L'extension nicematrix *

F. Pantigny
fpantigny@wanadoo.fr

15 mai 2024

Résumé

L'extension LaTeX nicematrix fournit de nouveaux environnements similaires aux environnements classiques `{tabular}`, `{array}` et `{matrix}` de array et amsmath mais avec des fonctionnalités plus étendues.

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

Produit	dimensions (cm)			Prix
	L	l	h	
petit	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
spécial	12	12	0.5	70

L'extension nicematrix est entièrement contenue dans le fichier `nicematrix.sty`. Ce fichier peut être placé dans le répertoire courant ou dans une arborescence `texmf`. Le mieux reste néanmoins d'installer nicematrix avec une distribution TeX comme MiKTeX, TeX Live ou MacTeX.

Remarque : Si vous utilisez un service LaTeX via Internet (ex. : Overleaf) vous pouvez télécharger le fichier `nicematrix.sty` dans le dossier de votre projet pour bénéficier de la dernière version de nicematrix. [1]

Cette extension peut être utilisée avec `xelatex`, `lualatex` et `pdflatex` mais aussi avec le cheminement classique `latex-dvips-ps2pdf` (ou Adobe Distiller). *Néanmoins, le fichier nicematrix-french.tex de la présente documentation ne peut être compilé qu'avec LuaLaTeX.*

Cette extension nécessite et charge les extensions l3keys2e, array, amsmath et pgfcore ainsi que le module shapes de PGF (l'extension tikz, qui est une surcouche de PGF, n'est *pas* chargée). L'utilisateur final n'a qu'à charger l'extension nicematrix avec l'instruction habituelle : `\usepackage{nicematrix}`.

L'idée de nicematrix est de créer des nœuds PGF derrière les cases et les positions des filets des tableaux créés par array et de les utiliser pour développer de nouvelles fonctionnalités. Comme toujours avec PGF, les coordonnées de ces nœuds sont écrites dans le fichier aux pour être utilisées à la compilation suivante. C'est pourquoi l'utilisation de nicematrix nécessite **plusieurs compilations successives**. [2]

La plupart des fonctionnalités de nicematrix sont accessibles sans avoir à utiliser explicitement PGF ou TikZ (ce dernier n'est d'ailleurs pas chargé par défaut).

Une commande `\NiceMatrixOptions` est fournie pour régler les options (la portée des options fixées par cette commande est le groupe TeX courant : elles sont semi-globales).

*Ce document correspond à la version 6.27x de nicematrix, en date du 2024/05/06.

1. La dernière version de `nicematrix.sty` peut être téléchargée sur le serveur SVN de TeXLive : `https:www.tug.org/svn/texlive/trunk/Master/texmf-dist/tex/latex/nicematrix/nicematrix.sty`

2. Si vous utilisez Overleaf, Overleaf effectue automatiquement un nombre de compilations suffisant.

1 Les environnements de cette extension

L'extension nicematrix définit les nouveaux environnements suivants :

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

Les environnements {NiceArray}, {NiceTabular} et {NiceTabular*} sont similaires aux environnements {array}, {tabular} et {tabular*} de l'extension array (qui est chargée par nicematrix).

Les environnements {pNiceArray}, {bNiceArray}, etc. n'ont pas d'équivalents dans array.

Les environnements {NiceMatrix}, {pNiceMatrix}, etc. sont similaires aux environnements correspondants de l'amsmath (qui est chargée par nicematrix) : {matrix}, {pmatrix}, etc.

L'environnement {NiceTabularX} est similaire à l'environnement {tabularx} de l'extension éponyme. [3]

On conseille d'utiliser prioritairement les environnements classiques et de n'utiliser les environnements de nicematrix que lorsqu'on utilise les fonctionnalités supplémentaires offertes par ces environnements (cela permet d'économiser la mémoire).

Tous les environnements de l'extension nicematrix acceptent, entre crochets, une liste optionnelle de paires de la forme *clé=valeur*. **Il doit n'y avoir aucun espace devant le crochet ouvrant ([) de cette liste d'options.**

2 L'espace vertical entre les rangées

Il est bien connu que certaines rangées des tableaux créés par défaut avec LaTeX sont trop proches l'une de l'autre. On en donne ci-dessous un exemple classique.

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

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

En s'inspirant de l'extension cellspace qui traite de ce problème, l'extension nicematrix propose deux clés cell-space-top-limit et cell-space-bottom-limit qui sont similaires aux deux paramètres \cellspacetoplimit et \cellspacebottomlimit proposés par cellspace.
Il existe aussi une clé cell-space-limits pour régler simultanément les deux paramètres.
La valeur initiale de ces paramètres est 0 pt pour que les environnements de nicematrix aient par défaut le même comportement que ceux de array et de l'amsmath mais une valeur de 1 pt serait un bon choix. On conseille de régler leurs valeurs avec la commande \NiceMatrixOptions. [4]

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

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

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

Il est également possible de changer ces paramètres pour certaines lignes seulement grâce à la commande \RowStyle (cf. p. 23).

3. Néanmoins, on peut aussi utiliser directement les colonnes X dans l'environnement {NiceTabular}, la largeur souhaitée pour le tableau étant spécifiée par la clé width : cf. p. 24.

4. On remarquera que ces paramètres s'appliquent aussi aux colonnes de type S de siunitx alors que cellspace n'est pas utilisable avec ces colonnes.

3 La clé baseline

L'extension nicematrix propose une option `baseline` pour la position verticale des tableaux. Cette option `baseline` prend comme valeur un entier qui indique le numéro de rangée dont la ligne de base servira de ligne de base pour le tableau.

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

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

L'option `baseline` peut aussi prendre les trois valeurs spéciales `t`, `c` et `b`. Ces trois lettres peuvent aussi être utilisées de manière absolue comme pour l'option des environnements `{tabular}` et `{array}` de array. La valeur initiale de `baseline` est `c`.

Dans l'exemple suivant, on utilise l'option `t` (synonyme de `baseline=t`) immédiatement après un `\item` de liste. On remarquera que la présence d'un `\hline` initial n'empêche pas l'alignement sur la ligne de base de la première rangée (avec `{tabular}` ou `{array}` de array, il faut utiliser `\firsthline`).

```
\begin{enumerate}
\item un 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. un item
2.

n	0	1	2	3	4	5
u_n	1	2	4	8	16	32

Il est également possible d'utiliser les outils de booktabs : `\toprule`, `\bottomrule`, `\midrule`, etc., à condition, bien entendu, d'avoir chargé booktabs.

```
\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.

n	0	1	2	3	4	5
u_n	1	2	4	8	16	32

On peut aussi utiliser la clé `baseline` pour aligner une matrice sur un filet horizontal (tracé par `\hline`). On doit pour cela donner la valeur `line-`*i* où *i* est le numéro de la rangée qui *suit* ce filet horizontal.

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

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

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

4 Les blocs

4.1 Cas général

Dans les environnements de nicematrix, on peut utiliser la commande `\Block` pour placer un élément au centre d'un rectangle de cases fusionnées. [5]
La commande `\Block` doit être utilisée dans la case supérieure gauche du bloc avec deux arguments obligatoires.

— Le premier argument est la taille de ce bloc avec la syntaxe i-j où i est le nombre de rangées et j le nombre de colonnes du bloc.
Si cet argument est laissé blanc, la valeur par défaut est `1-1`. Si le nombre de rangées n'est pas indiqué, ou bien est égal à `*`, le bloc s'étend jusqu'à la dernière rangée (idem pour les colonnes).

— Le deuxième argument est le contenu du bloc. On peut utiliser `\\` dans ce contenu pour avoir un contenu sur plusieurs lignes. Dans `{NiceTabular}`, `{NiceTabular*}` et `{NiceTabularX}`, le contenu est composé en mode texte tandis que, dans les autres environnements, il est composé en mode mathématique.

Voici un exemple d'utilisation de la commande `\Block` dans une matrice mathématique.

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

$$\left[\begin{array}{ccc:c} \multicolumn{3}{c:}{\multirow{3}{*}{A}} & 0 \\ & & & \vdots \\ & & & 0 \\ \hdashline 0 & \cdots & 0 & 0 \end{array}\right]$$

On peut souhaiter agrandir la taille du « A » placé dans le bloc de l'exemple précédent. Comme il est composé en mode mathématique, on ne peut pas directement utiliser une commande comme `\large`, `\Large` ou `\LARGE`. C'est pourquoi une option à mettre entre chevrons est proposée par `\Block` pour spécifier du code LaTeX qui sera inséré *avant* le début du mode mathématique. [6]

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

$$\left[\begin{array}{ccc:c} \multicolumn{3}{c:}{\multirow{3}{*}{A}} & 0 \\ & & & \vdots \\ & & & 0 \\ \hdashline 0 & \cdots & 0 & 0 \end{array}\right]$$

La commande `\Block` accepte en premier argument optionnel (entre crochets) une liste de couples *clé=valeur*.

Les premières clés sont des outils rapides pour contrôler l'apparence du bloc :

— la clé `fill` prend en argument une couleur et remplit le bloc avec cette couleur ;

— la clé `opacity` fixe l'opacité de la couleur de remplissage donnée par `fill` ;

— la clé `draw` prend en argument une couleur et trace le cadre avec cette couleur (la valeur par défaut de cette clé est la couleur courante des filets du tableau) ;

— la clé `color` prend en argument une couleur et l'applique au contenu et trace également le cadre avec cette couleur ;

5. Les espaces situés après une commande `\Block` sont supprimés.
6. Cet argument entre chevrons peut aussi être utilisé pour insérer une commande de fonte comme `\bfseries`, ce qui peut être utile dans le cas où la commande `\\` apparaît dans le contenu du bloc. On peut aussi y mettre la commmande `\rotate` fournie par nicematrix (cf. partie 14.5, p. 45).

— les clés `hlines`, `vlines` et `hvlines` tracent les filets correspondants dans le bloc [7] ;

— la clé `line-width` fixe la largeur utilisée pour tracer les filets (n'a d'intérêt que si `draw`, `hvlines`, `hlines` ou `vlines` est utilisée) ;

— la clé `rounded-corners` impose des coins arrondis (pour le cadre dessiné par `draw` et le fond dessiné par `fill`) avec un rayon égal à la valeur de cette clé (la valeur par défaut est 4 pt [8]).

Ces outils ne sont parfois pas suffisants pour contrôler l'apparence du bloc. Les clés suivantes sont plus puissantes, mais plus difficiles d'utilisation. Elles nécessitent également que TikZ soit chargé (par `\usepackage{tikz}`). Par défaut, nicematrix ne charge pas TikZ mais uniquement PGF, qui est une sous-couche de TikZ.

— La clé `borders` permet de ne tracer que certaines des bordures du bloc : cette clé prend comme valeur une liste d'éléments parmi les suivants : `left`, `right`, `top` et `bottom` ; on peut en fait, dans la liste qui est la valeur de la clé `borders` mettre une entrée de la forme `tikz={`*liste*`}` où *liste* est une liste de couples *clé=valeur* de TikZ spécifiant les caractéristiques graphiques des traits qui seront dessinés (pour un exemple, voir p. 62).

— Quand la clé `tikz` est utilisée, le chemin TikZ correspondant au rectangle délimitant le bloc est exécuté avec TikZ [9] en utilisant comme options la valeur de cette clé `tikz` (qui doit donc être une liste de clés TikZ applicables à un chemin de TikZ). Pour des exemples d'utilisation de cette clé `tikz`, voir p. 57.

  Nouveau 6.24 En fait, dans la liste des clés fournies à `tikz`, on peut mettre une clé `offset`. Cette clé n'est pas fournie par TikZ mais par nicematrix. Elle réduit le rectangle correspondant au bloc par une marge (horizontalement et verticalement) égale à la valeur (passée à `offset`). C'est ce rectangle réduit qui sera le chemin exécuté par TikZ avec comme options les autres clés passées à la clé `tikz`.

Enfin, il existe quelque clés techniques :

— la clé `name` donne un nom au nœud TikZ rectangulaire correspondant au bloc ; on peut utiliser ce nom avec TikZ dans le `\CodeAfter` (cf. p. 34) ;

— la clé `respect-arraystretch` évite la remise à 1 de `\arraystretch` en début de bloc (qui a lieu par défaut) ;

— Par défaut, les filets ne sont pas tracés dans les blocs (voir à ce sujet la partie sur les filets, section 5 p. 9). Néanmoins, si la clé `transparent` est utilisée, les filets seront tracés. [10]

  Pour un exemple, voir la section 18.1, page 57.

  Attention : cette clé n'implique pas du tout que le contenu du bloc sera transparent.

Il existe aussi des clés de positionnement horizontal et vertical du bloc qui sont décrites ci-dessous (cf. 4.5 p. 7).

On doit remarquer que, par défaut, les blocs ne créent pas d'espace. Il n'y a exception que pour les blocs mono-rangée et les blocs mono-colonne dans certaines conditions comme expliqué plus loin.

Dans l'exemple suivant, on a dû élargir à la main les colonnes 2 et 3 (avec la construction `w{c}{...}` de array).

```
\begin{NiceTabular}{cw{c}{2cm}w{c}{3cm}c}
rose      & tulipe & marguerite & dahlia \\
violette
& \Block[draw=red,fill=[RGB]{204,204,255},rounded-corners]{2-2}
        {\LARGE De très jolies fleurs}
```

7. Néanmoins, les filets ne sont pas tracés dans les sous-blocs du bloc, conformément à l'esprit de nicematrix : les filets ne sont pas tracés dans les blocs, sauf s'ils possèdent la clé `transparent` (cf. section 5 p. 9).

8. Cette valeur par défaut est la valeur initiale des *rounded corners* de TikZ.

9. TikZ doit être chargé préalablement (par défaut, nicematrix ne charge que PGF), faute de quoi, une erreur sera levée.

10. Par ailleurs, la commande `\TikzEveryCell` disponible dans le `\CodeAfter` et le `\CodeBefore`, ne s'applique aux blocs avec la clé `transparent`.