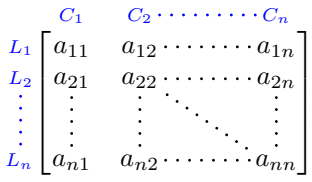
L'extension nicematrix *

F. Pantigny
fpantigny@wanadoo.fr

30 juin 2020

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.



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 ou TeXlive.

Cette extension peut être utilisée avec `xelatex`, `lualatex` et `pdflatex` mais aussi avec le cheminement classique `latex-dvips-ps2pdf` (ou Adobe Distiller).

Cette extension nécessite et charge les extensions l3keys2e, xparse, 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**.

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 4.4 of nicematrix, en date du 2020/06/30.

1 Les environnements de cette extension

L'extension nicematrix définit les nouveaux environnements suivants :

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

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

Il y a néanmoins quelques différences :

— Pour des raisons techniques, dans le préambule de ces environnements, l'utilisateur doit utiliser les lettres L, C et R [1] au lieu de l, c et r, y compris dans les commandes \multicolumn et dans les types de colonnes définis par \newcolumntype.

— Dans {NiceArray} (et ses variantes), les colonnes de type w (ex. : wc{1cm}) sont composées en mode mathématique alors que dans {array} de array, elles sont composées en mode texte.

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.

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 défault avec LaTeX sont trop proches l'une de l'autre. On en donne ci-dessous un exemple classique.

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

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

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. 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'amsmth mais une valeur de 1 pt serait un bon choix. On conseille de régler leurs valeurs avec la commande \NiceMatrixOptions.

```
\NiceMatrixOptions{cell-space-top-limit = 1pt,cell-space-bottom-limit = 1pt}

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

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

1. Les types de colonnes L, C et R sont définis localement à l'intérieur de {NiceTabular} ou {NiceArray} avec la commande \newcolumntype de array. Cette définition masque une éventuelle définition précédente.

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`[2]).

```
\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.

```
\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

4 Les blocs

Dans les environnements de nicematrix, on peut utiliser la commande `\Block` pour placer un élément au centre d'un rectangle de cases fusionnées. La commande `\Block` ne crée pas d'espace par elle-même. La commande `\Block` doit être utilisée dans la case supérieure gauche du bloc avec deux arguments. 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. Le deuxième argument, est, sans surprise, le contenu du bloc.
Dans `{NiceTabular}`, le contenu est composé en mode texte. Dans les autres environnements, il est composé en mode mathématique.

2. On peut aussi utiliser `\firsthline` dans les environnements de nicematrix.

```
\begin{NiceTabular}{CCCC}
rose      & tulipe & marguerite & dahlia \\
violette  & \Block{2-2}{\LARGE\color{blue} fleurs}  & & souci \\
pervenche & & & lys \\
arum      & iris & jacinthe & muguet
\end{NiceTabular}
```

rose	tulipe	marguerite	dahlia
violette	fleurs		souci
pervenche			lys
arum	iris	jacinthe	muguet

On remarquera que les centrage horizontal du contenu des blocs est correct même si un espacement des colonnes a été demandé par une instruction comme `!{\qquad}` dans le préambule (ce n'est pas le cas avec `\multicolumn`). Dans l'exemple suivant, l'en-tête « Premier groupe » est correctement centré.

```
\begin{NiceTabular}{C!{\qquad}CCC!{\qquad}CCC}
\toprule
& \Block{1-3}{Premier groupe} & & & \Block{1-3}{Deuxième groupe} \\
Rang & 1A & 1B & 1C & 2A & 2B & 2C \\
\midrule
 1 & 0.657 & 0.913 & 0.733 & 0.830 & 0.387 & 0.893\\
 2 & 0.343 & 0.537 & 0.655 & 0.690 & 0.471 & 0.333\\
 3 & 0.783 & 0.885 & 0.015 & 0.306 & 0.643 & 0.263\\
 4 & 0.161 & 0.708 & 0.386 & 0.257 & 0.074 & 0.336\\
\bottomrule
\end{NiceTabular}
```

	Premier groupe			Deuxième groupe		
Rang	1A	1B	1C	2A	2B	2C
1	0.657	0.913	0.733	0.830	0.387	0.893
2	0.343	0.537	0.655	0.690	0.471	0.333
3	0.783	0.885	0.015	0.306	0.643	0.263
4	0.161	0.708	0.386	0.257	0.074	0.336

On peut aussi utiliser la commande `\Block` dans des matrices mathématiques.

```
$\begin{bNiceArray}{CCC|C}[margin]
\Block{3-3}{A} & & & 0 \\
& \hspace*{1cm} & & \Vdots \\
& & & 0 \\
\hline
0 & \Cdots& 0 & 0
\end{bNiceArray}$
```

$$\left[\begin{array}{ccc|c} \Block{3-3}{A} & & & 0 \\ & \hspace*{1cm} & & \Vdots \\ & & & 0 \\ \hline 0 & \Cdots & 0 & 0 \end{array}\right]$$

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.

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

$$\left[\begin{array}{ccc|c} \Block{3-3}<\LARGE>{A} & & & 0 \\ & \hspace*{1cm} & & \Vdots \\ & & & 0 \\ \hline 0 & \Cdots & 0 & 0 \end{array}\right]$$

5 Les filets horizontaux et verticaux

Les techniques habituelles pour tracer des filets peuvent être utilisées dans les environnements de nicematrix, à l'exception de `\vline`.

5.1 L'épaisseur et la couleur des filets

Les environnements de nicematrix proposent une clé `rules/width` pour fixer la largeur (en fait l'épaisseur) des filets dans l'environnement. En fait, cette clé ne fait que fixer la valeur de `\arrayrulewidth`.

On sait que colortbl propose la commande `\arrayrulecolor` pour spécifier la couleur de ces filets.

Avec nicematrix, il est possible de spécifier une couleur même si colortbl n'est pas chargé. Par souci de compatibilité, la commande est nommée également `\arrayrulecolor`. Les environnements de nicematrix proposent également une clé `rules/color` qui permet de fixer cette couleur pour l'environnement en question.

```
\begin{NiceTabular}{|CCC|}[rules/color=[gray]{0.9},rules/width=1pt]
\hline
rose & tulipe & lys \\
arum & iris & violette \\
muguet & dahlia & souci \\
\hline
\end{NiceTabular}
```

rose	tulipe	lys
arum	iris	violette
muguet	dahlia	souci

Si on veut définir de nouveaux spécificateurs de colonnes pour des filets (par exemple plus épais ou bien d'une couleur spécifique), on aura peut-être intérêt à utiliser la commande `\OnlyMainNiceMatrix` décrite p. 23.

5.2 Une remarque concernant \cline

Les traits verticaux et horizontaux que l'on insère avec `\hline` et le spécificateur de colonne « | » de array rendent le tableau plus large ou plus long d'une quantité égale à la largeur du trait.

Pour des raisons historiques, il n'en est pas de même pour la commande `\cline`, comme on peut le voir avec l'exemple suivant.

```
\setlength{\arrayrulewidth}{2pt}
\begin{tabular}{cccc} \hline
A&B&C&D \\ \cline{2-2}
A&B&C&D \\ \hline
\end{tabular}
```

A	B	C	D
A	B	C	D

Dans les environnements de nicematrix, cette situation est corrigée (il est néanmoins possible de revenir au comportement par défaut de `\cline` avec la clé `standard-cline`).

```
\setlength{\arrayrulewidth}{2pt}
\begin{NiceTabular}{CCCC} \hline
A&B&C&D \\ \cline{2-2}
A&B&C&D \\ \hline
\end{NiceTabular}
```

A	B	C	D
A	B	C	D

5.3 Les clés hlines et vlines

La clé `hlines` demande un tracé de tous les filets horizontaux et la clé `vlines` demande un tracé de tous les filets verticaux. En fait, dans les environnements avec délimiteurs (comme `{pNiceMatrix}` ou `{bNiceArray}`), les filets extérieurs ne sont pas tracés (ce qui est le comportement certainement attendu).

```
$\begin{pNiceMatrix}[vlines,rules/width=0.2pt]
1 & 2 & 3 & 4 & 5 & 6 \\
1 & 2 & 3 & 4 & 5 & 6 \\
1 & 2 & 3 & 4 & 5 & 6
\end{pNiceMatrix}$
```

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 2 & 3 & 4 & 5 & 6 \end{pmatrix}$$

Il y a néanmoins une différence entre l'utilisation de l'option `vlines` et du spécificateur « | » dans le préambule de l'environnement : les filets tracés par `vlines` traversent les double-filets horizontaux tracés par `\hline\hline` (il n'y a pas besoin d'utiliser hhline).

```
$\begin{NiceMatrix}[vlines] \hline
a & b & c & d \\ \hline \hline
1 & 2 & 3 & 4 \\
1 & 2 & 3 & 4 \\ \hline
\end{NiceMatrix}$
```

a	b	c	d
1	2	3	4
1	2	3	4

Si vous utilisez booktabs (qui fournit `\toprule`, `\midrule`, `\bottomrule`, etc.) et que vous tenez absolument à mettre des filets verticaux (ce qui est contraire à l'esprit à booktabs), sachez que la clé `vlines` est compatible avec booktabs.

```
$\begin{NiceMatrix}[vlines] \toprule
a & b & c & d \\ \midrule
1 & 2 & 3 & 4 \\
1 & 2 & 3 & 4 \\ \bottomrule
\end{NiceMatrix}$
```

a	b	c	d
1	2	3	4
1	2	3	4

5.4 La clé hvlines

La clé `hvlines` demande le tracé de tous les filets horizontaux et verticaux *sauf dans les blocs.*[3]

```
\begin{NiceTabular}{CCCC}[hvlines,rules/color=blue,rules/width=1pt]
rose      & tulipe & marguerite & dahlia \\
violette  & \Block{2-2}{\LARGE\color{blue} fleurs}  & & souci \\
pervenche & & & lys \\
arum      & iris & jacinthe & muguet
\end{NiceTabular}
```

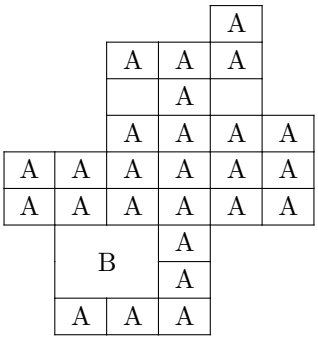
rose	tulipe	marguerite	dahlia
violette	fleurs		souci
pervenche			lys
arum	iris	jacinthe	muguet

5.5 La clé hvlines-except-corners

La clé `hvlines-except-corners` demande le tracé de tous les filets horizontaux et verticaux *sauf dans les blocs* et sauf dans les coins vides.

```
\begin{NiceTabular}{*{6}{C}}[hvlines-except-corners,cell-space-top-limit=3pt]
  &   &   &   & A \\
  &   & A & A & A \\
  &   &   & A \\
  &   & A & A & A & A \\
A & A & A & A & A & A \\
A & A & A & A & A & A \\
  & \Block{2-2}{B} & & A \\
  &   &   & A \\
  & A & A & A \\
\end{NiceTabular}
```

3. En fait, quand la clé `hvlines` (ou la clé `hvlines-except-corners` décrite juste après) est utilisée, les filets ne sont pas non plus tracés dans les blocs virtuels délimités par des cases reliées par des lignes pointillées (cf. p. 16).



Comme on le voit, un « coin vide » est constitué de la réunion de tous les rectangles vides partant de la case située exactement dans le coin.

```
\begin{NiceTabular}{*{6}{C}}%
  [hvlines-except-corners,cell-space-top-limit=3pt]
1\\
1&1\\
1&2&1\\
1&3&3&1\\
1&4&6&4&1\\
1&5&10&10&5&1
\end{NiceTabular}
```

1					
1	1				
1	2	1			
1	3	3	1		
1	4	6	4	1	
1	5	10	10	5	1

5.6 La commande \diagbox

La commande `\diagbox` (inspirée par l'extension diagbox) permet, quand elle est utilisée dans une case, de couper cette case selon une diagonale descendante. [4]

```
$\begin{NiceArray}{*{5}{C}}[hvlines]
\diagbox{x}{y}  & e & a & b & c \\
e & e & a & b & c \\
a & a & e & c & b \\
b & b & c & e & a \\
c & c & b & a & e
\end{NiceArray}$
```

x \ y	e	a	b	c
e	e	a	b	c
a	a	e	c	b
b	b	c	e	a
c	c	b	a	e

5.7 Filets en pointillés

Dans les environnements de nicematrix, il est possible d'utiliser la commande `\hdottedline` (fournie par nicematrix) qui est l'équivalent pour les pointillés des commandes `\hline` et `\hdashline` (cette dernière étant une commande de arydshln).

```
\begin{pNiceMatrix}
1 & 2 & 3 & 4 & 5 \\
\hdottedline
6 & 7 & 8 & 9 & 10 \\
11 & 12 & 13 & 14 & 15
\end{pNiceMatrix}
```

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ \hdotsfor{5} \\ 6 & 7 & 8 & 9 & 10 \\ 11 & 12 & 13 & 14 & 15 \end{pmatrix}$$

Dans les environnements avec un préambule explicite (comme `{NiceTabular}`, `{NiceArray}`, etc.), il est possible de dessiner un trait vertical en pointillés avec le spécificateur « `:` ».

```
\begin{pNiceArray}{CCCC:C}
1 & 2 & 3 & 4 & 5 \\
6 & 7 & 8 & 9 & 10 \\
11 & 12 & 13 & 14 & 15
\end{pNiceArray}
```

$$\begin{pmatrix} 1 & 2 & 3 & 4 & \vdots & 5 \\ 6 & 7 & 8 & 9 & \vdots & 10 \\ 11 & 12 & 13 & 14 & \vdots & 15 \end{pmatrix}$$

4. L'auteur de ce document considère que ce type de construction est un piètre choix graphique.

Il est possible de changer dans nicematrix la lettre utilisée pour indiquer dans le préambule un trait vertical en pointillés avec l'option `letter-for-dotted-lines` disponible dans `\NiceMatrixOptions`.

Remarque : Quand l'extension array (sur laquelle s'appuie nicematrix) est chargée, les traits verticaux et horizontaux que l'on insère rendent le tableau plus large ou plus long d'une quantité égale à la largeur du trait [5]. Avec nicematrix, les lignes en pointillés tracées par `\hdottedline` et « : » ont le même effet.

6 Les couleurs des rangées et des colonnes

L'extension colortbl permet de colorier des cases, des rangées ou des colonnes d'un tableau. Néanmoins, l'affichage du PDF résultant n'est pas toujours parfait, en particulier quand on utilise également des filets. Dans certains lecteurs de PDF, des filets verticaux semblent disparaître. De fines lignes blanches peuvent également apparaître.

L'extension nicematrix propose des outils similaires qui ne présentent pas ces inconvénients. Elle propose une clé `code-before` [6] pour du code qui sera exécuté *avant* le tracé du tableau. Dans ce `code-before`, de nouvelles commandes sont disponibles : `\cellcolor`, `\rectanglecolor`, `\rowcolor`, `\columncolor`, `\rowcolors` et `\chessboardcolors`.

Ces outils sont indépendants de colortbl. [7]

Toutes ces commandes acceptent un argument optionnel (entre crochets et en première position) qui est le modèle colorimétrique pour la spécification des couleurs.

— La commande `\cellcolor` tient son nom de la commande `\cellcolor` de colortbl.
Elle prend en arguments obligatoires une couleur et une liste de cases sous le format i-j où i est le numéro de ligne et j le numéro de colonne.

```
\begin{NiceTabular}{|C|C|C|}[code-before = \cellcolor{red!15}{3-1,2-2,1-3}]
\hline
a & b & c \\ \hline
e & f & g \\ \hline
h & i & j \\ \hline
\end{NiceTabular}
```

a	b	c
e	f	g
h	i	j

— La commande `\rectanglecolor` prend trois arguments obligatoires. Le premier est la couleur, les deux suivants fournissent la case en haut à gauche et la case en bas à droite du rectangle.

```
\begin{NiceTabular}{|C|C|C|}[code-before = \rectanglecolor{blue!15}{2-2}{3-3}]
\hline
a & b & c \\ \hline
e & f & g \\ \hline
h & i & j \\ \hline
\end{NiceTabular}
```

a	b	c
e	f	g
h	i	j

— La commande `\rowcolor` doit son nom à la commande `\rowcolor` de colortbl. Son premier argument obligatoire est la couleur et le deuxième est une liste de numéros de rangées ou bien d'intervalles de rangées sous la forme a-b (un intervalle de la forme a- représente toutes les rangées à partir de la rangée a).

5. En fait, cela est vrai pour `\hline` et « | » mais pas pour `\cline`.
6. Il existe aussi une clé `code-after` : cf. p. 17.
7. On peut donc colorier les filets, les cellules, les rangées, les colonnes, etc. sans charger colortbl. Le présent document ne charge pas colortbl.

```
$\begin{NiceArray}{LLL}[hvlines, code-before = \rowcolor{red!15}{1,3-5,8-}]
a_1 & b_1 & c_1 \\
a_2 & b_2 & c_2 \\
a_3 & b_3 & c_3 \\
a_4 & b_4 & c_4 \\
a_5 & b_5 & c_5 \\
a_6 & b_6 & c_6 \\
a_7 & b_7 & c_7 \\
a_8 & b_8 & c_8 \\
a_9 & b_9 & c_9 \\
a_{10} & b_{10} & c_{10} \\
\end{NiceArray}$
```

a_1	b_1	c_1
a_2	b_2	c_2
a_3	b_3	c_3
a_4	b_4	c_4
a_5	b_5	c_5
a_6	b_6	c_6
a_7	b_7	c_7
a_8	b_8	c_8
a_9	b_9	c_9
a_{10}	b_{10}	c_{10}

— La commande `\columncolor` doit son nom à la commande `\columncolor` de colortbl. Sa syntaxe est similaire à celle de `\rowcolor`.

— La commande `\rowcolors` (avec un *s*) doit son nom à la commande `\rowcolors` de xcolor [8]. Le *s* rappelle qu'il y a deux couleurs. Elle colorie alternativement les rangées avec les deux couleurs à partir de la rangée dont le numéro est donné en premier argument (obligatoire).

```
\begin{NiceTabular}{LR}[hlines,code-before = \rowcolors{1}{blue!10}{}]
Pierre & 12 \\
Jacques & 8 \\
Stéphanie & 18 \\
Amélie & 20 \\
Henri & 14 \\
Estelle & 15
\end{NiceTabular}
```

Pierre	12
Jacques	8
Stéphanie	18
Amélie	20
Henri	14
Estelle	15

— La commande `\chessboardcolors` prend en arguments obligatoires deux couleurs et colorie les cases en quinconces avec les deux couleurs.

```
$\begin{pNiceMatrix}[R,margin, code-before=\chessboardcolors{red!15}{blue!15}]
1 & -1 & 1 \\
-1 & 1 & -1 \\
1 & -1 & 1
\end{pNiceMatrix}$
```

$$\begin{pmatrix} 1 & -1 & 1 \\ -1 & 1 & -1 \\ 1 & -1 & 1 \end{pmatrix}$$

On a utilisé la clé `R` qui impose que toutes les colonnes soient alignées à droite (cf. p. 17).

On remarquera que ces commandes sont compatibles avec les commandes de booktabs (`\toprule`, `\midrule`, `\bottomrule`, etc).

```
\begin{NiceTabular}{LSSSS}%
[code-before = \rowcolor{red!15}{1-2} \rowcolors{3}{blue!15}{}]
\toprule
\Block{2-1}{Produit} \\
\Block{1-3}{dimensions (cm)} & & &
\Block{2-1}{\rotate Prix} \\
\cmidrule(rl){2-4}
 & L & l & h \\
\midrule
petit    & 3   & 5.5  & 1   & 30    \\
moyen    & 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    \\
\bottomrule
\end{NiceTabular}
```

Produit	dimensions (cm)			Prix
	L	l	h	
petit	3	5.5	1	30
moyen	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

On a utilisé le type de colonne `S` de siunitx.

8. La commande `\rowcolors` de xcolor est disponible quand xcolor est chargé avec l'option `table`.

7 La largeur des colonnes

Dans les environnements avec un préambule explicite (comme `{NiceTabular}`, `{NiceArray}`, etc.), il est possible de fixer la largeur d'une colonne avec les lettres classiques `w` et `W` de l'extension array. Dans `{NiceTabular}`, les cases des colonnes de ce type sont composées en mode texte mais, dans `{NiceArray}`, `{pNiceArray}`, etc., elles sont composées en mode mathématique (alors que dans `{array}` de array, elles sont composées en mode texte).

```
\begin{NiceTabular}{Wc{2cm}CC}[hvlines]
Paris  & New York & Madrid \\
Berlin & London   & Roma   \\
Rio    & Tokyo    & Oslo
\end{NiceTabular}
```

Paris	New York	Madrid
Berlin	London	Roma
Rio	Tokyo	Oslo

Dans les environnements de nicematrix, il est aussi possible de fixer la largeur *minimale* de toutes les colonnes de la matrice directement avec l'option `columns-width`.

```
$\begin{pNiceMatrix}[columns-width = 1cm]
1  & 12 & -123 \\
12 & 0  & 0    \\
4  & 1  & 2
\end{pNiceMatrix}$
```

$$\begin{pmatrix} 1 & 12 & -123 \\ 12 & 0 & 0 \\ 4 & 1 & 2 \end{pmatrix}$$

Notez que l'espace inséré entre deux colonnes (égal à 2 `\tabcolsep` dans `{NiceTabular}` et à 2 `\arraycolsep` dans les autres environnements) n'est pas supprimé (il est évidemment possible de le supprimer en mettant `\tabcolsep` ou `\arraycolsep` à 0 avant).

Il est possible de donner la valeur spéciale `auto` à l'option `columns-width` : toutes les colonnes du tableau auront alors une largeur égale à la largeur de la case la plus large du tableau. [9]

```
$\begin{pNiceMatrix}[columns-width = auto]
1  & 12 & -123 \\
12 & 0  & 0    \\
4  & 1  & 2
\end{pNiceMatrix}$
```

$$\begin{pmatrix} 1 & 12 & -123 \\ 12 & 0 & 0 \\ 4 & 1 & 2 \end{pmatrix}$$

Sans surprise, il est possible de fixer la largeur minimale de toutes les colonnes de toutes les matrices dans une certaine portion de document avec la commande `\NiceMatrixOptions`.

```
\NiceMatrixOptions{columns-width=10mm}
$\begin{pNiceMatrix}
a & b \\ c & d
\end{pNiceMatrix}
=
\begin{pNiceMatrix}
1   & 1245 \\ 345 & 2
\end{pNiceMatrix}$
```

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \begin{pmatrix} 1 & 1245 \\ 345 & 2 \end{pmatrix}$$

Mais il est aussi possible de fixer une zone dans laquelle toutes les matrices auront leurs colonnes de la même largeur, égale à la largeur de la case la plus large de toutes les matrices de la zone. Cette construction utilise l'environnement `{NiceMatrixBlock}` avec l'option `auto-columns-width` [10]. L'environnement `{NiceMatrixBlock}` n'a pas de rapport direct avec la commande `\Block` présentée précédemment dans ce document (cf. p. 3).

9. Le résultat est atteint dès la première compilation (mais Tikz écrivant des informations dans le fichier `.aux`, un message demandant une deuxième compilation apparaîtra).

10. Pour le moment, c'est le seul usage de l'environnement `{NiceMatrixBlock}` mais il pourrait y en avoir davantage dans le futur.

```
\begin{NiceMatrixBlock}[auto-columns-width]
$\begin{array}{c}
\begin{bNiceMatrix}
 9 & 17 \\ -2 & 5
 \end{bNiceMatrix} \\ \\
\begin{bNiceMatrix}
 1   & 1245345 \\  345 & 2
\end{bNiceMatrix}
\end{array}$
\end{NiceMatrixBlock}
```

$$\begin{bmatrix} 9 & 17 \\ -2 & 5 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 1245345 \\ 345 & 2 \end{bmatrix}$$

Plusieurs compilations peuvent être nécessaires pour obtenir le résultat désiré.

8 Les rangées et colonnes extérieures

Les environnements de nicematrix permettent de composer des rangées et des colonnes « extérieures » grâce aux options `first-row`, `last-row`, `first-col` et `last-col`.
Si elle est présente, la « première rangée » (extérieure) est numérotée par 0 (et non 1). Il en est de même pour la « première colonne ».

```
$\begin{pNiceMatrix}[first-row,last-row,first-col,last-col,nullify-dots]
       & C_1    & \Cdots &        & C_4    &        \\
L_1    & a_{11} & a_{12} & a_{13} & a_{14} & L_1    \\
\Vdots & a_{21} & a_{22} & a_{23} & a_{24} & \Vdots \\
       & a_{31} & a_{32} & a_{33} & a_{34} &        \\
L_4    & a_{41} & a_{42} & a_{43} & a_{44} & L_4    \\
       & C_1    & \Cdots &        & C_4    &
\end{pNiceMatrix}$
```

$$\begin{array}{c} \\ L_1 \\ \vdots \\ L_4 \\ \\ \end{array} \begin{array}{c} \begin{array}{cccc} C_1 & \cdots & & C_4 \end{array} \\ \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix} \\ \begin{array}{cccc} C_1 & \cdots & & C_4 \end{array} \end{array} \begin{array}{c} \\ L_1 \\ \vdots \\ L_4 \\ \\ \end{array}$$

Les lignes pointillées ont été tracées avec les outils présentés p. 12.

Il y a plusieurs remarques à formuler.

— Si on utilise un environnement avec préambule explicite (c'est-à-dire `{NiceArray}` ou l'une de ses variantes), on ne doit pas mettre dans ce préambule de spécification de colonne pour les éventuelles première et dernière colonne : ce sera automatiquement (et nécessairement) une colonne `R` pour la première colonne et une colonne `L` pour la dernière.

— On peut se demander comment nicematrix détermine le nombre de rangées et de colonnes nécessaires à la composition de la « dernière rangée » et de la « dernière colonne ».

  — Dans le cas d'un environnement avec préambule, comme `{NiceTabular}` ou `{pNiceArray}`, le nombre de colonnes se déduit évidemment du préambule.

  — Dans le cas où l'option `light-syntax` (cf. p. 19) est utilisée, nicematrix profite du fait que cette option nécessite de toutes manières le chargement complet du contenu de l'environnement (d'où l'impossibilité de mettre du verbatim dans ce cas-là) avant composition du tableau. L'analyse du contenu de l'environnement donne le nombre de rangées (mais pas le nombre de colonnes).

  — Dans les autres cas, nicematrix détermine le nombre de rangées et de colonnes à la première compilation et l'écrit dans le fichier `.aux` pour pouvoir l'utiliser à la compilation suivante. *Néanmoins, il est possible de donner le numéro de la dernière rangée et le numéro de la dernière colonne en arguments des options* `last-row` *et* `last-col`*, ce qui permettra d'accélérer le processus complet de compilation.* C'est ce que nous ferons dans la suite.

On peut contrôler l'apparence de ces rangées et colonnes avec les options `code-for-first-row`, `code-for-last-row`, `code-for-first-col` et `code-for-last-col`. Ces options sont des listes de tokens qui seront insérées au début de chaque case de la rangée ou de la colonne considérée.

```
\NiceMatrixOptions{code-for-first-row = \color{red},
                   code-for-first-col = \color{blue},
                   code-for-last-row = \color{green},
                   code-for-last-col = \color{magenta}}
$\begin{pNiceArray}{CC|CC}[first-row,last-row=6,first-col,last-col,nullify-dots]
       & C_1    & \Cdots &        & C_4    &        \\
L_1    & a_{11} & a_{12} & a_{13} & a_{14} & L_1    \\
\Vdots & a_{21} & a_{22} & a_{23} & a_{24} & \Vdots \\
\hline
       & a_{31} & a_{32} & a_{33} & a_{34} &        \\
L_4    & a_{41} & a_{42} & a_{43} & a_{44} & L_4    \\
       & C_1    & \Cdots &        & C_4    &
\end{pNiceArray}$
```

$$\begin{array}{c} \\ L_1 \\ \vdots \\ L_4 \end{array} \begin{array}{c} \begin{array}{cccc} C_1 & \cdots & & C_4 \end{array} \\ \left(\begin{array}{cc|cc} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ \hline a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{array}\right) \\ \begin{array}{cccc} C_1 & \cdots & & C_4 \end{array} \end{array} \begin{array}{c} \\ L_1 \\ \vdots \\ L_4 \end{array}$$

Remarques

— Comme on peut le voir dans l'exemple précédent, les filets horizontaux et verticaux ne s'étendent pas dans les rangées et colonnes extérieures.

Néanmoins, si on veut définir de nouveaux spécificateurs de colonnes pour des filets (par exemple plus épais), on aura sans doute intérêt à utiliser la commande `\OnlyMainNiceMatrix` décrite p. 23.

— Une spécification de couleur présente dans `code-for-first-row` s'applique à une ligne pointillée tracée dans cette « première rangée » (sauf si une valeur a été donnée à `xdots/color`). Idem pour les autres.

— Sans surprise, une éventuelle option `columns-width` (décrite p. 10) ne s'applique pas à la « première colonne » ni à la « dernière colonne ».

— Pour des raisons techniques, il n'est pas possible d'utiliser l'option de la commande `\\` après la « première rangée » ou avant la « dernière rangée » (le placement des délimiteurs serait erroné).

9 Les lignes en pointillés continues

À l'intérieur des environnements de l'extension nicematrix, de nouvelles commandes sont définies : `\Ldots`, `\Cdots`, `\Vdots`, `\Ddots`, et `\Iddots`. Ces commandes sont conçues pour être utilisées à la place de `\dots`, `\cdots`, `\vdots`, `\ddots` et `\iddots`. [11]

Chacune de ces commandes doit être utilisée seule dans la case du tableau et elle trace une ligne en pointillés entre les premières cases non vides [12] situées de part et d'autre de la case courante. Bien entendu, pour `\Ldots` et `\Cdots`, c'est une ligne horizontale ; pour `\Vdots`, c'est une ligne verticale et pour `\Ddots` et `\Iddots`, ce sont des lignes diagonales. On peut changer la couleur d'une ligne avec

11. La commande `\iddots`, définie dans nicematrix, est une variante de `\ddots` avec les points allant vers le haut. Si mathdots est chargée, la version de mathdots est utilisée. Elle correspond à la commande `\adots` de unicode-math.

12. La définition précise de ce qui est considéré comme une « case vide » est donnée plus loin (cf. p. 24).

l'option `color`. [13]

```
\begin{bNiceMatrix}
a_1      & \Cdots &        & & a_1     \\
\Vdots   & a_2    & \Cdots & & a_2     \\
         & \Vdots & \Ddots[color=red] \\
\\
a_1      & a_2    &        & & a_n
\end{bNiceMatrix}
```

$$\begin{bmatrix} a_1 & \cdots & & a_1 \\ \vdots & & & \\ & a_2 & \cdots & a_2 \\ \vdots & \vdots & & \\ a_1 & a_2 & & a_n \end{bmatrix}$$

Pour représenter la matrice nulle, on peut choisir d'utiliser le codage suivant :

```
\begin{bNiceMatrix}
0      & \Cdots & 0      \\
\Vdots &        & \Vdots \\
0      & \Cdots & 0
\end{bNiceMatrix}
```

$$\begin{bmatrix} 0 & \cdots & 0 \\ \vdots & & \vdots \\ 0 & \cdots & 0 \end{bmatrix}$$

On peut néanmoins souhaiter une matrice plus grande. Habituellement, dans un tel cas, les utilisateurs de LaTeX ajoutent une nouvelle ligne et une nouvelle colonne. Il est possible d'utiliser la même méthode avec nicematrix :

```
\begin{bNiceMatrix}
0      & \Cdots & \Cdots & 0      \\
\Vdots &        &        & \Vdots \\
\Vdots &        &        & \Vdots \\
0      & \Cdots & \Cdots & 0
\end{bNiceMatrix}
```

$$\begin{bmatrix} 0 & \cdots & \cdots & 0 \\ \vdots & & & \vdots \\ \vdots & & & \vdots \\ 0 & \cdots & \cdots & 0 \end{bmatrix}$$

Dans la première colonne de cet exemple, il y a deux instructions `\Vdots` mais une seule ligne en pointillés sera tracée (il n'y a pas d'objets qui se superposent dans le fichier PDF résultant [14]).

En fait, dans cet exemple, il aurait été possible de tracer la même matrice plus rapidement avec le codage suivant :

```
\begin{bNiceMatrix}
0       & \Cdots &        & 0      \\
\Vdots  &        &        &        \\
        &        &        & \Vdots \\
0       &        & \Cdots & 0
\end{bNiceMatrix}
```

$$\begin{bmatrix} 0 & \cdots & \cdots & 0 \\ \vdots & & & \vdots \\ \vdots & & & \vdots \\ 0 & \cdots & \cdots & 0 \end{bmatrix}$$

Il y a aussi d'autres moyens de changer la taille d'une matrice. On pourrait vouloir utiliser l'argument optionnel de la commande `\\` pour l'espacement vertical et la commande `\hspace*` dans une case pour l'espacement horizontal. [15]
Toutefois, une commande `\hspace*` pourrait interférer dans la construction des lignes en pointillés. C'est pourquoi l'extension nicematrix fournit une commande `\Hspace` qui est une variante de `\hspace` transparente pour la construction des lignes en pointillés de nicematrix.

```
\begin{bNiceMatrix}
0      & \Cdots & \Hspace*{1cm} & 0      \\
\Vdots &        &               & \Vdots \\[1cm]
0      & \Cdots &               & 0
\end{bNiceMatrix}
```

$$\begin{bmatrix} 0 & \cdots & & 0 \\ \vdots & & & \vdots \\ & & & \\ \vdots & & & \vdots \\ 0 & \cdots & & 0 \end{bmatrix}$$

13. Il est aussi possible de changer la couleur de toutes ces lignes pointillées avec l'option `xdots/color` (*xdots* pour rappeler que cela s'applique à `\Cdots`, `\Ldots`, `Vdots`, etc.) : cf. p. 15).

14. Et il n'est pas possible de tracer une ligne `\Ldots` et une ligne `\Cdots` entre les mêmes cases.

15. Dans nicematrix, il faut utiliser `\hspace*` et non `\hspace` car nicematrix utilise array. Remarquons aussi que l'on peut également régler la largeur des colonnes en utilisant l'environnement `{NiceArray}` (ou une de ses variantes) avec une colonne de type `w` ou `W` : cf. p. 10

9.1 L'option nullify-dots

Considérons la matrice suivante qui a été composée classiquement avec l'environnement `{pmatrix}` de amsmath.

```
$A = \begin{pmatrix}
h & i & j & k & l & m \\
x &   &   &   &   & x
\end{pmatrix}$
```

$$A = \begin{pmatrix} h & i & j & k & l & m \\ x & & & & & x \end{pmatrix}$$

Si nous ajoutons des instructions `\ldots` dans la seconde rangée, la géométrie de la matrice est modifiée.

```
$B = \begin{pmatrix}
h & i & j & k & l & m \\
x & \ldots & \ldots  & \ldots & \ldots & x
\end{pmatrix}$
```

$$B = \begin{pmatrix} h & i & j & k & l & m \\ x & \ldots & \ldots & \ldots & \ldots & x \end{pmatrix}$$

Par défaut, avec nicematrix, si nous remplaçons `{pmatrix}` par `{pNiceMatrix}` et `\ldots` par `\Ldots`, la géométrie de la matrice n'est pas changée.

```
$C = \begin{pNiceMatrix}
h & i & j & k & l & m \\
x & \Ldots & \Ldots & \Ldots & \Ldots & x
\end{pNiceMatrix}$
```

$$C = \begin{pmatrix} h & i & j & k & l & m \\ x & \ldots & \ldots & \ldots & \ldots & x \end{pmatrix}$$

On pourrait toutefois préférer la géométrie de la première matrice A et vouloir avoir la même géométrie avec une ligne en pointillés continue dans la seconde rangée. C'est possible en utilisant l'option `nullify-dots` (et une seule instruction `\Ldots` suffit).

```
$D = \begin{pNiceMatrix}[nullify-dots]
h & i & j & k & l & m \\
x & \Ldots & & & & x
\end{pNiceMatrix}$
```

$$D = \begin{pmatrix} h & i & j & k & l & m \\ x & \ldots & \ldots & \ldots & \ldots & x \end{pmatrix}$$

L'option `nullify-dots` « smashe » les instructions `\Ldots` (et ses variantes) horizontalement mais aussi verticalement.

9.2 La commande \Hdotsfor

Certaines personnes utilisent habituellement la commande `\hdotsfor` de l'extension amsmath pour tracer des lignes en pointillés horizontales dans une matrice. Dans les environnements de nicematrix, il convient d'utiliser `\Hdotsfor` à la place pour avoir les lignes en pointillés similaires à toutes celles tracées par l'extension nicematrix.

Comme avec les autres commandes de nicematrix (comme `\Cdots`, `\Ldots`, `\Vdots`, etc.), la ligne en pointillés tracée par `\Hdotsfor` s'étend jusqu'au contenu des cases de part et d'autre.

```
$\begin{pNiceMatrix}
1 & 2 & 3 & 4 & 5 \\
1 & \Hdotsfor{3} & 5 \\
1 & 2 & 3 & 4 & 5 \\
1 & 2 & 3 & 4 & 5
\end{pNiceMatrix}$
```

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 1 & \ldots & \ldots & \ldots & 5 \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \end{pmatrix}$$

Néanmoins, si ces cases sont vides, la ligne en pointillés s'étend seulement dans les cases spécifiées par l'argument de `\Hdotsfor` (par conception).

```
$\begin{pNiceMatrix}
1 & 2 & 3 & 4 & 5 \\
  & \Hdotsfor{3} \\
1 & 2 & 3 & 4 & 5 \\
1 & 2 & 3 & 4 & 5
\end{pNiceMatrix}$
```

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ & \ldots & \ldots & \ldots & \\ 1 & 2 & 3 & 4 & 5 \\ 1 & 2 & 3 & 4 & 5 \end{pmatrix}$$

Remarque : Contrairement à la commande `\hdotsfor` de amsmath, la commande `\Hdotsfor` est utilisable lorsque l'extension colortbl est chargée (mais vous risquez d'avoir des problèmes si vous utilisez `\rowcolor` sur la même rangée que `\Hdotsfor`).

9.3 Comment créer les lignes en pointillés de manière transparente

L'extension nicematrix fournit une option appelée `transparent` qui permet d'utiliser du code existant de manière transparente dans les environnements de l'amsmath : `{matrix}`, `{pmatrix}`, etc. En fait, cette option est un alias pour la conjonction de deux options : `renew-dots` et `renew-matrix`. [16]

— L'option `renew-dots`
Avec cette option, les commandes `\ldots`, `\cdots`, `\vdots`, `\ddots`, `\iddots` [11] et `\hdotsfor` sont redéfinies dans les environnements de nicematrix et agissent alors comme `\Ldots`, `\Cdots`, `\Vdots`, `\Ddots`, `\Iddots` et `\Hdotsfor` ; la commande `\dots` (points de suspension « automatiques » de `amsmath`) est aussi redéfinie et se comporte comme `\Ldots`.

— L'option `renew-matrix`
Avec cette option, l'environnement `{matrix}` est redéfini et se comporte comme `{NiceMatrix}` et il en est de même pour les cinq variantes.

Par conséquent, avec l'option `transparent`, un code classique donne directement le résultat fourni par nicematrix.

```
\NiceMatrixOptions{transparent}
\begin{pmatrix}
1      & \cdots & \cdots & 1      \\
0      & \ddots &        & \vdots \\
\vdots & \ddots & \ddots & \vdots \\
0      & \cdots & 0      & 1
\end{pmatrix}
```

$$\begin{pmatrix} 1 & \cdots & \cdots & 1 \\ 0 & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 \end{pmatrix}$$

9.4 Les labels des lignes en pointillés

Les commandes `\Ldots`, `\Cdots`, `\Vdots`, `\Ddots`, `\Iddots` et `\Hdotsfor` (ainsi que la commande `\line` dans le `code-after` décrite p. 17) peuvent en fait prendre deux arguments optionnels spécifiés par les caractères `_` et `^` pour des labels situés au-dessous et au-dessus de la ligne. Les arguments sont composés en mode mathématique avec `\scriptstyle`.

```
$\begin{bNiceMatrix}
1 & \hspace*{1cm}             & 0 \\[8mm]
  & \Ddots^{n \text{ times}} &   \\
0 &                           & 1
\end{bNiceMatrix}$
```

$$\begin{bmatrix} 1 & & 0 \\ & \overset{n \text{ times}}{\iddots} & \\ 0 & & 1 \end{bmatrix}$$

9.5 Personnalisation des lignes en pointillés

Les lignes pointillées tracées par `\Ldots`, `\Cdots`, `\Vdots`, `\Ddots`, `\Iddots` et `\Hdotsfor` (ainsi que par la commande `\line` dans le `code-after` décrite p. 17) peuvent être paramétrées par trois options (que l'on met entre crochets après la commande) :

— `color` ;

— `shorten` ;

— `line-style`.

Ces options peuvent aussi être fixées avec `\NiceMatrixOptions` ou bien au niveau d'un environnement mais elles doivent alors être préfixées par `xdots`, ce qui fait que leurs noms deviennent :

— `xdots/color` ;

— `xdots/shorten` ;

— `xdots/line-style`.

16. Comme toutes les autres options, les options `renew-dots`, `renew-matrix` et `transparent` peuvent être fixées avec la commande `\NiceMatrixOptions`, mais elles peuvent aussi être passées en option du `\usepackage` (ce sont les trois seules).

Pour la clarté, dans la suite, on utilisera ces noms-là.

L'option xdots/color

L'option `xdots/color` indique bien entendu la couleur de la ligne tracée. On remarquera néanmoins que les lignes tracées dans les rangées et colonnes extérieures (décrites plus loin) bénificient d'un régime spécial : cf. p. 11.

L'option xdots/shorten

L'option `xdots/shorten` indique la marge qui est laissée aux deux extrémités de la ligne. Le nom s'inspire des options « `shorten >` » et « `shorten <` » de Tikz, mais il faut remarquer que nicematrix ne propose que `xdots/shorten`. La valeur initiale de ce paramètre est de 0.3 em (il est conseillé d'utiliser une unité de mesure dépendante de la fonte courante).

L'option xdots/line-style

Il faut savoir que, par défaut, les lignes de Tikz tracées avec le paramètre `dotted` sont composées de points carrés et non pas ronds.[17]

```
\tikz \draw [dotted] (0,0) -- (5,0) ;
```

Voulant proposer des lignes avec des points ronds dans le style de celui de `\ldots` (au moins celui des fontes *Computer Modern*), l'extension nicematrix contient en interne son propre système de ligne en pointillés (qui, au passage, n'utilise que pgf et non tikz). Ce style est appelé le style `standard`. Cette valeur est la valeur initiale du paramètre `xdots/line-style`.

Néanmoins (quand Tikz est chargé), on peut utiliser pour `xdots/line-style` n'importe quel style proposé par Tikz, c'est-à-dire n'importe quelle suite d'options Tikz applicables à un chemin (à l'exception de « `color` », « `shorten >` » et « `shorten <` »).

Voici par exemple une matrice tridiagonale avec le style `loosely dotted` :

```
$\begin{pNiceMatrix}[nullify-dots,xdots/line-style=loosely dotted]
a      & b      & 0      &        & \Cdots & 0      \\
b      & a      & b      & \Ddots &        & \Vdots \\
0      & b      & a      & \Ddots &        &        \\
       & \Ddots & \Ddots & \Ddots &        & 0      \\
\Vdots &        &        &        &        & b      \\
0      & \Cdots &        & 0      & b      & a
\end{pNiceMatrix}$
```

$$\begin{pmatrix} a & b & 0 & \cdots & \cdots & 0 \\ b & a & b & \ddots & & \vdots \\ 0 & b & a & \ddots & \ddots & \\ \vdots & \ddots & \ddots & \ddots & & 0 \\ \vdots & & & & & b \\ 0 & \cdots & \cdots & 0 & b & a \end{pmatrix}$$

9.6 Les lignes pointillées et la clé hvlines

On a dit (cf. p. 6) que la clé `hvlines` trace tous les filets horizontaux et verticaux, exceptés dans les blocs. En fait, avec cette clé, les filets ne sont pas non plus tracés dans les blocs virtuels délimités par des cases reliées par des lignes pointillées.

17. La raison de départ est que le format PDF comporte un système de description de lignes en tiretés, qui, puisqu'il est incorporé dans le PDF, est affiché très rapidement par les lecteurs de PDF. Il est facile à partir de ce type de ligne de créer des lignes de points carrés alors qu'une ligne de points ronds doit être construite explicitement point par point.

```
\NiceMatrixOptions{nullify-dots}
$\begin{pNiceMatrix}[rules/color=gray,hvlines,margin]
0      & \Cdots &  &   &        & 0      \\
1      & \Cdots &  &   & 1      & 2      \\
0      & \Ddots &  &   & \Vdots & \Vdots \\
\Vdots & \Ddots &  &   &        &        \\
       &        &  &   &        &        \\
0      & \Cdots &  & 0 &  1     & 2      \\
\end{pNiceMatrix}$
```

$$\left(\begin{array}{cccccc|c} 0 & \cdots & \cdots & \cdots & \cdots & 0 & \\ \hline 1 & \cdots & \cdots & \cdots & 1 & 2 & \\ 0 & \ddots & & & \vdots & \vdots & \\ \vdots & \ddots & \ddots & & \vdots & \vdots & \\ \vdots & & \ddots & \ddots & \vdots & \vdots & \\ 0 & \cdots & \cdots & 0 & 1 & 2 & \end{array}\right)$$

10 Le code-after

L'option `code-after` peut être utilisée pour indiquer du code qui sera exécuté après la construction de la matrice. [18]
Une commande spéciale, nommée `\line` est disponible pour tracer directement des lignes en pointillés entre les cases. Elle prend deux arguments correspondant aux deux cases à relier, chacun de la forme *i*-*j* où *i* est le numéro de ligne et *j* est le numéro de colonne. Elle peut par exemple être utilisée pour tracer une ligne entre deux cases adjacentes comme dans l'exemple suivant.

```
\NiceMatrixOptions{xdots/shorten = 0.6 em}
\begin{pNiceMatrix}[code-after=\line{2-2}{3-3}]
I       & 0      & \Cdots  &0     \\
0       & I      & \Ddots  &\Vdots\\
\Vdots  &\Ddots  & I       &0     \\
0       &\Cdots  & 0       &I
\end{pNiceMatrix}
```

$$\begin{pmatrix} I & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & I & 0 \\ 0 & \cdots & 0 & I \end{pmatrix}$$

Pour améliorer la lisibilité du code, une syntaxe alternative est proposée : on peut spécifier les instructions du `code-after` à la fin de l'environnement, après le mot-clé `\CodeAfter`. Pour un exemple, voir page 28.

11 Autres fonctionnalités

11.1 Utilisation du type de colonne S de siunitx

Si l'extension siunitx est chargée (avant ou après nicematrix), il est possible d'utiliser les colonnes de type `S` de siunitx dans les environnements de nicematrix. L'implémentation n'utilise explicitement aucune macro privée de siunitx.

```
$\begin{pNiceArray}{SCWc{1cm}C}[nullify-dots,first-row]
{C_1} & \Cdots &  & C_n \\
2.3  & 0 & \Cdots & 0 \\
12.4 & \Vdots & & \Vdots \\
1.45 \\
7.2  & 0 & \Cdots & 0
\end{pNiceArray}$
```

$$\begin{array}{c} C_1 \cdots\cdots\cdots\cdots C_n \\ \left(\begin{array}{cccc} 2.3 & 0 & \cdots\cdots & 0 \\ 12.4 & \vdots & & \vdots \\ 1.45 & \vdots & & \vdots \\ 7.2 & 0 & \cdots\cdots & 0 \end{array}\right) \end{array}$$

En revanche, les colonnes `d` de l'extension dcolumn ne sont pas prises en charge par nicematrix.

11.2 Option d'alignement dans {NiceMatrix}

Les environnements sans préambule (`{NiceMatrix}`, `{pNiceMatrix}`, `{bNiceMatrix}`, etc.) proposent les options `l` et `r` (possédant `L` et `R` comme alias) qui imposent des colonnes alignées à gauche ou à droite.

18. Il existe aussi une clé `code-before` décrite p. 8.

```
$\begin{bNiceMatrix}[R]
\cos x & - \sin x \\
\sin x & \cos x
\end{bNiceMatrix}$
```

$$\begin{bmatrix} \cos x & -\sin x \\ \sin x & \cos x \end{bmatrix}$$

Il existe aussi l'option `S` qui impose que toutes les colonnes soient des colonnes `S` de siunitx (si cette extension est chargée). [19]

11.3 La commande \rotate

Utilisée au début d'une case, la commande `\rotate` (fournie par nicematrix) compose le contenu après une rotation de 90° dans le sens direct.
Dans l'exemple suivant, on l'utilise dans le `code-for-first-row`.

```
\NiceMatrixOptions%
 {code-for-first-row = \scriptstyle \rotate \text{image de },
  code-for-last-col = \scriptstyle }
$A = \begin{pNiceMatrix}[first-row,last-col=4]
e_1 & e_2 & e_3       \\
1   & 2   & 3   & e_1 \\
4   & 5   & 6   & e_2 \\
7   & 8   & 9   & e_3
\end{pNiceMatrix}$
```

$$A = \begin{array}{c} \begin{array}{ccc} \text{image de } e_1 & \text{image de } e_2 & \text{image de } e_3 \end{array} \\ \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \begin{array}{c} e_1 \\ e_2 \\ e_3 \end{array} \end{array}$$

Si la commande `\rotate` est utilisée dans la « dernière rangée » (extérieure à la matrice), les éléments qui subissent cette rotation sont alignés vers le haut.

```
\NiceMatrixOptions%
 {code-for-last-row = \scriptstyle \rotate ,
  code-for-last-col = \scriptstyle }
$A = \begin{pNiceMatrix}[last-row,last-col=4]
1   & 2   & 3   & e_1 \\
4   & 5   & 6   & e_2 \\
7   & 8   & 9   & e_3 \\
\text{image de } e_1 & e_2 & e_3
\end{pNiceMatrix}$
```

$$A = \begin{array}{c} \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \begin{array}{c} e_1 \\ e_2 \\ e_3 \end{array} \\ \begin{array}{ccc} \text{image de } e_1 & e_2 & e_3 \end{array} \end{array}$$

11.4 L'option small

Avec l'option `small`, les environnements de l'extension nicematrix sont composés d'une manière proche de ce que propose l'environnement `{smallmatrix}` de l'amsmath (et les environnements `{psmallmatrix}`, `{bsmallmatrix}`, etc. de mathtools).

```
$\begin{bNiceArray}{CCCC|C}[small,
                            last-col,
                            code-for-last-col = \scriptscriptstyle,
                            columns-width = 3mm ]
1 & -2 & 3 & 4 & 5 \\
0 & 3  & 2 & 1 & 2 & L_2 \gets 2 L_1 - L_2 \\
0 & 1  & 1 & 2 & 3 & L_3 \gets L_1 + L_3
\end{bNiceArray}$
```

$$\left[\begin{array}{cccc|c} 1 & -2 & 3 & 4 & 5 \\ 0 & 3 & 2 & 1 & 2 \\ 0 & 1 & 1 & 2 & 3 \end{array}\right] \begin{array}{l} \\ L_2 \leftarrow 2L_1 - L_2 \\ L_3 \leftarrow L_1 + L_3 \end{array}$$

On remarquera néanmoins que l'environnement `{NiceMatrix}` avec l'option `small` ne prétend pas être composé exactement comme l'environnement `{smallmatrix}`. C'est que les environnements de

19. Cela reprend une partie des fonctionnalités proposées par les environnements `{pmatrix*}`, `{bmatrix*}`, etc. de mathtools.

nicematrix sont tous fondés sur `{array}` (de array) alors que ce n'est pas le cas de `{smallmatrix}` (fondé directement sur un `\halign` de TeX).

En fait, l'option `small` correspond aux réglages suivants :

— les composantes du tableau sont composées en `\scriptstyle` ;

— `\arraystretch` est fixé à 0.47 ;

— `\arraycolsep` est fixé à 1.45 pt ;

— les caractéristiques des lignes en pointillés sont également modifiées.

11.5 Les compteurs iRow et jCol

Dans les cases du tableau, il est possible d'utiliser les compteurs LaTeX `iRow` et `jCol` qui représentent le numéro de la rangée courante et le numéro de la colonne courante [20]. Bien entendu, l'utilisateur ne doit pas modifier les valeurs de ces compteurs qui sont utilisés en interne par nicematrix.
Dans le `code-before` (cf. p. 8) et dans le `code-after` (cf. p. 17), `iRow` représente le nombre total de rangées (hors éventuelles rangées extérieures) et `jCol` le nombre total de colonnes (hors potentielles colonnes extérieures).

```
$\begin{pNiceMatrix}%
    [first-row,
     first-col,
     code-for-first-row = \mathbf{\alph{jCol}} ,
     code-for-first-col = \mathbf{\arabic{iRow}} ]
&   &    &    &   \\
& 1 & 2  & 3  & 4 \\
& 5 & 6  & 7  & 8 \\
& 9 & 10 & 11 & 12
\end{pNiceMatrix}$
```

$$\begin{array}{c} \\ \mathbf{1} \\ \mathbf{2} \\ \mathbf{3} \end{array} \begin{array}{c} \begin{array}{cccc} \mathbf{a} & \mathbf{b} & \mathbf{c} & \mathbf{d} \end{array} \\ \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{pmatrix} \end{array}$$

Si des compteurs LaTeX nommés `iRow` ou `jCol` sont créés dans le document par d'autres extensions que nicematrix (ou tout simplement par l'utilisateur final), ces compteurs sont masqués dans les environnements de nicematrix.

L'extension nicematrix propose aussi des commandes pour composer automatiquement des matrices à partir d'un motif général. Ces commandes sont nommées `\AutoNiceMatrix`, `\pAutoNiceMatrix`, `\bAutoNiceMatrix`, `\vAutoNiceMatrix`, `\VAutoNiceMatrix` et `\BAutoNiceMatrix`.
Chacune de ces commandes prend deux arguments obligatoires : le premier est la taille de la matrice, sous la forme *n*-*p*, où *n* est le nombre de rangées et *p* est le nombre de colonnes et le deuxième est le motif (c'est-à-dire simplement des tokens qui seront insérés dans chaque case de la matrice, exceptées celles des éventuelles rangées et colonnes extérieures).

```
$C = \pAutoNiceMatrix{3-3}{C_{\arabic{iRow},\arabic{jCol}}}$
```

$$C = \begin{pmatrix} C_{1,1} & C_{1,2} & C_{1,3} \\ C_{2,1} & C_{2,2} & C_{2,3} \\ C_{3,1} & C_{3,2} & C_{3,3} \end{pmatrix}$$

11.6 L'option light-syntax

L'option `light-syntax` (inspirée de l'extension spalign) permet d'alléger la saisie des matrices, ainsi que leur lisibilité dans le source TeX. Lorsque cette option est activée, on doit utiliser le point-virgule comme marqueur de fin de rangée et séparer les colonnes par des espaces ou des tabulations. On remarquera toutefois que, comme souvent dans le monde TeX, les espaces après les séquences de contrôle ne sont pas comptées et que les éléments entre accolades sont considérés comme un tout.

20. On rappelle que le numéro de la « première rangée » (si elle existe) est 0 et que le numéro de la « première colonne » (si elle existe) est 0 également.

```
$\begin{bNiceMatrix}[light-syntax,first-row,first-col]
{} a             b                 ;
a  2\cos a       {\cos a + \cos b} ;
b \cos a+\cos b  { 2 \cos b }
\end{bNiceMatrix}$
```

$$\begin{array}{c} \\ a \\ b \end{array}\begin{array}{c} \begin{array}{cc} a & b \end{array} \\ \begin{bmatrix} 2\cos a & \cos a+\cos b \\ \cos a+\cos b & 2\cos b \end{bmatrix} \end{array}$$

On peut changer le caractère utilisé pour indiquer les fins de rangées avec l'option `end-of-row`. Comme dit précédemment, la valeur initiale de ce paramètre est un point-virgule.

Lorsque l'option `light-syntax` est utilisée, il n'est pas possible de mettre d'éléments en verbatim (avec par exemple la commande `\verb`) dans les cases du tableau. [21]

11.7 L'environnement {NiceArrayWithDelims}

En fait, l'environnement `{pNiceArray}` et ses variantes sont fondés sur un environnement plus général, appelé `{NiceArrayWithDelims}`. Les deux premiers arguments obligatoires de cet environnement sont les délimiteurs gauche et droit qui seront utilisés dans la construction de la matrice. Il est possible d'utiliser `{NiceArrayWithDelims}` si on a besoin de délimiteurs atypiques ou asymétriques.

```
$\begin{NiceArrayWithDelims}
   {\downarrow}{\uparrow}{CCC}[margin]
1 & 2 & 3 \\
4 & 5 & 6 \\
7 & 8 & 9
\end{NiceArrayWithDelims}$
```

$$\left\downarrow \begin{array}{ccc} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{array} \right\uparrow$$

12 Utilisation de Tikz avec nicematrix

12.1 Les nœuds correspondant aux contenus des cases

L'extension nicematrix crée un nœud PGF-Tikz pour chaque case (non vide) du tableau considéré. Ces nœuds sont utilisés, entre autres, pour tracer les lignes en pointillés entre les cases du tableau.

Tous les nœuds du document doivent avoir des noms deux à deux distincts et le nom de ces nœuds doit donc faire intervenir le numéro de l'environnement courant. Les environnements créés par nicematrix sont en effet numérotés par un compteur global interne.

Si l'environnement concerné a le numéro n, alors le nœud de la rangée i et de la colonne j a pour nom `nm-`n`-`i`-`j.

La commande `\NiceMatrixLastEnv` donne le numéro du dernier de ces environnements (pour LaTeX, il s'agit d'une commande — complètement développable — et non d'un compteur).

Il est néanmoins recommandé de passer plutôt par la clé `name`. Celle-ci permet de donner un nom à l'environnement. Une fois l'environnement nommé, les nœuds sont accessibles à travers les noms « *nom-i-j* » où *nom* est le nom donné au tableau et i et j les numéros de rangée et de colonne de la case considérée. On peut les utiliser avec PGF mais l'utilisateur final préférera sans doute utiliser Tikz (qui est une sur-couche de PGF). Il faut néanmoins se souvenir que nicematrix ne charge pas Tikz par défaut.

```
$\begin{pNiceMatrix}[name=ma-matrice]
1 & 2 & 3 \\
4 & 5 & 6 \\
7 & 8 & 9
\end{pNiceMatrix}$
\tikz[remember picture,overlay]
     \draw (ma-matrice-2-2) circle (2mm) ;
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & \textcircled{5} & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Ne pas oublier les options `remember picture` et `overlay`.

21. La raison en est que lorsque l'option `light-syntax` est utilisée, le contenu complet de l'environnement est chargé comme un argument de commande TeX. L'environnement ne se comporte plus comme un « vrai » environnement de LaTeX qui se contente d'insérer des commandes avant et après.

Dans le `code-after`, et si Tikz est chargé, les choses sont plus simples. On peut (et on doit) désigner les nœuds sous la forme *i-j* : il n'y a pas besoin de préciser l'environnement qui est évidemment l'environnement courant.

```
$\begin{pNiceMatrix}
1 & 2 & 3 \\
4 & 5 & 6 \\
7 & 8 & 9
\CodeAfter
\tikz \draw (2-2) circle (2mm) ;
\end{pNiceMatrix}$
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & \textcircled{5} & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Dans l'exemple suivant, nous avons surligné tous nœuds de la matrice (on explique plus loin la technique utilisée : cf. p. 28).

$$\begin{pmatrix} a & a+b & a+b+c \\ a & a & a+b \\ a & a & a \end{pmatrix}$$

12.2 Les « nœuds moyens » et les « nœuds larges »

En fait, l'extension nicematrix peut créer deux séries de nœuds supplémentaires (*extra nodes* en anglais) : les « nœuds moyens » (*medium nodes* en anglais) et les « nœuds larges » (*large nodes* en anglais). Les premiers sont créés avec l'option `create-medium-nodes` et les seconds avec l'option `create-large-nodes`. [22]

Ces nœuds ne sont pas utilisés par défaut par nicematrix.

Les noms des « nœuds moyens » s'obtiennent en ajoutant le suffixe « `-medium` » au nom des nœuds normaux. Dans l'exemple suivant, on a surligné tous les « nœuds moyens ». Nous considérons que cet exemple se suffit à lui-même comme définition de ces nœuds.

$$\begin{pmatrix} a & a+b & a+b+c \\ a & a & a+b \\ a & a & a \end{pmatrix}$$

Les noms des « nœuds larges » s'obtiennent en ajoutant le suffixe « `-large` » au nom des nœuds normaux. Dans l'exemple suivant, on a surligné tous les « nœuds larges ». Nous considérons que cet exemple se suffit à lui-même comme définition de ces nœuds. [23]

$$\begin{pmatrix} a & a+b & a+b+c \\ a & a & a+b \\ a & a & a \end{pmatrix}$$

Les « nœuds larges » de la première colonne et de la dernière colonne peuvent apparaître trop petits pour certains usages. C'est pourquoi il est possible d'utiliser les options `left-margin` et `right-margin` pour ajouter de l'espace des deux côtés du tableau et aussi de l'espace dans les « nœuds larges » de la première colonne et de la dernière colonne. Dans l'exemple suivant, nous avons utilisé les options `left-margin` et `right-margin`. [24]

$$\begin{pmatrix} a & a+b & a+b+c \\ a & a & a+b \\ a & a & a \end{pmatrix}$$

22. Il existe aussi l'option `create-extra-nodes` qui est un alias pour la conjonction de `create-medium-nodes` et `create-large-nodes`.

23. Il n'y a pas de « nœuds larges » créés dans les rangées et colonnes extérieures (pour ces rangées et colonnes, voir p. 11).

24. Les options `left-margin` et `right-margin` prennent des dimensions comme valeurs mais, si aucune valeur n'est donnée, c'est la valeur par défaut qui est utilisée et elle est égale à `\arraycolsep` (par défaut, 5 pt). Il existe aussi une option `margin` pour fixer à la fois `left-margin` et `right-margin`.

Il est aussi possible d'ajouter de l'espace sur les côtés du tableau avec les options `extra-left-margin` et `extra-right-margin`. Ces marges ne sont pas incorporées dans les « nœuds larges ». Dans l'exemple suivant, nous avons utilisé `extra-left-margin` et `extra-right-margin` avec la valeur 3 pt.

$$\begin{pmatrix} a & a+b & a+b+c \\ a & a & a+b \\ a & a & a \end{pmatrix}$$

Attention : Ces nœuds sont reconstruits à partir des contenus des cases et ne correspondent donc pas nécessairement aux cases délimitées par des filets.

Voici un tableau qui a été composé de la manière suivante :

```
\large
\begin{NiceTabular}{wl{2cm}LL}[hvlines]
fraise & amande & abricot \\
prune & pêche & poire  \\[1ex]
noix & noisette & brugnon
\end{NiceTabular}
```

fraise	amande	abricot
prune	pêche	poire
noix	noisette	brugnon

Ci-contre, on a colorié toutes les cases de ce tableau avec `\chessboardcolors`.

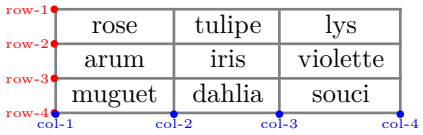
fraise	amande	abricot
prune	pêche	poire
noix	noisette	brugnon

Voici maintenant tous les « nœuds larges » de ce tableau (sans utilisation de `margin` ni de `extra-margin`).

fraise	amande	abricot
prune	pêche	poire
noix	noisette	brugnon

12.3 Les nœuds « row » et « col »

L'extension nicematrix crée un nœud PGF-Tikz indiquant la position potentielle de chaque filet horizontal (avec les noms `row-`i) et de chaque filet vertical (avec les noms `col-`j), comme décrit sur la figure ci-dessous. Ces nœuds sont accessibles dans le `code-before` et dans le `code-after`.



Si on utilise Tikz (on rappelle que nicematrix ne charge pas Tikz par défaut), on peut donc accéder (dans le `code-before` et le `code-after`) à l'intersection du filet horizontal i et du filet vertical j avec la syntaxe `(row-`i`-|col-`j`)`.

```
\[\begin{NiceMatrix}[
  code-before =
    {
      \tikz \draw [fill = red!15]
         (row-7-|col-4) -- (row-8-|col-4) -- (row-8-|col-5) --
         (row-9-|col-5) -- (row-9-|col-6) |- cycle ;
    }
]
1 \\
1 & 1 \\
```

```
1 & 2 &  1 \\
1 & 3 &  3 &  1 \\
1 & 4 &  6 &  4 &  1 \\
1 & 5 & 10 & 10 &  5 &  1 \\
1 & 6 & 15 & 20 & 15 &  6 &  1 \\
1 & 7 & 21 & 35 & 35 & 21 &  7 & 1 \\
1 & 8 & 28 & 56 & 70 & 56 & 28 & 8 & 1
\end{NiceMatrix}\]
```

$$
\begin{array}{ccccccccc}
1 & & & & & & & & \\
1 & 1 & & & & & & & \\
1 & 2 & 1 & & & & & & \\
1 & 3 & 3 & 1 & & & & & \\
1 & 4 & 6 & 4 & 1 & & & & \\
1 & 5 & 10 & 10 & 5 & 1 & & & \\
1 & 6 & 15 & 20 & 15 & 6 & 1 & & \\
1 & 7 & 21 & 35 & 35 & 21 & 7 & 1 & \\
1 & 8 & 28 & 56 & 70 & 56 & 28 & 8 & 1
\end{array}
$$

13 Remarques techniques

Première remarque : l'extension nicematrix doit être chargée après l'extension underscore. Si ce n'est pas le cas, une erreur est levée.

13.1 Pour définir de nouveaux types de colonnes

L'extension nicematrix fournit la commande `\OnlyMainNiceMatrix` qui est destinée à être utilisée dans des définitions de nouveaux types de colonnes. Son argument n'est exécuté que si on se place dans la partie principale du tableau, c'est-à-dire que l'on n'est pas dans l'une des éventuelles rangées extérieures.
Par exemple, si on souhaite définir un type de colonne `?` pour tracer un trait fort (noir) d'épaisseur 1 pt, on pourra écrire [25] :

```
\newcolumntype{?}{!{\OnlyMainNiceMatrix{\vrule width 1 pt}}}
```

Le trait fort correspondant ne s'étendra pas dans les rangées extérieures :

```
$\begin{pNiceArray}{CC?CC}[first-row,last-row=3]
C_1 & C_2 & C_3 & C_4 \\
a & b & c & d \\
e & f & g & h \\
C_1 & C_2 & C_3 & C_4
\end{pNiceArray}$
```

$$
\begin{array}{cccc}
C_1 & C_2 & C_3 & C_4 \\
\left(\begin{array}{cc|cc}
a & b & c & d \\
e & f & g & h
\end{array}\right) & & & \\
C_1 & C_2 & C_3 & C_4
\end{array}
$$

Le spécificateur `?` ainsi créé est aussi utilisable dans les environnements `{tabular}` et `{array}` (de array) et, dans ce cas, `\OnlyMainNiceMatrix` est sans effet.

13.2 Lignes diagonales

Par défaut, toutes les lignes diagonales [26] d'un même tableau sont « parallélisées ». Cela signifie que la première diagonale est tracée et que, ensuite, les autres lignes sont tracées parallèlement à la première (par rotation autour de l'extrémité la plus à gauche de la ligne). C'est pourquoi la position des instructions `\Ddots` dans un tableau peut avoir un effet marqué sur le résultat final.

Dans les exemples suivants, la première instruction `\Ddots` est marquée en couleur :

25. La commande `\vrule` est une commande de TeX (et non de LaTeX).
26. On parle des lignes créées par `\Ddots` et non des lignes créées par une commande `\line` dans le `code-after`.

Exemple avec parallélisation (comportement par défaut) :

```
$A = \begin{pNiceMatrix}
1      & \Cdots &        & 1      \\
a+b    & \Ddots &        & \Vdots \\
\Vdots & \Ddots &        &        \\
a+b    & \Cdots & a+b    & 1
\end{pNiceMatrix}$
```

$$A = \begin{pmatrix} 1 & \cdots & \cdots & 1 \\ a+b & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ a+b & \cdots & a+b & 1 \end{pmatrix}$$

```
$A = \begin{pNiceMatrix}
1      & \Cdots &        & 1      \\
a+b    &        &        & \Vdots \\
\Vdots & \Ddots & \Ddots &        \\
a+b    & \Cdots & a+b    & 1
\end{pNiceMatrix}$
```

$$A = \begin{pmatrix} 1 & \cdots & \cdots & 1 \\ a+b & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ a+b & \cdots & a+b & 1 \end{pmatrix}$$

Il est possible de désactiver la parallélisation avec l'option `parallelize-diags` mise à `false` :

Le même exemple sans parallélisation :

$$A = \begin{pmatrix} 1 & \cdots & \cdots & 1 \\ a+b & \ddots & & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ a+b & \cdots & a+b & 1 \end{pmatrix}$$

13.3 Les cases « vides »

Une instruction comme `\Ldots`, `\Cdots`, etc. essaye de déterminer la première case vide de part et d'autre de la case considérée. Néanmoins, une case vide n'est pas nécessairement sans contenu dans le codage TeX (c'est-à-dire sans aucun token entre les deux esperluettes `&`). En effet, une case dont le contenu est `\hspace*{1cm}` peut être considérée comme vide.

Pour nicematrix, les règles précises sont les suivantes :

— Une case implicite est vide. Par exemple, dans la matrice suivante

```
\begin{pmatrix}
a & b \\
c
\end{pmatrix}
```

la dernière case (deuxième rangée et deuxième colonne) est vide.

— Chaque case avec un rendu par TeX de largeur nulle est vide.

— Une case avec une commande `\Hspace` (ou `\Hspace*`) est vide. Cette commande `\Hspace` est une commande définie par l'extension nicematrix avec la même signification que `\hspace` excepté que la case où cette commande est utilisée est considérée comme vide. Cette commande peut être utilisée pour fixer la largeur des colonnes sans interférer avec le tracé des lignes en pointillés par nicematrix.

13.4 L'option exterior-arraycolsep

L'environnement `{array}` insère un espace horizontal égal à `\arraycolsep` avant et après chaque colonne. En particulier, il y a un espace égal à `\arraycolsep` avant et après le tableau. Cette caractéristique de l'environnement `{array}` n'était probablement pas une bonne idée [27]. L'environnement

27. Dans la documentation de `{amsmath}`, on peut lire : *The extra space of* `\arraycolsep` *that array adds on each side is a waste so we remove it [in* `{matrix}`*] (perhaps we should instead remove it from array in general, but that's a harder task).*

{matrix} et ses variantes ({pmatrix}, {vmatrix}, etc.) de amsmath préfèrent supprimer ces espaces avec des instructions explicites \hskip -\arraycolsep[28]. L'extension nicematrix fait de même dans *tous* ses environnements y compris l'environnement {NiceArray}. Néanmoins, si l'utilisateur souhaite que l'environnement {NiceArray} se comporte par défaut comme l'environnement {array} de array (par exemple pour faciliter l'adaptation d'un document existant), il peut contrôler ce comportement avec l'option exterior-arraycolsep accessible via la commande \NiceMatrixOptions. Avec cette option, des espaces extérieurs de longueur \arraycolsep seront insérés dans les environnements {NiceArray} (les autres environnements de l'extension nicematrix ne sont pas affectés).

13.5 Incompatibilités

L'extension nicematrix n'est pas compatible with threeparttable.

L'extension nicematrix n'est pas parfaitement compatible avec l'extension arydshln (parce que cette extension redéfinit de nombreuses commandes internes de array).

14 Exemples

14.1 Lignes en pointillés

Une matrice de permutation.
À titre d'exemple, on a augmenté la valeur du paramètre xdots/shorten.

```
$\begin{pNiceMatrix}[xdots/shorten=6em]
0       & 1 & 0 &        & \Cdots &   0    \\
\Vdots  &   &   & \Ddots &        & \Vdots \\
        &   &   & \Ddots &        &        \\
        &   &   & \Ddots &        &   0    \\
0       & 0 &   &        &        &   1    \\
1       & 0 &   & \Cdots &        &   0
\end{pNiceMatrix}$
```

$$\begin{pmatrix} 0 & 1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \iddots & \vdots \\ \vdots & & \ddots & \iddots & 0 \\ \vdots & & & \iddots & \vdots \\ 0 & 0 & & \ddots & 1 \\ 1 & 0 & \cdots & \cdots & 0 \end{pmatrix}$$

Un exemple avec \Iddots. On a augmenté encore davantage la valeur de xdots/shorten.

```
$\begin{pNiceMatrix}[xdots/shorten = 0.9em]
1       & \Cdots  &         & 1      \\
\Vdots  &         &         & 0      \\
        & \Iddots & \Iddots & \Vdots \\
1       & 0       & \Cdots  & 0
\end{pNiceMatrix}$
```

$$\begin{pmatrix} 1 & \cdots & 1 \\ \vdots & \iddots & 0 \\ \vdots & \iddots & \vdots \\ 1 & 0 & \cdots & 0 \end{pmatrix}$$

28. Et non en insérant @{} de part et d'autre du préambule, ce qui fait que la longueur des \hline n'est pas modifiée et elle peut paraître trop longue, surtout avec des crochets.

Un exemple avec `\multicolumn` :

```
\begin{BNiceMatrix}[nullify-dots]
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10\\
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10\\
\Cdots &  & \multicolumn{6}{C}{10 \text{ autres lignes}} & \Cdots \\
1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10
\end{BNiceMatrix}
```

$$\begin{Bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ \cdots\cdots & \multicolumn{6}{c}{10 \text{ autres lignes}} & \cdots\cdots \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{Bmatrix}$$

Un exemple avec `\Hdotsfor` :

```
\begin{pNiceMatrix}[nullify-dots]
0 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 0 \\
\Vdots  & \Hdotsfor{4} & \Vdots \\
 & \Hdotsfor{4} & \\
 & \Hdotsfor{4} & \\
 & \Hdotsfor{4} & \\
0 & 1 & 1 & 1 & 1 & 0
\end{pNiceMatrix}
```

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 \\ \vdots & \cdots\cdots & & & & \vdots \\ \vdots & \cdots\cdots & & & & \vdots \\ \vdots & \cdots\cdots & & & & \vdots \\ \vdots & \cdots\cdots & & & & \vdots \\ 0 & 1 & 1 & 1 & 1 & 0 \end{pmatrix}$$

Un exemple pour le résultant de deux polynômes :

```
\setlength{\extrarowheight}{1mm}
\begin{vNiceArray}{CCCC:CCC}[columns-width=6mm]
a_0   &      &&       &b_0    &      &     \\
a_1   &\Ddots&&       &b_1    &\Ddots&     \\
\Vdots&\Ddots&&       &\Vdots &\Ddots&b_0  \\
a_p   &      &&a_0    &       &      &b_1   \\
      &\Ddots&&a_1    &b_q    &      &\Vdots\\
      &      &&\Vdots &       &\Ddots&      \\
      &      &&a_p    &       &      &b_q
\end{vNiceArray}
```

$$\begin{vmatrix} a_0 & & & \vdots & b_0 & & \\ a_1 & \ddots & & \vdots & b_1 & \ddots & \\ \vdots & & \ddots & \vdots & & & b_0 \\ a_p & & a_0 & \vdots & b_q & & b_1 \\ & \ddots & & \vdots & & \ddots & \vdots \\ & & a_1 & \vdots & & & b_q \\ & & \vdots & \vdots & & & \\ & & a_p & \vdots & & & b_q \end{vmatrix}$$

Un exemple avec un système linéaire :

```
$\begin{pNiceArray}{*6C|C}[nullify-dots,last-col,code-for-last-col=\scriptstyle]
1      & 1 & 1 &\Cdots &   & 1      & 0      & \\
0      & 1 & 0 &\Cdots &   & 0      &        & L_2 \gets L_2-L_1 \\
0      & 0 & 1 &\Ddots &   & \Vdots &        & L_3 \gets L_3-L_1 \\
       &   &   &\Ddots &   &        & \Vdots & \Vdots \\
\Vdots &   &   &\Ddots &   & 0      & \\
0      &   &   &\Cdots & 0 & 1      & 0      & L_n \gets L_n-L_1
\end{pNiceArray}$
```

$$\left(\begin{array}{cccccc|c} 1 & 1 & 1 & \cdots & & 1 & 0 \\ 0 & 1 & 0 & \cdots & & 0 & \vdots \\ 0 & 0 & 1 & \ddots & & \vdots & \vdots \\ \vdots & & & \ddots & & 0 & \vdots \\ 0 & \cdots & & \cdots & 0 & 1 & 0 \end{array}\right) \begin{array}{l} \\ \scriptstyle L_2 \gets L_2-L_1 \\ \scriptstyle L_3 \gets L_3-L_1 \\ \vdots \\ \scriptstyle L_n \gets L_n-L_1 \end{array}$$

14.2 Des lignes pointillées qui ne sont plus pointillées

L'option `line-style` permet de changer le style des lignes tracées par `\Ldots`, `\Cdots`, etc. On peut de ce fait tracer des lignes qui ne sont plus pointillées.

```
\NiceMatrixOptions
  {nullify-dots,code-for-first-col = \color{blue},code-for-first-col=\color{blue}}
$\begin{pNiceMatrix}[first-row,first-col]
       &   & \Ldots[line-style={solid,<->},shorten=0pt]^{n \text{ columns}} \\
       & 1 & 1 & 1 & \Ldots & 1 \\
       & 1 & 1 & 1 &   & 1 \\
\Vdots[line-style={solid,<->}]_{n \text{ rows}}
       & 1 & 1 & 1 &   & 1 \\
       & 1 & 1 & 1 &   & 1 \\
       & 1 & 1 & 1 & \Ldots & 1
\end{pNiceMatrix}$
```

$$\overset{\longleftarrow n \text{ columns} \longrightarrow}{\begin{pmatrix} 1 & 1 & 1 & \ldots & 1 \\ 1 & 1 & 1 & & 1 \\ 1 & 1 & 1 & & 1 \\ 1 & 1 & 1 & & 1 \\ 1 & 1 & 1 & \ldots & 1 \end{pmatrix}} \quad (n \text{ rows})$$

14.3 Largeur des colonnes

Dans l'exemple suivant, nous utilisons `{NiceMatrixBlock}` avec l'option `auto-columns-width` parce que nous voulons la même largeur (automatique) pour toutes les colonnes.

```
\begin{NiceMatrixBlock}[auto-columns-width]
\NiceMatrixOptions
 { last-col,code-for-last-col = \color{blue}\scriptstyle,light-syntax}
\setlength{\extrarowheight}{1mm}
$\begin{pNiceArray}{CCCC:C}
  1  1  1   1   1 ;
  2  4  8  16   9 ;
  3  9 27  81  36 ;
  4 16 64 256 100
\end{pNiceArray}$
\medskip
$\begin{pNiceArray}{CCCC:C}
  1  1  1   1  1 ;
  0  2  6  14  7        { L_2 \gets -2 L_1 + L_2 } ;
  0  6 24  78 33        { L_3 \gets -3 L_1 + L_3 } ;
  0 12 60 252 96        { L_4 \gets -4 L_1 + L_4 }
\end{pNiceArray}$
...
\end{NiceMatrixBlock}
```

$$\left(\begin{array}{cccc:c} 1 & 1 & 1 & 1 & 1 \\ 2 & 4 & 8 & 16 & 9 \\ 3 & 9 & 27 & 81 & 36 \\ 4 & 16 & 64 & 256 & 100 \end{array}\right)$$

$$\left(\begin{array}{cccc:c} 1 & 1 & 1 & 1 & 1 \\ 0 & 2 & 6 & 14 & 7 \\ 0 & 6 & 24 & 78 & 33 \\ 0 & 12 & 60 & 252 & 96 \end{array}\right) \begin{array}{l} \\ L_2 \leftarrow -2L_1 + L_2 \\ L_3 \leftarrow -3L_1 + L_3 \\ L_4 \leftarrow -4L_1 + L_4 \end{array}$$

$$\left(\begin{array}{cccc:c} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 3 & 7 & \frac{7}{2} \\ 0 & 3 & 12 & 39 & \frac{33}{2} \\ 0 & 1 & 5 & 21 & 8 \end{array}\right) \begin{array}{l} \\ L_2 \leftarrow \frac{1}{2} L_2 \\ L_3 \leftarrow \frac{1}{2} L_3 \\ L_4 \leftarrow \frac{1}{12} L_4 \end{array}$$

$$\left(\begin{array}{cccc:c} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 3 & 7 & \frac{7}{2} \\ 0 & 0 & 3 & 18 & 6 \\ 0 & 0 & -2 & -14 & -\frac{9}{2} \end{array}\right) \begin{array}{l} \\ \\ L_3 \leftarrow -3L_2 + L_3 \\ L_4 \leftarrow L_2 - L_4 \end{array}$$

$$\left(\begin{array}{cccc:c} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 3 & 7 & \frac{7}{2} \\ 0 & 0 & 1 & 6 & 2 \\ 0 & 0 & -2 & -14 & -\frac{9}{2} \end{array}\right) \begin{array}{l} \\ \\ L_3 \leftarrow \frac{1}{3} L_3 \\ \\ \end{array}$$

$$\left(\begin{array}{cccc:c} 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 3 & 7 & \frac{7}{2} \\ 0 & 0 & 1 & 6 & 2 \\ 0 & 0 & 0 & -2 & -\frac{1}{2} \end{array}\right) \begin{array}{l} \\ \\ \\ L_4 \leftarrow L_3 + L_4 \end{array}$$

14.4 Comment surligner les cases

Les exemples suivants nécessitent d'avoir chargé Tikz (nicematrix ne charge que PGF) ainsi que la bibliothèque Tikz `fit`, ce qui peut se faire avec les deux instructions suivantes dans le préambule du document :

```
\usepackage{tikz}
\usetikzlibrary{fit}
```

Pour mettre en évidence une case, il est possible de « dessiner » l'un des nœuds (le « nœud normal », le « nœud moyen » ou le « nœud large »). Dans l'exemple suivant, on utilise les « nœuds larges » de la diagonale de la matrice (avec la clé de Tikz « `name suffix` », il est facile d'utiliser les « nœuds larges »).
Nous redessinons les nœuds avec de nouveaux nœuds en utilisant la bibliothèque fit de Tikz. Comme nous voulons recréer des nœuds identiques aux premiers, nous devons fixer `inner sep = 0pt` (si on ne fait pas cela, les nouveaux nœuds seront plus grands que les nœuds d'origine créés par nicematrix).

```
$\begin{pNiceArray}{>{\strut}CCCC}[create-large-nodes,margin,extra-margin = 2pt]
  a_{11} & a_{12} & a_{13} & a_{14} \\
  a_{21} & a_{22} & a_{23} & a_{24} \\
```

```
  a_{31} & a_{32} & a_{33} & a_{34} \\
  a_{41} & a_{42} & a_{43} & a_{44}
\CodeAfter
  \begin{tikzpicture}[name suffix = -large,
                      every node/.style = {draw,inner sep = 0 pt}]
     \node [fit = (1-1)] {} ;
     \node [fit = (2-2)] {} ;
     \node [fit = (3-3)] {} ;
     \node [fit = (4-4)] {} ;
  \end{tikzpicture}
\end{pNiceArray}$
```

$$\begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{pmatrix}$$

On remarquera que les traits que l'on vient de tracer sont dessinés *après* la matrice sans modifier la position des composantes de celle-ci. En revanche, les traits tracés par `\hline`, le spécificateur « `|` » ou les options `hlines`, `vlines` et `hvlines` « écartent » les composantes de la matrice. [29]

Il est possible de colorier une rangée avec `\rowcolor` dans le `code-before` (ou avec `\rowcolor` de colortbl dans la première case de la rangée). Les possibilités de réglages sont néanmoins limitées. C'est pourquoi nous présentons ici une autre méthode pour surligner une rangée de la matrice. Nous créons un nœud Tikz rectangulaire qui englobe les nœuds de la deuxième rangée en utilisant les outils de la bibliothèque Tikz fit. Ce nœud est rempli après la construction de la matrice. Pour que l'on puisse voir le texte *sous* le nœud, nous devons utiliser la transparence avec le `blend mode` égal à `multiply`.

```
\tikzset{highlight/.style={rectangle,
                           fill=red!15,
                           blend mode = multiply,
                           rounded corners = 0.5 mm,
                           inner sep=1pt,
                           fit=#1}}

$\begin{bNiceMatrix}[code-after = {\tikz \node [highlight = (2-1) (2-3)] {} ;}]
0 & \Cdots & 0 \\
1 & \Cdots & 1 \\
0 & \Cdots & 0
\end{bNiceMatrix}$
```

$$\begin{bmatrix} 0 & \cdots & 0 \\ 1 & \cdots & 1 \\ 0 & \cdots & 0 \end{bmatrix}$$

Ce code échoue avec `latex-dvips-ps2pdf` parce que Tikz pour `dvips`, pour le moment, ne prend pas en charge les *blend modes*. Néanmoins, le code suivant, dans le préambule du document LaTeX, devrait activer les *blend modes* pour ce mode de compilation.

```
\ExplSyntaxOn
\makeatletter
\tl_set:Nn \l_tmpa_tl {pgfsys-dvips.def}
\tl_if_eq:NNT \l_tmpa_tl \pgfsysdriver
  {\cs_set:Npn\pgfsys@blend@mode#1{\special{ps:~/\tl_upper_case:n #1~.setblendmode}}}
\makeatother
\ExplSyntaxOff
```

29. Pour la commande `\cline`, voir la remarque p. 5.

On rappelle que dans le cas d'un ensemble de cases fusionnées (avec la commande `\Block`), un nœud Tikz est créé pour l'ensemble des cases avec pour nom *i*`-`*j*`-block` où *i* et *j* sont les numéros de ligne et de colonne de la case en haut à gauche (où a été utilisée la commande `\Block`). Si on a demandé la création des nœuds `medium`, alors un nœud de ce type est aussi créé pour ce bloc avec un nom suffixé par `-medium`.

On considère maintenant la matrice suivante que l'on a appelée `exemple`.

```
$\begin{pNiceArray}{CCC}[name=exemple,last-col,create-medium-nodes]
a & a + b & a + b + c & L_1 \\
a & a     & a + b     & L_2 \\
a & a     & a         & L_3
\end{pNiceArray}$
```

$$\begin{pmatrix} a & a+b & a+b+c \\ a & a & a+b \\ a & a & a \end{pmatrix} \begin{matrix} L_1 \\ L_2 \\ L_3 \end{matrix}$$

Si on veut surligner chaque rangée de la matrice, on peut utiliser la technique précédente trois fois.

```
\tikzset{mes-options/.style={remember picture,
                             overlay,
                             name prefix = exemple-,
                             highlight/.style = {fill = red!15,
                                                 blend mode = multiply,
                                                 inner sep = 0pt,
                                                 fit = #1}}}

\begin{tikzpicture}[mes-options]
\node [highlight = (1-1) (1-3)] {} ;
\node [highlight = (2-1) (2-3)] {} ;
\node [highlight = (3-1) (3-3)] {} ;
\end{tikzpicture}
```

On obtient la matrice suivante.

$$\begin{pmatrix} a & a+b & a+b+c \\ a & a & a+b \\ a & a & a \end{pmatrix} \begin{matrix} L_1 \\ L_2 \\ L_3 \end{matrix}$$

Le résultat peut paraître décevant. On peut l'améliorer en utilisant les « nœuds moyens » au lieu des « nœuds normaux ».

```
\begin{tikzpicture}[mes-options, name suffix = -medium]
\node [highlight = (1-1) (1-3)] {} ;
\node [highlight = (2-1) (2-3)] {} ;
\node [highlight = (3-1) (3-3)] {} ;
\end{tikzpicture}
```

On obtient la matrice suivante.

$$\begin{pmatrix} a & a+b & a+b+c \\ a & a & a+b \\ a & a & a \end{pmatrix} \begin{matrix} L_1 \\ L_2 \\ L_3 \end{matrix}$$

Dans l'exemple suivant, on utilise les « nœuds larges » pour surligner une zone de la matrice.

```
\begin{pNiceArray}{>{\strut}CCCC}[create-large-nodes,margin,extra-margin=2pt]
  A_{11} & A_{12} & A_{13} & A_{14} \\
  A_{21} & A_{22} & A_{23} & A_{24} \\
  A_{31} & A_{32} & A_{33} & A_{34} \\
  A_{41} & A_{42} & A_{43} & A_{44}
\CodeAfter
  \tikz \path [name suffix = -large,fill = red!15, blend mode = multiply]
      (1-1.north west)
   |- (2-2.north west)
   |- (3-3.north west)
   |- (4-4.north west)
   |- (4-4.south east)
   |- (1-1.north west) ;
\end{pNiceArray}
```

$$\begin{pmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{pmatrix}$$

14.5 Utilisation directe des nœuds Tikz

Dans l'exemple suivant, on souhaite illustrer le produit mathématique de deux matrices.

L'utilisation de `{NiceMatrixBlock}` avec l'option `auto-columns-width` va permettre que toutes les colonnes aient la même largeur ce qui permettra un alignement des deux matrices superposées.

```
\begin{NiceMatrixBlock}[auto-columns-width]


\NiceMatrixOptions{nullify-dots}
```

Les trois matrices vont être disposées les unes par rapport aux autres grâce à un tableau de LaTeX.

```
$\begin{array}{cc}
&
```

La matrice B a une « première rangée » (pour C_j) d'où l'option `first-row`.

```
\begin{bNiceArray}{C>{\strut}CCCC}[name=B,first-row]
       &          & C_j                      \\
b_{11} & \Cdots   & b_{1j} & \Cdots & b_{1n} \\
\Vdots &          & \Vdots &        & \Vdots \\
       &          & b_{kj}                   \\
       &          & \Vdots                   \\
 b_{n1}  & \Cdots & b_{nj} & \Cdots & b_{nn}
\end{bNiceArray} \\ \\
```

La matrice A a une « première colonne » (pour L_i) d'où l'option `first-col`.

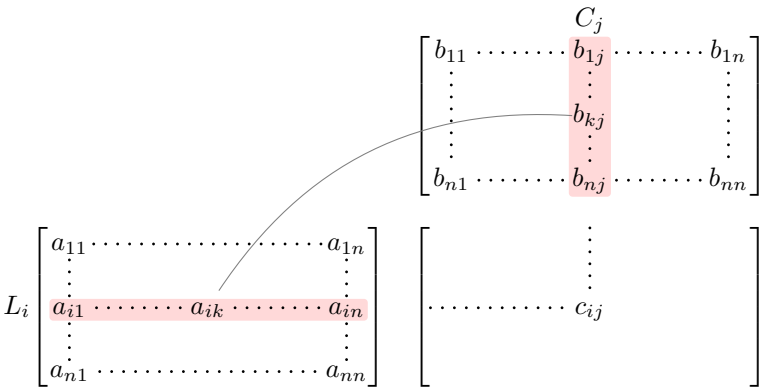
```
\begin{bNiceArray}{CC>{\strut}CCC}[name=A,first-col]
    & a_{11} & \Cdots &        &        & a_{1n} \\
    & \Vdots &        &        &        & \Vdots \\
L_i & a_{i1} & \Cdots & a_{ik} & \Cdots & a_{in} \\
    & \Vdots &        &        &        & \Vdots \\
    & a_{n1} & \Cdots &        &        & a_{nn}
\end{bNiceArray}
&
```

Dans la matrice produit, on remarquera que les lignes en pointillés sont « semi-ouvertes ».

```
\begin{bNiceArray}{CC>{\strut}CCC}
       & &        & & \\
       & & \Vdots     \\
\Cdots & & c_{ij}     \\
\\
\\
\end{bNiceArray}
\end{array}$

\end{NiceMatrixBlock}


\begin{tikzpicture}[remember picture, overlay]
 \node [highlight = (A-3-1) (A-3-5) ] {} ;
 \node [highlight = (B-1-3) (B-5-3) ] {} ;
 \draw [color = gray] (A-3-3) to [bend left] (B-3-3) ;
\end{tikzpicture}
```



Autre documentation

Le document `nicematrix.pdf` (fourni avec l'extension nicematrix) contient une traduction anglaise de la documentation ici présente, ainsi que le code source commenté et un historique des versions.