Le package `tnsana`

Code source disponible sur `https://github.com/typensee-latex/tnsana.git`.

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

Christophe BAL

2020-07-10

Table des matières

1 Introduction

Le package `tnsana` propose des macros pour écrire de l'analyse mathématique basique.

Remarque. Ce package s'appuie sur `tnscom` disponible sur `https://github.com/typensee-latex/tnscom.git`.

2 Constantes et paramètres

2.1 Constantes classiques

La liste complète

```
$\ggamma$ , $\ppi$ , $\ttau$ ,
$\ee$ , $\ii$ , $\jj$
et $\kk$ où $\ttau = 2 \ppi$
```

γ , π , τ , $\mathbf{e}$, $\mathbf{i}$, $\mathbf{j}$ et $\mathbf{k}$ où $\tau = 2\pi$

Remarque. Faites attention car `{\Large $\ppi \neq \pi$}` produit $\boldsymbol{\pi} \neq \pi$. Comme vous le constatez, les symboles ne sont pas identiques. Ceci est vraie pour toutes les constantes grecques.

2.2 Constantes latines personnelles

Exemple

La macro `\param` est surtout là pour une utilisation pédagogique.

```
$\param{a} x^2 + \param{b} x + \param{c}$
ou
$a x^2 + b x + c$
```

$\mathbf{a}x^2 + \mathbf{b}x + \mathbf{c}$ ou $ax^2 + bx + c$

3 La fonction valeur absolue

Exemple

```
$\abs{2}$ ,
$\abs {\dfrac{3}{5}}$ ou
$\abs*{\dfrac{3}{5}}$
```

$|2|$, $\left|\dfrac{3}{5}\right|$ ou $|\dfrac{3}{5}|$

Remarque. Le code LaTeX vient directement de ce poste : `https://tex.stackexchange.com/a/43009/6880`.

4 Fonctions nommées spéciales

4.1 Sans paramètre

Exemple

Quelques fonctions nommées supplémentaires où `fch` est pour `f`-rench soit *« français »* en anglais *(ce choix a été fait pour éviter des incompatibilités avec quelques autres packages)*. La liste complète des fonctions nommées est donnée un peu plus bas dans une section dédiée.

```
$\fch x \neq ch x$ ,
$\ppcm(x;y)$ ou
$\lg x$
```

$\fch x \neq ch x$, $\ppcm(x;y)$ ou $\lg x$

4.2 Avec un paramètre

Exemple

```
$\logb{2} x = \lg x$ ou
$\expb{6} y = 6^y$
```

$\logb{2} x = \lg x$ ou $\expb{6} y = 6^y$

4.3 Toutes les fonctions nommées en plus

`pgcd` : pgcd...
`ppcm` : ppcm...
`acos` : acos...
`asin` : asin...
`atan` : atan...
`arccosh` : arccosh...
`arcsinh` : arcsinh...
`arctanh` : arctanh...
`acosh` : acosh...
`asinh` : asinh...
`atanh` : atanh...
`fch` : ch...
`fsh` : sh...
`fth` : th...
`afch` : ach...
`afsh` : ash...
`afth` : ath...
`expb{p}` : $\exp_p$...
`logb{p}` : $\log_p$...

5 Calcul différentiel

5.1 Les opérateurs ∂ et d

Exemple 1

```
$\dd{t} = \dd[1]{t}$ ou $\dd[n]{x}$

$\pp{t} = \pp[1]{t}$ ou $\pp[n]{x}$
```

$\dd{t} = \dd[1]{t}$ ou $\dd[n]{x}$

$\pp{t} = \pp[1]{t}$ ou $\pp[n]{x}$

5.2 Dérivations totales d'une fonction – Version longue mais polymorphe

Exemple 1 – Différentes écritures possibles

La macro `\der` est stricte du point de vue sémantique car on doit lui fournir la fonction, l'ordre de dérivation et la variable de dérivation *(voir la section 5.3 qui présente la macro* `\sder` *permettant une rédaction efficace pour obtenir $f^{(1)}$ ou f')*. Voici plusieurs mises en forme faciles à taper via l'option de `\der`. Attention bien entendu à n'utiliser l'option par défaut `u` qu'avec un ordre de dérivation de valeur naturelle connue !

```
 $\der   {f}{3}{x}
= \der[e]{f}{3}{x}$

 $\der[i] {u}{k}{x}
= \der[f] {u}{k}{x}
= \der[sf]{u}{k}{x}$
```

$$f''' = f^{(3)}$$
$$\mathrm{d}_x^k u = \frac{\mathrm{d}^k u}{\mathrm{d}x^k} = \frac{\mathrm{d}^k u}{\mathrm{d}x^k}$$

On peut aussi ajouter autour de la fonction des parenthèses extensibles ou non. Ci-dessous on montre aussi une écriture du type *« opérateur fonctionnel »*.

```
 $\der[osf,sp]{\frac{1}{2} uv}{k}{x}
= \der[of,p]  {\dfrac{1}{2} uv}{k}{x}$
```

$$\frac{\mathrm{d}^k}{\mathrm{d}x^k}\left(\frac{1}{2}uv\right) = \frac{\mathrm{d}^k}{\mathrm{d}x^k}\left(\frac{1}{2}uv\right)$$

Remarque. Expliquons les valeurs des options.

1. `u`, la valeur par défaut, est pour `u`-suel soit l'écriture avec les primes. Cette option ne marchera pas avec un nombre symbolique de dérivations.
2. `e` est pour `e`-xposant.
3. `i` est pour `i`-ndice.
4. `f` est pour `f`-raction avec aussi `sf` pour une écriture réduite où `s` est pour `s`-mall soit *« petit »* en anglais.
5. `of` et `osf` utilisent le préfixe `o` pour `o`-pérateur.
6. `p` est pour `p`-arenthèse : dans ce cas les parenthèses seront extensibles.
7. `sp` est pour des parenthèses non extensibles.

Exemple 2 – Pas de uns inutiles

```
 $\der[i ]{u}{1}{x}
= \der[f ]{u}{1}{x}
= \der[sf]{u}{1}{x}
= \der[of]{u}{1}{x}$
```

$$\mathrm{d}_x u = \frac{\mathrm{d}u}{\mathrm{d}x} = \frac{\mathrm{d}u}{\mathrm{d}x} = \frac{\mathrm{d}}{\mathrm{d}x}u$$

Remarque. Voici comment forcer les exposants 1 si besoin.

```
 $\der[i ]{u}{\,\!1}{x}
= \der[f ]{u}{\,\!1}{x}
= \der[sf]{u}{\,\!1}{x}
= \der[of]{u}{\,\!1}{x}$
```

$$\mathrm{d}_x^1 u = \frac{\mathrm{d}^1 u}{\mathrm{d}x^1} = \frac{\mathrm{d}^1 u}{\mathrm{d}x^1} = \frac{\mathrm{d}^1}{\mathrm{d}x^1}u$$

5.3 Dérivations totales d'une fonction – Version courte pour les écritures standard

Dans l'exemple suivant le code manque de sémantique car on n'indique pas la variable de dérivation. Ceci étant dit à l'usage la macro `\sder` rend de grands services. Ici le préfixe `s` est pour `s`-imple voire `s`-impliste... Voici des exemples où de nouveau l'option par défaut `u` ne sera fonctionnelle qu'avec un ordre de dérivation de valeur naturelle connue !

```
 $\sder{f}{1} = \der{f}{1}{x}$

 $\sder{f}{1}
= \sder[e]{f}{1}$

 $\sder[sp ]{\dfrac{1}{2} uv}{2}
= \sder[e,p]{\dfrac{1}{2} uv}{2}$
```

$$f' = f'$$
$$f' = f^{(1)}$$
$$(\frac{1}{2}uv)'' = \left(\frac{1}{2}uv\right)^{(2)}$$

Remarque. Ici les seules options disponibles sont `u`, `e`, `p` et `sp`.

5.4 L'opérateur de dérivation totale

Ce qui suit peut rendre service au niveau universitaire. Les options possibles sont `f`, valeur par défaut, `sf` et `i` avec les mêmes significations que pour la macro `\der`.

```
 $\derope    {k}{x}
= \derope[sf]{k}{x}
= \derope[i] {k}{x}$
```

$$\frac{\mathrm{d}^k}{\mathrm{d}x^k} = \frac{\mathrm{d}^k}{\mathrm{d}x^k} = \mathrm{d}_x^k$$

Ici non plus il n'y a pas de uns inutiles.

```
 $\derope    {1}{x}
= \derope[sf]{1}{x}
= \derope[i] {1}{x}$
```

$$\frac{\mathrm{d}}{\mathrm{d}x} = \frac{\mathrm{d}}{\mathrm{d}x} = \mathrm{d}_x$$

5.5 Dérivations partielles

Exemple 1 – Différentes écritures possibles

La macro `\pder` [1] avec `p` pour `p`-artielle permet de rédiger des dérivées partielles en utilisant facilement plusieurs mises en forme via une option qui vaut `f` par défaut. Cette macro attend une fonction, les dérivées partielles effectuées et l'ordre total de dérivation. Voici les deux types de mise en forme où vous noterez comment `x | y^2` est interprété.

```
 $\pder    {f}{x | y^2}{3}
= \pder[sf]{f}{x | y^2}{3}$
ou
 $\pder[i] {f}{x | y^2}{3}$
```

$$\frac{\partial^3 f}{\partial x\,\partial y^2} = \frac{\partial^3 f}{\partial x\,\partial y^2} \text{ ou } \partial^3_{x,y(2)} f$$

On peut aussi ajouter autour de la fonction des parenthèses extensibles ou non. Ci-dessous on montre aussi une écriture du type *« opérateur fonctionnel »*.

```
 $\pder[of] {f}{x | y^2}{}
= \pder[osf]{f}{x | y^2}{}$
ou
 $\pder[i,sp]{u + v}{x | y^2}{}$
```

$$\frac{\partial}{\partial x\,\partial y^2} f = \frac{\partial}{\partial x\,\partial y^2} f \text{ ou } \partial_{x,y(2)}(u+v)$$

Remarque. Les options disponibles sont `f`, `sf`, `of`, `osf`, `i`, `p` et `sp` avec des significations similaires à celles pour la macro `\der`.

1. `\partial` existe déjà pour obtenir ∂.

Exemple 2 – Pas de uns inutiles

```
 $\pder    {u}{x}{1}
= \pder[sf]{u}{x}{1}
= \pder[i] {u}{x}{1}$
```

$$\frac{\partial u}{\partial x} = \frac{\partial u}{\partial x} = \partial_x u$$

5.6 L'opérateur de dérivation partielle

Ce qui suit peut rendre service au niveau universitaire. Les options possibles sont `f`, valeur par défaut, `sf` et `i` avec les mêmes significations que pour la macro `\pder`.

```
 $\pderope    {x | y^2}{3}
= \pderope[sf]{x | y^2}{3}
= \pderope[i] {x | y^2}{3}$
```

$$\frac{\partial^3}{\partial x\,\partial y^2} = \frac{\partial^3}{\partial x\,\partial y^2} = \partial^3_{x,y(2)}$$

5.7 Tableaux de variation et de signe

5.7.1 Les bases

Comment ça marche ?

Tout le boulot est fait par le package `tkz-tab` auquel on impose le choix d'une pointe de flèche plus visible via le réglage suivant `\tkzTabSetup[arrowstyle = triangle 60]`.

Nous donnons quelques exemples classiques d'utilisation proches ou identiques de certains proposés dans la documentation de `tkz-tab` *(les codes ont été mis en forme pour faciliter la compréhension de la syntaxe à suivre)*. Reportez vous à la documentation de `tkz-tab` pour des compléments d'information : vous y trouverez des réglages très fins.

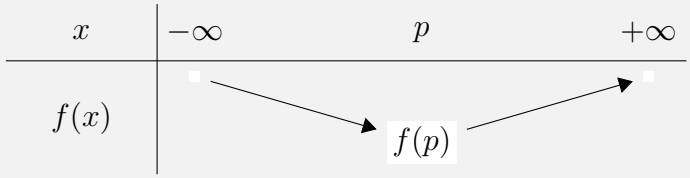
Exemple 1 – Avec des signes

```
\begin{tikzpicture}
    \tkzTabInit{
        $x$       / 0.75 , % Facteur d'échelle de 0.75 pour la hauteur de la 1er ligne.
        $\cos(x)$ / 1      % Facteur d'échelle de  1   pour la hauteur de la 2e ligne.
    }{
                $0$     , $\frac{\pi}{2}$     , $\pi$  % Valeurs utiles de x.
    }
    \tkzTabLine{    , + , z               , - ,      } % Signes et zéro.
\end{tikzpicture}
```

x	0		$\frac{\pi}{2}$		π
$\cos(x)$		$+$	0	$-$	

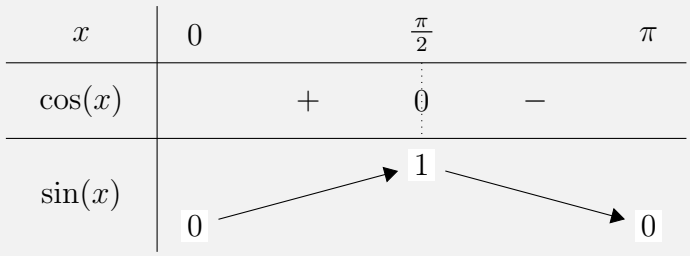
Exemple 2 – Avec des variations (sans cadre)

```
\begin{tikzpicture}
    \tkzTabInit[nocadre]{
        $x$    / 0.75 ,
        $f(x)$ / 1.5
    }{
              $-\infty$ , $p$        , $+\infty$
    }
    \tkzTabVar{+ /       , - / $f(p)$ , + /      } % Variations via position / valeur.
\end{tikzpicture}
```



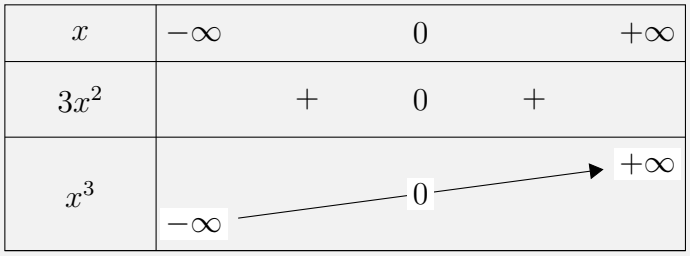
Exemple 3 – Variations via une dérivée (sans cadre)

```
\begin{tikzpicture}
    \tkzTabInit[nocadre]{
        $x$       / 0.75 ,
        $\cos(x)$ / 1    ,
        $\sin(x)$ / 1.5
    }{
             $0$     , $\frac{\pi}{2}$     , $\pi$
    }
    \tkzTabLine{    , + , z               , - ,      }
    \tkzTabVar {- / 0   , + / 1               , - / 0}
\end{tikzpicture}
```



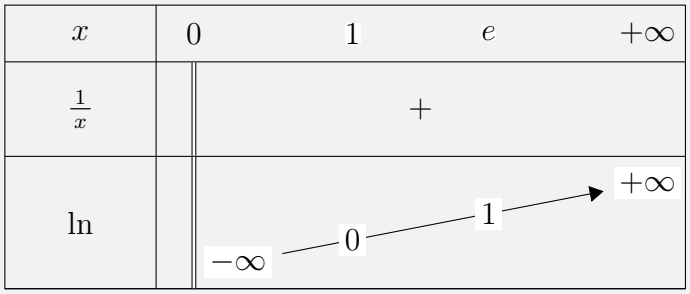
Exemple 4 – Une image intermédiaire avec une seule flèche

```
\begin{tikzpicture}
    \tkzTabInit{
        $x$     / 0.75 ,
        $3 x^2$ / 1    ,
        $x^3$   / 1.5
    }{
                $-\infty$          , $0$      , $+\infty$
    }
    \tkzTabLine{               , + , 0    , + ,               }
    \tkzTabVar {- / $-\infty$      , R        , + / $+\infty$}
    %
    \tkzTabIma{1}{3}{2} % Position entre les 1re et 3e valeurs puis rang relatif.
              {$0$}     % Valeur de l'image.
\end{tikzpicture}
```



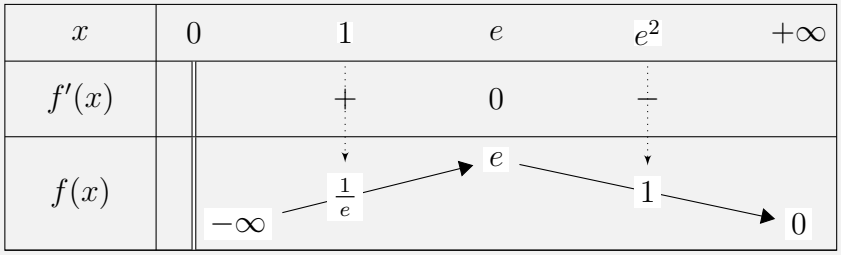
Exemple 5 – Valeurs interdites et valeurs supplémentaires

```
\begin{tikzpicture}
    \tkzTabInit[espcl = 6]{    % Largeur entre les valeurs du tableau.
        $x$            / 0.75 ,
        $\frac{1}{x}$ / 1.25 ,
        $\ln$          / 1.75
    }{
                $0$                , $+\infty$
    }
    \tkzTabLine{d               , + ,              }
    \tkzTabVar {D- / $-\infty$     , + / $+\infty$}
    %
    \tkzTabVal{1}{2}{0.35} % Position entre les 1re et 2e valeurs puis en proportion.
              {1}{0}       % x_1 et f(x_1)
    \tkzTabVal{1}{2}{0.65} % Position entre les 1re et 2e valeurs puis en proportion.
              {$e$}{1}   % x_2 et f(x_2)
\end{tikzpicture}
```



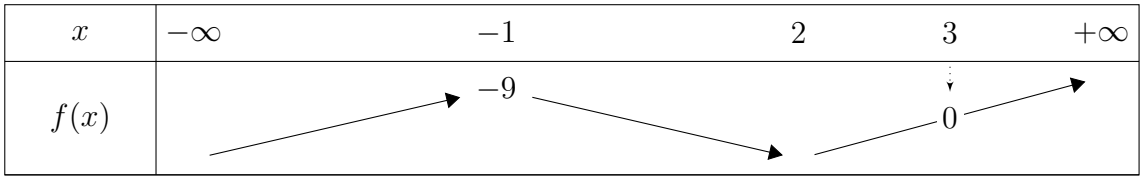
Voici un autre exemple pour comprendre comment utiliser `\tkzTabVal` avec en plus l'option `draw` qui peut rendre service.

```
\begin{tikzpicture}
    \tkzTabInit[espcl = 4]{
        $x$     / 0.75 ,
        $f'(x)$ / 1    ,
        $f(x)$  / 1.5
    }{
                $0$                 , $e$          , $+\infty$
    }
    \tkzTabLine{d               , + , 0          , - ,          }
    \tkzTabVar {D- / $-\infty$     , + / $e$     , - / $0$  }
    %
    \tkzTabVal[draw]{1}{2}{0.5} % Position entre les 1re et 2e valeurs au milieu.
                    {$1$}{$\frac{1}{e}$}
    \tkzTabVal[draw]{2}{3}{0.5} % Position entre les 2e et 3e valeurs au milieu.
                    {$e^2$}{$1$}
\end{tikzpicture}
```



Exemple 6 – Signe à partir des variations (un peu de pédagogie...)

Il est assez facile de produire des choses très utiles pédagogiquement comme ce qui suit en se salissant un peu les mains avec du code TikZ.



On en déduit le signe de la fonction f.

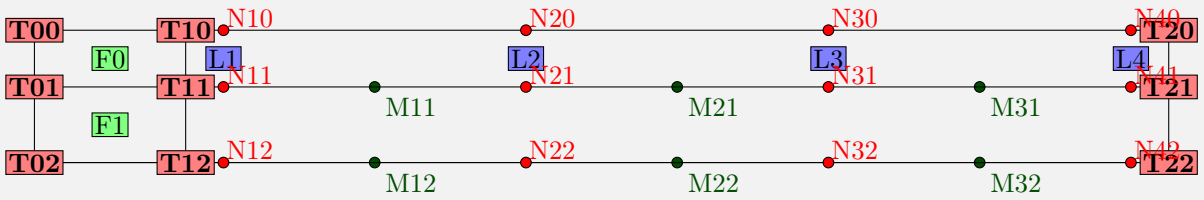
x	$-\infty$			3		$+\infty$
$f(x)$		$-$		0	$+$	

Voici le code du 1er tableau où le placement de la valeur 3 au milieu entre 2 et $+\infty$ va nous simplifier le travail pour le 2e tableau.

```
\begin{tikzpicture}
    \tkzTabInit[espcl = 4]{
        $x$    / 0.75 ,
        $f(x)$ / 1.5
    }{
                $-\infty$ , $-1$     , $2$ , $+\infty$
    }
    \tkzTabVar{- /       , + / $-9$ , - / , + /      }
    %
    \tkzTabVal[draw]{3}{4}{0.5} % Position entre les 3e et 4e valeurs au milieu.
              {$3$}{$0$}
\end{tikzpicture}
```

Pour produire lee code du 2e tableau, il a été utilisé au préalable ce qui suit en activant l'option `help` qui demande à `tkz-tab` d'afficher les noms de noeuds au sens TikZ qui ont été créés. Ceci permet alors d'utiliser ces noeuds pour des dessins TikZ faits maison [2].

```
\begin{tikzpicture}
    \tkzTabInit[
        espcl = 4,
        help       % <-- Pour voir les noms des noeuds.
    ]{
        $x$    / 0.75 ,
        $f(x)$ / 1
    }{
       $-\infty$ , $-1$ , $2$ , $+\infty$
    }
\end{tikzpicture}
```



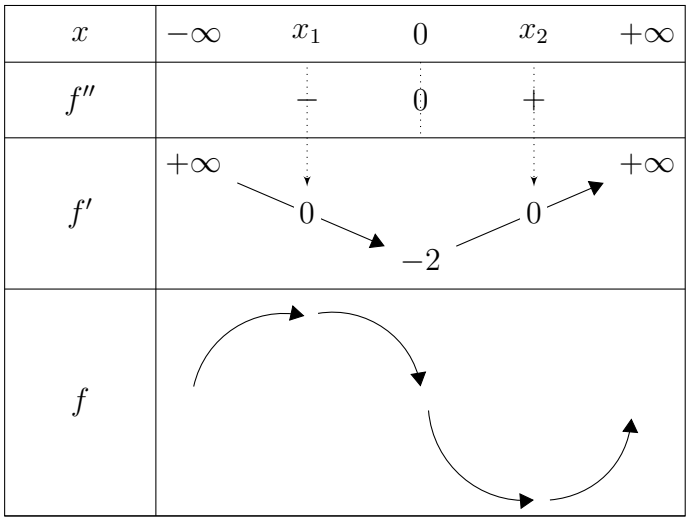
Maintenant que ls noms des noeuds sont connus, il devient facile de produire le code ci-après. Bien noter l'usage de valeurs utiles « vides » de x ainsi que les mystiques `\node at ($(A)!0.5!(B)$)` permettant de placer un noeud au milieu entre les deux noeuds `A` et `B`.

2. C'est grâce à ce mécanisme que `lymath` peut proposer des outils explicatifs des tableaux de signe : voir la section suivante.

```
\begin{tikzpicture}
    \tkzTabInit[
        espcl = 4,
%        help       % <-- Pour voir les noms des noeuds.
    ]{
        $x$    / 0.75 ,
        $f(x)$ / 1
    }{
        $-\infty$ , , , $+\infty$ % ATTENTION ! Ne pas oublier de virgules.
    }
    %
    \node at ($(N31)!0.5!(N40)$){$3$};
    \node at ($(M31)!0.5!(M32)$){$0$};
    \node at ($(M31)!0.5!(N42)$){$+$};
    \node at ($(N11)!0.5!(M32)$){$-$};
\end{tikzpicture}
```

Exemple 7 – Convexité et concavité symbolisées dans les variations

Voici un autre exemple s'utilisant la machinerie TikZ afin d'indiquer dans les variations la convexité et la concavité via des flèches incurvées *(cette convention est proposée dans la sous-section* « Exemple utilisant l'option `\help` » *de la section* « Gallerie » *de la documentation de* ***tkz-tab****)*.



Le code utilisé est le suivant.

```
\begin{tikzpicture}
    \tkzTabInit[
        espcl = 3,
%        help       % <-- Pour voir les noms des noeuds.
    ]{
        $x$   / 0.75 ,
        $f''$ / 1    ,
        $f'$  / 2    ,
        $f$   / 3
    }{
              $-\infty$          , $0$          , $+\infty$
    }
    \tkzTabLine{               , - , z        , + ,              }
    \tkzTabVar {+ / $+\infty$      , - / $-2$     , + / $+\infty$}

    \tkzTabVal[draw]{1}{2}{.5}{$x_1$}{$0$}
    \tkzTabVal[draw]{2}{3}{.5}{$x_2$}{$0$}

    \begin{scope}[->, > = triangle 60]
        \coordinate (Middle1) at ($(N13)!0.5!(N14)$);
        \coordinate (Middle2) at ($(N23)!0.5!(N24)$);
        \coordinate (Middle3) at ($(N33)!0.5!(N34)$);
        \path (N13) -- (N23) node[midway,below=6pt](N){};
        %
        \draw ([above=6pt]Middle1)
              to [bend left=45] ([left=1pt]N);
        \draw ([right=3pt]N)
              to [bend left=45] ([above=6pt]Middle2) ;
        \draw ([below right=3pt]Middle2)
              to [bend left=-45] ([above=6pt]M24) ;
        \draw ([above right=6pt]M24)
              to [bend right=40] ([below left=6pt]Middle3);
    \end{scope}
\end{tikzpicture}
```

5.7.2 Décorer facilement un tableau

Motivation

Considérons le tableau suivant et imaginons que nous voulions l'expliquer à un débutant.

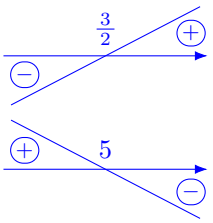
x	$-\infty$		$\frac{3}{2}$		5		$+\infty$
Signe de $2x-3$		$-$	0	$+$		$+$	
Signe de $-x+5$		$+$		$+$	0	$-$	
Signe de $(2x-3)(-x+5)$		$-$	0	$+$	0	$-$	

Deux options s'offrent à nous pour justifier comment a été rempli le tableau.

1. Classiquement on résout par exemple juste les deux inéquations $2x-3>0$ et $-x+5>0$ puis on complète les deux premières lignes [3] pour en déduire la dernière via la règle des signes d'un produit.
2. On peut proposer une méthode moins sujette à la critique qui s'appuie sur la représentation graphique d'une fonction affine en produisant le tableau suivant.

x	$-\infty$		$\frac{3}{2}$		5		$+\infty$
Signe de $2x-3$		$-$	0	$+$		$+$	
Signe de $-x+5$		$+$		$+$	0	$-$	
Signe de $(2x-3)(-x+5)$		$-$	0	$+$	0	$-$	

← Valeurs utiles de x



← Signe d'un produit.

Pour produire le 2e tableau, en plus du code `tkz-tab` pour le tableau de signe qui utilise les réglages optionnels `lgt = 3.5` et `espcl = 2.5` de `\tkzTabInit` [4], il a fallu ajouter les lignes données ci-dessous où sont utilisées les macros `\graphSign` et `\comLine` proposées par `lymath` *(la syntaxe simple à suivre sera expliquée dans la section suivante)*.

```
\begin{tikzpicture}
    % ------------------------------------------------------ %
    % -- Code tkz-tab pour les signes non reproduit ici -- %
    % ------------------------------------------------------ %
    \comLine[gray]{0}{$\leftarrow$ Valeurs utiles de $x$}

    \graphSign{1}{ax+b, ap}{$\frac{3}{2}$}
    \graphSign{2}{ax+b, an}{$5$}

    \comLine[gray]{3}{$\leftarrow$ Signe d'un produit.}
\end{tikzpicture}
```

Remarque. Les lignes pour les signes doivent utiliser un coefficient minimal de `1.5` pour la hauteur afin d'éviter que la superposition des graphiques.

Commenter une ligne

L'ajout de commentaires courts se fait via la macro `\comLine` pour `com`-ment a line soit *« commenter une ligne »* en anglais [5]. Cette macro possède un argument optionnel et deux obligatoires.

3. Notons que cette approche est un peu scandaleuse car il faudrait en toute rigueur aussi résoudre $2x-3<0$, $-x+5<0$, $2x-3=0$ et $-x+5=0$. Personne ne le fait car l'on pense aux variations d'une fonction affine. Dans ce cas pourquoi ne pas juste utiliser ce dernier argument ? C'est ce que propose la 2e méthode.

4. Ceci permet d'avoir de la place dans la 1re colonne pour le dernier produit et de réduire la largeur des colonnes pour les signes.

5. L'auteur de `lymath` n'est absolument pas un fan de la casse en bosses de chameau mais par souci de cohérence avec ce que propose `tkz-tab`, le nom `\comLine` a été proposé à la place de `\comline`.

1. ***L'argument optionnel : choix de la couleur du texte.***

   Ci-dessus, nous avons utilisé `\comLine[gray]{0}{...}` pour avoir un texte en gris.

2. ***Le 1er argument : numéro de ligne.***

   Par convention 0 est le numéro de la toute 1re ligne contenant les valeurs utiles de la variable. `\comLine{3}{...}` correspond donc à la 3e ligne de signes ou moins intuitivement à la (3+1)e ligne pour un humain non codeur.

3. ***Le 2^{e} argument : texte du commentaire.***

   Par défaut aucun retour à la ligne n'est possible. Si besoin se reporter à la page 17 où est montré comment écrire sur plusieurs lignes *(voir le tout dernier exemple de cette section)*.

Graphiques pour expliquer des signes

Pour le moment, la macro `\graphSign` propose deux types de graphiques[6]. Rappelons au passage que la convention est de prendre 0 pour numéro de la toute 1re ligne contenant les valeurs utiles de la variable.

1. **Fonctions affines non constantes.**

   Pour les fonctions du type $f(x) = ax + b$ avec $a \neq 0$, nous devons connaître le signe de a et la racine r de f.

   Le codage est assez simple. Par exemple, `\graphSign{2}{ax+b, an}{$5$}` indique pour la 2e ligne d'ajouter le graphique d'une fonction affine, ce qu'indique le code `ax+b` sans espace, avec la condition $a < 0$ via `an` pour `a négatif`, et enfin avec 5 pour racine.

   Donc si l'on veut ajouter pour la 4e ligne de signe le graphique de $f(x) = 3x$, on utilisera dans ce cas `\graphSign{4}{ax+b, ap}{$0$}` où `ap` pour `a positif` code la condition $a > 0$.

2. **Fonctions trinômiales du 2e degré.**

   Pour les fonctions du type $f(x) = ax^2 + bx + c$ avec $a \neq 0$, nous devons connaître le signe de a, celui du discriminant $\Delta = b^2 - 4ac$, ce dernier pouvant être nul, et les racines réelles éventuelles du trinôme f.

   Voici comment coder ceci. Par exemple `\graphSign{5}{ax2+bx+c, an, dp}{$r_1$}{$r_2$}` indique d'ajouter dans la 5e ligne de signe le graphique d'un trinôme du 2e degré via le code `ax2+bx+c` sans espace, avec les conditions $a < 0$ et $\Delta > 0$ via `an` et `dp`, le trinôme ayant r_1 et r_2 pour racines réelles.

   En plus de `dn` et `dp`, il y a `dz` pour `discriminant zéro`. Ainsi pour indiquer dans la 3e ligne de signe la courbe relative à $f(x) = -4x^2$, on utilisera `\graphSign{3}{ax2+bx+c, an, dz}{$0$}`.

   Enfin le graphique associé au trinôme $f(x) = 7x^2 + 3$, qui est sans racine réelle, s'obtiendra dans la 4e ligne de signe via `\graphSign{4}{ax2+bx+c, ap, dn}`.

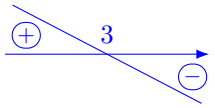
Exemple 1 – Un exemple avec une parabole

Il devient très facile de proposer un tableau décoré comme le suivant.

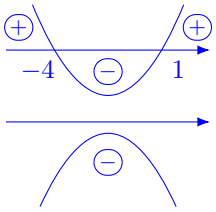
6. Le choix de la casse en bosses de chameau a été expliqué pour la macro `\comLine`.

x	$-\infty$		-4		1		3		$+\infty$
Signe de $-x+3$		$+$		$+$		$+$	0	$-$	
Signe de $f(x)$		$-$	0	$+$	0	$+$	0	$-$	
Signe de x^2+3x-4		$-$	0	$+$	0	$-$		$-$	
Signe de $-x^2+x-4$		$-$		$-$		$-$		$-$	
Signe du produit		$-$	0	$-$	0	$+$	0	$+$	

Schémas



Voir Q.1-a)



← Conclusion

En plus des deux exemples de schémas de paraboles, il faut noter dans le code supplémentaire ajouté l'utilisation de `\kern1.75em` dans `\comLine[gray]{0}{\kern1.75em Schémas}` afin de mettre un espace horizontal précis pour centrer à la main le texte *« Schémas »*.

```
\begin{tikzpicture}
    % ------------------------------------------------------ %
    % -- Code tkz-tab pour les signes non reproduit ici -- %
    % ------------------------------------------------------ %
    \comLine[gray]{0}{\kern1.75em Schémas}

    \graphSign{1}{ax+b, an}{$3$}
    \comLine  {2}{Voir Q.1-a)}
    \graphSign{3}{ax2+bx+c, ap, dp}{$-4$}{$1$} % Deux racines réelles.
    \graphSign{4}{ax2+bx+c, an, dn}            % Aucune racine réelle.

    \comLine[gray]{5}{$\leftarrow$ Conclusion}
\end{tikzpicture}
```

Exemple 2 – Commenter des variations

Pour finir, indiquons que les outils de décoration marchent aussi pour les tableaux de variation. Voici un exemple possible d'utilisation où le retour à la ligne a été obtenue affreusement, ou pas, via `\parbox{11.5em}{Les limites sont hors programme pour cette année.}`.

x	0		e		$+\infty$
$f'(x)$	‖	$+$	0	$-$	
$f(x)$	‖ $-\infty$	↗	e	↘	0

Les limites sont hors programme pour cette année.

6 Calcul intégral

6.1 Intégrales multiples

Commençons par un point important : le package réduit les espacements entres des symboles $\int$ successifs. Voici un exemple.

```
$\displaystyle
 \int \int \int
 F(x;y;z) \dd{x} \dd{y} \dd{z}$

$\displaystyle
 \int_{a}^{b} \int_{c}^{d} \int_{e}^{f}
 F(x;y;z) \dd{x} \dd{y} \dd{z}$
```

$$\displaystyle \int \int \int F(x\,;y\,;z)\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z$$
$$\displaystyle \int_{a}^{b} \int_{c}^{d} \int_{e}^{f} F(x\,;y\,;z)\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z$$

Remarque. Par défaut, LaTeX affiche $\int \int \int F(x\,;y\,;z)\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z$ et $\int_{a}^{b} \int_{c}^{d} \int_{e}^{f} F(x\,;y\,;z)\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z$. Nous avons obtenu ce résultat en utilisant `\stdint` qui est l'opérateur proposé de façon standard par LaTeX.

6.2 Un opérateur d'intégration clés en main

Exemple 1 – À quoi bon ?

Le 1er exemple qui suit semblera être une hérésie pour les habitués de LaTeX mais rappelons que le but de `lymath` est de rendre les documents facilement modifiables globalement ou localement comme le montre le 2e exemple.

```
 $\displaystyle
  \integrate{a}{b}{f(x)}{x}
= \int_{x=a}^{x=b} f(x) \dd{x}$

 $\displaystyle
  \integrate*{a}{b}{f(x)}{x}
= \integrate{a}{b}{f(x)}{x}$
```

$$\displaystyle \int_{x=a}^{x=b} f(x)\,\mathrm{d}x = \int_{x=a}^{x=b} f(x)\,\mathrm{d}x$$
$$\displaystyle \int_{a}^{b} f(x)\,\mathrm{d}x = \int_{x=a}^{x=b} f(x)\,\mathrm{d}x$$

Exemple 2 – Le mode `displaystyle`

La macro `\dintegrate*` présentée ci-dessous possède aussi une version non étoilée `\dintegrate`.

```
 $\dintegrate*{a}{b}{f(x)}{x}
= \integrate*{a}{b}{f(x)}{x}$
```

$$\displaystyle \int_{a}^{b} f(x)\,\mathrm{d}x = \int_{a}^{b} f(x)\,\mathrm{d}x$$

6.3 L'opérateur crochet

Exemple 1

```
 $\hook{a}{b}{F(x)}{x}
= F(b) - F(a)$

 $\dintegrate*{a}{b}{f(x)}{x}
= \hook*{a}{b}{F(x)}{x}$
```

$$\left[F(x) \right]_{x=a}^{x=b} = F(b) - F(a)$$
$$\int_a^b f(x)\,\mathrm{d}x = \left[F(x) \right]_a^b$$

Remarque. Il faut savoir que `\hook` signifie *« crochet »* en anglais mais la bonne traduction du terme mathématique est en fait *« square bracket »*. Ceci étant dit l'auteur de `lymath` trouve plus efficace d'utiliser `\hook` comme nom de macro.

Exemple 2 – Des crochets non extensibles

Dans l'exemple suivant, on utilise l'option `sb` pour `s`-mall `b`-rackets soit *« petits crochets »* en anglais. Les options sont disponibles à la fois pour `\hook` et `\hook*`.

```
 $\hook*{a}{b}%
       {\dfrac{x - 1}{5 + x^2}}{x}
= \hook*[sb]%
       {a}{b}%
       {\dfrac{x - 1}{5 + x^2}}{x}$
```

$$\left[\frac{x-1}{5+x^2} \right]_a^b = \left[\frac{x-1}{5+x^2} \right]_a^b$$

Exemple 3 – Un trait vertical épuré

Via les options `r` et `sr` pour `s`-mall et `r`-ull soit *« petit »* et *« trait »* en anglais, on obtient ce qui suit.

```
 $\hook[r] {a}{b}%
          {\dfrac{x - 1}{5 + x^2}}{x}
= \hook[sr]{a}{b}%
          {\dfrac{x - 1}{5 + x^2}}{x}$
```

$$\frac{x-1}{5+x^2} \Bigg|_{x=a}^{x=b} = \frac{x-1}{5+x^2} \Big|_{x=a}^{x=b}$$

7 Historique

Nous ne donnons ici qu'un très bref historique récent [7] de `tnsana` à destination de l'utilisateur principalement. Tous les changements sont disponibles uniquement en anglais dans le dossier `change-log` : voir le code source de `tnsana` sur `github`.

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

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

8 Toutes les fiches techniques

8.1 Constantes et paramètres

8.1.1 Constantes classiques

`\ggamma <macro> (Sans argument)`
`\ppi <macro> (Sans argument)`
`\ttau <macro> (Sans argument)`
`\ee <macro> (Sans argument)`
`\ii <macro> (Sans argument)`
`\jj <macro> (Sans argument)`
`\kk <macro> (Sans argument)`

8.1.2 Constantes latines personnelles

`\param <macro> (1 Argument)`

— `Argument`: un texte utilisant l'alphabet latin.

8.2 La fonction valeur absolue

8.2.1 Valeur absolue

`\abs <macro> (1 Argument)`
`\abs* <macro> (1 Argument)`

— `Argument`: l'expression sur laquelle appliquer la fonction valeur absolue.

8.3 Fonctions nommées spéciales

8.3.1 Fonctions nommées sans paramètre

`\pgcd <macro> (Sans argument)`
`\ppcm <macro> (Sans argument)`

`\acos <macro> (Sans argument)`
`\asin <macro> (Sans argument)`
`\atan <macro> (Sans argument)`

`\arccosh <macro> (Sans argument)`
`\arcsinh <macro> (Sans argument)`
`\arctanh <macro> (Sans argument)`

`\acosh <macro> (Sans argument)`
`\asinh <macro> (Sans argument)`
`\atanh <macro> (Sans argument)`

`\fch <macro> (Sans argument)` où `f = f-rench`
`\fsh <macro> (Sans argument)` où `f = f-rench`
`\fth <macro> (Sans argument)` où `f = f-rench`

8.3.2 Fonctions nommées avec un paramètre

`\expb <macro> (1 Argument)`

— `Argument:` la base de l'exponentielle

`\logb <macro> (1 Argument)`

— `Argument:` la base du logarithme

8.4 Calcul différentiel

8.4.1 Calcul différentiel

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

— `Option:` utilisée, cette option sera mise en exposant du symbole ∂ ou d.

— `Argument:` la variable de différentiation à droite du symbole ∂ ou d.

8.4.2 Dérivation totale

`\der <macro> [1 Option] (3 Arguments)`

— `Option:` la valeur par défaut est `u`.

1. `u` : écriture usuelle avec des primes *(ceci nécessite d'avoir une valeur entière naturelle connue du nombre de dérivations successives)*.
2. `e` : écriture via un exposant entre des parenthèses.
3. `i` : écriture via un indice.
4. `f` : écriture via une fraction en mode display.
5. `sf` : écriture via une fraction en mode non display.
6. `of` : écriture via une fraction en mode display sous la forme d'un opérateur *(la fonction est à côté de la fraction)*.
7. `osf` : écriture via une fraction en mode non display sous la forme d'un opérateur *(la fonction est à côté de la fraction)*.
8. `p` : ajout de parenthèses extensibles autour de la fonction.
9. `sp` : ajout de parenthèses non extensibles autour de la fonction.

— `Argument 1:` la fonction à dériver.

— `Argument 2:` l'ordre de dérivation.

— `Argument 3:` la variable de dérivation.

`\sder <macro> [1 Option] (2 Arguments)` où `s = s-imple`

— `Option:` la valeur par défaut est `u`. Les options disponibles sont `u`, `e`, `p` et `sp` : voir la fiche technique de `\sder` ci-dessus.

— `Argument 1:` la fonction à dériver.

— `Argument 2:` l'ordre de dérivation.

8.4.3 Dérivation totale – Opérateur fonctionnel

`\derope <macro> [1 Option] (2 Arguments)` où `ope = ope-rator`

— `Option:` la valeur par défaut est `f`. Les options disponibles sont `f`, `sf` et `i` : voir la fiche technique de `\der` donnée un peu plus haut.

— `Argument 1:` la fonction à dériver.

— `Argument 2:` l'ordre de dérivation.

8.4.4 Dérivation partielle

`\pder <macro> [1 Option] (2 Arguments)` où `p = p-artial`

— `Option:` la valeur par défaut est `f`.

1. `f` : écriture via une fraction en mode display.
2. `sf` : écriture via une fraction en mode non display.
3. `of` : écriture via une fraction en mode display sous la forme d'un opérateur *(la fonction est à côté de la fraction)*.
4. `osf` : écriture via une fraction en mode non display sous la forme d'un opérateur *(la fonction est à côté de la fraction)*.
5. `i` : écriture via un indice.
6. `p` : ajout de parenthèses extensibles autour de la fonction.
7. `sp` : ajout de parenthèses non extensibles autour de la fonction.

— `Argument 1:` la fonction à dériver.

— `Argument 2:` les variables utilisées avec leur ordre de dérivation pour la dérivation partielle en utilisant une syntaxe du type `x | y^2 | ...` qui indique de dériver suivant x une fois, puis suivant y trois fois... etc.

— `Argument 3:` l'ordre total de dérivation.

8.4.5 Dérivation partielle - Opérateur fonctionnel

`\pderope <macro> [1 Option] (2 Arguments)` où `p = p-artial` et `ope = ope-rator`

— `Option:` la valeur par défaut est `f`. Les options disponibles sont `f`, `sf` et `i` : voir la fiche technique de `\pder` juste avant.

— `Argument 1:` les variables utilisées avec leur ordre de dérivation via la syntaxe indiquée ci-dessus.

— `Argument 2:` l'ordre total de dérivation.

8.4.6 Tableaux de signes – Commentaires et graphiques explicatifs

`\comLine <macro> [1 Option] (2 Arguments)` où `com = com-ment`

— `Option:` couleur au format TikZ.

— `Argument 1:` le numéro de ligne où placer le commentaire, 0 étant le 1$^{\text{er}}$ numéro.

— `Argument 2:` le texte du commentaire.

`\graphSign <macro> [1 Option] (2..4 Arguments)`

— `Option:` couleur au format TikZ.

— `Argument 1:` le numéro de ligne où placer le graphique, 0 étant le 1er numéro.

— `Argument 2:` le type de fonctions avec des contraintes en utilisant la virgule comme séparateur d'informations.

1. `ax+b` sans espace indique une fonction affine avec un unique paramètre `a` non nul à caractériser.
2. `ax2+bx+c` sans espace indique une fonction trinôme du 2e degré avec une paramètre `a` non nul à caractériser ainsi que `d` pour son discriminant.
3. `ap` et `an` indiquent respectivement les conditions $a > 0$ et $a < 0$.
4. `dp`, `dz` et `dn` indiquent respectivement les conditions $d > 0$, $d = 0$ et $d < 0$.

— `Argument supplémentaire pour ax+b:` la racine de $ax + b$.

— `Arguments supplémentaires pour ax2+bx+c:` si $ax^2 + bx + c$ admet une ou deux racines, on donnera toutes les racines de la plus petite à la plus grande [8].

8.5 Calcul intégral

8.5.1 Intégration – Le symbole standard

```
\stdint <macro> (Sans argument)
```

8.5.2 Intégration – Fonctionnelle d'intégration

```
\integrate <macro> (4 Arguments)
\integrate* <macro> (4 Arguments)
\dintegrate <macro> (4 Arguments)  où  d = d-isplaystyle
\dintegrate* <macro> (4 Arguments)  où  d = d-isplaystyle
```

— `Argument 1:` ce qui est en bas du symbole $\int_{\bullet}$.

— `Argument 2:` ce qui est en haut du symbole $\int^{\bullet}$.

— `Argument 3:` la fonction intégrée.

— `Argument 4:` suivant quoi on intègre.

8.5.3 Intégration – L'opérateur crochet

```
\hook <macro> [1 Option] (4 Arguments)
\hook* <macro> [1 Option] (4 Arguments)
```

— `Option:` la valeur par défaut est `b`. Voici les différentes valeurs possibles.

1. `b` : des crochets extensibles sont utilisés.
2. `sb` : des crochets non extensibles sont utilisés.
3. `r` : un unique trait vertical extensible est utilisé à droite.
4. `sr` : un unique trait vertical non extensible est utilisé à droite.

— `Argument 1:` ce qui est en bas du crochet fermant.

— `Argument 2:` ce qui est en haut du crochet fermant.

— `Argument 3:` la fonction sur laquelle effectuer le calcul.

— `Argument 4:` la variable pour les calculs.

8. Notant $\Delta = b^2 - 4ac$, si $\Delta < 0$ il n'y aura pas d'argument supplémentaire, si $\Delta = 0$ il y en aura un seul et enfin si $\Delta > 0$ il faudra en donner deux, le 1er étant le plus petit.