no equivalent in array.

The environments `{NiceMatrix}`, `{pNiceMatrix}`, etc. are similar to the corresponding environments of amsmath (which is loaded by nicematrix): `{matrix}`, `{pmatrix}`, etc.

The environment `{NiceTabularX}` is similar to the environment `{tabularx}` from the eponymous package.[3].

It's recommended to use primarily the classical environments and to use the environments of nicematrix only when some feature provided by these environments is used (this will save memory).

All the environments of the package nicematrix accept, between square brackets, an optional list of *key=value* pairs. **There must be no space before the opening bracket (`[`) of this list of options.**

2 The vertical space between the rows

It's well known that some rows of the arrays created by default with LaTeX are, by default, too close to each other. Here is a classical example.

```
$\begin{pmatrix}
\frac{1}{2} & -\frac{1}{2} \\
\frac{1}{3} & \frac{1}{4} \\
\end{pmatrix}$
```

$$\begin{pmatrix} \frac{1}{2} & -\frac{1}{2} \\ \frac{1}{3} & \frac{1}{4} \end{pmatrix}$$

Inspired by the package cellspace which deals with that problem, the package nicematrix provides two keys `cell-space-top-limit` and `cell-space-bottom-limit` similar to the parameters of cellspace called `\cellspacetoplimit` and `\cellspacebottomlimit`.
There is also a key `cell-space-limits` to set both parameters at once.
The initial value of these parameters is 0 pt in order to have for the environments of nicematrix the same behaviour as those of array and amsmath. However, a value of 1 pt would probably be a good choice and we suggest to set them with `\NiceMatrixOptions`.[4]

```
\NiceMatrixOptions{cell-space-limits = 1pt}
```

```
$\begin{pNiceMatrix}
\frac12 & -\frac12 \\
\frac13 & \frac14 \\
\end{pNiceMatrix}$
```

$$\begin{pmatrix} \frac{1}{2} & -\frac{1}{2} \\ \frac{1}{3} & \frac{1}{4} \end{pmatrix}$$

[3] In fact, it's possible to use directly the `X` columns in the environment `{NiceTabular}` (and the required width for the tabular is fixed by the key `width`): cf. p. 25

[4] One should remark that these parameters apply also to the columns of type `S` of siunitx whereas the package cellspace is not able to act on such columns of type `S`.

It's also possible to change these parameters for only a few rows by using the command \RowStyle provided by nicematrix (cf. p. 23).

3 The vertical position of the arrays

The package nicematrix provides a option baseline for the vertical position of the arrays. This option takes in as value an integer which is the number of the row on which the array will be aligned.

```
$A = \begin{pNiceMatrix}[baseline=2]
\frac{1}{\sqrt{1+p^2}} & p & 1-p \\
1 & 1 & 1 \\
1 & p & 1+p
\end{pNiceMatrix}$
```

$$A = \begin{pmatrix} \frac{1}{\sqrt{1+p^2}} & p & 1-p \\ 1 & 1 & 1 \\ 1 & p & 1+p \end{pmatrix}$$

It's also possible to use the option baseline with one of the special values t, c or b. These letters may also be used absolutely like the option of the environments {tabular} and {array} of array. The initial value of baseline is c.

In the following example, we use the option t (equivalent to baseline=t) immediately after an \item of list. One should remark that the presence of a \hline at the beginning of the array doesn't prevent the alignment of the baseline with the baseline of the first row (with {tabular} or {array} of array, one must use \firsthline).

```
\begin{enumerate}
\item an item
\smallskip
\item \renewcommand{\arraystretch}{1.2}
$\begin{NiceArray}[t]{lcccccc}
\hline
n   & 0 & 1 & 2 & 3 & 4  & 5 \\
u_n & 1 & 2 & 4 & 8 & 16 & 32
\hline
\end{NiceArray}$
\end{enumerate}
```

1. an item
2. $\begin{array}[t]{lcccccc} \hline n & 0 & 1 & 2 & 3 & 4 & 5 \\ u_n & 1 & 2 & 4 & 8 & 16 & 32 \\ \hline \end{array}$

However, it's also possible to use the tools of booktabs[5]: \toprule, \bottomrule, \midrule, etc.

```
\begin{enumerate}
\item an item
\smallskip
\item
$\begin{NiceArray}[t]{lcccccc}
\toprule
n   & 0 & 1 & 2 & 3 & 4  & 5 \\
\midrule
u_n & 1 & 2 & 4 & 8 & 16 & 32
\bottomrule
\end{NiceArray}$
\end{enumerate}
```

1. an item
2. $\begin{array}[t]{lcccccc} \hline n & 0 & 1 & 2 & 3 & 4 & 5 \\ \hline u_n & 1 & 2 & 4 & 8 & 16 & 32 \\ \hline \end{array}$

It's also possible to use the key baseline to align a matrix on an horizontal rule (drawn by \hline). In this aim, one should give the value line-*i* where *i* is the number of the row *following* the horizontal rule.

```
\NiceMatrixOptions{cell-space-limits=1pt}
```

[5] The extension booktabs is *not* loaded by nicematrix.

```
$A=\begin{pNiceArray}{cc|cc}[baseline=line-3]
\dfrac{1}{A} & \dfrac{1}{B} & 0 & 0 \\
\dfrac{1}{C} & \dfrac{1}{D} & 0 & 0 \\
\hline
0 & 0 & A & B \\
0 & 0 & D & D \\
\end{pNiceArray}$
```

$$A=\left(\begin{array}{cc|cc} \frac{1}{A} & \frac{1}{B} & 0 & 0 \\ \frac{1}{C} & \frac{1}{D} & 0 & 0 \\ \hline 0 & 0 & A & B \\ 0 & 0 & D & D \end{array}\right)$$

4 The blocks

4.1 General case

In the environments of `nicematrix`, it's possible to use the command `\Block` in order to place an element in the center of a rectangle of merged cells of the array.[6]
The command `\Block` must be used in the upper leftmost cell of the cells of the block with two mandatory arguments.

- The first argument is the size of the block with the syntax i-j where i is the number of rows of the block and j its number of columns.

  If this argument is empty, its default value is `1-1`. If the number of rows is not specified, or equal to `*`, the block extends until the last row (idem for the columns).

- The second argument is the content of the block. It's possible to use `\\` in that content to have a content on several lines. In `{NiceTabular}`, `{NiceTabular*}` and `{NiceTabularX}`, the content of the block is composed in text mode whereas, in the other environments, it is composed in math mode.

Here is an example of utilisation of the command `\Block` in mathematical matrices.

```
$\begin{bNiceArray}{cw{c}{1cm}c|c}[margin]
\Block{3-3}{A} & & & 0 \\
& & & \Vdots \\
& & & 0 \\
\hline
0 & \Cdots& 0 & 0
\end{bNiceArray}$
```

$$\left[\begin{array}{ccc|c} & & & 0 \\ & A & & \vdots \\ & & & 0 \\ \hline 0 & \cdots & 0 & 0 \end{array}\right]$$

One may wish to raise the size of the "A" placed in the block of the previous example. Since this element is composed in math mode, it's not possible to use directly a command like `\large`, `\Large` and `\LARGE`. That's why the command `\Block` provides an option between angle brackets to specify some TeX code which will be inserted before the beginning of the math mode.[7]

```
$\begin{bNiceArray}{cw{c}{1cm}c|c}[margin]
\Block{3-3}<\Large>{A} & & & 0 \\
0 & & & \Vdots \\
& & & 0 \\
\hline
0 & \Cdots& 0 & 0
\end{bNiceArray}$
```

$$\left[\begin{array}{ccc|c} & & & 0 \\ & \Large A & & \vdots \\ & & & 0 \\ \hline 0 & \cdots & 0 & 0 \end{array}\right]$$

[6] The spaces after a command `\Block` are deleted.

[7] This argument between angular brackets may also be used to insert a command of font such as `\bfseries` when the command `\\` is used in the content of the block. It's also possible to put in that optional argument the command `\rotate` provided by `nicematrix` (cf. part 14.5, p. 46).

In fact, the command \Block accepts as first optional argument (between square brackets) a list of couples *key=value*.
First, there are keys which are quick tools to control the apperance of the block.

- the key fill takes in as value a color and fills the block with that color;
- the key opacity sets the opacity of the filling color specified by fill;
- the key draw takes in as value a color and strokes the frame of the block with that color (the default value of that key is the current color of the rules of the array);
- the key color takes in as value a color and apply that color the content of the block but draws also the frame of the block with that color;
- the keys hlines, vlines and hvlines draw all the corresponding rules in the block;[8]
- the key line-width is the width of the rules (is relevant only when one of the keys draw, hvlines, vlines and hlines is used);
- the key rounded-corners requires rounded corners (for the frame drawn by draw and the shape drawn by fill) with a radius equal to the value of that key (the default value is 4 pt[9]).

Sometimes, these tools are not sufficient to control the appearance of the block. The following keys are more powerful but also more difficult to use. Moreover, the require the loading of TikZ by the user (with \usepackage{tikz}). By default, nicematrix does not load TikZ but only PGF, which is a sublayer of TikZ.

- The key borders provides the ability to draw only some borders of the blocks; the value of that key is a (comma-separated) list of elements covered by left, right, top and bottom; it's possible, in fact, in the list which is the value of the key borders, to add an entry of the form tikz={*list*} where *list* is a list of couples *key=value* of TikZ specifying the graphical characteristics of the lines that will be drawn (for an example, see p. 63).
- When the key tikz is used, the TikZ path corresponding of the rectangle which delimits the block is executed with TikZ[10] by using as options the value of that key tikz (which must be a list of keys allowed for a TikZ path).

  New 6.24 In fact, in the list of the keys provided by the user as value of tikz, it's possible to put a key offset. That key is not provided by TikZ but by nicematrix. It will narrow the rectangular frame corresponding to the block by a margin (horizontally and vertically) equal to the value (of that key offset). That new frame, a bit narrower, will be executed by TikZ with options which are the other keys in the list of keys provided as value to the key tikz of \Block.

  For examples, cf. p. 58.

There is also some technical keys:

- the key name provides a name to the rectangular TikZ node corresponding to the block; it's possible to use that name with TikZ in the \CodeAfter of the environment (cf. p. 35);
- the key respect-arraystretch prevents the setting of \arraystretch to 1 at the beginning of the block (which is the behaviour by default) ;
- By default, the rules are not drawn in the blocks (see the section about the rules: section 5 p. 9). However, if the key transparent is used, the rules are drawn. For an example, see section 18.1 on page 58. Caution: that key does not imply that the content of the block will be transparent!

[8] However, the rules are not drawn in the sub-blocks of the block, as always with nicematrix: the rules are not drawn in the blocks, except when they have the key transparent (cf. section 5 p. 9).

[9] This value is the initial value of the *rounded corners* of TikZ.

[10] TikZ should be loaded (by default, nicematrix only loads PGF) and, if it's not, an error will be raised.