

Le package `tnscom`

Code source disponible sur <https://github.com/typensee-latex/tnscom.git>.

Version 0.0.0-beta développée et testée sur Mac OS X.

Christophe BAL

2020-07-10

Table des matières

1	Introduction	3
2	A propos des macros standards redéfinies	3
3	Comment lire cette documentation ?	3
4	Deux séparateurs d'arguments par défaut	3
5	Quelques modifications générales	3
5.1	Espace et point-virgule avec l'option <code>french</code> de <code>babel</code>	3
5.2	Espace et fractions	3
5.3	Espace et racines n-ièmes d'un réel	4
6	En coulisse...	4
6.1	Nouvelle macro modifiant les arguments en amont de l'application d'une ancienne macro	4
6.2	Macro avec un « multi-argument »	4
6.3	Intervalles « généralisés »	5
6.4	Décorer un opérateur	5
6.5	Produit en croix	6
6.6	Déterminant 2D ou produit en croix décoré	6
7	Historique	8
8	Toutes les fiches techniques	9
8.1	Deux séparateurs d'arguments par défaut	9
8.1.1	Deux séparateurs d'arguments par défaut	9
8.1.2	Espace et fractions	9
8.1.3	Espace et racines n-ièmes d'un réel	9
8.2	En coulisse...	9
8.2.1	Nouvelle macro modifiant les arguments en amont de l'application d'une an- cienne macro	9
8.2.2	Macro avec un « multi-argument »	9

8.2.3	Intervalles « généralisés »	10
8.2.4	Décorer un opérateur	10
8.2.5	Produit en croix	10
8.2.6	Déterminant 2D ou produit en croix décoré	10

1 Introduction

Ce package contient quelques macros utiles à différents packages de la série `tns` : la majorité des macros sont pour les coulisses et quelques unes concernent la mise en forme.

2 A propos des macros standards redéfinies

Certaines macros comme `\frac` sont un peu revues par `tnscom`. Dans ce cas, les versions standard restent accessibles en utilisant le préfixe `std` ce qui donne ici la macro `\stdfrac`.

3 Comment lire cette documentation ?

Le choix a été fait de fournir des exemples comme documentation du package suivis de fiches techniques des macros-commandes (*se reporter à la section 8*). Les exemples se présentent comme ci-dessous et sont généralement très courts.

```
\dfrac{1}{2} = \stdfrac{1}{2}
```

$$\frac{1}{2} = \frac{1}{2}$$

4 Deux séparateurs d'arguments par défaut

La macro `\tnsmathsep` définit le séparateur d'arguments de premier niveau et `\tnsmathsubsep` celui des arguments de deuxième niveau. Cette documentation utilisant l'option `french` de `babel`, la valeur de `\tnsmathsep` est `;` et celle de `\tnsmathsubsep` est `,`. Sans ce choix, les valeurs de `\tnsmathsep` et `\tnsmathsubsep` seront `,` et `;` respectivement à l'anglaise.

5 Quelques modifications générales

5.1 Espace et point-virgule avec l'option `french` de `babel`

Seulement si vous utilisez `babel` avec l'option `french`, comme c'est le cas dans cette documentation alors vous verrez le même espacement autour du point-virgule en mode mathématique comme dans $A(x;y)$. Que c'est beau !

5.2 Espace et fractions

Quand on utilise `\frac` ou `\dfrac` de petits espaces sont automatiquement ajoutés pour éviter d'avoir des traits de fraction trop petits. Le comportement par défaut se retrouve en utilisant les macros `\stdfrac` et `\stdfrac`. Voici un exemple.

```
\frac{2}{3} = \stdfrac{2}{3} ,  
\dfrac{2}{3} = \stdfrac{2}{3}
```

$$\frac{2}{3} = \frac{2}{3}, \frac{2}{3} = \frac{2}{3}$$

5.3 Espace et racines n-ièmes d'un réel

`\sqrt` a été redéfinie pour ajouter un peu d'espaces sur la droite sous le radical. Le comportement par défaut se retrouve en utilisant la macro `\stdsqrt`. Voici un exemple.

<pre><code>\sqrt{2} = \stdsqrt{2}\$, \sqrt{x_2} = \stdsqrt{x_2}\$ \sqrt[n]{45} = \stdsqrt[n]{45}\$, \sqrt[p]{7} = \stdsqrt[p]{7}\$</code></pre>	$\sqrt{2} = \sqrt{2} , \sqrt{x_2} = \sqrt{x_2}$ $\sqrt[n]{45} = \sqrt[n]{45} , \sqrt[p]{7} = \sqrt[p]{7}$
--	---

6 En coulisse...

Cette section présente les macros dites « *privées* » proposées par `tnscom`. Toutes ces macros ont de noms commençant par `\tns@` et aucune version étoilée n'est proposée¹.

6.1 Nouvelle macro modifiant les arguments en amont de l'application d'une ancienne macro

Comme cette fonctionnalité est utilisée par plusieurs packages de la suite `tns`, une mini macro permet de faciliter la définition de ce type de nouvelle macro. Par exemple, ci-dessous la 2^e macro ajoute juste une mise en forme particulière aux deux arguments juste avant d'appliquer la 1^{re} macro.

<pre><code>\twoargs{A}{B} : \newtwoargs{A}{B}</code></pre>	$A, B : ([A]), ([B])$
--	-----------------------

Ceci est géré facilement via la macro `\tns@apply@macro@two@args` comme suit.

```
\newcommand\twoargs[2]{#1, #2}
\newcommand\modify[1]{\textbf{([#1])}}

\def\newtwoargs{\tns@apply@macro@two@args\twoargs\modify}
```

Remarque. En interne la macro `\tns@apply@macro@two@args` est définie avec quatre arguments. En fait ci-dessus nous utilisons la machinerie L^AT_EX qui va manger les deux arguments manquants lors de l'utilisation de `\newtwoargs`.

6.2 Macro avec un « multi-argument »

La suite `tns` propose la possibilité d'avoir des macros avec un nombre variables d'arguments. Pour ce type de macros, le choix a été fait de passer via unique argument au sens L^AT_EX mais contenant des « *sous-arguemnts* » séparés par des barres verticales, ou pipe. Nous parlerons de « multi-argument ». Voici une exemple d'utilisation spécifique.

<pre><code>\verticalcoord{1}\$, \verticalcoord{1 2}\$, \verticalcoord{1 2 3}\$, \verticalcoord{1 2 3 4}\$, \verticalcoord{1 2 3 4 5}\$...</code></pre>	$[1] , \begin{bmatrix} 1 \\ 2 \end{bmatrix} , \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} , \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{bmatrix} , \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{bmatrix} \dots$
---	---

1. « Explicite » est mieux que « implicite ».

La macro `\verticalcoord` a été définie comme suit via la macro `\tns@multi@args`. L'environnement `bmatrix` via du très pratique package `nicematrix` qu'utilise déjà `tnscom`.

```
\newcommand\verticalcoord[1]{%
  \tns@multi@args{||}{#1}      % Séparateur et argument au sens LaTeX
  {\begin{bmatrix}} % Matériel avant
  {\|} % Ce qui remplace le séparateur
  {\end{bmatrix}} % Matériel après
}
```

6.3 Intervalles « généralisés »

Il est possible définir facilement des sortes d'intervalles « généralisés ».

Exemple 1 – Mode extensible

Il est assez facile de définir une macro ayant le comportement suivant.

<code>\strangeset{a}{\dfrac{b}{c}}\$</code>	$\left\{ a :: \frac{b}{c} \right)$
---	------------------------------------

La macro `\strangeset` a été définie comme suit via `\tns@generic@interval@ext`.

```
\newcommand\strangeset[2]{%
  \tns@generic@interval@ext{\{ % 1e délimiteur
    {#1} % 1e élément
    {::} % Séparateur entre les deux éléments
    {#2} % 2e élément
    {\} % 2e délimiteur
  }
}
```

Exemple 2 – Mode semi-extensible

Le mode semi-extensible correspond à des délimiteurs un peu plus grand qu'en mode non extensible comme le montre l'exemple ci-après.

<code>\myinter{a}{\dfrac{b}{c}}\$</code> ou <code>\{ a :: \dfrac{b}{c})\$</code>	$\left\{ a :: \frac{b}{c} \right) \text{ ou } \{ a :: \frac{b}{c}$
---	--

Il suffit d'utiliser `\tns@generic@interval@semi@ext` au lieu de `\tns@generic@interval@ext`. Voici le code utilisé.

```
\newcommand\myinter[2]{%
  \tns@generic@interval@semi@ext{\{ %
    {#1}{::}{#2}%
    {\}%
  }
}
```

6.4 Décorer un opérateur

Il est facile d'obtenir l'effet suivant via la macro `\tns@over@math@symbol`.

```
$1 \eqtxt un$
```

$$1 \overset{\text{texte}}{=} un$$

Le code qui a été utilisé est le suivant.

```
\newcommand\eqtxt{\tns@over@math@symbol{texte}{=}}
```

6.5 Produit en croix

Comme ce type de calculs apparait très souvent dans divers domaines, une macro privée est chargée de ce tout petit travail de mise en forme.

```
\makeatletter
$\tns@prop@prod{\cdot} % Opérateur
      {1}{2} % Les éléments
      {3}{4}$ % du tableau
\makeatother
```

$$1 \cdot 4 - 3 \cdot 2$$

6.6 Déterminant 2D ou produit en croix décoré

Des macros privées permettent, suivant le contexte, d'écrire un calcul de proportionnalité ou bien celui d'un déterminant avec différentes mises en forme possibles.

Exemple 1 – La totale

La macro `\tns@det@plane@deco` est facile d'utilisation comme le montre l'exemple suivant.

```
\makeatletter
$\tns@det@plane@deco%
  {vec} % Vecteurs visibles
  {u} % 1e vecteur 2D
  {x}{y} % Coord. du 1e vecteur
  {v} % 2e vecteur 2D
  {x'}{y'}$ % Coord. du 2e vecteur
\makeatother
```

$$\begin{array}{cc} u & v \\ \left| \begin{array}{cc} x & x' \\ y & y' \end{array} \right| \end{array}$$

Exemple 2 – Décoration mais sans vecteur

En choisissant `novec` au lieu de `vec`, les vecteurs ne seront pas affichés.

```
\makeatletter
$\tns@det@plane@deco{novec} %
      {u}{x}{y} %
      {v}{x'}{y'}$
\makeatother
```

$$\left| \begin{array}{cc} x & x' \\ y & y' \end{array} \right|$$

Exemple 3 – Sans décoration mais avec les vecteurs

En utilisant `\tns@det@plane@no@deco`, la boucle fléchée ne sera pas imprimée. Voici une 1^{re} utilisation possible.

<pre> \makeatletter \$\tns@det@plane@no@deco{vec} % {u}{x}{y} % {v}{x'}{y'}\$ \makeatother </pre>	$\begin{array}{cc} u & v \\ \left \begin{array}{cc} x & x' \\ y & y' \end{array} \right \end{array}$
--	--

Exemple 4 – Sans décoration ni vecteur

<pre> \makeatletter \$\tns@det@plane@no@deco{novec} % {u}{x}{y} % {v}{x'}{y'}\$ \makeatother </pre>	$\begin{array}{cc} x & x' \\ \left \begin{array}{cc} x & x' \\ y & y' \end{array} \right \end{array}$
--	---

7 Historique

Nous ne donnons ici qu'un très bref historique récent ² de `tnscom` à destination de l'utilisateur principalement. Tous les changements sont disponibles uniquement en anglais dans le dossier `change-log` : voir le code source de `tnscom` sur `github`.

2020-07-10 Première version `0.0.0-beta`.

2. On ne va pas au-delà de un an depuis la dernière version.

8 Toutes les fiches techniques

8.1 Deux séparateurs d'arguments par défaut

8.1.1 Deux séparateurs d'arguments par défaut

`\tnsmathsep <macro>` (Sans argument)

`\tnsmathsubsep <macro>` (Sans argument)

8.1.2 Espace et fractions

`\frac <macro>` (2 Arguments)

`\dfrac <macro>` (2 Arguments)

`\stdfrac <macro>` (2 Arguments)

`\stdfrac <macro>` (2 Arguments)

— Argument 1: le numérateur.

— Argument 2: le dénominateur.

8.1.3 Espace et racines n-ièmes d'un réel

`\sqrt <macro>` [1 Option] (1 Argument)

`\stdsqrt <macro>` [1 Option] (1 Argument)

— Option: l'exposant à indiquer pour une racine n-ième.

— Argument: le radicande, c'est à dire ce qui sera écrit sous le radical.

8.2 En coulisse...

8.2.1 Nouvelle macro modifiant les arguments en amont de l'application d'une ancienne macro

`\tns@apply@macro@two@args <macro>` (4 Arguments)

— Argument 1: macro finale avec deux arguments et c'est tout.

— Argument 2: macro qui sera appliqué sur chacun des deux arguments avant appel la macro donné en argument 1.

— Arguments 3-4: ils sont là pour la définition abstraite mais en pratique ils ne seront pas utilisés. Ils correspondent aux arguments de la nouvelle macro fournie par `\tns@apply@macro@two@args`.

8.2.2 Macro avec un « multi-argument »

`\tns@multi@args <macro>` (5 Arguments)

— Argument 1: le séparateur de « *sous-arguments* ».

— Argument 2: le « multi-argument ».

— Argument 2: le matériel a ajouté avant.

— Argument 4: le matériel que l'on met à la place du séparateur donné en 1^{er} argument.

— Argument 5: le matériel a ajouté après.

8.2.3 Intervalles « généralisés »

`\tns@generic@interval@ext <macro> (5 Arguments)`

`\tns@generic@interval@semi@ext <macro> (5 Arguments)`

- Argument 1: le 1^{er} délimiteur qui est à gauche.
- Argument 2: le 1^{er} élément de l'intervalle « généralisé ».
- Argument 3: le séparateur entre les deux éléments.
- Argument 4: le 2^e élément de l'intervalle « généralisé ».
- Argument 5: le 2^{er} délimiteur qui est à droite.

8.2.4 Décorer un opérateur

`\tns@over@math@symbol <macro> (2 Arguments)`

- Argument 1: le texte à ajouter au-dessus.
- Argument 2: le symbole à décorer.

8.2.5 Produit en croix

`\tns@prop@prod <macro> (5 Arguments)`

- Argument 1: l'opérateur de multiplication à imprimer.
- Arguments 2-5: les 4 éléments du tableau de proportionnalité lus ligne par ligne.

8.2.6 Déterminant 2D ou produit en croix décoré

`\tns@det@plane@deco <macro> (7 Arguments)`

`\tns@det@plane@no@deco <macro> (7 Arguments)`

- Argument 1: `vec` ou `novec` suivant que l'on veut afficher ou non les vecteurs.
- Argument 2: le 1^{er} vecteur.
- Arguments 3-4: les coordonnées du 1^{er} vecteur.
- Argument 5: le 2^e vecteur.
- Arguments 6-7: les coordonnées du 2^e vecteur.