

L'extension `nicematrix`*

F. Pantigny
fpantigny@wanadoo.fr

21 septembre 2019

Résumé

L'extension LaTeX `nicematrix` fournit de nouveaux environnements similaires aux environnements classiques `{array}` et `{matrix}` mais avec des fonctionnalités supplémentaires. Parmi ces fonctionnalités figurent la possibilité de fixer la largeur des colonnes et de tracer des traits en pointillés continus entre les cases du tableau.

1 Présentation

Cette extension peut être utilisée avec `xelatex`, `lualatex` et `pdflatex` mais aussi avec le cheminement classique `latex-dvips-ps2pdf` (ou Adobe Distiller). Deux ou trois compilations successives peuvent être nécessaires. Cette extension nécessite et charge les extensions `expl3`, `l3keys2e`, `xparse`, `array`, `amsmath` et `tikz`. Elle charge aussi la bibliothèque Tikz `fit`. L'utilisateur final n'a qu'à charger l'extension `nicematrix` avec l'instruction habituelle : `\usepackage{nicematrix}`.

Cette extension fournit quelques outils supplémentaires pour dessiner des matrices (au sens mathématique). Les principales caractéristiques sont les suivantes :

- des lignes en pointillés continues¹;
- des rangées et colonnes extérieures pour les labels;
- un contrôle sur la largeur des colonnes.

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

Une commande `NiceMatrixOptions` est fournie pour régler les options (la portée des options fixées par cette commande est le groupe TeX courant).

Un exemple d'utilisation pour les lignes en pointillés continus

Considérons par exemple le code suivant qui utilise un environnement `{pmatrix}` de l'extension `amsmath`.

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

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

Ce code compose la matrice A représentée à droite.

Maintenant, si nous utilisons l'extension `nicematrix` avec l'option `transparent`, le même code va donner le résultat ci-contre à droite.

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

*Ce document correspond à la version 3.2a of `nicematrix`, en date du 2019/09/21.

1. Si l'option de classe `draft` est utilisée, ces lignes en pointillés ne sont pas tracées pour accélérer la compilation.

2 Les environnements de cette extension

L'extension `nicematrix` définit les nouveaux environnements suivants :

<code>{NiceMatrix}</code>	<code>{NiceArray}</code>	<code>{pNiceArray}</code>
<code>{pNiceMatrix}</code>		<code>{bNiceArray}</code>
<code>{bNiceMatrix}</code>		<code>{BNiceArray}</code>
<code>{BNiceMatrix}</code>		<code>{vNiceArray}</code>
<code>{vNiceMatrix}</code>		<code>{VNiceArray}</code>
<code>{VNiceMatrix}</code>		<code>{NiceArrayWithDelims}</code>

Par défaut, les environnements `{NiceMatrix}`, `{pNiceMatrix}`, `{bNiceMatrix}`, `{BNiceMatrix}`, `{vNiceMatrix}` et `{VNiceMatrix}` se comportent quasiment comme les environnements correspondants de `amsmath` : `{matrix}`, `{pmatrix}`, `{bmatrix}`, `{Bmatrix}`, `{vmatrix}` et `{Vmatrix}`.

L'environnement `{NiceArray}` est similaire à l'environnement `{array}` de l'extension `{array}`. Néanmoins, pour des raisons techniques, dans le préambule de l'environnement `{NiceArray}`, l'utilisateur doit utiliser les lettres `L`, `C` et `R` au lieu de `l`, `c` et `r`. Il est possible d'utiliser les constructions `w{...}{...}`, `W{...}{...}`², `|`, `>{...}`, `<{...}`, `@{...}`, `!{...}` et `*{n}{...}` mais les lettres `p`, `m` et `b` ne doivent pas être employées. L'environnement `{NiceArray}` et ses variantes fournissent des options qui permettent de tracer des rangées et colonnes extérieures. Voir p. 7 la partie concernant `{NiceArray}`.

3 Les lignes en pointillés continues

À l'intérieur des environnements de l'extension `nicematrix`, de nouvelles commandes sont définies : `\Ldots`, `\Cdots`, `\Vdots`, `\Ddots`, and `\Iddots`. Ces commandes sont conçues pour être utilisées à la place de `\dots`, `\cdots`, `\vdots`, `\ddots` et `\iddots`.³

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 vides⁴ 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.

<code>\begin{bNiceMatrix}</code>	
<code>a_1 & \Cdots & & a_1 \\</code>	
<code>\Vdots & a_2 & \Cdots & a_2 \\</code>	
<code>& \Vdots & \Ddots \\</code>	
<code>\\</code>	
<code>a_1 & a_2 & & a_n</code>	
<code>\end{bNiceMatrix}</code>	

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

<code>\begin{bNiceMatrix}</code>	
<code>0 & \Cdots & 0 \\</code>	
<code>\Vdots & & \Vdots \\</code>	
<code>0 & \Cdots & 0</code>	
<code>\end{bNiceMatrix}</code>	

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

2. Pour les colonnes de type `w` et `W`, les cases sont composées en mode mathématique (dans les environnements de `nicematrix`) alors que dans `{array}` de `array`, elles sont composées en mode texte.

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

4. La définition précise de ce qui est considéré comme une «case vide» est donnée plus loin (cf. p. 14).

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 résultant⁵).

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

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

Il y a aussi d’autres moyens de changer la taille d’une matrice. On pourrait vouloir utiliser l’argument optionnel de la commande `\l` pour l’espace vertical et la commande `\hspace*` dans une case pour l’espace horizontal.⁶

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

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

3.1 L’option `nullify-dots`

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

```
$A = \begin{pmatrix}
a_0 & b \\
a_1 & \\
a_2 & \\
a_3 & \\
a_4 & \\
a_5 & b
\end{pmatrix}$
```

$$A = \begin{pmatrix} a_0 & b \\ a_1 & \\ a_2 & \\ a_3 & \\ a_4 & \\ a_5 & b \end{pmatrix}$$

Si nous ajoutons des instructions `\vdots` dans la seconde colonne, la géométrie de la matrice est

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

6. On peut aussi 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

modifiée.

```

$B = \begin{pmatrix}
a_0 & b \\
a_1 & \vdots \\
a_2 & \vdots \\
a_3 & \vdots \\
a_4 & \vdots \\
a_5 & b
\end{pmatrix}

```

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

```

$C = \begin{pNiceMatrix}
a_0 & b \\
a_1 & \Vdots \\
a_2 & \Vdots \\
a_3 & \Vdots \\
a_4 & \Vdots \\
a_5 & b
\end{pNiceMatrix}

```

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 colonne. C'est possible en utilisant l'option `nullify-dots` (et une seule instruction `\Vdots` suffit).

```

$D = \begin{pNiceMatrix}[nullify-dots]
a_0 & b \\
a_1 & \Vdots \\
a_2 & \\
a_3 & \\
a_4 & \\
a_5 & b
\end{pNiceMatrix}

```

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

Il doit n'y avoir aucun espace devant le crochet ouvrant ([) des options de l'environnement.

3.2 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}

```

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

```


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


```

La commande `\hdotsfor` de `amsmath` prend un argument optionnel (entre crochets) qui est utilisé pour un réglage fin de l'espace entre deux points consécutifs. Par homogénéité, `\Hdotsfor` prend aussi un argument optionnel mais cet argument est écarté silencieusement.

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`).

3.3 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 `{matrix}`. Cette option peut être passée en option du `\usepackage` ou bien être fixée avec la commande `\NiceMatrixOptions`.

En fait, cette option est un alias pour la conjonction de deux options : `renew-dots` et `renew-matrix`.

— L'option `renew-dots`

Avec cette option, les commandes `\ldots`, `\cdots`, `\vdots`, `\ddots`, `\iddots`³ et `\hdotsfor` sont redéfinies dans les environnements de `nicematrix` et agissent 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}

```

4 Les nœuds Tikz créés par l'extension `nicematrix`

L'extension `nicematrix` crée un nœud Tikz pour chaque case du tableau considéré. Ces nœuds sont utilisés pour tracer les lignes en pointillés entre les cases du tableau. Toutefois, l'utilisateur peut aussi utiliser directement ces nœuds. On commence par donner un nom au tableau (avec l'option `name`). Cela étant fait, 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 ligne et de colonne de la case considérée.

```


$$\begin{pNiceMatrix}[name=ma-matrice]
1 & 2 & 3 \\
4 & 5 & 6 \\
7 & 8 & 9 \\
\end{pNiceMatrix}$$


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


```

\tikz[remember picture,overlay]
\draw (ma-matrice-2-2) circle (2mm) ;

```


```

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

Dans l'exemple suivant, nous avons surligné toutes les cases de la matrice.

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

En fait, l'extension `nicematrix` peut créer des nœuds supplémentaires (*extra nodes* en anglais). Ces nouveaux nœuds sont créés si l'option `create-extra-nodes` est utilisée. Il y a deux séries de nœuds supplémentaires : les «nœuds moyens» (*medium nodes* en anglais) et les «nœuds larges» (*large nodes* en anglais).

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.⁷

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

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`.⁸

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

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.

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

Dans le cas présent, si on veut un contrôle sur la hauteur des rangées, on peut ajouter un `\strut` dans chaque rangée du tableau.

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

On explique plus loin comment surligner les nœuds créés par Tikz (cf. p. 18).

7. Il n'y a pas de «nœuds larges» créés dans les lignes et colonnes extérieures (pour ces lignes et colonnes, voir p. 7).

8. 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`.

5 Le code-after

L'option `code-after` peut être utilisée pour indiquer du code qui sera exécuté après la construction de la matrice (et donc, en particulier, après la construction de tous les nœuds).

Dans le `code-after`, les nœuds Tikz doivent être désignés sous la forme $i-j$ (sans le préfixe correspondant au nom de l'environnement).

De plus, une commande spéciale, nommée `\line` est disponible pour tracer directement des lignes en pointillés entre les nœuds).

```


$$\begin{pNiceMatrix}[\code-after = {\line {1-1} {3-3}}]
0 & 0 & 0 \\
0 & & 0 \\
0 & 0 & 0
\end{pNiceMatrix}$$


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


```

6 L'environnement {NiceArray}

L'environnement `{NiceArray}` est similaire à l'environnement `{array}`. Comme pour `{array}`, l'argument obligatoire est le préambule du tableau. Néanmoins, pour des raisons techniques, l'utilisateur doit utiliser les lettres `L`, `C` et `R`⁹ au lieu de `l`, `c` et `r`.

Il est possible d'utiliser les constructions `w{...}{...}`, `W{...}{...}`, `l`, `>{...}`, `<{...}`, `@{...}`, `!{...}` et `*{n}{...}` mais les lettres `p`, `m` et `b` ne doivent pas être employées.¹⁰

L'environnement `{NiceArray}` accepte les options classiques `t`, `c` et `b` de `{array}` mais aussi d'autres options définies par `nicematrix` (`renew-dots`, `columns-width`, etc.).

Un exemple avec un système linéaire (on a besoin de `{NiceArray}` pour le trait vertical)

```


$$\left[ \begin{NiceArray}{CCCC|C}
a_1 & ? & \Cdots & ? & ? \\
0 & & \Ddots & \Vdots & \Vdots \\
\Vdots & \Ddots & \Ddots & ? \\
0 & \Cdots & 0 & a_n & ?
\end{NiceArray} \right]$$


$$\left[ \begin{array}{cccc|c} a_1 & ? & \cdots & ? & ? \\ 0 & & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & ? & \vdots \\ 0 & \cdots & 0 & a_n & ? \end{array} \right]$$


```

Il existe également des variantes pour l'environnement `{NiceArray}` : `{pNiceArray}`, `{bNiceArray}`, `{BNiceArray}`, `{vNiceArray}` et `{VNiceArray}`.

Dans l'exemple suivant, on utilise un environnement `{pNiceArray}` (on n'utilise pas `{pNiceMatrix}` car on souhaite utiliser les types de colonne `L` et `R` — avec `{pNiceMatrix}`, toutes les colonnes sont de type `C`).

```


$$\begin{pNiceArray}{LCR}
a_{11} & \Cdots & a_{1n} \\
a_{21} & & a_{2n} \\
\Vdots & & \Vdots \\
a_{n-1,1} & \Cdots & a_{n-1,n}
\end{pNiceArray}$$


$$\begin{pmatrix} a_{11} & \cdots & a_{1n} \\ a_{21} & & a_{2n} \\ \vdots & & \vdots \\ a_{n-1,1} & \cdots & a_{n-1,n} \end{pmatrix}$$


```

L'environnement `{NiceArray}` et ses variantes permettent de composer des rangées et des colonnes «extérieures» grâce aux options `first-row`, `last-row`, `first-col` et `last-col`.

9. Les types de colonnes `L`, `C` et `R` sont définis localement à l'intérieur de `{NiceArray}` avec la commande `\newcolumntype` de `array`. Cette définition masque une éventuelle définition précédente. En fait, les types de colonnes `w` and `W` sont également redéfinis.

10. Dans une commande `\multicolumn`, on doit également utiliser les lettres `L`, `C` et `R`.

Il n’y a pas de spécification de colonne à fournir pour l’éventuelle «première colonne» (qui sera automatiquement une colonne de type R) ou pour l’éventuelle «dernière colonne» (qui sera automatiquement de type L).

```
\begin{pNiceArray}{CCCC}[first-row,last-row,first-col,last-col]
& C_1 & & C_2 & & C_3 & & C_4 & & \\
L_1 & a_{11} & & a_{12} & & a_{13} & & a_{14} & & L_1 \\
L_2 & a_{21} & & a_{22} & & a_{23} & & a_{24} & & L_2 \\
L_3 & a_{31} & & a_{32} & & a_{33} & & a_{34} & & L_3 \\
L_4 & a_{41} & & a_{42} & & a_{43} & & a_{44} & & L_4 \\
& C_1 & & C_2 & & C_3 & & C_4 & & \\
\end{pNiceArray}
```

$$\begin{array}{cccc}
& C_1 & C_2 & C_3 & C_4 \\
L_1 & \left(a_{11} & a_{12} & a_{13} & a_{14} \right) & L_1 \\
L_2 & \left(a_{21} & a_{22} & a_{23} & a_{24} \right) & L_2 \\
L_3 & \left(a_{31} & a_{32} & a_{33} & a_{34} \right) & L_3 \\
L_4 & \left(a_{41} & a_{42} & a_{43} & a_{44} \right) & L_4 \\
& C_1 & C_2 & C_3 & C_4
\end{array}$$

Néanmoins, il y a une particularité avec l’option `last-row` : quand LaTeX compose un tableau `{array}` (via une commande `\halign` de TeX), il le compose rangée par rangée et il n’y a pas de moyen direct de savoir que l’on est arrivé à la dernière rangée avant la composition de celle-ci. Pour remédier à ce problème, `nicematrix` écrit dans le fichier `aux` le nombre de rangées du tableau pour pouvoir l’utiliser à la compilation suivante. Il est néanmoins également possible de passer directement le nombre de rangées en argument de l’option `last-row` et, ainsi, `nicematrix` le connaîtra dès la première compilation.

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=5,first-col,last-col]
& C_1 & & C_2 & & C_3 & & C_4 & & \\
L_1 & a_{11} & & a_{12} & & a_{13} & & a_{14} & & L_1 \\
L_2 & a_{21} & & a_{22} & & a_{23} & & a_{24} & & L_2 \\
\hline
L_3 & a_{31} & & a_{32} & & a_{33} & & a_{34} & & L_3 \\
L_4 & a_{41} & & a_{42} & & a_{43} & & a_{44} & & L_4 \\
& C_1 & & C_2 & & C_3 & & C_4 & & \\
\end{pNiceArray}
```

$$\begin{array}{cccc}
& C_1 & C_2 & C_3 & C_4 \\
L_1 & \left(a_{11} & a_{12} & a_{13} & a_{14} \right) & L_1 \\
L_2 & \left(a_{21} & a_{22} & a_{23} & a_{24} \right) & L_2 \\
L_3 & \left(a_{31} & a_{32} & a_{33} & a_{34} \right) & L_3 \\
L_4 & \left(a_{41} & a_{42} & a_{43} & a_{44} \right) & L_4 \\
& C_1 & C_2 & C_3 & C_4
\end{array}$$

Remarques

- Comme on peut le voir dans l’exemple précédent, un filet horizontal (tracé avec `\hline`) ne s’étend pas dans les colonnes extérieures et un filet vertical (spécifié par un caractère «|» dans le préambule du tableau) ne s’étend pas dans les rangées extérieures.¹¹

11. Ce dernier point n’est pas valable si on a chargé, en plus de `nicematrix`, l’extension `arydshln`. Les extensions `nicematrix` et `arydshln` ne sont pas parfaitement compatibles car `arydshln` redéfinit beaucoup de structures internes à `array`. Par ailleurs, si on veut vraiment un filet qui s’étende dans la première et la dernière rangée, on peut utiliser `!\vline` dans le préambule à la place de `|`.

- La «première rangée» d'un environnement `{pNiceArray}` porte le numéro 0, et non 1 (de même pour la «première colonne»). Ce numéro est utilisé, par exemple, pour les noms des nœuds Tikz (les noms de ces nœuds sont utilisés, par exemple, par la commande `\line` dans le `code-after`).
- 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).

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.

```


$$\begin{NiceArrayWithDelims}
{\downarrow}{\downarrow}{CCC}[last-col]
1 & 2 & 3 & L_1 \\
4 & 5 & 6 & L_2 \\
7 & 8 & 9 & L_3
\end{NiceArrayWithDelims}$$


```

7 Les lignes en pointillés pour séparer les rangées et les colonnes

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`).

```


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


```

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

```


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


```

Ces lignes en pointillés ne s'étendent pas dans les rangées et colonnes extérieures.

```


$$\begin{pNiceArray}{CCC:C}%
[first-row,last-col,
code-for-first-row = \color{blue}\scriptstyle,
code-for-last-col = \color{blue}\scriptstyle ]
C_1 & C_2 & C_3 & C_4 \\
1 & 2 & 3 & 4 \\
5 & 6 & 7 & 8 \\
9 & 10 & 11 & 12 \\
\hdottedline
13 & 14 & 15 & 16
\end{pNiceArray}$$


```

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`.

Par exemple, dans ce document, nous avons chargé l'extension `arydshln` qui utilise la lettre «:» pour indiquer un trait vertical en tiretés. Par conséquent, en utilisant l'option `letter-for-dotted-lines`, on peut utiliser les traits verticaux fournis à la fois par `arydshln` et par `nicematrix`.

```
\NiceMatrixOptions{letter-for-dotted-lines = V}
\begin{pNiceArray}{C|C:CVC}
1 & 2 & 3 & 4 \\
5 & 6 & 7 & 8 \\
9 & 10 & 11 & 12
\end{pNiceArray}
```

$$\left(\begin{array}{c|ccc} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \end{array} \right)$$

8 La largeur des colonnes

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

```
\begin{pNiceArray}{Wc{1cm}CC}
1 & 12 & -123 \\
12 & 0 & 0 \\
4 & 1 & 2
\end{pNiceArray}
```

$$\left(\begin{array}{ccc} 1 & 12 & -123 \\ 12 & 0 & 0 \\ 4 & 1 & 2 \end{array} \right)$$

Avec `nicematrix`, il est aussi possible de fixer la largeur de toutes les colonnes de la matrice directement avec l'option `columns-width` (disponible dans tous les environnements de `nicematrix`).

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

$$\left(\begin{array}{ccc} 1 & 12 & -123 \\ 12 & 0 & 0 \\ 4 & 1 & 2 \end{array} \right)$$

Noter que l'espace inséré entre deux colonnes (égal à `2 \arraycolsep`) n'est pas supprimé (il est évidemment possible de le supprimer en mettant `\arraycolsep` à 0).

Il est possible de donner la valeur `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. **Deux ou trois compilations successives peuvent être nécessaires.**

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

$$\left(\begin{array}{ccc} 1 & 12 & -123 \\ 12 & 0 & 0 \\ 4 & 1 & 2 \end{array} \right)$$

Il est possible de fixer la largeur 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}
```

$$\left(\begin{array}{cc} a & b \\ c & d \end{array} \right) = \left(\begin{array}{cc} 1 & 1245 \\ 345 & 2 \end{array} \right)$$

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`.¹²

```
\begin{NiceMatrixBlock}[auto-columns-width]
$\begin{pNiceMatrix}
a & b \\ c & d
\end{pNiceMatrix}
=
\begin{pNiceMatrix}
1 & 1245 \\ 345 & 2
\end{pNiceMatrix}$
\end{NiceMatrixBlock}
```

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

9 Les matrices par blocs

Dans les environnements de `nicematrix`, on peut utiliser la commande `\Block` pour placer un élément au centre d'un rectangle de cellules fusionnées.

La commande `\Block` doit être utilisée dans la cellule 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 (en mode mathématique).

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

$$\left[\begin{array}{ccc|c} A & & 0 \\ & \hspace{1cm} & \vdots \\ & & 0 \\ \hline 0 & \dots & 0 & 0 \end{array} \right]$$

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.

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

$$\left[\begin{array}{ccc|c} A & & 0 \\ & \hspace{1cm} & \vdots \\ & & 0 \\ \hline 0 & \dots & 0 & 0 \end{array} \right]$$

Pour des raisons techniques, il n'est pas possible d'écrire `\Block{i-j}<>` (mais on peut écrire `\Block{i-j}<><>` avec le résultat attendu).

10 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

¹². Pour le moment, c'est le seul usage de l'environnement `{NiceMatrixBlock}` mais il pourrait y en avoir davantage dans le futur.

`{psmallmatrix}`, `{bsmallmatrix}`, etc. de `mathtools`).

```

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

```

$$\begin{bmatrix} 1 & -2 & 3 & 4 & 5 \\ 0 & 3 & 2 & 1 & 2 \\ 0 & 1 & 1 & 2 & 3 \end{bmatrix} \begin{matrix} L_2+2L_1-L_2 \\ L_3+L_1+L_3 \end{matrix}$$

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 `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 L’option `hlines`

Dans les environnements de `nicematrix`, on peut bien entendu ajouter des filets horizontaux entre les lignes avec la commande `\hline`. Par souci de commodité, l’extension `nicematrix` fournit l’option `hlines` qui impose directement que tous les filets horizontaux soient tracés (à l’exception, très naturelle, du filet avant l’éventuelle « première rangée » et après l’éventuelle « dernière rangée »).

```

 $\begin{NiceArray}{|*{4}{C|}}[hlines,first-row,first-col]
& e & a & b & c \\
e & e & a & b & c \\
a & a & e & c & b \\
b & b & c & e & a \\
c & c & b & a & e \\
\end{NiceArray}$ 

```

	<i>e</i>	<i>a</i>	<i>b</i>	<i>c</i>
<i>e</i>	<i>e</i>	<i>a</i>	<i>b</i>	<i>c</i>
<i>a</i>	<i>a</i>	<i>e</i>	<i>c</i>	<i>b</i>
<i>b</i>	<i>b</i>	<i>c</i>	<i>e</i>	<i>a</i>
<i>c</i>	<i>c</i>	<i>b</i>	<i>a</i>	<i>e</i>

12 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`. Les colonnes `d` de l’extension `dcolumn` ne sont pas prises en charge par `nicematrix`.

```

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

```

	C_1	C_n
$\left($	2.3	00
	12.4	⋮	⋮
	1.45	⋮	⋮
	7.2	00
$\right)$			

13 Remarques techniques

13.1 Intersection des lignes pointillées

Depuis la version 3.1 de `nicematrix`, les lignes en pointillées créées par `\Cdots`, `\Ldots`, `\Vdots`, etc. ne peuvent pas se croiser entre elles.¹³

Cela signifie qu'une ligne en pointillés créée par l'une des ces commandes s'arrête automatiquement quand elle arrive à une autre ligne pointillée déjà tracée par l'une de ces commandes. Par conséquent, l'ordre dans lequel les lignes sont tracées a son importance pour le résultat final. Voici cet ordre (c'est à dessein qu'il a été choisi ainsi) : `\Hdotsfor`, `\Vdots`, `\Ddots`, `\Iddots`, `\Cdots` and `\Ldots`.

De ce fait, on peut tracer la matrice suivante :

```


$$\begin{pmatrix} 1 & 2 & 3 & \dots & n \\ 1 & 2 & 3 & \dots & n \\ \vdots & \cdots & & \hspace*{15mm} & \vdots \\ & \cdots & & & \end{pmatrix}$$


```

13.2 Lignes diagonales

Par défaut, toutes les lignes diagonales¹⁴ 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 :

Exemple avec parallélisation (comportement par défaut) :

```


$$A = \begin{pmatrix} 1 & \dots & 1 \\ a+b & \color{blue}{\Ddots} & \Vdots \\ \Vdots & \Ddots & \\ a+b & \dots & a+b & 1 \end{pmatrix}$$


```

```


$$A = \begin{pmatrix} 1 & \dots & 1 \\ a+b & & \Vdots \\ \Vdots & \color{blue}{\Ddots} & \Ddots & \\ a+b & \dots & a+b & 1 \end{pmatrix}$$


```

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

Le même exemple sans parallélisation :

$$A = \begin{pmatrix} 1 & \dots & 1 \\ a+b & & \Vdots \\ \Vdots & & \Ddots & \\ a+b & \dots & a+b & 1 \end{pmatrix}$$

13. En revanche, les lignes créées par `\hdottedline`, la lettre “:” dans le préambule de la matrice et la commande `\line` dans le `code-after` peuvent croiser une autre ligne en pointillés.

14. On parle des lignes créées par `\Ddots` et non des lignes créées par une commande `\line` dans le `code-after`.

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 inférieure à 0.5 pt est vide.
- Une case qui contient une commande `\Ldots`, `\Cdots`, `\Vdots`, `\Ddots` ou `\Iddots` est vide. On rappelle que ces commandes doivent être employées seules dans une case.
- 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.¹⁵

L'environnement `{matrix}` et ses variantes (`{pmatrix}`, `{vmatrix}`, etc.) de `amsmath` préfèrent supprimer ces espaces avec des instructions explicites `\hskip -\arraycolsep` et `{NiceArray}` agit de même.

Néanmoins, l'utilisateur peut changer ce comportement avec l'option `exterior-arraycolsep` de la commande `\NiceMatrixOptions`. Avec cette option, `{NiceArray}` va insérer le même espacement horizontal que `{array}`.

Cette option est seulement pour la «compatibilité» puisque l'extension `nicematrix` fournit un contrôle plus précis avec les options `left-margin`, `extra-left-margin`, etc.

13.5 L'option de classe `draft`

L'extension `nicematrix` est relativement lente pour tracer les lignes en pointillés (créées par `\Cdots`, `\Ldots`, `\Ddots`, etc. mais aussi par le spécificateur «:»).¹⁶

C'est pourquoi, avec l'option de classe `draft`, les lignes en pointillés ne sont pas tracées, pour accélérer la compilation.

13.6 Un problème technique avec l'argument de `\\`

Pour des raisons techniques, si vous utilisez l'argument optionnel de la commande `\\`, l'espace vertical sera aussi ajouté au nœud («normal») correspondant à la case précédente.

¹⁵. 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)*. Il est possible de supprimer les espaces pour un environnement `{array}` donné par une construction du type `\begin{array}{@{}cccc@{}}`.

¹⁶. La principale raison est que nous voulons des lignes en pointillés avec des points ronds et non carrés. Pour atteindre ce but, il a fallu construire notre propre système de lignes en pointillés.

```

\begin{pNiceMatrix}
a & \frac{A}{B} \\
b & c
\end{pNiceMatrix}

```

$$\begin{pmatrix} a & \frac{A}{B} \\ b & c \end{pmatrix}$$

Il y a deux solutions pour résoudre ce problème. La première solution est d'utiliser une commande TeX pour insérer l'espace entre les deux rangées.

```

\begin{pNiceMatrix}
a & \frac{A}{B} \\
\noalign{\kern2mm}
b & c
\end{pNiceMatrix}

```

$$\begin{pmatrix} a & \frac{A}{B} \\ b & c \end{pmatrix}$$

L'autre solution est d'utiliser la commande `\multicolumn` dans la case précédente :

```

\begin{pNiceMatrix}
a & \multicolumn{1C}{\frac{A}{B}} \\
b & c
\end{pNiceMatrix}

```

$$\begin{pmatrix} a & \frac{A}{B} \\ b & c \end{pmatrix}$$

13.7 Environnements obsolètes

La version 3.0 de `nicematrix` a introduit l'environnement `{pNiceArray}` (et ses variantes) avec les options `first-row`, `last-row`, `first-col` et `last-col`.

Par conséquent, les environnements suivants, présents dans les versions précédentes de `nicematrix` sont devenus obsolètes :

- `{NiceArrayCwithDelims}` ;
- `{pNiceArrayC}`, `{bNiceArrayC}`, `{BNiceArrayC}`, `{vNiceArrayC}`, `{vNiceArrayC}` ;
- `{NiceArrayRCwithDelims}` ;
- `{pNiceArrayRC}`, `{bNiceArrayRC}`, `{BNiceArrayRC}`, `{vNiceArrayRC}`, `{vNiceArrayRC}`.

Ils seront peut-être supprimés dans une version future de `nicematrix`.

14 Exemples

14.1 Lignes en pointillés

Une matrice tridiagonale :

```

 $\begin{pNiceMatrix}[nullify-dots]
a & & b & & 0 & & & & & & \Cdots & & 0 & & \\
b & & a & & b & & & & & & \Ddots & & & & \Vdots \\
0 & & b & & a & & & & & & \Ddots & & & & \\
& & & & \Ddots & & \Ddots & & \Ddots & & & & & & 0 \\
\Vdots & & & & & & & & & & & & & & b \\
0 & & \Cdots & & & & 0 & & b & & & & & & a
\end{pNiceMatrix}$ 

```

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

Une matrice de permutation :

```

 $\begin{pNiceMatrix}$ 
0 & 1 & 0 & & \Cdots & 0 & \\
\Vdots & & & \Ddots & & \Vdots & \\
& & & \Ddots & & & \\
& & & \Ddots & & 0 & \\
0 & 0 & & & & 1 & \\
1 & 0 & & \Cdots & & 0 & \\
\end{pNiceMatrix}

```

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

Un exemple avec `\Iddots` :

```

 $\begin{pNiceMatrix}$ 
1 & & \Cdots & & 1 & \\
\Vdots & & & & 0 & \\
& & \Iddots & & \Iddots & \Vdots \\
1 & & 0 & & \Cdots & 0 \\
\end{pNiceMatrix}

```

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

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}

```

$$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\ \dots\dots\dots 10 \text{ autres lignes} \dots\dots\dots \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{pmatrix}$$

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}

```

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


```

\begin{NiceMatrixBlock}[auto-columns-width]
\NiceMatrixOptions{code-for-last-col = \color{blue}\scriptstyle}
\setlength{\extrarowheight}{1mm}
$\begin{pNiceArray}{CCCC:C}[last-col]
1&1&1&1&\backslash\backslash
2&4&8&16&9&\backslash\backslash
3&9&27&81&36&\backslash\backslash
4&16&64&256&100&
\end{pNiceArray}$
...
\end{NiceMatrixBlock}

```

$$\begin{array}{c}
\left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & \dots & 1 \\ 2 & 4 & 8 & 16 & \dots & 9 \\ 3 & 9 & 27 & 81 & \dots & 36 \\ 4 & 16 & 64 & 256 & \dots & 100 \end{array} \right) \\
\left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & \dots & 1 \\ 0 & 2 & 6 & 14 & \dots & 7 \\ 0 & 6 & 24 & 78 & \dots & 33 \\ 0 & 12 & 60 & 252 & \dots & 96 \end{array} \right) \begin{array}{l} L_2 \leftarrow -2L_1 + L_2 \\ L_3 \leftarrow -3L_1 + L_3 \\ L_4 \leftarrow -4L_1 + L_4 \end{array} \\
\left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & \dots & 1 \\ 0 & 1 & 3 & 7 & \dots & \frac{7}{2} \\ 0 & 3 & 12 & 39 & \dots & \frac{33}{2} \\ 0 & 1 & 5 & 21 & \dots & 8 \end{array} \right) \begin{array}{l} L_2 \leftarrow \frac{1}{2}L_2 \\ L_3 \leftarrow \frac{1}{2}L_3 \\ L_4 \leftarrow \frac{1}{2}L_4 \end{array} \\
\left(\begin{array}{cccc|c} 1 & 1 & 1 & 1 & \dots & 1 \\ 0 & 1 & 3 & 7 & \dots & \frac{7}{2} \\ 0 & 0 & 1 & 6 & \dots & 2 \\ 0 & 0 & 0 & -2 & \dots & -\frac{1}{2} \end{array} \right) \begin{array}{l} L_3 \leftarrow -3L_2 + L_3 \\ L_4 \leftarrow L_2 - L_4 \\ L_3 \leftarrow \frac{1}{3}L_3 \\ L_4 \leftarrow 2L_3 + L_4 \end{array}
\end{array}$$

14.3 Comment surligner les cases

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»). Pour avoir la continuité des lignes, on doit fixer `inner sep = -\pgflinewidth/2`.

```

$\begin{pNiceArray}{>{\strut}CCCC}%
[create-extra-nodes,margin,extra-margin=2pt,
code-after = {\begin{tikzpicture}
[name suffix = -large,
every node/.style = {draw,
inner sep = -\pgflinewidth/2}]
\node [fit = (1-1)] {} ;
\node [fit = (2-2)] {} ;
\node [fit = (3-3)] {} ;
\node [fit = (4-4)] {} ;
\end{tikzpicture}}]
a_{11} & a_{12} & a_{13} & a_{14} \backslash\backslash
a_{21} & a_{22} & a_{23} & a_{24} \backslash\backslash
a_{31} & a_{32} & a_{33} & a_{34} \backslash\backslash
a_{41} & a_{42} & a_{43} & a_{44}
\end{pNiceArray}$

```

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

L'extension `nicematrix` est construite au-dessus de l'environnement `{array}` et, par conséquent, il est possible d'utiliser l'extension `colortbl` dans les environnements de `nicematrix`. Les possibilités de réglage de `colortbl` sont néanmoins assez limitées. C'est pourquoi nous proposons 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`. Attention : certains lecteurs de PDF ne sont pas capables de rendre la transparence correctement.

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

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

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

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
```

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

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

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

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-,
                              every node/.style = {fill = red!15,
                                                    blend mode = multiply,
                                                    inner sep = 0pt}}}
```

```

\begin{tikzpicture}[mes-options]
\node [fit = (1-1) (1-3)] {};
\node [fit = (2-1) (2-3)] {};
\node [fit = (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 [fit = (1-1) (1-3)] {};
\node [fit = (2-1) (2-3)] {};
\node [fit = (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.

```

\left(\,\begin{NiceArray}{>{\strut}CCCC}%
[create-extra-nodes,left-margin,right-margin,
code-after = {\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) ; } ]
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{NiceArray}\,,\right)

```

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

14.4 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}
```

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_{nn} \\\
& \Vdots & & & \Vdots \\\
L_i & a_{i1} & \Cdots & a_{ik} & \Cdots & a_{in} \\\
& \Vdots & & & \Vdots \\\
& a_{n1} & \Cdots & & a_{nn}
```

```
&
```

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, fit = (A-3-1) (A-3-5) ] {} ;
```

```
\node [highlight, fit = (B-1-3) (B-5-3) ] {} ;
```

```
\draw [color = gray] (A-3-3) to [bend left] (B-3-3) ;
```

```
\end{tikzpicture}
```

